Bachelorarbeit

Zugangsüberwachung des Sprachlabors der BTU Cottbus – Senftenberg

Access Control of the Speechlab of the BTU Cottbus – Senftenberg

Brandenburgische Technische Universität
Cottbus – Senftenberg

Lehrstuhl Kommunikationstechnik
Fakultät 3

Autor:	Friedrich Eckert
Matrikel-Nr.:	3248994
Studiengang:	Elektrotechnik
E-Mail:	friedrich.eckert@b-tu.de
Betreuer:	Prof. Dr.-Ing. habil. Matthias Wolff
	Dipl.-Ing. Christian Richter

Datum der Abgabe:	29.01.2018

Brandenburgische Technische Universität Cottbus - Senftenberg

Lehrstuhl Kommunikationstechnik

Aufgabenstellung

Der Lehrstuhl Kommunikationstechnik verfügt über das Forschungslabor Speechlab. Ziel einer dort befindlichen Zugangskontrolle ist es, jederzeit in der Lage zu sein, die Anzahl der im Labor befindlichen Personen zu kennen. Die rechnermäßige Erfassung der Personenzahl erfordert den Einsatz spezieller Hardware (Geber) und deren Auswertung. Des Weiteren sollen die gewonnenen Daten zur Steuerung und Überwachung der im Speechlab befindlichen Technik Verwendung finden.

Diese Arbeit umfasst:

- Konzeption der Zugangsüberwachung
- Entwicklung und Fertigung der Hardware
- Programmierung der Mikrocontroller in C
- Programmierung des Raspberry Pi in Python
- Auswertungsprogramm für PC und Ethernet-Schnittstelle in Java
- Einbindung in die Cognitive Systems Lab (CSL) Oberfläche LCARS

Selbstständigkeitserklärung

Der Verfasser erklärt, dass er die vorliegende Arbeit selbstständig, ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt hat. Die aus fremden Quellen (einschließlich elektronischer Quellen) direkt oder indirekt übernommenen Gedanken sind ausnahmslos als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form oder auszugsweise im Rahmen einer anderen Prüfung noch nicht vorgelegt worden.

Ort, Datum	Unterschrift

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Formelverzeichnis

Abkürzungsverzeichnis

ADU	Analog Digital Umsetzer
BOR	Brownout Reset
CRC	Cyclic Redundancy Check
DB	Datenblatt
DUT	Device under Test
EIRP	Equivalent Isotropically Radiated Power
FVR	Fixed Voltage Reference
GFSK	Gauss'sian Frequency Shift Keying
GIE	Global Interrupt Enable
ICSP	In Circuit Serial Programming
IDE	Integrated Development Environment
IFA	Inverted-F Antenna
IOC	Interrupt on Change
IRQ	Interrupt Request
LCARS	Library Computer Access/Retrieval System
LPBOR	Low Power Brownout Reset
MCLR	Master Clear Reset
MCU	Microcontrolling Unit
OPV	Operational Amplifier
PGC	Programming Clock
PGD	Programming Data
SFR	Special Function Register
TVS	Transient Voltage Suppressor
WDT	Watchdog Timer

1 Einleitung

Das Sprachlabor der BTU Cottbus – Senftenberg wird vom Lehrstuhl Kommunikationstechnik betrieben. Von dessen Seite besteht der Wunsch, jederzeit bestimmen zu können, wer sich im Labor aufhält. Die erfassten Daten können dann intern zur Information anderer Kolleginnen und Kollegen dienen, sowie für zukünftige Forschungsprojekte einen Ausgangspunkt geben. Auch die Automatisierung bestimmter Systeme, wie das selbstständige hochfahren von Computern, ist denkbar. Die bisher im Sprachlabor eingerichtete Sensorik kann zwar erkennen, dass sich Personen im Raum befinden, jedoch nicht wie viele und wer. Das im Zuge dieser Arbeit zu entwickelnde System soll diese Lücke schließen und in der Lage sein, die Anwesenheit und Identität der Personen im Raum festzustellen.

2 Grundlagen

2.1 Drahtlose Datenübertragung

Als drahtlose Datenübertragung wird die Funkübertragung von Signalen mittels elektromagnetischer Wellen bezeichnet. Ein Vorteil zur drahtgebundenen Übertragung ist die höhere Mobilität, da Sender und/oder Empfänger nicht durch eine Leitung ortsgebunden sind. Um das zu sendende Basisbandsignal in ein Frequenzband zu verschieben, das besser an den Übertragungskanal angepasst ist, wird es einem Trägersignal aufgeprägt. Dies wird Modulation genannt. Die Frequenzmodulation soll hier speziell erwähnt werden, da sie in diesem Projekt zur Anwendung kommt. Sie ist eine Sonderform der Phasenmodulation und zählt zu den nichtlinearen Modulationsarten. Die Amplitude des Basisbandsignals wird verwendet, um die Frequenz eines Trägersignals zu verändern. In der Praxis wird zur Modulation zum Beispiel ein spannungsgesteuerter Oszillator (VCO) verwendet [Klostermeyer, 2001, S.17-19]. Wird ein digitales, also Zeitdiskretes Signal moduliert, so wird dies Umtastung statt Modulation genannt [Rudolph, 2011, S.13]. Unstetige Signale würden jedoch auch im Trägersignal Sprünge verursachen, die eine unendliche Bandbreite benötigten. Eine Bandbegrenzung des Basisbandsignals sorgt für einen kontinuierlichen Übergang zwischen den Trägerfrequenzen. Findet dabei ein gauss‘sches Tiefpassfilter Anwendung, so wird von der Gauss‘schen Frequenzumtastung (GFSK) gesprochen. Genau betrachtet geht dadurch die Umtastung in eine Modulation über, da das Signal durch die Filterung wieder zeitkontinuierlich wird [Rudolph, 2011, S.14]. GFSK kommt auch im hier verwendeten Funkmodul RFM75 zum Einsatz. Die Demodulation geschieht häufig mit einer Phasenregelschleife (PLL), wobei das Steuersignal des VCO das demodulierte Basisbandsignal repräsentiert [Klostermeyer, 2001, S.30].

Welche Frequenzbänder eine Funkanwendung nutzen darf, ist im Frequenzplan der Bundesnetzagentur geregelt. Der vom RFM75 unterstützte Frequenzbereich von 2450 bis 2483,5 MHz fällt in das ISM-Band (Industrial, Scientific and Medical Band). Laut Frequenzplan dürfen *„Funkanwendungen geringer Reichweite“* zur *„Übertragung von Daten, Ton- und Bildsignalen über kurze Entfernung“* lizenzfrei genutzt werden, wenn ihre Sendeleistung 10mW (EIRP) nicht übersteigt [Frequenzplan, 2016, Eintrag 303004].

2.2 Datenverschlüsselung mit 3-Way

Die elektronische Datenverschlüsselung soll verhindern, dass der Inhalt einer Nachricht von unbefugten Dritten mitgelesen werden kann, auch wenn diese Zugriff auf den Übertragungskanal und Wissen über das Verschlüsselungsverfahren haben. In diesem Projekt soll verhindert werden, dass Dritte die Sender einer bestimmten Person zuordnen und so nachverfolgen können. Auch sollen sie nicht mit einem nachgebauten Sender im Labor erkannt werden, oder sich als jemand Anderes ausgeben können. Um dies zu realisieren, soll die Blockchiffre „3-Way“ zum Einsatz kommen. Dieses

Verfahren wurde 1994 von Joan Daemen vorgestellt, der später auch an der Entwicklung von Rijndael und Keccak beteiligt war, welche die Wettbewerbe für AES und SHA-3 gewannen. 3-Way arbeitet mit 96Bit Block- und Schlüssellänge [Daemen, 1994, S.1]. Dies ist ein guter Kompromiss zwischen den gängigen naheliegenden 64 und 128Bit, da die Blocklänge die Nachrichtenlänge festlegt und ein längerer Schlüssel höhere Sicherheit gewährleisten kann.

Im Detail arbeitet 3-Way mit den sechs, in Tabelle 1 gelisteten, Funktionen [Daemen, 1994, S.2-4]:

Funktion	Beschreibung
μ	Bitrichtungsumkehr des gesamten 96Bit Blockes, gilt für Schlüssel und Nachrichtenblöcke
γ	Nichtlineare Substitution, andere Darstellung als drei Bit Substitutionsboxen
ϑ	Lineare Substitution, jedes Ausgangsbit setzt sich aus sieben Eingangsbits zusammen
π1, π2	Bitpermutationen, Eingangsbitfolgen werden um feste Stellenanzahl rotiert
δ(Ar)	Exklusiv-Oder Verknüpfung von Argument Ar mit der r-ten Rundenkonstante

Tabelle 1: Funktionen der 3-Way Verschlüsselung

Werden einige dieser Funktionen zur Rundenfunktion ρ verkettet:

$$\rho = \pi_2 \circ \gamma \circ \pi_1 \circ \theta \tag{1}$$

So gilt für die Verschlüsselung mit elf Runden:

$$E_k = \theta \circ \delta(K_{r=11}) \bigcirc_{r=0}^{10} \rho(\delta(K_r)) \tag{2}$$

Hierbei ist zu beachten, dass den elf Runden eine zwölfte nachgestellt ist, in der eine andere Operation ausgeführt wird. Um ein Chiffrat zu entschlüsseln ist eine Bearbeitung des Schlüssels notwendig:

$$K' = \mu(\theta(K)) \tag{3}$$

Außer dieser Veränderung muss der Verschlüsselung lediglich eine μ-Funktion voran- und nachgestellt werden, um die Entschlüsselungsroutine zu erhalten:

$$D_k = \mu \circ \left(\theta \circ \delta(K'_{r=11}) \bigcirc_{r=0}^{10} \rho(\delta(K'_r)) \right) \circ \mu \tag{4}$$

Ein Verschlüsselungssystem kann als wahrscheinlich sicher anerkannt werden, wenn die Kosten eines Angriffs den zu erlangenden Gegenwert übersteigen und wenn nach der dazu benötigten Zeit der Nachrichteninhalt nicht mehr geheim gehalten werden muss. Weiterhin muss nach der Kerckhoff'schen Maxime die Sicherheit vom Schlüssel ausgehen, nicht von Intransparenz durch Geheimhaltung des Algorithmus [Schneier, 1996, S.8]. Die Schlüssel sind im Idealfall zufällige Bitfolgen, deren Auftretenswahrscheinlichkeiten über den Schlüsselraum gleichverteilt sind. Der Informationsgehalt jedes Schlüssels sollte also gleich dem Entscheidungsgehalt des Schlüsselraumes sein. Außerdem wird empfohlen, die Schlüssel regelmäßig zu ändern [Schneier, 1996, S.203].

Im Jahr 1997 gelang es, mittels differenzieller Kryptoanalyse 3-Way zu brechen. Der Angriff macht sich zunutze, dass die Rundenschlüssel durch XOR-Verknüpfung mit öffentlichen Rundenkonstanten erzeugt werden. Werden nun gewählte Klartexte mit dem System verschlüsselt, so ist die Differenz zwischen mehreren Texten nachverfolgbar. Auch die Differenz zwischen den Rundenschlüsseln kann abgeschätzt werden. Im Test gelang es so, mit 2^{22} gewählten Klartexten den Schlüssel zu finden [Kelsey, 1997, S.2]. Voraussetzung für die Attacke ist, dass der Angreifer Zugriff auf eine ausreichende Anzahl von Klartext-Chiffre-Paaren hat. Die Implementation von 3 Way in der Zugangsüberwachung erlaubt es jedoch nicht, frei gewählte Klartexte verschlüsseln zu lassen. Auch verlassen diese nach dem Entschlüsseln nie vollständig den Mikrocontroller, ein Teil muss immer erraten werden. Außerdem dauert es bei einer Senderate von 12 Nachrichten pro Minute fast 243 Tage um 2^{22} Nachrichten zu erhalten, die mit dem gleichen Schlüssel chiffriert wurden. Dieser Angriff ist sehr aufwändig und übersteigt wahrscheinlich den zu erreichenden Gegenwert. Vermutlich ist das System mit 3-Way ausreichend geschützt.

2.3 Umgang mit PIC Mikrocontrollern und MPLABX

2.3.1 Das PICkit3

PIC Mikrocontroller verwenden die Programmierschnittstelle ICSP (In Circuit Serial Programming), JTAG wird erst bei komplexeren 16 und 32Bit Controllern unterstützt. ICSP ist ein Dreidrahtbus, der den Reseteingang als Aktivierungssignal nutzt und auch „In-Circuit debugging“ unterstützt. Neben der Bereitstellung der genannten Funktionen kann das PICkit3 auch unabhängig von einem Rechner zum flashen von Programmen verwendet werden, wenn dieses zuvor in dessen Gerätespeicher übertragen wurde. Diese Funktion wird von Microchip als „Programmer to go“ bezeichnet [DS51795B, 2010, S.37]. Das PICkit3 wird über fünf Leitungen mit dem Mikrocontroller verbunden (Abb.1). Der sechste Anschluss wird für das „low voltage programming“ verwendet. In diesem Projekt wird dieser Modus nicht benutzt, zum debuggen soll er sogar deaktiviert sein [DS51795B, 2010, S.22].

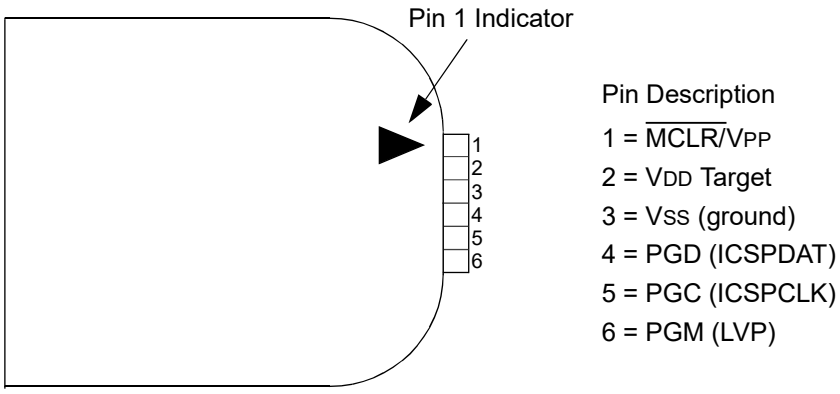


Abbildung 1: PICkit3 Anschlussbelegung (Quelle: DS51795B“, 2010, S.15)

2.3.2 Projekterstellung

MPLABX ist der Nachfolger der MPLAB IDE und läuft auf Grundlage der Netbeans IDE. Ein Projekt wird über „File → New Project... → Microchip Embedded → Standalone Project“ erstellt. In dem Dialog stehen sämtliche unterstützten Mikrocontroller zur Auswahl. Im nächsten Schritt wird das Programmiergerät „PICkit3“ und anschließend der passende „XC8/16/32“ Compiler ausgewählt. Nachdem abschließend der Projektname festgelegt ist, öffnet sich dieses in der Entwicklungsumgebung. Mit einem Rechtsklick auf den Ordner „Source Files“ im Projektexplorer wird über „New → main.c“ eine Programmdatei erstellt. Bestehende Programme und Bibliotheken können über Rechtsklick „Add existing Item“ hinzugefügt werden. Damit der Compiler diese Dateien findet, müssen die Ordner unter „File → Project Properties → XC8 Compiler“ eingetragen sein. Der ...-Button rechts im Feld „include directories“ öffnet ein Fenster, in dem via „Browse“ die Ordner hinzugefügt werden können. Die weitere Projektorganisation folgt den üblichen Konventionen: Programmdateien sollten nur über einen Programmheader eingebunden werden, der wiederum einen „include guard“ enthält, um die Mehrfacheinbindung zu verhindern.

2.3.3 Programmentwicklung

In jedem Projekt muss zuerst der Controller konfiguriert werden. In MPLABX gibt es dazu ein Werkzeug, das unter „Window → PIC Memory Views → Configuration Bits“ zu finden ist. Diese Konfigurationsbits legen die Quelle der Taktversorgung, Brownout (Reset bei zu geringer Betriebsspannung), spezielle Peripheriefunktionen, Speicherschutz (Auslesesperre) und mehr fest, die vom Start des Controllers an wirksam sind. Ein Klick auf „Generate Source Code to Output“ generiert den passenden C-Code in der Konsole, der dann zum Beispiel in den Header des Initialisierungsprogramms kopiert werden kann. Es folgen die projektspezifischen Initialisierungsroutinen für alle gewünschten Peripheriemodule und Funktionen, die mithilfe des Datenblattes entwickelt werden müssen. In jedem Modul muss der xc.h Header eingebunden

werden, um alle Registerbezeichner nutzen zu können [DS50002053G, 2016, S.18]. Alle wichtigen Vorschriften sind im MPLAB XC8 C Compiler User's Guide (DS50002053G, Stand 2016) beschrieben. Gleiches gilt für den XC16 und XC32 Compiler.

2.3.4 Programmübertragung und -debugging

Ist das Programm fehlerfrei und in einem Zustand, in dem es den Controller nicht durch falsche Pinkonfiguration physisch beschädigt, kann es mittels PICkit auf den Chip übertragen werden. Das PICkit kann die zu programmierende Schaltung bis zu einem Strom von 30mA selbst versorgen [DS51795B, 2010, S.20]. Hierzu kann über „File → Project Properties .→ PICkit3" in dem Aufklappmenü der Punkt „Power" angewählt und die gewünschte Ausgangsspannung eingestellt werden. Der Haken bei „Power Target from PICkit3" muss gesetzt sein. Benötigt das Projekt einen höheren Strom, so muss es extern versorgt werden. Ist das PICkit3 korrekt mit der Schaltung verbunden, kann mit einem Klick auf „Make and Programm Device Main Project"  in der Werkzeugleiste das Programm kompiliert und auf den Chip übertragen werden. Zum Debuggen wird „Debug → Debug Main Project" verwendet. Zu beachten ist, dass vor dem erneuten Debugstart, z.B. nach einer Programmänderung, ein eventuell noch laufender Debugprozess zuerst beendet werden muss, da es sonst zu Konflikten und Programmabstürzen kommen kann. Nützliche Werkzeuge zur Fehlersuche sind unter „Window → Debugging → Variables/ Disassembly" und unter „Window → PIC Memory Views → SFRs" zu finden. Die Ersten erlauben das Auslesen der Variableninhalte und die Gegenüberstellung des C Quelltextes mit dem daraus erzeugten Assemblercode. Letzteres bietet Einblicke in alle Register des Controllers.

3 Konzeption

3.1 Funktionsanforderungen

Das System soll in der Lage sein, zutrittsberechtigte Personen zu erkennen und zu entscheiden, ob sich diese im Labor aufhalten, oder nicht. Der Aktionsbedarf der Laborbesucher soll minimal sein, also keine bewusste Handlung zur Erkennung notwendig sein, die vergessen werden kann. Außerhalb der Laborräume darf das System die Identität der Personen nicht preisgeben, es ist vor unberechtigtem Auslesen zu schützen. Das System muss nach geltendem Recht betrieben werden dürfen, Funksysteme müssen den Bestimmungen der Bundesnetzagentur entsprechen. Die Herstellungs- und Betriebskosten sollen möglichst niedrig sein. Bestehende Labortechnik und das Erkennungssystem dürfen sich nicht gegenseitig in ihren Funktionen beeinträchtigen. Der Montageaufwand sollte so gering wie möglich sein und sich räumlich auf das Labor beschränken.

3.2 Technologieauswahl

3.2.1 Gesichtserkennnung

Gesichtserkennung basiert auf der Detektion von Gesichtern in einem Bild und dessen Charakterisierung, zum Beispiel mit einem künstlichen neuronalen Netzwerk. Eine konkrete Realisierung dieses Systems ist „DeepFace“. Es erreicht Erkennungsraten bis zu 97,35% [Kawulok, 2016, S.156]. Der betreffende Bildausschnitt, der als ein n-dimensionales Feld interpretiert werden kann (Farbwerte, Bildelemente, Positionsdaten, etc.), wird durch mehrfache Filterung in sogenannte „Feature Maps“ zerlegt. Ergebnis dieses Prozesses sind viele eindimensionale Felder, die dann mit einem Trainingsdatensatz verglichen werden können. So lassen sich Entscheidungen über den Bildinhalt treffen. Das Mitführen zusätzlicher Hardware ist zur Erkennung nicht nötig und auch ein direkter Blick in die Kamera ist nicht nötig, um erkannt zu werden, da das genannte Verfahren Gesichter aus unterschiedlichen Perspektiven erkennen kann. Allerdings muss das System vorher auf die zu erkennenden Personen trainiert werden, bei „DeepFace“ waren es etwa 1000 Bilder pro Person [Kawulok, 2016, S.156]. Das stellt einen unwirtschaftlichen Aufwand für die zu entwickelnde Raumüberwachung dar, sodass zumindest ein anderes Gesichtserkennungsverfahren verwendet werden muss. Probleme entstehen allerdings auch bei der tatsächlichen Implementierung. Um Personen beim Betreten und Verlassen des Labors zu erkennen, müssen Kameras im Raum und auf dem Flur montiert werden. Strom- und Datenleitung der Außenkamera müssten durch das Treppenhaus geführt werden. Aus den genannten Gründen eignet sich ein System auf Grundlage der Gesichtserkennung nicht für das Sprachlabor.

3.2.2 Passive Radiofrequenzidentifikation (passive RFID)

Bei RFID (Radio Frequency Identification) handelt es sich um ein automatisches Identifikationssystem [Tamm, 2010, S.9]. Passive RFID Tags sind sehr klein, preiswert und benötigen keine Wartung, da sie keine eigene Stromversorgung besitzen [RFID Insider]. Die Rücksendeenergie wird deshalb vom Lesegerät gestellt, also mitgesendet. Dies hat je nach System eine starke elektromagnetische Bestrahlung des Erfassungsbereiches zur Folge. Der kanadische Hersteller SkyRFID Incorporated gibt für sein passives 13,56MHz RFID System eine Sendeleistung von mindestens einem Watt (30dBm) an, wenn die Erfassungsdistanz einen Meter übersteigen soll. Das UHF System im Bereich von 860-960MHz erreicht sogar Reichweiten von 16m bei einer Sendeleistung von vier Watt EIRP (36dBm) [SkyRFID]. RFID-Tags, die in diesem Frequenzband arbeiten, wie zum Beispiel die Monza 4 Reihe des Herstellers Impinj oder die UCODE G2XM von NXP, erreichen laut Datenblatt noch höhere Leseentfernungen. Ersteres wird für hohe Reichweiten ohne Nennung exakter Werte, Letzteres für 7,1m bei zwei Watt EIRP (33dBm) angegeben. Während die Tags selbst preiswert sind, kosten die Lesegeräte sehr viel. Preise von 500 bis 2000 Euro sind laut AtlasRFIDstore einzuplanen [atlasRFIDstore]. Die genannten Baureihen Monza 4 und UCODE G2XM haben beide programmierbaren Speicher sowie die Möglichkeit, diesen vor unbefugtem Auslesen zu schützen. Eine Verschlüsselung der übertragenen Daten ist jedoch nicht implementiert. Allerdings wurde die praktische Realisierbarkeit eines Challenge-Response-Verfahrens in passiven RFID-Tags von Forschern der Tel Aviv University demonstriert [Arbit, 2014]. Der überwiegende Nachteil der passiven RFID sind die hohe benötigte Sendeleistung und der hohe Anschaffungspreis. Starke elektromagnetische Felder können störenden Einfluss auf die Laborelektronik haben, weshalb auf den Einsatz eines solchen Systems verzichtet wird.

3.2.3 Aktive Radiofrequenzidentifikation (active RFID)

Aktive RFID Tags besitzen eine eigene Stromversorgung, weshalb die Funkleistung zum Auslesen stark reduziert werden kann. Dies ist sogar notwendig, um die Batterielebensdauer zu maximieren. Aktive RFID-Systeme können in Transponder und Beacons unterteilt werden. Transponder antworten auf eine Anfrage des Lesegerätes, während Beacons, also „Leuchtfeuer“, regelmäßig Datenpakete senden, unabhängig davon, ob ein Empfänger in Reichweite ist, oder nicht. Aufgrund der verwendeten Bauteile sind aktive RFID Tags teurer als passive, haben aber bei geringerer Sendeleistung eine höhere Reichweite [RFID Insider]. Ein aktives RFID System eignet sich ideal für die Anwendung als Raumüberwachung, da die entstehenden Felder gering sind, Lesegeräte nur im Labor montiert werden müsssen und die mitzuführende Hardware kompakt ist. Sie erfordert auch keine spezielle Handlung beim betreten des Labors. Anstelle der Verwendung eines fertigen, teuren Systems (20-100 Euro pro Tag laut atlasRFIDstore), bietet sich hier eine Neuentwicklung an. Es lässt sich in der Funktionsweise genau an die Aufgabe anpassen und bietet die Möglichkeit der zukünftigen Erweiterung.

3.3 Hardwareauswahl

Unter Beachtung der in Kapiteln 3.1 und 3.2 genannten Aspekte, soll im Folgenden die zu verwendende Hardware bestimmt werden. Primäre Komponenten dabei sind die Mikrokontroller und das Funkmodul. Wegweisend ist der Sender, der einen Mikrocontroller und ein Funkmodul mit langer Batterielaufzeit und Verschlüsselungsfähigkeiten vereinen soll. Um eine Verschlüsselung auszuführen, seien an dieser Stelle 1KiB ROM und 128Byte RAM festgelegt. Diese Werte sind aufgerundet und entstammen einer Testimplementation mit Datenfeldern und Variablen folgender Art:

3 mal: uint8_t array[0xFF]; // Zufallszahlen und CRC8

6 mal: uint8_t variable; // Verschlüsselungsspeicher

10 mal:uint16_t variable; // andere Variablen, Zähler, etc.

Der Ruhestrom muss besonders gering sein, um die Batterie zu schonen, ein geringer Betriebsstrom ist wünschenswert, aber sekundär. Wenn das sogenannte Pin-remapping unterstützt wird, so ist es möglich, die Signale in der Software zu vertauschen, wenn sich die zugehörigen Leiterbahnen kreuzen würden. Diese Funktion ist nicht zwingend notwendig, beschleunigt jedoch die Platinenerstellung, macht die Leiterbahnführung übersichtlicher und ermöglich es, manche Layoutfehler auszugleichen. In Tabelle 2 sind zwei infrage kommende Mikrocontroller aufgeführt. Beide werden im Datenblatt als stromsparend bezeichnet. Funktionen die beiden Controllern gemein sind, wurden weggelassen.

Controller	I_{schlaf} ,WDT aktiv	$I_{Betrieb}$	RAM/ROM	Staffelpreis	Pin remapping
ATTiny84A	4µA[1)] DB S.189	2mA bei 8MHz[2)] DB S.184	512Byte/8KiB	0,601 EUR[3)]	nein
PIC16F1705	1,4µA[1)] DB S.377-378	1,2mA bei 16MHz[2)] DB S.375	1KiB/7KiB	0,782 EUR[3)]	ja

Tabelle 2: Vergleich der Mikrocontroller

1) bei 3V Betriebsspannung laut Datenblatt, WDT und ADU aktiviert

2) ATTiny hat 1MIPS/MHz (DB S.4), PIC hat 0,25MIPS/MHz (DB S.356)

3) ab 25Stk. ohne MwSt. bei Farnell, Stand 21.01.2018

Der Vergleich in Tabelle 2 zeigt, dass beide Controller imstande sind, nach oben genannten Kriterien eine Verschlüsselung durchzuführen. Der Stromverbrauch des ATTiny84A ist etwas höher, als der des PIC16F1705, sodass sich letzterer besser als Controller für den Sender eignet. Auch unterstützt der PIC das Pin-remapping, was insgesamt den etwas höheren Preis rechtfertigt und die Auswahl bekräftigt.

Für alle weiteren Module sollten Mikrocontroller des gleichen Herstellers verwendet werden, um die Entwicklungswerkzeuge und -umgebung nicht wechseln zu müssen. Die Auswahlkriterien für die restlichen Controller sind nicht durch die Stromaufnahme beschränkt, da sie nicht batteriegespeist sind. Die Wahl fiel auf den PIC16F1619 und den PIC18F25K40, da diese ebenfalls das Pin-remapping unterstützen und den Speicher- und Geschwindigkeitsanforderungen gerecht werden. Da hiervon nur geringe Stückzahlen nötig ist, sind die Preise von 1,10 Euro und 1,29 Euro akzeptabel. Eine Funktion die alle bisher genannten Controller gemein haben, ist der Ausleseschutz des Programmspeichers. Damit kann eine einfache Schlüsselextraktion verhindert werden. Vor Seitenkanalangriffen, wie der Stromattacke, oder dem freilegen des Siliziumchips schützt diese Funktion nicht. Angriffe wie die genannten übersteigen im Aufwand jedoch den zu erlangenden Gegenwert, weshalb darauf nicht weiter eingegangen werden soll.

Der zweite integrale Bestandteil des Senders ist das Funkmodul. In Tabelle 3 werden zwei passende Funkmodule verglichen.

Funkmodul	I_{Ruhe}	I_{Sende} (-18dBm)	Preis	Peripherie nötig	Formfaktor
RFM75	3µA	10,2mA	1,47 EUR[1)]	nein	Modul 1,27mm Pitch
NRF24L01	0,9µA	7mA	2,30 EUR[2)]	ja	QFN 0,5mm Pitch

Tabelle 3: Vergleich der Funkmodule

1) bei Pollin, 30 Stück, ohne MwSt. Stand 23.01.2018

2) bei Mouser, 30 Stück, ohne MwSt. Stand 23.01.2018

Beide in Tabelle 3 aufgeführten Funkmodule sind funktional geeignet. Sie unterscheiden sich vor Allem in der Stromaufnahme, die bei dem NRF24L01 deutlich niedriger ausfällt. Das RFM75 hingegen, kostet nur halb so viel, wenn bedacht wird, dass der NRF24L01 noch zusätzliche Bauteile benötigt. Genau das ist auch der Kritische Punkt in der praktischen Umsetzung. Der eigentlich besser geeignete NRF24L01 ist nur im QFN20 Gehäuse erhältlich und die Beschaltung, einschließlich impedanzangepassten Leiterbahnen und Anpassungsnetzwerk mit Antenne muss selbst umgesetzt werden. Es besteht das Risiko, dass mehrere Prototypen nötig sind, um eine einwandfreie Funktion sicherzustellen. Auch ist das QFN20 Gehäuse mit einem Kontaktabstand von 0,5mm sehr schwer aufzulöten. Um die genannten Risiken zu umgehen, wird der höhere Stromverbrauch in Kauf genommen. Sollte zu einem späteren Zeitpunkt der Bedarf nach noch längeren Batterielaufzeiten bestehen, so kann eine Weiterentwicklung unter Verwendung des NRF24L01 in Erwägung gezogen werden. Abschließend soll überprüft werden, ob das RFM75 Funkmodul unter Berücksichtigung der in Kapitel 2.1 genannten gesetzlichen Einschränkungen betrieben werden darf. Dafür wird zunächst der maximale zulässige Antennengewinn bei einer Sendeleistung von 4dBm bestimmt:

$$G_i = \frac{P_{EIRP}}{P_{ERP}} = \frac{10\,\cancel{mW}}{10^{\frac{4\,dBm}{10}} \cdot 1\,\cancel{mW}} = 3{,}981 \rightarrow 10 \cdot \log(3{,}981) = 6\,dBi \tag{5}$$

Formel 5 aus [Kark, 2017, S.224]

In der Applikationsbeschreibung AN043 von Texas Instruments wird eine ähnliche IFA auf ihre Charakteristika hin untersucht. Wird der gemessene Antennengewinn von 5,3dBi als Vergleichswert angesetzt, so liegt er noch unter den maximal zulässigen 6dBi [AN043, 2008, S.20]. Danach ist der Betrieb des Funkmoduls mit einer Sendeleistung von 4dBm erlaubt.

Das Raspberry Pi ist laut Aufgabenstellung vorgegeben und soll die Anbindung der Hardware an das lokale Netzwerk via Ethernet erleichtern. Dabei handelt es sich um einen Einplatinencomputer, auf dem standardmäßig ein Linux Betriebssystem läuft. Es ist möglich die simplen Schnittstellen wie SPI, I^2C, I^2S, UART und die GPIO Anschlüsse mit eigenen Programmen zu kontrollieren [raspberrypi.org]. Auf diese Weise kann eigene Hardware mit dem Betriebssystem interagieren.

4 Hardwareentwicklung

In den drei folgenden Unterkapiteln wird der Aufbau der in diesem Projekt entwickelten Gerätetypen erläutert: die tragbaren Sender, die stationär im Raum verteilten Empfängermodule und das Steuermodul.

4.1 Sender

Der Sender besteht im Wesentlichen aus dem Funkmodul RFM75 und dem Mikrocontroller PIC16F1705. Die Stromversorgung wird durch eine drei Volt Zelle des Typs CR2032 gewährleistet. Da die meisten Kupferfüllflächen Masseflächen sind und der Batteriehalter mit dem Pluspol verbunden ist, kann es zwischen diesen Punkten leicht zu einem Potenzialunterschied bei der Handhabung der Platine kommen. Um die Schaltung vor Spannungsspitzen beim Einlegen der Knopfzelle zu schützen, wird die Eingangsspannung von dem Transientensuppressor D1 begrenzt.

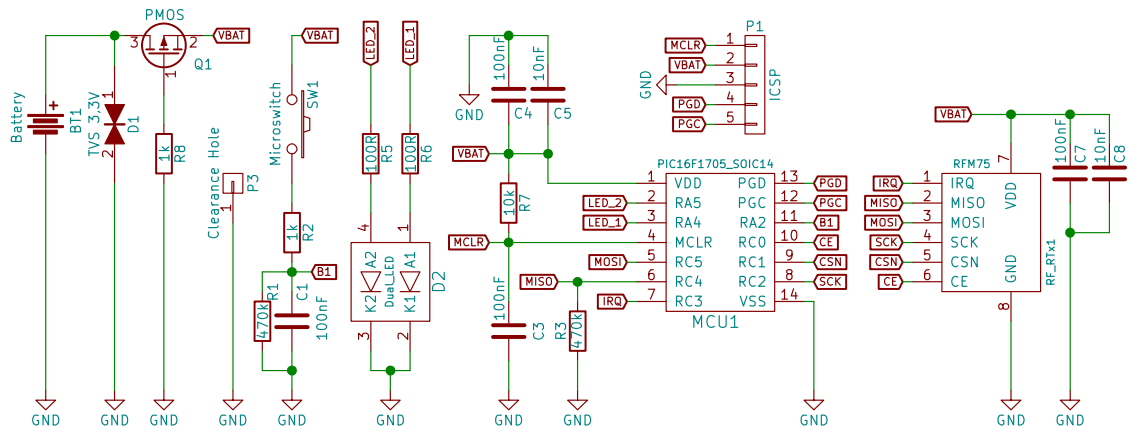


Abbildung 2: Stromlaufplan des Senders

Der P-Kanal Mosfet Q1 fungiert als Verpolschutz mit sehr niedriger Vorwärtsspannung, wie Gleichung 6 zeigt.

$$U_{DS} = R_{DSon} \cdot I = 200\,m\Omega \cdot 20\,mA = 4\,mV \tag{6}$$

Der Wert des Leitwiderstandes ist dem Datenblatt des PMV160UP entnommen, die Stromstärke ist abgeschätzt und übersteigt den erwarteten Spitzenwert. Der RC Tiefpass, bestehend aus R2 und C1, entprellt das Schaltsignal von Taster SW1. Widerstand R1 ist der Entladewiderstand (Pulldown). In dem Gehäuse von D2 befindet sich eine rote und eine grüne LED. Sie werden zur Anzeige der Zellenspannung und zum debuggen genutzt. Der Mikrocontroller fordert einen Pullup-Widerstand (R7) am Rücksetzeingang (MCLR), sowie mindestens einen Glättungskondensator (C4, C5) dicht am Spannungseingang. Die gleichen Funktionen erfüllen C7 und C8 beim Funkmodul

RFM75. Die Steckerleiste P1 führt die Programmierschnittstelle heraus. Sie wird auf der Senderplatine nicht bestückt, da das Modul dann zu hoch für das Gehäuse wäre (siehe Abb. 3). Stattdessen wird zum Programmieren eine Adapterleitung mit Steckerleiste durch die metallisierten Lötaugen in der Platine gesteckt und zur Seite gedrückt. Durch das verkanten der Anschlüsse ist ein guter elektrischer Kontakt gewährleistet.

Abbildung 3: Senderplatine im Gehäuse

Der Lötstopplack überzieht auf der Platinenrückseite den größten Teil der Kontaktflächen der Programmierschnittstelle, wodurch unerwünschte Kontaktierung beim Batteriewechsel verhindert wird. Die drei Bohrungen rechts unten (Abb.4) dienen der Positionierung des Funkmoduls beim Verkleben mit der Platine. Hierfür dient eine dreipolige Stiftleiste, die wieder entfernt wird, bevor der Klebstoff aushärtet. Um den Leiterbahnverlauf im Fehlerfall leichter nachverfolgen zu können, wurde die Platine mit gelbem Lötstopplack beschichtet, welcher einen höheren Kontrast im Vergleich zum grünen Lack bietet.

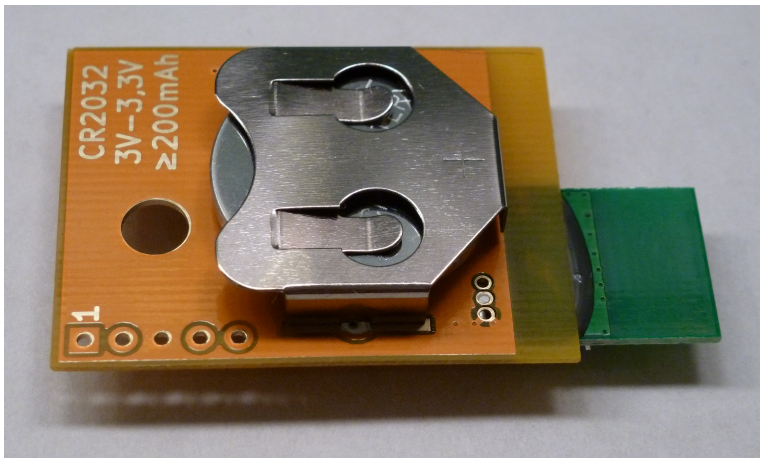


Abbildung 4: Rückseite der Senderplatine

4.2 Empfänger

Die prinzipielle Schaltungsstruktur (Abb.5) ähnelt der des Senders. Q2 dient als Verpolschutz und die Grundbeschaltung des Mikrocontrollers ist auch die gleiche. Die Eingangsspannung beträgt fünf Volt, weshalb Spannungsregler Vreg1 eine Spannung von 3,3V für die restliche Schaltung liefert. Zur Anzeige nach außen werden zwei antiparallele LEDs im gemeinsamen Gehäuse (D2)verwendet. Um diese anzusteuern, werden zwei Mikrocontrolleranschlüsse als H-Brücke genutzt. Durch die Verwendung dieses LED Typs müssen nur ein Loch in das Gehäuse und zwei Löcher in die Platine gebohrt werden.

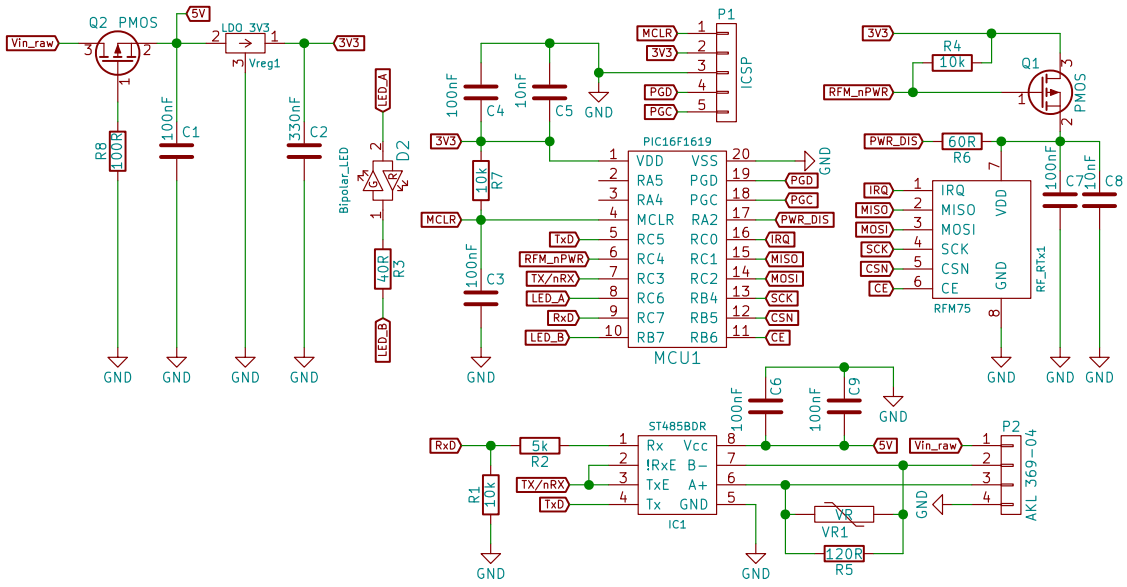


Abbildung 5: Stromlaufplan des Empfängers

Eine wichtige Anforderung an das gesamte System ist die eigenständige Erholung aus Fehlerzuständen. Das Funkmodul RFM75 bleibt bei mehrfachem Neustart in kurzen Abständen, etwa Schwankungen in der Versorgungsspannung, oft in Zuständen stecken, die den normalen Betrieb verhindern. Eine vollständige Rücksetzung des Moduls ist nur durch das Abschalten der Betriebsspannung für mindestens 100ms möglich, der Wert wurde erprobt. Deshalb ist diese über Q1 schaltbar und der Anschluss PWR_DIS erlaubt das gezielte Entladen der Stützkondensatoren. Der Empfänger besitzt eine drahtgebundene Datenschnittstelle nach dem RS485 Standard. Die Übertragung findet im Halbduplexbetrieb auf einem differenziellen Bus statt. IC1 wandelt die Signale zwischen Bus und Mikrocontroller. Da die Betriebsspannung des Transceivers 5V beträgt, senkt ein Spannungsteiler aus R1 und R2 das Empfangssignal auf etwa 3V ab. Der Bus benötigt einen Abschlusswiderstand von 120Ohm an beiden Enden. Bei einem Empfänger muss daher R5 bestückt werden. Varistor VR1 begrenzt die Spannungsdifferenz der Datenleitungen. Das Funkmodul RFM75 ist nahezu rechtwinklig zur Trägerplatine montiert (Abb.6 und 7). So ragt die

Antenne in den Raum und kann ungehindert zu allen Seiten abstrahlen. Die Ecken der Platine sind dem Gehäuse angepasst, wie in Abbildung 6 und 7 gezeigt. Zum Schutz vor Oxidation wurde die Leiterplatte mit Isolierlack beschichtet.

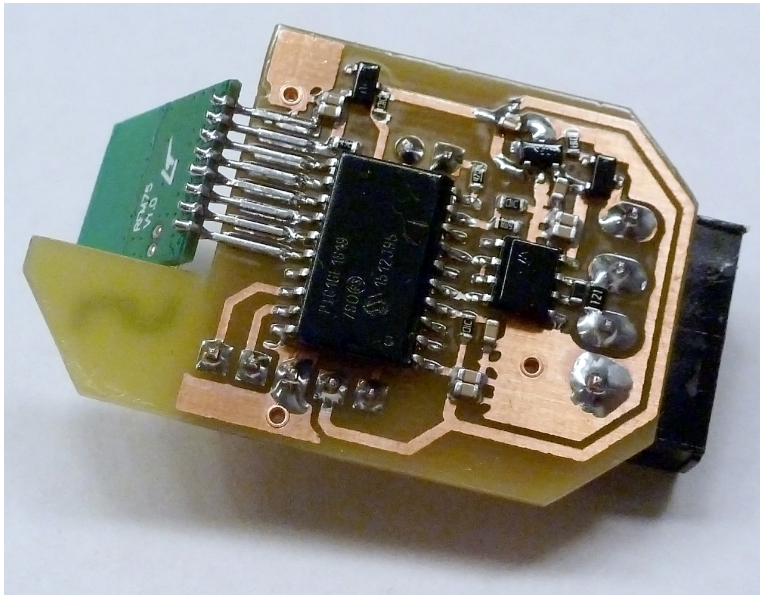


Abbildung 6: Rückseite der Empfängerplatine

Abbildung 7: Empfängermodul im Gehäuse

4.3 Steuermodul

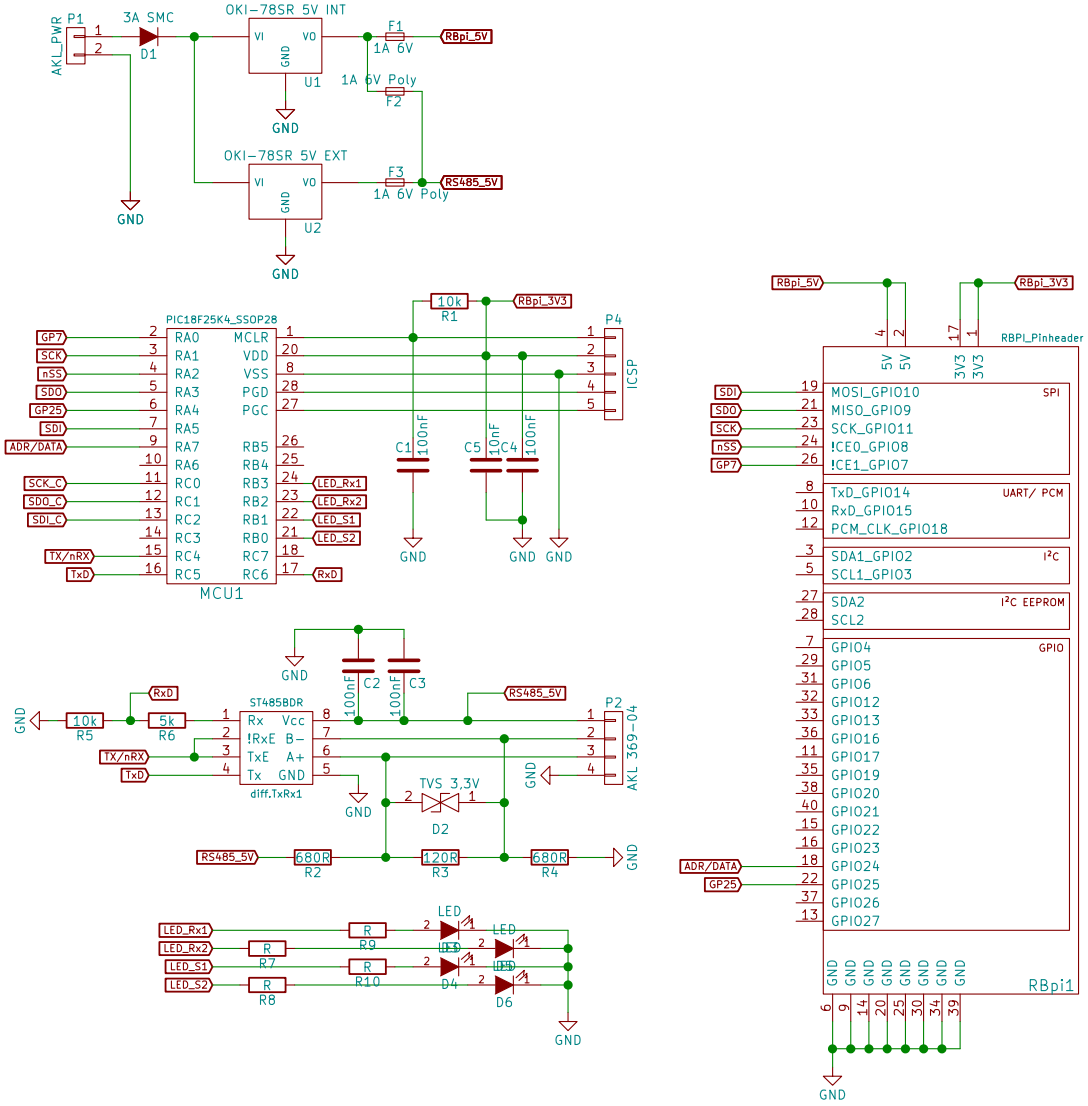


Abbildung 8: Stromlaufplan des Steuermoduls

Die Schaltung des Steuermoduls, dargestellt in Abbildung 8, ist in der Busanbindung identisch zum Empfängermodul aufgebaut. Einziger Unterschied ist das Vorspannungsnetzwerk, bestehend aus R2 bis R4, um die Spannungspegel stets auf definiertem Niveau zu halten. Das Steuermodul besitzt keinen eigenen 3,3V Spannungsregler. Stattdessen versorgt ein 5V Schaltwandler das Raspberry Pi, welches über einen 3,3V Regler verfügt und damit den Mikrocontroller speist. Das Modul ist mit einer Diode gegen Falschpolung geschützt. Die Vorwärtsspannung ist hier unkritisch, da das System mit 12V gespeist wird, jedoch maximal fünf Volt benötigt. Der Schaltwandler wird vor Überlast durch zwei Selbstrückstellende ein Ampere Sicherungen (PTC Sicherungen) geschützt. Zur Kommunikation des Mikrocontrollers PIC18F25K40 mit dem Raspberry Pi, sind beide Systeme über die SPI Schnittstelle verbunden. Vier LEDs dienen der Statusanzeige.

5 Softwareentwicklung

5.1 Firmware des Senders

Die Beacon Software ist auf kurze Programmlaufzeiten und geringen Stromverbrauch optimiert. Sie sorgt dafür, dass alle fünf Sekunden ein verschlüsseltes Datenpaket gesendet wird, welches die Sender-ID den Nachrichtenzählerstand, die Batteriespannung, die gemessene Temperatur und zwei Prüfwerte zur Validierung nach der Entschlüsselung enthält.

Die Inhalte einiger noch nicht initialisierter Register werden als Startwerte zur Erzeugung der Pseudozufallszahlen gespeichert. Anschließend wird die Initialisierung aller verwendeter Module durchgeführt. Der Watchdog Timer weckt die CPU regelmäßig aus dem Schlafmodus auf. Dann wird eine Nachricht zusammengestellt, verschlüsselt und in den Sendespeicher des Funkmoduls übertragen. Kurz nach Sendebeginn wird die CPU wieder in den Schlafmodus versetzt. So muss der Sender sich die Energie nicht mit ihr teilen. Ist die Nachricht gesendet, löst das Funkmodul im Mikrocontroller einen Interrupt aus. Die CPU wacht auf, deaktiviert den Sender und schläft dann die restliche Zeit, bis die nächste Nachricht gesendet werden soll. Die Schlafzeit ist in Segmente unterteilt, da bei einer Tastenbetätigung sonst die bisherige Schlafzeit nicht abgeschätzt werden kann (Siehe Abb.18, S.28, jede Stromspitze zeigt eine kurze Wachphase). Einmal pro Schlafphase wird beim Tastendruck via LED der Batteriestand angezeigt. Tastendrücke selbst werden immer registriert, gezählt und als Zufallselement verwendet. Bei jeder zwölften Nachricht, also einmal in der Minute, wird während des Sendevorganges zum Zeitpunkt des maximalen Spannungseinbruches die Betriebsspannung gemessen. Der Analog Digital Umsetzer (ADU) misst hierbei die Spannung einer 1,024V Referenzzelle mit der Batteriespannung als Bezug. Die Batteriespannung berechnet sich dann folgendermaßen:

$$U_{Bit} = \frac{U_B}{2^{Bittiefe}} \quad ; \quad U_{Bit} = \frac{U_{Ref}}{ADU_{Count}} \quad \rightarrow \quad U_B = \frac{2^{Bittiefe} \cdot U_{Ref}}{ADU_{Count}} \quad \rightarrow \quad U_B = \frac{2^{10} \cdot 1024\,mV}{ADU_{Count}} \tag{7}$$

Diese Berechnung wird nicht vom Sender ausgeführt, da der skalierte Wert in Volt erst von Interesse ist, wenn er tatsächlich empfangen wurde. Die Wachzeit wird auf diese Weise geringer. Weiterhin ist der Änderungsumfang des Messwertes geringer als die ADU Auflösung von zehn Bit. Die untere Spannungsgrenze liegt bei 1,9V, vorgegeben vom Funkmodul. Sowohl der Mikrocontroller, als auch das Funkmodul können ab 3,6V schaden nehmen (aus den Datenblättern), als Maximalspannung soll daher 3,5V angenommen werden. Nach Gleichung 7 folgen daraus aufgerundete ADU Messwerte von 300 bis 552. Das entspricht einer Differenz von 252, die mit acht Bit abgebildet werden kann. Die Spannungsauflösung liegt bei:

$$U_{res} = \frac{\Delta U_B}{\Delta ADU_{count}} = \frac{3{,}5V - 1{,}9V}{552\,Bit - 300\,Bit} = 7{,}3\,mV/Bit \tag{8}$$

Real haben die Werte einen noch geringeren Änderungsumfang, da die Batteriespannung unter Last einbricht und die Unterspannungserkennung (Brownout detector) des Mikrocontrollers toleranzbedingt bereits bei 2,1V auslösen kann. Aufgrund dessen kann ohne Genauigkeitsverlust von jedem ADU Messwert 300 subtrahiert werden. So werden bloß acht statt zehn Bit in der Nachricht belegt. Dagegen benötigt der skalierte Wert in mV, für eine Maximalspannung von 3500mV, zwölf Bit:

$$\left\lceil ld\left(\frac{3500\,\cancel{mV}}{1\,\cancel{mV}}\right)\right\rceil = 12\,Bit \tag{9}$$

5.2 Firmware des Empfängers

Die Satelliten (Empfangsmodule) empfangen eintreffende Nachrichten, speichern sie und leiten diese auf Befehl des Steuermoduls weiter. Sie können zur Laufzeit als Sender konfiguriert werden und Testnachrichten an die anderen Empfänger verschicken. Jedes Satellitenmodul vergleicht die von anderen Empfängern an das Steuermodul gesendeten Nachrichten mit denen im eigenen Speicher und löscht Duplikate. Bei Ausbleiben valider Instruktionen des Steuermoduls für mindestens acht Sekunden, startet das Empfangsmodul neu.

Standardmäßig werden die Funkmodule als Empfänger konfiguriert. Bei erfolgreichem Programmstart leuchtet die LED kurz grün auf, das Programm wartet nun auf Befehle vom Steuermodul, oder auf empfangene Funknachrichten. Das Funkmodul RFM75 signalisiert den Empfang eines Datensatzes mit einer fallenden Flanke am IRQ Anschluss. Im Mikrocontroller löst diese Transition einen Interrupt aus. Die Nachricht wird über SPI ausgelesen und das Interruptflag im Funkmodul zurückgesetzt. Dem Hauptprogramm wird der Empfang der Nachricht signalisiert, wo diese mit einem CRC Prüfwert versehen und der zuständige Speicherslot als sendebereit markiert wird. Solange weniger als 30 Nachrichten nach der letzten Speicherleerung empfangen wurden, wird für jede neue Nachricht ein leerer Slot gesucht. Dies geschieht mit aufsteigendem Index und kann in den meisten Fällen als linearer Speicher betrachtet werden. Darüber wird er als Ringspeicher interpretiert und die Slots werden, ebenfalls mit aufsteigendem Index, überschrieben. Der Empfang von Testnachrichten läuft fast identisch ab, mit dem Unterschied, dass im Hauptprogramm die Korrektheit der Nachricht geprüft wird und immer nur die letzte korrekte Testnachricht gespeichert wird. Um Befehle vom Steuermodul entgegenzunehmen, wartet die UART Schnittstelle auf ein neun Bit langes Datenwort, bei dem das Adressbit gesetzt ist. Die verwendeten Mikrocontroller können in dieser Situation einen Interrupt auslösen. Damit ist das Finden des Nachrichtenanfanges kein Problem. Wurde ein Adressbyte empfangen, wird entschieden, welcher Puffer zu verwenden ist (14Byte für Nachrichten, 4Byte für Befehle) und der Empfang fortgesetzt. Eine fehlerfreie Nachricht wird mit allen Nachrichten im Speicher abgeglichen und doppelt auftretende gelöscht. So bekommt das Steuermodul jede Nachricht nur einmal zugeschickt. Fehlerfrei empfangene Kommandos vom Steuermodul werden behandelt, wenn sie an den

Empfänger adressiert und im aktuellen Betriebszustand erlaubt sind oder andernfalls ignoriert. Mit jedem Empfang eines validen Befehls wird der Watchdog Timer zurückgesetzt. Bleiben diese, ungeachtet der Ursache, für mindestens acht Sekunden aus, setzt der Watchdog den Mikrocontroller zurück und das Programm startet neu. So kann ein funktionsfähiger Zustand garantiert werden. Treten während der Konfiguration des Funkmoduls Fehler auf, so geben diese Funktionen zunächst einen Fehlercode an die Ausnahmebehandlung weiter. Diese versucht das Problem zu lösen. Bleibt der Fehler weiterhin bestehen, wird der Watchdog verwendet, um den Controller neu zu starten.

5.3 Firmware des Steuermoduls

Das Steuermodul koordiniert den Betriebsablauf. Es liest die Nachrichten aus und initiiert in bestimmten Situationen Selbsttests. Es entschlüsselt die Nachrichten und stellt die dekodierten Datensätze dem Raspberry Pi zur Weiterverarbeitung zur Verfügung.

Nach Abschluss der Peripherieinitialisierung bereitet das Programm die Schlüssel zur schnelleren Entschlüsselung vor. Zur Ablaufsteuerung werden in festgelegten Abständen in der Interruptroutine eines Timers bestimmte Zustandsflags gesetzt, die im Hauptprogramm verschiedene Aktionen auslösen. Im Wesentlichen sind dies das Auslesen der empfangenen Nachrichten aus den Satellitenmodulen und der Selbsttest. Beide Prozesse sind ebenfalls als Zustandsautomaten implementiert. Empfangene Nachrichten werden im Intervall von fünf Sekunden von den Empfängern abgefragt. Selbsttests finden alle 15min sowie nach Ausbleiben von Funknachrichten für zwei Auslesezyklen statt. Außerdem wird der Zähler für die regelmäßigen Tests so initialisiert, dass einige Sekunden nachdem der Raspberry Pi hochgefahren ist, der erste Selbsttest stattfindet. Die UART Funktionen sind in Satelliten und Steuermodul identisch implementiert, einziger Unterschied ist, dass Letzteres als Busmaster auf Reaktionen warten muss. Dazu wurde eine Timeoutfunktion ergänzt, die den Bus nach gewisser Zeit wieder freigibt. So bleibt das Programm nicht hängen, wenn ein Teilnehmer nicht antwortet. Alle empfangenen Nachrichten werden auf Fehlerfreiheit geprüft und anschließend entschlüsselt. Da jeder Sender einen eigenen Schlüssel besitzt, müssen hierbei alle durchprobiert werden, bis die enthaltenen CRCs verifiziert werden können. Passt die in der Nachricht enthaltene Sender-ID zum verwendeten Schlüssel, wird sie als korrekt dechiffriert markiert und bei nächster Gelegenheit an das Raspberry Pi weitergeleitet. Das Steuermodul ist zwar als SPI Slave konfiguriert, signalisiert aber dem Raspberry Pi, wann und welche Daten ausgelesen werden können. Außerhalb dieser Zeit werden Leseversuche ignoriert. Eine spezielle Fehlerbehandlung ist nicht implementiert, da kein Funkmodul angeschlossen ist, welches in den anderen Geräten die Hauptfehlerquelle ist. Während des Langzeittests (Kapitel 6.1) konnten keine Probleme oder Unzuverlässigkeiten festgestellt werden, die eine Fehlerbehandlung nötig machen.

5.4 Raspberry Pi

Die Auswertung der Empfängerdaten ist in Python realisiert. Das Programm startet automatisch mit dem Betriebssystem und stellt anschließend die aufbereiteten Daten via UDP Verbindung zur Verfügung. Das Raspberry Pi reagiert nur auf Anfragen und sendet von sich aus keine Datenpakete.

Damit sich die Funktionen des Programms nicht überschneiden, werden diese in separaten Threads ausgeführt. Namentlich umfasst dies einen Hardwarethread, einen Netzwerkthread, einen Systemthread und das Hauptprogramm. Alle Threads können über eine Klasse auf gemeinsamen Speicher zugreifen. Die Methoden dieser Klasse verhindern Speicherkollisionen. Der Hardwarethread wartet auf einen Interruptbefehl vom Steuermodul und liest dann via SPI Schnittstelle den zur Verfügung stehenden Datensatz aus. Ist dieser laut CRC fehlerfrei empfangen worden, wird er an eine Klasse übergeben, die das Abspeichern übernimmt. Die Aufgabe des Systemthreads ist es, in regelmäßigen Abständen die Systemtemperatur auszulesen. Der Netzwerkthread wartet auf Anfragen via UDP Schnittstelle, prüft bei Empfang den Befehl und reagiert mit der jeweiligen Antwort. Implementiert sind bisher das Auslesen der Systemtemperatur, der ausgewerteten Datensätze und das Ändern des Schwellwerts zur Anwesenheitserfassung. Im Hauptprogramm werden alle Threads überwacht und bei Absturz neu gestartet. Tritt im Hauptprogramm ein Fehler auf, wird der Raspberry Pi neu gestartet.

5.5 CSL Hardwaretreiber und LCARS Panel

Der Treiber fragt regelmäßig das Raspberry Pi nach neuen Daten ab und zeigt diese aufbereitet in einem LCARS Panel an.

Der Konstruktor der Klasse „AccessController“ startet einen separaten Thread, in dem die Initialisierung aller Datenfelder und Variablen, sowie die Konfiguration des Netzwerksockels erledigt wird. Anschließend sendet der Thread sekündlich Anfragen zur Datenübergabe an das Raspberry Pi. Ein Timeout von einer Sekunde sorgt dafür, dass das Programm nicht blockiert wird. Die Daten werden als Zeichenfolge (String-Array) übertragen, haben aber eine feste Struktur. So wird zur Übertragung der Werte in ein Datenfeld kein Parser benötigt, sondern nur deren Positionen in der Zeichenkette. Um Kollisionen beim Lesen und Schreiben zu verhindern, sind alle Felder doppelt vorhanden und eine weitere Variable zeigt den aktuellen Puffer an. Neue Daten werden stets in den inaktiven Puffer geschrieben, dann gewartet, bis andere Prozesse die Zugriffssperre aufheben. Erst dann wird der aktualisierte Puffer aktiviert. Auch dieser Vorgang ist durch ein Timeout in der Laufzeit begrenzt. Abschließend wird die Updatemethode des LCARS Panels benachrichtigt und eine Sekunde bis zur nächsten Datenabfrage gewartet.

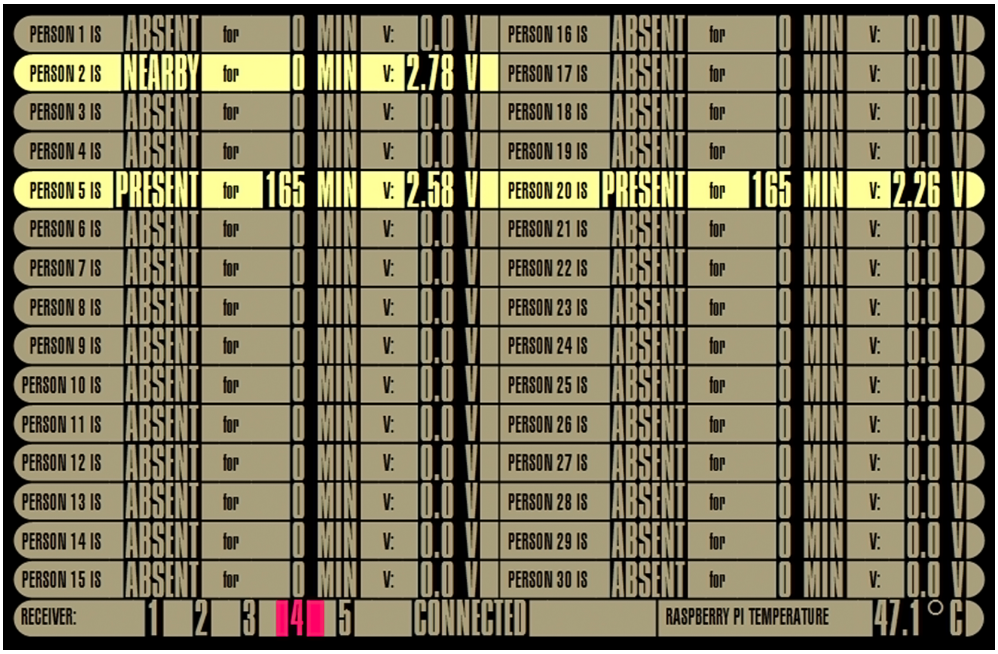


Abbildung 9: LCARS Panel, Empfänger 4 testweise nicht angeschlossen

Wie in Abbildung 9 zu sehen, zeigt das LCARS Panel für jeden vorhandenen Sender der Anwesenheitsstatus, die Zeitdauer seit dem letzten Statuswechsel, sowie die Batteriespannung an. Ist ein Sender nicht in Reichweite, so bleiben die letzten bekannten Werte erhalten. Sender in Reichweite werden hell dargestellt. Die Spannung wird in Volt angezeigt und auf zwei Nachkommastellen begrenzt, um keine höhere Genauigkeit vorzutäuschen. Laufzeiten werden wie folgt angezeigt:

$$
\begin{aligned}
0 \leq t_{run} < 3\,h &: \text{in min} \\
3\,h \leq t_{run} < 48\,h &: \text{in h} \\
t_{run} \geq 48\,h &: \text{in d}
\end{aligned}
$$

Die Fußleiste listet alle Empfänger auf und hinterlegt nicht angeschlossene in rot. Ist bei der letzten Abfrage ein Timeout eingetreten, so zeigt ein weiteres Feld „DISCONNECTED“ an, wie Abbildung 10 zeigt.

Abbildung 10: LCARS Fußleiste bei Verbindungsverlust zum Raspberry Pi

Der gesamte verbindungsbezogene Teil der Fußleiste wird dabei rot hinterlegt. Sobald bei erneuten Versuchen ein Datensatz empfangen wurde, wird „CONNECTED“ angezeigt und wieder auf das „Secondary“-Farbschema gewechselt. Zusätzlich kann in der Fußleiste die Systemtemperatur des Raspberry Pi abgelesen werden.

6 Praxistests und Optimierungen

6.1 Untersuchung der Empfangszuverlässigkeit

Wenn sich mehrere Sender in Empfangsreichweite befinden, besteht das Risiko einer Kollision, indem sich mindestens zwei Nachrichten zeitlich überlappen. Ebenfalls zu Empfangsverlust führt ein zu geringer Abstand zwischen den Nachrichten, da der Empfänger eine gewisse Zeit benötigt, um die 12 Bytes aus dem Speicher des Funkmoduls auszulesen. Zur Bestimmung der Verlustrate werden alle 30 Sender in Empfangsreichweite platziert und die entschlüsselten Datensätze über einen längeren Zeitraum protokolliert. Zur Auswertung wird der Verlauf des Nachrichtenzählers genutzt, der in der Logdatei bei Empfang einer neuen Nachricht aktualisiert wird und ansonsten konstant bleibt. Sei Z_n der Zählerstand zum Abtastpunkt n, so berechnen sich die unterschiedlichen Empfangszustände folgendermaßen:

$$Z_n - Z_{n-1} = \Delta Z \begin{cases} =0\,; \text{keine Nachricht gesendet} \\ =1\,; \text{eine Nachricht gesendet} \\ >1\,; \Delta Z - 1 \text{ Nachrichten verpasst} \end{cases}$$

Neben der grundsätzlichen Verlustrate, soll der Einfluss verschiedener Betriebszustände betrachtet werden, die in Tabelle 4 aufgelistet sind. Variiert werden die Sende und Ausleseintervalle, sowie die Empfängeranzahl:

Zustand	T_{TX}/s	T_{RX}/s	$N_{Empfänger}$	$R_{Verlust,max}$/%	$\overline{R}_{Verlust}$/%	T_{mess}/h
A	1,02	0,92	5	4,18	3,63	18,7
B	5,12	0,92	5	1,14	0,90	26,78
C	5,12	4,15	5	0,94	0,84	27,23
D	5,12	4,15	1	1,62	1,03	20,20

Tabelle 4: Verlustraten der Betriebszustände

In den Abbildungen 11 bis 14 ist das Verhältnis zwischen verpassten und gesendeten Nachrichten dargestellt.

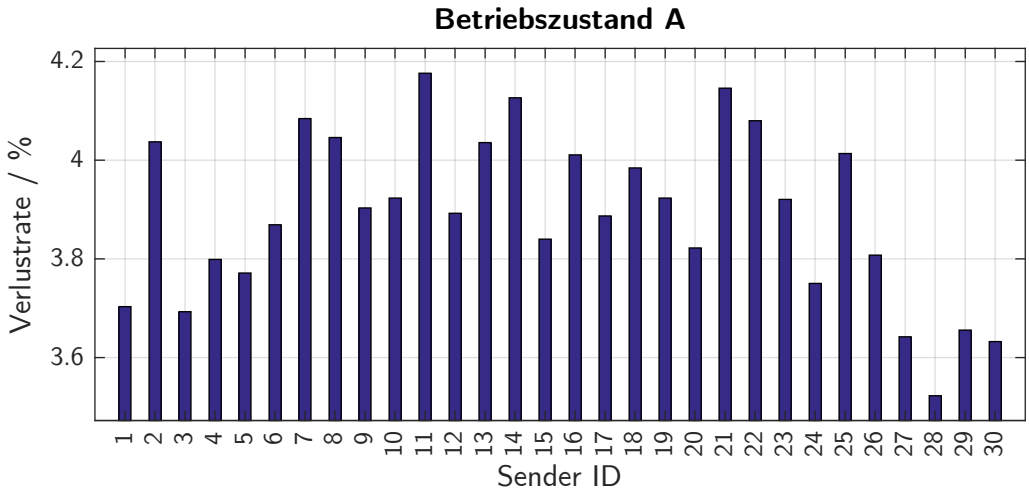


Abbildung 11: Empfangsverlustrate der Sender im Betriebszustand A

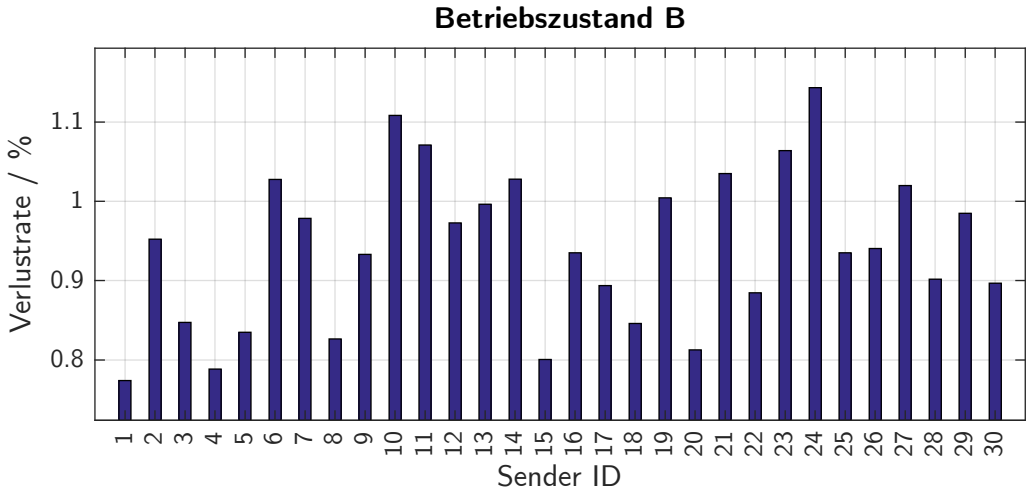


Abbildung 12: Empfangsverlustrate der Sender im Betriebszustand B

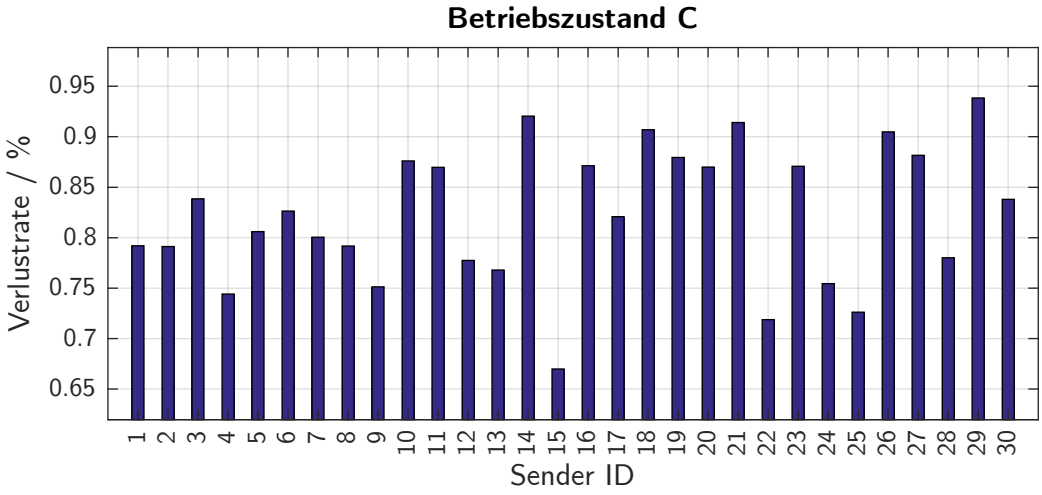


Abbildung 13: Empfangsverlustrate der Sender im Betriebszustand C

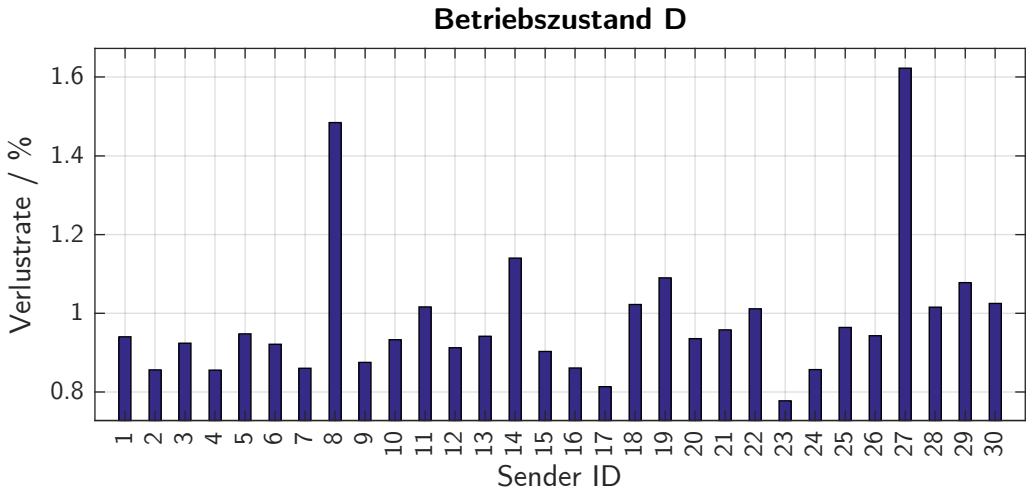


Abbildung 14: Empfangsverlustrate der Sender im Betriebszustand D

Der größte Verlust entstand demnach durch zu kurze Sendeintervalle. Eine Verfünffachung des Sendeintervalls senkte den maximalen Verlust auf etwa 1,2%, weniger als ein Drittel des Ausgangswertes. Ein längeres Ausleseintervall senkte die maximale Verlustrate nur leicht auf knapp unter ein Prozent, lieferte damit aber auch das beste Ergebnis aller Betriebszustände. Beim Ausfall von Empfängern verdoppelt sich die Verlustrate bezüglich Zustand C fast, wie Zustand D zeigt. Trotzdem bleibt die Verlustrate noch unter der zwei Prozent Schwelle. Außerdem fallen, wie Abbildung 14 zeigt, Sender 8 und 27 mit vergleichsweise hohen Verlusten auf. Ursache dafür kann die Positionierung beim Test sein. Der Durchschnittsverlust in Zustand D überschreitet, im Gegensatz zum Maximalwert, nur knapp die ein Prozent Schwelle. Das zeigt, dass durch Empfängerausfall das System nicht überlastet werden kann. Abschließend lässt sich sagen, dass ein langes Sende- und daran angepasstes Empfangsintervall die niedrigste Nachrichtenverlustrate bewirkt. Daher wurden die Einstellungen aus Betriebszustand C für den Regelbetrieb übernommen.

6.2 Stromverbrauch und Batterielaufzeit

Um den Stromverbrauch der Sender zu bestimmen, dient ein Messwandler auf der Grundlage des Präzisionsoperationsverstärkers AD8538.

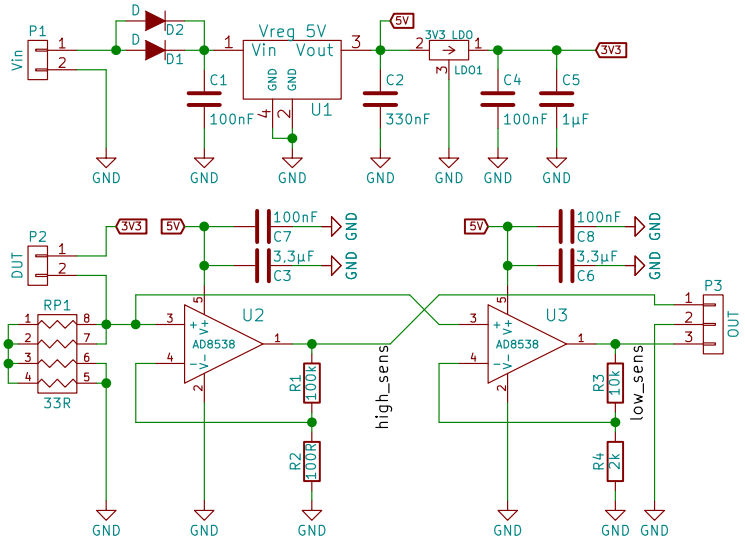


Abbildung 15: Stromlaufplan des Strommesswandlers

Abbildung 15 zeigt den Aufbau des Messwandlers. Oben befinden sich die Spannungsregler und Stützkondensatoren zur Erzeugung der Betriebsspannungen. 3,3V für den zu untersuchenden Sender und 5V für die Verstärkerschaltung. Diese teilt sich in zwei Empfindlichkeitsbereiche auf, da der erwartete Stromfluss während des Sendevorgangs um einige Magnituden größer ist, als der Ruhestrom. Gemessen wird der Strom über einen 33Ω Messwiderstand. Dies ist der größtmögliche Wert, bei dem ein fehlerfreier Sendebetrieb möglich ist. Er ist experimentell bestimmt. Der Spannungsabfall über RP1 wird von zwei separaten, nicht invertierenden Verstärkern mit unterschiedlichen Verstärkungsfaktoren aufgegriffen. Die Verstärkung eines Operationsverstärkers berechnet sich über das Rückkopplungsverhältnis, kann also aus dem Spannungsteiler im Rückkopplungszweig bestimmt werden. Da der OPV bestrebt ist die Eingangsspannungsdifferenz zu minimieren, muss im stationären Fall zwischen den Eingangsspannungen nicht unterschieden werden. Am Beispiel des ersten Verstärkers (U2) gilt:

$$r_V = \frac{U_{aus}}{U_{ein}} = \frac{R_1 + R_2}{R_2} \tag{10}$$

Aus der Schaltung ergeben sich Verstärkungsfaktoren von:

$$r_{V1} = \frac{100\,k\Omega + 100\,\Omega}{100\,\Omega} = 1001 \qquad r_{V2} = \frac{10\,k\Omega + 2\,k\Omega}{2\,k\Omega} = 6$$

Da reale Bauteile jedoch toleranzbehaftet sind, wurden die Widerstände nachgemessen, und mit den zugehörigen Toleranzen in Tabelle 5 festgehalten. (Voltcraft VC170-1)

R_{Ideal}/Ω	Messtoleranz	R_{Mess}/Ω	$R_{min,tol}/\Omega$	$R_{max,tol}/\Omega$
33	±1,6% ; 3Counts	32,7	31,8	33,5
100	±1,6% ; 3Counts	99,7	98,1	101,9
2000	±1,3% ; 2Counts	1.989	1.961	2.017
10.000	±1,3% ; 2Counts	9.950	9.800	11.000
100.000	±1,3% ; 2Counts	99.500	98.000	101.000

Tabelle 5: Toleranzbereiche der Widerstände

Daraus ergeben sich im Extremfall folgende Verstärkungsfaktoren:

$$r_{V1} = 962{,}73 \ldots 1030{,}56 \qquad r_{V2} = 2{,}86 \ldots 6{,}61$$

Nach dem Ohm'schen Gesetz berechnet sich die Verstärkereingangsspannung U_{ein} durch

$$U_{ein} = U_{Mess} = I_{Sender} \cdot R_{Mess} \tag{11}$$

Demnach ist die Verstärkerausgangsspannung U_{aus}

$$U_{aus} = r_V \cdot R_{Mess} \cdot I_{Sender} \;\rightarrow\; I_{Sender} = \frac{U_{aus}}{r_V \cdot R_{Mess}} \tag{12}$$

Werden die extremsten Kombinationen der Messwiderstandswerte und Verstärkungsfaktoren verrechnet, sowie die Ausgangsspannung des Messwandlers auf 1 Volt festgelegt, so ergeben sich folgende Skalierungsfaktoren S und Toleranzen:

Für den empfindlichen Bereich:

$$1V \mathrel{\hat{=}} 30{,}8\,\mu A \pm 1{,}9\,\mu A \;\rightarrow\; S_{empf} = 30{,}8\,\frac{\mu A}{V} \pm 1{,}9\,\mu A$$

und für den unempfindlicheren Bereich:

$$1V \mathrel{\hat{=}} 4{,}94\,mA \pm 0{,}43\,mA \;\rightarrow\; S_{unempf} = 4{,}94\,\frac{mA}{V} \pm 0{,}43\,mA$$

Der praktische Messaufbau sieht folgendermaßen aus:

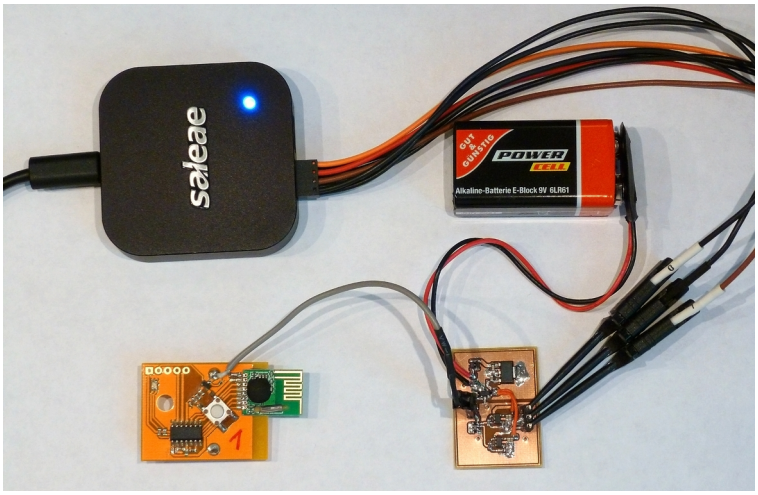


Abbildung 16: Aufbau der Strommessschaltung

Der Messwandler (Abb.16 unten rechts) wird von einer neun Volt Batterie (oben rechts) gespeist, wodurch eine sehr rauscharme Stromversorgung sichergestellt ist. Von dem Sender (unten links) der an dem DUT-Stecker (Device under Test) des Messwandlers angeschlossen ist (siehe Abbildung 15), wurden zuvor alle Stützkondensatoren entfernt. Dies verhindert, dass die Betriebsspannung lokal gepuffert und der fließende Strom verfälscht wird. Am Ausgang des Messwandlers ist ein USB Oszilloskop (Abb.16 oben links) angeschlossen. Es hat eine Bandbreite von 1MHz und tastet mit 10Bit Auflösung ab. Aus den gemessenen Werten wurden folgende Stromverläufe berechnet:

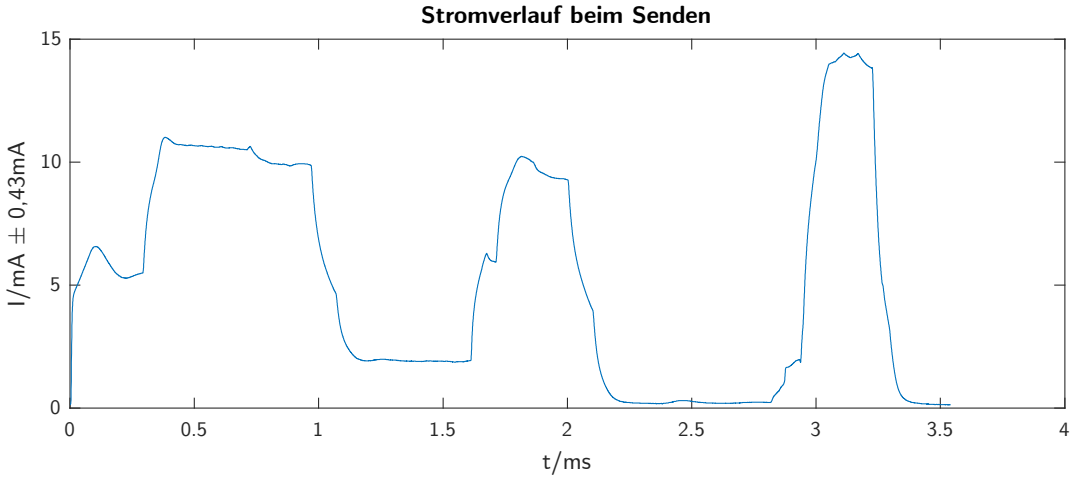


Abbildung 17: Sendestromverlauf

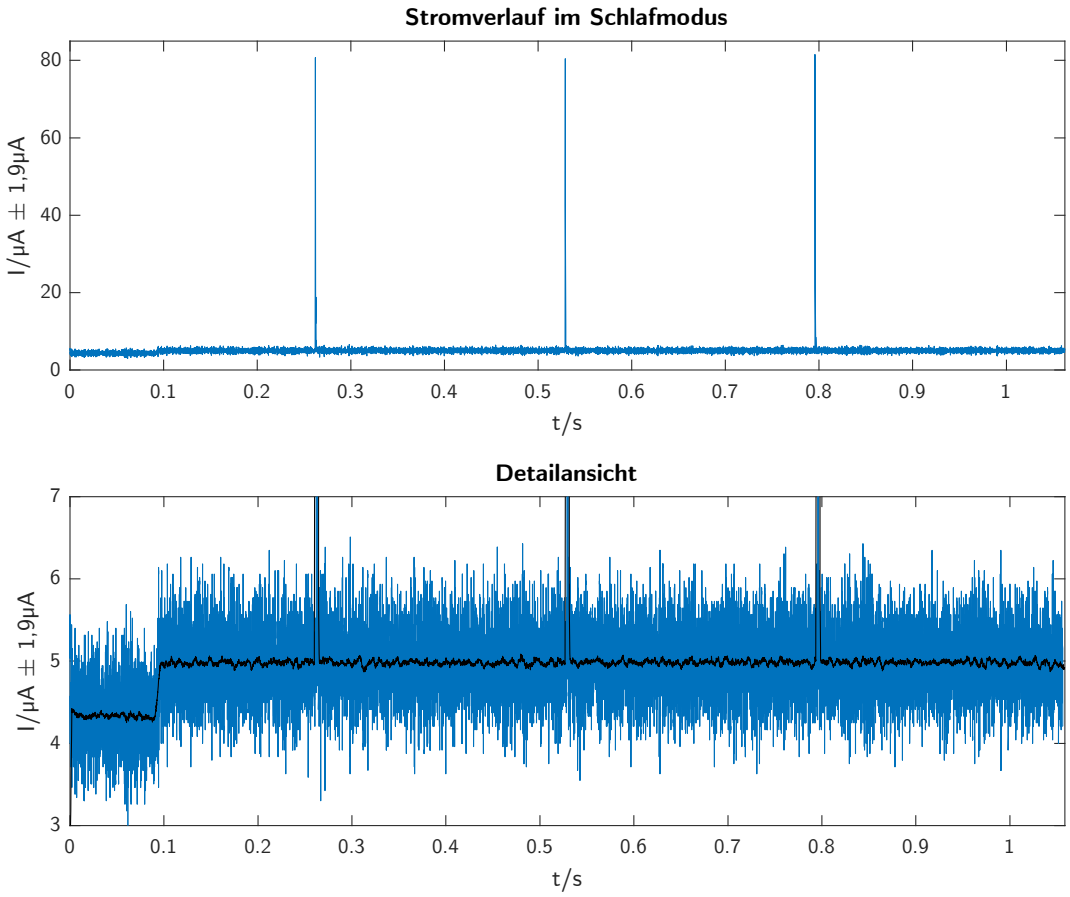


Abbildung 18: Stromverlauf im Schlafmodus, tiefpassgefilterter Verlauf überlagert

Ein Matlabscript liest die gemessenen Spannungsverläufe und berechnet daraus zunächst den Effektivwert des Stromes für das Intervall T. S ist hierbei der Skalierungsfaktor zur Umrechnung der gemessenen Spannung in den Strom.

$$I_T = T^{-1} \cdot \sum_{k=0}^{N} U(k) \cdot \Delta t \cdot S \tag{13}$$

Anschließend wird daraus die durchschnittliche Stromaufnahme berechnet:

$$\bar{I} = \frac{I_{Tx} \cdot t_{Tx} + I_{Sleep} \cdot t_{Sleep}}{t_{Tx} + t_{Sleep}} \tag{14}$$

Für die eingestellte Senderate von etwa einer Nachricht pro Sekunde ergibt sich ein durchschnittlicher Stromverbrauch von 22,39µA ± 1,9µA. Werden bei einer Nennladung der Batterie von 210mAh etwa 200mAh als praktisch nutzbar angenommen, so berechnet sich eine

Betriebszeit von 343 Tagen. Zur Berechnung wurde die Toleranz zum Nachteil der Betriebszeit einbezogen. Bei erhöhtem Nachrichtenintervall von fünf Sekunden fließt nur noch ein effektiver Strom von 8,47µA ± 1,9µA. Damit lässt sich eine Laufzeit von etwa 2,2 Jahren erzielen.

Als Plausibilitätskontrolle soll der theoretische Ruhestrom mit den Werten aus den Bauteildatenblättern berechnet werden. Tabelle 6 zeigt die einfließenden Ruhe- und Leckströme:

Verbraucher	**Bezeichnung**	**Bedingung**	$\mathbf{I_{leak}/\mu A}$	**I/µA**
PMOS	PMV160UP	V_{GS}=-8V, 25°C	0,1	-
TVS	ESD9B3.3ST5G	25°C	0,1	-
µC: WDT	PIC16F1705	3V, 25°C	-	0,8
µC: LPBOR	PIC16F1705	3V, 25°C	-	0,5
µC: ADC Idle	PIC16F1705	3V, 25°C, RC Oszillator	-	0,08
µC: Pinleakage	PIC16F1705	3V, 25°C	0,01	-
µC: Grundverbrauch	PIC16F1705	3V, 25°C	-	0,08
Funkmodul	RFM75	Power Down Modus, 3V	-	3
Gesamtstrom	4,67µA			

Tabelle 6: Theoretische Ruhestromaufnahme des Senders

Der Ruhestrom laut Datenblättern liegt mit 4,67µA innerhalb des Toleranzbereiches des gemessenen Wertes von 5,31µA ± 1,9µA. Dieser stammt aus dem in Abbildung 18 gezeigten Datensatz. Zu beachten ist, dass es sich bei den gemessenen Werten nur um eine einzelne Stichprobe handelt. Da die Batterieladung jedoch auch von anderen Parametern wie Temperatur, Innenwiderstand, Selbstentladung und der Stromaufnahme des Senders, auch von Bauteiltoleranzen, Oszillatordrift und der Firmware abhängt, wird auf eine detailliertere Untersuchung verzichtet.

6.3 Reichweite und Empfängerplatzierung

Um möglichst lückenlose Funkabdeckung im Labor sicherzustellen, soll in diesem Kapitel die optimale Empfängeranordnung bestimmt werden. Dazu wird zunächst die ungefähre Sendereichweite in verschiedenen Situationen bestimmt. Da der Empfänger das Auslesen eines RSSI (received signal strength indicator) nicht unterstützt, muss mit einfachen, praxisnahen Abschätzungen gearbeitet werden. Sender und Empfänger stehen sich in etwa einem Meter Höhe und unterschiedlichen Entfernungen gegenüber, die Sendeleistung ist auf -18dBm eingestellt. Die in Tabelle 7 angegebenen Gegenstände befinden sich zwischen Sender und Empfänger:

	1m	2m	3m	4m	5m
Freiraum	✓	✓	✓	✓	✓
Pullover-/ Jackentasche	✓	✓	✓	✓	✓
Hosentasche	✓	✓	✓	✓	✓
Hosentasche + Schlüsselbund	✓	✓	✓	✓	✓
Hosentasche + Smartphone	✓	✓	X	X	X
Hosentasche + Portemonnaie mit Metall-Kartenhüllen	✓	✓	(✓)	(✓)	X
Hintere Hosentasche	(✓)	(✓)	X	X	X

Tabelle 7: Empfangsreichweite

✓: zuverlässiger Empfang

(✓): Empfang mit deutlichen Unterbrechungen

X: kein Empfang möglich

Zu sehen ist, dass in fast allen Situationen der Empfang in zwei Metern Entfernung problemlos möglich ist. Das Signal kann bei der eingestellten Sendeleistung von -18dBm nicht durch den Körper hindurch empfangen werden. Daher müssen die Empfänger so platziert werden, dass in jeder Orientierung einer direkten Sichtkontakt hat. In Abbildung 19 ist ein vereinfachter Grundriss des CSL (cognitive systems lab) mit ungefährer Position der Arbeitsplätze zu sehen:

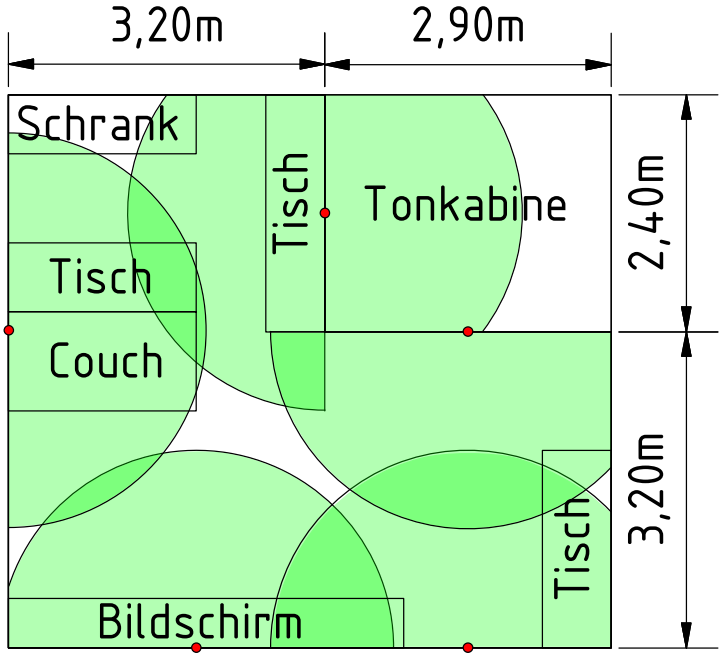


Abbildung 19: Grundriss des Labors

Anhand dieser Darstellung können Annahmen über die Aufenthaltsdauer in den Bereichen des Labors getroffen werden. Die Sitzplätze an den Tischen werden häufig genutzt. Die Aufenthaltsdauer vor dem Schrank wird gering sein. Der Bereich direkt vor dem Bildschirm wird ebenfalls nur kurz betreten, weil ein Mindestabstand zu betrachten nötig ist. In rot sind die geplanten Positionen der Empfänger eingezeichnet, in grün die Empfangsbereiche mit zwei Metern Radius. Einige Stellen werden in dieser Entfernung nicht abgedeckt. Der Schatten im Bereich des Schrankes kann ausgenommen werden. Die Tonkabine ist mit Lochblech ausgekleidet, an dem das Funksignal gut reflektiert wird. Auch sind durch die Abschirmung die äußeren Störeinflüsse reduziert. Mit einem Empfänger am Sichtfenster der Kabine konnten Funksignale von beiden Arbeitsplätzen innen empfangen werden. Vom Funkschatten mittig im Raum besteht Sichtkontakt zu allen fünf Empfängern. Wenn das Signal in eine Richtung nur schwach gedämpft wird, kann es trotz der höheren Entfernung von einem der Module erkannt werden. Stellt sich im Betrieb heraus, dass die Erfassung unzuverlässig ist, so können die Erkennungsschwellen gesenkt und/oder die Sendeleistung auf -12, beziehungsweise -7dBm erhöht werden. Letztere Option wirkt sich jedoch negativ auf die Batterielebensdauer aus, weshalb davon abgeraten wird.

6.4 Programmoptimierung

Drei wichtige Programmabschnitte sind die Ver- und Entschlüsselung, sowie das Löschen von doppelten Nachrichten. Da sie zeitkritisch sind und zumindest die Verschlüsselung auf dem batteriebetriebenen Sender läuft, ist eine kurze Laufzeit wichtig. So kann der reibungslose Ablauf gewährleistet und die Batterieladung besser ausgenutzt werden.

6.4.1 Verschlüsselung

Grundlage des Verschlüsselungsprogramms ist die 3way Implementation aus Bruce Schneiers Buch „Angewandte Kryptografie“. Dieses ist für 32Bit Prozessoren optimiert und hat auf dem acht Bit PIC16F1705 des Senders eine lange Ausführzeit.

In einem Testprogramm werden die originale und die auf acht Bit optimierte Implementation gegenübergestellt. Der PIC16F1705 wird mit 16MHz getaktet, durch die Ausführungsweise wird dieser geviertelt. Die Befehle werden also mit 4MHz ausgeführt. Zur Zeitmessung dient der „Saleae Logic 8“ Logikanalysator, der auch zur Messung der Stromaufnahme zum Einsatz kam. Mit 100MS/s werden die Pins der roten und grünen LED abgetastet, die zu Beginn und Ende der zu untersuchenden Programmabschnitte kurz ein- und wieder ausgeschaltet werden. Tabelle 8 zeigt die Laufzeitreduktion der einzelnen Funktionen:

	Originale Implementation	8 Bit optimierte Version	Reduziert um
Gesamtlaufzeit	19,05ms	1,616ms	91,52%
Gamma	53,65µs	23,2µs	56,76%
Theta	1205µs	32,2µs	97,33%
Pi 1	133,8µs	27,7µs	79,30%
Pi 2	133,8µs	25,7µs	80,79%

Tabelle 8: Laufzeitreduktion der Verschlüsselung

Diese Ergebnisse sind erreichbar, indem statische Werte im Voraus berechnet und in Datenfeldern hinterlegt werden. Zum Beispiel werden die Rundenkonstanten im Originalprogramm vor jeder Verschlüsselung neu berechnet, obwohl sie sich nicht ändern. Weiterhin wurden alle 32Bit Variablen durch mehrere 8Bit Variablen ersetzt. Hier am Beispiel der Theta-Funktion (Ausschnitt):

```
b[0] = a[0] ^ (a[0]>>16) ^ (a[1]<<16) ^ (a[1]>>16) ^ (a[2]<<16) ^
       (a[1]>>24) ^ (a[2]<<08) ^ (a[2]>>08) ^ (a[0]<<24) ^
       (a[2]>>16) ^ (a[0]<<16) ^ (a[2]>>24) ^ (a[0]<<08);
```

Abbildung 20: Originale Implementation

```
b00 = a00 ^ a02 ^ a12 ^ a13 ^ a21 ^ a22 ^ a23;
b01 = a01 ^ a03 ^ a13 ^ a20 ^ a22 ^ a23 ^ a00;
b02 = a02 ^ a10 ^ a20 ^ a21 ^ a23 ^ a00 ^ a01;
b03 = a03 ^ a11 ^ a21 ^ a22 ^ a00 ^ a01 ^ a02;
```

Abbildung 21: Optimierte Implementation

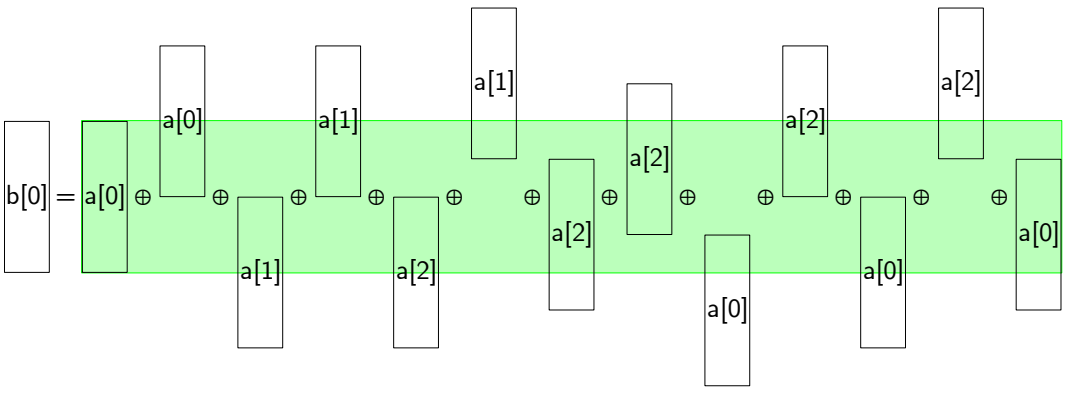


Abbildung 22: Schematische Darstellung eines Teils der Theta-Funktion

Durch die Zerlegung in kleinere Variablen können im Beispiel sämtliche Schiebeoperationen eliminiert werden, die der PIC16F1705 einzeln ausführen müsste, da er keinen Barrelshifter besitzt. Wie in Abbildung 22 zu sehen, ist nur ein Wert nicht verschoben. Dieser gibt den Bereich an, in dem tatsächlich gerechnet werden kann (grün markiert). Alle anderen Doppelwörter werden so verschoben, dass insgesamt die Hälfte aller Bytes außerhalb dieses Bereiches liegen, und mit Null Exklusiv-Oder verknüpft werden. Diese unnötigen Rechnungen entfallen in der optimierten Version. Der letzte große Unterschied ist, dass in der Originalversion die Werte im Feld b[0...2] nur zwischengespeichert und zum Ende der Funktion in das Feld a[0...2] zurück transferiert werden. In der optimierten Version werden die Werte zwischen den Variablensätzen a und b hin und her verschoben, weshalb der Klartext in a00...a23 vorliegen muss und dann verschlüsselt in b00...b23 zu finden ist. Der Klartext bleibt in beiden Implementationen nicht erhalten.

6.4.2 Entschlüsselung

Zur Entschlüsselung wird zunächst die Bitreihenfolge des Chiffrats und des Schlüssels mit der My-Funktion umgekehrt. Während dies im Original durch Schieben und Maskieren geschieht, wurde für die Optimierung eine Tabelle angelegt, in der jeder Index sein eigenes Inverses enthält. Auch die Theta-Funktion wird auf den Schlüssel angewendet. Beides wird vor dem Start der Hauptprogrammschleife für alle durchgeführt und die Originalschlüssel im RAM überschrieben. Da jeder Sender einen eigenen Schlüssel besitzt, muss der richtige ausprobiert werden. Da sich der Vorgang mehrfach wiederholen kann, wird die My-Funktion auf das Chiffrat angewendet und das Ergebnis separat gespeichert. Da die Entschlüsselung ab diesem Punkt fast identisch zur Verschlüsselung abläuft, werden die Eingangsdaten wieder überschrieben. Ist der Schlüssel falsch, kann die bitweise invertierte Nachricht aus dem Zwischenspeicher wiederhergestellt und der Nächste getestet werden. Tabelle 9 zeigt die gemessenen Zeiten:

Funktion	Benötigte Zeit
Twiddle Keys: $K'_n = \mu(\theta(K_n))$; $n \in [0,29]$	879,4µs
Twiddle Message: $M_{Reverse} = \mu(M_{Input})$	17,91µs
Decrypt: $M_{Decrypted} = decrypt(M_{Reverse}, K'_n)$	246,8µs

Tabelle 9: Laufzeiten der Entschlüsselungsroutinen

In dem Fall, dass alle Schlüssel probiert werden müssen, ergeben sich folgende Gesamtlaufzeiten:

$$T_{unoptimiert} = 879{,}4\,\mu s + 30 \cdot 17{,}91\,\mu s + 30 \cdot 246{,}8\,\mu s = 8{,}82\,ms$$
$$T_{optimiert} = 17{,}91\,\mu s + 30 \cdot 246{,}8\,\mu s = 7{,}42\,ms$$

Da die Laufzeit von der bereits optimierten Entschlüsselungsdauer dominiert wird, ist die Verbesserung nur gering. Die Laufzeitreduktion beträgt etwa 15,9%.

6.4.3 Löschen doppelter Nachrichten

Dieser Prozess ist bereits in Kapitel 5.2 erwähnt worden. Das Entfernen doppelter Nachrichten soll den Datenverkehr insgesamt und den Rechenaufwand des Steuermoduls senken. Je weniger Zeit dieses Programm benötigt, desto schneller wird auch die Übertragung der Nachrichten. Für die Zeitmessungen wurden alle Speicherslots mit der gleichen Nachricht belegt.

```
for (c = 0:29)                                                          Äußere Schleife: 1,46µs
{
   if (BUFFER[c] != EMPTY)                                              Validierung: 7,34µs + c · 361,4ns
   {
      if (BUFFER[c].CRC == TEMP.CRC)                                    CRC Vergleich: 16,29µs + c · 724,14ns
      {
         for (i = 0:11)
         {
            if (BUFFER[c].DATA[i] != TEMP.DATA[i]) goto NEXT_SLOT;      MSG Vergleich: 133,5µs + c · 4,71µs
         }
         BUFFER[c] = EMPTY;
      }
      Label: NEXT_SLOT;
   }
}
```

Abbildung 23: Nicht optimierte Suche nach doppelten Nachrichten

Der Ablauf ist in Pseudocode in Abbildung 23 zu sehen. Alle Speicherslots werden zunächst auf gültigen Inhalt hin überprüft. Stimmt das CRC Restpolynom mit dem der Vergleichsnachricht im Buffer „TEMP“ überein, so wird auch die restliche Nachricht verglichen. Sind beide identisch, so wird der Slot als leer markiert. Bei jeder Abweichung wird sofort zum nächsten Slot gesprungen. Aus der Abbildung ist bereits ersichtlich, dass viele Zeiten von der Vergleichsanzahl abhängen. Ursache dafür ist die Realisierung des Speicherzugriffs über eine Multiplikation der Laufindizes c und i mit einem Offset. Der PIC16F1619 besitzt keinen Hardwaremultiplizierer, weswegen diese Zugriffe sehr langsam sind.

```
for (c=0:29)                                                            Äußere Schleife: 1,46µs
{
   INST = &BUFFER[c];                                                   Instanziierung: 7,34µs + c · 361,4ns

   if (INST != EMPTY)                                                   Validierung: 1,34µs
   {
      if (INST->CRC == TEMP.CRC)                                        CRC Vergleich: 3,04µs
      {
         if (INST->DATA[ 0] != TEMP.DATA[ 0]) goto NEXT_SLOT;           MSG Vergleich: 17,54µs
         ⋮
         if (INST->DATA[11] != TEMP.DATA[11]) goto NEXT_SLOT;
         INST = EMPTY;
      }
      Label: NEXT_SLOT;
   }
}
```

Abbildung 24: Optimierte Suche nach doppelten Nachrichten

Um dieses Problem zu umgehen, wird in der optimierten Implementation in jeder Runde ein Zeiger auf den aktuellen Speicherslot gesetzt und die innere Schleife „hard coded“, also jeder Durchlauf statisch aufgelistet. Dadurch muss pro Speicherslot nur eine Multiplikation ausgeführt werden. Dadurch wurde eine Zeitsenkung von 8,38ms auf 1,1ms erzielt, was einer Reduktion um 86,8% entspricht. Beide Messungen stellen den Extremfall von 30 identischen Nachrichten dar, der aber praktisch nie eintritt.

7 Auswertung

Diese Arbeit dokumentiert die Auswahl einer geeigneter Technologie, sowie der passenden Bauteile zur Umsetzung eines Zugangsüberwachungssystems. Die Entwicklung der Schaltungen wurde beschrieben und die jeweilige Ausführungsweise begründet. Prototypen und überholte Hardwareversionen sind im Anhang dokumentiert. Die Funktionsweise und der Ablauf aller im Zuge der Arbeit entwickelten Programme wurde erklärt. Weiterhin werden die ergriffenen Maßnahmen zur Laufzeitoptimierung einiger Programmabschnitte detailliert dargelegt. In praktischen Versuchen konnten die optimalen Betriebsparameter bestimmt und das Verhalten bei Empfängerausfall untersucht werden. Der tatsächliche Stromverbrauch des Senders konnte durch den Aufbau eines Messwandlers gemessen und auf Grund dessen, die voraussichtliche Batterielebensdauer abgeschätzt werden. Verschiedene Alltagsgegenstände wurden auf ihre Fähigkeit hin, das Funksignal zu dämpfen, untersucht. Die Ergebnisse ermöglichten die Planung einer günstigen Empfängerplatzierung im Labor. Das Vorliegende System wurde unter Laborbedingungen getestet, ein Praxistest steht noch aus. Es ist nach bisherigen Erkenntnissen in der Lage, Personen binnen 20s nach Betreten des Labors auf der Nutzeroberfläche als „Anwesend“ anzuzeigen. Die Funktionsweise hat sich als zuverlässig herausgestellt. Fehlerzustände, wie das Trennen der Netzwerkleitung oder einzelner Empfänger, führten nicht zu unerwarteten Fehlfunktionen. Die genannten Gründen lassen darauf schließen, dass das vorliegende System zur Raumüberwachung die gestellten Anforderungen erfüllt. Weiterhin lassen sich einzelne Komponenten als Grundlage für weitere Laborsysteme nutzen. Das vorgelegte Steuermodul, in Verbindung mit dem Raspberry Pi ist eine erweiterbare Schnittstelle zum Raumsystem des CSL.

8 Ausblick

Trotz sorgfältiger Überlegung bei der Auswahl der Hardware, erwies sich insbesondere die Wahl des Funkmoduls RFM75 als Fehler. Dieser konnte zwar alle gewünschten Aufgaben dieses Projektes bewältigen, ist jedoch funktional mangelhaft. Besonders mit dem Tiefschlafmodus traten Probleme auf, die nur durch eine umständliche Bitmanipulation in einigen undokumentierten Registern gelöst werden konnten. Auch der Stromverbrauch ist höher als der des NRF24l01 von Nordic Semiconductors, wie bereits in Kapitel 3.3 angemerkt. Für jede weitere Generation von Sendern ist daher von der Verwendung des RFM75 Funkmoduls abzuraten. Zur Funktionserweiterung sollten auch andere Funkmodule in Erwägung gezogen werden. So stellt der MRF24XA von Microchip auch einen RSSI (received signal strength indicator) zur Verfügung und der Kinetis KW20Z von NXP vereint Funkmodul und ARM Cortex M0+ Prozessor in einem Chip.

Die Zuordnung der Nachrichten zu den betreffenden Empfängern wurde zwar versucht nachträglich zu implementieren, ist aber mit viel Aufwand verbunden. Das Busprotokoll ist auf geringen Overhead ausgelegt und schränkt deshalb die Skalierbarkeit erheblich ein. Momentan ist der Adressraum auf 15 Empfänger begrenzt. Das Packen und Entpacken von Busnachrichten ist umständlich, da der Header in einzelne Bits zerlegt werden muss. Besser sind leistungsstärkere Mikrocontroller in den Empfängermodulen, die mit einem höheren Bustakt einen größeren Overhead ausgleichen können. Insbesondere Funktionserweiterungen würden dadurch begünstigt.

Das Verschlüsselungssystem ist ausbaufähig. Ein Sicherheitskonzept für aktive RFID Systeme wurde von japanischen Wissenschaftlern vorgestellt. Größter Mangel an der bestehenden Technik sei demnach, dass bei geringer Verbreitung aktiver RFID-Systeme die Existenz der Funknachrichten zur Ortung oder anderem Missbrauch verwendet werden kann. Weil Nachrichten mitgelesen werden können, muss eine Verschlüsselung obligatorisch sein. Ein gänginger Angriff auf solche Systeme ist die Replayattacke, bei der eine Funknachricht gespeichert und wieder gesendet wird. Als Lösung wird vorgeschlagen, das Lesegerät Anfragen aussenden zu lassen, die von den Identifikationsgeräten empfangen und beantwortet werden. So wird nur gesendet, wenn ein Sender es empfangen kann. Mit der Anfrage kann ein Sitzungsschlüssel gesendet und so eine Replayattacke vereitelt werden [Yamada, 2005, S.2-4]. Die Realisierung dieser Vorschläge erfordern einen größeren Zeit- und Entwicklungsaufwand als im Zuge dieser Arbeit aufzubringen ist. Das größte Problem dabei ist die hohe Stromaufnahme des Funkmoduls im Empfangsbetrieb. Die bislang verwendeten 210mAh Batterien und die RFM75 Funkmodule wären so nur maximal 13h betriebsfähig.

Literaturverzeichnis

Bücher

R. Klostermeyer, „Digitale Modulation“, Vieweg, 1. Auflage, 2001, ISBN 978-3-528-03909-7

B. Schneier, „Angewandte Kryprografie“, Addison Wesley, 1996, ISBN 978-3-893-19854-2

M. Kawulok, M. E. Celebi, B. Smolka, „Advances in Face Detection and Image Analysis“, Springer, 2013, ISBN 978-3-319-25956-7

G. Tamm, C. Tribowski, „RFID“, Springer, 2010, ISBN 978-3-642-11460-1

K. W. Kark, „Antennen und Strahlungsfelder“, Springer Vieweg, 6. Auflage, 2017, ISBN 978-3-658-13964-3

Paper und Zusammenschriften

D. Rudolph, M. Rudolph, Vorlesungsskript „Modulationsverfahren (analog und digital)“, 2011,

http://diru-beze.de/modulationen/skripte/index.html, Zugriff 11.01.2018, 20:45 Uhr

J. Daemen, R. Govaerts, J. Vandewalle, „A New Approach To Block Cipher Design“, 1994

J. Kelsey, B. Schneier, D. Wagner, „Related-Key Cryptanalysis of 3-WAY, Biham-DES,CAST, DES-X, NewDES, RC2, and TEA“, 1997

I. Yamada, S. Shiotsu, A. Itasaki, S. Inano, K. Yasaki, M. Takenaka, „Secure Active RFID Tag System“ 2005

A. Arbit, Y. Livne, Y. Oren, „Implementing public-key cryptography on passive RFID tags is practical“, Springer, 2014

Internetquellen

atlasRFIDstore, Lesegeräte für passive RFID Systeme: https://www.atlasrfidstore.com/rfid-readers/, Zugriff 23.01.2018, 12:30Uhr

atlasRFIDstore, Lesegeräte für passive RFID Systeme: https://www.atlasrfidstore.com/active-rfid/, Zugriff 23.01.2018, 12:30Uhr

RFID Insider, S. Smiley, „Active RFID vs. Passive RFID: What‘s the Difference?“, Veröffentlicht 4.03.2016, https://blog.atlasrfidstore.com/active-rfid-vs-passive-rfid, Zugriff: 3.11.2017, 13:40 Uhr

SkyRFID, RFID Systeme und deren Reichweite, http://skyrfid.com/RFID_Tag_Read_Ranges.php, Zugriff 3.11.2017, 13:30 Uhr

raspberrypi.org, Hardwareübersicht: https://www.raspberrypi.org/learning/hardware-guide/components/raspberry-pi/, Zugriff: 18.01.2018, 16:40 Uhr

Applikationsbeschreibungen und Datenblätter

Frequenzplan, Bundesnetzagentur, Stand: April 2016,
https://www.bundesnetzagentur.de/DE/Sachgebiete/Telekommunikation/Unternehmen_Institutionen/Frequenzen/Grundlagen/Frequenzplan/frequenzplan-node.html, Zugriff: 4.11.2017, 9:00 Uhr

DS51795B, „PICkit 3 Programmer/Debugger User's Guide ", 2010,
http://ww1.microchip.com/downloads/en/DeviceDoc/51795B.pdf, Zugriff: 7.12.2017, 16:30 Uhr

DS50002053G, „MPLAB XC8 C Compiler User's Guide ", 2016,
http://ww1.microchip.com/downloads/en/DeviceDoc/50002053G.pdf, Zugriff: 7.12.2017, 17:00 Uhr

AN043, „Small Size 2.4 GHz PCB antenna", 2008,
http://www.ti.com/lit/an/swra117d/swra117d.pdf, Zugriff: 4.11.2017, 15:20 Uhr

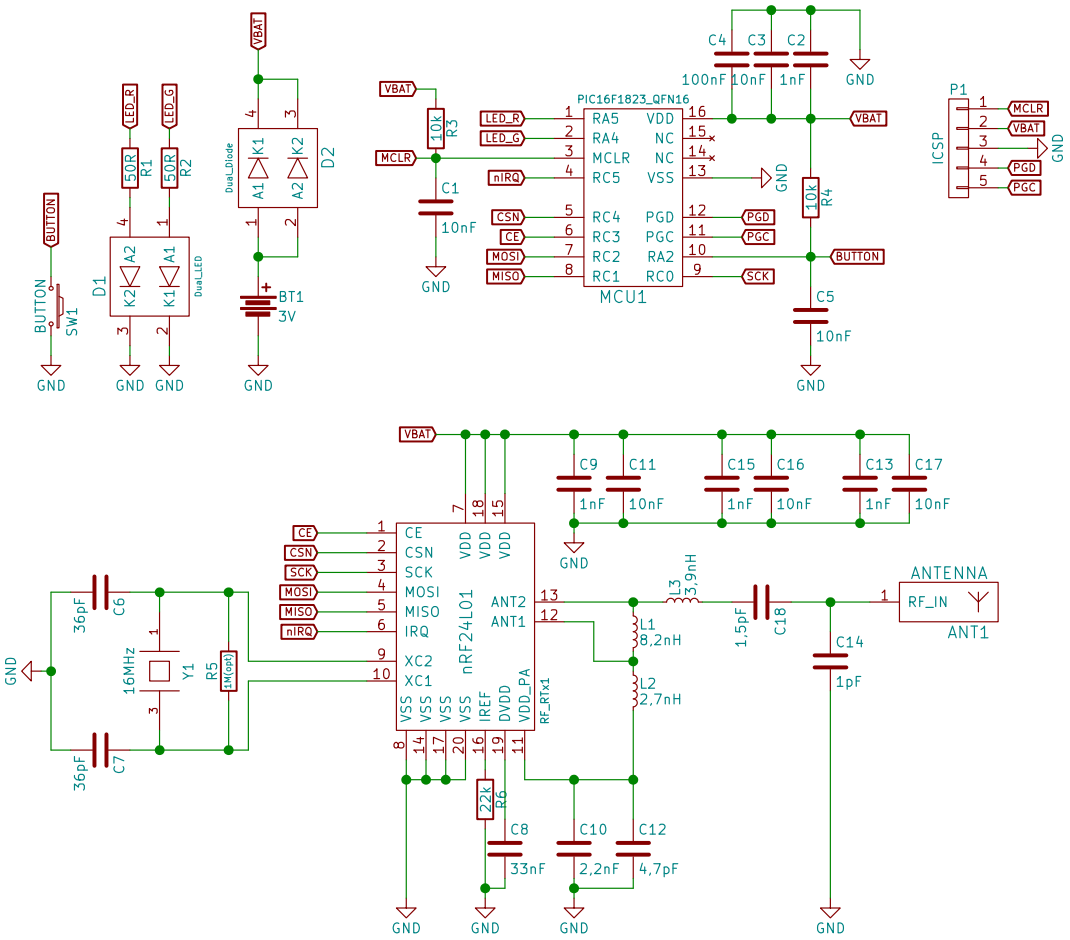
Anhang

Anhangsverzeichnis

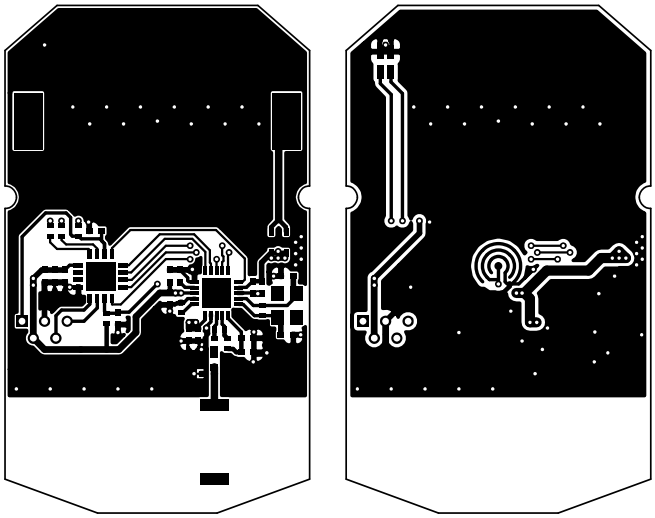
Anhangsabbildungsverzeichnis

1 Versionsverlauf des Senders

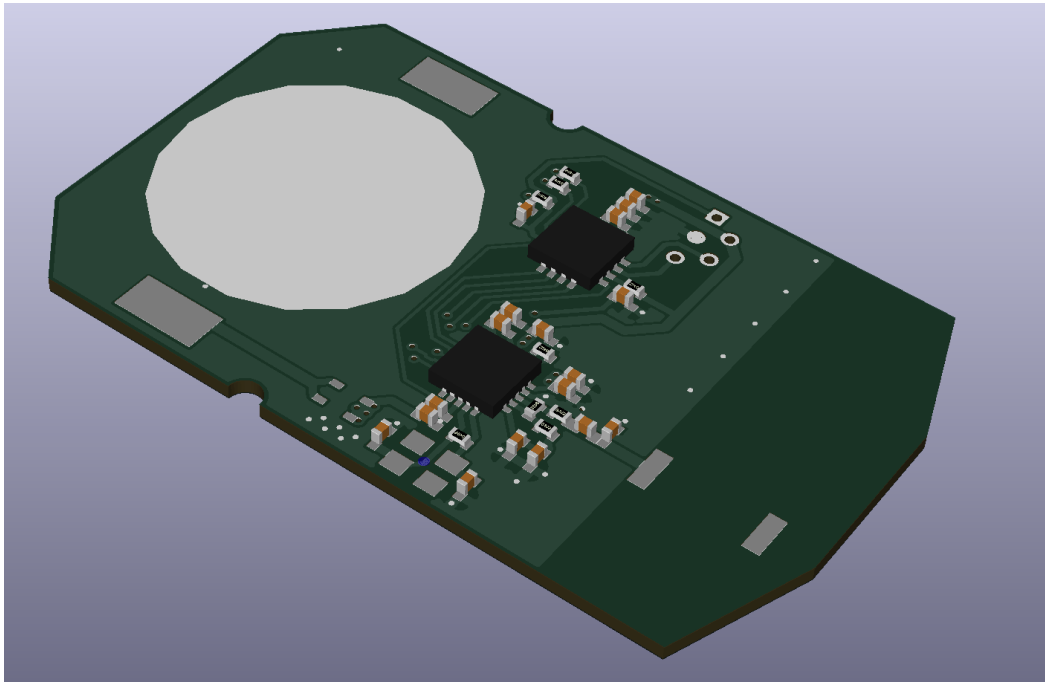
1.1 Sender Version 1



Anhang Abbildung 1: Schaltplan Sender Version 1



Anhang Abbildung 2: Platine, links: Unterseite, rechts Oberseite



Anhang Abbildung 3: 3D Darstellung der Platinenunterseite

Aufbau der kompletten Transceiverschaltung um den nRF24L01+. (Funkcontroller von Nordic Semiconductors)

-Keramik-Chipantenne

-Schnappscheibentaster/ Graphit-Gummi Taster

-Zwei parallele Dioden als Verpolschutz mit geringer Vorwärtsspannung

Stand: fertiges Platinenlayout, komplette Materialliste

Vorteile: robuster Aufbau (nur 1 Platine), weniger Maßvorgaben da frei platzierbare Bauteile

Nachteile: hohe Kosten, unzureichender Lagerbestand des Gehäuses mit langen Nachbestellzeiten

1.2 Sender Version 2

-statt nRF24L01+ den Funkchip RFM75 von HopeRF verwenden (nicht das gleichnamige Funkmodul). Ablöten der QFN-Gehäuse von den Funkmodulen oder Direktbestellung

-evtl. gedruckte Antenne

Stand: Konzept, Layout ähnlich wie V1

Vorteile: wie V1, etwas geringere Kosten

Nachteile: sehr umständlich, Quelle für einzelne RFM75 Funkchips nur über Aliexpress/ Alibaba (China)

1.3 Sender Version 3

-günstiges Transceivermodul mit RFM75 in COB-Technik (Chip on Board → nicht ablötbar) von Pollin

-Gehäuse von farnell

-PIC16F1823

-Pmos als Verpolschutz (geringerer Spannungsabfall → längere Batterienutzdauer)

-aufgelöteter Taster

-4-Pin Anschluss zu Testzwecken

Stand: gefertigt

Vorteile: vergleichsweise günstig, vertrauenswürdige Händler, geringere Versandkosten

Nachteile: Fehler im Layout, unzureichender Speicher zur Implementierung der Verschlüsselung, weniger Robust durch überhängendes Funkmodul (Gegenmaßnahme: verkleben), Taster zu hoch: Gehäuseteil kürzen

1.4 Sender Version 4

rev A: (Prototyp)

-PIC16F1825: mehr Speicher, ähnlicher Funktionsumfang wie PIC16F1823 aber familienfremd

-Verdrahtungsfehler einer V3 Platine mit Lackdraht korrigiert

-einige Widerstandswerte angepasst → geringerere parasitäre Ströme

-Test der bestehenden Hardware, Test der Verschlüsselung

Stand: gefertigt

Ziel: auffinden aller Fehler/ Verbesserungsmöglichkeiten, Test der

rev B: (Fertigungsversion)

-gleiches Gehäuse und Funkmodul wie V3, folgende Änderungen:

-Platinendicke 0,6mm statt 1mm (Taster hatte keinen Spielraum im Gehäuse, zu leicht zu betätigen)

-LED an korrekte Position verschoben

-PIC16F1705: mehr Speicher, Pinremapping → simpleres Platinenlayout

-4Pin Testanschluss entfällt

-3 Bohrungen zur Ausrichtung des Transceivers beim aufkleben

-geschlossene Lötstoppmaske über Programmieranschluss auf Batterieseite -> kein ungewollter Kontakt beim Batteriewechsel

-Beschriftung Batterieanforderungen

Stand: gefertigt

Vorteile: weiterverwendung der meisten Bauteile aus V3, mehr Speicherreserve für Erweiterungen

Nachteile: weniger Robust durch überhängendes Funkmodul (Gegenmaßnahme: verkleben)

2 Versionsverlauf des Empfängers

2.1 Empfänger Version 1

-RFM75 Transceiver

-PIC16F1823

-RS485 Transceiver

-Spannungsregler

-Passivbeschaltung

Stand: gefertigt

Vorteile: günstig

Nachteile: gleicher Layoutfehler wie im Sender V3: Busleitungen vertauscht, knapper Speicher der MCU, viele Durchkontaktierungen in der Platine (schlecht bei manueller Fertigung)

Funkmodul muss Steckerkräfte aufnehmen

2.2 Empfänger Version 2

Änderungen zu V1:

-PIC16F1619: Pinremapping, CRC in Hardware, mehr Speicher, 2 Pins für Ströme bis 100mA (statt 25mA), 6Pins mehr für zusätzliche Funktionen

-zweifarbige LED zur Statusanzeige

-Versorgungsspannung des Transceivers durch PIC geschaltet (Reset durch Software möglich)

-kaum Durchkontaktierungen und kompakterer Aufbau dank Pinremapping

-Transceiver anders positioniert → keine Lastaufnahme vom Stecker

-Platine lackiert mit Conformal Coating (Wasserschutz, Anlaufschutz)

Stand: gefertigt

Vorteile: fast alle Fehler behoben, einfachere Herstellung (Bohren, durchkontaktieren, bestücken), LED erleichtert debugging

Nachteile: kompliziertere Außenform, Spannungsabfall über Pmos im PIC zu hoch → zu wenig für Transceiver

Provisorische Reparatur: Beide Hochstrompins parallel, TxD Pin auf anderen Ausgang remappen und Verdrahtung ändern → Dank Pin-remapping sehr schnell und einfach

2.3 Empfänger Version 3

-Wie V2 aber mit separatem Pmos zum schalten der Versorgungsspannung des Transceivers

-zusätzlicher Pmos als Verpolschutz → bisher großes Fehlerpotenzial

-kleinerer Spannungsregler (150mA, SOT23 statt 1A, SOT223)

Stand: in Konzeption

3 Versionsverlauf des Steuermodul

3.1 Steuermodul Version 1

-2 5V Schaltwandler (1 optional)

-Verpolschutzdiode 5A

-selbstrückstellende PTC Sicherungen

-PIC16F1823

-RS485 Transceiver

Stand: gefertigt

Vorteile: günstig

Nachteile: viele Durchkontaktierungen, falsche Landeflächen für die Sicherungen, MCU zu langsam und zu wenig Speicher

3.2 Steuermodul Version 2

-Prinzip wie V1

-PIC18F25K40, doppelte Geschwindigkeit, mehr Speicher, Pinremapping, CRC in Hardware, 8x8 Bit Hardwaremultiplizierer beschleunigt Arrayzugriffe

-4 LEDs zur Statusanzeige

-weniger Durchkontaktierungen dank PPS und größerem Chipgehäuse

-Platine lackiert mit Conformal Coating (Wasserschutz, Anlaufschutz)

Stand: gefertigt

Vorteile: deutlich rechenstärker, mehr Programmierfreiraum, LEDs erleichtern Debugging

Nachteile: Pinremapping nicht für alle Pins möglich wie bei PIC16F1619

(Provisorische) Reparatur: abändern der Verdrahtung auf andere Pins