Design Concepts and Overview

CubeSat Camera System (C2S)

Felix Abbott — Perth Aerospace Student Team

ADCS Department — CubeSat 1 Mission

October 2025

Contents

1 Introduction

1.1 Background

The **CubeSat Camera System (C2S)** is the proposed mission payload for the Perth Aerospace Student Team's (PAST) first CubeSat. This inaugural mission represents a critical milestone in the team's broader goal of developing low-cost, modular spacecraft capable of performing meaningful scientific and engineering demonstrations in Low Earth Orbit (LEO). C2S is designed to operate as an Earth observation payload, taking photos of certain land- and oceanscapes. Beyond its imaging objectives, the system will serve as a testbed for evaluating the viability of student-developed space hardware in a real orbital environment. The payload aims to balance performance, reliability, and resource constraints typical of CubeSat-class missions.

This document outlines the key design drivers, system architecture, component selection strategies, and trade-offs that inform the development of the CubeSat Camera System prototype. It also provides an overview of the integration challenges and planned validation strategies for the final model.

Where appropriate, this document often refers to the preliminary literature review which can be found here. If there is assumed knowledge from the research, it will be mentioned e.g., ”see literature review for breakdown”.

1.2 Project Scope

The primary objective of this project is to design, develop, and demonstrate a standalone CubeSat-compatible camera system prototype. The prototype aims to meet key performance and integration requirements for future low Earth orbit missions, with a focus on modularity, efficiency, and feasibility within typical CubeSat constraints. The system must operate independently to capture, process, and store image data using commercially available components.

The specific scope includes:

- Acquisition of RAW image data from a CMOS image sensor with pixel binning capabilities to enhance low-light performance and reduce onboard processing demands.
- Onboard conversion of RAW data to compressed JPEG format in the Y'UV colour space. Chroma subsampling will be employed to reduce image file size while preserving perceptual image quality, which is critical for bandwidth and storage-constrained space environments.
- Integration of an uSD card breakout board to store processed images, enabling straightforward data access during ground testing and allowing for future adaptation to in-flight non-volatile memory solutions.
- Implementation of a fixed focal length lens system to simplify the optical design, reduce mechanical complexity, and optimise image consistency. Lens and sensor pairing will be chosen based on trade-offs between resolution, field of view, and compatibility with CubeSat mechanical form factors.
- System design intended for eventual microcontroller or FPGA control, with all power, data, and interface components selected to support future embedded control integration.

The following features are intentionally excluded from the current development phase to maintain focus and ensure feasibility within available time, budget, and resource constraints:

- Wireless or wired image transmission beyond the camera module, such as radio downlink or USB tethering.
- Integration with full CubeSat systems, including EPS, onboard computers, or attitude control systems.
- Design or implementation of redundancy, fault tolerance, or radiation hardening beyond basic best practices for PCB layout and component selection.
- In-orbit deployment or qualification testing (e.g., thermal vacuum, vibration).

This scope defines a foundational platform from which future iterations may extend toward full CubeSat integration and space qualification.

1.3 Project Milestones

The project has been divided into five key-phases that each act as sub-projects and each lead into the next. This helps compartmentalise the project for better management and tracking.

ID	Milestone
1	Design and build a fully custom CubeSat-compatible prototype camera module, including PCB layout, component selection, and sensor integration.
2	Develop firmware to configure and control the image sensor (e.g., via I^2C/SPI), supporting raw image capture and tuning of parameters like exposure, gain, and white balance.
3	Implement an image pipeline to convert raw sensor data into usable image formats, including demosaicing, gamma correction, and compression.
4	Design a mechanical housing that meets CubeSat volume, mass, and thermal constraints, and could interface with the rest of the CubeSat.
5	Test and validate the camera module under simulated space conditions (thermal cycling, vibration, and low light).

Table 1: Milestone objectives for the duration of the project

1.4 Project Responsibilities

The prolonged timeline and importance of the project has shown that it is critical to delegate parts of the project to other PAST members. For the software and mechanical aspects of the project, the milestone has been delegated to other members. The following table highlights the milestone each member is responsible for.

Name	Department	Milestone Responsibility
Felix Abbott	Electrical	1, 2, 3, 5
TBD	Software	2, 3
TBD	Mechanical	4

Table 2: Breakdown of project responsibilities

1.5 Timeline

The timeline provides a clear schedule to track milestone completion, outlining which project phase should be active each month and the expected duration of each phase to ensure steady progress.

Date	**Phase**	**Key Tasks**	**Relevant Objective ID**
Aug '25	Project Planning & Kick-off	Define system requirements; Select image sensor; Preliminary architecture planning; Initial parts research	1
Sep–Nov '25	Schematic Design & Component Selection	Design camera circuit schematic (breakout board); Select power regulation, oscillators, EEPROM, etc.; Create block diagrams	1
Dec '25 – Jan '26	PCB Layout & Review	Begin PCB layout; Design for manufacturability (DFM); Include test points, headers, debugging pads. Begin investigating camera lens	1
Feb '26	PCB Fabrication & Assembly	Send board to fabricate and order components; Begin manufacturing	1
Mar '26	Firmware: Sensor Bring-Up	I^{2}C/SPI communication tests; Basic sensor initialisation; Test register writes/reads	2
Apr–May '26	Image Processing Pipeline	Capture raw Bayer data; Implement demosaicing, gain, white balance; Export images via USB/SD/Wi-Fi	2, 3
Jun–Jul '26	Mechanical and PCB Prototype	Design simple 3D-printed housing; Consider heatsinking or fan-based thermal dissipation; Design lens mount if needed. Alongside this, developing the PCB layout for the prototype board (not a breakout).	4
Aug–Oct '26	Functional Testing	Send board to fabricate and source any further components; Manufacture board once arrived. Test camera stability, power usage, image quality under varied lighting; Conduct thermal monitoring; Log long-duration operation	5
Nov '26	Optimisation	Tune firmware; Improve image quality; Reduce power and memory usage; Prepare documentation	2, 3
Dec '26	Final Integration & Reporting	Compile results; Document all hardware/software; Prepare final presentation/demo/report	

Table 3: Project schedule and key tasks

2 Project Planning & Kick-off

This phase focuses on establishing the foundation for the project by defining system requirements, selecting the image sensor, planning the preliminary architecture, and conducting initial parts research.

Please note. Although certain components are investigated, I only focus on key-components like the sensor and processor. Resistors, capacitors, debug LEDs and other components mentioned in datasheets (like recommended circuits) are not discussed. For more information, please see subsection 3.2

This phase was reviewed by Dr. Robert Howie on August 12th, 2025.

2.1 System Requirements

After the literature review, the following requirements for C2S are established:

Requirement	Description
1	A pixel resolution of 500-750 metres was chosen to capture major urban areas, such as Perth, in a single frame. This resolution balances ground coverage and system simplicity by avoiding deployable or extendable lenses.
2	A fixed focal length lens simplifies the optical system and ensures consistent sharpness at the set focal distance. This reduces complexity, mass, and power needs, making it suitable for compact CubeSat prototypes.
3	The system will use YUV format with chroma subsampling and JPEG compression to reduce file size while maintaining image quality, optimising bandwidth for CubeSat transmission.
4	A CMOS image sensor was chosen for its lower power use, faster readout, and easier digital integration compared to CCDs. This suits compact, power-sensitive platforms like CubeSats.
5	If the imaging sensor exceeds 1MP, pixel binning will allow switching between high-resolution and lower-resolution images with improved quality through increased light sensitivity and reduced noise.
6	Although lenses won't be investigated in this milestone, the breakout board must allocate space for M12 lenses to mount for later milestones.

Table 4: C2S system requirements as of August '25, Phase 1.

2.2 Component Selection

Please note, the following components are not heavily optimised for the system and can be improved. They were selected because they fit the requirements and scope of the breakout board. Further investigation into components will be conducted for the prototype.

2.2.1 Local Power Supply

Opposed to a switching converter, an LDO was chosen because of the cost and fewer external components required. Since the required power lines will most likely come from other projects like EPS, LDOs were a great choice for cheap and compact local power supplies on the breakout board.

The TCR3DF series Linear Drop-Off Voltage Regulators (LDOs) were selected as the local power supplies for the breakout board because I could source all of the required power lines from the same series. An extensive search for LDOs was not conducted since the first results met the criteria and I did not see a need to further optimise the component selection.

2.2.2 Imaging Sensor

The main image sensor picked out for the breakout board was the MIRA220 (datasheet found here). It is a widely used image sensor, particularly in applications requiring high-speed, low-light, and near-infrared (NIR) imaging. Its key features include:

- 2.2 MP NIR-enhanced global shutter CMOS sensor, ideal for capturing motion without distortion and low-light near-infrared applications (see literature review for breakdown).
- It offers a flexible bit depth (8, 10, or 12 bits) and supports up to 90 fps at full resolution, allowing for high-quality imaging and fast frame capture.
- The sensor is available in monochrome, RGB, and RGBIR variants, enabling both colour and NIR-sensitive imaging options.
- Compact package and MIPI CSI-2 interface simplify board integration and compatibility with modern processors and FPGAs.
- Moderate power consumption (350 mW active) balances performance with CubeSat power constraints.
- Pixel size of 2.79 µm provides a good balance between resolution and light sensitivity for CubeSat imaging needs.
- Its global shutter with correlated double sampling reduces noise and improves image quality under variable lighting conditions.

While other sensors may offer advantages such as lower power consumption or smaller pixel sizes, the MIRA220 meets the key requirements for the breakout board, particularly its NIR sensitivity and global shutter capabilities which are valuable for enhanced surface imaging. Using a more specialised or lower-power sensor at this stage would significantly increase costs and design complexity, making the MIRA220 a practical choice for prototype development with room for upgrades in subsequent revisions.

Although the MIRA220 uses a different interface than LVDS, it maintains low power and complexity, which supports efficient prototyping without compromising image quality or system integration. It has native CSI-2, which means it can easily interface with the selected image processor subsubsection 2.2.4 without any external components like LVDS drivers and receivers that consume significant amounts of power. Moreover, AMS Osram have open sourced their camera board designs (found here), which means the reference schematic and PCB will make the development process much smoother and less prone to errors.

2.2.3 Lens

This section contains information relevant to the literature review.

There's a clear trade-off between performance and zoom. Consider a single pixel of size d with a focal point located at a distance f from the pixel. The angle formed between the pixel's centre and its top edge changes depending on the pixel's height. Now, imagine extending lines from the pixel's edges through the focal point out for several hundred kilometres. As the pixel size grows, the area it covers on the Earth's surface also expands. Therefore, to maintain a constant spatial resolution when increasing the pixel size, the focal length must be increased accordingly.

The obvious limitations to the our image are the file size and CubeSat size. I cannot be transmitting 2.2MP images down to Earth, so I must bin down or crop the image. If I bin down by n times, then the pixel size and focal length increases by n^2. If I place a hard limit of 1MP on the pixel count, then I only need a focal length of $<$10mm. If the sensor is binned once (2.224MP$\rightarrow$0.5MP, 2.79μm$\rightarrow$11.16μm) to improve performance and reduce file size, the required focal length for the camera is: $f = hd/s$ where d is the pixel size, h is the orbital height and s is the spatial resolution. I find that the expected focal length for a M12 lens would be 8.9mm for a spatial resolution of 500m at an altitude of 400km. This would capture an image 400km x 350km across. This fits within our (assumed) limits, however, I may adjust the scope towards the 750m spectral resolution range to comfortably sit within size contraints (depending on the results of testing).

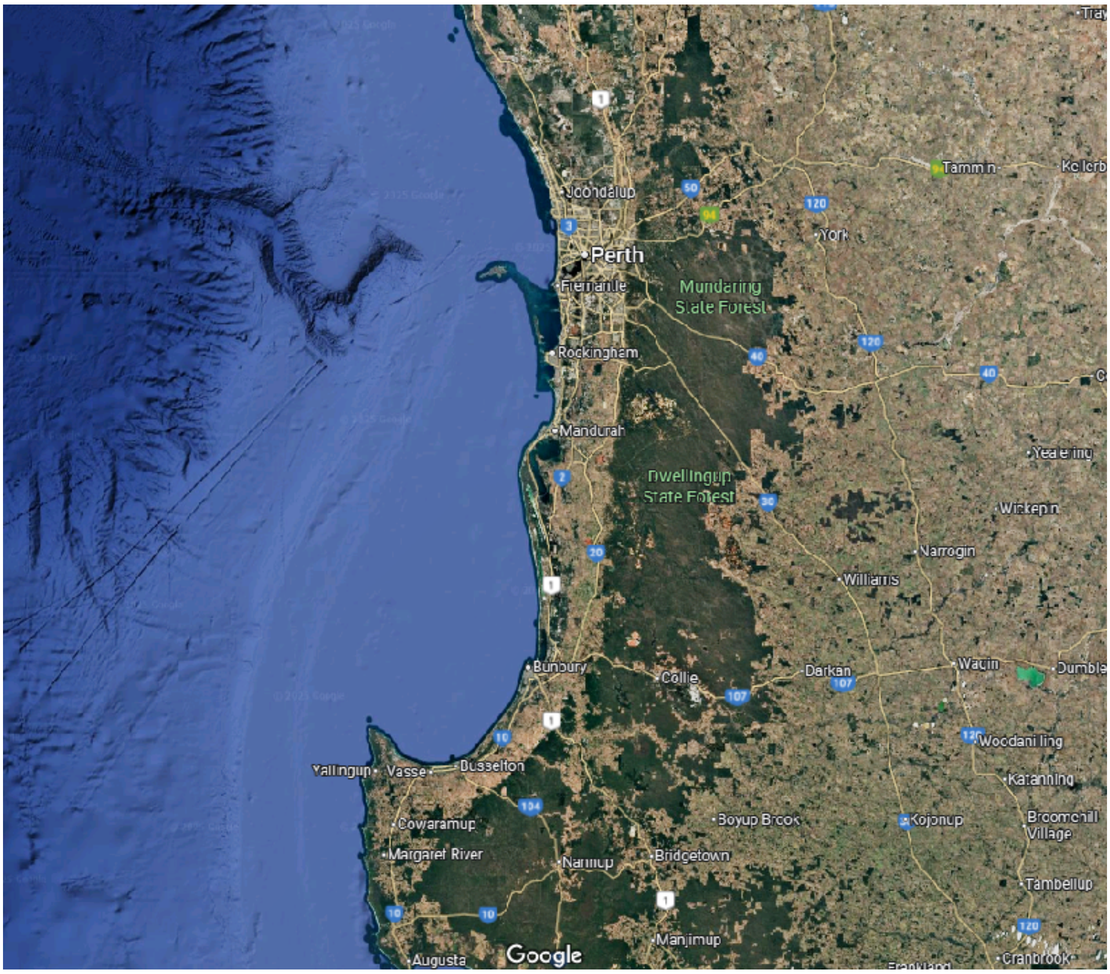


Figure 1: Expected image coverage and resolution for the MIRA220 sensor, binned once with a focal length of 8.9mm.

2.2.4 Image Processor

A lot of consideration was placed into the selection of the image processor. Initially, I investigated ASICs suited for image processing. I found that most active and supported imaging ASICs are manufactured for larger scopes i.e., video and AI processing, which exceeded the current power and thermal requirements of the project. Most ASICs that fit the current scope are more dated and less-supported. For example, the ADSP-BF70x line were prime candidates at the start of the project since their attributes fit the current scope and aren't ”overkill”. They processed low-resolution images at a high-energy efficiency, could utilise external RAM and interfaced with CMOS senors with PPI. Despite this, the line (as of mid-2025) is more than a decade old and has limited support. Other processors from NXP, Microchip and Texas Instruments either showed the same limited support or lacked imaging processing capabilities (most designed for power systems). After consulting with online forums like r/ECE and r/embedded, as well as with Dr. Robert Howie regarding his students' previous star tracker project, it was decided that a general processor like the STM32 was best suited for the current application.

STM32s have been widely used in CubeSat applications over the years and have become a widely supported platform for ameteur aerospace applications. Unlike FPGAs, which can compute parallel processing, STM32s use high-level languages like C/++, have faster development cycles and have a larger community. They are typically more power efficient and can have built-in image processing and encoders. Amongst the STM32 line, I selected the STM32N6 as the image processor over others like the H7 for the following reasons:

- Supports DCMI as well as MIPI CSI-2. H7s do not support the latter.
- Uses an image signal processor (ISP), digital signal processor (DSP) and neural processing

unit (NPU) for processing unlike a traditional software-powered CPU.

- Has hardware-accelerated subsampling techniques, whereas others use software.
- Has a higher power efficiency compared to the H7 (and can be underclocked to save more power).
- More recent and will be supported for longer.

The following table explores some of the key differences between the STM32 processors:

Feature	**STM32N6 Series**	**STM32H7 Series**	**STM32F4 Series**
Core	Arm Cortex-M55 + Neural ART NPU	Arm Cortex-M7 (single or dual-core M7/M4)	Arm Cortex-M4
Performance	Up to 480 MHz (core), includes vector extensions	Up to 550 MHz (M7), strong DSP/float perf	Up to 180 MHz, moderate DSP capability
Memory (RAM/Flash)	Up to 4 MB SRAM, 4 MB Flash	Up to 1 MB SRAM, 2 MB Flash	Up to 256 KB SRAM, 2 MB Flash
Imaging Interfaces	DCMI, SPI, QuadSPI, JPEG Codec, Chrom-ART, DMA2D	DCMI, JPEG Codec, Chrom-ART, DMA2D	DCMI (on select models), basic DMA
Subsampling / Encoding	Hardware JPEG with native Y'UV support	Hardware JPEG + DMA2D for YUV	Software-based only
Power Efficiency	Moderate to High (based on workload + NPU efficiency)	Moderate (high-performance, higher dynamic power)	Good (lower performance = lower power)
Target Applications	AI vision, edge inference, intelligent sensors, imaging pipelines	High-performance imaging, audio, motor control, GUI apps	General-purpose MCU, audio, lower-end imaging
Availability	Early access or newly launched (as of 2025)	Mature and widely supported	Legacy but stable

Table 5: Comparison of STM32N6, STM32H7, and STM32F4 Microcontroller Families

I selected the STM32N657I0H3Q as the image processor for the breakout board. I opted for this processor amongst its other counterparts (like the STM32N657X0H3Q) because of a similar feature-list at a lower price and significantly lower pin count. Although this component is designed for more complex- and intense-tasks, I will be able to evaluate which features will be utilised and which won't be, which will aid in selecting the optimal processor for the prototype. It serves as a solid middle ground between the more extensive and cheaper counterparts

Segment	**Code**	**Meaning / Explanation**
Manufacturer Family	STM32	STMicroelectronics 32-bit microcontroller family
Series	N6	STM32N6 series - based on Arm Cortex-M55 core with Neural ART NPU
Sub-Family / Feature Set	57	High-end variant in the STM32N6 family (e.g., more memory, interfaces, and image processing features)
Memory / Package Variant	I0	Midrange flash/RAM size and medium-pin-count package
Temperature Grade	H	High-temperature / industrial range (typically -40℃ to +85℃ or higher)
Variant Code	3	Specific feature or silicon variant (usually minor revisions or option sets)
Qualification Grade	Q	Automotive or industrial qualification

Table 6: Breakdown of STM32N657I0H3Q Part Number

Despite the current choice, it is highly recommended that if the scope of the project were to expand in the future, further research into ASICs should be completed. Compared to modern SoCs or FPGAs with image processing capabilities, some ASICs provide a lower-cost entry point with greater efficiency in real-time image processing.

Since the selected processor has sufficient internal memory to buffer a full-resolution image, external memory is not required to support real-time image acquisition and processing. If future revisions of the project (i.e., the prototype) require more memory to buffer the image, the selected processor is compatible with external uSDRAM like the IS45S16320D-7TLA2.

2.2.5 SD Card Reader

Since I need a short and effective method of checking the files in a remote environment without tethered access (USB connection), I opted for saving the files on an uSD card. Although this does not provide real-time imagery, the slightly smaller scope allows us to better focus on the functionality of the camera rather than how the fully processed image is moved around.

Instead of PCB-mounting an uSD card directly, I chose the 254 Adafruit breakout board because it can be easily connected via standard PCB headers. Although it appears this step has been short-cut, the prototype will not use an uSD card reader so it suffices as an effective solution without having to deal with the techincal aspects. As the prototype is expected to offload images to another device, this breakout-based approach approach works well for viewing the file in its intermediate stage (leaving the processor and being sent to FFS).

2.2.6 External Clock

The ECS-TXO-3225MV-240-TR was selected primarily because it provides a highly stable and precise 24 MHz clock signal, which perfectly suits our system's timing requirements. This temperature-compensated crystal oscillator (TCXO) ensures minimal frequency drift across varying thermal conditions, a critical factor for reliable operation in aerospace environments. Additionally, the Avionics Department Representative recommended a TXCO, reinforcing its suitability for our application. Since the design only requires a single clock output rather than a differential pair, this oscillator meets our needs efficiently without unnecessary complexity. Additionally, the MIRA220 image sensor also requires a 24MHz reference clock, so only a single external clock is required for the breakout. Although the clock is connected to two components, its load capacitance of 15pF should prevent any overloading symptoms.

2.3 Block Diagram

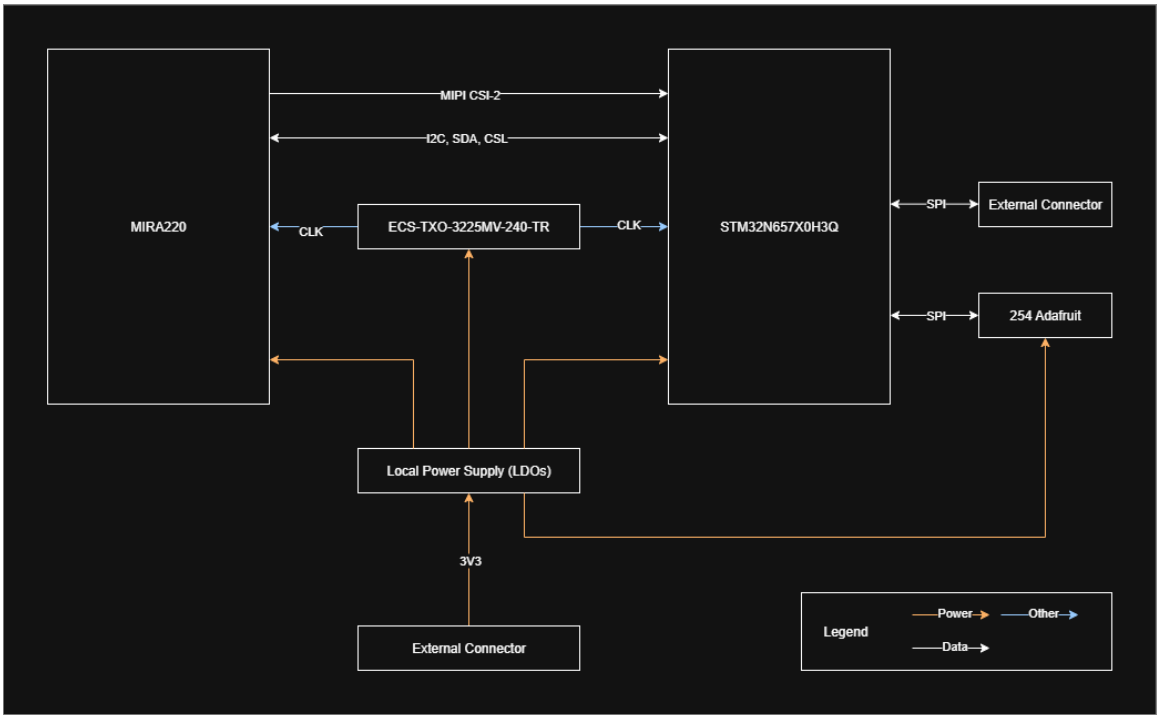


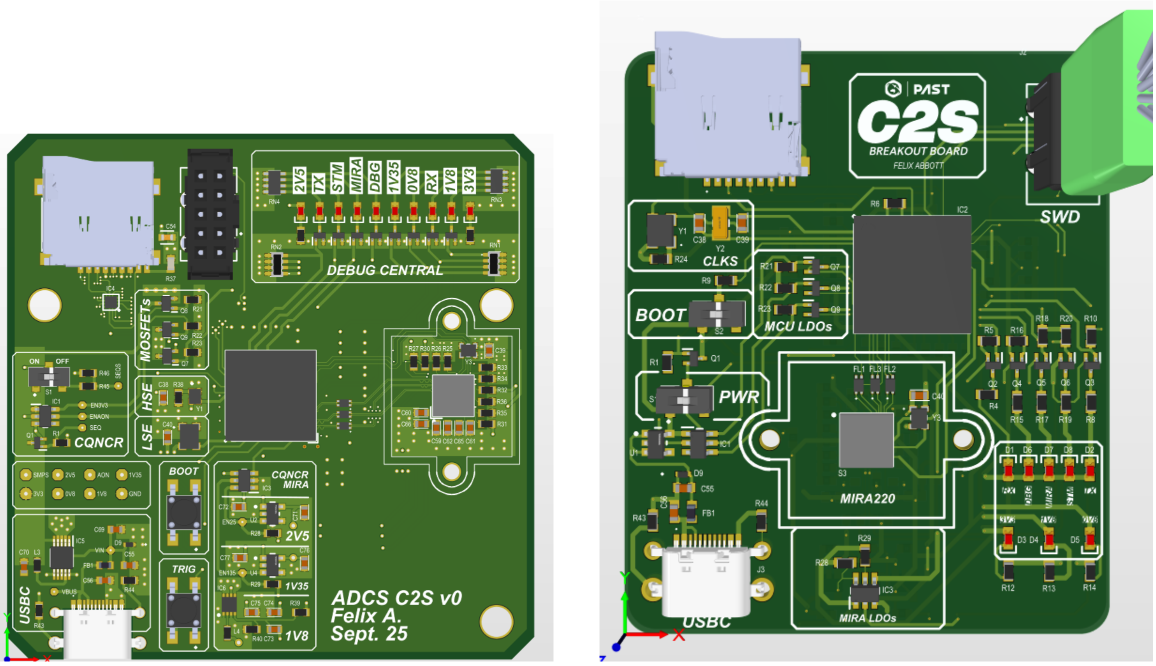
Figure 2: General block diagram of C2S breakout board. Components like resistors, capacitors and suggested circuits have been omitted from the diagram. This is meant to show the core functionality between the core components.

3 C2Sv0 (Shelved)

3.1 Remarks

Throughout the development phase of the breakout board, the project faced several scope revisions. *These remarks cover comments for both subsection 3.2 and subsection 3.3.*

The first revision focused on the structure of the breakout board. Initially, the breakout board was fully encapsulated, meaning that all the components (processor, sensor, uSD, etc.) were all housed on a single PCB. I found that it made more sense to split the board into the sensor and processor, given the prototype will use the same format. Not only would this prompt design considerations for the prototype, but if the breakout boards were successful, they could be used to debug the prototype i.e., the functioning-processor breakout could be used to program the prototype sensor, and vice-versa. This revision did not introduce any major roadblocks or inconveniences to the development since all draft PCBs at the time either had the sensor placed in an inconventient location, or the sensor was essentially ”disconnected” from the rest of the board (see Figure 3)



(a)

(b)

Figure 3: Draft PCBs of the C2S Breakout Board.

The second revision was centered around the selection of the image sensor. In section 2, I selected the MIRA220 as the image sensor for both the breakout and prototype PCB, however, throughout development, I found that the component's characteristics were too fine to reasonably justify (within the project budget). The BGA ball diameter of the MIRA220 was 0.2mm (200μm), much smaller than JLCPCB's manufacturing capabilities. Morever, the trace widths required to route the BGA was much too thin. These tight tolerances drastically increased the manufacturing cost of the board from what I expected to be $50AUD to ¿$300AUD (with some going beyond $500AUD). The price to manufacture the boards with this sensor was far too expensive, so, I pivoted to the next-best sensor with larger tolerances. Although the tolerances of the new sensor was tight, the boards were now manufacturable by JLCPCB at a reasonable cost.

The third revision aimed at reducing the failure rate of the breakout board. In addition to splitting the PCB into the sensor and processor boards, I decided to add the STM32N6 Nucleo Board to the project scope as well. This would allow us to interface with the sensor directly even if the processor board didn't work. Now that the minimum success criteria of the breakout board no longer required the entire processor board to work, the risk of not meeting the minimum criteria drastically reduced.

The fourth revision involved reworking the naming scheme. As mentioned in section 2, the first PCB was called the 'breakout' or 'breakout board'. However, as the number of PCBs in the project grew, I required a new method of referencing each board. In addition to the 'breakout' term, I had begun to call this version/iteration C2Sv0 to highlight on the project's initial development as a breakout board, whereas C2Sv1 would refer to the first functioning version of the project, the prototype. The terms v0 and 'breakout' have since been used interchangeably. Moreover, each PCB has been named according to its purpose:

- C2Sv0-P: (P)rocessor.
- C2Sv0-S: (S)ensor.
- C2Sv0-A: (A)daptor.

The fifth revision was dedicated to saving costs on the manufacturing process. After factoring in all of the component and manufacturing costs, I were facing approximately $813AUD and $1042AUD for one and two PCB sets (with one set of components for redundancy) respectively. This was far too expensive for just the breakout board, so modifications to v0 were required. To cut down on the cost, I decided to re-combine the processor and sensor board onto one PCB (with break points) to avoid the double-fee for tight tolerances and only be charged for tight tolderances on one PCB instead. Since JLCPCB does check the boards for this type of 'manipulation', I added some fake components and silk-screen onto the tabs (section to be snapped) to make it seem like it was one coherent PCB instead of two that were clearly going to be split after production.

On September 23rd, 2025, **this version was shelved** because of its high price point. After discussing the project with Trsitan Davies and Robert Howie, I concluded that the cost of the PCB ($750AUD) was far too great to justify for a single revision with one person working on it, given mutliple revisions may be required. The PCB was complete however, and the gerber files along with the components can be found on the C2S SharePoint folder in-case the version is picked up again in the future.

3.2 Schematic Design & Component Selection

3.2.1 Overview

This phase involves designing the camera circuit schematic using Altium Designer, laying the groundwork for subsequent PCB development and component selection.

The schematic is divided into eight sub-sheets:

1. Processor
2. Image Sensor
3. Clocks
4. uSD Card
5. LPS
6. SWD Connector
7. USBC Connector
8. Power Sequencer

Only critical paths were included on the Top-Level schematic. Routes like general power and ground were not included.

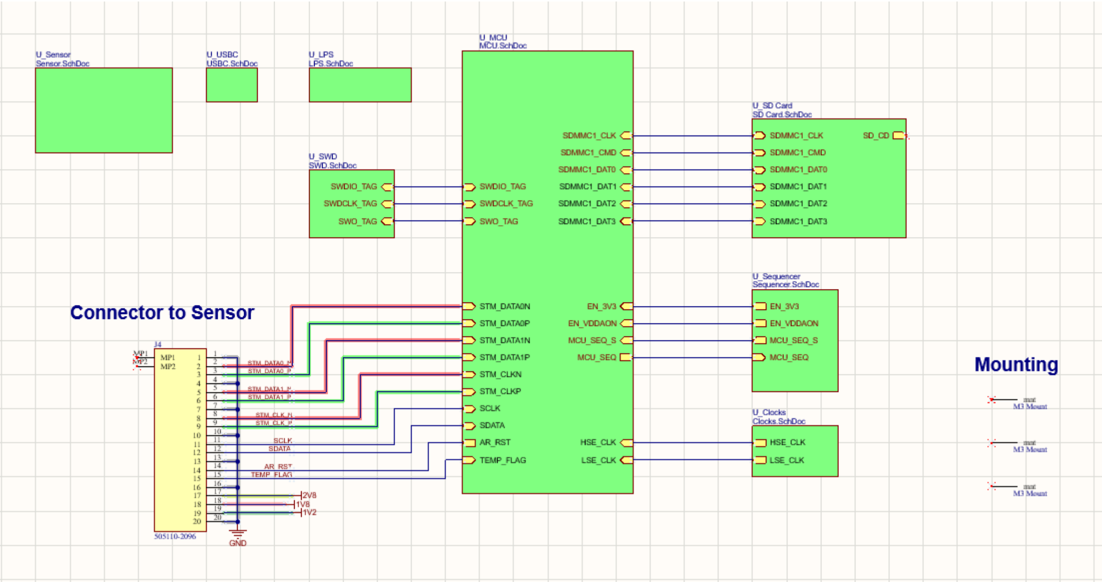


Figure 4: Top-Level schematic sheet from C2S Breakout Board Project.

3.2.2 Processor

The STM datasheet and getting started handbook were used to identify pins and their purpose. Most circuits and component values were obtained from the guide or datasheet. For example, pg. 7 of the getting started guide details the decoupling power figures required on most of the supply pins. For pins that were not explicitly described in the guide, I used the datasheet to find the pins. The decoupling inductor on VDDAON (near pin E1 on the schematic) had no specified value, so, I copied the inductor value used on the VDDCORE pull-down set. The schematic symbol was split in two to make it easier to connect the GPIO pins (opposed to the BGA order it was previously in: A1, A2, ..., B1, B2, ..., which made it difficult to cleanly connect components).

Additionally, debug LEDs were added to the schematic to make it easier to know which lines are active and which aren't. If I test one of the power lines once and it provides us with the correct voltage, then whenever the corresponding LED is active, I know it has the correct voltage. Similarly, an LED was added for reading from the sensor and writing to the uSD card. Resistor arrays were used rather than individual resistors to save on the number of components on the PCBs, but MOSFET arrays weren't used because their biasing voltage is typically higher than the regular MOSFET bias voltage (0.7V), so lines like 0.8V would not bias the transistor and activate the LED. Each LED will use the HSMH-C190 (green). According to the datasheet, the maximum current rating is 25mA, so with a supply voltage of 3V3, I require a minimum of 68Ω with the MOSFET configuration. I selected values of 470Ω since it's a very common resistor value and it satisfies the requirements without being too bright (too bright and they'd begin to blind us after a while). Although the MOSFET set will be slightly dimmer than the other, their brightness difference is not a concern. A high value pull-down resistor was included on the base of the MOSFETs to ensure that the MOSFET is not active when the driving signal is not actively pulling the gate to a defined voltage (also used a resistor array). Since the MOSFETs are used on non-time-critical areas (like debug LEDs and switches), the time difference between $10\mathrm{k}\Omega$ and lower resistances was not a concern. I picked $10\mathrm{k}\Omega$ since it is a common resistor value.

The 3V3, 1V8 and VDDAON enable MOSFETs use a different package since their drain current exceeds the maximum of the DMG1012T. Instead, I have selected the BSS214NWH6327XTSA1 for the power domains (continuous drain current of up to 1.5A)

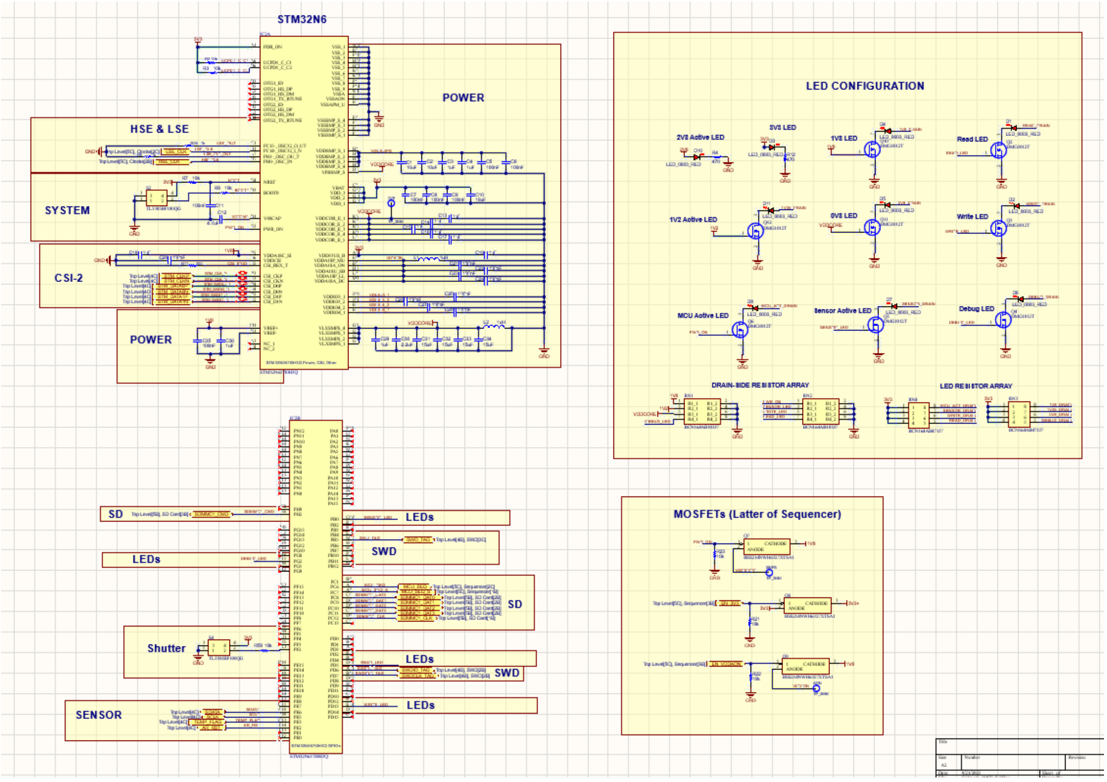


Figure 5: MCU schematic sheet from C2S Breakout Board Project.

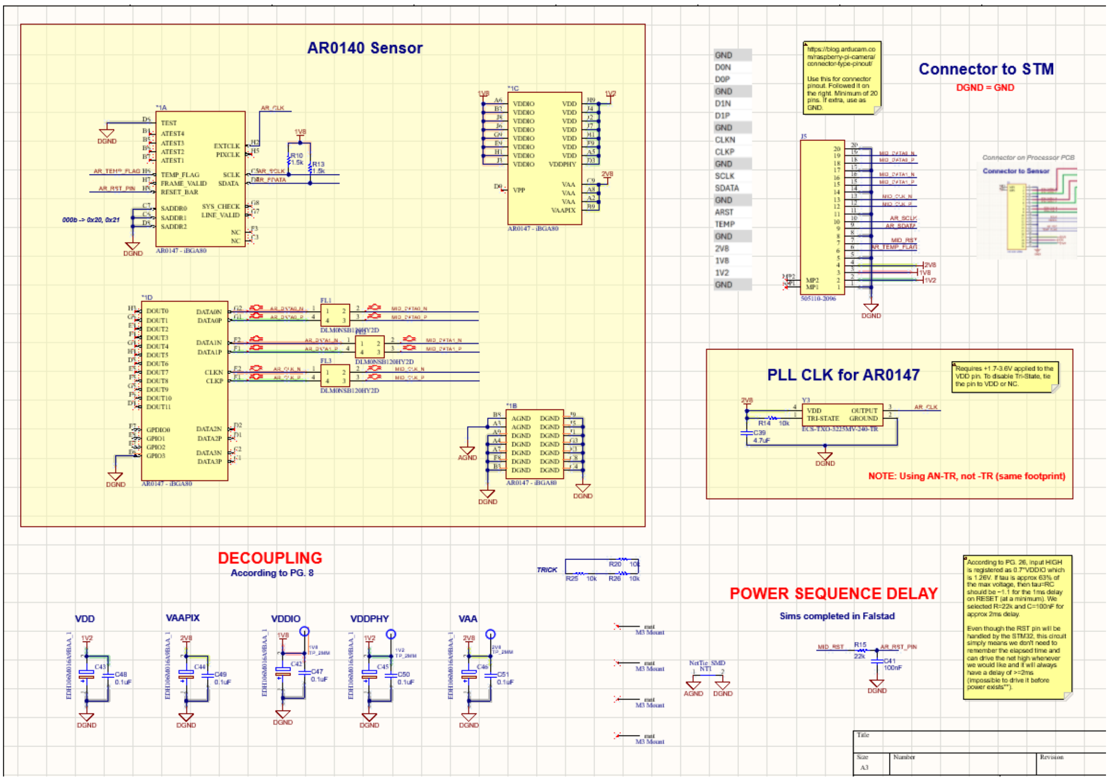
3.2.3 Image Sensor

Like the STM32N6, I followed the AR0147 datasheet on how to properly create the image sensor schematic. Although it wasn't specified in the datasheet, I included common mode chokes on the differential lines like the MIRA220 schematic.

For the connector to the processor, this guide was followed on how to properly set the pinout. I set every third connection to ground to ensure a reliable return path for all signals.

The most important design consideration with this schematic was the requirement of digital and analog ground (DGND/AGND). To understand how to work with different signal planes, the following resources from TI were used: 1 and 2. To connect both grounds, I used a simple star connection with a NetTie. I also set DGND as the equivalent ground on the processor PCB (GND=DGND on the connector).

I selected the ECS-TXO-3225MV-240-TR as the external PLL oscillator for the image sensor. This component can be operated on 3V3, however, I chose 2V8 so the oscillator will only start when the sensor's specific power domains are provided and doesn't 'burn' power when the sensor is not in use. Page 22 of the sensor datasheet specifies that the tEXTCLK signal must have a rise/fall of between 0.2ns and 12.5ns (12.5ns extrapolated from 0.3*tEXTCLK). These requirements are met by the selected oscillator since the oscillator datasheet says it has a rise/fall time of 5±0.5ns. Since the generated signal is a clipped sine wave, I can estimate the duty cycle is 50% since it's symmetric.



https://blog.arducam.com/raspberry-pi-camera/connector-type-pinout/

Use this for connector pinout. Followed it on the right. Minimum of 20 pins. If extra, use as GND.

Requires +1.7-3.6V applied to the VDD pin. To disable Tri-State, tie the pin to VDD or NC.

According to PG. 26, input HIGH is registered as 0.7*VDDIO which is 1.26V. If tau is approx 63% of the max voltage, then tau=RC should be ~1.1 for the 1ms delay on RESET (at a minimum). We selected R=22k and C=100nF for approx 2ms delay.

Even though the RST pin will be handled by the STM32, this circuit simply means we don't need to remember the elapsed time and can drive the net high whenever we would like and it will always have a delay of >=2ms (impossible to drive it before power exists**).

Figure 6: Sensor schematic sheet from C2S Breakout Board Project.

3.2.4 Clock

The C2S Breakout Board required three individual clocks for operation:

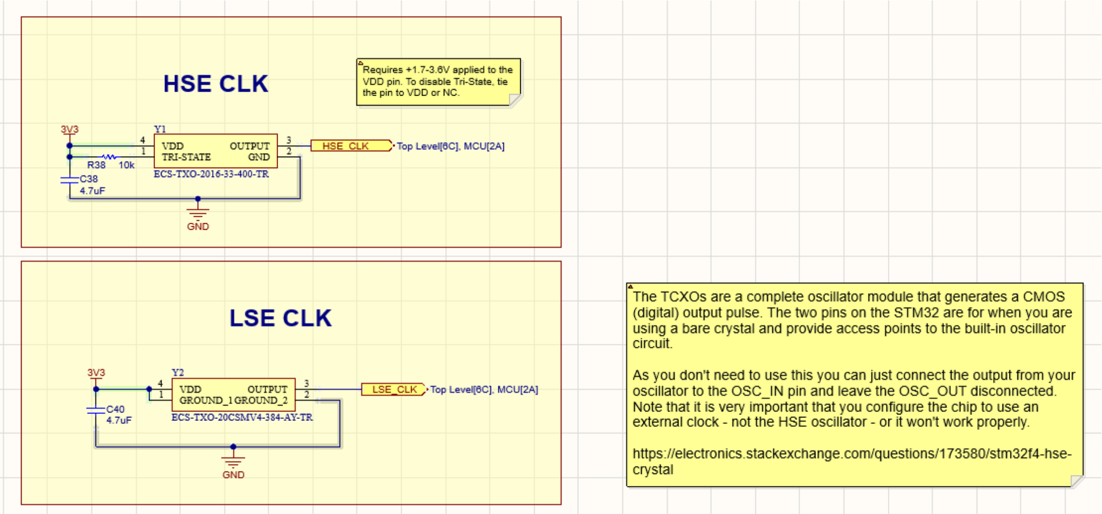
- High-speed 24MHz (HSE) oscillator for the STM32.
- Low-speed 32.768KHz (LSE) oscillator for the STM32.
- High speed 24MHZ external oscillator for the AR0147 image sensor (discussed in subsubsection 3.2.3, not here).

For the HSE configuration, I referred to the "Oscillator design guide" for STM32 clocks and obtained the following system requirements: For the N6 series (Table 6 of oscillator guide), I require a frequency range of 16-48MHz which is further verified in the datasheet with a typical frequency of 40MHz (Table 49 of datasheet). Additionally, they must operate at 3V3 or 1V8 supply voltage (since those are the only two supplies available pre-boot).

I selected the ECS-TXO-2016-33-400-TR TCXO oscillator to provide the HSE clock signal. I opted for a 40MHz oscillator opposed to the 24MHz selected in Phase 1 since the guides recommended to use 40MHz. I were not able to calculate $g_{\mathtt{mcrit}}$ since the $\mathtt{ESR}$ is not specified on the datasheet. However, since other ECS oscillators are recommended like the ECS-.327-6-16-TR3, I have assumed it is compatible. For the LSE oscillator, I selected the ECS-327TXO-33-TR. I opted for this component over others since it was the only available ECS TCXO oscillator in-stock at that frequency.

Since all three oscillators are TCXO, they already include internal circuitry (like the capacitor configuration) and simply need to be connected to the input and output pins of the STM32.

Each oscillator should continuously produce a clock signal from boot that doesn't interrupt startup or shut-down sequences.



Requires +1.7-3.6V applied to the VDD pin. To disable Tri-State, tie the pin to VDD or NC.

The TCXOs are a complete oscillator module that generates a CMOS (digital) output pulse. The two pins on the STM32 are for when you are using a bare crystal and provide access points to the built-in oscillator circuit.

As you don't need to use this you can just connect the output from your oscillator to the OSC_IN pin and leave the OSC_OUT disconnected. Note that it is very important that you configure the chip to use an external clock - not the HSE oscillator - or it won't work properly.

https://electronics.stackexchange.com/questions/173580/stm32f4-hse-crystal

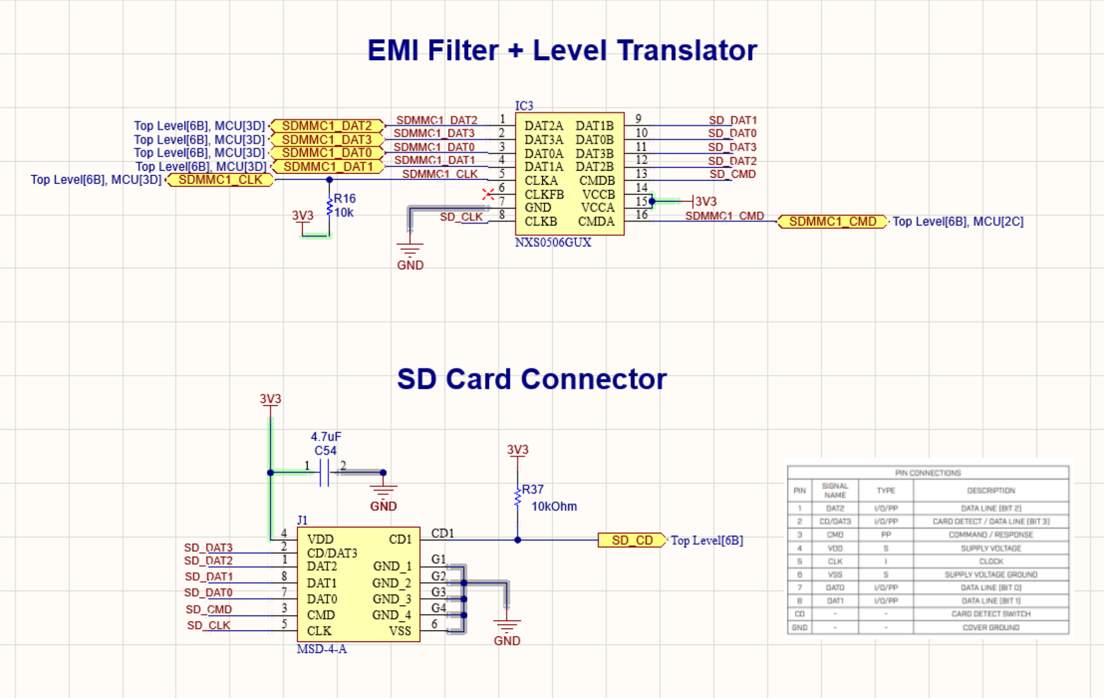
Figure 7: Clock/oscillator schematic sheet from C2S Breakout Board Project.

3.2.5 uSD Card

I used the getting started guide to construct the uSD Card schematic. Like the guide, I used the NXS0506GUX since the recommended component uses a BGA package that is too fine.

For faster readout speeds, I have connected the component in 4-bit mode, opposed to the simpler, but slower, 1-bit mode (connecting only DAT0).

Additionally, I added a decoupling capacitor to VCC using same value as the decoupling capacitors used on the image sensor.



PIN CONNECTIONS			
PIN	SIGNAL NAME	TYPE	DESCRIPTION
1	DAT2	I/O/PP	DATA LINE [BIT 2]
2	CD/DAT3	I/O/PP	CARD DETECT / DATA LINE [BIT 3]
3	CMD	PP	COMMAND / RESPONSE
4	VDD	S	SUPPLY VOLTAGE
5	CLK	I	CLOCK
6	VSS	S	SUPPLY VOLTAGE GROUND
7	DAT0	I/O/PP	DATA LINE [BIT 0]
8	DAT1	I/O/PP	DATA LINE [BIT 1]
CD	-	-	CARD DETECT SWITCH
GND	-	-	COVER GROUND

Figure 8: SD Card schematic sheet from C2S Breakout Board Project.

3.2.6 LPS

The Local Power Supply (LPS) schematic was mostly copied from the MIRA220 reference schematic. The LDO components were updated to support active components, and the 1V8 power line was modified to be generated when then 3V3 line was present, since the MCU requires 1V8 to boot. Moreover, Test points were added to the outputs of the LDOs as well as on the enable lines. Since each LDO uses the same package (SOT-25-5 / SMV), the schematic uses the 2V8 LDO as a placeholder. The selected LDOs were:

- TCR2EF28,LM(CT
- TCR2EF135,LM(CT

I were able to meet the following requirements from the datasheet:

- 3V3 input on V_{IN} with a $0.1\mu\text{F}$ decoupling capacitor (pin 1)
- $1\mu\text{F}$ ecoupling capacitor on the output (pin 5).

As mentioned in subsubsection 3.2.8, the system will require 2500mW in the worst case scenario. This is mostly attributed to $\mathtt{V_{DDCORE}}$, the most power-intensive domain (2000mW). The internal SMPS is supplied by the 1V8 power domain, which corresponds to a current requirement of 1.11A for just the STM32's SMPS (rounded to 1.25A to account for other components). Fortunately, the other power domains operate below the chosen LDO's current limit of 200mA. However, I still need to accomodate for $\mathtt{V_{DDCORE}}$. As a result, I selected the TPS62061DSGR as the buck converter to supply the 1V8 power domain. It can operate up to 1.6A output current at the required 1V8 domain.

Please see subsubsection 3.2.2 for more detail about the resistor selection of the LEDs.

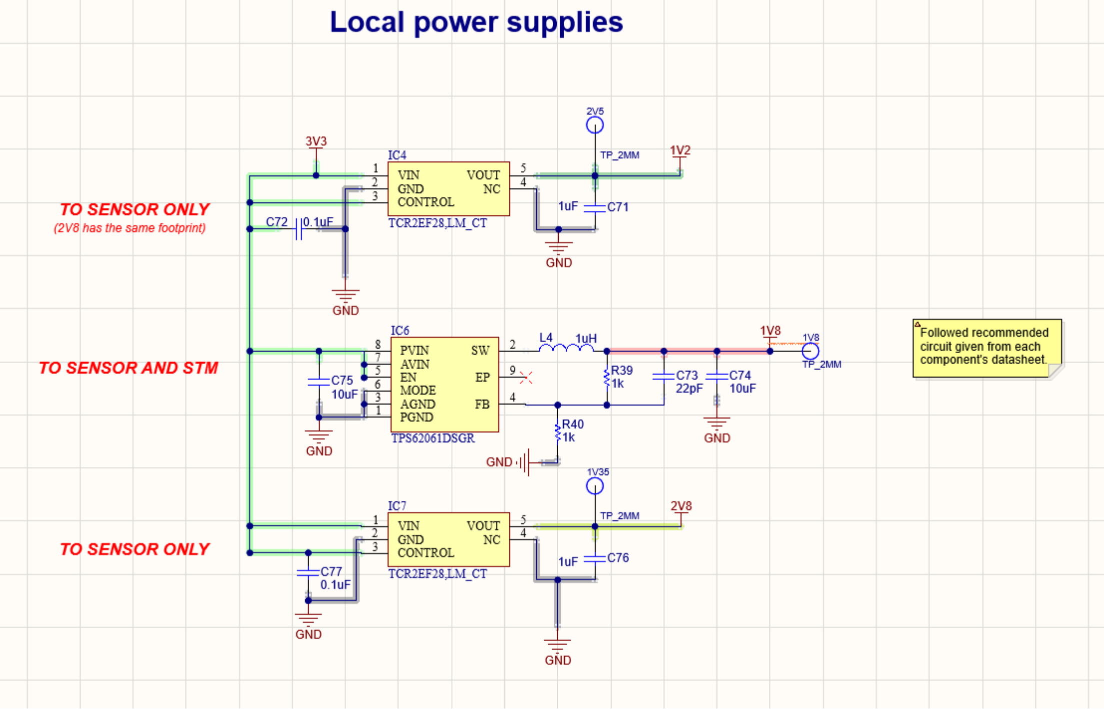


Figure 9: LDO schematic sheet from C2S Breakout Board Project.

3.2.7 SWD Connector

The SWD connector followed the Phil's Lab guide on STM32 PCB design, as well as referencing the ADCS Integration Board for the component.

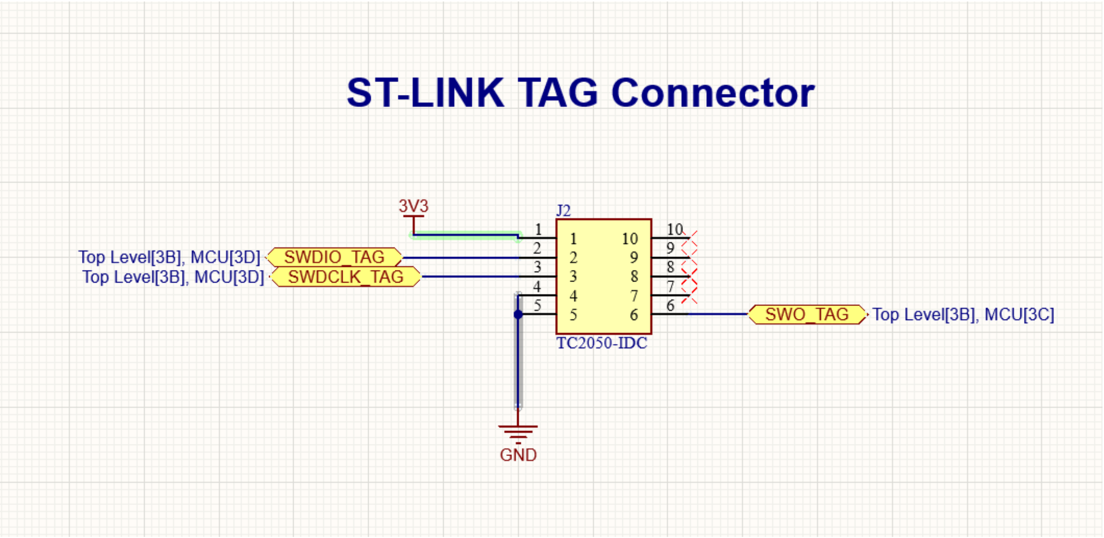


Figure 10: SWD TAG schematic sheet from C2S Breakout Board Project.

3.2.8 USB-C Connector

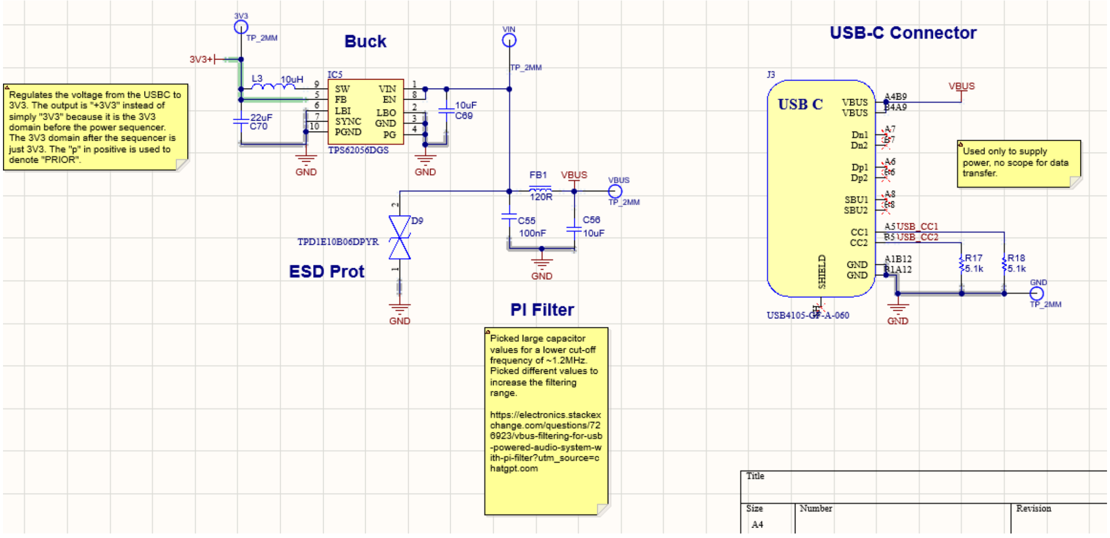
The USB-C connector was implemented as a source of power for the breakout board. Instead of needing to rely on FPC connectors or headers to jumper +3V3 power lines, I can simply use the power pins on the USBC connector. This enabled us to make use of other power supplies like powerbanks, rather than relying on stray wires or peripheral PCBs.

The schematic was referenced from the ADCS Integration Test Board. Unused pins were either left as NCs or pulled to ground. Test points were added on the main power levels (GND, VBUS, VIN, 3V3) to confirm potential readings during testing.

The ferrite bead and capacitors form a pi-filter (C-L-C) network. The input capacitor shunts high-frequency noise from the connector before it can reach the PCB. The ferrite bead acts as a supressor to high-frequency noise (low-pass filter) by absorbing the signal and dissipating it as heat. At DC and low frequencies, it acts as a short-circuit, allowing typical signals to pass. The output capacitor provides local decoupling for the PCB and helps smooth out any remaining noise. Using two different valued capacitors help increase its performance over a wider range of frequencies. With an inductor value of $L = 190\mathtt{nH}$, I chose capacitor values of $C_1 = 100\mathtt{nF}$ and $C_2 = 10\mu\mathtt{F}$ (subsection A.1) to create a cut-off frequency $C_{eq} = 1.2\mathtt{MHz}$. I chose these values over others since they were seperated by two magnitudes and these values were already being ordered, which saves on unique component quantity, and the maximum value I can select for a capacitor on a USB line is $10\mu\mathtt{F}$.

To introduce surge protection for electrostatic discharge, the schottky diode helps clamp voltage spikes to ground to protect other components in the PCB from being damaged. Although voltage spikes may not immediately break components, over time, they can wear down.

Lastly, I needed an effective and reliable method to maintain a +3V3 potential. If $\mathtt{V_{DDCORE}}$ requires 1500mW at its peak, I can estimate that the internal SMPS will require 2000mW to create the power domain (at 75% efficiency - Figure 18 of the STM datasheet illustrates the efficiency is 85%, however, I have selected 75% for padded margins). From the N6 power supply guide, our maximum power draw is approximately 2500mW (worst-case scenario). At a 3V3 voltage supply, this corresponds to 750mA of current (no losses). Unfortunately, the LDOs used in subsubsection 3.2.6 can only output a maximum of 200mA of current. Although I could chain multiple in parallel to reach the required current, I found it best to continue with a more efficient buck converter. I selected the TPS62056DGS as the buck converter to supply the main power domain to the PCB. It can output 800mA at 3V3 with an input of 2.7-10V at a very low operating supply current (12uA).



Regulates the voltage from the USBC to 3V3. The output is "+3V3" instead of simply "3V3" because it is the 3V3 domain before the power sequencer. The 3V3 domain after the sequencer is just 3V3. The "p" in positive is used to denote "PRIOR".

Picked large capacitor values for a lower cut-off frequency of ~1.2MHz. Picked different values to increase the filtering range.

https://electronics.stackexchange.com/questions/726923/vbus-filtering-for-usb-powered-audio-system-with-pi-filter?utm_source=chatgpt.com

Used only to supply power, no scope for data transfer.

Figure 11: USB-C Connector schematic sheet from C2S Breakout Board Project.

3.2.9 Power Sequencer

Unlike other STM32 MCUs like the H7, the N6 processors require a specific startup pattern to properly boot. As found on pg. 27 of the datasheet and pg. 9-10 of the getting started handbook, the startup sequence is:

1. V_{DD} and $V_{DDA18ON}$
2. V_{DDSMPS} via PWR_ON

From the power sequencer schematic (Figure 13), the component sequentially enables the FLAG3, FLAG2, FLAG1 pins. These enable pins are connected to the base of MOSFETs (see MCU schematic, Figure 5), which allow the sequencer to turn the power lines on and off in the required pattern. FLAG1 has been left as an NC since the final enable trigger is instead from PWR_ON. The PWR_ON pin connects to the base of the MOSFET on the VDDSMPS line to drive high SMPS when the MCU requires it.

To prevent the STM32 from prematurely shutting down from a hard-reset (switching the power switch, S1, off), a soft-reset function has been added. This means that the power sequencer, IC1, is controlled by two components, the switch and a MOSFET. Once the PCB is first turned on by closing the switch, the power regulator circuit properly boots the STM32. Once the STM is operational, it will drive the base of the MOSFET high via MCU_SEQ. This means that even if the power switch is opened, the sequencer remains enabled because the STM is holding the MOSFET circuit. To ensure that the STM actually shuts down, the MCU_SEQ_S (S meaning STATE) carries the state of the switch to the STM. This effectively tells the STM ”it’s time to shut down”, allowing it to finalise everything before disabling the MCU_SEQ pin and shutting down.

Since there did not appear to be any voltage drop across the switch stated in the datasheet, a voltage divider was used to read the state to protect the GPIO from any surges. A large value R_2 was selected since the net is connected to the main input power and little current is required to read the voltage.

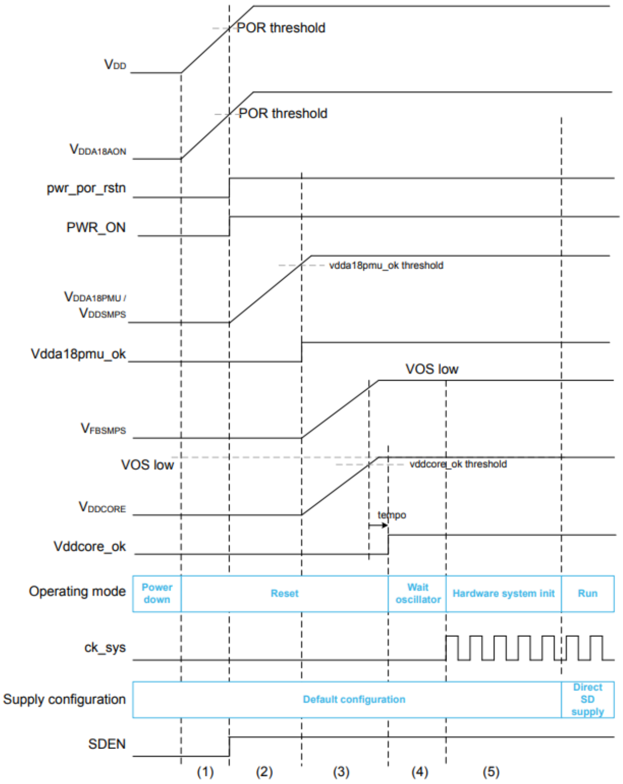
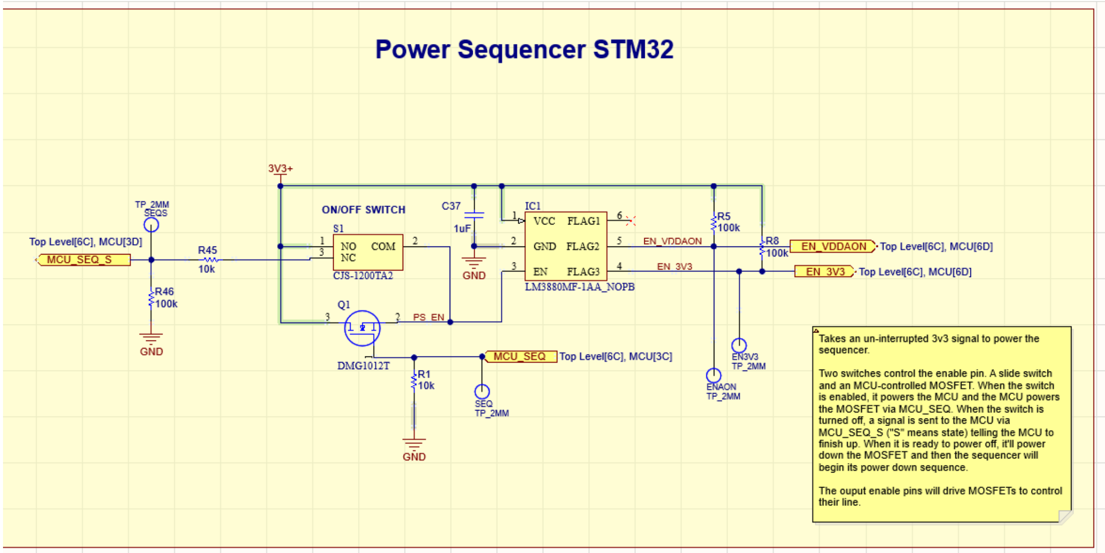


Figure 12: STM32N6 startup with VDDCORE supplied directly from internal SMPS step-down converter



Takes an un-interrupted 3v3 signal to power the sequencer.

Two switches control the enable pin. A slide switch and an MCU-controlled MOSFET. When the switch is enabled, it powers the MCU and the MCU powers the MOSFET via MCU_SEQ. When the switch is turned off, a signal is sent to the MCU via MCU_SEQ_S ("S" means state) telling the MCU to finish up. When it is ready to power off, it'll power down the MOSFET and then the sequencer will begin its power down sequence.

The ouput enable pins will drive MOSFETs to control their line.

Figure 13: Power Sequencer schematic sheet from C2S Breakout Board Project.

3.3 PCB Layout and Review

3.3.1 Selecting Manufacturers

I selected JLCPCB as the PCB manufacturer since they are capable of producing the PCBs within their capabilities, and I (PAST) have always used JLC to produce our PCBs.

Because of the BGA package size, I concluded that it was not feasible to solder those two components by hand. To place the BGAs, I consulted with Peter from AT&MWA to discuss their capabilities and whether they can place the two components or not. I found that:

- They are capable of placing BGAs of up to 0.2mm diameter and 0.5mm pitch.
- I should aim for a solder mask dam of 4mil (0.102mm) so the board can be fabricated with solder mask around each pad. I can use 2mil if required.
- I should produce a few spare boards in case I want to stress one or two out during testing.
- The two ICs should be placed in around 30-40 minutes a board. This would costs around $35-$47AUD+GST per board.
- They will produce the stentil themselves, and they will recommend the best way to panelise the boards to suit their machines.
- I are able to place the rest of the components first with the BGAs being last.

3.3.2 Current Carrying Capacity

Using the Saturn PCB Toolkit, I can estimate the current carrying capacity of the BGA traces with the following parameters:

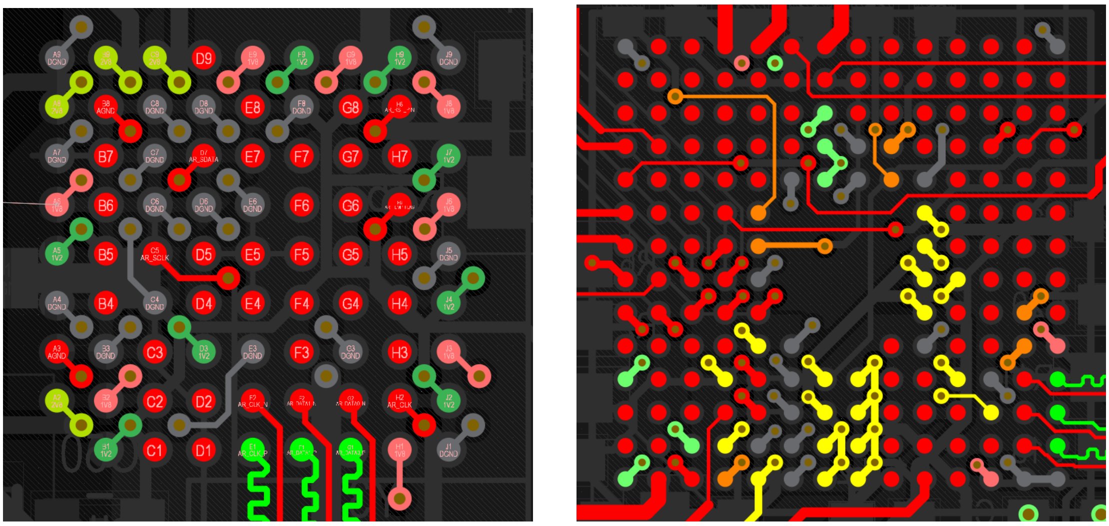
Parameter	**Value**	**Unit**
Conductor Width	4	mil
Conductor Length	50	mm
PCB Thickness	62	mil
Plane Present	Yes	-
Distance to Plane	10	mil
Conductor Current	1.13	A

Table 7: Current carrying capacity estimation of BGA traces in C2S Breakout Board.

As per Table 7, the maximum current on a 4mil trace is 1.1A, which is less than the typical current requirement. The current carrying capacity is not a concern for the STM32N6 since its power traces are 0.2mm, which can carry 1.8A, equal to the maximum 1.8A drawn by the MCU.

3.3.3 BGA Fanout

I encountered clearance issues with the AR0147 BGA fanout. Unlike most BGAs which have a ball radius of 0.5-1.0mm with large pitches, the sensor uses a small package profile. This package made it difficult to properly route the component. fanout methods like Via-in-Pad were not possible as such a small scale, simply because the PCB capabilities didn't permit small enough vias. For most PCB manufacturers, the minimum via diameter size is 0.2mm, with a 0.15mm annular ring (0.35mm total diameter). Since the via was larger than the BGA pad, via-in-pad was not a feasible option. I then opted for the dogbone fanout method which admit to the manufacturer's trace width and clearance of 3.5mil (see Figure 14a). Unlike the sensor, the STM32N6 has a more coarse BGA package which allowed us to fanout without issues. In this case, I opted for the dog bone fanout since it allowed us to place our ground vias closer to the power vias. It also provided more clearance routing diagonally, allowing us to use thicker traces for a better current carrying capacity.



(a) Fanout of AR0147 image sensor.

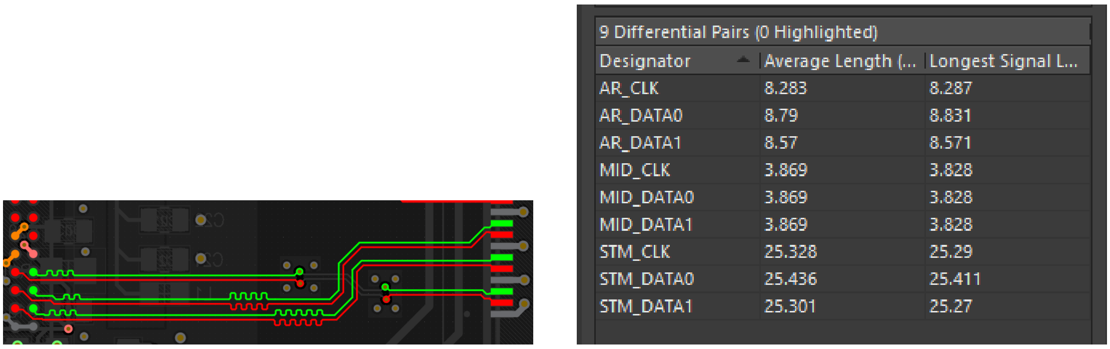
(b) Fanout of the STM32N6 MCU.

Figure 14: BGA fanout methods used in the C2S Breakout Board PCB.

3.3.4 CSI-2 Routing

I used TI's high-speed guidelines for routing the differential signals. From Table A-7 of the guide (pg. 19), I found that the differential pair impedance is $100 \pm 15\Omega$, with lane skew of $40\texttt{ps}$, with 2 total vias per pair. Using Altium's impedance calculator and guide, I found the required separation distance between the 3.5mil differential traces to be 3.542mil, or approximately the trace width. According to the STM32N6 getting started guide, I require a length match of $\pm 5\texttt{mil}$, and the MIRA220 datasheet (although not used) specifies a delay match of $40\texttt{ps}$ (which is greater than 5mil on the microstrip). I length matched the data pairs to achieve this low delay ($\pm 2\texttt{mil}$). Additionally, the getting started guide specifies a length match of $\pm 100 textttmil$ between the data and clock lines, which was achieved.

It was not possible to keep all the differential pairs on the top layer because of how the MIRA and STM pins aligned, one pair always ended up using vias. To ensure that each signal was grounded, I made sure to place sitching vias around the layer changes.



9 Differential Pairs (0 Highlighted)

Designator	Average Length (...	Longest Signal L...
AR_CLK	8.283	8.287
AR_DATA0	8.79	8.831
AR_DATA1	8.57	8.571
MID_CLK	3.869	3.828
MID_DATA0	3.869	3.828
MID_DATA1	3.869	3.828
STM_CLK	25.328	25.29
STM_DATA0	25.436	25.411
STM_DATA1	25.301	25.27

(a) Differential traces on the PCB.

(b) Signal lengths of each differential signal.

Figure 15: MIPI CSI-2 differential pairs on the C2S Breakout Board.

3.3.5 Debug LEDs

The TX and RX pins are **not representative of UART**. Since there was not enough space to add "READ" and "WRITE", I opted for the shorthand notation used in UART. In this case, RX means the MCU is receiving / reading from the image sensor, and TX means the MCU is transmitting / writing to the uSD card.

The power LEDs (3V3, 2V8, 1V8, 1V2, 0V8) show whether each line is active or not.

The ”AR0” and ”STM” LEDs show whether each component is active or not. They are each connected to a GPIO pin on the STM32. They will turn on when the STM32 is operating and can utilise its GPIO pins, and whether it can talk to the image sensor.

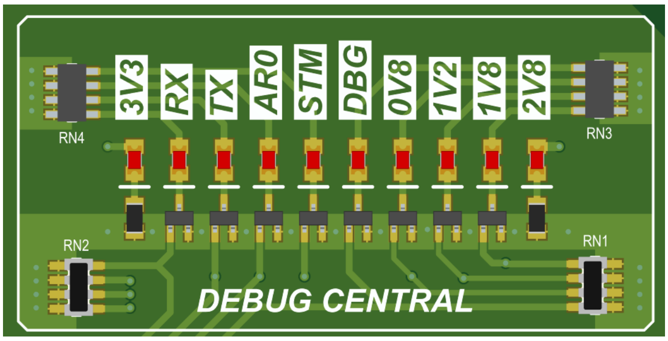


Figure 16: Debug LEDs on C2S Breakout Board.

4 C2Sv0.1

As mentioned in section 3, the version was shelved because of the high price point. Unlike version 0 which used MIPI CSI-2, v0.1 focuses on the DCMI (DSI) interface (for more information about DCMI, read here). DCMI is an older and slower interface for select STM32 chips, so, it's cheaper than a fast and modern interface like CSI-2. Unlike CSI-2, DCMI is available on non-BGA packages, which means I can avoid using the ENIG surface finish across the entire processor, and use much larger via sizes. Although the image sensor still requires a BGA package, sensors with DCMI often tend to have larger packages (because they are older), which means I can use larger vias to save more money.

Overall, this version has an arbitrary hard-limit of $250AUD (inclusive of all costs: components, manufacturing, shipping, etc). Exceeding this limit would make the budget on future revisions much tighter, especially if this version doesn't fully work and needs to be revised. The breakdown of the budget is as follows:

Item	**Cost ($AUD)**
PCBs (shipping inc.)	88.27
AT&MWA BGA Fee	50
2x AR0141	?
2x STM32F7x0	?
Other Componenets	?
Total	$240AUD

Table 8: Cost breakdown of C2Sv0.1.

From v0, the most notable lessons learnt were:

- Keeping the image sensor and processor on separate PCBs. Although this version can no longer use a nucleo board like the N6 for debugging, working v0.1 PCBs can be used to debug prototype (v1) PCBs.
- Spending more time planning out where to place the components before routing. With v0, I only landed on the desired PCB layout after three previous attempts at making a PCB because I was very eager to finish the PCB and very quickly got stuck in a tangled mess of an unplanned layout. I've found it best just to copy the PCB layout into Adobe Photoshop or MS Paint and draw out the lines to see if they'd actually work before doing so. Although I did end up changing my original layout for v0.1, I only required minor modifications, so the planning was an excellent step.
- Finding and using reference schematics. Although it may sometimes feel like cheating when following another design, it can save a lot of mistakes. If I hadn't found the MIRA220 reference schematic before starting v0, I would have struggled a lot with developing the schematic, and even more when trying to debug my errors. Especially for a new topic like camera design, having reference schematics is a must. Despite this, I still try and develop the schematic myself, and use the reference as a checklist/guide to know whether I've done the correct thing or not.
- Utilising free resources and documents. There are plenty of complete-documents published by ST, TI, etc. that go in-depth with some critical pieces of knowledge. Before starting v0, I had never known about DGND and AGND, but I read through the TI document and learnt more about what they are, how they differ, and what to do with them. ST has released lots of resources as well, virtually every topic has been covered by them (especially in camera design).

4.1 Schematic Design & Component Selection

4.1.1 Overview

This phase involves designing the camera circuit schematic using Altium Designer, laying the groundwork for subsequent PCB development and component selection.

The schematic is divided into eight sub-sheets:

1. MCU
2. Image Sensor
3. uSD Card
4. LPS
5. TAG Connector

6. USBC Connector
7. Memory

Only critical paths were included on the Top-Level schematic. Routes like general power and ground were not included.

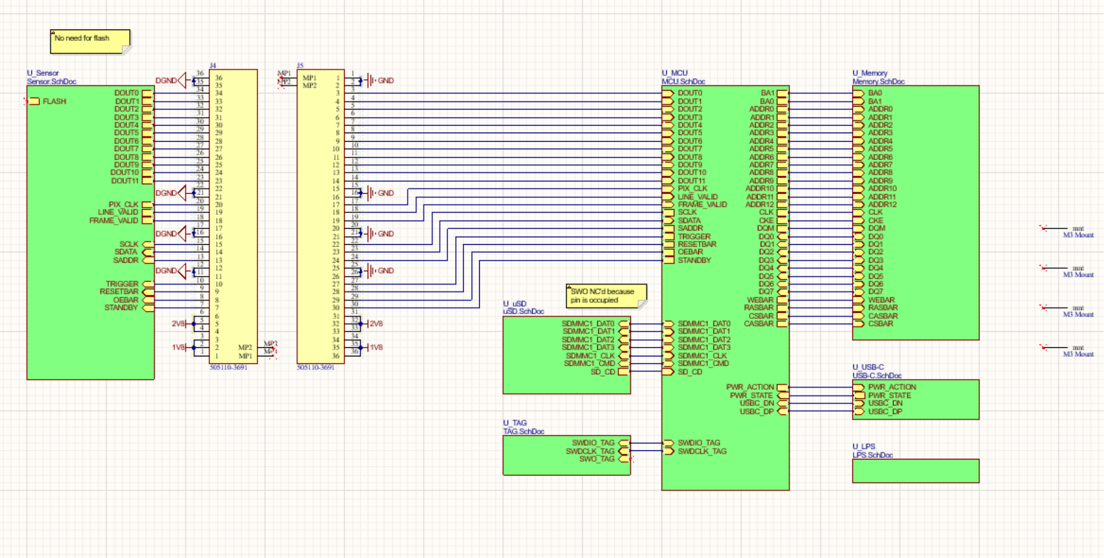


Figure 17: Top-Level schematic sheet from C2S Breakout Board Project.

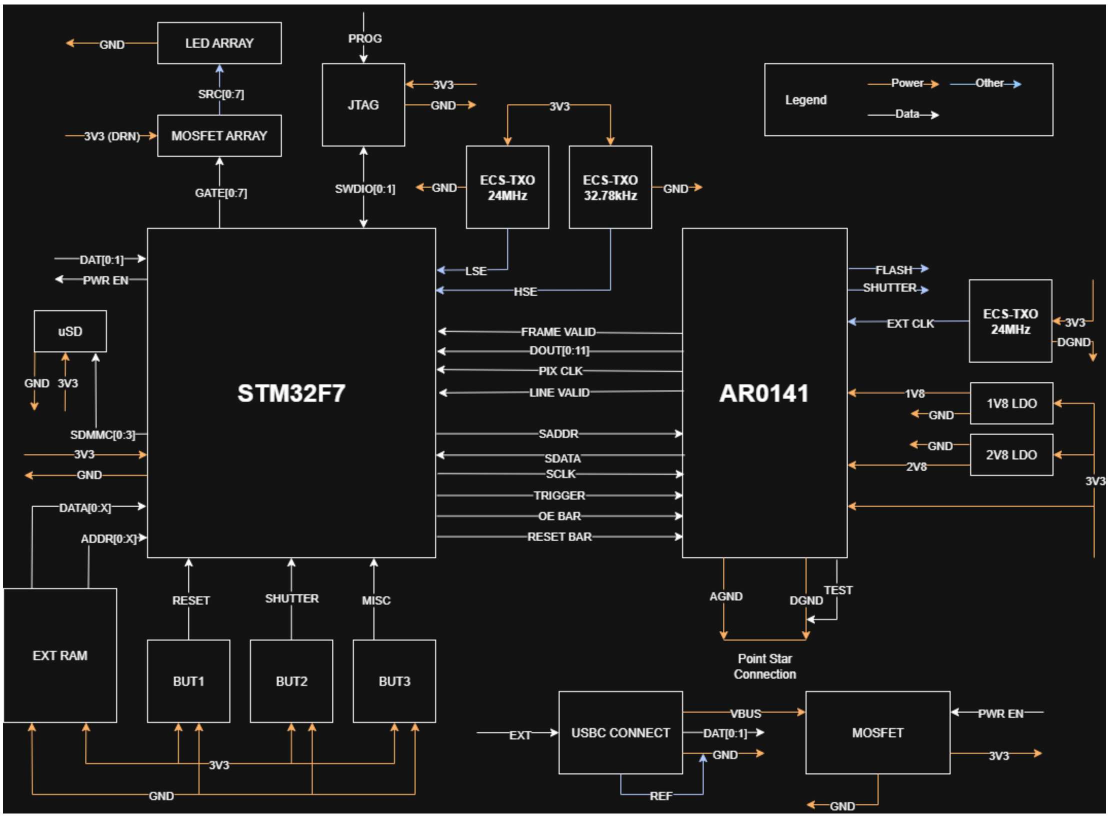


Figure 18: General block diagram of C2S breakout board. Components like resistors, capacitors and suggested circuits have been omitted from the diagram. This is meant to show the core functionality between the core components.

4.1.2 Image Sensor

The image sensor is a very difficult category to save money on. Most will cost upwards of $50AUD for decent/quality sensors, dipping below that line whilst seeking for the same quality has its drawbacks. The most notable drawback is the availability of the sensors, most that fit the requirements are either obsolete, or in its last-stock but requires bulk purchases. Some prime examples of this are the AR0544 and AR0830. The second drawback is the package. Image sensors are designed for compact systems, so finding sensors that have a BGA package within manufacturing capabilities is a challenge of its own. Sensors like the MIRA016 and AR0144 were both within our price range, but their BGA ball diameter and pitch no longer made it feasible to consider them. This left us with the AR0141CS2M00SUEA0-DPBR, the M meaning mono or monochrome. Although there were cheaper, colour image sensors like the MT9M114EBLSTCZ-CR, they were all affected by those two drawbacks. For the MT9, its package was too fine.

Although this model of the AR0141 uses a monochrome filter, I concluded that this would not be an issue since the only difference between is the number of channels. Both colour and monochrome images undergo the exact same process, so for a breakout board to test functionality, it was not an issue.

Since v0 and v0.1 share similar sensors, it was easy to develop the schematic. The only notable differences between them was the parallel interface. Since both sensors are from the same family, they both required a similar number of power, ground and data pins (just with different voltages).

To make the module vacuum-compatible, I replaced the electrolytic decoupling capacitors that were recommended with ceramic capacitors so I wouldn't encounter any issues during vacuum testing.

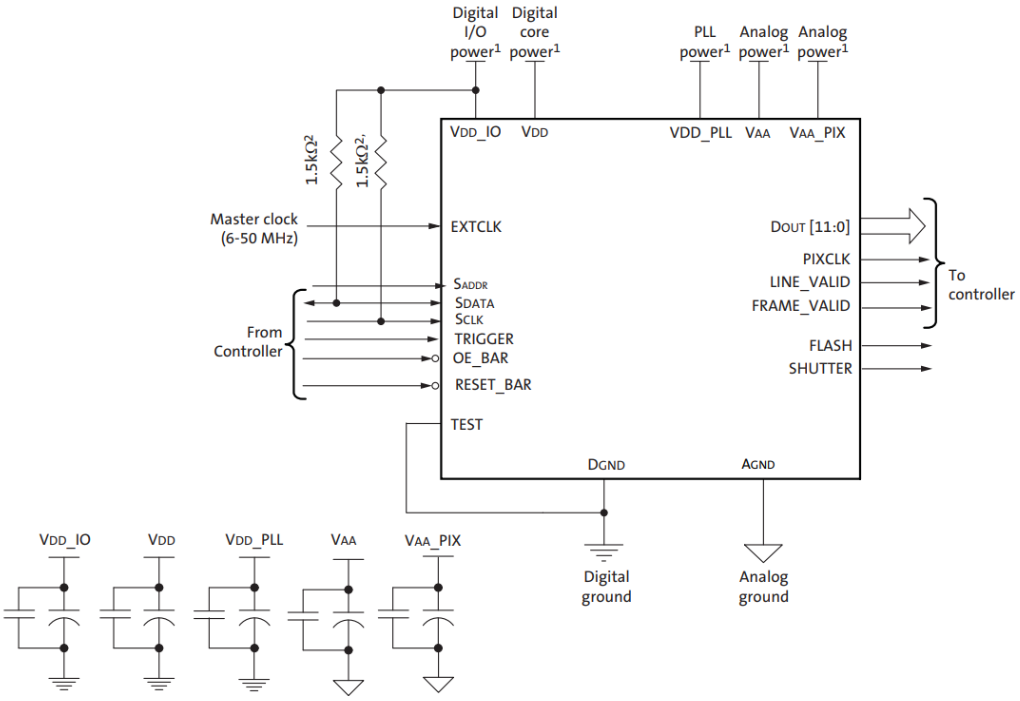


Figure 19: Typical configuration for parallel pixel data interface on AR0141CS2M00SUEA0-DPBR.

Since the board required a larger connector, I selected the same 36-pin FPC connector used on FISHv1.

Furthermore, to minimise the complexity of the board, I only kept the bare essentials to minimise production costs (because of the ENIG surface-finish).

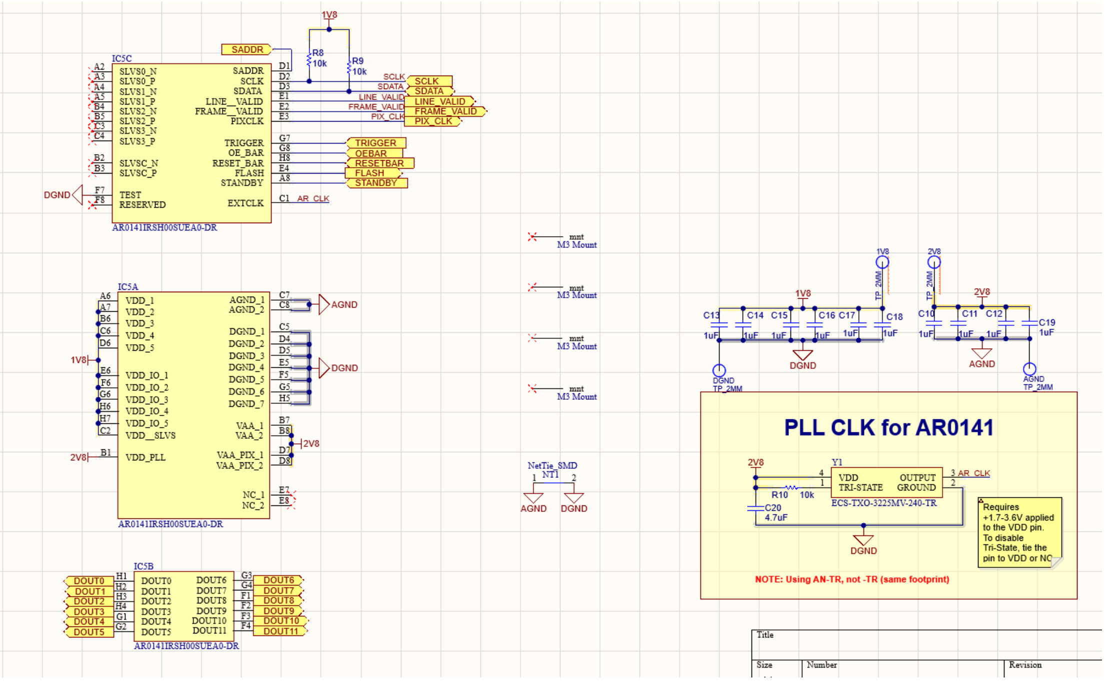
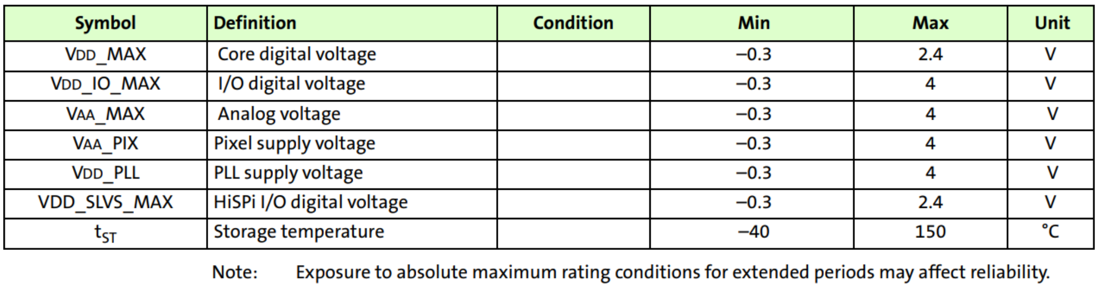


Figure 20: Image sensor schematic sheet from C2Sv0.1.

The maximum characteristics from the datasheet (Table 26, page 53) are:



Symbol	Definition	Condition	Min	Max	Unit
VDD_MAX	Core digital voltage		–0.3	2.4	V
VDD_IO_MAX	I/O digital voltage		–0.3	4	V
VAA_MAX	Analog voltage		–0.3	4	V
VAA_PIX	Pixel supply voltage		–0.3	4	V
VDD_PLL	PLL supply voltage		–0.3	4	V
VDD_SLVS_MAX	HiSPi I/O digital voltage		–0.3	2.4	V
t_{ST}	Storage temperature		–40	150	°C

Note: Exposure to absolute maximum rating conditions for extended periods may affect reliability.

Figure 21: Absolute maximum ratings of AR0141CS2M00SUEA0-DPBR

4.1.3 MCU

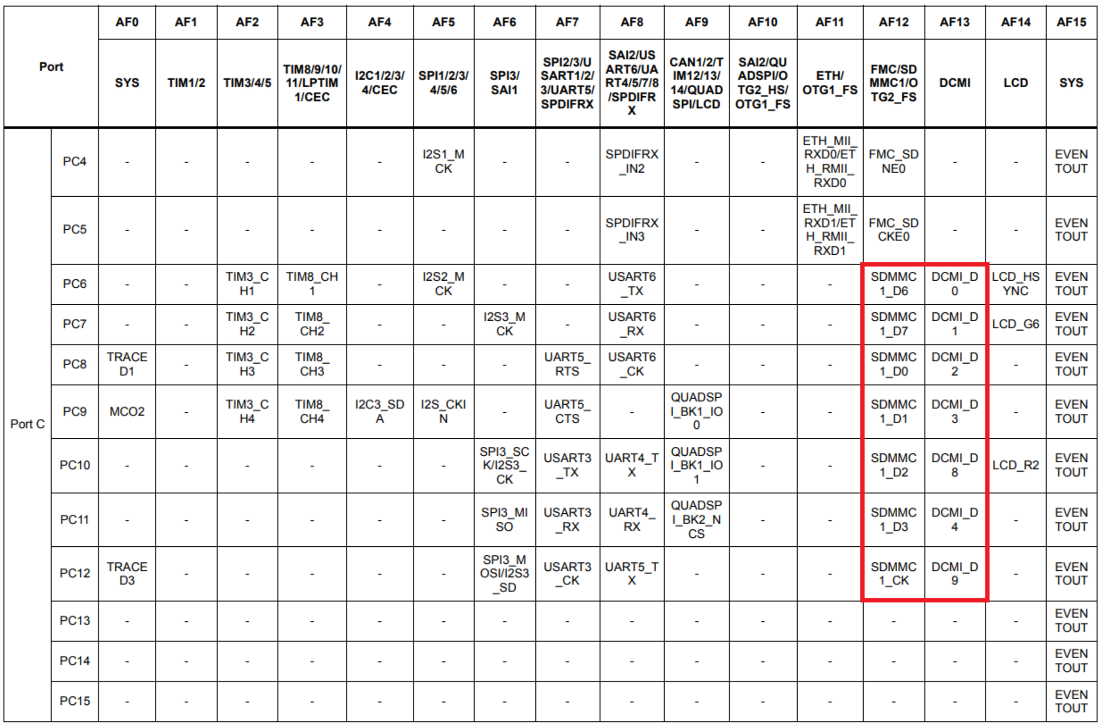
There were three STM32 families to pick from that met the requirements of the system; (1) F4; (2) F7; (3) H7.

	STM32F4	**STM32F7**	**STM32H7**
Core	Cortex-M4, 168 MHz	Cortex-M7, 216 MHz	Cortex-M7, up to 480 MHz
RAM	2 MB / 256 KB	512 KB / 128 KB	2 MB / 1 MB
FMC	Yes	Yes	Yes
DCMI	Yes	Yes	Yes, higher speed + parallel capture
DMA	Moderate	High	Very High (AXI + AHB buses)
Max Pixel Rate	24 MP/s	48 MP/s	80 MP/s
FPU / DSP	Single-precision FPU	Single-precision FPU	Single-precision FPU + Chrom-accel (optional)
Use Case	Simple camera + SDRAM buffering	Medium-res camera + basic image processing	High-res camera + advanced image processing and graphics
Power Consumption	100-120 mA core, +20-30 mA with DCMI/FMC	120-160 mA core, +30-50 mA with DCMI/FMC	200-300 mA core, +50-80 mA with DCMI/FMC
Price	Moderate	Moderate / higher	Higher-end

Table 9: Comparison of STM32F4, F7, and H7 for Image Processing Applications

Although the H7 offers more features and higher processing efficiency, its power consumption and cost make it excessive for the current application. The F4 is slightly older, but the F7 provides an excellent middle ground, delivering many of the H7's advantages while remaining comparable to the F4 in both price and power consumption. In-case the F7 does not pan out, I would select the H7 over the F4 because it is newer and has been used in PAST projects (better supported within the PAST community).

I followed the getting started guide and datasheet to connect all the pins. The guide included a very handy reference guide (page 38) which made the process a lot easier. I encountered an issue whilst pin-mapping since the DCMI and SDMMC1 interface both share the same pins. Because of this, I assigned the SDMMC1 datapins to other GPIO pins with the intent to remap them in the software. Other than that, I did not encounter any other issues with pin-mapping.



Port		AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7	AF8	AF9	AF10	AF11	AF12	AF13	AF14	AF15
		SYS	TIM1/2	TIM3/4/5	TIM8/9/10/11/LPTIM1/CEC	I2C1/2/3/4/CEC	SPI1/2/3/4/5/6	SPI3/SAI1	SPI2/3/USART1/2/3/UART5/SPDIFRX	SAI2/USART6/UART4/5/7/8/SPDIFRX	CAN1/2/TIM12/13/14/QUADSPI/LCD	SAI2/QUADSPI/OTG2_HS/OTG1_FS	ETH/OTG1_FS	FMC/SDMMC1/OTG2_FS	DCMI	LCD	SYS
Port C	PC4	-	-	-	-	-	I2S1_MCK	-	-	SPDIFRX_IN2	-	-	ETH_MII_RXD0/ETH_RMII_RXD0	FMC_SDNE0	-	-	EVENTOUT
	PC5	-	-	-	-	-	-	-	-	SPDIFRX_IN3	-	-	ETH_MII_RXD1/ETH_RMII_RXD1	FMC_SDCKE0	-	-	EVENTOUT
	PC6	-	-	TIM3_CH1	TIM8_CH1	-	I2S2_MCK	-	-	USART6_TX	-	-	-	SDMMC1_D6	DCMI_D0	LCD_HSYNC	EVENTOUT
	PC7	-	-	TIM3_CH2	TIM8_CH2	-	-	I2S3_MCK	-	USART6_RX	-	-	-	SDMMC1_D7	DCMI_D1	LCD_G6	EVENTOUT
	PC8	TRACED1	-	TIM3_CH3	TIM8_CH3	-	-	-	UART5_RTS	USART6_CK	-	-	-	SDMMC1_D0	DCMI_D2	-	EVENTOUT
	PC9	MCO2	-	TIM3_CH4	TIM8_CH4	I2C3_SDA	I2S_CKIN	-	UART5_CTS	-	QUADSPI_BK1_IO0	-	-	SDMMC1_D1	DCMI_D3	-	EVENTOUT
	PC10	-	-	-	-	-	-	SPI3_SCK/I2S3_CK	USART3_TX	UART4_TX	QUADSPI_BK1_IO1	-	-	SDMMC1_D2	DCMI_D8	LCD_R2	EVENTOUT
	PC11	-	-	-	-	-	-	SPI3_MISO	USART3_RX	UART4_RX	QUADSPI_BK2_NCS	-	-	SDMMC1_D3	DCMI_D4	-	EVENTOUT
	PC12	TRACED3	-	-	-	-	-	SPI3_MOSI/I2S3_SD	USART3_CK	UART5_TX	-	-	-	SDMMC1_CK	DCMI_D9	-	EVENTOUT
	PC13	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EVENTOUT
	PC14	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EVENTOUT
	PC15	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EVENTOUT

Figure 22: Example of overlap between SDMMC1 and DCMI pins from STM32F750Z8T6 datasheet (Table 12, page 71).

I was able to recycle the same LED, HSE and LSE configuration from v0. I removed two of the LEDs (no debug LED and VDDSMPS) to make the circuit fit nicely with the resistor arrays since they were no longer needed.

Just to slighlty increase the customisation of the board, I included an extra push-button for a miscellaneous purpose like starting a timelapse, or a timed-photo. The STM's software would need to be re-uploaded everytime the button's purpose is changed, but I doubt this would need to occur often throughout testing (maybe for outreach purposes).

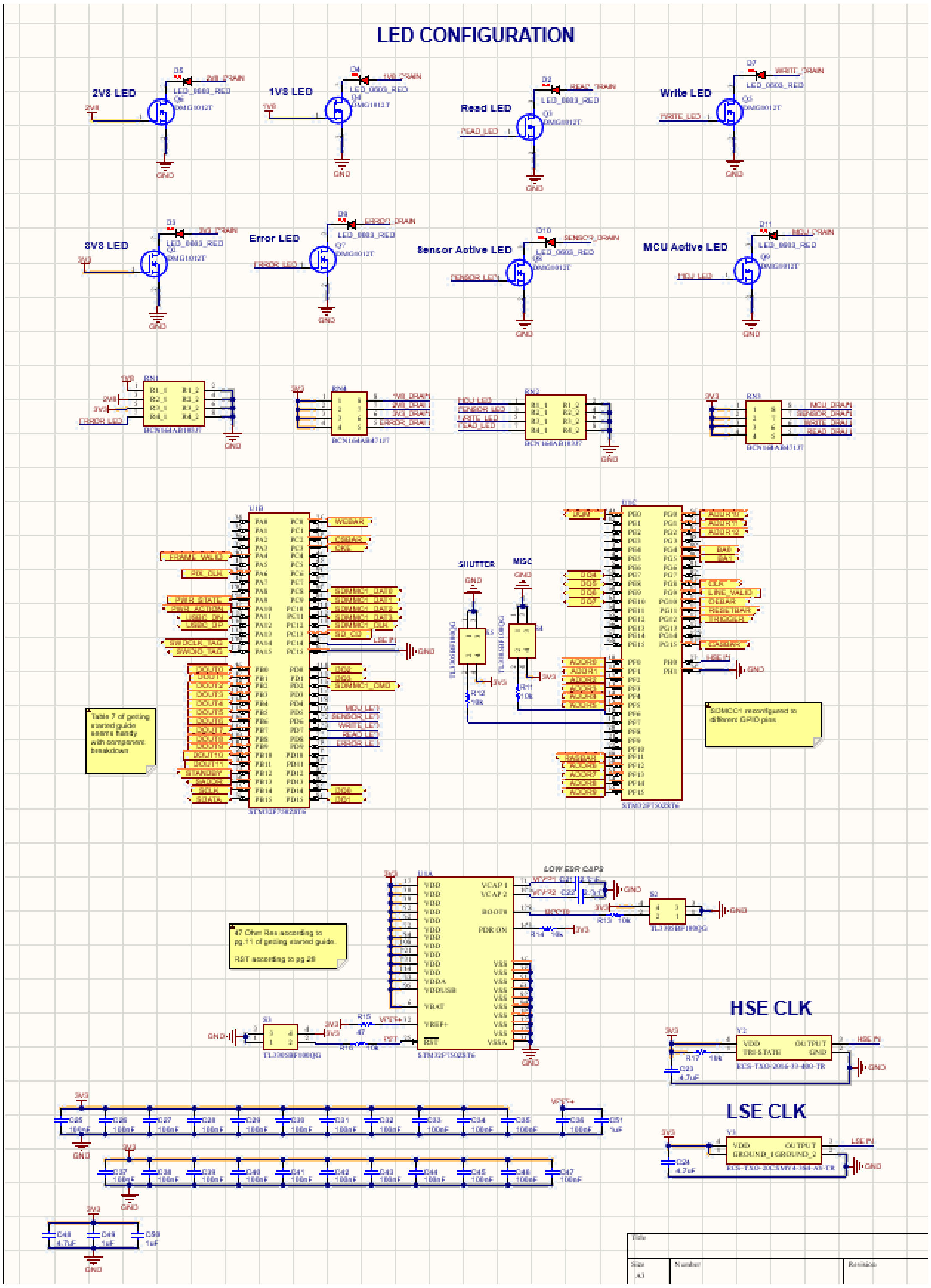
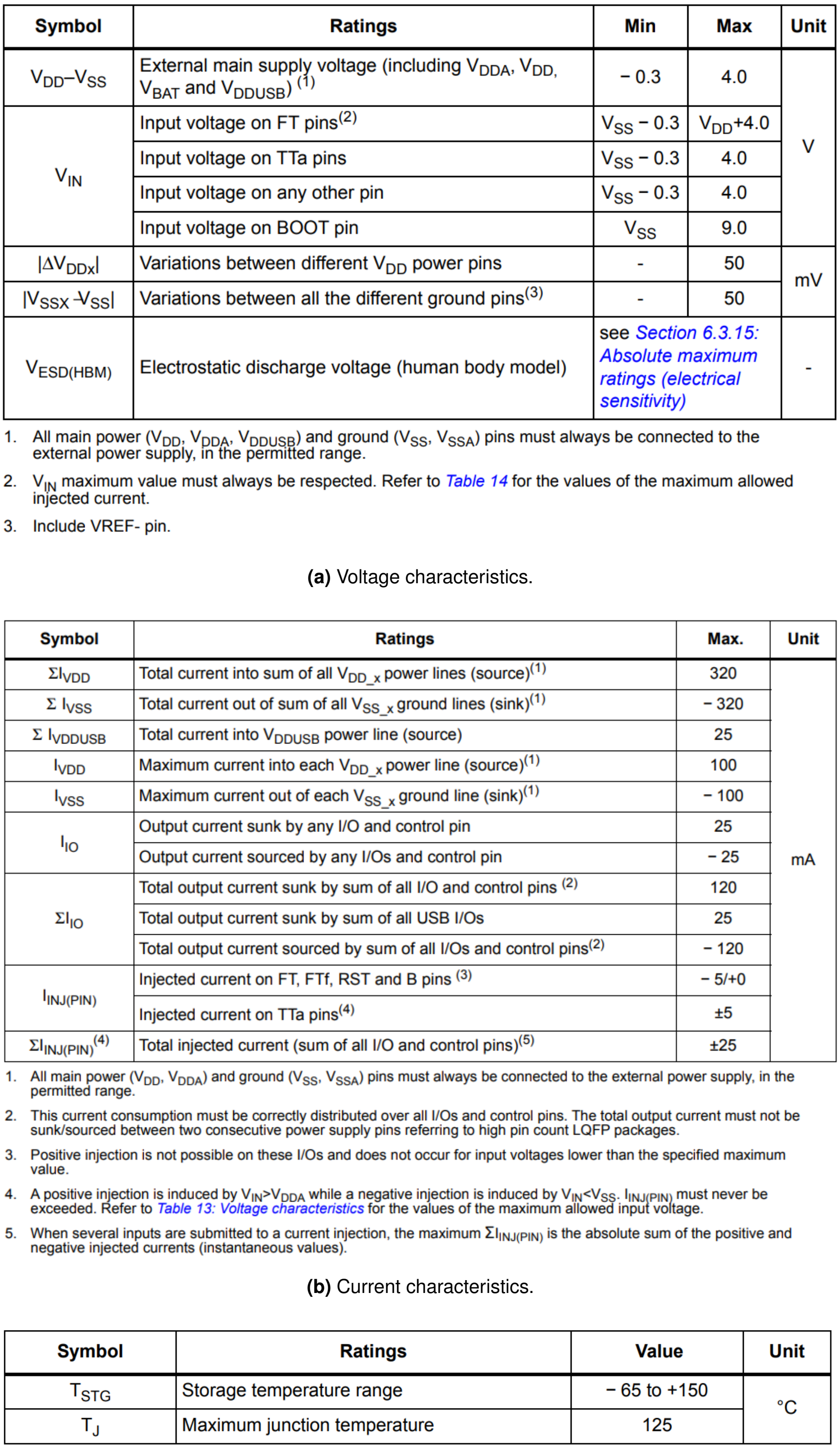


Figure 23: Image sensor schematic sheet from C2Sv0.1.

From the datasheet (section 6.2, pages 84-85), the absolute maximum ratings of the selected processor are:



Symbol	Ratings	Min	Max	Unit
V_{DD}–V_{SS}	External main supply voltage (including V_{DDA}, V_{DD}, V_{BAT} and V_{DDUSB}) (1)	− 0.3	4.0	V
V_{IN}	Input voltage on FT pins(2)	V_{SS} − 0.3	V_{DD}+4.0	
	Input voltage on TTa pins	V_{SS} − 0.3	4.0	
	Input voltage on any other pin	V_{SS} − 0.3	4.0	
	Input voltage on BOOT pin	V_{SS}	9.0	
$\|\Delta V_{DDx}\|$	Variations between different V_{DD} power pins	-	50	mV
$\|V_{SSX}$ -$V_{SS}\|$	Variations between all the different ground pins(3)	-	50	
$V_{ESD(HBM)}$	Electrostatic discharge voltage (human body model)	see *Section 6.3.15: Absolute maximum ratings (electrical sensitivity)*		-

1. All main power (V_{DD}, V_{DDA}, V_{DDUSB}) and ground (V_{SS}, V_{SSA}) pins must always be connected to the external power supply, in the permitted range.
2. V_{IN} maximum value must always be respected. Refer to *Table 14* for the values of the maximum allowed injected current.
3. Include VREF- pin.

(a) Voltage characteristics.

Symbol	Ratings	Max.	Unit
ΣI_{VDD}	Total current into sum of all V_{DD_x} power lines (source)(1)	320	mA
$\Sigma\ I_{VSS}$	Total current out of sum of all V_{SS_x} ground lines (sink)(1)	− 320	
$\Sigma\ I_{VDDUSB}$	Total current into V_{DDUSB} power line (source)	25	
I_{VDD}	Maximum current into each V_{DD_x} power line (source)(1)	100	
I_{VSS}	Maximum current out of each V_{SS_x} ground line (sink)(1)	− 100	
I_{IO}	Output current sunk by any I/O and control pin	25	
	Output current sourced by any I/Os and control pin	− 25	
ΣI_{IO}	Total output current sunk by sum of all I/O and control pins (2)	120	
	Total output current sunk by sum of all USB I/Os	25	
	Total output current sourced by sum of all I/Os and control pins(2)	− 120	
$I_{INJ(PIN)}$	Injected current on FT, FTf, RST and B pins (3)	− 5/+0	
	Injected current on TTa pins(4)	±5	
$\Sigma I_{INJ(PIN)}$(4)	Total injected current (sum of all I/O and control pins)(5)	±25	

1. All main power (V_{DD}, V_{DDA}) and ground (V_{SS}, V_{SSA}) pins must always be connected to the external power supply, in the permitted range.
2. This current consumption must be correctly distributed over all I/Os and control pins. The total output current must not be sunk/sourced between two consecutive power supply pins referring to high pin count LQFP packages.
3. Positive injection is not possible on these I/Os and does not occur for input voltages lower than the specified maximum value.
4. A positive injection is induced by V_{IN}>V_{DDA} while a negative injection is induced by V_{IN}<V_{SS}. $I_{INJ(PIN)}$ must never be exceeded. Refer to *Table 13: Voltage characteristics* for the values of the maximum allowed input voltage.
5. When several inputs are submitted to a current injection, the maximum $\Sigma I_{INJ(PIN)}$ is the absolute sum of the positive and negative injected currents (instantaneous values).

(b) Current characteristics.

Symbol	Ratings	Value	Unit
T_{STG}	Storage temperature range	− 65 to +150	°C
T_J	Maximum junction temperature	125	

(c) Thermal characteristics.

Figure 24: Absolute maximum ratings from STM32F750Z8T6 datasheet. Please note that these are snippets, so any hyperlinks or references are invalid.

4.1.4 Memory

Unlike the STM32N657I0H3Q, the STM32F750Z8T6 does not have a large enough internal memory to buffer an image. I selected 256Mb of memory compared to a value closer to the actual RGB image size (4MB) just so I did not have to worry about running out of memory. The increased memory size only result in a $2AUD increase in price, so the price change was justified by the increased safety threshold. Throughout testing, I expect to measure the consumed memory so the prototype uses only as much memory as it needs. Of the available options, the IS42S16160J-6TL was the cheapest.

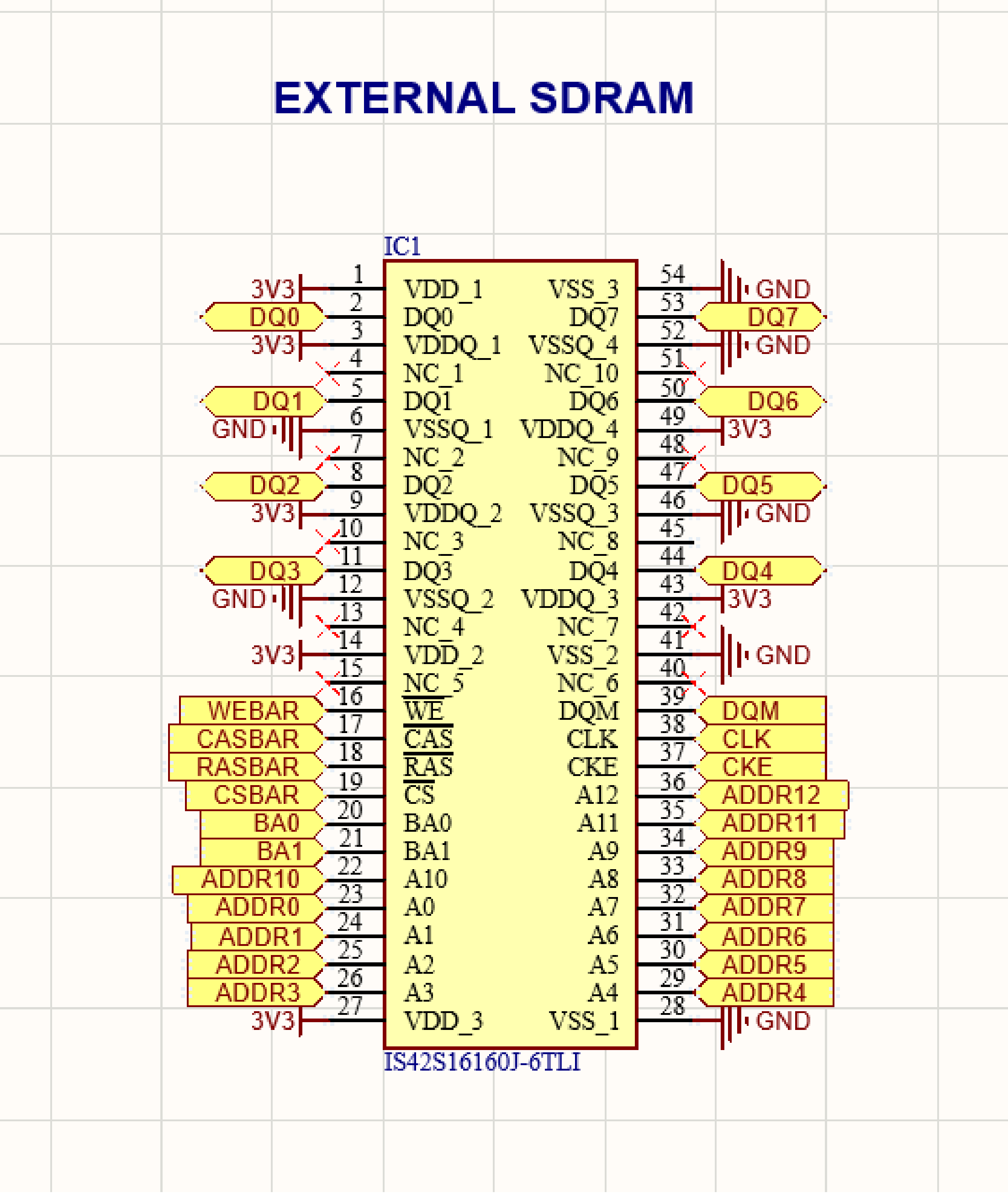
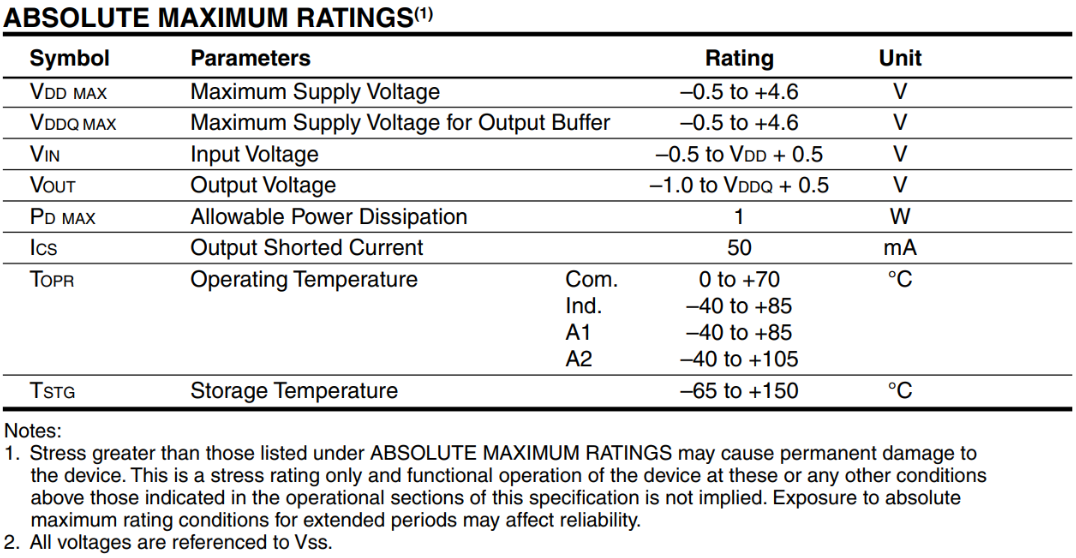


Figure 25: Image sensor schematic sheet from C2Sv0.1.

From the datasheet (page 16), the maxmium ratings for the component are:



ABSOLUTE MAXIMUM RATINGS[1]

Symbol	Parameters		Rating	Unit
VDD MAX	Maximum Supply Voltage		–0.5 to +4.6	V
VDDQ MAX	Maximum Supply Voltage for Output Buffer		–0.5 to +4.6	V
VIN	Input Voltage		–0.5 to VDD + 0.5	V
VOUT	Output Voltage		–1.0 to VDDQ + 0.5	V
PD MAX	Allowable Power Dissipation		1	W
ICS	Output Shorted Current		50	mA
TOPR	Operating Temperature	Com. Ind. A1 A2	0 to +70 –40 to +85 –40 to +85 –40 to +105	°C
TSTG	Storage Temperature		–65 to +150	°C

Notes:
1. Stress greater than those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect reliability.
2. All voltages are referenced to Vss.

Figure 26: Absolute maximum ratings for IS42S16160J-6TL.

4.1.5 USB-C

The USB-C schematic was mostly copied over from v0. I selected the USB4105-GF-A-060 component over others simply because that was the connector used on FISH and ADCS Integration board. I was able to remove the power sequencer (since the F7 does not require a specific startup sequence) and replace the buck converter with an LDO (because of the lower max power consumption).

Using the logic outlined in subsection A.3, I can estimate the power consumed by the circuit by placing a low-value resistor before the LDO and measuring the voltage drop across them from V1 to V2. Any voltage losses created by the resistor will be resolved by the LDO. Although this will not provide a highly accurate value, it will infer what the maximum current carrying capacitor of traces are for the prototype. Additionally, it will provide the Systems Engineering Lead with vital power requirements for the CubeSat. In-case this configuration is not neened, a jumper can be placed in parallel with the resistor.

In addition to the power estimation circuit, I have included the differential data lines to the connector. If configured correctly, I should be able to access the STM via a COM port for basic programming purposes instead of requiring a TAG connector all the time.

The component appears to have no absolute maximum characteristics listed. Since it's a connector and not an integrated circuit, I have assumed that it should not be a concern for thermal and vacuum testing. I will need to consider the maximum ratings of the connected battery/powerbank for stress testing however.

For more information about the pi-filter network, please see subsubsection 3.2.8.

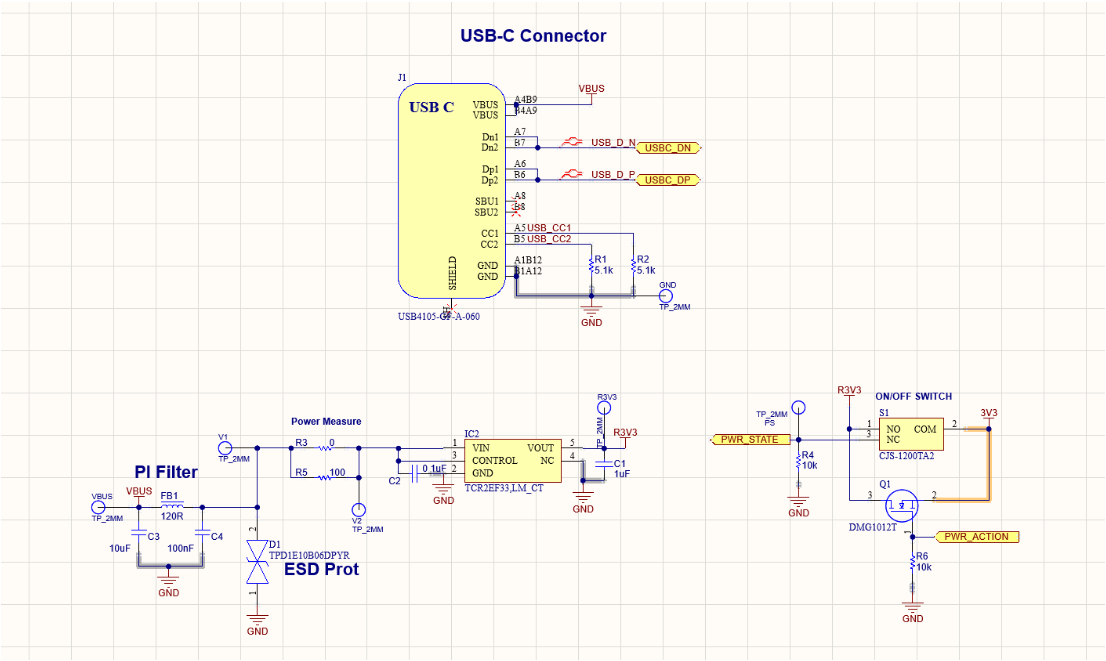
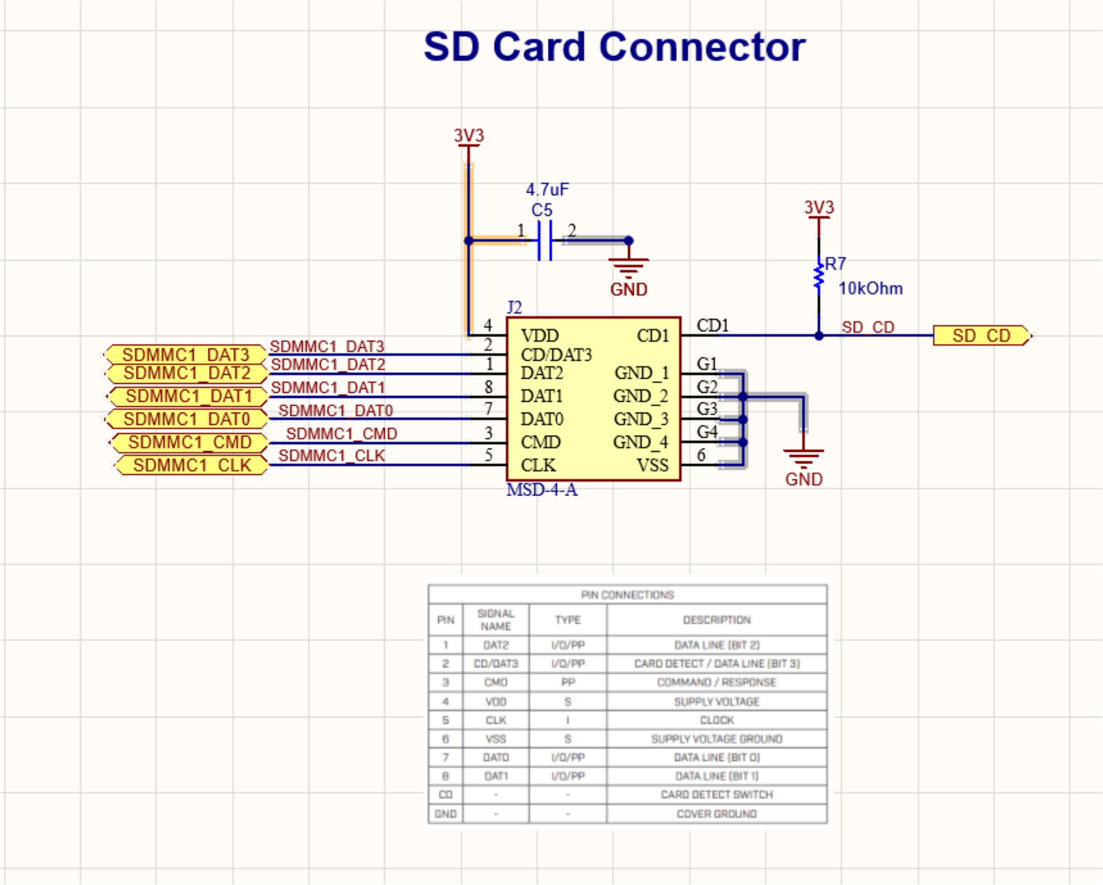


Figure 27: Image sensor schematic sheet from C2Sv0.1.

4.1.6 uSD

The uSD schematic was copied from v0. Unfortunately, the filter used in the original schematic sold out across most stores three days before v0.1 started, so, the component was removed from this schematic. Since the filter did act as EMI protection, I will make sure not to prematurely remove the uSD whilst the circuit is active just to mitigate any potential risks.

The component appears to have no absolute maximum characteristics listed. Since it's a connector and not an integrated circuit, I have assumed that it should not be a concern for thermal and vacuum testing. I will need to consider the maximum ratings for the actual uSD card used however.



PIN CONNECTIONS			
PIN	SIGNAL NAME	TYPE	DESCRIPTION
1	DAT2	I/O/PP	DATA LINE (BIT 2)
2	CD/DAT3	I/O/PP	CARD DETECT / DATA LINE (BIT 3)
3	CMD	PP	COMMAND / RESPONSE
4	VDD	S	SUPPLY VOLTAGE
5	CLK	I	CLOCK
6	VSS	S	SUPPLY VOLTAGE GROUND
7	DAT0	I/O/PP	DATA LINE (BIT 0)
8	DAT1	I/O/PP	DATA LINE (BIT 1)
CD	-	-	CARD DETECT SWITCH
GND	-	-	COVER GROUND

Figure 28: Image sensor schematic sheet from C2Sv0.1.

4.1.7 LPS

According to the datasheets of the F7 and AR0141, the only power domains required are: 3V3, 2V8 and 1V8. The F7 only requires 3V3 to operate, so the latter were only required for the image sensor. According to Table 27 of the AR0141 datasheet (page 54), the maximum operating current consumption in parallel output is 160mA. Because of this, I was able to re-use the same LDO package from v0 since their maximum current output is 200mA. As per the datasheet, I included decoupling capacitors on VCC and VOUT.

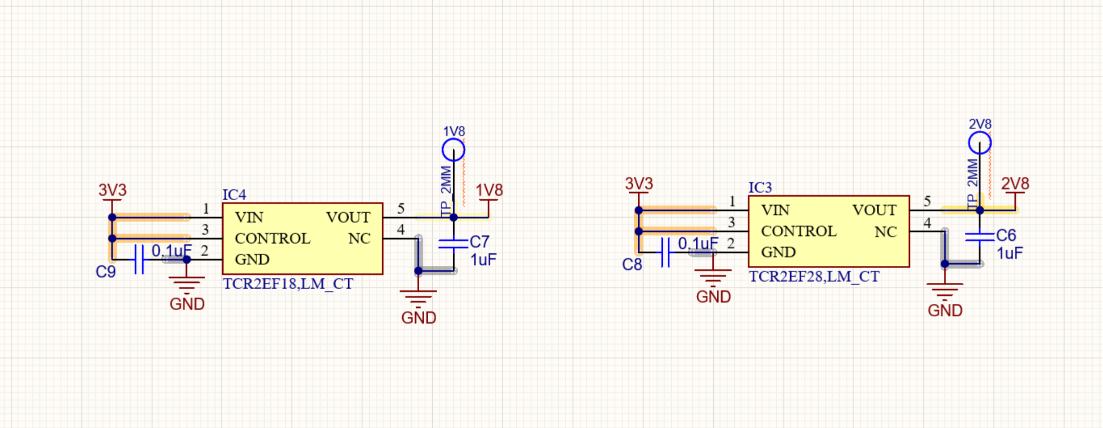
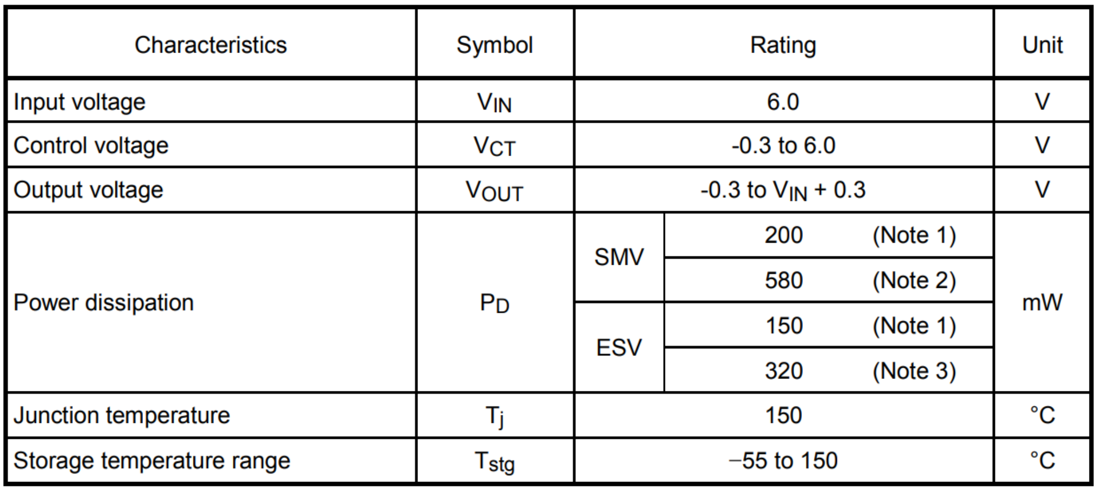


Figure 29: Image sensor schematic sheet from C2Sv0.1.

The absolute maximum ratings from the datasheet (page 2) are:



Characteristics	Symbol	Rating			Unit
Input voltage	V_{IN}	6.0			V
Control voltage	V_{CT}	-0.3 to 6.0			V
Output voltage	V_{OUT}	-0.3 to V_{IN} + 0.3			V
Power dissipation	P_D	SMV	200	(Note 1)	mW
			580	(Note 2)	
		ESV	150	(Note 1)	
			320	(Note 3)	
Junction temperature	T_j	150			°C
Storage temperature range	T_{stg}	−55 to 150			°C

Figure 30: Absolute maximum ratings of TCR2EF series (SMV).

4.1.8 TAG

The TAG connector schematic was copied from v0 (subsubsection 3.2.7). No changes were made. Robert Howie had advised me that Jacob, electrical engineer from Binar, has had issues in the past with TAG connectors, so the COM port should act as a failsafe in-case the TAG connector becomes unreliable after a while.

Since the component is just a series of pads and the connector will not be attached to the board during any tests, the maximum characteristics are not a concern (even though they aren't listed).

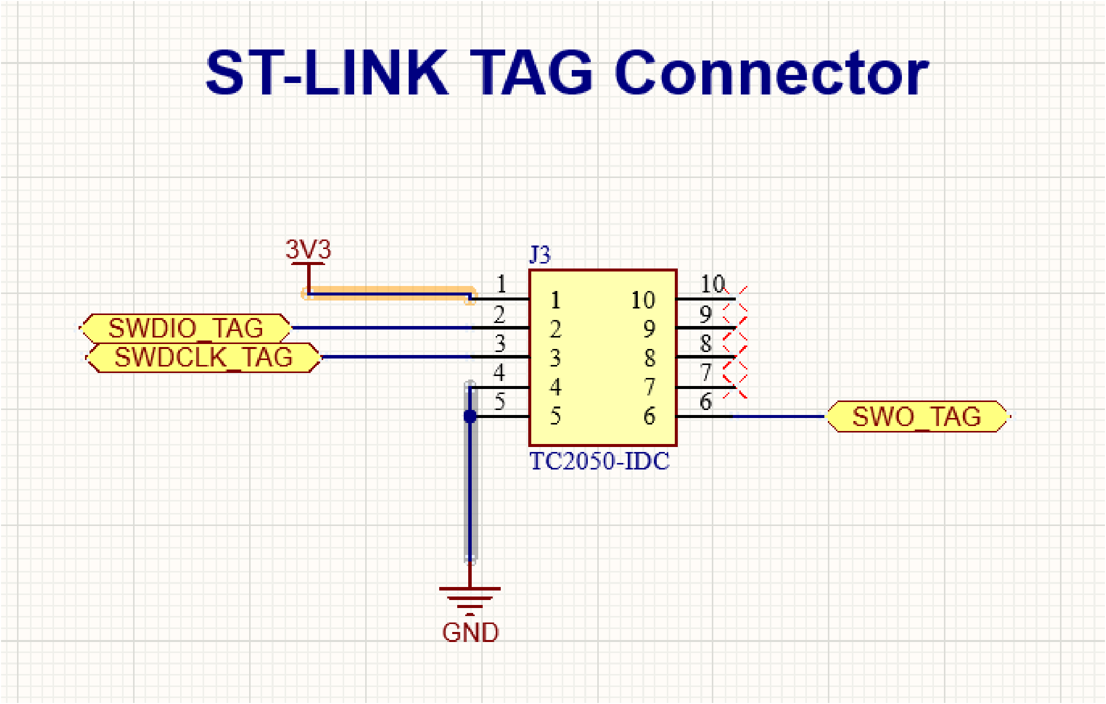


Figure 31: Image sensor schematic sheet from C2Sv0.1.

4.2 PCB Layout and Review

4.2.1 Overview

Both the processor and sensor PCBs use the 4-layer, 1oz, 1.6mm stackup and rules from JLCPCB.

4.2.2 Component Placement

I based the processor board around the STM32F7. After my mistakes from v0, I learnt that it's more important to place the most dense components (trace wise) in the centre, rather than the image sensor. The sensor board barely had any components, so I just placed the image sensor in the centre for simplicity and kept the PLL clock close to the IC.

I kept the USB-C, FPC and uSD connectors each on their own side of the board to avoid uneccesary trace overlap, and placed the SDRAM slightly far away from the F7 to give me enough space to route to each pin without being farther away than needed.

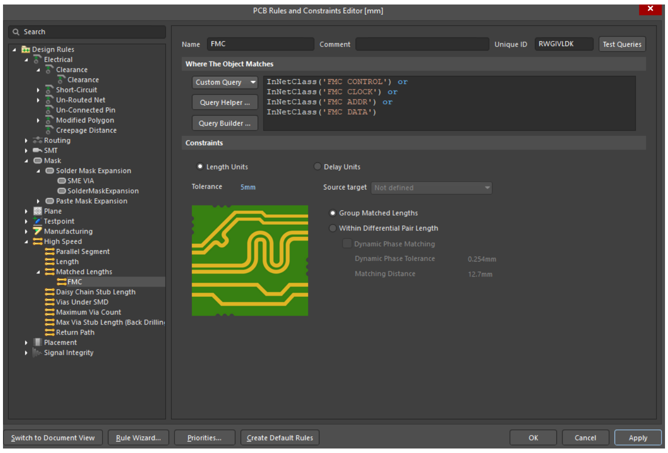
I kept each 'room' (silkscreen box) near each other and avoided spreading components out. This helped me keep track of each schematic sheet and should speed up the pick-and-place process since each set of components (LDOs, buttons, etc.) belong to one area of the PCB.

Lastly, I tried keeping all the decoupling capacitors as close to their ICs as possible. Some capacitors are a little further away than other since there just wasn't space directly next to the pin, but they're all within ¡5mm of where they need to be, so I doubt there will be any issues during testing.

4.2.3 Routing

- **Critical signal routing:**
  - Route high-speed signals (SDRAM D0-Dn, DQS, CK/CK, DCMI data/clock) first.
  - Keep traces short and direct, minimise vias.
  - Avoid right-angle bends; use 45° or curved traces.
- **Length matching:**
  - Match data bus traces and DQS within recommended skew.
  - Match address and control lines as needed.
- **Power and ground:**
  - Route power traces or planes with sufficient width for current.
  - Ensure continuous ground return paths under high-speed signals.
- **Via usage:**
  - Minimise vias in high-speed signal paths to reduce inductance.
  - Place vias carefully for layer changes of data/clock lines.
- **Routing strategy for buses:**
  - Keep parallel buses close together with consistent spacing.
  - Avoid crossing noisy signals.
  - Separate analog and digital traces if applicable.

To ensure that the FMC lines were correctly length matched, I created a rule to ensure that all FMC lines were within 5mm of each other.



Name FMC Comment Unique ID RWGIVLDK

Custom Query

```
InNetClass('FMC CONTROL') or
InNetClass('FMC CLOCK') or
InNetClass('FMC ADDR') or
InNetClass('FMC DATA')
```

● Length Units ○ Delay Units

Tolerance 5mm Source target Not defined

● Group Matched Lengths

○ Within Differential Pair Length

□ Dynamic Phase Matching

Dynamic Phase Tolerance 0.254mm

Matching Distance 12.7mm

Figure 32: FMC length match design rule.

4.2.4 Power Distribution

I implemented a 3V3 power plane on the second layer and a ground plane on the third layer. Other power lines were routed with thick 1mm traces to handle higher current.

On the sensor PCB, where 3V3 was not used, I created 2.8V and 1.8V power pours. A large DGND plane and a smaller AGND plane were included, connected using a point-to-point star connection to minimise noise coupling between digital and analog grounds.

4.2.5 Signal Integrity

- **High-speed signals:** Identify critical signals such as SDRAM data lines (D0-Dn), DQS, clocks, address/control lines, and DCMI parallel data bus, HSYNC/VSYNC, PCLK.
- **Trace design:**
  - Controlled impedance for high-speed traces.
  - Short, direct routing to minimise propagation delay.
  - Length matching for data, DQS, and address/control buses.
- **Parallel and differential signals:**
  - Differential pairs routed with matched lengths (if applicable).
  - Parallel buses length-matched and routed with consistent spacing.
- **Grounding and return paths:**
  - Solid ground planes for low-inductance return paths.
  - Separation of digital and analog grounds, if used.
  - Avoid splitting critical return paths under high-speed traces.
- **Decoupling and filtering:**
  - Proper placement of decoupling capacitors near ICs.
  - Use of series resistors or RC filters to reduce ringing.
- **Crosstalk and noise mitigation:**
  - Maintain spacing between high-speed and noisy signals.
  - Route sensitive analog traces away from digital buses.
- **Verification:**
  - Design Rule Checks (DRC) for trace spacing and impedance.
  - Optional signal integrity or timing simulations.
  - Adjustments such as serpentine routing for length matching.

I kept a minimum of 12mil between parallel traces, greater than the 3x spacing rule.

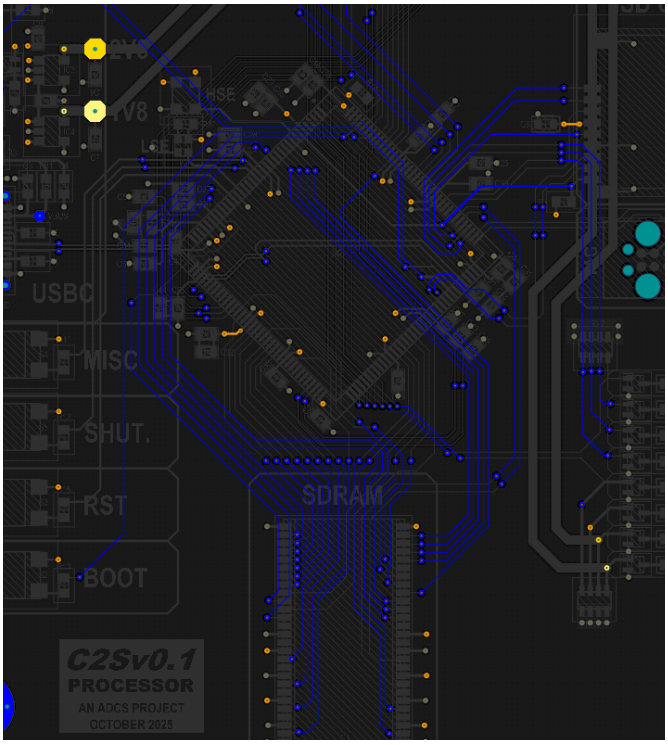


Figure 33: Parallel routes from C2Sv0.1 PCB.

4.2.6 Review and Verification

- **Design Rule Check (DRC):**
  - Check trace width and spacing.
  - Verify clearance to planes and other nets.
  - Confirm via sizes and pad clearances.
- **Electrical Rule Check (ERC):**
  - Check power/ground connectivity.
  - Verify decoupling and bypass capacitor placement.
  - Confirm net connections match schematic.
- **Critical signal verification:**
  - Confirm length matching for data, address, and control buses.
  - Verify controlled-impedance routing for clocks and high-speed signals.
  - Check for potential crosstalk or coupling issues.
- **Power integrity:**
  - Verify power planes and decoupling effectiveness.
  - Check for voltage drop or hot spots in power distribution.
- **Mechanical and connector checks:**
  - Confirm component placement for enclosure and mounting.
  - Verify correct pin assignments and orientation of connectors.
- **Documentation:**
  - Include PCB layer stackup.
  - Highlight critical traces, power planes, and decoupling in diagrams.
  - Note any compromises or deviations from ideal layout.

A Calculations

A.1 C-L-C Values

Model the ferrite bead as a series inductance L at low-mid frequencies. The two capacitors C_1 (input side) and C_2 (output side) form a pi filter with that series inductance. The pi network has a resonance frequency f where the L and the effective capacitance interact:

$$C_{eq} = \frac{C_1 \times C_2}{C_1 + C_2} \tag{1}$$

The resonance / cutoff frequency of the L-C_{eq} combination is:

$$f = \frac{1}{2\pi\sqrt{L \times C_{eq}}} \tag{2}$$

At frequencies well above f_r the filter attenuates; below fr the caps and ferrite appear largely as bypasses so DC and low frequency pass through.

For ferrite beads, its resonant frequency is typically 100MHz (unless stated otherwise). At frequencies well below resonance, the bead behaves approximately like an inductor:

$$Z \approx j2\pi f L \tag{3}$$

Rearranging:

$$L \approx \frac{Z}{2\pi f} \tag{4}$$

A.2 Transconductance of Oscillator

The gain margin ratio is determined by the formula $\texttt{gain}_{\texttt{margin}} = g_m/g_{\texttt{mcrit}}$, where g_m m is the oscillator transconductance specified in the STM32 datasheet. The HSE oscillator transconductance is in the range of a dozen of mA/V, while the LSE oscillator transconductance ranges from a few to a few dozens of μA/V, depending upon the product. $g_{\texttt{mcrit}}$ is defined as the minimal transconductance required to maintain a stable oscillation when it is a part of the oscillation loop for which this parameter is relevant. $g_{\texttt{mcrit}}$ is computed from oscillation loop passive components parameters.

Assuming $C_{L1} = C_{L2}$, and that the crystal sees the same C_L on its pads as the value given by the crystal manufacturer, gmcrit is expressed as follows:

$$g_{\texttt{mcrit}} = 4 \times \texttt{ESR} \times (2\pi F)^2 \times (C_0 + C_L)^2 \tag{5}$$

where:

- $\texttt{ESR}$ is the equivalent series resistance.
- C_0 is the crystal shunt capacitance.
- C_L is the crystal nominal load capacitance.
- F is the crystal nominal oscillation frequency.

If the oscillator transconductance parameter (g_m) is specified, make sure that the gain margin ratio ($\texttt{gain}_{\texttt{margin}}$) is bigger than 5.

A.3 Power Draw Estimation

If we have a series current-sense resistor of resistance R (Figure 34), the current flowing through the resistor is $I = V_d/R$, where V_d is the voltage drop across the resistor. Multiplying V_{IN} to both sides gives

$$V_{IN} I = \frac{V_d V_{IN}}{R} \tag{6}$$

Since $P = VI$, the power going into the circuit is $P_{IN} = \frac{V_d V_{IN}}{R}$.

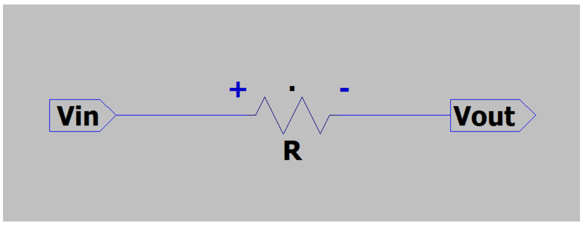


Figure 34: Resistor in series with a voltage source VIN with output voltage VOUT.