Final Report:
Development of the Xe-Link sensor adaptor

Xeelas,
The Hague University of Applied Sciences
Faculty Technology, Innovation & Society

Author: L. Pardon

Supervisor: Michaël Devid
Mentor: Paul Witte

May 28, 2020

Xeelas
Spaarneplein 2, 2515 VK, The Hague

The Hague University of Applied Sciences
Rotterdamseweg 137, 2628 AL, Delft

Foreword

This document has been set up for the The Hague University of Applied Sciences, Electronics studies. As the graduation report of Lex Pardon.

This project is executed in behalf of Xeelas B.V. , an Internet of Things (IoT) company in the industrial field. The project is aimed to improve the market range of Xeelas' products by adding a product called the Xe-Link. This device will be developed by Lex Pardon in his Thesis, which is documented in this document.

I would like to thank Xeelas for giving the opportunity to develop myself further in the field of electronic design.

With a special thanks to Michaël Devid for support on hardware development and the opportunity to do the internship in first place. Second I would like to thank Maarten van Dootingh for helping set up the software development tools and giving the opportunity to develop the software of the Xe-Link as it is of now.

Samenvatting/Summary (Dutch)

Dit verslag is opgesteld om het ontwerp-process van de Xe-Link Sensor interface board te documenteren, als afstudeeropdracht. Dit verslag is opgesteld door Lex Pardon voor de Haagse Hogeschool in de vorm van een eindverslag van een stage-periode bij Xeelas.

De Xe-Link sensor interface board is een klein stuk electronica dat een verbinding tussen de standaard Internet of Things (IoT) Gateway en sensoren kan maken. De Xe-Link zorgt voor de mogelijkheid om de Xeelas IoT Gateways meer verschillende soorten sensoren aan te sluiten. Dit resulteert in een grotere markt-breedte, wat resulteert in meer klanten, daardoor meer potentiele inkomsten.

De research, ontwerp en test fase van de Xe-Link worden in dit document toegelicht. Waarbij belangrijke, gefundamenteerde ontwerpkeuzes worden uitgelecht en besproken.

Deze samenvatting vat kort samen wat de hoofd-acties en gebeurtenissen zijn die besproken worden in dit document.

Methode en plan van aanpak

Het plan van aanpak om dit project aan te pakken is te omschrijven in verschillende stappen:

1. Probleem analyse
2. Research
3. Hardware Design
4. Software ontwikkeling
5. Test-fase

Nadat alle hierboven stappen zijn uitgevoerd, kan er een conclusie getrokken worden of het project volledig functioneel is of dat er nog verbeteringen toegepast moeten worden.

Meer informatie over de stappen in grotere details zijn te lezen in hoofdstuk 2

Probleem analyse

Een goed overzicht van het probleem is belangrijk, hierbij wordt gekeken naar het grotere plaatje. Door middel van een overzicht dat getekent is, is duidelijk dat er twee problemen verder onderzocht moeten worden. Dit zijn de communicatie met de IoT gateway en de verbonden Sensoren die de metingen moeten uitvoeren en uitgelezen worden.

In beide gevallen moet hier één of meerdere concepten onderzocht en gekozen worden.

In de probleem analyse wordt een lijst met vereisten opgesteld om de scope van het project duidelijker op te stellen. Deze lijst en overzichtelijke plaatjes zijn te zien in hoofdstuk 3

Research

Tijdens de research fase zijn er verschillende communicatie protocollen en sensoren onderzocht.

Er is onderzocht welk communicatie protocol dat verbinding maakt tussen de IoT-Gateway en de Xe-Link het best overeenkomt met de applicatie. De verschillende protocollen waar onderzoek naar gedaan is zijn:

- Simple Sensor Interface (SSI)
- SCPI
- ModBus RTU
- Cayenne LPP
- JSON
- MQTT

Na het vergelijken van de protocollen, waarbij de voor en nadelen onderzocht zijn, is er gekozen voor de Simple Sensor

Interface protocol. Dit protocol past het beste bij de applicatie van Xe-Link. Door de overeenkomende nodige functies en bi-directionele communicatie mogelijkheden.

Er is ook onderzocht welke sensoren de Xe-Link zal ondersteunen. Hierbij is ook gekeken naar verschillende sensoren:

- Analoge Sensor - 4-20 mA
- Analoge Sensor - 0-10 V
- Analoge Sensor - PT100
- Digitale Sensor - IO-Link
- Digitale Sensor - ModBus RTU (RS232 en/of RS485)
- Digitale Sensor - CAN-Bus
- Digitale Sensor - LIN-Bus

Na het onderzoek is geconcludeerd dat de Xe-Link veel potentie heeft in het ondersteunen van verschillende sensoren. Om het project echter nog goedkoop en klein te houden is ervoor gekozen om twee verschillende versies uit te brengen:
Xe-Link Sense: De analoge sensor variant van de xe-link, ondersteunt 4-20 mA, 0-10 V en PT100 sensoren. Om deze sensoren de correcte spanning te leveren, zorgt de Xe-Link Sense voor een variable uitgangsspanning tussen de 5 en 24 Volt.
Xe-Link Serial: De digitale sensor variant van de xe-link, ondersteunt ModBus, Can-Bus en Lin-Bus.

Tijdens de stage zal het ontwerp van de Xe-Link Sense verder ontwerpen worden. Waarbij de Xe-Link Serial later opgepakt kan worden als de Xe-Link Sense een succes is.

Hardware Design

Na het onderzoek is de ontwerpfase van de electronica van start gegaan. Hierin worden onder andere het telectrische schema toegelicht. Het schema is opgedeeld in meerdere subonderdelen:

1. Low Dropout regulator
2. Bosot converter to power the sensors
3. EMI/EMC suppression
4. Analogue input (0-10V and 4-20 mA)
5. Analogue input PT100
6. Microcrontroller
7. UART level shifting

Elk sub-onderdeel is apart toegelicht in hoofdstuk 5

Na het ontwerp van het elektrische schema, is de PCB ontworpen. Hierbij is rekening gehouden met onder andere de dimentie van de PCB, plaatsing van componenten en layout van gevoelige onderdelen van het schema. Meer informatie over de ontwerpkeuzes en handelingen tijdens het ontwerp van de pcb is te vinden in hoofdstuk 6.

Software ontwikkeling

Na het ontwerpen en het assembleren van het ontwerp wordt de software voor de Xe-Link micro-controller ontwikkeld. Dit process vindt plaats in Segger Studio's met de Nordic Software Development Kit. Hieruit worden hardware drivers gebruikt om de software voor de Xe-Link verder te ontwikkelen. Er is voor verschillende sub-onderdelen software ontwikkeld:

1. Boost Converter controller
2. SAADC Drivers
3. SAADC - 0-10V Sensor
4. SAADC - 4-20mA Sensor
5. SAADC - Uitgangsspanning
6. SAADC - Ingansspanning

Naast de hierboven opgenoemde software sub-onderdelen zijn er momenteel onderdelen die nog niet ontwikkeld zijn door tijdgebrek. Dit zijn de PT100 Sensor en het Communicatie-protocol.

Tests en test resultaten

Om de functionaliteit van de Xe-Link te verzekeren, moeten er tests uitgevoerd worden. Hieruit zal blijken of de hard en software voldoende of onvoldoende functioneert.

Hierbij is gekeken naar de volgende tests:

1. Power consumptie en ingang spanning range
2. Boost converter (spanningscontrolle, opstarttijd, efficentie en aan uit pin)
3. In en out gang spannings meting d.m.v. Analoog digitaal converter
4. Analoge 4-20 mA sensor
5. Analoge 0-10 V sensor

Er zijn drie onderdelen die tijdens de stageperiode niet getest zijn door een tekort aan tijd en meet-apparatuur. Dit zijn de PT100 ingang, communicatie met de gateway en de EMI/EMC suprressie.

Voor de rest is alle hard- en software getest, waar overal positieve resultaten uit kwamen op een klein probleem na: De Boost converter uitgang spanning kan momenteel niet de range van 5 tot 24 Volt behalen. Momenteel is de maximale spanning 21,6 Volt. Dit heeft een herontwerp en/of verder onderzoek nodig.

Conclusie en aanbevelingen

In conclusie wordt er vooral gekeken naar de lijst met eisen die eerder opgesteld is. Hier wordt gekeken naar welke onderdelen er volgens de test werken, wat er nog getest moet worden en wat er helemaal niet werkt. Hieruit is te concluderen dat een groot deel van het project nog verder getest moet worden, of verder ontwikkeld moet worden. Deze handelingen moeten eerst uitgevoerd worden voordat het project gezien kan worden als functioneel.

Door middel van deze conclusie wordt direct een aanbeveling gegeven. De aanbeveling voor nu is het afmaken van het project door middel van een lijst met gegeven activiteiten die uitgevoerd moet worden. Bijvoorbeeld het nog meten van de PT100 ingang, het schrijven van de communicatie software, etcetera.

Een volledig overzicht met welke handelingen er nu aangeraden worden en welke onderdelen momenteel wel of niet aan de eisen voldoen is te vinden in hoofdstuk 10

Contents

List of Figures

List of Tables

Glossary

Can-Bus A robust vehicle bus standard designed to allow microcontrollers and devices to communicate with each other's applications in real-time. 14, 15

Git A distributed version-control system for tracking changes in source code during software development. 34

IO-Link A communication protocol to read out sensors digitally. 12, 13, 15

Lin-Bus A robust vehicle bus standard designed to allow microcontrollers and devices to communicate with each other's applications in real-time. 14, 15

ModBus A serial communication protocol to interface sensors mostly used in PLC's. 14, 15, 18, 19

nRF A micro-controllers Series from Nordic Semiconductors. Focussed on low power applications. 29, 34, 46

RS232 a standard defining the electrical characteristics of drivers and receivers for use in serial communications systems. 14, 15, 18

RS485 a standard defining the electrical characteristics of drivers and receivers for use in serial communications systems. 14, 15, 18

Xe-Link The device to be designed during the Internship, a Sensor Interface Board that can connect to IoT Gateways. 4, 9, 11, 12, 14–21, 24, 25, 29–31, 33, 34, 40, 41, 44–47, 49, 51, 60–64

Xeelas The company where the Internship is held, the company is focused on the use of the Internet of Things in the industrial field. 7–9, 11, 12, 14, 18, 28–31, 34, 51

Acronyms

ADC Analogue Digital Converter. 12, 15, 26–29, 37, 38, 45, 47, 49

BoM Bill of Materials. 10, 33, 63

CRC Cyclic Redundancy Check. 17, 18

GPIO General Porpuse Input Output. 34, 39, 42

IC Integrated Circuit. 21–25, 31, 37, 42

IoT Internet of Things. 7, 11, 12, 16, 18, 20, 41, 49

JSON JavaScript Object Notation. 18, 19

LDO Low Dropout. 20, 21, 28, 29

LPP Low Power Payload. 18, 19

MQTT Message Queuing Telemetry Transport. 18, 19

PCB Printed Circuit Board. 10, 30, 32, 33, 63

PLC Programmable Logic Controller. 14

PWM Pulse Width Modulation. 22, 23, 29, 34–36, 42, 44

RTU Remote Terminal Unit. 14, 18

SAADC Succesive Aproximation Analogue Digital Converter. 37, 39, 45–48

SCPI Standard Commands for Programmable Instruments. 17, 19

SDK Software Development Kit. 34, 35, 38, 40, 46

SSI Simple Sensor Interface. 17, 19, 34, 51

UART Universal Asynchronous Receiver-Transmitter. 9, 11, 12, 15–18, 24, 25, 29, 30, 49, 51

1 Introduction and background

This document describes the development process performed to design, make, program and test the Xe-Link sensor breakout board. By giving a brief introduction an overview of the background behind this project and where the problem initially originated from is given.

1.1 Company

Xeelas is a start-up company, founded in 2015, with a focus on IoT devices in the industrial and build-sector. Providing several IoT devices and services that can measure and locate certain industrial processes or assets.

1.2 Products and services

With its 8 employees, the goal of the company is to widen its market range by improving its own products and services. Xeelas has several hardware devices available to connect to the cloud using their portal, the hardware and portal will be explained briefly below.

1.2.1 Hardware

The hardware produced by Xeelas is built to communicate with the internet at locations you normally cannot communicate with(e.g. construction fields). The Xeelas hardware is able to read out several sensors to read out industrial processes, while it is connected to the cloud. This hardware is working together with the portal explained in chapter 1.2.2.

Xeelas provides three different hardware solutions to hook up sensors to the cloud:

1. Xeelas Node S (as seen in figure 1)
2. Xeelas Node G IP67
3. Xeelas Node G IP42

Figure 1 – Xeelas IoT Node S (Solar-series)

1.2.2 Portal

The portal Xeelas provides is the place that our customers come and check on their assets that are connected using our hardware. In the portal aspects like the location, status, measured sensor are all listed and can be checked by our customers. The customer can also set alarms for themselves to note when an asset is reaching a certain threshold (e.g. low fuel), the customer will then get a notification about this alarm. A screenshot of the portal is given in figure 2, to give a clear view on how the portal looks for our customers.

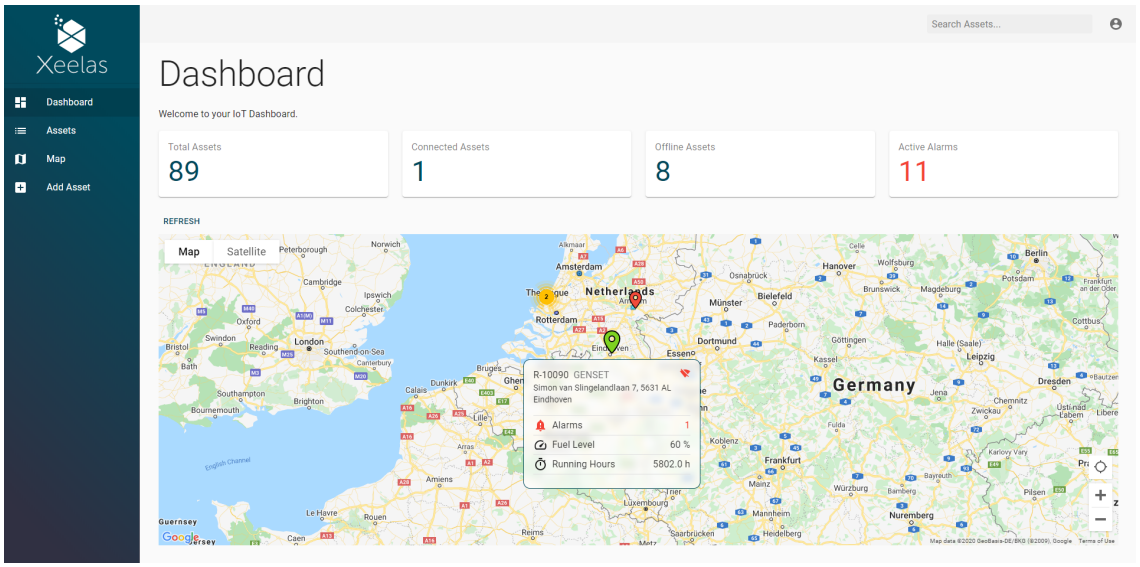


Figure 2 – Xeelas Portal, Source: Xeelas [1]

1.3 The project: Xe-Link

Currently the Xeelas hardware is able to support just a few sensors using specific communication protocols. This is resulting in a small choice of sensors, again resulting no possible sensors or far more expensive sensors than necessary. By designing an adaptor that is able to convert a standard protocol like Universal Asynchronous Receiver-Transmitter (UART) into any often used sensor protocol, the Xeelas products can be used with cheaper sensors, that are widely more available than what the choice was before. Which eventually will benefit in the market range of Xeelas.

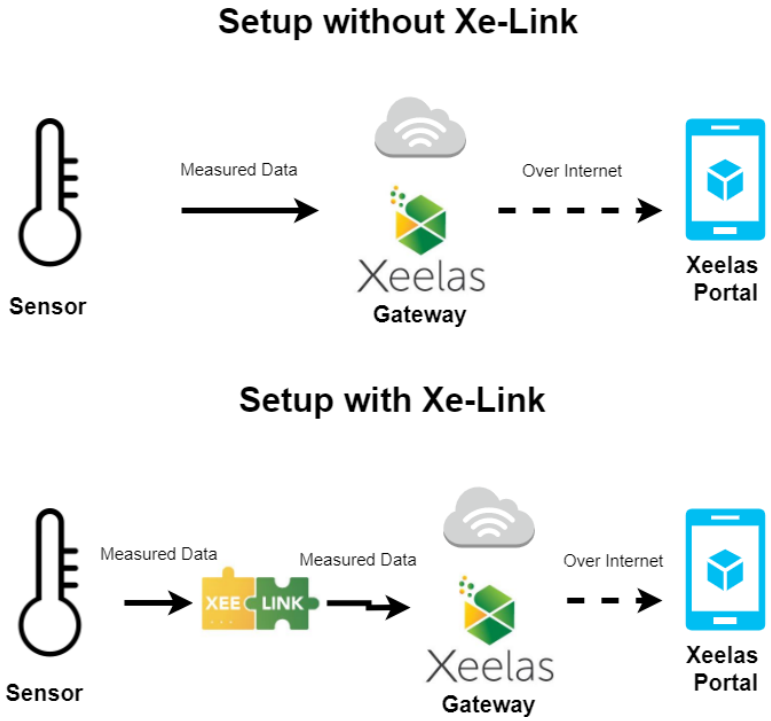


Figure 3 – Setup Xeelas system with and without xe-link

1.4 Given example: fuel and location tracking of aggregates

One of the ongoing projects at Xeelas is a customer that rents out aggregates, the customer needs to refuel on time and keep track of the aggregates status to keep their customers happy. By installing the system Xeelas is providing, the company is able to track the amount of fuel of the aggregates at any given moment and see if the machines are still working properly. Reducing the cost of refuelling the machines when it is not needed and performing services only when they need to be done.

Where does Xe-Link come into play in this scenario?

Xe-Link is able to add more types of sensor protocols that are able to measure the fuel level or other needed measurements. The current system Xeelas is providing, only supports a few different sensor protocols, where sensors that use these protocols can be expensive or don't even exist. This can be solved by adding the Xe-Link in the loop and giving the option to use cheaper, better and more sensor options.

2 Method and plan of approach

To get a clear overview on what the planning is for this project, figure 4 has been set up. Where each step holds several sub-steps. The chart simply gives a rough view on the timeline of the project, each step is explained in depth below the figure

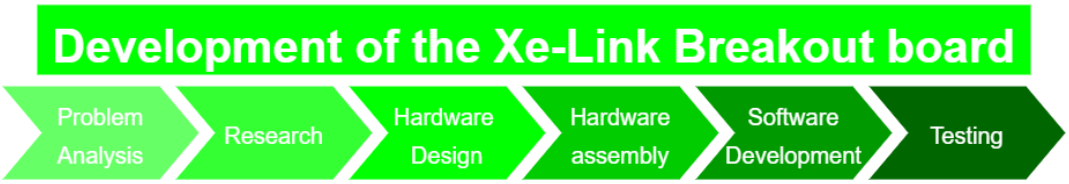


Figure 4 – Development Process Timeline

Here-under, all six given steps will be explained briefly. Whenever all these steps have been performed, a new list of improvements and encountered problems can be set up. This list will be given in the conclusion of this document in chapter 10

Problem Analysis
During the problem analysis phase, a clear overview of the project will be drawn, listing the problems and subjects to research.

Research
When the problems to research are clear, all problems have to be researched thoroughly on possibilities and what the eventual best choice is. For example, the communication protocol must be chosen, by comparing which protocol fits the project the best.

Hardware design
When the research has ended, a list of choices is provided. Using these choices, a hardware design can be made. The hardware design is separated in different parts:

1. Selecting components
2. Running simulations
3. Designing schematics
4. Designing Printed Circuit Board (PCB)

Hardware assembly
When the Hardware design phase is finished, the PCB can be ordered, a Bill of Materials (BoM) can be generated and ordered as well.

When the PCB and components arrive, the project can be assembled. If any problems occur when assembling the project, these problems will be documented and solved

Software development
When the hardware is ready for use, software can be written for the board to program the board. The software will be programmed to the board to interact with the hardware.

Testing phase
The combination of well written software and well designed hardware will result in a perfect project. However, to be certain that everything works as intended and has no bugs, errors or malfunctions the project has to be tested thoroughly.

A list of tests has to be set up to perform, these test will specify whether the functionality of the board needs improvements. Resulting in a revision. With the help of the earlier written list of requirements, the project can be validated whether it is functional or not.

3 Problem analysis

To further investigate the problem, an analysis has been done. This analysis gives a clearer image about the problem and how to solve it easier. This is done by dividing the problems in multiple sub-problems.

To give a clear picture of the problem, a system overview has been made in chapter 3.1

3.1 System overview

To give a clear picture figure 5 has been drawn. This figure will inform and visualise the connections with the Xe-Link sensor interface board.

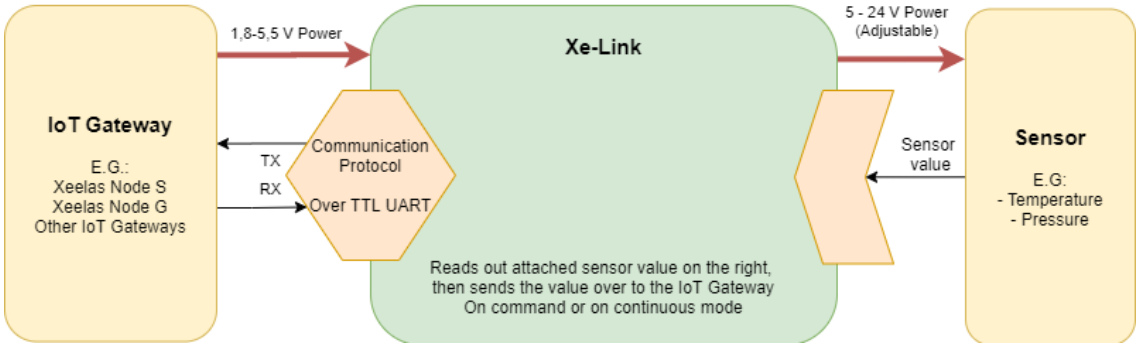


Figure 5 – Overview Xe-Link

The overview consists of three different parts, the connected gateway on the left, the Xe-Link breakout and the connected sensor on the right. Thus, the Xe-Link is connected with two different devices, information about these connections is given below:

IoT gateway
The IoT gateway is a gateway connected to the internet. This can be any gateway from Xeelas or any other companies.
To communicate with the Gateway, a connection over UART has to be established, this has to be a reliable connection that can be supported by both the Xe-Link and IoT Gateway. The protocol should be easy to implement for others using the Xe-Link so that is easy to adapt in other projects. To achieve these requirements a correct communication protocol has to be chosen.

Sensor
As stated before, the Xe-Link sensor interface has to read out sensors so that it can send values back to the gateway. To increase its value in the field, the board needs to be able to support sensors that are used often in the field. To increase the amount of sensors that are capable of connecting to the Xe-Link, multiple sensor protocols can be selected. These sensors need dedicated hard and software to communicate and need a power supply from the breakout box, which needs to be able to provide this power. Both analogue and digital sensor will be considered, where the possibility is to make separate versions of the Xe-Link that support certain types of sensor protocols.

3.2 List of Requirements

To fully give an understanding of what the system must, should or could be capable of a list of requirements has been set up. As seen in Appendix A. This list of requirements will be used while researching and making choices that need to be performed during the project.

4 Research

After defining what the exact problems are, several problems have to be researched in depth. To make sure the Xe-Link interface is able to support useful sensors, a research in both analogue and digital sensors will be done. As well as an in depth look at the communication protocol the system has to support.

4.1 Sensor compatibility

As stated in the problem analysis (Chapter 3), the supported sensor protocol has to be chosen. The sensor protocol has to be a standard sensor protocol that is being used often in the industrial market, so that the Xe-Link will be able to support as many sensors as possible. With this in mind a few different types of sensors have been considered.

Hereunder, a list of optional sensors that have been researched on their pro's and con's is given. Resulting in a chosen sensor protocol to be supported by the Xe-Link.

4.1.1 IO-Link

The initial idea of the Xe-Link was to create a sensor interface that converts the IO-Link protocol into a UART based protocol.

IO-Link is a standard communication protocol supported by dozens of companies that replaces the old analogue sensor equipment, like 4-20 mA and 0-10V protocols, making it digital, configurable and easily replaceable. The IO-Link sensors used with this communication protocol consume high amounts of power and have long startup time since they're mainly built for industrial purposes. Looking at the application of the Xeelas Node Gateways, who are IoT nodes running on small solar panels, the high power consumption is a big disadvantage for using IO-Link.

After some more investigation on how to set up IO-Link masters to interface and communicate with these IO-Link sensors a paper concluded that being a vendor of the IO-Link community was an obligation before putting IO-Link Masters or sensors on the market, as well as getting enough information to research and develop such IO-Link devices. The costs to become a vendor were €3000 euros per year [5], which was to high for Xeelas to pay for in the first place for a start-up idea like the Xe-Link.

Both the power consumption [6], start-up time and vendor costs have been mostly the reason to start looking at different protocols to read out and measure other Sensors that are usable in the IoT market.

4.1.2 Analogue sensors

Next to the too expensive and power intensive IO-Link sensors are older analogue sensors. These sensor exist for over 30 years and are still widely supplied. There are different types of sensors in this category:

- 4-20 mA sensors
- 0-10 V sensors
- PT100/PT1000 temperature sensors

The above given protocols are still in use and are very well supplied, resulting in a great perspective on market adaption. To fully understand what the power consumption of these sensors is, a list of 10 of each sensor has been set up with their power consumption, startup time and costs. This list has been set up by randomly visiting stores that provide these kind off sensors (with a wide variety of different types of sensor) and writing down their specifications.

4 - 20 mA Sensors
A 4 - 20 mA sensor generates a current source that changes current with the measured value of the sensor. Using the current as percentage of the maximum value of the sensor. The 4 mA minimum is set as the 0% point to differentiate 0 % and disconnected/malfunctioning from each other. Example given: a temperature sensor that measures from 0 to 100 degrees, where 4 mA is 0 % and thus 0 degrees. And 20 mA is 100% and thus 100 degrees.

This sensor type can easily be read out by measuring the voltage over a precise shunt resistor. The hardware to make this possible is not expensive or hard to set up since most micro-controllers have built in Analogue Digital Converter (ADC)'s.

To verify the power consumption of these 4 - 20 mA sensors, as spoken of earlier. A list of 10 different sensors from different manufacturers has been set up. This list can be found in table 1, where the power consumption is 0,45 Watt at their maximum power and goes as low as 0 Watts when self-powered. The power consumption of these sensors compared to the IO-Link sensors spoken of in chapter 4.1.1 are significantly lower.

	Vsupply (V)	Current (I)	Power (W)	Function	Manufacturer	Part. Nmbr.
1	15-30	0,03	0,45	nearing-sensor	Contrinex	DWAD 509 M30 390
2	15-30	0,03	0,45	Induction-sensor	EATON	E59-A12A104D01-C1
3	20-30	Self-powered	0	Current sensor	Phoenix	MCR-SL-CUC-100-I
4	10-30	0,03	0,3	Pressure sensor	BB sensors	DRTR-ED-XX_A
5	10-30	0,03	0,3	Light sensor	Sick Sensor	WL11-2P2430 — W11-2
6	8-32	0,03	0,24	water-level sensor	Sensors One	LMK306
7	8-30	0,03	0,24	Gas Pressure sensor	Sick Sensor	PBT-RB250SG1SSNAMA0Z
8	15-30	0,03	0,45	Current sensor	Hobut	CT132TRAN50A-4/20MA
9	12-80)	0,03	0,45	pH sensor	Newport	PHTM-014
10	8-30	0,03	0,24	hydropress sensor	Wika	12719383

Table 1 – List of different 4 - 20 mA Sensors and their power consumption

0 - 10 V Sensors

The 0 - 10 V Sensors work on the same principle as the 4 - 20 mA sensors spoken of above. The 0 - 10 V sensors have 3 wires, where 1 is the Vsupply, one Ground and the third an analogue signal, this signal is a voltage between 0 and 10 Volts that tells the level of the sensor. Example given: Temperature sensor at 0 Volts is 0 % of its range, and when 10 Volts at 100 % of its range.

It is possible to combine the hardware of the 4-20 mA and 0 - 10 V sensors in one single analogue system, resulting in lower cost and a smaller component population space on the PCB. This will result in decreasing cost and size of the board.

For the 0 - 10 Volt sensors a table has been set up the same way as before with the 4 - 20 mA sensors. In this table the power consumption is listed to see the minimum, maximum and standard power consumption of the regular sensors. In the case of the 0 - 10 V sensors the power consumption is about the same as the 4 - 20 mA sensors.

	Vsupply (V)	Current (I)	Power (W)	Function	Manufacturer	Part. Nmbr.
1	15-30	0,03	0,45	nearing-sensor	Contrinex	DWAD 509 M30 390
2	15-30	0,03	0,45	Induction-sensor	EATON	E59-A12A104D01-C1
3	12-30	0,005	0,06	Pressure sensor	BB sensors	DRTR-ED-XX_A
4	14-30	0,025	0,35	Pressure sensor	Sensor One	DMP 331
5	Self-powered	Self-powered	0	Current sensor	LEM	AT 10 B10
6	24V	0,07	1,05	Micropulse sensor	Balluf	BTL6-A110-M0800-A1
7	15-30V	0,003	0,045	Temperature sensor	Schneider Electric	MTN663596
8	4.5 - 24	0,01	0,045	Flow sensor	RS Pro	511 -4772
9	24	0,0012	0,018	Wind sensor	Gira	2143170

Table 2 – List of different 0 - 10 V Sensors and their power consumption

PT-100 & PT-1000 Sensors

A PT sensor is a simple resistor that change their resistance according to the temperature. Since these sensors are simply resistors, they are able to run on low voltages, resulting in less power consumption. These sensors, however, need a specified piece of hardware to be read out correctly and thus are not suitable to plug in the same circuit as the 4-20 mA and 0-10V inputs.

4.1.3 Digital sensors

There are a lot of other useful digital sensors that are used in the industrial but also the automotive sector. A few sensors have been considered to be used in the Xe-Link design:

- Mod-Bus RTU protocol Sensors (RS232 and/or RS485)
- CAN-Bus
- LIN-Bus

Mod-Bus Mod-Bus is sensor protocol used mostly in the field of PLC's. This protocol is register based and is more then 30 years old. The protocol itself is still being used often and is well supported in software development and sensor availability. Currently, one of the three Nodes(See chapter 1.2.1) from Xeelas supports the ModBus RTU protocol over both RS232 and RS485

The ModBus RTU is meant to run on a specified hardware-layer, RS232 or RS485. These pre-defined hardware layers are meant to prevent distortion of the messages, resulting in a more reliable communication over longer distance cables. This, however, requires specific hardware.

Can-Bus Can-Bus is a protocol mainly used in cars or other vehicles. It is possible to tap into cars using the standardized software layer. By tapping into the Can-Bus, information about certain vehicles can be obtained. For example the speed that it is driving, or any warnings that pop up during running-time. Can-Bus often works together with a Lin-Bus as spoken off here-under or seen in figure 6

Lin-Bus Can-Bus is a protocol mainly used in cars or other vehicles. A Lin-Bus is mostly used in combination of Can-Bus, where the Lin-Bus controls sub-systems of a vehicles systems. The configuration is clarified using figure 6. In this figure, the dark-grey lines are the Can-Bus and the blue lines are the Lin-Bus. In this figure is clearly seen that the main bus is controlled by the Can-Bus and the sub-systems are controlled by the Lin-Bus.

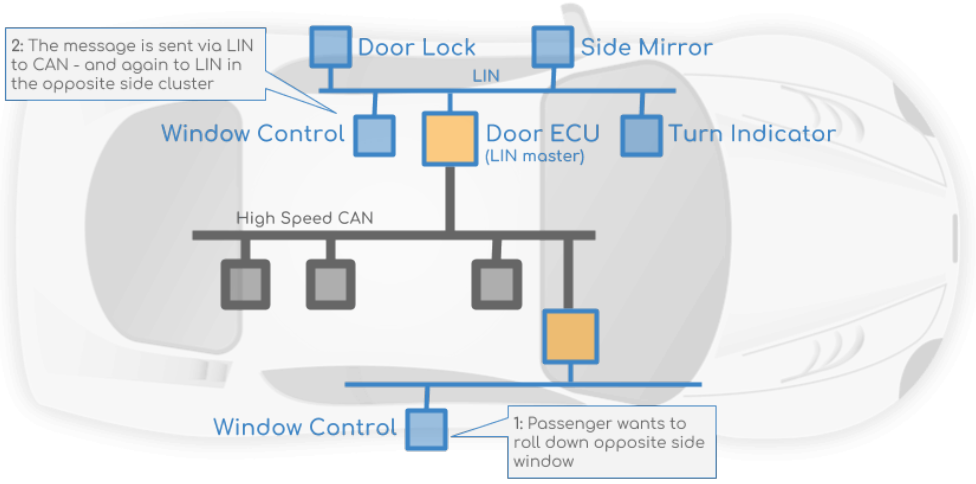


Figure 6 – Overview of Can and Lin together in one system Source: css-electronics [2]

4.1.4 Chosen sensor compatibility

After getting more in depth in the different available sensors, where some have been researched more in depth than others, a conclusion has to be given on what sensors are going to be supported by the Xe-Link.

Since a lot of sensor protocols need specific hardware that has to be included on the board, choices have to be made. The most promising protocols were:

1. Analogue sensor: 0 - 10 V
2. Analogue sensor: 4 - 20 mA
3. Analogue sensor: PT100/TP1000
4. Digital sensor using: RS232, RS485 using ModBus
5. Digital sensor using: Can-Bus
6. Digital sensor using: Lin-Bus

The IO-Link sensors were too expensive and power intensive to be added to the Xe-Link sensor adaptor. A clear list of analogue protocols has been chosen, with three different most used protocols.

Since there is not enough room on the board to fit both the analogue and digital protocols the choice has been made to separate the Xe-Link into two different versions: The Xe-Link Sense and the Xe-Link Serial. Where the Xe-Link Sense will be able to support all the listed analogue sensors above, and the Xe-Link Serial supports all the digital protocols. A small description of both versions is given below. And is seen in the sales pitch overview given in figure 7.

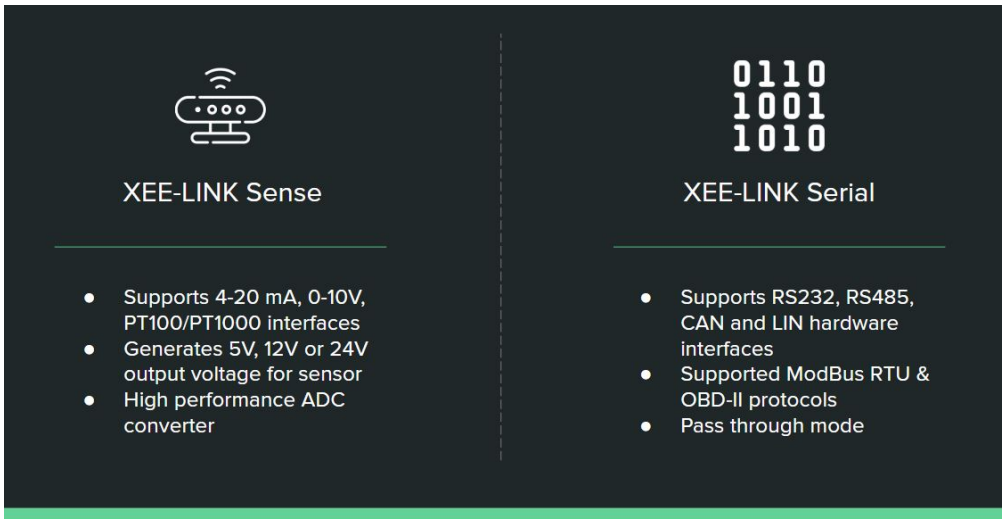


Figure 7 – Development Process Timeline

Xe-Link Sense:
As seen in figure 7, the Xe-Link Sense is capable of measuring 4 - 20 mA, 0 - 10 V and the PT100/PT1000 sensors using well constructed ADC. The Xe-Link Sense is also able to provide a supply voltage between 5 and 24 Volts to power the sensors.

Xe-link Serial:
The Xe-Link Serial supports RS232, RS485 Can-Bus and Lin-Bus. The pass through mode as stated in figure 7 is a mode that sends messages over UART to another device, which makes it possible to Daisy-chain multiple devices.

During this internship we will mainly work on the Xe-Link Sense, since there is more information about this versions requirements and possibilities. A few things that are currently being considered are also useful for the Xe-Link Serial. Example given: The communication protocol with the gateway, this will be implemented in both the different versions.

4.2 Gateway communication protocol

As stated in chapter 3.1, a communication protocol has to be chosen to use. This communication protocol has to be able to set up a communication link between the IoT Gateway and the Xe-Link. Before choosing which protocol suits the best, a small UML Use Case diagram has been made to give a better understanding of what the system will be doing. A quick overview of the UML Use cases can be found in figure 8

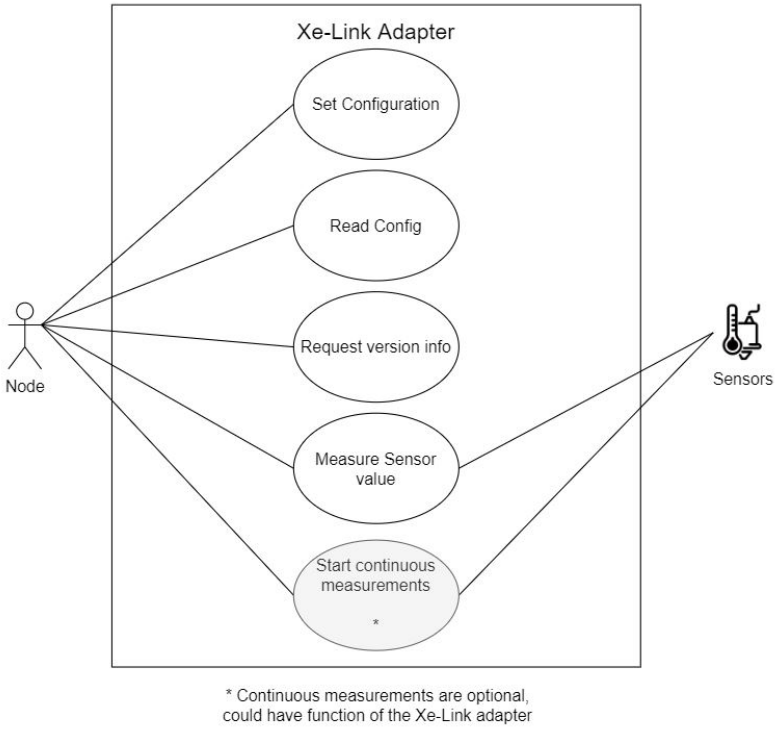


Figure 8 – UML use-case diagram

As seen in figure 8 five use cases are shown, these use cases are needed for the Xe-Link to function as intended. All use-cases are triggered by the Node, which is the IoT gateway that is connected over UART. The communication protocol must support all commands that are set in figure 8. Using this diagram a list of minimal functionalities of the communication protocol can be set up:

- Must be able to send messages in both directions
- Must be able to support messages containing changes to the configuration
- Must be able to send requests and later get a reply (also called query's)
- Should be able to stream data continuously or on timed events

Using the given requirements a small list of different communication protocols has been selected for further research, where each protocol has its pro's and con's. All protocols will be explained and gone over in depth below.

4.2.1 Simple Sensor Interface (SSI)

The Simple Sensor Interface (SSI) is a protocol designed to fulfil all the needs for basic sensors. With the focus on sensors the protocol is kept as simple as possible and fits very well with the Xe-Link. SSI is not used very often, which result in limited support. SSI is fully defined, with a set of pre-defined commands. All pre-defined commands match the Xe-Links specifications. SSI is built to work on a UART hardware layer, resulting in no needed changes to the hardware or software. This includes a pre-defined Cyclic Redundancy Check (CRC). [7]

Pro's:

- Predefined CRC
- Meant to run on UART, also capable of running on other hardware-layers
- Matches commands with Xe-Link functionality
- Bi-directional communication
- Able to send configuration settings to the Xe-Link sensor readout side
- Can stream data continuously

Con's:

- Limited support available

4.2.2 SCPI

Standard Commands for Programmable Instruments (SCPI) is a protocol commonly used in electronic bench measurement equipment, for example oscilloscopes. This protocol is able to run on any serial communication hardware layer and is easily readable by humans. This protocol is mainly built to establish connections between lab-bench measurement equipment and computers easily. The master side of the SCPI protocol (Computer) is easily programmable, the slave side however is harder since it has to fulfil a list of certain requirements set up according to the SCPI standards according to its guidelines [8].

Pro's:

- Human readable
- Frequently used
- Bi-directional communication
- Can stream data continuously
- Able to run on UART

Con's:

- Unspecified CRC
- Not really meant for Xe-Links application
- Hard to implement on Xe-Links firmware side

4.2.3 ModBus RTU

ModBus RTU is an industrial communication protocol, spoken of earlier in chapter 4.1.3 as sensor protocol to support. In this particular situation (as communication protocol between the Node and Xe-Link) it is possible to run ModBus on UART as well. This protocol is a register based and old. The protocol is still being used often in industries, with a preferred hardware layer like RS232 or RS485 to reduce interference. ModBus RTU is already being used in the software of the current Xeelas Nodes, resulting in an easy to implement protocol on the Nodes side. [9]

Pro's:

- Already implemented in Xeelas Nodes
- Lots of support and documentation available
- Pre-defined CRC
- Able to run on UART, although meant to run on RS232 or RS485

Con's:

- Not human readable

4.2.4 Cayenne LPP

Cayenne Low Power Payload (LPP) is a specially designed protocol for IoT sensors. This protocol is still in development and improving every month, this however makes the protocol unreliable and lacking in functionality. The goal of Cayenne LPP is to make low power IoT devices run on as low power as possible, resulting in an as light weight as possible system. Cayenne LPP however is also able support systems that the Xe-Link does not need (e.g. GPS). [10] **Pro's:**

- Able to run on UART, no pre-defined Hardware-layer
- Low power focused
- Lots of documentation available, including a community able to help out
- Pre-defined CRC

Con's:

- Still in development, chances on bugs and imperfections
- Limited to certain value types (like temperature)

4.2.5 JSON

JavaScript Object Notation (JSON) is a data format that is openly used for lots of different systems. This data format can be used while sending data over Serial communication and is highly configurable. This can be taken in advantage for the Xe-Link communication protocol when set up correctly. The process of setting this system up does take time and effort, instead of chosing a more pre-defined protocol. [11] **Pro's:**

- Highly configurable, shape-able to this situation
- Lots of support and big community behind this system

Con's:

- Takes more time to set up than pre-defined protocols
- No pre-defined CRC

4.2.6 MQTT

Message Queuing Telemetry Transport (MQTT) is a protocol mainly used in home-automation. This protocol needs a MQTT broker to work correctly and is only one-directional. [12]

This protocol is not suitable as communication protocol.

4.2.7 Chosen communication protocol

Now that a list of communication protocols have been investigated, a list of potential and non-potential communication protocols can be made. In the list below, the potential protocols can be found that fit the best to the project, the other not listed protocols are not as potent to be chosen and are taken out of the list of protocols that can be chosen.
Potential communication protocols:

- SSI
- SCPI
- ModBus
- Cayenne LPP

JSON and MQTT are not suitable enough for the project. JSON due to its time and effort it needs to be fully specified and fitted to the project. MQTT because of its uni-directional system and needed broker server.

From the list of the 4 most potent communication protocols, cayenne LPP is the must unreliable because of its under-develompent state. Thus not the best choice for the Xe-Link configuration, Leaving three potent protocols. These protocols will get a score on different requirements of the system:

1. How much does the protocol fit to the application? (scale: 1 - 10)
2. How easy is it to set the protocol up in this application (scale: 1 - 10)
3. Is the protocol human readable (Scale 0 - 1)
4. Is there pre-existent software that can be used (Scale 1 - 10)

In table 3 the three protocols can be seen with scores on each requirement.

	SSI	SCPI	ModBus
Fits application	9	6	7
Easy to setup	7	5	7
Human readable	1	1	0
Pre existent software / supoprt	7	6	8
Total score	24	18	22

Table 3 – Table of potential protocols with functionality scores

In table 3 a clear overview is given. With SCPI as the least potential of the three protocols, where SSI and ModBus are more useful for the project. SSI however is a better choice due to its fitting application, and thus will be chosen as communication protocol.

5 Hardware design - schematics

The research phase has ended, resulting in a clear list of what the Xe-Link has to be capable of and what protocols and sensors it will be able to support. In this document, the main focus will be that of the Xe-Link sense, whom focuses on the analogue sensor inputs. Before starting with the hardware design, a new overview has been drawn of the inside of the Xe-Link Sense to give a clear overview of what sub-systems the Xe-Link Sense has to contain to work properly. The overview is seen in figure 9.

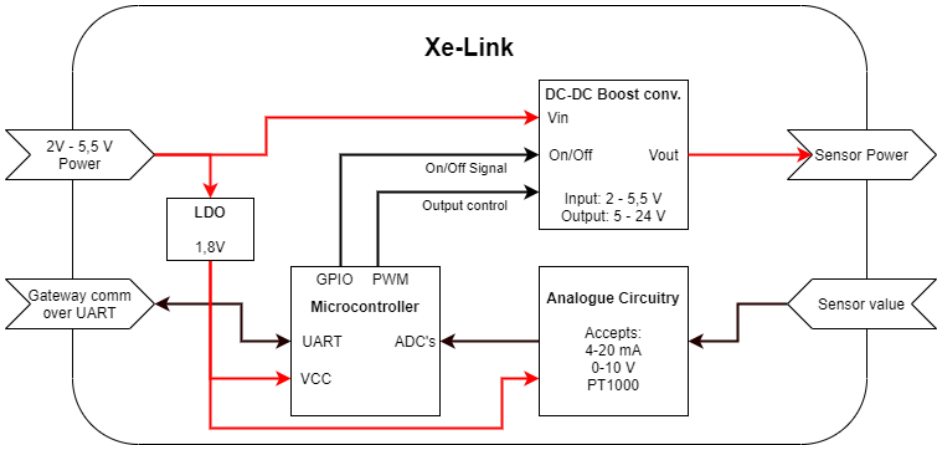


Figure 9 – Internal hardware overview Xe-Link

In this figure it is clear that there are several sub-systems to be developed:

1. Power - LDO
2. Power - Boost conv.
3. EMI/EMC suppression (not seen in figure)
4. Analogue inputs (4-20 mA and 0-10 V combined)
5. Analogue input of the PT1000/PT100
6. Microcontroller
7. UART level shifting (not seen in figure)

Each problem listed above is tackled separately, creating subsystems that are able to work together. Each subsystem is explained thoroughly in the sections below.

All the schematics can also be found in appendix B.

5.1 Power

The Xe-Link is powered by an IoT Gateway that provides a supply voltage between 2 and 5,5 Volts. This input voltage will be used and brought to different levels:

1. 1,8 Volts for the internal components like micro-controller
2. 5 - 24 Volts to power external sensors

The 1,8 Volt rail can be achieved with the use of a Low Dropout (LDO) regulator with a low dropout voltage. The 5 - 24 Volt output must be achieved by a boost-converter. The output voltage needs to be controllable by software, thus resulting in a more complicated system than regular.

5.1.1 Internal LDO regulator

The LDO regulator given has an input voltage between 2,0 and 5,5 Volt and an output voltage of 1,8 Volts. The input requirement sets a required maximum dropout voltage of 200 milli-Volts. The maximum current the LDO regulator needs to provide is very low, thus every LDO regulator with a current of 50 mA or more suffices. Here under a list of the requirements for the LDO regulator will be given to give a clear overview:

1. Minimum input voltage must be 2,0 Volts or lower
2. Maximum input voltage must be 5,5 Volts or higher
3. Maximum output current must be 50 mA or higher
4. Maximum dropout voltage must be 200 mV or lower

Using the above requirements a LDO regulator has been found. The TLV70018DCKR LDO fitted the best to these requirements according to its datasheet [13]. The maximum dropout voltage is 175 mV on the worst occasions, which is suitable for our application.

The schematic of the LDO regulator can be found in figure 10.

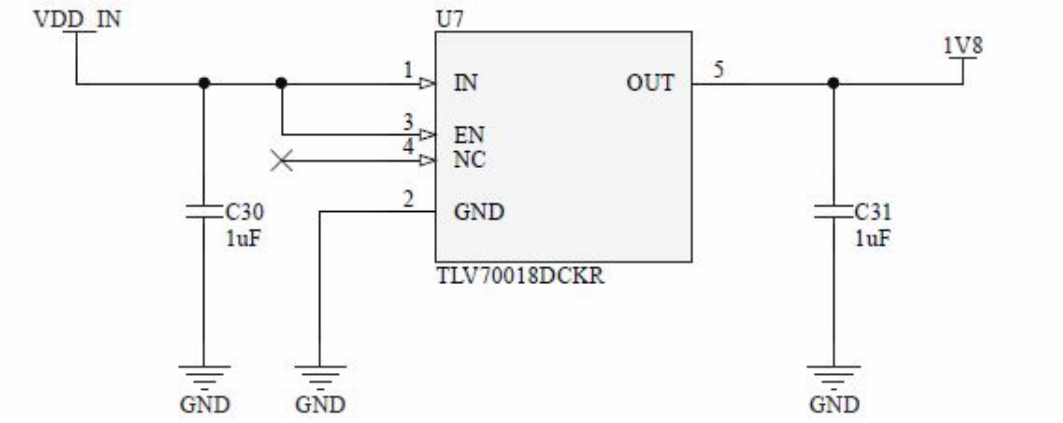


Figure 10 – 1,8V LDO circuit

In this figure, the simplicity of the LDO regulator can be seen, with just one 1 uF on the input to stabilize the input voltage. And a 1 uF capacitor on the output for the same effect on the output voltage. The enable pin of the LDO regulator is tied to the input voltage to let the LDO regulator always be on.

No further simulations for this circuit are necessary.

5.1.2 Boost converter

To power the external sensors attached to the Xe-Link, a boost converter has to be designed. According to the research done earlier a list of requirements for the boost-converter can be given:

1. Must be able to produce a variable output voltage between 5 and 24 Volts
2. Must be able to provide a current of 50 mA or higher at any set Voltage
3. Has an input voltage between 2,0 and 5,5 Volts

With the help of the Texas Instruments Webench tool [14], where the requirements can be filled in. A boost converter Integrated Circuit (IC) has to be chosen. To keep the board size as small as possible, it is preferred to choose a switching boost converter with integrated mosfet to keep the amount of components as low as possible. The chosen IC is the TLV6104 boost converter IC, which is capable of boosting from 1,8 Volts to 24 Volts.

The boost-converter circuit can be seen in figure 11

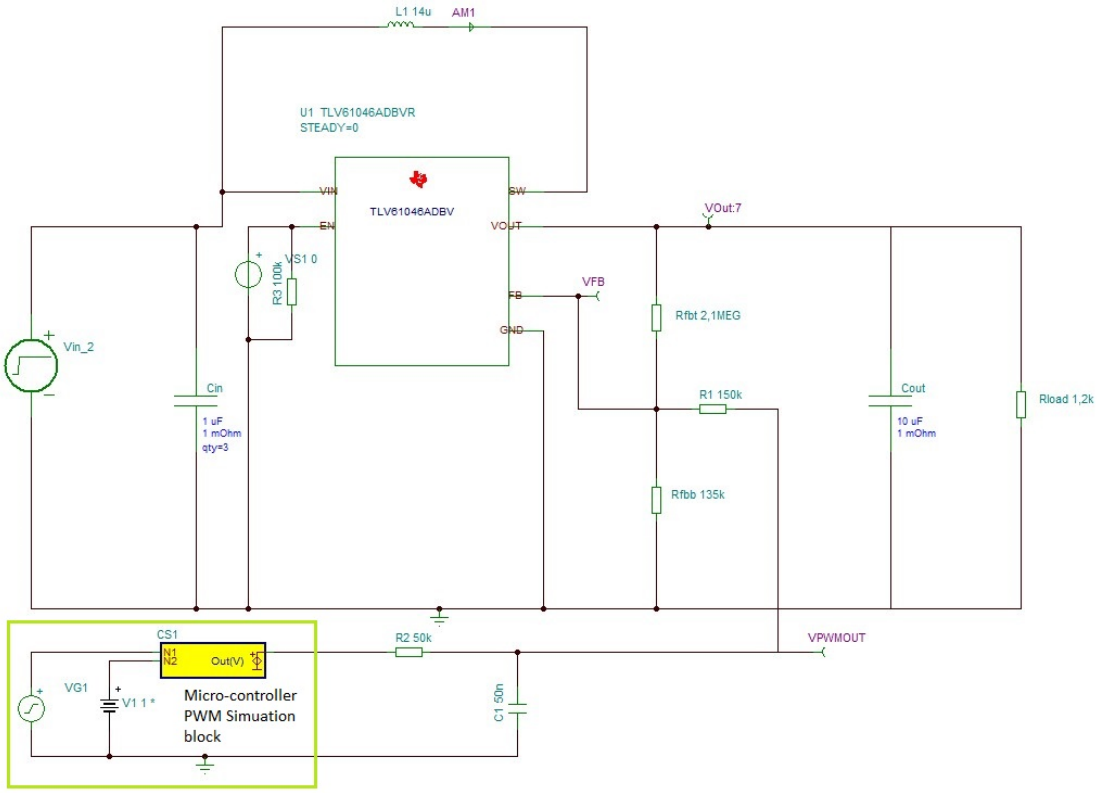


Figure 11 – Boost-converter Circuit with PWM input

Adjustability feedback-pin

To be able to change the output Voltage of the boost-converter, a simple trick has to be performed. Some converter IC's have a function built in to control the output Voltage, the type of boost-converter we had to choose did not have the option to add this. Thus resulting in a trick that is needed to still get the job done.

As seen in figure 11, the yellow squire represents a Pulse Width Modulation (PWM) output of a micro-controller. This signal is smoothed into a DC voltage by a Resistor and Capacitor low-pass filter. A rough formula of the ouput voltage of this filter can be seen in formula 1. Where PWM is a value between 0 and 1 according to its percentage on/off (75 % on will result in a multiplication of 0,75).

$$V_{PWMOUT} = 1,8 * PWM \tag{1}$$

This smoothed Voltage can alter the voltage on the Feedback pin of the IC, that normally measures the voltage over a voltage divider of the output voltage. Normally, by changing the voltage dividers resistor values the output voltage of the boost-converter can be changed using formula 2. This formula is provided by the datasheets [3]

$$\begin{aligned} R_1 &= (\frac{V_{OUT}}{V_{REF}} * R_2) \\ V_{REF} &= 0,795\ ,\ R_1 = 2M1 \end{aligned} \tag{2}$$

In this equation, the feedback pins reference voltage always will be 0,795 Volts. R1 is chosen to be high to minimise current leakage. Now that the normal way of approach is done, the feedback control pin using the PWM signal is used to alter the input voltage using an extra resistor as seen in figure 12.

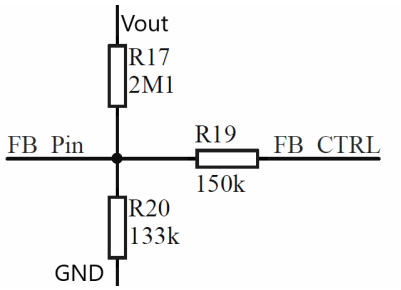


Figure 12 – Feedback control resistor network

By altering the FB_CRL Voltage using the PWM signal, the feedback to the feedback-pin can be changed slightly. A normal setup normally consists of just resistor R17 and R20 as seen in figure 12.

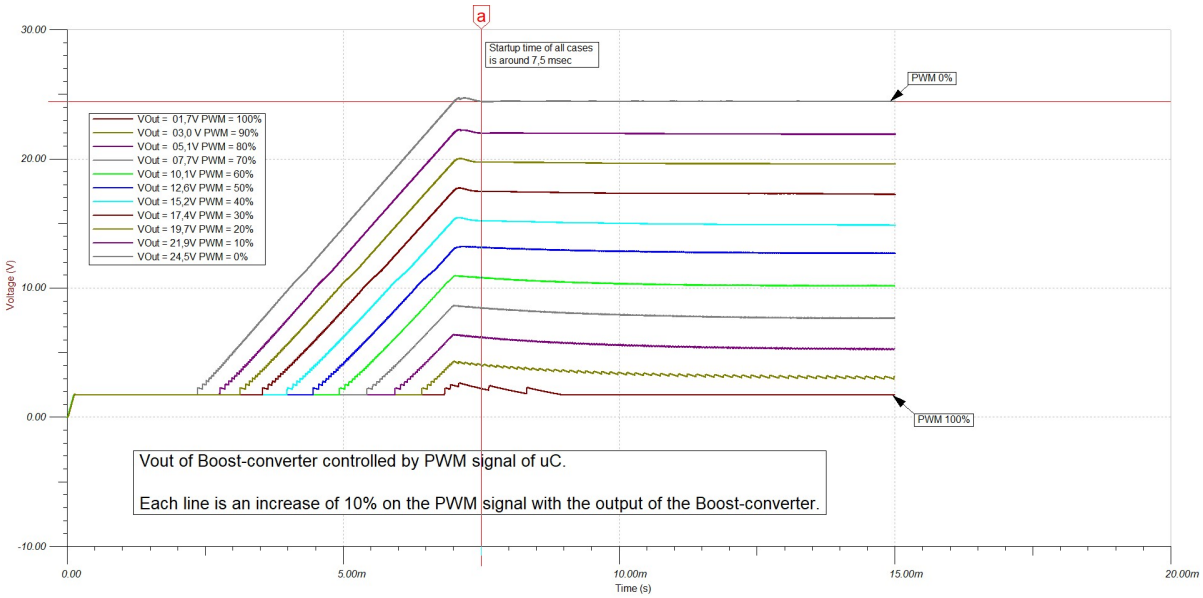


Figure 13 – Simulation of Vout of boost converter with different PWM settings (0-100%)

Using Tina-TI [15] and a spice model of the boost-converter IC, a simulation can be performed as seen in figure 13. In this simulation 10 different runs are plotted. Where in every run the PWM signal from the micro-controller is changed in intervals of 10 %. Where the 100 % PWM signal results in a voltage as low as the input voltage and the 0 % PWM signal results in a 24,5 Volt output. Concluding that the PWM control is working properly.

Turn on/off pin

To be able to turn on and off the boost converter using the micro-controller the Enable pin has to be connected. This is done by directly connecting the enable pin of the boost-converter to the micro-controller. The connection is pulled down by a resistor to set the default state to off.

Inductor choice

According to the datasheet [3] of the boost-converter IC, a few reccomendations have been given. Following these recommendations, we chose for a 15 uH inductor, with a maximum current of 900 mA. Also seen in the circuit in figure 11.

Startup time
Since the Xe-Link needs to be as power-efficient as possible, the startup time of the boost-converter is important as well. Since the longer the boost converter is taking to start up, the longer it takes before the system can go back to sleep. Looking at the simulations generated using Tina-TI, in figure 13, the startup time in all simulations is the same. The startup time of these simulations is 7,5 milli-seconds, which is quite long. However, the only way of lowering this startup-time is decreasing the capacitor at the output voltage, resulting in an unstable output Voltage. An unstable output Voltage has more risks to it than a higher startup time, and thus the capacitor will keep its set value of 10 uF.

The stability of the output voltage is crucial, a bigger capacitor has been considered. If a bigger capacitor has been added, the startup time will be increased significantly, which is not wanted either. In the final schematic design, we have chosen to add space for an optional super-capacitor to increase stability for the sensors for the cost of the startup time.

5.2 EMI/EMC suppression

Since the Xe-Link receives power from a gateway and uses a switching power supply, it is possible that the Xe-Link will produce noise that will be sent to the gateway. To prevent these kinds of problems, a noise suppression design is preferred.

There are different ways of reducing noise on power lines:

1. Common-mode filtering
2. Differential-Mode filtering
3. Using cable chokes
4. Using ferrite beads on the print

A switching power supply mainly produces Differential noise, thus the choice to reduce the noise using a differential-mode filter is the most obvious choice. Next to a differential-mode filter, the level-shifting IC's will be equipped with ferrite-beads to minimise the noise on the UART connection as well.

For the design of the differential filter a simple LC filter has been designed. Using formula 3, where the cut-off frequency (fc) is set to 50 kHz and L1 to 10 uH, a capacitor value of 4 uF came out. The total filter circuit can be seen in figure 14, in this circuit a two Ohm resistor can be found, which simulates the load of the Xe-Link itself.

$$f_c = \frac{1}{\pi\sqrt{LC}} \tag{3}$$

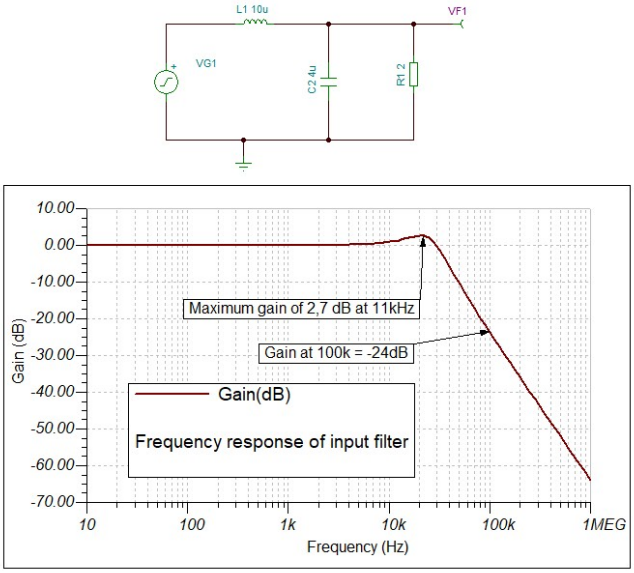


Figure 14 – LC EMI/EMC suppresion filter and simulation

As seen in figure 14, a clear Low-Pass filter is established, with a maximum gain of 2,7 dB at 11 kHz. This maximum gain point is not a crucial point to be filtered, resulting in a not further to be changed circuit.

The boost-converter works on a 100 kHz to 1 MHz switching-frequency, which is all being filtered using the LC filter circuit.

5.3 Level shifting in/output

The UART connection is running on the same voltage as the input voltage of the Xe-Link, thus making it variable and not set. The UART output of the micro-controller is set to 1,8 V and not changing. To let these two systems work together, a level shifting IC is needed. A list of channels and what direction is given below:

1. TX (Output)
2. RX (Input)
3. Interupt Out (Output)

Using this list the easiest choice is to get two IC's, one direction each. Since a pre-set direction is preferred due to reliability. This means that a two channel level switching IC is needed to shift two outputs from 1,8 Volts to the Input voltage and a one channel input level switching IC to do the opposite.

For the one channel IC, the 74LVC1T45GW is chosen. For the two cannel IC, the SN74LVC2T45DCTR is chosen. Both IC's Direction setting have been set by connecting the direction pin to the appropriate voltage or ground according to their datasheets [16] [17].

Both IC's are also supplied with both the 1,8 Volt and input Voltage with a ferrite bead and decoupling capacitor to minimise noise from the supply voltage.

5.4 Analogue input

For the analogue inputs, both the 4-20 mA and 0-10V inputs will be combined, needing less components and room but reducing the amount of sensor usable simultaneously. Using the circuit given in figure 15, both measurements can be performed.

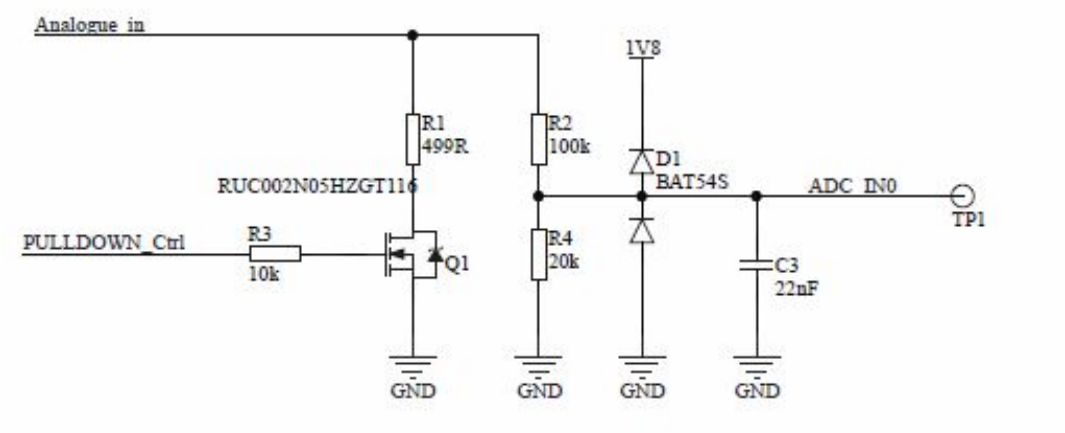


Figure 15 – Analogue input circuit used to measure 0-10 V and 4-20 mA

As seen in this figure a pulldown control pin from the micro-controller is controlling a mosfet. This pin is off (0 Volts) on default, and thus not pulling down R1.

0-10 V

When a voltage between 0 and 10 Volts is applied to the input of this schematic, the pull-down control pin must be set to 0 to get a reliable reading. The analogue input is scaled down by a voltage divider. The formula of the input voltage over the measured voltage at the ADC is given in formula 4.

$$V_{ADC} = V_{A-IN}\frac{100k}{100k + 20k}$$
$$V_{A-IN} = 6 * V_{ADC} \tag{4}$$

Two diodes have been added to the analogue input to prevent damage when an over or under voltage has been applied on the input. Limiting the input at the ADC between 0 and 1,8 Volts.

The circuit has been put in a simulation as seen in figure 16. In this figure it is clear that a good functioning linear line is seen, from 1,8 Volt to 0 Volt with an input voltage of 0 to 10 Volts. Where the input is set on the X-Axis and the output voltage at the ADC on the Y-axis.

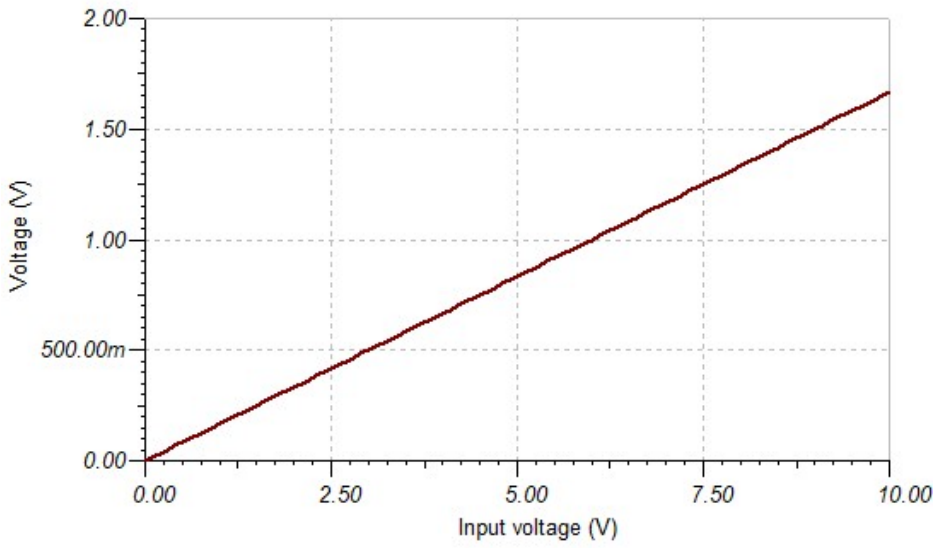


Figure 16 – Simulation of the 0 - 10 Volt analogue input

As seen in this figure, a reliable lineair line, from 0,0 to 1,8 Volt is seen. This is exactly as expected and wanted, and thus working well.

4-20 mA

When a high signal is sent (1,8 V) to the mosfets gate by the micro-controller, the mosfet is active and connects the 499 Ohm resistor to ground, making it possible to let current flow through the resistor. Using formula 5 based on the simple principle of Ohm's law, the voltage over the resistor can be measured and calculated to get the measured current.

$$I_{in} = \frac{V_R}{R_{Pulldown}} \qquad I_{in} = \frac{V_R}{499} \tag{5}$$

Looking at the circuit in figure 15, the output voltage of the pull-down will be lowered by a voltage divider. This voltage divider is calculated earlier in formula 4. Combining formula 5 and 4 gives formula 6, which is the resulting formula to calculate the current using the input from the ADC input.

$$I_{IN} = \frac{V_{ADC} * 6}{499} \tag{6}$$

As seen in figure 17 a simulation has been performed to verify the circuit with a current input. By simulating a current-source at the input and a 1,8 Volt signal on the pull down control pin the following graph was the result. In this graph the input current on the X-axis is shown, with the ADC output on the Y-axis.

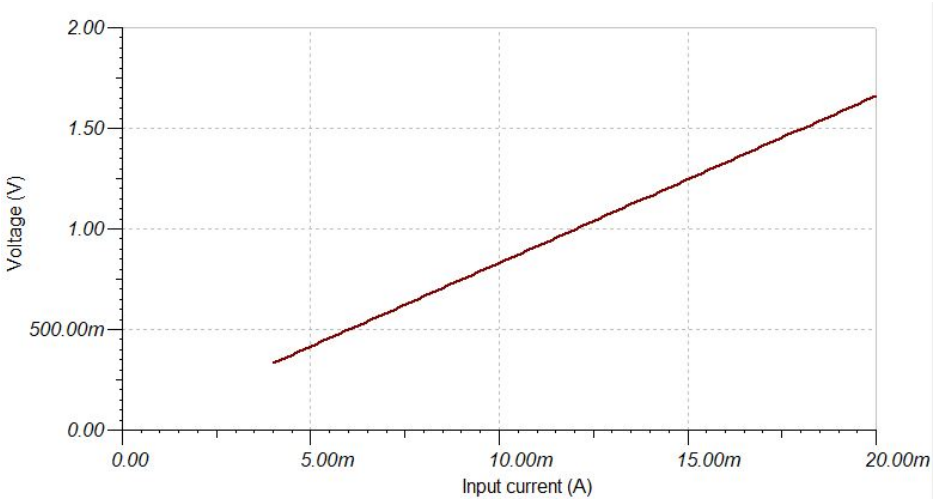


Figure 17 – Simulation of the 4 - 20 mA analogue input

As seen in the figure above, a lineair line is from 400 mV to 1,8 Volt is seen. This is a result that was expected and designed for, implying that the circuit functions correctly.

5.5 PT100

A PT100 sensor is a basic resistor with a resistance of 100 Ohm at 0 degrees C. Varying the resistance per degree C, while heating up or cooling down. To measure a change of temperature, the resistance of these sensors has to be measured. To design a system, the range of the resistor must be given. A temperature range has been provided by Xeelas, which is -20 to 150 degrees C. Resulting in a resistor value between 90 and 160 Ohms, according to graphs published by an article on PT100 [18].

To keep the system as low power as possible, the supply for the temperature measurement will be set to 1,8 Volt. This implies that the internal LDO regulator will be able to power the circuit. This will result in a stable and reliable power supply for the PT100. The circuit is given in figure 18, where the PT100 sensor is connected to ground and the negative input of the Operational amplifier.

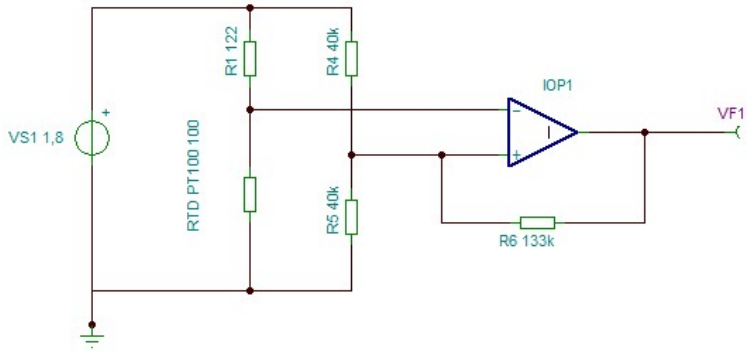


Figure 18 – PT-100 measurement circuit

This circuit has been simulated, as seen in figure 19.

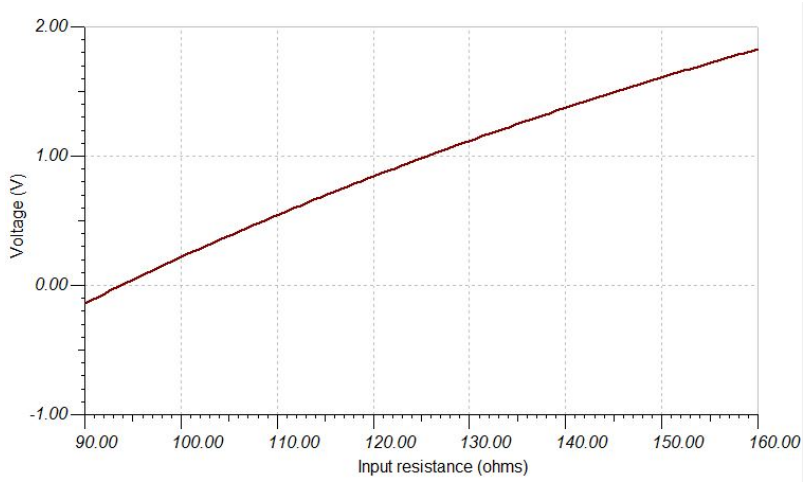


Figure 19 – Simulation of the PT-100 circuit with PT100 resistance as input

In this figure on the X-axis the resistance of the PT100 is shown, with the output voltage to the ADC on the Y-axis. With as result a not perfectly lineair line. This line can't get any straighter than on the given simulation. Thus a slight correction can be performed to the software after measuring the ADC signal.

5.6 Micro-controller

For the brain of the system, the micro-controller, a few options are available. Before choosing, a list of requirements of the system is set up. In this list the ”musts” and ”shoulds” of the micro-controller are given.

1. Should be low power
2. Must support at least two UART port
3. Must support at least four ADC channels
4. Must support at least one PWM output port
5. Should have an as small as possible footprint

Using this list, a few micro-controllers qualified during the search. As well as a few already implemented micro-controllers in older project at Xeelas. After discussing with the lead electronics, the choice for the already implemented micro-controller development environment was made. Resulting in a limited amount of micro-controllers from the nRF series from Nordic Semiconductors.

There were two potential options, the nRF52832 and the nRF52840, both from the same series and with the same aim for low power applications. A table of differences has been given in table 4. The information given in this table are gathered using the given datasheets [19] [20].

	nRF52832	nRF52840
Supply Voltage	1,7 - 3,6	1,7 - 5,5
PINS	48	73
Cost (100)	3,94	4,10
Footprint	QFN48t	QFN73 Ball soldering

Table 4 – Two optional nRF micro-controllers with differing specifications

Looking at the table, the nRF52832 is more apealing due to its package and amount of pins. Since the system does not need more than 30 pins. As well as a small price difference. The nRF52840 has a wider supply voltage that does not need to be regulated using a LDO regulator. This however is still needed for other components on the board, and thus not a real advantage of chosing the nRF52840.

The chosen micro-controller is the simpler, smaller nRF52832 as micro-controller to control the Xe-Link sensor interface board.

6 Hardware design - PCB

Now that the entire schematic has been defined and designed, the next step is to lay all the components out on a Printed Circuit Board (PCB). Where a list of things are important to keep in mind while designing the layout:

1. The dimensions of the board must be as small as possible
2. Low Noise transmission from and to components, separating power, analogue and digital components
3. In and output connectors that have to be defined

6.1 Dimensions

First of all, the size of the PCB is important to be defined, as well as the layer count and thickness of the PCB.

To keep the price as low as possible, the layer count is preferably two-layered, one top and one bottom with no inner layers. With the preference of placing all components on one layer, to keep manufacturing costs as low as possible as well. As last, the copper thickness of the PCB is preferably as thin as possible (1 oz), also to lower production costs.

With these requirements in mind it is possible to take a look at the size of the PCB. A dimension has been supplied by Xeelas, which is 20 mm by 50 mm. This will result in a to tiny space to fit all the components on one side of the PCB, which will increase the cost by a little fragment. Since it is more expensive to design a bigger casing than to place components on both the sides of PCB, the choice is made to keep the size at 20 by 50 mm and placing the components on both sides of the PCB.

6.2 IO ports to external devices

Since the input and output connector are not defined yet, the connectors have to be defined before placing the components. The board is 20 mm wide, and thus resulting in a connector on both the long ends at a maximal width of around 15 mm. Since the Xe-Link will be connected to two different devices (A Gateway and a sensor) it is preferred to separate these connectors. Resulting in two connectors on both ends of the PCB.

For both connectors, a strong and watertight connection is preferred.

Sensor connector
The sensor connector must be capable of providing all connections to the sensor that is attached. A list has been set up below:

1. VDD out for the Sensor
2. Analogue in (0 - 10 V and 4-20 mA)
3. Pt-100 Sensor
4. GND (preferably 2)

Two grounds have been chosen, so that both a PT100 and analogue sensor can be attached easily at the same time.

This connector must be an easy to attach but steady connection. Thus choosing for an on PCB connector that is easily attachable, then reinforcing it by using a cable sleeve at the encasing gives the best option. This will result in an easy changeable system, that is watertight and easy to change.

Gateway connector
The gateway consists of 5 wires:

1. VDD in
2. GND
3. TX (UART transceiving pin)
4. RX (UART receiving pin)
5. Interupt Out

Since the connection to the gateway has to be as noise-resistive as possible a shielded industrial standard is recommended. The gateway side of the cable is pre-defined, where a cable to a M12 5 Pin connector is a possibility. By choosing a direct PCB to M12 5 Pin header with a 90 degrees angle, both the water-tightness and structural reliability meet the requirements.

Concept layout connectors

In figure 20 an example has been given of the enclosure of the Xe-Link. This enclosure is designed within Xeelas to give a concept on how the system could look.

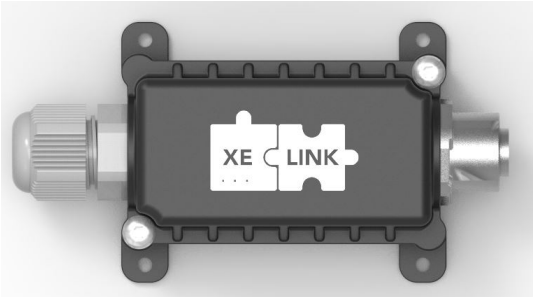


Figure 20 – Housing design example of the Xe-Link sensor interface

As in this figure can be seen, on the right side is a aluminium cap, which is the connector to the Gateway. On the left side a cable sleeve is seen for the sensors. The total width and length of the system are around 30 mm by 60 mm, making it a small system.

6.3 Component layout

The placement of components have been thoughtfully placed. Where analogue input handling circuits are placed as close to the incoming connector as possible. The boost converter has been placed as far as possible from all other systems, like the micro-controller, analogue inputs and level shifting IC's. All decoupling capacitors have been placed as close to the corresponding leads as possible and all components have been laid out according to their datasheets. Especially the Boost converter, which is spoken of in the next chapter.

6.4 Boost converter specific layout

Since the boost converter is a high current (900 mA max), high frequency (100 kHz - 1 MHz), the chance of generating noise is high. To reduce the noise and increase its efficiency it is important that the circuit is well laid out and well grounded. The datasheet of the boost-converter IC [3] provides a recommended layout as seen in figure 21.

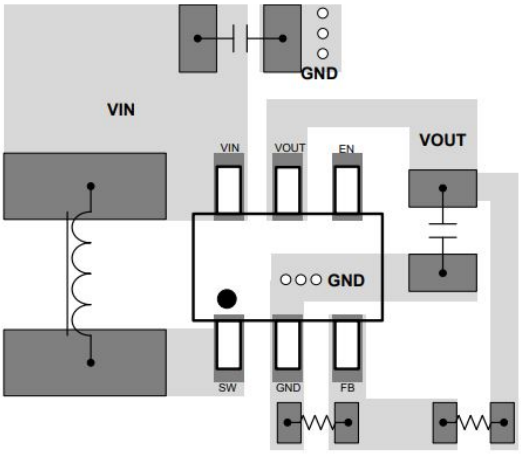
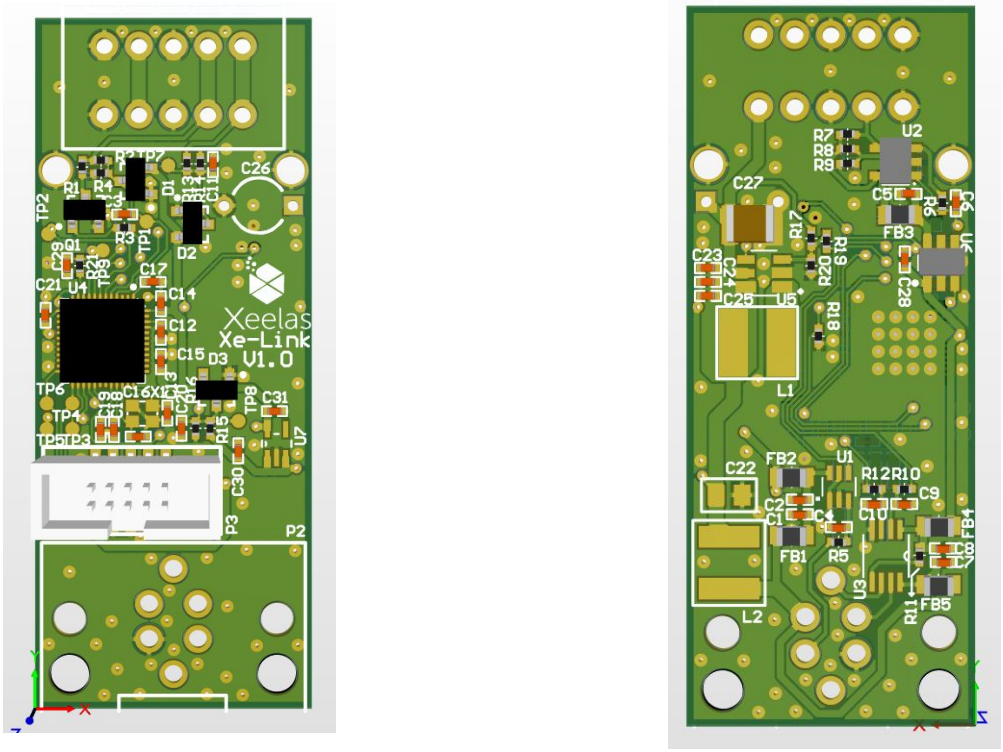


Figure 21 – Recommended layout of the boost-converter circuit, Source: datasheet TLV6104 [3]

6.5 Result

As seen in figure 22, the result is given of the process of the PCB-design. In this figure it is clearly noticeable how much space all the connectors need altogether. Overall, the result is promising and now ready for assembly and testing.

Note that some components pictured in figure 22 are drawn in 3D models, and others not. The real PCB will look slightly different from these pictures, but the principle will keep the same.



(a) Front side PCB

(b) Bottom side PCB

Figure 22 – Front and bottom side of PCB

7 PCB assembly

After designing and ordering the PCB, the next step is to assemble the circuit. This is done after ordering all other materials, for example, the components.

7.1 Bill of Materials (BoM)

To order all other components, a list of components has been generated using Altium's export Bill of Material option. The BoM can be found in appendix C.

Preferably all components have to be ordered at the same company, preferably from Digikey. Digikey is a supplier, focussed on electrical designs and components. Since the company is based in America, shipment costs are high. To prevent these costs, a bill of over 50 euro's will be ordered, resulting in free shipping. This results in an order consisting of the components for three PCB's. These extra ordered components can be used later on to assemble a second and third board to test re-occurring issues.

One component, the M12 5 Pin connector has been outsourced to a Chinese company using Alibaba Business. This component normally is around 15 euro's per piece, while from china this component costs around 5 euro's. Since the Xe-Link initial idea is to be manufactured in China, keeping contact with the supplier is recommended.

7.2 Soldering-process

After ordering and receiving all components, the soldering process has began. Using soldering-paste, a heat-gun and a microscope, this task has been performed perfectly. During the process no problems have been encountered and the system is ready to be tested.

In figure 23, the result after soldering of the PCB can be found. The PCB has been placed in a 3D printed holder to make handling easy, since the design is too small to handle easily.

Figure 23 – Result of soldered PCB in 3D printed holder

7.3 First test

By applying a voltage on the input voltage of the Xe-Link's input and then attaching a programmer to the programming header, a first connection has been made with the on-board micro-controller. Implying that the system is able to communicate with the computer and ready to be programmed. This was a crucial test, resulting to be positive. If this test did not work as intended, this would mean that the micro-controller was not able to receive any commands to be programmed and the PCB had to be revisited for changes.

8 Software development

As stated in chapter 7.3, the Xe-Link's micro-controller was able to be programmed. In this chapter, the software written during this internship will be discussed and explained.

8.1 Version control and GIT repository

Since the documentation, version control and accessibility of this projects software is important, the project is shared via Git. A Git repository has been created so that every colleague is able to access the projects code at any time. With the use of Git-Flow [21], a way to manage a repository by adding one feature at a time, the Git will be managed.

8.2 Nordic SDK and Segger Studio's

Since the chosen micro-controller is a nRF-series mircro-controller from Nordic-Semi. A by the supplier recommended development studio has been given. This programming studio is called Segger. The micro-controller also has a provided driver library for controlling all hardware drivers and other useful streamlined libraries. This list of library's and drivers is called the Nordic Software Development Kit (SDK). The provided SDK consists of an online documentation page from Nordic, listing all the included drivers and libraries in detail, including examples and guides.

Most of the software designed during this internship is based on these SDK drivers and libraries, since this is recommended by Nordic-semi and Xeelas to be included in the project.

8.3 Use-Case flow charts

To get a better understanding of what th Xe-Link must be capable of, multiple flowcharts have been drawn. One flow chart each per use case (Use cases seen in figure 8). These flowcharts give a rough understanding of what the system must do in any use case. The Flow-charts can be seen in appendix D. According to these flow-charts a few specific called functions and hardware drivers have to be set. These hardware drivers are listed below:

1. Boost converter control (PWM and on/off)
2. UART communication (using SSI)
3. SAADC readout - analogue sensors 4-20 mA
4. SAADC readout - analogue sensors 0-10 V
5. SAADC readout - input voltage
6. SAADC readout - output voltage

8.4 Boost-converter control

To control the boost-converter, a General Porpuse Input Output (GPIO) pin to the enable pin needs to be controlled. As well as a PWM pin that controls the output voltage of the boost converter.

8.4.1 Initialisation

Before using the boos-converter the GPIO and PWM pin have to be initialised. This is done by a seperate function that will be called at the startup of the micro-controller. The initialisation function can be found in listing 1.

```
1 void xelink_boost_init(void)
2 {
3   nrf_gpio_cfg_output(XELINK_BOOST_EN_PIN);
4   nrf_gpio_pin_write(XELINK_BOOST_EN_PIN, 0);
5
6   xelink_pwm_init(); //initialise PWM
7 }
```

Listing 1: Boost-converter initialisation function

PWM initialisation

In the function in listing 1 a PWM initialisation function is called. This function initialises the PWM driver from the SDK. The initialisation can be found in listing 2.

```
void xelink_pwm_init(void)
{
  nrfx_pwm_config_t const pwm_config =
  {
  .output_pins =
  {
            XELINK_BOOST_PWM_PIN,    // channel 0
            NRFX_PWM_PIN_NOT_USED,   // channel 1 (NOT USED)
            NRFX_PWM_PIN_NOT_USED,   // channel 2 (NOT USED)
            NRFX_PWM_PIN_NOT_USED,   // channel 3 (NOT USED)
  },
  .irq_priority = APP_IRQ_PRIORITY_LOWEST,
  .base_clock   = NRF_PWM_CLK_16MHz,  //
  .count_mode   = NRF_PWM_MODE_UP,
  .top_value    = 100,                      // Setting range from 0 to 100 %
  .load_mode    = NRF_PWM_LOAD_INDIVIDUAL,
  .step_mode    = NRF_PWM_STEP_AUTO
  };

nrfx_pwm_init(&m_pwm0, &pwm_config, NULL);

xelink_pwm_update_duty_cycle(0);
}
```

Listing 2: PWM driver initialisation function

Using the pre-configured SDK configuration of the PWM, some things have to be changed. As well as setting the channels. Only one channel of the driver is being used, selecting the micro-controller pin, frequency, priority and mode. By setting the top value to 100 ticks, 100 different steps can be made which is perfectly a range from 0 to 100%. The xelink_pwm_update_duty_cycle(0) function called at last is the function to set the percentage of the output PWM output. This function is given in listing 3, a value between 0 and 100 has to be given when calling this function, if not the function will normalize this and send that value to the SDK's driver.

```
void xelink_pwm_update_duty_cycle(uint8_t duty_cycle)
{
    // Check if value is outside of range. If so, set to 100%
    if(duty_cycle >= 100) {
        seq_values->channel_0 = 100;
    }
    else if(duty_cycle <= 0 ) {
        seq_values->channel_0 = 0;
    } else {
        seq_values->channel_0 = duty_cycle;
    }
    nrfx_pwm_simple_playback(&m_pwm0, &seq, 1, NRFX_PWM_FLAG_LOOP);
    }
}
```

Listing 3: Update PWM percentage function

The above function will be called by a function spoken of below, which calculates the PWM output percentage using a voltage inputted by the user or other parts of the software.

8.4.2 Boost converter control functions

Several functions for controlling the boost converter have been writen. Some functions need explanation, others are as simple as they can be and do not need any extra details. A list of functions to control the boost converter are below with a small description:

1. Initialisation boost converter and PWM driver (Spoken of in chapter 8.4.1)
2. Update PWM duty cycle (spoken of in chapter 8.4.1)
3. Set and get output voltage of boost-converter
4. Measure the output voltage of boost-converter (Spoken of in chapter 8.5.4)
5. Set and get on/off state of boost-converter

Set and get output voltage of boost-converter
To control the boost-converters output voltage the two functions given in listing 4 are written.

```
void xelink_boost_set_voltage(float voltage)
{
  boost_voltage = voltage;
  if (voltage < 3.0) {
    boost_pwm = 0.0 ;
  } else {
    boost_pwm = (5.348 * boost_voltage) - 15.5 ;
  }
  if (boost_pwm <= 0){
    xelink_pwm_update_duty_cycle(0);
  } else {
    xelink_pwm_update_duty_cycle(boost_pwm);
  }
}

float xelink_boost_get_voltage(void)
{
  return boost_voltage;
}
```

Listing 4: Set and get Voltage function of the boost-converter

As the listing implies, a global variable float called boost_voltage is the stored voltage that is the currently set voltage. By setting a new output voltage the newly set voltage will be saved in this global variable. After saving the voltage, the PWM signal has to be calculated, so that the correct value is sent to the PWM driver. This is done using formula 7, the formula is based on the measurements done in chapter 9.3.

The function also makes sure a correct voltage has been requested to be generated.

$$PWM = 5,348 * V_{Boost} - 15,5 \tag{7}$$

The get function which also can be seen in listing 4is rather simple, just returning the global variable. This function is currently not being used but can be useful while debugging or in later applications.

Set and get on/off state of boost-converter
By changing the on/off state of the boost-converter the user or program can easily turn on or off the IC. Resulting in the output voltage being either 0 when off, or the set voltage when turned on. The state is saved in a global variable, so that the system can get the state at any time.

The get function is a simple return function, which returns the current state of the boost-converter. As seen in listing 5.

The set function is made to accept a Boolean input, where 1 is on and 0 is off. By calling this function with a 0 or 1, the user is able to turn on or off the boost-converter. This function will be mainly used for limiting power consumption while in sleep mode.

```
void xelink_boost_set_state(bool state)
{
  boost_state = state;
  nrf_gpio_pin_write(XELINK_BOOST_EN_PIN, boost_state);
}

bool xelink_boost_get_state(void)
{
  return boost_state;
}
```

Listing 5: Set and get state functions of the boost-converter driver

8.5 SAADC driver

The Succesive Aproximation Analogue Digital Converter (SAADC) driver is the driver that controls all ADC channels. This driver makes it possible to measure voltages so that the user can read these values out digitally. The SAADC driver is used in 4 cases:

1. Analogue Sensor in (4-20 mA & 0-10 V)
2. Analogue Sensor in (PT100)
3. Input Voltage
4. Output Voltage (Boost-converter)

Before these measurement can be performed, the SAADC driver needs to be initialised.

8.5.1 Initialisation and settings

The code given in listing 6 is the initialisation function of the SAADC driver for one channel. All the channels currently have the same setting, thus showing one channel is currently enough. Some settings are important and will be spoken of separately.

```
void xelink_adc_init(void)
{
  nrfx_saadc_calibrate_offset();

  nrf_saadc_channel_config_t channel_config_vddout = NRFX_SAADC_DEFAULT_CHANNEL_CONFIG_SE(
    XELINK_ADC_VDDOUT);

  nrfx_saadc_config_t adc_config =
  {
    .resolution = 1,
    .oversample = 0,
    .interrupt_priority = APP_IRQ_PRIORITY_LOWEST,
    .low_power_mode = 0
  };
  nrfx_saadc_init(&adc_config, NULL);
  nrfx_saadc_channel_init(XELINK_ADC_CH_VDDOUT, &channel_config_vddout);
  nrf_gpio_cfg_output(XELINK_PULLDOWNCONTROL_PIN);
  nrf_gpio_pin_write(XELINK_PULLDOWNCONTROL_PIN, xelink_sensor_pulldown);
}
```

Listing 6: SAADC initialisation function

Resolution
The resolution is currently set to 1, which is a resolution of 10 bits according to the SDK documentation [4], the resolution settings are also shown in table 5.

Mode	Resolution
0	8 bits
1	10 bits
2	12 bits
3	14 bits

Table 5 – Table of different resolution settings for the SAADC, Source: nRF SDK [4]

Whenever the sample function will be called, an integer will be returned by the driver. To convert this in a Voltage at the lead of the micro-controller, a small calculation based on the given formula from the micro-controllers datasheet is needed. The calculation can be found in formula 8.

$$\begin{aligned} RESULT &= V * \frac{GAIN}{REFERENCE} * 2^{RESOLUTION-m} \\ V &= \frac{RESULT}{\frac{GAIN}{REFERENCE} * 2^{RESOLUTION-m}} \\ V &= \frac{RESULT}{284} \end{aligned} \tag{8}$$

In this equation, there are several variables given. Where result is the integer generated from the ADC, GAIN is a standard configured internal gain set to 1/6th, REFERENCE is an internal reference voltage of 0,6 Volt, m is 0 when using the single ended mode and RESOLUTION is the chosen resolution of 10 bits.

In Chapter 9 the results of the ADC will be measured tested and verified.

8.5.2 Analogue sensor (0-10 V and 4-20 mA)

0 - 10 V Sensor-ADC calculation
With the before calculated ADC readout formula 8, the sensor 0-10V input voltage can be measured as well. This input is lowered to the correct voltage to not blow up the micro-controller. This is done by a simple voltage divider using two resistors. In formula 9, the division is calculated so that the correct voltage from the sensor can be read out.

$$\begin{aligned} V_{nRF} &= V_{SENSOR} * \frac{R_2}{R_1 + R_2} \\ V_{SENSOR} &= V_{nRF} * 6 \end{aligned} \tag{9}$$

4 - 20 mA Sensor-ADC calculation
Since the 4 - 20 mA sensor is a pull-down resistor on the same input as the 0-10 Volt input. The same voltage divider is used, thus resulting in the same voltage step-down. However, to take in account that a current as output is wanted over a 500 Ohm resistor. The equation given in formula 10 is calculated. This formula will be used when a current is expected to be returned.

$$\begin{aligned} I_{SENSOR} &= \frac{V_{SENSOR}}{500} \\ I_{SENSOR} &= \frac{V_A DC * 6}{500} = 0,012 * V_{ADC} \end{aligned} \tag{10}$$

MosFet Pull-down control pin
Since both the 0-10V and 4-20 mA Sensors are functioning on the same circuit, while the 4-20 mA sensor needs a pull-down resistor and the 0-10 V sensor does not, this pull-down is controlled by a MosFet, controlled using a simple GPIO-Pin. This Pin must be set to the correct mode when a measurement will be done.

Since this circuit has two modes, current measurement and voltage measurement the called function to measure a sensor has an input Boolean to select what mode the measurement is performed. At the start of the function, a brief check will be performed to see whether the hardware is currently set to the correct mode. When a different mode is selected, the GPIO-pin will be switched to the correct mode so that the correct measurement will be done.

The correct output calculation will be performed according to the mode inputted by the user. When the CURRENT_MODE has been selected, the function will return a value calculated using formula 10 and vice versa.

Using the before initialised SAADC driver, the function to measure the 0-10V and 4-20 mA sensor is seen in listing 7

```
float xelink_adc_meas_sensor(bool mode)
{
  if (mode == XELINK_SENSOR_MODE_VOLTAGE) {
    xelink_adc_sensor_set_pulldown(XELINK_SENSOR_MODE_VOLTAGE);
    nrfx_saadc_sample_convert(XELINK_ADC_CH_SENSOR, &value);

    return (((value / 284.4) * 6) * 1) ;
  } else if (mode == XELINK_SENSOR_MODE_CURRENT) {
    xelink_adc_sensor_set_pulldown(XELINK_SENSOR_MODE_CURRENT);
    nrfx_saadc_sample_convert(XELINK_ADC_CH_SENSOR, &value);

    return (value / 284.4) * 0.012;
  } else {
    return 0;
  }
}
```

Listing 7: 0-10 V and 4-20 mA Sensor readout function

8.5.3 Input Voltage

To measure the input voltage a simple function can be called. The function is given in listing 8 and is simply calling the SDK driver single conversion and calculating the actual voltage. Again, this voltage has been stepped down using a voltage divider. The function will return the input voltage as a float in Volts.

```
float xelink_adc_meas_vddin()
{
  nrfx_saadc_sample_convert(XELINK_ADC_CH_VDDIN, &value);
  return ((value / 284.4) * 3.488) ;
}
```

Listing 8: Input voltage measurement function

With the help of this function, a check can be done on how healthy the supply voltage for the Xe-Link is.

8.5.4 Output Voltage

To measure the output Voltage the same procedure and almost same function is used. As seen in listing 9, the only differences are the calculation since a different voltage divider ratio is used, and the chosen channel where te conversion takes place.

```
float xelink_adc_meas_vddout()
{
  nrfx_saadc_sample_convert(XELINK_ADC_CH_VDDOUT, &value);
  return ((value / 284.4) * 14.3);
}
```

Listing 9: Output voltage measurement function

This function is useful to check the status of the boost converter, since this function measures the output voltage. As seen in the figures in appendix D, measurements to check what the status is of the output voltage are being used often.

8.6 UART (SSI) communication

During the internship there was no time left to work on the communication protocol. Resulting in a recommendation to continue on this part of the Xe-Link, before knowing for sure the system is capable of functioning as wanted.

The messages and actions the communication protocol are defined in the flowcharts, thus the realisation of the communication protocol mainly is the implementation of these commands.

9 Tests and test results

Now that the first hardware drivers have been made and written, some hardware and software results can be analysed. These results will inform whether the hard- and software are functioning correctly, not functioning at all or need some patching.

9.1 Test set up and equipment

During the testing phase, different tests will be performed. These test are done using specific measurement equipment. To give a clear overview of what measurement equipment is used a list will be supplied:

1. Lab-bench power supply Voltcraft PS-1302 D (To provide a variable voltage to power the Xe-Link)
2. Tenma 72-10390A Multi-meter (Measuring currents or voltages)
3. Tenma 72-2590 Multi-Meter (Measuring currents or voltages)
4. Won Smart DS7102V oscilloscope

The given multi-meters are not the most ideal equipment to measure voltages or currents, since they are limited to a maximum of two digits. This however is the only equipment currently available due to the Covid-19 work-from-home crisis.

9.2 Power consumption and input voltage

The Xe-Link sensor interface board is used in low power IoT application. Due to the low power application it is important that the Xe-Link uses low power, even when doing nothing.

The power consumption measurements have been done when no program was running on the micro-controller, thus running in an infinite while loop.

While performing the power consumption measurements, the input voltage will be changed from 1,8 Volts to 5,5 Volts. This changing input voltage will determine whether the Xe-Link is able to function within the required boundaries (between 2,0 and 5,5 Volts). The results are shown in table 6.

Input voltage (V)	Input current (mA)	Input power (mW)
1,8	5,5	9,9
2,0	5,5	11,0
2,2	5,5	12,1
2,5	5,5	13,8
3,0	5,5	16,5
3,5	5,7	20,0
4,0	5,9	23,6
4,5	6,3	28,4
4,8	6,5	31,2
5,0	6,6	33,0
5,3	6,9	36,6
5,5	7,1	39,1

Table 6 – Power consumption using different input voltages

From the above table two things can be concluded:

The input voltage requirement is met. The micro-controller was still able to communicate and be programmed at all voltages and thus the input voltage range of 2,0 to 5,5 Volt is met.

The input current is around 6 mA, which is a current that is expected while running in an infinite loop. Currently there is no sleep-mode or low-power mode activated, which is something that has to be considered. The power consumption is as low as expected, thus resulting in a positive test result.

9.3 Boost converter

9.3.1 Output voltage control

The boost converter must be able to produce a voltage between 5 and 24 Volts according to the requirements. This needs to be tested thoroughly.

The output voltage can be controlled via software spoken of in chapter 8.4 with a PWM signal as input. Using this function and increasing the PWM signal with steps of 10 %, a graph can be drawn. This graph can be seen in figure 24.

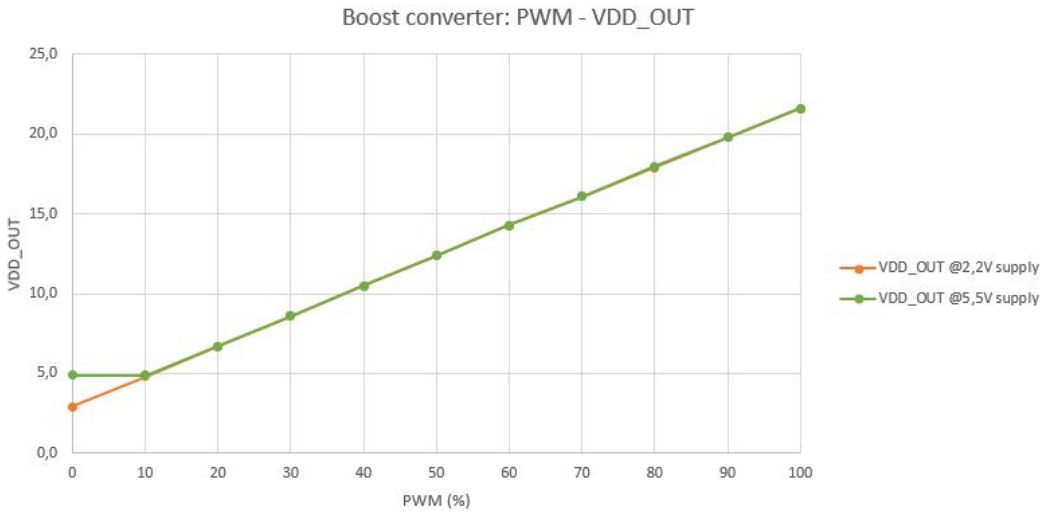


Figure 24 – Boost converter PWM input to output voltage curve

The figure above has the PWM input on the x-axis, where the resulting output voltage of the boost-converter is drawn on the Y-axis. Two different curves can be seen in the figure, these curves have a different input voltage to verify there is no big difference between the different input voltages. Since the curves both overlap each other perfectly this is not the case.

Concluding from this graph, it is clearly noticeable that the boost-converter is lineair, and reliable. However, the expected range from 5 to 24 Volts has not been reached entirely. The current range of the boost-converter according to this graph is from 5 to 21,6 Volts, which is not within the requirements.

Concluding from this test, it is important that the boost-converter needs a revisit after the internship has ended where the range has to be improved.

9.3.2 Turn on/off pin

The turn on and off pin of the boost converter is simply tested by turning on and off the boost converter using its written software function. By turning off the boost-converter a GPIO-pin will be set to low, that is connected to the boost-converter's IC. According to its datasheet [3], the IC will be shut off, isolating the output with the input.

By measuring the voltage when on and off, it is clearly noticeable that the voltage drops down when turned off. A resistance measurement between the in and output has been performed. The multi-meter measures the resistance between the in and output as an open circuit, implying that the connection is broken when turned off.

This results in a very low to no current draw when the boost-converter is off, even when a load is attached.

9.3.3 Startup time

As stated before in chapter 5.1.2 the startup time must be as low as possible. According to the simulation done in figure 13, the startup time should be around 7,5 msec.

By measuring the output Voltage of the boost-converter on an oscilloscope at startup the time that it takes can be measured. The result of this method can be seen in figure 25.

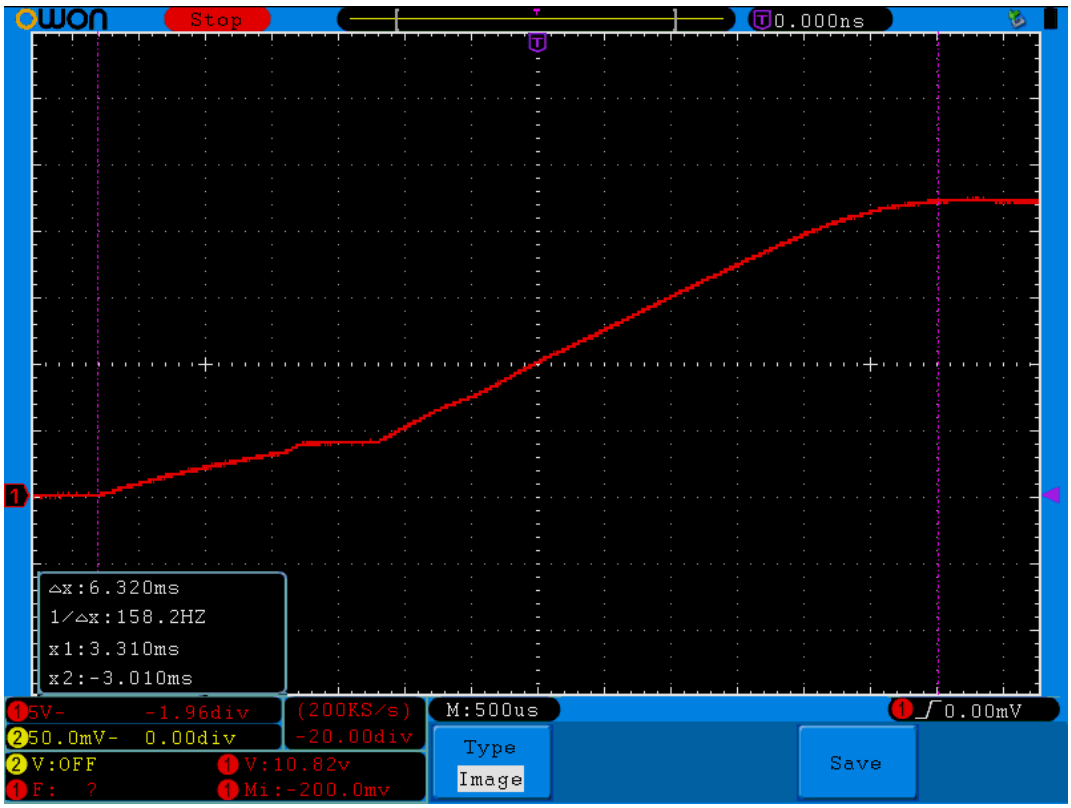


Figure 25 – Boost converter startup time measured on Oscilloscope with a 2000 Ohm load

In this figure the time-interval from the start of the curve to the end is measured. The time in between these points is 6,320 ms according to the Oscilloscope. This is within the expected value according to the simulations. This measurement has been done with and without load, and looked exactly the same. The conclusion from this test is that the startup time of the boost-converter is around 6,3 msec and thus functions correctly.

9.3.4 Efficiency

In order to verify the power efficiency of the boost-converter a few tests have to be performed. With the earlier given function to control the PWM signal and thus controlling the output voltage a variable ”PWM signal can be chosen. Next to a variable output voltage, a variable load has to be measured on efficiency as well. In figure 26 both these inputs have been put in with the efficiency of the boost-converter as output.

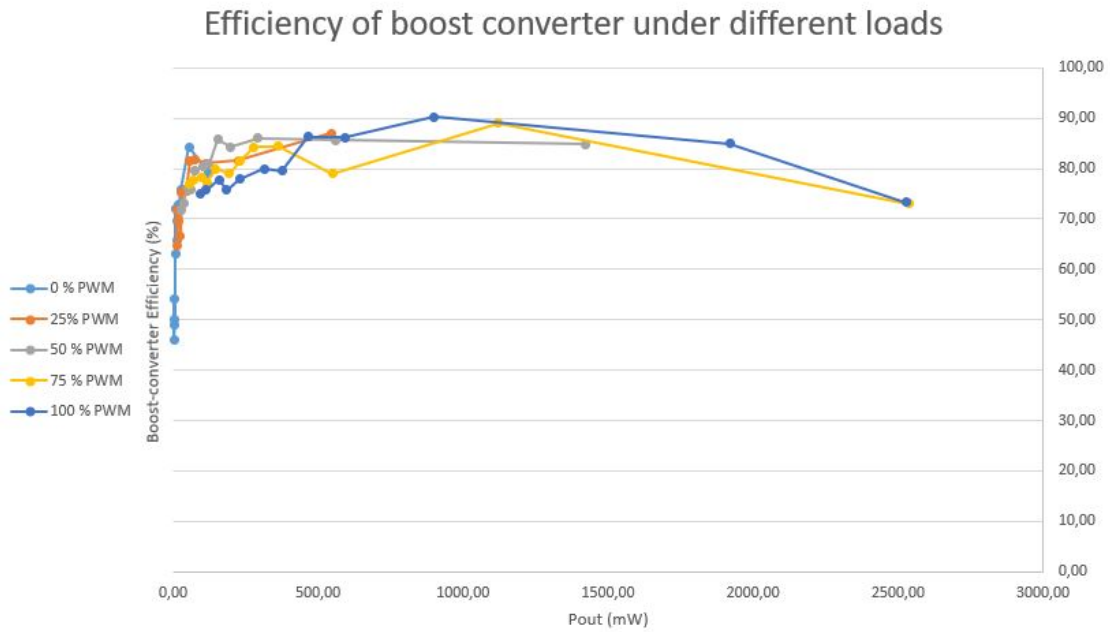


Figure 26 – Boost converter’s efficiency on different loads and pwm inputs

In the figure above multiple lines are visible, with each line having a set PWM value, thus changing to a higher or lower output voltage. A higher PWM value means a higher output voltage. For every PWM set line, multiple loads have been tested as seen on the x-axis.

Using the measured input power and measured output power, the efficiency can be calculated, which is drawn on the Y-axis.

Concluding from this figure, the efficiency of the 0% PWM signal is rather low at the beginning. This PWM-value will not be used during use of the Xe-Link and is therefore not a problem. The other curves of with differing PWM signals have a similar response compared ot each other, implying that the efficiency does not change when a different output voltage has been chosen. The overall efficiency of the boost-converter settles between and % which is as expected. Since a short circuit has been tested now as well, it is safe to conclude that the circuit is also capable of not burning out on maximum load.

The maximum power that the boost-converter was able to produce without an problems was around 2000 mW. Which at 24 Volts would be around 80 mA’s, which is enough to power almost all sensor given in the research table (table 1).

Concluding from these tests, the efficiency is as expected at around 80 %. The boost-converter is able to provide the needed power for the sensors and thus functions correctly. Of course, with the exception of the earlier stated conclusion for the output voltage maximum.

9.4 SAADC output Voltage

To verify the functionality of the boost-converter, one of the SAADC channesl is dedicated to measure the output voltage. Ranging from 5 to 24 Volts. To verify whether this ADC channel is working and how well the linearity and measurement errors turn out to be a few tests have been performed.

This SAADC channel is the first to be measured, thus resulting in changes to the software in overal for all SAADC channels, for example the settings.

To draw a graph with the response of the ADC, an input voltage to the circuit has to be generated. This voltage is generated by running the boost-converter of the Xe-Link. This voltage will be measured by both the ADC and a multimeter as baseline. By measuring the ADC's conversion, a graph can be made, as seen in figure 27.

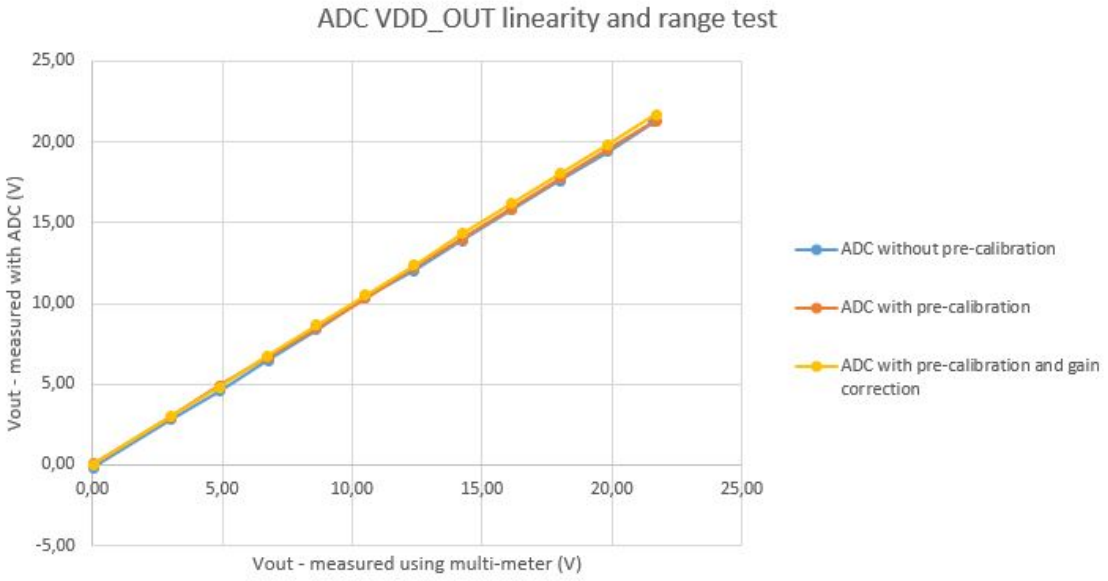


Figure 27 – Vout ADC measurement linearity and range test

As seen in this graph, multiple colloured lines can be seen. In the legend the difference between these lines is give. At first, the blue line, then after the offset error handling spoken of below the red, and at last after an attempt of gain correction the yellow line. The same line-colours can be seen in the error graph, seen in figure 28. In this graph the calculated error is shown, using formula 11.

$$V_{ERROR}[n] = V_{out}[n] - VMEASURED\_BY\_ADC[n] \tag{11}$$

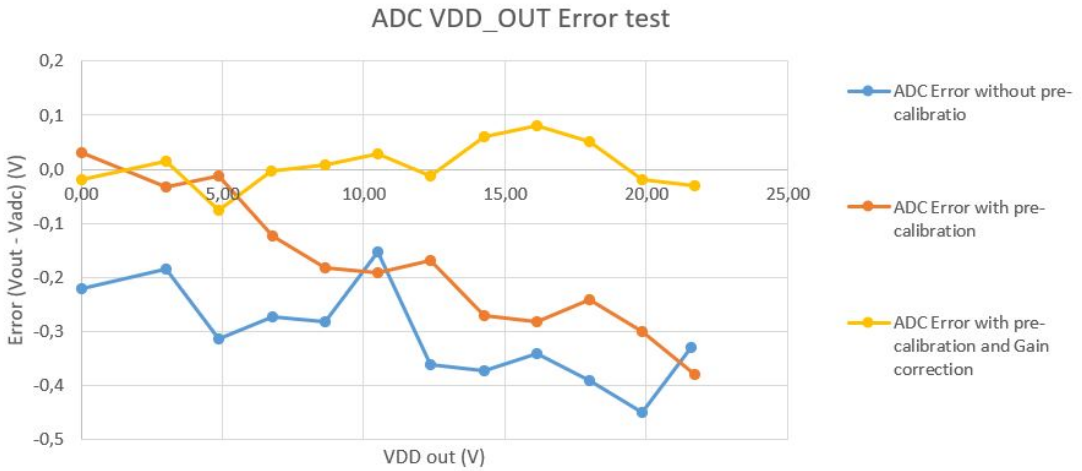


Figure 28 – Vout ADC error on different settings

In this graph, again three different curves are shown. At first, the blue line, the initial measurement without any error handling. There are two methods of decreasing the overall error of the SAADC. Offset and gain correction. Both spoken of below.

Offset error
The Offset error is measurable at the start of the curve, at 0 Volts. The error here is around -0,2 Volts, and can be improved by adding a offset calibration at the start-up of the Xe-Link. This function is part of the nRF SDK, and is easily called.

After adding this function in a new measurement has been performed, seen in figure 28 again, this time as the orange line. It is clearly noticeable that the offset error has been improved drastically.

Gain error
To optionally improve the SAADC even more, a gain correction can be perfomred as well. Calculating the gain error using formula 12, the gain correction can be added to the code. Resulting in the yellow line seen in figure 28.

$$\begin{aligned} Correctionmultiplier &= \frac{V_{ERROR(0V)} * V_{ERROR(21,6V)}}{21,6} \\ Correctionmultiplier &= \frac{0,3559}{21,6} \\ CorrectionMultiplier &= 1,01647 \end{aligned} \tag{12}$$

As seen, the overall error has been minimised. However, since the gain error is around 1,6 %, the error might be caused by imperfect components like resistors with a tolerance of 2 %. Thus making this implementation useless, this will however be taken in mind when a second Xe-Link will be assembled, looking at the possibility this might be a fundamental error and not a board specific issue.

9.5 SAADC input Voltage

The input voltage ADC measurement is performed to verify the state of the input voltage of the Xe-Link. To measure a voltage between 2 and 5,5 Volt an ADC has been configured including a voltage division performed hardware-wise. To test the SAADC input, a few measurements have been done, which puts out a graph seen in figure 29.

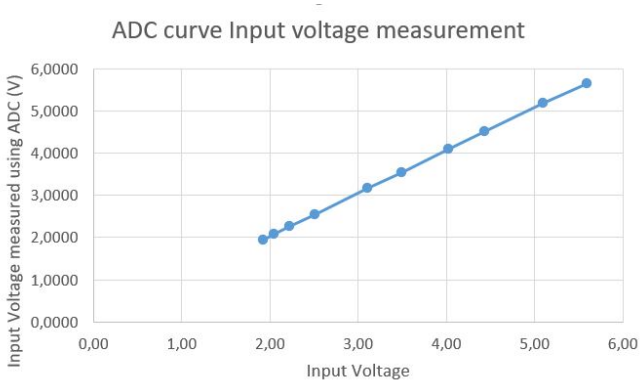


Figure 29 – Input Voltage measurement tests ADC

As seen in this figure, the SAADC is able to measure the entire input range. Thus making it possible to measure the required information.

9.6 Analogue sensor - 0-10 V

As the software function for the SAADC driver and measurement-function for the 0-10 V are defined in chapter 8.5.2. The hard and software is ready to be tested. Most of the measurements and adjustments done for the 0-10 V sensor will be carried over to the 4-20 mA sensor, since the same hardware and software driver are being used.

At first, by supplying a voltage between 0 and 10 Volts with increments of 0,5 Volts a curve can be generated. This curve will show whether the ADC is picking up the input voltage correctly over the entirety of the input range, and whether the line is lineair with no fundamental gain error. The results can be found in the graph in figure 30.

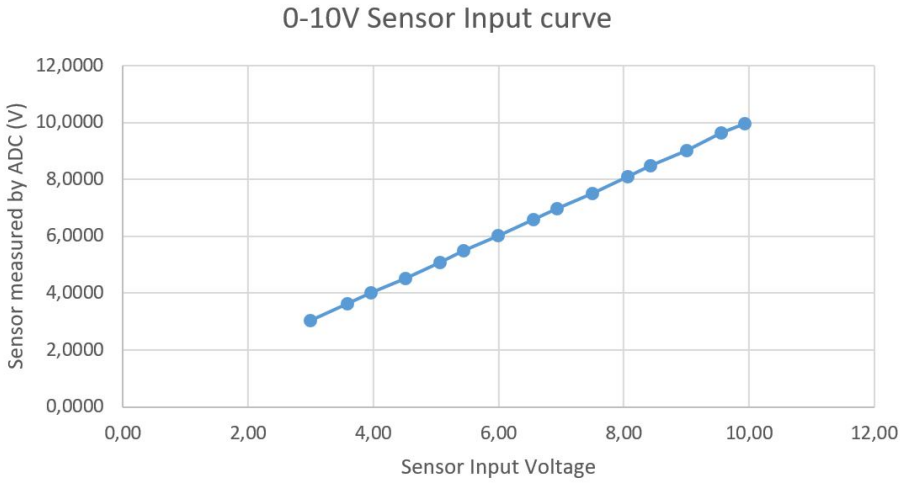


Figure 30 – 0-10 V Sensor input range and linearity test

Concluding from this graph, the line looks linear and no real errors are seen. To be entirely sure, a graph with the accuring error needs to be generated. As seen in figure 31.

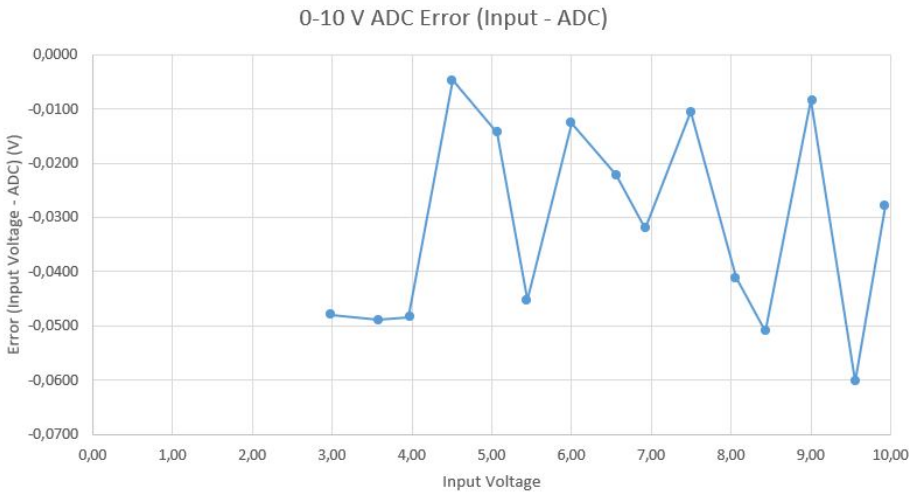


Figure 31 – 0-10 V Sensor Error curve

According to this graph, no fundamental errors are taking place, implying no actions need to performed. The jumps that can be seen in the error graph are jumps of around 0,02 Volts maximum, resulting from noise on the internal reference voltage and gain amplifier.

9.7 Analogue sensor - 4-20 mA

As the first measurements have been performed on the analogue sensor input channel. The next measurement to be done is the 4-20 mA on the same channel. By performing these tests, the linearity of the SAADC and the functionality of the MosFet will be tested. By applying a current source to the input of the analogue sensor, and varying this current a graph can be generated, as seen in figure 32.

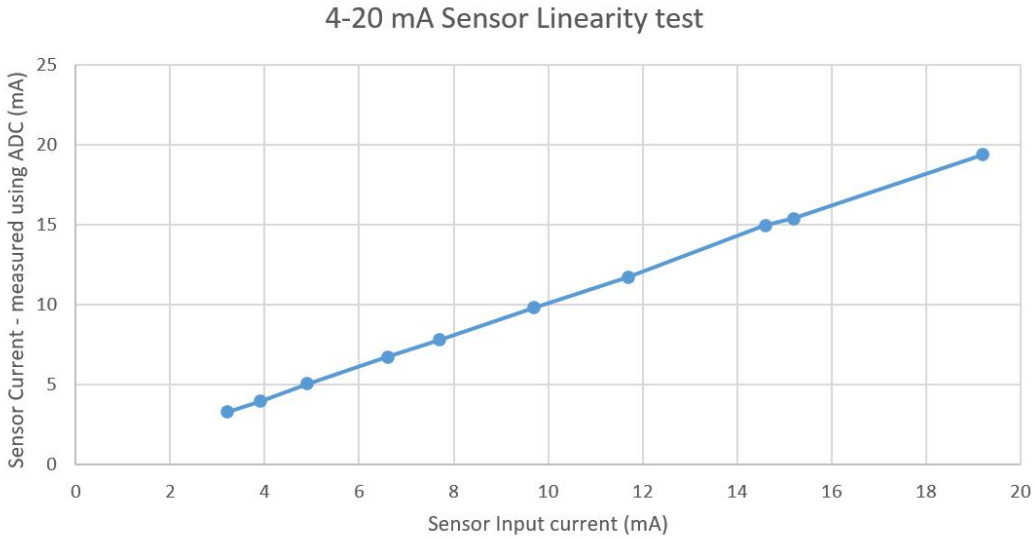


Figure 32 – 4-20 mA Sensor ADC test

In this figure the curve seen is clearly lineair and does read out the range that it is required to measure (4 - 20 mA). Not seen in this figure is the 0 point, this point also reads out a zero in the software. This is important to know so that whenever the sensor is not connected correctly or functioning correctly the systems will be able to notify that the there is something wrong.

To control whether the circuit or software needs an improvement a look at the measured current error needs to be performed. The ADC error is calculated using the equation given in formula 13.

$$I_{ERROR}[n] = I_{Sensor}[n] - IMEASURED\_BY\_ADC[n] \tag{13}$$

Using this formula, a new graph can be drawn. As seen in figure 33 This graph shows the error-curve over, with the input current as x-axis.

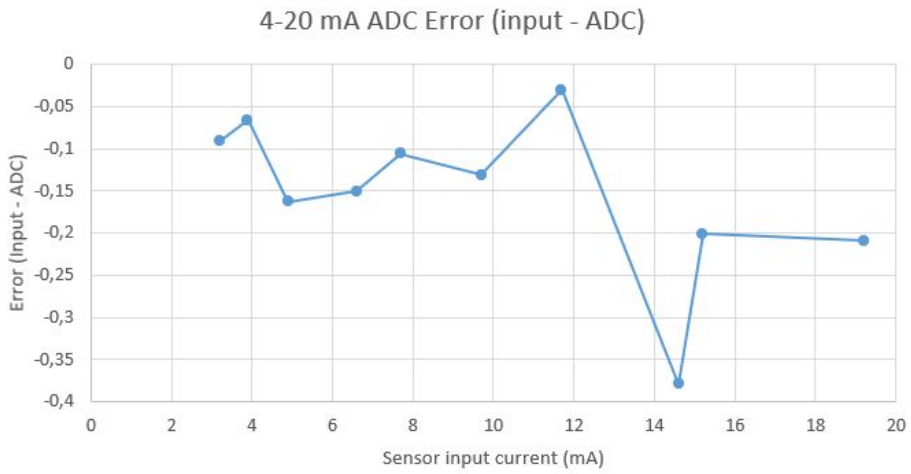


Figure 33 – 4-20 mA Sensor ADC Error

According to this graph, the maximum error is at around -0,35 mA's. This graph also shows there is no clear Gain error and thus does not require any other actions to increase performance.

9.8 To be tested hard- and soft-ware

Due to time constraint during the internship certain parts of the Xe-Link have not been tested yet. These parts need to be tested before the Xe-Link can be considered working and tested.

Currently these components are the communication over UART and the PT100 Analogue sensor.

9.8.1 Communication IoT Gateway

The communication with the IoT gateway has not been established and tested yet. For this particular sub-system of the Xe-Link is is important to measure how well the system will function under high noise area's. As well as how long cables it is able to support.

9.8.2 Analogue Sensor - PT100

The PT100 sensor has not been developed and tested yet either. The PT100 sensor needs to be tested on its base functionality first, by making a proper driver. After getting the PT100 measurements to work as expected, the reliability and linearity have to be tested.

10 Conclusion and recommendations

Since the internship has ended, although the project is not finished a conclusion with recommendations will be given. Informing the status of the project and what the recommendations are, from now on.

10.1 Conclusion

In conclusion, looking back at the entirety of the project, a lot of steps have been made. By looking back at the list of requirements given in appendix A, the status of the project can be seen and a conclusion can be given.

A table of the requirements can be found in table 7 and 8, this table will tell whether these requirements have been met, partially met or have not been met at all. Including notes when more information is needed.

#	Title	Requirement Met	Note
1.1.	Sensor Power	Partially	
1.1.a.	Sensor supply voltage	No	Currently capping at 21,6 Volts instead of 24 Volts
1.1.b.	Sensor supply current	Yes	
1.1.c.	Switching capablitity	Yes	
1.2.	Sensor Input	Partially	
1.2.a.	0-10 V sensor input	Yes	
1.2.b.	4-20 mA sensor input	Yes	
1.2.c.	PT-1000 sensor input	To be tested	Hardware designed, needs to be tested
1.2.d	Maximum input pins	Yes	Uses two pins on a 5 pin header. This requirement is considered succesfull since it is impossible to put everything on one pin.
1.3.	Gateway Power input	Partially	
1.3.a.	Accepted input voltage	Yes	
1.3.b.	Always on mode	To be programmed	
1.3.c.	ON/OFF mode	To be programmed	
1.4.	Gateway Communication	Partially	
1.4.a.	1.8 V TTL UART	To be tested	Hardware designed, needs to be tested
1.4.b.	Human readable	To be programmed	
1.4.c.	Event triggered	To be programmed	
1.4.d.	Error detection and correction	To be programmed	
1.4.e.	Variable message length	To be programmed	

Table 7 – Met requirements (table 1 of 2)

#	Title		Notes
2.1.	Enclosure	No	No enclosure has been selected and tested yet
2.2.	Cost	To be checked	The cost of the board has not been researched thoroughly, though is kept as cheap as possible during the design process
2.3.	Isolation	To be tested	

Table 8 – Met requirements (table 2 of 2)

Looking at both tables, a lot of requirements have been met partially or need to be programmed. The met requirements have been completely finished and are ready for the first deploy. The partial, to be tested and to be programmed stated requirements still need one or more actions before the requirement can be confirmed as functioning. The two requirements that are stated as not met at all are explained below.

Sensor supply voltage: This requirement is stated as not functional since it is not able to reach the correct voltage range. To meet this requirement a revisit to the hard- and soft-ware has to be made.

Enclosure: During the internship there was no time to design and find the correct enclosure. This is an action that needs to be done at the end of the entire development process of the product and thus does not need to be done right now.

10.2 Recommendations

Currently the project is not finished yet, as expected. The project still needs to be improved software wise significantly, as well as be tested thoroughly at some hardware parts.

A list of actions that still need to be performed are listed below, in order:

1. Improve boost-converter hard- and software to meet its requirements
2. Test PT-100 Schematic and change hard - and software if needed
3. Write software driver for UART communication
4. Implement the SSI-protocol to the UART communication and test the protocol thoroughly
5. Combine all functions and software as a whole functioning system, calling functions when needed

These given actions are needed to to get a functioning first version of the Xe-Link. After all the above tests have been done, the reliability has to be tested. Using several steps:

1. Order 4-20 mA, 0-10 V and PT100 Sensors to test the device with real sensors
2. Test the Xe-Link in the Xeelas environment, including back- and front-end software.
3. Assemble one or more Xe-Links to verify the hardware is functional all the time, or other fundamental problems occur.
4. Design, make and choose a correct enclosure

Concluding from these lists, a lot of actions still have to be performed, both long and short term. This is to get the Xe-Link Sense functional. The Xe-Link Serial however, needs to be developed entirely.

For now it is safe to say that the Xe-Link is potent and is worthy to be developed further. And thus the overall recommendation for the Xe-Link is to develop the Xe-Link Sense further, to see its full potential.

References

[1] Xeelas, "Xeelas portal." [Online]. Available: http://dev-portal.xeelas.nl/

[2] C. Electronics, "Lin bus explained - a simple intro." [Online]. Available: https://csselectronics.smoolis.com/screen/page/lin-bus-protocol-intro-basics/language/en

[3] T. Instruments, "Tlv61046a 28-v output voltage boost converter with power diode and isolation switch." [Online]. Available: http://www.ti.com/lit/ds/symlink/tlv61046a.pdf?&ts=1590049623093

[4] N. SDK, "Nodric info-centre." [Online]. Available: https://infocenter.nordicsemi.com/index.jsp?topic=%2Fsdk_nrf5_v16.0.0%2Findex.html

[5] P. group. Vendor member cost. [Online]. Available: https://profibusgroup.com/vendor-member-cost-savings/

[6] T. Instruments, "Io-link master reference design." [Online]. Available: http://www.ti.com/lit/ug/tidueb8b/tidueb8b.pdf?ts=1588262147568

[7] Nokia, "Ssi protocol specifications." [Online]. Available: http://www.janding.fi/iiro/papers/SSI%20protocol%20specification_12.pdf

[8] S. Consortium, "Scpi style and syntax rules." [Online]. Available: https://www.ivifoundation.org/docs/scpi-99.pdf

[9] ModBus.org, "Modbus speficiations and implementation guide." [Online]. Available: http://www.modbus.org/specs.php

[10] Thethingsnetwork, "Cayenne lpp guide." [Online]. Available: https://www.thethingsnetwork.org/docs/devices/arduino/api/cayennelpp.html

[11] D. Crockford, "How json works." [Online]. Available: https://www.json.org/json-en.html

[12] Steve, "Understanding the mqtt protocol packet structure." [Online]. Available: http://www.steves-internet-guide.com/mqtt-protocol-messages-overview/

[13] T. Instruments, "Tlv700 200-ma, low-iq, low-dropout regulator for portable devices." [Online]. Available: http://www.ti.com/lit/ds/symlink/tlv700.pdf?&ts=1589986938569

[14] ——, "Webench power designer." [Online]. Available: https://webench.ti.com/power-designer/switching-regulator

[15] ——, "Tina ti." [Online]. Available: http://www.ti.com/tool/TINA-TI

[16] ——, "Sn74lvc2t45 datasheet." [Online]. Available: http://www.ti.com/lit/ds/symlink/sn74lvc2t45.pdf?HQS=TI-null-null-digikeymode-df-pf-null-wwe&ts=1590068154088

[17] Nexperia, "74lvc1t45 datasheet." [Online]. Available: https://assets.nexperia.com/documents/data-sheet/74LVC_LVCH1T45.pdf

[18] Beamex, "Pt100 (385) temperature resistance relationship." [Online]. Available: https://blog.beamex.com/pt100-temperature-sensor

[19] N. Semiconductor, "nrf52832 product specifications." [Online]. Available: https://infocenter.nordicsemi.com/pdf/nRF52832_PS_v1.4.pdf

[20] ——, "nrf5284 product specifications." [Online]. Available: https://infocenter.nordicsemi.com/pdf/nRF52840_PS_v1.0.pdf

[21] Git-Kraken, "What is gitflow." [Online]. Available: https://support.gitkraken.com/git-workflows-and-extensions/git-flow/

A List of requirements

Functional requirements:

#	Title	Requirement	Description	Importance
1.1.	Sensor Power	The adapter must be capable of powering sensors that are used	The use of industrial sensors require the amount of power given in this requirement.	MUST HAVE
1.1.a.	Sensor supply voltage	The adapter must be able to provide a voltage of 5-24 Volts configurable by the user	The configurability provides the oppertunity to use more (efficient) sensors chosen by the users	MUST HAVE
1.1.b.	Sensor supply current	The adapter must be able to supply a current to the sensor of 0-50 mA	Some sensors use up to 50 mA of current	MUST HAVE
1.1.c.	Switching capablitity	The outpout power for the sensor must be able to turned on and off	To minimalize waisted energy, the supply has to be able to be turned off/on.	MUST HAVE
1.2.	Sensor Input	The adapter must be capable of measuring different analogue	This is the main funciton of the adapter, to measure external sensors with analogue inputs.	MUST HAVE
1.2.a.	0-10 V sensor input	0-10 V Sensors	0-10V industrial sensors are widely used and provided sensors that are easy to implement in industrial tasks.	MUST HAVE
1.2.b.	4-20 mA sensor input	4-20 mA Sensors	4-20 mA industrial sensors are widely used and provided sensors that are easy to implement in industrial tasks.	MUST HAVE
1.2.c.	PT-1000 sensor input	PT-1000 Sensors	PT-1000 industrial sensors are widely used and provided sensors that are easy to implement in industrial tasks.	MUST HAVE
1.2.d	Maximum input pins	measuring the analogue sensors must be done using a maximum of one input pin	To keep the amount of in/output pins of the adapter as low as possible, all inputs will be centralized on one pin.	MUST HAVE

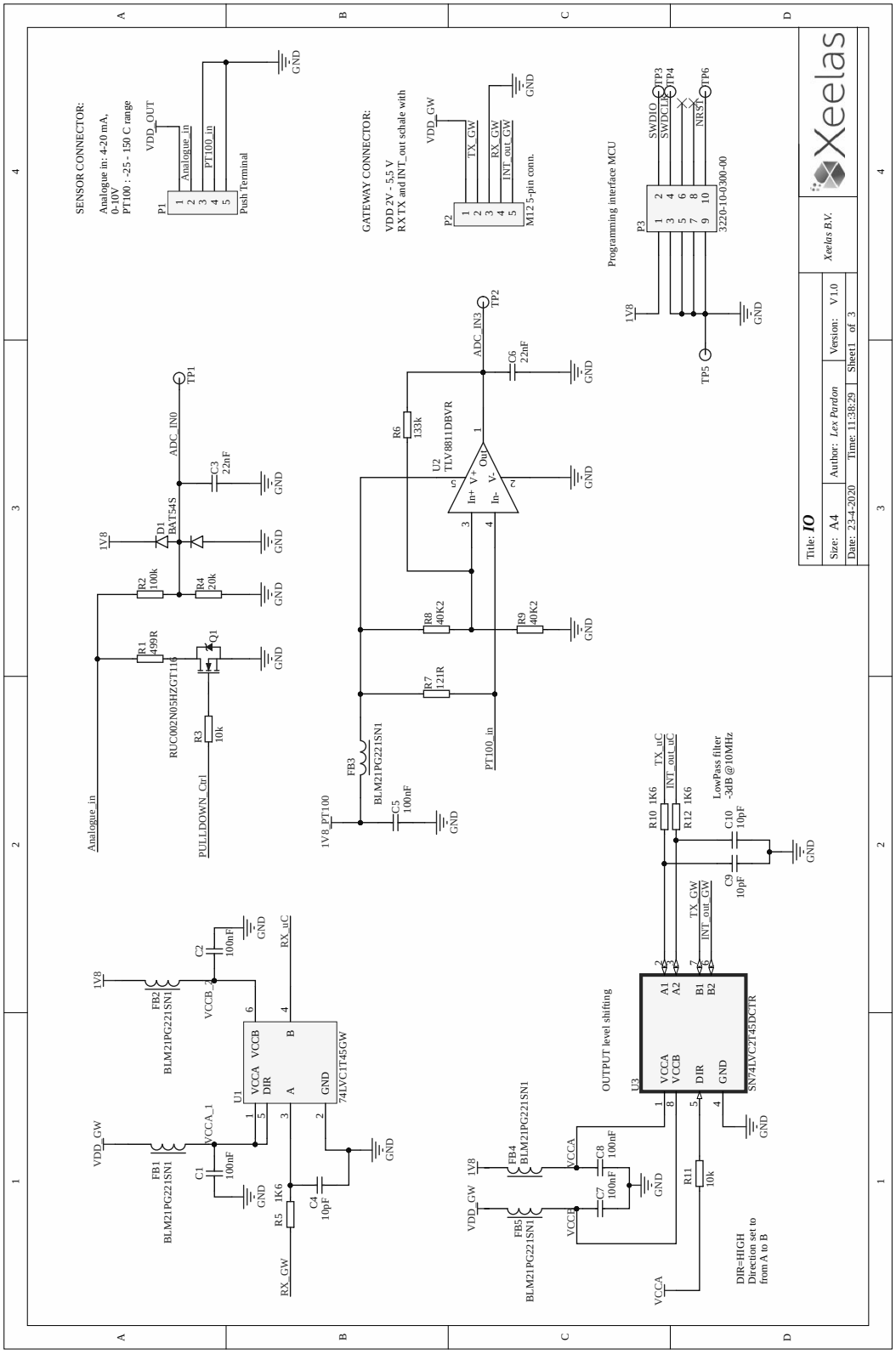
1.3.	Gateway Power input	The adapter is powered by the gateway	The adapter gets its power from the gateway that it is connected to, in different modes	**MUST HAVE**
1.3.a.	Accepted input voltage	The adapter must have an input voltage range between 1.8 to 5 Volts	To widen the range of inputs a range of 1.5 to 5 Volts input has been chosen	**MUST HAVE**
1.3.b.	Always on mode	The adapter should have an always on mode where the	Implementing two modes, where one is always on and the other is turned on and off when needed.	**SHOULD HAVE**
1.3.c.	ON/OFF mode	The adapter should have an on/off mode where the adapter is powered on to measure the sensor value and be turned off again.	To minimize the amount of energy used, whenever the adapter enters low-powered mode it needs to consume as little as current as possible.	**SHOULD HAVE**
1.4.	Gateway Communication	The adapter must communicate with IoT gateways.	A protocol between the IoT gateway and the adapter must be set up	
1.4.a.	1.8 V TTL UART	The adapter must communicate over TTL UART on 1.8 Volt	Since the adapter has to be able to connect to the Series S gateway, we're limited to TTL UART on 1,8 Volts	**MUST HAVE**
1.4.b.	Human readable	The protocol could be human readable	To simplify the implementation and usibility of the protocol the protocol could be human readable	**COULD HAVE**
1.4.c.	Event triggered	Should be capable of handling slave device events	The adapter will be able to measure the connected sensor every n time increment, whenever this value crosses a threshold it will trigger an event	**SHOULD HAVE**

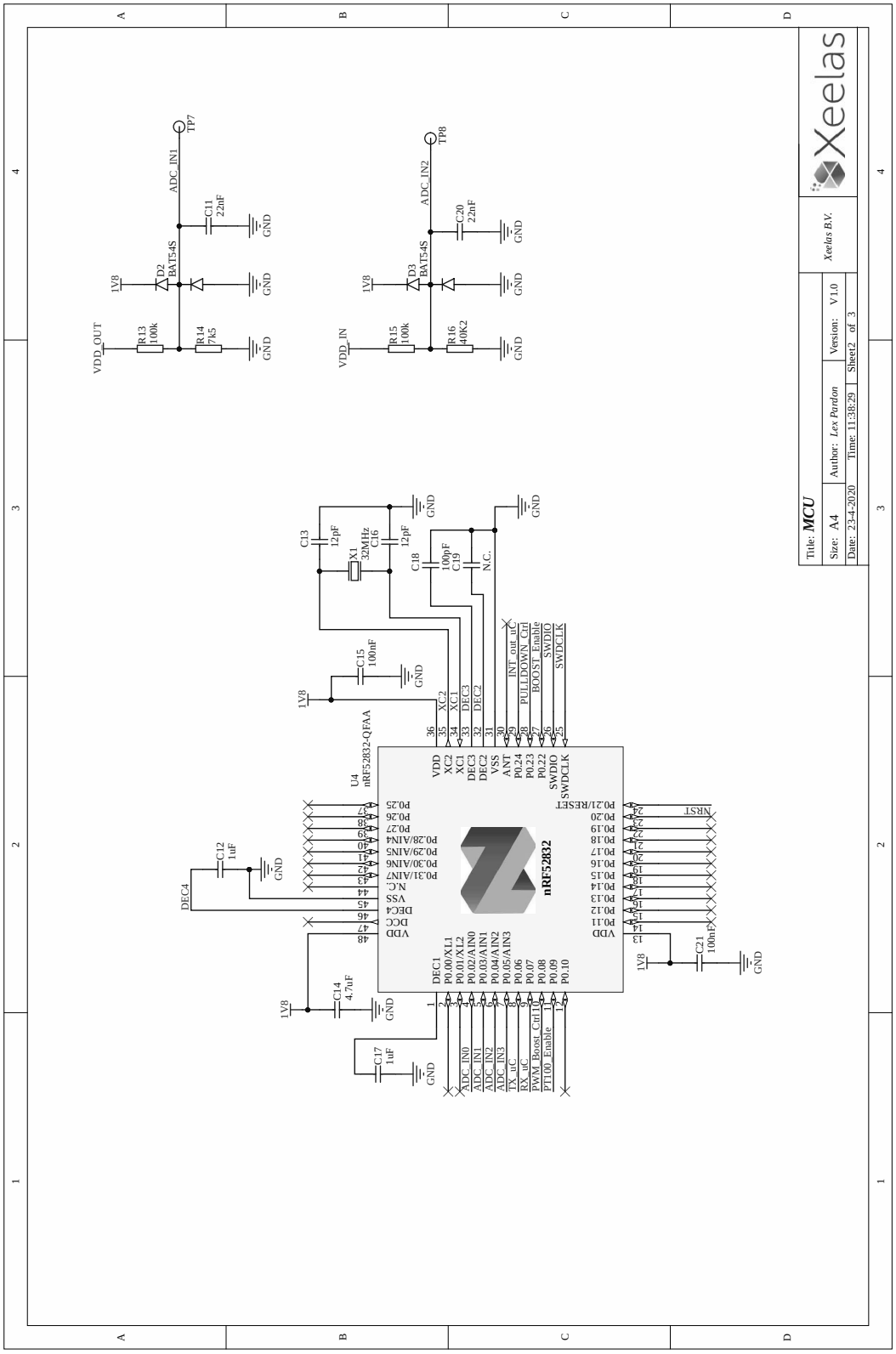
1.4.d.	Error detection and correction	The communication protocol should detect and correct errors when transmitting/receiving	The protocol should be capable of detecting and responding on communication errors when noise interferes.	SHOULD HAVE
1.4.e.	Variable message length	The communication protocol must have support for variable message lengths	To keep the flexibility of the software a variable message length is recommended.	MUST HAVE

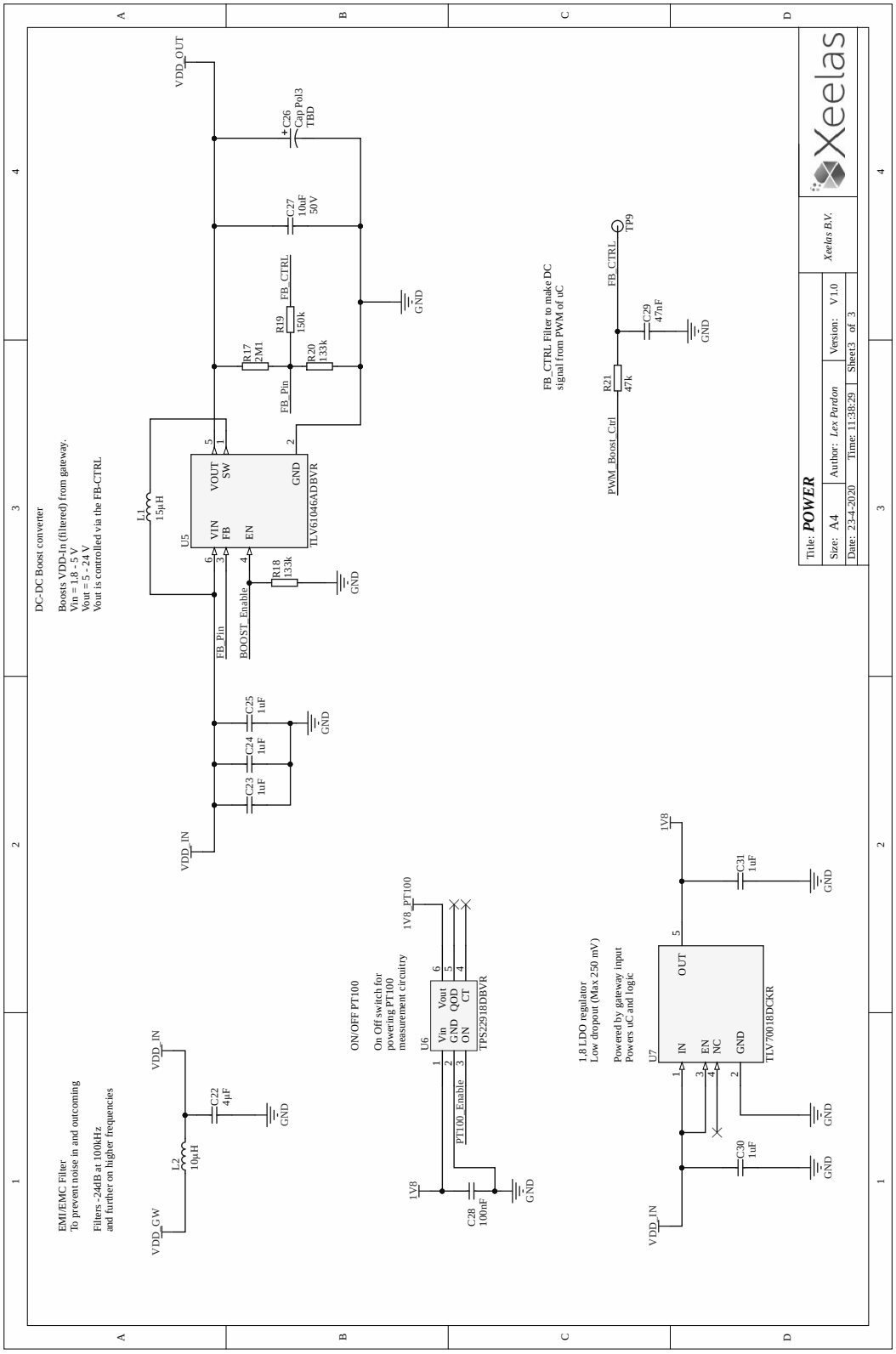
Non-functional requirements:

#	Title	Requirement	User Story	Importance
2.1.	Enclosure	The enclosure must be waterproofed with an IP67 casing	To maximize the situation the adapter can be used, a watertight enclosure is needed.	MUST HAVE
2.2.	Cost	The costs of the product must be below 25 euro's	To keep interests high in the adapter, the costs have to be as low as possible.	SHOULD HAVE
2.3.	Isolation	The adapter must be galvanically isolated	To prevent noise from and to the gateway.	MUST HAVE

B Schematics




Xeelas
Title: MCU
Size: A4
Author: Lex Pardon
Version: V1.0
Date: 23-4-2020
Time: 11:36:29
Sheet2 of 3
Xeelas B.V.
TP7
ADC_IN1
C11 22nF
D2 BAT54S
1V8
R13 100k
R14 7k5
VDD_OUT
GND
TP8
ADC_IN2
C20 22nF
D3 BAT54S
R15 100k
R16 40K2
VDD_IN
C13 12pF
X1 32MHz
C16 12pF
C18 100pF
C19 N.C.
C15 100nF
U4 nRF52832-QFAA
nRF52832
C12 1uF
DEC4
C14 4.7uF
C17 1uF
C21 100nF
NRST
INT_out_uC
PULLDOWN_Ctrl
BOOST_Enable
SWDIO
SWDCLK
ADC_IN0
ADC_IN1
ADC_IN2
ADC_IN3
TX_uC
RX_uC
PWM_Boost_Ctrl
PT100_Enable


DC-DC Boost converter
Boosts VDD-in (filtered) from gateway.
Vin = 1.8 - 5 V.
Vout = 5 - 24 V.
Vout is controlled via the FB-CTRL
EMI/EMC Filter
To prevent noise in and outcoming
Filters -24dB at 100kHz
and further on higher frequencies
ON/OFF PT100
On Off switch for
powering PT100
measurement circuitry
1.8 LDO regulator
Low dropout (Max 250 mV)
Powered by gateway input
Powers uC and logic
FB_CTRL Filter to make DC
signal from PWM of uC
VDD_OUT
VDD_IN
VDD_GW
1V8
1V8_PT100
PWM_Boost_Ctrl
FB_CTRL
FB_Pin
BOOST_Enable
PT100_Enable
GND
L1
15µH
L2
10µH
U5
TLV61046ADBVR
U6
TPS22918DBVR
U7
TLV70018DCKR
C22
4µF
C23
1uF
C24
1uF
C25
1uF
C26
Cap Pol3
TBD
C27
10uF
50V
C28
100nF
C29
47nF
C30
1uF
C31
1uF
R17
2M1
R18
133k
R19
150k
R20
133k
R21
47k
TP9
Title: POWER
Size: A4
Author: Lex Pardon
Version: V1.0
Date: 23-4-2020
Time: 11:38:29
Sheet 3 of 3
Xeelas B.V.
Xeelas

C Bill of Material

Quantity	Designator	Manufacturer	Manufacturer Part Number	Footprint Doc	Supplier	Supplier part number
8	C1, C2, C5, C7, C8, C15, C21, C28	Murata Electronics	GRM155R71H104ME14D	0402	Digikey	490-10701-1-ND
4	C3, C6, C11, C20	Yageo	CC0402KRX7R6BB223	0402	Digikey	311-3793-1-ND
3	C4, C9, C10	KEMET	C0402C100C8GACTU	0402	Digikey	399-8939-1-ND
7	C12, C17, C23, C24, C25, C30, C31	Samsung Electro-Mechanics	CL05A105MP5NNNC	0402	Digikey	1276-1448-1-ND
2	C13, C16	KEMET	C0402C120J8GAC7867	0402	Digikey	399-14417-1-ND
1	C14	Murata	GRM155R61A475MEAAD	0402	Digikey	490-14306-1-ND
1	C18	KEMET	C0402C101K8GAC7867	0402	Digikey	399-14413-1-ND
1	C19			0402		
1	C22	KEMET	C0805C395K8PACTU	1210	Digikey	399-3131-1-ND
1	C26	TBD	TBD	Radial, BIG	TBD	TBD
1	C27	Samsung Electro-Mechanics	CL32B106KBJNNWE	1210	Digikey	1276-3388-1-ND
1	C29	Samsung Electro-Mechanics	CL05B473KP5NNNC	0402	Digikey	1276-1573-1-ND
3	D1, D2, D3	Diodes Incorporated	BAT54S-7-F		Digikey	BAT54S-FDICT-ND
5	FB1, FB2, FB3, FB4, FB5	Murata	BLM21PG221SN1		Digikey	490-1054-1-ND
1	L1	Bourns	SDR0403-150ML		Digikey	SDR0403-150MLCT-ND
1	L2	Bourns	SRN5040TA-100M		Digikey	SRN5040TA-100MCT-ND
1	P1	Phoenix Contacts	1770911	2.5MM	Digikey	277-2082-1-ND
1	P2		M12-4/5P		Alibaba	
1	P3	CNC Tech	3220-10-0300-00	HDR_2x5-SMD-1.27mm	Digikey	1175-1629-ND
1	Q1	Rohm	RUC002N05HZGT116	SOT-23	Digikey	RUC002N05HZGT116CT-ND
1	R1	Vishay Dale	CRCW0402499RFKEDC	0402	Digikey	541-4007-1-ND
3	R2, R13, R15	Yageo	RC0402JR-07100KL	0402	Digikey	311-100KJRCT-ND
2	R3, R11	Yageo	RC0402JR-0710KL	0402	Digikey	311-10KJRCT-ND
1	R4	Yageo	RC0402FR-0720KL	0402	Digikey	311-20.0KLRCT-ND
3	R5, R10, R12	Yageo	RC0402JR-071K6L	0402	Digikey	YAG3280TR-ND
3	R6, R18, R20	Yageo	AC0402FR-07133KL	0402	Digikey	YAG5259CT-ND
1	R7	Yageo	RC0402FR-07121RL	0402	Digikey	311-121LRCT-ND
3	R8, R9, R16	Yageo	RC0402FR-0740K2L	0402	Digikey	311-40.2KLRCT-ND
1	R14	Yageo	RC0402JR-077K5L	0402	Digikey	311-7.5KJRTR-ND
1	R17	Vishay Dale	CRCW04022M10FKED	0402	Digikey	541-2.10MLCT-ND
1	R19	Yageo	RC0402JR-07150KL	0402	Digikey	311-150KJRCT-ND
1	R21	Yageo	RC0402JR-0747KL	0402	Digikey	311-47KJRCT-ND
1	U1	Nexperia	74LVC1T45GW,125		Digikey	1727-4560-1-ND
1	U2	Texas Instruements	TLV8811DBVR	SOT-23-5	Digikey	296-47362-1-ND
1	U3	Texas Instruments	SN74LVC2T45DCTR		Digikey	296-16845-1-ND
1	U4	Nordic Semiconductor	nRF52832-QFAA	QFN-48	Digikey	1490-1052-1-ND
1	U5	Texas Instruments	TLV61046ADBVR		Digikey	296-47674-1-ND
1	U6	Texas Instruments	TPS22918DBVR	SOT-23-6	Digikey	296-46278-1-ND
1	U7	Texas Instruments	TLV70018DCKR		Digikey	296-32409-1-ND
1	X1	Epson Toyocom	Q22FA12800025	XTAL_2016	Digikey	SER4233CT-ND

D Use-Case flow charts

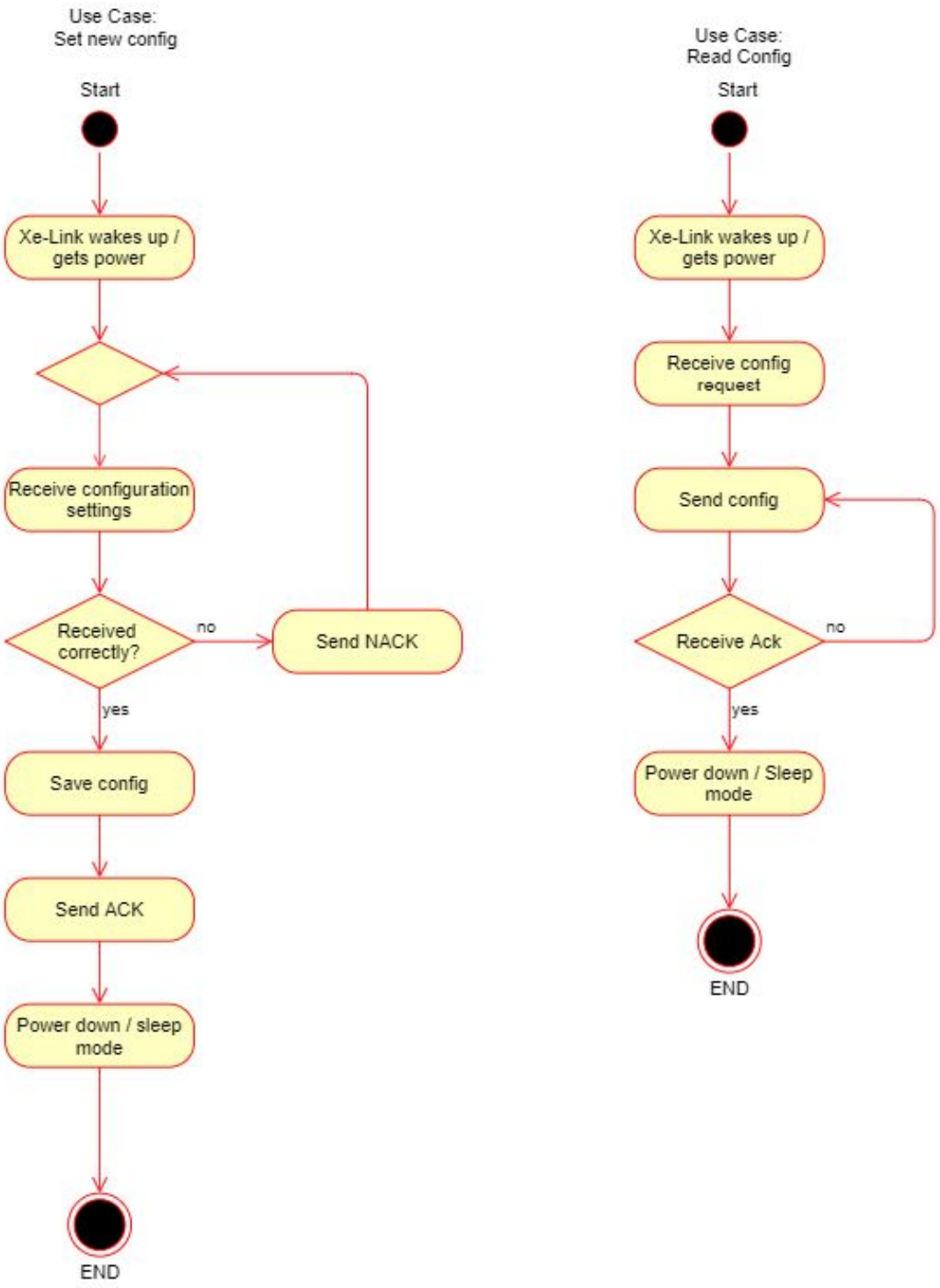


Figure 34 – Flowcharts of the Xe-Link Use cases (p. 1 of 3)

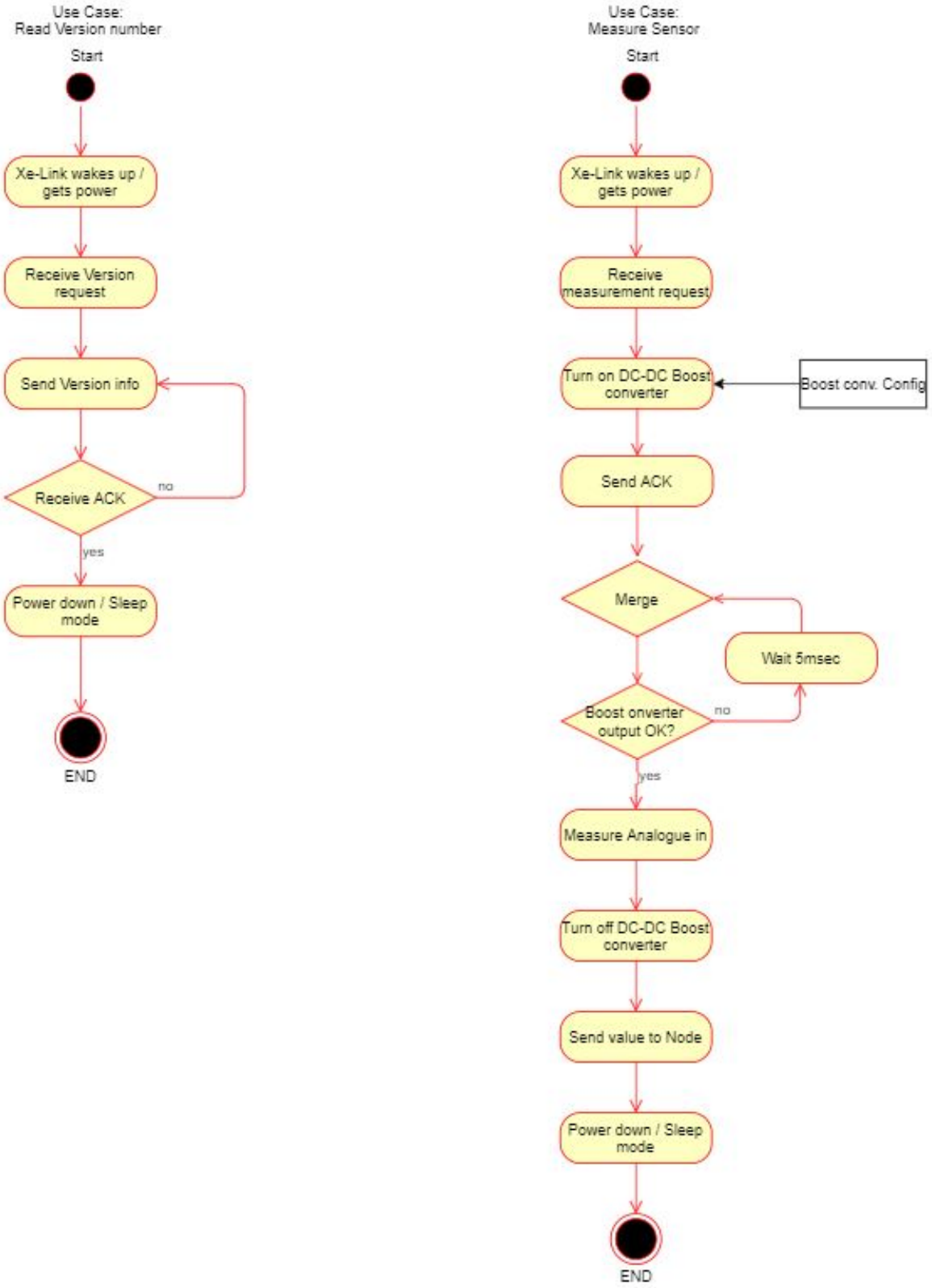


Figure 35 – Flowcharts of the Xe-Link Use cases (p. 2 of 3)

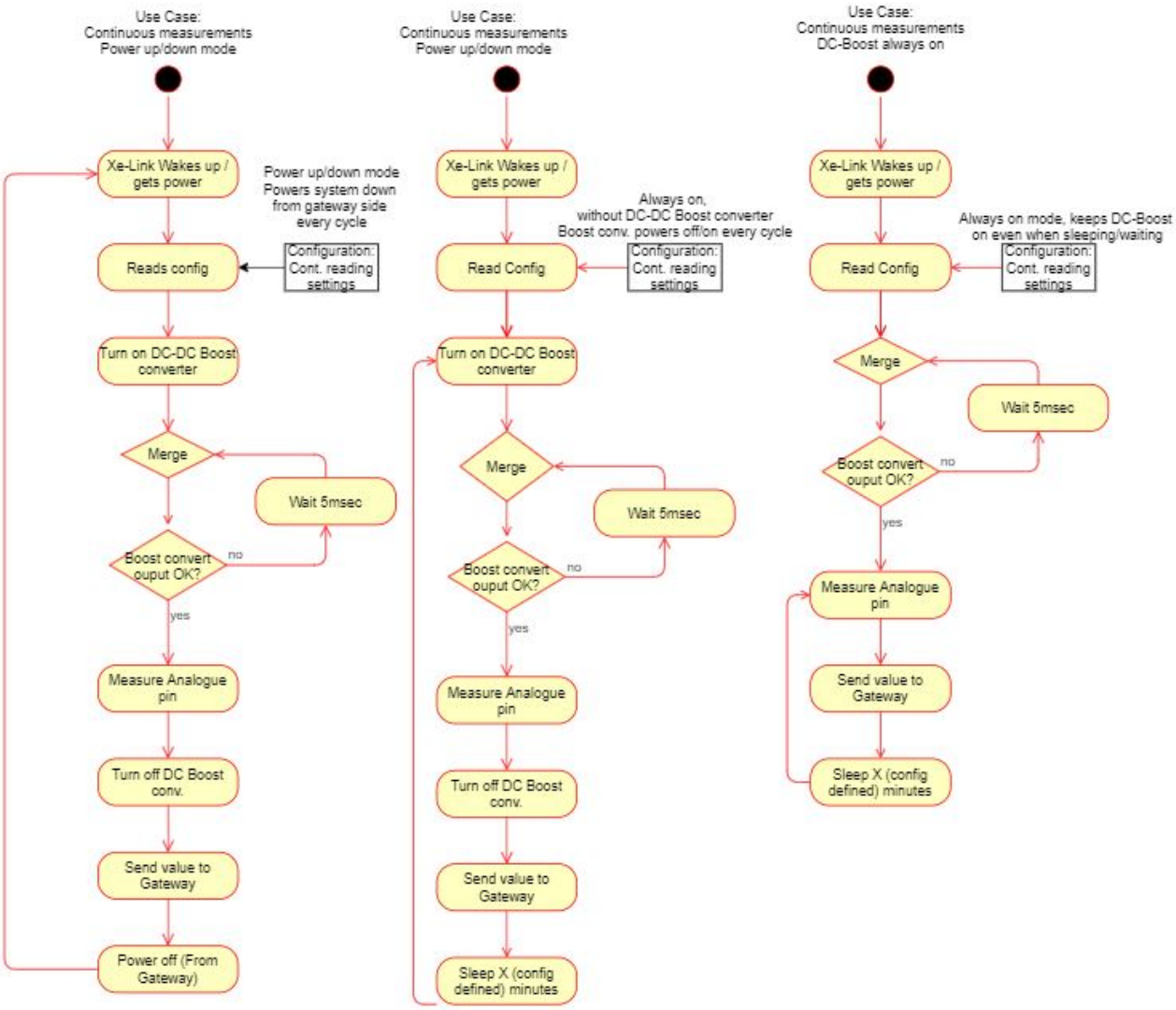


Figure 36 – Flowcharts of the Xe-Link Use cases (p. 3 of 3)

E Competences

At the end of the internship, the student has to give evidence and show that he or she has achieved and used given competences. These competences give a clear understanding of what the student was capable of during the internship.

During the internship five competences have to be used and improved to certain levels. The five competences are thoroughly explained below, with examples of when these competences have been used and how that effected the project.

Competence: Analysis - Level 3

During the internship the competence analysis is mainly used to give a clear view on what the ending product must be and be capable of. By first analysing the problem to be solved as seen in chapter 3 and then deriving a list of requirements from these problems as seen in chapter 3.2 this competence has been taken in use intensively to give a better understanding of the project.

Complexity of the problem
Since the project itself was un-defined at the beginning, a lot of unclear aspects were still undefined. To give a more clear overview several steps have been made to get a good understanding of the problem and what the project had to be capable of. Thus implying the complexity of this project and competence are Level 3 (Complex).

Competence: Design - Level 3

During the project, multiple designs had to be made. Starting from scratch with just a few requirements. Stating that the start was quite unclear and had to be cleared during the internship. After finding exactly out what was needed, the design process of both the schematic and PCB had to be made. All examples can be seen in chapter 5 and 6.

Complexity of the problem
Again, due to the projects un-clearness at the beginning with a lot of undefined aspects the complexity of the project was high. By slowly converting these unclear aspects to known parts of the projects the project slowly became solvable. Thus resulting in a complexity level 3 (Complex)

Nature of context
Several to be designed parts of the circuit have not been designed before, implying several unknown parts of the Xe-Link were unknown and complex.

Competence: Realisation - Level 3

During the internship the competence realisation came into play in almost every aspect. The result of the internship is a device that is able to be programmed and has been assembled by the student. During the assembly state of the project, several actions have been performed to have a resulting working prototype. From generating a BoM from the circuit, ordering this according to the company's standard and eventually assembling the device so that it can be tested. As seen in chapter 7.

Competence: Managing - Level 2

During the internship, a certain degree of managing has been applied. Not spoken of in the report, a planning beforehand has been made. The so called Plan of Approach included a weekly planning, general way of work and rough list of requirements to follow before a more definite version of the list of requirements was set up. In appendix F a Gannt-type planning can be seen, which is made at the beginning of the project.

This planning however, got changed multiple times due to fall-backs during the project. During the project the planning provided a well defined schedule.

Competence Research - Level 2

At the start of the internship, the research phase started. During research several aspects and possibilities of the project have been researched. As seen in chapter 4, several parts of the Xe-Links possibilities have been gone over to see their relevance and whether it is even possible to add certain functions or sensors to the Xe-Link.

During the research phase several sensors or systems have been chosen to not be used, or to be used. Resulting in the project being in the shape it currently is. As well as the choice to split the Xe-Link into two different versions, the sense and serial. All these decisions are made after extensive researching which can be found in chapter 4.

Nature of Context
During the research phase certain to be researched parts of the Xe-Link have been researched that have not been researched before. This implies the level of complexity of the context that has been gone through.

Independence of the student during the internship

During the internship, a virus called COVID-19 Broke out. Resulting in a work-from-home environment. This environment resulted in limited contact with co-workers and the internship mentor. Thus increasing the Independence of the student drastically. Since one of the requirements for level 3 competences is independence of the student, this competence level in any of the five competences is guaranteed.

F Gannt planning