حكومة دبي
GOVERNMENT OF DUBAI

مؤسسة الإمارات للعلوم والتقنية المتقدمة
EIAST

Program Partners:

SSL UC Berkeley

LASP Laboratory for Atmospheric and Space Physics University of Colorado **Boulder**

ASU ARIZONA STATE UNIVERSITY

Emirates Mars Mission (EMM):

System Engineering Report (SER)

SER-ADC-SPICE Unit Test

Agile # rev N/A

UNIT TEST RESULTS FOR SPICE INTERFACE MODEL

Prepared by	S. Piggott, ADCS

Status: Initial Test Results
Scope/Contents
This is a report documenting the results of the SPICE unit test created for the AVS Basilisk Simulation as part of the EMM project.

Rev:	**Change Description**	**By**
Draft	Initial Revision	S. Piggott

Contents

1 Introduction **1**

2 Test Design **1**

3 Test Results **2**

4 Test Coverage **2**

5 Conclusions **3**

1 Introduction

The SPICE Interface model in the AVS Basilisk simulation is used to generate information on what the overal universal time is (ex. UTC/UT1/etc) as well as ephemeris information for planetary bodies such as Mars, Earth, and the Sun. The time functionality of the model is required for the simulation because we must be able to input specific starting epochs as well as output accurate time tags in a format usable by others on the project. The planetary ephemeris information is required because we need to be able to compute the location of the various planetary bodies in order to simulate the effects of their gravity on the spacecraft.

2 Test Design

The unit test for the spice_interface module is located in:

`SimCode/environment/spice/UnitTest/SpiceUnitTest.py`

This unit test is designed to functionally test the simulation model outputs as well as get complete code path coverage. The test design is broken up into several parts:

1. Time Increment Check: The simulation is advanced at a fixed rate and all of the increments are checked to ensure that they are consistent.
2. Absolute Time Check: The simulation-computed GPS time at a specific UTC time is checked to ensure that it is correct with respect to the true value of GPS time for that UTC epoch.
3. Julian Date Time Check: The simulated Julian Date time is compared for the same UTC epoch used for the previous test.
4. Mars Position Check: The computed position of Mars in the inertial frame is compared against a value taken for Mars from the JPL Horizons ephemeris system for the same epoch time.
5. Earth Position Check: The computed position of Earth in the inertial frame is compared against a value taken for Earth from the JPL Horizons ephemeris system for the same epoch time.

6. Sun Position Check: The computed position of Sun in the inertial frame is compared against a value taken for Sun from the JPL Horizons ephemeris system for the same epoch time.

3 Test Results

1. Time Increment Check: The difference between the increments was checked to make sure that it was less than 1.0 microseconds which is less than our required time accuracy. Check successful.

2. Absolute Time Check: The difference between the truth GPS time at the UTC epoch (2016 June 16, 00:00:00.0 TDB) was within 1.0 microseconds of the simulated value. Check successsful.

3. Julian Date Time Check: The truth Julian date was within 0.1 seconds of the simulated Julian date which is near the precision of the Julian Date computation accuracy. Check successful.

4. Mars Position Check: The Position of Mars from the Horizons ephemeris system only agreed with the simulation output to 1626 meters. This is above the success criteria for the test. Check failed.

5. Earth Position Check: The Position of Earth from the Horizons ephemeris system only agreed with the simulation output to 1626 meters. This is above the success criteria for the test. Check failed.

6. Sun Position Check: The Position of the Sun from the Horizons ephemeris system was within 800 m of the simulated output. This meets the success criteria for the test although it is believed that there are issues with the Sun similar to the Mars/Earth tests. Check successful.

4 Test Coverage

The method coverage for all of the methods included in the spice_interface module are tabulated in Table 2

Table 2: Test Analysis Results

Method Name	Unit Test Coverage (%)	Runtime Self (%)	Runtime Children (%)
UpdateState	100.0	0.03	24.2
loadSpiceKernel	100.0	0.0	0.0
SelfInit	100.0	0.0	0.0
InitTimeData	100.0	0.0	0.0
ComputeGPSData	100.0	0.0	0.0
ComputePlanetData	100.0	0.18	13.26
SendOutputData	100.0	0.0	0.0

For all of the methods in the spice_interface modules, the code coverage percentage is 100% which meets our test requirements. The CPU usage drawn by the module is one of the highest percents in the entire test (24%). This is being almost entirely driven by external components (JPL's SPICE and boost). We could certainly look at building those modules with optimization to cut down the CPU usage, or even call this module at a lower rate with another module added to maintain the times/states at high rate.

5 Conclusions

The spice_interface module requires further work. We need to go and investigate why the planetary ephemeris computations are different from what we obtained from the Horizons system. In all actuality those results are fine, but we do need to go and figure out why the difference exists. It could be a hidden time error or other small effect that could impact our analysis going forward. However, it is close enough that we can definitely proceed with PDR analysis using the model as is.