Award Period: From 9/17/85 To 1/17/86

Type Agreement: Sub Project Under A-4316

Estimated: $47,317
Funded: $47,317

Cost Sharing Amount: N/A

Title: Sub-Project Under A-4316
A HIERARCHICAL AND INTELLIGENT CONTROLLER FOR THE SUBSCALE THERMAL CONTROL SYSTEM OF THE SPACE STATION

ADMINISTRATIVE DATA

OCA Contact: Ralph Grede X 4820

1) Sponsor Technical Contact:

2) Sponsor Admin/Contractual Matters:

Defense Priority Rating: None Indicated
Military Security Classification: N/A
(or) Company/Industrial Proprietary: None Indicated

RESTRICTIONS

See Attached N/A Supplemental Information Sheet for Additional Requirements.

Travel: Foreign travel must have prior approval — Contact OCA in each case. Domestic travel requires sponsor approval where total will exceed greater of $500 or 125% of approved proposal budget category.

Equipment: Title vests with Georgia Tech — See Agreement A-4316 Paragraph 11.

COMMENTS:

This is a sub-project to A-4316 / Scruggs / ECSL

COPIES TO: Project Director
Research Administrative Network
Research Property Management
Accounting

SPONSOR’S I. D. NO.
Procurement/GTRI Supply Services
Research Security Services
Reports Coordinator (OCA)
Research Communications (2)

GTRC
Library
Project File
Other A. Jones

FORM OCA 65:285
Date: May 14, 1986

Project No.: A-4316

Includes Subproject No.(s): E-21-619

Project Director(s): R. Scruggs

Sponsor: Boeing Aerospace Company

Title: Development of Innovative Technologies for the Space Station Common Module

A Hierarchical and Intelligent Controller for the Subscale Thermal Control System of the Space Station

Effective Completion Date: 1/17/86 (Performance) 1/17/86 (Reports)

Contract Closeout Actions Remaining:

- [X] Final Invoice or Final Fiscal Report

- None

- Closing Documents

- Final Report of Inventions

- Govt. Property Inventory & Related Certificate

- Classified Material Certificate

- Other

Continues Project No. ________________________________ Continued by Project No. ________________________________

IES TO:

- Library
- GTRC
- Research Communications (2)
- Project File
- Other: Jones, Embry

Director
Research Administrative Network
Research Property Management
Accounting
Measurement/EES Supply Services
Research Security Services
Parts Coordinator (OCA)
Library Services

OCA: 60:1028
A HIERARCHICAL AND INTELLIGENT CONTROLLER
FOR THE AEROSPACE THERMAL CONTROL SYSTEM
OF THE SPACE STATION
A HIERARCHICAL AND INTELLIGENT CONTROLLER
FOR THE SUBSCALE THERMAL CONTROL SYSTEM
OF THE SPACE STATION

A Final Report
for Project No. E-21-619

Submitted to
Boeing Aerospace Co.
Space Station Program
499 Boeing Boulevard
Huntsville, AL 35801

Prepared by
Georgia Institute of Technology
Atlanta, Georgia 30332

Authors:
George Vachtsevanos, School of Electrical Engineering
Frank Lewis, School of Electrical Engineering
Kent Davey, School of Electrical Engineering
Henry Hexmoore, Georgia Tech Research Institute
James Cheng, Graduate Research Assistant

November 1986
# Table of Contents

## Part I. The Hierarchical Controller

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I.1 Introduction</td>
<td>I-1</td>
</tr>
<tr>
<td>I.2 System Description</td>
<td>I-5</td>
</tr>
<tr>
<td>I.3 The Control Algorithm</td>
<td>I-6</td>
</tr>
<tr>
<td>I.4 Steady State Analysis and Derivation of the Optimum Control Algorithm</td>
<td>I-7</td>
</tr>
<tr>
<td>A. ECLSS Loop</td>
<td>I-8</td>
</tr>
<tr>
<td>B. Heat Exchanger 2</td>
<td>I-14</td>
</tr>
<tr>
<td>C. Parallel Loop</td>
<td>I-16</td>
</tr>
<tr>
<td>I.5 Simulation Results</td>
<td>I-32</td>
</tr>
<tr>
<td>I.6 Discussion of Results</td>
<td>I-33</td>
</tr>
<tr>
<td>I.7 Solution for Two Heat Sinks</td>
<td>I-55</td>
</tr>
</tbody>
</table>

## Part II. The Intelligent Controller

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>II.1 Introduction</td>
<td>II-1</td>
</tr>
<tr>
<td>II.2 Basic Methodology</td>
<td>II-3</td>
</tr>
<tr>
<td>II.3 Fault Triggering</td>
<td>II-13</td>
</tr>
<tr>
<td>A. Triggering for a Leaky Pipe</td>
<td>II-14</td>
</tr>
<tr>
<td>B. Triggering for a Stuck Valve</td>
<td>II-16</td>
</tr>
<tr>
<td>II.4 Sensor Data Validation</td>
<td>II-19</td>
</tr>
<tr>
<td>A. &quot;Measurement&quot; Variables</td>
<td>II-27</td>
</tr>
<tr>
<td>II.5 Fault Detection</td>
<td>II-45</td>
</tr>
<tr>
<td>II.6 The Rule Base</td>
<td>II-55</td>
</tr>
<tr>
<td>II.7 Fault Identification</td>
<td>II-58</td>
</tr>
<tr>
<td>II.8 Error Trending</td>
<td>II-66</td>
</tr>
<tr>
<td>II.9 Fault Isolation</td>
<td>II-69</td>
</tr>
<tr>
<td>A. Leaky Component Isolation</td>
<td>II-69</td>
</tr>
<tr>
<td>B. Sticking Valve Isolation</td>
<td>II-70</td>
</tr>
<tr>
<td>II.10 Conclusion and Recommendations</td>
<td>II-70</td>
</tr>
</tbody>
</table>

## References

II-73
APPENDIX A: USER DOCUMENTATION, PROGRAM LISTING, AND SIMULATION RESULTS FOR LEAK DETECTION AND IDENTIFICATION SIMULATION STUDIES

APPENDIX B: USER DOCUMENTATION AND PROGRAM LISTING OF HIERARCHICAL CONTROLLER SIMULATION
EXECUTIVE SUMMARY

This report describes the approach, design, and simulation results for the hierarchical and intelligent control algorithms developed for the Space Station Subscale Thermal Control System.

The common module Thermal Control System (TCS) is designed to integrate thermal distribution and thermal control functions. Under "normal" operating conditions, the control problem is to direct the transport fluid flow so as to remove excess heat and maintain an equilibrium such that temperatures at specified points in the network take on prescribed values. Overall thermal control system functions are enhanced if critical thermal loop component faults are detected to the finest practical resolution and isolated, for maintenance and repair purposes, in the shortest possible time.

To accomplish these broad objectives, a table of functional dependence is employed to extract a two-level hierarchical control structure. With this structure, the local input and output variables decouple into separate single-input/single-output systems. At the upper level, a max-min principle is used to design the global controller. This approach assures that the target equilibrium is maintained while the energy required to drive the transport fluid through the thermal network is minimized.

Furthermore, algorithmic approaches are developed to diagnose select fault conditions in the TCS, detect the fault type, identify the location of the faulty component in the thermal loop, perform error trending analysis, and provide appropriate status reports about means for isolating failed components with minimum disruption to the normal TCS functions. The proposed methodology employs analytic redundancy and multivalued ("fuzzy") logic to achieve these objectives.
For concept verification purposes, the initial conceptual design is followed by a sequence of computer simulation studies. The simulation results clearly demonstrate the robustness features of the control algorithms.

The intelligent controller uses hypothetical parameter values and requires extensive data from the subscale TCS laboratory facility for fine-tuning and fuzzy parameter modeling purposes.

For the hierarchical controller, a singular value analysis is called for to provide justification for the decoupling hypotheses made, as well as to enhance the controller robustness features.

This work can form the foundation for the design of the full-scale controller. Within the hierarchical control concept such issues as constrained flow scheduling or balancing may be easily addressed. Within the intelligent controller framework, "smart" sensing and data management aspects may be developed.

We appreciate the opportunity to work with the technical personnel of Boeing Aerospace Company and we are looking forward to continued cooperation in the future.

November 1986
PART I. THE HIERARCHICAL CONTROLLER

I.1 Introduction

The common module Thermal Control System (TCS) is designed to integrate thermal distribution and thermal control functions in order to transport heat and provide environmental temperature control through the common module. The heat rejection and transport component involves the transport of excess heat from all space station elements and attached payloads where appropriate to a radiator system, and the rejection of that heat for thermal control. The system includes the radiators, transport lines, transport fluid, system monitoring, and control hardware/software, pumps, and attached payload cold plates and heat exchangers.

The control problem is to direct the fluid flow so as to remove excess heat and maintain an equilibrium such that the temperatures at certain specified points in the network take on prescribed values. To achieve this objective, the available control inputs are the overall fluid mass flow rate and the relative closures of certain bypass valves. The available outputs consist of flow rates and temperatures measured at several points in the network.

In this report, the TCS model under consideration consists of five heat exchangers, two heat sinks, and seven bypass control valves. A schematic of the system is shown in Figure I-1. Temperatures at two points in the network \((T_{i1}, T_{i2})\) are prescribed and, in addition, five temperature difference limits are imposed. The cold plate/heat exchanger heat flows \(Q_1-Q_5\) are unknown, and so cause five disturbance inputs into the heat exchangers. Due to the algebraic relations between the fluid mass flow rates, the temperatures, and

I-1
FIGURE I-1. SCHEMATIC OF THE STCS.
the heat flow rates (e.g., \( Q = \dot{m}C_v\theta \) with \( Q \) = heat flow, \( \dot{m} \) = fluid mass flow, \( C_v \) = specific heat, \( \theta \) = temperature difference), this is a complex nonlinear multivariable control problem (with unknown disturbances).

The system equations determine the most natural structure for the controller. A "table of functional dependence" is used to extract a two-level hierarchical control structure. With this structure, the three lower level ("local") input and output variables decouple into separate single-input/single-output (SISO) systems. The three upper level ("global") variables then form a three-input/three-output nonlinear system. The hierarchical nature of the control problem is depicted in Figure 1-2.

At the local level, the controllers are designed using the root-locus method, so that their robustness is guaranteed. At the upper level, a quasi-steady state model of the system is used to come up with the "global" controller commands.

One advantage of the two-level controller is that, due to the local action of the lower level controllers, fluid transit delays in the piping network can be ignored. The design can, therefore, be accomplished using state equations and not delay-differential equations. Moreover, the hierarchical control results in a separation of time scales, with faster variables delegated to the lower control level.

Simulation results are included to show the effectiveness of the control scheme.

In actuality, a third and upper level intelligent controller is interacting with the "global" controller. The description of the intelligent controller and its interaction with the global controller is presented in the second part of this report.
DYNAMICS:
Continuous: For classical design. Robust control about set point.

ALGEBRAICS:
Discrete: Period to establish set point for flow rate $m_0$

FIGURE I-2. HIERARCHICAL STRUCTURE OF THE MULTIVARIABLE CONTROLLER.
I.2 System Description

The space station common module subscale thermal system is composed of five heat exchangers (HX-i, i = 1-5) to absorb waste heat dissipated in the system and two heat sinks (the 70° heat sink and the 35° heat sink). These two heat sinks dissipate the waste heat absorbed by the HX's outside of the module.

The algebraic relations describing the absorbed heat and the variables for each HX are:

\[ Q_i = \dot{m}_i C_v \Delta T_i \quad , \quad i = 1-5 \]  

(I.1-5)

where

\[ \dot{m}_i \quad : \quad \text{mass flow-rate through HX-i} \]

\[ \Delta T_i \quad : \quad \text{temperature difference across HX-i} \]

\[ C_v \quad : \quad \text{specific heat constant of water} \]

\[ i=1 \quad : \quad \text{ECLSS HX} \]

\[ i=2 \quad : \quad \text{low } \Delta T \text{ cold plate} \]

\[ i=3 \quad : \quad \text{high } \Delta T \text{ cold plate} \]

\[ i=4 \quad : \quad \text{avionic's HX} \]

\[ i=5 \quad : \quad \text{liquid-liquid HX.} \]

Analogously, the algebraic relations describing the heat sinks are:

\[ Q_{Hsi} = \dot{m}_{Hsi} C_v \Delta T_{si} \quad , \quad i = 1,2 \]  

(I.6-7)

where

\[ \dot{m}_{Hsi} \quad : \quad \text{mass flow-rate through } Hsi \]

\[ \Delta T_{si} \quad : \quad \text{temperature difference across } Hsi \]
i=1 : the 70°F HS
i=2 : the 35°F HS.

The nominal range of temperatures at which the TCS operates in from = 40°F to = 90°F. At this range, \( C_v \) is assumed to be constant. Moreover, it is assumed that water is not compressible.

Along with the above algebraic relations that should be consistent whenever the system is at rest (steady state), there are many imposed constraints that should be satisfied all the time. These constraints are:

- \( C_1 \) : The total generated heat within the system is less than or equal to a certain specified constant, \( Q_s \). In this case, \( Q_s < 10 \text{ kW} \).
  
  That is
  \[
  Q_s = \sum_{i=1}^{5} Q_i \leq 10 \text{ kW}.
  \] (I.8)

- \( C_2 \) : \( \Delta T_1 < 19^\circ F \)  
- \( C_3 \) : \( \Delta T_2 < 5^\circ F \)  
- \( C_4 \) : \( \Delta T_3 < 40^\circ F \)  
- \( C_5 \) : \( \Delta T_4 < 40^\circ F \)  
- \( C_6 \) : \( \Delta T_5 < 40^\circ F \)  
- \( C_7 \) : \( T_{11} = (40 \pm 2.5)^\circ F \)  
- \( C_8 \) : \( T_{12} = (70 \pm 2.5)^\circ F \)  

### I.3 The Control Algorithm

The control problem is to direct the fluid so as to remove excess heat \( Q_i, i = 1-5 \) and maintain an equilibrium such that the constraints \( C_i, i = 1-8 \) are satisfied. To achieve this objective, the available control inputs are the overall fluid mass flow rate and the relative closures of the
bypass valves. The available outputs consist of flow rates and temperatures measured at several points in the network.

In the following section, we will discuss the control strategy used to achieve the above requirements. Moreover, an additional objective of the control design is to minimize the amount of energy needed to control the complex system. In other words, the control strategy is optimum in the sense that the control action uses minimum energy (minimum "energy optimal control" problem). Specifically, since the energy dissipated to control the system is proportional to the mass flow rate times the pressure drop across the pumping system (i.e., \( E = \dot{m}_o \Delta P \)) and the mass flow rate is the "global" control variable, then the optimum control action is the one that minimizes the mass flow rate through the system. Note that minimizing energy corresponds to minimizing power even "better" since power is proportional to the mass flow rate cubed (i.e., \( P = \dot{m}_o^3 \)). Also, as a consequence of minimizing the mass flow rate, the pressure drops across the different elements of the system are automatically minimized which makes the optimization scheme more efficient.

1.4 Steady State Analysis and Derivation of the Optimum Control Algorithm

The optimum control algorithm is based on a max-min approach. Before developing the mathematical details, let us briefly discuss the philosophy behind this optimum control strategy. The idea is very simple and intuitively appealing. The max-min algorithm requires the computation of the minimum mass flow rate demanded by each HX. Next, the maximum mass flow rate out of the minimum set is chosen as the optimum "global" mass flow rate, (i.e., \( \dot{m}_o \)), hence, the max-min designation. Thus, the HX requiring the largest mass flow rate is satisfied while using the minimum flow rate needed by that HX. For
the remaining HX's, the excess of mass flow rate is taken care of by using the appropriate bypass valves. The optimality (I.8), as defined earlier, of this max-min algorithm is straightforward.

A. **ECLSS Loop** (Fig. I-3)

For the ECLSS loop, application of mass and energy conservation principles lead to the following analysis:

**HS-2**

\[
q_{s2} = \frac{\Delta Q_{s2}}{C_v} = m_s (T_s - T_i1) \tag{I.16}
\]

\[
= (1-\alpha)(T_s - T_i1)m_o \tag{I.17}
\]

**HX-1**

\[
q_1 = \frac{\Delta Q_1}{C_v} = m_1 (T_{o1} - T_i1) \tag{I.18a}
\]

\[
= (1-\alpha)(1-\delta)(T_{o1} - T_i1)m_o \tag{I.18b}
\]

**MIX**

\[
T_{o11} = \delta T_i1 + (1-\delta)T_{o1} \tag{I.18c}
\]

\[
T_{i2} = aT_s + (1-\alpha)T_{o11} \tag{I.19}
\]

\[
= aT_s + \delta(1-\alpha)T_i1 + (1-\delta)(1-\alpha)T_{o1} \tag{I.19}
\]

\[
(17) \rightarrow T_s = \frac{q_{s2}}{m_o(1-\alpha)} + T_i1 \tag{I.20}
\]

\[
(1-\delta)(1-\alpha)T_i1 = (1-\alpha)T_i1 - \delta(1-\alpha)T_i1
\]

I-8
FIGURE I-3. ECLSS LOOP.
\[ + \delta (1-a)T_{i1} = (1-a)T_{i1} - (1-\delta)(1-a)T_{i1} \quad (I.21) \]

Substituting (I.20) and (I.21) into (I.19):

\[
T_{i2} = a \left( \frac{q_{s2}}{m_o (1-a)} + T_{i1} \right) + (1-a)T_{i1} - (1-\delta)(1-a)T_{i1} + (1-\delta)(1-a)T_{o1} \\
= \frac{q_{s2}}{m_o} \frac{\alpha}{1-a} + aT_{i1} + (1-a)T_{i1} + (1-\delta)(1-a)(T_{o1} - T_{i1}) \\
= \frac{q_{s2}}{m_o} \frac{\alpha}{1-a} + T_{i1} + \frac{q_1}{m_o} \\
(17) + (1-a)(1-\delta)(T_{o1} - T_{i1}) = \frac{q_1}{m_o} \quad (I.22) \]

Solving for \( m_o \) from (I.22):

\[
T_{i2} - T_{i1} = \frac{q_{s2}}{m_o} \frac{\alpha}{1-a} + \frac{q_1}{m_o} \\
+ m_o = \frac{q_1}{T_{i2} - T_{i1}} \frac{1-a}{m_o} q_{s2} , \text{ where } T_{i2} = TC9, T_{i1} = TC6 \quad (I.23) \]

Also

\[
m_o > \frac{q_1/\Delta T_{im}}{(1-a)(1-\delta)} , \text{ where } \Delta T_{im} = 19^\circ F \quad (I.24) \]

Let

\[
a \Delta T_{i2} - T_{i1} \Delta 30^\circ F \quad (I.25) \]

\[
b \Delta T_{im} = 19^\circ F \quad (I.26) \]

Then

\[ (23) + \frac{m_o}{a} = (q_1 + \frac{\alpha}{1-a} q_{s2})/a \quad (I.27) \]
In (1.28), set $\delta = 0$ for minimum flow through the ECLSS loop.

\[ (24) + \frac{m_o}{q_{1/b}} > \frac{q_{1/b}}{(1-a)(1-\delta)} \]  

(1.28)

In order to arrive at the minimum flow through HX1, we choose (or require that)

\[ m_o = \frac{q_{1/b}}{(1-a)} \]  

(1.29a)

Now, solving (1.27) and (1.29b) simultaneously leads to a minimum required.

The final result is

\[ (m_o, a) \text{ optimum} = (m_o \min, a \min) \]

or

\[ (q_1 + \frac{a}{1-a} q_{s2})/a = \frac{q_{1/b}}{(1-a)} \]

\[ \frac{q_1}{a} (1-a) + \frac{aq_{s2}}{a} = \frac{q_1}{b} \]

\[ a (\frac{q_{s2}}{a} - \frac{q_1}{a}) = \frac{q_1}{b} - \frac{q_1}{a} \]

\[ (a)_{\min} = \frac{q_1 (\frac{1}{b} - \frac{1}{a})}{(q_{s2} - q_1)/a} = \frac{q_1 (a-b)}{(q_{s2} - q_1)/a} = \frac{q_1}{q_{s2} - q_1} (a-b) \]  

(1.30)

Substituting (1.30) into (1.29b) yields:
\[
(m_o)_{\text{min}} = \frac{q_1/b}{(1-a_{\text{min}})} = \frac{q_1}{b} \cdot \frac{1}{1 - \frac{q_1}{q_{s2} - q_1} (a-b)}
\]

\[
= \frac{q_1}{b} \cdot \frac{1}{(q_{s2} - q_1)b - q_1(a-b)} = \frac{q_1(q_{s2} - q_1)}{bq_{s2} - bq_1 - aq_1 + bq_1}
\]

\[
(m_o)_{\text{min,ECLSS}} = \frac{q_1(q_{s2} - q_1)}{bq_{s2} - aq_1}
\]

(always > 0 \( \forall q_{s2}/q_1 \) since by conversion \( q_1 > 0 \) and \( q_{s2} > 0 \)). Equation (I.31) is the minimum flow rate required for the ECLSS loop while assuring heat balance in the steady state mode. Since \((m_o)_{\text{min}}\) is a positive number, this implies

\[
\frac{q_1(q_{s2} - q_1)}{bq_{s2} - aq_1}.
\]

For Eq. (I.32) to be valid, we require:

(1) \[ q_{s2} - q_1 > 0 \quad \& \quad bq_{s2} - aq_1 > 0 \quad \text{(I.33)}\]

or

(2) \[ q_{s2} - q_1 < 0 \quad \& \quad bq_{s2} - aq_1 < 0 \quad \text{(I.34)}\]

where \( q_1, q_{s2} \) \( a \) and \( b \) are all positive.

\[
(1) \quad q_{s2} > q_1 \quad \& \quad q_{s2} > \frac{a}{b} q_1 = \frac{30}{1.5} q_1 = 1.58 q_1
\]

\[
\quad + q_{s2} > 1.58 q_1 \quad \text{(I.35)}
\]
Obviously, if (1.36) holds, the condition $\Delta T_{im} = 19^\circ$F will be violated. Therefore, we adopt Eq. (1.35). That is

$$q_{s2} > 1.58 q_1$$

where as stated above, $q_{s2} = \frac{Q_{s2}}{C_v}$, with $Q_{s2}$ the heat dissipated by heat sink 2 (HS-2).

Because of the overall heat sink balance (stated earlier in this section)

$$Q_s = \sum_{i=1}^{5} Q_i = Q_{s1} + Q_{s2}$$

where $Q_{s1}$ is the heat dissipated by heat sink 1 (HS-1). Consequently,

$$Q_s > 1.58 Q_1 + Q_{s1}$$

This inequality is very significant when considering the operating range of the system. It is obvious that we would like the system to work properly for any disturbance combinations. But as seen from Eq. (1.35), the system is constrained to operate within the range specified by this inequality.
With the underlying goal of realizing a wide operating range, it is obvious from Eq. (1.37) that $Q_{s1}$ should be kept as small as possible. The extreme case is to let $Q_{s1} = 0$ implying that only one heat sink, more precisely HS2 (the 35°F heat sink), may dissipate effectively the load. This configuration will lead to:

$$Q_{s2} = Q_s$$

which results again in

$$q_s > 1.58 q_1$$  \hspace{1cm} (I.38)

**B. Heat Exchanger 2** (see Figure I-4)

The constraint imposed on HX-2 is expressed as

$$\dot{m}_o > \frac{Q_2}{C_v (1-\beta) \Delta T_{2m}}$$

with

$$q_2 = \frac{\Delta Q_2}{C_v}$$

results in

$$\dot{m}_o > \frac{q_2}{(1-\beta) \Delta T_{2m}}$$

where

$$\Delta T_{2m} = 5°F$$

Following the same minimum flow policy through the system, we require a minimum flow through HX-2. This is achieved by making $\beta = 0$, that is

$$\dot{m}_{o, \text{min}} \bigg|_{HX-2} = \frac{q_2}{\Delta T_{2m}}$$  \hspace{1cm} (I.39)
FIGURE I-4. HEAT EXCHANGER-2.
C. **Parallel Loop** (see Figure I-5)

In order to determine the distribution of $\dot{m}_o$ among the three parallel HX's, the flow pressure characteristic of each HX belonging to the parallel loop is required. This derivation is based on experimental data collected for each HX in the parallel loop. The data was fitted to a second order polynomial in $\dot{m}_i$, $i = 1-3$. The result is

$$\Delta P_3 = 1 \times 10^{-5} \dot{m}_i^2 + 1.67 \times 10^{-3} \dot{m}_i$$  \hspace{1cm} (I.40)

$$\Delta P_4 = 2.48 \times 10^{-4} \dot{m}_i^2$$ \hspace{1cm} (I.41)

$$\Delta P_5 = 2.48 \times 10^{-5} \dot{m}_i^2$$ \hspace{1cm} (I.42)

where $\Delta P_i$, $i = 3, 5$ is the pressure drop across HX-$i$.

Assuming that pressure drop due to the piping is negligible, we can write

$$\Delta P_3 = \Delta P_4 = \Delta P_5 = \Delta P$$  \hspace{1cm} (I.43)

Now, let us determine the minimum flow required by each HX while satisfying constraints C4, C5, and C6 (i.e., Eqs. (I.11), (I.12), and (I.13)). Rewriting Eqs. (I.3), (I.4), and (I.5) yields

$$\dot{m}_i = \frac{Q_3}{C_v \Delta T_{i, m}} \Delta r_i \hspace{1cm} i = 1-3$$ \hspace{1cm} (I.44-46)

where

$$\Delta T_{i, m} = 40^\circ F \hspace{1cm} i = 1-3$$
FIGURE I-5. PARALLEL LOOP.
The $\Delta T$'s are the maximum temperature differences allowed across the parallel loop and the three HX's, respectively. The mass flow rate $m_0$ is distributed among the three HX's, HX-1, HX-2, and HX-3, according to the respective values of $r_1$, $r_2$, and $r_3$. The corresponding pressure differences developed across each HX are

$$\Delta P'_3 = (1.67 \times 10^{-3}) r_3 + (1 \times 10^{-5}) r_2^2 \quad (1.47)$$

$$\Delta P'_4 = (2.48 \times 10^{-4}) r_4^2 \quad (1.48)$$

$$\Delta P'_5 = (2.48 \times 10^{-4}) r_5^2 \quad (1.49)$$

Because of the parallel configuration of the three HX's, the pressure drop across the parallel loop is given by

$$\Delta P = \max(\Delta P'_3, \Delta P'_4, \Delta P'_5)$$

Now since $\Delta P$ is known, it is possible to compute the mass flow rate through each HX-1. This is accomplished by using Eqs. (1.40)-(1.42). The final result is

$$m_3 = \frac{-67 + \sqrt{(167)^2 + 4 \times 10^5 \Delta P}}{2} \quad (1.50)$$

$$m_4 = \sqrt{\Delta P/2.48 \times 10^{-4}} \quad (1.51)$$

$$m_5 = \sqrt{\Delta P/2.48 \times 10^{-4}} \quad (1.52)$$

As is usually the case, the minimum flow through the parallel loop is given by
\[(m_o)_{\text{min loop}} = \sum_{i=3}^{5} m_i = m_3 + m_4 + m_5 \quad (I.53)\]

It is interesting to see how the total flow \(m_o\) divides among the three heat exchangers. Let us define the ratios:

\[C_i = \frac{\Delta m_i}{\sum_{i=3}^{5} m_i}, \quad i = 3-5 \quad (I.54-56)\]

and

\[\gamma = \frac{m_3}{m_o} + \frac{m_4}{m_o} = \frac{m_o}{m_o} \gamma \quad (I.57)\]

Adding Eqs. (I.54-56) leads to

\[C_3 + C_4 + C_5 = 1 \quad (I.58)\]

We define

\[\text{RP} = \sum_{i=3}^{5} m_i = m_3 + m_4 + m_5 \quad (I.59)\]

as being the minimum mass flow rate required by the parallel loop. It is also true that

\[\text{RP} = m_o (1-\gamma) \quad (I.60)\]

therefore,

\[m_3 = C_3 \text{RP} = C_3 m_o (1-\gamma) \quad (I.61)\]

\[m_4 = C_4 \text{RP} = C_4 m_o (1-\gamma) \quad (I.62)\]

\[m_5 = C_5 \text{RP} = C_5 m_o (1-\gamma) \quad (I.63)\]
Using Eqs. (1.40-42), it is deduced that

\[ \Delta P = 1.67 \times 10^{-3}(C_3 \cdot m_0 (1-\gamma)) + 1 \times 10^{-5}(C_3 \cdot m_0 (1-\gamma))^2, \tag{1.64} \]

\[ \Delta P = 2.48 \times 10^{-4}(C_4 \cdot m_0 (1-\gamma))^2, \tag{1.65} \]

and

\[ \Delta P = 2.48 \times 10^{-4}(C_5 \cdot m_0 (1-\gamma))^2. \tag{1.66} \]

Now solving for \( C_3, C_4, \) and \( C_5 \) by using the above Eqs. (1.64-66), Eq. (1.64) yields:

\[ C_3 = \frac{-b + \sqrt{b^2 + 4a\Delta P}}{2(1-\gamma)m_0 a} \tag{1.67} \]

with

\[ a = 1 \times 10^{-5}, \quad b = 1.67 \times 10^{-3} \]

Eq. (1.65) yields

\[ C_4 = \frac{\sqrt{\Delta P}}{(1-\gamma)m_0 \sqrt{P_4}} \tag{1.68} \]

with

\[ P_4 = 2.48 \times 10^{-4} \]

and finally, Eq. (1.66) gives

\[ C_5 = \frac{\sqrt{\Delta P}}{(1-\gamma)m_0 \sqrt{P_5}} \tag{1.69} \]

with

\[ P_5 = 2.48 \times 10^{-4} \]

Eq. (1.58) implies that

\[ C_3 + C_4 + C_5 = 1, \]

that is, by using Eqs. (1.67-69)
\[
\frac{-b + \sqrt{b^2 + 4a\Delta P}}{(1-\gamma)m_o \cdot (2a)} + \frac{\sqrt{\Delta P}}{(1-\gamma)m_o \sqrt{P_4}} + \frac{\sqrt{\Delta P}}{(1-\gamma)m_o \sqrt{P_5}} = 1
\]

This leads to the following result

\[
(1-\gamma)m_o = \frac{-b + \sqrt{b^2 + 4a\Delta P}}{2a} + \frac{\sqrt{\Delta P}}{\sqrt{P_4}} + \frac{\sqrt{\Delta P}}{\sqrt{P_5}}
\]

(I.70)

Substituting Eq. (I.70) into Eq. (I.67) and solving for \(C_3\) yields

\[
C_3 = \frac{-b + \sqrt{b^2 + 4a\Delta P}}{2 \left[ -b + \sqrt{b^2 + 4a\Delta P} + \frac{\sqrt{\Delta P}}{\sqrt{P_4}} + \frac{\sqrt{\Delta P}}{\sqrt{P_5}} \right]}
\]

\[
= \frac{\left( \frac{-167}{\Delta P} + \sqrt{\frac{(2798a)}{\Delta P}} + 4 \times 10^5 \right)}{\frac{(-167)}{\Delta P} + \sqrt{\frac{(2798a)}{\Delta P}} + 4 \times 10^5 + 50.95}
\]

(I.71)

Again, substituting Eq. (I.70) into Eq. (I.68) and solving for \(C_4\) yields

\[
C_4 = \frac{\sqrt{\Delta P}}{\left( -b + \sqrt{b^2 + 4a\Delta P} + \frac{\sqrt{\Delta P}}{\sqrt{P_4}} + \frac{\sqrt{\Delta P}}{\sqrt{P_5}} \right) \sqrt{P_4} \sqrt{P_5}} = \frac{1}{-83.5 + \sqrt{\frac{6972}{\Delta P}} + 1 \times 10^5 + 127.4}
\]

(I.72)

Finally, substituting Eq. (I.70) into Eq. (I.69) and solving for \(C_5\) leads to

\[
C_5 = \frac{\sqrt{\Delta P}}{\left( -b + \sqrt{b^2 + 4a\Delta P} + \frac{\sqrt{\Delta P}}{\sqrt{P_4}} + \frac{\sqrt{\Delta P}}{\sqrt{P_5}} \right) \sqrt{P_5}} = \frac{1}{-83.5 + \sqrt{\frac{6972}{\Delta P}} + 1 \times 10^5 + 127.4}
\]

(I.73)
It is obvious from the above results that the C's are functions of $\Delta P$. That is, these ratios will vary with the pressure drop variation which itself varies due to variations in $m_o$.

We are now ready to use the above results, mainly the minimum flows needed for HX-1, HX-2, and the parallel loop, to synthesize an elegant control algorithm that will make the system behave in an optimal way while satisfying all the imposed constraints. It is appropriate to state the control problem and its objectives in a concise manner.

**Control Problem:**

Given the algebraic equation:

$$T_{i2} - T_{i1} = \frac{q_1 + \frac{\alpha}{1-\alpha} q_2}{m_o}$$  \hspace{1cm} \text{(I.74)}

subject to the algebraic inequalities (constraints):

\begin{align*}
\text{C}_1 & \quad \frac{m}{m_o} > \frac{q_1}{(1-\alpha)(1-\beta)\Delta T_{1M}} = \frac{r_1}{(1-\alpha)(1-\beta)} \\
\text{C}_2 & \quad \frac{m}{m_o} > \frac{q_2}{(1-\beta)\Delta T_{2M}} = \frac{r_2}{1-\beta} \\
\text{C}_3 & \quad \frac{m}{m_o} > \frac{q_3}{C_3(1-\gamma)\Delta T_{3M}} = \frac{r_3}{1-\gamma} \\
\text{C}_4 & \quad \frac{m}{m_o} > \frac{q_4}{C_4(1-\gamma)\Delta T_{4M}} = \frac{r_4}{1-\gamma} \\
\text{C}_5 & \quad \frac{m}{m_o} > \frac{q_5}{C_5(1-\gamma)\Delta T_{5M}} = \frac{r_5}{1-\gamma}
\end{align*}

\hspace{1cm} \text{(I.75-I.79)}
Select the values of \( m_0 \) and the value relative closures so that minimum energy is required for the control action while all constraints are maintained. Recall that the values of the valve openings \( \mu, \tau, \delta, \beta, \) and \( \gamma \) are controlled via local, classical robust continuous controllers. We will describe these controllers in the following sections of this report. The only command generated by the higher level controller is the mass flow rate \( m_0 \). The following paragraph explains, in detail, the different steps involved in the control algorithm.

**Selection Algorithm:**

First, define

\[
\rho = \frac{q_1}{T_{i2} - T_{i1}}
\]

then \( m_0 \) is selected using the following procedure

\[ m_0 = \max \left( r_0, \frac{r_1}{1-a}, r_2, r_3, r_5 \right) \]

Second, \( \tau \) is picked to maximize \( T_s \). This corresponds to making \( \tau \) as large as possible. Such a control action assists in achieving the set point

\[ T_{i2} = (1-a)T_{o11} + aT_s = 70°F \]

with a smaller \( a \) so that \( C_1 \) is kept small, resulting in a smaller \( m_0 \). It is worth noting that \( T_{o11} < T_{i1} + \Delta T_1 \) and that \( T_s > 70 \) all the time. Before presenting the simulation results, we must comment on the local controllers and the design philosophy pursued. Recall that the \( m_0 \) selection algorithm is the upper level controller in the control hierarchy. This natural hierarchy was deduced from the following functional dependence table.
To control the various valve openings, appropriate commands for every valve are required. These command signals are:

\[ \mu^c = a^c = \frac{^c \dot{m} - ^c m}{^c m_o} \]

where

\[ ^c m_s = \frac{q_s}{T_o - T_{i1}^c} \]

and

\[ T_o^c = T_{i2}^c + (q_s - q_1) / ^c m_o \]

\[ \beta^c = 1 - \frac{Q_2}{^c m_o v \Delta T_{2m}} \]
Similar expressions are derived for the \( \delta^C \) and \( \gamma^C \) command signals. The following pages present the design procedure for the continuous feedback control of valve settings.

The simulation package uses models of the actuator/valve motor/effective heat and fluid temperature dynamics.

Simple linear representations are adopted in these cases. The representations depict accurately the overall operational characteristics of the system. The actuator dynamics are presented first in the next pages followed by the valve motor description and the dynamic model of the effective heat dissipated by the system due to the heat disturbance inputs; fluid temperature dynamics are modeled last as they are used for simulation purposes.
LOCAL CONTROLLERS

VALVES

\[
\Delta T_2 \quad 1 - \frac{m_2 T_2}{m_0 T_{2m}} \quad \beta_c^e \quad e \quad kH(s) \quad V_B \quad G(s) \quad \beta
\]

\[
G(s) = \frac{km}{s(1 + s T_m)}
\]

\[
H(s) = \frac{1 + a\tau}{1 + s}, \quad a > 1 \text{ (phase lead)}
\]

ROOT-LOCUS
\[ G(s) = \frac{kp}{1 + stp} \]

\[ H(s) = \frac{1 + st}{s} \]
ACTUATOR DYNAMICS

I - PUMP

TIME-DOMAIN DESCRIPTION

\[
\frac{d}{dt} \dot{m}_o = \frac{1}{\tau} (-\dot{m}_o + k \nu \dot{m}_o), \quad \tau = 1 \text{ sec}
\]

S-DOMAIN DESCRIPTION

\[
\nu \dot{m}_o \rightarrow H(s) = \frac{k}{1 + s \tau} \rightarrow \dot{m}_o, \quad \tau = 1 \text{ sec}
\]

STEP-INPUT RESPONSE

\[
\nu \rightarrow \dot{m}_o \rightarrow t
\]
II - VALVE MOTORS

TIME-DOMAIN DESCRIPTION

\[ \frac{d}{dt} \mu = \mu \]

\[ \frac{d}{dt} \mu = \frac{1}{\tau} (\mu + k\nu), \tau = 0.5 \text{ sec} \]

S-DOMAIN DESCRIPTION

\[ H(s) = \frac{k}{s(1 + \tau s)} \]

RAMP-INPUT RESPONSE
EFFECTIVE HEAT DYNAMICS

TIME-DOMAIN DESCRIPTION

\[ \dot{Q}_i = \frac{1}{\tau_i} ( -Q_i + Q_{\xi_1} ) , \quad i = 1-5 \]

S-DOMAIN DESCRIPTION

\[ H(s) = \frac{1}{1 + s\tau_i} \]

STEP-INPUT RESPONSE

![Graph showing step-input response](image)
FLUID TEMPERATURE DYNAMICS

TIME-DOMAIN DESCRIPTION

\[ T_o = \frac{1}{C_v M} \left( \sum_{i=1}^{5} Q_i - Q_s \right) \]

S-DOMAIN DESCRIPTION

\[ \sum_{i=1}^{5} \frac{Q_i}{C_v M} \rightarrow \frac{1}{C_v M} \rightarrow \frac{1}{S} \rightarrow T_o \]

\[ T_o \rightarrow t \]

\( \alpha \) net heat flow
1.5 Simulation Results

Three test scenarios were implemented in this study. Test curves are generated for each case showing the variation of mass flow rate \( \dot{m}_0 \), temperatures \( T_{i2}, T_{i1} \), and various other parameter variations as several control strategies are pursued. The curves are plotted versus time (in minutes) and are grouped into three cases as follows:

(1) **Case 1:** External heat source \( Q_{1_{\text{ext}}} \) has been on for a long time and is turned off at time \( t = 0 \); this change translates into the following exponential change for \( Q_1 \)

\[
Q_1 = Q_{1_{\text{ext}}} \exp(-t/\tau)
\]

where \( \tau \) is the characteristic time constant for the ECLSS system (~8 minutes). The results are plotted for:

(a) Open loop controller - no control update, and

(b) Closed loop performance.

(2) **Case 2:** The analogous problem is examined here for heat source \( 2_{\text{ext}} \) \( (Q_{2_{\text{ext}}}) \). At \( t = 0 \), it is turned off, forcing \( Q_2 \) into an exponential decay

\[
Q_2 = Q_{2_{\text{ext}}} \exp(-t/\tau)
\]

Again, results are plotted for:

(a) Open loop, and

(b) Closed loop performance.

(3) **Case 3:** In the third case, we add the second heat sink and compare the results to the corresponding test (Case 2) cited above. Results are given for two cases:
(a) \( \tau = 1 \); the second \( (\tau) \) heat sink is completely bypassed.
(b) \( \tau = 0 \); the second \( (\tau) \) heat sink is in the loop without a bypass.

1.6 Discussion of Results

In the first case (part 1) with no control updates, we observe, as expected, a fall off in the temperature quantities \( \Delta T_i = T_{i2} - T_{i1} \) and \( T_{i2} \). This drop off is corrected by the closed loop controller which is seen to hold a tight tolerance on these quantities. The same simulation trends are noted in Case 2 as \( Q_2 \) is allowed to vary.

In Case 3, a trend is observed which places in question the usefulness of the second heat sink. When the second heat sink is in the loop, the flow rate \( m_o \) grows towards a large value. Except in the case where \( Q_1 \) is large compared to the sum of all the other \( Q \)'s, its presence causes the system to be less efficient. A low pass filter was added to the command control on \( m_o \) in this case; the response is much smoother in the presence of the low-pass filter. It was necessary to force the time constant \( \tau_2 \) on heat sink \( Q_2 \) to be smaller (than Cases 1 and 2) to allow steady state conditions to be reached.

Generally, the present version of the hierarchical controller provides satisfactory results. Further improvements will be included in the final version.
CASE 1(a)

Q_{ext} = 0

Open Loop
$Q_1$ \hspace{1cm} $\tau = 30 \text{ min} \hspace{1cm}$ Open-Loop

$Q_S$

I-35
CASE 1(b)

Q_{1,ext} = 0

Closed Loop
Closed-Loop, $T_c = 5$ sec

$Q_s$

$T_{12}$

$T_{11}$

$T_{12} - T_{11}$

$\Delta T_3 - \Delta T_5$

$\Delta T_1$

$\Delta T_2$
CASE 2(a)

\[ Q_{2 \text{ext}} = 0 \]

Open Loop
\( \sigma \) constraint

37.800

37.600

37.400

0.000   2.000   4.000   6.000   8.000
CASE 2(b)

$O_{2\text{ext}} = 0$

Closed Loop
Closed-Loop, $T_c = 5$ sec

$\Omega_s$

$\varphi_{2,ext}$

$\Delta T_2$

$\Delta T_1$

$\Delta T_{3-5}$

$T_{i1}$

$T_{i2 - T_{i1}}$
CASE 3(a)

\[ \tau = 1 \]
\[ \tau = 1 \]

- \( Q_s \)
- \( T_o \)
- \( m_o \)
- \( u \)
- \( \sigma \)
- \( \delta \)
- \( \beta \)
- \( \gamma \)
CASE 3(b)

\[ \tau = 0 \]
I.7 Solution for Two Heat Sinks

In the actual STCS, system waste heat will be dissipated by two heat sinks. The maximum capacity of each one of the two heat sinks is 5.5 kW. Under minimum energy criteria and the physical constraints imposed by the heat sinks, the following suboptimal two heat sink solution is developed:

\[ Q_s > 1.58 Q_1 \]  
\[ Q_s > 1.58 Q_1 + Q_{S1} \]  

Equations (I.35) and (I.37) are the natural constraints of the system. If \( Q_{S1} = 0 \), then \( Q_s > 1.58 Q_1 \). This is the lowest limit under which the ECLSS loop can achieve heat balance. If the system can operate at \( Q_{S1} = 0 \), then this condition will provide greater flexibility. As the maximum capacities of the two heat sinks are less than 5.5 kW, then

\[ Q_{S1} < 5.5 \text{ kW} \]  
\[ Q_{S2} < 5.5 \text{ kW} \]  

Under the condition of (I.4), the best procedure is to keep the temperature, \( T_s \), as high as possible. The highest temperature achievable for \( T_s \) is \( T_o \). When \( Q_{S1} = 0 \), then \( r = 1 \) and \( T_s = T_o \). But under the constraints (I.80), (I.81) and for easy control of the Tau valve, the control algorithm for the two heat sink operation is as follows:

(1) System is at a low load case (\( Q_s < 5.5 \text{ kW} \)). Let the Tau valve be fully open. It implies that all the flow is bypassed across heat sink #1 (70° heat sink).
\[ Q_{S1} = 0 \]
\[ Q_S = Q_{S1} + Q_{S2} = 0 + Q_{S2} = Q_{S2} \]
\[ T_S = T_o \text{ (best criteria)} \]

For low load case:

\[ Q_1 < 2.45 \text{ kW} \]
\[ Q_S = Q_{S2} > 1.58 Q_1 \]

When \( Q_1 = 2.45 \text{ kW} \), then \( Q_S = Q_{S2} > 3.87 \text{ kW} \).

At this state, the ECLSS loop can achieve a heat balance easier than when \( Q_{S1} \neq 0 \).

(2) System is at a high load case (\( Q_S > 5.5 \text{ kW} \)). When the system load is higher than 5.5 kW, heat sink \#2 is not sufficient to dissipate the total load. Let heat sink \#1 be activated to dissipate part of the heat. As the Tau valve is a thermal static control valve, it is desirable to set the temperature, \( T_S \), at a fixed level; then the Tau valve will act as a stable, local, and robust thermal static control valve.

\[ 0 < Q_{S1} < 5.5 \text{ kW} \]
\[ 0 < Q_{S2} < 5.5 \text{ kW} \]
\[ Q_S = Q_{S1} + Q_{S2} \]

\( T_S \) is fixed to the highest possible temperature that can accommodate any allowable combination of input \( Q_1 \)'s and \( Q_{S2} < 5.5 \text{ kW} \).

Combining (1) and (2), there are two operating states for heat sink number 1.

A. 1st State

\[ Q_S < 5.5 \text{ kW} \]
\[ \tau = 1 \]
\( T_s = T_o \)
\( Q_{S1} = 0 \)
\( Q_{S2} = Q_S \)

**B. 2nd State**

\( Q_S > 5.5 \text{ kW} \)
\( \tau = 1 - \frac{m_{s1}}{m_o} \)
\( T_s = \text{fixed to some temperature} \)
\( Q_{S1} = m_{s1} \cdot cv \cdot (T_o - T_s) \)
\( Q_{S2} = m_{s2} \cdot cv \cdot (T_s - T_{I1}) \)
\( Q_S = Q_{S1} + Q_{S2} \)

where

- \( m_o = \text{optimal flow rate for the system} \)
- \( m_{s1} = \text{flow rate through heat sink } 1 \)
- \( m_{s2} = \text{flow rate through heat sink } 2 \)
- \( \tau = \text{control parameter for control valve Tau} \)

**C. Find T_s**

For State 2, the fixed value of \( T_s \) is calculated under the following constraints:

- \( 5.5 \text{ kW} < Q_S < 10 \text{ kW} \)
- \( Q_{S2} < 5.5 \text{ kW} \)
- \( Q_{S2} > 1.58 Q_1 \)

Using the maxmin algorithm and under the above three constraints, the maximum allowable temperature for every possible combination of input Q's is found. First set \( Q_{S2} = 5.5 \text{ kW} \), under the system constraints (I.8)-(I.15), solve for the steady state solution of \( T_s \). Then the value of \( T_s \) is chosen as the one which is the maximum of
all possible answers. $T_g$ was found to be $77.5^\circ F$. It is thus
guaranteed that at any possible state (different heat input),
$Q_{S2} < 5.5$ kW.

Now the two level hierarchical control structure consists of
two upper level ("global") variables and six lower level ("local")
variables. The hierarchical nature of the control problem is
depicted in Figures I-6 and I-7.

D. Simulation Results

Two test scenarios were implemented in this study. Test curves are
generated for each case showing the variation of mass flow rate $m_o$,
temperatures $T_i$, $T_{il}$, and various other parameter variations as a
minimum energy consumption strategy is pursued. The curves are
plotted versus time (in minutes) and are grouped into two cases as
follows:

1. **Case 1:** External heat source $Q_{i}^{\text{ext}}$ at minimum load has been
   on for a long time and is reduced to the low load $Q_{i}^{\text{ext}}$ at
time $t = 0$; this change translates into the following exponen-
tial change for $Q_{i}$

$$Q_{i} = Q_{i}^{\text{ext f}} + (Q_{i}^{\text{ext i}} - Q_{i}^{\text{ext f}}) \exp \left(-t/\tau_{i}\right), \quad i = 1,2,3,4,5$$

where $\tau_{i}$ is the characteristic time constant for heat
exchanger $i$.

- $\tau_{1}$: time constant for the ECLSS Hx $\sim 30$ minutes
- $\tau_{2}$: time constant for low $\Delta T$ CP $\sim 2$ minutes
- $\tau_{3}$: time constant for high $\Delta T$ CP $\sim 8$ minutes
INTELLIGENT CONTROL
HIERARCHICAL CONTROL

INTELLIGENT CONTROL
FAULT DETECTION
AND ISOLATION

NONLINEAR
ALGEBRAIC
SOLUTION

ALGEBRAICS:
DISCRETE:
PERIOD $T_c$
ESTABLISHES SET
POINT FOR FLOW RATE $\dot{m}_o$

FIGURE I-6. HIERARCHICAL STRUCTURE OF MULTIVARIABLE CONTROLLER
FIGURE I-7. LOCAL STRUCTURE OF CONTROLLER
\[ \tau_4 : \text{time constant for high } \Delta T \text{ CP} \quad \sim 8 \text{ minutes} \]
\[ \tau_5 : \text{time constant for high } \Delta T \text{ CP} \quad \sim 8 \text{ minutes} \]

\[ Q_{1\text{ ext i}} = 2.45 \text{ kW} \quad Q_{1\text{ ext f}} = 0.5 \text{ kW} \]

\[ Q_{2\text{ ext i}} = 2.0 \text{ kW} \quad Q_{2\text{ ext f}} = 0.5 \text{ kW} \]

\[ Q_{3\text{ ext i}} = 2.8 \text{ kW} \quad Q_{3\text{ ext f}} = 0.5 \text{ kW} \]

\[ Q_{4\text{ ext i}} = 1.4 \text{ kW} \quad Q_{4\text{ ext f}} = 0.5 \text{ kW} \]

\[ Q_{5\text{ ext i}} = 1.4 \text{ kW} \quad Q_{5\text{ ext f}} = 0.5 \text{ kW} \]

This case is referred to as the "Hot Start" case, as the initial state is at full load corresponding to the highest temperature the system can achieve. The results are plotted for closed loop performance.

2. **Case 2**: The analogous problem is examined here for the system starting at a low load condition at time \( t = 0 \), then heating up to maximum load. This change translates into the following exponential change for \( Q_i \).

\[ Q_i = Q_{i\text{ ext f}} + (Q_{i\text{ ext i}} - Q_{i\text{ ext f}}) \exp (-t/\tau_i) \quad , \quad i = 1,2,3,4,5 \]

where \( \tau_i \) is the same as in Case 1.

\[ Q_{1\text{ ext i}} = 0.5 \text{ kW} \quad Q_{1\text{ ext f}} = 2.45 \text{ kW} \]

\[ Q_{2\text{ ext i}} = 0.5 \text{ kW} \quad Q_{2\text{ ext f}} = 2.0 \text{ kW} \]

\[ Q_{3\text{ ext i}} = 0.5 \text{ kW} \quad Q_{3\text{ ext f}} = 2.8 \text{ kW} \]
This case is called "Cold Start." As the initial state is at a low load, the results are plotted for closed loop performance.

E. Discussion of Results

Under two extreme conditions ("hot" and "cold" start), the controller performs very satisfactorily. In both cases, a stable steady-state is reached. Because of the impulsive nature of the switching action, some oscillations are observed around the switching point. In the simulation studies, these oscillations are reduced considerably by the introduction of low-pass filtering or micro-stepping techniques. The micro-stepping technique seems to be effective in smoothing the controller response. It is implemented by partitioning the switching impulse into 200 small steps which excite the system smoothly.

Generally, the two-step tau valve controller provides reasonable results. The system waste heat can thus be dissipated by two small capacity heat sinks.
CASE 1

Hot Start Case

Closed Loop
HOTST RESULT

Graph with axes labeled as follows:
- Vertical axis: $Q_1 (\text{KW})$
- Horizontal axis: TIME (MIN.)

The graph shows a downward curve, indicating a decrease in $Q_1$ with increasing time.
Hottest Result

Q2 (kW)

0.75

1.0

1.25

1.50

1.75

2.0

0 10 20 30 40 50

TIME (MIN.)
HOTST RESULT

M.O. (LBS./HR)

750
1000
1250

500

0 10 20 30 40 50

TIME (MIN.)
HOTST RESULT

TIME (MIN)

TO
HOTST RESULT
HOTST RESULTS
HOTSTI RESULT

TIME (MILLI SECONDS)

T - U

0.375
0.500
0.550
0.750
0.800
0.875

0
10
20
30
40
50
HOTST RESULT

The graph shows a significant increase in a parameter labeled 'SIG' over the time interval from 0 to 50 units. The parameter value rises sharply initially and then stabilizes over the subsequent time period.
HOTST RESULT
HOLEF R E S U L T

Graph showing a function $G(x)$ over time $T$ with values ranging from 0 to 0.6. The graph peaks at $G(x) = 0.6$ and experiences oscillations before settling back to a steady state.
CASE 2

Cold Start Case

Closed Loop
HOTST RESULT

Q2 (K/s)

TIME (s/si)

0 10 20 30 40 50
COLDST RESULT

\[ Q_1 (\text{KW}) \]

\[ \text{TIME (MIN.)} \]
COLDST RESULT

QS2 (kW)

TIME (MIN.)
COLEST RESULT

The graph shows the change in M0 (lbs/hr) over time (min). The y-axis represents the mass flow rate in pounds per hour (lbs/hr), while the x-axis represents time in minutes (min). The graph indicates a steady increase in mass flow rate with time, reaching a plateau at a certain point.
COLDST RESULT

![Graph showing a temperature vs time relationship with a minimum at around 40 minutes. The graph has a temperature axis ranging from 85 to 90 and a time axis from 0 to 50 minutes.]
COLDST RESULT
COLDST RESULT
COLDST RESULT

```
0
10
20
30
40
50

DEL
0.625
0.500
0.375
0.250
0.125

TIME (MIN.)

COL RESULT
```
COLDST RESULT

![Graph showing a decay curve with time (min.) on the x-axis and gamma (GAM) on the y-axis. The curve peaks at around 10 minutes and then decreases exponentially.]
PART II. THE INTELLIGENT CONTROLLER

II.1 Introduction

This section of the report details the development of algorithmic approaches, both conventional and AI-based, to diagnose select fault conditions in the STCS, determine best estimates of monitored process variable data, identify the type of fault and its location in the thermal loop, and provide appropriate status reports about means for isolating faulty components to the maintenance module with minimum disruption to the "normal" TCS functions.

In the operation of the STCS, interpreting and reporting to a large quantity of real-time data is a challenging task, particularly under off-normal conditions. The timely detection of fault conditions and the isolation of faulty components in the STCS is a task of paramount importance if the operational integrity of the system is to be maintained.

The expert system FDIC (Fault Diagnostics and Intelligent Control) has been developed to assist in determining the type of a fault, its precise location in the STCS configuration, and suggest means for isolating faulty components.

The design and development of the Intelligent Controller (IC), with FDIC as its major algorithmic component, is motivated by the assertion that overall thermal control system functions are enhanced if critical thermal loop component faults are detected to the finest practical resolution and isolated, for maintenance purposes, in the shortest possible time. The thermal loop control functions include both waste heat removal and rejection, as well as all appropriate control actions dictated by the conventional hierarchical
controller and aimed at maintaining specified temperature levels at critical network points.

The intelligent controller interacts and works harmoniously with the hierarchical controller to maintain system integrity. The STCS structure remains as simple as possible. The basic approach philosophy is to achieve stated control objectives by keeping the system topology to a "least" acceptable configuration while maintaining monitoring hardware, component, and software processing requirements to a minimum. Thus, reduced overall system complexity is a primary objective since it translates directly to energy economy and improved reliability.

The goals of the IC design are, therefore, to develop a fault detection and isolation scheme which

- Maximizes the sensitivity to component failure detection, and
- Minimizes the rate of failure detection-false alarms.

Usually these two design goals involve conflicting criteria and the IC design is called upon to optimize the trade-off.

Current practices in the area of sensor signal validation, fault detection, and isolation are limited to a few rather rudimentary techniques that rely, for the most part, on process symmetry characteristics and physical or analytic redundancy of observables. Typically, these techniques include like-sensor comparisons, limit checking, auctioneering, instrument-loop-integrity checking, calibration checking, and parity space representations. Major limitations of current techniques, such as computational complexity, large identifiable failure size, lack of automated processing, physical instrument redundancy, etc., can be overcome by use of a judicial combination of signal validation techniques based upon a parity-space representation, analytic
redundancy, and artificial intelligence tools. The proposed methodology incorporates novel analytical approaches and intelligent algorithms which guarantee its sensitivity to subtle failure modes that are transparent to most current techniques, offer through analytic redundancy a cost-effective alternative to additional sensor hardware, allow for detection of nonsensor components, provide acceptable fault isolation resolution, and assure robustness to false alarming.

The following sections describe, in detail, the basic methodology adopted for fault diagnosis and intelligent control, the assumptions, limitations, and constraints underlying the development of the algorithms, each one of the modules comprising the overall intelligent scheme, some preliminary simulation results, conclusion and recommendations for extending the salient features of the proposed approach to other fault conditions and subsystems of the common module.

II.2 Basic Methodology

Figure II-1 depicts, in block-diagrammatic form, the overall intelligent control philosophy. The system or thermal control process is viewed to consist of the STCS and the hierarchical controller with their combined objective directed towards achieving specified control goals under normal operating conditions. Temperature, flow, and pressure sensors provide, through the data management system, status information to the IC for all important process variables. Controller command signals are also provided to the IC directly from the hierarchical controller. On the basis of available information, the IC calls upon a triggering module to initiate the data validation/fault detection routines; upon identification of a fault, it
FIGURE II-1. BLOCK DIAGRAM OF THE INTELLIGENT CONTROLLER INTERFACE.
produces a status report detailing the fault type, its location, and such pertinent information as the degree of certainty and severity of the particular failure incident; it finally decides as to what appropriate measures must be taken to isolate effectively the faulty component.

It was agreed by BAC and the Georgia Tech research team that two types of faults will be considered for purposes of demonstrating the salient and innovative features of the proposed approach: a leaky pipe segment and a sticking valve. Three pipe segments and four valves were identified as "components" that may be subjected to a failure mode. They are shown in Figure II-2 which is a schematic representation of the STCS. The three pipe segments were chosen to represent typical piping configurations in the STCS network. Thus, component (1) is a segment in the STCS series configuration which carries the loop flow rate \( m_o \). Its failure implies a major system fault and requires system shut-down for maintenance and repair. Component (2) is the segment leading from the second heat sink, QHs2, to the ECLSS heat exchanger in parallel with the heat sink bypass which is providing one of the two inputs to the mixing valve. Failure of this component may be isolated with a temporary disrupting in waste heat removal from the ECLSS. Assuming that the hierarchical controller is capable of regulating flow and temperature conditions for the reconfigured system, the severity of the fault is not as pronounced as that of the first case. Finally, a leak in pipe segment (3) may be easily isolated with a consequent impact on the high \( \Delta T \) CP without affecting the operation of the rest of the system. The three components selected thus typify network branching conditions in the STCS. The detection methodology pursued in this study may be easily extended to include any number of pipe segments as failure candidates as long as a sufficient amount of
FIGURE 11-2. STCS SCHEMATIC.
either direct or analytic measurements or a combination of the two is available at the end nodes of each component. Such segments for which measurement data are available are designated as Least Process Units (LPUs). The four valves selected, V1, V2, V3, and V4, represent all possible automatic valving conditions encountered in the series STCS configuration. Again, the technique is not limited to this number of candidate faulty components. As a matter of fact, its basic principles may be applied to fault diagnosis and faulty component isolation for any component of the STCS (pumps, sensors, etc.) as long as the component is an LPU.

Some basic assumptions concerning the application of the proposed methodology to the STCS must be stated at this time:

1. It is assumed that all sensors are performing with zero probability of a faulty indication. Moreover, the list of faulty component candidates is limited to the one mentioned in the previous paragraph. Sensor accuracies, as specified by the manufacturers, have been considered in prioritizing and selecting analytical measurements for redundancy purposes as well as in estimating measurement errors.

2. It is assumed that component failures are of the single-point type. That is, for the case of the combination of three direct or analytic measurements of a variable, the single-point failure assumption allows the inconsistency of one of the measurements to be associated with the failure of the component for that measurement. However, if multiple failures are allowed, the inconsistency of one measurement of the three could be caused by either the failure of that component or the common-mode failure of the other two components. In general,
using a parity space approach in order to detect \( f \) failures unambiguously, the number of measurements, \( m \), must satisfy the condition \( m > 2f + 1 \) \([1]\). This condition holds for the scalar measurement case. Indeed, for the STCS all measured variables, such as flow rate, temperature, and pressure, are scalar quantities.

3. Detection to the level of a component (leaky pipe segment or stuck valve) can be guaranteed only for the first failure in the system. Thus, subsequent component failures may not be differentiated due to a lack of redundant direct or analytical measurements. If, for example, three measurements are available for a certain variable and the first failed component is detected, then only two measurements remain. The subsequent failure of one of the remaining two components can only be detected to the level of the sum, or the union, of the two remaining components. Basic limitations here refer to the STCS topology, number and placement of sensors, sensor accuracies, etc.

4. It is assumed that fault conditions remain fixed once they are initiated, i.e., no further degradation in component performance is observed after some initial triggering and fault identification. The methodology may comprise the basic element of a trend analysis algorithm capable of predicting the deteriorating behavior of a faulty component.

5. All pertinent data associated with the direct or analytic measurements of a variable for an LPU are taken in the immediate vicinity of the component and, therefore, time delays and data sampling rates are assumed to be negligible. In the event that they are not, data
synchronization must proceed the execution of the fault detection algorithm.

6. In order to achieve the control objectives, the design of the intelligent controller must be based on:
   a. The noise and bias characteristics of the sensors, and
   b. The noise and error propagation behavior of the analytic models.

Since details about sensor noise distributions are not available with sufficient accuracy, it is assumed that the sensor noise probability density function is uniform, i.e., modeled as shown in Figure II-4. The width b of the distribution is equal to the sum of the noise and bias errors. The uniform distribution assumption is usually a conservative estimate of sensor error statistics. Finally, it is assumed that the bias errors of the process measurements are equal to zero.

The assumptions considered above do not imply any limitations of the fault detection scheme. They are inherent to the STCS topology and the component/sensor selection. The intent is to demonstrate new and innovative concepts to be applied later, in their full development, to the design of the actual thermal control subsystem. The test cases described in this report demonstrate the power and robustness of the intelligent controller as a tool for fault diagnosis.

Figure II-3 is a block diagram of the major modules comprising the expert system Fault Diagnostics and Intelligent Control (FDIC). FDIC consists of the following programs: SEVA (SEnsor VAlidation), FADE (FAult DEtection), and FAIS (FAult ISolation). SEVA uses signal redundancy in a modified version of
FIGURE II-3. FAULT DETECTION AND ISOLATION MODULES.
FIGURE II-4. SENSOR NOISE PROBABILITY DENSITY FUNCTION.
the parity space algorithm for signal validation developed by the Charles Stark Draper Laboratories [2-12]. The validation procedure compares each reading to all other like readings. The "best" estimate from a set of good measurements is defined as that value which gives the minimum of a particular function of the measurements. Where more than the minimum number of measurements is available, the "best" estimate is taken in the least squares sense, i.e., it is that value which minimizes the square of the length of the measurement error vector.

FADE uses validated sensor and analytic data to analyze the status of the STCS components. Fault diagnosis involves the incremental accumulation of evidence in the form of symptoms to determine the "health" of the system components. Parity space representation and analytic redundancy, in addition to limit checking, are applied to accumulate the required evidence. Parity space representation transforms an array of redundant measurements into a new array, called a parity vector, in such a way that the true value of the underlying variable is suppressed, leaving only components of the measurement errors as elements of the parity vector [3]. In parity space, the parity vector remains near zero when the redundant measurements are consistent, i.e., when no faults are present. When a fault occurs, the parity vector grows in magnitude, and its direction of growth is uniquely associated with the faulty measurement. In this way, the parity vector behavior is an indication of both the presence of a fault and the identity of the fault measurement. Analytic redundancy refers to the utilization of process physical relationships, such as conservation of mass or energy, to calculate values for system variables from these analytic relations, thus augmenting available direct measurements. The analytic relations involve those system variables which are directly
monitored and their values have already been validated. Thus, for example, if only one direct measurement of a critical variable is available, the existence of two analytic measurements allows the failure of the direct measurement to be inferred via inconsistency with the analytic measurements.

A rule-based expert system within FADE incorporates multivalued logic to evaluate all the available symptoms accumulated through parity space or limit checking and to decide as to the degree of faultedness of a specific component, i.e., its fault status. The degree of confidence in the status, i.e., the degree to which the faultedness can be believed, is also ascertained.

FAIS, finally, suggests means for isolating faulty components once their faulty status and their location in the thermal loop have been identified. Component isolation is accomplished in a way that minimizes any adverse effects to the overall STCS operation.

II.3 Fault Triggering

It is of interest that the Fault Detection and Isolation program be initiated only when some evidence is present which implies the possible failure of a system component. Such triggering of the fault detection routines results in substantial computational savings and improves the reliability of the algorithm by reducing the frequency of the false alarms.

The STCS may be operating in either one of two normal modes: In steady-state, where all thermal loads remain constant and conventional controllers (valve motors, pump motors, etc.) maintain their set points, and transient state, implying that at least one thermal load is changing, resulting in a
continuous change of local or global control command signals. The controller action is, of course, intended to maintain specified temperature constraints.

A. **Triggering for a Leaky Pipe**

Under steady-state conditions, two triggering mechanisms are considered to initiate the leaky segment fault detection scheme. The first one is based simply upon the fact that controller command signals are presumed to remain unchanged whenever the STCS maintains a steady-state status. Thus, any change in a command signal $\Delta C$ beyond some noise threshold level, may be used to trigger the fault detection algorithm.

The second trigger mechanism requires some slight component modifications and relies upon the detection of fluid flow through the accumulation (make-up water) tank. The proposed configuration is shown in Figure II-5. A level indicator may be accommodated either inside the tank structure or externally as shown in Figure II-5. An on-off solenoid valve is used to control the refill line. A timing mechanism is triggered whenever the fluid level falls below a preset value (allowing for normal variations from the full level due to vaporization). The corresponding time instant $t_i$ is stored and compared to the time instant at which the fluid level has fallen to a predetermined value ($t_f$). In parallel, the refill line solenoid valve is actuated by the close solenoid valve command signal $F^C$ when the fluid level is at its maximum value (full level), and is commanded to open again ($E^C$) when the make-up fluid reaches the empty level. The make-up fluid flow is computed as

$$\Delta m = \frac{\Delta m}{t_f - t_i}$$
FIGURE II-5. MAKE-UP TANK TRIGGER MECHANISM FOR FAULT DETECTION.
where \( \Delta m \) is a known fluid mass. If \( \Delta m \) exceeds some predetermined threshold, it signifies the possible existence of a leaky pipe segment and triggers the fault detection algorithm. Figure II-6 is a flow chart for the computer implementation of the make-up tank trigger scheme.

Under transient conditions, only the make-up tank scheme may be used as a triggering mechanism.

Two points are worth noting at this time: First, the triggering mechanism described above is also exploited as a symptom in the fault detection algorithm, and second, flow discrepancies, as used in the detection scheme and described fully in the sequel, may be employed to perform the triggering function.

**B. Triggering for a Stuck Valve**

A motorized valve may remain at a certain position although a command signal is attempting to change its setting. An obvious triggering mechanism involves initiation of the fault detection routine whenever a change in valve position command is issued. Figure II-7(a) depicts a flow chart for this trigger mechanism. The command signal must exceed some threshold value and a trigger alarm is issued only after a delay time \( T \) elapses which is large compared to actuator time constants.

The triggering mechanism described above suffers from a lack of specificity to prevailing system conditions under a component (valve) failure. It is susceptible to frequent false alarms and compounds the magnitude of the computational problem. It is, therefore, more appropriate to consider changes in the TCS/hierarchical controller behavior as possible triggering mechanisms. A sticky valve will result in (1) failure to operate the system at its optimum setting (i.e., deviation from \( m_0 \) min), and/or (2) violation of some temper-
USE $F_c$ TO START TIME COUNTER

SAVE $t_i$

PROGRAM CONTINUES

INTERUPT BY EMPTY COMMAND $E_c$

GET $t_f$

CALCULATE

\[ \Delta m = \frac{\Delta m}{t_f - t_i} \]

POSSIBILITY OF LEAK

IF YES, CALL FDIC

PROGRAM CONTINUES

FIGURE II-6. COMPUTER IMPLEMENTATION OF MAKE-UP TANK TRIGGER SCHEME.
FIGURE II-7. FLOWCHART AND THRESHOLD LEVEL FOR STUCK VALVE TRIGGER MECHANISM.
ature constraint. The triggering algorithm can be based upon detection of a set point discrepancy beyond reasonable tolerance levels.

II.4 Sensor Data Validation

The tasks of the Sensor Validation (SEVA) program are the following:

1. At each measurement point to retrieve from the Data Management module direct measurement values for the variable of interest. A measurement point is defined to be the beginning or the end node of a leaky pipe segment and the immediate environment of a stuck valve. Figure II-2 indicates as $a_1$, $b_1$ the measurement points for component (1), $a_2$, $b_2$ for component (2), etc.

2. To calculate those analytic measurement values, at each measurement point, which are needed to obtain the "best" estimate of the measured variable.

3. To determine the normal error bounds for the measured variables.

4. To formulate and evaluate the final component measurement variables and their corresponding error bounds which enter the fault detection algorithm as the redundant "measurements" of the parity space technique.

A comment concerning the discriminating properties of the detection scheme is in order. How small a leak or what minimum change in valve position can be detected is, of course, a function of the instrument error statistics and the error propagation properties when analytic measurements are considered. Attention, therefore, is given to these topics below. It is shown that the use of thermocouples with greater accuracies will significantly improve the detection resolution of the scheme.

II-19
The STCS is presently designed to use three types of sensors: a flow meter with an accuracy of 0.05%, a thermocouple with an accuracy of 0.5°C (0.9°F), and a pressure gauge accurate to 0.5%.

An analytic measurement is expressed as some functional dependence on values of variables, which are obtained from process measurements. The latter are always associated with measurement uncertainties and bias errors. In general, an analytic measurement may be expressed as \( f(x_1, x_2, \ldots, x_n) \).

If \( \varepsilon_{x_1}, \varepsilon_{x_2}, \ldots, \varepsilon_{x_n} \) are the error bounds corresponding to the direct measurements \( x_1, x_2, \ldots, x_n \), then a Taylor's series expansion gives

\[
\begin{align*}
  f(x_1 \pm \varepsilon_{x_1}, \ldots, x_n \pm \varepsilon_{x_n}) &= f(x_1, x_2, \ldots, x_n) \pm \varepsilon_{x_1} \frac{\partial f}{\partial x_1} \pm \varepsilon_{x_2} \frac{\partial f}{\partial x_2} \pm \ldots \pm \varepsilon_{x_n} \frac{\partial f}{\partial x_n} \pm (0)^n
\end{align*}
\]

Thus, the analytic measurement error, \( \varepsilon_f \), may be expressed as

\[
\begin{align*}
  \varepsilon_f &= f(x_1 \pm \varepsilon_{x_1}, \ldots, x_n \pm \varepsilon_{x_n}) - f(x_1, x_2, \ldots, x_n) \\
  &= \pm \varepsilon_{x_1} \frac{\partial f}{\partial x_1} \pm \varepsilon_{x_2} \frac{\partial f}{\partial x_2} \pm \ldots \pm \varepsilon_{x_n} \frac{\partial f}{\partial x_n}
\end{align*}
\]

\( \text{(II.1)} \)

A conservative analytic estimate of the error threshold may be obtained by performing a standard error propagation calculation under a worst-case scenario. From equation (1), consider the worst case:

\[
\begin{align*}
  \varepsilon_f &= \varepsilon_{x_1} \left| \frac{\partial f}{\partial x_1} \right| + \varepsilon_{x_2} \left| \frac{\partial f}{\partial x_2} \right| + \ldots + \varepsilon_{x_n} \left| \frac{\partial f}{\partial x_n} \right| = \sum_{i=1}^{n} \varepsilon_{x_i} \left| \frac{\partial f}{\partial x_i} \right|
\end{align*}
\]

\( \text{(II.2)} \)
Let us examine those cases that are encountered in the STCS analytic measurement calculations:

(1) \( f(x_1, x_2) = x_1 + x_2 \)

\[ + \frac{\epsilon_f}{x_1 + x_2} = \frac{\epsilon}{x_1} + \frac{\epsilon}{x_2} \] (II.3)

(2) \( f(x_1, x_2) = x_1 - x_2 \)

\[ + \frac{\epsilon_f}{x_1 - x_2} = \frac{\epsilon}{x_1} + \frac{\epsilon}{x_2} \] (II.4)

(3) \( f(x_1, x_2) = x_1 x_2 \)

\[ + \frac{\epsilon_f}{x_1 x_2} = \frac{\epsilon}{x_1} x_2 + \frac{\epsilon}{x_2} x_1 \]
\[ = x_1 x_2 \left( \frac{\epsilon}{x_1} + \frac{\epsilon}{x_2} \right) \]
\[ = x_1 x_2 \left( \delta x_1 + \delta x_2 \right) \] (II.5)

where

\[ \delta x_1 = \frac{\epsilon}{x_1} \] relative errors
\[ \delta x_2 = \frac{\epsilon}{x_2} \]
(4) \[ f(x_1,x_2) = \frac{x_1}{x_2} \]

\[ + \epsilon_{f_{x_1 x_2}} = \epsilon_{x_1} \frac{1}{x_2} + \epsilon_{x_2} \frac{x_1}{x_2} \]

\[ = \frac{x_1}{x_2} \left( \frac{\epsilon}{x_1} + \frac{\epsilon_{x_2}}{x_2} \right) \]

\[ = \frac{x_1}{x_2} \left( \delta + \frac{\delta_{x_2}}{x_2} \right) \]

\[ = f(x_1,x_2) (\delta + \frac{\delta_{x_2}}{x_2}) \]  

(II.7)

(5) \[ f(x_1) = \sqrt{x_1} \]

\[ + \epsilon_{f_{x_1 x_1}} = \epsilon_{x_1} \cdot \frac{1}{2} \cdot \frac{1}{\sqrt{x_1}} \]

\[ = \frac{1}{2} \sqrt{x_1} \left( \frac{\epsilon_{x_1}}{x_1} \right) \]

\[ = f(x_1) \left[ \frac{1}{2} \delta_{x_1} \right] \]  

(II.8)

In summary:

From (II.3) and (II.4):

\[ \epsilon_{f_{x_1 x_2}} = \epsilon_{f_{x_1 x_2}} = \epsilon_{x_1} + \epsilon_{x_2} \]  

(II.9)

From (II.6) and (II.8)

\[ \epsilon_{f_{x_1 x_2}} = \epsilon_{f_{x_1 x_2}} = f(x_1,x_2) (\delta + \frac{\delta_{x_2}}{x_2}) \]  

(II.10)
From (II.9)

$$\varepsilon_{x_1} = f(x_1) \left[ \frac{1}{2} \delta_{x_1} \right]$$

(II.12)

The error propagation approach will be illustrated via an example. Analytic errors for other variables or parameters of interest are estimated using a similar detailed procedure. They will be summarized in the sequel.

Consider the pressure-flow characteristic for a valve as given in Figure II-8.

The pressure-flow relationship may be written as

$$C_v = \frac{Q(\text{GPM})}{\sqrt{\Delta P(\text{PSID})}}$$

(II.13)

where $C_v$ is a function of the percent open position of the valve. It is of interest to consider $C_v$ as a detecting parameter for a stuck valve. Indeed, if the valve sticks at a particular position, then $C_v$ holds the same value before and after the change in position command is issued. The coefficient $C_v$ may be used as the discriminating "measurement." Its value may be derived from (II.13) using direct measurements of flow rate and pressure drop. An analytic estimate of the error bound in $C_v$ is sought.

The typical sensor and piping configuration is shown in Figure II-9. Given the FM and $\Delta P$ measurement accuracies (.05% and 0.5%, respectively), the objective is to determine the threshold (error bound) to be used in deciding whether $C_v$ remains constant or not. From Figure II-9:

$$C_v = \frac{Q_2}{\sqrt{\Delta P}}$$
KURZ VALVE 731-1000-500

\[ *C_v = \frac{Q \text{ (GPM)}}{\sqrt{\Delta P \text{ (PSID)}}} \]

CONDITIONS:
1) WATER AT 100°F
2) INLET PRES = 30 PSIA

FIGURE II-8. KURZ VALVE CHARACTERISTICS.
FIGURE II-9. TYPICAL SENSOR AND PIPING CONFIGURATION.
Q₁ is directly measured using FM1 with an accuracy of 0.05%, while Q₀ is monitored directly with a flow meter at the pump outlet whose measurement accuracy is also 0.05%. Assuming that the measurement inaccuracy has uniform distribution, the measurements may be expressed as

\[ Q_0 + \varepsilon_{Q_0} = (Q_P + \varepsilon_{Q_P}) + (Q_1 + \varepsilon_{Q_1}) \]

\[ = 2Q_P + 2\varepsilon_{Q_P} \]

\[ = 2Q_P + 0.05\% (2Q_P) \quad \text{(II.14)} \]

and

\[ Q_2 + \varepsilon_{Q_2} = (Q_0 + \varepsilon_{Q_0}) - (Q_1 + \varepsilon_{Q_1}) \]

\[ = (Q_0 - Q_1) + (\varepsilon_{Q_0} + \varepsilon_{Q_1}) \quad \text{(II.15)} \]

where Qₚ is the (direct) reading of the pump outlet flow meter, and

\[ \varepsilon_{Q_P} = 0.05\% Q_P \]

Therefore,

\[ C_V = \frac{Q_2}{\sqrt{\Delta P}} + C_V + \varepsilon_{C_V} = \frac{Q_2 + \varepsilon_{Q_2}}{\sqrt{\Delta P} + \varepsilon_{\Delta P}} \quad \text{(II.16)} \]

From (II.9):

\[ \varepsilon_{f_{\Delta P}} = \sqrt{\Delta P} \left( \frac{1}{2} \delta_{\Delta P} \right) \]

and from (II.8):

\[ \varepsilon_{f_{Q_2}} = \frac{Q_2}{\sqrt{\Delta P}} \left( \delta_{Q_2} + \frac{1}{2} \delta_{\Delta P} \right) \quad \text{(II.17)} \]
where

\[
\delta Q_2 = \frac{0.05\% Q_0 + 0.05\% Q_1}{Q_0 - Q_1}
\]

\[
\delta \Delta P = 0.5\% \Delta P
\]

A. "Measurement" Variables

Fault detection for a leaky pipe segment or a stuck valve is based upon determining discrepancies or inconsistencies in the measured values (direct or analytic) of certain variables which are most sensitive to the particular fault condition. These measurements are defined below for the leaky pipe segment and the stuck valve and their "best" estimates are used subsequently to provide one of the symptom components in the fault detection algorithm.

Let us consider first the case of a leaky pipe segment. Figure II-10 depicts an idealized segment of piping with the flow direction from \(a_1\) to \(b_1\). A leak of magnitude \(\Delta m\) results in a flow discrepancy of the same magnitude between points \(a_1\) and \(b_1\). Assuming that the loss of fluid through the leaky segment is small enough so that it is made up from the accumulation tank. Figures II-11(a) and (b) show the approximate distribution of pressure and flow rate, respectively, throughout the thermal loop. A constant pressure pump is providing the head for the system. Sensitivity analysis and consideration of instrument inaccuracies dictate that a reasonable measure of the fault condition may be arrived at by estimating the flow rate discrepancies between points \(a_1\) and \(b_1\), i.e., by monitoring the "measurement"

\[
\dot{m}_1 = \dot{m}_{a_1} - \dot{m}_{b_1}
\]

(II.18)
FIGURE II-10. IDEALIZED SEGMENT OF PIPING WITH A LEAK.
FIGURE II-11. APPROXIMATE DISTRIBUTION OF PRESSURE AND FLOW RATE THROUGH THE THERMAL LOOP.
Similar flow rate measurements are considered for components (2) and (3) (see Figure II-2), i.e.,

\[ m_2 = m_{a_2} - m_{b_2} \]  

and

\[ m_3 = m_{a_3} - m_{b_3} \]

Equations (II.18), (II.19), and (II.20) form the measurement vector \( \hat{m} = [\hat{m}_1, \hat{m}_2, \hat{m}_3]^T \) that will be employed in the parity space algorithm.

The measurements \( m_{a_1}, m_{b_1}, m_{a_2}, \ldots \) could be either direct or analytic or a combination of both depending upon the sensor availability at points \( a_1, b_1, a_2, \ldots \). When more than one measurements are available at the beginning or end node of a given component, then the "best" estimate is used in Equations (II.18)-(II.20), in the sense that this estimate minimizes a least squares error criterion. In Appendix A, it is shown that, when redundant measurements are available, the best estimate, in the least squares sense, is that value which minimizes the square of the length of the measurement error vector. When the measured quantity is a scalar (as indeed is the case with all flow, pressure or temperature measurements), the least squares estimate is simply the average of the measurements.

We summarize below the derivation of the measurement vector for the leak detection algorithm and the associated error bounds. Refer to Figure II-2:

For component (1):

\[ \hat{m}_1 = \hat{m}_{a_1} - \hat{m}_{b_1} \]

where

\[ \hat{m}_{a_1} = \frac{1}{3} (FM20 + FM21) + FM26 \cdot \frac{TC10 - TC9}{TC11 - TC9} + \frac{AP38}{C_{Hx2}} \cdot \frac{TC10 - TC9}{TC11 - TC9} \]
with $C_{vHx2} = 2.48 \times 10^{-4}$ psi/(lb/hr)$^2$ and FM20, FM21, etc., are direct readings from the corresponding sensors shown in Figure II-2.

\[
\hat{m}_{b1} = \frac{1}{2} \left( \frac{\Delta m_{40}}{C_{vHx3}} + \frac{\Delta m_{41}}{C_{vHx4}} + \frac{\Delta m_{42}}{C_{vHx5}} \right) \left( \frac{TC_{17} - TC_{11}}{TC_{18} - TC_{11}} \right)
\]

with

\[
C_{vHx4} = C_{vHx5} = 2.48 \times 10^{-4} \text{ psi/(lb/hr)}^2
\]

The error bound, $b(m_1)$, for measurement $m_1$ is estimated to be 20.67 lbs/hr, on the basis of given instrument inaccuracies.

For component (2):

\[
m_2 = \hat{m}_{a2} - \hat{m}_{b2}
\]

where

\[
\hat{m}_{a2} = FM24
\]

\[
\hat{m}_{b2} = \frac{1}{2} \left( \frac{\Delta m_{37}}{C_{vHx1}} \cdot \frac{TC_{7} - TC_{6}}{TC_{8} - TC_{6}} \right)
\]

with

\[
C_{vHx1} = 8.16 \times 10^{-6} \text{ psi/(lb/hr)}^2
\]

The error bound is $b(m_2) = 14.3$ lbs/hr.

For component (3):

\[
m_3 = \hat{m}_{a3} - \hat{m}_{b3}
\]
where
\[
\dot{m}_{a_3} = FM27
\]
\[
\dot{m}_{b_3} = \frac{-167 + \sqrt{(167)^2 + 4 \times 10^5 \Delta P_{40}}}{2}
\]

The corresponding error bound is $b(m_3) = 0.71$ lbs/hr.

The error bounds for the leaky pipe measurement vector were estimated on the hypothesis that a temperature transducer will be used with an accuracy of 0.5% instead of the 0.9°F accurate thermocouples currently available. For the 0.9°F thermocouples, it was estimated that the flow discrepancy error bounds exceed 1000 lbs/hr. Normal error bounds of such magnitude reduce considerably the discriminating properties of the fault detection algorithm. Furthermore, the full utility of redundancy to validate sensor measurements is realized only when available direct and analytic data have comparable resolutions.

Next, we consider the control valve fault analysis. The bypass control valves $\beta, \gamma, \delta$, and $\sigma$ do not incorporate a positioner and, therefore, a stuck valve cannot be detected by direct measurement of the valve position. In this case, it becomes necessary to consider some indirect quantity in order to express an inconsistency measure useful in the parity space analysis. Three alternative means are examined below as possible candidates based on temperature measurements, flow rate measurements, or the valve pressure-flow characteristics.

**Case 1.** Consider Figure II-12 representing the Low ΔT Cold Plate valving and sensing configuration. From the continuity relation
FIGURE II-12. LOW AT COLD PLATE VALVING AND SENSING CONFIGURATION.
\[ m_o = m_1 + m_2 \]

Define \( \beta \) again as

\[ \beta = \frac{m_2}{m_o} \quad \text{or} \quad 1 - \beta = \frac{m_1}{m_o} \]

The loop flow rate \( m_o \) is shared equally by the two centrifugal pumps, each carrying a flow rate of \( m_p \). Therefore, \( m_o = 2m_p \). Assuming full load conditions for the STCS (worst-case scenario),

\[ m_1 = 1362 \pm 0.68 \quad \text{,} \quad m_p = 800 \pm 0.4 \]

then

\[ \beta = 1 - \frac{m_1}{2m_p} = 1 - \frac{1362 \pm 0.05\%}{1600 \pm 0.05\%} = 1 - (0.85 \pm 0.1\%) = 0.15 \pm 0.00085 = 0.15 \pm 0.57\% \]

The resolution is about 0.1\% of the whole operating range.

**Case 2.** The same flow rate ratio \( \beta \) may be expressed in terms of direct temperature measurements as follows:

From energy conservation considerations:

\[ m_1 T_2 + m_2 T_1 = m_o T_3 \]

or

\[ m_1 T_2 + m_2 T_1 = (m_1 + m_2)T_3 \]

which gives after rearranging
\[ m_1 = m_2 \left( \frac{T_3 - T_1}{T_2 - T_3} \right) \]

Then
\[ \beta = \frac{m_2}{m_0} = \frac{T_2 - T_3}{T_2 - T_1} \]

or
\[ 1 - \beta = \frac{T_3 - T_1}{T_2 - T_1} \]

Under full load conditions again
\[ \beta = \frac{T_2 - T_3}{T_2 - T_1} = \frac{(75 \pm 0.9) - (74.2 \pm 0.9)}{(75 \pm 0.9) - (70 \pm 0.9)} \]
\[ = \frac{0.8 \pm 1.8}{5 \pm 1.8} = \frac{0.8 \pm 257\%}{5 \pm 36\%} \]
\[ = 0.15 \pm 293\% = 0.15 \pm 0.41 \]

That is, the resolution over the full operating range is 41%. This result is characteristic of the choice of temperature sensors with rather gross inaccuracies compared to those of flow meters and pressure transducers. The problem is further compounded by the fact that we are called to monitor rather small temperature variations. As it is suggested in the sequel, if temperature sensors with an improved accuracy of 0.5% are used, then the fault detection resolution of the proposed algorithm may be substantially improved.

Case 3. For the Kurz valve, the coefficient \( C_v^* \) is related to the % open position of the valve (see Figure II-8) when the valve sticks and, therefore, its position remains constant which implies that \( C_v^* \) must stay constant. If a command is issued for a valve position change and \( C_v^* \) does not respond...
appropriately, then a measure of the discrepancy in the coefficient values may be used to detect the presence of a fault. The value of $C_v^*$ may be inferred from direct sensor measurements as follows:

Again refer to Figure II-12:

\[
C_v^* = \frac{m_2}{\sqrt{\Delta P_1}}
\]

$\Delta P_1$ may be directly measured with the differential pressure transducer and $m_2$ may be related to direct measurements of $m_o$ and $m_1$. Thus

\[
m_2 = 2m_p - m_1
\]

and

\[
C_v^* = \frac{m_2}{\sqrt{\Delta P_1}} = \frac{2m_p - m_1}{\sqrt{\Delta P_1}}
\]

Under full load conditions, with

\[
Q_p = 1.6 \text{ GPM} \quad Q_1 = 2.72 \text{ GPM} \quad \Delta P = 7.1 \text{ PSID}
\]

\[
C_v^* = \frac{(3.2 \pm 0.0016) - (2.72 \pm 0.0014)}{\sqrt{7.1 \pm 0.5\%}} = 0.48 \pm 0.625\%
\]

\[
= 2.66 \pm 0.25\%
\]

\[
= 0.18 \pm 0.0017
\]

The resolution is, therefore, about 0.2% on the basis of the measurement's full operating range. We summarize below similar results for the mixing valve and the bypass valves $\gamma$ and $\sigma$. 

II-36
For the mixing valve (component V4) refer to Figure II-13. Using the flow meter sensors, we can write

\[
\alpha = \frac{m_1}{m_o} = 1 - \frac{m_2}{m_o}
\]

Under typical load conditions again

\[
m_0 = 1362.4 \text{ lbs/hr} \pm 0.1\%
\]

\[
m_2 = 439.2 \text{ lbs/hr} \pm 0.05\%
\]

and

\[
1 - \alpha = \frac{m_2}{m_0} = \frac{439.2 \pm 0.05\%}{1362.4 \pm 0.1\%} = 0.32 \pm 0.00048
\]

or

\[
\alpha = 0.68 \pm 0.00048
\]

If the temperature sensors are employed, then

\[
\alpha = \frac{m_1}{m_o} = \frac{T_3 - T_2}{T_1 - T_2}
\]

and

\[
\alpha = \frac{(70 \pm 0.9) - (59 \pm 0.9)}{(75.2 \pm 0.9) - (59 \pm 0.9)} = \frac{11 \pm 1.8}{16.2 \pm 1.8} = 0.68 \pm 0.19
\]
FIGURE II-13. MIXING VALVE SENSOR CONFIGURATION.
A third possibility would involve use of the mixing valve positioner directly. Information on the positioner statistics is not currently available.

It is observed again that flow rate measurements are significantly more sensitive than temperature measurements. If, for example, a measure of the absolute difference $|a_{\text{new}} - a_{\text{old}}|$ is used to detect a sticking mixing valve after a command is issued, then for $|a_{\text{new}} - a_{\text{old}}| < 0.00048$ a fault condition might be present when flow meter data is used while the same criterion is expressed as $|a_{\text{new}} - a_{\text{old}}| < 0.19$ with temperature sensor data.

For the heat sink bypass valve $\sigma$ (component V3), refer to Figure II-14. Here

\[
\sigma = \frac{m_1}{m_s}
\]

and $m_s$ is measured with FM24, while $m_1$ is monitored with FM23. Under full load conditions, $m_s = 439.2$ lbs/hr and $m_1 = 39.5$ lbs/hr, therefore

\[
\sigma = \frac{39.5 \pm 0.05\%}{439.2 \pm 0.05\%} = 0.09 \pm 0.00009
\]

Temperature measurements can also be used since

\[
\sigma = \frac{TC_7 - TC_5}{TC_7 - TC_6}
\]

but the resolution would be substantially reduced compared to the previous case.

Finally, the bypass valve $\gamma$ flow ratio (component V2) may be expressed as (see Figure II-15):

\[
\gamma = \frac{m_o - (m_3 + m_4 + m_5)}{m_o}
\]
FIGURE II-14. HEAT SINK BYPASS VALVE SENSOR CONFIGURATION.
FIGURE II-15. PARALLEL HX BYPASS VALVE SENSOR CONFIGURATION.
or

\[ \gamma = \frac{TC17 - TC18}{TC17 - TC11} \]

with a preference again for the flow meter measurements.

After this necessary digression, let us return to the problem of defining an appropriate measurement vector for the valve components.

In Figure 11-16, the flow ratio \( C \) is defined as

\[ C \equiv \frac{m_1}{m_0} \]

Then, a measurement for component \( V_i \) may be defined by the relation

\[ m_{V_i} = 1 - \left| \frac{C_i^1 - C_i^2}{C_i^1 - C_i^2} \right| \]

where

- \( C_i^1 \) = actual value of \( C_i \) before command is issued
- \( C_i^2 \) = actual value of \( C_i \) after command is issued
- \( C_i^{C1} \) = command signal before change in command is issued
- \( C_i^{C2} \) = command signal after change in command is issued.

Thus

\[ m_{V_i} = 1 - \left| \frac{\Delta C_{\text{actual}}}{\Delta C_{\text{command}}} \right| \]  \hspace{1cm} (II.21)

In summary, the measurement vector, \( m_{V_i} \), for the four valve components will be computed as follows:

For Component \( V_i \):

Using only flow meter readings:
FIGURE II-16. DEFINITION OF FLOW RATIO PARAMETER C.
\[ C_{FM} = 1 - \frac{FM26}{FM20 + FM21} \]

and

\[ m_{V1} = 1 - \left| \frac{\Delta C_{FM}}{V_1/V_T} \right| \]

where

\( V_1 \) = command voltage needed to produce a \( \Delta C_{FM} \) change

\( V_T \) = voltage required to change the valve position from close to full open

For Component V2:

\[ C_{FM} = 1 - \frac{FM27 + FM28 + FM29}{FM20 + FM21} \]

and

\[ m_{V2} = 1 - \left| \frac{C_{FM \ initial} - C_{FM \ final}}{V_2/V_T} \right| \]

For Component V3:

\[ C_{FM} = \frac{FM23}{FM24} \]

and

\[ m_{V3} = 1 - \left| \frac{C_{FM \ initial} - C_{FM \ final}}{V_3/V_T} \right| \]

For Component V4:

\[ C_{FM} = 1 - \frac{FM24}{FM20 + FM21} \]

and

\[ m_{V4} = 1 - \left| \frac{C_{FM \ initial} - C_{FM \ final}}{V_4/V_T} \right| \]

The corresponding error bounds are estimated to be:
\[ b_{V1} = 0.002 \]
\[ b_{V2} = 0.002 \]
\[ b_{V3} = 0.002 \]
\[ b_{V4} = 0.002 \]

A FORTRAN program called SEVA has been written and is included in the appendix to provide the measurement vector and the error bounds. The program uses current data from the appropriate sensors. Output from SEVA is fed to the FADE module. Specifically, it is the input to the parity space algorithm.

II.5 Fault Detection

The fault detection module (FADE) gathers information from the Data Management System, SEVA, and any hierarchical controller setting discrepancies as symptoms for the possible existence of a component failure. Information about the measurement vectors which incorporate measures of flow discrepancies and valve positioning are provided to FADE from SEVA. Subsequently, all of the symptoms are normalized to a unique universe of discourse for diagnostic purposes. Normalized values of the symptoms are fed into a multivalued rule-base to determine the type, location, and the severity of a fault.

The FADE algorithm is initiated by prioritizing the available evidence for the existence of a faulty component. If the evidence strongly suggests that a possible fault condition is due to a leaky pipe segment, then it initiates the leak component detection part of the algorithm. If, on the other hand, the evidence is suggestive of a sticking valve, then the second part of the algorithm, involving the sticking valve detection, is initiated.
Clearly, strong evidence for a fluid leak is associated with loss of fluid from the accumulation tank and controller command signal changes under steady-state conditions. The possibility of a sticking valve is greatly enhanced whenever no loss in make-up fluid is observed while simultaneously conventional controller settings deviate substantially from desired or set values. This branching of the detection algorithm may be implemented with the following rules:

\[
\text{IF } \Delta m > \text{threshold}_1 \text{ and steady-state with controller command signal > threshold}_2, \text{ THEN initiate leaky component analysis}
\]

\[
\text{IF system is in transient condition and } \Delta m < \text{threshold}_3 \text{ and controller setting > set limit (bound), THEN initiate stuck valve component analysis}
\]

where

\[
\begin{align*}
\text{threshold}_1 &= 100 \text{ lb/hr} = 1.66 \text{ lb/min} \\
\text{threshold}_2 &= 1 \text{ volt} \\
\text{threshold}_3 &= 1 \text{ lb/hr} \\
\text{threshold}_4 &= \frac{|\dot{m}_{\text{op}} - \dot{m}_{\text{acr}}|}{\dot{m}_{\text{op}}} \times 10\%
\end{align*}
\]

Figure II-17 depicts a block diagram of the principle functions of the fault detection module.

The fault detection part of the algorithm capitalizes upon the behavior of the magnitude of the parity vector to detect the presence of a fault. Let us review first some of the mathematical details relating to the recognition of measurement inconsistencies:
FIGURE II-17. BLOCK DIAGRAM OF THE MAJOR FADE FUNCTIONS.
Let us suppose that

$$\mathbf{m} = \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_n \end{bmatrix} \quad (11.22)$$

is a redundant data vector whose elements are obtained from either direct sensor or analytic models. Let $x$ be the true value of the scalar process variable (flow rate, pressure, etc.) and

$$\mathbf{e} = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix} \quad (11.23)$$

is the error vector associated with the redundant data.

$$\mathbf{H} = [1,1,\ldots,1]^T \quad (11.24)$$

is the measurement matrix. Its elements are all unity since all the measurements are weighted equally. In the event of a bias for a subset of the measurements, this fact will be appropriately reflected in the elements of $\mathbf{H}$.

If a pair of measurements $m_1, m_2$ are made of the same physical quantity (i.e., mass flow rate), then with $e_1 = e_2 = 0$

$$m_1 - m_2 = 0$$

Such an equation is called a parity equation, since it asserts the equality of the two measurements. When the measurements contain errors, the parity equation becomes:
or, equivalently
\[ (m_1 - \varepsilon_1) - (m_2 - \varepsilon_2) = 0 \]
The value of the residual \( \eta \) may be considered a measure of the lack of agreement or "inconsistency" in the measurements.

The value of \( \eta \) is not directly a measure of the error. When a third measurement \( m_3 \) is added, then there are three possible parity equations:

\[ m_1 - m_2 = \varepsilon_1 - \varepsilon_2 = \eta_1 \]
\[ m_2 - m_3 = \varepsilon_2 - \varepsilon_3 = \eta_2 \]
\[ m_3 - m_1 = \varepsilon_3 - \varepsilon_1 = \eta_3 \]

Since the third equation is a linear combination of the other two, only two degrees of inconsistency between the three measurements may be identified. Consideration of a fourth measurement will introduce three more parity equations, but only one more degree of inconsistency.

In general, it is noted that the parity equations are independent of the true value \( x \) of the variable and are functions only of the measurement errors. The number of linearly independent parity equations is always equal to the number of excess sources of available information.

The space spanned by the independent parity equations is called **parity space**. Its dimension is equal to the number of measurements minus one for a scalar variable. For three measurements, the parity space is a two-
dimensional plane while for four measurements the parity space is a three-dimensional space. If there are \( l \) parity equations, then there are only \( q = l - 1 \) independent relations among the \( l \) parity equations. The inconsistency values may, therefore, be expressed in terms of \( q \) independent variables, say \( p_1, p_2, \ldots, p_q \). The vector \( p = [p_1, p_2, \ldots, p_q]^T \) is called the parity vector. All information contained in the \( C_1^l \) parity equations is available from the \( q \) components of the parity vector. Because the dimension of the parity vector is less than the dimension of the measurement vector \( m \), considerable simplification and insight is provided by expressing the failure detection algorithm in terms of the parity vector. Moreover, the parity vector embodies attributes of error distribution throughout the system components.

To construct a parity vector \( p \), let \( V \) be a \( q \times l \) matrix whose rows are any \( q \) linearly independent null vectors of \( H^T \). Such a set can be found, for example, if the rows are taken simply as the coefficients of the first \( q \) parity equations. Then by construction \( VH = 0 \) and the parity vector may be taken as

\[
p = Vm
\]  

(II.27)

It can be shown [3] that indeed \( p \) is a parity vector in the sense defined above.

Next the concept of a base vector is defined as follows: Consider a base vector \( x_b \) as

\[
x_b = Kn = K[Hx + \varepsilon] = x + K\varepsilon
\]  

(II.28)

where \( K \) is any \( 1 \times l \) left inverse of \( H \), so that \( KH = I \). The vector \( x_b \) equals \( x \) when there are no errors, and approximates it when the errors are small.
The base vectors for \( \mathbf{x} \) are not uniquely determined. One base vector of particular importance is the so-called least squares estimate. When redundant measurements are available, the best estimate in the least squares sense is that value which minimizes the square of the length of the error vector, i.e., \( \mathbf{e}^T \mathbf{e} \). It can be shown that the minimum is obtained when \( \mathbf{x} = \hat{\mathbf{x}} = \mathbf{Km} \), where

\[
\mathbf{K} = (\mathbf{H}^T \mathbf{H})^{-1} \mathbf{H}^T \tag{II.29}
\]

\( \hat{\mathbf{x}} \) is a legitimate base vector since \( \mathbf{KH} = 1 \). When the least squares estimate \( \hat{\mathbf{x}} \) is used as a base vector, the measurement equations may be written as

\[
\mathbf{m} = \hat{\mathbf{m}} + \hat{\mathbf{e}}
\]

where by definition

\[
\hat{\mathbf{m}} = \mathbf{HKm}
\]

and

\[
\hat{\mathbf{e}} = \mathbf{m} - \hat{\mathbf{m}} = \mathbf{V}^T \hat{\mathbf{p}}
\]

\[
\hat{\mathbf{p}} = \mathbf{Vm}
\]

Since \( \hat{\mathbf{x}} \) is by definition the value of \( \mathbf{x} \) that minimizes the length of \( \mathbf{m} - \mathbf{Hx} \), it follows that \( \hat{\mathbf{m}} \) and \( \hat{\mathbf{e}} \) are orthogonal. The projection matrix \( \mathbf{V} \) must satisfy the following properties:

\[
\mathbf{VV}^T = \mathbf{I}_{\ell-1}
\]

\[
\mathbf{VH} = \mathbf{0}_{(\ell-1) \times \ell} \tag{II.30}
\]
\[ V^T V = I_k - H(H^TH)^{-1}H^T \]

The specifications on \( V \) may be completed by requiring that it be upper triangular with positive (non-null) diagonal elements. It can be verified [3] that if

\[ W = I - HK \quad \text{(II.31)} \]

then \( V \) is given by the following formulas

\[ V_{11}^2 = W_{11}, V_{ij} = 0 \text{ for } j < i, \quad V_{ij} = W_{ij}/V_{11} \quad \text{for } j = 2, \ldots, k \]

\[ V_{ii}^2 = W_{ii} - \sum_{k=1}^{i-1} V_{ki}^2 \text{ for } i = 2, \ldots, q \quad \text{(II.32)} \]

\[ V_{ij} = (W_{ij} - \sum_{k=1}^{i-1} V_{ki}V_{kj})/V_{ii} \text{ for } i = 2, \ldots, q, \quad j = i+1, \ldots, k \]

For example, in the case of three scalar measurements,

\[ m_1 = x + \epsilon_1 \]

\[ m_2 = x + \epsilon_2 \]

\[ m_3 = x + \epsilon_3 \]

with \( \underline{m} = Hx + \epsilon \) and \( H = \begin{bmatrix} 1, 1, 1 \end{bmatrix}^T \) the base vector is a scalar (\( x_b \)) and
\[ \hat{x} = K \hat{m} \]

where

\[ K = (H^T H)^{-1} H^T = \frac{1}{3} [1, 1, 1] \]

and

\[ \hat{x} = \frac{1}{3} (m_1 + m_2 + m_3) \]

as expected.

Now

\[ V^T V = I - HK = \frac{1}{3} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \]

and using the formulas above

\[ V = \begin{bmatrix} \sqrt{2/3} & -1/\sqrt{6} & -1/\sqrt{6} \\ 0 & 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \]

Figure II-18 shows the two-dimensional parity space and the corresponding measurement axes. The parity vector has components

\[ p_1 = \left( \frac{1}{\sqrt{6}} \right) (2\varepsilon_1 - \varepsilon_2 - \varepsilon_3) \]

\[ p_2 = \left( \frac{1}{\sqrt{2}} \right) (\varepsilon_2 - \varepsilon_3) \]

The magnitude of the parity vector \( p \) is, in general

\[ p^T p = \sum_{i=1}^{\ell} \frac{2}{k} \varepsilon_i^2 - \frac{1}{k} \sum_{i=1}^{\ell} \varepsilon_i^2 \quad \text{(II.33)} \]
FIGURE II-18. THE TWO-DIMENSIONAL PARITY SPACE AND MEASUREMENT AXES.

\[
\begin{pmatrix}
\frac{1}{\sqrt{6}}, & \frac{1}{\sqrt{2}} \\
\frac{-1}{\sqrt{6}}, & \frac{-1}{\sqrt{2}}
\end{pmatrix}
\]
If the sensor noise probability density function is uniform and if the error in each measurement for an unfailed component is assumed to be uniformly bounded, i.e., $|\varepsilon_i| < b$, then it can be shown [9] that

$$\delta_{\ell} = \max \left[ \frac{1}{\ell} \sum_{i=1}^{\ell} \varepsilon_i^2 - \frac{1}{\ell} \sum_{i=1}^{\ell} \varepsilon_i^2 \right]$$

$$= \begin{cases} \ell b^2, & \text{even} \\ \ell b^2 - \frac{b^2}{2}, & \text{odd} \end{cases}$$

(II.34)

The quantity $\delta_{\ell}$ may be used as a threshold whether a fault has occurred in the system.

When the error bounds are not uniform, then

$$\delta_{\ell} = \max(\mathbf{p}^T \mathbf{p}) = \sum_{i=1}^{\ell} b_i^2 - \frac{1}{\ell} \left( \min_{i=1}^{\ell} b_i \right)^2$$

(II.35)

A search algorithm is used to locate a minimum of $\Sigma b_i$ and, finally, $\delta_{\ell}$ is obtained from (II.35).

**II.6 The Rule Base**

Fault detection is based upon the relative magnitude of the maximum allowable error bound compared to the length of the parity vector, i.e., $\delta_{\ell}/\mathbf{p}^T \mathbf{p}$. A threshold condition is reached when $\mathbf{p}^T \mathbf{p} = \delta_{\ell}$. In general, both $\delta_{\ell}$ and $\mathbf{p}^T \mathbf{p}$ are not known accurately and, therefore, a fuzzy representation of the quantity $\delta_{\ell}/\mathbf{p}^T \mathbf{p}$ is most appropriate.

For fault detection of a leaky component, additional symptoms, such as the flow out of the make-up fluid tank, may enter the rule base.
The rule base and inference scheme are intended to provide a quantitative measure of the severity of the fault depending upon prevailing process conditions. To illustrate the structure of the rule base, consider the variable $\delta_{i}/p_{i}^{T}$. $\delta_{i}$ is estimated from (II.35). When the magnitude of the parity vector $p_{i}^{T}$ is in the neighborhood of $\delta_{i}$, then a fuzzy belief curve or membership function is associated with $\delta_{i}/p_{i}^{T}$. The membership function may be linear or nonlinear (for example, a cosine curve) in this particular range of the universe of discourse. When $\delta_{i}/p_{i}^{T}$ is very small, it is certain that $p_{i}^{T}$ is sufficiently greater than $\delta_{i}$ and, therefore, a fault condition exists. On the other hand, when $\delta_{i}/p_{i}^{T}$ is much larger than unity, then the length of the parity vector is too small to allow for a positive detection of a faulty component. The compositional rule of inference [13] is used to infer the degree of severity of the fault condition.

For leaky pipe fault detection, Rule 1 in Figure II-19 is used. The local symptom is $\delta_{i}/p_{i}^{T}$ where

$$\delta_{i} = \sum_{i=1}^{l} (b_{i})^{2} - \left(\frac{1}{l}\right) \min \left(\sum_{i=1}^{l} b_{i}\right)^{2}$$

and $l$ is the number of pipe segments. The $b_{i}$'s are normal error bounds and for $l = 3$, they are taken to be $b_{1} = 20.17$, $b_{2} = 4.14$, and $b_{3} = 0.71$. Therefore, $\delta_{i} = 444.89 - 83.43 = 361.47$. A value of $p_{i}^{T} = 25^{2}$ (lbs/hr)$^{2}$ was chosen as a threshold above which the uncertainty concerning a leaky pipe segment is reduced to zero. It corresponds to $A_{i} = \delta_{i}/p_{i}^{T} = 0.58$. The set of values of $A_{i}$ from 0.58 to 1.0 is considered fuzzy and an appropriate fuzzy belief curve is presented in the figure to simulate the uncertainty associated with $A_{i}$. The global symptom for leaky pipe fault detection is the ratio of...
A - RULE - BASE FOR LEAKY COMPONENT FAULT DETECTION

If $A_2$ is and $B_2$ is

then component failure severity is

$0.5^*$

$10$

$62$

NOTE: $A_2 = \frac{\delta_2}{pT_p}$ $B_2 = \frac{\Delta m}{\Delta m_{\text{max}}}$

B - RULE - BASE FOR STUCK VALVE COMPONENT FAULT DETECTION

If $A_s$ is

then component failure severity is

$0^*$

$1$

$0$

Note: $A_s = \frac{\delta_s}{pT_p}$
the flow rate through the make-up tank and the maximum possible flow rate, $\Delta m_{\text{max}}$, taken to be equal to 25 lbs/hr. For values of $\Delta m$ from 0.1 lbs/hr to 25 lbs/hr ($B_2 = 0.004$ to $B_2 = 1.0$), a leaky pipe segment is suspected and a fuzzy membership function for $B_2$, as shown in the figure, represents the uncertainty associated with this process.

Similar arguments are used for the stuck valve fault detection Rule 2 as depicted in Figure II-19. In this case, we rely exclusively on a single local symptom of $A_s = \delta S / p^T p$.

11.7 Fault Identification

The fault identification part of the program is intended to identify which component is faulted. Following [3], we propose a technique based upon the concurrent checking of the relative consistency of smaller size subsets of measurements.

We consider only scalar measurements (i.e., $n = 1$). If $m$ is the $l \times 1$ measurement vector, and $v_1, v_2, \ldots, v_l$ represent the failure directions in the $(n-1)$-dimensional parity space, then only $k$ measurements are considered ($1 < k < l$). The projection of the $(l-1)$-dimensional parity vector (generated from all $l$ measurements) on the $(k-1)$-dimensional subspace orthogonal to the subspace spanned by $v_{k+1}, v_{k+2}, \ldots, v_l$, is the parity vector directly generated from the measurements $m_1, m_2, \ldots, m_k$.

The consistency of large subsets of measurements can be determined in terms of the consistency of all possible smaller subsets of $(n+1) = 2$ measurements. For $m$ measurements, we form $t$ subsets, $S_1, S_2, \ldots, S_t$, of $(n+1) = 2$ measurements each with all possible combinations of $m_i, i = 1, l$. That is, $t$ is equal to $\binom{l}{2} = \frac{l!}{2!(l-2)!}$. The algorithm proceeds as follows:
For each of these subsets, $S_1, S_2, \ldots, S_t$, we calculate the projection matrix $V_i$ (which is $1 \times (n+1) = (1 \times 2)$-dimensional since the reduced parity space is one-dimensional) using the equations $V V^T = I$ and $V H = 0$. That is

$$V V^T = I + [a_1, a_2] \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = a_1^2 + a_2^2 = 1$$

$$V H = 0 + [a_1, a_2] \begin{bmatrix} 1 \\ 1 \end{bmatrix} = a_1 + a_2 = 0$$

From which

$$V = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix}$$

We form the matrix $\text{sgn}V_i$ which is a $(n+1) \times (n+1) = (2 \times 2)$ diagonal matrix, with elements +1 or -1 depending on the sign of the associated elements in $V_i$. In this case

$$\text{sgn}V_i = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

The error vector $\beta_i$ for each subset $S_i (i = 1, t)$ that contains, let us say, the measurements $m_i$ and $m_j$ is formed as

$$\beta_i = \begin{bmatrix} b_i \\ b_j \end{bmatrix}$$

where $b_i, b_j$ are the corresponding normal error bounds. Next, the threshold value $\theta_i$ is calculated for the corresponding subset $S_i$ from

$$\theta_i = V_i (\text{sgn}V_i) \beta_i = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} b_i \\ b_j \end{bmatrix}$$

$$= \frac{1}{\sqrt{2}} (b_i + b_j)$$
For each subset $S_i$, we calculate the corresponding reduced parity vector $p_i$ from a knowledge of $V_i$ and $m$.

Finally, $p_i$ is compared with $e_i$ to determine whether the component associated with the $v_i$ measurement axis has failed or not.

This procedure will allow not only the identification of a faulty sensor but also the estimation of the "best" value for the corresponding variable from measurements which remain fairly consistent.

Let us illustrate the approach with some graphical representations by considering a three-measurement vector $m = [m_1, m_2, m_3]$. The basic idea is again the following: Instead of checking to see whether the error vector $e$ stays within the polyhedral region of the measurement space by considering its projection on the parity space and specifying the normal (allowable) error bounds, the projections of the error vector on the various subspaces which constitute the measurement space may be taken and compared with the corresponding error bounds.

For $m = [m_1, m_2, m_3]$, the measurement space is a three-dimensional space and the normal error bounds define a hexahedron as shown in Figure II-20. The error vector $e$ is represented as $OP$. Instead of estimating the relative position of the error vector with respect to the hexahedron boundaries, we consider its projections $OP_1$, $OP_2$, $OP_3$ on the three planes which constitute the measurement space. The problem thus reduces to comparing the error vector projections to the error bounds in a two-dimensional setting with the error regions now being simple parallelograms. Refer to Figure II-21 which depicts the vector component $OP_3$ on the $(e_1, e_3)$ plane. For the case of the $(m_1, m_3)$ measurement set, the parity space is a line perpendicular to the unity vector.
FIGURE II-20  THE MEASUREMENT AND ERROR BOUND
SPACE FOR $\mathbf{m} = \{m_1, m_2, m_3\}$
FIGURE II-21  THE PROJECTION OF THE ERROR VECTOR ON \((\varepsilon_1, \varepsilon_2)\).
of this plane. The parity vector \( \tilde{p} \) is indeed the projection of \( \text{OP}_3 \) on this axis and the error bounds are specified from the projections of the parallelogram which, in turn, is a representation of the normal error bounds in the measurement space. The parallelogram is shown as HEAD or CBFG in the figure. The maximum allowable value for the parity vector \( \tilde{p} \) (under normal operating conditions) is specified in accordance with the projection of E or D on the parity space. From Figure II-21, the projection of point E on the parity space (a line in this case) is given as

\[
b_1 \cos 45^\circ + b_3 \cos 45^\circ = \frac{1}{\sqrt{2}} (b_1 + b_3) = \theta_{1,3}
\]

This parity space is perpendicular to the failure direction \( V_2 \) which is defined by the third axis \( e_2 \) of the measurement space and \( \tilde{p} \) is the projection of the parity vector \( p \) on this single-dimensional space.

Thus, the pictorial representation for the parity space in the case of three measurements is as shown in Figure II-22. A band of normal error bounds is drawn, as shown in the figure, along the \( V_2 \) axis and at a distance of \( \theta_{1,3} \) to the left and to the right of it. With this procedure, three similar bands are generated on the two-dimensional parity space with a width of \( 2\theta_1 \) each and their common intersection defines the normal bounds for the parity vector \( p \).

In the case of four scalar measurements \( \{m_1, m_2, m_3, m_4\} \), a total of six subsets are constructed as follows:

\[
S_1 = \{m_1, m_2\} \\
S_2 = \{m_1, m_3\} \\
S_3 = \{m_1, m_4\}
\]
FIGURE II-22  PICTORIAL REPRESENTATION OF THE PARITY SPACE FOR THREE MEASUREMENTS.
\( S_4 = \{m_2, m_3\} \)
\( S_5 = \{m_2, m_4\} \)
\( S_6 = \{m_3, m_4\} \)

The corresponding residuals are

\[
\begin{align*}
\quad m_1 - m_2 &= \eta_1 \\
\quad m_1 - m_3 &= \eta_2 \\
\quad m_1 - m_4 &= \eta_3 \\
\quad m_2 - m_3 &= \eta_4 \\
\quad m_2 - m_4 &= \eta_5 \\
\quad m_3 - m_4 &= \eta_6
\end{align*}
\]

For each one of the six subsets, the projection matrix is again

\[
V_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}
\]

and

\[
\text{sgn}V_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}
\]

The error vector will be \( \theta_i = [b_i, b_j]^T \) and the corresponding threshold value for subset \( S_i \) may be written as

\[
\theta_i = \frac{1}{\sqrt{2}} (b_i + b_j)
\]

where \( \theta_i \) is the projection of the vector \( b_i - b_j \) in measurement space on the parity space (a line again) and signifies the maximum normal error without a
component failure. Figure 11-23 depicts the relevant geometric relationships. Each $\tilde{p}_i$, corresponding to $S_i$, is calculated from the measurements $m_i$, $m_j$ and the projection matrix $V_i$ as

$$\tilde{p}_i = V_i \begin{vmatrix} m_i \\ m_j \end{vmatrix} = \frac{1}{\sqrt{2}} (m_i - m_j)$$

Component fault evidence is accumulated by comparing each $\tilde{p}_i$ with the corresponding $\theta_i$ value. The rule base and inference scheme are constructed, for the fault identification algorithm, in a similar manner as for the fault detection case.

Figure II-24 shows three rules for fault identification of a leak in three pipe segments considered in this study. The rule structure for the fault identification algorithm for a stuck valve is similar. Program listings and typical results are appended to this report.

II.8 Error Trending

Further evidence for component or sensor failures can be obtained from the error history accumulated. Suppose that after each measurement sampling period, the error between an individual sensor and the estimate from the collection of sensors is calculated, and that the last "$k$" of these errors are stored. A fuzzy variable may be defined as the average of these $k$ values scaled by the error bound, providing an indication of bias. Drift may be indicated by another fuzzy variable. Although detailed statistical analysis could be used to calculate the error trending variable, good performance is achieved, in a simulated environment, by simply taking the difference between the averages of the last $k/2$ error values and the first $k/2$ error values, and again scaling by the error bound.
FIGURE II-23 GEOMETRIC RELATIONSHIPS IN MEASUREMENT SPACE.
RULE - BASE FOR FAULT IDENTIFICATION OF LEAKY COMPONENT

If $S_1$ is and $S_2$ is and $S_3$ is then component one is failing with severity.

If $S_1$ is and $S_2$ is and $S_3$ is then component one is failing with severity.

If $S_1$ is and $S_2$ is and $S_3$ is then component one is failing with severity.
II.9 Fault Isolation

The fault isolation algorithm is provided with information about the fault type (sticking valve or leaky pipe segment) and the severity of the fault, and it decides as to the action to be taken to isolate the faulty component for repair and maintenance purposes. It issues a status report and appropriate commands to the maintenance module for further action.

Since there is no provision in the status for automatic shut-down or throttle valving, the severity of the fault will dictate the need for operator intervention. Thus, corrective measures can be taken by transmitting the FAIS report to the scheduling and maintenance modules.

A. Leaky Component Isolation

In the event of a leak anywhere along component 1, the series path of the thermal loop becomes inoperative. In order to isolate this component, the following action is recommended: close valves V58, V60, V61, V65, and V67. Although a fault in component 1 is critical due to its location in the loop topology, it may be possible to maintain the ECLSS heat exchanger within acceptable operating limits by closing the δ-valve and opening both the PV (parallel system valve) and SV (series option valve) devices. Such an action would maintain $T_{11}$ within ±2.5°F of 40°F and $\Delta T_1$ within the limit of 19°F.

In order to isolate component 2, in the event of a fault, valves V52, V54, V56, and δ must be closed. The ECLSS heat exchanger will become inoperative until repairs are completed. After isolating the faulty component, it is possible that the $\Delta T_2$ constraint can be maintained below 5°F as well as $\Delta T_{3-5}$ to stay within 40°F. The degraded mode of the system will be unable to maintain $T_{11}$ and $T_{12}$ within 2.5°F of 40°F and 70°F, respectively. When component 3 fails, it may be isolated by closing valves V61 and V62. The
fault severity is minimal in this case. The system controller will be able to maintain all constraints except for the $\Delta T_3 < 40^\circ F$ one, since HX - 3 will have to be shut down for faulty pipe repairs.

B. Sticking Valve Isolation

In most cases of faulty valving components, the system controller will be capable of maintaining the normal operation of the STCS at probably lower than optimal conditions.

In the event of a fault at V1, this component may be isolated by closing valves V59 and V60. Although the controller is still maintaining the constraint bounds, it will be operating in a suboptimal mode in that the loop flow rate will be greater than the quantity required to minimize total system energy demand. Valve V2 may be isolated by closing V67 and V68. The controller is again meeting constraint bounds while the system is operating in a suboptimal mode.

Component 3 is isolated by closing V53 and V54. The system will maintain most constraints with the exception of $T_{i-1}$. Deviations from the 2.5°F tolerance bound might occur but their extent will be limited.

Finally, component 4 may be isolated by closing valves V58, V59, SV, and V55.

II.10 Conclusions and Recommendations

An intelligent control algorithm has been developed for STCS fault diagnosis, fault detection and isolation based upon a combination of signal redundancy techniques and multivalued logic.

The methodology proposed uses innovative concepts of parity space representation and analytic redundancy in combination with multivalued logic.
techniques in order to evaluate real-time sensor data, assess the aggregate of fault symptoms and identify component faults.

The detailed algorithms developed provide valuable insight to the STCS control functions. They reveal the precise role and the desirable characteristics of sensors and transducers; they make clear the control functions of conventional means; they suggest ways for improving the selection of sensors and other system components; they guide the topological design of the STCS; they, finally, provide an indispensable tool for integrating normal and emergency control functions of the thermal loop.

The test cases examined illustrate the robustness, viability, and flexibility of the proposed approach.

Future work is required in order to incorporate real STCS test data into the intelligent controller structure, improve and refine the rule-based system, strengthen the interface between the conventional hierarchical controller and the fault diagnostics routines, and achieve a more effective integration of STCS and other subsystem functions. Specifically, it is recommended that:

- The intelligent controller be expanded to include other types of faults;
- The rule base be enhanced and refined with symptomatic evidence from the hierarchical controller performance;
- "Smart" sensor concepts be introduced into the algorithmic development based on adaptive estimation techniques;
- Multiple faults be considered and techniques developed to provide "safe" and reliable estimates of system variables under off-normal conditions;
• Develop techniques for optimizing the design of sensing, transducing and, actuation devices;
• Incorporate actual error statistics of STCS sensors into the intelligent controller design;
• Investigate the optimum transfer of resources and information between the STCS and other functional subsystems of the space station common module.

The intelligent controller must be integrated into the subscale laboratory thermal control facility and a detailed schedule of testing procedures be developed in order to "train" the AI-based algorithms and improve the rule base. Adaptive and learning methods may be included in the intelligent scheme in order to assure robustness of performance and speed of convergence.
REFERENCES


APPENDIX A

USER DOCUMENTATION, PROGRAM LISTING, AND SIMULATION RESULTS
FOR LEAK DETECTION AND IDENTIFICATION SIMULATION STUDIES
The TCS Fault Diagnostics Program consists of two parts. The first part is called TCSITRL.FOR and generates the parameters. It is written in Fortran. The second part is called F.PRO and is the Fault Detection and Fault Identification program. F.PRO is written in Prolog.

In order to run the simulation program, the user should follow the following steps:

1. The program TCSITRL requires a math coprocessor (like one from the 8087 series) in order to support the FORTRAN.EXE file, since TCSITRL.EXE was compiled by the PROFESSIONAL FORTRAN COMPILER.

2. F.PRO can be executed without the coprocessor.

3. Insert the floppy disk into the driver of an IBM PC, XT, AT. (or compatible machine)

4. Programs TCSITRL AND F.PRO can be executed either separately or sequentially.

5. The procedure to run the simulation program is detailed below:

A. The procedure to run the first program:

<table>
<thead>
<tr>
<th>User input</th>
<th>Computer response</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) ITLCTRL &lt;CR&gt;</td>
<td>(1) NAME OF STATE DATA FILE ?</td>
</tr>
<tr>
<td>(2) LEAK1.DAT &lt;CR&gt;</td>
<td>(2) OUTPUT FILE NAME ?</td>
</tr>
<tr>
<td>(3) 01.DAT &lt;CR&gt;</td>
<td>(3) EXECUTION TERMINATED : 0</td>
</tr>
</tbody>
</table>

The state data file should be one of the following:

\{ LEAK1.DAT, LEAK2.DAT, LEAK3.DAT, STUCK2.DAT, STUCK3.DAT, STUCK4.DAT \}

The output file name can be any name allowed in Fortran.

Note: The output file name cannot be the same as that found in the disk directory. To observe the contents of the output file:

Type " TYPE 01.DAT <CR> ", Then the parameters will appear on
the screen.

B. The procedure to run the second program:

<table>
<thead>
<tr>
<th>User input</th>
<th>Computer response</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) PROLOG &lt;CR&gt;</td>
<td>(1) INITIAL LOGO</td>
</tr>
<tr>
<td>(2) &lt;CR&gt;</td>
<td>(2) (MENU)</td>
</tr>
<tr>
<td>(3) F</td>
<td>(3) (MENU)</td>
</tr>
<tr>
<td>(4) L</td>
<td>(4) FILENAME:</td>
</tr>
<tr>
<td>(5) F.PRO</td>
<td>(5) (PGM)</td>
</tr>
<tr>
<td>(6) R</td>
<td>(6) Please Type in the input file name &gt;&gt;</td>
</tr>
<tr>
<td>(7) L1.DAT &lt;CR&gt; (For example)</td>
<td>(7) (OUTPUT SCREEN)</td>
</tr>
</tbody>
</table>

The input file name should be either one of the following or the output file name of the TCSITRL.

{ L1.DAT, L2.DAT, L3.DAT, S2.DAT, S3.DAT, S4.DAT }

(8) Type "R" to continue. (Go back to (6)) or Type "Q" to quit.
PROGRAM ITLCTRL (INPUT, OUTPUT)

REAL FMI (10), FMF (10), TC (8), DP (5), V (4), ERBL (3), ERBS (4)
REAL CL (2, 3), CS (2, 4), ML (3), MS (4), SL (3, 2), SS (6, 2)
REAL BL (3, 2), BS (6, 2), PL (3), PS (6), STL (3), STS (6)
REAL DL (3), DS (6), DTL (3), DTS (4), CV (5), MSC (4)
REAL VT, TMO, TM1, TM2, TM3, TM4, TM5, TM6, TM7, TM8, TM9, TM10
REAL MAXERBL, MAXERBS, MINSUML, MINSUMS
REAL DLMIN, DSNMIN, SQRBL, SQRBS, MINERBL, MINERBS

CHARACTER *20 FILNAM

DATA (CV (1), 1=1,5) /.00000816, .00000214, .000248 /
DATA (ERBL (1), 1=1,3) /4.14, 20.67, .71 /
DATA (ERBS (1), 1=1,4) /4*.002 /
DATA VT, DMMAX/1000, 25.0 /

INPUT DATA FROM THE SENSORS

READ (*,*) (FMI (I), I=1,10)
READ (*,*) (TC (1), I=1,8)
READ (*,*) (DP (I), I=1,5)
READ (*,*) (V (1), I=1,4)
READ (*,*) DM
READ (*,*) (FMF (1), I=1,10)

FIND THE PARITY VECTOR FOR LEAKY PIPE ML (3)

DP (1) = CV (1) * FMI (6) * FM1 (6)
DP (2) = CV (2) * FMI (7) * FM1 (7)
DP (3) = CV (3) * FMI (8) * FM1 (8)
DP (4) = CV (4) * FMI (9) * FM1 (9)
DP (5) = CV (5) * FMI (10) * FM1 (10)
WRITE (*,*) 'DP (1), 1=1,5 ', DP (1), DP (2), DP (3), DP (4), DP (5)
TMO=FMI (1) + FMI (2)
TM1= FMI (7) + SQRT (DP (2) / CV (2))
TM2= (TC (5) - TC (4)) / (TC (6) - TC (4))
CL (1, 1) = (1./2.) * (TM1 * TM2)
TM3= FMI (8) + FMI (9) + FMI (10)
TM4= (TC (7) - TC (6)) / (TC (8) - TC (6))
TM5= SQRT (DP (3) / CV (3))
TM6=SQRT (DP (4) / CV (4)) + SQRT (DP (5) / CV (5))
CL (2, 1) = .5 * (TM3+TM5+TM6) * TM4
CL (1, 2) = FMI (5)
TM7= FMI (6) + SQRT (DP (1) / CV (1))
TMB= (TC (2) - TC (1)) / (TC (3) - TC (1))
CL (2, 2) = .5 * TM7 * TM8
CL (1, 3) = FMI (8)
CL (2, 3) = TM5
DO 30 I=1,3
ML (I) = CL (1, I) - CL (2, I)
WRITE (*,*) 'ML (1), 1=1,3: ', ML (1), ML (2), ML (3)

DISTANCE CALCULATION

N=0
DO 50 K=1,3
KK=K+1
DO 40 J=KK,3
N=N+1
I=N
SL (1, 1) = ML (K)
BL (1, 1) = ERBL (K)
SL (1, 2) = ML (J)
BL (1, 2) = ERBL (J)
CONTINUE
CONTINUE
DO 60 L=1,3
STL (L) = 0
PL (L) = ABS (SL (L, 1) - SL (L, 2))
IF (PL (L) . GT. .00025) GO TO 90
PL (L) = .00025
90
DL (L) = (BL (L, 1) + BL (L, 2)) / PL (L)
IF (DL (L) . GE. 1.) GO TO 60
STL (L) = 1.
CONTINUE

dtl (1) = STL (1) + STL (2)
dtl (2) = STL (1) + STL (3)
dtl (3) = STL (2) + STL (3)
WRITE (*,*) 'DL (1) , I=1,3: ' , DL (1), DL (2), DL (3)
N = 3
CALL MIN (N, ERBL, MINSUO)
WRITE (*,*) 'MINSUML: ' , MINSUML

C
FIND PARITY VECTOR FOR STUCK VALVE MS (4)

CS (1, 1) = 1 - FM1 (7) / TMO
CS (1, 2) = 1 - TM3 / TMO
CS (1, 3) = FM1 (4) / FM1 (5)
CS (1, 4) = 1 - FM1 (5) / TMO
TM10 = FMF (8) + FMF (9) + FMF (10)
TM9 = FMF (1) + FMF (2)
CS (2, 1) = 1 - FMF (7) / TM9
CS (2, 2) = 1 - TM10 / TM9
CS (2, 3) = FMF (4) / FMF (5)
CS (2, 4) = 1 - FMF (5) / TM9
DO 130 I = 1, 4
MSC (I) = ABS (CS (1, I) - CS (2, I))
IF (MSC (I) . EQ. 0) THEN
MS (I) = 1.0E - 8
ELSE
MS (I) = 1 - ABS ((CS (1, I) - CS (2, I)) / (V (I) / VT))
END IF
130 CONTINUE
WRITE (*,*) 'MS (I) , I=1,4: ' , MS (1), MS (2), MS (3), MS (4)

C
CALCULATION THE DISTANCE OF PARITY SPACE

N = 0
DO 150 K = 1, 4
KK = K + 1
DO 140 J = KK, 4
N = N + 1
I = N
SS (1, 1) = MS (K)
BS (1, 1) = ERGS (K)
SS (1, 2) = MS (J)
BS (1, 2) = ERBS (J)
140 CONTINUE
150 CONTINUE
DO 160 L = 1, 6
STS (L) = 0
PS (L) = ABS (SS (L, 1) - SS (L, 2))
IF (PS (L) . GT. .00025) GO TO 190
PS (L) = 1E - 8
190 DS (L) = (BS (L, 1) + BS (L, 2)) / PS (L)
IF (DS (L) . GE. 1.) GO TO 160
STS (L) = 1.
160 CONTINUE
DTS (1) = STS (1) + STS (2) + STS (3)
DTS (2) = STS (1) + STS (4) + STS (5)
DTS (3) = STS (2) + STS (4) + STS (6)
DTS (4) = STS (3) + STS (5) + STS (6)
WRITE (*,*) 'DS (1) , I=1,4: ' , DS (1), DS (2), DS (3), DS (4)
A FORTRAN PROGRAM TO FIND A COMBINATION OF ORDERED NUMBERS TO RESULT A MINIMUM

SUBROUTINE MIN(N,B,MINSUM)
PARAMETER (NDI=10)
REAL B (NDI),MINSUM,TEMP,MINWOS,MINWIS,SWITCH,LEFT,RIGHT
REAL MINSUM1,MINSUM2
INTEGER N

ORDERING THE RANDOM INPUT SEQUENCE

DO 12 J=2,N
A=B (J)
DO 11 I=J-1,1,-1
IF (B (I).GT.A) GO TO 13
B(I+1)=B (I)
CONTINUE
I=0
B(I+1)=A
CONTINUE

WRITE (*,*) 'D (I), I=1,N: ' WRITE (*,*) (B (I), I=1,N)

THIS IS A FORTRAN PROGRAM TO FIND A COMBINATION OF ORDERED NUMBERS TO RESULT A MINIMUM

SUBROUTINE MIN(N,ERBS,MINSUMS)
WRITE (*,*) 'MINSUMS: ',MINSUMS
SQRBL=ERBL (1)**2+ERBL (2)**2+ERBL (3)**2
SQRBS=0.
DO 170 I=1,4
SQRBS=SQRBS+ERBS (1)*ERBS (I)
MINERBL=(1./3.)*(MINSUML*MINSUML)
DELTAL=SQRBL-MINERBL
MINERBS=(1./4.)*(MINSUMS*MINSUMS)
DELTAS=SQRBS-MINERBS
WRITE (*,*) 'SQRBL,MINERBL,DELTAL: ',SQRBL,MINERBL,DELTAL
WRITE (*,*) 'SQRBS,MINERBS,DELTAS: ',SQRBS,MINERBS,DELTAS
PPTL=ML (1) **2+ML (2) **2+ML (3) **2
PPTS=MS (1) **2+MS (2) **2+MS (3) **2+MS (4) **2
WRITE (*,*) 'DELTAL,DELTAS,PPTL,PPTS: ',DELTAL,DELTAS,PPTL,PPTS
IF (PPTL.EQ.0) PPTL=.00025
IF (PPTS.LT.1.0E-8) PPTS=1.0E-8
ALEAK=DELTAL/PPTL
ALEAK=ALEAK/1.2
IF (ALEAK.GT.1.2) ALEAK=1.0
BLEAK=DM/MMAX
IF (BLEAK.GT.10.) BLEAK=10.
ASTUCK=DELTAS/PPTS
IF (ASTUCK.GT.1.) ASTUCK=1.
DLMAX=AMAX1 (DL (1) ,DL (2) ,DL (3))
WRITE (*,*) 'CL (1, I), I=1,3: ',CL (1,1) ,CL (1,2) ,CL (1,3)
WRITE (*,*) 'CL (2, I), I=1,3: ',CL (2,1) ,CL (2,2) ,CL (2,3)
WRITE (*,*) 'ML (I), I=1,3: ',ML (1) ,ML (2) ,ML (3)
WRITE (*,*) 'CS (1, I), I=1,4: ',CS (1,1) ,CS (1,2) ,CS (1,3) ,CS (1,4)
WRITE (*,*) 'CS (2, I), I=1,4: ',CS (2,1) ,CS (2,2) ,CS (2,3) ,CS (2,4)
WRITE (*,*) 'MSC (I), I=1,4: ',MSC (1) ,MSC (2) ,MSC (3) ,MSC (4)
WRITE (*,*) 'MS (I), I=1,4: ',MS (1) ,MS (2) ,MS (3) ,MS (4)
WRITE (*,*) 'DLEAK,BLEAK: ',ALEAK,BLEAK
WRITE (*,*) 'DL (1), I=1,3: ',DL (1) ,DL (2) ,DL (3)
WRITE (*,*) 'ASTUCK: ',ASTUCK
WRITE (*,*) 'DS (I), I=1,6: ',DS (1) ,DS (2) ,DS (3) ,DS (4) ,DS (5) ,DS (6)
END IF
END

THIS IS A FORTRAN PROGRAM TO FIND A COMBINATION OF ORDERED NUMBERS TO RESULT A MINIMUM

SUBROUTINE MIN(N,B,MINSUM)
PARAMETER (NDI=10)
REAL B (NDI),MINSUM,TEMP,MINWOS,MINWIS,SWITCH,LEFT,RIGHT
REAL MINSUM1,MINSUM2
INTEGER N

ORDERING THE RANDOM INPUT SEQUENCE

DO 12 J=2,N
A=B (J)
DO 11 I=J-1,1,-1
IF (B (I).GT.A) GO TO 13
B (I+1)=B (I)
CONTINUE
I=0
B (I+1)=A
CONTINUE

WRITE (*,*) 'D (I), I=1,N: ' WRITE (*,*) (B (I), I=1,N)
WRITE(*,*) 'N?'
READ(*,*) N
DO 10 I=1,N
WRITE(*,*) 'B[',I,']?'
READ(*,*) IB
B(I)= IB
10 CONTINUE
WRITE(*,*) '---------------------------'
SWITCH= 0
K= 1
15 IF (SWITCH.EQ.0) THEN
LEFT= 0
RIGHT= 0
DO 20 := 1,K
LEFT= LEFT + B(I)
20 CONTINUE
DO 30 I= K+1,N
RIGHT= RIGHT + B(I)
30 CONTINUE
WRITE(*,*) 'LEFT.SUM=',LEFT,'RIGHT.SUM=',RIGHT
IF (LEFT.LT.RIGHT) THEN
K= K+1
ELSE
SWITCH= 1
END IF
GO TO 15
ELSE
GO TO 45
END IF
WRITE(*,*) 'K=',K
45 MINWOS= ABS(LEFT-RIGHT)
SWITCHING OF MIDDLE NUMBERS
TEMP= B(K)
B(K)= B(K+1)
B(K+1)= TEMP
FINDING SUMS WITH THE MIDDLE NUMBERS SWITCHED
RIGHT= 0
LEFT= 0
DO 50 I= 1,K
LEFT=LEFT+B(I)
50 DO 60 I=K+1,N
RIGHT=RIGHT+B(I)
60 MINWIS= ABS(RIGHT-LEFT)
IF (MINWOS.LT.MINWIS) THEN
MINSUM1=MINWOS
ELSE
MINSUM1=MINWIS
END IF
IF (MINWIS.GT.MINWOS) THEN
WRITE(*,*) 'MIN WITHOUT SWITCH=',MINWOS
ELSE
WRITE(*,*) 'MIN WITH SWITCH=',MINWIS
END IF
WRITE(*,*) '---------------------------'
THIS IS AN ALTERNATE WAY OF FINDING THE MINIMUM
FOR N ODD OR EVEN, THERE IS SEPARATE SUMMING FACTOR
SWITCH THE MIDDLE NUMBERS BACK
TEMP= B(K+1)
B(K+1)= B(K)
B(K)= TEMP
IF (N.EQ.INT(N/2)*2) THEN
CALL NEVEN(N,MINSUM2,B)
ELSE
CALL NODD(N,MINSUM2,B)
END IF
IF (MINSUM1.LT.MINSUM2) THEN
  MINSUM=MINSUM1
ELSE
  MINSUM=MINSUM2
END IF

WRITE(*,*)'MINIMUM BY THE ALTERNATE METHOD=',MINSUM2
WRITE(*,*)'
RETURN
END

SUBROUTINE NEVEN(N,MINSUM,B)
N IS EVEN
PARAMETER(NDI=10)
INTEGER B(NDI)
MINSUM= 0
M= N/2
DO 10 I= 1,M
  MINSUM=MINSUM + (((-1)**(I+1))*(B(I)+B(N-I)))
RETURN
END

SUBROUTINE NODD(N,MINSUM,B)
N IS ODD
PARAMETER (NDI= 10)
INTEGER B(NDI)
MINSUM= B(1)
M= INT(N/2)
DO 10 I=1,M
  MINSUM= MINSUM + (((-1)**I)*(B(I+1)+B(N+1-I)))
RETURN
END
DATA FILE FOR SPACE STATION TCS COMMON MODULE
PROGRAM NAME TCSITRL.FOR DATAFILE NAME LEAK1
WITH Q1=2.0,Q2=1.5,Q3=2.0,Q4=1.0,Q5=1.0
THERE IS A FAULT IN COMPONENT NO 1 WITH LEAK 50#/hr
THERE IS NO HEAT CHANGE IN THE SYSTEM
DATA FILE FOR SPACE STATION TCS COMMON MODULE
PROGRAM NAME TCSITRL.FOR DATAFILE NAME LEAK2
WITH Q1=2.0,Q2=1.5,Q3=2.0,Q4=1.0,Q5=1.0
THERE IS A FAULT IN COMPONENT NO 2 WITH LEAK 20#/hr
THERE IS NO HEAT CHANGE IN THE SYSTEM
DATA FILE FOR SPACE STATION TCS COMMON MODULE
PROGRAM NAME TCSITRL.FOR DATAFILE NAME LEAK3
WITH Q1=2.0,Q2=1.5,Q3=2.0,Q4=1.0,Q5=1.0
THERE IS A FAULT IN COMPONENT NO 3 WITH LEAK 20#/hr
THERE IS NO HEAT CHANGE IN THE SYSTEM
DATA FILE FOR SPACE STATION TCS COMMON MODULE
PROGRAM NAME TCSITRL.FOR DATAFILE NAME STUCK2
WITH Q1I=2.0,Q2I=1.5,Q3I=2.0,Q4I=1.0,Q5I=1.0
AND Q1F=2.1,Q2F=1.6,Q3F=2.1,Q4F=1.1,Q5F=1.1
THERE IS A STUCK VALVE ON VALVE 2
DATA FILE FOR SPACE STATION TCS COMMON MODULE
PROGRAM NAME TCSITRL.FOR DATAFILE NAME STUCK3
WITH Q1I=2.0,Q2I=1.5,Q3I=2.0,Q4I=1.0,Q5I=1.0
AND Q1F=2.1,Q2F=1.6,Q3F=2.1,Q4F=1.1,Q5F=1.1
THERE IS A STUCK VALVE ON VALVE 3
DATA FILE FOR SPACE STATION TCS COMMON MODULE
PROGRAM NAME TCSITRL.FOR DATAFILE NAME STUCK4
WITH Q1I=2.0,Q2I=1.5,Q3I=2.0,Q4I=1.0,Q5I=1.0
AND Q1F=2.1,Q2F=1.6,Q3F=2.1,Q4F=1.1,Q5F=1.1
THERE IS A STUCK VALVE ON VALVE 4
FAULT DETECTION/IDENTIFICATION
THE SYSTEM IS ANALYZED FOR EITHER LEAKY PIPES OR STUCK VALVES

NOWARNINGS

DOMAINS

A, B, C, D, E, F, G = SYMBOL
U, V, W, X, Y, Z = REAL
REALLIST = ELEMENTS
ELEMENTS = REAL
M = INTEGER
FILE = DATA

PREDICATES

FUNCTION(A, X, Y).
REV_FUNCTION(B, X, Y).
DEG_OF_FF(X, A, B, Y).
DEG_OF_RFF(X, A, B, Y).
CAR(REALLIST, X).
CDR(REALLIST, REALLIST).
DELETE(X, REALLIST, REALLIST).
MINFIND(REALLIST, X).
LEAK_DETECT_RULE(M, A, B, C).
STUCK_VALVE_RULE(M, A, B).
PIPE_ID_RULE(M, A, B, C, D).
VALVE_ID_RULE(M, A, B, C, D, E, F, G).

INFER_LEAK_DETECT(M, X, Y).
INFER_STUCK_VALVE_DETECT(M, X).
INFER_PIPE_ID(M, X, Y, Z).

PIPE_FAULT_SCREEN.

SYSTEM_FAULT_FILE.

DELAY(M).

SCREEN_DISPLAY.

VALVE_FAULT_SCREEN.

GOAL

SYSTEM_FAULT_FILE.

CLASSES

FUNCTION(AL, X, Y):-
  X < 0.58,
  Y = 1.
FUNCTION(AL, X, Y):-
  X > 0.58,
  X <= 1.0,
  X2 = (3.14*(X-0.58))/(1-0.58),
  Y = (COS(X2) + 1) / 2.
FUNCTION(AL, X, Y):-
  X > 1.0,
  Y = 0.
FUNCTION(AS, X, Y):-
  X < 0.4,
FUNCTION (AS, X, Y) :-
  X > 0.4,
  X <= 1.0,
  X2 = (3.14*(X - 0.4)) / (1 - 0.4),
  Y = (COS(X2) + 1) / 2.

FUNCTION (AS, X, Y) :-
  X > 1.0,
  Y = 0.

FUNCTION (B, X, Y) :-
  X < 0.004,
  Y = 0.

FUNCTION (B, X, Y) :-
  X >= 0.004,
  X <= 0.9,
  X2 = (3.14*(X - 0.004)) / (0.90 - 0.004),
  Y = (SIN(X2) + 1) / 2.

FUNCTION (B, X, Y) :-
  X > 0.90,
  Y = 1.

FUNCTION (L, X, Y) :-
  X >= 0,
  X <= 10.0,
  X2 = (0.314*X),
  Y = (COS(X2) + 1) / 2.

FUNCTION (H, X, Y) :-
  X >= 0,
  X <= 10.0,
  X2 = (0.314*X),
  Y = (SIN(X2) + 1) / 2.

REV_FUNCTION (L, X, Y) :-
  Y >= 0,
  Y <= 1.0,
  X = -1/Y.

REV_FUNCTION (H, X, Y) :-
  Y >= 0,
  Y <= 1.0,
  X = Y.

LIST OPERATIONS /*

CAR ([HEAD], X) :-
  HEAD = X.

CDR ([TAIL], TAIL) :- !.

DELETE (_, [], []) :- !.
DELETE (X, L, L1) :-
  CAR (L, X),
  CDR (L, L1),
  !.
DELETE (X, [YL1], [YL2]) :-
  DELETE (X, L1, L2).
MINFIND([X[]], MIN):-
    MIN = X.
MINFIND([X,Y[]], MIN):-
    X <= Y,
    MIN = X.
MINFIND([X,Y[]], MIN):-
    Y <= X,
    MIN = Y.
MINFIND([X,Y,ZW], MIN):-
    Y <= X,
    MINFIND([Y,ZW], MIN).
MINFIND([X,Y,ZW], MIN):-
    X <= Y,
    U = Y,
    DELETE(U, [X,Y,ZW], M),
    MINFIND(M, MIN).

/* THE RULES */

LEAK_DETECT_RULE(1, AL, B, H):-!.
STUCK_VALVE_RULE(1, AS, H):-!.
PIPE_ID_RULE(1, L, L, H, H):-!.
PIPE_ID_RULE(2, L, H, L, H):-!.
PIPE_ID_RULE(3, H, L, L, H):-!.

VALVE_ID_RULE(1, L, L, L, H, H, H, H):-!.
VALVE_ID_RULE(2, L, H, H, L, L, H, H):-!.

/* THE INFERENCE */

INFER_LEAK_DETECT(I, X1, X2):-
    LEAK_DETECT_RULE(I, 11, 12, 01),
    IA = 11, IB = 12, XA = X1, XB = X2, OA = 01,
    DEG_OF_FF(XA, FUNCTION, IA, D1),
    DEG_OF_FF(XB, FUNCTION, IB, D2),
    DA = D1, DB = D2,
    MINFIND([DA, DB], MIN1),
    MINA = MIN1,
    DEG_OF_RFF(X, REV_FUNCTION, OA, MINA),
    X3 = X*100,
    NL, WRITE(" SEVERITY OF LEAK"), NL,
    WRITE(" IN PIPES IS"), NL,
    WRITE("", X3, " "), NL.

INFER_STUCK_VALVE_DETECT(I, X1):-
    STUCK_VALVE_RULE(I, 11, 01),
    IA = 11, XA = X1, OA = 01,
    DEG_OF_FF(XA, FUNCTION, IA, D1),
    DA = D1,
    DEG_OF_RFF(X, REV_FUNCTION, OA, DA),
    X3 = X*100,
    NL, WRITE(" SEVERITY OF STUCK"), NL,
    WRITE(" IN VALVE IS"), NL,
    WRITE("", X3, " "), NL.

INFER_PIPE_ID(I, X1, X2, X3):-
    PIPE_ID_RULE(I, 11, 12, 13, 01),
    IA = 11, IB = 12, IC = 13, XA = X1, XB = X2, XC = X3, OA = 01,
    DEG_OF_FF(XA, FUNCTION, IA, D1),
    DEG_OF_FF(XB, FUNCTION, IB, D2),
    DEG_OF_FF(XC, FUNCTION, IC, D3),
FER VALVE ID(I,X1,X2,X3,X4,X5,X6):-
  VALVE_ID_RULE(I,11,12,13,14,15,16,01),
  IA = 11, IB = 12, IC = 13, ID = 14, IE = 15, IG = 16,
  XA = X1, XB = X2, XC = X3, XD = X4, XE = X5, XF = X6, OA = 01,
  DEG_OF_FF(XA,FUNCTION,IA,D1),
  DEG_OF_FF(XB,FUNCTION,IB,D2),
  DEG_OF_FF(XC,FUNCTION,IC,D3),
  DEG_OF_FF(XD,FUNCTION,ID,D4),
  DEG_OF_FF(XE,FUNCTION,IE,D5),
  DEG_OF_FF(XF,FUNCTION,IG,D6),
  DA = D1, DB = D2, DC = D3, DD = D4, DE = D5, DF = D6,
  MINFIND([DA,DB,DC],MIN1),
  MINA = MIN1,
  DEG_OF_RFF(X,REV_FUNCTION,OA,MINA),
  X7 = 100*X,
  WRITE(" VALVE ",I," FAILURE:") ,NL,
  WRITE(" ",X7," %"),NL.

ROUTINES FOR ANALYSIS */

PE_FAULT_SCREEN:-
  WRITE("INPUT X1 >>"),
  READREAL(X1),
  WRITE("INPUT X2 >>"),
  READREAL(X2),
  INFER_LEAK_DETECT(1,X1,X2),
  WRITE("INPUT X1 >>"),
  READREAL(X3),
  WRITE("INPUT X2 >>"),
  READREAL(X4),
  WRITE("INPUT X3 >>"),
  READREAL(X5),
  INFER_PIPE_ID(1,X3,X4,X5),
  INFER_PIPE_ID(2,X3,X4,X5),
  INFER_PIPE_ID(3,X3,X4,X5).

PIPE_FAULT_FILE:-
  OPENREAD(DATA,"TESTDATA"),
  READDEVICE(DATA),
  READREAL(X1),
  READREAL(X2),
  INFER_LEAK_DETECT(1,X1,X2),
  READREAL(X3),
  READREAL(X4),
  INFER_PIPE_ID(1,X3,X4,X5),
  INFER_PIPE_ID(2,X3,X4,X5),
  INFER_PIPE_ID(3,X3,X4,X5).*

LVE_FAULT_SCREEN:-
  WRITE("INPUT X1 >>"),
  READREAL(X1),
  WRITE("INPUT X2 >>"),
  READREAL(X2),
  WRITE("INPUT X3 >>"),
  READREAL(X3),
  WRITE("INPUT X4 >>"),
  READREAL(X4),
WRITE ("INPUT X5 >>"),
READREAL(X5),
WRITE ("INPUT X6 >>"),
READREAL(X6),
INFER_VALUE_ID(1,X1,X2,X3,X4,X5,X6),
INFER_VALUE_ID(2,X1,X2,X3,X4,X5,X6),
INFER_VALUE_ID(3,X1,X2,X3,X4,X5,X6),
INFER_VALUE_ID(4,X1,X2,X3,X4,X5,X6).

DELAY(N):- N > 0, !, N1 = N - 1, DELAY(N1).
DELAY(_).

SCREEN_DISPLAY:-
MAKEWINDOW(1,112,"",0,0,25,80),
CLEARWINDOW,
MAKEWINDOW(1,7,7,"",1,55,4,10),
TIME(H,M,S,HS),
WRITE(H," ":",M," ":",S),
MAKEWINDOW(1,28,3,"",1,30,4,20),
WRITE("FAULT DIAGNOSTICS"),NL,
WRITE(" TRIGGER"),
MAKEWINDOW(2,28,3,"LEAKY PIPE",5,5,7,22),
MAKEWINDOW(3,28,3,"STUCK VALVE",5,53,7,22),
MAKEWINDOW(4,28,3,"",13,5,9,23),
MAKEWINDOW(5,28,3,"",13,53,12,23).

SYSTEM_FAULT_FILE:-
WRITE("PLEASE TYPE IN THE INPUT FILE NAME >> "),
READLN(FNAME),
OPENREAD(DATA,FNAME),
SCREEN_DISPLAY,
READDEVICE(DATA),
READREAL(X1),
READREAL(X2),
READREAL(X3),
READREAL(X4),
READREAL(X5),
READREAL(X6),
READREAL(X7),
READREAL(X8),
READREAL(X9),
READREAL(X10),
READREAL(X11),
READREAL(X12),
SHIFTWINDOW(2),
INFER_LEAK_DETECT(1,X1,X2),
SHIFTWINDOW(3),
INFER_STUCK_VALVE_DETECT(1,X6),
SHIFTWINDOW(4),
INFER_PIPE_ID(1,X3,X4,X5),
INFER_PIPE_ID(2,X3,X4,X5),
INFER_PIPE_ID(3,X3,X4,X5),
SHIFTWINDOW(5),
INFER_VALUE_ID(1,X7,X8,X9,X10,X11,X12),
INFER_VALUE_ID(2,X7,X8,X9,X10,X11,X12),
INFER_VALUE_ID(3,X7,X8,X9,X10,X11,X12),
INFER_VALUE_ID(4,X7,X8,X9,X10,X11,X12).
L1.DAT

1224668564411
00918643935804
001797240216245
0.

0.
0.
0.
0.
0.
0.
0.
0.
L2.DAT

.7560284978527
.8
.0006406867057788
10.
.0005521156224114
1.10.
10.
10.
10.
10.
10.
L3.DAT

7355530030024
3
).
2579366270329
.309883017525
.
).
).
).
).
).
).
).
.

A-22
S2.DAT

1.
0.
10.
10.
10.
0.0001637868533711
1.011729913677E-7
10.
10.
1.011744112321E-7
1.011865074554E-7
10.
S3.DAT

.  
.  0.  
.  0.  
.  0.  
.  00311590499832  
.  0.  
.  000001394846923726  
.  0.  
.  000001395488382867  
.  0.  
.  00000139742038376
S4.DAT

1.
0.
10.
10.
10.
0.00001793331304272
10.
10.
1.058655457792E-7
10.
1.058692393009E-7
1.058690873783E-7
TIME: 22:52:9

FAULT DIAGNOSTICS TRIGGER

--- LEAKY PIPES ---

SEVERITY OF LEAK IN PIPES IS: 100 %

PIPE 1 FAILURE: 50.079632646 %
PIPE 2 FAILURE: 0.000063413623025 %
PIPE 3 FAILURE: 0.000063413623025 %

--- STUCK VALVE ---

SEVERITY OF STUCK VALVE IS: 0.000063413623025 %

VALVE 1 FAILURE: 0.000063413623025 %
VALVE 2 FAILURE: 0.000063413623025 %
VALVE 3 FAILURE: 0.000063413623025 %
VALVE 4 FAILURE: 0.000063413623025 %

PRESS THE SPACE BAR
**FAULT DIAGNOSTICS TRIGGER**

---

**LEAKY PIPES**

Severities of leak in pipes:

- **PIPE 1 FAILURE:**
  0.0000063413623025 %
- **PIPE 2 FAILURE:**
  50.079632646 %
- **PIPE 3 FAILURE:**
  0.000063413623025 %

---

**STUCK VALVE**

Severities of stuck valve:

- **VALVE 1 FAILURE:**
  0.000063413623025 %
- **VALVE 2 FAILURE:**
  0.000063413623025 %
- **VALVE 3 FAILURE:**
  0.000063413623025 %
- **VALVE 4 FAILURE:**
  0.000063413623025 %

---

Press the space bar.
--- FAULT DIAGNOSTICS TRIGGER ---

**LEAKY PIPES**

SEVERITY OF LEAK IN PIPES IS: 67.24062757%

PIPE 1 FAILURE: 0.000063413623025%
PIPE 2 FAILURE: 0.000063413623025%
PIPE 3 FAILURE: 50.079632646%

**STUCK VALVE**

SEVERITY OF STUCK VALVE IS: 0.000063413623025%

VALVE 1 FAILURE: 0.000063413623025%
VALVE 2 FAILURE: 0.000063413623025%
VALVE 3 FAILURE: 0.000063413623025%
VALVE 4 FAILURE: 0.000063413623025%

PRESS THE SPACE BAR
--- FAULT DIAGNOSTICS TRIGGER ---

--- LEAKY PIPES ---

SEVERITY OF LEAK IN PIPES IS:
0%

PIPE 1 FAILURE:
0.000063413623025%

PIPE 2 FAILURE:
0.000063413623025%

PIPE 3 FAILURE:
0.000063413623025%

--- STUCK VALVE ---

SEVERITY OF STUCK VALVE IS:
100%

VALVE 1 FAILURE:
0.000063413623025%

VALVE 2 FAILURE:
50.079632646%

VALVE 3 FAILURE:
0.000063413623025%

VALVE 4 FAILURE:
0.000063413623025%

PRESS THE SPACE BAR
FAULT DIAGNOSTICS TRIGGER

---

**LEAKY PIPES**

SEVERITY OF LEAK IN PIPES IS: 0%

- PIPE 1 FAILURE: 0.000063413623025%
- PIPE 2 FAILURE: 0.000063413623025%
- PIPE 3 FAILURE: 0.000063413623025%

---

**STUCK VALVE**

SEVERITY OF STUCK VALVE IS: 100%

- VALVE 1 FAILURE: 0.000063413623025%
- VALVE 2 FAILURE: 0.000063413623025%
- VALVE 3 FAILURE: 50.079632646%
- VALVE 4 FAILURE: 

PRESS THE SPACE BAR
FAULT DIAGNOSTICS TRIGGER

--- LEAKY PIPES ---

SEVERITY OF LEAK IN PIPES IS:
0%

PIPE 1 FAILURE:
0.000063413623025%

PIPE 2 FAILURE:
0.000063413623025%

PIPE 3 FAILURE:
0.000063413623025%

--- STUCK VALVE ---

SEVERITY OF STUCK VALVE IS:
100%

VALVE 1 FAILURE:
0.000063413623025%

VALVE 2 FAILURE:
0.000063413623025%

VALVE 3 FAILURE:
0.000063413623025%

VALVE 4 FAILURE:
50.079632646%

PRESS THE SPACE BAR
USER DOCUMENTATION

This Documentation consists of three parts. Part (A) is the USER MANUAL which guides the user as to how to run the program. Part (B) is the INITIAL CONDITION FILE DOCUMENTATION. It provides information for constructing the initial condition file and explains the states in the simulation program. Part (C) is the INPUT and OUTPUT DATA DOCUMENTATION which describes the input and output variables in the program.

The simulation program is based on a Nonadaptive Runge-Kutta integration routine. This simulation program takes either a continuous or a discrete dynamics description of the plant.

Note: All the states (refer to part (B)) and all the outputs (refer to part (C)) may be shown graphically. The procedure for graphing the results is the following:

1. Set up the plotter in your plotting terminal.
2. Run the simulation program and obtain the desired states and outputs. The results will be saved in FOR099.DAT.
3. Quit the Simulation program.
4. Call your plotting package which uses FOR099.DAT as an input file to graph the desired variable.
PART (A) - USER MANUAL

The Simulation program consists of two parts. The first one is called TRESPTN and is the main program. The second one is called TCSSIM2. In order to run the simulation program the user should follow the following steps:

$ FOR TRESPTN
$ FOR TCSSIM2
$ LINK TRESPTN, TCSSIM2
$ RUN TRESPTN

(Note: FOR TRESPTN means compile program TRESPTN into object code. FOR TCSSIM2 does likewise for TCSSIM2. LINK TRESPTN, TCSSIM2 means link the two object codes together into one execution code. RUN TRESPTN means run the execution code.)

The Simulation program is run following the steps:

(*) - OPEN (O) OR CLOSED (C) - LOOP SYSTEM?:

  type:  O for open loop system
  or
  type:  C for closed loop system

  NAME OF INITIAL CONDITION FILE ("N" FOR NONE)
  type:  IF2O for open loop case.

       (Note: This is just a typical initial condition file, you can set up your own initial condition file by referring to "PART (B)".)
  or
  type:  HOTST for closed loop case.
or

type: N for no initial condition file

(Note: If you type N, then the following questions will show on the
screen.)

- HOW MANY STATES?

type: 17 for open loop case

or

type: 27 for closed loop case

- ENTER INITIAL STATES:?

(Note: Now you need to enter all the initial states, please refer
to "PART (B)"

- DISCRETE FUNCTION REQUIRED? (Y or N)

  type: N for open loop case

  or

  type: Y for closed loop case

(**) - HOW MANY STATES TO BE PLOTTED?

  type: 6

  (Note: This is just an example. You can plot from 1 to 17 states
  for an open loop case or you can plot from 1 to 27 states for
closed loop case.)

- WHICH ONES?

  type: 1  6  7  10  12  14

  (Note: By referring to "PART B", you can find the meaning of
all the states in both cases.)
- PLOT HOW MANY OUTPUTS?
  type: 3
  (Note: This is just an example, you can plot from 1 to 15 outputs for both cases.)

- WHICH ONES?
  type: 1 2 3
  (Note: By referring to "PART (C)", you can get all the information about the outputs.)

(***) - RUN TIME? (DEF = 5 min)
  type: / for default value
  or
  type: 10 if you want to run the simulation program for 10 mins.
  (Note: The time constant for Heat Exchanger 1 (Q1) is 30 mins.
    The time constant for Heat Exchanger 2 (Q2) is 2 mins.
    The time constant for Heat Exchanger 3-5 (Q3-Q5) is 8 mins.

The information about time constants could give you an idea as to how long you need to run the simulation program in order to get a steady-state solution for each Heat Exchanger).

- PRINTING TIME INTERVAL ON SCREEN? (DEF = 10 sec.)
  type:/ for default value
  or
  type: 1 for one min., then you can observe the result every minute on the screen.
- CONTROL TIME INTERVAL? (DEF = 5 sec.)
  type:/ for default value
  
  (Note: This control time interval is for the closed loop case only.)

- PLOTTING TIME INTERVAL? (DEF = 1 sec.)
  type:/ for default value
  or
  type: 0.08333 for 0.08333 min. (5 sec.)

  (Note: If you type 0.08333, it means that you are requesting output results every 0.08333 min. (5 sec.). For example, if you run the simulation for 10 min. and you choose a plotting time interval of 0.08333 min., then 600 points for each output file will be shown. Therefore, you can use these output data points for plotting. Check for the maximum number of points allowed in your GRAPHICS PACKAGE. You cannot exceed the maximum points allowed.

- SAMPLE PERIOD? (DEF = 0.25 sec.)
  type:/ for default value

After entering all the information into the program, you will observe the following results on the monitor:

<table>
<thead>
<tr>
<th>STATES</th>
<th>(OUTPUTS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIME</td>
<td>1</td>
</tr>
</tbody>
</table>

B-6
After RUN TIME, the following option menu will appear on the screen. You can choose an option by entering the corresponding number.

- Enter 0 TO FILE ANSWERS
- 1 TO QUIT
- 2 TO RESTART
- 3 TO PICK NEW STATES
- 4 TO CHANGE THE SCALE

type: 0 for saving your results

(Note: If you need to plot the results, then it is necessary that you save the file. Therefore, type "0", and the results will be saved in file FOR099.DAT)

or

type: 1 to quit the program

or

type: 2 to restart your program, i.e., go to step (*)

or

type: 3 to pick some other new states, i.e., go to step (**)

or

type: 4 to change the scale of "RUN TIME", "PRINTING TIME INTERVAL ON SCREEN", etc., i.e., go to step (***)

(Note: If you choose "0", then after your results (states and outputs) have been saved in FOR099.DAT, the option menu will appear on the screen again. This time you might want to select "1" so you can quit the program. Therefore, you could use the graphics package to plot the results.)
PART (B) INITIAL CONDITION FILE DOCUMENTATION

Initial conditions of the TCS system states are required in order to run the program. There are two ways available to the user for entering initial conditions into the program:

- Set-up an initial condition file before running the simulation program.
  When interacting with the program, you will be asked to enter the name of the initial condition file. Type in the name.
- You may enter the initial conditions one by one as input while the program is running, i.e., the simulation program may be run on line without an initial condition file. To select this option, type "N" when the program requests for the initial condition file name. Following that type the initial conditions into the program.

The following illustrates the meaning of the Initial Condition File data:

EXAMPLE: IF20

1. FILE NAME: IF20 ("O" means open loop case)

   | 17 | 6 | 5
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2.45</td>
<td>2</td>
<td>2.8</td>
</tr>
<tr>
<td>.7303</td>
<td>0</td>
<td>.103</td>
</tr>
<tr>
<td>1.362398</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2.45</td>
<td>2</td>
<td>2.8</td>
</tr>
</tbody>
</table>

2. EXPLANATION:

   17: PROGRAM VARIABLE IS NX. NX designates the number of states in the open loop system.

   6: PROGRAM VARIABLE IS NU

   NU designates the number of control inputs in the open loop system.

   There are 6 control inputs in the simulation program.
VMO (for pump), VMU (for Mix valve)

VSIG (for Heat Sink), VDEL (for Heat Exchanger 1)

VBET (for Heat Exchanger 2), VGAM (for Heat Exchangers 3-5)

5: PROGRAM VARIABLE is NL

NL designates the number of Heat Flow parameters (disturbances) in the open loop system.

There are 5 Heat Flow parameters (disturbances) entering the Heat Exchangers in the simulation program, i.e., Q1, Q2, Q3, Q4 and Q5

17:  2.45 (Q1) 2(Q2) 2.8 (Q3) 1.4(Q4) 1.4(Q5) 89 (T0) 1362.398 (M0)

.7303 (MU) 0 (DMU) .163 (SIG) 0 (DSIG) .02432 (DEL) 0 (DDEL)

0 (BET) 0 (DBET)

.625 (GAM) 0 (DGAM)

(Note: DMU stands for the derivative of MU, etc.)

6: 1.362398 (VM0) 0 (VMU) 0(VSIG) 0(VDEL) 0(VBET) 0 (VGAM)

5: 2.45 (Q1) 2 (Q2) 2.8 (Q3) 1.4(Q4) 1.4(Q5)

(Note: MU is the control input for Mixing Valve

SIG is the control input for Heat Sink 2

DEL is the control input for Heat Exchanger 1

BET is the control input for Heat Exchanger 2

GAM is the control input for Heat Exchanger 3-5)
EXAMPLE: HOTST

1. FILE NAME: HOTST (closed loop case)

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>1</td>
<td>5</td>
<td>2.45</td>
<td>2</td>
<td>2.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.4</td>
<td>1.4</td>
<td>89</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1362.398</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.7303</td>
<td>0</td>
<td>.103</td>
<td>0</td>
<td>.02432</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>.625</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1362.398</td>
<td>1.362398</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

2. DESCRIPTION: This initial condition file is used to test the situation from a maximum heat load to a small heat load.

3. EXPLANATION:

27: PROGRAM VARIABLE IS NX

NX designates the number of states in the closed loop system.

1: PROGRAM VARIABLE IS NU

NU designates the number of control inputs in the closed loop system.

There is only one control input, i.e., M0C (for pump motor speed).

5: PROGRAM VARIABLE IS NL

NL designates the number of Heat Flow parameters (disturbances) entering the Heat Exchangers in the closed loop system (Refer to IF2O).

27: 2.45(Q1) 2(Q2) 2.8(Q3) 1.4(Q4) 1.4(Q5) 89 (T0) 1362.398(M0)

.7303(MU) 0(DMU) .103(SIG) .02432 (DSIG) .2 (DEL) 0 (DDEL) 0(BET) 0(GAM)

1362.398(M0) 1.36298(VM0) 0(VMU) 0(VSIG) 0(VDEL) 0(VBET) 0(VGAM)

1 (TAU) 0 (DTAU) 0(VTAU)

1: 1362.398 (M0C)
5: 0.5 (Q1) (Q2) 0.5 (Q3) 0.5 (Q4) 0.5 (Q5)

* WARNING: When inputting values for the flow parameters (disturbances),

Q's, make certain that the condition Q1+Q2+Q3+Q4+Q5 > 1.6Q1, is not violated.)

(Note: TAU is the control input for Heat Sink 1.)

MOC is the overall command flow rate for the pump motor.)

EXAMPLE: COLDST

1. FILE NAME: COLDST (closed loop case)

```
27   1   5
0.5  0.5  0.5  0.5  0.5  90  340.599
.7303  0  0.103  0  0.02432  0  0  0 .625  0
.340599  0  0  0  0  0  1  0  0
340.599
2.45  2  2.8  1.4  1.4
```

2. DESCRIPTION: This initial condition file is used to simulate a loading condition

which starts from a light heating load and reaches maximum allowable

values for the Q's.

3. EXPLANATION: Refer to previous case.
PART (C) THE INPUT AND OUTPUT DATA DOCUMENTATION

This part describes the variables used as program input and output quantities. The user will be acquainted with the type of data to be entered into the simulation program and the available output data. These results could be used for plotting purposes.

INPUT VARIABLES:

- **ANSC**: Respond as to system configuration
  
  C - closed loop system
  
  O - open loop system

- **FILNAM**: Initial Condition File name

  1. You may enter the name of the Initial Condition file
     
     e.g., HOTST, IF20 (refer to INITIAL CONDITION FILE DOCUMENTATION)

     NOTE: The following input data are read from the initial condition
     file which is user constructed.

     - **NX**: Number of states.
     - **NU**: Number of control inputs
     - **NL**: Number of heat flow parameters
     - **SX(I)**: Initial states, for I = 1 to NX
     - **U(I)**: Control inputs for open loop system, for I = 1 to NU
     - **UC(I)**: Control inputs for closed loop system, for I = 1 to NU
     - **QIN(I)**: Heat flow parameters for I = 1 to NL

  2. You may enter "N" which implies that there is no initial condition
     file.

     - **NX**: number of states
     - **SX(I)**: Enter initial states, for I = 1 to NX
- ANS: Respond as to whether discrete function is required or not
  Y - to use closed loop system
  N - to use open loop system
- MX: Number of states to be plotted.
- IX(I): Which states to be plotted, for I = 1 to MX
- MZ: Number of outputs to be plotted
- IZ(I): Which outputs to be plotted.
- TR: Run time for the simulation program (DEF = 5 min.)
- TPR: Printing time interval on screen (DEF = 10 sec.)
  If ANS is Y (i.e., in closed loop case), then the following
  input variables are used:
  - TC: Control time interval (DEF = 5 sec.)
  - TPL: Plotting time interval (DEF = 1 sec.)
  - TS: Sampling period (DEF = .025 sec)

OUTPUT VARIABLES:
- IX(I): Which states to be plotted for I = 1 to MX

There are 17 states for the open loop case and 27 states
for the closed loop case, e.g., if states 1, 3 and 5 are to
be plotted, then IX(1) = 1, IX(2) = 3, IX(3) = 5. Therefore,
the following result will appear on the screen.

<table>
<thead>
<tr>
<th>STATES</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIME</td>
</tr>
</tbody>
</table>
- \( Y(1P,1) = QS \)
  : Total heat dissipated within the system.

- \( Y(1P,2) = T0 \)
  : The temperature of the transport line immediately after the pump arrangement.

- \( Y(1P,3) = T11 \)
  : The temperature of the transport line immediately after Heat Sink 2.

- \( Y(1P,4) = T12 \)
  : The temperature of the transport line immediately after Mix Valve.

- \( Y(1P,5) = T01 - T11 = DT1 \)
  : The temperature difference across Heat Exchanger 1.

- \( Y(1P,6) = T02 - T12 = DT2 \)
  : The temperature difference across Heat Exchanger 2

- \( Y(1P,7) = T0P(1) - TS(3) \)
- \( Y(1P,8) = T0P(2) - TI(3) \)
- \( Y(1P,9) = T0P(3) - TI(3) \)
  : The temperature difference across Heat Exchanger 3.

The temperature difference across Heat Exchanger 4.
The temperature difference across Heat Exchanger 5.

\( Y(1P,10) = DT1 = T12 - T11 \)
  : The temperature difference between the input of Heat Exchanger 2 and the input of Heat Exchanger 1.
Y(IP,11) = DELAY
   : Delay time for system pump input = MASS/hr.
   i.e., DELAY = 60 x 60 x MASS/M0
   where M0: overall fluid flow rate for pump.

Y(IP,12) = CONSTR.
   : Heat Sink 2 constraint, i.e., check existence of solution for SIG.

Y(IP,13) = TS
   : The temperature immediately after Heat Sink 1 and before Heat Sink 2.

Y(IP,14) = QS2
   : Total heat dissipated by Heat Sink 2.

Y(IP,15) = QS1
   : Total heat dissipated by Heat Sink 1.
DIGITAL SIMULATION FLOWCHART

RUNGE-KUTTA
\[ t = t + T_x \]

CONT.

LOC. CONT.  TCS (F)

COMMAND UPDATE?

START

N

Y

ALGEBRAIC SOLUTION (D)
TRESPTN
PROGRAM TO FIND TIME HISTORY
FOR TCS
USES NONADAPTIVE STEP SIZE RUNGE-KUTTA
NEEDS SUBROUTINES:
F( ) OR CONT( ) FOR CONTINUOUS DYNAMICS
D( ) FOR DISCRETE DYNAMICS

PROGRAM TRESPTN
PARAMETER (NDI=100)
REAL Y(0:1024,0:50),X(NDI),XD(NDI),SX(NDI)
INTEGER IX(50),IZ(50)
CHARACTER *20 FILNAM,ANS,ANSC
COMMON/HEATIN/QIN(5)
COMMON/CONTROL/U(32)
COMMON/COMMAND/UC(32)
COMMON/OUTPUT/Z(NDI)
EXTERNAL F, CONT
DATA TR,TPR,TC,TPL,TS/30,1,.5,.1, .000208333/
DATA TR,TPR,TC,TPL,TS/1.5,1.6667E-2,1 .6667E-2,
DATA TR,TPR,TC,TPL,TS/5.,.16667,.0833 3,.016667,4.16667E-4/
WRITE(*,*) 'OPEN(0) OR CLOSED(C) -LOOP SYSTEM?:
READ(*,'(A)') ANSC
WRITE(*,*)'NAME OF INITIAL CONDITION FILE ("N" FOR NONE)
READ(*,'(A)') FILNAM
IF(FILNAM.NE.'N') THEN
  OPEN(20,FILE=FILNAM,STATUS='OLD')
  READ(20,*) NX,NU,NL
  READ(20,*) (SX(I), I= 1,NX)
  IF(NU.GT.O.AND.ANSC.EQ.'O') READ(20,*) (U(I), I= 1,NU)
  IF(NU.GT.O.AND.ANSC.EQ.'C') READ(20,*) (UC(I), I= 1,NU)
  READ(20,*) (QIN(I), I= 1,NL)
  CLOSE(20)
ELSE
  WRITE(*,*)'HOW MANY STATES?'
  READ(*, *) NX
  WRITE(*,*)'ENTER INITIAL STATES:'
  READ(*, *) (SX(I), I= 1,NX)
END IF
DO 25 I= 1,NX
XD(I)= O.
WRITE(*,*) 'DISCRETE FUNCTION REQUIRED? (Y OR N)'
READ(*,'(A)') ANS
WRITE(*,*) 'HOW MANY STATES TO BE PLOTTED?'
READ(*,*) MX
DO 35 I= 1,MX
IX(I)= I
IF(MX.EQ.0) GO TO 40
WRITE(*,*) 'WHICH ONES?'
READ(*,*) (IX(I), I= 1,MX)
WRITE(*,*) 'PLOT HOW MANY OUTPUTS?'
READ(*,*) MZ
IF(MZ.EQ.0) GO TO 40
DO 37 I= 1,MZ

25 CONTINUE
35 CONTINUE
37 CONTINUE

$ 4.16667E-3,4.16667E-4/
DATA TR,TPR,TC,TPL,TS/5.,,16667,.08333,.016667,4.16667E-4/
WRITE(*,*) 'OPEN(0) OR CLOSED(C) -LOOP SYSTEM?:'
READ(*,'(A)') ANSC
WRITE(*,*)'NAME OF INITIAL CONDITION FILE ("N" FOR NONE)
READ(*,'(A)') FILNAM
IF(FILNAM.NE.'N') THEN
  OPEN(20,FILE=FILNAM,STATUS='OLD')
  READ(20,*) NX,NU,NL
  READ(20,*) (SX(I), I= 1,NX)
  IF(NU.GT.O.AND.ANSC.EQ.'O') READ(20,*) (U(I), I= 1,NU)
  IF(NU.GT.O.AND.ANSC.EQ.'C') READ(20,*) (UC(I), I= 1,NU)
  READ(20,*) (QIN(I), I= 1,NL)
  CLOSE(20)
ELSE
  WRITE(*,*)'HOW MANY STATES?'
  READ(*, *) NX
  WRITE(*,*)'ENTER INITIAL STATES:'
  READ(*, *) (SX(I), I= 1,NX)
END IF
DO 25 I= 1,NX
XD(I)= O.
WRITE(*,*) 'DISCRETE FUNCTION REQUIRED? (Y OR N)'
READ(*,'(A)') ANS
WRITE(*,*) 'HOW MANY STATES TO BE PLOTTED?'
READ(*,*) MX
DO 35 I= 1,MX
IX(I)= I
IF(MX.EQ.0) GO TO 40
WRITE(*,*) 'WHICH ONES?'
READ(*,*) (IX(I), I= 1,MX)
WRITE(*,*) 'PLOT HOW MANY OUTPUTS?'
READ(*,*) MZ
IF(MZ.EQ.0) GO TO 40
DO 37 I= 1,MZ
IZ(I) = I
WRITE(*,*) 'WHICH ONES?'
READ(*,*) (IZ(I), I = 1,MZ)
E = 1.E-3

WRITE(*,*) 'RUN TIME? (DEF= 5 MIN)' READ(*,*) TR
WRITE(*,*) 'PRINTING TIME INTERVAL ON SCREEN? (DEF= 10 SEC)' READ(*,*) TPR
IF(ANS.EQ.'Y') THEN WRITE(*,*) 'CONTROL TIME INT.? (DEF= 5 SEC.)' READ(*,*) TC
END IF
WRITE(*,*) 'PLOTTING TIME INTERVAL? (DEF= 1 SEC)' READ(*,*) TPL
WRITE(*,*) 'SAMPLE PERIOD? (DEF= .025 SEC)' READ(*,*) TS
NPR= NINT(TR/TPR)
NTC= NINT(TPR/TC)
NPL= NINT(TPR/TC)
NT= NINT(TPL/TS)
TC= TC/60.
TS= TS/60.
TIME= 0.
IT= 0
IP= 0
DO 60 I= 1,NX
   X(I)= SX(I)
   CALL F(TIME,X,XP)
   IF(ANSC.EQ.'C') CALL CONT(TIME,X,XP)
   Y(0,0)= TIME*60.
   DO 70 I= 1,MX
      Y(0,I)= X(IX(I))
      IF(MZ.GT.0) THEN
         DO 75 I= 1,MZ
            Y(0,MX+I)= Z(IZ(I))
         END IF
      END IF
   END DO
60
DO 110 I= 1,NPR
   DO 100 II= 1,NTC
      IF(ANSC.EQ.'Y') CALL D(IT,X)
      DO 100 J= 1,NPL
         DO 90 K= 1,NT
            IF(IT.EQ.0) THEN
               WRITE(*,*), (IX(IND), IND= 0,MX)
            END IF
            WRITE(*,80) (IX(IND), IND= 0,MX)
35X, 'STATES'/ TIME',10(I12))
35X, '11(1PE12.3)') (Y(0,IND), IND= 0,MX+MZ)
80 END IF
END IF
IF (ANSC.EQ.'O') CALL RUNKUT(F,TIME,TS,X,NX)
IF (ANSC.EQ.'C'.AND.FILNAM.EQ.'HOTST') THEN CALL RUNK(CONT,TIME,TS,X,NX)
ELSE
   IF (ANSC.EQ.'C') THEN CALL RUNKUT(CONT,TIME,TS,X,NX)
   END IF
ENDIF
IT= IT+1
90 TIME= FLOAT(IT)*TS
SUBROUTINE RUNKUT(F,TIME,TS,X,N)

TS SAMPLE PERIOD
X STATE VECTOR
N NUMBER OF STATES
XP DERIVATIVE OF STATE VECTOR

PARAMETER (NDIM=32)
REAL X(*), XP(NDIM), X1(NDIM), XP1(NDIM)

CALL F(TIME,X,XP)
DO 10 I= 1,N
   X1(I) = X(I) + .5*TS*XP(I)

TIME= TIME + .5*TS
CALL F(TIME,X1,XP1)
DO 20 I= 1,N
   XP(I) = XP(I) + 2.*XP1(I)
   X1(I) = X(I) + .5*TS*XP1(I)

END
SUBROUTINE RUNK (F, TIME, TS, X, N)

PARAMETER (NDIM=33)
REAL X(*), XP(NDIM), X1(NDIM), XP1(NDIM)

CALL F(TIME, X, XP)
DO 10 I=1,N
  X1(I) = X(I) + .5*TS*XP(I)
10
TIME = TIME + .5*TS
CALL F(TIME, X1, XP1)
DO 20 I=1,N
  XP(I) = XP(I) + 2.*XP1(I)
20   X1(I) = X(I) + .5*TS*XP1(I)

CALL F(TIME, X1, XP1)
DO 30 I=1,N
  XP(I) = XP(I) + 2.*XP1(I)
30   X1(I) = X(I) + TS*XP1(I)

TIME = TIME + .5*TS
CALL F(TIME, X1, XP1)
DO 40 I=1,N
  X(I) = X(I) + TS*(XP(I)+XP1(I))/6.
40
RETURN
END
FILE TCSSIM2
DYNAMIC SIMULATION OF SOUTH BUILDING TCS
USING TWO HEAT SINKS

FOR SERIES, RHO= 1
FOR PARALLEL-1, RHO= 0
FOR PARALLEL-2, RHO= 0, DEL= 0

SUBROUTINE F(TIME,X,XP)
REAL X(*), XP(*)
REAL MU,MO,M11,MASS,MS,MP1,MP2,M(5),TCON(27),C(3),TOP(3)
REAL KMO,KMU,KSIG,KDEL,KBET,KGAM,KTAU
COMMON/HEATIN/QIN(5)
COMMON/CONTROL/VMO,VMU,VSIG,VDEL,VBET,VGAM,VTAU
COMMON/OUTCON/DT(5),M,DTI,T11,M0,MS
COMMON/OUTPUT/QS,T0,ZT11,T12,ZDT(5),ZDTI,DELAY,CONSTR,TS,QS2,QS1
COMMON/OUTPUT/ZDTI,ZT11,ZDT(5)
COMMON/UPLINK/AL,Q1,R1,R2,RP,R0

DATA CV/.0002936/
DATA (C(I), I= 1,3)/3*0.333333333/
DATA MASS,TSA/10.,2.E-8/
DATA (TCON(I), I= 1,5)/0.5,.0333,3*0.133/
DATA (TCON(I), I= 6,17)/0.,.0001388889,0.,.000138889,
& 0.,.000138889,0.,.000138889,0.,.000138889,0.,.000138889/
DATA TCON(25),TCON(26)/0.,.000138889/
DATA KMO,KMU,KSIG,KDEL,KBET,KGAM,KTAU/100.
DATA RHO/1/

TO= X(6)
MO= X(7)
MU= X(8)
SIG= X(10)
DEL= X(12)
BET= X(14)
GAM= X(16)
TAU= X(25)

OMU= 1.-MU
OMB= 1.-BET
OMG= 1.-GAM
OMD= 1.-DEL
OMS= 1.-SIG
OMR= 1.-RHO
OMT= 1.-TAU
FACT= DEL + RHO - DEL*RHO
AL= MU*FACT/(MU*FACT + OMU)
OMA= 1.-AL

MASS FLOW RATES

M11= MO*AL
MS= MO*OMA
M(1)= MS*OMD
MP1= M(1)*OMR
MP2= MO*AL + M(1)*RHO + MS*DEL
M(2)= MP2*OMB
DO 10 I= 1,3
M(2+I)= MP2*OMG*C(I)
ALGEBRAIC RELATIONS AROUND THE LOOP

TEST
WRITE(*,*)'MS,M(1),SIG: 	 ',MS,M(1),SIG
IF((QS.LT.5.5).OR.(T0.LT.77.5))THEN
  TAU=1
  TS=T0
ELSE
  TS= FSINK1(T0,M0,TAU)
END IF
QS1= M0*CV*(T0-TS)
TI1= FSINK2(TS,MS,SIG)
QS2= MS*CV*(TS-TI1)
QS= QS1 + QS2
T01= TI1 + X(1)/(M(1)*CV)

PARALLEL-2
IF(DEL.EQ.0. .AND. RHO.EQ.0.) THEN
  TI2= TS
  GO TO 15
END IF

SERIES AND PARALLEL-1
T011= (DEL*TI1 + OMD*RHO*T01)/(DEL + OMD*RHO)
TI2= MU*TS + OMU*T011

15 T02= TI2 + X(2)/(M(2)*CV)
TI3= BET*TI2 + OMB*T02
DO 20 I= 1,3
  TOP(I)= TI3 + X(2+I)/(M(2+I)*CV)
T0= GAM*MP2*TI3 + MP1*T01
T0= ( T0 + M(3)*TOP(1) + M(4)*TOP(2) + M(5)*TOP(3 ) )/M0

DYNAMICS

HEAT SOURCES AND TEMP. DYNAMICS
DO 40 I= 1,5
  XP(I)= (-X(I) + QIN(I))/TCON(I)
  SUM= -QS
DO 50 I= 1,5
  SUM= SUM + X(I)
  XP(6)= SUM/(MASS*CV)

ACTUATOR LIMITS
IF(X(12).LE.0) THEN
  X(12)= 0
IF(X(13).LE.0) THEN
  X(13)= 0
  VDEL= AMAX1(VDEL,0.)
END IF
END IF

PUMP AND VALVE ACTUATOR MOTORS
\[
\begin{align*}
\text{XP}(7) &= (-X(7) + KMO*VMO)/TCON(7) \\
\text{XP}(8) &= X(9) \\
\text{XP}(9) &= (-X(9) + KMU*VMU)/TCON(9) \\
\text{XP}(10) &= X(11) \\
\text{XP}(11) &= (-X(11) + KSIG*VSIG)/TCON(11) \\
\text{IF}((R0.GT.R2.AND.R0.GT.RP).OR.(R1.GT.R2.AND.R1.GT.RP))\text{THEN} \\
& \quad \text{X}(12)=0 \\
& \quad \text{X}(13)=0 \\
& \quad \text{XP}(12)=0 \\
& \quad \text{XP}(13)=0 \\
\text{ELSE} \\
& \quad \text{XP}(12)= X(13) \\
& \quad \text{XP}(13)= (-X(13) + KDEL*VDEL)/TCON(13) \\
\text{ENDIF} \\
\text{IF}(R2.GE.MO)\text{THEN} \\
& \quad \text{X}(14)=0 \\
& \quad \text{X}(15)=0 \\
& \quad \text{XP}(14)=0 \\
& \quad \text{XP}(15)=0 \\
\text{ELSE} \\
& \quad \text{XP}(14)= X(15) \\
& \quad \text{XP}(15)= (-X(15) + KBET*VBET)/TCON(15) \\
\text{ENDIF} \\
\text{IF}(RP.GE.MO)\text{THEN} \\
& \quad \text{X}(16)=0 \\
& \quad \text{X}(17)=0 \\
& \quad \text{XP}(16)=0 \\
& \quad \text{XP}(17)=0 \\
\text{ELSE} \\
& \quad \text{XP}(16)= X(17) \\
& \quad \text{XP}(17)= (-X(17) + KGAM*VGAM)/TCON(17) \\
\text{ENDIF} \\
\text{IF}(QS.LT.5.5.0R.TO.LT.77.5)\text{THEN} \\
& \quad \text{X}(25) = \frac{(1-X(30))/200 + X(25)}{} \\
& \quad \text{IF}(X(25).GE.1.OR.INT(X(31)).EQ.1)\text{THEN} \\
& \quad \quad \text{X}(25) = 1. \\
\text{END IF} \\
& \quad \text{X}(25) = 1. \\
& \quad \text{X}(26) = 0 \\
& \quad \text{X}(32) = 0.48 \\
& \quad \text{X}(33) = 1.0 \\
& \quad \text{X}(27) = 0 \\
& \quad \text{XP}(25) = 0 \\
& \quad \text{XP}(26) = 0 \\
& \quad \text{XP}(27) = 0 \\
\text{ELSE} \\
& \quad \text{IF}(X(32).GT.0.5.OR.INT(X(33)).GE.200)\text{THEN} \\
& \quad \quad \text{XP}(25) = X(26) \\
& \quad \quad \text{XP}(26) = (-X(26)+KTAU*VTAU)/TCON(26) \\
& \quad \quad \text{X}(30) = X(25) \\
& \quad \quad \text{X}(31) = 1.0 \\
\text{ELSE} \\
& \quad \quad \text{X}(25) = X(25)-((1-X(32))/200) \\
& \quad \quad \text{X}(33) = X(33)+1.0 \\
& \quad \quad \text{X}(26) = 0 \\
& \quad \quad \text{X}(27) = 0 \\
& \quad \quad \text{XP}(25) = 0 \\
\end{align*}
\]
XP(26)=0
XP(27)=0
END IF
END IF

C OUTPUTS FOR CONTROL

DT(1)= T01 - TI1
DT(2)= T02 - TI2
DT(3)= TOP(1) - TI3
DT(4)= TOP(2) - TI3
DT(5)= TOP(3) - TI3
DTI= TI2 - TI1

C MEASURED OUTPUTS

DELAY= 60.*60.*MASS/M0
CONSTR= FCSTR(T0,MS,QS)
DO 60 I= 1,5
  ZDT(I)= DT(I)
  ZDTI= DTI
  ZTI1= TI1
60
RETURN
END

C HEAT SINK FUNCTION NUMBER 1 (70 DEG SINK)
C SOLVES FOR TS: THE FORWARD PROBLEM

REAL FUNCTION FSINK1(T0,M0,TAU)
REAL K1,K2,K3,MC,MO,MSINK
DATA T1,A,F,MC,CV,CVC/70,11,.95,10000,.0002936,.0004976/
DATA CO,C1,C2/.031,4.35E-5,-1.22E-8/
DATA EPS/1.E-7/

IF(ABS(1-TAU).LE.EPS.OR.TAU.GT.1) THEN
  FSINK1= T0
  RETURN
END IF

OMT= 1.-TAU
MSINK= M0*OMT
H= CO + C1*MSINK + C2*MSINK**2
K1= EXP(-H*A*F/(MC*CVC))
K2= H*A*F/CV
K3= CV/(MC*CVC)

C TEST

WRITE(*,*)
WRITE(*,*)'K1,K2,K3: ',K1,K2,K3
WRITE(*,*)'T0,M0,TAU: ',T0,M0,TAU
WRITE(*,*)'MSINK: ',MSINK
WRITE(*,*)

TOS= T1 + K3*MSINK*T0 + (T0-T1)*K1*EXP(-K2/MSINK)
TOS= TOS/(1.+K3*MSINK)
FSINK1 = TAU*T0 + OMT*TOS

RETURN
END

!EAT SINK FUNCTION NUMBER 2 (35 DEG SINK)
!OLVES FOR T1: THE FORWARD PROBLEM

REAL FUNCTION FSINK2(TS, MS, SIG)
REAL K1, K2, K3, MC, MS, MSINK
DATA T1, A, F, MC, CV, CVC/35, 11, .95, 10000, .0002936, .0004842/
DATA CO, C1, C2/.031, 4.35E-5, -1.22E-8/
DATA EPS/1.E-7/

IF(ABS(1-SIG).LE.EPS.OR.SIG.GT.1) THEN
   FSINK2 = TS
   RETURN
END IF

OMS = 1.-SIG
MSINK = MS*OMS
H = CO + C1*MSINK + C2*MSINK**2
K1 = EXP(-H*A*F/(MC*CVC))
K2 = H*A*F/CV
K3 = CV/(MC*CVC)

TEST

WRITE(*,*)
WRITE(*,*)'K1, K2, K3: ', K1, K2, K3
WRITE(*,*)'TS, MS, SIG: ', TS, MS, SIG
WRITE(*,*)'MSINK: ', MSINK
WRITE(*,*)

TOS = T1 + K3*MSINK*TS + (TS-T1)*K1*EXP(-K2/MSINK)
TOS = TOS/(1.+K3*MSINK)
FSINK2 = SIG*TS + OMS*TOS

RETURN
END

!EAT SINK CONSTRAINT
!CHECKS EXISTENCE OF SOLUTION FOR SIGMA

REAL FUNCTION FCONSTR(T0, MS, QS)
REAL K1, K2, K3, MC, MS
DATA T1, A, F, MC, CV, CVC/35, 11, .95, 10000, .0002936, .0004842/
DATA CO, C1, C2/.031, 4.35E-5, -1.22E-8/

H = CO + C1*MS + C2*MS**2
K1 = EXP(-H*A*F/(MC*CVC))
K2 = H*A*F/CV
K3 = CV/(MC*CVC)
FCONSTR = T1 + K3*QS + (T0-T1)*K1*EXP(-K2/MS)

RETURN
END
SUBROUTINE D(IT,X) TO PROVIDE DISCRETE CONTROLLER

SUBROUTINE D(IT,X)
REAL X(*),M0
COMMON/COMMAND/M0
COMMON/UPLINK/AL,Q1,R1,R2,RP,R0
DATA CV,TI1C,TI2C/.0002936,40,70/

R0= Q1/(CV*(TI2C-TI1C))
IF (AL.EQ.1.) AL=.99
R1= R1/(1-AL)
M0= AMAX1(R0,R1,R2,RP)
RETURN
END

SUBROUTINE TO CONNECT LOCAL CONTROLLERS

SUBROUTINE CONT(TIME,X,XP)
REAL X(*),XP(*)
DATA N/17/

C LOCAL CONTROLLERS
CALL LOCALP(X(N+1),XP(N+1))
CALL LOCALM(X(N+3),XP(N+3))
CALL LOCALS(X(N+4),XP(N+4))
CALL LOCALD(X(N+5),XP(N+5))
CALL LOCALB(X(N+6),XP(N+6))
CALL LOCALG(X(N+7),XP(N+7))
CALL LOCALT(X(N+10),XP(N+10))

C TCS
CALL F(TIME,X,XP)
RETURN
END

LOCAL CONTROLLER FOR PUMP (M0DOT)

SUBROUTINE LOCALP(Z,ZP)
REAL Z(2),ZP(2)
REAL K,M0,MOC,MOCF
COMMON/COMMAND/MOC
COMMON/CONTROL/VM0
COMMON/OUTCON/DUM(12),M0
DATA K,TCONF,A/2,.00833333,0.1/

C GENERATE ERROR
MOCF= Z(1)
E= MOCF-M0

B-26
FILTER AND COMPENSATOR DYNAMICS

\[ U = K \cdot E \]
\[ ZP(1) = \frac{(-Z(1) + M0C)}{TCONF} \]
\[ ZP(2) = U \]
\[ VM0 = Z(2) + A \cdot U \]

RETURN
END

LOCAL CONTROLLER FOR MU (MIXING VALVE)

SUBROUTINE LOCALM(Z,ZP)
REAL K,M0,MS
COMMON/CONTROL/DUM,VMU
COMMON/OUTCON/DUMO(10),DTI,DUMO1,M0,MS
COMMON/UPLINK/AL
DATA DTIC,K,TCON,A/30,200,3.47E-5,2/

;GENERATE UPLINK SIGNAL FOR NEXT LEVEL CONTROLLER
AL = 1 - MS/M0

;GENERATE ERROR
E = DTIC - DTI

;COMPENSATOR DYNAMICS
\[ U = K \cdot E \]
\[ ZP = \frac{(-Z + U)}{TCON} \]
\[ VMU = (1.-A) \cdot Z + A \cdot U \]
RETURN
END

LOCAL CONTROLLER FOR SIGMA (35 DEG. HEAT SINK)

SUBROUTINE LOCALS(Z,ZP)
REAL K,MS
COMMON/CONTROL/DUM(2),VSIG
COMMON/OUTCON/DUMO(11),TI1
DATA TIIC,K,TCON,A/40,200,3.47E-5,2/

;GENERATE ERROR
E = TIIC - TI1

;COMPENSATOR DYNAMICS
\[ U = K \cdot E \]
\[ ZP = \frac{(-Z + U)}{TCON} \]
\[ VSIG = (1.-A) \cdot Z + A \cdot U \]
RETURN
END
C LOCAL CONTROLLER FOR TAU (70 DEG. HEAT SINK)
SUBROUTINE LOCALT(Z,ZP)
REAL K,TSH
COMMON/CONTROL/DUM(6),VTAU
COMMON/OUTPUT/QS,T0,DUMO3(10),TS
DATA TSC,K,TCON,A/77.5,200,3.47E-5,2/
IF((QS.LT.5.5).OR.(T0.LT.TSC))THEN
  C TSH=T0
  TAU= 1
  VTAU=0
  C U=K*E
  C ZP=(-Z+U)/TCON
  C VTAU=(1.-A)*Z+A*U
  RETURN
ELSE
  C GENERATE ERROR
  E= TSC - TS
  C COMPENSATOR DYNAMICS
  U= K*E
  ZP= (-Z + U)/TCON
  VTAU= (1.-A)*Z + A*U
END IF
RETURN
END

C LOCAL CONTROLLER FOR DELTA (HS #1)
SUBROUTINE LOCALD(Z,ZP)
REAL K,M1
COMMON/CONTROL/DUM(3),VDEL
COMMON/OUTCON/DT,DUMO1(4),M1
COMMON/UPLINK/DUMU,Q,R
DATA CV,DTC,K,TCON,A/.0002936,19,200,3.47E-5,2/
C UPLINK SIGNALS FOR NEXT LEVEL CONTROLLER
  Q= M1*CV*DT
  R= Q/(CV*DTC)
C GENERATE ERROR
  E= DTC-DT
C COMPENSATOR DYNAMICS
  U= K*E
  ZP= (-Z + U)/TCON
  VDEL= (1.-A)*Z + A*U
RETURN
END

C LOCAL CONTROLLER FOR BETA (HS #2)
SUBROUTINE LOCALB(Z,ZP)
REAL K,M2
COMMON/CONTROL/DUM(4), VBET
COMMON/OUTCON/DUMO, DT, DUMO1(4), M2
COMMON/UPLINK/DUMU(3), R
DATA CV, DTC, K, TCON, A/.0002936, 5, 200, 3.47E-5, 2/

;PLINK SIGNAL FOR NEXT LEVEL CONTROLLER

Q = M2*CV*DT
R = Q/(CV*DTC)
IF (R2.GE.M0) THEN
  VBET=0
  RETURN
ELSE

;GENERATE ERROR

E = DTC-DT

;COMPENSATOR DYNAMICS

U = K*E
ZP = (-Z + U)/TCON
VBET = (1.-A)*Z + A*U
RETURN
END IF
END

LOCAL CONTROLLER FOR GAMMA (HS #3-#5)

SUBROUTINE LOCALG(Z,ZP)
REAL K, MP(3), C(3), RR(3), QP(3)
COMMON/CONTROL/DUM(5), VGAM
COMMON/OUTCON/DUMO(2), DT(3), DUMO1(2), MP
COMMON/UPLINK/DUMU(4), R
DATA CV, DTC, K, TCON, A/.0002936, 40, 200, 3.47E-5, 2/
DATA (C(I), I= 1,3)/3*.3333333/

;PLINK SIGNAL FOR NEXT LEVEL CONTROLLER

DO 10 I= 1,3
QP(I) = MP(I)*CV*DT(I)
RR(I) = QP(I)/(CV*C(I)*DTC)
R = AMAX1(RR(1), RR(2), RR(3))

;GENERATE ERROR

DTM = AMAX1(DT(1), DT(2), DT(3))
E = DTC-DTM

;COMPENSATOR DYNAMICS

U = K*E
ZP = (-Z + U)/TCON
VGAM = (1.-A)*Z + A*U
RETURN
END
C SUBROUTINE TO INJECT IMPULSE

SUBROUTINE CD(IT,X)
REAL X(*)
COMMON/CONTROL/VM0,VMU,VSIG,VDEL,VBET,VGAM

IF(IT.EQ.0) VSIG= 10
IF(IT.EQ.160) VSIG= 0

RETURN
END