F25-13 Small Satellite Readout System

Final Project Binder

Cody Bullock
Won Jae Chung
Javier Torres Quinones
Yifu Yang

December 5th, 2025

Contents

Updated Final Problem Analysis

0.1 Objective

Our goal was to develop a lightweight and low-power imaging detector readout system for a CCD (Charge-Coupled Device) sensor for small orbital and suborbital satellites. The CCD captures a 1024×1024 image, with its pins controlled by an MCU. The MCU processes the image data with its internal 12-bit ADC and transmits the image via Ethernet to the connected host computer.

0.2 Requirements

Category	REQ-#	Requirement	Version #	Notes	Verification Type
Functional	FUN-1	Shall be operational between -55C and 35C	v1		Test
	FUN-2	Shall survive between -65C and 45C	v1		Test
	FUN-3	Shall generate image	v1.2		Demonstration
	FUN-4	Shall be clocked my a microprocessor	v1	Microprocessor will provide the clocking essential to operate the CCD Sensor and all other components	Demonstration
Performance	PER-1	Shall not generate more than 3 electrons of read noise	v1		Analysis
	PER-2	Should use a 16-bit ADC	v1	A 12-bit ADC is the minimum requirement, but a 16-bit ADC is preferred	Analysis
Compliance	COM-1	All code shall be written in Python	v1		Demonstration
Environmental	ENV-1	Should use radiation-protected hardware	v1		Test
Reliability	REL-1	Shall stay operational in harsh conditions(temperatures and radiation)	v1		Test
Power	POW-2	Shall be powered by a 28V wired connection	v1		Demonstration
	POW-2	Shall not consume more than 5W	v1		Demonstration
	POW-3	Shall have DC voltage regulation	v1.3	Voltage biasing for 28V, 25V, 17V, 10V, 5V, 3.3V	Test
Mechanical	ME-1	Total system shall not be larger that 1/2U	v1		Inspection
	ME-2	Shall not be made of more that 3 boards	v1	Either 8-32 or 10-32 mounting holes	Inspection
	ME-3	Board must have mounting holes	v1.2		Inspection
	ME-4	Shall use the CCD sensor provided	v1		Inspection

Input/Output	I/O-1	Shall use either Ethernet or SpaceWire for data transfer	v1		Demonstration
	I/O-2	Shall have pre-amping for the digital signal produced from the CCD sensor	v1		Demonstration
	I/O-3	Shall develop a communication protocol	v1		Demonstration
Testing	TEST-1	Shall test the functionality of the prototype in a thermal vacuum chamber	v1		Test
Schedule	SCH-1	Shall finish prototype design by April 20th			Demonstration

0.3 Constraints

0.3.1 External Factors

The primary constraint of this project is ensuring reliable operation in a low-Earth orbit environment. The system must function in a vacuum while being exposed to radiation and extreme temperatures. Although no explicit radiation tolerance requirement was provided, the system must remain fully operational between $-55°C$ and $35°C$, and must survive non-operational conditions ranging from $-65°C$ to $45°C$. Failure to meet these environmental constraints could result in degraded performance or complete system failure once deployed in space.

Power consumption is another key requirement. The system must not exceed a total power draw of 5 W. The satellite provides a regulated 28 V DC supply, which will be stepped down to 12 V for the CCD sensor and 3.3 V for the MCU and Ethernet interfaces.

Mechanical and architectural requirements are less restrictive. The system must fit within a height of 1/2U (0.875 inches). Since the design uses PCBs, meeting this height limit is manageable as long as individual components do not exceed the allowable vertical clearance. There are no strict length or width limits, permitting horizontal expansion of the layout. Additionally, the system must be clocked by an MCU, a requirement that is straightforward given the nature of the design. The datasheet provides recommended minimum clock speeds for each relevant pin, without imposing strict maximum limits.

0.3.2 Social Factors

The design of the CCD small satellite readout system is influenced by broader social and environmental considerations. Space conditions—including severe thermal fluctuations, ionizing radiation, and limited power availability—can degrade circuit performance, requiring the team to incorporate mitigation strategies.

Thermal management is critical, as the CCD must maintain stable operating temperatures to minimize dark current noise. The team plans to incorporate thermal control mechanisms such as radiative cooling or thermoelectric regulation to ensure consistent performance. Power efficiency is equally important due to the limited energy budget aboard satellites; therefore, low-power electronics and efficient voltage regulation are necessary. Electromagnetic interference poses another challenge, as it can compromise signal integrity. The customer requires less than three electrons of read noise (PER-1), prompting careful PCB layout techniques and strategic shielding to protect sensitive signals.

NASA engineering standards for space electronics inform component selection, reliability practices, and fault-tolerant design. The team will follow appropriate ESD handling guidelines during development to protect critical components. Furthermore, the prototype will undergo testing in a thermal vacuum chamber to simulate space-like conditions, validating the system's integrity and stability before deployment.

0.3.3 Ethical Factors

Relevant aspects of the IEEE Code of Ethics apply directly to the CCD readout system project. These include prioritizing public safety and welfare, ensuring accuracy and transparency in technical work, and taking responsibility for professional decisions.

Ethical considerations intersect with global cooperation, environmental sustainability, and public safety. High-quality satellite data supports climate research and disaster response efforts worldwide, emphasizing the importance of reliability and transparency. Clear communication of system limitations ensures proper interpretation of data and helps maintain public trust. The project's low-power design and emphasis on durability also reinforce ethical commitments to sustainability and minimizing space debris.

The engineering team has a responsibility to make well-informed and transparent design decisions. For example, the use of commercial or non–space-rated components must be accompanied by a clear explanation of risks, including potential failures under extreme radiation or thermal stress. Comprehensive testing—especially under simulated space conditions—is essential to verify safety and reliability. Thorough documentation ensures that customers and stakeholders understand system capabilities and limitations.

From this ethical analysis, several customer requirements and target specifications were defined: strict operational and survival temperature ranges (FUN-1, FUN-2), a low read-noise target (PER-1), total system power consumption under 5 W (POW-1), adherence to programming standards (SR-2), and the selection of space-rated or appropriately screened components (SR-3). These requirements reinforce ethical best practices and support long-term mission success.

0.4 Engineering Standards

The design of the CCD Small Satellite Readout System incorporates multiple engineering standards that ensure safety, reliability, interoperability, and long-term performance in space environments. These standards guide component selection, PCB layout practices, software development, and testing methodologies throughout the project.

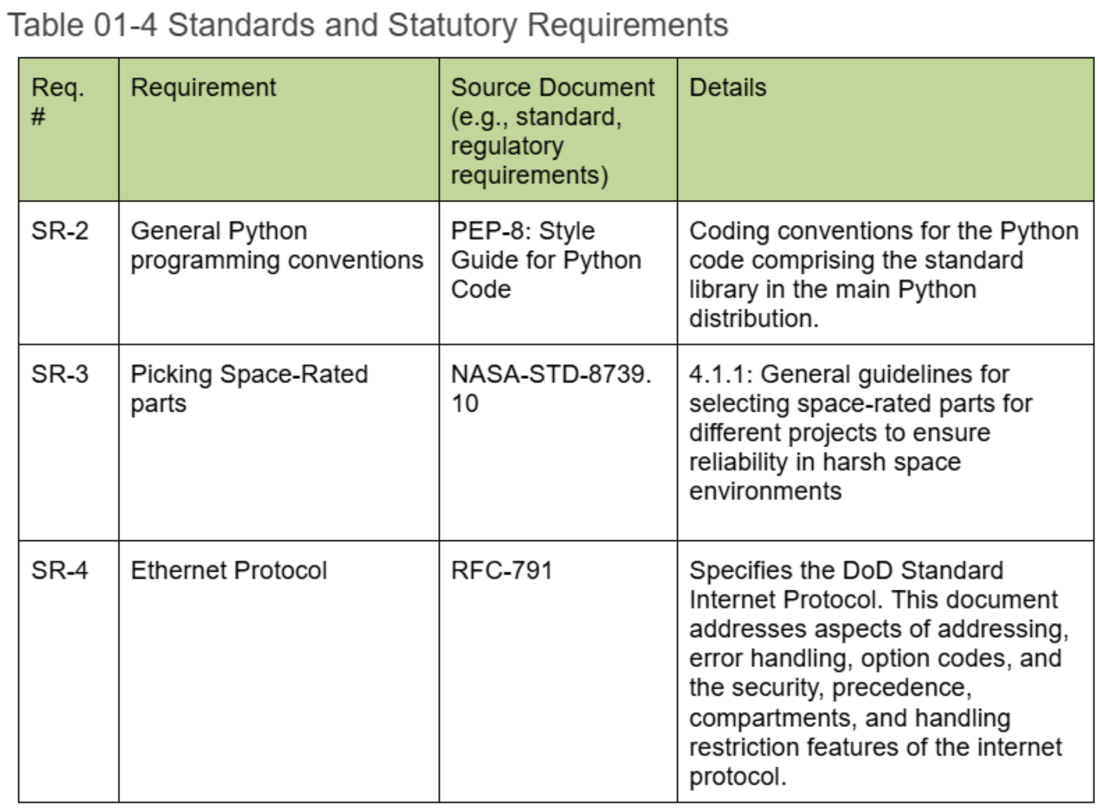

Table 01-4 Standards and Statutory Requirements

Req. #	Requirement	Source Document (e.g., standard, regulatory requirements)	Details
SR-2	General Python programming conventions	PEP-8: Style Guide for Python Code	Coding conventions for the Python code comprising the standard library in the main Python distribution.
SR-3	Picking Space-Rated parts	NASA-STD-8739.10	4.1.1: General guidelines for selecting space-rated parts for different projects to ensure reliability in harsh space environments
SR-4	Ethernet Protocol	RFC-791	Specifies the DoD Standard Internet Protocol. This document addresses aspects of addressing, error handling, option codes, and the security, precedence, compartments, and handling restriction features of the internet protocol.

Figure 1: Table Of Standards.

Updated Final Detailed Design

0.5 Detailed Design Documentation

0.5.1 Hardware

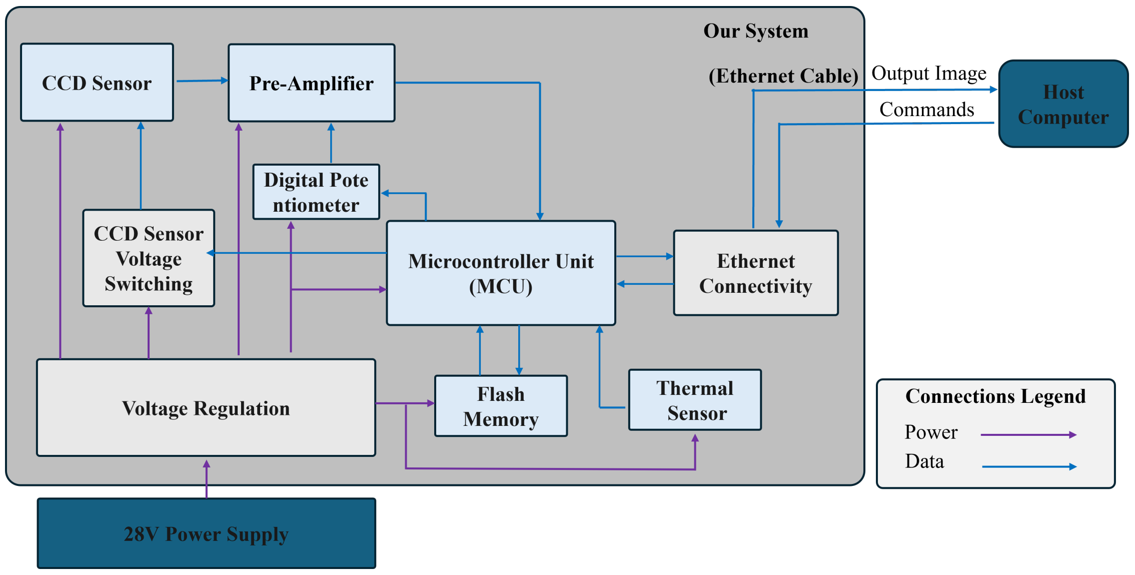

Figure 2: Overall Hardware Block Diagram.

Picture of the overall hardware. The specific implementations are shown in detail in the following subsections.

Microcontroller Overview

The microcontroller unit (MCU) of the circuit is the heart of the PCB. It handles all data communication through SPI and I2C connections, has analog-to-digital converters (ADC) for processing the CCD and temperature sensor outputs, and uses various GPIO pins for clocking pins on the CCD sensor. The chip used was from the STM32L073xx series of microcontrollers, which offer high-speed and decent performance at a low price and extremely low power consumption.

The average power draw for this chip is commonly in the order of microamps, only reaching milliamps when performing a significant number of tasks. Nearly all the communication channels on this 48-pin MCU were utilized for this prototype. Two SPI channels were used for flash memory and ethernet, the I2C channel was used for controlling the pre-amp's digital potentiometer, and two ADCs were used for reading a temperature voltage and the pre-amp's output. The MCU's internal timers would eventually be used for high precision timing of the CCD pins. STMicroelectronics provides a free-to-use IDE for programming their microcontrollers, which was used to program the MCU in C, and to configure the pin and timer setup.

Power Regulation

From the 28V primary supply, multiple regulated voltage levels are required to power the CCD sensor, microcontroller, and supporting circuitry. To generate these rails, a combination of buck converters and low-dropout regulators (LDOs) is used.

Buck converters are selected whenever the voltage drop exceeds 5V, as they provide significantly higher efficiency by minimizing power dissipation. LDOs are used for smaller voltage drops or where low noise is required.

The resulting voltage regulation chain is summarized below:

- 28V $\rightarrow$ 25V: LDO
- 25V $\rightarrow$ 17V: Buck Converter
- 17V $\rightarrow$ 10V: Buck Converter
- 10V $\rightarrow$ 5V: LDO
- 5V $\rightarrow$ 3.3V: LDO

The schematic diagrams for both the buck converter and the LDO implementation are shown in Fig. 3, Fig. 4, Fig. 5, Fig. 6, and Fig. 7, respectively.

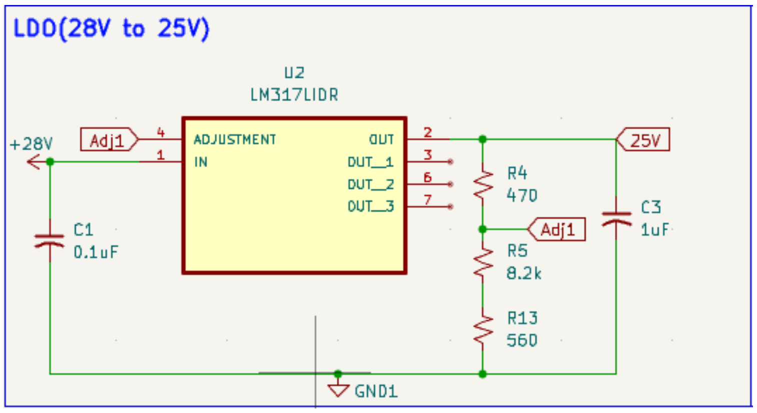

Figure 3: Schematic of the 28V to 25V LDO regulator stage.

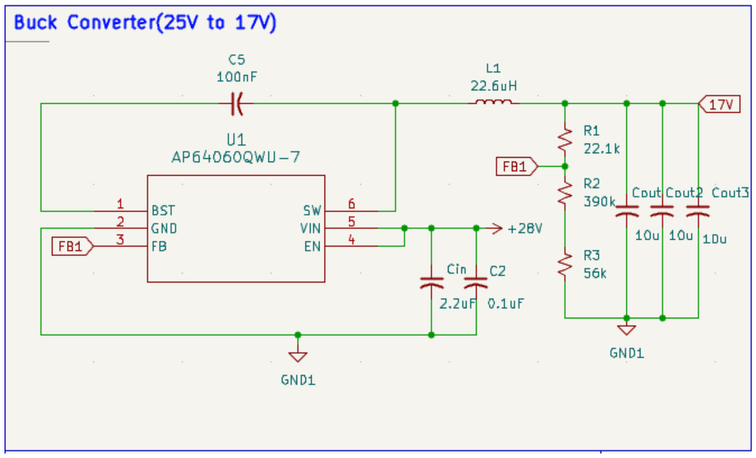

Figure 4: Schematic of the 25V to 17V Buck Converter stage.

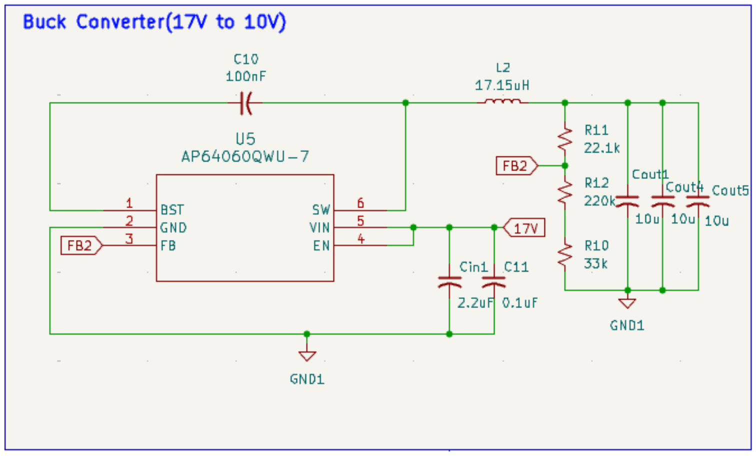

Figure 5: Schematic of the 17V to 10V Buck Converter stage.

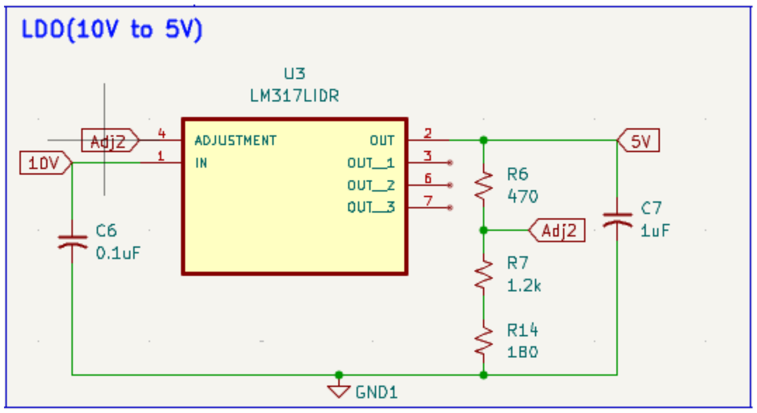

Figure 6: Schematic of the 10V to 5V LDO regulator stage.

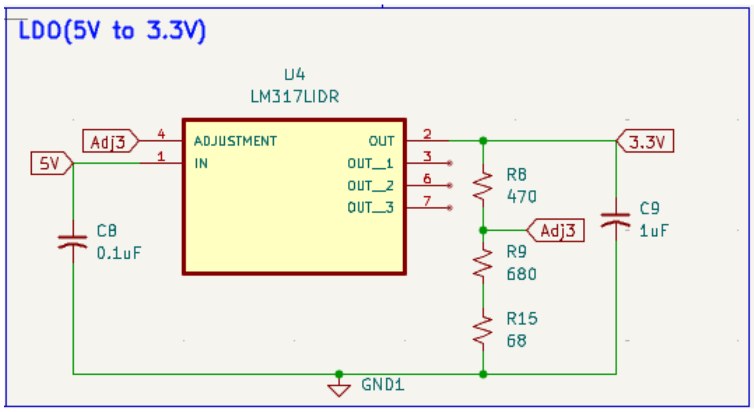

Figure 7: Schematic of the 5V to 3.3V LDO regulator stage.

MOSFET Switching

The CCD sensor requires various DC voltages for powering internal components, but the actual image capturing sequence is done by clocking individual input and output pins on the sensor. The clocking scheme, outlined in the CCD47-10 datasheet, is programmed and mainly controlled by the MCU, but the CCD requires that the clock pins be at a higher voltage than the MCU can provide. For a simple voltage switch, MOSFET inverters were used to allow the sensor to be controlled by high-speed signals between 10V-17V. The MOSFETs used were the EM6K7s because of their appropriate voltage range and high switching speed capabilities. For more stable control of the MOSFETs, gate drivers were used as a mediary between the MCU's GPIO pins and the gates of the transistors. The gate drivers also solve a small issue with the MOSFET inverter circuit, which by themselves, will invert the clocking signal coming from the MCU. The solution is to use the inverting input of the UCC27517 gate driver, as this will invert the signal coming from the MCU, which is then inverted again at the output of the MOSFET. In total, a GPIO pin from the MCU will power the inverting input of the gate driver, whose output will switch the gate on a MOSFET inverter. The output of this MOSFET inverter is the final CCD clocked signal, with a small shunt capacitor to filter peaks caused by high switching speeds.

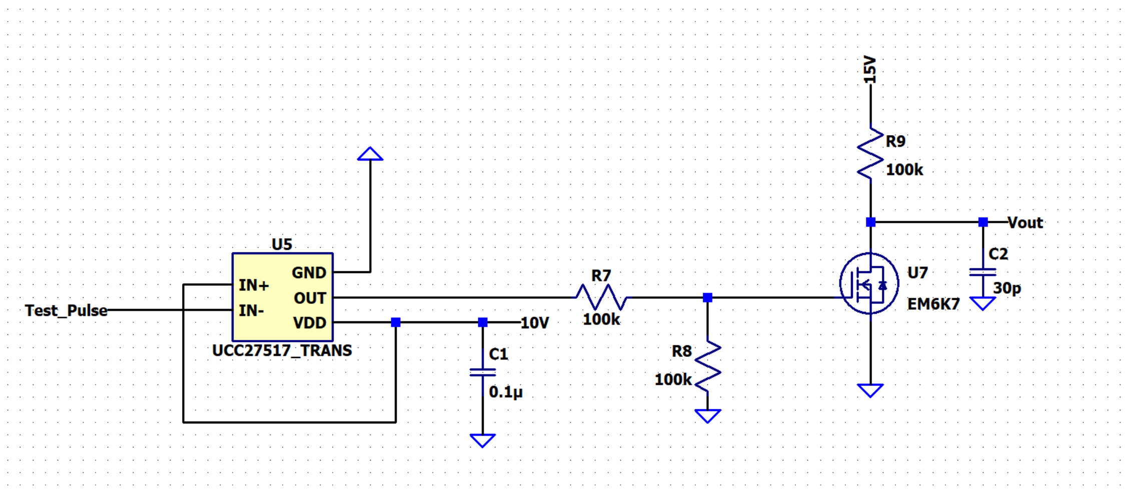

Figure 8: MOSFET Switch Diagram.

CCD Clocking Scheme

The CCD sensor converts incident light into charge, which is then represented as voltage levels on the output node. To sequentially shift out the pixel charges from the imaging array, specific clock signals must be driven in the correct timing sequence.

Row shifting is performed using the vertical transfer clocks, denoted as I_1, I_2, and I_3. These clocks advance the accumulated charges from one row to the next toward the serial register. Once a row is transferred, the column (serial) readout is controlled by the horizontal clocks R_1, R_2, and R_3. These signals shift charge packets across the serial register one pixel at a time toward the charge amplifier.

Together, the I_n (vertical) and R_n (horizontal) clocks define the full CCD readout sequence, enabling the system to extract pixel values in a structured and time-synchronized manner.

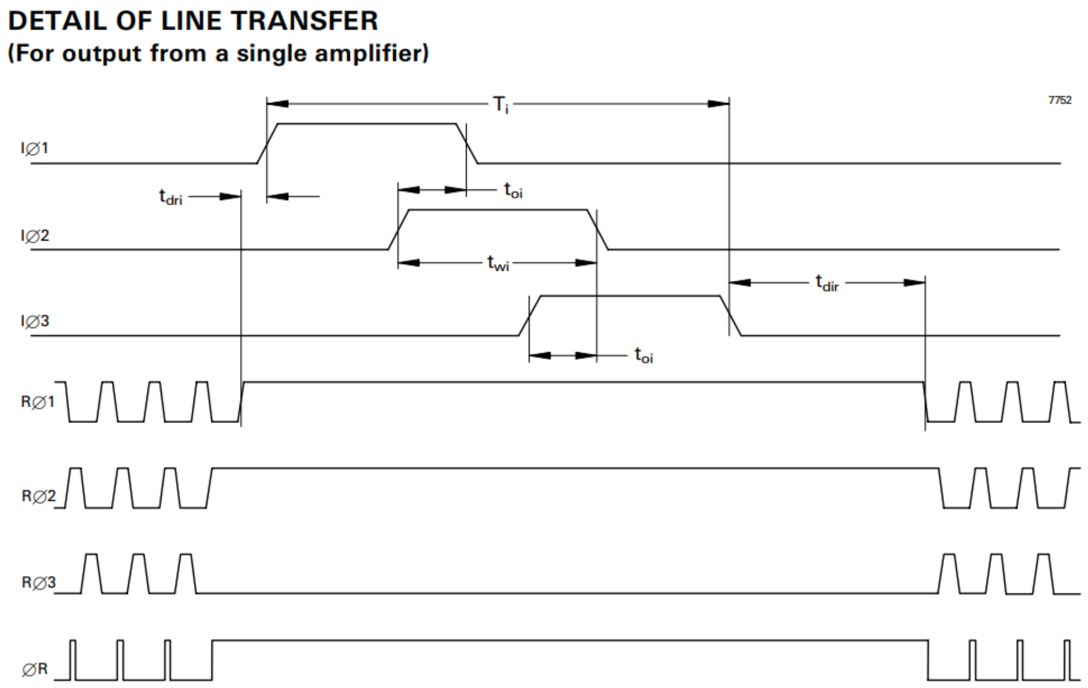

Figure 9: Time Scheme for Shifting a Row.

Table 1: Time Specification for Fig. 9

Pin	Time (s)
t_{dri}	$1\mu S$
t_{ri}, t_{fi}	$0.5\mu S$
T_i	$14\mu S$
t_{wi}	$5\mu S$
t_{oi}	$0.5\mu S$
t_{dir}	$2\mu S$

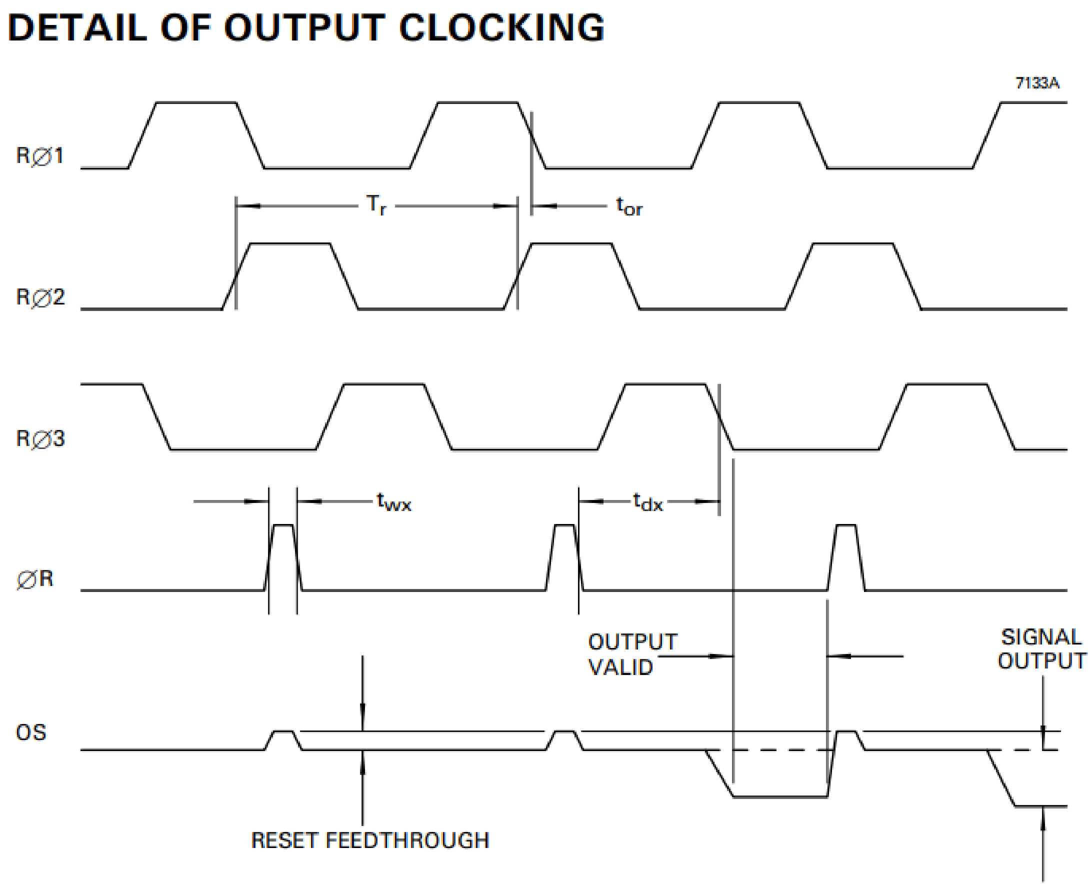

Figure 10: Time Scheme for Shifting a Column.

Table 2: Time Specification for Fig. 10

Pin	Time (s)
T_r	$1000ns$
t_{rr}, t_{fr}	$100ns$
t_{or}	$50ns$
t_{wx}	$100ns$
t_{dx}	$500ns$

Preamplifier with Adjustable Gain

The preamplifier is placed at the output of the CCD sensor to control the gain of its output. The output range of the CCD sensor isn't predefined, so the amplifier is there to account for a range of possible values. LT6200 op-amps were used for pre-amplification because they're rail-to-rail, and because they have an extremely low noise rating. The gain of the amplifier is adjustable through a digital potentiometer in the feedback loop, which is controlled by I2C connections that also go to the MCU. For testing purposes, the input to the preamplifier comes from a 2:1 multiplexor, the inputs of which are the CCD output, and an open pad on the surface of the PCB. This pad is intended to be connected to a waveform generator to test the functionality of the preamp before it's used by the CCD.

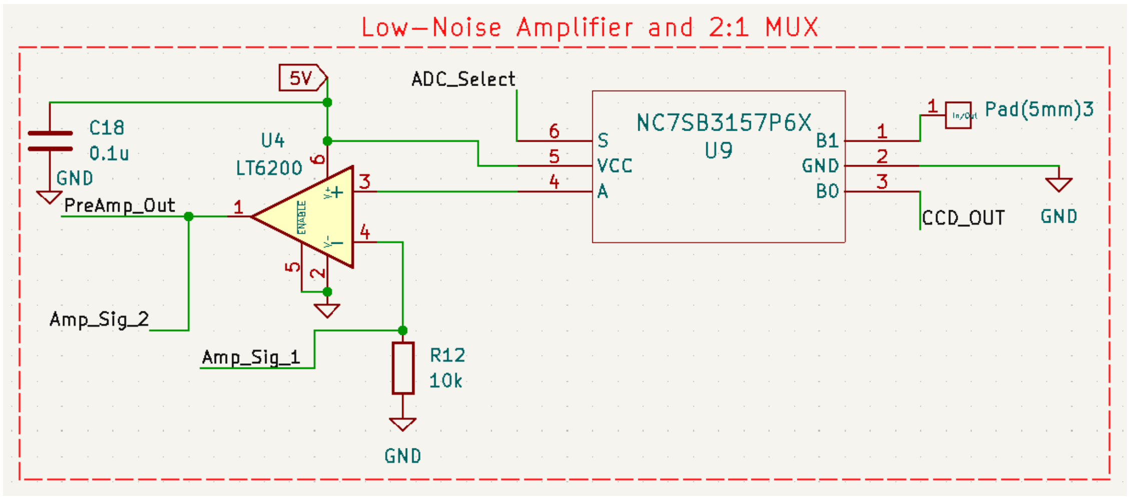

Figure 11: Low Noise Amplifier Diagram.

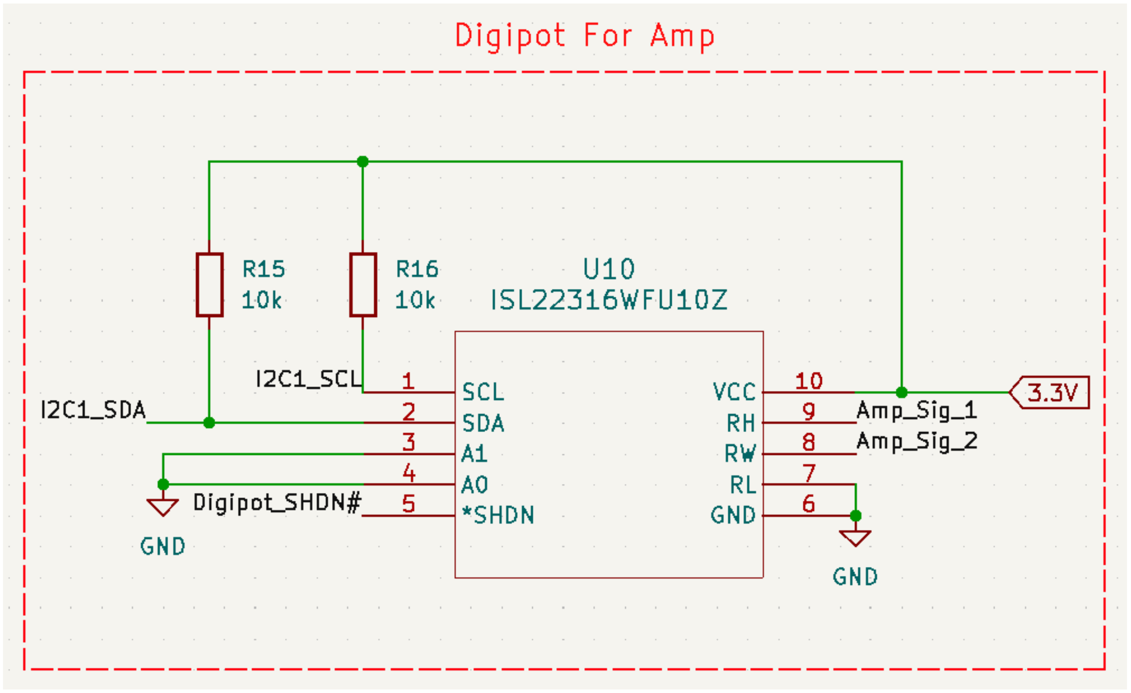

Figure 12: Digital Potentiometer Diagram.

Ethernet Connection

Originally, there was a section of the board dedicated to a fully integrated ethernet subsystem. Because of difficult implementation and time constraints, this ethernet system was purchased on a pre-made PCB that could connect through SPI to the custom board. In the current design, there are 10 pins near the bottom right of the PCB that can accommodate many different styles of this pre-built ethernet subsystem. The subsystem comes preprogrammed and in many cases is "plug and-play." On-board is an embedded ethernet controller, a 25MHz crystal oscillator, an RJ45 jack for the physical ethernet cable, and SPI pins that connect to an external MCU.

Flash Memory

The CCD47-10 has an image size of 1024x1024. With the 12-bit ADC inside the MCU, a full is about 1.5MB of data, which is significantly more than the MCU's 192KB of flash memory. A separate 128MBit flash memory chip was placed on the board to solve this problem. This flash memory chip communicates with the MCU through SPI connections and can be selected to be any size to allow for the storage of many images at once.

Temperature Sensing

The small temperature sending IC, the TMP20AIDRLR, was used for basic environment temperature sensing around the board [6]. This 6-pin IC outputs a voltage that can be converted to a temperature in Celsius using the following equation:

$$T = -1481.96 + \sqrt{(2.1962 * 10^6 + \frac{(1.8639 - V_o)}{(3.88 * 10^{-6})}}$$

Where T is the final temperature in Celsius, and Vo is the output voltage from the chip. The output voltage of this chip was connected to one of the ADCs on the MCU.

CCD Functionality

The CCD sensor itself is a small device that has a dedicated spot at the top right of the board. It has 24 pins, about half of which operate at DC voltages to power internal components like transistors. The other half drives the image capture sequence by receiving clocking signals from an outside source. In this case, it's the MCU and MOSFET switching circuit. The main idea is that the CCD sensor converts light to voltage at every pixel, then shifts one row of pixels into a readout register, then this register is shifted into the ADC of the MCU. Once every row has been shifted into the MCU, the image can be viewed by scaling the values of the array into a grayscale image.

0.5.2 Software

The software for the small satellite readout system is split into two main pieces:

- On-board firmware running on the STM32L073 microcontroller.
- A ground-side application running on a PC that sends commands and receives image data over Ethernet.

The MCU firmware is written in C using STM32CubeIDE and the HAL (Hardware Abstraction Layer) libraries. It configures the CCD control pins, ADCs, SPI buses, flash memory, temperature sensors, and the W5500 Ethernet controller. On top of these hardware drivers, we built a small command/response protocol so the PC can trigger image capture, read out status, and request stored image data.

Firmware Architecture

The firmware is organized into modules rather than one large `main.c` file. Each module is responsible for a specific subsystem:

- `main.c` - top-level initialization and main loop.
- `net_if.c` - network interface setup for the W5500 (IP address, socket creation, RX/TX buffering).
- `w5500_port.c` and `w5500_port.h` - low-level SPI read/write functions and chip select / reset control for the W5500.
- `system_controller.c` - system-level bring-up and basic network tests
- `CCD_Module.c` - CCD timing state machine and pixel readout control.
- `spimem.c` and `w25n01gvzeig.c` - external SPI flash driver and image storage.
- `Temperature.c` - ADC-based temperature measurement and conversion.
- `command_handler.c` - parses incoming Ethernet packets and dispatches them to the correct subsystem.
- `bsp_timer.c` - millisecond/microsecond timing utilities for CCD clocks and delays.

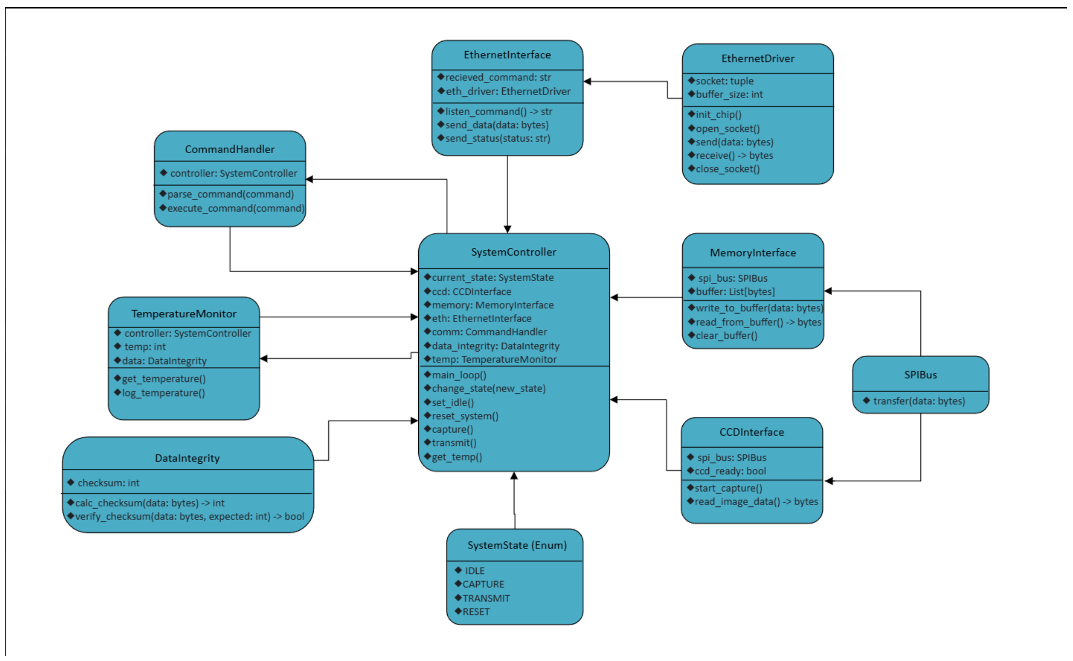

Figure 13: High-level firmware architecture showing the relationship between main, drivers, and application modules.

At startup, `main` performs the following steps:

1. Initializes the system clock and HAL.
2. Configures GPIO, ADC, SPI, and I2C peripherals.
3. Initializes the W5500 through `net_if.c` and `w5500_port.c` (sets MAC, IP, subnet, and opens a UDP/TCP socket).
4. Initializes the CCD timing module and external flash driver.
5. Enters the main loop, where it continuously checks for incoming Ethernet packets and runs periodic housekeeping tasks.

The main loop is intentionally simple: it calls a function to poll the network interface, processes any decoded command, and then returns to idle until another packet arrives or a timer expires.

Network Interface and Command Protocol

The W5500 Ethernet controller is connected to the MCU via SPI. All network traffic is handled by the ioLibrary driver from WIZnet, which provides the `socket()`, `sendto()`, and `recvfrom()` APIs. Our board is assigned a static IPv4 address, and the PC is configured with a matching address on the same subnet.

On top of the raw UDP/TCP socket, the firmware implements a small binary command protocol. Each packet from the PC contains:

- A one-byte **command ID**.
- Optional payload bytes (image index, time between photos).

The `command_handler.c` module inspects the command ID and calls the appropriate handler function. Typical commands include:

- **PING** – sanity check; the board responds with a short reply so we can confirm connectivity.
- **CAPTURE_IMAGE** – starts a full CCD capture sequence and stores the result in SPI flash.
- **READ_IMAGE_CHUNK** – returns a portion of the image so the PC can reconstruct the full frame.
- **READ_TEMPERATURE** – returns the latest temperature reading from the on-board sensors.
- **RESET_SYSTEM** – performs a soft reset of key subsystems (CCD module, flash, or network stack).

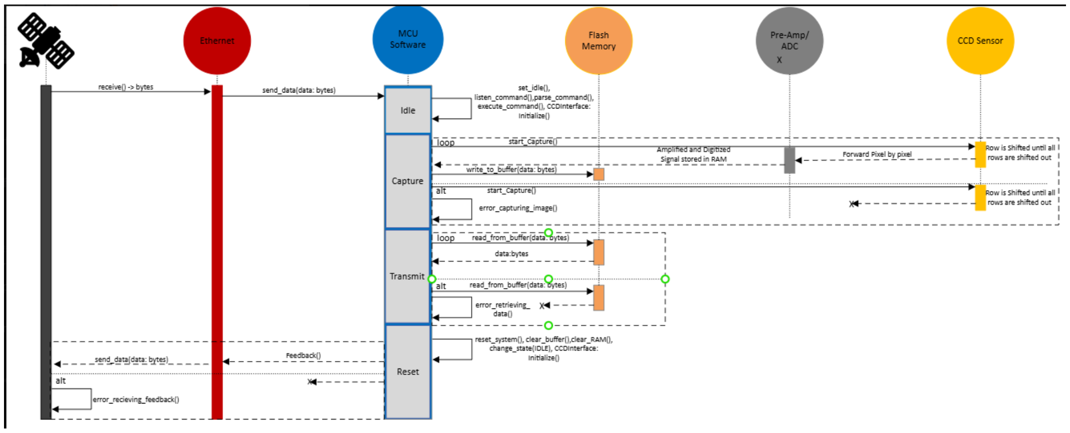
Ethernet
MCU Software
Flash Memory
Pre-Amp/ ADC
X
CCD Sensor
receive() -> bytes
send_data(data: bytes)
Idle
set_idle(), listen_command(),parse_command(), execute_command(), CCDInterface: Initialize()
Capture
loop
start_Capture()
Amplified and Digitized Signal stored in RAM
Forward Pixel by pixel
Row is Shifted until all rows are shifted out
write_to_buffer(data: bytes)
alt
start_Capture()
error_capturing_image()
Row is Shifted until all rows are shifted out
Transmit
loop
read_from_buffer(data: bytes)
data:bytes
alt
read_from_buffer(data: bytes)
error_retrieving_ data()
Reset
reset_system(), clear_buffer(),clear_RAM(), change_state(IDLE), CCDInterface: Initialize()
send_data(data: bytes)
Feedback()
alt
error_recieving_feedback()

Figure 14: Example command/response flow between the PC and the CCD readout board.

This command-based approach keeps the firmware flexible. New functionality can be added later by defining additional command IDs and attaching new handler functions without changing the overall structure.

CCD Readout and Image Storage

The `CCD_Module.c` file is responsible for translating high-level actions (like “capture one frame”) into the exact sequence of clock transitions the CCD needs. This module works closely with `bsp_timer.c` to generate microsecond-level delays between pulses.

A simplified capture sequence looks like this:

1. Assert the correct DC bias and clock rails using the MOSFET gate drivers.
2. For each row:
   (a) Pulse the vertical clocks I_1, I_2, and I_3 using GPIO.
   (b) Shift one row into the serial register.
   (c) For each column in the row, pulse the horizontal clocks R_1, R_2, and R_3, then sample the preamp output using the ADC.
3. Pack the 12-bit ADC readings into bytes and stream them into external SPI flash through `spimem.c`.

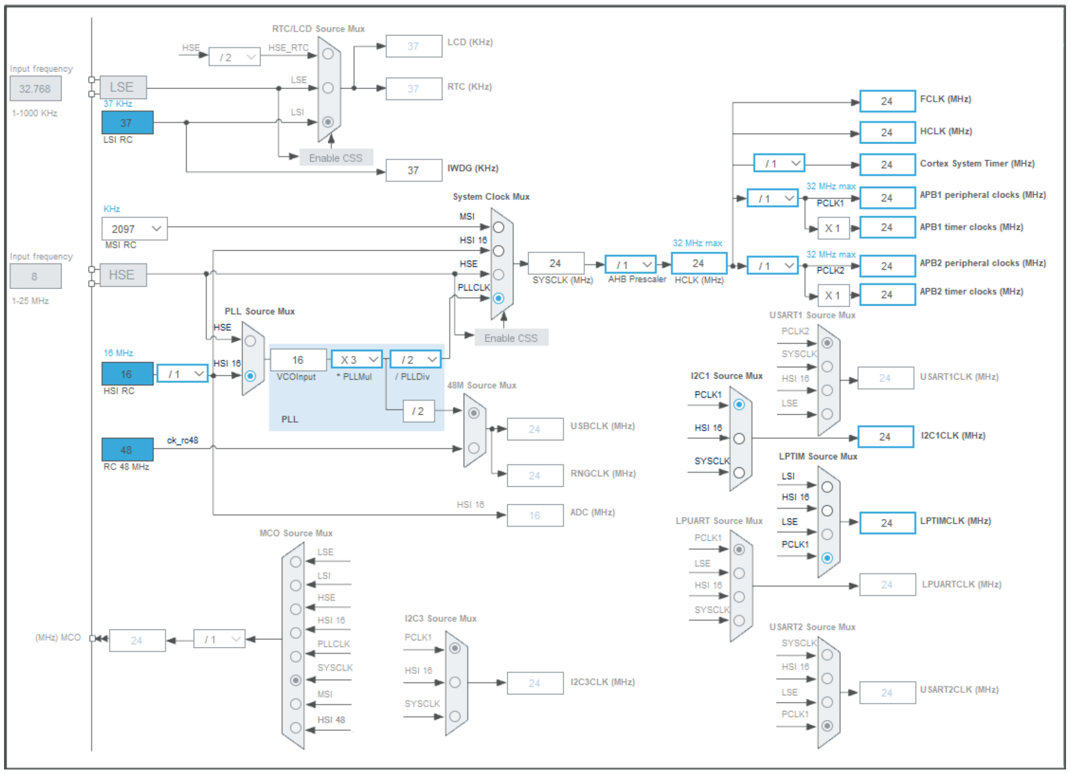

Figure 15: Clock Diagram implementation for CCD integration

Because a full 1024 × 1024 frame is larger than the MCU's internal flash, the image is split into logical blocks and written into the external W25N01GV SPI NAND. The `spimem.c` and `w25n01gvzeig.c` modules expose simple APIs such as:

- `nand_spi_init()` – initialize the flash device.
- `nand_write_page()` – write a page of pixel data.
- `nand_read_page()` – read a page back for transmission.

The command handler uses these functions to fetch specific portions of the stored image when the PC sends a **READ_IMAGE_CHUNK** command.

Temperature Monitoring and Housekeeping

The `Temperature.c` module reads the output of the TMP20AIDRLR temperature sensor using the MCU's ADC and converts the measured voltage into a temperature in degrees Celsius using the provided transfer function. This temperature can be requested by the PC as part of a housekeeping status frame or polled periodically.

Other small housekeeping tasks (such as updating simple status flags or watchdog refresh) are handled in the main loop and in periodic callbacks defined in `bsp_timer.c`.

PC-Side Software

On the host side, a simple Python application is used to control the board and visualize data. The program:

- Opens a UDP/TCP socket to the board's IP address and port.
- Sends user-selected commands (PING, CAPTURE_IMAGE, READ_IMAGE_CHUNK, etc.).
- Receives binary pixel data and reconstructs it into a 2D image array.
- Displays a grayscale preview of the image and logs basic status information.

In its current form, the GUI provides buttons for the main operations and a status window showing connection state and most recent temperature reading. This interface can be expanded later to support batch captures, automated test sequences, or exporting frames for external analysis.

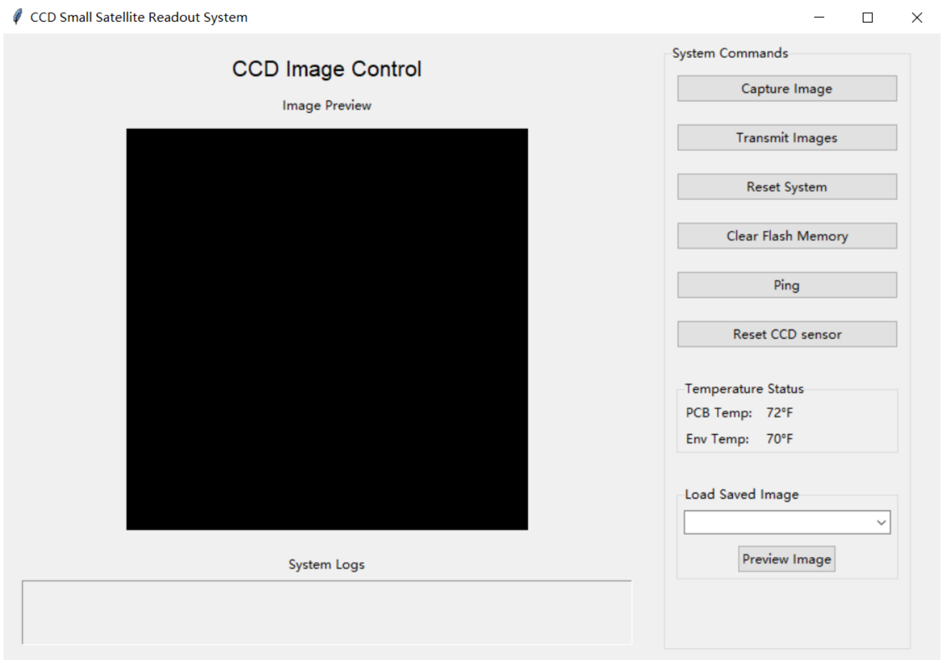

Figure 16: PC-side control GUI used to send commands and view captured images.

Overall, the software stack turns the CCD readout board into a simple networked instrument: the firmware abstracts away the low-level timing and SPI details, and the PC application talks to it using a small, well-defined command protocol over Ethernet.

0.6 Test Cases

0.6.1 Hardware Testing

Voltage Verification Testing

Once the board is connected to the 28 V power supply, the multimeter will be used to measure and verify the output voltage levels produced by the boost converter and linear dropout regulators. This is done to ensure proper bias voltages for use around the circuit, and also testing whether the circuit uses more than 5W.

Bench Waveform Testing

A voltage signal will be applied to the board using a function generator. The signal will pass through the 2:1 MUX, then the pre-amplifier, and finally into the ADC. The resulting digital output will be monitored and printed to the console for verification. This test is conducted to ensure that we can control the gain of the pre-amplifier effectively.

Switching Verification Testing

First, the development board will be flashed with the software responsible for generating the CCD sensor's timing outputs. Once programmed, an oscilloscope will be used to probe the CCD sensor's input pins. After running the software, the resulting timing waveforms will be observed and checked to ensure they match the planned timing scheme. This test is critical, as it tests each stage of the CCD pin switching configuration to ensure that we can properly clock its input pins.

Bench Temperature Monitoring

The MCU's ADC will be used to read the voltage outputs from the temperature sensor. These voltages will then be converted to temperatures in Celsius using the Steinhart–Hart equation. The calculated temperatures will be compared to the thermometer's measured room temperature for verification.

0.6.2 Software Testing

Data Storage Testing

Data storage testing involves sending data to and from the on-board flash memory chip. The test is very simple, we send some data to the memory chip and see if we can retrieve it later. This process will be integral to the image capture process, as images will all be stored in flash memory.

Data Sending and Receiving Testing

Data send and receive testing involves sending data through the Ethernet portion and out of the board onto the host computer, and sending data from the host computer onto the board. This test is done to ensure that we can both send commands to the PCB, and also receive data from it.

0.6.3 System Testing

Noise Level Testing

Noise level testing is done to see whether the board adds more that 3 electrons of read noise. To conduct noise level testing, with the CCD connected, two images are taken back to back very quickly. These images are stored and subtracted from each other. The resulting difference image is then multiplied by $\sqrt{2}$ to convert it to Digital Units (DNs). Once in DNs, the value can be converted to electrons using the sensor's gain.

Full Image Testing

Full image testing tests every subsystem except for temperature sensing. First, with the CCD attached, the "Capture Image" command is given to start the capture process. The command will prompt the MCU to start clocking the image, which will then start outputting voltages to the pre-amplifier, which will then move into the ADC of the MCU. Once the entire image has been shifted into the flash memory of the board, the "Receive Image" signal will be send to the board, which will begin to move the image from the flash memory to the Ethernet module. From the Ethernet module, the image will be sent over an Ethernet cable back to the host computer that send the commands to the PCB. The image will be displayed in the GUI to check its integrity and accuracy.

Build & Test

0.7 Project Status / Results

The layout of the PCB was done in sections that isolate each major component group. The power regulation section takes up the full left column, and the MOSFET switching is just to the right of it. The CCD sensor has a section dedicated to it at the top right of the board, and below it in the bottom right houses the rest of the components. The MCU sits in the middle for easy access by the rest of the components, and there are pin connections on the edges for programming and ethernet connectivity. The flash memory and temperature chips are set above the MCU, and the pre-amp section, which contains the amplifier, digital potentiometer, and 2x1 MUX, are set to the left of the MCU.

The PCB itself is six layers. The top and bottom layers are dedicated to high-speed signals, which is mostly the switching signals for the CCD. This, and ground pouring, was done to improve heat dissipation, and reduce noise in the final image. There are two internal layers that are ground planes, one power plane for 3.3V, and the final layer for all other general signals. All components were placed on the top layer of the board. Small shunt capacitors (usually 0.1uF or 10uF) are placed very close to the input power pins of each IC, which help with decoupling and current stability.

We ran out of testing time, so we're only fully sure that the voltage regulation and MOSFET switching on the board work. With further testing and programming we could have completed more, it was simply a time issue.

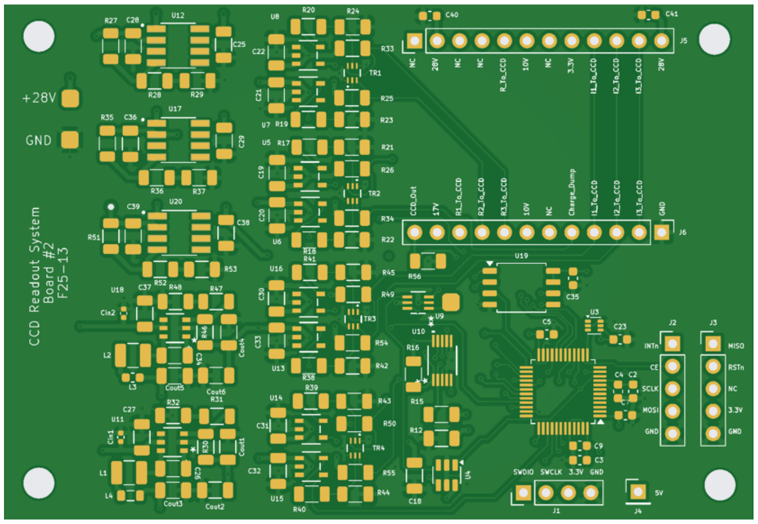

Figure 17: PCB Top Layer.

0.8 Test Results & Analysis

Due to tie constraints and issues with the code, we were only able to test a few of the hardware subsystems on our board.

0.8.1 Hardware Results

Voltage Verification Testing Results

The purpose of voltage verification was to ensure that the combination of buck converters and linear regulators on our board were outputting the proper voltages, and that the total power usage was under 5W. We plugged in a 28V power, supply, then used a multimeter to probe each power stage's output.

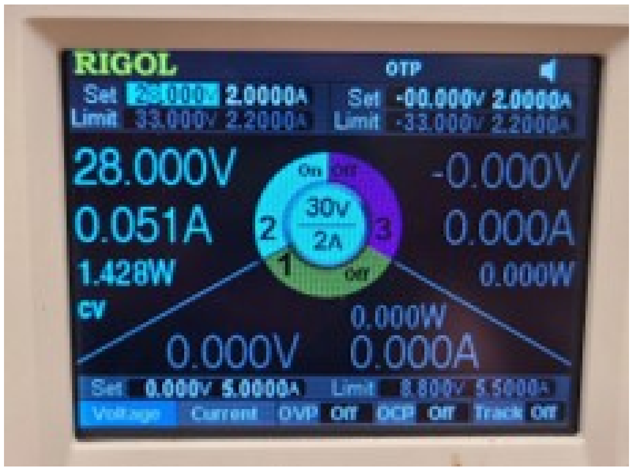

Figure 18: Input Voltage and Power Usage.

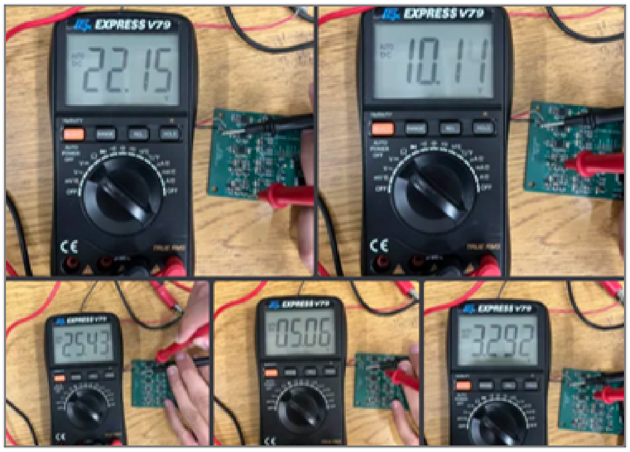

Figure 19: Output Voltage Levels.

We read correct voltage values for 25V, 10V, 5V, and 3.3V, but the 17V stage is outputting 21V. We're not sure what's causing the issue, but we suspect it's an issue with our inductors. In future designs, we'd like to experiment with larger inductor sizes.

Switching Verification Testing Results

Switching verification testing was conducted to make sure that the MOSFETs and gate drivers on the board were working correctly. We applied a 1kHz test pulse on one of the GPIO pins connected to a CCD clock pin, and the test pulse was compared to the output at the pin.

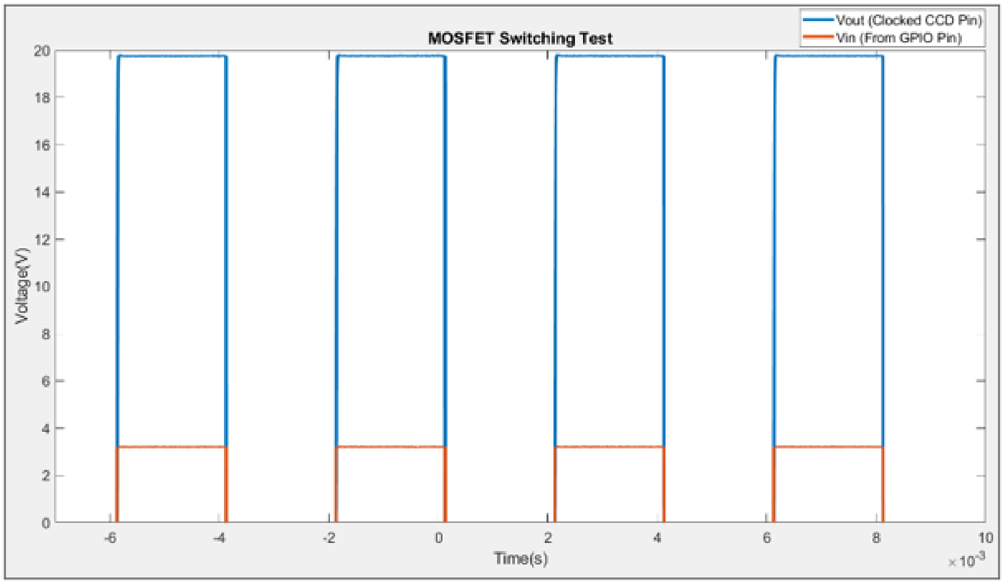

Figure 20: Output Voltage Levels.

The test was a success because the GPIO pin properly switched the MOSFET inverters.

0.9 Key Accomplishments

The team have completed one revision of our original design. The result is a low-power, 6-layer PCB capable of capturing and storing an image, with plans to finalize the functionality for sending that image over Ethernet.

Delivery

0.10 Overall Project Performance

While we weren't able to fully finish programming or testing our product, we made sufficient prototype that nearly accomplishes what we were reaching for. Our PCB has fully functioning voltage regulation, and the MOSFET switching works with only a few minor issues.

The first thing we'd like to test in the future is how our board works with an actual CCD sensor. We weren't able to test with one this semester, but our design should be fully functional assuming we did have a sensor. Other work would be thermal vacuum chamber testing, designing an on-board Ethernet circuit, and making the power design more efficient.

0.11 Customer Satisfaction

N/A

0.12 Deliverable Status

All parts for this project were purchased through the ECE Department, and all parts will be returned to the school on Monday, December 8th. All of the code is available in a public GitHub repository at https://github.com/Rheinjaeger/CCD.git

0.13 Challenges/Issues

Our first PCB did not work, which prevented us from testing any of our code and resulted in about eight capacitors being damaged. Because redesigning takes several weeks, we had to repeat the entire process: updating the schematics, completing a new PCB layout and routing, waiting for the new boards and components to arrive, and then performing all the soldering work. Since our ICs were only available in SMD packages, we were limited in how much breadboard testing we could perform. As a result, we were not able to test the system with the image sensor. We also transitioned from MicroPython-based firmware to a C-based implementation. Although we were able to establish an ICMP connection, we were still unable to send data through UDP.

0.14 Schedule and Cost Status

In our original schedule for the Fall 2025 semester, we planned to design a maximum of three boards. The general flow of a three week period was to design and order a board, test it, find the issues, then design again, and repeat. We went through a lot of intermediary designs from week to week, but we only ended up fully designing and ordering two boards. As for our ordering schedule, we were on track for most of the semester, but we ran out of time near the end with some last minute design changes. Overall, we spent $128.89 on individual PCB components from Digikey, and $127.88 on PCBs from JLCPCB.

0.15 Lessons Learned

Throughout the development of the CCD Small Satellite Readout System, our team gained a deeper understanding of how complex it is to integrate hardware, firmware, and customer requirements into a single functioning prototype. One major lesson was the importance of early, detailed requirements analysis, especially for critical constraints such as low noise performance, clock timing precision, and strict power limits, all of which were essential for operating the CCD sensor effectively. We also learned that clear communication and consistent documentation were vital to keep the work synchronized across hardware and software divisions. Several design chanllenges such as managing noise sources, interpreting CCD clocking diagrams, and validating digital interfaces proved the value of incremental testing and simulation before executing hardware changes. Finally, we learned that unforeseen issues are inevitable in such a multidisciplinary engineering project, and success depend on adaptibility, structured problem solving, and maintaining strong collaborative habits. These lessons will guide our future engineering work and prepare us for larger scale, real world system development.