Global Oceans 2021: San Diego - Porto

Flexible Arbitrary Signal Generation and Acquisition System for Compact Underwater Measurement Systems and Data Fusion

Fabian John*, Sven Ole Schmidt*, Horst Hellbrück*†
*Technische Hochschule Lübeck, Germany
Department of Electrical Engineering and Computer Science, Center of Excellence CoSA
Email: fabian.john@th-luebeck.de, sven.ole.schmidt@th-luebeck.de, horst.hellbrueck@th-luebeck.de
† University of Luebeck, Germany, Institute of Telematics

***Abstract*—For the implementation of compact underwater measurement systems, much time is spent integrating various sensor and processing components and their control and synchronization. Especially for multi-sensor applications, proprietary solutions do not scale easily and are not designed for use in a distributed measurement system. We propose a well-established protocol Message Queuing Telemetry Transport (MQTT) for integration of sensors with a broker as a central data hub as a key system component. We implemented a flexible, distributed multi-sensor measurement system, by using open-source libraries. The multi-sensor system was successfully tested and operated with five underwater sensor devices and user data rates up to $2.73\ \mathrm{MB/s}$.**

***Index Terms*—Ultrasonic, MQTT, RedPitaya, Signalgenerator, Oscilloscope**

I. Introduction

Underwater multi-sensor systems analyze the environment with several sensors to detect and observe phenomona in the water or in the seabed. One of these application is the detection and localization of buried elements by their acoustic or electromagnetic properties which differ from the surrounding. Our measurement system targets position detection of buried objects in a layer of sediment, like, e.g., high-voltage cables, or the exploration of the sediment in general.

Ultrasonic-based underwater applications require a flexible and fast arbitrary signal generation and acquisition system [1]. While multiple systems exist, which fulfill the demand for controlling such applications, the usage for underwater localization systems yields some additional requirements. The generation and the analysis of ultrasonic signals need to be provided by a compact mobile system, supporting an open interface for control. This approach is also suitable for other applications like underwater electric impedance measurements [2], [3].

Based on these requirements, we present a system to generate and receive ultrasonic signals with a sample rate up to $125\ \mathrm{Msps}$ [4]. For localization, a communication link between the underwater unit and the processing unit for fusion algorithms is required, that is able to transfer the large amount of data with high reliability. Therefore, we implemented a communication link based on the Message Queuing Telemetry Transport (MQTT) protocol [5], [6], [7]. MQTT is an asynchronous event-based communication protocol, that provides reliable transfer of fast acquired digitized analog signals. The core component is the MQTT broker that serves as a a data hub and central component for flexible data acquisition and fusion in a distributed multi-sensor system.

With MQTT as a basis, we achieve a high degree of modularization in software and hardware in our distributed system independent of programming languages. The proposed MQTT-based solution is scalable and extensible with new algorithms, sensors, and software and hardware components. To increase the reusability of software and hardware components developed in a laboratory environment, we designed an MQTT-based control protocol for data aggregation and data post-processing.

The contributions of the paper are as follows:

- We develop a reliable underwater unit to flexible generate and acquire arbitrary signals, digitized with a sample rate up to $125\ \mathrm{Msps}$.
- Furthermore, we propose a MQTT protocol to increase the reusability of components developed during the research

Global Oceans 2021: San Diego - Porto

and development phases.
- We present the performance and data rates of the multi-sensor system in a gigabit laboratory network for our MQTT-based approach compared to the Standard Commands for Programmable Instrumentation (SCPI) based data transfer (the manufacturer's standard procedure for remote data acquisition for Matlab [8], [9]).

Th rest of the paper is structured as follows. First, we describe our approach and its implementation for a modular, robust, scalable, and performant multi-sensor system with analog sensors. Secondly, we present an example implementation for our performance analysis. The results of the performance analysis are shown in Section IV-B. Finally, we summarize the results of our work and give an outlook on further work.

II. Approach

The demand for a central data interface is derived from the research and development of multi-sensor systems for underwater object detection. Data from various systems with their specific interfaces and sampling rates are processed further in sensor fusion systems to provide a measurement result. Especially for the focus on algorithm development, a messaging protocol (MQTT) for data connection was chosen, in which data from all systems are integrable for fusion.

According to the publish and subscribe principle, the MQTT protocol distributes data, with the broker acting as the central data hub [10]. An MQTT message consists of a topic that describes the data and a flexible payload. Clients register with the broker for one or more topics (subscription). When a message is received, the broker forwards it to all clients that have subscribed to the topic. The message payload is not defined in MQTT itself and is specified by the application that provides the data. Applications register as clients with the broker preferable via TCP/IP that provides reliable transfer in the distributed system. For integration in the measurement system, clients connect to the broker via the given address and port.

For our underwater unit with flexible signal generation and acquisition, we use a RedPitaya 125-14. For reliable communication, we have developed software adapts the MQTT messages to the RedPitaya API. The MQTT-API-adapter is a software written in C, and uses the Eclipse Mosquitto library for MQTT.

III. Implementation

The requirements for multi-sensor underwater measurement systems are very complex. In particular, the underwater components of the distributed system must be simple to set up, waterproof, energy-efficient, and robust to interference in the communication channel. In this section, we first describe the hardware structure of the underwater box. Subsequently, we elaborate on the MQTT-API-adapter software and describe the measures taken to ensure robust communication. Finally, we explain the architecture of our distributed measurement system and show the flexible integration possibility of other components.

A. RedPitaya Underwater Box

We select a RedPitaya for signal generation and acquisition in our measurement system. Figure 1 shows our compact RedPitaya box, which is applicable underwater. The box contains a RedPitaya 125-14 (Figure 1 C), a printed circuit board (PCB) for analog signal amplification [1], and two boards for DC voltage adaption. The box provides a water-resistant connection to the mother vessel (Figure 1 A) and four cable ducts (Figure 1 B) to apply sensors, like the hydrophones shown in Figure 1 D. Unused cable ducts are sealed with a dummy plug. The box is closed with a screwed lid watertight.

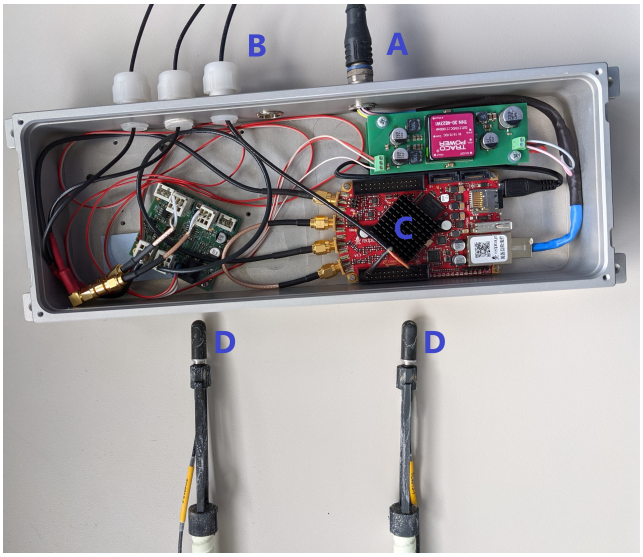

Figure 1. Compact underwater RedPitaya Box with A: Power and network connector, B: Sensor and trigger connectors, C: RedPitaya, D: Rx and Tx Hydrophones.

Table I summarizes the specifications of our underwater box. Any analog sensors and actuators are connectable to the box. The analog voltage gain via the PCB is variably adjustable in steps up to a factor of 100. The components in the underwater box start as soon as the power supply is applied; manual interactions for commissioning are not required. The monitoring of the operating status is described in Section III-B.

TABLE I
Specification of the RedPitaya underwater box.

Parameter	**Value**
Data rate	up to 2.73 MB/s
Sample rate	up to 125 Msps
Max output	± 10 V
Power supply	$18 - 36$ VDC
Connector	Female M12-PoE
Dimension	300 mm $\times$ 110 mm $\times$ 55 mm
Depth rating	100 m
Weight	< 2 kg
Underwater weight	≈ 0.2 kg

B. MQTT-API-Adapter

In this section, we describe the software developed to use the RedPitaya 125-14 with MQTT. First, the RedPitaya was

Global Oceans 2021: San Diego - Porto

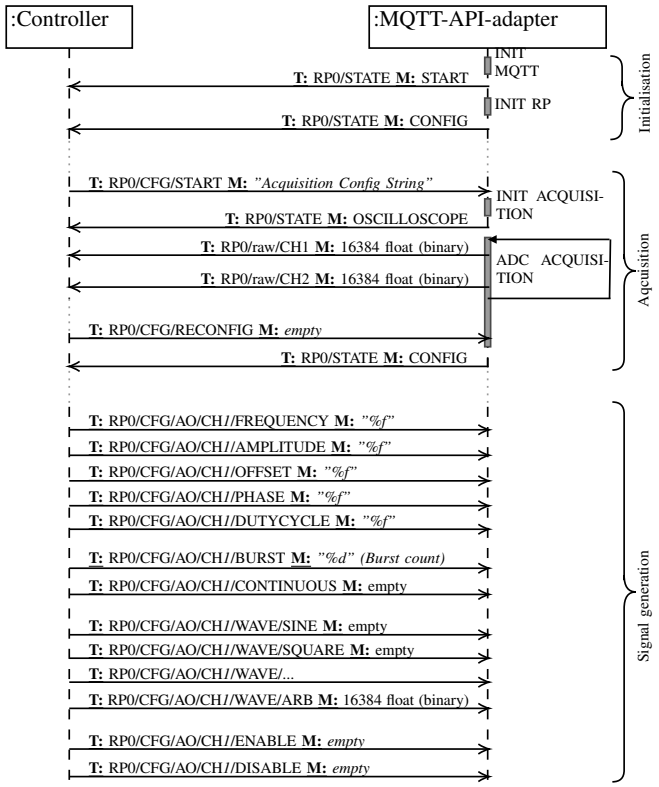

Figure 2. Sequence diagram of MQTT-API-adapter software.

put into operation with the current OS image STEMlab 1.04-7. No changes were made to the image, and thus the full functionality of RedPitaya (e.g., web interface, SCPI, etc.) is still available. The RedPitaya is used with a dynamic IP address, that is received over a DHCP-server in the network. Only the IP address or domain name of the MQTT brokers' computer must be known at the client side.

Our MQTT-API-adapter is implemented in C and uses the mosquitto library for MQTT communication and the API of the RedPitaya device to generate and acquire signals. The software starts automatically after the network services are started, and the service restarts automatically when the program exits with an error code. Thus, even in the event of a fault, high availability is achieved without manual interaction.

Figure 2 shows the message flow between a controller and the RedPitaya to access data acquisition and signal generation functionalities. At program start the MQTT connection is established and then the initialization of the RedPitaya library is performed. The state of the program is observed through the related topic. The broker IP address and device identifier, that is used as root topic (e.g., *RP0* in the example in Figure 2), are adaptable via the configuration file.

The data acquisition loop is implemented, based on the fast signal acquisition example in [8]. The acquisition is starts with a message, containing a configuration string as payload. The configuration string contains information to adjust the trigger, the sample rate, and sample time. After receiving the configuration, the data acquisition is initialized, and the acquisition loop is started. In the acquisition loop, the two input channels are read according to the trigger settings, and the floating-point buffers are transmitted via MQTT as two messages with binary payload of $\approx 65\,\mathrm{kB}$. The acquisition loop runs continuously until the device is reset to configuration mode by a message from the controller (see Figure 2).

The signal generation is configured via MQTT messages. The output of various standard signals (e.g., sine, square, sawtooth, etc.), as well as arbitrarily defined signals, is possible. Arbitrary signals are sent to the device as a sequence of 16384 floating-point values. Other signal properties (e.g., frequency, amplitude, etc.) are transmitted via MQTT messages for each channel. The signal output can be continuous or in burst mode, which is also set via an MQTT message. The activation and deactivation of each channel are done selectively via corresponding messages.

The RedPitaya reports the current operating state and occurring errors via related topics *RP0/STATE* and *RP0/ERROR*, whereby problems can be detected and corrected in the controller. It is also possible to perform a software reset of the MQTT-API-adapter via MQTT message. In addition, the MQTT connection to the broker in the adapter is monitored cyclically, and the software restarts automatically if problems occur in the connection channel. With the measures taken, a robust and flexible use of RedPitaya is guaranteed.

For further details or reuse of our MQTT-API-adapter, we refer to the software project repository[1].

C. Distributed Measurement System

Figure 3 shows the architecture of a distributed underwater multi-sensor system, consisting of several AD/DA units underwater (*RP0 - RPN*), a computer as MQTT-Broker (MQTT-server) and DHCP-server (Server 1), a computer to run algorithms and perform the data fusion (PC 2), and a control computer for displaying, user interaction and controlling the operations. All devices are connected via Ethernet. The system shown in Figure 3 was used to perform the performance measurements in Section IV for example. Depending on the required computing power for display, control, and data fusion, the hardware on the mother vessel is further distributed, combined, or virtualized.

The analog sensors (e.g., ultrasonic transducers, EIT-electrodes) are connected to the AD/DA underwater units. The Meas Controller in Figure 3 controls the signal generation and acquisition from the sensors via MQTT as described in Section III-B. The acquired measurement data is further processed in the data fusion to produce a measurement result, and this is also provided via MQTT. Graphical user interfaces (GUI's) are used for status and error monitoring and for displaying measurement data and the measurement result. All components are implemented as independent modules and connected via the MQTT broker as a data hub.

The expansion of the multi-sensor system with additional components or the exchange of individual components is possible during operation. The only requirement for integration into

[1] https://git.mylab.th-luebeck.de/fabian.john/rp-mqtt-api-wrapper.git

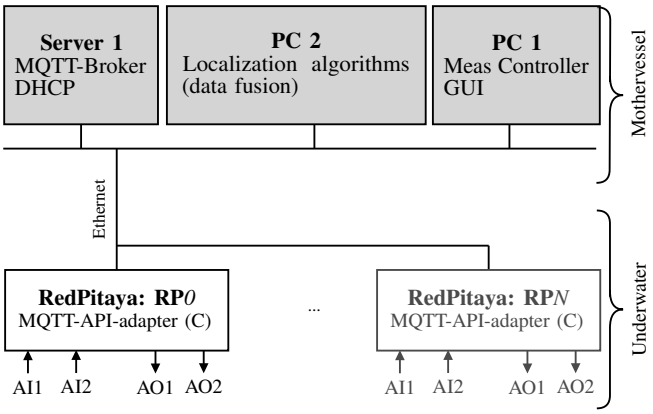

Figure 3. MQTT-based multi-sensor system with data fusion.

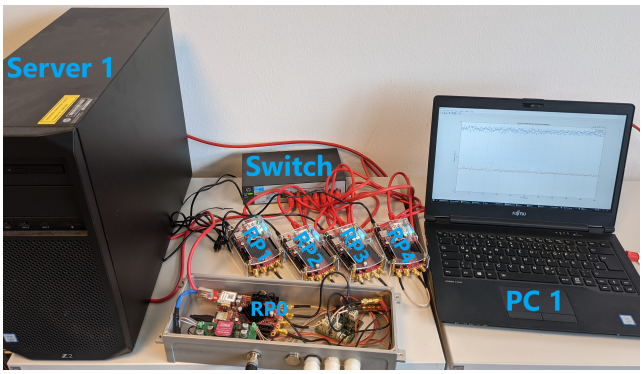

Figure 4. Setup for performance analysis measurements.

the merged system is using the MQTT protocol and knowledge of the MQTT broker IP address for new sensor components. The integration of sensors connected serially via UART is also done via a RedPitaya with an MQTT-serial-adapter[2]. This software allows sending serial commands remotely via the MQTT protocol and forwarding serially received packets via MQTT. Our MQTT-serial adapter integrates, e.g., Tritech's Micron Echosounder altimeter [11] into the fused system to determine the position of the multi-sensor system to the seafloor.

The distributed architecture shown in Figure 3 is modular, scalable. It can be extended to meet different measurement requirements, with the number of underwater units limited by the speed of the network connection to the mother vessel. Due to the asynchronous and event-based data flow of the MQTT messages, the system can be operated robustly even if larger latencies occur in the packet switching. However, the temporal allocation of the distributed recorded measurements requires additional measures (e.g., external trigger, synchronization).

IV. Performance Evaluation

In this section, we show the results of the performance analysis performed for our RedPitaya underwater box. The analysis was performed in a simplified network, without data fusion algorithms. To measure our underwater box's maximum achievable user data rate of our underwater box, the measurement was first performed with one underwater box. Then the measurement was repeated with 2 and 5 underwater boxes to measure the performance in the multi-sensor system. The setup and execution of the measurements are described in Section IV-A, and the results are presented in Section IV-B.

A. Performance Analysis Setup

To measure the maximum achievable data rate, the architecture shown in Figure 3 was set up in a simplified manner in the laboratory Server 1, PC 1, and one, two, and five underwater boxes. The components were connected via a local Gigabit network through a switch. All components used for the evaluation measurements are shown in Table II and Figure 4.

TABLE II
Components used for the evaluation measurements.

Parameter	Value
Server 1	
CPU	Intel(R) Core(TM) i7-9700K @3.6 GHz
RAM	16 GB
Network	1 GBit
OS	64 Bit Windows 10
PC 1	
CPU	Intel(R) Core(TM) i7-8550U @1.8 GHz
RAM	16 GB
Network	1 GBit
OS	64 Bit Windows 10
RedPitaya (RP0 - RP4)	
CPU	Dual-Core ARM Cortex-A9 MP-Core @1.8 GHz
RAM	512 MB
Network	1 GBit
OS	STEMlab OS 1.04-7
Network Switch	
Type	HP 1810-8G
Network	1 GBit

For the performance measurement, the first analog output on RedPitaya was connected to the first analog input. Then, a continuous sine wave signal with 200 kHz was output, and triggered to the first input channel. Both input channels were recorded at the highest sample rate (125 Msps) and transmitted as a $\approx$ 56 kB packet to PC 1. The measured data of both channels were stored in PC 1 in a floating-point array. The time T required to receive the packets, including storage in the array, was measured for both channels in PC 1.

For comparison, the measurement with a RedPitaya was also performed via SCPI, the manufacturer's standard procedure for remote data acquisition for Matlab [8], [9], and the time required to receive the packets, including storage in the floating-point array, was measured too. The measurement was performed repetitively $N = 500$ times in each case to calculate the

[2]https://git.mylab.th-luebeck.de/fabian.john/rp-mqtt-serial-wrapper.git

maximum achievable data rate as the mean value $\overline{T}$ of the measured times T_i with Equation 1.

$$\overline{T} = \frac{\sum_{i=1}^{N} T_i}{N} \tag{1}$$

For each measurement, $2 * k$ channels of 16384×4 B user data are transmitted (with k RedPitaya devices), resulting in the maximum achievable data rate R calculated by Equation 2.

$$R = \frac{2 * \mathrm{k} * 16384 * 4}{\overline{\mathrm{T}}} \ \mathrm{B/s} \tag{2}$$

The results of the performance analysis are presented in the next section Section IV-B.

B. Results of Performance Analysis

Figure 5 shows the measured times T for the performance analysis with one, two, and five RedPitaya devices in parallel for our proposed MQTT solution compared to the SCPI based acquisition.

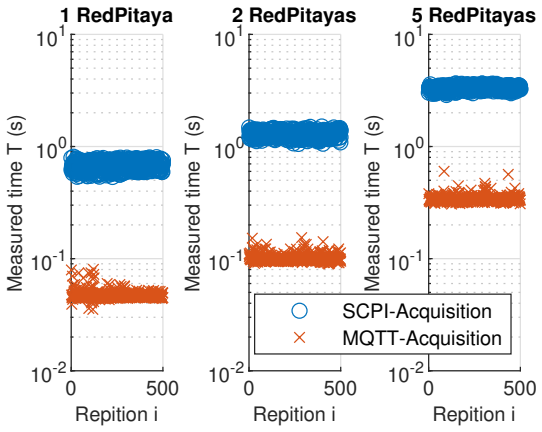

Figure 5. Results of measured time to receive and store analog data.

Our proposed MQTT acquisition is $\approx$ 10 times faster compared to the SCPI based acquisition for all setups. Our MQTT-based data transfer sends the floating-point array in the binary representation as message payload, unlike the SCPI that uses marshalling to generate the MQTT message payload of the floating point array, which produces a high message parsing effort.

The calculated mean values $\overline{T}$ of the measured times for the performance analysis and the data rates R are presented in Table III.

The highest data rate was achieved with one RedPitaya in the network with our MQTT-based data transfer with 2.73 MB/s. The data rates are higher by a factor of ≈ 10 with the MQTT-based data transfer compared to the SCPI-based data transfer in all measurements performed.

V. Conclusion

This paper presents a multi-sensor measurement system with an MQTT-broker as a central data hub for the flexible integration of sensors and algorithms. The asynchronous event-based distribution of data results in a modular and highly scalable system. Computationally intensive algorithms can be outsourced to additional hardware resources without having to reconfigure dependent components.

TABLE III
MEASUREMENT RESULTS OF THE PERFORMANCE ANALYSIS.

Measurement	$\overline{T}$ (SD) **in** s	R **in** MB/s
1 RedPitaya		
MQTT	0.048 (0.005)	2.73
SCPI	0.676 (0.060)	0.19
2 RedPitayas		
MQTT	0.101 (0.008)	2.60
SCPI	1.291 (0.083)	0.20
5 RedPitayas		
MQTT	0.343 (0.025)	1.90
SCPI	3.319 (0.135)	0.20

The binary representation of the floating-point array, that we use as MQTT message payload, achieve a very high data rate in our MQTT-based system due to less data transfer and fast datatype cast operation of the received message payload compared to marshalling and unmarshalling used in the SCPI based data transfer. The data rate of up to 2.73 MB/s is $\approx$ 10 higher compared to SCPI based data transfer, which is a standard approach for remote data transfer in Matlab [9].

The MQTT-API-adapter software presented in this paper is available as an open source project[3]. The proper operational state of the devices is monitored continuously via MQQT protocol (status and error messages). Connection problems of the MQTT communication are detected by software and trigger a restart of the software to reconnect to the broker. With these error detection and error handling routines, our system is very robust.

In future, we use a setup with 10 RedPitaya underwater boxes for a localziation system based on EIT measurements that is currently under development. We will extend the MQTT-API-adapter further with missing features for this and future applications.

Acknowledgments

This publication results from the research of the Center of Excellence CoSA at the Technische Hochschule Lübeck and is funded by the Federal Ministry of Economic Affairs and Energy of the Federal Republic of Germany (Id 03SX467B, Project EXTENSE, Project Management Agency: Jülich PTJ). Horst Hellbrück is an adjunct professor at the Institute of Telematics of University of Lübeck.

References

[1] F. John, R. Kusche, F. Adam, and H. Hellbrück, "Differential ultrasonic detection of small objects for underwater applications," in *Global Oceans 2020: Singapore – U.S. Gulf Coast*, Oct 2020, pp. 1–7, https://ieeexplore.ieee.org/document/9389186. [Online]. Available: http://cosa.th-luebeck.de/download/pub/john-2020-differential-ultrasonic-oceans.pdf

[3]https://git.mylab.th-luebeck.de/fabian.john/rp-mqtt-api-wrapper.git

Global Oceans 2021: San Diego - Porto

[2] A. Schuldei, F. John, G. Ardelt, T. Suthau, and H. Hellbrück, "Development of an Electro Impedance Tomography-based Platform for Measurement of burial Depth of Cables in Subsea Sediments," in *Oceans 2019*, 2019.

[3] G. Ardelt, M. Mackenberg, and H. Hellbrück, "Wireless underwater communication via analog ofdm modulated light," in *Proceedings of the International Conference on Underwater Networks & Systems*, ser. WUWNET'17. New York, NY, USA: ACM, 2017, pp. 18:1–18:3. [Online]. Available: http://doi.acm.org/10.1145/3148675.3148722

[4] "RedPitaya fast analogue io," https://redpitaya.readthedocs.io/en/latest/developerGuide/125-14/fastIO.html, accessed: 2021-04-30.

[5] N. Naik, "Choice of effective messaging protocols for iot systems: Mqtt, coap, amqp and http," in *2017 IEEE international systems engineering symposium (ISSE)*. IEEE, 2017, pp. 1–7.

[6] S. Profanter, A. Tekat, K. Dorofeev, M. Rickert, and A. Knoll, "Opc ua versus ros, dds, and mqtt: performance evaluation of industry 4.0 protocols," in *2019 IEEE International Conference on Industrial Technology (ICIT)*. IEEE, 2019, pp. 955–962.

[7] D. Silva, L. I. Carvalho, J. Soares, and R. C. Sofia, "A performance analysis of internet of things networking protocols: Evaluating mqtt, coap, opc ua," *Applied Sciences*, vol. 11, no. 11, p. 4879, 2021.

[8] "RedPitaya on trigger signal acquisition," https://redpitaya.readthedocs.io/en/latest/appsFeatures/examples/acqRF-exm1.html, accessed: 2021-07-02.

[9] Z. Xu, J. Yao, Z. Wang, Y. Liu, H. Wang, B. Chen, and H. Wu, "Development of a portable electrical impedance tomography system for biomedical applications," *IEEE Sensors Journal*, vol. 18, no. 19, pp. 8117–8124, 2018.

[10] G. Hillar, *MQTT Essentials - A Lightweight IoT Protocol*. Packt Publishing, 2017. [Online]. Available: https://books.google.de/books?id=40EwDwAAQBAJ

[11] "Tritech Micron Echosounder altimeter," https://www.tritech.co.uk/media/products/Micron\%20Echosounder.pdf?id=8332446d, accessed: 2021-07-20.