



Pilot Evaluation of Model Based Design Tooling for Guidance, Navigation, and Control Flight Software Development

Brian R. Jamison, Mike R. Hannan,
James T. Kaidy, Juan I. Orphee, Nick S. Olson

Guidance, Navigation, & Control

NASA Marshall Space Flight Center

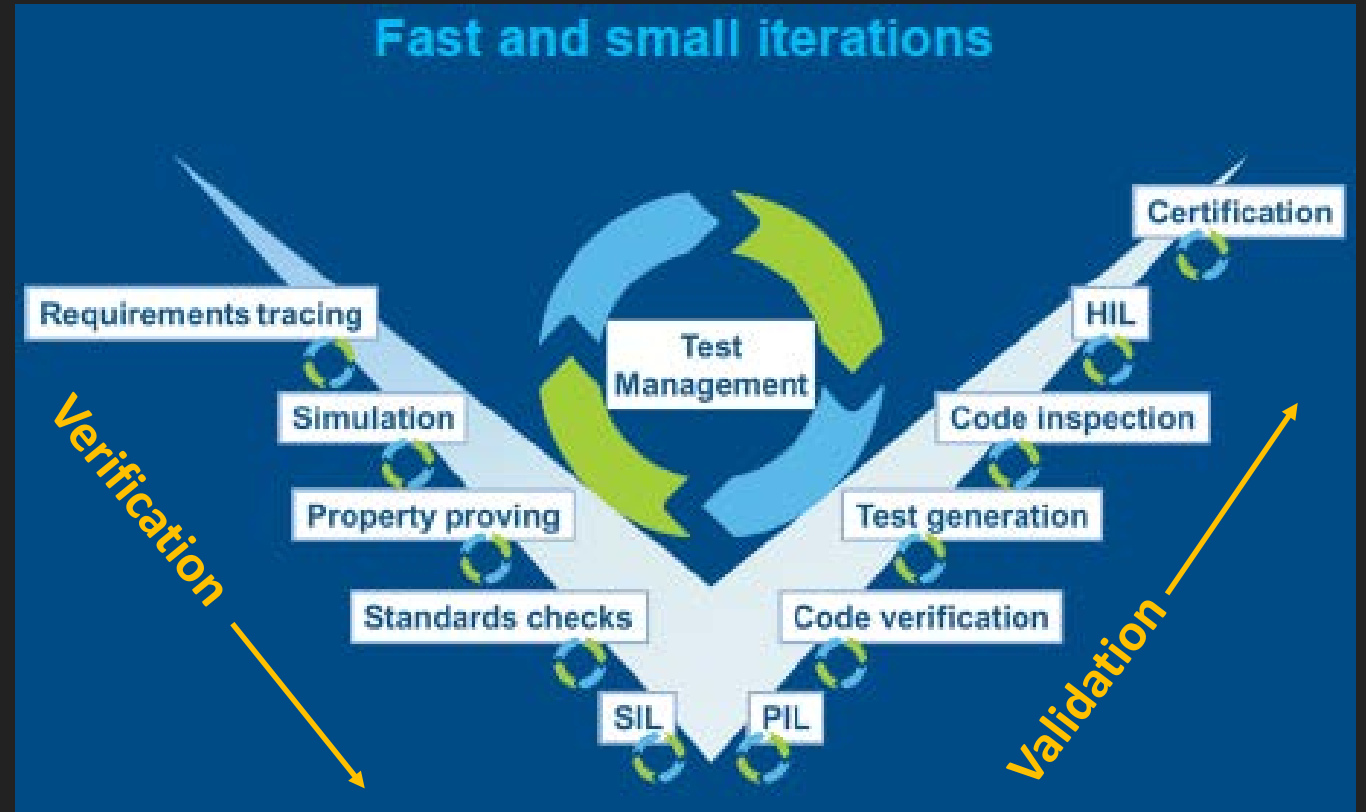
9-12

December

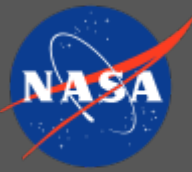
2019

What & Why Model Based Design (MBD)

- MBD systematically uses models throughout the development process for requirements, design, analysis, simulation, verification and validation, and documentation
- An MBD approach seeks to incorporate models into an automated, concurrent design process intended to minimize potential for human error
- Improvements offered by an MBD approach include efficiency improvements by automating aspects of requirements testing and documentation
- An advantage of MathWorks MBD tooling is the model visualization
Simulink naturally incorporates into the design process



NASA LADEE MBD Experience



- “Compared with using Model-Based Design, hand-coding the flight software would have taken longer and made collaboration more difficult. Managers and hardware system engineers understand Simulink models, making it easy to achieve consensus because everyone knows what’s going on in the software.”
- *Dr. Karen Gundy-Burlet, NASA Ames Research Center*

Model Based Development

- **Issues:**
 - Low Cost Mission and fixed schedule demanded rapid, low cost flight software development process
 - Simulations needed for FSW Verification and Mission Operations development, training, and command verification.
- **Solution:**
 - Use model based development approach
 - Automatic conversion of Models to FSW allows development and testing of algorithms which then becomes Software. Avoids “throwing it over the fence to be coded”.
 - Developed multiple simulators of varying degrees of fidelity (WSIM, PIL, HIL)
 - Developed Simulink Interface Layer
 - Allows immediate translation from models to Code allowing rapid turnaround
 - Developed an automated test harness for rapid turnaround of verification results
- **Result:**
 - Model Based Development coupled with “push button” code generation and testing was highly effective for rapid software development.
 - Models and Simulations used extensively in Mission Operations.
 - WSIM provided faster than real time capability for rapid command verification.
 - Processor in the Loop and Hardware in the Loop simulations provided high fidelity simulations for critical maneuver verification, Ops training, and debugging anomalies
 - Fully Coupled Simulations (Power, Thermal, Propulsion, Attitude Control) provided better insight for coupled problems.

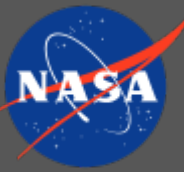
Iterative Development

Iterate Early and Often


Verification

- Develop Models of FSW, Vehicle, and Environment
- Automatically generate High-Level Control Software
- Integrate with hand-written and heritage software.
- Iterate while increasing fidelity of tests – Workstation Sim (WSIM), Processor-In-The-Loop (PIL), Hardware-in-the-Loop (HIL)
- Automated self-documenting tests providing traceability to requirements


NASA LADEE MBD Experience



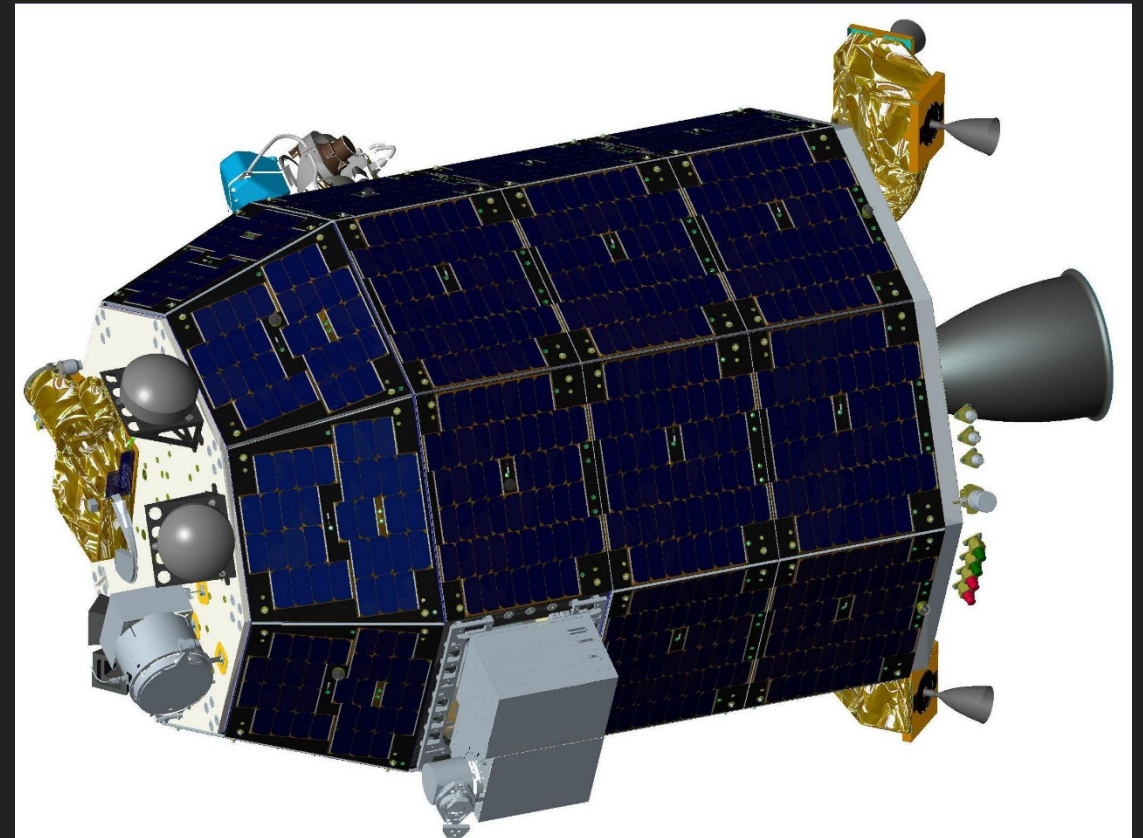
- “Compared with using Model-Based Design, hand-coding the flight software would have taken longer and made collaboration more difficult. Managers and hardware system engineers understand Simulink models, making it easy to achieve consensus because everyone knows what’s going on in the software.”
 - *Dr. Karen Gundy-Burlet, NASA Ames Research Center*



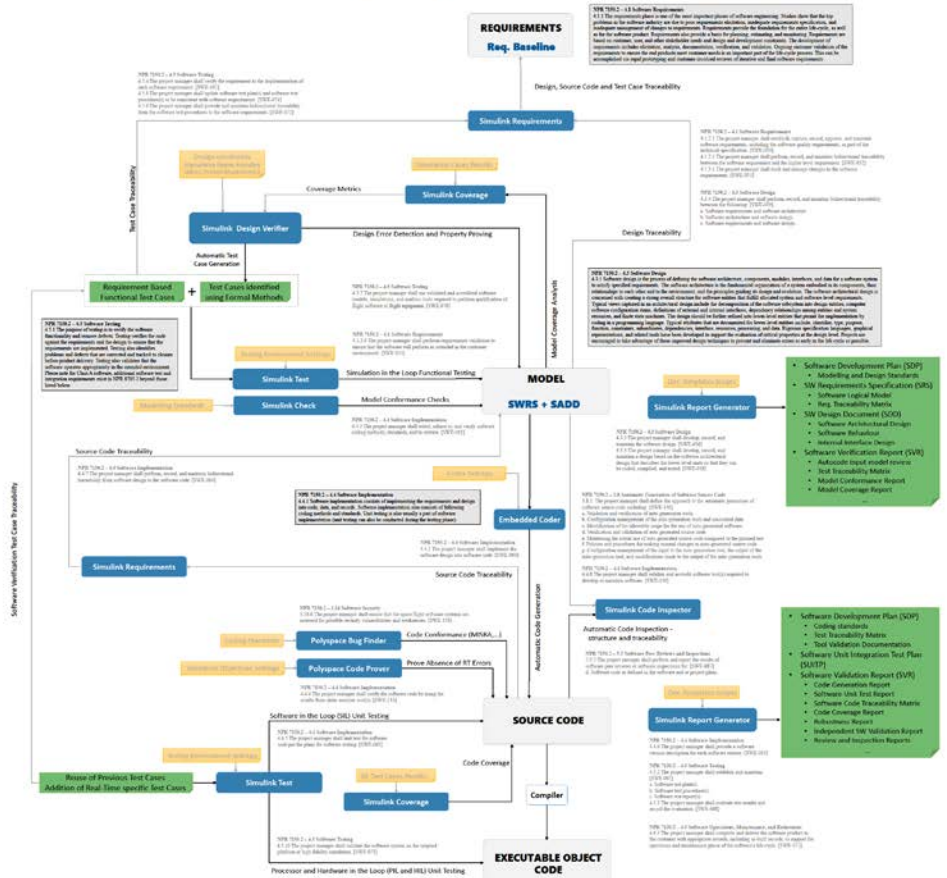
Model Based Development



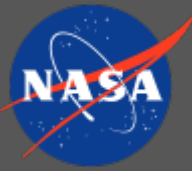
- Issues:
 - Low Cost Mission and fixed schedule demanded rapid, low cost flight software development process
 - Simulations needed for FSW Verification and Mission Operations development, training, and command verification.
- Solution:
 - Use model based development approach
 - Automatic conversion of Models to FSW allows development and testing of algorithms which then becomes Software. Avoids “throwing it over the fence to be coded”.
 - Developed multiple simulators of varying degrees of fidelity (WSIM, PIL, HIL)
 - Developed Simulink Interface Layer
 - Allows immediate translation from models to Code allowing rapid turnaround
 - Developed an automated test harness for rapid turnaround of verification results
- Result:
 - Model Based Development coupled with “push button” code generation and testing was highly effective for rapid software development.
 - Models and Simulations used extensively in Mission Operations.
 - WSIM provided faster than real time capability for rapid command verification.
 - Processor in the Loop and Hardware in the Loop simulations provided high fidelity simulations for critical maneuver verification, Ops training, and debugging anomalies
 - Fully Coupled Simulations (Power, Thermal, Propulsion, Attitude Control) provided better insight for coupled problems.



NASA NPR 7150.2 Compliant Flight Software Development Workflow



MathWorks MBD Tooling



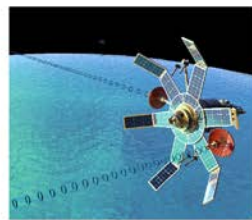
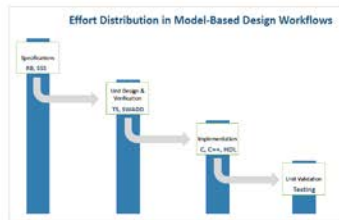
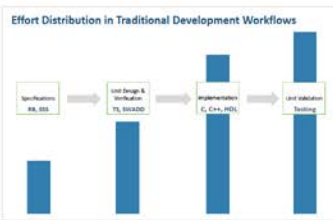
• MathWorks worked with the NASA Engineering & Safety Center (NESC) to develop tooling that addresses about 80% of NPR 7150.2 requirements, including:

- Software requirements
- Software design
- Software implementation
- Software testing

• NPR 7150.2 Requirements outside of the MathWorks workflow include:

- Software architecture requirements
- Project management requirements

• Consultation ongoing with the Marshall Space Flight Center (MSFC) software division to ensure they concur with our MBD approach



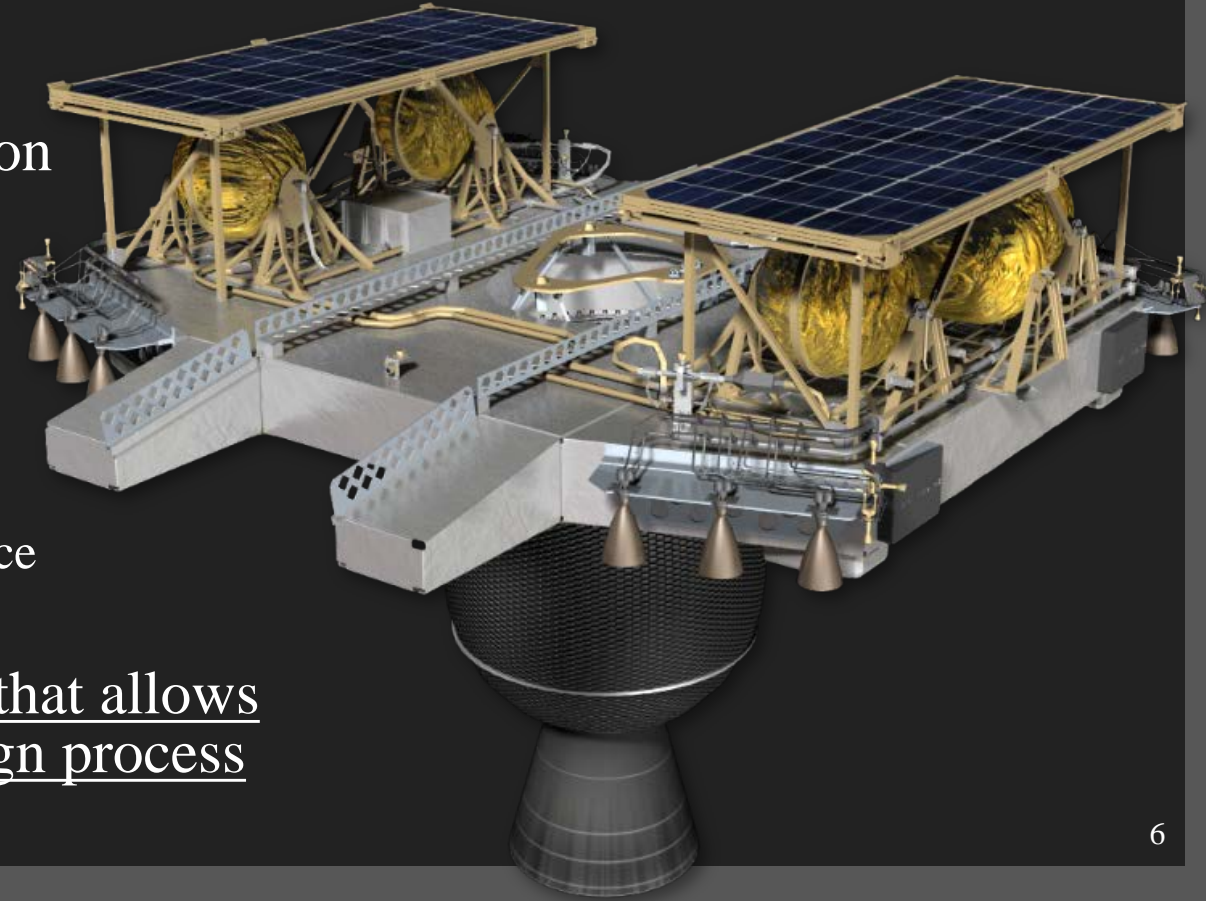
VIPER Lunar Lander Pilot Program



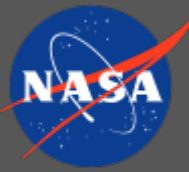
- MSFC Guidance, Navigation, & Control (GNC) group used the VIPER Lander as a pilot program to use of MathWorks MBD tools for GNC FSW development
- Goal: Apply an MBD approach and tooling to condense schedule, reduce needed resources, and improve quality

Expected MBD approach benefits:

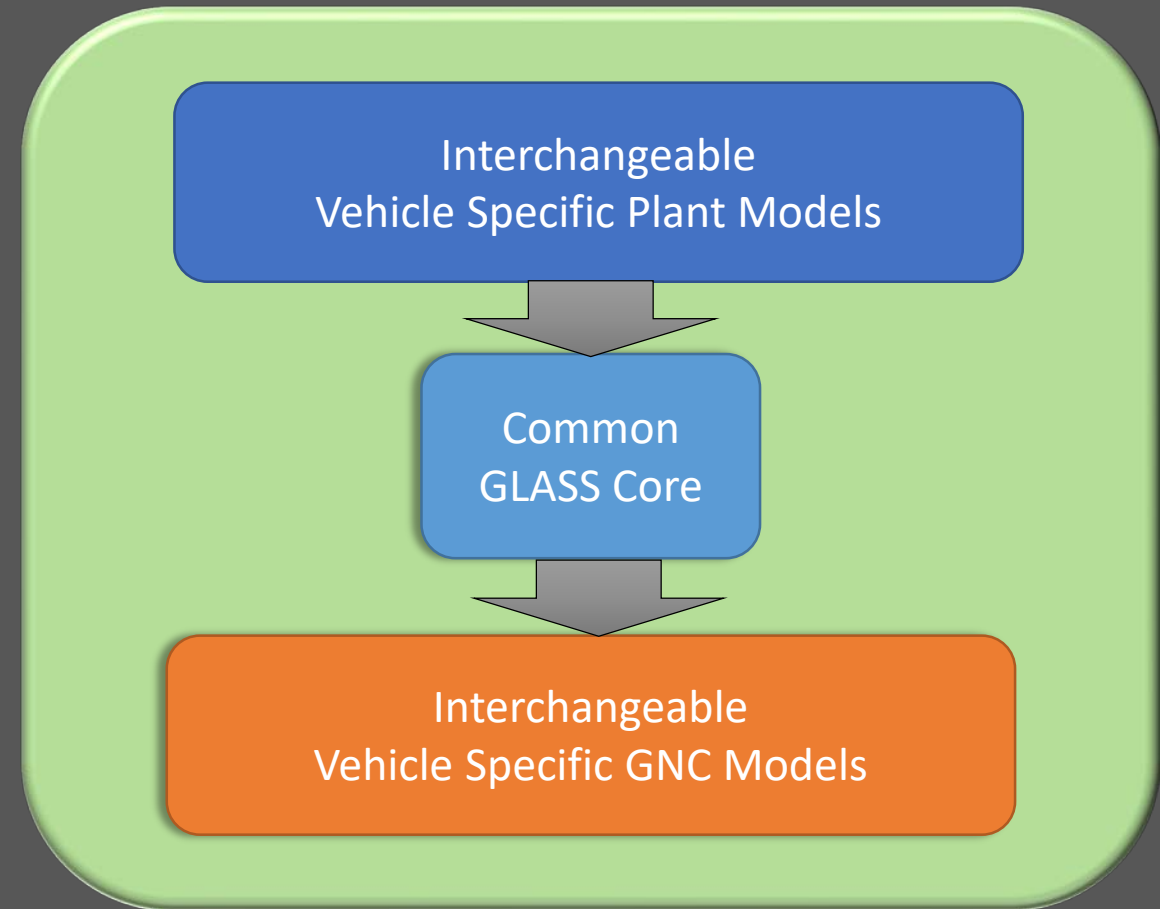
- Facilitates requirements implementation verification
- Automates:
 - Requirements verification testing
 - Continuous model and flight code testing
 - Modeling standards (DO-178C) enforcement
 - Code-generation from the Simulink model
 - Static code analysis to ensure coding standard compliance
 - Report generation
- Establish a highly automated, disciplined process that allows repeated testing of the system throughout the design process



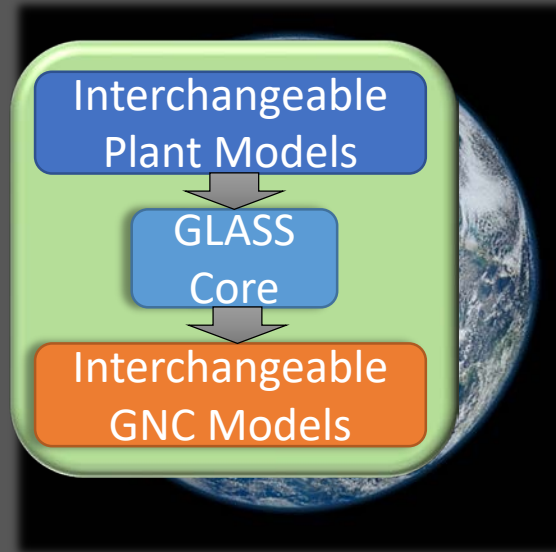
Generalized Lander Simulation in Simulink (GLASS)



- 6-DoF aerospace vehicle simulation environment designed to be:
 - Modern
 - Flexible
 - User-friendly
- Features
 - Simulink Framework
 - Interfaces with MathWorks MBD products
 - Supports auto-coding to C/C++
 - Dynamics constructed using Simscape Multibody
 - Provides flexible & modular physics engine for simulation
 - GLASS Core is common to all simulations
 - Modular GNC algorithms
 - Mirror FSW functions in generated code



MBD for GNC FSW Development



Check Requirements: Simulink Requirements & Simulink Test

- ✓ Ensure code satisfies requirements

Model & GNC Code Development: Simulink & Model Advisor

- ✓ Highly modular software development
- ✓ Enforce DO178 and custom standards

Generate Automated Reports: Report Generator

Software Unit Testing: Simulink Model & Code Coverage

- ✓ Ensure software modules perform as expected
- ✓ Ensure all code is exercised

Auto-code GNC Software: Simulink Embedded Coder

- ✓ Customize auto-code to meet FSW standards
- ✓ Enforce selected standards
- ✓ cFS compatible (optional)

Static Code Check: Polyspace

- ✓ Catch run time errors,
- ✓ Enforce MISRA, JSF, etc.

```
1 // NASA MSFC GNC
2 Quality
3 Autocode
4 GNC
5 FSW
```


MBD Workflow



Requirements, e.g., in Excel

Requirement ID	Requirement Name	Local & Global Requirements	Baseline	Requirement Text	Unit Value	Units	Comments	File Name
14-000-1.1	Rolling Control			During roll phase the ACS shall be capable of providing authority of 10 degrees of roll.		degrees		
14-000-1.2	Angular Rate Control			During roll phase the ACS shall be capable of providing 10 degrees/second of roll rate.		degrees/second		
14-000-1.3	Roll Rate Authority			During roll phase the ACS shall be capable of providing roll rate authority of 10 degrees/second.	10	degrees	Roll rate authority limit	UPL_VECTOR
14-000-1.4	Roll Rate Authority			During roll phase the ACS shall be capable of providing roll rate authority of 10 degrees/second.	10	degrees	Roll rate authority limit	UPL_VECTOR_LIMITATION
14-000-1.5	Roll Rate Authority			During roll phase the ACS shall be capable of providing roll rate authority of 10 degrees/second.	10	degrees	Roll rate authority limit	UPL_VECTOR_LIMITATION

Requirement Statements

Simulink Requirements

Req ID	Req Text	Implemented	Verified
1.1	Switch precedence	Yes	Yes
1.2	Avoid repeating commands	Yes	Yes
1.3	Long Switch recognition	Yes	Yes
1.4	Cancel Switch Detection	Yes	Yes
1.5	Set Switch Detection	Yes	Yes
1.6	Enable Switch Detection	Yes	Yes
1.7	Resume Switch Detection	Yes	Yes
1.8	Increment Switch Detection	Yes	Yes
1.9	Decrement Switch Detection	Yes	Yes
2.0	Cruise Control Mode	Yes	Yes
2.1	Disable Cruise Control system	Yes	Yes
2.2	Operation mode determination	Yes	Yes
2.3	Calculate Target Speed and Throttle	Yes	Yes
3.1	Disabled case	Yes	Yes
3.2	Enabled case	Yes	Yes
3.3	Activated case	Yes	Yes
3.3.1	Theoretic Value Computation	Yes	Yes
3.3.2	Next Target Speed Computation	Yes	Yes
3.4	Resume mode	Yes	Yes
3.5	System Interface	Yes	Yes
3.5.1	Inputs	Yes	Yes
3.5.2	Outputs	Yes	Yes
3.5.3	Parameters	Yes	Yes
4	Justifications	Yes	Yes
4.1	References to cts_req	Yes	Yes
1.1	1_Overview	Yes	Yes
1.2	2_Systems	Yes	Yes
1.3	3_FUNCTION - FUNCTIONAL REQUIREMENTS	Yes	Yes

Iteration Updates Synchronized via Script



Test Parameters

Requirements Verification

Table of Contents

- Chapter 1. Introduction
 - 1.1. Model-Based Design (MBD) for Guidance, Navigation, and Control (GNC)
- Chapter 2. Demonstration Requirements Implementation and Verification Status
 - 2.1. L4-GNC-5.1.1
 - 2.1.1. L4-GNC-5.1.1.1
 - 2.1.2. L4-GNC-5.1.1.2
 - 2.1.3. L4-GNC-5.1.1.3
 - 2.1.4. L4-GNC-5.1.1.4
 - 2.1.5. L4-GNC-5.1.1.5

Automated Report Generation

Chapter 3. L4-GNC-5.1.1.1, Landing/Touchdown Requirements Set

L4-GNC-5.1.1.1 Implementation Status

Implemented: 0, Assessed: 0, No Status: 1

L4-GNC-5.1.1.1 Verification Status

Passed: 3, Assessed: 0, Failed: 2, Not Executed: 5, No Status: 10

Description:
At touchdown the minimum vertical velocity magnitude shall be no greater than 25 meter/second.

Rationale:

Contact Limits:

Status:

Simulink Test

Unit Testing & Model Coverage Testing

Model Element	Complete	Test	Results	Passes
1. Model	100%	100%	100%	100%
2. Controller	NA	100%	100%	100%
3. Propulsion	100%	100%	100%	100%
4. Drag Force	NA	100%	100%	100%
5. Bank Dynamics	NA	100%	100%	100%
6. Lateral/Heading	40%	100%	100%	100%
7. Roll/Heading	NA	100%	100%	100%
8. Yaw	100%	100%	100%	100%
9. Roll/Yaw	100%	100%	100%	100%
10. ... TBD and BDC Junction	NA	100%	100%	100%
11. ... TBD and BDC Junctions	NA	100%	100%	100%

Full Coverage

Model: Coverage (this object): 4

Coverage (dec. descendants): 75% (8/12) decision outcomes

Execution: 100% (2/2) discrete outcomes

Unit Testing & Model Coverage Testing

Requirements.xls [Compatibility Mode] - Excel

FILE HOME INSERT PAGE LAYOUT FORMULAS DATA REVIEW VIEW DEVELOPER

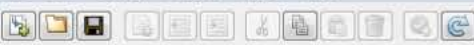
Clipboard Font Alignment Number Styles Cells Editing

Calibri 12 A A Wrap Text General Accent1.2 Normal 2 Normal Bad Good Neutral Calculation Check Cell Explanatory... Input

AutoSum Fill Clear Sort & Find & Filter Select

164 0.5

	D	E	F	G	H	I	J
	Requirement ID	Requirement Name	Level 4 GN&C Requirement	Rationale	Level 4 GN&C Requirement Text	Spec Value	Units
62	L4-GNC-5.3	Landing/Touchdown Upper Limit Requirements					
63	L4-GNC-5.3.1	Maximum Vertical Velocity at Touchdown	At touchdown, the maximum vertical velocity magnitude shall be no greater than 2.5 meters/second.	Contact Limits	At touchdown, the maximum vertical velocity magnitude shall be no greater than	2.5	meters/second
64	L4-GNC-5.3.2	Maximum Lateral Velocity at Touchdown	At touchdown, the maximum lateral velocity magnitude shall be no greater than 0.5 meters/second.	Contact Limits	At touchdown, the maximum lateral velocity magnitude shall be no greater than	0.5	meters/second
65	L4-GNC-5.3.3	Maximum Pitch Angle at Touchdown	At touchdown, the maximum pitch angle with respect to local vertical shall be no greater than 3 degrees.	Tip Over Limits	At touchdown, the maximum pitch angle with respect to local vertical shall be no greater than	3	degrees
66	L4-GNC-5.3.4	Maximum Angular Rate at Touchdown	At touchdown, the maximum magnitude of the inertial angular rate shall be no greater than 0.1 degrees/second.	Tip Over Limits	At touchdown, the maximum magnitude of the inertial angular rate shall be no greater than	0.1	degrees/second
67	L4-GNC-5.3.5	Maximum Position Error	At touchdown, the maximum position error from the nominal landing site shall be no greater than 100 meters.	Landed Payload Limits	At touchdown, the maximum position error from the nominal landing site shall be no greater than	100	meters
68	L4-GNC-5.3.6	Maximum Lateral CM Offset	At touchdown the lateral CM offset shall be no greater than 77 millimeters.	Control Stability Margin	At touchdown the lateral CM offset shall be no greater than	77	millimeters
69							
70							
71							
72							
73							
74							



View: Requirements

Search

Update completed. Refer to Comments on Import1.

Index	ID	Summary	Implemented	Verified
Requireme...				
Import1	Requirements!Sheet1	References to Requirements.xls (Shee...		
1	L4-GNC-1	Coast Phase Requirements		
2	L4-GNC-2	TCM Burns Requirements		
3	L4-GNC-3	SRM Burn Phase Requirements		
4	L4-GNC-4	Liquid Descent Requirement		
5	L4-GNC-5	Landing/Touchdown Requirements		
5...	L4-GNC-5.1	Landing/Touchdown Control Require...		
5...	L4-GNC-5.2	Landing/Touchdown Knowledge Requ...		
5...	L4-GNC-5.3	Landing/Touchdown Upper Limit Req...		
	L4-GNC-5.3.1	Maximum Vertical Velocity at Touchd...		
	L4-GNC-5.3.2	Maximum Lateral Velocity at Touchdo...		
	L4-GNC-5.3.3	Maximum Pitch Angle at Touchdown		
	L4-GNC-5.3.4	Maximum Angular Rate at Touchdown		
	L4-GNC-5.3.5	Maximum Position Error		
	L4-GNC-5.3.6	Maximum Lateral CM Offset		

Properties

Type: Functional
Index: 5.3.2
Custom ID: L4-GNC-5.3.2
Summary: Maximum Lateral Velocity at Touchdown

Description Rationale

At touchdown, the maximum lateral velocity magnitude shall be no greater than 0.5 meters/second.

Keywords:

Revision information:
SID: 64
Revision: 42
Updated on: 15-Nov-2019 10:27:24
Created by:
Created on: 07-Aug-2019 13:19:11
Modified by:
Modified on: 15-Nov-2019 10:27:21

Show in document Unlock

Links

Verified by:
L4-GNC-5.3.2

Comments

Test Browser Results Artifacts

Filter results by name or tags, e.g. tags: test

NAME	STATUS
▶ Results: 2019-Nov-15 10:27:27	199 1
▶ Results: 2019-Nov-15 10:33:25	197 3
▶ Results: 2019-Nov-15 10:39:15	199 1
▶ Results: 2019-Nov-15 10:45:14	200
▶ Results: 2019-Nov-15 10:51:10	200

PROPERTY	VALUE

L4-GNC-5.3.2 x

L4-GNC-5.3.2

Tests ▶ L4-GNC-5 ▶ L4-GNC-5.3 ▶ L4-GNC-5.3.2

Simulation Test

Select releases for simulation:

Create Test Case from External File

- ▶ TAGS
- ▶ DESCRIPTION
- ▶ REQUIREMENTS*
- ▼ SYSTEM UNDER TEST* ?
 - Model:
 - ▶ TEST HARNESS
 - ▶ SIMULATION SETTINGS OVERRIDES*
- ▶ PARAMETER OVERRIDES ?
- ▶ CALLBACKS ?
- ▶ INPUTS ?
- ▶ SIMULATION OUTPUTS ?
- ▶ CONFIGURATION SETTINGS OVERRIDES ?
- ▶ ITERATIONS* ?
- ▶ LOGICAL AND TEMPORAL ASSESSMENTS ?
- ▶ CUSTOM CRITERIA* ?
- ▶ COVERAGE SETTINGS ?

Test Browser Results and Artifacts

Filter results by name or tags, e.g. tags: test

NAME	STATUS
▶ Results: 2019-Nov-15 10:27:27	199 ● 1 ●
▼ Results: 2019-Nov-15 10:33:25	197 ● 3 ●
▶ L4-GNC-5.3.2	197 ● 3 ●
▶ Scripted_Iteration1	●
▶ Scripted_Iteration10	●
▶ Scripted_Iteration100	●
▶ Scripted_Iteration101	●
▶ Scripted_Iteration102	●
▶ Scripted_Iteration103	●
▶ Scripted_Iteration104	●
▶ Scripted_Iteration105	●
▶ Scripted_Iteration106	●
▶ Scripted_Iteration107	●
▶ Scripted_Iteration108	●
▶ Scripted_Iteration109	●
▶ Scripted_Iteration11	●
▶ Scripted_Iteration110	●
▶ Scripted_Iteration111	●
▶ Scripted_Iteration112	●
▶ Scripted_Iteration113	●

PROPERTY	VALUE

L4-GNC-5.3.2 x

L4-GNC-5.3.2

Tests ▶ L4-GNC-5 ▶ L4-GNC-5.3 ▶ L4-GNC-5.3.2

Simulation Test

Select releases for simulation:

Create Test Case from External File

▶ TAGS

▶ DESCRIPTION

▶ REQUIREMENTS*

▼ SYSTEM UNDER TEST* ?

Model:

▶ TEST HARNESS

▶ SIMULATION SETTINGS OVERRIDES*

▶ PARAMETER OVERRIDES ?

▶ CALLBACKS ?

▶ INPUTS ?

▶ SIMULATION OUTPUTS ?

▶ CONFIGURATION SETTINGS OVERRIDES ?

▶ ITERATIONS* ?

▶ LOGICAL AND TEMPORAL ASSESSMENTS ?

▶ CUSTOM CRITERIA* ?

▶ COVERAGE SETTINGS ?

Enabled

Test Browser Results and Artifacts

Filter results by name or tags, e.g. tags: test

NAME	STATUS
▶ Scripted_Iteration136	✓
▶ Scripted_Iteration137	✓
▶ Scripted_Iteration138	✓
▶ Scripted_Iteration139	✓
▶ Scripted_Iteration14	✓
▶ Scripted_Iteration140	✓
▶ Scripted_Iteration141	✓
▶ Scripted_Iteration142	✓
▶ Scripted_Iteration143	✓
▶ Scripted_Iteration144	✓
▼ Scripted_Iteration145	✗
▶ Sim Output (readIn3x1Vector)	
▼ Custom Criteria Result	✗
▶ Landing Lateral Speed: 0.0	✗
▶ Scripted_Iteration146	✓
▶ Scripted_Iteration147	✓
▶ Scripted_Iteration148	✓
▶ Scripted_Iteration149	✓
▶ Scripted_Iteration15	✓
▶ Scripted_Iteration150	✓

PROPERTY	VALUE
Diagnostic Result	✓ Landing Lateral Speed: 0.517...
Status	✗

Scripted_Iteration145 x L4-GNC-5.3.2 x

▼ SUMMARY

Name	Scripted_Iteration145
Outcome	1 ✗
Start Time	11/15/2019 10:38:44
End Time	11/15/2019 10:38:44
Type	Simulation Test
Test File Location	D:\Landers_20190214\p\MAINMBD\Tests.mdatx
Test Case Definition	✗
Rerun Test Iteration	▶
Cause of Failure	Failed criteria: Custom
▶ Simulation Metadata	

▼ TEST REQUIREMENTS ?

▶ ITERATION SETTINGS ?

▼ ERRORS ?

▼ LOGS ?

▼ DESCRIPTION ?

Double-click to edit

▼ CUSTOM CRITERIA LOGS ?

```

-----
VerificationFailed in custom criteria of sltest.testmanager.TestCase.

-----
Test Diagnostic:
Landing Lateral Speed: 0.51722 meters/second.

-----
Framework Diagnostic:
verifyLessThanOrEqual failed.
--> The value must be less than or equal to the maximum value.

Actual Value:
0.517217612807295
Maximum Value (Inclusive):
0.5000000000000000
-----

```

FILE HOME INSERT PAGE LAYOUT FORMULAS DATA REVIEW VIEW DEVELOPER

Clipboard Font Alignment Number Styles Cells Editing

Conditional Formatting Table

AutoSum Fill Clear Sort & Find & Filter Select

	D	E	F	G	H	I	J
1	Requirement ID	Requirement Name	Level 4 GN&C Requirement	Rationale	Level 4 GN&C Requirement Text	Spec Value	Units
62	L4-GNC-5.3	Landing/Touchdown Upper Limit Requirements					
63	L4-GNC-5.3.1	Maximum Vertical Velocity at Touchdown	At touchdown, the maximum vertical velocity magnitude shall be no greater than 2.52 meters/second.	Contact Limits	At touchdown, the maximum vertical velocity magnitude shall be no greater than	2.52	meters/second
64	L4-GNC-5.3.2	Maximum Lateral Velocity at Touchdown	At touchdown, the maximum lateral velocity magnitude shall be no greater than 0.52 meters/second.	Contact Limits	At touchdown, the maximum lateral velocity magnitude shall be no greater than	0.52	meters/second
65	L4-GNC-5.3.3	Maximum Pitch Angle at Touchdown	At touchdown, the maximum pitch angle with respect to local vertical shall be no greater than 3.25 degrees.	Tip Over Limits	At touchdown, the maximum pitch angle with respect to local vertical shall be no greater than	3.25	
66	L4-GNC-5.3.4	Maximum Angular Rate at Touchdown	At touchdown, the maximum magnitude of the inertial angular rate shall be no greater than 0.1 degrees/second.	Tip Over Limits	At touchdown, the maximum magnitude of the inertial angular rate shall be no greater than	0.1	
67	L4-GNC-5.3.5	Maximum Position Error	At touchdown, the maximum position error from the nominal landing site shall be no greater than 100 meters.	Landed Payload Limits	At touchdown, the maximum position error from the nominal landing site shall be no greater than	100	meters
68	L4-GNC-5.3.6	Maximum Lateral CM Offset	At touchdown the lateral CM offset shall be no greater than 77 millimeters.	Control Stability Margin	At touchdown the lateral CM offset shall be no greater than	77	millimeters
69							
70							
71							
72							
73							
74							

Highlighting indicates and is expected to be executing updateRequirementsT

For more information, ImportantMBDInstruc 2.

EDITOR PUBLISH VIEW

New Open Save Find Files Compare Print Go To Find Comment Indent Breakpoints Run Run and Advance Run Section Advance Run and Time

FILE NAVIGATE EDIT BREAKPOINTS RUN

```

1 %% Beginning from an Excel spreadsheet, as determined necessary, this script creates or
2 % updates a Simulink Data Dictionary and the corresponding Simulink Requirements file, and
3 % loads and reruns the associated Simulink Test files.
4
5 clearvars
6 clc
7
8 %% display a pop-up indicating the need to now close open Excel requirements window to avoid the possibility of errors when clearing highlighting in the .xls requirement file
9 uifig = figure; % Create and then hide the UI as it is being constructed.
10 h = uicontrol('Position',[180 20 200 40],'String','Continue','Callback','uiresuma(gcf)','FontSize',16);
11 uidirections1 = 'To prevent errors during execution of updateRequirementsTesting.m, ensure ';
12 uidirections2 = 'the Excel Requirements.xls window is closed, and then select Continue.';
13 htext = uicontrol('Style','text','String',[uidirections1 uidirections2],'Position',[40,100,480,300],'FontSize',23);
14 uiwait(gcf);
15 close(uifig);
16
17 %% Open the MAIN project
18 open('..\MAIN.prj') % among other things, sets up needed path(s), including to readDict and editDict
19 cd('MBD') % change directory back to MBD directory since previous command went back to MAIN directory
20
21 %% Setup filenames and ensure they are valid
22 xlsFilename = 'Requirements.xls';
23 slreqxFilename = 'Requirements.slreqx';
24 slddFilename = 'Dictionary.slidd';
25 testsFilename = 'Tests.mdatx';
26 testsResultsFilename = 'TestsResults_20190807_1434.mdatx';
27 if ~strcmp(xlsFilename(1:end-3),slreqxFilename(1:end-6))
28     error('The filenames of the .xls and .slreqx files should match.') % e.g., relative to importing materials from the .xls for creation of the .slreqx file if necessary
29 end
30
31 %% Create the data dictionary if necessary
32 if ~exist(sliddFilename,'file') % if the data dictionary does not exist, ...
33     Simulink.data.dictionary.create(sliddFilename); % create it
34 end
35
36 %% specify the number of Monte Carlo iterations, and, as necessary, define or update a corresponding data dictionary parameter
37 NUM_MONTE_CARLO_ITERATIONS_ThisTime = 200;%0;
38 try % if the data dictionary has a NUM_MONTE_CARLO_ITERATIONS parameter such that the following line doesn't result in an error, ...
39     NUM_MONTE_CARLO_ITERATIONS = readDict(sliddFilename,'NUM_MONTE_CARLO_ITERATIONS'); % get it's value (which is expected to be an integer) to see if it needs to be updated
40     if (NUM_MONTE_CARLO_ITERATIONS ~= NUM_MONTE_CARLO_ITERATIONS_ThisTime) % if the NUM_MONTE_CARLO_ITERATIONS parameter does not match the one already in the data dictionary, ...
41         editDict(sliddFilename,'NUM_MONTE_CARLO_ITERATIONS',NUM_MONTE_CARLO_ITERATIONS_ThisTime); % update the NUM_MONTE_CARLO_ITERATIONS parameter in the data dictionary
42     end
43 catch % otherwise, ...
44     editDict(sliddFilename,'NUM_MONTE_CARLO_ITERATIONS',NUM_MONTE_CARLO_ITERATIONS_ThisTime); % add the NUM_MONTE_CARLO_ITERATIONS parameter to the data dictionary
45 end
46
47 %% Check the xls file to see if it has been modified, and if so, set the flag to update the data dictionary and the requirements file
48 xls_file_obj = dir(xlsFilename); % expected to contain 'name' (filename), 'folder' (folder location and name), 'date' (string), 'isdir' (boolean), and 'datenum' (float) fields
49 xls_modified_datenum = xls_file_obj.datenum; % datenum of last modification; their difference, this is a "Modification date as a local dependent MATLAB serial date number"

```

Initialization.mlx

oldLandedTests.m

setupCallback.m

RefineFinalDescentTimeEstimate.m

compareGLASSResults.m

MBDReqsImplStatusColorbarImagePath.m

MBDReqsVerifStatusColorbarImagePath.m

addMBDReqsStatusImages.m

datenum.m

updateRequirementsTesting.m

loadMCIterationsLandingVelocities.m

loadMCIterationsLandingAngularVelocities.m

loadMCIterationsLandingQuaternions.m

loadMCIterationsLandingPositions.m

testTouchdownVerticalVelocityResult.m

testTouchdownLateralSpeedResult.m

testTouchdownPitchResult.m

testTouchdownAngularRateResult.m

testTouchdownDistanceResult.m

GNC_MBD_RptGen.m

	D	E	F	G	H	I	J
1	Requirement ID	Requirement Name	Level 4 GN&C Requirement	Rationale	Level 4 GN&C Requirement Text	Spec Value	Units
62	L4-GNC-5.3	Landing/Touchdown Upper Limit Requirements					
63	L4-GNC-5.3.1	Maximum Vertical Velocity at Touchdown	At touchdown, the maximum vertical velocity magnitude shall be no greater than 2.52 meters/second.	Contact Limits	At touchdown, the maximum vertical velocity magnitude shall be no greater than	2.52	meters/second
64	L4-GNC-5.3.2	Maximum Lateral Velocity at Touchdown	At touchdown, the maximum lateral velocity magnitude shall be no greater than 0.52 meters/second.	Contact Limits	At touchdown, the maximum lateral velocity magnitude shall be no greater than	0.52	meters/second
65	L4-GNC-5.3.3	Maximum Pitch Angle at Touchdown	At touchdown, the maximum pitch angle with respect to local vertical shall be no greater than 3.25 degrees.	Tip Over Limits	At touchdown, the maximum pitch angle with respect to local vertical shall be no greater than	3.25	degrees
66	L4-GNC-5.3.4	Maximum Angular Rate at Touchdown	At touchdown, the maximum magnitude of the inertial angular rate shall be no greater than 0.1 degrees/second.	Tip Over Limits	At touchdown, the maximum magnitude of the inertial angular rate shall be no greater than	0.1	degrees/second
67	L4-GNC-5.3.5	Maximum Position Error	At touchdown, the maximum position error from the nominal landing site shall be no greater than 100 meters.	Landed Payload Limits	At touchdown, the maximum position error from the nominal landing site shall be no greater than	100	meters
68	L4-GNC-5.3.6	Maximum Lateral CM Offset	At touchdown the lateral CM offset shall be no greater than 77 millimeters.	Control Stability Margin	At touchdown the lateral CM offset shall be no greater than	77	millimeters
69							
70							
71							
72							
73							
74							

Test Browser Results and Artifacts

Filter results by name or tags, e.g. tags: test

NAME	STATUS
▶ Results: 2019-Nov-15 10:27:27	199 1
▶ Results: 2019-Nov-15 10:33:25	197 3
▶ Results: 2019-Nov-15 10:39:15	199 1
▶ Results: 2019-Nov-15 10:45:14	200
▶ Results: 2019-Nov-15 10:51:10	200
▶ Results: 2019-Nov-15 12:25:46	200
▶ Results: 2019-Nov-15 12:30:54	200
▶ Results: 2019-Nov-15 12:36:06	197 2 1

PROPERTY	VALUE

Scripted_Iteration145 x L4-GNC-5.3.2 x

▼ SUMMARY

Name	Scripted_Iteration145
Outcome	1
Start Time	11/15/2019 10:38:44
End Time	11/15/2019 10:38:44
Type	Simulation Test
Test File Location	D:\Landers_20190214\p\MAINMBDI\Tests.mdatx
Test Case Definition	↕
Rerun Test Iteration	
Cause of Failure	Failed criteria: Custom
▶ Simulation Metadata	

▼ TEST REQUIREMENTS ?

▶ ITERATION SETTINGS ?

▼ ERRORS ?

▼ LOGS ?

▼ DESCRIPTION ?

Double-click to edit

▼ CUSTOM CRITERIA LOGS ?

```

-----
VerificationFailed in custom criteria of sltest.testmanager.TestCase.

-----
Test Diagnostic:
Landing Lateral Speed: 0.51722 meters/second.

-----
Framework Diagnostic:
verifyLessThanOrEqual failed.
--> The value must be less than or equal to the maximum value.

Actual Value:
0.517217612807295
Maximum Value (Inclusive):
0.5000000000000000
-----
    
```



View: Requirements

Search

Update completed. Refer to Comments on Import1.

Index	ID	Summary	Implemented	Verified
Requireme...			<input type="checkbox"/>	<input type="checkbox"/>
Import1	Requirements!Sheet1	References to Requirements.xls (Shee...	<input type="checkbox"/>	<input type="checkbox"/>
1	L4-GNC-1	Coast Phase Requirements	<input type="checkbox"/>	<input type="checkbox"/>
2	L4-GNC-2	TCM Burns Requirements	<input type="checkbox"/>	<input type="checkbox"/>
3	L4-GNC-3	SRM Burn Phase Requirements	<input type="checkbox"/>	<input type="checkbox"/>
4	L4-GNC-4	Liquid Descent Requirement	<input type="checkbox"/>	<input type="checkbox"/>
5	L4-GNC-5	Landing/Touchdown Requirements	<input type="checkbox"/>	<input type="checkbox"/>
5...	L4-GNC-5.1	Landing/Touchdown Control Require...	<input type="checkbox"/>	<input type="checkbox"/>
5...	L4-GNC-5.2	Landing/Touchdown Knowledge Requ...	<input type="checkbox"/>	<input type="checkbox"/>
5...	L4-GNC-5.3	Landing/Touchdown Upper Limit Req...	<input type="checkbox"/>	<input type="checkbox"/>
	L4-GNC-5.3.1	Maximum Vertical Velocity at Touchd...	<input type="checkbox"/>	<input type="checkbox"/>
	L4-GNC-5.3.2	Maximum Lateral Velocity at Touchdo...	<input type="checkbox"/>	<input type="checkbox"/>
	L4-GNC-5.3.3	Maximum Pitch Angle at Touchdown	<input type="checkbox"/>	<input type="checkbox"/>
	L4-GNC-5.3.4	Maximum Angular Rate at Touchdown	<input type="checkbox"/>	<input type="checkbox"/>
	L4-GNC-5.3.5	Maximum Position Error	<input type="checkbox"/>	<input type="checkbox"/>
	L4-GNC-5.3.6	Maximum Lateral CM Offset	<input type="checkbox"/>	<input type="checkbox"/>

Requirement Interchange

Update

Export

Unlock all

Properties

Attribute Mapping

Links

Comments

Report Generation



Requirements Verification Testing Status

VIPER GN&C Model Based Design

Brian R. Jamison, James T. Kaidy

22-May-2019

VIPER Guidance, Navigation & Control Group
NASA, Marshall Space Flight Center

Table of Contents

Chapter 1. Introduction	1
1.1. Model-Based Design (MBD) for Guidance, Navigation, and Control (GNC)	1
1.2. Requirements Verification Testing	
Chapter 2. Demonstration Requirements Implementation and Verification Status	
Chapter 3. L4-GNC-5. Landing/Touchdown Requirements Set	
3.1. L4-GNC-5.3. Landing/Touchdown Upper Limit Requirements Subset	
3.1.1. L4-GNC-5.3.1. Max Vertical Velocity at Touchdown	
3.1.2. L4-GNC-5.3.2. Max Lateral Velocity at Touchdown	
3.1.3. L4-GNC-5.3.3. Max Pitch Angle at Touchdown	
3.1.4. L4-GNC-5.3.4. Max Angular Rate at Touchdown	
3.1.5. L4-GNC-5.3.5. Max Position Error	

Chapter 1. Introduction

1.1. Model-Based Design (MBD) for Guidance, Navigation, and Control (GNC)

"In MBD, a system model is at the center of the development process, from requirements development, through design, implementation, and testing" [1]. The VIPER GNC Team has been given the task of developing the GNC system for the VIPER Lunar Lander and has also been directed to implement an MBD approach to the design process. It is anticipated such an approach will become a fundamental aspect of designing GNC systems for future space vehicles developed at the Marshall Space Flight Center (MSFC).

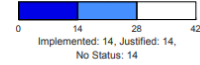
1.2. Requirements Verification Testing

It is expected a significant portion of the MBD approach to the design of the GNC system will involve testing intended to verify the design satisfies the requirements imposed on the system. As needed, versions of this document will be produced to summarize the status of such requirements verification testing.

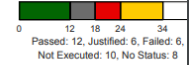
1

Chapter 2. Demonstration Requirements Implementation and Verification Status

Demonstration Implementation Status



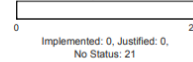
Demonstration Verification Status



2

Chapter 3. L4-GNC-5. Landing/Touchdown Requirements Set

L4-GNC-5 Implementation Status

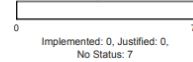


L4-GNC-5 Verification Status

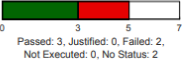


3.1. L4-GNC-5.3. Landing/Touchdown Upper Limit Requirements Subset

L4-GNC-5.3 Implementation Status



L4-GNC-5.3 Verification Status



3.1.1. L4-GNC-5.3.1. Max Vertical Velocity at Touchdown

Description:

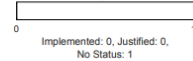
At touchdown the maximum vertical velocity magnitude shall be no greater than 2.5 meters/second.

Rationale:

Contact Limits

Status:

L4-GNC-5.3.1 Implementation Status



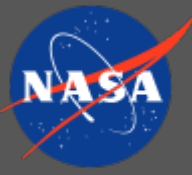
L4-GNC-5.3.1 Verification Status



3.1.2. L4-GNC-5.3.2. Max Lateral Velocity at Touchdown

Description:

3



Simulink Auto-Coding for MBD

- Auto-coding enforces coding standards automatically and consistently
- Utilizes Simulink Coder & Embedded Coder to generate C/C++ code directly from Simulink models
 - One interface for plant modeling, algorithm development, & code deployment
 - Access to MathWorks control toolboxes and other analysis tools

- Used by:

- NASA

- GLASS

- NEA-Scout

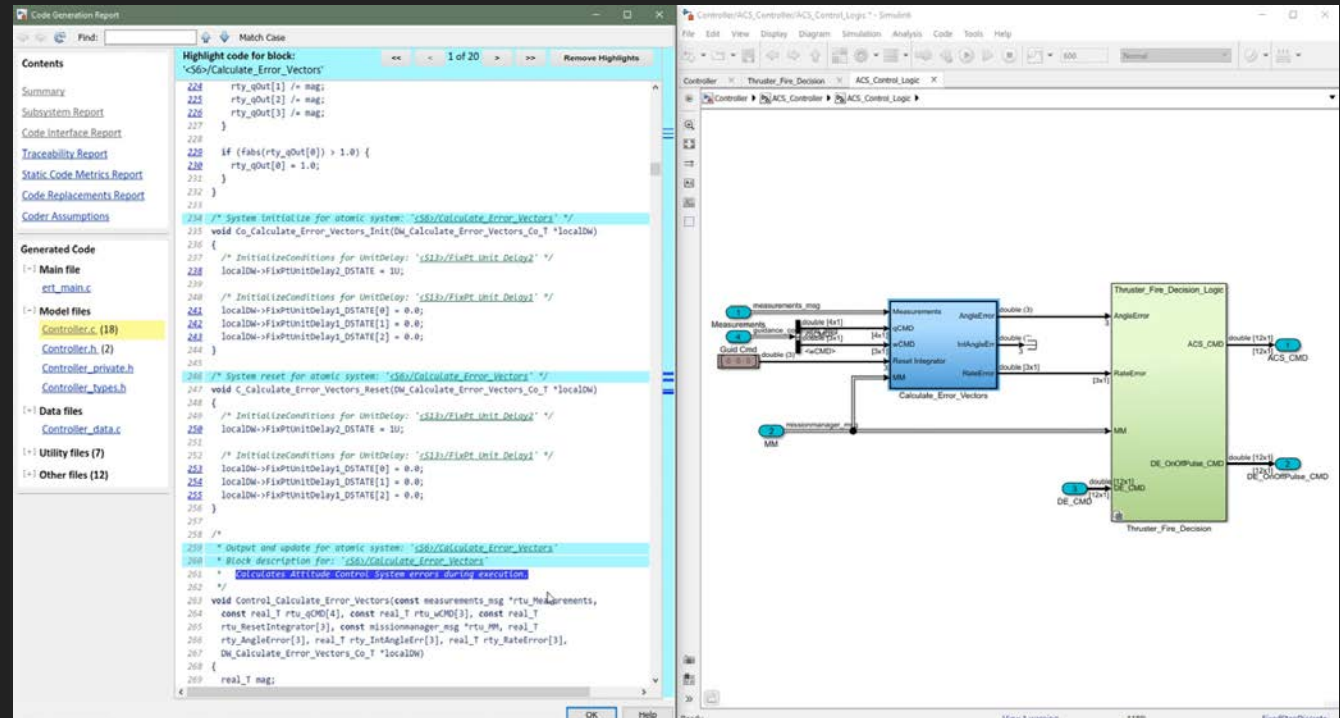
- Orion GN&C

- Goddard programs – PACE, JEDI

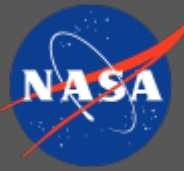
- APL

- Lockheed Martin

- Automotive Industry



Simulink Auto-Coding for MBD – Example



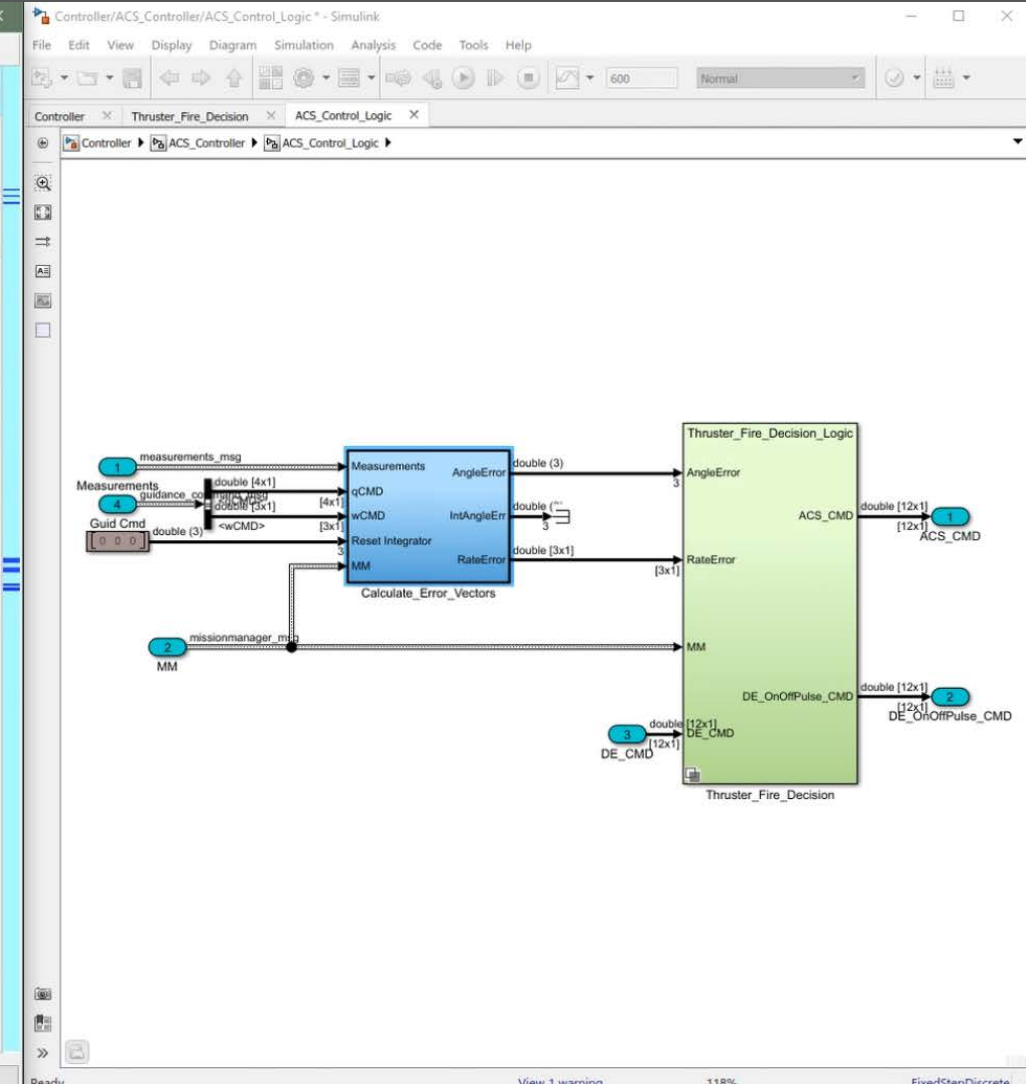
- Code & model linking
- Code comments
- Improved readability

```
Code Generation Report
Find:
Match Case
1 of 20
Remove Highlights

Contents
Summary
Subsystem Report
Code Interface Report
Traceability Report
Static Code Metrics Report
Code Replacements Report
Coder Assumptions

Generated Code
[-] Main file
ert_main.c
[-] Model files
Controller.c (18)
Controller.h (2)
Controller_private.h
Controller_types.h
[-] Data files
Controller_data.c
[+] Utility files (7)
[+] Other files (12)

Highlight code for block: '<S6>/Calculate_Error_Vectors'
224 rty_qOut[1] /= mag;
225 rty_qOut[2] /= mag;
226 rty_qOut[3] /= mag;
227 }
228
229 if (fabs(rty_qOut[0]) > 1.0) {
230 rty_qOut[0] = 1.0;
231 }
232 }
233
234 /* System initialize for atomic system: '<S6>/Calculate_Error_Vectors' */
235 void Co_Calculate_Error_Vectors_Init(DW_Calculate_Error_Vectors_Co_T *localDW)
236 {
237 /* InitializeConditions for UnitDelay: '<S13>/FixPt_Unit_Delay2' */
238 localDW->FixPtUnitDelay2_DSTATE = 1U;
239
240 /* InitializeConditions for UnitDelay: '<S13>/FixPt_Unit_Delay1' */
241 localDW->FixPtUnitDelay1_DSTATE[0] = 0.0;
242 localDW->FixPtUnitDelay1_DSTATE[1] = 0.0;
243 localDW->FixPtUnitDelay1_DSTATE[2] = 0.0;
244 }
245
246 /* System reset for atomic system: '<S6>/Calculate_Error_Vectors' */
247 void C_Calculate_Error_Vectors_Reset(DW_Calculate_Error_Vectors_Co_T *localDW)
248 {
249 /* InitializeConditions for UnitDelay: '<S13>/FixPt_Unit_Delay2' */
250 localDW->FixPtUnitDelay2_DSTATE = 1U;
251
252 /* InitializeConditions for UnitDelay: '<S13>/FixPt_Unit_Delay1' */
253 localDW->FixPtUnitDelay1_DSTATE[0] = 0.0;
254 localDW->FixPtUnitDelay1_DSTATE[1] = 0.0;
255 localDW->FixPtUnitDelay1_DSTATE[2] = 0.0;
256 }
257
258 /*
259 * Output and update for atomic system: '<S6>/Calculate_Error_Vectors'
260 * Block description for: '<S6>/Calculate_Error_Vectors'
261 * Calculates Attitude Control System errors during execution.
262 */
263 void Control_Calculate_Error_Vectors(const measurements_msg *rtu_Measurements,
264 const real_T rtu_qCMD[4], const real_T rtu_wCMD[3], const real_T
265 rty_ResetIntegrator[3], const missionmanager_msg *rtu_MM, real_T
266 rty_AngleError[3], real_T rty_IntAngleErr[3], real_T rty_RateError[3],
267 DW_Calculate_Error_Vectors_Co_T *localDW)
268 {
269 real_T mag;
```



Coverage Testing



- Ensures model/code are fully exercised
 - Used during unit testing
- Checked in the Simulink model and generated code
- Report links un-executed portions of the model and code
- Simplifies repair/justification
- Report provides metric about work remaining

The screenshot displays a web browser window showing a Simulink Coverage Report. The report is titled "sldemo_engine Coverage Report" and is located at "file:///D:/Landers/glass_core/CoverageReport.html".

Summary

Model Hierarchy/Complexity	Test 1 Decision	Test 1 Execution
1. sldemo_engine	9 75%	100%
2. ... Combustion	NA	100%
3. ... Compression	2 100%	100%
4. ... Drag Torque	NA	100%
5. ... Engine Dynamics	NA	100%
6. ... Throttle & Manifold	4 63%	100%
7. ... Intake Manifold	NA	100%
8. ... Throttle	2	
9. ... valve timing	2	
10. ... TDC and BDC detection		
11. ... positive edge to dual edge conversion	2	

Details

1. Model "sldemo_engine"

Child Systems: [Combustion](#)

Metric	Coverage
Cyclomatic Complexity	1
Decision	NA
Execution	NA

Full Coverage

Model Object

Gain block "rad/s to rpm"

Step block "Throttle Angle Profiles (d"

2. SubSystem block "Combustion"

[Justify or Exclude](#)

Parent: [/sldemo_engin](#)

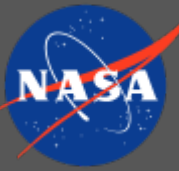
Coverage Report by Model

Top Model: mCFcn

	Complexity	Decision	Execution
TOTAL COVERAGE		33%	71%
1. ... Model(s)	0	--	100%
1-1. ... mCFcn	0	--	100%
2. ... Custom Code File(s)	3	33%	67%
2-1. ... hfoo.c	3	33%	67%

The report also includes a diagram of a C Function Caller block with input 'u1' and output 'out'. A block parameters window for 'C Function Caller' is shown, with the function name 'foo' selected. The corresponding C code is displayed, showing a switch statement for 'u1' with cases 0, 1, and default, all leading to 'break;'. The function returns 'u1'.

MBD Static Code Checking



- Static Code Check – Polyspace
 - Integrated with Simulink for traceability from the source code back to the original model
 - Looks for concurrency issues, security vulnerabilities, & runtime errors, including arithmetic overflow, buffer overrun, division by zero, out-of-bounds array access, and others
 - Ariane 5 failed (4 June 1996) due to overflow
 - Enforces coding guidelines
 - MISRA C, MISRA C++, JSF++, CERT® C, CERT® C++, etc.
 - Static code checks are typically required for FSW

```
static void pointer_arithmetic (void) {
    int array[100];
    int *p = array;
    int i;

    for (i = 0; i < 100; i++) {
        *p = 0;
        p++;
    }

    if (get_bus_status() > 0) {
        if (get_oil_pressure() > 0) {
            *p = 5;
        } else {
            i++;
        }
    }

    i = get_bus_status();

    if (i >= 0) {
        *(p - i) = 10;
    }
}
```

Green: reliable
safe pointer access

Red: faulty
out of bounds error

Gray: dead
unreachable code

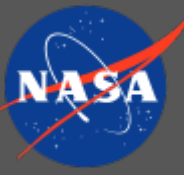
Orange: unproven
may be unsafe for some conditions

Purple: violation
MISRA-C/C++ or JSF++ code rules

Range data
tool tip

variable 'i' (int32): [0 .. 99]
assignment of 'i' (int32): [1 .. 100]

MBD Static Code Checking



- Static Code Check – Polyspace
 - Integrated with Simulink for traceability from the source code back to the original model
 - Looks for concurrency issues, security vulnerabilities, & runtime errors, including arithmetic overflow, buffer overrun, division by zero, out-of-bounds array access, and others
 - Ariane 5 failed (4 June 1996) due to overflow
 - Enforces coding guidelines
 - MISRA C, MISRA C++, JSF++, CERT® C, CERT® C++, etc.
 - Static code checks are typically required for FSW



Example: Traditional vs. MBD

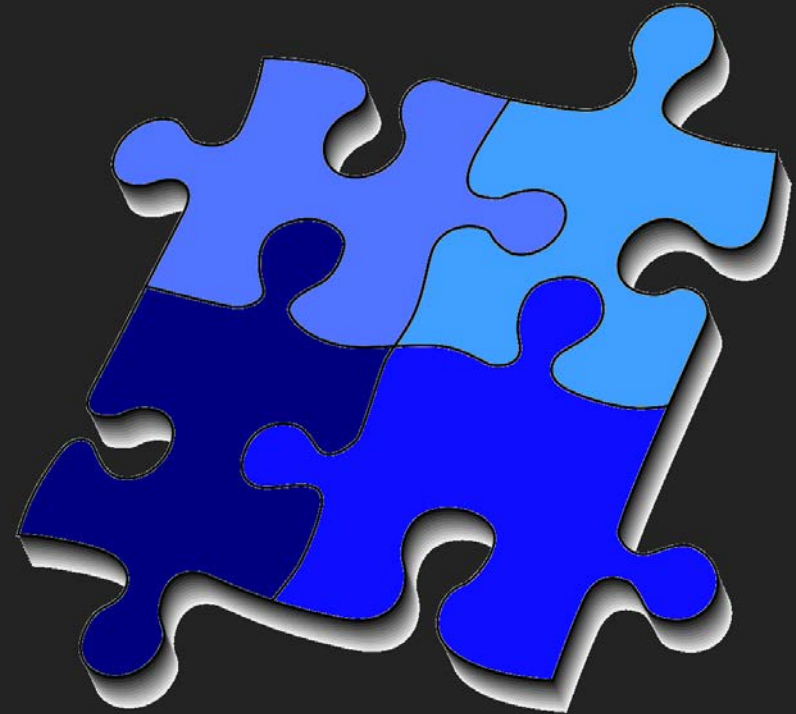


- Traditional (manual, labor intensive, error prone):
 - Near Earth Asteroid Scout (NEA-Scout) Project had ~33 GNC L4 requirements, verified through 4 major analysis packages
 - When changes were introduced, analyses and documentation were (manually) repeated to assess impacts to requirements
 - NEA-Scout relied on the FSW integrator (JPL) to run static code checks, involving manual trace-backs from the code to the model
- MBD (largely automated):
 - L4 requirements are checked within the model, and impacts to the design changes can be assessed with every execution of the model
 - When change is introduced, automated testing confirms requirements are satisfied
 - Static code checks are done by code developer prior to delivery to FSW integrator using Polyspace, which seamlessly links code violations with the source code and the model

A Synergistic Environment



- Automated processes
 - Testing, Auto-coding, report generation
- Testing
 - Tools such as Simulink Test are capable of exercising both the Simulink model and the generated code for requirements validation
- Auto-coding
- Reporting is part of the process
- All pieces work together to produce a highly automated, disciplined process

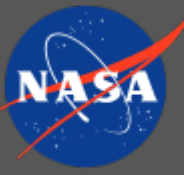


Summary



- A preliminary process has been established to generate quality GNC code using MBD tools from MathWorks (and Microsoft Excel)
- Initial discussions with the NASA MSFC Flight Software group have taken place to ensure processes work together
- Goals:
 - Increase development speed
 - Reduce manual tasks (e.g., testing, hand coding, report writing)
 - Traceability from requirements to model/code verification
 - Consistent quality

Image Sources



- NASA logo: https://commons.wikimedia.org/wiki/File:NASA_logo.svg
- LADEE image: https://www.nasa.gov/mission_pages/ladee/multimedia
- Dictionary image: <https://commons.wikimedia.org/wiki/File:Gnome-dictionary.svg>
- Simulink Requirements Editor: <https://www.mathworks.com/products/simulink-requirements/features.html#author-and-organize-requirements-in-simulink>
- Puzzle pieces image, modified from: https://commons.wikimedia.org/wiki/Jigsaw_puzzle#/media/File:Jigsaw.svg