SLB-OBDH2-DOC-v0.9

OBDH 2.0 Documentation

OBDH 2.0 Documentation
SpaceLab, Universidade Federal de Santa Catarina, Florianópolis - Brazil

November 2021

OBDH 2.0 Documentation
November, 2021

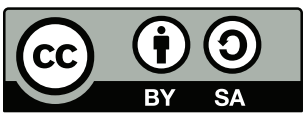
Project Chief:
Eduardo Augusto Bezerra

Authors:
Gabriel Mariano Marcelino
André Martins Pio de Mattos
Yan Castro Azeredo

Contributing Authors:

Revision Control:

Version	Author(s)	Changes	Date
0.1	Gabriel M. Marcelino	Document creation	10/2019
0.5	Gabriel M. Marcelino	First stable hardware	08/2020
0.6	A. M. P. Mattos, G. M. Marcelino, Y. A. Azeredo	First hardware integration	04/2021
0.7	A. M. P. Mattos, G. M. Marcelino, Y. A. Azeredo	General firmware and hardware improvements	06/2021
0.8	Gabriel M. Marcelino	Adding the FRAM memory	10/2021
0.9	Gabriel M. Marcelino	TBC	TBC



List of Figures

List of Tables

Nomenclature

EDC	*Environmental Data Collector.*
PCB	*Printed Circuit Board.*
PCB	*Printed Circuit Board.*
ADC	*Analog-to-Digital Converter.*
FRAM	*Ferroelectric Random-Access Memory.*
GPIO	*General Purpose Input/Output.*
HAL	*Hardware Abstraction Layer.*
I2C	*Inter-Integrated Circuit.*
IC	*Integrated Circuit.*
IDE	*Integrated Development Environment.*
JTAG	*Joint Test Action Group.*
LED	*Light-Emitting Diode.*
OBC	*On-Board Computer.*
OBDH	*On-Board Data Handling.*
RTOS	*Real Time Operating System.*
SPI	*Serial Peripheral Interface.*
UART	*Universal Asynchronous Receiver/Transmitter.*
USCI	*Universal Serial Communication Interfaces.*

Contents

CHAPTER 1

Introduction

The OBDH 2.0 is an On-Board Computer (OBC) module designed for nanosatellites. It is one of the service modules developed for the FloripaSat-2 CubeSat Mission [1]. The module is responsible for synchronizing actions and the data flow between other modules (i.e., power module, communication module, payloads) and the Earth segment. It packs the generated data into data frames and transmit back to Earth through a communication module, or stores it on a non-volatile memory for later retrieval. Commands sent from Earth segment to the CubeSat are received by radio transceivers located in the communication module and redirected to the OBDH 2.0, which takes the appropriate action or forward the commands to the target module.

The module is a direct upgrade from the OBDH of FloripaSat-1 [2], which grants a flight heritage rating. The improvements focus on providing a cleaner and more generic implementation in comparison with the previous version, more reliability in software and hardware implementations, and adaptations for the new mission requirements. All the project, source and documentation files are available freely on a GitHub repository [3] under the GPLv3 license.

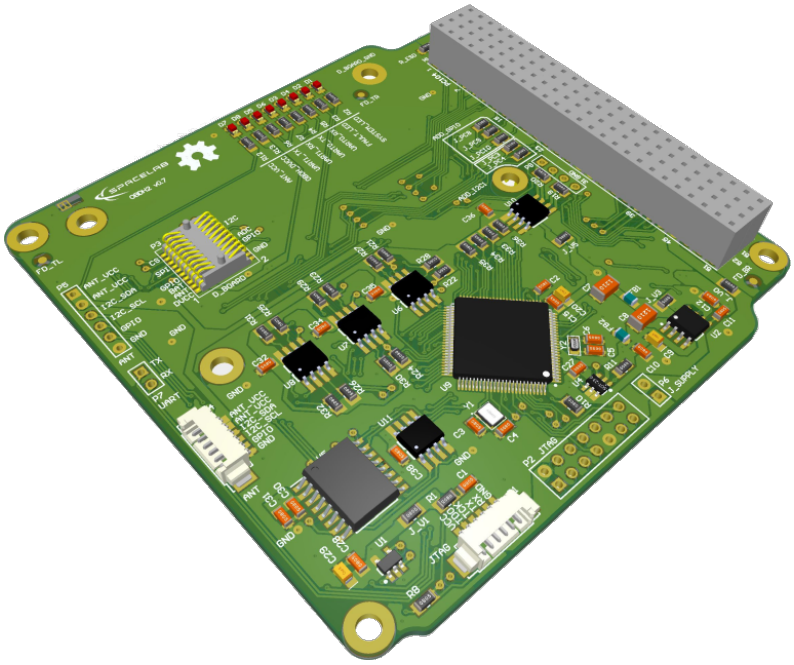


Figure 1.1: 3D view of the OBDH 2.0 PCB.

CHAPTER 2

System Overview

The board has a MSP430 low-power microcontroller that runs the firmware application and several other peripherals for an extended operation and physical interfaces (i.e., non-volatile memory, watchdog timer, service modules and payloads interfaces, daughterboard interface, current monitor). The microcontroller manages the others sub-modules within the board using serial communication buses, synchronizes actions, handles communication with the ground segment, and manages the data flow. The programming language used is C and the firmware was developed using the Code Composer Studio IDE (a.k.a. CCS) for compiling, programming and testing. The module has many tasks, such as interfacing peripherals and other MCUs, over distinct protocols and time requirements. Then, in order to improve predictability, a Real Time Operating System (RTOS) is used to ensure that the deadlines are observed, even under a fault situation in a routine. The RTOS chosen is the FreeRTOS (v10.0.0), since it is designed for embedded systems applications and it was already validated in space applications. The firmware architecture follows an abstraction layer scheme to facilitate higher level implementations and allow more portability across different hardware platforms.

2.1 Product tree

The product tree of the OBDH 2.0 module is available in Figure 2.1.

2.2 Block Diagram

The Figure 2.2 presents a simplified view of the module subsystems and interfaces. The microcontroller has a programming JTAG and 6 communication buses, divided in 3 different protocols (I2C, SPI and UART), that are shared between all the peripherals and external interfaces. Besides this channels, there are GPIO connections for various functions, from control ports to status pins. There is a non-volatile memory device to store the satellite data frames and critical status indicators. There are buffers and transceivers intended to allow secure and proper communication with external modules. A watchdog timer with voltage monitor and a current sensor are attached to the system for improving the overall reliability and generating essential housekeeping data. There is a generic daughterboard interface for extending the module capabilities with an auxiliary application board. Also, directly connected to the microcontroller, there is a UART debug interface. More details and descriptions about these components and interfaces are provided in the chapter 3.

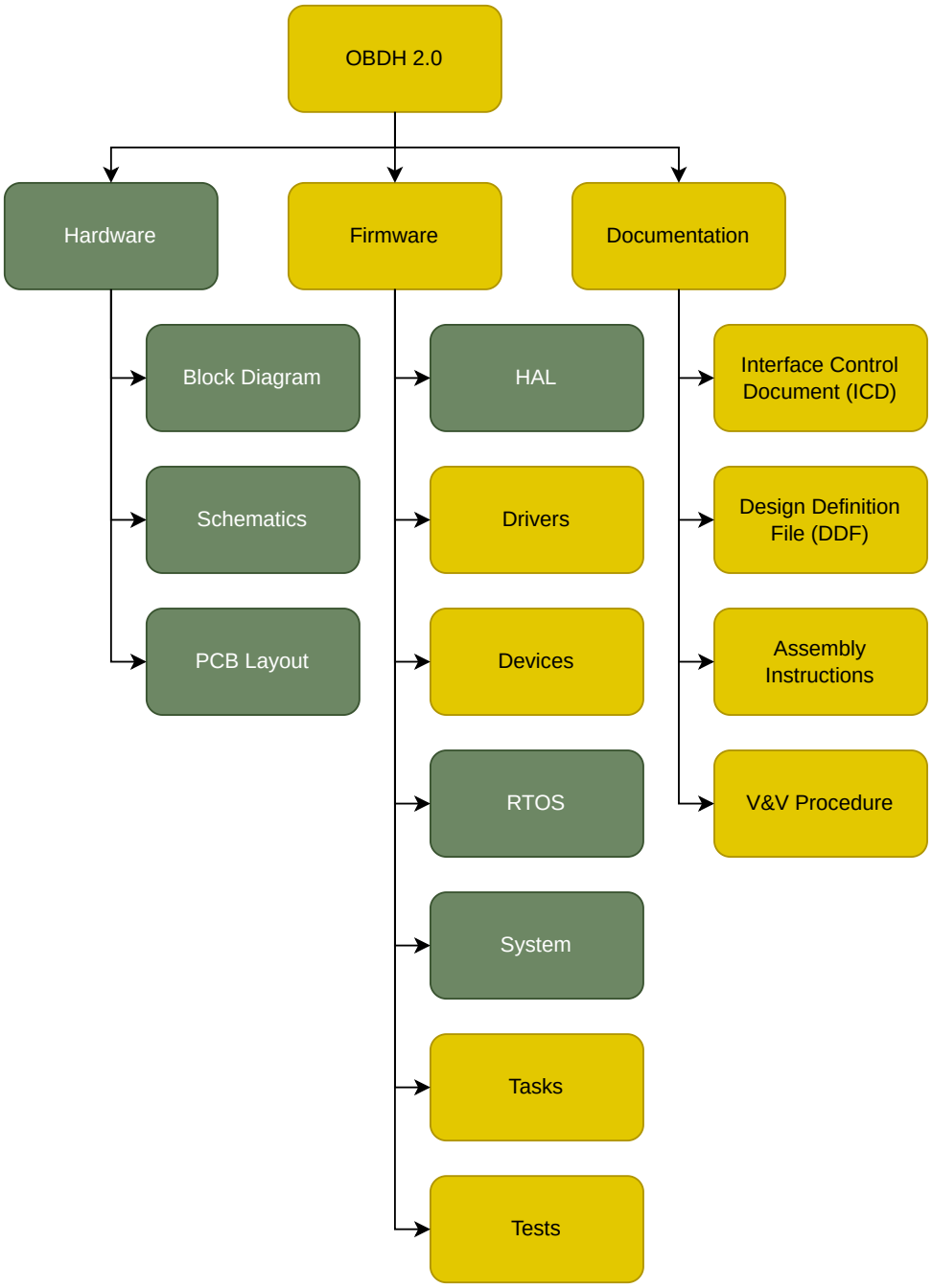


Figure 2.1: Product tree of the OBDH 2.0 module.

2.3 System Layers

As herein mentioned, the system is divided in various abstraction layers to favor high level firmware implementations. The Figure 2.3 shows this scheme, which is composed of third-part drivers at the lowest layer above the hardware, the operating system as the base building block of the module, the devices handling implementation, and the application tasks in the highest layer. More details are provided in the chapter 4.

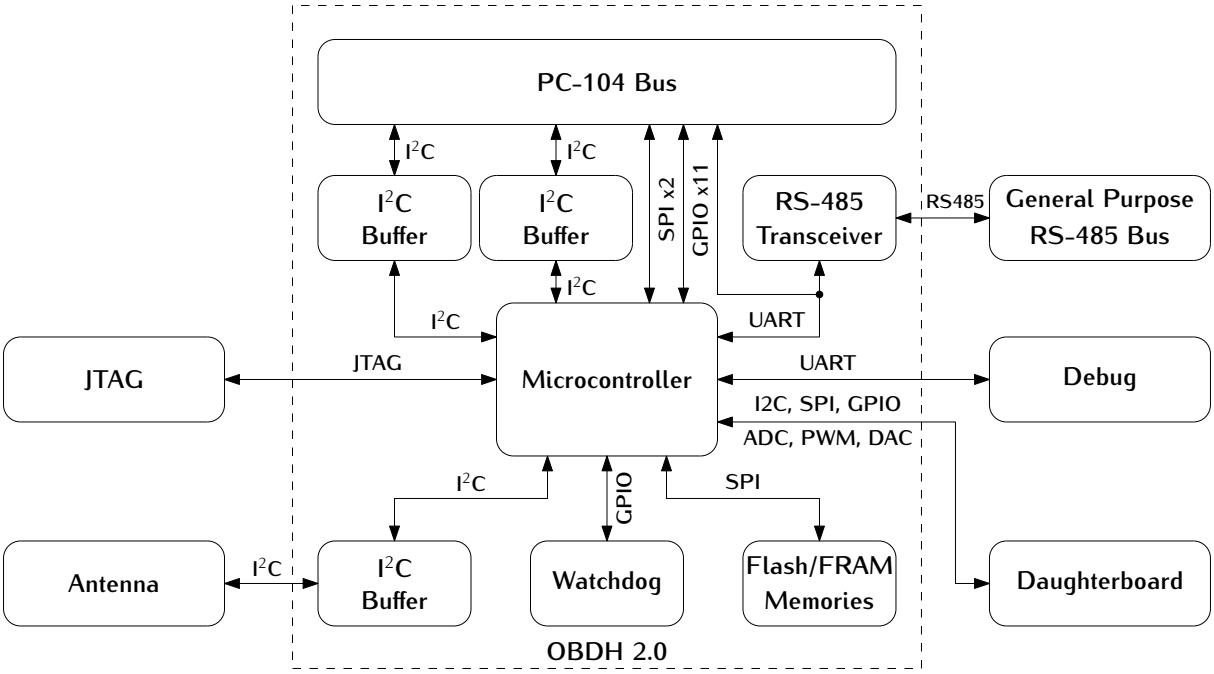


Figure 2.2: OBDH 2.0 Block diagram.

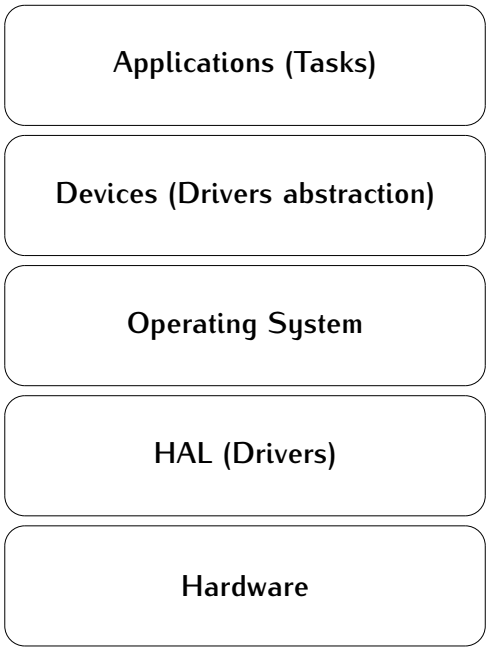


Figure 2.3: System layers.

2.4 Operation

The system operates through the sequencial execution of routines (tasks in the context of the operating system) that are scheduled and multiplexed along the time. Each routine has a priority and a periodicity, which determine the following execution, the set of functionalities currentely running, and the memory usage management. Besides this deterministic scheduling system, the routines have communication channels with each other

through the usage of queues, which provides a robust synchorization scheme. In the chapter 4 the system operation and the internal nuances are described in detail. Then, this section use a top view user perspective to describe the module operation.

2.4.1 Execution Flow

2.4.2 Data Flow

2.4.3 Status LEDs

On the development version of the board, there are eight LEDs that indicates some behaviours of the systems. This set of LEDs can be seen on Figure 2.4.

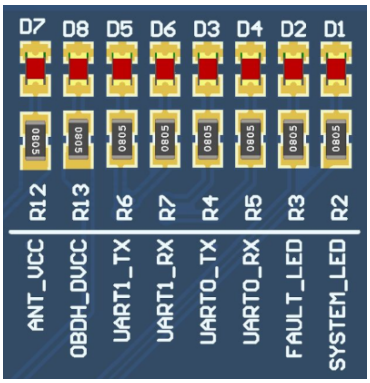


Figure 2.4: Available status LEDs.

A description of each of these LEDs are available below:

- **D1 - System LED**: Heartbeat of the system. Blinks at a frequency of 1 Hz when the system is running properly.
- **D2 - Fault LED**: Indicates a critical fault in the system.
- **D3 - UART0 TX**: Blinks when data is being transmitted over the UART0 port.
- **D4 - UART0 RX**: Blinks when data is being received over the UART0 port.
- **D5 - UART1 TX**: Blinks when data is being transmitted over that UART1 port.
- **D6 - UART1 RX**: Blinks when data is being received over the UART1 port.
- **D7 - Antenna VCC**: Indicates that the antenna module board is being power sourced.
- **D8 - OBDH VCC**: Indicates that the OBDH board is being power sourced.

These LEDs are not mounted in the flight version of the module.

CHAPTER 3

Hardware

The OBDH 2.0 architecture focus on low-power operation and low-cost production, maintaining performance and proposing different approaches to increase the overall reliability. Therefore, the board was developed using these criteria and the changes from the original design were necessary to improve bottlenecks and achieve the requirements of further space mission. The Figure 2.2 presents the module architecture from the hardware perspective, including the main PCB components and interfaces: microcontroller, buffers, transceivers, memory, watchdog and voltage monitor, and connectors. In the following sections, the hardware design, interfaces, and standards are described in detail. The Figures 3.1, 3.2 and 3.3 present 3D rendered images of the top, bottom and side views of the board, respectively.

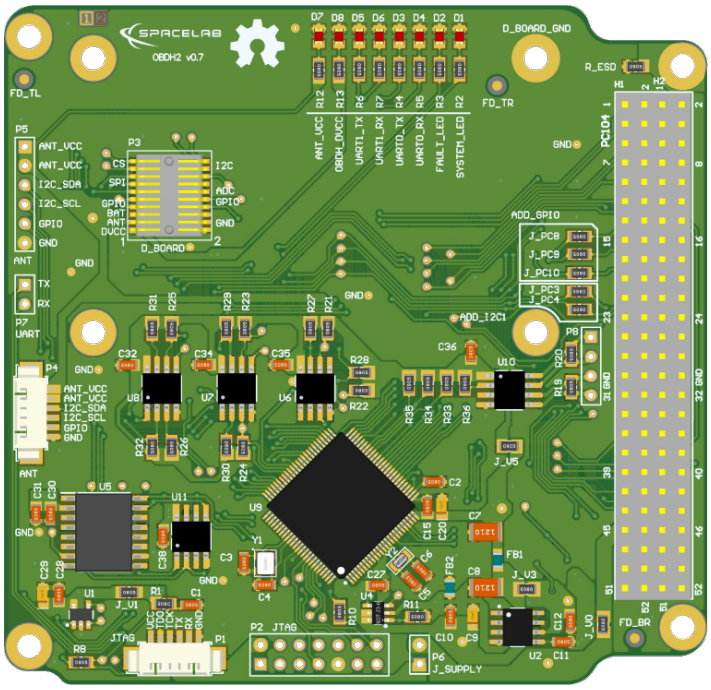


Figure 3.1: Top side of the PCB.

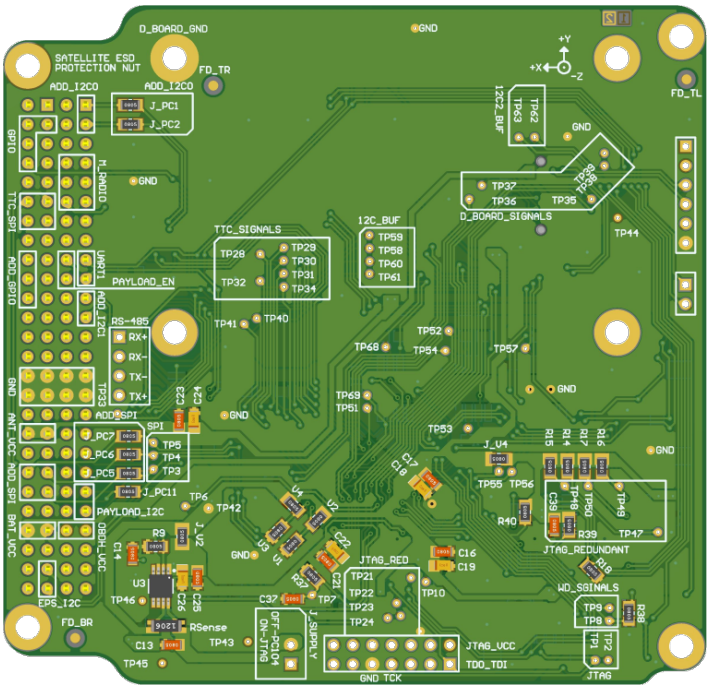


Figure 3.2: Bottom side of the PCB.

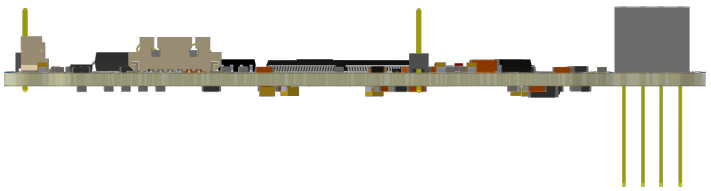


Figure 3.3: Side view of the PCB.

3.1 Interfaces

The Figure 3.4 presents the board interfaces, which consists of communication with other modules, debug access points, and internal peripherals. From the perspective of the microcontroller, there are 6 individual and shared communication buses and the JTAG interface, in the following scheme: A0-SPI (shared with Radio, TTC, and external memory chip); A1-UART (shared with redundant payloads); A2-UART (dedicated for debug); B0-I2C (dedicated for the payload); B1-I2C (dedicated for the EPS); B2-I2C (dedicated for the Antenna module). Currently, "*Payload 1*" and "*Payload 2*" are "*Payload-X*" and "*Payload EDC*" respectively.

3.2 External Connectors

The external interfaces are connected to the microcontroller using different connector types: EPS, TTC, Radio, and Payloads through PC-104; Antenna module with 6H header and 6P picoblade connectors; JTAG through 14H header and 6P picoblade connectors;

[2]The communication protocol of the payload ports depends of the used payload.

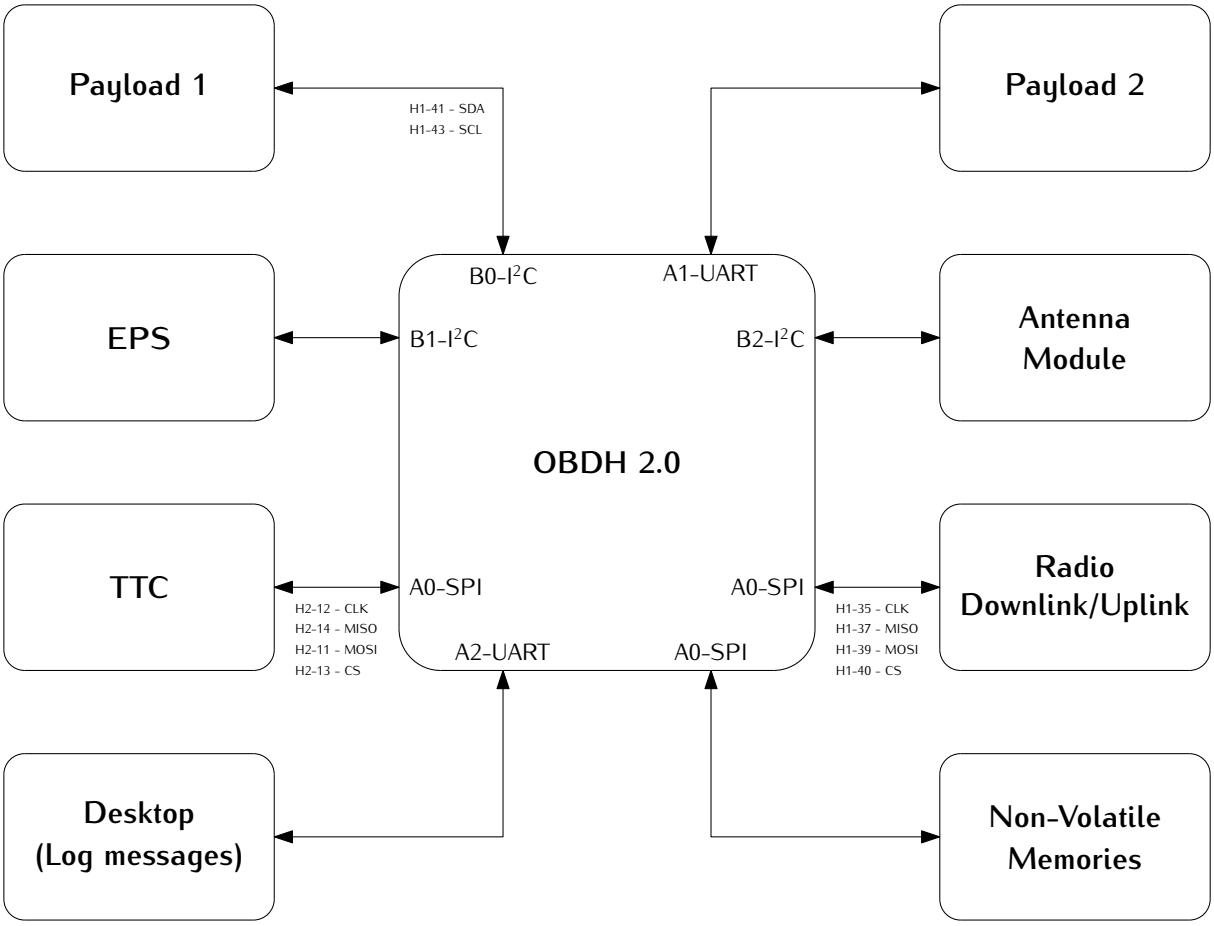


Figure 3.4: Interfaces diagram.

Peripheral	USCI	Protocol	Comm. Protocol
TTC	A0	SPI	Register read/write
Radio (downlink/link)	A0	SPI	Radio config./NGHam
NOR Memory	A0	SPI	-
FRAM Memory	A0	SPI	-
Payload port	A1	UART	-[1]
PC (log messages)	A2	UART	ANSI messages
Payload port	B0	I^2C	-[2]
EPS	B1	I^2C	Register read/write
Antenna Module	B2	I^2C	-

Table 3.1: Boards interfaces.

and debug access using a dedicated 2H header and shared with the JTAG connectors. The following topics describe these interfaces and present the connectors pinout.

3.2.1 PC-104

The connector referred as PC-104 is a junction of two double row 28H headers (*SSW-126-04-G-D*). These connectors create a solid 104-pin interconnection across the different

satellite modules. The Figure 3.5 shows the PC-104 interface from the bottom side of the PCB, which allows visualize the simplified label scheme in the board. Also, the Table 3.2 provides the connector pinout[3] for the pins that are connected to the module.

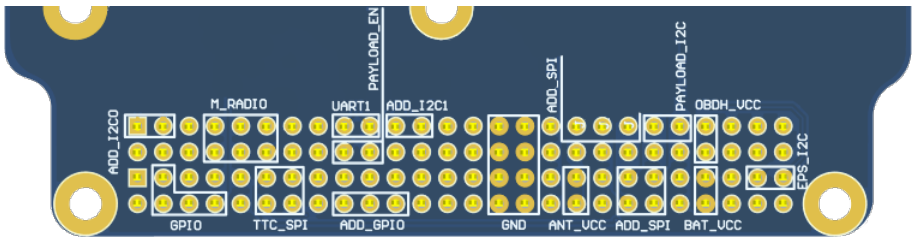


Figure 3.5: Bottom view of PC-104 and simplified labels

Pin [A-B]	H1A	H1B	H2A	H2B
1-2	-	-	-	-
3-4	-	-	GPIO_4	GPIO_5
5-6	-	-	-	-
7-8	GPIO_0	GPIO_1	-	GPIO_6
9-10	GPIO_2	-	-	-
11-12	GPIO_3	GPIO_7	SPI_0_MOSI	SPI_0_CLK
13-14	-	-	SPI_0_CS_1	SPI_0_MISO
15-16	-	-	-	-
17-18	UART_1_RX	GPIO_8	-	-
19-20	UART_1_TX	GPIO_9	-	-
21-22	-	-	-	-
23-24	-	-	-	-
25-26	-	-	-	-
27-28	-	-	-	-
29-30	GND	GND	GND	GND
31-32	GND	GND	GND	GND
33-34	-	-	-	-
35-36	SPI_0_CLK	-	VCC_3V3_ANT	VCC_3V3_ANT
37-38	SPI_0_MISO	-	-	-
39-40	SPI_0_MOSI	SPI_0_CS_0	-	-
41-42	I2C_0_SDA	-	-	-
43-44	I2C_0_SCL	-	-	-
45-46	VCC_3V3	VCC_3V3	VCC_BAT	VCC_BAT
47-48	-	-	-	-
49-50	-	-	I2C_1_SDA	-
51-52	-	-	I2C_1_SCL	-

Table 3.2: PC-104 connector pinout.

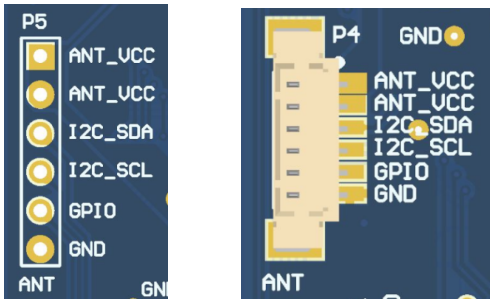
[3]This pinout is simplified since additional interfaces were omitted. Refer to *option sheet* in chapter 5.

3.2.2 Antenna Module

The communication with the Antenna module is performed through external connectors, which are presented in the Figure 3.6. Both connectors have the same connections, but the 3.6(a) (6H header) is used for development and the 3.6(b) (6P picoblade) as the connector for the flight model. This interface consists of a dedicated I2C, power supply, and GPIO, which are described in the Table 3.3.

Pin	Row
1	VCC_3V3_ANT
2	VCC_3V3_ANT
3	I2C_SDA
4	I2C_SCL
5	GPIO
6	GND

Table 3.3: Antenna module connectors pinout.



(a) Debug interface of the antenna module.

(b) Main interface of the antenna module.

Figure 3.6: Antenna module conectors.

3.2.3 Programmer and Debug

The interface with the microcontroller programmer is performed through external connectors, which are presented in the Figure 3.7. Both connectors have the same JTAG and UART interfaces, but the 14H header is used during development and the 6P picoblade (provide more compact and reliable attachment) as the connector for the flight model, which are described in the Table 3.4 and Table 3.5, respectively. This interface consists of a dedicated debug UART, a JTAG, and an external power supply. The debug UART connection has another access point in a dedicated 2H header (P7), as shown in Figure 3.8. Also, to use this external supply, it is necessary to connect both pins of a 2H header jumper (P6).

Figure 3.7: Programmer (P1 and P2) and jumper (P6) connectors.

Pin [A-B]	Row A	Row B
1-2	TDO_TDI	VCC_3V3
3-4	-	-
5-6	-	-
7-8	TCK	-
9-10	GND	-
11-12	-	UART_TX
13-14	-	UART_RX

Table 3.4: Programmer header connector pinout.

Pin	Row
1	VCC_3V3
2	TDO_TDI
3	TCK
4	UART_TX
5	UART_RX
6	GND

Table 3.5: Programmer picoblade connector pinout.

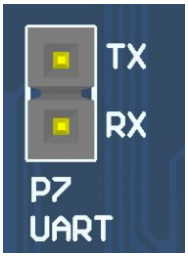


Figure 3.8: Dedicated UART debug connectors (P7).

3.2.4 Daughterboard

The daughterboard interface uses the Samtec FSI-110-D connector [4], which can be seen in the Figure 3.9. This connector has metal contacts in format of arcs that are flexible and four polymer guide pins (a pair for top and bottom). When the daughterboard is attached, there is some pressure to this metal contacts that bend and create a meaningful

pin connection to the daughterboard copper pads[4]. A picture of this connector on the PCB can be seen in Figure 3.10.

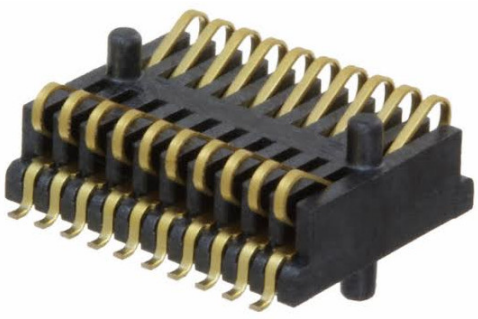


Figure 3.9: Samtec FSI-110-03-G-D-AD connector.

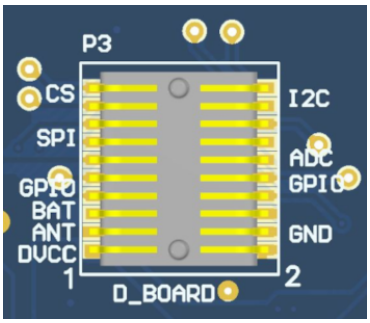


Figure 3.10: Daughterboard connector (P3).

The pinout of the daughterboard interface are available in the Table 3.6. There are different power supply lines (OBDH, Antenna, and battery), communication buses (I2C and SPI), GPIO, and ADC interfaces available. Besides the GPIO and ADC pins, the other interfaces are shared with other modules and peripherals.

Guidelines

The recommended shape and size of the daughterboard can be seen in the Figure 3.11. Besides that, there are mandatory and suggested elements placement: four M3 holes for mechanical attachment, required; contact connector pads (in light gray on the bottom layer), required; two debug headers in the left and bottom sides, suggested; and a general purpose flight model picoblade, suggested.

3.3 Microcontroller

The OBDH 2.0 uses a low-power and low-cost microcontroller family from Texas Instruments, the MSP430F6659 [5]. This device provide sufficient performance for low and medium complexity software and algorithms, allowing the module to execute the required

[4]These daughterboard pads are similar to the ones used as footprint in the OBDH, despite a slightly bigger size.

Pin [A-B]	Row A	Row B
1-2	VCC_3V3	GND
3-4	VCC_3V3_ANT	GND
5-6	VCC_BAT	GND
7-8	GPIO_0	GPIO_1
9-10	GPIO_2	GPIO_3
11-12	SPI_0_CLK	ADC_0
13-14	SPI_0_MISO	ADC_1
15-16	SPI_0_MOSI	ADC_2
17-18	SPI_0_CS_0	I2C_2_SDA
19-20	SPI_0_CS_1	I2C_2_SCL

Table 3.6: Daughterboard connector pinout.

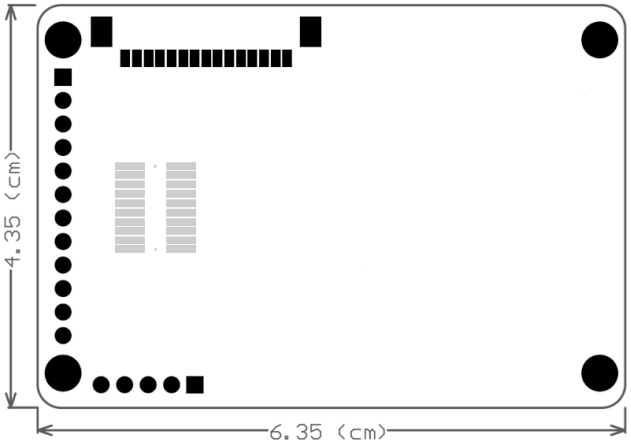


Figure 3.11: Recommended shape and size of the daughterboard.

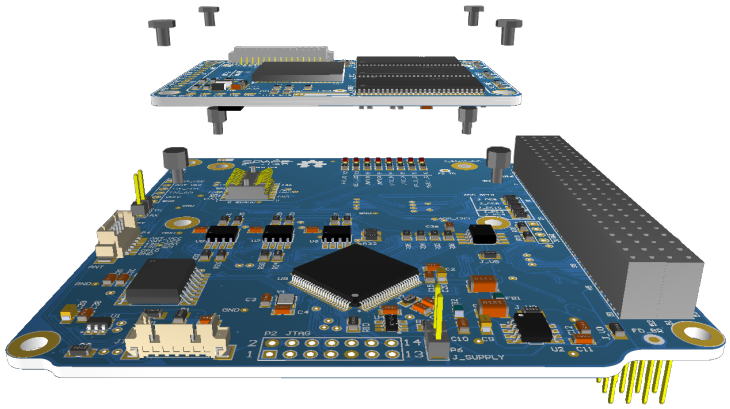


Figure 3.12: Illustrative daughterboard integration.

tasks. The Table 3.7 presents a summary of the main available features and Figure 3.13 shows the internal subsystems, descriptions, and peripherals. The microcontroller interfaces, configurations, and auxiliary components are described in the following topics.

Flash	SRAM	Timers	USCI	ADC	DAC	GPIO
512KB	64KB	2	6 (SPI / I2C / UART)	12	2	74

Table 3.7: Microcontroller features summary.

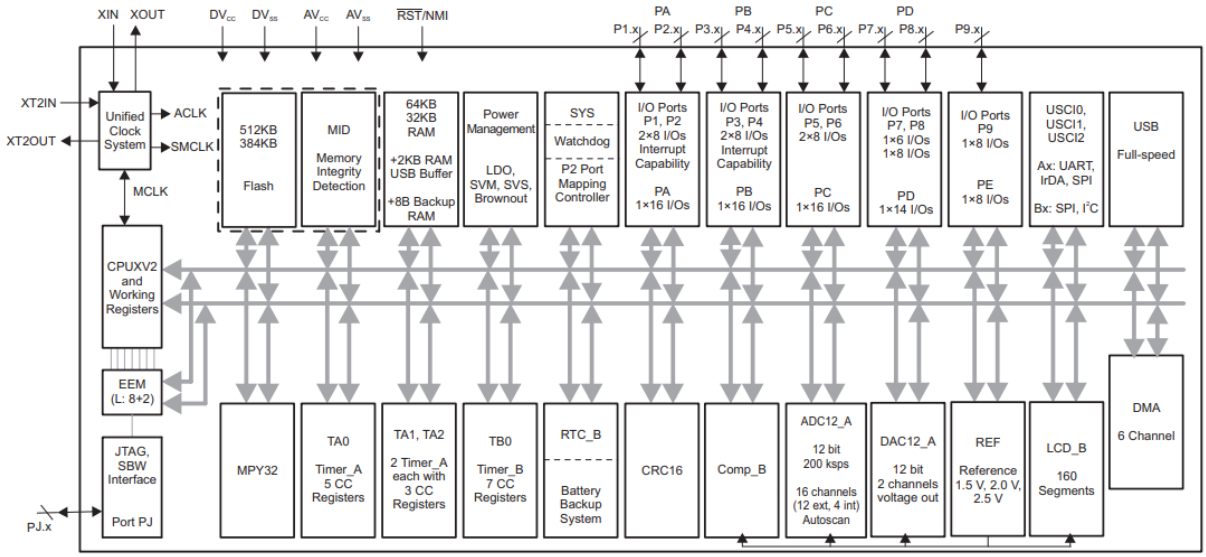


Figure 3.13: Microcontroller internal diagram.

3.3.1 Interfaces Configuration

The microcontroller has 6 Universal Serial Communication Interfaces (USCI) that can be configured to operate with different protocols and parameters. These interfaces are connected to different modules and peripherals, as presented in the Figure 3.4. The Table 3.8 describes each interface configurations.

Interface	Protocol (Index)	Mode	Word Length	Data Rate	Configuration
USCI_A0	SPI	Master	8 bits	1 Mbps	Phase: High Polarity: Low
USCI_A1	UART1	-	8 bits	115200 bps	Stop bits: 1 Parity: None
USCI_A2	UART0	-	8 bits	115200 bps	Stop bits: 1 Parity: None
USCI_B0	I2C0	Master	8 bits	100 kbps	Adr. len: 7 bits
USCI_B1	I2C1	Master	8 bits	100 kbps	Adr. len: 7 bits
USCI_B2	I2C2	Master	8 bits	100 kbps	Adr. len: 7 bits

Table 3.8: USCI configuration.

3.3.2 Clocks Configuration

Besides the internal clock sources, the microcontroller has two dedicated clock inputs for external crystals: the main clock and the auxiliary clock inputs. There are a 32 *MHz*

crystal and a 32.769 *kHz* connected to these inputs, respectively. The first source is used for generating the Master Clock (MCLK) and the Subsystem Master Clock (SMCLK), which are used by the CPU and the internal peripheral modules. The second source is used for generating the Auxiliary Clock (ACLK) that handles the low-power modes and might be used for peripherals.

3.3.3 Pinout

An illustration of the microcontroller pinout positions can be seen in the Figure 3.14. The Table 3.9 presents the OBDH 2.0 microcontroller pins assignment.

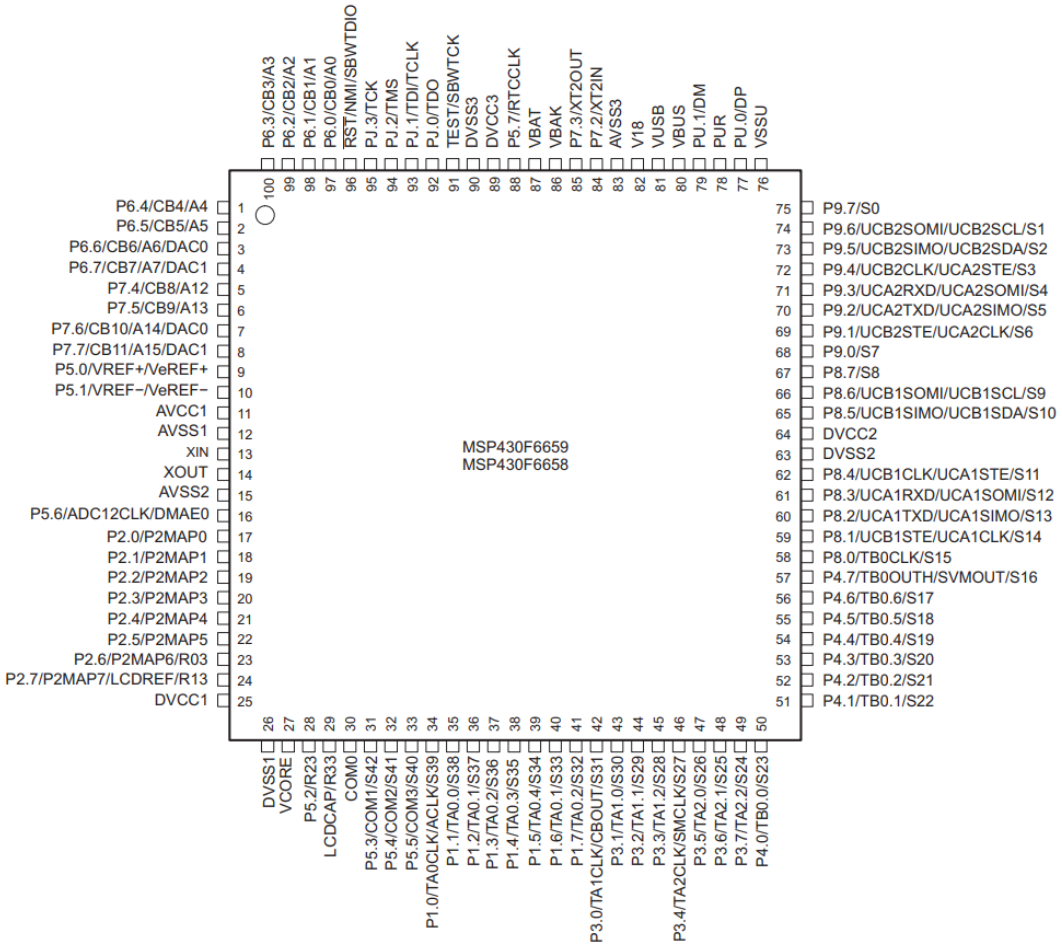


Figure 3.14: Microcontroller pinout positions.

Pin Code	Pin Number	Signal
P1.0	34	MAIN_RADIO_ENABLE
P1.1	35	MAIN_RADIO_GPIO0
P1.2	36	MAIN_RADIO_GPIO1
P1.3	37	MAIN_RADIO_GPIO2
P1.4	38	MAIN_RADIO_RESET
P1.5	39	MAIN_RADIO_SPI_CS
P1.6	40	TTC_MCU_SPI_CS

P1.7	41	-
P2.0	17	SPI_CLK
P2.1	18	I2C0_SDA
P2.2	19	I2C0_SCL
P2.3	20	-
P2.4	21	SPI_MOSI
P2.5	22	SPI_MISO
P2.6	23	-
P2.7	24	-
P3.0	42	I2C0_EN
P3.1	43	I2C1_EN
P3.2	44	I2C2_EN
P3.3	45	I2C0_READY
P3.4	46	I2C1_READY
P3.5	47	I2C2_READY
P3.6	48	PC104_GPIO0
P3.7	49	PC104_GPIO1
P4.0	50	PC104_GPIO2
P4.1	51	PC104_GPIO3
P4.2	52	MEM_HOLD
P4.3	53	MEM_RESET
P4.4	54	MEM_SPI_CS
P4.5	55	-
P4.6	56	-
P4.7	57	-
P5.0	9	VREF
P5.1	10	AGND
P5.2	28	SYSTEM_FAULT_LED
P5.3	31	SYSTEM_LED
P5.4	32	PAYLOAD_0_ENABLE
P5.5	33	PAYLOAD_1_ENABLE
P5.6	16	-
P5.7	88	-
P6.0	97	D_BOARD_ADC0
P6.1	98	D_BOARD_ADC1
P6.2	99	D_BOARD_ADC2
P6.3	100	OBDH_CURRENT_ADC
P6.4	1	OBDH_VOLTAGE_ADC
P6.5	2	D_BOARD_SPI_CS0
P6.6	3	D_BOARD_SPI_CS1
P6.7	4	-
P7.0	-	-
P7.1	-	-
P7.2	84	XT2_N
P7.3	85	XT2_P
P7.4	5	D_BOARD_GPIO0

P7.5	6	D_BOARD_GPIO1
P7.6	7	D_BOARD_GPIO2
P7.7	8	D_BOARD_GPIO3
P8.0	58	-
P8.1	59	-
P8.2	60	UART1_TX
P8.3	61	UART1_RX
P8.4	62	-
P8.5	65	I2C1_SDA
P8.6	66	I2C1_SCL
P8.7	67	ANTENNA_GPIO
P9.0	68	-
P9.1	69	FRAM_SPI_CS
P9.2	70	UART0_TX
P9.3	71	UART0_RX
P9.4	72	WDI_EXT
P9.5	73	I2C2_SDA
P9.6	74	I2C2_SCL
P9.7	75	MR_WDOG
PJ.0	92	TP21
PJ.1	93	TP22
PJ.2	94	TP23
PJ.3	95	TP24
-	13	XT1IN
-	14	XT1OUT
-	96	JTAG_TDO_TDI
-	91	JTAG_TCK

Table 3.9: Microcontroller pinout and assignments.

3.4 External Watchdog

Additionally to the internal watchdog timer of the microcontroller, to ensure a system reset in case of a software freeze, an external watchdog circuit is being used. For that, the TPS3823 IC from Texas Instruments [6] was chosen. This IC is a voltage monitor with a watchdog timer feature. This circuit can be seen in the Figure 3.15.

This circuit works this way: if the WDI pin remains high or low longer than the timeout period, then reset is triggered. The timer clears when reset is asserted or when WDI sees a rising edge or a falling edge.

The watchdog timer task clears the TPS3823 timer by toggling the WDI pin at every 100 *ms*. If the WDI pin state stays unmodified for more than 1600 *ms*, the reset pin is cleared and the microcontroller is reseted.

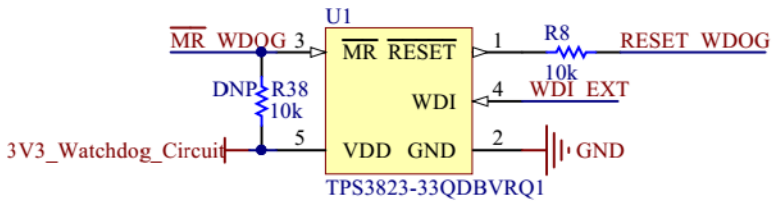


Figure 3.15: External watchdog timer circuit.

3.5 Non-Volatile Memories

There are two non-volatile memories available on the module: one flash NOR memory and one FRAM memory.

3.5.1 Flash NOR

The flash NOR non-volatile memory model is the Micron MT25QL01GBBB, which is composed by a NOR flash architecture with 1 Gb of capacity (or 128 MB) and features extended SPI configurations. As can be seen in Figure 3.4, a SPI bus is used to communicate with this peripheral, using the Table 3.8 configurations. Also, there are some control pins that are connected to microcontroller GPIOs: HOLD#, RESET#, and W#.

When RESET# is driven LOW, the device is reset and the outputs are tri-stated. The HOLD# signal pauses serial communications without deselecting or resetting the device, consequently outputs are tri-stated and inputs are ignored. The W# signal handles as a write protection, which freezes the status register, turning its non-volatile bits read-only and preventing the write operation to be executed.

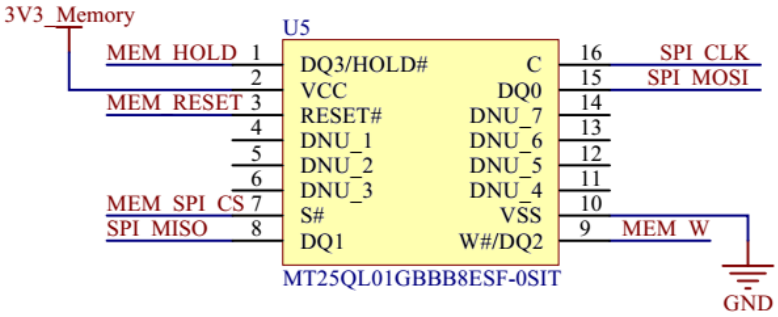


Figure 3.16: External memory circuit.

3.5.2 FRAM

3.6 I2C Buffers

The microcontroller I2C interfaces have dedicated IC buffers, which improve the signal quality throughout the various connectors and offers reliability enhancements, since it protects the bus in case of failures. This measure was adopted in all the satellite modules

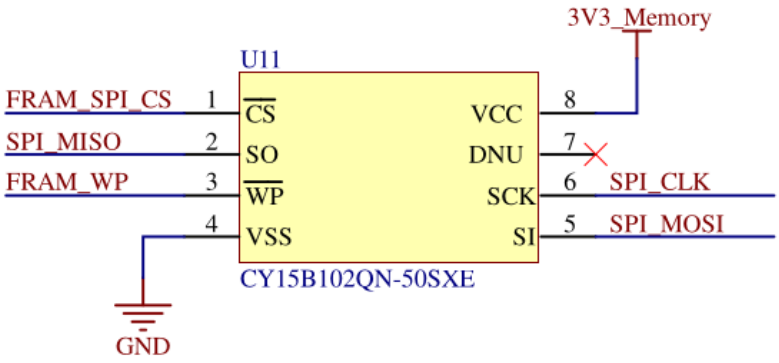


Figure 3.17: FRAM memory circuit.

due to previous failures in I2C buses. Using this scheme, the modules connected though this protocol might have shared connections without losing performance or reliability.

The buffer selected for this function is the Texas Instruments TCA4311 device. Besides the I2C inputs and outputs, it features control and status signals that are connected to GPIOs in the microcontroller: an enable and an operation ready status. Also, both inputs and outputs in these I2C lines have external pull-up resistors.

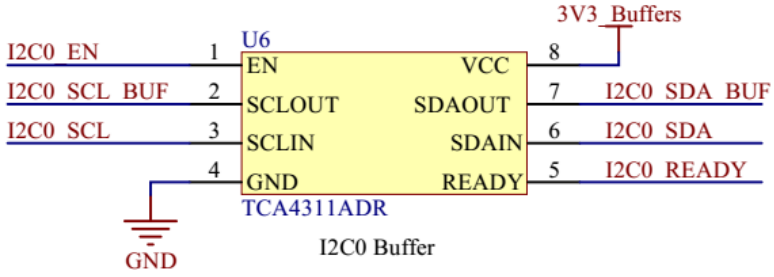


Figure 3.18: I2C buffer circuit.

3.7 RS-485 Transceiver

The module features an RS-485 interface, which is connected to a 4H header (P8). This interface uses a transceiver (THVD1451) to convert the incoming RS-485 signals to UART and vice-versa. The outputs are 120 Ω differential pairs that have termination resistors before connecting to the header pins.

3.8 Voltage and Current Sensors

In order to monitor the board overall current and voltage, the module have a current sensor using a Maxim Integrated IC (MAX9934) and a buffered voltage divider circuit with a Texas Instruments IC (TLV341A). These circuits have direct analog outputs that are connected to ADC inputs. The microcontroller internal ADC peripheral has a dedicated

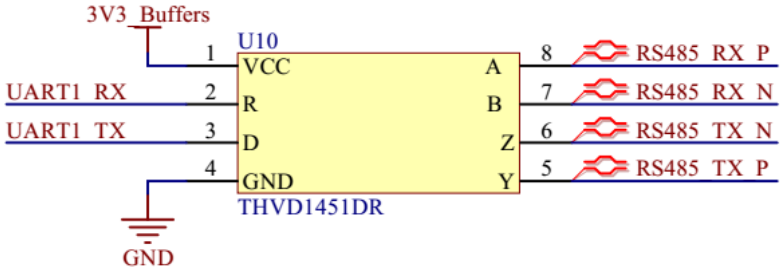


Figure 3.19: RS-485 transceiver circuit.

input for a voltage reference, which is connected to the REF5030A IC. This device generates a precise 3 V output that enhances the measures and convertions performed by the microcontroller.

CHAPTER 4

Firmware

4.1 Product tree

The product tree of the firmware part of the OBDH 2.0 module is available in Figure 4.1.

4.2 Dependencies

The firmware depends on external libraries to access the embedded hardware or to communicate with other modules. A list of these libraries and the used version is available in Table 4.1.

Library	Version
MSP430 DriverLib	v2.91.11.01
FreeRTOS	v10.2.1
NGHam protocol	-
libcsp	v1.5.16

Table 4.1: External libraries and dependencies of the firmware.

4.3 Tasks

A list of the firmware tasks can be seen in the Table 4.2.
All these tasks are better described below.

4.3.1 Antenna deployment

.

4.3.2 Antenna reading

.

4.3.3 Beacon

.

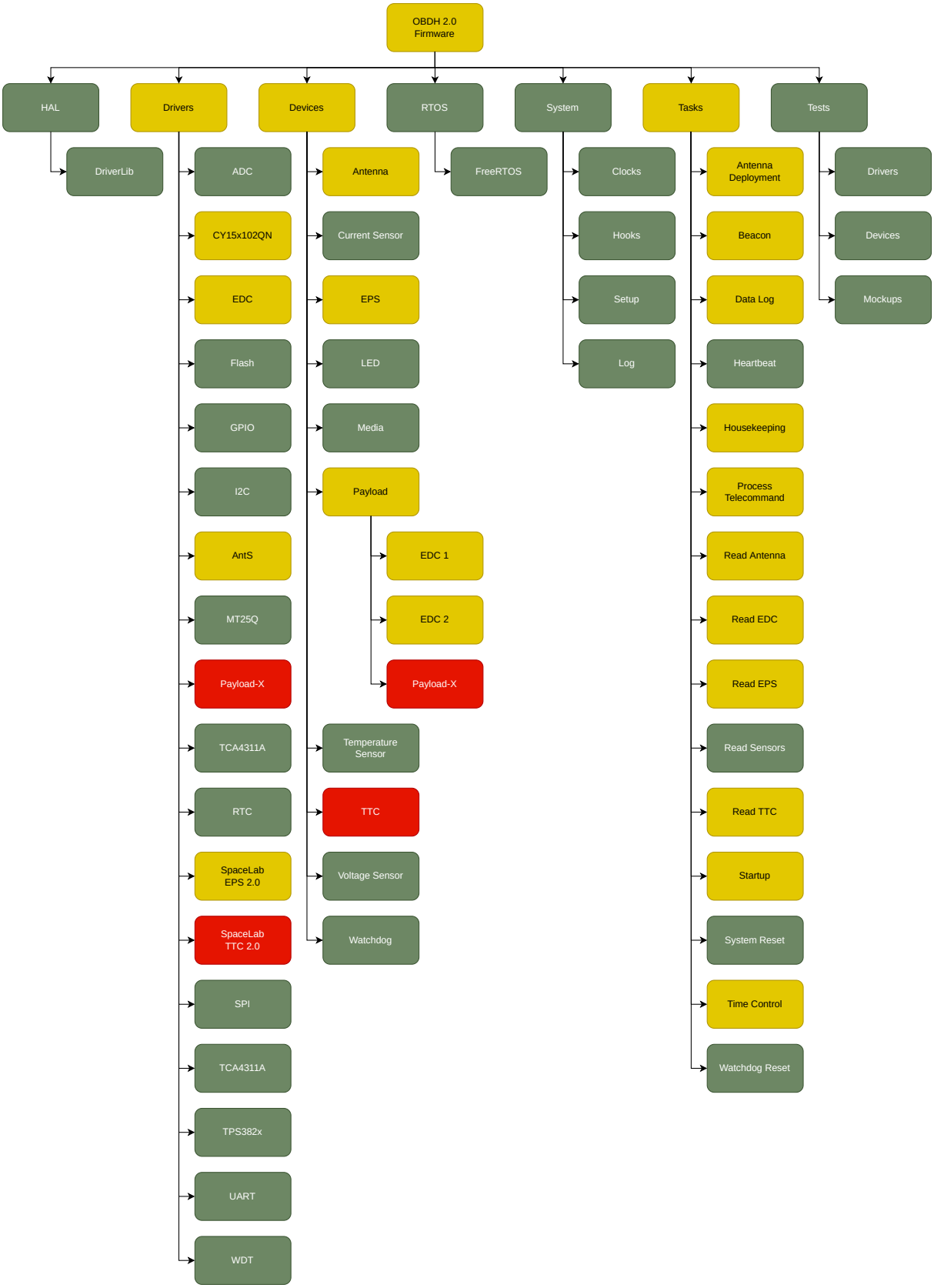


Figure 4.1: Product tree of the firmware of the OBDH 2.0 module.

Name	Priority	Initial delay [ms]	Period [ms]	Stack [bytes]
Antenna deployment	Highest	0	Aperiodic	150
Antenna reading	Medium	2000	60000	150
Beacon	High	1000	60000	1000
Data log	Medium	2000	600000	225
EDC reading	Medium	2000	60000	128
EPS reading	Medium	2000	60000	384
Heartbeat	Lowest	2000	500	160
Housekeeping	Medium	2000	10000	160
Payload X reading	Medium	5000	5000	TBD
Radio periodoc reset	High	60000	60000	128
Read sensors	Medium	2000	60000	140
Startup (boot)	Highest	0	Aperiodic	350
System reset	High	0	36000000	128
Telecommand processing	High	1000	5000	500
Time control	Medium	1000	1000	128
TTC reading	Medium	2000	60000	384
Uplink	Low	1000	10000	500
Watchdog reset	Lowest	0	100	150

Table 4.2: Firmware tasks.

4.3.4 Data log

This tasks saves the housekeeping data of the satellite in the flash memory at every 10 minutes.

4.3.5 EDC reading

.

4.3.6 EPS reading

.

4.3.7 Heartbeat

The heartbeat task keeps blinking a LED ("*System LED*" in Figure 2.4) at a rate of 1 Hz during the execution of the system. Its purpose is to give a visual feedback of the execution of the scheduler. This is tasks does not have a specific purpose on the flight version of the module (the flight version of the PCB does not have LEDs).

4.3.8 Housekeeping

.

4.3.9 Payload X reading

.

4.3.10 Radio periodic reset

.

4.3.11 Read sensors

This task reads the internal sensors of the OBDH at every 60 seconds.

4.3.12 Startup (boot)

This task is the first executed task when the system starts. All devices, libraries and data structures are initialized in this task. When the execution is done, the remaining tasks of the system are allowed to execute.

4.3.13 System reset

This task resets the microcontroller by software at every 10 hours. This can be useful to cleanup possible wrong values in variables, repeat the antenna deployment routine (limited to *n* times), cleanup the RAM memory, etc.

4.3.14 Telecommand processing

.

4.3.15 Time control

This task is responsible for the time management of the system. At every second it increments the system time (epoch). Also, at every minute it saves the current system time in the non-volatile memory.

4.3.16 TTC reading

.

4.3.17 Uplink

.

4.3.18 Watchdog reset

This task resets the internal and external watchdog timer at every 100 ms. The internal watchdog has a maximum count time of 500 ms, and the external watchdog a maximum of 1600 ms (see chapter 3 for more information about the watchdog timers).

To prevent the system to not reset during an anomaly on some task (like an execution time longer than planned), this task has lowest possible priority: 0.

4.4 Variables and Parameters

The internal variables and parameters of the OBDH firmware can be seen in Table 4.3.

ID	Name/Description	Type
0	Time counter in milliseconds	uint32
1	Temperature of the μC in Kelvin	uint16
2	Input current in mA	uint16
3	Input voltage in mV	uint16
4	Last reset cause: - 0x00 = No interrupt pending - 0x02 = Brownout (BOR) - 0x04 = RST/NMI (BOR) - 0x06 = PMMSWBOR (BOR) - 0x08 = Wakeup from LPMx.5 (BOR) - 0x0A = Security violation (BOR) - 0x0C = SVSL (POR) - 0x0E = SVSH (POR) - 0x10 = SVML_OVP (POR) - 0x12 = SVMH_OVP (POR) - 0x14 = PMMSWPOR (POR) - 0x16 = WDT time out (PUC) - 0x18 = WDT password violation (PUC) - 0x1A = Flash password violation (PUC) - 0x1C = Reserved - 0x1E = PERF peripheral/configuration area fetch (PUC) - 0x20 = PMM password violation (PUC) - 0x22 to 0x3E = Reserved	uint8
5	Reset counter	uint16
6	Last valid telecommand (uplink packet ID)	uint8
7	Temperature of the radio in Kelvin	uint16
8	RSSI of the last valid telecommand	uint16
9	Temperature of the antenna in Kelvin	uint16
10	Antenna status bits: - Bit 15: The antenna 1 is deployed (0) or not (1) - Bit 14: Cause of the latest activation stop for antenna 1 - Bit 13: The antenna 1 deployment is active (1) or not (0) - Bit 11: The antenna 2 is deployed (0) or not (1) - Bit 10: Cause of the latest activation stop for antenna 2 - Bit 9: The antenna 2 deployment is active (1) or not (0) - Bit 8: The antenna is ignoring the deployment switches (1) or not (0) - Bit 7: The antenna 3 is deployed (0) or not (1) - Bit 6: Cause of the latest activation stop for antenna 3 - Bit 5: The antenna 3 deployment is active (1) or not (0) - Bit 4: The antenna system independent burn is active (1) or not (0) - Bit 3: The antenna 4 is deployed (0) or not (1) - Bit 2: Cause of the latest activation stop for antenna 4 - Bit 1: The antenna 4 deployment is active (1) or not (0)	uint16

	- Bit 0: The antenna system is armed (1) or not (0)	
11	Hardware version	uint8
12	Firmware version (ex.: "v1.2.3" = 0x00010203)	uint32
13	Mode ("Normal" = 0, "Hibernation" = 1)	uint8
14	Timestamp of the last mode change	uint32
15	Mode duration in sec. (valid only in hibernation mode)	uint32
16	Initial hibernation executed	boolean
17	Initial hibernation time counter (minutes)	uint8
18	Antenna deployment executed	boolean
19	Antenna deployment counter	uint8

Table 4.3: Variables and parameters of the OBDH 2.0.

4.5 Telemetry

4.5.1 Beacon

The beacon packet is transmitted at every 1 minute and contains a basic telemetry data of the satellite. The content of this packet can be seen in Table 4.4.

- Period: 60 seconds
- Band: UHF
- Condition to operate: Always on

Parameter	**Content**	**Length [bytes]**
Packet ID	10h	1
Satellite callsign	"0PY0EGU"	7
μC temperature	Raw μC temperature	2
μC voltage	Raw μC voltage	2
μC current	Raw μC current	2
Last reset cause	Last reset cause ID	1
System time	System time in ticks	4
Radio temperature	Raw radio temperature	4
Last TC RSSI	Raw RSSI value	2
Last received TC	Last received TC ID	1
Battery 1 voltage	Raw battery 1 voltage	2
Battery 2 voltage	Raw battery 2 voltage	2
Battery current	Raw battery current	2
Battery charge	Raw battery charge	2
...	...	...
Total	-	34

Table 4.4: Beacon packet.

4.5.2 EDC Information

Parameter	Content	Len. [bytes]
Packet ID	11h	1
Satellite callsign	"0PY0EGU"	7
	PTT Decoder	
Time tag	PTT signal receiving time	4
Error code	Error code	1
Carrier frequency	Carrier frequency	2
Carrier Abs	Carrier amplitude at ADC interface output	2
Message length	User message length in bytes	1
User message	ARGOS-2 PTT-A2 user message	35
	HK Info	
Current time	Current time since J2000 epoch	4
Elapsed time	Elapsed time since last reset	4
Current supply	System current supply in mA	2
Voltage supply	System voltage supply in mV	2
Temperature	EDC board temperature	1
PLL sync bit	RF front end LO...	1
ADC RMS	RMS level at front-end output	2
Num of RX PTT	Generated PTT packages since last initialization	1
Max		1
Memory error count		1
	System State	
Current time		4
PTT available	Number of PTT Package available for reading	1
PTT is paused	PTT decoder task status	1
Sampler state	ADC sampler state	1
Total	-	79

Table 4.5: EDC information packet.

4.5.3 EDC Samples

The EDC samples are XX bytes long and are transmitted in Y packets with 219 bytes each

4.6 Telecommands

4.6.1 Enter hibernation

4.6.2 Leave hibernation

Parameter	Content	Length [bytes]
Packet ID	12h	1
Satellite callsign	"0PY0EGU"	7
Time tag	Elapsed time since J2000 epoch	4
Packet counter	ADC sample packet number	1
I sample[n]	First ADC I-sample	2
Q sample[n]	First ADC Q-sample	2
...	...	...
I sample[n+102]	First ADC I-sample	2
Q sample[n+102]	First ADC Q-sample	2
Total	-	219

Table 4.6: EDC samples packet.

Name	Parameters	Access
Enter hibernation	Hibernation period in seconds	Private
Leave hibernation	None	Private
Activate beacon	None	Private
Deactivate beacon	None	Private
Activate downlink	None	Private
Deactivate downlink	None	Private
Activate EDC	None	Private
Deactivate EDC	None	Private
Get EDC info	None	Private
Activate Payload X	Experiment period in seconds	Private
Deactivate Payload X	None	Private
Set system time	Time value (epoch)	Private
Ping	None	Public
Message broadcast	ASCII message	Public
Request data	Data flags	Public

Table 4.7: System telecomamnds.

Parameter	Content	Length [bytes]
Packet ID	20h	1
Ground station callsign	Any callsign (ASCII, filled with "0"s)	7
Hibernation period	Period in minutes (1 to 65535)	2
Key	Telecommand key (ASCII)	10
Total	-	20

Table 4.8: Enter hibernation telecommand.

4.6.3 Activate beacon

.

Parameter	Content	Length [bytes]
Packet ID	21h	1
Ground station callsign	Any callsign (ASCII, filled with "0"s)	7
Key	Telecommand key (ASCII)	10
Total	-	18

Table 4.9: Leave hibernation telecommand.

4.6.4 Deactivate beacon

.

4.6.5 Activate EDC

.

4.6.6 Deactivate EDC

.

4.6.7 Get EDC info

This telecommand request information from the EDC payload. When received, the OBDH transmits the housekeeping and state frames of the EDC module (28 bytes). This telecommand does not requires a key.

4.6.8 Activate Payload X

.

4.6.9 Deactivate Payload X

.

4.6.10 Set system time

.

4.6.11 Ping

4.6.12 Message broadcast

4.6.13 Request data

.

Parameter	Content	Length [bytes]
Packet ID	22h	1
Ground station callsign	Any callsign (ASCII, filled with "0"s)	7
Total	-	8

Table 4.10: Ping telecommand.

Parameter	Content	Length [bytes]
Packet ID	12h	1
Satellite callsign	"PY0EGU"	7
Destination callsign	Requester callsign (ASCII, filled with "0"s)	7
Total	-	15

Table 4.11: Ping telecommand answer.

Parameter	Content	Length [bytes]
Packet ID	23h	1
Ground station callsign	Any callsign (ASCII, filled with "0"s)	7
Message	Message to broadcast (ASCII)	up to
Total	-	8

Table 4.12: Message broadcast telecommand.

4.7 Operating System

As operating system the FreeRTOS 10 [7] is being used. FreeRTOS is a market-leading real-time operating system (RTOS) for microcontrollers and small microprocessors. Distributed freely under the MIT open source license, FreeRTOS includes a kernel and a growing set of IoT libraries suitable for use across all industry sectors. FreeRTOS is built with an emphasis on reliability and ease of use.

The main configuration parameters of the operating system in this project are availabe in Table 4.13.

Parameter	Value	Unit
Version	v10.2.0	-
Tick rate (Hz)	1000	Hz
CPU clock (HZ)	32	MHz
Max. priorities	5	-
Heap size	40960	bytes
Max. length of task name	20	-

Table 4.13: FreeRTOS main configuration parameters.

More details of the used configuration parameters can be seen in the file *firmware/config/FreeRTOSConfig.h* from [3].

4.8 Hardware Abstraction Layer (HAL)

As the Hardware Abstraction Layer (HAL), the DriverLib [8] from Texas Instruments is begin used. It is the official API to access the registers of the MSP430 microcontrollers.

The DriverLib is meant to provide a "software" layer to the programmer in order to facilitate higher level of programming compared to direct register accesses. By using the high level software APIs provided by DriverLib, users can create powerful and intuitive code which is highly portable between not only devices within the MSP430 platform, but between different families in the MSP430/MSP432 platforms.

4.9 Memory Management

.

4.10 Protocols

4.10.1 NGHam

NGHam [9], short for Next Generation Ham Radio, is a set of protocols for packet radio communication. Its usage is similar to the existing AX.25 protocol.

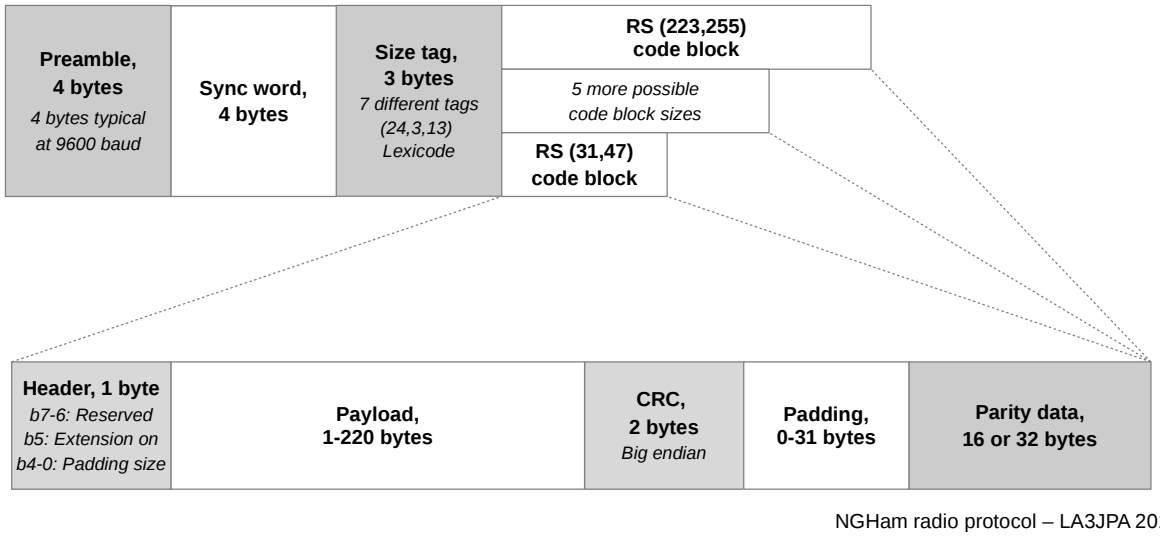


Figure 4.2: NGHam packet structure.

4.10.2 CSP

The CubeSat Space Protocol (CSP) [10] is a small protocol stack written in C. CSP is designed to ease communication between distributed embedded systems in smaller networks, such as CubeSats. The design follows the TCP/IP model and includes a transport protocol, a routing protocol and several MAC-layer interfaces. The core of libcsp includes a router, a connection oriented socket API and message/connection pools.

The idea is to give sub-system developers of CubeSats the same features of a TCP/IP stack, but without adding the huge overhead of the IP header. The small footprint and simple implementation allows a small 8-bit system to be fully connected on the network. This allows all subsystems to provide their services on the same network level, without any master node required. Using a service oriented architecture has several advantages compared to the traditional mater/slave topology used on many cubesats.

The OBDH's firmware currently uses the version v1.5.16 of the libcsp library.

CHAPTER 5

Board Assembly

The OBDH2 has some DNP components to provide flashing, debugging, testing or providing extra interfaces if needed. These components may not be necessary for the flight model of the board. The draftsman document can be viewed for more detailed information regarding their location and board dimensions [11].

5.1 Development Model

5.1.1 Debug and programming connectors

The P2 and P6 connectors are used for flashing and debugging the OBDH2 board. See again chapter 3 (Hardware) and subsection 3.2.3 (Programmer and Debug) for more information.

5.1.2 Status leds

As already exposed before in the document OBDH2 has status leds to be used during development and test phases. See again chapter 2 (System Overview) and the subsection 2.3.1 (Status Leds) for more information.

5.1.3 Analog circuits

TBD

5.2 Flight Model

TBD

5.3 Custom Configuration

On the PC104 connector of OBDH2 there are some jumper resistors to enable extra I2C, SPI and GPIOs interfaces if desired. Note that the I2C0, I2C1 and SPI channels should not be used with shared devices. The corresponding table and location on the pcb of these components are showed below.

Label	Interface
J_PC1	I2C0_SDA
J_PC2	I2C0_SCL
J_PC3	I2C1_SDA
J_PC4	I2C1_SCL
J_PC5	SPI_MOSI
J_PC6	SPI_MISO
J_PC7	SPI_CLK
J_PC8	GPI0
J_PC9	GPIO1
J_PC10	GPIO2
J_PC11	GPIO3

Table 5.1: Additional PC104 inferfaces.

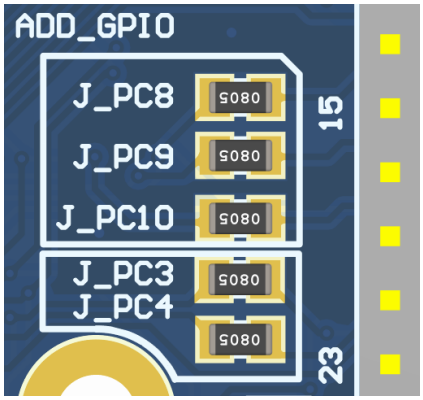


Figure 5.1: Additional GPIOs and I2C channel 1.

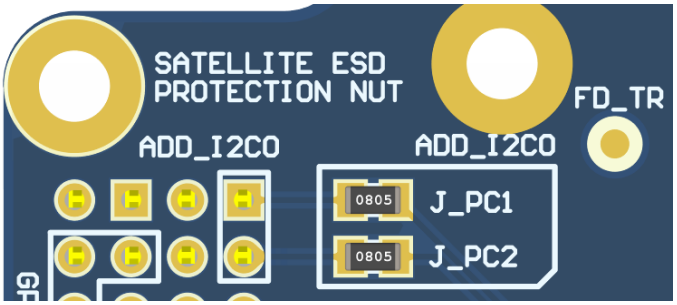


Figure 5.2: Additional I2C channel 0.

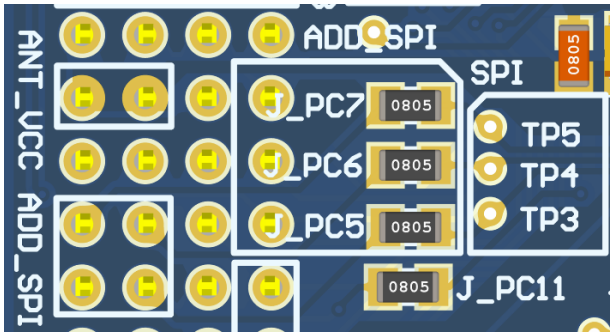


Figure 5.3: Additional GPIO and SPI channel.

CHAPTER 6

Usage Instructions

6.1 Powering the Board

Since the OBDH 2.0 is a service module within a satellite bus, in order to properly provide it power supply, it is required an external 3.3 ± 0.2 V power input and a current capability of at least 100 mA (might change depending on the daughterboard requirements). As presented in the PC-104 and programming interface sections, there is some options concerning supplying power to the module for improving flexibility during development. The board has two power schemes: using the JTAG interface, for debug, or the PC-104, for the flight configuration. The first case uses both P1 or P2 connectors as power input (besides the JTAG and UART interfaces) and requires a jumper connection in the P6 connector. The second uses the PC-104 pins H1-45 and H1-46 to provide the power and the P6 connector should remain open. For pinout details, refer to the external connectors in the hardware chapter.

6.2 Log Messages

The OBDH 2.0 has a UART interface dedicated for debugging, which is described in Table 3.8. It follows a log system structure to improve the information provided in each message. The Figure 6.1 shows an example of the logging system, more specifically the initialization sequence. The messages use the following scheme: in green inside brackets, the timestamp; in magenta, the scope (or origin) of the log; and lastly the actual message, which might be white (info or note), yellow (warning) and red (error).

6.3 Daughterboards Installation

The daughterboard requirements might change for each application board attached. Then, it is important to check at least a minimal set of mandatory characteristics. First it is important to verify mechanical parameters that concerns to size (recommended 63.5×43.5 mm), maximum height (no higher than 7 mm), screws attachment (refer to mechanical sheet [11]) and contact connector positioning. After this, the electrical interface must be checked (refer to subsection 3.2.4). There are 3 different power supply options, a 3.3 V source shared with the OBDH board itself, another 3.3 V source shared with the antenna deployer, and the battery main bus that range from 5.4 to 8.4 V. Lastly, depending on the application board design, it is necessary to check communication interfaces protocols and parameters, control inputs and outputs, and external interfaces with other modules.

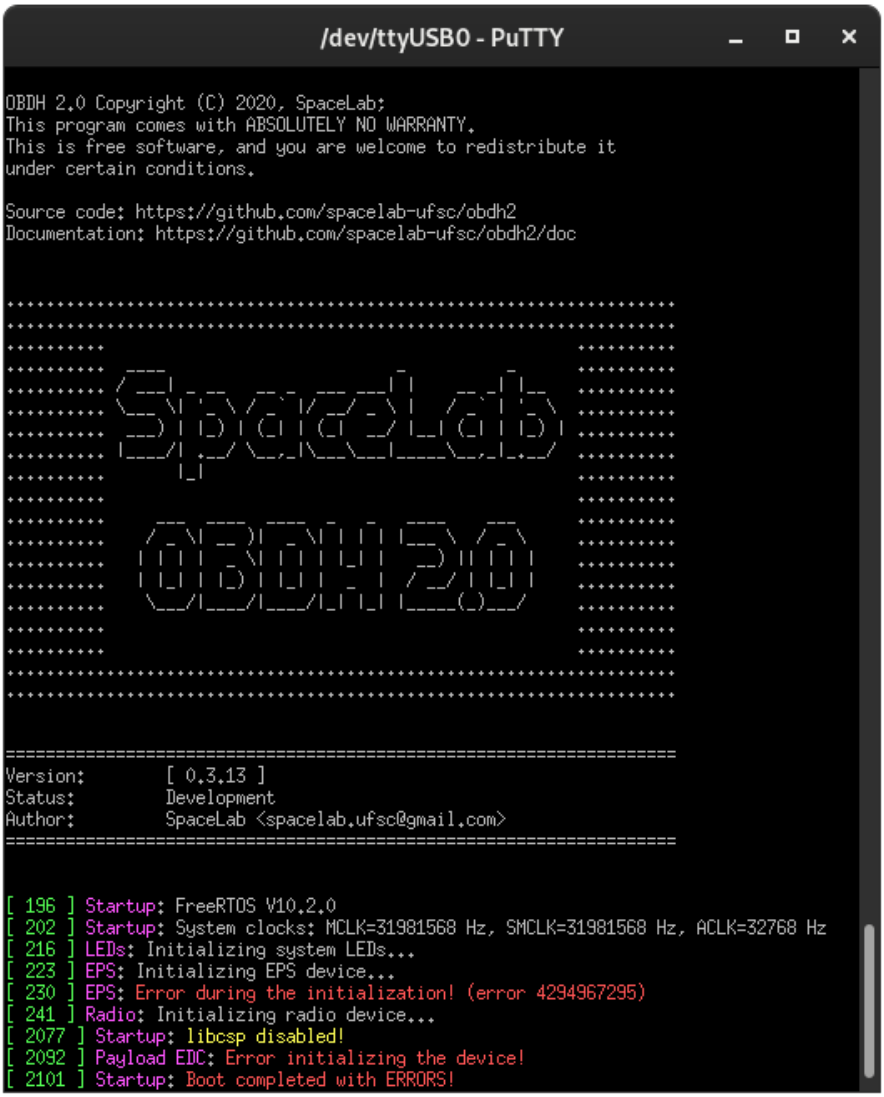


Figure 6.1: Firmware initialization on PuTTy.

Bibliography

[1] SpaceLab. FloripaSat-2 Documentation, 2021. Available at <https://github.com/spacelab-ufsc/floripasat2-doc>.

[2] SpaceLab. On-Board Data Handling, 2019. Available at <https://github.com/floripasat/obdh>.

[3] SpaceLab. On-Board Data Handling 2.0, 2020. Available at <https://github.com/spacelab-ufsc/obdh2>.

[4] Samtec. FSI-110-03-G-D-AD, 2020. Available at <https://www.samtec.com/products/fsi-110-03-g-d-ad>.

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

[6] Texas Instruments Inc. *TPS328x Voltage Monitor With Watchdog Timer*, July 2020.

[7] Amazon Web Services, Inc. FreeRTOS - Real-time operating system for microcontrollers, 2020. Available at <https://www.freertos.org/>.

[8] Texas Instruments. MSP Driver Library, 2020. Available at <https://www.ti.com/tool/MSPDRIVERLIB>.

[9] Jon Petter Skagmo. Ngham protocol, 2014. Available at <https://github.com/skagmo/ngham>.

[10] GomSpace. The CubeSat Space Protocol, 2020. Available at <https://github.com/GomSpace/libcsp>.

[11] SpaceLab. On-board data handling 2.0 draftsman document, 2020. Available at <https://github.com/spacelab-ufsc/obdh2/tree/master/hardware/outputs/board_draftsman>.

APPENDIX A

Test Report of v0.5 Version

This appendix is a test report of the first manufactured and assembled PCB (version v0.5).

- **PCB manufacturer**: PCBWay (China)
- **PCB assembly**: PCBWay (China)
- **PCB arrival date**: 2021/04/14
- **Execution date**: 2021/04/16 to TBC
- **Tester**: G. M. Marcelino

A.1 Visual Inspection

- **Test description/Objective**: Inspection of the board, visually and with a multimeter, searching for fabrication and assembly failures.
- **Material**:
  - Multimeter UNI-T DT830B
- **Results**: The results of this test can be seen in Figures A.1 (top view of the board) and A.2 (bottom view of the board).
- **Conclusion**: No problems were identified on this test.

A.2 Firmware Programming

- **Test description/Objective**: Inspection of the board, visually and with a multimeter, searching for fabrication and assembly mistakes.
- **Material**:
  - Code Composer Studio v9.3.0
  - MSP-FET Flash Emulation Tool
  - USB-UART converter

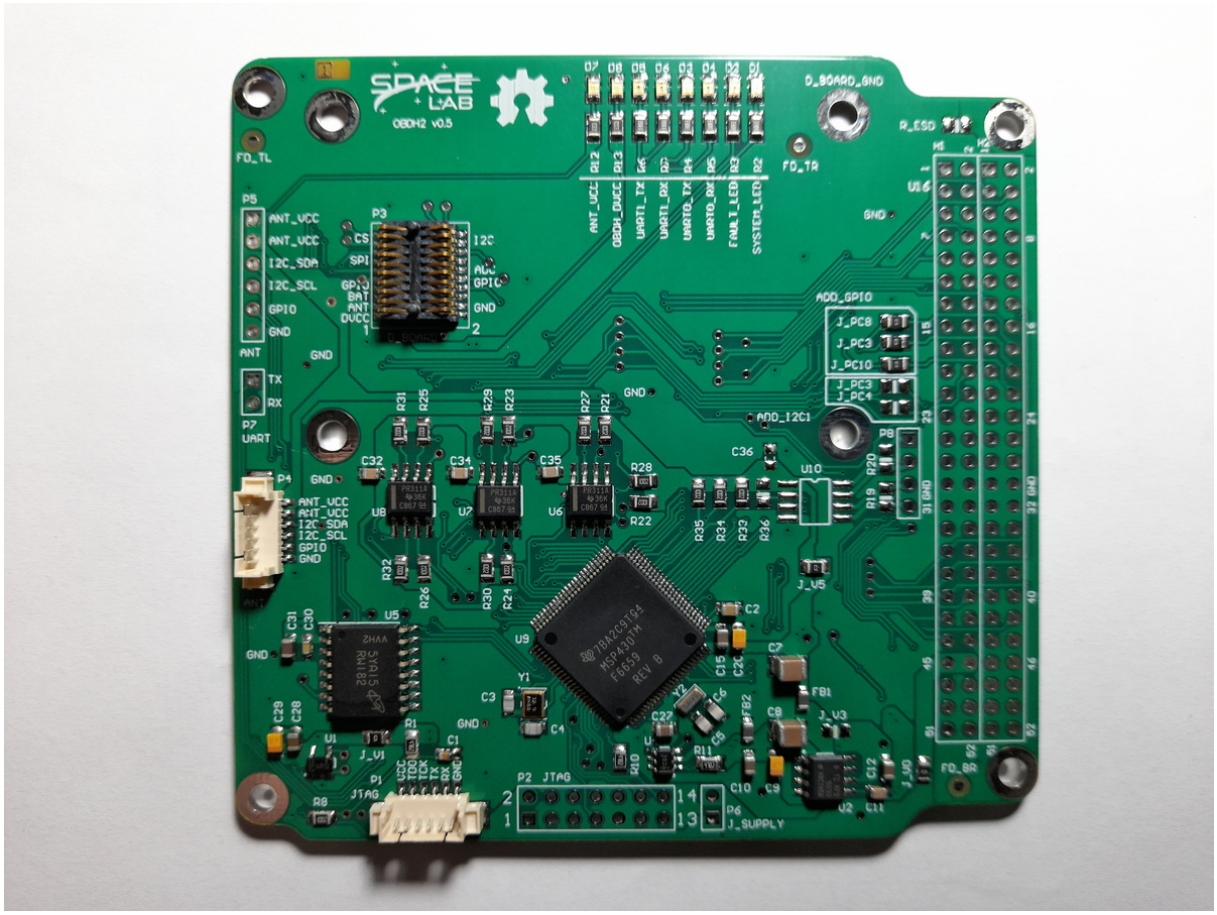


Figure A.1: Top view of the OBDH 2.0 v0.5 board.

  - Screen (Linux software)

- **Results**: The results of this are available in Figure A.3, where the log messages of the first boot of the board can be seen.
- **Conclusion**: No problems were identified on this test.

A.3 Communication Busses

- **Test description/Objective**: Test the communication busses of the board, as listed below:
  - I^2C Port 0
  - I^2C Port 1
  - I^2C Port 2
- **Material**:
  - Saleae Logic Analyzer (24 MHz, 8 channels)
  - Saleae Logic software (v1.2.18)

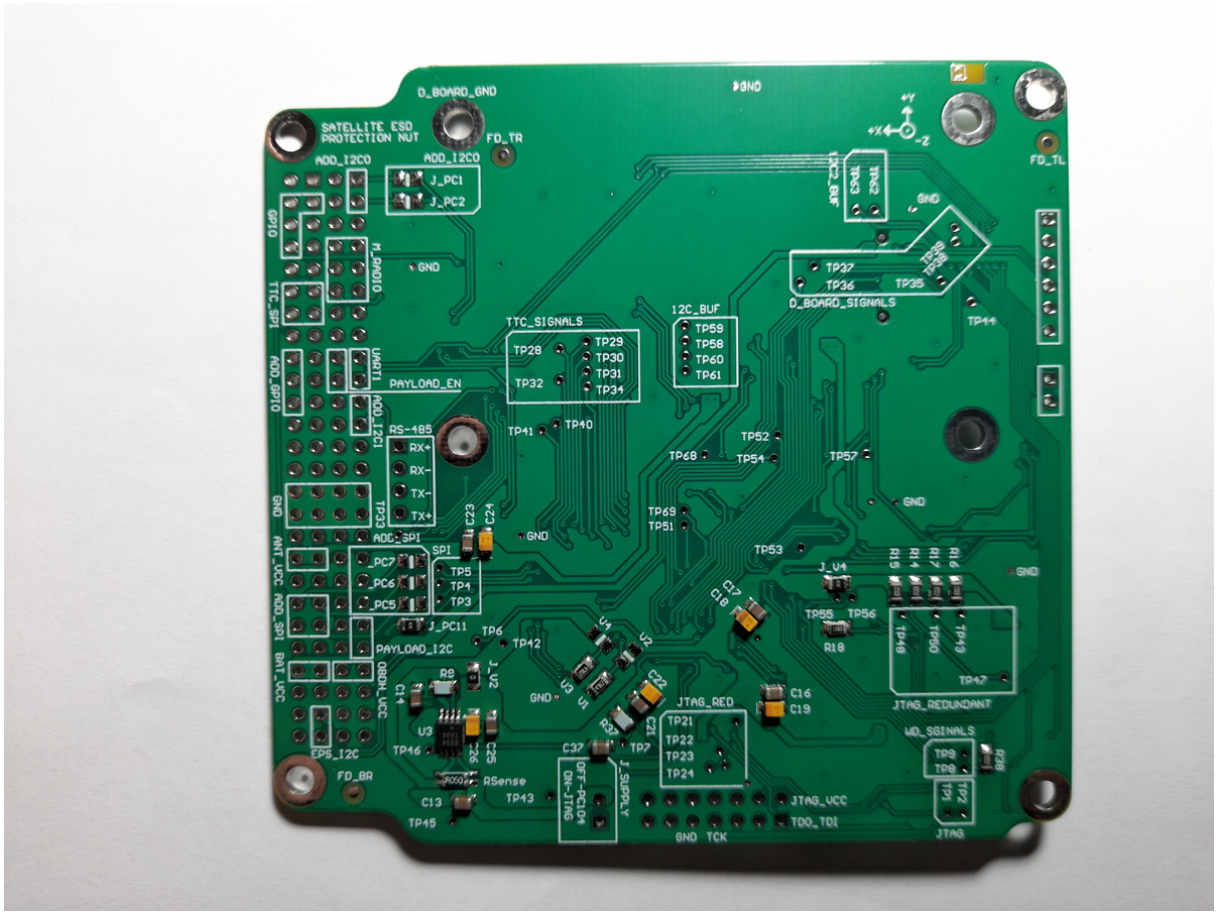


Figure A.2: Bottom view of the OBDH 2.0 v0.5 board.

  - MSP-FET Flash Emulation Tool

- **Results**: The results of this test can be seen in Figures A.4, A.5 and A.6.
- **Conclusion:** No problems were identified on this test, all buses are working as expected.

A.4 Sensors

A.4.1 Input Voltage

- **Test description/Objective**: .
- **Material**:
  - Code Composer Studio v9.3.0
  - MSP-FET Flash Emulation Tool
  - USB-UART converter
  - Screen (Linux software)

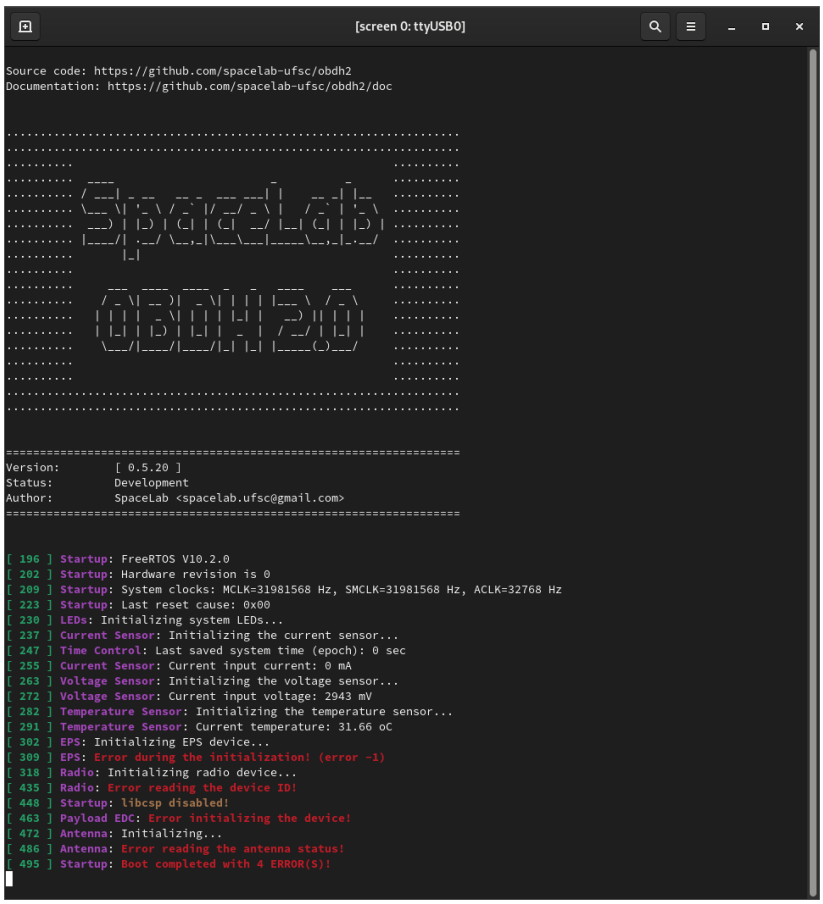
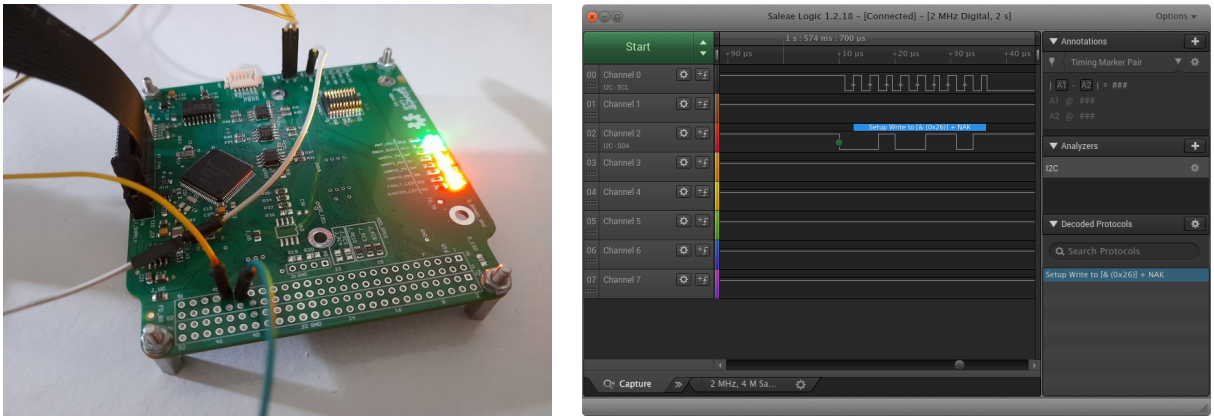


Figure A.3: Log messages during the first boot.

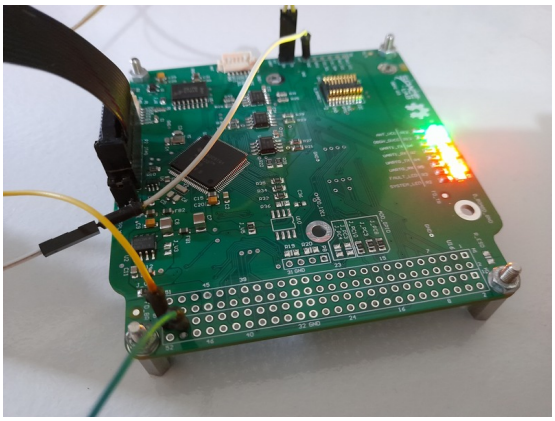


(a) Connections of the I^{2}C port 0 test.

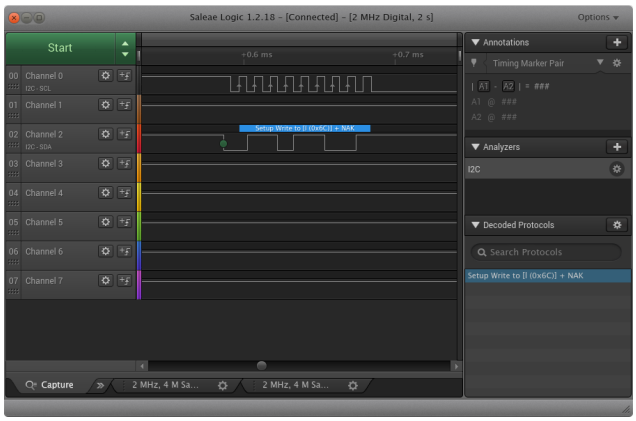
(b) Waveforms of the I^{2}C port 0 test.

Figure A.4: I^{2}C port 0 test.

- **Results**: .
- **Conclusion:** .

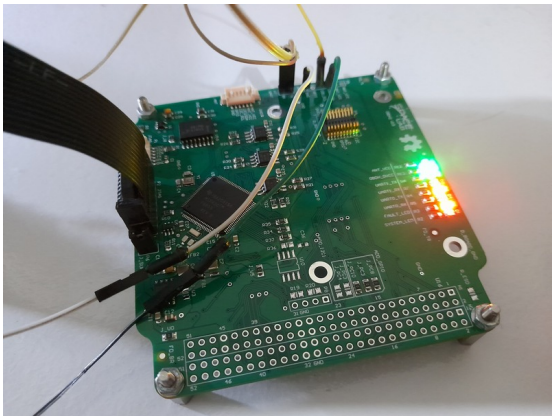


(a) Connections of the I^{2}C port 1 test.

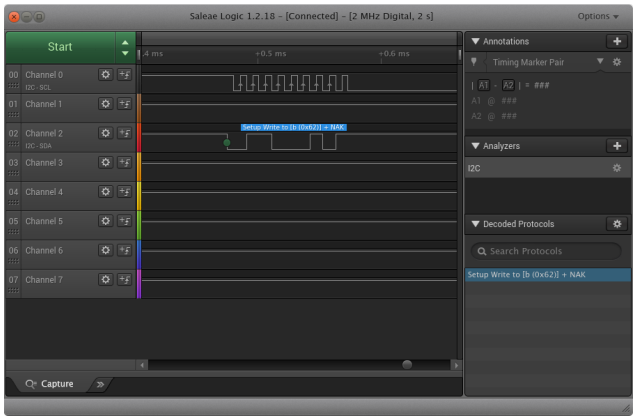


(b) Waveforms of the I^{2}C port 1 test.

Figure A.5: I^{2}C port 1 test.



(a) Connections of the I^{2}C port 2 test.

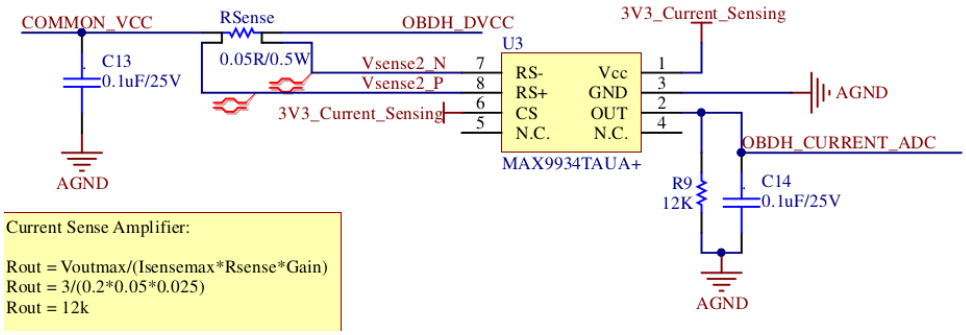


(b) Waveforms of the I^{2}C port 2 test.

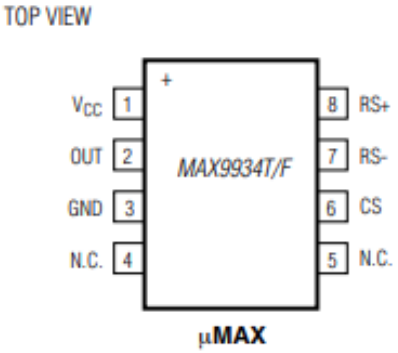
Figure A.6: I^{2}C port 2 test.

A.4.2 Input Current

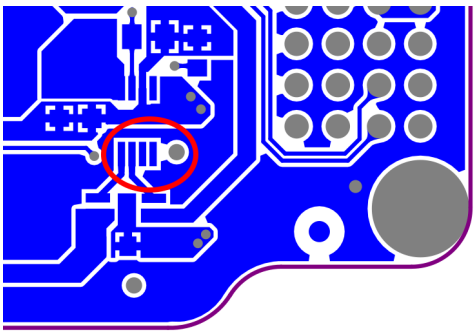
- **Test description/Objective**: .
- **Material**:
  - Code Composer Studio v9.3.0
  - MSP-FET Flash Emulation Tool
  - USB-UART converter
  - Screen (Linux software)
- **Results**: .
- **Conclusion:** .



(a) Current sensing circuit.



(b) MAX9934 pinout.



(c) Current sensing layout (bottom layer).

Figure A.7: .

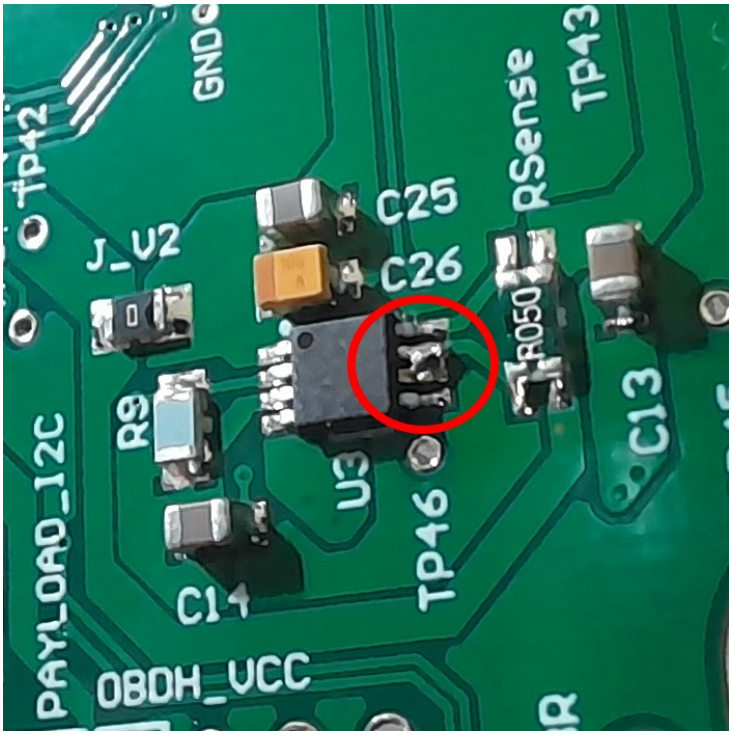


Figure A.8: Current sensor fix.

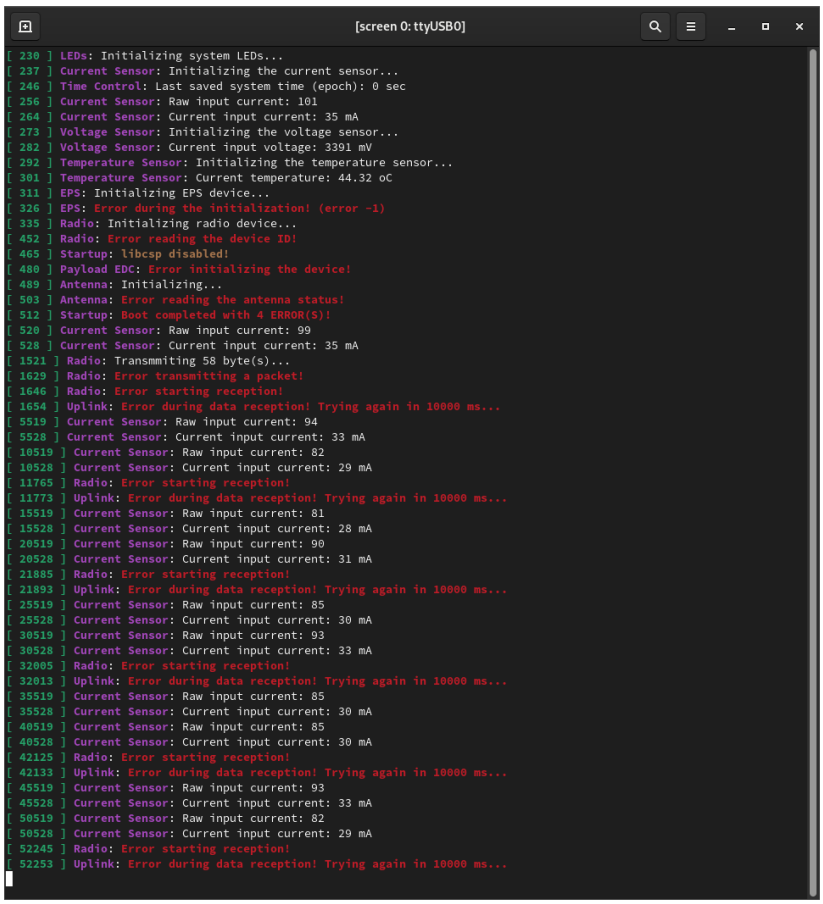
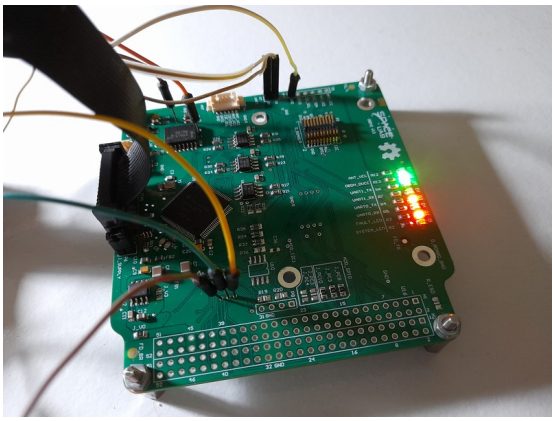


Figure A.9: Log messages with the read values from the current sensor.

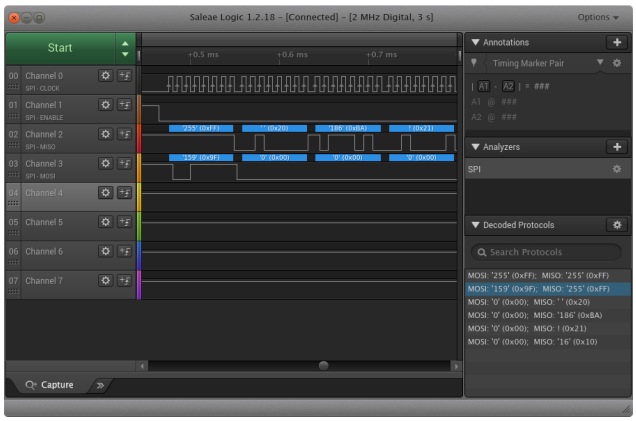
A.5 Peripherals

A.5.1 NOR Flash Memory

- **Test description/Objective**: Test the functionality of the NOR flash memory by verifying the device ID register of the IC.
- **Material**:
  - Saleae Logic Analyzer (24 MHz, 8 channels)
  - Saleae Logic software (v1.2.18)
  - MSP-FET Flash Emulation Tool
- **Results**: The results of this test can be seen in Figure A.10.
- **Conclusion:** No problems were identified on this test, as can be seen in A.10(b), the device ID register was read as expected.



(a) Connections of the NOR flash memory test.



(b) Waveforms of the NOR memory SPI.

Figure A.10: NOR memory SPI test.

A.6 Conclusion

Excluding the current sensor issue, no major problems were identified during the executed tests. For the next fabrication round, the identified mistakes will be corrected.