SLB-TTC-DOC-v1.2

TTC Documentation

TTC Documentation
SpaceLab, Universidade Federal de Santa Catarina, Florianópolis - Brazil

January 2021

TTC Documentation
January, 2021

Project Chief:
Eduardo Augusto Bezerra

Authors:
Gabriel Mariano Marcelino

Contributing Authors:
Sara Vega Martinez

Revision Control:

Version	Author	Changes	Date
1.2	G. M. Marcelino	Template update	2021/01/16

List of Figures

List of Tables

Nomenclature

ADC Analog-To-Digital Converter.

BSL Bootstrap Loader.

CDS CubeSat Design Specification.

CPU Central Processing Unit.

DMA Direct Memory Access.

GFSK Gaussian Frequency-Shift Keyring.

GPIO General Purpose Input/Output.

HAL Hardware Abstraction Layer.

I^2C Inter-Integrated Circuit.

ISR Interruption Service Routine.

OBC Onboard Computer.

OBDH Onboard Data Handling.

P-POD Poly-Picosatellite Orbital Deployer.

PCB Printed Circuit Board.

RAM Random Access Memory.

RF Radio Frequency.

SPI Serial Peripheral Interface.

TMR Telemetry, Tracking and Command Module Requirements.

TTC Telemetry, Tracking and Command.

UART Universal Asynchronous Receiver/Transmitter.

USB Universal Serial Bus.

Contents

CHAPTER 1

Introduction

This document is the general documentation of the Telemetry, Tracking and Command module of the FloripaSat project.

The TTC (or TT&C) is the communication module of the FloripaSat[1] cubesat. It is responsible to make the communication between the earth (A ground station) and the satellite, and is divided in two sub-modules: Beacon and telemetry.

The beacon is a independent sub-module who transmits a periodic signal containing an identification data (ID) of the satellite and some basic telemetry data.

The telemetry sub-module is the main communication device. It has a bidirectional data link to receive telecommands from the earth and transmit all the requested data.

The telemetry sub-module is controlled by the OBDH module (The OBC of the satellite), and the software for handling with this sub-module is under development in the OBDH module, and is not documented here.

[2]

The beacon executes the following tasks during its execution:

- 145 MHz band antenna deployment.
- One-way communication (RX) with the EPS module, using the FSP protocol.
- Two-way communication (TX and RX) with the OBDH module, using the FSP protocol.
- Transmission of the beacon packets (In two protocols: NGHam and AX.25), containing the data from the EPS or the OBDH module and the satellite ID ("FLORIPASAT").
- When required by the OBDH module, the transmissions are stopped for an hibernation period (shutdown).
- In case of an critical failure of the OBDH module, or the uplink channel, the beacon activates its reception and is capable of make its own hibernation (shutdown).

1.1 Module Requirements

In the list below, the TTC module requirements for the mission are described. These requirements are nominated as TMR, or Telemetry, Tracking and Command Module Requirements.

TMR 1 The FloripaSat shall have a physical device to inhibit radio frequency (RF) transmission.

Compliance with CDS 3.3.9: The use of three independent inhibits is highly recommended and can reduce required documentation and analysis.

TMR 2 The CubeSat will have the RF power output to the transmitting antenna input no greater than 1,5 W.

Compliance with CDS 3.3.9.1.

TMR 3 The CubeSat will have the RF power output to the transmitting antenna input no less than 1,0 W (or 30 dBm).

Defined by team analysis.

TMR 4 No CubeSats shall generate or transmit any RF signal from the time of integration into the P-POD through 45 minutes after on-orbit deployment from the P-POD.

Compliance with CDS.

TMR 5 TTC transceiver shall transmit and receive on the frequency of 437,9 Mhz.

Defined by the team, based on available spectrum allocation to Amateur communication.

TMR 6 TTC beacon shall transmit on the frequency of 145,9 Mhz.

Defined by the team, based on available spectrum allocation to Amateur communication.

TMR 7 TTC shall modulate and demodulate information using GFSK.

Defined by the team.

TMR 8 TTC Beacon must transmit periodic beacon messages at an interval of 10 seconds, except when in hibernation or shutdown mode. Allows ground stations to track and receive satellite data even if telecommand was not sent to the satellite.

TMR 9 TTC transceiver must receive signals from ground stations and demodulate them.

TMR 10 TTC must interface with OBDH, exchanging encoded raw data received or to be transmitted.

TMR 11 TTC must interface with OBDH using the SPI protocol (@2 KHz).

Defined by the team.

TMR 12 TTC radio must modulate raw data received from OBDH using GFSK, prior to transmission, and demodulate received data and forward raw data to OBDH.

TMR 13 TTC transceiver shall transmit and receive data at a baud rate of 2400 bps. Defined by the team, based on link budget analysis.

TMR 14 TTC beacon shall transmit data at a baud rate of 1200 bps.

Defined by the team, based on link budget analysis.

TMR 15 TTC beacon shall transmit packets using the NGHam and AX.25 protocols.

Defined by the team.

TMR 16 A same beacon packet must be transmitted in both NGHam and AX.25 protocols.

Defined by the team.

TMR 17 TTC must receive the batteries voltages from the EPS module at every 10 seconds.

Defined by the team.

TMR 18 The payload from the packets transmitted by the beacon must contain at least the satellite ID ("FLORIPASAT") and batteries voltages (received from the EPS module at every 10 seconds).

TMR 19 TTC uC must perform the antenna deployment of the VHF band antenna.

Defined by the team.

TMR 20 Between the beacon packets transmissions, the beacon MCU and radio must operate in low power mode, to save energy.

Defined by the team.

TMR 21 TTC PAs (Power amplifiers) must just only be activated during transmissions. When they are not in operation, they must be turned off.

Defined by the team.

TMR 22 All the beacon critical data, like time control and antenna deployment status, must be stored in a non-volatile memory.

Defined by the team.

TMR 23 TTC beacon must be able to receive a 24 hour shutdown command from the OBDH module.

Compliance with AMSAT/IARU regulations.

CHAPTER 2

Hardware

The TTC board is composed by the following main components:

- MSP430F6659, as the beacon microcontroller.
- RF4463F30, as the radio module for the beacon and the telemetry link.

In the figure 2.1, ...

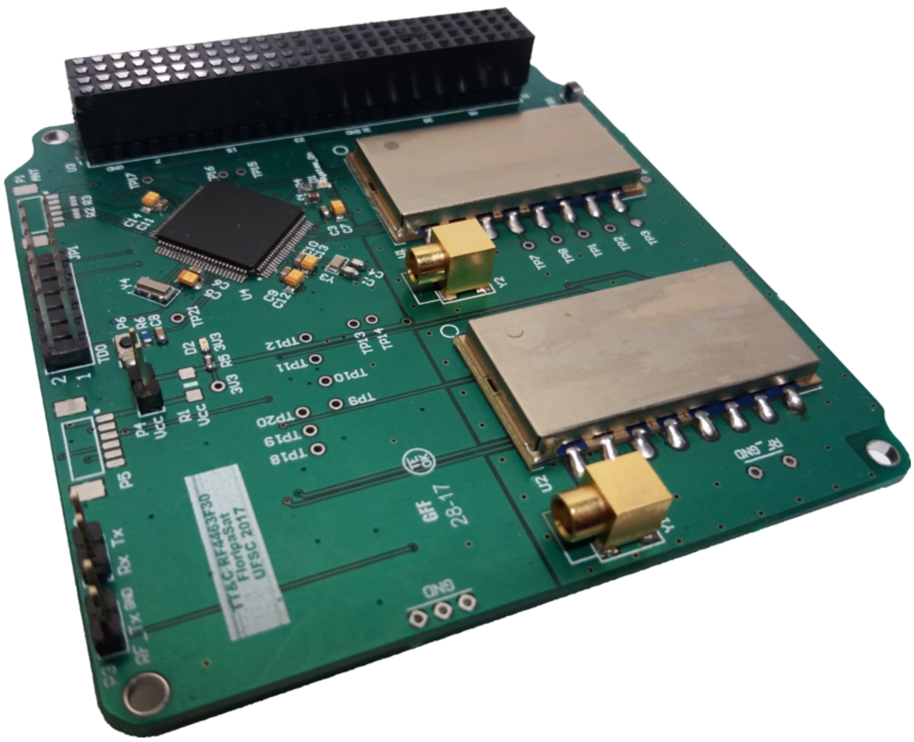


Figure 2.1: TTC PCB.

2.1 General Diagram

In the figure 2.2, a general hardware diagram can be seen, with the connection and protocols between the components.

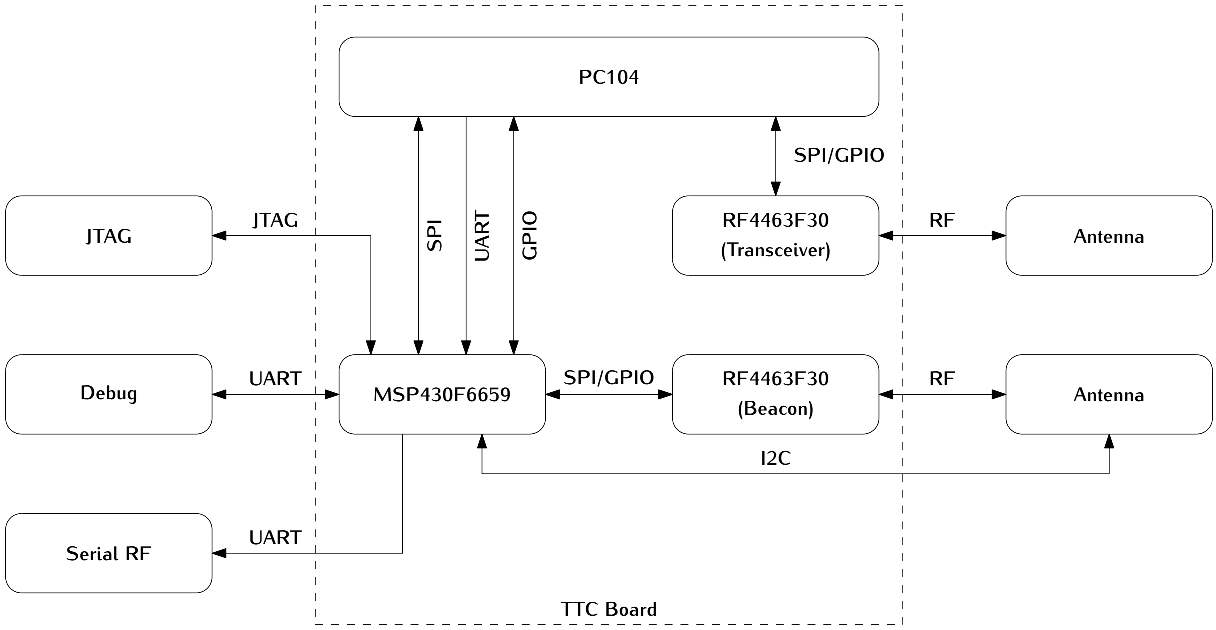


Figure 2.2: Hardware diagram of the TTC module.

2.2 Main Components

2.2.1 Microcontroller

The beacon microcontroller is the MSP430F6659IPZR [3]. Its main characteristics can be found in the table 2.1.

2.2.2 Radio Modules

The NiceRF RF4463F30 [4] is a transceiver module based on the Silicon Labs Si4463 [5] radio. This module also contains a PA module to increase the output power up to 31 dBm.

In compliance with the *TMR 3*, the RF4463F30 module operates with 5 V as input voltage in the TTC board, to achieve 30 dBm (1 W) in its output (The RF connector).

Si4463

The RF4463F30 module uses the Si4463 radio module. The main characteristics of this IC can be found in the table 2.2.

2.3 External Connections

This section describes the external available connections of the TTC module.

In the figure 2.3, all the external connections are enumerated.

A brief description of each connection is presented in the table 2.3.

The connections 1, 2, 4 and 6 were designed to be used during the software development stage, and not during the satellite operation.

Characteristic	*Value*
CPU	MSP430
Frequency	Up to 20 MHz
Non-volatile memory	512 kB
RAM	66 kB
GPIO pins	74
I^2C	3
SPI	6
UART	3
DMA	6
ADC	ADC12-12ch
Comparators	12 inputs
Timers - 16-bit	4
Multiplier	32 × 32
BSL	USB
Min V_{cc}	1,8 V
Max V_{cc}	3,6 V
Active Power	360 $\mu A/MHz$
Standby Power (LMP3)	$2,6\ \mu A$
Wakeup Time	3 μs
Operating Temperature Range	-40 to 80 $°C$

Table 2.1: MSP430F6659 features.

Characteristic	*Value*	*Unit*
Frequency range	119-1050	MHz
Receiver sensitivity	-126	dBm
Modulation	(G)FSK, 4(G)FSK, (G)MSK and OOK	-
Max. output power	+20	dBm
PA support	+27 to 30	dBm
Ultra low current powerdown modes	30 (shutdown), 50 (standby)	nA
Data rate	100 bps to 1 Mbps	-
Power supply	1,8 to 3,6	V
TX and RX FIFOs	64 bytes for each or 129 bytes shared	-

Table 2.2: Si4463 features.

2.3.1 PCI-104 Pins

The table 2.4 describes the PCI-104 connector used pins. The first column is the row number of the connector, and the remaining columns are the respective columns (Named as H1A, H1B, H2A and H2B respectively). If the pin has no description, it is not connected to the TTC board.

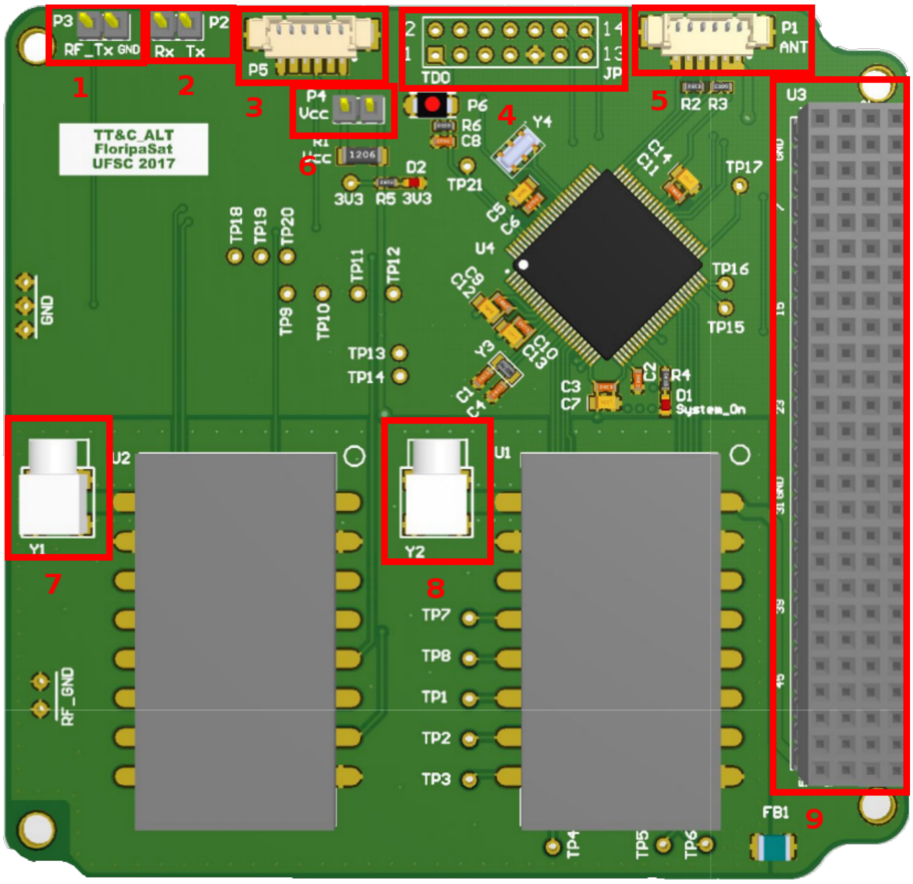


Figure 2.3: External connections on the board.

2.4 PCB

The PCB (Printed Circuit Board) of the TTC module has basically the MSP430F6659 ic, the RF4463F30 module and all the external connectors.

Some characteristics of this PCB are described bellow:

- The components and traces of the board are distributed over two layers (one in each side of the board).
- The RF modules are placed in a isolated ground plane (Connected to the main GND using a ferrite bead). This region can be seen in the figure 2.4.
- The antenna output of the RF4463F30 modules are connected to their respective connector over a 50 Ohm coupled trace.

2.4.1 Types of assembly

The TTC PCB has two types of assembly: Develop and flight.

The develop type has the purpose to be used during the development stage and tests. In this version, the connectors 3 and 5 are not soldered (Using the figure 2.3 as reference).

The flight version were designed to be used during the satellite operation. In this version, the connectors 1, 2, 4 and 6, and the reset button, are not soldered.

Number	*Connector*	*Description*
1	Male pin header (1 × 2)	UART TX @4800 bps. These pins transmit the beacon packets over a serial connection (It is enable in the configuration file, setting the BEACON_RADIO variable as UART_SIM).
2	Male pin header (1 × 2)	Debug UART TX/RX @115200 bps. These pins transmit a description of the main events of the beacon software during it's execution. This feature is only available in DEBUG_MODE.
3	Male PicoBlade^{TM} (×6)	JTAG and Debug. This connection contains the relevant pins of the connectors 2 and 4.
4	Male pin header (2 × 2)	MSP430 JTAG. This connection is for programming the uC code, using a MSP-FET debugger.
5	Male PicoBlade^{TM} (×6)	Antenna I2C. I2C bus for a communication channel with the antenna module.
6	Male pin header (1 × 2)	Power supply jumper. With a jumper, the beacon microcontroller power source comes from the JTAG connector. Without a jumper, the uC power supply comes from a pin of the PC104 connector.
7	Female Angled MCX	437 MHz band RF signal (Goes to the antenna module).
8	Female Angled MCX	145 MHz band RF signal (Goes to the antenna module).
9	Male/Female PCI-104	PCI-104. Power supply and communication buses with others stacked up modules.

Table 2.3: External connections description.

2.5 Power Budget

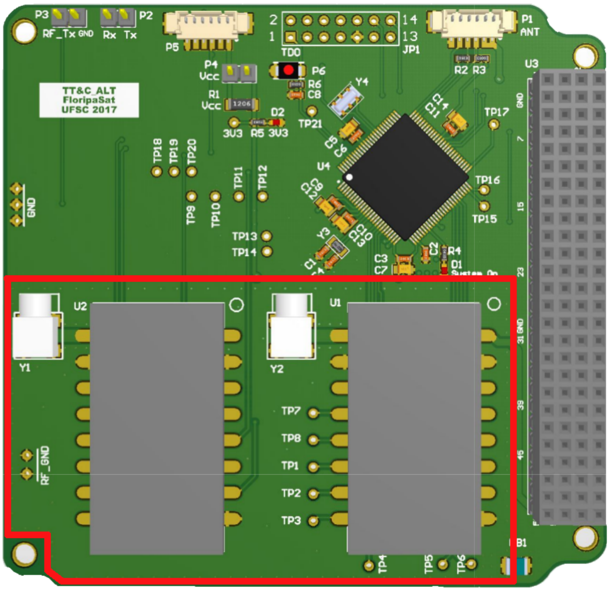


Figure 2.4: RF ground plane region (red) of the TTC board.

Functionality	Components	Quantity	Current [A]		Power [W]		Notes
			Min.	Max.	Min.	Max.	
Microcontroller	MSP430F6659	1	0,155m	0,0031	0,5115m	0,01023	4 MHz, Flash. Datasheet p.54
	XT1 (Crystal 1 - 32,768 kHz)	1	$\cong 0$	0,00029	$\cong 0$	0,000957	Datasheet p. 60
	XT2 (Crystal 2 - 4 MHz)	1	$\cong 0$	0,0002	$\cong 0$	0,00066	Datasheet p. 61
Total			**0,155m**	**0,00359**	**0,5115m**	**11,85μ**	Beacon/EPS DC/DC 3V3/2A

Figure 2.5: Static power budget of the TTC module.

Functionality	Components	Quantity	Between transmissions				During transmissions			Power			Notes
			Activation Period [s]	Active Time [s]	Current [A]	Charge [As]	Active Time [s]	Current [A]	Charge [As]	Min.	Max.	Mean	
Radio	RF4463F30	1	10	9,9	0,5μ	49,52μ	0,1	0,6	0,058	17μ	1,98	0,01916	$\cong$ 50 % efficiency

Figure 2.6: Dynamic power budget of the TTC module.

Row	*H1A*	*H1B*	*H2A*	*H2B*
1	GND	GND	GND	GND
2	GND	GND	GND	GND
3	-	-	UART RX @4800 bps from the EPS module.	-
4	Telemetry radio GPIO0	Telemetry radio GPIO1	-	-
5	Telemetry radio GPIO2	Enable beacon radio power supply	-	-
6	Telemetry radio SDN	-	OBDH communication (SPI MOSI)	OBDH communication (SPI clock)
7	-	-	OBDH communication (SPI chip select)	OBDH communication (SPI MISO)
8	-	-	-	-
9	-	-	-	-
10	-	-	-	-
11	-	-	-	-
12	-	-	-	-
13	-	-	-	-
14	-	-	Beacon uC power supply (3,3 V/50 mA)	3,3 V beacon uC power supply (3,3 V/50 mA)
15	GND	GND	GND	GND
16	GND	GND	GND	GND
17	-	-	-	-
18	Telemetry radio SPI clock	-	-	-
19	Telemetry radio SPI MISO	-	-	-
20	Telemetry radio SPI MOSI	Telemetry radio SPI chip select	-	-
21	-	-	-	-
22	-	-	-	-
23	-	-	-	-
24	-	-	-	-
25	Telemetry radio power supply (5 V/500 mA)	-	-	-
26	Beacon radio power supply (5 V/500 mA)	-	-	-

Table 2.4: PCI-104 connector reference.

CHAPTER 3

Software

The beacon software is responsible to transmit periodic beacon signals, containing the satellite identification and a basic telemetry data.

To achieve it, this software communicates with other modules of the satellite (to acquire data to transmit in the beacon packets), controls the beacon's radio module and its antenna module.

It's written in C, using the Code Composer Studio IDE (Version 7.4.0). The radio module (Si4463 IC) is configured using the WDS (Version 3.2.11), a step-by-step tutorial is available as an apendix.

3.1 Dependencies

The beacon software is dependent of the following libraries:

- NGHam: Used to generate and interpret the NGHam protocol packets (This library was modified for this firmware).
- DriverLib: Used to handle the internal peripherals of the MSP430F6659 uC.

3.2 Software Layers

The figure 3.1 describes the software layers of the beacon system.

It is composed by five layers:

- Hardware layer: All the TTC hardware (MSP430F6659 and the RF4463F30).
- Drivers layer: The software to make the interface with the hardware (Internal peripherals of the uC, the radio module and the antenna module).
- System layer: General usage functions and resources of the beacon system (like data structures, power management, etc.).
- Application layer: The main beacon software, where all the tasks were implemented.

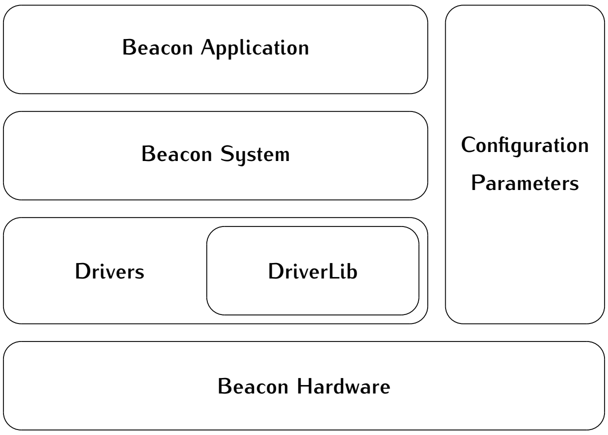


Figure 3.1: Beacon software stack-up.

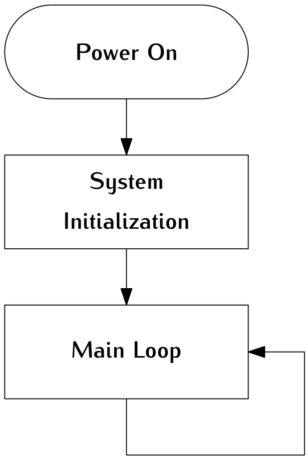


Figure 3.2: Main flowchart of the beacon software.

3.3 Flowcharts

3.3.1 Main

3.3.2 Initialization

Watchdog

The watchdog is initialized using the ACLK as clock source and a 512k clock divider (generating a 16 second watchdog). The counter starts right after the initialization.

CPU

For the CPU configuration, the core voltage is setted to level 2 (required for a clock between 12 and 20 MHz). The DCO FLL reference is setted to REF0 (with the clock divider equal to 1). The ACLK clock is setted to be equal to REF0 (same as DCO, clock divider equal to 1). The SMCLK reference is setted to be the DCO, but with a divider factor equal to 4.

So, the final configuration is: MCLK = 16 MHz, SMCLK = 4 MHz and ACLK = 32,768 kHz.

Memory

UNDER DEVELOPMENT.

Time

During the time initialization, the last value stored in the memory is loaded and the counter continues from it value, preventing the lost of time reference after a system reboot or fault.

Timer

For the system time, a 1 second timer is used (using the timer interruption to increment a second counter variable, 32-bit unsigned integer). This counting starts from the last value stored into the memory.

Antenna

UNDER DEVELOPMENT.

EPS Communication Bus

The EPS communication bus (UART) is configured at a transfer rate of 4800 bps, no parity bit, LSB first and one stop bit. The clock source of this UART bus is the SMCLK. The EPS queue is also initialized.

OBDH Communication Bus

The same occurs for the OBDH communication bus (SPI): the transfer rate is 2400 bps and the endianness is MSB first. The OBDH queue is also initialized.

Radio

During the initialization of the radio, it is reseted and all the configuration parameters are transfered to the radio module. More informations about this process can be found in the radio IC documentation.

RF Protocols

The NGHam, AX.25 and the FSP protocols also need an initialization, to set internal variables and counters.

3.3.3 ISRs

OBDH and EPS Communication

The flowchart of the interruption service routine (ISR) of the OBDH communication can be seen in the figure 3.4(a). Basically, when a new byte arrives, it is pushed to a queue (if it isn't full), and periodically, the OBDH communication task process these new bytes (taking out the bytes from the queue).

The flowchart of the interruption service routine (ISR) of the EPS communication can be seen in the figure 3.4(b). It's operation is equal to the OBDH communication ISR.

3.4 Operation Modes

There are three operation modes in the beacon software:

- DEBUG MODE: In this mode, the operation of the beacon is described through an UART port. At every operation, a message describing what is happening is transmitted to the debug UART port.
- TEST MODE: In this mode, the antenna deployment routines are not executed. This is the main mode to use during the satellite integration tests.
- FLIGHT MODE: This is the mode to use in flight, with all the available resources enabled.

The selection of the operation mode can be done using the "BEACON_MODE" variable in the "config.h" file.

3.5 Tasks

In the table 3.1, the beacon tasks are listed with a brief description and its period of execution.

3.6 Packets Payload

In the normal satellite operation, the beacon packets contains the data from the table 3.2.

If a fault on the OBDH module occurs, only the EPS data are transmitted, the contents of this kind of packet can be seen in the table 3.3.

If a fault occurs in the OBDH and the EPS modules, only the satellite ID is transmitted. It can be seen in the table 3.4.

The decodification of the packets data are done automatically by the GRS software.

3.7 USCIs Configuration

3.8 Timers

Task	*Description*	*Period*
System initialization	Submodules initialization (CPU, memory, radio, etc.)	Aperiodic (Executed only at initializations)
Antenna deployment	145 MHz band antenna deployment after the satellite launch	Aperiodic (Executed once)
NGHam packet transmission	Transmission of the beacon packets with a basic telemetry data using the NGHam protocol	10, 20 or 30 seconds (EPS energy level dependent)
AX.25 packet transmission	Transmission of the beacon packets with a basic telemetry data using the AX.25 protocol	10, 20 or 30 seconds (EPS energy level dependent)
OBDH data processing	The OBDH module can send telemetry data or commands	Aperiodic (OBDH module dependent)
EPS data processing	The EPS modules sends telemetry data	Aperiodic (EPS module dependent)
Radio data processing	Processing of an incoming packet from the beacon radio	Aperiodic (Only when the OBDH fails and an hibernation is required)
Beacon radio reception activation	When a critical failure occur in the OBDH module, the beacon activates its reception between the beacon transmissions	Aperiodic (Only when the OBDH module fails)
Radio reset	Beacon radio reset	10 minutes
Beacon reset	Beacon system reset	12 hours

Table 3.1: Beacon software tasks.

Information	*Length (Bytes)*
Satellite ID: “FLORIPASAT"	10
Batteries voltages	4
Batteries temperatures	6
Total charge of batteries	2
Solar panels currents	12
Solar panels voltages	6
Overall status of the satellite	2
Accelerometer and gyroscope	12
Time since boot	4
Number of OBDH module resets since launch	2

Table 3.2: Normal content of the beacon packets.

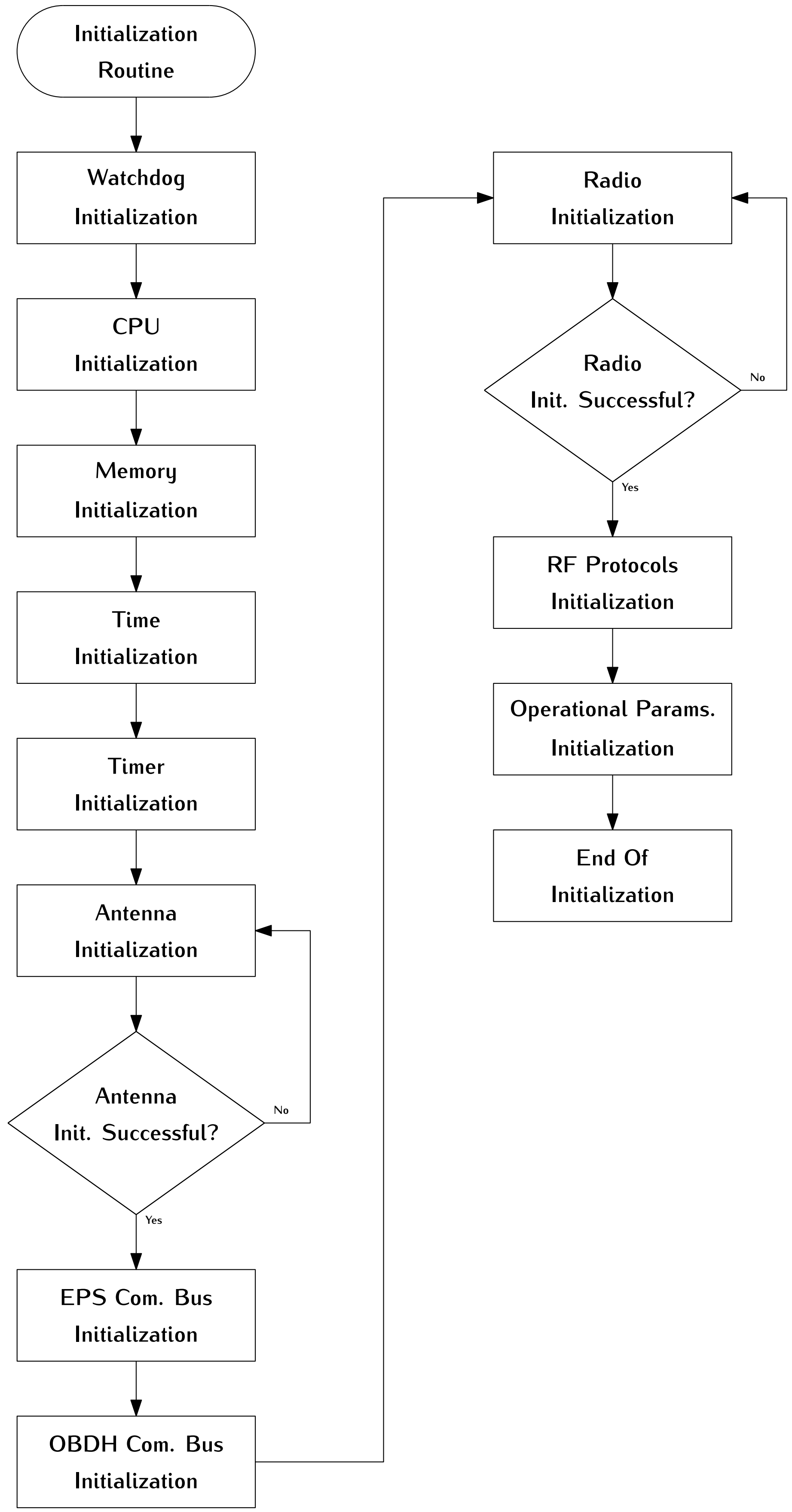
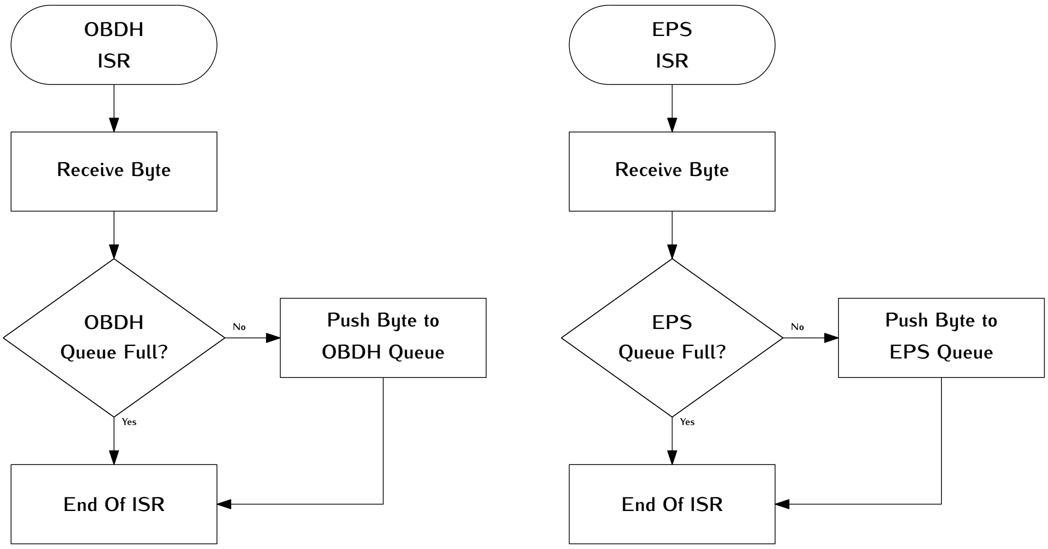


Figure 3.3: Flowchart of the beacon initialization.



(a) OBDH communication ISR flowchart.

(b) EPS communication ISR flowchart.

Figure 3.4: OBDH and EPS modules comunication ISRs routines.

Information	*Length (Bytes)*
Satellite ID: “FLORIPASAT"	10
Batteries voltages	4
Batteries temperatures	6
Total charge of batteries	2
Solar panels currents	12
Solar panels voltages	6
Energy level	1

Table 3.3: Content of the beacon packets with a fault in the OBDH module.

Information	*Length (Bytes)*
Satellite ID: “FLORIPASAT"	10

Table 3.4: Content of the beacon packets with a fault in the OBDH and EPS modules.

MSP Interface	*Mode*	*Connected Components*
USCI_A0	UART RX	EPS Bus
USCI_A0	UART TX	Packets transmission (Only for tests)
USCI_A1	UART TX/RX	Debug
USCI_A2	SPI Slave	OBDH Bus
USCI_B0	SPI Master	Beacon transceiver
USCI_B2	I^2C Master	Antenna bus

Table 3.5: USCIs configuration of the beacon microcontroller.

Timer Interface	*Mode*	*Period*	*Function*
TIMER_A1	Continuous-Compare	1 s	Time Control

Table 3.6: Timers configuration of the beacon microcontroller.

CHAPTER 4

Tests

This...

4.1 RF Signal Frequency

4.1.1 Beacon

- Target: 145,9 MHz
- Measurement: $\cong$ 145,9 MHz

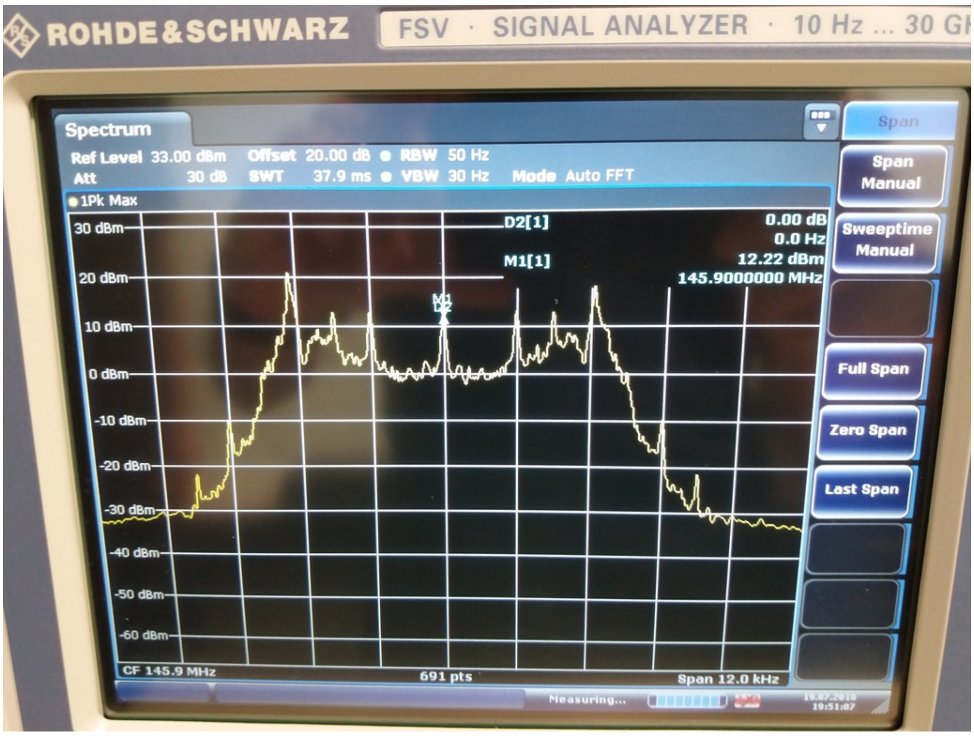


Figure 4.1: Beacon output frequency.

4.1.2 Downlink

- Target: 437,9 MHz

- Measurement: ≅ 437,9 MHz

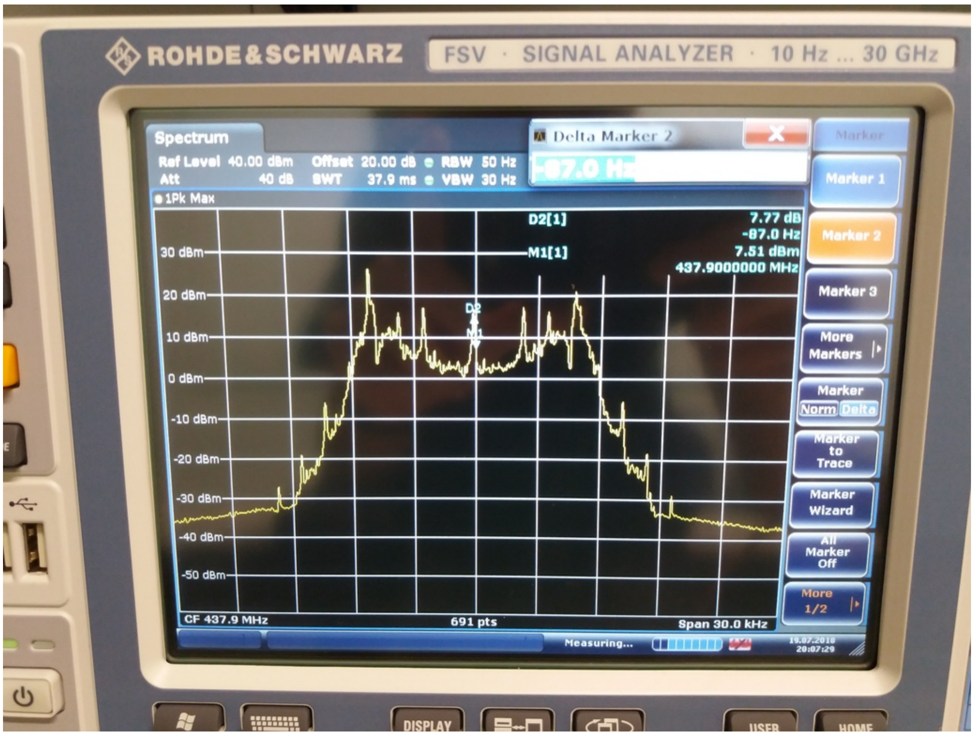


Figure 4.2: Downlink output frequency.

4.2 RF Signal Power

4.2.1 Beacon

- Target: 30 dBm

- Measurement: ≅ 22 dBm (+3 dBm including the cable and connectors loss)

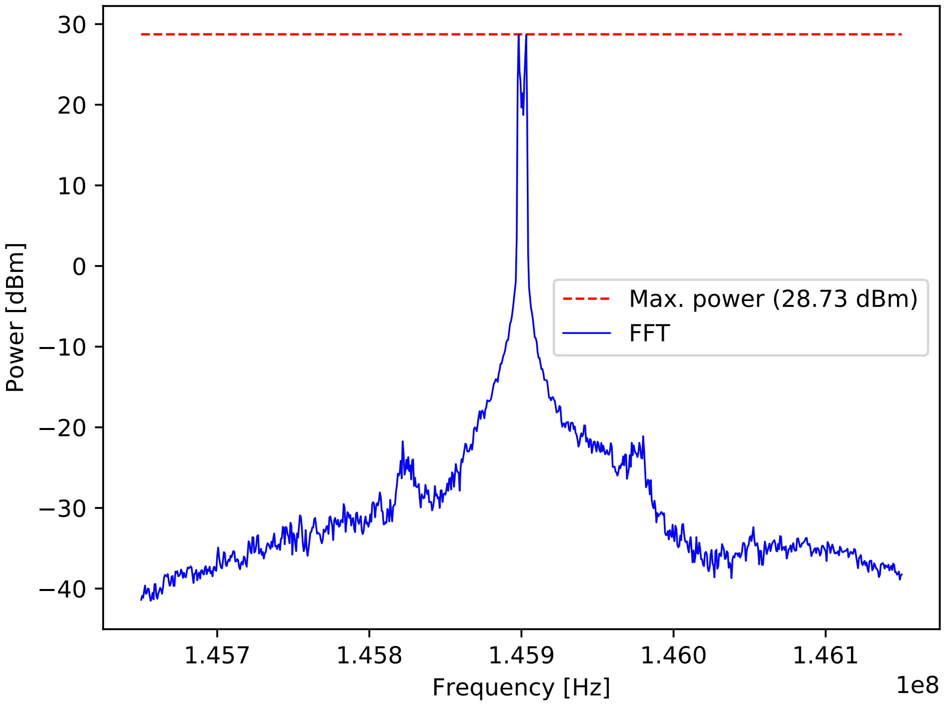


Figure 4.3: Beacon output power.

4.2.2 Downlink

- Target: 30 dBm
- Measurement: ≅ 27 dBm (+3 dBm including the cable and connectors loss)

The output power of the downlink radio can be seen in Figure 4.2.

4.3 Harmonics

4.3.1 Beacon

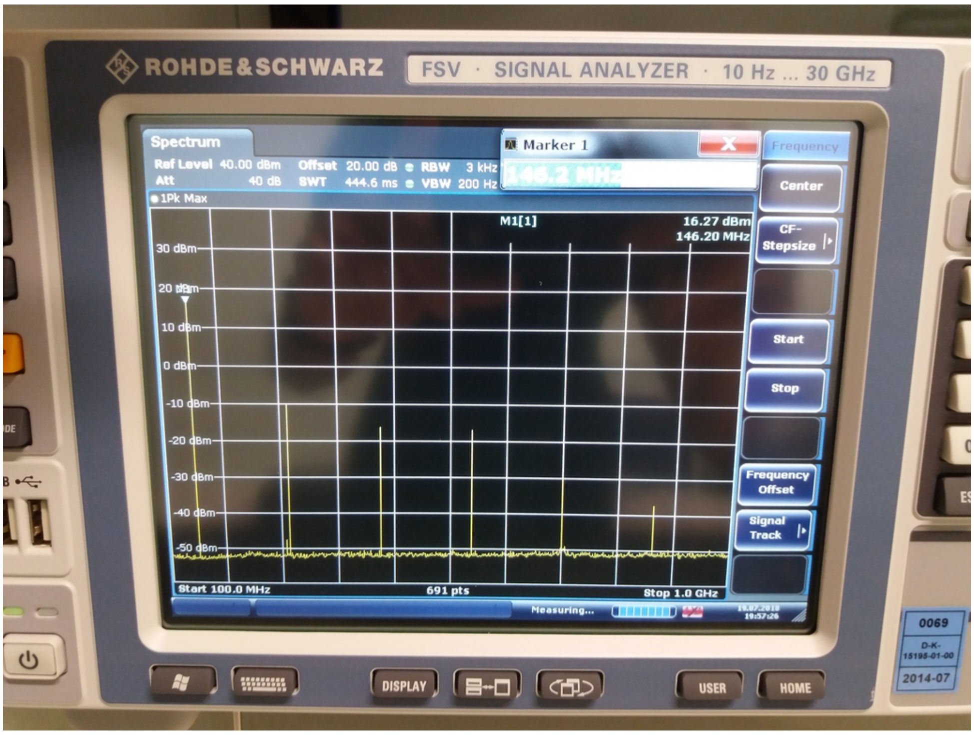


Figure 4.4: Beacon signal harmonics.

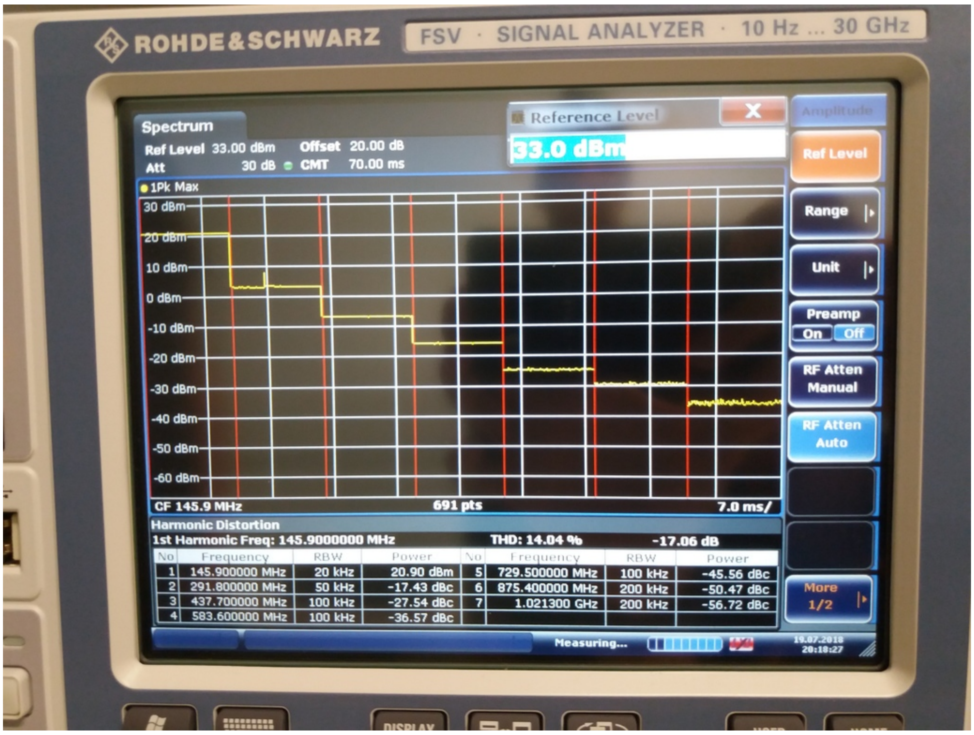


Figure 4.5: Analysis of the beacon signal harmonics.

4.3.2 Downlink

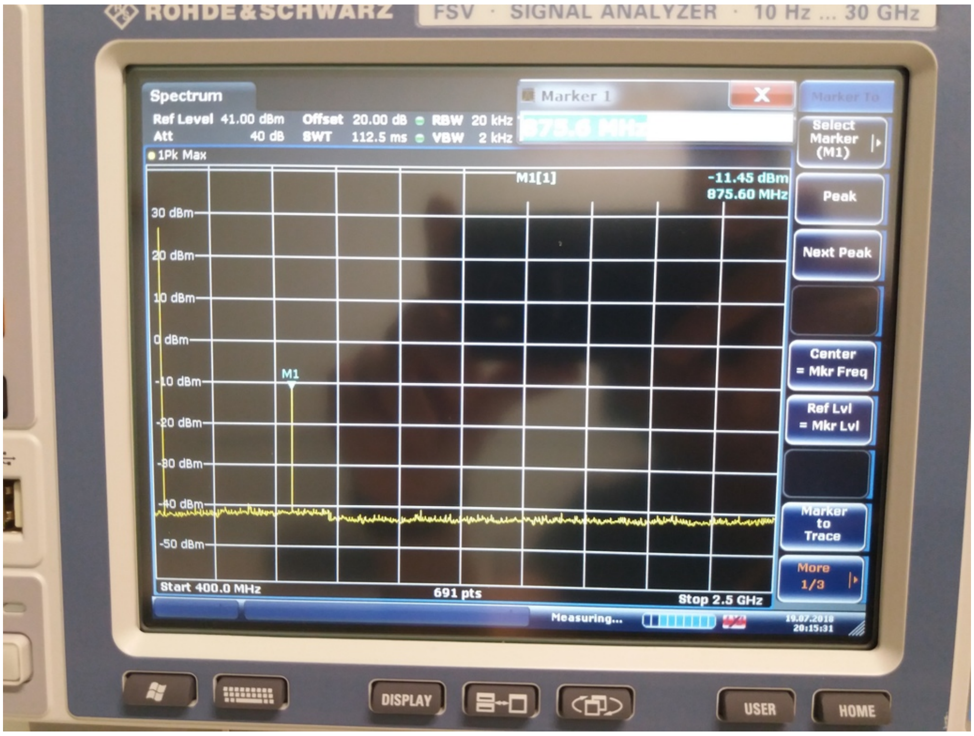


Figure 4.6: Downlink signal harmonics.

CHAPTER 5

Assembly Instructions

5.1 Radio Configuration

This appendix is a tutorial with the purpose of generate a source code file with basic configuration parameters of the radio module. For it, the WDS software from Silicon Labs will be used (version 3.2.11.0).

Unfortunately, the software in only available for Windows platforms.

5.1.1 Steps

After the installation of the software, the procedures to configure the beacon radio are described bellow (When a parameter to configure the telemetry link radio differs from the beacon radio, a note describes the difference).

Step 1

1. Open the WDS software.
2. The following box will appear in the center of the window.
3. Click in "Simulate radio" and go to the next step.

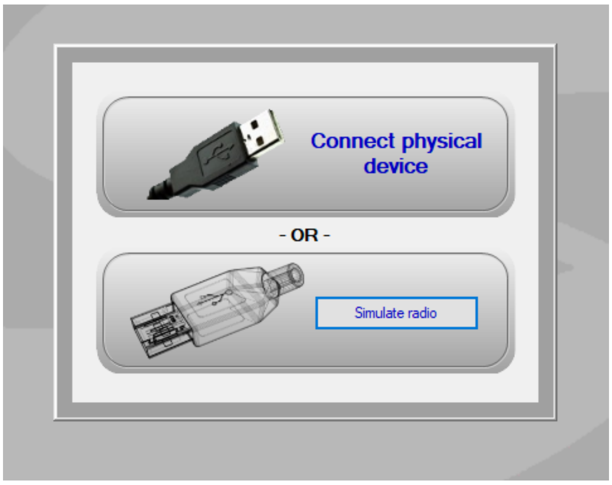


Figure 5.1: Step 1 of the radio configuration.

Step 2

1. In the list of radios that appeared on the new window, select the chip type "Si4463".
2. In the revision column, select "B1".
3. Click on "Select Radio" to go to the next step.

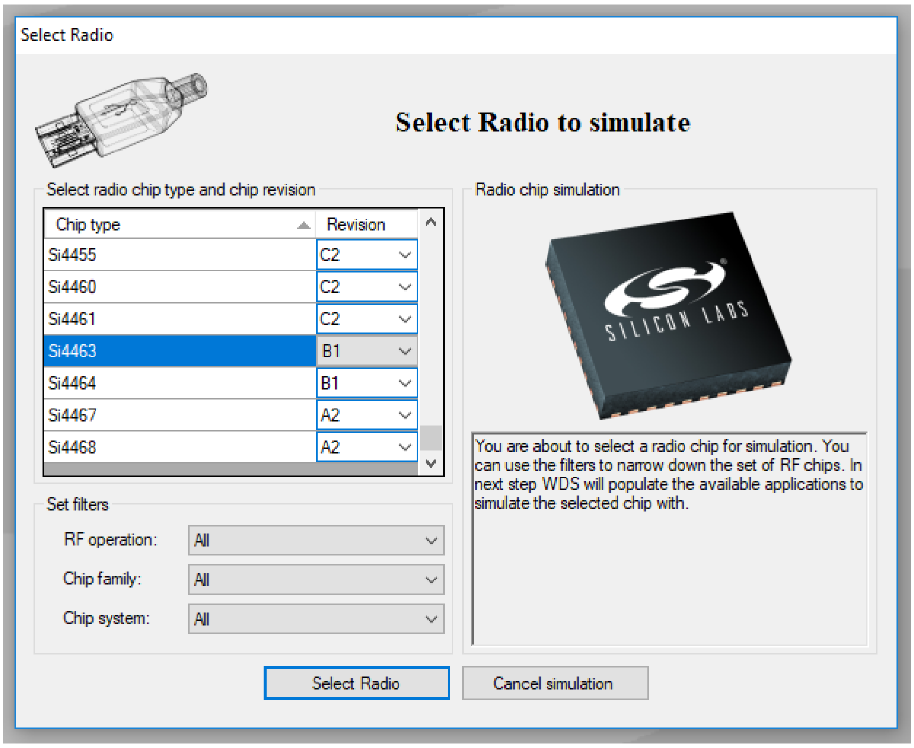


Figure 5.2: Step 2 of the radio configuration.

Step 3

1. Select "Radio Configuration Application".
2. Click on "Select Application" to go to the next step.

Step 4

1. In the "Frequency and power" tab, change the base frequency to 145,9 MHz.
2. Change the channel spacing to 0 kHz.
3. Change the crystal tolerance to 10,0 ppm (Both RX and TX).
4. Go to the next step.

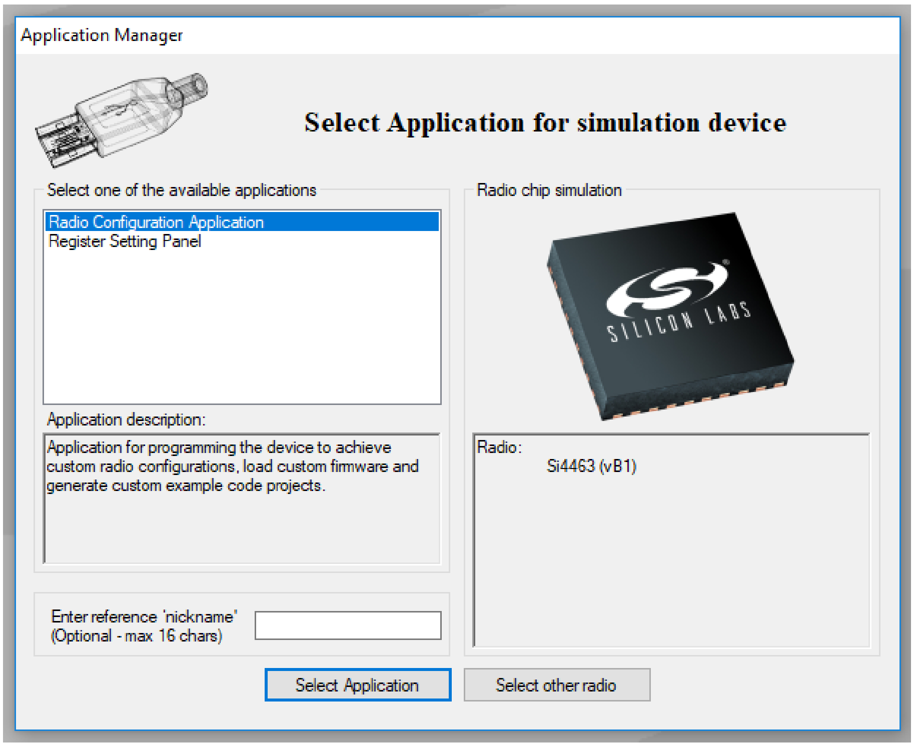


Figure 5.3: Step 3 of the radio configuration.

Step 5

1. In the “RF parameters" tab, change the modulation type to “2GFSK".
2. Change the the data rate to 1,2 kbps.
3. Change the deviation to $2,5\ kHz$.
4. Go to the next step.

Step 6

1. In the “Packet" tab, many subtabs will appear. In “Preamble" change the “Preamble TX length" to 4 bytes.
2. Again, in “Preamble", change the “Preamble pattern" to “Std. 1010 pattern (>= 32 and < 40 bits)".
3. Go to the next step.

Step 7

1. In the “Sync Word" tab, change the sync word length field to “4 bytes".
2. In the “Sync Word (on air int.)", enter the following sequence: 5D E6 2A 7E. This sequence is the sync word used by the NGHam protocol.
3. Go to the next step.

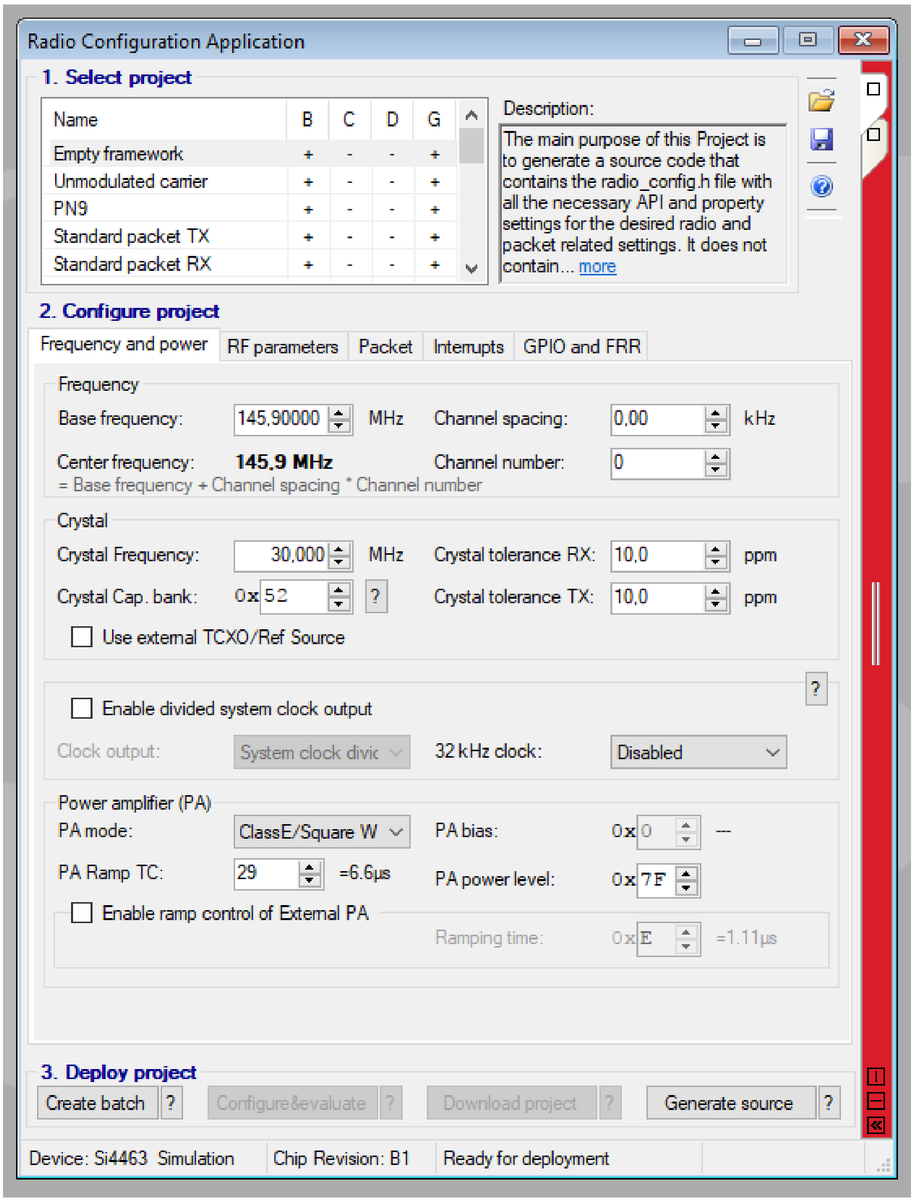


Figure 5.4: Step 4 of the radio configuration.

Step 8

1. In the “Field 1" tab, change the “Field length" to 50 bytes.
2. Go to the next step.

Step 9

1. In the “Variable length config" tab, there is no values to change.
2. Go to the next step.

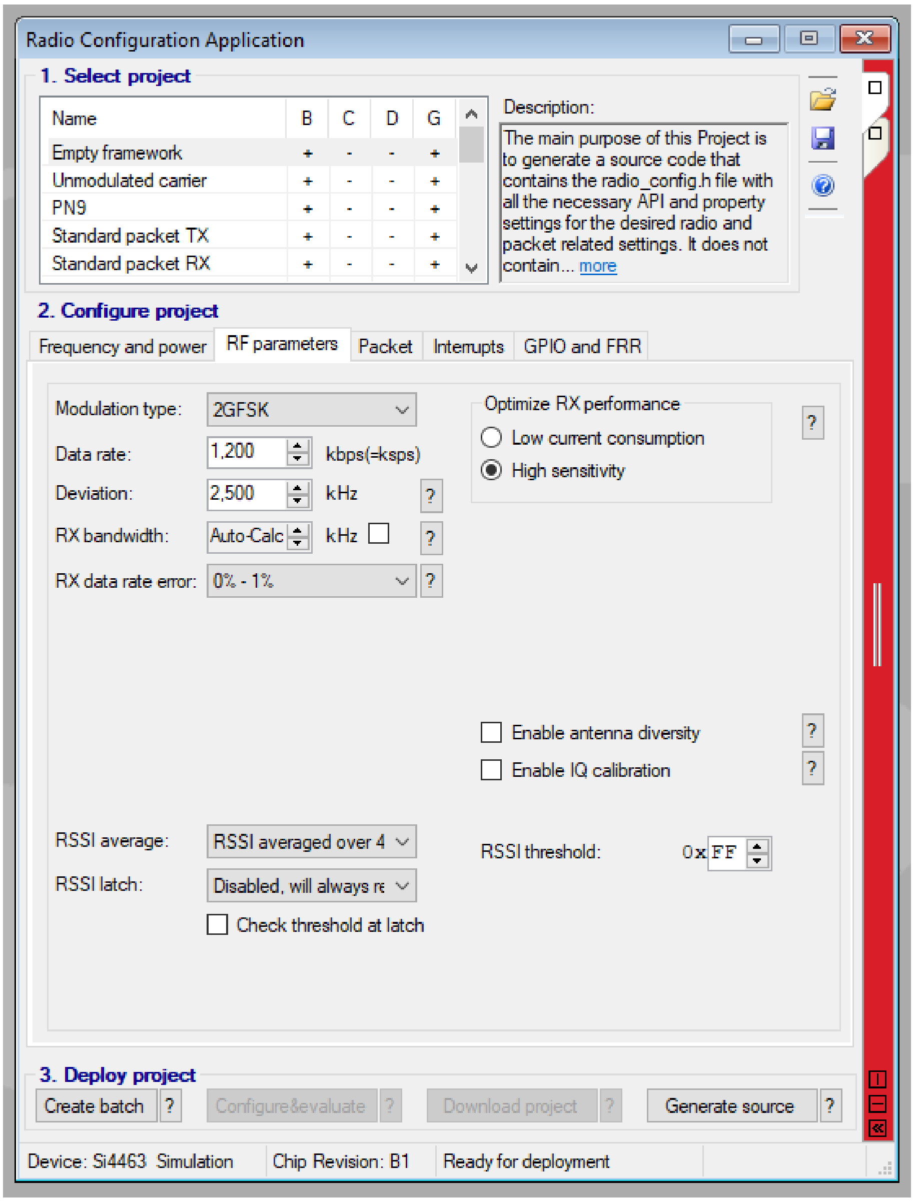


Figure 5.5: Step 5 of the radio configuration.

Step 10

1. In the "CRC config" tab, choose "No CRC." in "CRC polynomial".
2. Go to the next step.

Step 11

1. In the "Whitening config" tab, there is no values to change.
2. Go to the next step.

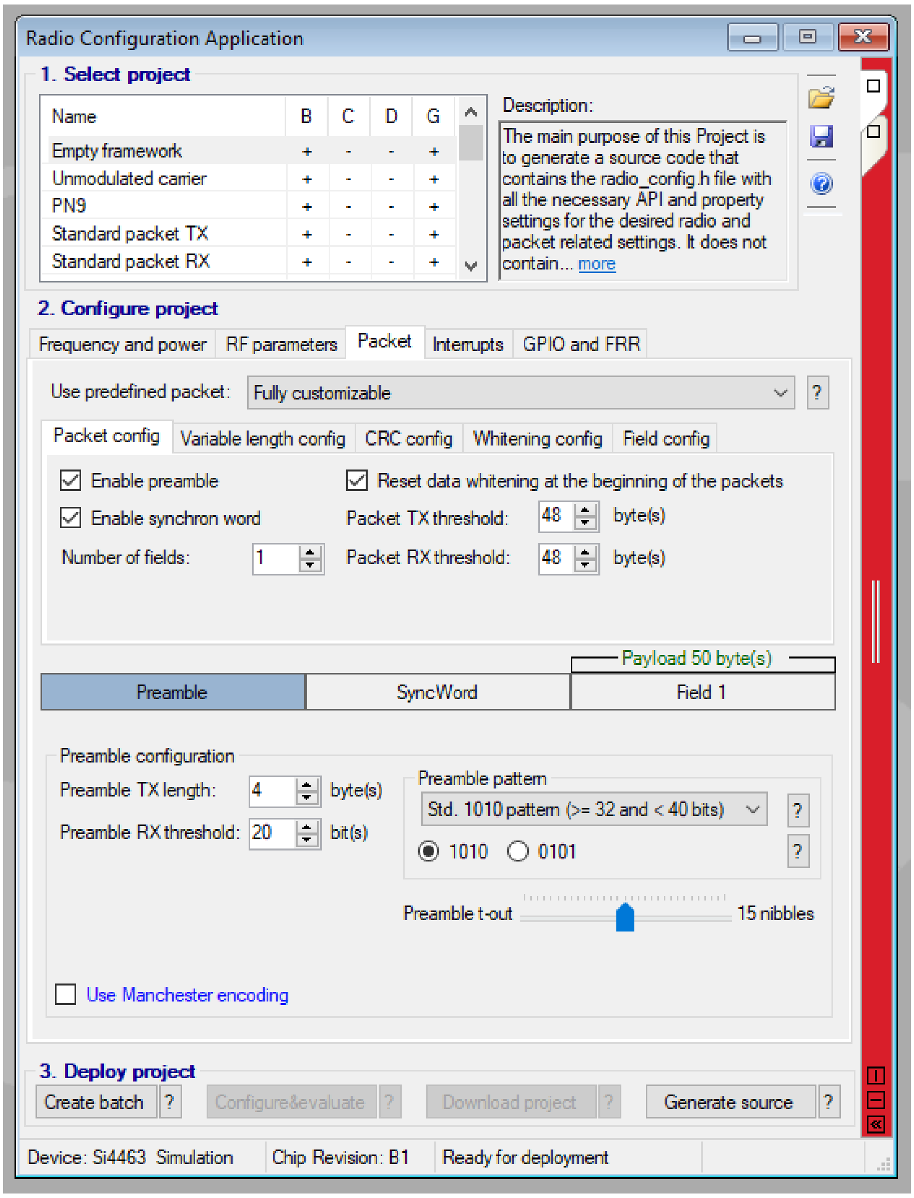


Figure 5.6: Step 6 of the radio configuration.

Step 12

1. In the "Field config" tab, there is no values to change.
2. Go to the next step.

Step 13

1. In the "Interrupts" tab, there is no values to change.
2. Go to the next step.

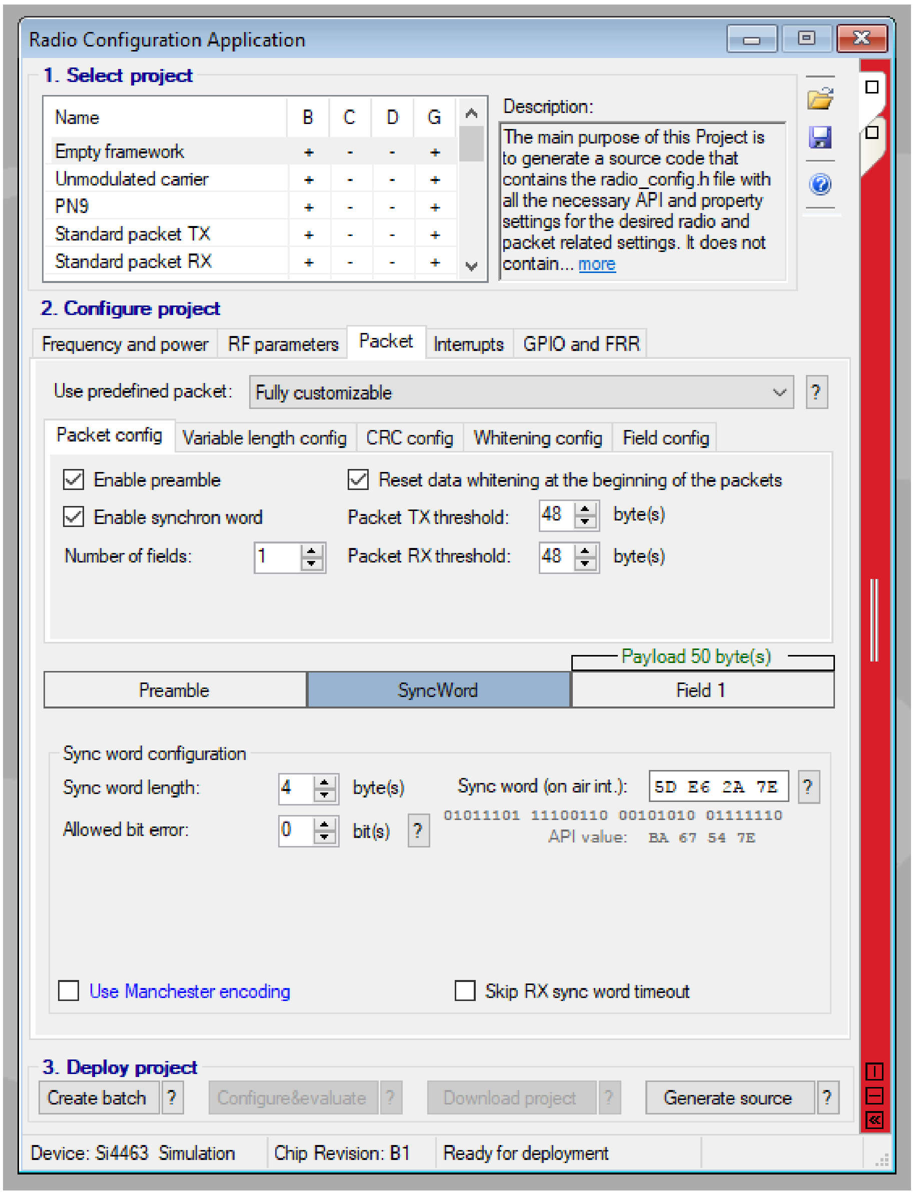


Figure 5.7: Step 7 of the radio configuration.

Step 14

1. In the “GPIO and FRR" tab, enable pullup in GPIO1 and choose “TX_FIFO_EMPTY – This output is..." as functionality.
2. Enable pullup in GPIO2 and choose “RX_STATE – This output is..." as functionality.
3. Enable pullup in GPIO3 and choose “TX_STATE – This output is..." as functionality.
4. Enable pullup in NIRQ and choose “Active low interrupt signal" as functionality.
5. Enable pullup in SDO and choose “SDO – Output SPI Serial data out." as functionality.
6. Select “Global status" for the “Fast Response Register A".

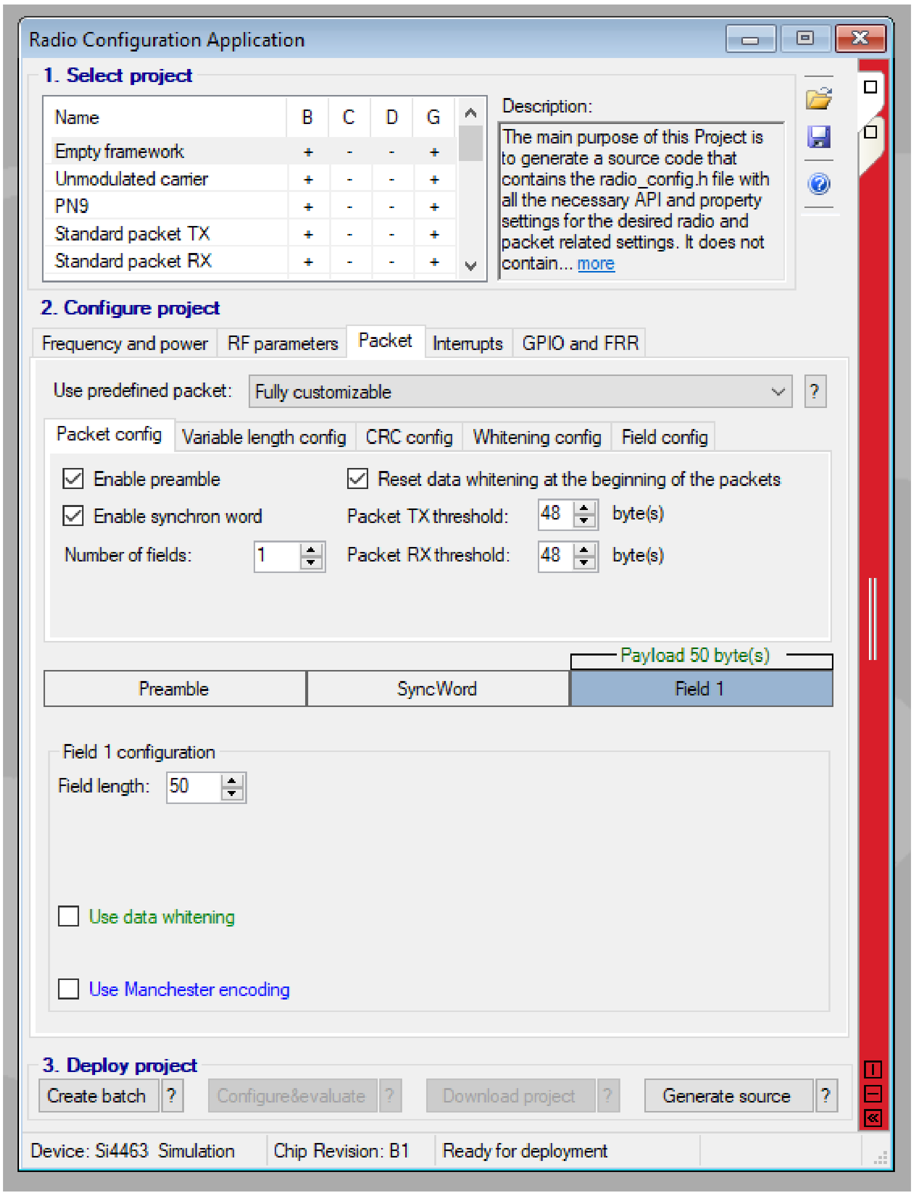


Figure 5.8: Step 8 of the radio configuration.

7. Select "Global interrupt status" for the "Fast Response Register B".
8. Select "Packet Handler status" for the "Fast Response Register C".
9. Select "Chip status status" for the "Fast Response Register D".
10. Go to the next step.

Step 15

1. Click in "Generate Source" and select the ".h" type of source.
2. The software will ask where to save the generated file.
3. The generated *.h file must be copied to the directory of the rf4463 driver.

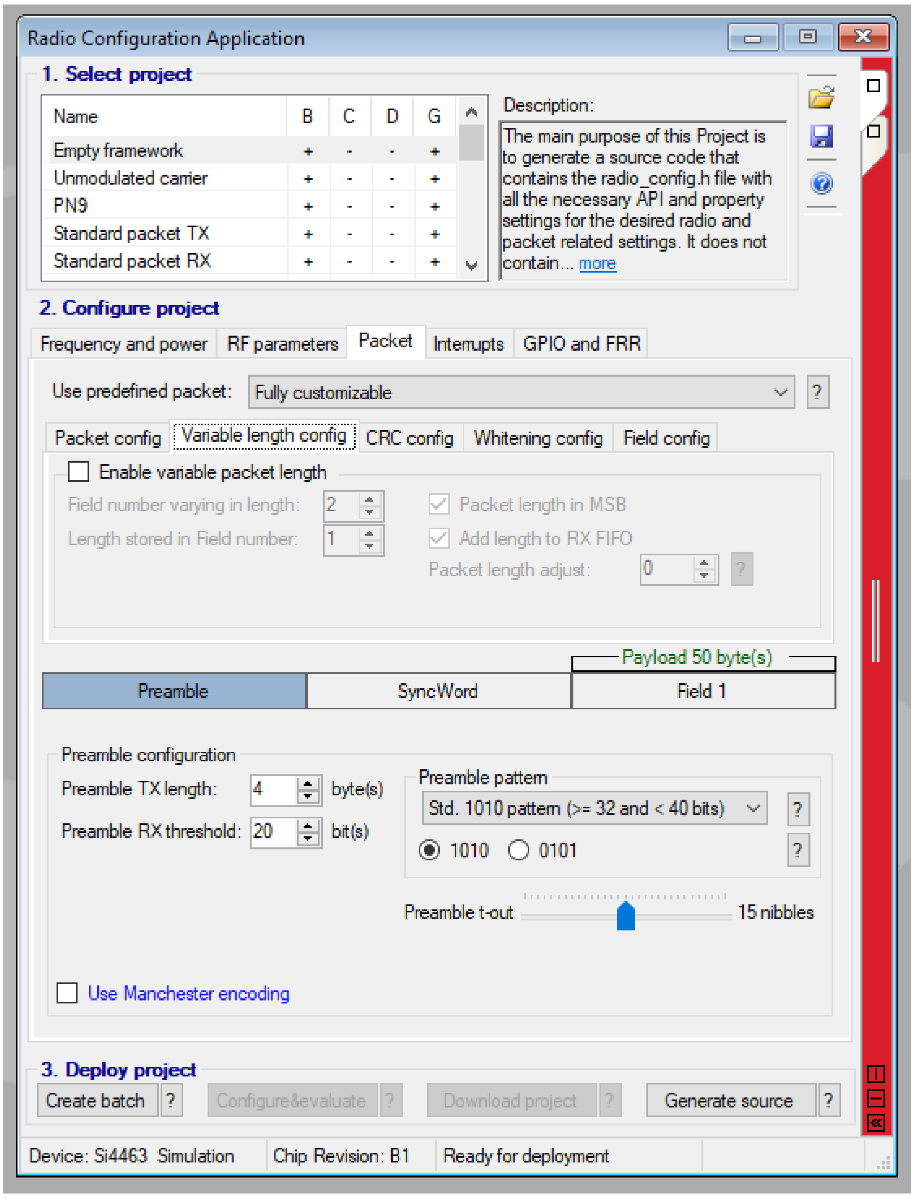


Figure 5.9: Step 9 of the radio configuration.

5.1.2 Final Remarks

This tutorial has the objective of generate a basic configuration parameters of the radio, some functionalities of the radio are not covered by the WDS software, and so, must be configured/controlled in the device driver.

5.2 Compiling and Building the Beacon Firmware

This tutorial is a reference to compile, build and flash the firmware of the beacon of the TTC module. All the software development was made using the Code Composer Studio (CCS) IDE, version 7.3.0. To load the code into the TTC MCU, the MSP-FET can be used.

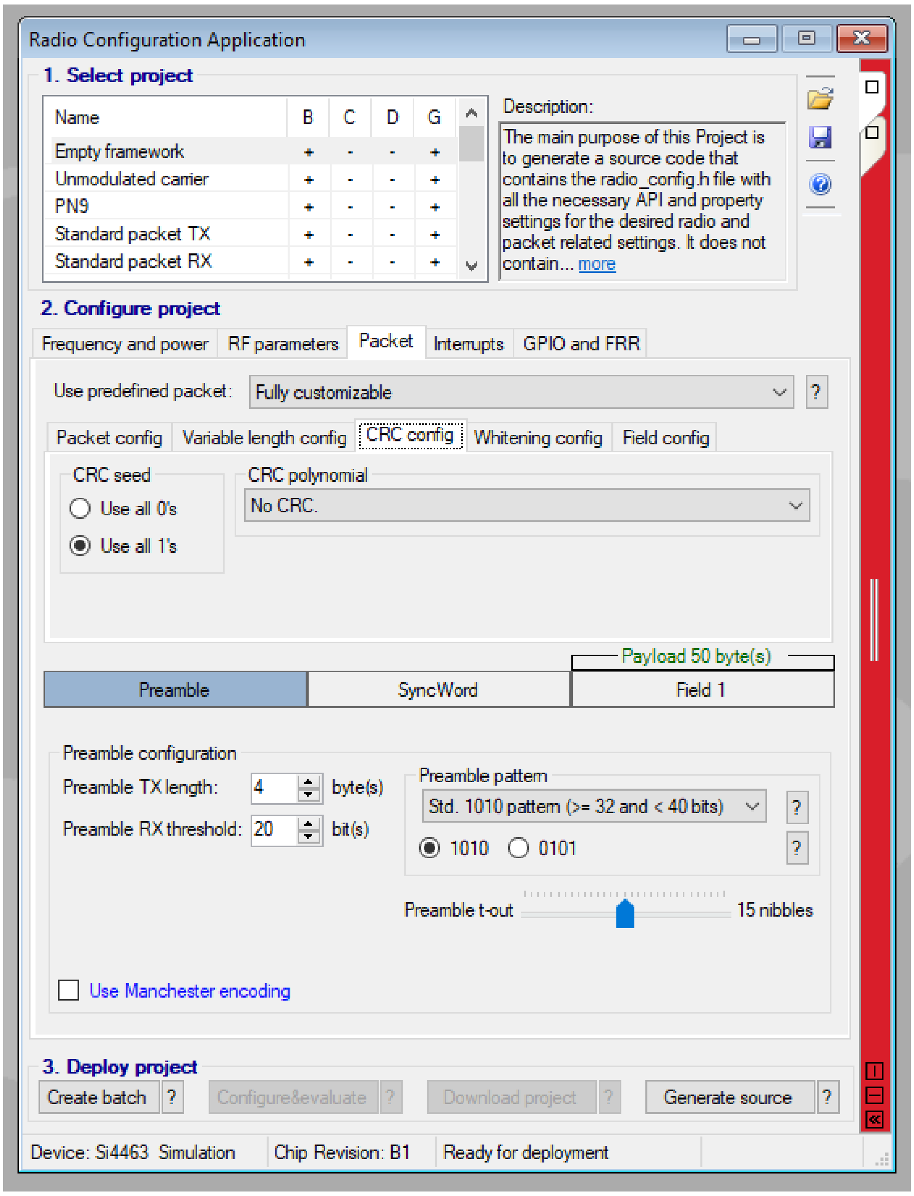


Figure 5.10: Step 10 of the radio configuration.

5.2.1 Creating the Project

After the download and installation of the CCS, open it and create a new project with following steps:

1. Go to “File” -> “New” -> “CCS Project”.
2. The window from the figure 5.15 will appear on the screen.
3. On “Target”, select MSP430F6659.
4. On “Project name”, enter the name of the project (It can be “beacon”, “beacon_firmware”, or whatever name you want).
5. On “Project templates and examples” box, select “Empty project”.

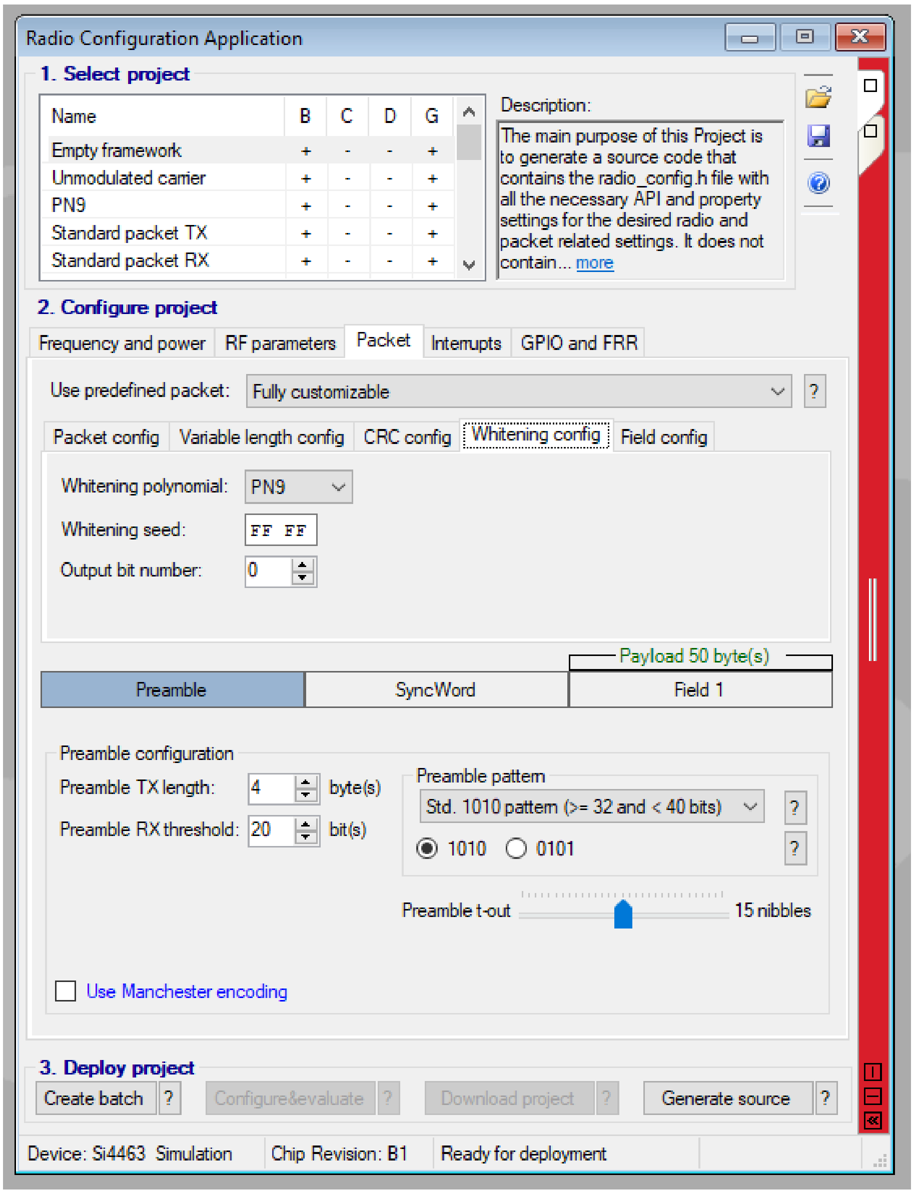


Figure 5.11: Step 11 of the radio configuration.

6. Click on “Finish”.

Now the beacon project with the correct parameters has been created, and the project source code files must be moved to the project directory. Copy the files inside the “beacon” folder from the TTC git repository to the project folder.

5.2.2 Compiling and Building

To compile and build the firmware, click on "Project" -> "Build Project". If the process finish with no errors, the code is ready to go to the MCU of the beacon.

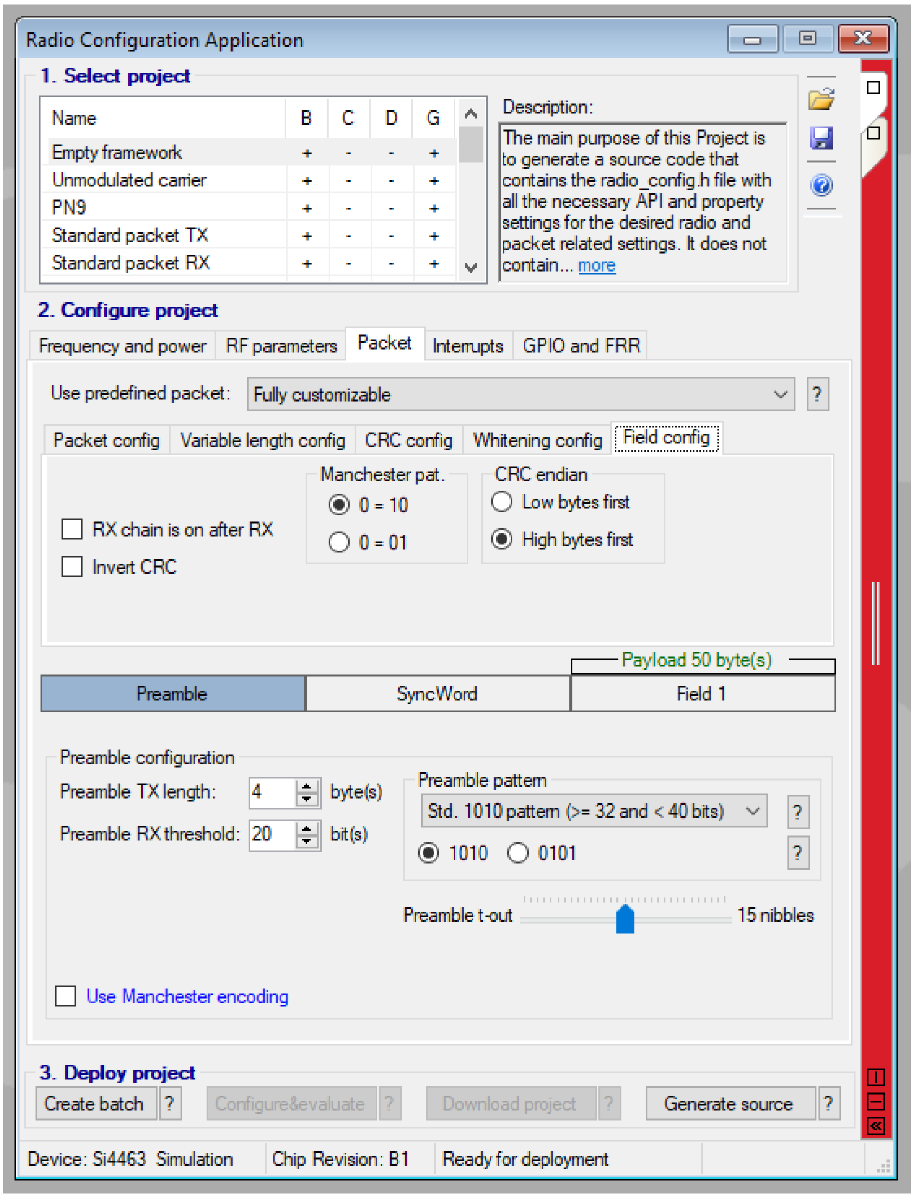


Figure 5.12: Step 12 of the radio configuration.

5.2.3 Flashing

To load the code into the board, connect the MSP-FET in the computer and follow the power-on tutorial.

With the board turned on and the MSP-FET connected, click on “Run" -> “Debug" or just press F11. If no errors occur, the firmware was loaded succesfully to the board.

5.3 Power-On the TTC Module

To power-on the TTC module, there are three different possibilities:

1. Using three different power sources: one for the MCU, one for the beacon radio module and one for the telemetry radio module.

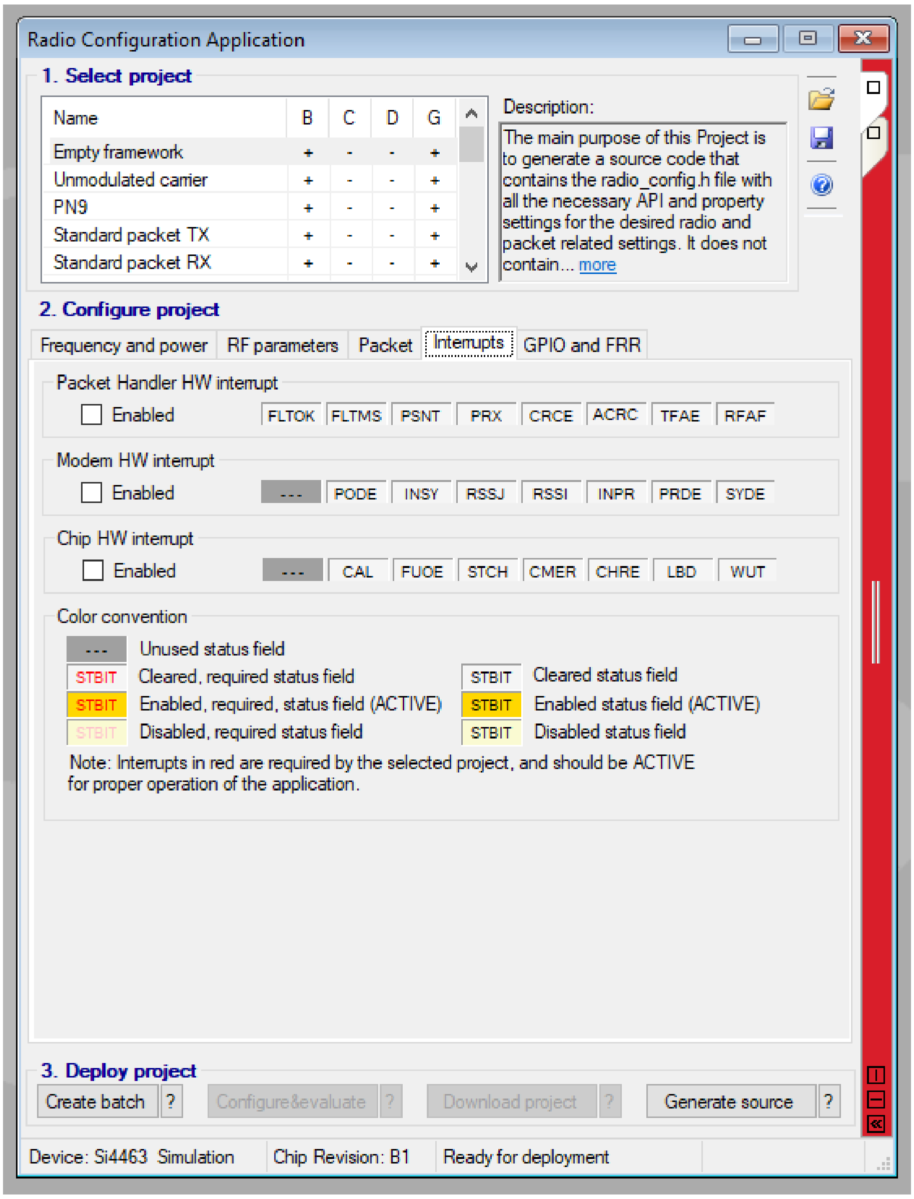


Figure 5.13: Step 13 of the radio configuration.

2. Using two different power sources: one for each radio module (in this case, the beacon MCU is powered using the JTAG bus).

3. Using just one power source to all components (THIS METHOD IS NOT SAFE: there is no control over the power consumption of each component).

To test just the beacon, there is no need to power-on the telemetry radio module. Turning on the telemetry radio only makes sense if an external module is connected and controlling it (Like an OBC or an OBDH).

The connections reference of the TTC board can be found in External Connections.

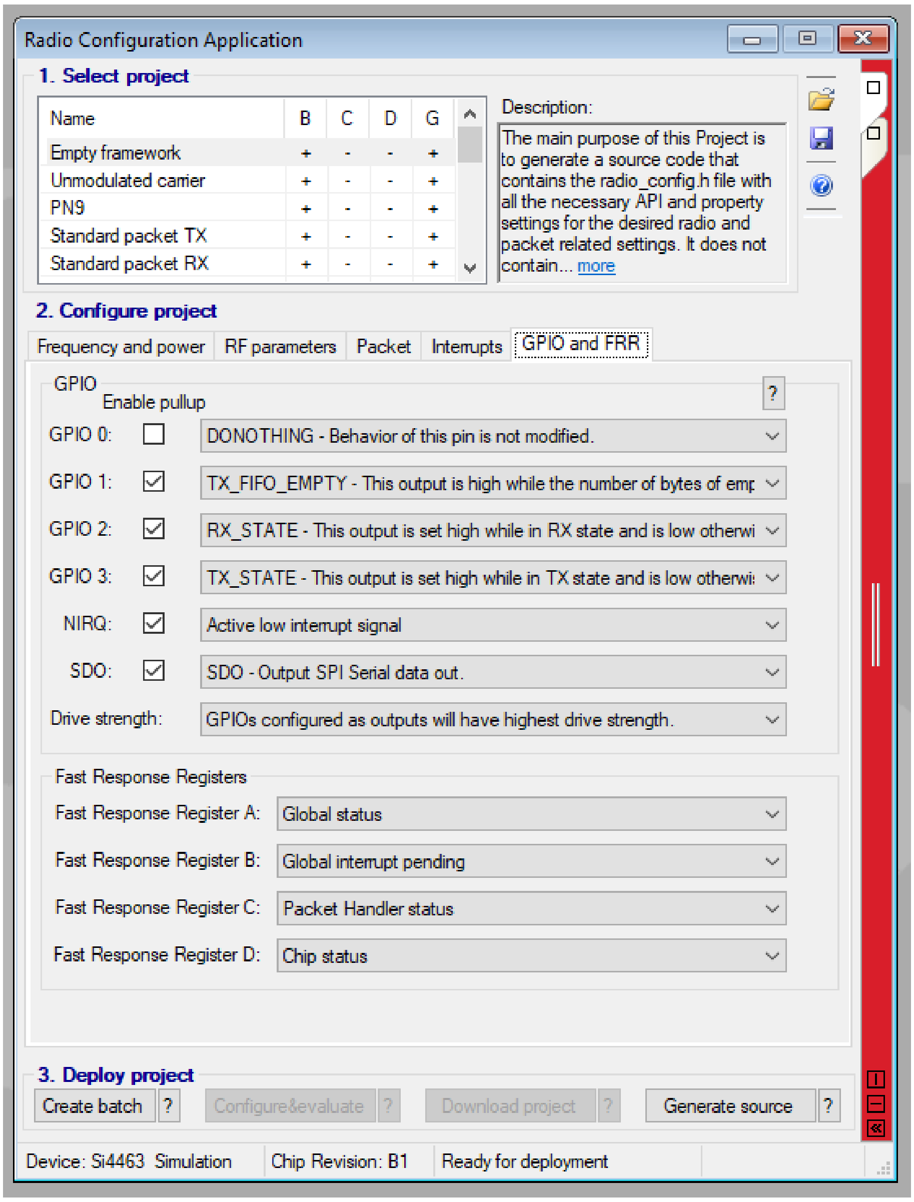


Figure 5.14: Step 14 of the radio configuration.

5.3.1 Using an External Power Source for the Beacon MCU

Power-On the Beacon MCU

1. Set an output of a channel of the power source to $3,3\ V$ and $30\ mA$ (Cut-off current).
2. Connect the positive cable to the H2A-14 or H2B-14 pin of the PCI-104 connector.
3. Connect the ground cable to any GND pin of the PCI-104 connector.
4. Turn on this channel of the power source.

Power-On the Beacon Radio Module

1. Set another output of a channel of the power source to $5,0\ V$ and $500\ mA$ (Cut-off current).

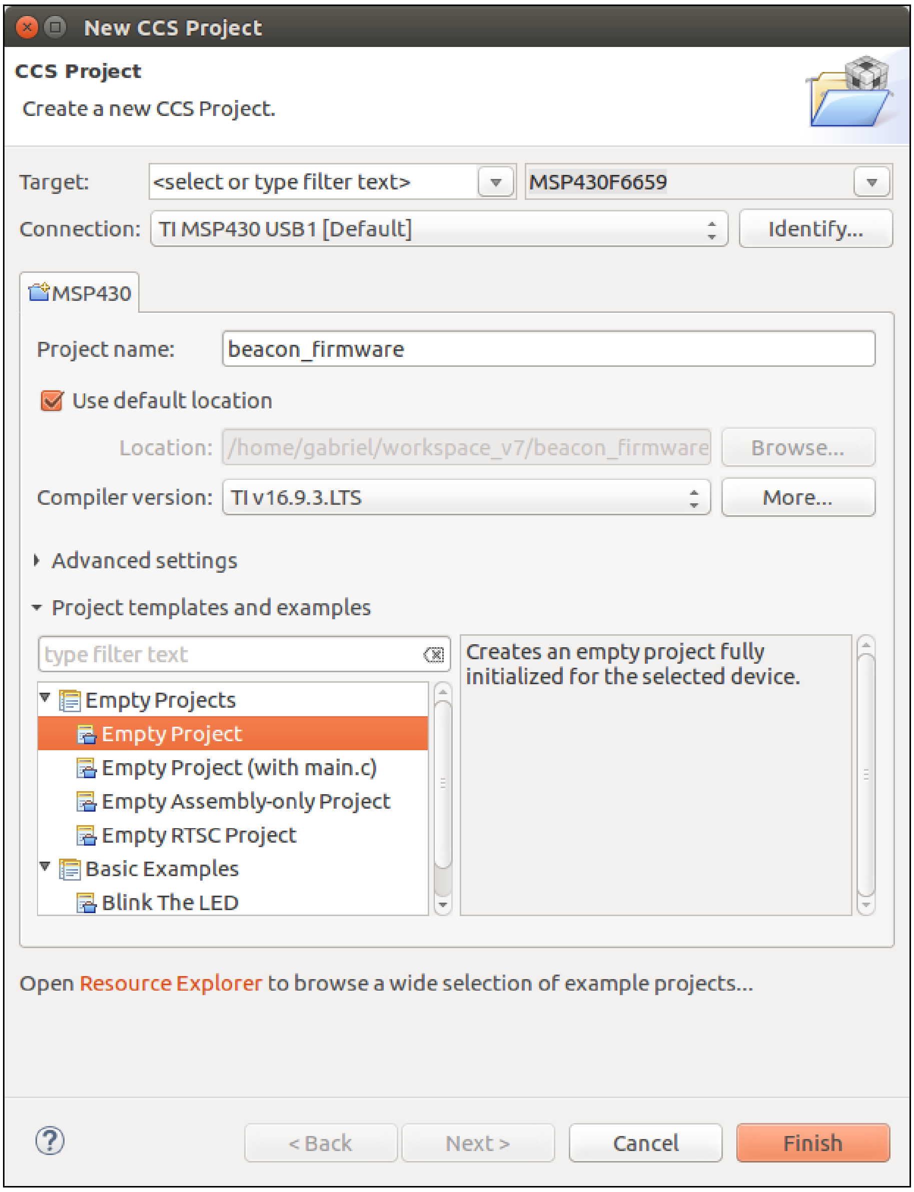


Figure 5.15: New CCS project window.

2. Connect the positive cable to the H1A-26 pin of the PCI-104 connector.
3. Connect the ground cable to any GND pin of the PCI-104 connector.
4. Turn on this channel of the power source.

Power-On the Telemetry Radio Module

1. Set another output of a channel of the power source to 5,0 V and 500 mA (Cut-off current).
2. Connect the positive cable to the H1A-25 pin of the PCI-104 connector.
3. Connect the ground cable to any GND pin of the PCI-104 connector.
4. Turn on this channel of the power source.

5.3.2 Using the JTAG as Power Source for the Beacon MCU

In this case, to power on the beacon MCU, just connect a jumper in the P4 connector of the board and after, connect a MSP-FET debbuger to the JTAG connector.

The procedure to power-on the radio modules is the same as above.

5.4 Receiving the Beacon Data

5.4.1 Required Softwares

- GQRX (For just receive the signal).
- RTL-SDR drivers (The GRQX will install all the required drivers).
- FloripaSat GRS (For receive the signal and data).

Supported SDRs

- RTL-SDR
- FunCube Dongle Pro+

5.4.2 Receiving the Beacon Signal

In this method, you will not be able to see the beacon data in real time. This only shows the presence of the beacon signals in the air.

The instructions to receive the beacon signal in the GQRX are available bellow:

1. Open the GQRX software.
2. Configure it for your SDR model.
3. Press the power button on the upper left corner.
4. Select the frequency to 145,9 MHz.
5. Wait the spectrum of the beacon signal appear.

The figure 5.16 illustrates the reception of the beacon signal in the GQRX software.

5.4.3 Receiving the Beacon Data

Using this method, you will be able to see the beacon (and the telemetry) data in real time on the computer screen. For that, the FloripaSat GRS sofware is required.

1. Open the FloripaSat GRS software.
2. Connect a the SDR device.
3. Select the source of the signal.
4. Press the play button in the beacon frame.

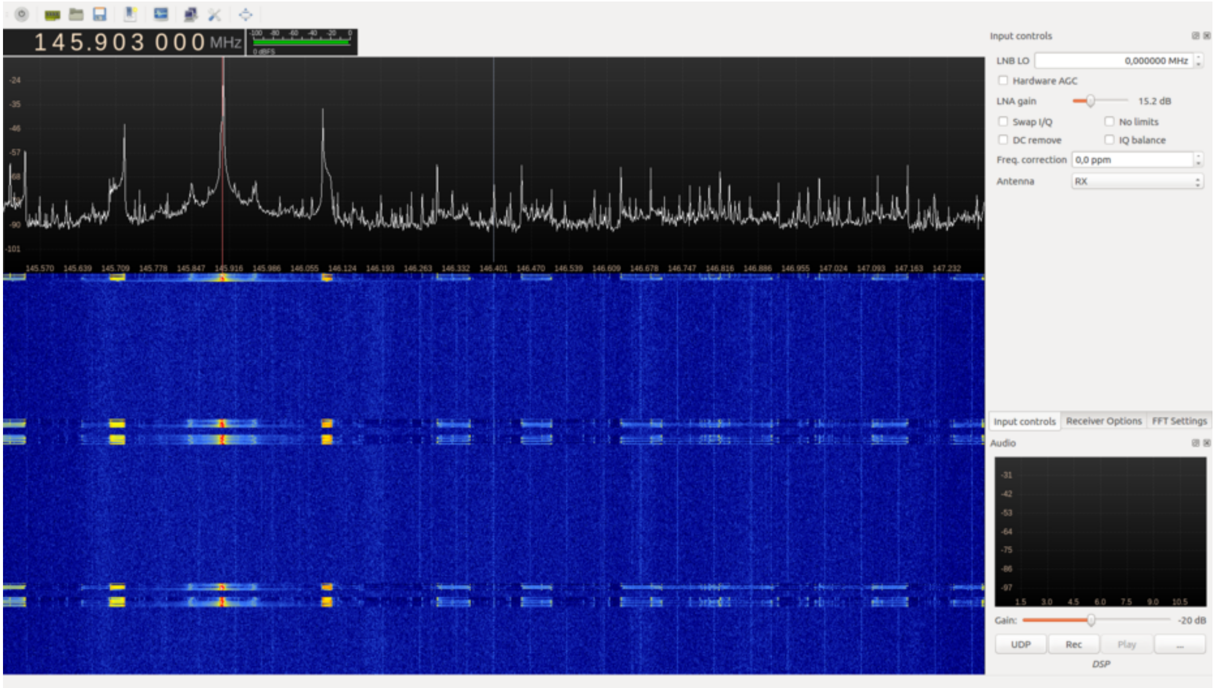


Figure 5.16: Beacon signal in GQRX.

5. A new window will appear in the screen, with the FFT and Waterfall plot of the received signal (in real time) of the connected SDR.

6. Power on the beacon module of the TTC board.

7. Adjust the frequency of the receive to the center frequency of the beacon signal (Use the FFT and the Waterfall plot for that).

8. With the correct frequency sinthonization, the beacon data should appear in the main window of the FloripaSat GRS softwate.

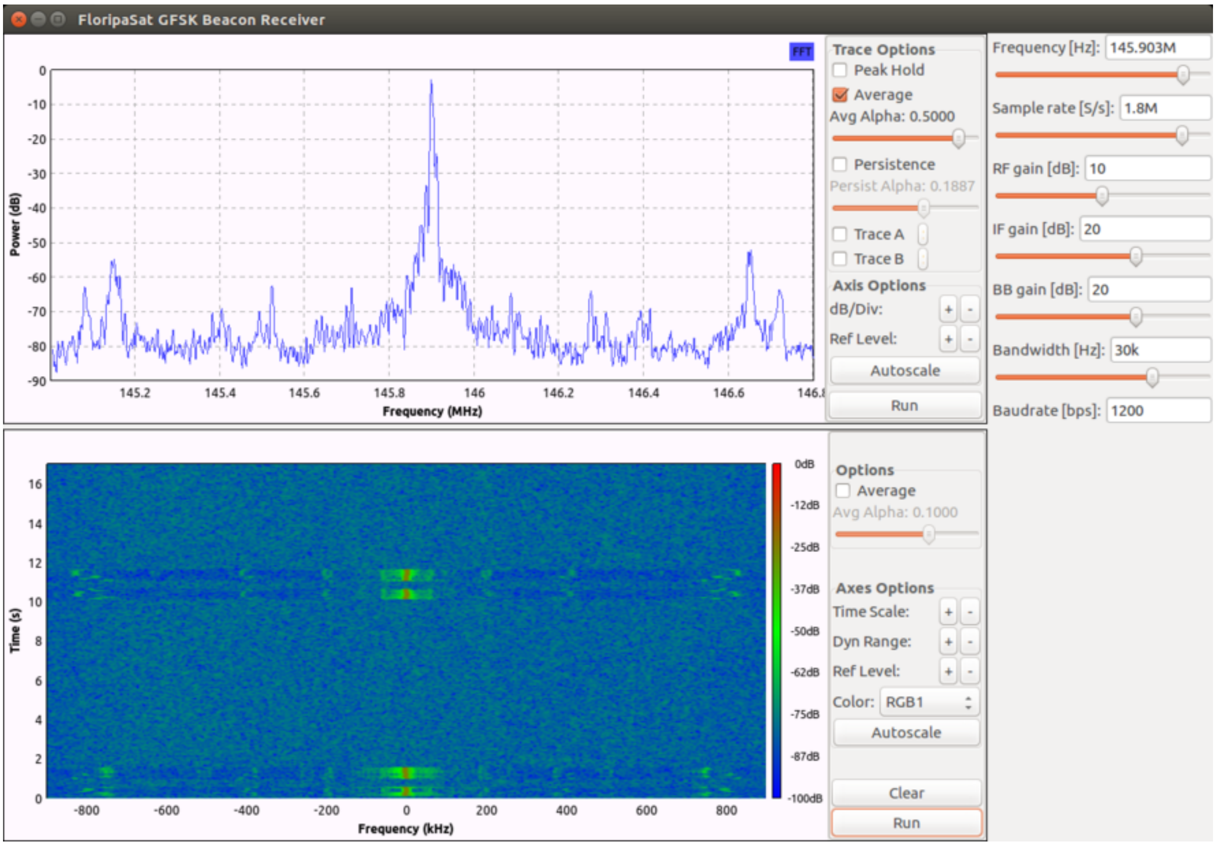


Figure 5.17: Beacon signal in the GNURadio receiver from the FloripaSat GRS.

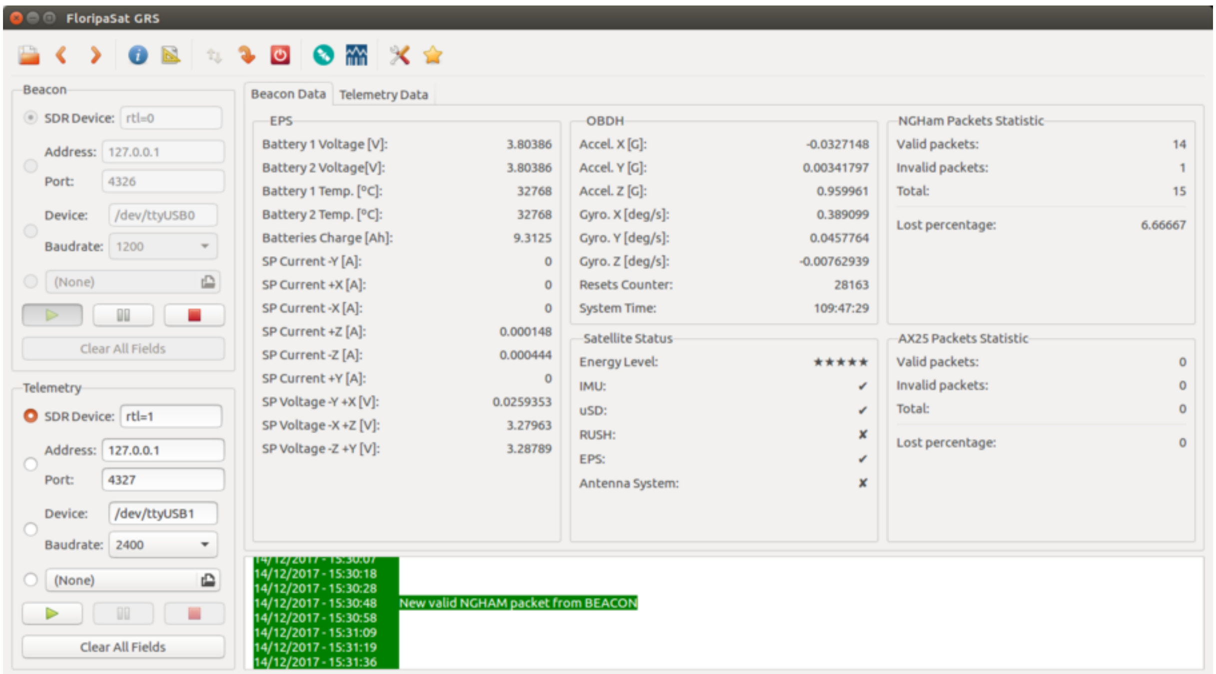


Figure 5.18: Beacon data in the FloripaSat GRS.

Bibliography

[1] Rafael P. Alevato. <*http://floripasat.space/*>. FloripaSat Project, December 2017.

[2] SpaceLab. <*https://github.com/floripasat/ttc*>, December 2017.

[3] Texas Instruments Inc. *MSP430x5xx and MSP430x6xx Family User's Guide*, October 2016.

[4] NiceRF Wireless Technology Co., Ltd. *RF4463F30 1W High Sensitivity Wireless Transceiver Module Product Specification*, rev 2.1 edition, June 2016.

[5] Silicon Laboratories Inc. *Si4464/63/61/60 High-Performance, Low-Current Transceiver*, rev 1.2 edition, 2016.