Subsystem Design Specification

University of Toronto Aerospace Team
Payload Electrical

August 18, 2025

Contents

1 Revision History

Revision / Date	Author	Change Log
v1.0 / Aug. 8, 2025	Coby Silayan	First release

2 Subteam Overview

2.1 System Introduction

The Payload Electrical team (PAY) is responsible for the electronics necessary to control the payload. The payload for the FINCH mission is a pushbroom hyperspectral imager that uses the Teledyne FLIR Tau camera. The fact that the payload is such already implies a couple of constraints: first, one spatial axis can be limited by the frame rate of the system. Second, the system must be able to deal with a lot (a lot relative to the memory of the flash of the MCU used) in a short amount of time which necessitates methods for rapid data transfer.

In order to fulfill the mission requirements, not only do the electrical interfaces need to be fast enough for data acquisition, the operations within the MCU must also be fast enough so as to not bottle neck the transfer of images to storage. The subsystem design specification exists to give an overview of the system: the electrical design, as well as methods to store the datacube.

2.2 The Hardware We're Working With

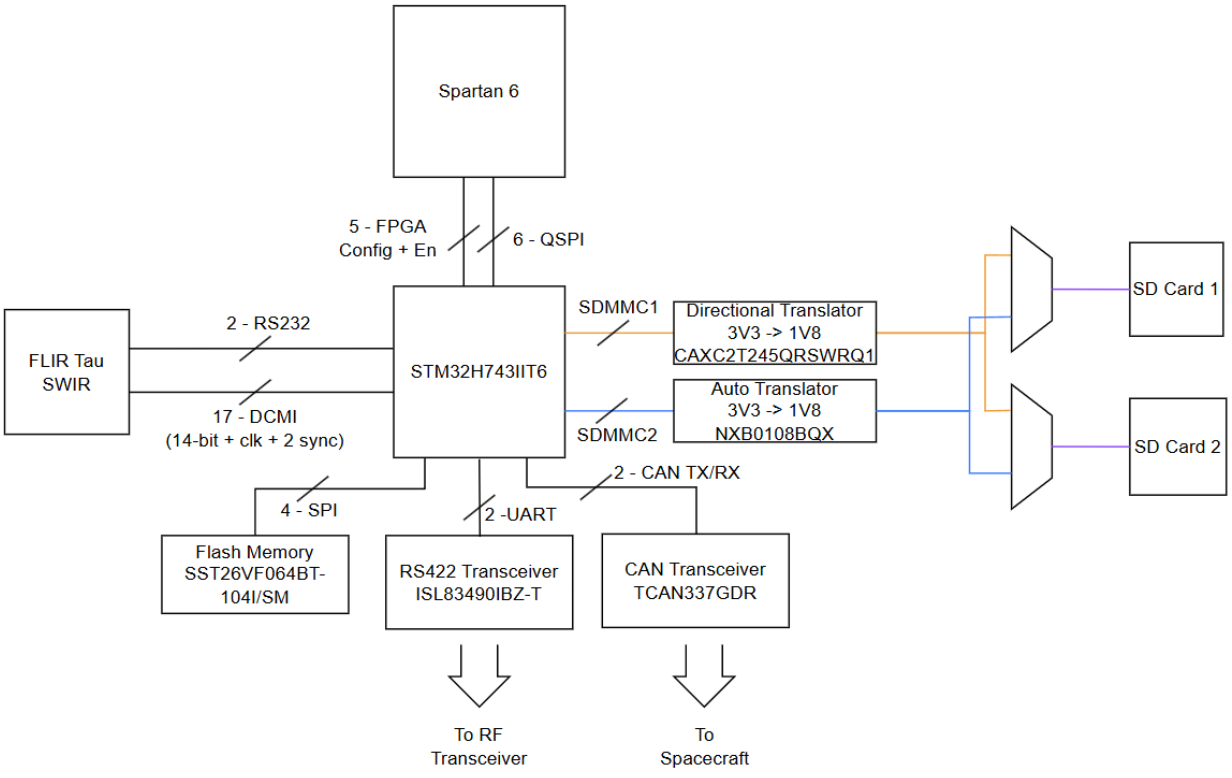

Figure 1: PAY v1.0 Block Diagram

2.2.1 STM32H743IIT6

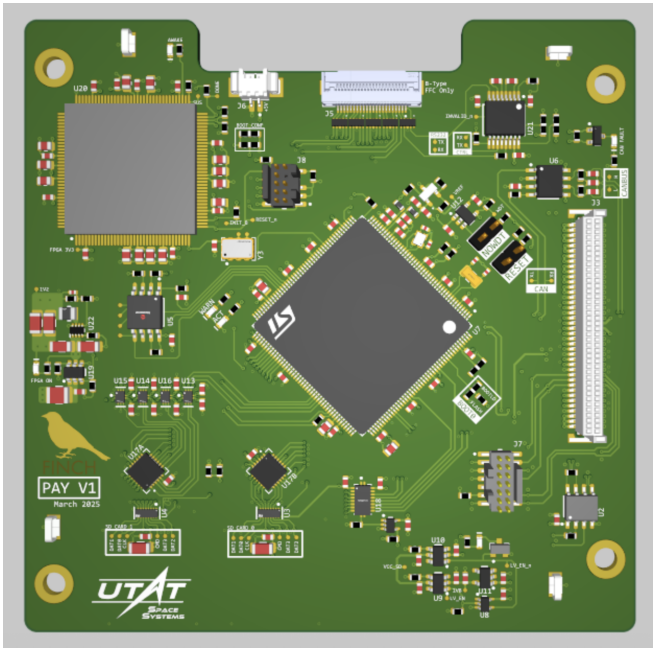

Figure 2: PAY v1.0

The Payload Electrical System utilizes an STM32H743IIT6 as the main controller. The H7 has the following specs:

Core	Single Core 32-bit ARM Cortex M7
Clocking	480MHz Max
On-chip Memory	2 MBytes Flash 1 MByte RAM 512kB Largest DMA Chunk
Peripherals	CAN I2C SPI SDMMC DCMI
Package	LQFP-176

Table 1: STM32H743IIT6 Relevant Electrical Specifications

2.2.2 FLIR TAU SWIR

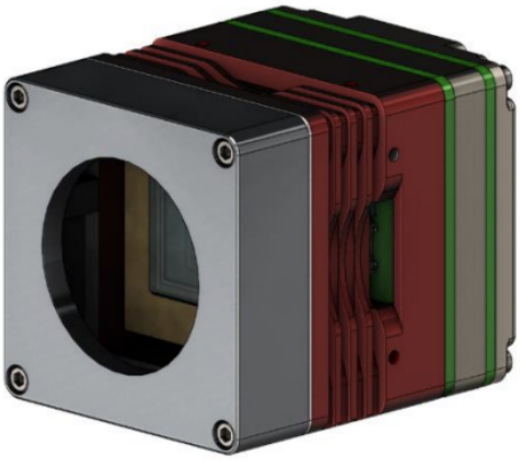

Figure 3: FLIR Tau SWIR

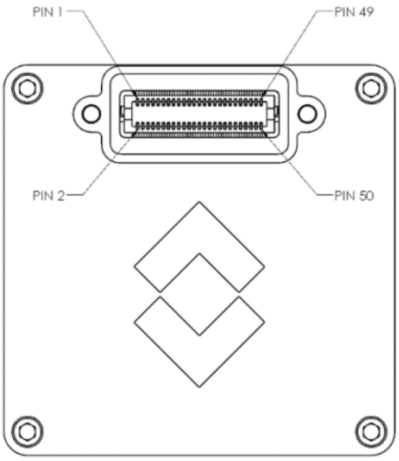

Figure 4: FLIR Tau Connector

The camera used is the FLIR Tau SWIR. A few important notes about the use of this camera: (1) A 256x320 subset of the detector is used rather than the 512x640 full frame. (2) image readout can be done while the camera is exposing[1]. (3) Commands are sent to the camera via RS232. (4) DCMI is used to readout the image with a pixel clock of 5.7048MHz. (5) The 14-bit pixels are stored in 2 bytes. (6) The team still needs to measure the QE of this camera.

For the 256x320 subset, the readout time is then given by

$$\frac{256 \times 320}{f_{CLKPIX}} \approx 14.36\text{ms} \tag{1}$$

Given this readout time, the max frame rate is then

$$1\text{ frame}/14.36\text{ms} \approx 69.63\text{fps} \tag{2}$$

Of course, the actual frame rate will be less considering also the time for commanding. 60fps should be a safe baseline for now. The docs claim a much faster frame rate of 120fps but we are assuming that a buffer within a camera is what makes that possible. Assuming the buffer, there must be a max number of frames that can be taken at 120fps and so this will not be suitable for the FINCH mission since imaging would last to the order of 30s.

The amount of storage needed for a single image is

$$\frac{256 \times 320 \times 16\text{bits}}{8\text{bits/byte}} = 163.84\text{kBytes} \tag{3}$$

Note that despite the camera being 14-bit, 16-bit is used for the calculation since 2 bytes will be

[1] *Tau SWIR Product Specification*. Oct. 2018. URL: `https://drive.google.com/file/d/1hDU8Dk9fygd2EyuG_VCbWGHtwMtvR_7e/view`.

used to store each pixel. This is just simpler on the firmware side. For seamless data transfer, the data transfer speed should be such that an image is saved to memory in the amount of time it takes to readout a frame.

$$\frac{\text{Image Size}}{\text{Readout Time}} \approx 11.12\text{MBytes/s} \tag{4}$$

As a rough estimate for the size of a datacube, assume 60fps for 30s.

$$163.84\text{kBytes} \times 60 \times 30 = 294.912\text{MBytes} \tag{5}$$

Resolution	512x640 pixels 14-bit analog resolution
Commanding Interface	RS232
Data Interface	14-bit DCMI 5.7048MHz Pixel Clock

Table 2: FLIR Tau SWIR Electrical Summary

2.2.3 SD Cards

The data is to be stored on SD cards onboard PAY. PAY has two SD cards that it can access using a MUX. Hardware is implemented so that the logic level can go down to 1.8V to enable greater than 25 MBytes/s transfer speed to the SD card (UHS mode). More details on the SD cards are provided in Section 7.

2.2.4 Spartan 6 FPGA

Yes, there is an FPGA onboard. Initially, the FPGA was on the board for compression which recently became a "Should" requirement. Currently, the only pins broken out of the FPGA are the QSPI lines to the STM32H7. The FPGA does not have any connection to the camera.

3 Applicable Documents and Standards

3.1 General Documents

- STM32H742xI/G STM32H743xI/G Datasheet
- Master Connection Sheet
- *Tau SWIR Product Specification*. Oct. 2018. URL: https://drive.google.com/file/d/1hDU8Dk9fygd2EyuG_VCbWGHtwMtvR_7e/view
- *Tau SWIR ISC1202 Software Interface Description Document*. Oct. 2018. URL: https://drive.google.com/file/d/1tCeISX1MTPkfJr_OYQdCwi8xV5pYETpT/view
- *Tau SWIR ISC1202 User's Guide*. Oct. 2018. URL: https://f.hubspotusercontent10.net/hubfs/20335613/tau-swir-isc1202-user-guide.pdf

3.2 Specifications, Standards, and Handbooks

- Spacecraft Grounding: NASA-HDBK-4001

4 Subsystem Requirements

4.1 Electrical System Requirements

From the FINCH-Spacecraft-ElectricalSystem, requirements for the existence of OBC and PAY are derived. Seen in Table 3 are also other higher-lever requirements of the system including the common bus to connect OBC, PAY, and the board for the Electrical Power System (EPS).

Req. ID	Description	Parent Req.	Verification Method
FINCH-Spacecraft-ElectricalSystem	The electrical system shall provide the necessary electrical functionality for the mission to function.	FINCH-Mission-Objective	Demonstration
FINCH-OBC-ControlAndOps	The OBC shall control the modes of operation of the satellite.	FINCH-Spacecraft-ElectricalSystem	Demonstration
FINCH-Payload Controller-PayloadOps	The Payload Controller shall support necessary operations for executing the mission of the payload	FINCH-Spacecraft-ElectricalSystem	Demonstration
FINCH-Spacecraft-CommonBus	The spacecraft shall utilize a common bus which includes the lines for power and communication between OBC, EPS, and PAY.	FINCH-Spacecraft-ElectricalSystem	Test
FINCH-Spacecraft-CANBus	The spacecraft shall use CAN Bus for communication between nodes on the electrical system.	FINCH-Spacecraft-CommonBus	Test
FINCH-Spacecraft-Electrical Grounding	The grounding system for the spacecraft shall include a separate chassis and signal ground.	FINCH-Spacecraft-ElectricalSystem	Analysis
FINCH-Spacecraft-Electrical Soldering Standard	Electrical components shall be soldered in accordance to IPC Type 3 or equivalent	FINCH-Spacecraft-ElectricalSystem	Inspection

Table 3: Electrical System Requirements

4.2 PAY Requirements

Notable is the compression requirement which has only become a "should" recently after PAY was designed. The reason why the requirement suddenly became a "should" comes from a spec from optics saying that the whole sensor will not be used. Initial requirement for compression assumed the whole sensor would be used.

Req. ID	Description	Parent Req.	Verification Method
FINCH-PayloadController-CameraControl	The Payload Controller shall control the camera module	FINCH-PayloadController-PayloadOps	Test, Analysis
FINCH-PayloadController-Storage	The Payload Controller shall store the images taken from the camera.	FINCH-PayloadController-Storage	Test
FINCH-PayloadController-ImagingStorageSpeed	The Payload Controller shall write image data to storage device at speeds no less than 34.4 MB/s	FINCH-PayloadController-Storage	Test
FINCH-PayloadController-ImageStorageQuantity	The Payload Controller shall support the storage of at least 734 MB worth of imaging data, higher is better, as verified through functional testing	FINCH-PayloadController-Storage	Test
FINCH-PayloadController-TelemetryStorage	The Payload Controller shall be capable of storing telemetry data.	FINCH-PayloadController-Storage	Test
FINCH-PayloadController-DownlinkSpeeds	The Payload Controller system shall not bottleneck downlink to the ground station.	FINCH-PayloadController-PayloadOps	Test
FINCH-PayloadController-RFLink	The Payload Controller shall utilize the faster interface with the RF transceiver that makes it possible to send data to the transceiver such that it does not bottleneck communication.	FINCH-PayloadController-DownlinkSpeeds	Test
FINCH-PayloadController-UplinkSpeeds	The Payload Controller system shall be capable of receiving data from the RF module directly.	FINCH-PayloadController-PayloadOps	Test
FINCH-PayloadController-SendToRF	The Payload Controller shall be capable of directly sending payload and telemetry data to the RF module.	FINCH-PayloadController-PayloadOps	Test
FINCH-PayloadController-Compression	The Payload Controller should support image compression.	FINCH-PayloadController-PayloadOps	Test

Table 4: Electrical System Requirements

5 Verification and Validation Plan

Electrical requirements will be verified via tests and demonstrations. The verification and validation plan will follow the following timeline:

6 High Level System Architecture

6.1 Block Diagram

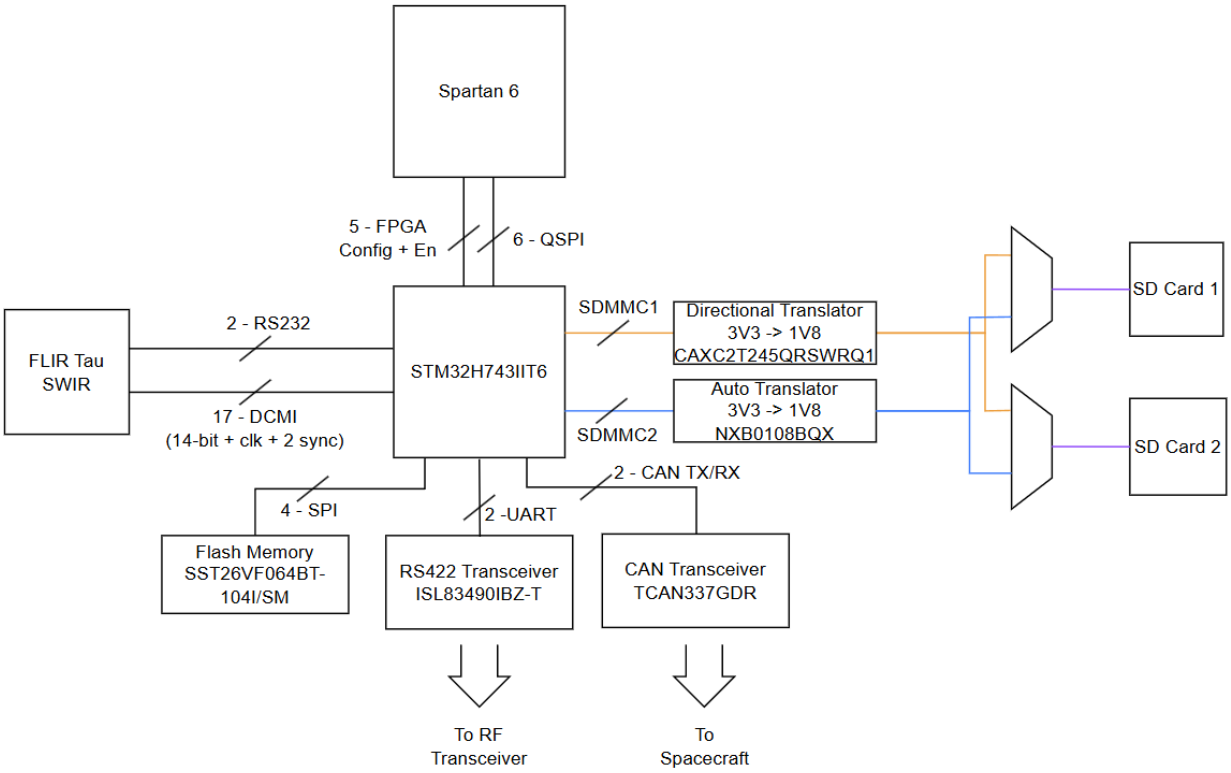

Figure 5: PAY v1.0 Block Diagram

6.2 Connection to the Rest of the Spacecraft

PAY talks to the rest of the spacecraft via the CAN bus. Fortunately, the STM32H7 has native CAN support so all that is needed for CAN communication is the CAN transceiver. As with other electronic systems on the spacecraft, PAY will also be a node on the Cubesat Space Protocol.

CAN is also used to talk to the RF Satlab SRS4 transceiver. However, RS422 is used for faster data transfer to the SRS4.

6.3 Power

PAY simply accepts 3V3 from the bus. With the previous mechanical layout of boards, this was fine. However, there might be a harness of non-negligible length that will provide power to PAY and OBC. Consequently, end-point conversion might now be necessary.

Within the board, there is also 1V8 via a buck converter which is supplied to the FPGA as an intermediate voltage. There is another 1V8 via an LDO which is supplied to the SD card if it is run at UHS mode.

6.4 Grounding Scheme

The initial grounding scheme before the payload redesign was a Star grounding scheme. This was easy since the boards were just stacked together and then cables simply emanated from that stack. For the new mechanical layout, though, the EPS board will be separate from the rest of the boards.

7 Detailed System Architecture

7.1 SD Cards

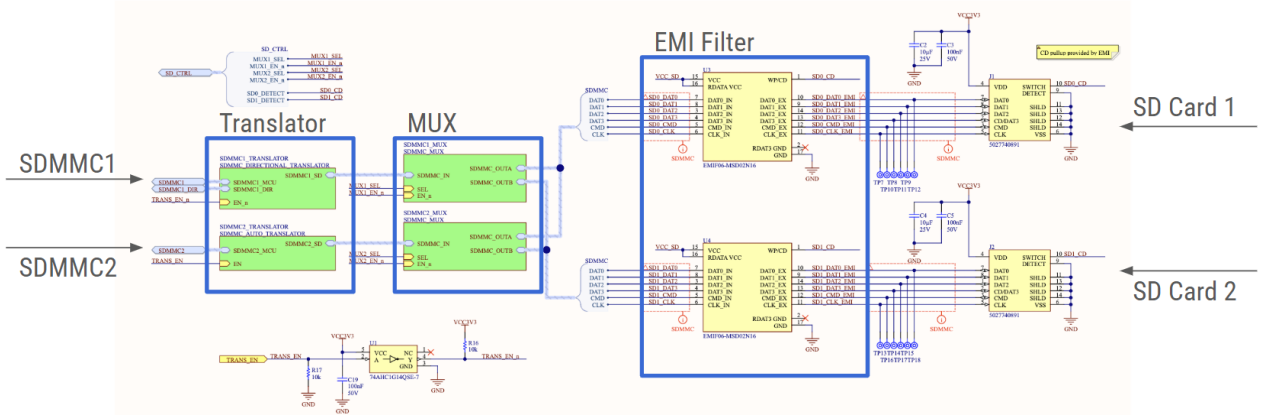

Figure 6: SD Card Schematic

The SD card portion was designed so that both SDMMC interfaces of the H7 could be used. The reason is not for redundancy– if one SDMMC failed, probably the whole chip failed. Rather, it is because the two SDMMC's are different. Quoting Reid when he designed the board: Critically, SDMMC1 has direction pins, which can tell the voltage translator which direction data is flowing. Without these pins, you must rely on auto-direction sensing which can be finicky. The old plan was to use the STM32H743ZI, which only supported SDMMC2 while using the DCMI peripheral, but we've decided to use the higher-pinned STM32H743II which allows for the coexistence of both peripherals. However, for timeline safety, I will be implementing both methods in the hopes that at least one will work.

An important thing to note are the different modes for the SDMMC protocol which enable faster transfers shown in Figure 7. Given the calculation for the required data transfer speed in Eq. (4), going to UHS may not be necessary.

Table 1. SDMMC supported speed modes

SD & SDIO	Max bus speed [Mbyte/s](1)	Max clock frequency [MHz](2)	Signal voltage (V)
DS (default speed)	12.5	25	3.3
HS (high speed)	25	50	3.3
SDR12	12.5	25	1.8
SDR25	25	50	1.8
SDR50	50	50	1.8
DDR50	50	100	1.8
SDR104	104	208	1.8
MMC cards			
Legacy compatible	26	26	3/1.8/1.2
High speed SDR	52	52	3/1.8/1.2
High speed DDR	104	104	3/1.8/1.2
High speed HS200	200	200	1.8/1.2

1. *Maximum bus speed in 4-bit mode for SD& SDIO and 8-bit mode for MMC cards.*
2. *The maximum data transfer depends on the maximum allowed I/O speed.*

Figure 7: SDMMC Modes

7.2 Image Transfer Method

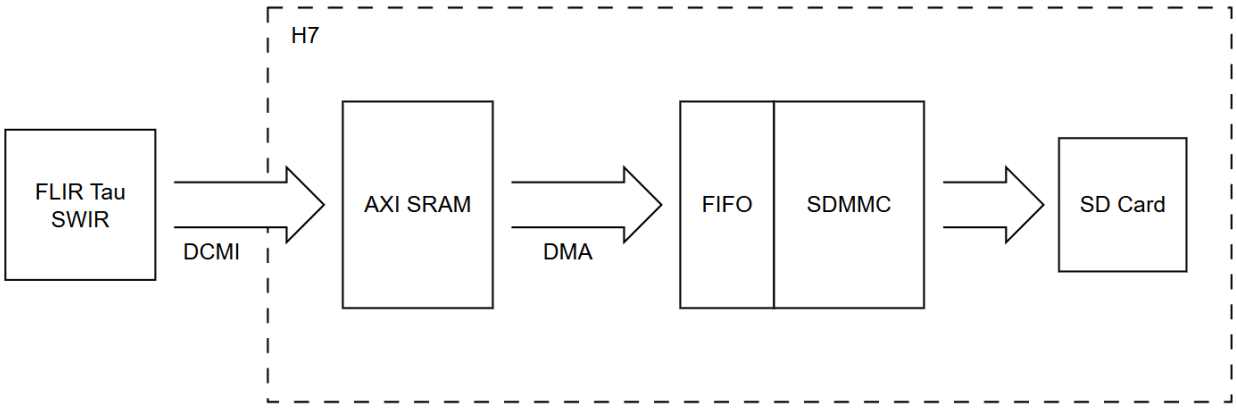

Figure 8: Image Transfer Sequence

Figure 8 depicts the image transfer sequence and Figure 9 shows the block diagram for the SDMMC peripheral. The images from the DCMI peripheral are first written to the IDMA portion on the SRAM that will be allocated the transfer processs. From (3), around 163kB of memory is needed. This requirement makes SDMMC1 easier than SDMMC2 to use. From

Figure 2 on the STM32H7 Datasheet[2], while SDMMC1 has access to 512kB of contiguous SRAM, SDMMC2 has access to 2x 128kB + 32kB of SRAM. SDMMC2 could still work but it would be less convenient.

The image data is then streamed from the SRAM into the FIFO via the 64-bit AXI bus. It is the FIFO that then pushes the data onto the SDMMC line and to the SD card.

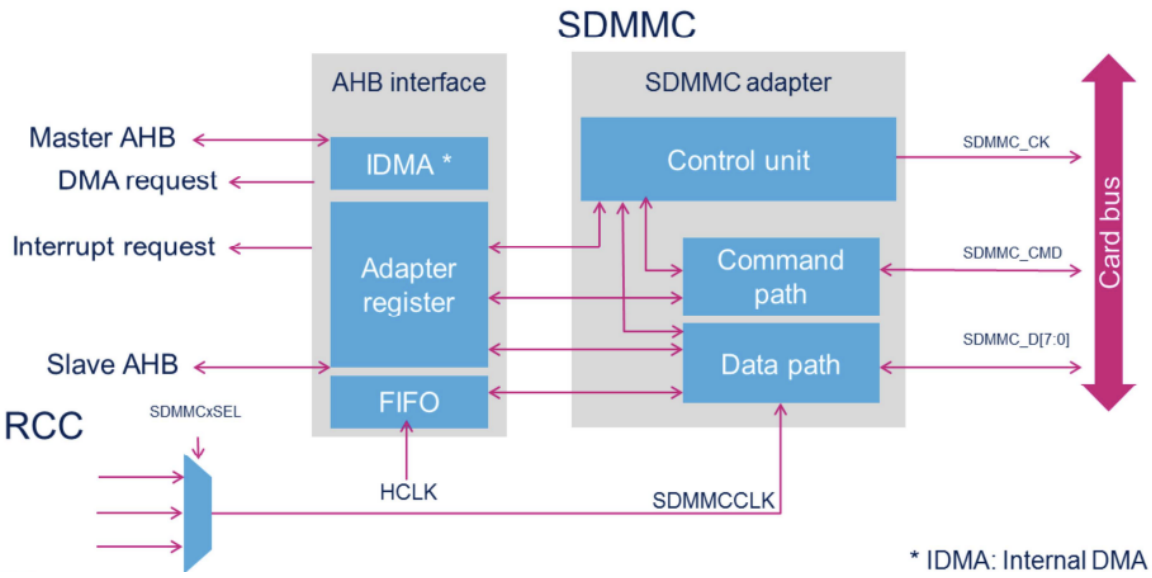

Figure 9: SDMMC Block Diagram[3]

8 Possible Risks

The firmware uses the ZephyrRTOS. It has its strengths but using SDMMC with it does not seem to be too trivial as there is no native support.

Development will target the use of HS mode. A risk is that the firmware may not reach the necessary transfer speeds. This seems like a low risk.

9 Development Schedule and Status

Early summer 2025: verified that the H7 could be programmed.

July 19, 2025: Zephyr was flashed on PAY and ran on HSE. Recorded a current draw of 172mA running the program at 480MHz.

[2]STMicroelectronics. *Datasheet - STM32H742xI/G STM32H743xI/G*. Mar. 2023. URL: `https://www.st.com/resource/en/datasheet/stm32h743vi.pdf`.

10 Open Issues and Future Work

- Current draw seems high for nominal operation. Will be good to explore low power modes of the H7.
- Actually saving an image to the SD card
- Integration with OBC once CAN is brought up and a CSP node has been programmed onto the H7.

References

ELECTRICAL GROUNDING ARCHITECTURE FOR UNMANNED SPACECRAFT. Feb. 1998. URL: https://s3vi.ndc.nasa.gov/ssri-kb/static/resources/NASA-HDBK-4001.pdf.

STMicroelectronics. *Datasheet - STM32H742xI/G STM32H743xI/G*. Mar. 2023. URL: https://www.st.com/resource/en/datasheet/stm32h743vi.pdf.

– *STM32H7-Peripheral-SDMMC interface*. n.d. URL: https://www.st.com/content/ccc/resource/training/technical/product_training/group0/06/8b/84/d3/16/f6/4b/95/STM32H7-Peripheral-SDMMC_interface_SDMMC/files/STM32H7-Peripheral-SDMMC_interface_SDMMC.pdf/_jcr_content/translations/en.STM32H7-Peripheral-SDMMC_interface_SDMMC.pdf.

Tau SWIR ISC1202 Software Interface Description Document. Oct. 2018. URL: https://drive.google.com/file/d/1tCeISX1MTPkfJr_OYQdCwi8xV5pYETpT/view.

Tau SWIR ISC1202 User's Guide. Oct. 2018. URL: https://f.hubspotusercontent10.net/hubfs/20335613/tau-swir-isc1202-user-guide.pdf.

Tau SWIR Product Specification. Oct. 2018. URL: https://drive.google.com/file/d/1hDU8Dk9fygd2EyuG_VCbWGHtwMtvR_7e/view.