Payload Manager Project Objectives

This project helps answer the common cFS question, “How should I design a cFS app to manage a science payload?” There are multiple approaches towards solving this problem. This project is not intended to provide a survey of designs, rather it shows a specific design that you can evaluate within the context of your situation.

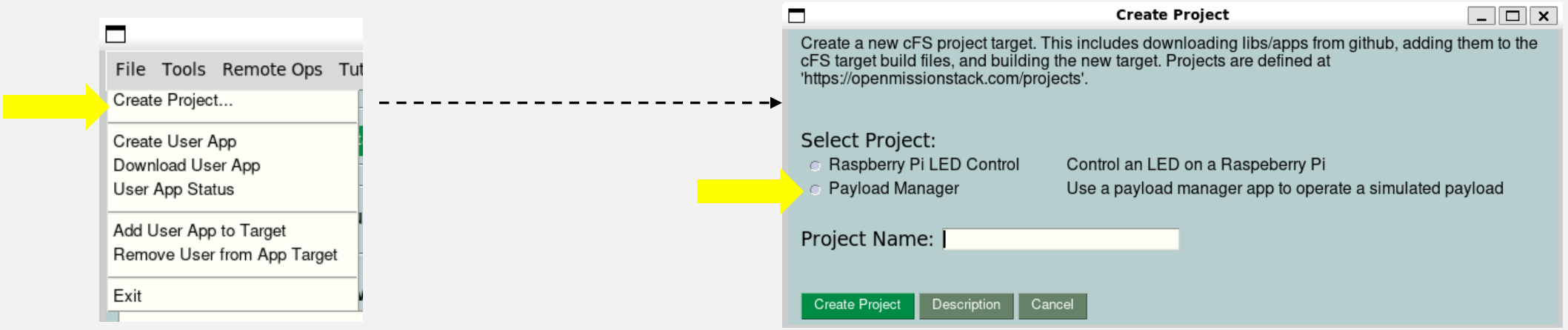
A secondary objective is to show how a cFS library and application can be used to simulate a payload. This effective strategy lets you run your payload manager app prior to having a test configuration with the target hardware. It’s also a flexible environment that lets you test error paths.

Detailed project instructions with videos can be found at

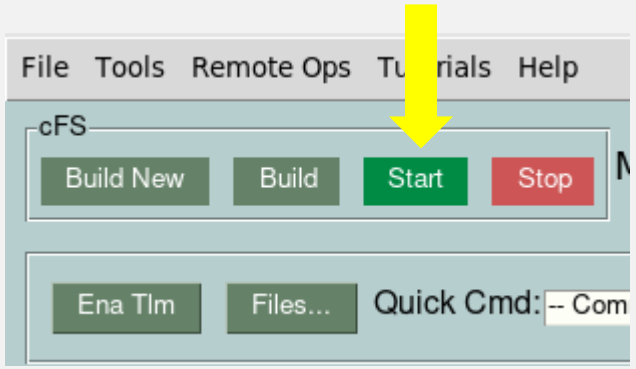
https://spacesteps.com/2024/10/12/cfs-payload-manager-app/

Software Installation

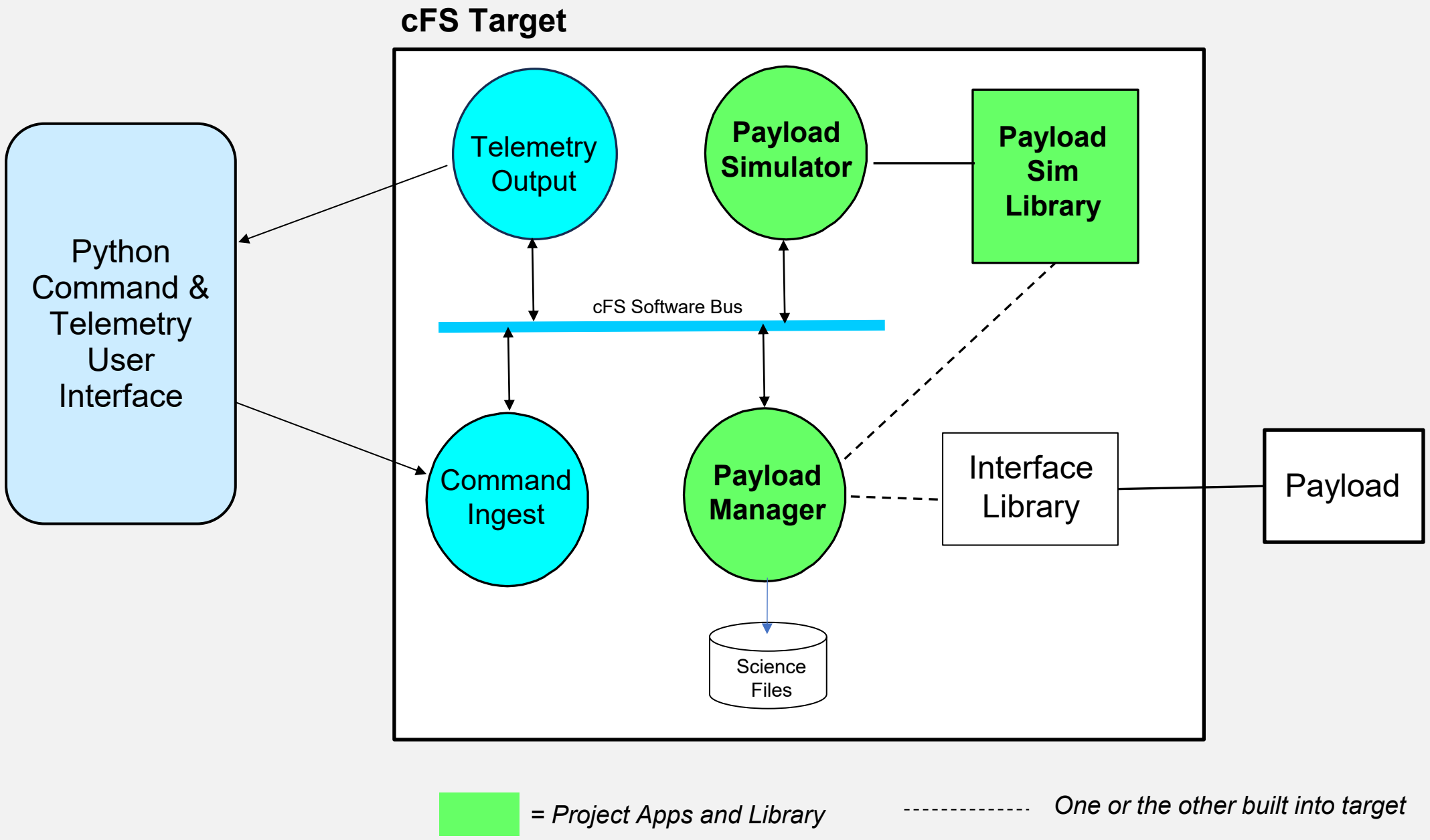
1. Create Payload Manager project using the Create Project tool



2. Start the cFS



Project Architecture



Library and Application Summary

PL_SIM_LIB

- Simulate payload power states, detector states, and detector science data
- Provides an interface to set and clear a detector fault. Science data is corrupted when the fault is present
- JSON initialization table defines number of 1Hz cycles for power initialization and detector reset

PL_SIM App

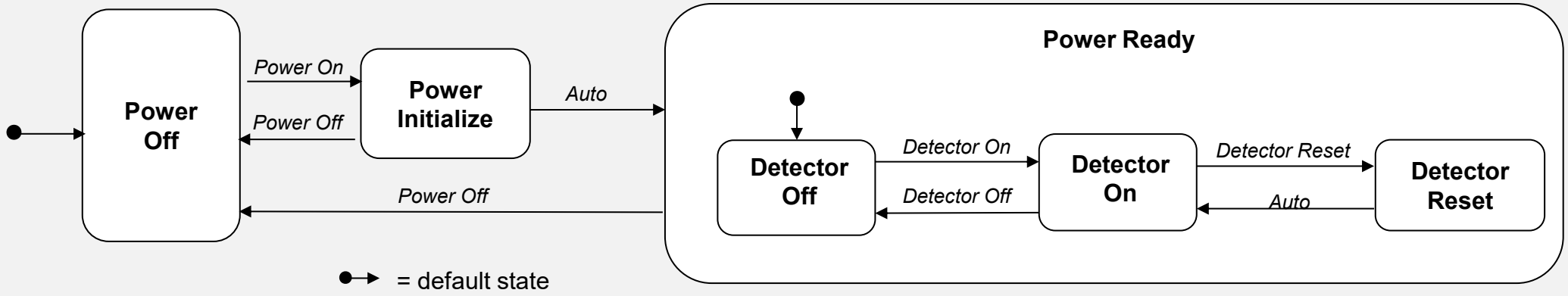
- Provides a ground command and telemetry interface to PL_SIM_LIB
- Command include: power on, power off, set fault, and clear fault

PL_MGR

- Manage the data interface to the payload and the creation of science data files
  - Reads detector data and writes images to files
- Commands to start and stop science data that turn on and off the detector, respectively
- JSON initialization table defines the science file path, base science filename and number of images per file

Simulated Payload: Power

- This state diagram shows the power and detector states



- The payload initializes into the *Power Off* state
- When a *Power On* command is received the payload transitions to the *Power Initialize* state where it waits for the number of seconds defined in PL_SIM_LIB's JSON ini table. Then it autonomously transitions to the *Power Ready* state
- In the *Power Ready* state, the detector can be turned on and off
- When the detector is on it produces image data
- The detector has a reset command that simulates an electronic reset that is used to clear a simulated fault

Simulated Payload: Detector

- A fictitious payload that has a science data detector
- The detector produces “images” and each image has ten rows of data
- Each row has ten pairs of text digits. The first digit in the pair is the row number and the second digit increments from 0..9 within a row. Here’s a complete image:

```
00010203040506070809
10111213141516171819
20212223242526272829
30313233343536373839
40414243444546474849
50515253545556575859
60616263646566676869
70717273747576777879
80818283848586878889
90919293949596979899
```

- An image is read out one row at a time

PL_SIM App

Commands

- Power On, Power Off
- Set Fault, Clear Fault

Telemetry

```
StatusTlm.Payload.ValidCmdCnt              : 1
StatusTlm.Payload.InvalidCmdCnt            : 0
StatusTlm.Payload.LibPowerState            : READY
StatusTlm.Payload.LibPowerInitCycleCnt     : 0
StatusTlm.Payload.LibDetectorResetCycleCnt: 0
StatusTlm.Payload.LibDetectorState         : ON
StatusTlm.Payload.LibDetectorFault         : FALSE
StatusTlm.Payload.LibDetectorReadoutRow    : 4
StatusTlm.Payload.LibDetectorImageCnt      : 1
```

PL_MGR App

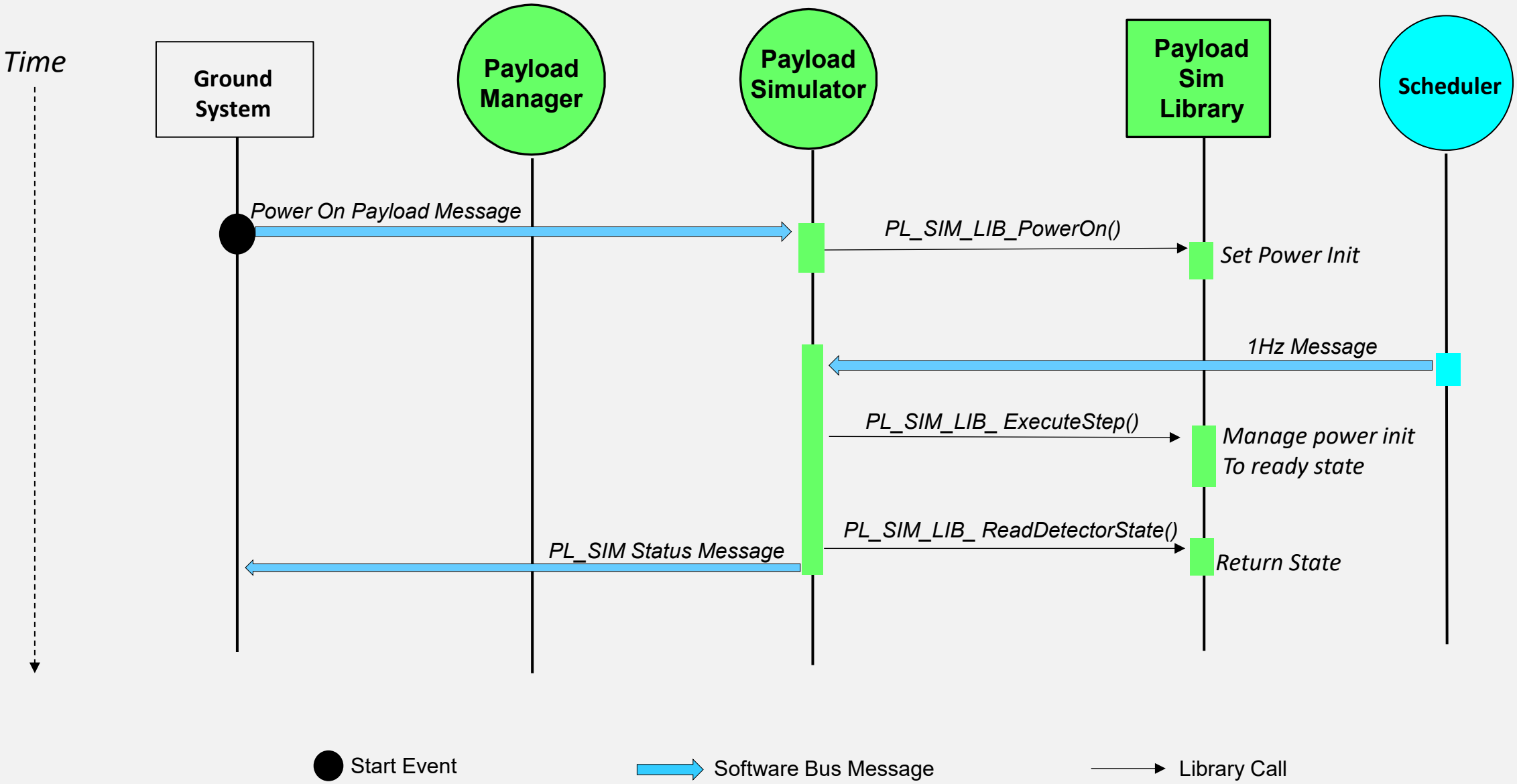
Commands

- Start Science, Stop Science
- Reset Detector
- Configure Science File Parameters

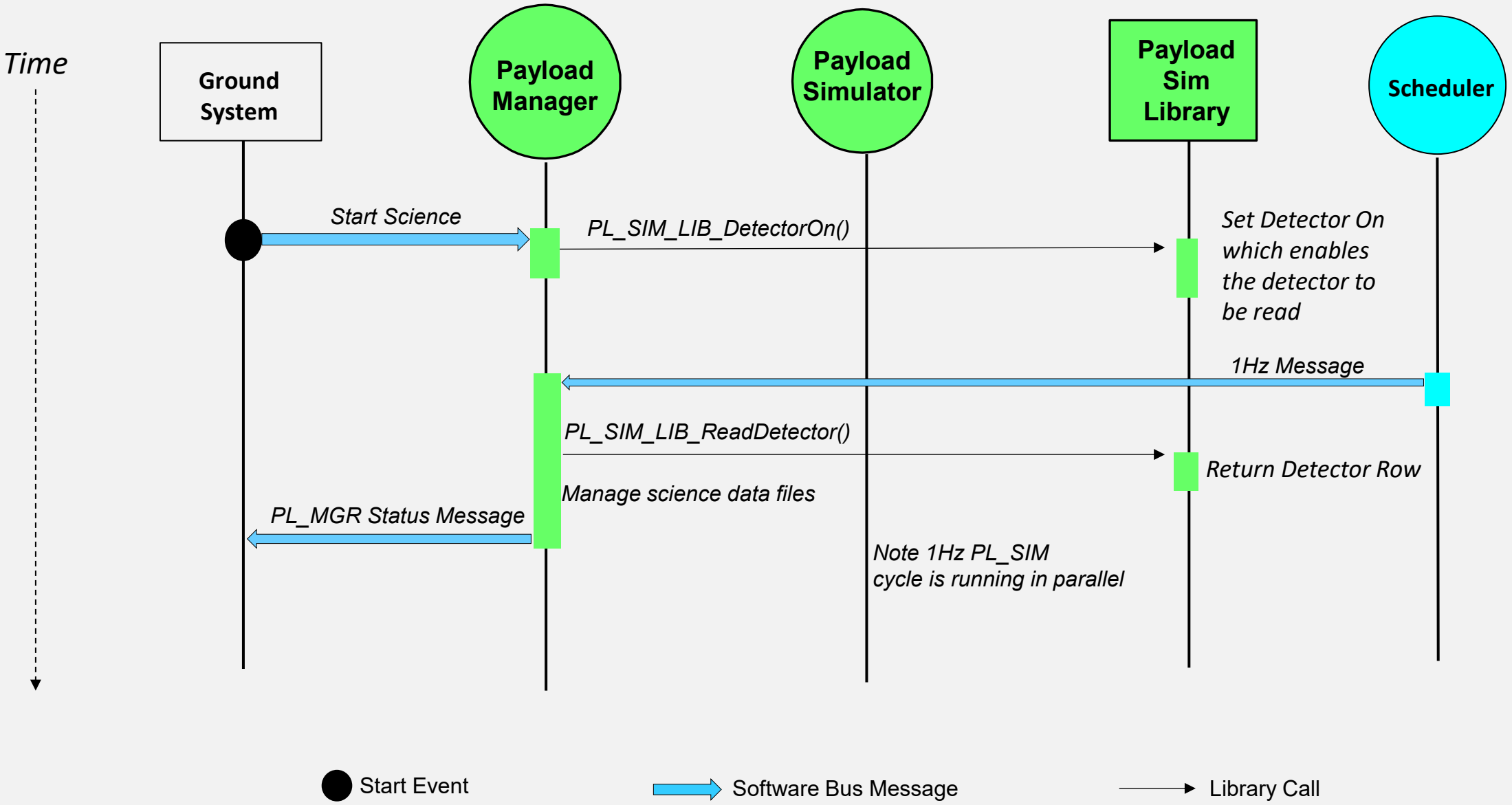
Telemetry

```
StatusTlm.Payload.ValidCmdCnt              : 1
StatusTlm.Payload.InvalidCmdCnt            : 0
StatusTlm.Payload.PayloadPowerState        : READY
StatusTlm.Payload.PayloadDetectorFault     : FALSE
StatusTlm.Payload.PayloadDetectorReadoutRow: 7
StatusTlm.Payload.PayloadDetectorImageCnt  : 4
StatusTlm.Payload.SciFileOpen              : TRUE
StatusTlm.Payload.SciFileImageCnt          : 1
StatusTlm.Payload.SciFilename              : /cf/pl_sci_003.txt
```

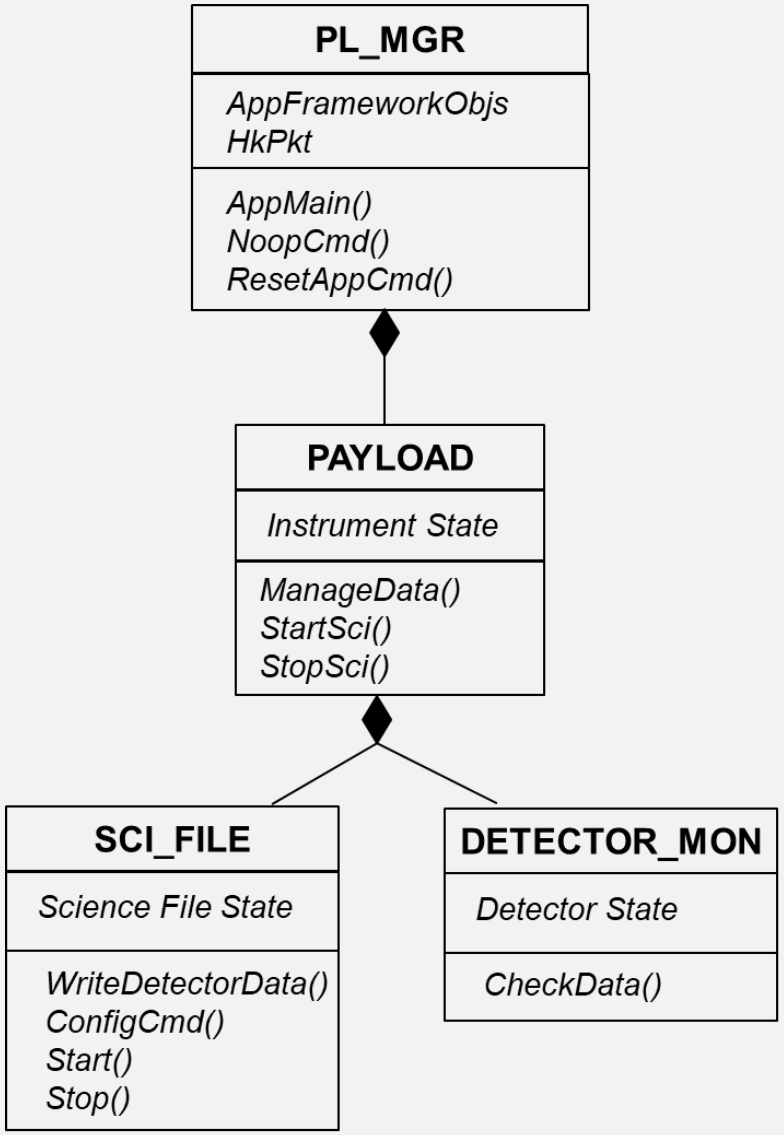
Power On Payload



Start Science



Payload Manager App Object Design



PL_MGR

- Manages app initialization, main runtime loop, and status telemetry
- Dispatches commands to objects

PAYLOAD

- Manage payload interface
- Has knowledge of the detector control and data interface
- Simulated vs actual payload conditional compilation flags should be limited to this object

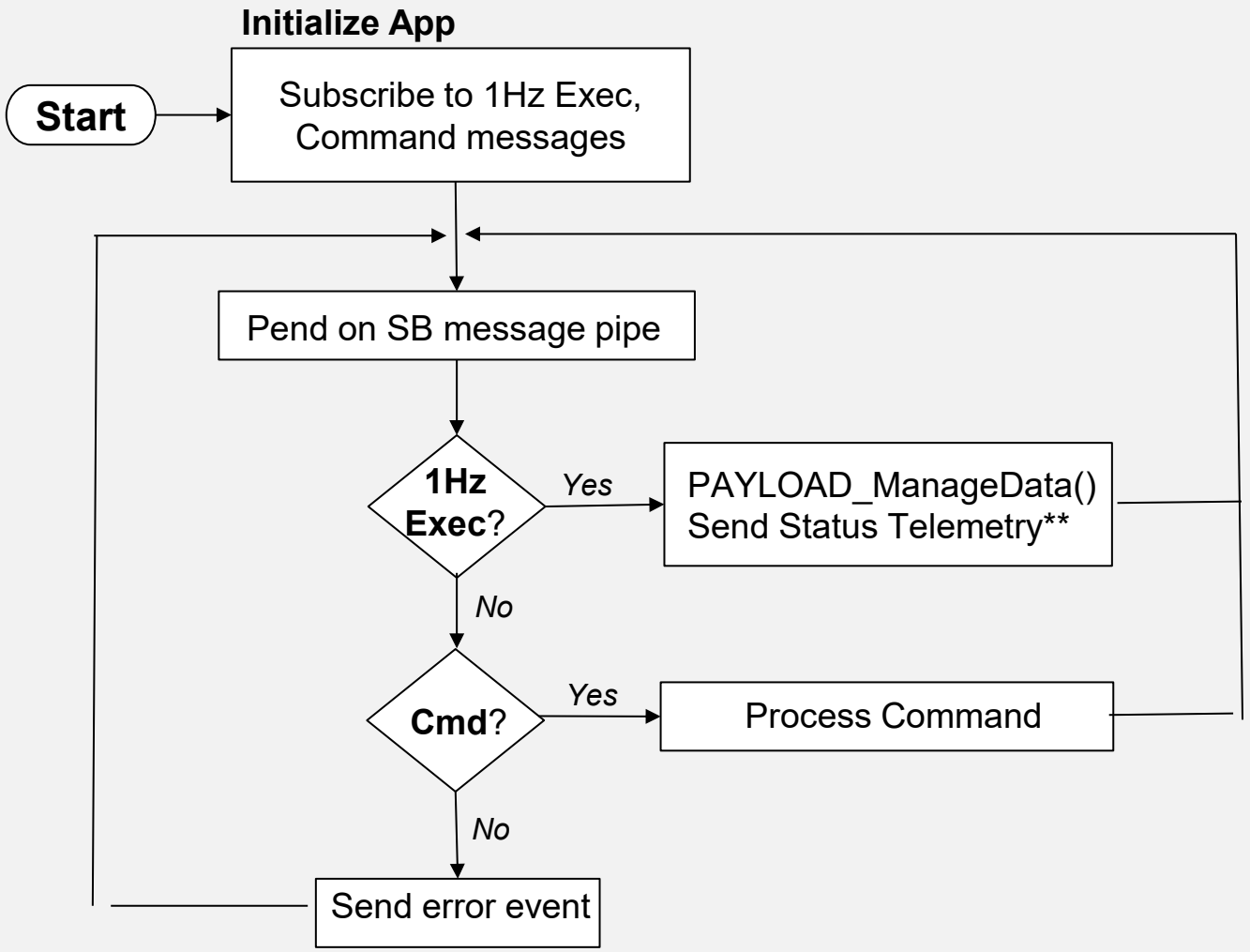
SCI_FILE

- Manage science data files
- Only needs to know detector science data format to minimize coupling

DETECTOR_MON

- Monitors detector status and data for faults

Payload Manager App Control Flow



** When instrument is on status telemetry is sent at the execution rate