**Spacecraft FuelTank Model Feature Spec and Test Procedures**

Lead: Steve C

Developer: Darrel

STE/GUI Tester: Shawn

[Explore](#h.blttnebuh1yr)

[Known Bugs](#h.fy9l5djhxus3)

[Failing Tests](#h.x5nxs9swj86t)

[Other Findings](#h.255a5j5c5vh3)

[Requirements](#h.hg7ht1kc8jq2)

[Interface/Functional Spec](#h.dkhm82tb6lds)

[Overview](#h.tj492u8tvx1i)

[Description](#h.y647aplqfkxw)

[Fields](#h.s4uk9hw5pbnt)

[GUI](#h.7h4nj4qlhx86)

[Remarks](#h.7vnqcfgs2slc)

[Examples](#h.32fsayctuf6m)

[Test Procedures](#h.afp33sm1hbfq)

[Assumptions](#h.evplg86fl6xx)

[Existing Tests](#h.c0clhzqg8we)

[Recommended Additional Tests](#h.3g0z56b7d67s)

[Appendix A - Miscellaneous Data](#h.23w2fud34m1v)

[STK Tank GUI screen snap:](#h.t44hzik2mkui)

[Maneuver Test Object Definitions](#h.npq1fn9873yf)

[Re: Maneuver Test Object Definitions](#h.c1gia24kfqxi)

[Re: Maneuver Test Object Definitions](#h.r049l1ohwc8z)

[Re: Maneuver Test Object Definitions](#h.v2e6hil142pa)

[[GMT-360] Poor Error Message for Partially configured Finite Burn Command Created: 23/Feb/12 Updated: 27/Mar/12](#h.nr8ua4ud685t)

[[GMT-534] Implement command mode assignments for cloned objects hidden from the user Created: 13/Mar/12 Updated: 02/Apr/12](#h.3rfm7cxwayez)

[Spec User’s Guide](#h.wrlcgpub5u8b)

[FuelTank\_PressureModel\_Cmd failed regression test](#h.l61gm6im6xt4)

[Truth](#h.goikkzd5hcx4)

[GMAT Run (SVN Repository)](#h.cjj69rkvq0l2)

[GMAT (locally run with fuel tank params set in resource tree):](#h.w9r0l6sipm0z)

[Possible Future Fine Tuning of Test Cases (for later builds)](#h.z4jxatb4q6rg)

# Explore

## Known Bugs

Known bugs/issues [committed in JIRA](http://li64-187.members.linode.com:8080/secure/Dashboard.jspa) against this feature:

|  |  |  |
| --- | --- | --- |
| **JIRA ID** | **Summary** | **Comment** |
| [GMT-2000](http://www.google.com/url?q=http%3A%2F%2Fli64-187.members.linode.com%3A8080%2Fbrowse%2FGMT-2000&sa=D&sntz=1&usg=AFQjCNFDNVJFktmKa2OdGeZOGk7J0EnYuA) | The PressureModel field on Spacecraft Tank does not catch all disallowed input. All field settings in the attached script are disallowed. It only catches three of them. | -Can’t find attached script mentioned in JIRA. Steve Hughes didn’t know where to find attachment.  -There currently are 5 validation tests associated with the PressureModel field (Tank\_Validation\_PressureModel\*). The current regression tests needed to be fixed. See “Other Findings” Item 3. All 5 of these tests now pass.  -Will confer with Steve H. to determine if OK to close bug. Update 6/14/12: Steve H. requested that bug be closed.  -See [Appendix](#h.z4jxatb4q6rg) (Possible Future Fine Tuning of Test Cases, item 4) for methods to improve the validation tests for future releases. |
| [GMT-2162](http://www.google.com/url?q=http%3A%2F%2Fli64-187.members.linode.com%3A8080%2Fbrowse%2FGMT-2162&sa=D&sntz=1&usg=AFQjCNGbz7LdCfx1rxtWqTCLUhRyUDa5UA) | [Unexpected change of value in Fuel Mass text box](http://li64-187.members.linode.com:8080/browse/GMT-2162) | -Error condition replicated. Once bug fixed, should be verified by GUI testers.  -Update 6/16/12: The bug was fixed during the writing of this feature spec. Issue now closed. |
| [GMT-534](http://www.google.com/url?q=http%3A%2F%2Fli64-187.members.linode.com%3A8080%2Fbrowse%2FGMT-534&sa=D&sntz=1&usg=AFQjCNG2qamaA9VbhbWgiyEYP3szyx6SDg) | Implement command mode assignments for cloned objects hidden from the user | -Bugs GMT-535-538 are related. Bugs listed here for information purposes. (There are set/get subtleties, related to mission mode vs. resource mode, for this feature.)  -Update 6/14/12: Confirmed Darrel’s comment that these bugs have now been closed. |

Note that [GMT-2587](http://li64-187.members.linode.com:8080/browse/GMT-2587) is the Fuel Tank QA task (not a bug).

## Failing Tests

(Run Identification: [Gmat-buildtest] Test results: 2012-05-09 (Win7-64/GMAT-32/M2010a/VS))

Two tests ran to completion but failed:

Failed Test 1: FuelTank\_AllowNegativeFuelMass\_Cmd (script) [pos err 5.439568 vel err

0.007751 mass err 56.658000]

Comments/Actions Taken:

* This script has the following lines in the mission sequence. (As a side note, GMAT does not allow one to set DefaultSC.FuelTank1.Pressure and related FuelTank parameters in the resource tree)
  + DefaultSC.FuelTank1.Pressure = 1700.3452;
  + DefaultSC.FuelTank1.FuelMass = 699.3425;
  + DefaultSC.FuelTank1.Pressure = 1700.3452;
  + DefaultSC.FuelTank1.Temperature = 17.8435;
  + DefaultSC.FuelTank1.RefTemperature = 15.823;
  + DefaultSC.FuelTank1.Volume = 3.7624;
  + DefaultSC.FuelTank1.FuelDensity = 1000.32;
  + DefaultSC.FuelTank1.AllowNegativeFuelMass = false;
  + DefaultSC.FuelTank1.PressureModel = PressureRegulated;
* Setting of the FuelTank1 parameters in the mission sequence above did not work. Bug [GMT-2686](http://li64-187.members.linode.com:8080/browse/GMT-2686), as mentioned below, was submitted.
* The current script did not set the AllowNegativeFuelMass flag to true .
* If one makes the following (local) changes to the script, the test passes.
  + Comment out all (non real-valued) assignments to the DefaultSC.FuelTank1 object.
  + Set FuelTank1.AllowNegativeFuelMass = true in either the resource or the mission sequence.
  + Change the initial fuel mass value so that, after the propagation is over, the fuel tank mass is negative.
* Updated script, FuelTank\_AllowNegativeFuelMass\_Cmd, in the Jazz repository
  + Set DefaultSC.FuelTank1.AllowNegativeFuelMass = true in the mission tree.
  + Change the initial fuel mass value so that, after the propagation is over, the fuel tank mass is negative.
  + Change the pass/fail criteria to only look at fuel mass. (Note that STK will not allow fuel mass to be negative.)
  + Test should pass once bug [GMT-2686](http://li64-187.members.linode.com:8080/browse/GMT-2686), is fixed

Failed Test 2: FuelTank\_PressureModel\_Cmd (script) [pos err 5.439568 vel err 0.007751 mass err 56.658000]

Actions Taken/Comments:

* No need to change script in the SVN repository.
  + The test failed because of the bug found below, [GMT-2686](http://li64-187.members.linode.com:8080/browse/GMT-2686)).
  + Verification that GMT-2686 caused test to fail: Ran test locally after setting the FuelTank parameters in the resource tree and achieved expected values. See [FuelTank\_PressureModel\_Cmd failed regression test](#h.l61gm6im6xt4).
  + Test should pass once bug GMT-2686 is fixed.

## Other Findings

1. Submitted bug report, [GMT-2686](http://li64-187.members.linode.com:8080/browse/GMT-2686), *Some Fuel Tank assignments do not work in command mode.*
2. Submitted bug report, [GMT-2753](http://li64-187.members.linode.com:8080/browse/GMT-2753), *GMAT data that is "settable" should be "gettable."* Description below:

As a general rule, if you can set a GMAT variable, then one should be able to "get" the variable. (This issue was created as a bug, as opposed to an enhancement, per direction from Steve H.) Examples of commands which should be valid but are not:

(1) Report rf FuelTank1.Pressure

(2) Report rf DefaultSC.FuelTank1.PressureModel

1. Many of the “Validation” regression tests were not working correctly even though the tests passed. The desired error condition was not being correctly verified because of a syntax error when creating the FuelTank object. In the 50 test files below, “Create Tank” was replaced by “Create FuelTank.”

Tank\_Validation\_AllowNegativeFuelMass.script

Tank\_Validation\_AllowNegativeFuelMass\_2.script

Tank\_Validation\_AllowNegativeFuelMass\_3.script

Tank\_Validation\_AllowNegativeFuelMass\_4.script

Tank\_Validation\_AllowNegativeFuelMass\_5.script

Tank\_Validation\_FuelDensity.script

Tank\_Validation\_FuelDensity\_2.script

Tank\_Validation\_FuelDensity\_3.script

Tank\_Validation\_FuelDensity\_4.script

Tank\_Validation\_FuelDensity\_5.script

Tank\_Validation\_FuelDensity\_6.script

Tank\_Validation\_FuelDensity\_7.script

Tank\_Validation\_FuelMass.script

Tank\_Validation\_FuelMass\_2.script

Tank\_Validation\_FuelMass\_3.script

Tank\_Validation\_FuelMass\_4.script

Tank\_Validation\_FuelMass\_5.script

Tank\_Validation\_FuelMass\_6.script

Tank\_Validation\_FuelMass\_7.script

Tank\_Validation\_Pressure.script

Tank\_Validation\_Pressure\_2.script

Tank\_Validation\_Pressure\_3.script

Tank\_Validation\_Pressure\_4.script

Tank\_Validation\_Pressure\_5.script

Tank\_Validation\_Pressure\_6.script

Tank\_Validation\_Pressure\_7.script

Tank\_Validation\_PressureModel.script

Tank\_Validation\_PressureModel\_2.script

Tank\_Validation\_PressureModel\_3.script

Tank\_Validation\_PressureModel\_4.script

Tank\_Validation\_PressureModel\_5.script

Tank\_Validation\_RefTemperature.script

Tank\_Validation\_RefTemperature\_2.script

Tank\_Validation\_RefTemperature\_3.script

Tank\_Validation\_RefTemperature\_4.script

Tank\_Validation\_RefTemperature\_5.script

Tank\_Validation\_RefTemperature\_6.script

Tank\_Validation\_Temperature.script

Tank\_Validation\_Temperature\_2.script

Tank\_Validation\_Temperature\_3.script

Tank\_Validation\_Temperature\_4.script

Tank\_Validation\_Temperature\_5.script

Tank\_Validation\_Temperature\_6.script

Tank\_Validation\_Volume.script

Tank\_Validation\_Volume\_2.script

Tank\_Validation\_Volume\_3.script

Tank\_Validation\_Volume\_4.script

Tank\_Validation\_Volume\_5.script

Tank\_Validation\_Volume\_6.script

Tank\_Validation\_Volume\_7.script

# Requirements

These are working requirements. They are included here for review and convenience purposes. After review, requirements are maintained in the formal SRS located at SourceForge in /trunk/doc/SystemDocs/Requirements.

|  |  |
| --- | --- |
| **ID** | **Requirements** |
| FRR-8.1 | The system shall allow the user to create and configure a spacecraft tank object. |
| FRR-8.2.0 | The tank model shall allow the user to set the following properties: |
| FRR-8.2.1 | 1) Initial fuel mass |
| FRR-8.2.2 | 2) Initial fuel pressure |
| FRR-8.2.3 | 3) fuel temperature |
| FRR-8.2.4 | 4) fuel density |
| FRR-8.2.5 | 5) Fuel reference temperature |
| FRR-8.2.6 | 6) Tank volume |
| FRR-8.3.0 | The tank model shall support the following depletion modes: |
| FRR-8.3.1 | 1) Pressure regulated |
| FRR-8.3.2 | 2) Blow down using Boyle’s law |
| FRR-8.3.3 | The tank model shall optionally allow the mass of a fuel tank to be negative. |

# Interface/Functional Spec

## Overview

FuelTank is a model of a chemical fuel tank.

## Description

A FuelTank is a thermodynamic model of a tank and is required for finite burn modeling or for impulsive burns that use mass depletion. The thermodynamic properties of the tank are modeled using Boyle’s law and assume that there is no temperature change in the tank as fuel is depleted. To use a FuelTank, you must first create the tank, and then attach it to the desired spacecraft and associate it with a thruster as shown in the example below.

When working in the script, you must add tanks to spacecraft before the begin mission sequence

command.

## Fields

See the [User Interface Spec](https://docs.google.com/spreadsheet/ccc?key=0Am0YJMLTVzxedGxmYmtnVHVvbG45eWd5aWFkbTY2a1E&hl=en_US#gid=0) spreadsheet for reference information for fields.

## GUI

|  |
| --- |
| For GUI Testers and Developers.  The GUI layout is static. All fields are always active. From the perspective of the GUI user, there are no obvious coupled fields. However, “under the hood,” there is a coupling between fuel mass, density, and tank volume such that the fuel volume cannot exceed the tank volume. |

### Introduction

The FuelTank dialog box allows you to specify properties of a fuel tank including fuel mass, density, and temperature as well as tank pressure and volume. The layout of the FuelTank dialog box is shown below.

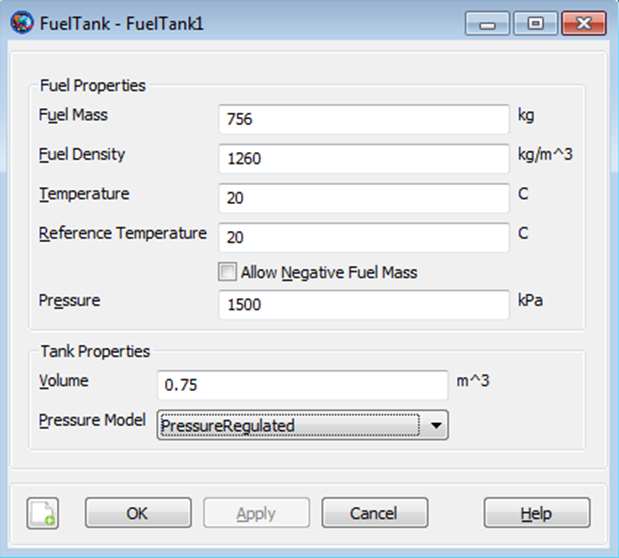
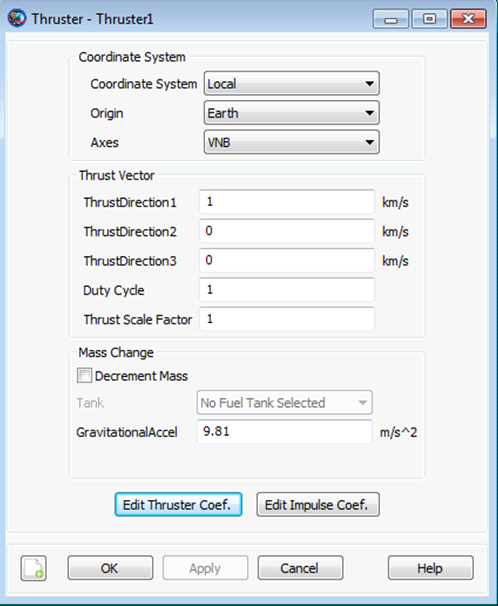


Figure 1 FuelTank Properties

The Thruster resource is closely related to the Fuel Tank resource and thus, we also discuss it here. The Thruster dialog box allows you to specify properties of a thruster including the coordinate system of the Thrust acceleration direction vector, the thrust magnitude and Isp. The layout of the Thruster dialog box is shown below.



When performing a finite burn, you will typically want to model fuel depletion. To do this, select the Decrement Mass button and then select the previously created Fuel Tank as shown below.

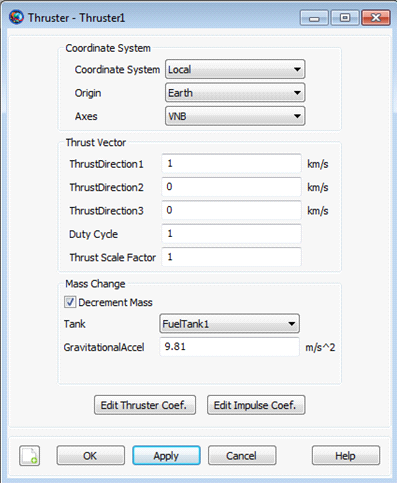


Figure 2 Thruster Properties

Thus far, we have created both a Fuel Tank and a Thruster, and we have associated a Fuel Tank with our Thruster. We are not done yet. We must tell GMAT that we want to attach both the Fuel Tank and the Thruster to a particular spacecraft. To do this, double click on the desired spacecraft under the Spacecraft resource to bring up the associated GUI panel. Then click on the *Tanks* tab to bring up the following GUI display.

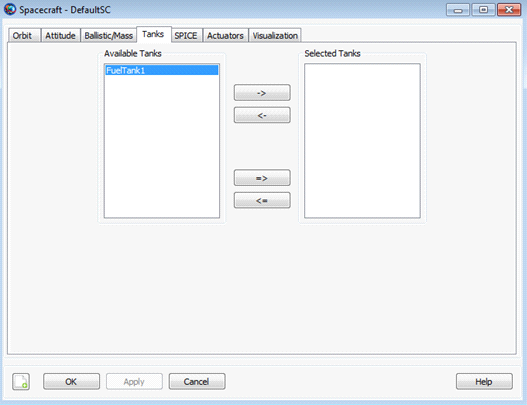


Figure 3 Spacecraft Tanks Properties

Next, select the desired Fuel Tank and use the right arrow button to attach the Fuel Tank to the spacecraft. Then, click the *Apply* button as shown below.

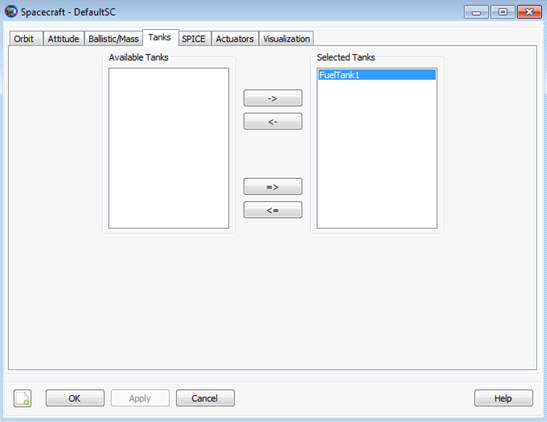


Figure 4 Spacecraft - Select Tank

Similarly, to attach a Thruster to a spacecraft, double click on the desired spacecraft under the Spacecraft resource and then select the *Actuators* tab. Then select the desired thruster and use the right arrow to attach the thruster to the spacecraft. Finally, click the Apply button as shown below.

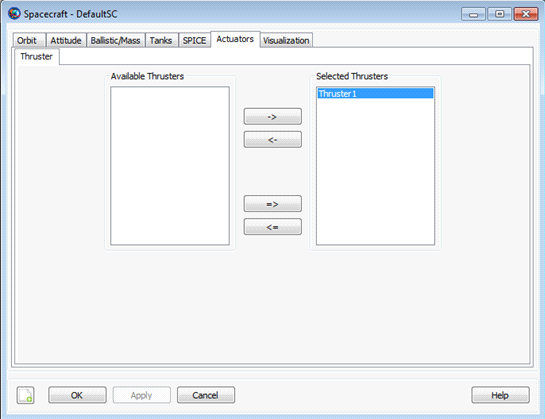


Figure 5 Spacecraft - Select Thruster

## Remarks

**Behavior When Configuring Tank and Attached Tank Properties**

Create a default FuelTank and attach it to a Spacecraft and Thruster.

|  |
| --- |
| % Create the Fuel Tank Object  Create FuelTank aTank;  aTank.AllowNegativeFuelMass = false;  aTank.FuelMass = 756;  aTank.Pressure = 1500;  aTank.Temperature = 20;  aTank.RefTemperature = 20;  aTank.Volume = 0.75;  aTank.FuelDensity = 1260;  aTank.PressureModel = PressureRegulated;  % Create a Thruster and assign it a FuelTank  Create Thruster aThruster;  aThruster.Tank = {aTank};    % Add the FuelTank and Thruster to a Spacecraft  Create Spacecraft aSpacecraft  aSpacecraft.Tanks = {aTank};  aSpacecraft.Thrusters = {aThruster}; |

As exhibited below, there are some subtleties associated with setting and getting parent vs. cloned objects. In the example above, aTank is the parent ‘FuelTank’ object and the field aSpacecraft.Tanks is populated with a cloned copy of aTank.

Create a second spacecraft and attach a fuel tank using the same procedure used in the previous example. Set the FuelMass in the parent object, aTank, to 900 kg. .

|  |
| --- |
| % Add the FuelTank and Thruster to a second Spacecraft  Create Spacecraft bSpacecraft  bSpacecraft.Tanks = {aTank};  bSpacecraft.Thrusters = {aThruster};  aTank.FuelMass = 900; %Can be performed in both resource and command modes |

Note that, in the example above, setting the value of the parent object, aTank, changes the fuel mass value in both cloned fuel tank objects. More specifically, the value of both aSpacecraft.aTank.FuelMass and bSpacecraft.aTank.FuelMass are both now equal to the new value of 900 kg. We note that the assignment command for the parent object, aTank.FuelMass, can be performed in both resource and command modes.

To change the value of the fuel mass in only the first created spacecraft, aSpacecraft, we do the following.

|  |
| --- |
| % Create the Fuel Tank Object  aTank.FuelMass = 756; %Fuel tank mass in both spacecraft set back to default  aSpacecraft.aTank.FuelMass = 1000; %Can only be performed in command mode. |

As a result of the commands in the previous example, the value of aSpacecraft.aTank.FuelMass is 1000 kg and the value of bSpacecraft.aTank.FuelMass is 756 kg. We note that the assignment command for the cloned object, aSpacecraft.aTank.FuelMass, can only be performed in command mode.

**Caution: Value of AllowNegativeFuelMass flag can affect iterative processes**

By default, GMAT will not allow the fuel mass to be negative. However, occasionally in iterative processes such as targeting, a solver will try values of a maneuver parameter that result in total fuel depletion. Using the default tank settings, this will throw an exception stopping the run unless you set the AllowNegativeFuelMass flag to true. GMAT will not allow the the total spacecraft mass to be negative. If DryMass + FuelMass is negative GMAT will throw an exception and stop.

## Examples

Create a default FuelTank and attach it to a Spacecraft and Thruster.

|  |
| --- |
| % Create the Fuel Tank Object  Create FuelTank aTank;  aTank.AllowNegativeFuelMass = false;  aTank.FuelMass = 756;  aTank.Pressure = 1500;  aTank.Temperature = 20;  aTank.RefTemperature = 20;  aTank.Volume = 0.75;  aTank.FuelDensity = 1260;  aTank.PressureModel = PressureRegulated;    % Create a Thruster and assign it a FuelTank  Create Thruster aThruster;  aThruster.Tank = {aTank};    % Add the FuelTank and Thruster to a Spacecraft  Create Spacecraft aSpacecraft  aSpacecraft.Tanks = {aTank};  aSpacecraft.Thrusters = {aThruster};  BeginMissionSequence |

# Test Procedures

## Assumptions

None

## Existing Tests

There are 89 \*.tc files, 91 \*.script files, and 79 \*.truth files. The truth files can be re-used but it would seem that number of TC and SCRIPT files should agree. There are probably 2 \*.script files that are not used. (May warrant further investigation at a later date.)

Scripts FuelTank\_IBurn\_Earth\_ScA\_IBA\_CS0\_TankJ\_cmd\_Tank\_FuelMass.script and FuelTank\_IBurn\_Earth\_ScA\_IBA\_CS0\_TankJ\_Tank\_FuelMass\_Cmd.script are essentially the same script. It is possible that one of these scripts is not used.

The 89 test cases, corresponding to the 89 \*.tc, files are described below.

|  |  |
| --- | --- |
| Test Name | Description |
| FuelTank\_AllowNegativeFuelMass\_Cmd | Verifies that during a finite maneuver (pressure-regulated tank), if the AllowNegativeFuelMass flag is set in command mode (via the cloned DefaultSC.FuelTank1 object), the fuel mass will go negative. |
| FuelTank\_PressureRegulated\_Earth | Verifies propagation results and mass used for a finite maneuver using a pressure regulated tank. |
| FuelTank\_PressureModel\_Cmd | Verifies propagation results and mass used for a finite maneuver using a pressure regulated tank. The PressureModel (through the cloned DefaultSC.FuelTank1 object) is set in command mode. |
| FuelTank\_SetPhysicalProperties\_Cmd | Verifies propagation results and mass used for a finite maneuver using a pressure regulated tank. The 6 real-valued FuelTank parameters are set (through the cloned DefaultSC.FuelTank1 object) in command mode. |
| FuelTank\_IBurn\_Earth\_ScA\_IBA\_CS0\_TankX | Verifies final position, velocity, and mass after an impulsive burn. (There are 22 tests of this type)   * The Impulsive burn components are defined in a system with Earth as origin. * ScA refers to Spacecraft “A” * IBA refers to Impulsive Burn “A” * CS0 refers to Coordinate System “0” * The “X” in TankX can take values A-V. Tanks A-K are pressure-regulated and Tanks L-V Blowdown tanks. |
| FuelTank\_IBurn\_Earth\_ScA\_IBA\_CS0\_TankP\_Tank\_X\_Cmd | Verifies final position, velocity, and mass after an impulsive burn. In each test, one of the fuel tank parameters is set in command mode. (7 tests of this type)   * The “X” in Tank\_X\_Cmd refers to a settable FuelTank parameter such as volume. |
| \*Validation\* | Validates Fuel tank parameter inputs. (56 tests) |

See the [forum entry by JTichy](http://gmat.ed-pages.com/forum/viewtopic.php?f=16&t=292) for additional information on these test definitions.

## Recommended Additional Tests

There are no additional tests recommended at this time.

Nominal Tests

|  |  |  |
| --- | --- | --- |
| Priority | Status | Summary |
| P1 |  | Create two tanks and set all tank properties on Tank 1 to non-default values in initialization. In the mission sequence, set Tank2 = Tank1 and use the tank in a finite and impulsive maneuver and check numeric results |
|  |  |  |
|  |  |  |

Edge/Corner/Stress

|  |  |  |
| --- | --- | --- |
| Priority | Status | Summary |
|  |  |  |
|  |  |  |
|  |  |  |

Unique Validation

|  |  |  |
| --- | --- | --- |
| Priority | Status | Summary |
|  |  |  |
|  |  |  |
|  |  |  |

Unique Mode Tests

|  |  |  |
| --- | --- | --- |
| Priority | Status | Summary |
|  |  |  |
|  |  |  |
|  |  |  |

Unique GUI Tests

These are tests that are unique to the GUI interface for this feature that are not covered by the standard GUI test template and procedures.

|  |  |  |
| --- | --- | --- |
| Priority | Status | Summary |
|  |  |  |
|  |  |  |
|  |  |  |

# Appendix A - Miscellaneous Data

Note that this data is primarily for the internal use of the feature lead. It does not need to be reviewed by the SDE and STE.

## STK Tank GUI screen snap:

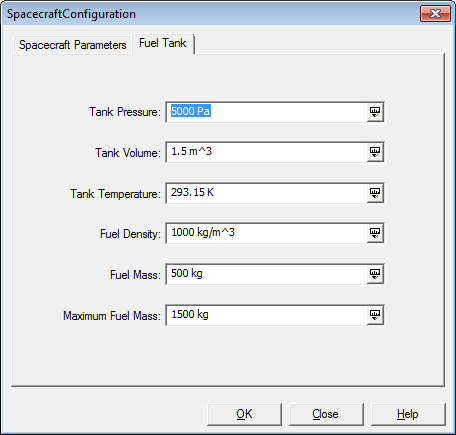


Figure 6 Spacecraft Fuel Tank Configuration

## [Maneuver Test Object Definitions](http://gmat.ed-pages.com/forum/viewtopic.php?f=16&t=292&sid=e490f03003223b74f99b536821ae825b)

(Note: This is a cut and paste from [forum entry by JTichy](http://gmat.ed-pages.com/forum/viewtopic.php?f=16&t=292))

Spacecraft Definitions

ScA is the Baseline spacecraft with the following definition

Sc.CentralBody = Planet;

'//State information';

'Sc.Epoch = 21545.000000000;'

'//Physical Information';

'Sc.VehicleDryMass = 850;'

'Sc.Cd = 2.2;'

'Sc.Cr = 1.8;'

'Sc.DragArea = 15;'

'Sc.SRPArea = 1;'

'//Attitude';

'Sc.AttitudeRefFrame = ''MJ2000Eq'';'

'Sc.Q1 = 0;'

'Sc.Q2 = 0;'

'Sc.Q3 = 0;'

'Sc.Q4 = 1;'

ScB has drymass increased to 1000kg

ScC has a lower Cd value and should not change the script in any way because drag is not modeled

ScD has a lower Cr and should also not affect the output of the scripts due to not being modeled

ScE has a higher drag area and should not affect the output due to not being modeled

ScF has Sc.Q1 as the main direction of the attitude

ScG has Sc.Q2

ScH has Sc.Q3

ScI has higher SRP area and should never affect the outcome of the results

[JTichy](http://gmat.ed-pages.com/forum/memberlist.php?mode=viewprofile&u=762)

**Posts:** 26

**Joined:** Tue May 12, 2009 5:02 pm

[Top](http://gmat.ed-pages.com/forum/viewtopic.php?f=16&t=292&sid=e490f03003223b74f99b536821ae825b#wrap)

### [Re: Maneuver Test Object Definitions](http://gmat.ed-pages.com/forum/viewtopic.php?f=16&t=292&sid=e490f03003223b74f99b536821ae825b#p820)

by [**JTichy**](http://gmat.ed-pages.com/forum/memberlist.php?mode=viewprofile&u=762) » Thu Oct 08, 2009 3:38 pm

Tank Definitions

TankA being the baseline tank with the following values

'// Tank A Baseline Tank';

'TankA.TankMass = 725;'

'TankA.TankPressure = 1200;'

'TankA.TankTemperature = 20;'

'TankA.TankRefTemperature= 12;'

'TankA.TankVolume = 0.8;'

'TankA.TankFuelDensity = 1029;'

'TankA.TankPressureControl = 1;'

TankB has the tank filled at Maximum Capacity for the given parameters where Tank Mass = 820

TankC has ultra-high pressure of 2500

TankD has lower pressure of 725

TankE has extremely high temp of 200

TankF has a low temperature of 2

TankG has a high ref temp of 100

TankH has a low ref temp of 2

TankI has a large tank Volume = 80

TankJ has a low fuel density and altered tank volume to compensate

TankK has extremely high fuel density 2500

L - V are same as above except with Blowdown modeled for pressure regulation

[JTichy](http://gmat.ed-pages.com/forum/memberlist.php?mode=viewprofile&u=762)

**Posts:** 26

**Joined:** Tue May 12, 2009 5:02 pm

[Top](http://gmat.ed-pages.com/forum/viewtopic.php?f=16&t=292&sid=e490f03003223b74f99b536821ae825b#wrap)

### [Re: Maneuver Test Object Definitions](http://gmat.ed-pages.com/forum/viewtopic.php?f=16&t=292&sid=e490f03003223b74f99b536821ae825b#p824)

by [**JTichy**](http://gmat.ed-pages.com/forum/memberlist.php?mode=viewprofile&u=762) » Tue Oct 13, 2009 9:30 am

Thruster Definitions

Thruster A is the chosen baseline thruster and has the following definition

ThrusterA.CoordinateSystem = ',CoordSys,'

ThrusterA.Origin = ',Planet,'

ThrusterA.Axes = 'AxesCode'

ThrusterA.Element1 = 1

ThrusterA.Element2 = 0

ThrusterA.Element3 = 0

ThrusterA.DutyCycle = 1

ThrusterA.ThrustScaleFactor = 1

ThrusterA.DecrementMass = true

ThrusterA.Tank = {'tanksin'}

ThrusterA.GravitationalAccel = 9.81

ThrusterA.C1 = 10

ThrusterA.C2 = 0.25

ThrusterA.C3 = 0.25

ThrusterA.C4 = 0

ThrusterA.C5 = 0

ThrusterA.C6 = 0

ThrusterA.C7 = 0

ThrusterA.C8 = 0

ThrusterA.C9 = 0

ThrusterA.C10 = 0

ThrusterA.C11 = 0

ThrusterA.C12 = 0

ThrusterA.C13 = 0

ThrusterA.C14 = 0

ThrusterA.C15 = 0

ThrusterA.C16 = 0

ThrusterA.K1 = 300

ThrusterA.K2 = 0.25

ThrusterA.K3 = 0.25

ThrusterA.K4 = 0

ThrusterA.K5 = 0

ThrusterA.K6 = 0

ThrusterA.K7 = 0

ThrusterA.K8 = 0

ThrusterA.K9 = 0

ThrusterA.K10 = 0

ThrusterA.K11 = 0

ThrusterA.K12 = 0

ThrusterA.K13 = 0

ThrusterA.K14 = 0

ThrusterA.K15 = 0

ThrusterA.K16 = 0

Thruster B has its direction equally spaced in all directions of the coordinate system where X = Y = Z = 0.5774

Thruster C has a low duty cycle of 0.1

Thruster D has a low thrust scale factor of 0.1

Thruster E has an alternate g\_accel of 12.14

Thruster F has a loaded thruster polynomial which evaluates to 10 Newtons when coupled with Tank A

Thruster G has a loaded Isp polynomial which evaluates to 300 seconds when coupled with Tank A

Thruster H has decrement mass turned off

[JTichy](http://gmat.ed-pages.com/forum/memberlist.php?mode=viewprofile&u=762)

**Posts:** 26

**Joined:** Tue May 12, 2009 5:02 pm

[Top](http://gmat.ed-pages.com/forum/viewtopic.php?f=16&t=292&sid=e490f03003223b74f99b536821ae825b#wrap)

### [Re: Maneuver Test Object Definitions](http://gmat.ed-pages.com/forum/viewtopic.php?f=16&t=292&sid=e490f03003223b74f99b536821ae825b#p825)

by [**JTichy**](http://gmat.ed-pages.com/forum/memberlist.php?mode=viewprofile&u=762) » Tue Oct 13, 2009 9:35 am

Coordinate Systems

CS0 is Mean of J2000 Equatorial

CS1 is VNB

CS2 is LVLH where X\_GMAT = Z\_FreeFlyer; Y\_GMAT = Y\_FreeFlyer; Z\_GMAT = -X\_FreeFlyer

CS3 is Spacecraft Body Coordinate System

CS4 is a custom defined VNB frame and compared against FreeFlyers regular VNB for accuracy (CS1 and CS4 scripts should produce the same output)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

|  |  |
| --- | --- |
| [GMT-360] [Poor Error Message for Partially configured Finite Burn Command](http://li64-187.members.linode.com:8080/browse/GMT-360) Created: 23/Feb/12 Updated: 27/Mar/12 |  |
| **Status:** | Open |
| **Project:** | [GMAT](http://li64-187.members.linode.com:8080/secure/BrowseProject.jspa?id=10000) |
| **Component/s:** | None |
| **Affects Version/s:** | None |
| **Fix Version/s:** | [2013a](http://li64-187.members.linode.com:8080/secure/IssueNavigator.jspa?reset=true&mode=hide&sorter/order=ASC&sorter/field=priority&pid=10000&fixfor=10201) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Type:** | Bug | **Priority:** | P3 |
| **Reporter:** | [Steven Hughes](http://li64-187.members.linode.com:8080/secure/ViewProfile.jspa?name=shughes) | **Assignee:** | [Darrel Conway](http://li64-187.members.linode.com:8080/secure/ViewProfile.jspa?name=djcinsb) |
| **Resolution:** | Unresolved | **Votes:** | 0 |
| **Labels:** | None |  |  |
| **Remaining Estimate:** | Not Specified |  |  |
| **Time Spent:** | Not Specified |  |  |
| **Original Estimate:** | Not Specified |  |  |

|  |
| --- |
|  |

|  |  |
| --- | --- |
| **Description** |  |

|  |
| --- |
| If you build and run this script:  % Create objects  Create Spacecraft aSat  Create FiniteBurn aFiniteBurn  Create Propagator aPropagator  BeginMissionSequence  % Fire thruster for 2 minutes  BeginFiniteBurn aFiniteBurn(aSat)  Propagate aPropagator(aSat,{aSat.ElapsedSecs = 120})  EndFiniteBurn aFiniteBurn(aSat)  you get this error message:  "Propagator Exception: MassFlow is not a known propagation parameter on aSat"  We agreed in the finite burn spec review to change to this:  "Propagator Exception: You have attempted to perform a finite burn for spacecraft InsertSatName without configuring at least one of the following objects: Spacecraft, Tank, Thruster, FiniteBurn. |

|  |  |
| --- | --- |
| [GMT-534] Implement command mode assignments for cloned objects hidden from the user Created: 13/Mar/12 Updated: 02/Apr/12 |  |
| **Status:** | Open |
| **Project:** | [GMAT](http://li64-187.members.linode.com:8080/secure/BrowseProject.jspa?id=10000) |
| **Component/s:** | None |
| **Affects Version/s:** | [2012a](http://li64-187.members.linode.com:8080/secure/IssueNavigator.jspa?reset=true&mode=hide&sorter/order=ASC&sorter/field=priority&pid=10000&version=10000) |
| **Fix Version/s:** | [2012a](http://li64-187.members.linode.com:8080/secure/IssueNavigator.jspa?reset=true&mode=hide&sorter/order=ASC&sorter/field=priority&pid=10000&fixfor=10000) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Type:** | Bug | **Priority:** | P1 |
| **Reporter:** | [Darrel Conway](http://li64-187.members.linode.com:8080/secure/ViewProfile.jspa?name=djcinsb) | **Assignee:** | [Darrel Conway](http://li64-187.members.linode.com:8080/secure/ViewProfile.jspa?name=djcinsb) |
| **Resolution:** | Unresolved | **Votes:** | 0 |
| **Labels:** | None |  |  |
| **Σ Remaining Estimate:** | 2 weeks, 4 hours | **Remaining Estimate:** | Not Specified |
| **Σ Time Spent:** | 1 week | **Time Spent:** | Not Specified |
| **Σ Original Estimate:** | 2 weeks, 3 days, 4 hours | **Original Estimate:** | Not Specified |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Key** | **Summary** | | **Type** | **Status** | **Assignee** |
| [GMT-535](http://li64-187.members.linode.com:8080/browse/GMT-535) | [Design cloned object updates](http://li64-187.members.linode.com:8080/browse/GMT-535) | | Sub-task | Open | Darrel Conway |
| [GMT-536](http://li64-187.members.linode.com:8080/browse/GMT-536) | [Obtain cloned object update design ap...](http://li64-187.members.linode.com:8080/browse/GMT-536) | | Sub-task | Open | Darrel Conway |
| [GMT-537](http://li64-187.members.linode.com:8080/browse/GMT-537) | [Code the cloned object update](http://li64-187.members.linode.com:8080/browse/GMT-537) | | Sub-task | Closed | Darrel Conway |
| [GMT-538](http://li64-187.members.linode.com:8080/browse/GMT-538) | [Test and debug the cloned object upda...](http://li64-187.members.linode.com:8080/browse/GMT-538) | | Sub-task | Open | Darrel Conway |
| **Attachments:** | | FuncTest.gmf RunFuncTests.script | | | | |
| **Sub-Tasks:** | |  | | | | |

|  |  |
| --- | --- |
| **Description** |  |

|  |
| --- |
| Users can script settings for some objects in the Mission Control Sequence that do not affect the objects used in the run. The current known set of such objects is the PropSetup members (ForceModel, Integrator, and other Propagators) and Hardware attached to a Spacecraft (specifically Tanks and Thrusters that we want to change globally). This work is broken into 4 pieces:  1. Design cloned object updates  2. Obtain cloned object update design approval  3. Code the cloned object update  4. Test and debug the cloned object update code  This issue is a roll-up for those 4 items. |

|  |  |
| --- | --- |
| **Comments** |  |

|  |
| --- |
| Comment by [Darrel Conway](http://li64-187.members.linode.com:8080/secure/ViewProfile.jspa?name=djcinsb) [ 13/Mar/12 9:44 PM ] |
| I added watchers listing the folks that will need to review the design, so as not to take them completely unaware. |
| Comment by [Steven Hughes](http://li64-187.members.linode.com:8080/secure/ViewProfile.jspa?name=shughes) [ 14/Mar/12 5:01 PM ] |
| There are at least two tall poles for this task/bug.  1) The design must capture the complexity of owned-object/cloned-ojbect relationships. I put a start to this list below that is quite scary.  2) The design needs to capture when object reinitialization is required if a field is changed. For example, if an "mode" type field is changed such as switching integrator type, or changing an ouput file.  Here is the start to the list of owned object relationships. Given the shear number of them, the design will need to be very general.  ==== Object Coupling  Spacecraft  ---Tank  ---Thruster  Formation  ---Spacecraft  Libaration Point  ---Barycenter  ---Celestial body  Barycenter  ---Celestial body  Orbit View  ---Spacecraft  ---Celestial body  ---Libration point  ---Barycenter  XYPlot  ---Variable  ---Array  ReportFile  ---Variable  ---Array  ---Spacecraft  ---Maneuver  ---Tank  Ephemeris File  ---Spacecraft  Coordinate System  Spacecraft  Celestial body  Barycenter  Libration point  Ground station  —  =====Command Coupling  Optimize  ---vf13ad  ---fmincon  Vary  ---Spacecraft  ---Maneuver  ---Variable  ---Array  Target  ---Differential Corrector  Control Flow  ---Spacecraft  ---Variable  ---Array  Propagate  ---Prop setup  ---Spacecraft |
| Comment by [Steven Hughes](http://li64-187.members.linode.com:8080/secure/ViewProfile.jspa?name=shughes) [ 14/Mar/12 5:15 PM ] |
| I spoke with Linda and she explained why we don't see this issue inside of GMAT functions. It is because Prop Setups are global in functions as specified in the requirements. So changes made to the "core" prop setup are automatically used by all Propagate commands. |
| Comment by [Steven Hughes](http://li64-187.members.linode.com:8080/secure/ViewProfile.jspa?name=shughes) [ 14/Mar/12 6:18 PM ] |
| These attachments contain a GMAT function that configures a Propagator inside of a function and the function produces the correct output compared to truth data (and script version). If I move propagator configuration to below BeginMissionSequence in the script file (i.e non-function version), GMAT does not execute correctly. Since everthing in Functions executes in command mode, it is interesting that the function case works but the command mode script doesn't. |
| Comment by [Darrel Conway](http://li64-187.members.linode.com:8080/secure/ViewProfile.jspa?name=djcinsb) [ 14/Mar/12 9:07 PM ] |
| Something to keep in mind here is that object to object references need not all be impacted by the design for this piece. Most of those object references are not made through cloning. We only clone in a few specific instances where a copy of a single core object needs to be locally tailored for use without impacting other references to the object.  As an example, the Spacecraft in a Formation are not cloned into the formation. They are just referenced, and the object at Sandbox scope is the same thing as is used for Formation manipulations. Basically, we have 3 different types of objects we need to think about here:  (1) Referenced objects - Objects that are accessed by pointer to the Global or Local Object Store,  (2) Owned objects - Objects that are entirely encompassed by another object, and  (3) Cloned objects - Objects that have a pristine version in an object stores, but that are used in the command or other object by creating a clone.  Cases 1 and 2 are handled in the current code; case 3 is what needs a design and implementation. I'm a bit off track on this comment now, though, so I'll discuss this piece a bit further in my next comment.  By far the most used example of the cloned object case is the PropSetup/ForceModel amalgamation, which is already a pretty tricky beast. Initialization of the ODEModel (the class formerly known as ForceModel) is performed at the command level, and must adapt to it's immediate environment when it is prepared for use in propagation. That means that the current state of the objects must be assessed, any transient forces that the user has toggled (FiniteThrust, I'm looking at you!) must be added to the model, the state vector may need to be resized, and so forth. Because of the local nature of this piece, it's simpler to have each propagation enabled command manage its ODEModel than to use a globally (Sandbox scope) instance that is constantly transmorphing based on its local environment.  I'll take a look at what is going on in the function example. I don't understand how functions can be forcing the ODEModel used by the Propagate command to be the object in the global object store. When the model is set up by the Propagate command, the command makes a clone of the object store's PropSetup in its Initialize() method. Here's the code:  for (StringArray::iterator i = propName.begin(); i != propName.end(); ++i)  {  // Lots of checking things before hand, then we get to this:  ...  if ((mapObj = FindObject(pName)) == NULL)  throw CommandException(  "Propagate command cannot find Propagator Setup \"" + (pName) +  "\"\n");  ...  // HERE IS WHERE THE CLONE IS MADE: vvvvvvvvvvvvvvvvv  PropSetup \*clonedProp = (PropSetup \*)(mapObj->Clone());  propagators.push\_back(clonedProp);  ++cloneCount;  if (!propagators[index])  return false;  Propagator \*p = propagators[index]->GetPropagator();  if (!p)  throw CommandException("Propagator not set in PropSetup\n");  p->TakeAction("PrepareForRun");  ...  Clearly the PropSetup is a clone, not the global instance, when the Initialize method returns from its call. |
| Comment by [Darrel Conway](http://li64-187.members.linode.com:8080/secure/ViewProfile.jspa?name=djcinsb) [ 14/Mar/12 9:19 PM ] |
| Here is my current list of objects used by cloning:   * PropSetups inside of Propagate Commands * PropSetups inside of the other PropagationEnabledCommands. (Likely that there is a refactoring piece that needs to be done here!) * Tanks on Spacecraft * \*\*One piece we need to address: can a user script both global and local hardware changes? * Thrusters on Spacecraft * Other hardware on Spacecraft * Solvers inside of their solver control sequence commands (Target, Optimize, Simulate, Estimate)   There may be others – I would have missed the Solver piece if not for your list, Steve. |
| Comment by [Steven Hughes](http://li64-187.members.linode.com:8080/secure/ViewProfile.jspa?name=shughes) [ 16/Mar/12 3:28 PM ] |
| Darrel, I reviewed your document and think you are on the right track. It looks great so far. From your comment above about missing cloned objects on Target. I hope the design can be robust/maintainable and by that I mean help avoid bugs/failures because of human oversight in identifying all permutations of Referenced, Owned, and Cloned objects in the system. Hopefully there is some balance between robustness and performance. Your idea of making methods abstract sounds like it would partially solve the problem by requiring developers to write the method for each object. However, if we were to add a dependency later, if there is not an automated way to handle the problem then we are vulnerable to bugs. (don't claim to have the first idea how to solve that problem though.)  One important issue missed in the problem statement is the design for re-initializing an object when a run-time change by the user, in command mode, requires an object to be re-initialized. Changing a gravity file or opening a new report file, are the cases that come to mind first.  Here are a few more known failures of this type:  Thruster\_FullPoly\_EarthSat\_EarthProp\_EarthThruster\_UserLVLH\_Cmd  Thruster\_FullPoly\_EarthSat\_EarthProp\_EarthThruster\_UserLVLH\_Cmd2 |
| Comment by [Darrel Conway](http://li64-187.members.linode.com:8080/secure/ViewProfile.jspa?name=djcinsb) [ 16/Mar/12 4:16 PM ] |
| Actually, the initialization issue is something I was planning to address in the design rather than in the overview. We do this already in the commands – there is a flag in most commands named "initialized" that is cleared on construction and set at the (successful) end of the Initialize() method. The design here will move that flag into GmatBase and use it to track settings that corrupt the initialized state. |
| Comment by [Wendy Shoan](http://li64-187.members.linode.com:8080/secure/ViewProfile.jspa?name=gmat_wcs) [ 21/Mar/12 4:21 PM ] |
| I believe the failure of this script is because of this issue as well:  GMAT\_GEO\_Earth\_JGM3\_0\_0\_SolidAndPoleTide\_Cmd  The script is setting the EarthTideModel in Command Mode, but the value is not getting to the GravityField object that is actually being used in the computations. Looks like there is a GravityField cloning happening after the setting of the EarthTideModel, but its not cloning the object on which the value was set.  Some debug:  Running mission...  Cloning the solar system in the Sandbox  Successfully set Planetary Source to use: DE405  Successfully set Planetary Source to use: DE405  GravityField GravityField.Earth <0x687a200> COPIED, and earthTideModel = None  GravityField GravityField.Earth <0x68cbe00> COPIED, and earthTideModel = None  For body Earth, not using potential file, so using default mu (398600.441500000015366822)  GravityField GravityField.Earth <0x68e4000> COPIED, and earthTideModel = None  For body Earth, not using potential file, so using default eq. radius (6378.136300000000119326)  Now attempting to set string EarthTideModel on object GravityField.Earth to SolidAndPole  successfully set earthTideModel to SolidAndPole  GravityField GravityField.Earth <0x68e4000> COPIED, and earthTideModel = None  BEFORE CalculateFullField, bodyName = Earth, and earthTideModel = None |
| Comment by [Joel Parker](http://li64-187.members.linode.com:8080/secure/ViewProfile.jspa?name=jjkparker) [ 02/Apr/12 6:41 PM ] |
| Moving all 2012a M3 items to 2012a. |

Generated at Thu May 10 19:13:33 UTC 2012 by Steve Cooley using JIRA 4.4.4#664-r167664.

These tests run to completion but fail:

([Gmat-buildtest] Test results: 2012-05-09 (Win7-64/GMAT-32/M2010a/VS))

Formation\_Validation\_MultiFiniteBurnsInOneCmd (script) [no warn or err]

FuelTank\_AllowNegativeFuelMass\_Cmd (script) [pos err 5.439568 vel err

0.007751 mass err 56.658000]

FuelTank\_PressureModel\_Cmd (script) [pos err 5.439568 vel err 0.007751 mass err 56.658000]

ImpulsiveBurn\_Validation\_GravitationalAccel\_7 (script) [no error or exception]

ImpulsiveBurn\_Validation\_Isp\_7 (script) [no error or exception]

Thruster\_FBurn\_Earth\_ScA\_ThrusterAThrusterF\_CS0\_TankA (script) []

Thruster\_FBurn\_Jupiter\_ScA\_ThrusterB\_CS3\_TankA (script) [pos err

8522.603141 vel err 4.498468]

Thruster\_FBurn\_Luna\_ScA\_ThrusterB\_CS3\_TankA (script) [pos err 3849.515704 vel err 5.130226]

Thruster\_FBurn\_Mars\_ScA\_ThrusterB\_CS3\_TankA (script) [pos err 5735.462891 vel err 5.904614]

Thruster\_FBurn\_Mercury\_ScA\_ThrusterB\_CS3\_TankA (script) [pos err

4112.302119 vel err 4.975586]

Thruster\_FBurn\_Neptune\_ScA\_ThrusterB\_CS3\_TankA (script) [pos err

9714.375310 vel err 5.600302]

Thruster\_FBurn\_Pluto\_ScA\_ThrusterB\_CS3\_TankA (script) [pos err 3979.909732 vel err 4.908047]

Thruster\_FBurn\_Pluto\_ScA\_ThrusterE\_CS0\_TankA (script) [ vel err 0.000015]

Thruster\_FBurn\_Saturn\_ScA\_ThrusterB\_CS3\_TankA (script) [pos err

5446.647337 vel err 2.448408]

Thruster\_FBurn\_Uranus\_ScA\_ThrusterB\_CS3\_TankA (script) [pos err

8587.825939 vel err 4.779854]

Thruster\_FBurn\_Venus\_ScA\_ThrusterB\_CS3\_TankA (script) [pos err 7177.873817 vel err 5.341257 mass err 354.986237]

Thruster\_FiniteBurn\_EarthSat\_EarthProp\_EarthThruster\_LocalVNB (script) [pos err 0.115353, tol 0.000100]

Thruster\_FiniteBurn\_EarthSat\_EarthProp\_EarthThruster\_UserVNB (script) [pos err 0.115353, tol 0.000100]

Thruster\_FiniteBurn\_EarthSat\_EarthProp\_MoonThruster\_LocalVNB (script) [pos err 0.115201, tol 0.000100]

Thruster\_OtherNumericProperties\_Cmd (script) [pos err 1381.956492 vel err

1.061481 mass err 67.730840]

Thruster\_Validation\_GravitationalAccel\_7 (script) [no error or exception]

Thruster\_Validation\_Tank (script) [no error or exception]

Thruster\_Validation\_Tank\_2 (script) [no error or exception]

Thruster\_Validation\_Tank\_4 (script) [no error or exception]

Thruster\_Validation\_ThrustDirection1\_3 (script) [no error or exception]

Thruster\_Validation\_ThrustDirection2\_3 (script) [no error or exception]

Thruster\_Validation\_ThrustDirection3\_3 (script) [no error or exception]

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

The pressure model describes how pressure in the tank changes as fuel is depleted. The user has the choice of PressureRegulated or BlowDown models.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

## Spec User’s Guide

This section is for spec writers to explain how we write specs. Assume for this release that this document will be used by software testers to test the software, and by technical writers to write end-user documentation. When possible write the specifications so that content can be used with little or no modification in user-documentation and have therefore written the document as if the end-user is the reader. The organization of the functional spec section is modelled after MATLAB’s help. See existing GMAT example for the For command, Spacecraft Epoch, and Propagator for good examples of finished specifications.

**General spec writing guidelines**

* Follow all rules in the [GMAT Style Guide](https://docs.google.com/document/d/1dgoXJNL5XqyGQQXd_RGCxDAmqyCOQDiiPZo9p20Fslk/edit?authkey=CLDDzqoI&hl=en_US&authkey=CLDDzqoI#heading=h.pt9u8o9vvvcj).
* Use clear, simple, active voice
* Assume tech-writer/end-user is the primary audience for all material not contained in message boxes (see below). i.e. (write with end-user quality material in main sections, use whatever is necessary to convey the point in message boxes).
* Read existing specs for example of style before writing
  + See Force-Model for a rigorous Resource Example
  + See Target for a rigorous Command example
* If you are a feature lead drafting a spec chapter, emphasize completeness of content. The style does not have to be perfect, it will be reworked by the doc owner.
* If you are the PDL, focus on clarity and consistency of the text.
* Here are special annotation styles

|  |
| --- |
| Open Issue: Use a red table box like this to point out issues in behavior or functionality that are not resolved. THESE ARE NOT BUGS!! Must be resolved before spec is finished. |

|  |
| --- |
| Caution: Use a red table box to include potentially confusing or very important information for users. Examples include when a feature only works in the script and not in the GUI, or if a feature can potentially be misused if the user does not understand something critical. |

|  |
| --- |
| For GUI Tester: Put information intended for a specific document user in a separate grey table box. For example, if a feature may require a unique GUI test type, let the GUI tester know by including the information in a box like this. |

Put script snippets in a grey table box and use monospace font.

|  |
| --- |
| BeginFiniteBurn aFiniteBurn(aSat)  BeginFiniteBurn aFiniteBurn(aSat)  BeginFiniteBurn aFiniteBurn(aSat) |

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Non-urgent question for later investigation: How does STK use the “Maximum Fuel Mass” input parameter?

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

## FuelTank\_PressureModel\_Cmd failed regression test

### Truth

12 Jul 2011 11:09:27

Satellite-TankTest: J2000 Position & Velocity

Time (ModJDate) x (km) y (km) z (km) vx (km/sec) vy (km/sec) vz (km/sec) Total\_Mass (kg)

------------------ ------------------ ----------------- ----------------- --------------- --------------- --------------- ---------------

54892.766367160002 6857.006000000000 -347.034000000000 2881.123000000000 -0.430660000000 6.994217000000 2.081014290000 1383.46540000

54892.787200500003 -1742.585350602479 7268.788361436424 1588.177124113620 -6.544157864949 -0.714846966421 -3.095412921227 1278.77862736

### GMAT Run (SVN Repository)

1500 756

24893.26636716435 6857.006 -347.034 2881.123 -0.43066 6.994217 2.08101429 1440.1234

24893.2872004977 -1740.898839498974 7263.698263339102 1587.263074706112 -6.541514829883046 -0.7220381586293397 -3.096589758397391 1335.436627363813

### GMAT (locally run with fuel tank params set in resource tree):

1700.3452 699.3425

24893.26636716435 6857.006 -347.034 2881.123 -0.43066 6.994217 2.08101429 1383.4659

24893.2872004977 -1742.585334422391 7268.788311110617 1588.177114847435 -6.544157839907467 -0.7148470373822584 -3.095412933273428 1278.779127363813

## Possible Future Fine Tuning of Test Cases (for later builds)

1. Develop matlab script to determine final position, velocity, and mass for the FuelTank\_AllowNegativeFuelMass\_Cmd test case.
2. Add a test case to show that if fuel goes negative and AllowNegativeFuelMass=false, then an error message pops up. I have verified this locally. (would modify FuelTank\_AllowNegativeFuelMass\_Cmd test case.)
3. Add more and/or refine PressureModel validation tests (e.g., GMT-2000)
4. As a GMAT-wide general comment, refine/verify Validation tests work as expected. Need to verify that test fails for reason expected. Example below.
   1. Test Tank\_Validation\_PressureModel\_4, received following error:

1: \*\*\*\* ERROR \*\*\*\* Interpreter Exception: Setting "Tank1.PressureModel" to "str" is not allowed before BeginMissionSequence in line:

" 10: Tank1.PressureModel = str;"

* 1. The line, Tank1.PressureModel = str;" should probably (check with Steve H) be moved to after the BeginMissionSequence line. If one does this, a different error message is obtained:

Hardware Exception Thrown: The value of "myStr" for field "PressureModel" on object "Tank1" is not an allowed value.

The allowed values are: [ PressureRegulated, BlowDown ]. in

"Tank1.PressureModel = str;"

1. Delete unused test files. Recall that there are 89 \*.tc files, 91 \*.script files, and 79 \*.truth files.

## Notes

1. The Math specs incorrectly list the pressure units for the Thrust/ISP polynomials as Pascal. The units are in kPa.