Entwicklung und testen eines Ultraschall-Entfernungsmessers als Vorbereitung eines Produktentwurfes

---

Projektarbeit

erstellt an der
Fachschule für Technik des Carl-Severing-Berufskollegs
für Metall- und Elektrotechnik der Stadt Bielefeld

Erstellt durch:

Eduard Meiser
Omar Hachimi
Stephan Dannat
FET6A

in Zusammenarbeit mit der Fa. Tinkerforge
betreut durch
Herr Simon

Bielefeld, 20. März 2018

Persönliche Erklärung

Hiermit bestätigen wir, dass die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt wurden. Die Stellen der Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken (dazu zählen auch Internet-quellen) entnommen sind, wurden unter Angabe der Quellen kenntlich gemacht.

Bielefeld,_____

_____

Eduard Meiser

_____

Omar Hachimi

_____

Stephan Dannat

Inhaltsverzeichnis

1 Einleitung

1.1 Lastenheft

1.1.1 Über Tinkerforge

Die Tinkerforge GmbH wurde Ende 2011 mit dem Ziel gegründet, die Handhabung eingebetteter Systeme zu vereinfachen. Das Tinkerforge Baukastensystem besteht aus aktuell fast 80 verschiedenen Modulen, die vom Anwender flexibel für die jeweilige Aufgabe kombiniert werden können. Zu den Modulen zählen diverse Sensor- Aktor- und Schnittstellenmodule, die alle über Hochsprachen wie C#, Python und Java gesteuert werden können. Tinkerforge unterstützt aktuell 17 verschiedene Programmiersprachen. Sowohl Hardware als auch die Software aller Module sind OpenSource. Die Stärke des Baukastensystems ist aus Anwendersicht die enorme Flexibilität, die Einfachheit und die Schnelligkeit mit der Projekte realisiert werden können. Es eignet sich daher besonders im Bereich Rapid Prototyping. Daher findet das Tinkerforge Baukastensystem Anwendung in vielen Forschungsinstituten, in diversen Entwicklungsabteilungen bekannter Automobilhersteller und Ingenieurbüros.

1.1.2 Motivation

Diese Technikerarbeit soll die Grundlage zur Entwicklung eines Entfernungssensors für das Baukastensystem bilden, der auf einer Ultraschall-Entfernungsmessung basiert. Das Baukastensystem verfügt aktuell über so einen Sensor. Bei diesem handelt es sich aber im wesentlichen um ein zugekauftes Modul, welches nicht die gewünschten Leistungen liefert. Daher soll an einem zu entwerfenden Prototypen Forschung betrieben werden, um eine eigene Lösung entwerfen zu können.

1.1.3 Aufgabenbeschreibung

Innerhalb dieser Arbeit soll der Entwurf eines Prototypen des Entfernungssensors und die damit verbundene Forschung durchgeführt werden. Dabei ist durch Recherche zu erarbeiten, welche Möglichkeiten zur Realisierung zur Verfügung stehen. Durch Messungen am Prototypen soll festgestellt werden, welche dieser Möglichkeiten funktional und finanziell realisierbar sind, um ein eigenes Produkt zu erstellen. Sollte im Anschluss noch die Möglichkeit bestehen, sind die Ergebnisse in ein serienreifes Modul umzusetzen.

Diverse Teilaufgaben sind zu erledigen:

- **Recherche**
  Zu Anfang muss recherchiert werden, welche Möglichkeiten es gibt mittels Ultraschall eine Entfernung zu ermitteln und wie diese technisch umgesetzt werden können. Zusätzlich müssen die Techniker sich mit dem Tinkerforge Baukastensystem und seiner internen Funktionsweise vertraut machen.
- **Bauteilauswahl**
  Abhängig von der gewählten technischen Umsetzung müssen geeignete Komponenten ausgewählt werden. Die Auswahl sollte auch unter dem Gesichtspunkten Preis, der Bauteilverfügbarkeit und der technischen Anforderungen erfolgen.

- **Schaltplanentwurf und Layouterstellung**
  Von Tinkerforge wird das Open Source CAD Programm KiCad verwendet. Mit diesem Programm ist ein Schaltplan für den Prototypen und anschließend ein Leiterplattenlayout zu erstellen.

- **Leiterplattenbestückung**
  Die erstellte Leiterplatte wird von Tinkerforge in Auftrag gegeben. Diese muss mit den gewählten Komponenten bestückt werden. Die Tinkerforge GmbH stellt dazu die notwendigen Werkzeuge bereit.

- **Einrichten und Einarbeitung in die Tinkerforge Toolchain**
  Viele Softwarekomponenten werden von der Tinkerforge Toolchain automatisch generiert. Um diese Nutzen zu können muss ein Linux System eingerichtet werden. Anschließend muss sich mit der Funktionsweise des Generators und der Softwareversionsverwaltung"Git"vertraut gemacht werden.

- **Testsoftware und Forschung**
  Um Messungen an der Hardware durchführen zu können gilt es Programmblöcke zu entwerfen, mit denen die einzelnen Funktionen der Baugruppen getestet werden können. So soll ermittelt werden, wie zum einen das Ultraschallsignal effektiv ausgegeben werden kann und wie sich die Signalamplitude auf die Reichweite auswirkt und zum anderen wie das zurückkommende Signal verarbeitet werden kann. Auch soll erarbeitet werden, wie gut das Signal unter verschiedenen Bedingungen verarbeitet werden kann und ob eine zuverlässige Verarbeitungsqualität ohne großen Aufwand realisierbar ist.

1.2 Management des Projektes

1.2.1 Trello

Zur zeitlichen Planung und Übersicht des Ablaufes wurde auf das Onlinetool Trello zurückgegriffen. Dieses ist ein kostenfreies, webbasiertes Projektmanagementtool. Es ermöglicht den Gruppenmitgliedern gleichzeitig von verschiedenen Orten auf die Oberfläche zuzugreifen und Änderungen vorzunehmen. So kann ein Teilnehmer auch neue Termine mit Kennzeichnung der Fälligkeit für andere Gruppenmitglieder einfügen, oder bereits erledigte Aufgaben für alle abhaken. Auch können hier relevante Dokumente, die alle Gruppenmitglieder lesen sollen hochgeladen, und bei Bedarf noch kommentiert werden. Für die Dokumentation lässt sich an diesem System auch einfach abgleichen, zu welchen Zeitpunkten die einzelnen Aufgaben abgeschlossen wurden.

1.2.2 Github

Bei Github handelt es sich um einen webbasierten Online-Dienst, der Server für Entwicklungsprojekte mit einer Versionsverwaltung bereitstellt. So können alle Daten nach einer Änderung im Programm wieder hochgeladen und mit einem Kommentar versehen werden. Sollte nach mehreren Änderungen ein Problem auftreten, kann einfach auf eine ältere Version zurück gegriffen und so der Fehler eingegrenzt werden. Auch kann ein Projekt auf mehrere Abschnitte aufgeteilt werden, damit mehrere Personen unabhängig voneinander daran arbeiten können. Nach der Bearbeitung können diese wieder zusammengefügt werden. Dabei ist erkennbar, welche Änderungen von wem vorgenommen wurden. So können alle Vorgänge jederzeit verfolgt werden, um eine größtmögliche Übersicht zu gewährleisten. Durch das Kommentieren der Änderungen kann die Nachvollziehbarkeit dieser ebenfalls deutlich gesteigert werden. Des weiteren ist diese Plattform gerade für Unternehmen wie Tinkerforge, die ihren Quellcode als Open-Source anbieten besonders reizvoll, da die Nutzer hier schnell an alle Dateien ran kommen.

1.3 Richtlinien zu Erstellung von Schaltplänen und Platinenlayout anhand einem ECAD-Programmpaket

Das Open Source ECAD-Programm KiCAD ist eine Anwendung zum Erstellen von Schaltplänen und elektronischen Leiterplatten.

1.3.1 Schaltplan

Beim Schaltplanentwurf gilt es auf gewisse Regeln zu achten, zu dem sollte die Übersichtlichkeit des Schaltplans nicht außer Acht gelassen werden. So sollten beispielsweise bei Anschluss längerer Leitungen, Kondensatoren zum Ausfiltern eingefangener Funksignale und anderer EMV-Belastungen, angebracht werden. Die Bauteile sollten möglichst so geführt und platziert werden, wie sie später im Layout liegen sollten, wie z.B. gemeinsame Potenzial-Bezugspunkte.
Im Schaltplan wurden die Baugruppen Empfänger, Sender, Hochsetzsteller und die Anschlüsse vom Controller getrennt und so positioniert, dass die beste Übersicht dargestellt wird.
Die wichtigen Grenzpunkte wurden mit einem null Ohm Widerstand bestückt, um die Platine etappenweise in Betrieb nehmen zu können. Somit ist es bei der Erstinitialisierung sicher zustellen das keine Baugruppe eine andere Beschädigen kann.

1.3.2 Platinenlayout

Beim Entwerfen eines Platinenlayouts gibt es viele Möglichkeiten ein Ergebniss zu erzielen. So können alle Bauteile so angeordnet werden, dass alle parallelen Bauteile nebeneinander aufgereiht werden und die in Reihe dazu liegenden Bauteile darunter angeordnet sind. So sähe die Platine zwar ähnlich eines Kontaktplanes aus, allerdings ist diese Variante aus Sicht der EMV nicht sonderlich empfehlenswert.
Eine andere Möglichkeit wäre, die Bauteile im Schaltplan in Gruppen zusammenzulegen und den Schaltplan auf der Platine exakt nachzubilden. Auch bei dieser Variante ergeben sich gelegentlich Probleme, was die Führung der Leitungen und vor allem den Verlauf der Ströme angeht.
So sollte ein Augenmerk auf den stromführenden Leitungen liegen. Je höher der Strom ist, desto breiter und kürzer ist die Leitung auszulegen, um weniger EMV-Störungen zu erzeugen. Auch sollte die Rückführung (GND) günstigerweise als eigene Leiterschicht ausgeführt werden, um einen großen Leiterquerschnitt zu gewährleisten. So kann bei der Rückführung der Ströme auch das Risiko vermieden werden, durch Bildung von größeren Schleifen, Antennen zu erzeugen. Die GND-Schicht sollte so wenig wie möglich unterbrochen werden, vorallem sind Unterbrechungen quer zur Stromflussrichtung zu vermeiden. Zusätzlich ist zu beachten, dass Kondensatoren, zur Verringerung von Störungen(EMV), nahe digitalen Bauteilen angebracht werden sollten. Die optimale Platzierung ist direkt am VDD oder VCC des ICs, so dass die Leiterbahn vom IC mit einem höheren Querschnitt am Kondensator liegt und dann mit leicht verringertem Querschnitt weiter verläuft.

2 Vorbereitung

2.1 Recherche der Funktionsweise

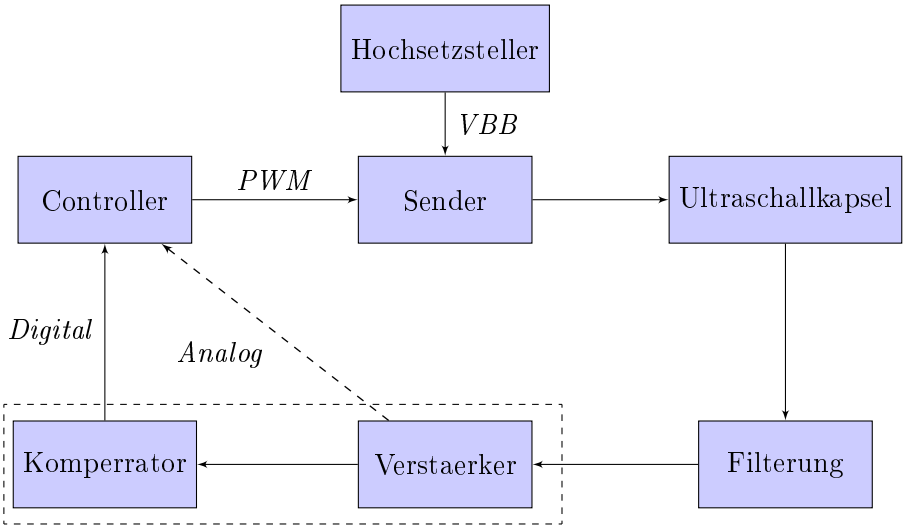

Abbildung 2.1: *Blockschaltbild*

Das Blockschaltbild, siehe Abbildung2.1, zeigt eine vereinfachte Funktionsübersicht. Der Schaltplan befindet in den Anlagen.

2.1.1 Controller

Der Controller soll eine stabile und variable PWM ausgeben können. Es werden für die Zeiterfassung und die PWM Ausgabe mehrere Timer benötigt, die Intern und von der Hardware gesteuert werden können. Ein Analog/Digital Wandler sollte vorhanden sein.

2.1.2 Sender

Der Sender soll Schallwellen im Ultraschallbereich (40 kHz) aussenden, die einen ausreichenden Schalldruck haben, um von Hindernissen, die mehrere Meter entfernt sind, ein Echo erhalten zu können. Dafür muss die Ultraschallkapsel mit einem sinusähnlichen Signal angesteuert werden, dessen Amplitude angemessen hoch ist, um den gewünschten Schalldruck zu erzeugen. Der Nachteil ist, dass der Controller nur auf seine Spannungsebene von 3,3 V begrenzt ist und durch höhere Spannungen zerstört werden würde. Um höhere Spannungen an der Ultraschallkapsel realisieren zu können, muss ein Schalter als weiteres Bauteil dazwischen geschaltet werden. Dieses zusätzliche Bauteil dient als Trennung zwischen den 3,3 V des Mikrocontrollers und der höheren Spannungsebene der Lautsprecherkapsel. Dabei wird darauf zu achten sein, dass der Schalter schnell genug arbeitet, um die 40 kHz auch zuverlässig schalten zu können.

2.1.3 Hochsetzsteller

Der Hochsetzsteller dient dazu, aus den 5 V Versorgungsspannung eine höhere Spannung für den Sendebetrieb zu schaffen. So kann der Schalldruck, der ausgegeben wird erhöht werden (größere Reichweite/größeres Rücksignal) siehe Kaptiel 3.
Die Funktionsweise des Hochsetzstellers (Spannungspumpe/Aufwärtswandler/Aufwärtsregler) ist überschaubar und findet in vielen Bereichen Anwendung. Grundsätzlich wird eine Induktivität in

Reihe mit einer Freilaufdiode vor einen Ladekondensator geschaltet. Dieser liegt parallel zum Ausgang. Zwischen der Spule und der Diode ist ein Schalter angeschlossen, der die Spule gegen Masse schaltet. So lädt sich die Spule bei Betätigung des Schalters auf, wobei durch den Stromfluss ein Magnetfeld entsteht und beim Öffnen steigt die Spannung am sekundären Ende der Spule, durch das zusammenbrechende Magnetfeld, an und lädt den Kondensator auf.Dieser Vorgang wird wiederholt bis der Kondensator so weit aufgeladen ist, dass die Ausgangsspannung den gewünschten Wert hält. Natürlich ist die mögliche Ausgangsspannung nicht unbegrenzt über das Schaltspiel regelbar, sondern ist von den Eigenschaften der Komponenten abhängig.

2.1.4 Ultraschallkapsel

Für das Senden und Empfangen des Ultraschalls wird eine auf Piezomodulen basierende Ultraschallkapsel verwendet, weil Sie die Möglichkeit hat ein Signal auszugeben sowie zu empfangen und in Kleinbauform erhältlich ist. Wie stark das Nachschwingen der Kapsel eine Auswirkung auf die Empfängerschaltung hat sollte überprüft werden, sowie größe der Störfrequenz bei verschiedenen Amplituden auf unterchiedlichen Entfernungen. Es sollte auch getestet werden welche Kapseln sich für einen stand-alone Betrieb am besten eignen.

2.1.5 Filter

Die Filterschaltung soll am Eingang des Empfängers mögliche Störfrequenzen herauszufiltern um eine Verfälschung der gemessenen Strecke zu vermeiden, da die Ultraschallkapsel nicht ausschließlich Ultraschallsignale aufnimmt. Für Hochefrequenzen eignet sich eine passive Hochpassfilter-Schaltung, es wird zu einem Kondensator oder einer Spule ein Widerstand parallel zur Masse bestückt.

2.1.6 Empfänger

Im Empfangsbetrieb werden die zurückkommenden Schallsignale, die auf die Ultraschallkapsel treffen, in sinusförmige Spannungssignale umgewandelt. Die Amplitude hängt von dem Schalldruck der empfangenen Signale ab und ist deutlich niedriger als die Amplitude der gesendeten Signale. Diese Signale müssen anschließend verstärkt werden, um sie mit dem Mikrocontroller auswerten zu können. Zur Vereinfachung der Auswertung wird das eingehende analoge Signal in ein digitales Signal umzuwandeln, um Verarbeitungsaufwand einsparen zu können, denn die Auswertung eines Analogsignals erfordert mehr Programmieraufwand als die Auswertung eines Digitalsignals. Um die Qualität der gemessenen Entfernung besser zu beurteilen sollte eventuell auch das Analogsignal ausgewertet und mit dem digitalen Signal verglichen werden.

2.2 Verwendete sowie Dimensionierung der Bauteile

2.2.1 Controller

Der Infineon XMC 1xxx48 gehört zu der Familie der ARM Cortex -M0 Prozessoren und ist ein 32-bit Industrial Microcontroller und wird mit 48 MHz externer Clock betrieben. Die 48 im Namen des Prozessors steht für die Anzahl der Pins. Der interne Timer läuft mit 96Mhz. Unteranderem besitzt die CCU4 von dem Prozessor 2x4 16 bit Timer. Außerdem bietet der XMC einen 12 bit A/D Wandler, welcher für die Analogmessung eine viel genauere Auflösung bieten kann als ein 8 bit A/D Wandler. Die Betriebsspannung des Prozessors beträgt 3,3 V. Die Auswahl des Controller wurde getroffen, weil der standardmäßig auch schon bei Tinkerforge eingsetz wird und uns für den Prototypen vorgeben wurde. Der Controller bietet eine weitere Besonderheit beim Programmieren, siehe dazu das Unterkapitel x.x

2.2.2 Sender

Um eine Trennung zwischen des CPU Kreises (3,3 V) und des Hochsetzstellers (6 V-20 V) dessen Aufgabe ist eine höhere Spannung an der Ultraschallkapsel zur erzielen, dafür eignete sich das IC A5950 (Voll Brücke). Wie in Abbildung **??** zu sehen ist, kann die H-Brücke eine angeschlossene Last mit der vom Hochsetzstellers erzeugten Spannung versorgen, deren Frequenz über das Signal an der Anschluss Phase der CPU vorgegeben werden an Out1 und Out2 wird das Ausgangssignal abgegriffen. Für die genaue Beschaltung des ICs siehe Anhang Datenblatt "Schimatic A5950".

2.2.3 Hochsetzsteller

Tinkerforge nutzt schon eine Variante eines Hochsetzstellers auf ihren Platinen somit konnte der Aufbau und die Bauteilauswahl nur auf die Variable Ausgangspannungs erweitert werden. Die Standardschaltung, von Tinkerforge, wurde an dem Eingang der Feedbackspannung mit einem Potentiometer (RV1) erweitert, um die gewünschte Variable Spannung zubekommen.
Die Ausgangsspannung wird mit den externen Widerständen RV1, R13 und R2 eingestellt (siehe Grund Schaltung Datenblatt S.11). Ein Wert von ca. $13,3\ K\Omega$ wurde für R2 empfohlen. Mit den Potentiometer (RV1) lässt sich die Ausgangsspannung bei 5 V Eingangsspannung bis 21,5 V einstellen. In der Formel werden die Widerstände RV1 und R13 zum Ersatz widerstand(Re) zusammengefasst.

$$Re = R2 * \left(\frac{Vout}{1,23} - 1\right) \Rightarrow Vout = \left(\frac{Re}{R2} + 1\right) * 1,23$$

2.2.4 Ultraschallkapsel

Für den Prototypen wurden mehrere Kapseln vereschiedener Hersteller bestellt. Dieses geschah um Unterschiede der verschiedenpreisigen Bauteile zu ermitteln und festzustellen, welches Preissegment die nötige Qualität für die vorliegende Anwendung erfüllt.

2.2.5 Filter

Die Filterschaltung wurde mit einem Hochpassfilter (CR Glied) bestückt bestehend aus einem Kodensator (C12) und einem Widerstand (R5) um unerwünschte Signalanteile mit Frequenzen, die unter 40 kHz liegen, zu unterdrücken. Der Widerstand wurde nach der e24 Reihe ausgewählt. Die Kapazität des Kondensators C12 wurde an die Grenzfrequenz von 40 kHz und den Widerstand angepasst.

$$C12 = \frac{1}{2 * pi * fg * R} \Rightarrow \frac{1}{2 * pi * 40\ kHz * 100\ K\Omega} \approx 40\ pF$$

Anhand der Berechnung wurde ein Kondensator mit 39 pF genommen für den Hochpassfilter

2.2.6 Empfänger

Die Abbildung **??** zeigt die Empfängerschaltung. Durch diese Verschaltung von Operationsverstärkern(OPVs) wird das ankommende Sinusförmige Signal verstärkt und in ein digitales Signal umgewandelt. Für die Verstärkung der Amplitude so wie der Umwandlung des analogen Signals in ein Rechtecksignal mit 40 KHz standen zwei Operationsverstärker zur Auswahl, LT1112 mit einem Stückpreis von 4,80 € und den TLC272 mit einem Stückpreis von 0,88 € trotz der besseren Performanz, wurde der TLC272 für die Prototypen ausgewählt um die preislich günstigen Möglichkeiten zu prüfen. Die Versorgungsspannung der OPV's von 3,3 V wird durch den Kondensator C16 (EMV Störfilter) stabilisiert.
Für die Verstärkung der Amplitude ist der Operationsverstärker TLC272 U2B als nicht invertierender Verstärker geschaltet. Wenn eine Gleichspannung anliegt, wirkt der Kondensator (C10) in der Operationsverstärkerschaltung als Impedanzwandler, also mit einer Verstärkung von eins, geht nun die Eingangsfrequenz hoch, nimmt der Widerstand des Kondensators (C10) ab, somit beginnt der Operationsverstärker auch zu verstärken, und zwar mit zunehmender Verstärkung, bis irgendwann

die Impedanz des Kondensators vernachlässigt werden kann und die Verstärkung nur noch durch das Verhältnis der Widerstände beeinflusst wird, R6 ist zudem notwendig um das schwingen der Amplitude zu verhindern, somit kann die Verstärkung mit folgender Formel berechnet werden:
$Vu = \frac{R6 + R8 + R12}{R6}$

Für die Umwandlung des Analogen Signales in ein Digitales wurde der Operationsverstärker TLC272 U2C als Komparator geschaltet. Beim Auftreten von Differenzen zwischen den eingangs Signalen, wechselt der Ausgang des Komparators zwischen Low (0 Volt) auf High (3,3 Volt).
Die Referenz (*Uref*). Spannung wird durch den Spannungsteiler R9 und R8 bestimmt.
$Uref = \frac{Uges * R9}{R8 + R9} \Rightarrow \frac{3,3\ V * 120\ K\Omega}{100\ K\Omega + 120\ K\Omega} = 1,8\ V$

2.2.7 Besonderheit der Software

Um die die Lesbarkeit des Programms zu fördern wurden APIs (Application Programming Interface) verwendet die schon das passende Register ansteuern anstatt per Hand aus dem Reference Manuel jedes einzelne Register rauszusuchen. Das erlaubt Programme leichter zuwarten und zu optimieren. Die APIs lassen sich in der XMC Lib finden, dies ist eine Bibilothek die der Hersteller zuverfügung stellt. Um .mode zu bearbeiten und nach der erforderten Funktion zu konfigurieren reicht es die *xmc1-gpio.h* zu betrachten. Somit reicht es in das Feld .mode=gewünschte Funktion ein zu eintragen, in diesem Beispiel ist es *XMC GPIO MODE NPUT PULL UP*. Siehe2.2 : Die API für ein Programmbeispiel anhand einer PULL-UP Initialisierung.

```
Code vom Infineon Prozessor XMC 1xxx48 mit der API:
void pin_in_pullup_init(XMC_GPIO_PORT_t *const port, const uint8_t pin)
{
  const XMC_GPIO_CONFIG_t pin_in_config = {
        .mode             = XMC_GPIO_MODE_INPUT_PULL_UP, /**< Defines the direction and characteristics of a pin */
        .output_level     = XMC_GPIO_OUTPUT_LEVEL_HIGH, /**< Defines output level of a pin */
        .input_hysteresis = XMC_GPIO_INPUT_HYSTERESIS_STANDARD /**< Defines input pad hysteresis of a pin */
      };

  XMC_GPIO_Init(port, pin, &pin_in_config);
/*
 * Initializes input / output mode settings like, pull up / pull down devices,push pull /open drain, and pad driver mode.
 * Also configures alternate function outputs and clears hardware port control for selected \a port and \a pin .
 * It configures hardware registers Pn_IOCR,Pn_OUT,Pn_OMR,Pn_PDISC and Pn_PDR.\n
*/
}
```

Abbildung 2.2: *Die API für ein Programmbeispiel anhand einer PULL-UP Initialisierung*

In der2.3 Teilausschnitt von der GPIO Init in der xmc1 gpio.c Datei, veranschaulicht gut was hinter API passiert. Es erfordert viel Einarbeitung in den Prozessor um die einzelnen Funktionsaufrufe zu verstehen, somit erleichter die API die Programmierarbeit und ist zeitgleich Anwenderfreundlich.

```
1  void XMC_GPIO_Init(XMC_GPIO_PORT_t *const port, const uint8_t pin, const
       XMC_GPIO_CONFIG_t *const config)
2  {
3    XMC_ASSERT("XMC_GPIO_Init: Invalid port", XMC_GPIO_CHECK_PORT(port));
4    XMC_ASSERT("XMC_GPIO_Init: Invalid mode", XMC_GPIO_IsModeValid(config->mode));
5    XMC_ASSERT("XMC_GPIO_Init: Invalid input hysteresis",
       XMC_GPIO_CHECK_INPUT_HYSTERESIS(config->input_hysteresis));
6
7    /* Switch to input */
8    port->IOCR[pin >> 2U] &= ~(uint32_t)((uint32_t)PORT_IOCR_PC_Msk << (
       PORT_IOCR_PC_Size * (pin & 0x3U)));
9
10   /* HW port control is disabled */
11   port->HWSEL &= ~(uint32_t)((uint32_t)PORT_HWSEL_Msk << ((uint32_t)pin << 1U));
12
13   /* Set input hysteresis */
14   port->PHCR[(uint32_t)pin >> 3U] &= ~(uint32_t)((uint32_t)PORT_PHCR_Msk << ((
       uint32_t)PORT_PHCR_Size * ((uint32_t)pin & 0x7U)));
15   port->PHCR[(uint32_t)pin >> 3U] |= (uint32_t)config->input_hysteresis << ((
       uint32_t)PORT_PHCR_Size * ((uint32_t)pin & 0x7U));
16 .
17 .
18 .
```

Abbildung 2.3: *Teilausschnitt von der GPIO Init in der xmc1 gpio.c Datei*

xmc1-gpio.c sowie xmc1-gpio.h lassen sich in der XMC Libary im Anhang finden.

2.3 Entwicklung der Software zum Betrieb des Prototypen

2.3.1 Benötigte Kenntnisse

Die Variable x=0-3 dient als Index.
Um das Programm zu erstellen sollten Kenntnisse z.B. für die Timer CCU4x sowie deren Slices CC40-43 vorhanden sein. Die Funktionen der CCU4x, die Capture Compare Unite ist die Timer/-Zählereinheit des Mikrocontrollers, wird in der Reference Manuel beschrieben, siehe dazu Anhang. Um die Timer zu Initialisieren wird die XMC Lib benötigt die eine fertige API mit bringt, siehe Kapitel Besonderheiten der Software, so wird nur noch die API mit den richtigigen Parametern beschrieben. Der Werte der Register mit denen die Zeit gemessen wird müssen sofort zum beginn der Interrupt Service Routine zwischen gespeichert werden damit sichergestelt wird das nicht die Zeit zur Berechung mit gemessen wird.

Durchgeführte Berechnungen

Auch für die Programmierung waren diverse Berechnungen notwendig. So zum Beispiel musste zur Erzeugung der Ultraschallimpulse ein Pulsweitenmoduliertes Rechteck signal geschaffen werden. Dafür wurde ein Timer der CCU4x auf einen Takt von 40kHz eingestellt. So musste bei einem Timertakt von 96MHz eine Periodendauer von 2400 Takten konfiguriert sein und ein Compare-Wert von 1200 Takten, siehe unten die Berechnung. Im Zählvorgang des Timers wird der Ausgang nach erreichen des Compare-Wertes auf 1 gesetzt, und nach erreichen der Periodendauer wieder auf 0 zurückgesetzt. Dadurch ergibt sich eine Periodendauer von 25us , was einer Frequenz von 40kHz entspricht.
Auch zur Erfassung der Zeit, die vergeht bis das Echo des Ultraschall-Impulses zurück kommt wird über einen Timer der CCU4x erfasst.

$$Periodendauer = \frac{96MHz}{40kHz} = 2400 \qquad Compare - Wert = \frac{2400}{2} = 1200$$

Siehe Kapitel Messungen am Ausgang des Controller für das PWM Signal sowie die Einstellung der einzelnen Timer, Anhang PWM Configuration sowie Timer Configuration.

2.3.2 Quellcodeentwurf

Programmstruktur: Anstatt alles in der Distance US main.c ,siehe Abbildung 2.4, an Programmcode zu verfassen was bei sehr komplexen Programmen schnell zu Unübersichtlichkeit führt hat das Auslagern den Vorteil das der Quellcode Logisch getrennt werden kann und so einer verschlankerung des Codes mit sich bringt. Somit stehen in der Main vor allem die Aufrufe der verschiedenen benötigten Funktionen. So sieht man in der Main jetzt deutlich, welche Funktionen beim Starten initialisiert werden, und welche Unterprogramme regelmäßig aufgerufen werden. Auch vereinfacht diese Struktur gerade bei Prototypen das Testen der Funktion, so kann im Falle einerfehlerhaften Funktion einfach der Aufruf auskommentiert werden um zu testen, ob der Fehler wirklich von der Funktion herrührt. Dadurch müssen nicht etliche Zeilen Programmcode der Funktion auskommentiert werden, wodurch schnell Fehler entstehen könnten, durch übriggebliebene Zeichen, oder gar beim entfernen der Auskommentierung gelöschte Zeichen.

```
#include <stdio.h>
#include <stdbool.h>
#include "bricklib2/logging/logging.h"
#include "bricklib2/bootloader/bootloader.h"
#include "communication.h"
/****Eigene Include Dateien*******/
#include "configs/config.h"
#include "system_timer/system_timer.h"
#include "a16pt.h"
int main(void)
{
  logging_init();
  logd("Start Distance US V2 Bricklet/n/r");     //For the Debugmodus
  communication_init();            //Function call
  a16pt_init();                   //Function call
  while(true)
  {
    a16pt_tick();               //Function call
    bootloader_tick();            //Function call
    communication_tick();           //Function call

  }
}
```

Abbildung 2.4: *Die main.c des Distance US*

Um die FUnktionsaufrufe zu verstehen muss die Abbildung 2.5: config a16pt.h näher betrachtet werden. In der der a16pt.h werden die Funktionen definiert die dann in der main.c aufgerufen werden und die Funktionsanweisungen stehen dafür in der a16pt.c. 2.6

```
#ifndef A16PT_H
#define A16PT_H
#include <stdint.h>
void a16pt_init(void);          //Functional definition
void a16pt_tick(void);        //Functional definition
uint16_t a16pt_get_distance(void); //Functional definition
#endif
```

Abbildung 2.5: *config a16pt.h*

Init Aufruf: Auch in der 2.6void a16pt-init(void) gibt es weitere Funktionsaufrufe wo z.B. die PWM oder der Externe Interrupt Initialisiert werden.

```
void a16pt_init(void)
{

/*****************Externe_Interrupt******************/

  eru_init(eru_port);

/************PWM_Init*****************************/

  XMC_CCU4_Init(CCU41, XMC_CCU4_SLICE_MCMS_ACTION_TRANSFER_PR_CR_PCMP);
  XMC_CCU4_StartPrescaler(CCU41);

  ccu4_pwm_init(pwm_port_0,cc40, period_1); //P4_4
  ccu4_pwm_set_duty_cycle( cc40, compare_1);

  ccu4_pwm_init(pwm_port_1,cc42, period_0); //P4_6
  ccu4_pwm_set_duty_cycle( cc42, compare_0);
.
.
.
.
```

Abbildung 2.6: *Ein Teilausschnitt von der a16pt.c mit dem Init Aufruf*

Interrupt Aufruf: In der Abbildung 2.7 :a16pt.c Interrupt Aufruf, werden die für die Entfernungsmessung notwendigen Funktionen und die Interrupt anweisungen, in dem Fall die IRQ21, abgearbeitet außerdem werden die Timer Synchron abgeschaltet und aus experementellen gründen wurde ein weitere Impuls generiert um zu beobachten wie sich das nachschwingen verhält bei einer längeren Kurzschlusszeit an der Ultraschallkapsell. Die IRQ wird auch als Interrupt bezeichnet und wird von der Hardware oder von der Software ausgelöst.

```
1
2  /************Interrupt_Funktionen***************/
3
4  void IRQ_Hdlr_21(void) // Compare Interrupt counter 10
5  {
6
7    // Disable IRQs so we can't be interrupted
8    __disable_irq();
9
10   // Set CCU trigger to low, otherwise ccu counter is restarted
11   XMC_SCU_SetCcuTriggerLow(XMC_SCU_CCU_TRIGGER_CCU41);
12
13   // Stop slice 2
14   XMC_CCU4_SLICE_StopClearTimer(CCU41_CC40);
15
16   // For slice 1 we wait until PWM is run through (to get exactly 10 pwm peaks on
       P4_4 and P4_6)
17   while(XMC_CCU4_SLICE_GetTimerValue(CCU41_CC42) > compare_1) {
18
19     __NOP();
20   }
21
22   //new pin configuration
23   const XMC_GPIO_CONFIG_t pin_out_config  = {
24       .mode                 = XMC_GPIO_MODE_OUTPUT_PUSH_PULL,
25       .output_level         = XMC_GPIO_OUTPUT_LEVEL_HIGH,
26     };
27
28    XMC_GPIO_Init(P4_6, &pin_out_config);
29   //Creat a high impulse
30   for(s=0; s<50; s++)
31     {
32       __NOP();
33     }
34   // Stop slice 0
35   XMC_CCU4_SLICE_StopClearTimer(CCU41_CC42);
36
37   //pin configuration back to the PWM-Mode
38   const XMC_GPIO_CONFIG_t gpio_out_config1  = {
39     .mode                 = XMC_GPIO_MODE_OUTPUT_PUSH_PULL_ALT9,
40     .input_hysteresis     = XMC_GPIO_INPUT_HYSTERESIS_STANDARD,
41     .output_level         = XMC_GPIO_OUTPUT_LEVEL_LOW,
42   };
43
44   XMC_GPIO_Init(P4_6, &gpio_out_config1);
45   // Enable IRQs again
46   __enable_irq();
47
48
49 }
```

Abbildung 2.7: *Ein Teilausschnitt von der a16pt.c mit dem Interrupt Aufruf*

3 Messungen und Auswertung der Ergebnisse

Für die Messungen wurden Laufe des Projekts zwei Versionen an Prototyp-Platinen entworfen, an denen Messungen und Verbesserungen vorgenommen wurden.

3.1 Prototyp 1

Bei dem ersten Prototyp wurden die Sendereinheit und die Empfängereinheit auf getrennten Platinen aufgebaut. So bestand die Möglichkeit, den Senderkreis und den Empfängerkreis getrennt zu untersuchen, ohne dass sich elektrische Signale der beiden Schaltkreise überlagern konnten.

3.1.1 Senderkreis

Zu erst wurden Signale direkt an der CPU gemessen, um sicher zu stellen, dass die Einstellungen im Programm auch die gewünschten Ausgaben zur Folge haben, und keine Gefärdung der Bauteile entsteht. Um das Signal für die Entfernungsmessung zu generieren wurde der Mikrocontroller so programmiert, dass zehn Impulse mit einer Frequenz von 40kHz ausgegeben werden. Danach erfolgt eine Pause, um das zurückkehrende Signal abzuwarten und auszuwerten.

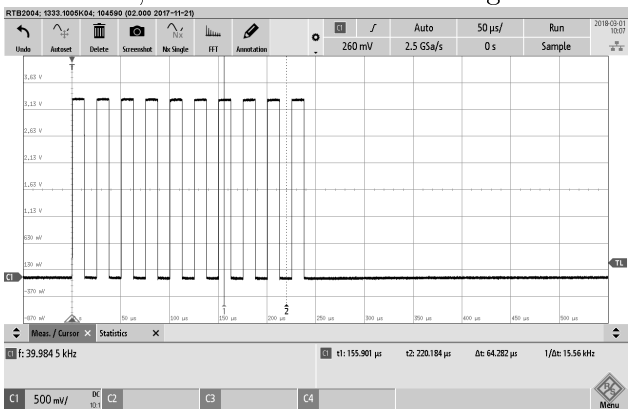

Abbildung 3.1: *PWM-Burst auf 40kHz Basis an der CPU*

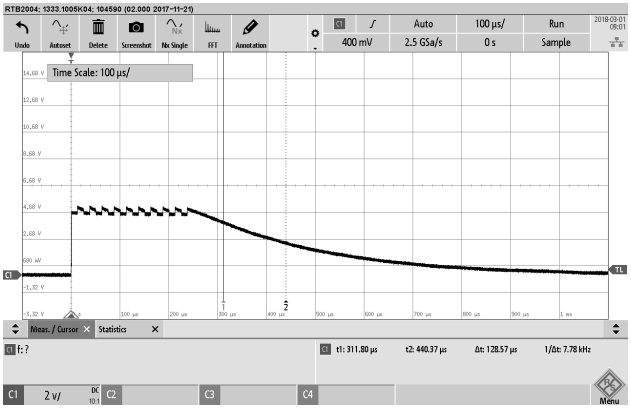

Abbildung 3.2: *PWM Ausgabe über einen Hi-Side*

In der Abbildung 3.1 ist zu sehen, dass der gewünschte Burst aus zehn Impulsen mit einer Periodendauer von jeweils 25us vom Mikrocontroller generiert wurde. Diese Messung wurde auch vorgenommen, um zu überprüfen, wie sich das Signal durch die eingesetzten Bauteile verändert.
Die Abbildung 3.2 zeigt, wie das Ausgangssignal nach einer Hi-Side aussieht. So wird zwar im Takt des PWM-Signals geschaltet, allerdings fehlt es an einem Gegenpool, um das Potential in den Schaltpausen wieder auf Null zu ziehen. Dadurch bleibt die Spannung während des Schaltens immer auf einem erhöhten Pegel und sinkt erst nach Ende des PWM-Signals langsam ab. Dadurch kann natürlich keine vernünftige Ausgabe am Lautsprecher erzeugt werden, denn ohne deutliche Potentialunterschiede kann dieser auch nicht in Schwingungen versetzt werden. Der ausgegebene Schalldruck würde maximal für kürzeste Entfernungsmessungen reichen, wenn überhaupt und dann würde das zurückkommende Signal noch von der abklingenden Spannung des Hi-Side überlagert. Somit ist dieser Aufbau nicht operabel.
Um die Spannung nicht nur auf einen Hi-Pegel, sondern auch auf einen LOW-Pegel schalten zu können wurde danach auf eine Halbbrücke gewechselt. Mit dieser lässt sich der Ausgang, über zwei durch das PWM-Signal gesteuerte MOSFETs, sauber auf Hi- oder LOW-Pegel schalten. Mit der verwendeten Halbbrücke ergab sich die Abbildung 3.3

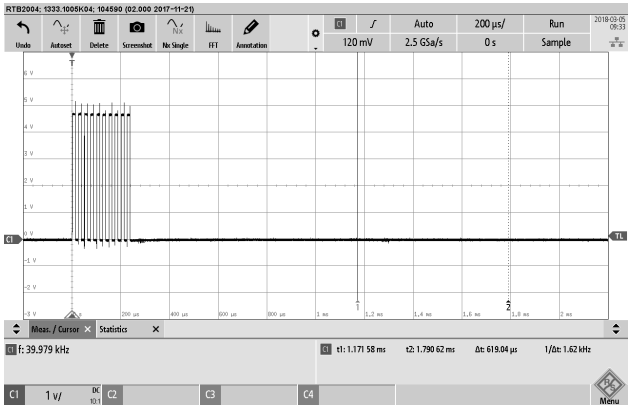

Abbildung 3.3: *PWM Ausgabe über eine Halbbrucke*

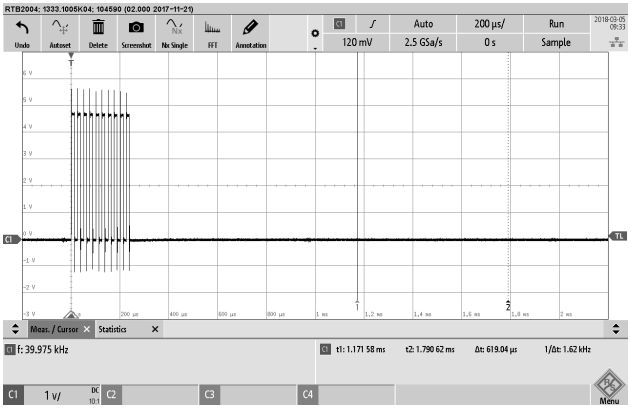

Abbildung 3.4: *Ausgabe der PWM an der Ultraschallkapsel*

Es zeigt sich, dass das Signal nach der Erweiterung auf eine Halbbrücke wieder wie das von der CPU ausgegebene PWM-Signal 3.1 aussieht, nur dass die Amplitude wie geplant höher ausfällt. Somit kann die Höhe der Amplitude über die Spannungspumpe variiert werden um die Stärke des ausgegebenen Signals zu verändern, ohne die CPU durch die höhere Spannung zu beschädigen. Wie in der Abbildung 3.4 zu entnehmen ist, entstehen durch die angeschlossene Ultraschallkapsel höhere Spannungsimpulse in den Schaltmomenten. Diese Spannungsspitzen, die durch die Ultraschallkapsel entstehen, wurden in den Versuchen vernachlässigt, da keine Gefährdung anderer Bauteile entstand. Das somit generierte Ausgangssignal entsprach den Anforderungen und musste für die Versuche nicht weiter bearbeitet werden.

3.1.2 Empfängerkreis

Die Platinen des Sender- und Empfängerkreises wurden gemeinsam auf einer Halterung montiert, so dass die Ultraschallkapseln zum senden und empfangen der Signale nebeneinander befestigt werden konnten. Ziel war es, durch verschieben eines Hindernisses die Signaländerungen an den Platinen beobachten zu können, ohne die gesamten Messaufbauten bewegen zu müssen. Bei der Aufnahme der Messungen wurde der Empfängerkreis Schritt für Schritt überprüft um zu erfahren, wie sich das empfangene Signal durch die einzelnen Bauteile verändert.

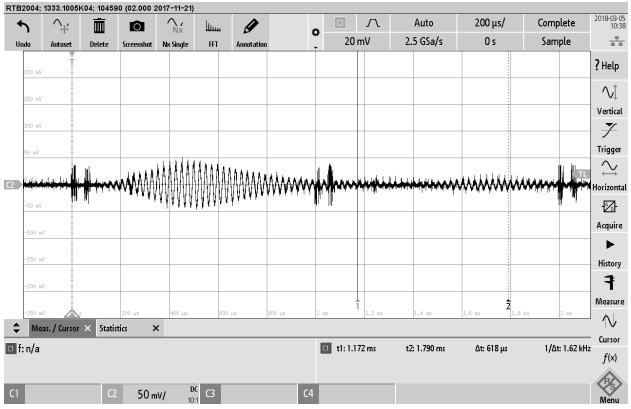

Abbildung 3.5: *Signal Empfang*

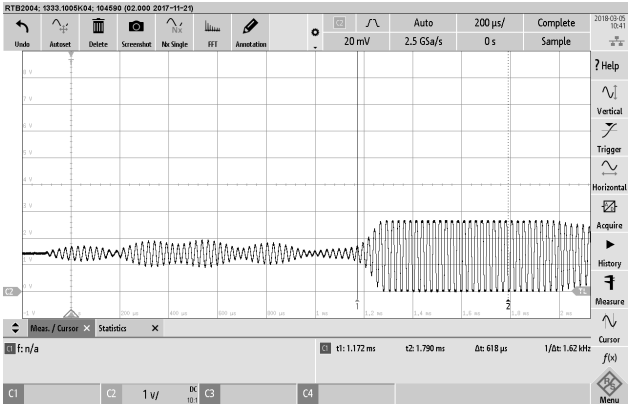

Abbildung 3.6: *Signal nach Verstärkung*

Die Abbildung 3.5 zeigt das Signal, das direkt am Empfänger zu messen war. Hier sind verschiedene vorerst nicht zuordenbare Signale zu sehen. Allein aus diesem Bild lässt sich daher keine Aussage zu den einzelnen Signalen machen. Fest steht nur, dass ebenfalls Signale die nicht der gewünschten Frequenz entsprechen, vom Empfänger aufgenommen werden. Dies gilt es natürlich schnellst möglich auszumerzen, um unerwünschte Störungen zu vermeiden. Die Abbildungen 3.6 und 3.7 zeigen den Verlauf des Signals nach der Filterung und Verstärkung in zwei verschiedenen Zeitauflösungen. Dabei entspricht 3.6 den ersten drei Kästchen von 3.7 und dient um darzustellen, dass die Verstärkung eine maximale Aussteuerung von 3,3V nicht überschreitet.

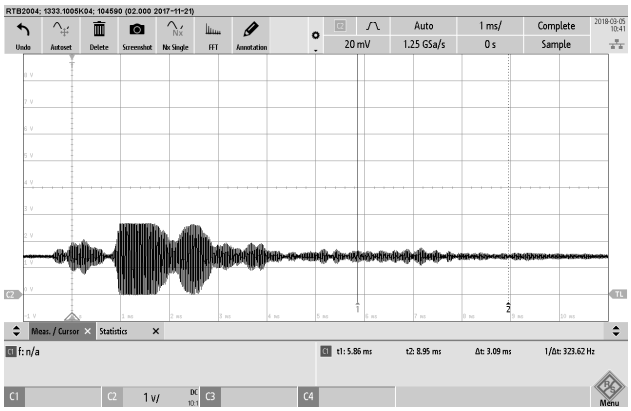

Abbildung 3.7: *Signal nach Verstärkung2*

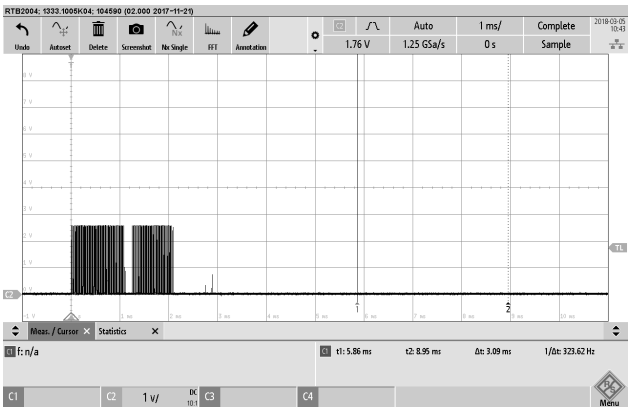

Abbildung 3.8: *Signal nach Komparator*

Nach dem das Signal den Komparator passiert hat, ergibt sich das Bild wie in Abbildung 3.8 zu sehen ist. Bei einem Vergleich mit dem Signal nach der Verstärkung 3.7 wird sichtbar, dass der Komparator nur Signale, die über seinem Schwellwert liegen, durchschaltet. Die Aufteilung in zwei Signalblöcke in den Abbildungen kommt daher, dass der erste Block das Signal der Sender-Kapsel ist, das direkt beim Senden seitlich auf die Empfänger-Kapsel abgestrahlt wurde. Der zweite Block ist bereits das Echo, das vom 20cm entfernten Hindernis zurückgeworfen wurde.
Anhand dieser Ergebnisse kann festgehalten werden, dass die einfachere Version des Ultraschall-Entfernungsmessers durchaus simpel umzusetzen ist. Es fehlt nur noch ein Programm, das die Zeit, die bis zum eintreffen des Echo-Signals in einen Abstand vom Hindernis konvertiert. Besagtes Programm wurde nicht für diese Prototypversion erstellt, da der Anspruch bestand, den Betrieb nicht nur über eine Platine, sondern auch noch über eine Ultraschallkapsel ablaufen zu lassen.

3.2 Prototyp 2

Bei der zweiten Prototyp-Version wurden der Sender- und der Empfängerkreis auf einer Platine aufgebaut und es wurde nur noch eine Ultraschallkapsel für beide Anwendungen vorgesehen. Dadurch bestand die Notwendigkeit, die Halbbrücke der ersten Prototyp-Version durch eine voll gesteuerte Halbbrücke zu ersetzen. Dies geschah, weil für den gewünschten Betrieb drei und nicht zwei Schaltzustände benötigt wurden. Nicht nur sollte die Ultraschallkapsel mit HI-, oder LOW-Signal ansteuerbar sein, auch ein dritter Potentialfreier Zustand war nötig, damit die Echo-Signale auch empfangen werden können. Um bei diesem Aufbau, einen fehlerfreien Betrieb der verwendeten voll gesteuerten Halbbrücke sicherzustellen, wurden durch den Mikrocontroller zwei getrennte PWM-Signale generiert, die wie in Abbildung 3.9 zu sehen ist, durch Lücken getrennt sind. So ist sichergestellt, dass auch trotz Verzögerungen im Schaltbetrieb der Halbleiter, keine Kurzschlüsse entstehen können.

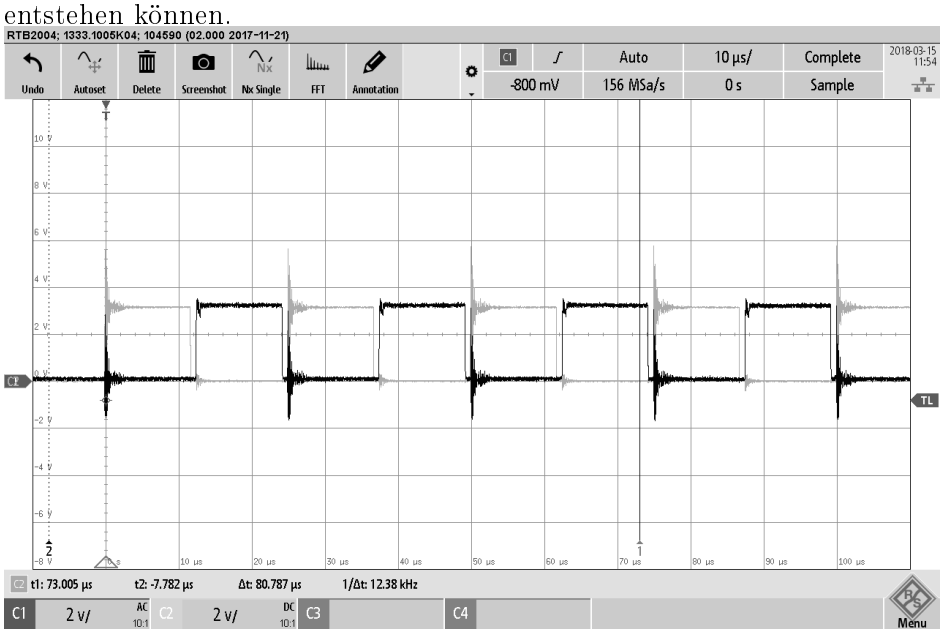

Abbildung 3.9: *Verlauf der zwei generierten PWMs für den Betrieb der voll gesteuerte Halbbrücke*

Nachdem dieser Betrieb sichergestellt war, wurden Messungen am Verstärker (obere Linie), und am Komparator (untere Linie) vorgenommen. Dabei wurde die Verstärkung so eingestellt, dass unerwünschte Störungen gerade so nicht vom Komparator weitergegeben wurden. Die Spannung für den Sendebetrieb wurde für die Versuche zwischen 5V und 20V variiert, um betrachten zu können, wie sich das auf die Reichweite und Genauigkeit der Messungen auswirkt. Als Hindernis wurde bei allen Versuchen eine glatte Holzplatte der Maße 50x64cm verwendet und in einem Abstand von ein bis fünf Metern von der Ultraschallkapsel entfernt aufgestellt. In den Abbildungen 3.10 bis 3.13 sind die Ergebnisse einer Messreihe mit einer Spannung von 5V für den Sendebetrieb dargestellt. Die Ansicht wurde so eingestellt, dass zwei Sendeimpulse zu sehen sind. Dadurch wird deutlicher, welches die Sende Impulse sind, und welches die von der Entfernung abhängigen Echos sind.

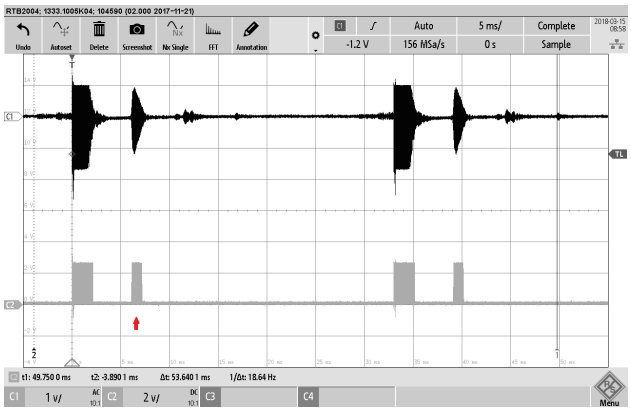

Abbildung 3.10: *Signalverlauf bei 5V auf 1m Abstand*

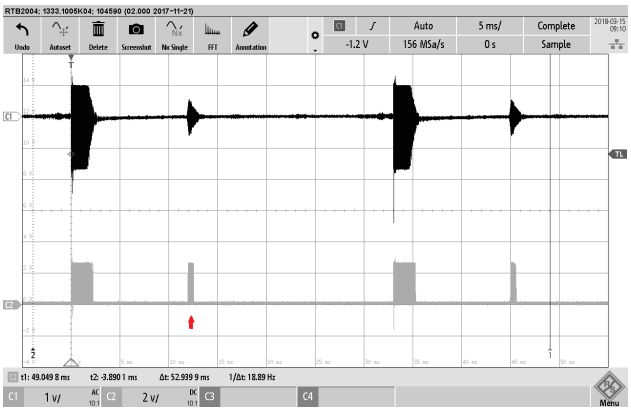

Abbildung 3.11: *Signalverlauf bei 5V auf 2m Abstand*

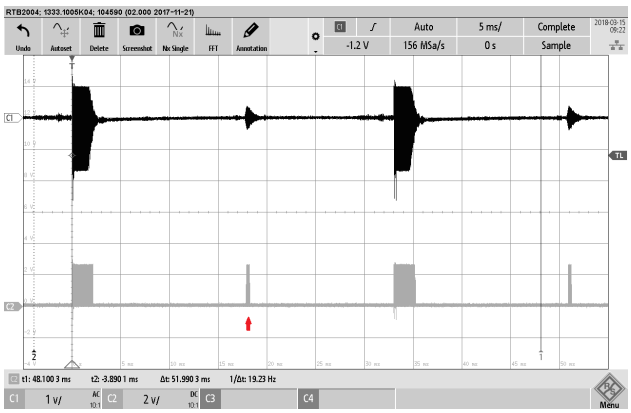

Abbildung 3.12: *Signalverlauf bei 5V auf 3m Abstand*

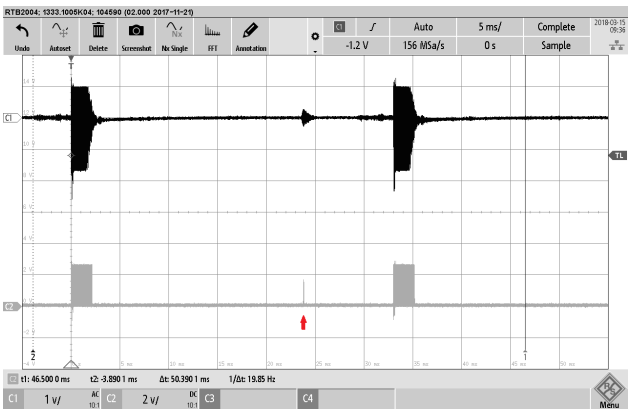

Abbildung 3.13: *Signalverlauf bei 5V auf 4m Abstand*

Bei den Abbildungen ist zu sehen, dass das Echo-Signal mit zunehmender Entfernung immer schwächer wird. Bei einer Entfernung von vier Metern (Abbildung 3.13) wird das Echo-Signal so schwach, dass die Signalstärke nach dem Komparator nicht mehr für eine eindeutige Auswertung über den Mikrocontroller ausreicht. Nachfolgend sind die Abbildungen einer Messreihe mit verschiedenen Spannungseinstellungen für den Sendebetrieb zu sehen. Anhand dieser Messreihe soll dargestellt werden, welchen Einfluss die eingestellte Spannung im Sendebetrieb auf die Reichweite des Ultraschallsignals hat. Für die Darstellung wurden die Messungen bei 5 Meter Abstand ausgewählt.

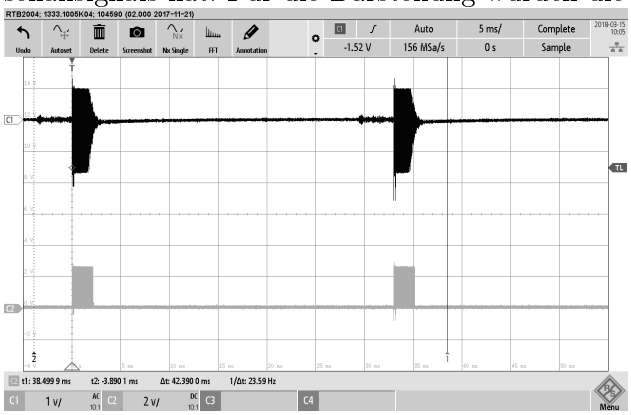

Abbildung 3.14: *Signalverlauf bei 5V auf 5m Abstand*

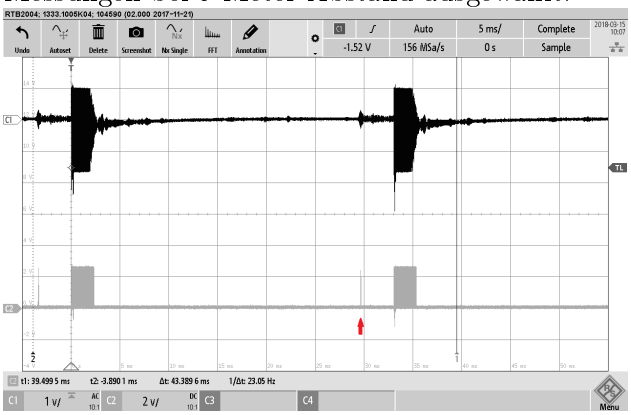

Abbildung 3.15: *Signalverlauf bei 10V auf 5m Abstand*

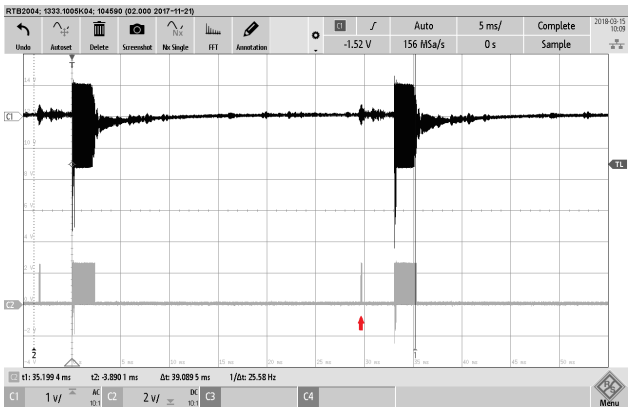

Abbildung 3.16: *Signalverlauf bei 15V auf 5m Abstand*

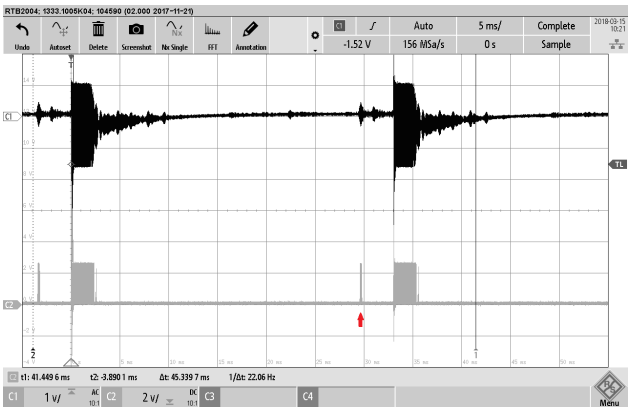

Abbildung 3.17: *Signalverlauf bei 20V auf 5m Abstand*

Bei Vergleich der Abbildungen 3.14 und 3.15 mit den Abbildungen 3.16 und 3.17 ist zu sehen, dass bei einem Abstand von 5 Metern erst bei einer Sendespannung von über 10V, auch am Komparator ein über den Mikrocontroller auswertbares Signal vorhanden ist.
In der nachfolgenden Tabelle 3.1 wurden die vom ersten Impuls des gesendeten Signals, bis zum ersten Impuls des Echo-Signals aufgetragen, und anhand der Schallgeschwindigkeit die Entfernung, die der Schall zurückgelegt hat berechnet. Dazu wurde noch die Abweichung der berechneten Entfernung von der eingestellten Entfernung angegeben.

Entfernung [m]	Zeit bis Anfang Echo [ms]	Errechnete Entfernung [m]	Abweichung [cm]
1	6,07	1,0416	4,16
1,5	8,97	1,5392	3,92
2	11,92	2,0454	4,54
2,5	14,8	2,5396	3,96
3	17,75	3,0459	4,59
3,5	20,65	3,5435	4,35
4	23,56	4,0428	4,28
4,5	26,49	4,5456	4,56
5	29,46	5,0553	5,53

Tabelle 3.1: *Entfernungsmessung mit Abweichung bei 20V Sendespannung*

Wie befürchtet sind bei der errechneten Entfernung Abweichungen im Bereich weniger Zentimeter aufgetreten. Bei Betrachtung der Abweichungen wird allerdings deutlich, dass die Werte bis auf einen, alle im Bereich von 4cm bis 4,5cm liegen. Ähnliche Abweichungen waren auch bei den anderen Messreihen zu beobachten. Somit ließe sich die Abweichung durch einen Korrekturwert auf ein Minimum reduzieren und würde einen Zentimeter nur noch selten überschreiten. Des weiteren sollte festgehalten werden, dass der erste Impuls des Echo-Signals als Referenz für die Entfernungsberechnung zu nutzen ist. Die anfänglichen Überlegungen, den letzten Impuls des Echo-Signals, oder einen Mittelwert aus allen empfangenen Impulsen zu verwenden beinhaltet ein höheres Fehlerpotential, da die Dauer des Echo-Signals durch niederfrequente Störgeräusche deutlich verlängert werden kann. Dies ließ sich bei einem der Tests beobachten, als im Hintergrund ein Lasercutter betrieben wurde. Dabei entstanden entgegen der Befürchtung keine Störsignale, stattdessen war die Signalintensität des Echo-Signals deutlich höher als bei Versuchen in einer stillen Umgebung.

3.3 Fazit aus den Ergebnissen für den Auftraggeber

Aus den Versuchen und Messungen lassen sich mehrere Aussagen treffen.
Als erstes, eine Ultraschall-Entfernungsmessung ist mit wenigen Bauteilen, sowohl als zwei Kapsel Variante, als auch als ein Kapsel Variante durchführbar. Bei der ein Kapsel Variante ist darauf

zu achten, dass der Verstärker eine ausreichende Spannungsfestigkeit besitzt , um nicht durch das Sendersignal zerstört zu werden. Auch ist wichtig, dass bei der Erzeugung des PW-Modulierten Ausgangssignals die MOSFETs so angesteuert werden, dass das MOSFET, welches die High-Impulse schaltet, genug Zeit zum abschalten hat , bevor das MOSFET , das das LOW-Signal schaltet, einschaltet, und umgekehrt um Kurzschlüsse an dieser Stelle zu vermeiden.
Eine Auswertung des Echo-Signals ist prinzipiell sowohl Digital, als auch Analog möglich. Die Digitale Auswertung läuft recht simpel ab, prinzipiell muss nur die Zeit erfasst werden, die zwischen dem Senden des PWM-Signals und dem Empfangen des Echo-Signals vergeht. Die errechnete Strecke ist zu halbieren, da die vergangene Zeit sowohl den Hin-, als auch den Rückweg beinhaltet. Bei der analogen Auswertung besteht zwar die Möglichkeit über einen Frequenzvergleich auch Signale mit noch kleinerer Echo-Amplitude zu erkennen und auszuwerten, allerdings beinhaltet dieses Vorgehen einen deutlich höheren Programmieraufwandt.
Bei der Berechnung der Zeiten muss berücksichtigt werden, dass das eingehende Echo-Signal die Ultraschallkapsel langsam in Schwingungen versetzt, die ersten eintreffenden Schwingungen eines PWM-Signals erzeugen also kleinere Spannungssignale als die darauf folgenden. Außerdem schwingt die Ultraschallkapsel auch nach Ende des eingehenden Signals noch etwas nach, was zur Folge hat, dass das analoge Abbild des gesendeten PWM-Signals leicht versetzt und etwas verlängert wirkt. All diese Faktoren müssen für eine genauere Berechnung der Strecke, die das Signal zurück gelegt hat, berücksichtigt werden. Allein durch die Tatsache, dass das empfangene Signal an der Ultraschallkapsel derart verändert wird, mach eine absolut exakte Messung nicht möglich. Eine Beschränkung des Fehlers auf einzelne Zentimeter ist aber realisierbar.

4 Reflektion über den Projektablauf

Nach Erhalt der Aufgabenbeschreibung war das Bild, das man sich von den bevorstehenden Aufgaben machte, doch sehr anders, als die Abeiten später aussahen. So war mit der Programmierung, auf Grund des doch deutlich komplexeren Mikrocontrollers, ein enormer Lernaufwandt verbunden. Die Recherche der Hardwarebauteile und das Erstellen der Schaltpläne und Platinen hingegen verlief ähnlich den Erwartungen.
Eddy: Die Einarbeitung in die Tinkerforge verwendeten Programme, KiCad und GitHub, erforderten genausoviel Aufmerksamkeit wie die Programmierung trotz der Einarbeitungsphasen war stest konstant ein Lernerfolg zu verbuchen und konnten bis zum ende des Projektes uns neues Wissen sowie Fähigkeiten erwerben.

5 Anhänge

Abbildungen der Messreihen mit verschiedenen Spannungen bei Abständen von ein bis fünf Metern.

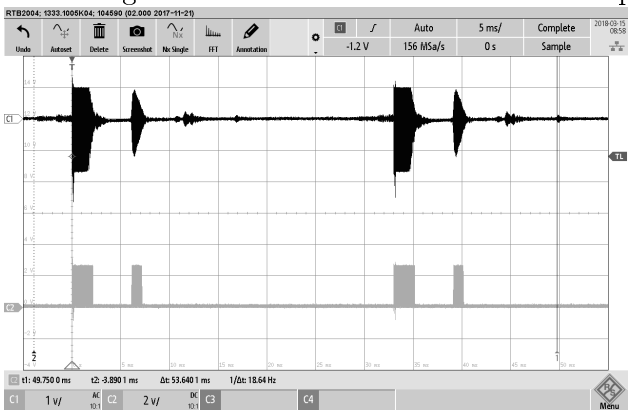

Abbildung 5.1: *Signalverlauf bei 5V auf 1m Abstand*

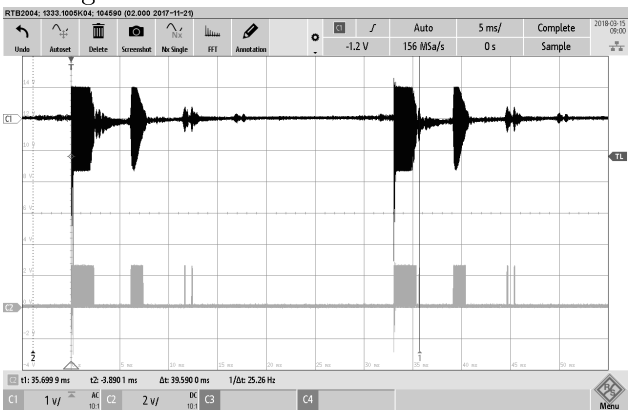

Abbildung 5.2: *Signalverlauf bei 10V auf 1m Abstand*

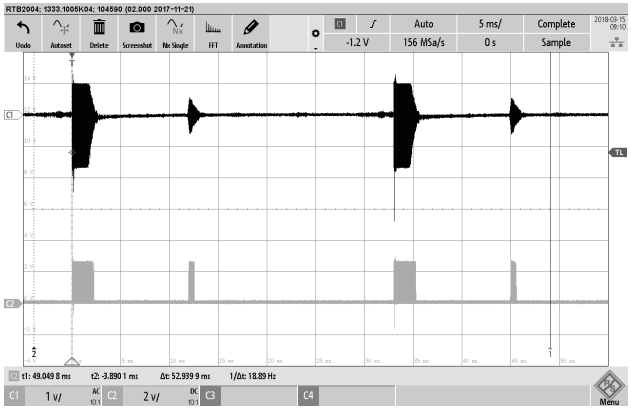

Abbildung 5.3: *Signalverlauf bei 5V auf 2m Abstand*

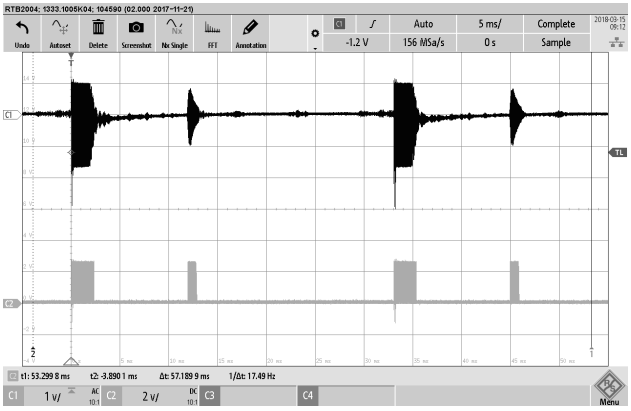

Abbildung 5.4: *Signalverlauf bei 10V auf 2m Abstand*

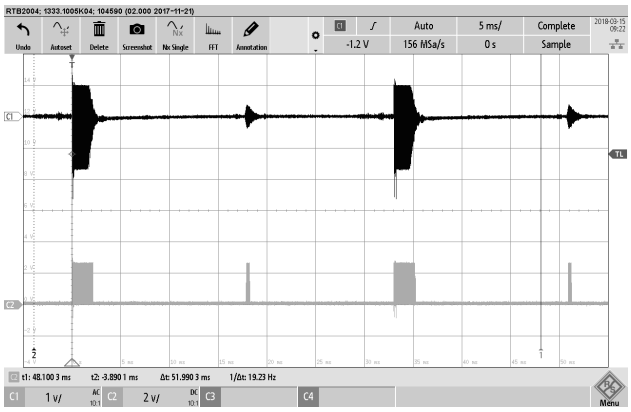

Abbildung 5.5: *Signalverlauf bei 5V auf 3m Abstand*

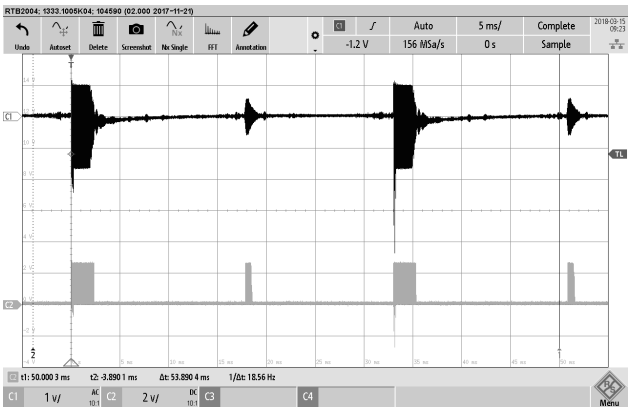

Abbildung 5.6: *Signalverlauf bei 10V auf 3m Abstand*

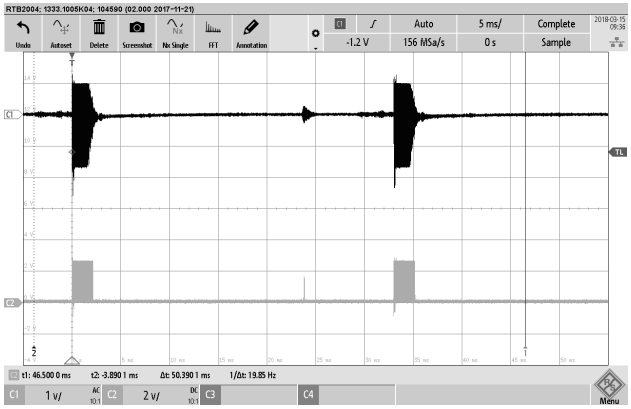

Abbildung 5.7: *Signalverlauf bei 5V auf 4m Abstand*

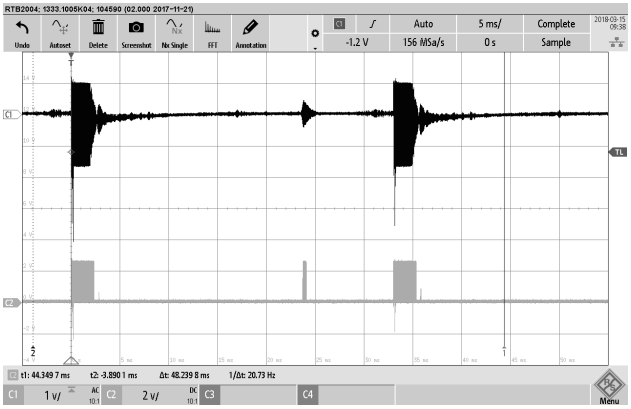

Abbildung 5.8: *Signalverlauf bei 10V auf 4m Abstand*

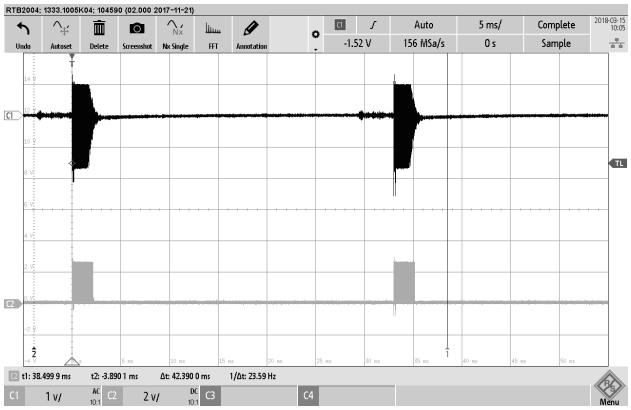

Abbildung 5.9: *Signalverlauf bei 5V auf 5m Abstand*

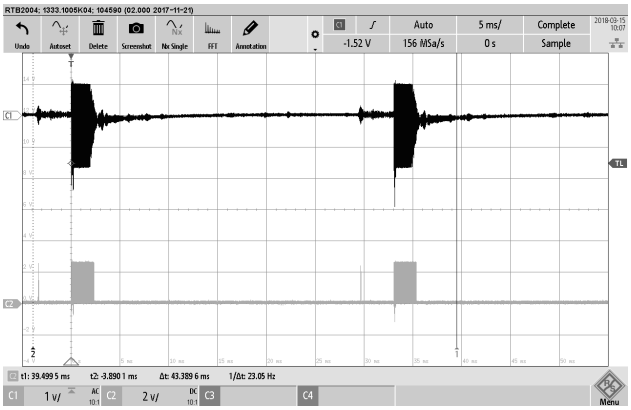

Abbildung 5.10: *Signalverlauf bei 10V auf 5m Abstand*

Abbildungsverzeichnis

Tabellenverzeichnis