Bauhaus-Universität Weimar
Fakultät Medien
Studiengang Medieninformatik

Bluetooth-based Mesh Networking for Smoke Detectors

Bachelorarbeit

Matti Wiegmann
geb. am: 24.01.1990 in Suhl

Matrikelnummer 112174

1. Gutachter: Junior-Prof. Dr. Florian Echtler
2. Gutachter: Prof. Dr. Unknown Yet

Datum der Abgabe: 31. Februar 2022

Erklärung

Hiermit versichere ich, dass ich diese Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Weimar, 31. Februar 2022

.....
Matti Wiegmann

Zusammenfassung

A short summary.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Kurze Beschreibung der Motivation:

- Was es werden soll
- Was es schon gibt (properitär): zu teuer, zu mächtig, warum wifi als basis von IoT schlecht ist (wenn ich dazu quellen finde die das belegen)
- warum man diese schwächen mit BLE lösen kann
- kurz was genau gemacht und untersucht wurde und was dabei rum kam
- das werde ich wohl als letztes schreiben?

Kapitel 2

Related Work

Das Gebiet der automatisierten Erkennung von Feuer ist mit Bezug zur Informatik ausführlich erforscht worden. Dabei stechen vor allem drei Teilbereiche hervor:

- Studien zu verschiedenen Sonortypen und denen kombinationen. Dazu zählen Versuche mit Fuzzy Logics, neuronalen Netzen und Fusion Algorithms um vor allem Fehlalarme, spezielle unter widrigen Umständen, zu reduzieren.
- Studien zur Erkennung von Waldbränden, unter anderem Bild- und Videoanalysen von Flugzeug- und Satellitenaufnahmen sowie GSM-Sensornetze.
- Erkennung von Feuern in Gebäuden und bewohnten Gebieten.

Eine gute Übersicht zu diesen Themen findet man in **Automatic Fire Detection: A Survey from Wireless Seonsor Network Perspective** von Majid Bahrepour, Nirvana Meratnia und Paul Havinga[6]. Im Folgenden sollen einige für diese Arbeit besonders relevante Arbeiten kurz vorgestellt werden.

In **Building Fire Emergency Detection and Response Using Wireless Sensor Networks** [12] beschrieben Zeng et al. drei Routing-Protokolle für die Kommunikation zwischen vernetzen Funkrauchmeldern.
Das Fundament dabei bildet das von den Autoren entwickelte Verfahren "Real-time and Robust Routing in Fire (RTRR)". Dieses beschreibt das Routing von fest verbauten Sensoren (nodes) zu vorher bekannten Emfangsstationen (sinks). Jede Node kennt dabei einen von den vier Zuständen: safe (kein Feuer), lowsafe (angrenzende Node schlägt Alarm), infire (schlägt Alarm) and unsafe (nicht Funktionstüchtig).
Die anderen beiden Protokolle erweitern dieses Prinzip um mobile Nodes und

Sinks (an den Rettungskräften) für den Fall dass Melder während des Einsatzes ausfallen. Anschließend wird ein Konzept beschrieben wie der bestmögliche Rettungsweg anhand des Routingprotokolles berechnet werden kann.

Ismail, Husny und Abdullah präsentieren in **Smoke Detection Alert System via Mobile Application** [7] ein Konzept für einen Rauchmelder, der den Nutzer durch den Short Messaging System (SMS) informiert, wenn der Melder Alarm schlägt.
Dabei haben die Autoren einen Rauchsensor sowie ein Bluetooth Classic Funkmodul in einen Arduino verbaut. Wenn der Sensor aktiv wird, verbindet sich das Bluetooth-Modul mit einem nahegelgenen "Sender"-Smartphone. Dieses sendet wiederum eine speielle SMS-Nachricht an das Empfänger-Smartphone. Dort wird die Nachricht von einer App abgefangen, die wiederum den Alarm am Empfänger auslöst und Möglichkeiten bietet, den Alarm abzuschalten.

Ein auf Routing basierendes, speziell für mehrgeschossige Gebäude entworfenes Design, wurde von Lei Zhang und Gaofeng Wang in **Design and Implementation of Automatic Fire Alarm System based on Wireless Sensor Networks** [13] vorgestellt.
Das Netzwerk setzt auf ein kabelgebundenes Backbone-Netz aus zentralen und lokalen Verteilern (Local Center und Surveillance Center). Die Weiterleitung zwischen Detektoren und lokalen Verteilern übernehmen dedizierte Repeater. Die Sensor-Nodes verbinden sich mit einem Repeater in Reichweite, beziehen eine dynamisch generierte Adresse und werden vom Repeater am Netzwerk angemeldet.
Die Kommunikation setzt auf den RF CC1100 Chipset von Texas Instruments. Das vorgestellte Konzept erlaubt eine Weiterleitung von Alarmsignalen in maximal drei Hops. Außerdem kann erkannt werden, welcher Sensor ausgelöst hat und wo der Repeater lokalisiert ist.

In **An Intelligent Fire Detection and Mitigation System Safe from Fire (SFF)** [9] wird ein experimentelles Brandschutzsystem vorgestellt. Es besteht aus einem Arduino als zentraler Verwaltungseinheit an den je zwei Rauch-, Gas-, und Temperatursensoren angeschlossen sind. Die Sensoren werden durch einen Fusion-Algorithmus ausgewertet um Fehlalarme zu reduzieren. Die Position des Feuers im abgedeckten Bereich wird durch eine Fuzzy Logic approximiert und die Sprinkler über einen angeschlossenen Motor zum Brandherd hin ausgerichtet. Wird ein Feuer erkannt, benachtichtigt der Arduino automatisch über ein angeschlossenes GSM-Modul die Feuerwehr sowie den Zuständigen Gebäudeservice.

Kapitel 3

Vorbetrachtung

Wichtigste Frage von allen: Ist das zuviel Theorie? Kann ich das meiste davon einfach vorraussetzen oder als irrelevant deklarieren und dsa ganze Kapitel auf die Hälfte kürzen?

Um eine Implementierung planen zu können, müssen zuerst die technischen Rahmenbedingungen und Vorschriften analysiert werden. In diesem Kapitel sollen die wichtigen Bestandteile von Rauchmeldern sowohl allgemein als auch an einem passenden Testgerät herausgestellt werden. Außerdem wird beschrieben welche Möglichkeiten der Vernetzung bestehen und worauf bei Hard- und Software geachtet werden muss.

3.1 Aufbau und Funktionsweise von Rauchmeldern

Wie eingangs 1 erwähnt kann man einen Rauchmelder als System aus Sensoren und Aktuatoren ansehen, die über einen Integrated Circuit (IC) gesteuert werden. Der Aktuator ist der Alarmgeber, üblicherweise ferroelektrisch (Piezo-Summer). Die Sensoren können grob in zwei Kategorien unterteilt werden: Rauchsensoren und ergänzende Sensoren.

Bei Rauchsensoren git es zwei übliche Typen: Ionisationsrauchsensoren und Photoelektronische (Optische) Rauchsensoren [10]. Erstere ionisieren mit Hilfe radioaktivem Materials die Luft zwischen zwei Metallplatten. Eindringender Rauch unterbricht dabei den fluss der Ionen und löst so den Alarm aus. Photoelektronische Rauchsensoren bestehen aus einer Rauchkammer, einer (Infrarot-)LED und einem Lichtsensor. Je nach Bauart des Sensors unterbricht der Rauch dabei entweder den Lichtstrahl oder löst den Lichtsensor durch ver-

änderte Reflexionseingenschaften erst aus. Dieses Prinzip funktioniert besser, je größer die Partikel im Rauch sind. Die Größe ist vor allem abhängig von der Temperatur des Feuers. So sind bei Schwelbränden die Rauchpartikel größer als bei offenem Feuer. Da diese bei Wohnungs- und Gebäudebränden sehr häufig sind und Ionisationsrauchsensoren aufgrund ihrer unsauberen Technologie in Europa nicht vertrieben werden dürfen [2], sind Photoelektronische Rauchsensoren der de facto Standard.
Photoelektronische Rauchsensoren sind allerdings anfällig für Falschalarm, ausgelöst durch Wasserdampf, u.a. in Badezimmern, oder Rauch der beim Kochen entsteht. Um diese schwächen auszugleichen werden in hochwertigen Rauchmeldern zusätzliche Sensoren Verbaut. Dazu zählen Temperatursensoren, CO-Sensoren und Feuersensoren [10] (siehe Kapitel 2).

Für Rauchmelder gelten in Deutschland zwei zentrale Normen. Die DIN 14767 [1] regelt Einbau, Betrieb und Test und wird in Kapitel 15 näher beschrieben. Die für Aufbau und Funktionsweise geltende Norm ist die DIN EN 14604 [2]. Die Norm ist für diese Arbeit besonders relevant, da sie das Verhalten der Rauchmelder und das Vorhandensein sowie Verhalten von LED und Testknopf festlegt. Das bedeutet, das sowohl die folgende Analyse eines Testgerätes als auch das in Kapitel 13 vorgestellte Konzept so oder so ähnlich für alle zertifizierten Modelle gilt.

Als Testrauchmelder wurde ein handelsüblicher, nicht vernetzter optischer Rauchmelder Flamingo FA23 von Smartwares ohne zusätzliche Sensoren ausgewählt.

In Abbildung 3.1 ist der Rauchmelder abgebilded. Er besteht aus sechs wichtigen Komponenten.

- Dem optische Rauchsensor (Rauchkammer) mit Infrarot-LED und Lichtsensor, einer 9-Volt Blockbatterie, dem Testschalter, einer roten LED als Warn- und Funktionsanzeige und zusätzlich:
- Dem Signalgeber (Horn). Bei dem Tetsgerät wurde ein 3-Pin Self-Drive Piezo-Summer verbaut, der mit 9V Wechselspannung versorgt wird. Einfache Piezo-Summer werden üblicherweise über zwei Pins mit einer Wechselspannung zwischen 10 und 100 Volt in Schwingung versetzt und erzeugen so einen hochfrequenten Ton. Der dritte Pin liefert ein Feedback-Signal für den Oszillator. Das hat zur Folge, dass der Schallgeber mit Eigenfrequenz schwingt und so eine höhere Lautstärke erreicht [4].
- Dem IC KD-5810 von Kingstar Technology Ltd. Er befindet sich auf der Unterseite der Platine und ist in Abbildung 3.1 nicht erkennbar. Dieser

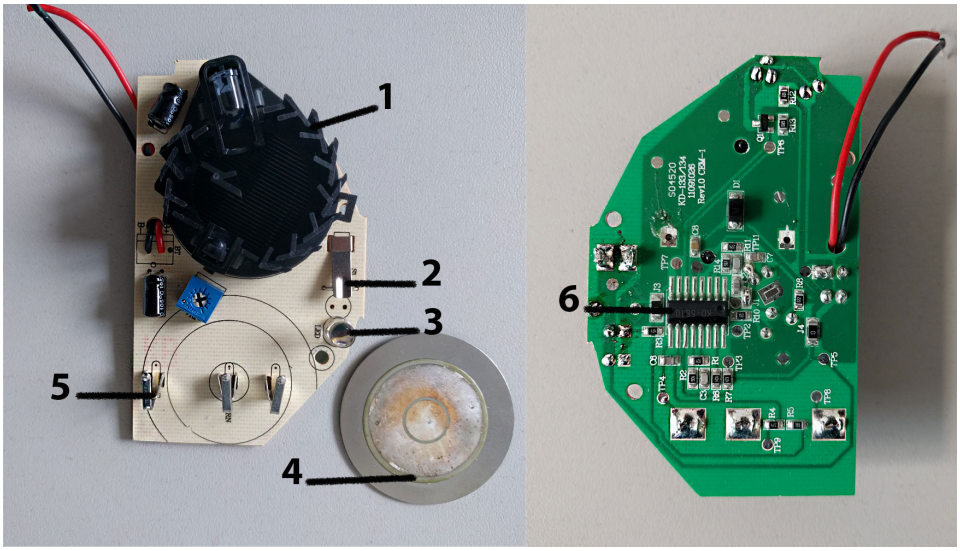

Abbildung 3.1: Innenleben des Testrauchmelders. (links) Oberseite: (1) Rauchsensor (geöffnet), (2) Testschalter, (3) Warnanzeige (LED), (4) Piezo-Element, (5) Signalgeber Pins, (rechts) Unterseite mit Schaltkreis, (6) IC ohne Abdeckung

IC ist ein Nachbau des verbreiteten MC145010 [3] von Freescale Semiconductor mit veränderter Pin-Belegung. Der IC ist von einer Matallplatte verdeckt, so dass die Pins nicht direkt zugägnglich sind, ohne den Rauchmelder zu beschädigen.

Der Rauchmelder hat die folgende vier Verhaltensmuster [3]:

- **Idle:** Der Signalgeber ist aus. Der Rauchmelder führt alle 38.9 bis 47.1 Sekunden eine Prüfung der Elektronik durch (Low-Supply und Chamber-Sensitivity). Die LED blinkt alle 38.9 bis 47.1 Sekunden.
- **Test:** Wenn der Testschalter gedrückt ist. Der Signalgeber ist durchgehend an und die LED blinkt alle 0.6 bis 0.74 Sekunden.
- **ON:** Wenn der Testschalter nicht gedrückt ist und Rauch in der RAuchkammer ist. Der Signalgeber ist durchgehend an und die LED blinkt alle 0.6 bis 0.74 Sekunden
- **Fehler:** Wird bei der Prüfung der Elektronik ein Fehler festgestellt Pipst Rauchmelder alle 38.9 bis 47.1 Sekunden. Die LED blinkt alle 38.9 bis 47.1 Sekunden. Bei Low-Supply pipst der Melder zeitgleich mit dem blinken der LED, bei Chamber-Sensitivity pipst er in der Mitte des Blinkintervalls.

Die tatsächliche Intervalldauer ist abhängig von der Taktung des Oszillators, bestimmt durch die größe der verbauten Widerstände und Kondensatoren. Für das Testmodell wurden 44 Sekunden fur das lange Intervall und 0.7 Sekunden für das kurze Intervall gemessen. Laut Vorschrift müssen diese kürzer als eine Minute beziehungsweise eine Sekunde sein.
Die Zustände ON und Test sind vom Verhalten her identisch. Sie können dadurch Unterschieden werden dass entweder der Testschalter direkt oder der Pin16 des ICs (TEST) abgetastet wird.
Ohne Rauch kann der Rauchmelder über einen externen Microprozessor (MPU) ausgelöst werden indem entweder der Testschalter kurzgeschlossen wird oder eine Spannung von mindestens 8.5 Volt an Pin16 des ICs angelegt wird. Eine dritte Möglichkeit besteht darin, das Horn durch externen MPU direkt anzusprechen.
Alternativ dazu kann der Pin7 (I/O) des ICs dazu benutzt werden. Dieser Pin ist konfigurationsabhängig dafür vorgesehen, beim auslösen des Rauchsensors angeschlossene Geräte zu Informieren oder von angeschlossenen Geräten ausgelöst zu werden. Im Gegensatz zu Pin16 hat Pin7 ein LOW von 1.5 Volt und ein HIGH von 3.2 Volt und kann von einem 3.3 Volt Microprozessor ohne Spannungstransformation angesteuert bzw. ausgelesen werden, solange die Konfiguration bekannt ist.

Ohne den IC direkt abzutasten kann der Zusatnd des Rauchmelders auch anhand des Blinkmusters ausgelesen werden. Dabei kann allerdings nicht festgestellt werden ob eine Störung der Elektronik vorliegt, da das Blinkmuster mit Störung identisch ist zu dem im Idle-Modus.

3.2 Wireless Sensor Networks für Sicherheitskritische Anwendungen

Im folgenden Abschnitt soll erläutert werden, welche Technologien für Wireless Sensor Networks (WSN) in Frage kommen und wieso sich diese Arbeit mit Bluetooth LE (BLE) Broadcast-Mesh-Netzwerken befasst. Außerdem soll die benutzte Hardware sowie für die Implementation wichtige Konzepte des BLE-Stacks beschrieben werden.

3.2.1 Technologische Grundlagen

Die Idee hinter dem Internet der Dinge (IoT) ist jedes Gerät eindeutig Addressierbar zu machen und zentral zu steuern. Die Schlüsseltechnologien dafür sind die Internettechnologien TCP/IP und Wifi. Dabei bekommt jedes Gerät (mit

Sensoren/Aktuatoren) eine Adresse aus dem IPv6-Adressraum. Die Kommunikation innerhalb des Netzwerkes und mit der Außenwelt wird durch (dedizierte) Router geregelt, die in Empfangsreichweite der IoT-Geräte sein müssen.

Wifi ist eine mächtige Technologie für das Internet der Dinge, vor allem da sie den vollen Internet-Protokollstack implementiert, einen hohe Reichweite und Datenrate hat. Das bringt jedeoch das Problem mit sich, dass der Energieverbraucht relativ hoch ist. Low-Power Geräte wie Sensor-Nodes sollten mit einer gewöhnlichen 3 Volt Knopfzelle (etwa 50-300 mAh) viele Monate bis Jahre betrieben werden können, was mit herkömmlichen Wifi nicht möglich ist.

cite: http://www.rs-online.com/designspark/electronics/knowledge-item/eleven-internet-of-things-iot-protocols-you-need-to-know-about ???

Um dieses Problem zu lösen wurde eine Vielzahl neuer Technologien und Standards speziell für den Low-Power Bereich entwickelt. Die wichtigesten davon sind:

- **6LoWPAN** (IPv6 Low-Power Wireless Personal Area Network) und Thread sind Netzwerk-Protokolle. Sie basieren auf IPv6 und sind für Low-Power Geräte und Mesh-Netzwerke entwickelt worden. Der Standard für 6LoWPAN ist der RFC6282, für Thread IEEE802.15.4 und 6LowPAN.

- **Zigbee** basiert auf IEEE802.15.4. Es sendet im 2,4 GHz Band und unterstützt Datenraten von bis zu 250 Kbit/s auf einer Reichweite von bis zu 100 Metern.

- **Z-Wave** ermöglicht Datenraten von bis zu 100 Kbit/s auf bis zu 30 Meter Reichweite. Gesendet wird im Gegensatz zu Zigbee und BLE im 900 MHz Band, wodurch es weniger anfällig gegenüber Interferenzen ist.

- **Bluetooth Low Energy** (auch Bluetooth Smart / BLE) ist eine Erweiterung der Bluetooth Core Specification ab Version 4.0. Gesendet wird im 2.4 GHz Band mit Datenraten von bis zu 1 Mbit/s und einer Reichweite von theoretisch über 100 Metern. Seit Core Specification 4.2 implementiert BLE auch 6LoWPAN, so dass Geräte über das Internet Protocol Support Profile mit IP-basierten Netzwerken kommunizieren können.

Zigbee und Z-Wave implementieren Mesh-Funktionalitäten bereits, für BLE ist nativer Support in Entwicklung. Bluetooth hat allerdings den Vorteil, dass es eiine wesentlich weitere Verbreitung im privaten Sektor hat und von vielen Smartphones und ähnlichen Geräten uterstützt wird. (Quelle nötig? genaue

Zahlen/Prognosen?).
Da in industriellen und öffentlichen Gebäuden meistens verkablete Brandmeldeanlagen vorgeschrieben werden [2] ist der private Sektor für diese Arbeit am interessantesten. Daher bietet sich Bluetooth Low Energy als technologische Basis an.

Die meisten der erwähnten Technologien für das Internet der Dinge setzen auf IPv6 und zählen somit zu den Routed-Networks. Das heißt, dass jedes Gerät eine Adresse hat. Die Daten, die über das Netzwerk versendet werden, werden mit den Adressen des Absenders und des Empfängers versehen. Die Router wissen, im Gegensatz zu den Endgeräten, wie diese Pakete über das Netzwerk versendet werden müssen und sind deswegen nötig um die Kommunikation zwischen Endgeräten zu ermöglichen.
Dieses Prinzip ist dann essenziell, wenn Absender und Zeil genau identifiziert werden müssen. Ist das nicht der Fall, bleiben einige Nachteile:

- Es sind dedizierte Geräte (Router) notwendig, damit das Netzwerk funktioniert.
- Jedes Gerät im Netzwerk muss eingerichtet werden.
- Fällt ein Gerät aus, müssen im besten Fall die Routen neu bestimmt werden. Im schlechtesten Fall können Teile des Netzwerkes nicht mehr kommunizieren.

Um diese Probleme zu umgehen, hat sich die Wissenschaft mit alternativen Lösungen beschäftigt (siehe 2).

Sicherheitskritische Anwendungen, wie zum Beispiel Rauchmeldernetzwerke, sind nicht darauf angewiesen, dass jedes Gerät eindeutig Adressierbar ist. Die Aufgabe des Netzwerkes ist, das jeder Rauchmelder im Netzwerk Alarm schlägt, wenn nur ein einziger Melder Rauch erkennt.
Strasser et al. haben für diese Art von Netzwerken drei "fundamental challenges"[11] herausgestellt: zuverlässige Datenübertragung (reliable data delivery), geringe Latenz (low latency) und geringer Energieverbrauch (low energy consumption).
Um diesen Anforderungen gerecht zu werden schlagen Strasser et al. vor, Flooding anstatt Routing zu benutzen. Beim Flooding wird die Nachricht von der Sender-Node an alle Nodes in Empfangsreichweite gesendet. Diese leiten das Paket wiederum an alle Nodes in Reichweite weiter, bis alle Nodes im Netzwerk die Information erhalten haben. Die Autoren weisen allerdings darauf hin, dass Broadcast-Flooding dem Anspruch der Energiesparsamkeit nicht gerecht wird. Beim Broadcast-Flooding, im Gegensatz zum Unicast-Flooding, werden

die Daten unselektiert an alle Nodes in Reichweite weiter gesendet. Damit das Netzwerk allerdings Funktionsfähig bleibt, müssen in regelmäßigen Abständen Status-Nachrichten gesendet werden. Passiert das nicht, wissen die Nodes und eventuelle Supervisor nicht ob das Netzwerk noch vollständig besteht.
Um energiesparendes Broadcast-Flooding zu ermöglichen, haben Levis et al. 2011 den Trickle-Algorithmus (RFC6206)[8] entwickelt. Trickle verfügt über ein variables Rebroadcasting-Intervall. Dieses Intervall wird länger, je länger der Abstand zum letzten Update wird. Dadurch kann der Abstand zwischen zwei Broadcasts auf bis zu mehrere Stunden ansteigen. Empfängt eine Node einen inkonsistenten Wert, überprüft sie anhand eines Alters-Flags ob es sich um eine Update handelt. Ist das der Fall, wird das Broadcasting-Intervall wieder auf den Minimalwert zurückgesetzt und so sichergestellt, dass updates so schnell wie möglich über das Netzwerk propagiert werden.
Trickle genauer erklären? Oder reicht der Verweis auf den RFC?
Für diese Arbeit soll also ein auf Trickle basierendes Broadcast-Flooding-Mesh-Network für Rauchmelder mit Bluetooth Kommunikationstechnologie entwickelt und seine Einsatzfähigkeit verifiziert werden.

3.2.2 Bluetooth Low Energy

Da Bluetooth Low Energy ein offener Standard, beschrieben in der Bluetooth Core Specification [5], ist, gibt es eine Vielzahl an Hardwareherstellern. Für dieses Projekt wurde die auf dem ARM Cortex M0 basierende nRF51822 MCU von Nordic Semiconductor ausgewählt. Aus der Menge der möglichen Geräte wurde ein nRF52822 Breakout-Board (siehe Abbildung 3.2 ausgewählt. Es besteht aus dem SoC, einer integrierten Antenne. Die Pins des SoC sind als Pinheader ausgeführt.
Das Breakout-Board ist 2.5 cm breit und 3 cm lang und kostet etwa 6 Euro. Programmiert werden kann der Chip in C99, kompiliert mit gcc oder der KEIL IDE von ARM. Um den Chip zu flashen kann ein Buspirate oder ein JTAG-Debugger von J-Link verwendet werden. (Sinvoll das ganze Prinzip/Verkabelung etc. genauer zu erklären?).

Da der Bluetooth Stack sehr komplex ist, sollen die für diese Arbeit wichtigsten Konzepte kurz erläutert werden.
Ein BLE Gerät ist Grundsätzlich entweder ein Central oder ein Peripheral. Peripherals sind verteilte, low-energy Geräte die an den jeweiligen Sensoren oder Aktuatoren angeschlossen sind. Centrals sind die mächtigeren, zentralen Geräte. Sie sollen die Datan aus mehreren Peripherals auslesen. Die Kommunikation zwischen BLE(-fähigen) Geräten wird durch zwei sogennante Profile geführt.

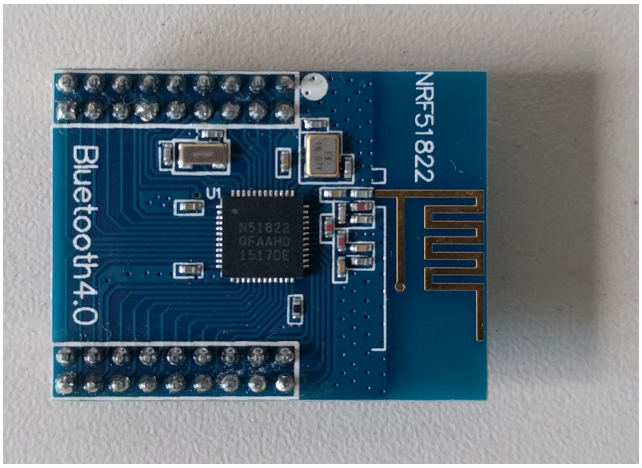

Abbildung 3.2: nRF51822 Breakout-Board. Unterseite.

Das Generic Access Profile (GAP) managed Advertising und Verbindungsfähigkeit. Das Advertising ist ein Datenpaket, das regelmäßig und unaufgefordert vom Peripheral gesendet wird und von allen anderen BLE Geräten empfangen werden kann. GAP muss implementiert und aktiv sein, damit die Node von anderen Geräten gesehen werden kann. Laut Spezifikation kommunizieren BLE Geräte über 40 Channel mit je 2 MHz Abstand im 2,4 GHz ISM Band. Um Interferenzen mit Wifi zu vermeiden werden für GAP-Advertisments nur die Channel 37, 38 und 39 genutzt, auf denen Wifi nicht sendet.
cite: https://devzone.nordicsemi.com/question/30138/multiple-advertisements-using-the-timeslot-api/??

Das Advertisment ist die Broadcast-Methode von BLE. In der Core Specification ist allerdings nicht vorgesehen, dass Peripherals über Advertisments kommunizieren. Außerdem ist es im Standard nicht vorgesehen, dass während einer Verbindung gebroadcastet werden kann. Für diese Funktionen kann aber das Softdevice der Nordic-SDK benutzt werden.
Advertising Byte-Struktur erklären?? Mit Grafik?
Die BLE-Funktionen sind für Geräte von Nordic Semiconductor in einer Art Betriebssystem-API implementiert, die Softdevice genannt wird. Dieses Softdevice ist vorkompiliert und abhängig von der benutzten Version der Nordic SDK und dem Typ des BLE-Gerätes. Zusätzlich zu den festgelegten Funktionen bietet dieses SDK ab Version 8.0 die sogenannte Timeslot-API.
Diese API ermöglicht genau das, dass mehrere Advertising-"Threads"nebeneinander stattfinden und von den anderen Peripherals mit der gleichen Softdevice Verion auch empfangen werden können. Das GAP-Advertisment, dass von Centrals

gescant und zum Verbindungsaufbau benötigt wird, wird davon nciht beinflusst.
Timeslot funktionsweise detaillierter erklären? cite? https://devzone.nordicsemi.com/tutorials/16/

Über GAP können die Peripheral und Central miteinander verbunden werden, was das Advertising in der Regel für die Dauer der Verbindung stoppt. Das Generic Attribute Profile (GATT) beschreibt die Kommunikation zwischen verbundenen Peripheral und Central. Das Peripheral implementiert dafür Profiles, Services und Characteristics. Das sind hierarchisch verschachtelte Objecte, wobei jedes Profile mehrere Services und jeder Service mehrere Characteristics beinhalten kann.
Das Central kann Lese- und Schreibanfrangen an eine Characteristic des verbundenen Peripherals schicken. Das Peripheral kann ein verbundenes Central informieren, wenn sich der Wert einer Characteristic ändert. GATT benutzt für die Datenübertragung die verbliebenen 37 Channel im ISM-Band und ermöglicht so Übertragungsraten von bis zu 1 MHz/s.

GATT ist für das Flooding nicht sinnvoll. Selbst wenn die Nodes sich schnell genug Verbinden und Trennen würden, wäre der Energiebedraf dafür trotzdem zu hoch um lange Laufzeiten beizubehalten. Es kann jedoch als Schnittstelle für ein Central-Device dienen, dass den Status des Netzes überwacht oder zu Testzwecken manipuliert.

Kapitel 4

Implementierung

4.1 Verbinden von Rauchmelder und Microprozessor

siehe Vorbetrachtung -> welche Anschlussmöglichkeit IN/OUT sind am besten geeignet für diesen zweck. Was würde man machen wenn man den Rauchmelder mit SoC direkt bauen würde. Auf was muss man bei der Software achten (LED blinken usw). Verweis auf Future Work bzgl. Piezo direkt ansprechen

4.2 Software Microprozessor

4.2.1 Konzept

Welche funktionen muss die software haben / schnittstellen nach außen, abtasten sensor / an/aus verhalten

4.2.2 Implementierung

Was wird aus dem Framework genommen, wo wird welche Softwarekomponenten hinzugefügt, GPIO, Timer zum abtasten, bei welchen Events werden Werte im Netzwerk geupdated.

OpenMesh

Was wird wann ins Adv-Packet geschrieben. Was ist mit den Services (Die 2 Standard-Services vom FW wurden aktiviert um das netzwerk zu testen –> hinweisen auf Sicherheitsrisiko / Future Work)

4.3 Software Android

hier nur eine Section? kurz erklären wie das Mesh vom Smartphone aus überwacht wird. Ich bin mir noch nicht sicher wie umfangreich ich das machen sollte ... einfach nur Advertising-Werte auslesen und anzeigen? Notifications anzeigen wenn sich Werte ändern? Batterieüberwachung mit einbauen? Wie wichtig ist das für den Erfolg der Arbeit? Visualisierung notwendig?

Kapitel 5

Evaluation

5.1 Test der Funktionsweise

Was habe ich gemacht um sicherzustellen dass alles funktioniert

5.2 Praxistauglichkeit????

was sind vorschriften über die Verteilung von SDs in Wohnräumen, kurzer Testaufbau der dem etwa entspricht Test der Reichweite, in wie weit spielen Wände, Decken, Interferenzen eine Rolle. Das könnte wohl ein ziemlich dünnes Kapitel werden. Aber ich denke es ist wichtig das Netz einmal unter realistischen bedingungen systematisch zu testen und zu schauen ob irgendwas auffällt.

http://www.eielectronics.de/downloads/Ei$_E lectronics_G rundlagenwissen_R auchwarnmelder.p$
$3 meterbreiteansonstenmax 60 m^2 DIN 14676 -> abstandzwischenrauchmeldern$:
$1 proraum < 60 m^2, L formzhltals 2 Rume$

DIN 14675 -> rauchmeldeanlagen nach anordnung §46 der Thüringer Bauordnung folgender Absatz 4 pflicht in schlaf und kindezimmern sowie fluren die als zugang zu rettungswegen dienen

Kapitel 6

Ausblick

Piezo direkt ansprechen -> Verschiedene Tonhöhen, Tonfolgen etc. um zu zeigen wie weit der Ursprung weg ist / was das problem ist

Einbinden in bestehende IoT Systeme / Zugriff über Interwebz ??

6.1 Sicherheit vor Angriffen und Fälschung

neuer Sensor änmeldenän gateway –> sensor-ID muss bestätigt werden - mesh nimmt nur updates von

6.2 Energieverbrauch

Erweitern der Software um Batterie-Messungen. Dann Testen verschiedener Software/Hardware: nRF mit eigener Batterie und Software -> Nordic Beacon Example, Mesh Template (quasi ohne alles), mein SD-Mesh. Danach eventuell wenn die Zeit reicht nRF an 9V des SD anschließen und einfach mal schauen was passiert. Vergleich von openMesh mit implementierung vom neuen Standard + properitäre lösungen

6.3 Überwachung der Funktionstauglichkeit

Literaturverzeichnis

[1] DIN 14676:2012-09, Rauchwarnmelder für Wohnhäuser, Wohnungen und Räume mit wohnungsähnlicher Nutzung - Einbau, Betrieb und Instandhaltung. 3.1

[2] DIN EN 14604:2009-02, Rauchwarnmelder; Deutsche Fassung (EN 14604:2005). 3.1, 3.2.1

[3] Photoelectric Smoke Detector IC with I/O - MC145010. `http://www.nxp.com/files/analog/doc/data_sheet/MC145010.pdf`. letzter Zugriff 23.07.2016. 3.1

[4] What's the third wire on a piezo buzzer? `http://electronics.stackexchange.com/questions/18212/whats-the-third-wire-on-a-piezo-buzzer`. letzter Zugriff 23.07.2016. 3.1

[5] Bluetooth Core Specification 4.2. `https://www.bluetooth.com/specifications/adopted-specifications`, Dezember 2014. 3.2.2

[6] Majid Bahrepour, Nirvana Meratnia, and Paul Havinga. Automatic Fire Detection: A Survey from Wireless Sensor Network Perspective. `http://doc.utwente.nl/65223/1/Automatic_Fire_Detection.pdf`, 2008. letzter Zugriff 13.07.2016. 2

[7] Wan Hazimah Wan Ismail, Herny Ramadhani Mohd Husny, and Norhaiza Ya Abdullah. Smoke Detection Alert System via Mobile Application. In *Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication*, Danang, Vietnam, Januar 2016. 2

[8] P. Levis, T. Clausen, J. Hui, O. Gnawali, and J. Ko. RFC 6206, Trickle-Algorithm. `https://tools.ietf.org/html/rfc6206`, März 2011. 3.2.1

[9] Iftekharul Mobin, Abid Ar-Rafi, Neamul Islam, and Rifat Hasan. An intelligent fire detection and mitigation system safe from fire (sff). *International Journal of Computer Applications*, 133(6), Januar 2016. 2

[10] Tyco Fire Safety. Consultant's Guide for Designing Fire Detection Alarm Systems. `http://tfppemea.com/en/_layouts/fsassets/Docs/Detection_FireClass/UKFireClassConsultantsGuide(LR).pdf`. letzter Zugriff 23.07.2016. 3.1

[11] Mario Strasser, Andreas Meier, Koen Langendoen, and Philipp Blum. Dwarf: Delay-Aware Robust Forwarding for Energy-Constrained Wireless Sensor Networks. In *Proceedings of the 3rd IEEE international conference on Distributed computing in sensor systems*, Heidelberg, Deutschland, Juni 2007. 3.2.1

[12] Y. Zeng, S. Murphy, L. Sitanayah, T. Tabirca, T. Truong, K. Brown, and C. Sreenan. Building Fire Emergency Detection and Response using Wireless Sensor Networks. In *Ninth IT and T Conference, Dublin Institute of Technology*, Dublin, Irland, Oktober 2009. 2

[13] Lei Zhang and Gaofeng Wang. Design and Implementation of Automatic Fire Alarm System based on Wireless Sensor Networks. In *Proceedings of the 2009 International Symposium on Information Processing*, Huangshan, China, August 2009. 2