NATIONAL TECHNICAL UNIVERSITY OF ATHENS
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING
CIRCUITS & SYSTEMS GROUP

Design and Implementation of a Real-Time Industrial Data Acquisition System for Data Analytics and Predictive Maintenance

DIPLOMA THESIS

of

Epameinondas Orestis Batsis

Supervising Professor: Paul - Peter Sotiriadis
Professor, ECE NTUA

Athens 2022

NATIONAL TECHNICAL UNIVERSITY OF ATHENS
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING
CIRCUITS & SYSTEMS GROUP

Design and Implementation of a Real-Time Industrial Data Acquisition System for Data Analytics and Predictive Maintenance

DIPLOMA THESIS

of

Epameinondas Orestis Batsis

Supervising Professor: Paul - Peter Sotiriadis
Professor, ECE NTUA

Aproved by the three-member committee, on June 14 2022.

……..	……..	……..
Paul P. Sotiriadis	Yoannis Papananos	Evangelos Hristoforou
Professor NTUA	Professor NTUA	Professor NTUA

Athens 2022

……
Epameinondas Orestis Batsis

Electrical & Computer Engineer NTUA Graduate

Abstract

It has become clear that **industrial production data** is becoming more and more important these days since companies try to predict and optimize production plans. Lots of progress has been made in the development of models that achieve the aforementioned, but without the actual raw information, it is impossible to make accurate real-time assumptions.

Precise, continuous, and robust machine data is one of the most important sources of information when trying to make decisions on a production plan.

This diploma thesis analyses the design decisions and the implementation methods of a **Real-Time Industrial Data Acquisition System**, that can connect to a variety of external industrial sensors and monitor key measurements for performance and reliability. Data collected by the system can be then utilized for analytics such as production remote monitoring, real-time optimization, and predictive maintenance.

The system offers many different ways of connectivity to cover a large part of the industry and a fully configurable architecture that allows customization, without the need for extra technical skills. It can be deployed to almost any Industrial environment with minimum infrastructure, allowing for a fast and reliable data acquisition plant. It is essentially a platform that can host custom applications, adapting to each solution thanks to the enhanced modularity with which it has been designed.

This thesis includes six chapters. The first chapter, “Introduction”, contains a brief description of the System’s key features and overall architecture, as well as some high-level design decisions. In the second chapter, “System Analysis”, the overview of the system’s architecture and the main circuit analysis are described. In chapter three: “Implementation”, the methodology and techniques used in the design of the printed circuit board and the firmware development are presented. In chapter four, “Results”, the outcome of the experiments is presented and the overall performance of the system is evaluated. In chapter five,“Conclusion”, the Thesis outcome is discussed, and the key points of the System are pointed out. Lastly, in chapter six “Discussion and Recommendations for Future Research”, all known issues, limitations, and improvements are presented, as well as elements for further development and upgrade of the system.

Acknowledgments

First of all, I would like to warmly thank Mr. Pavlos P. Sotiriadis for the supervision of my thesis, and the opportunity he gave me to be part of the Circuits & System lab of National Technical University of Athens. Also, I would like to thank all the staff of the department, and especially Konstantinos Papafotis for the excellent cooperation and the friendly atmosphere we had throughout the preparation of this project.

Finally, I would like to thank all those who contributed to my journey during these five years. My family who supported me financially and psychologically, my friends who stood by me every step of the way and the teaching and administrative staff of the school, for helping me acquire such a high level of scientific knowledge in the field of electronics and programming.

Table of Contents

List of Tables

List of Figures

Abbreviations

- ADC: Analog-to-Digital Converter
- API: Application Programming Interface
- BOM: Bill of materials
- CAN: Controller Area Network
- COM: Communication Port
- CRC: Cyclic Redundancy Check
- DAQ: Data Acquisition
- DC: Direct Current
- DC-DC: DC to DC Converter
- DEV Board: Development Board
- DHCP: Dynamic Host Configuration Protocol
- DRC: Design Rule Check
- EDA: Electronic Design Automation Software
- ERC: Electrical Rule Check
- ESD: Electrostatic Discharge
- FW: Firmware
- GND: Ground
- GPIO: General Purpose Input Output
- HAL: Hardware Abstraction Layer
- HW: Hardware
- I2C: Inter-Integrated Circuit
- IC: Integrated circuit
- INL: Integral Nonlinearity
- LED: Light Emitting Diode
- LDO: Low Dropout Regulator
- LSB: Least Significant Bit
- LWIP: LightWeight IP
- MAC: Media Access Control
- MCU: Microcontroller Unit
- MSB: Most Significant Bit
- MSPS: Mega Samples Per Second
- PC: Personal Computer
- PCB: Printed Circuit Board
- PHY: Physical Layer
- PoE: Power over Ethernet
- PSRR: Power Supply Rejection Ratio
- RC: Resistor Capacitor Network
- RMII: Reduced Media Independent Interface
- RX: Receive, Receiver
- SAR: Successive Approximation Register
- SPI: Serial Parallel Interface
- SW: Software
- SWD: Serial Wire Debug
- TX: Transmit, Transmitter
- UART: Universal Asynchronous Receiver Transmitter
- USART: Universal Synchronous Asynchronous Receiver Transmitter
- USB: Universal Serial Bus

1 Introduction

This diploma thesis was developed in the **Circuits & Systems Lab** of the National Technical University of Athens. The objective was to design and implement a Data Acquisition System that can be deployed in almost any Industrial environment and stream machine production data to cloud-based applications. This data can be used to remotely monitor key measures that affect the production performance of a company, combine different measures of a machine to predict its required maintenance before it is needed, thus reducing unsuspected stop times, and optimizing the usage to fit the demand of the market.

The target was to design the system in such a way that it is modular enough to be used in different environments, while at the same time being easy for someone without advanced technical skills to install it and set it according to the application and the measurement requirements.

Since the system is not a specialized acquisition machine but rather a **general-purpose low-cost DAQ platform**, the specifications of various parameters such as sampling frequency and bandwidth, linearity, accuracy, and repeatability, were intentionally not fully evaluated in order to further reduce cost and development time. The system is meant to interface with generic industrial analog and digital sensors and provide accurate enough data for someone to make long-term decisions and log the performance of a machine.

This thesis covers the design and implementation of the hardware and the firmware of the system but not the implementation of the application for forecasting and applying predictive behavior. As mentioned earlier the scope of the thesis is the design of a hardware platform that can easily adapt to different measuring applications and provide constant data for later processing.

However, for the reason of demonstration, an initial monitoring application was developed to provide proof of concept, evaluate the overall performance of the system and showcase its abilities.

1.1 System Requirements

As the first step of the study, the system requirements had to be fully defined. In this way, existing commercial solutions and other similar studies/open-source products could be evaluated and compared against the requirements in order to get a full overview of the market and the products already running.

Modularity was the first and most important aspect of the system under study. This means easy connection with power and sensors, as well as the ability to expand the number of sensing elements without the need for excess hardware and installation. The minimization of extra cables running through an industrial plant was also one of the main concerns.

Reliability was another very important aspect because the decisions taken will be based on the output of the system. The data produced needs to be accurate to avoid taking the wrong decisions and costing unnecessary money to a company.

Serviceability had to be considered, but as the system had to be almost plug-and-play, the major part would be reconfiguring parameters or adding new, rather than actually servicing or updating the hardware.

Robustness in an Industrial environment is vital and determines the life of a product targeted for these kinds of applications. To ensure endurance and compliance, the proper best practices had to be applied to give the system the ability to withstand harsh environments and function properly.

Connectivity is a part of modularity mentioned before. To cover the largest part of the industry, the system had to be TCP/IP compatible in order to connect directly to a network and stream the data through it. As for the sensors' connectivity, the target was to be able to read measurements from almost all industrial sensors, including

- Analog sensors with output up to 10V, min 12 bits of resolution (at least 2)
- Sensors with RS232 and RS485 digital output (at least 1 of each)
- Sensors with 4-20mA output (at least 2)
- Sensors with CAN-BUS 2.0 output (at least 1)
- Sensors with digital I/O channels up to 24V. (at least 2)

Cost is one of the factors that have to be considered when producing multiple devices and not only prototypes. However, since all the above requirements are really crucial when designing a product for the industry, it was decided that it would be evaluated after all previous requirements were met, to further reduce manufacturing and parts costs. However, since this Thesis also demonstrates the manufacturing of some prototypes of the system, a maximum cost of 200 euros per prototype had to be set.

Considering the Sampling rate, and signal frequency range, it was decided that the sensors targeted for this system were going to be sampled at a maximum of **10 Hz** and the frequency of the signal to be measured would be **1 to 5 Hz**.

Table 1 summarizes the System's requirements and their respective priorities according to the study objective.

Requirement	Priority
Modularity	HIGH
Reliability	HIGH
Serviceability	MEDIUM
Robustness	HIGH
Connectivity	HIGH
Cost	MEDIUM

Table 1.

With the System's requirements defined, research of existing solutions had to be made.

1.2 Existing Solutions

An evaluation of existing commercial and open source solutions was made, in order to extend the view of what is available in the market, as well as the capabilities and the cost of the products. Below is a list of the most common industrial DAQ systems on the market with various capabilities and different price tags.

dataTaker DT82I

2 Channel Universal Input Industrial Data Logger

- Up to 6 Analog (±50V) Sensor Inputs
- 8 Flexible Digital Terminals
- 2 Serial 'Smart Sensor' Ports
- Programmable Analog Output
- Modbus for SCADA Connection
- Web & FTP Client / Server
- USB Memory for Easy Data and Program Transfer
- Download Specification Sheet

$1,950.00

dataTaker DT85

16 Channel Universal Input Data Logger

- Up to 48 Analog (± 50V) sensor inputs
- Expandable to 960 analog inputs
- 12 flexible digital terminals
- 2 Serial 'Smart Sensor' ports
- SDI-12 (multiple networks)
- Modbus for SCADA connection
- Web & FTP client / server
- USB memory for easy data & program transfer
- Programmable Analog Output
- Download Specification Sheet

$3,980.00

DIN rail data acquisition system
ELOG DATA LOGGER | Enerdis

modular · multi-channel · multi-sensor

Sold by:

Chauvin Arnoux — Macherio, Italy

★★★★★ Feedback on the quality of responses (from 1 buyers)

$ Request price options

Request a personalized quote

DESCRIPTION

ELOG DATA LOGGER from Enerdis®, a high-performance, stand-alone, economical data logger open to all your analytical tools.
Collect, record and export all your energy data
Essential for energy data monitoring

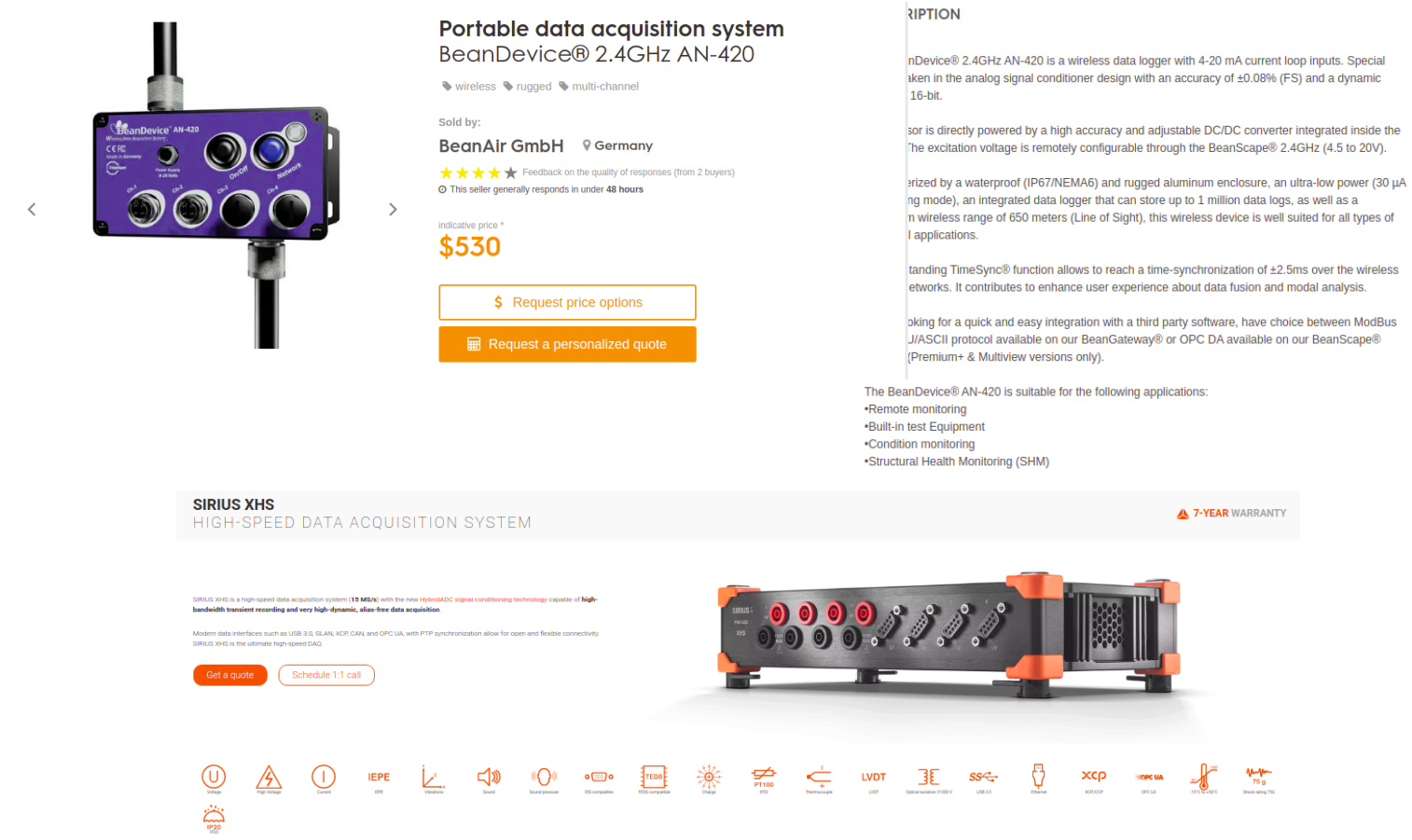
Open data logger - all equipment manufacturers
Remote retrieval of all energy data
Data file export for analytical applications
Local backup of data files covering several years
Simplified setup via web pages

COLLECT
Several communication ports for remote data retrieval from the multi-function, multi-manufacturer equipment on an installation (meters, PLCs, power monitors, transducers)

RECORD
Recording of up to 100 variables
Configurable recording periodicity (5 s to each variable
History of last 3 months

CALCULATE
Integrated calculation functions for the variables so that you can view and record data which can be processed directly (ax+b, x/y)

EXPORT
Data file export (csv, xml) is fully configurable (choice of variables, file formats, transmission periodicity, FTP servers)
Specific formats can be developed on request



Although this list of DAQs is quite small, it summarizes the different categories of systems out in the market. However, it can be seen from the specifications and the prices of the products that, although some of them cover the technical requirements set in the previous paragraph, their prices are very high (for the products that do not name a price, an estimation was made, considering their specifications). It is therefore concluded that none meets the requirements set in the previous paragraph and that a logical solution would be the design of a custom DAQ System, including the PCB design and the Firmware development that covers the requirements while staying inside the budget.

What is also worth mentioning is that although wireless technology was considered as a possible solution, due to the fact that it minimizes the number of cables for the installation, a number of downfalls were spotted. First of all, if we considered a battery-powered solution, the serviceability requirement would be compromised since service in changing batteries would be needed. Due to the TCP/IP compatibility stated in the requirements, regardless of the device's power plan, extra HW would be needed to support a wireless solution, since the data should have to be transferred wirelessly and then through another device that translates them in TCP/IP, thus further increasing the cost and the development time.

2 System Analysis

Having already concluded that a custom solution would be needed to meet the requirements of the system, the overview of the solution had to be fully defined before the design phase could begin.

W-model was chosen as the methodology to design and implement the System. In contrast with the classic Waterfall and V-model, the W-model is oriented on checking the successful testing of every stage of the development. The W-model fundamentally represents the standard development cycle where each development stage is shadowed by a testing or evaluation stage, ensuring conformity and compliance with the pre-defined targets and requirements.

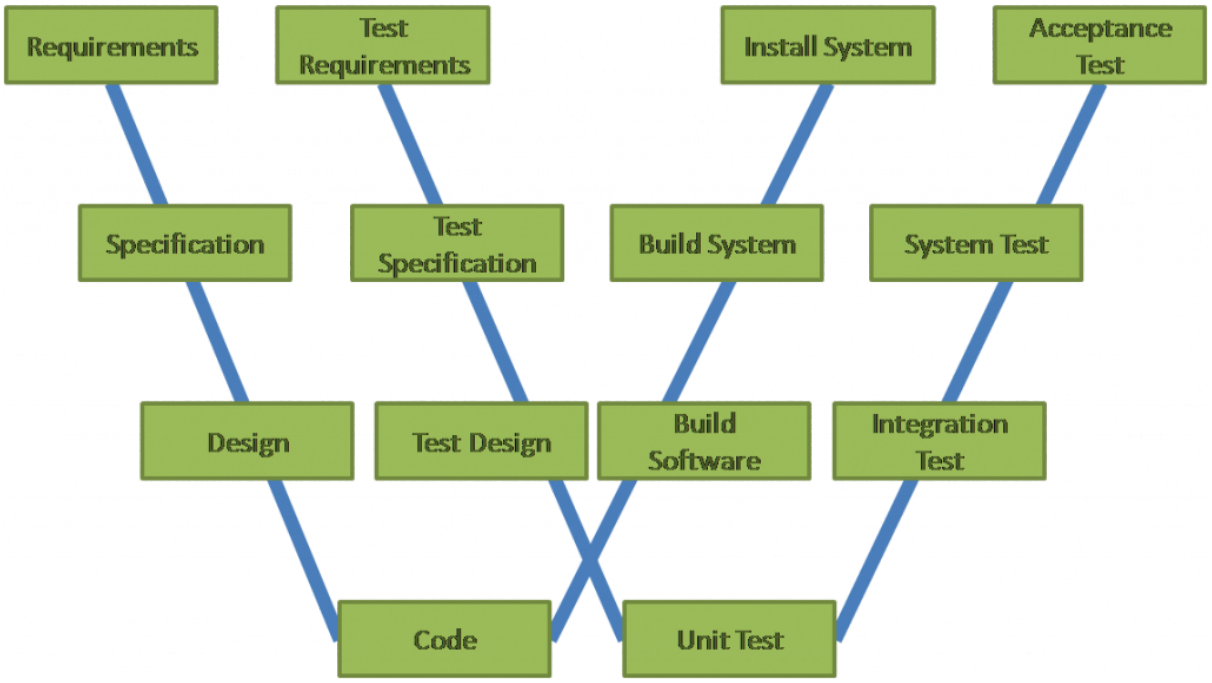


Figure 1.

Since all the requirements of the system were defined in paragraph one, it was time to determine the architecture and the specifications.

2.1 System Overview

Since the system is targeting general-purpose industrial sensors, a low-cost microcontroller-peripheral architecture would ensure fast development and adequate performance while at the same time making the firmware development much simpler.

For these reasons, the decision was made to proceed with this kind of architecture, keeping in mind the requirements set in paragraph 1.1.

An alternative solution would be to use a much more expensive MCU with higher specifications and dedicated internal peripherals. However, to avoid increasing the cost, and the development time (HW and FW), this solution was not further evaluated as it would also deviate from this thesis scope.

At this stage, several decisions needed to be made before starting the design phase. The specifications of each component needed to be reviewed, in order to ensure conformity with the requirements, while at the same time providing system-level compatibility between the various peripherals.

Figure 2 presents an overview of the system's components and an abstract architecture of the MCU and peripherals.

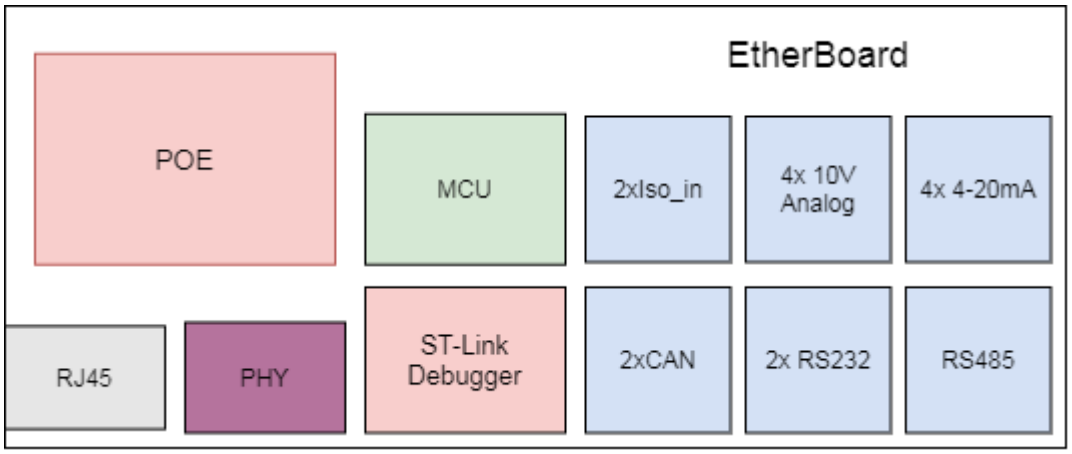


Figure 2.

2.1.1 MCU Selection

As a first step, the selection of the MCU was needed to take place, in order to be able to select the appropriate peripherals.

Table 2 presents some of the potential MCU for the selection, as well as their specifications, number, and variety of peripherals, FW development tools, and available DEV boards.

At this point, it is important to mention that in order to follow the W-model and to be able to perform testing at every step, the use of a DEV Board was deemed crucial. That is the reason why the existence and the variety of DEV Boards of each MCU were also evaluated.

Part Number	Vendor	Arch	Core Frequency (MHz)	Number of pins	Peripherals	DEV Boards	FW Tools
ATMEGA328P	Microchip	8-bit	16	32	USART, SPI, I2C, 10bit ADC	Arduino Uno	Atmel Studio
STM32F103	STM	32-bit	72	100	3xUSART, 2xSPI, 1xCAN, 2xI2C, USB 2.0, 2x12bit ADC	Blue Pill	STM32 Ecosystem
TM4C129x	Texas Instruments	32-bit	120	128	8xUSART, 4xSSI, 2xCAN, 10xI2C, 10/100 Ethernet MAC, USB 2.0, 2x12bit ADC	TIDM-TM4C1 29POE	Code Composer Studio
STM32F407	STM	32-bit	160	140	4xUSART, SDIO, 2xCAN, 3xSPI, 3xI2C, 10/100 Ethernet MAC, USB 2.0,2x12bit ADC	NUCLEO-F42 9ZI	STM32 Ecosystem
ESP32	Espressif	32-bit	240	48	Ultra-low-power co-processor, WiFi, Bluetooth, 3xUSART, SDIO, CAN, 4xSPI, 2xI2C, 2xI2S, 10/100 Ethernet MAC, 12bit ADC	ESP32 DevKit	ESP-IDF

Table 2.

Looking at Table 2, we can see different solutions with cheaper and smaller ICs that would need extra peripherals to fulfill the requirements, as well as larger more expensive MCU ICs that incorporate the peripherals in them.

The decision was made to work with the larger ICs that contain the extra peripherals, in order to save some development time in the HW (no need to use extra ICs, decoupling capacitors, and rout the connections), as well as in the FW, since the risk for potential problems with the external peripherals could take time from the design and implementation phase, as well as from the debugging phase later on. Furthermore, the compatibility with the peripheral would be already established by the vendor, making the coding phase much easier.

In the case of external peripherals, the standard connectivity with the MCU is SPI or I2C. In both cases, low-level libraries have to be developed in order to interface with the peripherals. These libraries, when on runtime, increase the execution time of the MCU, and when multiple peripherals are connected (as in our case), the code execution speed can increase dramatically.

These are the main reasons that the cheap general purpose MCUs were ruled out from the selection.

Comparing the larger MCU ICs, we can see that they have almost identical specifications. The selection criteria in these cases lie on the vendor selection rather than the actual MCU. The reason is that different vendors offer different development solutions. Sometimes the solutions are free to use and others not. Some vendors offer low-level libraries for all the peripherals of the IC and the core of the MCU, which drastically reduces the FW development time.

Looking at STMicroelectronics, all of their development tools are free to use, which not only allows for an easy-to-start solution, but it ensures that the software support and the tutorials available on the internet are substantially more than those of paid software. STM offers the **STM32CubeIDE** which is an integrated development environment that includes the MCU selection section, the initialization of the HW via a graphical interface tool called STM32MXCube, an eclipse based editor, and a debugger all in one package.

So you can initialize the HW, which in turn generates all the low-level libraries for the peripherals and the core and automatically transfers all of that into the editor where the application can be implemented.

Having all the low-level code written and documented efficiently by an auto-generated tool before you even start developing is a major advantage overwriting everything by the manual and risking making mistakes that in turn cost development time.

That said, choosing **STM32f407** as the MCU of the system was an easy choice, considering all the above aspects, including the sufficient peripherals, clock speed, pin count, and the development tools available.

In order to program the MCU, an external debugger would be needed. Since the first device designed would be the prototype and a lot of different FW solutions would be tested, it was decided to i**ntegrate the debugger in the system**, in order to have easy and quick access to uploading and debugging code.

2.1.2 Power Plan

The term Power Plan refers to the components in the System that are responsible for the power delivery. These can be voltage regulators, DC-DC converters powered by external sources, or special purpose power sources.

For the needs of the System, and looking at the requirements, it was decided to develop a **PoE stage** that can power the board through the ethernet wires. In this way, true minimization of the installation cables could be achieved, since the device would need a single ethernet cable for the data and the power connection (sensor wiring would be extra either way).

However, for the purpose of testing the System in the early stages, an extra DC-DC converter was chosen to be added to the power plan, in order to be able to power the system eternally with a range of voltages without the need of a PoE switch while performing tests.

Since the onboard debugger mentioned in the previous section needs to be connected via a USB 2.0 to transfer the information to and from the System, it was decided to utilize the connection and deliver the +5 Volts of the USB directly to the System, increasing the options of power delivery further.

For the PoE stage, **TPS23753Aa** was used. It is a PoE interface and a DC-DC converter with enhanced ESD protection, enabling the coupled power into the ethernet pairs to be rectified, and converted to usable low-level voltages. This part was chosen because it combines all the needed features with an affordable price as well as availability in the market.

The extra DC-DC converter chosen was **ACT4065ASH-T** because it combines a wide input range of 6 to 30V with high efficiency of 95% and output power of 2A, which is more than enough for the System.

All of the above have a fixed output voltage of +5V with respect to the ground of the System. However, most of the external peripherals and the MCU require +3.3V to operate. For that reason, a Low Dropout Regulator (LDO) was used to transform the +5V of the various inputs to the +3.3V usable voltage. The part used was **NCP187** because it can deliver enough power to the MCU and peripherals with **low output noise** and **very good PSRR**, incorporating several **protection features** such as Thermal Shutdown, Soft Start, Current Limiting, and also Power Good Output signal for easy MCU interfacing.

Since the onboard debugger also needs +3.3V to operate, a separate LDO (**MCP1812B**) was selected, as we only want the debugger to be powered when the USB is connected and not when other +5V power sources are present.

Table 3 contains all the power sources of the System, as well as their input and output voltage ranges and the condition at which they function.

Part Number	Device Type	Input Voltage	Output Voltage	Condition
TPS23753Aa	PoE DC-DC	36-58V	5V	When ethernet is connected to a PoE Switch
ACT4065ASH-T	DC-DC	6-30V	5V	When external supply is connected
-	USB	-	5V	When Debugger USB is connected
NCP187	LDO	3.6 - 5.5V	3.3V	When +5 volts are present on the board
MCP1812B	LDO	4.0 - 5.5V	3.3V	When Debugger USB is connected

Table 3.

It is noted that the main 3.3V LDO's input is attached to a +5V rail that is connected to all 5V components. In this way, if any or multiple devices that output +5V are powered, the main LDO is also powered.

2.1.3 Peripheral Selection

The MCU selected in paragraph 2.1.1 contains most of the peripherals stated in the system's requirements. Some of them are the CAN controller, the 10/100 MAC ethernet RMII, and the GPIOs. The MCU also contains several UART interfaces, and one of them was used to communicate with the debugger in order to transfer data through a virtual COM port.

Ethernet

The MCU contains the interface for the ethernet called a MAC controller, but as with most interfaces, a PHY is needed. The PHY is responsible for the connection with the world outside the system and it contains all of the necessary protection features to ensure safety in case of malfunctions or wrong handling of the device.

DP83848 of Texas Instruments and **LAN8720** of Microchip were the two PHYs that were evaluated and matched the requirements. However, due to the chip shortage of 2020-2021, only **DP83848** was available on the market at that time, so this was the one selected for the system. The two ICs have very similar performance and some of the differences are that the LAN8720 has more I/O ports for status LEDs and other functions, whereas DP83848 comes in a smaller form factor with fewer functions. Both PHYs incorporate ESD protection, Auto-negotiation, Automatic polarity detection, and correction and they are Compliant with IEEE802.3/802.3u (Fast Ethernet) and ISO 802-3/IEEE 802.3 (10BASE-T).

CAN-BUS

Similar to Ethernet, the MCU includes two CAN controllers that handle all the functionality of the bus, but external transceivers are needed in order to interface with the world outside the system. The transceivers protect the MCU from unwanted overvoltages and ESD events and they also translate the differential bus voltage into 3.3V TX and RX signals.

SN65HVD230D was selected since it was compatible with the 3.3V logic of the MCU, it combines a relatively small package with a reasonable price and it was available in the market at that time.

RS232

Since the MCU has a lot of USART peripherals that are not used and at least one transmitter and one receiver in the same package were needed, **MAX232E** was chosen as the RS232 interface IC. It contains 2 sets of ESD-protected transmitter/receivers and it comes at a low cost and a small form factor.

RS485

Similarly, for the RS485, a transceiver was needed in order to interface with other RS485 compatible devices. **SN65HVD12D** was chosen since it contains a transmitter and a receiver that are ESD protected at a low price.

ADC

Although the MCU contains three ADC modules, it was decided to use an external ADC in order to increase the accuracy of the system and to be able to measure higher voltages.

The MCU's internal ADC modules are three 12bit SAR ADCs running at 2.4 MSPS, which is a high sampling frequency for an integrated ADC. However, their measuring range is limited to 0-3.3V. The overall error of the internal ADCs can be seen in Table 4 and Figure 1 below.

Symbol	Parameter	Test conditions	Typ	Max(1)	Unit
ET	Total unadjusted error	f_{PCLK2} = 60 MHz, f_{ADC} = 30 MHz, R_{AIN} < 10 kΩ, V_{DDA} = 1.8(2) to 3.6 V	±2	±5	LSB
EO	Offset error		±1.5	±2.5	
EG	Gain error		±1.5	±3	
ED	Differential linearity error		±1	±2	
EL	Integral linearity error		±1.5	±3	

Table 4.

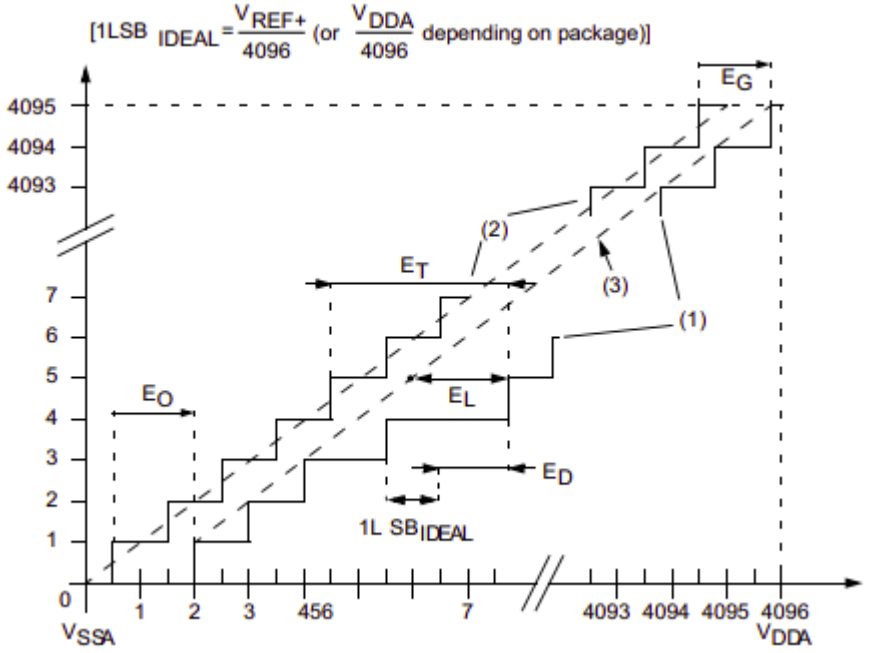


Figure 3.

As seen from Table 4 and Figure 3 above, and as verified by a quick experiment, the overall error of the internal ADCs of the MCU is close to ±5 LSBs.

Since the Internal ADCs of the MCU could not reach the desired voltage range, and also their performance is not the best, it was decided to evaluate the solution of placing an external ADC that could communicate with the MCU via SPI.

It is worth mentioning that the performance of the internal ADC can be increased if we introduce filtering in the HW in the form of an RC circuit and the FW by applying a digital filter, or average through a series of continuous samples. However, the above solutions, although feasible, can not substitute the performance of a better ADC. A HW solution to measure 10V with the 3.3V ADC is to place a voltage divider in the input. This however would require precision resistors to work, and it would limit the range of each channel to the selected resistor ratio placed on the PCB.

It was obvious that the internal ADCs could not meet the requirements and the proposed fixes could only make a part work.

So, in order to acquire more accurate samples and to measure the desired voltage range, several external ADC modules were evaluated. Table 5 presents the specifications of the ADCs that best fit the requirements and come at a reasonable price.

It is important to mention that even though the measuring range of the desired ADC is 10V, the supply voltage could only be +5V since this is the higher regulated voltage of the system.

Part Number	Resolution (bits)	Measuring Range	Sampling Rate	Number of channels	Supply Voltage	INL	Vendor
ADS8668	12	±10 0-10	500 ksps	8	+5V	±0.5LSB	Texas Instruments
LTC185X	12	±10 0-10	100 ksps	8	+5V	±1LSB	Linear Technologies
MAX1032	14	±12 0-12	115 ksps	8	+5V	±0.25LSB	Maxim Integrated

Table 5.

We can see that the specifications of the three ADCs are very close. MAX1032 has a higher resolution and lower INL but it also has a lower sampling rate. Since 12 bits of resolution are enough in order to measure industrial general purpose sensors, the decision to use ASD8668 was made since the higher sampling rate can be utilized in order to perform averaging for even better results.

Isolated Digital I/Os

In order for the system to be able to interface digital I/O signals safely, isolation is needed between the MCU and the other device. For that reason, a 24-60V Isolated two digital input receiver (**ISO1212DBQR**) with reverse polarity protection was chosen to fully protect the MCU from unwanted overvoltages and ESD events. This solution minimizes the components needed in comparison with a traditional optocoupler and current limiting resistor network that would also need complementary components for the ESD protection.

At this point, it is worth mentioning that all of the above peripherals were selected carefully in order to ensure that the system would stay protected and comply with the industrial standards. Through ESD and surge protection features, all of the peripherals are immune to unwanted overvoltages and spikes, which safeguards the system from unwanted damage and malfunction.

2.2 Circuit Analysis

Having selected all the major parts of the system, it was time to design the electrical schematics that fully describe the connectivity between each component and can then be used for the actual HW design.

The software used for the schematics and HW design was Altium Designer. It is one of the most advanced and used EDA software in the market, offering large flexibility and numerous features that make the design process easier and more precise.

After starting a new project, separate schematic files were created in a hierarchical manner, each one for every section of the system. These are

- Connectors
- MCU
- Ethernet section
- Interfaces
- Power Stage
- ST-Link

2.2.1 Hierarchical Schematic

Figure 4 shows the hierarchical schematic that describes the connection between each of the above sections. Each section (green/brown parts), contains a number of ports (yellow parts) that describe actual signals in the respective schematic. Ports have the same name as the signal and a direction (input, output, bidirectional, or none). These features are used to identify the signals and prevent the designer from making potential mistakes, such as using an input as an output.

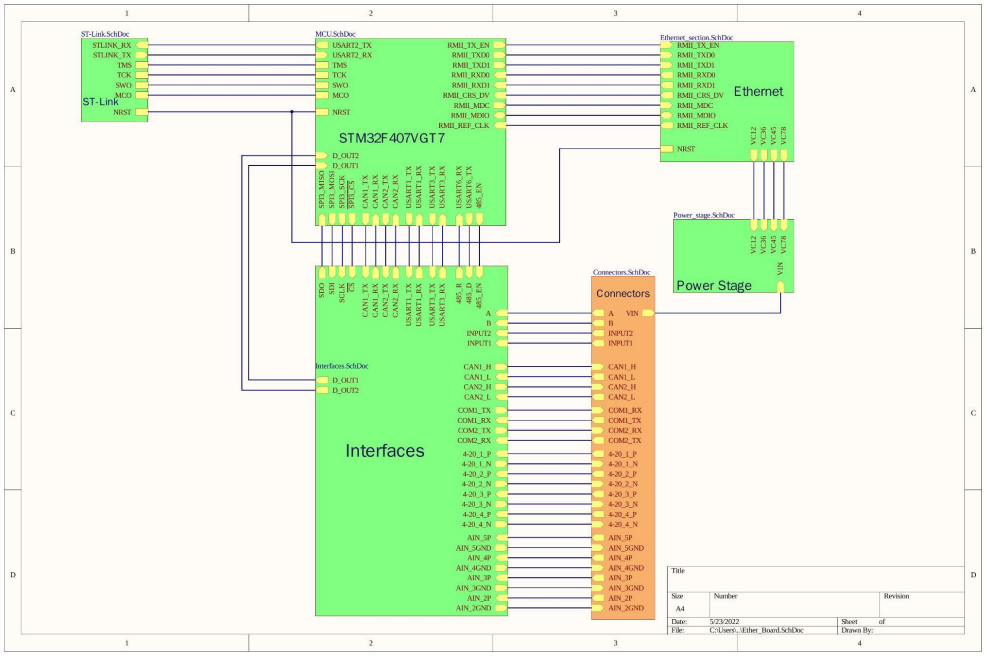


Figure 4.

Even though at this stage the design just describes the connectivity of the system, the characteristic of each signal, when created, is transferred through the schematics and eventually reaches the model of an actual part that has defined inputs, outputs, and power pins according to its manufacturer. For that reason, it is very important to fully and correctly define each net (as the signals are named in Altium), and its parameters, in order to be able to run the ERC which in turn reports all the design errors and prevents potential design mistakes.

2.2.2 Connectors' Schematic

The first schematic, which is shown in Figure 5, populates the various connectors that will be present in the physical board, which will give the system the ability to interact with actual electrical signals. As seen, the choice was made to use multiple smaller connectors instead of one or two large ones. This was done to physically separate the different groups of signals such as analog, CAN, RS232. This gives the ability to connect and disconnect connectors of a group without interfering with other groups while the system is powered.

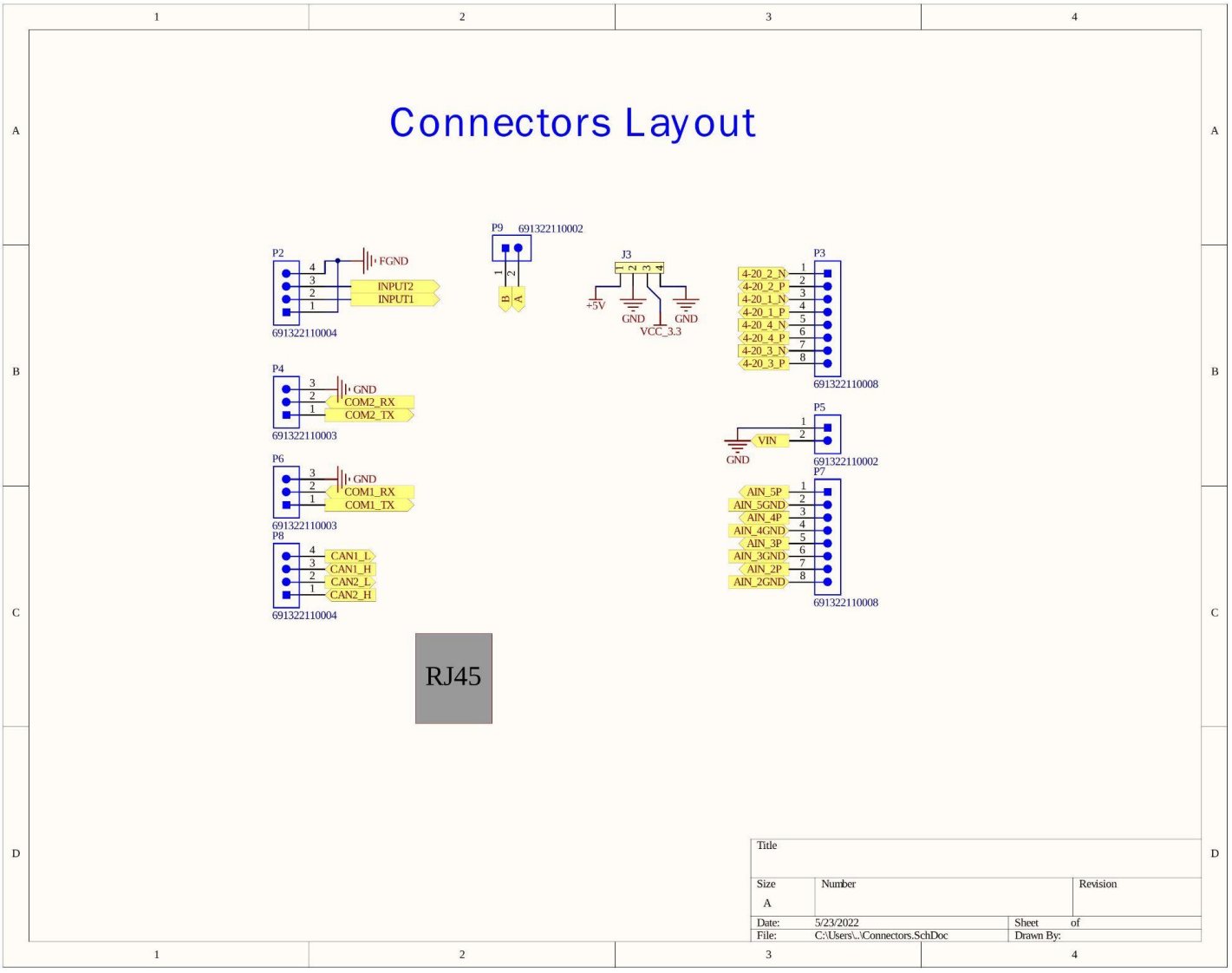


Figure 5.

The type of connectors selected is the standard green industrial style with a set screw for each pin, giving the ability to connect external wires with the use of an ordinary Philips or flat-head screwdriver.

2.2.3 MCU Schematic

The next schematic shown in Figure 6 contains the MCU and all the passive components recommended by the part's datasheet for its proper use, as well as some complementary components.

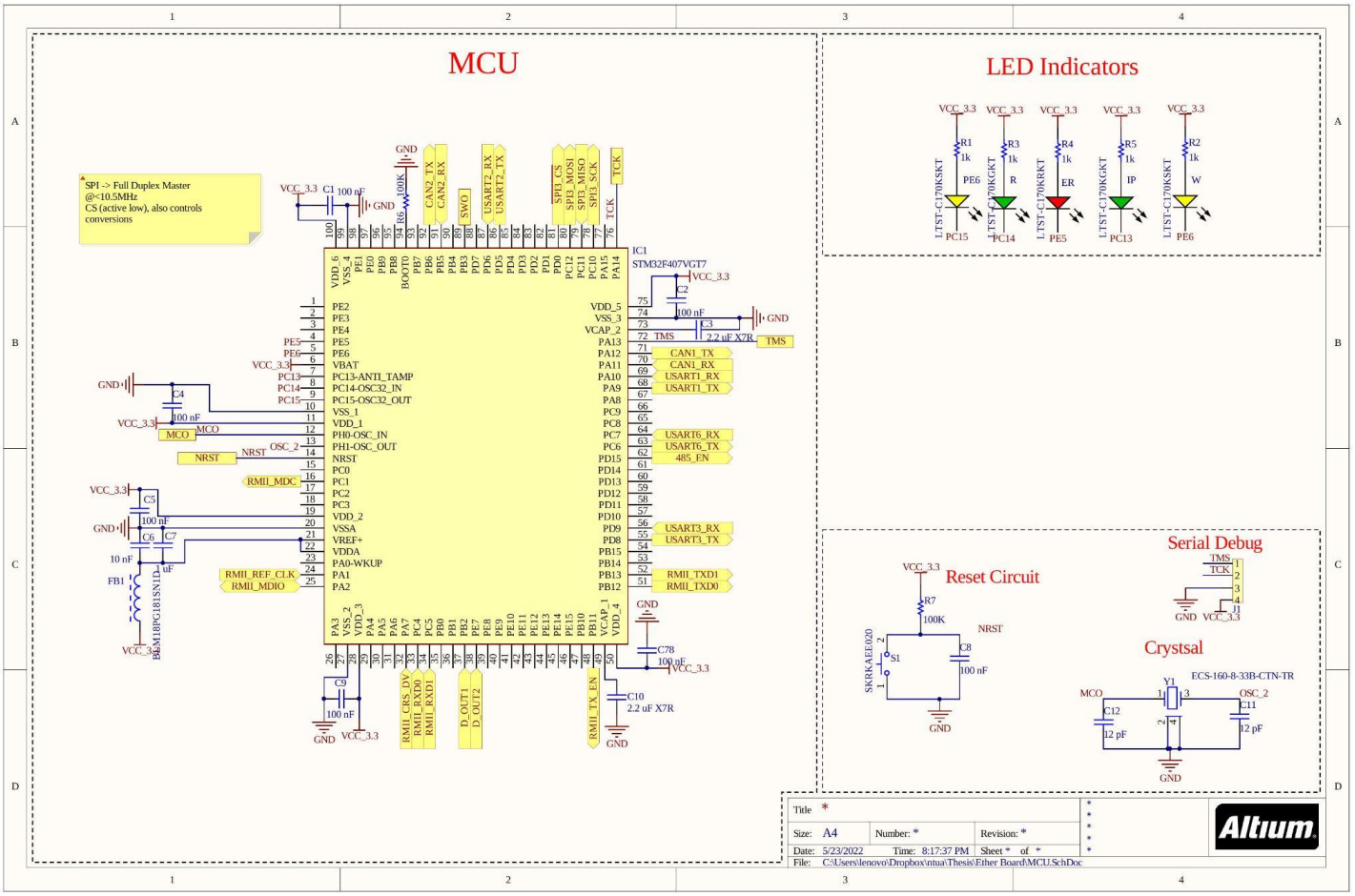


Figure 6.

One of the most important passive components of digital electronics when dealing with ICs is the decoupling capacitors. Their main purpose is to filter out high-frequency power supply ripple and to provide an instant charge when the IC requires slightly greater power. For that reason, they are placed very close to the IC's power pins and they are sized so that they provide enough charge, without pulling large amounts of current and needing substantial time during charging. Some of the most common capacity values used as decoupling elements for commercial ICs are 100nF, 1uF, and 4.7uf, as well as parallel combinations of those.

We can see one 100nF decoupling capacitor at each supply pin of the MCU, as well as other complementary capacitors needed for the proper use of the chip.

The analog supply of the chip is further filtered by the combination of a 10nF capacitor and a ferrite bead that attenuates high-frequency noise, providing a more clean and stable supply voltage.

As recommended by the schematic, the reset pin is decoupled with one 100nF capacitor and pulled up to 3.3V with a 100K resistor, in order to avoid accidental resets caused by noise or supply ripple. The ceramic capacitor in combination with the pull up resistor, helps absorb the mechanical oscillations of the switch and prevent the MCU from resetting multiple times.

The resistor R6 (100k) is present to set the boot mode of the MCU. The chip offers the options of booting (starting to execute instructions) from the embedded flash memory, or the embedded bootloader that can mainly execute instructions from the UART, CAN, or ethernet. With the R6 we pull the dedicated pin to GND to ensure that during startup the MCU will boot from the FLASH memory where the instructions programmed by STM32CubeIDE are stored.

The capacitors C3 and C10 (2.2uf X7R) are placed in the dedicated VCAP pins as per the manual of the chip. Their role is to decouple the internal regulator of the chip and provide a smooth voltage.

A 16MHz crystal ceramic oscillator is connected to a pair of specified pins of the chip (PH0, PH1) in order to provide a sinusoidal signal with a stable frequency that is used to synchronize the internal clock of the MCU. In this way, the clock source that feeds the microcontroller, the timers, and the rest of the peripherals is continuously adjusted to achieve greater accuracy, allowing it to measure and capture events with higher time precision. For its proper use, two load capacitors are needed (between each pin and GND) the values of which is calculated by the following formula:

$$C_{Load} = 2 * (C_L - C_{stray})$$

Where,

C_{Load} is the capacitance of each load capacitor,

C_L is the load capacitance of the ceramic crystal oscillator, and

C_{stray} is the capacitance of the joint between the pads of the crystal and the PCB pads.

By reviewing the crystal's datasheet we can see that C_L is 8pF. Assuming C_{stray} is between 2 and 5pF, we can see that a standard 12pF capacitor is a good choice for each load capacitor. [1]

A set of five small LEDs controlled by the MCU was placed in order to provide a visual indication of the state and various events and faults. For the LEDs to operate correctly and not be damaged, a current limiting resistor was placed in series with each LED. The value of each resistor can be calculated by solving a simple Kirchoff Voltage Law in the MCU LED loop.

We have

$$3.3V = I_{LED} * R + V_f$$

Where,
the voltage supply is $3.3V$,
I_{LED} is the loop current,
R is the resistor value in Ohms, and
V_f is the forward voltage of the LED specified in the datasheet.

Each different color LED has a different forward voltage. At 20mA they typically have:
Green: 2.1 V
Red: 2.0 V
Yellow: 2.4 V

We need to decide the current we want to pass through the LED. Since we only need a visual indication when looking closely at the PCB, the light intensity should be dimmed in order to reduce current consumption. A value of 1 to 2mA was selected, which could then be adjusted by changing the current limiting resistor to fine-tune the desired intensity. Solving the above equation we can see that a good value of resistance is 1kOhm.

At this point, it is worth mentioning that when designing a PCB, it is very helpful to try and use the same components where possible, when no strict rules need to be followed. This is one of these cases, where we do not have an exact intensity specification for the LEDs, but we just want to have a visual sign when developing the firmware and when we want to indicate faults and/or events.

By doing the above, we simplify the BOM, which in turn saves us time for searching and sourcing the new parts as well as an extra cost of adding a new part in the BOM (the unit price especially in passive components is very small, and the quantity to price relation is not linear).

Thus, selecting the current limiting resistor for all status LEDs to be 1k is a move in the right direction.

Except for the passive and the complementary components, another kind of attribute can be seen in the schematic. The so-called ports.
They are connected to the MCU pins and are usually named after the signal they attach to. Ports allow these signals to be connected with other ports in different schematics, thus making the process of designing more clean and modular. The connectivity between schematics has already been explained in paragraph 2.2.1 and it is shown in Figure 3.

For this case, most of the ports describe connections between the MCU and the various peripherals, as well as with the onboard debugger, that are both designed in different schematics.

2.2.4 Ethernet section Schematic

Some of the reasons that separate schematics exist, are the modularity, which allows the reuse of certain parts for and of new designs, the amount of information a person receives when trying to debug or understand the drawing (imagine searching for something in a schematic that includes all of the design's sections in one place), as well as the group design perspective which is what happens in most industries.

Figure 7 represents the Ethernet section of the design. It includes the PHY and its complementary components, as well as the RJ45 connector. The reason that this connector was not included in the dedicated connectors' schematic is that the connection with the PHY needed to have some specific features for the Ethernet to work properly. [2]

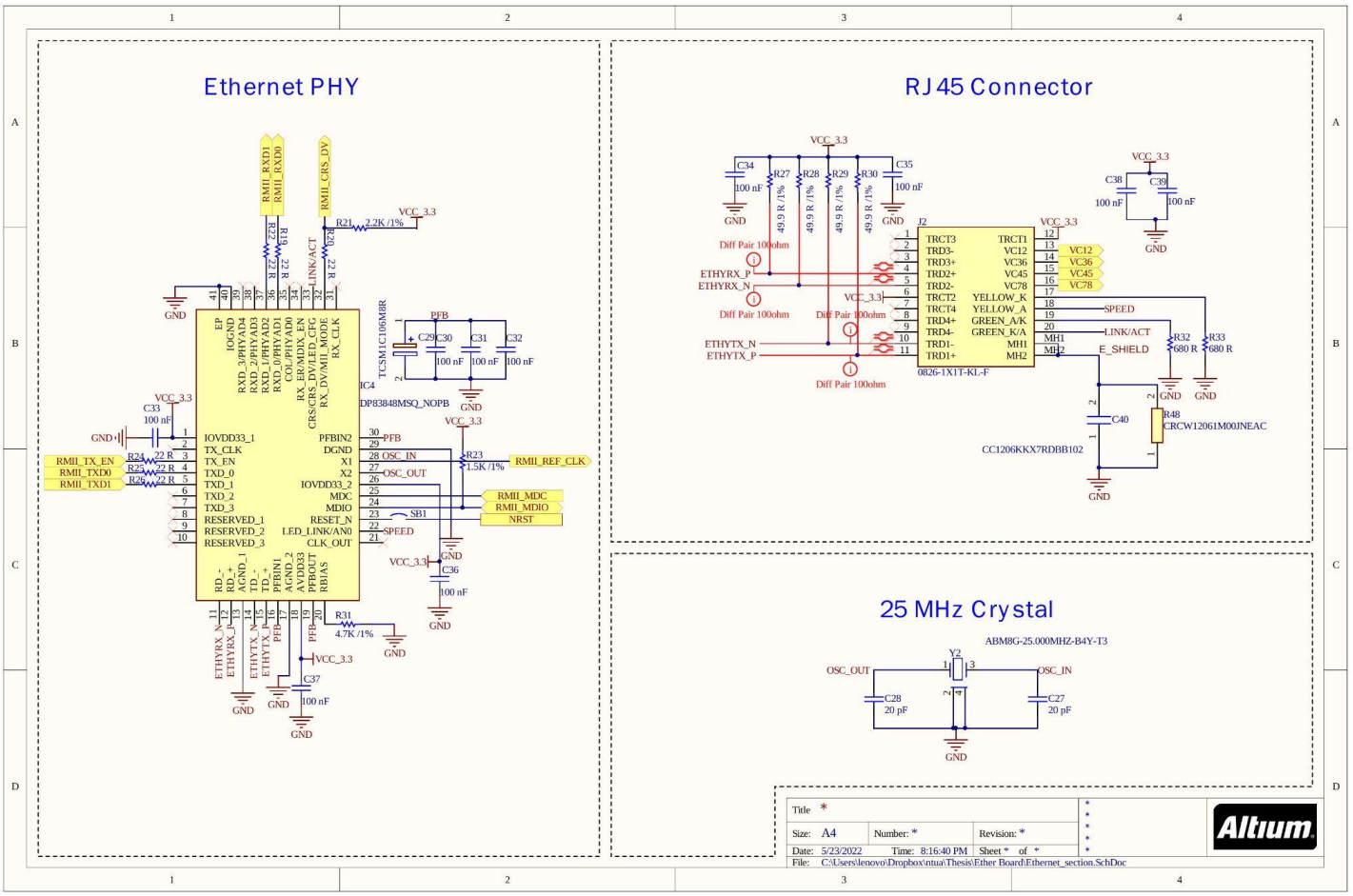


Figure 7.

The ports that connect to the PHY are routed to the MCU's RMII and then internally connected with the MAC peripheral. This set of 9 signals is responsible for carrying information between the MCU and the PHY. Information such as transmit and receive buffers, status registers, and potential bus and packet errors.

For the PHY to work properly and be able to produce the differentially modulated ethernet signals at a maximum of 100 Mbps or 100BASE-T as it is often called, a 50MHz clock is

needed. This can be seen in the bottom right corner of the schematic, with its appropriate load capacitors, whose values are calculated exactly as the ones presented in paragraph 2.2.3.

Once more we can see all the decoupling capacitors that will later be placed near the PHY, as well as some complementary passive components that are needed and are described in the PHY's reference design section of the datasheet. For example, R31 is a 4.87k resistor that is placed outside the PHY but serves as the bias resistor for the internal voltage regulator.

The ethernet connector, also known as RJ45, is responsible for the connectivity of the PHY with the outside world. This means that it has to be able to withstand ESD events while at the same time isolating the system from other unwanted electrical hazards. For that reason, the connector incorporates a set of one-to-one ration transformers that isolate the ethernet differential signals without distorting them. These transformers are also responsible for the separation of the high-frequency data signals from the biased high-power DC signals used for the PoE.

Network cables, such as Cat 5e and Cat 6, comprise eight wires arranged as four twisted pairs. In 10 and 100BASE-T Ethernet, two of these pairs are used for sending information (pairs 1-2 and 3-6), and these are known as the data pairs. The other two pairs (4-5, 7-8) are unused and are referred to as the spare pairs (Gigabit Ethernet uses all four pairs). [3]

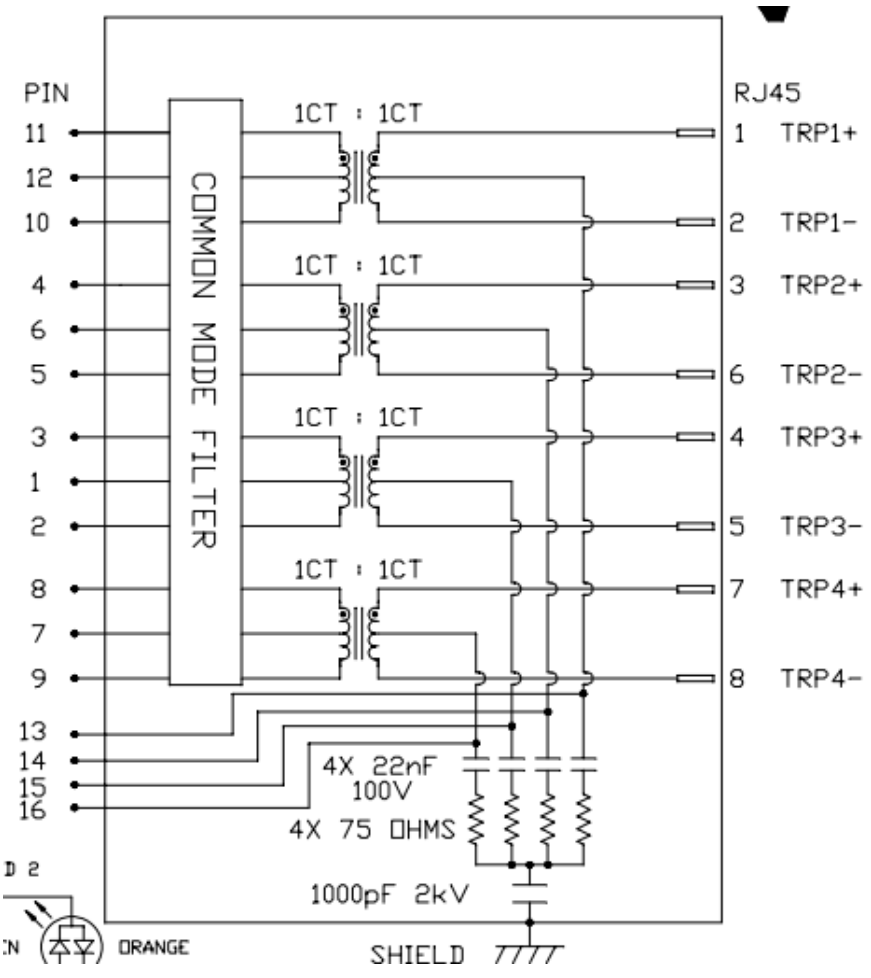


Figure 8.

Because electrical currents flow in a loop, two conductors are required to deliver power over a cable. PoE treats each pair as a single conductor and can use either the two data pairs or the two spare pairs to carry electrical current.

Power over Ethernet is injected onto the cable at a voltage between 44 and 57 volts DC. This relatively high voltage allows efficient power transfer along the cable, while still being low enough to be regarded as safe.

Figure 8 illustrates the internals of such a connector capable of carrying the ethernet data, while at the same time being able to handle the DC-biased voltages used for power transmission. This internal circuitry is often referred to as Magnetics. [4]

The most interesting part of the Ethernet section schematic, however, is the description of the ethernet lines from the PHY to the RJ45 connector. It is well known that ethernet lines are differential and thus, must have a controlled impedance of 100 Ohms between each terminating point on the PCB. This means that each pair (TX and RX) has to have its two conductors carefully sized and spaced on the PCB, in order to achieve the exact impedance while in the MHz region.

Altium offers the capability to mark certain signals as differential (and with specific impedance) and create a class of nets that will carry these characteristics later on in the design process. That is exactly what those red transmission lines indicate. By adding the attributes of differential lines and controlled impedance, Altium adds these nets to the aforementioned class, ensuring that these characteristics will be respected.

2.2.5 Interfaces Schematic

In Figure 9 we can see the various peripherals that will be used for the sensors' voltage measurement. More specifically, the isolated digital input chip is located on the top left corner and the external ADC on the top right corner. Below them we can see the 2 CAN transceivers as well as the RS232 and RS485 interfaces.

For most peripherals, critical information is included in the datasheet. Complementary components, decoupling capacitors, and often a recommended PCB layout will be given by the manufacturer. In the case of the aforementioned, following the schematic recommendations is most of the times the best decision to ensure the proper functionality of the chip.

In this schematic, we can see the mating ports of the ones present in the MCU and connectors' schematics, that describe the connectivity between the MCU and peripherals, as well as the mapping of the interfaces with the various connectors.

All IC supplies are properly decoupled with a 100nf ceramic capacitor, and when needed (like in the case of the RS232) tantalum electrolytic capacitors are placed as per the datasheet. In the case of RS485, special care has been taken to ensure protection from electrical hazard, by placing an ESD protection IC at the input side.

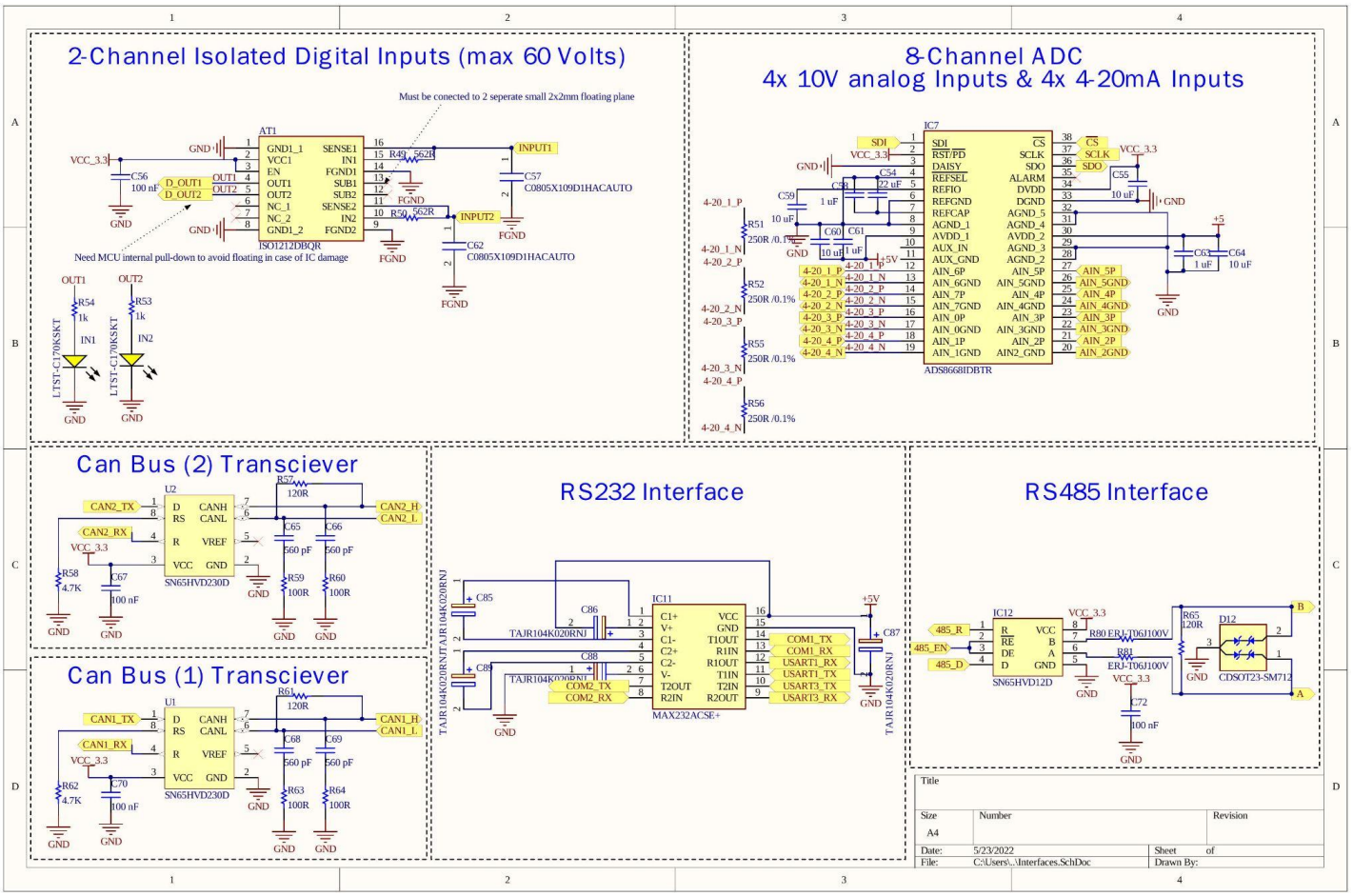


Figure 9.

The two CAN transceivers are connected with the MCUs internal peripherals with the help of dedicated TX and RX lines. These lines carry the transmit and receive information from the CAN channel inside the MCUs peripherals where the protocol implementation takes place. Arbitration, message filtering, header detection and CRC are being performed by dedicated HW that communicates internally with the MCU core. Lastly, received and transmitted bytes are being handled in TX and RX mailboxes that are available to the MCU at all times.

During a transmission, a signal propagates through the conductors and it reaches all the devices that are placed in stubs (conductors that branch out of the main line), but it also reaches the end of the bus lines. If the ends are not terminated, the signal can reflect and interfere with the next data signal coming down the line. Since CAN is a high speed data bus, these constant collisions would be catastrophic. For that reason, two 120 ohm resistors are placed in the far ends of the main bus conductor as per the standard [5], in order to match the impedance of the (often) long cables.

For that reason, terminating resistors have been placed on the PCB for both CAN transceivers, to allow the system to act as an end node, in a 2 device bus, which is fairly

common in industrial applications, as it allows for a high speed communication of up to 40 meters.
The RS232 and RS485 peripherals are connected to the MCUs generic UART internal peripherals, allowing for configurable baud rate and protocol implementation.

The external ADC has several capacitors attached to its analog and digital supply pins, as the datasheet recommends. Figure 10 demonstrates the recommended layout of the ADS8668, as well as the appropriate capacitor values and placement on the PCB.

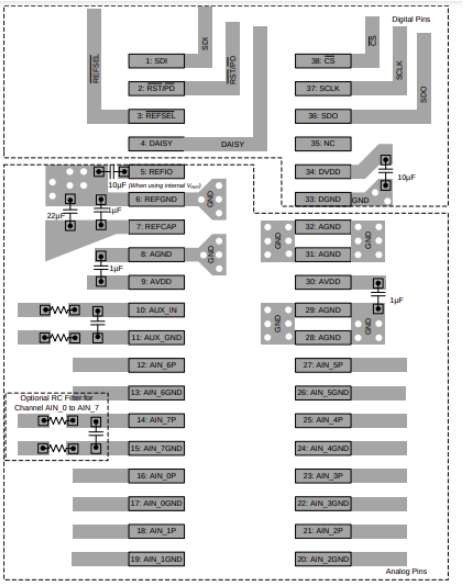


Figure 10.

The communication of the external ADC and the MCU is done through an SPI channel. The MCU is the master of the channel, which leaves the ADS8668 as the slave. Essentially, the MCU will send commands over the SPI and the ADC will respond. This communication uses 4 wires (hence 4-wire interface) and a GND connection. Since SPI can have one master but multiple slave devices, a chip select (CS) signal is dedicated so that the master can enable one device at a time, without the need of separate connections. For timing reasons, and in order to achieve selectable speeds, a signal called SCLK (serial clock) that is generated from the master is responsible for synchronizing the data flow at a precise frequency. Lastly, two signals called MISO (master in slave out) and MOSI (master out slave in) are the ones actually carying the communication data. The reason SPI has 2 data lines is to allow for a full duplex communication (much like ethernet). This means that the master and the slave can exchange data at the same time, using the SCLK as the timing reference on when to transmit or when to receive a bit. As mentioned in previous chapters, SPI is one of the most common on-board communication methods between ICs, because it allows fast transfer speeds with just 4 wires, and can easily be expanded with the use of an extra signal per device.

Figure 11 shows the wiring diagram of the ADS8668 with the MCU.

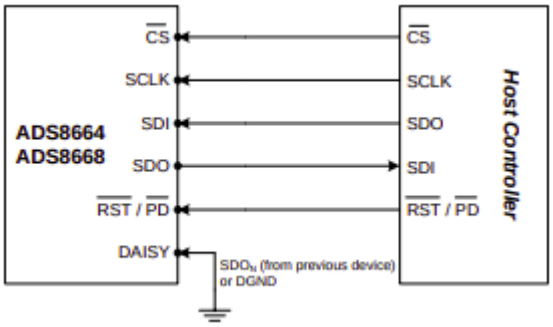


Figure 11.

One of the most interesting parts of this schematic however, is the part where we measure the 4-20mA signals. Traditionally, one could find a dedicated 4-20mA transceiver in the market and just connect the current loop directly at its inputs. However, there is a simple and cost-effective solution that allows us to achieve the same, while at the same time maintaining most of the accuracy needed for these kinds of measurements.

The idea [6] is to place an external resistor, through which the current of the 4-20mA loop will flow. Connecting the ends of the resistor at an input of the ADC will result in the direct measurement of the voltage drop across the resistor. In order to achieve an accurate enough measurement, a precision resistor is used, which ensures that the relation between the nominal and the actual resistance is as close as possible, thus minimizing the conversion error. By using this method, we don't need an extra IC to measure the 4-20mA compatible sensors, but with some precision resistors and the external ADC that is already placed on the PCB, we achieve an accurate enough measurement. Yet another step to further reduce our BOM’s cost.

2.2.6 Power Stage Schematic

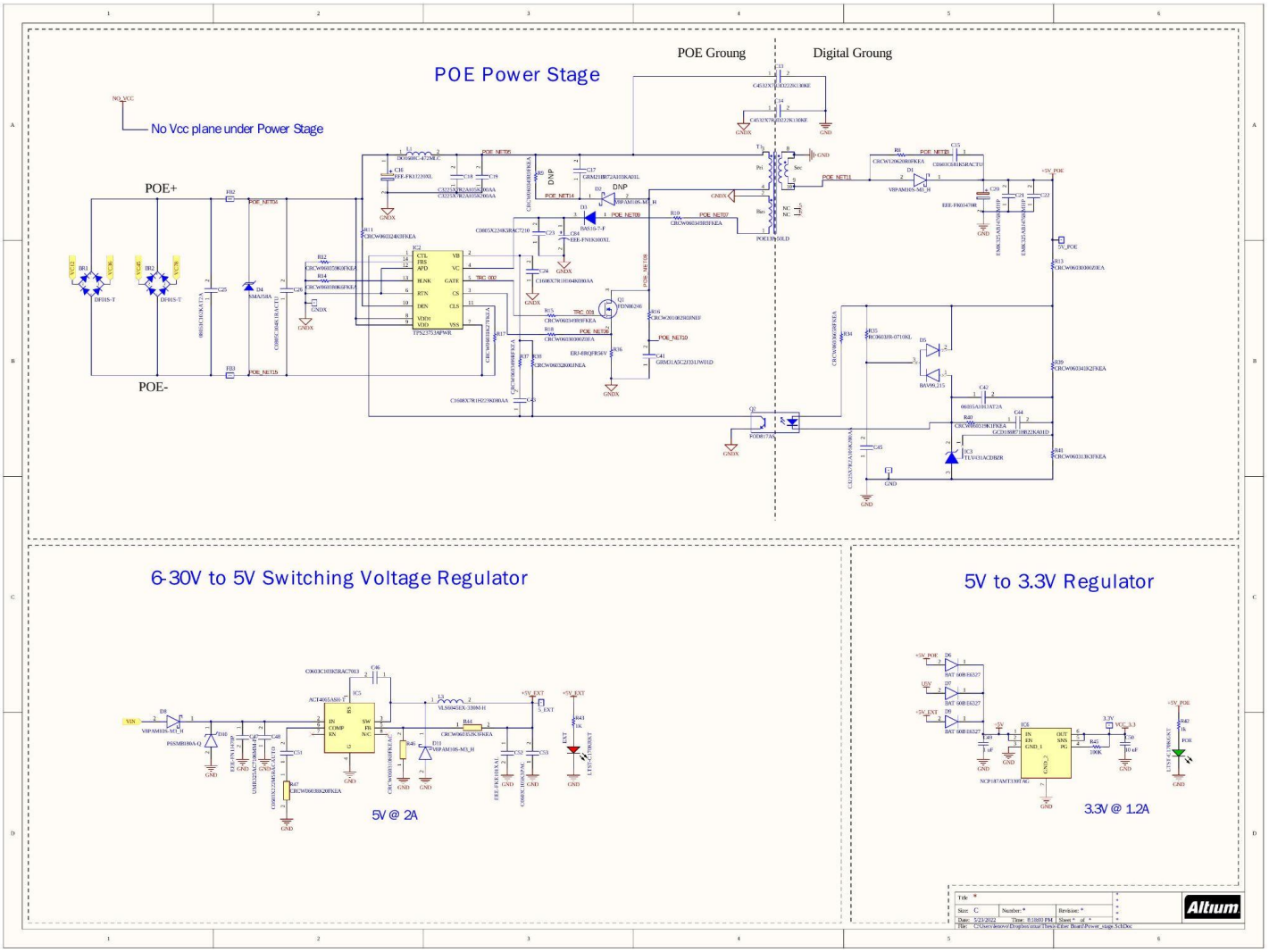


Figure 12.

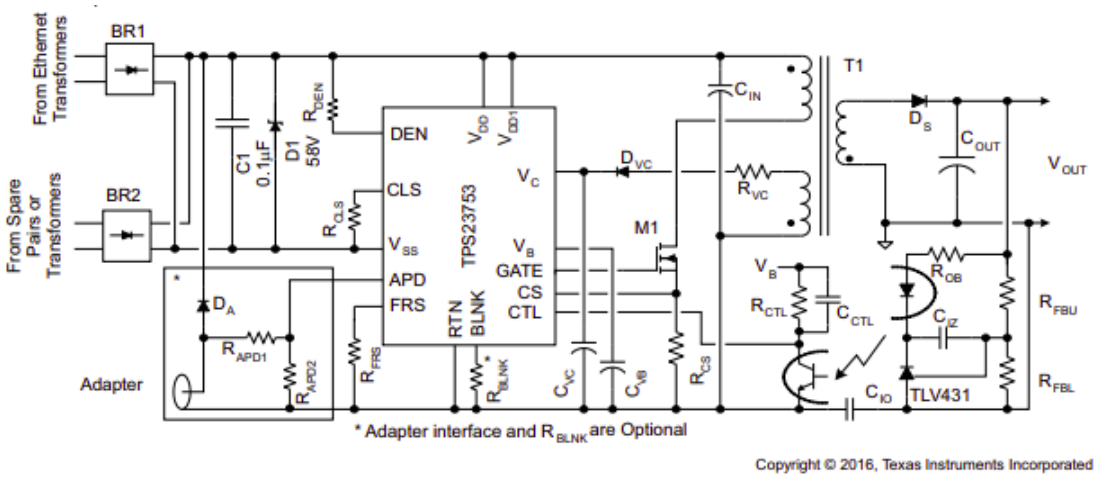
Figure 12 shows the Power stage of the System. As mentioned in paragraph 2.1.2, the power plan comprises several different methods of powering the board, with the main being the PoE stage. From the Ethernet section schematic, we saw that the magnetics of the RJ45 connector are responsible for separating the high-frequency data and the DC-biased signals, providing approximately 48 volts between 2 of the 4 pairs (depending on the Power Sourcing Equipment). These 48 volts however are not purely DC, as they contain distortion from the transmission lines of the ethernet.

The target is to acquire a usable 5 volts supply, that can then be used to supply the various onboard chips and components. For that reason, a dedicated PoE DC-DC converter was selected, in order to maximize the efficiency of the conversion, while maintaining complete isolation between the system and the outside world. TPS23753A is an IEEE 802.3 PoE Interface and Converter Controller with Enhanced ESD Immunity that supports a number of input-voltage ORing options including highest voltage, external adapter preference, and PoE preference.

The schematics and HW design of the DC-DC PoE stage was highly influenced by TIMD-TM4C129POE, which is an open source reference design around a PoE enabled TM4C129ENCPDT MCU application by Texas Instruments.

As a first step, the 4 high power signals are passed through two full-bridge rectifiers, eliminating all AC components. The first stage of smoothing capacitor is then applied to ensure minimum voltage ripples in combination with two ferrite beads and a Zener diode that is used to prevent overvoltages. Another smoothing capacitor is used, and then, the DC smoothed voltage enters the TPS23753A.

Following the datasheet's basic implementation guidelines shown in Figure 13, we can see a number of passive components, diodes, switching elements, and complementary components.



Basic TPS23753A Implementation

Figure 13.

Since the circuit is very close to the one of a normal DC-DC converter, we are not going to explain in detail the theory of operation, as this would go out of the scope of this thesis. However, since the use case involves some rather interesting components, we are going to highlight their purpose and function.

The higher amplitude DC voltage enters the IC, which calculates the correct PWM duty cycle that needs to be applied on the gate of a switching element (in this case a MOSFET). The converter always takes feedback from the output to adjust the duty cycle of the pulsed signal, in order to take into account supply needs, voltage drops, and short circuit events.

Some of the differences from a normal DC-DC converter are that in this design, isolation is provided between the higher voltage and the 5V with the use of a transformer. It can be seen in both the basic implementation and the schematic. Moreover, the feedback is not connected directly to the converter, but through an optocoupler, ensuring complete isolation and protection from electrical hazards.

Another interesting part is the Adjustable Precision Shunt Regulator (TLV431) that is present in the feedback subcircuit of the converter. Its main purpose is to create a stable voltage in

its cathode pin, that is then connected to the cathode side of the feedback optocoupler. This is achieved by the two resistors R39 (41.2k) and R41 (13.3k) that form a voltage divider of the output voltage of the converter. The relation between the output voltage of the DC-DC and the output of the voltage divider is

$$V_{divider} = \frac{R41}{R41 + R39} * V_{DCDC}$$

Which gives as an output of $V_{divider} = 0.244036697 * V_{DCDC}$

Since the expected output of the DC-DC is close to 5V and the internal V_{ref} voltage of TLV431 is set to 1.24V, we can see that minimum fluctuations above and beyond the voltage set by the resistor network, will influence the behavior of the component. In detail, the cathode voltage will be:
Equal to V_{DCDC} when

$$V_{DCDC} * 0.244036697 < 1.24V$$

and 0V when

$$V_{DCDC} * 0.244036697 \geq 1.24V$$

This means that when the output voltage of the DC-DC exceeds 5.081203013V the cathode of the TLV431 and also the cathode of the optocoupler will be 0V. That means that the optocoupler will be drawing current and thus turning on the feedback pin in the DC-DC, indicating that the voltage in the output has reached the upper set limit. The opposite happens when the output voltage of the DC-DC drops below 5.08 volts. In this case, the cathode of both the TLV431 and the optocoupler stay high at V_{DCDC} , thus not allowing the optocoupler to draw current.

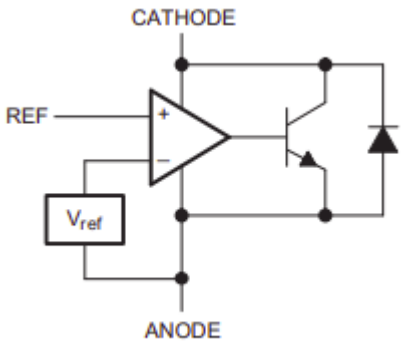


Figure 14.

Viewing the block diagram of TLV431 in Figure 14, we can see that it acts as an adjustable comparator, creating a pulse that is high when the voltage is above the desired limit and low when it is below. This component is the main source of the feedback of the DC-DC, ensuring that the output voltage is stable regardless of the imperfections of the resistors or the temperature changes, as it features a very small temperature drift, making it the perfect component for the DC-DC feedback loop.

Except for the PoE power stage, a separate DC-DC converter was placed in the design, in order to give the option to run the system with normal ethernet connectivity while still bein able to power the board from an external power supply. The part was chosen carefully after considering the physical size, the price and the input voltage range. Since the system targets industrial environments, in which several different voltages exist, the range of 6-30 V covers most of them with the 12V and 24V being the most common. As far as the output current is concerned, the converter can output up to 2A at 5V, which is more than enough considering the system's needs. The topology of the complementary components is the standard that every general purpose DC-DC converter has. The 2 diodes on the input side serve as reverse polarity and overvoltage protection (clamping action), while the input capacitors ensure a smooth input voltage to the converter. The output is fed through a 330uH inductor to eliminate voltage ripple and create a stable output voltage in combination with the output diode which serves as clamping mechanism to restrict the output voltage from exceeding the rated level. Lastly, the output capacitors serve as filtering elements to eliminate as much of the high frequency components that are induced to the output voltage by the switching element as possible. At the same time, they provide an instant charge in the event of a higher current demand that causes a voltage drop.

However, since most of the chips need a 3.3V power supply that is very stable with minimum voltage fluctuation, an extra LDO was placed in the design to make sure these requirements were met and all the components of the system are supplied by a stable 3.3V and not by the noisy output of a DC-DC converter. The input of the LDO is connected to all 3 different 5V sources of the system with the help of diodes that prevent the current from flowing from one 5V voltage source to another, acting like a 3 way OR gate. In this way, current can only flow from the 5V sources to the input of the LDO, ensuring that regardless of the power source of the system, the 3.3V regulator will always be supplied properly. Again, we can see the smoothing capacitors connected on the input and output of the LDO, making sure that the high frequency noise is filtered and the supply is clean and stable.

2.2.7 ST-Link Schematic

Figure 15 shows the schematic of the onboard debugger called ST-Link. In essence it is another ST32 MCU (STM32F103CBT6) that is loaded with a special firmware, capable of interfacing with the SWD (Serial Wire Debug) interface of the main MCU. It allows the STM32CubeIDE to run the desired program on the main MCU (or target) in full debug mode, giving the options for step execution, breakpoint setting, in depth memory inspection, and many more features. Moreover, it creates a Virtual COM port through which the target MCU can pass strings to the host PC using one of the target's UART peripherals. In this way, we can print status messages in the host computer and read them using a common Serial terminal program (like Putty), which makes the FW development phase much faster.

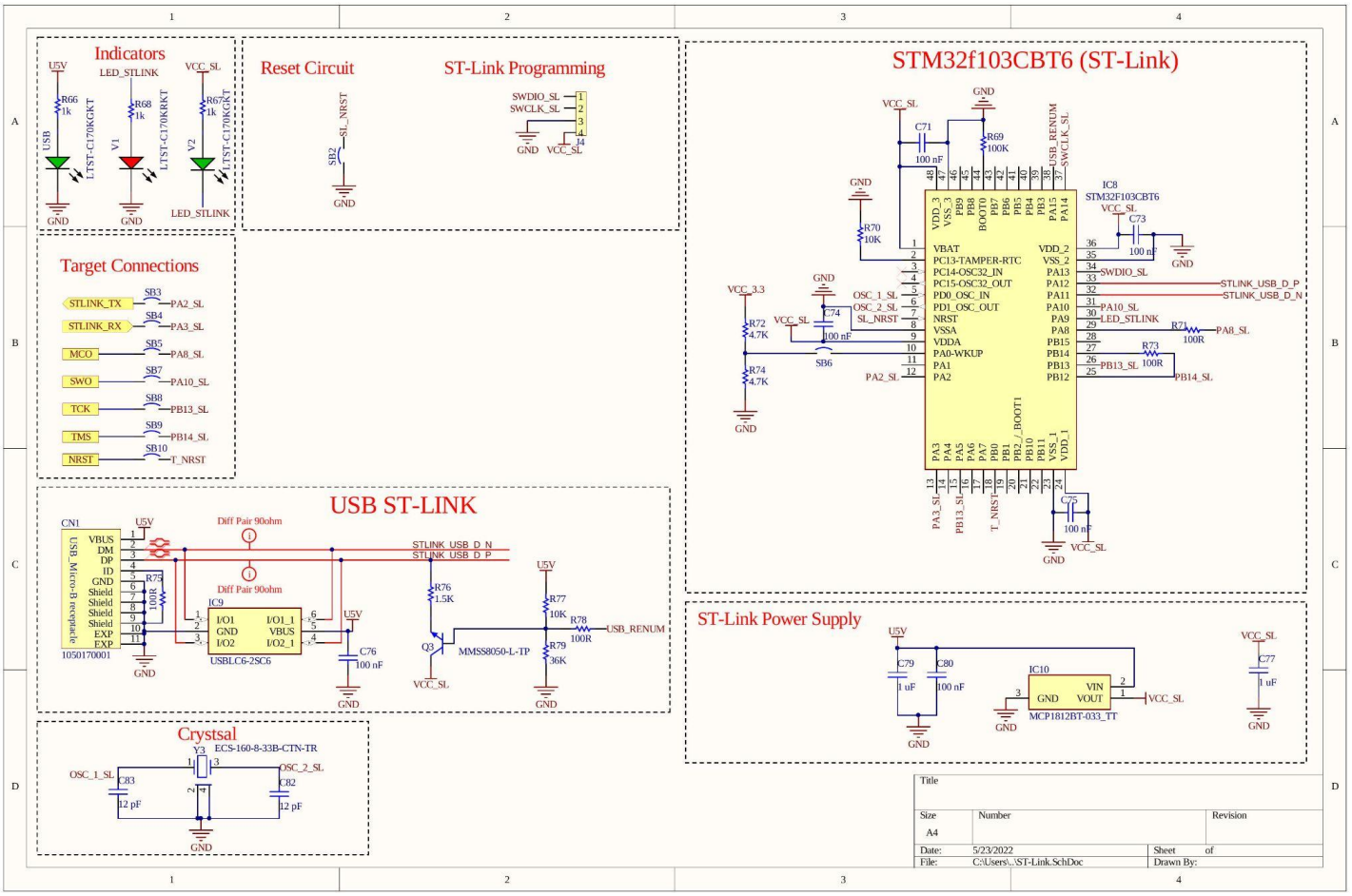


Figure 15.

In order for the ST-Link MCU to be able to interface with the host computer, a dedicated micro USB connector is placed in the design. It connects directly to the MCUs pins that are internally routed to the chip's USB 2.0 peripheral. As per the USB standard, the positive and negative lines need to have a specific impedance of 90 Ohms and a matched length, hence the appropriate Altium attributes were placed .Connected to the positive line of the USB is also a NPN transistor whose base is controlled by the ST-Link MCU. This feature is used by the special FW to initiate the detection phase of the USB protocol and be recognized by the

host PC. Other than that, a dedicated ESD protection chip is placed near the micro USB connector to make sure that no electrical hazards be people touching the connector while plugging/unplugging can damage the rest of the system.

Exactly like the target MCU that we say in paragraph 2.2.3, decoupling capacitors are placed near each one of the MCUs supply pins, ensuring a stable with minimum noise voltage. Once more, a crystal oscillator is placed and connected with the MCU alongside its load capacitors, to synchronize the internal clock structures of the chip.

Again, status LEDs are placed to indicate the various modes of the debugger, and notify the user/programmer of the ST-Link's status.

An interesting aspect is that the signals connecting the ST-Link MCU and the target MCU are interrupted by 0 Ohm resistors (practically short circuit) each. This technique is frequently used when we want to be able to isolate the two ends of one signal, without having to interfere a lot with the PCB, or place bulky switches or jumper pins (which are other techniques to achieve the same goal). In this way, if we have a HW issue, or we want to isolate or probe certain signals, desoldering a single resistor will do the trick. The cost of the resistors is also significantly lower than that of switches or header pins, which is another plus.

Lastly, we can observe a separate LDO with its complementary capacitors, which is responsible for powering only the ST-Link MCU. This is done in order to be able to power the ST-Link only when the micro USB is connected, since only then we truly need the MCU to be powered, for debugging and Virtual COM port functionality. When the system is powered by PoE or the external DC-DC solution, as we saw in paragraph 2.2.6, the ST-Link MCU is not powered to reduce the consumption. However, connecting the micro USB allows the system to be powered normally, which is very helpful when you only need to develop new FW and debug without having the need for any other external supply.

3 Implementation

In this chapter, the methods of the actual HW and FW design will be presented, with special attention to interesting points and design decisions that affected the end result. The full cycle of the design, ordering and assembly process will be presented, as well as some of the initial HW testing and the methods and tools used for the FW development.

It is very important to mention that the methods and techniques presented below do not follow the industry standard, but rather a kind of an RnD perspective. The reason is that since we are developing prototypes, the number of parts is very small compared to an actual product, and the design has not yet been tested, which automatically makes the system not ready for the market. Hence, some well known methods, services and procedures were not considered, since they target a much larger number of devices and the production cost would rise dramatically in our case.

3.1 Hardware

With the schematics of the system well defined, it was time to design the actual HW. For that reason, a separate file, called PcbDoc, was created inside Altium. However, before transferring the schematic design to the physical world, a last ERC was executed to make sure no mistakes exist in the schematics.

3.1.1 PCB Design

After that, all the components and their relation to each other were transferred to the PCB document. Each component in the schematic was associated with a footprint, a pattern of copper and non conductive material to allow the physical part to be soldered (glued) on the board. Since each component may have more than one footprint available from the manufacturer, the selection was based on the available space on the board, unit price as well as the availability on the market (chip shortage was at its peak during the period of the design). The components were then positioned in a way that the connections were as short as possible, and the interferences between different connections as little as possible. Then, the actual copper tracks were designed and extra features, such as text and mounting holes were added.

Figure 16 shows the finished PCB design with all the appropriate features and details in order for the system to function properly and the user to be able to identify the various components. The text however is hidden in the figure below in order to make other details more easily visible and understandable.

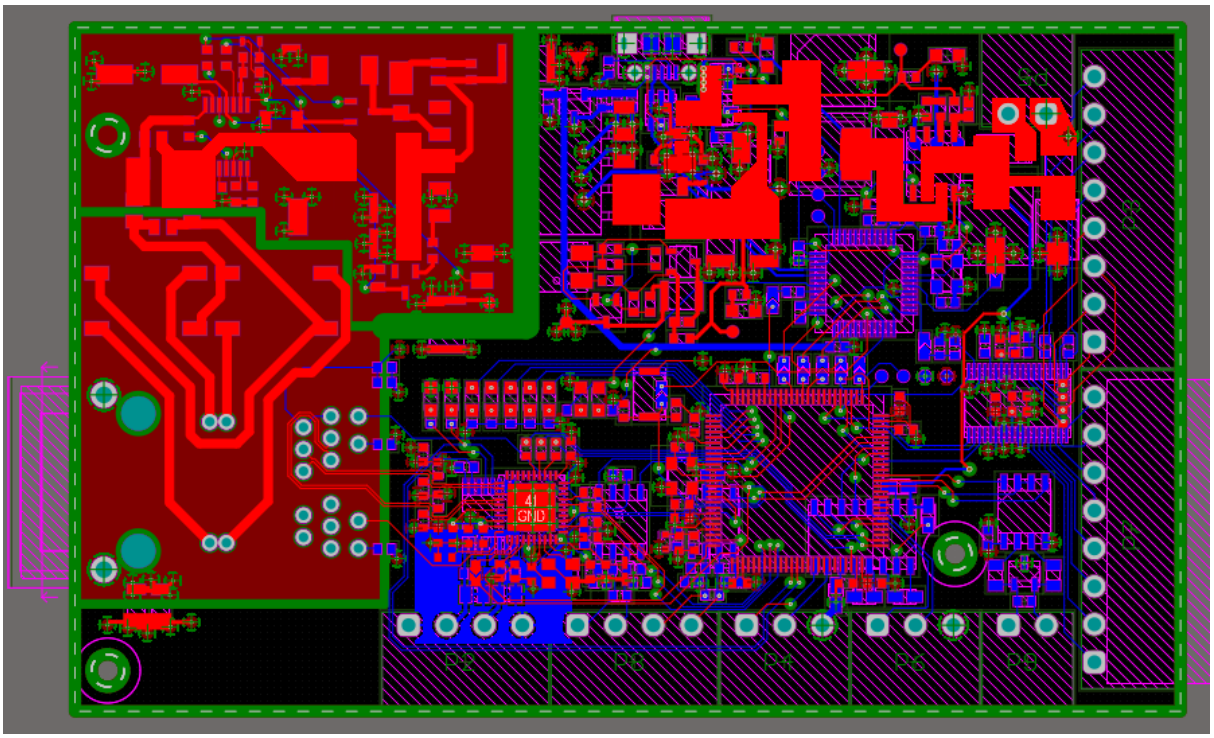


Figure 16.

To better explain the decisions that led to the design of Figure 16, the following design points will be presented:

- Layer stack up
- Design Rules
- Overall Layout decisions
- Special Tracks and Techniques
- Miscellaneous Design Features

Layer stack up

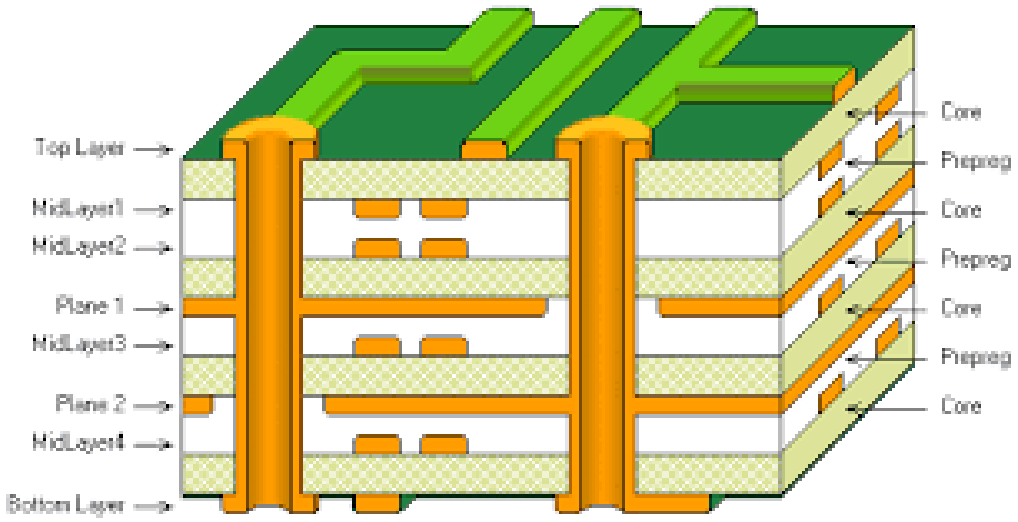


Figure 17.

Layer stack is defined as the sequence of materials (and their properties) that are placed one on top of the other to form the board. The base material, called core, is usually made from a special glass cloth called FR4. On either side of the core material, rests a layer of prepreg epoxy material that is used to attach the copper layers to the core. In the case of a simple two layer PCB, a special material that protects the bare copper by preventing oxidation and mechanical damage, called solder mask, is placed on either side of the copper layers. Lastly, a layer of ink, called silkscreen, is printed on top of the solder mask, and it is used to identify components, indicate special features, and generally mark the PCB. In higher layer count PCB, like the one presented in Figure 16, there are more than one cores that are placed between the copper layers.

The first decision when designing a PCB, is the desired layer count. This can be affected by a number of factors. Usually, the PCB size and the number of components are proportional. However, there are occasions where the PCB has to be small, in order to minimize the weight, the dimensions, or even the manufacturing cost. Another factor is the number and the complexity of signals that need to be routed between the components, which sometimes interferes with the power distribution and/or the actual parts that are placed on the top and bottom layer of the board. For that reason, using higher layer PCBs, can help the designer route the connections in more than one layer with the use of small copper plated holes that are called vias. Vias can connect signals between different layers, giving great freedom to the placement and the path of the copper conductors and the parts.

Another aspect is the characteristics of the materials. More specifically, the electrical properties of the core and prepreg that is placed between the copper layers, as well as their physical dimensions. It is well known that the energy of a signal transmitted inside a PCB is not transferred through the conductors, but rather through the dielectric material in the form of an electric field. The electric field in turn helps the electrons of the conductor material to move, which then creates the electric current and as a result, the magnetic field around the conductor. It is clear that the properties of these materials significantly affect the performance and efficiency of data transmission inside a PCB, which is crucial for the correct operation of all the ICs, especially in higher frequencies.

In our case, since we want to minimize production cost, and we also have some special characteristics that we want to achieve, four layers is the best compromise between production cost and ease of design.

Looking at some of the most known and affordable manufacturers of PCBs (located in China), we can see that the materials and services offered are standardized and identical in most cases, which makes the decision of selecting a supplier come down to production cost, shipping time and overall quality.

Each manufacturer has specific capabilities, or rather some limitations that are usually stated in their website, or by asking them directly with the form of a quote. In our case, the capabilities of the manufacturer were clearly stated in the website, giving information about tolerances, materials, production techniques, and overall specifications.

In the case of the 4 layer PCB, the available options were:
a)

Layer	Material Type	Thickness	
Top Layer1	Copper	0.035 mm	
Prepreg	7628*1	0.2 mm	
Inner Layer2	Copper	0.0175 mm	1.1mm (with copper core)
Core	Core	1.065 mm	
Inner Layer3	Copper	0.0175 mm	
Prepreg	7628*1	0.2 mm	
Bottom Layer4	Copper	0.035 mm	
0.2mm (7.87 mil) is nominal thickness of 7628 prepreg. Use 7.1 mil as the thickness when the controlled impedance tracks are on top/bottom, use 8.1 mil when tracks are inside.			

Figure 18.

b)

Layer	Material Type	Thickness	
Top Layer1	Copper	0.035 mm	
Prepreg	2313*1	0.1 mm	
Inner Layer2	Copper	0.0175 mm	1.3 mm (with copper core)
Core	Core	1.265 mm	
Inner Layer3	Copper	0.0175 mm	
Prepreg	2313*1	0.1 mm	
Bottom Layer4	Copper	0.035 mm	
0.1mm (3.94 mil) is nominal thickness of 7628 prepreg. Use 3.5 mil as the thickness when the controlled impedance tracks are on top/bottom, use 4.5 mil when tracks are inside.			

Figure 19.

With the main difference being the Prepreg material type and thickness. Option b was selected, since it minimizes the distance between copper layer 1 and 2 which is the most critical distance for the special tracks in our design (more details in Special Tracks and Techniques subparagraph).

In Altium Designer, we adopt our layer stack with the one selected from the manufacturer, in order to be able to represent the board as close to reality as possible.

#	Name	Material	Type	Thickness	Dk	Df	Weight
	Top Overlay		Overlay				
	Top Solder	SM-001	Solder Mask	1mil	4	0.03	
1	Top Layer	CF-004	Signal	1.378mil			1oz
	Dielectric 1	PP-017	Prepreg	7.874mil	4.6	0.02	
2	GND	CF-004	Plane	0.689mil			1/2oz
	Core	Core-039	Core	41.929mil	4.8	0.02	
3	PWR	CF-004	Plane	0.689mil			1/2oz
	Dielectric 2	PP-017	Prepreg	7.874mil	4.6	0.02	
4	Bottom Layer	CF-004	Signal	1.378mil			1oz
	Bottom Solder	SM-001	Solder Mask	1mil	4	0.03	
	Bottom Overlay		Overlay				

Figure 20.

Design Rules

As mentioned before, each PCB manufacturer has specific limitations when it comes to tolerances, distances and other design metrics. In our case, all the important values are clearly stated in the manufacturer's website, leaving no room for uncertainty during the design phase.

Altium, like most EDA software, provides the live and/or asynchronous rule check. That means that every feature designed, as well as the combination of different features, is being compared to a set of design rules that are specifically indicated by the designer. It is clear that in order to produce a manufacturable PCB, all the features need to be inside the manufacturer's specification, otherwise they can reject the design and ask for corrections, or even worse, produce it with defects.

To avoid mistakes and to optimize the ordering procedure, a strict set of design rules need to be defined before the HW design process can begin. Once more we extract all the information needed from the manufacturer's website by visiting the "Capabilities" section.

Some of the most important information we gather are:

PCB Specifications	
Layer count	1,2,4,6 layers
Controlled Impedance	4/6 layer, default layer stack-up
Material	FR4
Dielectric constant	4.5 (double-sided PCB)
Max. Dimensions	400x500mm
Dimension Tolerance	±0.2mm
Thickness Tolerance	± 10%
Finished Outer Layer Copper	1 oz/2 oz (35um/70um)
Finished Inner Layer Copper	0.5 oz (17.5um)
Drill/Hole Size	
Drill Hole Size	0.20mm- 6.30mm
Drill Hole Size Tolerance	+0.13/-0.08mm
Blind/Buried Vias	Not supported
Min. Via hole size	0.2mm
Min. Via diameter	0.4mm
PTH hole Size	0.20mm - 6.35mm
Pad Size	0.70mm- 6.35mm
Min. Non-plated holes	0.50mm
Min. Plated Slots	0.65mm

Min. Castellated Holes	0.60mm
Hole size Tolerance (Plated)	+0.13mm/-0.08mm
Minimum clearance	
Hole to hole clearance(Different nets)	0.5mm
Via to Via clearance(Same nets)	0.254mm
Pad to Pad without hole	0.127mm
Pad to Pad with hole	0.5mm
Via to Track	0.254mm
PTH to Track	0.33mm
NPTH to Track	0.254mm
Pad to Track	0.2mm
Minimum trace width and spacing	
1-2 Layers trace width	5mil (0.127mm)
1-2 Layers minimum spacing	5mil (0.127mm)
4-6 Layers trace width	3.5mil (0.09mm)
4-6 Layers minimum spacing	3.5mil (0.09mm)
2oz Copper weight trace width	8mil (0.2mm)
2oz Copper weight min spacing	8mil (0.2mm)
Solder Mask	
Solder mask opening/ expansion	0.05mm
Solder bridge	0.2mm(green)
Solder mask dielectric constant	3.8
Solder mask thickness	10-15um
Legend	
Minimum Line Width	6 mil (0.153mm)
Minimum text height	40 mil (1.0mm)
Character width to height ratio	1:6
Pad To Silkscreen	0.15mm
Board Outlines	
Trace to Outline	0.2mm

Table 6.

Then, we import these specifications to the Altium's dedicated Design Rule section, to ensure conformity and continuous monitoring for potential design mistakes. It is very important to mention that the manufacturer specifies the absolute specification that can be achieved during the manufacturing procedure. In most cases however, there is a substantial price difference when it comes to tighter specifications or some special features. For that reason, an evaluation of the actual needs of the design was made in order to stay within a certain level of the manufacturer's capabilities and further reduce the unit price of a PCB.

Figure 21 illustrates the Design Rule section of Altium Designer.

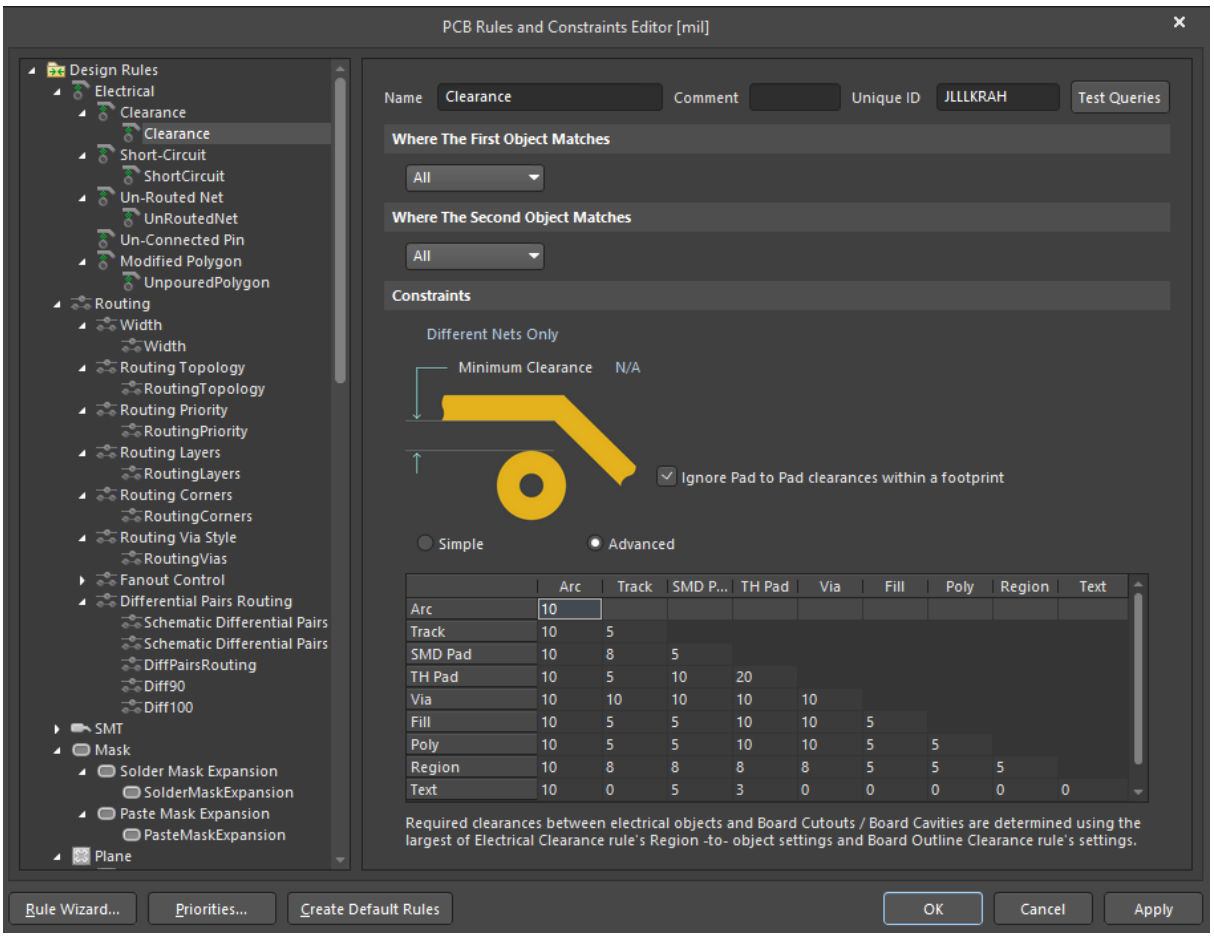


Figure 21.

With the Design Rules set up correctly, after designing our PCB we can run the DRC which is an automatic integrated tool that detects and reports all design features that are out of spec. In this way we ensure that the design files sent to the manufacturer are correct and can actually be manufactured.

Overall layout decisions

Since we are essentially designing a mixed signal PCB that contains both analog and high speed digital signals, certain aspects need to be taken into account. First of all, there has to be a spatial separation between the different stages. That means, the analog signals need to be far enough from high speed digital lines, to avoid coupling and signal integrity issues.

For that reason, it was decided to place the ethernet section (and connector) on one side of the PCB and the analog interface on the other. In this way, the physical distance between the

two would be the maximum possible. The same applies for the PoE DC-DC circuit and other high speed interfaces, such as CAN.

As seen in Figure 22, the different areas are well separated and positioned on the PCB in a way that the possibility of signal coupling from one to another is minimized.

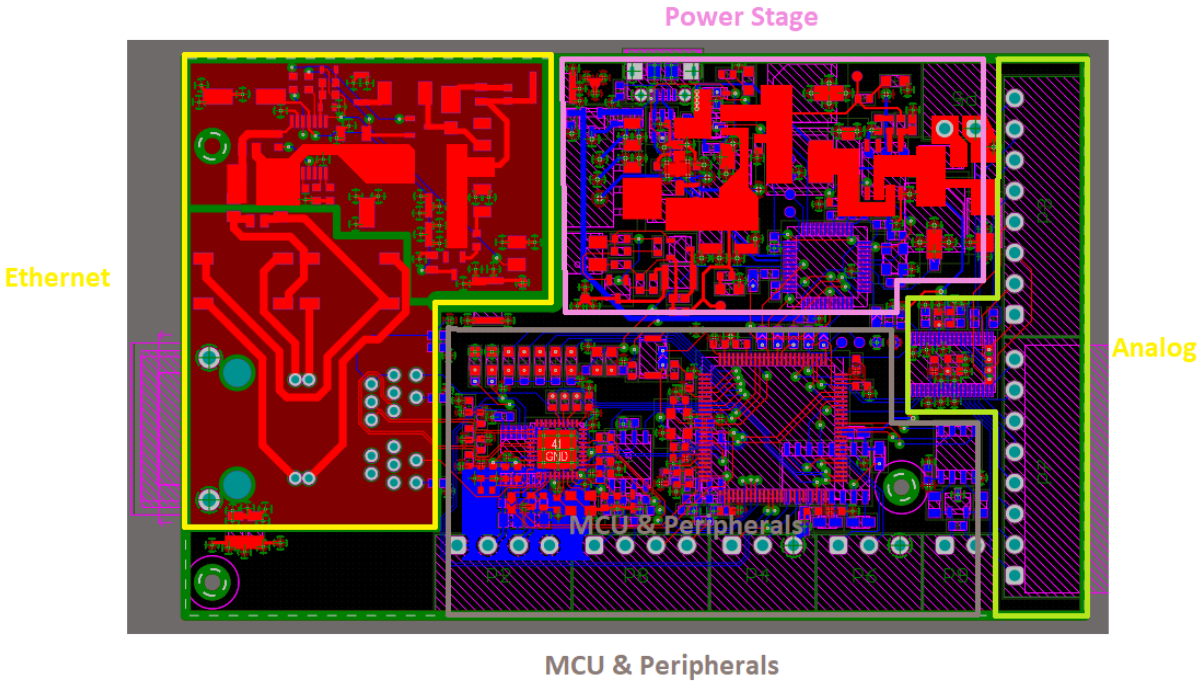


Figure 22.

An extra measure that was taken to eliminate induced noise in sensitive areas on the PCB, was to split the ground planes between the PoE DC-DC region, the RJ45 connector, and the rest of the circuits. This was done to avoid high frequency noise from the DC-DC as well as ESD events on the RJ45 connector from passing to the main system GND. The two isolated GND planes were connected with the main GND plane with the help of High voltage Multilayer Ceramic Capacitors. Since the ADC inputs were isolated and bipolar, there was not any need to split the ADC GND from the rest of the system's main GND. This is because each input is referenced to another pin of the ADC (pair of inputs) and there is no connection to the main GND.

Moreover, to ensure that the Ethernet section functions properly, special care was taken to position the PHY and RJ45 connector as close as possible, in order to minimize the differential pairs' length and in turn possibility to compromise signal integrity. Since the MCU communicates with the PHY through RMII which is a relatively low speed digital interface with error detection, the routing was not very critical, and thus it was done lastly, to allow the more critical positioning of the PHY and its complementary components.

The MCU was placed on the top side of the PCB along with the ethernet PHY and the power regulators and DC-DC converters. These components, when positioned on the top side, define the overall dimensions of the PCB. This is done to avoid the placement of components such as the PoE DC-DC and the ethernet PHY below or above other chips such as interfaces, or even the MCU. In this way, we ensure that with the adequate spacing mentioned above, there will be no interference with the high frequency switching and the other components of the system. Once these "noisy" components were placed on the top

side, the rest of the interfaces and the ADC were positioned mainly under the MCU and the connectors, on the bottom side of the PCB.

Special Tracks and Techniques

As mentioned before, the most critical distance in the layers of the PCB is the one between the top copper and the first inner layer. Since we positioned the ethernet PHY on the top layer of the PCB, its contacts and therefore signals will be connected on the top copper layer. To achieve the controlled impedance stated in the standard, we need to carefully design the differential pairs, both in size and spacing, as well as take into account the distance of the top copper layer and the nearest GND plane at which the signals will be referenced. It is clear that the first inner layer should be tied to GND to minimize the distance to the top layer where the ethernet differential signals are routed, and decrease the impedance to where small adjustments on the ethernet lines can get us to the 100 ohm differential and 50 ohm to the GND each. Another reason to tie the first inner layer to GND and not to, for example another signal layer, is that, in case we do the opposite and rout other lower ar even higher frequency signals below the ethernet lines, since both are referenced to GND, which will subsequently be another layer further down, the 2 signals will get coupled and the risk of having major signal integrity issues rises significantly. For that reason, inner layer one was selected to be GND, inner two, power, and finally, bottom copper layer was selected to be the second signal layer. With this stack up, we ensure that no signals interfere with each other in the z-axis (through the layers) and we only need to pay attention to spacing of signals on the same layer.

As mentioned in the circuit design phase, the two lines of each pair have to be equal in length, to allow each part of the differential signal to arrive and depart at the same time on and from the PHY. This is automatically managed by altium, since we placed the differential pair symbol on the schematic. However, it is still needed to define the track width and the track spacing for the two conductors in each pair.

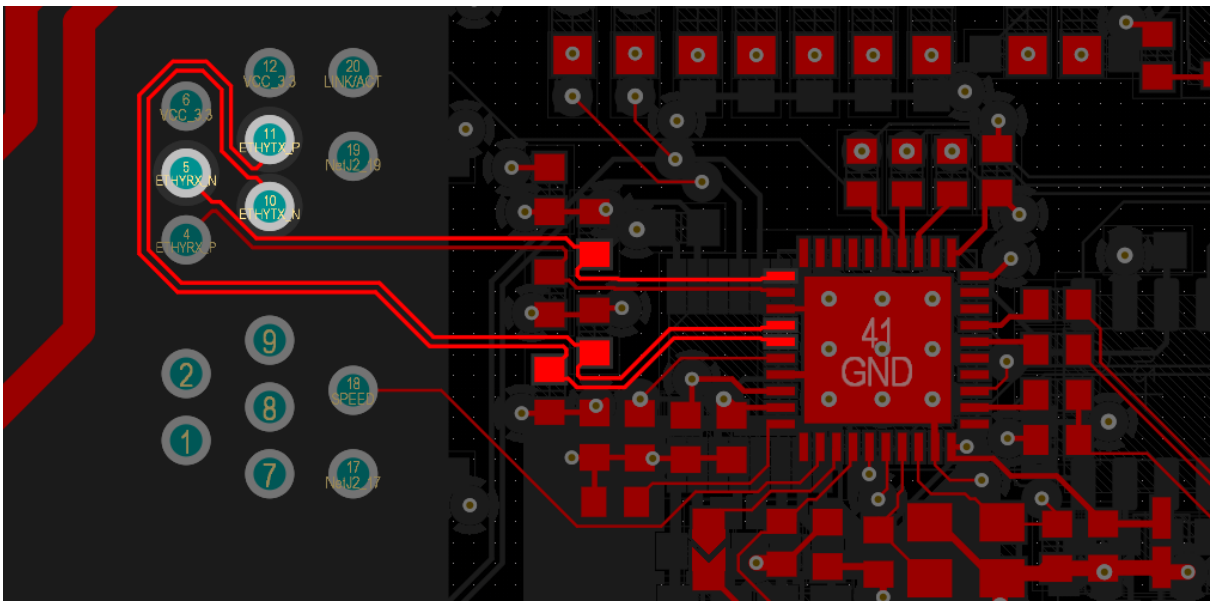


Figure 23.

To calculate these two numbers, we will use a free software called Saturn PCB toolkit. It incorporates various different calculation tools for thermal, current and sizing calculations. We will use the Conductor Impedance calculator, inserting the desired impedance, distance between the closest GND plane and we will get as a result the target conductor width and spacing of the pair.

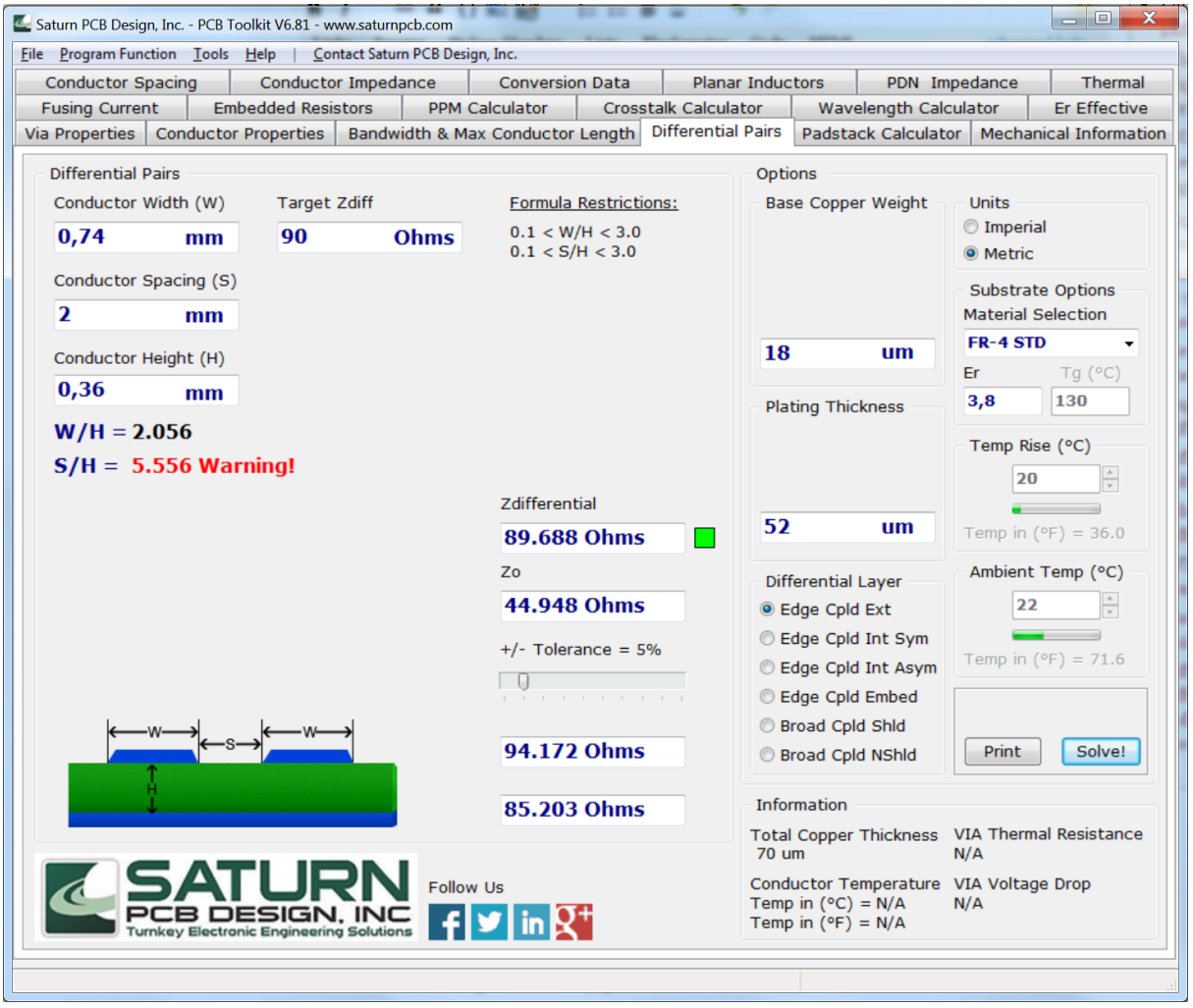


Figure 24.

Miscellaneous Design Features

Some extra features that are worth mentioning are the test points that were placed on various positions on the PCB (top and bottom) to allow for a quick and safe way to probe signals when testing the PCN. Test points are very common and useful when designing a prototype PCB like the one presented in this thesis. Figure 25 presents two kinds of test points, throughole with the attached ring and SMD pads that act as a place to touch with a multimeter or oscilloscope probe. In our case, SMD pads were used to save space and reduce the amount of parts in the BOM, since we only needed to probe the various power supplies.

Figure 25.

Another design aspect that was evaluated is the placement of mounting holes to allow the PCB to be attached in an enclosure. Since the overall size was determined mainly by the interface connectors and the main circuits, there were two different approaches to the positioning of the mounting holes. To place them on the PVB perimeter, which means that we need extra material to facilitate the screws and the holes. Or to find empty spots inside the PCB, preferably one near every corner, and to incorporate them around chips and other components. The second method was selected, in order to maintain the overall PCB size the same while being able to bount the PCB inside an enclosure in the future.

3.1.2 Ordering and Assembly

Once the PCB was designed, it was time to order the board and the various parts, and assemble them. For that reason, Gerber Files were generated through Altium, in order to be uploaded on the manufacturer’s website and request their production. Gerber files essentially comprise the layout of each layer of the board (top copper layer, solder mask, silkscreen) in physical dimensions, and are the files that are imported into the CNC machines that produce PCBs.

Except for the boards, the actual components needed to be ordered, so that the system can function as intended. For that reason, again from Altium, the BOM was exported. BOM is an excel list of all the components that are placed on a PCB, their quantities, part numbers, manufacturers, designators and other information such as description and footprint. In our case, since we are doing the sourcing by ourselves, there wasn't the need to follow a specific format. However, for good practices, a generic format was chosen, consisting of the minimum columns to perform the ordering and the assembly. Figure 26 presents some of the components and the respective attributes mentioned above.

	A	B	C	D	E	F	G	H	I
1	Added	Quantity	Availability	Comment	Quantity	Part Number	Designator	Mouser link	Description
2	ok	100	yes	1 pF 100V 50% (0805) X8R	2	C0805X109D1HA	C57, C62	https://gr.mouser.co	Capacitor
3	ok	100	yes	12 pF 6.3V 10% (0603) X7R	4	C0603C120K9RA	C11, C12, C82, C8	https://gr.mouser.co	Chip Capacitor, 2.2 nF, +/- 10%, 50 V, -55 to 125 degC, 0603 (1608 Metric), RoHS, Tape and Ree
4	ok	100	yes	20 pF 16V 2% (0603) X8R	2	C0603C200G4HA	C27, C28	https://gr.mouser.co	Chip Capacitor, 2.2 nF, +/- 10%, 50 V, -55 to 125 degC, 0603 (1608 Metric), RoHS, Tape and Ree
5	ok	100	yes	100 pF 50V 5% (0603)	1	06035A101JAT2A	C42	https://gr.mouser.co	CAP, CERM, 100 pF, 50 V, +/- 5%, C0G/NP0, 0603
6	ok	10	yes	330 pF 630V 5% (1206) C0G (	1	GRM31A5C2J331	C41	https://gr.mouser.co	CAP, CERM, 330 pF, 630 V, +/- 5%, C0G/NP0, 1206
7	ok	100	yes	560 pF 50V 5% (0603) C0G (N	4	C0603C561J5GA(	C65, C66, C68, C6	https://gr.mouser.co	Chip Capacitor, 2.2 nF, +/- 10%, 50 V, -55 to 125 degC, 0603 (1608 Metric), RoHS, Tape and Ree
8	ok	100	yes	680 pF 50V 10% (0603) X7R	1	C0603C681K5RA(	C15	https://gr.mouser.co	CAP, CERM, 680 pF, 50 V, +/- 10%, X7R, 0603
9	ok	10	yes	1 nF 2KV 10% (1206) X7R	1	CC1206KKX7RDB	C38	https://gr.mouser.co	Capacitor
10	ok	100	yes	1 nF 100V 10% (0805) X7R	1	08051C102KAT2A	C25	https://gr.mouser.co	1nF Capacitor
11	ok	100	yes	2.2 nF 20V 20% (0603) X7R	1	C0603C222K4RE(	C51	https://gr.mouser.co	SMD/SMT 50V 2200pF X7R AUTO 20% Flex 0603
12	ok	10	yes	2.2 nF 2.2KV 10% (1812)X7R	2	C4532X7R3D222(	C13, C14	https://gr.mouser.co	CAP, CERM, 2200 pF, 2000 V, +/- 10%, X7R, 1812
13	ok	20	yes	8.2 nF 50V 10% (0603) X7R	1	GCD188R71H822	C44	https://gr.mouser.co	CAP, CERM, 8200 pF, 50 V, +/- 10%, X7R, 0603
14	ok	100	yes	10 nF 50V 10% (0603) X7R	3	C0603C103K5RA(	C46, C6	https://gr.mouser.co	SMD/SMT 50V .01uF X7R 0603 10%
15	ok	100	yes	22 nF 50V 10% (0603) X7R	1	C1608X7R1H223(	C43	https://gr.mouser.co	22nF Capacitor
16	ok	150	yes	100 nF 50V 10% (0603) X7R	29	C1608X7R1H104(	C24, C1, C2, C4, C	https://gr.mouser.co	100 nF 10% X7R Capacitor
17	ok	50	yes	10 nF 100V 10% (0805) X7R	2	GRM21BR72A103	C17, C26	https://gr.mouser.co	CAP, CERM, 0.01 µF, 100 V, +/- 10%, X7R, 0805
18	ok	30	yes	220 nF 50V 10% (0805) X7R	1	C0805C224K1RE(	C23	https://gr.mouser.co	CAP, CERM, 0.22 µF, 25 V, +/- 10%, X7R, 0805
19	ok	20	yes	1 uF 100V 10% (1210) X7R	3	C3225X7R2A105(	C18, C19, C45	https://gr.mouser.co	1uF 100V X7R 10% 1210 Capacitor
20	ok	100	yes	1 uF 25V 10% (0603) X5R	8	C0603C105K3PA(	C53, C7, C49, C58	https://gr.mouser.co	SMD/SMT 1uF 25V X5R 10%
21	ok	50	yes	2.2 uF 10V 10% (0603) X7R	3	C0603C225K8RA(	C3, C10, C54	https://gr.mouser.co	Chip Capacitor, 2.2 nF, +/- 10%, 50 V, -55 to 125 degC, 0603 (1608 Metric), RoHS, Tape and Ree
22	ok	10	yes	10 uF 50V 20% (1210) X7S	1	UMR325AC7106(	C48	https://gr.mouser.co	SMD/SMT 1210 50VDC 10uF 20% X7S AEC-Q200
23	ok	150	yes	1 uF 16V 10% (0603) X7R	5	C0603C105K4RA(	C50, C55, C59, C6	https://gr.mouser.co	Chip Capacitor, 2.2 nF, +/- 10%, 50 V, -55 to 125 degC, 0603 (1608 Metric), RoHS, Tape and Ree
24	ok	10	yes	47 uF 16V 10% (1210) X5R	2	EMK325ABJ476K	C21, C22	https://gr.mouser.co	CAP, CERM, 47 µF, 10 V, +/- 10%, X5R, 1210

Figure 26.

For the ordering, one of the 3 well known components distributors was used, and the procedure was very simple. Searching for the desired part number, the quantity needed, and adding the part to the cart.

Once everything arrived, it was time for assembly. The tools used were a manual soldering station equipped with a fine nose and a hot air station. The firs components to be assembled were the PoE power stage as well as the external DC-DC converter. The reason is that while soldering, various tests were performed to ensure that the different sections of the PCB worked as intended. More specifically, the PoE stage was tested first, without the PHY and MCU being even soldered. In this way, potential mistakes could be corrected before any other components were also included in the system, thus reducing the risk of damaging chips. The same procedure was followed with the external DC-DC converter. In both cases, an oscilloscope was connected to the output of each section and the voltage was monitored, in order to evaluate if they behave as intended, and they sure did. Next, the ethernet section, ST-Link and MCU were assembled, and some basic firmware to test the ethernet functionality (basic ping function) was uploaded. The interfaces (bottom side of the PCB) were soldered and tested last, when all the other parts were already proven to be functioning correctly. The green connectors were soldered lastly, to leave space for the various tools for the placement and soldering of the components.
Figure 27 and 28 illustrate some pictures taken during the manual assembly of the first PCB.

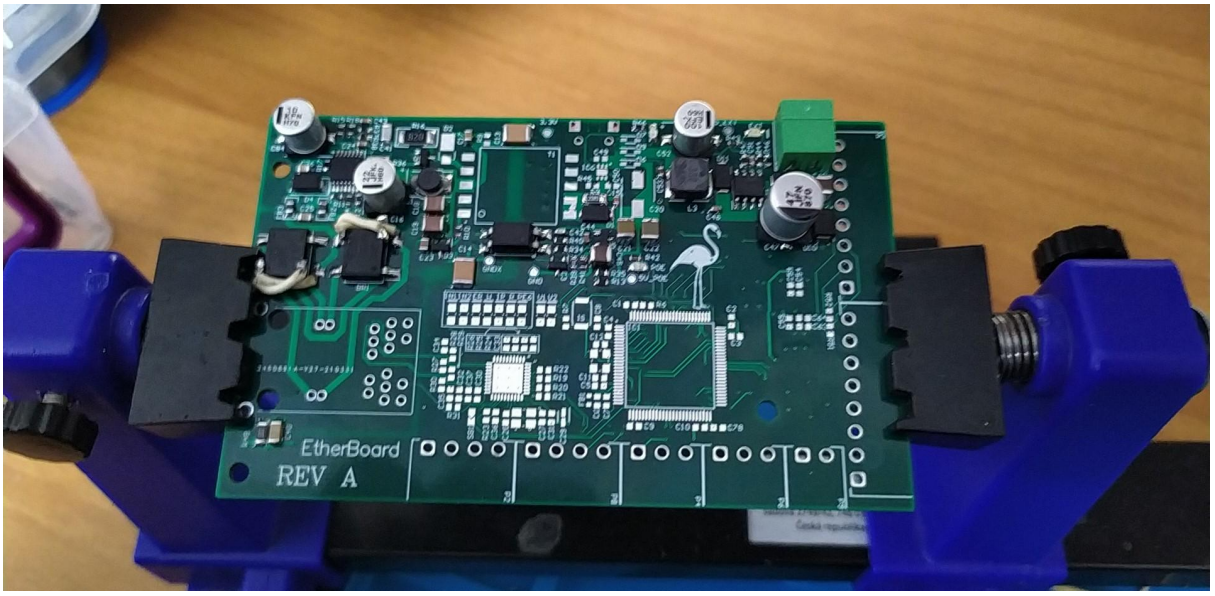


Figure 27.

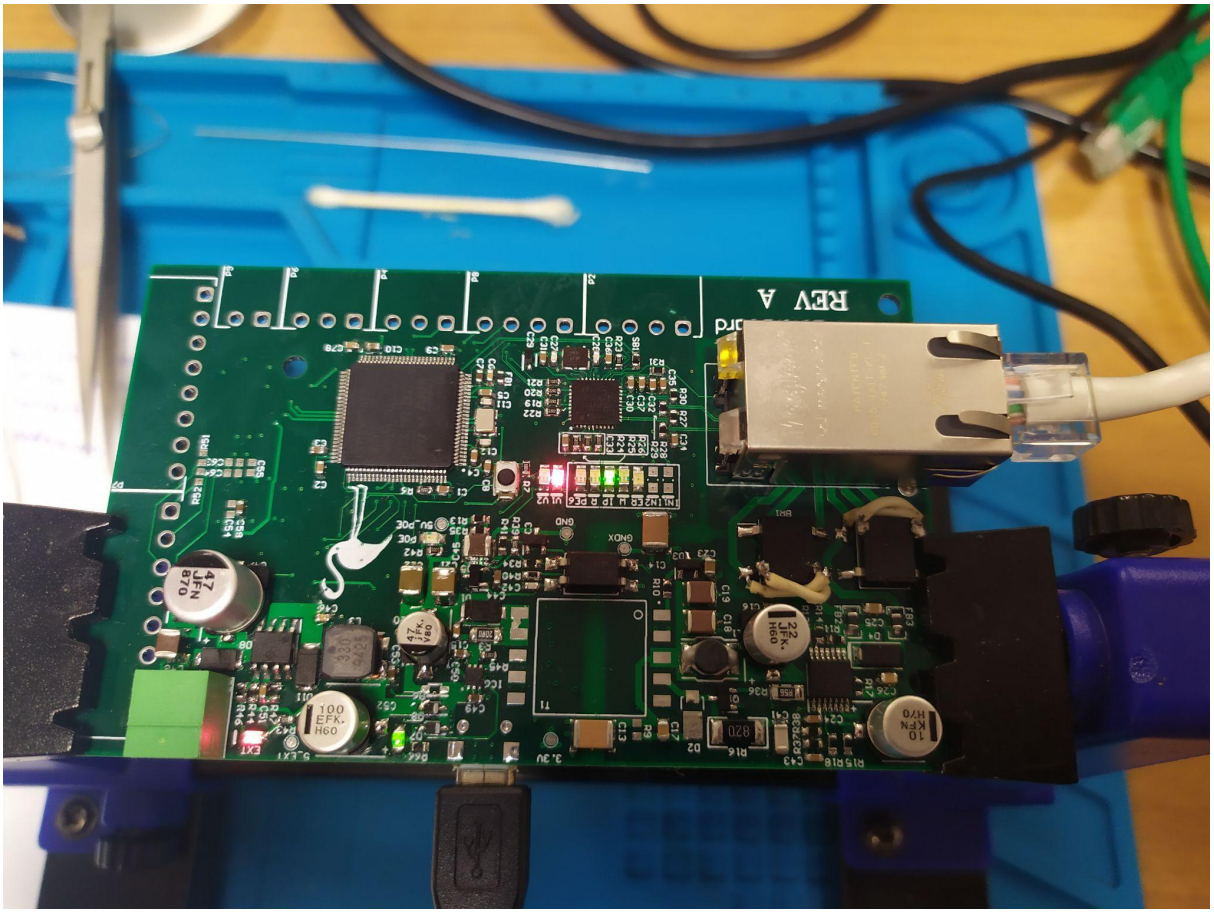


Figure 28.

3.1.3 Testing and Troubleshooting

While assembling and testing the various parts of the system, some errors of the design prevented the system from behaving as intended. More specifically, the ethernet functionality was not functioning due to a short circuit between two pins of the MCU that were spotted after viewing the traffic of the RMII interface and seeing the TX and RX line moving together. Since RMII is a full duplex bus, this behavior led to further PCB optical and continuity testing inspection, which revealed the short circuit between the pins. After reflowing the solder, the error was resolved.

Another Similar situation was occurring when trying to interface with the external ADC through the SPI line. With the use of a logic analyzer, it was found that the chip didn't actually respond to any commands and messages. After probing some of the supply pins of the chip, it was noticed that one auxiliary pin was not properly supplied with 5 volts, which was the reason the chip was not responding to the SPI traffic. With the use of a quick hook up wire, the pin was connected to 5 volts, and the ADC started sending all the expected messages.

Overall, the assembly of the prototype PCB took a full day of work, whereas a pick and place machine could do the same in minutes. However, as mentioned, the aim was to minimize the cost, and design a prototype system which in turn was expected to have flaws and errors. Soldering by hand however, introduces more errors in the HW which is why pick and place and autoclave machines are used in industry.

3.2 Firmware

The development of the FW as a process, started during the system's initial analysis and design. Way before the board was designed. This happened to allow any limitations and obstacles of the selected MCU and peripherals to be seen before the HW design could start. In this way, corrections and changes to the design would be very easy, since the design process was still in its early stages.

In order to develop some initial FW without the need of designing HW, a development board was purchased and used as a platform for the initial stages of the system application code and its various components. By using a premade development board (NUCLEO-F429ZI) we could focus entirely on the FW development without worrying about HW mistakes or design errors. Moreover, this approach is in line with the W model, since we are essentially developing and testing every part of the system before moving to the next stage.

3.2.1 Initialization

The first step before starting the FW development was to initialize the MCU's peripherals, clocks and generate the backbone of the application code. As mentioned in previous chapters, ST offers a software tool that achieves the above through an easy-to-use graphical environment. With the help of STM32CubeMX, the desired peripherals were enabled and set up appropriately. Timer peripherals were also enabled to help with various time measurements and flow control. The debugging interface was enabled to allow us to upload and debug code on the development board with the help of the on board ST-Link debugger. The clock structure was configured to achieve the maximum frequency, to allow for code execution margin during the development.

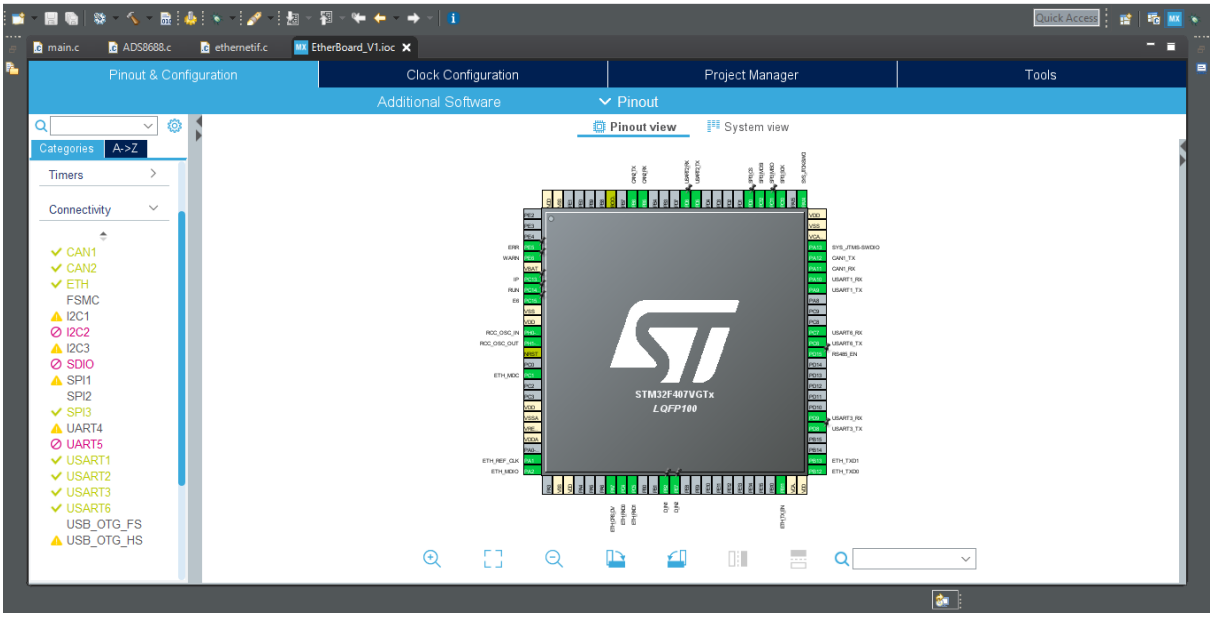


Figure 29.

At this stage, all the peripherals of the MCU were configured and with the help of MXCube, the initial code was generated. The tool generated the directory tree of the project that included HW low level drivers, peripheral initialization code, register defines and all the functions to get the MCU up and running. The next step was to start developing user code for the various functions of the system.

3.2.2 Peripheral Drivers and Utilities

In order to make the FW development more efficient and rapid, individual peripheral code was developed that was responsible for interacting with the already existing HAL API. This code was still considered a part of a driver because it was generic and ready to be used by an application. This modular code design makes the FW portable and easy to reuse to other projects, or even to the open source community.

As a fist step, the ethernet module was studied and the main HAL functions were used to initialize the peripheral, and start the DHCP module in order to acquire an IP address from the network. The ping function was tested with success, as the board responded to incoming ICMP packets. The next step was to be able to listen to specified ports , accept incoming connections and pass data through them. This would be the core of the system's functionality, so it was developed as the first driver.

The next step was to communicate with the external ADC peripheral and acquire its voltage conversions. For that reason, an SPI driver was needed. Unfortunately, after searching for such a driver in the open source community, we were unable to locate one. It was then decided to develop from scratch a simple driver that initializes the ADS8668, and acquires the 8 values of its inputs by polling the SPI line. The driver was designed with portability and ease of use in mind and it surely was incorporated smoothly in the application code. The driver is available in the open source community under an MIT license. [7]

The CAN, RS232 and RS485 initializations and send receive functions were created in a format that is easy to be used by higher level code, offering an abstraction layer between the HW functions and the application.

For the reasons of debugging and performance reporting, extra functions that interact with the Serial peripheral connected to the ST-Link were developed. By using these functions, the application code could send debug messages to the PC via the virtual COM port that was created by the ST-Link. The same was done with one of the available TCP ports on the ethernet side, giving the option to report debugging messages via ethernet.

In order to be able to save application configuration data, such as a static IP address, the predefined ports and other information even when the device was powered down, without the need to upload new code, a dedicated Flash memory driver was developed. This driver is able to read and write blocks of data from and to the Flash memory of the MCU, allowing information to be saved and restored in the event of a power cycle.

All the above drivers were developed and tested individually in order to simplify the design process and avoid mistakes that cost in development time. Once every component was tested, it was incorporated to the main FW and the code was tested once again to ensure that the new component interacts correctly with the system, without causing errors.

3.2.3 Application

Since all of the lower level functions were completed, it was time to develop the actual application code. For that reason, a simple scheduler was implemented. Its tasks were to scan the system's inputs, including the 8 analog ports, the CAN, the 3 Serials and the digital inputs, write the data to all the active TCP connections and report any errors or events that occurred during the current code execution.

As an extra functionality, and to allow the user to change the configuration of the system, a custom HTML page was implemented that is being served by the LWIP driver which is an open-source TCP/IP stack designed for embedded systems. The page is interactive, meaning that it displays the current configuration data that are present in the MCU and the user can change various variables and commit the changes to the system. In this way, it is not needed to upload new code every time you want to make a small modification to the system's parameters.

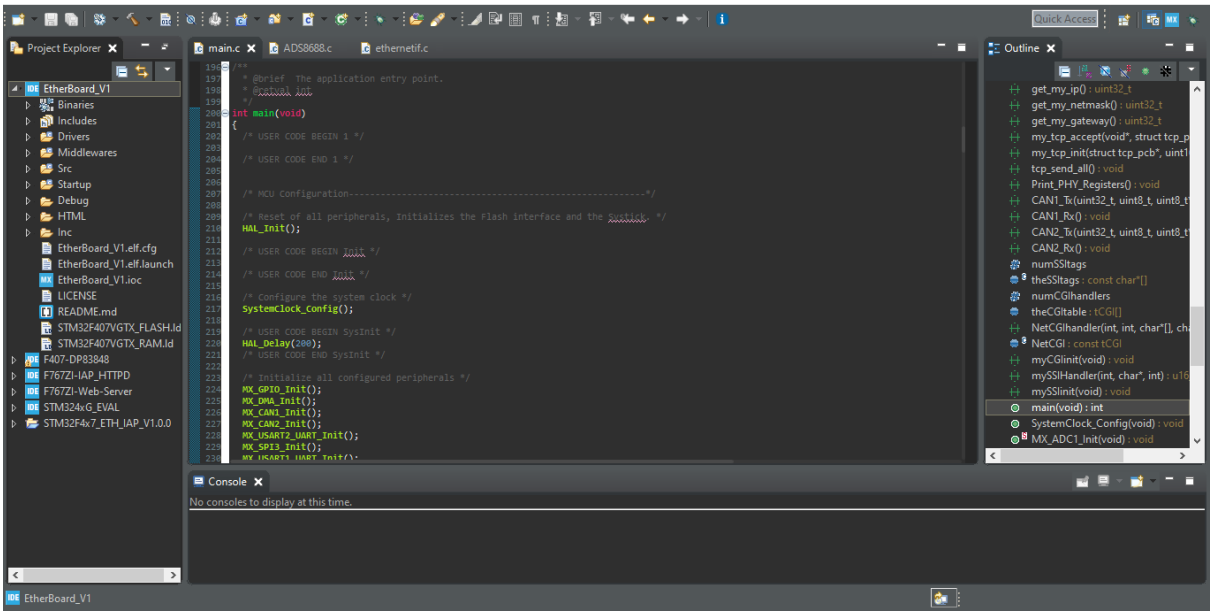


Figure 30.

The complete code is available on github under an MIT license. [8]

4 Results

4.1 Testing Platform

After implementing and testing all of the HW and FW components, it was time to test the complete system using an actual installation. The designed and assembled HW, and the FW that was developed and tested on the NUCLEO dev board were used in combination with simple general purpose sensors. In order to evaluate the functionalities of the system, the sensors were connected to the board in order to acquire their measurements and then forward them to a simple application that visualizes them. The board was connected to a simple ethernet network that included just another PC where the visualization application was running.

The aforementioned sensors were selected for the system evaluation and in order to produce time-variable data that can then be visualized. Since the target was to quickly evaluate the system's functionality, the selected sensors were low cost with general use and an analogue output. They were selected so that they simplify the application code, and to avoid developing protocols for the communication. A simple voltage measurement that translates to the physical measurement was acquired by the ADC and sent via the ethernet (one measurement per ethernet port).

The sensors used can be seen below.

Sound Sensor CO Sensor Simple Switch Light Sensor Current Sensor

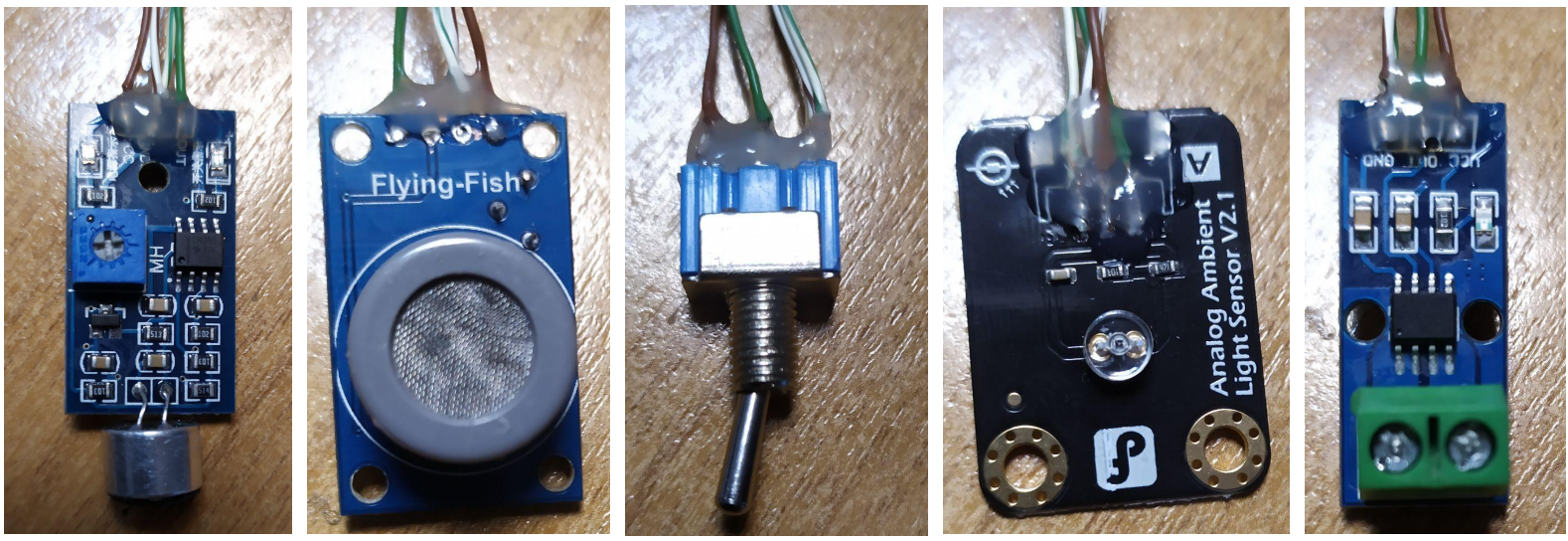


Figure 31.

All of them have an output range of 0-5V and are connected to the inputs of the external ADC of the PCB. An extra voltage measurement is also being taken from the supply of the system and the sensors. Figure 32 illustrates the connectivity of the system, the sensors and the PC.

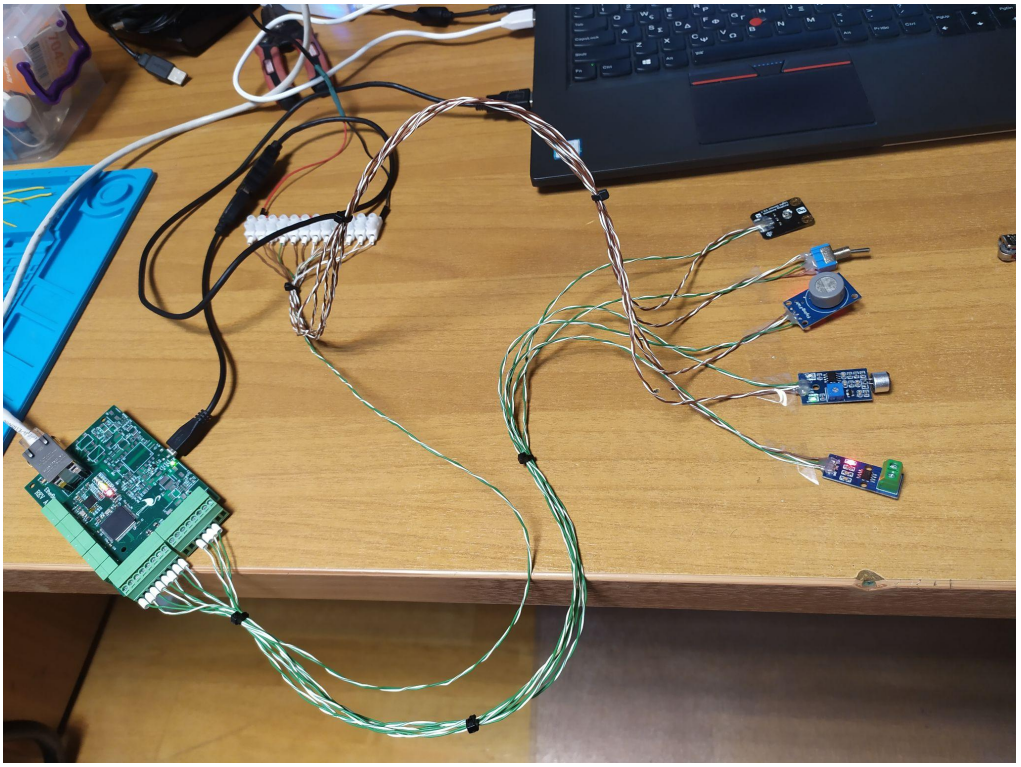


Figure 32.

4.2 Measurement Visualization

The voltage measurements are being collected by the system and sent through ethernet to the connected PC. Each measurement is sent with a frequency of 1Hz at different ports, starting from port 5000. The connected PC then collects these values through NodeRed, which is an open source python based application that makes network and connectivity related tasks very easy to handle. One TCP connection block was created for every port of the system that fed the measured data of each sensor. The measurement data was the voltage of the output of each sensor. This voltage was then passed through a conversion rule that translated the measurement to the actual physical value of each sensor.

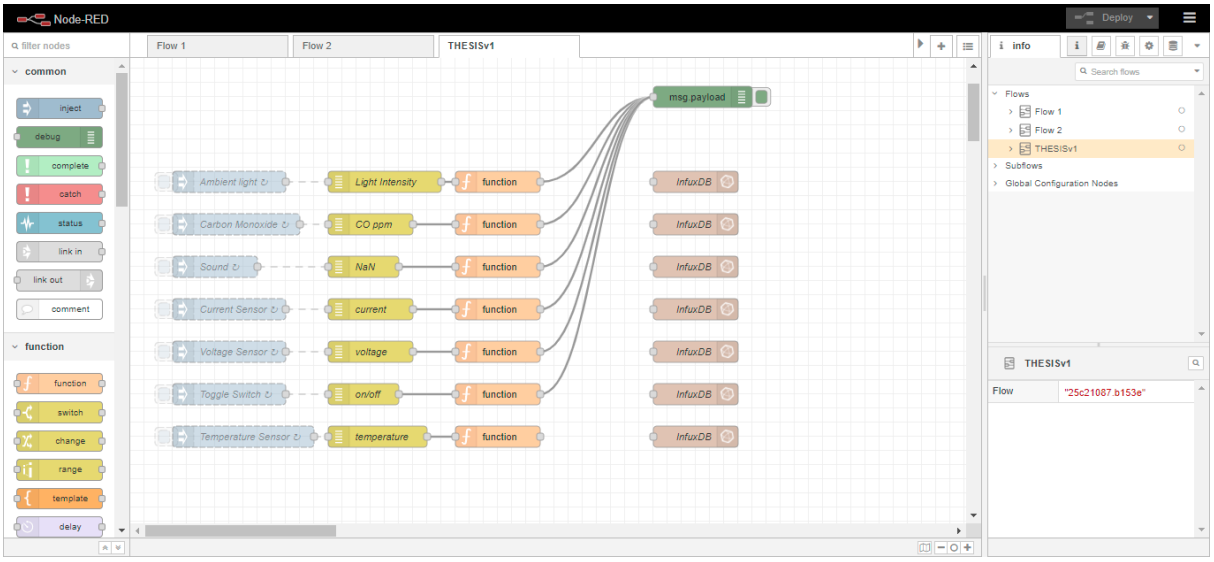


Figure 33.

In order to visualize those data, the fist step was to save them in a database. For that reason, InfluxDB was selected, as it is very easy to setup and interface through NodeRed. Each sensor's measurements are saved in the database in the form of time series, in order to have access to previous measurements, as well as live data feed.

For the actual visualization, another open source software called Grafana was selected. It connects to the previously mentioned database and can access all saved data and timestamps. By creating different dashboards, it is possible to add plots, graphs and other widgets, in order to visualize data, inspect previous measurements, create derived channels and create alerts.

In our case, one widget was used for each sensor measurement, and depending on the type of measurement, previous data are also visible. FIgure 34 presents the dashboard with the various widgets.

Figure 34.

5 Conclusion

Looking back at the design specification and system requirements, we can clearly see that a great amount of them has been fulfilled. All of the required system inputs are in place and functioning with their specified performance and behavior. The system is able to interface with generic industrial analog and digital sensors and provide accurate, continuous data, which can feed predictive maintenance and forecasting applications, as well as serve as a long-term machine logging database.

The system was designed in a way that it is modular enough to be used in different environments, while at the same time being easy for someone without advanced technical skills to install it and set it according to the application and the measurement requirements. Despite the lack of options in the web interface, it manages to make the setup experience quick and pleasant, while at the same time not hiding critical information and parameters from the user.

In the previous chapters, we covered the design process, starting from a high level, going all the way to the circuit and HW design, as well as some insights about the FW development. As mentioned in the Introduction, this thesis did not cover the implementation of an application that handles the data, as would be the case for a real industrial application. However, for the reason of demonstration, an initial monitoring application was developed to provide proof of concept, evaluate the overall performance of the system and showcase its abilities. By attaching some low-cost sensors to the system and connecting it to a simple network, through the help of open source software we were able to set up a monitoring application with enough capabilities for a small business. The desired measurements are fed at 1Hz to the computer running the application, and visualized in a simple way, without hiding any information from the end user. Historical data are also available, in order to make quick comparisons.

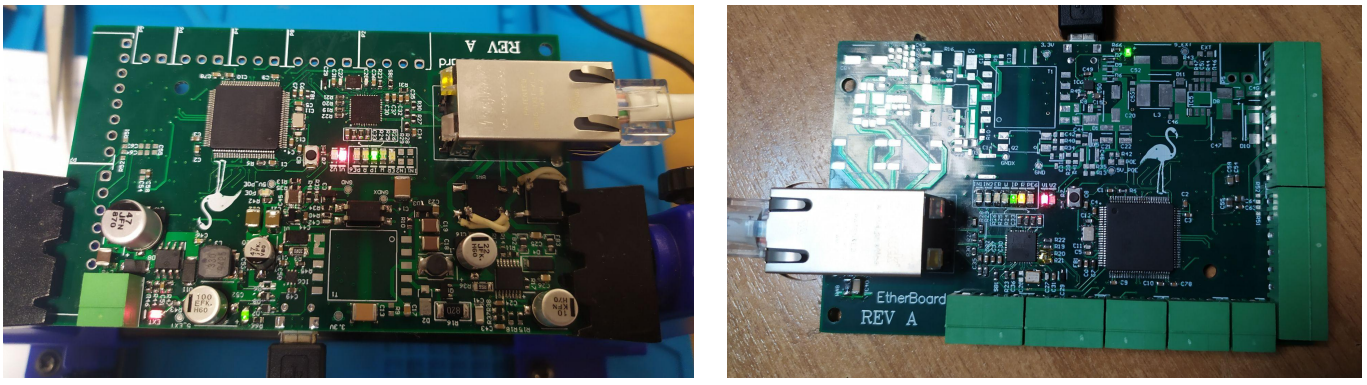


Figure 35.

Overall, this thesis covered the majority of the design decisions as well as the techniques that are needed in order to design and implement a simple PCB with ethernet and multi sensor connectivity. Moreover, various supply options, including PoE and DC-DC circuits were demonstrated and implemented, making the device a HW and FW platform, on which additions and modifications can be made to fit the needs of a different application.

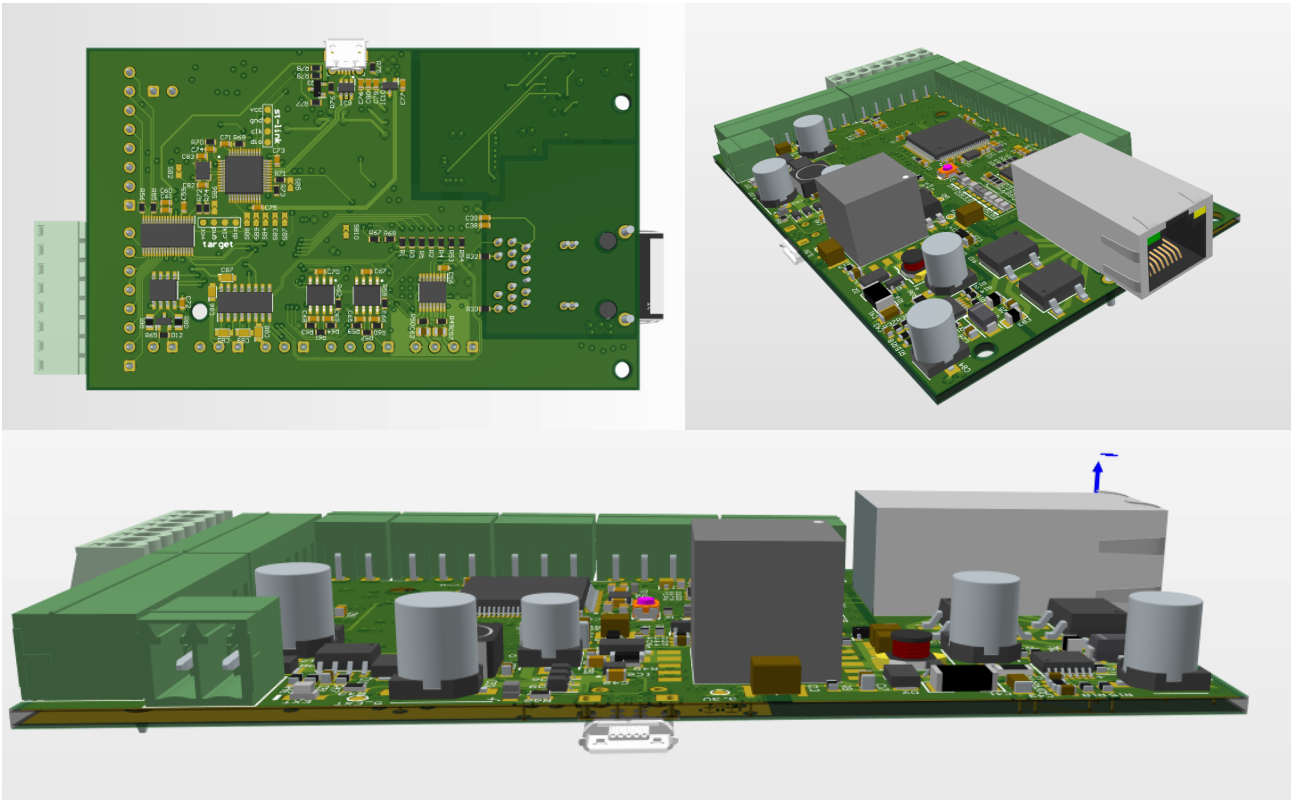


Figure 36.

5.1 Practical Application

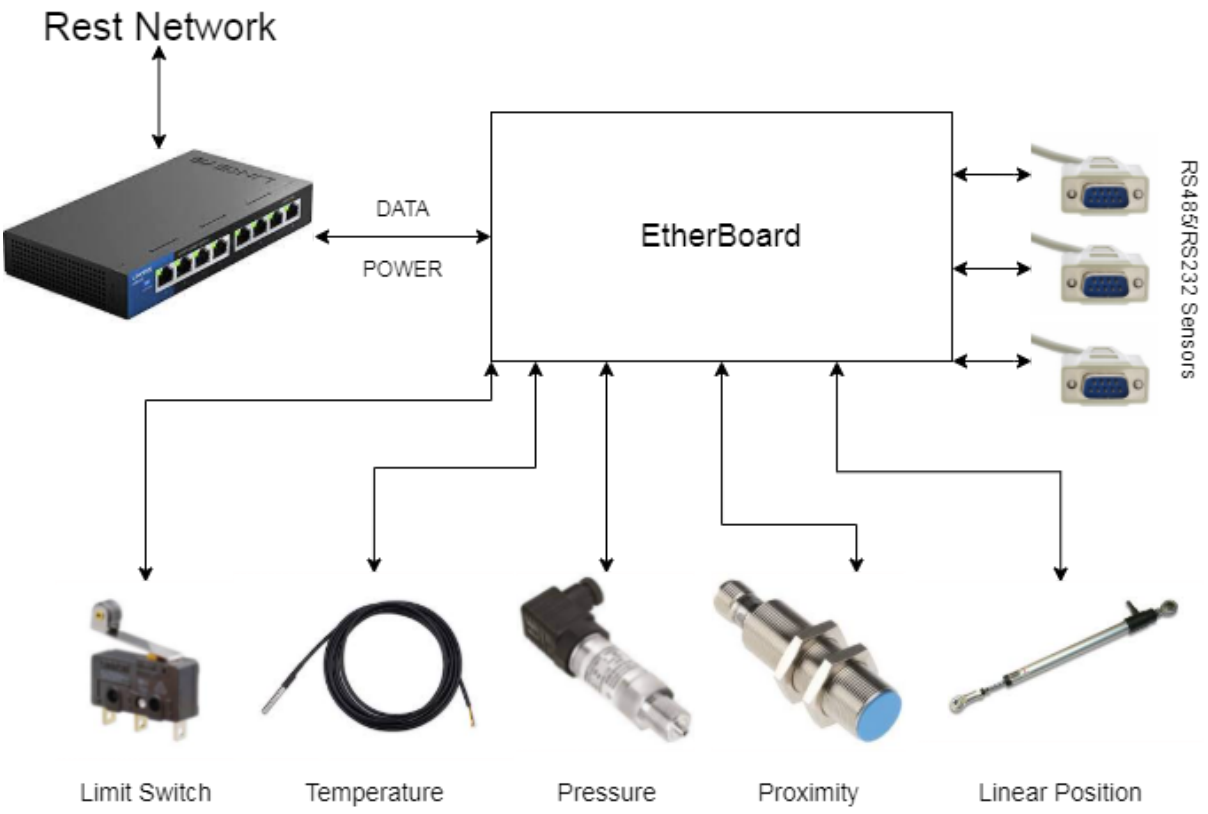


Figure 37.

Figure 37 illustrates just a number of possible sensors that can be connected to the system. Typical industrial applications that can be implemented using this thesis system include:

- Door/Hatch state with a simple limit switch
- Axle rpm with proximity sensor
- Width measurement with optical laser
- Multiple spot temperature measurement for temperature mapping
- Machine displacement using linear potentiometers
- Machine vibration using analog accelerometers (predictive maintenance metric)
- Hydraulic/Air pressure and temperature with industrial 10V sensors
- Ambient Light, Quality of air, CO/CO2 concentration, Sound level

As per the specifications, the device needs a single PoE line that provides the power supply and transfers the data from the device to the network and vice versa. The system can be deployed when there is a need for additional monitoring with external sensors, or even act as the machine sensor data source for old machines that do not have network connectivity.

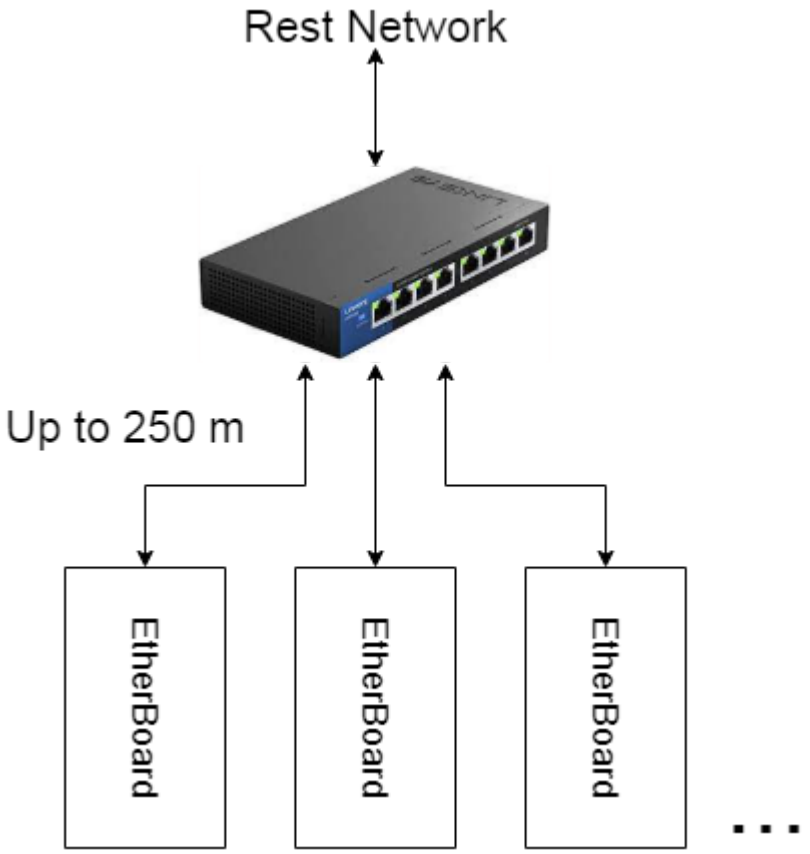


Figure 38.

Since the system utilizes common TCP over IP connections, the expansion of the system becomes trivial, and can be handled as any other network device, passing through switches and even different networks. FIgure 38 illustrates the utilization of multiple devices connected to a single PoE switch. These devices however, can be scattered around and connected through multiple layers to the company network, or even a standalone dedicated network.

5.2 Production Cost

As far as the cost of the system, Table 7 contains the complete order cost for five devices, as well as the unit cost.

Category	Cost for Project	Cost for 1 device
Electronics Components	637.56 €	127.512 €
PCBs	28.37 €	5.674 €
Total cost per device		**133.159 €**

Table 7.

As seen from the table above, the cost for a single device does not exceed the predefined 200 euros amount that was set in the system requirements chapter. As mentioned in several points in these thesis, the overall cost is a result of a prototype design. This means that when producing a greater amount of devices, the price per device will be slightly lower. Also, removing the aspects of the device that are not needed in a product can lower the cost even more.

Below, we can see a comparison of the unity price of various devices with similar capabilities. It is worth mentioning that in our case, the calculated cost does not contain a protective enclosure and possible shipping fees, as the other products do. These products were the most affordable DAQs with similar specifications.

Device	Unit Price
Digilent 6069-410-011	1022 €
Allen-Bradley 2080-IQ4OB4	305 €
Digilent 6069-410-015	419 €
DATAQ DI-4000	370 €

Table 8.

It is clearly visible that even if we consider another 100 euros for the design and manufacturing of an appropriate protective enclosure, our device is still more cost effective than other similar devices on the market.

6 Discussion and Recommendations for Future Research

Below are presented some of the Limitations and Issues that were found during the development of the system, as well as propositions and recommendations for the system's future research.

6.1 Limitations and Known Issues

The first limitation of the system is, as mentioned in the beginning, the overall performance and characteristics of the analog inputs. These are defined by the external ADC specifications and were selected since they provided a good compromise between speed, cost and measurement accuracy.

The web interface did not get fully developed, and at this stage it only provides the possibility to change the IP address of the system, or enable the DHCP to acquire one. This was done intentionally since it was not the main function of the system and priority was given to peripheral driver and actual application development.

The first revision of the PCB had some notable mistakes that, with combination to hand soldering, led to undefined behaviors of the FW and the peripheral communication. A second revision could be implemented to correct those mistakes and also remove the debugger circuitry to lower the overall cost.

6.2 Future Work/Research

The first step in improving the system is to actually improve the sensor measurements. As a fist step, a digital filter could be implemented, to smooth out the values coming from the external ADC. Since this used SPI to transfer the converted voltages, the implementation of the communication via DMA would save substantial core code execution, and free up some performance on the application code. The filter and the DMA SPI could be implemented all in interrupt context with a circular buffer, where half is used to accept new measurements and the other half is used to perform an averaging method or another type of filter. The size of this buffer would be determined by the filtering strategy.

As a second step, the web interface of the system could be improved and enriched with features that make it more modular. More specifically:

- Abstract sensor channels could be added that can be later assigned to a peripheral
- Parameters such as:
    - logging frequency
    - ethernet port mapping
    - voltage range
    - math operations (gain and offset)
    - and other functionalities

could be added to make the system's setup easier and faster. Another part of the web interface could be dedicated to uploading new firmware to the system over ethernet, without the need to physically access any device.

As an addition to the system, a separate PCB could be designed that extends the input capabilities of the system and also introduces new types of inputs. This extension could be connected to the system via one of the two available CAN buses and exchange data and control information with the system's main PCB. This information would then be redirected to the main flow of data through the ethernet TCP connection. This solution can be very cost effective since the most expensive part is the main PCB and the extensions are less expensive to design and manufacture, but contribute a lot to the system's modularity.

To further expand the capabilities of the device's connectivity, an extra module, or another revision that implements a simple GPRS (mobile data) module could be added. In this case, the system could send the sensor measurements wirelessly through a 4g/5g network to a dedicated data server. The device would then need external supply, since the network cable would not be needed anymore. This solution is ideal for small and remote businesses that either do not have, or are not able to host a local network.

The same could be implemented with WiFi or even LoRa networks, automatically making the device a node in the internet of things (IoT) network.

References

[1] Oscillator design guide for STM8/STM32 MCUs and MPUs
https://www.st.com/resource/en/application_note/an2867-oscillator-design-guide-for-stm8afals-stm32-mcus-and-mpus-stmicroelectronics.pdf

[2] NXP Application note for Automotive Ethernet
https://www.nxp.com/docs/en/application-note/AN12088.pdf

[3] PoE Explained
https://www.veracityglobal.com/resources/articles-and-white-papers/poe-explained-part-2.aspx

[4] Ethernet Connector Datasheet
https://eu.mouser.com/datasheet/2/643/dr-MAG-0826-1X1-TKL-F-1670779.pdf

[5] TI Controller Area Network Physical Layer Requirements
https://www.ti.com/lit/an/slla270/slla270.pdf

[6] 4-20mA Current Loop Cost effective design
https://resources.altium.com/p/how-to-design-a-4-20ma-current-loop-receiver-with-minimal-components

[7] ADS8668 SPI Driver
https://github.com/obatsis/ADS8688-STM32

[8] Application Code
https://github.com/obatsis/EtherBoard

[9] Data Acquisition Handbook
https://www.mccdaq.com/pdfs/anpdf/data-acquisition-handbook.pdf

[10] The resistor Guide
http://www.resistorguide.com/pictures/The_Resistor_Guide.pdf
Some STM32 tutorials and seminars

[11] TIDM-TM4C129POE Reference design
https://www.ti.com/tool/TIDM-TM4C129POE#tech-docs

[12] Design Considerations for Mixed-Signal, How to Design a PCB Layout
https://www.sparkfun.com/datasheets/BreakoutBoards/mixed_signal_design.pdf

[13] Partitioning and Layout of a Mixed-Signal PCB
http://ee.mweda.com/imgqa/smqh/circuit/circuit-1012qafr1lhzomy.pdf