Diplomarbeit

im Studiengang Informatik

Thema:

Entwicklung einer Demo-Anwendung unter Nutzung des EnOcean-Funkprotokolls

Torsten Ketelhut
Matrikel-Nr. 33324

eingereicht am: 15.08.2018

Betreuer: Prof. Dr.-Ing. Jörg Vogt
Zweitgutachter: Prof. Dr.-Ing. Robert Baumgartl

Inhaltsverzeichnis

Abbildungsverzeichnis V
Tabellenverzeichnis VII
Verzeichnis verwendeter Abkürzungen und Symbole IX
1 Einleitung 1
1.1 Motivation 1
1.2 Beschreibung der Aufgabenstellung 1
1.3 Inhalt der Arbeit 1
2 Grundlagen 3
2.1 Grundlagenübersicht 3
2.2 868 MHz Funkband 3
2.3 Serielle Kommunikation über UART 4
2.4 EnOcean Equipment Profile (EEP) 6
2.5 EnOcean Serial Protocol (ESP) 7
2.6 EnOcean Radio Protocol (ERP) 9
2.7 Funktionsweise von kapazitiven Sensoren 11
3 Entwurf 13
3.1 Verwendete Programme und Tools 13
3.1.1 Dolphin View Basic 13
3.1.2 HTerm und Virtual Serial Port Driver 14
3.1.3 Node.js und Node Package Manager (npm) 14
3.1.4 Atmel Studio 7 15
3.2 Datenübertragungsweg 15
3.2.1 Auswahl des geeigneten Senders 16
3.2.2 Auswahl des geeigneten Empfängers 16
3.3 Datenauswertung durch einen externen Mikrocontroller 17
3.4 Auswahl des Erdfeuchtesensors 17
3.5 Darstellung der Messwerte 18
4 Umsetzung der Demoapplikation 20
4.1 Aufbau der Hardware 20

4.1.1 Benötigte Komponenten ... 20
4.1.2 Verdrahtung des Aufbaus ... 20
4.2 Scavanger Transmitter Module 310 (STM310) ... 23
4.2.1 Allgemeines und Typenbeschreibung ... 23
4.2.2 Energieverbrauch ... 23
4.2.3 Steuereinheit des STM310 ... 24
4.2.4 Konfigurationsverfahren des STM310 ... 25
4.2.5 Daten- und Teach-In Telegramm ... 27
4.2.6 Analog Digital Converter (ADC) ... 28
4.3 Transceiver Module 320 (TCM320) ... 30
4.3.1 Allgemeines und Typenbeschreibung ... 30
4.3.2 Spannungsversorgung und Energieverbrauch ... 30
4.3.3 Konfigurationsverfahren des TCM320 ... 31
4.3.4 Serieller Übertragungsmodus und Übertragungsgeschwindigkeit ... 33
4.4 Externer Mikrocontroller ATmega8 ... 33
4.4.1 Allgemeines ... 33
4.4.2 UART Initialisierung und Auswahl des passenden Quarz ... 33
4.4.3 Interrupt Service Routine ... 36
4.4.4 Analyse empfangener Daten ... 37
4.5 User Interface ... 39
4.5.1 Kommunikationsweg ... 39
4.5.2 Tests zur Datenübertragung ... 39
4.5.3 Datenanalyse und Darstellung ... 41
5 Auswertung ... 44
5.1 Nutzbarkeit ... 44
5.2 Datenübertragung ... 45
5.3 Geräte- und Datenidentifizierung ... 45
5.4 Konfigurationsfähigkeit ... 46
6 Messversuch zur Bestimmung der Erdfeuchtigkeit ... 48
6.1 Versuchsaufbau ... 48
6.1.1 Messversuch mit Potentiometer ... 48

6.1.2 Messversuch mit zusätzlicher Elektronik und einem kapazitiven Sensor.. 48
6.2 Ergebnisauswertung 50
7 Zusammenfassung 53
7.1 Fazit 53
7.2 Ausblick 54
Literatur- und Quellenverzeichnis 55
Anhang 59
Thesen zu den wesentlichen Arbeitsergebnissen i
Selbständigkeitserklärung iii

Abbildungsverzeichnis

Abbildung 2-1: Übersicht der Datenübertragung mit ESP, EEP und ERP aus [ENA17] 3
Abbildung 2-2: UART Verbindung zwischen zwei Geräten 5
Abbildung 2-3: TTL-Spannungspegel aus [WÜS11] 5
Abbildung 2-4: 4 BS-Teach-In Telegramm aus [ENA17] 6
Abbildung 2-5: Überblick zur Erstellung und Ausgabe neuer EEP-Profile aus [ENA17] 7
Abbildung 2-6: ESP Paket mit serieller Kommunikation aus [ENO18] 7
Abbildung 2-7: Unterschiede ESP2 zu ESP3 aus [ENO18] 8
Abbildung 2-8: Aufbau ESP3 aus [ENO18] 9
Abbildung 2-9: Aufbau ESP2 aus [ENO17e] 9
Abbildung 2-10: Struktur des Subtelegramms aus [ENO13] 10
Abbildung 2-11: Optionen des Status Byte aus [ENO13] 10
Abbildung 2-12: Veränderung des Dielektrikums aus [HER18]l 11
Abbildung 3-1: Ausschnitte von Dolphin View Basic 13
Abbildung 3-2: Beispieldarstellung empfangener Telegramme in HTerm über USB300-Stick 14
Abbildung 3-3: Datenübertragungsweg 15
Abbildung 3-4: Capacitive Soil Moisture Sensor der Firma DFRobot 18
Abbildung 3-5: Darstellung offener Kondensator aus [HER18] 18
Abbildung 4-1: Beschaltung Empfängerseite TCM320 / ATmega8 21
Abbildung 4-2: Beschaltung Senderseite STM310 mit Potentiometer als Sensor 22
Abbildung 4-3: Betriebszeit STM310 aus [ENO12] 24
Abbildung 4-4: Blockdiagramm EO3000I aus [ENO14] 24
Abbildung 4-5: Konfigurationsmöglichkeiten zur erneuten Übertragung von redundanten Daten aus [ENO12] 25
Abbildung 4-6: Programmierung über Serial Interface aus [ENO12] 26
Abbildung 4-7: Aufbau eines Teach-In Telegramms aus [ENO12]l 27
Abbildung 4-8: Aufbau der Datenbytes von einem Datentelegramm aus [ENO12] 28
Abbildung 4-9: Gain- und Offset-Error aus [ENO12] 29
Abbildung 4-10: Beispiel „Rail-to-Rail“-Sensoranschluss an STM310 aus [ENO12] 29
Abbildung 4-11: Darstellung Pinout für TCM300 und TCM320 aus [ENO17e] 30

Abbildung 4-12: Spannungsbereiche und zugehörige Betriebsmodi des TCM320 aus [ENO17e] 32
Abbildung 4-13: Erläuterung der Betriebsmodi des TCM320 aus [ENO17e] 32
Abbildung 4-14: UART Initialisierung verändert nach [ANO17] 34
Abbildung 4-15: Codebeispiel zur Nutzung einer ISR in Verbindung mit der UART 36
Abbildung 4-16: Funktionsweise des Schreibzugriffs auf den Ringbuffer 37
Abbildung 4-17: Codebeispiel zur Datenspeicherung aus dem Ringbuffer verändert nach [BEC14] 37
Abbildung 4-18: Funktionsweise des Lesezugriffs auf den Ringbuffer 38
Abbildung 4-19: Codebeispiel zur Erkennung von Geräte-ID und Teach-In Telegramm 38
Abbildung 4-20: Kommunikationsweg vom STM310 zum virtuellen COM-Port aus [ENO17c] 39
Abbildung 4-21: Darstellung von Daten- und Teach-In Telegramm nach ESP3 in HTerm 40
Abbildung 4-22: Datenverlauf durch HTerm, Virtual Serial Port Driver und Node.js 40
Abbildung 4-23: Startprozedur des User Interface per Batch-Datei 41
Abbildung 4-24: Server-Client Kommunikation am Beispiel eines Starts der Messwertaufzeichnung 42
Abbildung 4-25: User Interface mit Messwerten 43
Abbildung 5-1: Initialisierungsprozess mit erster Datenübertragung 46
Abbildung 6-1: Messaufbau mit einem Potentiometer und Referenz- bzw. Messspannungen 48
Abbildung 6-2: Messversuch mit kapazitivem Sensor und zusätzlicher Elektronik 49
Abbildung 6-3: Sensorelektronik des kapazitiven Erdfeuchtesensors von DFRobot aus [DFR15] 49
Abbildung 6-4: Entwurf der zusätzlichen elektronischen Schaltung 50
Abbildung 6-5: PSPICE Simulation des Messversuchs mit Potentiometer bei 0 Ω 50
Abbildung 6-6: PSPICE Simulation des Messversuchs mit Potentiometer bei 9,6 kΩ 51

Tabellenverzeichnis

Tabelle 2-1: Frequenzlimitierung verändert nach [ETS16] 4

Tabelle 2-2: Permittivitätszahlen verschiedener Materialien verändert nach [HER18] und [HIP95] 11

Tabelle 4-1: Fehleranfälligkeit bei genutzter Baudrate und Betriebsfrequenz des ATmega8 verändert nach [GRE06] 34

Tabelle 4-2: Aufbau und Beschreibung des UCSRB-Registers verändert nach [ANO17] 35

Tabelle 4-3: Aufbau und Beschreibung des UCSRC-Registers verändert nach [ANO17] 35

Tabelle 6-1: Potentiometerstellung mit den dazugehörigen Empfangsdaten des Messversuchs 51

Verzeichnis verwendeter Abkürzungen und Symbole

µ	Mikro-
µC	Mikrocontroller
1 BS	EnOcean 1 Byte communication
4 BS	EnOcean 4 Byte communication
A	Plattenfläche in [m²], Ampere
ADC	Analog Digital Converter
AES	Advanced Encryption Standard
ANM	Analoges Messsignal
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
Bd	Baud
C	Kapazität in [F]
COM	Communication Port
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CSS	Cascade Style Sheet
d	Abstand der Platten in [m]
DI	Digital Input
e.r.p.	effective radiated power
EEP	EnOcean Equipment Profile
EN	Europäische Norm
ERP	EnOcean Radio Protocol
ESP	EnOcean Serial Protocol
ESPB	EnOcean Serial Protocol Bytes
ETSI	European Telecommunications Standards Institute
ε_0	elektrische Feldkonstante in [As/Vm]
ε_r	Permittivitätszahl
F	Farad
FTDI	Future Technology Devices International
GND	Ground
h	Stunde
HTML	Hypertext Markup Language
Hz	Hertz
I²C	Inter-Integrated Circuit
ID	Identifikator
ISR	Interrupt Service Routine
k	Kilo-
LBT	Listen Before Talk, Listen Before Transmit
LED	Light Emitting Diode
lx	Lux
m	Meter, Milli-
M	Mega-
n	Nano-
NC	Not Connected
NO-Taster	Normally Open Taster
npm	node package manager

ORG	Organizational number for EnOcean radio telegram types
OSI	Open Systems Interconnection
PSPICE	Personal Simulation Program with Integrated Circuit Emphasis
PTM	Pushbutton Transmitter Module
RF	Radio Frequency
RORG	Radio ORG
RPS	EnOcean telegram type for Repeated Switch Communication
RS-232	Recommended Standard 232
RSSI	Receive signal strength
Rx	Receive
RXC	UART Receive Complete
s	Sekunde
SPI	Serial Peripheral Interface
STM	Scavanger Transmitter Module
TCM	Transceiver Module
TTL	Transistor-Transistor-Logik
Tx	Transceive
UART	Universal Asynchronous Receiver and Transmitter
UBRR	UART Baud Rate Register
UBRRH	UART Baud Rate Register High
UBRRL	UART Baud Rate Register Low
UCPOL	UART Clock Polarity
UCSRA	UART Control and Status Register A
UCSRB	UART Control and Status Register B
UCSRC	UART Control and Status Register C
UCSZ	UART Characters Size
UDR	UART Data Register
UMSEL	USART Mode Select
UPM	UART Parity Mode
URSEL	UART Register Select
USB	Universal Serial Bus
USBS	USART Stop Bit Select
V	Volt
VLD	EnOcean Variable Length Data Telegram
W	Watt
XML	Extensible Markup Language
Ω	Ohm

1 Einleitung

1.1 Motivation

Heutzutage existiert eine Vielzahl von Sensoren für verschiedene Anwendungszwecke. Schalter und Positionskontrollen sind in der Regel kabelgebunden. Dies hat einen großen Installationsaufwand zur Folge. Selbst funkbasierte Sensoren, wie ein Rauchmelder, existieren schon. Allerdings sind diese in der Regel nicht wartungsfrei, da sie mit einer Batterie betrieben werden, die in Regelmäßigkeit ausgetauscht werden muss. Die EnOcean GmbH hat mit ihren Funkprodukten eine wartungsfreie Übertragung von geringen Datenmengen geschaffen. Durch den geringen Energiebedarf und die Selbstversorgung kann der Installationsaufwand der Sensoren erheblich gesenkt werden. Die verschiedenen Methoden der Energiegewinnung, wie z. B. durch Tastendruck oder per Solarenergie, lassen verschiedene Anwendungen zu. Diese Technologie ist zukunftsorientiert und wird sich im Laufe der Zeit mehr und mehr etablieren und vermutlich die bisherigen Systeme ablösen.

In der vorliegenden Arbeit soll die EnOcean Technologie genutzt werden. Es ist interessant, die Möglichkeiten und das Prinzip der Funkübertragung mittels EnOcean Funkprotokoll zu erforschen und mit dem speziellen Anwendungsfall „Messung von Erdfeuchtigkeit“ zu kombinieren.

1.2 Beschreibung der Aufgabenstellung

In der vorliegenden Diplomarbeit soll eine Demoapplikation unter Verwendung des EnOcean Funkprotokolls entwickelt werden. Die Funktionsweise des Funkprotokolls ist ebenso Bestandteil, wie die Erforschung verschiedener EnOcean Module in Bezug auf etwaige Anwendungsmöglichkeiten. Mit den ausgewählten Komponenten wird eine Messung durchgeführt und über das Funkprotokoll an eine Empfangsstelle gesendet. Anschließend erfolgt eine Auswertung der übertragenen Daten. Die Messwerte sollen für einen Nutzer ebenso einsehbar sein. Dazu werden die Messergebnisse auf dem PC dargestellt.

Die Messwerte geben Auskunft über die Erdfeuchtigkeit von Blumenerde. Die mögliche Sensorart ist dabei in Verbindung mit einem passenden Modul auszuwählen.

1.3 Inhalt der Arbeit

Um die Erkenntnisse der Arbeit übersichtlich darzustellen, wird diese kapitelweise strukturiert.

In Kapitel 2 sind die wesentlichen Begriffe und Technologien der verwendeten Funkübertragung dargestellt. Außerdem ist die Funktionsweise der genutzten Messmethode erklärt.

Kapitel 3 befasst sich mit dem Entwurf zur Umsetzung der Aufgabenstellung. Es werden die zur Umsetzung ausgewählten Programme erläutert. Des Weiteren wird auf die Auswahl der nötigen Module und des Erdfeuchtesensors eingegangen. Abschließend erfolgt eine Beschreibung zur Darstellung der Messwerte.

Im Kapitel 4 wird eine genauere Betrachtung der verwendeten Komponenten durchgeführt und ein Überblick über die Verknüpfung dieser untereinander dargestellt. Die elektrischen Gegebenheiten, die Datenübertragung und die möglichen Konfigurationsmethoden der Elemente sind wesentlicher Bestandteil dieses Kapitels. Ebenso wird die Funktionsweise zur Datendarstellung für den Benutzer und die Datenauswertung eines eingegangenen Telegramms betrachtet.

Die Auswertung zur Nutzbarkeit der EnOcean Module und der Datenübertragung via EnOcean Funk- und Serialprotokoll ist in Kapitel 5 dargestellt. Es wird die Notwendigkeit hinsichtlich der Daten- und Geräteerkennung beschrieben. Außerdem werden die Konfigurationsmöglichkeiten von den Komponenten ausgewertet.

In Kapitel 6 wird ein Messversuch zur Erdfeuchtigkeit dargelegt. Der Versuchsaufbau und die Ergebnisauswertung sind ebenfalls Teil dieses Kapitels.

Das abschließende Kapitel 7 gibt ein Fazit der Arbeit und einen Ausblick auf die mögliche Erweiterung der Demoapplikation.

2 Grundlagen

2.1 Grundlagenübersicht

Die hier beschriebenen Grundlagen sind zum Verständnis der EnOcean Funkübertragung notwendig. Die wesentlichen Bestandteile der Datenübertragung mit EnOcean-Funkmodulen sind:

- das 868 MHz Funkband,
- die serielle Kommunikation über UART,
- das EnOcean Equipment Profile (EEP),
- das EnOcean Serial Protocol (ESP) und
- das EnOcean Radio Protocol (ERP).

Der generelle Ablauf einer Datenübertragung mittels EnOcean Technologie ist in **Abbildung 2-1** dargestellt.

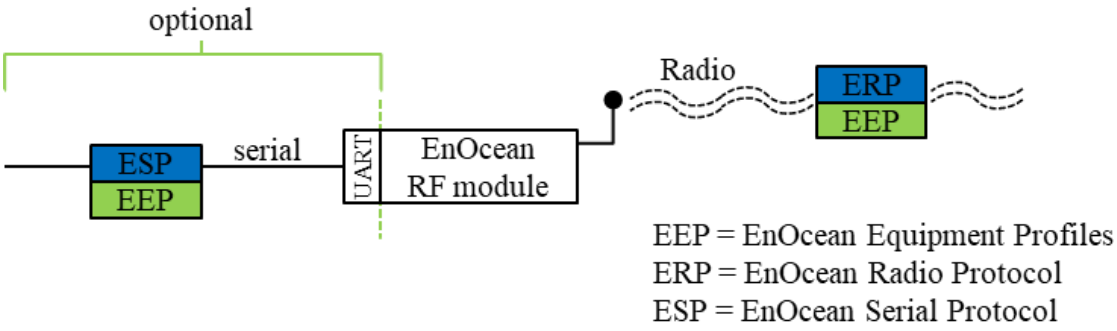


Abbildung 2-1: Übersicht der Datenübertragung mit ESP, EEP und ERP aus [ENA17]

Für die vorliegende Arbeit ist es ebenso wichtig, die Funktionsweise eines kapazitiven Sensors zu erläutern. Außerdem wird begründet, weshalb dieser für eine Erdfeuchtigkeitsmessung gut geeignet ist.

2.2 868 MHz Funkband

Das 868 MHz Frequenzband ist ein in Europa zugelassenes Frequenzband, welches unter gewissen Bedingungen lizenzfrei genutzt werden kann. Es gilt als Short Range Device Band und ist in dem Bereich von 868 MHz bis 870 MHz gültig. Innerhalb dieses Bereiches sind weitere Unterteilungen gemäß der ETSI EN 300 220-2 Norm (vgl. [ETS16]) geregelt. Das European Telecommunications Standards Institute (ETSI) unterteilt das gesamte Band in verschiedene Frequenzbereiche, in denen bestimmte Limitierungen gelten. In nachfolgender **Tabelle 2-1** sind die Teilfrequenzen und deren Limitierungen dargestellt.

Tabelle 2-1: Frequenzlimitierung verändert nach [ETS16]

Operational Frequency Band	Maximum effective radiated power, e.r.p.	Channel access and occupation rules (e.g. Duty cycle or LBT + AFA)	Maximum occupied bandwidth	Other usage restrictions
868,000 MHz to 868,600 MHz	25 mW e.r.p.	≤ 1 % duty cycle or polite spectrum access	The whole band except for audio & video applications limited to 300 kHz	-
868,700 MHz to 869,200 MHz	25 mW e.r.p.	≤ 0,1 % duty cycle or polite spectrum access	The whole sub-band except for audio & video applications limited to 300 kHz	-
869,400 MHz to 869,650 MHz	25 mW e.r.p.	≤ 0,1 % duty cycle or polite spectrum access	The whole band	-
869,400 MHz to 869,650 MHz	500 mW e.r.p.	≤ 10 % duty cycle or polite spectrum access	The whole band	-
869,700 MHz to 870,000 MHz	5 mW e.r.p.	No requirement	The whole band	Audio and video applications are excluded.
869,700 MHz to 870,000 MHz	25 mW e.r.p.	≤ 1 % duty cycle or polite spectrum access	The whole band	Analogue audio applications are excluded. Analogue video applications are excluded.

Wie in der Tabelle 2-1 zu sehen, ist eine maximale effektive Strahlungsenergie (maximum effective radiated power, kurz e.r.p.) von 5 mW, 25 mW bzw. 500 mW zugelassen. Des Weiteren ist eine maximale Kanalzugriffszeit (Duty cycle) von 0,1 % bis 10 % pro Stunde erlaubt. Nach [KUB12] ist auch ein häufigerer Kanalzugriff über Listen Before Talk (LBT) erlaubt.

Das LBT Prinzip, wie in [ETS04] beschrieben, sieht vor, dass der Kanal jederzeit genutzt werden kann, sofern dieser nicht von einem anderen Gerät blockiert wird.

Innerhalb der Frequenzbereiche 868,0 – 868,6 MHz und 868,7 – 869,2 MHz kann die gewählte Frequenz unter der Bedingung, dass die gesamte genutzte Bandbreite auch im Extremfall innerhalb von 300 kHz verbleibt, frei gewählt werden. Bei Audio- und Videoanwendungen ist keine Begrenzung von 300 kHz vorgesehen. Diese können eine größere Bandbreite innerhalb der Frequenzbereiche nutzen.

2.3 Serielle Kommunikation über UART

Der Universal Asynchronous Receiver and Transmitter (UART) dient dazu, eine serielle Kommunikation zwischen zwei Geräten zu ermöglichen. Dafür sind zwei Datenleitungen im bidirektionalen und eine Datenleitung im unidirektionalen Modus nötig. Im unidirektionalen Modus wird der Übertragungsanschluss (Tx) des Daten

sendenden Gerätes (UART 1) an den Empfangsanschluss (Rx) des Daten empfangenden Gerätes (UART 2) angeschlossen. Im bidirektionalen Modus wird zusätzlich eine Verbindung zwischen Tx des UART 2 und Rx des UART 1 hergestellt. Die beschriebenen Verbindungen sind in **Abbildung 2-2** dargestellt.

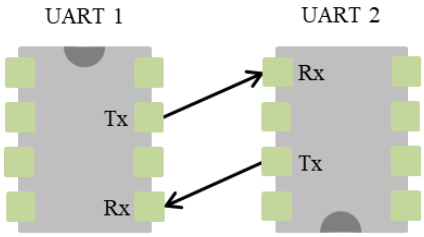


Abbildung 2-2: UART Verbindung zwischen zwei Geräten

Für den UART wird die Datenübertragungsgeschwindigkeit in einer Baudrate (Maßeinheit: Bd) angegeben. Diese gibt nicht direkt die Anzahl an Bits pro Sekunde an. Nach [ANO07] wird „[…] die Anzahl der Bits pro Zustand mit der Baudrate [multipliziert], [um] die Übertragungsgeschwindigkeit“ zu erhalten.

Der UART arbeitet mit den normalen TTL[1]-Pegeln (vgl. [ANO17]). Wie in [WÜS11] beschrieben, gelten bei 5 V Betriebsspannung bestimmte Pegel. Hierbei sind $\leq 0{,}8$ V (logisch 0) und $\geq 2{,}0$ V (logisch 1) die Spannungsunterschiede an dem Eingang. An dem Ausgang des UART werden $\leq 0{,}4$ V (logisch 0) und $\geq 2{,}4$ V (logisch 1) ausgegeben. Die Reserven, die sich daraus ergeben, sind dafür vorgesehen, dass Störungen von außen und bauteilbedingte Toleranzen kein Verfälschen der Übertragungsdaten bewirken. Die Zwischenzustände von Low zu High sind, wie in **Abbildung 2-3** zu sehen, verboten.

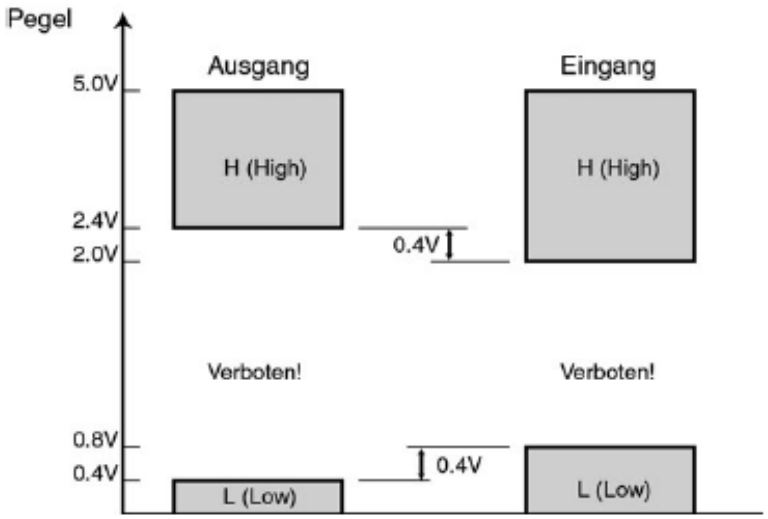


Abbildung 2-3: TTL-Spannungspegel aus [WÜS11]

Eine Kommunikation mit dem UART beinhaltet immer ein Startbit und mindestens ein Stoppbit. Des Weiteren gibt es die Möglichkeit, ein weiteres Stoppbit und ein Paritybit mitzusenden. Letzteres dient zur Überprüfung auf Bitfehler, die während der

[1] Transistor-Transistor-Logik

Übertragung entstanden sein könnten. Da in dieser Arbeit weder ein zusätzliches Stoppbit noch ein Paritybit eine Rolle spielen, wird dies nicht weiter betrachtet.

2.4 EnOcean Equipment Profile (EEP)

Die in diesem Abschnitt beschriebenen Informationen sind [ENA17] entnommen.

Generell dient das EnOcean Equipment Profile (EEP) zur Interpretation der empfangenen Daten. Die Kennzeichnung, um welches EEP es sich handelt, wird durch eine Teach-In Übertragung mitgeteilt. Das Empfangsgerät kann dann die Sendegerät-ID mit der zugehörigen (R)ORG[2], deren Funktion (FUNC) und des Typs (TYPE) abspeichern. Das nachfolgende Datentelegramm kann nun eindeutig interpretiert, verarbeitet und gegebenenfalls dargestellt werden. In **Abbildung 2-4** ist das 4 BS-Teach-In Telegramm dargestellt, welches die EEP Kennzeichnung beinhaltet.

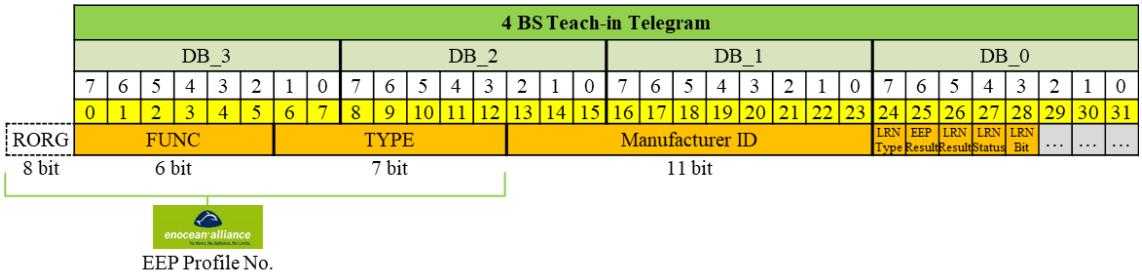


Abbildung 2-4: 4 BS-Teach-In Telegramm aus [ENA17]

Es existieren RPS-, 1 BS-, 4 BS- und VLD-Telegramme. Die Definition, um welches Telegramm es sich handelt, wird über die (R)ORG dargestellt. RPS-Telegramme dienen dazu, Informationen von z. B. Rocker-Switches, Position-Switches und Detektoren zu übertragen. 1 BS- Telegramme sind universelle Telegramme, die keine direkte Eigenschaft besitzen. In diesem Telegrammtyp wird nur ein Byte übertragen. Innerhalb dieses Bytes ist nur ein Bit zur Auswertung bestimmt. Das VLD-Telegramm hat eine variable Datenlänge und bietet viele Varianten zur Nutzung an. Ein Telegrammtyp, wie das 4 BS-Telegramm, überträgt eine feste Datenmenge von vier Byte. In der Liste der bereits existierenden EEPs sind in diesem Typ Sensoren, wie Temperatur- und Lichtsensoren, hinterlegt.

Im Verlauf dieser Arbeit wird nur der universale 4 BS Telegrammtyp genutzt, weshalb nicht weiter auf die anderen Telegrammtypen eingegangen wird. Die bereits existierenden EnOcean Equipment Profiles sind auf der Website von EnOcean-Alliance als EEP V 2.6.8 Spezifikation hinterlegt und erklärt (siehe [ENA17]).

Die Profile sind in einer XML-Datei abgelegt. Diese XML-Datei kann, wie in **Abbildung 2-5** zu sehen, um ein eigenes Profil erweitert und als C# oder JAVA Code ausgegeben werden. Bei Bedarf ist auch die Eintragung bei EnOcean-Alliance möglich. Die Voraussetzungen und weitere Informationen zum Erstellen und Eintragen eines neuen Profils sind ebenfalls auf der EnOcean-Alliance Website nachzulesen.

[2] ORG nur in Verbindung mit ESP2

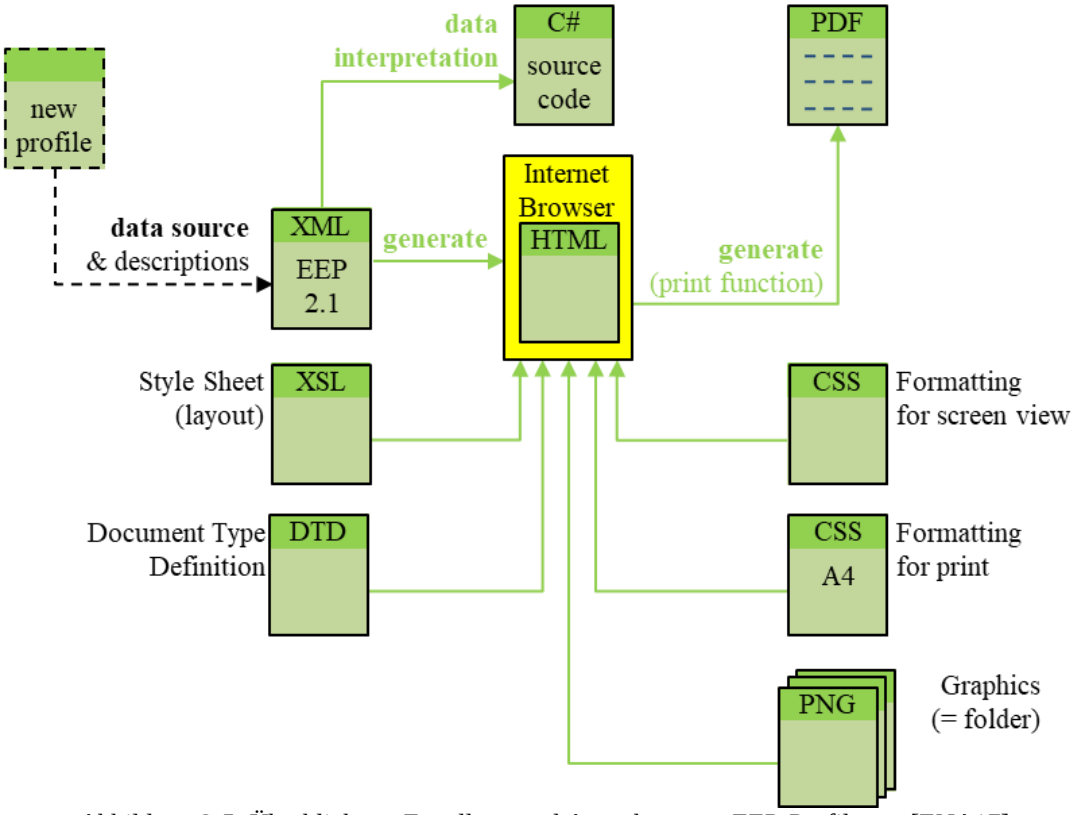


Abbildung 2-5: Überblick zur Erstellung und Ausgabe neuer EEP-Profile aus [ENA17]

2.5 EnOcean Serial Protocol (ESP)

Die in diesem Abschnitt dargestellten Informationen stammen aus [ENO18].

Das EnOcean Serial Protocol (ESP) definiert die serielle Kommunikation zwischen einem Host, dies kann ein PC oder ein externes Gerät (z. B. Mikrocontroller) sein, und einem EnOcean Modul. Die Kommunikation basiert auf RS-232 mit einer 3-wire Verbindung. Diese drei Leitungen sind Rx, Tx und GND. Ebenso ist es wichtig zu wissen, dass die Kommunikation mit einem Software-Handshake läuft und wenn nötig im Full-Duplex-Verfahren kommuniziert werden kann. Im Verlauf dieser Arbeit wird nur im unidirektionalen Modus kommuniziert und das Full-Duplex-Verfahren nicht benötigt. In **Abbildung 2-6** ist der beschriebene Sachverhalt verdeutlicht.

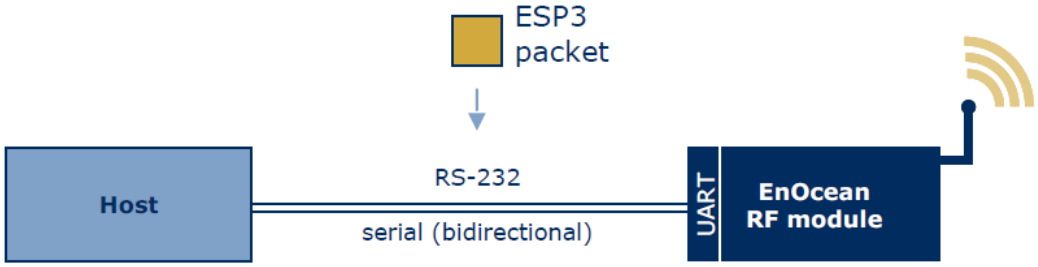


Abbildung 2-6: ESP Paket mit serieller Kommunikation aus [ENO18]

Das ESP3 verbessert die Funktionalität des ESP2. Es sind im ESP3 weitere zukunftssicherere Strukturen enthalten. In **Abbildung 2-7** sind die Unterschiede bzgl. ESP2 und ESP3 aufgeführt. Hervorzuheben sind besonders die mögliche Kommunikationsgeschwindigkeit und die Anzahl der maximal übertragbaren Bytes.

	ESP 2.0	ESP 3.0
Subtelegram count	--	•
Receive signal strength (RSSI)	--	•
Upward compatible with 'Optional Data'	--	•
Data verification	Checksum	CRC8
UART Synchronization (packet detection)	2 bytes	6 bytes
Max. number of ESP packet types	8	256
Types of data	Radio, Command	Any type of data
Max. size of transferred data	28 bytes	65535 bytes
Communication speed	9600 baud	**57600 baud** 115200 baud 230400 baud 460800 baud

Abbildung 2-7: Unterschiede ESP2 zu ESP3 aus [ENO18]

Die Begriffe aus Abbildung 2-7 sind folgendermaßen definiert:

- *Subtelegram count*: Anzahl der empfangenen Subtelegramme
- *Receive signal strength (RSSI)*: Übertragung der Funksignalstärke
- *Upward compatible with ‚Optional Data'*: Ohne Verletzung der Kompatibilität können optionale Daten angefügt werden.
- *Data verification*: Erhöhte Datensicherheit durch CRC8 Datenverifikation
- *UART Synchronization (packet detection)*: zuverlässigere Erkennung von ESP3 Datenpaketen bei seriellem Datenstrom
- *Max. number of ESP packet types*: Anzahl von unterschiedlichen Pakettypen, die übertragen werden können. Diese Pakettypen sind in [ENO18] unter Abschnitt „1.6.2 Packet types" aufgeführt.
- *Types of data*: Art der Daten, die übertragen werden können. Der Begriff Radio sagt aus, dass Daten von und zu einem EnOcean Radio Modul gesendet bzw. empfangen werden können. Bei dem Typ Command ist es möglich, einen Befehl an das EnOcean Modul zu senden, dies ist aber nicht umgekehrt möglich.
- *Max. size of transferred data*: Anzahl der maximal übertragbaren Bytes. Da das ESP2 nicht über die ‚Optional Data' Struktur verfügt, kann das Protokoll nur 28 Bytes übertragen. Bei ESP3 sind dadurch bis zu 65 535 Bytes möglich.
- *Communication speed*: Übertragungsgeschwindigkeit über UART Schnittstelle

Eine Kommunikation zwischen zwei Geräten, die einerseits auf ESP2 und andererseits auf ESP3 aufgebaut ist, ist nicht möglich. Die Datenstrukturen sind von verschiedener Art und müssen demnach unterschiedlich ausgewertet werden. Die Struktur der Pakete ist in **Abbildung 2-8** und **Abbildung 2-9** dargestellt.

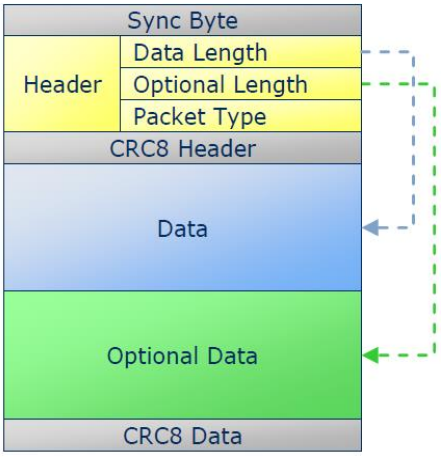


Abbildung 2-8: Aufbau ESP3 aus [ENO18]

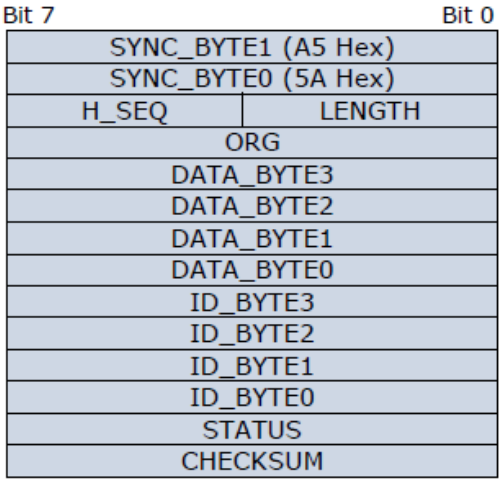


Abbildung 2-9: Aufbau ESP2 aus [ENO17e]

Das ESP3 hat im Vergleich zum ESP2 lediglich ein Synchronisationsbyte anstatt zwei. Das ESP3 besitzt einen Header, welcher mit zwei Bytes für die Datenlänge, einem Byte für die optionale Länge (Länge der optionalen Daten in Bytes) und einem Byte mit dem Pakettyp ausgestattet ist. Zur Headerkontrolle dient anschließend an den Header eine CRC8, die aus den Headerdaten gebildet wird. Die Länge der Daten und der optionalen Daten ergibt sich aus der im Header angegebenen Länge. Im ESP3 ist der Telegrammtyp mit RORG angegeben und Inhalt des Data Containers. Am Ende des ESP3 wird ein CRC8-Hashwert übertragen. Dieser wird über die Daten und optionalen Daten gebildet.

Im Vergleich hat das ESP2 einen Header, der aus einem Byte besteht. Dieser beinhaltet die Header Identifikation (H_SEQ) mit drei Bits und die Telegrammlänge (LENGTH) mit fünf Bits. LENGTH beschreibt die Anzahl der nach dem Header folgenden Bytes. Im ESP2 wird mit ORG, wie in Abschnitt 2.4 beschrieben, der Telegrammtyp angegeben. Als Data_Bytes werden Benutzerdaten, z. B. Messwerte oder die Auswertung der digitalen Inputs, bezeichnet. Die ID_Bytes geben die eindeutige ID des Sendegerätes an. Das Status Byte dient der Telegrammsteuerung. Dies wird erst genutzt, wenn das Telegramm durch einen Repeater erneut gesendet wird. Die Checksum des ESP2 wird über alle Bytes außer der Sync- und der Checksum-Bytes gebildet.

2.6 EnOcean Radio Protocol (ERP)

Die nachfolgenden Informationen zum EnOcean Radio Protocol 1 (ERP1) stammen aus [ENO13]. Zusätzlich existiert das ERP2, auf welches nicht näher eingegangen wird. [ENO17b] beinhaltet weitere Informationen über dieses Protokoll.

Das EnOcean Radio Protocol (ERP) beschreibt die Verfahren innerhalb der Sicherungsschicht (data link layer) und der Vermittlungsschicht (network layer) nach dem OSI-Modell. Das Kommunikationsprotokoll ist paketbasierend. Dabei sind zwei wesentliche Pakettypen zu nennen:

- Frame
- Subtelegram

Als Frame werden die codierten Daten bezeichnet, welche über die Bitübertragungsschicht (physical layer) bitweise gesendet werden. Ein Frame beinhaltet Kontroll- und Synchronisationsinformationen.

Ein Subtelegramm beinhaltet keine Kontroll- und Synchronisationsinformationen. Es wird in der Sicherungsschicht verarbeitet. In **Abbildung 2-10** ist der Aufbau eines Subtelegramms aus [ENO13] dargestellt.

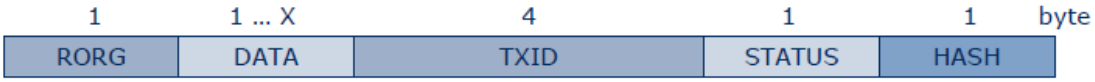


Abbildung 2-10: Struktur des Subtelegramms aus [ENO13]

Ein Subtelegramm besteht aus dem Identifikationsbyte RORG, einer je nach Telegrammtyp variablen Datenlänge, der vier Byte großen einzigartigen Transmitter-ID, einem Status Byte sowie einem Hash Byte. Das Status Byte gibt an, ob das Telegramm von einem Repeater gesendet wurde. In **Abbildung 2-11** sind die vier möglichen Optionen innerhalb des Status Byte dargestellt. Es ist vorgesehen, dass ein Telegramm maximal zweimal wiederholt wird.

Repeater level bits				Description
2^3	2^2	2^1	2^0	
0	0	0	0	Original sender
0	0	0	1	Subtelegram was repeated 1 time
0	0	1	0	Subtelegram was repeated 2 times
1	1	1	1	Telegram shall not be repeated

Abbildung 2-11: Optionen des Status Byte aus [ENO13]

Der Wert des Hash Byte wird über alle Bytes von RORG bis inklusive dem Status Byte ermittelt. Stimmt der übermittelte Hashwert nicht mit dem errechneten Hashwert der empfangenen Daten überein, wird das Subtelegramm verworfen. Es existieren zwei unterschiedliche Hash-Verfahren. Entweder kann ein Hashwert über eine 8-Bit Checksum oder ein 8-Bit CRC erzeugt werden. Im Status Byte wird das achte Bit mit einer 0 für Checksum und einer 1 für CRC beschrieben. Dadurch wird dem Empfänger mitgeteilt, um welches Hashverfahren es sich handelt.

Das EnOcean Radio Protocol unterstützt das Listen Before Talk Verfahren. Es wird bei der Nutzung von Funkübertragungen empfohlen. Bei Energy Harvesting Produkten, wie z. B. dem STM310/330, wird dieses Verfahren nicht unterstützt.

2.7 Funktionsweise von kapazitiven Sensoren

Dieser Sensortyp baut im Wesentlichen auf den kapazitiven Effekten auf. Wie in [HER18] beschrieben, existiert ein „Zusammenhang der Kapazität mit elektrischen und geometrischen Größen im Gleichstromfall […]“. Ein kapazitiver Sensor ist mit einem Plattenkondensator vergleichbar. Die Kapazität C ist abhängig von der elektrischen Feldkonstante ε_0, der Permittivitätszahl ε_r, der Plattenfläche A und dem Abstand d der beiden Platten. Der Zusammenhang dieser Größen ist in Formel (1) dargestellt:

$$\mathrm{C} = \varepsilon_0 \varepsilon_r \frac{A}{d} \;\; in\, Farad\; [F] \qquad (1)$$

Gemäß [HER18] kann die Kapazität durch folgende drei Punkte geändert werden:

- „Veränderung des Plattenabstandes d.“ Allerdings muss beachtet werden, dass kein linearer Zusammenhang zwischen Plattenabstand und Kapazität existiert.
- „Veränderung der aktiven Fläche A […]. Diese Beziehung zur Kapazität ist linear […].“
- „Veränderung des Dielektrikums, d.h. der Permittivitätszahl ε_r. Die Kapazität des Kondensators hängt davon linear ab […].“

Bei der Messung von Erdfeuchtigkeit ist vor allem die Permittivitätszahl von Wasser von Bedeutung. In **Tabelle 2-2** sind Beispiele für unterschiedliche Materialien und deren Permittivitätszahlen dargestellt. Trockene Erde hat eine im Vergleich zu Wasser sehr geringe Permittivitätszahl, was eine deutlich geringere Kapazität zur Folge hat. Außerdem beeinflussen Temperatur und Frequenz die Permittivitätszahl.

Tabelle 2-2: Permittivitätszahlen verschiedener Materialien verändert nach [HER18] und [HIP95]

Material	**Permittivitätszahl ε_r**
Vakuum	1
Wasser	81
feuchte Erde	29
trockene Erde	3,9

In **Abbildung 2-12** ist eine Veränderung des Dielektrikums zu erkennen. Ausgehend davon, dass ε_{r2} die Permittivitätszahl von Wasser und ε_{r1} die Permittivitätszahl von Erde ist, wird durch die Änderung des Wassergehaltes, dargestellt durch b in der Parallelschaltung, eine Änderung der Kapazität erreicht.

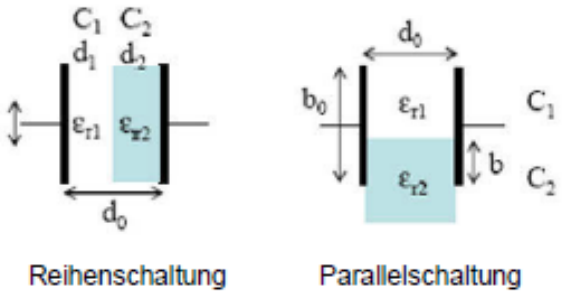


Abbildung 2-12: Veränderung des Dielektrikums aus [HER18]l

Der kapazitive Sensor ist dem zu messenden Material nicht direkt ausgesetzt. Die Messung findet indirekt statt. Der Sensor kann z. B. außerhalb eines Plastikgefäßes zur Füllstandsmessung angebracht werden. Generell gilt, dass zwischen dem Material und den Sensorflächen eine Isolierschicht vorhanden sein muss. Durch die indirekte Messung zur Bestimmung der Erdfeuchtigkeit wird die Pflanzenerde und somit die Pflanze nicht beeinträchtigt. Es entsteht keine Elektrolyse, was zur Vergiftung der Pflanzenerde führen würde.

3 Entwurf

3.1 Verwendete Programme und Tools

3.1.1 Dolphin View Basic

Zu Beginn der Entwicklung einer Demoapplikation wird Dolphin View Basic genutzt, um eine korrekte Datendarstellung sicherzustellen. Dolphin View Basic wird auf der EnOcean Website nach kostenloser Anmeldung als Download angeboten. Dieses Programm kann die durch ein EnOcean Modul gesendeten Daten, unter Nutzung vom USB300-Stick (EnOcean Produkt), empfangen und darstellen. Die daraus resultierenden, angezeigten Daten bilden die Grundlage, um eigene Auswerte- und Anzeigemechanismen zu entwickeln.

Um das EEP des Senders zu erhalten und die nachfolgenden Daten korrekt anzuzeigen, muss der Sender zu Beginn ein Teach-In Telegramm senden. Anschließend wird der Sender mit seiner ID und seinem Namen im Workspace- bzw. Unassigned-Slot mit dem genutzten EEP angezeigt (vgl. **Abbildung 3-1 a)**). Es ist auch möglich, den Sender ohne ein Teach-In Telegramm anzuzeigen. Dazu muss der Sender ein Datentelegramm senden, allerdings ist dann kein EEP als Auswertestrategie voreingestellt. Dies kann manuell im Programm nachgetragen werden (siehe **Abbildung 3-1 b)**). Somit ist ebenfalls die korrekte Auswertung der Daten garantiert.

Das Telegrammlog, wie in **Abbildung 3-1 c)** dargestellt, zeigt den kompletten Payload[3] in Hexadezimal an. Ebenso wird u. a. Datum/Uhrzeit, ID, RORG, Status und Signalstärke angezeigt.

Dolphin View Basic legt ein XML-Logfile an. In diesem werden Radio-Telegramme und Serial-Telegramme gespeichert.

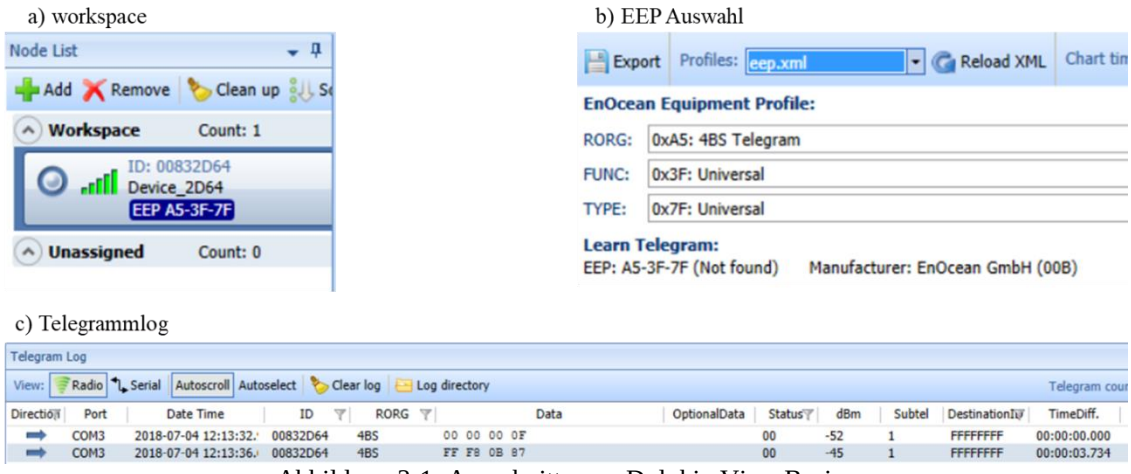


Abbildung 3-1: Ausschnitte von Dolphin View Basic

Die Ansicht der kompletten Dolphin View Basic Oberfläche ist in Anhang A angefügt.

[3] Messdaten des Senders (Data)

3.1.2 HTerm und Virtual Serial Port Driver

Der Virtual Serial Port Driver Version 9.0.567 von Eltima Software und HTerm Version 0.8.1 beta von Tobias Hammer werden hauptsächlich genutzt, um das User Interface in der Erstellungsphase mit Daten zu versorgen und zu testen. HTerm ist eine kostenlose Software und bedarf keiner Lizenz. Im Gegensatz dazu ist die Software Virtual Serial Port Driver ein kostenpflichtiges Programm, welches als 14-tägige Testversion vollständig genutzt werden kann.

Mit dem Virtual Serial Port Driver können virtuelle COM-Ports erstellt und gebrückt werden. Durch diese Maßnahme ist es möglich, einen Empfänger zu simulieren. Es werden auf einem virtuellen COM-Port durch HTerm Daten geschrieben und auf dem anderen virtuellen COM-Port, welcher als Empfänger-Port gilt, die Daten bereitgestellt. Dies simuliert den Dateneingang eines USB300-Stick für das User Interface. **Abbildung 3-2** zeigt die beispielhafte Darstellung der empfangenen Daten.

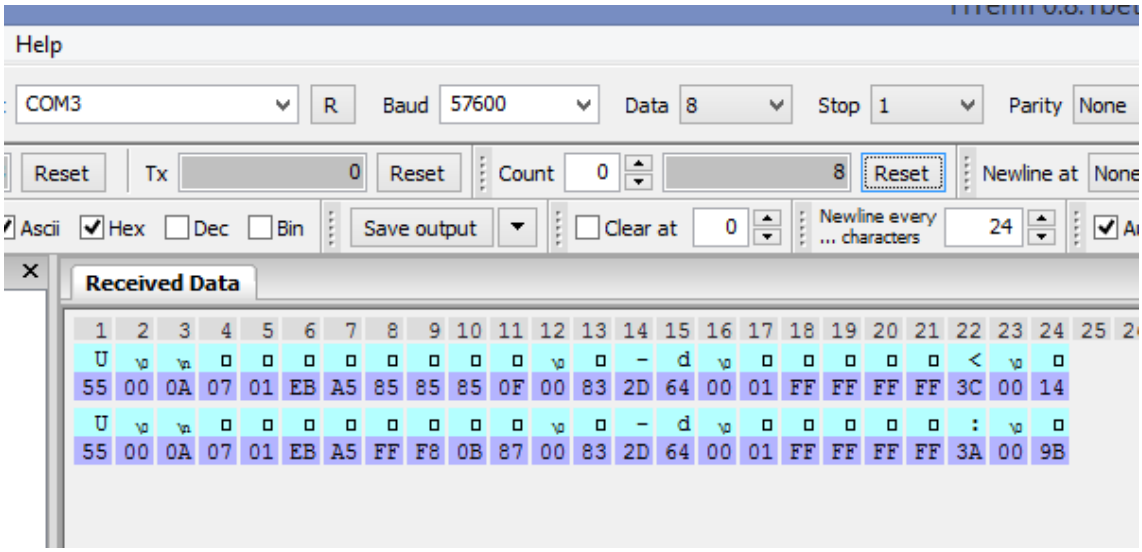


Abbildung 3-2: Beispieldarstellung empfangener Telegramme in HTerm über USB300-Stick

Über HTerm können Daten an einen COM-Port gesendet und ebenso von einem empfangen werden. Es ist möglich, Baudrate, Port und Datenmodus (z. B. 8N1) einzustellen. Die Einstellung, dass eine neue Zeile nach 24 Bytes beginnt, ergibt sich aus den verwendeten Kommunikationsprotokollen. Weitere Einstellungsmöglichkeiten wurden nicht berücksichtigt. Die empfangenen und gesendeten Daten können in ASCII, hexadezimal, dezimal oder binärer Form angezeigt werden.

3.1.3 Node.js und Node Package Manager (npm)

Node.js wird zur benutzerseitigen Darstellung von Daten gewählt, da im Praxissemester mit dieser Umgebung Erfahrungen gesammelt wurden. Node.js ist eine asynchrone, ereignisgesteuerte JavaScript-Laufzeitumgebung. Durch meist fertige Modulgruppen kann Node.js leicht erweitert werden. Die Nutzung von npm ist optional, wird aber empfohlen, da die Installation von den nötigen Erweiterungen für Node.js erleichtert

wird. Insbesondere sind hier die Erweiterungen socket.io, serve-static und connect zu nennen, welche während der Arbeit genutzt werden.

Die Nutzung von Node.js wird durch die Programmierung des serverseitigen und clientseitigen JavaScriptes gesteuert. Das serverseitige Script beinhaltet das Senden und Empfangen von Daten über den eingestellten COM-Port. Außerdem wird in diesem Script die Kommunikation zwischen dem Computer und dem externen Gerät eingestellt und durchgeführt. Der USB300-Stick belegt einen virtuellen COM-Port. Somit können Daten über diesen Port gesendet und empfangen werden. Des Weiteren kommuniziert das serverseitige mit dem clientseitigen JavaScript. Dieses dient zur Datenauswertung und Darstellung in einem Browser. Ebenso ist es damit möglich Daten vom Browser zu empfangen und an das serverseitige Script weiterzuleiten oder hinterlegte Befehle zu senden. Die Kommunikation der Client- und Serverseite findet über bestimmte Strings statt. Diese können frei gewählt werden.

3.1.4 Atmel Studio 7

Atmel Studio 7 wird zur Programmierung von einem Atmel ATmega8 genutzt. Es hat eine einfache Programmierumgebung und wird mit den nötigen Einstellungen für Atmel Mikrocontroller installiert. Diese Einstellungen werden nur noch beim Anlegen eines neuen Projektes ausgewählt. Ebenso existiert die Möglichkeit einen eigenen programmer, im Falle dieser Arbeit ein USBasp programmer von Baite, nach [MAN16] hinzuzufügen und einzustellen. Durch diese Maßnahme kann der ATmega8 mit nur einem Klick programmiert werden. Es ist nicht zwingend erforderlich Atmel Studio 7 zu nutzen. Die Programmierung kann ebenfalls durch den Einsatz der Konsole erfolgen.

3.2 Datenübertragungsweg

Ein Überblick des Datenübertragungsweges ist in **Abbildung 3-3** zu sehen.

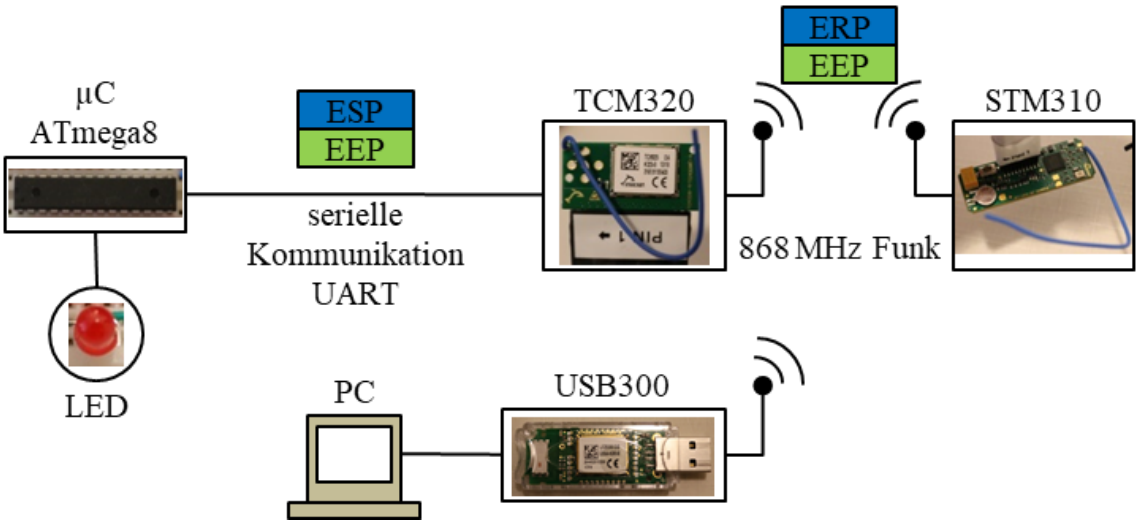


Abbildung 3-3: Datenübertragungsweg

Die Datenübertragung vom Sende- zum Empfangsmodul wird über das 868 MHz Funkband realisiert. Für die Funkübertragung wird das EnOcean Radio Protocol 1 (ERP1) genutzt. Die Funkmodule sind mit einer vorinstallierten Firmware ausgestattet. Das Empfangsmodul (Transceiver Module 320) leitet die vom Sendemodul (Scavanger Transmitter Module 310) empfangenen Daten über die serielle Schnittstelle weiter. Hierzu wird das EnOcean Serial Protocol 2 (ESP2) genutzt. Ein externer Mikrocontroller nimmt die Daten über die UART Schnittstelle auf, liest die Daten aus der ESP-Struktur aus und verarbeitet diese. Daraus resultierend soll eine LED beim Unterschreiten eines bestimmten Messwertes angeschaltet werden. Bei dem Überschreiten eines zweiten eingestellten Messwertes soll diese LED erlöschen. Die beiden Schwellwerte werden im Mikrocontroller hinterlegt.

Die Datendarstellung auf dem PC wird durch einen USB300-Stick erreicht. Darin ist nach [ENO17c] ein TCM310 verbaut. Außerdem ist zur Übertragung der Daten vom integrierten TCM auf den PC ein FTDI Chip verbaut, der die USB Kommunikation ermöglicht.

3.2.1 Auswahl des geeigneten Senders

Als Auswahl für ein Sendemodul standen ein STM310, STM330 und ein Pushbutton Transmitter Module (PTM) zur Verfügung. Die Informationen über diese Module wurden aus [ENO12], [ENO17d] und [ENO17a] entnommen.

Je nach Bauart des STM3xx verfügt dieser über Digital Inputs (DI) sowie Analog Digital Converter (ADC). Der PTM verfügt ausschließlich über DIs. Durch diese Erkenntnis wird der PTM ausgeschlossen, da eine Messung der Erdfeuchtigkeit über einen analogen Wert dargestellt wird.

Die Unterschiede zwischen einem STM310 und einem STM330 liegen darin, dass ein STM310 mit drei analogen und drei digitalen Eingängen universell einsetzbar ist. Ein STM330 verfügt nur über einen analogen Eingang, der als Sollwertregler des integrierten Temperatursensors eingesetzt wird. Voreingestellte EEPs sind bei beiden Modulen vorhanden und können u. U. über ein Programmierboard und Dolphin Studio geändert werden. Ausgewählt wird das STM310, da dieser über frei nutzbare analoge Inputs verfügt. Außerdem war dieser bereits am Lehrstuhl vorhanden.

Beim STM310 können die analogen Inputs mit einer eigenen elektronischen Beschaltung genutzt werden, sodass keine eigene Programmierung notwendig ist.

Für weitere Informationen über andere STM-Typen steht die EnOcean Website zur Verfügung.

3.2.2 Auswahl des geeigneten Empfängers

Bei der Auswahl eines geeigneten Empfängers fiel die Entscheidung auf einen TCM320. Vorteile bei einem TCM320 sind die vorinstallierte Antenne und die Pinout-

Steckerleiste. Das Modul muss nicht programmiert werden. Der Transceiver ist mit einer universalen Gateway Firmware ausgestattet, die es ermöglicht, mit dem EnOcean Netzwerk zu kommunizieren. Dieses Gateway ist ideal geeignet, um mit einem zweiten Mikrocontroller betrieben zu werden (vgl. [ENO17f]). Der TCM wird auf der Kommunikationsseite eingesetzt, auf der eine ständige Stromversorgung vorhanden ist und keine Energie eingespart werden muss.

3.3 Datenauswertung durch einen externen Mikrocontroller

Zur Datenanalyse wird ein ATmega8 von Atmel verwendet, da dieser bereits vorhanden war. Es muss nicht auf eine geringe Energieaufnahme geachtet werden, da dieser über eine dauerhafte Energieversorgung verfügt. Die Hauptkriterien, die ein externer Mikrocontroller erfüllen muss, sind eine UART-Schnittstelle, die eine Baudrate von 9 600 Bd verarbeiten kann, und mindestens ein digitaler Output Pin, um eine LED anzusteuern.

Wie in Abschnitt 3.2 beschrieben, wird das komplette Datentelegramm, welches vom TCM320 empfangen wird, über die serielle Kommunikation an den ATmega8 übertragen. Die Auswertung der mitgelieferten Daten erfolgt nach einer Kontrolle der Sende-ID des STM310. Die Sende-ID wird fest implementiert und kann nur durch erneutes Programmieren des ATmega8 geändert werden. Der Abgleich der empfangenen Daten mit den Schwellwerten soll innerhalb des Mikrocontrollers stattfinden. Die Schwellwerte werden fest im Programmcode verankert. Diese sind zum einen für die Aktivierung der LED bei einem Minimalwert und zum anderen für die Deaktivierung der LED bei einem Maximalwert zuständig. Die LED kann stellvertretend für eine Wasserpumpe stehen. Somit kann simuliert werden, dass eine Wasserpumpe bei einem Minimalwert anläuft und bei einem Maximalwert abschaltet.

3.4 Auswahl des Erdfeuchtesensors

Ein geeigneter Erdfeuchtesensor darf die Erde und somit die Pflanzen nicht belasten. Ausgewählt wird ein kapazitiver Erdfeuchtesensor. Bei der Recherche nach einem passenden Sensor für das STM310 waren vor allem die Spannung und der maximal nutzbare Strom von Bedeutung.

Wie in [ENO12] beschrieben kann das STM310 eine maximale Stromstärke von 5 mA bei einer Spannung von 1,8 V bereitstellen. Dadurch war die Suche nach einem Sensor, der diese Kriterien erfüllt, erschwert. Ebenso ist bei der Auswahl eines Sensors zu beachten, dass die im STM310 hinterlegte Anlaufzeit (standardmäßig 2 ms) nicht überschritten wird. Die Anlaufzeit ist über Dolphin Studio in einem Bereich von 0 ms bis 508 ms mit einer Schrittweite von 2 ms konfigurierbar. Die Messwerte müssen vom Sensor in einem Spannungsbereich von 0,1 V bis 1,7 V als Analogsignal ausgegeben werden. Da das STM310 zur Aufnahme von Messwerten nur über ADCs verfügt, sind Sensortypen, die per I²C oder SPI kommunizieren, nicht zu berücksichtigen. Ein passender Sensor, der die genannten Voraussetzungen aufweist, wurde nicht gefunden.

Letztendlich ist die Wahl auf einen kapazitiven Erdfeuchtesensor der Firma DFRobot gefallen (siehe **Abbildung 3-4**).

Abbildung 3-4: Capacitive Soil Moisture Sensor der Firma DFRobot

Dieser Sensor ist nach [DFR17] für einen Arduino konzipiert. Um damit die Erdfeuchtigkeit zu messen, ist eine externe Spannungsquelle bereitzustellen. Diese wird durch eine 5 V USB-Powerbank realisiert. Der Sensor soll je nach Wassergehalt der Erde eine Ausgangspannung von 0 V bis 3 V liefern. Eine zusätzliche elektronische Schaltung ist erforderlich, um diesen Sensor in Verbindung mit dem STM310 einzusetzen.

Die Funktionsweise des Sensortyps ist als Schema in **Abbildung 3-5** dargestellt. Eine Änderung des Dielektrikums ε_r (vgl. Formel (1) aus Abschnitt 2.7) hat eine proportionale Änderung der Kapazität zur Folge.

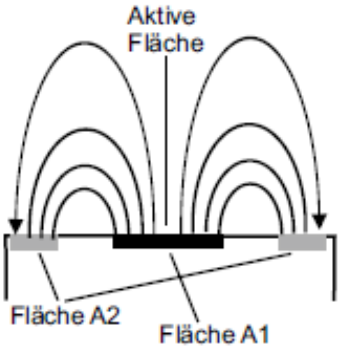


Abbildung 3-5: Darstellung offener Kondensator aus [HER18]

3.5 Darstellung der Messwerte

Die Darstellung der Messwerte für den Benutzer findet in einem Browser statt. Hierzu wird ein USB300-Stick von EnOcean benötigt. Die vom USB300 empfangenen Daten werden über den integrierten FTDI Chip auf den USB-Port weitergeleitet. Das serverseitige JavaScript erhält die Daten über den virtuellen COM-Port. Diese erhaltenen Daten werden auf ihre Länge überprüft. Anschließend werden die Sender-ID und die Messdaten herausgefiltert. Wenn die Sender-ID mit der durch den User eingestellten ID übereinstimmt, werden die Messdaten an das clientseitige JavaScript gesendet.

Im Browser wird durch ein Drop-Down Menü der anzuzeigende Sender eingestellt. Dementsprechend werden nur Messdaten von dem ausgewählten Sender angezeigt. Es erfolgt die Darstellung von Maximalwert, Minimalwert, Durchschnittswert und letztem

gemessenen Wert. Ebenso soll eine grafische Darstellung in Form eines Messwertverlaufs angezeigt werden. Die Anzahl der darzustellenden Werte wird in dem clientseitigen JavaScript implementiert. Eine Veränderung der maximal angezeigten Messwerte ist durch die Anpassung des clientseitigen JavaScripts möglich.

Die Ausführung des browserbasierten User Interface wird durch eine Batch-Datei gestartet. Bevor die Messwerte angezeigt werden, muss der Sender dem User Interface seine ID, über ein Teach-In Telegramm, mitteilen. Hierzu soll ein „Hinzufügen“- Button im Browser betätigt werden. Anschließend soll der Teach-In Button des STM310 gedrückt werden und der Sender wird im Browser hinzugefügt. Ist der anzuzeigende Sender ausgewählt, kann über den „Start“-Button die Darstellung der Messwerte gestartet werden. Die neuesten (bis zu 500) eingehenden Messwerte werden nun im Browser angezeigt. Eine Speicherung der Daten ist nicht vorgesehen. Mit der Betätigung des „Stop“-Button werden keine neuen Messwerte aufgenommen, aber die vorhandenen weiterhin angezeigt. Es ist nun möglich einen weiteren Sender hinzuzufügen oder einen bereits hinzugefügten Sender auszuwählen. Das Löschen der angezeigten Daten wird über einen „Reset“-Button realisiert.

4 Umsetzung der Demoapplikation

4.1 Aufbau der Hardware

4.1.1 Benötigte Komponenten

Wie in Kapitel 3 erwähnt sind zur Umsetzung der Demoapplikation folgende Bauteile nötig:

- STM310 (dient der Messung über einen ADC und dem Senden der Messdaten per Funk),
- TCM320 (dient dem Empfang von Funktelegrammen und der Weiterleitung per serieller Kommunikation über die integrierte UART Schnittstelle) und
- ATmega8 (dient der Datenauswertung und Aktivierung einer LED)

Des Weiteren werden folgende elektronische Komponenten benötigt:

- 11,0592 MHz Quarz (als Takterzeugung für den ATmega8),
- Spannungsregler auf 3,3 V (genutzt wird ein LD1117V33C von STMicroelectronics),
- Kondensatoren: 10 µF, 10 nF und 100 nF,
- eine LED,
- Widerstände: zweimal 10 kΩ sowie einmal 1 kΩ,
- zweimal NO-Taster und
- 10 kΩ Potentiometer

Zur Energieversorgung des Empfängers sowie des externen Mikrocontrollers wird ein Netzteil mit einer Spannungsversorgung von 5 V genutzt. Die Energieeinspeisung für den kapazitiven Sensor ist durch eine USB-Powerbank realisiert.

4.1.2 Verdrahtung des Aufbaus

In **Abbildung 4-1** ist die Verdrahtung der empfängerseitigen Schaltung erkennbar. Dargestellt ist die Beschaltung im TCM Mode 0. Zusätzlich ist die Anpassung für den TCM Mode 1 (blau dargestellt) ersichtlich. Die unterschiedlichen Modi sind in Abschnitt 4.3 erklärt. Im Mode 0 ist nur die Verbindung von ADIO7 am TCM und PD0 am ATmega8 herzustellen, da auf TCM-Seite nur übertragen und auf Mikrocontroller-Seite nur empfangen wird. Bei der Nutzung von Mode 1 muss zusätzlich eine Verbindung von ADIO6 zu PD1 hergestellt werden. Der Reset-Pin des TCM wird über einen 10 kΩ Widerstand (R3) an dessen Betriebsspannung (VDD) angelegt. Parallel dazu ist nach [ENO17e] ein 10 nF Kondensator (C1) empfohlen und installiert. Um die Spannungsversorgung von 3,3 V für den TCM herzustellen, ist ein Spannungswandler notwendig. Nach [STM13] ist ein 100 nF Kondensator (C2) zwischen Input und GND sowie ein 10 µF Kondensator (C3) zwischen Output und GND zu installieren. Der ATmega8 wird nach [ATM13] mit einer Betriebsspannung von 4,5 - 5,5 V betrieben.

Mit einer durch das Netzteil generierten Spannung von 5 V ist der Betrieb somit gewährleistet.

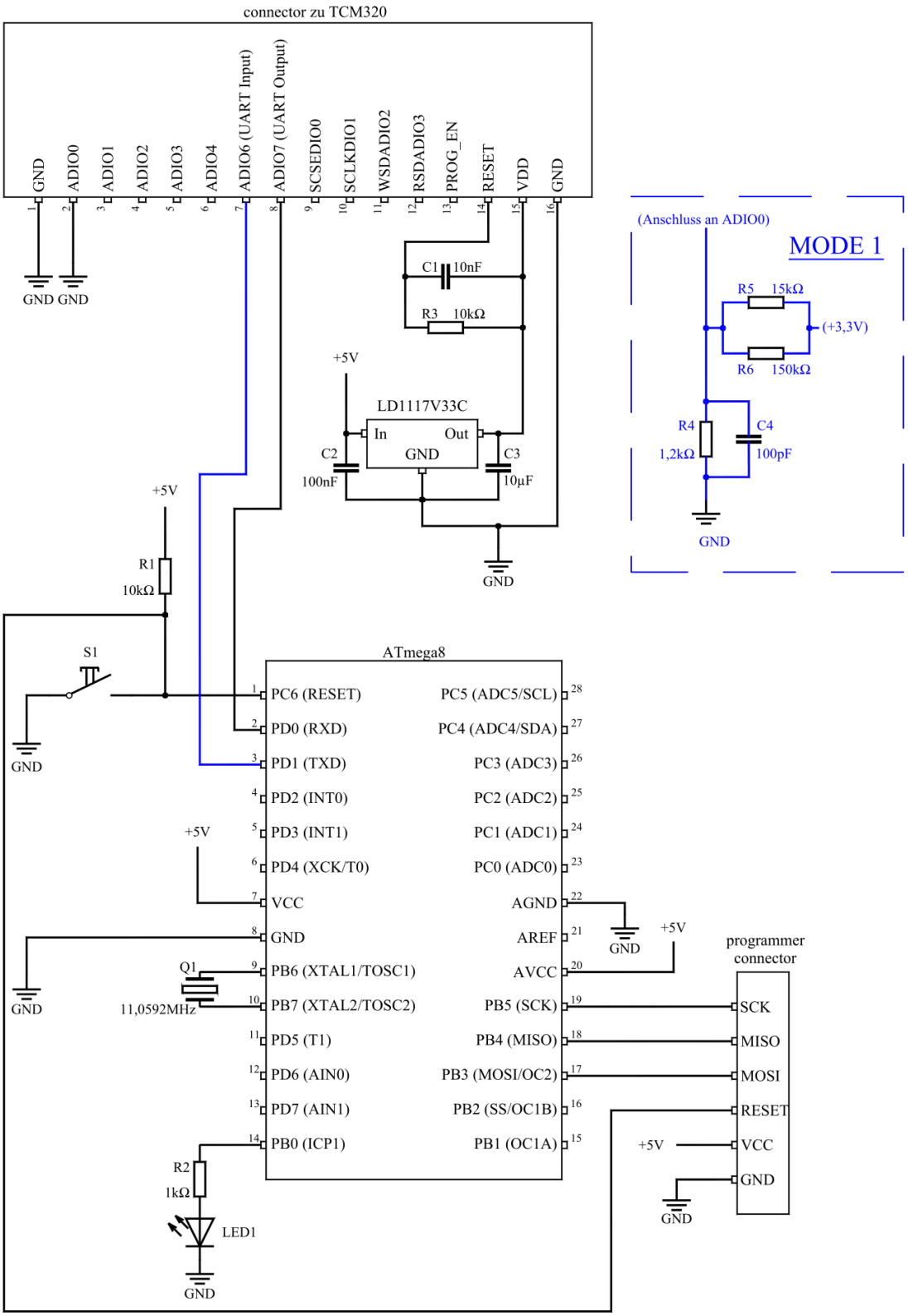


Abbildung 4-1: Beschaltung Empfängerseite TCM320 / ATmega8

Der programmer connector ist die Verbindungsschnittstelle zwischen Mikrocontroller und dem verwendeten USBasp programmer. Diese Verbindung ist nur bei einer Neuprogrammierung des Mikrocontrollers nötig. Ein Neustart des Mikrocontrollers kann durch den installierten Taster S1 erfolgen, welcher den PC6-Pin (RESET) des ATmega8 auf GND legt. Die Vorwiderstände für die LED (R2) und den RESET-Pin (R1) sind nach [BEC14] ausgewählt.

Der Schaltungsaufbau zu Testzwecken in Verbindung mit dem STM310 ist in **Abbildung 4-2** dargestellt. Das STM310 verfügt über drei ADC Inputs. Somit wäre es möglich drei unterschiedliche Messwerte von verschiedenen Sensoren aufzunehmen. Bei der Nutzung von nur einem Sensor werden dennoch alle ADC Inputs genutzt, um einen undefinierten Zustand zu vermeiden und daraus resultierend eine Telegrammübertragung, ohne nennenswerte Messwertveränderung, zu unterbinden.

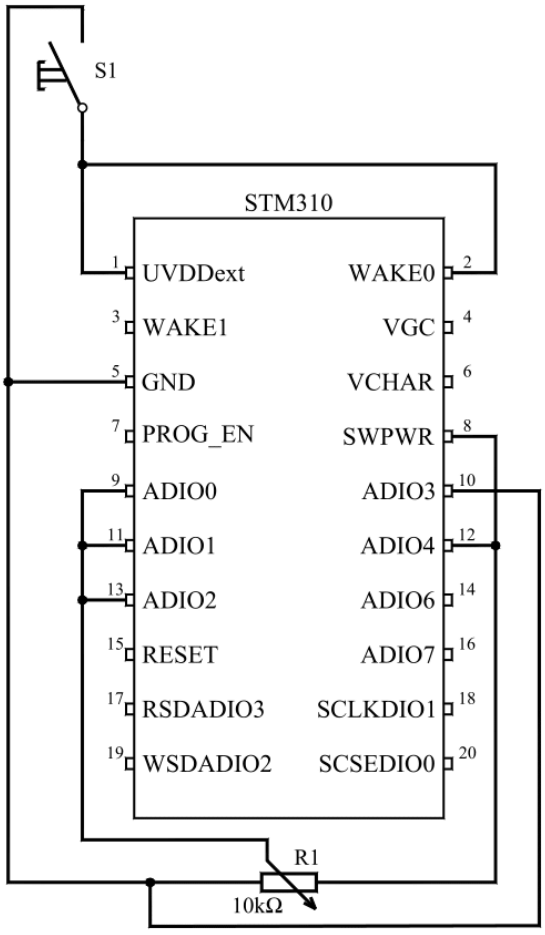


Abbildung 4-2: Beschaltung Senderseite STM310 mit Potentiometer als Sensor

Der Taster S1 ist dafür vorgesehen, dass manuell per Tastendruck eine Funkübertragung stattfindet. Die Verbindung von Pin 1 (UVDDext) und Pin 2 (WAKE0) dient dem Einstellen der WAKE-Konfiguration. Pin 2 muss sich nach [ENO12] immer in einem definierten Zustand befinden. Der Taster legt Pin 2 kurzfristig auf GND, sodass ein Strom von 1 µA fließt. Da intern ein 1,8 MΩ Widerstand verbaut ist, wird der Strom somit reguliert. UVDDext dient nicht zur Versorgung von externen Schaltungen. Das Potentiometer R1 dient als Beispiel für einen angeschlossenen Sensor.

4.2 Scavanger Transmitter Module 310 (STM310)

4.2.1 Allgemeines und Typenbeschreibung

Ein Scavanger Transmitter Module dient der Übertragung von Messdaten mit Hilfe des 868 MHz Funkbandes. Die verschiedenen STM-Versionen beinhalten nicht nur einen anderen Hardwareaufbau, sondern, laut einem Telefonat mit Herrn Christian Bach von EnOcean, auch eine andere Firmware. Dies konnte nicht widerlegt oder bestätigt werden.

Das STM310 ist eine veraltete Ausführung, die von EnOcean nicht mehr unterstützt oder verkauft wird. Diese Art besitzt bereits eine installierte Solarzelle, einen Energiespeicher und eine Whip-Antenne. Das universelle A5-3F-7F EEP ist vorprogrammiert und gibt keine Auskunft darüber, um welche Messwerte es sich handelt. Durch das genutzte EEP müssen die gemessenen Daten demnach selbst interpretiert werden. Dies erfolgt in dem externen Mikrocontroller. Da das STM310 über keine Verschlüsselungstechnik verfügt, werden Daten im Klartext übertragen. Dies ist bei sicherheitskritischen Daten zu beachten. Die STM320/329/330/331 können hingegen mit AES128 verschlüsseln.

Eine ähnliche Variante zum STM310 ist das STM300. Die freien ADCs und DIs können ebenso wie beim STM310 genutzt werden. Die Spannungsversorgung und die Antenne müssen jedoch extra installiert werden.

4.2.2 Energieverbrauch

Der Energiebedarf mit diesem Modul ist sehr gering. Je nach Einstellung zwischen Wake-cycle und Transmit Intervall kann die Betriebszeit erheblich verlängert werden.

In **Abbildung 4-3** ist der Energiebedarf für das STM310 ohne externe Beschaltung zu erkennen. Bei einem Wake-cycle von einer Sekunde weist der Sender nur eine geringe Betriebszeit auf. Je nach Sonneneinstrahlung auf die Solarzelle kann die Betriebszeit verlängert werden. Eine gute Einstellung ist ein Wake-cycle von 100 s. Bei diesem Zyklus ist ein Übertragungsintervall von einer Übertragung pro Wake-Up eine gute Option, um den Sender dauerhaft zu betreiben. Bei diesen Einstellungen benötigt das STM310 lediglich 7,8 h bei einer Beleuchtungsstärke von 200 lx, um einen Betrieb über 24 h zu gewährleisten. Außerdem ist die geringe Stromaufnahme von durchschnittlich 1,6 µA zu erkennen.

Wenn die Speicherkapazität erhöht wird (ggf. in Verbindung mit einer weiteren oder größeren Solarzelle), können auch geringere Wake-cycle eingestellt und der 24 h Betrieb gewährleistet werden. Im Beispiel der Erdfeuchtigkeitsmessung ist dies nicht notwendig. Der Grund hierfür ist, dass eine Messung der Erdfeuchtigkeit nicht so oft erfolgen muss. Dennoch könnte es erforderlich sein, die Kapazität und die Energieeinspeisung in Form von einer zusätzlichen Solarzelle zu erhöhen. Dies sollte in Betracht gezogen werden, wenn externe Sensorelemente installiert werden, welche unter Umständen einen höheren Energiebedarf besitzen.

Wake cycle [s]	Transmit interval	Operation Time in darkness [h] when storage fully charged	Required reload time [h] at 200 lux within 24 h for continuous operation	24 h operation after 6 h illumination at x lux	Illumination level in lux for continuous operation	Current in µA required for continuous operation
1	1	0.5	storage too small	storage too small	5220	130.5
1	10	1.7	storage too small	storage too small	1620	40.5
1	100	2.1	storage too small	storage too small	1250	31.3
10	1	5.1	storage too small	storage too small	540	13.5
10	10	16	21	storage too small	175	4.4
10	100	20	16.8	storage too small	140	3.5
100	1	43	7.8	260	65	1.6
100	10	98	3.6	120	30	0.8
100	100	112	3	100	25	0.6

Abbildung 4-3: Betriebszeit STM310 aus [ENO12]

4.2.3 Steuereinheit des STM310

Die Steuereinheit des STM310 trägt die Bezeichnung „Dolphin EO3000I“. Diese beinhaltet das Powermanagement, einen RF Transceiver und einen Mikrocontroller. In **Abbildung 4-4** ist eine detaillierte Ansicht des EO3000I dargestellt.

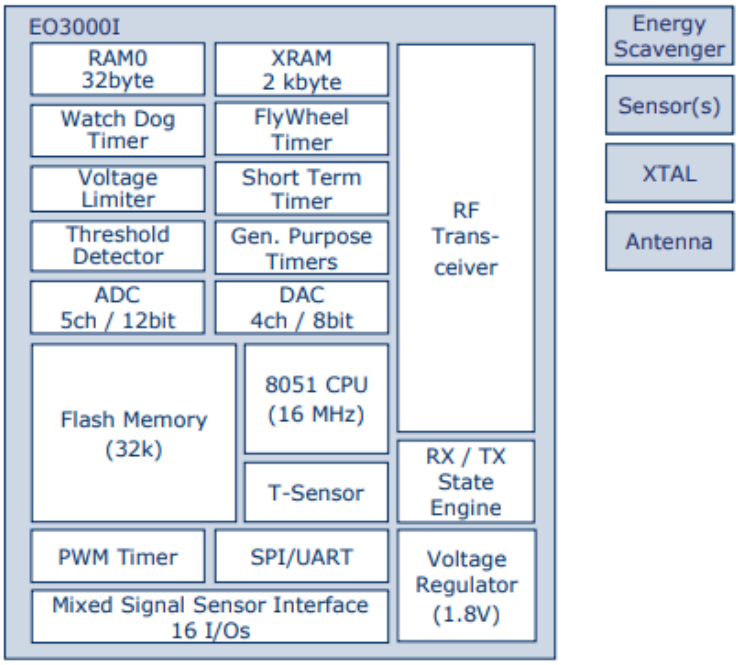


Abbildung 4-4: Blockdiagramm EO3000I aus [ENO14]

Bei der Entwicklung einer eigenen Firmware für das STM310 oder andere STM-Produkte ist es wichtig zu erwähnen, dass eine 8051 CPU verbaut ist. Weiteres dazu ist im nachfolgenden Abschnitt 4.2.4 erläutert.

4.2.4 Konfigurationsverfahren des STM310

Es existieren drei Möglichkeiten das STM310 zu konfigurieren. Diese Konfigurationsverfahren sind:

- Konfiguration über Pins,
- Konfiguration über das Serial Interface und
- Konfiguration durch eigene Firmware

Die Einstellungsmöglichkeiten nehmen absteigend zu.

Konfiguration über Pins

Die Konfiguration über Pins lässt lediglich Einstellungen des Wake-cycle zu. Dies ist für den Energieverbrauch sowie die Häufigkeit der Datenübertragung wichtig. In **Abbildung 4-5** sind die vier zugehörigen Einstellungsmöglichkeiten dargestellt.

CONF_0	CONF_1	Wake-up cycle time	Number of wake-ups that trigger a redundant retransmission
NC	NC	100 s ±20%	Every 7th - 14th timer wake-up signal, affected at random
NC	GND	10 s ±20%	Every 70th - 140th timer wake-up signal, affected at random
GND	NC	100 s ±20%	Every timer wake-up signal
GND	GND	No cyclic wake-up	No redundant retransmission

Abbildung 4-5: Konfigurationsmöglichkeiten zur erneuten Übertragung von redundanten Daten aus [ENO12]

Über die Verbindung der Pins CONF_0 und CONF_1 mit GND können die in Abbildung 4-5 dargestellten Optionen eingestellt werden. Das offene Ende eines Pins wird mit Not Connected (NC) beschrieben. Das Transmit-Intervall kann nicht über diese Einstellung geändert werden.

Bei einem Test wurde nach jeder Funkübertragung die Eingangsspannung am ADC durch ein Potentiometer geändert. Dies ergab einen Telegrammempfang aller 80 – 120 s, was vermuten lässt, dass das STM310 standardmäßig eine Übertragung mit jedem Wake-Up durchführt. In [ENO12] konnte diesbezüglich keine Aussage gefunden werden. Bei gleichbleibender Eingangsspannung am ADC-Input werden die redundanten Daten nach jedem 7. bis 14. Wake-Up gesendet, was einer Zeitspanne von 560 s bis 1 680 s entspricht. Die Anzahl der erforderlichen Wake-Ups, bevor ein Telegramm gesendet wird, wird nach jedem Senden durch eine Zufallszahl neu generiert. Somit kann nicht eindeutig vorhergesagt werden, wann das nächste Telegramm mit redundanten Daten gesendet wird.

Es wird ein Wake-cycle von 100 s eingestellt, da eine Messung von Erdfeuchtigkeit nicht alle 10 s stattfinden muss. Außerdem ist ein erneutes Senden von redundanten Daten nach dem 7. bis 14. Wake-Up ausreichend. Somit sind die Pins CONF_0 und CONF_1 nicht anzuschließen.

Bei zeitlich relevanten Messungen, wo es unter Umständen wichtig sein kann, auch redundante Daten zu senden, ist CONF_0 auf GND zu legen. Im Gegensatz dazu können redundante Daten auch komplett unterdrückt werden, was eine Energieeinsparung durch das Nichtsenden eines Telegramms zur Folge hat. Hierzu sind beide CONF_X Pins auf GND zu legen. Eine Änderung dieser Einstellung wird erst nach dem nächsten Wake-Up registriert und umgesetzt.

Konfiguration über Serial Interface

Erforderliche Komponenten zum Einstellen über das Serial Interface sind ein USB/SPI programmer und Dolphin Studio, wie in **Abbildung 4-6** zu sehen. Als USB/SPI programmer kann von EnOcean das EOP350 verwendet werden.

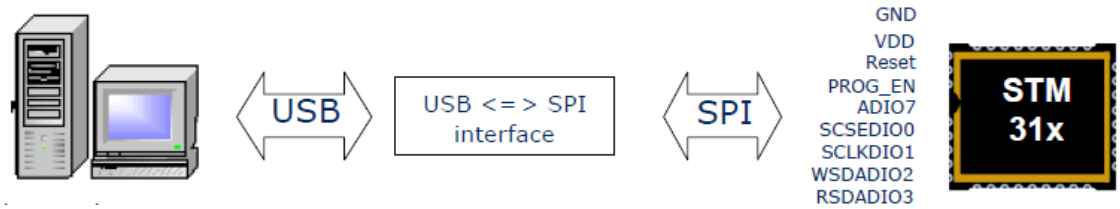


Abbildung 4-6: Programmierung über Serial Interface aus [ENO12]

Die Installation von Windows XP oder einer nachfolgenden Version ist nötig, da Dolphin Studio nur als Windows Version zur Verfügung steht. Neue Einstellungen werden erst nach einem RESET oder Power-On Reset übernommen. Mögliche Änderungen durch diese Methode sind nach [ENO12]: Wake-cycle, redundant retransmission cycle, threshold values für analoge Inputs, ADC Auflösung, input mask (um Veränderungen an den digitalen Input Pins ggf. zu ignorieren), Zeitverzögerung zwischen der Aktivierung des SWPWR-Pins und dem Messvorgang des ADC und vor allem Änderungen an der Manufacturer ID und dem EEP.

Weitere Informationen können hier nicht gegeben werden, da nicht mit dem Dolphin Studio gearbeitet wurde.

Konfiguration durch eigene Firmware

Bei der Entwicklung einer eigenen Firmware wird, nach Aussage von Herrn Christian Bach von EnOcean, ein kommerzieller C-Compiler von Keil benötigt. Dieser Compiler hat einen Preis von ca. 8 000 Euro. Außer dem Compiler ist die Dolphin API von der EnOcean Website herunterzuladen und, wie bei der Konfiguration via Serial Interface, ein USB/SPI programmer notwendig. Die Entwicklung einer eigenen Firmware hat den Vorteil, dass sämtliche auf dem EO3000I befindlichen Komponenten genutzt werden können. Vorausgesetzt der reibungslose Ablauf zur Telegrammerzeugung und das Energiemanagement werden dabei nicht beeinträchtigt. Diese Möglichkeit wurde nicht weiter betrachtet.

Ist ein Programmierboard (EOP350) oder ein USB/SPI programmer vorhanden, können auch bereits fertige Firmware Files von der EnOcean Website heruntergeladen werden. Demnach können auch Module, die vorher einen anderen Sensor betrieben und ein anderes EEP genutzt haben, für eine neue Aufgabe programmiert werden.

4.2.5 Daten- und Teach-In Telegramm

Die zwei Telegrammtypen sind in ihrer Datenmenge identisch, jedoch anders strukturiert und auszuwerten. Ein Teach-In Telegramm dient der Erkennung eines Moduls mit dessen EEP. Der Aufbau eines 4 BS Teach-In Telegramms ist in **Abbildung 4-7** zu sehen.

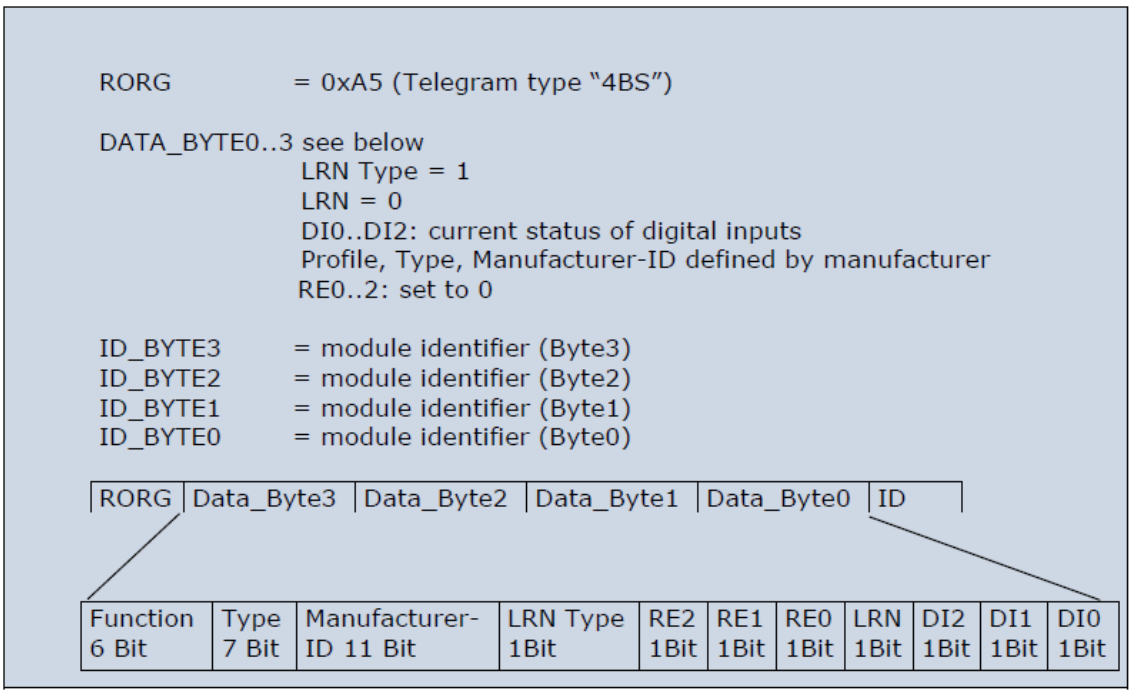


Abbildung 4-7: Aufbau eines Teach-In Telegramms aus [ENO12]l

Innerhalb eines Teach-In Telegramms werden außer dem EEP auch die Manufacturer-ID und die Geräte-ID gesendet. Wie in Abschnitt 2.4 beschrieben, sind RORG, Function und Type für die Definition des EEP zuständig. Die Manufacturer-ID gibt den Gerätehersteller an.

Um zu erkennen, dass es sich um ein Teach-In Telegramm handelt, wird das LRN-Bit auf 0 gesetzt. Der LRN-Type gibt an, ob ein Telegramm mit EEP und Manufacturer-ID gesendet wird. In welchem Zusammenhang dies von Nutzen ist, konnte nicht geklärt werden.

Der Aufbau des verwendeten 4 BS Datentelegramms unterscheidet sich in den vier Datenbytes gegenüber dem Teach-In Telegramm. Es wird weder ein EEP noch die Hersteller-ID übertragen. Die vier Datenbytes werden von RORG und der Geräte-ID eingerahmt. In **Abbildung 4-8** ist der Aufbau der vier Datenbytes ersichtlich.

ORG = 0x07 (Telegram type "4BS")

Data_Byte1..3
3x8bit mode:

DATA_BYTE3	= Value of AD_2 analog input
DATA_BYTE2	= Value of AD_1 analog input
DATA_BYTE1	= Value of AD_0 analog input

1x8bit, 1x6it, 1x10bit mode:

DATA_BYTE3	= Value of AD_2
DATA_BYTE2	= Upper 2 bits of AD_0 and value of AD_1
DATA_BYTE1	= Lower 8 bits Value of AD_0 analog input

DATA_BYTE3								DATA_BYTE2								DATA_BYTE1							
AD_2								AD_1						AD_0									
7	6	5	4	3	2	1	0	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0

DATA_BYTE0 = Digital sensor inputs as follows:

Bit 7				Bit 0
Reserved, set to 0	1	DI_2	DI_1	DI_0

ID_BYTE3	= module identifier (Byte3)
ID_BYTE2	= module identifier (Byte2)
ID_BYTE1	= module identifier (Byte1)
ID_BYTE0	= module identifier (Byte0)

Abbildung 4-8: Aufbau der Datenbytes von einem Datentelegramm aus [ENO12]

In der verwendeten Firmware sind die drei ADCs jeweils mit einer Standardauflösung von acht Bit eingestellt. Demnach ist pro ADC ein Datenbyte vorhanden. Das Data_Byte0 beinhaltet lediglich die drei digitalen Inputs mit je einem Bit. Die restlichen Bits sind mit 0 bzw. 1 festgelegt (vgl. Abbildung 4-8). Bei der Auswertung ist zu beachten, dass die ADC-Werte an der korrekten Stelle ausgelesen werden, je nachdem welcher ADC Verwendung findet und in welcher Auflösung dieser betrieben wird.

Der Aufbau von Daten- und Teach-In Telegramm sollte bei den anderen Telegrammtypen wie 1 BS, VLD und RPS in den jeweiligen Moduldatenblättern und/oder im EEP Datenblatt nachgelesen werden.

4.2.6 Analog Digital Converter (ADC)

Der ADC, der auf dem STM310 Verwendung findet, kann nicht Rail-to-Rail messen. Rail-to-Rail bedeutet, dass eine Messung nicht komplett von der unteren Grenze (0 V) bis zur oberen Grenze (1,8 V) durchgeführt werden kann. Es bleiben an den jeweiligen Grenzspannungen (Rails) sogenannte Gain und Offset Errorbereiche, die nicht gemessen werden können. In **Abbildung 4-9** ist der Messbereich samt Gain Error und Offset Error dargestellt.

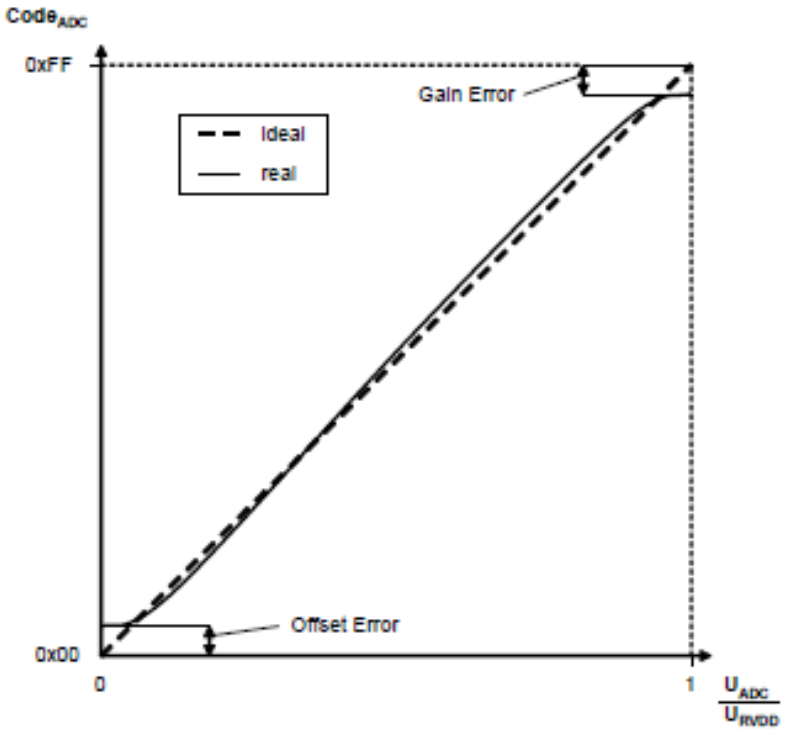


Abbildung 4-9: Gain- und Offset-Error aus [ENO12]

Der Messbereich des ADC wird durch eine positive sowie negative Referenzspannung festgelegt. Innerhalb dieser Grenzen findet eine Aufteilung mit einer 8-Bit Auflösung des ADCs statt. Ebenfalls kann der Bereich mit einer 10-Bit Auflösung dargestellt werden. Diese Auflösung kann durch das Serial Interface für ADC0 eingestellt werden. Daraus resultiert eine Veränderung des ADC1 auf eine geringere Auflösung von 6 Bit (siehe [ENO12] Abschnitt 3.8.1). Eine Beispielbeschaltung mit einem Rail-to-Rail Sensor ist in **Abbildung 4-10** dargestellt.

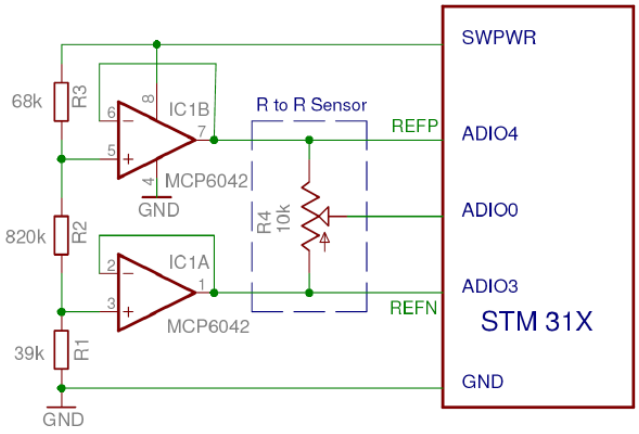


Abbildung 4-10: Beispiel „Rail-to-Rail“-Sensoranschluss an STM310 aus [ENO12]

Nach [ENO12] darf an den analogen Eingängen eine maximale Spannung von 2 V anliegen. Dies hat zur Folge, dass Sensoren mit höheren Betriebs- und Messspannungen mit einer zusätzlichen Beschaltung ausgestattet werden müssen. Generell lässt sich sagen, dass Sensorelemente mit einem analogen Messsignal in einem Spannungsbereich

von 0,1 V bis 1,7 V sehr speziell sind. Die Recherche nach einem passenden Erdfeuchtesensor, der diese Spannungswerte erreicht, ergab keine positiven Ergebnisse. Zu Testzwecken wird ein Potentiometer zwischen SWPWR und GND angeschlossen (vgl. Abbildung 4-2 in Abschnitt 4.1.2). Der Eingang REFN wird auf GND und REFP auf SWPWR gelegt. Durch den Anschluss von allen verfügbaren ADCs konnte ein Test der drei ADC-Module, bzgl. der Messgenauigkeit, durchgeführt werden. Die Tests zeigten, dass die ADCs mit einer 8-Bit Auflösung untereinander eine Maximalabweichung von 0,4 % aufweisen.

4.3 Transceiver Module 320 (TCM320)

4.3.1 Allgemeines und Typenbeschreibung

Bei dem Transceiver Module 320 handelt es sich um ein Sender-Empfänger Modul. Das Modul besitzt eine Pinout-Steckerleiste. Dadurch ergibt sich die Möglichkeit, ohne löten zu müssen, ein TCM zu nutzen. In **Abbildung 4-11** ist rechts ein TCM320 samt Steckerleiste und links ein TCM300 dargestellt.

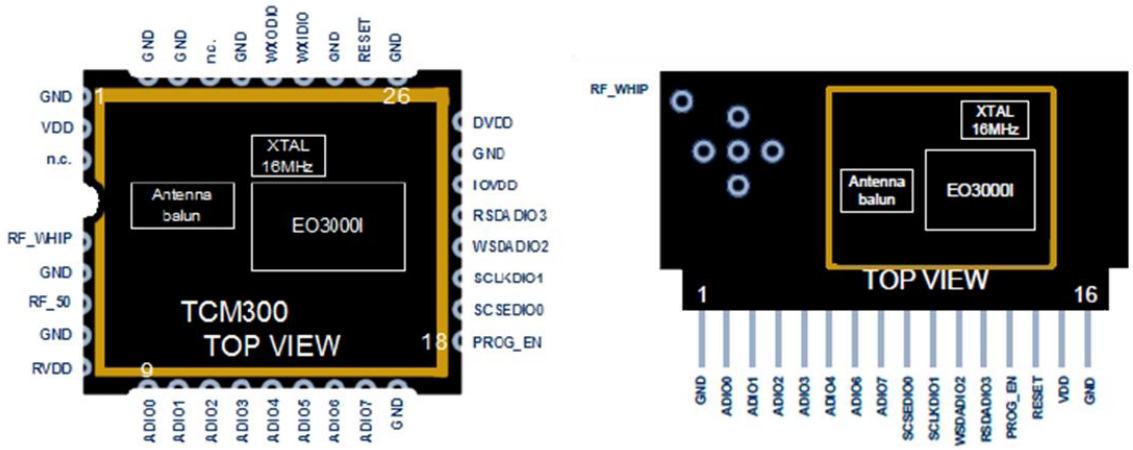


Abbildung 4-11: Darstellung Pinout für TCM300 und TCM320 aus [ENO17e]

Die Bezeichnung des Moduls dient, genauso wie beim STM3xx, zur Erkennung der Bauweise und Firmware eines Moduls. Bei einer Eigenentwicklung der Energieversorgung sowie der Antenne kann auch ein TCM300 ausgewählt werden. Das TCM320 beinhaltet die gleiche Steuereinheit wie der TCM300. Ein Nachteil bei der Nutzung des TCM320 gegenüber dem TCM300 ist, dass nicht alle Inputs/Outputs verwendet werden können. Dennoch sind die Möglichkeiten bzgl. Empfang und Senden sowie Repeater Funktion vollständig gegeben. Nach den Recherchen zu urteilen, besitzt das TCM320 keine Ver- und Entschlüsselungstechniken. Die Daten müssten durch ein externes Gerät ver- oder entschlüsselt werden.

4.3.2 Spannungsversorgung und Energieverbrauch

Ein Transceiver Module ist auf eine externe Spannungsversorgung angewiesen. Die Ausführung des TCM spielt dabei keine Rolle. Die benötigte Spannung zum Betrieb des

TCM320 beläuft sich auf 2,6 V – 3,3 V. Dies wird durch ein Netzteil mit 5 V Spannungsoutput und einem Spannungsregler (LD1117C33V) erreicht. Der LD1117C33V liefert laut [STM13] eine Ausgangsspannung von 3,24 V – 3,36 V und einen Ausgangsstrom von 800 mA bis 1 300 mA. Die Versorgungsspannung von 3,3 V ist somit für das TCM320 sichergestellt. Die Stromaufnahme im Betrieb beläuft sich im Empfangsmodus üblicherweise auf 33 mA und maximal auf 43 mA. Unter der Annahme, dass das TCM dauerhaft im Empfangsmodus läuft, wird nach Formel (2) mit U = 3,3 V und I = 43 mA eine maximale Leistungsanforderung (P_{max}) von 141,9 mW errechnet.

$$P_{max} = U * I \; in \; Watt \; [W] \qquad (2)$$

Der Sendemodus benötigt normalerweise einen Betriebsstrom von 24 mA und maximal 33 mA. Bei einer Stromaufnahme von 33 mA ergibt dies nach Formel (2) eine maximale Leistungsanforderung von 108,9 mW. Die Funkfrequenz des TCM320 beläuft sich auf 868,3 MHz. Nach [ETS16] darf demnach die maximale Sendedauer, unter der Annahme, dass kein LBT verwendet wird, nicht mehr als 1 % pro Stunde betragen. Das entspricht einer maximalen Sendezeit von 36 s. Die Leistung von 108,9 mW wird demnach für lediglich 36 s pro Stunde abgerufen.

Der LD1117C33V kann den benötigten Strom liefern und ist somit für diese Anwendung gut einsetzbar.

4.3.3 Konfigurationsverfahren des TCM320

Die recherchierten Konfigurationsmöglichkeiten des TCM320 sind auf zwei Verfahren begrenzt. Die Entwicklung einer eigenen Firmware ist ebenso möglich, wie die Nutzung der bereits vorprogrammierten Optionen durch die Beschaltung der Pins oder durch Kommandos über die UART Schnittstelle. Eine Möglichkeit zur Konfiguration über Dolphin Studio, wie dies beim STM310 möglich ist, konnte nicht gefunden werden.

Konfiguration durch eigene Firmware

Zum Programmieren mit einer neuen Firmware wird ein EOP350 programmer von EnOcean benötigt. Diese Variante wurde nicht weiter betrachtet und kann somit nicht erklärt werden.

Konfiguration über Pins oder Kommandos von der UART Schnittstelle

Eine wesentliche Konfiguration des TCM320 wird durch die Beschaltung des ADIO0 Pins durchgeführt. Dieser ist ein Eingangspin für einen ADC. Nach der Auswertung der Spannung wird je nach Spannungsbereich, wie in **Abbildung 4-12** zu sehen, ein bestimmter Betriebsmodus eingeschaltet. Die Beschaltung mit den Widerständen R1 bis R3, sowie dem Kondensator C1 geben genau an, wie die Betriebsmodi eingeschaltet werden können.

Mode	ADIO0 (MODE_SEL) input voltage range	Proposed component values
0	0% to 3.99% VDD	R1: 0 Ohm R2: leave open R3: leave open C1: leave open
1	4% to 11.99% VDD	R1: 1k2 ±1% R2: 15k ±1% R3: 150k ±1% C1: 100p
2	12% to 19.99% VDD	R1: 2k2 ±1% R2: 12k ±1% R3: 270k ±1% C1: 100p
3	20% to 27.99% VDD	R1: 3k9 ±1% R2: 15k ±1% R3: 68k ±1% C1: 100p
4	28% to 35.99% VDD	R1: 4k7 ±1% R2: 12k ±1% R3: 56k ±1% C1: 100p
5	36% to 39.99% VDD	R1: 5k6 ±1% R2: 10k ±1% R3: 56k ±1% C1: 100p

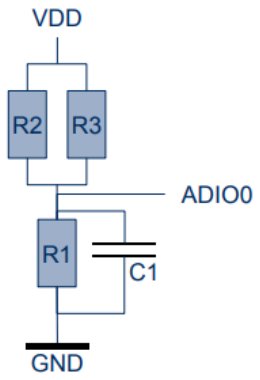


Abbildung 4-12: Spannungsbereiche und zugehörige Betriebsmodi des TCM320 aus [ENO17e]

In **Abbildung 4-13** sind die verschiedenen Modi aufgelistet. In der vorliegenden Diplomarbeit wird der Modus 0 verwendet. Dieser Modus erlaubt das Empfangen von sämtlichen Telegrammtypen und der Weitergabe per UART auf einen externen Mikrocontroller. Da das TCM320 nur als Empfänger eingesetzt wird, muss keine Sendefähigkeit gewährleistet werden.

Mode	Function	Output signal description	No. of channels
0	Unidirectional serial (ESP2)	SER_TX: UART output, supplies standard data blocks of information from all received EnOcean radio telegrams (9600 bps; 8 data bits, no parity bit, one start bit, one stop bit). For further information see chapter A.1	
1	Bidirectional serial interface (ESP2), teach-in capability for up to 30 entries[1]	SER_RX, SER_TX: Asynchronous bidirectional Interface, supplies standard data blocks of information from all received EnOcean radio telegrams (9600 bps; 8 data bits, no parity bit, one start bit, one stop bit). For further information see chapter A.1 LRN_TOGGLE: Learning mode status indicator	
2	Rocker Switch - 1 channel, teach-in capability for up to 30 entries[1]	Supplies the desired logic switching state "on/off" at CHANNEL0 when pushing the switch rockers	1
3	Rocker Switch - 4 channels, teach-in capability for up to 30 entries[2]	Same as Mode 2 but operation of 4 receiver channels (CHANNEL0, CHANNEL1, CHANNEL2, CHANNEL3)	4
4	Dimming - 1 channel, teach-in capability for up to 30 entries[1]	PWM is the PWM output ■ I-button pressed for shorter than 0.5 s: ON (Restore duty cycle stored before last switch-off). ■ O-button pressed for shorter than 0.5 s: OFF ■ O-/I-button pressed longer than 0.5 s: Duty cycle variation from 10% up to 100% (O=less, I=more; ~300 PWM steps, increment every 15 ms). Duty cycle variation stops when button is released. PWM_IND is active as long as duty cycle is not 0%	1
5	Reserved		

Abbildung 4-13: Erläuterung der Betriebsmodi des TCM320 aus [ENO17e]

Im Modus 0 wird automatisch das Repeater Level 1 gesetzt. Dies hat zur Folge, dass ein Originaltelegramm, welches noch nicht repeated wurde, nach einer zufälligen Verzögerung einmalig repeated wird. Um welche Zeitspanne bzgl. der Verzögerung es sich handelt, ist nicht geklärt. Außerdem kann der Repeater durch Anlegen einer positiven Spannung an ADIO2 abgeschaltet werden.

Die Sensitivity ist durch ADIO3 (Pin 5) mit den Optionen LOW und HIGH einstellbar. Dies sollte unbedingt beachtet werden, da eine LOW Einstellung nur Sendertelegramme weiterleitet bzw. empfängt, die sich in unmittelbarer Umgebung befinden. Die Option HIGH kann auch Sendertelegramme von entfernteren Orten empfangen.

4.3.4 Serieller Übertragungsmodus und Übertragungsgeschwindigkeit

Die Datenübertragung findet per UART mit einer Geschwindigkeit von 9 600 Bd statt. Die Übertragung von einem Byte beinhaltet acht Datenbits, ein Startbit und ein Stoppbit. Die Daten werden mit dem ESP2 übertragen. Der UART Output ist auf der Steckerleiste Pin 8. Es existiert keine Möglichkeit die Übertragungsgeschwindigkeit und den Übertragungsmodus per Beschaltung zu ändern. Bei der Entwicklung einer eigenen Firmware oder dem Programmieren des TCM mit einer neuen Firmware von EnOcean könnte es möglich sein, die Geschwindigkeit zu erhöhen. Dies setzt jedoch voraus, dass das ESP3 verwendet wird. Ob dies möglich ist, kann nicht abschließend geklärt werden.

4.4 Externer Mikrocontroller ATmega8

4.4.1 Allgemeines

Zur Auswertung eines Datentelegramms dient ein ATmega8 von Atmel. Der Mikrocontroller kann mit einer Spannungsversorgung von 4,5 V – 5,5 V betrieben werden. Laut [ATM13] beträgt die Stromaufnahme bei einer Versorgungsspannung von 3 V im Active-Mode 3,6 mA, im Idle-Mode 1 mA und im Power-Down Mode 0,5 µA. Die Stromaufnahme bei 5 V Versorgungsspannung wurde nicht getestet. Dies sollte im Zuge einer Nutzung mit einer Batterie nachgeholt werden, um eine eindeutige Aussage über die Betriebszeit tätigen zu können. Die Verwendung eines ATmega8 ist nicht zwingend erforderlich. Es wäre durchaus denkbar auch einen anderen Mikrocontroller, welcher eine UART Schnittstelle und mindestens einen digitalen Output anbietet, zu nutzen. Der komplette Quellcode des ATmega8 ist in Anhang B angefügt.

4.4.2 UART Initialisierung und Auswahl des passenden Quarz

Durch die Nutzung des ESP2 wird eine Datenübertragungsgeschwindigkeit von 9 600 Bd vorgegeben. Um die Kommunikation möglichst ohne Fehler durchführen zu können, muss ein passender Quarz für den Betrieb des ATmega ausgewählt werden.

In **Tabelle 4-1** ist ersichtlich, dass eine fehlerfreie Kommunikation mit einer ATmega8 Betriebsfrequenz von 11,0592 MHz bei einer Baudrate von 9 600 Bd möglich ist (blau

eingerahmt). Ebenso wären Betriebsfrequenzen von 14,7456 MHz sowie 18,432 MHz möglich. Für weitere Informationen bzgl. der nutzbaren Frequenzen des ATmega8 im Zusammenhang mit der UART Kommunikation ist [ATM13] zu nutzen.

Tabelle 4-1: Fehleranfälligkeit bei genutzter Baudrate und Betriebsfrequenz des ATmega8 verändert nach [GRE06]

		Betriebsfrequenz ATmega8					
		11,0592 MHz	14,318 MHz	14,7456 MHz	16 MHz	18,432 MHz	20 MHz
Baudrate	2400	0 %	-0 %	0%	-0,10%	0%	-0%
	4800	0 %	0,2 %	0%	0,20%	0%	0,20%
	9600	0 %	0,2 %	0%	0,20%	0%	0,20%
	14400	0 %	0,2 %	0%	0,60%	0%	-0,20%
	19200	0 %	-0,8 %	0%	0,20%	0%	0,20%
	28800	0 %	0,2 %	0%	-0,80%	0%	0,90%
	38400	0 %	1,3 %	0%	0,20%	0%	-1,40%
	57600	0 %	-2,9 %	0%	2,10%	0%	-1,40%
	76800	0 %	-2,9 %	0%	0,20%	0%	1,70%
	115200	0 %	-2,9 %	0%	-3,50%	0%	-1,40%
	230400	0 %	-2,9 %	0%	8,50%	0%	8,50%
	250000	-7,8 %	-10,5 %	-7,80%	0%	-7,80%	0%

Die Einstellungen des UART erfolgen über die Register UBRRH, UBRRL, UCSRB und UCSRC. In **Abbildung 4-14** ist ein Ausschnitt des Programmcodes dargestellt.

```
void initUART(){
    /* Setting baud rate */
    UBRRH = (unsigned char) (UBRR>>8);
    UBRRL = (unsigned char) (UBRR&0xff);

    /* RXEN Receive enable / RXCIE Receive complete Interrupt enable */
    UCSRB = (1<<RXEN)|(1<<RXCIE);

    /* Setting data format: 8 data bits, 1 stop bit, no parity bit */
    UCSRC = (1<<URSEL)|(1<<UCSZ1)|(1<<UCSZ0); // 8N1

    // Flush Receive-Buffer
    do
    {
        uint8_t dummy;
        (void) (dummy = UDR);
    }
    while (UCSRA & (1 << RXC));
    sei();
}
```

Abbildung 4-14: UART Initialisierung verändert nach [ANO17]

UBRR ist die errechnete Baudrate. Diese wurde vorher per #define und nach der Formel (3) von [ANO17] definiert.

$$UBRR = \frac{F_{CPU}}{Baudrate * 16} - 1 \qquad (3)$$

F_{CPU} stellt die Taktfrequenz des Mikrocontrollers dar. Ebenso wird die gewünschte Baudrate von 9 600 Bd eingesetzt.

Die Baudrate wird über die Register UBRRH/L eingetragen. Laut [ANO17] muss UBRRH vor UBRRL beschrieben werden. Der nach Formel (3) errechnete Wert wird als 16 Bit Wert in UBRR gespeichert. Für die Eintragung in das 8-Bit Register UBRRH wird das höherwertige Byte von UBRR um acht Bit nach rechts verschoben. Die Eintragung in das 8-Bit Register UBRRL wird durch eine Und-Verknüpfung von UBRR mit 0xFF, um die höherwertigen Bits von UBRR zu eliminieren, durchgeführt.

Die Aktivierung des Empfangs von Daten und des passenden Interrupts werden durch das UCSRB Register durchgeführt. Hierzu werden die Bits RXEN und RXCIE gesetzt. Der UART und Interrupt sind somit aktiviert. Das UCSRB Register ist in **Tabelle 4-2** dargestellt.

Tabelle 4-2: Aufbau und Beschreibung des UCSRB-Registers verändert nach [ANO17]

Bit	Name	Bedeutung
7	RXCIE	enable Interrupt receive complete
6	TXCIE	enable Interrupt transmit complete
5	UDREIE	enable Interrupt transmit data register free
4	RXEN	enable receive
3	TXEN	enable transmit
2	UCSZ2	Anzahl von Datenbits zusammen mit UCSZ1 / UCSZ0 des UCSRC
1	RXB8	bei gesetztem CHR9-Bit: 9. empfangenes Datenbit
0	TXB8	bei gesetztem CHR9-Bit: 9. zu übertragendes Datenbit

Der Interrupt wird nach einem kompletten Datenempfang ausgelöst. Über das UCSRC Register, dargestellt in **Tabelle 4-3**, werden der Asynchronmodus, das Paritätsbit, die Stoppbits, sowie die Anzahl der Datenbits eingestellt. Die Taktpolarität UCPOL ist nur im Synchronmodus erforderlich und muss nicht eingestellt werden.

Tabelle 4-3: Aufbau und Beschreibung des UCSRC-Registers verändert nach [ANO17]

Bit	Name	Bedeutung	Funktion
7	URSEL		
6	UMSEL	synchron/asynchron Modus	0: asynchron 1: synchron
5	UPM1	Parität	00: none 01: reserved 10: even priority 11: odd priority
4	UPM0		
3	USBS	Stoppbits	0: 1 Stoppbit 1: 2 Stoppbits
2	UCSZ1	Anzahl von Datenbits zusammen mit UCSZ2 des UCSRB	000: 5 bit 001: 6 bit 010: 7 bit 011: 8 bit 111: 9 bit
1	UCSZ0		
0	UCPOL	synchron: Phasenlage	

Wichtig ist, dass das URSEL Bit gesetzt werden muss. Dies hat laut einem Beitrag nach [ANO17] folgende Bewandtnis:

> „Um Zugriffsadressen einzusparen, wurde von Atmel ein etwas seltsamer Weg gewählt. Das UCSRC Register und das High-Byte des Baudratenregisters teilen sich dieselbe Registeradresse! Um der Hardware mitzuteilen, welche Bedeutung ein zugewiesener Wert haben soll, entweder neue Belegung des Baudratenregisters oder eben Konfiguration des UCSRC Registers, dient dieses Bit. Ist es nicht gesetzt, dann wird eine Zuweisung immer als Zuweisung an das High-Byte des Baudratenregisters angesehen, selbst wenn das so nicht beabsichtigt war. Nur dann wenn dieses Bit gesetzt ist, dann wird eine Zuweisung auch tatsächlich als eine Zuweisung an das UCSRC Register gewertet und die Konfiguration verändert. Lässt man das Bit irrtümlich weg, dann verursacht eine Zuweisung an UCSRC eine Veränderung der Baudrateneinstellung!“

Der Initialisierungsprozess beinhaltet das Flushen des UART Data Register (UDR), um eventuell anstehende Daten nach einem Reset auszuschließen. Dies erfolgt durch das ständige Auslesen des UDR, solange das RXC Bit des UCSRA Registers gesetzt ist. Ein gesetztes RXC Bit bedeutet, dass eine vollständige Datenübertragung stattgefunden hat. Abschließend werden Interrupts für den Mikrocontroller erlaubt und aktiviert.

4.4.3 Interrupt Service Routine

Die Erstellung der Interrupt Service Routine (ISR) wurde unter Zuhilfenahme von [ANO17] und [BEC14] erstellt.

Die Interrupt Service Routine wird immer dann ausgelöst, wenn eine vollständige UART Übertragung stattgefunden hat. Der Code der ISR ist in **Abbildung 4-15** dargestellt.

```
// Interrupt Service Routine: triggered when sign is rdy @ UDR
ISR(USART_RXC_vect)
{
    static unsigned char zeichen=0;

    zeichen=UDR;
    buffer[irw]=zeichen;
    irw+=1;
    if (irw==RING_SIZE)
        irw=0;
    count ++;
}
```

Abbildung 4-15: Codebeispiel zur Nutzung einer ISR in Verbindung mit der UART

Um die ankommenden Daten aus dem UDR zu speichern, wird ein Ringbuffer genutzt. Die Schreibposition innerhalb des Ringbuffers wird durch die short Variable irw bestimmt. Nach jedem erfolgreichen Speichern eines Bytes wird irw um eins erhöht. Ist die letzte Position des Ringbuffers erreicht und eine erfolgreiche Speicherung durchgeführt, wird irw auf 0 gesetzt. Dadurch wird die Schreibposition an den Anfang zurückgesetzt (siehe **Abbildung 4-16**). Die Größe des Ringbuffers wird durch

RING_SIZE angegeben. Außerdem wird die Anzahl der empfangenen Bytes durch count gezählt. Dies ist beim Auslesen des Ringbuffers von Bedeutung.

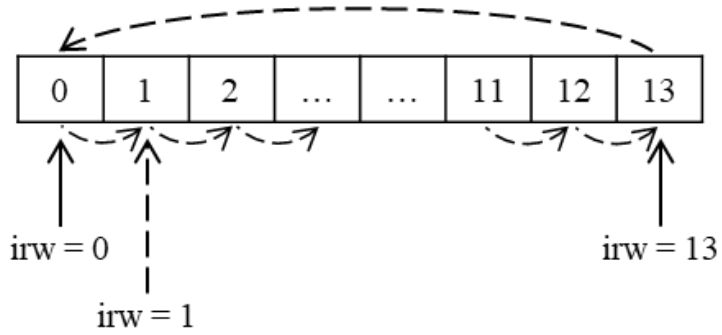


Abbildung 4-16: Funktionsweise des Schreibzugriffs auf den Ringbuffer

Der Ringbuffer besitzt eine Größe von 14 Bytes, da die Datenmenge des TCM320 durch das ESP2 definiert ist. Der Vorteil bei der Nutzung eines Interrupts liegt darin, dass der eigentliche Programmablauf nur dann gestört wird, wenn eine neue vollständige Übertragung stattgefunden hat.

4.4.4 Analyse empfangener Daten

Bevor die Analyse der Daten durchgeführt werden kann, werden die einzelnen Bytes in einem Array gespeichert. In **Abbildung 4-17** ist das Verfahren als Code-Ausschnitt dargestellt. Das Array ESPB hat eine Größe von 14 Bytes.

```
while(1){
    temp = ugetchar();
    if(temp!=-1){
        if(irr == 0){
            ESPB[RING_SIZE-1]=temp;
            ...
        }
        else{
            ESPB[irr-1]=temp;
        }
    }
}
```

```
short int ugetchar(void)
{
    unsigned char c;
    if (count>0)
    {
        cli();
        c=buffer[irr];
        irr++;
        if (irr==RING_SIZE)
            irr=0;
        count --;
        sei();
        return c;
    }
    else return -1;
}
```

Abbildung 4-17: Codebeispiel zur Datenspeicherung aus dem Ringbuffer verändert nach [BEC14]

Im Hauptprogramm wird die Funktion ugetchar() mit jedem Schleifendurchlauf aufgerufen. Sind noch nicht ausgelesene Zeichen im Ringbuffer vorhanden, dargestellt durch den Zähler count, wird kurzfristig der Interrupthandler deaktiviert. Je nachdem auf welches Byte die Leseposition (irr) im Buffer verweist, wird dieses ausgelesen und die Leseposition um eins erhöht. Wenn sich die Leseposition am Ende des Buffers

befindet, dann wird diese auf die Startposition des Buffers gesetzt, wie in **Abbildung 4-18** zu sehen.

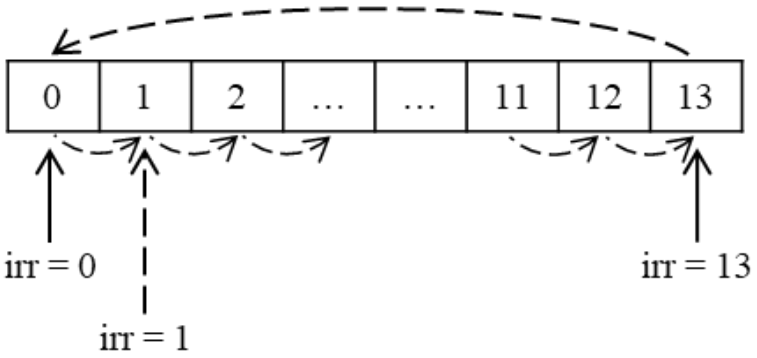


Abbildung 4-18: Funktionsweise des Lesezugriffs auf den Ringbuffer

Abschließend wird count um eins verringert und der Interrupthandler wieder aktiviert. Das ausgelesene Byte wird an die aufrufende Funktion zurückgegeben. Im Hauptprogramm wird nach jedem Aufruf von ugetchar() überprüft, ob ein neues Byte ausgelesen wurde. Ist dies der Fall, wird geprüft, ob die Leseposition (irr) der Startposition (irr = 0) entspricht. Wenn dies zutrifft, ist eindeutig geklärt, dass volle 14 Byte im ESPB Array gespeichert wurden und demnach eine komplette Datenübertragung per UART vom TCM320 stattgefunden hat.

Die Geräte-ID wird durch den Abgleich der Bytes 8, 9, 10 und 11 des ESPB überprüft. Wenn eine korrekte ID übertragen wurde, findet abschließend die Überprüfung zur Erkennung des Teach-In Telegramms statt (vgl. **Abbildung 4-19**).

```
temp = ugetchar();
if(temp!=-1){
    if(irr == 0){
        ESPB[RING_SIZE-1]=temp;
        //ID check
        if(ESPB[8] == 0x00 && ESPB[9] == 0x83 && ESPB[10] == 0x2D && ESPB[11] == 0x64){
            //Teach_In or Data?
            //ESPB[7]-Bit7 => LRN Type: 1->Teach-In | 0->Data
            if( ESPB[7] <= 0x8F && ESPB[7] >= 0x80 ){
```

Abbildung 4-19: Codebeispiel zur Erkennung von Geräte-ID und Teach-In Telegramm

Die Geräte-ID des verwendeten STM310 wurde in dem Code implementiert und kann nicht ohne erneutes Flashen des ATmega8 geändert werden. Wenn es sich um ein Teach-In Telegramm handelt, wird die angesteuerte LED ausgeschaltet und blinkt im Abstand von 200 ms zweimal, um einen Empfang von einem Teach-In Telegramm anzuzeigen. Handelt es sich nicht um ein Teach-In Telegramm, wird das Byte 4 des ESPB ausgelesen und mit den Schwellwerten für das Ansteuern der LED verglichen. Wenn ein Schwellwert über- oder unterschritten wird, wird die LED an- bzw. ausgeschaltet.

4.5 User Interface

Die Entwicklung des nachfolgend näher beschriebenen User Interface basiert auf [KET15]. Für diese Diplomarbeit sind jedoch Anpassungen vorgenommen worden.

4.5.1 Kommunikationsweg

Das User Interface wird über einen USB300 Stick von EnOcean mit Daten versorgt. In **Abbildung 4-20** ist ein Überblick über den Kommunikationsweg zwischen STM310 und dem, vom FTDI erzeugten, Virtual COM Port zu erkennen.

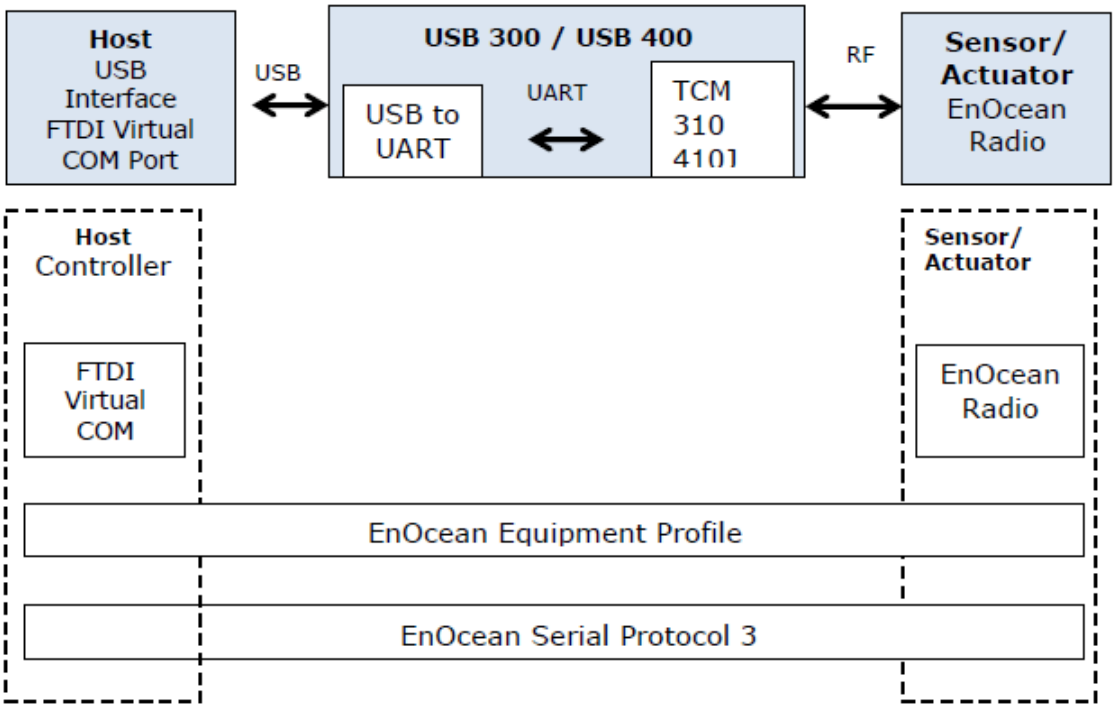


Abbildung 4-20: Kommunikationsweg vom STM310 zum virtuellen COM-Port aus [ENO17c]

Der Host ist in diesem Fall der PC, welcher vom Benutzer genutzt wird. Die Übertragung der Daten findet durch das ESP3 statt. Das EEP wird durch das Senden eines Teach-In Telegramms übertragen. Durch den integrierten FTDI Chip werden die Daten auf dem virtuellen COM-Port bereitgestellt. Über Node.js wird dieser virtuelle COM-Port abgefragt.

4.5.2 Tests zur Datenübertragung

Zum Verständnis der Datenübertragung des USB300 wurde HTerm verwendet. Es wurde der virtuelle COM-Port genutzt und testweise Daten durch das STM310 gesendet. In **Abbildung 4-21** sind Daten- und Teach-In Telegramme zu erkennen.

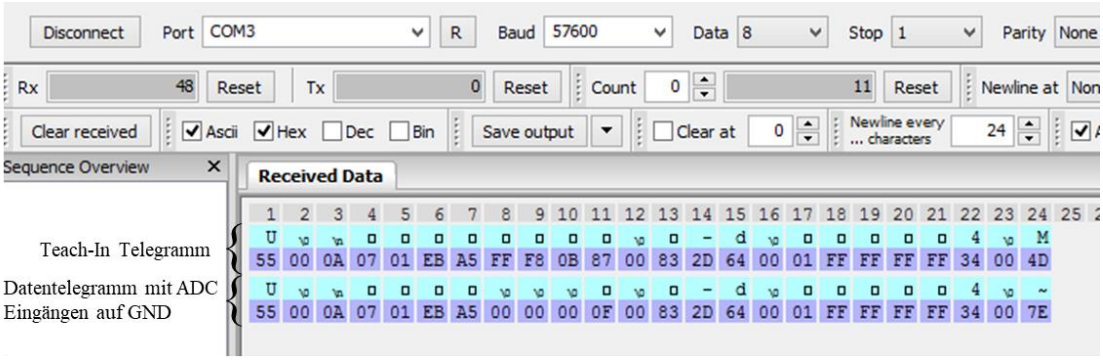


Abbildung 4-21: Darstellung von Daten- und Teach-In Telegramm nach ESP3 in HTerm

Es wurde eine Geschwindigkeit von 57 600 Baud eingestellt, da dies als Standardgeschwindigkeit des ESP3 gilt. Höhere Geschwindigkeitseinstellungen in HTerm sind nur dann sinnvoll, wenn eine erhöhte Geschwindigkeit im TCM310 des USB300 eingestellt wird. Weiterführende Informationen müssen dazu recherchiert werden. Ebenso ist der Datenübertragungsmodus 8N1 (8 Datenbits, kein Paritätsbit und ein Stoppbit) einzustellen. Die zwei Telegramme aus Abbildung 4-21 wurden im weiteren Verlauf zu Testzwecken eingesetzt, um Daten zu simulieren und das User Interface zu entwickeln. Es wurde das Programm Virtual Serial Port Driver eingesetzt, um Eingaben in HTerm und dessen eingestellten Virtual COM-Port zu dem vom User Interface genutzten Virtual COM-Port weiterzuleiten, wie auch in nachfolgender **Abbildung 4-22** ersichtlich.

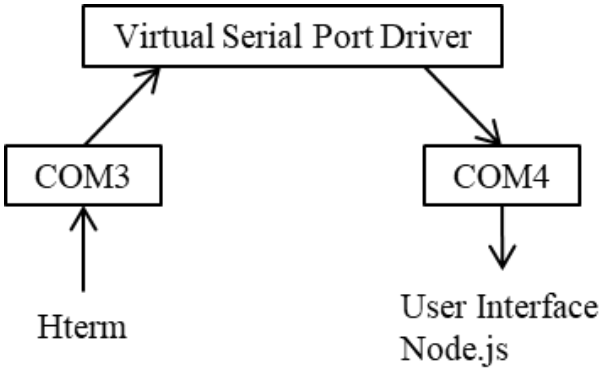


Abbildung 4-22: Datenverlauf durch HTerm, Virtual Serial Port Driver und Node.js

Dadurch können Datentelegramme auf einen virtuellen COM-Port geschrieben werden und auf einem anderen abgeholt werden. Um das User Interface zu testen und zu entwickeln, müssen somit nicht ständig Datentelegramme vom STM310 gesendet werden. Die original empfangenen Telegramme vom STM310 wurden zu Testzwecken speziell in dem Bereich der Messdaten und der Geräte-ID geändert. Bei der Nutzung vom ESP3 werden bei der Entwicklung der Demoapplikation 24 Bytes übertragen. Bei einer Datenübertragung mit ESP2 muss berücksichtigt werden, dass lediglich 14 Bytes übertragen werden.

4.5.3 Datenanalyse und Darstellung

Die Datendarstellung findet über einen frei wählbaren Browser statt. Erfolgreich getestet wurde mit den Browsern Google Chrome und Mozilla Firefox. Die Ausführung der Batch-Datei „Start_UI_STM310.bat“ dient zum schnellen Starten des gesamten Prozesses. In dieser Datei wird lediglich der auf dem Host-PC eingestellte Standardbrowser mit „http://localhost:8000“ aufgerufen. Der Port 8000 ist nicht zwangsläufig zu wählen, sondern kann geändert werden. In diesem Zusammenhang ist zu beachten, dass auch der Port in dem serverseitigen JavaScript der Node.js Anwendung „userinterface.js“ abgeändert werden muss. Das serverseitige JavaScript wird unter der Verwendung von Node.js ebenfalls durch Ausführen der Batch-Datei gestartet (siehe **Abbildung 4-23**).

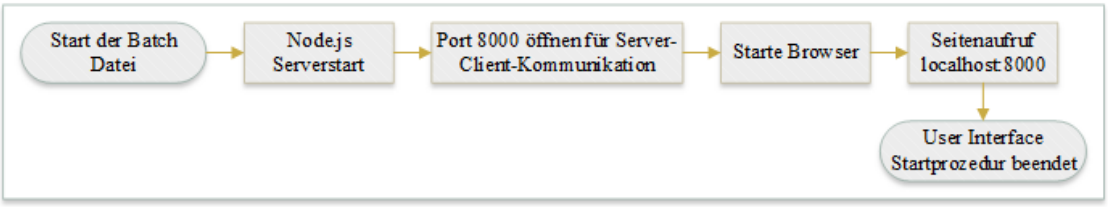


Abbildung 4-23: Startprozedur des User Interface per Batch-Datei

Der Seitenaufbau im Browser wird durch eine html-Datei erzeugt. Diese Datei muss sich im selben Ordner wie die userinterface.js befinden und als Index.html bezeichnet sein. Dies wird durch das node.js Zusatzpaket serve-static definiert.

Die Server-Client Kommunikation am Beispiel eines Starts der Messwertaufzeichnung ist in **Abbildung 4-24** dargestellt. Die Anzeige von den Daten wird erst dann aktiv, wenn der „Start“-Button betätigt wurde. Die Kommunikation zwischen dem USB300 und dem User Interface erfolgt immer über das serverseitige JavaScript. Es wird darauf gewartet, dass entweder neue Daten über den virtuellen COM-Port eingehen oder dass die clientseitige Anwendung Befehle sendet.

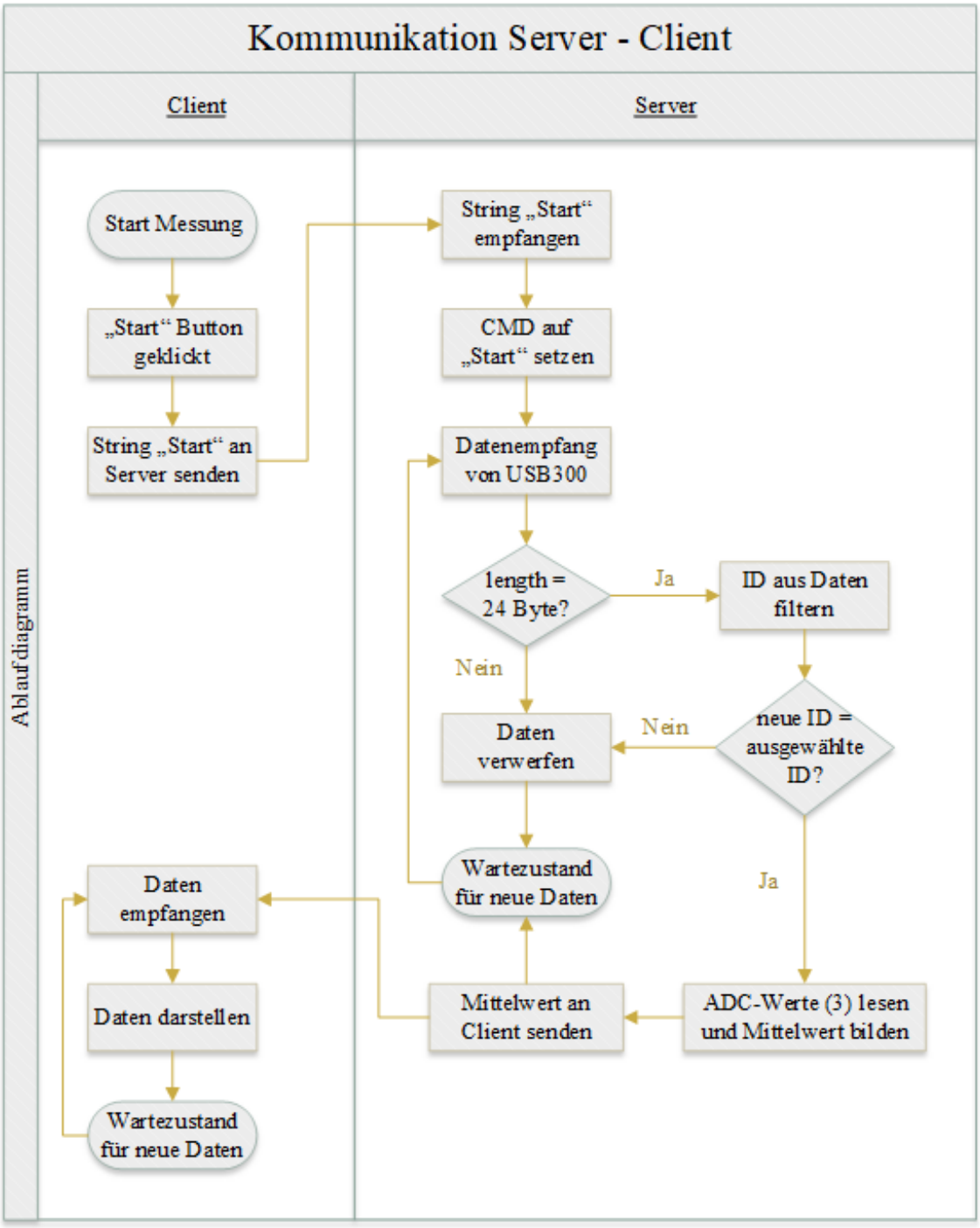


Abbildung 4-24: Server-Client Kommunikation am Beispiel eines Starts der Messwertaufzeichnung

Bei der Verwendung des USB300 wird das ESP3 genutzt. Vorausgesetzt wird das 4-BS-Format, um die Funktionsweise des User Interface zu gewährleisten. Daraus ergibt sich, dass die Bytes 12, 13, 14 und 15 die Geräte-ID darstellen. Die Kontrolle der ID wird nach der Überprüfung auf die Datenlänge durchgeführt. Stimmt die ID mit der ausgewählten ID überein, wird aus den drei ankommenden ADC-Messwerten ein Mittelwert gebildet und an das clientseitige JavaScript gesendet.

Bei Betätigung des „Hinzufügen“ Button wird auf eine neue Geräte-ID gewartet. Um diese neue ID hinzuzufügen, muss ein Teach-In Telegramm vom STM310 gesendet werden. Bei dem Hinzufügen von der ersten Geräte-ID wird diese sofort ausgewählt. Ein neuer Sender kann nur hinzugefügt werden, wenn keine aktuelle Anzeige über „Start“ aktiviert wurde. In **Abbildung 4-25** ist das User Interface mit Messwerten dargestellt.

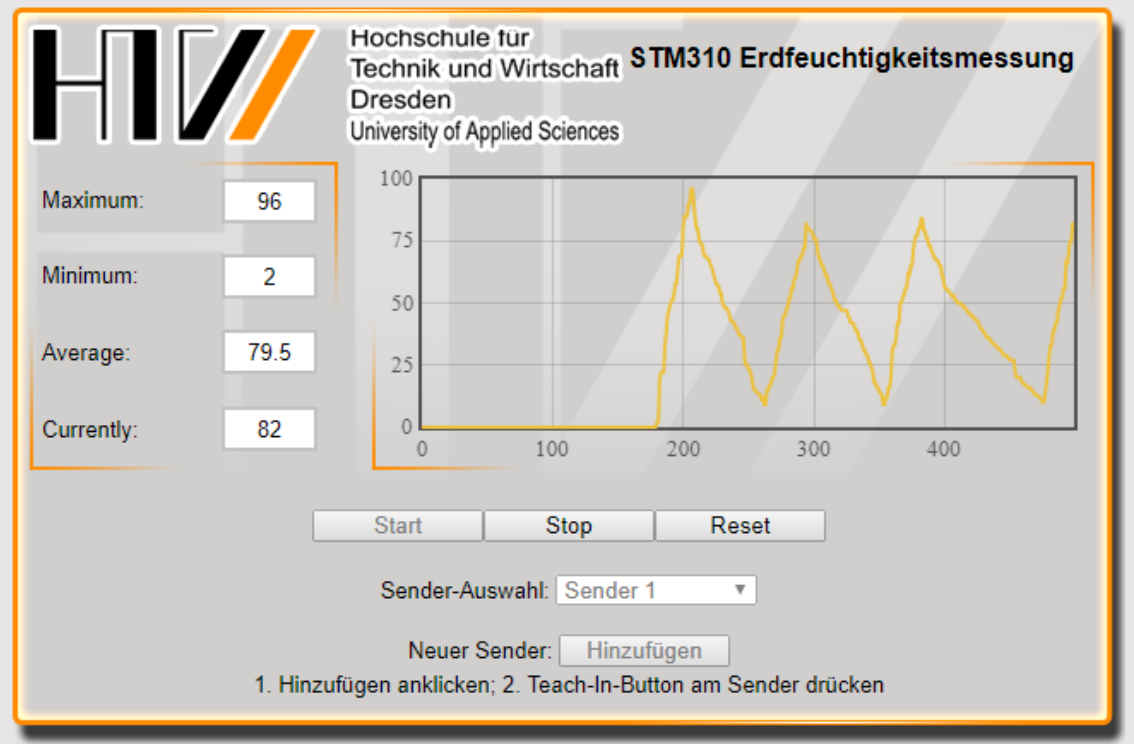


Abbildung 4-25: User Interface mit Messwerten

Bei der Registrierung von mehreren Sendern können nur Messwerte von dem ausgewählten Sender angezeigt werden. Die Anzeige wird jedoch bei einer Veränderung der Senderauswahl zurückgesetzt. Durch das Betätigen des „Reset“ Button wird die Anzeige ebenfalls zurückgesetzt.

Das Erscheinungsbild des User Interface ist durch eine CSS-Datei und erstellten Bilddateien definiert.

5 Auswertung

5.1 Nutzbarkeit

Das EnOcean Scavanger Module 310 ist in seiner Nutzbarkeit eingeschränkt. Die Pinout-Steckerleiste begrenzt den Zugang zu den möglichen Pins. Daraus ergeben sich nur geringe Möglichkeiten bei der Nutzung von externen Sensoren. Diese Sensoren können durch das STM maximal mit 1,8 V betrieben werden. Eine maximale Stromaufnahme von 5 mA grenzt die Nutzbarkeit weiter ein. Unter der Annahme, dass ein Sensorelement (externe Elektronik) mit den elektrischen Werten betrieben werden kann, sind begrenzende Faktoren die Anzahl an ADCs sowie DIs. Vor allem letzteres grenzt die Nutzbarkeit von Schaltern oder binären Sensoren ein. Dennoch ist die Möglichkeit gegeben, mit nur einem STM310 drei analoge Sensorwerte im Bereich von 0,1 V bis 1,7 V zu messen. Zusätzlich sind drei binäre Sensoren bzw. Schalter/Taster abprüfbar. Demnach können mit nur einem STM310 bis zu drei analoge sowie drei binäre Messsignale von unterschiedlichen Sensoren gleichzeitig übertragen werden. Dies hängt von dem verwendeten Telegrammtyp ab.

Ein STM300 ist universell einsetzbar und bietet gegenüber dem STM310 mehr Möglichkeiten in Verbindung mit einer eigenen entwickelten Elektronik. Durch die Entwicklung einer eigenen Spannungsversorgung für ein STM können auch externe Sensoren mit einem höheren Energieverbrauch genutzt werden. Dies ist speziell bei Sensoren wichtig, welche ausgehend von ihrer Messtechnologie schon mehr Energie benötigen. Die Analyse und Entwicklung eines externen Sensorelementes mit passenden elektrischen Werten muss von einer entsprechenden Fachkraft durchgeführt werden und wird nicht weiter erläutert.

Die Nutzung von dem STM330 ist nur bedingt zu empfehlen, vorausgesetzt es soll keine Luftfeuchtigkeit und/oder Temperatur gemessen werden. Ein Temperatursensor ist auf dem STM330 installiert. Ein Luftfeuchtesensormodul (HSM100) kann über die Pinout-Steckerleiste installiert werden. In der Firmware sind beide EEPs zu den Sensoren implementiert.

Abschließend ist zu sagen, dass ein STM3xx mit den existierenden Firmwares hauptsächlich als Sender dient. Da die Steuereinheit des STM3xx identisch mit der des TCM3xx ist, kann unter Umständen mit einer neuen Firmware auch das Empfangen von Telegrammen implementiert werden. Daraus resultiert eine erhöhte Stromaufnahme. Um eine genauere Aussage zu treffen, müsste dies noch analysiert werden. Für den STM300 besteht laut EnOcean Website bereits die Möglichkeit eine Receiver Funktionalität zu implementieren.

Ein Transmitter-Receiver-Modul kann sowohl senden als auch empfangen und muss durch die erhöhte Stromaufnahme über ein Netzteil mit einer Spannung von 2,7 V - 3,3 V betrieben werden, was den Einsatz als mobile, energieautarke Sendeeinheit ausschließt. Durch die hardwareseitige Beschaltung des ADC0 können bestimmte Modi ohne Programmierung eingestellt werden. In Verbindung mit einem externen Mikrocontroller, der zur Auswertung dient, kann das TCM320 als Schnittstelle

zwischen Funk- und serieller Kommunikation dienen. Es ist auch möglich schon innerhalb des TCM eine Filterung der ankommenden Telegramme durchzuführen. Das TCM bietet diesbezüglich ebenfalls Einstellungsmöglichkeiten, ohne eine Programmierung durchführen zu müssen. Die Filterung ist auf 30 Einträge begrenzt.

5.2 Datenübertragung

Die Datenmenge und die Übertragungsgeschwindigkeit sind im Wesentlichen vom EnOcean Radio Protocol und dem EnOcean Serial Protocol abhängig. Das ERP2 gibt eine maximale Payloadlänge von 59 Bytes für Implementationen auf EO3000I und EO1000I Steuereinheiten vor (vgl. [ENO17b]). Bei anderen Steuereinheiten soll eine Übertragung von 255 Bytes möglich sein. Eine weitere Betrachtung diesbezüglich ist nötig, um eine genauere Aussage zu treffen.

Das verwendete STM310 nutzt das ERP1. Dieses Protokoll gibt keine Payloadlänge an, stattdessen werden die Bytes von ORG bis inkl. dem Hash-Byte gezählt. Bei festen Payloads, wie bei dem 1 BS, 4 BS und RPS-Telegramm, kann so die korrekte Anzahl an Bytes erkannt werden. Über die maximale Länge des Telegrammtyps VLD kann keine Aussage getroffen werden.

Es ist wichtig zu verstehen, dass das ERP1/2 und ESP2/3 aufeinander abzustimmen sind. Bei einer Übertragung von 59 Bytes mit ERP2 zum Empfangsgerät TCM, welches das ESP2 nutzt, werden Bytes verloren gehen, da ESP2 eine maximale Übertragungsmenge von 28 Bytes aufweist. Bei der Interpretation der Daten innerhalb des TCM und ohne serielle Übertragung an einen externen Mikrocontroller, könnte es sein, dass keine Daten verloren gehen. Das müsste genauer betrachtet werden und kann hier nicht erläutert werden.

Die serielle Übertragungsgeschwindigkeit unter Nutzung des ESP2 beträgt 9 600 Baud. Diese Übertragungsrate kann nicht geändert werden. Im Gegensatz dazu ist eine Übertragungsgeschwindigkeit durch das ESP3 zwischen 57 600 und 460 800 Baud angegeben. Eine Verringerung auf weniger als 57 600 Baud wurde nicht geprüft.

Generell lässt sich sagen, dass keine großen Datenmengen per Funk übertragen werden können.

5.3 Geräte- und Datenidentifizierung

Mit jedem Datentelegramm wird eine Geräte-ID mitgesendet. Dadurch lassen sich die Messdaten eindeutig zuweisen. Eine Filterung der Geräte-ID kann innerhalb des TCM320 oder in einem externen Mikrocontroller geschehen. Je nach Einstellungsmerkmalen des TCM320 werden nicht registrierte Geräte-IDs geblockt und das Telegramm nicht per UART weitergeleitet. Getestet wurde dieses Verfahren nicht.

Die Registrierung der Geräte-ID innerhalb des TCM320 erfolgt über das Teach-In Telegramm. Wenn keine ID Filterung im TCM320 vorgenommen wird, sollte dennoch

ein Teach-In Telegramm gesendet werden, um eine ID Filterung innerhalb der externen Auswertetechnik (PC über USB300 oder externer Mikrocontroller in Verbindung mit TCM320) durchführen zu können. Das Senden eines Teach-In Telegramms wird der externen Auswertetechnik vorher vom Benutzer mitgeteilt (z. B. durch das Betätigen eines Buttons). Somit können unbekannte Sender und deren Daten zur Auswertung ausgeschlossen werden. Der erläuterte Vorgang ist in **Abbildung 5-1** grafisch dargestellt.

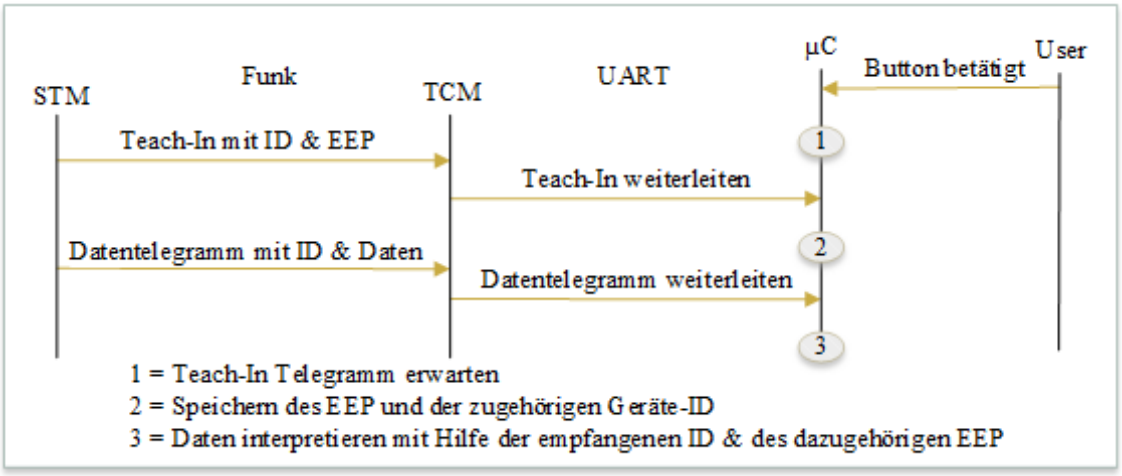


Abbildung 5-1: Initialisierungsprozess mit erster Datenübertragung

Mit dem Teach-In Telegramm wird ebenfalls das EEP des Sendegerätes übertragen. Dieses EEP dient der Auswertetechnik als Information, um welche Daten es sich handelt und wie diese zu interpretieren sind.

Mit dem Teach-In Telegramm wird auch die Hersteller-ID gesendet. In welchem Zusammenhang die Geräte-ID und die Hersteller-ID zur Erkennung des Sendegerätes sowie zur Interpretation der Daten bei aktuell gängigen Geräten stehen, muss noch weiter erforscht werden. Es könnte durchaus möglich sein, dass eine Geräte-ID identisch mit einer zweiten ist und nur die Hersteller-IDs unterschiedlich sind. Dies würde bei der Erkennung eines Telegramms eventuell zu einer Missdeutung der Daten führen, wenn nur auf die Geräte-ID überprüft wird. Eine genaue Recherche konnte aus Zeitgründen nicht durchgeführt werden.

5.4 Konfigurationsfähigkeit

Die teuerste und aufwändigste Konfigurationsmethode für die EnOcean Module ist die Entwicklung einer kompletten Firmware. Der Bedarf an einem kommerziellen Compiler von Keil, mit einem Preis von ca. 8 000 Euro (nach Aussage von Herrn Christian Bach von EnOcean), schränkt diese Methode jedoch sehr ein.

Bei der Konfigurationsmethode via Dolphin Studio und des passenden programmers, können die meisten und wichtigsten Einstellungen getätigt werden. Diese Methode ist deutlich günstiger. Das programmerboard EOP350 ist Bestandteil des EDK350. Der Preis beläuft sich nach Informationen von der Internetseite der Firma Farnell auf einen

Preis von ca. 170 Euro. Die benötigte Software ist nach einer Registrierung auf EnOcean.com kostenlos herunterzuladen. Einstellungen, wie das EEP, die Manufacturer-ID und die Threshold-Values, sind auf diese Weise schnell und bequem zu ändern. Es können u. a. STM und PTM (Ausnahme PTM210/215) auf diese Weise konfiguriert werden. Diese Methode ist bei der Entwicklung eigener EnOcean Projekte unbedingt zu empfehlen.

Die günstigste Variante ist die Konfiguration über Pin-Verbindungen. Allerdings bietet diese Variante bei dem verwendeten STM310 die wenigsten Optionen. Die möglichen Konfigurationsoptionen sind von Modul zu Modul verschieden. Ein TCM320 bietet über die Pin-Belegung schon sehr viele Einstellungsmöglichkeiten. In Verbindung mit der UART Kommunikation, welche zur Übertragung von Kommandos an den TCM320 genutzt werden kann, sind noch mehr Einstellungen möglich.

Je nachdem wie die Weiterleitung bzw. Auswertung der Daten erfolgen soll, muss die entsprechende Konfigurationsmethode ausgewählt werden. Bei einer unbedingten Änderung des Threshold-Values, um eventuell eine Änderung des Messwertes früher zu senden, würde die Konfiguration über PINs nicht ausreichen.

6 Messversuch zur Bestimmung der Erdfeuchtigkeit

6.1 Versuchsaufbau

Bei der Durchführung des Messversuchs wurden zwei unterschiedliche Messaufbauten genutzt. Dabei bleiben die getroffenen Einstellungen bzgl. des STM310 und des TCM320 gleich. Nur die Beschaltung der ADCs des STM310 ändert sich.

6.1.1 Messversuch mit Potentiometer

Der Versuchsaufbau mit einem Potentiometer dient zu Testzwecken und soll eine Simulation unterschiedlicher Sensorspannungen darstellen. In dem Aufbau, wie in **Abbildung 6-1** zu sehen, dienen die beiden Widerstände R1 und R3 mit jeweils 680 Ω, in Verbindung mit dem Potentiometer R2, zur Erzeugung der positiven und negativen Referenzspannungen. Durch diese Maßnahme kann der variable Ausgang des Potentiometers von 0 Ω (blau) bis 9,6 kΩ (rot) genutzt werden. Der Maximalwiderstand von 9,6 kΩ wurde durch ein Multimeter gemessen. Die Messspannungen sollen durch den vorgenommenen Messaufbau nicht in den Gain bzw. Offset Error reichen. Eine Simulation dieser Maßnahme wird durch das Programm PSPICE vorgenommen.

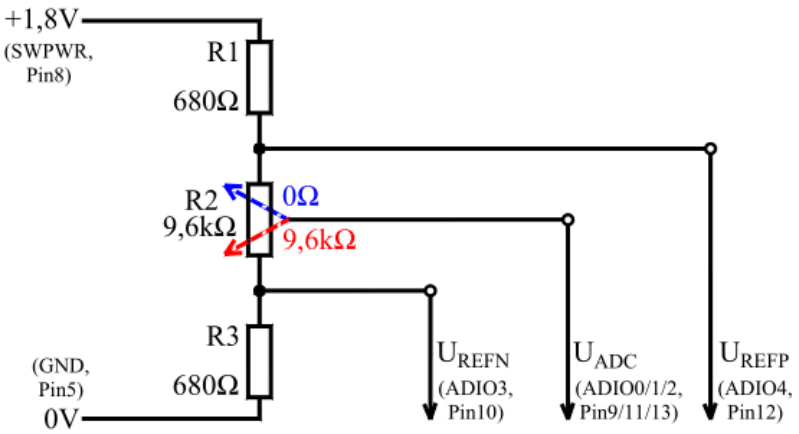


Abbildung 6-1: Messaufbau mit einem Potentiometer und Referenz- bzw. Messspannungen

Bei einer Stellung des Potentiometers von 980 Ω (entspricht einem STM310 Messwert von 0x1A) soll eine LED angeschaltet werden. Bei einem Potentiometerwert von 7,83 kΩ (0xD0) soll die LED wieder ausgeschaltet werden. Außerdem sind Telegramme von anderen Sendern zu verwerfen.

6.1.2 Messversuch mit zusätzlicher Elektronik und einem kapazitiven Sensor

Der zweite Messversuch wird nach **Abbildung 6-2** aufgebaut. Die zusätzliche elektronische Schaltung wurde nicht selbst entwickelt, sondern durch die Anleitung und einen groben Entwurf von Herrn Christian Heinrich (Preh Carconnect) aufgebaut. Das Signal, welches an den ADC0 des STM310 geführt werden soll, wird vor dem

tatsächlichen Messversuch mit einem Oszilloskop geprüft, da ein Defekt des STM310 durch eine Überspannung ausgeschlossen werden soll.

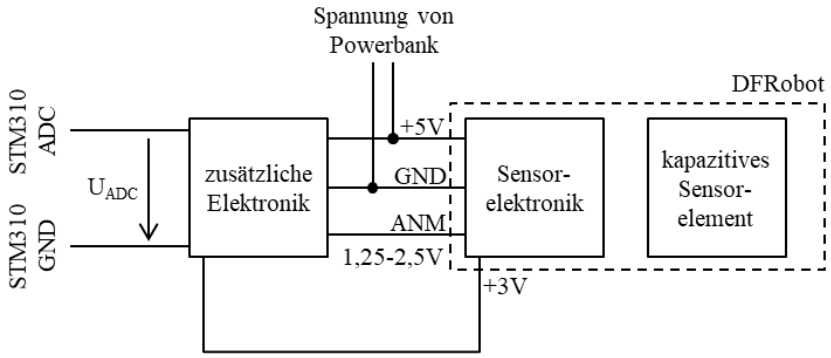


Abbildung 6-2: Messversuch mit kapazitivem Sensor und zusätzlicher Elektronik

Es werden zusätzlich zu den im Abschnitt 4.1.1 benötigten Komponenten folgende Bauteile für die zusätzliche Elektronik verwendet:

- Kondensator: 100 nF,
- Widerstände: zweimal 100 kΩ, 3 kΩ sowie 2,1 kΩ,
- 500 Ω Potentiometer,
- Operationsverstärker (genutzt wird ein OPA237 von Texas Instruments) und
- Erdfeuchtigkeitssensor (genutzt wird einer von DFRobot)

Die zusätzliche elektronische Schaltung ist in Verbindung mit dem Schaltplan des kapazitiven Sensors (siehe **Abbildung 6-3**) entwickelt worden. Das analoge Messsignal (ANM) am Pin 3 vom kapazitiven Sensor soll nach [DFR17] je nach Erdfeuchtigkeit zwischen 0 V und 3 V liefern. Daraus soll ein invertiertes Signal von ca. 0,1 V bis 1,7 V entstehen, da der Sensor bei geringer Feuchtigkeit eine hohe Messspannung ausgibt.

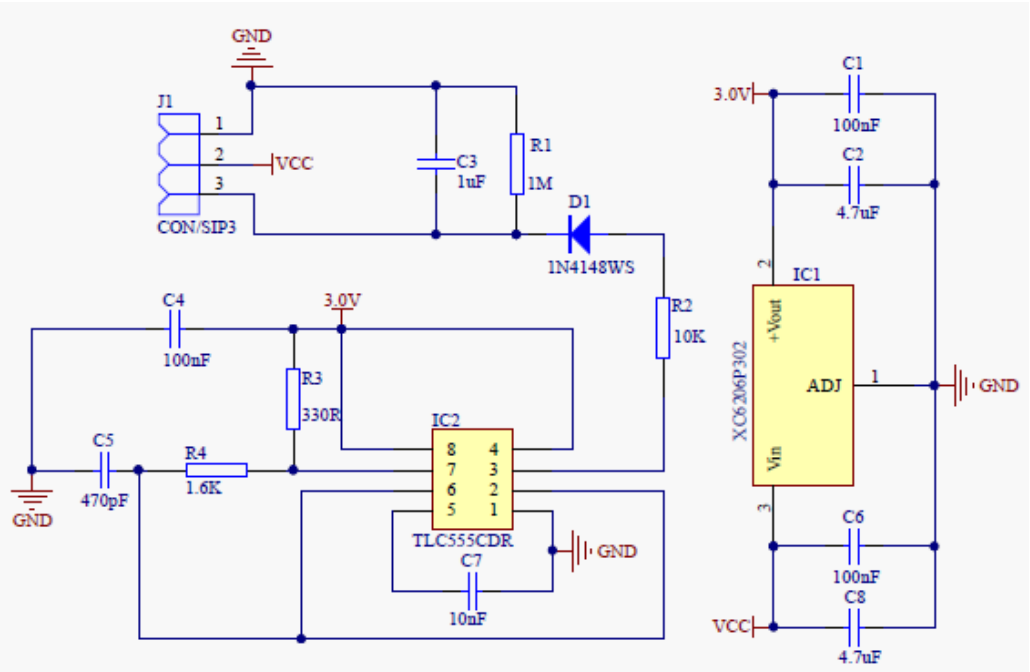


Abbildung 6-3: Sensorelektronik des kapazitiven Erdfeuchtesensors von DFRobot aus [DFR15]

Die drei Leitungen von Pin 1 (GND), Pin 2 (VCC) und Pin 3 (ANM), welche herausgeführt werden, sind für den Betrieb des Sensors notwendig. Die Spannungsversorgung für den Sensor soll 5 V betragen. Dies wird durch eine USB-Powerbank realisiert.

Das Potentiometer R4 in der zusätzlichen elektronischen Schaltung (vgl. **Abbildung 6-4**) dient lediglich zum Einstellen der unteren Messwertgrenze. Die Spannung von +3 V wird aus dem kapazitiven Sensor entnommen.

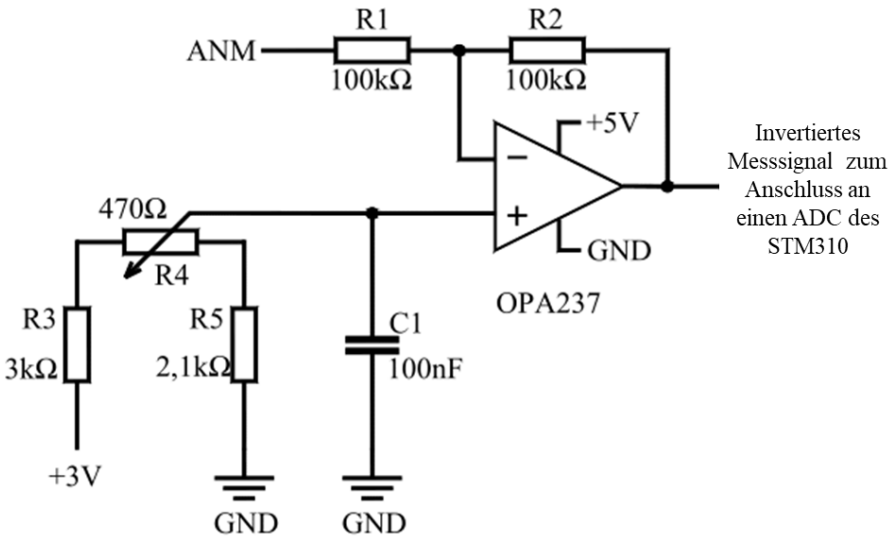


Abbildung 6-4: Entwurf der zusätzlichen elektronischen Schaltung

6.2 Ergebnisauswertung

Die PSPICE Ergebnisse des ersten Messversuchs sind in **Abbildung 6-5** (Potentiometerstellung 0 Ω) und **Abbildung 6-6** (Potentiometerstellung 9,6 kΩ) dargestellt. Ersichtlich sind die positive Referenzspannung U_{REFP} mit 1,688 V und die negative Referenzspannung U_{REFN} von 111,67 mV.

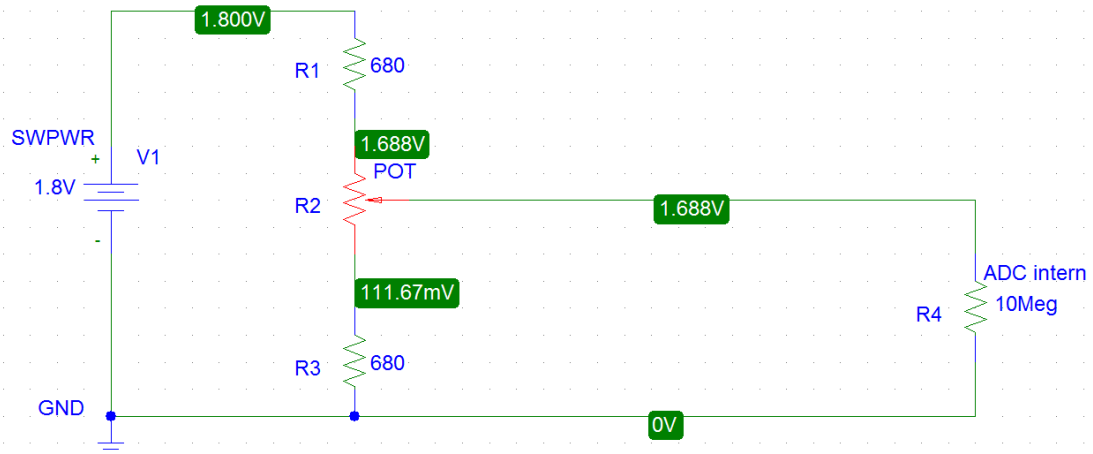


Abbildung 6-5: PSPICE Simulation des Messversuchs mit Potentiometer bei 0 Ω

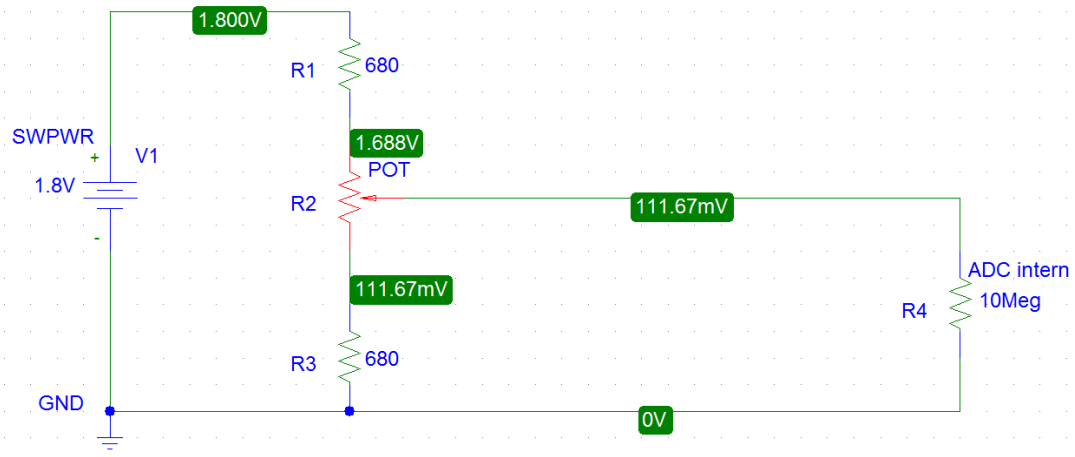


Abbildung 6-6: PSPICE Simulation des Messversuchs mit Potentiometer bei 9,6 kΩ

Die minimal sowie maximal auftretenden Spannungen U_{ADC} sind ebenso ersichtlich. Da diese innerhalb des möglichen Messbereichs liegen, wurde ein Test mit dem STM310 durchgeführt. Die angezeigten Werte des STM310 in Dolphin View Basic, stimmten mit der Einstellung am Potentiometer nahezu überein. In **Tabelle 6-1** wird ein kleiner Überblick mit acht verschiedenen Potentiometerstellungen und den dazugehörigen Empfangsdaten gezeigt.

Tabelle 6-1: Potentiometerstellung mit den dazugehörigen Empfangsdaten des Messversuchs

Potentiometer		**Dolphin View Basic**	
Werte in kΩ	Potentiometerstellung in %	Werte in HEX	Werte in %
9,60	100,0	FF	100,0
8,51	88,6	E1	88,2
6,39	66,5	A6	65,1
5,05	52,6	82	51,0
3,92	40,8	64	39,2
1,90	19,8	2F	18,4
0,66	6,9	0E	5,5
0,00	0,0	00	0,0

Der Empfang der korrekten Sendertelegramme und die Analyse der Daten bzgl. Ein- und Ausschalten der LED, verliefen wie gewollt. Telegramme von fremden Sendern wurden vom Mikrocontroller nicht beachtet und verworfen.

Der Messversuch mit dem kapazitiven Erdfeuchtesensor und einer zusätzlichen Elektronik wurde zuerst mit einem Oszilloskop geprüft. Die analoge Messspannung des Sensors zeigte bei einer Messung in der Luft und bei Raumtemperatur eine Ausgangspannung von 2,5 V. Bei einer Messung in einem mit Wasser gefüllten Behältnis lag die Ausgangsspannung bei 1,25 V. Ausgehend von diesen Werten wurde die in Abbildung 6-2 dargestellte Schaltung eingesetzt. Die von der zusätzlichen Elektronik ausgehende Spannung (U_{ADC}) ist in dem Bereich von ca. 100 mV bei Trockenheit und ca. 500 mV bei Nässe gemessen worden. Diese Messung konnte nicht dauerhaft durchgeführt werden. Die Messwerte veränderten sich nach einer Zeit von ca. 5 s bis 10 s sehr stark. Danach lag der Spannungsoutput (U_{ADC}) größtenteils nur noch bei 0 mV - 50 mV, obwohl das analoge Messsignal des kapazitiven Sensors, durch das

Ändern des Wassergehaltes, unterschiedliche Spannungen von 1,25 V – 2,5 V bereitstellte. Daraus resultiert, dass die entwickelte Schaltung nicht für einen Messversuch einzusetzen ist. Ein möglicher Grund dafür könnte u. U. eine Beschädigung durch zu viel Hitze beim Löten auf eine Platine oder eine fehlerhafte Entwicklung der zusätzlichen Elektronik sein. Messergebnisse sind demnach nicht vorhanden.

7 Zusammenfassung

7.1 Fazit

EnOcean bietet verschiedene Komponenten für verschiedene Aufgabenbereiche an. Um eine Demoapplikation zu entwickeln, ist es wichtig zuerst zu verstehen, welches Produkt zu welchem Zweck eingesetzt wird.

Das Verständnis der genutzten Protokolle und Profile ist essenziell, um die Informationen aus den Datenblättern zu verstehen und nutzen zu können. Für den Funkübertragungsweg wird das ERP verwendet. Dabei ist entscheidend, welche Version eines Moduls genutzt wird. Eine leitungsgebundene Übertragung erfolgt mittels ESP. Welche ESP-Version genutzt wird, ist abhängig vom Transceiver Modul. Das EEP dient einzig und allein zur Interpretation der empfangenen Daten. Dieses muss vor der ersten Datenübertragung mittels Teach-In Telegramm gesendet werden.

Bei der Auswahl der Produkte sind wesentliche Punkte, wie die Stromversorgung und Sende- und Empfangsfähigkeit zu berücksichtigen. Das verwendete STM310 besitzt die Möglichkeit, analoge Spannungen per ADC zu messen. Es existieren außerdem digitale Inputs zum Erfassen von binären Signalen. Das STM310 ist eine veraltete Version und wird von EnOcean nicht mehr unterstützt. Das Prinzip der Datenübertragung und der Einstellungsmöglichkeiten lässt sich jedoch auf alle STMs übertragen. Die mögliche Nutzung der Eingänge variiert je nach Bauweise. Nur das STM300 bietet die Möglichkeit alle verfügbaren Eingänge frei zu nutzen, da keine vorinstallierten Sensoren vorhanden sind. Nachteile des STM300 sind die nicht vorhandene Spannungserzeugung, keine installierte Antenne und wie bei dem STM310 keine Verschlüsselungstechnik.

Ein TCM benötigt eine externe Stromversorgung, da dieses eine höhere Leistung als das STM benötigt. Es existiert die Möglichkeit sowohl zu senden als auch zu empfangen. Das verwendete TCM320 besitzt eine vorinstallierte Steckerleiste, wodurch die Handhabung, z. B. Spannungsversorgung und Anbindung an ein externes Gerät (z. B. Mikrocontroller), erleichtert wird. Die Kommunikation zwischen externem Gerät und dem TCM320 erfolgt über deren UART-Schnittstellen. Je nach Modulausführung wird das ESP2 oder ESP3 zur seriellen Kommunikation genutzt. Außerdem besitzen die TCM-Ausführungen eine Repeater Funktion, wodurch eine Verlängerung der Übertragungsstrecke realisiert werden kann. Das TCM320 verfügt über keine Ver- und Entschlüsselungstechnik.

Der verwendete Mikrocontroller dient ausschließlich der Datenauswertung und kann frei programmiert werden. Der zu verwendende Mikrocontroller kann unter der Voraussetzung, dass eine UART-Schnittstelle vorhanden ist, frei gewählt werden. Außerdem ist mindestens ein digitaler Output, beispielsweise zum Steuern einer LED, nötig.

Für den Empfang der darzustellenden Daten wird ein USB300 Stick verwendet. Der integrierte FTDI Chip sorgt für die Datenübertragung auf einen virtuellen COM-Port.

Die Daten werden durch ein Programm auf Basis von Node.js ausgewertet. Die Anzeige für den Nutzer wird in einem Browser realisiert. Die Darstellung oder Verarbeitung der Daten kann auch über andere Programme, welche auf den COM-Port zugreifen können, erfolgen.

In den durchgeführten Messversuchen konnte nur der Messaufbau mit einem Potentiometer als Sensor erfolgreich getestet werden. Der Messversuch mit einem kapazitiven Erdfeuchtesensor von DFRobot blieb erfolglos. Die nötige Umwandlung des Messsignals von 1,2 V - 2,5 V auf geringere Spannungen von 0,1 V – 1,7 V konnte nur kurzfristig erreicht werden. Die Entwicklung einer passenden Elektronik muss durchgeführt werden, um Messwerte mit einem STM310 aufnehmen zu können. Die Übertragung der Messwerte verlief problemlos. Der TCM320 leitet die Telegramme an den externen Mikrocontroller weiter und dieser schaltet eine LED, je nach Höhe des Messwertes, an bzw. aus.

7.2 Ausblick

Derzeit wird bei einigen Modulen (STM, PTM) das AES128 Verschlüsselungsverfahren eingesetzt. Als Erweiterung dieser Arbeit kann die Verwendung von dieser Verschlüsselungstechnik erforscht werden. Ebenso ist es interessant zu erfahren, ob eine Implementierung von anderen Verschlüsselungsverfahren möglich wäre. Dafür muss vermutlich bei STMs und PTMs eine eigene Firmware geschrieben werden. Da dies sehr kostenintensiv ist, müssen die Vor- und Nachteile genau abgewogen werden. Eine kostengünstigere Variante könnte eine verschlüsselte Kommunikation zwischen TCM und TCM darstellen. Ob zwei TCMs untereinander kommunizieren können, muss vorher ohne Verschlüsselung erprobt werden. Die Ver- und Entschlüsselungstechnik kann auf beiden Seiten durch externe Mikrocontroller realisiert werden.

Da jedes Gerät eine Hersteller- und Geräte-ID besitzt, wäre es interessant zu erfahren, ob die Geräte-ID herstellerunabhängig einzigartig ist. Ist dies nicht der Fall, ist es durchaus denkbar, dass die Daten von Geräten unterschiedlicher Hersteller falsch interpretiert werden. Zumindest mit ERP1 wird keine Hersteller-ID im Datentelegramm, sondern nur im Teach-In Telegramm, versendet.

Die Darstellung der Messergebnisse auf portablen Endgeräten, wie z. B. einem Tablet oder einem Smartphone, wäre ebenfalls eine Erweiterung aufbauend auf dieser Arbeit. Dazu ist es notwendig herauszufinden, mit welchen zusätzlichen Komponenten dies umsetzbar wäre. Eine Kommunikation im 2,4 GHz oder 5 GHz Netzwerk wäre in diesem Zusammenhang durchaus denkbar.

Eine Erweiterung der ATmega8 Software zur Speicherung und Erkennung von mehreren Geräte-IDs und der passenden EEPs, sowie eine Installation eines Teach-In-Incoming Buttons mit dem dazugehörigen Code, sind durchaus denkbar. Der Teach-In-Incoming Button soll verhindern, dass sämtliche Teach-In Telegramme, welche von unerwünschten Sendern empfangen wurden, gespeichert werden.

Literatur- und Quellenverzeichnis

[ANO07] ANONYM: *„Baud“*. URL: https://www.itwissen.info/Baud-baud-Bd.html (veröffentlicht: 04.07.2007) (Stand: 07.08.2018).

[ANO17] ANONYM: *„AVR-GCC-Tutorial/Der UART“*. URL: https://www.mikrocontroller.net/articles/AVR-GCC-Tutorial/Der_UART (veröffentlicht: 28.03.2017) (Stand: 07.08.2018).

[ATM13] Atmel: *„Complete Datasheet: 8-bit Atmel with 8KBytes In-System Programmable Flash – Atmega8, ATmega8L – Rev. 2486AA–AVR–02/2013“*. URL: http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-2486-8-bit-AVR-microcontroller-ATmega8_L_datasheet.pdf (Stand: 09.06.2018).

[BEC14] BECK, A.: *„Programmierung und Beschaltung von Atmel Mikrocontrollern“*. Hochschule für Technik und Wirtschaft Dresden. Fakultät Informatik. Vorlesungs- und Praktikumsunterlagen. 2014.

[DFR15] DFRobot: *„Schematic of Capacitive Soil Moisture Sensor – Version 1.0 – 27.08.2015“*. URL: https://github.com/Arduinolibrary/DFRobot_Capacitive_Soil_Moisture_Sensor/raw/master/SEN0193%20%20Capacitive%20Soil%20Moisture%20SensorV1.0.PDF (Stand: 09.08.2018).

[DFR17] DFRobot: *„Capacitive Soil Moisture Sensor SKU:SEN0193“*. URL: https://www.dfrobot.com/wiki/index.php/Capacitive_Soil_Moisture_Sensor_SKU:SEN0193 (veröffentlicht: 25.05.2017) (Stand: 09.08.2018).

[ENA17] EnOcean Alliance: *„EnOcean Equipment Profiles (EEP) – Version: 2.6.8 – 31.12.2017“*. URL: https://www.enocean-alliance.org/wp-content/uploads/2018/02/EEP268_R3_Feb022018_public.pdf (Stand: 27.06.2018).

[ENO12] EnOcean GmbH: *„User Manual Scavanger Transmitter Module STM 31x / STM 31xC – Version: 1.04 – 10.12.2012“*. URL: https://www.enocean.com/en/enocean_modules/stm-312-user-manual.pdf https://www.enocean.com/en/enocean_modules/stm-331/user-manual-pdf/(Stand: 11.01.2018).

[ENO13] EnOcean GmbH: *„EnOcean Radio Protocol 1 – Version: 1.1 – 25.09.2013“*. URL: https://www.enocean.com/fileadmin/redaktion/pdf/tec_docs/EnOceanRadioProtocol1.pdf (Stand: 21.06.2018).

[ENO14] EnOcean GmbH: *„Dolphin Core Description – Version: 1.1 – 21.07.2014“*. URL: https://www.enocean.com/fileadmin/redaktion/pdf/tec_docs/Dolphin_Core_Description_V1.1.pdf (Stand: 10.08.2018).

[ENO17a] EnOcean GmbH: *„User Manual PTM 330 / PTM 335 / PTM 330U / PTM 430J Sub GHz Transmitter Modules – Version: 1.6 – Januar 2017“*. URL: https://www.enocean.com/en/enocean_modules/ptm-330/user-manual-pdf/ (Stand: 12.07.2018).

[ENO17b] EnOcean GmbH: *„EnOcean Radio Protocol 2 – Version: 1.2 – 20.06.2017“*. URL: https://www.enocean.com/fileadmin/redaktion/pdf/tec_docs/EnOceanRadioProtocol2.pdf (Stand: 02.07.2018).

[ENO17c] EnOcean GmbH: *„User Manual USB 300 / USB 300U / USB 400J USB Gateway for EnOcean Radio – Version: 1.7 – Juni 2017“*. URL: https://www.enocean.com/en/enocean_modules/usb-300-oem/user-manual-pdf/ (Stand: 13.06.2018).

[ENO17d] EnOcean GmbH: *„User Manual Energy Harvesting Temperature Sensor Transmitter Module STM 330 / STM 331 / STM 331U / STM 332U / STM 333U – Version: 1.4 – Juni 2017“*. URL: https://www.enocean.com/en/enocean_modules/stm-331/user-manual-pdf/ (Stand: 12.07.2018).

[ENO17e] EnOcean GmbH: *„User Manual TCM 300 / TCM 320 (868 MHz), TCM 300U / TCM 320U (902 MHz) Transceiver Module – Version: 1.7 – Juli 2017“*. URL: https://www.enocean.com/de/enocean_module/tcm-300/user-manual-pdf/ https://www.enocean.com/en/enocean_modules/stm-331/user-manual-pdf/(Stand: 12.01.2018).

[ENO17f] EnOcean GmbH: *„TCM320 - Funktransceivermodul für programmierbare Systemkomponenten“*. URL: https://www.enocean.com/de/enocean_module/tcm-320/ (Stand: 09.08.2018).

[ENO18] EnOcean GmbH: *„EnOcean Serial Protocol 3 (ESP3) – Version: 1.46 – 29.05.2018“*. URL: https://www.enocean.com/fileadmin/redaktion/pdf/tec_docs/EnOceanSerialProtocol3.pdf (Stand: 01.07.2018).

[ETS04] ETSI TR 102 313 V1.1.1 (2004-07): *„Electromagnetic compatibility and Radio spectrum Matters (ERM); Frequency-agile Generic Short Range Devices using Listen-Before-Transmit (LBT); Technical Report“*. European Telecommunications Standards Institute. 2004.

[ETS16] ETSI EN 300 220-2 V3.1.1 (2016-11): *„Short Range Devices (SRD) operating in the frequency range 25 MHz to 1 000 MHz; Part 2: Harmonised Standard covering the essential requirements of article 3.2 of Directive 2014/53/EU for non specific radio equipment“*. European Telecommunications Standards Institute. 2016.

[GRE06] GREIF, F.: *„Baudraten Tabelle für AVRs“*. URL: http://www.kreatives-chaos.com/artikel/baudraten-tabelle-fuer-avrs (veröffentlicht: 05.10.2006) (Stand: 10.08.2018).

[HER18] HERING, E; SCHÖNFELDER, G. (Hrsg.): *„Sensoren in Wissenschaft und Technik: Funktionsweise und Einsatzgebiete“*. 2., überarbeitete und aktualisierte Auflage. Springer Vieweg. Wiesbaden. 2018. ISBN 978-3-658-12561-5.

[HIP95] HIPPEL, A. (Hrsg.): *„Dielectric Materials and Applications“*. 3. überarbeitete Auflage. Artech House. London. 1995. ISBN 0-89006-805-4.

[KET15] KETELHUT, T: *„Untersuchungen zur Implementierung von Distanzmessungen in Mikrocontroller“*. Hochschule für Technik und Wirtschaft Dresden. Fakultät Informatik. Praktikumsbericht (unveröffentlicht). 2015.

[KUB12] KUBA, M.: *„Automatische Klassifikation von Kommunikationsstandards im europäischen 868 MHz Short Range Device-Band“*. Universitätsbibliothek der Universität Erlangen-Nürnberg. Erlangen. 2012. Universität Erlangen-Nürnberg. Dissertation. URL: http://www.opus.ub.uni-erlangen.de/opus/volltexte/2012/3518/ (Stand: 22.06.2018)

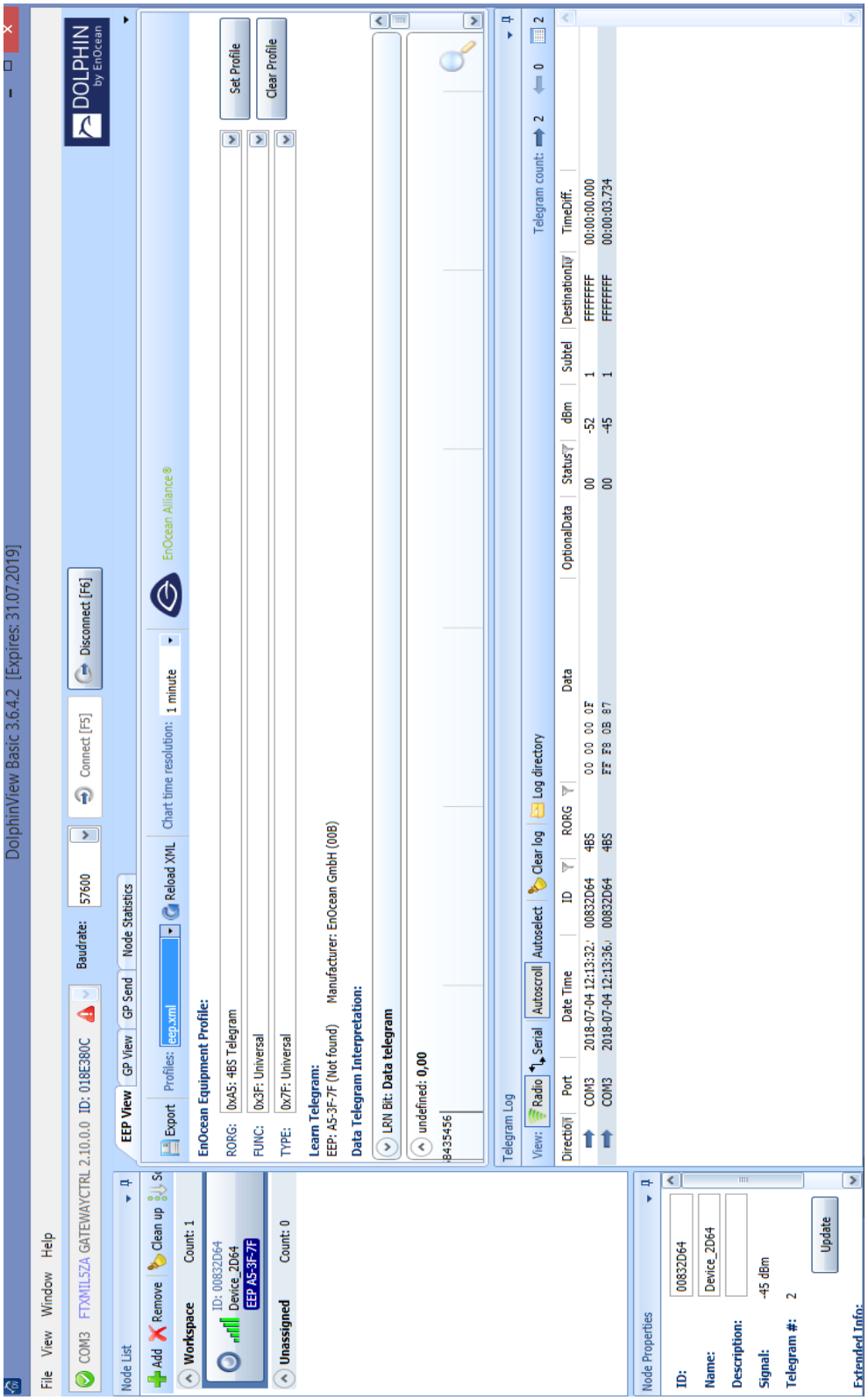
[MAN16] MANDAL, M. K.: *„Setting up Atmel Studio for USBasp and AVR Programming“*. URL: https://blog.manash.me/setting-up-atmel-studio-for-usbasp-and-avr-programming-802bb4dcf1e9 (veröffentlicht: 04.10.2016) (Stand: 08.08.2018).

[STM13] STMicroelectronics: *„LD1117 - Adjustable and fixed low drop positive voltage regulator – Rev 33 – November 2013“*. URL: https://www.st.com/resource/en/datasheet/ld1117.pdf (Stand: 09.08.2018).

[WÜS11] WÜST, K: *„Mikroprozessortechnik: Grundlagen, Architekturen, Schaltungstechnik und Betrieb von Mikroprozessoren und Mikrocontrollern“*. 4., aktualisierte und erweiterte Auflage. Vieweg + Teubner. Wiesbaden. 2011. ISBN 978-3-8348-0906-3.

Anhang

A Dolphin View Basic Oberfläche



B Quellcode des ATmega8 zur Analyse des ESP2-Telegramms

```
#define F_CPU 11059200UL
#define BAUD 9600
#define UBRR ((F_CPU/(16UL*BAUD))-1)
#define RING_SIZE 14      //14 Bytes, because the TCM320 use ESP2
#define UPPERLIMIT 0xD0
#define LOWERLIMIT 0x1A

#include <avr/io.h>
#include <avr/pgmspace.h>
#include <util/delay.h>
#include <avr/interrupt.h>

void delay_ms(unsigned short ms){
    for(ms /= 10; ms > 0; ms--)
    _delay_ms(10);
}

void initUART(){
    /* Setting baud rate */
    UBRRH = (unsigned char) (UBRR>>8);
    UBRRL = (unsigned char) (UBRR&0xff);

    /* RXEN Receive enable / RXCIE Receive complete Interrupt enable */
    UCSRB = (1<<RXEN)|(1<<RXCIE);

    /* Setting data format: 8 data bits, 1 stop bit, no parity bit */
    UCSRC = (1<<URSEL)|(1<<UCSZ1)|(1<<UCSZ0); // 8N1

    // Flush Receive-Buffer
    do
    {
        uint8_t dummy;
        (void) (dummy = UDR);
    }
    while (UCSRA & (1 << RXC));
    sei();
}

volatile unsigned char  buffer[RING_SIZE]; // Ringpuffer
volatile unsigned short irr=0;             // Leseindex
volatile unsigned short irw=0;             // Schreibindex
volatile char count=0;                     // Zaehler

// Interrupt Service Routine: triggered when sign is rdy @ UDR
ISR(USART_RXC_vect)
{
    static unsigned char zeichen=0;

    zeichen=UDR;
    buffer[irw]=zeichen;
    irw+=1;
    if (irw==RING_SIZE)
        irw=0;
    count ++;
}
```

```
// Read out the ringbuffer without blocking
// Return: Sign or -1, if no new sign is ready at the ringbuffer
short int ugetchar(void)
{
    unsigned char c;
    if (count>0)
    {
        cli();
        c=buffer[irr];
        irr++;
        if (irr==RING_SIZE)
            irr=0;
        count --;
        sei();
        return c;
    }
    else return -1;
}

int main(void){
    short int temp;
    char ESPB[RING_SIZE];

    DDRB = 1 << PINB0;

    initUART();

    while(1){
        temp = ugetchar();
        if(temp!=-1){
            if(irr == 0){
                ESPB[RING_SIZE-1]=temp;
                //ID check
                if(ESPB[8] == 0x00 && ESPB[9] == 0x83 && ESPB[10] == 0x2D && ESPB[11] == 0x64){
                    //Teach_In or Data?
                    //ESPB[7]-Bit7 => LRN Type: 1->Teach-In | 0->Data
                    if(ESPB[7] <= 0x8F && ESPB[7] >= 0x80){
                        //Blinking 2 times, if a Teach_In has arrived
                        PORTB &= ~(1 << PINB0);
                        delay_ms(500);
                        PORTB = 1 << PINB0;
                        delay_ms(200);
                        PORTB &= ~(1 << PINB0);
                        delay_ms(200);
                        PORTB = 1 << PINB0;
                        delay_ms(200);
                        PORTB &= ~(1 << PINB0);
                    }
                    else{
                    //ESPB 4-7 Data Bytes
                    if(ESPB[4] < LOWERLIMIT)
                        PORTB = 1 << PINB0;
                    else if (ESPB[4] > UPPERLIMIT)
                        PORTB &= ~(1 << PINB0);
                    }
                }
            }
            else{
                ESPB[irr-1]=temp;
            }
        }
    }
    return 0;
}
```

Thesen zu den wesentlichen Arbeitsergebnissen

1. Ein neuer Sender muss dem Host immer vor der eigentlichen Datenübertragung per Teach-In Telegramm bekannt gemacht werden, um Daten eindeutig identifizieren zu können.
2. Aktuell gibt es keinen passenden kapazitiven Erdfeuchtesensor für die Nutzung mit einem STM310 zu kaufen.
3. Die Verwendung eines TCM320 kann ohne softwareseitige Einstellungen mit vielen Optionen gewährleistet werden.
4. Bei der Verwendung eines STM wird unbedingt ein Programmierboard (z. B. ein EOP350) empfohlen, da Einstellungsänderungen fast ausschließlich über die Software durchzuführen sind.
5. Über die reine Datenübertragung werden keine Informationen bzgl. des Verwendungszwecks mitgesendet. Dies wird über das EEP des zuvor gesendeten Teach-In Telegramm mitgeteilt.
6. Ein externer Mikrocontroller kann in Verbindung mit einem TCM320 mehrere Sender verarbeiten. Daten von verschiedenen Sendern können kombiniert sowie durch das zugehörige EEP ausgewertet werden.
7. Bei nicht signifikanten Änderungen des Messwertes wird die Datenübertragung des STM310 je nach Einstellung unterdrückt, um zusätzlich Energie einzusparen.

Selbständigkeitserklärung

Ich versichere, dass ich die Diplomarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

_____

Torsten Ketelhut

Dresden, 15.08.2018