This document explains the function of the IHU, its schematic level design, its board level design, and its functional testing

IHU

Internal Housekeeping Unit

Revision: 1.0.1

Gabe Adamson

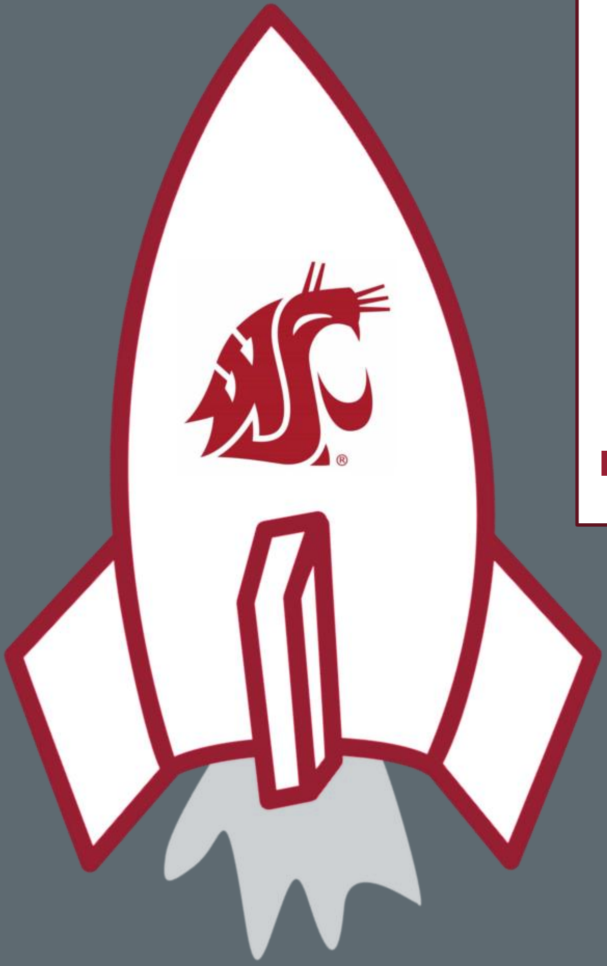

Table of Contents

1 Introduction

This document explains how the IHU will fulfil the following functions and conform to the following Requirements. This document refers to the Avionics Board version 1.1.

1.1 Function

The IHU is responsible for the following:

- Managing and processing all forms of data on the satellite
- Prepare and interpret communications to and from the ground
- Keep and distribute the satellite's time
- Enforce the current operation and mode of the satellite
- Perform periodic status inquiries on all subsystems

1.2 Requirements

The requirements and design requirements for the IHU can be found on GitHub.

2 Detailed Description

This section references the Avionics Board schematic. Page numbers will be listed and may have coordinates listed (number and letter combination found around the frame).

2.1 Functional Block Diagram

The block diagram can be found on the first page of the schematic.

2.1.1 IHU Microcontroller

The IHU is responsible for executing commands sent to the satellite from the ground[1], ensures subsystems are working properly,[2] processes packages from the ground, and prepares packages to be sent to the ground. The IHU also performs periodic inquiries on the status of all subsystems to ensure each subsystem is working as expected.[3]

2.1.2 C&DH SD Cards

The C&DH SD cards serve as redundant, non-volatile memory to store collected and processed information from all subsystems and payloads.[4] All collected data is stored in non-volatile memory in order to buffer the data and avoid corruption.[5]

2.1.3 C&DH ADCS

The Analog to Digital Converter, or ADC, converts the analog signals from the thermistors, which monitor temperatures of the components in the ADCS subsystem.

2.2 Schematic

2.2.1 IHU Microcontroller

The IHU microcontroller[6] (page 11) communicates with other microcontrollers and its ADCs through two I^2C buses, the In-Flight JTAG Reprogrammer (IFJR) through an SPI bus, and its two SD cards through a separate SPI bus. Reprogramming in orbit is achieved by the IFJR through a JTAG bus.

[1] Requirement CDH-003
[2] Requirement CDH-009
[3] Requirement CDH-010
[4] Requirement CDH-006
[5] Requirement CDH-008
[6] STM32L476RG

2.2.2 C&DH SD Cards

The two SD Cards[7] (page 12) serve as redundant non-volatile memory for the IHU microcontroller. The IHU microcontroller performs reads and writes through an SPI bus.

2.2.3 ADC

The Analog to Digital Converter[8], or ADC, (page 12) communicates with the IHU microcontroller using the required I^{2}C communication protocol. Thermistors are supplied with a reference voltage supplied by the ADC.

2.2.4 I^{2}C Bus

The IHU has two I^{2}C buses (page 11, B3). Both are for communicating with other microcontrollers, however the I^{2}C1 bus also has the C&DH ADC and a GPIO expander. The microcontrollers on the I^{2}C0 bus are ADCS and EPS. Payload uses the GPIO expander on I^{2}C1 to receive communications. On the IHU I^{2}C buses, the IHU microcontroller is the master.

2.2.4.1 ADC

There are two ADCs connected to the IHU microcontroller. One ADC is located on the Avionics Board, the other is off board. Its address is as follows:

ADC	I^{2}C Address	AS1	AS0
IHU ADC	0x23	H	L
Off Board	0x22	H	NC

2.2.4.2 GPIO Expander

There is a GPIO expander[9] connected to the IHU microcontroller. It is located off board.

Device	I^{2}C Address	A2	A1	A0
GPIO Expander	0x20	L	L	L

2.2.5 SPI Bus

The IHU has three SPI buses (page 11, B2, B3, B4). BUS_SPI is for communicating with the IFJR microcontroller, COM_SPI is for communicating with the Comms microcontroller, and IHU_SPI is for operating the C&DH SD cards.

[7] MOLEX_5025700893
[8] AD7291
[9] TCA9535

2.3 Board

The board shall be double layered with $2\ oz$ copper and ENIG finish. The board shall also conform to the dimensions specified by the CougSat Module Standard.

2.3.1 Layout Constraints

Unless specified in the following subsections, all signals shall use the default parameters below. Signals in the following subsections do not include their sense signals unless otherwise specified. Trace width can be broken if a trace needs to bottleneck down to a pin, the bottleneck shall be minimized.

Trace width:	$0.16mm$
Vias:	$\emptyset 0.3mm$, unlimited count
Separation:	$0.16mm$
Length:	unlimited

Devices with specific placement and routing considerations are called out on the schematic, see "CAD Note:"

Trace width:	$0.6mm$

2.3.1.1 JTAG - IHU_[JTCK, JTDI, JTDO, JTMS]

Length:	Each node shall be length matched $\pm 1.0mm$
Stubs:	$< 10.0mm$

2.3.1.2 I^2C - BUS_I2C0_[SDA, SCL], BUS_I2C1_[SDA, SCL, IRQ]

Length:	Each node shall be length matched $\pm 1.0mm$
Stubs:	$< 10.0mm$

2.3.1.3 SPI - BUS_SPI_[SCK, MISO, MOSI], COM_SPI_[SCK, MISO, MOSI], IHU_SPI_[SCK, MISO, MOSI],

Length:	Each node shall be length matched $\pm 1.0mm$
Stubs:	$< 10.0mm$

3 Testing

All tests shall be performed at room temperature and not under vacuum unless otherwise specified. If any modifications are performed, take note. Include enough information to understand circuit behavior and for others to replicate the results. Include any software written to execute the test and link it in the test notes section. Save all software, waveforms, etc. in a subfolder of the board's test folder for each test.

- Waveforms shall be captured whenever appropriate
- Have the event take fill the screen (for fast events, zoom in; for slow events, zoom out)
- Label each channel accurately

- Only have bandwidth limiting if necessary for the test (this applies to the oscilloscope and probe settings)
- If ringing or overshoot occurs, use a ground spring or differential probe

Common test instructions can be found on the wiki.

3.1 Before First Power-On Check

Configuration:

3.1.1 Test Instructions

Measure the resistance of various points in reference to *PGND* located at the backplane. When measuring in circuit resistances, flip the probes and take the lower value.

3.1.2 Test Data

Node	Resistance		Node	Resistance
3.3V-0			IHU_SPI_MOSI	
AVDD-1			IHU_SPI_MISO	
AVREF-2			BUS_I2C0_SCL	
BUS_SPI_SCK			BUS_I2C0_SDA	
BUS_SPI_MOSI			BUS_I2C1_SCL	
BUS_SPI_MISO			BUS_I2C1_SDA	
COM_SPI_SCK			IHU_JTCK	
COM_SPI_MOSI			IHU_JTDI	
COM_SPI_MISO			IHU_JTDO	
IHU_SPI_SCK			IHU_JTMS	

3.1.3 Test Notes

3.2 Command

Configuration:

This test evaluates the ability to command other microcontrollers.

3.2.1 Test Instructions

Request a reply from a device on both I^{2}C buses. Check if reply is received.

3.2.2 Test Data

Check if reply is received on BUS_I^{2}C0			
Device	Reply Received	Passing Criteria	Pass / Fail
ADCS microcontroller		*Reply Received*	

Check if reply is received on BUS_I^{2}C1			
Device	Reply Received	Passing Criteria	Pass / Fail
		Reply Received	

3.2.3 Test Notes

3.3 IFJR Communication

Configuration:

This test evaluates the ability to communicate with the IFJR.

3.3.1 Test Instructions

Request a reply from a IFJR on BUS_SPI. Check if reply is received.

3.3.2 Test Data

Check if reply is received on BUS_SPI			
Device	Reply Received	Passing Criteria	Pass / Fail
IFJR microcontroller		*Reply Received*	

3.3.3 Test Notes

3.4 Comms Communication

Configuration:

This test evaluates the ability to communicate with the Comms.

3.4.1 Test Instructions

Request a reply from Comms on COM_SPI. Check if reply is received.

3.4.2 Test Data

Check if reply is received on COM_SPI			
Device	Reply Received	Passing Criteria	Pass / Fail
Comms microcontroller		*Reply Received*	

3.4.3 Test Notes

3.5 Storage

Configuration:

This test evaluates the functionality of accessing storage.

3.5.1 Test Instructions

Write test package to both SD Cards and read back both test packages.

3.5.2 Test Data

Check if package sent is received when read			
Device	Package Received	Passing Criteria	Pass / Fail
IHU SD0		*Package Received*	
IHU SD1		*Package Received*	

3.6 Temperature Monitoring

Configuration:

This test evaluates the functionality of the temperature monitoring system.

3.6.1 Test Instructions

Heat thermistor location. Read resultant voltage through ADC.

3.6.2 Test Data

Check if temperature read by the ADC is the same as being supplied					
Device	Temperature Applied	Temperature Read	Error	Passing Criteria	Pass / Fail
IHU microcontroller				$Error < 2\,^{\circ}C$	
IFJR microcontroller				$Error < 2\,^{\circ}C$	
IHU SD0				$Error < 2\,^{\circ}C$	
IHU SD1				$Error < 2\,^{\circ}C$	
IFJR SD0				$Error < 2\,^{\circ}C$	
IFJR SD1				$Error < 2\,^{\circ}C$	

3.6.3 Test Notes

3.7 IHU Programming

Configuration:

This test evaluates the ability to program the IHU microcontroller

3.7.1 Test Instructions

Connect a SWD programmer to the SWD header and upload an image, validate the IHU is properly programmed. Connect a JTAG programmer to the backplane and upload an image, validate the IHU is properly programmed.

Note: Follow the programming instructions on the wiki.

3.7.2 Test Data

Program the IHU via SWD and JTAG, validate the IHU is properly programmed		
Programmer	Passing Criteria	Pass / Fail
SWD	*IHU properly programmed*	
JTAG	*IHU properly programmed*	

3.7.3 Test Notes