LARGE LIQUID ROCKET ENGINE TRANSIENT PERFORMANCE SIMULATION SYSTEM

FINAL REPORT

Prepared for
NASA-George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

Prepared Under
United Technologies Corporation
Pratt & Whitney
Government Engine Business
P.O. Box 109600,
West Palm Beach, FL 33410-9600

J. R. Mason
J. R. Mason
P&W Program Manager

R. D. Southwick
R. D. Southwick
Propulsion Systems
Technology Manager

FOREWORD

Pratt & Whitney, Government Engine Business of United Technologies Corporation conducted this program for the National Aeronautics and Space Administration, George C. Marshall Space Flight Center under contract NAS8-36994. The NASA project manager for this contract was Mr. W.A. Adams, Jr. of the MSFC, mechanical systems control branch. The P&W program manager was Mr. J.R. Mason with technical contributions of Mr. D.L. Baker, Mr. C.R. Byrd, Mr. T.F. Denman, Mr. H. P. Frankl, Mr. S.M. Mericle, Mr. R.W. Parham, Mr. J.W. Park, Mr. J.E. Pollard, Mr. T.J. Roadinger, Mr. R.S. Rosson, Mr. M.H. Sabatella, Mr. D.H.Spear, Mr. J.P. Spinn, and Mr. P.W. McLaughlin of "The Simulation and Modeling Workshop".

CONTENTS

SECTION I

SUMMARY

A new simulation system, ROCETS, was designed and developed to allow cost-effective computer predictions of liquid rocket engine transient performance. The system allows a user to generate a simulation of any rocket engine configuration using component modules stored in a library thru high-level input commands. The system library currently contains 24 component modules, 57 sub-modules and maps, and 33 system routines and utilities. FORTRAN models from other sources can be operated in the system upon inclusion of interface information on comment cards. Operation of the simulation is simplified for the user by Run, Execution and Output Processors. The simulation system makes available steady-state trim balance, transient operation, and linear partial generation. The system utilizes a modern equation solver for efficient operation of the simulations. Transient integration methods include integral and differential forms for the trapezoidal, first order Gear, and second order Gear corrector equations.

A detailed technology test bed engine (TTBE) model was generated to be used as the acceptance test of the simulation system. The general level of detail of the model was that reflected in the SSME DTM (Reference 2). The model sucessfully obtained stady-state balance in main stage operation and simulated throttle transients including engine start and shutdown. A NASA fortran control model was obtained, ROCETS interface installed in comment cards, and operated with the TTBE model in closed-loop transient mode.

This page left intentionally blank.

SECTION II
INTRODUCTION

The National Aeronautics and Space Administration (NASA) Facilities such as the George C. Marshall Space Flight Center (MSFC) require analysis and simulation of pump fed liquid rocket engine transient performance. The types of analysis and simulation include control design and analysis, design parametric studies, research and development, failure investigation, real-time simulation, feasibility studies, and software design, development, and testing. Therefore, multiple simulations representing different engine configurations with various levels of fidelity and transient response ranges are needed to support these studies. An analytical tool to meet these needs in a cost-effective manner is a digital computer simulation system.

A computer simulation system named ROCket Engine Transient Simulation (ROCETS) was designed and developed under this program. An engine transient performance simulation normally consists of mathematical representations of the engine components interfaced together to describe the engine system performance. These component-by-component engine simulations (Figure 2-1) require interfacing the component models together in a computer program, with appropriate program controls to interpret user commands, execute the program, and provide outputs to the user. All of this can be accomplished with in-line computer code that is a free-standing simulation. However, a simulation system provides many benefits relative to individual, free-standing simulations.

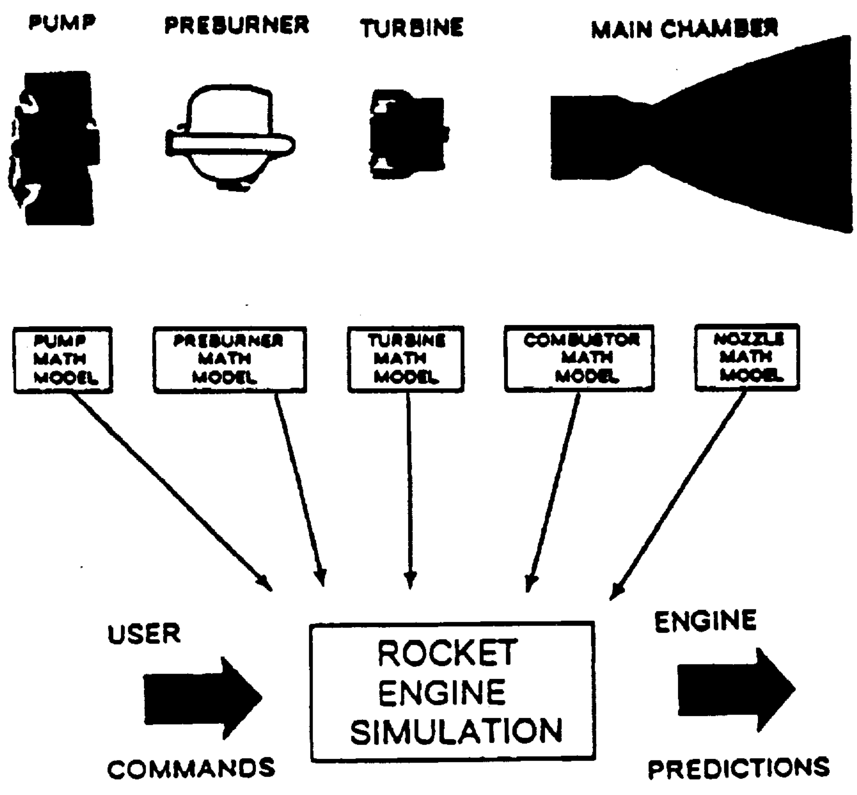

Figure 2-1. A Rocket Performance Simulation Consist of Component-By-Component Models

A simulation system (Figure 2-2) allows generation of simulations representing different engine configurations without expensive new computer code production and verification. The system acts as a repository so that the same engineering methodology representing the components is utilized in different simulations to ensure prediction consistency. In addition, the system provides the latest modeling technology of verified numerical techniques and utilities; new advances placed in the system can easily be shared by all simulations operating in the system. A simulation system also provides a common operating base for all users to minimize required operational training after the initial start-up experience is obtained.

- Re-Use Of Developed/Verified Model Codes
- Repository For Methodology
- Advanced Modeling Technology & Techniques Easily Adaptable
- Reduces Required User Training

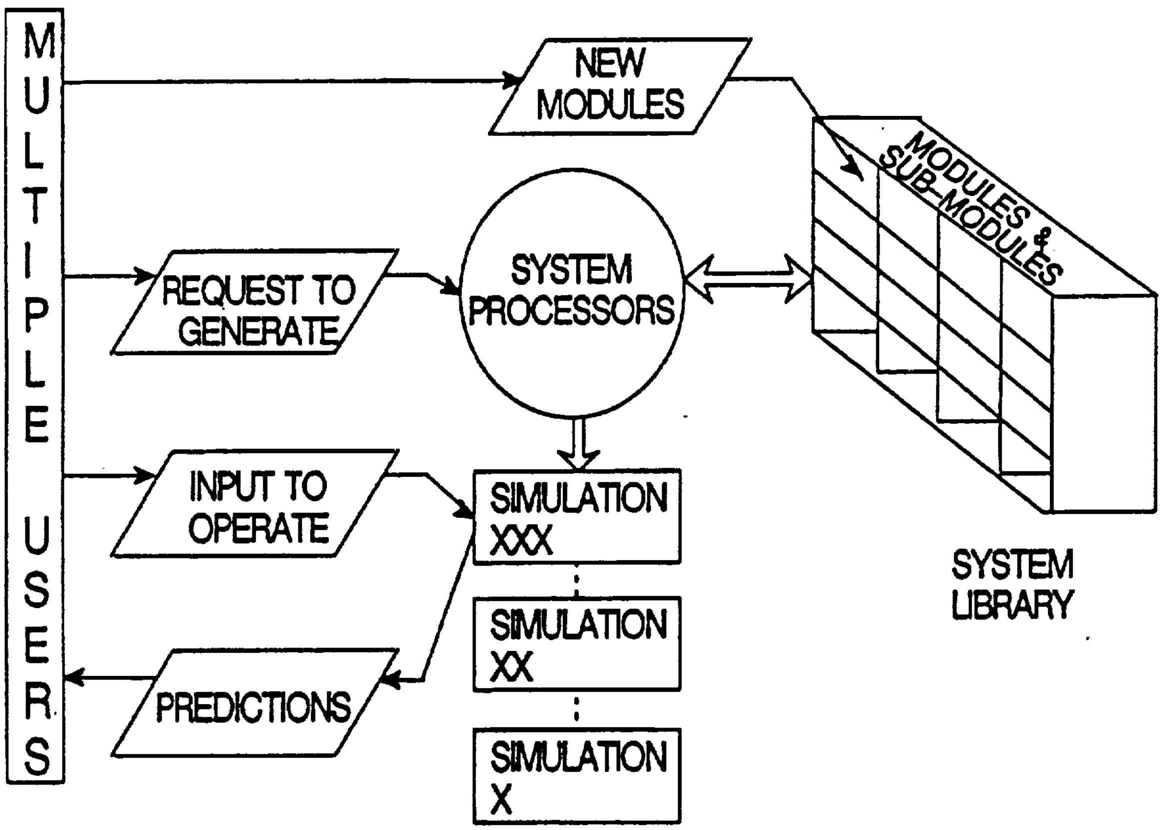

Figure 2-2. Simulation Systems Are Effective Tools

The ROCETS program to design and develop a simulation system consisted of nine (9) technical tasks:

1. Architecture
2. System Requirements
3. Component and Submodel Requirements
4. Submodel Implementation
5. Component Implementation
6. Submodel Testing and Verification

7. Subsystem Testing and Verification
8. TTBE Model Data Generation
9. System Testing & Verification

The Architecture definition determined there would be five major components of the ROCETS system:

1. Library System
2. Executive programs (or Processors)
3. Simulation Input and Output
4. Documentation
5. Maintenance Procedures

The requirements were developed and documented in the System Requirements Specification (SRS) of P&W FR-20283, 25 November 1988 (Reference 3). The component and submodel implementation and testing/verification is contained in the System Design Specification (SDS) of P&W FR-20284, 25 July 1990 (Reference 4). The Technology Test Bed Engine (TTBE) Model description and system testing/verification are contained in this report.

This page left intentionally blank.

SECTION III
SYSTEM DESCRIPTION

The Rocket Engine Transient Simulation (ROCETS) System was designed to use modular building blocks to represent engine components, and an architecture to interface these modules in any configuration desired by the user when generating an engine simulation. The architecture structure does not include any specific rocket engine configuration, and thus the flexibility exists to configure any rocket cycle of the future. The five components of the ROCETS system are:

1. Library – A central source of all software code to allow multiple-users.
2. Executive Programs – Software processors that conduct system functions.
3. Simulation Input/Output – User inputs to configure a simulation, to execute the simulation, and to output the desired parameters.
4. Documentation – System standards, engineering descriptions, user's manual, programmers manual, and qualification test plants (contained in FR-20284).
5. Maintenance Procedures – Instructions for system upkeep (contained in FR-20284).

3.1 SYSTEM OVERVIEW

The ROCETS System has engineering models of all major engine components which are implemented as FORTRAN subroutines. These subroutines are called "modules". Standard engineering modules, once fully verified and documented, are put into a library so they can be accessed by all system users. A unique aspect of the ROCETS system is that engineering modules use comment cards to interface with the system. This allows ROCETS modules to be used outside the system as well as the ability to quickly adapt existing code to be used inside the system.

Virtually any engine cycle can be represented by connecting the engineering modules in a desired order. While the modules could be connected by hand (i.e., an engineer building a main concatenating routine), this is time consuming, tedious, and error prone. The ROCETS system uses a Configuration Processor to accomplish this task (Figure 3-1). An engineer builds a configuration input file using high-level commands and the Configuration Processor generates an executable FORTRAN main program. The Configuration Processor also scans the execution order to identify algebraic loops required by the model. Algebraic loops are caused by variables which are used before being calculated, or variables which are outputs of more than one module.

The ROCETS systems also uses a high-level command language to supply necessary inputs to run a particular model experiment. The input is read and interpreted by a Run Processor. The Run Processor initializes necessary inputs and sets appropriate flags to carry out the users instructions. The Run Processor allows the user to input schedules, set-up additional algebraic balances, and tailor a variety of integration options.

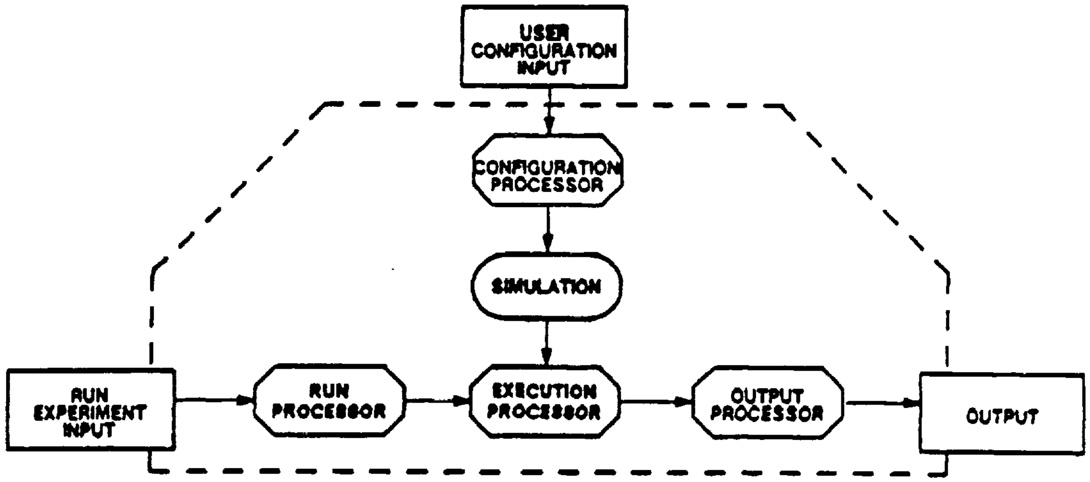

Figure 3-1. ROCETS System Overview

Execution control is provided by an Execution Processor. It controls looping, print, balancing, and linearization. Within the Execution Processor are calls to numerical utilities that provide steady-state balancing, transient integration, and linearization. It provides a centralized location for all numerical operations so that adding new features to the system in the future is simplified.

Output is controlled by an Output Processor that allows the user to specify parameters to be printed and plotted. Plot information is supplied to a interface routine designed for a particular plotting software package. Therefore, implementing plotting on a different system requires only a change in the plot interface routine. Linearization output is not controlled by the Output Processor, but rather all necessary information is passed to a separate interface routine. This feature allows tailoring of the linearization output by changing only the interface routine.

The ROCETS system has three run modes: Steady-state trim balance, Transient, and Linearization. The steady-state trim balance mode iterates dynamic states and algebraic balance variables until time derivatives and algebraic balance error terms are zero (within a specified tolerance). Transient integration normally integrates dynamic states using a predictor-corrector scheme with the corrector equations and algebraic balances closed simultaneously. The linearization mode linearizes about a steady-state or transient point and provides state-space model partials which can be used for other applications. Table 3-1 presents a summary of the ROCETS system significant features.

Table 3-1. ROCETS System Significant Features

Library

1. Storage for re-use of developed codes.
2. Access for multi-users
3. Repository for modeling methodology
4. Allows adaptable future modeling technology

Component Based

1. Component models are non-linear representation
2. Generic component modules; unique characteristics in distinct sub-modules.

Interface Structure

1. Component models use comment cards to interface with system
2. Existing models linked in any arrangement to simulate all engine cycles.
3. Any FORTRAN model can easily be used in system simulations.

Configuration Processor

1. Structured, English-like input
2. Automatic scanning for required inputs and algebraic balances
3. Generates FORTRAN main program

Run Processor

1. Structured, English-like input
2. Schedule (curve) input available
3. Definition of additional algebraic balances
4. Activation for states and balances
5. Three run modes: steady-state, transient, linearization

Close-loop Integration With State-of-the Art Numerical Utilities

1. Trapezoidal, first and second order Gear methods; others can be adapted.
2. Ability to activate/deactivate states
3. Ability to remove dynamic effects of states (force derivatives to zero)
4. Advanced non-linear equation solver to close corrector and algebraic balances simultaneously.

Automatic Linear Partial Generation

1. Repeatability and linearity checking
2. Analytic handling of algebraic balances
3. Analytic handling of states forced to steady-state.

3.2 ENGINEERING MODULES

Engineering modules are stand-alone engineering representations of individual entities that are singular in purpose. The modular approach separates engineering modules, sub-modules, component data and generic data (properties) into the basic building blocks of the simulation. For example, a generic turbine module can be used multiple times in a single simulation simply by changing the component performance characteristics or map as well as being used in multiple simulations. This reduces the amount of code required while providing consistent methodology.

The approach taken in modeling gives primary preference to engineering first principals followed by empirical correlations and transfer functions. However, modules of similar functions can be built with different modeling approaches and varying levels of complexity. The user then has the flexibility to select different approaches and level of detail used in a simulation.

During the design phase of ROCETS, it was evident that the use of existing engineering representations would be desirable. To achieve this goal, it was decided to separate system functions from the engineering representations. This was accomplished by using call lists for communication to the engineering modules and keeping all system dependent code out of the individual modules. An additional benefit is that the modules can be operated as individual entities during design and verification. Modules only communicate to the ROCETS system through the subroutine call list. Commons are not used to communicate with the main or other modules. However, common blocks can be used in certain cases for communication between a module and a sub-module.

Modules are interfaced to the ROCETS system using three blocks of comment cards at the beginning of the subroutine. These comment card blocks are called "interface cards" and are read by the Configuration Processor. The interface blocks relate call list names to system names, define the status of each variable for system operation, define the I/O status of each variable, and the FORTRAN variable type. Virtually any FORTRAN subroutine can easily be converted to the ROCETS system by adding the interface information on comment cards. However, the module history including author, dated revisions and internal code documentation should also be included.

3.3 TRANSIENT MODELLING ASPECTS

In general, the dynamics which are modelled in a rocket engine consist of volume dynamics, flow inertia, rotor speed integration, and thermal capacitance. Volume dynamics implement the laws of conservation of mass and energy using density and internal energy as dynamic states. Flow inertia dynamics implement conservation of momentum using flow rate as the dynamic state. Thermal nodes implement heat transfer laws and the energy equation applied to a metal mass using the metal temperature as the dynamic state.

The baseline transient integration scheme is a predictor-corrector with the corrector equations closed by a modified Newton-Raphson iteration. Using a closed-loop integration offers advantages which are incorporated into the system. One item of particular usefulness is the capability of forcing states to their steady-state value during a transient. This is accomplished by using a steady-state error term (i.e., forcing the time derivative to zero) instead of closing the corrector equation for specified states. When this is done, the dynamic effects of the specified states are removed thereby allowing a variety of studies to be conducted. An obvious use of this feature is to obtain reduced order linear state-space models. However, it has also proven extremely valuable during model verification and validation.

With the closed-loop integration, using density and internal energy as states causes numerical problems in liquid systems due to the extreme sensitivity to pressure to density and the difficulty in providing first guesses for internal energy. It would be considerably better to use pressure and enthalpy as states but it is not possible to write appropriate differential equations. The solution to this problem is to make a change of iteration parameters. Instead of using density and internal energy as the iteration parameters to close the corrector equations, pressure and enthalpy are used.

3.4 GLOBAL COMMUNICATION

While engineering modules communicate through call lists, the system functions do not because ROCETS provides maximum flexibility by dynamically building system communication without using a predefined data structure. Therefore global commons are constructed which contain all the variables passed into or out of the engineering modules. These commons are used to communicate between the engineering modules, the interpretive reader, and the Execution Processor. Figure 3-2 depicts the communication flow.

The global commons are divided by FORTRAN variable type: real, integer, character, double precision, complex, and logical. In addition to the 6 global variable commons, additional system commons are used to communicate information concerning states, derivatives, and additional balances as well as other necessary information.

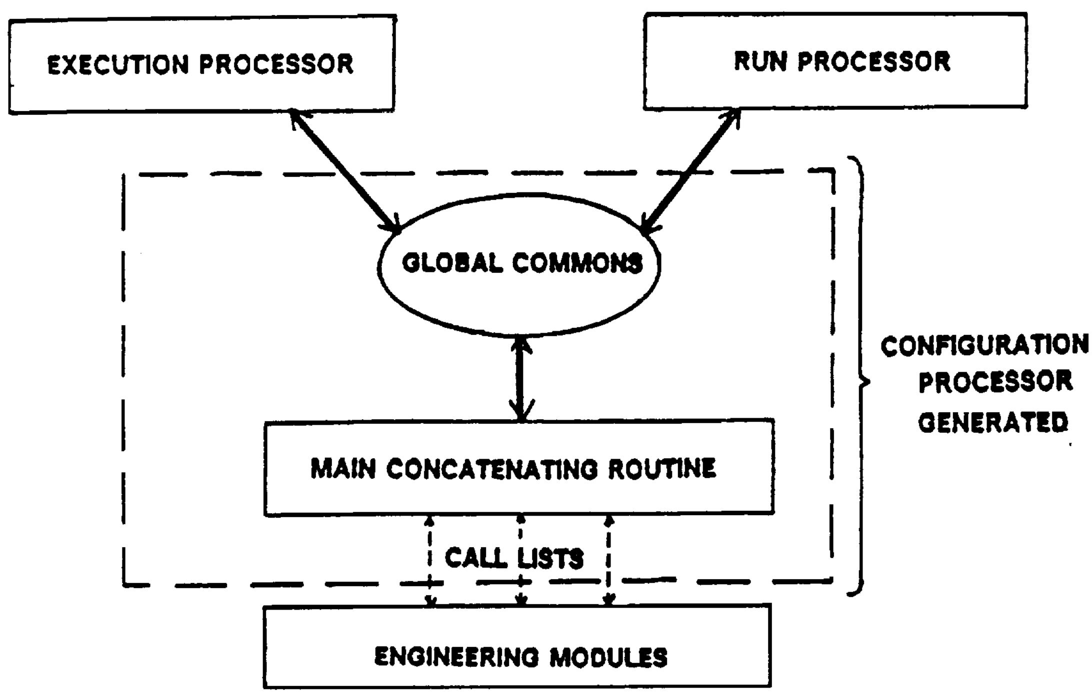

Figure 3-2. ROCETS Global Communication

3.5 CONFIGURATION PROCESSOR

The goal when defining a simulation is to converts an abstract concept into a mathematical representation in a flexible, reliable, and convenient manner. Therefore it is desirable to automate the simulation creation to the extent possible, freeing the user from the tedious aspect of assembling a simulation. In ROCETS, a Configuration Processor is used to automatically create a simulation. (Figure 3-1).

The configuration input consists of user commands defining a particular simulation in a simple, structured high-level format. The user defines the system to be modeled in the configuration input file by specifying component types, design characteristics, the relationship between various elements of the system, property packages to be used and what properties are to obtained, and definition of algebraic balances. (Note that algebraic balances can also be defined at run time).

The processor performs two functions in generating a simulation: first it reads the configuration input, then it reads the interface definitions for the modules specified in the configuration. The processor needs the engineering module interfaces to determine the required variables, call lists, variable status (input, output, state, derivative, etc.) and variable type (real, integer, array, character, etc.). The processor cross references the configuration input and the module interface information to generate the specific variable names. These names are used to generate the appropriate call lists for the FORTRAN main program/module communication. This methodology is what allows the engineering modules to remain separate from the system code.

The global commons are dynamically built for individual simulations during configuration. The commons consist of the variable names created from the module call lists along with required system variables.

3.6 RUN PROCESSOR

Run input consists of user commands to execute a configured simulation (Figure 3-1). The user input contains information required to define schedules, set inputs, define algebraic loops, specify output, and control execution. The input is in a high-level structured language.

The ability to define and use schedules is quite powerful. Besides allowing schedules for time inputs, schedules can be set-up to define desired functional relationships in conjunction with algebraic loops. As an example, schedules can be defined to set a requested chamber pressure and mixture ratio and algebraic balances defined to vary valve areas until the requested values are obtained.

Integration options can be tailored through run input to optimize model operation. The integration method, perturbation sizes, tolerance, convergence criteria, and activation can be set. The inputs are divided into "defaults" and "exceptions". It is generally easier to set-up default information which is adequate for most states and then to override the defaults for specific states when necessary. Currently the system includes Euler, trapezoidal, first order Gear, and second order Gear integration schemes. However, other integration schemes can be easily added.

The default for all states is to be active. However, it is often convenient to turn states off at various times. Three selections are possible for operations with each state:

ON = the state is active
OFF = the state is inactive and held constant
STEADY-STATE = the state is always iterated to steady-state thereby removing the dynamic effect of the state

ROCETS provides the capability to define algebraic balances at run time. An independent variable can be varied until a dependent variable is equal to another dependent variable or until a dependent variable is equal to a value. The value may be an input or read from a schedule. This is especially useful when running operating lines or generating control schedules.

Balance options can be tailored through run input similar to the integration options. The perturbation sizes, tolerance, convergence criteria, and activation can be set. Like the integration options, the inputs are divided into "defaults" and "exceptions".

Linearization options can also be defined at run time. Included are values to be used for repeatability and linearity checking. Partials are generated by making a forward perturbation, backward perturbation, and repeating the forward perturbation. Repeatability is checked by comparing variable values on the two forward perturbation passes. If the percentage difference is more than the specified value a warning message will be written. Linearity is checked by comparing the forward and backward difference partials. If the percentage difference is more than the specified amount a warning message will be written. In addition to the check values, parameter names for linear model inputs and outputs are specified by the user.

Linearization defaults are defined to establish the perturbation size for generating partials and exceptions to the defaults may also be specified. These functions are similar to default and exception declarations for states and balances.

Simulation output options are handled through the Run Processor. This includes optional print during Jacobian evaluations and convergence attempts, options for debug output, and specifying simulation output.

Three run modes are supported: Steady state, Transient, and Linearize. For steady-state, the number of consecutive steady-state points to be run is also specified. A system parameter POINT is available for reading schedules with the steady-state point number. This is useful for running steady-state operating lines or generating control schedules. For a transient, the time increment, print time, plot time, and termination time are specified. Time is available for reading schedules. The linearization mode perturbs each state and specified input to generate state-space model partial derivative matrices.

3.7 MULTI-VARIABLE NEWTON-RAPHSON SOLVER

ROCETS employs a state-of-the-art non-linear equation solver which is the heart of efficient system operation. It is a modified multi-variable Newton-Rahpson technique, which has been optimized to operate effectively with the large systems of equations encountered in the rocket modeling problems. The basic method operates on the matrix equation.

$$\Delta X = J^{-1} \Delta Y$$

Where ΔY is the amount the errors, or dependent variables, need to change to be zero and ΔX is the associated change in the independent variable. The solver Jacobian, J, is a matrix of partial derivatives generated with the model. The solver makes a number of passes equal to the number of iteration variables plus 1 through the model to generate the Jacobian.

To improve the efficiency of the solver, the Jacobian is scaled using a modified version of the method given by McLaughlin (Reference 8). The normalization factors from Jacobian scaling are then used in determining convergence as well as limiting allowed change of the independent variables. (This is necessary in non-linear systems to prevent excessive movement leading to exceeding map bounds, etc.)

Further enhancements include an algorithm following the Broyden method for updating the inverse Jacobian. Broyden's method updates the inverse Jacobian without evaluating or inverting a new matrix, providing a large savings in number of total passes through the model. The matrix update is basically a secant-type method and is performed during convergence attempts.

In steady-state operation the solver is used to drive all state time derivatives to zero while simultaneously driving algebraic balance error terms to zero (within a specified tolerance). Transiently, the solver is used to provide simultaneous closed-loop integration of states and closing of algebraic loops. Closed-loop integration entails iteration on the simulation state variables (or state iteration variables) until they are equal to calculated values. This technique provides great flexibility since integration and algebraic balances are handled simultaneously.

3.8 TRANSIENT INTEGRATION METHODS

Rocket engine simulations comprise a set of stiff differential equations that require special methods for integration. Integral methods are efficient when the model time increment is small relative to the time constant associated with the state being integrated. However, as the model time increment is increased, a critical point is reached where convergence failure results. This limits the maximum time increment that can be used.

An alternate method is to use the differential form of the corrector equations instead of the integral form, with the error term being formed as the difference between the actual and calculated derivative. Scaling by the time constant associated with each state is recommended by McLaughlin (Reference 8). The differential method improves convergence when the model time increment is large compared to the state time constant.

The integration routines used on the ROCETS system automatically uses the appropriate form of the corrector equation. The engineering modules approximate the time constant for each state and use this information to select the appropriate integration form. Both the integral and differential forms are incorporated for trapezoidal, first order Gear, and second order Gear corrector equations.

3.9 LINEARIZATION

The ROCETS system was designed to provide accurate linearization about a steady-state or transient operating point. Linearization provides state-space matrices of partial derivatives which can be used for subset model generation, transfer function creation, or multi-variable control analysis.

Generation of accurate partial derivatives is critical to control design, analysis, and development. However, complications arise with large simulations using real properties and many dynamics components. These complications are due to changing iteration variables, the necessity to close algebraic balances, and from discontinuities associated with thermodynamics properties around the saturation dome.

The ROCETS linearization methodology automatically accommodates any change of iteration variables for states and automatically closes all active algebraic balances. With the assumption of small perturbations such that the partials represent a linear model, the algebraic balances and state iteration parameters can be solved from a linear set of equations after all partials have been generated.

The basic set of equations describing a linear model are:

$$Xdot = A * X + B * U$$

$Y = C * X + D * U$

where Xdot is a vector of state derivatives, Y is a vector of outputs, X is the state vector, and U is a vector of inputs. When using pressure and enthalpy (or internal energy) as iteration variables the X's cannot be directly perturbed, so that the matrices cannot be measured directly. However, equations can be measured directly that allows for an analytic substitution of variables and solution of algebraic balances.

Let the nomenclature be that T is a vector of state iteration variables and Z is a vector of algebraic balance independent variables. Then the equations that can be directly measured through perturbations are:

Xdot = A1 * T + Beta1 * U + Alpha2 * Z

Errors = A3 * T + Beta2 * U + Alpha4 * Z

Y = C1 * T + Zeta * U + Theta2 * Z

X = Omega * T

where the matrices represent the appropriate partial derivatives and Errors is a vector of error terms from algebraic balances which are to be zero. The change of variables from T's to X's is accomplished by solving the last equation for T and substituting Omega-inverse * X for T. Let

Alpha1 = A1 * (Omega-inverse)
Alpha3 = A3 * (Omega-inverse)
Theta1 = C1 * (Omega-inverse)

so that the equations become;

Xdot = Alpha1 * X + Beta1 * U + Alpha2 * Z

Errors = Alpha3 * T + Beta2* U + Alpha4 * Z

Y = Theta1 * X + Zeta * U + Theta2 * Z

Now solve for the algebraic balance parameters by noting that the error terms are to be zero. Then Z's are given by:

Z = -(Alpha4-inverse) * (Alpha 3 * X + Beta 2 * U)

Substitution yields:

Xdot = (Alpha1 - Alpha2 * (Alpha4-inverse) * Alpha3) * X +
(Beta1 - Alpha2 * (Alpha4-inverse) * Beta 2) * U

y = (Theta1 - Theta2 * (Alpha4-inverse) * Alpha3) * X +
(Zeta - Theta2 * (Alpha4-inverse) * Beta 2) * U

so that the actual matrices desired are given by;

A = (Alpha1 - Alpha2 * (Alpha4-inverse) * Alpha 3)
B = (Beta1 - Alpha2 * (Alpha-inverse) * Beta2)
C = (Theta1 - Theta2 * (Alpha4-inverse) * Alpha3).
D = (Zeta - Theta2 * (Alpha4-inverse) * Beta2)

In actual practice, an extremely accurate matrix inversion routine is necessary to preserve the integrity of the partials. A standard Gauss-Jordan reduction does not have sufficient accuracy. Therefore a Gauss-Jordan reduction has been combined with a recursion formula to obtain extremely accurate inversion and all matrix operations are performed in double precision.

3.10 RUN TIME ERROR CHECKING

Run-time error checking is provided to warn of possible invalid model conditions and run-time errors can also be used as a transient termination criteria. When curves are read with out-of-range inputs, when internal iterations fail, or any other condition that results in invalid conditions, the user is informed by appropriate messages in a "debug" file and a numerical status indicator (NSI) is set.

Each error location is identified by two eight-character names called "module name" and "module location". The numerical status indicator is sent to a specific value depending on the error severity. Through run-time input, the user can control the NSI at which print will be provided and the NSI which is considered fatal. In addition to the NSI for fatal errors, the user supplies the number of occurrences of each fatal error before execution terminates.

A list of error codes and their corresponding errors follows:

0000	No error
1000 - 2999	Map Extrapolation
3000 - 4999	Input out of Range
5000 - 6999	Internal Iteration Failure
7000 - 9999	Invalid Solution
10000	Invalid Option (No Default), execution halted immediately by ERCK00

3.11 DOCUMENTATION

ROCETS documentation starts in the module source code where the system standards require in the comments cards: A list of all module inputs & outputs with their definitions & units, an engineering description of the module, a list of sub-modules needed, and a history including qualification, author, and revision dates. The next level of documentation is contained in the ROCETS System Design Specification (SDS) of Ref. 4 which contains:

Section 3.4	Documentation
3.4.1	Standards
3.4.2	Engineering Manual
3.4.3	Programmer's Manual
3.4.4	User's Manual
3.4.5	Qualification Test Plans

3.12 ROCETS SYSTEM STATUS

The ROCETS system software library contains approximately 100,000 lines of code of the executive programs (processors) and engineering modules and sub-modules, numerical utilities, and properites. The engineering generic modules represent engine components, and the sub-modules in general provide the specific component performance characteristics. Listed below are the 24 engineering modules representing the engine components with the corresponding 15 sub-modules. A brief description of each follows. Engineering write-ups including all equations are contained in the SDS (Ref 2). A sample pump module "PUMP01" is presented in Appendix B.

ENGINE COMPONENTS	MODULE	SUB-MODULE
PUMP	PUMP01	PMAP04
		PMAP05
		PMAP06
		PMAP07
		PMAP08
TURBINE	TURB01	TBMP03
		TBMP04
		TBMP05
		TBMP06
	TURB02	
TURBOPUMP	ROTR00	
	ROTR01	
PREBURNER	PBRN01	
MAIN CHAMBER	MCHB01	
	QCHM01	
NOZZLE	NOZL00	CDNZ00
	NCLV00	
	QN0Z01	

ENGINE COMPONENTS	MODULE	SUB-MODULE
PLUMBING	PIPE00	
	PIPE01	
	PIPE02	FLPM02
	PIPE03	
	PIPE04	
	PIPE05	PRFP04
		PRFP06
		PRFP07
	PIPE06	
	VOLM00	
	VOLM01	
	VOLM02	
VALVES	VALV00	
POGO SUPPRESSOR	POG000	
GENERAL HEAT TRANSFER	METL00	PRPM01

Module General Description Summary:

MCHB01	H2/02 COMBUSTION AND VOLUME DYNAMICS WITH UNBURN CAPABILITY AND HELIUM DILUTION.
METL00	ROUTINE IS A LUMPED MASS ANALYSIS OF A METAL WITH MULTIPLE HEAT TRANSFER NODES.
NCLV00	ROUTINE FOR THE ENERGY ANALYSIS OF A LUMPED COOLING VOLUME USING DENSITY AND INTERNAL ENERGY AS STATES.
NOZL00	CALCULATES FLOW AND THRUST FOR ISENTROPIC EXPANSION NOZZLE.
PBRN01	PERFECT GAS COMBUSTION (H2/02) WITH VOLUME DYNAMICS AND HELIUM PURGE.
PIPE00	CALCULATES THE FLOW DERIVATIVE AND CRITICAL TIME FOR INCOMPRESSIBLE FLUID FLOW IN PIPE WITH INERTIA AND LOSS.
PIPE01	CALCULATE INCOMPRESSIBLE FLUID FLOW THROUGH A PIPE WITH A LOSS.
PIPE02	CALCULATES COMPRESSIBLE FLUID FLOW THROUGH AN ORIFICE.
PIPE03	CALCULATES THE FLOW DERIVATIVE AND CRITICAL TIME FOR INCOMPRESSIBLE FLUID FLOW IN PIPE WITH LOSS, INERTIA, AND CHANGE IN ELEVATION.
PIPE04	CALCULATES UPSTREAM PRESSURE FOR LIQUID FLOW.
PIPE05	CALCULATES UPSTREAM PRESSURE FOR COMPRESSIBLE FLOW.
PIPE06	CALCULATES UPSTREAM PRESSURE FOR LIQUID FLOW.
POG000	MODELS THE PRIMARY DYNAMICS OF THE POGO SUPPRESSOR.
PUMP01	ROUTINE REPRESENTS A CONSTANT DENSITY PUMP.
QCHM01	CALCULATES HEAT TRANSFER RATE BETWEEN MULTIPLE METAL NODES AND THE HOT GAS FLOW PATH FOR ROCKET MAIN CHAMBER COOLING USING BARTZ CORRELATION.
QN0Z01	CALCULATES HEAT TRANSFER RATE BETWEEN MULTIPLE METAL NODES AND THE HOT GAS FLOW PATH FOR ROCKET NOZZLE COOLING USING BARTZ CORPORATION.
ROTR00	CALCULATES THE ROTOR SPEED DERIVATIVE FOR A ROTOR SYSTEM.
ROTR01	ROTOR WITH BREAKAWAY TORQUE FOR STARTING SIMULATION.
TURB01	ROUTINE IS AN ISENTROPTIC ANALYSIS OF A TURBINE THAT IS DRIVEN BY AN IDEAL GAS.
TURB02	ROUTINE IS AN ISENTROPTIC ANALYSIS OF A TURBINE THAT IS DRIVEN BY A SINGLE CONSTITUENT FLUID.
VALV00	CALCULATES INCOMPRESSIBLE FLUID FLOW THROUGH A VALVE USING LIQUID FLOW CORRELATIONS.
VOLM00	ENERGY AND CONTINUITY ANALYSIS OF A VOLUME WITH ONE INLET MASS FLOW, ONE EXIT MASS FLOW AND ONE HEAT FLOW.
VOLM01	GENERAL MULTI-FLOW LUMPED VOLUME FOR SINGLE CONSTITUANT FLUIDS USING DENSITY AND INTERNAL ENERGY AS STATES.
VOLM02	ROUTINE IS USED FOR VOLUMES WITH MULTI-FLOWS, (BOTH IN AND OUT). IT ASSUMES PERFECT GAS PROPERTIES AND CAN HANDLE FLOW REVERSALS.

CDNZ00	ROUTINE CALCULATES VARIOUS PARAMETERS FOR A CONVERGENT-DIVERGENT NOZZLE.
FLPM02	FLOW PARAMETER BASED ON TOTAL TO TOTAL PRESSURE RATIO AND NUMBER OF "VELOCITY HEADS" LOST.
MACH03	CALCULATES MACH NUMBER FROM FLOW PARAMETER AND GAMMA FOR ADIABATIC FLOW OF A PERFECT GAS.
MACH04	CALCULATES MACH NUMBER FROM AREA RATIO AND GAMMA USING AN ISENTROPIC RELATIONSHIP.
PMAP04	ROUTINE DETERMINES THE PUMP CHARACTERISTICS FROM A MAP FOR THE ROCKETDYNE HIGH PRESSURE FUEL.PUMP.
PMAP05	ROUTINE DETERMINES THE PUMP CHARACTERISTICS FROM A MAP FOR THE ROCKETDYNE LOW PRESSURE FUEL PUMP.
PMAP06	ROUTINE DETERMINES THE PUMP CHARACTERISTICS FROM A MAP FOR THE ROCKETDYNE HIGH PRESSURE OXIDIZER PUMP.
PAMP07	ROUTINE DETERMINES THE PUMP CHARACTERISTICS FORM A MAP FOR THE ROCKETDYNE LOW PRESSURE OXIIDIZER PUMP.
PMAP08	ROUTINE DETERMINES THE PUMP CHARACTERISTICS FROM A MAP FOR THE ROCKETDYNE PREBURNER OXIDIZER PUMP.
PRFP04	ROUTINE GIVES PRESSURE RATIO (TOTAL TO TOTAL) FROM FLOW PARAMETER, RKLS AND GAMMA USING TOTAL TEMPERATURE AND TOTAL PRESSURE.
PRFP06	ROUTINE GIVES PRESSURE RATIO (TOTAL TO STATIC) FROM FLOW PARAMETER USING TOTAL TEMPERATURE AND STATIC PRESSURE.
PRFP07	ROUTINE GIVES PRESSURE RATIO (TOTAL TO TOTAL) FROM FLOW PARAMETER, RKLS AND GAMMA USING TOTAL TEMPERATURE AND TOTAL DOWNSTREAM PRESSURE.
TBMP03	ROUTINE DETERMINES THE TURBINE CHARACTERISTICS FROM MAPS FOR THE ROCKETDYNE HIGH PRESSURE FUEL TURBINE.
TBMP04	ROUTINE DETERMINES THE TURBINE CHARACTERISTICS FROM MAPS FOR THE ROCKETDYNE LOW PRESSURE FUEL TURBINE.
TBMP05	ROUTINE DETERMINES THE TURBINE CHARACTERISTICS FROM MAPS FOR THE ROCKETDYNE HIGH PRESSURE OXIDIZER TURBINE.
TBMP06	ROUTINE DETERMINES THE TURBINE CHARACTERISTICS FROM MAPS FOR THE ROCKETDYNE LOW PRESSURE OXIDIZER TURBINE.

Listed below are the 29 ROCETS utilities subroutines which perform functions like integration, table reads, and error checks. A brief description of each follows the list.

UTILITY FUNCTION	ROUTINE
DIRECT MODEL EXECUTION	XPR000
ERROR CHECKS	ERCK00
	ERCK01
	ERCK02
	ERCK03
WRITES DUMMY GUESS ROUTINE	DMGS01
INPUT SCHEDULE PROCESSOR	SPR000
INTEGREATION	RINT01
MATRIX OPERATIONS	DPVN01
	DPVN03
	MTMU02
OUTPUT	LWRT01
	PRPL01
	WRIT01
PARTIAL GENERATION	LMRD01
	PART01
	PRTB01
SOLVER	ITER05
	SMIT03
	SSBL04
SORT LIST	SORTA4
TABLE READ	SUNB00
	SUNB01
	SUNB03
	CPMR02
	CPMR04
	CPMR05
	CPMR06
UNITS CONVERSION	UNIT00

Numerical/System Utilities General Descirption Summary

CPMR02	CORRESPONDING POINT BI-VARIENT MAP READER.
CPMR04	CORRESPONDING POINT BI-VARIENT MAP READER THAT ALSO RETURNS PARTIALS.
CPMR05	CORRESPONDING POINT BI-VARIENT MAP READER WITH NEW MAP INDEX POINTER.
CPMR06	CORRESPONDING POINT BI-VARIENT MAP READER THAT ALSO RETURNS PARTIALS WITH NEW MAP INDEX DEFINITIONS.
DMGS01	CREATES A DUMMY GUESS ROUTINE WITH LOCATIONS FOR THE APPROPRIATE REQUIRED GUESSES FOR A NEWLY CONFIGURED MODEL,
DPVN01	DOUBLE PRECISION MATRIX INVERSION USING GAUSS-JORDAN ELIMINATION WITH PARTIAL MAXIMUM PIVOTING.
DPNV03	DOUBLE PRECISION MATRIX INVERSION USING COMBINATION GAUSS-JORDAN AND RECURSION.FORMULA.
ERCK00	ONE OF A PACKAGE OF FOUR ROUTINES TO PROVIDE RUN-TIME ERROR CHECKING (SEE ERCK01, 02, AND 03). THIS ROUTINE IS CALLED AT THE POINT OF A ERROR TO PASS IN THE ERROR NUMBER ALONG WITH INFORMATION TO IDENTIFY THE ERROR. ERCK01 IS USED TO PROVIDE ERROR PRINT AND ERCK02 IS USED TO SPECIFY PRINT AND KILL LEVELS.
ERCK01	ONE OF A PACKAGE OF FOUR ROUTINES TO PROVIDE RUN-TIME ERROR CHECKING (SEE ERCK00, 02, AND 04). THIS ROUTINE IS CALLED TO PROVIDE OUTPUTING ERROR STATUS IN NORMAL TRANSIENT PRINT/PLOT AND FOR PRINTING END-OF-RUN ERROR STATUS. ERCK00 IS USED TO ENTER ERRORS AND ERCK02 IS USED TO SPECIFY PRINT AND KILL LEVELS.
ERCK02	ONE OF A PACKAGE OF FOUR ROUTINES TO PROVIDE RUN-TIME ERROR CHECKING (SEE ERCK00, 01, AND 03). THIS ROUTINE IS CALLED BY THE USER TO SPECIFY THE PRINT LEVEL, KILL LEVEL, AND NUMBER OF FATAL ERRORS ALLOWED. ERCK00 IS USED AT THE POINT OF AN ERROR TO SET THE STATUS AND ERCK01 IS USED TO PROVIDE ERROR PRINT.
ERCK03	ONE OF A PACKAGE OF FOUR ROUTINES TO PROVIDE RUN-TIME ERROR CHECKING (SEE ERCK00 - ERCK02). THIS ROUTINE IS CALLED TO SEARCH THE NAME ARRAYS AND RETURN THE LOCATION. IT IS A UTILITY FOR THE OTHER ROUTINES AND IS NOT USER CALLABLE.
ITER05	SECANT METHOD ITERATION WHICH CAN BE USED WITH NESTED LOOPS
LMRD01	ROUTINE WRITES OUT THE RESULTS OF THE LINEARIZATION OF ROCETS SIMULATION.
MTMU02	DOUBLE PRECISION MATRIX MULTIPLICATION.
OPCK01	TRANSFORMS THE PROPERTY OPTION CHARACTER FIELD INTO A STANDARD FORM AND TO READ THE INDEPENDENT & DEPENDENT PROPERTY NAMES.
PART01	MEASURES AND CHECKS THE PARTIALS FOR THE ROCKET ENGINE TRANSIENT SIMULATION SYSTEM.

PRPL01	A GENERALIZED PRINT/PLOT ROUTINE FOR TRANSIENT DECKS PROVIDING COLUMNAR PRINT, INTERFACING FOR PLOTS, AND OPTIONAL USER PRINT/PLOT HEADER SPECIFICATION AS WELL AS TAILORED PRINT FORMAT.
PRTB01	ROUTINE PERTURBATES THE INDEPENDENT VARIABLE (X) AND MEASURE THE PARTIAL OF THE VECTOR OF DEPENDENT VARIABLE (Y) WITH RESPECT TO INDEPENDENT VARIABLE.
RINT01	ROUTINE PERFORMS THE CLOSED LOOP INTEGRATION FOR THE ROCKET ENGINE TRANSIENT SIMULATION SYSTEM. A CHOICE OF THREE IMPLICIT INTEGRATION TECHNIQUES (TRAPEZOIDAL, FIRST ORDER GEAR, AND SECOND ORDER GEAR) IS INCLUDED. AN EULER INTEGRATION IS ALSO AVAILABLE.
SMIT03	ROUTINE SOLVES A SET OF SIMULTANEOUS NONLINEAR EQUATIONS USING NEWTON'S METHOD WITH BROYDEN'S INVERSE JACOBIAN UPDATE SCHEME.
SORTA4	ROUTINE SORTS A LIST OF CHARACTER WORDS.
SPR000	SCHEDULE PROCESSOR WHICH PROCESSES RUN-TIME SCHEDULES FOR ROCTES SIMULATION SYSTEM.
SSBL04	THE COMPANION ROUTINE TO THE TRANSIENT INTEGRATION ROUTINE (RINT01). SSBL04 IS USED TO ACHIEVE A STEADY STATE BALANCE OF STATES AND CONVERGENCE OF ADDITIONAL BALANCES.
SUNBOO	UNI-VARIENT OR BI-VARIENT SEPARATE INTERPOLATION MAP READER WITH OPTIONAL EXTRAPOLATION FOR OUT-OF-RANGE DATA.
SUNB01	UNIVARIENT OR BIVARIENT SUNBAR-TYPE MAP READER WITH OPTION TO READ MAP IN ANY DIRECTION AND EITHER EXTRAPOLATE OR RETURN CORNER VALUES.
SUNB03	TRI-VARIENT SEPARATE INTERPOLATION MAP READER WITH OPTIONAL EXTRAPOLATION FOR OUT-OF-RANGE DATA AND MULTI-DIRECTION READ OPTION.
UNITOO	ROUTINE SETS CONVERSION FACTORS AND CONSTANTS IN THE UNITS COMMON BASED ON THE SYSTEM REQUESTED.
WRIT01	INTERFACE ROUTINE FOR PRP01 TO WRITE PLOT FILE FOR MSFC IBM 3080 SYSTEM IN THE ORIGINAL UNIVAC FILE FORMAT.
XPRO01	PURPOSE: INTERFACES WITH SYSTEM COMMONS TO DIRECT MODEL EXECUTION. THIS ROUTINE CALLS NECESSARY NUMERICAL ROUTINES AND DIRECTS EXECUTION IN THE CALLING PROGRAM BY MEANS OF AN OUTPUT SIGNAL.

The properties contained in the ROCETS system include combustion for $H_2/0_2$, hydrogen, oxygen, helium, nitrogen, methane, and various metals. They are listed below and followed with a brief description.

PROPERTY	MODULE	SUB-MODULE
COMBUSTION	HGPROP	COMB01
		COMB02
		ZGAS00
		ZZH201
HYDROGEN	H2PROP	HHCP05
		HHCV05
		HHPK05
		HHPS05
		HHPU05
		HPSH01
		HPUT05
		HRHP05
		HRHP06
		HRUP05
OXYGEN	02PROP	OHCP05
		OHCV05
		OHPK05
		OHPS05
		HHPU05
		HPSH01
		HPUT05
		HRHP05
		HRHP06
		HRUP05
HELIUM	HEPROP	EHPS05
		EHPT05
		ERHP05
NITROGEN	N2PROP	NHPS05
		NPHT05
		NRHP05
METHANE	N2PROP	MCPT05
		MCVT05
		MHPS05
		MHPT05
		MRHP05
METALS		PRPM01

Properties General Description Summary:

COMB01	OBTAINS HOT GAS TRANSPORT PROPERTIES FOR H2/02 COMBUSTION PRODUCTS.
COMB02	PERFECT GAS COMBUSTION (H2/02) WITH HELIUM DILUTION.
EHPS05	OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR HELIUM FROM AN ENTHALPY, PRESSURE, ENTROPY MAP.
EPTH05	OBTAINS THEMOPHYSICAL FLUID PROPERTIES FOR HELIUM FROM A PRESSURE, ENTHALPY TEMPERATURE MAP.
ERHP05	OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR HELIUM FROM A DENSITY, ENTHALPY, PRESSURE MAP.
HEPROP	SUBROUTINE ACCESSES HELIUM PROPERTIES VIA HELIUM PROPERTY MAPS.
HGPROP	MAIN DRIVER FOR HOT GAS PROPERTIES (H2/02 COMBUSTION PRODUCTS) FROM MAPS (INCLUDES HELIUM PURGE EXCEPT FOR Z, MU, AND K)
HHCP05	OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR PARA-HYDROGEN FROM AN ENTHALPY, PRESSURE, CONSTANT PRESSURE SPECIFIC HEAT MAP
HHCV05	OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR PARA-HYDROGEN FROM AN ENTHALPY, PRESSURE, CONSTANT VOLUME SPECIFIC HEAT MAP
HHPK05	OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR PARA-HYDROGEN FROM AN ENTHALPY, PRESSURE, THERMAL CONDUCTIVITY MAP
HHPS05	OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR PARA-HYDROGEN FROM AN ENTHALPY, PRESSURE, ENTROPY MAP.
HHPU05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR PARA-HYDROGEN FORM AN ENTHALPY, PRESSURE, VISCOSITY MAP.
HPSH01	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR PARA-HYDROGEN FROM A PRESSURE, ENTROPY, ENTHALPY MAP.
HPUT05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR PARA-HYDROGEN FROM A DENSITY, ENTHALPY, PRESSURE MAP.
HRHP05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR PARA-HYDROGEN FROM A DENSITY, ENTHALPY, PRESSURE MAP.
HRHP06	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR PARA-HYDROGEN FROM A DENSITY, ENTHALPY, PRESSURE MAP.
HRUP05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR PARA-HYDROGEN FROM A DENSITY, INTERNAL ENERGY, PRESSURE MAP.
H2PROP	SUBROUTINE ACCESSES HYDROGEN PROPERTIES VIA HYDROGEN PROPERTY MAPS.
MCPT05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR METHANE FROM A TEMPERATURE, PRESSURE, CONSTANT PRESSURE SPECIFIC HEAT MAP.
MCVT05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR METHANE FROM A TEMPERATURE, PRESSURE, CONSTANT VOLUME SPECIFIC HEAT MAP.
MEPROPR	SUBROUTINE ACCESSES METHANE PROPERTIES VIA METHANE PROPERTY MAPS.
MHPS05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR METHANE FROM AN ENTHALPY, PRESSURE, ENTROPY MAP.

MPHT05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR METHANE FROM AN ENTHALPY, PRESSURE, TEMPERATURE MAP.
MRHP05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR METHANE FROM A DENSITY, ENTHALPY, PRESSURE MAP.
NHPS05	SUBROUTINE OBTAINS OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR NITROGEN FROM AN ENTHALPY, PRESSURE, ENTHROPY MAP.
NPHT05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR NITROGEN FROM A PRESSURE, ENTHALPY TEMPERATURE MAP
NRHP05	SUBROUTINE SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR NITROGEN FROM A DENSITY, ENTHALPY, PRESSURE MAP
N2PROP	SUBROUTINE ACCESSES NITROGEN PROPERTIES VIA NITROGEN PROPERTY MAPS.
OHCP05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR OXYGEN FORM AN ENTHALPY, PRESSURE, CONSTANT PRESSURE SPECIFIC HEAT MAP
OHCV05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR OXYGEN FROM AN ENTHALPY, PRESSURE, CONSTANT VOLUME SPECIFIC HEAT MAP
OHPK05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR OXYGEN FROM AN ENTHALPY, PRESSURE, ENTROPY MAP.
OHPS05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR OXYGEN FROM AN ENTHALPY, PRESSURE, ENTROPY MAP.
OHPU05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR OXYGEN FROM A PRESSURE, ENTROPY, ENTHALPY MAP.
OPUT05	SUBROUTINE THERMOPHYSICAL FLUID PROPERTIES FOR OXYGEN FROM A PRESSURE, INTERNAL ENERGY, TEMPERATURE MAP
ORHP05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR OXYGEN FROM A DENSITY, ENTHALPY, PRESSURE MAP
ORHP06	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES AND PARTIALS FOR OXYGEN FROM A DENSITY, ENTHALPY, PRESSURE MAP
ORUP05	SUBROUTINE OBTAINS THERMOPHYSICAL FLUID PROPERTIES FOR OXYGEN FROM A DENSITY, INTERNAL ENERGY, PRESSURE MAP
02PROP	SUBROUTINE ACCESSES OXYGEN PROPERTIES VIA OXYGEN PROPERTY MAPS
PRPM01	SUBROUTINE ACCESSES OXYGEN PROPERTIES VIA OXYGEN PROPERTY MAPS.
PRPM01	SUBROUTINE GIVES THE SPECIFIC HEAT AND CONDUCTIVITY OF VARIOUS METALS AS A FUNCTION OF TEMPERATURE.
ZGAS00	SUBROUTINE CALCULATES THE REAL GAS COMPRESSIBILITY FACTOR FOR H2/H202
ZZH201	SUBROUTINE CALCULATES COMPRESSIBILITY FACTOR FOR H2.

SECTION IV
TTBE MODEL

Two models of the Technology Test Bed Engine (TTBE) were generated under the program. The initial model was a simple model without boost turbopumps, and with a simulation complexity of 55 state variables and 2 algebraic loops. After testing and verification of this model, a detailed TTBE model with the boost turbopumps and a POGO system was configured with 122 state variables and 14 algebraic loops.

4.1 SIMPLE TTBE SIMULATION

A simple model of the Technology Test Bed Engine (TTBE) was generated as the initial system verification vehicle for the simulation system. Figure 4-1 shows a schematic of the simple TTBE along with the 42 specific stations in the simulation. By using generic code, only the following 13 component modules were required by the simulation:

1. INJT00 - Main Injector
2. MCHB00 - Main Chamber
3. MIXR00 - Flow Mixer
4. NOZL00 - Nozzle Thrust Calculations
5. PBRN00 - Preburner
6. PIPE00 - Incompressible flow with inertia
7. PIPE01 - Incompressible flow without inertia
8. PIPE02 - Compressible flow without inertia
9. PUM00 - Polytropic Pump
10. ROTR00 - Rotor Torque Balance/Speed Derivative
11. SPLT00 - Flow Splitter
12. TURB00 - Turbine
13. VOLM00 - Volume

The modules described above were configured into the simple TTBE simulation along with required property relationships and numerical utilities. There were 55 state variables and 2 algebraic loops required in the simulation as shown in Table 4-1. State derivatives and outputs are calculated form model inputs and states.

Using initial guesses from data of the Digital Transient Model (DTM) of Reference 2 at 100% RPL, SMITE successfully obtained all TTBE model state derivatives and algebraic loop parameters to within specific tolerances. This demonstrated the capability of the ROCETS system to converge a rocket simulation to a steady-state point without running a transient. Transient capability was demonstrated by running the simple TTBE simulation with small perturbations of valve areas about the 100% RPL point. The results of these tests are presented in Section 5.0 - System testing and verification.

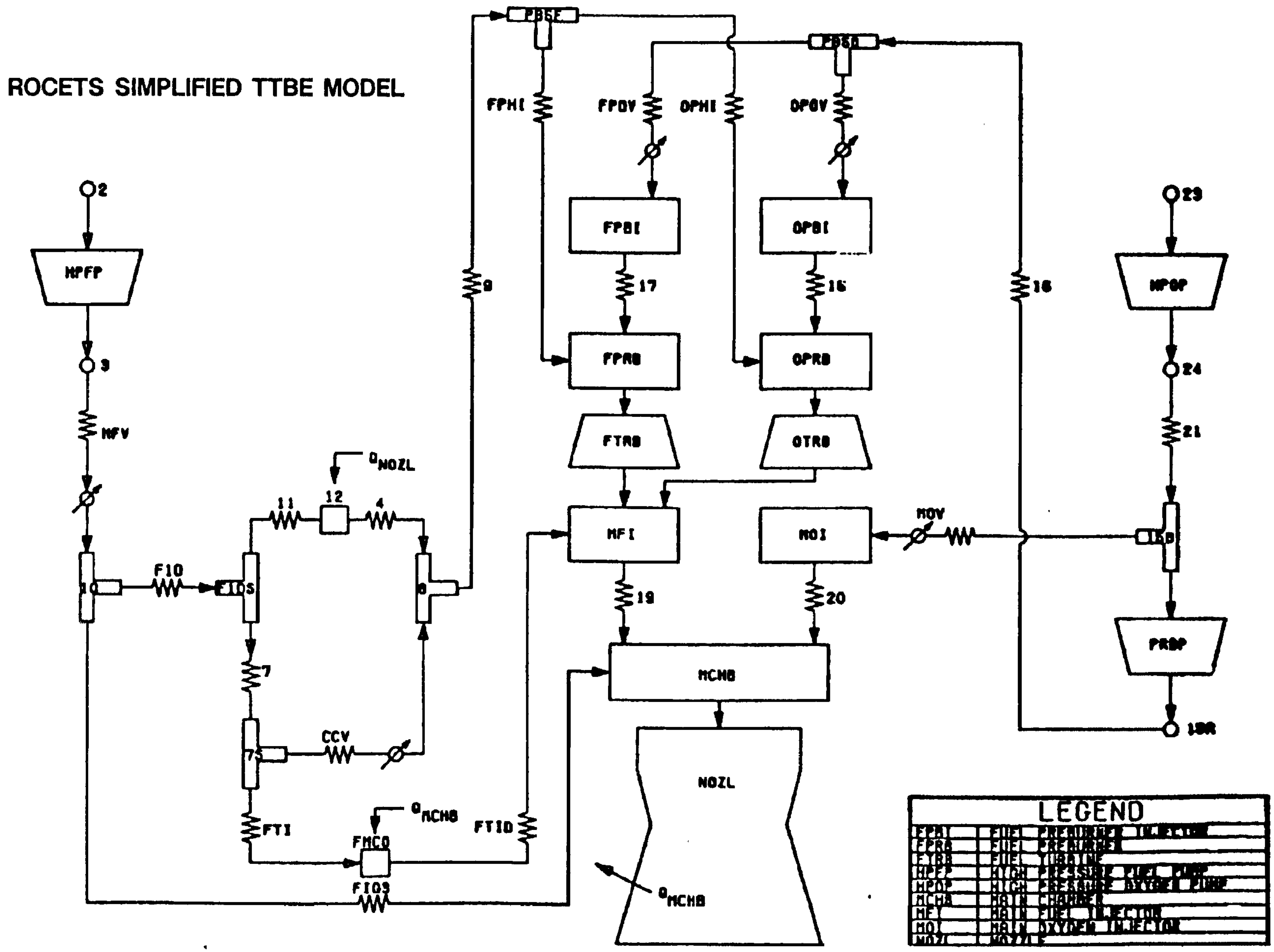

Figure 4-1. ROCETS Simplified TTBE Model

TABLE 4-1 - Simple TTBE Simulation States and Algebraic Loops

STATE	DESCRIPTION
OFRFPRB	Fuel Preburner Oxidizer Fraction
OFRMCHB	Main Chamber Oxidizer Fraction
OFRMFI	Main Fuel Injector Oxidizer Fraction
OFROPRB	Oxidizer Preburner Oxidizer Fraction
RHOFMCO	Volume FMCO Density
RHOFPBI	Fuel Preburner Injector Density
RHOFPRB	Oxidizer Preburner Injector Density
RHOF10S	Volume F10S Density
RHOMCHB	Main Chamber Density
RHOMFI	Main Fuel Injector Density
RHOMOI	Main Oxidizer Injector Density
RHOOPBI	Oxidizer Preburner Injector Density
RHOOPRB	Oxidizer Preburner Density
RHOPBSF	Preburner Fuel Splitter Density
RHOPBSO	Preburner Oxidizer Splitter Density
RHO10	Volume 10 Density
RHO12	Volume 12 Density
RHO15B	Volume 15B Density
RHO8	Volume 8 Density
SNF2	Fuel Turbomachinery Speed
SN02	Oxidizer Turbomachinery Speed
TTFPRB	Fuel Preburner Temperature
TTMCHB	Main Chamber Temperature
TTMFI	Main Fuel Injector Temperature
TTOPRB	Oxidizer Preburner Temperature
UTFMCO	Volume F10S Internal Energy
UTMOI	Main Oxidizer Injector Internal Energy
UTOPBI	Oxidizer Preburner Injector Internal Energy
UTPBSF	Preburner Fuel Splitter Internal Energy
UTPBSO	Preburner Oxidizer Splitter Internal Energy
UT10	Volume 10 Internal Energy
UT12	Volume 12 Internal Energy
UT15B	Volume 15B Internal Energy
UT7S	Volume 7S Internal Energy
UT8	Volume 8 Internal Energy
WCCV	Coolant Control Valve Flow
WFPHI	Fuel Preburner Fuel Flow
WFPOV	Fuel Preburner Oxidizer Flow
WFTI	Line FTI Flow
WFTID	Line FTID Flow
WF10	Line F10 Flow
WMFV	Main Fuel Valve Flow
WOPHI	Oxidizer Preburner Fuel Flow
WOPOV	Oxidizer Preburner Oxidizer Flow
W11	Line 11 Flow
W16	Line 16 Flow

W20	Line 20 Flow
W21	Line 21 Flow
W4	Line 4 Flow
W7	Line 7 Flow Flow
W9	Line 9 Flow

ITERATION VARIABLE	DESCRIPTION
WFTRB	Fuel Turbine Flow - Iterated until equal to calculated value
WOTRB	LOX Turbine flow - Iterated until equal to calculated value

4.2 DETAILED TTBE SIMULATION

After successful verification of the simple TTBE model, a detailed TTBE model simulation was developed. The approach was to model the lox side and test, then the fuel side and test, then the hot gas system and test, and finally connect the three sub-systems and test. A schematic of the entire engine simulation is presented on Figure 4-2. There are 122 states and 14 additional balances in the simulation. Each of the station names are labeled on the schematic.

A description of the modules used to configure the TTBE along with a list of the TTBE schematic names that use that particular module follows.

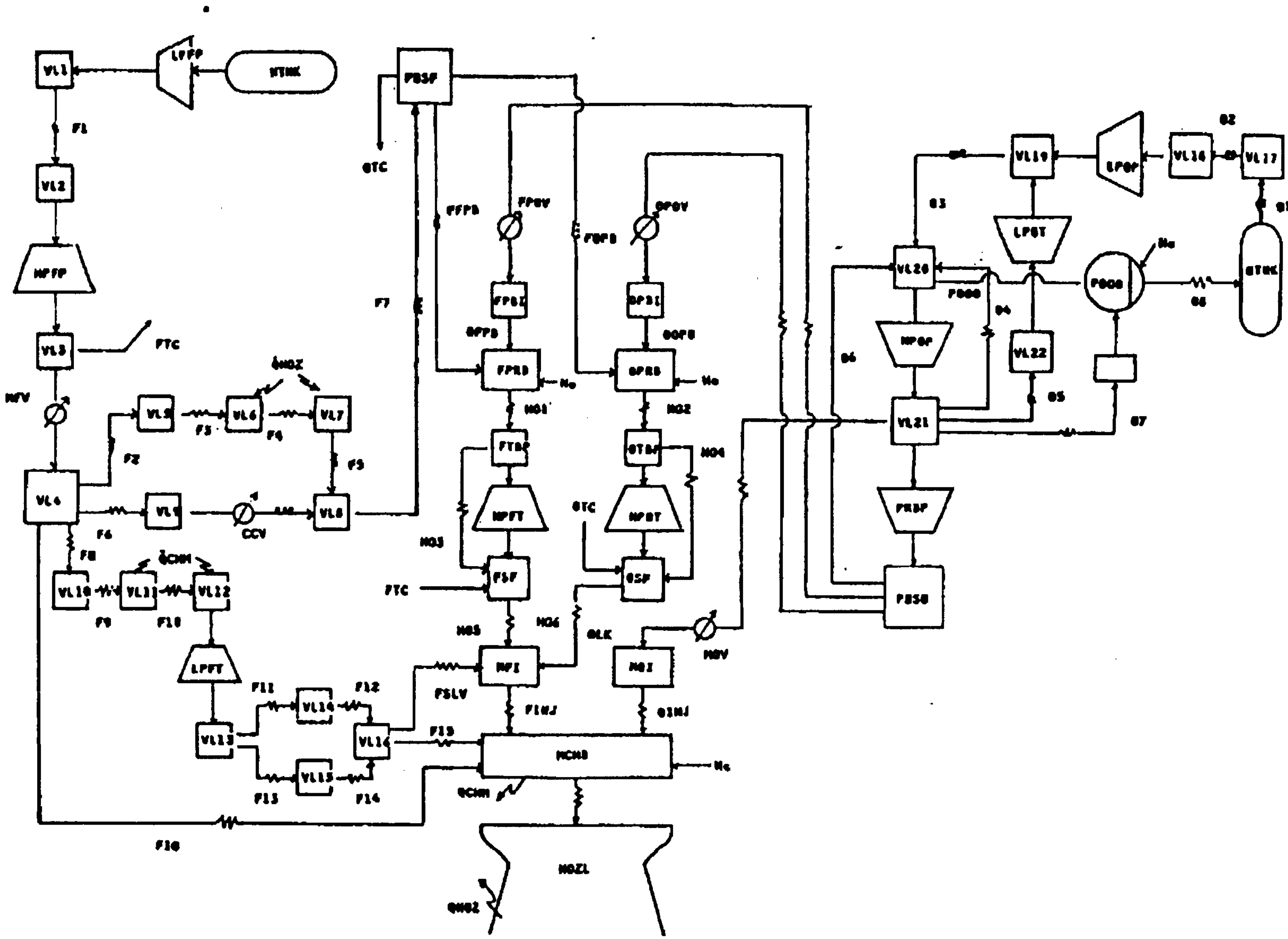

Figure 4-2. ROCETS TTBE Detailed Configuration

ORIGINAL PAGE IS
OF POOR QUALITY

Pipe Modules

Six pipe modules were used to configure the TTBE. They are PIPE00, PIPE01, PIPE02, PIPE03, PIPE05, PIPE06. Following is a list of the TTBE schematic pipe names that use the corresponding pipe routines, a description of each module, and the states of each module.

Module PIPE00

Schematic Names: F1, F2, F5, F6, F7, F8, FFPB, FOPB, 02, 03, 05 OFPB, OOPB, OINJ

Description:

This module calculates the flow derivative for incompressible fluid flow in a pipe with inertial effects.

States and State Derivatives:

PIPE00 has flow as a state and calculates the flow derivative.

Module PIPE01

Schematic Names: F3, F4, F11, F12, F13, F14, F15, FSLV, FIG, FTC, OTC, 04, 06, 07, MOV, FPOV, OPOV

Description:

This module calculates flow through a pipe for an incompressible fluid.

States and State Derivatives:

PIPE01 calculates flow but it is not treated as a state. This module does not treat any parameters as states and therefore does not calculate any state derivatives.

Module PIPE02

Schematic Names: HG3, HG4, OLK

Description:

This module calculates flow through a pipe for a compressible fluid.

States and State Derivatives:

PIPE02 calculates flow but it is not treated as a state. This module does not treat any a parameters as states and therefore does not calculate any state derivatives.

Module PIPE03

Schematic Names: 01

Description:

This module calculates the flow derivative for incompressible fluid flow in a pipe with inertial effects and elevation change.

States and State Derivatives:

PIPE03 has flow as a state and calculates the flow derivative.

Module PIPE05

Schematic Names: HG1, HG2, HG3, HG6, FINJ, HG5

Description:

This module calculates upstream pressure for compressible flow.

States and State Derivatives:

PIPE05 uses flow but it is not treated as a state. This module does not treat any parameters as states and therefore does not calculate any state derivatives.

Module PIPE06

Schematic Names: F9, F10

Description:

This module calculates upstream pressure for incompressible flow.

States and State Derivatives:

PIPE06 uses flow but it is not treated as a state. This module does not treat any parameters as states and therefore does not calculate any state derivatives.

Valve Module

One valve module, VALV00, was used to configure the TTBE. Following is a list of the TTBE schematic valve names that use the VALV00 routine, a description of the VALV00 module, and the states of the VALV00 module.

Module VALV00

Schematic Names: MFV, CCV

Description:

This module calculates flow through VALVE for an incompressible fluid.

States and State Derivatives:

VALV00 calculates flow but it is not treated as a state. This module does not treat any a parameters as states and therefore does not calculate any state derivatives.

Volume Module

Module VOLM00

Schematic Names: VL1, VL2, VL5, VL9, VL10, VL14, VL15, VL17, VL18, VL22

Description:

This module performs a continuity and energy analysis of a volume for pure fluids with one inlet flow, one exit flow, and one heat flow.

States and State Derivatives:

VOLM00 has density and internal energy as states. Corresponding derivatives are calculated for each state.

Module VOLM01

Schematic Names: VL3, VL4, VL8, VL13, VL16, VL19, VL20, VL21, PBSF, PBSO

Description:

The module performs a continuity and energy analysis of a volume for pure fluids with multiple inlet flows, multiple exit flows, and multiple heat flows.

States and State Derivatives:

VOLM01 has density and internal energy as states. Corresponding derivatives are calculated for each state.

Module VOLM02

Schematic Names: FTBP, OTBP, FSF, OSF, MFI

Description:

This module performs a continuity and energy analysis of a volume for perfect gases with oxygen, hydrogen, and helium as possible constituents. The analysis is performed with multiple inlet flows, multiple exit flows, and multiple heat flows.

States and State Derivatives:

VOLM02 has pressure, temperature, oxidizer fraction, and helium fraction as states. Corresponding derivatives are calculated for each state.

Module NCLV00

Schematic NAMES: VL6, VL7, VL11, VL12

Description:

This module models the cooling of the chamber and nozzle. The module performs a continuity and heat transfer analysis of a volume of pure fluids with one inlet flow, one exit flow, multiple node metal temperatures, and multiple node heat transfer surface areas. The heat transfer rate is calculated for each node.

States and State Derivatives:

NCLV00 has density and internal energy as states. Corresponding derivatives are calculated for each state.

Rotor Module

The inertial and transient speed effects for turbopumps are modeled by the ROTR00 module which mechanically links each turbopump together. Following is a list of the TTBE schematic turbopump names that use the rotor routine, a description of the ROTR00 module, and the states of the ROTR00 module.

Module ROTR00

Schematic Names: LPFP/LPFT, HPFP/HPFT, LPOP/LPOT, (HPOP and PRBP)/HPOT

Description:

Given supply torques, required torques, rotative speed and the overall polar moment of inertia, this routine calculates the speed derivative for the given system.

States and state Derivatives:

ROTR00 has rotative speed as a state and calculates the corresponding speed derivative.

Pump Module

One pump module, PUMP01, was used to configure the TTBE. Following is a list of the TTBE schematic pump names that use the pump routine, a description of the PUMP01 module, and the states of the PUMP01 module.

Module PUMP01

Schematic Names: LPFP, HPFP, LPOP, HPOP, PRBP

Description:

By assessing the appropriate pump performance map, this module calculates exit enthalpy, exit pressure, and required torque for a constant density pump.

States and State Derivatives:

PUMP01 uses rotative speed but it is not treated as a state (see module ROTR00 above). This module does not treat any parameters as states and therefore does not calculate any state derivatives.

Turbine Modules

Two turbine modules were used to configure the TTBE and they are TURB01 and TURB02. Following is a list of the TTBE schematic turbine names that use the corresponding turbine routines, a description of each module, and the states of each module.

Module TURB01

Schematic Name: LPFT, HPFT, HPOT

Description:

By accessing the appropriate turbine performance map, using isentropic efficiency this module calculates exit enthalpy, supply torque, and required turbine flowrate for a turbine driven by a perfect gas.

States and State Derivatives:

TURB01 uses rotative speed but it is not treated as a state (see module ROTR00 above). This module does not treat any parameters as states and therefore does not calculate any state derivatives.

Module TURB02

Schematic Name: LPOT

Description:

By accessing the appropriate turbine performance map, using intropic efficiency this module calculates exit enthalpy, supply torque, and required turbine flowrate for turbine driven by a liquid.

States and State Derivatives:

TURB02 uses rotative speed but it is not treated as a state (see module ROTR00 above). This module does not treat any parameters as states and therefore does not calculate any state derivatives.

POGO Module

A POGO module, POGO00, was used to configure the POGO suppression system for the TTBE. Following is a list of the TTBE schematic component names that use the POGO routine, a description of the POGO module, and the states of the POGO module.

Module POGO00

Schematic Name: POGO

Description:

This module models the POGO suppression system. Given the oxygen-side conditions, this module calculates the required exit oxygen flowrate and appropriate derivatives.

States and State Derivatives:

POGO00 has pressure, liquid oxygen flowrate, liquid oxygen mass, and helium fraction as states. Corresponding derivatives are calculated for each state.

Main Chamber Combustion Module

One main chamber combustion module, MCHB01, was used to configure the TTBE. Following is the TTBE schematic name for the main chamber combustion, a description of the MCHB01 module, and the states of the MCHB01 module.

Module MCHB01

Schematic Names: MCHB

Description:

This module models perfect gas hydrogen/oxygen combustion with helium dilution, unburn capability, and volume dynamics.

States and State Derivatives:

MCHB01 has pressure, temperature, oxidizer fraction, and helium fraction as states. Corresponding derivatives are calculated for each state.

Preburner Module

One preburner module, PBRN01, was used to configure the TTBE. Following is a list of the TTBE schematic preburner names that use the PBRN01 routine, a description of the PBRN01 module, and the states of the PBRN01 module.

Module PBRN01

Schematic Names: FPRB, OPRB

Description:

This module models perfect gas hydrogen/oxygen combustion with helium dilution, and volume dynamics.

States and State Derivatives:

PBRN01 has pressure, temperature, oxidizer fraction, and helium fraction as states. Corresponding derivatives are calculated for each state.

METAL MODULE

The transient metal temperature effecters are modeled by the METL00 module. Following is a list of the TTBE schematic component names that use the METL00 routine, a description of the METL00 module, and the states of the METL00 module.

Module METL00

Schematic Names: VL6/QDOTNZ1, VL7/QDONTNZ2. VL11/QDOTCHM1, VL12/QDOTCHM2.

Description:

Given the mass of the metal and the temperature of the metal, this routine calculates the metal temperature derivative for each of the given multiple nodes.

States and State Derivatives:

METL00 has metal temperature as a state and calculates the corresponding derivative.

Chamber Hot Side Heat Transfer Module

One chamber heat transfer module, QCHM01, was used to configure the TTBE. Following is a list of the TTBE schematic names that used the QCHM01 routine, a description of the QCHM01 module, and the states of QCHM01 module.

Module QCHM01

Schematic Names: QDOTCHM1, QDOTCHM2

Description:

This module calculates the multiple node heat flowrate through the chamber wall from the combustion gases using a Bartz empirical correlation.

States and State Derivatives:

WCHM01 uses metal temperature but it is not treated as a state (See METL00 above). This module does not treat any parameters as states and therefore does not calculate any state derivatives.

Nozzle Hot Side Heat Transfer Module

One nozzle heat transfer module, QNOZ01, was used to configure the TTBE. Following is a list of the TTBE schematic names that use the QNOZ01 routine, a description of the QNOZ01 module, and the states of QNOZ01 module.

Module QNOZ01

Schematic Names: QDOTNZ1, QDOTNZ2

Description:

This module calculates the multiple node heat flowrate through the nozzle wall from the combustion gases using a Bartz empirical correlation.

States and State Derivatives:

QNOZ01 uses metal temperature but it is not treated as a state (See METL00 above). This module does not treat any parameters as states and therefore does not calculate any state derivatives.

Module NOZL00

Schematic Names: NOZL

Description:

This module calculates the gross thrust, flow through the nozzle, and the exit mach number using isentropic relations.

States and State Derivatives:

NOZL00 calculates flow but it is not treated as a state. This module does not treat any parameters as states and therefore does not calculate any state derivatives.

TTBE States

The TTBE simulation has 122 states. The states, a description of the states, the module names where the states are differentiated, and the corresponding schematic names are listed for the fuel side in Table 4-1, for the oxidizer side in Table 4-2, and for the hot gas side in Table 4-3.

The thermodynamic states, density and internal energy are difficult parameters to iterate. To overcome this difficulty pressure and enthalpy are iterated to solve the density and internal energy corrector equations. The thermodynamic states, a description of the thermodynamic states, the corresponding iteration parameters, and a description of the corresponding iteration parameters are listed for the fuel side in Table 4-4 and for the oxidizer side in Table 4-5.

TTBE Additional Required Balances

Fourteen additional balances are required to close the loop on pressures and temperatures to achieve a power balance. The iteration parameters and the two balance parameters along with their descriptions are listed in Table 4-6.

Table 4-1. Fuel Side States

State	Description	Module Name	Schematic Name
SNFL	Low-pressure fuel turbopump speed	ROTR00	LPFP/LPFT
RHOVL1	Density Of Vol. 1	VOLM00	VL1
UTVL1	Internal Energy Of Vol. 1	VOLM00	VL1
WF1	Flow Rate Through Fuel Line 1	PIPE00	F1
RHOVL2	Density Of Vol. 2	VOLM00	VL2
UTVL2	Internal Energy Of Vol. 2	VOLM00	VL2
SNFH	High-Pressure Fuel Turbopump Speed	ROTR00	HPFP/HPFT
RHOVL3	Density of Vol. 3	VOLM01	VL3
UTVL3	Internal Energy of Vol. 3	VOLM01	VL3
RHOVL4	Density of Vol. 4	VO.M01	VL4
UTVL4	Internal Energy of Vol. 4	VOLMO1	VL4
WF2	Flow Rate Through Fuel Line 2	PIPE00	F2
RHOVL5	Density of Vol. 5	VOLM00	VL5
UTVL5	Internal Energy of Vol 5	VOLM00	VL5
RHOVL6	Density of Vol. 6	NCLV00	VL6
UTVL6	Internal Energy of Vol. 6	NCLV00	VL6
TMMTL1	Metal Temp. for Qdotnoz1	METL00	Qdotnoz1
RHOVL7	Density of Vol. 7	NCLV00	VL7
UTVL7	Internal Energy of Vol. 7	NCLV00	VL7
TMMTL2	Metal Temp. for Qdotnoz2	METL00	Qdotnoz2
WF5	Flow rate through fuel line 5	PIPE00	F5
RHOVL8	Density of vol. 8	VOLMO1	VL8
UTVL8	Internal energy of vol. 8	VOLM01	VL8
RHOVL9	Density of vol. 9	VOLM00	VL9
UTVL9	Internal energy of vol. 9	VOLM009	VL9
WF6	Flow rate through fuel line 6	PIPE00	F6
WF8	Flow rate thorough fuel line 8	PIPE00	F8
RHOVL10	Density of vol. 10	VOLM00	VL10
UTVL10	Internal energy of vol. 10	VOLM00	VL10
RHOVL11	Density of vol. 11	NCLV00	VL11
UTVL11	Internal energy of vol 11	NCLV00	VL11
TMMTL3	Metal temp. for Qdotchm1	METL00	Qdotchm1
RHOVL12	Density of vol. 12	NCLV00	VL12
UTVL12	Internal energy of vol. 12	NCLV00	VL12
TMMTL4	metal temp. for Qdotchm2	METL00	Qdotchm2
RHOVL13	Density of vol. 13	VOLM01	VL13

Table 4-1. Fuel Side States (Continued)

State	Description	Module Name	Schematic Name
UTVL13	Internal energy of vol. 13	VOLM01	VL13
RHOVL14	Density of vol. 14	VOLM00	VL14
UTVL14	Internal energy of vol. 14	VOLM00	VL14
RHOVL15	Density of vol. 15	VOLM00	VL15
UTVL15	Internal energy of vol. 15	VOLM00	VL15
RHOVL16	Density of vol. 6	VOLM01	VL16
UTVL16	Internal energy of vol. 16	VOLM01	VL16
WF7	Flow rate through fuel line 7	PIPE00	F7
PHOPBSF	Density of PB fuel splitter vol.	VOLM01	PBSF
UTBSF	Internal energy of PB fuel splitter vol.	VOLM01	PBSF

Table 4-2. Oxidizer Side States

State	Description	Module Name	Schematic Name
WO1	Flow rate through oxid. line 1	PIPE03	01
RHOVL17	Density of vol. 17	VOLM00	VL17
UTVL17	Internal energy of vol. 17	VOLM00	VL17
WO2	Flow rate through oxid. line 2	PIPE00	02
RHOVL18	Density of vol. 18	VOLM00	VL18
UTVL18	Internal energy of vol. 18	VOLM00	VL18
SNOL	Low-pressure oxid. turbopump speed	ROTR00	LPOP/LPOT
RHOVL19	Density of vol. 19	VOLM01	VL19
UTVL19	Internal energy of vol. 19	VOLM01	VL19
WO3	Flow rate through oxid. line 3	PIPE00	03
RHOVL20	Density of vol. 20	VOLM01	VL20
UTVL20	Internal energy of vol. 20	VOLM01	VL20
SNOH	High-pressure oxid. turbopump speed	ROTROO	HPOP/HPOT
RHOVL21	Density of vol. 21	VOLM09	VL21
UTVL21	Internal energy of vol. 21	VOLM01	VL21
WO5	Flow rate through oxid. line 5	PIPE00	05
RHOVL22	Density of vol. 22	VOLM00	VL22
UTVL22	Internal energy of vol. 22	VOLM00	VL22
WPOGO	Liquid oxid. flow into POGO	POGO00	POGO
PTPOGO	Pressure in POGO vol.	POGO00	POGO
HFRPOGO	Helium frac. in POGO vol	POGO00	POGO
RMLPOGO	Mass of liquid in POGO vol.	POGO00	POGO
WO9	Flow rate through oxid. line 9	PIPE00	O9
RHOVL23	Density of vol. 23	VOLM00	VL23
UTVL23	Internal energy of vol. 23	VOLM00	VLWE
RHOMOI	Density of main oxid. inj. vol.	VOLM00	MOI
UTMOI	Internal energy of main oxid. inj. vol.	VOLM00	MOI
WOINJ	Flow rate through oxid. line OINJ	PIPE00	OINJ
RHOPBSO	Density of PB oxid. splitter vol.	VOLM01	PBSO
UTPBSO	Internal energy of PBoxid. splitter vol.	VOLM01	PBSO
WO10	Flow rate through oxid. line 10	PIPE00	010
WO11	Flow rate through oxid. line 11	PIPE00	011
RHOVL24	Density of vol. 24	VOLM00	VL24
UTVL24	Internal energy of vol. 24	VOLM00	VL24
RHOVL25	Density of vol. 25	VOLM00	VL25
UTL25	Internal energy of vol. 25	VOLM00	VL25

Table 4-3. Hot Gas Side States

State	Description	Module Name	Schematic Name
RHOFPBI	Density of fuel PB injector vol.	VOLM00	FPBI
UTFPBI	Internal energy of fuel PB injector vol.	VOLM00	FPBI
RHOOPBI	Density of oxid. PB injector vol.	VOLM00	OPBI
UTOPBI	Internal energy of oxid. PB injector vol.	VOLM00	OPBI
WFFPB	Fuel flow rate to fuel PB	PIPE00	FFPB
WFOPB	Fuel flow rate to oxidizer PB	PIPE00	FOPB
WOFPB	Oxid. flow rate to fuel PB	PIPE00	OFPB
WOOPB	Oxid. flow rate to oxid. PB	PIPE00	PPPB
PTFPRB	Pressure in fuel PB vol.	PBRN01	FPRB
TTFPRB	Temperature in fuel PB vol.	PBRN01	FPRB
OFRFPRB	Oxid. fraction in fuel PB vol.	PBRN01	FPRB
HFRFPRB	Helium fraction in fuel PB vol.	PBRN01	FPRB
PTOPRB	Pressure in oxid. PB vol.	PBRN01	OPRB
TTOPRB	Temperature in oxid. PB vol.	PBRN01	OPRB
OFROPRB	Oxid. fraction in oxid. PB vol.	PBRN01	OPRB
HFROPRB	Helium fraction in oxid. PB vol.	PBRN01	OPRB
PTFTBP	Pressure in fuel turb. bypass vol.	VOLM02	FTBP
TTFTBP	Temperature in fuel turb. bypass vol.	VOLM02	FTBP
OFRFTBP	Oxid. fraction in fuel turb. bypass vol.	VOLM02	FTBP
HFRFTBP	helium fraction in fuel turb. bypass vol.	VOLM02	FTBP
PTOTBP	Pressure in oxid. turb. bypass vol.	VOLM02	OTBP
TTOTBP	Temperature in oxid. turb. bypass vol.	VOLM02	OTBP
OFROTBP	Oxid. fraction in oxid. turb. bypass vol.	VOLM02	OTBP
HFROTBP	Helium fraction in oxid. turb. bypass vol.	VOLM02	OTBP
PTFSF	Pressure in fuel secondary flow vol.	VOLM02	FSF
TTFSF	Temperature in fuel secondary flow vol.	VOLM02	FSF
OFRFSF	Oxid. fraction in fuel secondary flow vol.	VOLM02	FSF
HFRFSF	Helium fraction in fuel secondary flow vol.	VOLM02	FSF
PTOSF	Prewssure in oxid. secondary flow vol.	VOLM02	OSF
TTOSF	Temperature in oxid. secondary flow vol.	VOLM02	OSF
OFROSF	Oxid. fraction in oxid. secondary flow vol.	VOLM02	OSF

Table 4-3. Hot Gas Side States (Continued)

State	Description	Module Name	Schematic Name
HFROSF	Helium fraction in oxid. secondary flow vol.	VOLM02	OSF
PTMFI	Pressure in main fuel inj. vol.	VOLM02	MFI
TTMFI	Temperature in main fuel inj. vol.	VOLM02	MFI
OFRMFI	Oxid. fraction in main fuel inj. vol.	VOLM02	MFI
HFRMFI	Helium fraction in main fuel inj. vol.	VOLM02	MFI
PTMCHB	Pressure in main chamber vol.	MCHB01	MCHB
TTMCHB	Temperature in main chamber vol.	MCHB01	MCHB
OFRMCHB	Oxid. fraction in main chamber vol.	MCHB01	MCHB
HFRMCHB	Helium fraction in main chamber vol.	MCHB01	MCHB

Table 4-4. Fuel Side Thermodynamic States and Their Corresponding Iteration Parameters

Thermo-dynamic State	Thermodynamic State Description	Corresp. Iteration Parameter	Corresponding Iteration Parameter description
RHOVL1	Density of vol. 1	PTVL1	Pressure in vol. 1
UTVL1	Internal energy of vol. 1	HTVL1	Ethalpy of vol. 1
RHOVL2	Density of vol. 2	PTVL2	Pressure in vol. 2
UTVL2	Internal enegy of vol. 2	HTVL2	Enthalpy of vol. 2
RHOVL3	Density of vol 3	PTVL3	Pressure in vol. 3
UTVL3	Internal energy of vol. 3	HTVL3	Enthalpy of vol. 3
RHOVL4	Density of vol. 4	PTVL4	Pressure in vol. 4
UTVL4	Internal energy of vol. 4	HTVL4	Enthalpy of vol. 4
RHOVL5	Density of vol. 5	PTVL5	Pressure in vol. 5
UTVL5	Internal energy of vol. 5	HTVL5	Enthalpy of vol. 5
RHOVL6	Density of vol. 6	PTVL6	Pressure in vol. 6
UTVL6	Internal energy of vol. 6	HTVL6	Enthalpy of vol. 6
RHOVL7	Density of vol. 7	PTVL7	Pressure of vol. 7
UTVL7	Internal energy of vol. 7	HTVL7	Enthalpy of vol. 7
RHOVL8	Density of vol. 8	PTVL8	Pressure of vol. 8
UTVL8	Internal energy of vol. 8	HTVL8	Enthalpy of vol. 8
RHOVL9	Density of vol. 9	PTVL9	Pressure of vol. 9
UTVL9	Internal energy of vol. 9	HTVL9	Enthalpy of vol. 9
RHOVL10	Density of vol. 10	PTVL10	Pressure of vol. 10
UTVL10	Internal energy of vol. 10	HTVL10	Enthalpy of vol. 10
RHOVL11	Density of vol. 11	PTVL11	Pressure of vol. 11
UTVL11	Internal energy of vol. 11	HTVL11	Enthalpy of vol. 11
RHOVL12	Density of vol. 12	PTVL12	Pressure of vol. 12
UTVL12	Internal energy of vol. 12	HTVL12	Enthalpy of vol. 12
RHOVL13	Density of vol. 13	PTVL13	Pressure of vol. 13
UTVL13	Internal energy of vol. 13	HTVL13	Enthalpy of vol. 13
RHOVL14	Density of vol. 14.	PTVL14	Pressure of vol. 14
UTVL14	Internal energy of vol. 14	HTVL14	Enthalpy of vol. 14
RHOVL15	Density of vol. 15	PTVL15	Pressure of vol. 15
UTVL15	Internal energy of vol. 15	HTVL15	Enthalpy of vol. 15
RHOVL16	Density of vol. 16	PTVL16	Pressure of vol. 16
UTVL16	Internal energy of vol. 16	HTVL16	Enthalpy of vol. 16
RHOPBSF	Density of PB fuel splitter vol.	PTPBSF	Pressure in PB fuel splitter vol.
UTPBSF	Internal energy of PB fuel splitter vol.	HTPBSF	Enthalpy of PB fuel splitter vol.
RHOFPBI	Density of fuel PB injector vol.	PTFPBI	Pressure in fuel PB Injector vol.
UTFPBI	Internal energy of fuel PB injector vol.	HTFPBI	Enthalpy of fuel PB Injector vol.

Table 4-5. Oxidizer Side Thermodynamic States and Their Corresponding Iteration Parameters

Thermo- dynamic State	Thermodynamic State Description	Corresp. Iteration Parameter	Corresponding Iteration Parameter description
RHOVL17	Density of vol. 17	PTVL17	Pressure in vol. 17
UTVL17	Internal energy of vol. 17	HTVL17	Enthalpy of vol. 17
RHOVL18	Density of vol. 18	PTVL18	Pressure in vol. 18
UTVL18	Internal energy of vol. 18	HTVL18	Enthalpy of vol. 18
RHOVL19	Density of vol. 19	PTVL19	Pressure in vol. 19
UTVL19	Internal energy of vol. 19	HTVL19	Enthalpy of vol. 19
RHOVL20	Density of vol. 20	PTVL20	Pressure in vol. 20
UTVL20	Internal energy of vol. 20	HTVL20	Enthalpy of vol. 20
UTVL21	Density of vol. 21	PTVL21	Pressure in vol. 21
RHOVL21	Internal energy of vol. 21	HTVL21	Enthalpy of vol. 21
UTVL22	Density of vol. 22	PTVL22	Pressure in vol. 22
RHOVL22	Internal energy of vol. 22	HTVL22	Enthalpy of vol. 22
UTVL23	Density of vol. 23	PTVL23	Pressure in vol. 23
RHOVL23	Internal energy of vol. 23	HTVL23	Enthalpy of vol. 23
RHOMOI	Density of main oxid. injector vol.	PTMOI	Pressure in main oxid injector vol.
UTMOI	Internal energy of main oxid. injector vol.	HTMOI	Enthalpy of main oxid injector vol.
RHOPBSO	Density of PB oxid. splitter vol.	PTPBSO	Pressure in PB oxid. splitter vol.
UTPBSO	Internal energy of PB oxid splitter vol.	HTPBSO	Enthalpy of PB oxid. splitter vol.
RHOVL24	Density of vol. 24	PTVL24	Pressure in vol. 24
UTVL24	Internal energy of vol. 24	HTVL23	Enthalpy of vol. 24
RHOVL25	Density of vol. 25	PTVL25	Pressure in vol. 25
UTVL25	Internal energy of vol. 25	HTVL23	Enthalpy of vol. 25
RHOOPBI	Density	PTOPBI	Pressure in oxid. PB injector vol.
UTOPBI	Internal energy of oxid. PB injector vol.	HTOPBI	Enthalpy of oxid. PB injector vol.

Table 4-6. TTBE Model Additional Required Balances

Iterated Parameter	Iterated Parameter Description	Balance Parameter 1	Balance Parameter 1 Description	Balance Parameter 2	Balance Parameter 2 Description
WLPFP	LPFP flow	PTVL1	Vol. 1 pressure	PTLPFD	LPFP Disch. Pressure
WHPFP	HPFP Flow	PTVL3	Vol. 3 Pressure	PTHPFD	HPFP Disch. Pressure
WLPOP	LPOP Flow	PTVL19	Vol. 19 Pressure	PTLPOD	LPOP Disch. Pressure
WHPOP	HPOP Flow	PTVL21	Vol. 21 Pressure	PTHPOD	HPOP Disch. Pressure
WPRBP	PBRP Flow	PTPBSO	Vol. PBSO Pressure	PTPBPD	PRBP Disch. Pressure
TTHTFD	HPFT Disch. Temp.	TTHTFD	HPFT Disch. Temp.	TTHTFDC	Calc. HPFT Disch. Temp.
TTHTOD	HPOT Disch. Temp.	TTHTOD	HPOT Disch. Temp.	TTHTODC	Calc. HPOT Disch. Temp.
WHG1	FPRB Disch. Flow	PTFPRB	FPRB Pressure	PTFPRBC	Calc. FPRB Pressure
WHG2	OPRB Disch. Flow	PTOPRB	OPRB Pressure	PTOPRBC	Calc. OPRB Pressure
WHG5	FSF to MFI Flow	PTFSF	FSF Pressure	PTFSFC	Calc. FSF Pressure
WHG6	OSF to MFI Flow	PTOSF	OSF Pressure	PTOSFC	Calc. OSF Pressure
WFINJ	FINJ Flow	PTMFI	MFI Pressure	PTMFIC	Calc. MFI Pressure
WF9	FL10 to VL11 Flow	PTVL10	Vol. 10 Pressure	PTVL10C	Calc. Vol. 10 Pressure
WF10	FL11 to VL12 Flow	PTVL11	Vol. 11 Pressure	PTVL11C	Calc. Vol. 11 Pressure

This page left intentionally blank.

SECTION V
SYSTEM TESTING AND VERIFICATION

System simulations were generated and operated to verify the proper operation of the ROCETS system. Math model simulations utilized for this testing included:

- Simple TTBE Model
- Detailed LOX side Model
- Detailed TTBE Model
- Sub-set Model

Because the models did not use all the same equations and calculations of the SSME DTM (Reference 2), the resulting predictions were not expected to reproduce exactly the DTM. However, comparisons to the DTM results were used as a guide that the models were functioning properly in the ROCETS system.

5.1 SIMPLE TTBE MODEL TEST

The simple TTBE Model (as defined in Section 4.1) had 55 state variables and 2 algebraic loops. The initial test was obtaining a steady state balance at 100% RPL by driving the state derivatives to zero and closing the algebraic loops (to within a specific tolerance). This demonstrated the capability of the ROCETS system to converge a rocket simulation to a steady-state point without running a transient. Transient capability was demonstrated by running the simple TTBE simulation with small perturbations of preburner valve areas about the 100% RPL point.

Additional algebraic loops (balances) were placed in the model to set preburner valve coefficients at points other than 100% RPL. The fuel preburner valve coefficient was iterated until chamber pressure (PTMCHB) was equal to the request:

$$PTMCHB_{Request} = PTMCHB_{100\ RPL} \times \%RPL$$

The LOX preburner valve coefficient was iterated until chamber oxidizer fraction (OFRMCHB) was equal to a constant value of 0.865. (this is equivalent to a mixture ratio of 6.407). Figure 5-1 shows main chamber pressure and oxidizer fraction as a function of RPL. A series of steady-state points between 60% and 119% RPL were then run with the solver iterating on valve coefficients until chamber pressure and LOX fraction were satisfied. this demonstrated the ability of the model to use the solver as a means to set a model parameter based on an input constraint. Output of the run gives a reference steady-state operating characteristic for the model and provides data for SMITE guess curves. Figure 5-2 shows turbine speeds as a function of RPL.

SIMPLE TTBE SIMULATION
STEADY-STATE CHARACTERISTICS

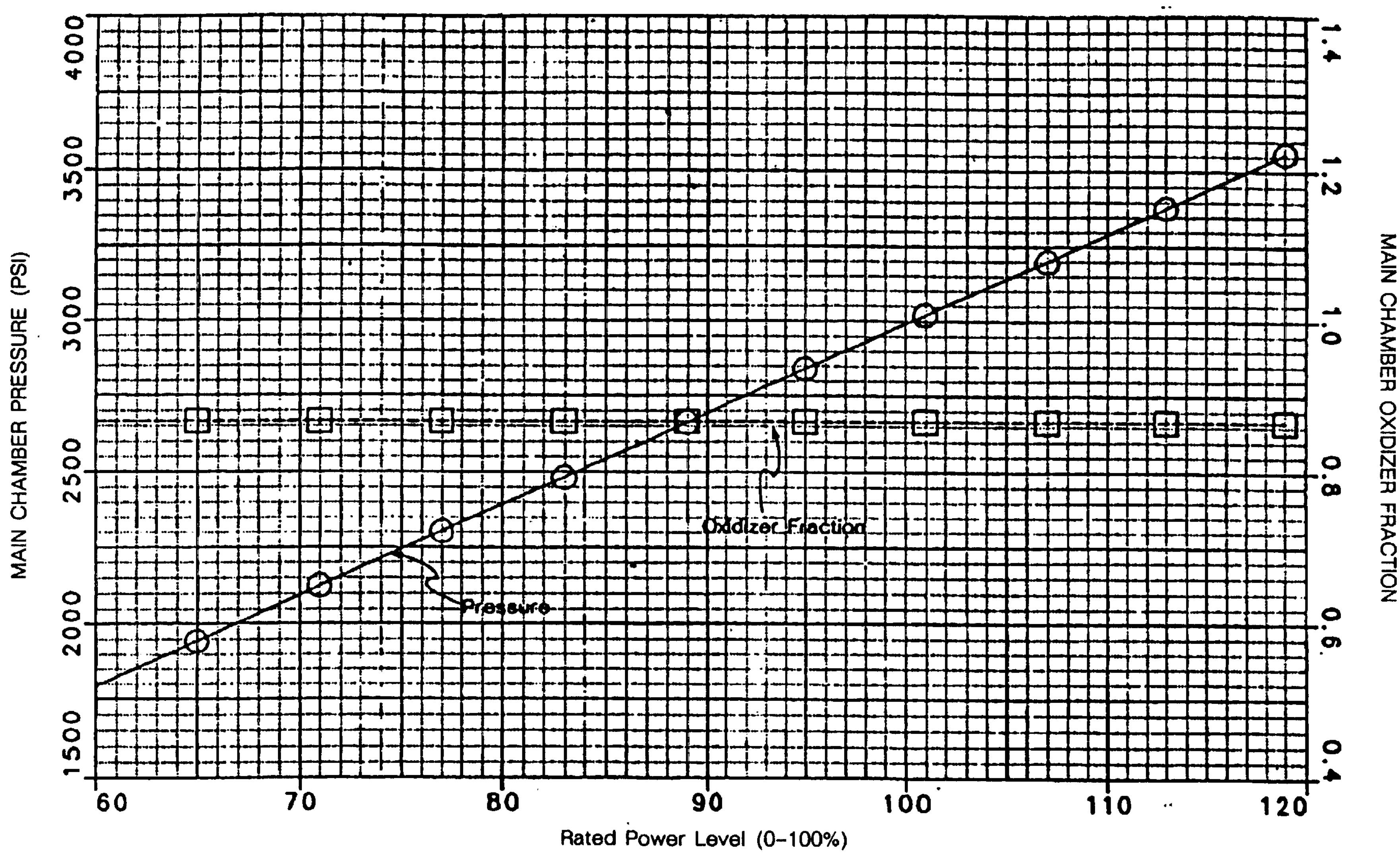

Figure 5-1 - Main Chamber Conditions as a Funciton of RPL

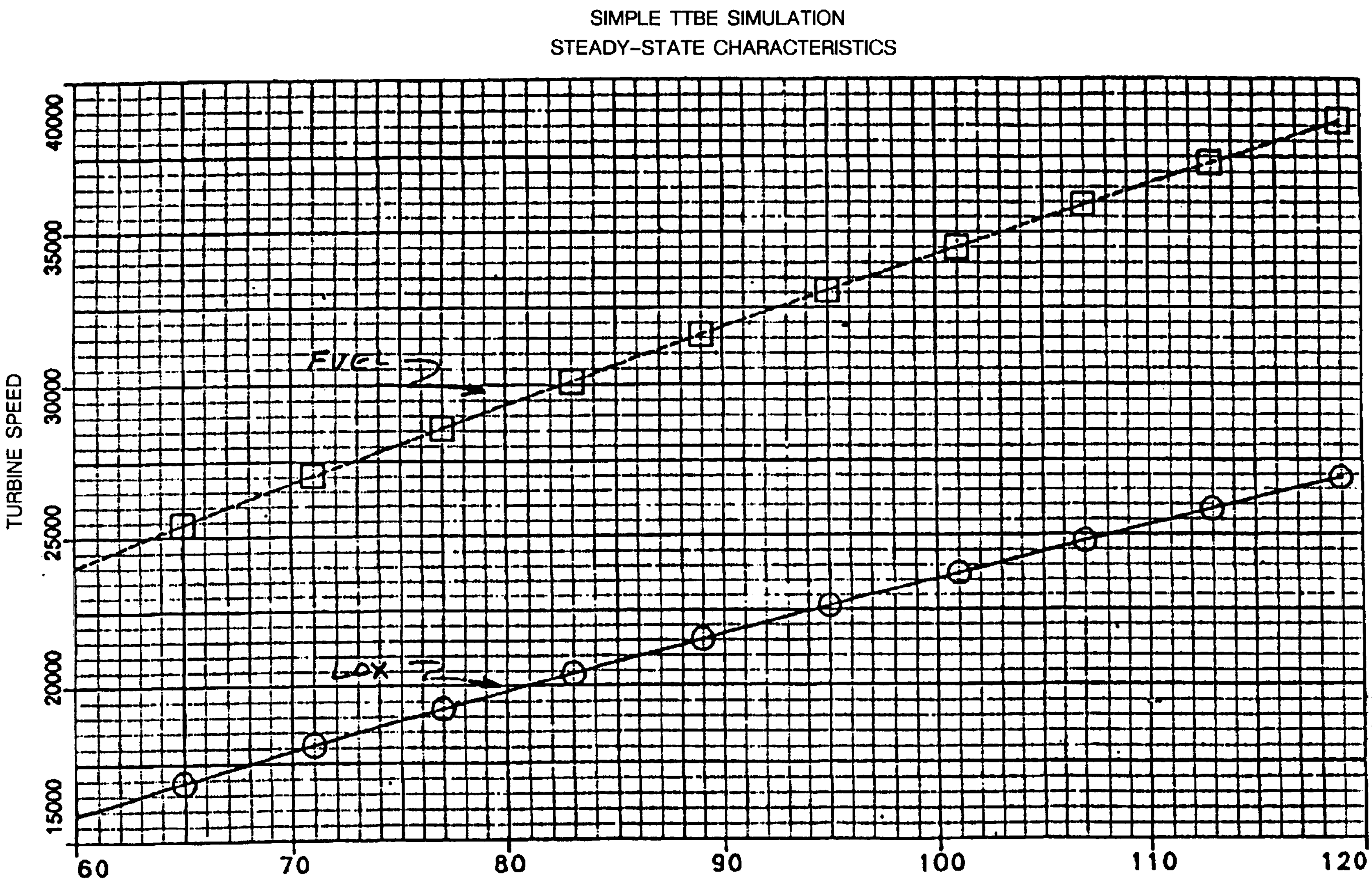

Figure 5-2 - Turbine Speeds as a Function of RPL

The steady-state data as a function of RPL was tabularized and used to construct an open-loop control with RPL request input and valve data area requests calculated from the table. The valve request were put through a first order lag to simulate actuator dynamics. Gross throttle transients were run by inputting an RPL request as a function of time and using the open-loop control to provide valve areas. Figures 5-3 and 5-4 show results of a transient run from 100% to 65% RPL decel and figures 5-5 and 5-6 show a transient run from 65% to 109% RPL. These test with the simple TTBE model demonstrated the ability of ROCETS to obtain a steady-state balance, obtain steady-state valve schedules based on imposed constraints, and to operate transiently.

SIMPLE.TTBE SIMULATION
STEADY-STATE CHARACTERISTICS

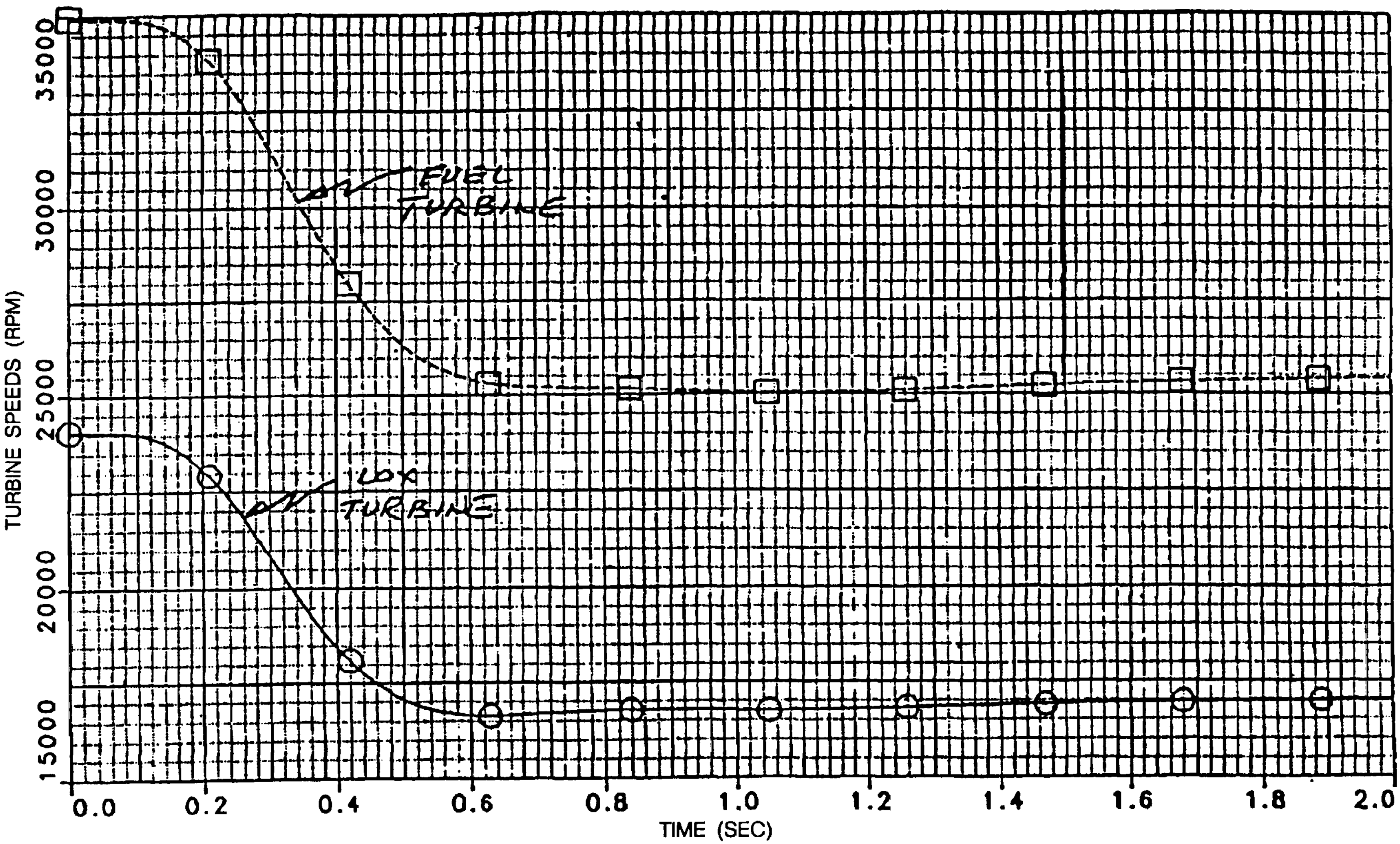

Figure 5-3 - Turbine Speeds as a Function of Time

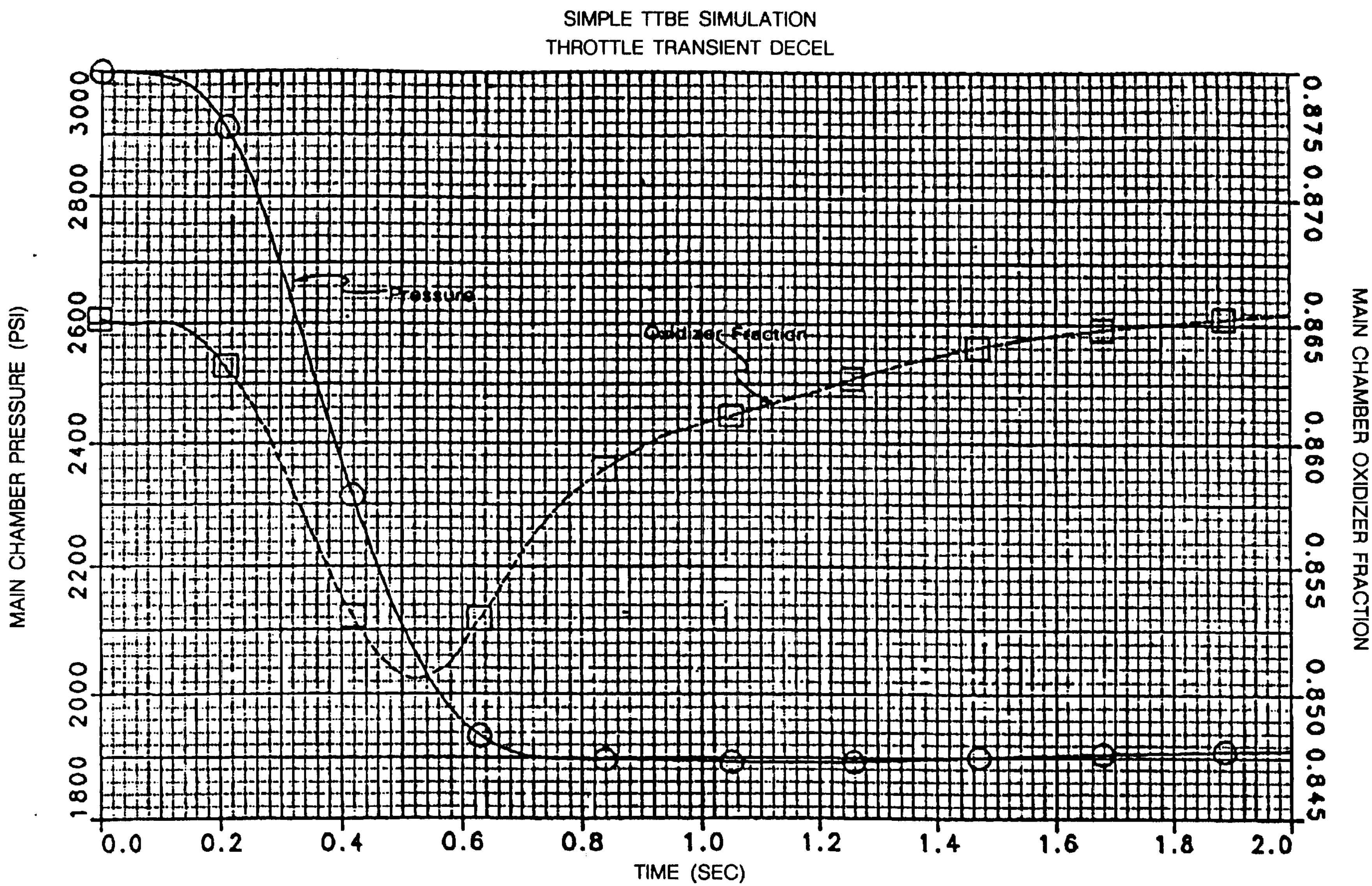

Figure 5–4 – Main Chamber Conditions as a Function of Time

SIMPLE TTBE SIMULATION
THROTTLE TRANSIENT ACCEL

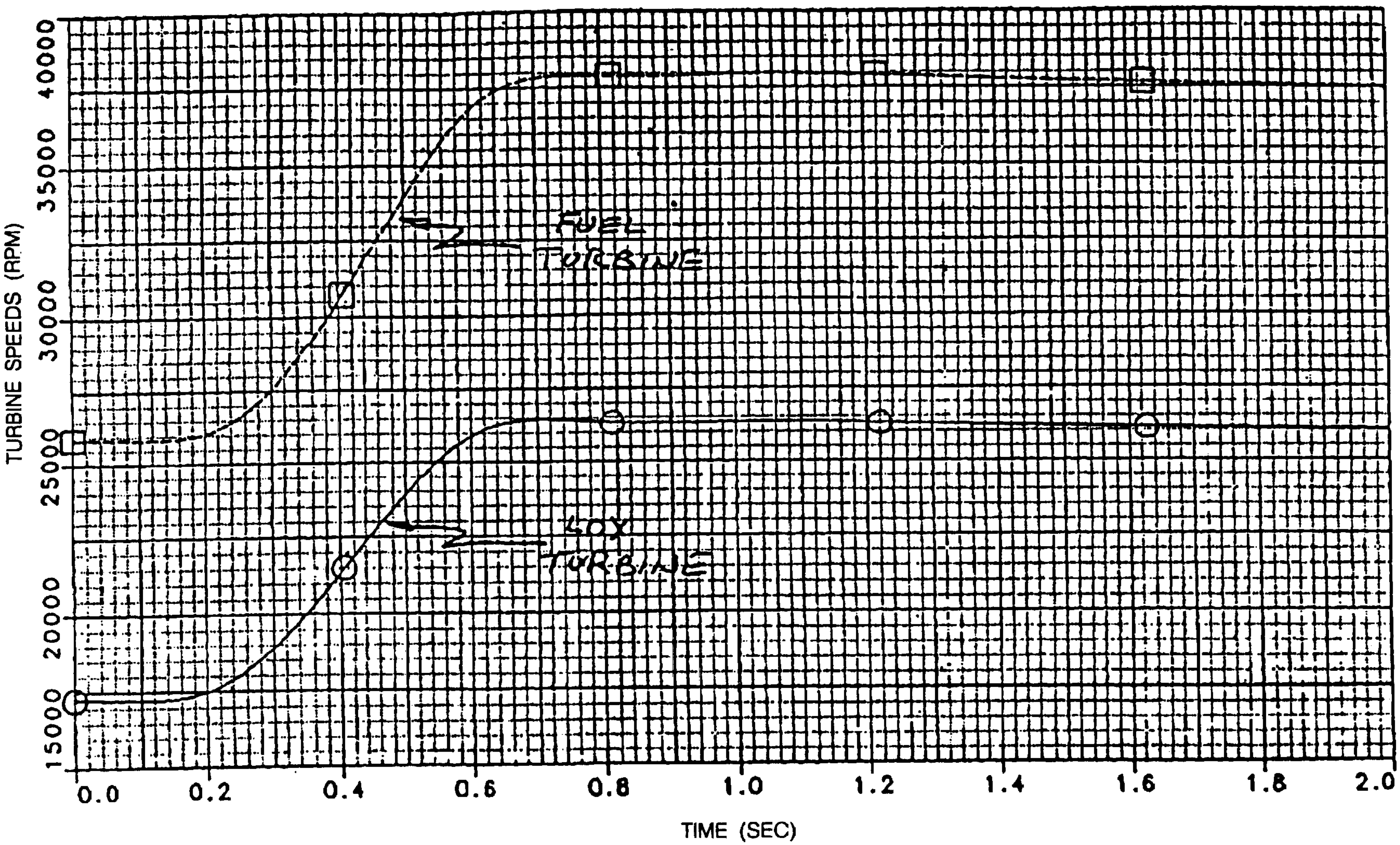

Figure 5-5 - Turbine Speeds as a Function of Time

SIMPLE TTBE SIMULATION
THROTTLE TRANSIENT ACCEL

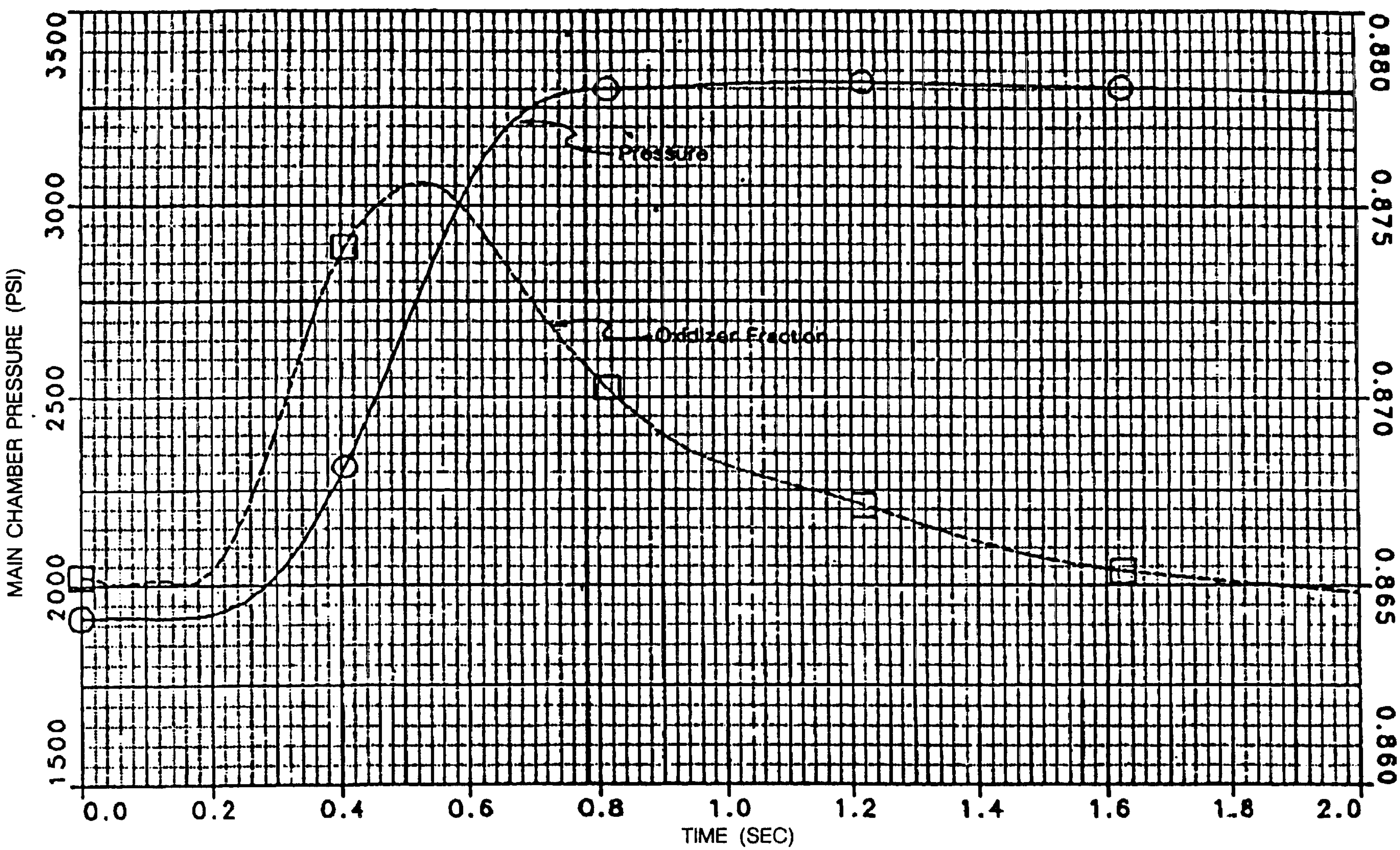

Figure 5-6 - Main Chamber Conditions as a Fucntion of Time

5.2 DETAILED LOX SIDE MODEL TEST

The Oxidizer side of the detailed Technology Test Bed Engine (TTBE) model was configured to verify component fidelity and basic model definition. Figure 5-7 shows a schematic of the model with the station names labeled. This version of the model did not include a pogo system, which was added later.

In order to verify the configured model, a shutdown transient was executed by giving the detailed lox side model transient inputs from the Rocketdyne Digital Transient Model (DTM) of reference 2. These inputs were the high-pressure rotor speed, the low-pressure pump inlet pressure, the pogo flowrate, and the oxidizer flowrates to the preburners and main chamber. Figures 5-8 and 5-9 show these inputs as a function of time. Some results from operating the lox-side TTBE simulation are shown in Figures 5-10 to 5-13 with overlays to the DTM predictions. A 10ms time step, which is approximately 50 times larger than the DTM, was used with the implicit integration scheme of ROCETS.

With 5 passes, or less, at each time step to obtain the implicit integration, an order of magnitude savings exist in computer calculations for a simulated transient. Figure 5-10 & 5-11 presents the low pressure pump and line 2 flowrates for both simulations showing excellent agreement. The comparisons between the two simulations of low pressure rotor speeds (Figure 5-12) and mixer 2 total pressure (figure 5-13) are also very good transiently with only some initial steady-state level differences.

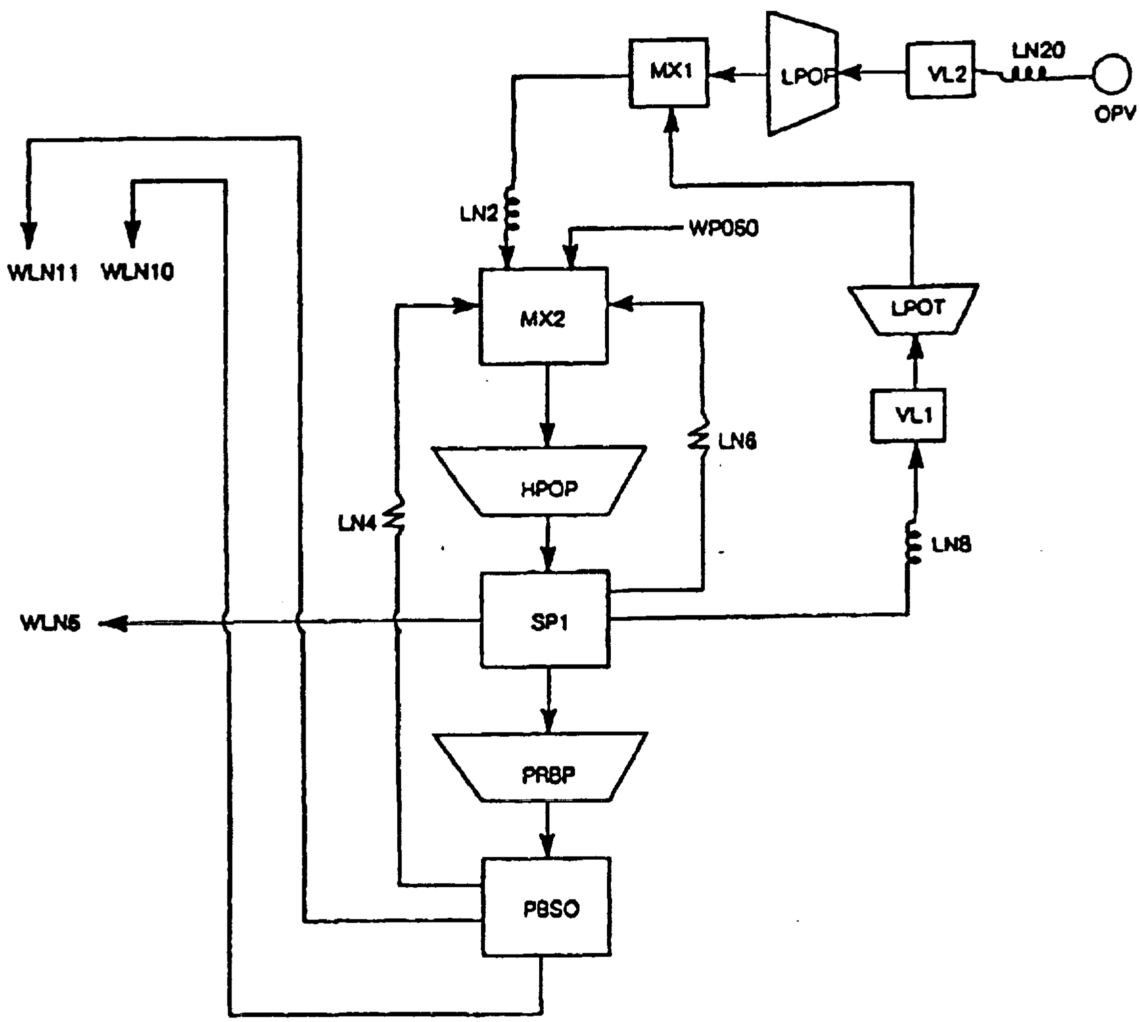

Figure 5-7. ROCETS TTBE Model Loxside Schematic

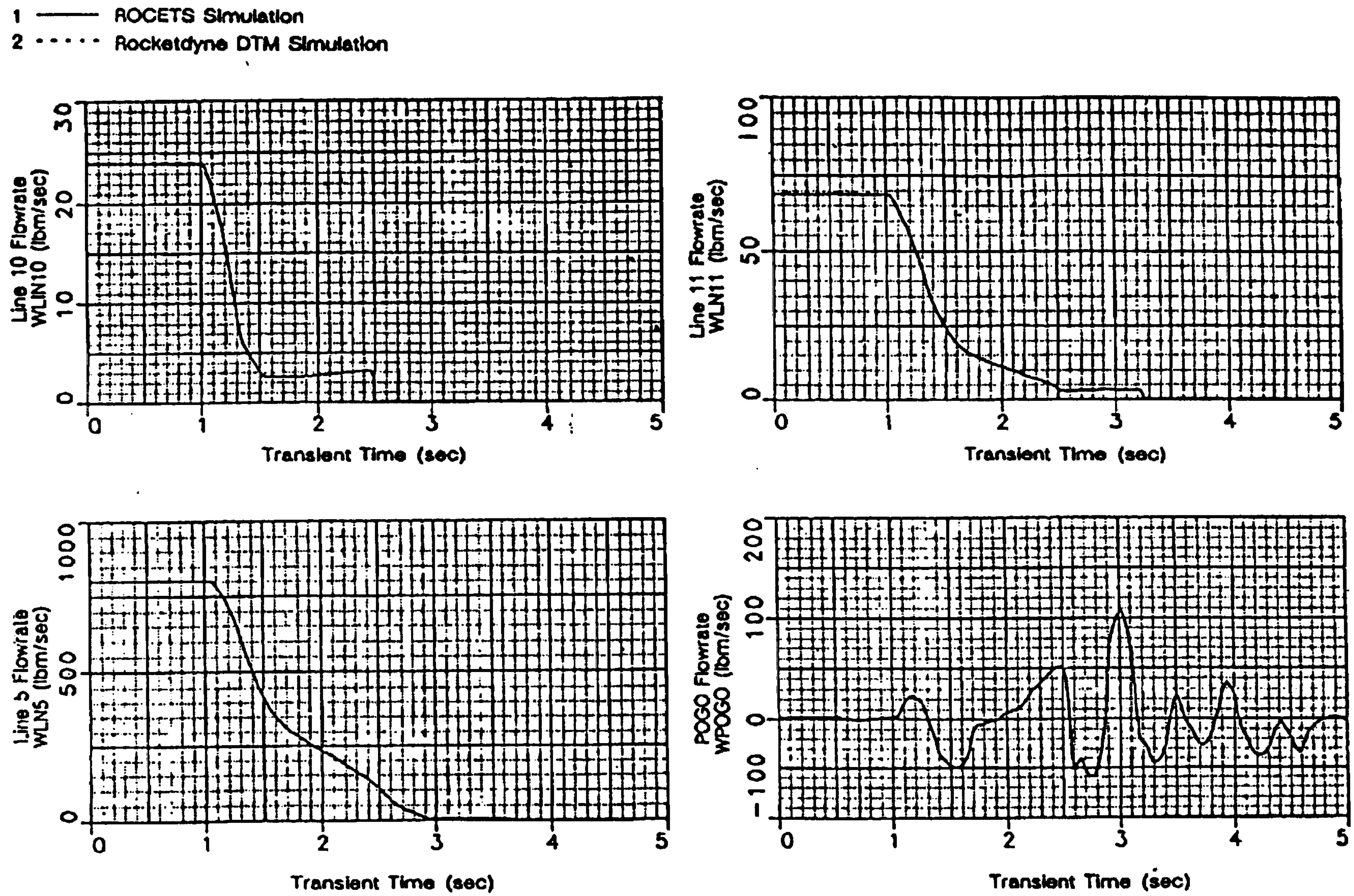

Figure 5-8. ROCETS TTBE Loxside Shutdown Transient - Transient Inputs

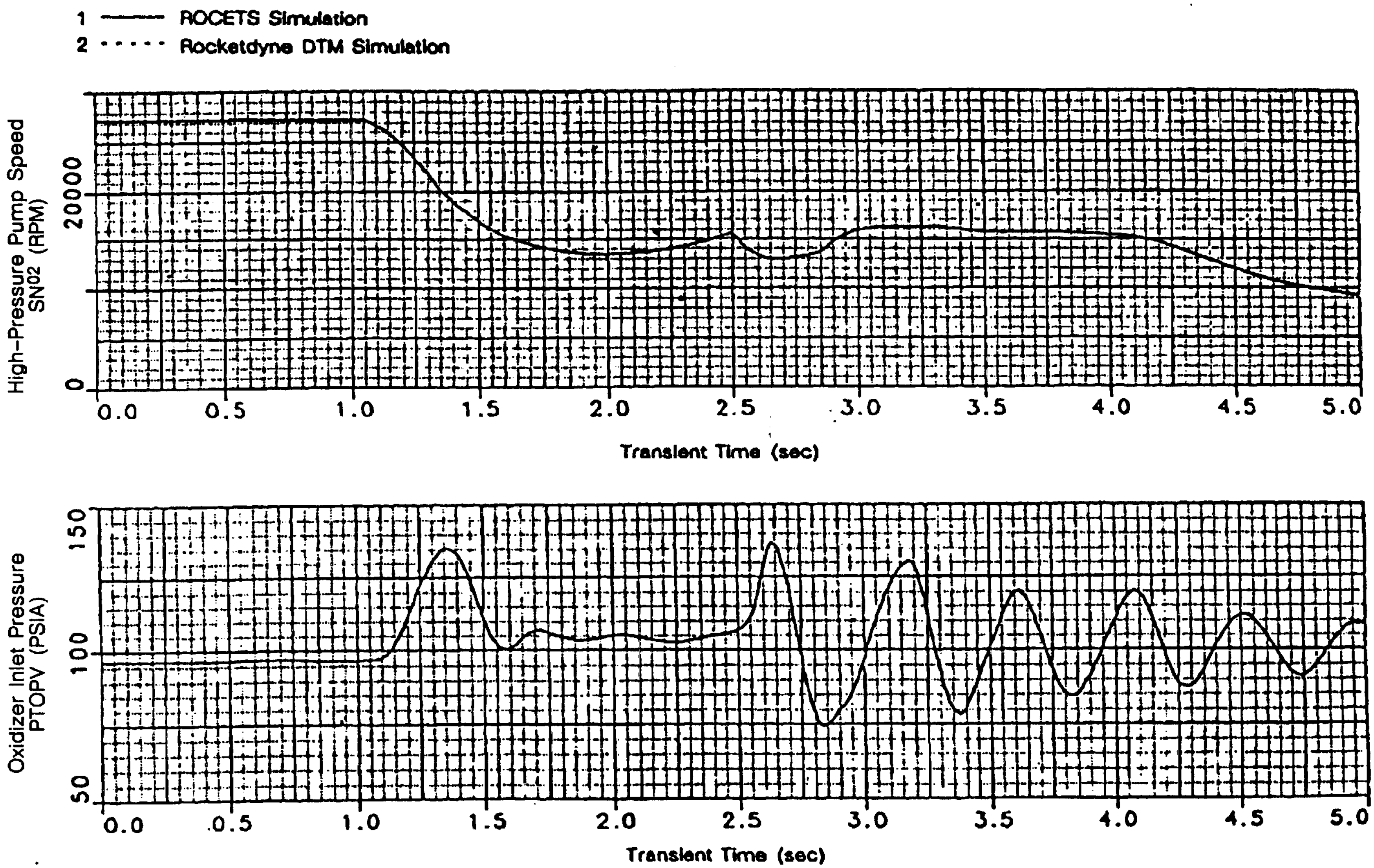

Figure 5-9. ROCETS TTBE Loxside Shutdown Transient - Transient Inputs

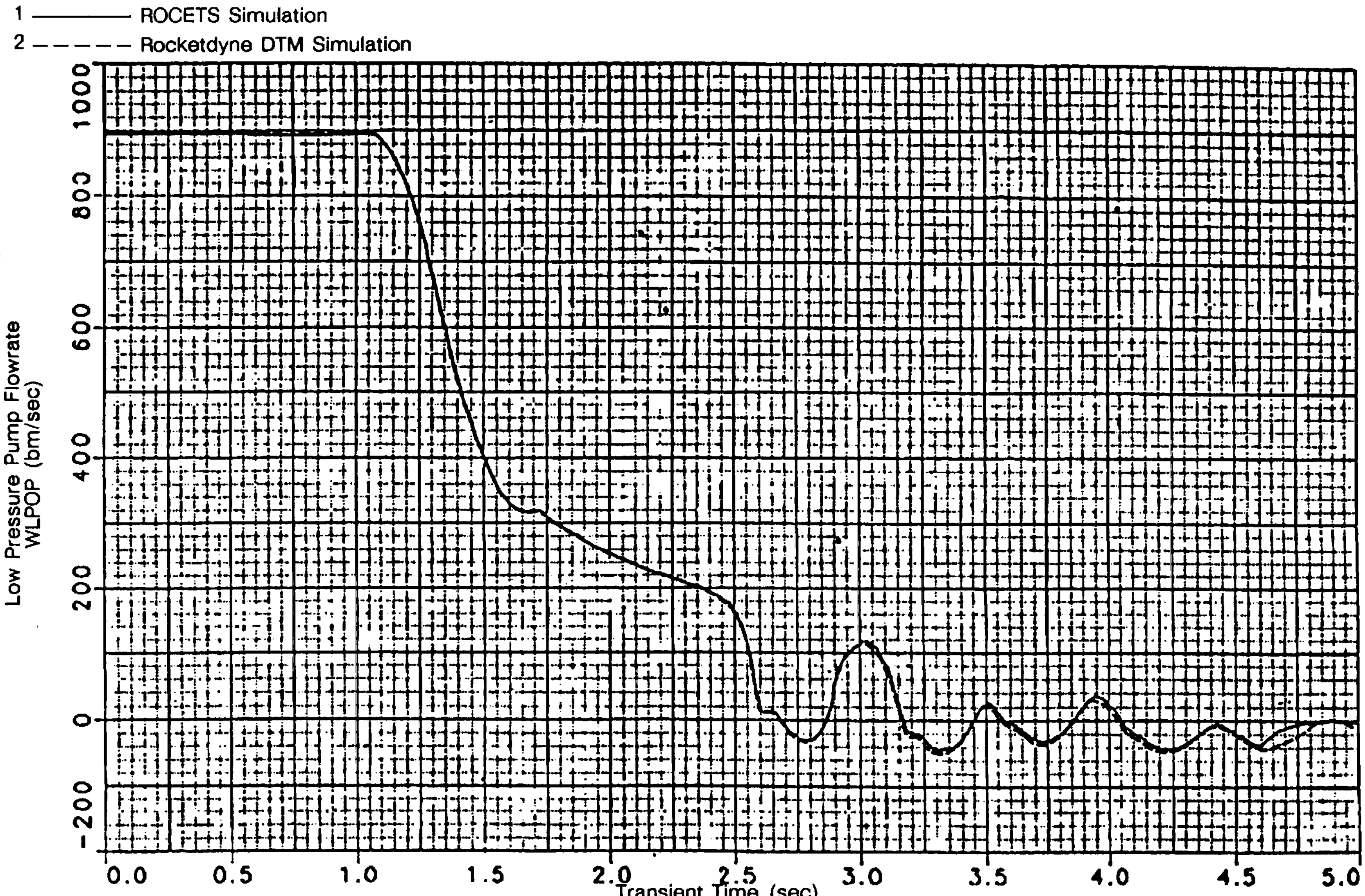

Figure 5-10. ROCKETS TTBE Loxside Shutdown Transient - Low Pressure Pump Flowrate

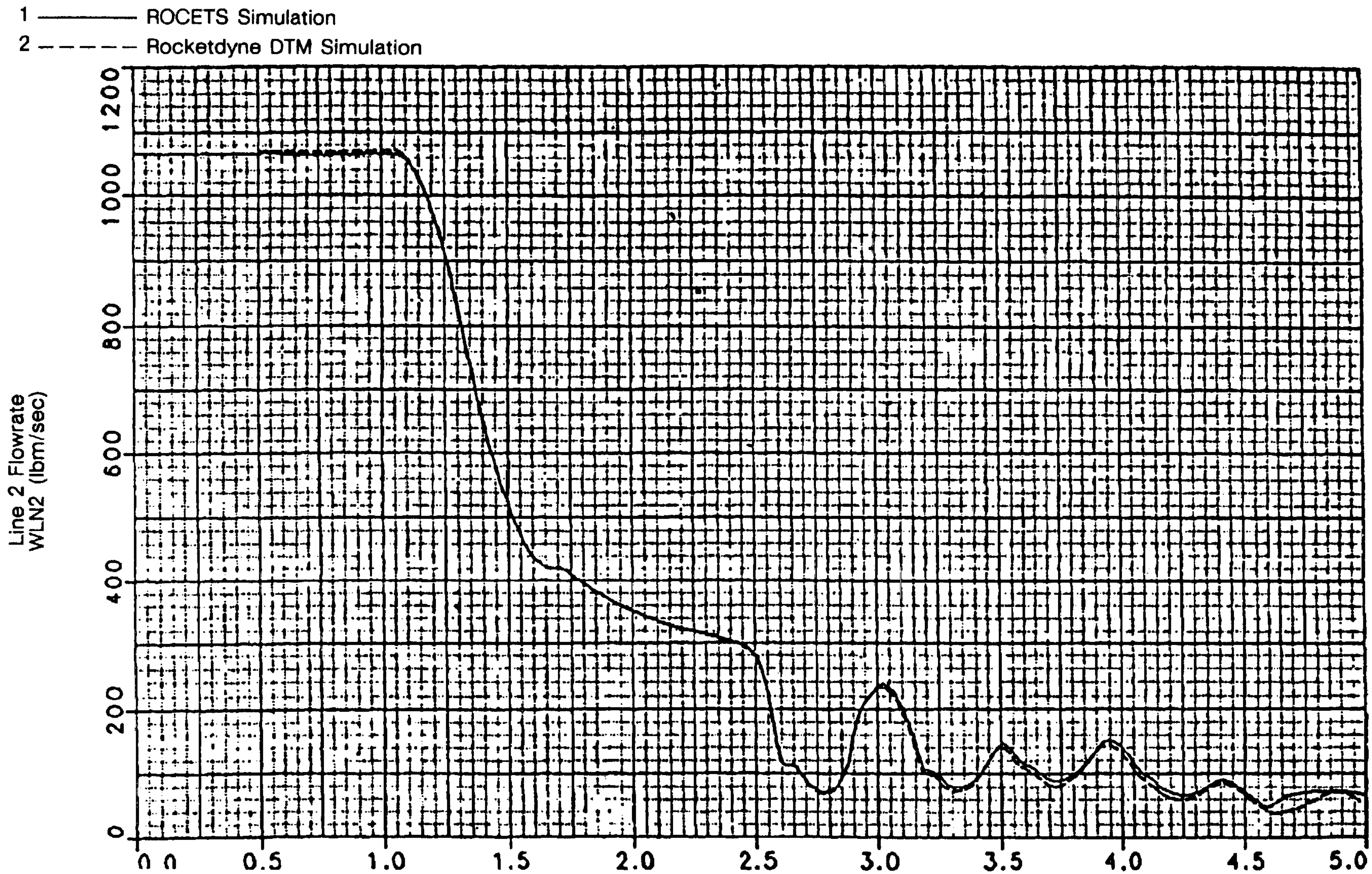

Figure 5-11. ROCETS TTBE Loxside Shutdown Transient - Line 2 Flowrate

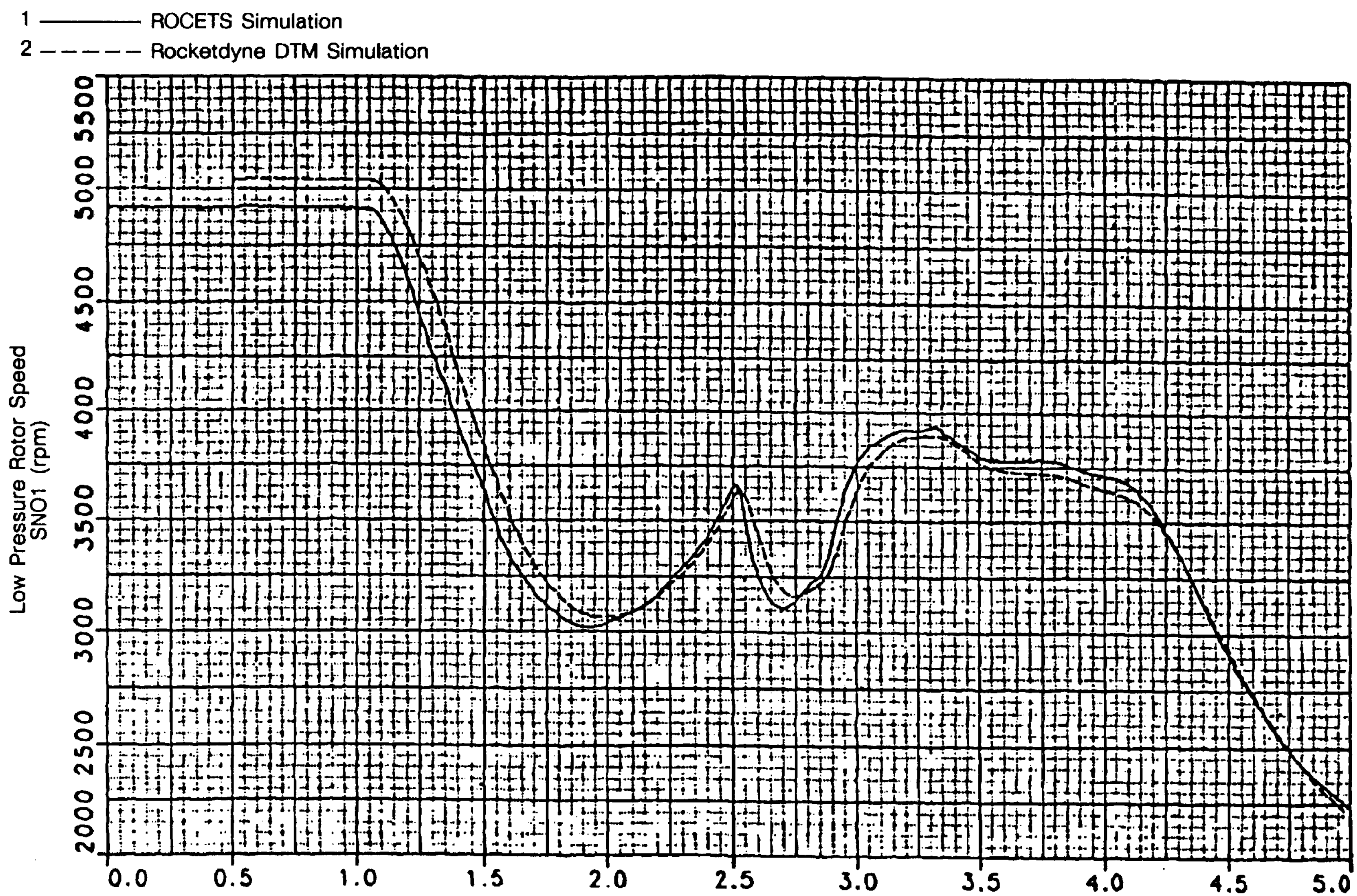

Figure 5-12. ROCETS TTBE Loxside Shutdown Transient - Low Pressure Rotor Speed

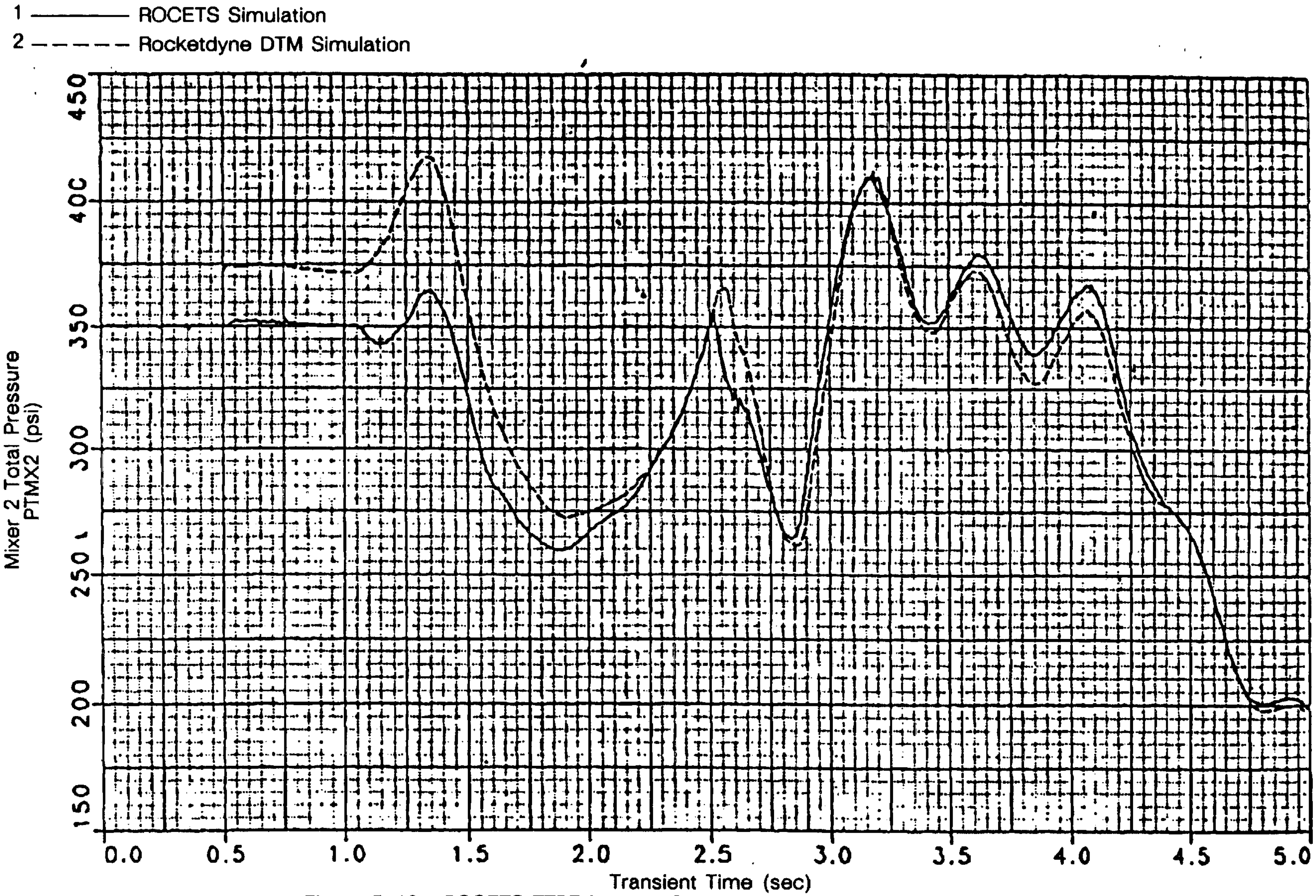

Figure 5-13. ROCETS TTBE Loxside Shutdown Transient – Mixer 2 Total Pressure

5.3 DETAILED TTBE MODEL TEST

The complete detailed TTBE model was balanced at 100% RPL and compared to the DTM. As shown on Table 5-1, the balance condition was close to the values of the DTM parameters. Various open loop transients in main stage operation were exercised to verify proper operation of the model operating in the ROCETS system. Figures 5-14 through 5-21 present the results of one of these experiments. The engine was initiated at 100% RPL, and then the fuel preburner valve was closed 10% (Figure 5-14). The transient response to selected parameters of flows, pressure, speeds, and pogo flowrate and pressure are presented on Figure 5-15 through 5-21.

Table 5 -1. Comparison Between DTM & TTBE

	DTM	TTBE	% DIFF
LPFT Speed	15605.41	15198.48	2.595
HPFT Speed	34189.85	34173.44	0.048
LPFP Flowrate	148.732	149.494	0.512
HPFP Flowrate	145.155	149.494	2.989
HPFP Pr Rise	5805.45	5926.521	2.051
LPFT Flowrate	28.063	28.773	2.530
PBSF Temp.	278.182	293.347	5.451
LPOT Speed	5041.484	5059.559	0.359
HPOT Speed	27240.76	27289.37	0.178
LPOP Flowrate	896.205	905.328	1.018
HPOP Flowrate	1078.373	1087.344	0.832
HPOP Pr Rise	3733.740	3659.952	1.976
PRBP Pr Rise	3025.265	3129.196	3.435
LPOT Flowrate	176.084	176.964	0.450
MFV Flow	145.155	145.257	0.070
CCV Flow	61.258	65.613	7.109
MOV Flow	799.967	812.811	1.605
FPOV Flow	68.530	64.868	5.344
OPOV Flow	23.976	24.022	0.192
Flow to V16	52.911	49.593	6.271
Flow to CCV	61.261	65.613	7.104
Flow to LPFT	28.063	28.772	2.526
Fuel Ign Flow	0.947	0.959	1.267
Flow to MOV	799.967	812.790	1.603
Flow to LPOT	176.084	176.964	0.500
Flow to POGO	0.392	0.447	14.031
Flow to PRBP	98.728	95.168	3.606
Flow (In 04)	1.966	1.975	0.458
Flow (In 06)	6.181	6.278	1.569
Flow to FPOV	68.530	64.868	5.344
Flow to OPOV	23.976	24.022	0.192
Flow to FPRB	78.955	79.328	0.472
Flow to OPRB	36.355	35.334	2.808
FPRB Temp	1786.976	1747.538	2.207
FPRB Press	4940.559	4995.051	1.103
OPRB Temp	1437.797	1519.739	5.700
OPRB Press	4995.957	5082.129	1.725
HPFT Inlet T	1786.976	1747.548	2.206
HPFT PR	1.449	1.471	1.518
HPFT Flowrate	147.524	144.195	2.257
HPOT Inlet T	1437.797	1519.746	5.700
HPOT PR	1.508	1.5280	1.326
HPOT Flowrate	59.036	58.078	1.625
MFI Temp	1476.750	1475.104	0.111
MFI Press	3218.715	3205.637	0.406
MOI Temp	191.586	191.315	0.141
MOI Press	3894.036	3923.581	0.759
MCHB Press	3004.181	3004.785	0.020
MCHB Temp	6487.148	6540.789	0.827
Thrust	374218.1	375071.7	0.228
Spec Impusle	359.61	356.815	0.777
Ch. Cool Tdis	285.424	291.4841	2.123
Ch. Cool Pdis	5560.598	5656.090	1.717

ORIGINAL PAGE IS
OF POOR QUALITY

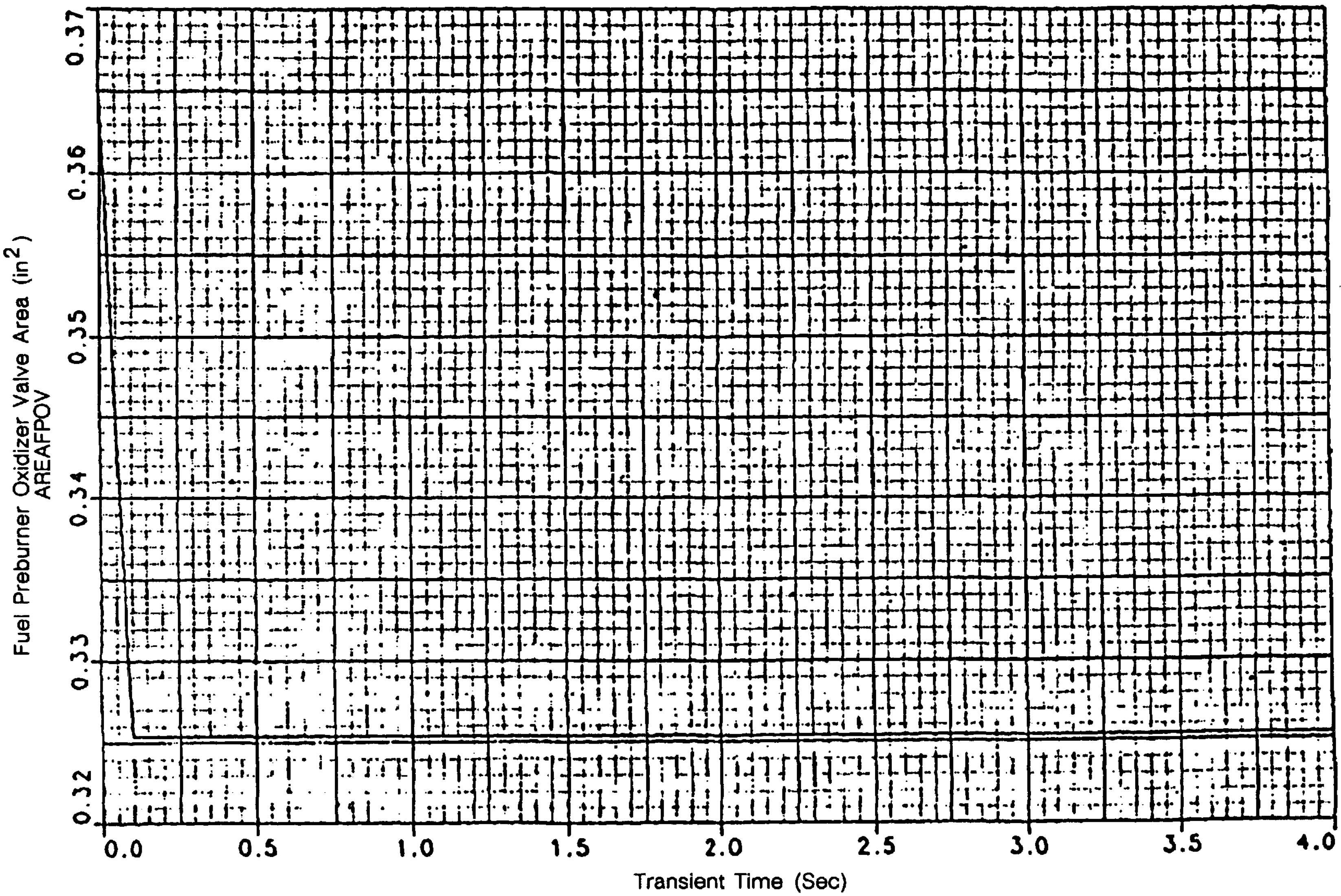

Figure 5-14 – Imposed Valve Transent at 100% RPL on TTBE Model

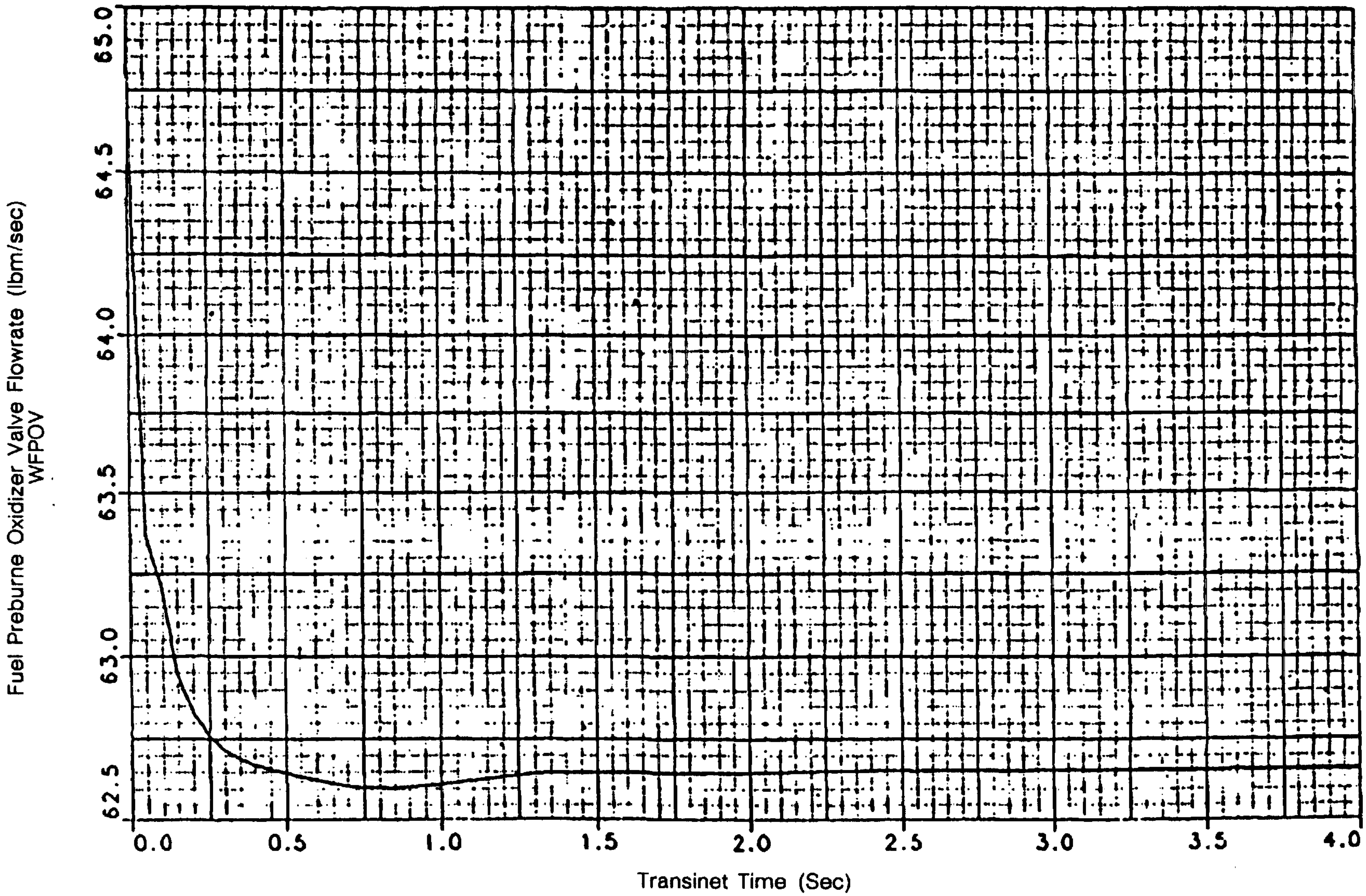

Figure 5-15 - TTBE Model Response - Fuel P/B Oxidizer Flowrate

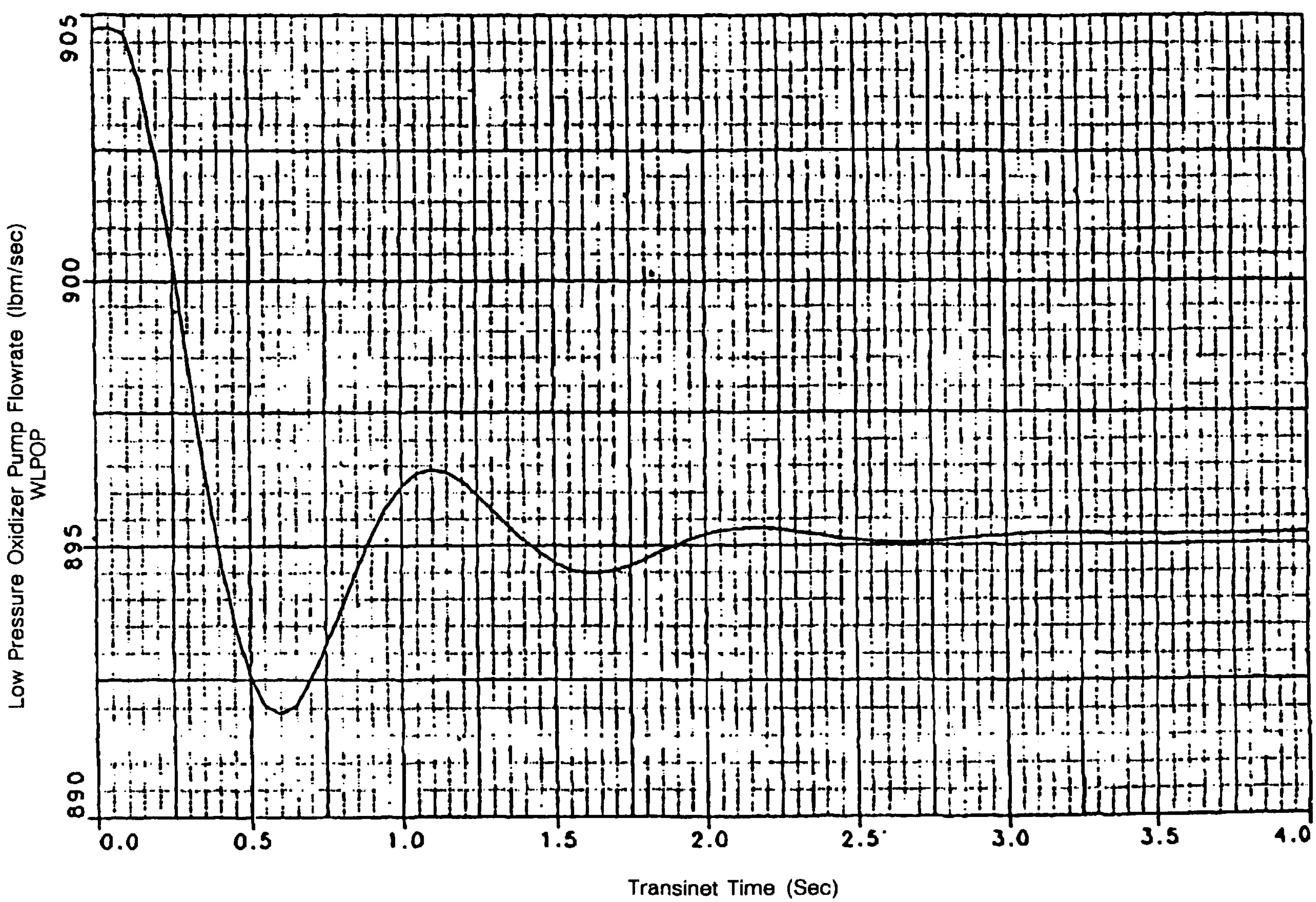

Figure 5-16 – TTBE Model Response – Low Pressure Oxidizer Pump Flowrate

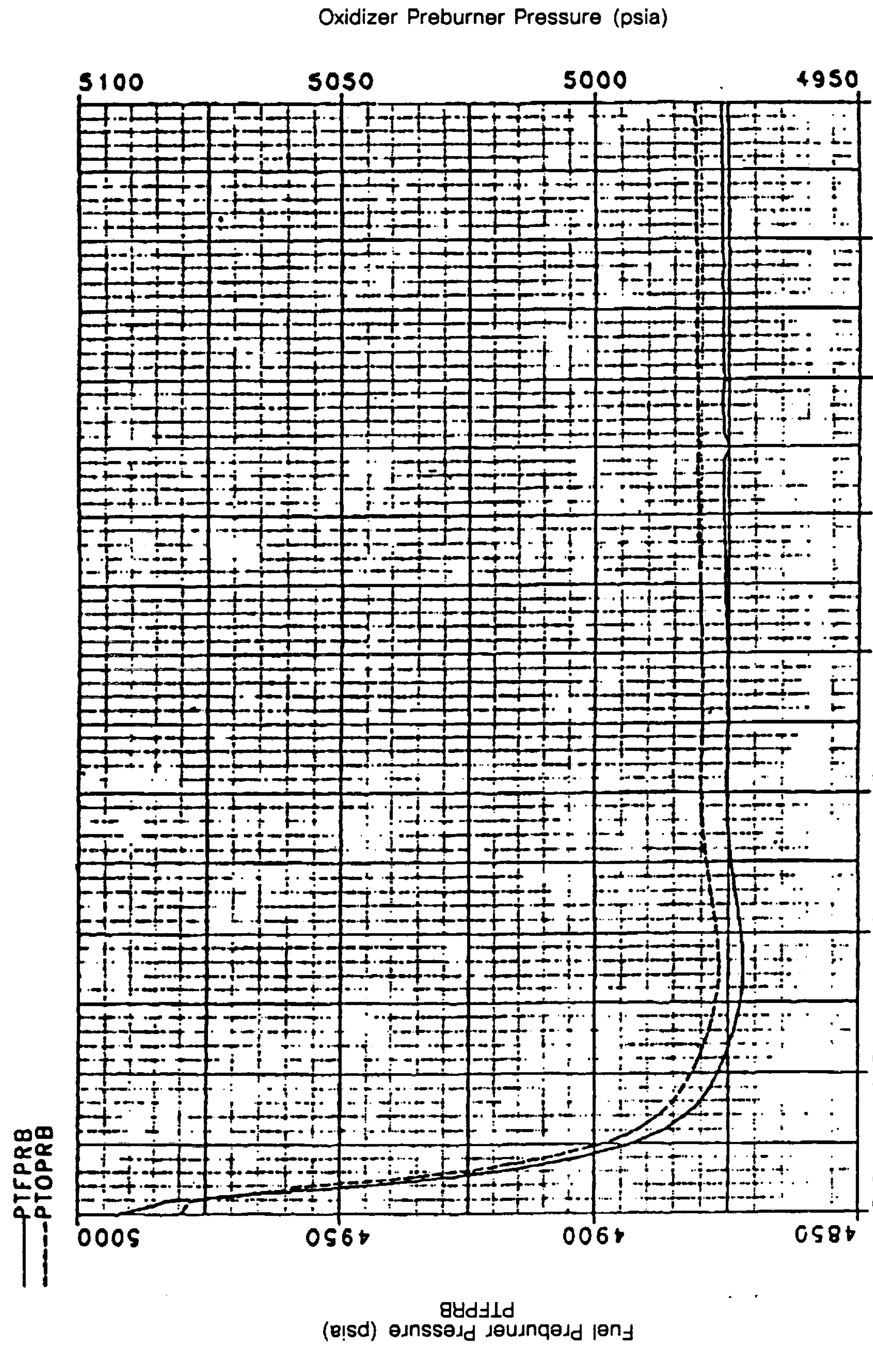

Figure 5-17 – TTBE Model Response – Fuel and Oxidizer Preburner Pressures

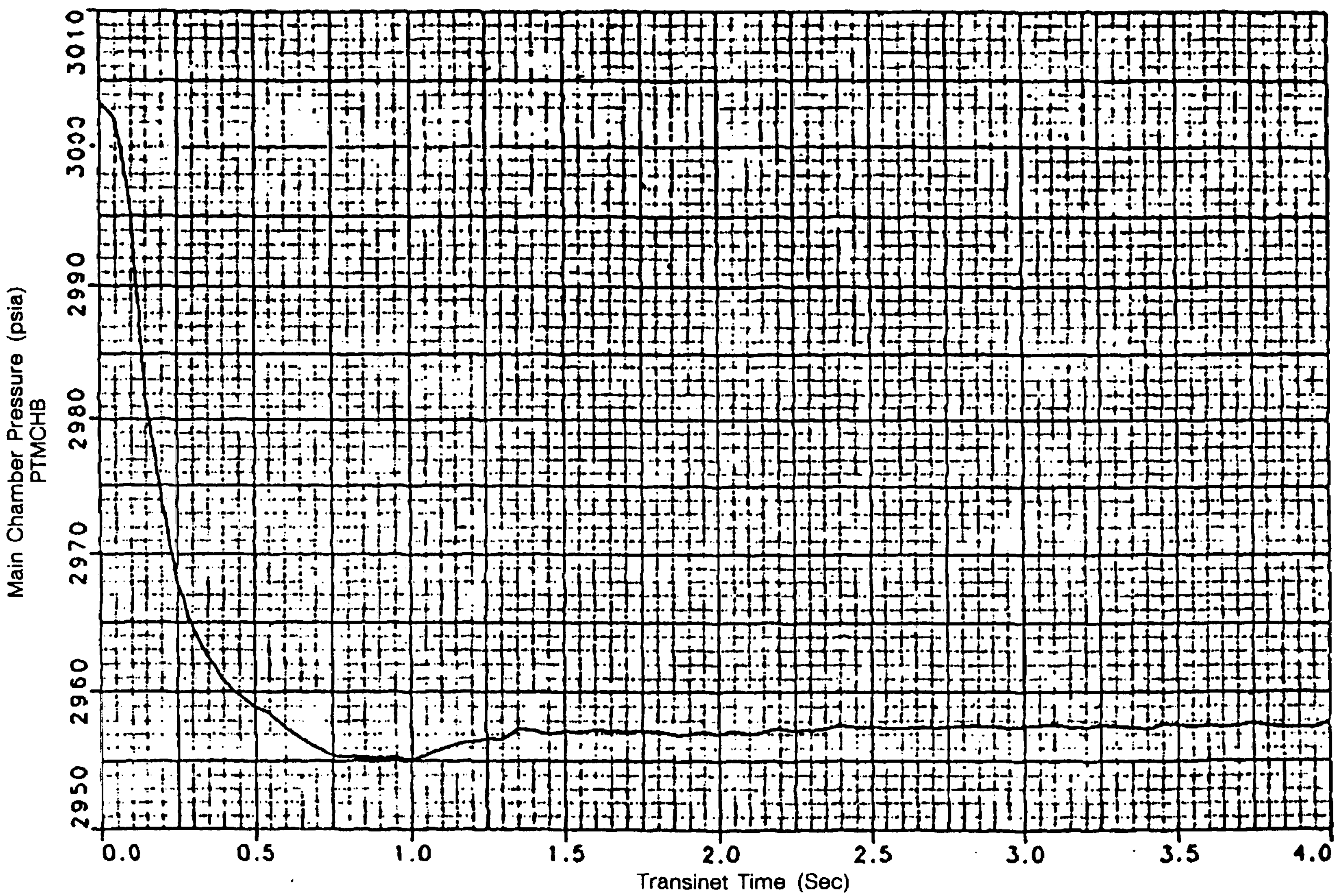

Figure 5-18 – TTBE Model Response – Main Chamber Pressure

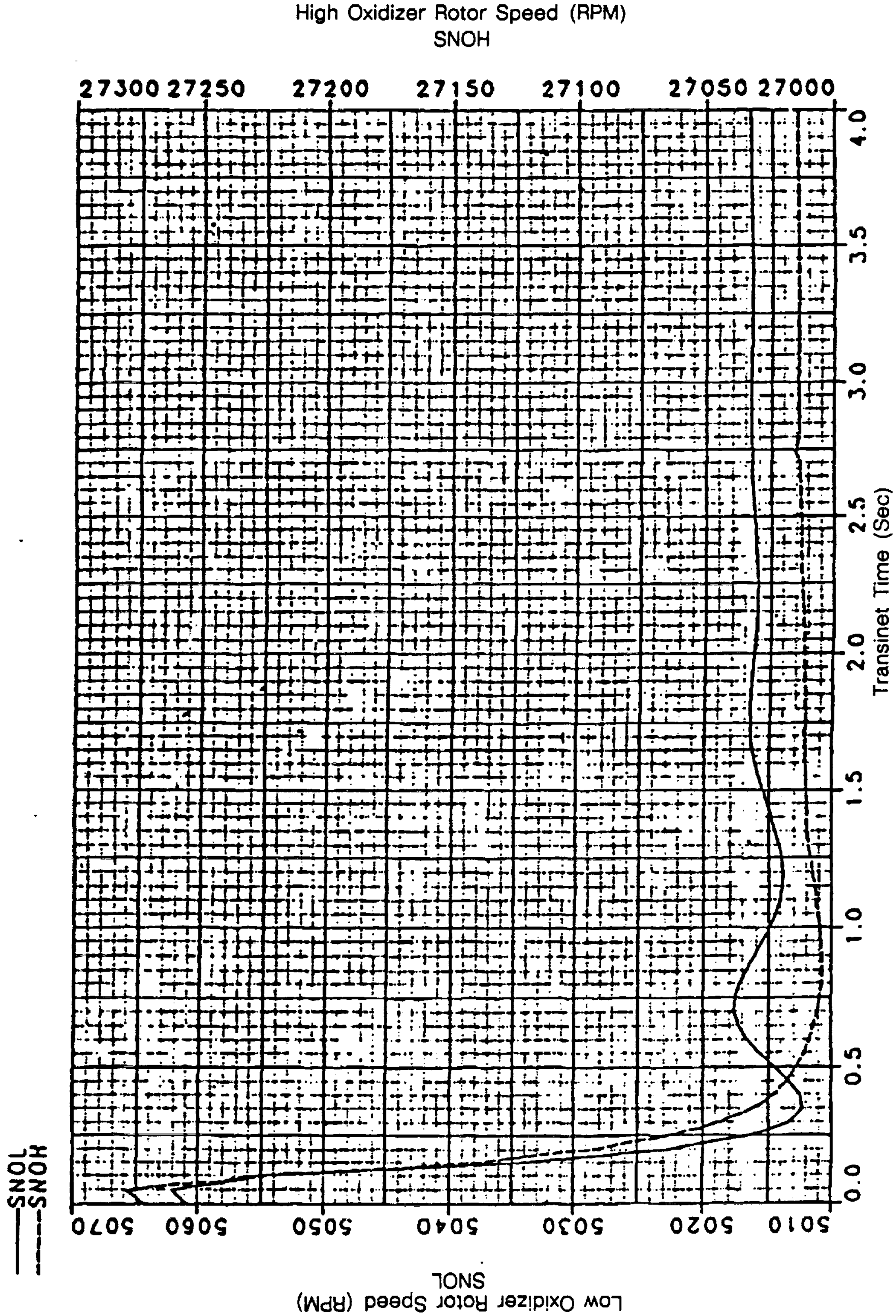

Figure 5-19 – TTBE Model Response – Speeds of Oxidizer Pumps

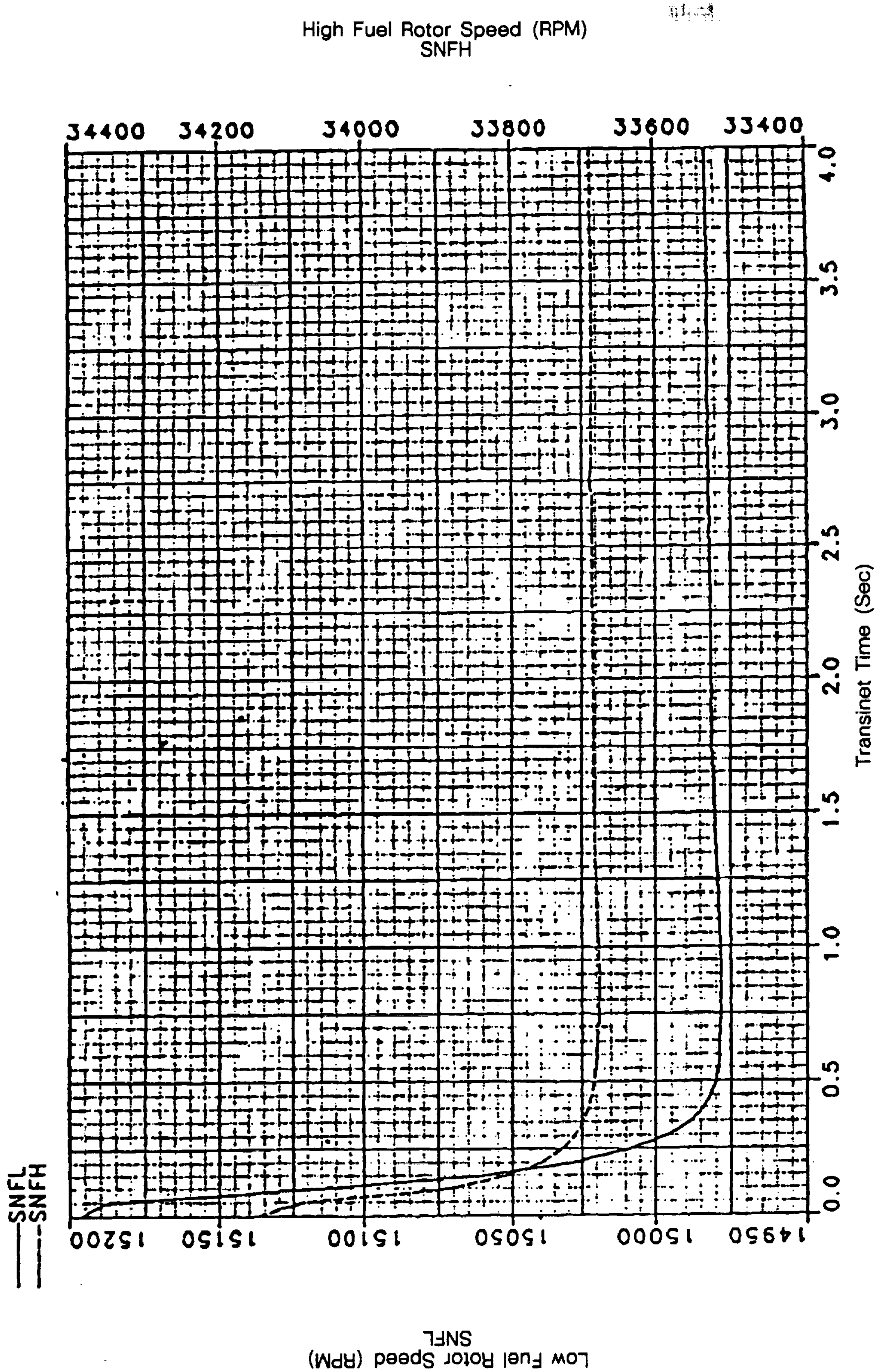

Figure 5-20 - TTBE Model Response - Speeds of Fuel Pumps

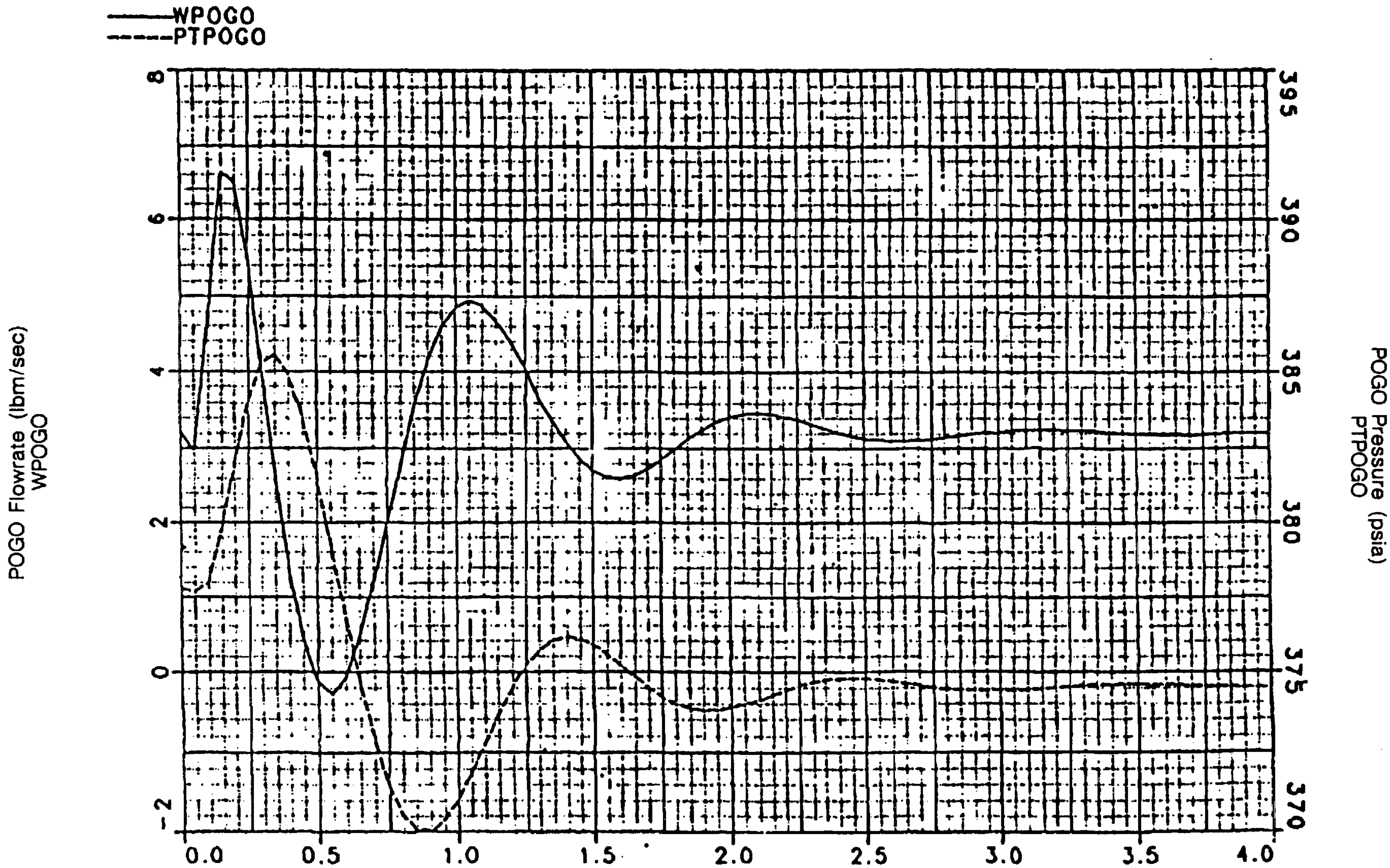

Figure 5-21 - TTBE Model Response - POGO Flowrate and Pressure

5.3.1 Shutdown Transient

The detailed TTBE model exercised a shutdown transient from 100% RPL. The open loop valve schedules were taken from a DTM shutdown and imposed on the TTBE model as presented on Figures 5-22 and 5-23. Comparison plots of TTBE and DTM predictions of selected model parameters are presented on Figures 5-24 thru 5-31. Parameters presented are main chamber and preburner pressures and temperatures along with the rotor speeds of the four turbopumps. In general the TTBE decelerated faster than the DTM, but no attempt was made to tune the TTBE model. The significant verification from the test was the model could operate successfully, including implicit integration, through this transient of such drastic operating changes.

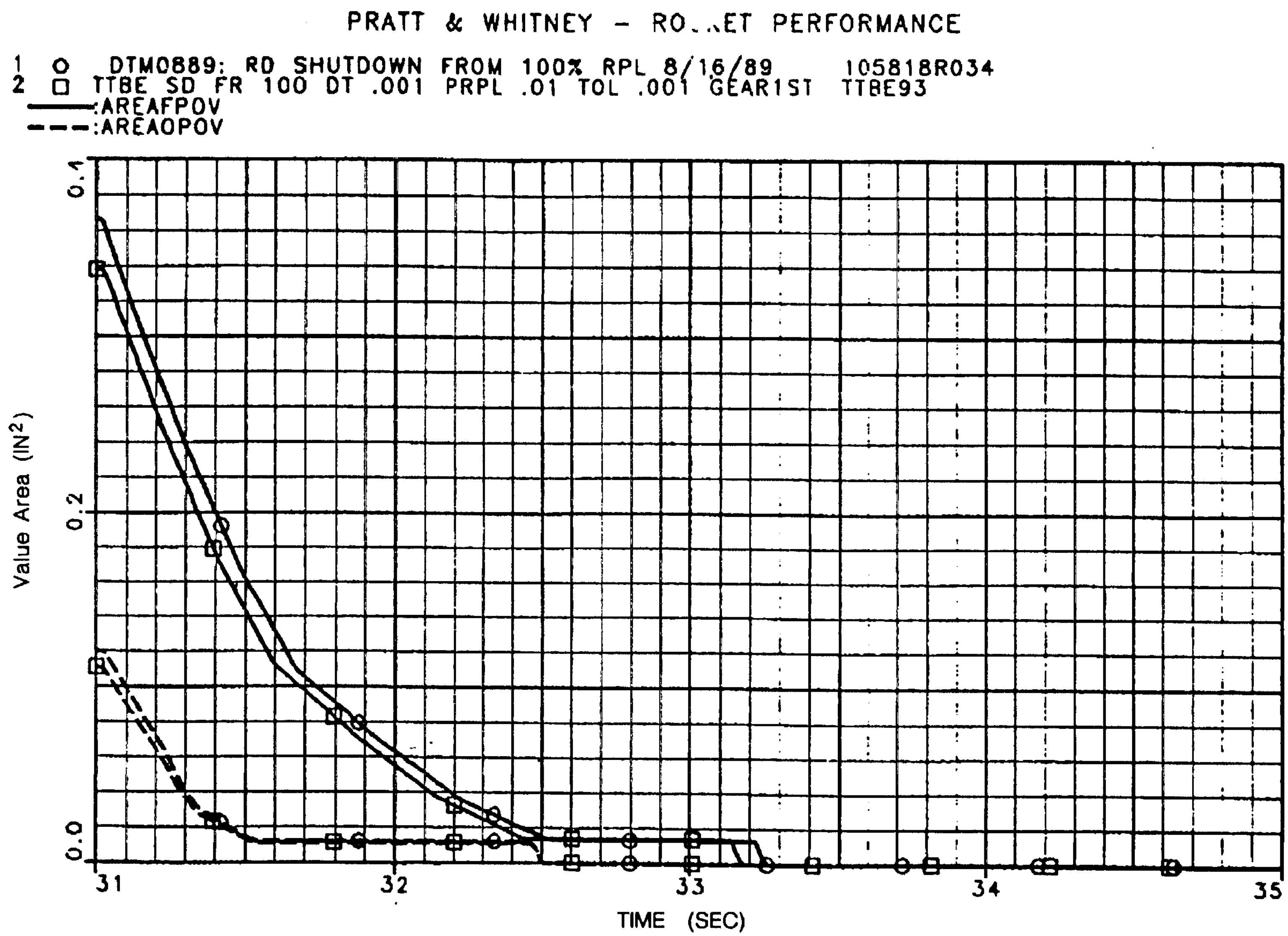

Figure 5-22 - Shutdown Open-Loop Valve Schedules

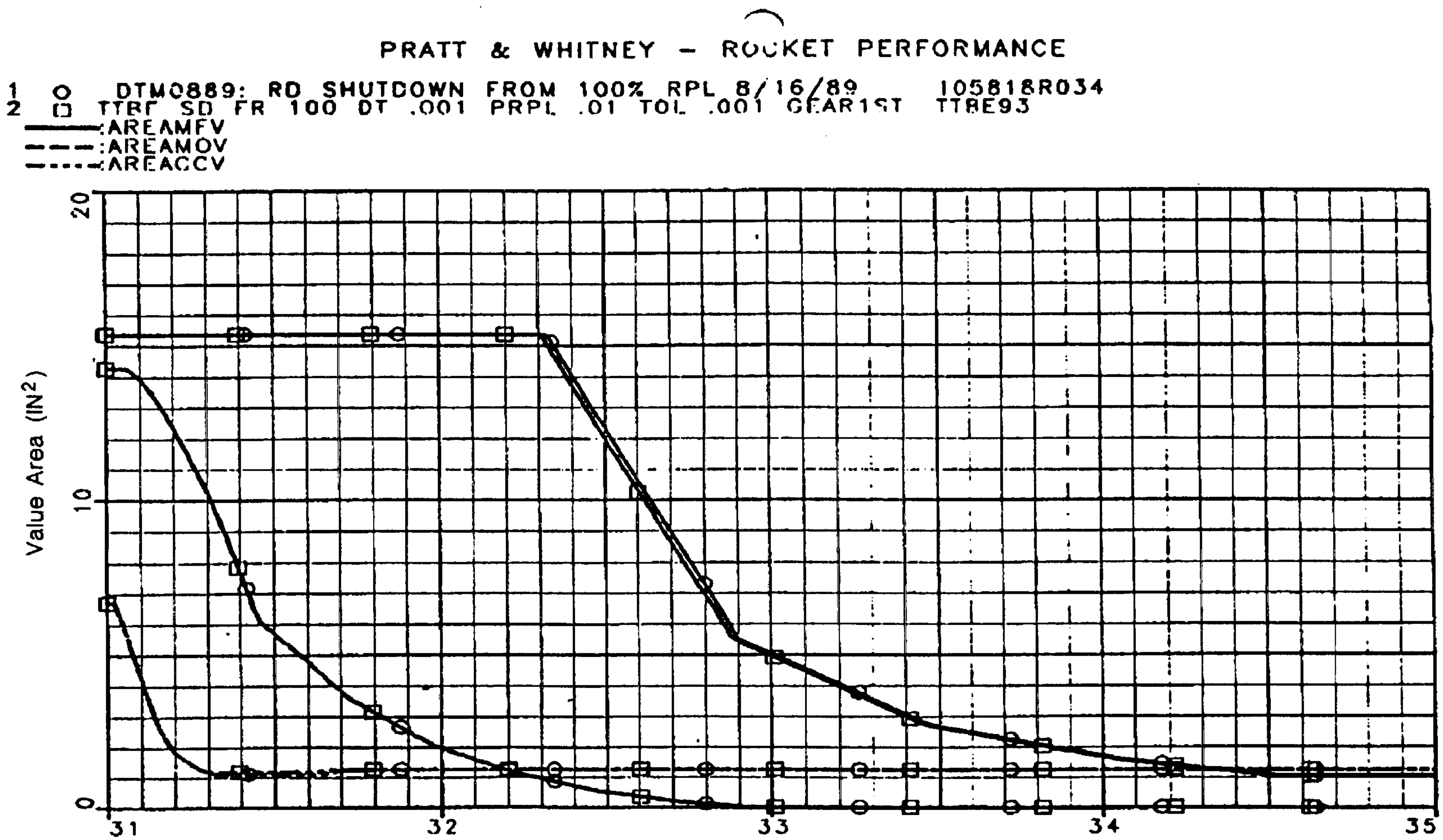

Figure 5-23 - Shutdown Open-Loop Valve Schedules

77 INTENTIONALLY

PRECEDING PAGE BLANK NOT FILMED

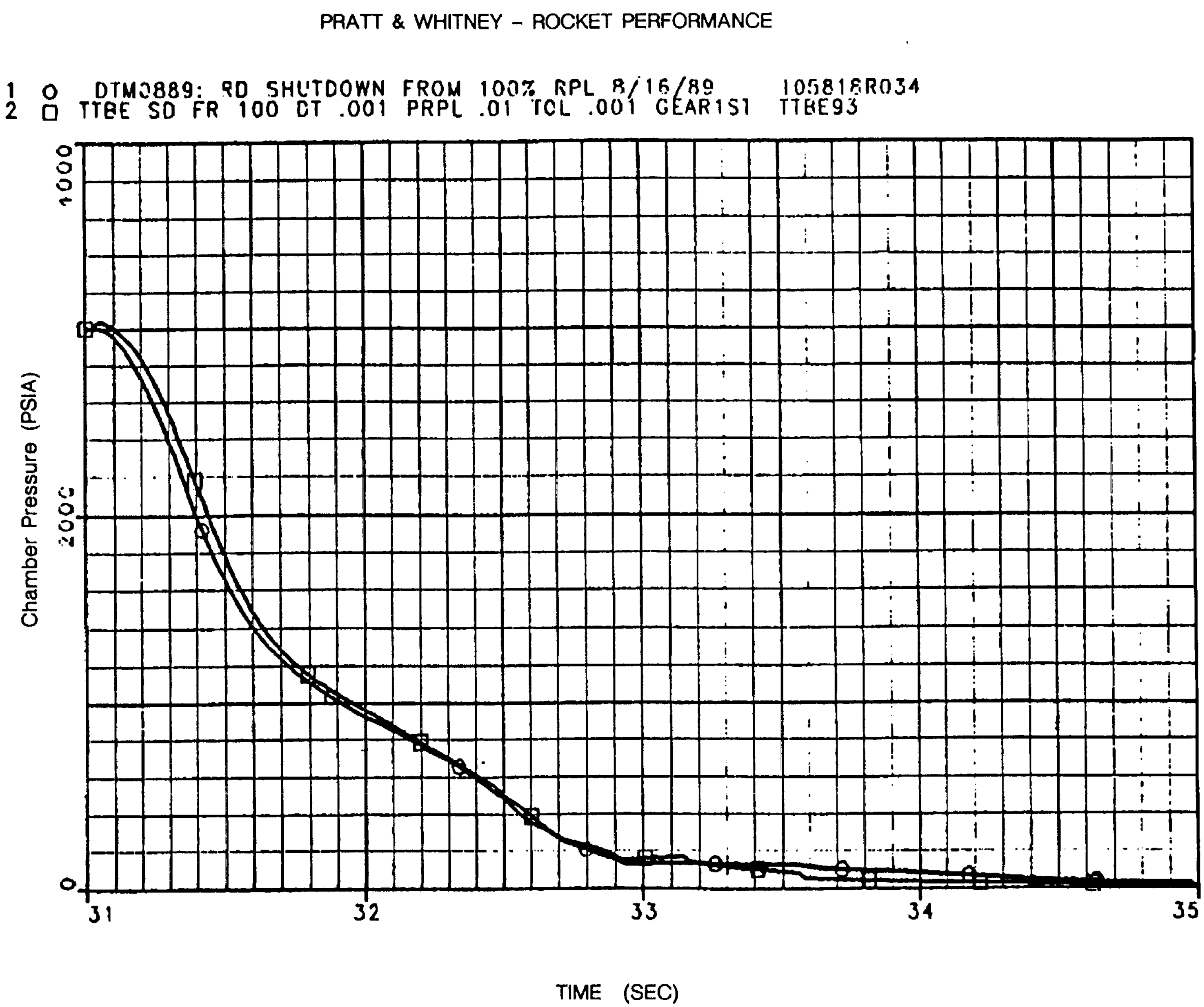

Figure 5-24 – Shutdown Chamber Pressure

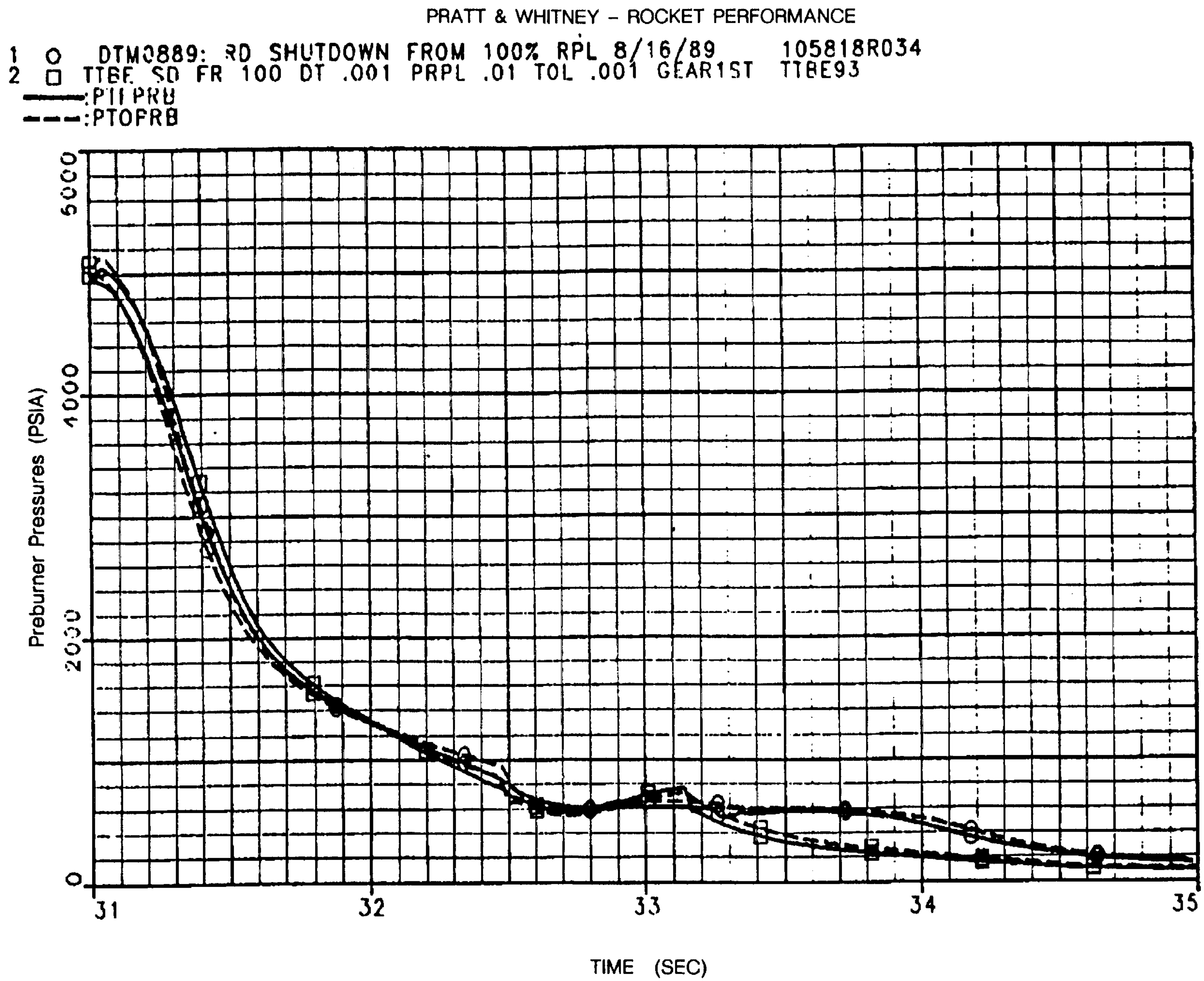

Figure 5-25 - Shutdown Pressures of Both Preburners

PRATT & WHITNEY - ROCKET PERFORMANCE

1 O DTM0889: RD SHUTDOWN FROM 100% RPL 8/16/89 105818R034
2 □ TTBE SD FR 100 DT .001 PRPL .01 TOL .001 GEAR1ST 11BE93

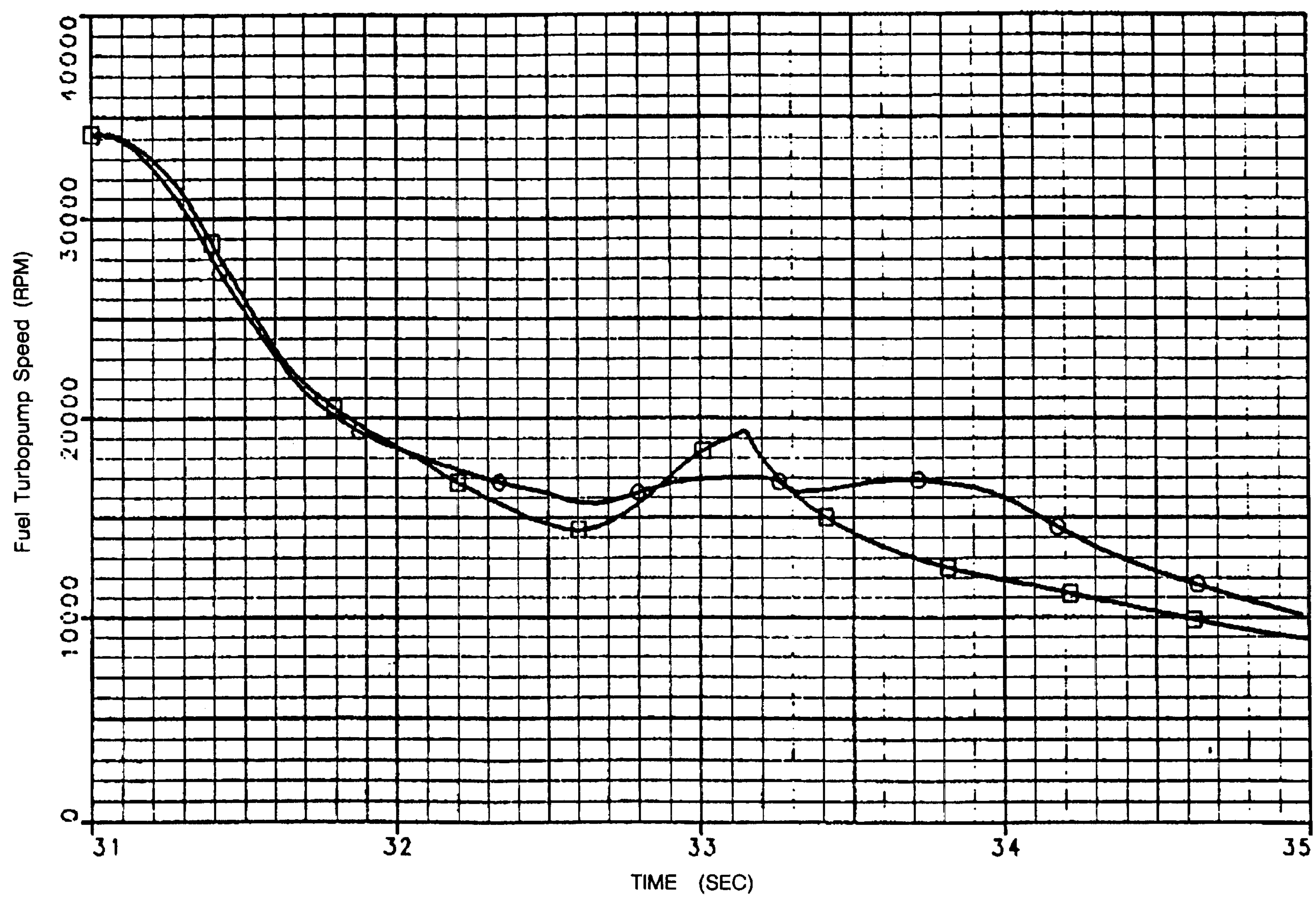

Figure 5-26 - Shutdown Fuel Turbopump Speeds

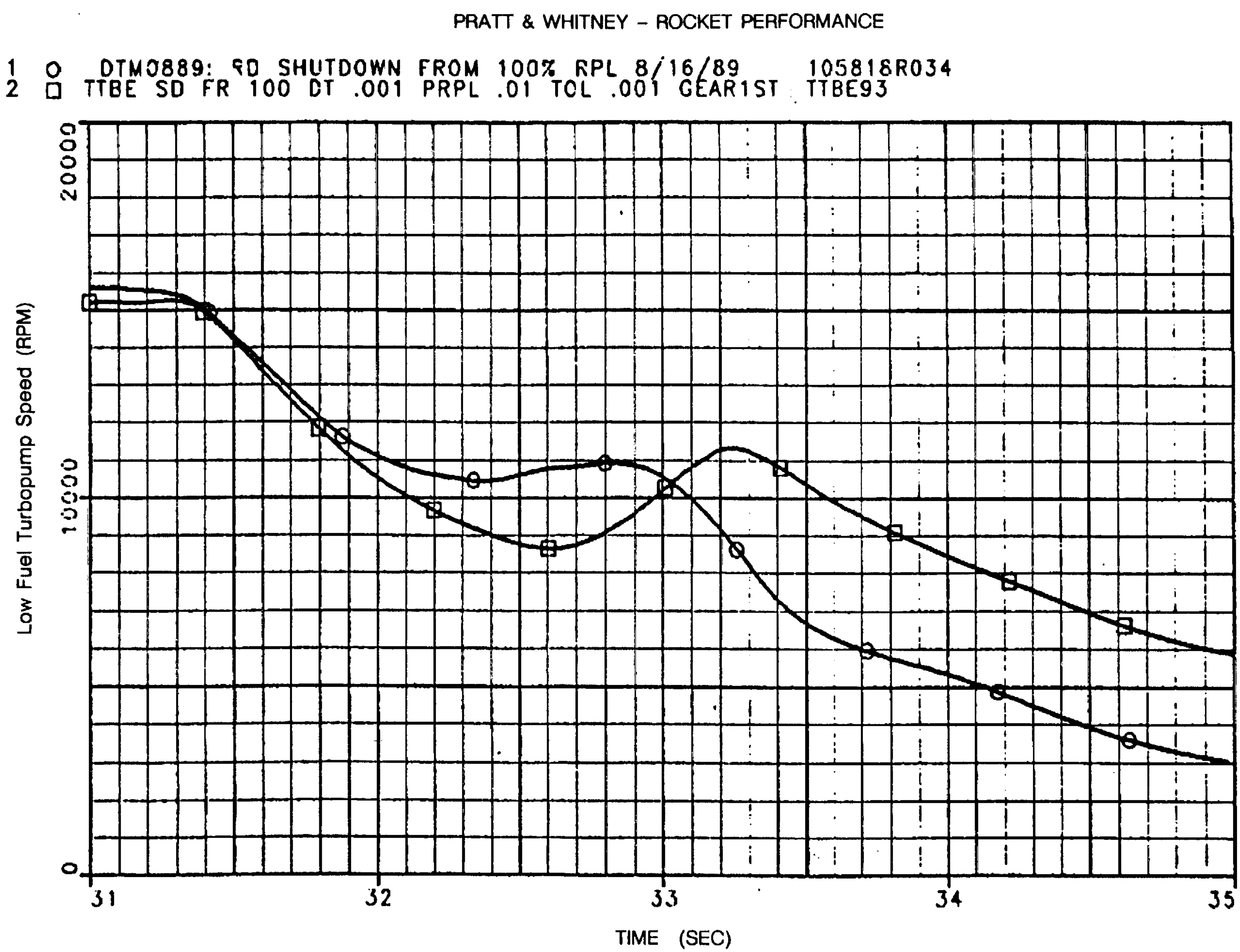

Figure 5-27 – Shutdown Low Fuel Turbopump Speed

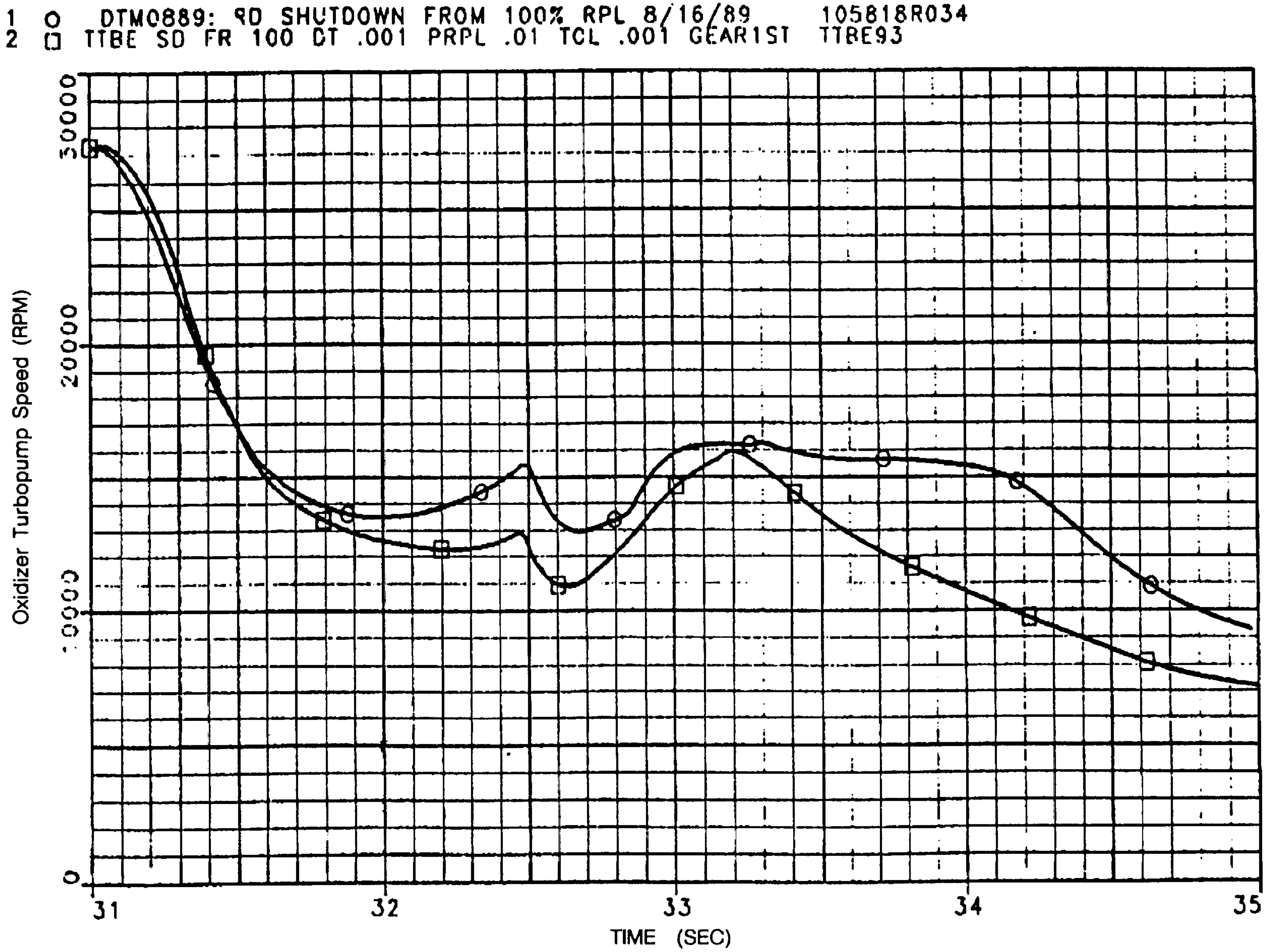

Figure 5-28 - Shutdown Oxidizer Turbopump Speed

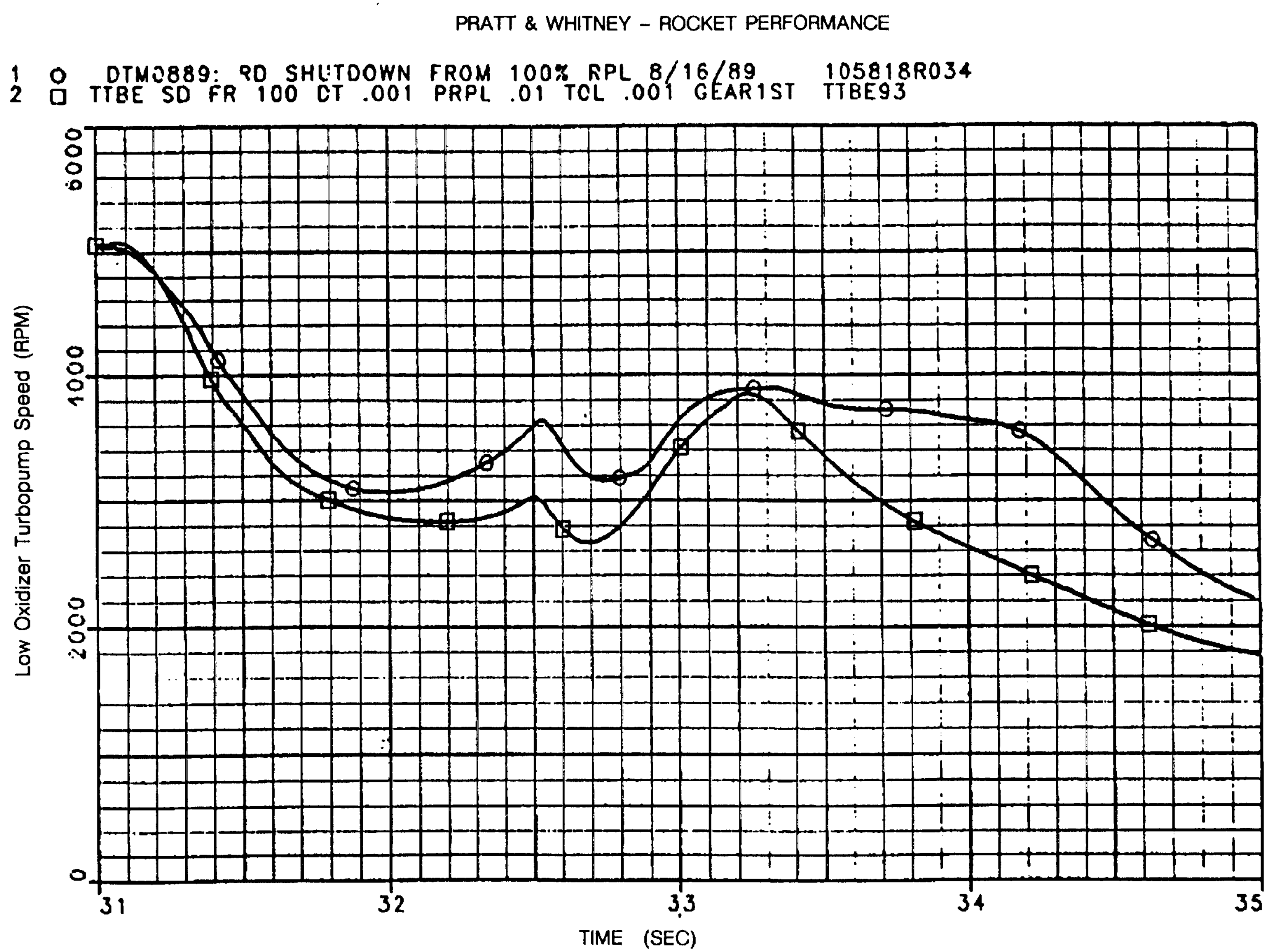

Figure 5-29 - Shutdown Low Oxidizer Turbopump Speed

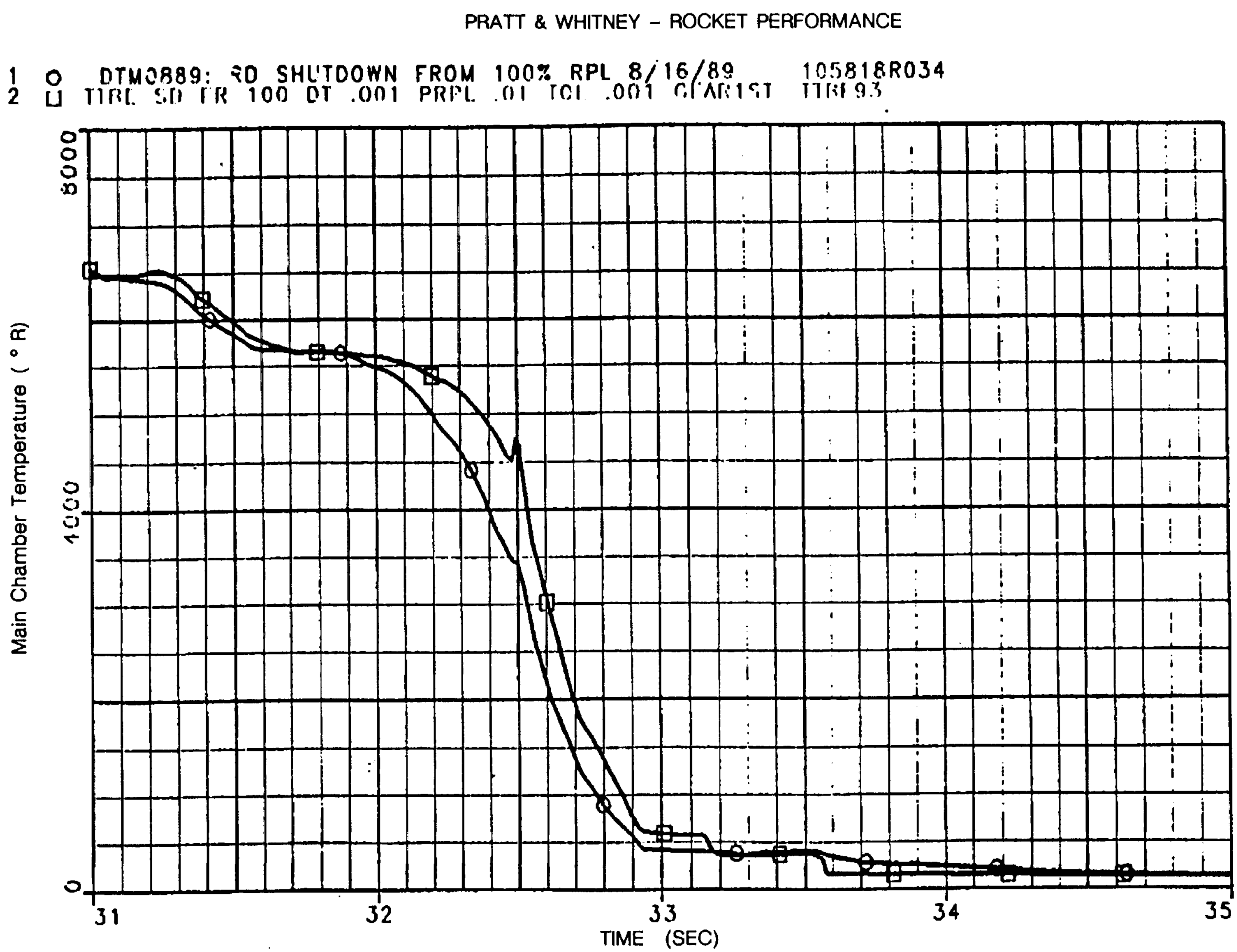

Figure 5-30 - Shutdown Main Chamber Temperature

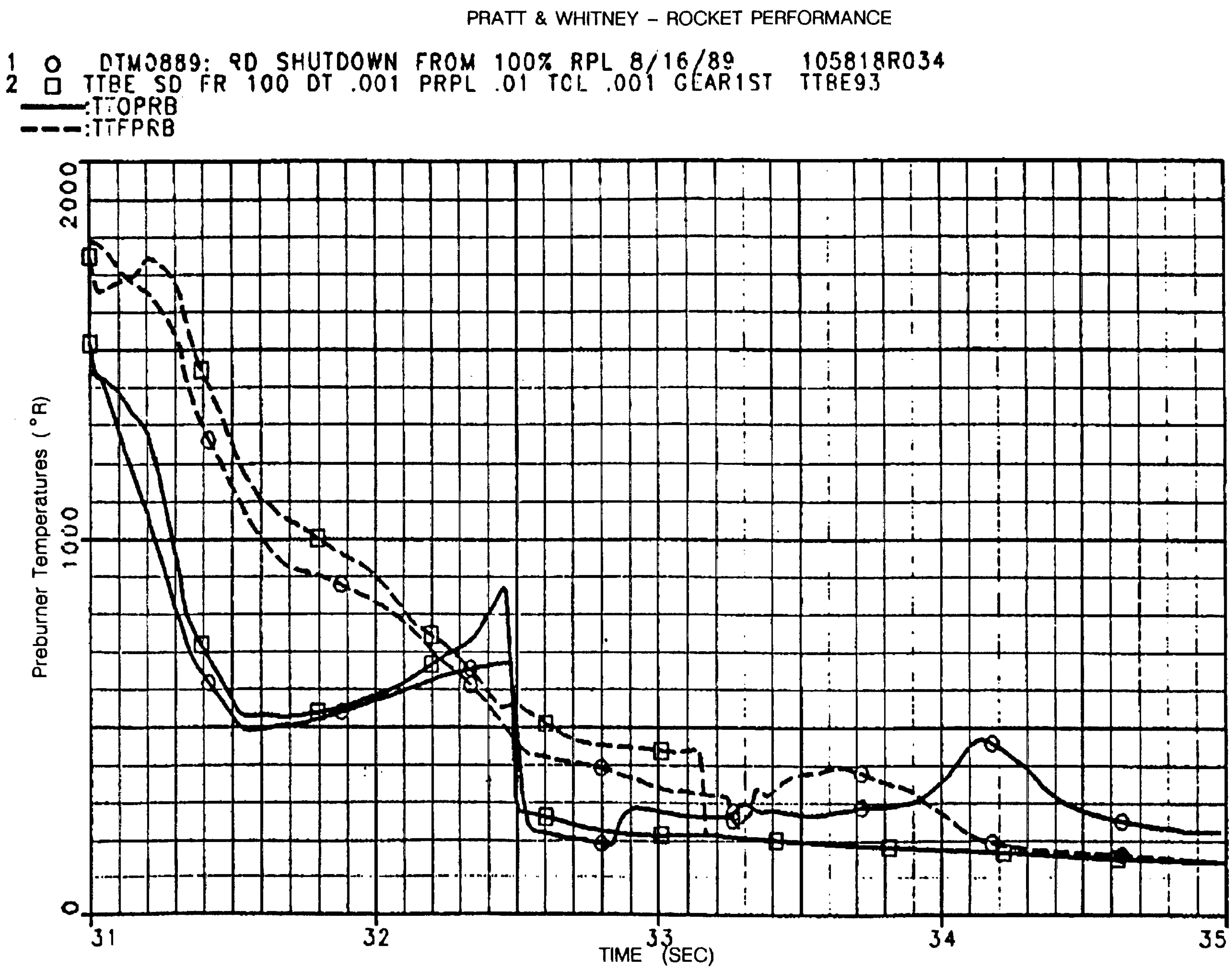

Figure 5-31 - Shutdown Preburners Temperatures

5.3.2 Start Transient

The start transient was simulated by providing the following changes in the TTBE MODEL. 1) Test 'ROTR01' was created with a minimum break-away torque requirement before allowing pump rotation. 2) Heat transfer Q's representing the latent heat of the nozzle were input as schedules of times. 3) The temperatures of the hardware metals were not integrated. 4) Calculations were added to simulate the filling, or priming, of the LOX injectors for the two preburners and the main chamber. The filling representations were the simple models taken from the DTM, but not the detailed, multi-volumes models of the preburners which were used in the DTM predictions.

The start predictions of the TTBE model are presented on Figures 5-32 to 5-40 with predictions of the DTM for reference. The rotor speeds and pressure/temperature of the main chamber and preburners are presented. Some of the differences between the predictions of the two models is due to the LOX injectors filling times (see Table 5-2). With the earlier fuel preburner priming the fuel speed of the TTBE leads the DTM at the 1.5 sec time (Figure 5-32). The higher chamber pressure results from the higher temperature of the TTBE after ignition (Figures 5-36 and 5-38). While other differences exist between the predictions of the two models, the verification test was to show the TTBE model in ROCETS could operate through all the transient phases. This was successfully accomplished including operation with the implicit integration scheme.

Table 5-2. LOX Injector Priming Times (Sec)

LOX Injector Priming Times (SEC)		
	TTBE	DTM
Fuel P/B	1.22	1.40
Main Chamber	1.55	1.50
Oxidizer P/B	2.00	1.60
Combustor Ignition Times (SEC)		
Fuel P/B	0.45	0.45
Oxidizer	0.90	0.90
Main Chamber	1.45	1.45

PRATT & WHITNEY - ROCKET PERFORMANCE

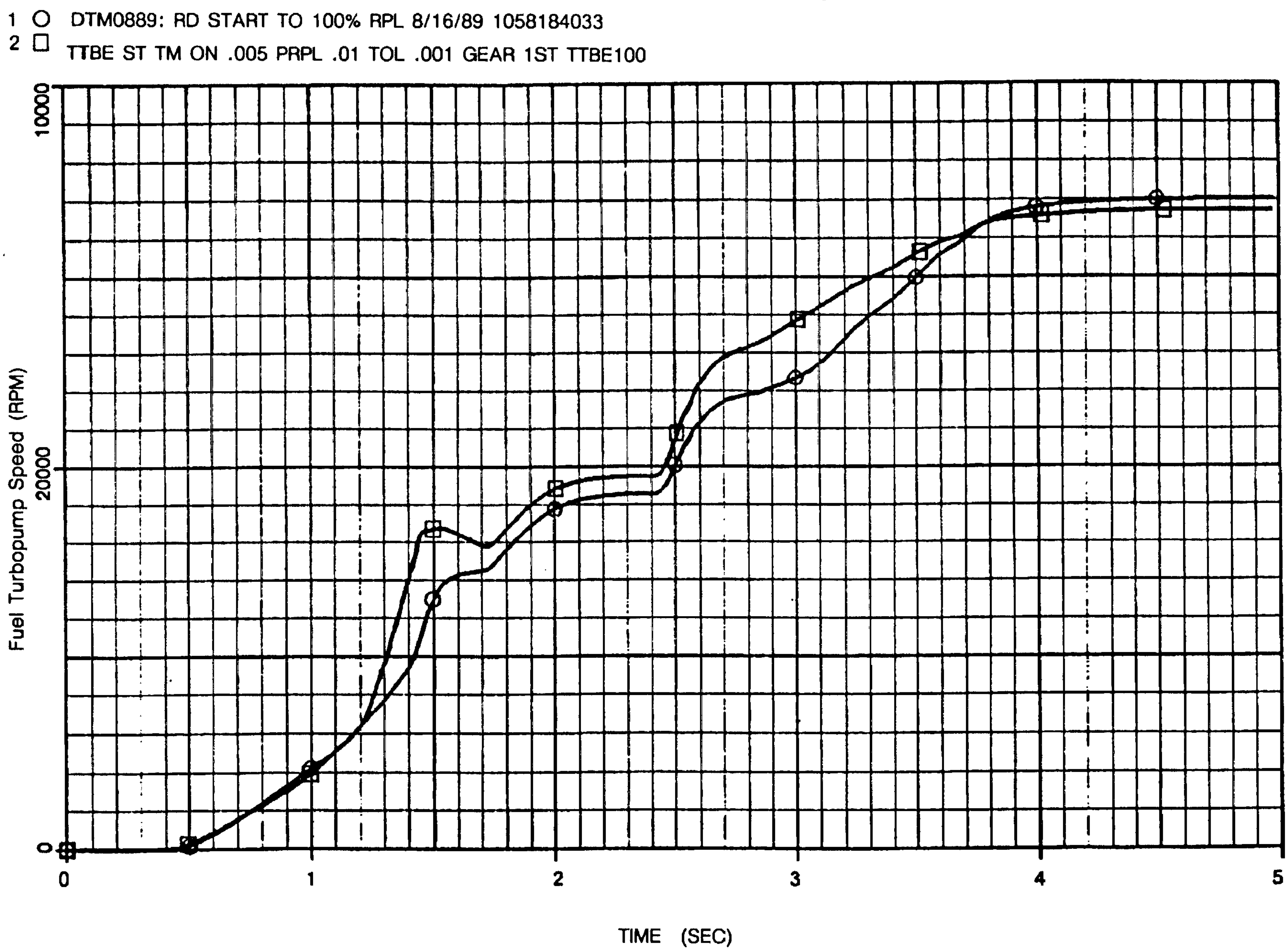

Figure 5-32. Start Fuel Turbopump Speed

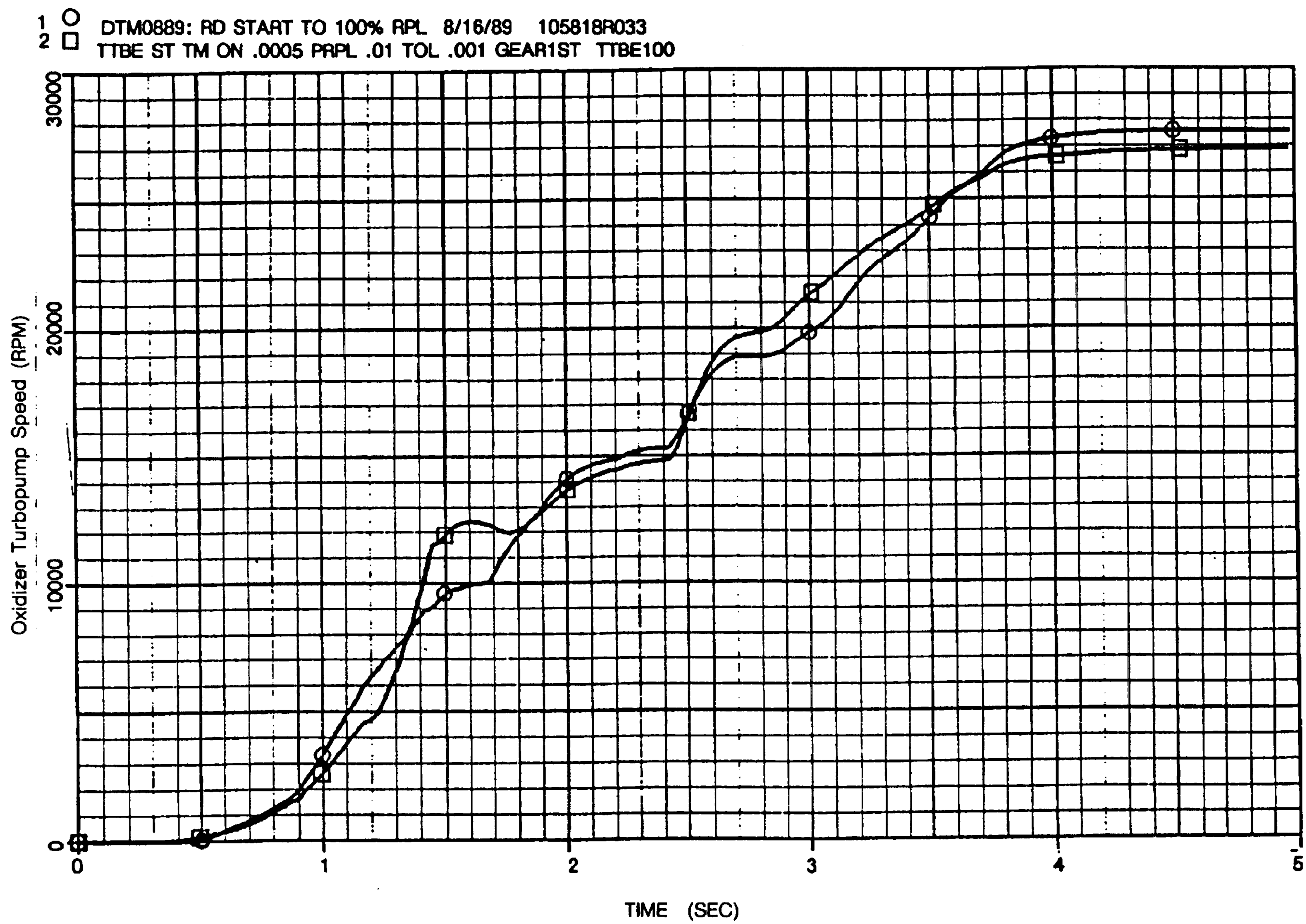

Figure 5-33. Start Oxidizer Turbopump Speed

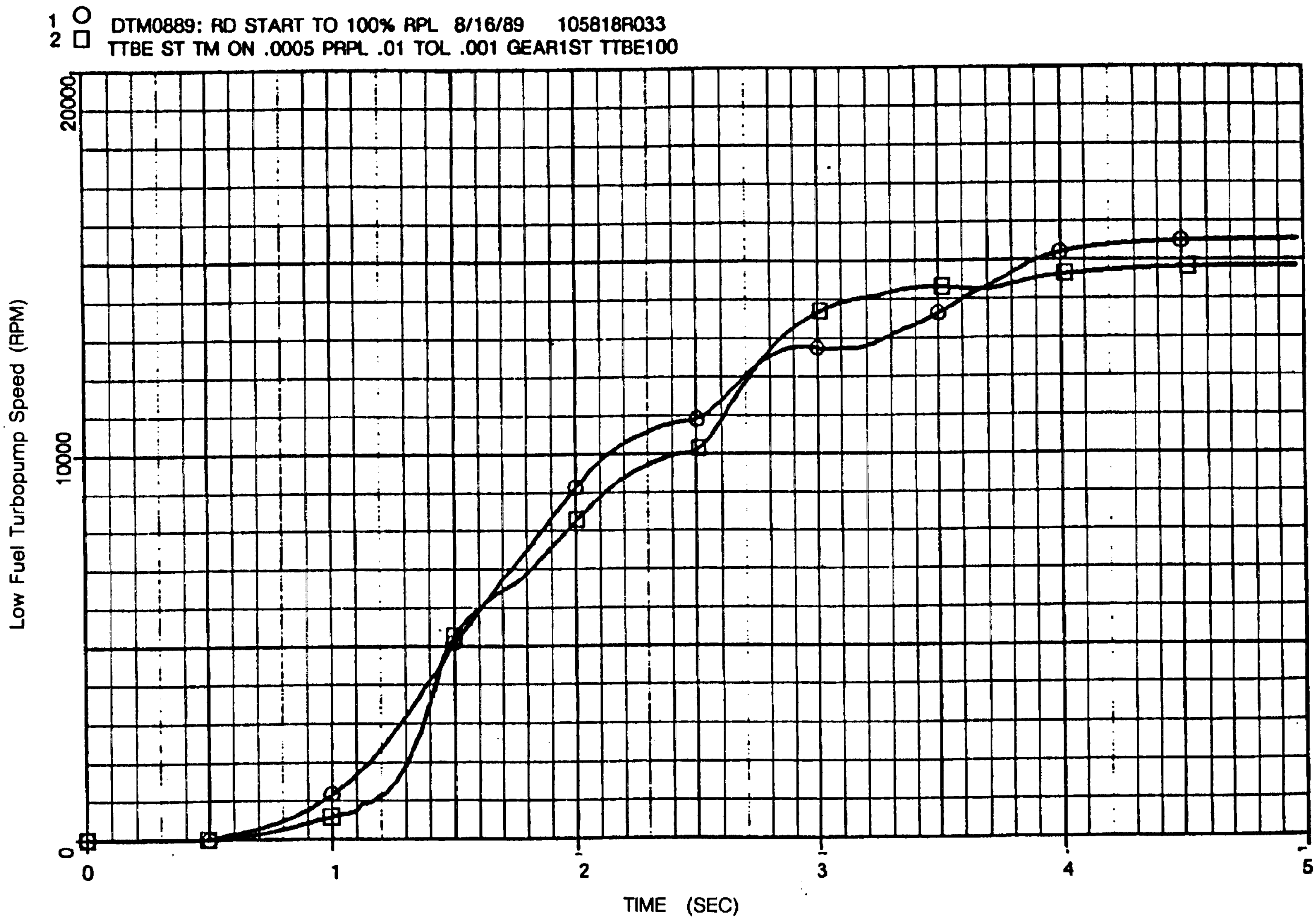

Figure 5-34. Start Low Fuel Turbopump Speed

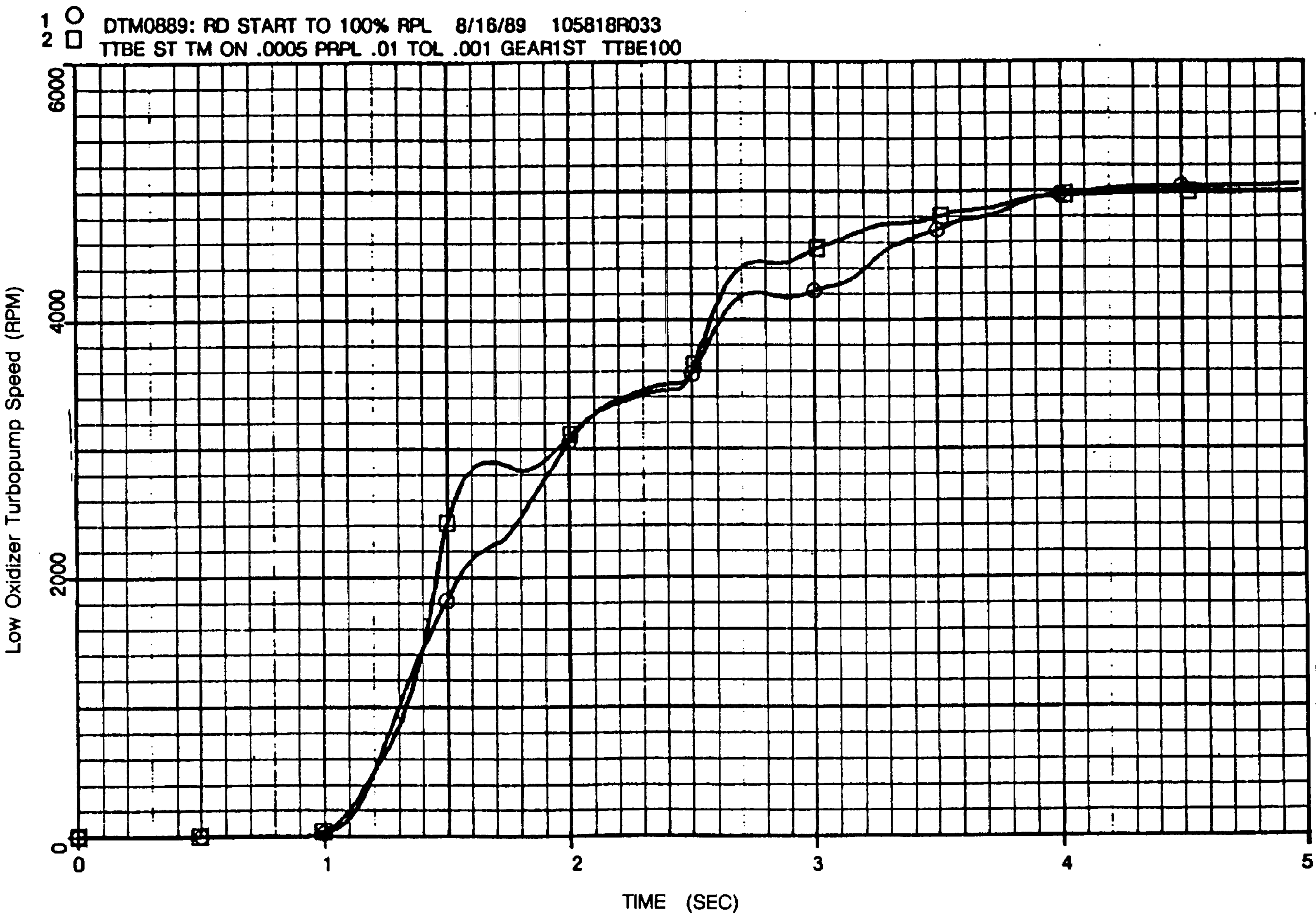

Figure 5-35. Start Low Oxidizer Turbopump Speed

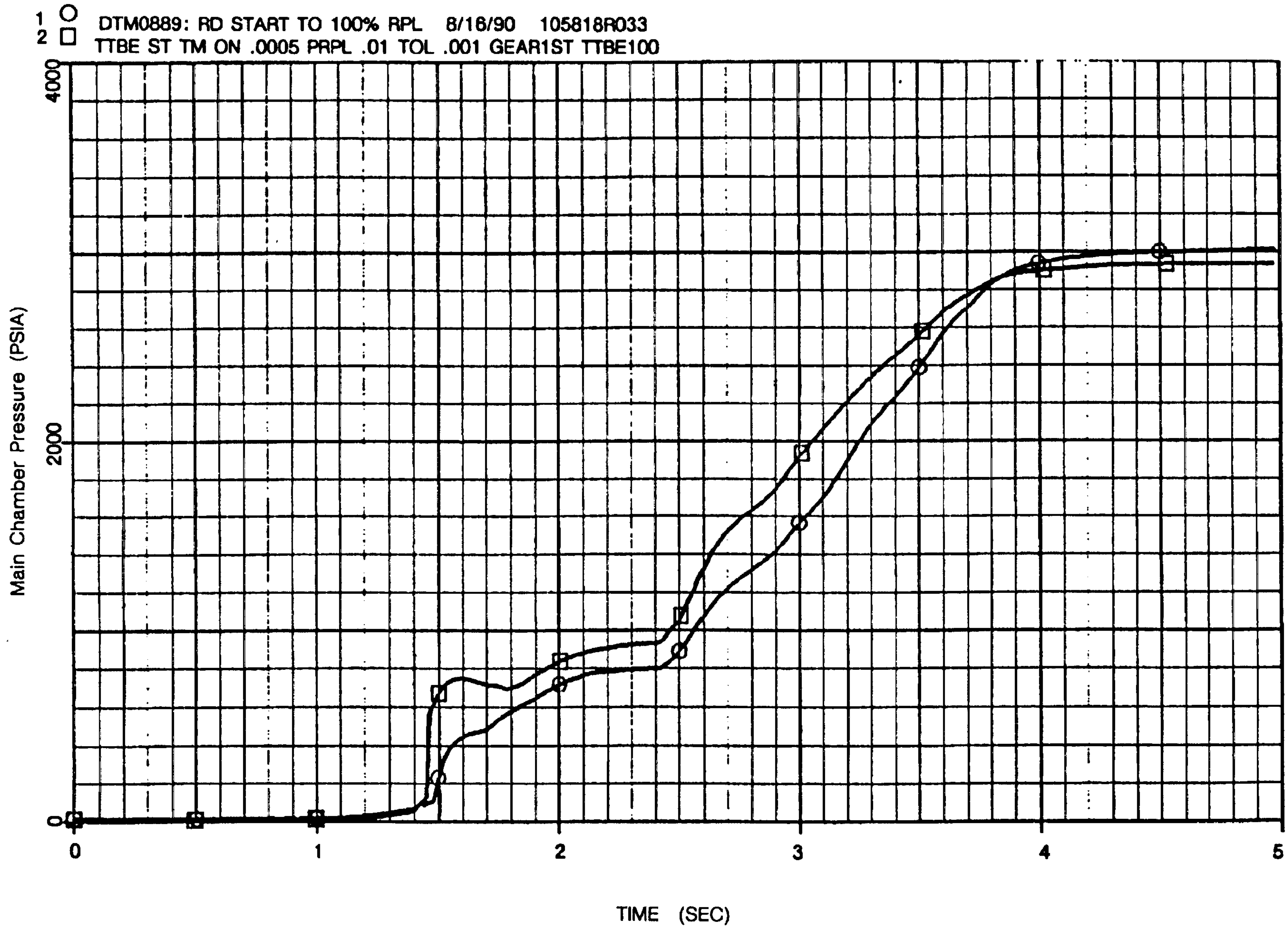

Figure 5-36. Start Main Chamber Pressure

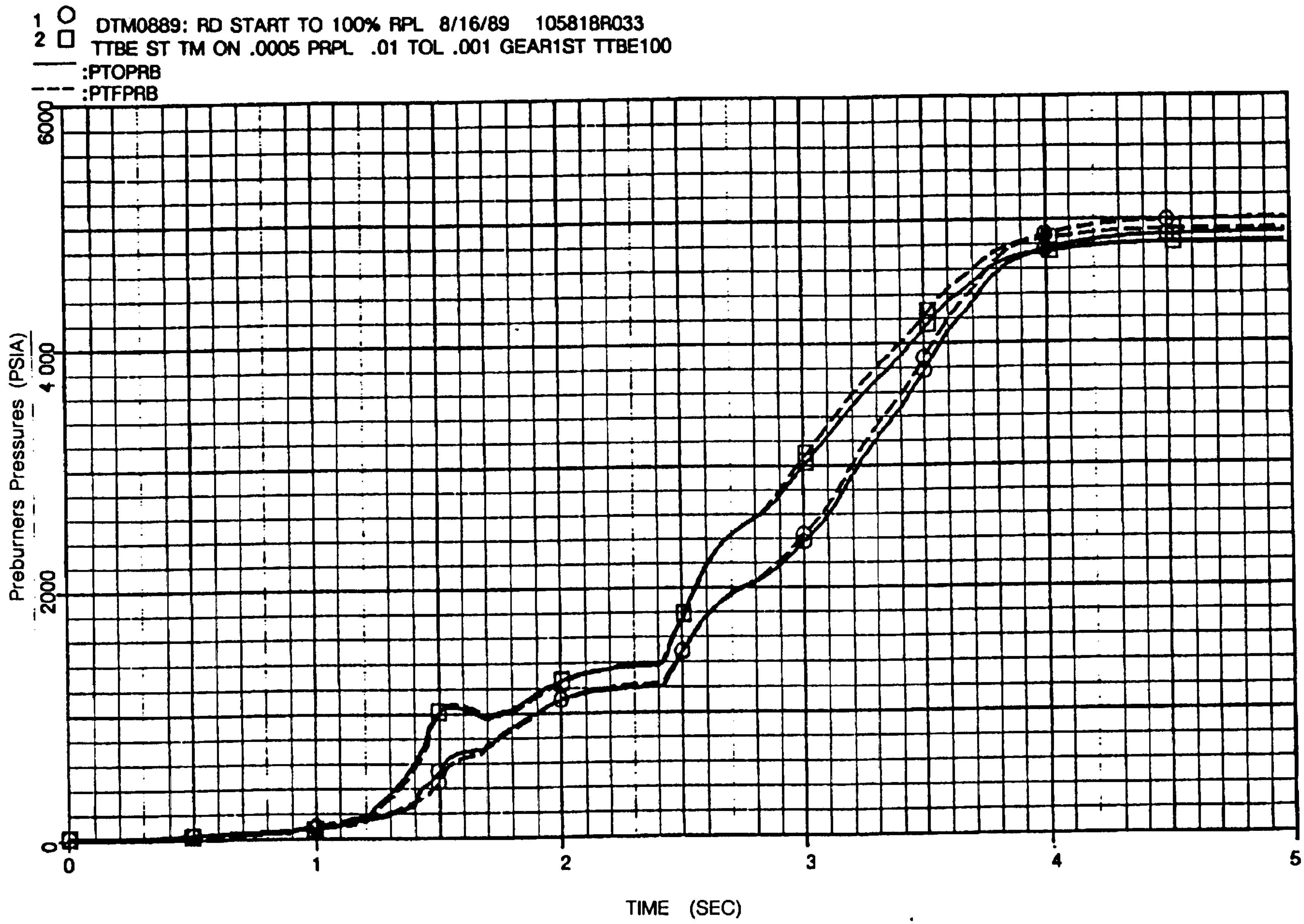

Figure 5-37. Start Preburner Pressure

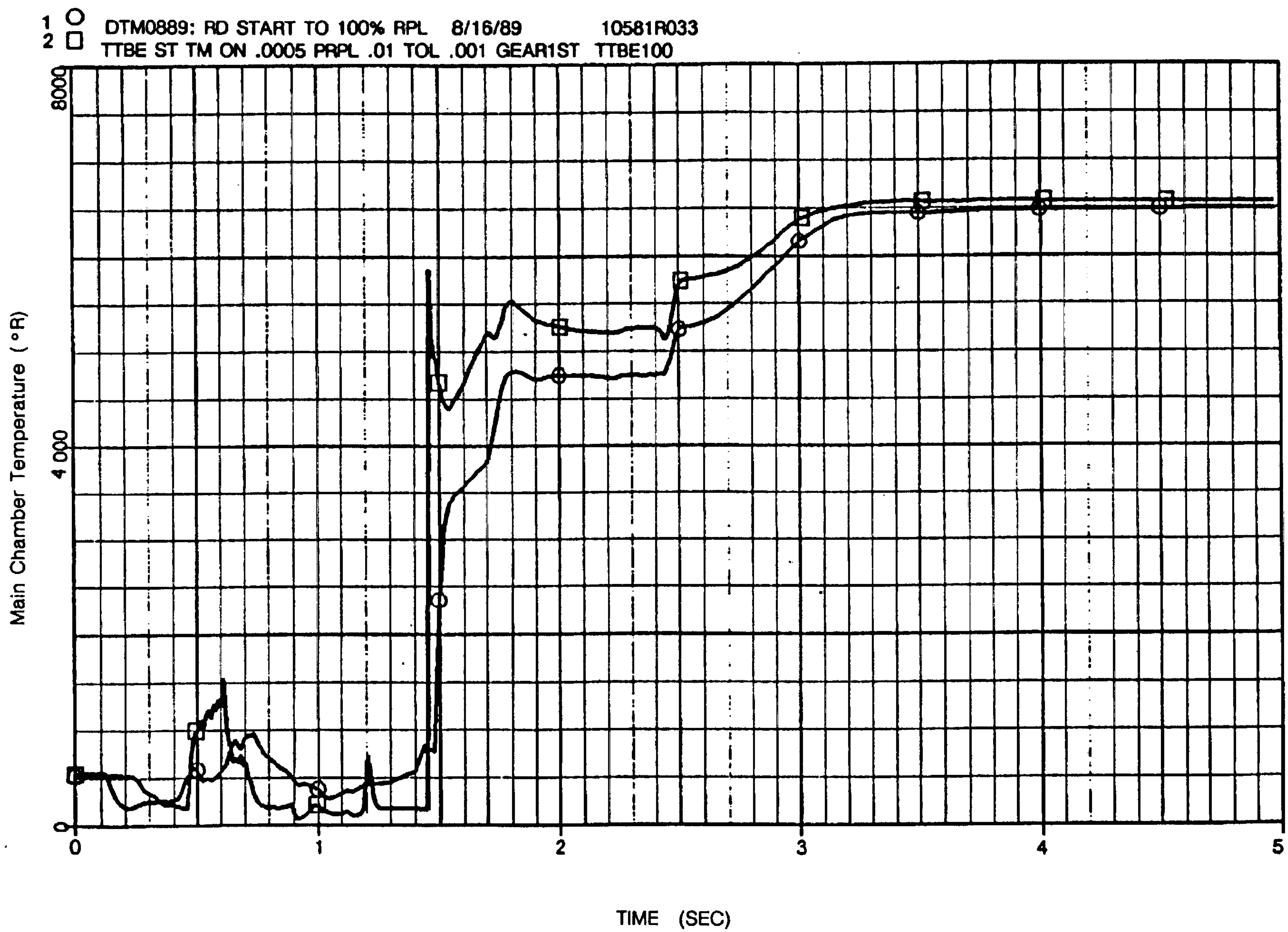

Figure 5-38. Start Main Chamber Temperature

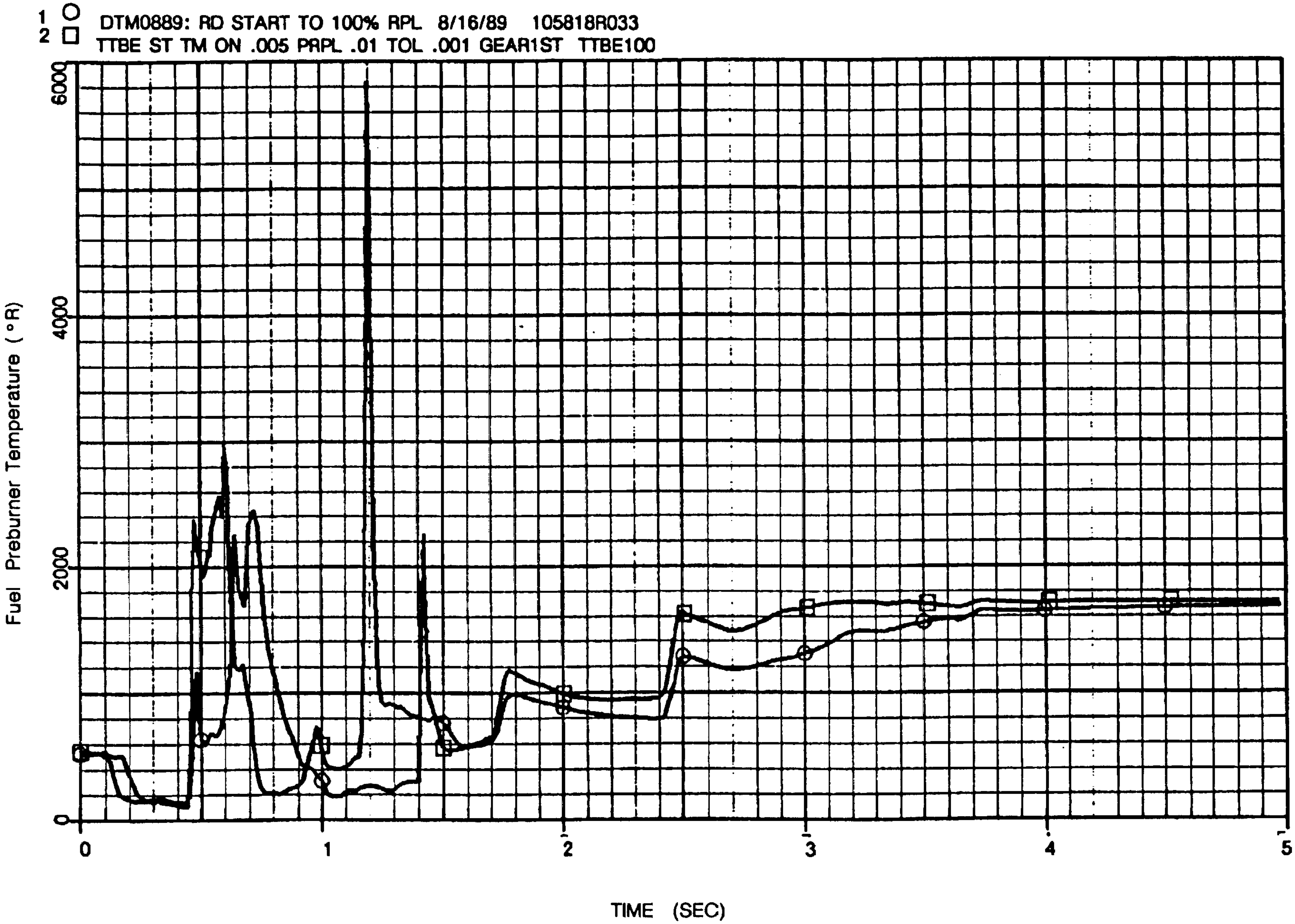

Figure 5-39. Start Fuel Preburner Temperature

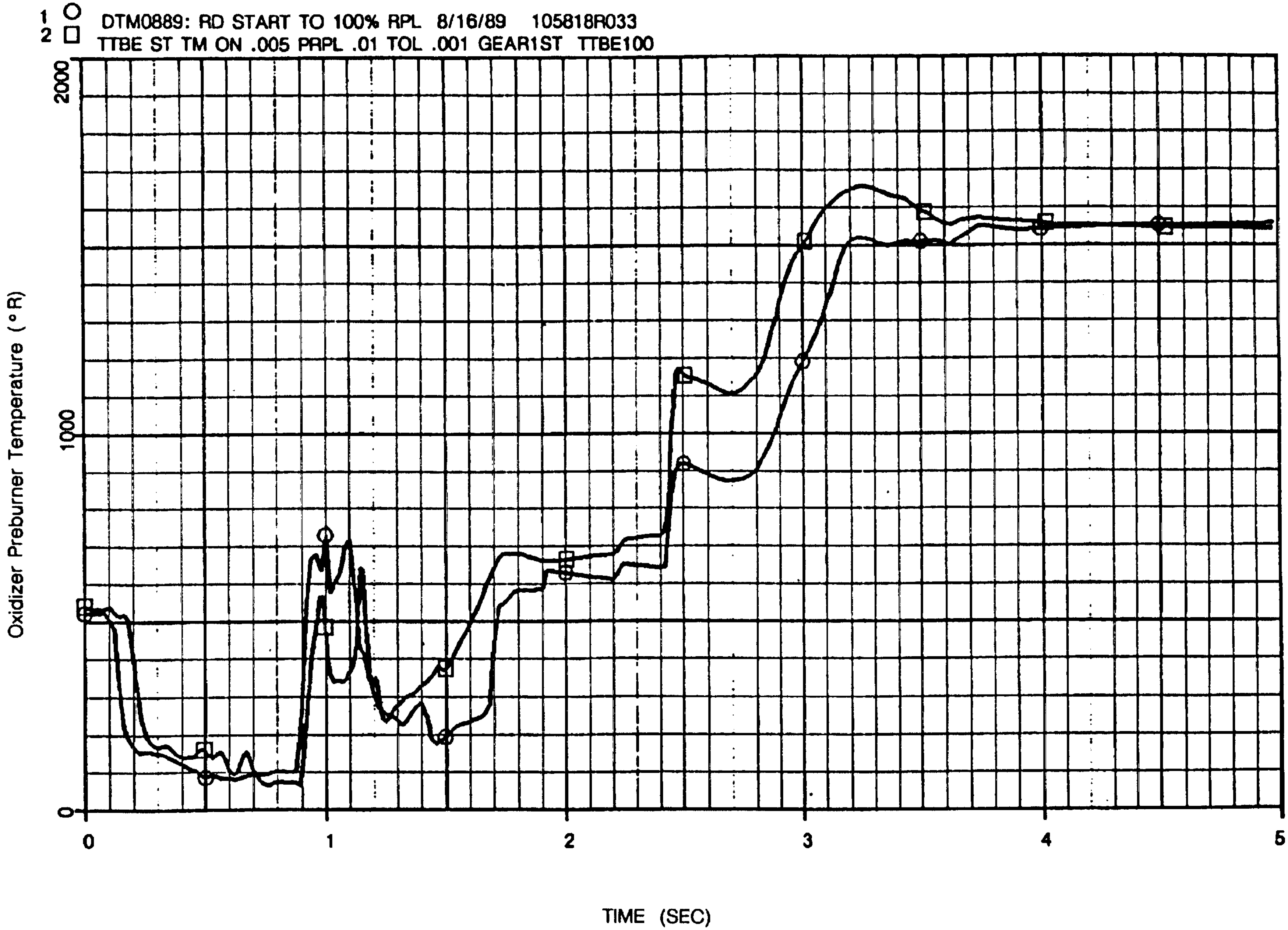

Figure 5-40. Start Oxidizer Preburner Temperature

5.3.3 Closed-Loop With Control

A NASA-MSFC control Model was interfaced in the ROCETS system (Appendix C) and used for closed-loop operation with the TTBE Model. Figures 5-41 through 5-44 present results for a throttle transient from 100% power to 65% power and back to 100% power. Parameters shown are chamber pressure, mixture ratio, and the four rotor speeds.

5.4 SUB-SET MODEL GENERATION TEST

To verify the generation of the linear model partials, a linear, sub-set model of the detailed TTBE model was created. Then the linear model time domain response was compared to the non-linear model predictions.

The detailed TTBE model had 122 states and 14 algebraic balances. Of the 122 states, 60 are using iteration parameters other than the states (i.e., 30 volumes are using pressure and enthalpy as the iteration variables to close the density and internal energy corrector equations). To reduce the linear model order to a manageable size for the verification test of the new partial generation technique, all states were set to be driven to their steady-state values except for the four rotor speeds. Thus, of the 136 TTBE simulation equations, 132 were analytically eliminated leaving a 4 state model.

The linear model was generated at 100% RPL with a 0.1% perturbation size. The oxidizer preburner oxidizer valve area was used as the model input, and pressure at the low pressure fuel pump discharge was the model output. Time domain results were obtained using approximately a 1.25% step on valve area by first generating transfer functions from the linear model matrices and performing an inverse Laplace transform.

The non-linear model was executed using the same constraints (i.e., all states forced to steady-state except for the four rotor speeds) for comparison to the linear model results. It should be noted that a steady-state balance was not performed prior to initiating the time transient, so some initial drift is observed.

Figures 5-45 through 5-48 present comparisons of the linear model to the non-linear model. Excellent agreement is observed, especially considering that the time response has an order of magnitude larger step than the perturbation size used to generate the partials.

The excellent agreement is verification of the linear model generation method. it involves a change-of-variables for 60 states and analytically eliminating the 14 algebraic balances and the 118 states which were set to steady-state. The partial generation technique provides a powerful tool for performing linear analysis and generation of reduced-order models.

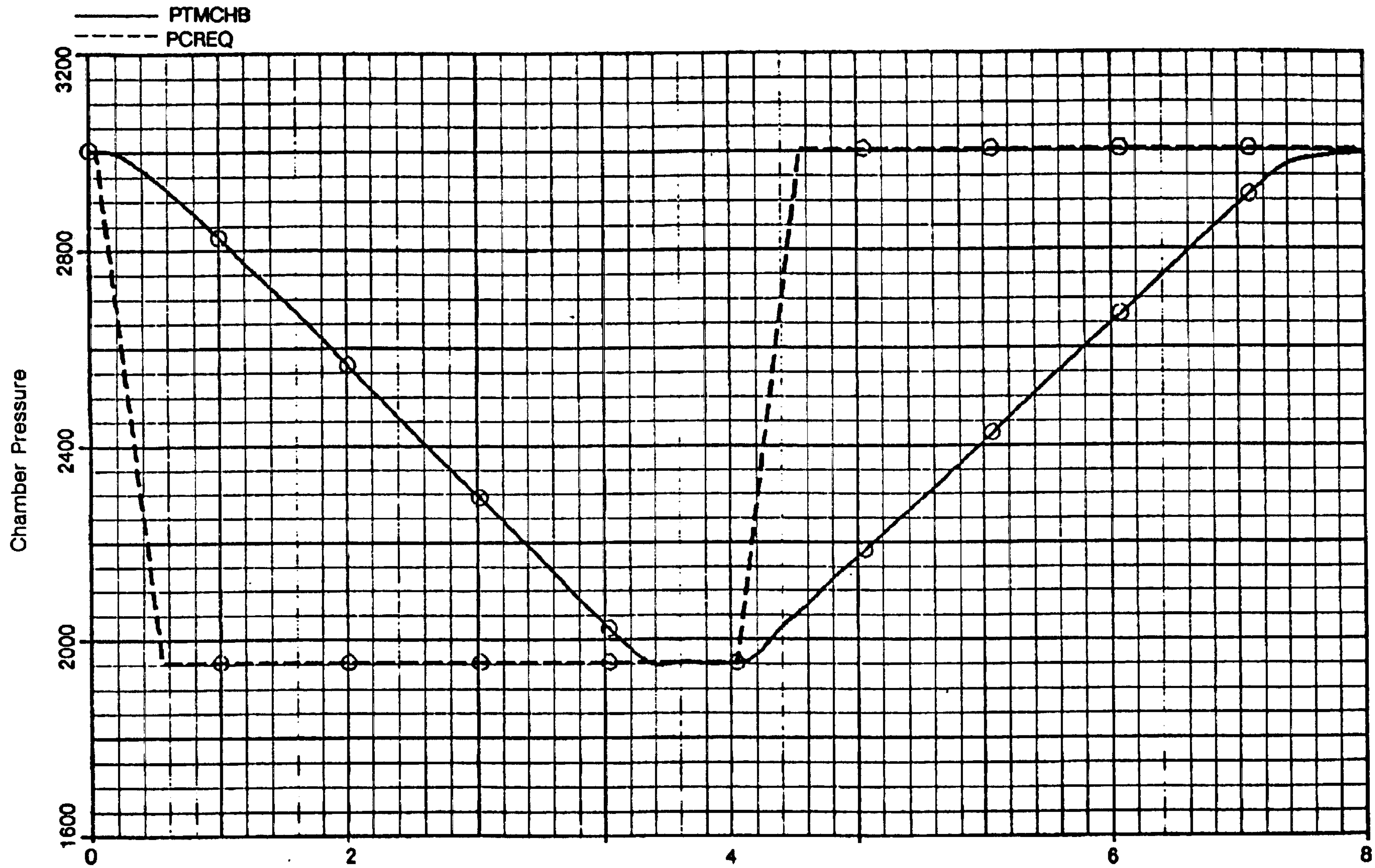

Figure 5-41. Chamber Pressure During Throttle Transient

C-2

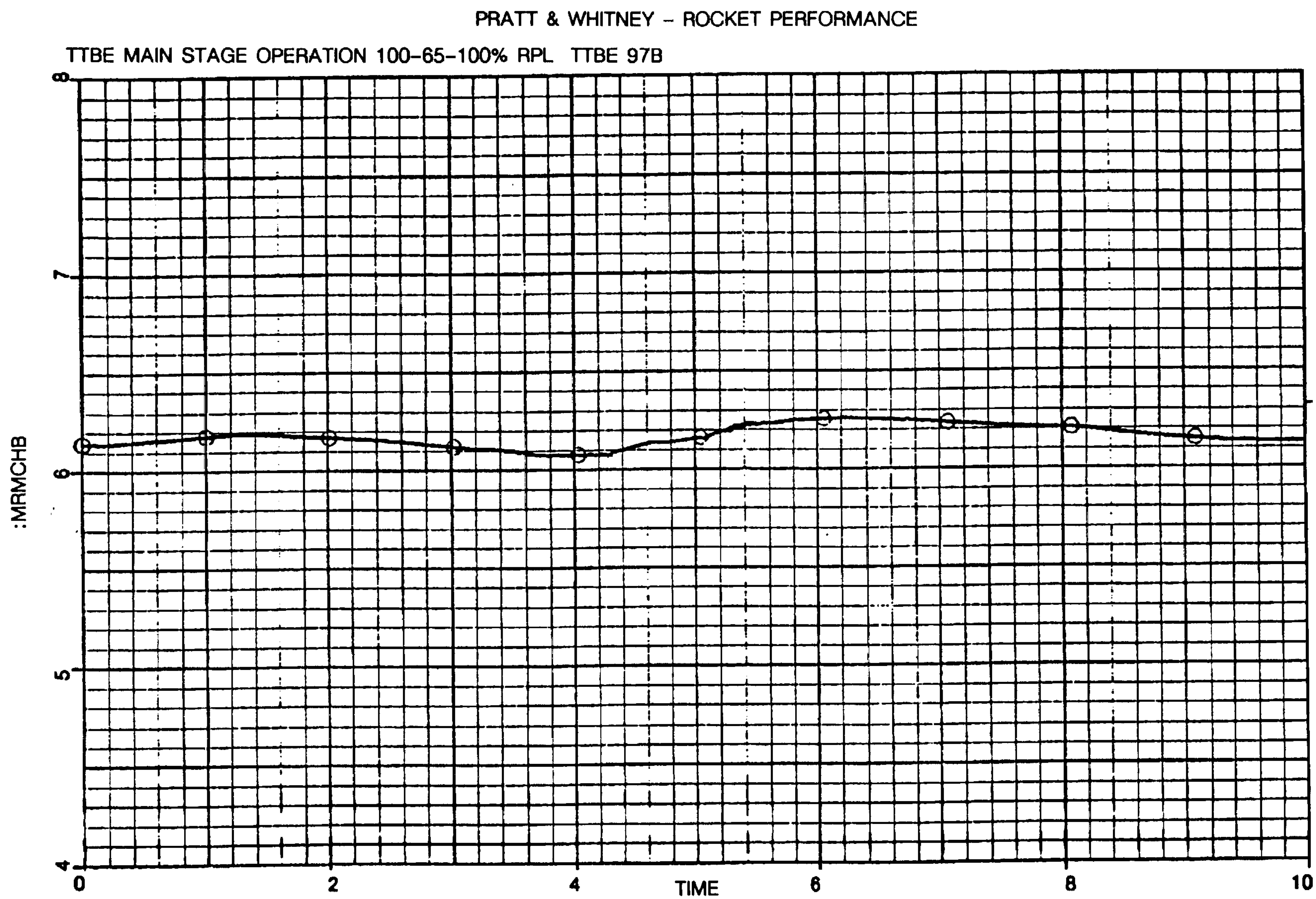

Figure 5-42. Mixture Ratio During Throttle Transient

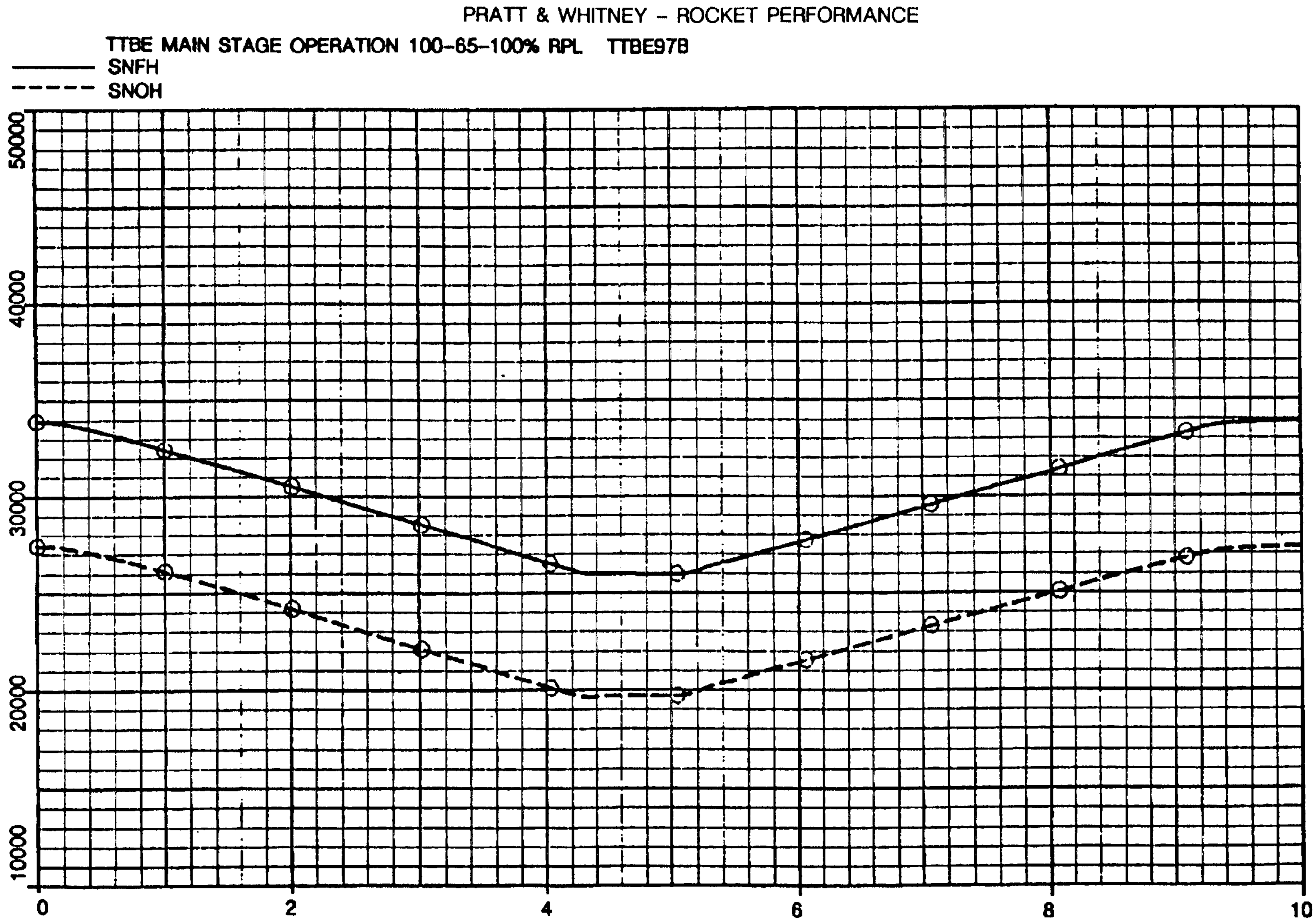

Figure 5-43. High Spool Speeds During Throttle Transient

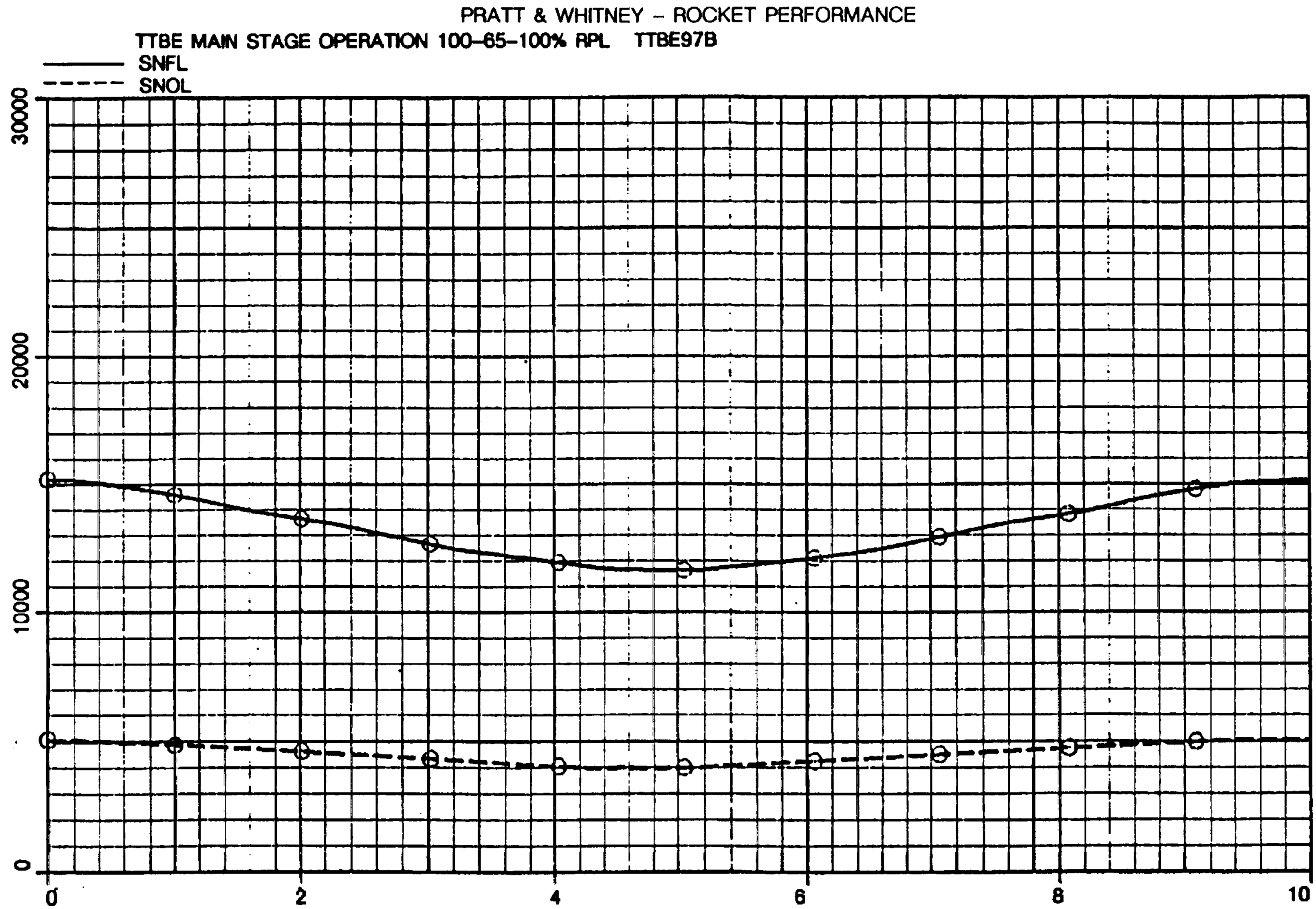

Figure 5-44. Low Spool Speeds During Throttle Transient

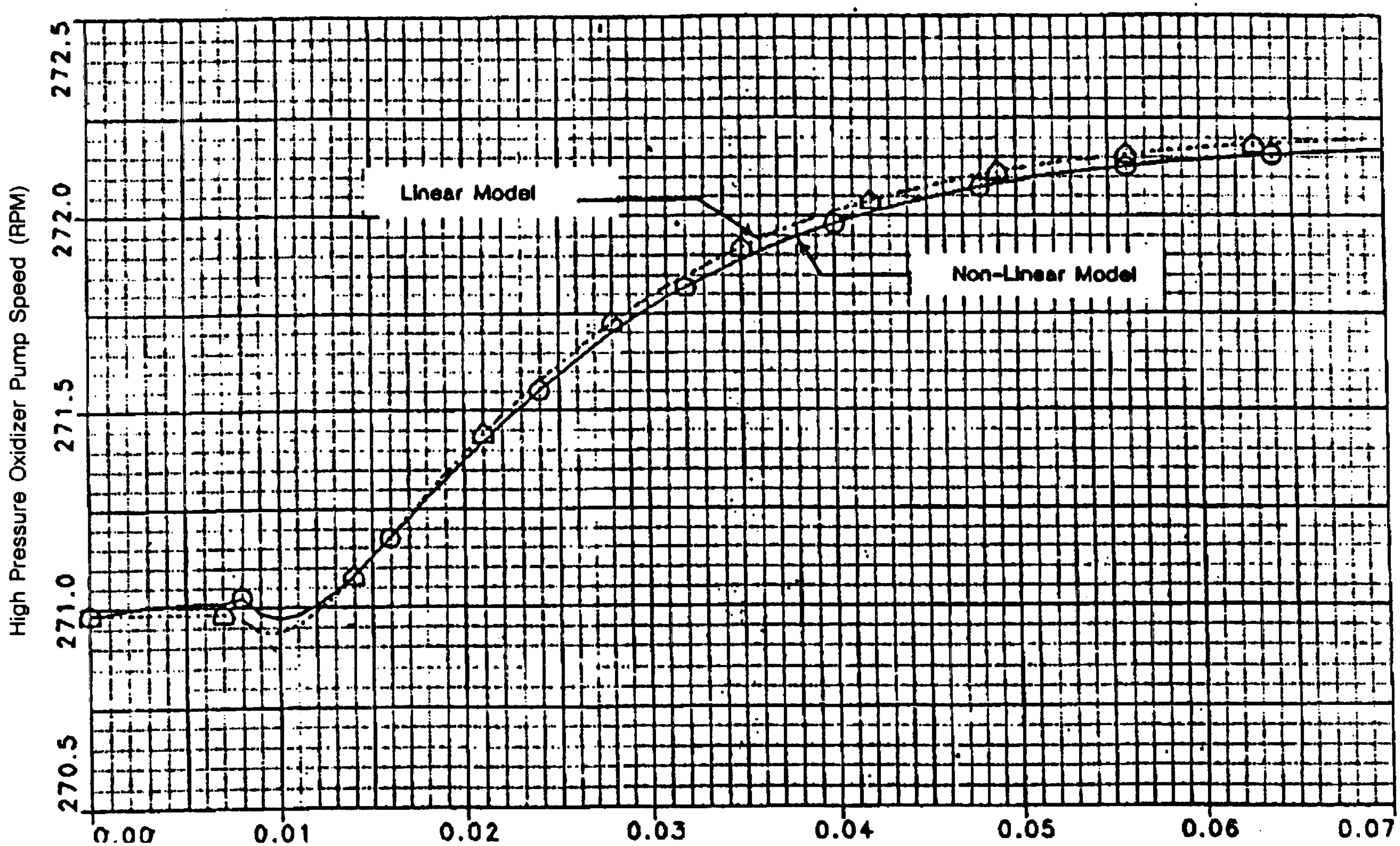

Figure 5-45. Linear and Non-Linear Model Comparison for High Pressure Oxidizer Pump Speed

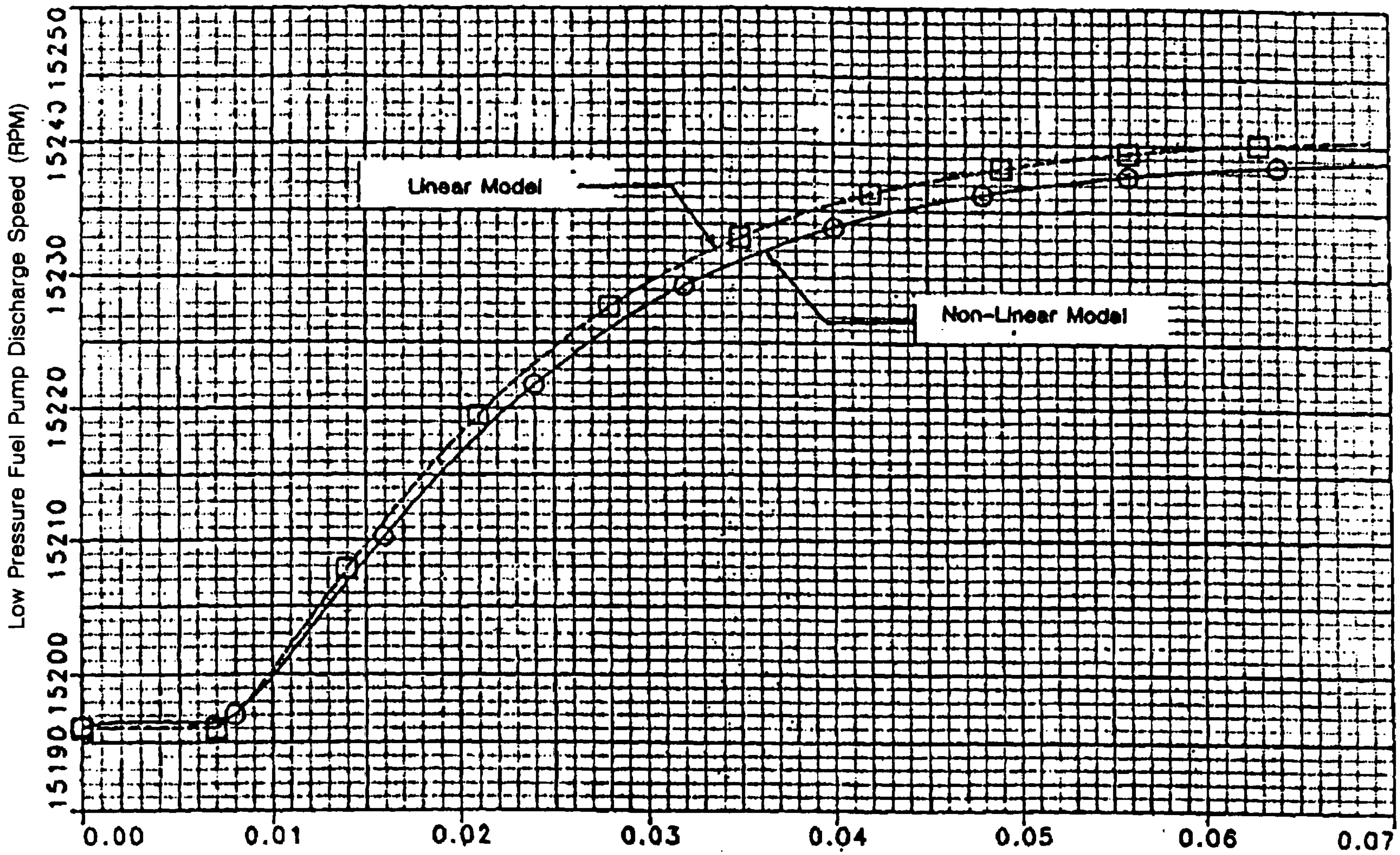

Figure 5-46. Linear and Non-Linear Model Comparison for Low Pressure Fuel Pump Spped

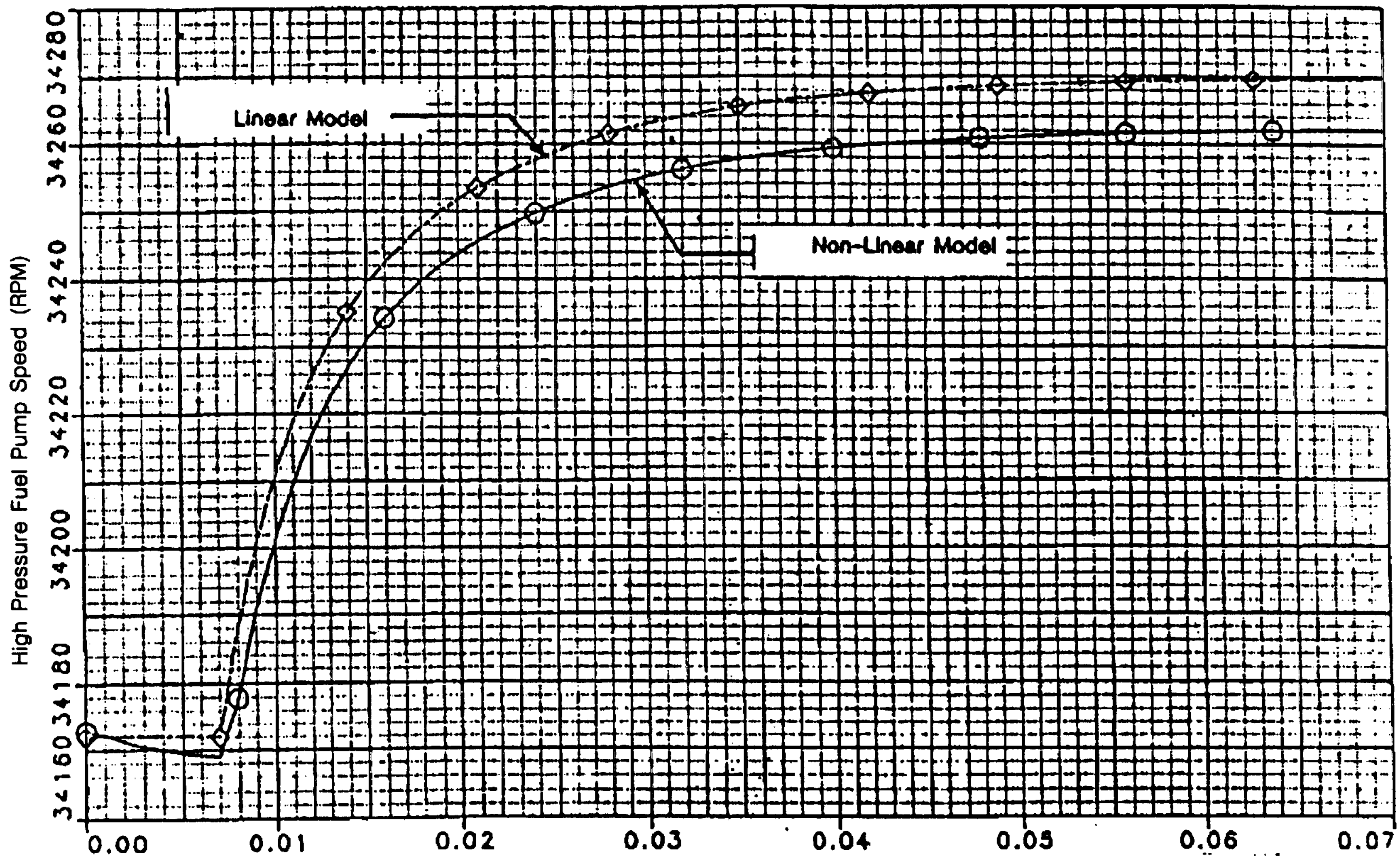

Figure 5-47. Linear and Non-Linear Model Comparison for
High Pressure Fuel Pump Speed

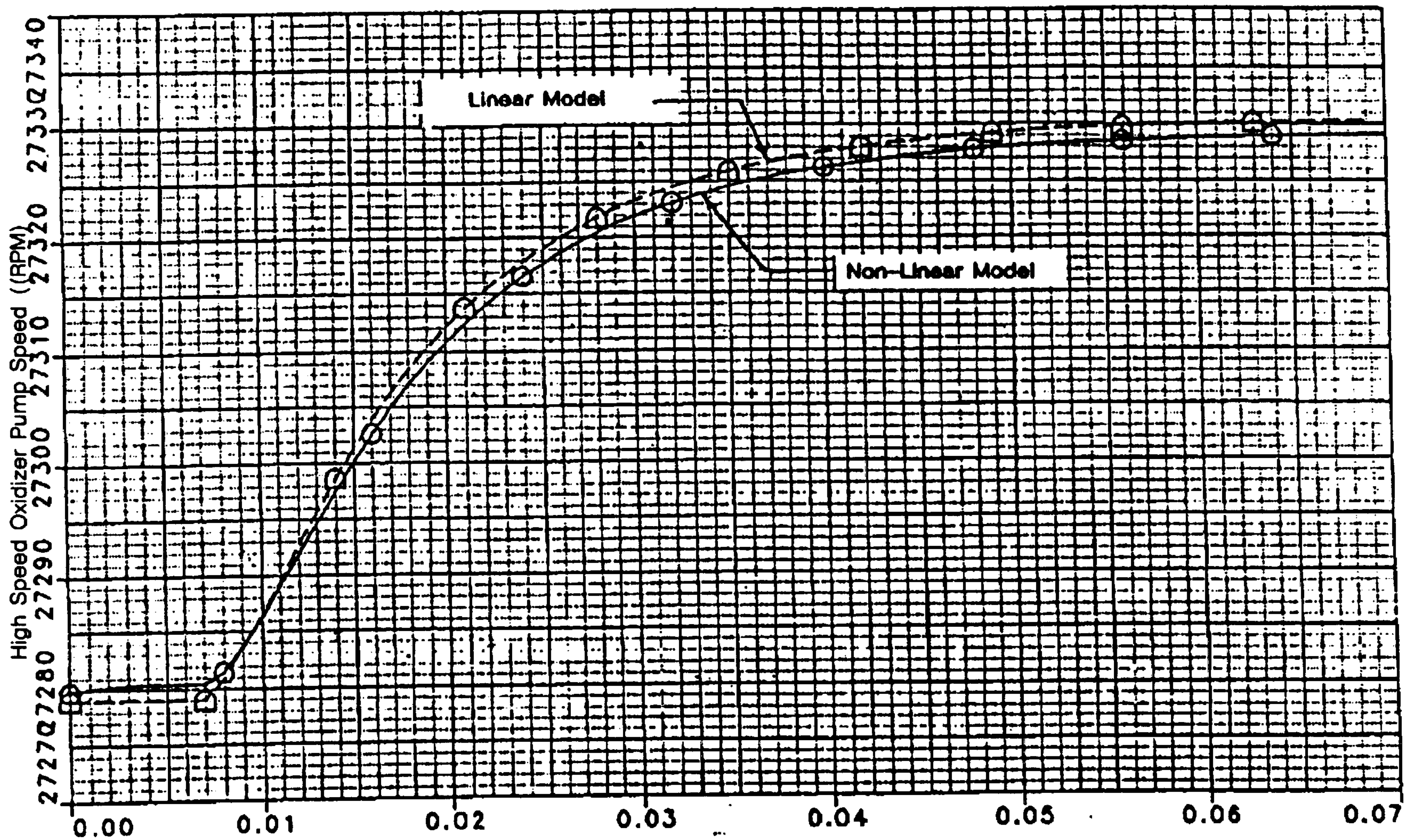

Figure 5-48. Linear and Non-Linear Model Comparison for High Pressure Oxidizer Pump Speed

This page left intentionally blank.

SECTION VI
CONTRACT END ITEMS

This Section presents the 23 contract end items which in general provided the overall requirements for the simulation system. Each contract end item is presented, followed by the program accomplishments.

1. The simulation system must have the inherent capability to be applied to current and projected liquid rocket engine cycles including: Staged combustion, Expander, Gas Generator, and Tri-Propellant. Projected engine applications include: TTBE, STBE, STME, and OTVE.

   ROCETS is a very flexible simulation system without any built-in rocket engine cycle or configuration. The user can select component modules and designate the interfaces to configure virtually any conceivable rocket engine.

2. The simulation system must simulate start, main stage and shutdown phases of engine operation.

   The TTBE model was operated in all three modes as presented in Section 5 - System Testing and Verification.

3. The system must include the capability to operate the plant in both open and closed loop control. This requires that provision must be made for attaching a control submodel which may be either sampled data or continuous.

   The TTBE model start and shut-down transients were operated open-loop. A NASA-MSFC FORTRAN control model (Appendix C) was operated closed-loop with the TTBE model in main stage operation as presented in Section 5 - System Testing and Verification.

4. Methodology must be created to allow representation of various failure modes and off nominal operating conditions including random parameter variations within each submodel.

   ROCETS was designed with generic component modules which (based on user input) call a designated sub-module which provides the specific component performance characteristic. Component failure implies the component performance (map characteristic) changes drastically. ROCETS can utilize modules which accept a failure flag to switch from a normal operating characteristic to a failed performance characteristic.

5. The simulation system shall be organized so that multiple levels of detail may be user selected for each sub-model where appropriate. This requires that both highly detailed simulation modes and "quick and dirty" simulation modes may be selected at user discretion.

   ROCETS is designed to configure an engine simulation based on user defined modules which leaves the amount of model detail up to the user. During the program, a simple TTBE model and a detailed TTBE model were generated and operated.

6. The simulation system shall be designed with a minimum bandwidth goal of 300 Hz for the detailed simulation mode. In some instances it might be appropriate to model higher frequency dynamics.

   ROCETS has several features to enhance transient operation. Integration methods include trapezoidal and Gear (first and second order), while other methods can be adapted if required in the future. Implicit (closed-loop) integration is recommended for cost-effective computer operation, but open-loop Euler integration is also available. A relative time constant is calculated in each module and compared to the simulated time step, to automatically select integration or differentiation to be used. Therefore, ROCETS can not only operate models up to 300 Hz, it also operates them in an efficient manner.

7. The simulation system shall be designed so major system components, i.e., the generic submodel library, the engine specific data, the generic data, and the simulation experiment data, will be separate and distinct.

   The ROCETS system has component modules with generic calculations which call sub-modules with specific component performance characteristics and data. Properties are called when required through a module for each fluid and sub-modules which contain the various property maps. After a simulation has been configured, the user defines the simulation experiment with inputs to the run processor.

8. Input data to define a particular engine shall be defined in terms of design data as opposed to model parameter data. Likewise, the empirical data necessary to characterize should be defined in terms of industry standard practice. For example, a dynamic model of a sensor will usually be given in transfer function form. Turbine performance maps will be nondimensionalized. This requires that a design data to model data to model data (sic) translator component be developed.

   The ROCETS system was designed with a component-by-component module and sub-module performance characteristic concept. This allows the user to build-in conversions of design data to model parameter data as required.

9. A consistent set of nomenclature, model generation coding style, and documentation requirements shall be defined and adhered to. This requires that the code must be self documenting to the extent possible.

   The ROCETS system software standards are presented in the SDS, P&W FR-20284 (Reference 4). An example of self-documenting code based on these standards is presented in Appendix B.

10. The simulation system shall be designed so that subset simulations may be readily derived from the transient simulation of an engine. These subset simulations include linear operating point simulations for controls design, fast operating nonlinear simulations for controls analysis and parametrics, and real time simulations for hardware-in-the-loop- testing.

    ROCETS provides the capability to generate linear partial derivatives around transient, or steady-state operating points, The matrices of these partials are output by the system for use in subset simulations or linear control analysis. Because ROCETS can quickly eliminate states in the non-linear simulations by forcing the derivatives to zero, it can be used to develop real-time models which require limited number of states.

11. The simulation system will provide some method of warning the user when a simulation run uses out of range data, such as requesting thermodynamic property routines to extrapolate to 6000 psia when the data is good to 5000 psia. It also shall be the user's option to limit the warning and/or utilize it as a stopping condition.

    This was accomplished with good traceability and warnings arranged in different levels of severity as discussed in the User's Manual (Appendix A).

12. All generic data and mathematical models utilized in the simulation system shall be documented in the code such that the user will know the source of the data and will know the limitations and assumptions under which the data was generated and employed in the system. Specifically, internal documentation shall include: precise explanation of program and subprogram purpose, identification of version data and number, identification and description of all inputs and outputs, and identification of all blocks of mathematical calculations.

    This was accomplished and can be viewed in the example pump module (Appendix B) and in the other system modules and sub-modules of the SDS (Reference 4).

13. The simulation system shall be generated in the "Advanced Continuous Simulation Language" and in FORTRAN 77 unless an overriding justification can be made for an alternate approach. Such justification would be if an alternative were shown to be obviously and substantially superior to ACSL, or if a necessary capability were identified which would be prohibitive to develop in ACSL.

    The ACSL requirement was eliminated at the Critical Design Review at MSFC on 21 July 1988, because of the following justification: The ACSL system uses a FORTRAN labeled common structure to communicate between the ACSL FORTRAN modules. These common statements are built through an internal algorithm and are not structured in a predictable format, making user interfacing with ACSL modules very difficult. On the other hand, ACSL as a system is not structured to generate large, detailed rocket simulations from user supplied FORTRAN modules and operate the simulation in an efficient manner. Therefore, the ROCETS system should not be generated in ACSL.

14. The approach to be taken in mathematical modelling shall always give preference to first principals models first, empirical correlations second, and a transfer function approach third. For example, it is important to use first principals models of volume filling and gross heat transfer when modelling an injector prime. On the other hand, turbomachinery performance can be obtained by nondimensional performance maps so that simulation run time may be kept reasonable. Likewise, a sensor model need only be in transfer function form since any increase in detail would greatly encumber the simulation.

    In general, these guidelines were utilized in generating the modules to represent the TTBE model. The module building-block architecture of ROCETS allows component models with different levels of detail to be substituted if required for particular application.

15. In general the detailed mode of simulation should be sufficient to reflect the influence as would be measured by performance instrumentation and reflected in aggregate internal parameters of the following: design changes, property changes, start phenomena, shutdown phenomena, control logic performance, key parameters that limit operation like turbine temperature limits, instrumentation performance and location effects, engine performance variation, interface condition changes, and purge effects. This list is not all inclusive. The detail generally required is that reflected in the SSME DTM.

    The SSME DTM (Reference 2) was used as a guide to provide the amount of detail in the TTBE simulation.

16. The acceptance test of the simulation system shall be a complete simulation of the Technology Test Bed Engine. All thermodynamic and thermophysical property data generated for the simulation system must reflect the requirements that the TTBE has for such data. Likewise, heat transfer correlations must be valid in TTBE operating ranges. To provide

capability for the modeling system to be utilized in the study of hydrocarbon engines, thermodynamic and thermophysical data must also be supplied for at minimum the most likely hydrocarbon propellant candidate. NASA will specify the choice during Phase II efforts. These statements require that the data is for characterization of liquid hydrogen, liquid oxygen, hydrocarbon fuel, purge gasses, and their materials utilized in the TTBE.

The TTBE simulation was generated and used to verify the simulation system. Because of decreased interest in tri-propellant engines, NASA did release P&W from the hydrocarbon requirement at the 14 November 1989 meeting. The P&W system to be delivered will include methane thermodynamic properties as part of the property package, but will not include combustion properties of methane. The simulation system will accept data tables of combustion properties, and NASA can generate the properties in data table format if required for tri-propellant simulations in the future.

17. All typical liquid rocket engine components such as turbines, pumps, valves, ducts, accumulators, etc., shall be defined in generic fashion such that they can be connected in any user desired manner to simulate any of the engines or engine types listed earlier in this document.

    The ROCETS configuration processor allows the flexibility to generate simulations of any engine.

18. To verify proper operation, all normal operating modes of the TTBE will be simulated in both the detailed and the quick and dirty modes. In addition, the subset simulation generation capability must be exercised.

    As discussed in this report, a simple TTBE model and a detailed TTBE model were generated and operated. Linear partials were generated and verified by comparing a linear model prediction with the non-linear model prediction in the time domain.

19. To verify submodel operation, test requirements defined in task must include testing the operation of the submodel against known analytical solutions and experimentally verified data, when available in open literature.

    The system qualification test plans are written to verify module code by specifying tests to be performed and the required evaluation, including comparison source and acceptance criteria. As an example, the values in the property tables were compared to National Bureau of Standards data.

20. The simulation system shall be installed and proper operation verified on the MSFC EADS IBM 3083 computer system.

    This was accomplished.

21. At the completion of each sub-model or component, the code must be delivered to NASA MSFC for testing and utilization. all submodels and components must be delivered at least 3 months prior to contract completion in order to assure timely testing.

    The initial software delivery to NASA-MSFC was 27 December 1989, with updates on 5 March 1990 and 10 August 1990.

22. A review visit to MSFC will occur on or about six months intervals. A Critical Design Review will be performed as a part of the first review, with MSFC concurrence required for work to proceed. The results of tasks 1, 2 and 3, in Phase I of the activities shall be delivered as a document to be utilized in Critical Design Review.

The Critical Design Review was conducted 21 July 1988. Other reviews occurred on 9 December 1988, 27 July 1989, and 16 May 1990.

23. The final report will include a section listing the equations utilized with all ROCETS code.

    All of the equations of ROCETS are presented in the SDS, P&W FR-20284 (Reference 4).

This page intentionally was left blank.

SECTION VII
CONCLUSIONS

1. The ROCETS system is a valuable new tool which will save time and money in developing and using liquid rocket engine transient simulations.

2. The implicit integration scheme saves computing calculation time, and has been used successfully with the detailed TTBE model in simulating start, main stage, and shut-down transients.

3. The same simulation can be used for steady-state cycle balance as well as transient operation.

4. FORTRAN models developed outside the ROCETS system can easily be interfaced with and operate in the ROCETS system.

This page left intentionally blank.

SECTION VIII
RECOMMENDATIONS

1. The detailed TTBE model should be enhanced by verification with engine data.
2. The ROCETS system should be maintained with future changes and enhancements.
3. Potential ROCETS enhancements include:
   - All-electronic documentation and on-line user assistance
   - Improved linear partial generation technique

This page left intentionally blank.

SECTION IX
REFERENCES

1. Pratt & Whitney Engineering Software Development Standards (IDS–T0400)

2. Space Shuttle Main Engine (SSME) Digital Transient Model (DTM) 0889

3. ROCETS System Requirements Specification, Pratt & Whitney FR–20283, 25 November 1988.

4. ROCETS System Design Specification, Pratt & Whitney FR–20284, 25 July 1990.

5. Gear, C.W., "Numerical Initial Value Problems in Ordinary Differential Equations", Prentice–Hall, 1971.

6. Gear, C.W., "The Automatic Integration of Stiff Ordinary Differential Equations", University of Illinois, 1969.

7. Broyden, C.G. " Class of Methods for Solving Nonlinear simultaneous Equations", mathematics of computation, Vol. 19, 1965, pp. 577–593.

8. McLaughlin, P. "Replicas: A new Continuous System Simulation Language"

9. Biggs, R. "Space Shuttle Main Engine The First Ten Years", American Astronautical Society Paper, November 2, 1989.

This page was intentionally left blank.

Apppendix A
User's Manual

The User's Manual for ROCETS is contained in the SDS, P&W FR-20284 (Reference 4). It is reproduced in this report for reference.

Pratt & Whitney
FR - 20284
31 May 1990

ROCETS USER'S MANUAL

31 May 1990

United Technologies
Pratt & Whitney
Government Engine Business
West Palm Beach, Florida

Table of Contents

3.4.4.1 Introduction to ROCETS

ROCETS is an acronym for ROCket Engine Transient Simulation. The objective of the ROCETS system is to apply the state-of-the-art in modeling and simulation technology to simulating liquid rocket engines. The versatility of this system makes it ideal for the performance engineer, the system engineer, and the control engineer. Also, the structure of ROCETS makes it highly adaptable to simulate any type of rocket engine cycle with varying levels of modeling detail as desired by the user.

The ROCETS system is designed for use by engineers with average experience. While extensive modeling experience is not required, it is assumed that the user is familiar with modeling practices and techniques. The goal of ROCETS is to aid the user in creating and using a simulation by automatically generating an executable model from input, scanning the model for undefined variables or variables which require algebraic loops, and supplying state-of-the-art numerical techniques. A flexible run-time processor aids in defining inputs for a particular model experiment. In addition, the ROCETS system makes available fully verified engineering representations of most rocket engine components. The modules in ROCETS which implement non-linear engineering representations are written in structured FORTRAN77. The system also has provisions to generate linear partial derivatives at user selected points for subset models.

3.4.4.2 General Description

The ROCETS system implements engineering representation in the form of FORTRAN subroutines called "modules" The modules are stored on the ROCETS library and are accessible for generating simulations. A configuration processor is used to generate an executable simulation from user input. Once a simulation is generated, input is supplied to a run processor to execute a particular simulation experiment.

3.4.4.2.1 The Module

A module in the ROCETS system is a stand alone FORTRAN 77 subroutine which implements the engineering equations to represent a particular engine component A module is distinct from other types of subroutines in that only modules communicate directly with the main program All communication between modules is via the main program using named variables

As part of required user input when defining a particular simulation, each selected module must include a character name to distinguish the variables associated with that module from other variables in the simulation. The name can be up to four characters The actual variable names are formed by concatenating the module name with pre- defined system names for each type of variable. As an example, consider the variable name for density inside a volume Let the volume module name supplied by the user be 'VOL1'. The system name for density is 'RHO' so that the actual variable name is 'RHOVOL1'.

In addition to the system names to be used for the variables of each module a variable "tag" is contained in the comment cards at the beginning of each module The variable tag is used to group all the variables comprising the model into several categories depending on their function in the simulation. The categories are important because only certain variable types can be used for various functions.

3.4.4.2.2 The Sub-Module

Sub-modules are called by modules or other sub-modules by a FORTRAN subroutine call list. They are stand alone subroutines but, unlike modules, they do not communicate with the main program. Sub-modules are divided into map sub-modules and utility sub-modules. Map sub-modules are performance characteristics representing a particular component The user selects which map to use for a given component along with the ability to "scale" the map Utility sub-modules implement generalized functions and are typically analytic engineering representations or mathematical operations.

3.4.4.2.3 Variable Tags

States are variables for which derivatives are calculated and whose values will normally be obtained through numerical integration using a predictor-corrector scheme In addition to states, there can be State Iteration Variables. These are variables used as the independent variables for the iteration to close the corrector equations. In particular for rocket applications, it is useful to use pressure and enthalpy as the iteration variables for the density and internal energy states. First guesses for the states or the state iteration variables must be supplied.

External inputs are variables that are used but never calculated. They are tagged as external inputs because they must be supplied externally in some form by the user.

Design variables represent variables that are normally fixed for a given engine cycle. Examples are volume sizes, line lengths, etc. They have the same function as external inputs but a distinction is made for future system enhancements.

Outputs are simply variables that are output of a module and may be used as input to other modules downstream. No action concerning outputs is necessary by the user.

Independent Balance Parameters are variables that are used as the independent iteration parameter for an algebraic loop. Dependent Balance Parameters are the variables that form the error term.

3.4.4.2.4 Executive Programs

Four processors are used in the ROCETS system. A configuration processor reads the user configuration input, retrieves the specified modules and sub- modules from the ROCETS library system and builds the simulation. In setting up the simulation, the processor builds the communication structure along with global commons. It also builds the main program (subroutine ROCETS) with the calls to the engineering modules, and any property calls or equations as specified in the configuration input.

An input processor is used to interpret user input specifying parameters to define a particular model experiment. It consists of a set of callable FORTRAN subroutines that read user input, interpret the input, can load input variables into the commons, establish balances, and set necessary flags for model execution.

Execution control is provided by an execution processor. It controls looping, print, balancing, and linearization. Within the execution processor are calls to the numerical utilities that provide steady-state balancing, transient integration, and linearization. It provides a centralized location for all numerical operations so that adding new features to the system is simplified.

Output processing is controlled by an output processor that accepts input as to what parameters are to be printed and plotted. It has an interface routine for plot information so that it can be used with a variety of plotting software simply by changing the interface routine. However, linearization output is not controlled by the output processor, but rather all necessary information is passed to a separate interface routine for linearization output. This feature allows tailoring of the linearization output by changing only the interface routine.

3.4.4.3 Configuration Input

Configuration Input consists of user commands to build a particular simulation. It consists of required information regarding the algebraic engineering modules to be used and their placement. The system to be modeled is defined using the following keywords:

OPTIONS
INSTREAM
EXTERNALS
INTEGRATION
BALANCES
SYSTEM ABOVE, INSIDE, or BELOW

3.4.4.3.1 OPTIONS Keyword

The options block contains optional input to the processing program and should be located at the top of the configuration input file. Included are engineering units options, provision for a title, provision for specifying a PDS for those operating on MVS/TSO, and a cross reference option. The form of the OPTIONS block is:

```
DEFINE OPTIONS
    UNITS : ( ENGLISH or SI ) ;
    TITLE : ( The title of the model to be configured ) ;
    PDS : ( Data dictionary file name ) ;
    CROSS : ( ON / OFF ) ;
END OPTIONS
```

The UNITS keyword is used to specify the engineering units to be used. Units must be either ENGLISH or SI.

The TITLE keyword is used to specify a 50 character title that will be placed at the top of the main FORTRAN program.

The PDS keyword is used to specify the name of the file which contains the data dictionary. This file should contain the INTERFACE, UNITS, and KEYWORDS blocks for all of the modules that are in the ROCETS System

The CROSS keyword is used to turn the cross reference output from the configuration processor on or off. The cross reference output contains an alphabetized list of every occurrence of every parameter in the model (with the exception of some global variables like IPRPL and IUPDATE). The cross reference output can be very useful in debugging a simulation and in verifying that the configuration processor produced the desired simulation.

3.4.4.3.2 INSTREAM Keyword

The define instream block contains a list of modules to be used which are not in the system. The list includes both the module name and a designation of the file in which the module source code resides. For CMS users the file designation consists of a file name, a file type and a file mode. For MVS/TSO users the file designation is the complete file name. The processor will read each file and interpret the interface information. The format is:

```
DEFINE INSTREAM
     ( Module name ) : ( File  designation ) ;

     ( Module name ) : ( File designation )  ;
END INSTREAM
```

For example, if you wanted to test a new heat transfer module that is named HEAT05 and is in a FORTRAN file named NEWHEAT on your D-disk the DEFINE INSTREAM block would have the form:

```
DEFINE INSTREAM
     HEAT05 : NEWHEAT FORTRAN D;
END INSTREAM
```

For MVS/TSO, the DEFINE INSTREAM block might have the form:

```
DEFINE INSTREAM
     HEAT05 : ABCD123.NEWHEAT.FORTRAN;
END INSTREAM
```

3.4.4.3.3 EXTERNALS Keyword

The define externals block contains a list of external inputs to the simulation. The variables must be separated by a comma and the list must end with a semicolon. These are variables which are used but never calculated. The processor requires this information to scan for undefined variables and required balances.

```
DEFINE EXTERNALS
     (Variable name list)
END EXTERNALS
```

For example, to have a tank pressure and enthalpy as inputs to the model the DEFINE EXTERNALS block would have the form:

```
DEFINE EXTERNALS
     PTHTNK,
     HTHTNK;
END EXTERNALS
```

3.4.4.3.4 INTEGRATION Keyword

The integration block allows a change of iteration variables for implicit integration. With predictor-corrector methods, it is not necessary that the state be the iteration parameter. The most common example of this for rocket applications is the choice of iteration variables for thermodynamic states. The engineering states are generally density and internal energy. However, density and internal energy are difficult to provide first guesses for, and more importantly, they are difficult parameters to iterate upon. Better convergence is achieved by iterating on pressure and enthalpy to satisfy the density and internal energy corrector equations. The form of the INTEGRATION block is:

```
DEFINE INTEGRATION
     ITERATE: ( var )   for   ( state ) ;


END INTEGRATION
```

If a change of iteration variables for states is being used, it is specified by the ITERATE keyword, followed by the desired iteration parameter and the state for which it is to be used. If there is no change of iteration variables then nothing needs to be be specified in the DEFINE INTEGRATION block.

An example of an integration block for a volume that models the dynamics for a single constituent fluid and uses a change of iteration variables follows:

```
DEFINE INTEGRATION
     ITERATE : HTVOL1 for UTVOL1   ;
     ITERATE : PTVOL1 for RHOVOL1 ;
END INTEGRATION
```

3.4.4.3.5 BALANCES Keyword

This block is used to define algebraic balances at configuration time. It is normally used for required algebraic balances. however it is often useful to set-up other commonly used balances at configuration time. The format is.

```
DEFINE BALANCES
     balance ( balname ) : ( var )  until ( var ) = ( var ) ;

END BALANCES
```

The functional form has up to an 8 character name to uniquely identify the balance followed by an independent parameter name to be iterated until two dependent variables will be equal. A problem arises when a module output is used as another module's input before the output has been calculated and the parameters have the same name. It would be desirable to simply concatenate a tag. a 'C' for example. on to the end of one of the parameter names. However. for most systems the maximum number of characters a variable can have is eight. all of which could be used when following the ROCETS nomenclature. So a 'C' cannot be concatenated on to the end of the variable name. Therefore the configuration processor assigns a system defined parameter name (of the form SYBL000N) to the second occurrence of the parameter. Once the configuration processor has assigned the system name. a balance can be set up to drive the parameter in question to be equal to the system defined parameter.

For example. suppose you wanted to calculate pressure upstream of a pipe but that pressure has been defined as a state for the volume upstream of the pipe. The configuration processor would tag the pressure upstream of the pipe as requiring an additional balance and would rename one of the pressures with a system defined name. To achieve a balance. the independent parameter WPIPE is varied until the upstream pressure is equal to the volume pressure. A balance can be set up in the following form and the model reconfigured

```
Balance PUPBAL : WPIPE  Until  PUP = SYBL0001 ;
```

If it is desired to establish a balance to drive a dependent variable to a constant value. a name should be assigned to the requested value and then used as the second dependent value. The requested value must also be added to the external input list. An optional method would be to set the balance up at run time instead of configuration time.

3.4.4.3.6 SYSTEM ABOVE, INSIDE, BELOW Keywords

The engineering representation for the simulation is specified in the SYSTEM blocks. Sub-blocks within the SYSTEM blocks are used in specifying engineering modules. thermodynamic properties and equations. The SYSTEM ABOVE keyword directs the various sub-blocks within the SYSTEM ABOVE block to be placed above the iteration loop. Likewise. the SYSTEM INSIDE keyword directs the various sub-blocks within the SYSTEM INSIDE block to be placed inside the iteration loop. Finally. the SYSTEM BELOW keyword directs the various sub-blocks within the SYSTEM BELOW block to be placed below the iteration loop. The form of the SYSTEM blocks is:

```
DEFINE SYSTEM (ABOVE, or INSIDE, or BELOW)
     ( Sub-Block )
          ( entry(s) )

     END ( Sub-Block )

END SYSTEM (ABOVE, or INSIDE, or BELOW)
```

3.4.4.3.6.1 MODULE Sub-Block

The MODULE sub-block is used to specify the engineering module to represent a given component along with necessary information concerning the module. The form of the MODULE sub-block is:

```
MODULE : ( Module Subroutine Name )
     NAME          : ( Component designation ) ;
     I/O LIST      : ( Node Keyword ) = ( name(s) ) , ... ;
     DESIGN VALUES : ( name ) = ( value ) , ... ;
     MAP           : ( name ) ;
     CMT           : ( 65 character message ) ;
END MODULE
```

The module subroutine name appears after the MODULE keyword and specifies the engineering module desired to represent a component of the physical system. The NAME keyword is used to input a 4 character alphanumeric component designation that is specific to the particular engine component.

Nodal connections for the modules are specified by the I/O LIST keyword. Node keywords are part of the interface cards for each module and are used to specify input and output locations.

Design values for the component can be input by the DESIGN VALUES keyword. This will set the default value for component design parameters but they may also be input at run time or can be an iteration parameter.

If the module uses any external maps, this must be entered with the MAP keyword. For readability of the final main program, a 65 character comment can be input with the CMT keyword. The comment will be placed as a comment card prior to the module call.

As an example, consider a multi-node-volume (VL3) for single constituent fluids with two inlet flows from pipe R1 and pipe R2, with corresponding inlet properties from volume VL1 and volume VL2, two exit flows to pipe R3 and pipe R4, with corresponding exit properties from volume VL4 and volume VL5, and one heat flow from COOL. A schematic and the configuration input for this volume follows

```
                                | COOL
 -----------                    |                    -----------
| VL1       |                  \|/                  | VL4       |
|           |       R1      -----------      R3     |           |
|           |------X-------|           |------X-----|           |
 -----------               | VL3       |             -----------
                           |           |
 -----------        R2     |           |     R4      -----------
| VL2       |------X-------|           |------X-----| VL5       |
|           |               -----------             |           |
 -----------                                         -----------
```

```
MODULE: VOLM01
    NAME: VL3
    I/O LIST: INLET FLOW  = R1  , R2  ,
              EXIT FLOW   = R3  , R4  ,
              INLET PROP  = VL1 , VL2 ,
              EXIT PROP   = VL4 , VL5 ,
              QDOT        = COOL;
    DESIGN VALUES: VOL = 500.;
    CMT: VOLUME THREE;
END MODULE
```

3.4.4.3.6.2 PROPERTY Sub-Block

The property sub-block is used to obtain thermodynamic properties as a function of two other thermodynamic properties. The form of the property block is:

```
PROPERTY PACKAGE: ( Package )
    LOCATION ( Node Name ) : (dependent) = f( (indep1, indep2 ) ),

                             (dependent) = f( (indep1, indep2 ));
END PROPERTY
```

The package name appears after the PROPERTY PACKAGE keyword

Within the block, the keyword LOCATION followed by the node name is used to specify at which location the properties are requested. The particular properties are obtained by specifying the dependent property type as well as two independent property types

Currently the packages and corresponding options allowed are:

```
H2PROP - Para-Hydrogen Properties From Maps

  P=F(RHO,H)    - Pressure as a function of density and enthalpy
  RHO=F(P,H)    - Density as a function of pressure and enthalpy
  T=F(P,H)      - Temperature as a function of pressure and enthalpy
  P=F(RHO,U)    - Pressure as a function of density and internal energy
  RHO=F(P,U)    - Density as a function of pressure and internal energy
  T=F(P,U)      - Temperature as a function of pressure and internal energy
  S=F(H,P)      - Entropy as a function of enthalpy and pressure
  H=F(S,P)      - Enthalpy as a function of entropy and pressure
  CP=F(H,P)     - Constant Pressure Specific Heat as a function of
                  enthalpy and pressure
  CV=F(H,P)     - Constant Volume Specific Heat as a function of
                  enthalpy and pressure
  GAMA=F(H,P)   - Gamma as a function of enthalpy and pressure
  MU=F(H,P)     - Viscosity as a function of enthalpy and pressure
  K=F(H,P)      - Thermal Conductivity as a function of enthalpy
                  and pressure

O2PROP - Oxygen Properties From Maps

  P=F(RHO,H)    - Pressure as a function of density and enthalpy
  RHO=F(P,H)    - Density as a function of pressure and enthalpy
  T=F(P,H)      - Temperature as a function of pressure and enthalpy
  P=F(RHO,U)    - Pressure as a function of density and internal energy
  RHO=F(P,U)    - Density as a function of pressure and internal energy
  T=F(P,U)      - Temperature as a function of pressure and internal energy
  S=F(H,P)      - Entropy as a function of enthalpy and pressure
  H=F(S,P)      - Enthalpy as a function of entropy and pressure
  CP=F(H,P)     - Constant Pressure Specific Heat as a function of
                  enthalpy and pressure
  CV=F(H,P)     - Constant Volume Specific Heat as a function of
                  enthalpy and pressure
  GAMA=F(H,P)   - Gamma as a function of enthalpy and pressure
  MU=F(H,P)     - Viscosity as a function of enthalpy and pressure
  K=F(H,P)      - Thermal Conductivity as a function of enthalpy
                  and pressure
```

```
HEPROP - Helium Properties From Maps

   P=F(RHO,H)   - Pressure as a function of density and enthalpy
   RHO=F(P,H)   - Density as a function of pressure and enthalpy
   T=F(P,H)     - Temperature as a function of pressure and enthalpy
   S=F(H,P)     - Entropy as a function of enthalpy and pressure
   H=F(S,P)     - Enthalpy as a function of entropy and pressure

N2PROP - Nitrogen Properties From Maps

   P=F(RHO,H)   - Pressure as a function of density and enthalpy
   RHO=F(P,H)   - Density as a function of pressure and enthalpy
   S=F(H,P)     - Entropy as a function of enthalpy and pressure
   H=F(S,P)     - Enthalpy as a function of entropy and pressure
   T=F(P,H)     - Temperature as a function of pressure and enthalpy

MEPROP - Methane Properties From Maps

   P=F(RHO,H)   - Pressure as a function of density and enthalpy
   RHO=F(P,H)   - Density as a function of pressure and enthalpy
   T=F(H,P)     - Temperature as a function of pressure and enthalpy
   H=F(T,P)     - Enthalpy as a function of temperature and pressure
   S=F(H,P)     - Entropy as a function of enthalpy and pressure
   H=F(S,P)     - Enthalpy as a function of entropy and pressure
   CP=F(T,P)    - Constant Pressure Specific Heat as a function of
                  temperature and pressure
   CV=F(T,P)    - Constant Volume Specific Heat as a function of
                  temperature and pressure

HGPROP - Ideal Gas H2/O2 Combustion Properties From Maps

   CP=F(P,T)    - Constant Pressure Specific Heat as a function of
                  pressure and temperature
   GAMA=F(P,T)  - Gamma as a function of pressure and temperature
   R=F(P,T)     - Gas Constant as a function of pressure and temperature
   RHO=F(P,T)   - Density as a function of pressure and temperature
   K=F(P,T)     - Thermal conductivity as a function of pressure
                  and temperature
   MU=F(P,T)    - Viscosity as a function of pressure and temperature
   Z=F(P,T)     - Compressibility Factor as a function of pressure
                  and temperature
```

Additional inputs for combustion properties are the oxygen fraction (OFR) and helium fraction (HFR). However, these are always required inputs to the HGPROP property package and do not need to be specified by the user within the PROPERTY PACKAGE block.

Examples:

Using H2PROP for hydrogen, obtain density and temperature as a function of pressure and enthalpy at several locations.

```
PROPERTY PACKAGE:  H2PROP
     LOCATION  10    :  RHO = F(PT, HT), TT = F(PT, HT) ;
     LOCATION  PBSF  :  RHO = F(PT, HT), TT = F(PT, HT) ;
     LOCATION  FMCO  :  RHO = F(PT, HT), TT = F(PT, HT) ;
END PROPERTY
```

Using HGPROP, obtain gas constant, specific heat, and specific heat ratio Note that the oxygen and helium fractions do not have to be specified since they are always required and can therefore be included in the call list automatically by the processor.

```
PROPERTY PACKAGE: HGPROP
     LOCATION  OPRB : RGAS = F(PT, TT), CP = F(PT , TT),
                      GAMA = F(PT , TT);
     LOCATION  FPRB : RGAS = F(PT, TT), CP = F(PT , TT),
                      GAMA = F(PT , TT);
END PROPERTY
```

A property variable for the executable code is created from the the property type specified for the given location, concatenated with the given location. For the H2PROP example the following variables would be created for LOCATION PBSF: PTPBSF, HTPBSF, RHOPBSF, TTPBSF.

3.4.4.3.6.3 EQUATION Sub-Block

The equation sub-block is used to enter FORTRAN equations into the simulation. The format is:

```
EQUATION : { Fortran Equation } ;
```

Most standard FORTRAN mathematical symbols and intrinsic functions are allowed. Note also that the equation input may be continued on up to one subsequent line and is closed by a semicolon.

An example of the use of the EQUATION sub-block follows:

```
EQUATION : RHO = PT / RGAS / TT ;
```

3.4.4.4 Run Input

Run input consists of user commands to execute a configured simulation. It consists of required information to set inputs, define algebraic loops, specify output, and control execution. The following set of keywords accomplish this:

1. SCHEDULES
2. INPUTS
3. INTEGRATION DEFAULTS
4. INTEGRATION EXCEPTIONS
5. BALANCES
6. BALANCE DEFAULTS
7. BALANCE EXCEPTIONS
8. LINEARIZATION
9. LINEARIZATION DEFAULTS
10. LINEARIZATION EXCEPTIONS
11. RESTART
12. OUTPUT
13. RUN

If a line within the run input is to be ignored, this can be accomplished by placing an asterisk (*) in column one.

Debug output for the run input will be generated when then following line is located on the first line starting in the first column of the run input file.

```
*<DEBUG>
```

The blocks within the run input are processed as they are encountered, thus the order in which the blocks are arranged is important.

3.4.4.4.1 SCHEDULES Keyword

The define schedules block is used to input univariant or bivariant curves representing a functional relationship for a model input. For steady-state schedules a system counter named POINT counts points for reading schedules and TIME is available for reading transient schedules.

Schedule dependent parameters can be single precision real or integers but the table itself is single precision real. Schedule independent parameters must be POINT, TIME, an external input, a state, or an independent balance parameter. A model output cannot be used since this would require an implied algebraic loop.

The SCHEDULE block consists of two parts. The first is the schedule definition and the second is the loading of data into the schedule. The format is:

```
DEFINE SCHEDULES
     Schedule      : { name } is { dep } = F( { ind } ) ;
     Set schedule  : { name } = { data } ;
END SCHEDULES
```

Schedules have a unique eight character name identifier. The functional relation can be either univariant or bivariant. If the schedule is bivariant, two independent parameters are required separated by a comma. The schedule data are in standard map-reader format. The first two numbers of the schedule data should be set to zero. They are used as pointer storage locations by the table reader. The third and fourth numbers of the schedule data indicate the number of data points contained in the schedule for the two independent parameters. If the schedule is univariant, the fourth number must be set to zero. If extrapolation of the schedule is desired, the third and/or fourth (if bivariant) schedule data points should be made negative. The rest of the schedule data consist of a list of data separated by commas for the first independent parameter in ascending order, followed by a list of data separated by commas for the second independent parameter in ascending order for a bivariant schedule, followed by a list of data separated by commas for the dependent parameter. The dependent parameter data is arranged with the dependent data corresponding to the first data point for the first independent parameter and all of the corresponding second independent parameter data points, followed by the second data point for the first independent parameter and all of the corresponding second independent parameter data points and so on.

Example: Set up a schedule to vary tank pressure (PTANK) from 200 to 100 over 10 seconds, represented as a linear variation with a two point curve.

```
DEFINE SCHEDULES
     Schedule : htnkpres is PTANK = F(TIME) ;
     Set schedule: htnkpres =   0.,    0.,     2.,     0.,
                                0.,    10.,
                                200.,  100.;
END SCHEDULES
```

If extrapolation is desired the schedule should be set up as follows:

```
DEFINE SCHEDULES
     Schedule : htnkpres is PTANK = F(TIME) ;
     Set schedule: htnkpres =   0.,    0.,    -2.,     0.,
                                0.,    10.,
                                200.,  100.;
END SCHEDULES
```

Example: Set up a schedule of drag coefficient (DRAG) as a function of altitude (ALT) and vehicle mach number (VM) with a bivariant schedule.

```
DEFINE SCHEDULES
     Schedule : vdrag is DRAG = F(ALT,VM) ;
     Set schedule: vdrag     =   0.,    0.,        3.,        3.,
                                        0.,   10000.,   100000.,
                                        0.,       1.,        5.,
                                        0.,       .7,       1.4,
                                        0.,       .5,       1.2,
                                        0.,       .3,       1.0;
END SCHEDULES
```

3.4.4.4.2 INPUT Keyword

The define input block is used to define model input values for a particular run. Generally the inputs will have been previously defined as external inputs during configuration. However, there are no

ORIGINAL PAGE IS
OF POOR QUALITY

restrictions on what may be actually input. Inputs from a schedule are specified by entering the keyword SCHEDULE and the schedule name. The format is:

```
DEFINE INPUT
     ( varname ) = ( data ) ,
     ( varname ) = schedule ( schedule name ) ;
END INPUT
```

The following example shows how to define inputs from a schedule and various model input values for a particular run:

```
DEFINE INPUT
     PTIN = schedule PIPEABC ,
     CF   =  95.0 ,
     RPL  =  65.0 ;
END INPUT
```

3.4.4.4.3 INTEGRATION DEFAULTS Keyword

The integration defaults block sets-up default integration information at run time. It is generally easier to set-up default information which is adequate for most states and then to override the defaults for specific states when necessary. The form is:

```
DEFINE INTEGRATION DEFAULTS
     Method                  :  ( keyword ) ;
     Convergence Criteria    :  ( keyword ) ;
     Tolerance               :  ( value ) ;
     Perturbation            :  ( value ) ;
     Allowed Change          :  ( value ) ;
     State Bias              :  ( value ) ;
     State Normalizer        :  ( value ) ;
END INTEGRATION DEFAULTS
```

The method keywords allow user selection of the integration technique within the limits allowed by the integration method selected at configuration. Currently there are no limits, but as new integration routines are added limits will be necessary. Current methods are: EULER, TRAPEZOIDAL, GEAR 1ST and GEAR 2ND WARNING: If the balances are on or the state iteration variables are active when using the Euler method an error will occur.

For predictor-corrector schemes, the corrector equation is iterated to convergence using a multi-variable Newton-Raphson method The multi-variable iteration routine includes internal Jacobian scaling to improve convergence with stiff systems. In effect both the rows and columns are normalized by the maximum element in the row and column The error term used for convergence testing can be the actual error or it can be normalized as part as part of the Jacobian conditioning

Convergence criteria, tolerance, perturbation, and allowed change apply only to iterative methods. The keywords to specify the convergence criteria are: ABSOLUTE ERROR or NORMALIZED ERROR.

Tolerance is defined as how close to zero the error term must be before solution is considered converged Experimentation to determine a good tolerance value is usually necessary.

The perturbation is the amount each independent variable is moved for generating a Jacobian. It is specified as a percentage of the biased state, input as a decimal fraction. The allowed change is the amount an independent variable is allowed to change each pass. This is necessary in non-linear systems to prevent excessive movement leading to exceeding map bounds, etc. The allowed change is also a percentage of the biased state, input as a decimal fraction.

The state bias is the value that is to be added to the state to bias the state. This is necessary if the state is going to change sign or approach zero during the run. The state normalizer is the value that the state is to be divided by to normalize the state for the first point. If the state normalizer is set to zero, the state is normalized by the initial state guess plus the state bias for the first point. For all successive points, the state is normalized by the previous converged value of the biased state.

The defaults for the various keywords and values are: Method defaults to TRAPEZOIDAL, Convergence Criteria defaults to ABSOLUTE ERROR, Tolerance defaults to 0.001, Perturbation defaults to 0.01, Allowed Change defaults to 0.1, State Bias defaults to 0.0, and State Normalizer defaults to 0.0.

3.4.4.4.4 INTEGRATION EXCEPTIONS Keyword

This block defines exceptions to the default integration set-up. With a large number of states, it is convenient to set-up defaults which will take care of most states and override the defaults for specific states. The form and allowed items are:

```
DEFINE INTEGRATION EXCEPTIONS
    Activation for              ( state ) :  ( on, off, steady-state ) ;
    Convergence Criteria for    ( state ) :  ( keyword ) ;
    Tolerance for               ( state ) :  ( value ) ;
    State Bias for              ( state ) :  ( value ) ;
    Independent Bias for        ( state ) :  ( value ) ;
    Perturbation for            ( state ) :  ( value ) ;
    Allowed Change for          ( state ) :  ( value ) ;
    State Normalizer for        ( state ) :  ( value ) ;
    Independent Normalizer for ( state ) :  ( value ) ;
END INTEGRATION EXCEPTIONS
```

The default for all states is to be active always. However, it is often convenient to turn states off at various times. The activation keyword has options for when the state is active:

steady-state	=	the state is always iterated to steady-state, thereby removing the dynamic effect of the state
on	=	the state is active
off	=	the state is inactive and held constant

The independent bias and the independent normalizer refer to the state iteration variables defined in the DEFINE INTEGRATION block. All of the other items have already been discussed.

3.4.4.4.5 BALANCES Keyword

This block is used to define algebraic balances at run time. The independent variable can be a model input or a module design parameter. The dependent variable can be either model output, a state, or a constant. The maximum number of balances that can be defined at run time is ten. The form is:

```
DEFINE BALANCES
    balance ( balname ) : ( var ) until ( var ) = ( var ) ;

END BALANCES
```

For example, suppose you wanted to balance the flow exiting a pump, WPUMP, to a flow that is calculated downstream of the pump, WCALC, by varying the speed of the pump SNPUMP.

```
DEFINE BALANCES
    BALANCE WBAL : SNPUMP until WPUMP = WCALC ;
END BALANCES
```

The DEFINE BALANCES block can also be used to allow a parameter to be one DT out of phase. This is accomplished by setting up a balance in the form: vary X until X = XC and then turning the balance off in the DEFINE BALANCE EXCEPTIONS block. This allows X to be used during the convergence attempt and then be updated to XC after convergence has been achieved. If you want to turn the balance off and not run one DT out of phase, the balance should be turned off and the balance description should be rearranged in the form: vary X until XC = X.

3.4.4.4.6 BALANCE DEFAULTS Keyword

The define balance defaults block is similar to the define integration defaults section. It is used to define parameters for configuration or run time defined balances.

```
DEFINE BALANCE DEFAULTS
        Convergence Criteria    : { keyword } ;
        Dependent Normalizer    : { value } ;
        Independent Normalizer  : { value } ;
        Tolerance               : { value } ;
        Bias                    : { value } ;
        Perturbation            : { value } ;
        Allowed Change          : { value } ;
END BALANCE DEFAULTS
```

The value of the normalizers are set according to the following tables

For the first point in a transient run or a steady-state point:

Condition	Normalizer
Dependent Normalizer Set to 0.	Initial guess for the 2nd Dependent Variable
Dependent Normalizer Set to 0. and 2nd Dependent Variable = 0.	Initial guess for the Independent Variable plus the bias
Independent Normalizer Set to 0.	Initial guess for the Independent Variable plus the bias

For the successive points of a transient run:

Condition	Normalizer
Dependent Normalizer Set to 0. or a value	Previous converged value of the 2nd Dependent Variable
Dependent Normalizer Set to 0. or a value and 2nd Dependent Variable = 0.	Previous converged value of the Independent Variable plus the bias
Independent Normalizer Set to 0. or a value	Previous converged value of the Independent Variable plus the bias

The defaults for the various keywords and values are. Convergence Criteria defaults to ABSOLUTE ERROR, Dependent Normalizer defaults to 0.0, Independent Normalizer defaults to 0.0, Tolerance defaults to 0.0001, Bias defaults to 0.0, Perturbation defaults to 0.001, and Allowed Change defaults to 0.1.

3.4.4.4.7 BALANCE EXCEPTIONS Keyword

The define balance exceptions block is similar to the define integration exceptions section. It is used to define exceptions to the defined balance defaults for configuration or run time defined balances. The default for all balances is to be active always.

```
DEFINE BALANCE EXCEPTIONS
    Activation for              { balname } :   { on, off } ;
    Convergence Criteria for    { balname } :   { keyword } ;
    Tolerance  for              { balname } :   { value } ;
    Bias for                    { balname } :   { value } ;
    Perturbation for            { balname } :   { value } ;
    Allowed Change for          { balname } :   { value } ;
    Dependent Normalizer for    { balname } :   { value } ;
    Independent Normalizer for  { balname } :   { value } ;
END BALANCE EXCEPTIONS
```

3.4.4.4.8 LINEARIZATION Keyword

This block is used to define linearization options and set-up at run time. Note that when linearizing, the states are controlled through the INTEGRATION DEFAULTS and EXCEPTIONS blocks and the balances are controlled through the BALANCE DEFAULTS and EXCEPTIONS blocks.

```
DEFINE LINEARIZATION
    REPEATABILITY CHECK   : { value } ;
    LINEARITY CHECK       : { value } ;
    INPUTS                : { variable list } ;
    OUTPUTS               : { variable list } ;
END LINEARIZATION
```

Typically, the base point (steady-state or transient) about which linearization is desired is run first and then the linearization is performed Multiple linearization runs about different base points can be accomplished by setting up multiple pairs of base point and linearization runs in series.

When linearizing, states and balances which are not turned off, and variables which are defined as inputs for linearization are perturbed from the selected base point. When one variable is being perturbed the other variables which are to be perturbed are held constant. The perturbation procedure is to first make a positive perturbation, then make a negative perturbation, then repeat the positive perturbation. To perform the repeatability check, the two positive perturbations are used to calculate a percent difference which is compared to the repeatability check value. If the percent difference is greater than the repeatability check value then a message is printed that describes the nonrepeatability. To perform the linearity check, the positive and negative partials are used to calculate an average partial, then a percent difference for the partials is calculated which is compared to the linearity check value. If the percent difference is greater than the linearity check value then a message is printed that describes the nonlinearity.

INPUTS are variables that are to be inputs to the linear model (maximum of 15)

OUTPUTS are non-state variables that are to be outputs from the linear model (maximum of 20).

The defaults for the values are: REPEATABILITY CHECK defaults to 0.01, and LINEARITY CHECK defaults to 0.1.

3.4.4.4.9 LINEARIZATION DEFAULTS Keyword

The define linearization defaults block is similar to the define integration defaults section. It is used to define values that control the perturbation size and bias of the linear model INPUTS.

```
DEFINE LINEARIZATION DEFAULTS
    Perturbation : { value } ;
    Bias         : { value } ;
END LINEARIZATION DEFAULTS
```

The defaults for the values are: Perturbation defaults to 0.01, and Bias defaults to 0.0.

3.4.4.4.10 LINEARIZATION EXCEPTIONS Keyword

The define linearization exceptions block is similar to the define integration exceptions section. It is used to define any exceptions to the LINEARIZATION DEFAULTS values for perturbation and bias of the linear model INPUTS.

```
DEFINE LINEARIZATION EXCEPTIONS
     Perturbation for { var } : { value } ;
     Bias for         { var } : { value } ;

END LINEARIZATION EXCEPTIONS
```

The following example shows a typical set-up for the linearization block:

```
DEFINE LINEARIZATION
     REPEATABILITY CHECK   : 0.1 ;
     LINEARITY CHECK       : 0.1 ;
     INPUTS: AREAFPOV, AREAOPOV ;
     OUTPUTS: FG ;
END LINEARIZATION

DEFINE LINEARIZATION DEFAULTS
     PERTURBATION : 0.001 ;
     BIAS         :   0.0 ;
END LINEARIZATION DEFAULTS

DEFINE LINEARIZATION EXCEPTIONS
     PERTURBATION for AREAOPOV : 0.0005 ;
     BIAS for AREAOPOV         : 0.1    ;
END LINEARIZATION EXCEPTIONS
```

3.4.4.4.11 RESTART Keyword

The RESTART block is used to specify information necessary for either restarting a simulation from a previously saved balanced point, or for specifying the time and successive time increment at which a restart file is to be written during a run, or for both restarting and writing restart files. It is also used to specify a value that is to be passed into the GUESS routine and to determine if the GUESS routine is to be called. (A blank GUESS routine is generated by the configuration processor and must be completed by the user). The form is:

```
DEFINE RESTART
     INPUT FILE      :   { file designation } ;
     OUTPUT FILE     :   { file designation } ;
     BEGIN TIME      :   { time } ;
     DT              :   { delta time } ;
     GUESS           :   { guess value } ;
END RESTART
```

The INPUT FILE file designation allows the user to specify the file from which the restart information is to be read. Likewise, the OUTPUT FILE file designation allows the user to specify the file to which the restart information is to be written. For CMS users the file designation consists of a file name, a file type and a file mode, while for MVS/TSO users the file designation is the complete file name. NOTE: The first restart file that is to be written will not write over an existing file of the same name. If this is attempted, execution will be halted by the run time reader and a warning message will be issued.

The location of the RESTART block within the run input file can affect the set-up of the run. If the RESTART block is located at the top of the run input file, then the remaining blocks can change the set-up of the run However, if the RESTART block is located somewhere else within the run input file, then the run input specified in the preceding blocks could be overridden

The BEGIN TIME time specifies the time at which the first restart file is to be written. The DT delta time specifies the time increment at the end of which a restart file is to be written. Restart files will continue to be written over the previously outputted restart file until the length of the run has been completed. If only one restart file is desired, the sum of the BEGIN TIME and the DT must be greater than the length of the run.

The guess value is a R*4 value that will be passed into the GUESS routine. The user can then utilize this value to access different sets of guess data within the guess routine. If a guess value is not entered for GUESS, then the GUESS routine will not be called.

The defaults for the various keywords and values are: BEGIN TIME defaults to 99 999, DT defaults to 0.0, and the default is for the GUESS routine not to be called.

3.4.4.4.12 OUTPUT Keyword

The output block is used to specify output desired from a simulation run. The format is:

```
DEFINE OUTPUT
    TRANSIENT PRINT        : { on/off } ;
    LINEARIZATION PRINT    : { on/off } ;
    STEADY-STATE PRINT     : { on/off } ;
    PRINT : { option } , { print parameter list } ;
    PLOT  : { option } , { plot  parameter list } ;
    PLOT FILE  : { file designation } ;
    PLOT TITLE : { 48 character title } ;
    ERROR HANDLING for { modname modloc } :
         PRINT LEVEL   = { val } ,
         DIELEVEL      = { val } ,
         DIECOUNT      = { val } ,
END OUTPUT
```

If the TRANSIENT PRINT is on and convergence is not achieved, the last pass of the convergence attempt is output. If the TRANSIENT PRINT is off, no convergence information is output. If the STEADY-STATE PRINT is on a full print of both the Jacobian evaluation and each convergence attempt is provided. If the STEADY-STATE PRINT is off, a short message is printed that specifies if convergence was achieved and how many passes were made.

If the linearization print is on, exceptions to the linearity check and/or repeatability check are printed

A variety of options control print and plot output for the PRINT and PLOT keywords. The options are:

```
NOPRINT  - No print output
NOPLOT   - No plot output
DUMPALL  - Output all occurrences of all parameters
DUMPONCE - Output the first occurrences of all parameters
ALL      - Output all occurrences of specified parameters
ONCE     - Output first occurrences of specified parameters
OMITALL  - Omit all occurrences of specified parameters and output all
           others
OMITONCE - Omit all occurrences of specified parameters and output
           the first occurrence of all others
```

If specified parameters are necessary, the parameter list follows the option keyword.

The defaults for the various on/off flags and options are: TRANSIENT PRINT defaults to on, LINEARIZATION PRINT defaults to on, STEADY-STATE PRINT defaults to off. PRINT defaults to DUMPALL, and PLOT defaults to NOPLOT.

The plot file keyword is used to specify the file designation for the file that is to contain the plot data. For CMS users the file designation consists of a file name, a file type and a file mode, while for MVS/TSO users the file designation is the complete file name.

The ERROR HANDLING keyword allows selection of a print error level and an error level and count at which to stop execution.

The following example will print out the first occurrence of the specified parameters and produce plot output of the first occurrence of all the parameters for a transient run using the CMS file designation.

```
DEFINE OUTPUT
    TRANSIENT PRINT : ON;
    PRINT      =  ONCE    , TIME, PTMCHB, SNOH, SNFH, PTOPRB, PTFPRB ;
    PLOT       =  DUMPONCE ;
    PLOT FILE  = CAPAFILE BINARY D;
    PLOT TITLE = TRANSIENT RUN NUMBER ONE;
DEFINE OUTPUT
```

The following example will print out all occurrences of all parameters and and produce plot output of the first occurrence of FG for a steady-state run with error handling designations using the MVS/TSO file designation.

```
DEFINE OUTPUT
    STEADY STATE PRINT : ON;
    PRINT      =  DUMPALL;
    PLOT       =  ONCE, POINT, FG;
    PLOT FILE  = ABCD123.CAPAFILE.BINARY;
    PLOT TITLE = STEADY STATE BALANCE;
    ERROR HANDLING for COMB02 MCHB 1ST;
          PRINT LEVEL = 3000,
          DIELEVEL    = 10000,
          DIECOUNT    = 1,
END OUTPUT
```

3.4.4.4.13 RUN Keyword

The run block is used to define necessary simulation control inputs for a particular simulation run. The syntax is :

```
DEFINE RUN
    ( STEADY STATE, TRANSIENT, or  LINEARIZE ) : ( options )  ;
END RUN
```

The options for STEADY-STATE are:

```
POINTS  =  ( value )
MAXPASS =  ( value )
```

For POINTS, value is the number of consecutive points to be run. A system variable POINT will be set to one and incremented by one on each steady-state balance for use in schedules.

For MAXPASS, value is the maximum number of iteration passes that will be made before a convergence attempt is halted.

The default value for POINTS is 1 and the default value for MAXPASS is 50.

The following is an example of a steady-state run block:

```
DEFINE RUN
    STEADY STATE : POINTS = 3, MAXPASS = 35;
END RUN
```

The following is a list of the TRANSIENT options:

```
STOP TIME    -  ending time for transient operation
DT           -  transient time increment
PRINT TIME   -  time increment for print
PLOT TIME    -  time increment for plot
MAXPASS      -  maximum number of convergence passes
```

Note that if the PLOT TIME is an even multiple of the model DT, it will be changed to an odd multiple to avoid masking numerical instabilities.

The default value for DT is 0.0001 and the default value for MAXPASS is 20.

If it is required to start the POINT count or TIME at some value other than one or zero respectively, this can be accomplished by setting POINT or TIME to the desired value in the INPUT block.

The following is an example of a transient run block:

```
DEFINE RUN
    TRANSIENT : DT = .001, STOP TIME = .01,
          PRINT TIME .001, PLOT TIME = .001, MAXPASS =  50 ;
END RUN
```

Currently there are no options for the LINEARIZE keyword.

The following is an example of a linearize run block:

```
DEFINE RUN
    LINEARIZE :  ;
END RUN
```

3.4.4.5 Building a Module

A FORTRAN subroutine can easily be converted to the ROCETS system. The following sections must be included in the system module.

1. Subroutine call list
2. Interface section
3. History of the module including author and dated list of revisions
4. Listing of all subroutines and commons required by the module

Information required to interface a module into the ROCETS system will be contained in comment cards within the prologue of the module. The interface section will be in three parts

1. Interface information
   ```
   *BEGIN INTERFACE ( Module Name )
   *END INTERFACE   ( Module Name)
   ```
2. Keyword information
   ```
   *BEGIN KEYWORDS ( Module Name)
   *END KEYWORDS   ( Module Name )
   ```
3. Units information
   ```
   *BEGIN UNITS  ( Module Name )
   *END UNITS ( Module Name)
   ```

3.4.4.5.1 Module Communication

Modules may only communicate to the ROCETS system through the subroutine call list of the module. Commons cannot be used to communicate with the main or other modules. However, common blocks may be used in certain cases for communication between a module and a sub-module.

3.4.4.5.2 Interface Data Section

The interface data section of the module allows the configuration processor to create the communication link with the ROCETS system. Specific standards for the interface section of modules follow:

*** INTERFACE BLOCK ***

The interface block relates call list names to system names, defines the status of each variable for system operation, defines the I/O status of each variable, and the FORTRAN variable type. The set-up consists of 6 pieces of information for each variable:

1. Call list name
2. System name

3. System tag
4. Array status
5. I/O status
6. FORTRAN variable type.

A sample interface block is shown below.

xBEGIN INTERFACE VOLMXX

CALL LIST NAME	SYSTEM NAME	SYSTEM TAG	ARRAY STATUS	I/O STATUS	VAR TYPE
IPRPL	IPRPL	GLOBAL		IN 0	I*4
IUPDAT	IUPDAT	GLOBAL		IN 0	I*4
MODN	MODN	NAME 0		IN 0	C*4
NOD1	NOD1	NAME 1		IN 0	C*4
NOD2	NOD2	NAME 2		IN 0	C*4
NOD3	NOD3	NAME 3		IN 0	C*4
NOD4	NOD4	NAME 4		IN 0	C*4
NOD5	NOD5	NAME 5		IN 0	C*4
VOL	VOL	DESIGN		IN 0	R*4
HTIN	HT	VARIABLE		IN 1	R*4
WIN	W	VARIABLE		IN 2	R*4
WOUT	W	VARIABLE		IN 3	R*4
HTOUT	HT	VARIABLE		IN 4	R*4
QDOT	QDOT	VARIABLE		IN 5	R*4
RHOVOL	RHO	STATE		IN 0	R*4
UTVOL	UT	STATE		IN 0	R*4
HTVOL	HT	VARIABLE		IN 0	R*4
DRDT	DRDT	DERIVATIVE		OUT 0	R*4
DUDT	DUDT	DERIVATIVE		OUT 0	R*4
TAUCR	TAU1	TAUC		OUT 0	R*4
TAUCU	TAU2	TAUC		OUT 0	R*4

xEND INTERFACE VOLMXX

The first entry is the name in the module call list. It should follow ROCETS naming convention, but this is not required for proper operation within the system. The call list must include as input 4 character user defined names for the module name and any associated nodes.

The second entry is the system name. Unlike the call list name, a consistent naming convention must be used for proper operation in the system. The actual variable name will be constructed at configuration time by concatenating the system name with the proper module/node name. Unless consistent nomenclature is used, the constructed variable name will not match the names constructed for other modules. For parameters that are arrays (except for nodes), the first four characters of the call list name of the array must be different than the system name. This is required to avoid duplicate names for arrays that have the same system name. The third entry is the system tag. This informs the configuration processor what the parameter is used for so proper action can be taken. The current system tags are:

<u>GLOBAL</u>

A constructed name will not be generated -- the call list name will be the actual name. This is used for standard flags which are the same for all modules. At present, the flag IPRPL used for enabling print and the flag IUPDAT used for initialization are global flags.

<u>NAME n</u>

The keyword name followed by an integer value is used for module/node names. The integer value specifies the particular node for concatenating names. In all cases the integer value 0 should be used for the module name. If the module has configuration dependent arrays, the node name will also be an array.

DESIGN

Specifies that the variable is a design value for the module. At present no special action is taken with the DESIGN keyword (ie., the keyword VARIABLE will also work), but it is included as a special tag for future enhancements.

VARIABLE

Specifies that the parameter is an input/output and has no special significance to the system.

STATE

Specifies that the parameter is a state variable.

DERIVATIVE

Specifies that the parameter is a state derivative. States and derivatives hold special significance in that pointers must be constructed to locate the states and derivatives within the global commons. NOTE: it is required that when a module has multiple states and derivatives, they must be ordered in the call list. That is, the first derivative must match the first state, and so on. Also, any module that calculates a derivative must have the state in the call list even if the state is not required.

DSC

Specifies that the parameter is a discrete flag. Discrete flags are used to "freeze" operation about a discontinuity.

DSCR

Specifies that the parameter is a discrete flag request. The request is used to inform the system on which side of a discontinuity the module should be operating. Some action will be required when the discrete and discrete request are different after a converged point.

TAUC

Specifies that the parameter is a critical time associated with a state. As with derivatives, if a module has multiple states, the number of critical times must equal the number of states and be ordered in the call list. However, it is not required that a module output critical times. The configuration processor will assign default values for any states for which the critical time has not been defined.

MAP

Specifies that the parameter is the external name of a map subroutine.

The fourth entry is array status. For non-array parameters this field is left blank. For parameters that are arrays, the word ARRAY is entered followed by either an asterisk or an integer number. An asterisk specifies that the array size is configuration dependent and the configuration processor will count the number of elements in the array. Additionally, for configuration dependent arrays, the processor will put the number of elements in the first location and dimension the array to the number of elements plus one. An integer number specifies that the array is not configuration dependent and the processor takes no special action other than to dimension the array to the specified value.

The fifth entry is the I/O status. Each parameter must be tagged as either an input (IN), output (OUT), input/output (I/O), or output/input (O/I). In addition, each parameter must include an integer number corresponding to the named node with which it is associated.

The final entry is the FORTRAN variable type. The types are specified by:

R*4 = Single precision real variable
I*4 = Integer variable
R*8 = Double precision real variable
X*8 = Single precision complex variable
C*4 = 4 byte character. Longer character strings must be treated as C*4 arrays.

*** KEYWORD BLOCK ***

Keywords may be defined for inputting node information to the configuration processor and are required for all configuration dependent arrays. This allows the user to use engineering terms instead of system terms when specifying the inputs/outputs for a module. Keywords can be specified only for node names.

Keywords are contained on comment cards within a %BEGIN KEYWORDS / %END KEYWORDS block. The format of the keywords is:

```
{ call list name for node } : { keyword list } ;
```

The keyword list may be one or more keyword phrases separated by commas. A semicolon is used to terminate the list. The list for any node may be on more than one line. A sample keyword block is shown below:

```
%BEGIN KEYWORDS VOLMXX

    NOD1    : UPSTREAM PROP, INLET PROP                              ;
    NOD2    : UPSTREAM FLOW, WIN, ENTERING FLOW, INLET FLOW          ;
    NOD3    : DOWNSTREAM FLOW, WOUT, EXIT FLOW                       ;
    NOD4    : DOWNSTREAM PROP, EXIT PROP                             ;
    NOD5    : QDOT                                                   ;

%END KEYWORDS VOLMXX
```

A standard set of keywords is necessary to avoid confusion and promote consistency and readability for configuration input. A preliminary set has been defined and as the ROCETS system begins to be used, user comments on appropriate keywords will be used to refine the keyword list. A preliminary set of standard keywords follows:

Properties:

- Inlet Prop, Upstream Prop
- Exit Prop, Downstream Prop
- Fuel Prop, Oxidizer Prop, Helium Prop

Flows:

- Inlet Flow, Upstream Flow
- Exit Flow, Downstream Flow

Heat Transfer:

- Qdot
- Metal Temp, Tmetal

Shafts:

- Torq
- Shaft, Rotor

*** UNITS BLOCK ***

Call list parameter units are contained in a %BEGIN UNITS / %END UNITS block. Both English and SI units are required. Twelve characters are allotted for each set of units. A sample units block is shown below:

%BEGIN UNITS VOLMXX

CALL LIST NAME	ENGLISH	SI
IPRPL	D'LESS	D'LESS
IUPDAT	D'LESS	D'LESS
MODN	D'LESS	D'LESS
NOD1	D'LESS	D'LESS
NOD2	D'LESS	D'LESS
NOD3	D'LESS	D'LESS
NOD4	D'LESS	D'LESS
NOD5	D'LESS	D'LESS
WI	LBM/S	KG/S
WO	LBM/S	KG/S
HTI	BTU/LBM	J/KG
HTO	BTU/LBM	J/KG
VOL	IN**3	M**3
QDT1	BTU/S	J/S
HTVOL	BTU/LBM	J/KG
RHOVOL	LBM/IN**3	KG/M**3
UTVOL	BTU/LBM	J/KG
DRDT	LBM/IN**3/S	KG/M**3/S
DUDT	BTU/LBM/S	J/KG/S
TAUCR	S	S
TAUCU	S	S

%END UNITS VOLMXX

3.4.4.5.3 Module Print/Plot Output

Print and plot output is handled by the utility module PRPL01. Parameter output is controlled through user requests in the run (execution) input file but is actually output from within modules and sub-modules. The structure of a PRPL01 call within a routine is

```
CALL PRPL01 ( ( n ) , ( outname ) , ( paramunit ) , ( paramname ) )
```

Where n is the number of parameters to be output for this call. If you are outputting an array, this is the number of elements of the array that you want to output. Outname is the eight character name that will represent the parameter in the print and plot output. Parmunit is the 12 character string of the units of the output parameter, however, parmunit is left blank in the PRPL01 calls that are within a module or sub-module. The proper units are passed to PRPL01 by the system and then output. Parmname is the parameter real type variable name as it appears in the module or sub-module. This call will be made every time there is a request for plot or print output. A detailed description of PRPL01 usage can be found in the SDS Section 3.2.3.

3.4.4.6 Simulation Debug

FORTRAN subroutines ERCK00, ERCK01, ERCK02 are called each time there is the possibility for an error in modules and sub-modules. This run- time error checking aids the user in pinpointing possible fatal errors and debugging them. ERCK00 is called from the routines, and then in turn calls ERCK01 and ERCK02. The call list for ERCK00 is as follows:

```
CALL ERCK00 ( IUPDAT, MODNAM, MODLOC, IERCODE, MSG )

Where:
          IUPDAT   =  Update flag (I*4)
          MODNAM   =  Calling program name (C*8)
          MODLOC   =  Location in calling program (C*8)
          IERCODE  =  Error level code (I*4)
          MSG      =  Error message (C*50)
```

3.4.4.6.1 Error flags

The error code and MODLOC allow for both multiple error checks per module and multiple error levels. The error code currently is set between 0 and 10000, with 0 being no error and 10000 being the fatal error kill level. The error codes are saved in arrays for print/kill checking. A list of error codes and their corresponding errors follows:

```
 0000              No error
 1000 - 2999       Map Extrapolation
 3000 - 4999       Input out of Range
 5000 - 6999       Internal Iteration Failures
 7000 - 9999       Invalid Solution
10000              Invalid Option (No Default),
                   execution halted immediately by ERCK00
```

Note that if an error level of 9000 to 9999 is encountered, execution is halted after the current pass is completed

For multiple error calls within a module, MODLOC must be unique for each call. This can be accomplished by using the four character module name concatenated with a string that denotes the order of error occurrence within the module, 1st, 2nd, 3rd ...., for example.

The error print level may be set by the user at run-time. This allows the suppression of lower level errors that may have little or no effect upon the overall solution. By using the error checking routine wisely the user can ensure the model is fully debugged.

3.4.4.7 Running a Model

Both the Configuration Processor and the resultant configured model execute in the MVS batch environment. A clist, ROCETS, contained in the ROCETS.CLIST library is provided to generate the JCL submittal dataset necessary to run either the Configuration Processor or a previously configured model. The name of this dataset is "prefix.ROCETS.TEMP.JCL".

3.4.4.7.1 Running the Configuration Processor (TSO)

The ROCETS clist requires no arguments, all input is obtained through prompts. No validation is done on input items. A misspelled dataset name will cause subsequent job failure. ROCETS first tries to obtain the TSO logon account number to be inserted in the job card. If a valid account number cannot be obtained then ROCETS will prompt for it. The following is a sample dialog ROCETS execution to run the configuration processor.

1. DO YOU WANT TO ROUTE OUTPUT TO A DIFFERENT NODE (YES/NO)? = = = >

   Answer NO and prompting will proceed to the next topic. Output will be held in the output queue where it can be viewed via the IOF option of SPF. Answer YES and ROCETS prompts for an alternate node and userid.

   ENTER DESTINATION NODE, (IE. PWAGPDH.E092928) = = = >

2. DO YOU WANT TO RECONFIGURE (YES/NO)? = = = >

   Answer YES to run the Configuration Processor.

3. ENTER DATASET CONTAINING CONFIG INPUT, NO QUOTES = = = >

   Enter here the complete TSO dataset name containing the configuration input without quotes. For example, ROCETS.DATA(CTTBE001).

4. ENTER DATASET FOR CONFIG FORTRAN OUTPUT, NO QUOTES = = = >

   Enter here the complete TSO data name for the configured model output. For example userid.MYMODEL.FORTRAN.

At this time the following message is displayed and the job is submitted to the MVS batch machine for execution.

FILE userid.ROCETS.TEMP.JCL CONTAINS SUBMITTED JCL.

When the job finishes the dataset specified above for configuration output will contain the configured fortran model. In addition a dataset, userid.GUESS.FORTRAN, will contain skeleton fortran for initial guesses. This dataset must be completed and merged with the model fortran prior running the model. A sample guess dataset, ROCETS.FORTRAN(GTTBE001) is provided. If a guess routine is not appended to the model fortran dataset the library copy mentioned above will be used.

3.4.4.7.2 Running a configured model (TSO)

The ROCETS clist requires no arguments, all input is obtained through prompts. No validation is done on input items. A misspelled dataset name will cause subsequent job failure. ROCETS first tries to obtain the TSO logon account number to be inserted in the job card. If a valid account number cannot be obtained then ROCETS will prompt for it. The following is a sample dialog ROCETS execution to run a previously configured model.

1. DO YOU WANT TO ROUTE OUTPUT TO A DIFFERENT NODE (YES/NO)? = = = >

   Answer NO and prompting will proceed to the next topic. Output will be held in the output queue where it can be viewed via the IOF option of SPF. Answer YES and ROCETS prompts for an alternate node and userid.

   ENTER DESTINATION NODE, (IE. PWAGPDH.E092928) = = = >

2. DO YOU WANT TO RECONFIGURE (YES/NO)? = = = >

   Answer NO to run a previously configured model

3. ENTER DATASET CONTAINING ROCETS FORTRAN, NO QUOTES = = = >

   Enter here the complete TSO dataset name containing a configured model without quotes. For example: userid.MYMODEL.FORTRAN.

4. ENTER DATASET CONTAINING ROCETS INPUT, NO QUOTES = = = >

   Enter here the complete TSO data name containing run time input. For example: ROCETS.DATA(RTTBE001).

5. ENTER DATASET TO CONTAIN LOAD MODULE, NO QUOTES, PRESS ENTER FOR TEMPORARY LOAD

   Press enter, no input, for a temporary load dataset. Enter a complete dataset name with the correct DCB attributes for a load module library to keep the load module. If a dataset with incorrect DCB attributes is specified the job will fail.

At this time the following message is displayed and the job is submitted to the MVS batch machine for execution.

FILE userid.ROCETS.TEMP.JCL CONTAINS SUBMITTED JCL.

This page was intentionally left blank.

Appendix B
Example Pump Module

A FORTRAN code listing of a Pump Module (PUMP01) and the engineering documentation is presented as an example. all of the engineering modules, sub-modules, and utilities are contained in the SDS, P&W FR-20284 (Reference 4).

```
----------------------------------------------------------------------
                         SUBROUTINE PUMP01
----------------------------------------------------------------------

      SUBROUTINE PUMP01 ( IPRPL   , IUPDAT  , MAP     , MODN    ,
     *                    NOD1    , NOD2    , NOD3    , NOD4    ,
     *                    HDD     , HTIN    , PTIN    , RHOIN   ,
     *                    RHOOUT  , SNREF   , GRATIO  , SND     ,
     *                    TRQD    , WD      , WIN     , HTOUT   ,
     *                    PTOUT   , TORQ    )
C*********************************************************************
C xBEGIN CLASS PUMP01                                                 *
C                                                                     *
C   SUBPROGRAM PUMP01            UNCLASSIFIED               SID: E950 *
C                                                                     *
C                         UNITED TECHNOLOGIES CORPORATION             *
C .-                               PRATT & WHITNEY                    *
C                              WEST PALM BEACH, FLORIDA               *
C xEND CLASS PUMP01                                                   *
C*********************************************************************
C xBEGIN PURPOSE PUMP01                                               *
C                                                                     *
C        THIS ROUTINE REPRESENTS A CONSTANT DENSITY PUMP.             *
C                                                                     *
C xEND PURPOSE PUMP01                                                 *
C*********************************************************************
C xBEGIN HISTORY PUMP01                                               *
C                                                                     *
C  - WRITTEN                                   08/23/89 M.H.SABATELLA *
C                                                       AND J.P.SPINN *
C  - ADDED CHECK FOR FLOW APPROX. EQUAL         3/27/90 T.F. DENMAN   *
C    TO ZERO TO ELIMINATE DIVIDE CHECK                                *
C    IN DH CALCULATION.                                               *
C  - ADDED CHECK FOR TORQ APPROX. EQUAL         4/10/90 T.F. DENMAN   *
C    TO ZERO TO ELIMINATE DIVIDE CHECK                                *
C    IN ETA CALCULATION.                                              *
C                                                                     *
C xEND HISTORY PUMP01                                                 *
C*********************************************************************
C xBEGIN SCHEMATIC PUMP01                                             *
C                                                                     *
C                                /|\ NOD3                             *
C                                 |                                   *
C                                ***|                                 *
C                                *  |                                 *
C                               *   |                                 *
C                              *    |                                 *
C                            *      |                                 *
C                         ***       |                                 *
C          NOD1   ----> |           |==============                   *
C                 ----> |           |    NOD4                         *
C          NOD2   ----> |           |==============                   *
C                         ***       |                                 *
C                            *      |                                 *
```

SUBROUTINE PUMP01

```
C                               *    |                                  *
C                                *   |                                  *
C                                 *  |                                  *
C                                  ***|                                 *
C                                                                       *
C *END SCHEMATIC PUMP01                                                 *
C************************************************************************
C *BEGIN DESCRIPTION PUMP01                                             *
C                                                                       *
C    INPUTS :                                                           *
C              IPRPL  -    PRPL01 OUTPUT FLAG                           *
C                          0 = NO PRINT                                 *
C                          1 = PRINT                                    *
C              IUPDAT -    UPDATE FLAG                                  *
C                         -1 = INITIALIZATION/SS BALANCE                *
C                          0 = TRANSIENT ITERATION PASS                 *
C                          1 = TRANSIENT UPDATE PASS                    *
C              MAP    -    EXTERNAL PUMP CHAR. MAP                      *
C              MODN   -    MODULE NAME (4 CHARACTERS)                   *
C              NOD1   -    FLOW NODE (4 CHARACTERS)                     *
C              NOD2   -    INLET THERMAL NODE (4 CHAR.)                 *
C              NOD3   -    EXIT THERMAL NODE (4 CHAR.)                  *
C              NOD4   -    SHAFT NODE (4 CHARACTERS)                    *
C              HDD    -    PUMP DESIGN HEAD                             *
C              HTIN   -    INLET ENTHALPY                               *
C              WIN    -    INLET FLOW                                   *
C              PTIN   -    INLET PRESSURE                               *
C              RHOIN  -    INLET DENSITY                                *
C              RHOOUT -    EXIT DENSITY                                 *
C              SNREF  -    SPEED OF THE SHAFT                           *
C              SN     -    SPEED OF THE PUMP                            *
C              SND    -    PUMP DESIGN SPEED                            *
C              TRQD   -    PUMP DESIGN TORQUE                           *
C              WD     -    PUMP DESIGN FLOW                             *
C                                                                       *
C    OUTPUTS:                                                           *
C              HTOUT  -    EXIT ENTHALPY                                *
C              PTOUT  -    EXIT PRESSURE                                *
C              TORQ   -    TORQUE REQUIRED                              *
C                                                                       *
C    INPUTS FROM GUNITS COMMON:                                         *
C              GC     -    UNITS CONVERSION FACTOR                      *
C              GR     -    GRAVITATIONAL CONSTANT                       *
C              RJ     -    PROPORTIONALITY FACTOR J                     *
C                                                                       *
C *END DESCRIPTION PUMP01                                               *
C************************************************************************
C *BEGIN DERIVATION PUMP01                                              *
C                                                                       *
C DERIVATION OF PUMP DISCHARGE PRESSURE CALCULATION                     *
```

SUBROUTINE PUMP01

```
C
C    FOR A CONSTANT DENSITY PUMP, GIVEN THE FLUID DENSITY,
C    ROTATIONAL SPEED, AND FLOW THROUGH THE PUMP, THE
C    PUMP HEAD RISE AND REQUIRED TORQUE CAN BE DETERMINED
C    FROM THE PUMP MAP.
C
C    FROM CONSERVATION OF ENERGY, THE DISCHARGE PRESSURE
C    CAN BE CALCUALTED AS:
C
C    -----------------------------------
C    |                        G         |
C    |   POUT = HEAD * RHO * (--) + PIN |
C    |                        GC        |
C    -----------------------------------
C
C
C DERIVATION OF EFFICIENCY
C
C    DEFINE EFFICIENCY, ETA
C
C               WORK DONE ON THE FLUID
C       ETA = ----------------------
C                  ENERGY AVAILABLE
C
C    CALCULATE ANGULAR VELOCITY, OMEGA
C
C                     2 * PI
C       OMEGA = N * (------)
C                       60
C
C    WHERE N IS THE ROTATIONAL SPEED IN RPM
C
C    ----------------------------
C    |         HEAD * N * (G/GC) |
C    |  ETA = ----------------- |
C    |          TORQUE * OMEGA   |
C    ----------------------------
C
C
C DERIVATION OF DISCHARGE ENTHALPY CALCULATION
C
C    POWER CAN BE DEFINED IN TERMS OF TORQUE AND OMEGA AS:
C
C                TORQUE * OMEGA
C       POWER = --------------                                (1)
C                     RJ
C
C    POWER CAN ALSO BE DEFINED IN TERMS OF THE CHANGE IN ENTHALPY
C    DH AND THE FLOWRATE W AS:
C
```

```
--------------------------------------------------------------------
                         SUBROUTINE PUMP01
--------------------------------------------------------------------

C       POWER = W * DH                                    (2)    *
C                                                                *
C    EQUATING (1) AND (2) AND SOLVING FOR DH YIELDS:             *
C                                                                *
C               TORQUE * OMEGA                                   *
C       DH = --------------                                      *
C                   W * RJ                                       *
C                                                                *
C    GIVEN THE INLET ENTHALPY, THE EXIT ENTHALPY IS:             *
C                                                                *
C    ---------------------                                       *
C    | HOUT = HIN + DH  |                                        *
C    ---------------------                                       *
C                                                                *
C xEND DERIVATION PUMP01                                         *
C*****************************************************************
C xBEGIN COMMENTS PUMP01                                         *
C                                                                *
C                                                                *
C        1.  THE EXTERNAL MAP RETURNS EXIT PRESSURE AND TORQUE   *
C            AS A FUNCTION OF FLOW, EXIT DENSITY, AND SPEED.     *
C                                                                *
C        2.  FOR DESCRIPTION OF COMMON "GUNITS", SEE SUBROUTINE  *
C            "UNIT00".                                           *
C                                                                *
C        3.  THE GEAR RATIO IS DEFINED AS THE SHAFT SPEED DIVIDED *
C            BY THE PUMP SPEED.                                  *
C                                                                *
C        4.  THE SIGN OF THE TORQUE FOR A PUMP IS NEGATIVE BY    *
C            CONVENTION FOR PROPER INTERFACING WITH ROTR00.      *
C                                                                *
C xEND COMMENTS PUMP01                                           *
C*****************************************************************
C xBEGIN INTERFACE PUMP01                                        *
C                                                                *
C----------------------------------------------------------------*
C| CALL LIST|  SYSTEM  |  SYSTEM TAG  |    ARRAY    |  I/O   | VAR  |*
C|   NAME   |   NAME   |              |    STATUS   | STATUS | TYPE |*
C----------------------------------------------------------------*
C| IPRPL    | IPRPL    | GLOBAL       |             | IN 0   | I*4  |*
C| IUPDAT   | IUPDAT   | GLOBAL       |             | IN 0   | I*4  |*
C| MAP      | MAP      | MAP          |             | IN 0   | ---  |*
C| MODN     | MODN     | NAME 0       |             | IN 0   | C*4  |*
C| NOD1     | NOD1     | NAME 1       |             | IN 0   | C*4  |*
C| NOD2     | NOD2     | NAME 2       |             | IN 0   | C*4  |*
C| NOD3     | NOD3     | NAME 3       |             | IN 0   | C*4  |*
C| NOD4     | NOD4     | NAME 4       |             | IN 0   | C*4  |*
C| HDD      | HDD      | DESIGN       |             | IN 0   | R*4  |*
C| HTIN     | HT       | VARIABLE     |             | IN 2   | R*4  |*
C| PTIN     | PT       | VARIABLE     |             | IN 2   | R*4  |*
```

```
-------------------------------------------------------------------------
                          SUBROUTINE PUMP01
-------------------------------------------------------------------------

C| RHOIN    | RHO      | VARIABLE     |                | IN 2   | R*4  |*
C| RHOOUT   | RHO      | VARIABLE     |                | IN 3   | R*4  |*
C| SNREF    | SN       | VARIABLE     |                | IN 4   | R*4  |*
C| GRATIO   | GEAR     | DESIGN       |                | IN 0   | R*4  |*
C| SND      | SND      | DESIGN       |                | IN 0   | R*4  |*
C| TRQD     | TRQD     | DESIGN       |                | IN 0   | R*4  |*
C| WD       | WD       | DESIGN       |                | IN 0   | R*4  |*
C| WIN      | W        | VARIABLE     |                | IN 1   | R*4  |*
C| HTOUT    | HT       | VARIABLE     |                | OUT 3  | R*4  |*
C| PTOUT    | PT       | VARIABLE     |                | OUT 3  | R*4  |*
C| TORQ     | TORQ     | VARIABLE     |                | OUT 0  | R*4  |*
C-------------------------------------------------------------------------*
C                                                                         *
C *END INTERFACE PUMP01                                                   *
C**************************************************************************
C *BEGIN UNITS PUMP01                                                     *
C                                                                         *
C----------------------------------------                                 *
C| CALL LIST|   ENGLISH   |      SI       |                               *
C|   NAME   |             |               |                               *
C----------------------------------------                                 *
C| IPRPL    | D'LESS      | D'LESS        |                               *
C| IUPDAT   | D'LESS      | D'LESS        |                               *
C| MAP      | D'LESS      | D'LESS        |                               *
C| MODN     | D'LESS      | D'LESS        |                               *
C| NOD1     | D'LESS      | D'LESS        |                               *
C| NOD2     | D'LESS      | D'LESS        |                               *
C| NOD3     | D'LESS      | D'LESS        |                               *
C| NOD4     | D'LESS      | D'LESS        |                               *
C| HDD      | IN          | M             |                               *
C| HTIN     | BTU/LBM     | J/KG          |                               *
C| PTIN     | LBF/IN**2   | N/M**2        |                               *
C| RHOIN    | LBM/IN**2   | KG/M**3       |                               *
C| RHOOUT   | LBM/IN**2   | KG/M**3       |                               *
C| SNREF    | RPM         | RPM           |                               *
C| GRATIO   | D'LESS      | D'LESS        |                               *
C| SND      | RPM         | RPM           |                               *
C| TRQD     | IN-LBF      | N-M           |                               *
C| WD       | LBM/S       | KG/S          |                               *
C| WIN      | LBM/S       | KG/S          |                               *
C| HTOUT    | BTU/LBM     | J/KG          |                               *
C| PTOUT    | LBF/IN**2   | N/M**2        |                               *
C| TORQ     | IN-LBF      | N-M           |                               *
C----------------------------------------                                 *
C                                                                         *
C *END UNITS PUMP01                                                       *
C**************************************************************************
C *BEGIN KEYWORDS PUMP01                                                  *
C                                                                         *
C  NOD1     : UPSTREAM FLOW, INLET FLOW                                   *
```

```
--------------------------------------------------------------------------
                              SUBROUTINE PUMP01
--------------------------------------------------------------------------

C  NOD2     : UPSTREAM PROP, INLET PROP                              ;   *
C  NOD3     : DOWNSTREAM PROP, EXIT PROP                             ;   *
C  NOD4     : SHAFT, ROTOR                                           ;   *
C                                                                        *
C *END KEYWORDS PUMP01                                                   *
C*************************************************************************
C *BEGIN SUBROUTINES REQUIRED PUMP01                                     *
C                                                                        *
C   SUBROUTINES REQUIRED :  PMAPXX   MAP (EXTERNAL)                      *
C                           PRPL01                                       *
C                                                                        *
C *END SUBROUTINES REQUIRED PUMP01                                       *
C*************************************************************************
C *BEGIN COMMONS REQUIRED PUMP01                                         *
C                                                                        *
C   COMMONS REQUIRED :  GUNITS                                           *
C                                                                        *
C *END COMMONS REQUIRED PUMP01                                           *
C*************************************************************************
      CHARACTER*4 MODN, NOD1, NOD2, NOD3, NOD4
      EXTERNAL MAP
      COMMON / GUNITS /   IUNIT  , GC     , GR     , RJ     , RU     ,
     *                    CLEN   , CMASS  , CFORCE , CTEMP  , CENERGY,
     *                    FLOCON
      DATA PI        / 3.141592654  /
C*********************************
C  MISCELLANEOUS INITIALIZATIONS  *
C*********************************
      SN    = SNREF / GRATIO
      SNRAD = (SN*2.*PI/60.)
C***********************************************************
C READ MAP WITH FLOW AND SPEED FOR HEAD RISE AND TORQUE  *
C***********************************************************
      CALL MAP ( IUPDAT, HDD   , RHOOUT, SN    , SND   , TRQD  ,
     $             WD    , WIN   , HD    , TORQ  )
C**********************************************
C CALCULATE DISCHARGE PRESSURE AND EXIT ENTHALPY  *
C**********************************************
      PTOUT  =  HD*RHOOUT*GR/GC + PTIN
      IF( TORQ .GT. .01 ) THEN
        ETA    =  WIN*HD*GR/(SNRAD*TORQ*GC)
      ELSE
        ETA    =  0.0
      ENDIF
      POWR   =  TORQ*SNRAD/RJ
      IF( WIN .GT. .01 ) THEN
        DH     =  POWR/WIN
      ELSE
        DH     =  0.0
      ENDIF
```

SUBROUTINE PUMP01

```
      HTOUT  =  HTIN + DH
      TORQ   =  - TORQ
C*******************
C  PURPLE SECTION  *
C*******************
      IF (IPRPL .GT. 0) THEN
        CALL PRPL01(-9,' MODULE ',MODN//' OUTPUT ',DUMMY)
        CALL PRPL01(1,'HD'//MODN//'  ','            ',HD     )
        CALL PRPL01(1,'TRQD'//MODN    ,'            ',TRQD   )
        CALL PRPL01(1,'HDD'//MODN//' ','            ',HDD    )
        CALL PRPL01(1,'ETA'//MODN//' ','            ',ETA    )
        CALL PRPL01(1,'SN'//NOD4//'  ','            ',SNREF  )
        CALL PRPL01(1,'SN'//MODN//'  ','            ',SN     )
        CALL PRPL01(1,'HD'//MODN//'  ','            ',HD     )
        CALL PRPL01(1,'HT'//NOD2//'  ','            ',HTIN   )
        CALL PRPL01(1,'HT'//NOD3//'  ','            ',HTOUT  )
        CALL PRPL01(1,'PT'//NOD3//'  ','            ',PTOUT  )
        CALL PRPL01(1,'TORQ'//MODN    ,'            ',TORQ   )
        CALL PRPL01(1,'RHO'//NOD2//' ','            ',RHOIN  )
        CALL PRPL01(1,'RHO'//NOD3//' ','            ',RHOOUT)
        CALL PRPL01(1,'W'//NOD1//'   ','            ',WIN    )
        CALL PRPL01(1,'DH'//MODN//'  ','            ',DH     )
        CALL PRPL01(1,'POWR'//MODN    ,'            ',POWR   )
        CALL PRPL01(1,'SND'//MODN//' ','            ',SND    )
        CALL PRPL01(1,'PT'//NOD2//'  ','            ',PTIN   )
      ENDIF
 99   CONTINUE
      RETURN
      END
```

PUMP01

This routine represents a constant density pump.

```
                                /|\ NOD3
                                 |
                               ***|
                               *  |
                              *   |
                             *    |
                            *     |
                         ***      |
      NOD1   ---->  |             |**************
             ---->  |             |     NOD4
      NOD2   ---->  |             |**************
                         ***      |
                            *     |
                             *    |
                              *   |
                               *  |
                               ***|
```

I/O DESCRIPTION:

```
INPUTS :
              IPRPL   -    PRPL01 OUTPUT FLAG
                           0 = NO PRINT
                           1 = PRINT
              IUPDAT  -    UPDATE FLAG
                          -1 = INITIALIZATION/SS BALANCE
                           0 = TRANSIENT ITERATION PASS
                           1 = TRANSIENT UPDATE PASS
              MAP     -    EXTERNAL PUMP CHAR. MAP
              MODN    -    MODULE NAME (4 CHARACTERS)
              NOD1    -    FLOW NODE (4 CHARACTERS)
              NOD2    -    INLET THERMAL NODE (4 CHAR.)
              NOD3    -    EXIT THERMAL NODE (4 CHAR.)
              NOD4    -    SHAFT NODE (4 CHARACTERS)
              HDD     -    PUMP DESIGN HEAD
              HTIN    -    INLET ENTHALPY
              WIN     -    INLET FLOW
              PTIN    -    INLET PRESSURE
              RHOIN   -    INLET DENSITY
              RHOOUT  -    EXIT DENSITY
              SNREF   -    SPEED OF THE SHAFT
              SN      -    SPEED OF THE PUMP
              SND     -    PUMP DESIGN SPEED
              TRQD    -    PUMP DESIGN TORQUE
              WD      -    PUMP DESIGN FLOW

OUTPUTS:
              HTOUT   -    EXIT ENTHALPY
              PTOUT   -    EXIT PRESSURE
              TORQ    -    TORQUE REQUIRED

INPUTS FROM GUNITS COMMON:
              GC      -    UNITS CONVERSION FACTOR
              GR      -    GRAVITATIONAL CONSTANT
              RJ      -    PROPORTIONALITY FACTOR J
```

COMMENTS:

1. The external map returns exit pressure and torque as a function of flow, exit density, and speed.

2. For description of common "GUNITS", see subroutine "UNIT00".

3. The gear ratio is defined as the shaft speed divided by the pump speed.

4. The sign of the torque for a pump is negative by convention for proper interfacing with ROTR00.

KEYWORDS:

Node keywords are part of the interface cards for each module. In the configuration input for a module, an I/O list containing the node keywords is used to specify the nodal connections. The node keywords for this module are:

```
NOD1    : UPSTREAM FLOW, INLET FLOW
NOD2    : UPSTREAM PROP, INLET PROP
NOD3    : DOWNSTREAM PROP, EXIT PROP
NOD4    : SHAFT, ROTOR
```

DERIVATIONS:

A derivation of the calculations used in this module follows:

```
DERIVATION OF PUMP DISCHARGE PRESSURE CALCULATION

   FOR A CONSTANT DENSITY PUMP, GIVEN THE FLUID DENSITY,
   ROTATIONAL SPEED, AND FLOW THROUGH THE PUMP, THE
   PUMP HEAD RISE AND REQUIRED TORQUE CAN BE DETERMINED
   FROM THE PUMP MAP.

   FROM CONSERVATION OF ENERGY, THE DISCHARGE PRESSURE
   CAN BE CALCUALTED AS:

   -----------------------------------
   |                       G         |
   |  POUT = HEAD * RHO * (--) + PIN |
   |                       GC        |
   -----------------------------------


DERIVATION OF EFFICIENCY

   DEFINE EFFICIENCY, ETA

              WORK DONE ON THE FLUID
      ETA = ----------------------
                ENERGY AVAILABLE

   CALCULATE ANGULAR VELOCITY, OMEGA

                       2 * PI
      OMEGA = N * (------)
                         60

   WHERE N IS THE ROTATIONAL SPEED IN RPM

   ----------------------------
   |         HEAD * W * (G/GC) |
   |  ETA = ----------------   |
   |          TORQUE * OMEGA   |
   ----------------------------


DERIVATION OF DISCHARGE ENTHALPY CALCULATION
```

POWER CAN BE DEFINED IN TERMS OF TORQUE AND OMEGA AS:

$$POWER = \frac{TORQUE * OMEGA}{RJ} \qquad (1)$$

POWER CAN ALSO BE DEFINED IN TERMS OF THE CHANGE IN ENTHALPY DH AND THE FLOWRATE W AS:

$$POWER = W * DH \qquad (2)$$

EQUATING (1) AND (2) AND SOLVING FOR DH YIELDS:

$$DH = \frac{TORQUE * OMEGA}{W * RJ}$$

GIVEN THE INLET ENTHALPY, THE EXIT ENTHALPY IS:

$$\boxed{HOUT = HIN + DH}$$

MODULE INTERFACE CARDS:

The configuration processor uses the following interface cards to generate the main program call list.

CALL LIST NAME	SYSTEM NAME	SYSTEM TAG	ARRAY STATUS	I/O STATUS	VAR TYPE
IPRPL	IPRPL	GLOBAL		IN 0	I*4
IUPDAT	IUPDAT	GLOBAL		IN 0	I*4
MAP	MAP	MAP		IN 0	---
MODN	MODN	NAME 0		IN 0	C*4
NOD1	NOD1	NAME 1		IN 0	C*4
NOD2	NOD2	NAME 2		IN 0	C*4
NOD3	NOD3	NAME 3		IN 0	C*4
NOD4	NOD4	NAME 4		IN 0	C*4
HDD	HDD	DESIGN		IN 0	R*4
HTIN	HT	VARIABLE		IN 2	R*4
PTIN	PT	VARIABLE		IN 2	R*4
RHOIN	RHO	VARIABLE		IN 2	R*4
RHOOUT	RHO	VARIABLE		IN 3	R*4
SNREF	SN	VARIABLE		IN 4	R*4
GRATIO	GEAR	DESIGN		IN 0	R*4
SND	SND	DESIGN		IN 0	R*4
TROD	TROD	DESIGN		IN 0	R*4
WD	WD	DESIGN		IN 0	R*4
WIN	W	VARIABLE		IN 1	R*4
HTOUT	HT	VARIABLE		OUT 3	R*4
PTOUT	PT	VARIABLE		OUT 3	R*4
TORQ	TORQ	VARIABLE		OUT 0	R*4

The configuration processor uses the following interface cards to provide units for parameters whose output is requested by a PRPL01 call from the main program

CALL LIST NAME	ENGLISH	SI

IPRPL	D'LESS	D'LESS
IUPDAT	D'LESS	D'LESS
MAP	D'LESS	D'LESS
MODN	D'LESS	D'LESS
NOD1	D'LESS	D'LESS
NOD2	D'LESS	D'LESS
NOD3	D'LESS	D'LESS
NOD4	D'LESS	D'LESS
HDD	IN	M
HTIN	BTU/LBM	J/KG
PTIN	LBF/IN**2	N/M**2
RHOIN	LBM/IN**2	KG/M**3
RHOOUT	LBM/IN**2	KG/M**3
SNREF	RPM	RPM
GRATIO	D'LESS	D'LESS
SND	RPM	RPM
TRQD	IN-LBF	N-M
WD	LBM/S	KG/S
WIN	LBM/S	KG/S
HTOUT	BTU/LBM	J/KG
PTOUT	LBF/IN**2	N/M**2
TORQ	IN-LBF	N-M

Following is a list of the subroutines that are required by this module.

SUBROUTINES REQUIRED : PMAPXX MAP (EXTERNAL)
PRPL01

Following is a list of the commons that are required by this module.

COMMONS REQUIRED : GUNITS

Appendix C
Interfaced NASA Control Model

Presented is the listing of the NASA MSFC FORTRAN Control Model with the ROCETS interface incorporated in the comment cards.

SUBROUTINE CNTL00

```
      SUBROUTINE CNTL00(IPRPL     , IUPDAT   , TIME     , MODN     , QN     ,
     $                  PCN       , PFDN     , REFN     , QFFM     , PC     ,
     $                  PFD1      , TFP1     , PCREF    , MRREF    , DXOPI  ,
     $                  DXFPI     , XMFVC    , XMOVC    , XCCVC    , XFPVC  ,
     $                  XOPVC     , EOPI     , EFPI     , AAA      )
C*********************************************************************
C %BEGIN CLASS CNTL00                                                 *
C                                                                     *
C   SUBPROGRAM CNTL00            UNCLASSIFIED              SID: E950  *
C                                                                     *
C %END CLASS CNTL00                                                   *
C*********************************************************************
C %BEGIN PURPOSE CNTL00                                               *
C                                                                     *
C        SSME CONTROL FOR MAINSTAGE OPERATION                         *
C                                                                     *
C %END PURPOSE CNTL00                                                 *
C*********************************************************************
C %BEGIN HISTORY CNTL00                                               *
C                                                                     *
C        OBTAINED FROM NASA/MSFC FOR TESTING ROCETS SYSTEM  MAY 1990  *
C                                                                     *
C %END HISTORY CNTL00                                                 *
C*********************************************************************
C %BEGIN DESCRIPTION CNTL00
C
C IPRPL    PRPL01 OUTPUT FLAG                                  D'LESS
C            0 = NO PRINT
C            1 = PRINT
C IUPDAT   UPDATE FLAG                                         D'LESS
C           -1 = INITIALIZATION/SS BALANCE
C            0 = CONTROL BYPASSED DURING ITERATION
C            1 = CALCULATIONS PERFORMED AFTER CONVERGED POINT
C TIME     SIMULATED TIME                                      SEC
C MODN     MODULE NAME                                         D'LESS
C QN       FUEL FLOW FEEDBACK NODE                             D'LESS
C PCN      CHAMBER PRESSURE NODE                               D'LESS
C PFDN     FUEL PUMD DISCHARGE NODE                            D'LESS
C REFN     NODE NAME FOR 'REFERENCE' INPUTS (PCREF, MRREF)     D'LESS
C QFFM     MEASURED FUEL FLOW                                  GPM
C PC       MEASURED CHAMBER PRESSURE                           PSIA
C PFD1     MEASURED LOW PRESSURE FUEL PUMP DISCHARGE PRESSURE  PSIA
C TFP1     MEASURED LOW PRESSURE FUEL PUMP DISCHARGE TEMPERATURE DEG. R
C PCREF    REFERENCE (REQUESTED) CHAMBER PRESSURE              PSIA
C MRREF    REFERENCE (REQUESTED) MIXTURE RATIO                 D-LESS
C
C INPUT ON IUPDAT=-1, OTHERWISE OUTPUT
C
C DXOPI    OPV INTEGRATOR VALUE                                PCT
C DXFPI    FPV INTEGRATOR VALUE                                PCT
```

SUBROUTINE CNTL00

```
C
C OUTPUTS
C
C XMFVC    MFV ACTUATOR POSITION COMMAND                          PCT
C XMOVC    MOV ACTUATOR POSITION COMMAND                          PCT
C XCCVC    CCV ACTUATOR POSITION COMMAND                          PCT
C XFPVC    FPV ACTUATOR POSITION COMMAND                          PCT
C XOPVC    OPV ACTUATOR POSITION COMMAND                          PCT
C EOPI     OPV INTEGRATOR ERROR                                   ---
C EFPI     FPV INTEGRATOR ERROR                                   ---
C AAA      STORAGE ARRAY FOR RESTART                              ---
C
C *END DESCRIPTION CNTL00
C**********************************************************************
C *BEGIN COMMENTS CNTL00                                              *
C                                                                     *
C *END COMMENTS CNTL00                                                *
C**********************************************************************
C *BEGIN INTERFACE CNTL00                                             *
C                                                                     *
C--------------------------------------------------------------------*
C| CALL LIST| SYSTEM  | SYSTEM TAG |     ARRAY    | I/O    | VAR |*
C|   NAME   |  NAME   |            |     STATUS   | STATUS | TYPE|*
C--------------------------------------------------------------------*
C| IPRPL    | IPRPL   | GLOBAL     |              | IN 0   | I*4 |*
C| IUPDAT   | IUPDAT  | GLOBAL     |              | IN 0   | I*4 |*
C| TIME     | TIME    | GLOBAL     |              | IN 0   | R*4 |*
C| MODN     | MODN    | NAME 0     |              | IN 0   | C*4 |*
C| QN       | QN      | NAME 1     |              | IN 0   | C*4 |*
C| PCN      | PCN     | NAME 2     |              | IN 0   | C*4 |*
C| PFDN     | PFDN    | NAME 3     |              | IN 0   | C*4 |*
C| REFN     | REFN    | NAME 4     |              | IN 0   | C*4 |*
C| QFFM     | Q       | VARIABLE   |              | IN 1   | R*4 |*
C| PC       | PT      | VARIABLE   |              | IN 2   | R*4 |*
C| PFD1     | PT      | VARIABLE   |              | IN 3   | R*4 |*
C| TFP1     | TT      | VARIABLE   |              | IN 3   | R*4 |*
C| PCREF    | PC      | VARIABLE   |              | IN 4   | R*4 |*
C| MRREF    | MR      | VARIABLE   |              | IN 4   | R*4 |*
C| DXOPI    | DXOP    | VARIABLE   |              | IN 0   | R*4 |*
C| DXFPI    | DXFP    | VARIABLE   |              | IN 0   | R*4 |*
C| XMFVC    | XMFV    | VARIABLE   |              | OUT 0  | R*4 |*
C| XMOVC    | XMOV    | VARIABLE   |              | OUT 0  | R*4 |*
C| XCCVC    | XCCV    | VARIABLE   |              | OUT 0  | R*4 |*
C| XFPVC    | XFPV    | VARIABLE   |              | OUT 0  | R*4 |*
C| XOPVC    | XOPV    | VARIABLE   |              | OUT 0  | R*4 |*
C| EOPI     | EOPI    | VARIABLE   |              | OUT 0  | R*4 |*
C| EFPI     | EFPI    | VARIABLE   |              | OUT 0  | R*4 |*
C| AAA      | AAA     | VARIABLE   | ARRAY 21     | OUT 0  | R*4 |*
C--------------------------------------------------------------------*
C                                                                     *
```

SUBROUTINE CNTL00

```
C   SUBROUTINES REQUIRED :  PRPL01                                        *
C                                                                         *
C *END SUBROUTINES REQUIRED CNTL00                                        *
C*************************************************************************
C *BEGIN COMMONS REQUIRED CNTL00                                          *
C                                                                         *
C   COMMONS REQUIRED :  NONE                                              *
C                                                                         *
C *END COMMONS REQUIRED CNTL00                                            *
C*************************************************************************
      CHARACTER * 4  MODN, QN, PCN, PFDN, REFN
C
      REAL MRREF,MRCONT,MRG,MRREFX,MRG65,MRG100
      REAL MFVRL,MOVRL,MFVRX,MOVRX
      DIMENSION A0TAB(10),A1TAB(10),EPLTAB(10)
C*************************************************************************
C
C                          CONTROLER GAINS
C
C  PCG50    LOW  CORNER POINT FOR CHAMBER PRESSURE PROPORTIONAL GAIN
C  PCG100   HIGH CORNER POINT FOR CHAMBER PRESSURE PROPORTIONAL GAIN
C  PCRL     CHAMBER PRESSURE RATE LIMIT
C
      DATA PCG50 /.6/, PCG100/1.0/, PCRL/300./
C
C  XFG50    LOW  CORNER POINT FOR MAINSTAGE CROSSFEED GAIN
C  XFG100   HIGH CORNER POINT FOR MAINSTAGE CROSSFEED GAIN
C
      DATA XFG50 /1.15 /, XFG100/1.15/
C
C  MRG65    LOW  CORNER POINT FOR MIXTURE RATIO PROPORTIONAL GAIN
C  MRG100   HIGH CORNER POONT FOR MIXTURE RATIO PROPORTIONAL GAIN
C
      DATA MRG65 /.2/,  MRG100/.5/
C
C  XOPPG    OPOV PROPORTIONAL GAIN
C  XOPIG    OPOV INTEGRAL GAIN
C  XOPVST   START BIAS FOR OPOV
C  XOPDCO   OPOV DELTA COMMAND OFFSET
C
      DATA XOPPG /.0113/, XOPIG/.00068/, XOPVST/64.52/, XOPDCO/ 0./
C
C  XFPPG    FPOV PROPORTIONAL GAIN
C  XFPIG    FPOV INTEGRAL GAIN
C  XFPVST   START BIAS FOR FPOV
C
      DATA XFPPG / 7./, XFPIG/.40/, XFPVST/77.22/
C
C*************************************************************************
C                       RATE LIMITS
```

SUBROUTINE CNTL00

```
C    SUBROUTINES REQUIRED :  PRPL01                                  *
C                                                                    *
C *END SUBROUTINES REQUIRED CNTL00                                   *
C*********************************************************************
C *BEGIN COMMONS REQUIRED CNTL00                                     *
C                                                                    *
C    COMMONS REQUIRED :  NONE                                        *
C                                                                    *
C *END COMMONS REQUIRED CNTL00                                       *
C*********************************************************************
      CHARACTER * 4  MODN, QN, PCN, PFDN, REFN
C
      REAL MRREF,MRCONT,MRG,MRREFX,MRG65,MRG100
      REAL MFVRL,MOVRL,MFVRX,MOVRX
      DIMENSION A0TAB(10),A1TAB(10),EPLTAB(10)
C*********************************************************************
C
C                          CONTROLER GAINS
C
C  PCG50    LOW  CORNER POINT FOR CHAMBER PRESSURE PROPORTIONAL GAIN
C  PCG100   HIGH CORNER POINT FOR CHAMBER PRESSURE PROPORTIONAL GAIN
C  PCRL     CHAMBER PRESSURE RATE LIMIT
C
      DATA PCG50 /.6/, PCG100/1.0/, PCRL/300./
C
C  XFG50    LOW  CORNER POINT FOR MAINSTAGE CROSSFEED GAIN
C  XFG100   HIGH CORNER POINT FOR MAINSTAGE CROSSFEED GAIN
C
      DATA XFG50 /1.15 /, XFG100/1.15/
C
C  MRG65    LOW  CORNER POINT FOR MIXTURE RATIO PROPORTIONAL GAIN
C  MRG100   HIGH CORNER POONT FOR MIXTURE RATIO PROPORTIONAL GAIN
C
      DATA MRG65 /.2/,  MRG100/.5/
C
C  XOPPG    OPOV PROPORTIONAL GAIN
C  XOPIG    OPOV INTEGRAL GAIN
C  XOPVST   START BIAS FOR OPOV
C  XOPDCO   OPOV DELTA COMMAND OFFSET
C
      DATA XOPPG /.0113/, XOPIG/.00068/, XOPVST/64.52/, XOPDCO/ 0./
C
C  XFPPG    FPOV PROPORTIONAL GAIN
C  XFPIG    FPOV INTEGRAL GAIN
C  XFPVST   START BIAS FOR FPOV
C
      DATA XFPPG / 7./, XFPIG/.40/, XFPVST/77.22/
C
C*********************************************************************
C                        RATE LIMITS
```

```
C
      DATA CCVRL /200./,FPVRL/200./,MFVRL/200./,OPVRL/200./,MOVRL/200./
C*********************************************************************
C                 CONSTANTS FOR FUEL DENSITY EQUATION
C
C     RHO = (A0 + (A1+B1*PLPFD)*TLPFD + (A2+B2*PLPFD)*TLPFD*TLPFD
C
      DATA A0/ .38956E+01/, A1/ .6522E-01 /, A2/-.14013E-02/,
     *     B0/ .42739E-02/, B1/-.21467E-03/, B2/ .30926E-05/
C*********************************************************************
C CONVERSION FACTOR:  (448 GPM = 1 FT**3/SEC)
      DATA GTOC/448.83303/
C*********************************************************************
C
C               CONSTANTS FOR LOX FLOW CALCULATION
C
C          C2 = C2A * (PC/RPL)**2 + C2B*(PC/RPL) + C2C
C                WO = (PC + 14.5) / C2 - WH
C
C     NOTE:  USE PCREF INSTEAD OF PC DURING THRUST LIMITING
C
      DATA C2A/-.030621/, C2B/.016555/, C2C/2.92104/
C*********************************************************************
      DATA A0TAB/ 42.75, 42.75, 39.00, 35.00, 35.00, 17.00, 9.40,
     $             9.40,  9.40,-32.40/
      DATA A1TAB/ 0.20, 0.20, 0.25, 0.30, 0.30, 0.50, 0.58, 0.58,
     $            0.58, 0.96/
      DATA EPLTAB/   0., 70., 75., 80., 85., 90., 95., 100., 105.,
     $             110.0/
C
C
C
      DATA RPL   /  3006.  / , XOPVMX/  64.43  / , IFIRST/ 1      /
      DIMENSION AAA(21)
C*********************************************************************
C                           INITIALIZATION
C*********************************************************************
      IF(IUPDAT .LT. 0)THEN
C
         TNEXT    = TIME-.0025
         MNCYC    = 3
         QFX      = QFFM
         PCX      = PC
         PFD1X    = PFD1
         TFP1XX   = TFP1
         TFP1X    = TFP1XX
         PCRFXL   = PCREF
         PCREFX   = PCREF
         MRREFX   = MRREF
         EOPIL    = 0.
```

```
        EFPIL    = 0.
C
        DXFPIL   = DXFPI
        DXOPIL   = DXOPI
C
        RPL50 = RPL * .5
        RPL65 = RPL * .65
C
C  CONVERT PC RATE LIMIT TO PSI PER 20 MSEC
        PCRLX = PCRL / 50.
C SCHEDULE DELTAS
        DPCG = PCG100 - PCG50
        DMRG = MRG100 - MRG65
        DXFG = XFG100 - XFG50
        DXCCV = 100. - 52.
C
C  COMPUTE OPOV DELTA POWER LEVEL
        OPOVDL = (1.5 * XOPVMX) - 97.5
C
C  CONVERT VALVE RATE LIMITS FROM % PER SECOND TO % PER 20 MSEC
        CCVRX = CCVRL / 50.
        FPVRX = FPVRL / 50.
        MFVRX = MFVRL / 50.
        MOVRX = MOVRL / 50.
        OPVRX = OPVRL / 50.
        IFIRST= 0
      ELSEIF(IFIRST.EQ. 1) THEN
C******************************************************************
C UNLOAD THE ARRAY CONTAINING THE VARIABLES REQUIRED FOR RESTART *
C******************************************************************
        TNEXT    = AAA( 1)
        MNCYC    = IFIX(AAA( 2)+.1)
        QFX      = AAA( 3)
        PCX      = AAA( 4)
        PFD1X    = AAA( 5)
        TFP1XX   = AAA( 6)
        TFP1X    = AAA( 7)
        PCRFXL   = AAA( 8)
        PCREFX   = AAA( 9)
        MRREFX   = AAA(10)
        EOPIL    = AAA(11)
        EFPIL    = AAA(12)
        DXFPIL   = AAA(13)
        DXOPIL   = AAA(14)
C
        XOVPS    = AAA(15)
        XFPVC    = AAA(16)
        XCCVXL   = AAA(17)
        XFPVXL   = AAA(18)
        XMFVXL   = AAA(19)
```

SUBROUTINE CNTL00

```
         XMOVXL  = AAA(20)
         XOPVXL  = AAA(21)
C
         RPL50  = RPL * .5
         RPL65  = RPL * .65
         PCRLX  = PCRL / 50.
         DPCG   = PCG100 - PCG50
         DMRG   = MRG100 - MRG65
         DXFG   = XFG100 - XFG50
         DXCCV  = 100. - 52.
         OPOVDL = (1.5 * XOPVMX) - 97.5
         CCVRX  = CCVRL / 50.
         FPVRX  = FPVRL / 50.
         MFVRX  = MFVRL / 50.
         MOVRX  = MOVRL / 50.
         OPVRX  = OPVRL / 50.
         IFIRST = 0
      ENDIF
      IF(IUPDAT .EQ. 0)GO TO 50
CXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
C       T R A N S I E N T     C O N T R O L     S E C T I O N
CXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
      IF(TIME .LT. TNEXT) GO TO 50
      TNEXT = TIME + 0.005
      MNCYC = MNCYC + 1
      GO TO (100,200,300,400),MNCYC
C
C  MINOR LOOP 1
 100  QFX = QFFM
      PCX = PC
      PFD1X = PFD1
      GO TO 50
C
C  MINOR LOOP 2
 200  CONTINUE
      GO TO 50
C
C  MINOR LOOP 3
 300  PCREFX = PCREF
      MRREFX = MRREF
      TFP1X = TFP1XX
      TFP1XX = TFP1
      GO TO 50
C
C  MINOR LOOP 4
 400  PCPCTX = PCX / RPL
      PCPCT = PCPCTX * 100.
      MNCYC = 0
C
C  RATE LIMIT ON PCREF
```

```
 410  D = PCREFX - PCRFXL
      IF(ABS(D).GT.PCRLX) PCREFX = PCRFXL + SIGN(PCRLX,D)
 412  PCRFXL = PCREFX
C
C  COMPUTE OPOV COMMAND LIMIT
 420  EPL = (PCREFX * 100.0 / RPL) + OPOVDL
      DO 421 I=10,1,-1
      IF(EPL.GE.EPLTAB(I)) GO TO 422
 421  CONTINUE
      I = 1
 422  OPOVCL = (A1TAB(I) * EPL) + A0TAB(I) + XOPDCO
      OPOVCL = MIN(OPOVCL,100.0)
C
C  COMPUTE GAIN SCHEDULES
 430  DRPL50 = (PCX - RPL50) / (RPL - RPL50)
      PCG = PCG50 + (DRPL50 * DPCG)
      PCG = MAX(PCG50,MIN(PCG100,PCG))
      XFG = XFG50 + (DRPL50 * DXFG)
      XFG = MAX(XFG50,MIN(XFG100,XFG))
      DRPL65 = (PCX - RPL65) / (RPL - RPL65)
      MRG = MRG65 + (DRPL65 * DMRG)
      MRG = MAX(MRG65,MIN(MRG100,MRG))
C
C  PC ERROR & PROPORTIONAL
 450  DPC = PCREFX - PCX
      EOPV = DPC * PCG
      DXOPP = EOPV * XOPPG
C
C  OPV INTEGRATOR
      EOPI = EOPV
C
C -CHECK FOR THRUST LIMIT
      IF(IUPDAT .LT. 0)XOVPS  = XOPVST + DXOPI
      IF(IUPDAT .GT. 0) THEN
         IF((XOPVS.GE.OPOVCL).AND.(EOPI.GT.0.0)) EOPI = 0.0
      ENDIF
C
 451  IF(IUPDAT .GT. 0) DXOPI = DXOPIL + (XOPIG*(EOPI + EOPIL))
C
      DXOPIL = DXOPI
      EOPIL = EOPI
C  OPOV SUM & LIMIT CHECK
      DXOPV = DXOPP + DXOPI
      XOPVS = DXOPV + XOPVST
      XOPVX = XOPVS
C
C  COMPUTE FUEL DENSITY (RHOH) AND FLOWRATE (WH)
 470  RHOH = ((B2*PFD1X + A2) * TFP1X + (B1*PFD1X + A1)) * TFP1X +
     $B0*PFD1X + A0
C
```

SUBROUTINE CNTL00

```
C  CONVERT Q FROM GAL/MIN TO CU FT/SEC
      QFC = QFX / GTOC
      WH = QFC * RHOH
C
C  CALCULATE OXIDIZER FLOWRATE (WO) AND MIXTURE RATIO
C   USE MEASURED PC IF IN NORMAL MODE
C   USE PCREF IF IN THRUST LIMITING MODE
C
      T = PCX
      TT = T / RPL
      C2 = ((C2A*TT + C2B) * TT) + C2C
      WO = ((T + 14.5)/C2) - WH
      MRCONT = WO / WH
C
C  CROSSFEED
      DXFPX = DXOPV * XFG
C
C  FPV CONTROL
 500  DMR = MRCONT - MRREFX
      EFPV = DMR * MRG
      EFPI = EFPV
C  XFPV ERROR LIMIT
      IF(IUPDAT .LT. 0)XFPVC = DXFPX + DXFPI + XFPVST
C
      IF(IUPDAT .GT. 0)THEN
         IF((XFPVC.GE.102.0).AND.(EFPV.GT.0.)) EFPI = 0.
         IF((XFPVC.LT.0.00) .AND.(EFPV.LT.0.)) EFPI = 0.
      ENDIF
C
      IF(IUPDAT .GT. 0)DXFPI = DXFPIL + (XFPIG * (EFPI + EFPIL))
C
      DXFPIL = DXFPI
      EFPIL = EFPI
      DXFPP = XFPPG * EFPV
      DXFPV = DXFPI + DXFPP
C XFPV SUM
      XFPVX = DXFPV + DXFPX + XFPVST
      XFPVX = MIN(XFPVX,100.0)
C
C  SCHEDULED VALVES
 80   XMFVX = 100.
      XMOVX = 100.
      XCCVX = 52. + (DRPL50 * DXCCV)
      XCCVX = MAX(52.,MIN(100.,XCCVX))
C
C  VALVE RATE LIMITS
C
      IF(IUPDAT .LT. 0)THEN
         XCCVXL = XCCVX
         XFPVXL = XFPVX
```

```
         XMFVXL = XMFVX
         XMOVXL = XMOVX
         XOPVXL = XOPVX
      ENDIF
C
 90   XCCVC = XCCVX
      D = XCCVX - XCCVXL
      IF(ABS(D).GT.CCVRX) XCCVC = XCCVXL + SIGN(CCVRX,D)
 92   XFPVC = XFPVX
      D = XFPVX - XFPVXL
      IF(ABS(D).GT.FPVRX) XFPVC = XFPVXL + SIGN(FPVRX,D)
 94   XMFVC = XMFVX
      D = XMFVX - XMFVXL
      IF(ABS(D).GT.MFVRX) XMFVC = XMFVXL + SIGN(MFVRX,D)
 96   XMOVC = XMOVX
      D = XMOVX - XMOVXL
      IF(ABS(D).GT.MOVRX) XMOVC = XMOVXL + SIGN(MOVRX,D)
 98   XOPVC = XOPVX
      D = XOPVX - XOPVXL
      IF(ABS(D).GT.OPVRX) XOPVC = XOPVXL + SIGN(OPVRX,D)
C
      XCCVXL = XCCVX
      XFPVXL = XFPVX
      XMFVXL = XMFVX
      XMOVXL = XMOVX
      XOPVXL = XOPVX
C*******************************************
C LOAD STORE ARRAYS FOR RESTART CAPABILITY *
C*******************************************
      AAA( 1) =     TNEXT
      AAA( 2) =     FLOAT(MNCYC)
      AAA( 3) =     QFX
      AAA( 4) =     PCX
      AAA( 5) =     PFD1X
      AAA( 6) =     TFP1XX
      AAA( 7) =     TFP1X
      AAA( 8) =     PCRFXL
      AAA( 9) =     PCREFX
      AAA(10) =     MRREFX
      AAA(11) =     EOPIL
      AAA(12) =     EFPIL
      AAA(13) =     DXFPIL
      AAA(14) =     DXOPIL
C
      AAA(15) =     XOVPS
      AAA(16) =     XFPVC
      AAA(17) =     XCCVXL
      AAA(18) =     XFPVXL
      AAA(19) =     XMFVXL
      AAA(20) =     XMOVXL
```

```
      AAA(21) =    XOPVXL
 50   CONTINUE
CXXXXXXXXXXXXXX
C PRINT CALLS *
CXXXXXXXXXXXXXX
      IF(IPRPL .EQ. 0)GO TO 99
      CALL PRPL01(-9,' MODULE ',MODN//' CNTL00 ',DUMMY   )
      CALL PRPL01(1,'A0       ','           ',A0       )
      CALL PRPL01(1,'A1       ','           ',A1       )
      CALL PRPL01(1,'A2       ','           ',A2       )
      CALL PRPL01(1,'B0       ','           ',B0       )
      CALL PRPL01(1,'B1       ','           ',B1       )
      CALL PRPL01(1,'B2       ','           ',B2       )
      CALL PRPL01(1,'CCVRL    ','           ',CCVRL    )
      CALL PRPL01(1,'CCVRX    ','           ',CCVRX    )
      CALL PRPL01(1,'C2       ','           ',C2       )
      CALL PRPL01(1,'C2A      ','           ',C2A      )
      CALL PRPL01(1,'C2B      ','           ',C2B      )
      CALL PRPL01(1,'C2C      ','           ',C2C      )
      CALL PRPL01(1,'D        ','           ',D        )
      CALL PRPL01(1,'DMR      ','           ',DMR      )
      CALL PRPL01(1,'DMRG     ','           ',DMRG     )
      CALL PRPL01(1,'DPC      ','           ',DPC      )
      CALL PRPL01(1,'DPCG     ','           ',DPCG     )
      CALL PRPL01(1,'DRPL50   ','           ',DRPL50   )
      CALL PRPL01(1,'DRPL65   ','           ',DRPL65   )
      CALL PRPL01(1,'DXCCV    ','           ',DXCCV    )
      CALL PRPL01(1,'DXFG     ','           ',DXFG     )
      CALL PRPL01(1,'DXFPI    ','           ',DXFPI    )
      CALL PRPL01(1,'DXFPIL   ','           ',DXFPIL   )
      CALL PRPL01(1,'DXFPP    ','           ',DXFPP    )
      CALL PRPL01(1,'DXFPV    ','           ',DXFPV    )
      CALL PRPL01(1,'DXFPX    ','           ',DXFPX    )
      CALL PRPL01(1,'DXOPI    ','           ',DXOPI    )
      CALL PRPL01(1,'DXOPIL   ','           ',DXOPIL   )
      CALL PRPL01(1,'DXOPP    ','           ',DXOPP    )
      CALL PRPL01(1,'DXOPV    ','           ',DXOPV    )
      CALL PRPL01(1,'EFPI     ','           ',EFPI     )
      CALL PRPL01(1,'EFPIL    ','           ',EFPIL    )
      CALL PRPL01(1,'EFPV     ','           ',EFPV     )
      CALL PRPL01(1,'EOPI     ','           ',EOPI     )
      CALL PRPL01(1,'EOPIL    ','           ',EOPIL    )
      CALL PRPL01(1,'EOPV     ','           ',EOPV     )
      CALL PRPL01(1,'EPL      ','           ',EPL      )
      CALL PRPL01(1,'FPVRL    ','           ',FPVRL    )
      CALL PRPL01(1,'FPVRX    ','           ',FPVRX    )
      CALL PRPL01(1,'GTOC     ','           ',GTOC     )
      CALL PRPL01(1,'I        ','           ',FLOAT(I        ))
      CALL PRPL01(1,'IPRPL    ','           ',FLOAT(IPRPL    ))
      CALL PRPL01(1,'IUPDAT   ','           ',FLOAT(IUPDAT   ))
```

SUBROUTINE CNTL00

```
CALL PRPL01(1,'MFVRL   ','                  ',MFVRL    )
CALL PRPL01(1,'MFVRX   ','                  ',MFVRX    )
CALL PRPL01(1,'MNCYC   ','                  ',FLOAT(MNCYC    ))
CALL PRPL01(1,'MOVRL   ','                  ',MOVRL    )
CALL PRPL01(1,'MOVRX   ','                  ',MOVRX    )
CALL PRPL01(1,'MRCONT  ','                  ',MRCONT   )
CALL PRPL01(1,'MRG     ','                  ',MRG      )
CALL PRPL01(1,'MRG100  ','                  ',MRG100   )
CALL PRPL01(1,'MRG65   ','                  ',MRG65    )
CALL PRPL01(1,'MRREF   ','                  ',MRREF    )
CALL PRPL01(1,'MRREFX  ','                  ',MRREFX   )
CALL PRPL01(1,'OPOVCL  ','                  ',OPOVCL   )
CALL PRPL01(1,'OPOVDL  ','                  ',OPOVDL   )
CALL PRPL01(1,'OPVRL   ','                  ',OPVRL    )
CALL PRPL01(1,'OPVRX   ','                  ',OPVRX    )
CALL PRPL01(1,'PC      ','                  ',PC       )
CALL PRPL01(1,'PCG     ','                  ',PCG      )
CALL PRPL01(1,'PCG100  ','                  ',PCG100   )
CALL PRPL01(1,'PCG50   ','                  ',PCG50    )
CALL PRPL01(1,'PCPCT   ','                  ',PCPCT    )
CALL PRPL01(1,'PCPCTX  ','                  ',PCPCTX   )
CALL PRPL01(1,'PCREF   ','                  ',PCREF    )
CALL PRPL01(1,'PCREFX  ','                  ',PCREFX   )
CALL PRPL01(1,'PCRFXL  ','                  ',PCRFXL   )
CALL PRPL01(1,'PCRL    ','                  ',PCRL     )
CALL PRPL01(1,'PCRLX   ','                  ',PCRLX    )
CALL PRPL01(1,'PCX     ','                  ',PCX      )
CALL PRPL01(1,'PFD1    ','                  ',PFD1     )
CALL PRPL01(1,'PFD1X   ','                  ',PFD1X    )
CALL PRPL01(1,'QFC     ','                  ',QFC      )
CALL PRPL01(1,'QFFM    ','                  ',QFFM     )
CALL PRPL01(1,'QFX     ','                  ',QFX      )
CALL PRPL01(1,'RHOH    ','                  ',RHOH     )
CALL PRPL01(1,'RPL     ','                  ',RPL      )
CALL PRPL01(1,'RPL50   ','                  ',RPL50    )
CALL PRPL01(1,'RPL65   ','                  ',RPL65    )
CALL PRPL01(1,'T       ','                  ',T        )
CALL PRPL01(1,'TFP1    ','                  ',TFP1     )
CALL PRPL01(1,'TFP1X   ','                  ',TFP1X    )
CALL PRPL01(1,'TFP1XX  ','                  ',TFP1XX   )
CALL PRPL01(1,'TIME    ','                  ',TIME     )
CALL PRPL01(1,'TNEXT   ','                  ',TNEXT    )
CALL PRPL01(1,'TT      ','                  ',TT       )
CALL PRPL01(1,'WH      ','                  ',WH       )
CALL PRPL01(1,'WO      ','                  ',WO       )
CALL PRPL01(1,'XCCVC   ','                  ',XCCVC    )
CALL PRPL01(1,'XCCVX   ','                  ',XCCVX    )
CALL PRPL01(1,'XCCVXL  ','                  ',XCCVXL   )
CALL PRPL01(1,'XFG     ','                  ',XFG      )
CALL PRPL01(1,'XFG100  ','                  ',XFG100   )
```

```
----------------------------------------------------------------
                       SUBROUTINE CNTL00
----------------------------------------------------------------

      CALL PRPL01(1,'XFG50   ','        ',XFG50   )
      CALL PRPL01(1,'XFPIG   ','        ',XFPIG   )
      CALL PRPL01(1,'XFPPG   ','        ',XFPPG   )
      CALL PRPL01(1,'XFPVC   ','        ',XFPVC   )
      CALL PRPL01(1,'XFPVST  ','        ',XFPVST  )
      CALL PRPL01(1,'XFPVX   ','        ',XFPVX   )
      CALL PRPL01(1,'XFPVXL  ','        ',XFPVXL  )
      CALL PRPL01(1,'XMFVC   ','        ',XMFVC   )
      CALL PRPL01(1,'XMFVX   ','        ',XMFVX   )
      CALL PRPL01(1,'XMFVXL  ','        ',XMFVXL  )
      CALL PRPL01(1,'XMOVC   ','        ',XMOVC   )
      CALL PRPL01(1,'XMOVX   ','        ',XMOVX   )
      CALL PRPL01(1,'XMOVXL  ','        ',XMOVXL  )
      CALL PRPL01(1,'XOPDCO  ','        ',XOPDCO  )
      CALL PRPL01(1,'XOPIG   ','        ',XOPIG   )
      CALL PRPL01(1,'XOPPG   ','        ',XOPPG   )
      CALL PRPL01(1,'XOPVC   ','        ',XOPVC   )
      CALL PRPL01(1,'XOPVMX  ','        ',XOPVMX  )
      CALL PRPL01(1,'XOPVS   ','        ',XOPVS   )
      CALL PRPL01(1,'XOPVST  ','        ',XOPVST  )
      CALL PRPL01(1,'XOPVX   ','        ',XOPVX   )
      CALL PRPL01(1,'XOPVXL  ','        ',XOPVXL  )
      CALL PRPL01(1,'XOVPS   ','        ',XOVPS   )
 99   RETURN
      END
```

Appendix D
TTBE Model Configration Input

The detail TTBE model described in this report was delivered to NASA-MSFC. Presented in this appendix is the listing of the configuration input which the ROCETS system interprets to generate the TTBE simulation without the control model.

CONFIG. INPUT

```
*************************************************************************
*   R O C E T S   C O N F I G U R A T I O N   I N P U T   F I L E     *
*-----------------------------------------------------------------------*
*                                                                       *
* MODEL: TTBE                                                           *
*                                                                       *
* CONFIGURATION FILE : CTTBE002                                         *
*                                                                       *
* PURPOSE : USED TO GENERATE TTBE WITHOUT CONTROL                       *
*                                                                       *
* DATE: 08/15/90                                                        *
*                                                                       *
* ENGINEER: T.F. DENMAN, PRATT & WHITNEY                                *
*                                                                       *
*************************************************************************
**********************************************
*   DEFINE CONFIGRUATION PROCESSOR OPTIONS   *
**********************************************
 DEFINE OPTIONS
   UNITS : ENGLISH;
   TITLE : TTBE WITHOUT THE CONTROL - GENERATED FROM CTTBE002;
   CROSS : ON ;
   PDS   : HDHS206.ROCETS.DATADICT;
   MAXIMUM :     SYMXSCHD  -  20     ,
                 SYMXSCVL  -  5000   ;
 END    OPTIONS
***************************************************************
*   THE MAXIMUMS ON THE FOLLOWING VARIABLES CAN BE ALTERED    *
*     BY USING THE MAXIMUM KEYWORD IN THE OPTIONS BLOCK       *
***************************************************************
*  MAXIMUM :     SYMXNAME  -  10000  ,
*                SYMXSCHD  -     20  ,
*                SYMXSCVL  -   5000  ,
*                SYMXLIN   -    100  ,
*                SYMXLOUT  -     50  ,
*                SYMXSTAT  -    200  ,
*                SYMXBAL   -     50  ,
*                SYMXITMP  -  20000  ,
*                SYMXRIMP  -  20000  ,
*                SYMXCTMP  -  20000  ,
*                SYMXDTMP  -  20000  ,
*                SYMXPRNT  -  10000  ,
*                SYMXBLOK  -   5000  ,
*                SYMXCOL   -     10  ,
*                SYMXDISC  -    500  ;
```

CONFIG. INPUT

```
************************************************
*   DEFINE MODULES FROM OUTSIDE LIBRARY       *
************************************************
*
* DEFINE INSTREAM
*
*  FOR EXAMPLE :
*
*  PIPE10 : HDHS206.PIPE.FORTRAN ;
*
* END    INSTREAM
*
************************************************
*   DEFINE ADDITIONAL EXTERNAL INPUTS         *
************************************************
 DEFINE EXTERNALS
   PRIMMOI ,
   PRIMFPB ,
   PRIMOPB ,
   PCREQ   ;
 END    EXTERNALS
************************************************
*   CHANGE ITERATION VARIABLES FOR STATES     *
************************************************
 DEFINE INTEGRATION
   ITERATE: HTPBSF    FOR UTPBSF   ;
   ITERATE: HTPBSO    FOR UTPBSO   ;
   ITERATE: HTVL1     FOR UTVL1    ;
   ITERATE: HTVL10    FOR UTVL10   ;
   ITERATE: HTVL11    FOR UTVL11   ;
   ITERATE: HTVL12    FOR UTVL12   ;
   ITERATE: HTVL13    FOR UTVL13   ;
   ITERATE: HTVL14    FOR UTVL14   ;
   ITERATE: HTVL15    FOR UTVL15   ;
   ITERATE: HTVL16    FOR UTVL16   ;
   ITERATE: HTVL17    FOR UTVL17   ;
   ITERATE: HTVL18    FOR UTVL18   ;
   ITERATE: HTVL19    FOR UTVL19   ;
   ITERATE: HTVL2     FOR UTVL2    ;
   ITERATE: HTVL20    FOR UTVL20   ;
   ITERATE: HTVL21    FOR UTVL21   ;
   ITERATE: HTVL22    FOR UTVL22   ;
   ITERATE: HTVL3     FOR UTVL3    ;
   ITERATE: HTVL4     FOR UTVL4    ;
   ITERATE: HTVL5     FOR UTVL5    ;
```

CONFIG. INPUT

```
  ITERATE: HTVL6    FOR UTVL6    ;
  ITERATE: HTVL7    FOR UTVL7    ;
  ITERATE: HTVL8    FOR UTVL8    ;
  ITERATE: HTVL9    FOR UTVL9    ;
  ITERATE: PTPBSF   FOR RHOPBSF  ;
  ITERATE: PTPBSO   FOR RHOPBSO  ;
  ITERATE: PTVL1    FOR RHOVL1   ;
  ITERATE: PTVL10   FOR RHOVL10  ;
  ITERATE: PTVL11   FOR RHOVL11  ;
  ITERATE: PTVL12   FOR RHOVL12  ;
  ITERATE: PTVL13   FOR RHOVL13  ;
  ITERATE: PTVL14   FOR RHOVL14  ;
  ITERATE: PTVL15   FOR RHOVL15  ;
  ITERATE: PTVL16   FOR RHOVL16  ;
  ITERATE: PTVL17   FOR RHOVL17  ;
  ITERATE: PTVL18   FOR RHOVL18  ;
  ITERATE: PTVL19   FOR RHOVL19  ;
  ITERATE: PTVL2    FOR RHOVL2   ;
  ITERATE: PTVL20   FOR RHOVL20  ;
  ITERATE: PTVL21   FOR RHOVL21  ;
  ITERATE: PTVL22   FOR RHOVL22  ;
  ITERATE: PTVL3    FOR RHOVL3   ;
  ITERATE: PTVL4    FOR RHOVL4   ;
  ITERATE: PTVL5    FOR RHOVL5   ;
  ITERATE: PTVL6    FOR RHOVL6   ;
  ITERATE: PTVL7    FOR RHOVL7   ;
  ITERATE: PTVL8    FOR RHOVL8   ;
  ITERATE: PTVL9    FOR RHOVL9   ;
 END   INTEGRATION
***********************
*  SET-UP BALANCES   *
***********************
 DEFINE BALANCES
  BALANCE WCCVBAL  : WCCV    UNTIL PTVL9    - PTVL9C    ;
  BALANCE WF9BAL   : WF9     UNTIL PTVL10   - SYBL0001  ;
  BALANCE WF10BAL  : WF10    UNTIL PTVL11   - SYBL0002  ;
  BALANCE WHG2BAL  : WHG2    UNTIL PTOPRB   - SYBL0003  ;
  BALANCE THTODBAL : TTHTOD  UNTIL TTHTOD   - SYBL0004  ;
  BALANCE WHG6BAL  : WHG6    UNTIL PTOSF    - SYBL0005  ;
  BALANCE WHG1BAL  : WHG1    UNTIL PTFPRB   - SYBL0006  ;
  BALANCE THTFDBAL : TTHTFD  UNTIL TTHTFD   - SYBL0007  ;
  BALANCE WHG5BAL  : WHG5    UNTIL PTFSF    - SYBL0008  ;
  BALANCE WFINJBAL : WFINJ   UNTIL PTMFI    - SYBL0009  ;
  BALANCE WHPFPBAL : WHPFP   UNTIL PTVL3    - PTHPFD    ;
  BALANCE WHPOPBAL : WHPOP   UNTIL PTVL21   - PTHPOD    ;
```

CONFIG. INPUT

```
  BALANCE WLPFPBAL : WLPFP    UNTIL PTVL1    - PTLPFD    ;
  BALANCE WLPOPBAL : WLPOP    UNTIL PTVL19   - PTLPOD    ;
  BALANCE WPRBPBAL : WPRBP    UNTIL PTPBSO   - PTPBPD    ;
 END    BALANCES
************************************************************************
************************************************************************
*                                                                      *
*                        DEFINE CONFIGURATION                          *
*                                                                      *
************************************************************************
************************************************************************
*----------------------------------------------------------------------*
*                                                                      *
*                       ABOVE THE ITERATION LOOP                       *
*                                                                      *
*----------------------------------------------------------------------*
*
*DEFINE SYSTEM ABOVE
*
*  ABOVE THE ITERATION LOOP MODULES AND EQUATIONS GO HERE
*
*END SYSTEM ABOVE
*
*----------------------------------------------------------------------*
*                                                                      *
*                       INSIDE THE ITERATION LOOP                      *
*                                                                      *
*----------------------------------------------------------------------*
*
 DEFINE SYSTEM INSIDE
*
***********************************************************************
* -------------------- HYDROGEN PROPERTIES -------------------------- *
***********************************************************************
  PROPERTY PACKAGE: H2PROP;
     LOCATION HINK : RHO-F(PT,HT), TT-F(PT,HT);
     LOCATION VL1  : RHO-F(PT,HT), TT-F(PT,HT);
     LOCATION VL2  : RHO-F(PT,HT), TT-F(PT,HT);
     LOCATION VL3  : RHO-F(PT,HT), TT-F(PT,HT), GAMA-F(HT,PT);
     LOCATION VL4  : RHO-F(PT,HT), TT-F(PT,HT);
     LOCATION VL5  : RHO-F(PT,HT), TT-F(PT,HT);
     LOCATION VL6  : RHO-F(PT,HT), TT-F(PT,HT),
                     CP-F(HT,PT), MU-F(HT,PT),    K-F(HT,PT);
     LOCATION VL7  : RHO-F(PT,HT), TT-F(PT,HT),
                     CP-F(HT,PT), MU-F(HT,PT),    K-F(HT,PT);
```

CONFIG. INPUT

```
    LOCATION VL8  : RHO-F(PT,HT), TT-F(PT,HT);
    LOCATION VL9  : RHO-F(PT,HT), TT-F(PT,HT);
    LOCATION VL10 : RHO-F(PT,HT), TT-F(PT,HT);
    LOCATION VL11 : RHO-F(PT,HT), TT-F(PT,HT),
                     CP-F(HT,PT), MU-F(HT,PT),     K-F(HT,PT);
    LOCATION VL12 : RHO-F(PT,HT), TT-F(PT,HT),     S-F(HT,PT),
                     CP-F(HT,PT), MU-F(HT,PT),     K-F(HT,PT);
    LOCATION VL13 : RHO-F(PT,HT), TT-F(PT,HT);
    LOCATION VL14 : RHO-F(PT,HT), TT-F(PT,HT);
    LOCATION VL15 : RHO-F(PT,HT), TT-F(PT,HT);
    LOCATION VL16 : RHO-F(PT,HT), TT-F(PT,HT), GAMA-F(HT,PT);
    LOCATION PBSF : RHO-F(PT,HT), TT-F(PT,HT), GAMA-F(HT,PT);
  END PROPERTY
**********************************************************************
* ----------------------- OXYGEN PROPERTIES ------------------------ *
**********************************************************************
  PROPERTY PACKAGE: O2PROP;
    LOCATION OTNK : RHO-F(PT,HT), TT-F(PT,HT);
    LOCATION VL17 : RHO-F(PT,HT), TT-F(PT,HT);
    LOCATION VL18 : RHO-F(PT,HT), TT-F(PT,HT);
    LOCATION VL19 : RHO-F(PT,HT), TT-F(PT,HT);
    LOCATION VL20 : RHO-F(PT,HT), TT-F(PT,HT);
    LOCATION VL21 : RHO-F(PT,HT), TT-F(PT,HT);
    LOCATION VL22 : RHO-F(PT,HT), TT-F(PT,HT), S-F(HT,PT);
    LOCATION PBSO : RHO-F(PT,HT), TT-F(PT,HT);
  END PROPERTY
**********************************************************************
* --------------------- HOT GAS PROPERTIES ------------------------- *
**********************************************************************
  EQUATION: PTHTFD   =  PTFSF     ;
  EQUATION: OFRHTFD  =  OFRFTBP   ;
  EQUATION: HFRHTFD  =  HFRFTBP   ;
  EQUATION: PTHTOD   =  PTOSF     ;
  EQUATION: OFRHTOD  =  OFROTBP   ;
  EQUATION: HFRHTOD  =  HFROTBP   ;
  EQUATION: OFRVL16  =  0.0       ;
  EQUATION: HFRVL16  =  0.0       ;
  EQUATION: RGASAMB  =  640.0     ;
  EQUATION: GAMAAMB  =  1.4       ;
  EQUATION: OFRAMB   =  0.0       ;
  EQUATION: HFRAMB   =  0.0       ;
  EQUATION: OFRPBSF  =  0.0       ;
  EQUATION: HFRPBSF  =  0.0       ;
*
  PROPERTY PACKAGE: HGPROP;
```

CONFIG. INPUT

```
    LOCATION MCHB :   K-F(PT,TT), MU-F(PT,TT),RHO-F(PT,TT),GAMA-F(PT,TT),
                      R-F(PT,TT), CP-F(PT,TT);
    LOCATION FPRB :GAMA-F(PT,TT),  R-F(PT,TT), CP-F(PT,TT), RHO-F(PT,TT);
    LOCATION OPRB :GAMA-F(PT,TT),  R-F(PT,TT), CP-F(PT,TT), RHO-F(PT,TT);
    LOCATION FTBP :GAMA-F(PT,TT),  R-F(PT,TT), CP-F(PT,TT),    Z-F(PT,TT),
                   RHO-F(PT,TT);
    LOCATION OTBP :GAMA-F(PT,TT),  R-F(PT,TT), CP-F(PT,TT),    Z-F(PT,TT),
                   RHO-F(PT,TT);
    LOCATION HTFD :  CP-F(PT,TT),  GAMA-F(PT,TT);
    LOCATION HTOD :  CP-F(PT,TT),  GAMA-F(PT,TT);
    LOCATION FSF  :GAMA-F(PT,TT),  R-F(PT,TT), CP-F(PT,TT), RHO-F(PT,TT);
    LOCATION OSF  :GAMA-F(PT,TT),  R-F(PT,TT), CP-F(PT,TT), RHO-F(PT,TT);
    LOCATION MFI  :GAMA-F(PT,TT),  R-F(PT,TT), CP-F(PT,TT), RHO-F(PT,TT);
    END PROPERTY
*************************************************************************
* ------------------- EQUATIONS FOR PROPERTIES ------------------------ *
*************************************************************************
*
* INTERNAL ENERGIES - FUEL SIDE
    EQUATION: UTHTNK  -  HTHTNK - (1./RJ) * PTHTNK  /  RHOHTNK ;
    EQUATION: UTVL1   -  HTVL1  - (1./RJ) * PTVL1   /  RHOVL1  ;
    EQUATION: UTVL2   -  HTVL2  - (1./RJ) * PTVL2   /  RHOVL2  ;
    EQUATION: UTVL3   -  HTVL3  - (1./RJ) * PTVL3   /  RHOVL3  ;
    EQUATION: UTVL4   -  HTVL4  - (1./RJ) * PTVL4   /  RHOVL4  ;
    EQUATION: UTVL5   -  HTVL5  - (1./RJ) * PTVL5   /  RHOVL5  ;
    EQUATION: UTVL6   -  HTVL6  - (1./RJ) * PTVL6   /  RHOVL6  ;
    EQUATION: UTVL7   -  HTVL7  - (1./RJ) * PTVL7   /  RHOVL7  ;
    EQUATION: UTVL8   -  HTVL8  - (1./RJ) * PTVL8   /  RHOVL8  ;
    EQUATION: UTVL9   -  HTVL9  - (1./RJ) * PTVL9   /  RHOVL9  ;
    EQUATION: UTVL10  -  HTVL10 - (1./RJ) * PTVL10  /  RHOVL10 ;
    EQUATION: UTVL11  -  HTVL11 - (1./RJ) * PTVL11  /  RHOVL11 ;
    EQUATION: UTVL12  -  HTVL12 - (1./RJ) * PTVL12  /  RHOVL12 ;
    EQUATION: UTVL13  -  HTVL13 - (1./RJ) * PTVL13  /  RHOVL13 ;
    EQUATION: UTVL14  -  HTVL14 - (1./RJ) * PTVL14  /  RHOVL14 ;
    EQUATION: UTVL15  -  HTVL15 - (1./RJ) * PTVL15  /  RHOVL15 ;
    EQUATION: UTVL16  -  HTVL16 - (1./RJ) * PTVL16  /  RHOVL16 ;
    EQUATION: UTPBSF  -  HTPBSF - (1./RJ) * PTPBSF  /  RHOPBSF ;
* INTERNAL ENERGIES - OXYGEN SIDE
    EQUATION: UTOTNK  -  HTOTNK - (1./RJ) * PTOTNK  /  RHOOTNK   ;
    EQUATION: UTVL17  -  HTVL17 - (1./RJ) * PTVL17  /  RHOVL17   ;
    EQUATION: UTVL18  -  HTVL18 - (1./RJ) * PTVL18  /  RHOVL18   ;
    EQUATION: UTVL19  -  HTVL19 - (1./RJ) * PTVL19  /  RHOVL19   ;
    EQUATION: UTVL20  -  HTVL20 - (1./RJ) * PTVL20  /  RHOVL20   ;
    EQUATION: UTVL21  -  HTVL21 - (1./RJ) * PTVL21  /  RHOVL21   ;
    EQUATION: UTVL22  -  HTVL22 - (1./RJ) * PTVL22  /  RHOVL22   ;
```

CONFIG. INPUT

```
  EQUATION: UTPBSO  =  HTPBSO - (1./RJ) * PTPBSO  /  RHOPBSO  ;
* ENTHALPIES - HOT GAS SIDE
  EQUATION: HTFPRB  -  CPFPRB * TTFPRB  ;
  EQUATION: HTOPRB  -  CPOPRB * TTOPRB  ;
  EQUATION: HTFTBP  -  CPFTBP * TTFTBP  ;
  EQUATION: HTOTBP  -  CPOTBP * TTOTBP  ;
  EQUATION: HTFSF   -  CPFSF  * TTFSF   ;
  EQUATION: HTOSF   -  CPOSF  * TTOSF   ;
  EQUATION: HTMFI   -  CPMFI  * TTMFI   ;
  EQUATION: HTMCHB  -  CPMCHB * TTMCHB  ;
  EQUATION: HTHTFD  -  CPHTFD * TTHTFD  ;
  EQUATION: HTHTOD  -  CPHTOD * TTHTOD  ;
***********************************************************************
* ---------------------- FUEL SIDE NON-DERIVATIVE MODULES ------------ *
***********************************************************************
*
  EQUATION : AREAFPOV - (.337998 / .32236 ) * AREAFPV ;
  EQUATION : AREAOPOV - (.110886 / .11835 ) * AREAOPV ;
*****************************
* --- LPFP EXIT DENSITY --- *
*****************************
  EQUATION: RHOLPFD - RHOVL1;
**********************************
* --- LOW PRESSURE FUEL PUMP --- *
**********************************
  MODULE: PUMP01;
    NAME: LPFP;
    I/O LIST: INLET FLOW       - LPFP,
              INLET PROPERTIES - HTNK,
              EXIT  PROPERTIES - LPFD,
              SHAFT            - FL  ;
    DESIGN VALUES: SND     -  15603.4,
                   TRQD    -  12924.8,
                   WD      -    148.7,
                   HDD     - 101574.6,
                   GEAR    -      1.0;
    MAP: PMAP05;
    CMT: LOW PRESSURE FUEL PUMP;
  END MODULE
*****************************
* --- HPFP EXIT DENSITY --- *
*****************************
  EQUATION: RHOHPFD - RHOVL3;
***********************************
* --- HIGH PRESSURE FUEL PUMP --- *
```

CONFIG. INPUT

```
***********************************
   MODULE: PUMP01;
      NAME: HPFP;
      I/O LIST: INLET FLOW       - HPFP,
                INLET PROPERTIES - VL2 ,
                EXIT  PROPERTIES - HPFD,
                SHAFT            - FH  ;
      DESIGN VALUES: SND  -   34189.8,
                     TRQD -  110141.9,
                     WD   -     148.7,
                     HDD  - 2229273.6,
                     GEAR -       1.0;
      MAP: PMAP04;
      CMT: HIGH PRESSURE FUEL PUMP;
   END MODULE
**************************************************
* —— NON-INERTIAL FUEL TURBINE COOLING LINE —— *
**************************************************
   MODULE: PIPE01;
      NAME: FTC ;
      I/O LIST: INLET PROPERTIES - VL3 ,
                EXIT PROPERTIES  - FSF ;
      DESIGN VALUES: CF   - 2.025;
      CMT: FUEL TURBINE COOLING FLOW FROM VOLUME 3 TO VOLUME FSF;
   END MODULE
***************************
* —— MAIN FUEL VALVE —— *
***************************
   MODULE: VALV00;
      NAME: MFV ;
      I/O LIST: UPSTREAM PROPERTY   - VL3 ,
                DOWNSTREAM PROPERTY - VL4 ;
      DESIGN VALUES: AREA - 15.35313,
                     RKLS - 2.289;
      CMT: MAIN FUEL VALVE;
   END MODULE
******************************************
* —— FUEL IGNITER NON-INERTIAL LINE —— *
******************************************
   MODULE: PIPE01;
      NAME: FIG ;
      I/O LIST: INLET PROPERTIES - VL4 ,
                EXIT PROPERTIES  - MCHB;
      DESIGN VALUES: CF   -  0.44;
      CMT: FUEL IGNITER FLOW FROM VOLUME 4 TO THE MAIN CHAMBER;
```

CONFIG. INPUT

```
  END MODULE
******************************************
* ---- NON-INERTIAL FUEL LINE THREE ---- *
******************************************
  MODULE: PIPE01;
     NAME: F3  ;
     I/O LIST: INLET PROPERTIES - VL5 ,
               EXIT PROPERTIES  - VL6 ;
     DESIGN VALUES: CF   - 195.6;
     CMT: PIPE FLOW FROM VOLUME 5 TO VOLUME 6;
  END MODULE
*****************************************
* ---- NON-INERTIAL FUEL LINE FOUR ---- *
*****************************************
  MODULE: PIPE01;
     NAME: F4  ;
     I/O LIST: INLET PROPERTIES - VL6 ,
               EXIT PROPERTIES  - VL7 ;
     DESIGN VALUES: CF   - 133.6;
     CMT: PIPE FLOW FROM VOLUME 6 TO VOLUME 7;
  END MODULE
***********************************
* ---- COOLANT CONTROL VALVE ---- *
***********************************
  EQUATION : RHOVL9C - RHOVL9 ;
  MODULE: PIPE04;
     NAME: CCV ;
     I/O LIST: UPSTREAM PROPERTY   - VL9C,
               DOWNSTREAM PROPERTY - VL8 ;
     DESIGN VALUES: RKLS - 1.763;
     CMT: COOLANT CONTROL VALVE;
  END MODULE
*****************************************
* ---- NON-INERTIAL FUEL LINE NINE ---- *
*****************************************
  MODULE: PIPE06;
     NAME: F9  ;
     I/O LIST: INLET PROPERTIES - VL10,
               EXIT PROPERTIES  - VL11;
     DESIGN VALUES: CF   - 43.2;
     CMT: UPSTREAM PRESSURE CALC FROM VOLUME 11 TO VOLUME 10;
  END MODULE
****************************************
* ---- NON-INERTIAL FUEL LINE TEN ---- *
****************************************
```

CONFIG. INPUT

```
   MODULE: PIPE06;
      NAME: F10 ;
      I/O LIST: INLET PROPERTIES = VL11,
                EXIT PROPERTIES  = VL12;
      DESIGN VALUES: CF    = 29.3;
      CMT: UPSTREAM PRESSURE CALC FROM VOLUME 12 TO VOLUME 11;
   END MODULE
*********************************************************
* —— LOW PRESSURE FUEL TURBINE DISCHARGE PRESSURE —— *
*********************************************************
   EQUATION: PTLTFD = PTVL13;
**************************************
* —— LOW PRESSURE FUEL TURBINE —— *
**************************************
   MODULE: TURB02;
      NAME: LPFT;
      I/O LIST: INLET PROPERTIES = VL12,
                EXIT  PROPERTIES = LTFD,
                SHAFT WORK       = FL  ;
      DESIGN VALUES: ETAD = 0.53,
                     PRD  = 1.33,
                     SND  = 15603.4 ,
                     AREA = 1.0    ,
                     DIAM = 6.63   ,
                     DC1     =     0  ,
                     DC2     =     0  ,
                     GEAR    =     1.0 ;
      MAP: TBMP04;
      CMT: LOW PRESSURE FUEL TURBINE;
   END MODULE
******************************************
* —— NON-INERTIAL FUEL LINE ELEVEN —— *
******************************************
   MODULE: PIPE01;
      NAME: F11 ;
      I/O LIST: INLET PROPERTIES = VL13,
                EXIT PROPERTIES  = VL14;
      DESIGN VALUES: CF    = 105.6;
      CMT: PIPE FLOW FROM VOLUME 13 TO VOLUME 14;
   END MODULE
******************************************
* —— NON-INERTIAL FUEL LINE TWELVE —— *
******************************************
   MODULE: PIPE01;
      NAME: F12 ;
```

CONFIG. INPUT

```
      I/O LIST: INLET PROPERTIES - VL14,
                EXIT PROPERTIES  - VL16;
      DESIGN VALUES: CF   - 147.4;
      CMT: PIPE FLOW FROM VOLUME 14 TO VOLUME 16;
   END MODULE
********************************************
* —— NON-INERTIAL FUEL LINE THIRTEEN —— *
********************************************
   MODULE: PIPE01;
      NAME: F13 ;
      I/O LIST: INLET PROPERTIES - VL13,
                EXIT PROPERTIES  - VL15;
      DESIGN VALUES: CF   - 143.5;
      CMT: PIPE FLOW FROM VOLUME 13 TO VOLUME 15;
   END MODULE
********************************************
* —— NON-INERTIAL FUEL LINE FOURTEEN —— *
********************************************
   MODULE: PIPE01;
      NAME: F14 ;
      I/O LIST: INLET PROPERTIES - VL15,
                EXIT PROPERTIES  - VL16;
      DESIGN VALUES: CF   - 199.4;
      CMT: PIPE FLOW FROM VOLUME 15 TO VOLUME 16;
   END MODULE
*******************************************
* —— NON-INERTIAL FUEL LINE FIFTEEN —— *
*******************************************
   MODULE: PIPE01;
      NAME: F15 ;
      I/O LIST: INLET PROPERTIES - VL16,
                EXIT PROPERTIES  - MCHB;
      DESIGN VALUES: CF   -  76.9;
      CMT: PIPE FLOW FROM VOLUME 16 TO THE MAIN CHAMBER;
   END MODULE
*********************************************************
* —— NON-INERTIAL FUEL LINE TO MAIN FUEL INJECTOR —— *
*********************************************************
   MODULE: PIPE01;
      NAME: FSLV;
      I/O LIST: INLET PROPERTIES - VL16,
                EXIT PROPERTIES  - MFI ;
      DESIGN VALUES: CF   -  28.2;
      CMT: PIPE FLOW FROM VOLUME 16 TO THE MAIN FUEL INJECTOR;
   END MODULE
```

CONFIG. INPUT

```
*****************************************************
* —— NON-INERTIAL OXIDIZER TURBINE COOLING LINE —— *
*****************************************************
  MODULE: PIPE01;
     NAME: OTC ;
     I/O LIST: INLET PROPERTIES - PBSF,
               EXIT PROPERTIES  - OSF ;
     DESIGN VALUES: CF   - 0.661;
     CMT: OXYGEN TURBINE COOLING FLOW FROM VOLUME PBSF TO VOLUME FSF;
  END MODULE
* MANIFOLD COOLING HEAT TRANSFER
  EQUATION: TKMCO   - .00167  ;
  EQUATION: TKMCF   - .00131  ;
  EQUATION: AHTMCO  -  728.0  ;
  EQUATION: AHTMCF  -  872.0  ;
  EQUATION: QDOTFMCI -TKMCO*AHTMCO*(TTMFI-TTVL13)*
  SQRT(AMAX1(0.,(WF13/11.36)));
  EQUATION: QDOTFMCO - - QDOTFMCI ;
  EQUATION: QDOTOMCI -TKMCF*AHTMCF*(TTMFI-TTVL13)*
  SQRT(AMAX1(0.,(WF11/15.50)));
  EQUATION: QDOTOMCO - - QDOTOMCI ;
***********************************************************************
* ———————————————————— FUEL SIDE DERIVATIVE MODULES —————————————— *
***********************************************************************
*
*********************************
* —— INERTIAL FUEL LINE ONE —— *
*********************************
  MODULE: PIPE00;
     NAME: F1  ;
     I/O LIST: INLET PROPERTIES - VL1 ,
               EXIT PROPERTIES  - VL2 ;
     DESIGN VALUES: CF   - 699.2,
                    AREA - 28.26,
                    XLEN - 20.00;
     CMT: FLOW DERIVATIVE FROM VOLUME 1 TO VOLUME 2;
  END MODULE
*********************************
* —— INERTIAL FUEL LINE TWO —— *
*********************************
  MODULE: PIPE00;
     NAME: F2  ;
     I/O LIST: INLET PROPERTIES - VL4 ,
               EXIT PROPERTIES  - VL5 ;
     DESIGN VALUES: CF   - 190.1,
```

CONFIG. INPUT

```
                    AREA - 10.60,
                    XLEN - 17.50;
      CMT: FLOW DERIVATIVE FROM VOLUME 4 TO VOLUME 5;
   END MODULE
***********************************
* —— INERTIAL FUEL LINE SIX —— *
***********************************
   MODULE: PIPE00;
      NAME: F6  ;
      I/O LIST: INLET PROPERTIES - VL4 ,
                EXIT PROPERTIES  - VL9 ;
      DESIGN VALUES: CF   -  79.6,
                     AREA -  1.00,
                     XLEN -  4.027;
      CMT: FLOW DERIVATIVE FROM VOLUME 4 TO VOLUME 9;
   END MODULE
*************************************
* —— INERTIAL FUEL LINE EIGHT —— *
*************************************
   MODULE: PIPE00;
      NAME: F8  ;
      I/O LIST: INLET PROPERTIES - VL4 ,
                EXIT PROPERTIES  - VL10;
      DESIGN VALUES: CF   -  65.2,
                     AREA -  3.14,
                     XLEN -134.00;
      CMT: FLOW DERIVATIVE FROM VOLUME 4 TO VOLUME 10;
   END MODULE
************************************
* —— INERTIAL FUEL LINE FIVE —— *
************************************
   MODULE: PIPE00;
      NAME: F5  ;
      I/O LIST: INLET PROPERTIES - VL7 ,
                EXIT PROPERTIES  - VL8 ;
      DESIGN VALUES: CF   - 142.3,
                     AREA - 17.56,
                     XLEN - 92.00;
      CMT: FLOW DERIVATIVE FROM VOLUME 7 TO VOLUME 8;
   END MODULE
*************************************
* —— INERTIAL FUEL LINE SEVEN —— *
*************************************
   MODULE: PIPE00;
      NAME: F7  ;
```

CONFIG. INPUT

```
      I/O LIST: INLET PROPERTIES - VL8 ,
                EXIT PROPERTIES  - PBSF;
      DESIGN VALUES: CF   - 682.1,
                     AREA -   4.73,
                     XLEN - 77.50;
      CMT: FLOW DERIVATIVE FROM VOLUME 8 TO VOLUME PBSF;
   END MODULE
*****************************************************
* —— INERTIAL FUEL LINE TO THE FUEL PREBURNER —— *
*****************************************************
   MODULE: PIPE00;
      NAME: FFPB;
      I/O LIST: INLET PROPERTIES - PBSF,
                EXIT PROPERTIES  - FPRB;
      DESIGN VALUES: CF   - 124.0,
                     AREA -  1.00,
                     XLEN - 50.00;
      CMT: FLOW DERIVATIVE FROM VOLUME PBSF TO VOLUME FPRB;
   END MODULE
*********************************************************
* —— INERTIAL FUEL LINE TO THE OXIDIZER PREBURNER —— *
*********************************************************
   MODULE: PIPE00;
      NAME: FOPB;
      I/O LIST: INLET PROPERTIES - PBSF,
                EXIT PROPERTIES  - OPRB;
      DESIGN VALUES: CF   -  60.5,
                     AREA -  1.00,
                     XLEN - 50.00;
      CMT: FLOW DERIVATIVE FROM VOLUME PBSF TO VOLUME OPRB;
   END MODULE
**********************
* —— VOLUME ONE —— *
**********************
   MODULE: VOLM00;
      NAME: VL1 ;
      I/O LIST: UPSTREAM PROPERTIES    - LPFD,
                INLET FLOW             - LPFP,
                EXIT FLOW              - F1   ,
                DOWNSTREAM PROPERTIES - VL2  ,
                QDOT                   - VL1  ;
      DESIGN VALUES: VOL  - 365.2,
                     QDOT - 0.0  ;
      CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 1;
   END MODULE
```

CONFIG. INPUT

```
***********************
* ---- VOLUME TWO ---- *
***********************
   MODULE: VOLM00;
      NAME: VL2 ;
      I/O LIST: UPSTREAM PROPERTIES   - VL1 ,
                INLET FLOW            - F1  ,
                EXIT FLOW             - HPFP,
                DOWNSTREAM PROPERTIES - HPFD,
                QDOT                  - VL2 ;
      DESIGN VALUES: VOL  - 200.0,
                     QDOT - 0.0  ;
      CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 2;
   END MODULE
*************************
* ---- VOLUME THREE ---- *
*************************
   MODULE: VOLM01;
      NAME: VL3 ;
      I/O LIST: UPSTREAM PROPERTIES   - HPFD,
                INLET FLOW            - HPFP,
                EXIT FLOW             - MFV , FTC ,
                DOWNSTREAM PROPERTIES - VL4 , FSF ,
                QDOT                  - VL3 ;
      DESIGN VALUES: VOL  - 347.9;
      CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 3;
   END MODULE
************************
* ---- VOLUME FOUR ---- *
************************
   MODULE: VOLM01;
      NAME: VL4 ;
      I/O LIST: UPSTREAM PROPERTIES   - VL3 ,
                INLET FLOW            - MFV ,
                EXIT FLOW             - F2  , F6  , F8  , FIC ,
                DOWNSTREAM PROPERTIES - VL5 , VL9 , VL10, MCHB,
                QDOT                  - VL4 ;
      DESIGN VALUES: VOL  - 186.0;
      CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 4;
   END MODULE
************************
* ---- VOLUME FIVE ---- *
************************
   MODULE: VOLM00;
      NAME: VL5 ;
```

CONFIG. INPUT

```
    I/O LIST: UPSTREAM PROPERTIES   - VL4 ,
              INLET FLOW            - F2  ,
              EXIT FLOW             - F3  ,
              DOWNSTREAM PROPERTIES - VL6 ,
              QDOT                  - VL5 ;
    DESIGN VALUES: VOL  - 866.3,
                   QDOT - 0.0  ;
    CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 5;
  END MODULE
*************************************
* —— NOZZLE COOLING VOLUME SIX —— *
*************************************
  MODULE: NCLV00;
    NAME: VL6 ;
    I/O LIST: UPSTREAM PROP       - VL5 ,
              INLET FLOW          - F3  ,
              EXIT FLOW           - F4  ,
              DOWNSTREAM PROP     - VL7 ,
              QDOT                - 6HOT, 6AMB,
              METAL TEMPERATURES - MTL1, MTL5;
    DESIGN VALUES: VOL  -3160.0,
                   AREA - 17.572;
    CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 6;
  END MODULE
***************************************
* —— NOZZLE COOLING VOLUME SEVEN —— *
***************************************
  MODULE: NCLV00;
    NAME: VL7 ;
    I/O LIST: UPSTREAM PROP       - VL6 ,
              INLET FLOW          - F4  ,
              EXIT FLOW           - F5  ,
              DOWNSTREAM PROP     - VL8 ,
              QDOT                - 7HOT, 7AMB,
              METAL TEMPERATURES - MTL2, MTL6;
    DESIGN VALUES: VOL  -1616.0,
                   AREA - 40.376;
    CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 7;
  END MODULE
************************
* —— VOLUME EIGHT —— *
************************
  MODULE: VOLM01;
    NAME: VL8 ;
    I/O LIST: UPSTREAM PROPERTIES   - VL7 , VL9 ,
```

CONFIG. INPUT

```
                INLET FLOW             - F6  , CCV ,
                EXIT FLOW              - F7  ,
                DOWNSTREAM PROPERTIES - PBSF,
                QDOT                   - VL8 ;
     DESIGN VALUES: VOL  - 1000.;
     CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 8;
   END MODULE
***********************
* —— VOLUME NINE —— *
***********************
   MODULE: VOLM00;
     NAME: VL9 ;
     I/O LIST: UPSTREAM PROPERTIES    - VL4 ,
                INLET FLOW             - F6  ,
                EXIT FLOW              - CCV ,
                DOWNSTREAM PROPERTIES - VL8 ,
                QDOT                   - VL9 ;
     DESIGN VALUES: VOL  - 500.0,
                    QDOT - 0.0  ;
     CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 9;
   END MODULE
**********************
* —— VOLUME TEN —— *
**********************
   MODULE: VOLM00;
     NAME: VL10;
     I/O LIST: UPSTREAM PROPERTIES    - VL4 ,
                INLET FLOW             - F8  ,
                EXIT FLOW              - F9  ,
                DOWNSTREAM PROPERTIES - VL11,
                QDOT                   - VL10;
     DESIGN VALUES: VOL  - 1000.,
                    QDOT - 0.0  ;
     CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 10;
   END MODULE
*****************************************
* —— CHAMBER COOLING VOLUME ELEVEN —— *
*****************************************
   MODULE: NCLV00;
     NAME: VL11;
     I/O LIST: UPSTREAM PROP       - VL10,
                INLET FLOW          - F9  ,
                EXIT FLOW           - F10 ,
                DOWNSTREAM PROP     - VL12,
                QDOT                - 11HT, 11AM,
```

CONFIG. INPUT

```
            METAL TEMPERATURES - MTL3, MTL7;
    DESIGN VALUES: VOL  - 144.2 ,
                   AREA - 10.927;
    CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 11;
  END MODULE
******************************************
* —— CHAMBER COOLING VOLUME TWELVE —— *
******************************************
  MODULE: NCLV00;
    NAME: VL12;
    I/O LIST: UPSTREAM PROP      - VL11,
              INLET FLOW         - F10 ,
              EXIT FLOW          - LPFT,
              DOWNSTREAM PROP    - LTFD,
              QDOT               - 12HT, 12AM,
              METAL TEMPERATURES - MTL4, MTL8;
    DESIGN VALUES: VOL  - 144.2,
                   AREA - 10.927;
    CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 12;
  END MODULE
****************************
* —— VOLUME THIRTEEN —— *
****************************
  MODULE: VOLM01;
    NAME: VL13;
    I/O LIST: UPSTREAM PROPERTIES   - LTFD,
              INLET FLOW            - LPFT,
              EXIT FLOW             - F11 , F13 ,
              DOWNSTREAM PROPERTIES - VL14, VL15,
              QDOT                  - VL13;
    DESIGN VALUES: VOL  - 500.0;
    CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 13;
  END MODULE
****************************
* —— VOLUME FOURTEEN —— *
****************************
  MODULE: VOLM00;
    NAME: VL14;
    I/O LIST: UPSTREAM PROPERTIES   - VL13,
              INLET FLOW            - F11 ,
              EXIT FLOW             - F12 ,
              DOWNSTREAM PROPERTIES - VL16,
              QDOT                  - OMCI;
    DESIGN VALUES: VOL  - 500.0;
    CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 14;
```

CONFIG. INPUT

```
  END MODULE
**************************
* --- VOLUME FIFTEEN --- *
**************************
  MODULE: VOLM00;
    NAME: VL15;
    I/O LIST: UPSTREAM PROPERTIES   - VL13,
              INLET FLOW            - F13 ,
              EXIT FLOW             - F14 ,
              DOWNSTREAM PROPERTIES - VL16,
              QDOT                  - FMCI;
    DESIGN VALUES: VOL  - 500.0;
    CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 15;
  END MODULE
**************************
* --- VOLUME SIXTEEN --- *
**************************
  MODULE: VOLM01;
    NAME: VL16;
    I/O LIST: UPSTREAM PROPERTIES   - VL14, VL15,
              INLET FLOW            - F12 , F14 ,
              EXIT FLOW             - FSLV, F15 ,
              DOWNSTREAM PROPERTIES - MFI , MCHB,
              QDOT                  - VL16;
    DESIGN VALUES: VOL  - 500.0;
    CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 16;
  END MODULE
******************************************
* --- PREBURNER FUEL SPLITTER VOLUME --- *
******************************************
  MODULE: VOLM01;
    NAME: PBSF;
    I/O LIST: UPSTREAM PROPERTIES   - VL8 ,
              INLET FLOW            - F7  ,
              EXIT FLOW             - FFPB, FOPB, OTC ,
              DOWNSTREAM PROPERTIES - FPRB, OPRB, OSF ,
              QDOT                  - PBSF;
    DESIGN VALUES: VOL  - 500.0;
    CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME PBSF;
  END MODULE
***********************************************************************
* --------------------- OXYGEN SIDE NON-DERIVATIVE MODULES ---------- *
***********************************************************************
*
* LPOP EXIT DENSITY
```

CONFIG. INPUT

```
  EQUATION: RHOLPOD = RHOVL19;
****************************************
* —— LOW PRESSURE OXIDIZER PUMP —— *
****************************************
  MODULE: PUMP01;
     NAME: LPOP;
     I/O LIST: INLET FLOW       = LPOP,
               INLET PROPERTIES = VL18,
               EXIT  PROPERTIES = LPOD,
               SHAFT            = OL  ;
     DESIGN VALUES: SND    =   5041.5,
                    TRQD   =  18815.1,
                    WD     =    896.2,
                    HDD    =   7573.6,
                    GEAR   =      1.0;
     MAP: PMAP07;
     CMT: LOW PRESSURE OXYGEN PUMP;
  END MODULE
* HPOP EXIT DENSITY
  EQUATION: RHOHPOD = RHOVL21;
*****************************************
* —— HIGH PRESSURE OXIDIZER PUMP —— *
*****************************************
  MODULE: PUMP01;
     NAME: HPOP;
     I/O LIST: INLET FLOW       = HPOP,
               INLET PROPERTIES = VL20,
               EXIT  PROPERTIES = HPOD,
               SHAFT            = OH  ;
     DESIGN VALUES: SND  =   27240.8,
                    TRQD =   50543.4,
                    WD   =    1078.4,
                    HDD  =   91778.2,
                    GEAR =       1.0;
     MAP: PMAP06;
     CMT: HIGH PRESSURE OXYGEN PUMP;
  END MODULE
*********************************************
* —— NON-INERTIAL OXIDIZER LINE FOUR —— *
*********************************************
  MODULE: PIPE01;
     NAME: O4  ;
     I/O LIST: INLET PROPERTIES = VL21,
               EXIT PROPERTIES  = VL20;
     DESIGN VALUES: CF   = 0.221;
```

CONFIG. INPUT

```
    CMT: HPOP RECIRC. FLOW FROM VOLUME 21 TO VOLUME 20;
  END MODULE
* LINE 7 DOWNSTREAM DENSITY
  EQUATION: RHOPOCO = RHOVL21;
********************************************
* —— NON-INERTIAL OXIDIZER LINE SEVEN —— *
********************************************
  MODULE: PIPE01;
    NAME: 07  ;
    I/O LIST: INLET PROPERTIES = VL21,
              EXIT PROPERTIES  = POGO;
    DESIGN VALUES: CF   = 0.050;
    CMT: LIQUID POGO FLOW FROM VOLUME 21 TO POGO;
  END MODULE
************************************************************
* —— LOW PRESSURE OXIDIZER TURBINE DISCHARGE PRESSURE —— *
************************************************************
  EQUATION: PTLTOD = PTVL19;
*****************************************
* —— LOW PRESSURE OXIDIZER TURBINE —— *
*****************************************
  MODULE: TURB02;
    NAME: LPOT;
    I/O LIST: INLET PROPERTIES = VL22,
              EXIT  PROPERTIES = LTOD,
              SHAFT WORK NODE  = OL  ;
    DESIGN VALUES: ETAD = 0.64,
                   PRD  = 9.73,
                   SND  = 5041.5,
                   AREA = 1.0    ,
                   DIAM = 0.0    ,
                   GEAR = 1.0    ,
                   DC1     =      0  ,
                   DC2     =      0  ;
    MAP: TBMP06;
    CMT: LOW PRESSURE OXYGEN TURBINE;
  END MODULE
*******************************
* —— MAIN OXIDIZER VALVE —— *
*******************************
* LUMP LINE, VALVE, AND INJECTOR RESISTANCES
  EQUATION: RKLSMOV= 0.551;
  EQUATION: CFMOVX = 37.98 * AREAMOV / SQRT(RKLSMOV) ;
  EQUATION: CF09   = 729.5 ;
  EQUATION: CF9OV  = SQRT(CFMOVX**2 * CF09**2 / (CFMOVX**2 + CF09**2));
```

CONFIG. INPUT

```
* SCALE INJECTOR RESISTANCE WITH PRIMING FRACTION
   EQUATION: CFOIJ  = CFOINJ / PRIMMOI ;
   EQUATION: CFMOV  = SQRT(CF9OV**2 * CFOIJ**2 / (CF9OV**2 + CFOIJ**2));
   MODULE: PIPE01;
      NAME: MOV ;
      I/O LIST: UPSTREAM PROPERTY   = VL21,
                DOWNSTREAM PROPERTY = MCHB;
      CMT: MAIN OXIDIZER VALVE;
   END MODULE
   EQUATION: WOINJ  = WMOV * PRIMMOI ;
   EQUATION: TTMOI  = TTVL21 ;
*
* PRBP EXIT DENSITY
*
   EQUATION: RHOPBPD = RHOPBSO;
**************************
* ---- PREBURNER PUMP ---- *
**************************
   MODULE: PUMP01;
      NAME: PRBP;
      I/O LIST: INLET FLOW       = PRBP,
                INLET PROPERTIES = VL21,
                EXIT  PROPERTIES = PBPD,
                SHAFT            = OH  ;
      DESIGN VALUES: SND  =   27240.8,
                     TRQD =    3127.3,
                     WD   =     88.73,
                     HDD  =   74264.7,
                     GEAR =       1.0;
      MAP: PMAP08;
      CMT: PREBURNER OXYGEN PUMP;
   END MODULE
******************************************
* ---- NON-INERTIAL OXIDIZER LINE SIX ---- *
******************************************
   MODULE: PIPE01;
      NAME: O6  ;
      I/O LIST: INLET PROPERTIES = PBSO,
                EXIT PROPERTIES  = VL20;
      DESIGN VALUES: CF   = 0.513;
      CMT: PRBP RECIRC. FLOW FROM VOLUME PBSO TO VOLUME 20;
   END MODULE
*****************************************
* ---- FUEL PREBURNER OXIDIZER VALVE ---- *
*****************************************
```

CONFIG. INPUT

```
* LUMP LINE, VALVE, AND INJECTOR RESISTANCES
  EQUATION: RKLSFPOV - 0.62851;
  EQUATION: CFFPVX - 37.98 * AREAFPOV / SQRT(RKLSFPOV) ;
  EQUATION: CFO11  - 17.156;
  EQUATION: CF11FP - SQRT(CFFPVX**2 * CFO11**2/(CFFPVX**2 + CFO11**2));
* SCALE INJECTOR RESISTANCE WITH PRIMING FRACTION
  EQUATION: CFFPB  - CFOFPB / PRIMFPB ;
  EQUATION: CFFPOV - SQRT(CF11FP**2 * CFFPB**2/(CF11FP**2 + CFFPB**2));
  MODULE: PIPE01;
     NAME: FPOV;
     I/O LIST: UPSTREAM PROPERTY   - PBSO,
               DOWNSTREAM PROPERTY - FPRB;
     CMT: FUEL PREBURNER OXIDIZER VALVE;
  END MODULE
  EQUATION: WOFPB  - WFPOV * PRIMFPB ;
  EQUATION: TTFPBI - TTPBSO ;
*********************************************
* —— OXIDIZER PREBURNER OXIDIZER VALVE —— *
*********************************************
* LUMP LINE, VALVE, AND INJECTOR RESISTANCES
  EQUATION: RKLSOPOV - 0.54173;
  EQUATION: CFOPVX - 37.98 * AREAOPOV / SQRT(RKLSOPOV) ;
  EQUATION: CFO10  - 6.7420;
  EQUATION: CF10OP - SQRT(CFOPVX**2 * CFO10**2/(CFOPVX**2 + CFO10**2));
* SCALE INJECTOR RESISTANCE WITH PRIMING FRACTION
  EQUATION: CFOPB  - CFOOPB / PRIMOPB ;
  EQUATION: CFOPOV - SQRT(CF10OP**2 * CFOPB**2/(CF10OP**2 + CFOPB**2));
  MODULE: PIPE01;
     NAME: OPOV;
     I/O LIST: UPSTREAM PROPERTY   - PBSO,
               DOWNSTREAM PROPERTY - OPRB;
     CMT: OXIDIZER PREBURNER OXIDIZER VALVE;
  END MODULE
  EQUATION: WOOPB  - WOPOV * PRIMOPB ;
  EQUATION: TTOPBI - TTPBSO ;
*********************************************************************
* ———————————————— OXYGEN SIDE DERIVATIVE MODULES ———————————— *
*********************************************************************
*
***************************************
* —— INERTIAL OXIDIZER LINE ONE —— *
***************************************
  MODULE: PIPE03;
     NAME: O1  ;
     I/O LIST: INLET PROPERTIES - OTNK,
```

CONFIG. INPUT

```
               EXIT PROPERTIES  - VL17;
    DESIGN VALUES: CF   - 1049.2,
                   AREA -  11.00,
                   XLEN -  1128.,
                   DLTZ -  1128.;
    CMT: FLOW DERIVATIVE FROM OTNK TO VOLUME 17;
  END MODULE
**************************************
* —— INERTIAL OXIDIZER LINE TWO —— *
**************************************
  MODULE: PIPE00;
    NAME: 02  ;
    I/O LIST: INLET PROPERTIES - VL17,
              EXIT PROPERTIES  - VL18;
    DESIGN VALUES: CF   - 5274.6,
                   AREA -  11.00,
                   XLEN -  300.0;
    CMT: FLOW DERIVATIVE FROM VOLUME 17 TO VOLUME 18;
  END MODULE
****************************************
* —— INERTIAL OXIDIZER LINE THREE —— *
****************************************
  MODULE: PIPE00;
    NAME: 03  ;
    I/O LIST: INLET PROPERTIES - VL19,
              EXIT PROPERTIES  - VL20;
    DESIGN VALUES: CF   - 1262.6,
                   AREA -   1.00,
                   XLEN -  5.330;
    CMT: FLOW DERIVATIVE FROM VOLUME 19 TO VOLUME 20;
  END MODULE
***************************************
* —— INERTIAL OXIDIZER LINE FIVE —— *
***************************************
  MODULE: PIPE00;
    NAME: 05  ;
    I/O LIST: INLET PROPERTIES - VL21,
              EXIT PROPERTIES  - VL22;
    DESIGN VALUES: CF   -  147.4,
                   AREA -   1.00,
                   XLEN -  66.67;
    CMT: FLOW DERIVATIVE FROM VOLUME 21 TO VOLUME 22;
  END MODULE
****************************
* —— VOLUME SEVENTEEN —— *
```

CONFIG. INPUT

```
*****************************
   MODULE: VOLM00;
      NAME: VL17;
      I/O LIST: UPSTREAM PROPERTIES   - OTNK,
                INLET FLOW            - 01  ,
                EXIT FLOW             - 02  ,
                DOWNSTREAM PROPERTIES - VL18,
                QDOT                  - VL17;
      DESIGN VALUES: VOL  - 12408.,
                     QDOT - 0.0   ;
      CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 17;
   END MODULE
****************************
* ---- VOLUME EIGHTEEN ---- *
****************************
   MODULE: VOLM00;
      NAME: VL18;
      I/O LIST: UPSTREAM PROPERTIES   - VL17,
                INLET FLOW            - 02  ,
                EXIT FLOW             - LPOP,
                DOWNSTREAM PROPERTIES - LPOD,
                QDOT                  - VL18;
      DESIGN VALUES: VOL  - 3300.,
                     QDOT - 0.0  ;
      CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 18;
   END MODULE
****************************
* ---- VOLUME NINETEEN ---- *
****************************
   MODULE: VOLM01;
      NAME: VL19;
      I/O LIST: UPSTREAM PROPERTIES   - LPOD, LTOD,
                INLET FLOW            - LPOP, LPOT,
                EXIT FLOW             - 03  ,
                DOWNSTREAM PROPERTIES - VL20,
                QDOT                  - VL19;
      DESIGN VALUES: VOL  -1771.0;
      CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 19;
   END MODULE
**************************
* ---- VOLUME TWENTY ---- *
**************************
   EQUATION : HTPOGO - HTVL20 ;
   MODULE: VOLM01;
      NAME: VL20;
```

CONFIG. INPUT

```
   I/O LIST: UPSTREAM PROPERTIES    - VL19, PBSO, VL21,
             INLET FLOW             - 03  , 06  , 04  ,
             EXIT FLOW              - HPOP, POGO,
             DOWNSTREAM PROPERTIES - HPOD, POGO,
             QDOT                   - VL20;
   DESIGN VALUES: VOL  -4936.0;
   CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 20;
 END MODULE
*****************************
* —— VOLUME TWENTYONE —— *
*****************************
 MODULE: VOLM01;
   NAME: VL21;
   I/O LIST: UPSTREAM PROPERTIES    - HPOD,
             INLET FLOW             - HPOP,
             EXIT FLOW              - 04  , 05  , 07  , MOV , PRBP,
             DOWNSTREAM PROPERTIES - VL20, VL22, POGO, VL21, PBPD,
             QDOT                   - VL21;
   DESIGN VALUES: VOL  -1260.0;
   CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 21;
 END MODULE
*****************************
* —— VOLUME TWENTYTWO —— *
*****************************
 MODULE: VOLM00;
   NAME: VL22;
   I/O LIST: UPSTREAM PROPERTIES    - VL21,
             INLET FLOW             - 05  ,
             EXIT FLOW              - LPOT,
             DOWNSTREAM PROPERTIES - LTOD,
             QDOT                   - VL22;
   DESIGN VALUES: VOL  - 995.0,
                  QDOT - 0.0  ;
   CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME 22;
 END MODULE
*************************************************
* —— PREBURNER OXIDIZER SPLITTER VOLUME —— *
*************************************************
 MODULE: VOLM01;
   NAME: PBSO;
   I/O LIST: UPSTREAM PROPERTIES    - PBPD,
             INLET FLOW             - PRBP,
             EXIT FLOW              - 06  , OPOV, FPOV,
             DOWNSTREAM PROPERTIES - VL20, PBSO, PBSO,
             QDOT                   - PBSO;
```

CONFIG. INPUT

```
    DESIGN VALUES: VOL  - 347.0;
    CMT: DENSITY AND INTERNAL ENERGY DERIVATIVES FOR VOLUME PBSO;
  END MODULE
***************************
* --- POGO SUPRESSION --- *
***************************
  MODULE: POGO00;
    NAME: POGO;
    I/O LIST: INLET HELIUM FLOW     - HE4 ,
              LIQUID OXIDIZER PROP  - VL20,
              GASEOUS OXIDIZER FLOW - O7  ,
              EXIT FLOW             - O8  ,
              EXIT PROP             - VL17;
    DESIGN VALUES: VOL  - 2000.;
    CMT: POGO SUPPRESSOR;
  END MODULE
************************************************************************
* -------------------- HOT GAS SIDE NON-DERIVATIVE MODULES ------------ *
************************************************************************
*
******************************************
* --- NON-INERTIAL HOT GAS LINE TWO --- *
******************************************
  MODULE: PIPE05;
    NAME: HG2 ;
    I/O LIST: INLET PROPERTIES - OPRB,
              EXIT PROPERTIES  - OTBP;
    DESIGN VALUES: AREA - 12.34,
                   RKLS - 1.000;
    CMT: CALCULATES OPRB PRESSURE FROM OTBP PRESSURE AND FLOW;
  END MODULE
*******************************************
* --- NON-INERTIAL HOT GAS LINE FOUR --- *
*******************************************
  MODULE: PIPE02;
    NAME: HG4 ;
    I/O LIST: INLET PROPERTIES - OTBP,
              EXIT PROPERTIES  - OSF ;
    DESIGN VALUES: AREA - 0.0567,
                   RKLS - 1.0;
    CMT: CALCULATES FLOW FROM VOLUME OTBP TO VOLUME OSF;
  END MODULE
**************************************************************
* --- HIGH PRESSURE OXIDIZER TURBINE DISCHARGE PRESSURE --- *
**************************************************************
```

CONFIG. INPUT

```
   EQUATION: ZHPOT  = ZOTBP;
******************************************
* --- HIGH PRESSURE OXIDIZER TURBINE --- *
******************************************
   MODULE: TURBO1;
      NAME: HPOT;
      I/O LIST: INLET PROPERTIES = OTBP,
                EXIT  PROPERTIES = HTOD,
                SHAFT WORK       = OH  ;
      DESIGN VALUES: ETAD = 0.752,
                     PRD  = 1.51,
                     PSID = 1.00,
                     SND  = 27240.8,
                     AREA = 1.0      ,
                     DIAM = 10.19    ,
                     GEAR = 1.0      ;
      MAP: TBMP05;
      CMT: HIGH PRESSURE OXYGEN TURBINE;
   END MODULE
*******************************************************
* --- NON-INERTIAL HOT GAS OVERBOARD LEAKAGE LINE --- *
*******************************************************
   MODULE: PIPE02;
      NAME: OLK ;
      I/O LIST: INLET PROPERTIES = OSF ,
                EXIT PROPERTIES  = AMB ;
      DESIGN VALUES: AREA = 0.0,
                     RELS = 1.0;
      CMT: CALCULATES LEAKAGE FLOW FROM VOLUME OSF;
   END MODULE
*****************************************
* --- NON-INERTIAL HOT GAS LINE SIX --- *
*****************************************
   MODULE: PIPE05;
      NAME: HG6 ;
      I/O LIST: INLET PROPERTIES = OSF ,
                EXIT PROPERTIES  = MFI ;
      DESIGN VALUES: AREA = 11.276,
                     RELS = 1.000;
      CMT: CALCULATES OSF PRESSURE FROM MFI PRESSURE AND FLOW;
   END MODULE
*****************************************
* --- NON-INERTIAL HOT GAS LINE ONE --- *
*****************************************
   MODULE: PIPE05;
```

CONFIG. INPUT

```
    NAME: HG1 ;
    I/O LIST: INLET PROPERTIES = FPRB,
              EXIT PROPERTIES  = FTBP;
    DESIGN VALUES: AREA = 31.86,
                   RKLS = 1.000;
    CMT: CALCULATES FPRB PRESSURE FROM FTBP PRESSURE AND FLOW;
  END MODULE
*******************************************
* --- NON-INERTIAL HOT GAS LINE THREE --- *
*******************************************
  MODULE: PIPE02;
    NAME: HG3 ;
    I/O LIST: INLET PROPERTIES = FTBP,
              EXIT PROPERTIES  = FSF ;
    DESIGN VALUES: AREA = 0.0,
                   RKLS = 1.0;
    CMT: CALCULATES FLOW FROM VOLUME FTBP TO VOLUME FSF;
  END MODULE
**********************************************************
* --- HIGH PRESSURE FUEL TURBINE DISCHARGE PRESSURE --- *
**********************************************************
  EQUATION: ZHPFT  = ZFTBP;
**************************************
* --- HIGH PRESSURE FUEL TURBINE --- *
**************************************
  MODULE: TURB01;
    NAME: HPFT;
    I/O LIST: INLET PROPERTIES = FTBP,
              EXIT  PROPERTIES = HTFD,
              SHAFT WORK       = FH  ;
    DESIGN VALUES: ETAD = 0.804,
                   PRD  = 1.45,
                   PSID = 1.00,
                   SND  = 34189.8,
                   AREA = 1.0   ,
                   DIAM = 10.19  ,
                   GEAR = 1.0    ;
    MAP: TBMP03;
    CMT: HIGH PRESSURE FUEL TURBINE;
  END MODULE
******************************************
* --- NON-INERTIAL HOT GAS LINE FIVE --- *
******************************************
  MODULE: PIPE05;
    NAME: HG5 ;
```

CONFIG. INPUT

```
    I/O LIST: INLET PROPERTIES - FSF ,
              EXIT PROPERTIES  - MFI ;
    DESIGN VALUES: AREA - 21.35,
                   RKLS - 1.000;
    CMT: CALCULATES FSF PRESSURE FROM MFI PRESSURE AND FLOW;
  END MODULE
**************************************************
* —— NON-INERTIAL HOT GAS FUEL INJECTOR LINE —— *
**************************************************
  MODULE: PIPE05;
    NAME: FINJ;
    I/O LIST: INLET PROPERTIES - MFI ,
              EXIT PROPERTIES  - MCHB;
    DESIGN VALUES: AREA - 27.90,
                   RKLS - 1.000;
    CMT: CALCULATES MFI PRESSURE FROM MCHB PRESSURE AND FLOW;
  END MODULE
******************************
* —— NOZZLE PERFORMANCE —— *
******************************
  MODULE: NOZL00;
    NAME: NOZL;
    I/O LIST: INLET PROPERTIES - MCHB,
              EXIT  PROPERTIES - AMB ;
    DESIGN VALUES: AREA - 82.05,
                   AR   - 77.5,
                   CS   - 0.98;
    CMT: NOZZLE PERFORMANCE CALCUALTION;
  END MODULE
********************************************
* —— NOZZLE HOT GAS SIDE HEAT TRANSFER —— *
********************************************
  EQUATION : AREANZLG - AREANOZL ;
  MODULE: QNOZ01;
    NAME: QNOZ;
    I/O LIST: HOT GAS PROPERTIES - MCHB,
              QDOT               - NOZ1, NOZ2,
              TM                 - MTL1, MTL2,
              NOZZLE AREA        - NZLG;
    DESIGN VALUES: RCRV - 6.1;
    CMT: NOZZLE HEAT TRANSFER RATES;
  END MODULE
*********************************************
* —— CHAMBER HOT GAS SIDE HEAT TRANSFER —— *
*********************************************
```

CONFIG. INPUT

```
MODULE: QCHM01;
   NAME: QCHM;
   I/O LIST: HOT GAS PROPERTIES - MCHB,
             QDOT               - CHM1, CHM2,
             TM                 - MTL3, MTL4,
             NOZZLE AREA        - NZLG;
   DESIGN VALUES: RCRV - 6.1;
   CMT: CHAMBER HEAT TRANSFER RATES;
 END MODULE
************************************************************************
* -------------------- HOT GAS SIDE DERIVATIVE MODULES ------------- *
************************************************************************
*
******************************
* --- OXIDIZER PREBURNER --- *
******************************
 MODULE: PBRN01;
   NAME: OPRB;
   I/O LIST: FUEL FLOW            - FOPB,
             FUEL PROPERTIES      - PBSF,
             OXIDIZER FLOW        - OOPB,
             OXIDIZER PROPERTIES  - OPBI,
             HELIUM FLOW          - HE2 ,
             HELIUM PROPERTIES    - HETK,
             EXIT FLOW            - HG2 ;
   DESIGN VALUES: VOL  - 347. ,
                  OFBL - 0.08 ,
                  OFLT - 0.4  ,
                  ILIT - 1    ;
   CMT: PRESS., TEMP., OXID. FRAC., AND HE. FRAC. DERIV. FOR OPRB;
 END MODULE
*******************************************
* --- OXIDIZER TURBINE BY-PASS VOLUME --- *
*******************************************
 MODULE: VOLM02;
   NAME: OTBP;
   I/O LIST: UPSTREAM PROPERTIES   - OPRB,
             INLET FLOW            - HG2 ,
             EXIT FLOW             - HPOT, HG4 ,
             DOWNSTREAM PROPERTIES - OSF , OSF ,
             QDOT                  - OTBP;
   DESIGN VALUES: VOL  - 500.0;
   CMT: PRESS., TEMP., OXID. FRAC., AND HE. FRAC. DERIV. FOR OTBP;
 END MODULE
*****************************************************
```

CONFIG. INPUT

```
* —— OXIDIZER PREBURNER SECONDARY FLOW VOLUME —— *
*******************************************************
   MODULE: VOLM02;
      NAME: OSF ;
      I/O LIST: UPSTREAM PROPERTIES   - HTOD, OTBP, PBSF,
                INLET FLOW            - HPOT, HG4 , OTC ,
                EXIT FLOW             - HG8 , OLK ,
                DOWNSTREAM PROPERTIES - MFI , AMB ,
                QDOT                  - OSF ;
      DESIGN VALUES: VOL  - 500.0;
      CMT: PRESS., TEMP., OXID. FRAC., AND HE. FRAC. DERIV. FOR OSF;
   END MODULE
***************************
* —— FUEL PREBURNER —— *
***************************
   MODULE: PBRN01;
      NAME: FPRB;
      I/O LIST: FUEL FLOW           - FFPB,
                FUEL PROPERTIES     - PBSF,
                OXIDIZER FLOW       - OFPB,
                OXIDIZER PROPERTIES - FPBI,
                HELIUM FLOW         - HE1 ,
                HELIUM PROPERTIES   - HETK,
                EXIT FLOW           - HG1 ;
      DESIGN VALUES: VOL  - 347.0,
                     OFBL - 0.08 ,
                     OFLT - 0.4  ,
                     ILIT - 1    ;
      CMT: PRESS., TEMP., OXID. FRAC., AND HE. FRAC. DERIV. FOR FPRB;
   END MODULE
*****************************************
* —— FUEL TURBINE BY-PASS VOLUME —— *
*****************************************
   MODULE: VOLM02;
      NAME: FTBP;
      I/O LIST: UPSTREAM PROPERTIES   - FPRB,
                INLET FLOW            - HG1 ,
                EXIT FLOW             - HPFT, HG3 ,
                DOWNSTREAM PROPERTIES - FSF , FSF ,
                QDOT                  - FTBP;
      DESIGN VALUES: VOL  - 500.0;
      CMT: PRESS., TEMP., OXID. FRAC., AND HE. FRAC. DERIV. FOR FTBP;
   END MODULE
**************************************************
* —— FUEL PREBURNER SECONDARY FLOW VOLUME —— *
```

CONFIG. INPUT

```
*************************************************
   EQUATION : OFRVL3 = 0.0 ;
   EQUATION : HFRVL3 = 0.0 ;
   MODULE: VOLM02;
      NAME: FSF ;
      I/O LIST: UPSTREAM PROPERTIES   - HIFD, FTBP, VL3 ,
                INLET FLOW            - HPFT, HG3 , FTC ,
                EXIT FLOW             - HG5 ,
                DOWNSTREAM PROPERTIES - MFI ,
                QDOT                  - FSF ;
      DESIGN VALUES: VOL  = 500.0;
      CMT: PRESS., TEMP., OXID. FRAC., AND HE. FRAC. DERIV. FOR FSF;
   END MODULE
**************************************
* —— MAIN FUEL INJECTOR VOLUME —— *
**************************************
   MODULE: VOLM02;
      NAME: MFI ;
      I/O LIST: UPSTREAM PROPERTIES   - FSF , OSF , VL16,
                INLET FLOW            - HG5 , HG6 , FSLV,
                EXIT FLOW             - FINJ,
                DOWNSTREAM PROPERTIES - MCHB,
                QDOT                  - FMCO, OMCO;
      DESIGN VALUES: VOL  =4210.0;
      CMT: PRESS., TEMP., OXID. FRAC., AND HE. FRAC. DERIV. FOR MFI;
   END MODULE
************************************
* —— MAIN CHAMBER COMBUSTION —— *
************************************
   MODULE: MCHB01;
      NAME: MCHB;
      I/O LIST: FUEL FLOW            - F15 ,
                FUEL PROPERTIES      - VL16,
                FUEL IGNITER FLOW    - FIG ,
                OXIDIZER FLOW        - OINJ,
                OXIDIZER PROPERTIES  - MOI ,
                OXIDIZER IGNITER FLOW- OIG ,
                MFI INLET FLOW       - FINJ,
                MFI INLET PROPERTIES - MFI ,
                HELIUM FLOW          - HE3 ,
                HELIUM PROPERTIES    - HETK,
                EXIT FLOW            - NZLG,
                QDOT                 - CHM1, CHM2,
                ILITPB1              - FPRB,
                ILITPB2              - OPRB;
```

CONFIG. INPUT

```
   DESIGN VALUES: VOL  -2225.0,
                  OFBL - 0.08 ,
                  OFLT - 0.4  ,
                  ILIT - 1    ;
   CMT: PRESS., TEMP., OXID. FRAC., AND HE. FRAC. DERIV. FOR MCHB;
  END MODULE
*******************************************************
* —— NOZZLE NODE ONE METAL TEMPERATURE DYNAMICS —— *
*******************************************************
  MODULE: METL00;
   NAME: MTL1;
   I/O LIST: QDOT - 6HOT, NOZ1;
   DESIGN VALUES: IMTL - 1   ,
                  RM   - 18.2;
   CMT: METAL TEMPERATURE DERIVATIVE FOR NOZ1/MTL1;
  END MODULE
*******************************************************
* —— NOZZLE NODE TWO METAL TEMPERATURE DYNAMICS —— *
*******************************************************
  MODULE: METL00;
   NAME: MTL2;
   I/O LIST: QDOT - 7HOT, NOZ2;
   DESIGN VALUES: IMTL - 1   ,
                  RM   - 48.5;
   CMT: METAL TEMPERATURE DERIVATIVE FOR NOZ2/MTL2;
  END MODULE
********************************************************
* —— CHAMBER NODE ONE METAL TEMPERATURE DYNAMICS —— *
********************************************************
  MODULE: METL00;
   NAME: MTL3;
   I/O LIST: QDOT - 11HT, CHM1;
   DESIGN VALUES: IMTL - 1     ,
                  RM   - 31.0  ;
   CMT: METAL TEMPERATURE DERIVATIVE FOR CHM1/MTL3;
  END MODULE
********************************************************
* —— CHAMBER NODE TWO METAL TEMPERATURE DYNAMICS —— *
********************************************************
  MODULE: METL00;
   NAME: MTL4;
   I/O LIST: QDOT - 12HT, CHM2;
   DESIGN VALUES: IMTL - 1     ,
                  RM   - 31.0  ;
   CMT: METAL TEMPERATURE DERIVATIVE FOR CHM2/MTL4;
```

CONFIG. INPUT

```
  END MODULE
*******************************************************
* —— LOW PRESSURE FUEL TURBOPUMP ROTOR DYNAMICS —— *
*******************************************************
  MODULE: ROTROO;
    NAME: FL  ;
    I/O LIST: TORQUE - LPFT, LPFP;
    DESIGN VALUES: PMOM   - 1.2853 ;
    CMT: ROTOR SPEED DERIVATIVE FOR LPFT/LPFP;
  END MODULE
********************************************************
* —— HIGH PRESSURE FUEL TURBOPUMP ROTOR DYNAMICS —— *
********************************************************
  MODULE: ROTROO;
    NAME: FH  ;
    I/O LIST: TORQUE - HPFT, HPFP;
    DESIGN VALUES : PMOM   - 2.8505    ;
    CMT: ROTOR SPEED DERIVATIVE FOR HPFT/HPFP;
  END MODULE
***********************************************************
* —— LOW PRESSURE OXIDIZER TURBOPUMP ROTOR DYNAMICS —— *
***********************************************************
  MODULE: ROTROO;
    NAME: OL  ;
    I/O LIST: TORQUE - LPOT, LPOP;
    DESIGN VALUES : PMOM   - 2.3900    ;
    CMT: ROTOR SPEED DERIVATIVE FOR LPOT/LPOP;
  END MODULE
************************************************************
* —— HIGH PRESSURE OXIDIZER TURBOPUMP ROTOR DYNAMICS —— *
************************************************************
  MODULE: ROTROO;
    NAME: OH  ;
    I/O LIST: TORQUE - HPOT, HPOP, PRBP;
    DESIGN VALUES : PMOM   - 1.4496    ;
    CMT: ROTOR SPEED DERIVATIVE FOR HPOT/HPOP;
  END MODULE
*
 END SYSTEM INSIDE
*
*------------------------------------------------------------*
*                                                            *
*                  BELOW THE ITERATION LOOP                  *
*                                                            *
*------------------------------------------------------------*
```

CONFIG. INPUT

```
*
DEFINE SYSTEM BELOW
*
******************************************************************
* ---- EULER INTEGRATE THE DENSITY OF THE INJECTOR VOLUMES ---- *
******************************************************************
   EQUATION : VOLMOI   -1158.9;
   EQUATION : VOLOPBI  -  41.5;
   EQUATION : VOLFPBI  -  81.8;
   EQUATION : RHOMOI   - RHOMOI  + DT * (WMOV  - WOINJ) / VOLMOI ;
   EQUATION : RHOOPBI  - RHOOPBI + DT * (WOPOV - WOOPB) / VOLOPBI;
   EQUATION : RHOFPBI  - RHOFPBI + DT * (WFPOV - WOFPB) / VOLFPBI;
*********************************************************
* ---- LOW PRESSURE FUEL TURBOPUMP ROTOR BREAK AWAY - *
*********************************************************
   MODULE: ROTR01;
      NAME: FLBR;
      I/O LIST: TORQUE - LPFT, LPFP,
                SHAFT  - FL  ;
      DESIGN VALUES: BTQ  - 40.0;
      CMT: ROTOR BREAK AWAY FOR LPFT/LPFP;
   END MODULE
**********************************************************
* ---- HIGH PRESSURE FUEL TURBOPUMP ROTOR BREAK AWAY - *
**********************************************************
   MODULE: ROTR01;
      NAME: FHBR;
      I/O LIST: TORQUE - HPFT, HPFP,
                SHAFT  - FH  ;
      DESIGN VALUES : BTQ  - 150.0;
      CMT: ROTOR BREAK AWAY FOR HPFT/HPFP;
   END MODULE
*************************************************************
* ---- LOW PRESSURE OXIDIZER TURBOPUMP ROTOR BREAK AWAY - *
*************************************************************
   MODULE: ROTR01;
      NAME: OLBR;
      I/O LIST: TORQUE - LPOT, LPOP,
                SHAFT  - OL  ;
      DESIGN VALUES: BTQ  - 80.0;
      CMT: ROTOR BREAK AWAY FOR LPOT/LPOP;
   END MODULE
**************************************************************
* ---- HIGH PRESSURE OXIDIZER TURBOPUMP ROTOR BREAK AWAY - *
**************************************************************
```

```
 MODULE: ROTR01;
    NAME: OHBR;
    I/O LIST: TORQUE - HPOT, HPOP,
              SHAFT  - OH  ;
    DESIGN VALUES : BTQ  -  90.0;
    CMT: ROTOR BREAK AWAY FOR HPOT/HPOP;
 END MODULE
*
END SYSTEM BELOW
```