

POLITECNICO DI TORINO

Master's Degree in Electronic Engineering



Politecnico di Torino

Master's Degree Thesis

Design of IoT node for smart agriculture

Supervisors

Prof. Danilo DEMARCHI

Ph.D. Umberto GARLANDO

Dott. Mattia BAREZZI

Candidate

Simone CONIGLIONE

DECEMBER 2023

*A chi mi ha sempre sostenuto,
supportandomi e sopportandomi,
durante ogni istante del mio
percorso.*

*A Catia, sole e luna nella mia
vita, che ha condiviso con me i
momenti più luminosi e più bui,
che mi ha donato il respiro quando
credevo di affogare.*

*Alla mia famiglia che,
sostenendomi e supportando le
mie scelte, è stata e sempre sarà
un pilastro fondamentale della mia
vita.*

A Nonno Emanuele, mi manchi.

Summary

This thesis is focused in the design of a long-range low-power hardware platform based on the STMicroelectronics microcontroller STM32WL55CC that is compliant with different kinds of LPWAN communication protocols such as LoRa, (G)FSK, (G)MSK, and BPSK modulation. The expected output is a standard module that can be used as a central core for LPWAN application, simplifying the design process of different LPWAN electronic systems. In particular, a configuration file is developed to configure the device as a LoRa end node. The prototype, called LoRaTo, is part of the Piedmont regional project WAPPFRUIT and is designed to improve the current version of an existing electronic system.

The first chapter is an **introduction** that gives a general view of the project WAPPFRUIT and the reasons behind the birth of the project.

The second chapter explains the **background**, which is the basis of this thesis work, describing the starting point of the proposed design.

The **proposed design** chapter is divided into different parts: starting from the system specification, the desired electrical schematic is derived, including the following main blocks:

- Power supply;
- RF path;
- Oscillators;
- Programming and reset interfaces.

Once the electrical schematic is obtained, the physical design phase starts: the PCB design is realized considering the interconnections between the components and needed design constraints to get good performances and reducing noise emission and susceptibility. Finally, a pre-defined firmware configuration is provided to simplify the design process for future users.

After the prototype manufacturing, the fourth chapter describes the performed **tests** to verify the functionalities of the board and find possible problems.

The last chapter summarises the **conclusion** at the end of this thesis work, explaining the main obtained results and providing hints for future thesis works.

Acknowledgements

Con questa tesi finisce una tappa importante del viaggio della mia vita, iniziato nella mia terra natale, Catania, e giunto ora a Torino. Un viaggio lungo e tortuoso, che mi ha però reso l'uomo che sono oggi.

A Catia, che mi è stata accanto nel periodo più difficile e intenso della mia vita, come nessuno ha mai fatto, restando con me anche quando io stesso sarei scappato via, lontano da tutti. Un semplice grazie non può bastare, ti sarò per sempre riconoscente per tutto quello che hai fatto per me e ti starò sempre accanto così come tu hai fatto con me.

A papà Salvo, per tutti gli insegnamenti e per aver creduto in me. A mamma Luisa, per aver protetto la nostra famiglia. Ad Andrea, per sempre mio fratello. Mi siete stati vicini gettando le fondamenta su cui ho costruito, un mattone per volta, la mia strada.

Agli “amici di giù”, lontani ma comunque vicini. A Daniele, cugino e fratello fin da quando ero bambino, sei riuscito a capirmi e a sostenermi quando più ne avevo bisogno. A Dario e Nicoletta, per tutti i momenti felici che abbiamo passato insieme. Un grazie ai miei cari amici Pippo, Angelo, Andrea, Federico, Alisea e Clara.

Agli “amici di su”, che in realtà sono in parte “amici di giù”. A Nicola, Federica e Silvia, che hanno alleggerito l'impatto del trasferimento nella grande città, so che su di voi potrò sempre contare. A Paolo, Luca e Davide, abbiamo affrontato parte di questo percorso insieme, siete degli amici ancor prima che colleghi e vi ringrazio per il supporto che mi avete dato durante questi anni al Politecnico.

Desidero ringraziare il Prof. Demarchi per avermi dato l'opportunità di affrontare quest'ultima sfida insieme al suo gruppo e i miei correlatori Mattia e Umberto per l'aiuto e il supporto che hanno dato, contribuendo ad ampliare la mia conoscenza e allo sviluppo di questo progetto di tesi.

Table of Contents

List of Tables	IX
List of Figures	X
Acronyms	XIII
1 Introduction	1
2 Background	3
2.1 The starting point: WAPPFRUIT electronic systems	3
2.2 STM32WL series microcontroller	4
2.2.1 Sub-GHz characteristics	4
2.3 Altium Designer	5
2.3.1 Workspaces	5
3 Proposed design	10
3.1 Microcontroller selection	11
3.2 Electrical schematic	12
3.2.1 Voltage supply	12
3.2.2 RF path	14
3.2.3 RF power supply	19
3.2.4 Oscillators	20
3.2.5 Programming interface	23
3.2.6 Complete schematic	23
3.3 Hardware design	26
3.3.1 PCB stackup	26
3.3.2 RF layout design rules	28
3.3.3 Oscillators design rules	31
3.3.4 PCB manufacturer design rules	33
3.4 PCB design	34
3.4.1 PCB area	34

3.4.2	PCB layout	34
3.4.3	PCB Validation and Manufacturing files	39
3.5	Firmware configuration	47
3.5.1	LoRa configuration on CubeIde	47
3.5.2	External files needed	52
3.5.3	Test code	58
4	Prototype and tests	65
4.1	Software tests	66
4.1.1	Programming interface	66
4.1.2	LoRa class-A test	66
4.2	Output power measurements	67
4.3	Qualitative LoRa coverage test	69
4.4	Future work	70
5	Conclusion and future prospectives	71

List of Tables

3.1	Applied Design Rules from [22].	33
3.2	BOM.	41
4.1	Output power measurements.	68
4.2	Qualitative downlink test results.	70

List of Figures

1.1	Countries footprint from [2].	2
2.1	Sub-GHz radio system block diagram from [6].	4
2.2	Schematic Library Workspace.	6
2.3	PCB Library Workspace.	7
2.4	3D footprint example - 0402 capacitor.	7
2.5	3D footprint with component model - 0402 capacitor.	7
2.6	Schematic workspace.	8
2.7	PCB workspace.	9
2.8	Output Job File workspace.	9
3.1	Draft block diagram.	10
3.2	UFBGA73 side view.	11
3.3	UFBGA73 bottom view.	11
3.4	UFQPFN48 side view.	12
3.5	UFQPFN48 bottom view.	12
3.6	Internal Power Supply scheme from [6].	13
3.7	Power supply schematic.	14
3.8	U.FL connector, form [8].	15
3.9	SMA connector.	15
3.10	Antenna and π net.	16
3.11	RF switch truth table from [10].	17
3.12	RF switch circuit.	17
3.13	Filtering and matching stage from [9].	18
3.14	Balun types.	18
3.15	RF balun circuit.	19
3.16	RF power supply configuration.	20
3.17	TXCO electric schematic.	21
3.18	LSE electrical schematic.	22
3.19	JTAG connector.	23
3.20	GPIOs, JTAG, RESET, and ESD protection electrical schematic.	24

3.21	Complete electrical schematic.	25
3.22	Characterized stackup from [11].	27
3.23	Altium Designer Stackup manager.	27
3.24	Appcad impedance calculator.	28
3.25	Recommanded layout from [11].	29
3.26	Recommanded inductor layout from [21].	29
3.27	Recommanded capacitor layout from [21].	30
3.28	Pad width recommendation from [21].	30
3.29	Oscillator guard ring example from [12].	32
3.30	Oscillator ground plane example from [12].	32
3.31	Board shape definition.	34
3.32	Top layer layout.	35
3.33	Ground plane.	36
3.34	Power plane.	37
3.35	Bottom layer	38
3.36	Component information from schematic.	39
3.37	STM32CubeMX ioc configuration.	47
3.38	STM32CubeMX LoRa config. recommendation.	51
3.39	STM32CubeMX clock tree.	51
4.1	LoRaTo 3D model.	65
4.2	LoRaTo prototype.	65
4.3	STM32CubeProgrammer.	66
4.4	The Things Network live data interface.	67
4.5	TX and RX real time current profile.	67
4.6	LoRaTo measured output power.	68
4.7	Nucleo-board measured output power.	68
4.8	Stops of qualitative coverage test.	69

Acronyms

PCB

Printed Circuit Board

TTN

The Things Network

LPWAN

Low-Power Wide-Area Network

LoRa

Long Range

LoRaWAN

Long-Range Wide-Area Network

RF

Radio Frequency

LSE

Low Speed External clock

HSE

High Speed External clock

TCXO

Temperature Compensated Crystal Oscillator

Chapter 1

Introduction

Nowadays, humanity is dealing with several problems related to using and managing our planet's available resources: pollution, Earth Overshoot Day, and renewable resources are the keywords of the future. Earth overshoot day is a term born to indicate the day when the resources available and produced in a year ends before the end of the year itself, and this is a significant problem to deal with because if the earth's resources are not carefully managed there will be a day in which the humanity will be left without water, food and other essential resources. The Global Footprint Network [1] analyses and elaborates the data related to resource consumption, showing whether the footprint of a country is greater or lower than its biocapacity. As shown in fig. 1.1, many countries have a biocapacity deficit, and Italy is among them. Many steps are needed to reach a sustainable lifestyle, and proper management of the resources is one of them. When discussing waste of resources, it is also included the case in which a particular resource is used more than necessary. A simple case is water use in agricultural fields: with old techniques, water was delivered to the area based on the farmer's knowledge. Thanks to modern technology, it has been found that each type of plant has its specific need for water, which is only sometimes a considerable amount. Smart technologies help people in their daily lives, simplifying or even improving existing services. Moreover, they are also a powerful tool to deal with the world problems cited up to now: pollution and renewable resources are strictly related, but resource management is always at the basis of each problem. The work presented with this thesis is part of the regional project WAPPFRUIT [3] and deals with the issue of resource management, particularly water management in agricultural fields. As part of this project, an electronic system composed by [4] and [5] has been designed: its role is to realize an automatic drip irrigation system for orchard employing LoRa technology. One node [4] is in charge to collect soil data from soil industrial sensor, the other one [5] to drive the electrovalve connect to the dripper in the orchard.

This thesis aims to show the needed design process to make a standard PCB that

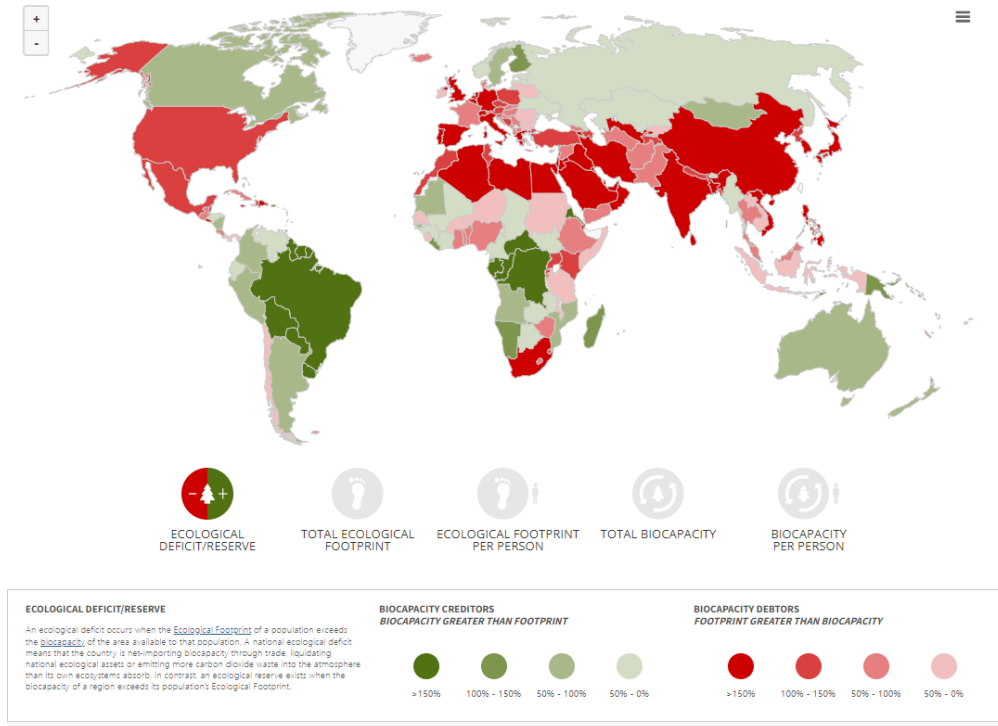


Figure 1.1: Countries footprint from [2].

can be used as the central core of different end nodes. The PCB is designed to include only the necessary circuit and components to enable the microcontroller to work and communicate through an adapted 50Ω antenna using LoRa technology. The expected output is a ready-to-use PCB that only needs to be programmed and could be used as core element of application-specific electronic systems for precision agriculture applications, simplifying and speeding up the PCB design process.

Chapter 2

Background

As already mentioned, the starting point of this thesis is an already existing electronic system [4],[5] that has to be improved, and the realization of a standard central core is one of the steps needed. Before starting to talk about the main work of this thesis, the preliminary required information is provided in this chapter.

2.1 The starting point: WAPPFRUIT electronic systems

The key concept under the design of the electronic system is the possibility to collect data from a wide area and process them to be able to manage the irrigation of agricultural fields properly: the system is composed of several elements called end nodes, which are the modules that collect data from the farm area, and a central element called gateway that manage all the data coming from the end nodes to make them available on a cloud server. In particular, two electronic boards have been developed: a soil moisture monitoring end node, called WappSen [4], and a module responsible for the actuation of solenoid valves, called WappAct [5]. In particular, the actual version of the WappAct end node is composed of a NUCLEO-STM32WL55 board and a custom shield that contains the part related to the power supply and the circuit needed to drive the solenoid valve: the goal is to obtain a smaller system with lower power consumption, and a fundamental step to do that is to replace the Nucleo board with a custom PCB that contains only the essential blocks for the target application.

2.2 STM32WL series microcontroller

The PCB designed in this thesis is realized around the STM32WL55 microcontroller. As stated in [6], the STM32WL55 family is based on two 32-bit Arm processors, respectively Arm Cortex-M4 and Arm Cortex-M0+, which are low-power processors for embedded applications, that guarantee good performances with low power consumption. The peculiarity of STM32WL55 microcontrollers is that they embed a subsystem dedicated to the sub-GHz communication that is interfaced internally with the system CPU through a SPI interface, as it is shown in fig. 2.1. The usage of this microcontroller simplifies the design process since the microcontroller manages the LoRa modulation, so it is only needed to configure the desired characteristics via software and implement the correspondent firmware.

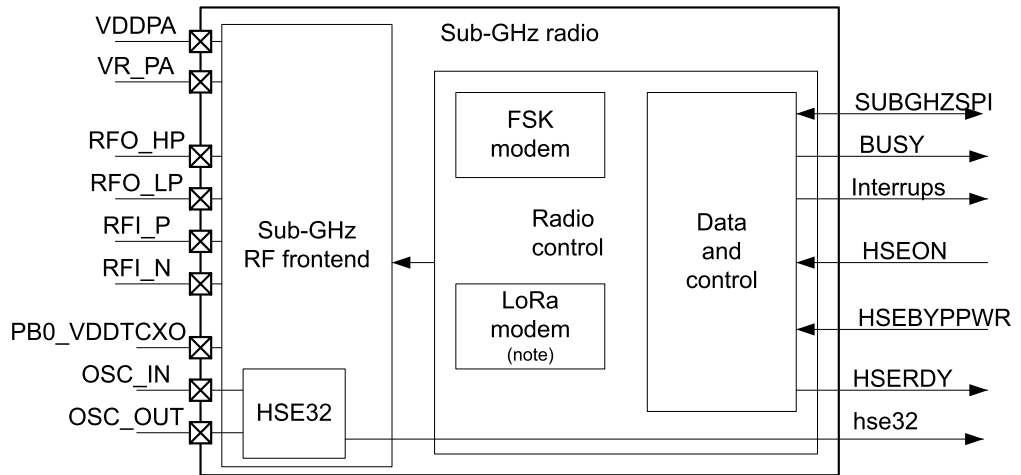


Figure 2.1: Sub-GHz radio system block diagram from [6].

2.2.1 Sub-GHz characteristics

Output Power As mentioned, the microcontroller implements a sub-GHz subsystem that complies with LoRa (and not only) communication protocol. The analog front-end transceiver can provide output signals up to $+15dBm$ through the RFO_LP output pin and up to $+22dBm$ through the RFO_HP output pin; the proposed design will use the high power output.

Frequency Range The radio system can continuously cover frequencies from 150 to 960 MHz. LoRa protocol in Europe works on 433 and 868 MHz carrier

frequencies.

2.3 Altium Designer

Altium Designer is the software EDA used to realize the physical design of the PCB. The design process is divided into different stages that are briefly described:

- Realisation of the **electrical schematics**: the complete schematic is described in one or more schematic sheets.
- Physical sizing of **component footprints**: each component in the schematic is associated with the physical dimensions imported in the PCB layout environment. This step is essential to evaluate the total PCB footprint.
- **Physical design of the PCB**: in this phase, the number of layers (and material) is defined, and the physical layout of each part of the PCB is performed, such as the routing of interconnection, the floor planning of the component, etc.
- **Design rule verification**: the PCB physical layout must respect some design rules, depending on both the manufacturer and other design rules.
- Generation of **manufacturing files** and **project documentation**.

2.3.1 Workspaces

Altium Designer has a dedicated workspace for almost each design step to simplify the design process. They are briefly described below for clarity.

Schematic library The schematic library workspace is shown in fig. 2.2 and contains all the components used in the project design. Importing a symbol or drawing it using the **toolbar** is possible. On the right side, two important sections are placed: in **properties**, it is possible to write all the component information that is going to be displayed in the schematic sheet; in **component link**, it is possible to find the available component from different suppliers, this is very useful because, as it will be discussed later, in the BOM the component information, such as the current price, are displayed automatically. Each component must contain the pin references to associate each pin with the correspondent pad in the **footprint**, which contains the physical dimension of the pads (and not only) that will be considered in the physical design phase.

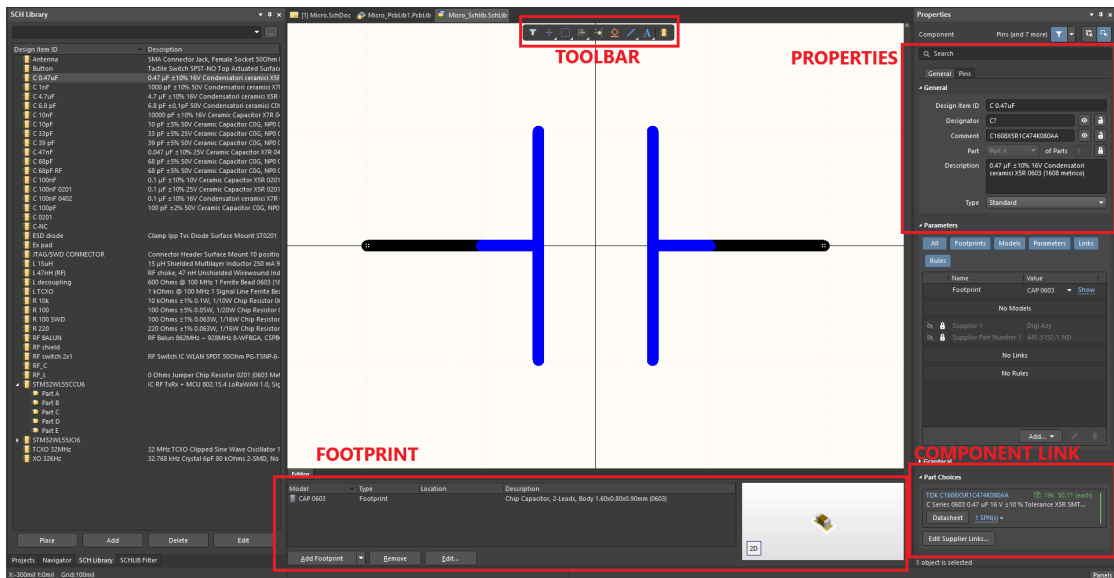


Figure 2.2: Schematic Library Workspace.

It is possible to divide a component into more blocks to simplify the electric schematic. For example, a microcontroller can be described as utilizing multiple blocks (one containing supply voltage pins, one for I/O, one for the RF stage, and so on) and drawing each type of pin in a specific block.

PCB library In fig. 2.3, the workspace, where the component footprints are designed, is shown. There is the possibility to import already existing footprints or to make them starting from zero. The footprint contains all the information related to the physical dimension of each layer of the component, which are:

- **contact pads**, associated with the pin of the schematic symbol
- **silk-screen printing**
- **assembly layer** (for PCB documentation)
- **solder mask layer**, which defines the area to be etched to expose the pads
- **designator**, which is the identifier of the component

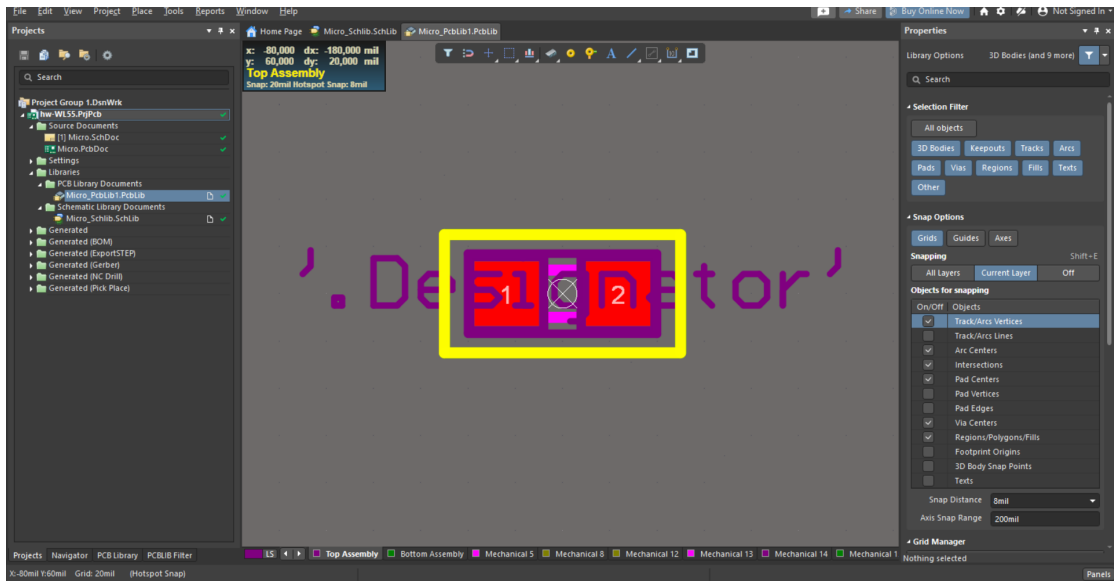


Figure 2.3: PCB Library Workspace.

in addition of them, it is possible to include a 3D model of the component. It is not always mandatory, but in some cases, such as when a metal shield is placed on the top of a specific portion of the PCB, it can be helpful to check the dimensions and see if some components are in contact with others. An example is reported in fig. 2.4 and fig. 2.5.

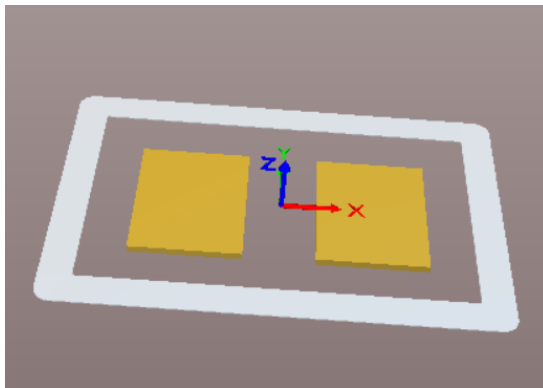


Figure 2.4: 3D footprint example - 0402 capacitor.

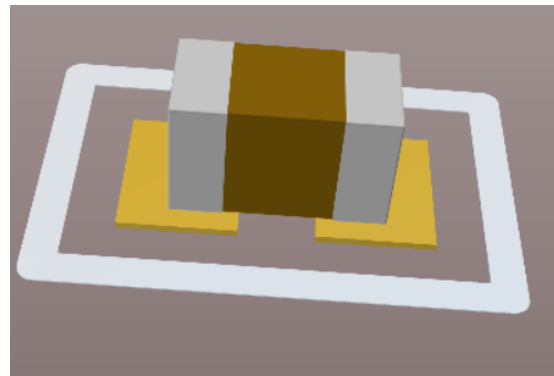


Figure 2.5: 3D footprint with component model - 0402 capacitor.

Schematic Sheet Once the schematic library is populated with the desired component, the electrical schematic is realized in the schematic workspace. As shown

in fig. 2.6 on the right side, all the components from the library are available and can be placed inside the schematic sheet. Then, by the toolbar, many instruments are available to realize interconnection, power supply and ground references, etc. Once the schematic is ready, if a change of components is needed, there are two possibilities: changes the parameter of the single component from the schematic sheet or changes the parameter from the schematic library and automatically updates all the corresponding components in the schematic sheet.

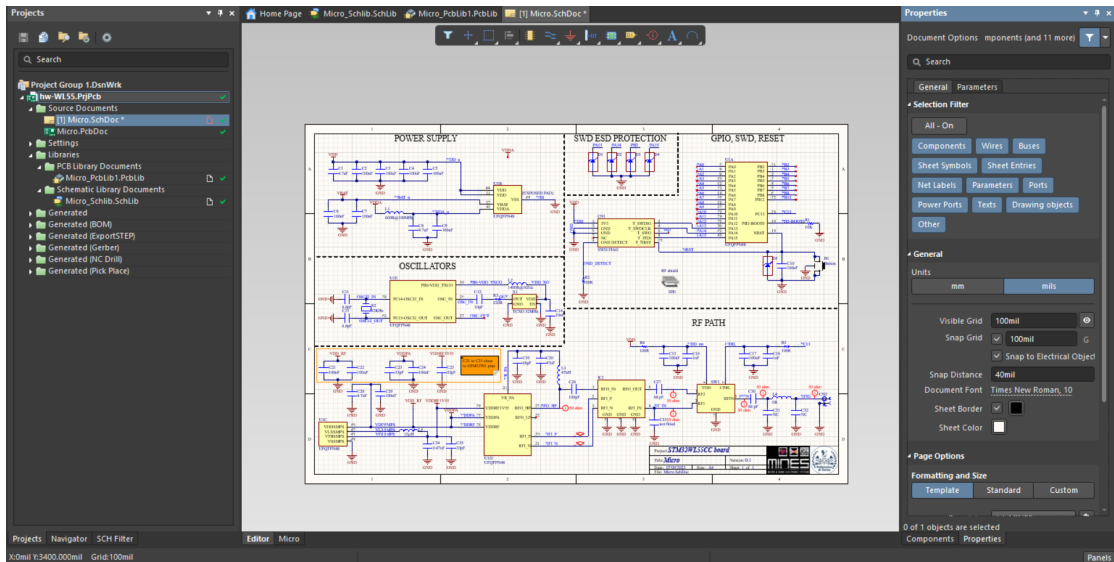


Figure 2.6: Schematic workspace.

PCB workspace Finally, once the schematic is complete, the physical design is realized in the PCB workspace, shown in fig. 2.7, where are placed the component, the interconnection, and all the elements that will be discussed in detail in chapter 3.

Output job file The last important workspace is shown in fig. 2.8. In this window, it is possible to automatize the generation process of all files that are typically used in the production of a printed circuit board (PCB):

- **Electrical schematic.**
- **Assembly files.**
- **Gerber files.**
- **Drill files.**
- **Bill of Material.**

In this way, it is possible to save all the needed parameters of the PCB documentation and manufacturing files. The required files will be discussed in chapter 3.

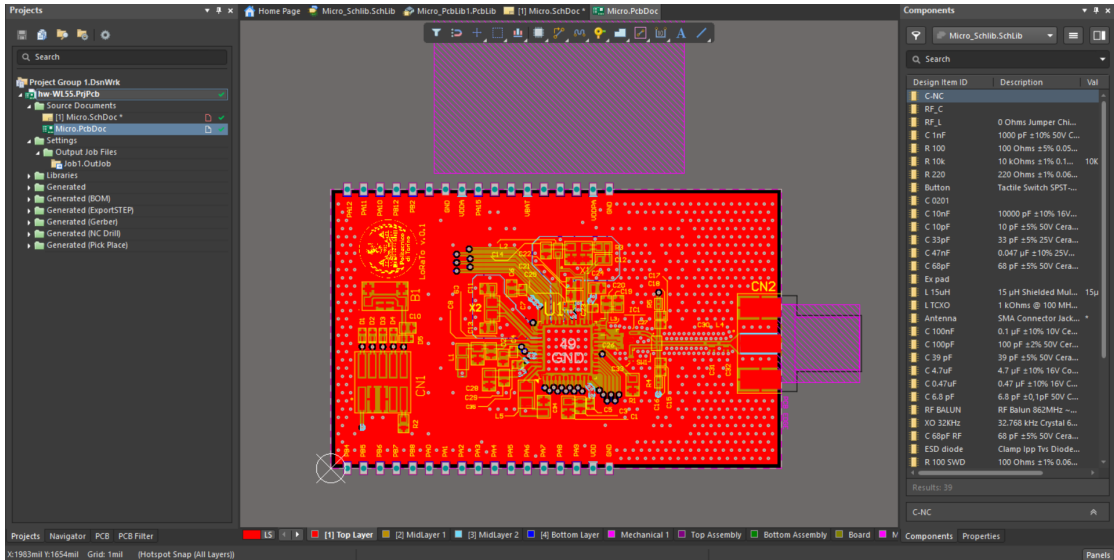


Figure 2.7: PCB workspace.

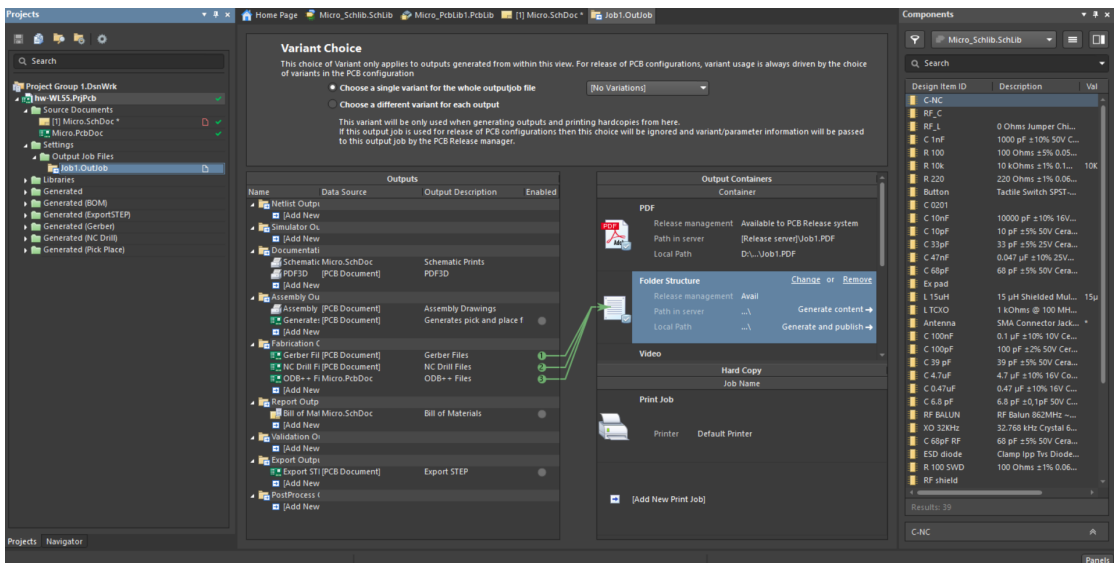


Figure 2.8: Output Job File workspace.

Chapter 3

Proposed design

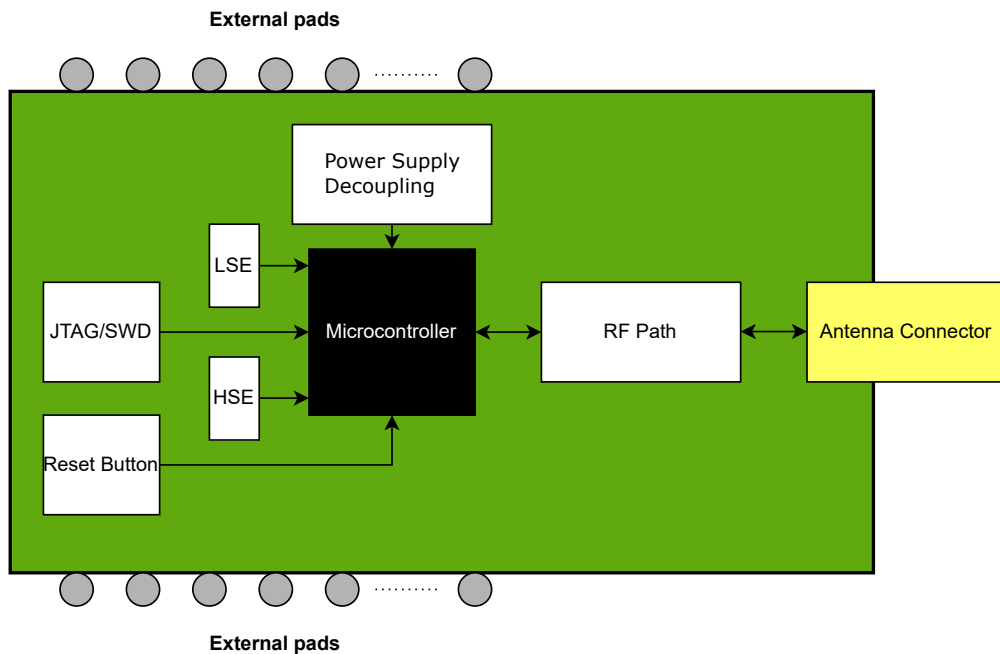


Figure 3.1: Draft block diagram.

As mentioned in chapter 2, the goal of this thesis is to realize a PCB with only the minimum bill of material to work as a LoRa core end node: the needed main blocks to make the microcontroller able to work and receive/transmit data are shown in fig. 3.1 and are:

- **Microcontroller;**
- **Voltage supply decoupling** and filtering components;

- **Programming interface:** a JTAG connector allows the communication with the microcontroller via SWD protocol, and a reset button is placed on board;
- **RF path:** it includes the filtering and matching network of the RF transmission line and the RF power supply configuration;
- **Antenna connector;**
- **Oscillators:** two oscillators are required for the target application in order to provide the clock for the real-time clock and the clock for the RF-PLL synthesizer, respectively called LSE and HSE oscillators;
- **External pads** that allow to solder the PCB on an application-specific PCB.

3.1 Microcontroller selection

STM32WL55 is available in two different packages:

- UFBGA73 with 73 solder balls, as shown in shown in fig. 3.2 and fig. 3.3;

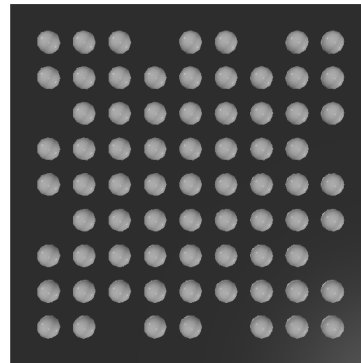
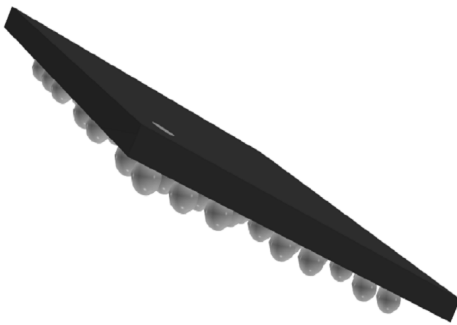


Figure 3.2: UFBGA73 side view.

Figure 3.3: UFBGA73 bottom view.

- UFQPFN48 with 49 flat contacts, as shown in fig. 3.4 and fig. 3.5.

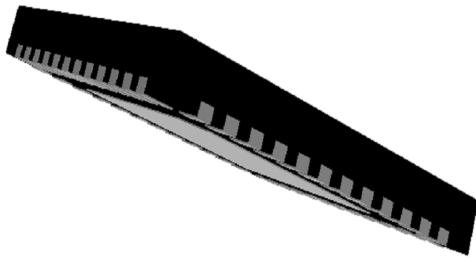


Figure 3.4: UFQPFN48 side view.

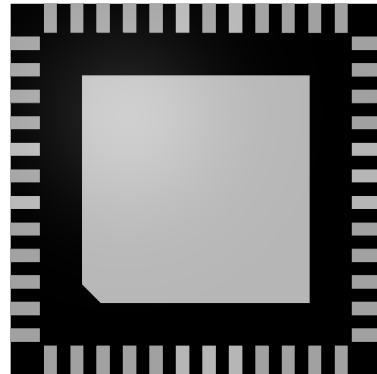


Figure 3.5: UFQPFN48 bottom view.

It is important to note that UFQPFN48 has fewer GPIOs than UFBGA73, which is irrelevant to the target application due to the fact that in precision agriculture use-case it is not needed many GPIOs. In addition of it, UFQPFN48 has been chosen since flat contacts are easier to solder on the PCB. In particular, the microcontroller at the basis of this thesis project is the **STM32WL55CCU6**.

3.2 Electrical schematic

There are different possibilities to implement the electrical schematic in Altium Designer: the design tool allows the user to make the complete schematic in a single sheet or to divide it into different blocks; moreover, the relation among these blocks can be "hierarchical" or "flatten". STMicroelectronics provides several resources to allow customers to deal with its product. In particular, in addition to datasheets and user manuals, there are some reference designs [7] that facilitate the design process around a certain application.

3.2.1 Voltage supply

The voltage supply scheme is shown in fig. 3.6. It is possible to distinguish three main power domains:

- **Core (digital)** power domain;
- **Analog** power domain;
- **RF** power domain, which will be discussed better in the next section.

In addition, a V_{BAT} pin is available and used to connect an external battery that will supply the backup phase when the main power supply drops unexpectedly to zero. As stated in the user manual [6], each power supply pin must be decoupled by means of ceramic capacitors as indicated in the power supply internal schematic shown in fig. 3.6. In fig. 3.7, the correspondent decoupling circuit for the power supply from Altium Designer is shown.

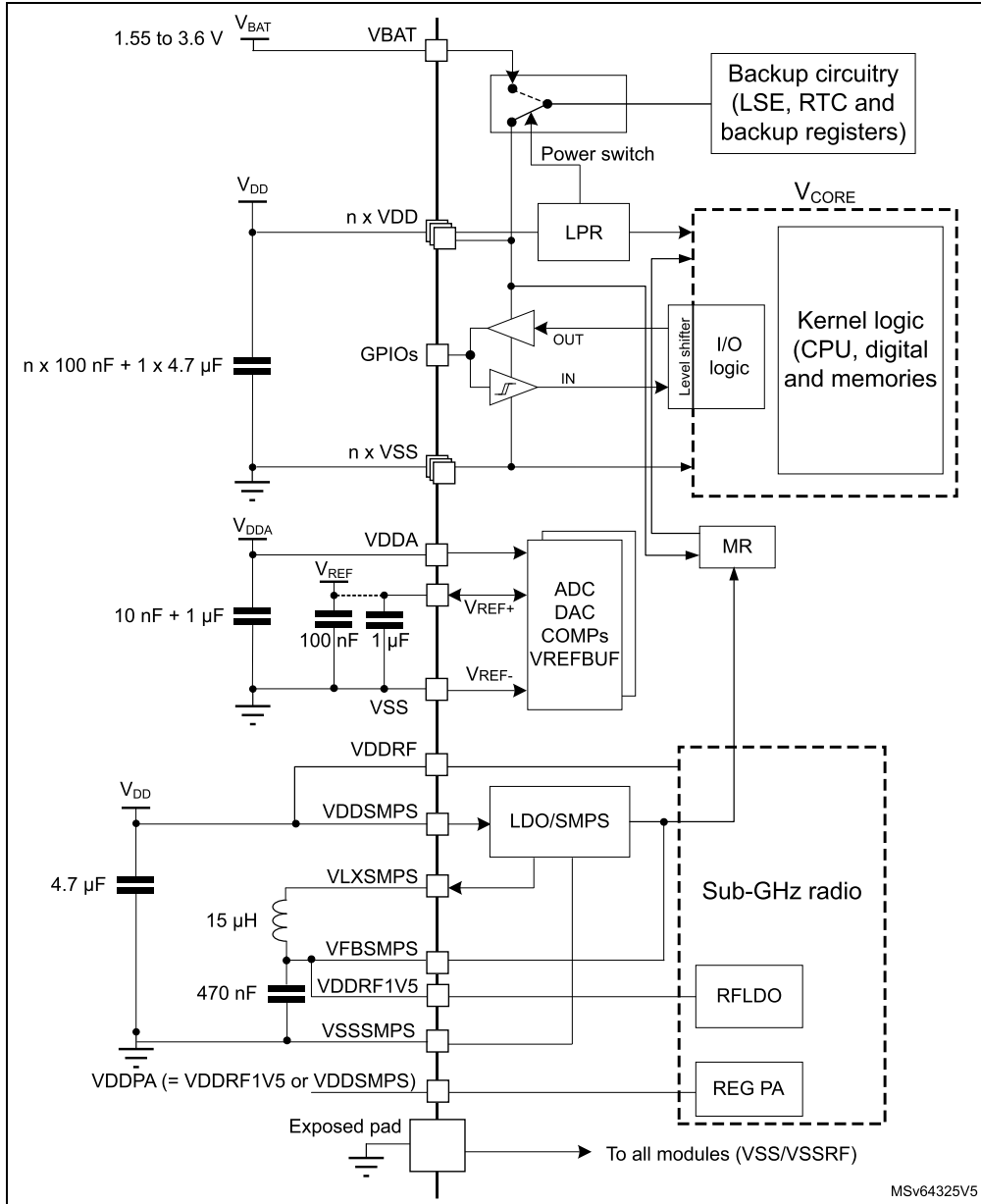


Figure 3.6: Internal Power Supply scheme from [6].

POWER SUPPLY

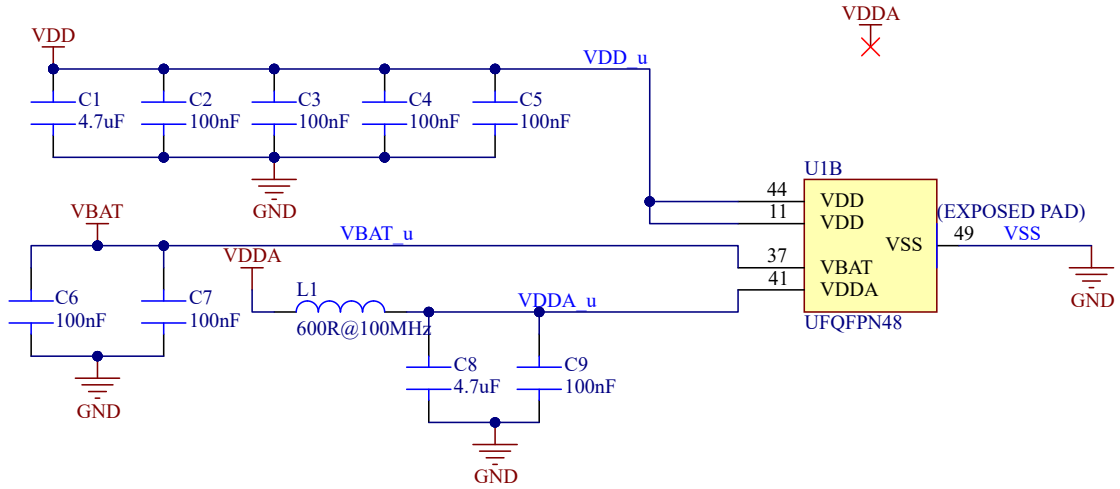


Figure 3.7: Power supply schematic.

As it is shown in fig. 3.7, each power domain is associated with a different power reference symbol, and this is due to the fact that in the PCB layout workspace every component and its interconnections, including power references, are indicated by the editor, and so by naming differently the power references it's easy to divide correspondent power plane in the PCB layout (this will be better seen in the chapter 3). The RF power supply is not included in this part of the schematic due to the fact that, as stated in the user manual [6], its configuration changes depending on some conditions that are going to be discussed in section 3.2.2.

3.2.2 RF path

As mentioned in section 2.2.1, the sub-GHz RF system presents two output pins and a differential input. The proposed design plans to have a shared RF line connected to the antenna, so the needed main blocks to design the RF path are:

- a switch to drive the signal from the antenna to the input or to connect the output line;
- a filtering and matching stage to ensure low noise and low losses;
- a net called π net to adjust eventually the impedance matching;
- the antenna connector.

These elements need to be designed accurately in such a way to have the best performances:

Antenna connector As stated in [8], different connectors can be used for LoRa gateway and end nodes. Since the target device should be as small as possible, the two possible choices are the following:

- **U.FL connector:** it is an ultra-small SMD connector (fig. 3.8) that is used in minimal designs when the footprint is very critical. The disadvantage is that it has to be managed very carefully, and typically, only a few reconnections (or mating cycles) are guaranteed before breaking the connector.



Figure 3.8: U.FL connector, form [8].

- **SMA connector:** it is a bigger than the U.FL connector (fig. 3.9) that guaranty better mechanical performances and mating cycles.

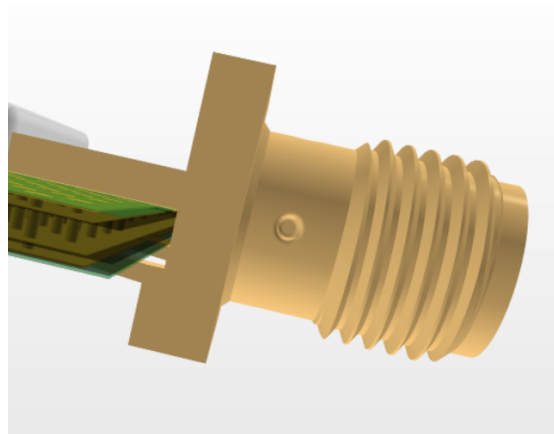


Figure 3.9: SMA connector.

The target application is designed to be as small as possible, but in an agricultural context, an outdoor application where the temperature can change in a wide range, some mechanical performances should be guaranteed. Moreover, the dimension

of the antenna connector is important but not too critical; therefore the SMA connector is chosen to ensure better performance.

Π net As stated in [9], one of the necessary circuits that allow to reach good RF performances is a $C - L - C$ π filtering and matching network. Actually, during the first design phase, this net is only implemented with an 0Ω resistor instead of the inductor, leaving the capacitors not connected since if the other parts of the RF path are well designed, the action of this net is unnecessary. Otherwise, the VNA (vector network analyzer) will be used to reach the impedance matching. In fig. 3.10, the correspondent portion of the circuit where the antenna (CN2) is connected to the π net is shown. It is also important when designing an RF path or, in general, a transmission line to distinguish lines with different characteristics impedance, and this is done employing the **parameter set indicator** in Altium Designer that shows that the line is a 50Ω line.

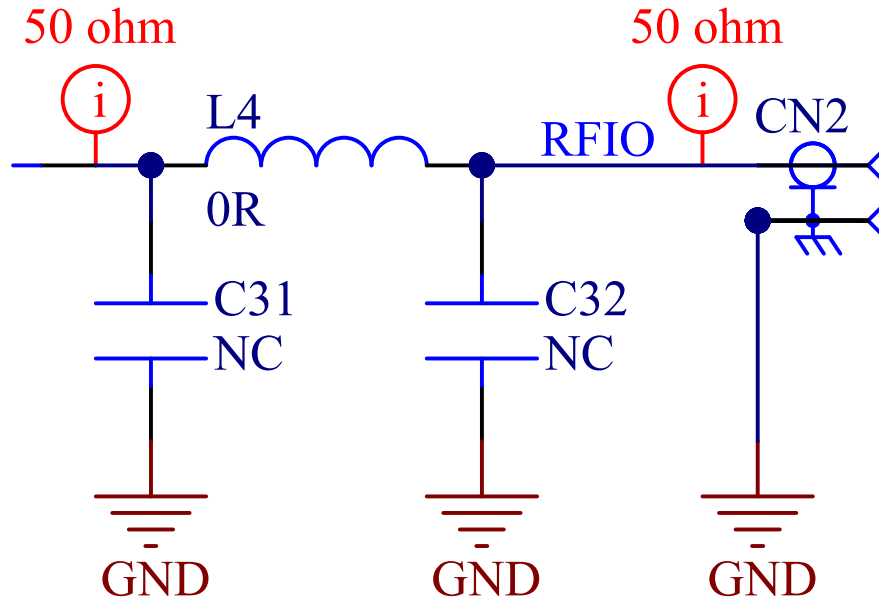


Figure 3.10: Antenna and π net.

RF switch Since only one output is needed for the target application, a two-port RF switch has been chosen [10]. Its working principle is very simple: as shown in fig. 3.11 and considering the portion of schematic in fig. 3.12 when it is on, one control signal is used to switch the connection between the RF_{IN} net to RF_{IN} when the control signal is **LOW** or to RF_{OUT} when the control signal is **HIGH**. The switch is always supplied by VDD and controlled through one microcontroller GPIO.

Switched Paths	Ctrl
RFin - RF1	0
RFin - RF2	1

Figure 3.11: RF switch truth table from [10].

In fig. 3.12, it is possible to see also some blocking capacitors, precisely C27 and C30: they are used to block input and output DC current.

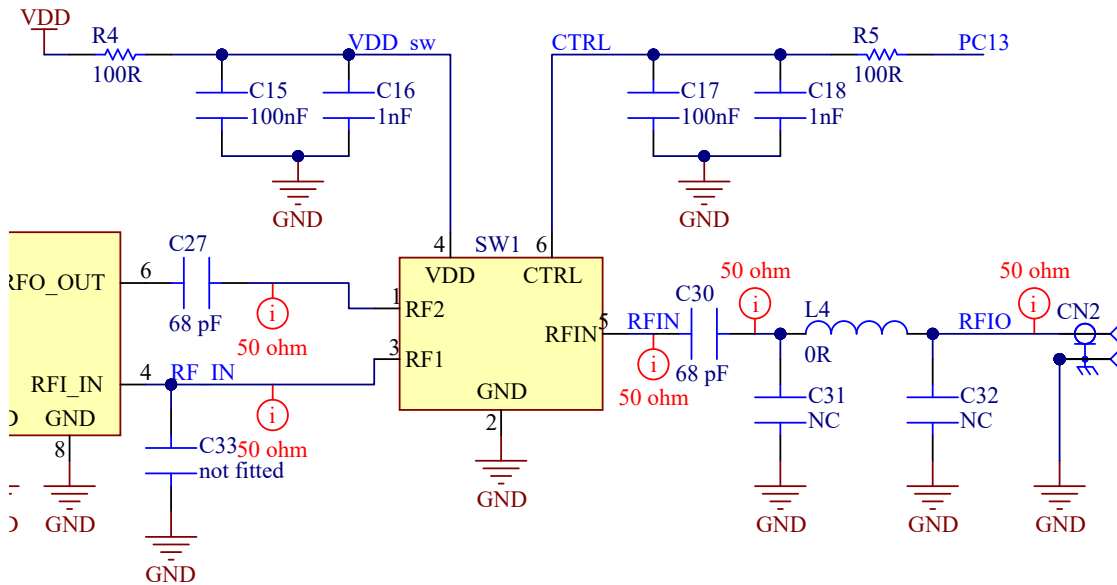


Figure 3.12: RF switch circuit.

Filtering and matching stage The application note [9] shows a complete design process that allows the design of a proper filtering and matching stage through several nets that are used to change the characteristics impedance through the Smith chart, decouple the RF power amplifier supply, and convert a single mode input to a differential mode one. An example is reported in fig. 3.13. This approach allow to reach high performances in term of power losses, but is also expensive in term of design complexity, PCB area and components.

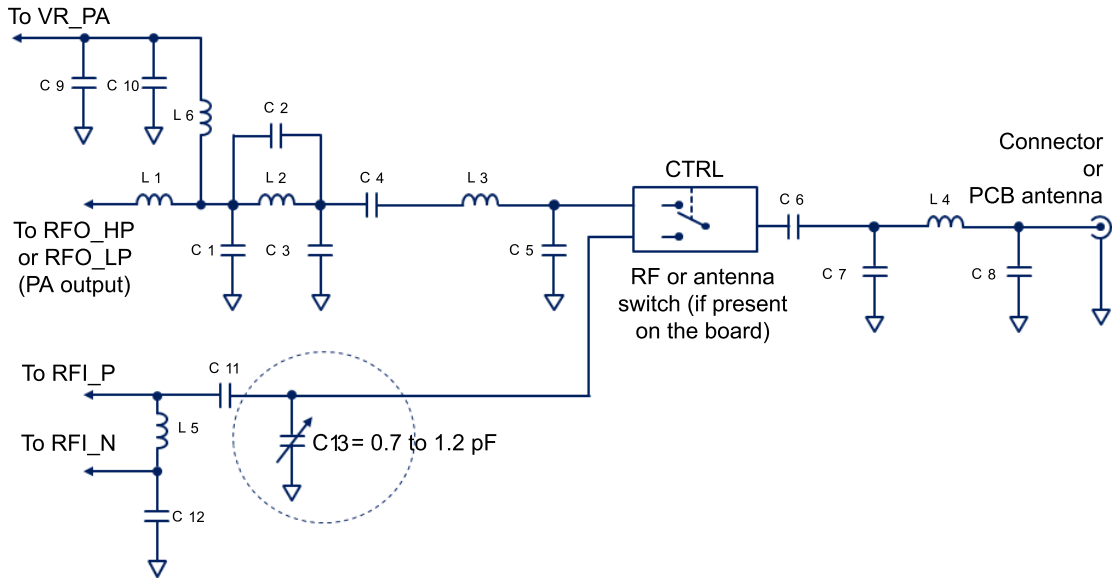


Figure 3.13: Filtering and matching stage from [9].

Integrated passive filtering and matching IC A second approach is to use a ready-to-use passive component provided by STMicroelectronics that implements the filtering and matching stage in a small integrated circuit, leading to fewer components and less footprint. These components, called **baluns**, are available for different microcontrollers/PCB stack-ups/desired output power combinations, as shown in fig. 3.14. In particular, the **BALFHB-WL-02D3** [11] is used in this

Power frequency	22 dBm 864-928 MHz		15 dBm 864-928 MHz	
	#PCB layers			
STM32WL BGA	4	BALFHB-WL-01D3	2	BALFHB-WL-04D3
STM32WL QFN	4	BALFHB-WL-02D3	BALFHB-WL-03D3	BALFHB-WL-05D3 BALFHB-WL-06D3
Power frequency	17 dBm 470-530 MHz		STM32WL BGA	STM32WL5xJxIx STM32WLExJxIx
	#PCB layers		STM32WL QFN	STM32WL5xCxUx STM32WLExCxUx
	STM32WL BGA	4	BALFLB-WL-07D3	
STM32WL QFN	4	BALFLB-WL-08D3	2	BALFLB-WL-09D3

Figure 3.14: Balun types.

thesis since the microcontroller package is the STM32QFN and, as it is explained in section 3.3.1, the PCB is composed by four layers. The balun configuration in terms of electrical schematic, shown in fig. 3.15, is very simple since it is a passive device and thus needs only the connection with the correspondent RF

tracks. Moreover, notice that the differential input presents an indicator that imposes the two wire tracks to be a differential pair: this is an important setting of the electrical schematic because, in this way, in the PCB layout editor, the two physical wires can be designed automatically to have the same length, that is a mandatory condition for a differential pair of the transmission line. The last part

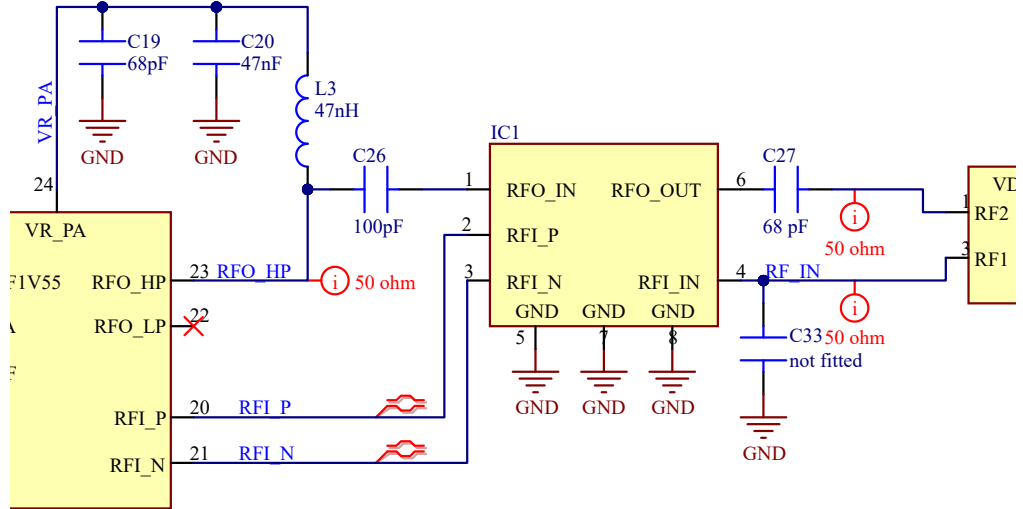


Figure 3.15: RF balun circuit.

of this portion of the circuit is the connection with the microcontroller pins: since the RF balun is used, only the decoupling with power supply is necessary, which is performed by the components C19, C20, and L3 as stated in [9]. The outputs of the RF balun are already matched at 50Ω for the single-end output and 100Ω for the differential input, so the only need is to connect the microcontroller to the component with tracks with the correspondent characteristics impedance and, as stated in the balun datasheet [11] and the reference design [7], only a DC block capacitor (C26) is needed.

3.2.3 RF power supply

The RF power supply scheme is discussed in this section because it is dependent on the RF configuration. As stated in the user manual, depending on the *RF_OUT* chosen, the power supply has a different configuration and, as it has been mentioned in section 2.2.1, the RF output used is the **high power** one. Moreover, it is possible to configure the microcontroller to use its LDO regulator (the DC/DC mode) to improve the power management of the RF stage; this option has been chosen for the

proposed design. Considering all this information, the power supply configuration is shown in fig. 3.16.

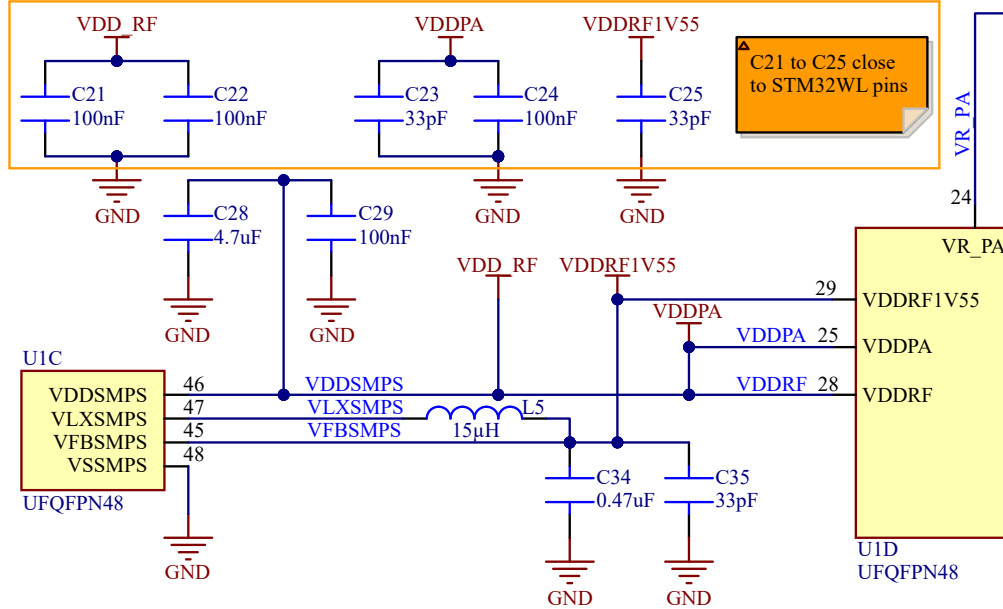


Figure 3.16: RF power supply configuration.

In order to have a good schematic and simplify the hardware layout's design phase, each net and power reference has its indicator; this also imposes a physical connection in the PCB layout, even if there is no direct connection in the schematic with a wire. For example, to have a more readable schematic, the decoupling capacitors of power supply pins are placed in the upper part of fig. 3.16 and seem not connected to the correspondent pin. Actually, naming the supply reference with the same name as the reference placed on the pin wire guarantees the physical connection, as mentioned before. The configuration used is described as follows:

- **High output power:** the VDD_PA pin must be connected to a power supply up to 3.6V and connected so to the VDD_RF power domain, both connected with $VDDSMPS$ which is the power supply for the supply of the SMPS step-down converter.
- **SMPS mode:** a $15\mu H$ inductor is connected between the pins $VLXSMPS$ and $VFBSMPS$, as indicated by the user manual [6].

3.2.4 Oscillators

The microcontroller STM32WL55 offers different possibilities for what concern the clock source: in fact, it has a set of internal clock generator that can be used as

system clock for a certain application, but it also has two clock input sources that are the so-called **LSE** and **HSE** and are used from the RF sub-system and not only. The application note [12] describes the oscillator design guidelines, which explains the kind of oscillator to be used and their functions. It also lists compatible oscillators and examples of electrical schematic and PCB layouts.

High Speed External clock The HSE clock is a 32MHz clock source that is used by the microcontroller as reference by the radio PLL. There are two kinds of source supported:

- Crystal oscillator;
- TCXO regulator.

The TCXO is a thermally compensated crystal oscillator: the difference with respect to the standard crystal oscillator is that the frequency is maintained constant over the temperature by means of a control voltage. Since the RF path can generate significant heat on the PCB that could influence the center frequency of the oscillator, the TCXO is recommended to ensure a precise central frequency and improve RF performance. As stated in the application note [12], the circuit needed by the TCXO [13] is shown in fig. 3.17. The contact *OSC_OUT* is unnecessary since no feedback is needed to drive the target frequency; voltage decoupling and current blocking elements are needed.

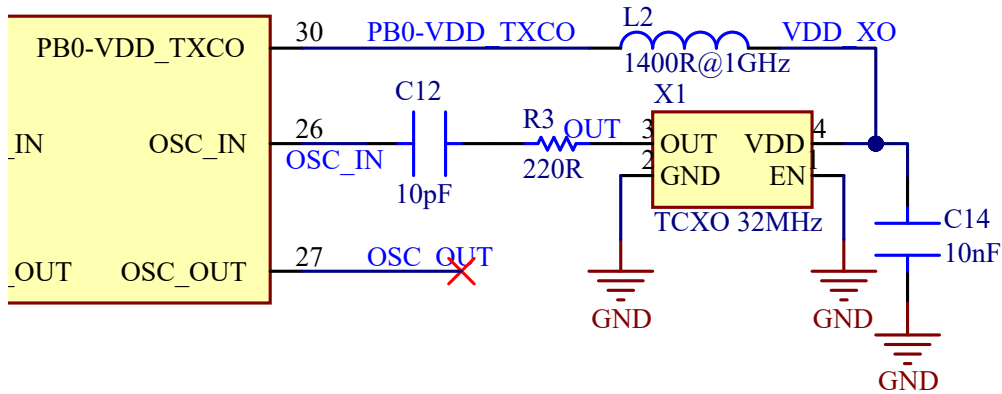


Figure 3.17: TXCO electric schematic.

Low Speed External clock The LSE oscillator is a $32.768kHz$ crystal oscillator that is used to provide the signal for the **Real Time Clock peripheral**: it is used to start a precise digital calendar that allows the implementation of different functions such as the so-called *Alarm* which are at the basis of the low-power mode and wake-up routines. This kind of oscillator requires a load capacitance to control the output frequency, given by eq. (3.1).

$$C_L = \frac{C_{L1}C_{L2}}{C_{L1} + C_{L2}} + C_S \quad (3.1)$$

where:

- $C_{L1} = C_{L2}$ are the capacitor connected to the oscillator
- C_S is the parasitic capacitance of the contact and track

The load capacitance is defined in the oscillator datasheet [14], and C_S should be computed. C_S is hard to estimate because it is needed to evaluate the parasitic capacitance of a complex element as a PCB. A typical PCB design process employs typical values (rule of thumb) from $2pF$ to $8pF$ [15] that cover standard PCB stacks. In case of fault, it is possible to adjust C_L working on C_{L1} and C_{L2} . Taking into account the reference design provided by STMicroelectronics, a similar layout has been implemented that considers a value of $C_{L1} = C_{L2} = 6.8pF$, which implies a parasitic capacitance $C_S = 5.6pF$. These assumptions are made considering that, as it will be shown in section 3.3.1, the usage of the RF balun requires a particular stack of PCB materials, and since it influences the parasitic capacitances of pads and tracks, it can be assumed that the needed capacitors by the oscillator have the same value of the ones used by the reference design.

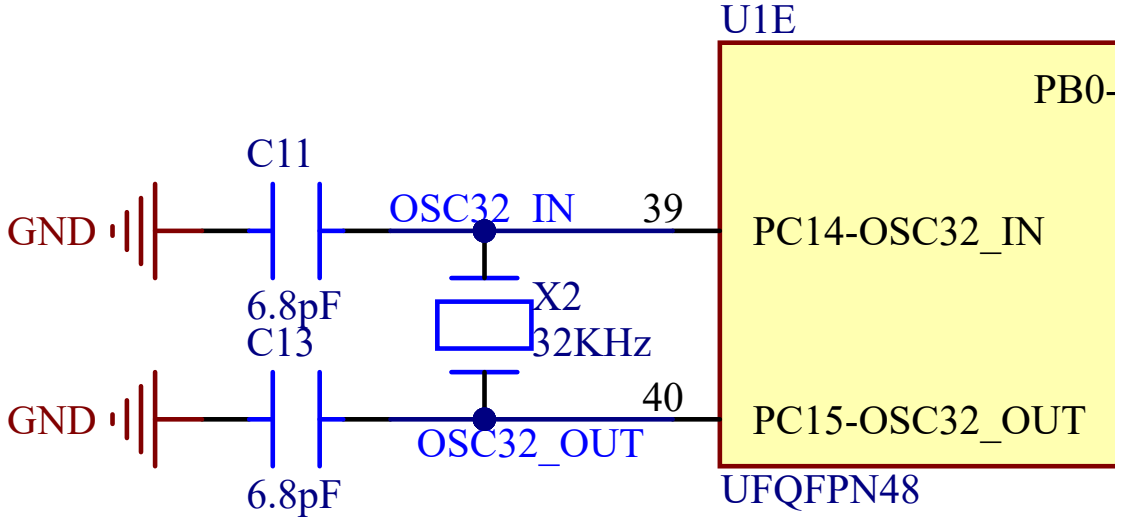


Figure 3.18: LSE electrical schematic.

3.2.5 Programming interface

The last fundamental element of the proposed design is the block that allows the user to program the microcontroller and reset it directly from the board. The microcontroller supports the SWD protocol that allows the programming and debugging of the device through the programmer tool **STlink-v3**. A JTAG connector [16], shown in fig. 3.19, is required. In fig. 3.20, the configuration of the connector and the reset button is shown: it is possible to see the presence of some diodes [17] that are used to protect the device from ESD event, as stated in [18]. Moreover, it is recommended to use an RF metal shield (SH1) [19] that covers the portion of PCB where the microcontroller, RF components, and oscillators are placed to reduce the emission of electromagnetic noises to the circuits around it.

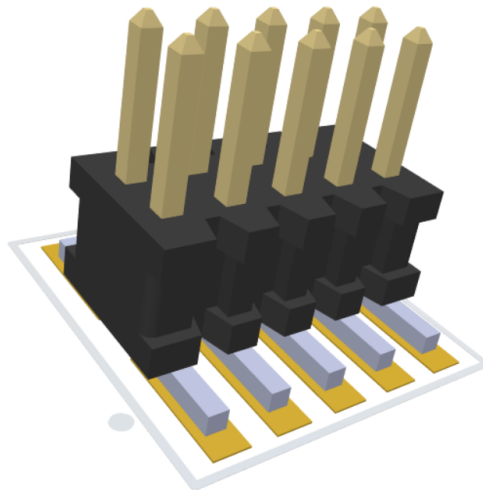


Figure 3.19: JTAG connector.

3.2.6 Complete schematic

In fig. 3.21, it is shown the complete schematic implemented in Altium Designer. From this schematic sheet, all the components with their characteristics and connections will be imported into the PCB design workspace, and the physical design can finally start and is explained in section 3.3. The final schematic contains all the components and their interconnections. It is possible to draw boxes and write comments or titles to divide the schematic into specific sections: in this case, the schematic is subdivided into the main building blocks described in this chapter. In a schematic sheet, it is a good practice to indicate all the useful information, such as:

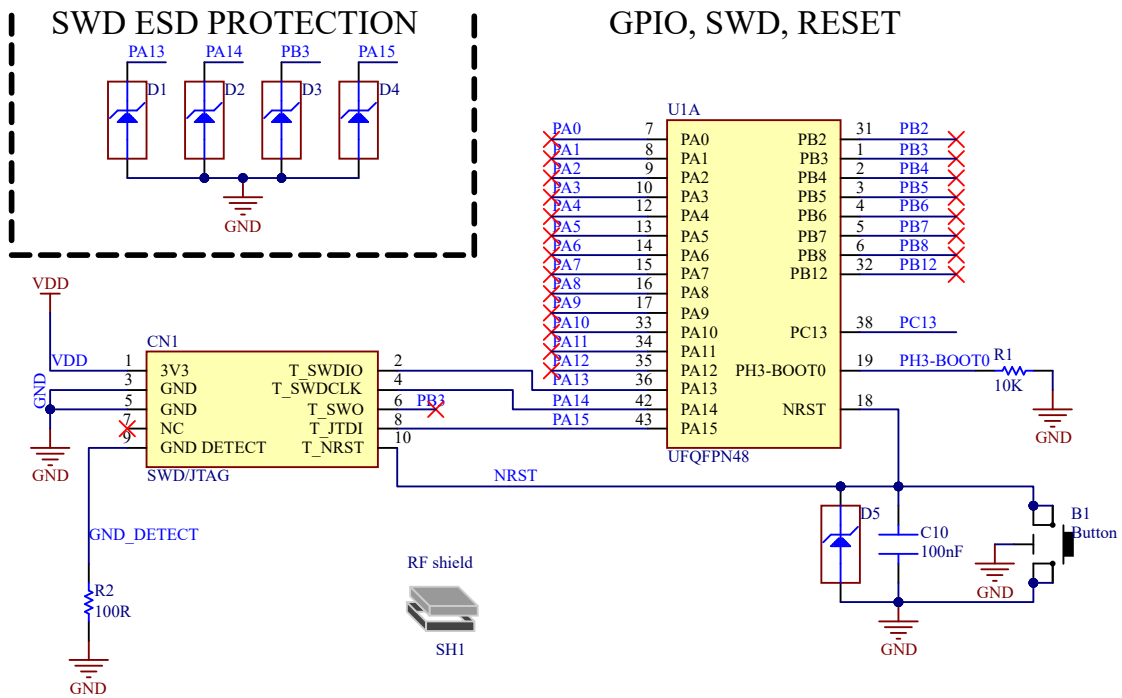


Figure 3.20: GPIOs, JTAG, RESET, and ESD protection electrical schematic.

- Project name;
- Title of the sheet;
- Version of the project: to distinguish different revisions of the same board;
- Size of the schematic;
- Number of sheets;
- Name of the Altium project file;
- Eventual logo of the company or, in this case, university and group.

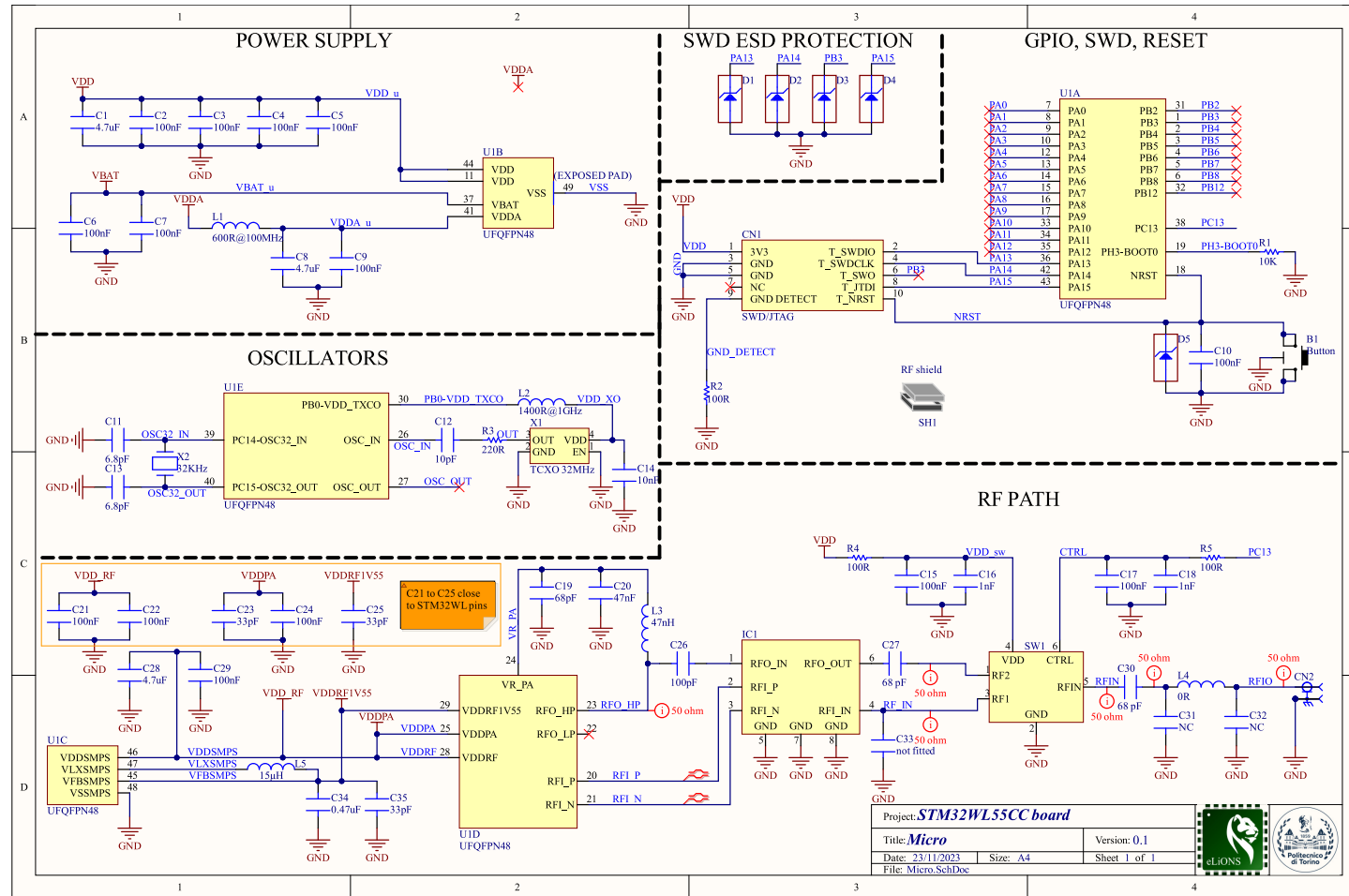



Figure 3.21: Complete electrical schematic.

Project: STM32WL55CC board		
Title: Micro	Version: 0,1	
Date: 23/11/2023	Size: A4	
File: Micro.SchDoc	Sheet 1 of 1	

3.3 Hardware design

This section will present the PCB physical layout, considering different aspects of the design process, such as design rules, PCB structure, and Altium Designer properties. This design phase starts when the electrical schematic is complete; however, if some changes occur in the electric schematic, such as the substitution of a circuit portion or a change in a component footprint, it is possible to automatically update the new components, footprint or connection from the schematic sheet workspace.

3.3.1 PCB stackup

The starting point of the PCB design is the definition of the PCB stack-up, which defines the number of layers and the material used for the PCB production. There are many possibilities that depend on the target application requirements: the presence of RF tracks (and signals) implies the necessity to decouple the RF signals from DC power circuits and introduce a reference plane for the RF return current (thus a transmission line return path). It is possible to design both 2-layer and 4-layer modules. The proposed design is realized on a 4-layer PCB in order to optimize different aspects of the system as stated in [20]:

- The **top layer** contains all the components on its surface, some interconnection, and the RF path. All the components are surrounded by a ground plane to provide a return path for high-frequency noises (both RF and oscillators) that reduces the coupling with the power supply and signals.
- The **first inner layer** is the reference ground plane and is also designed in order to provide low impedance for the RF return current.
- The **second inner layer** is used as power plane: each power domain is isolated from the others, and the choice of designing a power island instead of a small track to deliver the power supply provides a low impedance path. Moreover, as stated in [20] and [21] surrounding the power planes with ground traces/planes leads to a reduction of radiated emission.
- The **bottom layer** will contain the remaining interconnection to divide the signal traces from all the other components and simplify the signal routing phase. Moreover, as mentioned in the introduction, the bottom surface is left without components to allow the PCB to be soldered on other application-specific board surfaces.

The second step is the choice of the material for the PCB stackup. the balun datasheet [11] provides the stackup where the component has been characterized (fig. 3.22).

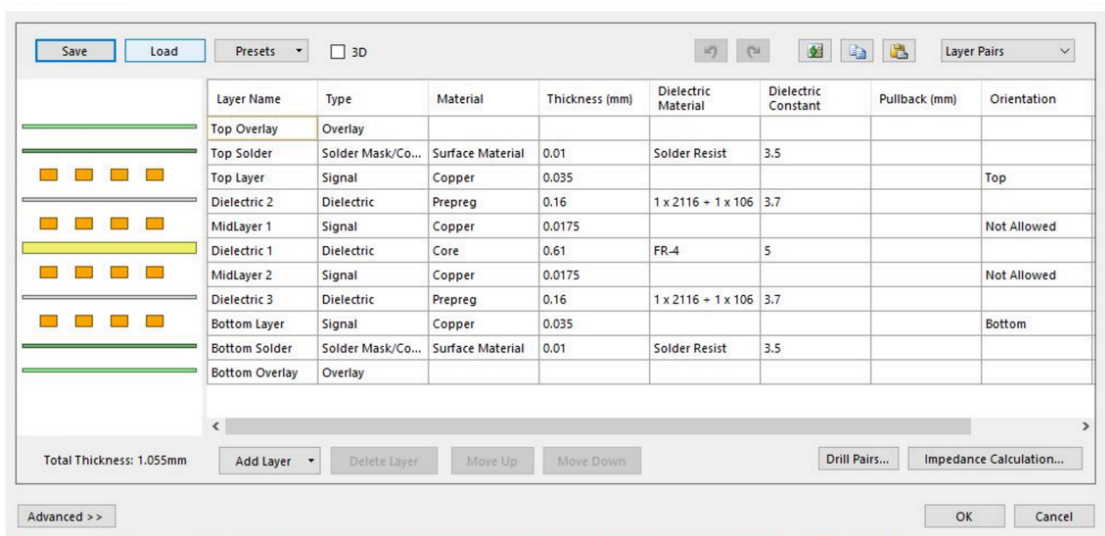


Figure 3.22: Characterized stackup from [11].

In fig. 3.23 is shown the chosen stackup with similar material and dimension, according to the information provided by the chosen manufacturer. Moreover, once the stackup is updated, there is the possibility to compute the parameter of a defined characteristics impedance: the tool returns a first approximation of the width of the wire track to have the desired characteristics impedance (essential parameter for RF track design).

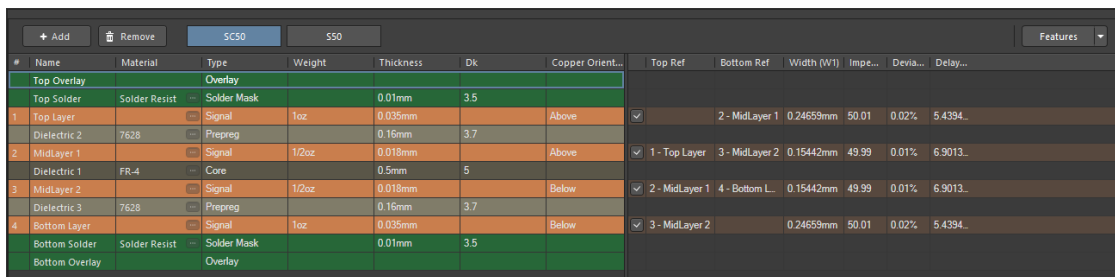


Figure 3.23: Altium Designer Stackup manager.

3.3.2 RF layout design rules

The RF layout is one of the most critical parts of the physical design since it is easy to significantly reduce performance due to impedance mismatch if some rules are not respected. Here are presented the main design rules:

Characteristics Impedance The first and most important parameter to be controlled is the characteristic impedance of the tracks that carry the RF signal. It depends on several parameters and can be evaluated using different tools, such as the integrated tools provided by Altium Designer. However, to validate the computation done by Altium Designer, another tool was used: AppCad, a simulation tool that allows the calculation of different parameters for RF, microwave, and wireless applications. In fig. 3.24, the simulation of the characteristics impedance of a coplanar waveguide, the implemented waveguide in this board, returns a value of 50Ω with a width equal to 0.3mm and a clearance equal to 0.16mm , that will be the dimension of the waveguide implemented in the proposed design. The 0.3mm transmission line width is used for the RF waveguide in almost all the PCB layout except for the connection between the RF balun IC and the microcontroller pins. In fact, the RF balun datasheet [11] provides the dimension of width and clearance for the output waveguide and the differential input waveguides, as shown in fig. 3.25.

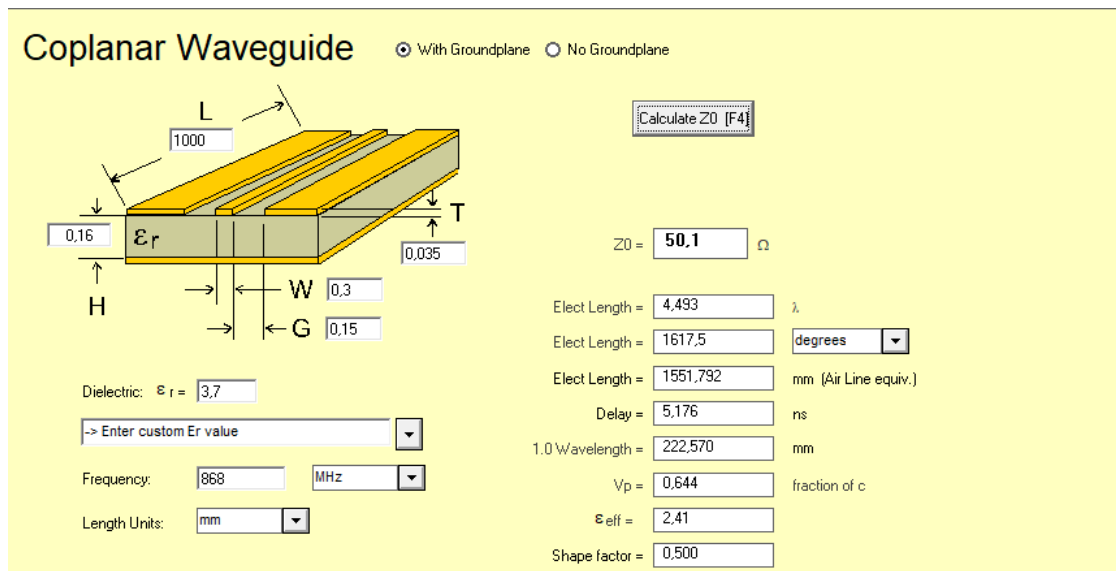


Figure 3.24: Appcad impedance calculator.

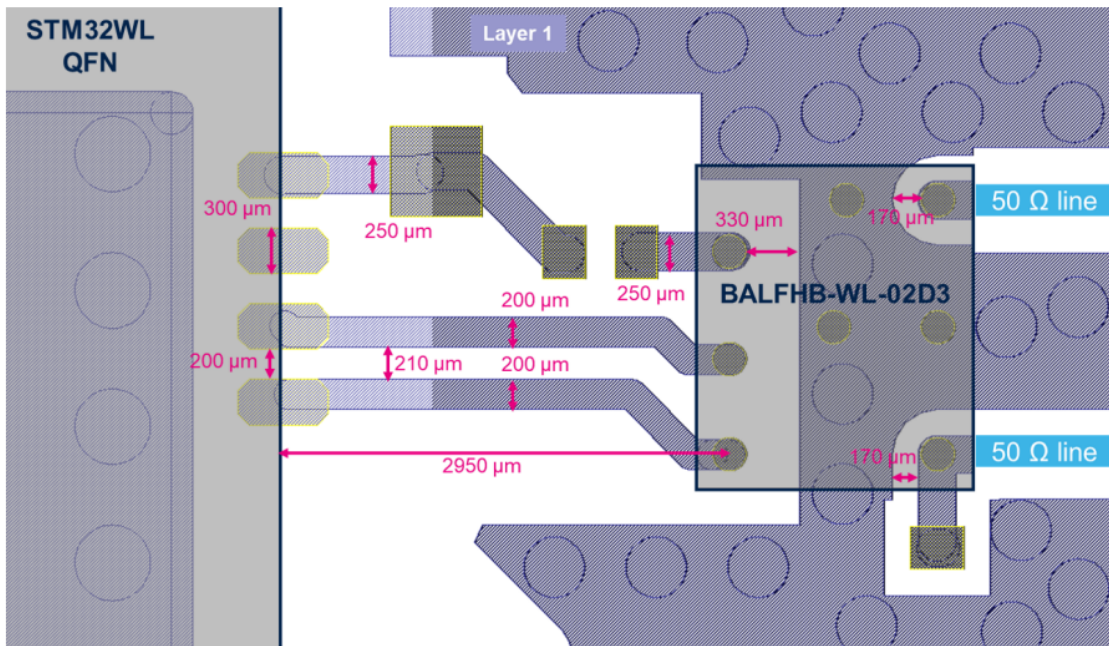


Figure 3.25: Recommended layout from [11].

Components design rules There are some recommendations to improve the performance of the RF stage that are applied to the components layout. For instance, the dimension of the pads and the presence of vias can influence the effective value of capacitors or inductors, as shown in fig. 3.26 and fig. 3.27.

Performance	Inductor pad type	Comment
Recommended		Short and same PAD width access traces, maintaining the original value of the inductance and Q-Factor
Not good		Be careful with this kind of tricks. This narrow trace contributes to increase the inductance, but this can decrease the equivalent Q-factor of the inductor. RF inductors are carefully made to have a high Q-factor. Do not ruin it.

Figure 3.26: Recommended inductor layout from [21].


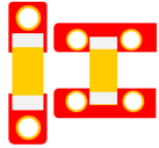



Performance	Capacitor pad type	Comment
Recommended		Short traces with multiples vias reducing return current impedance
Better		Short traces
Better		
Poor		Long traces between capacitor increasing series inductance
Not good		Thinner access track increasing the equivalent series inductance of the capacitor

Figure 3.27: Recommended capacitor layout from [21].

The final recommendation about components design is to design the width of pads equal to the transmission line width to avoid impedance discontinuities. An example is shown in fig. 3.28, where a capacitor is sized to have a footprint that does not exceed the limit of the RF line.

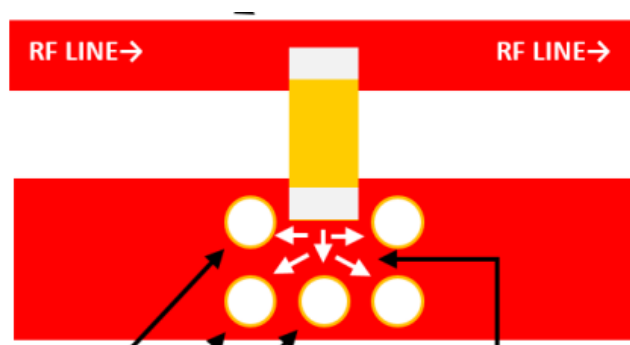


Figure 3.28: Pad width recommendation from [21].

Transmission line design rules The last thing to care about in the RF path, as stated in [21], regards the transmission line design to avoid different kinds of issues:

- **RF return path current:** the current return path must be a direct current path, so it's forbidden to place vias in the ground plane below the RF transmission line, otherwise losses and discontinuities will increase.
- **slots in the ground plane** where the high-frequency current flows can act as an antenna, so they must be avoided
- **track transition:** if a pad of a component has a different size from the nominal transmission line width, the recommendation is to design a smooth transition of the TL.
- when a **track has to change direction**, 90° transition are not good. The line should always maintain the same width or change direction with a shape of at least 45° .

Ground flooding, metal shield and power plane It is strongly suggested by [21] to:

- Place a metal shield and try to reduce apertures from the bottom to avoid high-order harmonics causing interferences with other circuits.
- Flooding unused PCB areas with a GND plane and multiple vias that connect all the GND planes of the PCB to reduce EMC issues.
- uses a different power plane for each power domain and allows the GND vias to pass through them to avoid floating GND.

Decoupling capacitors Decoupling capacitors must be placed as close as possible to the correspondent power supply pin to have the lower-value ones near the pin and then continue in ascending order.

3.3.3 Oscillators design rules

In order to guarantee the correct working of the oscillators, the application note [12] provides a few design rules that must be considered:

Oscillators guard rings Each oscillator must be surrounded by a guard ring connected to the ground in order to isolate the rest of the circuit from the disturbances caused by the oscillator. An example is shown in fig. 3.29. Moreover, high-speed signals should be routed far away from the oscillator to avoid any coupling between them.

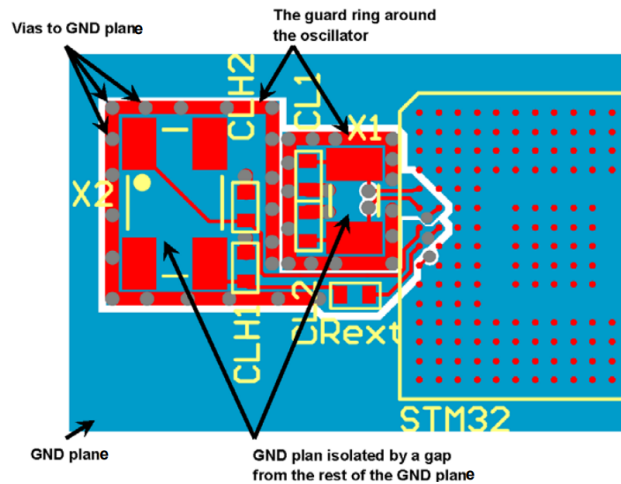


Figure 3.29: Oscillator guard ring example from [12].

Oscillators ground plane In order to ensure the return current to flow directly to the microcontroller GND pins, the area of the ground plane around the oscillator connected to the oscillator guard ring should be separated by a gap from the rest of the ground plane, as shown in fig. 3.30.

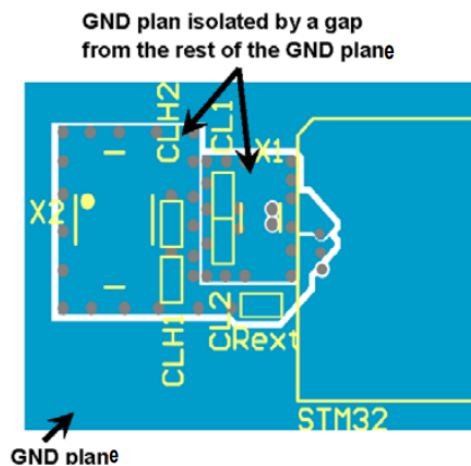


Figure 3.30: Oscillator ground plane example from [12].

Heat isolation It is advised to isolate the oscillators near an RF source by means of a cutout in the PCB top layer around the oscillators unless a TCXO is used. A TCXO has been employed in this thesis, so no cutouts are designed in the PCB.

3.3.4 PCB manufacturer design rules

Once the design constraint has been defined, other constraints depend on the manufacturer's production capability. In fact, even if there are no other constraints with respect to the ones mentioned in section 3.3.2 and section 3.3.3, all manufacturers provide a set of design rules that are typically the minimum dimension that they can deal with, such as tracks width dimension, minimum clearance and so on. The rules derived from the chosen manufacturer are listed in table 3.1

Rule	Min. value	Description
Spacing	0.15 mm	spacing between copper element
Track Width	0.15 mm	track width for copper thickness $\leq 70 \mu\text{m}$
Drill Size	0.2 mm	diameter of drill entities (e.g. via)
Annular Ring	0.15 mm	minimum annular ring for via
Solder Mask Swell	0.2 mm	distance between solder mask and copper elements
Silkscreen	0.2 mm	distance between silkscreen and copper elements
Contour Routing	0.2 mm	distance between copper elements and board edges
Board to Edge Clearance	0.2 mm	distance between edges and solid plane (e.g., power or ground planes)
Silkscreen Printing Dimension	0.16 mm	dimension of printing on the board surface
Ground Plane Spacing	0.15 mm	spacing between ground planes and copper elements

Table 3.1: Applied Design Rules from [22].

3.4 PCB design

The PCB layout phase can start once all the elements and constraints needed to realize the physical design are ready. In this phase, the layout with the actual physical dimension is realized in the PCB design workspace, where all the components with their footprints are imported from the electrical schematic.

3.4.1 PCB area

Firstly, the PCB board shape is defined through the board planning mode: in fig. 3.31 it is shown the correspondent tool in Altium Designer. It is possible to see the dimension, area, and number of components in the property panel under the section **board information**.

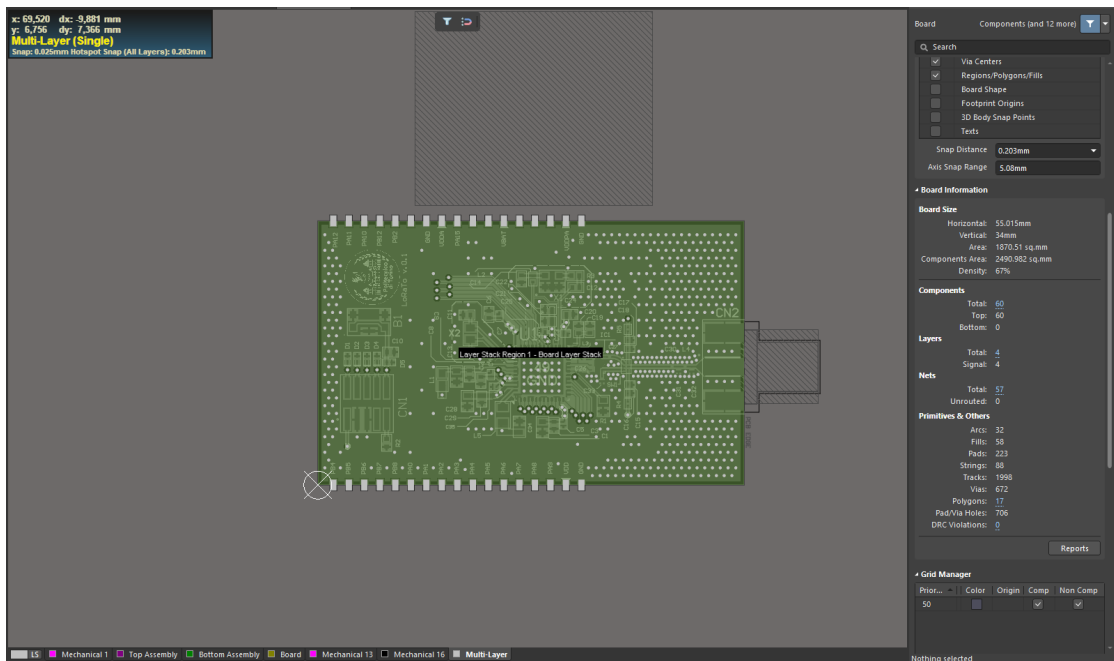


Figure 3.31: Board shape definition.

3.4.2 PCB layout

All components must be placed in the PCB area when designing the physical layout, and, in the proposed design, the PCB will contain all its components on the top surface, while the bottom one is left without elements.

Top layer As shown in fig. 3.32, all the components and the silkscreen are designed on the top of the PCB.

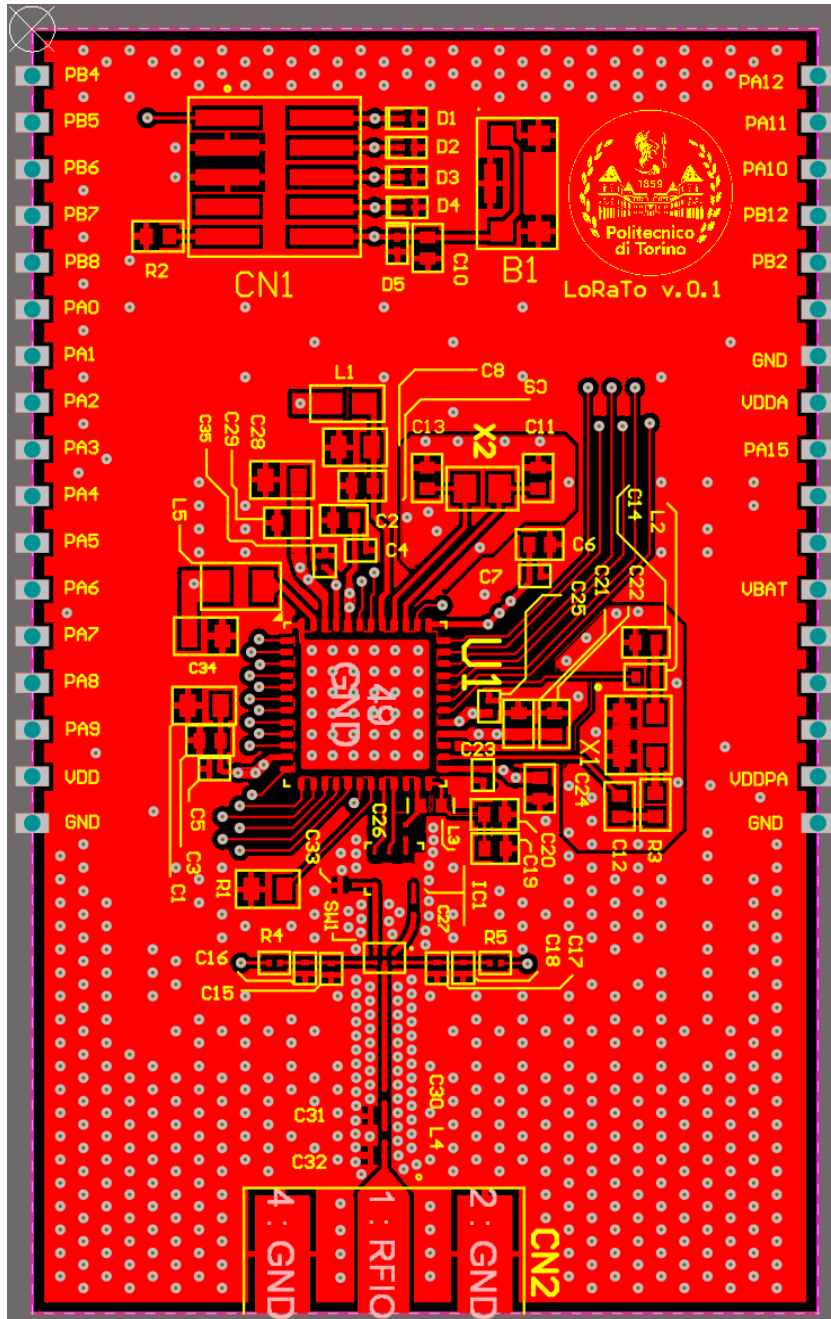


Figure 3.32: Top layer layout.

Ground plane The ground plane is shown in fig. 3.33; it is possible to distinguish the separation of the oscillators ground planes and the RF return path without vias.

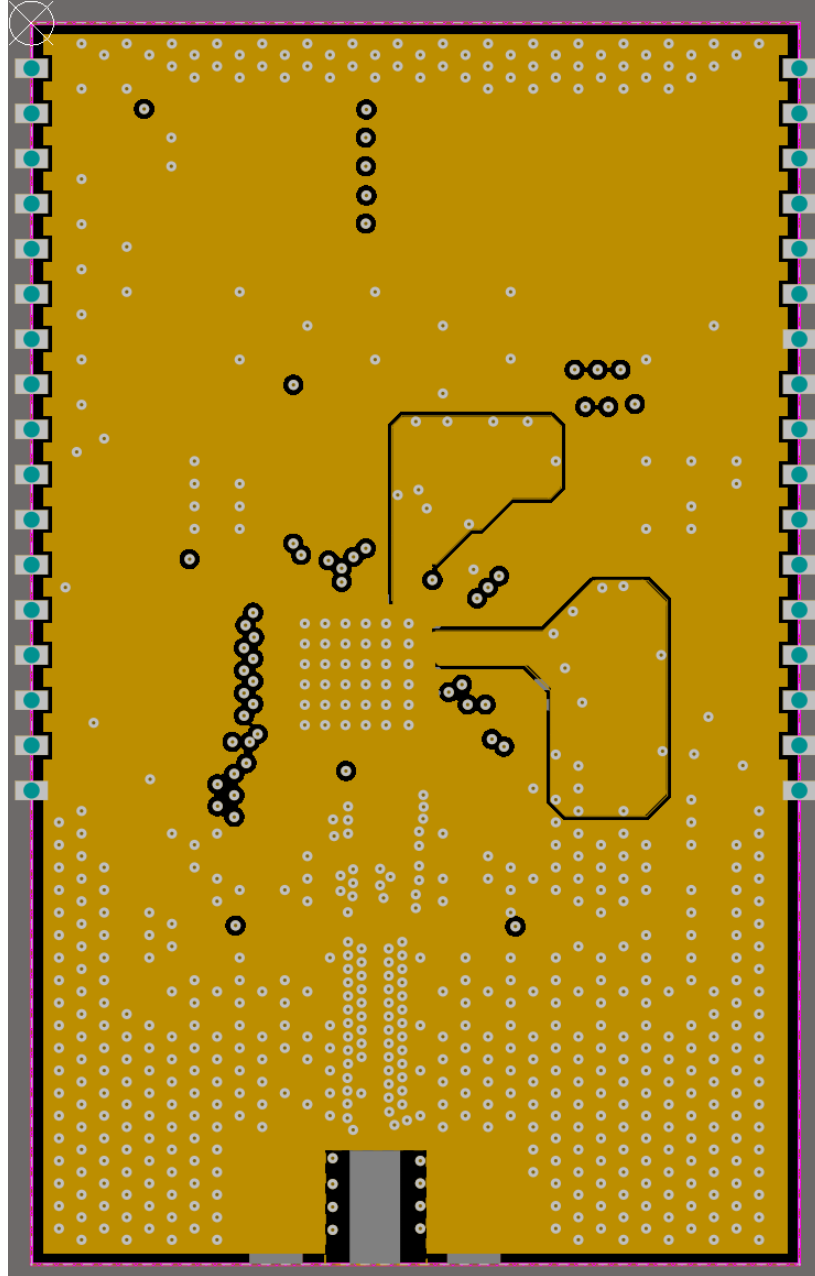


Figure 3.33: Ground plane.

Power plane The power plane is shown in fig. 3.34, it is possible to distinguish the main power domain island (VDD, VDDPA, VBAT, VDDA), and some few interconnections that are still power supply connection. The remaining part of the plane is filled with the polygon pour connected to GND.

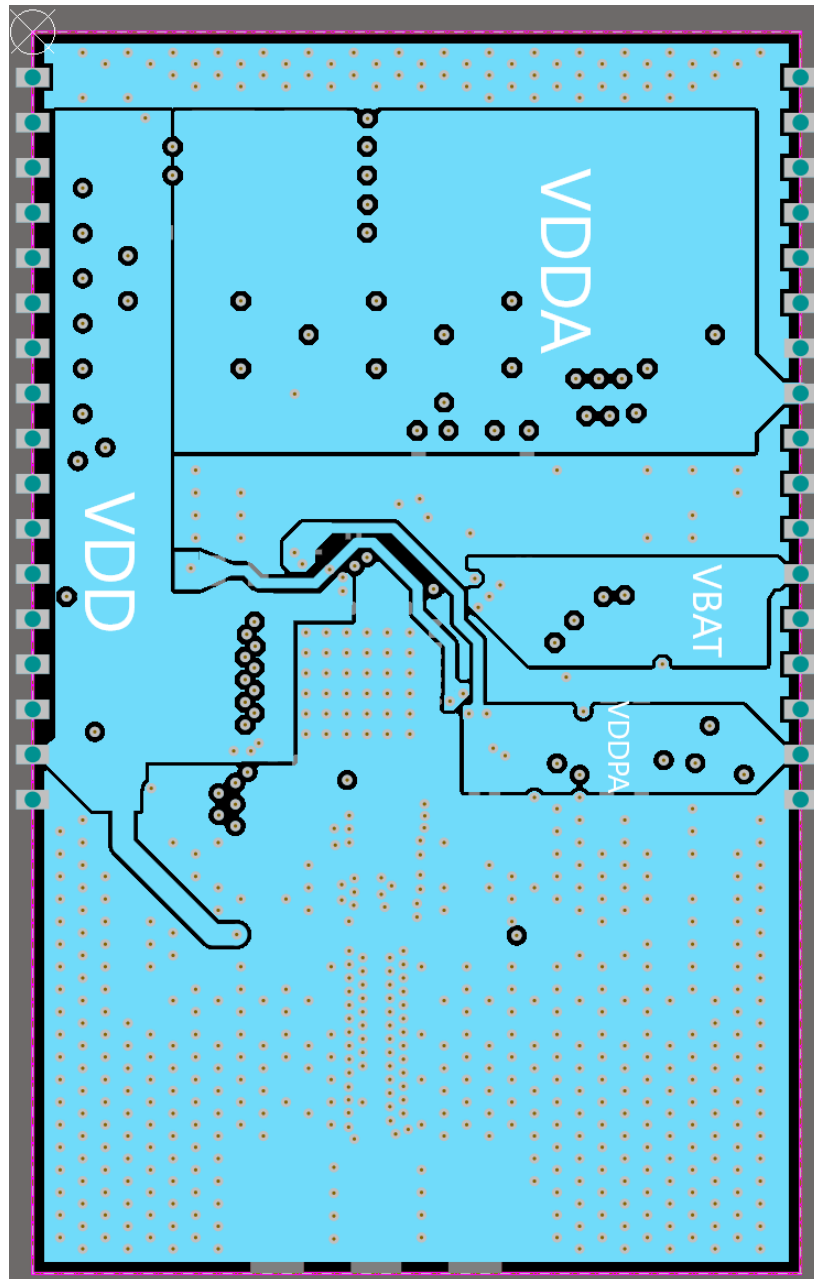


Figure 3.34: Power plane.

Bottom layer. The last layer shown in fig. 3.35 is used for signal routing.

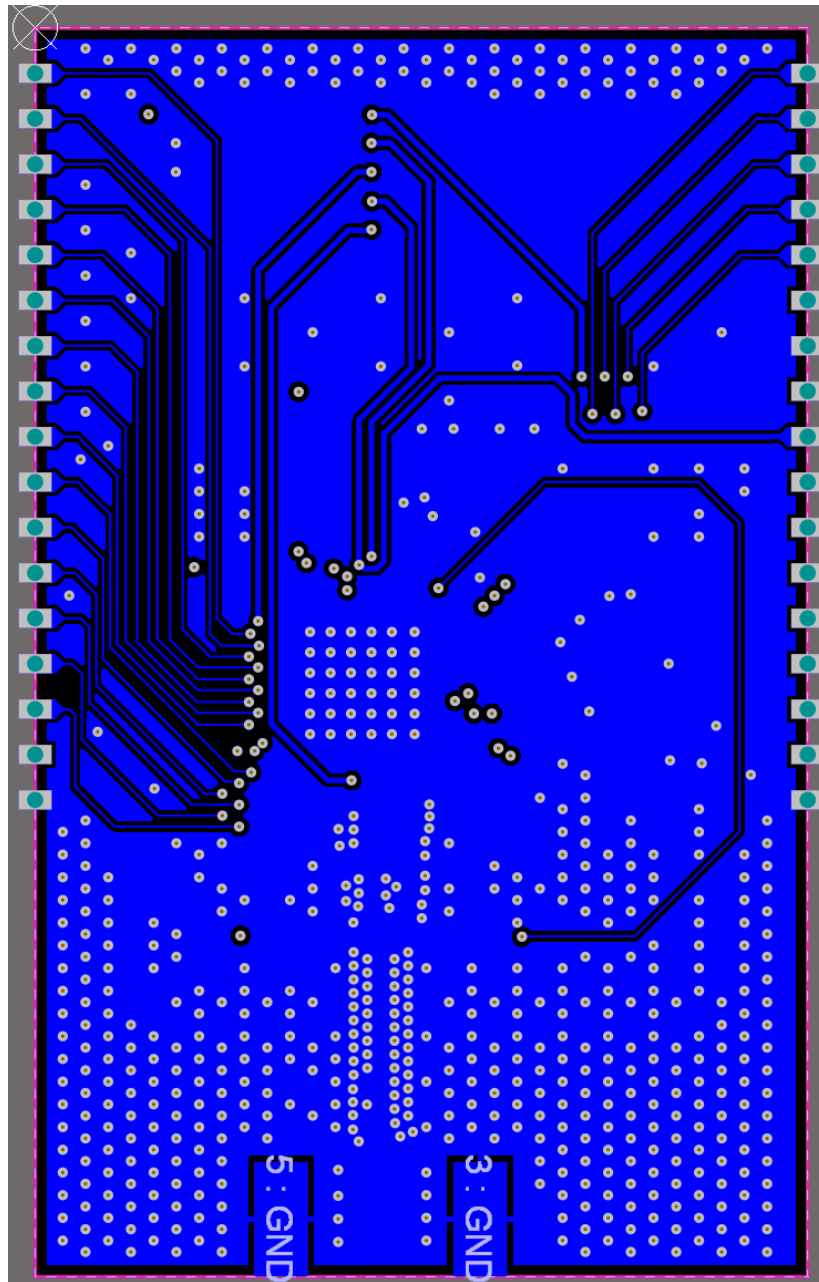


Figure 3.35: Bottom layer

3.4.3 PCB Validation and Manufacturing files

Once the physical design is completed, only few steps are required to manufacture the board prototype. First, the PCB validation is performed through Altium Designer: all the design rules described up to now are listed in the proper section, and the software automatically checks the PCB layout. Once there are no errors on the design, the last step is to generate the needed files for the manufacturing:

- **Gerber files:** files that represent the elements of the PCB, such as interconnections, power planes, vias, pads, and so on;
- **Drill files:** files that contain information about the position of all the drills present on board;
- **Pick and place file:** file where information about the physical position of the components is stored. This file is needed if the soldering of the component is done employing an automatic pick-and-place machine;
- **Assembly drawings:** PDF files that contain the drawing of the board and the components with 1:1 representation;
- **Schematic:** PDF files generated to display all logical connections of the board (fig. 3.21) and the components information (fig. 3.36);
- **BOM:** the bill of materials contains information about all the used components on the board, including price, component part number, quantity, etc. An example of BOM for the proposed design is shown in table 3.2.

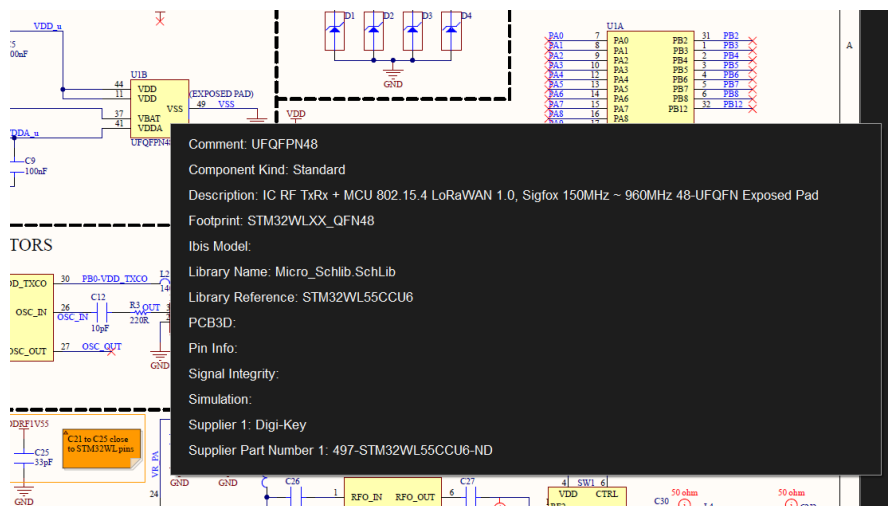


Figure 3.36: Component information from schematic.

table 3.2 legend:

- D = Designator.
- QTY = Quantity.
- VAL = Value.
- M. P/N = Manufacturer part number.
- DESC = Description.
- A. = To assemble.
- P. = Provided by customer.

D.	QTY.	VAL	M. P/N	DESC.	A.	P.
B1	1		KMR231GLFS	Tactile Switch SPST-NO Top Actuated Surface Mount	YES	NO
C1, C8, C28	3	4.7uF	CC0603KRX5R7BB475	4.7 μ F \pm 10% 16V Ceramic Capacitor X5R0603 (1608 Metric)	YES	NO
C2, C3, C6, C9, C10, C21, C22, C24, C29	9	100nF	CC0402KRX7R7BB104	0.1 μ F \pm 10% 16V Ceramic Capacitor X7R0402 (1005 Metric)	YES	NO
C4, C5, C7	3	100nF	C0603X5R1E104K030BB	0.1 μ F \pm 10% 25V Ceramic Capacitor X5R0201 (0603 Metric)	YES	NO
C11, C13	2	6.8pF	CQ0402BRNPO9BN6R8	6.8 pF \pm 0.1pF 50V Ceramic Capacitor C0G, NP0 0402 (1005 Metric)	YES	NO
C12	1	10pF	CC0402JRNPO9BN100	10 pF \pm 5% 50V Ceramic Capacitor C0G, NP0 0402 (1005 Metric)	YES	NO

Table 3.2: BOM.

D.	QTY.	VAL	M. P/N	DESC.	A.	P.
C14	1	10nF	CC0402KRX7R7BB103	10000 pF $\pm 10\%$ 16V Ceramic Capacitor X7R 0402 (1005 Metric)	YES	NO
C15, C17	2	100nF	GRM033R61A104KE15D	0.1 μ F $\pm 10\%$ 10V Ceramic Capacitor X5R 0201 (0603 Metric)	YES	NO
C16, C18	2	1nF	GRM033R71H102KA12J	1000 pF $\pm 10\%$ 50V Ceramic Capacitor X7R 0201 (0603 Metric)	YES	NO
C19	1	68pF	GCM1555C1H680JA16D	68 pF $\pm 5\%$ 50V Ceramic Capacitor C0G, NP0 0402 (1005 Metric)	YES	NO
C20	1	47nF	GCM155R71E473KA55D	0.047 μ F $\pm 10\%$ 25V Ceramic Capacitor X7R 0402 (1005 Metric)	YES	NO

Table 3.2: BOM.

D.	QTY.	VAL	M. P/N	DESC.	A.	P.
C23, C25, C35	3	33pF	GJM0335C1E330JB01D	33 pF $\pm 5\%$ 25V Ceramic Capacitor C0G, NP0 0201 (0603 Metric)	YES	NO
C26	1	100pF	0201N101G500CT	100 pF $\pm 2\%$ 50V Ceramic Capacitor C0G, NP0 0201 (0603 Metric)	YES	NO
C27, C30	2	68 pF	GRM0335C1H680JA01D	68 pF $\pm 5\%$ 50V Ceramic Capacitor C0G, NP0 0201 (0603 Metric)	YES	NO
C34	1	0.47uF	C1608X5R1C474K080AA	0.47 μ F $\pm 10\%$ 16V Ceramic Capacitor X5R 0603 (1608 Metric)	YES	NO
CN1	1		FTSH-105-01-L-DV-K-TR	"Connector Header Surface Mount 10 position 0.050" (1.27mm)"	YES	NO
CN2	1		142-0701-841	SMA Connector Jack, Female Socket 50Ohm Board Edge, End Launch Solder	YES	NO

Table 3.2: BOM.

D.	QTY.	VAL	M. P/N	DESC.	A.	P.
D1, D2, D3, D4, D5	5		ESDALC6V1-1U2	Clamp Ipp Tvs Diode Surface Mount ST0201	YES	NO
L1	1	600R@100MHz	BLM18AG601SN1D	600 Ohms @ 100 MHz 1 Ferrite Bead 0603 (1608 Metric) 500mA 380mOhm	YES	NO
L2	1	1400R@1GHz	BLM15HG102SN1D	1 kOhms @ 100 MHz 1 Signal Line Ferrite Bead 0402 (1005 Met- ric) 250mA 1.1Ohm	YES	NO
L3	1	47nH	LQW15AN47NG00D	RF choke, 47 nH Unshielded Wirewound Inductor 210 mA 1.08Ohm Max 0402 (1005 Metric)	YES	NO
L4	1	0R	RC0201JR-070RL	0 Ohms Jumper Chip Resistor 0201 (0603 Metric) Thick Film	YES	NO

Table 3.2: BOM.

D.	QTY.	VAL	M. P/N	DESC.	A.	P.
L5	1	15uH	MLZ2012N150LTD25	15 μ H Shielded Multi-layer Inductor 350 mA 470mOhm 0805 (2012 Metric)	YES	NO
R1	1	10k ohm	RC0603FR-0710KL	10 kOhms \pm 1% 0.1W, 1/10W Chip Resistor 0603 (1608 Metric) Moisture Resistant Thick Film	YES	NO
R2	1	100 ohm	RC0402FR-07100RL	100 Ohms \pm 1% 0.063W, 1/16W Chip Resistor 0402 (1005 Metric) Moisture Resistant Thick Film	YES	NO
R3	1	220 ohm	RC0402FR-07220RL	220 Ohms \pm 1% 0.063W, 1/16W Chip Resistor 0402 (1005 Metric) Moisture Resistant Thick Film	YES	NO
R4, R5	2	100 ohm	CRCW0201100RJNED	100 Ohms \pm 5% 0.05W, 1/20W Chip Resistor 0201 (0603 Metric) Thick Film	YES	NO

Table 3.2: BOM.

D.	QTY.	VAL	M. P/N	DESC.	A.	P.
IC1	1		BALFHB-WL-02D3	RF Balun 862MHz 928MHz 8-WFBGA, CSPBGA	YES	NO
SW1	1		BGS12WN6E6327XTSA1	RF Switch IC WLAN SPDT 50Ohm PG- TSNP-6-10	YES	NO
U1	1		STM32WL55CCU6	IC RF TxRx + MCU 802.15.4 LoRaWAN 1.0, Sigfox 150MHz 960MHz 48-UFQFN Exposed Pad	YES	NO
X1	1	32MHz	ECS-TXO-25CSMV-320- DY-TR	32 MHz TCXO Clipped Sine Wave Oscillator 1.7V 3.465V 4-SMD, No Lead	YES	NO
X2	1	32.768 KHz	NX2012SA-32.768KHZ- EXS00A-MU00527	32.768 kHz Crystal 6pF 80 kOhms 2-SMD, No Lead	YES	NO

Table 3.2: BOM.

3.5 Firmware configuration

This thesis aims to provide a complete and ready-to-use core module that will simplify the design process of future applications. In this context, the hardware design is not the only required step: in this section, the code configuration is described, and as the final output, a skeleton code is provided to abstract the application-specific code from the hardware implementation. In this way, when a new application is required, the code that considers the board hardware configuration is provided to the user that has only to deal with its application-specific main code.

3.5.1 LoRa configuration on CubeIde

The tool used to manage LoRaTo firmware is STM32CubeIde, which allows the user to manage the microcontroller configuration, write, upload, and debug the code. The first step is to configure some parameters related to the target application: in fig. 3.37 is shown the graphical interface of the STM32CubeMX application, that opens the **ioc** extension file, from which it is possible to configure the GPIO settings, clock configuration and other functionalities of the microcontroller.

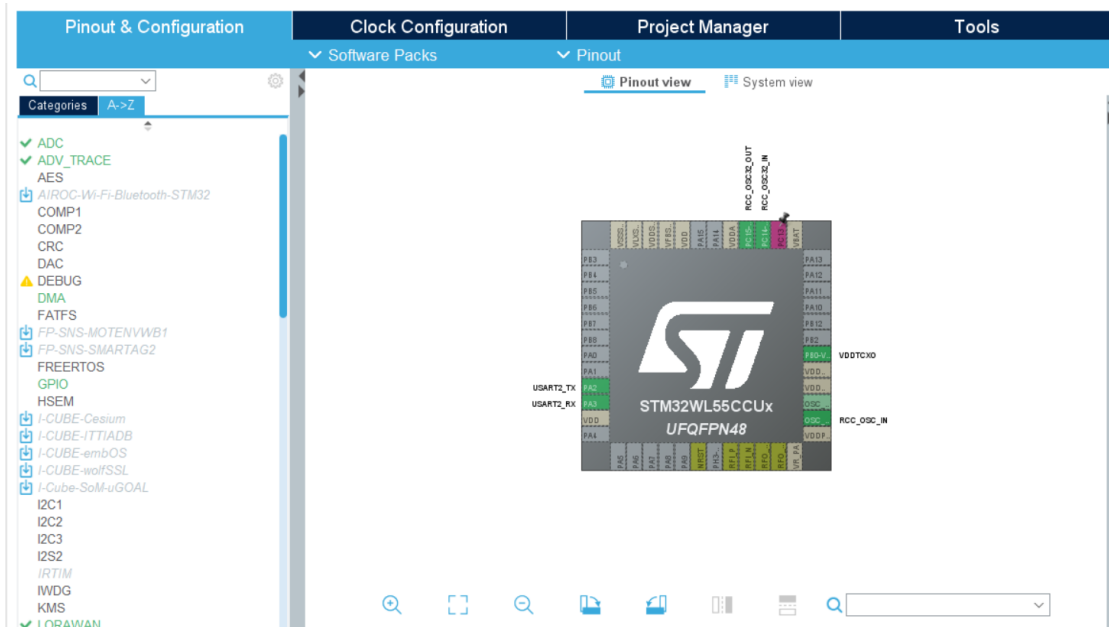


Figure 3.37: STM32CubeMX ioc configuration.

Pinout & configuration

The first part of the microcontroller configuration regards the GPIOs and functionalities settings. In particular, all the settings listed in this section are needed by the custom hardware and LoRa target application.

ADC In the ADC section under the **mode** menu tick the boxes correspondent to:

- Temperature Sensor Channel.
- Vrefint Channel

then under the **configuration** menu set:

- Clock Prescaler = Synchronous clock mode divided by 4.
- Overrun behaviour = Overrun data overwritten.
- Sampling Time Common 1 and 2: 160.5 Cycles

RTC Under the **mode** menu tick the boxes:

- Activate Cock Source
- Activate calendar

and set:

- Alarm A = Internal Alarm A

under the **configuration** menu:

- enable both interrupts in the **NVIC settings** window;
- set in the **Parameter settings** window:
 - Asynchronous Predivider value = RTC_PREDIV_A ;
 - Bin Mode = Free running Binary mode;
 - Alarm A Binary AutoControl = $RTC_ALARMSUBSECONDBIN_AUTOCLR_NO$.
- add the following user constants in the **user constants** window:
 - $RTC_N_PREDIV_S = 10$;
 - $RTC_PREDIV_S = ((1 \ll RTC_N_PREDIV_S) - 1)$;
 - $RTC_PREDIV_A = ((1 \ll (15 - RTC_N_PREDIV_S)) - 1)$.

RCC under the **mode** menu configure:

- High Speed Clock(HSE) = TCXO;
- Low Speed Clock(LSE) = Crystal/Ceramic Resonator;

under the **configuration** menu:

- in the **parameters settings** windows set:
 - Flash Latency(WS) = 2 WS (3 CPU cycle);
 - Power Regulator Voltage Scale = Scale 1.

USART2 under the **mode** menu set:

- Mode = Asynchronous.

under the **configuration** menu set:

- in **DMA Settings** add *USART2_TX* DMA Request with channel DMA1 Channel 5;
- in **NVIC** enable both interrupts.

NVIC in the **configuration** menu set in the **NVIC** the preemption priority equal to 2 for both USART2 and DMA interrupts .

SUBGHZ under the **mode** menu:

- tick the box **activated**.

under the **configuration** menu:

- in the **parameter settings** window set **Baudrate Prescalar Value = 4**;
- in the **NVIC** window enable the interrupt.

LORAWAN Here, all the parameters related to the desired LoRa protocol are configured. Enable LoRa in the **mode** menu, then in the **configuration** menu:

- in the **LoRaWAN application** window set:
 - Application = end node skeleton;
 - Send Tx on Timer or Button Evt = *TX_ON_TIMER*;

- Active region = `LORAMAC_REGION_EU868` (to set the carrier frequency depending on the geographical area);
 - Transmission duty cycle = desired delay between two consecutive transmissions (in milliseconds);
 - Application user port = 2 (port used to receive downlink messages);
 - Switch class port = 3;
 - Default class = `CLASS_A`;
 - Handler Adaptive Data Rate = ON;
 - Default activation type = OTAA;
 - Force rejoin at each reboot: check the box;
 - Default Tx output power = `TX_POWER_0`;
 - Default Unicast ping slots periodicity = 4;
 - Default response timeout (ms) = 8000.
- in the **LoRaWAN middleware** window:
 - Tick the box **Region Europe freq. 868**;
 - Tick the box **Enable context management storage** ;
 - Select the LoRaWAN Link Layer specification version = v1.0.3;
 - Select radio Driver 0 Bsp via ext Settings.
 - in the **Platform Settings** window select the correspondent elements for ADC, USART, and RTC that are available after the previous configurations.
 - **LoRaWAN commissioning** window contains the information for the connection between the end node and the gateway, which are:
 - LoRaWAN device EUI: is a unique device ID that is used to identify the end node;
 - App/Join EUI: is a unique device ID that is used to identify the application server;
 - Application key: is the encryption key, unique for each device.

Clock Configuration

The clock tree can be configured in this panel, shown in fig. 3.39. For the target application, the TCXO has been chosen as HSE, but as it is explained in section section 4.4 for preliminary tests, the MSI is used therefore the clock is configured such as:

- in the box **input frequency** on the top-left side of the clock tree, the value of the chosen LSE oscillator is selected ($32.768kHz$), and as a consequence, the LSE mux, indicated as **RTC Clock Mux** is set to LSE.
- in the box **input frequency** on the middle-left side of the clock tree, the HSE is set to $32MHz$, which is the operating frequency of the TCXO, nevertheless the **System Clock Mux** the MSI with value $32MHz$ is selected as system clock, as suggested from the LoRa recommendation configuration shown in figure fig. 3.38.

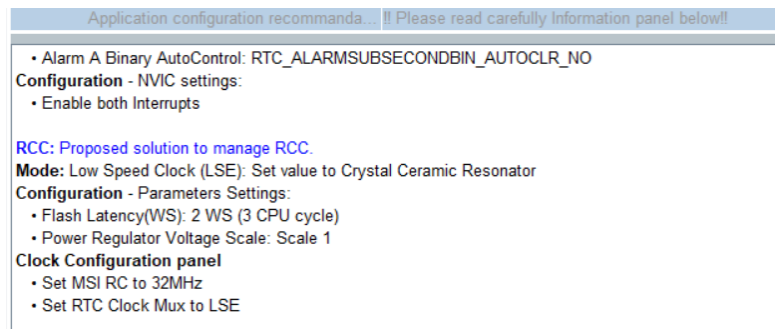


Figure 3.38: STM32CubeMX LoRa config. recommendation.

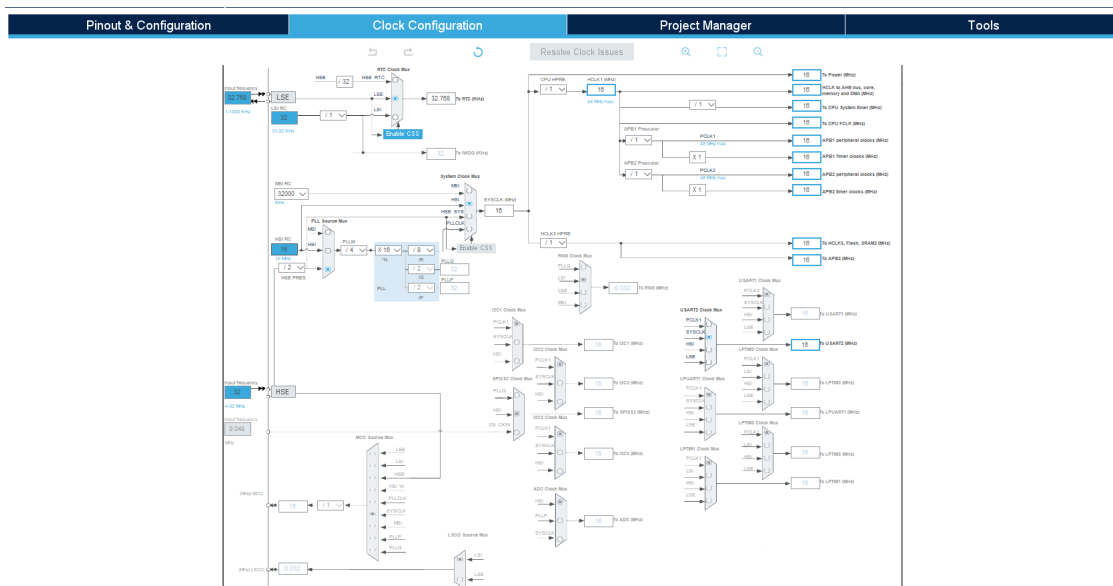


Figure 3.39: STM32CubeMX clock tree.

Project Manager

In this section, be sure that the following tickbox in the section **Code Generator** is checked in STM32CubeMX:

- add necessary library files as reference in the toolchain project configuration file.
- generate peripheral initialization as pair of '.c/.h' files per peripheral.
- keep user code when re-generating.
- delete previously generated files when not re-generating.

3.5.2 External files needed

Once the configuration file is completed, the whole end node skeleton code is generated simply by saving it. Notice that, as indicated by its name, this 'skeleton' code is almost empty or has some configurations that do not correspond to the desired ones. Firstly, some files must be manually added in order to have a working code. In fact, the end node skeleton is generated to take information about some hardware-specific configuration, called BSP: as stated in [23], there are two possibilities:

- implements the BSP functionalities directly in the generated code.
- implements the BSP functionalities by modifying the example code files and adding them as extra files after the code generation.

The second option has been chosen in order to prevent the user from dealing with this function and facilitate future design processes. The following text describes the needed hardware-dependent files that allows LoRaTo to work correctly.

stm32wlxx_nucleo_radio.h

This file is used to declare constants, GPIOs, and functions the RF driver uses. For the specific hardware of the proposed design, only the RF switch control pins must be modified as shown in the portion of code in listing 3.1.

Listing 3.1: RF BSP define

```

1
2 /** @defgroup STM32WLXX_NUCLEO_RADIO_LOW_LEVEL_RFSWITCH RADIO LOW
3     LEVEL RF SWITCH Constants
4 */
5

```

```

6 |
7 | #define RF_SW_CTRL1_PIN GPIO_PIN_13
8 | #define RF_SW_CTRL1_GPIO_PORT GPIOC
9 | #define RF_SW_CTRL1_GPIO_CLK_ENABLE() __HAL_RCC_GPIOC_CLK_ENABLE()
10 | #define RF_SW_RX_GPIO_CLK_DISABLE() __HAL_RCC_GPIOC_CLK_DISABLE()
11 |
12 |
13 | #define RF_TCXO_VCC_PIN GPIO_PIN_0
14 | #define RF_TCXO_VCC_GPIO_PORT GPIOB
15 | #define RF_TCXO_VCC_CLK_ENABLE() __HAL_RCC_GPIOB_CLK_ENABLE()
16 | #define RF_TCXO_VCC_CLK_DISABLE() __HAL_RCC_GPIOB_CLK_DISABLE()
17 | /**
18 | * @}
19 | */

```

This library calls other two libraries that must be inserted:

- **stm32wlxx_nucleo_errno.h.**
- **stm32wlxx_nucleo_conf.h.**

both can be taken from the *end node* example code provided by the example selector feature of STM32CubeIde.

stm32wlxx_nucleo.h

This library is not actually needed: it is included automatically when the source code is generated from the *ioc* file and is used to set some parameters and configuration only when the STM32WLxx nucleo board is used. Since the corresponding include statement is always generated from the CubeMX, there are two possibilities:

- Delete the include statement each time the **ioc** file updates the code.
- Create or import from the example code the library and leave it empty. If the library is imported, the unwanted configuration statement must be removed.

stm32wlxx_nucleo_radio.c

The code represented in listing 3.2 is the last "hardware dependent" file and implements the RF BSP function: a modified version of the file taken from the example code on STMCubeIde.

Listing 3.2: RF BSP functions

```

1 |
2 |
3 |
4 | /**

```

```

5  *****
6  * @file    stm32wlxx_nucleo_radio.c
7  * @author  MCD Application Team
8  * @brief   This file provides set of firmware functions to manage:
9  *          – RF circuitry available on STM32WLXX–Nucleo
10 *          Kit from STMicroelectronics
11 *****
12 * @attention
13 *
14 * Copyright (c) 2020–2021 STMicroelectronics.
15 * All rights reserved.
16 *
17 * This software is licensed under terms that can be found in the
18 * LICENSE file
19 * in the root directory of this software component.
20 * If no LICENSE file comes with this software, it is provided AS-IS
21 *
22 *****
23 */
24 /* Includes
25 _____*/
26 #include "stm32wlxx_nucleo_radio.h"
27 /** @addtogroup BSP
28 * @{
29 */
30
31 /** @addtogroup STM32WLXX_NUCLEO
32 * @{
33 */
34
35 /** @addtogroup STM32WLXX_NUCLEO_RADIO_LOW_LEVEL
36 * @brief This file provides set of firmware functions to Radio
37 *        switch
38 *        available on STM32WLXX–Nucleo Kit from STMicroelectronics.
39 * @{
40 */
41 /** @addtogroup STM32WLXX_NUCLEO_RADIO_LOW_LEVEL_Exported_Functions
42 * @{
43 */
44
45 /**
46 * @brief Init Radio Switch
47 * @retval BSP status
48 */
49 int32_t BSP_RADIO_Init(void)

```



```

50 {
51     GPIO_InitTypeDef  gpio_init_structure = {0};
52
53     /* Enable the Radio Switch GPIO Clock */
54     RF_SW_CTRL1_GPIO_CLK_ENABLE();
55
56     /* Configure the Radio Switch pin */
57     gpio_init_structure.Pin    = RF_SW_CTRL1_PIN;
58     gpio_init_structure.Mode  = GPIO_MODE_OUTPUT_PP;
59     gpio_init_structure.Pull  = GPIO_NOPULL;
60     gpio_init_structure.Speed = GPIO_SPEED_FREQ_VERY_HIGH;
61
62     HAL_GPIO_Init(RF_SW_CTRL1_GPIO_PORT, &gpio_init_structure);
63
64     /* RX channel is set as default state */
65     HAL_GPIO_WritePin(RF_SW_CTRL1_GPIO_PORT, RF_SW_CTRL1_PIN,
66                       GPIO_PIN_RESET);
67
68     return BSP_ERROR_NONE;
69 }
70 /**
71  * @brief DeInit Radio Switch
72  * @retval BSP status
73  */
74 int32_t BSP_RADIO_DeInit(void)
75 {
76     //RF_SW_CTRL3_GPIO_CLK_ENABLE();
77
78     //the switch is always ON so there is not a deinit function
79
80     return BSP_ERROR_NONE;
81 }
82
83 /**
84  * @brief Configure Radio Switch.
85  * @param Config: Specifies the Radio RF switch path to be set.
86  *             This parameter can be one of following parameters:
87  *             @arg RADIO_SWITCH_OFF
88  *             @arg RADIO_SWITCH_RX
89  *             @arg RADIO_SWITCH_RFO_LP
90  *             @arg RADIO_SWITCH_RFO_HP
91  * @retval BSP status
92  */
93 int32_t BSP_RADIO_ConfigRFSwitch(BSP_RADIO_Switch_TypeDef Config)
94 {
95     switch (Config)
96     {
97         case RADIO_SWITCH_OFF:

```

```
98     {
99         /*The switch is always powered, so this functionality is not
100         supported and the switch is only left on RX mode */
101         HAL_GPIO_WritePin(RF_SW_CTRL1_GPIO_PORT, RF_SW_CTRL1_PIN,
102         GPIO_PIN_RESET);
103         break;
104     }
105     case RADIO_SWITCH_RX:
106     {
107         /*Turns On in Rx the RF Switch */
108
109         HAL_GPIO_WritePin(RF_SW_CTRL1_GPIO_PORT, RF_SW_CTRL1_PIN,
110         GPIO_PIN_RESET);
111         break;
112     }
113     case RADIO_SWITCH_RFO_LP:
114     {
115
116         /*There is no LP channel, so this function is not supported and
117         the switch is left at default state*/
118         HAL_GPIO_WritePin(RF_SW_CTRL1_GPIO_PORT, RF_SW_CTRL1_PIN,
119         GPIO_PIN_RESET);
120         break;
121     }
122     case RADIO_SWITCH_RFO_HP:
123     {
124         /*Turns On in Tx High Power the RF Switch */
125
126         HAL_GPIO_WritePin(RF_SW_CTRL1_GPIO_PORT, RF_SW_CTRL1_PIN,
127         GPIO_PIN_SET);
128         break;
129     }
130     default:
131         break;
132 }
133 return BSP_ERROR_NONE;
134 }
135
136 /**
137  * @brief Return Board Configuration
138  * @retval
139  * RADIO_CONF_RFO_LP_HP
140  * RADIO_CONF_RFO_LP
```

```
141 | * RADIO_CONF_RFO_HP
142 | */
143 | int32_t BSP_RADIO_GetTxConfig(void)
144 | {
145 |     /*Set the configuration as HP */
146 |     return RADIO_CONF_RFO_HP;
147 | }
148 |
149 | /**
150 |  * @brief Get If TCXO is to be present on board
151 |  * @note  never remove called by MW,
152 |  * @retval
153 |  * RADIO_CONF_TCXO_NOT_SUPPORTED
154 |  * RADIO_CONF_TCXO_SUPPORTED
155 |  */
156 | int32_t BSP_RADIO_IsTCXO(void)
157 | {
158 |     return RADIO_CONF_TCXO_SUPPORTED;
159 | }
160 |
161 | /**
162 |  * @brief Get If DCDC is to be present on board
163 |  * @note  never remove called by MW,
164 |  * @retval
165 |  * RADIO_CONF_DCDC_NOT_SUPPORTED
166 |  * RADIO_CONF_DCDC_SUPPORTED
167 |  */
168 | int32_t BSP_RADIO_IsDCDC(void)
169 | {
170 |     return RADIO_CONF_DCDC_SUPPORTED;
171 | }
172 |
173 | /**
174 |  * @brief Return RF Output Max Power Configuration
175 |  * @retval
176 |  * RADIO_CONF_RFO_LP_MAX_15_dBm for LP mode
177 |  * RADIO_CONF_RFO_HP_MAX_22_dBm for HP mode
178 |  */
179 | int32_t BSP_RADIO_GetRFOMaxPowerConfig(
180 |     BSP_RADIO_RFOMaxPowerConfig_TypeDef Config)
181 | {
182 |     int32_t ret;
183 |     /* In Europe, the maximum allowed power is 14dBm*/
184 |     ret = RADIO_CONF_RFO_HP_MAX_14_dBm;
185 |
186 |     return ret;
187 | }
188 |
```

```

189 /**
190  * @}
191  */
192
193 /**
194  * @}
195  */
196
197 /**
198  * @}
199  */
200
201 /**
202  * @}
203  */

```

3.5.3 Test code

In order to test the functionalities of the LoRaTo PCB, the simple code shown in listing 3.3 has been implemented. The code is based on the example code *LoRaWAN_End_Node application* provided by STMicroelectronics and performs cyclically a sent uplink and a received downlink implementing a ping pong mechanism in which the byte received by the downlink message is sent back in the payload of the following uplink message.

In particular, the functions that has been written are:

- `LoRaWAN_Init()`.
- `OnRxData()`.
- `SendTxData()`.

and all the corresponding variables definition.

Listing 3.3: Test code

```

1  /* External variables
2  _____*/
3  /* USER CODE BEGIN EV */
4  uint8_t var_ping_pong = 0;
5  /* USER CODE END EV */
6  .
7  .
8  .
9
10

```

```

11 /* USER CODE BEGIN PV */
12 static uint8_t AppDataBuffer[LORAWAN_APP_DATA_BUFFER_MAX_SIZE];
13
14 /**
15  * @brief User application data structure
16  */
17 static LmHandlerAppData_t AppData = { 0, 0, AppDataBuffer };
18
19 /* USER CODE END PV */
20
21 .
22 .
23 .
24
25 void LoRaWAN_Init(void)
26 {
27     /* USER CODE BEGIN LoRaWAN_Init_LV */
28     uint32_t feature_version = 0UL;
29     /* USER CODE END LoRaWAN_Init_LV */
30
31     /* USER CODE BEGIN LoRaWAN_Init_1 */
32
33     /* Get LoRaWAN APP version*/
34     APP_LOG(TS_OFF, VLEVEL_M, "APPLICATION_VERSION: V%X.%X.%X\r\n",
35             (uint8_t)(APP_VERSION_MAIN),
36             (uint8_t)(APP_VERSION_SUB1),
37             (uint8_t)(APP_VERSION_SUB2));
38
39     /* Get MW LoRaWAN info */
40     APP_LOG(TS_OFF, VLEVEL_M, "MW_LORAWAN_VERSION: V%X.%X.%X\r\n",
41             (uint8_t)(LORAWAN_VERSION_MAIN),
42             (uint8_t)(LORAWAN_VERSION_SUB1),
43             (uint8_t)(LORAWAN_VERSION_SUB2));
44
45     /* Get MW SubGhz_Phy info */
46     APP_LOG(TS_OFF, VLEVEL_M, "MW_RADIO_VERSION: V%X.%X.%X\r\n",
47             (uint8_t)(SUBGHZ_PHY_VERSION_MAIN),
48             (uint8_t)(SUBGHZ_PHY_VERSION_SUB1),
49             (uint8_t)(SUBGHZ_PHY_VERSION_SUB2));
50
51     /* Get LoRaWAN Link Layer info */
52     LmHandlerGetVersion(LORAMAC_HANDLER_L2_VERSION, &
53     feature_version);
54     APP_LOG(TS_OFF, VLEVEL_M, "L2_SPEC_VERSION: V%X.%X.%X\r\n",
55             (uint8_t)(feature_version >> 24),
56             (uint8_t)(feature_version >> 16),
57             (uint8_t)(feature_version >> 8));
58
59     /* Get LoRaWAN Regional Parameters info */

```

```

59     LmHandlerGetVersion (LORAMAC_HANDLER_REGION_VERSION, &
feature_version);
60     APP_LOG(TS_OFF, VLEVEL_M, "RP_SPEC_VERSION:      V%X-%X.%X.%X\r\
n",
61             (uint8_t)(feature_version >> 24),
62             (uint8_t)(feature_version >> 16),
63             (uint8_t)(feature_version >> 8),
64             (uint8_t)(feature_version));
65
66
67     if (FLASH_IF_Init(NULL) != FLASH_IF_OK)
68     {
69         Error_Handler();
70     }
71
72     /* USER CODE END LoRaWAN_Init_1 */
73
74     UTIL_TIMER_Create(&StopJoinTimer, JOIN_TIME, UTIL_TIMER_ONESHOT,
OnStopJoinTimerEvent, NULL);
75
76     UTIL_SEQ_RegTask((1 << CFG_SEQ_Task_LmHandlerProcess), UTIL_SEQ_RFU
, LmHandlerProcess);
77
78     UTIL_SEQ_RegTask((1 << CFG_SEQ_Task_LoRaSendOnTxTimerOrButtonEvent)
, UTIL_SEQ_RFU, SendTxData);
79     UTIL_SEQ_RegTask((1 << CFG_SEQ_Task_LoRaStoreContextEvent),
UTIL_SEQ_RFU, StoreContext);
80     UTIL_SEQ_RegTask((1 << CFG_SEQ_Task_LoRaStopJoinEvent),
UTIL_SEQ_RFU, StopJoin);
81
82     /* Init Info table used by LmHandler*/
83     LoraInfo_Init();
84
85     /* Init the Lora Stack*/
86     LmHandlerInit(&LmHandlerCallbacks, APP_VERSION);
87
88     LmHandlerConfigure(&LmHandlerParams);
89
90     /* USER CODE BEGIN LoRaWAN_Init_2 */
91     /* USER CODE END LoRaWAN_Init_2 */
92
93     LmHandlerJoin(ActivationType, ForceRejoin);
94
95     if (EventType == TX_ON_TIMER)
96     {
97         /* send every time timer elapses */
98         UTIL_TIMER_Create(&TxTimer, TxPeriodicity, UTIL_TIMER_ONESHOT,
OnTxTimerEvent, NULL);
99         UTIL_TIMER_Start(&TxTimer);

```

```

100     }
101     else
102     {
103         /* USER CODE BEGIN LoRaWAN_Init_3 */
104
105         /* USER CODE END LoRaWAN_Init_3 */
106     }
107
108     /* USER CODE BEGIN LoRaWAN_Init_Last */
109
110     /* USER CODE END LoRaWAN_Init_Last */
111 }
112
113 /* USER CODE BEGIN PB_Callbacks */
114
115 /* USER CODE END PB_Callbacks */
116
117 /* Private functions
118 _____*/
119
120 /* USER CODE BEGIN PrFD */
121
122 /* USER CODE END PrFD */
123
124 static void OnRxData(LmHandlerAppData_t *appData, LmHandlerRxParams_t
125                    *params)
126 {
127     /* USER CODE BEGIN OnRxData_1 */
128
129     switch (appData->Port)
130     {
131     case LORAWAN_USER_APP_PORT:
132
133         var_ping_pong = appData->Buffer[0];
134         break;
135
136     default:
137         break;
138     }
139
140     /* USER CODE END OnRxData_1 */
141 }
142
143 static void SendTxData(void)
144 {
145     /* USER CODE BEGIN SendTxData_1 */
146
147     LmHandlerErrorStatus_t status = LORAMAC_HANDLER_ERROR;
148     UTIL_TIMER_Time_t nextTxIn = 0;
149     uint32_t i = 0;

```

```

147
148     if (LmHandlerIsBusy() == false)
149     {
150
151         AppData.Port = LORAWAN_USER_APP_PORT;
152
153         AppData.Buffer[i++] = var_ping_pong;
154
155         AppData.BufferSize = i;
156
157         status = LmHandlerSend(&AppData, LmHandlerParams.
158 IsTxConfirmed, false);
159         if (LORAMAC_HANDLER_SUCCESS == status)
160         {
161             APP_LOG(TS_ON, VLEVEL_L, "SEND REQUEST\r\n");
162         }
163         else if (LORAMAC_HANDLER_DUTYCYCLE_RESTRICTED == status)
164         {
165             nextTxIn = LmHandlerGetDutyCycleWaitTime();
166             if (nextTxIn > 0)
167             {
168                 APP_LOG(TS_ON, VLEVEL_L, "Next Tx in : ~%d second(s)\r\n
169 ", (nextTxIn / 1000));
170             }
171         }
172
173         if (EventType == TX_ON_TIMER)
174         {
175             UTIL_TIMER_Stop(&TxTimer);
176             UTIL_TIMER_SetPeriod(&TxTimer, MAX(nextTxIn, TxPeriodicity));
177             UTIL_TIMER_Start(&TxTimer);
178         }
179         var_ping_pong = 0;
180
181     /* USER CODE END SendTxData_1 */
182 }

```


Chapter 4

Prototype and tests

In fig. 4.1 and fig. 4.2, the 3D model and the assembled prototype, manufactured by *MD SRL*, are shown.

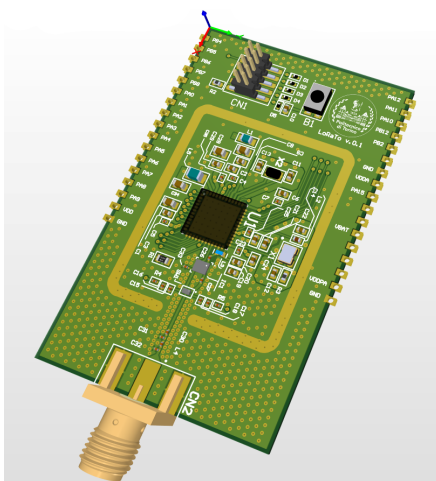


Figure 4.1: LoRaTo 3D model.

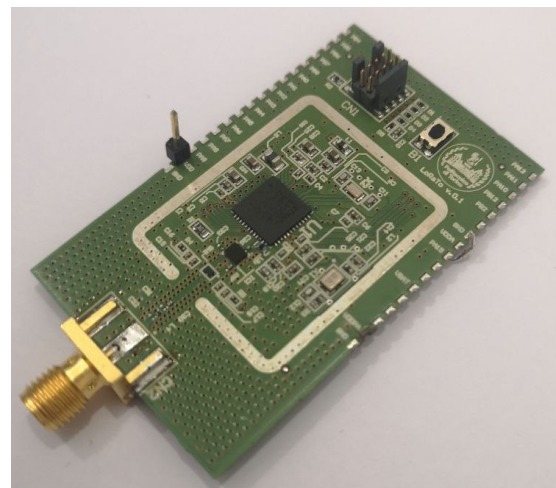


Figure 4.2: LoRaTo prototype.

LoRaTo is a 4-layer PCB of size 34 mm x 55 mm (lower than the dimension of the STM32WL55 Nucleo board, which is 64.5 mm x 70 mm), it has a JTAG connector that allow the programming and debugging through SWD protocol and a reset button on-board. All the microcontroller peripherals are accessible through external pads placed on both sides of the PCB, allowing the user to solder the board on other application-specific PCBs.

4.1 Software tests

4.1.1 Programming interface

The PCB has been connected by the SWD interface to an STLink-V3, a tool that provides programming and debugging features. In fig. 4.3 is shown the interface of STM32CubeProgrammer, which allows the user to read the microcontroller registers and information: it is possible to see that the board is accessible. Moreover, it is possible to perform READ/WRITE/ERASE operation on the microcontroller flash memory.

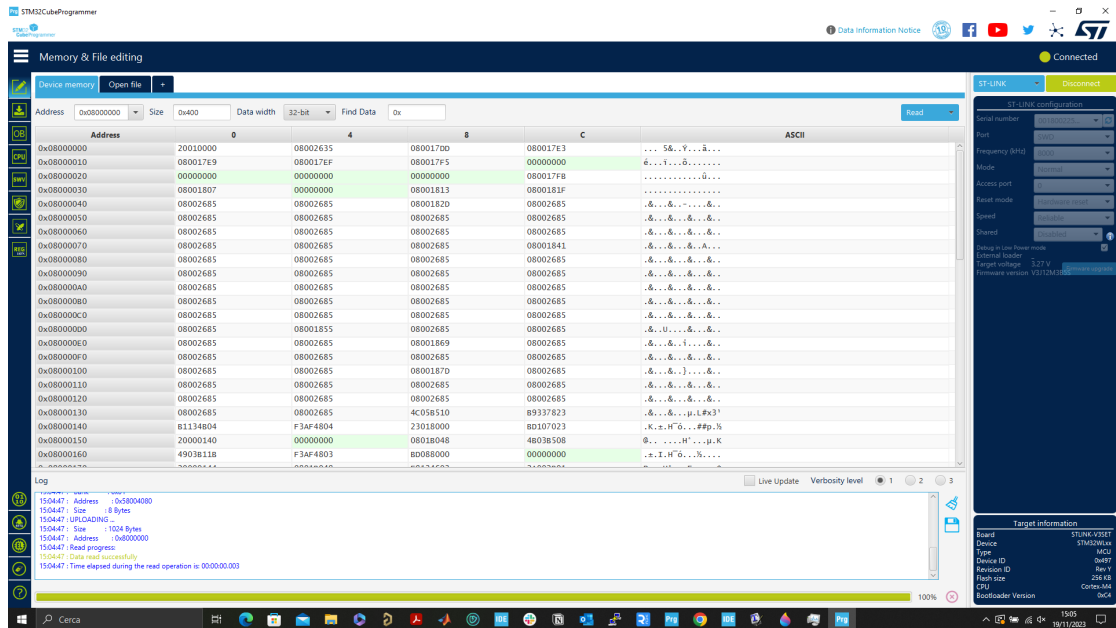


Figure 4.3: STM32CubeProgrammer.

4.1.2 LoRa class-A test

The PCB has been tested using the firmware discussed in chapter 3 to verify the correct behavior of LoRaTo when it implements the RF features. In particular, it is possible to see in fig. 4.4 the phases in which the end node joins the server and starts to send empty packets until a downlink is scheduled and then sends back the received message in the following uplink. In fig. 4.5, it is possible to see the current profile correspondent to a LoRa class-A cycle: the board absorbs a constant current with two different peaks that correspond to the uplink and downlink phases.

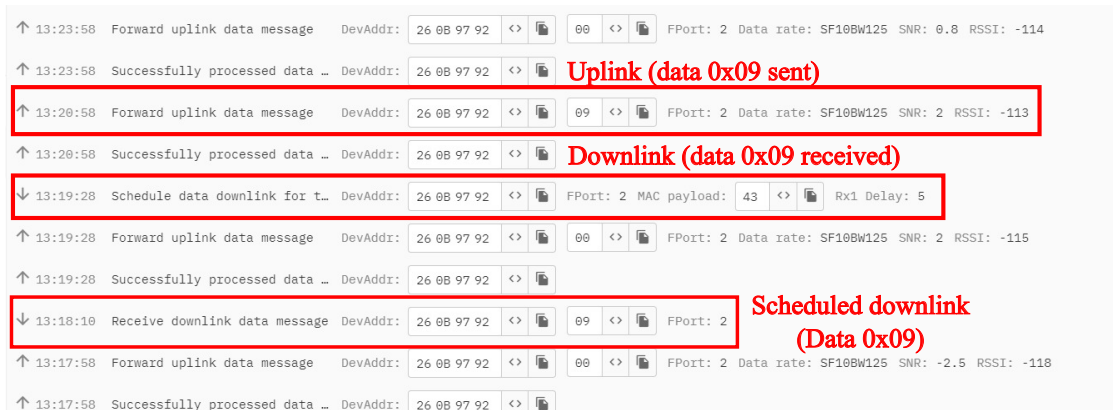


Figure 4.4: The Things Network live data interface.

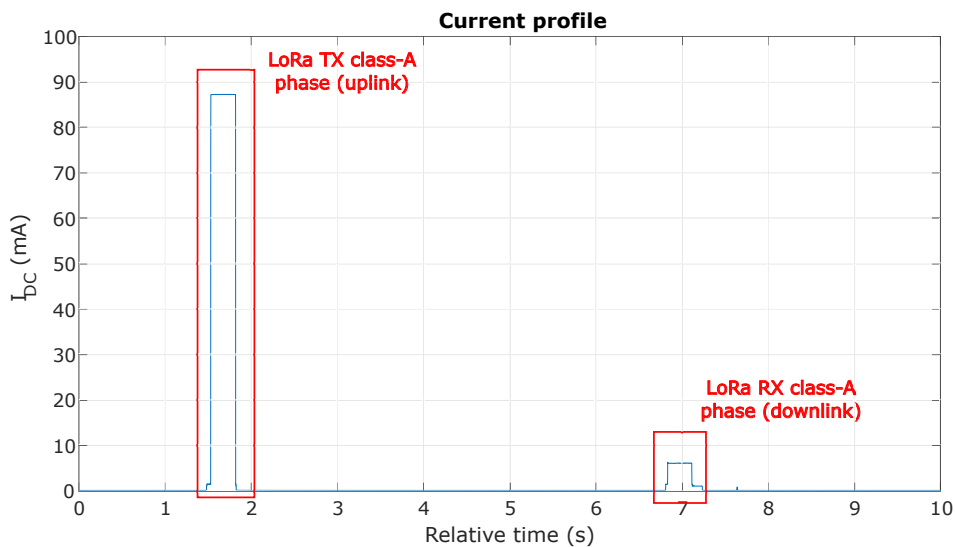


Figure 4.5: TX and RX real time current profile.

4.2 Output power measurements

In order to compare the achieved performances with respect to the nucleo-board ones, the output power of the RF signals has been measured using a spectrum analyzer, connecting the SMA connector of the board directly to the instrument: this test is directly correlated to the RF path design, since, if the board is well designed, the measured output power should match the nominal one. In fact, the RF output power is an essential characteristic of the system that allows the board to communicate properly across long distances. The measurements have been

performed for both LoRaTo and the nucleo-board, and the obtained results are shown in fig. 4.6, fig. 4.7 and table 4.1.

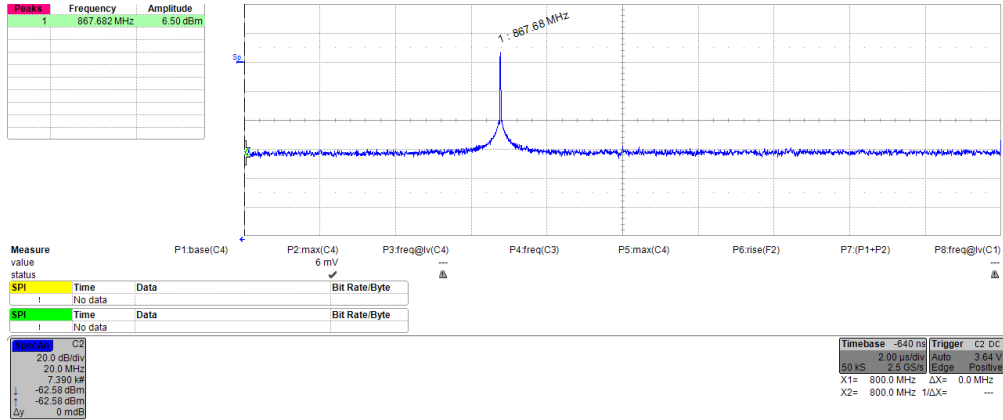


Figure 4.6: LoRaTo measured output power.

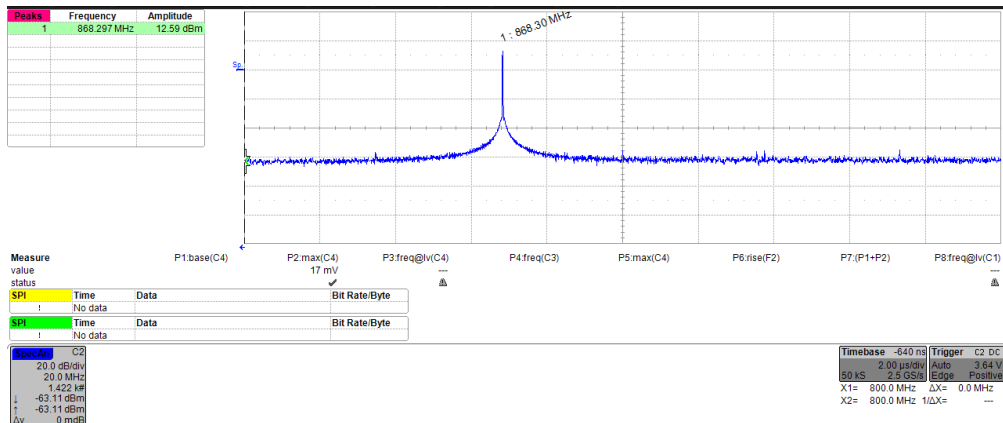


Figure 4.7: Nucleo-board measured output power.

Board	Nominal output power	Measured output power	Power loss
LoRaTo	14 dBm	6.50 dBm	7.50 dBm
Nucleo STM32WL55	14 dBm	12.59 dBm	1.41 dBm

Table 4.1: Output power measurements.

The LoRaTo board shows an evident power loss (7.50 dBm) that could be caused by different problems, such as the incorrect working of the balun IC or a not-perfect matching impedance on the RF track.

4.3 Qualitative LoRa coverage test

In order to obtain also qualitative results about the LoRaTo coverage performances, the LoRa class-A "ping-pong" firmware has been used to estimate the module's coverage in the urban area near the gateway. The same tests have been performed in the past using the Nucleo-WL55JC1 board and the same urban context to verify the downlink performances, the most critical operation for the LoRa transducer. In fig. 4.8 are shown the positions for which the test has been implemented, where the point **A** is the position of the gateway. Since the critical part of the LoRa class-A communication is the downlink, in table 4.2 the results are reported in terms of consecutive correctly received downlinks acquired by LoRaTo.

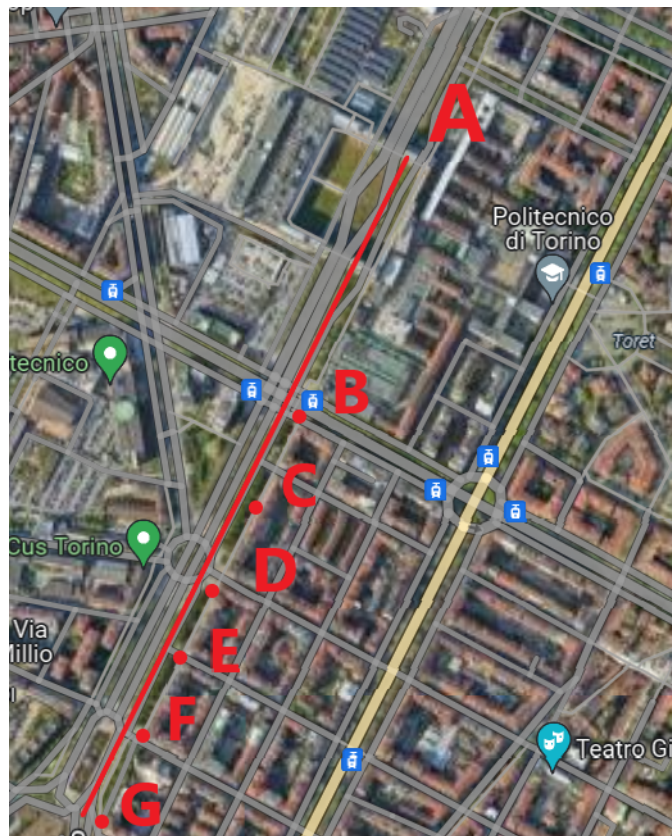


Figure 4.8: Stops of qualitative coverage test.

Distance	Distance (m)	N° of received downlink
A-B	392 m	3/3
A-C	495 m	3/3
A-D	630 m	2/3
A-E	720 m	3/3
A-F	830 m	2/3
A-G	970 m	3/3

Table 4.2: Qualitative downlink test results.

This test shows that despite the output power’s attenuation, the module can send and receive class-A LoRa packages in an urban context.

4.4 Future work

Among the tests that allow a complete characterization of the board, the following tests have yet to be performed:

TCXO test For preliminary tests, only the internal MSI has been used. The TCXO only works as an external system clock source when the LoRa stack is unused. This problem has not been solved since many possibilities should be considered:

- The firmware configuration could be more complex than the case of study of this thesis;
- The TCXO could not be designed properly; a prototype revision is needed in this case.

Ultra-Low-Power mode management This feature is directly correlated to the problem described in section 4.4 since the board doesn’t work when the *STOP2 mode* is used while the TCXO is chosen as system clock. A proper study on low power management must be performed to understand if it is correlated to the usage of the TCXO.

Power consumption characterization The board’s power consumption also depends on the low power configuration; for this reason, this test has not been performed since, as said in section 4.4, the low power management has to be implemented STM32CubeMX.

Chapter 5

Conclusion and future perspectives

The custom core module, LoRaTo, is a PCB based on the STM32WL55JC microcontroller designed and tested to work as a LoRa end node. It has been designed, starting from the target application specification, to reduce the footprint and cost, implementing only the essential sub-blocks that allow the board to communicate with LoRa.

The design process was divided in different parts. It was started drawing the electrical schematic design, which is strictly related to the application-specific characteristics: in this phase, all the components and the board configuration was defined depending on the LoRa-application needing. The hardware design is completed with the design of the board physical layout: in this phase, all the needed constraint that allow the LoRaTo board to work properly were considered, such as the RF block constraints that reduces the power loss, the physical sizing and division of the power island, and the signal routing of all the components present on board. The last design phase was related to the hardware-dependent firmware: a standard C code was developed and will be provided to future users to abstract the LoRa application-code from the LoRaTo hardware configuration; in this way, the user can build its application without the problem of dealing with the designed custom hardware. Finally, some tests has been performed in chapter 4, showing that LoRaTo works properly in almost all its functionalities, and the possibilities to send and receive data makes it a suitable element to be integrated into all the typical application in a precision agriculture context. However, there are additional goals that can be the starting point for future works:

- **TCXO behavior study:** as already mentioned, the microcontroller seems to be able to work with the MSI as system clock, and the TCXO is dedicated

only to the RF PLL. This behavior is not compliant with the one of the nucleo-board, so it is needed an investigation to determine if the TCXO is not well designed or if some firmware configurations has not been implemented yet:

- **Ultra-Low-Power modes:** the analysis done in this thesis has been performed considering the simple cases of the sleep mode. The stop2 mode functionalities must be implemented to reach lower power consumption (and also to deal with the TCXO issue), and an in-depth study must be performed on both the firmware implementation and current measurements;
- **RF output power:** the power measurements have shown a non-ideal output power level that can be caused by several factors. A study of the RF path impedance must be performed to achieve the impedance matching employing the Π net already set up along the RF path;
- **Implement of class-B and class-C features:** the two classes allow the device to receive downlink with different timing with respect to the class-A implementation. In particular, the LoRa class-B allow the end node to receive downlink at fixed intervals, regardless form the uplink phase, while the LoRa class-C allow to receive downlink at any time;
- **Tests in real use-case,** such as an upgraded version for WAPPFRUIT project, a smart irrigation system for orchards.
- **Multi-core implementation:** the STM32WL55CCU6 is a dual-core micro-controller, so it is possible to active both core for different tasks. If it is needed to work with the dual-core architecture, the firmware must be reconfigured taking into account the information given by the *application configuration* voice in **LoRaWAN application** windows of STM32CubeMX mentioned in section 3.5.1.

Bibliography

- [1] *Global Footprint Network*®. URL: <https://www.footprintnetwork.org/> (cit. on p. 1).
- [2] *Global Footprint Network*®. URL: https://data.footprintnetwork.org/?_ga=2.63520748.297969210.1697644145-269066249.1696236439#/ (cit. on p. 2).
- [3] Davide Gisolo et al. «Wappfruit: a project for the oprimisation of water use in agriculture». In: (Mar. 2023), pp. 1–2 (cit. on p. 1).
- [4] Mattia Barezzi, Umberto Garlando, Francesca Pettiti, Luca Nari, Davide Gisolo, Davide Canone, and Danilo Demarchi. «Long-Range Low-Power Soil Water Content Monitoring System for Precision Agriculture». In: (Mar. 2022), pp. 3–4 (cit. on pp. 1, 3).
- [5] Mattia Barezzi, Francesca Pettiti, Luca Nari, Davide Gisolo, Davide Canone, Danilo Demarchi, and Umberto Garlando. «Long-Range Low-Power Electronic System for Drip Irrigation in Precision Agriculture». In: (Dec. 2023) (cit. on pp. 1, 3).
- [6] *Multiprotocol LPWAN dual core 32-bit Arm®Cortex®-M4/M0+ LoRa®, (G)FSK, (G)MSK, BPSK, up to 256KB flash, 64KB SRAM*. STMicroelectronics. 2022 (cit. on pp. 4, 13, 14, 20).
- [7] *Reference designs for STM32WL5x and STM32WLEx microcontrollers*. STMicroelectronics (cit. on pp. 12, 19).
- [8] *The Things Network - Antenna Connectors*. URL: <https://www.thethingsnetwork.org/docs/lorawan/antenna-connectors/> (cit. on p. 15).
- [9] **AN5457** *RF matching network design guide for STM32WL Series*. STMicroelectronics (cit. on pp. 16–19).
- [10] **BGS12PL6** *General purpose RF CMOS power SPDT Switch in ultra small package with 0.77mm² footprint*. Infineon Technologies AG (cit. on pp. 16, 17).

- [11] **BALFHB-WL-02D3** 50 Ω nominal input / conjugate matched balun to QFN-4L STM32WL in high power mode, 862-928 MHz with integrated harmonic filter. STMicroelectronics (cit. on pp. 18, 19, 26–29).
- [12] **AN2867** Oscillator design guide for STM8AF/AL/S, STM32 MCUs and MPUs. STMicroelectronics (cit. on pp. 21, 31, 32).
- [13] **ECS-TXO-25CSMV**. ECS Inc. (cit. on p. 21).
- [14] **NX2012SA**. NDK America, Inc. (cit. on p. 22).
- [15] *Crystal Load Capacitance*. URL: <https://suntsu.com/engineering-services/suntsu-application-notes/crystal-load-capacitance/#:~:text=Measuring%20stray%20capacitance%20is%20also,ranges%20from%202pF%20to%208pF>. (cit. on p. 22).
- [16] **FTSH-105-01-L-DV-K-TR**. Samtec Inc. (cit. on p. 23).
- [17] **ESDALC6V1-1U2**. STMicroelectronics. (cit. on p. 23).
- [18] **AN5612** ESD protection of STM32 MCUs and MPUs. STMicroelectronics (cit. on p. 23).
- [19] **S300-10M**. Masach Tech Ltd. (cit. on p. 23).
- [20] *APPA5.02 SGW2828 LoRa Module RF Design and PCB Layout Guideline*. SGWireless (cit. on p. 26).
- [21] **AN5407** Optimized RF board layout for STM32WL Series. STMicroelectronics (cit. on pp. 26, 29–31).
- [22] **MD SRL** - tech info. URL: <https://www.mdsrl.it/mddesignrules.html/> (cit. on p. 33).
- [23] **AN5406** How to build a LoRa® application with STM32CubeWL. STMicroelectronics (cit. on p. 52).