

NASA/TM-2003-212270



## **System Modeling and Diagnostics for Liquefying-Fuel Hybrid Rockets**

*Scott Poll, David Iverson, Jeremy Ou, Dwight Sanderfer, Ann Patterson-Hine*

---

**June 2003**

## The NASA STI Program Office . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) Program Office plays a key part in helping NASA maintain this important role.

The NASA STI Program Office is operated by Langley Research Center, the Lead Center for NASA's scientific and technical information. The NASA STI Program Office provides access to the NASA STI Database, the largest collection of aeronautical and space science STI in the world. The Program Office is also NASA's institutional mechanism for disseminating the results of its research and development activities. These results are published by NASA in the NASA STI Report Series, which includes the following report types:

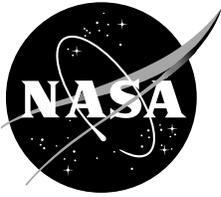
- **TECHNICAL PUBLICATION.** Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA's counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.
- **TECHNICAL MEMORANDUM.** Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.
- **CONTRACTOR REPORT.** Scientific and technical findings by NASA-sponsored contractors and grantees.

- **CONFERENCE PUBLICATION.** Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or cosponsored by NASA.
- **SPECIAL PUBLICATION.** Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.
- **TECHNICAL TRANSLATION.** English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services that complement the STI Program Office's diverse offerings include creating custom thesauri, building customized databases, organizing and publishing research results . . . even providing videos.

For more information about the NASA STI Program Office, see the following:

- Access the NASA STI Program Home Page at <http://www.sti.nasa.gov>
- E-mail your question via the Internet to [help@sti.nasa.gov](mailto:help@sti.nasa.gov)
- Fax your question to the NASA Access Help Desk at (301) 621-0134
- Telephone the NASA Access Help Desk at (301) 621-0390
- Write to:  
NASA Access Help Desk  
NASA Center for AeroSpace Information  
7121 Standard Drive  
Hanover, MD 21076-1320



## **System Modeling and Diagnostics for Liquefying-Fuel Hybrid Rockets**

*Scott Poll*

*Ames Research Center, Moffett Field, California*

*David Iverson*

*Ames Research Center, Moffett Field, California*

*Jeremy Ou*

*QSS Group, Inc.*

*Ames Research Center, Moffett Field, California*

*Dwight Sanderfer*

*Ames Research Center, Moffett Field, California*

*Ann Patterson-Hine*

*Ames Research Center, Moffett Field, California*

National Aeronautics and  
Space Administration

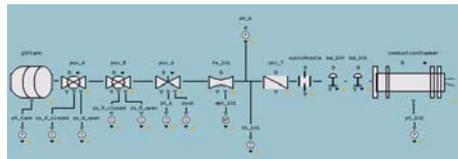
Ames Research Center  
Moffett Field, California 94035-1000

Available from:

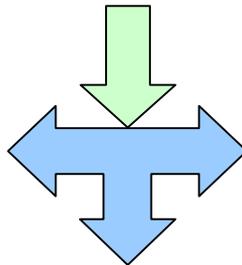
NASA Center for AeroSpace Information  
7121 Standard Drive  
Hanover, MD 21076-1320  
(301) 621-0390

National Technical Information Service  
5285 Port Royal Road  
Springfield, VA 22161  
(703) 487-4650

# System Modeling and Diagnostics for Liquefying-Fuel Hybrid Rockets

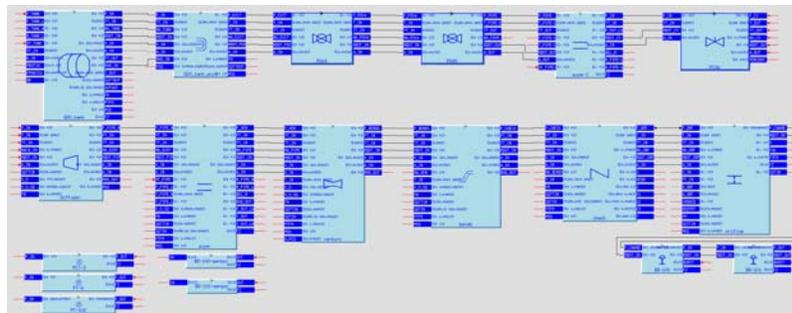
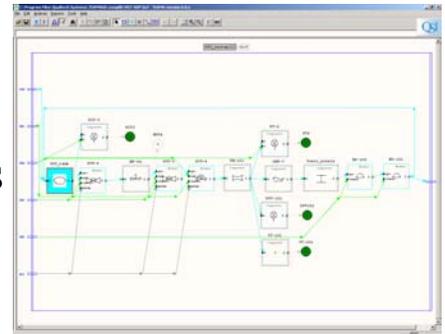


L2



TEAMS

RODON



Scott Poll  
David Iverson  
Jeremy Ou  
Dwight Sanderfer  
Ann Patterson-Hine

## ***Table of Contents:***

<b>1</b>	<b>Summary .....</b>	<b>1</b>
<b>2</b>	<b>Introduction.....</b>	<b>2</b>
<b>3</b>	<b>HCF Overview.....</b>	<b>3</b>
<b>4</b>	<b>Fault Detection and Diagnostic Methods.....</b>	<b>5</b>
<b>4.1</b>	<b>TEAMS .....</b>	<b>6</b>
<b>4.2</b>	<b>L2 .....</b>	<b>8</b>
<b>4.3</b>	<b>RODON .....</b>	<b>10</b>
<b>4.4</b>	<b>ICS – Interval Constraint Simulator .....</b>	<b>12</b>
<b>4.5</b>	<b>IMS – Inductive Monitoring System.....</b>	<b>13</b>
<b>5</b>	<b>HCF Data Characterization.....</b>	<b>13</b>
<b>5.1</b>	<b>Data Sources .....</b>	<b>14</b>
<b>5.2</b>	<b>Data Issues .....</b>	<b>14</b>
<b>5.3</b>	<b>Merging LabVIEW and Controller Data .....</b>	<b>15</b>
<b>6</b>	<b>General Architecture for Model-Based Diagnosis .....</b>	<b>17</b>
<b>7</b>	<b>HCF Models .....</b>	<b>19</b>
<b>7.1</b>	<b>TEAMS .....</b>	<b>19</b>
7.1.1	Controller Subsystem.....	20
7.1.2	Pneumatic Subsystem .....	22
7.1.3	LOX Subsystem .....	24
7.1.3.1	LOX_pump.....	27
7.1.3.2	POV-3 & POV-2 .....	28
7.1.3.3	Burst Disk 99.....	31
7.1.4	GOX Subsystem .....	32
7.1.4.1	POV-4 & POV-5 .....	35
7.1.4.2	PCV-6 .....	36
7.1.4.3	Burst Disks 100 & 101.....	38
7.1.5	Ignition Subsystem .....	38
7.1.5.1	POV-11 & POV-21.....	39
7.1.6	Combustion Chamber Subsystem.....	39
7.1.7	Testability analysis .....	43
7.1.8	Real Data to TEAMS.....	45
<b>7.2</b>	<b>L2.....</b>	<b>46</b>
7.2.1	Feedline Contents .....	46
7.2.2	Sensors, Limit Switches, and Position Feedback .....	47
7.2.3	GOX Tank.....	48
7.2.4	Shutoff Valve POV-4 .....	50

7.2.5	Shutoff Valve POV-5 .....	51
7.2.6	Control Valve (PCV-6).....	53
7.2.7	Venturi (FE-101) .....	60
7.2.8	Check Valve (CKV-7) .....	60
7.2.9	Sonic Nozzle .....	61
7.2.10	Burst Disks (BD-100 and BD-101) .....	61
7.2.11	Combustion chamber .....	62
7.2.12	Real Data to Livingstone .....	64
<b>7.3</b>	<b>RODON .....</b>	<b>66</b>
7.3.1	GOX Tank.....	68
7.3.2	GOX Tank Exit .....	68
7.3.3	Shutoff Valves POV-4, POV-5 .....	69
7.3.4	Pipe-1 .....	72
7.3.5	Control Valve PCV-6 .....	72
7.3.6	Diffuser .....	76
7.3.7	Pipe .....	77
7.3.8	Venturi .....	78
7.3.9	Bends .....	79
7.3.10	Check Valve .....	80
7.3.11	Sonic Orifice .....	80
7.3.12	Burst Disks .....	81
7.3.13	Sensors .....	81
7.3.14	RODON Equations .....	82
7.3.15	Simplified Model.....	83
7.3.16	Real Data to RODON .....	85
7.3.17	Simulations.....	88
<b>8</b>	<b>Simulations Using ICS .....</b>	<b>93</b>
<b>9</b>	<b>Model Comparison.....</b>	<b>96</b>
<b>10</b>	<b>Fault Detection and Diagnostic Demonstrations.....</b>	<b>101</b>
<b>10.1</b>	<b>L2 .....</b>	<b>101</b>
<b>10.2</b>	<b>TEAMS .....</b>	<b>109</b>
<b>10.3</b>	<b>RODON .....</b>	<b>115</b>
<b>10.4</b>	<b>Fault Detection with IMS.....</b>	<b>118</b>
<b>11</b>	<b>Conclusions.....</b>	<b>121</b>
<b>12</b>	<b>Suggestions for Further Study .....</b>	<b>122</b>
<b>13</b>	<b>References.....</b>	<b>123</b>

## ***List of Figures:***

<b>Figure 1: Ames Hybrid Combustion Facility.....</b>	<b>4</b>
<b>Figure 2: Schematic of HCF. ....</b>	<b>5</b>
<b>Figure 3: Basic idea of model-based reasoning.....</b>	<b>6</b>
<b>Figure 4: Testability Engineering and Maintenance System Tool Set.....</b>	<b>7</b>
<b>Figure 5: Relation of signals to failure modes and tests. ....</b>	<b>8</b>
<b>Figure 6: Livingstone as employed in the Deep Space-1 Remote Agent Experiment.</b> .....	<b>10</b>
<b>Figure 7: RODON functionality.....</b>	<b>12</b>
<b>Figure 8: Problems with controller logged data. ....</b>	<b>15</b>
<b>Figure 9: Example of smoothed controller GOX tank pressure data.....</b>	<b>16</b>
<b>Figure 10: Merging LabVIEW and controller logged data.....</b>	<b>17</b>
<b>Figure 11: Combustion chamber pressure traces after aligning delivery pressures.....</b>	<b>17</b>
<b>Figure 12: Architecture for model-based reasoning tools. ....</b>	<b>18</b>
<b>Figure 13: Top-level schematic of TEAMS HCF model.....</b>	<b>20</b>
<b>Figure 14: Controller subsystem. ....</b>	<b>21</b>
<b>Figure 15: Pneumatic subsystem. ....</b>	<b>23</b>
<b>Figure 16: LOX subsystem module. ....</b>	<b>25</b>
<b>Figure 17: LOX pump module. ....</b>	<b>27</b>
<b>Figure 18: Valve module.....</b>	<b>29</b>
<b>Figure 19: Ball valve failure modes.....</b>	<b>31</b>
<b>Figure 20: Burst disk module. ....</b>	<b>32</b>
<b>Figure 21: GOX subsystem module.....</b>	<b>34</b>
<b>Figure 22: Control valve module.....</b>	<b>37</b>
<b>Figure 23: Ignition subsystem. ....</b>	<b>38</b>
<b>Figure 24: Combustion chamber subsystem.....</b>	<b>41</b>
<b>Figure 25: Testability Figures of Merit for the HCF model using all tests.....</b>	<b>44</b>
<b>Figure 26: Testability Figures of Merit for the HCF model using logged data. ....</b>	<b>45</b>
<b>Figure 27: Livingstone HCF model.....</b>	<b>46</b>
<b>Figure 28: Example of typical GOX tank pressure trace with L2 bins.....</b>	<b>49</b>
<b>Figure 29: POV-5 component.....</b>	<b>53</b>
<b>Figure 30: Pressures and control valve characteristics for a nominal 4 kg/sec firing.</b> .....	<b>55</b>
<b>Figure 31: PCV-6 component. ....</b>	<b>56</b>
<b>Figure 32: Example of typical smoothed control valve command trace with L2 bins.</b> .....	<b>57</b>
<b>Figure 33: Example of typical smoothed control valve position trace with bins. ....</b>	<b>58</b>
<b>Figure 34: Example of delivery pressure trace with L2 bins.....</b>	<b>59</b>
<b>Figure 35: An example of a typical combustion chamber trace with L2 bins. ....</b>	<b>63</b>
<b>Figure 36: Combustion chamber component.....</b>	<b>64</b>
<b>Figure 37: RODON HCF model.....</b>	<b>67</b>
<b>Figure 38: GOX tank equations. ....</b>	<b>68</b>
<b>Figure 39: GOX tank exit equations. ....</b>	<b>69</b>
<b>Figure 40: Shutoff valve (POV-4, POV-5) equations.....</b>	<b>71</b>
<b>Figure 41: Shutoff valve CV characteristics. ....</b>	<b>71</b>

<b>Figure 42: Valve open and close characteristics.</b>	72
<b>Figure 43: Control valve command and position tracking.</b>	74
<b>Figure 44: Effect of data time shift on PID error parameter.</b>	76
<b>Figure 45: Diffuser equations.</b>	77
<b>Figure 46: Pipe pressure loss equations.</b>	78
<b>Figure 47: Venturi pressure loss equations.</b>	79
<b>Figure 48: Pipe bends equations.</b>	80
<b>Figure 49: Sonic orifice equations.</b>	81
<b>Figure 50: Simplified RODON HCF model.</b>	84
<b>Figure 51: RODON monitoring for firing 8.</b>	87
<b>Figure 52: RODON simulation attempts on a valve, pipe, and orifice system using mass flow at the end of the time step.</b>	90
<b>Figure 53: Example of inverse interval relationship.</b>	90
<b>Figure 54: RODON simulation attempts on valve, pipe, and orifice system using mass flow at beginning of time step.</b>	91
<b>Figure 55: Illustration of valve sensitivity to pressure in valve-pipe-orifice system.</b>	92
<b>Figure 56: Illustration of divergence of RODON simulation using simplified model.</b>	93
<b>Figure 57: ICS simulation of firing 8 using PID control for control valve position.</b>	94
<b>Figure 58: ICS simulation of firing 8 using experimental data for control valve position.</b>	96
<b>Figure 59: Notional comparison of HCF model scope and completeness.</b>	96
<b>Figure 60: Perspective view of the combustion chamber.</b>	102
<b>Figure 61: Firing 2 smoothed control valve command and feedback position traces.</b>	103
<b>Figure 62: Pressure traces of firing 2.</b>	104
<b>Figure 63: Diagnosis after first block of scenario file.</b>	108
<b>Figure 64: Diagnosis after third block of scenario file.</b>	108
<b>Figure 65: Diagnosis after fourth block of scenario file.</b>	109
<b>Figure 66: TEAMS-RT tests before POV-4 opens.</b>	110
<b>Figure 67: TEAMS-RT diagnosis before POV-4 opens.</b>	111
<b>Figure 68: TEAMS-RT tests after ignition command.</b>	112
<b>Figure 69: TEAMS-RT diagnosis after ignition command.</b>	113
<b>Figure 70: TEAMS-RT tests after second failure.</b>	114
<b>Figure 71: TEAMS-RT diagnosis after second failure.</b>	115
<b>Figure 72: An example of RODON monitoring fault detection.</b>	116
<b>Figure 73: Simulated failure scenario with POV-4 stuck partially open after command to close.</b>	117
<b>Figure 74: Simulated transient failure scenario with POV-4 momentarily stuck partially open after command to close.</b>	118
<b>Figure 75: HCF data vector definition used for IMS.</b>	119

## ***List of Tables:***

<b>Table 1: HCF feed line and combustion chamber instrumentation.....</b>	<b>4</b>
<b>Table 2: Controller subsystem components and tests.....</b>	<b>22</b>
<b>Table 3: Pneumatic subsystem components and tests. ....</b>	<b>24</b>
<b>Table 4: LOX subsystem components/modules and tests.....</b>	<b>26</b>
<b>Table 5: LOX pump components and tests.....</b>	<b>28</b>
<b>Table 6: Valve components and tests. ....</b>	<b>30</b>
<b>Table 7: Burst disk components and tests.....</b>	<b>32</b>
<b>Table 8: GOX subsystem components/modules and tests. ....</b>	<b>35</b>
<b>Table 9: Control valve components and tests. ....</b>	<b>37</b>
<b>Table 10: Ignition subsystem components and tests.....</b>	<b>39</b>
<b>Table 11: Combustion chamber components and tests. ....</b>	<b>42</b>
<b>Table 12: Sensors, limit switches, and position feedback components.....</b>	<b>47</b>
<b>Table 13: GOX tank component. ....</b>	<b>48</b>
<b>Table 14: POV-4 component. ....</b>	<b>50</b>
<b>Table 15: POV-5 component. ....</b>	<b>51</b>
<b>Table 16: PCV-6 component.....</b>	<b>54</b>
<b>Table 17: Venturi component.....</b>	<b>60</b>
<b>Table 18: Check valve component.....</b>	<b>60</b>
<b>Table 19: Sonic nozzle component.....</b>	<b>61</b>
<b>Table 20: Burst disk component.....</b>	<b>61</b>
<b>Table 21: Combustion chamber component. ....</b>	<b>62</b>
<b>Table 22: Control valve <math>C_v</math> characteristics. ....</b>	<b>73</b>
<b>Table 23: Tool comparison. ....</b>	<b>100</b>
<b>Table 24: Firing 2 event timing. ....</b>	<b>103</b>
<b>Table 25: Thresholds for 2 kg/sec, 600 psi firing.....</b>	<b>105</b>

## **Acronyms and Abbreviations**

ARC	Ames Research Center
COTS	Commercial Off The Shelf
FMEA	Failure Mode and Effects Analysis
GOX	Gaseous Oxygen
HCF	Hybrid Combustion Facility
ICS	Interval Constraint Simulator
IMS	Inductive Monitoring System
IVHM	Integrated Vehicle Health Management
L2	Livingstone 2
LOX	Liquid Oxygen
PID	Proportional Integral Derivative Control
PLC	Programmable Logic Controller
PSI	pounds per square inch
RODON	Greek for ROSE, Reasoning Over Systems in their Entirety
RTI	Real-time Interface
TEAMS	Testability Engineering and Maintenance System
TFOM	Testability Figures Of Merit

# 1 Summary

A Hybrid Combustion Facility (HCF) was recently built at NASA Ames Research Center to study the combustion properties of a new fuel formulation that burns approximately three times faster than conventional hybrid fuels. The improved fuel performance means that, for the first time, hybrid rockets have the potential to be safer, less expensive replacements to the solid and liquid rockets of current launch systems. Researchers at Ames working in the area of Integrated Vehicle Health Management (IVHM) recognized a good opportunity to apply IVHM techniques to a candidate technology for next generation launch systems.

Five tools were selected to examine various IVHM techniques for the HCF. It should be emphasized that the tools were selected purely on their availability and familiarity to the researchers; no extensive survey was performed to find out which techniques might be best for the task at hand. Three of the tools, TEAMS (Testability Engineering and Maintenance System), L2 (Livingstone2), and RODON, are model-based reasoning (or diagnostic) systems. Two other tools in this study, ICS (Interval Constraint Simulator) and IMS (Inductive Monitoring System) do not attempt to isolate the cause of the failure but may be used for fault detection.

Models of varying scope and completeness were created. The TEAMS model has the largest scope but does not include all of the software components that are necessary to produce a diagnosis directly from the data. The L2 model has a reduced scope and also does not implement all of the software components needed to automatically diagnose the system. The RODON model has the smallest scope but is able to work with the logged data directly to produce a diagnosis. The TEAMS and L2 models are qualitative whereas the RODON model is quantitative.

In each of the models, the structure and behavior of the physical system are captured. In TEAMS, the behavior is highly abstracted to a list of signals, or attributes. In L2, the behavior is modeled with qualitative propositional formulae. In RODON, qualitative and quantitative formulae are used. In the qualitative models, the temporal aspects of the system behavior and the abstraction of sensor data are handled outside of the model and require the development of additional code. In the quantitative model, less extensive processing code is also necessary. Examples of fault diagnoses are given.

The IMS should be useful for real-time fault detection of systems that are difficult to model or as a first pass monitoring tool that can catch problems before passing them on to diagnostic tools for further analysis. The IMS is able to learn system behavior from experimental data and simulation of nominal runs and demonstrates a fast fault detection capability.

For the HCF, the introduction of model-based reasoning tools would add little value. The pre-defined sequence of events, limited subsystem interactions, expert knowledge of the operators, and the ability to perform visual inspections after the firing make the addition of an IVHM system superfluous for fault diagnosis of the HCF. Once the technologies

being tested at the HCF are incorporated into a vehicle, the case for an IVHM system becomes stronger. We expect that there will be many more system interactions that are not scheduled. Fault isolation becomes much harder for a human to do efficiently as the system becomes more complex with many measurements and interactions. In addition, we must rely on on-board sensors for diagnostic information during flight.

## 2 Introduction

A Hybrid Combustion Facility (HCF) was recently built at NASA Ames Research Center to investigate the combustion properties of a new fuel formulation developed by Stanford University researchers. A hybrid rocket is one in which the fuel is in solid form and the oxidizer is in liquid or gaseous form. The fuel being tested at the HCF is paraffin-based, similar to candle wax, and the oxidizer is gaseous oxygen.

The primary advantage of hybrid rockets over liquid and solid rockets is the inherently safe nature of the fuel—in manufacturing, handling, and operationally. The fuel by itself is not volatile, which leads to a number of cost reductions of a vehicle launch system. In addition, the products of combustion are harmless carbon dioxide and water. Unlike solid rockets, hybrid rockets can be throttled to change the thrust after they are ignited.

Hybrid rockets have been studied since the 1940's but the hybrid fuels considered did not burn fast enough to make it a viable concept for large rockets. Recent theoretical and experimental studies at Stanford University have shown that low-viscosity solids like paraffin form a liquid melt layer on their surface that ejects droplets into the flame, greatly enhancing fuel transfer and increasing the regression rate (how fast the fuel burns) by approximately three times that of the hybrid fuels previously tested [2]. Because of the dramatically improved performance of the paraffin-based fuel compared to conventional hybrid fuels, hybrid rockets now have the potential to be safer, less expensive replacements to the solid rocket boosters on the space shuttle or other launch systems.

The HCF was constructed to see if the promising results of the bench-top experiments would scale up to a size that is closer to an operational rocket and to examine the physical mechanisms of the liquid-layer combustion process. Several firings of the facility have been completed and more are planned in the future.

Researchers at Ames working in the area of Integrated Vehicle Health Management (IVHM) recognized a good opportunity to apply IVHM techniques and concepts to a candidate technology for next generation launch systems. One of the most difficult and time-consuming aspects of putting together an IVHM system is knowledge acquisition. Here, that problem is mitigated because of easy access to the experts (and their willingness to share their knowledge) and the facility. This gives us a chance to examine a variety of IVHM tools, compare and contrast their approaches, and assess the feasibility of using such techniques in a hybrid rocket health management system.

### 3 HCF Overview

A detailed description of the Hybrid Combustion Facility can be found in [1]. A brief overview is presented here to give the reader the necessary background to understand the facility operation and the diagnostic models. Figure 1 and Figure 2 [1] show a composite picture and simplified sketch of the facility, respectively. There are six main systems: liquid oxygen (LOX) feed, gaseous oxygen (GOX) feed, combustion, ignition, pneumatics, and controller.

Prior to a firing, the oxidizer stored in the LOX tank is pumped through the vaporizer and gasified before entering the GOX tank. Over a period of up to an hour, GOX flows into the GOX tank until the pressure reaches the required level for the desired mass flow rate and run duration. At this point, the LOX feed system is isolated from the GOX tank by closing a shut off valve between the vaporizer and the GOX tank.

The operator enters the desired run setpoints into the control computer. These include parameters for control valve scheduling, ignition timing, the desired delivery pressure, and configuration information. After a firing countdown, the upstream shutoff valve is opened (the redundant downstream valve has previously been opened). The GOX flow chokes at the orifice (sonic nozzle) and continues into the combustion chamber. A short time later, the ignition system oxidizer and fuel flow are turned on and ignited by a spark. High temperature combustion products from the ignition system are injected into the combustion chamber and vaporize the paraffin fuel, which mixes with the free stream oxidizer and the ignition products to ignite the paraffin in a self-sustaining combustion reaction.

As the GOX tank pressure decreases during the course of a firing, the control valve opens to maintain constant delivery pressure (and mass flow) to the combustion chamber. A venturi in the GOX feed line measures the oxygen mass flow rate but is accurate only for steady state operating conditions. The orifice measures the mass flow rate more accurately during transients and also serves to isolate any pressure fluctuations in the combustion chamber from the feed system.

A check valve upstream of the orifice prevents reverse flow of combustion gases from entering the GOX feed line. Two burst disks located just downstream of the orifice and one located upstream of the GOX tank protect against over pressurization.

Pressure sensors are located at the GOX tank, orifice, and combustion chamber. There is also a high frequency pressure sensor measurement of the combustion chamber pressure and a differential pressure measurement at the venturi. The GOX temperature is measured upstream of the orifice. All pneumatically actuated valves report open/close status feedback and the burst disks indicate burst/not burst status. Table 1 lists the instrumentation in the GOX feed system and the combustion chamber.

<b>ID</b>	<b>Measures</b>	<b>Location</b>	<b>Range</b>
PIT-3	GOX tank pressure	Upstream of GOX tank	0-3000 psi
ZSO-4	Valve POV-4 open limit switch	POV-4	0 (not open) or 1 (open)
ZSC-4	Valve POV-4 close limit switch	POV-4	0 (not closed) or 1 (closed)
ZSO-5	Valve POV-5 open limit switch	POV-5	0 (not open) or 1 (open)
ZSC-5	Valve POV-5 close limit switch	POV-5	0 (not closed) or 1 (closed)
ZSO-6	Valve PCV-6 open limit switch	PCV-6	0 (not open) or 1 (open)
ZSC-6	Valve PCV-6 close limit switch	PCV-6	0 (not closed) or 1 (closed)
ZT-6	Valve PCV-6 position feedback	PCV-6	0-100%
DPT-101	Venturi differential pressure	Venturi	0-30 psi
PT-6	GOX delivery pressure	Between venturi and check valve	0-3000 psi
TT-101	GOX temperature	Between venturi and check valve	-200-1250 °C
ZBD-99	Burst disk status	Upstream of GOX tank	0 (burst) or 1 (not burst)
ZBD-100	Burst disk status	Upstream of combustion chamber	0 (burst) or 1 (not burst)
ZBD-101	Burst disk status	Upstream of combustion chamber	0 (burst) or 1 (not burst)
PT-102	Combustion chamber pressure	Pre-combustion chamber	0-3000 psi
PT-201	Combustion chamber dynamic pressure	Pre-combustion chamber	0-15000 psi

**Table 1: HCF feed line and combustion chamber instrumentation.**



**Figure 1: Ames Hybrid Combustion Facility.**

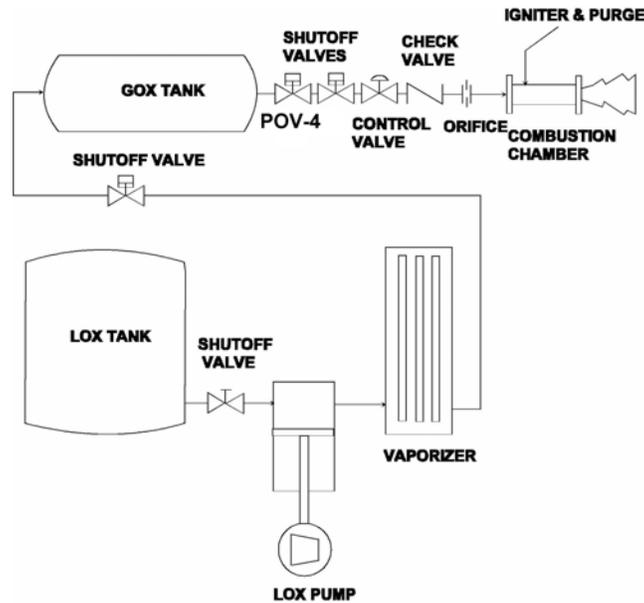
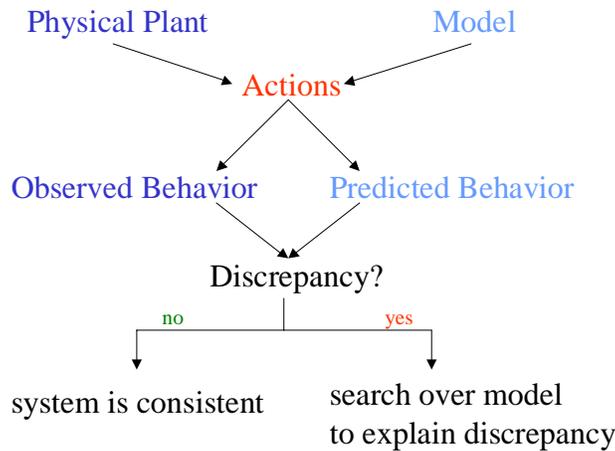


Figure 2: Schematic of HCF.

## 4 Fault Detection and Diagnostic Methods

During a firing of the HCF we wish to know whether any of the observed parameters deviate from usual or nominal values. This is fault detection (or anomaly detection if the deviation is not the result of a failure). We may also want to determine the kind of fault, the location and behavior of the fault, and the time at which the fault occurred. This is fault diagnosis. The HCF controller monitors individual signals and reports a fault when a parameter exceeds a threshold or indicates an unacceptable value (for example, if a burst disk sensor indicates that the burst disk has ruptured). Depending on the severity of the fault, the controller will perform an emergency shutdown to safe the facility. Since individual sensors are used for the fault detection, no information about the structure of the system or the relationships among the different sensor readings is captured. A single physical fault may result in many fault indications and it is up to the operator to sort through them to find the common cause and isolate the failure. Alternatively, one could attribute certain combinations of sensor indications to an underlying cause if the operator has built up enough experience with the facility to know how the faults are manifested. Since experience is a key factor, this rule-based approach may not cover certain faults that have not yet been observed and the rules may become invalid if the facility is modified. To overcome these difficulties, fault management systems have increasingly made use of model-based reasoning systems in which the physical structure and behavior of the plant are captured in hierarchical, compositional models. If the observations of the plant deviate from what is expected, a fault is detected and various algorithms may be used to isolate the fault. Figure 3, adapted from [3], illustrates the basic idea of model-based reasoning. While there are many model-based reasoning tools, only a few are examined in this study. It should be emphasized that the tools were selected purely on their availability and familiarity to the researchers; no extensive survey was performed to find out which techniques might be best for the task at hand. Rather, the goal was to see how to tackle the problem with the available tools and compare and contrast their

approaches. Three of the tools, TEAMS (Testability Engineering and Maintenance System), L2 (Livingstone2), and RODON, are model-based diagnostic (or reasoning) systems. Two other tools in this study, ICS (Interval Constraint Simulator) and IMS (Inductive Monitoring System) do not attempt to isolate the cause of the failure but may be used to detect faults. The next sections give an introductory overview of TEAMS, L2, RODON, ICS and IMS.



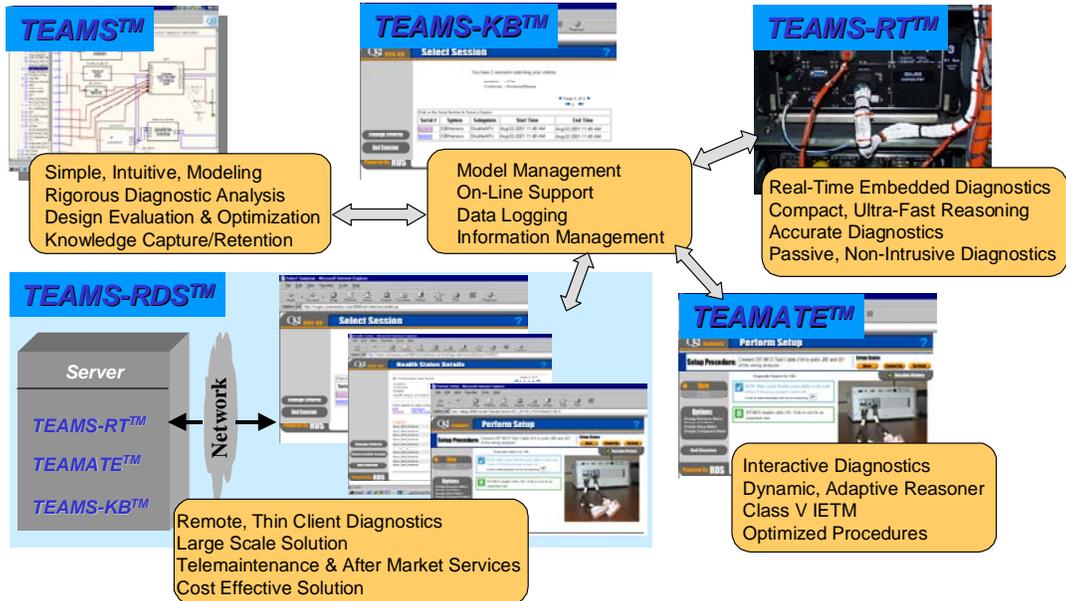
**Figure 3: Basic idea of model-based reasoning.**

#### 4.1 TEAMS

One method selected for study is the causal model-based technique implemented in a commercial tool set from Qualtech Systems, Inc (QSI). Qualtech’s integrated tool set (see Figure 4, courtesy Qualtech) for design-for-testability, interactive trouble-shooting and on-line monitoring and diagnostics, includes the three tools developed using NASA Small Business Innovation Research funding: TEAMS, TEAMS-RT, TEAMS-RDS. These tools are founded on the multi-signal flow graph modeling methodology and the concomitant fault-isolation algorithms. TEAMS integrates the modeling methodology and fault-isolation algorithms in an easy-to-use graphical user interface. TEAMS is mainly a design-for-testability tool, but the same diagnostic model is used with its companion tool, TEAMS-RT (a real-time diagnostics tool) to perform passive on-line fault-diagnosis using asynchronously arriving anomaly reports, alarms, or applied test-results. The TEAMS-RDS (Remote Diagnostic Server) product incorporates TEAMS-RT and other Qualtech software components on a server computer that can “serve” intelligent, optimized diagnostics to thin clients over the Internet or any computer network. All diagnostic reasoning and technical data can be maintained and upgraded on a central server and instantly made available to clients on the system (via local area network or Internet).

## What is the **TEAMS™** Tool Set?

*A Comprehensive (“Common Model”) Software Solution for Designing, Deploying, and Supporting Your Systems for the Entire Life Cycle*

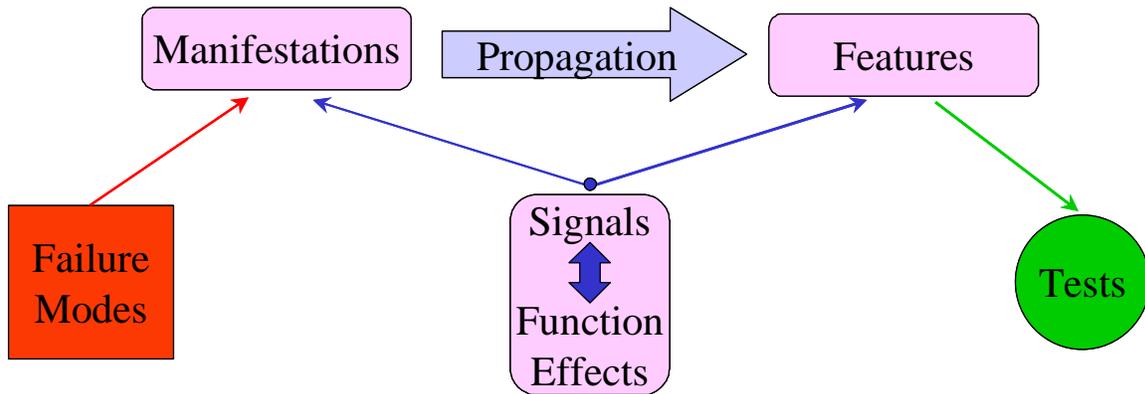


**Figure 4: Testability Engineering and Maintenance System Tool Set.**

The multi-signal modeling methodology is a hierarchical modeling methodology where the propagation paths of the effects of a failure are captured in terms of a directed graph [4]. The model is developed by entering the structure of the model, based on a schematic diagram or conceptual block diagram, and then adding signals to the modules as well as to test points. Signals describe the unique attributes of the variables in a system. Test points designate locations of visibility into the system. For example, physical locations of sensors such as pressure transducers would have a corresponding test point on the multi-signal flow graph. Multiple tests can be defined at a given test point. Test results can come from simple limit checks, feature extractions using signal analysis techniques, more complex data analysis algorithms, or even other diagnostic reasoners. Propagation algorithms convert this graph to a single global fault dictionary for a given mode of the system. This dictionary contains the basic information needed to interpret test results and diagnose failures during on-board monitoring. Multi-signal modeling allows the modeler to hierarchically describe the structure of a system and then specify its functional attributes via signals. It is not a simulation model but is ideally suited for building accurate, low-cost models that can be used by a reasoner in real-time to interpret test results and assess system health.

An important aspect of multi-signal modeling is the identification of signals—a process in which the modeler summarizes his understanding of the functions of components in the system in terms of their distinct attributes. Tests are procedures that look at the data from the sensors and make decisions about system attributes associated with those

measurements. Figure 5 shows the relation of signals, tests, and failure modes of the model. The test definition can include additional information such as test cost, test time (time required to perform the test), detection probability, false alarm probability, as well as a test label. Some of the test parameters are used by the TEAMS algorithm to optimize a troubleshooting tree. Test labels are useful for assessing the diagnosability of the system using various levels of instrumentation. The detection and isolation coverage available with a particular instrumentation configuration is determined by performing a testability analysis. The testability analysis is done on the same model that is then used by the QSI's real-time reasoning software. Using a consistent model during the design phase through operations enables continuous verification of the models by system experts and increased confidence in the automated system.



**Figure 5: Relation of signals to failure modes and tests.**

The testability analysis can aid advanced sensor development efforts by suggesting optimal sensor placement and analysis of possible redundancies in sensor coverage. QSI's toolset also includes TEAMATE, a portable maintenance aid used to support interactive troubleshooting, and TEAMS-KB, a knowledge base that supports model management as well as data logging during operations. Other outputs of TEAMS include FMEA generation and text reports that describe the component relationships that contribute to failure propagation throughout the system.

## 4.2 L2

Livingstone is an open-source model-based reasoning tool that was developed at NASA Ames in the past decade and recently enhanced to L2. Some relevant papers are Williams and Nayak [5] and Kurien and Nayak [6]. Livingstone was one of the component technologies demonstrated in the Deep Space-1 Remote Agent Experiment [7] as depicted in Figure 6.

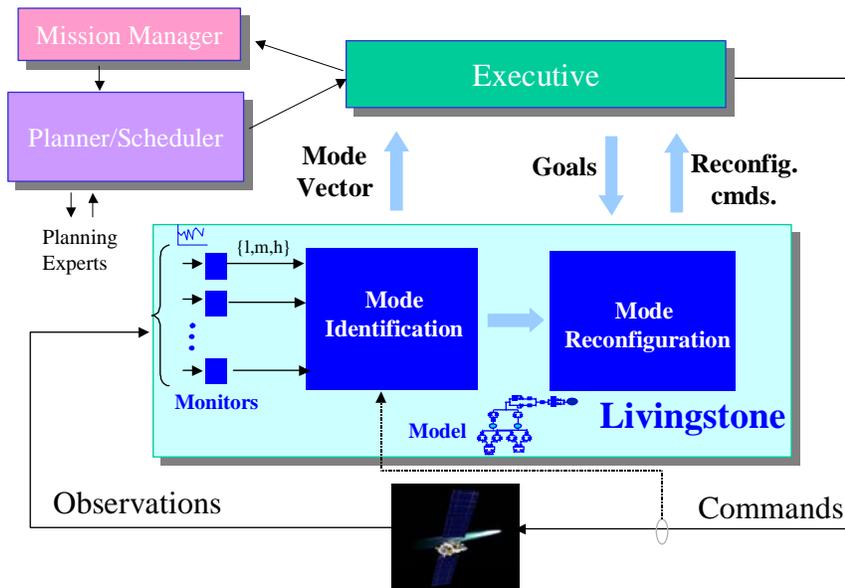
The fundamental tasks of Livingstone are to eavesdrop on system commands, to determine whether those commands had the desired effect on the spacecraft or plant, and to recommend reconfiguration actions to achieve a desired goal in the event of failures. To accomplish these tasks, a declarative model of the system is built in a hierarchical, compositional framework. Automata (components) describe system behavior with a

finite number of nominal and failure modes and by transitions between the modes. Within each mode, propositional formulae relate the component inputs to outputs and define the functional behavior of the component in that particular mode. The inputs and outputs may be connected to other components in the model. Transitions between nominal modes represent commanded configuration changes while transitions from nominal to failure modes are unexpected failure events. The nominal transitions have an associated cost while the failure transitions have an associated prior probability of occurrence. Each automaton has a transition variable with the nominal and failure transitions in its domain. The set of assignments to each transition variable at each time step describes a trajectory of system evolution. L2 incrementally generates multiple trajectories, in order of likelihood, that are consistent with the commands and observations.

Given the commands to the system, L2 attempts to transition the model to the next nominal configuration. If the observed sensor values violate the constraints imposed by the propositional formulae of the intended nominal modes, the conflict results in a fault being detected, and L2 diagnoses what the failed system state is. L2 uses a conflict-directed, best-first search to efficiently find consistent candidate trajectories. Conflict-directed refers to using the conflict (discrepancy) to avoid generating candidates that contain assignments to the variables that would include the conflict. Best-first search refers to checking the consistency of the most likely candidates first. If none of the candidates of a certain likelihood are consistent, the next most probable candidates are checked for consistency and so on. Diagnosis amounts to finding the most likely assignments to the transition variables that are consistent with the commands and observations. Recovery, or reconfiguration, addresses finding the least costly actions (transitions) required to move the system into the desired state. The recovery feature of L2 was not used in this study.

A Livingstone model is a discrete representation of a physical system. Consequently, real-valued sensor data and system behavior must be abstracted into a discrete space. Discrete variable values typically represent a range of real-valued numbers or a range of the rate of change of real-valued numbers and are chosen to aid the diagnosis of the system. Monitor code external to L2 performs the discretization of the real-valued sensor data to the discrete variable values used in L2.

## Livingstone in Remote Agent



**Figure 6: Livingstone as employed in the Deep Space-1 Remote Agent Experiment.**

### 4.3 RODON

RODON is a commercial model-based reasoning software tool produced by ROSE Informatik. It uses a mathematical system description for simulation, performance analysis, risk analysis, monitoring, and diagnosis of complex technical systems. The behavior of each system component is modeled using familiar engineering equations combined with logic clauses. The topology of the system is modeled by connecting these components together via input and output ports. These models are entered into RODON using a graphical system editor and component editor.

In contrast to other diagnostic tools used on this project, RODON performs numerical calculations and reasons directly with system sensor values rather than abstracting these values into discrete bins or working with results of external tests. This allows RODON to reason with higher fidelity system models that can provide tighter monitoring tolerances and, in many cases, more accurate diagnoses. One potential draw back of this detailed modeling capability is an increase in computational complexity that can lead to a slower diagnostic response time than we might experience with the other, more abstract, diagnostic tools. The response time issue can be addressed by the modeler to a certain extent by removing unnecessary detail from the model. System sensor placement, type, and tolerance as well as the desired diagnostic response time can help the modeler determine an appropriate level of model detail.

The basis of RODON's reasoning is a constraint satisfaction algorithm that determines if the data provided by the system sensors can be used to derive a consistent instantiation of the mathematical model. In other words, can the sensor readings be propagated through

the model in such a way that a set of equations describing nominal system behavior all hold true? RODON performs system monitoring by collecting system sensor data and executing this consistency check. If the sensor values lead to a consistent model solution, RODON determines that the system is functioning properly. If no consistent solution can be found, RODON can use the same system model to perform a diagnosis. Diagnoses are accomplished by suspending the constraints imposed by the equations describing each component and attempting to solve the resulting model with the provided sensor data. If a consistent solution can be found when the constraints for a particular component are suspended, that component is presented as a suspected failure. Although the underlying algorithm is more complex and efficient, the end effect is to suspend nominal behavior for each component, one by one, and check for model consistency. While not required, RODON also permits the modeler to mathematically describe failure modes of each component. If, during a diagnosis, a consistent solution can be found by substituting the failure mode constraints for a component's nominal constraints, then that failure mode will be presented as a suspected failure.

One other unique feature of RODON is the use of interval arithmetic for its calculations. Rather than requiring that each variable used in an equation be assigned a single numeric value, it can be assigned ranges of values or intervals. These intervals will be used in the solution of all the equations containing that variable. If certain values in the interval cause one of the equations containing the variable to become invalid, those values are removed from the interval. If there are no values in the interval that make the equation valid, then a conflict occurs and that particular equation is no longer allowed for consideration in the model solution. The use of intervals in RODON provides a convenient method for incorporating uncertainty, such as component or sensor tolerances, into the system model. For instance, consider a pressure sensor known to be accurate to within plus or minus one percent. If this sensor provides a reading of 100 PSI, that value could be presented to the RODON model as an interval ranging from 99 PSI to 101 PSI. This would ensure that the actual pressure is contained somewhere in the interval used for calculations.

Although we only utilized RODON's simulation, monitoring, and diagnosis features on this project, there are RODON software modules that allow the user to extract additional information from the system model. One module can produce a set of diagnostic rules from the model. While these rules may not provide quite as much diagnostic fidelity as the full system model, they provide a more compact knowledge base that allows for faster diagnosis while using significantly less computer memory and processor power. RODON can also automatically build diagnostic troubleshooting trees for use by service personnel, perform automated FMEA and similar risk analyses, and provide system testability and diagnosability analysis (see Figure 7).

# RODON

## Application Areas

RODON is a Software-Tool for the Integration and Support of Essential Phases of the Product Lifecycle Process

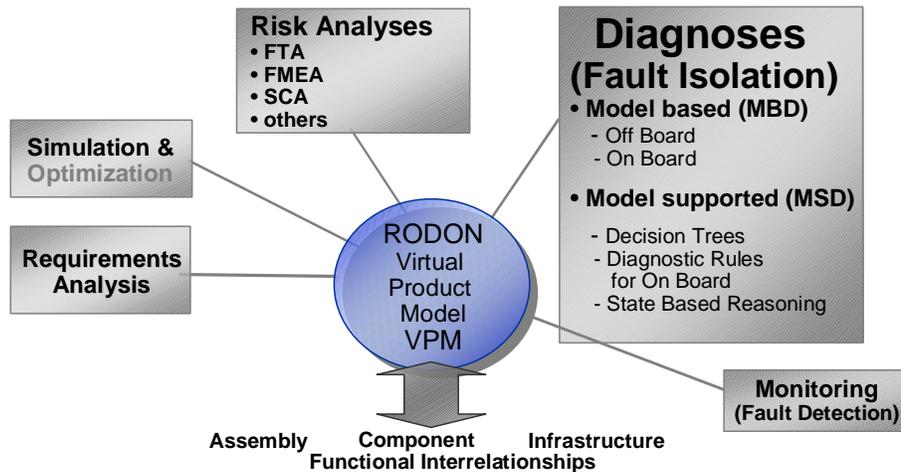


Figure 7: RODON functionality.

### 4.4 ICS – Interval Constraint Simulator

The Interval Constraint Simulator (ICS) software was developed in ARC Code IC to provide a fast, flexible system modeling and simulation tool. ICS calculates with interval arithmetic and uses the same type of system model as the RODON software, so it can implement the same type of simulations as RODON. Unlike RODON, ICS does not provide diagnostic ability. This eliminates the computational overhead associated with the diagnosis algorithm. Decreased overhead, combined with an efficient implementation in the C programming language, enables ICS to perform system simulations much faster than the RODON tool in most circumstances. The ICS modeling language also provides a more convenient way to implement certain common mathematical functions than the RODON modeling language.

Access to the ICS source code allows researchers to incorporate extra functionality in their simulations (e.g., a custom PID control loop) that might be difficult or impossible to model with a commercial, closed source tool. The speed and flexibility of ICS also facilitates experimentation in defining system model components and parameters before including them in a diagnostic model. Additionally, ICS can be used to produce simulated nominal and off-nominal data sets for use in development and testing of IVHM tools. ICS may also be useful as a system monitoring tool that can provide initial fault detection prior to analysis by a slower, more complex diagnostic program.

The ICS tool is still under development and will require additional work before moving out of the laboratory for general use. Despite its current juvenility and primitive interface it has been helpful in the HCF IVHM studies. ICS promises to be a useful modeling and simulation tool as it is more fully developed.

#### **4.5 IMS – Inductive Monitoring System**

Another IVHM tool under development by Code IC researchers is the Inductive Monitoring System (IMS). It utilizes techniques from the fields of model-based reasoning, machine learning, and data mining to build system monitoring knowledge bases from archived sensor data. IMS was motivated by the difficulty of producing detailed diagnostic models of some system components due to complexity or unavailability of design information. IMS will also allow for fast monitoring performance. Initial experiments show that IMS should be able to provide real-time monitoring of 10 Hz data, and may be able to process 1 KHz data in real time.

All that is required to build an IMS monitoring knowledge base are several sets of nominal system sensor data. These data sets can be collected directly from the system to be monitored or from system simulations. Unlike some other machine learning techniques, such as neural networks, IMS does not require examples of anomalous (failure) behavior. IMS automatically analyzes the nominal system data to form general classes of expected system sensor values. These classes are used to build the monitoring knowledge base.

When monitoring a system, IMS simply checks to see if the incoming sensor data fits into one of the classes derived from the training data. If so, the system is assumed to be operating normally since it is behaving in a manner similar to previous nominal behavior. Otherwise IMS will alert the operator or diagnosis system that the data is suspicious and there may be a problem that should be investigated. If IMS is trained on data sets that are representative of anticipated operating conditions, the resulting monitoring knowledge base should provide a good characterization of nominal system behavior and an effective system monitoring capability.

The IMS concepts and algorithms have been tested with data sets from previous HCF firings with encouraging results. It was able to build a monitoring knowledge base that accurately characterized several “unseen” HCF data sets (data that were not used for training). Like the ICS tool, IMS is still under development and will require additional programming and integration before it is ready for general use.

## **5 HCF Data Characterization**

Each one of the tools introduced above requires data from the HCF as input. The quantity and quality of the data directly affect the degree and accuracy of fault detection and isolation. The next sections describe the sources of data, some problems with the data, and the attempts to rectify at least a few of those problems.

## **5.1 Data Sources**

The primary focus of the HCF firings has been to obtain regression rate data for the paraffin fuel. A LabVIEW data acquisition (DAQ) system is used to acquire the relevant data at 1000 Hz. The following parameters are recorded: venturi differential pressure, temperature, combustion chamber pressure (Rosemount transducer), dynamic combustion chamber pressure (Kistler transducer), and GOX delivery pressure. Data are written to a file at the end of a firing.

More data, including pressures, operator setpoints, permissives, fault indications, valve commands and statuses, are logged in the HCF controller at 10 Hz. These data are typically accessed when trouble-shooting the facility operation and are not used for regression rate calculations. Data are logged in a database and user-selected parameters can be exported to an ASCII text file. The time stamps in the LabVIEW and controller data files are not synchronized.

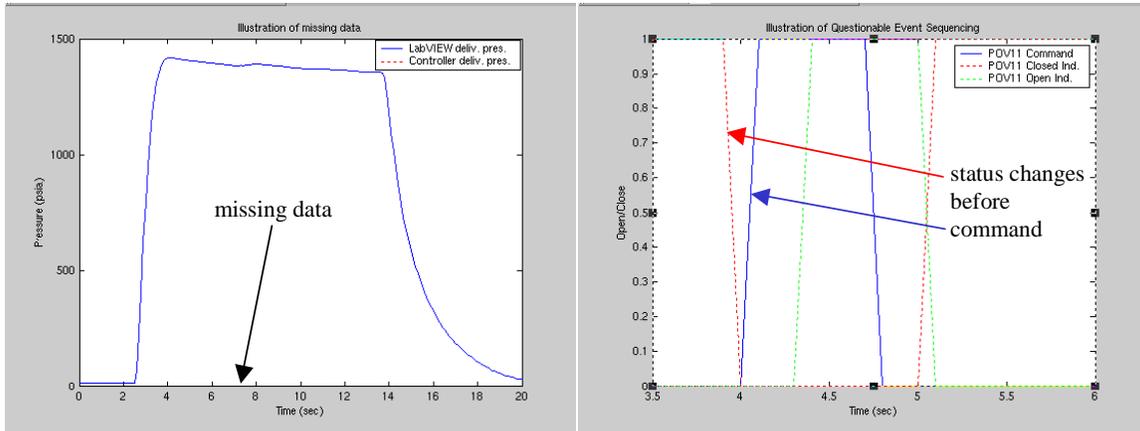
Each firing is recorded with a digital camera. The video is digitally processed to produce a time history of plume length during the run. Phototube measurements, recorded with the LabVIEW DAQ system, are used to generate a time history of plume intensity. Many firings have acoustic measurements of the jet noise and IR measurements of the plume. Those measurements are not discussed in this report.

In addition to the quantitative measurements mentioned above, there are heuristic data such as qualitative human visual, auditory, and tactile sensory measurements made during and after the run. They include observations such as plume color, width, length, flicker, and popping or crackle noises. Post-run inspections might reveal unexpected burn patterns, cracked insulators, or asymmetric combustion chamber heating.

## **5.2 Data Issues**

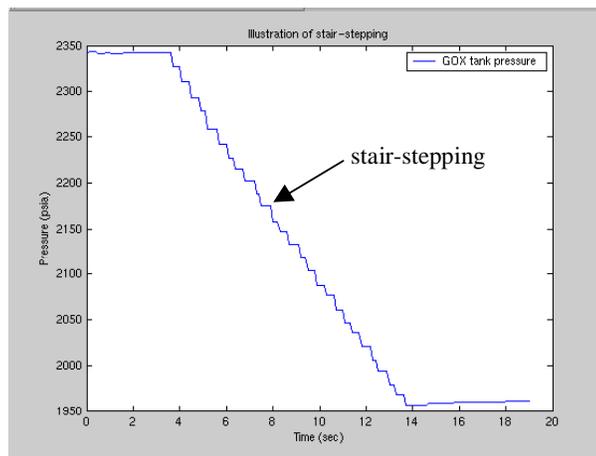
The HCF controller uses COTS (commercial off-the-shelf) WonderWare software to control the facility operation. This software also logs the system data for historical trending. However, the primary tasks of the controller are to command and monitor system operation, not to acquire high-fidelity data for analysis. Consequently, there are a number of issues with controller data quality. First, nearly half of the 30 firings completed thus far are missing data for certain parameters—most notably, the GOX delivery pressure and control valve position feedback (see Figure 8a). The delivery pressure is recorded by the LabVIEW system but not having the controller data makes merging the data sets more difficult (see the next section). Second, there are sequencing problems because of the low priority of the logging function relative to the control function of the software and because of the relatively low logging rate (10 Hz). This leads to firings where the reported valve position changes in the same time step as the command to the valve or even before, thus making event timing and sequencing information from the data highly questionable (see Figure 8b). Finally, the data “stair-steps” because the controller software does not record parameters continuously but only when a certain percentage change has occurred, as demonstrated in Figure 8c. All of

these problems make diagnosing the system more difficult and may lead to incorrect conclusions without correcting or allowing for the data irregularities.



a) Illustration of missing controller data.

b) Illustration of data sequencing problems.



c) Illustration of stair-stepping data.

**Figure 8: Problems with controller logged data.**

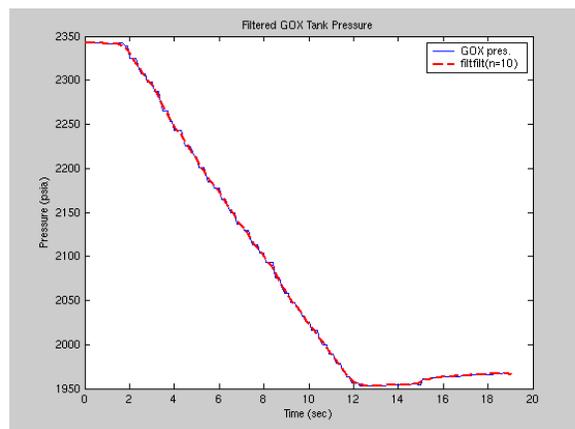
### 5.3 Merging LabVIEW and Controller Data

The parameters recorded by the LabVIEW system are those relevant to calculating the fuel regression rate and do not include any of the other system parameters that might be of interest to a diagnostic system such as commands and statuses to and from valves and operating setpoints. The controller data contain most of the parameters of interest but there is the issue of data quality mentioned in the previous section.

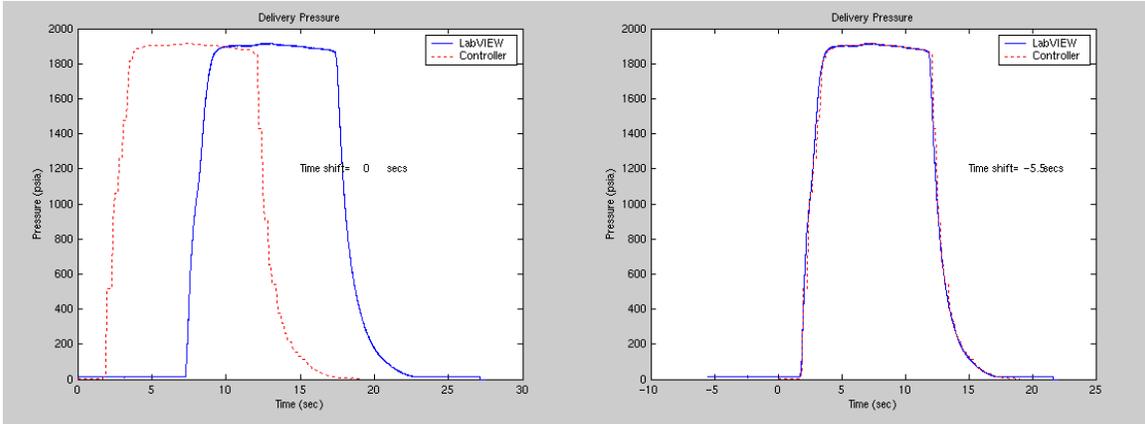
An attempt was made to merge the data sets so as to replace the stair-stepped controller data with the smooth LabVIEW data for the two parameters that were recorded by both systems (GOX delivery pressure and combustion chamber pressure). First, the LabVIEW data were filtered using the Matlab `filtfilt()` function with a 100 point window. This function essentially calculates a moving average with no phase shifts. One hundred points were used because this represents the value over a 0.1 second window, the controller logging time interval. It was not possible to replace the GOX tank pressure data since the LabVIEW system did not record this parameter so the signal was just filtered with a 10 point window as shown in Figure 9. Second, the GOX delivery pressures recorded by both data systems were plotted together. The LabVIEW data was shifted iteratively by eye until the two traces lined up. Figure 10 illustrates this simple process.

The firings for which the controller did not log the delivery pressure were slightly problematic. Figure 11 shows the combustion chamber pressure traces plotted with the same time shift as the data in Figure 10. Notice that by using the delivery pressure traces to synchronize the two data files, the combustion chamber pressure traces show a slight offset. This is due to the way the signals are received by the controller and LabVIEW systems. The delivery pressure is passed from the controller I/O block to the LabVIEW DAQ whereas the combustion chamber pressure is passed from the DAQ board to the controller. Because LabVIEW processes data at a much faster rate (1000 Hz compared to 10 Hz) there is less offset in the delivery pressure, as LabVIEW will register the value from the controller within 0.001 seconds. For the runs that were missing the controller delivery pressure, the combustion chamber pressure traces had to be used to align the data. A slight offset of the “aligned” traces was estimated by examining the offset in other runs.

After the filtered data had been aligned, the LabVIEW data was sub-sampled to 10 Hz and added to the parameters available in the controller log files.



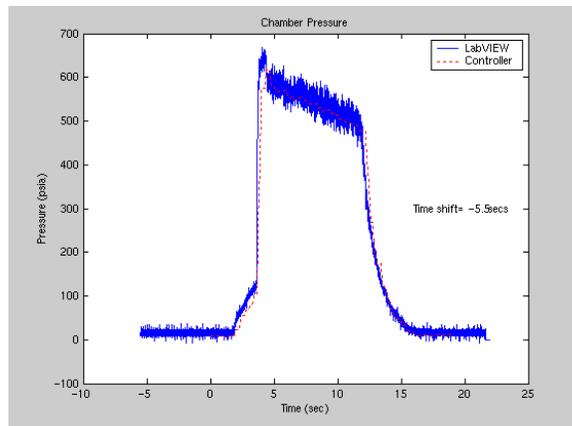
**Figure 9: Example of smoothed controller GOX tank pressure data.**



a) Controller and LabVIEW delivery pressures before alignment.

b) Controller and LabVIEW delivery pressures after alignment.

**Figure 10: Merging LabVIEW and controller logged data.**



**Figure 11: Combustion chamber pressure traces after aligning delivery pressures.**

## 6 General Architecture for Model-Based Diagnosis

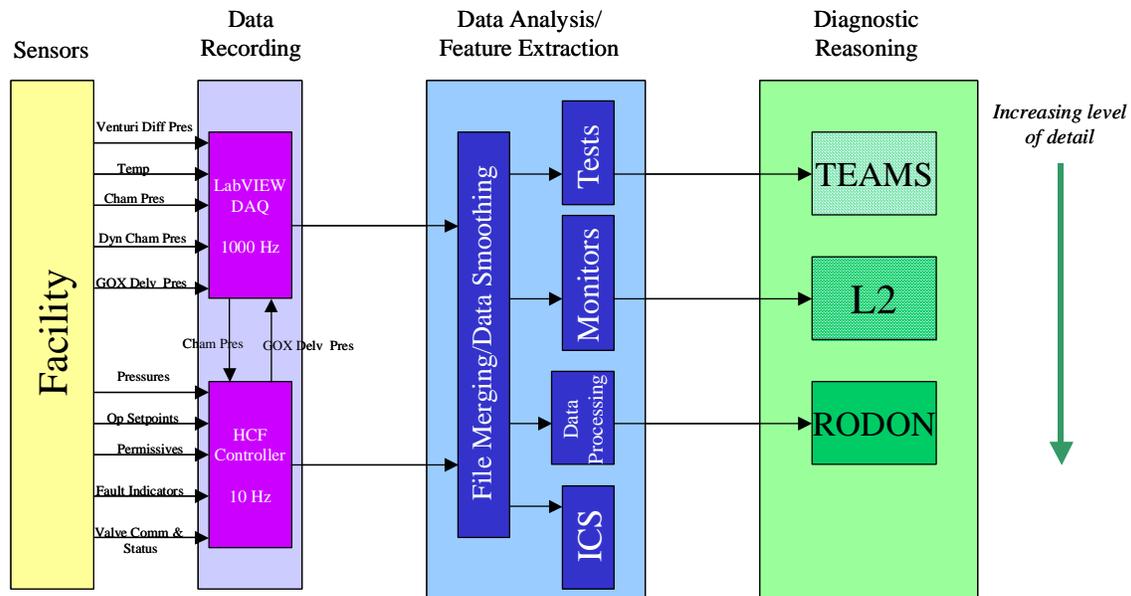
The overall architecture for the tools used in this study is depicted in Figure 12. Facility sensors and operator inputs are monitored and recorded by the HCF controller. Measurements important for regression rate calculation are acquired at a higher frequency on a separate LabVIEW data acquisition system. These two data files are then manipulated and joined together as described in the previous section.

L2 and TEAMS require that the continuous data be abstracted into a discrete space. For TEAMS, the abstraction is to a binary pass/fail test result for each test defined. For L2, the abstraction can be to a finite number of bins. The process of abstracting the data may be complex (e.g., vibration spectral analysis, wavelets or other signal processing techniques) or simple (e.g., thresholds). In general, the number of tests or bins will

increase with the amount of diagnostic information that can be associated with a signal. A RODON model does not need discrete data even if qualitative models are built. Tolerances are assigned directly to the sensor data input and the intervals are propagated through the model. Some data processing may be done however to speed up program execution or to simplify the model (e.g., taking an FFT of a signal would not be an easy task in RODON).

The processed data, including commands to the system, are then fed into TEAMS, L2, or RODON. Each tool checks whether or not the data corresponds to nominal system operation. If model constraints are violated or tests fail, the system is diagnosed.

An implementation of a real-time system with any one of the tools would require that the data currently being logged to files be fed to code that would perform the feature extraction and system calls to the diagnostic reasoner in an integrated fashion. Data sequencing is an important issue for all model-based reasoning tools. Time tags on data must show the correct order of commands and feedback. A consistent policy for diagnostic queries must be designed to avoid erroneous diagnosis. This is more of a challenge for tools working with discrete time and abstracted data. Some considerations for a real-time system with these tools include waiting for a time-out period after a command has been issued or after an unexpected observation is made to allow the system transients to settle, handling overlapping commands, and the buffering and debuffering of commands and diagnoses [8]. Real-time systems were not developed in this study.



**Figure 12: Architecture for model-based reasoning tools.**

## **7 HCF Models**

Models of the hybrid combustion facility were created using TEAMS, L2, and RODON. The next sections describe the models in detail.

### **7.1 TEAMS**

The scope of the TEAMS model includes the controller, pneumatics, LOX, GOX, ignition, and combustion subsystems. Figure 13 shows the top-level schematic of the multi-signal hierarchical model. The rectangular boxes represent each subsystem. Subsequent figures show the details of the subsystems. Facility schematics and design review documents were used to identify the components, their functional behaviors, and the connectivity between components and subsystems. The modeling process began with creating module blocks for the major subsystems and then dropping down into each subsystem and adding module blocks for the constituent components. The components and subsystems were connected with links that represent couplings such as piping, electrical wires or pneumatic tubing. Test points were added to the model where sensors are located on the facility and at a few locations where observations of the system operation could be made. Color-coding was used for the components, links, and tests to aid interpretation of the model. After the structure of the model was defined, functional behavior was included by attaching signals to the components and tests. This process will be described later. The following sections discuss the details of each of the subsystems.

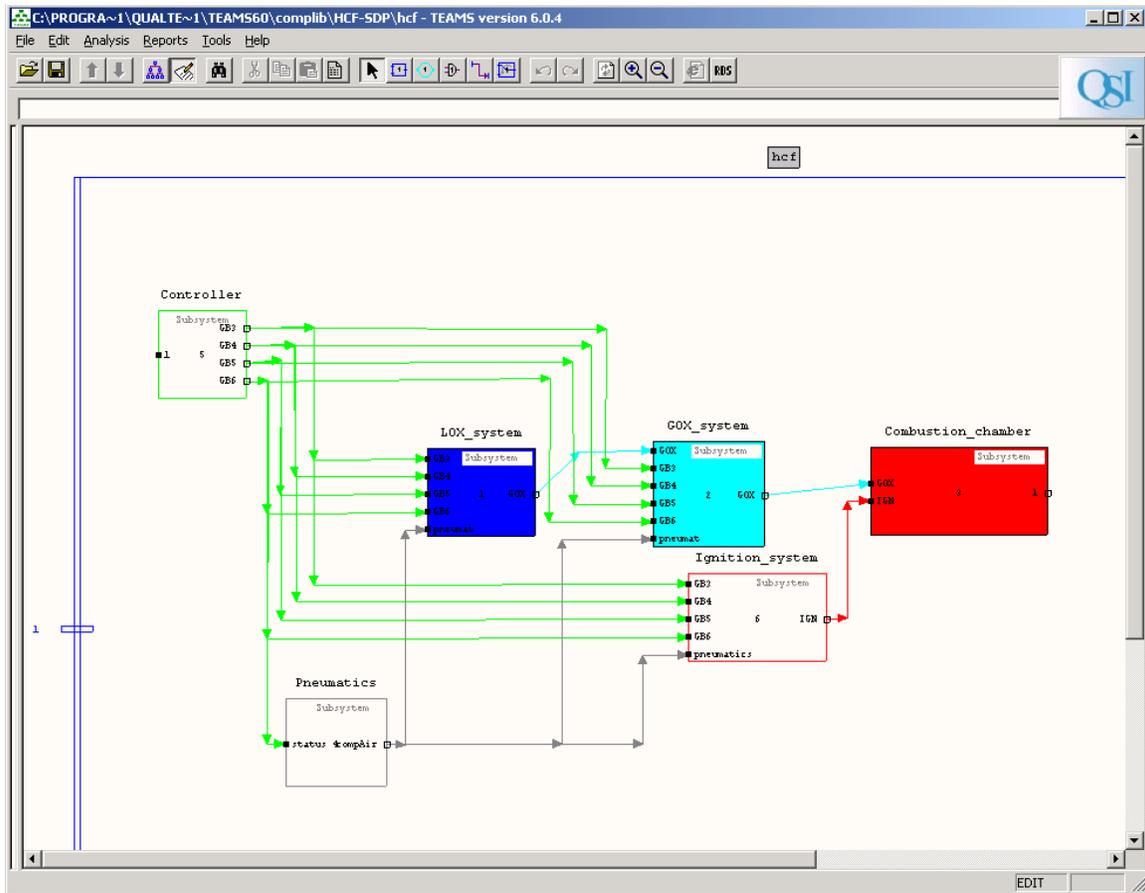
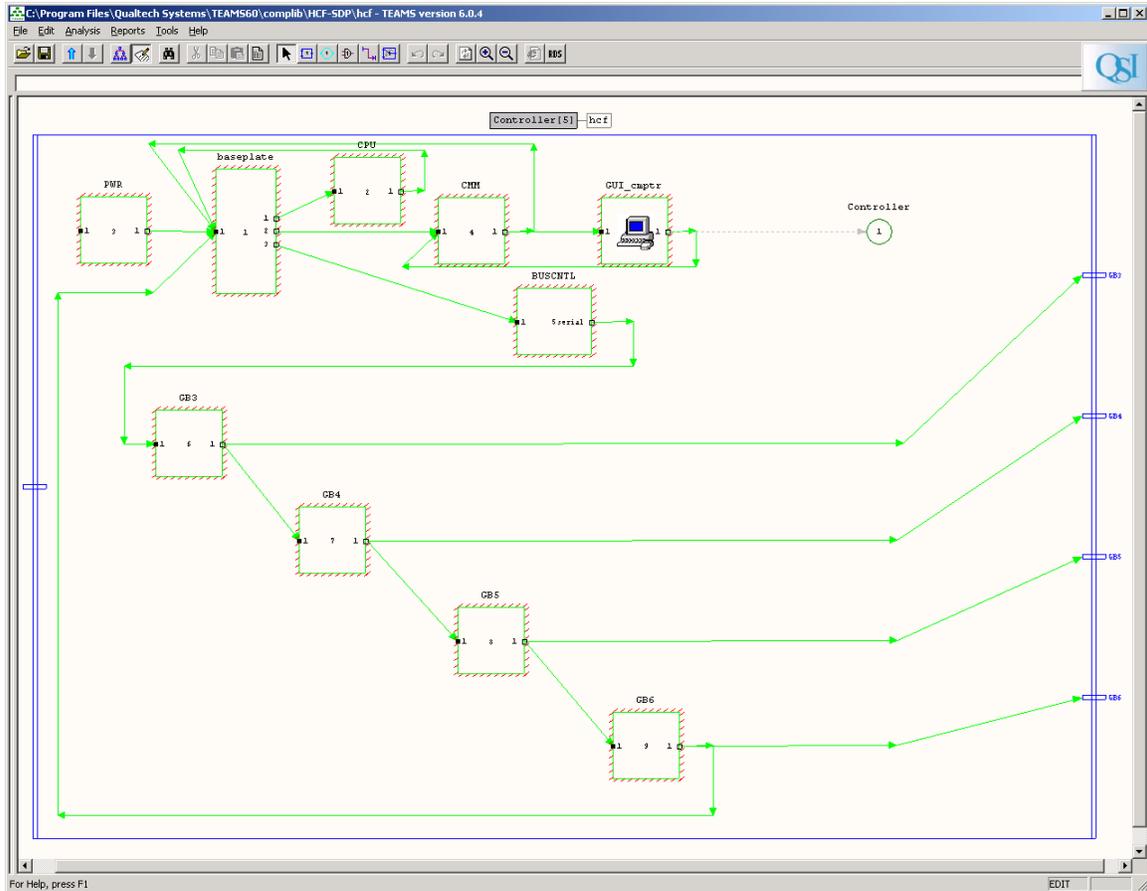


Figure 13: Top-level schematic of TEAMS HCF model.

### 7.1.1 Controller Subsystem

The controller uses an industrial Programmable Logic Controller (PLC) to control and monitor the LOX pumping process, the delivery of GOX to the combustion chamber, and the ignition of the fuel in the combustion chamber. Figure 14 shows the contents of the controller subsystem module. The circle on the schematic denotes a test point. Table 2 lists the components, tests, and model signals. The PLC and accompanying I/O blocks are mounted on a rack near the facility. Operator control and monitoring are achieved using a PC in the control room that is connected via Ethernet to the PLC. The power supply, CPU, Ethernet module, and Genius bus controller are mounted on a CPU baseplate. Four Genius blocks connected to the Genius bus controller handle the input and output signals to the facility hardware. The controller subsystem is modeled at a high level and details such as the individual wires connecting to each Genius block were not included. The software functions of the PLC ladder logic were not modeled. The feedback loops in the figure reflect the fact that the operator computer is used to control and monitor the components through the Ethernet interface. The high-level status bits monitored by the control system computer were used for assigning the signals of the components and tests. For example, there is a summary fault indication on the computer that indicates whether there is a problem with the CPU, I/O, LAN, or Genius blocks. Since this is a summary fault, we assign a signal (*PLC*) to every component in the

controller subsystem because a failure in any one of the components should trigger the fault indication. There are also status indications on the individual Genius blocks so we add a distinct signal to each one, as shown in the table. A similar procedure is followed for the other components and tests.



**Figure 14: Controller subsystem.**

<b>Component</b>	<b>Description</b>	<b>Signals</b>
PWR	GE 90-30 Power Supply	PLC
baseplate	CPU baseplate with I/O slots	PLC
CPU	CPU-360 PLC	CPU, PLC
CMM	Ethernet TCP/IP module	CMM, PLC
BUSCNTL	Genius bus controller	GBC(BEM), PLC
GB3	Genius block 3 – 24 VDC	GB3, PLC
GB4	Genius block 4 – Analog In/Out	GB4, PLC
GB5	Genius block 5 – 115 V	GB5, PLC
GB6	Genius block 6 – 24 VDC	GB6, PLC
GUI_cmprtr	Operator computer	CMM, CPU, GBC(BEM), PLC
<b>Test Point: Controller</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
PLCFLT	PLC: CPU, I/O, LAN, & Genius summary fault	PLC
CPUFLT	Alarm in PLC CPU fault table	CPU
GB3	Genius block 3 status bit	GB3
GB4	Genius block 4 status bit	GB4
GB5	Genius block 5 status bit	GB5
GB6	Genius block 6 status bit	GB6
GBC(BEM)	Genius bus controller status bit	GBC(BEM)
CMM	PLC interface LAN status	CMM

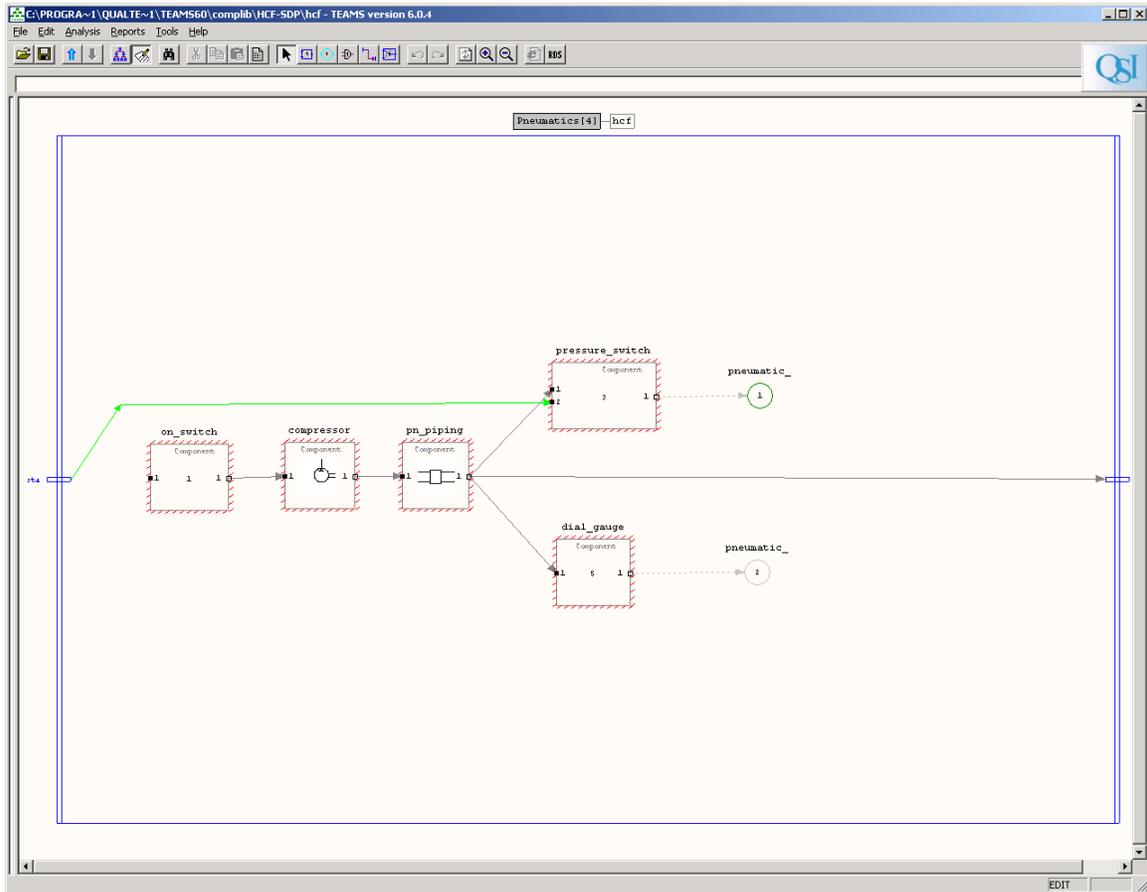
**Table 2: Controller subsystem components and tests.**

### 7.1.2 Pneumatic Subsystem

The pneumatic subsystem supplies compressed air to the pneumatically actuated valves. There is a pressure switch in the pneumatic plumbing that opens if the air pressure is too low and generates a fault indication in the controller. Figure 15 shows the contents of the pneumatic subsystem module. Table 3 lists the components, tests, and signals. Note that one can use observations such as whether the operator hears the compressor running for a test—attach a signal to the compressor and the same signal to a test that asks the operator if they hear the compressor running. We can associate test labels to tests that require observations, inspections, and other types of physical intervention and assess the effects of removing such tests on the fault isolation of the system; this might represent the reduction in fault isolation for an operational vehicle versus a vehicle at a ground-based test-bed, for example.

If the controller indicates that there is insufficient pneumatic pressure (test point 1), all of the components in the pneumatic subsystem are implicated (via the signal *pneumatic* which is attached to each component and to the *pneumatic\_pressure* test) as well as Genius block 6 in the controller, which is the I/O block for the pressure switch. The additional observations at test point 2 can be used to isolate the fault. If the compressor is

not making any sound (*comp\_sound*), then either the *on\_switch* or the compressor is suspected. If the compressor was initially running however, the compressor alone is implicated. If the dial gauge indicates that there is in fact sufficient pneumatic pressure in the line, then either the dial gauge is faulty or the pressure switch/Genius block is at fault.<sup>1</sup>



**Figure 15: Pneumatic subsystem.**

<sup>1</sup> Since the individual channels of the Genius blocks were not modeled, any functional failure of the Genius block that might affect a particular channel would be implicated at the sensor attached to that channel rather than at the Genius block since any other sensor connected to the block whose test passes implies that the block is working correctly.

Component	Description	Signals
on_switch	Electrical on switch/circuitry	comp_sound, comp_start_sound, pneumatic
compressor	Air compressor	pneumatic, comp_sound
pn_piping	Pneumatic piping	pneumatic
pressure_switch	Low pressure switch	(none)
dial_gauge	Analog pressure gauge	pneumatic
<b>Test Point 1: pneumatic_pressure_switch</b>		
Tests	Description	Signals
pneumatic_pressure	Low pressure indication at the controller (fail if yes)	pneumatic, GB6
<b>Test Point 2: pneumatic_pressure_obs</b>		
Tests	Description	Signals
compressor_sound	Audible compressor sound	comp_sound
dial_gauge	Analog pressure gauge reading	pneumatic
compressor_start_sound	Audible compressor sound at startup	comp_start_sound

**Table 3: Pneumatic subsystem components and tests.**

### 7.1.3 LOX Subsystem

The LOX subsystem is used to charge the GOX tank before a firing. The process of charging the GOX tank is a fairly manual one and includes opening and closing a number of hand valves in the LOX piping. Figure 16 shows the contents of the LOX subsystem module. Note the switch located between PRV-91 and POV-2. The switch has two positions: the down position is set when the GOX tank is being charged; the up position pertains to all other times. After the GOX tank is charged, valve POV-2 is closed and physically separates the LOX system from the GOX system. The switch in the model removes connectivity to the LOX components that are irrelevant to the failure manifestations during GOX operations. For example, if the GOX tank were not maintaining constant pressure after being charged and before the firing, we have no reason to implicate any components upstream of POV-2. The position of the switch is set externally to the model based on the command to POV-2. Table 4 lists the components, modules, signals, and tests at the top level of the LOX subsystem module. Subsequent sections describe the details of the modules and associated tests.

Also relevant to the LOX operations are the GOX tank and POV-4 in the GOX subsystem module and the test labeled “PIT3\_GOX\_charging” at test point 4 in Table 8. This test requires some feature extraction of the GOX pressure sensor measurement to determine the rate at which the pressure is increasing. If the pressure is not increasing at an expected rate (based on past experience) we conservatively assume that any of the components in the LOX subsystem, the GOX tank, or POV-4 at the outlet of the tank may be the cause and attach the signal *fluid* to these components and to the test

“PIT3\_GOX\_charging”. We also attach the signal *pneumatic* to test “PIT3\_GOX\_charging” since a failure of the pneumatic system would prevent POV-2 from opening and cause the test to fail. Other tests are used to reduce the number of suspected components.

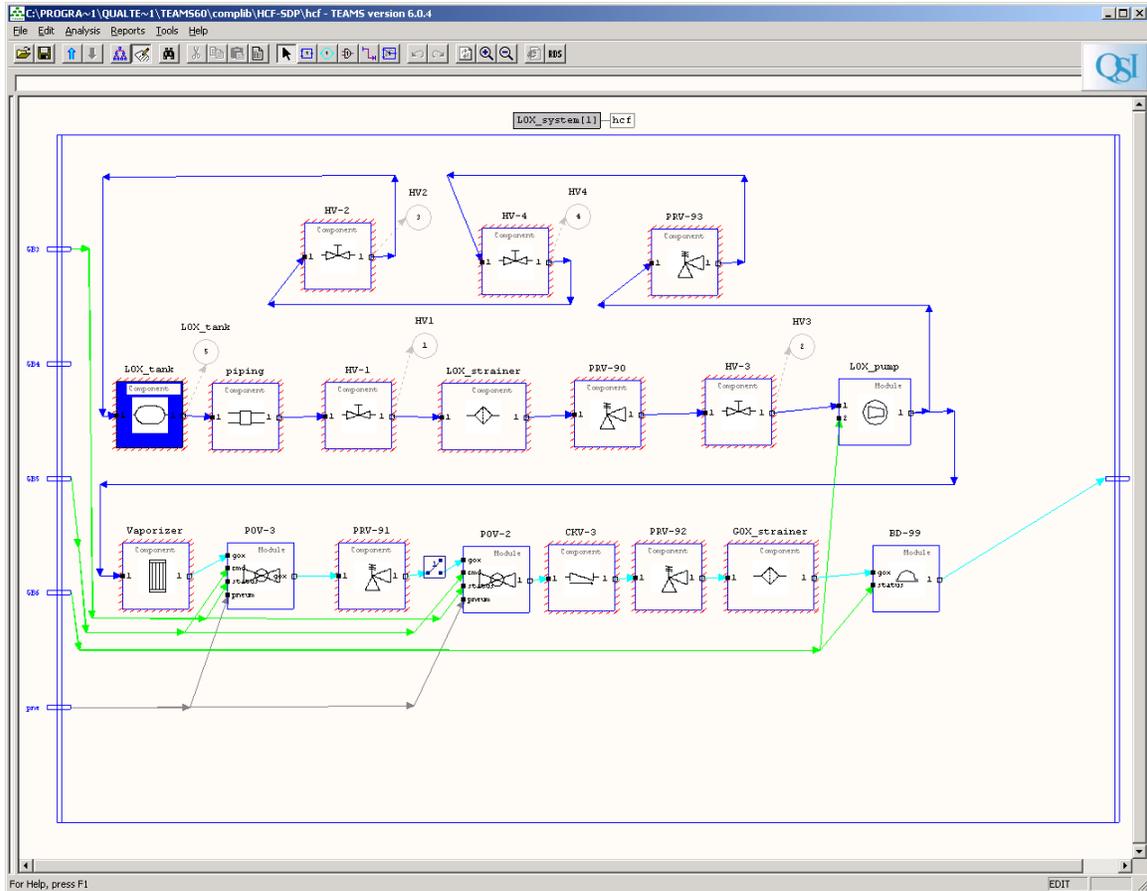


Figure 16: LOX subsystem module.

<b>Component or Module</b>	<b>Description</b>	<b>Signals</b>
LOX_tank	LOX storage	fluid
pipng	Pipes and connections	fluid
HV-1	LOX shutoff hand valve at LOX tank	fluid, HV-1
LOX_strainer	LOX particulate filter	fluid
PRV-90	Pressure relief valve between LOX tank and pump	fluid
HV-3	Hand valve for venting LOX line	fluid, HV-3
<i>LOX_pump</i>	LOX pump	see 7.1.3.1
PRV-93	Pressure relief valve in LOX return line	fluid
HV-4	Hand valve for purging LOX return line	fluid, HV-4
HV-2	Hand valve for LOX return line	fluid, HV-2
Vaporizer	Vaporize LOX to GOX	fluid
<i>POV-3</i>	GOX line vent valve	see 7.1.3.2
PRV-91	Pressure relief valve between LOX pump and shutoff valve	fluid
<i>POV-2</i>	Primary GOX line shutoff valve	see 7.1.3.2
CKV-3	Check valve in GOX line	fluid
PRV-92	Pressure relief valve between GOX tank and POV-2	fluid
GOX_strainer	GOX particulate filter	fluid
<i>BD-99</i>	Burst disk for emergency pressure relief	see 7.1.3.3
<b>Test Point 1: HV1</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
HV-1	Hand valve open	HV-1
<b>Test Point 2: HV3</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
HV-3	Hand valve closed	HV-3
<b>Test Point 3: HV2</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
HV-2	Hand valve open	HV-2
<b>Test Point 4: HV4</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
HV-4	Hand valve closed	HV-4
<b>Test Point 5: LOX_tank</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
LOX_tank	Liquid level, tank pressure, local tank tests	fluid

**Table 4: LOX subsystem components/modules and tests.**

### 7.1.3.1 LOX\_pump

The LOX\_pump module contains two components: a remote on-switch, and the pump itself. Figure 17 shows the contents of the LOX pump module. The pump has two failure modes as shown in Table 5. If the pump has a loss of prime, it will be detectable by an audible pitch change. Additionally, there is a LOX pump status displayed at the controller. Note the notation  $A \rightarrow B$  in the signals column. This denotes a signal mapping from signal A to signal B. For example, the signal *GB6* from the Genius block gets converted to the signal *pump* at the remote\_on\_switch component. A failure of the test “LOXpumpOK” implicates the signal *pump*, which will also implicate the signal *GB6* via the signal mapping. An alternative way of achieving this result is to attach both *pump* and *GB6* to the test (see the “pneumatic\_pressure” test in Table 3, for example). Signal mapping is useful to reduce the clutter in the model and is also a natural way of expressing the transformation of one type of input to a component into a different type of output.

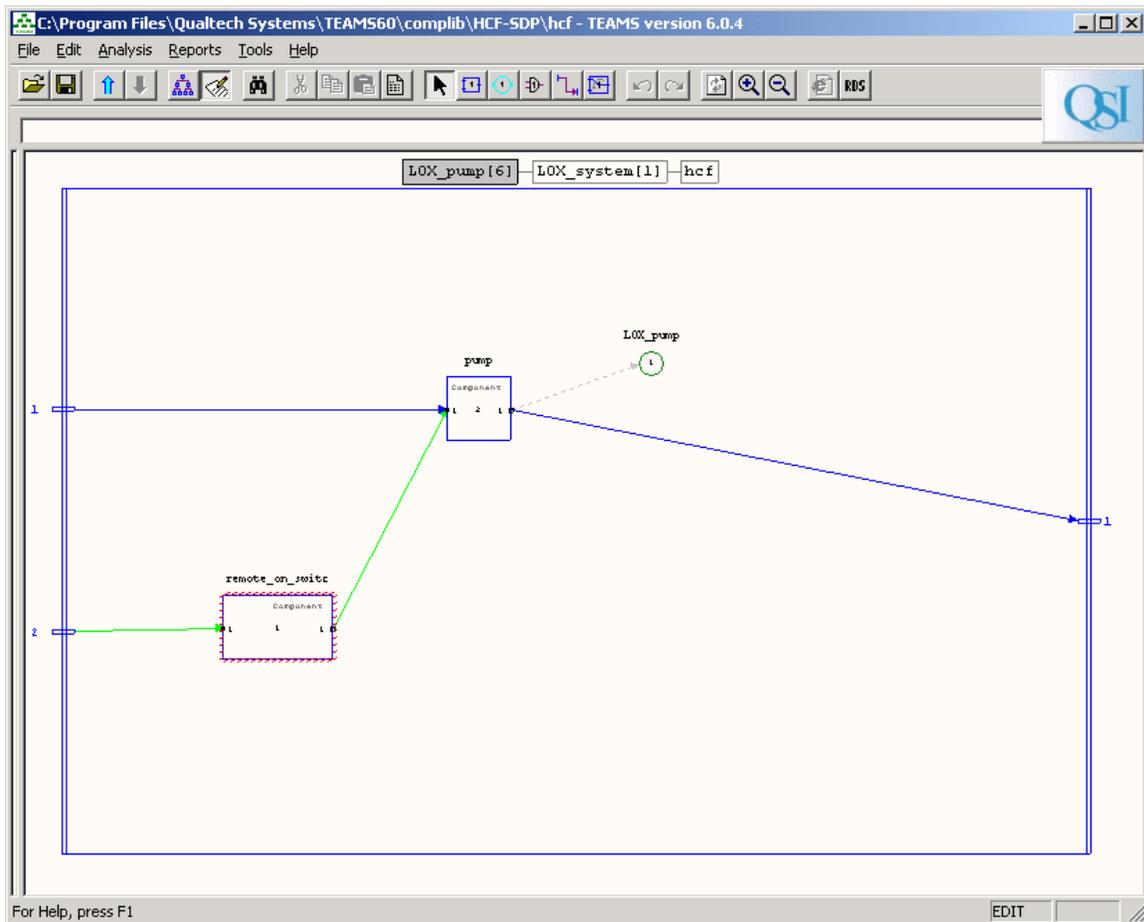


Figure 17: LOX pump module.

<b>Component:failure mode</b>	<b>Description</b>	<b>Signals</b>
remote_on_switch	Starts LOX pump	pump, fluid, GB6→pump
pump:loss_of_prime	Loss of pump prime	loss_of_prime, fluid
pump:pump_unknown	Other pump failures	pump, fluid
<b>Test Point: LOX_pump</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
loss_of_prime	Audible pitch change of pump (fail if yes)	loss_of_prime
LOXpumpOK	Pump status at controller	pump

**Table 5: LOX pump components and tests.**

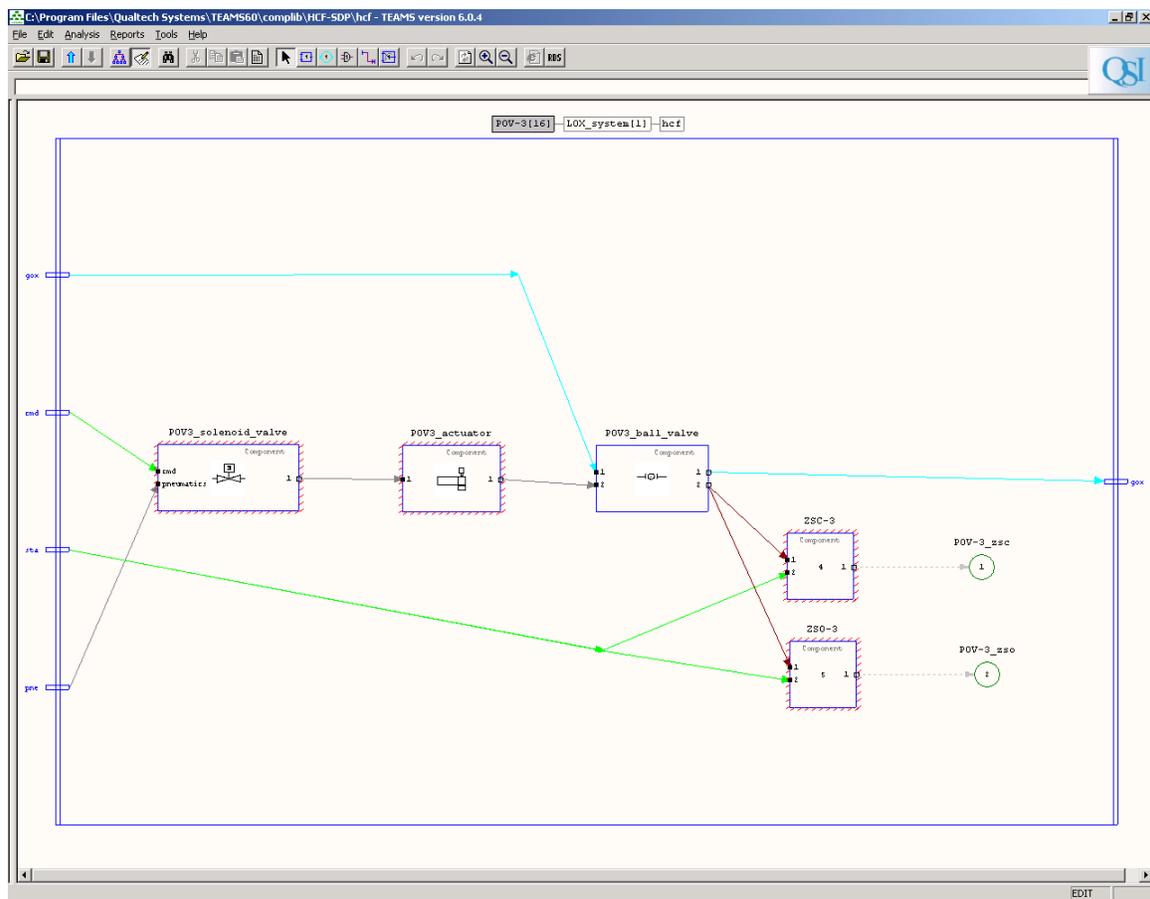
### 7.1.3.2 POV-3 & POV-2

Valves POV-3 and POV-2 are functionally similar and the models for them are the same apart from implementation details. Figure 18 shows the contents of the POV-3 valve module. Figure 19 shows the failure modes of the ball valve. POV-2 is similar. Table 6 lists the components, tests, and signals of the valve module. Code external to the model determines when a test is applicable. For example, after an open command has been sent to the valve, the tests labeled “POV3C\_open\_cmd” and “POV3O\_open\_cmd” would be set active and report a pass/fail test result to the real-time diagnostic engine whereas the tests labeled “POV3C\_close\_cmd” and “POV3O\_close\_cmd” would be inactive and would not report any test results. In this case, the active tests would verify whether or not the open command had the expected effect on the close and open limit switches—i.e., the open limit switch should report “open” and the close limit switch should report “notClosed”. If the valve did not open the tests will fail and at least one of the signals attached to each test is implicated as the cause. The signal *GB3* is attached to the tests to allow for the possibility that a bad Genius block will report the incorrect switch states. While it is highly unlikely that a Genius block fault will cause both tests to fail, the signal is attached to the tests to show the dependence of the reported switch state on the Genius block. The signal *pneumatic3* is attached to the open\_cmd tests and to the components that could prevent the valve from opening, thus causing the tests to fail. This includes the components in the valve as well as the pneumatic subsystem (via the signal mapping), since the valves are pneumatically actuated.

When the valves are commanded closed, the signals attached to the close\_cmd tests are attached only to the valve components since pneumatic subsystem failures will not cause the normally closed valves to fail open. We allow for the case of a valve being stuck in an intermediate position by attaching a signal to the appropriate valve failure mode and limit switch tests; when the valve is commanded open, the test on the close limit switch (POV3C\_open\_cmd) will pass since it will report “notClosed” but the test on the open limit switch (POV3O\_open\_cmd) will fail since it will report “notOpen”. Similarly, when the valve is commanded closed, the test on the open limit switch (POV3O\_close\_cmd) will pass since it will report “notOpen” but the test on the close

limit switch (POV3C\_close\_cmd) will fail since it will report “notClosed”. If we attach a unique signal (*inter3*) to the open limit switch test for the open command, the close limit switch test for the close command, and to the valve failure mode *POV3\_intermediate*, the valve will be implicated if the limit switches indicate “notOpen” and “notClosed” following a close or open command.

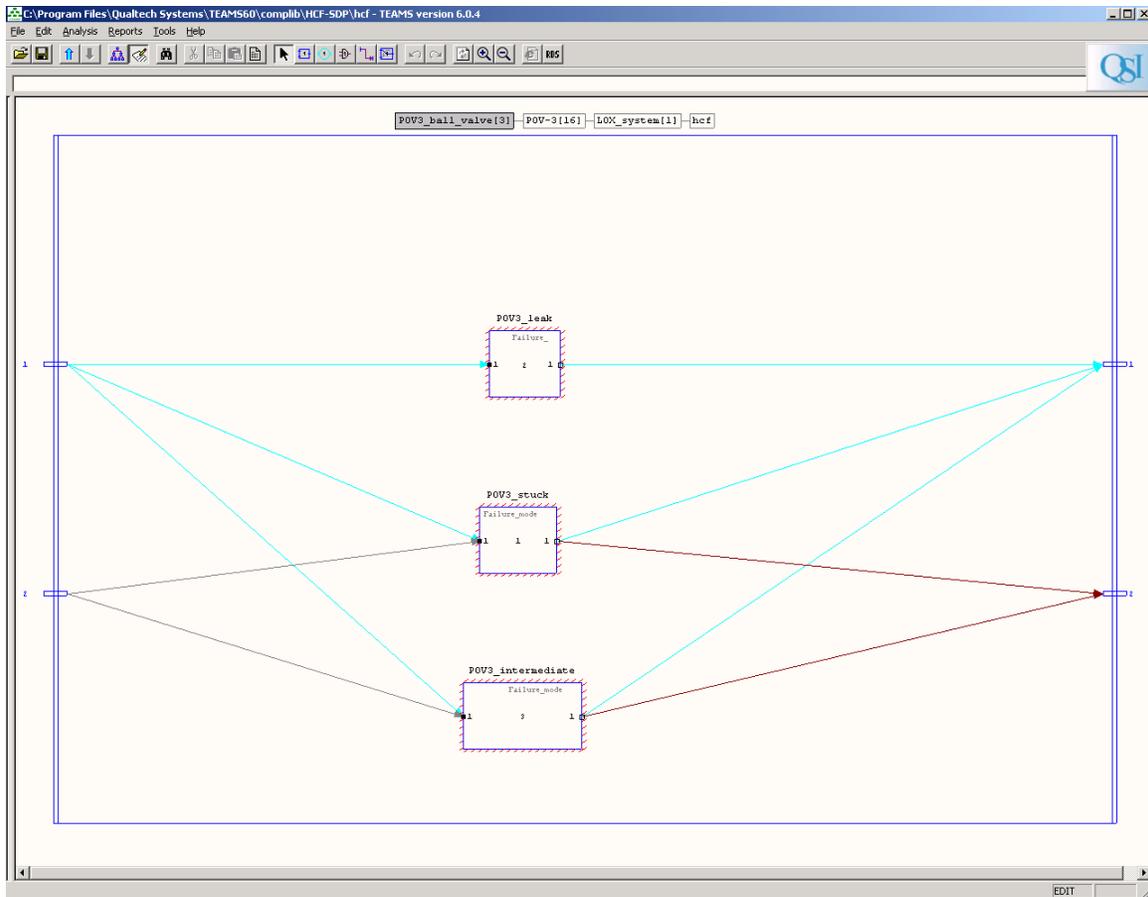
Another valve failure mode is included for a leaky valve. In this case, the limit switches will report the correct valve position but downstream tests will detect if there is an unexpected flow or pressure. Additionally, a faulty limit switch will be implicated if, for example, the limit switch reports closed but a downstream test on pressure or mass flow shows that the valve is in fact open.



**Figure 18: Valve module.**

<b>Component:failure mode</b>	<b>Description</b>	<b>Signals</b>
POV3_solenoid_valve	Opens to supply pneumatic air to actuator	fluid, pneumatic3, unexpected_open3, GB5→pneumatic3, pneumatic→pneumatic3
POV3_actuator	Rotates ball valve	fluid, pneumatic3, unexpected_open3
POV3_ball_valve:POV3_leak	Valve seat or assembly doesn't seal	fluid
POV3_ball_valve:POV3_stuck	Valve doesn't rotate either open or closed	fluid, pneumatic3, unexpected_open3
POV3_ball_valve:POV3_intermediate	Valve is between open and closed positions	fluid, inter3
ZSC-3	Close limit switch	(none)
ZSO-3	Open limit switch	(none)
<b>Test Point 1: POV-3_zsc</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
POV3C_close_cmd	Close limit switch "closed" after valve close command	inter3, unexpected_open3
POV3C_open_cmd	Close limit switch "notClosed" after valve open command	GB3, pneumatic3
<b>Test Point 2: POV-3_zso</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
POV3O_close_cmd	Open limit switch "notOpen" after valve close command	unexpected_open3
POV3O_open_cmd	Open limit switch "open" after valve open command	GB3, inter3, pneumatic3

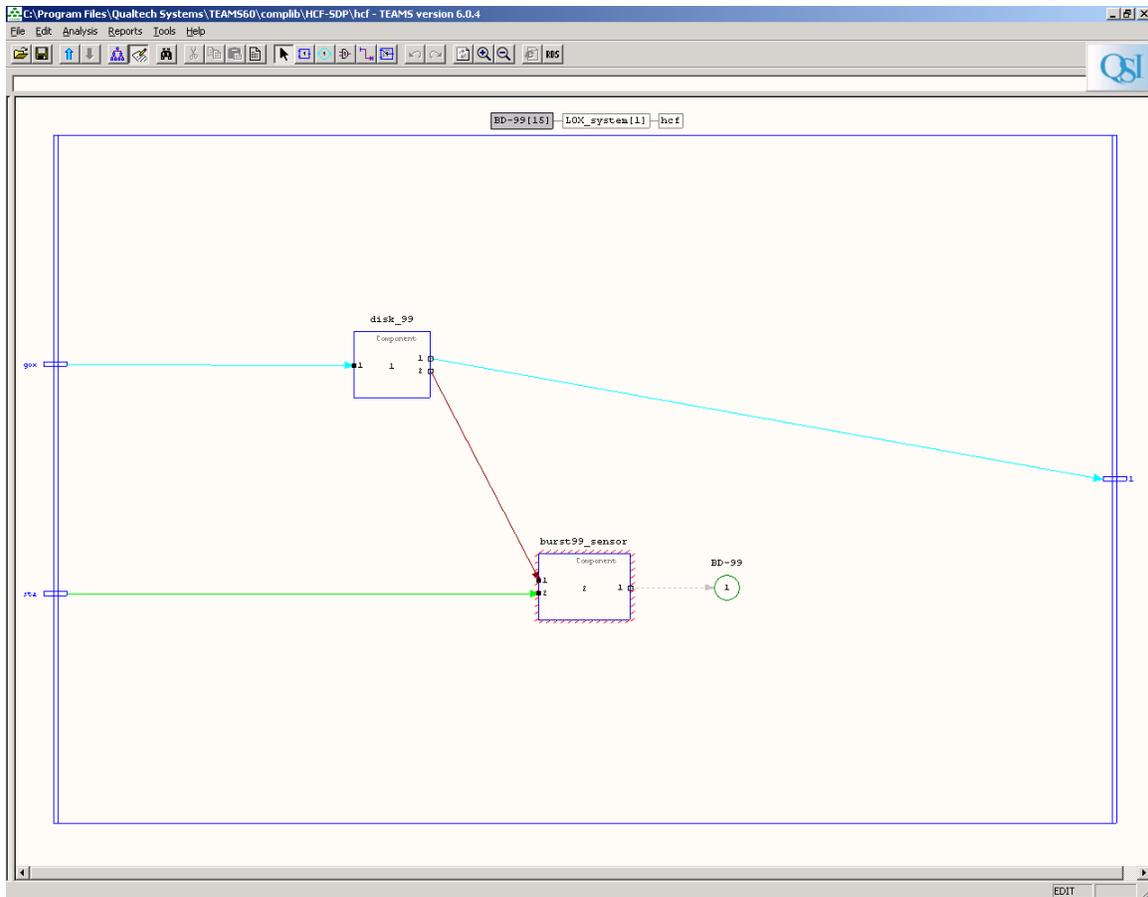
**Table 6: Valve components and tests.**



**Figure 19: Ball valve failure modes.**

### 7.1.3.3 Burst Disk 99

The burst disk is located just upstream of the GOX tank and provides emergency pressure relief in the event of an over-pressurization of the system. A continuity switch connected to the disk opens when the disk ruptures. Figure 20 shows the contents of the burst disk module. Table 7 lists the components, tests, and signals. The burst disk has two failure modes, one for when the burst disk ruptures and the other for when the pressure exceeds the rated pressure of the burst disk. The first failure mode would be detected by test point 1 within the burst disk module while the latter failure would be detected by a downstream test (see “PIT3\_burstdisk” in Table 8).



**Figure 20: Burst disk module.**

<b>Component:failure mode</b>	<b>Description</b>	<b>Signals</b>
burst99_sensor	Burst disk sensor	(none)
disk_99:burst99	Burst disk ruptures	burst99, fluid
disk_99:notburst99	Burst disk fails to rupture	notburst99
<b>Test Point 1: BD-99</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
burst99	Burst sensor indicates rupture (fail if yes)	burst99

**Table 7: Burst disk components and tests.**

### 7.1.4 GOX Subsystem

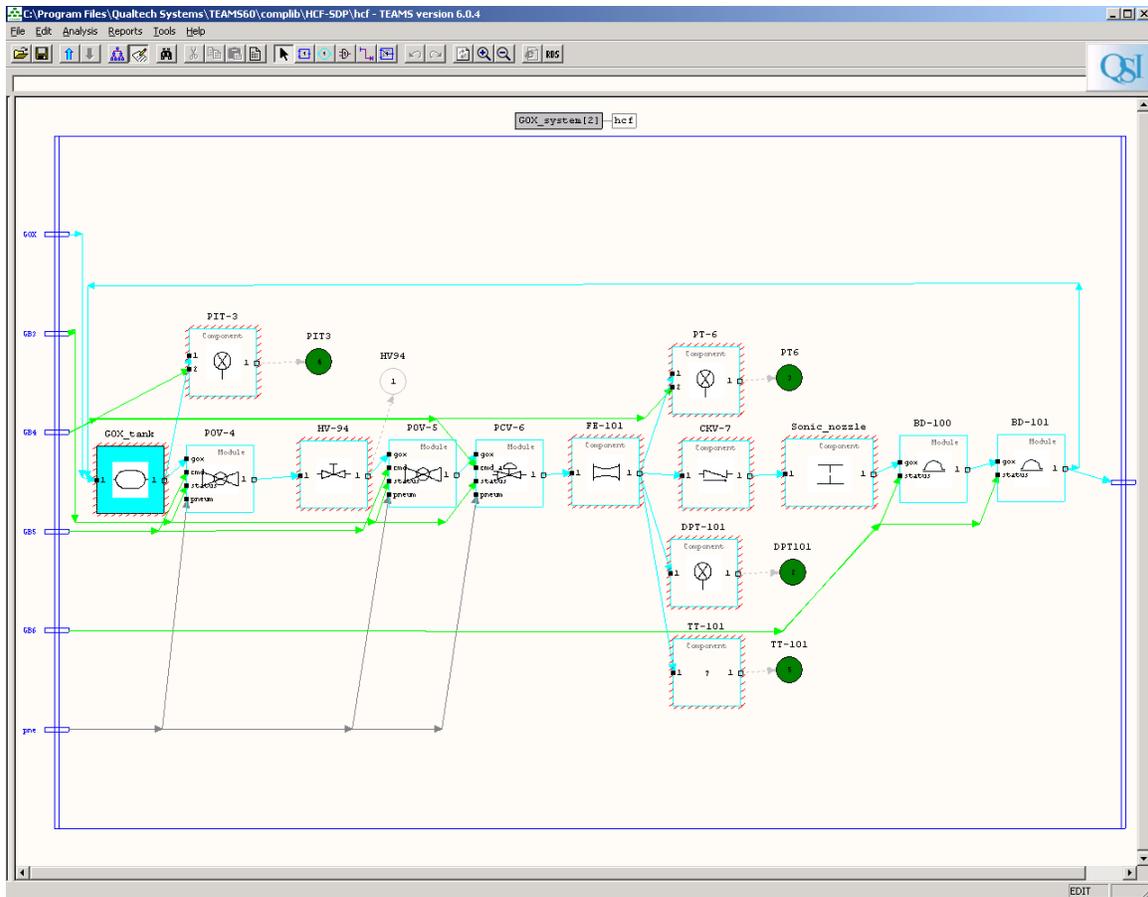
The purpose of the GOX subsystem is to deliver gaseous oxygen to the combustion chamber at a desired mass flow rate. Figure 21 shows the contents of the GOX subsystem module. Note the connection from BD-101 to the GOX\_tank. This accounts for the inherent feedback of pressure systems. For example, if one or both of the burst disks ruptured, the effects would be observed at the upstream pressure sensors. Table 8 lists the components, signals, and tests. Some of the tests are active only for certain portions of facility operations. Tests “PIT3\_GOX\_ready”, “PT6\_GOX\_ready”, and

“DPT101\_GOX\_ready” are monitored before POV-4 opens to begin the firing; tests “PIT3\_GOX\_firing”, “PT6\_firing\_level”, “PT6\_firing\_rate”, and “DPT101” are monitored during the firing when POV-4 is open; tests “PT6\_bleed” and “PIT3\_GOX\_ready” are monitored after POV-4 closes at the end of a firing. As mentioned previously, test code external to the model will report the relevant test results.

When POV-4 is closed before the firing, it is expected that the downstream pressure is ambient. If it is not, then either valve POV-4 is leaking (POV-5 is assumed to be open) or it has popped partially or fully open (highly unlikely). The latter cases would result in at least one of the POV-4 valve limit switch tests failing as well. We attach the signals associated with valve POV-4 leaking or failing open to the downstream tests that monitor the pressure. None of the other components are implicated, apart from faulty sensors, since they cannot produce the unexpected pressure. Additionally, the tank pressure should not be decreasing when POV-4 is closed. We attach the signal *fluid* to the test “PIT3\_GOX\_ready” because the signal is also attached to every component between POV-2 and POV-4 and a leak in any one of them would cause the test to fail.

When the facility is firing, we conservatively assume that a failure of any of the components in the GOX line could cause the delivery pressure to deviate from the desired setpoint and that the effects of the failure are observable at all pressure sensors in the line. While some failures are less likely than others, no failure probabilities were included in the model.

After the run is terminated by closing POV-4, the GOX line bleeds down to ambient pressure. We monitor the rate at which the pressure is decreasing with test “PT6\_bleed”. If the test fails, valve POV-4 is implicated. While it is true that a failure in one of the downstream components may cause the test to fail (e.g., part of the sonic nozzle breaks off), it is assumed that the failure would have been observed with the tests during the firing (i.e., the failure would not likely occur when depressurizing the system). The model could be easily modified to implicate these components during bleed by adding another signal, say *bleed*, to the components downstream of POV-4 and to test “PT6\_bleed”.



**Figure 21: GOX subsystem module.**

<b>Component or <i>Module</i></b>	<b>Description</b>	<b>Signals</b>
GOX_tank	GOX storage	fluid, gox
PIT-3	Pressure transducer	(none)
<i>POV-4</i>	Primary GOX shutoff valve	see 7.1.4.1
HV-94	Hand valve to bleed trapped pressure	gox, HV-94
<i>POV-5</i>	Secondary GOX shutoff valve	see 7.1.4.1
<i>PCV-6</i>	Pressure control valve	see 7.1.4.2
FE-101	Venturi for mass flow measurement	gox
PT-6	Pressure transducer	(none)
CKV-7	Check valve	gox
DPT-101	Differential pressure transducer for Venturi	(none)
TT-101	Thermocouple	TT-101
Sonic_nozzle	Isolates combustion line from feed line, flow measurement	gox
<i>BD-100</i>	Burst disk for emergency pressure relief	see 7.1.4.3
<i>BD-101</i>	Burst disk for emergency pressure relief	see 7.1.4.3

<b>Test Point 1: HV94</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
HV-94	Hand valve closed	HV-94
<b>Test Point 2: DPT101</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
DPT101	Differential pressure within expected range for run parameters	GB5, fluid, gox, pneumatic
DPT101_GOX_ready	No differential pressure before firing starts	inter4, leak, unexpected_open4
<b>Test Point 3: PT6</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
PT6_GOX_ready	Ambient pressure before run	inter4, leak, unexpected_open4
PT6_firing_level	Steady state pressure level acceptable	GB5, fluid, gox, pneumatic
PT6_firing_rate	Steady state pressure free of oscillations or bumps	actuation
PT6_bleed	Pressure decreasing as expected after firing termination	inter4, leak, unexpected_open4
<b>Test Point 4: PIT3</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
PIT3_GOX_charging	Tank pressure increasing as expected during GOX charging process	GB5, fluid, pneumatic
PIT3_burstdisk	Pressure less than burst disk rating	notburst99
PIT3_GOX_ready	Pressure holding steady before run and after run	fluid
PIT3_firing	Pressure dropping as expected during firing	GB5, fluid, gox, pneumatic
<b>Test Point 5: TT-101</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
TT-101	GOX temperature	TT-101

**Table 8: GOX subsystem components/modules and tests.**

#### 7.1.4.1 POV-4 & POV-5

The discussion in section 7.1.3.2 is relevant for valves POV-4 and POV-5 as well with some slight modifications. First, valve POV-4 has the signal *leak* added to the POV4\_leak failure mode. This signal is also attached to downstream tests that check for ambient pressure downstream of a closed POV-4 valve (valve POV-5 is assumed to be open). We don't attach the signal *fluid* to these tests because we don't want to implicate

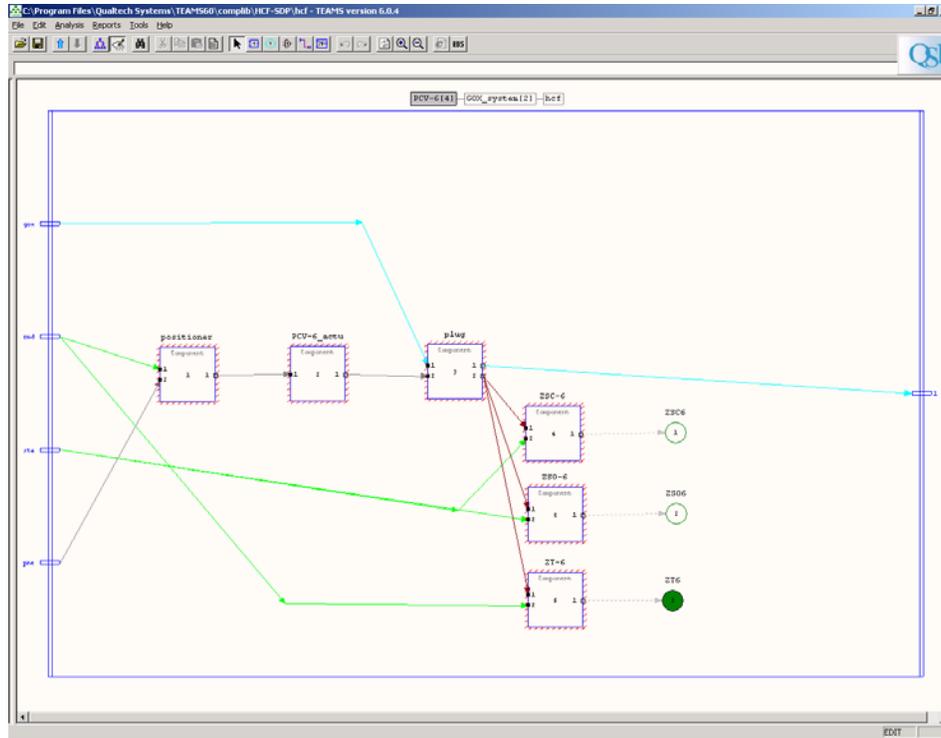
the components upstream of valve POV-4. Second, the signal *fluid* is replaced by the signal *gox* for POV-5. Third, signals, tests, and components with numbers (e.g., *inter3*) are modified appropriately (e.g., *inter4* or *inter5*).

#### **7.1.4.2 PCV-6**

As the pressure of the oxygen in the GOX tank decreases during a firing, the control valve opens to maintain a constant delivery pressure to the combustion chamber. A valve positioner converts a 4-20 mA control signal to a pneumatic pressure that is fed to one side of the diaphragm on the valve actuator. The control system uses a pressure feedback from the pressure sensor upstream of the sonic orifice (PT-6) in a Proportional Integral Derivative (PID) loop to calculate the valve position. Figure 22 shows the contents of the control valve module. Table 9 lists the components, tests, and signals.

A position feedback sensor (ZT6) on the valve is monitored to determine if the valve position is tracking the commanded position. The test must be designed to allow for mechanical play and hysteresis (stiction) in the valve. While different approaches may be taken to quantify the valve tracking, the test result passed to TEAMS must be binary (pass/fail).

Because of the closed-loop feedback control, there is a potential that the action of the PID loop to control pressure will mask failures in the system. For example, if there is a leak in the GOX line the control valve will open faster than normal to maintain the setpoint pressure. It is possible that all of the tests defined in the model will pass since we have said nothing about the command to the valve. We might wish to compare the command to the valve to a reference value for the given initial conditions in order to detect potential failures in other components. This was not done in the current study.



**Figure 22: Control valve module.**

Component	Description	Signals
positioner	Converts current input to pressure output	actuation, GB4 → actuation, pneumatic → actuation
PCV-6_actuator	Pneumatic actuator	actuation
plug	Valve plug	gox
ZSC-6	Closed limit switch	(none)
ZSO-6	Open limit switch	(none)
ZT-6	Position feedback sensor	(none)
<b>Test Point 1: ZSC6</b>		
Tests	Description	Signals
ZSC6	Valve reports closed after commanded closed	actuation
<b>Test Point 2: ZSO6</b>		
Tests	Description	Signals
ZSO6	Valve reports open after commanded 100% open	actuation
<b>Test Point 3: ZT6</b>		
Tests	Description	Signals
ZT6	Valve response tracks command acceptably	actuation

**Table 9: Control valve components and tests.**

### 7.1.4.3 Burst Disks 100 & 101

The burst disks in the GOX subsystem are similar to the one in the LOX subsystem but with the signal *fluid* replaced by *gox*. Refer to section 7.1.3.3 for a discussion of the burst disk module.

### 7.1.5 Ignition Subsystem

A short time after the flow of gaseous oxygen reaches the combustion chamber, the ignition system is commanded to ignite the fuel. A spark ignites methane and oxygen supplied from two K-cylinders and a high velocity gas burner injects the hot jet of combustion products into the pre-combustion chamber together with the oxygen from the GOX line, which causes the fuel grain to ignite. Metering valves in the igniter oxygen and methane lines control the mixture of fuel and oxidizer in the igniter. Additional fuel can be injected into the pre-combustion chamber for reliable ignition (this has not been done at the facility but the related components have been kept in the model). Figure 23 shows the contents of the ignition subsystem module. Table 10 lists the components, tests, and signals.

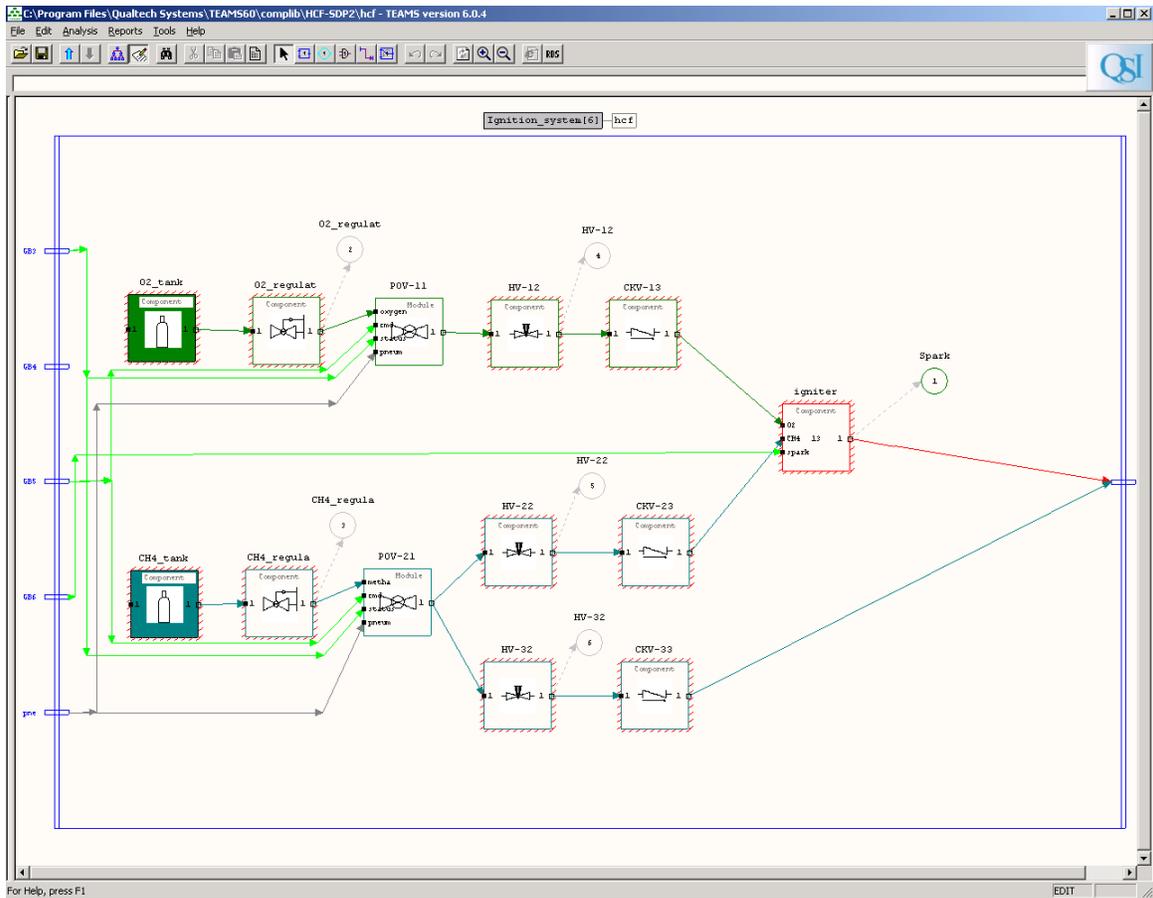


Figure 23: Ignition subsystem.

<b>Component or <i>Module</i></b>	<b>Description</b>	<b>Signals</b>
O2_tank	Oxygen K-cylinder	ignition
O2_regulator	Oxygen pressure regulator	ignition
<i>POV-11</i>	Oxygen shutoff valve	see 7.1.5.1
HV-12	Oxygen metering valve	ignition, HV-12
CKV-13	Oxygen line check valve	ignition
CH4_tank	Methane K-cylinder	ignition
CH4_regulator	Methane pressure regulator	ignition
<i>POV-21</i>	Methane shutoff valve	see 7.1.5.1
HV-22	Methane metering valve	ignition, HV-22
CKV-22	Methane line check valve	ignition
HV-32	Methane aux. metering valve	ignition, HV-32
CKV-32	Methane aux. line check valve	ignition
igniter	Gas-gas igniter	ignition, spark, GB6→spark
<b>Test Point 1: Spark</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
spark	Status of ignition circuit	spark

**Table 10: Ignition subsystem components and tests.**

### 7.1.5.1 POV-11 & POV-21

The discussion in section 7.1.3.2 is relevant for valves POV-11 and POV-21 as well with some slight modifications. The signal *leak2* is added to the leak failure mode on the ball valve for both valves and the signal *fluid* in Table 6 is changed to *ignition*. Also, signals, tests, and components with numbers are modified appropriately.

### 7.1.6 Combustion Chamber Subsystem

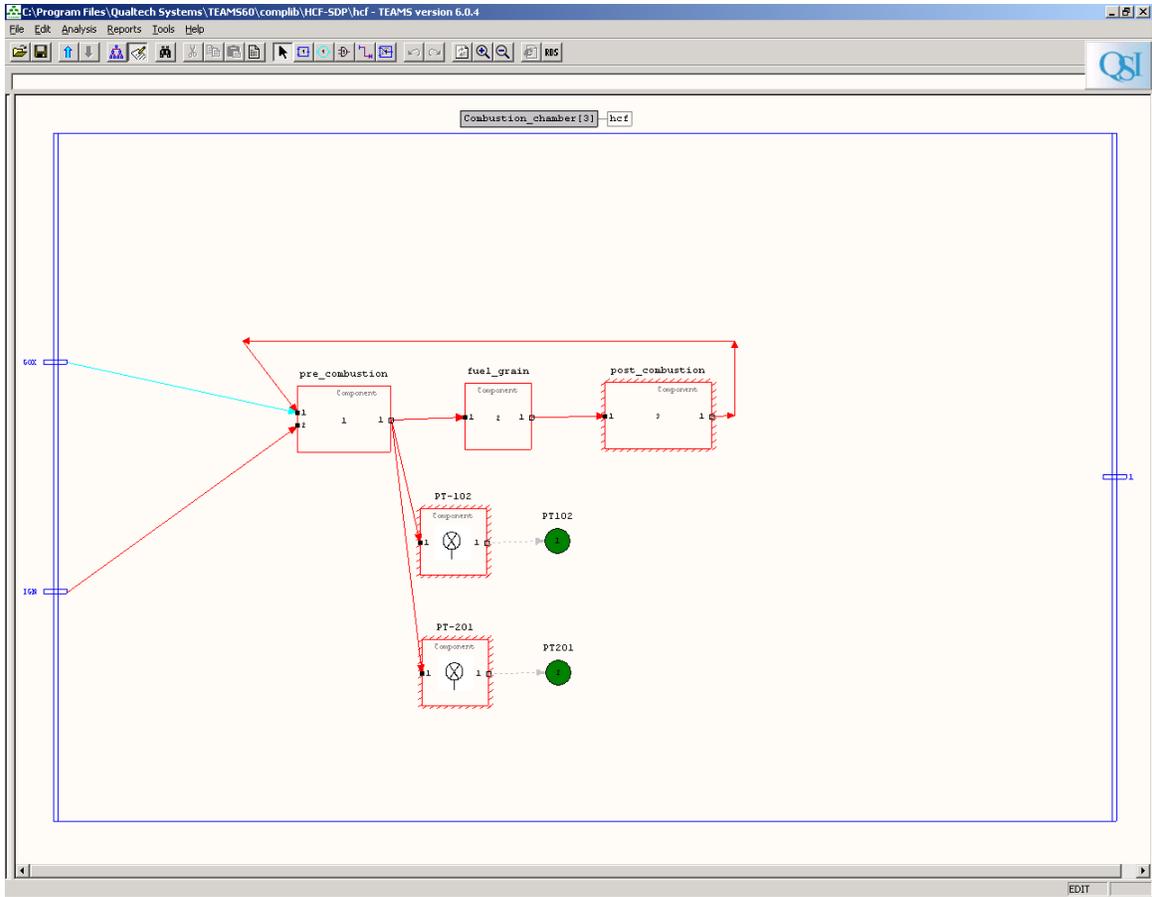
The combustion chamber holds the paraffin-based fuel. Gaseous oxygen from the GOX line and the hot ignition jet from the ignition system enter the pre-combustion chamber axially and radially, respectively. The combustion chamber pressure is measured in the pre-combustion chamber because of the lower temperatures compared to other locations. Two transducers measure the pressure; one of them is a dynamic pressure transducer that is useful for measuring unsteady pressure fluctuations. The post-combustion chamber enhances mixing of the unburned fuel and converts the thermal energy to directed kinetic energy via a nozzle. Figure 24 shows the contents of the combustion chamber subsystem module. Table 11 lists the components, tests, and signals.

The combustion chamber was modeled at a high level and details such as insulators and o-rings were not included. The tests at sensor PT102 are meant to be active sequentially: “GOX\_ready” before the primary shutoff valve POV-4 is opened, “pre\_ignition” after POV-4 has opened but before the ignition of the fuel, “ignition” after the command is sent to ignite the fuel, “firing” after the ignition, and “bleed” after POV-4 is closed to

terminate the run. As mentioned previously, test code external to the model will decide when the tests will be active. In each case, the tests must make a determination as to whether the measured pressure is close to what is expected by simple threshold tolerances or more sophisticated techniques. If a test fails, we associate a cause by the signals attached to the test.

The “GOX\_ready” test checks to see that the pressure at PT102 is ambient prior to the main shutoff valve opening. If it is not then we suspect either POV-4 or at least one of the valves in the ignition system is malfunctioning. After the command to open POV-4, we expect to observe the pressure in the combustion chamber rising. A failed “pre-ignition” test will implicate components in the GOX, pneumatic, combustion, and ignition (e.g., a valve inadvertently opens) subsystems. Other tests in those subsystems will provide more fault isolation. The ignition command should produce a sharp rise in chamber pressure. A failure of the “ignition” test will point to problems in the ignition system or combustion chamber. Once the fuel grain has ignited, the pressure level and rate of change are monitored. The most likely cause of problems will be in the combustion chamber itself but we allow for failures in the GOX or ignition systems to cause the “firing” test to fail. After the run terminates, the “bleed” test will monitor if the pressure is decreasing as expected. An unexpected fire in the combustion chamber or valves sticking or popping open could cause this test to fail.

The test “PT201\_frequency” requires analysis of the dynamic pressure trace to determine if there were any combustion instabilities during the burn. An FFT of the signal could show unexpected frequencies with significant amplitudes. While the presence of instabilities might be a result of the overall combustion chamber geometry, for simplicity the signal *frequency* is attached only to the fuel grain component.



**Figure 24: Combustion chamber subsystem.**

<b>Component:failure mode</b>	<b>Description</b>	<b>Signals</b>
pre_combustion:ignition	Pre-combustion chamber ignition failure	ignition
pre_combustion:combustion	Pre-combustion chamber failure	combustion
fuel_grain:instability	Combustion instability	frequency
fuel_grain:combustion	Structural or other failure	combustion
fuel_grain:ignition	Ignition failure	ignition
post_combustion	Post-combustion chamber	combustion
PT-102	Low frequency pressure transducer	(none)
PT-201	Dynamic pressure transducer	(none)
<b>Test Point 1: PT102</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
PT102_GOX_ready	Ambient pressure before run	inter11, inter21, inter4, leak, leak2, unexpected_open11, unexpected_open21, unexpected_open4
PT102_pre_ignition	Pressure increases as expected before ignition	GB5, fluid, gox, inter11, inter21, pneumatic, unexpected_open11, unexpected_open21
PT102_ignition	Fuel ignites successfully	GB5, ignition, pneumatic
PT102_firing	Chamber pressure level acceptable	combustion, fluid, gox, inter11, inter21, unexpected_open11, unexpected_open21
PT102_bleed	Pressure discharging as expected	combustion, inter11, inter21, inter4, leak, leak2, unexpected_open11, unexpected_open21, unexpected_open4
PT102_burstdisks	Pressure does not exceed burst disk rating	notburst100, notburst101
<b>Test Point 2: PT201</b>		
<b>Tests</b>	<b>Description</b>	<b>Signals</b>
PT201_frequency	No instabilities in pressure trace	frequency

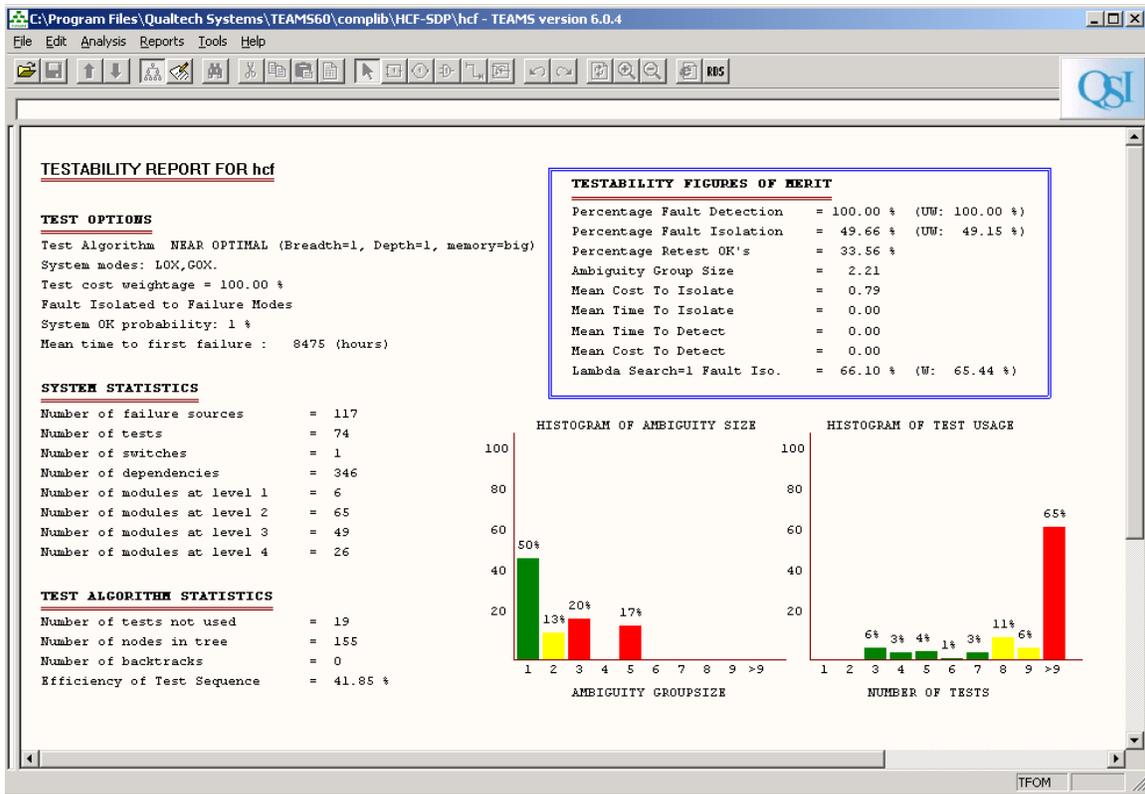
**Table 11: Combustion chamber components and tests.**

### 7.1.7 Testability analysis

The testability of the HCF system was analyzed using TEAMS to characterize the expected fault coverage utilizing the existing facility instrumentation. The testability analysis produces a summary report call the Testability Figures of Merit (TFOM). This report includes:

- 1) Percent of fault detection
- 2) Percent of fault isolation
- 3) Percent Retest OK at the desired isolation level
- 4) Average ambiguity group size
- 5) Mean cost to isolate
- 6) Mean time to isolate
- 7) Mean time to detect
- 8) Mean cost to detect
- 9) Percent fault isolation when all failures in an ambiguity group except for the most likely failure are ignored (the so-called  $\lambda$ -Search).

Details needed for items 3, 5, 6, 7, 8, and 9 were not included in the model. The TFOM for the HCF model is shown in Figure 25. The analysis shows that with the current measurements of the facility instrumentation and model scope, there would be 100% fault detection and a fault isolation of 50% with an ambiguity group size of just over 2. The testability analysis could be redone after changing the TEAMS model to include new facility instrumentation and/or expanding the scope of the model to show the change in fault detection and fault isolation.



**Figure 25: Testability Figures of Merit for the HCF model using all tests.**

Figure 26 shows the TFOM for the HCF model using only the tests that can be answered using logged data. As expected, if we don't use the information that is available to persons observing the facility, the fault detection and isolation is reduced.

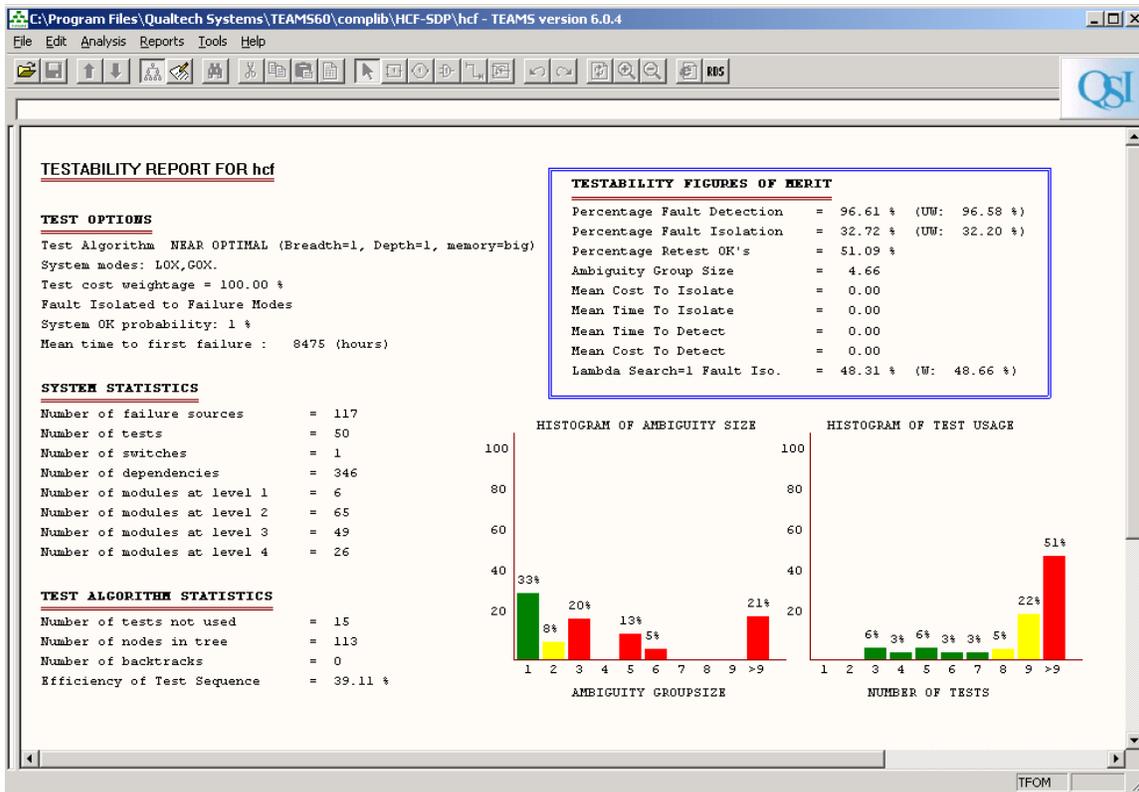


Figure 26: Testability Figures of Merit for the HCF model using logged data.

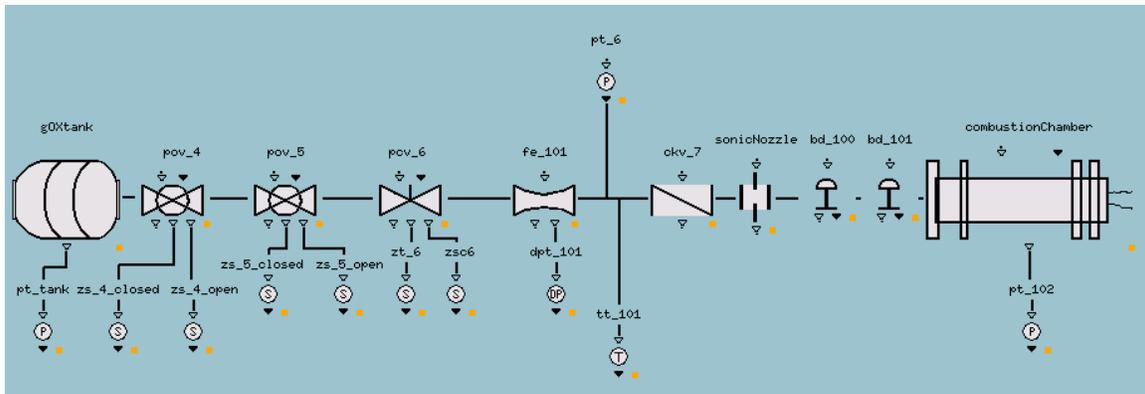
### 7.1.8 Real Data to TEAMS

Facility data can be supplied to the model to perform real-time fault detection and isolation. QSI developed a LabVIEW application to interface continuous and discrete sensor data to a TEAMS model. This interface was developed in a Phase III effort titled “An Onboard Real-time Aircraft Diagnosis and Prognosis System”, NASA contract number NAS2-01078. The data are stored in an Access database that can be played back and thresholded, with the subsequent test results read by the TEAMS-RT real-time reasoner/diagnostic tool. This simulation tool is useful in the pre-deployment phase of the diagnostic software to analyze the responses of the reasoner to real data. It can also be used for post-test fault detection and isolation studies. TEAMS-RT can be embedded as a real-time application with monitoring software that performs the desired tests and then sends the test results to TEAMS-RT. A real-time system was not developed in this study, but one has been developed and is currently in use on a UH-60 helicopter at Ames Research Center [10].

## 7.2 L2

The scope of the L2 model includes the GOX feed and combustion subsystems. Figure 27 shows the top-level schematic of the qualitative model. The main components of the model are the GOX tank, shutoff valves (POV-4 and POV-5), control valve (PCV-6), venturi (FE-101), check valve (CKV-7), sonic nozzle, burst disks (BD-100 and BD-101), and the combustion chamber. The model also includes many sensors. The pressure sensors include PT-Tank (tank pressure), DPT-101 (differential pressure in the venturi), PT-6 (GOX delivery pressure), and PT-102 (chamber pressure). Valve limit switches provide the position statuses of the shutoff valves. The open limit switches ZS-4-open and ZS-5-open indicate whether POV-4 or POV-5, respectively, are open or not open. Similarly, close limit switches ZS-4-closed and ZS-5-closed report whether these valves are closed or not closed. ZSC6 is the close limit switch on valve PCV-6; the open limit switch was not included because the valve does not normally go full open during a firing. There is also a position feedback sensor, ZT-6, on the control valve that reports the percentage open of the valve. There is one temperature sensor, TT-101. Pipes are not included in the model as they are passive components that are not likely to fail. In addition, it would be difficult to isolate a pipe failure with the current instrumentation.

The small triangles in the figure are the input and output terminals of the components. Discretized sensor values and commands are associated with the filled triangles; unfilled triangles are terminals that are connected to other terminals in the model, thereby propagating information between the components.



**Figure 27: Livingstone HCF model.**

### 7.2.1 Feedline Contents

The connections between the components in the GOX line are of a structured data-type called *feedline*. *Feedline* connections propagate properties of the oxygen, namely pressure and temperature, throughout the model and can be thought of as pipes that do not fail.

The pressure characteristics in *feedline* are level, rate (of change), and the determination of whether the pressure is above or below the burst disk threshold. The pressure level is

discretized into ambient, low, nominal, and high. The thresholds separating the bins are set based on location of the component and the expected pressures during the run. For example, the nominal bin for components downstream of the control valve and upstream of the sonic nozzle includes a band around the desired steady-state GOX delivery pressure that is used in the pressure-control feedback loop. Upstream of the control valve, the nominal pressure range is wider because it must allow for the reduction of the tank pressure during the run. The pressure rate of change can take the values rise, steady, drop slow, and drop fast. Like the pressure level bins, the pressure rate bins are defined based on the location and expected pressure derivatives during the run. Finally, the burst disk threshold property is set to above threshold or below threshold depending on whether the pressure exceeds the burst pressure rating of BD-100 (and BD-101).

The temperature characteristic of the model is simply a level property of low, nominal, or high. The temperature readings remain relatively steady during the entire run, so all that is expected during the run is a nominal temperature level.

### 7.2.2 Sensors, Limit Switches, and Position Feedback

Inputs	Sensed value	
Outputs	Sensor reading	
Nominal Modes	Nominal	Input = output
Failure Modes	Faulty	No restrictions

**Table 12: Sensors, limit switches, and position feedback components.**

The sensors, limit switches, and position feedback are functionally similar and essentially report the value that they measure. They are all modeled in the same way. The input to the component is the property being measured and the output is the sensor indicated value. There is one *nominal* mode in which the input equals the output. There is also only one fault mode without any logic in it. This is the default *faulty* mode, and will be the mode of the component if the input and output are not equal.

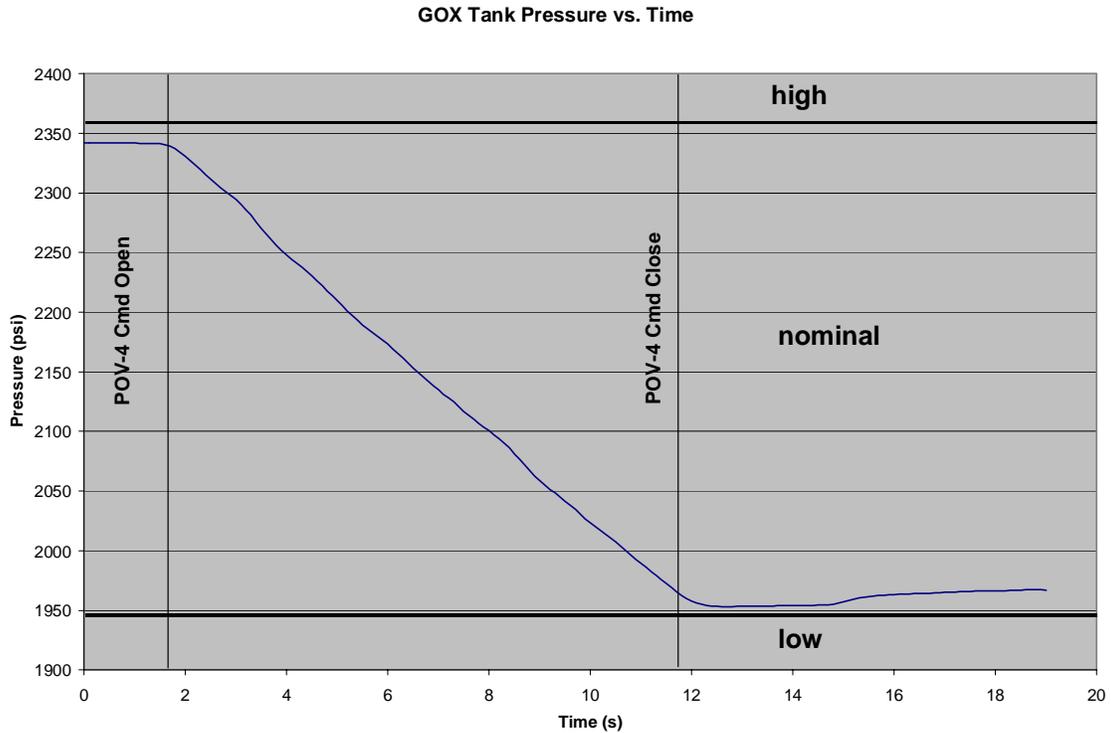
### 7.2.3 GOX Tank

Inputs	None		
Outputs	<i>Feedline</i>		
Attributes	State	Steady	Either shutoff valve closed
		Discharging	Both shutoff valves open
Nominal Modes	Nominal	Temperature out = nominal Pressure level out = nominal If state is steady, pressure rate out = steady If state is discharging, pressure rate out = dropping slow	
Failure Modes	Tank filled too much	Pressure level out = high	
	Leak	Pressure level out below nominal OR Pressure rate out = dropping	
	Unknown fault	No restrictions	

**Table 13: GOX tank component.**

There are no inputs into the GOX tank component. There is one output of type *feedline*, which carries with it the properties of the gas that is coming out of the tank. An attribute, *state*, is used to apply different constraints based on whether or not the system is configured to flow oxygen. The value of the attribute is set externally to the component (but not externally to the model) depending on the modes of the shutoff valves. If either valve is closed, the value of the attribute is set to steady; if both valves are open, the value is set to discharging. Consideration of the valve configuration includes nominal and failure modes; a valve that has failed stuck closed would cause the attribute to be set to steady.

A typical tank pressure trace is shown in Figure 28. Prior to POV-4 opening, the pressure remains steady. After POV-4 opens, the pressure decreases nearly linearly until the valve is closed at the end of the run. The pressure thresholds are set so that the upper bound of the nominal bin is just above the initial expected GOX tank pressure and the lower bound is just below the final expected pressure, which can be estimated using the desired run duration and mass flow rate. Thus, the pressure level is expected to be nominal throughout the run. In addition, the tank pressure is steady at the beginning and end of the run and drops slowly during the run. It should never rise or drop too fast at any time during a nominal run. (Figure 28 shows a slight rise in the pressure at the end of the run, but we simply bin the tank pressure rate so that it is still steady when the pressure rises slowly as shown.)



**Figure 28: Example of typical GOX tank pressure trace with L2 bins.**

The component has one *nominal* mode, meaning that the tank is expected to stay in this mode during the run. In this mode we assert that the pressure level should be nominal and the pressure rate should be steady or dropping slow, depending on the *state* attribute. The model also includes three fault modes. One mode occurs if the tank is filled with too much oxygen and is characterized by a high pressure level. Another mode occurs if there is a leak and would be indicated by the pressure dropping when the tank should be maintaining constant pressure or by the tank pressure dropping too rapidly during a run. We also include the case of the pressure level being below nominal in this fault mode. Finally, there is an *unknown fault* mode, with no restrictions. This fault mode is assigned a very low probability.

Currently, the model of the GOX tank does not allow for the possibility that a downstream failure might cause the pressure to drop at a different rate than what is expected during a nominal run. If a downstream leak causes more GOX to flow out of the tank to compensate for it, the pressure rate and possibly pressure level will not satisfy the propositions in the *nominal* mode and will result in a failure indication of the GOX tank even though the cause of the deviation from nominal behavior is downstream.

## 7.2.4 Shutoff Valve POV-4

Inputs	Valve command (open, close)		
	<i>Feedline</i>		
Outputs	Open/not open limit switch status		
	Closed/not closed limit switch status		
	<i>Feedline</i>		
Always true	Pressure level out is qualitatively equal to or less than pressure level in		
Attributes	Valve position	Closed (not open & closed)	Pressure level out is ambient OR pressure rate out is dropping
		Intermediate (not open & not closed)	If pressure level in is not ambient, pressure level out is not ambient
		Open (open & not closed)	<i>Feedline</i> in and out are equal
Nominal Modes	Open	Valve position is open	
	Closed	Valve position is closed	
Failure Modes	Stuck open	Valve position is open	
	Stuck intermediate	Valve position is intermediate	
	Stuck closed	Valve position is closed	
	Unknown fault	No restrictions	

**Table 14: POV-4 component.**

The POV-4 component has an open/close command input and a *feedline* input that represents the flow of oxygen into the valve. A *feedline* output is the flow out of the valve and two limit switches report the valve status. A valve position attribute is defined to simplify the propositional statements in the nominal and failure modes, which refer only to the valve position. The constraints that apply to the three valve positions are given in the table above.

The component has two nominal modes, *open* and *closed*. When the valve is closed the open limit switch should report “not open” and the closed limit switch should report “closed”. If the valve has been closed for a long time we expect the pressure downstream of the valve to be ambient; however, if the valve has recently closed the downstream pressure could be higher than ambient but should be dropping. Therefore, in the logic for the *closed* mode we state that either the pressure level out is ambient or the pressure rate equals dropping. This is done so that a diagnosis can be done relatively soon after the close command rather than waiting for the pressure to reach the ambient level, which could take several seconds of a short run. When the valve is open the open limit switch should report “open” and the closed limit switch should report “not closed”. In the *open* mode we set the input and output to be equal. This makes an assumption that the valve is not choked, which is true except for a short time immediately after opening the valve. The open command will transition the mode from *closed* to *open* while the close

command will transition the mode from *open* to *closed* (assuming the transitions are nominal).

Four fault modes are included in the model and will be considered if the constraints in the intended nominal mode are not satisfied. Three of the fault modes are for a valve that is stuck in an open, intermediate, or closed position.

POV-4 is a fast-acting valve and opens and closes in roughly half a second. After a command is issued to the valve, a diagnosis would not be requested until the valve has completely transitioned so that the constraints in the intended mode are satisfied. Since pressures take more time to stabilize than limit switches, their values may be unassigned before a diagnosis to prevent conflicts or the constraints in the mode may be coded to allow for the observed behavior, as was done for the *closed* mode. The policy to request diagnoses and unassign observations is implemented in code external to the model.

### 7.2.5 Shutoff Valve POV-5

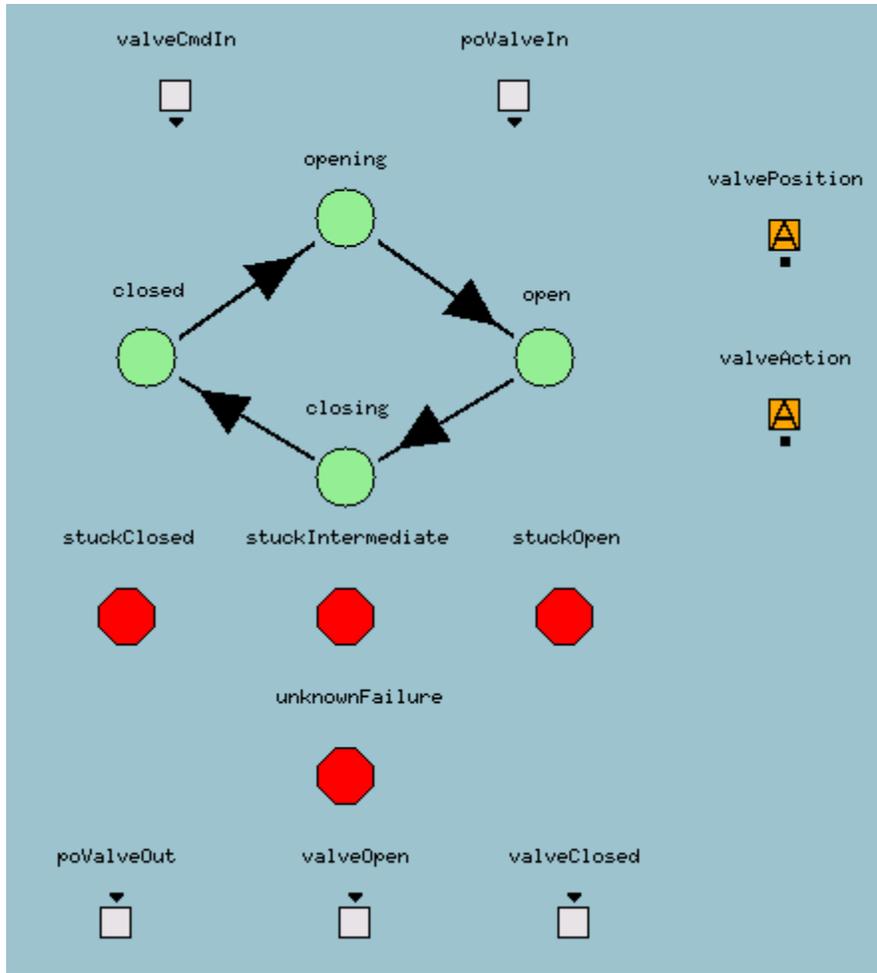
Inputs	Valve command (open, close, limitSwitchOpen, limitSwitchClosed)		
	<i>Feedline</i>		
Outputs	Open/not open limit switch status		
	Closed/not closed limit switch status		
	<i>Feedline</i>		
Always true	Pressure level out is qualitatively equal to or less than pressure level in		
Attributes	Valve position	Closed (not open & closed)	Pressure level out is ambient OR pressure rate out is dropping
		Intermediate (not open & not closed)	If pressure level in is not ambient, pressure level out is not ambient
		Open (open & not closed)	<i>Feedline</i> in and out are equal
	Valve action	Opening	If pressure level in is ambient, pressure rate out is steady; otherwise pressure rate out is rising or steady
		Closing	Pressure rate out is not rising
	Nominal Modes	Opening	Valve action is opening, valve position is not open
Open		Valve position is open	
Closing		Valve action is closing, valve position is not closed	
Closed		Valve position is closed	
Failure Modes	Stuck open	Valve position is open	
	Stuck intermediate	Valve position is intermediate	
	Stuck closed	Valve position is closed	
	Unknown fault	No restrictions	

**Table 15: POV-5 component.**

Redundant shutoff valve POV-5 is a slow-acting valve that takes approximately three seconds to close. Right after the command to close, the valve may not move at all until the pressure in the actuator bleeds down to the point where the spring-generated closing force can overtake the pressure force. It is undesirable to wait a few seconds to do a diagnosis so we must either unassign the limit switch feedback for a long while to avoid a conflict in the closed mode, or modify the shutoff valve model discussed in the previous section. Figure 29 shows a modified valve component that adds two transitory modes and a valve action attribute. In this model, when the valve is commanded to close it first transitions from the *open* to the *closing* mode. Since the valve may not rotate right away, the limit switches may report that the valve is in an open position or in an intermediate position and the pressure out of the valve may be dropping or remaining steady. The mode transitions from *closing* to *closed* when the closed limit switch indicates that the valve has closed. Here, we treat the limit switch observation as a command to trigger a transition. However, relying solely on the limit switch observation is insufficient because the limit switch could fail. If the limit switch failed the valve component would remain in the *closing* mode; the constraints in that mode would continue to be satisfied and the limit switch failure would go undetected. To remedy this, we must include in the policy that the closing to closed command should be issued either when the affirmative closed limit switch feedback is received or a specified period of time after the initial close command. The time condition will force a transition to the *closed* mode and implicate the faulty limit switch. Alternatively, one could transition strictly based on time and allow for the *closing* mode to have any limit switch indication. The *opening* mode is similar.<sup>2</sup>

---

<sup>2</sup> The valve opens faster than it closes; including an *opening* mode may be unnecessary but is included here to demonstrate the approach. The simpler timeout policy of POV-4 should be sufficient after a POV-5 open command.



**Figure 29: POV-5 component.**

### 7.2.6 Control Valve (PCV-6)

Inputs	Valve command (close completed, close, open low, open high)	
	<i>Feedline</i>	
Outputs	Position feedback	Position (closed, closed too much, open low, open high, open too much)
		Action (closing, not moving, opening slow, opening nominal, opening fast)
	Closed/not closed limit switch status	
	<i>Feedline</i>	
Attributes	PO valve status	Whether POV-4 is open or closed
	Downstream failure	Whether a downstream failure is indicated

Nominal Modes	Open low	Close limit switch reports “not closed”, valve position is not “closed” Pressure level out is less than or equal to pressure level in If there is no downstream failure: { Valve position is “open low” If and only if pressure level in is ambient, pressure level out is ambient and pressure rate out is steady If PO valve status is open, pressure rate out is not dropping If PO valve status is closed, <i>feedline</i> in equals <i>feedline</i> out }
	Open high	Close limit switch reports “not closed” If there is no downstream failure: { Valve position is open low or open high If pressure level in is high, pressure level out is high If PO valve status is open: If pressure level in is nominal: If valve position is open low, then pressure level out is not ambient, pressure rate out is rising or steady, valve action is opening nominal or opening fast If valve position is open high, then pressure level out is nominal, pressure rate out is steady, valve action is opening nominal If pressure level is low, then pressure level out is not nominal, valve action is opening fast If pressure level is ambient, then pressure level out is ambient and valve action is opening fast If PO valve status is closed, <i>feedline</i> in equals <i>feedline</i> out }
	Closing	Close limit switch reports “not closed” If valve position is not open high and there is no downstream failure, valve action is closing If PO valve status is closed, <i>feedline</i> in equals <i>feedline</i> out
	Closed	Close limit switch reports “closed” Valve action is not moving, valve position is closed Pressure level out is ambient or low Pressure rate out is steady or dropping slow
	Failure Modes	Too far closed
Too far open		Valve position is open high or open too much Close limit switch reports “not closed” Pressure level out is not less than pressure level in
Stuck open low		Valve is in open low position, valve action is not moving Pressure level out is less than or equal to pressure level in
Stuck open high		Valve is in open high position, valve action is not moving Close limit switch is “not closed”
Unknown fault		No restrictions

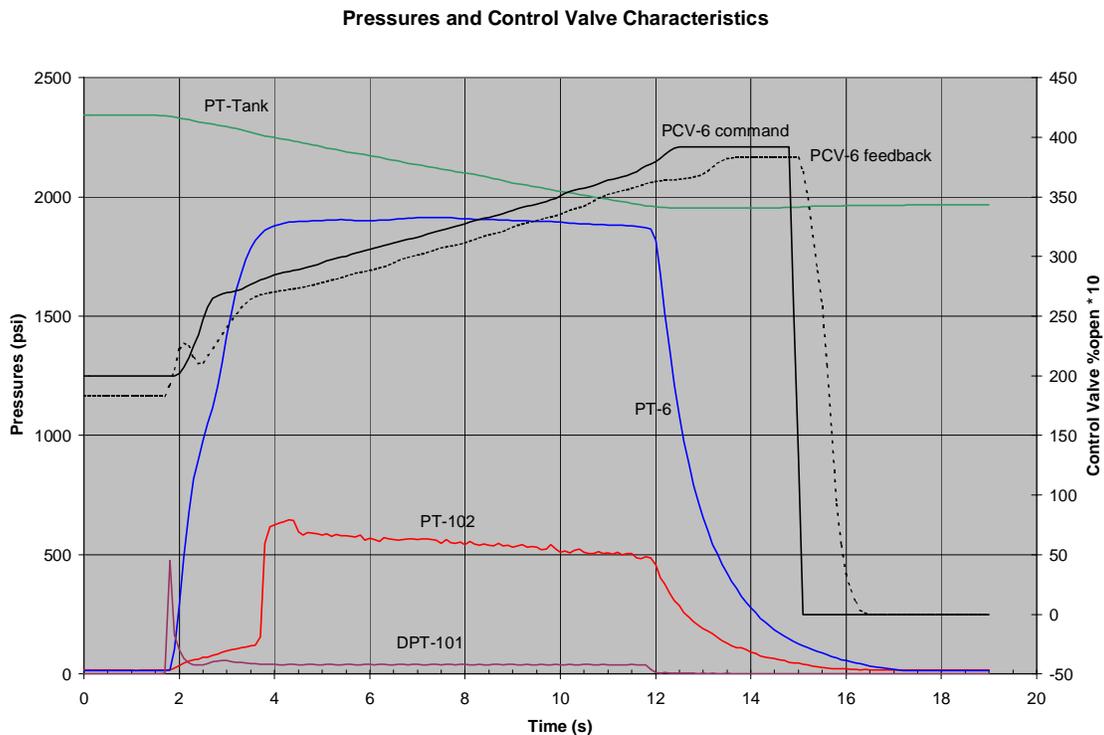
**Table 16: PCV-6 component.**

The PCV-6 component has a command input and a *feedline* input. Unlike the PO valves, the command to the valve is not simply an open/close command. The control valve command is a continuous signal from 0% open to 100% open. The discretization of this signal is described later. The component outputs include a *feedline* type that represents the GOX flow out of the valve, a closed limit switch status, and a valve position feedback. Two attributes are defined and used in the logic in the mode formulae. The values of the attributes are set externally to the component. The downstream failure

attribute is defined to try to deal with deviations from expected behavior due to faults in other components.

The primary difficulty in creating a model of the control valve is the continuous behavior of the valve. It is impractical to create many bins for the valve position and qualitatively describe the behavior in each bin, especially since the flow through the valve depends not only on valve position but also on the upstream and downstream pressures. Rather than attempting to capture the valve behavior based on a multitude of qualitative mappings of pressures and valve positions, we consider the intended function of the control valve and examine the data from a nominal firing to determine how to model the control valve.

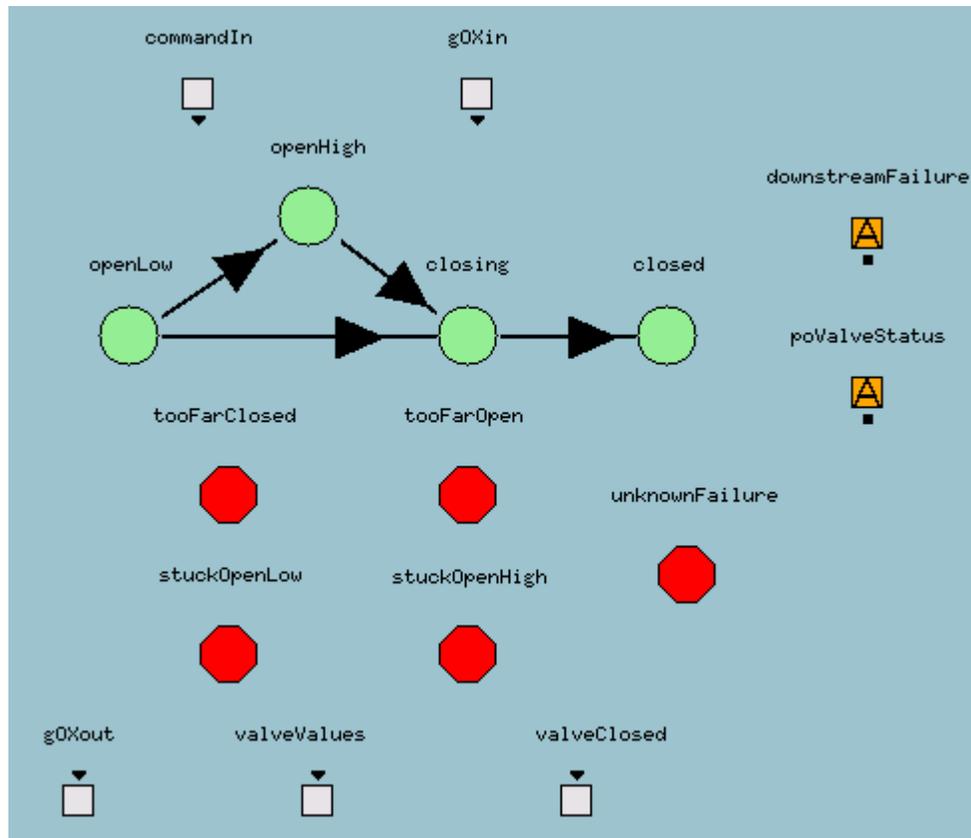
Figure 30 shows the facility pressures and control valve command and feedback for a nominal firing.



**Figure 30: Pressures and control valve characteristics for a nominal 4 kg/sec firing.**

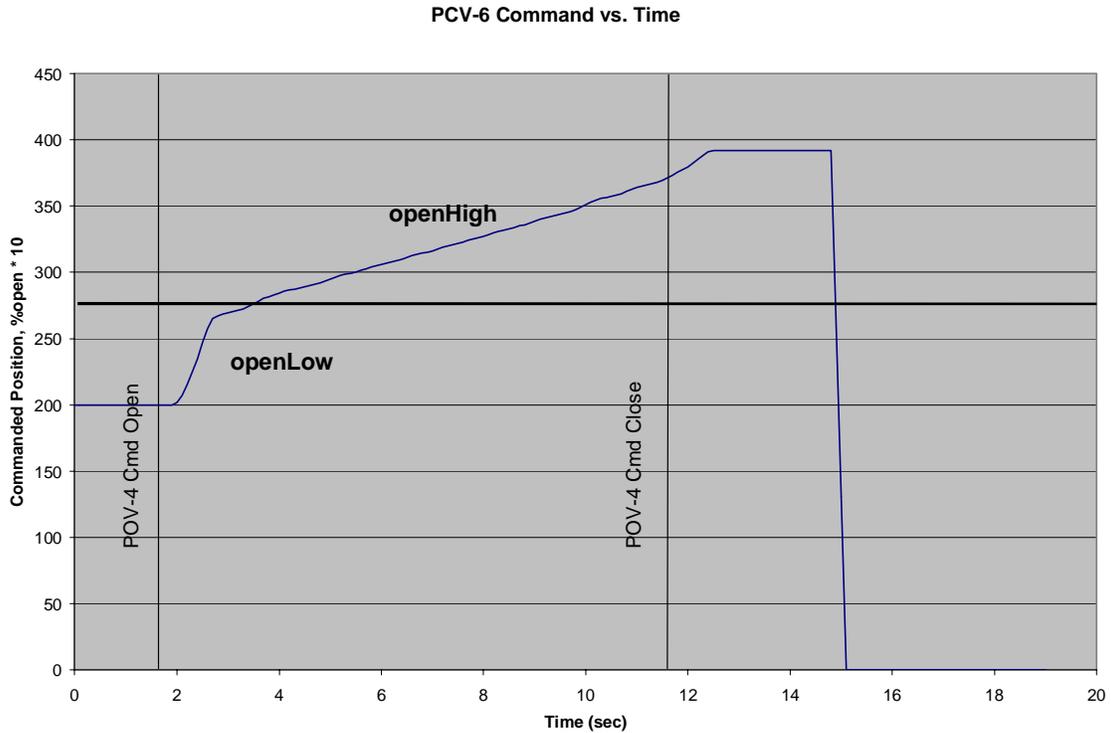
After POV-4 opens, the pressure in the tank decreases and the control valve opens to maintain constant pressure at PT-6. While the pressure at PT-6 is constant the control valve opens at a nearly constant rate. We can use these kinds of observations to model the control valve component based on the firing timeline, rather than valve position alone. Figure 31 shows the PCV-6 component. The initial mode is *open low* and is relevant for the initial valve setting of approximately 20% open and just after POV-4 opens. A transition is made to the *open high* mode when the valve command is increasing at a constant rate to control the downstream pressure. Figure 32 suggests when this transition might occur; it also factors in where PT-6 typically goes steady (see Figure 30). After POV-4 is commanded to close, we transition to the *closing* mode. Finally, the transition to the *closed* mode occurs when the closed limit switch indicates

that the valve has completed its closing action. The details of the modes are discussed in the next few paragraphs.



**Figure 31: PCV-6 component.**

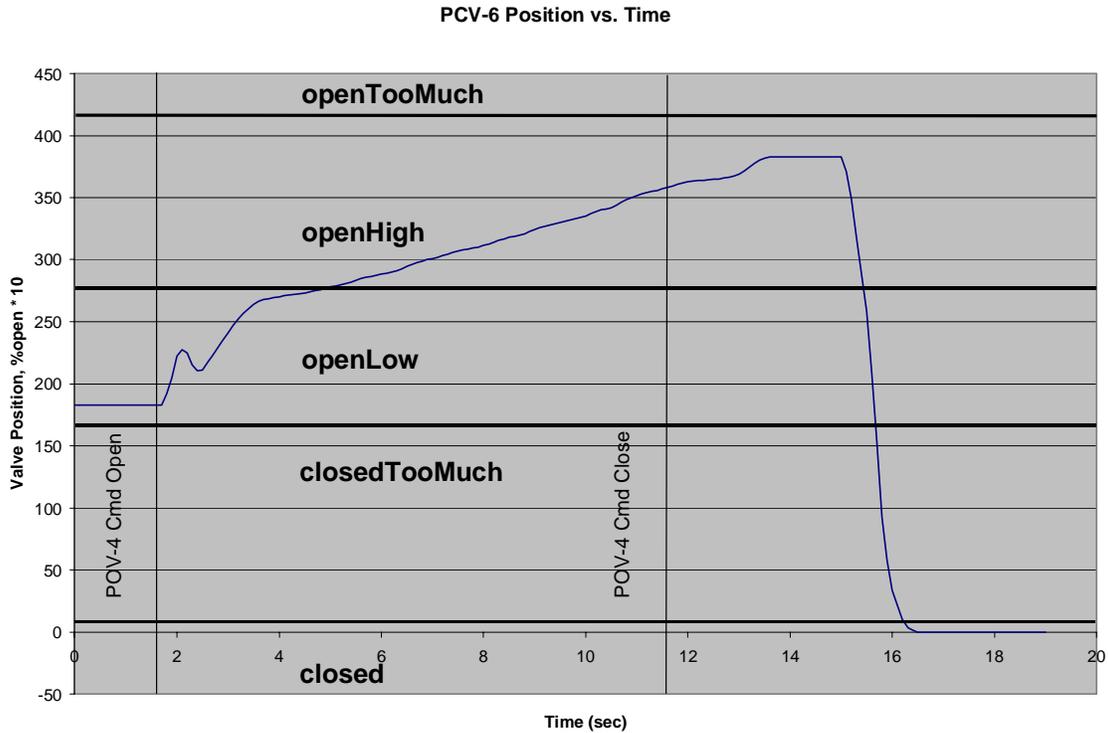
In the initial *open low* mode, we expect that the valve position feedback will be in the open low bin as shown in Figure 33. Note that the valve gets jostled a little bit after POV-4 opens and the position trace exhibits a bump before rising. Because of this, no assertions are made about the valve action (binned according to the derivative of the position trace) since it goes from opening to not moving to closing. Just after POV-4 is opened, the downstream pressure should not be dropping but one cannot assert a specific pressure level since it progresses from ambient to nominal (see Figure 34). If POV-4 is closed, the input and output (*gOXin* and *gOXout*) of PCV-6 are equated.



**Figure 32: Example of typical smoothed control valve command trace with L2 bins.**

Once the continuous valve command crosses the threshold for the open high (command) bin, a discrete command is given to the L2 component to transition the mode to *open high*. In the *open high* mode, we expect that the outgoing pressure level is nominal, the pressure rate is steady, and the control valve is opening at a nominal rate. Since the valve position feedback lags the command slightly<sup>3</sup>, we allow for the valve position feedback to be either in the open low or open high bins. If the valve position is in the open low bin the outgoing pressure may be increasing and the valve may be opening at a faster rate. If the incoming pressure is low due to a failure upstream, we anticipate that the valve will open faster than normal to maintain the setpoint pressure downstream of the valve.

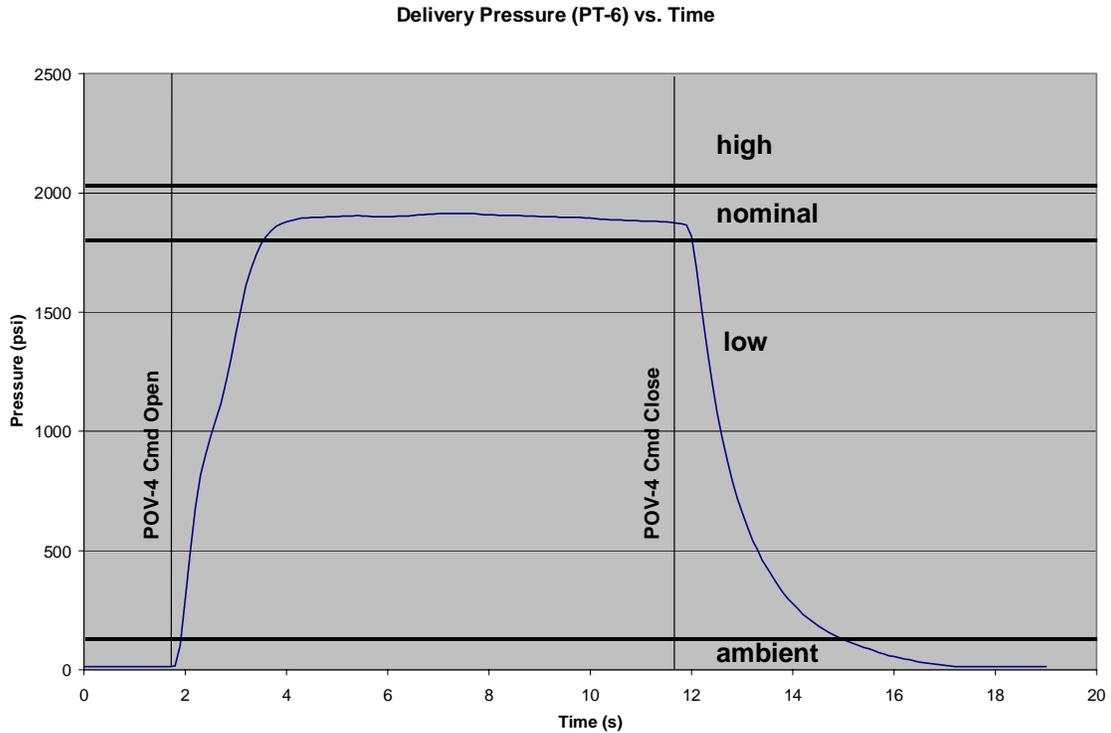
<sup>3</sup> The threshold between the open low and open high bins for the valve position feedback is also set slightly lower than the threshold for the valve command to account for the lag.



**Figure 33: Example of typical smoothed control valve position trace with bins.**

After POV-4 is commanded to close, the mode of PCV-6 transitions to *closing*. The constraints of the closed PO valve (pressure is ambient or dropping) are propagated through the control valve by equating the input and output *feedline* variables. As shown in Figure 33, the control valve continues to open after the POV-4 close command so no constraints are given for the control valve action while the valve is still in the open high bin.

The transition from *closing* mode to *closed* mode happens when the close limit switch feedback for the control valve reports closed. Similar to the limit switches in POV-5, the policy would be written so that the command is issued either when the affirmative closed limit switch feedback is received or after a specified amount of time has elapsed. In the *closed* mode, the valve position feedback should confirm that the valve is in the closed bin and not moving. Furthermore, by the time the valve closes, the pressure level out should be ambient or low and the pressure rate should be steady or dropping slow as the remaining oxygen bleeds from the system.



**Figure 34: Example of delivery pressure trace with L2 bins.**

There are five failure modes including the PCV-6 model. Two of them are specific to a valve that has stuck at a fixed position; two are more generic, and the catchall fault mode, *unknown fault*, has no constraints.

An attempt has been made to allow for unexpected behavior of the control valve if a fault occurs elsewhere in the system. For example, if the effective diameter of the sonic nozzle is reduced because of excessive wax buildup, the backpressure will be higher and the control valve will open at a lower rate while maintaining the desired pressure setpoint. In this case, we do not necessarily want to implicate a faulty control valve because it was not opening as expected. A simple-minded approach was taken to relax the constraints in the control valve if a downstream component had failed. The transition from the *open low* mode to the *closing* mode (see Figure 31) is required in this kind of failure scenario as well. It is not clear that this approach would be robust and more fault data is needed to refine the model.

### 7.2.7 Venturi (FE-101)

Inputs	<i>Feedline</i>	
Outputs	Differential pressure (ambient, low, nominal, high)	
	<i>Feedline</i>	
Attributes	Monitor	Whether we should look at DPT-101
	Downstream failure	Whether there is a downstream failure
Nominal Mode	Nominal	<i>Feedline</i> in equals <i>feedline</i> out If monitor is on: If pressure in is nominal and there are no downstream failures, the differential pressure is nominal
Failure Modes	Faulty	No restrictions

**Table 17: Venturi component.**

The venturi component has a *feedline* input and output and a differential pressure output. Two attributes are used in the nominal mode logic. The values of the two attributes are assigned externally to the component. The differential pressure measured by DPT-101 is used to calculate the mass flow through the GOX line. As shown in Figure 30, the DPT-101 pressure spikes after POV-4 is opened. The reading is inaccurate during transients and should only be monitored during steady-state conditions. The *monitor* attribute is turned on when the configuration of the other components in the model is such that steady-state conditions are expected. The differential pressure should be in the nominal bin when the delivery pressure is also in the nominal bin.

### 7.2.8 Check Valve (CKV-7)

Inputs	<i>Feedline</i>	
Outputs	<i>Feedline</i>	
Nominal Mode	Nominal	<i>Feedline</i> in equals <i>feedline</i> out
Failure Modes	Stuck closed	Pressure level out is ambient, pressure rate out is steady
	Unknown fault	No restrictions

**Table 18: Check valve component.**

The purpose of the check valve is to prevent backflow through the system. The valve has a swing disk that swivels open to allow flow in the forward direction. The rest of the model assumes that there is no reverse flow in the system; that implies that the check valve will not fail to close so it is not included as a fault mode. A *stuck closed* failure mode is included although it is very unlikely. In normal operation, the check valve simply passes the GOX through the valve with a slight pressure drop.

## 7.2.9 Sonic Nozzle

Inputs	<i>Feedline</i>	
Outputs	<i>Feedline</i>	
Nominal Mode	Nominal	Pressure level in equals pressure level out Pressure rate in equals pressure rate out Temperature in equals temperature out If pressure in is below burst disk threshold, pressure out is below the threshold as well
Failure Modes	Faulty	No restrictions

**Table 19: Sonic nozzle component.**

The sonic nozzle is used to measure the GOX flow rate and to prevent any pressure fluctuations in the combustion chamber from propagating upstream to the GOX line. There is a considerable, but predictable pressure drop across the sonic nozzle so when the logic states that the pressure level in equals the pressure level out, it does not mean that the absolute pressures are the same. The nominal bin level downstream of the sonic nozzle corresponds to pressures that are roughly half as much as the pressures in the nominal bin level upstream of the sonic nozzle. Because of the large pressure drop, the burst disk threshold variable, which is based on an absolute pressure, could be above threshold upstream and below threshold downstream.

## 7.2.10 Burst Disks (BD-100 and BD-101)

Inputs	<i>Feedline</i>	
Outputs	<i>Feedline</i>	
	Burst signal (burst, not burst)	
Nominal Mode	Nominal	Burst signal is “not burst” Pressure in and out are below the burst disk threshold <i>Feedline in equals feedline out</i>
Failure Modes	Burst	Burst signal is “burst”
	Burst disk failure	Burst signal is “not burst” and pressure in or out is above burst disk threshold
	Unknown fault	No restrictions

**Table 20: Burst disk component.**

The functions of the burst disks are to provide emergency pressure relief and to alert the control system when the pressure has exceeded a certain threshold. When the pressure exceeds the threshold, the disks will rupture to relieve the pressure. A sensor on the burst disk gives feedback as to whether or not the disk has burst. In the nominal mode we expect that the burst signal will report “not burst” and that the pressure will not exceed the burst disk threshold. The burst disks are mounted in tubes that are perpendicular to the flow so there is no effect on the GOX passing through the line if they have not ruptured. There are three failure modes. The first failure mode, *burst*, occurs when the burst disk has burst. When this happens, the burst signal gives an indication of a burst. We do not require the pressures to exceed the thresholds because there have been

occurrences when the burst disk has burst but the pressure spike was so brief that the pressure sensors did not register a value above the burst disk threshold. The second failure mode, *burst disk failure*, occurs if there is a failure in the burst disk, meaning that it does not burst when it should (i.e., when the pressure exceeds the rated burst pressure). The third failure mode, *unknown fault*, is a catchall mode that has no restrictions.

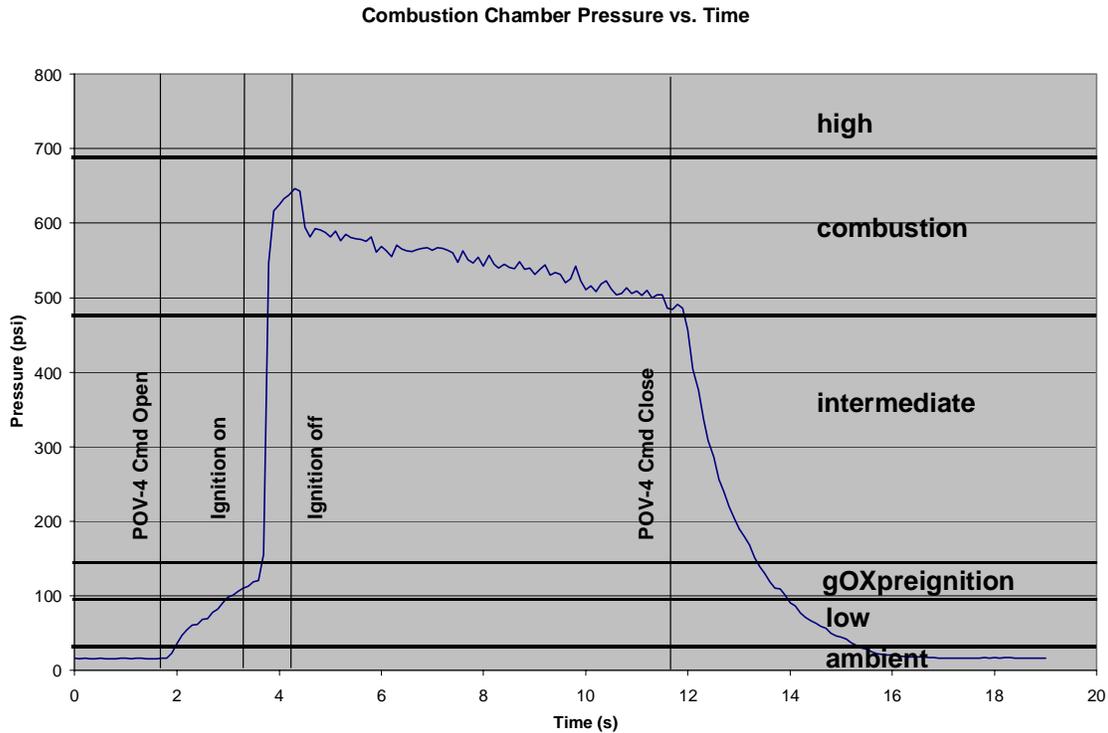
### 7.2.11 Combustion chamber

Inputs	Ignition command (on, off)	
	<i>Feedline</i>	
Outputs	Chamber pressure	Level (ambient, low, GOX pre-ignition, intermediate, combustion, high)
		Rate (rise fast, rise slow, steady, drop fast, drop slow)
		Burst disk threshold (below threshold, above threshold)
Nominal Modes	Pre-burn	<p>If pressure level in is not high, chamber pressure level is not intermediate, combustion, or high and chamber pressure level is not rising fast</p> <p>If and only if pressure rate in is not dropping, chamber pressure rate is not dropping</p> <p>If pressure level in is ambient, chamber pressure may be low or ambient</p> <p>If pressure level in is low, chamber pressure is not ambient</p> <p>If pressure level in is nominal, chamber pressure is not ambient or low</p> <p>If chamber pressure rate is rising slowly, the incoming pressure must be rising</p> <p>Burst disk is below threshold</p>
	During ignition	<p>If and only if pressure rate in is not dropping, chamber pressure rate is not dropping</p> <p>If pressure level in is ambient, chamber pressure is ambient (or low)</p> <p>If pressure level in is low, chamber pressure is not ambient</p> <p>If pressure level in is nominal, chamber pressure is not ambient or low</p> <p>Burst disk is below threshold</p>
	After ignition	<p>If and only if pressure level in is nominal, chamber pressure level is combustion</p> <p>If pressure rate in is steady, chamber pressure rate is steady</p> <p>Burst disk is below threshold</p>
	Bleed	If the pressure in is dropping or ambient, the chamber pressure is dropping or ambient (or low)
Failure Modes	Faulty	No restrictions

**Table 21: Combustion chamber component.**

The combustion chamber component has a command input and a *feedline* input. The only output is the chamber pressure. The chamber pressure characteristics include level, rate, and a burst disk threshold property. The pressure level has six bins, as depicted in Figure 35. Prior to POV-4 opening, the pressure should be in the ambient bin; after POV-4 opens and before the ignition command, the pressure will usually rise to the GOX pre-ignition bin; after the ignition command, the pressure will rise to the combustion bin.

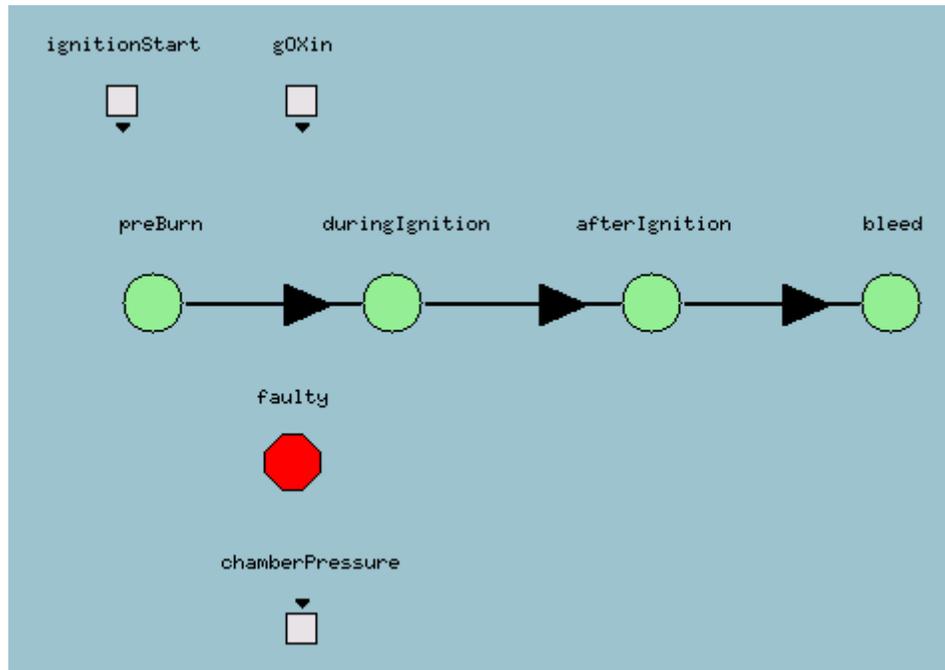
Similar to the control valve, the combustion chamber component is modeled based on the expected features as the firing progresses. The nominal modes transition as shown in Figure 36. After the ignition command, the mode changes from *pre burn* to *during ignition*. When the ignition command goes from on to off, the fuel is combusting in the *after ignition* mode. Finally, after POV-4 is commanded to close, a transition is made to the *bleed* mode. The following paragraphs describe the modes in more detail.



**Figure 35: An example of a typical combustion chamber trace with L2 bins.**

The *pre burn* mode has constraints that are relevant for the combustion chamber before the ignition of the fuel grain. The pressure level should not exceed the GOX pre-ignition level and it is clear that the pressure should not be dropping in this mode. The pressures are binned such that the combustion chamber pressure will switch bins before the incoming pressure; if the incoming pressure is nominal, the combustion chamber pressure will be at the GOX pre-ignition level but we cannot assert that the incoming pressure will be nominal if the combustion chamber pressure is in the GOX pre-ignition bin. Furthermore, the ambient pressure threshold for the incoming flow is set higher than the ambient pressure threshold for the combustion chamber pressure, so the combustion chamber pressure may be in the ambient or low bins if the incoming pressure is in the ambient bin. The ambient pressure bin of the incoming flow was increased because originally the derivative of the pressure trace during depressurization was binned to steady while the pressure level was binned to low, which violated the constraint that the pressure must be dropping or ambient. Another approach is to set the ambient threshold to be the same for all pressures and modify the logic in the components accordingly.

The *during ignition* mode is when the fuel is ignited and the pressure rapidly increases. The constraints are similar to the previous mode except we do not limit the maximum pressure. Since there is typically a small overshoot during the ignition, the threshold for the high bin could be lowered below the overshoot pressure without causing a conflict. We could add a constraint that the chamber pressure should not be high when the incoming pressure is not high if the pressures were binned as in Figure 35.



**Figure 36: Combustion chamber component.**

The mode transitions to *after ignition* when the ignition command changes from on to off. In this mode, we expect that the chamber pressure should be in the combustion bin and that the pressure should be holding steady. As seen in Figure 35, the pressure drops slightly while the fuel is burning. This is due to a moderate erosion of the exhaust nozzle and is considered acceptable. The “steady” bin is defined to include this acceptable pressure rate of change.

When POV-4 is commanded to close to terminate the firing, the mode transitions to *bleed*. Here we constrain the pressure to be dropping or at a low level.

### 7.2.12 Real Data to Livingstone

In order to perform a real-time diagnosis with Livingstone, at least two software components need to be developed along with the Livingstone model. The first is the monitor code, whose job is to translate the values read from the sensors into the discrete values needed for Livingstone. Livingstone cannot handle real-valued numbers nor can it process the data to obtain the variables used within the model. The monitors smooth data, calculate rates, compare data to thresholds, or do other statistical operations to

discretize the data. The other component that would need to be developed is the real-time interface, or RTI. This component performs three main functions [8]. First, it must translate the monitor information format (low, nominal, high) into a format that is understood by the Livingstone model (variable value enumeration of 0, 1, 2). Second, it must use the timing information associated with the events to package the information into discrete Livingstone time steps using an appropriate policy. This includes accounting for transients in the physical system and overlapping commands. Third, it must decide when to request a diagnosis.

Instead of developing a full-blown RTI, a simpler approach was taken in this study. An excel spreadsheet was used to smooth data, calculate derivatives, and bin the logged data using thresholds. A policy was implemented manually by inserting requests for diagnoses and unassigning variable values during transients. For example, after a valve is commanded to open, we wait a certain amount of time before requesting a diagnosis so that the valve completes its opening action. The limit switches will report the valve state before the pressures have stabilized and it may be necessary to unassign some pressure readings before the diagnosis (or before another command) so that the constraints in the valve open mode are not violated. The values are reassigned after a sufficient amount of time has elapsed. A diagnosis is also requested after an unexpected event has occurred. The bin changes and manually inserted policy were put into a scenario file in the proper syntax; the hand-generated scenario file is the fully abstracted data after performing the monitoring and policy functions and can be processed by L2 directly to diagnose the system.

### **7.3 RODON**

The scope of the RODON model includes the GOX subsystem only. Figure 37 shows the top-level schematic of the qualitative and quantitative model. Each box represents a component (or assembly if it has sub-components). The smaller boxes at the left and right edges of the components are ports and correspond to the input and output variables. Some ports have links that represent physical or information pathways connecting them to ports on different components. Many of the variables are local variables and are not connected to other components. Other ports are connected globally to another port in the model as indicated by an arrow pointing to the port. For example, the output port of pressure sensor PIT-3 is connected globally to an input port of the GOX tank. In contrast to the previous methods, a component's input and output port values are related by qualitative and/or quantitative formulae. Most of the equations used to model the components were extracted from a steady-state compressible flow analysis program written in FORTRAN [1]. In that program, pressure loss calculations are computed from the sonic orifice upstream to the control valve and from the GOX tank downstream to the control valve. For simplicity, a similar approach was followed for the RODON model although the interval constraint algorithms function in the same way if one specifies either the input or output values for any of the components. The steady-state model was extended to include transients. The following paragraphs discuss the details of each component.



### 7.3.1 GOX Tank

The GOX tank is modeled as a pressure vessel undergoing adiabatic discharge. The initial mass of the GOX in the tank is calculated using the measured pressure and temperature and the ideal gas law. As the vessel is discharged the GOX mass is computed from the measured pressure change using the adiabatic relation shown in Figure 38. The change in mass divided by the change in time is proportional to the mass flow rate out of the tank. The proportional constant adjusts the ideal mass flow to the actual mass flow and is set to 0.8 based on HCF data. Isentropic relations are used to determine the conditions at the outlet.

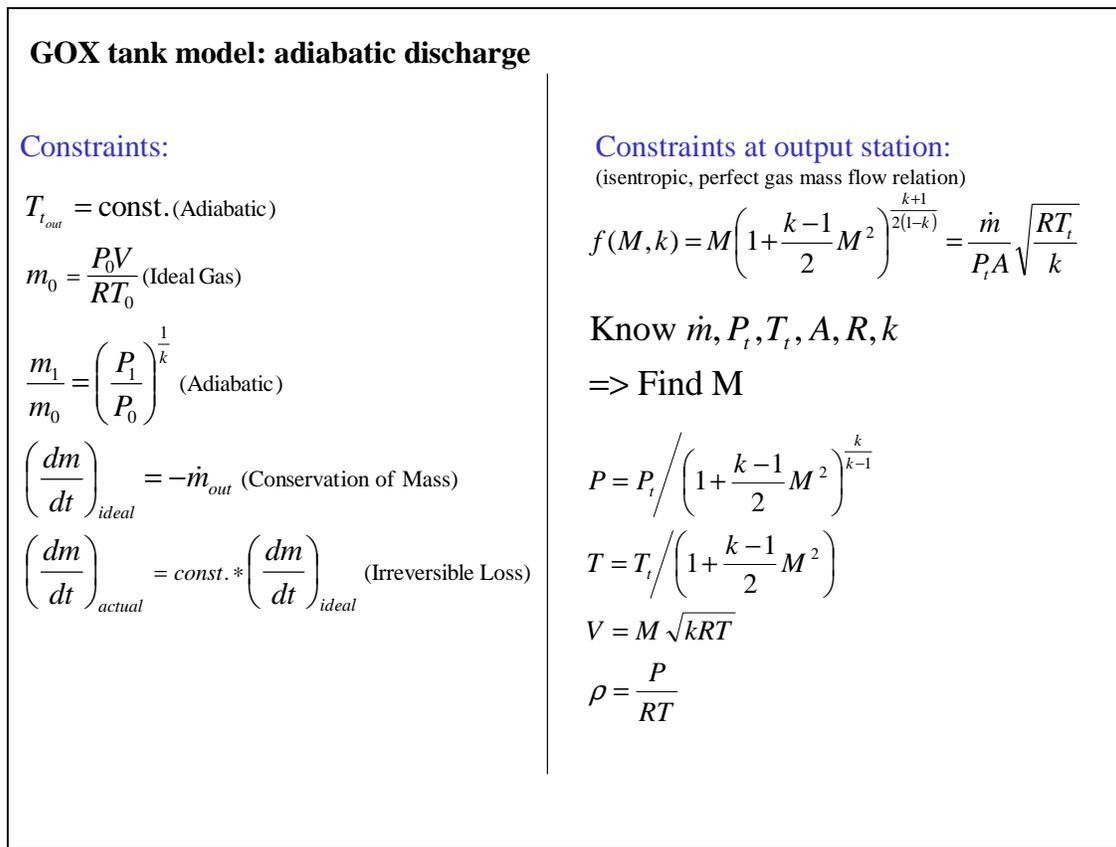
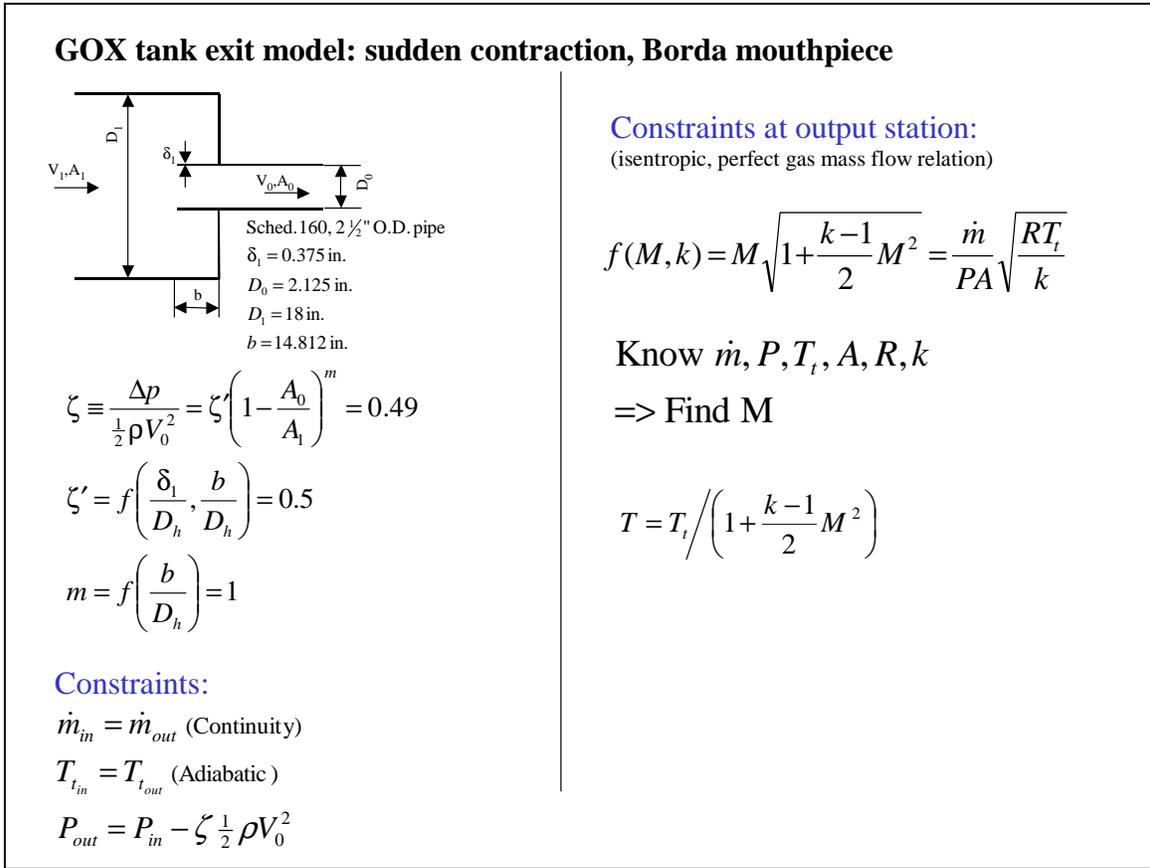


Figure 38: GOX tank equations.

### 7.3.2 GOX Tank Exit

Flow exits the GOX tank through a tube that extends into the tank. This is modeled as a sudden contraction through a Borda mouthpiece as shown in Figure 39. The equations and pressure loss coefficient values are from [9]. Isentropic relations are used to determine the conditions at the outlet.



**Figure 39: GOX tank exit equations.**

### 7.3.3 Shutoff Valves POV-4, POV-5

Two redundant ball valves are located back-to-back in the feed line to shut off or enable the flow of oxygen to the combustion chamber. Solenoid valves control the flow of compressed air to pneumatically driven actuators that turn the ball valves. During normal operation the downstream valve, POV-5, does not play an active role in controlling the flow of oxygen but functions as a backup valve should POV-4 fail.

The model of the shutoff valve consists of an assembly of four components—one component corresponds to the ball valve, another represents the function of the solenoid valve and actuators to open or close the valve, and the last two are the closed and open limit switches on the ball valve.

The equations relating pressure drop to mass flow through the valve are taken from vendor literature and are shown in Figure 40. Note that the valve flow coefficient of  $C_v = 322$  is for a fully open valve. The valve manufacturer was contacted to get the flow coefficients as the valve rotates from fully closed to fully open. A theoretical estimate of the valve flow coefficient as a function of degrees open is shown in Figure 41 and is used instead of the constant  $C_v$  in the valve mass flow equation.

The valve position (degrees open) is set by a valve driver component that connects to the ball valve component. No attempt is made to model the physics of the actuation process. Instead, the rate at which the actuator opens the ball valve is specified to be between a lower bound and an upper bound. Also specified is an interval for the time delay between the solenoid open command and the start of activation. The intervals are chosen to be slightly greater than the experimentally measured valve response and allow for windows of valve position versus time as sketched in Figure 42. These windows are computed directly in the component. An alternative approach is to use the average time delay and actuation rates to compute the valve position outside of the RODON model (in the data processing, or monitoring code). A value for the valve degrees open is then passed to the model and has an associated tolerance that allows for deviation from the average values used to compute the valve position. In the physical system, the rate of opening and the observed delay are functions of the pneumatic pressure, starting torque, moving torque, and the geometry of the actuator cylinder and piston, among other things.

As the ball valve rotates from 0 to 90 degrees, the close limit switch value changes from 1 to 0 followed by the open limit switch changing from 0 to 1. The contacts on the switches are adjusted so that the indications will occur within a couple degrees of rotation from the close and open hard stops. In the model, errors in the data alignment process, uncertainty in the logged data, and variability in the valve actuation complicate asserting the value of the open and close limit switch based purely on computed valve rotation. Instead, the statuses of the valve switches are checked only after the intended action is completed. For example, if the valve is commanded to open the indications of the open and close switches are checked after the computed valve rotation angle has reached 90 degrees rather than checking the close limit switch after only a few degrees of rotation. This was necessary to eliminate spurious diagnoses when processing nominal data from HCF firings.

### Shutoff valve model: Instrumentation, Systems, and Automation (ISA) standards

$$x = \frac{P_1 - P_2}{P_1} \quad \text{limit } x \leq x_T = 0.25$$

$$F_k = \frac{k}{1.4}, F_P = 1$$

$$Y = 1 - \frac{x}{3F_k x_T}$$

$$C_V = 322$$

$$Q = \frac{1360}{60} C_V F_P P_1 Y \sqrt{\frac{x}{G_g T Z}} \quad [\text{SCFM}]$$

$$\dot{m} = \frac{1.33}{60 \times 3.28^3} Q \quad [\text{kg/sec}]$$

Know  $\dot{m}, T, Z = f(M), P_1$

=> Find  $P_2$

Constraints:

$$\dot{m}_{in} = \dot{m}_{out} \quad (\text{Continuity})$$

$$T_{t_{in}} = T_{t_{out}} \quad (\text{Adiabatic})$$

If valve is closed,  $\dot{m} = 0$

Constraints at output station:

(isentropic, perfect gas mass flow relation)

$$f(M, k) = M \sqrt{1 + \frac{k-1}{2} M^2} = \frac{\dot{m}}{PA} \sqrt{\frac{RT_t}{k}}$$

Know  $\dot{m}, P, T_t, A, R, k$

=> Find M

$$T = T_t / \left( 1 + \frac{k-1}{2} M^2 \right)$$

Figure 40: Shutoff valve (POV-4, POV-5) equations.

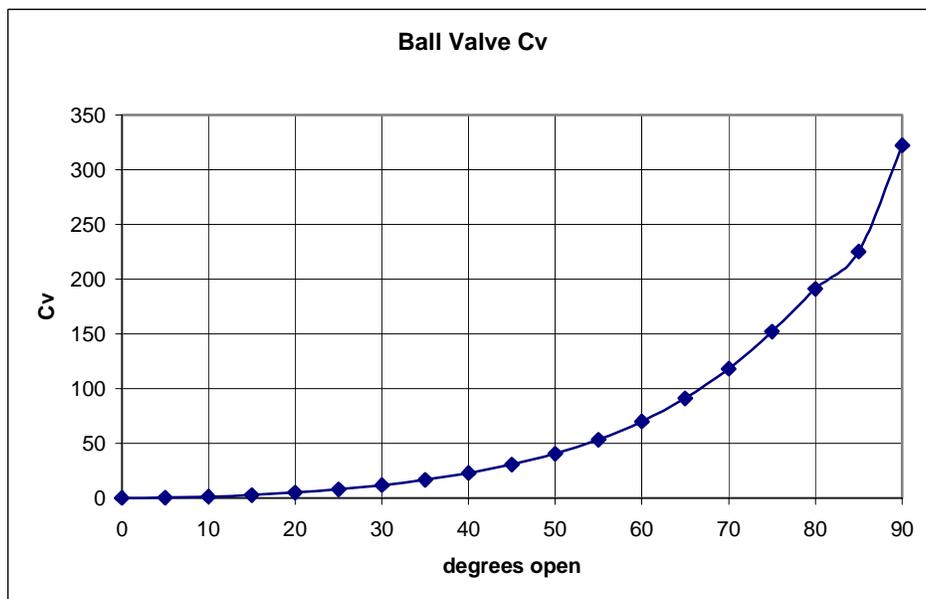
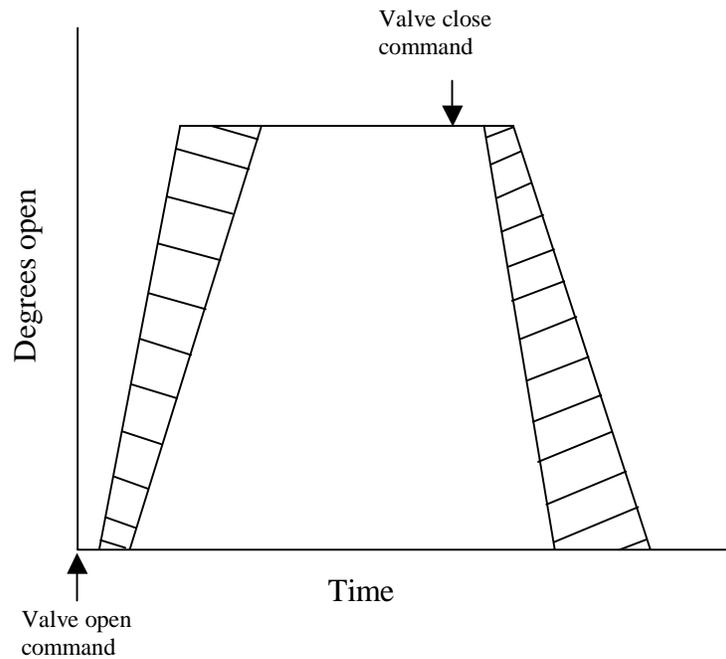


Figure 41: Shutoff valve CV characteristics.



**Figure 42: Valve open and close characteristics.**

### 7.3.4 Pipe-1

This component represents the volume of the pipe segment between the shutoff valve and the control valve and enforces conservation of mass. The change of mass in the pipe is equated to difference of mass flowing into the pipe and mass flowing out of the pipe as shown below. For simplicity, a constant temperature was assumed. During steady-state conditions the change in mass in the pipe will be zero but during transient conditions the pipe serves as a capacitive element that limits the rate of pressure increase and allows for different mass flows in the connecting components.

$$M_1 = \frac{P_1 V_{pipe}}{RT}$$

$$M_0 = \frac{P_0 V_{pipe}}{RT}$$

$$\Delta M = M_1 - M_0 = (\dot{m}_{in} - \dot{m}_{out}) \Delta t$$

### 7.3.5 Control Valve PCV-6

The control valve opens during the firing to maintain a constant GOX delivery pressure to the combustion chamber. The equations relating pressure drop across the valve to the

mass flow rate are the same as the shutoff valve (see Figure 40) but with  $x_T = 0.7$  and with  $C_V$  given by the following table.

% open	0	10	20	30	40	50	60	70	80	90	100
$C_V$	0	1.51	4.87	11	20.3	30.9	41.5	50.2	57	61.4	64.8

**Table 22: Control valve  $C_V$  characteristics.**

At the HCF, the valve's position commands are computed by a Proportional, Integral, Derivative (PID) loop in a Programmable Logic Controller (PLC). The electronic signal output of the PLC is converted by a valve positioner to a pneumatic pressure that drives the actuator to the desired position. While the details of the positioner, actuator, and the PLC are not modeled, two aspects of control valve operation are monitored.

First, the PID command output of the controller is compared to a calculated value using the logged set point pressure and delivery pressure according to the following equation:

$$pid = bias_i + K_P \epsilon_i + K_I \sum \epsilon_i + K_D (\epsilon_i - \epsilon_{i-1})$$

where,

$pid$  is the output of the pid loop (in this example, the control valve commanded position)

$bias_i$  is the control valve position offset

$K_P, K_I, K_D$  are the coefficients for the proportional, integral, and derivative terms, respectively

$\epsilon_i$  is the difference (error) between the current set point and process variable (delivery pressure)

$\sum \epsilon_i$  is the sum of all previous errors

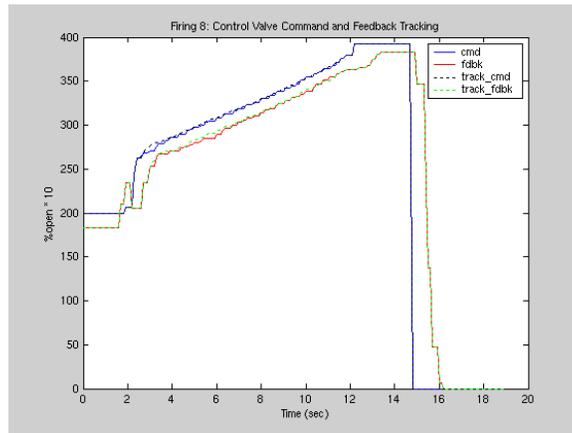
$\epsilon_i - \epsilon_{i-1}$  is a discrete approximation of the rate of change of error

The PID loop becomes active only after a specified amount of time has elapsed from the POV-4 open command. Prior to this, the error is set to zero and the PID output equals the bias. After the control loop becomes active, the bias and the set point values may change at predefined times. The HCF operator sets the values and timing information prior to the firing. These same parameters are used to compute the expected PID output.

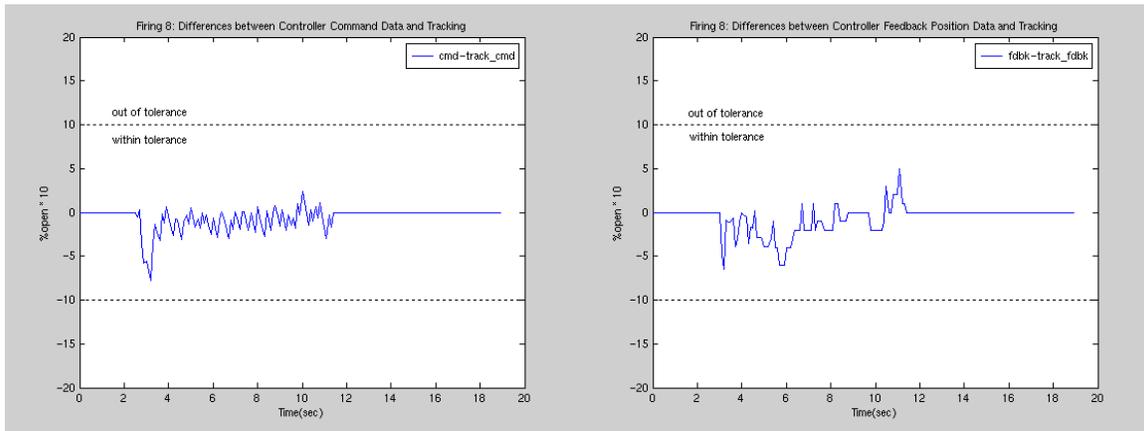
Second, the control valve feedback position is compared to an expected value that is calculated by applying a pseudo transfer function to the control valve command signal. This function attempts to account for the observed offset between the command and feedback position.

The PID command tracking and position feedback tracking are shown in Figure 43a for a nominal run. The differences between the PID command and computed PID output are shown in Figure 43b and the differences between the valve feedback position and

computed position are shown in Figure 43c. The comparison between the actual and computed values is only for a window of time during the run. For the control valve command, this window starts a short time after the PID loop becomes active and ends when valve POV-4 is commanded to close. For the control valve position, the window starts a short time after the valve begins to respond to the valve command and ends when valve POV-4 is commanded to close. Outside of these windows, the expected values of the valve command and position are set equal to the actual values and the differences are zero. We can characterize nominal behavior by setting maximum allowable differences between the actual and computed control valve command and position as shown in Figure 43b and c. If the differences exceed the tolerances, faulty behavior is implied.



a) Control valve position command and feedback and tracking.



b) Differences between actual and expected control valve command.

c) Differences between actual and expected control valve position.

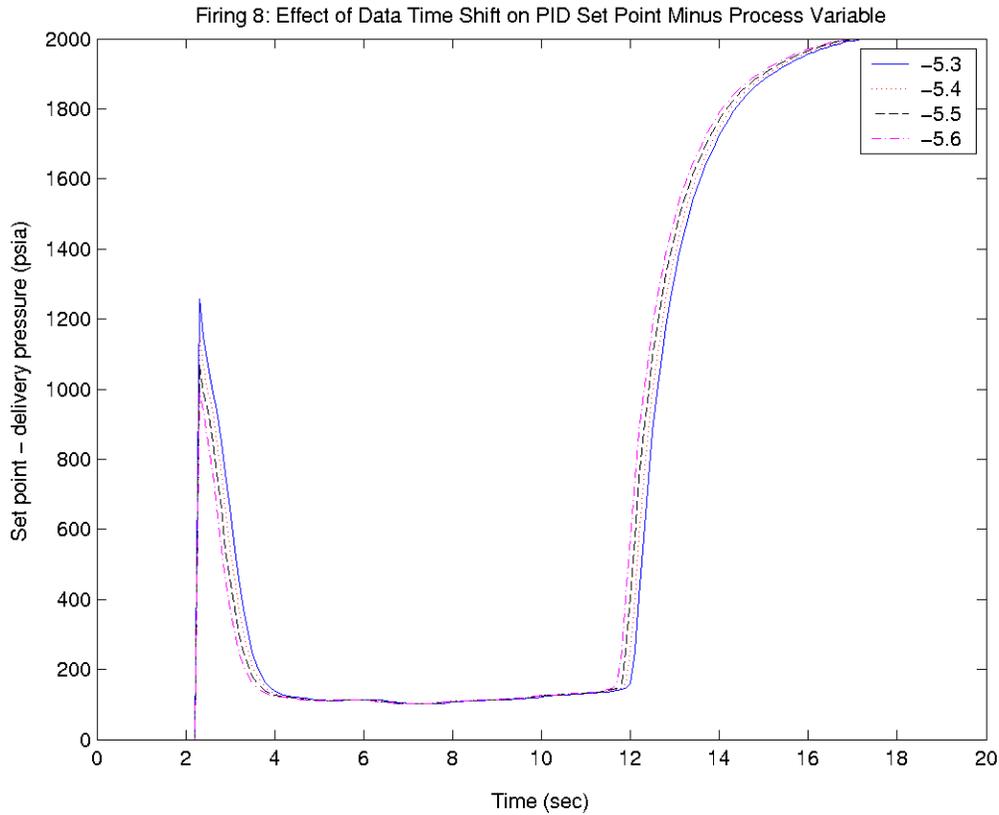
**Figure 43: Control valve command and position tracking.**

For simplicity, the code to track the control valve command and position is not implemented in the PCV-6 component itself but rather in the monitoring (processing)

code, which typically passes the sensor values to the model (more will be said about the monitoring function later). Here, the monitoring code is used to track whether the control valve command and position are within the tolerances defined for nominal operation and to send flags to the control valve component that indicate in or out of tolerance. Thus, it is analogous to a TEAMS pass/fail test. The control valve position tracking flag is passed through the position sensor and allows for the possibility of a faulty sensor causing the out of tolerance indication. The PCV-6 component contains simple statements expressing the fact that the flags should indicate in tolerance for nominal operation. An out of tolerance flag will cause the PCV-6 component to be diagnosed with a fault. This could be extended to implicate failures in other subsystems, such as the pneumatics.

Several issues are worth mentioning. The calculated PID output is sensitive to the data alignment procedure described in section 5.3 since it depends on the difference between the set point and delivery pressure at each time step after the PID control becomes active. Figure 44 shows the effect of different data alignment time shifts on the value of the PID error parameter,  $\varepsilon$  (set point – delivery pressure). Note that the area under the curve represents the integral term. While the effect of a 0.1 sec difference in time shift appears small, it is enough to significantly alter the output of the PID. The accuracy and frequency of the controller logged data also affect the PID output since the PID activation occurs a certain time after the value of POV-4 changes from 0 to 1. If the recording of that event is slightly off (see the discussion in section 5.2) the PID output will start sooner or later than desired and has an effect that is similar to the time shift mentioned previously. Furthermore, it is difficult to duplicate the results of the PLC output since the PLC calculates the PID at 50 Hz and the monitor code calculates at 10 Hz (the frequency of the recorded data). There is also some uncertainty as to the implementation details of the PID loop in the PLC. All of these factors lead to the result that the PID tracking is not very robust with the current set of data. Typically, the PID constants used in the monitoring code need to be adjusted for each firing under consideration.

The applicability of the pseudo transfer function used for tracking the control valve position feedback is limited to firings where the valve command is monotonically increasing. The function could be extended to account for valve hysteresis in those runs where the valve actuation changes directions. Note that the transfer function was tuned so that the computed value closely agreed to the actual value for nominal firings. Although this deviates from a “first-principles” approach, it was necessary to avoid modeling the complexity of the valve actuation while still monitoring whether the control valve is behaving nominally.

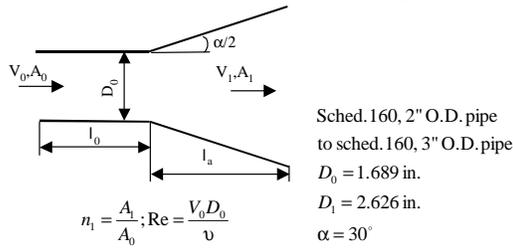


**Figure 44: Effect of data time shift on PID error parameter.**

### 7.3.6 Diffuser

The diffuser connects the 2-inch section of piping to the 3-inch section of piping. The pressure loss equations shown in Figure 45 are taken from [9]. Notice that the input values are solved from the downstream values. This is the approach taken in [1] and is followed here for all components between the control valve and the sonic orifice.

### Diffuser model: free discharge from a circular straight wall diffuser



$$\zeta \equiv \frac{\Delta p}{\frac{1}{2} \rho V_0^2} = \zeta_{tot} = f(\alpha, n_1, \text{Re}) = 0.635$$

assumption: uniform velocity at entrance

#### Constraints:

$$\dot{m}_{in} = \dot{m}_{out} \text{ (Continuity)}$$

$$T_{t_{in}} = T_{t_{out}} \text{ (Adiabatic)}$$

$$\rho = \frac{P_{t_{out}}}{RT_{t_{out}}} \text{ (Incompressible)}$$

$$V_0 = V_{out} \frac{A_{out}}{A_{in}} \text{ (Incompressible)}$$

$$P_{in} = P_{out} + \zeta \frac{1}{2} \rho V_0^2$$

#### Constraints at input station:

(isentropic, perfect gas mass flow relation)

$$f(M, k) = M \sqrt{1 + \frac{k-1}{2} M^2} = \frac{\dot{m}}{PA} \sqrt{\frac{RT_t}{k}}$$

Know  $\dot{m}, P, T_t, A, R, k$

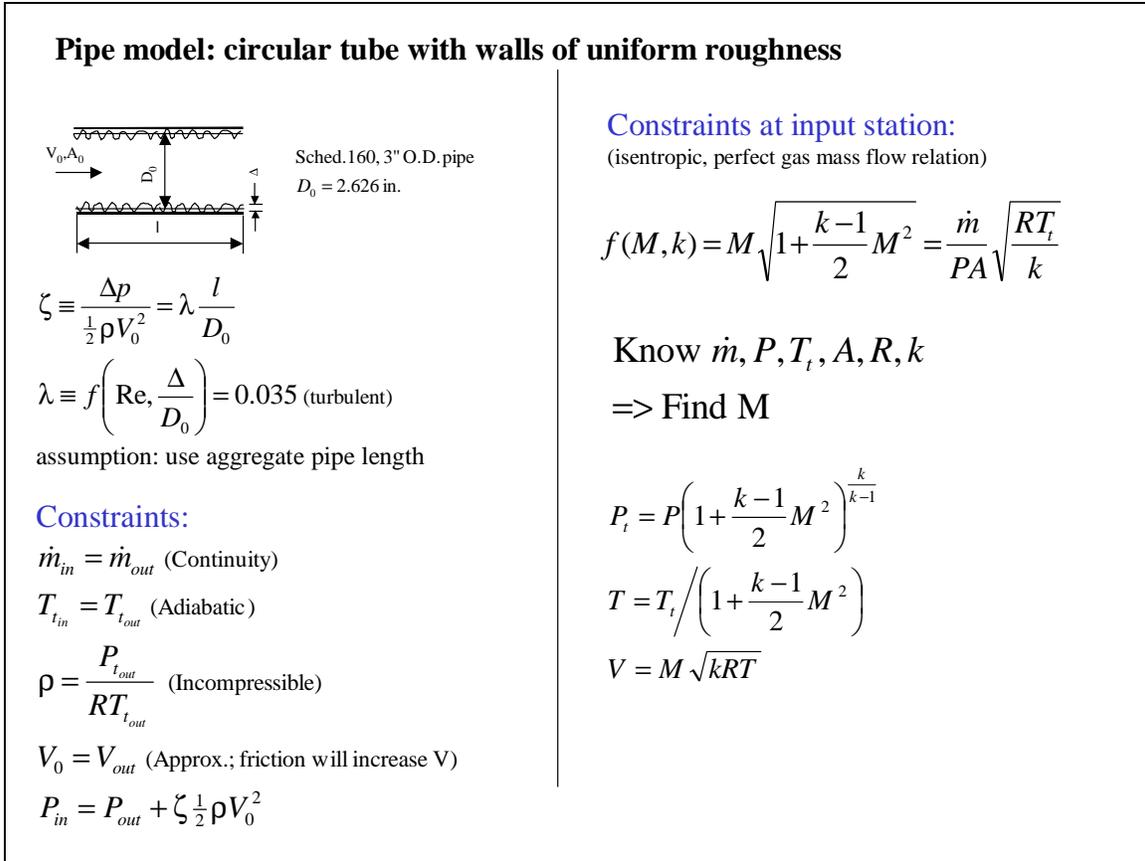
=> Find M

$$T = T_t / \left( 1 + \frac{k-1}{2} M^2 \right)$$

Figure 45: Diffuser equations.

### 7.3.7 Pipe

The pipe represents the volume between the control valve and the sonic orifice. The discussion in section **Pipe-1** applies here as well. In addition, this pipe models the pressure loss due to friction using the equations shown in Figure 46. The aggregate pipe length is used as an approximation to simplify the analysis.

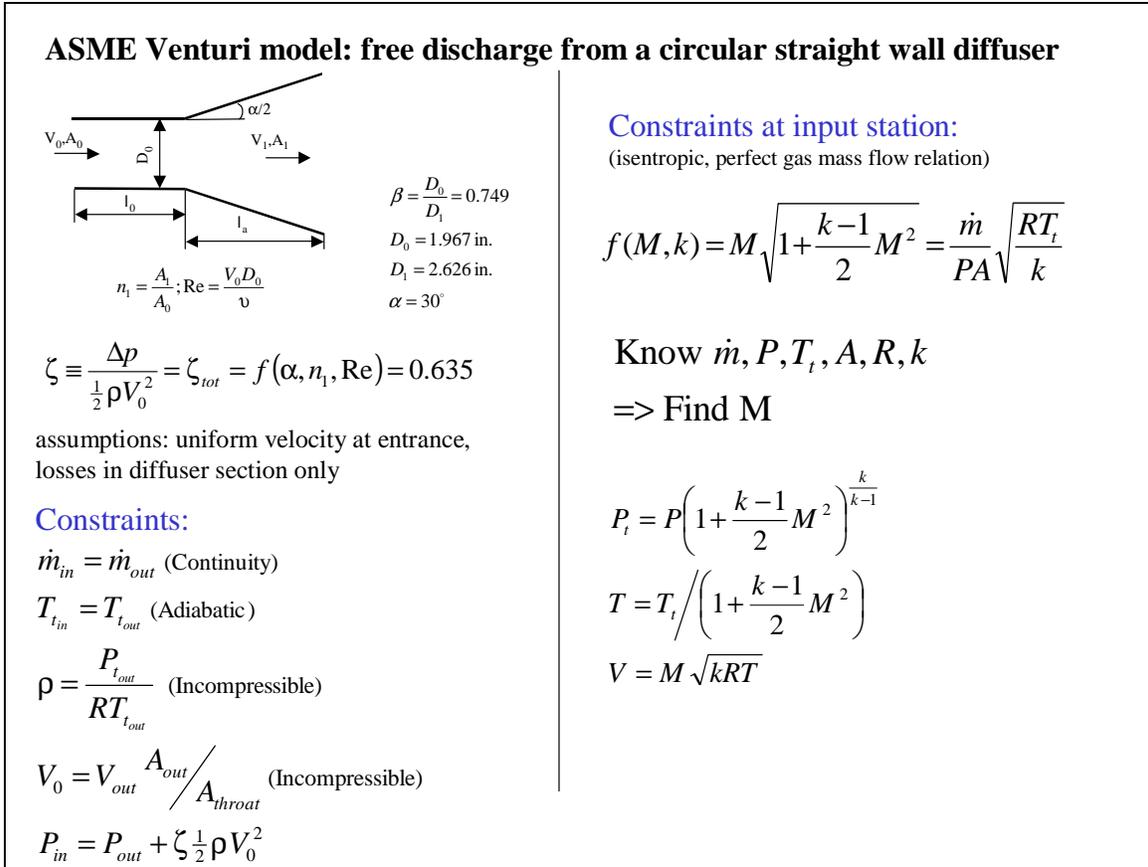


**Figure 46: Pipe pressure loss equations.**

### 7.3.8 Venturi

A differential pressure transducer on the venturi is used to compute the mass flow through the GOX feed line. The computation is inaccurate during transients and is not included in the RODON model since the sonic orifice also computes the mass flow. The addition of the venturi differential pressure transducer and the mass flow computation during steady state conditions could be used to implicate a biased pressure transducer, either at the venturi or sonic orifice. A severely biased differential pressure transducer at the venturi would not impact system operation whereas a biased transducer at the sonic orifice would alter the delivery pressure to the combustion chamber since it is used by PCV-6 to control pressure.

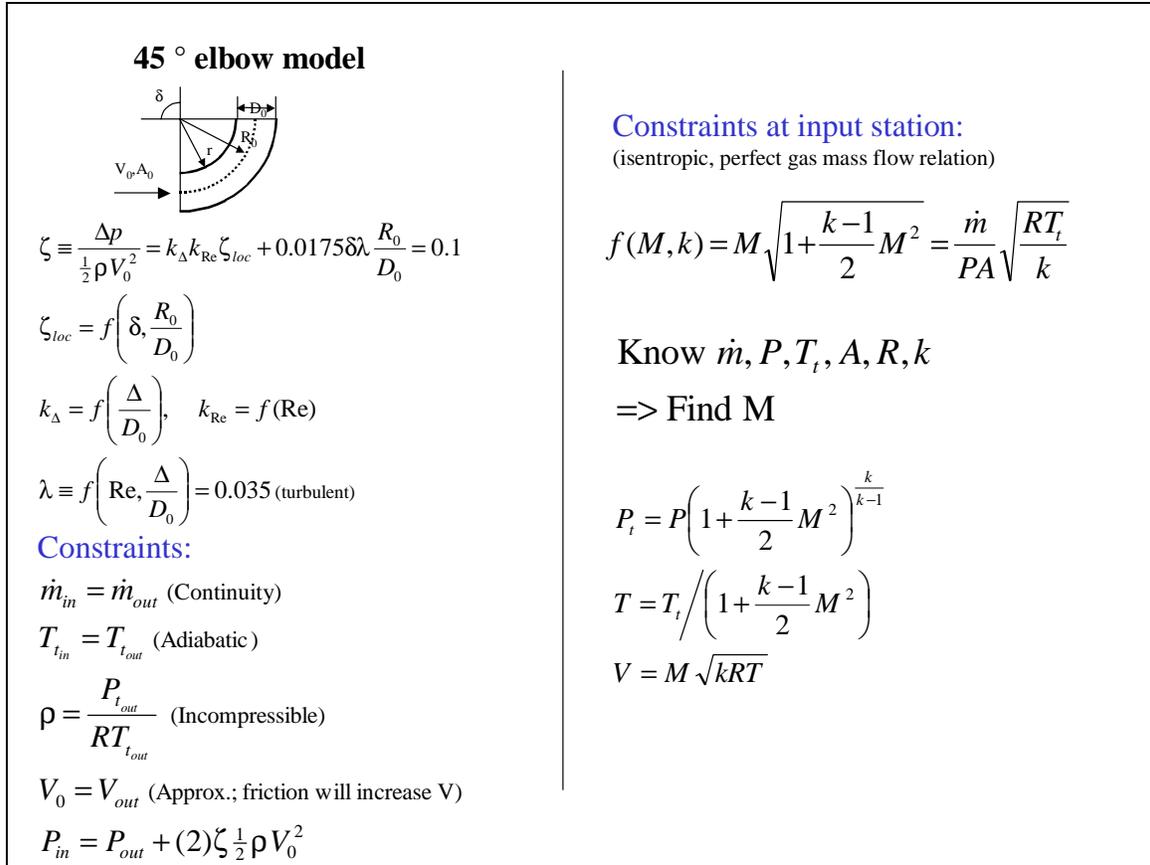
The pressure drop across the venturi is given by the equations in Figure 47, taken from [9].



**Figure 47: Venturi pressure loss equations.**

### 7.3.9 Bends

Two 45 degree bends direct the GOX flow up to the combustion chamber and cause pressure losses as computed in Figure 48 [9].



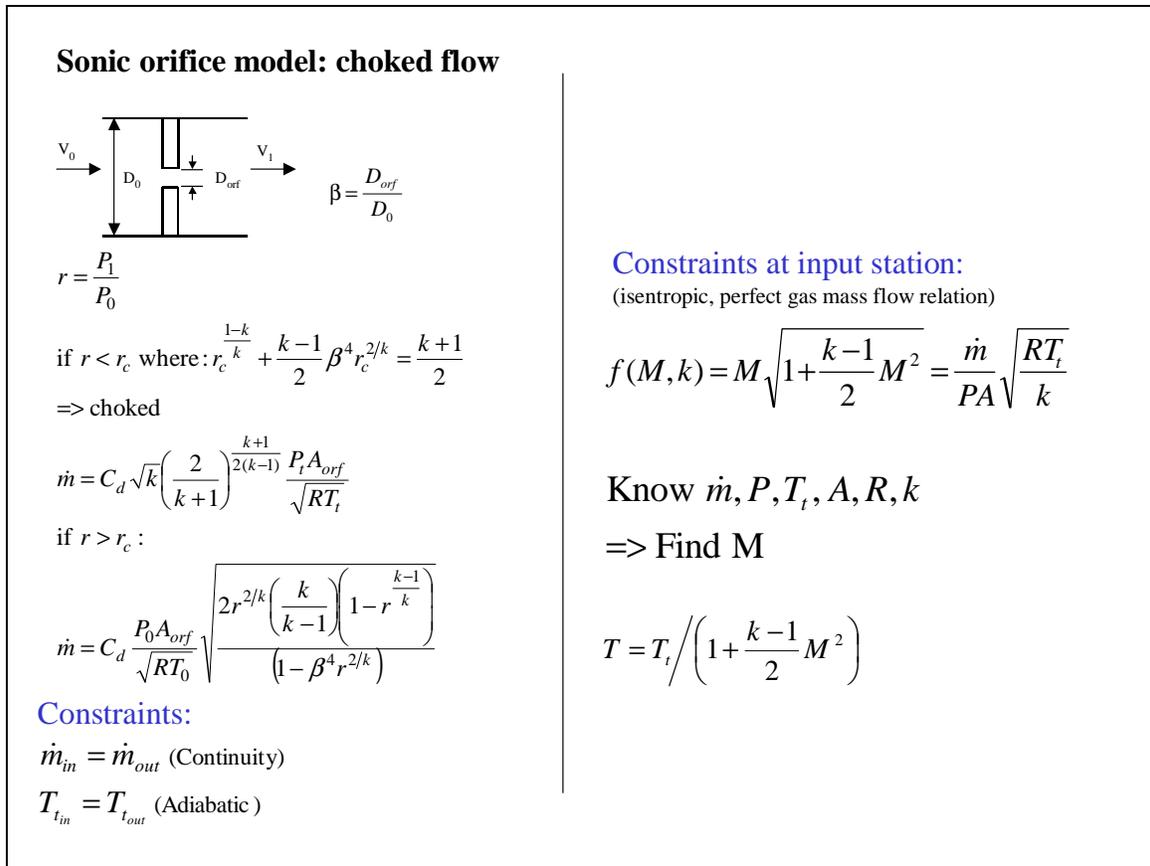
**Figure 48: Pipe bends equations.**

### 7.3.10 Check Valve

The check valve prevents reverse flow in the GOX line. The pressure loss equations are the same as the shutoff valves (see Figure 40) but with constants  $x_T = 0.3$  and  $C_V = 240$ . The constraints were written to find the input conditions instead of the output conditions. In addition, it was necessary to restrict the value of  $x$  to be less than  $x_T$  in order to get convergence of the intervals. This is equivalent to assuming that the check valve remains unchoked, a good assumption.

### 7.3.11 Sonic Orifice

The sonic orifice sets the oxygen mass flow rate and isolates the GOX feed line from pressure fluctuations in the combustion chamber. The equations for the sonic orifice are shown in Figure 49.



**Figure 49: Sonic orifice equations.**

### 7.3.12 Burst Disks

Two burst disks are located upstream of the combustion chamber to provide emergency pressure relief. Qualitative nominal and failure behaviors are defined for the burst disks. The model for the nominal behavior constrains the input and output pressures and mass flows to be equal and also constrains the pressure to be less than the pressure at which the disk bursts. A failure mode is modeled for when the burst disk has ruptured and has no constraints (the observed pressure may not exceed the burst threshold due to possibility of extremely short-duration pressure spikes). One could argue that a ruptured burst disk due to over pressurization is not a failure since the burst disk itself has functioned according to specifications but we model it as a failure to capture the possibility of the burst disk rupturing prematurely. Another failure mode is modeled for the situation where the pressure exceeds the rated burst disk pressure without rupturing the disk.

### 7.3.13 Sensors

There are three pressure sensor components and two burst disk sensor components in the model. The pressure sensors convert the input values to SI units and check to make sure the pressure is greater than zero. The outputs of the sensors are connected to other

components in the model where the physical sensors connect in the facility except for pressure gage PT-6. This gage is physically located upstream of the check valve but is modeled as being upstream of the sonic orifice in order to propagate the constraints properly.

### 7.3.14 RODON Equations

The preceding figures spell out the equations used in the components. However, some reformulation is needed in order to implement them in RODON. The version of RODON used for this study requires that the equations be linearized and restricts the arithmetic operators to +, -, /, \*. Therefore, an equation such as

$$M \left( 1 + \frac{k-1}{2} M^2 \right)^{\frac{k+1}{2(1-k)}} = \frac{\dot{m}}{P_t A} \sqrt{\frac{RT_t}{k}}$$

might be coded as the following series of statements:

```
linear 10 {240 320} sqrt_tt Tt_out (sqrt (Tt_out * R / k) )
AND
fmk = Mdot / (P*A) * sqrt_tt
AND
linearp fmk M (0 0 x2 y2 ... xn yn)
```

The first statement uses the `linear` operator to compute the square root quantity on the right hand side of the governing equation. The modeler specifies the number of equally spaced linear segments (10) and lower and upper bounds (240 320) of the independent variable, `Tt_out`. The expression `(sqrt (Tt_out * R / k) )` defines the non-linear dependence of the dependent variable (`sqrt_tt`) on the independent variable. Note that `R` and `k` are constants. The second statement completes the right hand side of the equation. The third statement uses another linearization operator, `linearp`, to define the functional relationship between the dependent variable, `fmk` (represents the left hand side of the governing equation), and the independent variable, `M`. `linearp` connects the consecutive value pairs in straight line segments, similar to `linear`. However, the modeler can specify an arbitrary (but consecutive) spacing of the abscissa values, allowing one to define denser value pairs in the areas of rapid change and sparser value pairs where the function may be more linear. In addition, a function that may require several `linear` operators (e.g.,  $a_0+a_1x+a_2x^2+a_3x^3+a_4x^4$ ) can be described by one `linearp` operator.

Increasing the number of line segments in the linearization operators increases the model accuracy but adds to the computational overhead and increases program execution time.

### 7.3.15 Simplified Model

The model described above includes some components that add to the model complexity without markedly improving the model fidelity. Passive components such as the Gox Tank Exit, Diffuser, Venturi, and Bends cause only small pressure losses and can be removed from the model without noticeably affecting interval values. The check valve also causes small pressure losses and is not active except in the case of reverse flow, which is not modeled, so this can be safely removed from the model as well. One could argue that a check valve that has failed shut would affect the system operation and should be kept in the model. In that case, a simpler check valve component could be included that ignores the small pressure drop and qualitatively equates the input and output pressures for nominal behavior. Although not passive, valve POV-5 does not function as the primary on/off valve because it is opened before POV-4 at the start of the firing and closed after POV-4 at the end of a firing.

Figure 50 shows a simplified model with the minimum number of components for comparing RODON simulations to actual data. It includes the GOX tank, a valve driver and ball valve, two pipe segments, the control valve and the orifice plate. It omits components that do not affect the pressures in the feed line but might otherwise be useful for diagnostic purposes such as limit switches and other sensors. It also assumes that the fluid Mach number is zero in all of the components. This model was used primarily to tweak model parameters until data from a nominal run could be processed without faults.

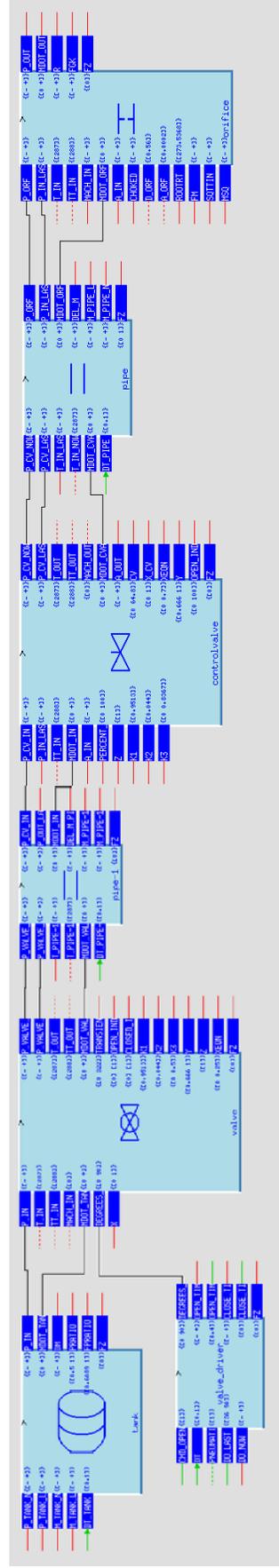


Figure 50: Simplified RODON HCF model.

### 7.3.16 Real Data to RODON

A monitoring function, implemented in C, is used to read the run data from a file and pass it to the model. Certain ports in the model require initialization values and these are read from an initialization file or derived from the run data file (e.g., the initial mass of GOX in the tank is derived from pressure and temperature of the GOX and the internal volume of tank). After the initialization data is read and after each line of run data is read, a system call is made to RODON. For the simplified model described previously, the GOX tank pressure, orifice pressure, and control valve position feedback are sent to RODON for each time stamp of data. The monitoring function also sends the ball valve commands, the control valve position feedback, and the valve tracking flags (described in section 7.3.5) to the model. [In the larger model, it sends the statuses of the valves and burst disks as well.] The values passed to RODON may have associated tolerances defined at the relevant ports in the model. These tolerances are specified as a percentage of the reported value or as a plus/minus of the reported value. So, for example, if a pressure port in the model is given a tolerance of 50,000 Pascals and the monitoring function reports the pressure as 200,000 Pascals, RODON will treat that value as being somewhere between 150,000 and 250,000 Pascals.

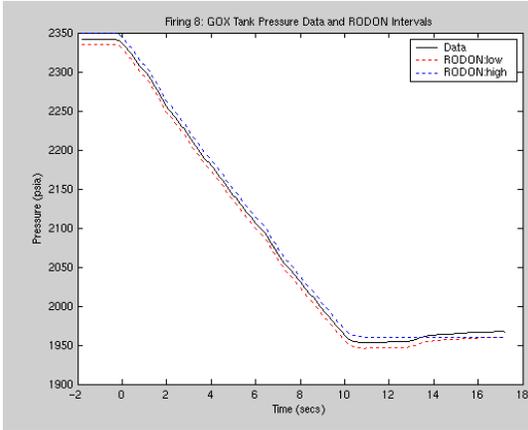
For each time step, RODON propagates the intervals associated with the sensor values through the constraint network that defines the nominal system behavior for that moment in time. If the reported sensor values do not violate any constraints, the model is consistent. If some of the constraints are violated, RODON will attempt to narrow the intervals until consistency is achieved. If no consistent solution can be found, the nominal description of the system behavior cannot explain the reported sensor readings and a fault is indicated. RODON will then search for suspect components.

This monitoring technique was used to process HCF run data. Figure 51 displays the results of monitoring firing 8, a nominal firing. The figures on the left side show the sensor values passed to RODON as solid lines and the upper and lower bounds of the sensor intervals as dashed lines. The data has been shifted so that POV-4 opens at time = 0 seconds. Tolerances of 50,000 PA (7.25 psia) were assigned to the pressure sensors and 1.2 to the control valve position feedback. These correspond to 0.25% and 1.2% of the full-scale readings, respectively. The figures on the right show the differences between the RODON upper and lower bounds and the sensor data. For time steps that admit a feasible solution (i.e., no constraints are violated) for all values in the sensor value interval, the upper bound will be greater than the sensor value by the tolerance amount and the lower bound will be less than the sensor value by the tolerance amount so that the width of the interval about the reported sensor value is two times the specified tolerance. If certain values in the interval would violate the constraints imposed by the component equations, those values are discarded and the interval is narrowed. Figure 51 a) and b) show that the interval width for the GOX tank pressure at the end of the run approaches zero. Note that the RODON upper bound for a consistent solution is actually below the reported sensor value. However, since the solution is still within the uncertainty range (tolerance) of the reported sensor value, no fault is indicated. If the

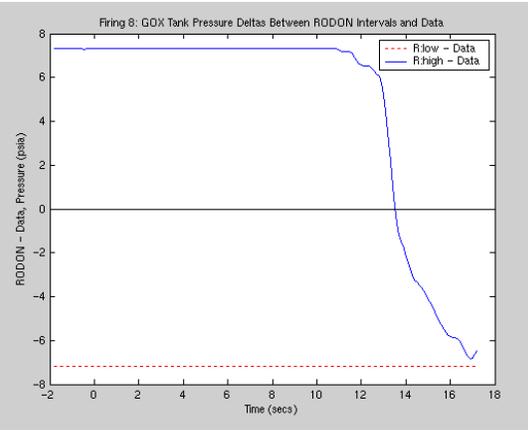
upper and lower bounds of the interval were to meet or cross, an admissible solution could not be found and the nominal system equations can no longer explain the sensor measurements.

The reason the GOX tank pressure interval narrows at the end is due to heat transfer effects that were not modeled. During the run, the expansion of the pressurized GOX cools the gas. During the run and after the shutoff valve has closed, heat is transferred from the relatively warm GOX tank shell to the cooler gas, increasing the pressure slightly. Since the model does not include heat transfer effects, it predicts that the pressure should remain the same once the shutoff valve has closed and the mass in the tank remains constant. Notice that the upper bound remains fixed while the lower bound tracks upward, staying within the specified tolerance of the sensor reading.

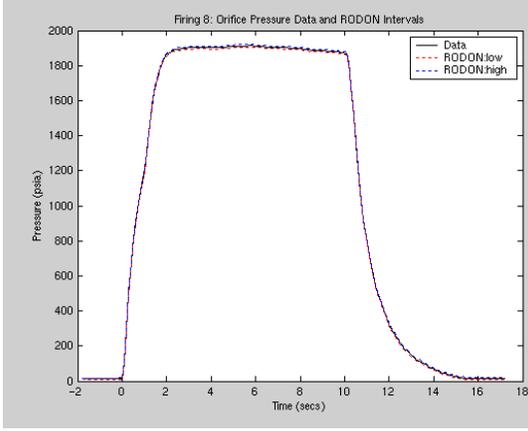
Figure 51 e) and f) show several instances of interval narrowing for the control valve position feedback in the first two seconds of the run. A close examination of the position trace will show that the data exhibits the stair-stepping quality that was mentioned in a previous section (this signal was not smoothed as in the L2 study). In effect, the fact that the intervals narrow means that the data is being smoothed by RODON. This demonstrates the power of associating an uncertainty with the sensor measurements and propagating this uncertainty through the model. In this case, RODON is treating the stair-stepping as noise in the measurement. It is likely that the tolerance for the control valve position feedback could be reduced if the stair-stepping was eliminated. Other possible contributing factors to the interval narrowing include uncertainty in the ball valve opening characteristics, the relatively large time step (0.1 sec), and the unsteady nature of the flow field as the ball valve opens.



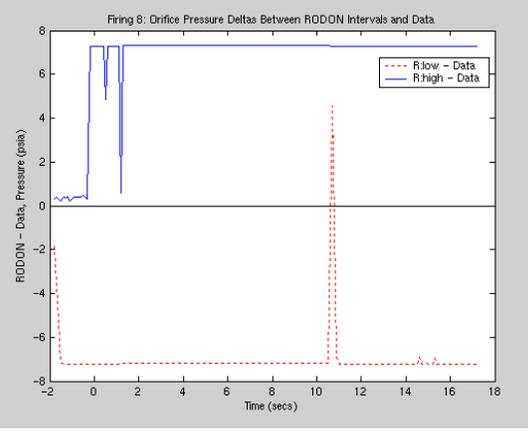
a) GOX tank experimental pressure and RODON intervals.



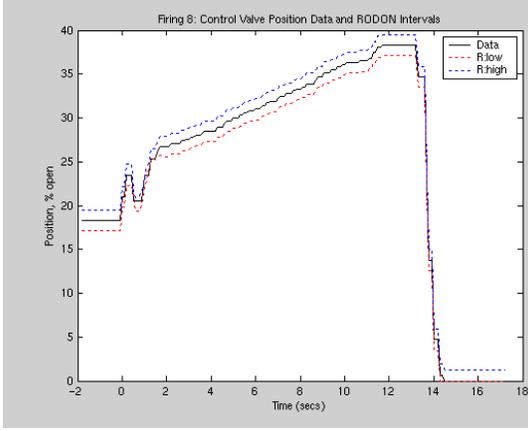
b) Differences between GOX tank pressure data and intervals.



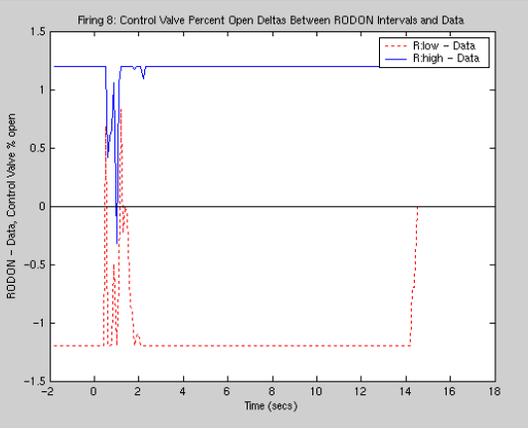
c) Experimental delivery pressure and RODON intervals.



d) Differences between delivery pressure and intervals.



e) Experimental control valve position feedback and RODON intervals.



f) Differences between control valve position and RODON intervals.

**Figure 51: RODON monitoring for firing 8.**

### 7.3.17 Simulations

RODON can be used as a simulation tool to investigate the effects of changing various parameters on system operation. The first attempts at modeling the HCF were aimed at producing a simulation that could predict the evolution of GOX tank pressure and orifice pressure during a run. Problems were quickly encountered when attempting such a simulation with RODON.

The first problem can be demonstrated on a valve, pipe, and orifice system, a subset of the simplified model described above. Figure 52a shows the predicted system pressure “branching” at  $t = 0.3$  seconds. The input pressure to the valve remains a constant value with no uncertainty. At  $t = 0$  the valve is opened and around  $t = 1$  sec the valve is closed. In order to understand why the solution admits an interval at  $t = 0.3$  sec, consider the pipe equation introduced previously:

$$M_1 - M_0 = \Delta M = (\dot{m}_{in} - \dot{m}_{out})_1 \Delta t$$

This is the conservation of mass equation. The left hand side is the change in mass of the fluid in the pipe from the previous time step to the current time step. The right hand side is the net mass flow into the pipe from the last time step to the current time step. Note that the mass flow rates are calculated at the current time step. More will be said about this later. Both the change of mass in the pipe and the mass flow rates into and out of the pipe are functions of pressure in the pipe. The mass flow rate out of the pipe (i.e., the orifice flow rate) is proportional to pipe pressure and increases linearly with increasing pressure. However, the mass flow rate into the pipe (i.e., out of the valve) decreases as the pipe pressure increases and is much more sensitive to changes in pipe pressure.

We can calculate the intervals for the left and right hand sides of the mass conservation equation and show that they agree for the given pipe pressure interval. At time  $t = 0.2$  sec the mass in the pipe is 0.725 kg. At time  $t = 0.3$  sec, the proposed pressure interval results in an interval for the mass in the pipe of [0.284 3.327] kg. This leads to an interval for left hand side of the equation of [0.284 3.327] – 0.725 = [-0.44 2.60] kg. At time  $t = 0.3$ , using the pressure interval in the pipe of [162 1900] and the orifice mass flow equations for choked flow (see Figure 49), we calculate the interval for  $\dot{m}_{out}$  as [0.38 4.41] kg/sec. Using a constant valve inlet pressure of 1900 psi, the interval for the outlet pressure [162 1900] psia, the assumed valve opening characteristics, and the mass flow equations for the valve (see Figure 40), RODON calculates the interval for  $\dot{m}_{in}$  as [0 26.4] kg/sec<sup>4</sup>.

We can now calculate the interval for  $\Delta M$ . Using a time step of 0.1 sec, the maximum

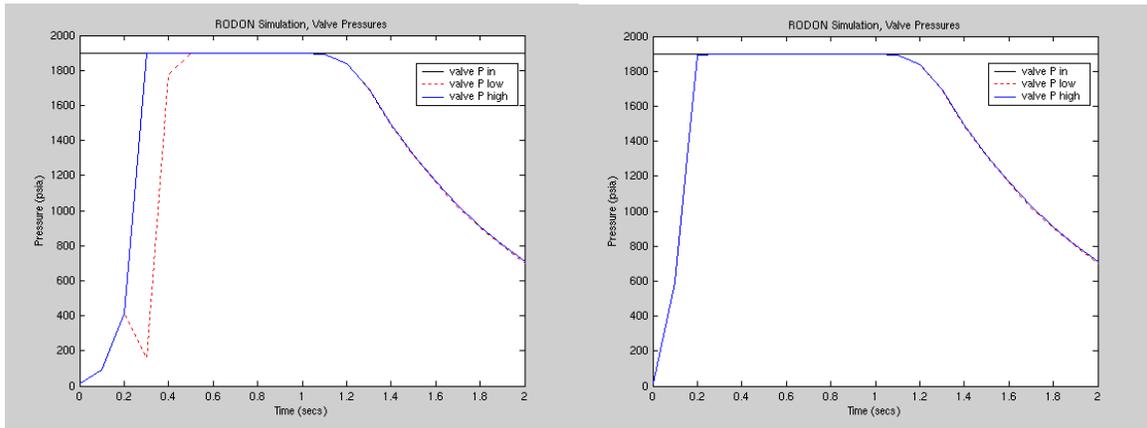
---

<sup>4</sup> A choked flow solution yields approximately 19 kg/sec. However, the equation for mass flow uses intermediate parameters that also have intervals. The interval for one parameter has the choked flow solution as the lower bound and the unchoked solution as the upper bound. When computing the maximum mass flow, the maximums of the variable intervals are used. The problem here is similar to what is illustrated in Figure 53. It is possible that a reformulation of the governing equations could address this problem.

value the right hand side of the equation can take is  $(\max(\dot{m}_{in}) - \min(\dot{m}_{out})) * \Delta t = 2.60$  kg. The minimum value is  $(\min(\dot{m}_{in}) - \max(\dot{m}_{out})) * \Delta t = -0.44$  kg. Therefore, the interval for the right hand side is  $[-0.44 \ 2.60]$  kg.

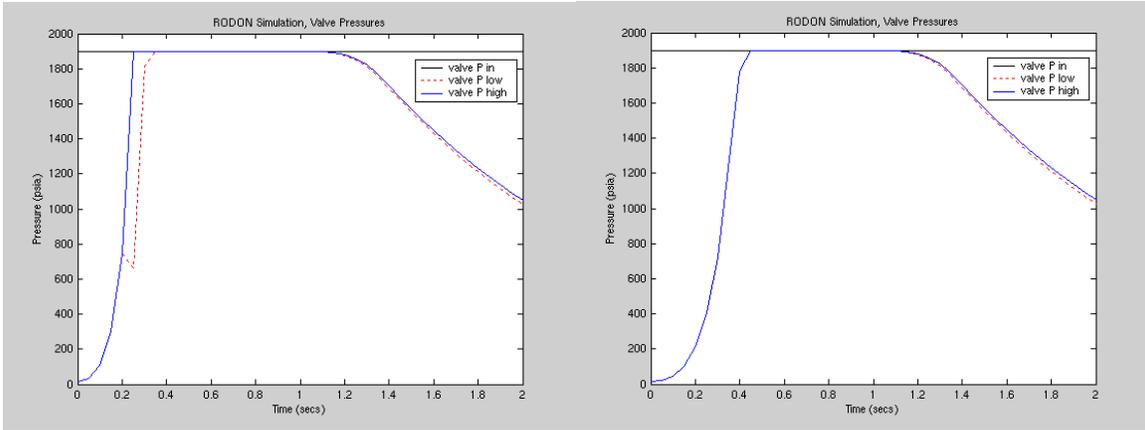
Observe that the intervals are the same but consider for a moment how we arrived at them. On the left hand side the lower pipe pressure leads to the lower bound of the interval for  $\Delta M$ . On the right hand side the higher pipe pressure leads to the lower bound of the interval for  $\Delta M$ . This is sketched in Figure 53. The mass flow equation is satisfied since the intervals are numerically equivalent and the interval for pressure does not get narrowed even though we expect a single scalar value. This branching typically occurs where the valve transitions from choked flow to unchoked flow. Various attempts to force a choked or unchoked flow solution proved unsuccessful.

A few variables would affect whether there was branching or not. These included the volume of the pipe, the rate at which the valve opened and the time step used in the solution. Figure 52 shows some typical results for various parameter values.



a) volume=0.019 m<sup>3</sup>, dt=0.1 sec, valve open rate=180°/sec

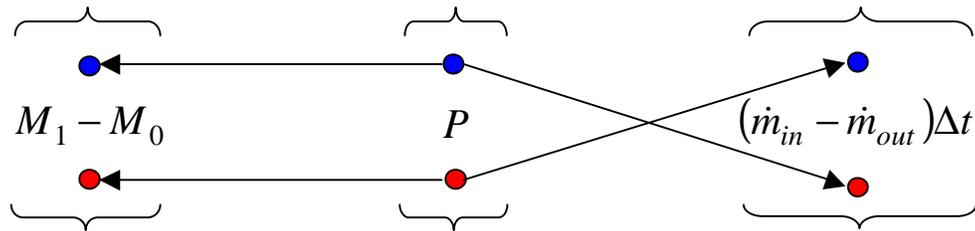
b) volume=0.019 m<sup>3</sup>, dt=0.1 sec, valve open rate=450°/sec



c) volume=0.03 m<sup>3</sup>, dt=0.05 sec, valve open rate=450°/sec

d) volume=0.03 m<sup>3</sup>, dt=0.05 sec, valve open rate=180°/sec

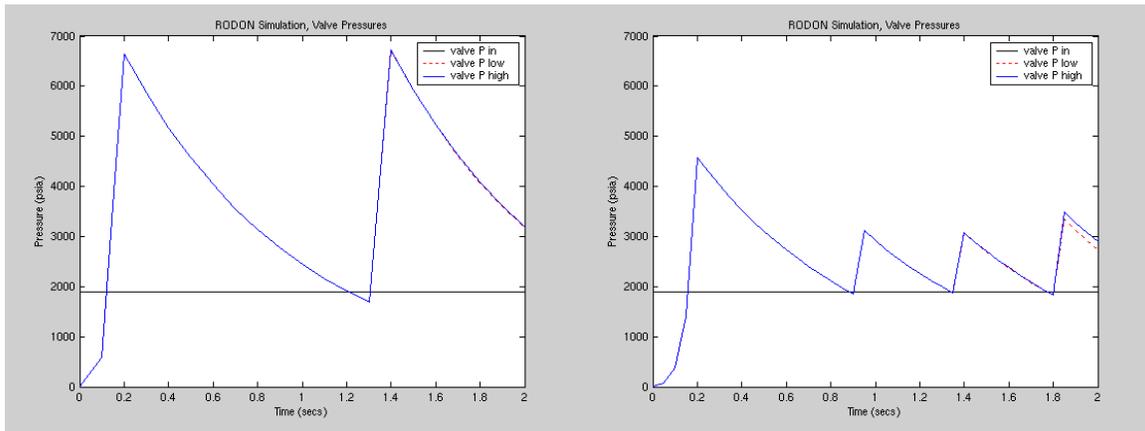
**Figure 52: RODON simulation attempts on a valve, pipe, and orifice system using mass flow at the end of the time step.**



**Figure 53: Example of inverse interval relationship.**

In the discussion above, the implementation of the conservation of mass equation assumes that the mass flow into and out of the pipe during the time step is calculated using the pressure at the end of the time step. It is solving the following problem: find the pressure in the pipe such that the net mass flow into the pipe as calculated with that pressure would be the same as the resulting change of mass in the pipe from the previous time step. It was believed that iterating on pressure to satisfy this constraint was causing the branching and so an alternate implementation of the mass equation was considered. Instead of using the pressure at the end of the time step to calculate the mass flow into and out of the pipe, the pressure at the beginning of the time step was used. We are now solving the following problem: using the current pressure, calculate the net amount of mass flow into the pipe for the time step duration; at the end of the time step, calculate the pressure that corresponds to the new total mass in the pipe. Figure 54 shows the results of such an approach. The calculated pressure in the pipe exceeds the input pressure to the valve. This is an artifact of the discrete time step and of course would not happen in a physical system. Since reversed flow through the valve was not modeled, the

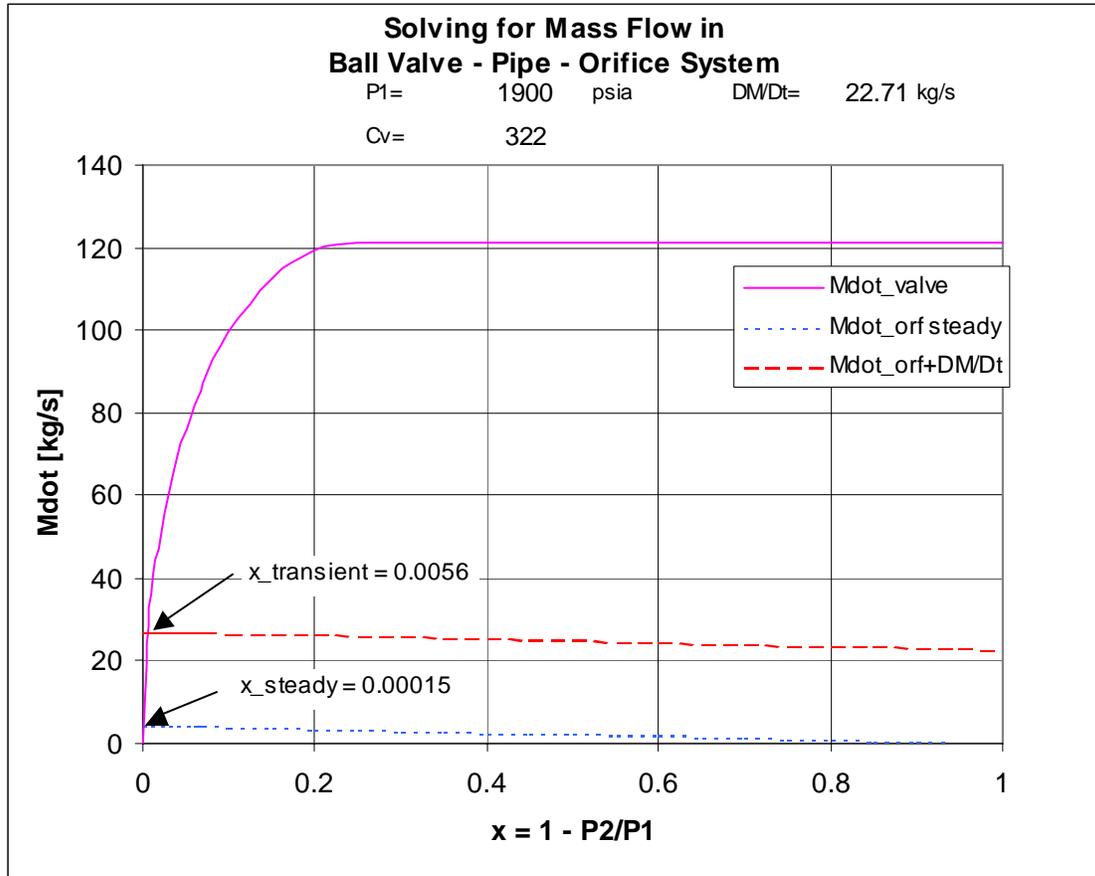
valve mass flow was set to zero in this situation and the pressure bleeds down until the pressure falls below the input pressure. At this point the pressure jumps up again because the calculated valve mass flow is very large. Figure 55 demonstrates that the valve mass flow is extremely sensitive to the pressure for the mass flows of interest. The intersection of the orifice mass flow line and the valve mass flow curve is the solution of the conservation of mass equation. The steady state solution corresponds to a mass flow of approximately 4.4 kg/sec. A small change in pipe pressure from this solution results in a large change of valve mass flow and leads to the saw-tooth behavior observed in the figures. Decreasing the time step can reduce the problem but not eliminate it. It is also likely that introducing numerical damping would help the problem. Because of the problems with converging on a steady-state solution, this approach was abandoned.



a) volume=0.019 m<sup>3</sup>, dt=0.1 sec, valve open rate=450°/sec

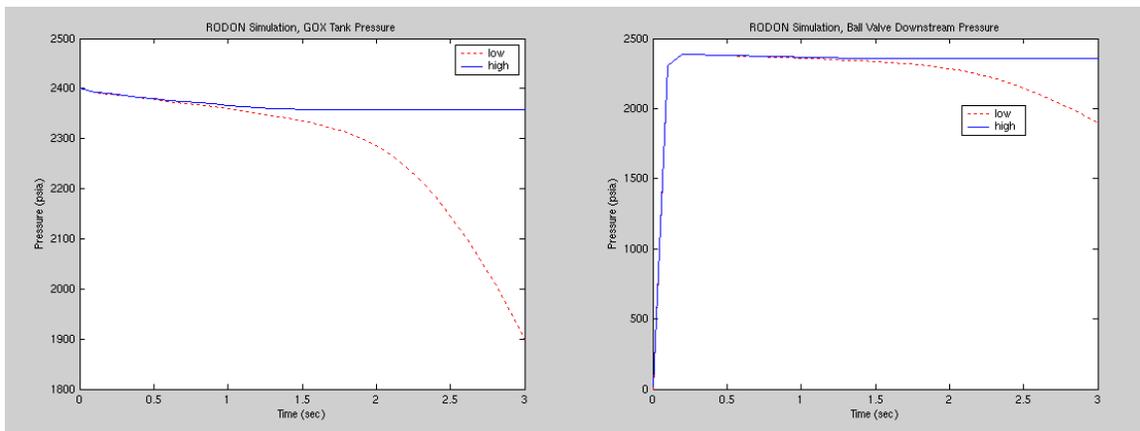
b) volume=0.019 m<sup>3</sup>, dt=0.05 sec, valve open rate=450°/sec

**Figure 54: RODON simulation attempts on valve, pipe, and orifice system using mass flow at beginning of time step.**



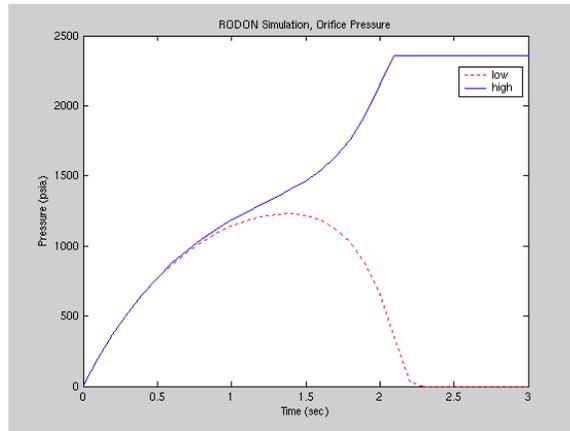
**Figure 55: Illustration of valve sensitivity to pressure in valve-pipe-orifice system.**

A simulation was attempted using the simplified model described in a previous section. This includes the tank and control valve in addition to the ball valve, pipe and orifice. Simulation parameters were chosen in such a way so that the solution did not branch as the ball valve opened. Figure 56 shows that the solution quickly diverges. Increasing the precision of interval computations could delay this divergence somewhat but still did not produce a useful result.



a) GOX tank pressure.

b) Valve downstream pressure.



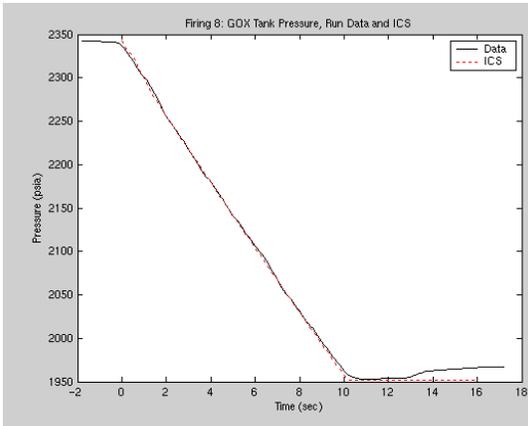
c) Orifice pressure.

**Figure 56: Illustration of divergence of RODON simulation using simplified model.**

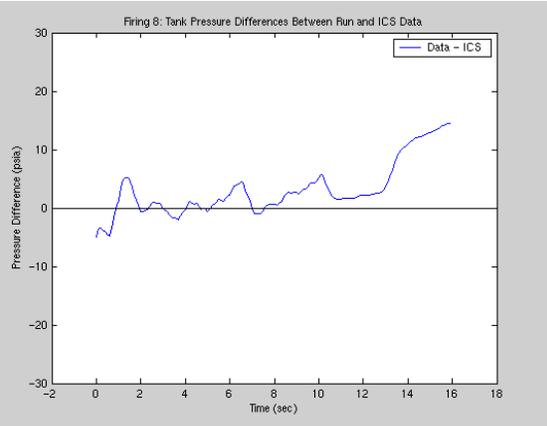
## 8 Simulations Using ICS

The interval constraint simulator was being developed while the HCF RODON models were being built. ICS proved to be an effective way to generate simulations that compared favorably to experimental data. ICS was able to squash the divergence seen in the RODON simulations by using the midpoints of all intervals as the initial conditions for the next time step. In addition, a PID control was implemented on the control valve.

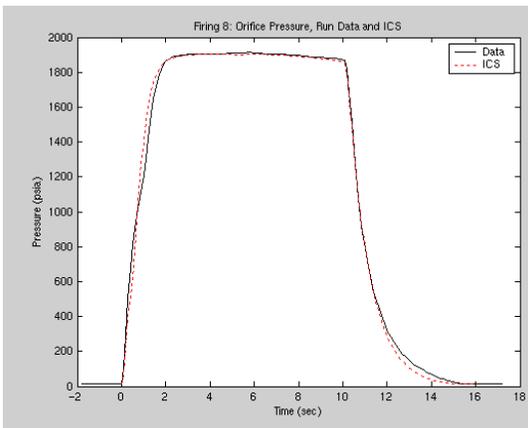
The results of the simulations can be seen in Figure 57 and Figure 58. For the data in the first set of figures, the control valve is assumed to respond immediately to the commands of the PID controller. The PID constants were tuned so that the ICS PID command would follow the actual valve position as closely as possible during the time the valve is maintaining constant orifice pressure. The most significant deviations from the experimental data occur just after the ball valve opens and as the control valve begins to move from its initial position. The simulation under predicts the pressure after the ball valve opens but before the control valve begins moving and over predicts the pressure after the control valve begins to move. The reasons for these discrepancies are evident in the control valve position plot (Figure 57e). After the ball valve is opened the force of the flow impacting the control valve causes the control valve to open slightly before decreasing to a level that is somewhat greater than its initial position. The increased area of the control valve increases the mass flow through the valve and causes a greater pressure at the orifice than would be predicted with the control valve position held constant. After the control valve position command jumps from the initial position to a new position, there is a slight delay before the valve actually moves in response to that command. In addition, the rate at which the valve moves is limited by mechanics of the valve. Therefore, the valve position will lag the commanded position and the predicted mass flow through the valve (and orifice pressure) will be greater than what is observed until the ICS command matches the valve position.



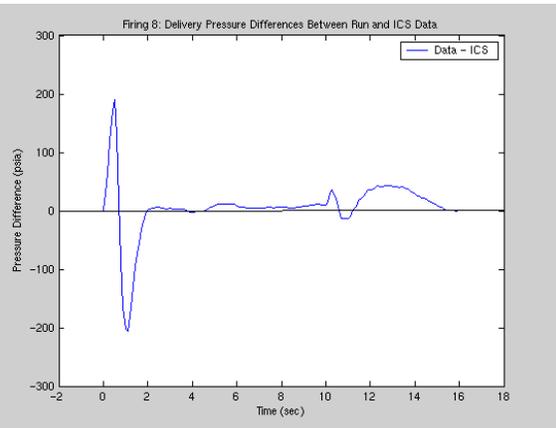
a) Tank pressure experimental data and simulation.



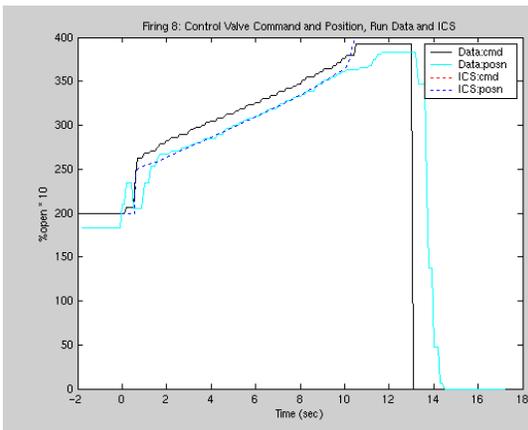
b) Differences between tank pressure data and simulation.



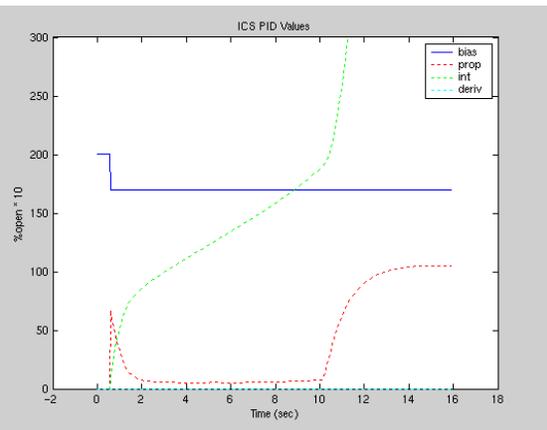
c) Delivery pressure experimental data and simulation.



d) Differences between delivery pressure and simulation.



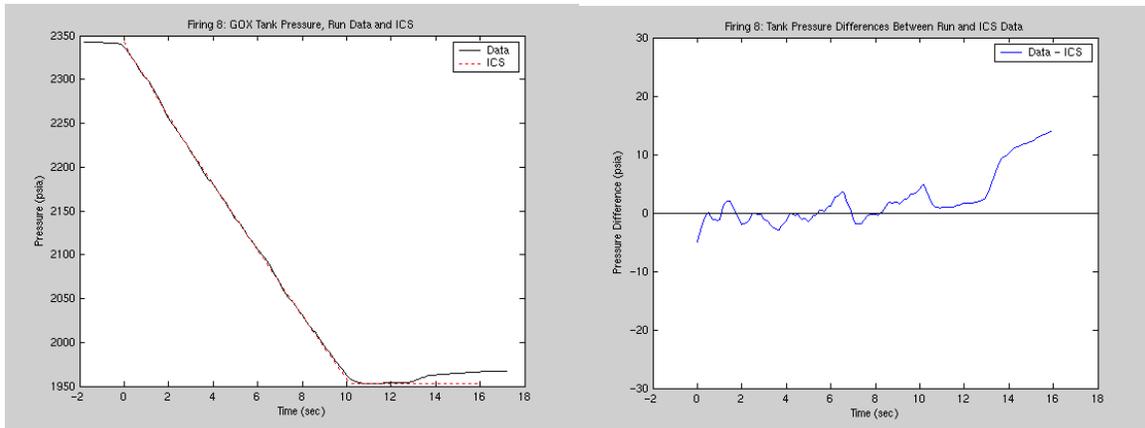
e) Control valve command and feedback position data and simulation.



f) ICS proportional, integral, and derivative contributions to command.

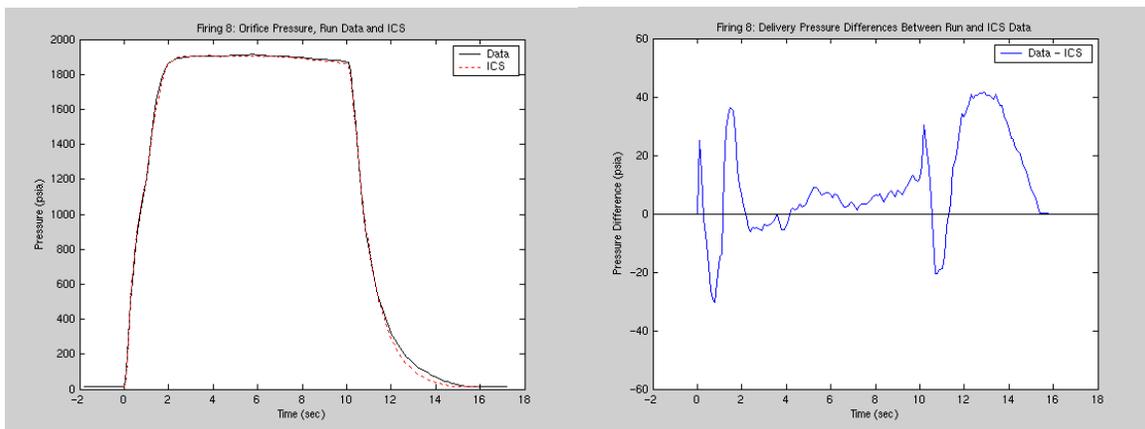
**Figure 57: ICS simulation of firing 8 using PID control for control valve position.**

We can check our explanation of the discrepancies by using the actual control valve position instead of the ICS commanded position. Figure 58 shows that the predicted tank and orifice pressures are in excellent agreement with experimental data. The tank pressure is generally within 0.2% of the full-scale pressure (3000 psia) and the orifice pressure is within 1% of the full-scale pressure (3000 psia). This fidelity comes without smoothing the choppy control valve position data.



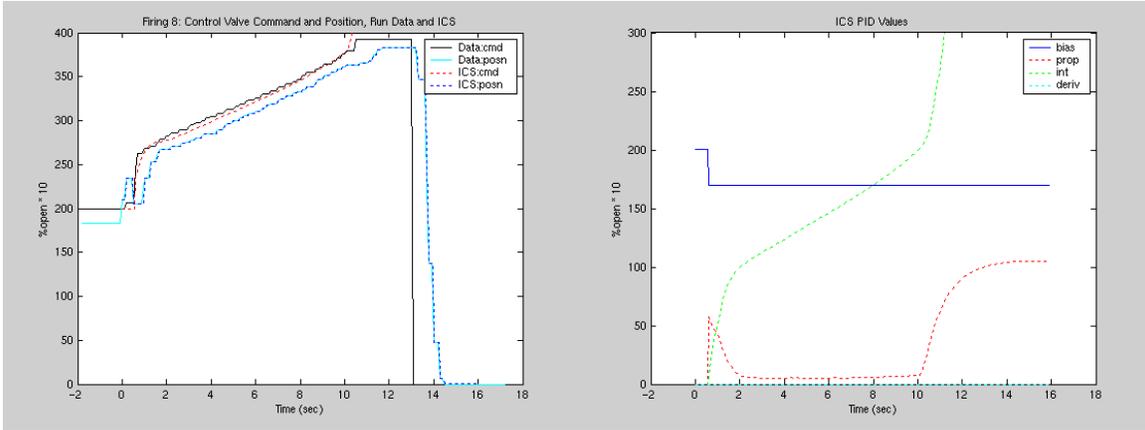
a) GOX tank pressure experimental data and simulation.

b) Differences between tank pressure data and simulation



c) Delivery pressure experimental data and simulation.

d) Differences between delivery pressure and simulation.



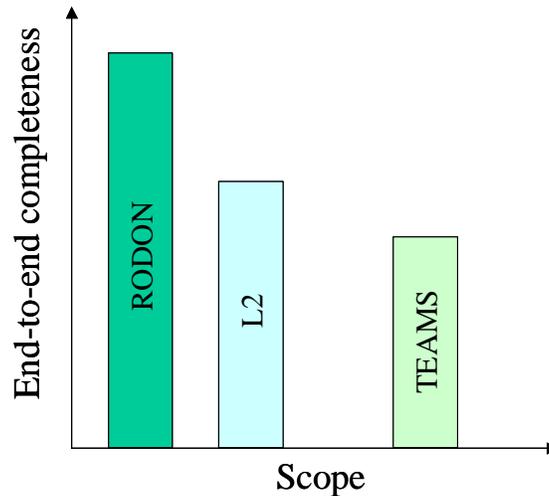
e) Control valve command and feedback position data and simulation.

f) ICS proportional, integral, and derivative contributions to command (not used).

**Figure 58: ICS simulation of firing 8 using experimental data for control valve position.**

## 9 Model Comparison

The scope and completeness of the models created with the different tools vary as depicted in Figure 59. The TEAMS model has the greatest scope, including all of the subsystems of the HCF. But the test code that is necessary to abstract the sensor data to the binary test results that TEAMS uses in the fault diagnosis was not developed. The L2 model has a reduced scope but the data is abstracted somewhat automatically using a spreadsheet application. However, the necessary function of the RTI was implemented manually. The RODON model has the smallest scope but is the most complete as far as being able to use the logged data directly to produce a diagnosis. The scopes of the ICS and IMS tools are the same as the RODON model.



**Figure 59: Notional comparison of HCF model scope and completeness.**

The TEAMS, L2, and RODON models closely resemble the physical structure of the HCF. Hardware components that have a specific function at the HCF are treated as the lowest level components in the model. For example, each open/close limit switch in the valve actuator is modeled as a component in each one of the tools. Assemblies of components at the HCF that work together to achieve and monitor some desired objective – like the shutoff valve, its actuator, and the limit switches – may be grouped together at a higher hierarchy level in the model. This approach was followed in the TEAMS and RODON models. For the L2 model a flat hierarchy was chosen. Here, all of the components in the model are defined at the same level, giving the modeler a quick overview of the entire system. Multiple hierarchy levels would have been used if the model schematic got too cluttered or confusing.

In each of the models, the components capture the functional behaviors of the modeled devices. In TEAMS, the functions affected by a component's failure are highly abstracted to a list of signals, or attributes. These signals can be thought of the independent variables that describe the functions of the component. The data acquired from sensors, represented as test points in the model, are also abstracted to a list of signals that represent the system attributes the sensors are monitoring. The signals attached to a test may be attached to one or more components in the model. Similarly, the signals attached to a component may be attached to zero, one, or many tests. The signals relate the failure causes (the components) to the manifestations of the failure effects (the tests). The links in the directed graph are used to determine which faults are observable at each test. A test may have only a binary yes/no (or pass/fail) result but there may be many tests associated with one system sensor. A failed test implies that one or more of the attached signals is witnessed (or implicated) in the failure. TEAMS compiles which components on the propagation path to the test point have one or more of those signals attached as well and are therefore suspected in the failure. By combining multiple test results, the number of suspected components is reduced. While nominal behavior is not modeled explicitly, the device is working properly if all of the tests defined at the test points pass.

Some important aspects of modeling with TEAMS include creating tests from the facility measurements, designing the tests to provide binary results, defining the signals, determining which signals to attach to each test and component, and dealing with system transients. What information do the sensors provide, how many tests are required to extract the features of interest, and how can the features be captured accurately in terms of binary test results? If a test fails, what does it mean in terms of system operation and how do we associate significance to that event by attaching signals that represent the system attributes? Similarly, if a component fails, in what ways does it fail? Can we attach a signal that represents a specific way in which the fault is manifested or do we need to allow for a general failure that will affect all of its functioning and show up in any test connected downstream of the component? We may decide to do both. Do the tests remain relevant during the entire system operation or do we need to enable and disable tests or even parts of the model depending on the operating mode? For example, a valve that is opened or closed will change the expected downstream pressure measurement. A test that checks if the pressure is near ambient downstream of a closed

valve is no longer relevant once that valve is opened. To handle different system configurations, switches may be inserted into the model that control which tests and components are active in that mode of operation or logic may be inserted to the test code to control which tests are active for different configurations. The test results and relevant switch configurations must be generated externally to the model. System transients are also handled outside of the model. Tests must be carefully coded not to give incorrect results during transients.

In L2, the nominal and failure behaviors of the component are described by propositional formulae that relate, or constrain, the qualitative input and output values of the components. A component may have a number of nominal modes that represent the configurations of the modeled device, like an open or closed valve. The modes have different constraints among the input and output variables that capture the function of the element in that particular configuration. Transitions between the nominal modes occur when commands are given to the device. Each component also has one or more failure modes. The modes may have constraints that model a specific way in which the device fails or have no constraints to allow for behavior that is not modeled. System modes are handled by the composite modes of the components and do not require switches or external test logic as in TEAMS. Facility sensor readings are abstracted to a qualitative value by assigning the data point or a derived quantity of the data (e.g., the derivative) to one of a finite number of bins that divide the real number line into a discrete space. The qualitative values are assigned to the variables at the terminals of the sensor components in the model. As in the other tools, links connecting the components pass information between them. This information may correspond to fundamental measurements such as voltages, currents, pressures, temperatures, and switch positions, or derived quantities such as the mass flow rate or the pressure rate of change. L2 is a predictive model; the assignment to the values of some of the inputs or outputs of a component, together with the propositional formulae in the commanded nominal mode, constrain the undefined input or output variables to take on specific values in their domain. Therefore, given the values of some of the sensors, L2 can predict the qualitative values of other sensors in the model. If the abstracted sensor value is asserted to be different than the predicted value, some of the constraints in the nominal model description (implemented as a set of clauses) are violated and the system is inconsistent. An efficient search procedure then tries to find the most probable combinations of each component's state (one of the nominal or failure modes) that make the observations fit with the constraints that define the system behavior. L2 revises the list of candidates as more information becomes available; it will remove inconsistent candidates and suggest new ones, perhaps adding another fault to the ones observed earlier or changing the original fault assumption.

Some important aspects of modeling with L2 include abstracting the real-valued space of the sensor data to a discrete space that captures the behavior of the system, defining the component functionality in the qualitative terms of the discrete sensor values, and dealing with transients of the system. For each sensor, what are the values it reports during system operation? How do they depend on the modes of the system? How many bins are necessary to characterize the expected values for the various system modes? Is there much variation in the observed values for a particular mode—where do we place the

upper and lower bounds of the bins to identify faulty behavior quickly while avoiding false positives? Might the bins overlap? If so, we need to define multiple variables for the value of a sensor so that some values in each variable's domain can correspond to the same segment of the real valued number line. Do the bins change depending on the system run settings and initial conditions? For each component, how are the inputs and outputs related for a particular mode in terms of the discrete values of the variables? Do we have to assign a value to a variable that may be important in another component in order to propagate its value through the model? Are system transients an important aspect of the system operation? Do we create additional bins and modes to attempt to diagnose during transient periods or do we just model the steady-state conditions and wait for transients to settle before diagnosing? Would we be missing important signatures that might help to isolate failures if we simply waited for the transients to settle? If some measurements have shorter settling times (limit switches vs. pressures) we may assert their values while making no assertions about other measurements that may not have stabilized. Some consistent policy for dealing with system transients must be designed and implemented. Typically, much of this happens outside of the L2 model.

In RODON, the nominal behavior of the component is specified with quantitative and/or qualitative formulae and logic clauses. The logic clauses can be used to define various operating modes. For example, a component variable is used to represent the state of a valve. An OR clause is used to say that the valve can either be in the opened or closed state. AND clauses are used in each state to include all equations relevant to that state. An IF clause checks whether the valve state variable has an opened or closed value (set by the command to the valve) and enforces the appropriate equations. Unlike L2, the variables in the equations may be continuous; they do not have to be discretized. Furthermore, the formulae may be mathematical equations. This means that it is possible to describe the system evolution, including transients, with equations that include time as a variable. It is not required to define fault states (modes) but they may help with fault isolation. Facility sensor data can be fed directly into the model without abstraction. The reported value may have an associated tolerance that accounts for the uncertainty of the sensor measurement. Links in the model propagate interval values rather than qualitative signals or values (although it can propagate this type of information as well). RODON dynamically resizes the intervals of the variables in the model so that all values in an interval satisfy the equations (implemented as a constraint network) that characterize the nominal system operation. Like L2, the model is predictive. Instead of predicting a qualitative value, RODON predicts an interval of possible values. If the measured sensor value, after factoring in uncertainty, does not fall within the predicted interval the constraint network is violated and the system is inconsistent. A search procedure is used to find suspect components. The constraints from a component are removed from the constraint network and the consistency is rechecked. If the constraint network is made consistent by removing the constraints of a component, that component is suspect. If the constraints of a particular fault mode are consistent that fault mode is presented as a suspected failure.

Some important aspects of modeling with RODON include acquiring sufficient information to characterize a component's behavior, deciding when to perform

calculations in the monitoring code instead of the model, determining how to implement the temporal behavior of the system, and allowing for imperfections in the model when assigning tolerances. A quantitative model using mathematical equations to specify the behavior of the components requires more information about the components than qualitative models. Has the component been fully characterized so that the equations or transfer functions relating input and output parameters are known? Can we neglect certain parameters in the modeled component or even use a qualitative approach if the equations are expected to have a small effect on the intended function? Should all of the calculations be done in the model so that the uncertainties in the prime variables are propagated correctly or can we perform some calculations externally to RODON to speed up the execution time without compromising the validity of the model? How should we linearize the equations in the model? Which components need to have equations involving time and which variables need to be saved from one time step to the next? Can we use sensor uncertainty specifications as tolerances or do we need to allow for inaccuracies in the model of the system by increasing the tolerances of the variables beyond the uncertainty levels?

Note that TEAMS models have no notion of time. Temporal progression of a system is dealt with by using TEAMS-RT together with code that tracks the modes of the system and feature extraction software. L2 models have a discrete time that advances when commands are issued to the system. Therefore, the time steps usually correspond to a configuration change and the actual time between time steps is not uniform. Similar to TEAMS, temporal aspects of the system are mostly handled by the monitor code and the RTI. RODON models may have uniform time steps that correspond to actual time. Table 23 summarizes the tool comparison.

	<b>TEAMS</b>	<b>L2</b>	<b>RODON</b>
Behavior description	Via signals in multi-signal flow graph (describes fault propagation)	Via propositional logic	Via logic clauses and numerical equations with interval arithmetic
Abstraction	To binary pass/fail	To discrete bins	Not necessary
Nominal behavior	In test points	In nominal modes	In nominal description
Transients	External	External	Internal or external
Diagnostic strategy	Pre-compiled fault dictionary	Conflict directed best-first search	Constraint suspension

**Table 23: Tool comparison.**

Finally, note that there are similarities between the TEAMS abstractions to binary test results and L2 abstractions to bins. Consider for a moment an L2 abstraction that has three bins: low, nominal, and high. Let's consider these values to be the pressure downstream of a valve. When the valve is closed we expect the value to be low, when the valve is open we expect the value to be nominal (e.g., equal to a pressure that is regulated upstream), and a value of high represents a failure in an upstream component—

for example, a regulator that is regulating high. In TEAMS, we can define three tests to capture this same information. The first test will be: is the pressure below the threshold value that separates the low bin and nominal bin (i.e., in the low bin)? The second test will be: is the pressure above the threshold level that separates the low bin and nominal bin (i.e., in the nominal or high bins)? Notice that we did not test for just the nominal bin; more will be said about this later. The third test will be: is the pressure above the threshold level that separates the nominal bin and high bin (i.e., in the high bin)? For this test, an answer of “yes” means the test has failed; for the other tests an answer of “no” means the test has failed. [Alternatively, we could have posed the last question as: is the pressure *below* the threshold level that separates the nominal bin and high bin]. Since a valve mode is involved, a switch is placed between the TEAMS valve component and the tests<sup>5</sup>. Two of the tests, the second and third tests, are enabled when the valve is open and the first test is enabled when the valve is closed. We attach the same signal to the first test and to the valve component. We do not attach the signal to the regulator since a failure of the regulator, by itself, would not explain pressure downstream of a closed valve being greater than the low bin. We attach another signal to the second test and to both the valve and the regulator since a failure of either one could explain the pressure being below the nominal bin. To help discriminate whether the regulator or the valve has failed, we can use the third test. We attach a signal to the high bin test and the regulator. We do not attach it to the valve since a valve failure will not produce higher pressure downstream than upstream. If the second test passes and the third test fails (i.e., the pressure is in the high bin), we suspect the regulator but if the second and third tests fail (i.e., the pressure is in the low bin), we suspect either the regulator or the valve; we cannot isolate the failure. If we had defined the second test to check if the pressure was in the nominal bin instead of either the nominal or high bins, then both the second and third tests would have failed if the pressure were in the high bin and we would have had both the valve and the regulator as suspects instead of just the regulator.

## 10 Fault Detection and Diagnostic Demonstrations

In this section we will use the HCF models to demonstrate a diagnosis with L2, TEAMS and RODON. In addition, an example of fault detection using IMS is presented. Aside from IMS, computational requirements were not measured. As expected, the more qualitative the data becomes, the faster the performance.

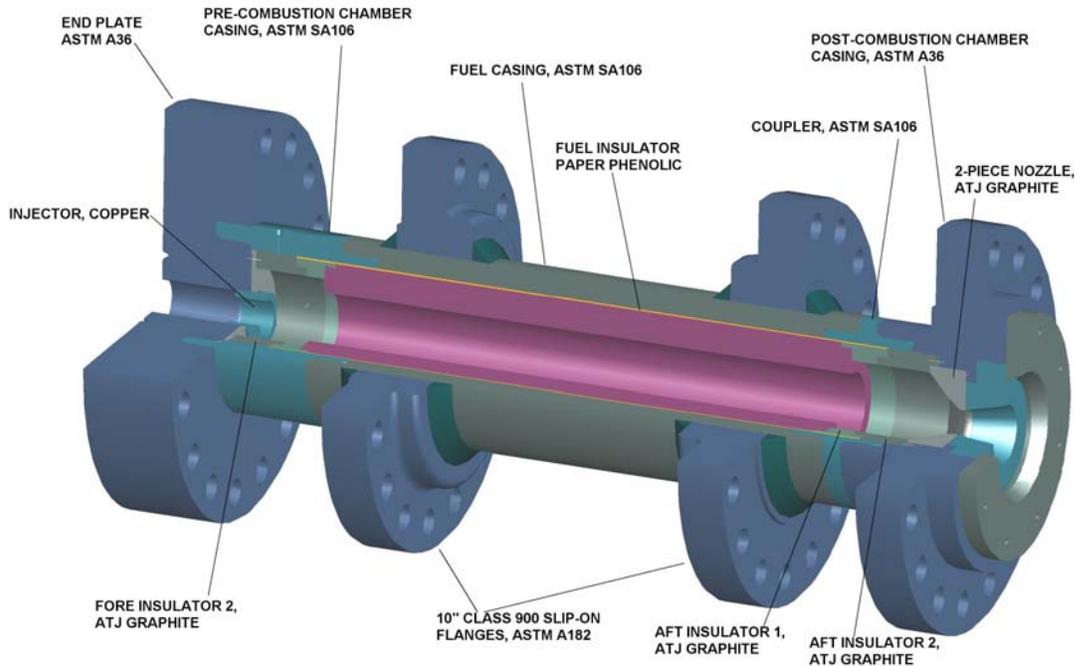
### 10.1 L2

For a diagnostic demonstration using L2 we will consider HCF firing 2, which had a failure. During this firing, an insulator in the pre-combustion chamber failed and the fragments impacted and removed a section of the exhaust nozzle. The sudden increase in the nozzle exit area caused a sharp drop in combustion chamber pressure. Figure 60 [1]

---

<sup>5</sup> It is not necessary to place a switch in the model in this case. One could also assign test labels to the test points that would correspond to when the valve was opened or closed. During run time, logic in the test (like whether the valve has been commanded open) would be used to determine whether a particular test reports a result or not.

shows a perspective, cut-away view of the combustion chamber. In addition to the physical fault, the controller did not properly record the control valve feedback position; it logged a value of  $-7\%$  for the entire run even though the valve obviously moved. L2 should be able to catch both of these faults.



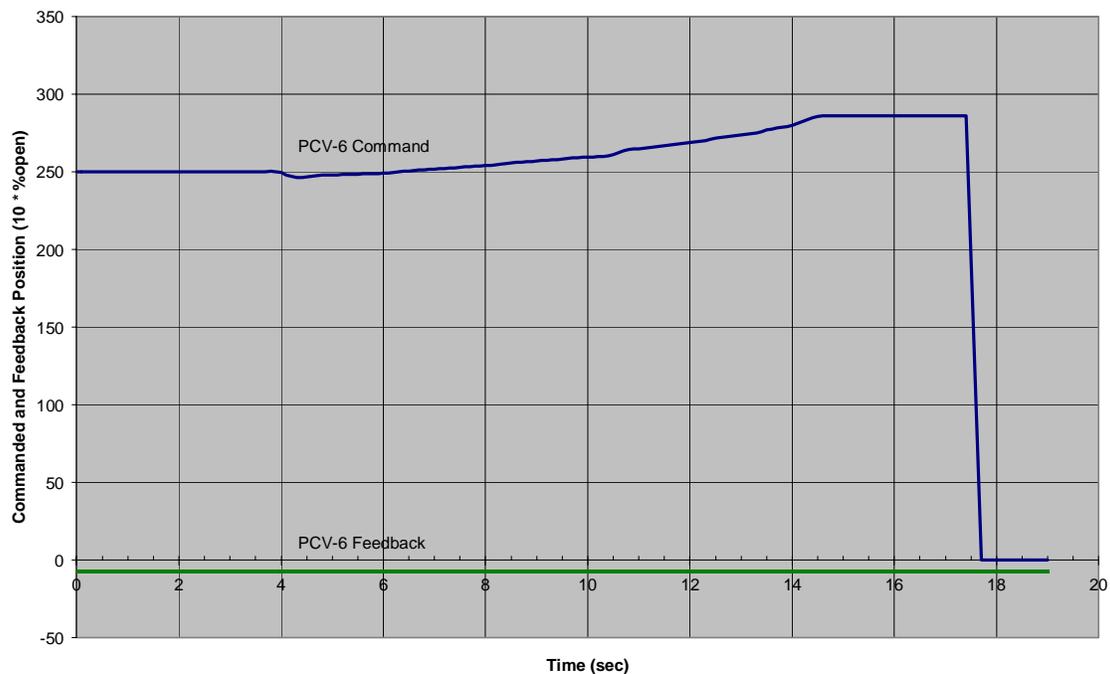
**Figure 60: Perspective view of the combustion chamber.**

Figure 61 shows the smoothed command and feedback traces for the control valve. Figure 62 shows the pressure traces during the run. The expected combustion chamber pressure for this firing was 600 psi. Note the sharp drop in chamber pressure near Time = 5.5 sec. This corresponds to the insulator failure and subsequent nozzle failure. The times when other events occur are as follows:

Event	Time (sec)
POV-4 Open command	3.1
POV-4 Not Closed feedback	3.4
POV-4 Open feedback	3.6
Chamber Ignition On command	4.5
Chamber Ignition Off command	5.7
POV-4 Close command	13.9
POV-4 Not Open feedback	14.3
POV-4 Closed feedback	14.5
POV-5 Close command	14.5
POV-5 Not Open feedback	17.2
POV-5 Closed feedback	17.5
PCV-6 Closed feedback	18.3

**Table 24: Firing 2 event timing.**

**Run 2 PCV-6 Command and Feedback vs. Time**



**Figure 61: Firing 2 smoothed control valve command and feedback position traces.**

Run 2 Pressures vs. Time

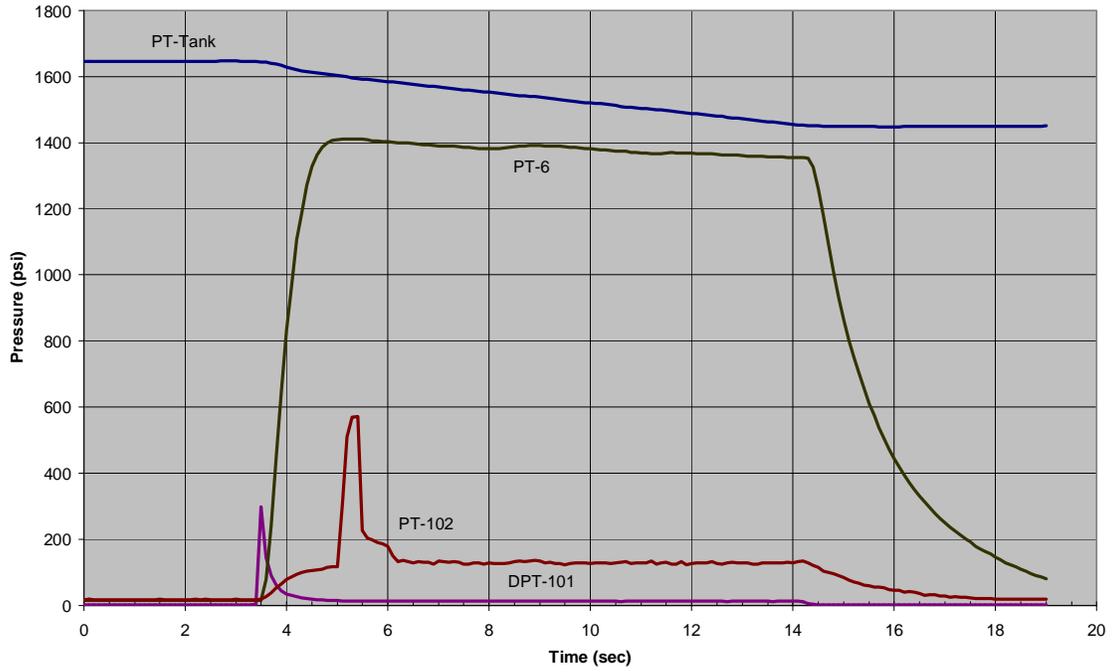


Figure 62: Pressure traces of firing 2.

As mentioned previously, L2 requires that the real-valued sensor data be abstracted to discrete space. Table 25 shows the numerical values assigned to the variable bins for a run of 2 kg/sec and expected chamber pressure of 600 psi. Different values would necessarily be assigned for different run conditions.

Quantity measured	Type	Threshold	Value
Tank pressure	Level (psi)	Low/nominal	1418
		Nominal/high	1660
	Rate (psi/s)	Drop fast/drop slow	-45
		Drop slow/steady	-3
		Steady/rise	5.5
Control valve command	Position (10 * %open)	Open low/open high	252
Control valve position feedback	Position (10 * %open)	Closed too much/open low	160
		Open low/open high	242
		Open high/open too much	410
	Rate (10 * %open /s)	Closing/not moving	-4
		Not moving/opening slow	0.1
		Opening slow/opening nominal	0.4
		Opening nominal/opening fast	15
Supply pressure (PT-6)	Level (psi)	Ambient/low	110
		Low/nominal	1273
		Nominal/high	1429
	Rate (psi/s)	Drop fast/drop slow	-350
		Drop slow/steady	-50
		Steady/rise	40
Temperature (TT-101)	Level	Low/nominal	208
		Nominal/high	316
Burst disk	Level (psi)	Threshold	2200
Chamber pressure	Level (psi)	Ambient/low	22
		Low/GOX preIgnition	105
		GOX preIgnition/intermediate	137
		Intermediate/combustion	470
		Combustion/high	650
	Rate (psi/s)	Drop fast/drop slow	-120
		Drop slow/steady	-60
		Steady/rise slow	45
		Rise slow/rise fast	135

**Table 25: Thresholds for 2 kg/sec, 600 psi firing.**

Using the procedure described in section 7.2.12, the following scenario for firing 2 was hand generated:

```

scenario Run_02 HCF
1 {
  assign test.pt_tank.pressureReading.level=nominal
    test.pt_tank.pressureReading.rate=steady
    test.pt_tank.pressureReading.burstDiskThreshold=belowThreshold
  assign test.zs_4_closed.closedReading=closed
  assign test.zs_4_open.openReading=notOpen
  assign test.zs_5_closed.closedReading=notClosed
  assign test.zs_5_open.openReading=open
  assign test.zt_6.reportedValues.position=closed
    test.zt_6.reportedValues.action=notMoving
  assign test.dpt_101.pressureDifferenceReading=ambient
  assign test.pt_6.pressureReading.level=ambient
    test.pt_6.pressureReading.rate=steady
    test.pt_6.pressureReading.burstDiskThreshold=belowThreshold
  assign test.zsc6.closedReading=notClosed
  assign test.tt_101.temperatureReading=nominal
  assign test.bd_100.burstSignal=notBurst
  assign test.bd_101.burstSignal=notBurst
  assign test.pt_102.reportedPressure.level=ambient
    test.pt_102.reportedPressure.rate=steady
  fc
2 {
  progress test.pov_4.valveCmdIn=open
  assign test.pt_tank.pressureReading.rate=dropSlow
  assign test.zs_4_closed.closedReading=notClosed
  assign test.dpt_101.pressureDifferenceReading=low
  assign test.dpt_101.pressureDifferenceReading=high
  assign test.zs_4_open.openReading=open
  assign test.pt_6.pressureReading.rate=rise
  assign test.pt_102.reportedPressure.level=low
  assign test.pt_6.pressureReading.level=low
  assign test.pt_102.reportedPressure.rate=riseSlow
  fc
3 {
  progress test.combustionChamber.ignitionStart=on
  assign test.pt_102.reportedPressure.level=gOXpreIgnition
  assign test.pt_6.pressureReading.level=nominal
  assign test.pt_102.reportedPressure.rate=steady
  fc
  assign test.dpt_101.pressureDifferenceReading=nominal
  assign test.pt_6.pressureReading.rate=steady
  assign test.pt_102.reportedPressure.level=intermediate
    test.pt_102.reportedPressure.rate=riseFast
  assign test.pt_102.reportedPressure.level=combustion
  fc
4 {
  assign test.pt_102.reportedPressure.level=intermediate
  progress test.combustionChamber.ignitionStart=off
  unassign test.pt_102.reportedPressure.rate
  fc
  assign test.pt_102.reportedPressure.level=gOXpreIgnition
  fc
  progress test.pcv_6.commandIn=openHigh
  assign test.pt_102.reportedPressure.rate=steady
  fc
  progress test.pov_4.valveCmdIn=close
  unassign test.zs_4_open.openReading
  unassign test.zs_4_closed.closedReading
  unassign test.pt_tank.pressureReading.rate
  unassign test.pt_6.pressureReading.rate
  unassign test.pt_102.reportedPressure.rate
  progress test.pcv_6.commandIn=closedTooMuch

```

```

progress test.combustionChamber.ignitionStart=off
fc
assign test.zs_4_open.openReading=notOpen
assign test.dpt_101.pressureDifferenceReading=low
assign test.pt_6.pressureReading.rate=dropSlow
fc
progress test.pov_5.valveCmdIn=close
assign test.zs_4_closed.closedReading=closed
assign test.pt_6.pressureReading.level=low
    test.pt_6.pressureReading.rate=dropFast
assign test.pt_102.reportedPressure.level=low
fc
assign test.pt_tank.pressureReading.rate=steady
assign test.pt_102.reportedPressure.rate=dropSlow
fc
assign test.pt_102.reportedPressure.rate=steady
fc
assign test.dpt_101.pressureDifferenceReading=ambient
assign test.pt_6.pressureReading.rate=dropSlow
fc
assign test.zs_5_open.openReading=notOpen
fc
progress test.pov_5.valveCmdIn=limitSwitchClosed
assign test.zs_5_closed.closedReading=closed
assign test.pt_102.reportedPressure.level=ambient
fc
progress test.pcv_6.commandIn=closedComplete
assign test.zsc6.closedReading=closed
fc
assign test.pt_6.pressureReading.level=ambient
fc

```

Each line signifies a change in a variable value and the events are in sequential order. “Assigns” are used to tell L2 that the value is an observation. “Progresses” inform L2 that the value is a command. “fc”, short for “find candidates”, instructs L2 to perform a diagnosis. An “unassign” statement is used to set the variable value to “unknown”, which means that any inferred value for the variable is consistent. The scenario is the sequence of events and L2 functions that would be generated by the monitors and RTI. The scenario file is examined in the following paragraphs and the diagnosis results are discussed.

In the first part of the scenario, labeled “1”, the initial values of the sensors are given. The only value that is not nominal is the control valve position feedback, which reports “closed” when it should be reading “open low”. After the “fc”, L2 gives the diagnosis shown in Figure 63. Three distinct candidates are listed (the classes tab lists the distinct candidates). The ranks relate to the probability given to the fault mode(s) by the modeler, with lower ranks corresponding to higher probabilities. The first candidate indicates that the position feedback, ZT-6, could be faulty. The second candidate indicates that a problem with the valve has resulted in the valve being too far closed. There are currently no observations that would contradict this, so it remains a candidate. This candidate is removed from the list later in the scenario. The final candidate is an unknown failure in the control valve, which has low probability.

Num	Rank	Time	Failures
0	3		zt_6=faulty
1	3		pcv_6=tooFarClosed
2	10		pcv_6=unknownFailure

CBFS: exhaustive search, returned fewer than 64 candidate(s) (searche...

**Figure 63: Diagnosis after first block of scenario file.**

The primary shut-off valve, POV-4, is commanded open in the second block of the scenario. The candidates remain the same. The additional statements do not add any constraints that contradict the faults from the previous block, nor do they provide evidence of additional faults.

The ignition command is given in the third block of the scenario. At this time, the pressure in the GOX line reaches the nominal level. After approximately a half second, the combustion chamber pressure rises fast and reaches the combustion level. Figure 64 shows the diagnosis after this block. Because the pressure in the GOX line has reached the nominal level, the candidate that states the control valve is too far closed has been eliminated. If the control valve were too far closed, we would not expect the downstream pressure to reach the nominal level.

Num	Rank	Time	Failures
0	3		zt_6=faulty
1	10		pcv_6=unknownFailure

CBFS: no search, remaining consistent candidates

**Figure 64: Diagnosis after third block of scenario file.**

In the fourth block of the scenario the pressure drops to the intermediate level before the ignition is terminated. When the ignition is commanded off, the combustion chamber component attempts to transition to the *after ignition* mode and the constraints in that mode are checked for consistency. The pressure rate is constrained to be steady but we have allowed for an expected momentary decrease in pressure immediately after ignition termination by unassigning the pressure rate that is calculated from the combustion chamber pressure sensor. However, the pressure level is expected to be in the “combustion” bin but its value is “intermediate”. Therefore, a conflict results with the *after ignition* mode and a search is made for mode assignments that make the system

consistent. Figure 65 shows the first ten candidates that result from the search procedure. The most likely candidate indicates a fault with the combustion chamber in addition to the fault in the control valve feedback. The next two candidates implicate sensor errors for the unexpected observations. The other candidates offer more unlikely possibilities. Running the rest of the scenario does not significantly alter the candidates.

Num	Ra	nk	Time	Failures
0	5	3		zt_6=faulty
		2		combustionChamber=faulty
1	6	3		zt_6=faulty
		3		pt_102=faulty
2	6	3		zt_6=faulty
		3		pt_6=faulty
3	8	3		zt_6=faulty
		5		sonicNozzle=faulty
4	9	3		zt_6=faulty
		6		ckv_7=unknownFailure
5	12	10		pcv_6=unknownFailure
		2		combustionChamber=faulty
6	13	10		pcv_6=unknownFailure
		3		pt_102=faulty
7	13	3		zt_6=faulty
		10		bd_101=unknownFault
8	13	10		pcv_6=unknownFailure
		3		pt_6=faulty
9	13	3		zt_6=faulty
		10		bd_100=unknownFault

CBFS: search candidate(s) limited by max search space = 3000

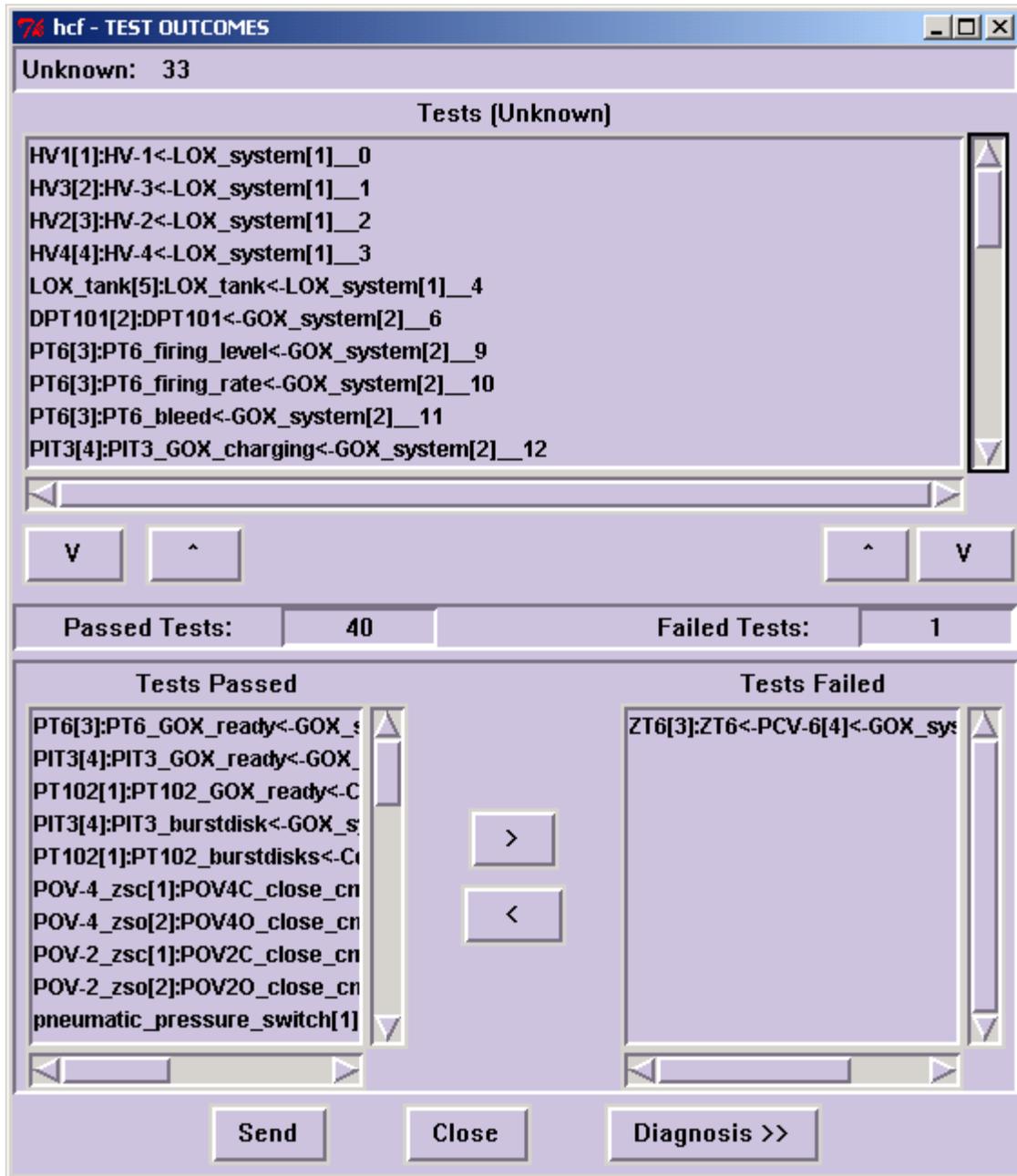
**Figure 65: Diagnosis after fourth block of scenario file.**

## 10.2 TEAMS

We will use the same firing (number 2) as in the previous section to illustrate a TEAMS diagnosis. The demonstration does not include the feature extraction and test logic code that would be necessary for a diagnosis directly from the data. Instead, test results are inferred from an examination of the data and the anticipated outcomes of properly designed test code. Test results are injected manually into TEAMS-RT, which generates a diagnosis based on the model that was exported from TEAMS.

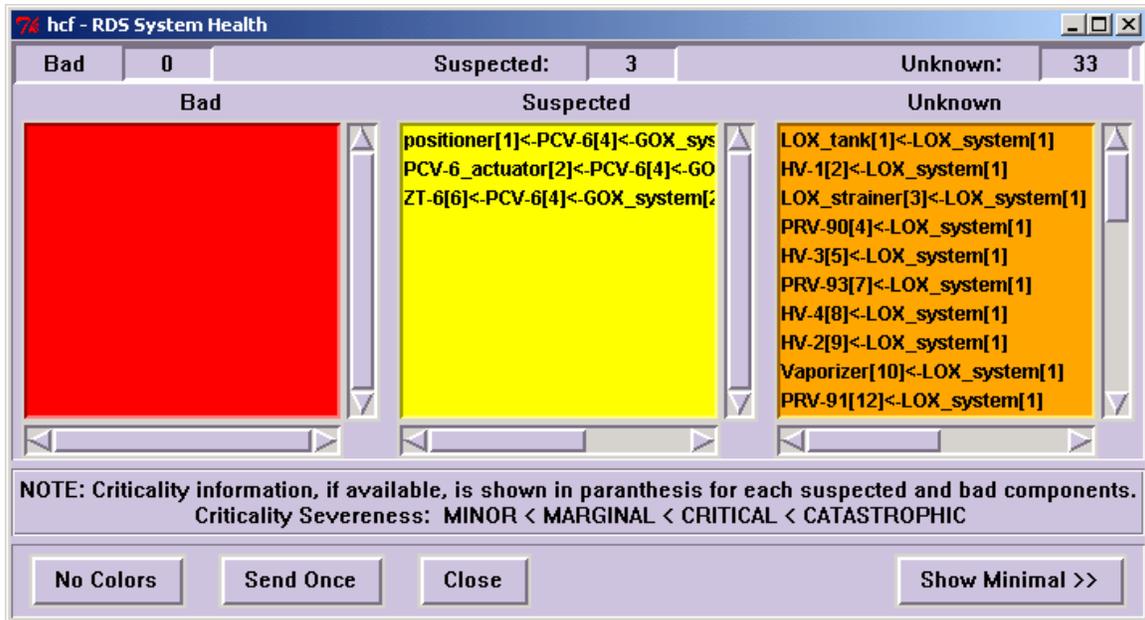
Prior to POV-4 opening, the tank pressure is holding steady, the downstream pressure sensors indicate ambient pressure, and the PO valve limit switches all report the correct positions. The control valve feedback, however, indicates the valve position is at -7% open while being commanded to 20% open. Of the 74 tests defined in the TEAMS model, we pass those that are satisfied, fail the test that compares the commanded and

feedback control valve positions, and leave those that are not relevant to the current configuration unknown as displayed in Figure 66.



**Figure 66: TEAMS-RT tests before POV-4 opens.**

If “Diagnosis” is clicked, three suspected components are listed as shown in Figure 67: the control valve positioner that converts the electrical signal to a pneumatic pressure, the control valve actuator, and the position feedback sensor. We have no information regarding 33 of the failure sources in the model. The rest, 117 (# of failure sources, see Figure 25) – 33 – 3 are considered good.



**Figure 67: TEAMS-RT diagnosis before POV-4 opens.**

After POV-4 opens, some of the tests become irrelevant while others will now apply. Continuing to play the role of the test code, we manually configure the test results as shown in Figure 68 after the ignition command is sent.

Because of the tests that are now passing, which imply proper operation of the control valve, the diagnosis changes to what is shown in Figure 69. The position feedback sensor has moved from the suspected components box to the bad components box while the other components that had been suspected have been cleared.

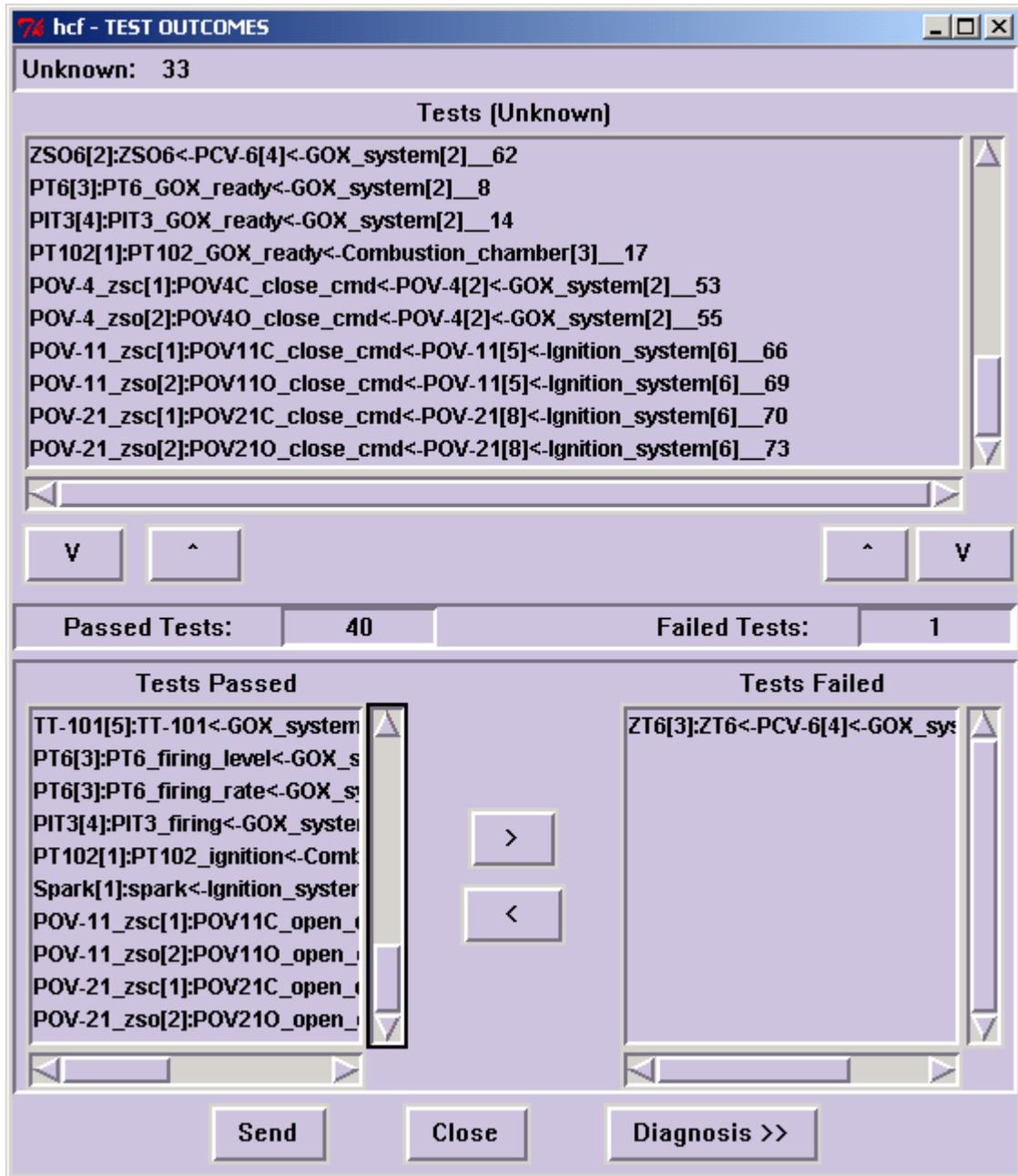
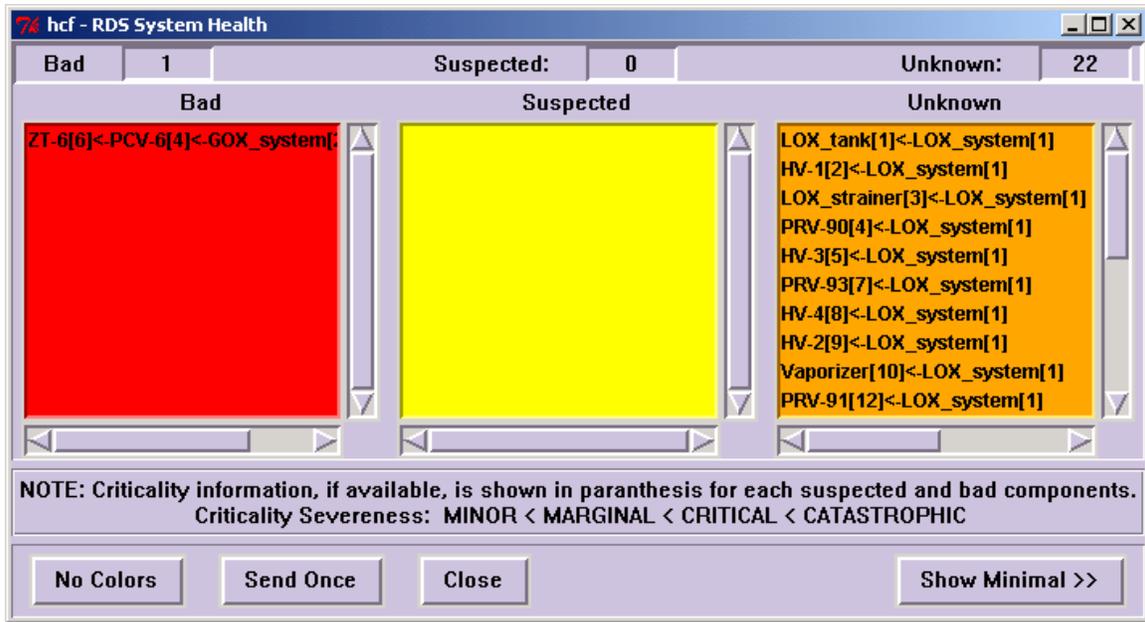


Figure 68: TEAMS-RT tests after ignition command.

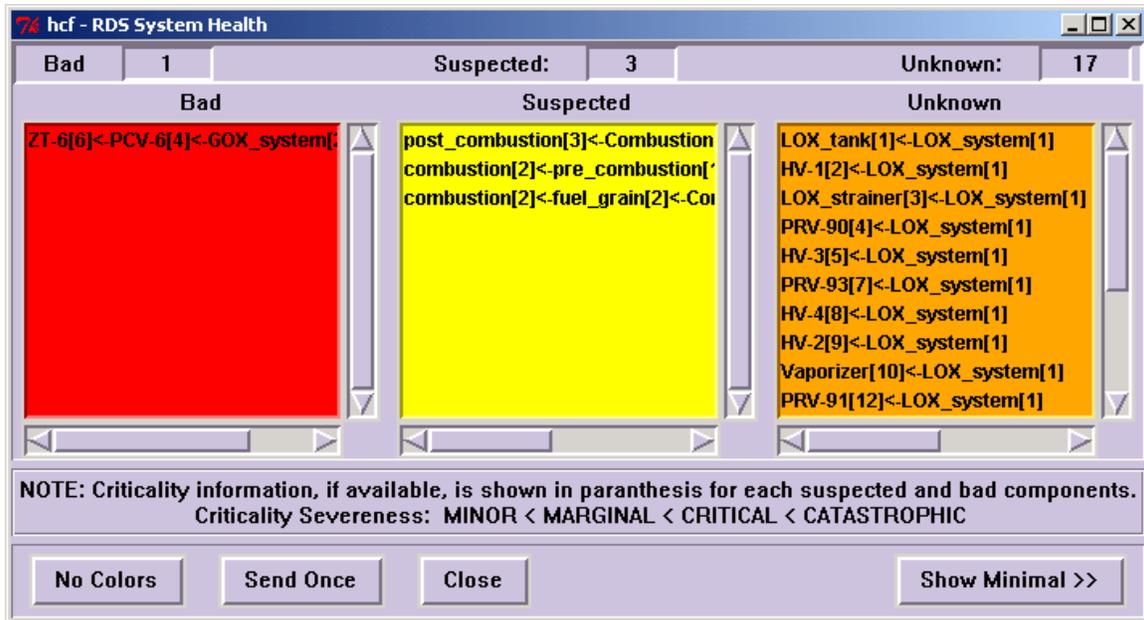


**Figure 69: TEAMS-RT diagnosis after ignition command.**

Finally, after the pressure drops due to the failure in the nozzle, an additional test will fail as shown in Figure 70. Now the diagnosis, Figure 71, includes the three components in the combustion chamber as being suspected of faults. It is not possible to resolve this ambiguity group further with the given sensors. Note that the test for ignition of the fuel grain was assumed to have passed. If the ignition test were coded to check the pressure level only after the ignition command returns to off rather than checking the pressure level after the on command, both PT102\_ignition and PT102\_firing tests would have failed and the suspected failure sources would have been slightly different, including the ignition failure modes of the combustion chamber components and some failure sources in the ignition system.



Figure 70: TEAMS-RT tests after second failure.

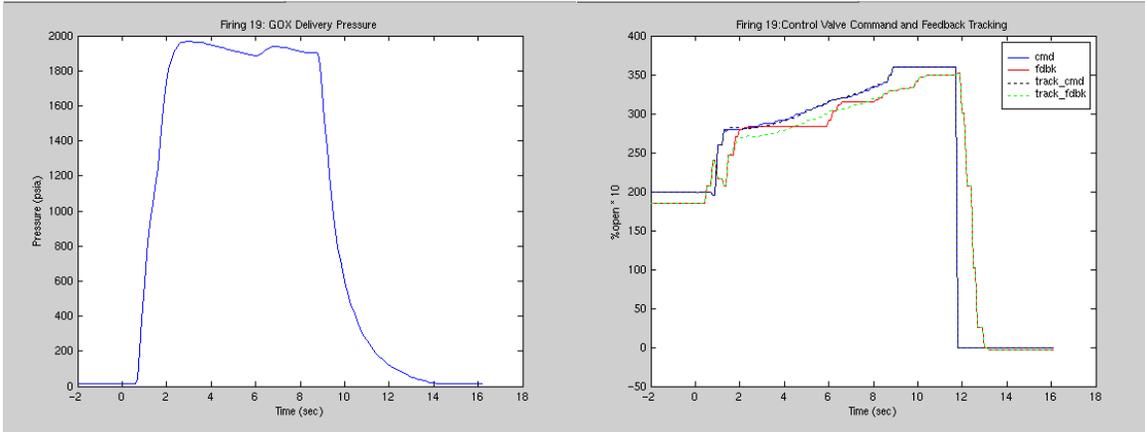


**Figure 71: TEAMS-RT diagnosis after second failure.**

### 10.3 RODON

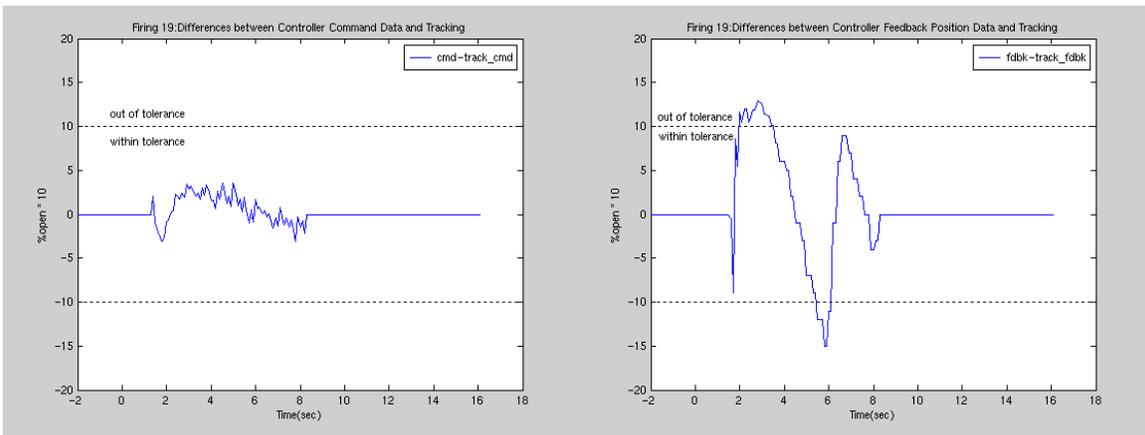
Since the RODON model did not include the combustion chamber, we are unable to provide a diagnosis of firing 2 to compare to the L2 and TEAMS diagnoses. Instead, we'll examine a firing for which there was anomalous behavior in the GOX feed line system. In addition, because the experimental data had very few faults in the feed line, we'll inject some faults using simulated data (from ICS).

HCF firing 19 exhibited some undesirable bumps in delivery pressure as shown in Figure 72a. Examining the control valve position (dotted line) in Figure 72b reveals the reason for this behavior—the control valve did not respond to the commands as expected, most likely due to stiction in the valve. This firing was processed using the RODON monitoring technique and produced a diagnosis of a control valve fault or a control valve position feedback sensor fault when the control valve position tracking exceeded the predefined tolerance for nominal behavior (when the solid line is above or below the dashed lines in Figure 72d). As described in section 7.3.5, the monitoring code calculates expected valve command and position values and compares those to the sensor values. If the expected and actual values differ by more than a predefined tolerance, a flag is sent to the model that results in a failure candidate in one or more of the components. Note that this same approach can be used in TEAMS and L2 since the flag represents a binary test result or variable with two bins, respectively.



a) GOX delivery pressure.

b) Control valve command and position tracking.



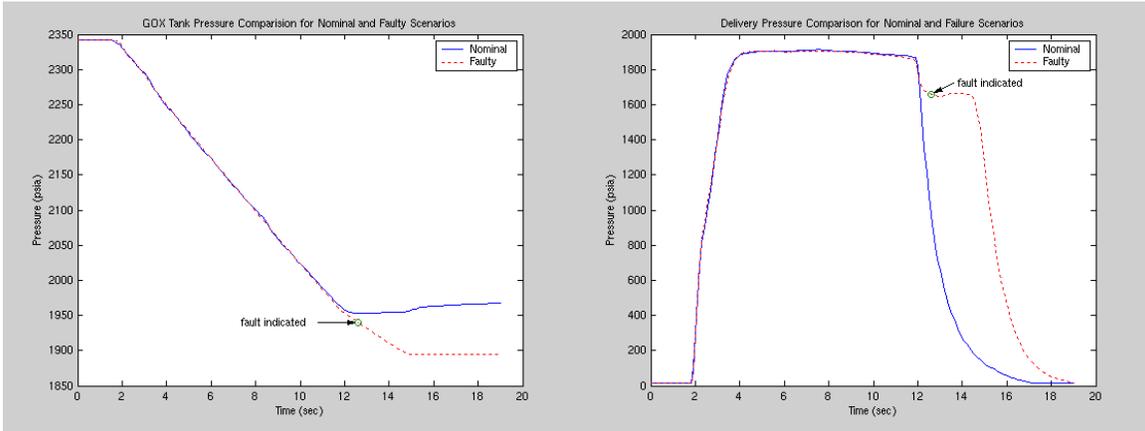
c) Differences between actual and expected control valve command.

d) Differences between actual and expected control valve position.

**Figure 72: An example of RODON monitoring fault detection.**

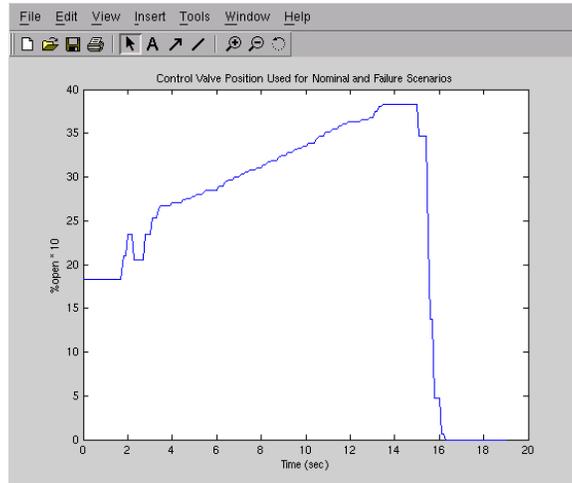
Another diagnostic demonstration was put together using ICS to generate faulty data. Firing 8 was used as a starting point for initial conditions, commands and statuses, and control valve feedback position. To generate the faulty behavior, the command to and feedback from POV-4 were altered. For this case, when valve POV-4 was commanded to close it was closed until reaching 30 degrees and then held at this value to simulate a valve that is stuck partially open. POV-5 was assumed to close normally. The simulated faulty delivery pressure trace, shown as the dashed line in Figure 73b, seems plausible for this scenario. The simulated delivery and tank pressures replaced the corresponding actual data for firing 8 and were processed using the RODON monitoring technique. The time of initial fault indication is shown on the figures. At this time, RODON lists all components that affect the pressure in the GOX line as suspected components since suspending the constraints of any of those components make the system consistent. A short time later, when RODON fails to see the valve feedback indicate close contact, it lists the valve, valve driver, and closed limit switch as suspects.

The additional information has helped to prune the list of suspected components.



a) Nominal and simulated faulty tank pressure.

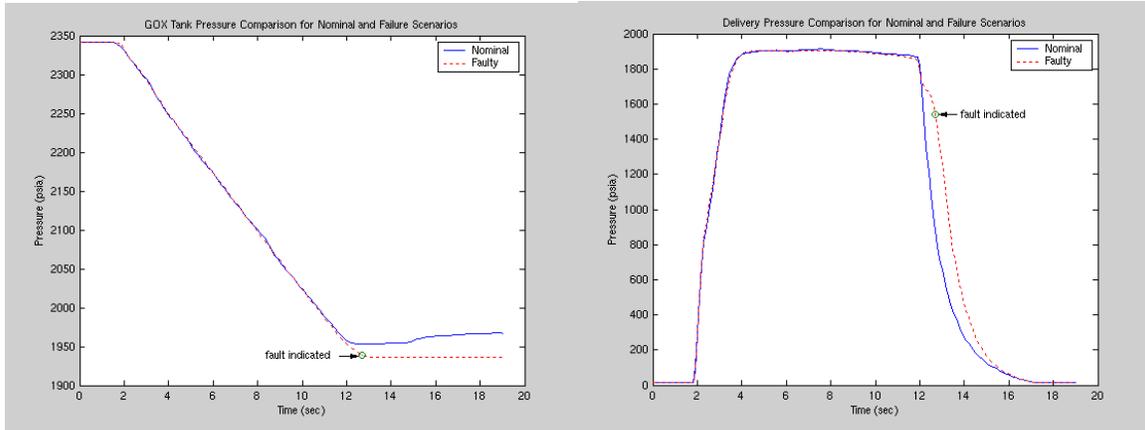
b) Nominal and simulated faulty delivery pressure.



c) Control valve position used in both nominal and failure scenarios.

**Figure 73: Simulated failure scenario with POV-4 stuck partially open after command to close.**

Finally, consider a similar failure where valve POV-4 momentarily sticks partially open before closing fully after being jostled free by the activation of POV-5. This and the previous scenario may be unlikely, but they give us some idea how RODON would deal with similar occurrences. Figure 74 shows that the fault is indicated shortly after the time of fault injection. This could potentially be a difficult failure for L2 or TEAMS to catch since the values do not deviate much from nominal operation.



a) Nominal and simulated faulty tank pressure.

b) Nominal and simulated faulty delivery pressure.

**Figure 74: Simulated transient failure scenario with POV-4 momentarily stuck partially open after command to close.**

#### 10.4 Fault Detection with IMS

The Inductive Monitoring System (IMS) tool was developed to provide a technique to produce health monitoring knowledge bases for systems that are difficult to model or which require models that are too complex to use for real-time monitoring. IMS uses nominal data sets collected directly from the system or from simulations to build a knowledge base that can be used to detect anomalous behavior in the system. IMS ‘learns’ typical system behavior by defining general classes of nominal data based on archived data sets and is able to monitor the system by correlating real-time data with these classes. Although IMS is still under development, we were able to perform some preliminary studies using data collected from the HCF and HCF models developed for this project.

The basic structure used in IMS is the data vector, an ordered set of system data parameters. The data vector is used to define the nominal behavior classes and system monitoring is accomplished by formatting collected data into the defined vector format and comparing the vectors with the nominal classes. The data vector must be defined in a way that captures relevant system behavior. For our HCF study, we used IMS to build a monitoring knowledge base for the gaseous oxygen (GOX) delivery system. The primary function of that system is to provide oxygen at an appropriate rate to fuel the HCF combustion process. A malfunction in the GOX delivery system would result in an unexpected GOX flow rate for a given pressure and valve configuration. To capture the flow rate in our IMS data vector definition we combined sensor readings from two consecutive data samples into a single vector. Each vector has seven members as shown in Figure 75. The first member is the current position (in degrees open) of the shutoff valve; the second is the current control valve position. Next comes the current GOX tank pressure (PIT3), the feed pressure just upstream of the sonic orifice (PT6), and the difference between those two readings. Finally we have the change in the GOX tank

pressure since the last data sample, and the change in the orifice feed pressure since the last data sample. This data vector definition captures the current operating parameters as well as relevant changes from the previous data frame. Incorporating the pressure drop between PIT3 and PT6 provides a normalization parameter that allows IMS to form more general behavior classes that don't depend on specific PIT3 and PT6 pressure values. There may be other data vector definitions that would work for the HCF monitoring task, but we found that this definition provided adequate results for our preliminary studies.

POV-4 position	PCV-6 position	PIT3 pressure	PT6 pressure	PIT3-PT6 difference	PIT3 change	PT6 change
-------------------	-------------------	------------------	-----------------	------------------------	----------------	---------------

**Figure 75: HCF data vector definition used for IMS.**

Several HCF data sets have been collected during test firings at the facility. As noted previously, some of those data sets are missing key data points so only fifteen were complete enough to use for testing the IMS techniques. There were no significant GOX delivery system anomalies recorded in any of these data sets. However, different sonic orifice diameters were used on some of these firings. The orifice diameter has a noticeable effect on the system, so one could consider system performance with a smaller orifice as 'anomalous' when compared to performance expected with a larger orifice. This was the basis of our IMS testing. IMS was trained on data from the test firings that used a larger orifice size. This resulted in a monitoring knowledge base characterizing 'nominal' HCF operation that only included information from the large orifice firings. When this knowledge base is used to analyze data from firings using a smaller orifice, the system behavior should be reported as suspect.

IMS was trained using nine 'nominal' data sets collected during large orifice HCF firings. The vectors from three of these data sets were used as basis classes, with each vector forming a nominal class containing one member. Three more data sets were used to expand those basis class definitions through interpolation. For most vectors in these sets, the already defined class that was 'closest' to a vector was expanded to include the vector and any data values between the vector and the class. If a training vector was too far away from any known nominal classes, a new nominal class was formed containing that vector. The class closest to a vector is defined as the class that would require the least amount of expansion to incorporate that vector. Vector distance or required expansion is measured as the sum of the percent changes in each vector parameter value required to include the new vector. For instance, if the training vector contained the values {85 23 2027 1202 825 41 6} and a class was found that contained the vector {85 22 2030 1200 830 44 8}, the difference between those two vectors would be {0 1 3 2 5 3 2}. The difference for each parameter would be divided by the range of possible values for that parameter to obtain a percent change, then those percentages would be summed to find the distance between the new vector and the closest vector in the class. Once the class with the least required expansion (lowest percentage sum) is identified, the new vector and vectors with parameter values falling between the new vector and the closest vector in the class will be added to the class if it is within a specified maximum distance from the class.

The last training step used three additional data sets to estimate an upper and lower error bound for each data parameter. Once again, IMS identified the class that was ‘closest’ to each new training vector. This time, however, instead of adding the new vector and its neighbors to the class, IMS adjusted a global weight value to add or subtract from each vector parameter when testing for class membership. The weight values were intended to compensate for inaccuracies in the sensor data and limitations of the training data sets. When a set of weights was found that classified all of the data sets in the third group as ‘nominal’ data the IMS training was complete.

After training on archived HCF data, IMS was tested using two ‘nominal’ (large orifice) data sets and four ‘off-nominal’ (smaller orifice) data sets that were not included in the IMS training data. The data records from each set were processed with the IMS monitoring knowledge base, presented in the same order they were collected from the HCF. IMS correctly identified the ‘off-nominal’ data sets as suspect soon after the shutoff valve was opened. IMS also correctly processed the ‘nominal’ data sets, finding that all data sequences were included in the nominal classes in the knowledge base. If IMS were installed in the facility, the monitoring program could send an alert to the operator, or perhaps initiate a system shut down, as soon as an off nominal situation was detected.

To test the effectiveness of IMS when trained on simulated data, we used the ICS simulator with the previously mentioned HCF model to produce data from 1200 simulated runs. These data sets were divided into three groups, and IMS was trained in the same manner as with the HCF archived data. Since the simulation did not reproduce the noise characteristics of the actual HCF data the knowledge base produced solely with simulated data was not effective in monitoring actual data sets. However, results improved after adding three actual data sets to simulated training sets. Adding the actual data sets allowed IMS to incorporate the data noise characteristics into the knowledge base. When tested on archived HCF data sets that were not used for training, this updated knowledge base provided monitoring results similar to the knowledge base produced with archived HCF data. The ability to train with simulated data enables IMS to produce useful monitoring knowledge bases for systems that don’t have an extensive archive of data available. It also allows IMS to include information about previously unexercised system operating regimes in the monitoring knowledge base by simulating them prior to an actual system run.

In another test, IMS was able to detect a stuck valve failure in a simulated scenario. The ICS simulator was run with the same HCF model used to produce the simulated training data. A failure was injected near the end of the simulation that caused the shutoff valve to stick open at 20 degrees instead of closing all the way. An IMS monitoring knowledge base was built using only simulated nominal data. When the data from the failure simulation was processed by IMS, the valve failure was detected within two data frames (0.2 simulated seconds) of the failure occurrence.

Although the IMS monitoring algorithm has not yet been optimized for speed, initial timing tests were encouraging. A simple linear search was used to match input data with

classes in the monitoring knowledge base formed from the 1200 simulated training data sets. Data records were read sequentially from a disk file. Running on a Sun Microsystems Blade 1000 workstation with a 750 MHz processor, IMS was able to process about 2000 data records per second. On an older 300 MHz Sun Ultra 10 the processing rate was about 700 records per second. These figures indicate that IMS should be able to monitor at kilohertz data rates if the data acquisition interface was able to transfer data quickly enough.

Additional development and more thorough testing will be required before the IMS system is ready for application in the field. However, these preliminary results are encouraging and indicate that the IMS tool could be useful for real-time system fault detection. Since a diagnostic capability has not been developed for IMS yet, another diagnostic tool could be invoked for failure isolation after IMS detected a system anomaly. This would allow the use of a powerful software tool for failure diagnosis even if that tool did not provide real-time monitoring capability.

## 11 Conclusions

Models of the HCF were created using three model-based reasoning tools in order to examine their approaches to fault diagnosis of hybrid rocket technologies. In addition, two fault detection techniques were developed and tested. The models varied from strictly qualitative (TEAMS and L2) to quantitative (RODON). For the qualitative models, the abstraction code needed for an end-to-end implementation of the fault diagnostic system was not developed. Although rigorous timing studies were not performed, the quantitative model takes much longer to process the data and diagnose the system than the qualitative models, even after considering the additional time that would be required by the abstraction code. However, no attempt was made to optimize the quantitative model or to compile out a rule base, which would substantially improve the performance. The quantitative model offers a relatively straightforward way to handle the system transients by using time as a variable directly in the model. In the qualitative models, more thought must be given to what is expected as the firing progresses and how to capture those expectations in the model and tests (or bins). In general, more knowledge of the system operation must be included outside of the model as the model becomes more abstract.

The IMS was able to learn system behavior from experimental data and simulation of nominal runs and demonstrated a fast fault detection capability. It should be useful for real-time fault detection of systems that are difficult to model or as a first pass monitoring tool that can catch problems before passing them on to diagnostic tools for further analysis.

For the HCF, the introduction of model-based reasoning tools would add little value. The current control system provides effective system safing by monitoring the valve states, burst disks, PLC, over and/or under pressures on a couple of pressure measurements, and by having an emergency shutdown button. The HCF operators can do troubleshooting very quickly by looking at the controller alarms and a few pressure measurements, and by

examining the facility after a firing. The HCF is a topologically linear system with few subsystem interactions, which makes the troubleshooting easier. Furthermore, some of the interactions occur for short durations of time that are scheduled. The pre-defined sequence of events, limited subsystem interactions, expert knowledge of the operators, and the ability to perform visual inspections after the firing make the addition of an IVHM system superfluous for fault diagnosis of the HCF.

Once the technologies being tested at the HCF are incorporated into a vehicle, the case for an IVHM system becomes stronger. We expect that there will be many more system interactions that are not scheduled. Fault isolation becomes much harder for a human to do efficiently as the system becomes more complex with many measurements and interactions. In addition, we must rely on on-board sensors for diagnostic information during flight. The IVHM system has to be tightly integrated with a reconfiguration manager or intelligent controller to ensure that the vehicle continues to meet the mission objectives in a degraded state or aborts the mission.

## **12 Suggestions for Further Study**

Not all of the diagnostic information available from a rocket firing has been included in the models. In particular, the plume measurements have not been utilized. Researchers in another division at Ames have developed off-line feature extraction algorithms that have been used to detect anomalies in recorded firings. Current research is aimed at producing fast data reduction algorithms for use in a real-time system. Incorporating plume diagnostics in an IVHM system of a rocket is an important step that should be pursued.

Some of the subsystems were not included in the models and could be added to provide increased fault detection and diagnostic capability. One advantage of model-based tools is the ability to reason about system wide interactions to isolate a failure that might require a lot of effort on the part of a human observer. The benefit of this automated fault diagnosis increases as the complexity of the system increases.

The ICS and IMS tools show promise for fault detection and warrant further development. The potential for IMS to quickly identify anomalies should be exploited for detecting faults in subsystems that are hard to model, like the combustion chamber.

This paper discussed the application of a few different IVHM tools to the same set of components and data. A more relevant and challenging problem is to apply various fault detection and isolation tools and algorithms at the subsystem level and then fuse those results using an IVHM architecture that provides a system level health monitoring capability.

## 13 References

- [1] G. Ziliac, M. A. Karabeyoglu, B. Cantwell, R. Hunt, S. DeZilwa, M. Shoffstal, P. Soderman. Ames Hybrid Combustion Facility. *NASA TM 211864*, February 2003.
- [2] M. A. Karabeyoglu, B. J. Cantwell, D. Altman. Development and Testing of Paraffin-based Hybrid Rocket Fuels. *AIAA/ASME/SAE/ASEE 37th Joint Propulsion Conference and Exhibit*, AIAA Paper No. 2001-4503, July 2001.
- [3] R. Davis, W. Hamscher. Model-based reasoning: Troubleshooting. In *Readings in Model-based Diagnosis*, pp.3-24. Editors: W. Hamscher, L. Console, J. de Kleer. Morgan Kaufmann Publishers, Inc., 1992.
- [4] S. Deb, K. R. Pattipati, V. Raghavan, M. Shakeri, R. Shrestha. Multi-Signal Flow Graphs: A Novel Approach for System Testability Analysis and Fault Diagnosis. In *Proc. IEEE AUTOTESTCON*, Anaheim, CA, pp. 361-373, Sept. 1994.
- [5] B. C. Williams, P. P. Nayak. A Model-based Approach to Reactive Self-Configuring Systems. In *Proceedings of AAAI-96*, pp. 971-978, 1996.
- [6] J. Kurien, P. Nayak. Back to the Future for Consistency-based Trajectory Tracking. In *Proceedings of the 7<sup>th</sup> National Conference on Artificial Intelligence (AAAI'2000)*, pp. 370-377, 2000.
- [7] N. Muscettola, P. Nayak, B. Pell, B. Williams. Remote Agent: To Boldly Go Where No AI System Has Gone Before. *Artificial Intelligence*, 103(1-2), pp. 5-47, 1998.
- [8] C. Meyer, H. Cannon et al. Propulsion IVHM Technology Experiment Overview. In *IEEE Aerospace Conference Proceedings*, March 2003.
- [9] I. E. Idelchik. *Handbook of Hydraulic Resistance*. 3<sup>rd</sup> edition, CRC Press, 1994.
- [10] A. Patterson-Hine, W. Hindson, D. Sanderfer, S. Deb, and C. Domagala. A Model-based Health Monitoring and Diagnostic System for the UH-60 Helicopter. In *Proceedings of the 57<sup>th</sup> Annual Forum of the American Helicopter Society*, May 2001.

# REPORT DOCUMENTATION PAGE

*Form Approved*  
*OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

<b>1. AGENCY USE ONLY (Leave blank)</b>	<b>2. REPORT DATE</b> June 2003	<b>3. REPORT TYPE AND DATES COVERED</b> Technical Memorandum	
<b>4. TITLE AND SUBTITLE</b> System Modeling and Diagnostics for Liquefying-Fuel Hybrid Rockets		<b>5. FUNDING NUMBERS</b>  UPN 721-24-28	
<b>6. AUTHOR(S)</b> Scott Poll, David Iverson, Jeremy Ou, Dwight Sanderfer, Ann Patterson-Hine			
<b>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</b>  Ames Research Center Moffett Field, CA 94035-1000		<b>8. PERFORMING ORGANIZATION REPORT NUMBER</b>	
<b>9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)</b>  National Aeronautics and Space Administration Washington, DC 20546-0001		<b>10. SPONSORING/MONITORING AGENCY REPORT NUMBER</b>  NASA/TM-2003-212270	
<b>11. SUPPLEMENTARY NOTES</b> Point of Contact: Scott Poll, Ames Research Center, MS 269-1, Moffett Field, CA 94035-1000 (650) 604-2143			
<b>12a. DISTRIBUTION/AVAILABILITY STATEMENT</b>  Unclassified — Unlimited Subject Category 66                      Distribution: Standard Availability: NASA CASI (301) 621-0390		<b>12b. DISTRIBUTION CODE</b>	
<b>13. ABSTRACT (Maximum 200 words)</b>  A Hybrid Combustion Facility (HCF) was recently built at NASA Ames Research Center to study the combustion properties of a new fuel formulation that burns approximately three times faster than conventional hybrid fuels. Researchers at Ames working in the area of Integrated Vehicle Health Management recognized a good opportunity to apply IVHM techniques to a candidate technology for next generation launch systems. Five tools were selected to examine various IVHM techniques for the HCF. Three of the tools, TEAMS (Testability Engineering and Maintenance System), L2 (Livingstone2), and RODON, are model-based reasoning (or diagnostic) systems. Two other tools in this study, ICS (Interval Constraint Simulator) and IMS (Inductive Monitoring System) do not attempt to isolate the cause of the failure but may be used for fault detection. Models of varying scope and completeness were created, both qualitative and quantitative. In each of the models, the structure and behavior of the physical system are captured. In the qualitative models, the temporal aspects of the system behavior and the abstraction of sensor data are handled outside of the model and require the development of additional code. In the quantitative model, less extensive processing code is also necessary. Examples of fault diagnoses are given.			
<b>14. SUBJECT TERMS</b> IVHM, Integrated Vehicle Health Management, hybrid combustion, model-based reasoning, fault diagnosis, system modeling		<b>15. NUMBER OF PAGES</b> 134	
		<b>16. PRICE CODE</b>	
<b>17. SECURITY CLASSIFICATION OF REPORT</b> Unclassified	<b>18. SECURITY CLASSIFICATION OF THIS PAGE</b> Unclassified	<b>19. SECURITY CLASSIFICATION OF ABSTRACT</b>	<b>20. LIMITATION OF ABSTRACT</b>