Project Introduction

ES52 proposes to develop core Flight System (cFS) application source code to be run on a JPL-developed Sphinx Flight Computer to control a specific hardware component (i.e. valve, IMU, camera), providing flight software that will ultimately be used by MAV and Lander Tech projects. This work will allow ES52 a broad view of implementing cFS on a representative set of hardware and develop mission-specific applications that have benefits to upcoming projects. It will also give ES52 an opportunity to consider the development of more generic sensor applications that can be used across many future projects and missions. Such generic applications will allow MSFC to contribute to the catalog of capability available with cFS, reducing time and cost to develop software for future missions. cFS is widely used by other NASA Centers, Space Agencies around the world, and commercial space partners, but MSFC has not had the opportunity to work in-depth with cFS on a full flight system. Building this capability in-house benefits MSFC by positioning the Center to capture additional flight software work on current and future projects; allowing MSFC flight software engineers to take a more active role in helping private partners develop flight software for the CATALYST missions; and enabling MSFC to be smarter buyers of software from commercial vendors.

Anticipated Benefits

The Core Flight System (cFS), developed at Goddard Space Flight Center (GSFC), provides a platform independent, reusable software architecture, which allows projects to focus software development efforts on mission-specific needs in the form of a cFS application, a source code module developed to perform a specific task. These applications can communicate across the cFS architecture to carry out the necessary mission tasks, such as commanding, hardware control, and fault detection.

Several NASA missions, including the Lunar Reconnaissance Orbiter (LRO), Global Precipitation Measurement (GPM), and Magnetospheric Multiscale (MMS), have successfully used cFS. cFS has also been used by the European Space Agency and various universities and commercial companies. In addition, cFS has also been chosen as the flight software architecture for the upcoming Mars Ascent Vehicle (MAV) and multiple Lander Technology projects and proves promising for future lander, habitat, small satellite and other applications.

Expertise in the development of mission-specific flight software that will interface with the Core Flight System (cFS) architecture is critical to securing flight software development work at MSFC on future missions to support landers, ascent vehicles, habitats and other potential applications. Currently, MSFC ES52 is component-level software for Lunar CATALYST partners, but this work has been largely isolated from full-scale flight system development. While this has provided the team to gain some basic working knowledge of cFS in a generic Linux environment, it has not afforded the opportunity to work through all the design considerations when building a flight system using the cFS.
architecture or work with understanding its performance on relevant flight computers and operating systems. In this project, ES52 proposes to use the Sphinx Flight Computer (currently baselined for the Mars Ascent Vehicle mission) as a hardware platform to run cFS and develop mission-specific application source code that will control a hardware component (i.e. valve, IMU, camera) to be used by the MAV and Lander Tech projects.

cFS comes with a series of built-in ‘Core Applications’ that are ready to use out of the box. There are also a series of cFS applications that require customization based on the specific mission requirements. These applications provide a framework which can be extended and customized, which can save developers time.

A mission also requires the development of specific applications, shown in the figure as empty lots, since these applications must be created from the ground up. Software development time could be improved if more “framework” applications were available.

As such, ES52 proposes to survey the existing specific and high level cFS applications available across the Agency, and work with the MAV and Lander Tech projects to determine which area of the flight system needs to be addressed most and which application would provide the most benefits. ES52 will then use proven software development processes to define high level requirements for the application, and develop it with creating a framework application that will be beneficial to many future projects in mind.

The ES52 team will work with ES36 to get the MAV-provided hardware connected and running in a lab environment and to determine available sensors that can be used for development and testing. The teams will collaborate to install and run the cFS open source release on the Sphinx flight computer. The ES52 team will spend the majority of the time during this TIP project developing the cFS applications, with input and oversight from the MAV and Lander Tech projects. The application will likely focus on Guidance, Navigation, and Control sensors such as an Inertial Measurement Unit (IMU), but that is subject to change based on hardware availability and project needs.

While the work proposed is similar to what would normally be project-funded flight software development work, to date, MSFC has not been tasked with software development work on the MAV or Lander Tech projects outside of the Lunar CATALYST task currently funded through ED’s Strategic Initiatives. Completing this project will allow MSFC to work with the projects to prototype code that will anticipate future project needs while demonstrating MSFC’s expertise. In addition, MSFC would be able to consider cross-project needs and abstract project-specific code to begin building a portfolio of applications for future projects. By the end of the project, the overall technical objectives of the
work are for ES52 to:

1. Gain expertise implementing software using cFS in a more realistic flight environment on the Sphinx flight computer,
2. Apply this new expertise to develop flight software that will be delivered to the MAV and Lander Tech projects for future use, and
3. Consider ways to improve code for inclusion in a more generic portfolio of applications available at the start-up of a new project.

Consideration of TRL for a software project is difficult to assess; however, the open source release of cFS (with a few exceptions on specific applications) has flown successfully on numerous projects and can be considered a TRL of 9. Anytime additional applications are created, they would need to be reassessed independently. As such, at project beginning, the application code would be non-existent with no TRL. By the end of the project, ES52 will have demonstrated working source code in a lab environment with an integrated flight computer and hardware sensor, placing it at a TRL 4.

Primary U.S. Work Locations and Key Partners

Technology Areas

Primary:
- Communications, Navigation, and Orbital Debris Tracking and Characterization Systems (TA 5)

Other/Cross-cutting:
- Science Instruments, Observatories, and Sensor Systems (TA 8)
- Modeling, Simulation, Information Technology and Processing (TA 11)
- Ground and Launch Systems (TA 13)

Target Destinations
- Mars, Others Inside the Solar System, Outside the Solar System

Supported Mission Type
- Projected Mission (Pull)
### Organizations Performing Work

<table>
<thead>
<tr>
<th>Role</th>
<th>Type</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marshall Space Flight Center (MSFC)</td>
<td>Lead Organization</td>
<td>NASA Center, Huntsville, AL</td>
</tr>
<tr>
<td>Marshall Space Flight Center (MSFC)</td>
<td>Supporting Organization</td>
<td>NASA Center, Huntsville, AL</td>
</tr>
</tbody>
</table>

### Co-Funding Partners

<table>
<thead>
<tr>
<th>Type</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Office of the Chief Technologist (OCT)</td>
<td>NASA Office</td>
</tr>
</tbody>
</table>

### Primary U.S. Work Locations

- Alabama