A close up of a logo

Description automatically generated

**Hack-A-Sat (HAS)**

**Flatsat User’s Guide**

Version 1.0

24 June 2020

Contents

[Introduction 3](#_Toc44061190)

[Flatsat Architecture 4](#_Toc44061191)

[Flatsat Rules of Engagement 5](#_Toc44061192)

[C&DH Firmware Update Instructions 7](#_Toc44061193)

[Writing cFS image to RAM 7](#_Toc44061194)

[Writing cFS image to Flash 7](#_Toc44061195)

[HAS Flatsat Nominal Operations Description 8](#_Toc44061196)

[Air Bearing Operation 8](#_Toc44061197)

[Initialize Satellite 8](#_Toc44061198)

[Launch Cosmos 8](#_Toc44061199)

[Enable Telemetry 9](#_Toc44061200)

[Dip Switch Configuration 9](#_Toc44061201)

[EyasSat Configuration 9](#_Toc44061202)

[Satellite Operations 10](#_Toc44061203)

[Pointing Modes 10](#_Toc44061204)

[IMU Calibration 11](#_Toc44061205)

# Introduction

Congratulations on being selected as a finalist in the Hack-A-Sat Space (HAS) Security Challenge 2020! As a finalist you are being provided a HAS flatsat that shares many components with the units that will be used during the Finals competition. The flatsat is intended to allow you to familiarize yourself with its architecture and nominal operations prior to the final event. The flatsat is a modified commercially available training nano satellite produced by EyasSat, Inc. Their Gen 5 model includes many of the components used in real spacecraft and provide a laboratory satellite for experimentation and learning. Cromulence, LLC enhanced the commercial item with a more powerful Command and Data Handling (C&DH) subsystem and Payload module to provide a more suitable platform for a cyber Capture the Flag (CTF) competition. The EyasSat Gen 5 User’s Guide is included with your kit but is mostly useful for a general overview of the hardware and use of the provided power supplies and support equipment. Many of the described exercises will not work as written due to the Cromulence modifications. A representative HAS flatsat is shown below in Figure 1.

A circuit board

Description automatically generated

Figure 1: Hack-A-Sat (HAS) Flatsat

# Flatsat Architecture

As previously mentioned, the HAS flatsat started as a commercial item, but has been modified by the addition of hardware and software to provide a more capable CTF platform. All the original EyasSat hardware is intact, but the original C&DH board is moved into a secondary role behind the new C&DH assembly. The new C&DH is running the NASA open source cFS Flight Software on top of the RTEMS operating system. More information is available in the open source community regarding these products/software baselines. The *cFS Deployment Guide* and *cFE Application Developers Guide* are included as attachments. This document is intended to provide additional clarification specific to the HAS flatsat.

The C&DH software runs on a LEON3 processor implemented as a synthesizable VHDL model in an Artix-7 FPGA. The LEON3 was designed as a radiation hardened processor core implementing the Sparc V8 instruction set.

The schematic diagram and parts list for the C&DH assembly are provided as supplements to this document. A logical block diagram of the flatsat and its external interface is shown in Figure 2 and a block diagram of the cFS software is shown in Figure 3.

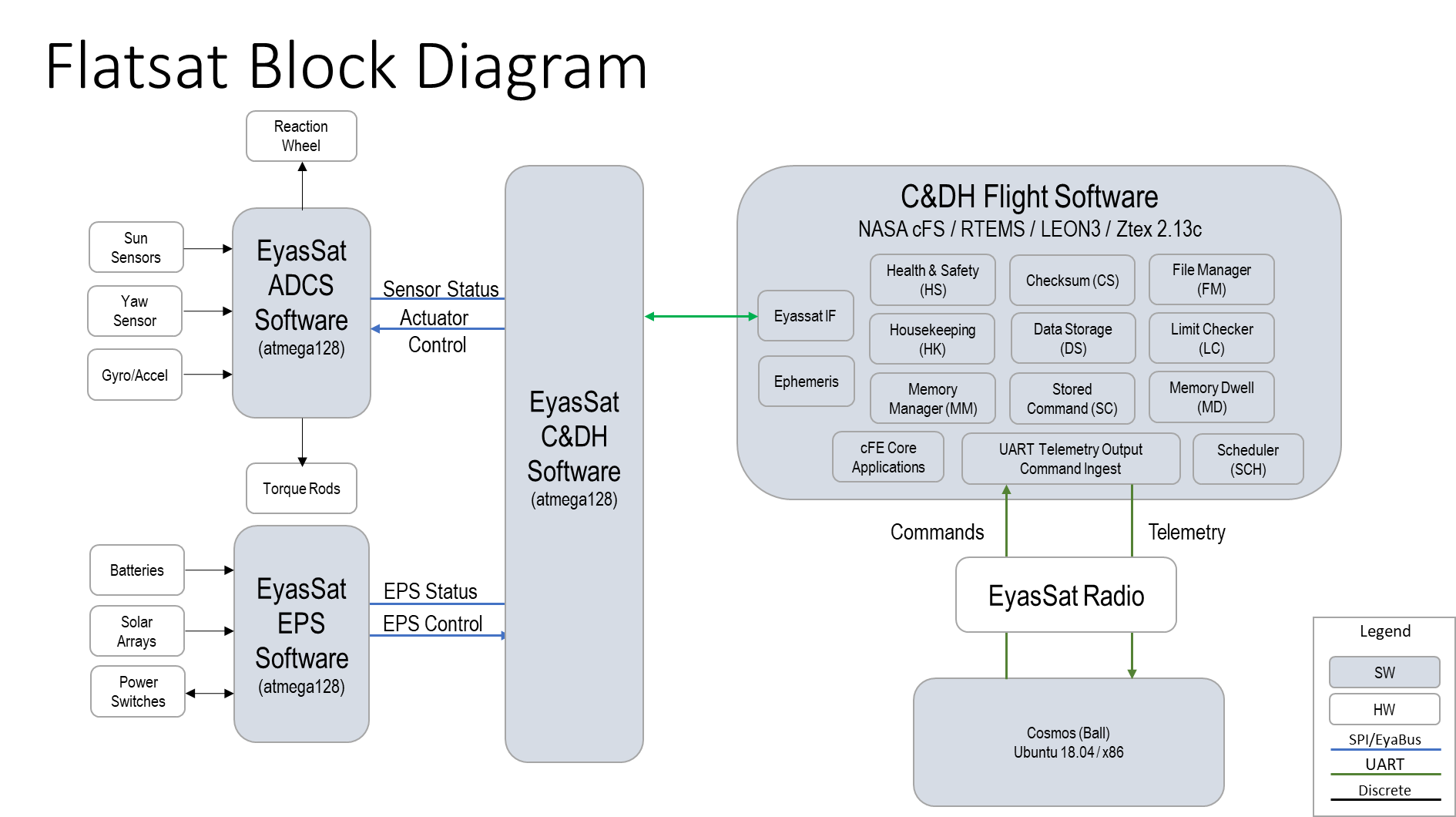


Figure 2: Flatsat Block Diagram

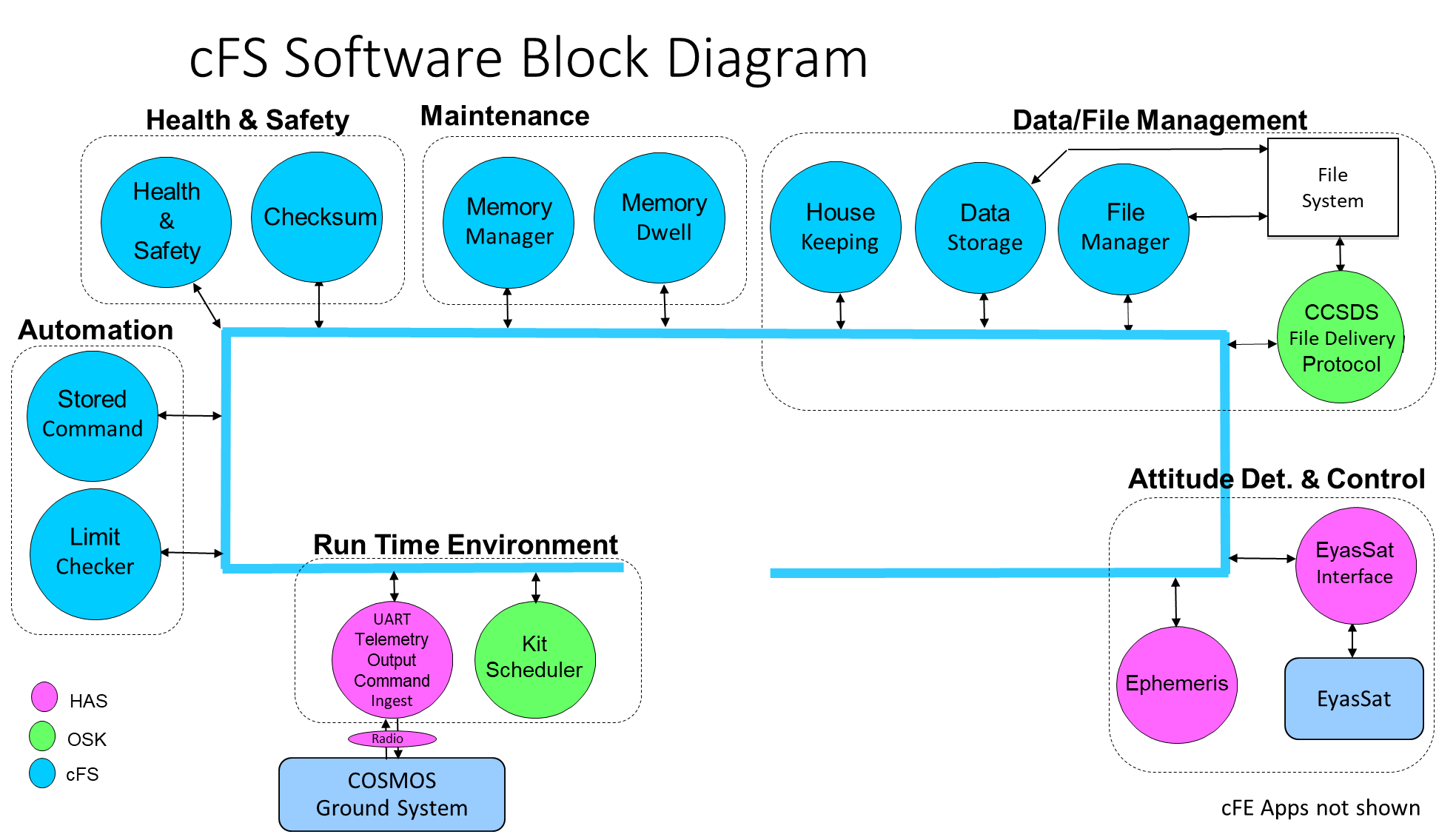


Figure 3: cFS Software Block Diagram

# Flatsat Rules of Engagement

While some reverse engineering of the flatsat is expected, it is intended that the contestants keep the flatsat in an operational state. In fact, keeping the flatsat in working condition will likely be advantageous in the competition because it can be used to dry-run operations planned for the game hardware. To that end, the organizers are providing the following guidance to the teams. This is not an all-inclusive list but is intended to provide some guidance regarding what should and should not be done with the flatsat.

* Do not manipulate the contents of the FPGA. Doing so will result in a configuration inconsistent with the contest architecture.
* The FPGA contents is not an intended target of the competition.
* Schematics and flatsat system block diagrams are being provided as part of this documentation package and are intended to reveal the parts of the FPGA relevant to the competition. There is no need to attempt to reverse engineer the hardware.
* A final FPGA bitstream will be provided prior to finals that is consistent with the architecture being utilized during the Finals competition.
* Utilize the power supplies provided within the EyasSat kit to power the flatsat. A 12V supply is provided for battery charging. A 9V supply is provided for powering the units without batteries. Power is provided to the unit via a DB-9 connector that is part of the EyasSat Ground Support unit. No additional power supplies are required for the units. The units can also be powered using the included lithium ion batteries. Damage induced by use of other power supplies is not the responsibility of the organizers.
* There is no need to pull firmware from the EyasSat PWR, ADCS, or C&DH boards. They are not an intended target of the competition. They are being utilized as a platform for sensor data and physical actuators. If the organizers update the firmware of the EyasSat prior to finals, binaries and instructions will be provided for performing firmware updates. Damage or loss of firmware from one of the EyasSat boards is not the responsibility of the organizers.
* Take care when operating the air bearing. Some guidance is provided in the included nominal operations description later in this document. In addition, care should be taken to avoid scratching the mating surfaces of the air bearing. Alcohol can be used to clean the air bearing before use. When not in use, the mating surfaces of the air bearing should be kept separate and protected.
* The RaspberryPi Zero daughterboard on the C&DH is not currently used. An SD card image and instructions will be provided before the final event.
* All digital logic is 3.3V (applies to UART interfaces for debug UART, system console UART, and telemetry Umbilical)

# C&DH Firmware Update Instructions

The HAS flatsat has shipped with a customized version of cFS written to flash memory. Prior to the Finals competition the organizers will provide an updated version of cFS that mirrors the final event architecture. In addition, the cFS image that ships with the flatsat is being provided independently to the teams to allow for re-flashing if necessary.

The cFS image can be written to flash memory or RAM on the C&DH board.

* Ensure that DIP switch 4 on SW1 is in the on position to enables flash access
* Connect FTDI cables to the C&DH board’s Debug UART on J3 using the pinout below. See the C&DH schematic for more details.
  + Debug UART – J3 Connector

|  |  |  |
| --- | --- | --- |
| Gnd – Pin 2 | Tx – Pin 15 | Rx – Pin 13 |

## Writing cFS image to RAM

$ > ~/grmon-eval-3.2.1/linux/bin64/grmon -uart /dev/ttyUSB0 -baud 115200

grmon3> load /opensatkit/cfs/build/exe/cpu3/core-cpu3.exe

grmon3> run

## Writing cFS image to Flash

$ > ~/grmon-eval-3.2.1/linux/bin64/grmon -uart /dev/ttyUSB0 -baud 115200

grmon3> spim flash detect

grmon3> spim flash erase

grmon3> spim flash load <path to prom>/core-cpu3.prom

grmon3> spim flash verify <path to prom>/core-cpu3.prom

grmon3> run

* cFS will run now run from flash on power-up

# HAS Flatsat Nominal Operations Description

## Air Bearing Operation

Along with the flatsat, an air bearing is provided so that teams can become familiar with the various pointing modes supported by the flatsat. The following items are relevant to the operation of the air bearing.

* Take care not to scratch the mating surfaces of the air bearing
* When storing the air bearing, store the mating surfaces separately or place a soft cloth between then
* Clean the mating surfaces of the air bearing between uses with alcohol
* Center the satellite on the air bearing and ensure that the satellite is balanced (air bearing puck is level)
* When placing the flatsat on the air bearing, align the small separation (SEP) switch on the bottom of the flatsat so that it fits into the groove in the air bearing puck.
* Connect an air compressor to the quick release connection and set the regulator to 1 bar or 15 PSI

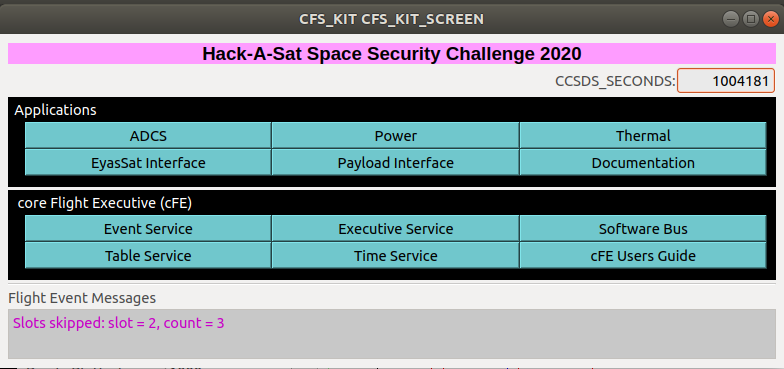
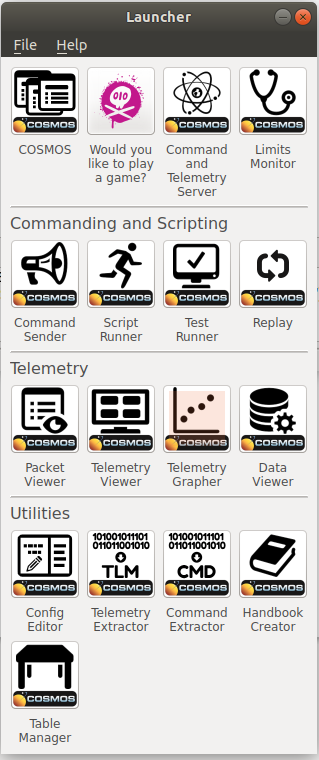
## Initialize Satellite

Place the flatsat carefully on the center of the operating air bearing, aligning it as described above. Install the Arming Plug in the DB-9 connector to enable flatsat battery operation.

## Launch Cosmos

A custom configured version of Cosmos is provided with the flatsat. It is tailored from the version provided within OpenSatKit. In addition, it includes support for the custom HAS applications. Installation instructions are included as part of the collection of files from the Cosmos repository. Ubuntu 16.04 or 18.04 are recommended.

Selecting the HAS logo, launches the command and telemetry server and several Cosmos applications.

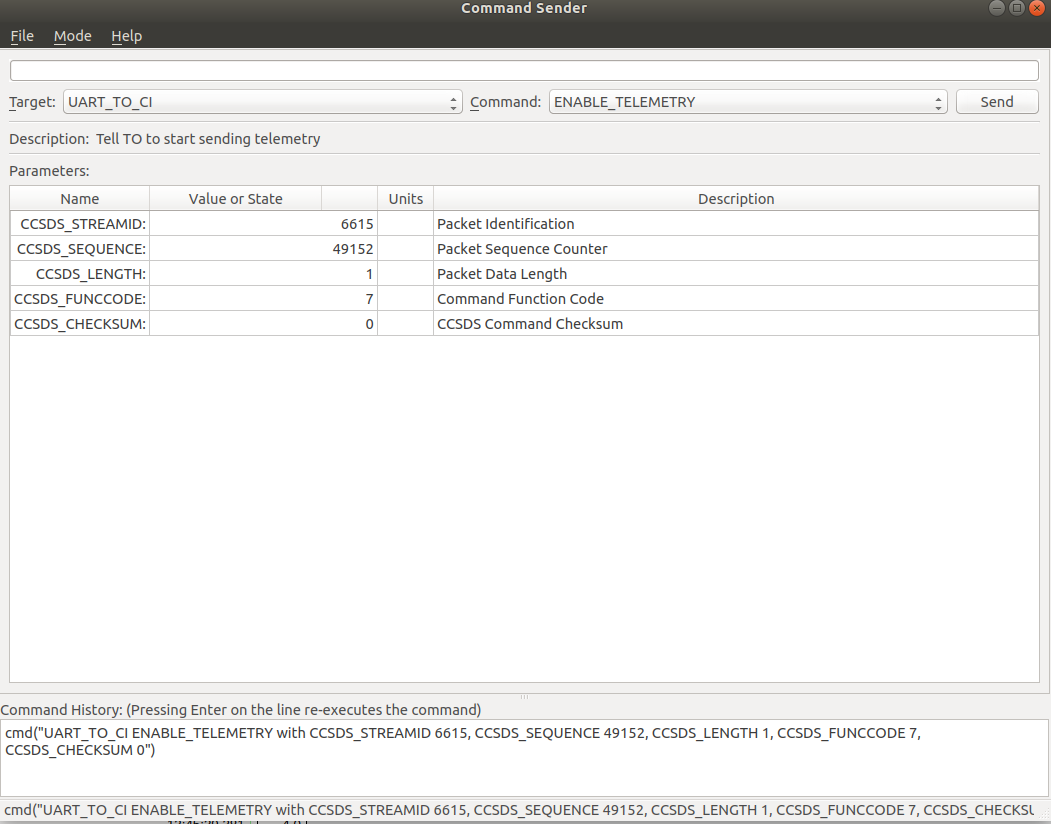


*Figure 1 - Cosmos Launcher and Main Screen*

## Enable Telemetry

Telemetry is available using the built-in radio, or via an umbilical that is selectable using S1 on the C&DH board. S1 in the off position is used to select the umbilical and the on position is used to select the built-in radio. Cosmos configuration may need to be updated for the appropriate serial port on your system. The enable telemetry command must be sent before the system sends telemetry.

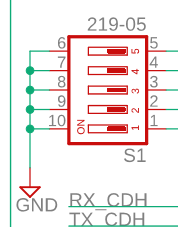
Umbilical physical connections are noted on the C&DH board schematic. All logic connections to the C&DH board are 3.3V.



*Figure 2 - Enable Telemetry Command*

## Dip Switch Configuration

As described above, switch one on S1 controls the configuration of the radio. For nominal operations, all switches should be kept in the off position, except for switch 4 which should be kept in the on position.



On

Umbilical/Radio Select

*Figure 3 - C&DH Board Dip Switch Configuration*

## EyasSat Configuration

The HAS cFS deployment includes a custom application that manages communicating with the EyasSat sensors and actuators. The application also manages initializing the EyasSat hardware into a nominal state. The cFS application issues the following commands on start-up. They can be issued to the EyasSat from Cosmos if the EyasSat boards have been power cycled via ground commands.

* EYASSAT\_IF PWR\_TLM  
  EYASSAT\_IF PWR\_3V
* EYASSAT\_IF PWR\_ADCS
* EYASSAT\_IF ENABLE\_IMU
* EYASSAT\_IF ADCS\_TLM

The cFS EyasSat interface application uses a JSON file to read in configuration values and issue commands to the EyasSat hardware upon startup. See Appendix A for an example JSON configuration file. This file resides within the eeprom directory on the LEON3 and is loaded by cFS upon startup.

## Satellite Operations

Most of the information regarding the nominal operations of the HAS flatsat can be found by studying the EyasSat User’s Manual and the various cFS documents. The pointing modes provided by the HAS flatsat have been customized specifically for the competition and are described below.

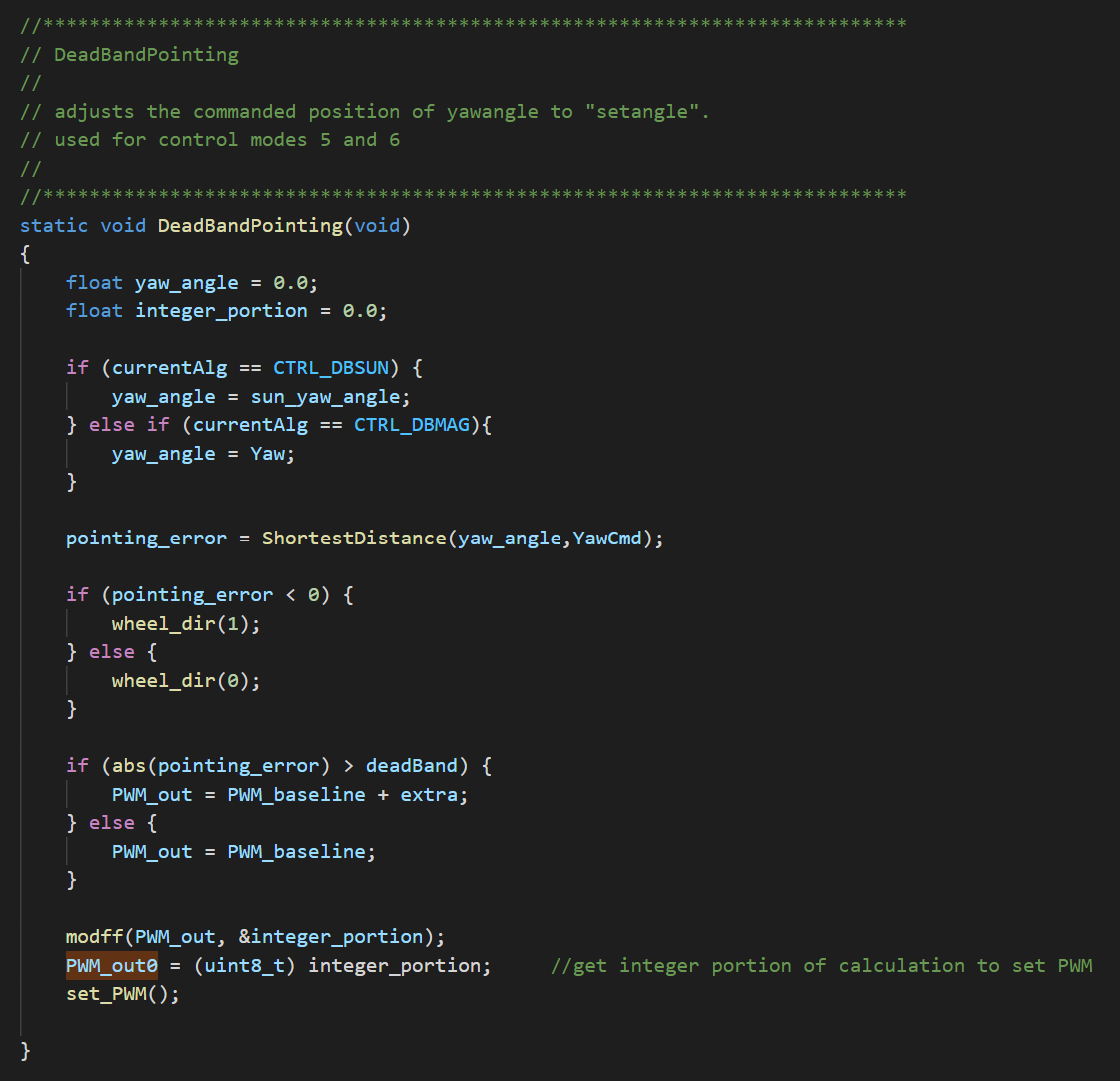
### Pointing Modes

The HAS flatsat has two sources of pointing information, a Sun sensor and a Magnetometer. The Sun sensor can be used to orient the satellite in relation to the brightest light source in proximity to the flatsat. The magnetometer can be used to orient the satellite in relation to the strongest magnetic field in proximity to the flatsat. Pointing modes are provided that utilize both sensors and allow the body of the flatsat to rotated about the yaw axis in relation to these two sensor sources. Two flavors of these pointing modes exist, Deadband and PD controller. These pointing modes will be described in more detail below.

#### Deadband

The deadband pointing mode is the simpler of the two modes. It can be configured using the following parameters.

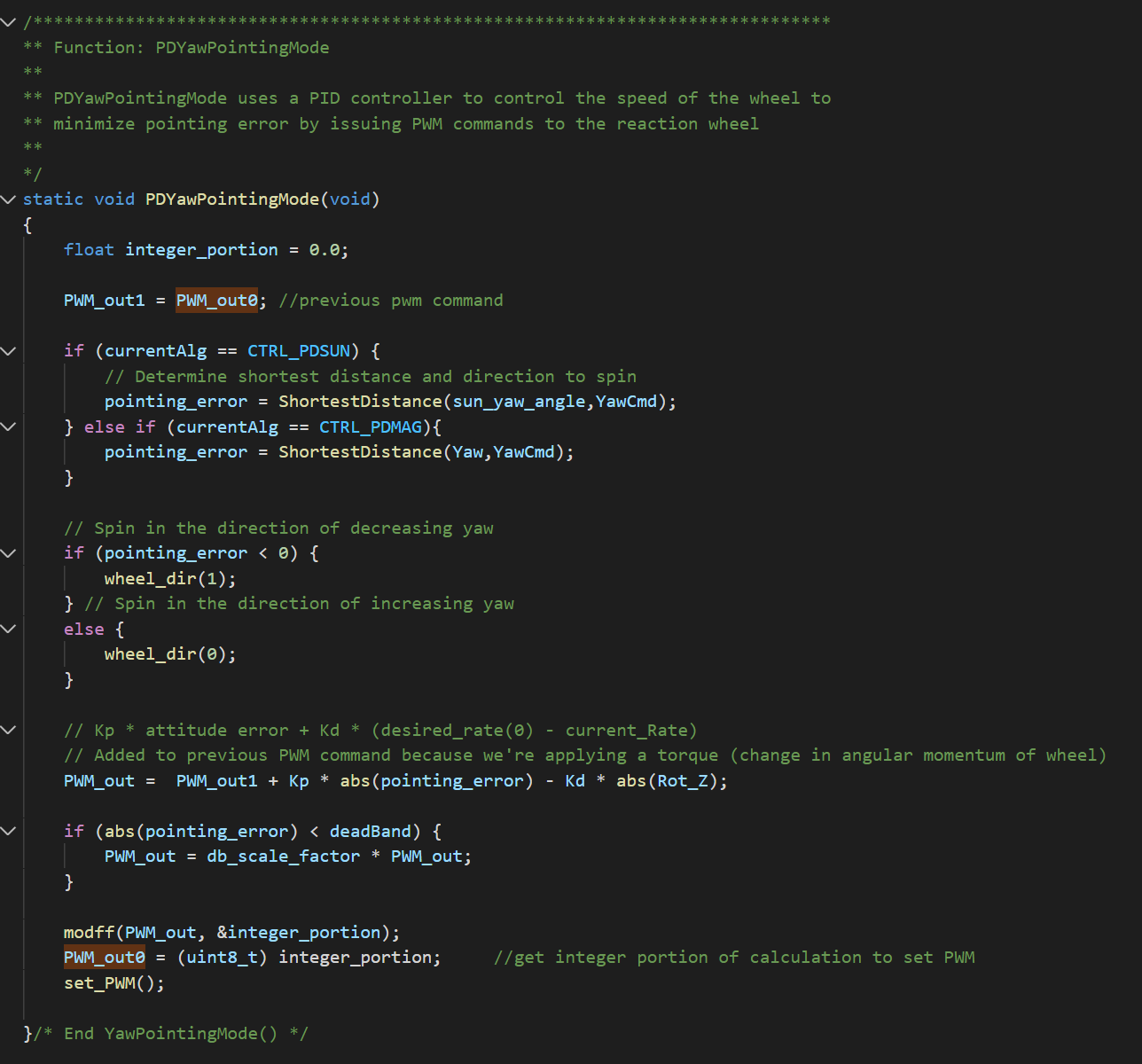
* **Deadband**: Angular region where less motor speed is utilized
* **Extra**: Additional motor speed that is applied outside of the deadband region
* **PWM\_Baseline**: Baseline PWM applied to the reaction wheel motor. Within the deadband, PWM\_Baseline is applied to the motor. Outside the deadband, PWM\_Baseline + Extra is applied to the motor. See the following code snippet below for more detail on the deadband pointing mode.



*Figure 4 - Deadband Pointing Mode Code Snippet*

#### PD Controller

The PD controller uses a conventional Proportional Derivative controller. It adjusts the PWM input to the reaction wheel motor to minimize pointing error to a fixed yaw offset and to zero angular rates about the Z axis of the EyasSat. It shares the use of the P\_CONST and D\_CONST configuration parameters with the wheel speed control mode. While switching between modes, updates to these parameters may be required. In addition to the typical parameters utilized by a PD controller a DEADBAND\_SCALE\_FACTOR has been included. The PWM output of the motor is scaled by this factor within the configured deadband.



*Figure 5 - PD Pointing Mode Code Snippet*

### IMU Calibration

For optimal operations, the IMU should be calibrated by applying fixed offsets. These offsets can be applied by modifying the Eyassat Configuration JSON file shown in Appendix A, or by sending discrete commands. It should be noted that the calibration offsets are sent to the EyasSat on startup, therefore if new calibration constants are being generated, the calibration offsets should be commanded to zero.

#### Magnetometer Calibration Offsets

Magnetometer calibration offsets are calculated by capturing magnetometer readings while rotating the EyasSat in the X, Y, and Z planes and capturing the magnetometer readings. The midpoint of each reading is the new calibration constant.

#### Gyro Calibration Offsets

Gyro calibration offsets are calculated by capturing gyro readings while the EyasSat is stationary and sitting on a rigid object. The midpoint of each reading is the new calibration constant.

Appendix A – EyasSat Configuration JSON File

{

"name": "Eyassat ADCS Control Table",

"description": "Attitude Control Parameters",

"Config": {

"Mode": 0,

"YawCmd": 0.0,

"PWM\_Baseline": 0

},

"DB\_Coef": {

"Deadband": 10.0,

"Deadband\_ScaleFactor": 0.1,

"Extra": 20.0

},

"PID\_Coef": {

"Kp": 5.0,

"Ki": 2.0,

"Kd": 2.0

},

"MagCal": {

"X": 47.74,

"Y": -9.24,

"Z": 502.32

},

"GyroCal": {

"X": 0.32,

"Y": 1.13,

"Z": 0.21

}

}