

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO



BALU: a modular computational framework for maritime vehicle systems

Ricardo Manuel de Carvalho Almeida Falcão

Mestrado em Engenharia Eletrotécnica e de Computadores

Supervisor: João Borges Sousa

July 1, 2022

Abstract

The currently available off-the-shelf computational solutions concerning the development of maritime vehicle systems are often tailored to a specific application and lack the flexibility to be further upgraded. This leads to the establishment of legacy and out-of-date hardware architectures, which not only results in an increase in the project's cost and complexity but also prevents future development and implementation of modern technologies.

When developing engineering projects, there is often a set of common requirements concerning the hardware peripherals' availability or the processing capabilities. However, the need to incorporate additional features on a project-level - such as a newer revision of a communications modem or the requirement of a high-speed storage solution - leads to the establishment of custom hardware solutions per project.

The problem of developing a scalable, modular, energy-efficient, and processor independent hardware framework with applications to unmanned maritime vehicles is addressed in this dissertation. The challenges posed by the development of a circuit board are discussed to provide the necessary background for this work. These challenges include the need to guarantee the reliability of high-speed lines, such as clock signals and their routing. A set of design rules is investigated to minimize these electromagnetic interference problems.

The development of the concrete requirements is based on a targeted questionnaire to the hardware and firmware teams at LSTS, to discover the aspects which require the most effort and attention.

The proposed solution is a variable-sized circuit board architecture, code-named BALU, that guarantees scalability and modularity via specialized hardware expansion modules, via the M.2 connector.

To demonstrate the architecture, a Main Board capable of yielding 9 expansion modules was designed and manufactured. To validate its modularity, 4 commercially available expansion cards are tested - a Solid-State Drive, a Wi-Fi and Bluetooth modem, and two Cellular modems. In addition to that, 3 custom expansion modules were also designed and tested - a GPS modem, a Iridium modem, and a 4-channel RS232 expansion module.

In order to accomplish all the requirements and prevent delays, the development and manufacturing process was divided into two phases, each tested separately through a verification and validation process.

The final product is an elegant solution to the challenges in the hardware design of maritime systems, that provides an almost cable-free implementation, increasing reliability and maintainability. The expansion module design around a standard connector allows the usage of commercial modules that would otherwise be impossible to easily integrate in the target applications. The small form-factor of the expansion card system could prove to be beneficial in future engineering projects with tighter space constraints.

Resumo

As soluções computacionais no desenvolvimento de sistemas de veículos marítimos são muitas vezes desenhadas especificamente para uma aplicação e não apresentam muita flexibilidade de atualizações futuras. Isto leva ao estabelecimento de arquiteturas de hardware que se tornam antiquadas e desatualizadas, o qual resulta não só num aumento do custo e da complexidade do projeto, mas também impede futuras revisões e a implementação de tecnologias modernas.

Esta dissertação aborda o desenvolvimento de uma estrutura de hardware escalável, modular, energeticamente eficiente e independente do processador, com aplicações em veículos marítimos não tripulados. Para fornecer as bases necessárias para este trabalho, são discutidos os problemas existentes no desenvolvimento de placas de circuito impresso. Esses desafios incluem a necessidade de garantir a fiabilidade na comunicação de linhas de alta velocidade, como sinais de clock e o seu roteamento. Para minimizar estes problemas de interferência eletromagnética, também é investigado um conjunto de boas práticas no processo de desenho.

O desenvolvimento dos requisitos é baseado num questionário que foi direcionado às equipas de hardware e firmware do LSTS, para descobrir os aspectos que exigem mais esforço e atenção.

A solução proposta é uma arquitetura composta por uma placa de circuito impresso de tamanho variável, designada por BALU, que garante a escalabilidade e modularidade do sistema através de módulos de expansão de hardware especializados, com o conector M.2.

Para demonstrar a arquitetura, foi projetada e fabricada uma Main Board com capacidade para 9 módulos de expansão. Para validar a sua modularidade, 4 placas de expansão comerciais são testadas - um Solid-State Drive (SSD), um modem Wi-Fi e Bluetooth e dois modems de comunicações móveis. Para além disso, 3 módulos de expansão personalizados também foram projetados e testados - um modem GPS, um modem Iridium e um módulo de expansão de RS232 de 4 canais.

Para atender a todos os requisitos e evitar atrasos, o processo de desenvolvimento e fabrico foi dividido em duas fases, cada uma testada separadamente através de um processo de verificação e validação.

O produto final é uma solução elegante para projetos de hardware de sistemas marítimos, que proporciona uma implementação quase livre de cabos, aumentando a fiabilidade e a manutenção do sistema. O design de módulos de expansão em cima de um conector standard permite a utilização de produtos comerciais que, de outra forma, seriam impossíveis de integrar facilmente nos projetos alvo. O reduzido tamanho das placas de expansão também pode ser benéfico em futuros projetos de engenharia com restrições de espaço mais apertadas.

Acknowledgments

I would like to express my sincere gratitude to my supervisor, Prof. João Borges Sousa, for all his help, advice and support, during the preparation of this dissertation.

I am very much indebted to the LSTS team, for their constant support and recommendations. They provided the perfect environment for the development of this dissertation, and were crucial when troubleshooting some of its issues. I am specially grateful to Jorge Neiva, for his guidance and the enthusiasm that he has always shown for this project.

Finally, I wish to thank my family and colleagues for their friendly support. A very special thank you is included to my mother and to Diana for their patience and understanding, during the entire period of study and preparation of this work.

Ricardo Falcão

“Long is the road from conception to completion.”

Molière

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Trends in SBCs	2
1.3	Objectives	3
1.4	Approach	3
1.5	Contributions	4
1.6	Outline of the document	4
2	Background	7
2.1	Autonomous Underwater Vehicles	7
2.1.1	Light Autonomous Underwater Vehicle	7
2.1.2	Iver	9
2.2	Operational deployments	10
2.3	M.2 Specification	11
2.3.1	Terms and Definitions	11
2.3.2	Mechanical Specification	11
2.3.3	Electrical Specification	13
3	State of the art	15
3.1	Challenges in the design of PCBs	15
3.2	Communication interfaces	17
3.3	Commercial off-the-shelf solutions	19
3.3.1	Rhodeus PC/104	19
3.3.2	Raspberry Pi 4	20
3.3.3	NVIDIA Jetson Nano Developer Kit	20
3.3.4	Compute Module 4 IO Board	21
3.3.5	Turing Pi 2	22
3.3.6	Alftel x12	23
4	Problem specification	25
4.1	Questionnaire	25
4.2	Analysis	25
4.3	Requirements	26
5	Approach	29
5.1	Design Principles	29
5.2	Architecture	30
5.3	Detailed design of the Main Board	31

5.3.1	Processing Unit	31
5.3.2	IO	32
5.3.3	Modules	38
5.3.4	Power	47
5.4	Detailed design of the Expansion Modules	49
5.4.1	GPS Module	49
5.4.2	Iridium Module	52
5.4.3	Serial Module	54
6	Implementation	61
6.1	Main Board	62
6.1.1	Operating System	62
6.1.2	USB validation	64
6.1.3	Ethernet validation	66
6.1.4	UART validation	67
6.2	Expansion Modules	69
6.2.1	Solid-State-Drive	69
6.2.2	Wi-Fi and Bluetooth modem	71
6.2.3	Cellular modem	73
6.2.4	GPS module	77
6.2.5	Iridium module	78
6.2.6	Serial module	79
6.3	Discussion of results	79
7	Conclusion	81
A	Questionnaire	83
	References	89

List of Figures

1.1	Examples of SoM	2
2.1	LAUVs developed at LSTS	8
2.2	The Iver vehicles	10
2.3	M.2 Module Sizes	12
3.1	Connectors	18
3.2	The PCIe Mini Card connector	18
3.3	The M.2 connector	18
3.4	RHODEUS PC/104	19
3.5	Raspberry Pi 4	20
3.6	NVIDIA Jetson Nano Developer Kit	21
3.7	Compute Module 4 IO Board	21
3.8	Turing Pi 2	22
3.9	Alftel x12 Carrier Board	23
5.1	BALU architecture	30
5.2	Daisy-chaining rows	31
5.3	Additional IO in Main Board design	33
5.5	Schematic of USB ports	34
5.6	Schematic of Ethernet port	34
5.7	Schematic of Micro-SIM port	35
5.8	Schematic of SD Card slots	36
5.9	Schematic of USB-C port	37
5.10	USB hub schematic (USB2514B)	39
5.11	UART multiplexer block diagram (4-port)	40
5.12	Comparison of UART multiplexers logic levels	42
5.13	Block diagram of MAX14830 UART multiplexer	42
5.14	Schematic of MAX14830 UART multiplexer	43
5.15	PCIe switch block diagram (3-port)	44
5.16	Comparison of IC packages	45
5.17	Schematic of PCIe switch power	45
5.18	Schematic of PCIe switch data lines	46
5.19	Schematic of PCIe switch configuration pins	46
5.20	Schematic of PCIe switch miscellaneous pins	47
5.21	Block diagram of power supplies	48
5.22	Schematic of GPS Module connector	50
5.23	Schematic of GPS Module power supply	50
5.24	Schematic of GPS Module modem	50

5.25	Schematic of GPS Module TIMEPULSE signal output	51
5.26	PCB of GPS Module	51
5.27	Schematic of Iridium Module connector	53
5.28	Schematic of Iridium Module Boost converter	53
5.29	Schematic of Iridium Module modem	54
5.30	PCB of Iridium Module	55
5.31	Schematic of Serial Module connector	56
5.32	Schematic of Serial Module multiplexer	57
5.33	Schematic of Serial Module RS232 transceivers	58
5.34	PCB of Serial Module	59
6.1	First revision of the Main Board	61
6.2	Final revision of the Main Board	63
6.3	Ethernet performance tests	67
6.4	Toshiba RC100 SSD	70
6.5	Performance comparison between SSD and internal eMMC	70
6.6	AX200 Wi-Fi and Bluetooth modem	71
6.7	AX200 performance tests	72
6.8	ME906S Cellular module	73
6.9	EM06-E Cellular module	75
6.10	Position reported by the GPS module	78

List of Tables

2.1	M.2 Keys	13
3.1	Comparison of the throughput of popular communication protocols	17
4.1	Requirement Analysis of Main Board	27
4.2	Requirement Analysis of Expansion Modules	27
5.1	Comparison of the commercially available SoM	32
5.2	Comparison of the commercially available USB-Hub ICs	38
5.3	Comparison of the commercially available UART Multiplexer ICs	41
5.4	Comparison of the commercially available PCIe switch ICs	44
5.5	Comparison of the commercially available Iridium modems	52
5.6	Comparison of the commercially available USB-UART Multiplexer ICs	55

Acronyms

AUV Autonomous Underwater Vehicle. [1](#), [7](#), [9](#)

CPU Central Processing Unit. [2](#), [4](#), [17](#), [23](#), [31](#), [62](#), [81](#), [82](#)

GPU Graphics Processing Unit. [20](#), [21](#), [32](#)

IC Integrated Circuit. [17](#), [38](#), [39](#), [40](#), [41](#), [43](#), [44](#), [45](#), [46](#), [47](#), [49](#), [52](#), [53](#), [55](#), [56](#), [59](#)

IoT Internet of Things. [1](#), [2](#)

LAUV Light Autonomous Underwater Vehicle. [xi](#), [1](#), [7](#), [8](#), [9](#), [82](#)

LSTS Laboratório de Sistemas e Tecnologias Subaquáticas. [xi](#), [1](#), [2](#), [3](#), [4](#), [7](#), [8](#), [10](#), [19](#), [25](#), [81](#), [82](#)

PCB Printed Circuit Board. [3](#), [4](#), [15](#), [26](#), [51](#), [54](#), [59](#), [65](#)

PCIe Peripheral Component Interconnect Express. [13](#), [14](#), [15](#), [17](#), [18](#), [22](#), [23](#), [29](#), [31](#), [43](#), [46](#), [49](#), [62](#), [69](#), [71](#), [81](#)

RAM Random-Access Memory. [2](#)

SBC Single-Board Computer. [2](#), [4](#), [15](#), [19](#), [20](#)

SDIO Secure Digital Input Output. [15](#)

SoC System on a Chip. [2](#), [19](#), [25](#)

SoM System on a Module. [xi](#), [2](#), [82](#)

SPI Serial Peripheral Interface. [15](#), [40](#), [41](#), [55](#), [67](#), [79](#)

SSD Solid-State Drive. [i](#), [iii](#), [29](#), [69](#), [79](#)

UART Universal Asynchronous Receiver-Transmitter. [14](#), [31](#), [40](#), [41](#), [49](#), [52](#), [55](#), [56](#), [62](#), [67](#), [78](#), [79](#)

UIM User Identity Module. [18](#), [29](#), [35](#), [49](#), [62](#)

USB Universal Serial Bus. [14](#), [15](#), [17](#), [18](#), [22](#), [29](#), [31](#), [33](#), [37](#), [38](#), [39](#), [40](#), [41](#), [49](#), [51](#), [55](#), [59](#), [62](#), [64](#), [65](#), [66](#), [67](#), [71](#), [73](#), [77](#)

UUV Unmanned Underwater Vehicle. [7](#)

Chapter 1

Introduction

In this chapter, we discuss the motivation behind the dissertation, its main objectives, as well as the structure of this document.

1.1 Motivation

This dissertation is motivated by the developments in ocean-going vehicles undertaken by the Laboratório de Sistemas e Tecnologias Subaquáticas (LSTS) at Porto University.

The LSTS is an interdisciplinary research laboratory established in 1997 with researchers drawn from Electrical and Computer Engineering, Mechanical Engineering and from Computer Science, specialized in the design, construction, and operation of unmanned underwater, surface, and air vehicles and on the development of tools and technologies for the deployment of networked vehicle systems [1]. The LSTS has been involved in fostering and growing a worldwide research network in the area of networked vehicle systems with yearly conferences and workshops.

Nowadays, the LSTS is one of the world-leading laboratories in the field of networked air, underwater, and surface vehicles. The LSTS also spun off the Oceanscan Marine Systems and Technologies company that commercializes the LSTS award-winning Light Autonomous Underwater Vehicle (LAUV). Over 70 LAUVs have seen operations worldwide.

The currently available off-the-shelf solutions concerning the development of maritime vehicle systems are often tailored to a specific application and lack the flexibility to be further upgraded. This leads to the establishment of legacy and out-of-date hardware architectures, which not only results in an increase in the project's cost and complexity but also prevents future development and implementation of modern technologies.

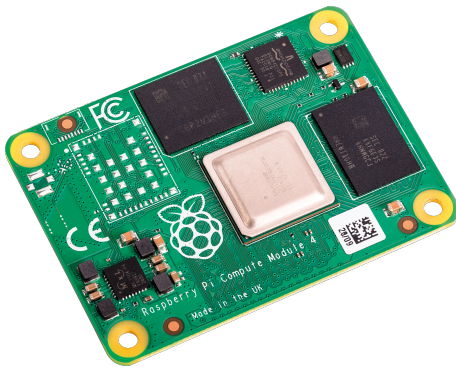
The challenge here is to create a new and modern hardware framework that will be the foundation of computational systems for Autonomous Underwater Vehicle (AUV), *Autonomous Surface Vehicles*, *Remotely Piloted Vehicles*, and *Communication Gateways* at LSTS.

In our vision for future maritime operations, we will have multiple nodes interacting over inter-operating communication networks, forming autonomous robotic organizations to perform complex tasks in the ocean. This is very much in line with the Internet of Things (IoT) concept,

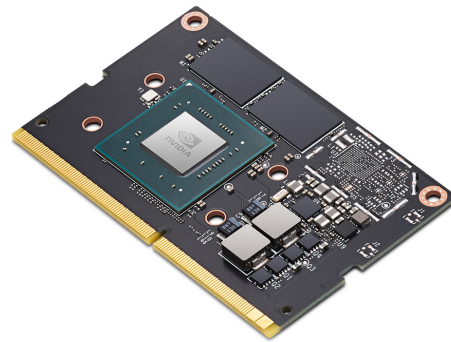
which has gained a lot of popularity in recent years. As defined in [2], IoT is a "cloud" that connects various devices - an array of intelligently connected devices and systems interacting and communicating. If the foundation of these devices is the same, the integration and interaction of these systems into new and modern inter-operating networks will be more manageable and maintainable.

1.2 Trends in SBCs

Single-Board Computers (SBCs) are becoming increasingly popular in recent years. An SBC is a computer built on a single circuit board, featuring a microprocessor (CPU), memory (RAM), and input/output ports (IO). These components are included in a single monolithic chip, the System on a Chip (SoC), which can be directly used by the SBC or be included in a smaller board-level circuit called System on a Module (SoM). The result is a low-cost, compact solution with various levels of processing capabilities and energy consumption [3]. Figure 1.1 shows two examples of SoM:



(a) Raspberry Pi Compute Module 4 [4]



(b) NVIDIA Jetson Nano [5]

Figure 1.1: Examples of SoM

SBCs are often considered when developing an engineering project, especially when it comes to the development of unmanned vehicle systems because of tight space and power constraints. However, finding an acceptable solution to the problem in the market is sometimes impossible, which leads to the need to create custom hardware solutions. For example, the maritime vehicles developed at LSTS require a lot of RS232 ports, and pretty much any offer in the market does not offer enough. While most manufacturers focus on delivering as many features as possible, they are often constrained and not scalable. On top of that, the market for the SoCs and SoMs is also sometimes limiting: it might be unfeasible to find a processor that checks all the requirements and has a low cost or, even worse, sometimes it is impossible to fulfill all of the requirements at the same time (for example, there is no processor with 10 serial ports (requirement) that respects the required energy consumption limit). These issues lead to an increase in time and resources spent not only on developing the project but also on testing and maintenance.

Most of the time, a processor with adequate computational power might be considered inappropriate because it does not meet the number of peripherals needed, the target energy consumption, or even the form factor. Instead of opting for a higher-end alternative, which often means an unnecessary higher computational power at the cost of higher power consumption, a better solution would be to extend the functionality of an adequate processor.

While this can sometimes be achieved through the usage of adapters or related systems, most of the time, it translates into an increase in cabling and complexity of the design, which decreases maintainability and is more error-prone.

1.3 Objectives

When developing engineering projects, there is often a set of common requirements concerning the hardware peripherals' availability or the processing capabilities. However, the need to incorporate additional features on a project level - such as a newer revision of a communications modem or the requirement of a high-speed storage solution - leads to the establishment of custom hardware solutions per project. The main objective of this work is to propose a framework to standardize hardware architectures across multiple projects and simultaneously provide each one with a way to scale individually. It should be CPU-independent but also modular, meaning that the features and peripherals the system offers can be added and removed. The concept shares many characteristics with the traditional computer architecture: it is composed of the *motherboard*, and each end user customizes the target product through expansion cards.

The adopted solution is a Printed Circuit Board (PCB) of the Main Board, capable of extending the target processor's peripherals via specialized expansion modules. As a demonstration, three custom-made expansion modules were designed, and four commercially available cards were validated.

1.4 Approach

In order to assess the technical requirements of the project, a questionnaire was developed and shared with the LSTS team. From the feedback of the firmware and hardware teams, a set of requirements was established and a priority assigned to each one. This is crucial to determine the best path of development and the aspects which require the most effort and attention.

With that in mind, it is unfeasible to design all the circuit boards with all the target features simultaneously. It is necessary to divide the design process into multiple phases, each with a functional prototype and increased functionality from the previous phase.

This iterative process guarantees that the system is still functional if some features are missing from the finished product. It also simplifies the testing process, by testing the system and patching the issues at the end of each phase instead of allocating a testing phase at the end of the development process.

However, while the design process of this dissertation followed the iterative process explained above, the current ongoing semiconductor crisis and long fabrication time of the PCBs restrained the number of prototypes to two. Nevertheless, even though it lacked some functionality, the first prototype allowed for validating the architecture's features and patching major issues in the circuit board's designs.

1.5 Contributions

This work's main contributions are:

- An overview of the M.2 specification and its most important aspects;
- The state of the art on PCB design, communication interfaces, and the current off-the-shelf solutions on Single-Board Computers;
- An assessment of the most important requirements regarding the design of maritime systems at LSTS, by means of a questionnaire to the team;
- A modular, scalable, and CPU-independent hardware architecture built around these requirements;
- An implementation of this architecture's Main Board, capable of wielding nine expansion modules;
- An implementation of this architecture's expansion modules, specifically three custom-developed modules: GPS, Iridium, and RS232;
- The verification and validation of both systems and their main advantages compared to traditional hardware designs in maritime vehicles.

1.6 Outline of the document

This document is organized as follows:

Chapter 2 describes the required background for the understanding of this work. In particular, the currently available vehicles at LSTS are introduced, and the M.2 specification is briefly reviewed.

State of the art in PCB design and challenges in high-speed lines is discussed in Chapter 3, as well as the most common communication interfaces in computational systems and some of the most popular commercial-off-the-shelf solutions concerning Single-Board Computers.

Chapter 4 describes the work done towards the establishment of the technical requirements by means of a questionnaire answered by the LSTS team and further analysis.

The design of the hardware architecture and the development of the Main Board and expansion modules is thoroughly detailed in Chapter 5.

Chapter 6 discusses the verification and validation process of the developed hardware systems, as well as their capabilities and limitations.

Finally, the document ends with the conclusions that have been reached and future research directions.

Chapter 2

Background

This chapter introduces several autonomous underwater vehicles developed at LSTS before describing some operational deployments. It also mentions the M.2 specification, which is strongly followed throughout the project’s development.

2.1 Autonomous Underwater Vehicles

The main application of this work will be in the development of Autonomous Underwater Vehicle (AUV). An AUV is a robot that travels underwater autonomously, without requiring input from an operator. AUVs are a class of a larger group of undersea systems, the Unmanned Underwater Vehicles (UUVs).

Nowadays, because their practical uses span a large range, AUVs are a popular solution for multiple applications, such as oceanographic surveys, bathymetric measurements, oil spillage monitoring, and even military surveillance and defense.

For the purpose of the work, we restrict our attention to a subset of commercial underwater vehicles: the LAUVs, offered by LSTS; and the Iver3 family, by L3Harris.

2.1.1 Light Autonomous Underwater Vehicle

LSTS specializes in the development of Light Autonomous Underwater Vehicles (LAUVs) [1], a subset of AUVs that is “an affordable, highly operational and effective surveying tool. It is a lightweight, one-man-portable, and operated vehicle that can be easily launched, operated, and recovered with a minimal operational setup. Thus, the operation of the LAUV does not require extensive operator training. The minimalist functional system includes communications, a computational system, and basic navigation sensors, and the LAUV capabilities are built up by adding optional payload modules”.

LSTS offers two main variants of the system: the Mapping version and the Upper water column version. They share some similarities in the construction and feature set but have very different use cases. Figure 2.1 shows both models.



Figure 2.1: LAUVs developed at LSTS [1]
Mapping (orange) and Upper water column (yellow)

All the vehicles are supplied from a pack of 18650 Lithium-ion batteries. The basic block is a 7 series/3 parallel set of batteries (7S3P), creating a nominal voltage of 25.9 V. In order to increase the battery life of the vehicle, multiple battery packs are arranged in parallel.

Every vehicle has a basic set of features, independently of its applications:

- **Wi-Fi modem** - On the surface, the vehicles can communicate with the ground station to quickly and easily exchange information.
- **GSM modem** - For long-distance communications, on the surface, the vehicles use the cellular network to not only communicate with the Internet but also to send SMS.
- **Acoustic Modem** - Used for underwater communications. Sourced from WHOI or Evologics.
- **BMS** - The battery management system, *Batman*, responsible for protecting and managing the vehicle's battery;
- **PCTL** - The energy management and distribution system, featuring a mini-atx power supply unit, responsible for converting the main supply (25.9 V) to several other voltages (12 V, 5V, and 3.3 V).
- **Primary CPU** - A PM-LX2 AMD Geode microprocessor responsible for the vehicle's navigation software.
- **Secondary CPU** - A Raspberry Pi, responsible for lower power applications, to increase the vehicle's efficiency and endurance.

Mapping

The Mapping systems are high-end LAUVs, specialized in security and surveillance, search and rescue, habitat mapping, and bathymetric surveys.

Beyond the base specs, the vehicle has the following payload:

- **Sidescan Sonar** - Responsible for efficiently 2D mapping large areas of the seafloor. Sourced from Edgetech, Klein, or Imagenex;
- **Multibeam Sonar** - In contrast to the sidescan sonar, this sonar fires multiple sound beams. It is responsible for the 3D mapping of the sea bed;
- **Forward Looking Sonar** - Single beamed sonar, responsible for detecting objects in front of the vehicle;
- **HD Camera** - Responsible for capturing images from below;
- **DVL** - *Doppler Velocity Logger*, used to measure the vehicles velocity relative to the sea floor;
- **IMU** - *Inertial Measurement Unit*, used to measure accelerations, orientations or angular rates;
- **INS** - *Inertial Navigation System*, used for estimating the vehicle's position;
- **LEDs** - Used for illumination.

Upper water column

Unlike the Mapping variant, the Upper water column version does not feature any sonar. It focuses on the collection of hydrographic data, used mainly for water quality measurements and biological metrics. This version is further subdivided into two classes: the base class; and the SOI class, with increased endurance.

Beyond the base specifications, the vehicle has the following payload:

- **CTD sensor** - Conductivity, Temperature, Depth sensors, the primary tool for determining essential physical properties of seawater;
- **Turbidity sensor** - Responsible for measuring the amount of light that is scattered by the suspended solids in the water;
- **Chlorophyll** - Responsible for estimating the concentration of phytoplankton.
- **IMU** - *Inertial Measurement Unit*, used to measure accelerations, orientations, or angular rates. It is not as powerful as the Mapping, but has a lower cost.

2.1.2 Iver

Iver is a family of low-cost Autonomous Underwater Vehicles, developed by L3Harris. They are ideal for coastal applications such as sensor development, general survey work, sub-surface security, research, and environmental monitoring [6]. It directly competes with the Mapping LAUV, sharing some features and applications. Figure 2.2 shows the vehicle.



Figure 2.2: The Iver vehicles [6]

2.2 Operational deployments

The LSTS has successfully fielded unmanned air, ground, surface, and underwater vehicles in innovative operations in Europe, Asia, and in the United States of America. These include some world firsts.

The LSTS has been organizing, in cooperation with the Portuguese Navy, the Rapid Environmental Picture (REP) annual exercise since 2010. REP(MUS) is now a large-scale experimentation exercise conducted on an annual basis in which Operational Communities work together with Academia and Industry to develop and test operational concepts, requirements, interoperability, technological advances, C3, tactics, and procedures in Maritime Unmanned Systems (MUS). The REP(MUS) 21 organization was supported by the Portuguese Navy, Porto University, NATO STO Center for Maritime Research and Experimentation (CMRE), and the NATO MUS Initiative. The exercise concept encompasses the experimentation of MUS in operations in a serial schedule framework. The exercise is conducted in several maritime areas in Sesimbra and Tróia.

The primary objectives of the 2021 edition of the exercise were to: (1) demonstrate NATO's and Nations' technological and operational capability concerning MUS; (2) promote the development and evaluation of operational concepts (e.g. concept for elements of a potential ASW barrier); (3) evaluate the performance of multi-static systems in ASW; (4) perform interoperability tests Unmanned-Unmanned and Unmanned-Manned; (5) demonstrate Command, Control and Communications (C3) capabilities in the operation of underwater, surface and aerial maritime systems through interoperable communication networks (underwater, radio and satellite); (6) demonstrate MUS capabilities in the contribution to an enhanced maritime situational awareness including in the underwater domain (e.g. wide area persistent underwater sensing and detection); (7) evaluate the performance of ISR sensors (sonar, underwater communications, radar, cameras, etc.) and associated communication systems and networks installed in MUS, in several multi-domain environments.

The LSTS also led the *Exploring Fronts with Multiple Robots* Schmidt Ocean Institute cruise that took place in 2018. The 3-week cruise onboard the Research Vessel Falkor successfully

demonstrated a novel approach to observing the changing ocean with networked underwater, surface, and air vehicles operation in coordination with the Falkor. The operations area was the North Pacific Subtropical Front located approximately 1000 nautical miles West of San Diego, California.

2.3 M.2 Specification

M.2 is a specification [7] designed by PCI-SIG for internal add-in cards and associated connectors. PCI-SIG is an electronics industry consortium responsible for developing and maintaining the standardized approach to peripheral component I/O data transfers [8]. This section summarizes the most important aspects to consider during the development of this project.

2.3.1 Terms and Definitions

The specification begins by declaring some terms and definitions commonly used throughout the document [7]:

- **Host** - Typically referring to the electrical interface source/master;
- **Platform** - Typically referring to the physical location. Usually, a Main Board on which the Modules/Add-in Cards are mounted (via a connector or soldered down);
- **Module** - The Add-in card or device that is either plugged into the Platform connector or soldered down onto the Platform Mother Board
- **Add-in Card** - A card or module plugged into a connector and mounted in a chassis socket.
- **x1, x2, x4** - x1 refers to one Lane of basic bandwidth; x4 refers to a collection of four Lanes; etc.

2.3.2 Mechanical Specification

The specification defines a family of M.2 modules and interconnects based on a 75-position edge card connection, comprised by the following module sizes (see Figure 2.3):

- Type 1216
- Type 1630
- Type 2226
- Type 2230
- Type 2242
- Type 2260

- Type 2280
- Type 3026
- Type 3030
- Type 3042
- Type 22110

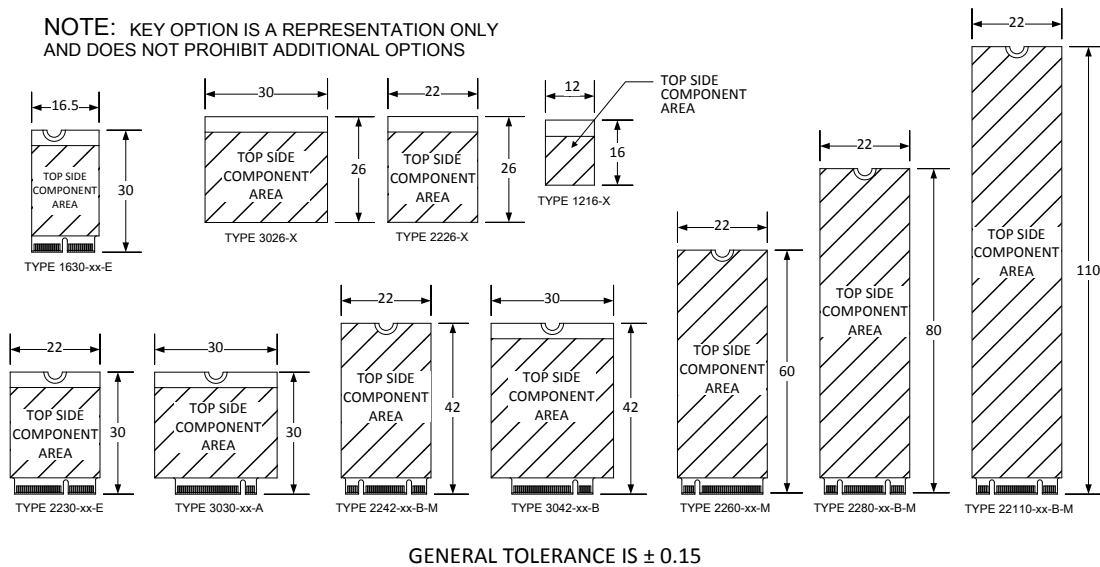


Figure 2.3: M.2 Module Sizes [7]

In addition to these physical dimensions, the specification lists 12 variants of the M.2 interconnect, called *keys*. The existing *keys*, as well as their respective provided communication interfaces, are specified in the Table 2.1.

The module naming convention is then XXYY-Z, where XX is the width, YY is the height of the card and Z is the key. A 2242-B module is 22x42 mm long with B-key. The card thickness is also fixed at $0.8\text{ mm} \pm 10\%$ for every variant.

Table 2.1: M.2 Keys [7]

Key ID	Provided interfaces
A	2x PCIe x1, USB 2.0, I ² C and DisplayPort x4
B	PCIe x2, SATA, USB 2.0 and 3.0, audio, UIM, HSIC, SSIC, I ² C and SMBus
C	Reserved for future use
D	Reserved for future use
E	2x PCIe x1, USB 2.0, I ² C, SDIO, UART, PCM and CNVi
F	Future Memory Interface (FMI)
G	Reserved for future use
H	Reserved for future use
J	Reserved for future use
K	Reserved for future use
L	Reserved for future use
M	PCIe x4, SATA and SMBus

2.3.3 Electrical Specification

In addition to detailing the pin-out of every available M.2 key, the specification describes the available communication interfaces and concerns regarding the power management of the modules. This section covers the most essential electrical aspects.

The specification states two different voltages used for the signals in the interface: 3.3 V and 1.8 V. Depending on the Host system's configuration, it might be necessary to translate some data signals to comply with the specification.

Power

The M.2 socket utilizes a single 3.3 V power source. The specification expects that this voltage source is available during the system's standby and suspend state to support wake event processing. This means that the modules' power source can't be simply cut-off. Instead, the Host is responsible for putting the module into a suspended state.

The number of 3.3 V pins is determined by the maximum required current of a solution associated with each type of socket. The specification defines the current capability per pin as 500 mA.

PCI Express Interface

Some M.2 sockets feature one or more PCIe Lanes. A Lane consists of an input and output high-speed differential pair and a differential reference clock.

By default, the transmitter differential pair of the connector must be connected to the transmitter differential pair on the system board and the receiver differential pair on the add-in card. The

receiver differential pair is analogous. However, the positive and negative signals of the differential pair may be reversed to simplify PCB trace routing as all PCIe receivers feature an automatic Lane polarity inversion.

USB Interface

The Universal Serial Bus (USB) interface supports USB 2.0 (Low Speed, Full Speed, and High Speed). The USB functions implemented in an add-in card are expected to report as self-powered devices from the 3.3 V power source.

UART Interface

The Universal Asynchronous Receiver-Transmitter (UART) interface is also available for some types of sockets. It can handle 8-bit data frames with one start and stop bit.

The specification details the following 4-wire interface:

- **UART_RXD (Input)** - Receive data line
- **UART_TXD (Output)** - Transmit data line
- **UART_RTS (Output)** - Request to Send signal
- **UART_CTS (Input)** - Clear to Send signal

In addition to this configuration, it also specifies an additional, non-standard, UART interface (5-wire):

- **UART_WAKE# (Output)** - Wake the Host signal

By default, the transmitter signal of the connector must be connected to the transmitter pin on the system board and the receiver pin on the add-in card. The receiver signal is analogous.

Chapter 3

State of the art

This chapter discusses state of the art in the fields of interest to the dissertation. It starts with some technical aspects concerning the design of circuit boards and its associated challenges, and best practices in routing high-speed signals. The most common communication interfaces are then described and compared, as well as some of the most popular commercial-off-the-shelf solutions in the Single-Board Computers market.

3.1 Challenges in the design of PCBs

To aid the design and development of PCBs it is necessary to be familiar with some of the intricacies of the process.

Dedicating time and resources to the research of the matter is vital in order to prevent common errors from the beginning, and guarantee the reliability of the circuit board. According to [9], to make the design of a PCB reliable, factors such as strain inductance, simplicity, modularity, and compactness are required.

On top of that, high-power or high-speed lines must be approached early. Since the project to be developed is expected to have low-power environments, some of the considerations taken by [9] on high-power design might seem unnecessary, but can still be considered good practices in the design process. However, the higher the switching frequencies are, the more radiation occurs on a PCB [10]. Because the framework uses a lot of low-level, high-speed protocols, such as SPI, SDIO, USB, Ethernet, or even PCIe (which can achieve transmission frequencies of a few GHz), the issues arriving from these frequencies must be taken into consideration.

In [10], the authors list some of the properties and potential problems of a high-speed transmission line:

- The over and undershoots caused by the changes in the impedance of the signal chain, which require a proper termination of the line;
- The mutual influence of two nearby high-speed signals, *crosstalk*, caused by electromagnetic interference;

- The caution in routing differential signals, where the two signals must have the same length in order to guarantee the same propagation delay;
- The return path of the high-speed signals, which must be directly underneath the signal trace in order to reduce EMI.

There is a need to mitigate these issues without requiring access to expensive and complicated simulation software. As such, over the years, some best practices have been generally agreed to achieve great results.

To minimize the effect of *crosstalk*, the authors in [10, 11, 12] give the following recommendations:

- Widen spacing between signal lines as much as possible and try not to bring traces closer than three times the dielectric height.
- The distance between the centers of two adjacent traces should be lesser than four times the trace width.
- Design the transmission lines so that the conductor is as close to a ground plane as possible. This will help decouple the traces from adjacent signals.
- Two signals on adjacent layers must be routed orthogonally to each other.

The authors in [11] also address differential lines and mention that the distance between differential traces must be at least two times the gap of the pair.

Regarding signal integrity, there is also a set of best practices specified in [11, 10]:

- Traces should be as straight as possible. Right-angle bends must be avoided. When needed, use 45° bends.
- Avoid vias in high-speed lines as vias introduce impedance changes and reflections. If needed, use vias in both traces to match the delays introduced [10].
- Place a ground plane directly beneath the signal. If the signal is in an inner layer, sandwich the layer between reference planes.

In [10], the authors add that the designer has to make sure that the return current can flow ideally underneath the signal trace. A good way to achieve this is to add some ground vias around the signal via.

Differential high-speed lines, according to [13], must minimize pair intra-pair skew by matching the length of each of the traces. In addition, the total trace length must be kept at a minimum to minimize reflections in the traces.

As per [11, 10], another consideration is power distribution:

- Ground planes must not be split. Split ground planes act as slot antennas and radiate [10].

- Add the recommended decoupling capacitors for as many VCC/GND pairs of an IC as possible. Also keep these capacitors as close to the pins as possible.
- Use wide, short traces between the via and capacitor pads.
- Connect each ground pin or via to the ground plane individually. The same applies for power pins if a power plane is used.

3.2 Communication interfaces

There are multiple ways of communicating with a traditional CPU, each one with advantages and disadvantages. Table 3.1 compares some of the most popular interfaces.

Table 3.1: Comparison of the throughput of popular communication protocols

Protocol	Maximum Throughput
Serial port (RS232)	1 Mbps
USB 2.0	480 Mbps
USB 3.0	4.8 Gbps
PCIe 2.0 x1	4 Gbps
PCIe 4.0 x1	16 Gbps

The serial port is a good option for low-speed, low-bandwidth, simple communications. The most common connector for this communication protocol is the DB9 connector (Figure 3.1a). Introduced in 1962 by the EIA's Radio Sector, it is one of the oldest communication interfaces that is still commonly used today.

For higher throughputs, the USB port is an alternative. Although it uses a much more complicated protocol, the USB protocol, it can reach speeds of 480 Mbps in the 2.0 specification [14] (released in 2000) and 4.8 Gbps in the 3.0 specification [15] (released in 2008). The USB specification is still receiving updates, with the latest revision in 2019 - USB 4.0, capable of 40 Gbit throughput. The USB protocol is, however, specialized for plug-and-play devices and is not the typical choice for internal expansion cards. Figure 3.1b shows a typical USB-A connector.

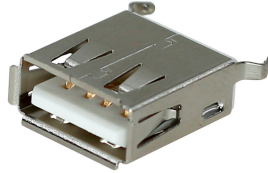
Nowadays, the standard choice for high-speed internal communications is usually the PCIe interface. This interface is composed of multiple lanes, ranging from x1 to x16. It has multiple specifications, each with improvements and increasing throughput [16, 17]. Naturally, the higher the number of lanes, the higher the throughput is. Although introduced in 2002, the latest commercial specification is the PCIe 4.0, announced in 2017, reaching speeds of 256 Gbps on a x16 lane connection. The x16 slot is shown in Figure 3.1c. The most common expansion cards using the PCIe interface are:

- Graphics cards
- High-speed solid-state drives

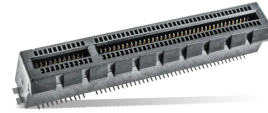
- Wi-Fi, Bluetooth, and Cellular modems



(a) DB9 Connector [18]



(b) USB-A connector [19]



(c) PCIe x16 connector [20]

Figure 3.1: Connectors

The goal of this project is to create a building block capable of taking advantage of these communication interfaces and provide a way to scale hardware applications. As such, it is crucial to select a connector that can provide some or all of these interfaces, while maintaining a low cost and a small form factor.

The PCIe Mini Card (see Figure 3.2) is a form-factor designed for PCIe devices, developed by PCI-SIG, in 2005. The host device supports both PCIe and USB connectivity, as well as a UIM interface. PCI Express Mini Cards are typically 30x50.95 mm 52-pin edge connectors, consisting of two staggered rows on a 0.8 mm pitch [21].

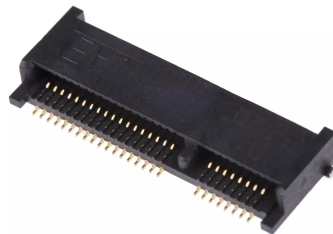


Figure 3.2: The PCIe Mini Card connector [22]

M.2 is an extremely versatile form-factor for expansion cards (Figure 3.3). Introduced in 2012, it has become very popular in the past years due to its small form-factor and high performance. M.2 cards are typically 22 mm wide 75-pin edge connectors, consisting of two staggered rows on a 0.5 mm pitch.

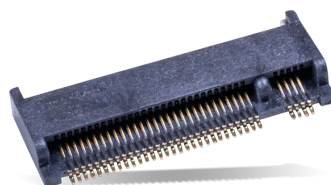


Figure 3.3: The M.2 connector [23]

The specification lists multiple versions, each providing different connectivity to the host device, as previously mentioned in Section 2.3.

3.3 Commercial off-the-shelf solutions

In this section, some commercial Single-Board Computers are presented. In addition to that, a couple of products can be found on the market that, although they have been designed with different goals, have characteristics that are intended to be replicated in the new product. In the next subsections, some of these products are discussed, and their useful characteristics are highlighted.

3.3.1 Rhodius PC/104

Rhodius is a fan-less PC/104 SBC (Figure 3.4), developed by Diamond Systems, which features the AMD Geode LX800 low power consumption processor running at 500MHz. It uses high speed DDR SDRAM providing maximum capacity at 1GB. Rhodius offers fully integrated PC functionality with low power consumption, which allows Rhodius to operate in harsh environments where the airflow for heat dissipation is restricted, and heat sinks or fans are unacceptable [24].

This product's key features include a 10/100 Ethernet, USB 2.0, parallel port for peripheral connection, 1 ATA-33 channel, Compact Flash Type-II socket, PC/104 expansion bus, LCD & CRT interface, floppy, and watchdog timer.

This SoC was introduced by AMD in 2005, while the SBC was announced in 2008. Although it clearly shows its age, it is still an efficient solution used in all LSTS vehicles. It has been battle-tested and validated but is in need of an upgrade.

This SBC is often paired in vehicles with an additional 4-channel RS232 expansion board - the FSCC-104, by Fastcom, also in the PC/104 form factor.

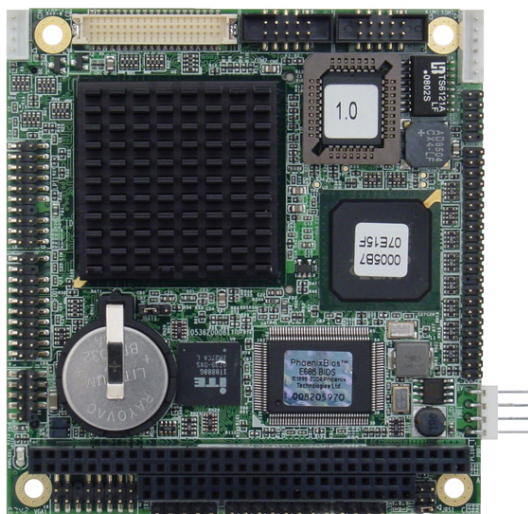


Figure 3.4: RHODEUS PC/104 SBC [24]

3.3.2 Raspberry Pi 4

The Raspberry Pi 4 (Figure 3.5) is the latest SBC offered by Raspberry Pi. It offers increases in processor speed, multimedia performance, memory, and connectivity compared to the prior generation, while retaining backwards compatibility and similar power consumption [25].

Introduced in 2019, this product's key features include a high-performance 64-bit quad-core processor, dual-display support at resolutions up to 4K via a pair of micro-HDMI ports, hardware video decode at up to 4Kp60, up to 4GB of RAM, dual-band 2.4/5.0 GHz wireless LAN, Bluetooth 5.0, Gigabit Ethernet, USB 3.0 [25].

The dual-band wireless LAN and Bluetooth have modular compliance certification, allowing the board to be designed into end products with significantly reduced compliance testing, improving both cost and time to market [25].

These features make it a perfect candidate for general-purpose desktop applications, offering good performance and high power efficiency.

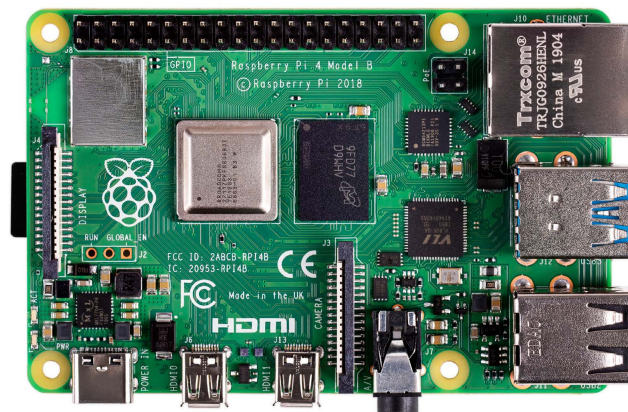


Figure 3.5: Raspberry Pi 4 [25]

3.3.3 NVIDIA Jetson Nano Developer Kit

The Jetson Nano Developer Kit (Figure 3.6) is a small, powerful computer developed by NVIDIA that lets you run multiple neural networks in parallel for applications like image classification, object detection, segmentation, and speech processing, in an easy-to-use platform that runs in as little as 5 watts [26].

Announced in 2019, this product's key features include a high-performance 64-bit quad-core processor, a 128-core Maxwell GPU, hardware video decode at up to 4Kp60, hardware video encode at up to 4Kp30, 4GB of RAM, Gigabit Ethernet, USB 3.0 [26] and a M.2 E-Key interface.

Jetson is used to deploy a wide range of popular Deep Neural Network (DNN) models and Machine Learning (ML) frameworks to the edge with high performance inferencing, for tasks like real-time classification and object detection, pose estimation, semantic segmentation, and natural language processing (NLP) [5].

These features make it a perfect candidate for GPU intensive applications, that take advantage of the chip's Artificial Intelligence (AI) capabilities, while maintaining a high power efficiency.



Figure 3.6: NVIDIA Jetson Nano Developer Kit [26]

3.3.4 Compute Module 4 IO Board

The Compute Module 4 IO Board (Figure 3.7) is an official open-source companion board for Raspberry Pi Compute Module 4. It is designed for use both as a development system for Compute Module 4 and as an embedded board integrated into end products [27].

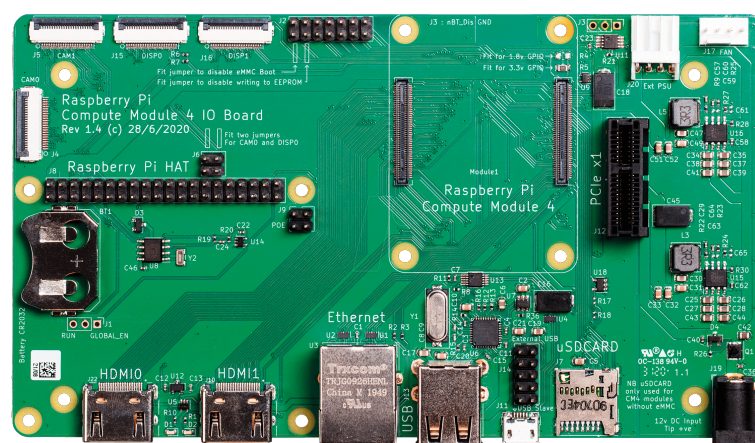


Figure 3.7: Compute Module 4 IO Board [27]

The IO board is designed to allow you to create systems quickly using off-the-shelf parts such as HATs and PCIe cards, which might include NVMe, SATA, networking, or USB. The board also features USB 2.0, Gigabit Ethernet and HDMI ports, which are located along one side to make enclosures simple [27].

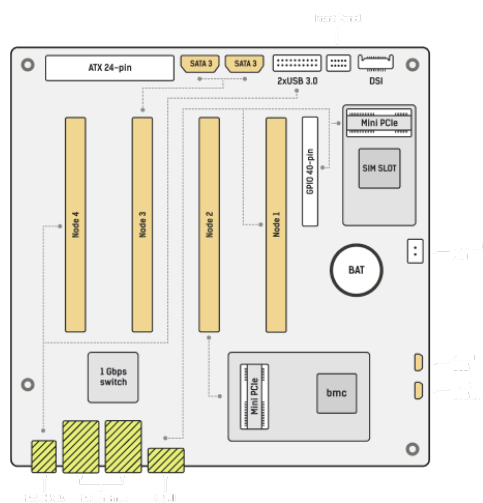
The Compute Module 4 IO Board also provides an excellent way to prototype systems using a Compute Module 4.

3.3.5 Turing Pi 2

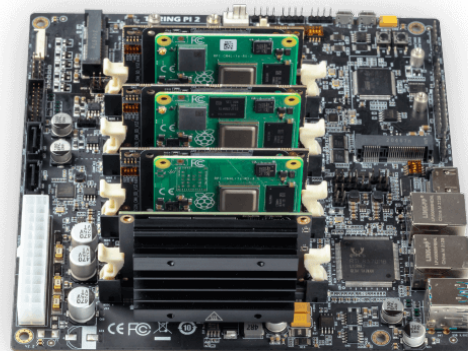
The Turing Pi 2 (Figure 3.8) is a compact ARM cluster developed by Turing Machines Inc, that unites four compute modules on a Mini ITX motherboard [28].

Introduced in 2022, this product's key features include 4 compute module nodes, a 7 port managed Gigabit switch, 2x Mini PCIe Gen2 slots, 2x SATA III 6 Gbps ports, 4x USB 3.0 connectors. The compute modules support NVIDIA Jetson and Raspberry Pi CM4.

It shares some similar features to the project to be developed, but their objective is different. Instead of providing a way to scale the peripherals of a system, they focus on the scalability of the processing power, by connecting multiple processors forming a cluster. It is a good example of a processor-independent architecture.



(a) Block diagram of the system [28]



(b) Fully populated circuit board [28]

Figure 3.8: Turing Pi 2

3.3.6 Alftel x12

The Alftel x12 PCIe-M.2 Carrier Board (Figure 3.9) is a standard full-height/half-length PCIe add-on card purposely designed to host up to x12 standard PCIe M.2 radio cards (Key A/E), developed by Alftel Systems [29]. It was introduced in 2021.

Low power CPUs usually provide only one PCIe port, and this product is a good example of the expansion of the PCIe ports, using a PCIe switch integrated circuit.

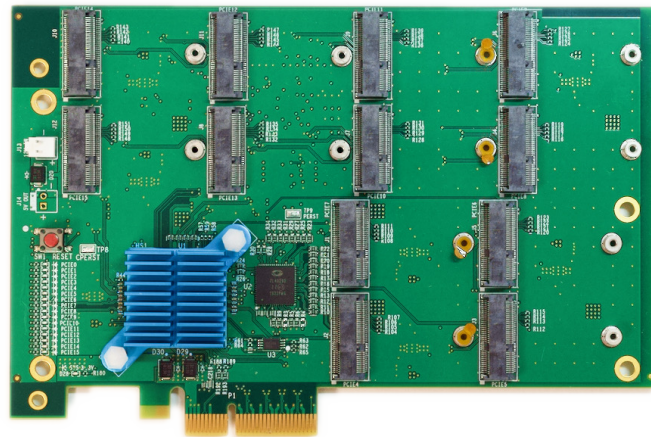


Figure 3.9: Alftel x12 Carrier Board [29]

Chapter 4

Problem specification

In this chapter, we discuss the work done towards the establishment of the technical requirements. This was accomplished through constructing and analyzing a questionnaire addressed to the LSTS team, which is also detailed in this chapter.

4.1 Questionnaire

In order to understand the primary requirements and focus of development, a questionnaire was proposed to the LSTS hardware and firmware teams. It is divided into 4 sections:

- The first section focuses on the most important processors for the target application.
- Section two presents the hardware expansion system and its most important features.
- The third section explains the software integration of the system with the current LSTS ecosystem.
- Section 4 shows some additional minor experimental features and assesses their importance.

Some solutions focused on AI were also polled, such as the NVIDIA Jetson and the TPU Accelerator, to understand if they would be valuable for the application in maritime systems.

The questionnaire is included in Appendix A but also publicly available at:

<https://forms.gle/tw7e8jWWnYP5GVQW8>

4.2 Analysis

The questionnaire was answered by two team members: one specialized in firmware and software development; the other one in hardware development.

Starting with the first section of the questionnaire, they were in sync regarding the most important SoC to explore, with the Raspberry Pi Compute Module 4 coming up unanimously. The team did not consider any other SoC to be as important, so the focus of this work revolves around

the CM4. When it comes to the number of processing units available in the system, one of the members found it interesting to feature two processors, while the other did not. After some discussion, we decided to assign this feature the lowest priority because while it could prove to be useful, it does not justify the increase in complexity of the design, at least for now.

The second section of the questionnaire was related to the types of expansion modules that would be valuable for the application in maritime systems. The team pointed out, unanimously, that the UART expansion module would be the most important, as well as modems for Iridium, Wi-Fi, and GPS. The member specialized in firmware and software development also considered the possibility of creating a microcontroller custom module to be useful, although with no particular application yet. Both members also considered the SATA expansion module to be interesting to approach, albeit with a low priority. While there are commercially available SATA expansion modules, the availability is relatively low, and we could not source any of them. Regarding features regarding the interaction of the processing unit with the modules, the team considered the ability to turn on and off the supply of each module the most valuable feature. Finally, both members think the number of PCIe ports should be scalable.

In the third section, the team considered that a lower-level integration of the modules with their software tool-chain was not really a priority; only the ability to turn on and off the modules was valuable.

Finally, the fourth section was received as an extremely experimental suggestion. The team considered it to be quite exciting but also dangerous to approach. The insufficient documentation of the Kicad API and the small currently available parametric generation software regarding PCBs were factors that led to them assigning the feature the lowest priority.

They were very helpful in the definition of the problem, not only by answering the previously mentioned questionnaire but also by participating in some discussions and providing suggestions and recommendations. Regarding the questions on AI-focused hardware, the team found it to be interesting but not really useful, at least for now. As such, these solutions were not used.

After retrieving their feedback, an analysis and classification of the proposed features were made, and organized into a set of requirements for the Main Board and the expansion modules.

4.3 Requirements

As a result of the previously mentioned study, Table 4.1 and Table 4.2 present the outlined plan for the Main Board and expansion modules, respectively.

Table 4.1: Requirement Analysis of Main Board

Requirement	Priority
The hardware solution must be independent of the CPU	High
The PCB must not exceed 4 layers	High
The PCB must have some peripherals integrated (1xUSB, 1xRJ45, 1xRS232)	High
The processor functionalities can be increased via expansion modules	High
Some expansion modules must feature a PCIe interface	High
All expansion modules must feature a USB and UART interface	High
At least one expansion module must feature a UIM interface	High
The number of expansion modules must be flexible	Medium
The circuit board dimensions must be flexible	Medium
Each module must have a separate power management system	Medium
The power management system must be integrated with DUNE	Medium
Modules can be hot-swapped	Medium
A secondary processor can be attached	Low

Table 4.2: Requirement Analysis of Expansion Modules

Requirement	Priority
Must be compatible with the u-blox NEO family of modems	High
The PCB must not exceed 4 layers	High
The antenna connector must be uFL	Medium
The GPS modem synchronized time pin must be accessible, through RS232	Medium

(a) GPS

Requirement	Priority
Must be compatible with the Iridium 9603 modem	High
The PCB must not exceed 4 layers	High
The Iridium modem ring indication pin must be accessible	Low

(b) Iridium

Requirement	Priority
Must provide 4 serial ports	High
The PCB must not exceed 4 layers	High
The logic level must be RS232	High
The serial channels must have visual indicators of data, with LEDs	Low

(c) Serial

Chapter 5

Approach

This chapter begins by discussing the proposed hardware architecture, detailing its functionality and constraints. Following this architecture, the design process of the Main Board and some custom expansion modules is presented in detail.

5.1 Design Principles

Compatibility with existing products on the market is not guaranteed unless standard communication interfaces and connectors are utilized, such as PCIe (with M.2 or mini PCIe connector) and USB. From Table 2.1, we can see the vast number of communication protocols provided by the M.2 interface. These factors, along with the small form-factor of the connector made it a perfect candidate for this application.

- For high-speed communications, the B and M *keys* are the most suited, because of the high number of lanes of the PCIe interfaces. They are typically used for high-speed Solid-State Drives (SSDs);
- For cellular modems, the only option is the B *key*, because it is the only one providing a User Identity Module (UIM) interface;
- For serial port communications, the E *key* is the only option.

As such, the options selected for the implementation of this project were keys *B* and *E*. The *E* key is the most versatile and will be used as the primary connector for this project's expansion cards. Keep in mind that most of the provided interfaces of these connectors are not necessary for the main application of this project, and therefore will be ignored.

When using standard connectors and interfaces, it is mandatory to understand and closely follow the specifications. As such, the M.2 specification in [7] will be referenced in further sections.

5.2 Architecture

In engineering, an hardware architecture is a view of the physical architecture that represents the hardware components and their interrelationships. The proposed architecture, BALU, is a hardware solution based on a variable-sized circuit board (the Main Board) that guarantees scalability and modularity via specialized slots where expansion modules can be connected. These modules add additional features to the system, such as extending peripherals and IO or adding wireless communication modems. Figure 5.1 shows the architecture.

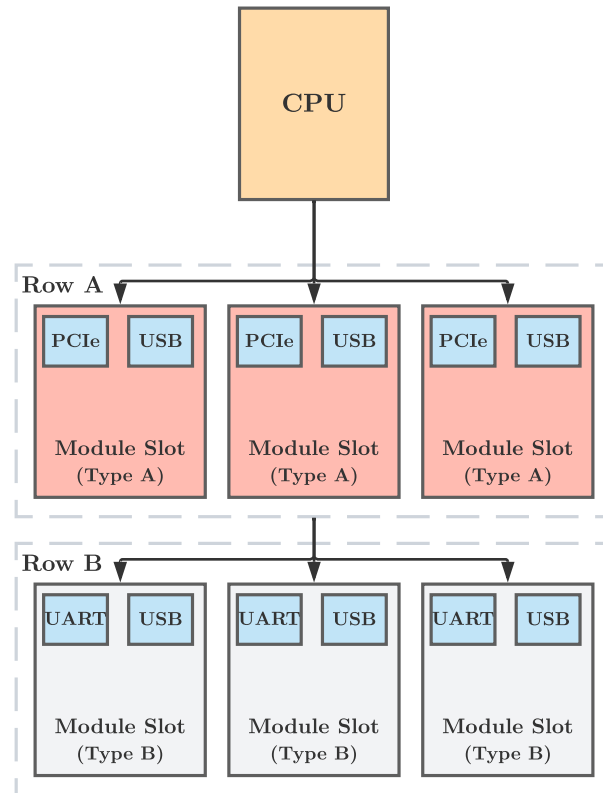


Figure 5.1: BALU architecture

The architecture features a 3-column expansion module slot matrix. However, the number of rows is not enforced - each application defines the required amount. This scalability is achievable because each row is daisy-chained (see Figure 5.2), which means that each row is wired together in sequence.

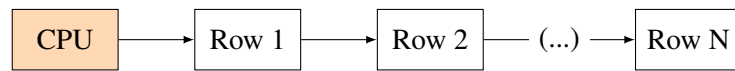


Figure 5.2: Daisy-chaining rows

The architecture specifies two types of module connectors:

- **Type A** - in red, targeted at high-speed modules using the PCIe interface, such as solid-state drives, Wi-Fi modems, or even processing accelerators used in machine-learning algorithms.
- **Type B** - in gray, targeted at general-purpose modules using the USB and UART interfaces, such as Cellular and Bluetooth modems.

While Type A slots provide the most powerful and feature-rich interfaces, they are also more expensive and challenging to implement. Furthermore, the generic modules are the most useful for the target application because most of the time, the expansion modules only require a USB or a Serial interface. The expansion rows can also be categorized by the type of connectors they feature (Row A and Row B).

5.3 Detailed design of the Main Board

In order to demonstrate the application of the architecture, a general-purpose solution was developed. It is a 3-row circuit board, which totals to 9 expansion slots. This section details the development of the board.

5.3.1 Processing Unit

The architecture does not require the usage of any specific processing unit. This means that it is possible to design a circuit board around a current commercial CPU and, in the future, tune the design to accommodate a newer released processor. However, there is still one detail that must be decided by the design: the peripherals' voltage levels. Nowadays two main voltages are used for data signals: 1.8 and 3.3 V. Some processors are bound to one of the voltages, while others provide a way to select the desired option. As the majority of the selected integrated circuits use 3.3 V signals, that seems like the most straightforward approach. However, in some signals on the M.2 connector, there needs to be a translation of voltages because the specification [7] enforces the 1.8 V logic levels. While it might be interesting to study an approach around 1.8 V in the future, the selected voltage level was 3.3 V.

As mentioned in Section 1.2, using a System on a Module instead of the bare metal processing unit greatly simplifies the integration of a CPU into the design. To evaluate and decide on the best solution, the following set of criteria were applied to the candidates:

- **Processing Capabilities** - *important*, the number of cores and their performance;
- **Typical Operating Power** - *important*, lower power consumption is optimal;

- **RAM** - *important*, higher amounts of RAM is optimal;
- **Availability** - *important*, with the current instability in the semiconductor market, it is often the deciding factor;
- **Price (per unit)** - *nice to have*, cheaper solutions are always desirable.

The application of such criteria led to the comparison results displayed in Table 5.1

Table 5.1: Comparison of the commercially available SoM

Reference	Mfr.	Processing	RAM	Power	Avail.	Price
CM4	Raspberry Pi	4x Cortex-A72	≤ 8 GB	3-7 W	Very Low	30-80 €
Jetson Nano	NVIDIA	4x Cortex-A57	4 GB	5-10 W	Very Low	100 €
Pico i.MX6	Tech Nexion	(1-4)x Cortex-A9	≤ 2 GB	1.6-5 W	Low	30-150 €

The results are inconclusive because these processors feature a lot of different configurations and can't be really compared - they have different target applications.

Take for example the offer from NVIDIA. At first glance, it features an older generation processing core compared to the Compute Module 4 (CM4), however, it comes with a significant increase both in power consumption and price. Although this could be justified by an increase in efficiency of the newer cores, the most significant culprit is the difference in the Graphics Processing Unit. While the Raspberry Pi is targeted at more general purpose applications, with a basic but capable GPU, NVIDIA's goal is towards building power-efficient Artificial Intelligence systems. This justifies the much more powerful GPU featured in the Jetson module.

Furthermore, the power consumption of the offers from Raspberry Pi and NVIDIA are prohibitive for some applications, and as such an alternative is necessary. That is what Tech Nexion brings to the table. The Pico i.MX6 features a much less powerful processing unit, but it brings a lower power consumption with it.

There isn't a perfect option, as it depends on the target application's performance requirements. Due to the extremely high documentation and community around the offers from Raspberry Pi, the selected processing unit was the Compute Module 4. Still, it is possible to change the processor on a design without major changes, as previously said.

5.3.2 IO

Although the BALU architecture only specifies the expansion modules as a means to augment the systems IO, this design includes a set of common additional ports (see Figure 5.3) from the selected processing unit (CM4), both for development and production environments.

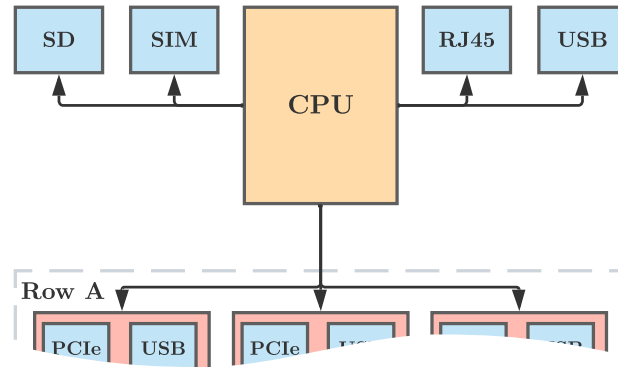
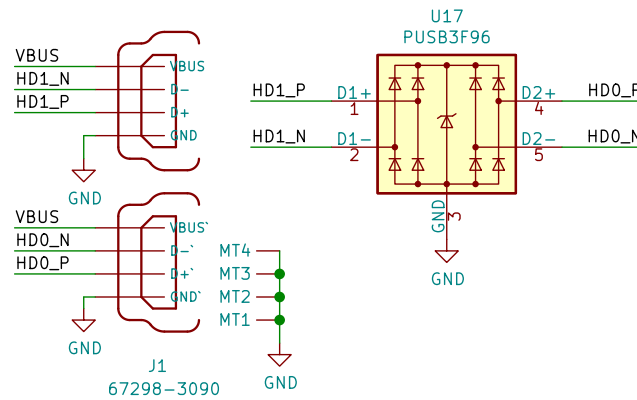


Figure 5.3: Additional IO in Main Board design

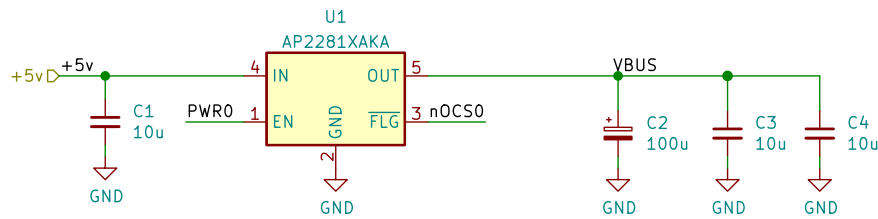
5.3.2.1 USB-A

There are two available USB-A ports, used for general-purpose USB, such as flash drives. The ports also feature over-current protection, halting the power supply for loads greater than 1.5 A.

The USB expansion circuitry will be further discussed in detail, in Section 5.3.3.1. Figure 5.5 shows the remaining circuitry of the USB section: the power switch (with over-current protection), the USB connector, and ESD protection.



(a) Stacked connector



(b) Power switch

Figure 5.5: Schematic of USB ports

5.3.2.2 Ethernet

For network connectivity, there is one Ethernet port, reaching speeds of up to 1 gigabit per second (Gbps), or 125 megabytes per second (MBps). Figure 5.6 shows the schematic of the Ethernet circuitry, featuring the RJ45 connector and ESD protection.

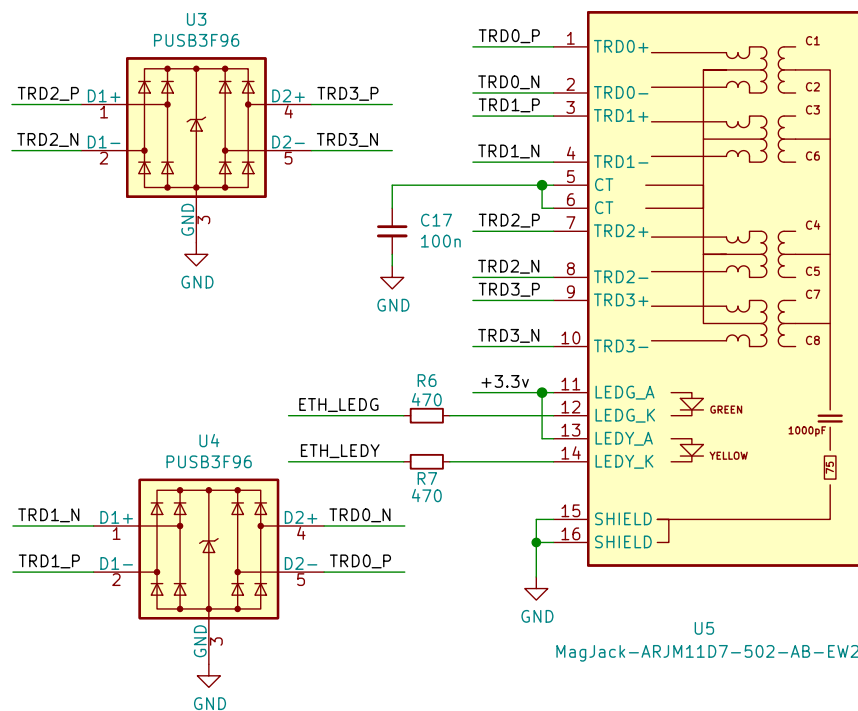


Figure 5.6: Schematic of Ethernet port

5.3.2.3 Micro SIM

This port is used for cellular communications, providing a UIM interface to one of the Type-A slots. In this design, the UIM-capable slot is in row 1 and column 2 of the expansion slot matrix. Figure 5.7 shows the schematic of the connector, as well as the ESD protection.

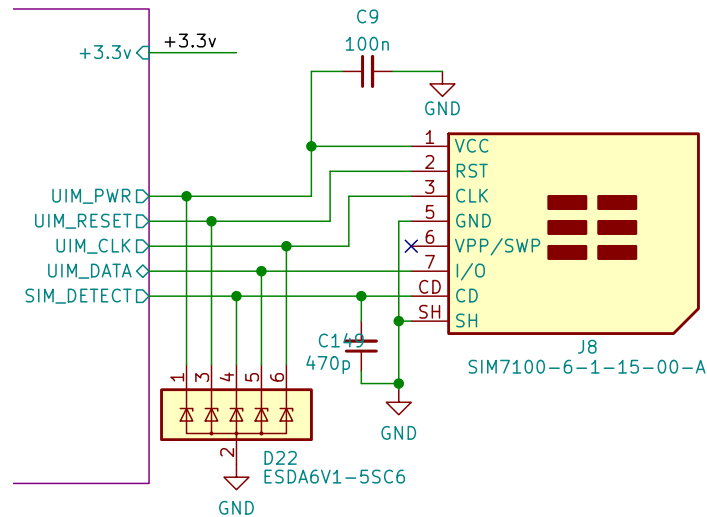


Figure 5.7: Schematic of Micro-SIM port

5.3.2.4 SD Card

There are two SD card slots in this design. One of them is responsible for booting the operating system, the other is used for auxiliary general-purpose storage. However, some Compute Module 4 have an internal eMMC memory for storage, ignoring the external boot SD card. Nevertheless, the option to provide a boot SD card is useful for some applications. Figure 5.8 shows the schematic of both of the connectors. The main difference between them is the additional power switch included in the boot slot - the processor might turn off the card's supply if it is unused.

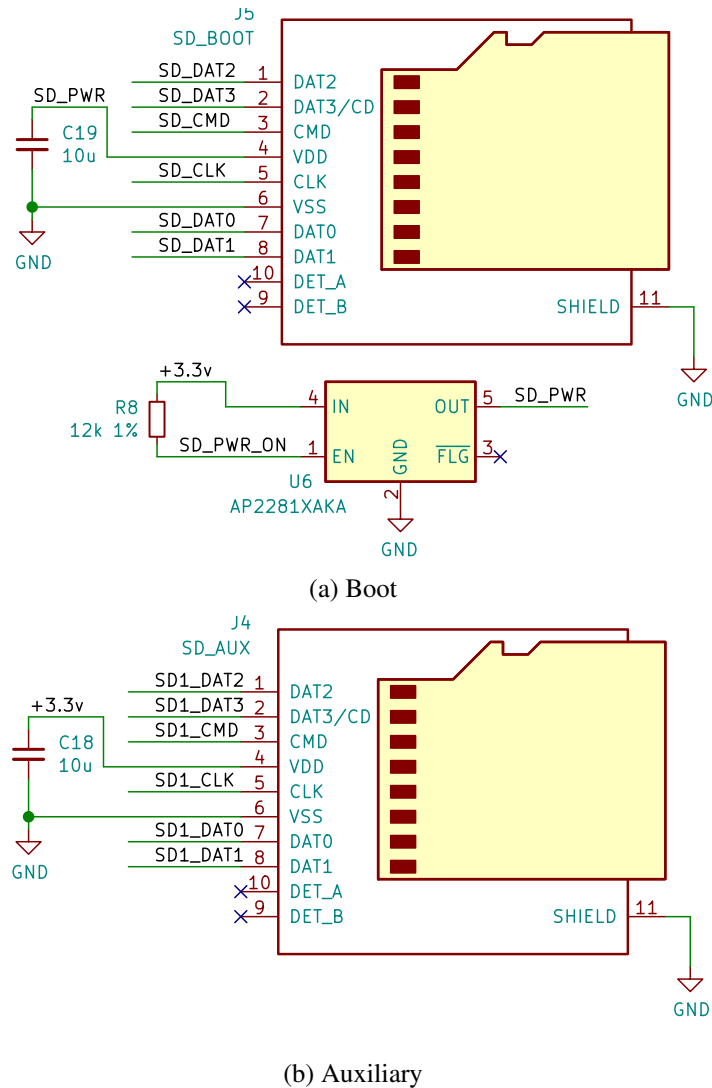


Figure 5.8: Schematic of SD Card slots

When developing, there are a couple of useful interfaces for interacting with the processing unit. The first one is a basic serial port, useful for creating a terminal to the processor and executing the Linux shell. The other, less straightforward, is a USB connection, for communicating with the Compute Module 4 internal eMMC storage. Figure 5.9 shows the schematic of the USB-C connector and its connection to the processing unit.

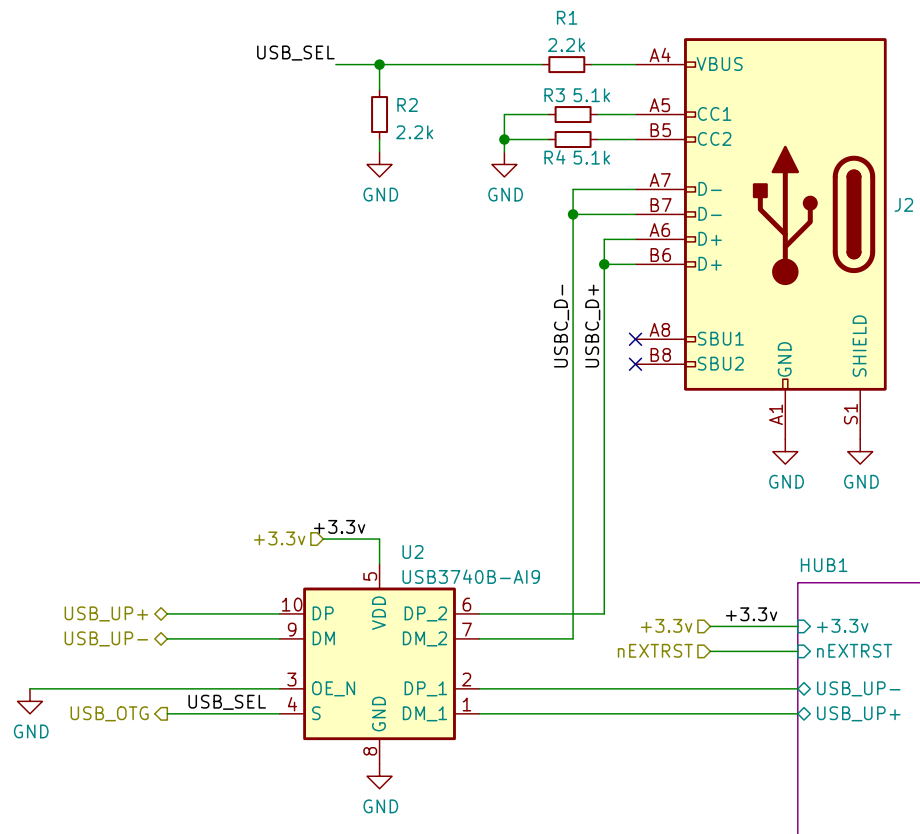


Figure 5.9: Schematic of USB-C port

In the official Raspberry Pi documentation [30], it is explained that when the processor RPI-BOOT pin is pulled low during power-up, the device enters into a programming mode. In this mode, the internal storage is accessible through the USB port, as a Device. However, the Raspberry Pi’s USB port is normally used as a Host which connects to USB devices. To set the processor’s USB controller as a Device, the OTG pin must be pulled high, by using a voltage divider into the connector’s VBUS pin. The same signal is used to multiplex the USB port of the processor between the USB-C connector and the USB-Hub used in the IO USB connectors.

Additionally, two 5.1 k Ω resistors connected to the CC pins are necessary according to the USB-C specification [31], in the detection of the cable orientation.

5.3.3 Modules

As previously mentioned, this design features 3 expansion slot rows, totaling 9 modules. As the target application does not leverage high-speed communications, the design settled on a ABB layout - 1x Row-A and 2x Row-B. This subsection will focus on the challenges and decisions made regarding the design of these rows.

5.3.3.1 USB connectivity

The architecture specifies that each module must feature a USB connection. As the processor cannot provide enough USB ports, it is necessary to expand the existing ones. This is accomplished using a USB-hub Integrated Circuit (IC).

Each row features 3 modules, which means that each hub needs to have at least 4 ports downstream - the additional port is used to daisy-chain into the next-row USB hub. Additionally, it is desirable to use a hub that is capable of driving High-Speed (HS) USB signals, with data rates of up to 480 Mbps.

To evaluate and decide on the best solution, the following set of criteria were applied to the candidates:

- **Number of ports** - *required*, must not be lesser than 4;
- **Speed** - *required*, must be able to achieve HS signals;
- **Availability** - *required*, with the current instability in the semiconductor market, it is often the deciding factor;
- **Operating Power** - *required*, lower power consumption is optimal;
- **Package** - *nice to have*, simpler packages are easier to solder and debug, leading to fewer issues;
- **Price (per unit)** - *nice to have*, cheaper solutions are always desirable.

The application of such criteria led to the comparison results displayed in Table 5.2

Table 5.2: Comparison of the commercially available USB-Hub ICs

Reference	Manufacturer	Ports	Speed	Power	Availability	Package	Price
USB2514B	Microchip	4	HS	230 mW	Very low	SQFN-36	2.8 €
CY7C65634	Cypress	4	HS	270 mW	Very low	QFN-28	3.3 €
UPD720115	Renesas	4	HS	320 mW	Very low	QFN-40	4 €
TUSB2046	Texas	4	FS	130 mW	Very low	LQFP-32	3.4 €

From the table above, it is easy to verify that the main issue in the selection of the IC is the current availability, as the demand for this type of integrated circuit is extremely high. The offers from Microchip, Cypress, and Renesas both fulfill all the required criteria, with very similar

packages. Texas is also included in the list because even though it does not check the speed requirement, it would be used as a last resort, failing to source any of the previous solutions.

In the end, the USB2514B was the only one that was acquirable, which was not a problem since it was the overall best product.

Microchip offers a lot of documentation that helps in the design of the schematic and the circuit board. From Section 3 of the IC's datasheet [32], a preliminary schematic was designed. Figure 5.10 shows the design:

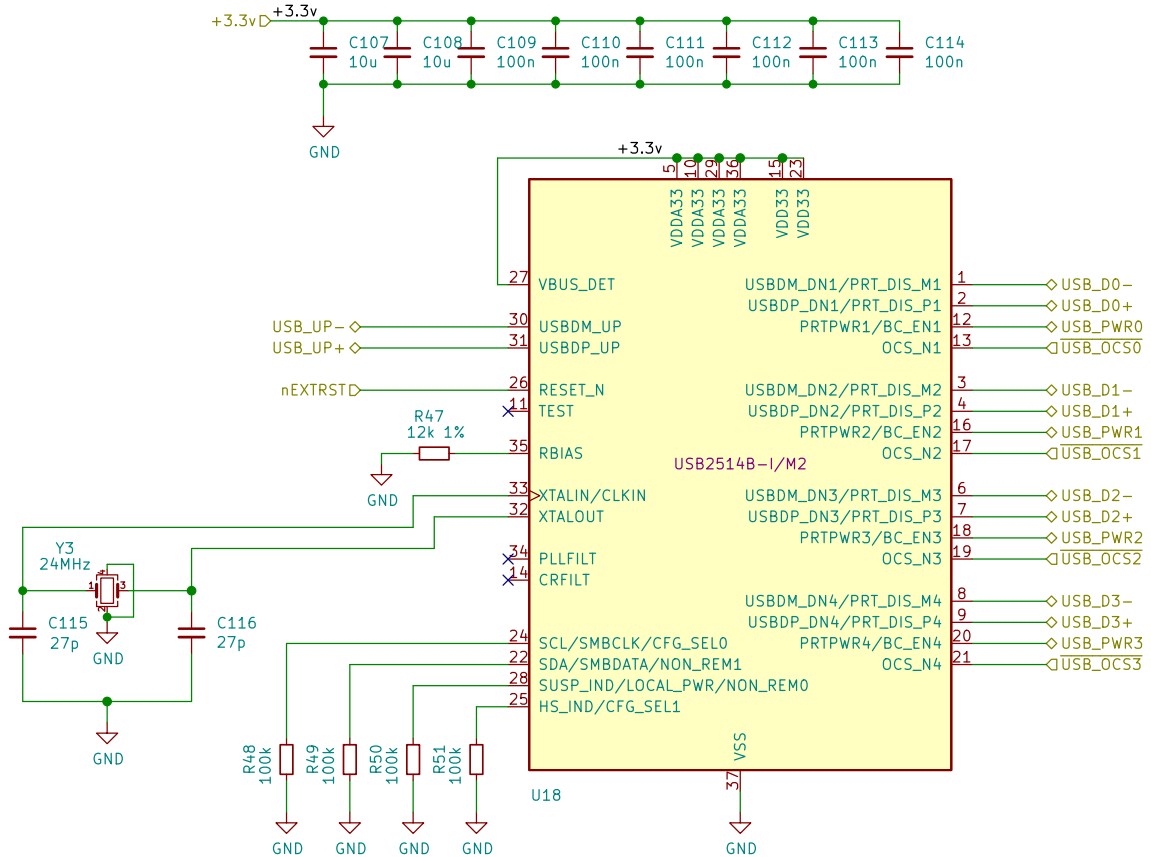


Figure 5.10: USB hub schematic (USB2514B)

This design was further validated against the provided Microchip schematic checklist, in [33]. It is also worth noting that this exact design was used for all the USB hubs in the circuit board, even the one responsible for USB-A ports described in Section 5.3.2. However, only the further uses the power control pins provided by the integrated circuit: PRTPWR#, responsible for switching the power supply to the target USB channel; OCS_N#, responsible for overcurrent detection.

5.3.3.2 UART ports

For Type-B modules, it is necessary to provide one serial connection. This means that each Row-B requires at least 3 serial ports. Even though some processors are able to provide multiple serial ports, the architecture tries to abstract away from their provided connectivity and instead focuses on providing self-expanding peripherals. This guarantees that different processors with different sets of peripherals are all compatible with the same set of rows.

It is then necessary to create a way to increase the number of UART ports provided by the processor. This is achieved using a UART multiplexer IC, that uses another type of connection (eg. SPI, USB) to create additional serial ports in the system, as demonstrated in Figure 5.11.

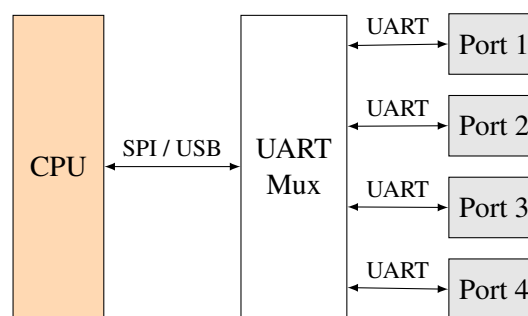


Figure 5.11: UART multiplexer block diagram (4-port)

Contrary to what happened in the USB section, this multiplexer does not have a UART upstream port, which means that it can't be daisy-chained. Therefore, it will be necessary to design another way to guarantee that the system is scalable, that depends on the selected upstream protocol.

To evaluate and decide on the best solution, the following set of criteria were applied to the candidates:

- **Number of ports (#P)** - *required*, must not be lesser than 3;
- **Speed** - *required*, must be able to achieve 1M data rates;
- **Upstream Protocol** - *required*, the selected protocol dictates the complexity of the implementation;
- **Availability** - *required*, with the current instability in the semiconductor market, it is often the deciding factor;
- **Operating Power** - *required*, lower power consumption is optimal;
- **Package** - *nice to have*, simpler packages are easier to solder and debug, leading to fewer issues;
- **Price (per unit)** - *nice to have*, cheaper solutions are always desirable.

Table 5.3: Comparison of the commercially available UART Multiplexer ICs

Reference	Mfr.	#P	Proto	Speed	Power	Avail.	Package	Price
MAX14830	Maxim	4	SPI	6M	16 mW	Low	TQFN-48	18 €
XR21B1424	MaxLinear	4	USB	12M	50 mW	Normal	LQFP-64	9 €
FT4232	FTDI	4	USB	12M	100 mW	Very Low	LQFP-64	7 €
SC16IS762	NXP	2	SPI	5M	7 mW	Normal	TSSOP-28	7.7 €

The application of such criteria led to the comparison results displayed in Table 5.3.

On the surface, the function executed by these solutions is the same. However, they achieve it very differently, and a more cautious analysis is required.

Regarding the number of ports, neither solution is perfect. The required number of ports per row is 3, and while the offers from Maxim and MaxLinear surpass that requirement, NXP falls short. Choosing this last option would require two ICs per row.

The upstream protocol might seem to be irrelevant at first, but the XR21B1424 and FT4232's USB ports mean that each row must have access to 5 ports, instead of the previously planned 4. The commercially available USB-hub ICs featuring more than 4 ports come with an exponential increase in cost and complexity of its implementation, leading these alternatives impracticable.

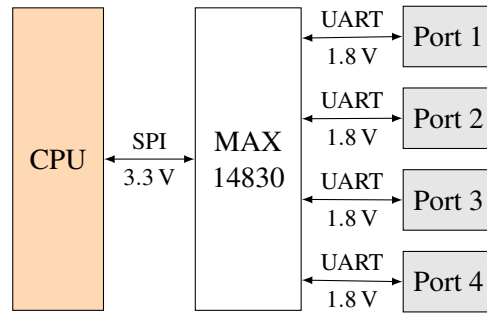
At this point, both the Maxim and NXP solutions seem viable, although a bit more work is needed on the second. Even though the NXP product is cheaper, has a more forgiving package, and has a lower power consumption, it lacks a significant feature. Although this design follows a 3.3 V level for signals, the M.2 specification ([7]) requires that the UART port's logic levels must be 1.8 V. This means that somewhere between the multiplexer IC and the output serial port there must exist a translation of the data signals' voltage. However, the Maxim IC can do this translation internally, while the NXP solution requires an additional external IC (see Figure 5.12). This increases the cost and complexity of its implementation. Therefore, the selected IC was the MAX14830.

Maxim documentation helped in the design of the schematic and the circuit board. Figure 5.13 shows the IC's functional diagram.

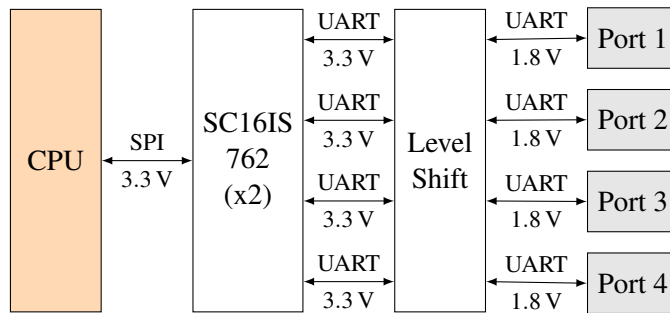
From the diagram, each channel features 8 signals: 4 lines for serial communication and flow control; 4 general purpose inputs/outputs (GPIOs). This means that in addition to fulfilling the required serial port per module, it also provides 4 extra GPIOs that can be used by the target expansion slot.

The previously mentioned internal translation of signal voltages is also present in the diagram, between the SPI interface, and the UART channels.

The IC also features an internal crystal oscillator, leading to a decrease in the number of external components which simplifies the development process. However, during testing, the crystal seemed unresponsive. After a discussion with the Maxim engineers, they recommended the usage of an external 3.6864 MHz crystal. The datasheet also mentions that the IC already features internal crystal capacitors, so it isn't required to place them externally.



(a) MAX14830



(b) SC16IS762

Figure 5.12: Comparison of UART multiplexers logic levels

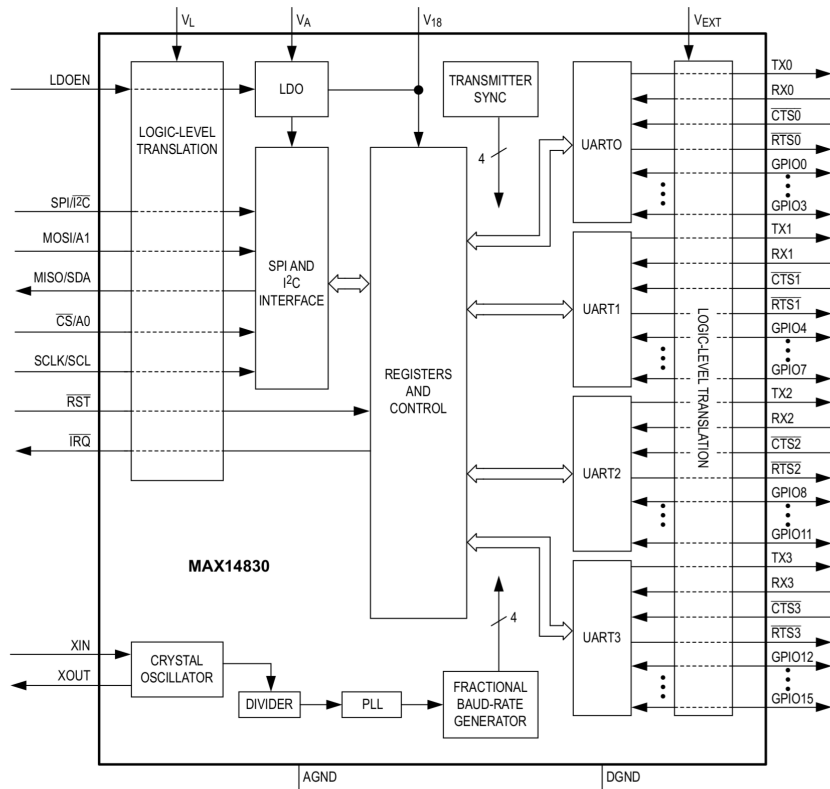


Figure 5.13: Block diagram of MAX14830 UART multiplexer [34]

Figure 5.14 shows the schematic resulting from this analysis.

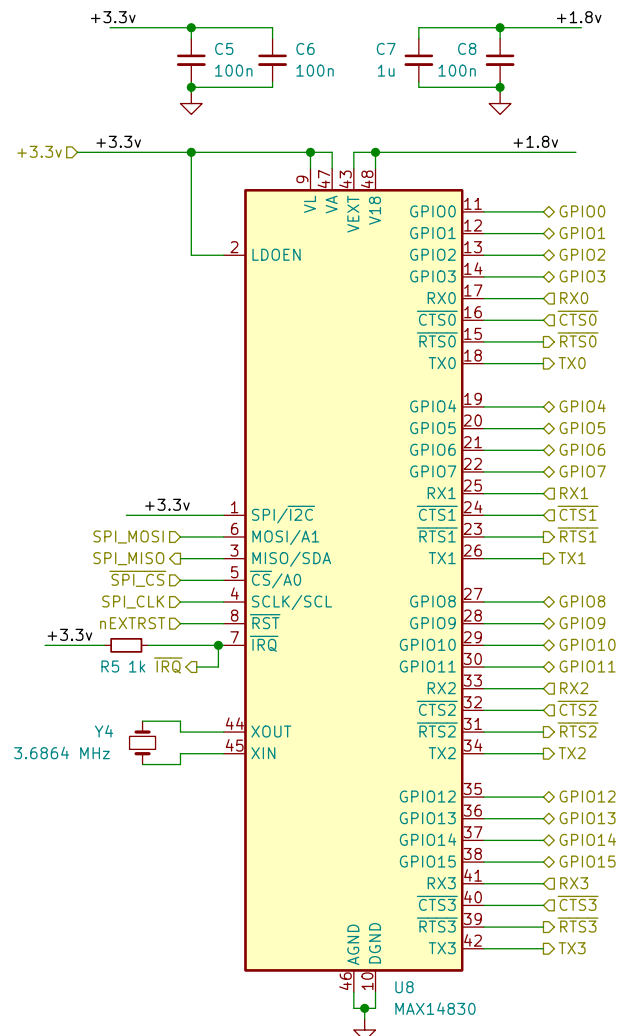


Figure 5.14: Schematic of MAX14830 UART multiplexer

5.3.3.3 PCIe connectivity

For Type-A modules, it is necessary to provide one PCIe 2.0 lane. This means that each Row-A requires at least 3 PCIe lanes. However, most of the commercially available offers in the embedded processors' space only offer up to 1 PCIe lane.

It is then necessary to create a way to increase the number of PCIe lanes provided by the processor. This is achieved using a PCIe switch IC, that uses creates multiple downstream lanes from one upstream, as demonstrated in Figure 5.15.

As each row features 3 modules, this means that each switch needs to have at least 4 ports downstream - the additional port is used to daisy-chain into the next-row PCIe switch. However, since this implementation only features one Row-A, a 3-port switch is enough.

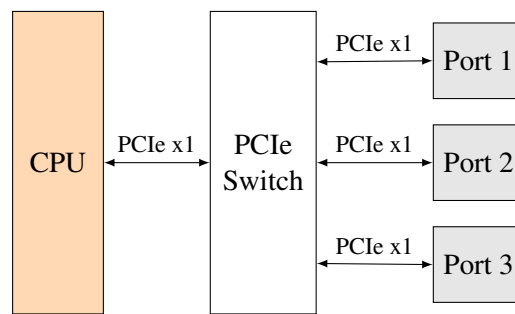


Figure 5.15: PCIe switch block diagram (3-port)

To evaluate and decide on the best solution, the following set of criteria were applied to the candidates:

- **Number of ports** - *required*, must not be lesser than 3;
- **PCIe version** - *required*, must be at least PCIe 2.0;
- **Availability** - *required*, with the current instability in the semiconductor market, it is often the deciding factor;
- **Operating Power** - *required*, lower power consumption is optimal;
- **Package** - *nice to have*, simpler packages are easier to solder and debug, leading to fewer issues;
- **Price (per unit)** - *nice to have*, cheaper solutions are always desirable.

The application of such criteria led to the comparison results displayed in Table 5.4

Table 5.4: Comparison of the commercially available PCIe switch ICs

Reference	Mfr	Ports	Power	Availability	Package	Price
PI7C9X2G404SV	Diodes	3	300 mW	Normal	LQFP-128	13.5 €
PI7C9X2G404SL	Diodes	3	750 mW	Low	LQFP-128	12.8 €
PI7C9X2G606PR	Diodes	5	600 mW	Very Low	BGA-196	20.5 €
PEX8604	Broadcom	3	1 100 mW	Very Low	BGA-196	38 €
PEX8505	Broadcom	4	800 mW	None	BGA-196	<i>unknown</i>
ASM1184e	ASMedia	4	<i>unknown</i>	None	BGA-196	14 €

There ain't many manufacturers offering this type of IC, and their products and documentation often aren't easily acquirable.

Regarding the number of ports, every solution is adequate, although the PI7C9X2G606PR features 5 ports, which is excessive.

One of the most important factors in the decision of this component is the Package. BGA packages, which stand for *Ball Grid Array*, are extremely difficult to solder without proper industrial equipment. For a prototype phase, the LQFP packages are much more forgiving. Figure 5.16 shows a comparison of both packages.



Figure 5.16: Comparison of IC packages

This means that it is impossible to achieve a number of ports greater than 3 without using BGA packages. However, as previously said, this design features only 1 Row-A, which means that only 3 ports is necessary.

The decision between both versions of the PI7C9X2G404 IC offered by Diodes is made not only by looking at the power consumption but also the current availability. The selected IC was the PI7C9X2G404SV.

This is one of the most difficult and time-consuming parts of this project. As such, a lot of attention is needed to prevent issues further down the development process. The manufacturer has a collection of extremely useful documents both for the design of the schematic and the routing of the circuit board. Although they aren't available to the public on their website, they provide these technical documents when formally requested, free of charge.

Starting with the power section, the datasheet [37] specifies that in addition to the typical 3.3 V power source, the IC requires an additional 1 V supply. Figure 5.17 shows the power section of the schematic.

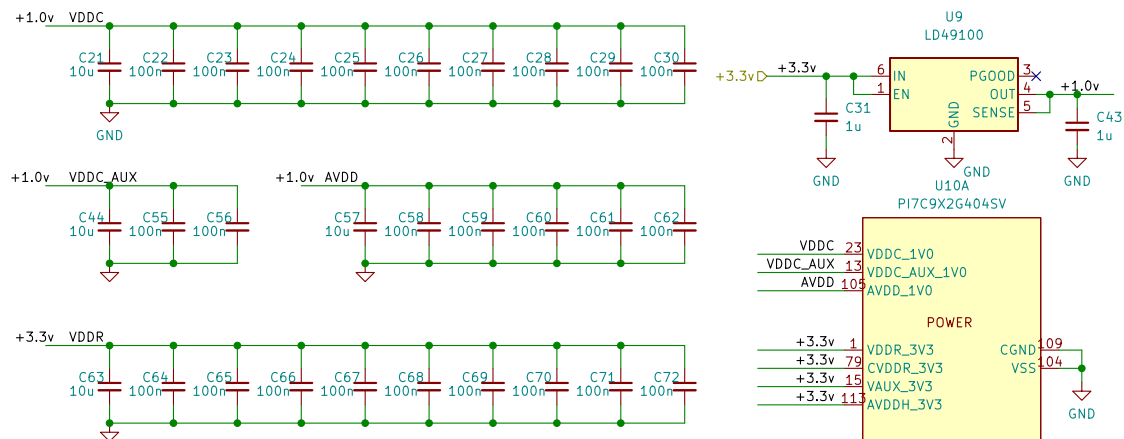


Figure 5.17: Schematic of PCIe switch power

Finally, the manufacturer also requires some pins to be either pulled up or pulled-down, to set the device into a functional mode. These pins are listed in Section 3 of the datasheet [37]. Additionally, an I2C EEPROM is also included in order to allow for further configuration changes in the future, if needed. Although this wasn't actually necessary, it is still useful to be able to use it. This miscellaneous section of the schematic is shown in Figure 5.20.

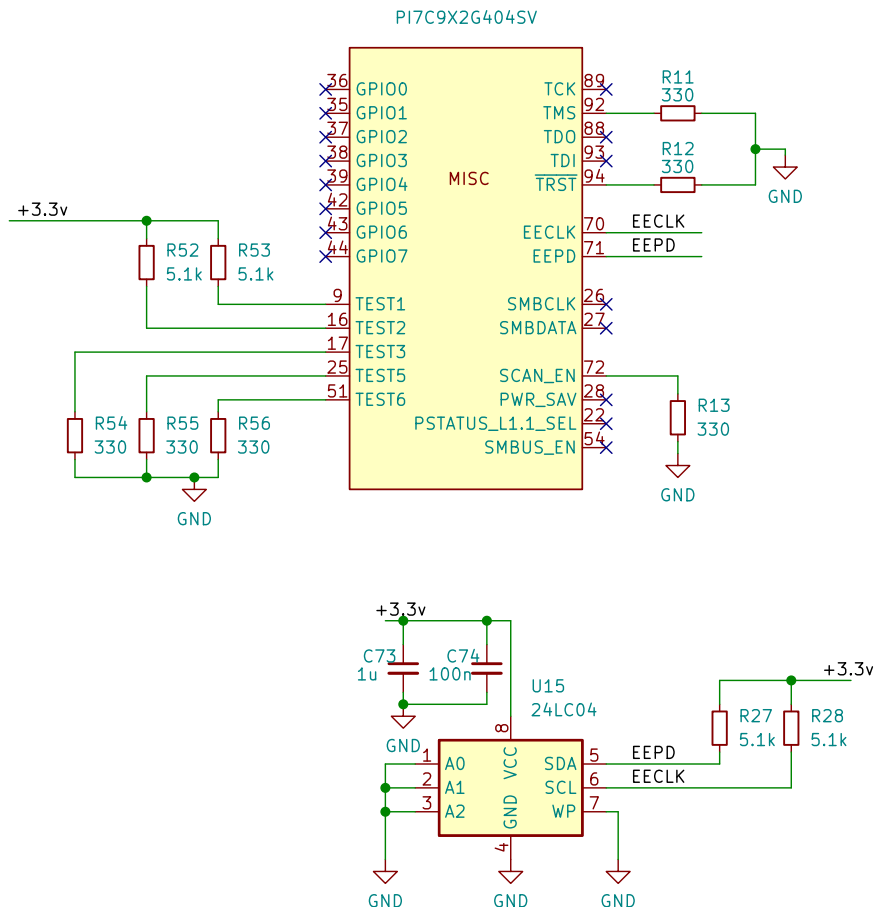


Figure 5.20: Schematic of PCIe switch miscellaneous pins

This design was further validated against a schematic checklist provided by the manufacturer in [38]. In addition to that, Diodes also includes reference schematics in the documentation, which were heavily used.

5.3.4 Power

To power the different components in the circuit board there needs to exist two power supplies: a 3.3 V supply, responsible for powering the expansion modules and most of the board's ICs; and a 5 V supply, for powering the processing unit.

To accomplish this, there are two approaches to power distribution that need to be evaluated (see Figure 5.21).

- **External** - The power supplies are placed externally, feeding the two different voltages to the circuit board.
- **Internal** - The circuit board has one unregulated power supply, and uses it to create the other two, internally.

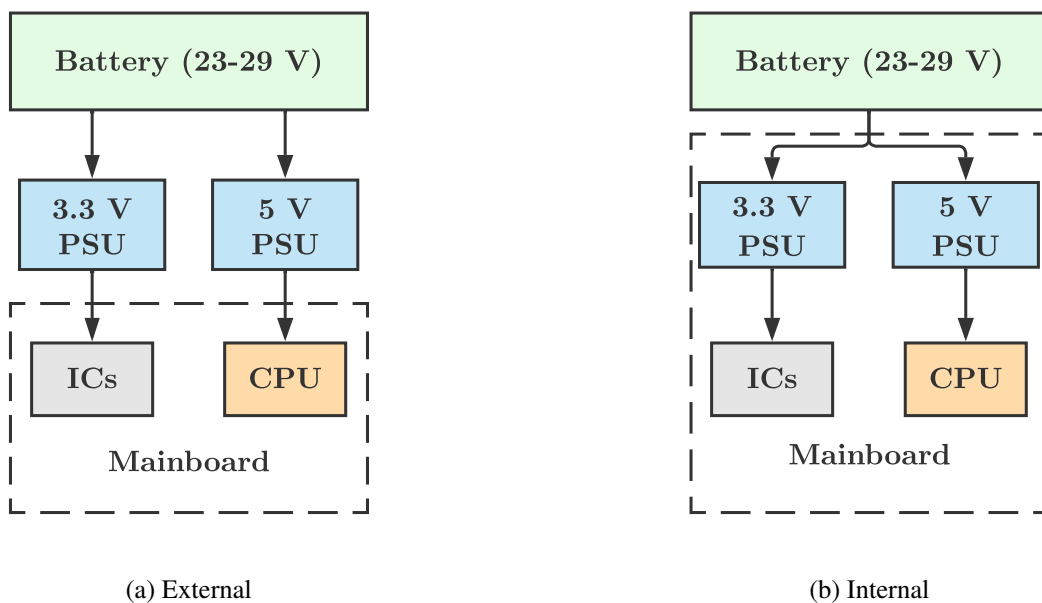


Figure 5.21: Block diagram of power supplies

Selecting the best approach is not an easy task, both solutions have advantages and disadvantages. At a first glance, it is easy to conclude that placing the power supplies externally leads to a decrease in complexity in the design of the circuit board, which is true. However, at the cost of simplicity, this design leads to less efficient power distribution.

When placed outside of the circuit board, there needs to be additional wiring from the power supplies. At lower voltages (3.3 and 5 V), the current becomes substantial, and with that the power losses in the cables become a problem. These power losses are directly proportional to the length of the cable and to the square of the current passing through it. And if that wasn't enough, the increase in inductance can also lead to issues when delivering peaks of current. To decrease these negative effects, it is crucial to minimize the length of the cable.

Placing the power supplies on the circuit board is then the best solution when it comes to efficiency, even if that increases the complexity of the development. It also provides a simpler way to power the device and it ensures that the selected power components are optimized for this type of application, further increasing power efficiency. This was the approach taken in this design.

5.4 Detailed design of the Expansion Modules

To test and validate the modularity of the designed Main Board, a set of useful modules was developed. This section specifies the development process of each of these modules.

5.4.1 GPS Module

Some commercially available cellular modems also feature a GPS antenna. However, the bundled GPS antennas are often low-performance and can't be easily upgraded. Therefore, a standalone GPS module was valuable to the target application.

The currently used modems are from the NEO family offered by u-blox. There are three main series used by LSTS: NEO-M8, NEO-M8P, and NEO-M8T.

The NEO-M8 series is a general-purpose low-cost solution, used for applications that don't require a very high position precision. The NEO-M8T module delivers high integrity, and precision timing in demanding applications worldwide. The NEO-M8P module combines the high-performance u-blox M8 positioning engine with u-blox's Real-Time Kinematic (RTK) technology, achieving a very high position precision. Although they have different features, they are all pin-compatible, which means that it is possible to design one solution compatible with all products.

5.4.1.1 Schematic

As the modem does not feature a PCIe nor a UIM interface, the designed card is a Type-B module with an M.2 E-Key. The selected card size was the M.2 2242. As previously detailed, this option provides two communication interfaces: UART and USB. While the modems provide both interfaces, the UART interface provided by the M.2 connector uses a 1.8 V logic level, which is not compatible with the modems. Therefore, instead of implementing an additional logic-level translator, the selected communication interface was USB.

Figure 5.22 shows the schematic of the M.2 edge-card connector. In addition to having the USB differential data line, it also uses the two onboard status LEDs, as well as a disable pin to cut off the modem's supply.

The modem's supply features a AP22817AKA power switch, with a current limit of 1.5 A, and its schematic is detailed in Figure 5.23.

Figure 5.24 shows the schematic of the GPS modem. Starting with the USB line, the manufacturer specifies in Section 1.4.2 of [39] that the lines require a 27 Ohm serial termination resistor. Pins 15 and 16 are connected to the onboard status LEDs. However, only the M8P variant uses those pins. The antenna section of the schematic was designed according to the guidelines in Section 2.4.2 of [39]. The TIMEPULSE pin generates pulse trains synchronized with GPS or UTC time grid, with intervals configurable over a wide frequency range. It is sometimes as a 3.3 V logic level, but it is more common to see it as an RS232 signal.

Figure 5.25 shows the schematic of the TIMEPULSE signal output, that uses the MAX3221 IC to convert the signal to an RS232 logic level.

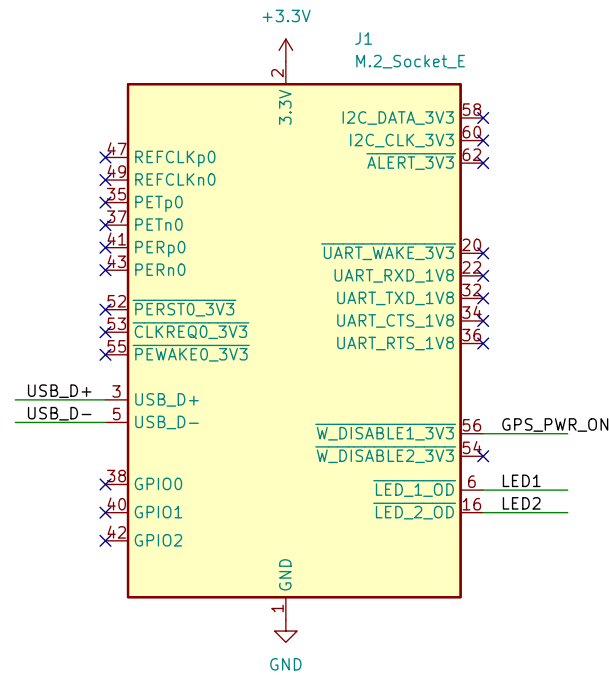


Figure 5.22: Schematic of GPS Module connector

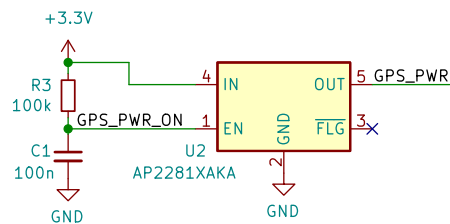


Figure 5.23: Schematic of GPS Module power supply

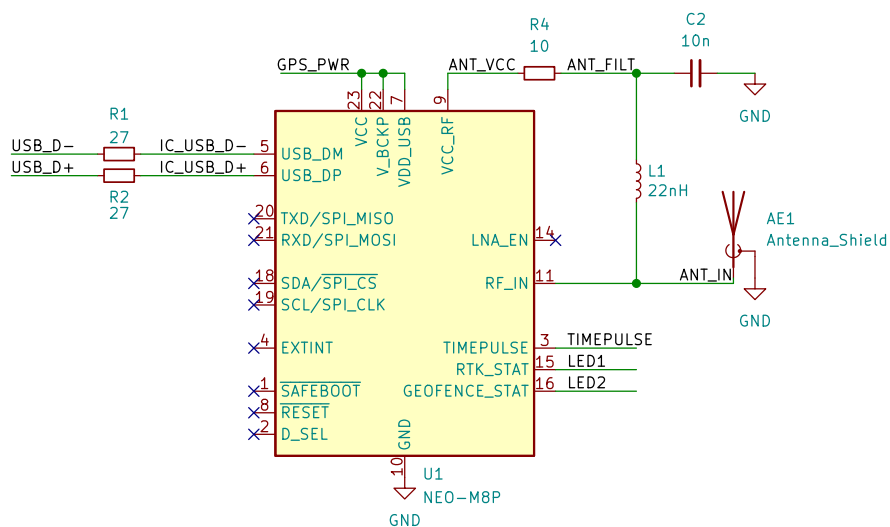


Figure 5.24: Schematic of GPS Module modem

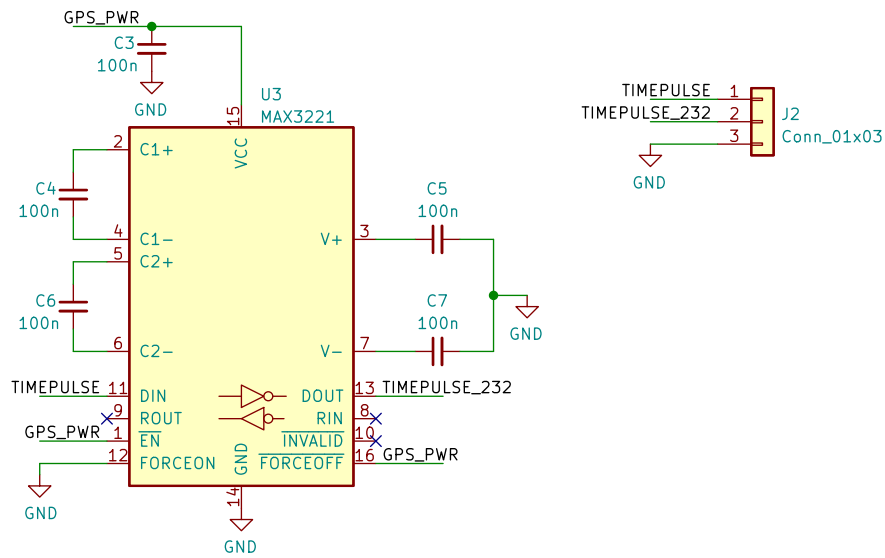


Figure 5.25: Schematic of GPS Module TIMEPULSE signal output

5.4.1.2 Layout

The designed board is a 4-layer PCB, with two inner power planes, and two outer signal layers. The placement and routing are pretty straightforward, except for some aspects that will be further discussed. Figure 5.26 shows the PCB layout.

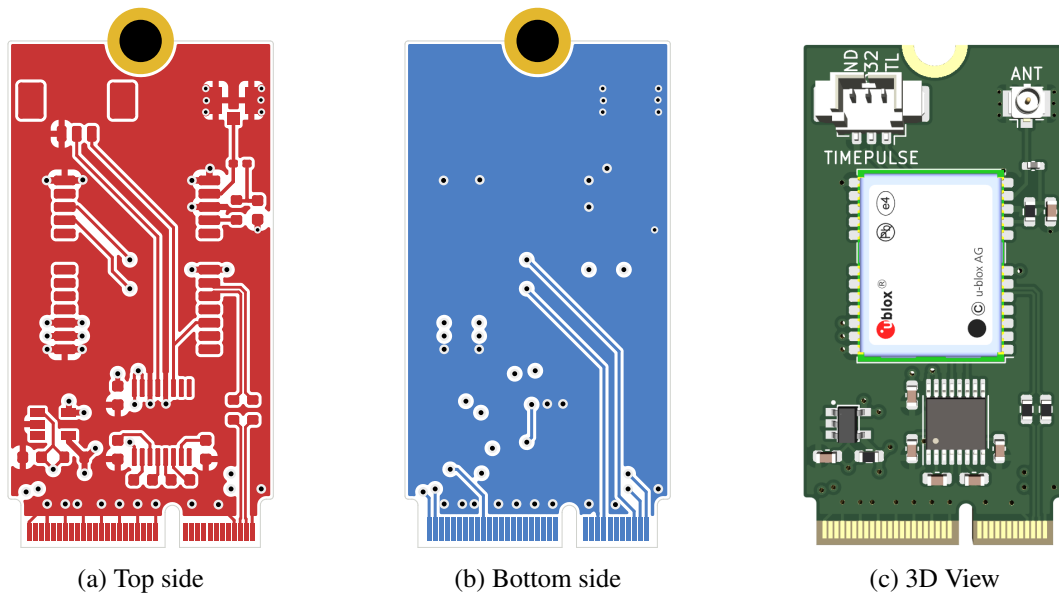


Figure 5.26: PCB of GPS Module

First of all, the USB specification requires the data signals to have a $90\Omega \pm 15\%$ differential impedance. This results in a 0.2289 mm track width, and a 0.1524 mm differential clearance.

Furthermore, the antenna section of the PCB requires a more careful design. When working with RF, there are additional considerations that need to be taken into account. In [40], the authors

specify that the antenna trace impedance must be $50\ \Omega$, with a length as short as possible, while avoiding layer changes through vias. They also recommend the usage of dedicated power planes, with the ground layer directly beneath the antenna connector, which is achieved with the selected layer stack up.

5.4.2 Iridium Module

Another type of wireless communication used by maritime applications is satellite. Thus, the next developed card was the Iridium module.

Iridium is a global satellite communication company, providing access to data services anywhere on Earth [41]. In order to use their services, a custom-developed modem, developed by Iridium, must be used. This subsection describes how this proprietary module was integrated into an expansion card for the Main Board.

There are two main modems currently used by LSTS. These products are compared in Table 5.5.

Table 5.5: Comparison of the commercially available Iridium modems

Reference	Release Date	Dimensions	Price
9602	2010	41.0x45.0x13 mm	250 €
9603	2012	31.5x29.6x8.1 mm	250 €

The original 9602 was superseded by the 9603. Although it is still used in some applications, it is strongly recommended that new projects use the newer modem. Furthermore, the M.2 specification lists a maximum card width of 30mm, and while the 9603 barely fits, the 9602 can't be really integrated into the form factor. The selected modem was the Iridium 9603.

5.4.2.1 Schematic

Just like the GPS module, the Iridium card is a Type-B module with an M.2 E-Key. However, while the GPS card uses the 2242 size, this module uses a 3042 size, to accommodate the Iridium modem.

The only way of communicating with the Iridium modem is through the UART interface. Luckily, the Type-B module features just that. Figure 5.27 shows the schematic of the M.2 edge-card connector.

Just like the previous module, it uses the two onboard status LEDs and it also features a pin to turn off the module. However, the communication interface is now the UART port. Unfortunately, the M.2 specification requires these signals to have a 1.8 V logic level, while the modem expects 3.3 V (Section 3.1.2 in [42]). The solution is to use a voltage level translator. The TXB0104D IC was the selected option. To create the 1.8 V voltage reference, the MIC5365 Low-Dropout linear regulator (LDO) was used.

The modem also provides a Ring Indicator pin, to notify that a message is queued on the Gateway. This signal is also translated to 1.8 V to comply with the specification.

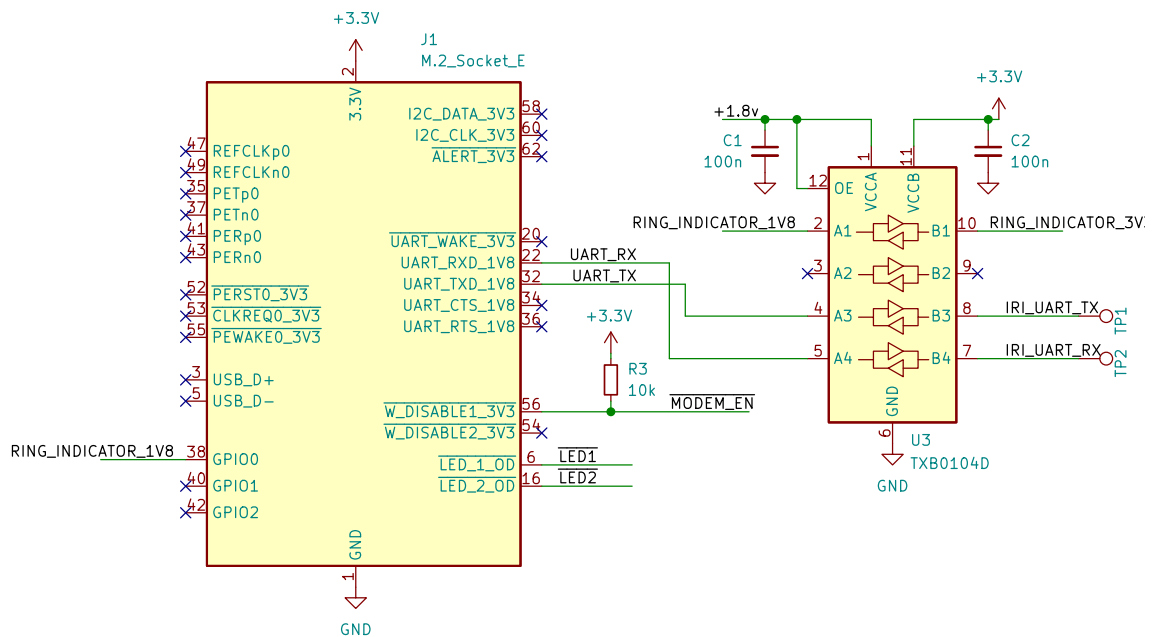


Figure 5.27: Schematic of Iridium Module connector

One of the most important and challenging aspects of this module is the modem's power supply. The Iridium modem requires a stable 5 V power supply, but the M.2 connector only provides 3.3 V. The solution is to use a DC to DC Boost converter. Unfortunately, the modem has a peak current of 1.3 A, and the current availability of high-current Boost converters is so low that there weren't many solutions to evaluate, and the only acquirable IC was the LM2623. While this product has all the required features, it might be useful to compare it to other solutions in the future, when they are available. The schematic of the converter is shown in Figure 5.28.

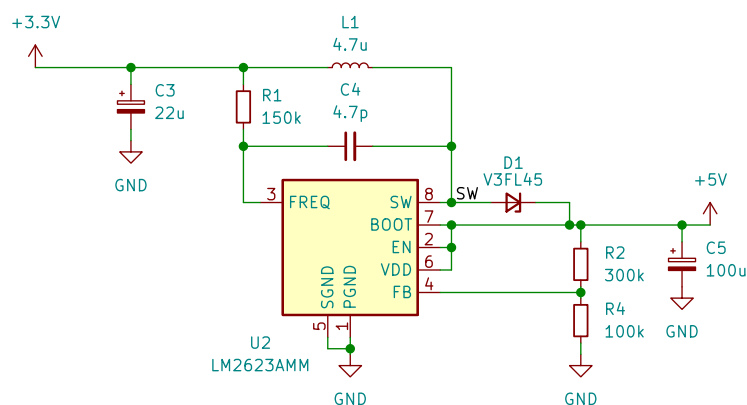


Figure 5.28: Schematic of Iridium Module Boost converter

Fortunately, the manufacturer supplies an extremely complete datasheet [43], with application guidelines that were followed in this design.

Finally, the modem's schematic is shown in Figure 5.29.

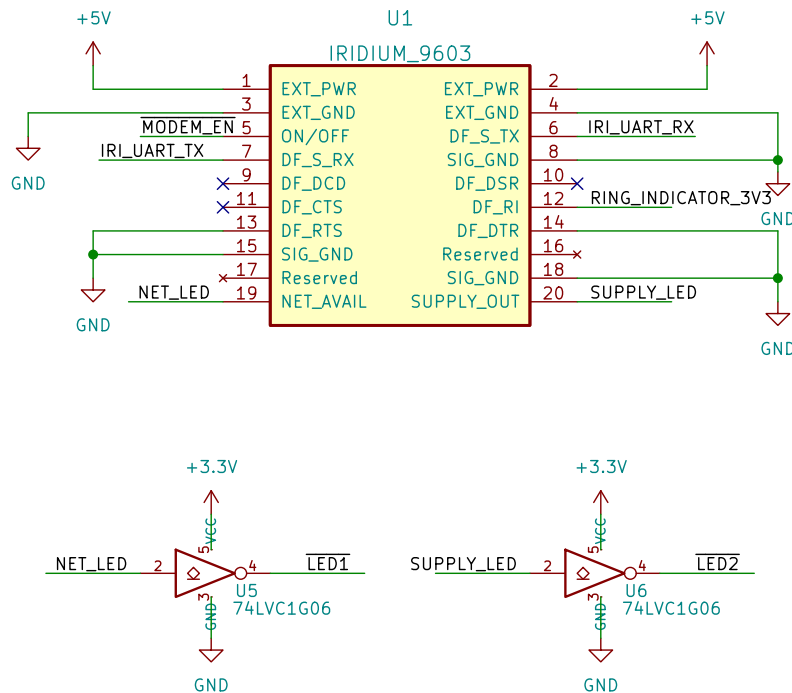


Figure 5.29: Schematic of Iridium Module modem

Here we can see all the pins previously described in the M.2 card-edge of the module. In addition to that, the modem also connects the on-board status LEDs to the NET_AVAIL and SUPPLY_OUT pins. These pins signal when the modem can see an available network, and when the modem is being powered respectively. However, these pins follow a positive logic while the LEDs expect the reverse. To achieve that, two basic 74LVC1G06 inverters were used.

5.4.2.2 Layout

The designed board is a 4-layer PCB, with two inner power planes and two outer signal layers. It follows the exact PCB stack of the GPS module. Figure 5.30 shows the PCB layout.

In comparison to the GPS module, this layout is considerably simpler. It doesn't require any impedance matching or an onboard antenna. The most important aspect is the trace width of the power lines, which needs to be increased. The two mounting holes on the bottom are used to secure the Iridium modem with screws.

5.4.3 Serial Module

One of the most useful communication interfaces is the serial port. Therefore, it is useful to design a way to increment the number of serial ports in a system. This module not only creates 4 new serial ports but also provides them in an RS232 logic-level, which is one of the most used forms of communication between systems in a vehicle.

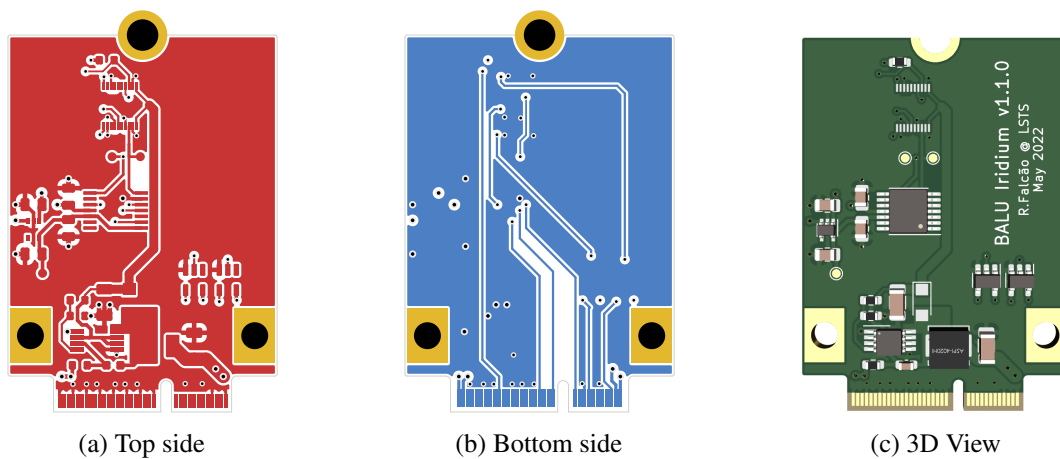


Figure 5.30: PCB of Iridium Module

Section 5.3.3.2 describes the process of choosing a UART multiplexer for the Main Board. The selected option was an IC offered by Maxim that used the SPI interface to create 4 new UART channels. However, neither M.2 keys offer access to the SPI bus. Nevertheless, when researching the available solutions, there was a set of ICs that used the USB port instead. Table 5.6 shows the previously featured USB to UART multiplexers.

Table 5.6: Comparison of the commercially available USB-UART Multiplexer ICs

Reference	Mfr.	#P	Proto	Speed	Power	Avail.	Package	Price
XR21B1424	MaxLinear	4	USB	12M	50 mW	Normal	LQFP-64	9 €
FT4232	FTDI	4	USB	12M	100 mW	Very Low	LQFP-64	7 €

These two solutions are very similar functionally, and either one would be suitable for the design. However, the lower power consumption and the higher availability of the offer from MaxLinear led to the selection of the product.

5.4.3.1 Schematic

As the module only uses a USB interface, the designed card is also a Type-B module with an M.2 E-Key. The selected card size was the M.2 2242.

Figure 5.31 shows the schematic of the M.2 edge-card connector. Here we can see the differential USB lines as well as one Main Board status LED being used. The USB lines feed into the multiplexer IC, shown in Figure 5.32.

The first thing to notice is the four channels created by the IC. Although they provide hardware flow-control pins and additional GPIOs, the only used functions of each channel are the transmit and receive pins. Additionally, the module also features a pair of LEDs per channel signaling when data is input or output, located on the left side of the schematic. Finally, the Main Board status LED is connected to the USB_STAT1 pin, which is used to indicate the USB communication status [44].

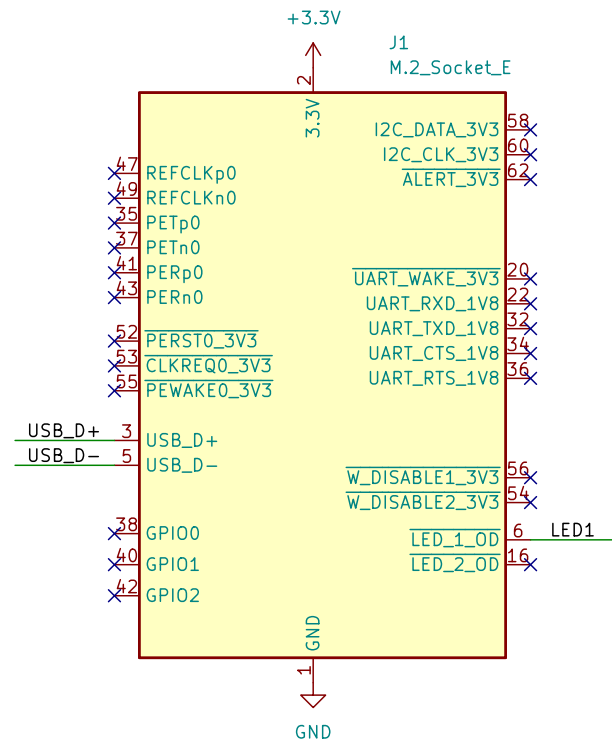


Figure 5.31: Schematic of Serial Module connector

The conversion of the UART to RS232 signals is all that remains to be discussed. This is accomplished using an RS232 transceiver. While there aren't many commercial ICs featuring 4 channels, it is possible to use two ICs each with 2 channels. This is exactly what was designed, using the ICL3232 IC, from Renesas.

Following the datasheet supplied by the manufacturer and its application information [45], the schematic in Figure 5.33 was developed.

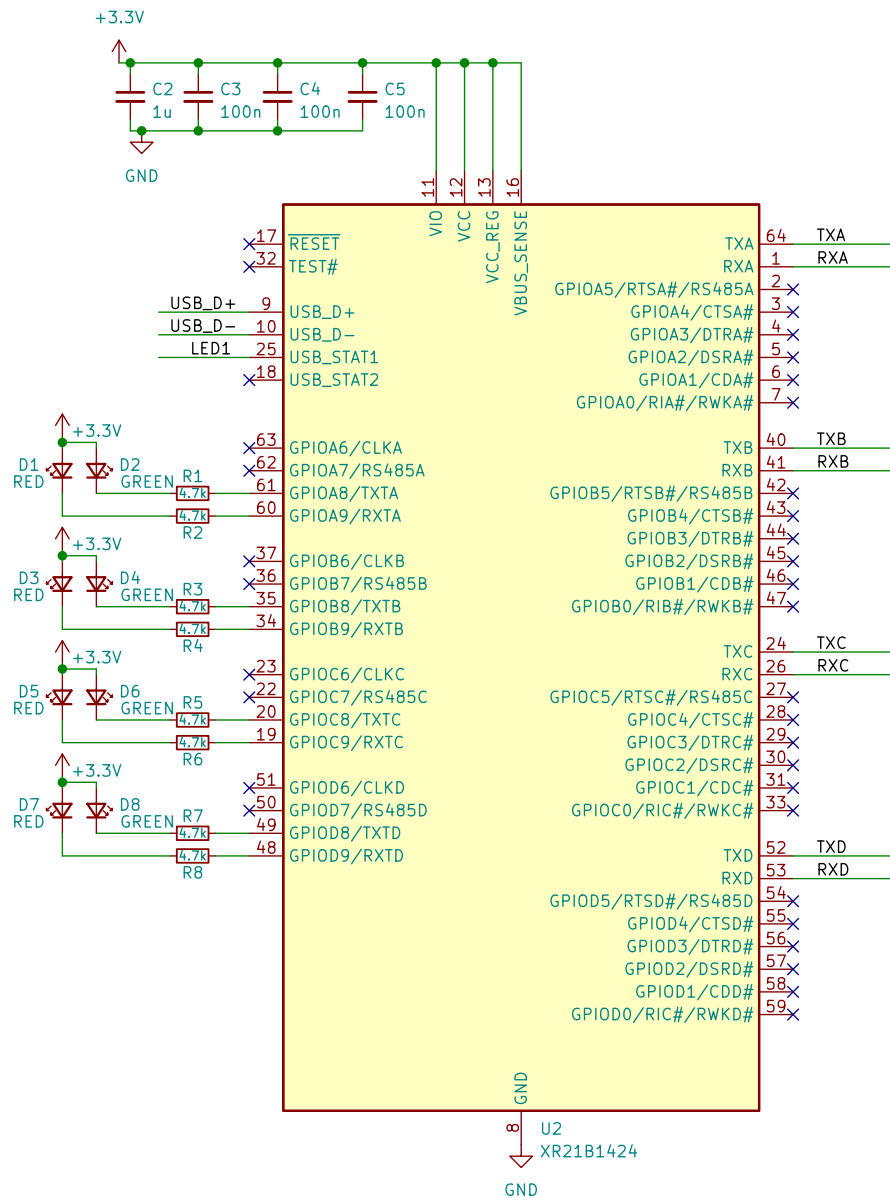


Figure 5.32: Schematic of Serial Module multiplexer

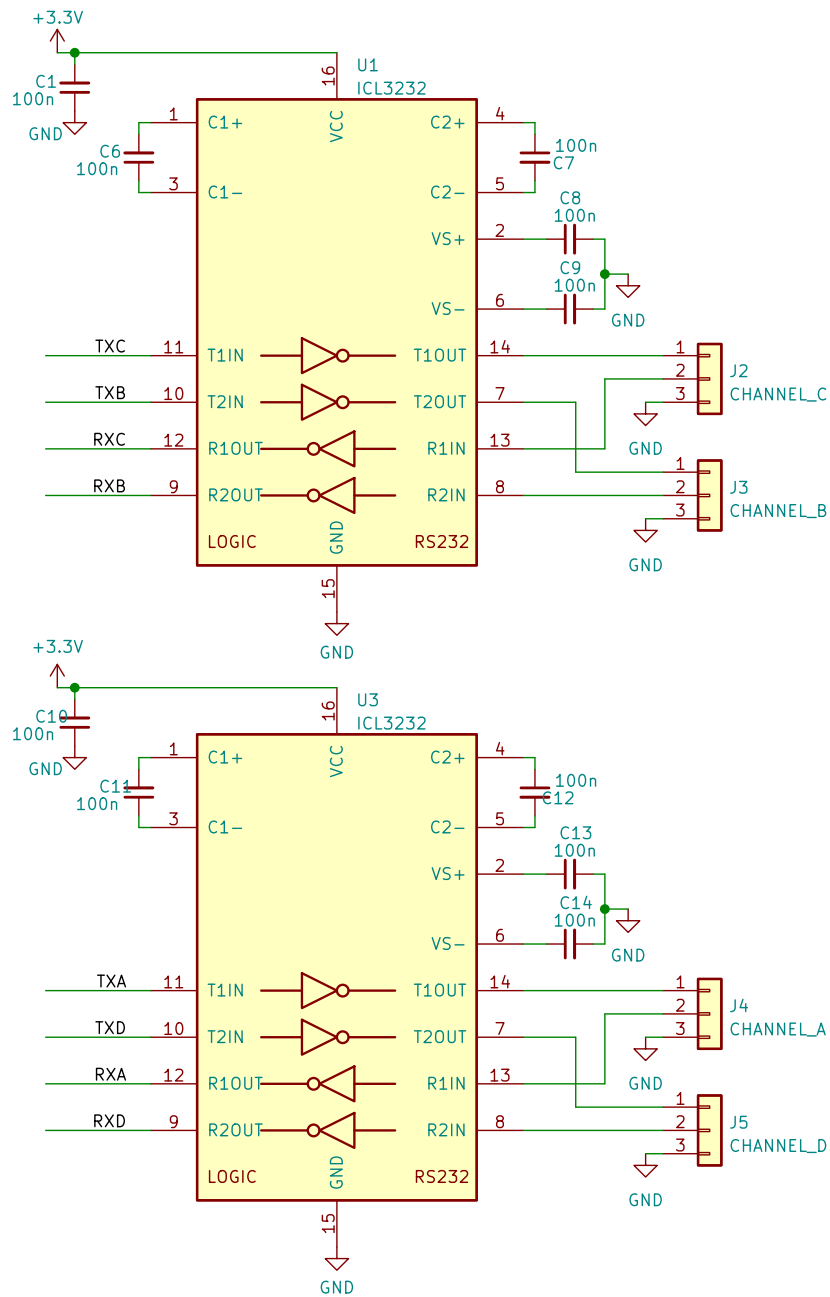


Figure 5.33: Schematic of Serial Module RS232 transceivers

5.4.3.2 Layout

The designed board uses the exact same PCB stack as the previous modules. Figure 5.34 shows the PCB layout.

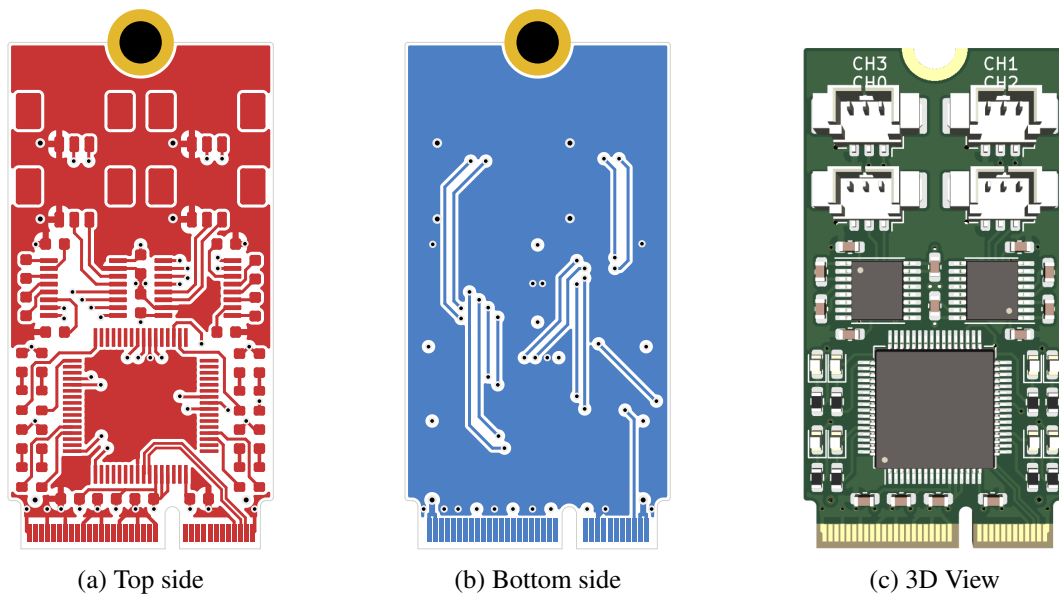


Figure 5.34: PCB of Serial Module

The most important aspect of the routing of this PCB is the differential impedance of the USB line. Like the GPS module, the pair was tuned to a 90Ω differential impedance, which led to a 0.2289 mm track width with a 0.1524 mm differential clearance.

One minor issue is the layout of the 4 channel connectors. They do not follow any natural ordering, mainly because of the positioning of the pins in the multiplexer IC. However, this is perfectly acceptable because the circuit board also has a silk-screen label on each of the channels, minimizing errors in the application of this card.

Chapter 6

Implementation

Before designing and manufacturing the Main Board and expansion modules, it was extremely important to create an early minimal functioning prototype, not only to understand if the architecture is viable and applicable but also prevent major issues in the final revision.

As such, the manufacturing process began with the development of an extremely basic revision of the Main Board (Figure 6.1), as well as the two most challenging expansion modules: GPS and Iridium.

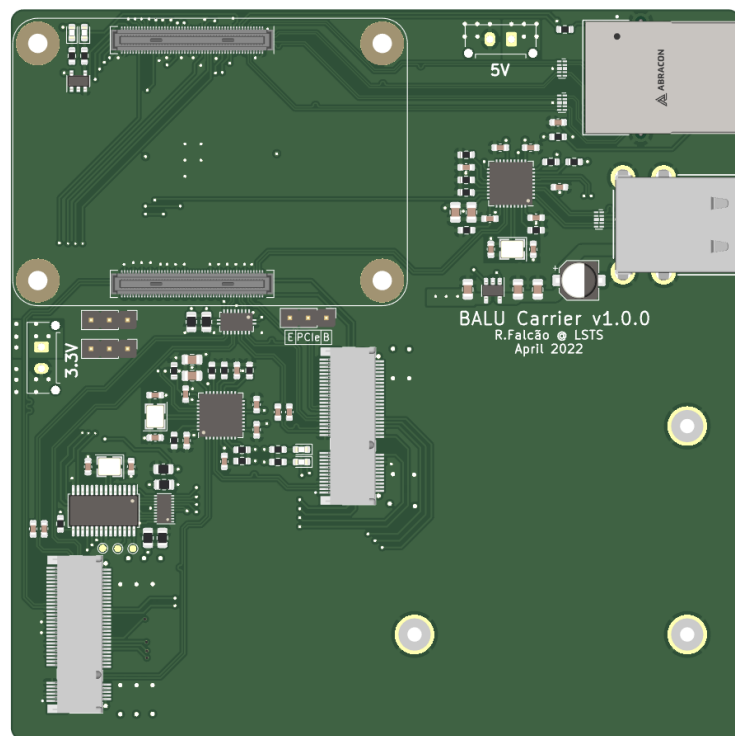


Figure 6.1: First revision of the Main Board

The board features the Raspberry Pi Compute Module 4 and some basic peripherals: an Ethernet port and two USB ports. In addition to that, it featured only two M.2 ports. One E-Key

connector, featuring PCIe, USB, and UART interfaces; and a B-Key connector, with PCIe, USB, and UIM interfaces.

It should also be noted that this revision didn't feature the internal power supplies. Instead, it sourced the required 5 V and 3.3 V externally.

This allowed the validation of the USB hub and the UART multiplexer circuitry, as the goal was to use them as it is in the final revision.

In addition to that, the M.2 connectors were used to test a number of different commercial expansion cards, and also the custom-developed modules. One issue of this revision is that the CPU only features one PCIe port, and no PCIe switch was yet integrated. This meant that the slot in which the PCIe port is connected must be selected before turning on the system, preventing the usage of two PCIe cards at the same time.

After this initial prototype, the final revision was designed and manufactured, fixing the issues found on the first approach. The final design of the Main Board is shown on Figure 6.2.

Both the Main Board and the designed expansion modules were run through a verification and validation process. The verification step checks the project's functionality against the design requirements, and assesses if everything was achieved. The validation process actually tests the project to see if their functionality is working correctly.

6.1 Main Board

When compared against the requirements from Table 4.1, some conclusions can be drawn. First of all, every high priority requirement was successfully fulfilled. Secondly, unfortunately the modules power management system could not be designed in time. This would be an useful addition in the future but for now, it isn't a high priority functionality. Finally, the secondary processor couldn't be implemented too.

6.1.1 Operating System

In order to simplify the validation of the hardware components, the operating system used while testing was the latest version of the official 64-bit Raspberry Pi OS, with the Linux Kernel on version 5.15. However, some of the modules and drivers used by the Main Board were not included out of the box, leading to the need of recompiling the Linux Kernel.

The processor's architecture is ARM64, and the typical laptops and desktop computers feature an x86 architecture. There are then two ways of compiling the ARM Linux Kernel:

1. Compile the Kernel on an ARM64 processor. As there isn't a usable ARM64 workstation, the Raspberry Pi must be used.
2. Cross-compile the Kernel on an x86 processor. This requires the installation of a cross-compiler (`aarch64-linux-gnu-*`) on the host platform.

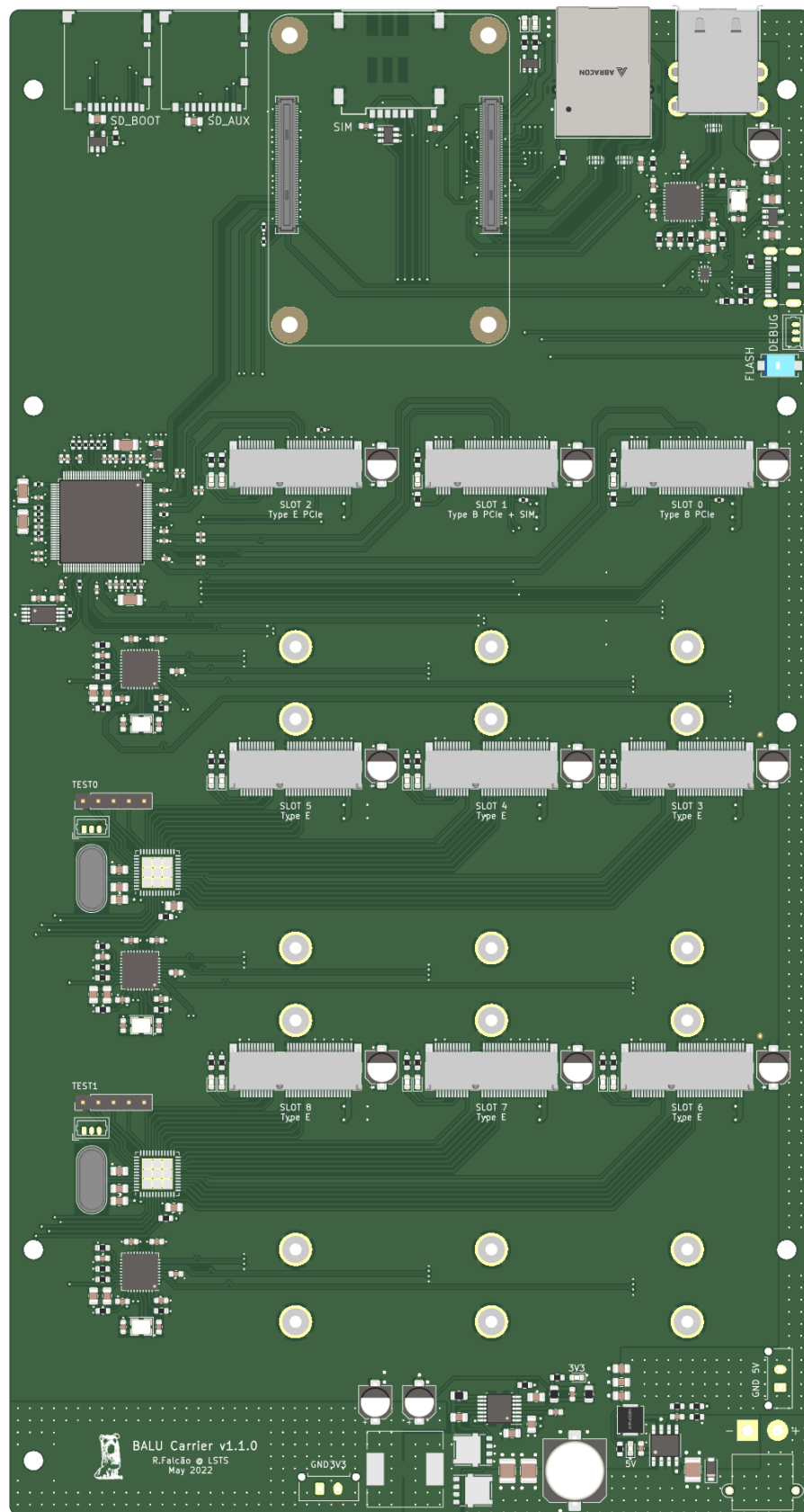


Figure 6.2: Final revision of the Main Board

While the first option is simpler, it also is extremely slow, as the Raspberry Pi can't compete with modern desktop processors. To give a comparison, compiling the Kernel on the Raspberry Pi takes around 1 hour to complete while a modern laptop achieved it in 7 minutes. This, of course, depends on the type of x86 processor used, but in this case, it became clear that it is advantageous to cross-compile. As such, following the cross-compiling documentation on Raspberry Pi's website [30], the Kernel was compiled using the second option.

To summarize, the cross-compiling process is the following:

1. Clone the official Raspberry Pi Linux repository:

```
$ git clone --depth=1 https://github.com/raspberrypi/linux
```

2. Initialize the default Kernel configuration file (.config):

```
$ make ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- \
    bcm2711_defconfig
```

3. Change the configuration file:

```
$ make ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- menuconfig
```

4. Compile the Kernel image, modules, and device tree blobs. The `-j 16` flag indicates the number of processor threads that are used in the compilation.

```
$ make -j 16 ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- Image
$ make -j 16 ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- modules
$ make -j 16 ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- dtbs
```

After these steps, the Kernel is compiled and ready to be used. If the target processor uses an external SD card, it must be inserted and sent the generated files. However, since the used Compute Module 4 features an internal eMMC storage, it is required to connect to the processor's USB port and execute the manufacturer `usbboot` tool:

1. Clone the tool's repository:

```
$ git clone --depth=1 https://github.com/raspberrypi/usbboot
```

2. Compile the source code:

```
$ make
```

3. Execute the tool:

```
$ sudo ./rpiboot
```

This will automatically map any connected Raspberry Pi to a flash drive in the computer, allowing access to the internal storage.

6.1.2 USB validation

In order to decrease the power consumption, the Compute Module 4's USB controller is disabled by default. After flashing the Raspberry Pi OS to the processor's internal storage, as specified

in the previous section, the USB controller must be enabled in the device's configuration file (`/boot/config.txt`) by adding the following line:

```
dtoverlay=dwc2,dr_mode=host
```

After rebooting the device, although enabled, the USB controller didn't detect any USB-hub connected to the processor's port. Listing the USB devices output the following:

```
$ lsusb -t
/: Bus 02.Port 1: Dev 1, Class=root_hub, Driver=xhci-hcd/0p, 5000M
/: Bus 01.Port 1: Dev 1, Class=root_hub, Driver=xhci-hcd/1p, 480M
```

Unfortunately, the issue was actually in the hardware of the Main Board. In the USB-hub IC's datasheet [32], the manufacturer highlights its PortSwap feature, which allows changing the data signals polarities to avoid uneven trace length or crossing of the USB differential signals on the PCB. This was used in some of the USB differential lines in the PCB to facilitate trace routing and prevent signal layer changes. However, while at first glance it seems that this polarity inversion is automatically detected, this feature must be manually enabled through an external EEPROM, which wasn't included in the prototype. This oversight was easily fixed by manually swapping the differential pair lines, using a very thin 0.1mm copper wire. While not elegant and extremely dangerous in production environments, it fixed the issue and allowed the testing of USB devices to continue. This issue was obviously addressed and fixed in the final revision of the circuit board.

The output of the USB devices list is now the following:

```
$ lsusb -t
/: Bus 02.Port 1: Dev 1, Class=root_hub, Driver=xhci-hcd/0p, 5000M
/: Bus 01.Port 1: Dev 1, Class=root_hub, Driver=xhci-hcd/1p, 480M
   |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/4p, 480M
   |__ Port 3: Dev 3, If 0, Class=Hub, Driver=hub/4p, 480M
```

Here we can see that both the IO USB hub and the M.2 USB hub are correctly detected, and their communication speed is successfully negotiated (480M). In addition to the detection of the USB hubs, it is necessary to validate each of the USB channels.

Starting with the USB ports on the circuit board, a flash drive was connected to each of the ports. Analyzing the Kernel's messages, we can see that the flash drive is correctly detected and mapped to the `/dev/sda` device:

```
$ dmesg
(...)

usb 1-1.1: new high-speed USB device number 5 using xhci-hcd
usb 1-1.1: New USB device found, idVendor=0930, idProduct=1408,
    bcdDevice= 0.01
usb 1-1.1: New USB device strings: Mfr=1, Product=2, SerialNumber=3
usb 1-1.1: Product: USB FLASH DRIVE
```

```

usb 1-1.1: Manufacturer: TOSHIBA
usb 1-1.1: SerialNumber: C412F52D6C86C17160020C59
usb-storage 1-1.1:1.0: USB Mass Storage device detected
scsi host0: usb-storage 1-1.1:1.0
scsi 0:0:0:0: Direct-Access TOSHIBA USB FLASH DRIVE PQ: 0 ANSI: 6
sd 0:0:0:0: [sda] 30310400 512-byte logical blocks: (15.5 GB/14.5
    GiB)
sd 0:0:0:0: [sda] Write Protect is off
sd 0:0:0:0: [sda] Mode Sense: 4f 00 00 00
sd 0:0:0:0: [sda] Write cache: disabled, read cache: enabled, doesn'
    t support DPO or FUA
sd 0:0:0:0: Attached scsi generic sg0 type 0
sd 0:0:0:0: [sda] Attached SCSI removable disk

```

To test the M.2 USB hub, it is necessary to connect an expansion module that uses the USB interface. When connecting the designed GPS module (which uses the USB interface), it is correctly recognized, as shown in the Kernel's messages:

```

$ dmesg
(...)

usb 1-1.3.1.3: new full-speed USB device number 6 using xhci-hcd
usb 1-1.3.1.3: New USB device found, idVendor=1546, idProduct=01a8,
    bcdDevice= 3.01
usb 1-1.3.1.3: New USB device strings: Mfr=1, Product=2,
    SerialNumber=0
usb 1-1.3.1.3: Product: u-blox GNSS receiver
usb 1-1.3.1.3: Manufacturer: u-blox AG - www.u-blox.com

```

This module will be further discussed in the following sections.

6.1.3 Ethernet validation

The prototype features an Ethernet port, capable of reaching speeds of up to 1 Gbps. When connected to a local network, we can see that the system automatically connects and sets up an internal IP. Listing the system's network interfaces outputs the following:

```

$ ip addr
(...)

2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state
    UP group default qlen 1000
    link/ether e4:5f:01:50:04:98 brd ff:ff:ff:ff:ff:ff

```

```

inet 10.0.2.75/24 brd 10.0.2.255 scope global noprefixroute eth0
    valid_lft forever preferred_lft forever
inet6 fe80::20b0:66:8885:8ce7/64 scope link
    valid_lft forever preferred_lft forever

```

However, in addition to connecting to a network, it is necessary to benchmark the transmission and reception speed of the system. To do so, a tool named iPerf3 was used.

iPerf3 [46] is a tool for active measurements of the maximum achievable bandwidth on IP networks between two systems - one as a server, and the other as a client. To get started, the server is initialized in another computer connected to the network using

```
$ iperf3 -s
```

and the Raspberry Pi must start the benchmark using

```
$ iperf3 -c 10.0.2.76
```

where 10.0.2.76 is the server's local IP.

The test was executed both as a client and as a server, and the results are listed in Figure 6.3.

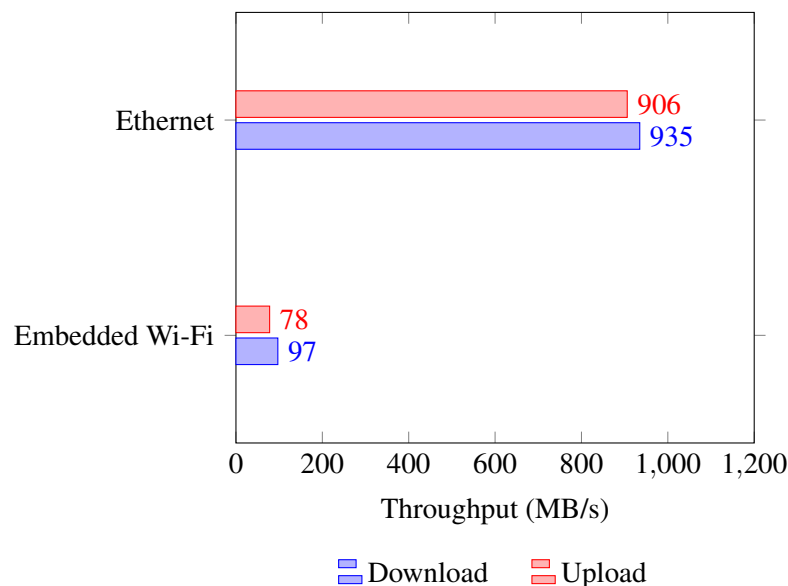


Figure 6.3: Ethernet performance tests

Analyzing these results, we can conclude that the Ethernet port is working as expected.

6.1.4 UART validation

While USB devices are automatically detected, we need to configure the system to recognize the UART multiplexers connected to the SPI bus. This is accomplished by configuring the Linux device tree.

In [47], the authors define the device tree as a "tree structure used to describe the physical hardware in a system. Each node in the tree describes the characteristics of the device being

represented. The purpose of the device tree is to describe device information in a system that cannot necessarily be dynamically detected or discovered by a client program."

On a Raspberry Pi or Compute Module there are several files in the boot partition of the operating system that are binary 'Device Tree' files. These binary files (usually with extension .dtb) are compiled from human-readable text descriptions (usually files with extension .dts) by the Device Tree compiler.

The MAX14830 Linux driver is located in the `drivers/tty/serial/max310x.c` file. However, the driver is not compiled and included in the Raspberry Pi's kernel, by default. To change this, it is necessary to follow the compilation process in Section 6.1.1 and enable the module in the Kernel's configuration (Device Drivers → Character devices → Serial drivers → MAX310X support)

The driver also provides an example device tree description in `Documentation/device-tree/bindings/serial/maxim,max310x.txt`. From the example, the following preliminary description file (`max14830.dts`) was developed:

```
/ {
    clocks {
        spi_uart_clk: osc_max14830 {
            compatible = "fixed-clock";
            #clock-cells = <0>;
            clock-frequency = <3686400>;
        };
    };
};

&spi0 {
    max14830: max14830@0 {
        compatible = "maxim,max14830";
        reg = <0>;
        clocks = <&spi_uart_clk>;
        clock-names = "xtal";
        interrupt-parent = <&gpio>;
        interrupts = <17 8>;
        gpio-controller;
        #gpio-cells = <2>;
    };
};
```

It creates a single instance of the MAX14830 driver, connected to the processor's SPI0 bus on chip select 0. The IC has an external 3.6864 MHz crystal and uses GPIO17 as an interrupt request pin. To compile it, the following command was used:

```
$ dtc -I dts -O dtb -o /boot/overlays/max14830.dtbo ./max14830.dts
```

The file must then be enabled in the boot configuration file (`/boot/config.txt`) by adding the following line:

```
dtoverlay=max14830
```

After rebooting, the system correctly detects the MAX14830 chip, as seen in the Kernel messages:

```
spi0.0: ttyMAX0 at I/O 0x0 (base_baud = 2764800) is a MAX14830
spi0.0: ttyMAX1 at I/O 0x20 (base_baud = 2764800) is a MAX14830
spi0.0: ttyMAX2 at I/O 0x40 (base_baud = 2764800) is a MAX14830
spi0.0: ttyMAX3 at I/O 0x60 (base_baud = 2764800) is a MAX14830
```

In addition to that, it also creates four new devices, one for each UART channel:

```
$ ls /dev/ttyMAX*
/dev/ttyMAX0 /dev/ttyMAX1 /dev/ttyMAX2 /dev/ttyMAX3
```

6.2 Expansion Modules

In this section, the testing process of expansion modules is discussed. It starts by validating commercial expansion modules and then it details the custom-developed expansion cards.

6.2.1 Solid-State-Drive

A Solid-State Drive (SSD) is a solid-state device that stores data persistently. It serves the same function as a Hard Drive while featuring an extremely higher throughput and reliability. There are two types of M.2 solid-state drives: SATA and NVMe. SATA drives use the traditional SATA interface, while NVMe devices use the PCIe interface. The main difference between the two types is the performance of the drive - NVMe drives have significantly better throughput. Furthermore, the design Main Board doesn't feature a SATA interface, which means that only NVMe drives are compatible.

Therefore, the Toshiba RC100 NVMe drive was selected, with a capacity of 240 GB and an M.2 2242 form-factor (see Figure 6.4).

When connected to the system, we can see from the Kernel's messages that the drive is correctly recognized and mapped to the `/dev/nvme0` device:

```
$ dmesg
(...)

pci 0000:03:00.0: [1179:0113] type 00 class 0x010802
pci 0000:03:00.0: reg 0x10: [mem 0x00000000-0x00003fff 64bit]
pci 0000:03:00.0: PME# supported from D0 D3hot
```

```
pci 0000:03:00.0: 4.000 Gb/s available PCIe bandwidth, limited by
    5.0 GT/s PCIe x1 link at 0000:00:00.0 (capable of 15.752 Gb/s
    with 8.0 GT/s PCIe x2 link)
nvme nvme0: pci function 0000:03:00.0
nvme 0000:03:00.0: enabling device (0000 -> 0002)
nvme nvme0: allocated 38 MiB host memory buffer.
nvme nvme0: 4/0/0 default/read/poll queues
```



Figure 6.4: Toshiba RC100 SSD [48]

Figure 6.5 compares the throughput of the SSD and the Compute Module 4's internal eMMC drive using three benchmarks: *Sequential Reads*, where a large file is read sequentially; *4K Random Reads* and *4K Random Writes*, where multiple small 4 kB files are randomly read and written.

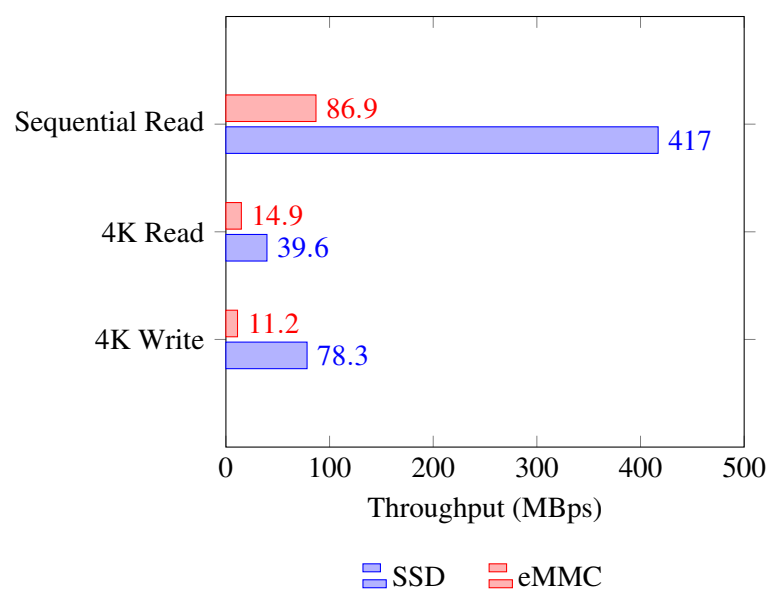


Figure 6.5: Performance comparison between SSD and internal eMMC

It is clear that the NVMe drive greatly outperforms the internal eMMC, making it the ideal solution for high-throughput file operations, like logging sensor data. Still, the performance of the device is not limited by the drive, but by the communication interface. This drive features 4 lanes of PCIe 3.0, reaching a theoretical maximum bandwidth of 4 GBps (although this limit isn't actually achievable, as the manufacturer specifies a maximum of 1600 MBps and 1100 MBps sequential read and write speeds respectively). The Raspberry Pi features only one PCIe 2.0 lane, with a theoretical maximum bandwidth of 500 MBps.

6.2.2 Wi-Fi and Bluetooth modem

The AX200 is a Wi-Fi 6 and Bluetooth 5.2 modem, developed by Intel (see Figure 6.6), with an M.2 2232 form factor. As said in a previous section, this device uses the PCIe interface for Wi-Fi, and USB for Bluetooth.



Figure 6.6: AX200 Wi-Fi and Bluetooth modem [49]

When connected to the system, we can see from the Kernel's messages that the modem is correctly detected:

```
$ dmesg
(...)

iwlwifi 0000:05:00.0: enabling device (0000 -> 0002)
iwlwifi 0000:05:00.0: api flags index 2 larger than supported by
    driver
iwlwifi 0000:05:00.0: TLV_FW_FSEQ_VERSION: FSEQ Version: 89.3.35.37
iwlwifi 0000:05:00.0: loaded firmware version 66.f1c864e0.0 cc-a0
    -66.ucode op_mode iwlvm
iwlwifi 0000:05:00.0: Detected Intel(R) Wi-Fi 6 AX200 160MHz, REV=0
    x340
iwlwifi 0000:05:00.0: Detected RF HR B3, rfid=0x10a100
iwlwifi 0000:05:00.0: base HW address: 10:51:07:c9:21:65
```

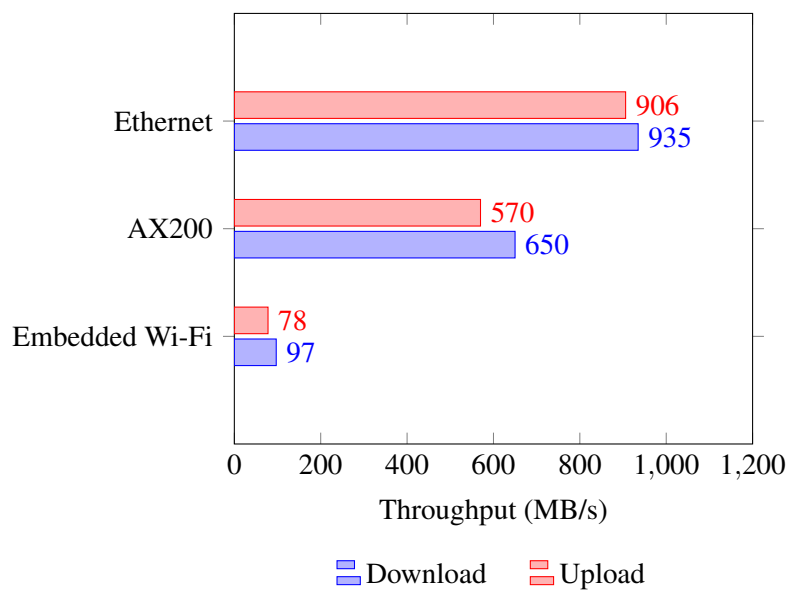


Figure 6.7: AX200 performance tests

Figure 6.7 shows the performance of the modem compared to other network solutions.

It should be noted that the bottleneck of this connection is not actually the AX200 device. The test equipment featured a Wi-Fi 5 router limited to around 640 Mbps of realistic throughput, and a Gigabit network switch. And even if the wireless equipment was able to exceed the Gigabit mark, the wired connection would be the bottleneck. To correctly test this device's capabilities, a Wi-Fi 6 router and a 2.5 Gbit network switch would be necessary.

The manufacturer actually lists a maximum theoretical throughput of 2.4 Gbps, with some users reporting a realistic performance of around 1.1 Gbps in the Raspberry Pi.

Even so, the measured performance makes the AX200 modem clearly a superior alternative to the embedded Wi-Fi modem in the Raspberry Pi.

To test the Bluetooth modem, the device was successfully paired to a smartphone (Mi 9T), using the `bluetoothctl` tool:

1. Open the bluetooth shell:

```
$ bluetoothctl
```

2. Start scanning for nearby devices:

```
[bluetooth]# scan on
[NEW] Device A8:9C:ED:6B:66:82 Mi 9T
```

3. Pair to the address of the smartphone:

```
[bluetooth]# pair A8:9C:ED:6B:66:82
[CHG] Device A8:9C:ED:6B:66:82 Connected: yes
Attempting to pair with A8:9C:ED:6B:66:82
Confirm passkey 933840 (yes/no): yes
[CHG] Device A8:9C:ED:6B:66:82 Paired: yes
```

```
Pairing successful
```

6.2.3 Cellular modem

Cellular modems use only one USB interface. However, the functionality provided by that USB interface greatly varies on the type of modem. There are two main communications protocols used by modems:

- **Mobile Broadband Interface Model (MBIM)** - an official USB standard;
- **Qualcomm Mobile Station Modem Interface (tI)** - proprietary protocol, developed by Qualcomm.

In this section, we will discuss the implementation of both types of modems.

6.2.3.1 MBIM

The ME906S is an LTE module with an M.2 interface and a 3042 form-factor (see Figure 6.8), developed by Huawei. It uses MBIM as the communication protocol.

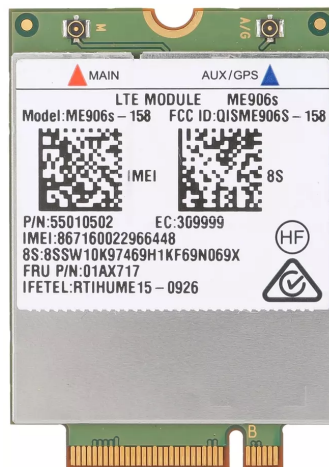


Figure 6.8: ME906S Cellular module [50]

When connected to the system, we can see from the Kernel's messages that the modem is correctly detected and the MBIM interface is mapped to the device `/dev/cdc-wdm0` and to the `wwan0` network interface:

```
$ dmesg
(...)
```

```
usb 1-1.3.3: new high-speed USB device number 7 using xhci-hcd
usb 1-1.3.3: New USB device found, idVendor=12d1, idProduct=15c1,
bcdDevice= 1.02
```

```

usb 1-1.3.3: New USB device strings: Mfr=1, Product=2, SerialNumber
=3
usb 1-1.3.3: Product: HUAWEI Mobile
usb 1-1.3.3: Manufacturer: Huawei Technologies Co., Ltd.
usb 1-1.3.3: SerialNumber: 0123456789ABCDEF

cdc_mbim 1-1.3.3:3.0: setting rx_max = 16384
cdc_mbim 1-1.3.3:3.0: NDP will be placed at end of frame for this
device.
cdc_mbim 1-1.3.3:3.0: cdc-wdm0: USB WDM device
cdc_mbim 1-1.3.3:3.0 wwan0: register 'cdc_mbim' at usb-fe9c0000.xhci
-1.3.3, CDC MBIM, be:32:8d:1f:77:df
usbcore: registered new interface driver cdc_mbim
8021q: adding VLAN 0 to HW filter on device wwan0

```

However, while the device shows up as a network interface, the modem isn't able to reach the Internet. To achieve it, it is necessary to set up the device configurations, using a tool called libmbim.

1. Verify if the device is recognized:

```
$ mbimcli -d /dev/cdc-wdm0 -p --query-device-caps
```

2. Create a new configuration file (mbim-network.conf) with the following information:

```

APN=YOUR_APN
APN_USER=YOUR_USER
APN_PASS=YOUR_PASSWORD
APN_AUTH=YOUR_AUTH_PROTOCOL
PROXY=yes

```

3. If necessary, enter the SIM pin:

```
$ mbimcli -d /dev/cdc-wdm0 -p --enter-pin=1234
```

4. Start the MBIM data connection:

```
$ mbim-network --profile=./mbim-network.conf /dev/cdc-wdm0
start
```

5. Request an IP to the DHCP server:

```
$ udhcpc -q -f -i wwan0
```

The interface must now have an IP address assigned and be able to connect to the Internet:

```
$ ping -I wwan0 google.com
```

```

PING google.com (142.250.184.14) from 31.22.237.161 wwan0: 56(84)
  bytes of data.
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=1
  ttl=113 time=33.5 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=2
  ttl=113 time=27.6 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=3
  ttl=113 time=36.1 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=4
  ttl=113 time=28.7 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=5
  ttl=113 time=28.5 ms

```

6.2.3.2 QMI

The EM06 is an LTE module with an M.2 interface and a 3042 form-factor, developed by Quectel (see Figure 6.9). It can use both MBIM and QMI as the communication protocol but uses QMI by default.



Figure 6.9: EM06-E Cellular module [51]

When connected to the system, we can see from the Kernel's messages that the modem is correctly detected and the QMI interface is mapped to the device `/dev/cdc-wdm0` and to the `wwan0` network interface:

```

$ dmesg
(...)

usb 1-1.3.3: new high-speed USB device number 8 using xhci-hcd
usb 1-1.3.3: New USB device found, idVendor=2c7c, idProduct=0306,
  bcdDevice= 3.10

```

```
usb 1-1.3.3: New USB device strings: Mfr=1, Product=2, SerialNumber
=3
usb 1-1.3.3: Product: EM06-E
usb 1-1.3.3: Manufacturer: Quectel
usb 1-1.3.3: SerialNumber: 0123456789ABCDEF

qmi_wwan 1-1.3.3:1.4: cdc-wdm0: USB WDM device
qmi_wwan 1-1.3.3:1.4 wwan0: register 'qmi_wwan' at usb-fe9c0000.xhci
-1.3.3, WWAN/QMI device, c6:f2:0e:f7:1d:46
```

However, while the device shows up as a network interface, the modem isn't able to reach the Internet. To achieve it, it is necessary to set up the device configurations, using a tool called `libqmi`. The process is extremely similar to the MBIM setup.

1. Verify if the device is recognized:

```
$ qmicli -d /dev/cdc-wdm0 -p --dms-get-operating-mode
```

2. Create a new configuration file (`qmi-network.conf`) with the following information:

```
APN=YOUR_APN
APN_USER=YOUR_USER
APN_PASS=YOUR_PASSWORD
PROXY=yes
```

3. If necessary, enter the SIM pin:

```
$ qmicli -d /dev/cdc-wdm0 -p ---dms-uim-verify-pin=PIN,1234
```

4. Start the QMI data connection:

```
$ qmi-network --profile=./qmi-network.conf /dev/cdc-wdm0 start
```

5. Request an IP to the DHCP server:

```
$ udhcpc -q -f -i wwan0
```

The interface must now have an IP address assigned and be able to connect to the Internet:

```
$ ping -I wwan0 google.com

PING google.com (142.250.185.14) from 95.69.56.26 wwan0: 56(84)
bytes of data.
64 bytes from mad41s11-in-f14.1e100.net (142.250.185.14): icmp_seq=1
ttl=56 time=102 ms
64 bytes from mad41s11-in-f14.1e100.net (142.250.185.14): icmp_seq=2
ttl=56 time=31.6 ms
64 bytes from mad41s11-in-f14.1e100.net (142.250.185.14): icmp_seq=3
ttl=56 time=33.2 ms
```

```
64 bytes from mad41s11-in-f14.1e100.net (142.250.185.14): icmp_seq=4
    ttl=56 time=32.2 ms
64 bytes from mad41s11-in-f14.1e100.net (142.250.185.14): icmp_seq=5
    ttl=56 time=31.2 ms
```

6.2.4 GPS module

This module uses only one USB interface, which means that it is a plug-and-play device. When connected to the system, the Kernel outputs the following messages:

```
$ dmesg
(...)

usb 1-1.3.1.3: new full-speed USB device number 6 using xhci-hcd
usb 1-1.3.1.3: New USB device found, idVendor=1546, idProduct=01a8,
    bcdDevice= 3.01
usb 1-1.3.1.3: New USB device strings: Mfr=1, Product=2,
    SerialNumber=0
usb 1-1.3.1.3: Product: u-blox GNSS receiver
usb 1-1.3.1.3: Manufacturer: u-blox AG - www.u-blox.com
cdc_acm 1-1.3.1.3:1.0: ttyACM0: USB ACM device
usbcore: registered new interface driver cdc_acm
cdc_acm: USB Abstract Control Model driver for USB modems and ISDN
    adapters
```

The system creates a virtual serial port (ttyACM0) which is the communication interface to the GPS modem. Reading the serial port outputs the following messages:

```
$ screen /dev/ttyACM0 115200

$GNGSA,A,3,01,07,21,10,27,16,,,,,,,,,1.68,0.84,1.45*19
$GNGSA,A,3,75,85,76,74,84,,,,,,,,,1.68,0.84,1.45*1E
$GPGSV,3,1,12,01,27,251,33,07,12,275,25,08,68,328,31,10,43,060,24*72
$GPGSV,3,2,12,14,01,325,,16,29,165,24,21,57,270,28,23,17,041,*7F
$GPGSV,3,3,12,26,03,160,,27,60,089,20,30,08,302,19,32,08,110,21*72
$GLGSV,3,1,09,68,04,016,,69,31,058,,70,27,130,,74,10,193,19*62
$GLGSV,3,2,09,75,42,241,30,76,31,314,31,84,26,060,09,85,57,340,19*6D
$GLGSV,3,3,09,86,27,285,*58
$GNGLL,4110.71609,N,00835.76850,W,114408.00,A,A*62
```

From the \$GNGSA messages, we can see satellite has currently two active satellites. The \$GPGSV and \$GLGSV messages show the satellites from GPS and GLONASS respectively that the receiver can view, even though some of them aren't active at the moment. Finally, the \$GNGLL

message reports the current latitude and longitude, which is $41^{\circ}10.71609'$ N $8^{\circ}35.76850'$ W. Figure 6.10 shows the position in the map, which is near the LSTS facilities as expected.



Figure 6.10: Position reported by the GPS module

These messages seem to show that the system is working correctly. As such, all of the requirements in Table 4.2a were met.

6.2.5 Iridium module

As this module uses the UART interface, it is first necessary to set up the UART multiplexer, as shown in Section 6.1.4. For demonstration purposes, this module was connected to the first channel of the first B row, which is assigned to the `ttymax0` device.

Following the command specification in [52], we can see that the module is working and responding to requests.

```
$ picocom --omap lfcr --baud 19200 /dev/ttymax0
> AT
OK
> AT+CGMI
Iridium
> AT+CGMM
IRIDIUM 9600 Family SBD Transceiver
```

In order to properly validate the transmission of messages, it is required to purchase line rental and credits. Unfortunately, this wasn't achievable but would be the next logical validation step. Other than that, all of the requirements in Table 4.2b were met.

6.2.6 Serial module

In contrast to the UART multiplexer used in the Main Board, this module uses USB as the communication protocol to the processor, instead of the SPI bus. This means that the chip is automatically detected by the operating system, without the need to create a device tree description. When connected to the system, the Kernel outputs the following messages:

```
$ dmesg
(...)

usb 1-1.3.1.3: new full-speed USB device number 11 using xhci-hcd
usb 1-1.3.1.3: New USB device found, idVendor=04e2, idProduct=1424,
    bcdDevice= 0.02
usb 1-1.3.1.3: New USB device strings: Mfr=1, Product=2,
    SerialNumber=3
usb 1-1.3.1.3: Product: Exar USB UART
usb 1-1.3.1.3: Manufacturer: Exar Corp.
usb 1-1.3.1.3: SerialNumber: J413856982
cdc_acm 1-1.3.1.3:1.0: ttyACM0: USB ACM device
cdc_acm 1-1.3.1.3:1.2: ttyACM1: USB ACM device
cdc_acm 1-1.3.1.3:1.4: ttyACM2: USB ACM device
cdc_acm 1-1.3.1.3:1.6: ttyACM3: USB ACM device
```

As expected, it also creates four new devices, one for each UART channel:

```
$ ls /dev/ttyACM*
/dev/ttyACM0 /dev/ttyACM1 /dev/ttyACM2 /dev/ttyACM3
```

The module LEDs also correctly signal when data is being transmitted and received, per channel. All of the requirements in Table 4.2c were met.

6.3 Discussion of results

The proposed hardware design brings not only new functionality that couldn't be achieved before, but also improves some of the components already used in maritime systems. The traditional construction of the current vehicles offered by LSTS is already detailed in Section 2.1.1. In this subsection we compare both approaches.

The first obvious feature offered by BALU is the possibility to use PCIe devices, which isn't achievable in the current traditional design. This unlocks new high-speed devices such as Solid-State Drives and high-bandwidth Wi-Fi modems, that could prove to be quite beneficial for maritime applications, mainly in the storage and transmission of logs generated by the sensors on-board. Another interesting application of the PCIe interface is the usage of Machine Learning accelerators for Computer Vision, although the current navigation software is not able to take advantage of that.

Secondly, the peripherals of the traditional system is constrained by the capabilities of the Geode stack. The processor features only two RS232 ports, two USB 2.0 ports, and the Ethernet interface is limited to 10/100 Mbps operations. Nowadays, this is extremely limited and the off-the-shelf expansion offers is also very scarce. This leads to the need of using an expensive RS232 8-channel expansion board, the FSCC-104. The required functionality of the system, such as wireless connectivity and storage is implemented externally, and connected via cables. In comparison, the BALU design's module system is extremely versatile in the expansion of the processing unit capabilities. The M.2 modules are becoming the standard expansion card form-factor, and their market is becoming very competitive, with great offers from various manufacturers. Not only that, but the design of custom modules tailored to a specific application is easy and cheap. The three developed expansion card are an example of the ease-of-integration of such modules. These factors contribute to the high modularity and scalability of the design, achieved by adding and removing modules to the system.

Finally, the BALU design replaces some of the previously used external modules connected with cables with M.2 alternatives, heavily decreasing the amount of cabling and volume. The reasoning behind the decrease in volume is related to the improvements in manufacturing of these modules, as some of the currently used products are extremely dated. In addition to that, the M.2 interface is also the standard interface for laptop and desktop computers, which makes the demand for M.2 modules, with smaller form-factors, extremely high. This results in smaller, cheaper and reliable offers in comparison to other form-factors. It is estimated that the BALU Main Board together with the required expansion modules can achieve the same functionality as the current hardware design with a third of the total volume.

Chapter 7

Conclusion

With the development of the project, a deep understanding of how the hardware in maritime systems is developed at LSTS was acquired. With that knowledge, we could find some out-of-date aspects that could be modernized to increase their systems' reliability and manufacturability.

The project specification was designed with the LSTS team according to their most essential features. Then, it went through an iterative design and manufacturing process, which featured two Main Board revisions. Each circuit board followed an extensive verification and validation process to assess if its requirements were met and if its real-world functionality and performance were as expected.

The final product - the BALU architecture - is an elegant solution to the challenges in the hardware design of maritime systems. Traditionally, the system would be composed of a CPU (the Geode stack), and different external modules would be connected through cables. However, with this design, the expansion card design leads to an almost cable-free implementation by replacing these external modules with an M.2 alternative, increasing reliability and maintainability. It also unlocks new possibilities in future maritime systems designs because of its small form factor.

Separating the system's hardware components into standard expansion modules allows for future revisions and implementation of new features with minimal effort. The M.2 connector enables the integration of commercial modules effortlessly, which previously could not be used without designing a custom solution. Furthermore, the PCIe interface unlocks new possibilities in future high-speed applications.

In summary, the project's main achievements are:

- An almost cable free implementation, by replacing external modules with an M.2 alternative;
- A smaller form factor;
- The possibility of using high-performance modules, with the PCIe interface;
- Increased reliability and maintainability, because of the modularity of the system;

For future work, the most essential aspect would be to further test and validate the design under typical operation. The system has all the necessary features to be integrated into a vehicle. As a matter of fact, in the future, this architecture will be integrated into a new LAUV at LSTS, replacing the existing out-of-date Geode stack. It unlocks the ability to use modern CPUs with better power consumption and processing power. The existing PC/104 RS232 expansion module can be replaced by only two RS232 M.2 cards, which have a combined area equal to one-fifth of the original.

On the design of this work's circuit boards, there are also a couple of aspects that would be interesting to experiment with: (1) changing the peripherals logic-level to 1.8 V could prove to be beneficial by decreasing power consumption and reducing the number of required voltage translators; (2) the integration of another System on a Module would be the next step to validate the architecture's CPU-Independence.

Appendix A

Questionnaire

Dissertation Questionnaire

Theme: In order to reduce development time and the amount of resources needed for systems to be developed in the future, it is necessary to create a new hardware/software solution that is scalable, modular, robust, energy efficient and processor independent. .

The offer of microprocessors on the market is somewhat limited, which makes it difficult to find a solution exactly according to the requirements of a project. To meet all these requirements, it is often necessary to resort to products that are too expensive and inefficient, which end up having a negative impact on some aspect.

Ex: A microprocessor with 10 serial ports will probably be considered "top of the range", with a greater processing capacity that will not be used and consequently greater energy consumption.

The idea is to create a PCB board that can use a low-cost processor and processing power, and at the same time expand its capabilities using expansion modules in order to build a system exactly as needed. At the same time, integrate this architecture and modules into GLUED and DUNE.

1.

What processors (SoM) does it make sense to consider in a first approach? *

Check all that apply.

☐ Raspberry Pi Compute Module 4

☐ Nvidia Jetson

☐ NXP i.MX 6/8

☐ Other: _____

2.

Will it be useful to be able to have multiple processors on the board? For example, a primary processor with high processing capacity and a secondary processor with better energy efficiency. If yes, how many processors?

Mark only one oval.

☐ Não

☐ 2

☐ 3

☐ Other: _____

Expansion Modules

Expansion modules are used to augment processor capabilities. The interface used for each module will be M.2.

3.

Which expansion modules would be useful?

Check all that apply.

- ☐ Custom (ex: a microcontroller used for measuring battery charge)
- ☐ USB Hub (increase the number of USB ports)
- ☐ UART Hub (increase the number of UART ports - RS232 ou RS485)
- ☐ CAN (creates a CAN interface)
- ☐ SATA (creates SATA ports, used for storage)
- ☐ Micro SD
- ☐ Modems (WiFi, Bluetooth, GPS, LoRa, etc.)
- ☐ Tensor Processing Unit (Google M.2 Accelerator)
- ☐ Other: _____

4.

What features must exist on the Main board independently of the expansion modules?

Check all that apply.

- ☐ USB
- ☐ UART
- ☐ SPI
- ☐ I2C
- ☐ PCIe
- ☐ Ethernet
- ☐ CAN
- ☐ Micro SD
- ☐ SATA
- ☐ Other: _____

5.

What features are most important in expansion modules?

Check all that apply.

- ☐ Power management system (allows you to turn off and on each module individually)
- ☐ Hotswap (to be able to remove and remove modules while the system is running)
- ☐ Debug interface (check that everything is OK with the device, diagnostics)
- ☐ Other: _____

6.

How many PCIe ports do you find useful on the card?

Mark only one oval.

- ☐ 0
- ☐ 1
- ☐ 2
- ☐ 3
- ☐ Scalable (you can expand the number of ports)

Software

The idea is to integrate all this framework in GLUED and DUNE. The architectures of the new processors must be created as well as the addition of drivers for the different expansion modules

7.

What features are important to have in DUNE?

Check all that apply.

- ☐ List the modules present on the board
- ☐ Power off/on each module
- ☐ Receive a diagnosis of each module
- ☐ Hotswap events (fires code when a module is inserted/removed)
- ☐ Other: _____

PCB Parametric generation

One idea to be able to include the project in different form-factors would be to be able to reconfigure the organization of the PCB. In order to not have to re-design the PCB whenever you want to change the size of the board, a solution would be to create a board generation program based on some parameters (number of modules needed, board design, ports needed, etc. .) using the Kicad API.

8.

Would it be interesting?

Mark only one oval.

- ☐ Yes
- ☐ No

9.

If so, which parameters would be more important?

Check all that apply.

☐ Number of module slots

☐ PCB dimensions

☐ Number of PCIe ports

☐ Other: _____

This content is neither created nor endorsed by Google.

Google Forms

References

- [1] LSTS. Laboratório de Sistemas e Tecnologia Subaquática [online]. URL: <https://lsts.feup.pt/> [last accessed Jan. 08, 2022].
- [2] P.V. Dudhe, N.V. Kadam, R. M. Hushangabade, and M. S. Deshmukh. Internet of Things (IOT: An overview and its applications. IEEE, 8 2017. doi:10.1109/ICECDS.2017.8389935.
- [3] José Luis Álvarez, Juan Daniel Mozo, Eladio Durán, and Antonio Guerrieri. Analysis of Single Board Architectures Integrating Sensors Technologies. 2021. URL: <https://doi.org/10.3390/s21186303>, doi:10.3390/s21186303.
- [4] Raspberry Pi. Raspberry Pi | Compute Module 4 [online]. URL: <https://www.raspberrypi.com/products/compute-module-4> [last accessed Feb. 14, 2022].
- [5] NVIDIA. NVIDIA | Jetson [online]. URL: <https://www.nvidia.com/en-us/autonomous-machines/jetson-store/> [last accessed Feb. 12, 2022].
- [6] L3Harris. Iver3 Standard System UUV | L3Harris™ Fast. Forward [online]. URL: <https://www.l3harris.com/all-capabilities/iver3-standard-system-uuv> [last accessed Jun. 27, 2022].
- [7] PCI Express M.2 Specification. Technical report, PCISIG, 2013. Revision 1.0. URL: www.pcisig.com.
- [8] PCI-SIG. Welcome to PCI-SIG [online]. URL: <https://pcisig.com/> [last accessed Jun. 14, 2022].
- [9] Vishal Anand, Varsha Singh, and Vinay Kumar Ladwal. Study on PCB Designing Problems and their Solutions. *2019 International Conference on Power Electronics, Control and Automation, ICPECA 2019 - Proceedings*, 2019-November, 11 2019. doi:10.1109/ICPECA47973.2019.8975402.
- [10] High-Speed Layout Guidelines Application Report High-Speed Layout Guidelines. Technical Report SCAA082A, Texas Instruments, 2006. URL: <https://www.ti.com/lit/an/scaa082a/scaa082a.pdf>.
- [11] High Speed Layout Design Guidelines. Technical Report AN2536, Freescale Semiconductor Inc, 2006. URL: <https://www.nxp.com/docs/en/application-note/AN2536.pdf>.
- [12] High-Speed Design Guidelines High-Speed Design Guidelines Application Note. Technical Report G53065 AN-56, Microsemi Corporation, 2003. URL: <https://ww1.microchip.com/downloads/en/Appnotes/G53065.pdf>.

- [13] Layout Guidelines of PCIe ® Gen 4.0 Application With the TMUXHS4412 Multiplexer. Technical Report SLAAE45, Texas Instruments, 2021. URL: <https://www.ti.com/lit/an/slaae45/slaae45.pdf>.
- [14] Universal Serial Bus Specification. Technical report, USB 2.0 Promoter Group, 2000. Revision 2.0. URL: <https://www.usb.org/document-library/usb-20-specification>.
- [15] Universal Serial Bus 3.2 Specification. Technical report, USB 3.0 Promoter Group, 2022. Revision 1.1. URL: <https://www.usb.org/document-library/usb-32-revision-11-june-2022>.
- [16] PCI Express ® Base Specification. Technical report, PCISIG, 2006. Revision 2.0. URL: www.pcisig.com.
- [17] PCI Express ® Base Specification. Technical report, PCISIG, 2014. Revision 4.0. URL: www.pcisig.com.
- [18] 9 Pin Male D-Sub Connector [online]. URL: <https://www.circuitspecialists.com/de-09p.html> [last accessed Jun. 29, 2022].
- [19] USB Type A Female Dip Type Connector [online]. URL: <https://www.pimfg.com/product-detail/USB-ASDA30> [last accessed Jun. 29, 2022].
- [20] PCIe Gen 3 Card Edge Connectors - Amphenol FCI [online]. URL: <https://pt.mouser.com/new/amphenol/amphenol-fci-gen3-pcie/> [last accessed Jun. 29, 2022].
- [21] PCI Express ® Mini Card Electromechanical Specification. Technical report, PCISIG, 2007. Revision 1.2. URL: www.pcisig.com.
- [22] JAE 52 Way Right Angle Mini PCIe, PCI Memory Card Connector With Solder Termination [online]. URL: <https://uk.rs-online.com/web/p/memory-card-connectors/0435689> [last accessed Jun. 29, 2022].
- [23] M.2 Connectors - Amphenol Commercial [online]. URL: <https://pt.mouser.com/new/amphenol/amphenol-m2-connectors/> [last accessed Jun. 29, 2022].
- [24] PC/104 Single Board Computer with 500MHz Geode LX800 CPU [online]. URL: <https://www.diamondsystems.com/products/rhodeus> [last accessed Jun. 22, 2022].
- [25] Raspberry Pi 4 Computer Model B. Technical report, Raspberry Pi, 2019. URL: <https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-4-Product-Brief.pdf>.
- [26] NVIDIA. NVIDIA Jetson Nano Developer Kit [online]. URL: <https://developer.nvidia.com/embedded/jetson-nano-developer-kit> [last accessed Jun. 22, 2022].
- [27] Raspberry Pi. Raspberry Pi Compute Module 4 IO Board. Technical report, Raspberry Pi, 2020. URL: <https://datasheets.raspberrypi.com/cm4io/cm4io-product-brief.pdf>.

- [28] Turing Machines Inc. Raspberry Pi cluster on a mini ITX board [online]. URL: <https://turingpi.com/> [last accessed Feb. 15, 2022].
- [29] ALFTEL Systems Ltd. ALFTEL Systems Ltd [online]. URL: <https://www.alfotel.com/> [last accessed Jun. 27, 2022].
- [30] Raspberry Pi. Raspberry Pi Documentation - Compute Module Hardware [online]. URL: <https://www.raspberrypi.com/documentation/computers/compute-module.html> [last accessed Jun. 17, 2022].
- [31] Universal Serial Bus Type-C Cable and Connector Specification, institution = USB 3.0 Promoter Group. Technical report, 2019. Revision 2.0. URL: <https://www.usb.org/sites/default/files/USBType-CSpecR2.0-August2019.pdf>.
- [32] USB 2.0 Hi-Speed Hub Controller. Technical Report USB251xB/xBi, Microchip. URL: <http://ww1.microchip.com/downloads/en/devicedoc/00001692c.pdf>.
- [33] Schematic Checklist for the USB2514B, 36-pin QFN Package. Technical Report AN26.21, Microchip. URL: <https://www.microchip.com/en-us/application-notes/an2621>.
- [34] Quad Serial UART with 128-Word FIFOs. Technical Report MAX14830, Maxim Integrated. URL: <https://datasheets.maximintegrated.com/en/ds/MAX14830.pdf>.
- [35] Distrelec. Microchip Ethernet Controller [online]. URL: <https://www.distrelec.biz/en/ethernet-controller-parallel-tqfp-140ma-microchip-lan91c111-nu/p/30046541> [last accessed Jun. 29, 2022].
- [36] DigiKey. VSC8572XKS-05 Microchip Technology | Integrated Circuits (ICs) | DigiKey [online]. URL: <https://www.digikey.pt/en/products/detail/VSC8572XKS-05/VSC8572XKS-05-ND/9005080> [last accessed Jun. 29, 2022].
- [37] PCIe 2.0 Packet Switch 4-Port. Technical Report PI7C9X2G404SV, Diodes Incorporated, 2020. URL: <https://www.diodes.com/assets/Datasheets/PI7C9X2G404SV.pdf>.
- [38] Pericom Schematic Checklist. Technical report, Diodes Incorporated, 2020.
- [39] NEO-8Q / NEO-M8: u-blox 8 / M8 GNSS modules Hardware integration manual. (UBX-15029985 - R09). URL: https://content.u-blox.com/sites/default/files/NEO-8Q-NEO-M8-FW3_HIM_UBX-15029985.pdf.
- [40] Michael Bailey. General Layout Guidelines for RF and Mixed-Signal PCBs. 2011.
- [41] Iridium. Iridium Satellite Communications | Your World. Our Network. [online]. URL: <https://www.iridium.com/> [last accessed Jun. 15, 2022].
- [42] Iridium 9603: SBD Transceiver Developer's Guide. Technical report, Iridium. URL: https://www.cls-telemetry.com/wp-content/uploads/2018/12/Iridium-9603-9603N_Developers-Guide.pdf.
- [43] General-Purpose, Gated-Oscillator-Based DC-DC Boost Converter datasheet. Technical Report LM2623, Texas Instruments. URL: <https://www.ti.com/lit/ds/symlink/lm2623.pdf>.

- [44] Enhanced 4-Ch Full-Speed USB UART General Description. Technical Report XR21B1424, MaxLinear. URL: <http://www.maxlinear.com/>.
- [45] One Microamp Supply-Current, +3V to +5.5V, 250kbps, RS-232 Transmitters/Receivers. Technical Report ICL32XX, Renesas. URL: <http://www.renesas.com/>.
- [46] iPerf - The TCP, UDP and SCTP network bandwidth measurement tool [online]. URL: <https://iperf.fr/> [last accessed Jun. 19, 2022].
- [47] Introduction to Device Trees - Application Note. Technical Report AN5125, Freescale Semiconductor, 2015. URL: <https://www.nxp.com/docs/en/application-note/AN5125.pdf>.
- [48] PC-MAG. Toshiba RC100 NVMe SSD Review [online]. URL: <https://www.pcmag.com/reviews/toshiba-rc100-nvme-ssd> [last accessed Jun. 29, 2022].
- [49] Intel. Intel WiFi 6 AX200 Gig Product Specifications [online]. URL: <https://ark.intel.com/content/www/us/en/ark/products/189347/intel-wifi-6-ax200-gig.html> [last accessed Jun. 29, 2022].
- [50] 4G-LTE Mall. Huawei ME906s-158 4G LTE 75-pin M.2/NGFF Module for sale [online]. URL: <https://www.4gltemall.com/huawei-me906s-158-4g-lte-m-2-module.html> [last accessed Jun. 29, 2022].
- [51] Quectel. LTE-A EM06 series [online]. URL: <https://www.quectel.com/product/lte-a-em06-series> [last accessed Jun. 29, 2022].
- [52] Iridium Proprietary ISU AT Command Reference. Technical Report MAN0009, Iridium.