Designing a Low-Power Autonomous Embedded System for Aircraft Collision Prevention

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

Designing a Low-Power Autonomous Embedded System for Aircraft Collision Prevention

Bachelor's Thesis

Jonas Elmiger

joelmiger@ethz.ch

Distributed Computing Group
Computer Engineering and Networks Laboratory
ETH Zürich

Supervisors:
Peter Belcák
Prof. Dr. Roger Wattenhofer

August 1, 2022

Acknowledgements

I want to thank Peter Belcák for his continuous support and his availability for advice, questions and discussions. Peter and the DISCO group have provided the hardware and instruments needed to realize this project. I am grateful for the opportunity to do such an interesting project offered by Prof. Dr. Wattenhofer and his group. Furthermore, Patrick Oberlin has done excellent groundwork on the algorithms needed for the project and was quick to respond to questions about them, for which I am very thankful. Finally, I appreciate the support and help from my friends and family who always had a helpful remark or tip and sometimes gave me a much needed different perspective. As a final addendum I want to acknowledge my neighbor's cat which kept me motivated to go for a walk and refresh my mind.

Abstract

In aviation, safety has top priority. Large aircraft are equipped with on-board radar systems for local airspace surveillance which allows them to predict and mitigate collisions in-flight. In contrast, light and sporting aircraft (LSA) cannot carry such massive systems and therefore require a lightweight and compact alternative capable of predicting upcoming encounters. Such a device provides LSA pilots with an additional protection system which aides them to keep track of hazards, especially in crowded airspaces.

This thesis documents the development of a mobile embedded system capable of in-flight collision predictions and an aiding stationary device that broadcasts information about ground obstructions. Together they enable prognostic flight path tracking by monitoring the aircraft's position and trajectory and transmitting it to other aircraft. The mobile device runs previously developed algorithms [1] to predict probable future paths from which early warnings about collisions with aircraft or obstacles can be derived. Both the mobile and the stationary device are optimized towards low-power long-range communication and autonomous operation in the air such that they are able to perform their functionality precisely and reliably.

Contents

Chapter 1

Introduction and Terminology

1.1 Introduction

The goal of this work is to develop the hardware for a device capable of predicting flight trajectories and collisions and sending this data to other devices of its type. Its capabilities must include low-power long-range transmission which is needed for in-flight communication and it is required to find its coordinate position in airspace. Additionally it should operate autonomously and return live-feedback to the pilot, assisting their flight decisions and allowing them to react to possibly undetected hazards. This network of aircraft devices is aided by stationary obstruction broadcast devices, alarming pilots about the prominent on-ground obstacles. This stationary device was developed in conjunction with the mobile counterpart to verify their behavior amongst each other.

As a result of this project, two working prototypes were created. A stationary device, which is configurable over the integrated USB interface and communicates successfully with the mobile embedded system. The mobile device is capable of flight-trajectory predictions in airspace, allowing it to achieve collision prevention capabilities in combination with data from other devices. This work continues the development of the complete project that aims to create finished and user-accessible devices which solve the collision avoidance problem.

1.2 Terminology Conventions

In this thesis the following terms and notions will be used as defined here:

- Device: An electronic apparatus serving a special purpose.
- LSA: Light and Sporting Aircraft.
- PACAS: Practical Aircraft Collision Avoidance System

- (Mobile) Beacon: The mobile monitoring and communication device.
- Landmark: The stationary broadcasting device.
- GNSS: Global Navigation Satellite System; collective term for the various navigation systems such as GPS, GLONASS, Galileo and BeiDou.
- PIC: Microcontroller family from Microchip Technology Inc.; 'PIC' is used as an abbreviation for the chosen PIC18LF14K50.
- MSP430: Microcontroller family from Texas Instruments Incorporated; 'MSP' is used as an abbreviation for the utilized MSP430FR5994 in this paper.

Chapter 2

Communication

The following chapter explains the communication between the embedded devices and describes the most important aspects, limitations and challenges of it. Both the Landmark and the Mobile Beacon will be used in airspace where long distances between devices are the norm, which is why they must be capable of long-range communication. At the same time, they are desired to be as low-power as possible. Since wireless communication — especially transmission over several kilometers — is highly energy-intensive, the protocol and modulation have to be carefully chosen and their disadvantages considered in relation to the tasks of the devices. Additionally, with Mobile Beacons constantly moving, the communication protocol is required to be adaptive, further adding to the complexity of the system.

2.1 Wireless Signal Modulation

When comparing different modulation techniques, LoRa by Semtech [2] stands out as it was developed for low-power Internet of Things applications. It uses a chirp-spread-spectrum modulation which makes it exceptionally noise resistant. Therefore even low-power transmissions can be decoded over long distances. The regulations and requirements around the LoRa PHY are generally compatible with the expectations put on our embedded system. LoRa utilizes the 868MHz band in Europe and other ISM bands around the globe which are reserved for licence-free usage as long as transmission times remain below 1%. This should be feasible as each device has only small data packets which can be sporadically transmitted in reasonable intervals. On the other hand, as LoRa was developed for static sensor networks, a few downsides of it need to be considered

2.1.1 LoRa Transceiver Module

The RFM95W-868S2 module [3] by Seeed offers a certified transceiver which is capable of LoRa communication as well as FSK, GFSK, MSK, GMSK and OOK. It is based on a Semtech SX127x design and can therefore be effortlessly

exchanged with similar modules if necessary. For prototype development the FSK capability is an important feature as the data rate of LoRa is limited at 37.5kbps [3, p.17] including preamble and checksum. This is a severe restriction as the airspace around thermals and similar points of interest can become crowded and the aircraft might fill the channel capacity. Upon performing outdoor tests the channel use should be closely monitored and a change to other modulations evaluated.

2.2 Transmission Protocol

Although there is the supplementary network and communication standard LORAWAN, which was specifically developed for use with LoRa, it has several design features that render it unusable for our purposes. Its major hindrance is that it requires a predefined host (called gateway) which requests data from devices and streams it into the internet or a server network. This internet layer is unnecessary for the purposes of local communication between devices.

A simple protocol to broadcast the packets is non-persistent CSMA. When a device has a packet which is prepared for transmission it examines the carrier frequency for other traffic. Should another packet be in transmission the device backs off for a random time and reexamines the channel. Depending on the aggressiveness of the CSMA strategy, the packet is either sent immediately in the first available spot or sent with a certain probability. In the Landmark and Beacon prototypes 1-persistent CSMA is implemented and packets are transmitted upon detecting an idle channel. Naturally this is vulnerable to collisions happening within the first milliseconds of the channel turning idle. However, it provides stable behaviour to test the devices on their own and enabled verification of correct behaviour. There are additional downsides to this protocol, especially in combination with LoRa. Further work should test and optimize these issues in a multiple device network. One of the technical limitations manifests itself in LoRa's utilization of a form of chirp spread spectrum modulation which complicates channel activity detection. Energy-efficient traffic detection with the communication module can only detect packet preambles, otherwise regular RSSI has to be used. Otherwise CSMA is also unable to detect packet collisions and corrupted data is lost and not retransmitted.

Slotted O-persistent CSMA would allow for an improved throughput by using predefined device-assigned discrete timeslots. Each Mobile Beacon is equipped with a GNSS module which provides it with a high precision time signal featuring only nanosecond deviation between different devices. On the foundation of this common time base the Beacons have the fundamentals needed to transmit in predefined slots. At this stage, however, the Landmarks lack any sense of time and would rely on Beacon information for that. Furthermore the Mobile Beacons move in a constantly changing environment while in flight. This requires an algo-

rithm which defines slot allocations that all participants agree on and functions among the entire network since devices cannot see all other devices.

2.3 Packet Structure

The following section and table 2.1 describe the data structure of the packets broadcast by Landmark and Mobile Beacon devices. To distinguish between the two device types the message must contain a device identifier. Furthermore the devices must be distinguishable amongst each other which can be realized with a device specific ID. Both types inform about their current position with a Landmark sending a slightly lower altitude resolution to shorten the message length. A Landmark provides additional obstruction information such as the dimensions and the type of it. Beacon message contents change permanently with their movements and therefore an additional time stamp is added which allows devices to filter for the latest packets. Moreover, these messages provide data about the size and current speed of the aircraft combined with probable trajectories. Altogether these packets provide a full picture of the airspace situation to the surrounding network of devices.

Beacon Message	Landmark Message		
Device	0xF0	Device	0x0F
ID	4B	ID	4B
Latitude	4B	Latitude	4B
Longitude	4B	Longitude	4B
Altitude	4B	Altitude	2B
Time	4B	Obstacle	1B
Direction 1 X	4B	BOX Lat	1B
Direction 1 Y	4B	BOX Long	1B
Direction 1 Z	4B	BOX Alt	1B
Direction 2 X	4B		
Direction 2 Y	4B		
Direction 2 Z	4B		
Size	2B		
Speed	2B		
Trajectories	x∗4B		

Table 2.1: Packet Structure

Chapter 3

PACAS Landmark

This chapter covers the most important design aspects of the PACAS Landmark embedded system and the major constraints that influence these. The Landmark is developed to warn its surroundings from the ground about obstructions such as ropes or radio masts. For this, on-demand broadcasts of hard-coded positional information are sufficient and can be sent upon a Mobile Beacon coming within transmission range of the Landmark. It should be straightforward to configure this obstacle data previously to installing a Landmark device. Such a mounting location of a Landmark ought to be close to the obstacle and will presumably be difficult to access, which is why it needs to be as self-sufficient as possible and maintenance-free. This requires the Landmark to be extremely low-power and energy-efficient to survive from off-grid auxiliary power sources such as photovoltaic cells.

3.1 Design Aspects

3.1.1 Microcontroller Selection

To function as a stationary broadcasting device in remote and hard-to-reach locations the Landmark needs its own power storage and power source. To deplete these as slow as possible the Landmark should spend most of its time dormant and only wake up occasionally. The simplicity of its tasks and the large idle percentage favor a simple microcontroller with excellent low-current sleep features as the brain of the device. The choice fell on the PIC18LF14K50 [4] as it is a plain 8-bit microcontroller with outstanding low-power operating modes. Additionally it features integrated USB hardware which makes interfacing to host devices for configuration simple without requiring additional modules. Finally, even while the microcontroller is in operation, it consumes considerably less power than more capable processors.

3.1.2 Regular Operation

Although a low-power and efficient transmission-standard was selected, sending and receiving is still the most energy-consuming task of the device. According to the LoRa transceiver modules data sheet [3, Table 5] between 20mA and 120mA are used while transmitting and 10.8mA to 12.1mA during reception. These specifications give the reason to mainly listen for traffic and only broadcast when necessary. Transmissions by Landmarks without any Beacon in range being able to receive them do not provide any benefit for the distributed network. The approach taken here is illustrated in figure 3.1 and can be described as follows: The Landmark wakes up in regular intervals, listens for channel activity by Mobile Beacons and then only transmits if a Beacon was detected. The distinction between the two packet sources is especially important as a few Landmarks could keep each other transmitting in an infinite loop. This would be caused by Landmarks responding to packets from other Landmarks, sending their own packets, and in turn triggering new transmissions.

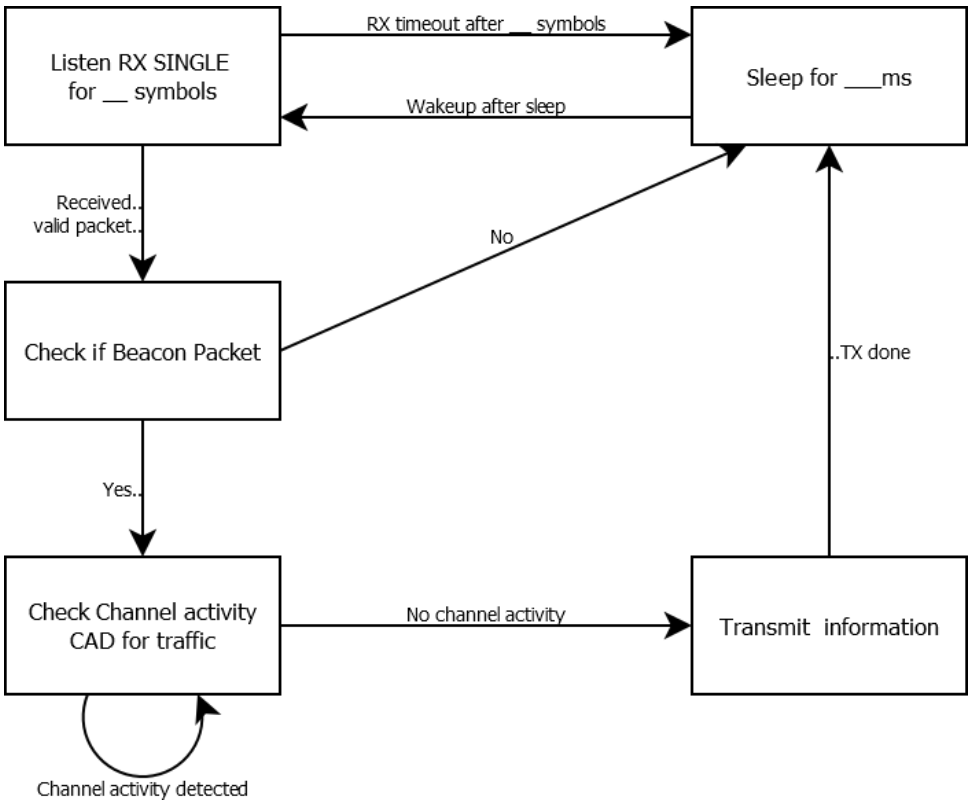


Figure 3.1: Landmark regular operation automaton.

Obstacle information is programmed with the USB application into the microcontroller-integrated EEPROM where it is stored long-term and which is resistant to power

loss. Broadcast data is read from this permanent memory to guarantee that up-to-date information is transmitted. Table 3.1 illustrates how the Landmark data is arranged in the PIC's EEPROM. From the first address in memory the stored bytes are structured according to the transmitted message as specified in Table 2.1. This simplifies the composition of the message to be sent since it can be assembled directly from the first bytes of the nonvolatile EEPROM.

Address	Bytes	Data
A_LM	1B	Constant **0x0F** designating it as a Landmark
A_ID	4B	ID number to differentiate between Landmarks
A_LAT	4B	Latitude of Landmark position
A_LON	4B	Longitude of Landmark position
A_ALT	2B	Altitude of Landmark position, signed short
A_TYP	1B	Obstacle type, possibly useful to consider
A_BOX_LAT	1B	Latitude extension of obstacle, from Pos to both sides
A_BOX_LON	1B	Longitude extension of obstacle, from Pos to both sides
A_BOX_ALT	1B	Height of obstacle, from position upwards
...	...	*... additional space if necessary*
A_USER	63B	User defined 63 character string, address fixed at 192
DWF ('>')	1B	Data written flag at 255, '>' if EEPROM was modified

Table 3.1: Landmark EEPROM storage (256 Bytes)

3.1.3 USB Interface

The integrated USB hardware of the PIC18LF14K50 simplifies interfacing with host devices to program the data in the EEPROM memory. Only needing a cable and the application, which is further explained in section 5.1, makes the user experience as comfortable as possible. For USB communication, the PIC needs an external clock providing it with 48MHz. This clock runs exclusively during USB operation and is therefore powered by a pin from the MCU which allows the PIC to disable the clock completely without any standby current consumed by the oscillator. Its current draw is far below the limit for the PIC's I/O pins, which makes a direct connection between the pin and clock possible.

An example library to handle USB communication from Microchip was used unmodified and their supporting application files were customized for Landmark functionality. Since the Landmark acts as a slave, it waits for requests from the host and answers them. The requests are structured as outlined in 3.2 with memory read requests from the host only containing the first character.

Command	Symbol	Content
Set LM values	>	63 data bytes
Get LM values	<	63 data bytes, answer by Landmark
Set user text	U	63 string characters
Get user text	u	63 string characters, answer by Landmark
Disconnect USB	0x0F	–

Table 3.2: Landmark USB packet structure (64 Bytes)

3.2 Programming

Coding and debugging was carried out in the IDE provided by Microchip; MPLAB X IDE v6.00 [5] and the configuration of some general settings was aided by the MPLAB Code Configurator [6]. The program was compiled with the XC8 Compiler v2.36 [7] using the 'Config_2' setting for memory optimization. Both a library for USB functionality and one for interfacing with the LoRa module were found in Microchip's Libraries for Applications [8]. These libraries were used as a starting point and expanded upon to implement the Landmark's features. In the LoRa library all LORAWAN functionality was removed and the few FSK aspects were disabled.

3.3 Resulting Landmark Device

The final Landmark prototype was developed into a standalone PCB device from which further improvements and adjustments are simple. In conjunction with the Beacon it demonstrated the desired behaviour and performs its tasks reliably. Its schematic can be found in Appendix A where the general structure and connections can be observed. To power the Landmark, a step-down power supply was chosen, which converts any voltage between 3.3V and 6V to the required 3.3V supply voltage. This allows further evaluation of different voltage sources and their efficiency in combination with the device. A USB type C port is mounted, over which the Landmark can be powered and configured. To change the installed code, the necessary ICSP pins are broken out to a 0.1" header. The LVP pin is routed to a selection switch as not all programmers control this pin and therefore the user can select to program the Landmark directly via the switch. Furthermore, the RFM95 communication module is removable for debugging and standalone power measurements. Its transmission pin is routed to an external SMA connector to mount a frequency-matched antenna. They should provide better energy-efficiency than PCB-integrated strip-antennas. Due to the sub-optimal path through the pin headers the module currently suffers from poor transmission power, which is why the final Landmark should mount the transceiver-module directly on the PCB.

Chapter 4

PACAS Mobile Beacon

This chapter outlines the important design considerations that support the reliable operation of the PACAS Mobile Beacon. The Beacon aims to help pilots to keep track of other aircraft and obstructions around their own aircraft and informs them about upcoming hazardous situations. To achieve that, it is equipped with a long-range transceiver, a GNSS positioning module, a high-resolution pressure sensor and an interface via which information is streamed to the pilot. Data about the aircraft and some adjustable parameters can be entered by the pilot over a USB interface to configure the Beacon to the preferred behaviour.

For autonomous operation in unpowered aircraft the Mobile Beacon has a built-in rechargeable battery. It still should remain portable and compact and thus the Beacon has to be energy-efficient so the battery can be kept at a reasonable size.

4.1 Communication

The Mobile Beacon not only receives packets from Landmark devices on the ground, but also communicates with other Beacons around it to get the full picture of the airspace. As described in section 2.3, the regular transmissions from a Beacon contain its current position, its current speed and multiple possible trajectories on which it might fly. This data allows a Beacon to compare future positions of other devices against its own path and detect possible collisions. Feedback about obstructions, dangers and recommended mitigation can be returned to the pilot via either the fitted Bluetooth Low-Energy interface or other additional modules.

4.2 Design aspects

4.2.1 Regular Operation

For reliability of the network, the data from which the Beacon computes its own path has to be up to date. Otherwise other Beacons use its outdated and imprecise positional data in their collision calculations, leading to significant errors which eliminate any benefit for the pilot and can create hazardous situations. Thus the first priority of a Beacon must be reading its current position and then extrapolating the trajectories from it. Reception of data from other devices is of similar importance and can be done continuously by interrupting the main program flow upon receiving a packet.

GNSS positioning

In the Beacon centimeter-precision is not necessary; instead, reliability and energy efficiency matter. The NEO-M8 family by u-blox provides single-meter precision, supports GPS, GLONASS, BeiDou and Galileo for reliable positional data and offers great configurability. Furthermore it features a low idle current draw and supports fast restarts. Like most GNSS modules, the NEO-M8 returns altitude measurements with significant deviations; this is compensated with a high-resolution barometric pressure sensor.

Upon starting the Beacon, the GNSS module is reset to get it into a known state from where it is configured to send the current position once every full second. The first altitude value is fixed to the current atmospheric pressure. From this starting point the altitude can be calculated out of the pressure difference to the base value. In the main program operation the position message is used as an interrupt source in the microcontroller which upon arrival triggers the computations to start anew. The module provides the positional information in two standard NMEA-messages [9] containing the coordinates, altitude, speed and heading of the Beacon additionally to the current time. As every microcontroller reacts to this information immediately when it receives it, all Beacons are synchronized amongst each other at this moment. After the information is extracted from the messages, the trajectory and collision algorithms run.

Trajectory and Collision Computations

The computation of likely trajectories of the Beacon is done according to the algorithms described in [1, Algorithm 1]. As the MSP430FR5994 does not feature a floating point unit which could accelerate float variable calculations, most values were changed to high precision 32-bit integers. This means that the radian values used to describe the Beacon's position are scaled by a factor of 10^8 to use

the full range of the four-word integers and achieve 10 nanoradians of precision. This is taken into account in the calculations and reduces trajectory computation times from up to a second when using floats to around 40 milliseconds. For the linear approximation of a possible trajectory the full singular value decomposition was omitted by only calculating the first eigenvalue with its corresponding eigenvector according to the pseudo-code in [10]; which accelerates computation time significantly.

Distance calculations were done according to the Haversine formula described in [11] as recommended by Patrick Oberlin's Thesis. They are optimized for swift execution times as many distances are calculated per iteration; especially in collision analysis. The main optimization was made from the following observation:

Observation 4.1. *The largest possible distances that must be computed are over the range of transmissions. Due to power and regulatory limitations, this transmission distance stays well below 10 kilometers; which is equivalent to a latitudinal or longitudinal difference of at most* $10km = 0.0098512rad = \Delta_{MAX}$

Which justifies the assumption:

Assumption 4.2. Radian latitude and longitude differences $\Delta \leq \Delta_{MAX}$ allow for the approximations $sin(\Delta) = \Delta$ and $cos(\Delta) = 1$ as $\Delta_{MAX} \ll 1$

Therefore all sines of a latitude or longitude difference in the Haversine formula can be substituted as assumed above; accelerating computation noticeably.

After finishing calculations the microcontroller will attempt to transmit the finished packet using the previously described 1-persistent CSMA. Afterwards collision estimations are calculated and pilot feedback prepared.

Data Reception

Parallel to the computations a Beacon is always listening for incoming messages. Any correctly decoded packet received by the RFM95 module [3] is signalled by an interrupt. When such an interrupt is triggered the microcontroller briefly pauses the calculations, reads the received packet, rearms the LoRa module to listen for new packets and resumes the trajectory computation.

Visual Interface

If a possible collision was detected it could either be visually indicated on the Beacon itself or broadcast to the external monitor with collision-avoidance recommendations. Generally, information to an additional display could contain the current trajectory, relative positions of other aircraft and Landmarks and possibly thermals. For that, the Beacon is equipped with a Bluetooth Low-Energy

transmission module. However, the current module is limited to transmission; should communication with external sensors or other data-reception be desired it can be upgraded to a BLE-transceiver.

4.3 USB Communication

Contrary to the PIC in the Landmark; the MSP430FR5994 does not contain integrated USB hardware. Therefore an external USB HID interfacing module is required for configuration consistency across the two device types. The global restriction in integrated circuit variety and their availability made it impossible to obtain a dedicated USB HID conversion controller. Instead it was substituted with the PIC18LF14K50 as only few modifications were necessary to transform it into a USB HID to UART controller. Cost-wise it is nearly on par with specialized controllers, however it requires additional effort and has to be programmed. Therefore, a dedicated USB HID controller should preferably be used in future versions of the Beacon.

USB operation terminates the main algorithm running in the Beacon and focuses all resources on data exchange with the host application. For the user-configurable data there is a reserved memory section where it is stored permanently. The MSP features FRAM memory which is a nonvolatile low-power data storage technology. It allows the USB application to write the desired configuration parameters to it and have them permanently stored there.

4.4 Programming

The Beacon's MSP microcontroller was programmed in the Texas Instruments Code Composer Studio 11.2.0 [12]. The program was compiled with the integrated compiler. The same LoRa module library from the Landmark code was utilized which resolve the Microchip copyright disclaimers in the Texas instruments environment. Additionally the MIKROE BLE TX library [13] was used as a foundation for the Bluetooth communication code. All MIKROE Click functionality was removed and simple UART interfacing implemented.

4.5 Resulting Mobile Beacon Device

The final breadboard-prototype of the Mobile Beacon demonstrates its capability to achieve the desired functionality. It reliably configures its modules and sensors, starts regular operation and performs the desired calculations. The results of the computations were checked as well as possible but require further verification. The comprehensive wiring and connections are shown in the schematics in Appendix B where a general interface overview is provided as well.

Chapter 5

USB Applications

The simple, standardized and widespread USB interface allows for predictable communication results over a wide range of operating systems and devices without the need for additional or specialized hardware. The Landmark and the Mobile Beacon can be efficiently configured via their applications to program their behavior over USB. This requires nothing more from the user than a computer and a fitting USB cable. Since the Landmark and the Beacon are based on the HID device class, no further drivers are needed on the host side and their applications allow for direct configuration.

The original code for the USB applications was taken from the Microchip MLA library [8] 'Cross-Platform Custom HID Demo'. Additionally to being brought up to date the code was heavily modified and adapted to suit the applications. It was created with Qt Creator 7.0.0 (Community) and the resulting applications were only tested on Windows 10 Home (32bit/64bit) and Windows 11 Home(32bit/64bit). The individual functions utilize the unmodified HIDAPI [14] library to transmit and receive USB packets. Note that the main code is unable to use the API until the HIDAPI project is compiled.

5.1 Landmark Configuration

Landmark devices transmit about a static obstruction and therefore get configured with the position and dimensions of the obstacle they inform about. The Landmark configuration application provides a simple interface to enter the positional data and store it on the Landmark.

Upon plugging it into the host, the Landmark identifies and attaches itself; which is typically confirmed by the host with a pop-up message or an audible indication. Then the connect button in the application can be pressed which initiates the software to search for the Landmark among the attached devices. If it has successfully established a connection to a Landmark it reads its data and enables the input fields. Then the user is free to enter the desired data into the corresponding fields. Upon pressing the 'Set Values' button all entered

values are sent to the Landmark and stored there. The 'Get Values' button is useful to erase incorrectly entered values and resetting the menu to the stored data but otherwise not needed. After the Landmark is correctly configured and the application displays the desired values it is recommended to disconnect the device with the corresponding 'Disconnect' button. Any values which have not yet been successfully transmitted to the Landmark, are stored on it, before the device detaches itself from the host.

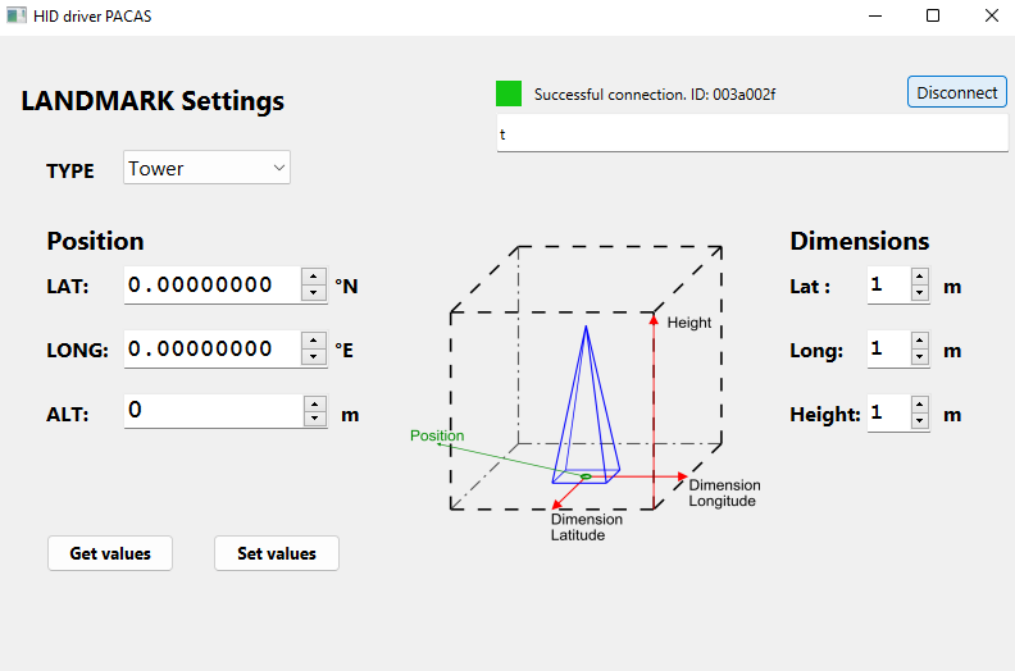


Figure 5.1: Landmark Configuration Application.

5.2 Beacon Configuration

The Mobile Beacon is carried aboard different kinds of LSAs and requires configuration to the specific parameters of the aircraft. Values such as size, gliding ratio and warning radius should be easily adjustable by a pilot previously to taking off. For that there is a USB interface installed similar to the Landmark and an adjusted application provided. This might need improvement and the hardware should be changed to a dedicated interfacing chip. Also it might be considered that a user could configure the Beacon via the external pilot interface over the BLE-application.

Chapter 6

Conclusion and Future Work

Turning the prototypes into a completely operational and reliable network of devices still requires some improvements and testing. Additional work could focus on:

- Power efficiency, power source and storage. Especially the Landmark, which will be situated in remote and hard to reach locations, is required to be energy self-sufficient. As it will remain outside it must be capable of surviving all seasons and their temperatures. Its mode of operation should be tuned for minimal energy consumption and a suitable power storage solution could be added. Furthermore the Beacon has to be equipped with a suitable battery which can provide the energy needed for operation during a full flight.
- Beacon feedback interface. The pilot needs a system which can display the data which the Beacon calculated and received. A suitable combination between audible and visual indicators could be evaluated for maximal usability comfort. The module could also advise about directions where the collision or hazardous situation will be mitigated. With the already installed Bluetooth module, connection to Smartphones – which provide a screen and an audio interface – is made simple.
- Transmission. As mentioned previously, LoRa provides a noise resistant low-power long-range modulation. However, the channel throughput is severely limited and its maximum capacity is reached at around 20 devices. It should be evaluated if crowded airspaces cause problems and if the protocol should be adjusted for reliability or the modulation changed for higher data rates.

- Testing and logging. To test the regular operation of the distributed network, some outdoor tests with in-flight operation can be performed. Logging the flight path and storing it on an SD card might not only help with testing but also remain a useful feature for pilots. It could be stored in the common .igc format which is widely used for flight path data. On start-up of the Beacon the current date is read from the GNSS module, such that the Beacon has all the needed information to create an igc-file.

Bibliography

[1] P. Oberlin, P. Belcák, and R. Wattenhofer, "Collision detection algorithm for a practical airborne collision avoidance system," in *DISCO, ETH Zürich*, Mar. 2022.

[2] Semtech.com. (2022, Jul.) What are LoRa and LoRaWAN? LoRa Alliance. [Online]. Available: https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-and-lorawan

[3] *Low Power Long Range Transceiver Module*, HOPERF, 2019, RFM95(W)-868S2. [Online]. Available: https://www.hoperf.com/data/upload/portal/20190801/RFM95W-V2.0.pdf

[4] *20-Pin USB Flash Microcontrollers with XLP Technology*, Microchip Technology Inc., 2015, PIC18LF14K50. [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/40001350F.pdf

[5] (2022, Jan.) MPLAB X IDE v6.00. [Online]. Available: https://www.microchip.com/en-us/tools-resources/develop/mplab-x-ide

[6] (2022, Mar.) MPLAB Code Configurator (MCC) v5.1.2. [Online]. Available: https://www.microchip.com/en-us/tools-resources/configure/mplab-code-configurator

[7] (2022, Feb.) MPLAB XC8 Compiler v2.36. [Online]. Available: https://www.microchip.com/en-us/tools-resources/develop/mplab-xc-compilers#tabs

[8] (2018, Nov.) Microchip Libraries for Applications (MLA) v2018-11-26. [Online]. Available: https://www.microchip.com/en-us/tools-resources/develop/libraries/microchip-libraries-for-applications

[9] NMEA. (2022, Jul.) What are LoRa and LoRaWAN? NMEA organization. [Online]. Available: https://www.nmea.org/

[10] wikipedia.org. (2022, Jul.) Principal component analysis: Iterative computation. Wikimedia Foundation, Inc. [Online]. Available: https://en.wikipedia.org/wiki/Principal_component_analysis#Iterative_computation

[11] C. Veness. (2022, Jul.) Calculate distance, bearing and more between latitude/longitude points. Movable Type Scripts. [Online]. Available: https://www.movable-type.co.uk/scripts/latlong.html

[12] (2022, Apr.) Code Composer Studio 11.2.0. [Online]. Available: https://www.ti.com/tool/CCSTUDIO#downloads

[13] github.com: MikroElektronika. (2022, Jul.) BLE TX click library. MikroElektronika. [Online]. Available: https://github.com/MikroElektronika/mikrosdk_click_v2/tree/master/clicks/bletx

[14] github.com: libusb/hidapi. (2022, Jul.) HIDAPI library for Windows, Linux, FreeBSD and macOS. [Online]. Available: https://github.com/libusb/hidapi

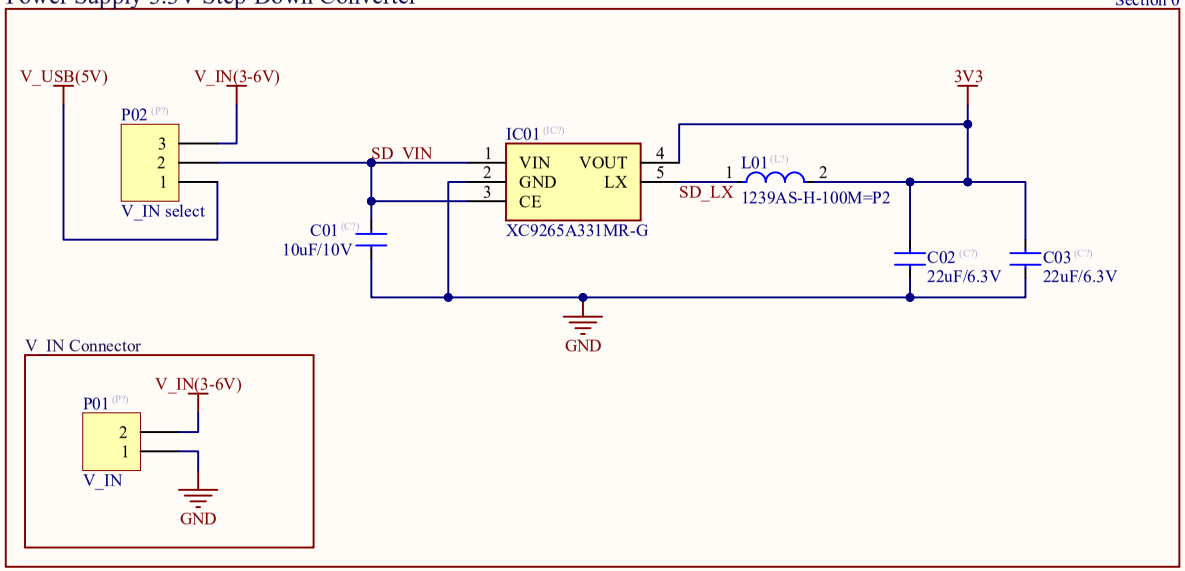
Appendix A

Landmark Schematic

LANDMARK schematic

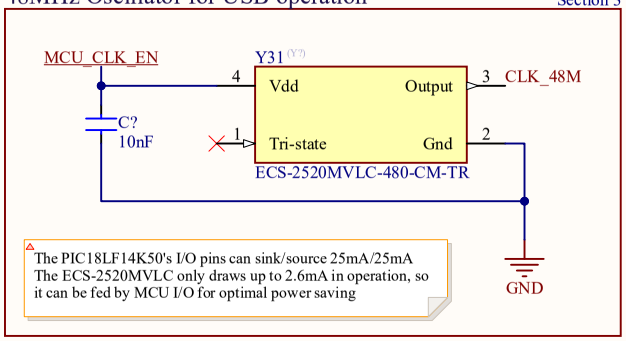
Power Supply 3.3V Step-Down Converter

Section 0



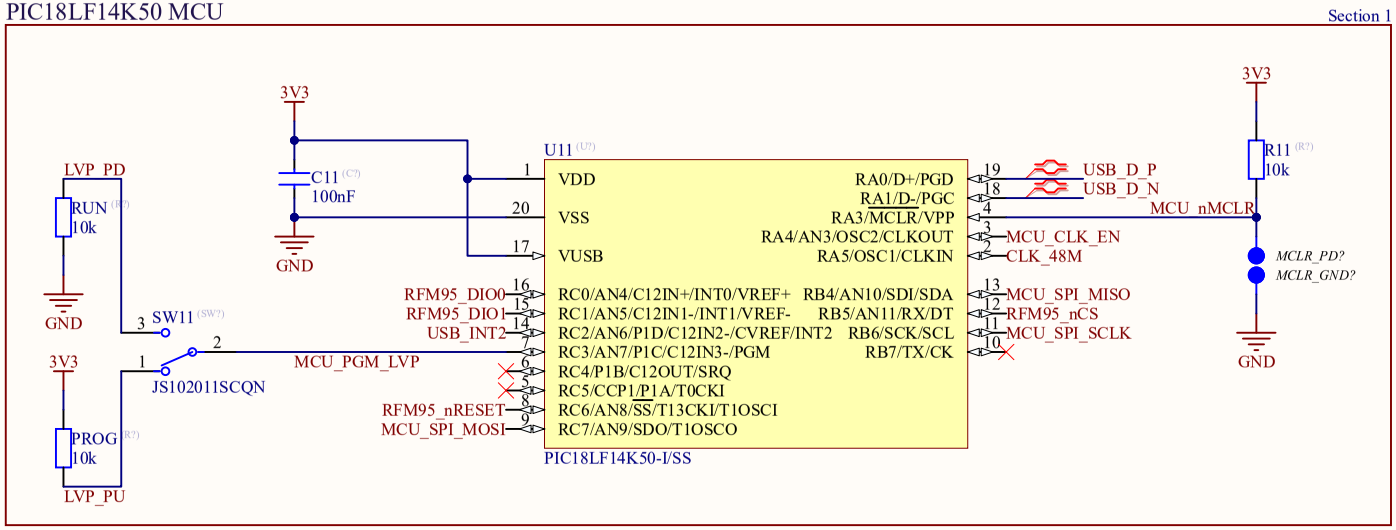
48MHz Oscillator for USB operation

Section 3



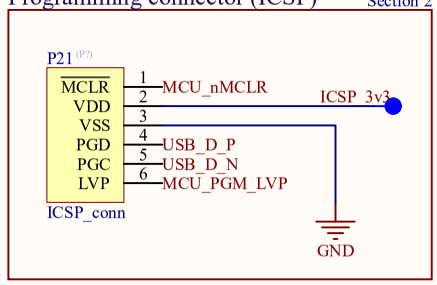
PIC18LF14K50 MCU

Section 1



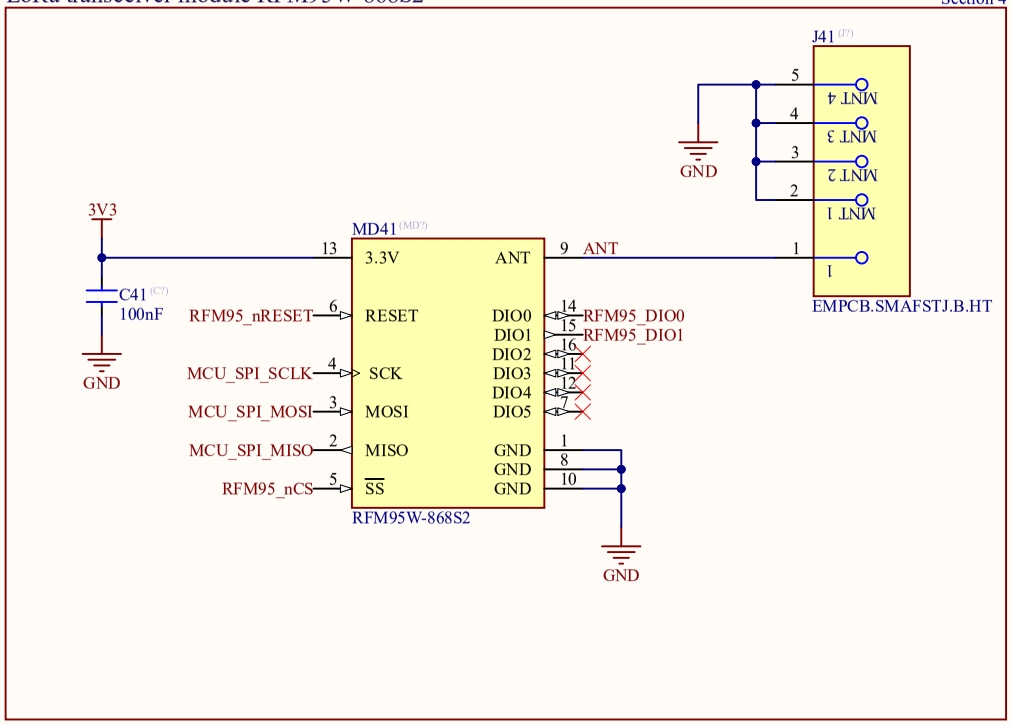
Programming connector (ICSP)

Section 2



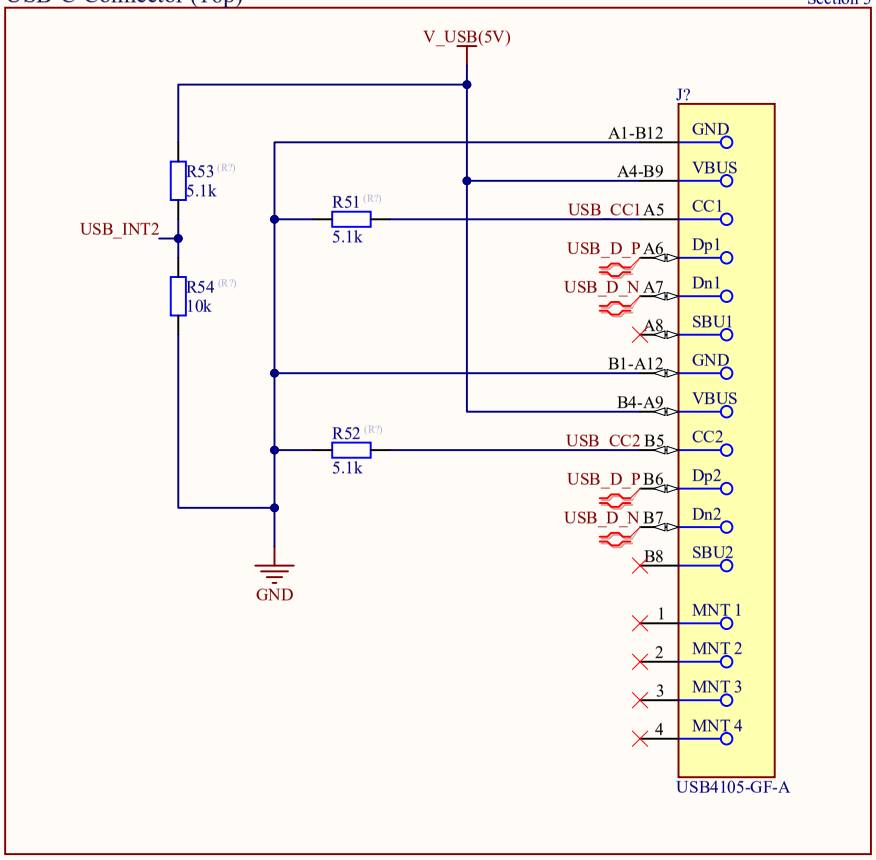
LoRa transceiver module RFM95W-868S2

Section 4



USB-C Connector (16p)

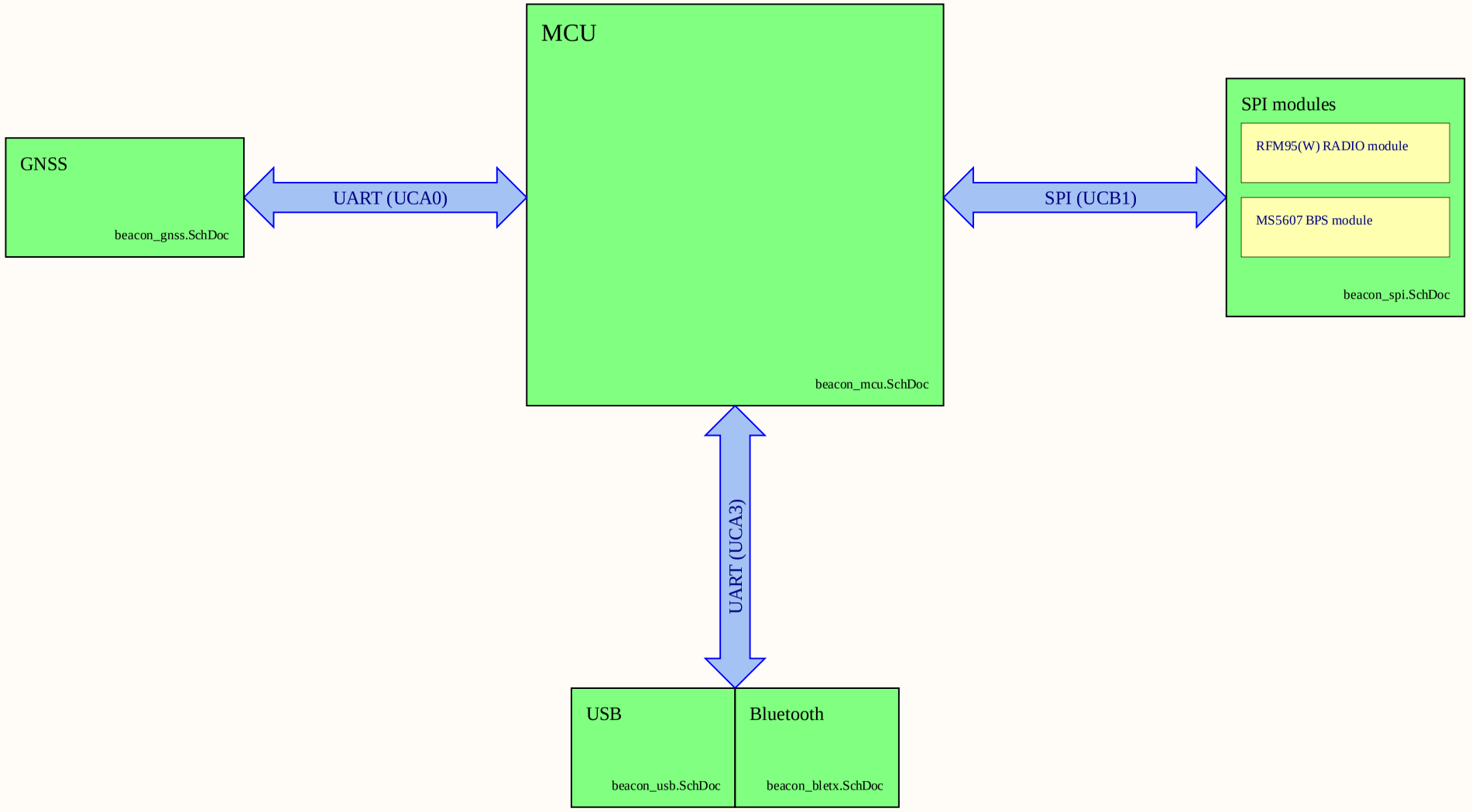
Section 5



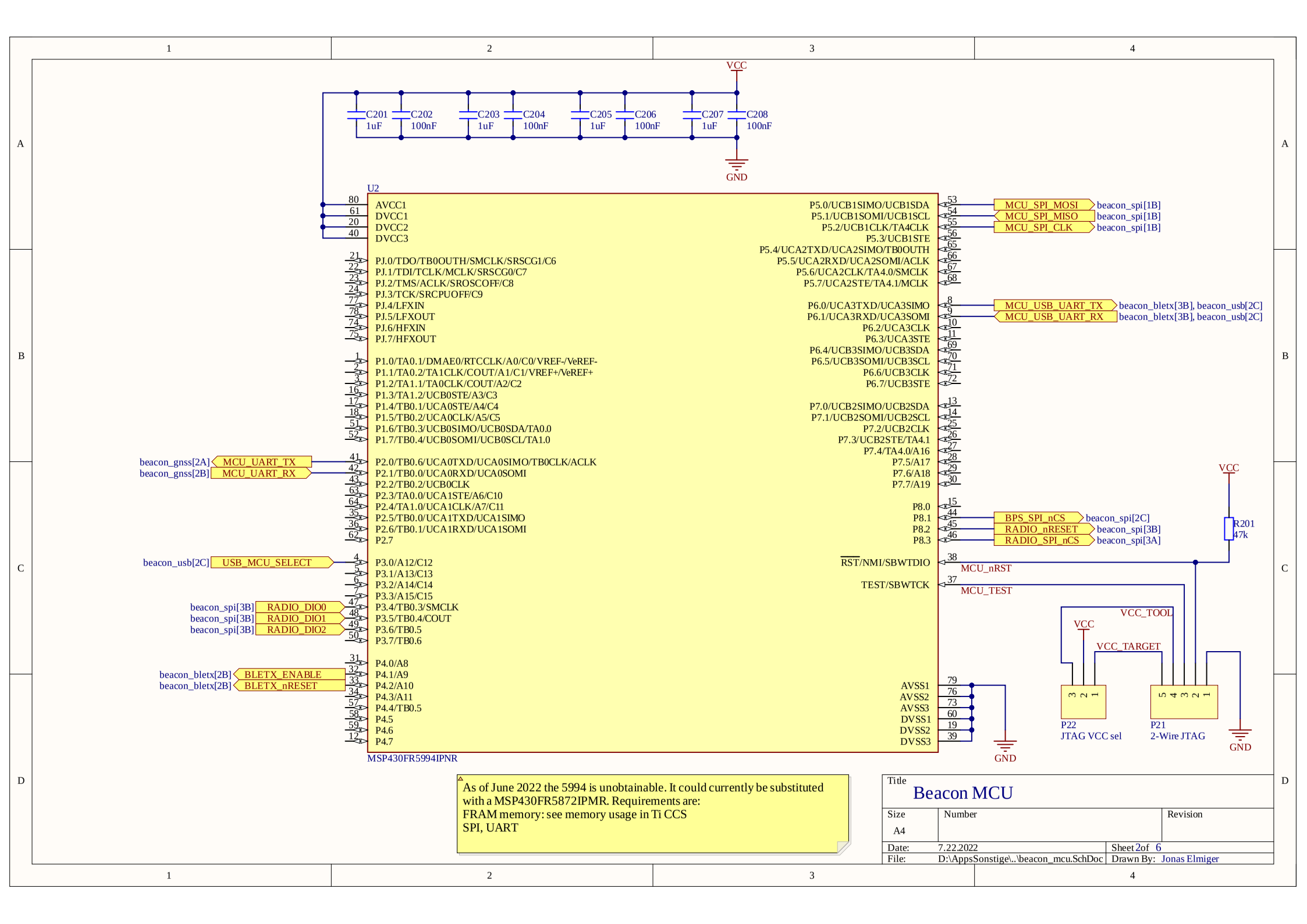
Title: LANDMARK schematic		
Size: A4	Number:	Revision:
Date: 7.11.2022	Sheet 2of 2	
File: D:\jonas\..\landmark_connectivity.SchDoc	Drawn By: Jonas Elmiger	

Appendix B

Mobile Beacon Schematic



Title: PACAS Beacon Overview		
Size: A4	Number: 1	Revision:
Date: 7.22.2022	Sheet 1 of 6	
File: D:\AppsSonstige\..\beacon_overview.SchDoc	Drawn By:	


VCC
C201 1uF
C202 100nF
C203 1uF
C204 100nF
C205 1uF
C206 100nF
C207 1uF
C208 100nF
GND
U2
AVCC1
DVCC1
DVCC2
DVCC3
PJ.0/TDO/TB0OUTH/SMCLK/SRSCG1/C6
PJ.1/TDI/TCLK/MCLK/SRSCG0/C7
PJ.2/TMS/ACLK/SROSCOFF/C8
PJ.3/TCK/SRCPUOFF/C9
PJ.4/LFXIN
PJ.5/LFXOUT
PJ.6/HFXIN
PJ.7/HFXOUT
P1.0/TA0.1/DMAE0/RTCCLK/A0/C0/VREF-/VeREF-
P1.1/TA0.2/TA1CLK/COUT/A1/C1/VREF+/VeREF+
P1.2/TA1.1/TA0CLK/COUT/A2/C2
P1.3/TA1.2/UCB0STE/A3/C3
P1.4/TB0.1/UCA0STE/A4/C4
P1.5/TB0.2/UCA0CLK/A5/C5
P1.6/TB0.3/UCB0SIMO/UCB0SDA/TA0.0
P1.7/TB0.4/UCB0SOMI/UCB0SCL/TA1.0
P2.0/TB0.6/UCA0TXD/UCA0SIMO/TB0CLK/ACLK
P2.1/TB0.0/UCA0RXD/UCA0SOMI
P2.2/TB0.2/UCB0CLK
P2.3/TA0.0/UCA1STE/A6/C10
P2.4/TA1.0/UCA1CLK/A7/C11
P2.5/TB0.0/UCA1TXD/UCA1SIMO
P2.6/TB0.1/UCA1RXD/UCA1SOMI
P2.7
P3.0/A12/C12
P3.1/A13/C13
P3.2/A14/C14
P3.3/A15/C15
P3.4/TB0.3/SMCLK
P3.5/TB0.4/COUT
P3.6/TB0.5
P3.7/TB0.6
P4.0/A8
P4.1/A9
P4.2/A10
P4.3/A11
P4.4/TB0.5
P4.5
P4.6
P4.7
P5.0/UCB1SIMO/UCB1SDA
P5.1/UCB1SOMI/UCB1SCL
P5.2/UCB1CLK/TA4CLK
P5.3/UCB1STE
P5.4/UCA2TXD/UCA2SIMO/TB0OUTH
P5.5/UCA2RXD/UCA2SOMI/ACLK
P5.6/UCA2CLK/TA4.0/SMCLK
P5.7/UCA2STE/TA4.1/MCLK
P6.0/UCA3TXD/UCA3SIMO
P6.1/UCA3RXD/UCA3SOMI
P6.2/UCA3CLK
P6.3/UCA3STE
P6.4/UCB3SIMO/UCB3SDA
P6.5/UCB3SOMI/UCB3SCL
P6.6/UCB3CLK
P6.7/UCB3STE
P7.0/UCB2SIMO/UCB2SDA
P7.1/UCB2SOMI/UCB2SCL
P7.2/UCB2CLK
P7.3/UCB2STE/TA4.1
P7.4/TA4.0/A16
P7.5/A17
P7.6/A18
P7.7/A19
P8.0
P8.1
P8.2
P8.3
RST/NMI/SBWTDIO
TEST/SBWTCK
AVSS1
AVSS2
AVSS3
DVSS1
DVSS2
DVSS3
MSP430FR5994IPNR
beacon_gnss[2A] MCU_UART_TX
beacon_gnss[2B] MCU_UART_RX
beacon_usb[2C] USB_MCU_SELECT
beacon_spi[3B] RADIO_DIO0
beacon_spi[3B] RADIO_DIO1
beacon_spi[3B] RADIO_DIO2
beacon_bletx[2B] BLETX_ENABLE
beacon_bletx[2B] BLETX_nRESET
MCU_SPI_MOSI beacon_spi[1B]
MCU_SPI_MISO beacon_spi[1B]
MCU_SPI_CLK beacon_spi[1B]
MCU_USB_UART_TX beacon_bletx[3B], beacon_usb[2C]
MCU_USB_UART_RX beacon_bletx[3B], beacon_usb[2C]
BPS_SPI_nCS beacon_spi[2C]
RADIO_nRESET beacon_spi[3B]
RADIO_SPI_nCS beacon_spi[3A]
VCC
R201 47k
MCU_nRST
MCU_TEST
VCC_TOOL
VCC
VCC_TARGET
P22 JTAG VCC sel
P21 2-Wire JTAG
GND
GND
As of June 2022 the 5994 is unobtainable. It could currently be substituted with a MSP430FR5872IPMR. Requirements are:
FRAM memory: see memory usage in Ti CCS
SPI, UART
Title
Beacon MCU
Size A4
Number
Revision
Date: 7.22.2022
Sheet 2 of 6
File: D:\AppsSonstige\..\beacon_mcu.SchDoc
Drawn By: Jonas Elmiger

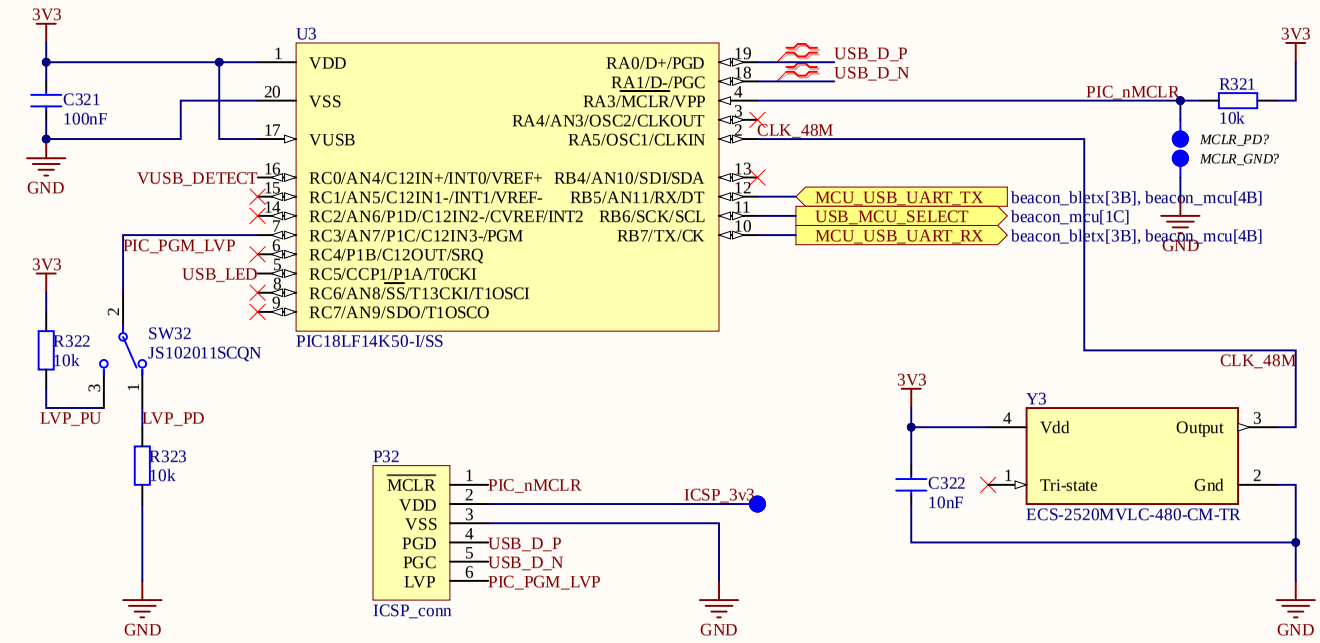
Power supply for USB communication

Section 1

Possibly not needed if a 5V tolerant MCU is used and the oscillator is powered from an I/O pin

PIC18LF14K50 MCU

Section 2

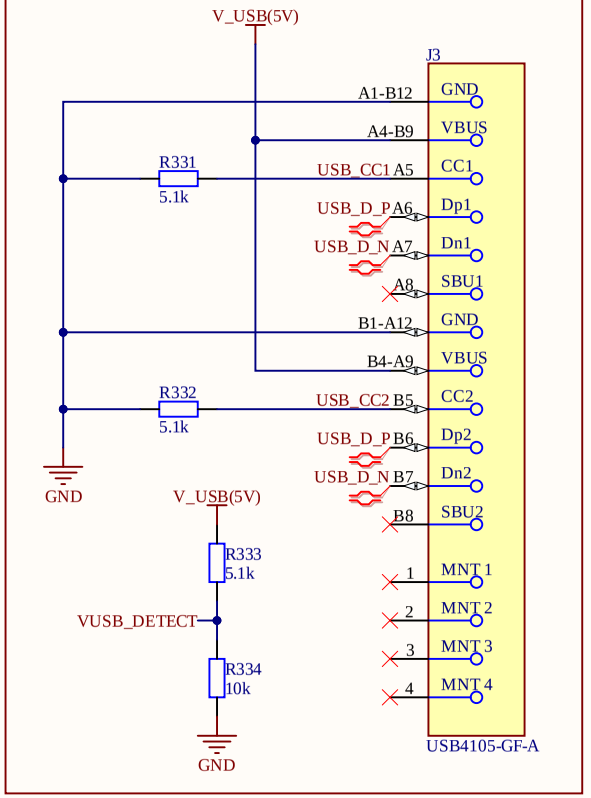


USB HID interface chips were unobtainable in June 2022. Landmark interfacing code already existing makes for simple PIC USB connectivity here.

On further development availability of USB communication chips can be checked.
Total cost of this is around 4.50CHF (06/22). Replacing the PIC18 with a simpler and cheaper PIC16F1455 which also has integrated USB hardware brings down costs to 3.50CHF which rivals some of the cheapest USB communication ICs (FTDI)

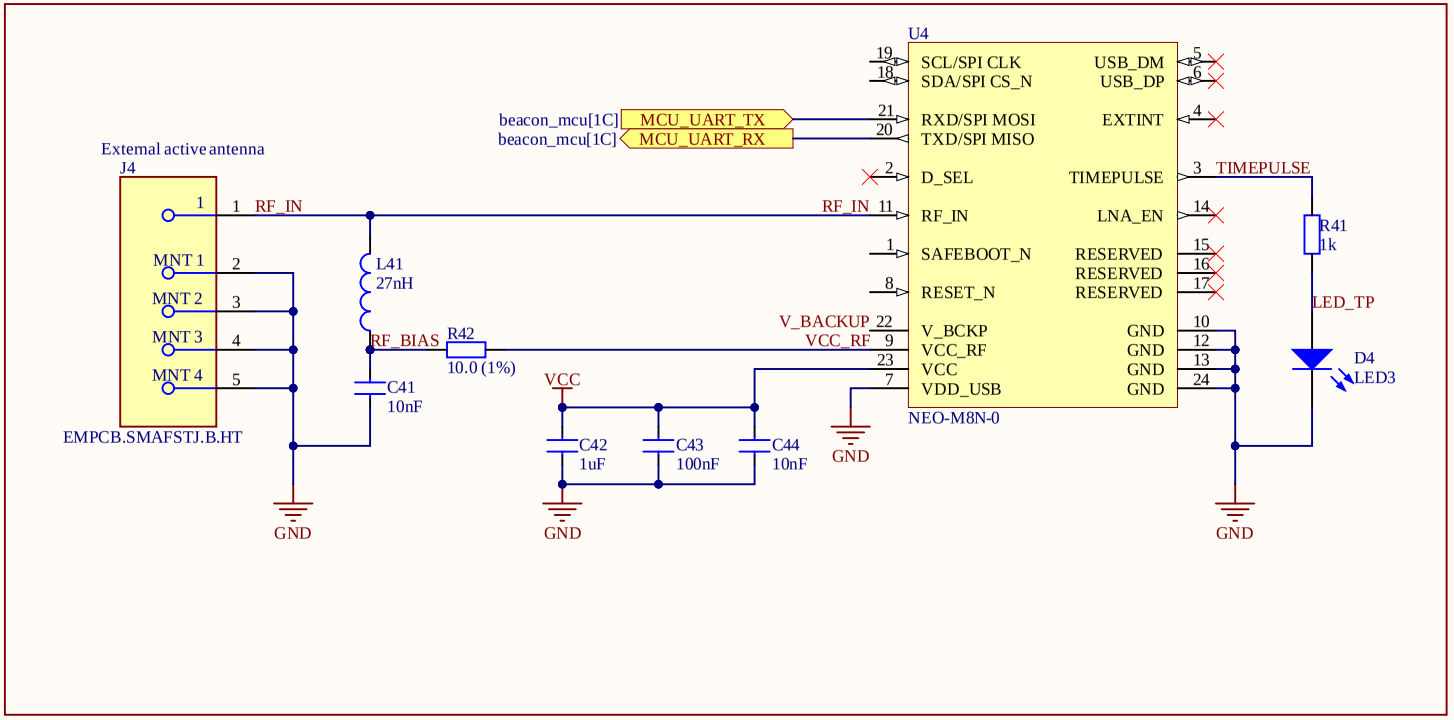
USB-C Connector (16p)

Section 3

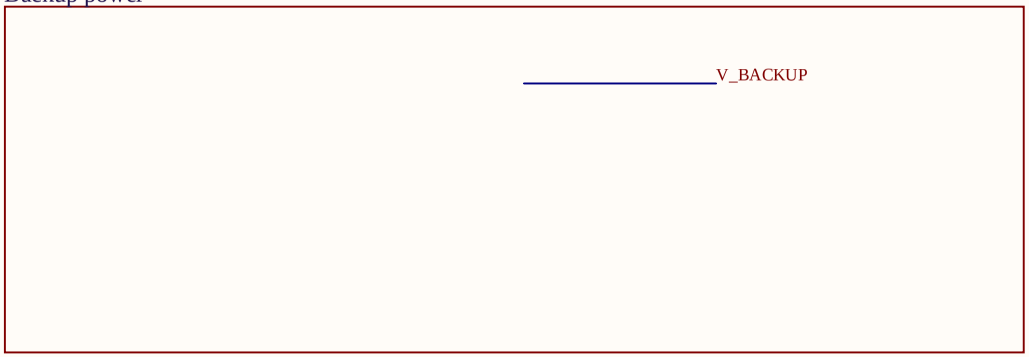


Title: Beacon USB		
Size: A4	Number:	Revision:
Date: 7.22.2022		Sheet 3 of 6
File: D:\AppsSonstige\..\beacon_usb.SchDoc		Drawn By: Jonas Elmiger

u-blox NEO-M8N GNSS module with active antenna



Backup power



VCC: 3.3V, max 67mA
V_BCKP: 1.4-3.6V, 15uA
VDD_USB: if unused must GND
VCC_RF: Supply for ext ant/LNA, max 100mA
SAFEBOOT_N: Leave open

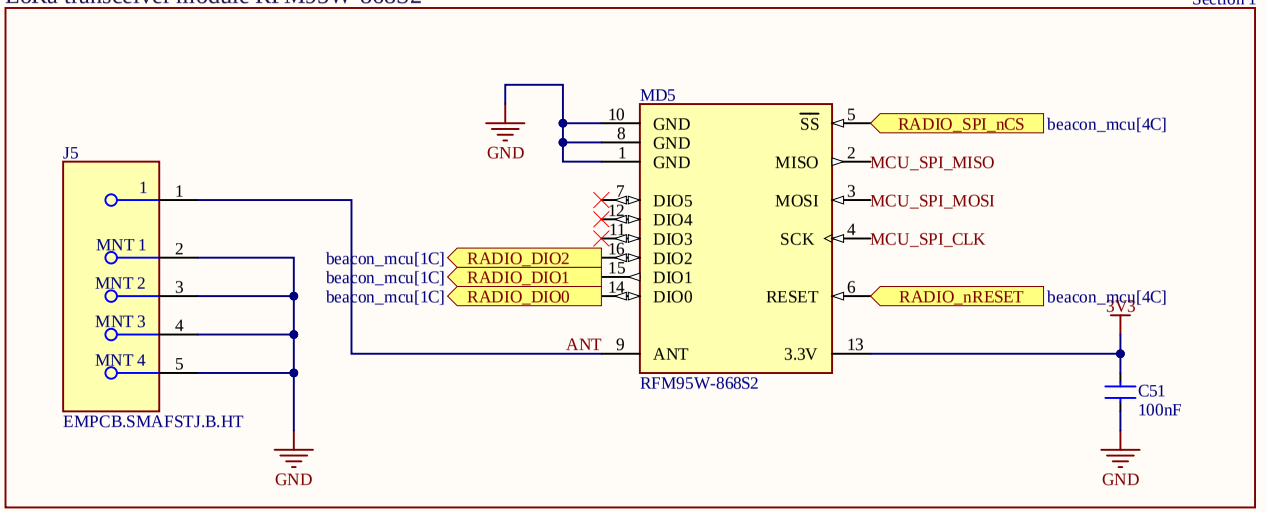
D_SEL: open->UART, low->SPI

RESERVED: leave open

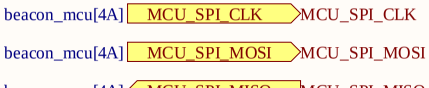
Title: Beacon GNSS		
Size: A4	Number:	Revision:
Date: 7.22.2022		Sheet 4 of 6
File: D:\AppsSonstige\..\beacon_gnss.SchDoc		Drawn By: Jonas Elmiger

LoRa transceiver module RFM95W-868S2

Section 1

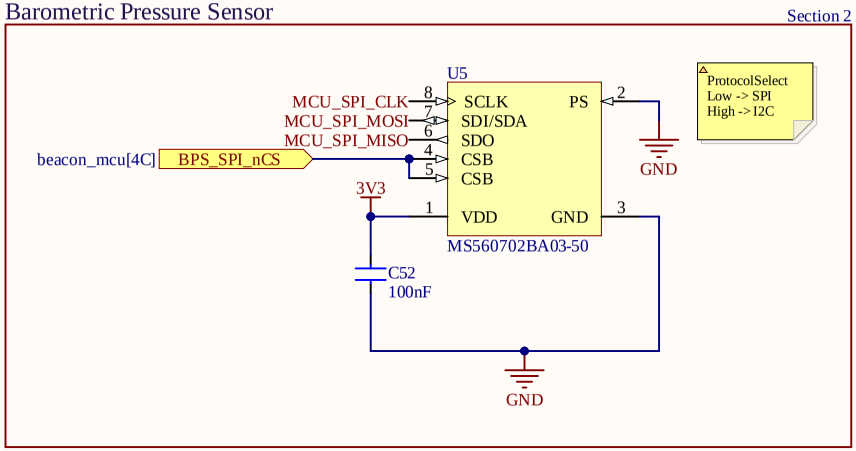


SPI connection

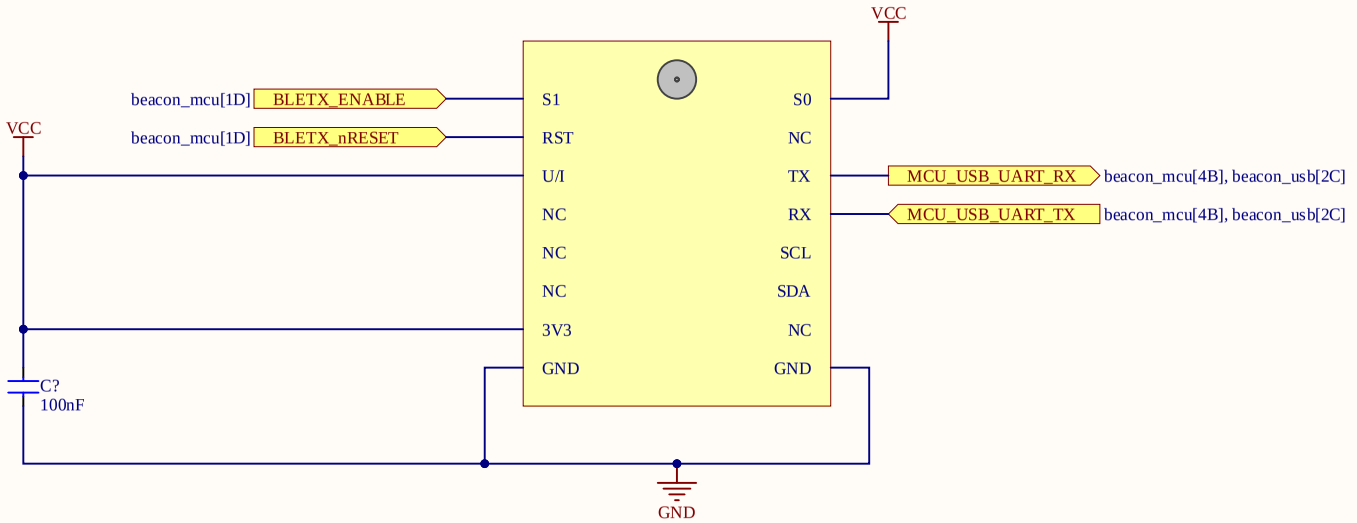


Barometric Pressure Sensor

Section 2



Title: Beacon SPI modules		
Size: A4	Number:	Revision:
Date: 7.22.2022		Sheet 5 of 6
File: D:\AppsSonstige\..\beacon_spi.SchDoc		Drawn By: Jonas Elmiger



Title: Beacon BLE module		
Size: A4	Number:	Revision:
Date: 7.22.2022		Sheet 6 of 6
File: D:\AppsSonstige\..\beacon_bletx.SchDoc		Drawn By: Jonas Elmiger