Freie Universität Berlin

Fachbereich Mathematik und Informatik

Institut für Informatik

Prof. Dr.-Ing. habil. Jochen H. Schiller

Diplomarbeit

Abstandsmessung durch Laufzeitmessung in drahtlosen Sensornetzwerken

Betreuer: Dipl.-Ing. (FH) Stephan Adler

Berlin, 15. Dezember 2010

Verfasser: Stefan Pfeiffer

Erklärung

Ich versichere hiermit, dass die vorliegende Arbeit selbständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe. Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind als solche kenntlich gemacht. Die Zeichnungen oder Abbildungen sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen.

Berlin, 15. Dezember 2010

Stefan Pfeiffer

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Die Frage nach der Position eines Sensorknotens in einem drahtlosen Sensornetzwerk (w*ireless sensor network, WSN*) ist eine häufige Fragestellung in der aktuellen Forschung. Lokalisierungsverfahren können die Position eines Sensorknotens meist auf Grundlage verschiedener Messmethoden zur Abstandsapproximation zwischen den Sensorknoten berechnen. Mit der Kenntnis der Position erschließen sich neue Einsatzbereiche für *WSNs*. So wurde im Projekt *FeuerWhere* an der Freien Universität Berlin eine prototypische Lösung erforscht, die durch Kombination von Personenlokalisierung und Vitaldatenerfassung die Sicherheit von Einsatzkräften der Feuerwehr verbessern soll [FEU10]. In vielen Anwendungen ist ein geographischer Kontext für die Sensordaten essentiell. Im Projekt *AVSExtrem* an der Freien Universität Berlin soll ein *WSN* zur Überwachung von Baustellen eingesetzt werden. Ziel dieser Überwachung ist die Erkennung von sicherheitsrelevanten Ereignissen und deren geographische Position. Die Umsetzung erfolgt über eine im Sensornetzwerk verteilt arbeitende Mustererkennung, für diese ist die Kenntnis der Position eines jeden Sensorknotens Grundvoraussetzung. Jenseits der *GPS* Technologie, welche nur außerhalb von Gebäuden eingesetzt werden kann existieren keine Lösungen für das Lokalisierungsproblem, die in einem *WSN* angewendet werden können.

Setzt man den Fokus auf Hardware für *WSNs*, so schränken sich die Möglichkeiten, die zur Lösung des Lokalisierungsproblem zur Verfügung stehen stark ein. Es existieren Schranken für Energie- und Platzbedarf, welche den Einsatz von spezialisierter Hardware verbieten. In vorherigen Arbeiten wurde deshalb an Lösungen geforscht, die durch Abstandmessungen mittels der bereits in der Sensorknotenhardware enthaltenen Transceiver eine Lokalisierung ermöglichen. Die Erkenntnis, dass mit der Leistungsdichtemessung[1] keine präzisen Abstände zwischen Sensorknoten bestimmbar sind, verschob das Interesse der Forschungsgemeinde in die Richtung der

1 zur Bestimmung des *RSSI* Wertes. (Vgl. Kapitel 1.1.1.1)

Lokalisierungsalgorithmen, wobei die benötigten Abstandbestimmungen zumeist simuliert wurden. Es fehlten alternative Abstandsmessmethoden, was unter anderem durch die geringe Taktrate der bisherigen Sensorknotenprozessoren begründet wurde. Außerdem bieten die verfügbaren kommerziellen Lösungen von denen Eine in beiden oben genannten Projekten untersucht worden ist, bislang nicht die Präzision die für eine Innenraumlokalisierung nötig ist.

Die aktuelle Generation von Sensorknotenhardware wird mit Taktfrequenzen betrieben, welche zeitbasierte Abstandsmessmethoden umsetzbar erscheinen lassen. An diesem Punkt setzt die Aufgabenstellung für diese Diplomarbeit an. Es soll untersucht werden welche Abweichungen eine auf Laufzeitmessung basierende Abstandsmessmethode aufweist, die auf den Einsatz der vorhandenen Sensorknotenhardware beschränkt ist. Es wird eine Lösung vorgestellt, diese basiert auf dem *CC1100* Transceiver der Firma *Texas Instruments* und dem *LPC2387* Mikrocontroller der Firma *NXP Semiconductors*.

Gegenstand aktueller Forschung ist die Kombination verschiedener Abstandsmessmethoden zur Verbesserung der Abstandsapproximation in Innenräumen.
A. Bahillo et al. beschreiben in ihrer Arbeit [BAH10] ein Lokalisierungssystem, das durch die Verwendung zweier Abstandsmessverfahren zu mehr als drei Ankerknoten eine Positionsbestimmung im Innenraumbereich ermöglichen soll. Die Lokalisierungspräzision der aufkommenden *UWB* – Technologie zeigen Ryosuke Fujiwara et al. prototypisch in ihrer Arbeit [FUJ09]. Es wird ein hybrides Abstandsmessverfahren auf Grundlage zweier Laufzeitmessmethoden vorgestellt, das unter Verwendung der *UWB* – Transceiver eine Abweichung von *25 cm* pro Einzelmessung aufweist. Beide Arbeiten basieren auf experimenteller Hardware. Ziel dieser Diplomarbeit soll es deshalb weiterhin sein zu überprüfen ob sich die Abweichungen eines auf Serienhardware basierenden Laufzeitmesssystems in Innenräumen mit hybriden Methoden verringern lassen.

Die vorliegende Arbeit beschreibt einleitend die bekannten Methoden zur Abstandsbestimmung, welche in *WSNs* verwendet werden können, und sie zeigt deren

physikalische Begrenzungen auf. Kapitel *2* stellt die dieser Arbeit zu Grunde liegende Hard- und Softwareplattform vor, dabei wird die theoretisch erreichbare Präzision anhand der technischen Gegebenheiten erörtert. Im Rahmen dieser Arbeit wurden umfassende messtechnische Analysen des Zeitverhaltens der verwendeten Hard- und Softwarekomponenten durchgeführt. Die Ergebnisse dieser Analysen werden im ersten Teil des dritten Kapitels präsentiert, und sie bilden die Grundlage für die im zweiten Teil des dritten Kapitels erläuterte Implementierung des Laufzeitmesssystems. Darauf folgend wird der Aufbau und die Durchführung der in unterschiedlichen Umgebungen durchgeführten Experimente bis hin zum praxisnahen Lokalisierungsversuch beschrieben. In diesem Lokalisierungsversuch wird das in der Arbeit entwickelte System mit einem kommerziellen Produkt verglichen. Im Kapitel Analyse und Evaluierung erfolgt die Vorstellung der erreichten Präzision des entwickelten Laufzeitmesssystems. Um diese abschließend werten zu können wird sie mit der Präzision eines kommerziellen Produktes verglichen.

1.1 Lokalisierung in drahtlosen Sensornetzen

Die Lokalisierung dient zur Bestimmung der Position von Sensorknoten in drahtlosen Sensornetzen. Manche Verfahren zur Lokalisierung setzten eine mehr oder weniger komplexe Infrastruktur voraus, andere können ohne eine vorher installierte Infrastruktur Positionen innerhalb des drahtlosen Sensornetzes bestimmen. Dazu ist eine Kommunikation zwischen den Sensorknoten notwendig. Die Kommunikation erfolgt dabei mit Funk- oder Schallwellen oder Kombinationen derer. Es ist Voraussetzung, dass sich das zu lokalisierende Objekt in der Kommunikationsreichweite eines oder mehrerer Anker befindet, deren Position bekannt ist. Wenn ein Anker einen zu lokalisierenden Sensorknoten in seinem Kommunikationsbereich detektiert, spricht man von Zellortung (CoO, engl.: cell of origin). Hierbei ist eine grobe Positionsbestimmung möglich. Sie wird von Systemen mit engmaschigen Infrastrukturen genutzt (Vgl. [HAR99]). Manche Systeme nutzen mehrere Emitter- und Detektorkombinationen, installiert an einem Anker, um die Zelle aufzuteilen. Um die Position des Objektes noch genauer bestimmen zu können, wird nun der Abstand zu einem oder meist

mehreren Ankern durch unterschiedliche Messmethoden bestimmt. Eine Übersicht über die aktuell verbreiteten Methoden zur Abstandsbestimmung erfolgt in den folgenden Unterkapiteln. Es ist auch möglich den Winkel der auftreffenden Wellen zu messen, um die Postion besser bestimmen können. Solche Verfahren finden aber wegen der Komplexität der dafür notwendigen Hardware und der unbedingt notwendigen genauen Ausrichtung der Anker, keine Verwendung im Umfeld von drahtlosen Sensornetzen, und werden deshalb nicht näher betrachtet.

Allgemein ist die Genauigkeit des jeweilig eingesetzten Messverfahrens zur Abstandsbestimmung ausschlaggebend für die Präzision der Ergebnisse des Lokalisierungsverfahrens. Oft werden diese Messungen wiederholt durchgeführt um, durch entsprechende Nachverarbeitung mit stochastische Methoden oder anderen Algorithmen wie zum Beispiel *AFL* [PRI03], die Präzision der Positionsbestimmung zu verbessern. Die anschließende Weiterverarbeitung kann unterschiedlich, auf Grundlage der Präzision des Ergebnisses der Messung, erfolgen. Mathematische Verfahren wie Trilateration lassen, präzise Abstandsmessergebnisse vorausgesetzt, gute Positionsbestimmungen erwarten und können mit aktueller Sensorknotenhardware zum Einsatz kommen. Bei fehlerbehafteten Messergebnissen ist der mathematische Ansatz, aufgrund der nichttrivialen Modellierung der Eigenschaften des Messsystems zur Fehlerkompensation, schwieriger zu handhaben. Da der Fehler der Messwerte keine gaußsche Verteilung aufweist und die benutzen Modelle sehr komplex werden, sind mathematische Verfahren sehr rechenintensiv und für Sensorknoten zu schwergewichtig und werden deshalb oftmals auf externe leistungsstärkere Systeme ausgelagert. Ein Ansatz, welcher der vorgenannten Problematik Rechnung trägt und den Stand der aktuellen Forschung widerspiegelt, wird von Widyawan et al. [WID08] vorgestellt, hier wird die Umgebung in einer vorbereitenden Initialisierungsphase kartographiert. Die Position des Sensorknotens wird in der aktiven Phase des Systems durch Vergleich der aktuell ermittelten Messwerte mit den Kartendaten ermittelt, und durch sogenanntes *map filtering* oder auch *map matching* verifiziert und korrigiert. Die *map matching* Technik filtert unmögliche Positionen, wie zum Beispiel Wände oder andere Hindernisse, an denen sich das zu lokalisierende Objekt nicht befinden kann, aus. Zeichnet man diese möglichen Positionen auf, erhält man mögliche Pfade, auf welchen sich das Objekt bewegt haben könnte. Diese Pfade las-

sen sich nun wiederum mittels *map matching* Technik verifizieren und man kann im die wahrscheinlichsten Positionen auswählen. Da sich die Umgebungsparameter aber ständig ändern können, ist es notwendig die Kartendaten aktuell zu halten. Dies ist ein sehr aufwändiger Prozess.

Ein gänzlich anders funktionierendes Lokalisierungsverfahren bedient sich Inertialmesssystemen und kommt ohne Kommunikation und Anker aus. Warum auch hier Abstandsmessungen interessant sind folgt im Kapitel Inertialmesssysteme IMU. Das folgende Kapitel geht näher auf funkbasierte Abstandsmessverfahren ein und beschreibt deren Technologien und Methodik.

1.1.1 Funkbasierte Abstandsmessverfahren

Funkbasierte Abstandsmessverfahren werden auf Basis vieler verfügbarer Funktechnologien realisiert. Sie lassen sich aber in wenige, in den folgenden Unterkapiteln genauer beschriebene, Methoden klassifizieren. Die verbreitetsten Funktechnologien für *WSNs* sind:

- **RFID** ist eine Technologie welche zur automatischen Identifizierung eingesetzt wird. Dabei werden Transponder, welche die Informationen zur Identifizierung enthalten, oftmals im UHF-Band mit Lesegeräten ausgelesen. Man unterscheidet passive und aktive Transponder. Passive Transponder sind klein und besitzen keine eigene Stromversorgung. Wenn sich ein passive Transponder einem Lesegerät nähert, wird er induktiv mit Strom versorgt so dass die auf dem Transponder befindlichen Daten gelesen werden können. Aktive Transponder besitzen eine eigene Stromversorgung und können, wie zum Beispiel im System von C. Chien [CHI03] beschrieben, zur Lokalisierung eingesetzt werden. In diesem System wir die Leistungsdichte, des empfangenen Signals, zur Abstandsbestimmung eingesetzt.
- **Wi-Fi** (*IEEE802.11*) ist die drahtlosen lokalen Netzwerken (*WLAN*) zugrundeliegende Technologie. Aufgrund der weiten Verbreitung und der im, *IEEE802.11* Standard spezifizierten, hardwareunterstützten Empfangsleistungsmessung wird die Wi-Fi-Technologie oft für Lokalisierungssysteme einge-

setzt und wird auch schon in der Praxis zum Beispiel von *GoogleMaps*[1] in Kombination mit anderen Technologien, zum Beispiel *GPS* erfolgreich eingesetzt. Aufgrund des relativ hohen Stromverbrauchs der verfügbaren Transceiver dieser Technologie, wird sie nicht in drahtlosen Sensornetzen eingesetzt. [WIB09] Spezifiziert ein Abstandsmesssystem auf Basis der Wi-Fi-Technologie mit eine Präzision im zwei Meter Bereich.

- Die **Bluetooth** (IEEE802.15.1) Technologie findet in drahtlosen *ad hoc* Netzwerken Verwendung. Hierbei liegt der Fokus auf Datenübertragung auf kurzen Distanzen (< *100 m*). Aufgrund dessen und wegen des Stromverbrauch der Bluetoothtransceiver ist sie in *WSNs* wenig vertreten.

- **ZigBee (**und IEEE802.15.4) spezifiziert eine weitere drahtlose Kurzstreckenfunktechnologie, welche auch in *WSNs* zum Einsatz kommt. Dabei wird ein Protokollstapel definiert, wobei die Bitübertragungs- und Sicherungsschicht in IEEE802.15.4 definiert sind. Die weiteren Schichten, wie Vermittlungs-, Sicherheits- und Anwendungsschicht, standardisiert ZigBee. Es werden verschiedene Gerätearten, wie Endgerät, Router und Koordinator definiert und deren Verhalten, zum Beispiel beim Aufbau der Netzwerktopologie, spezifiziert. Diese Geräte bilden ein hierarchisches Netzwerk mit genau einem Router als Koordinator an der Spitze. Ein in [GRO07] beschriebenes Messsystem, mit einer Lokalisierungsabweichung von *2 m*, stützt sich auf den *link quality index* (*LQI* Vgl. 2.1.2), welcher im Kapitel Digitale Signalverarbeitung erläutert wird.

- Für Datenübertragungen über kurze Distanzen mit hoher Datenrate wurde die ***UWB*** Technologie entwickelt. Bei dieser Technologie werden die Daten über ein sehr breites Frequenzspektrum in sehr kurzer Zeit übertragen, deshalb ist diese weniger empfindlich gegenüber Störungen. Es sind erste *UWB* Transceiver speziell für drahtlose Sensornetze vorgestellt worden, welche bei äquivalenten Stromverbrauch wie herkömmliche Funktechnologien, größere Datenraten erreichen können. T. Nakagawa stellt in [NAK08] einen nur einen Kubikzentimeter großen Sensorknoten vor, welcher mit einem *UWB* Impuls-

1 Google Maps ist eine Software welche für viele Handy – Plattformen verfügbar ist. Weitere Informationen im Internet unter http://www.google.de/mobile/maps/ (2.11.2010)

radiochip ausgestattet ist. Eine Einschränkung der *UWB* Technologie ist die geringe Reichweite, sie liegt beispielsweise im zuvor genannten System beispielsweise bei 30 m. Mit dem in [SHA08] vorgestellten Innenraumlokalisierungssystem zur Roboternavigation kann, auf Grundlage der *UWB* Technologie und Laufzeitmessung, eine mittlere Lokalisierungsgenauigkeit von *15 cm* erreicht werden.

1.1.1.1 Leistungsdichtemessung RSSI

Der Kennwert für die Leistungsdichte des empfangenen Signals ist der *received signal strength indicator* (*RSSI*, auch *RSS*). Dieser wird im Empfänger beim Eintreffen eines Funksignals von der gemessenen Eingangssignalstärke und vom Signalverstärkungsgrad des Empfangsteils des Transceivers abgeleitet [TEX10]. Dieser Wert dient zur Erkennung eines freien Funkkanals und wird auch zur Einstellung der internen Signalverstärkung des Transceivers ausgewertet. Elektromagnetische Wellen verlieren bei ihrer Ausbreitung in der Luft an Leistungsdichte. Geht man von einem isotropen Kugelstrahler als Sendeantenne aus, so breiten sich diese Wellen im freien Raum gleichmäßig aus. Ist nun die Leistung P_R,an der isotropen Empfängerantenne gesucht, welcher sich in einem Abstand von r befindet, so entspricht P_R anschaulich einem Punkt auf einer Kugel mit Radius r. Also errechnet sich P_R aus der abgestrahlten Leistung am Sender P_T geteilt durch die Fläche der Kugel mit dem Durchmesser r also:

$$P_R = \frac{P_T}{4\pi r^2} \tag{1.1}$$

Die Fläche welche eine reale Antenne in der Wellenfront einnimmt ist größer als die einer idealisierten isotropen Antenne. Die Leistung welche sie der Wellenfront entnehmen kann ist abhängig von der Wellenlänge und der Leistungsdichte der ausgesendeten Wellen. Harald T. Friis formuliert die Gleichung für die zu erwartende Leistung am Empfänger P_R in Abhängigkeit vom Gewinn beider Antennen G_T G_R und der Wellenlänge des Signals λ wie folgt [TEX08]:

$$P_R = P_T \frac{G_T G_R \lambda^2}{(4\pi)^2 r^n} \quad , n=2 \quad \textit{für Freiraumausbreitung} \tag{1.2}$$

Da für ein auf der Empfangsleistungsdichte basierendes Messsystem der Antennengewinn, die Sendeleistung sowie die Trägerfrequenz hinreichend genau bekannt sind, lässt sich in der Theorie die Entfernung zwischen Sender und Empfänger ermitteln. Über den Parameter *n* lassen sich Umgebungsparameter in die Gleichung (1.2) einbringen um zum Beispiel den Leistungsverlust in verschiedenen Materialien ableiten zu können. Wie dieser Parameter zu bemessen ist, ist in der Realität besonders im Innenraumbereich, aufgrund der verschiedenen Einflüsse auf die Ausbreitung der Wellen (Vgl. 1.1.4), nur experimentell für eine bestimmte Umgebung zu ermitteln.

Es ist erwiesen, dass sich nur unter Laborbedingungen Entfernungen hinreichend genau vom RSSI ableiten lassen [LOP10]. Obwohl bessere Modelle zur Abschätzung der Entfernung anhand des Leistungsverlustes entwickelt wurden, sind diese zu komplex für Sensorknoten oder nicht präzise genug um allein anhand des *RSSI* den Abstand, bei ständig wechselnden Umgebungsparametern, verlässlich zu ermittelten. In der Realität wird deshalb, wie einleitend beschrieben und in [WID08] beispielhaft gezeigt, die Position anhand initial gesammelter Daten über stochastische Verfahren angenähert. Somit können auf Grundlage des RSSI nur Infrastruktur basierte Systeme entwickelt werden, deren Infrastruktur kartographiert werden kann und Rechenleistung zur Verfügung stellt. In der Arbeit [WID08] wird auch ein Ansatz zur automatischen Anpassung an veränderte Umgebungsparameter vorgestellt, welcher durch Messungen zwischen den Basisstationen initiiert wird. Ohne eine Infrastruktur, mit der sich Abweichungen korrigieren lassen, eignet sich der *RSSI* nur zur Validierung anderer Messmethoden (siehe Kapitel Hybride Methoden).

1.1.1.2 Laufzeitmessung

Elektromagnetische Wellen breiten sich im Vakuum mit Lichtgeschwindigkeit *c* aus:

$$c = 299792458 \ \frac{m}{s} \qquad (1.3)$$

Um präzise Abstandsmessungen durchführen zu können muss man Zeitintervalle von einer Nanosekunden auflösen können. Dazu ist es notwendig das Funksignal mit einer Abtastrate im Gigahertzbereich abzutasten. Das leisten die Funkchips welche für drahtlose Sensornetze entwickelt sind nicht. In wissenschaftlichen Arbeiten werden

deshalb oft s*oftware defined radio* - Entwicklungsplattformen eingesetzt. Grundlage dieser Plattform ist ein breitbandiges *HF-/Analog Front-End* mit nachgeschalteten hochfrequent getakteten *AD/DA*-Wandler. Danach erfolgt die gesamte Signalverarbeitung per Software welche auf einem *FPGA* ausgeführt wird. Somit ist es möglich einen für die Laufzeitmessung angepassten Transceiver zu entwickeln, welcher mit den benötigten Taktraten arbeitet.

Es gibt bereits eigens für die Entfernungsmessung entwickelte Transceiver, wie zum Beispiel der *NanoPAN* [NAN09] Transceiver, ein kommerzielles Produkt der Firma *Nanotron.* Dieser Transceiver wird im *FeuerWhere* Projekt zur Lokalisierung von Personen eingesetzt. Die Laufzeitmessung erfolgt bei *NanoPAN* vollständig im Transceiver, da dieser einen eigenen Mikrocontroller für die Laufzeitmessung besitzt. In [WIB09] zeigen Wibowo et al. das es durch statistische Methoden möglich ist eine höhere Genauigkeit zu erreichen als die Abtastrate es vorgibt.

Bei der Laufzeitmessung wird das Zeitintervall gemessen, welches sich vom Aussenden bis zum Empfangen eines Signals aufspannt. Dieses Zeitintervall wird durch die Ausbreitungsdauer der elektromagnetischen Wellen in der Luft definiert. Aus dem Zeitintervall t und der Ausbreitungsgeschwindigkeit der Wellen c lässt sich der Abstand d wie folgt ermitteln:

$$d = c \cdot t \tag{1.4}$$

Um dieses Zeitintervall ermitteln zu können gibt es mehrere, im folgenden erläuterte Methoden. Die am schwierigsten zu realisierende Methode, da sie auf der Annahme basiert, dass Sender und Empfänger zeitsynchronisiert sind, wird als ***time of flight*** (*TOF*) bezeichnet. Ein Sender sendet ein Signal welches den Zeitpunkt t_0 des Versendens enthält. Den Empfänger erreicht das Signal zum Zeitpunkt t_1, der dann mit dem zwischen t_0 und t_1 aufgespannten Zeitintervall in Gleichung (1.4) eingesetzt den Abstand wie folgt ermitteln kann:

$$d = c \cdot (t_1 - t_0) \tag{1.5}$$

Die ***time difference of arrival*** - Methode(*TDoA*) geht von zeitsynchronisierten Ankerknoten aus, was im Umfeld von *WSNs* realistischer ist, als diese Bedingung für alle teilnehmenden Knoten aufzustellen. Um eine Lokalisierung durchzuführen sendet der zu lokalisierende mobile Knoten *M* ein Signal aus. Zur eindeutigen Lokalisierung müssen mindestens drei Ankerknoten (*A*, *B* und *C)*, deren Positionen zum Beispiel mit $(x_A,y_A),(x_B,y_B)$ und (x_C,y_C) in einem zweidimensionalen Koordinatensystem gegeben sind, dieses Signal empfangen. Da, wie Abbildung 1.1 zeigt, der Abstand des Knotens *M* zu den einzelnen Ankerknoten in aller Regel unterschiedlich ist, erreicht das Signal die Ankerknoten *A*,*B* und *C* zu den Zeitpunkten t_a, t_b und t_c. Dieser Abstand lässt sich, wenn alle drei Zeitpunkte bekannt sind, für jeden Ankerknoten, durch die Ausbreitungsgeschwindigkeit *c* (1.3) als Abstandsdifferenz Δd, zwischen den Abständen zweier Ankerknoten und dem mobilen Knoten, ausdrücken:

$$\begin{aligned} \Delta d_{AB} &= c \cdot |t_A - t_B| = d_{AM} - d_{BM} \\ \Delta d_{AC} &= c \cdot |t_A - t_C| = d_{AM} - d_{CM} \\ \Delta d_{BC} &= c \cdot |t_B - t_C| = d_{BM} - d_{CM} \end{aligned} \tag{1.6}$$

Wobei d_{AM}, d_{BM} und d_{CM} den Abstand zwischen dem jeweiligen Ankerknoten und dem mobilen Knoten definiert. Der Abstand zweier Punkte in einem zweidimensionalen Koordinatensystem lässt sich, mit Hilfe des Satzes von Pythagoras bestimmen. Es ergeben sich für die Abstände d_{AM}, d_{BM} und d_{CM} also die Gleichungen:

$$\begin{aligned} d_{AM} &= \sqrt{(x_M - x_A)^2 + (y_M - y_A)^2} \\ d_{BM} &= \sqrt{(x_M - x_B)^2 + (y_M - y_B)^2} \\ d_{CM} &= \sqrt{(x_M - x_C)^2 + (y_M - y_C)^2} \end{aligned} \tag{1.7}$$

Nach Einsetzung der Gleichungen (1.7) in die Gleichungen aus (1.6) ergeben sich:

$$\begin{aligned} \Delta d_{AB} &= \sqrt{(x_M - x_A)^2 + (y_M - y_A)^2} - \sqrt{(x_M - x_B)^2 + (y_M - y_B)^2} \\ \Delta d_{AC} &= \sqrt{(x_M - x_A)^2 + (y_M - y_A)^2} - \sqrt{(x_M - x_C)^2 + (y_M - y_C)^2} \\ \Delta d_{BC} &= \sqrt{(x_M - x_B)^2 + (y_M - y_B)^2} - \sqrt{(x_M - x_C)^2 + (y_M - y_C)^2} \end{aligned} \tag{1.8}$$

Da die Positionen der Ankerknoten mit $(x_A,y_A),(x_B,y_B)$ und (x_C,y_C) bekannt sind und die Abstanddifferenzen Δd_{AB}, Δd_{AC} und Δd_{BC} durch die Zeitpunkte t_a, t_b und t_c mit (1.6) ermittelt werden können, sind mögliche Positionen für mobilen Knoten $M(x_M,y_M)$ als Lösungen für (1.8) definiert.

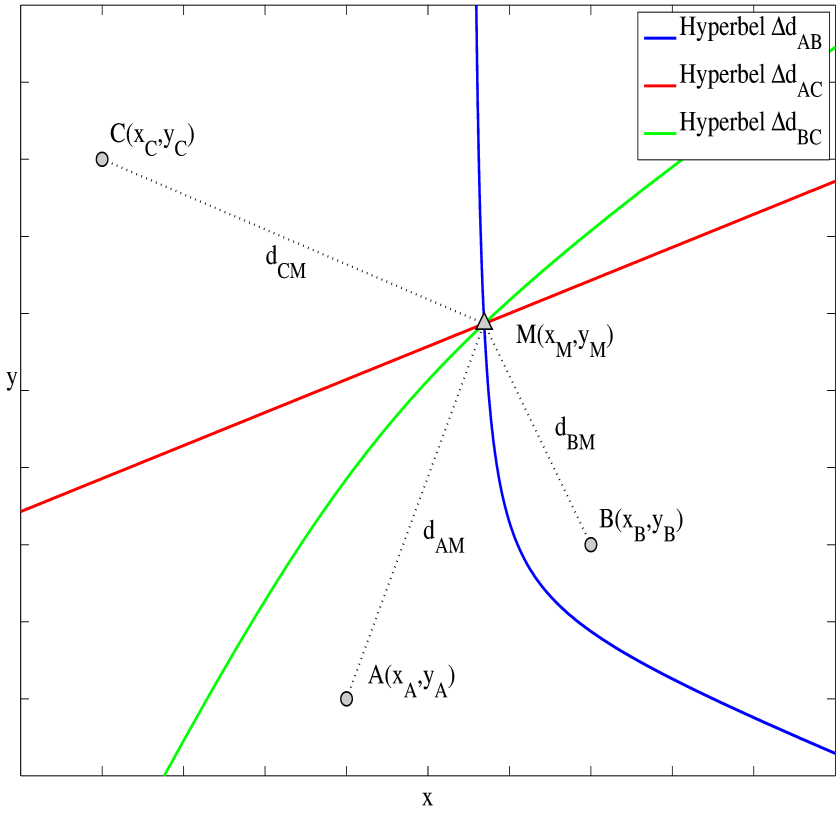

Abbildung 1.1 Berechnung der Position mit TDoA

Wie in Abbildung 1.1 graphisch dargestellt, beschreibt die Lösungsmenge jeder Gleichung eine Hyperbel zwischen den beiden in der Gleichung referenzierten Ankerknoten. Es ist die Kurve der Hyperbel als Lösung zu wählen, welche sich näher an dem Anker befindet, welcher das Signal zuerst empfangen hat. Der Schnittpunkt zweier Hyperbelkurven ergibt dann die gemessene Position (x_M,y_M) für den zu lokalisierenden Knoten *M*. Da sich zwei Hyperbeln auch in zwei Punkten schneiden können, präzisiert die Bestimmung des Schnittpunktes einer dritten Hyperbel die Position.

Gänzlich ohne Zeitsynchronisation kommt die ***round trip time of flight*** – Methode (*RToF*, auch *RTT*, oft ebenfalls *TOF* bezeichnet) aus. Mit Dieser ist es möglich den Abstand zweier Knoten direkt zu bestimmen. Dazu antwortet der Empfänger *E* dem Absender *A* eines, zum Zeitpunkt t_0 ausgesandten, Laufzeitmesssignals wiederum mit einem Signal. Dieses Signal erreicht den Absender *A* zum Zeitpunkt t_1. Ist nun das Bearbeitungszeitintervall t_{Latenz} zum Versenden des Antwortsignals bekannt

und wird dem Absender *A* übermittelt so ergibt sich die Distanz *d*, analog zu (1.5),wie folgt:

$$d = c \cdot (t_1 - t_0 - t_{Latenz}) \tag{1.9}$$

Der Fokus dieser Arbeit liegt auf der infrastrukturlosen *RToF* - Methode, da es Ziel dieser Arbeit ist *ad hoc* Distanzmessungen in einem drahtlosen Sensornetz durchführen zu können.

1.1.1.3 Hybride Methoden

Die Genauigkeit der Abstandsmessmethoden ist, wie in Kapitel Fehlerquellen der Messmethoden erläutert, gerade in Innenräumen beschränkt. Hybride Methoden verbinden mehrere Messmethoden, wie in [BAH10] von A. Bahillo et al. vorgestellt, um Messwerte aus Leistungsmessungen mit solchen aus Laufzeitmessungen zu validieren. Dazu wird der *RSSI* - Wert gemessen und zusätzlich dazu eine Laufzeitmessung mittels *RToF* – Methode durchgeführt. Aus dem Abstandswert welcher aus der Laufzeitmessung berechnet wurde wird mit dem erwarteten Fehler der Messung ein minimaler und ein maximaler Abstand bestimmt. Aus diesen beiden Abständen wird mit der Gleichung für die Freiraumdämpfung (1.2) eine obere und untere Schranke für den *RSSI* – Wert errechnet. Es wurde in [BAH10] gezeigt, dass sich auf diesem Wege der mittlere Lokalisierungsabweichung reduzieren lässt, obwohl die Größe des Fehlers beider Werte von den selben Umgebungsparametern abhängt. Eine weitere, von Nakagawa et al. in [NAK08] entwickelte, hybride Methode nutzt die Messmethoden RToF und TDoA, um die Anzahl der benötigten Ankerknoten zu reduzieren. Das hybride inertial System [KÖP10] wurde oben bereits vorgestellt. In den Experimenten, welche im Rahmen dieser Diplomarbeit durchgeführt wurden, ist zusätzlich untersucht worden, ob die Einbeziehung der Werte *RSSI* und *LQI* eine Verbesserung der Werte der Laufzeitmessung bewirkt.

1.1.2 Inertialmesssysteme IMU

Mit Beschleunigungs- und Drehratensensoren verfolgen Inertialmesssysteme Positions- und Lageveränderungen. Durch die dreiachsige Auslegung der genannten Sensoren können Beschleunigungen und Drehungen in jede Richtung aufgezeichnet wer-

den. Anhand der aufgezeichneten Daten kann, ausgehend von einer bekannten Startposition, die Position berechnet werden. Inertialmesssysteme sind seit langem unter anderen in der Luft- und Raumfahrttechnik etabliert. Durch die Verkleinerung der Sensoren auf Chipgröße und die Senkung des Stromverbrauchs können Inertialmesssysteme im Bereich von WSNs, zur Lokalisierung benutzt werden. Inertialmesssysteme haben die Eigenschaft das auftretende Abweichungen, wie die Drift[1] eines Drehratensensors, sich aufsummieren und negativ auf die Präzision der berechneten Position auswirken. Um die Drift des Drehratensensors korrigieren zu können werden Magnetkompasssensoren eingesetzt, mit welchen eine detektierte Drehung verifiziert werden kann. Mit Sensoren für den barometrischen Druck kann eine Veränderung der Höhe überprüft werden. Mit funkbasierten Abstandsmessungen kann die Präzision des Verfahrens verbessert werden, dabei wird die inertial errechnete Position mit der aus den Abstandsmessung überprüft und ggf. korrigiert. Ein solches hybrides inertiales Messsystem wurde zur *IPIN 2010*[2] vorgestellt und erreicht eine mittlere Lokalisierungsgenauigkeit von 2m.[KÖP10]

1.1.3 Akustische Verfahren

Da Mikrocontroller verbreiteter Sensorknotenplattformen mit einer niedrigen Taktfrequenz (z.B. *8 MHz MSB430-H* [HIL07]) betrieben werden, ist es mit diesen nicht möglich kurze Zeitintervalle zu messen. Schallwellen breiten sich in Luft bedeutend langsamer aus als elektromagnetische Wellen, sodass es zur Abstandbestimmung ausreichend ist, Zeitintervalle im Mikrosekundenbreich messen zu können. Das leisten alle Sensorknotenplattformen zweifelsfrei ohne großen Aufwand. Das von Andy Harter et al. vorgestellte *BAT* System ([HAR99]) basiert auf Laufzeitmessung von Schallwellen im Ultraschallbereich und kann Abstände mit einem mittleren Fehler von *15 cm* (ohne Filterung *50 cm*) bestimmen. Ultraschallwellen werden in der Luft stark gedämpft, deshalb ist die Reichweite eingeschränkt und eine komplexe Infrastruktur mit einer große Anzahl von Lokalisierungsankern notwendig. Zusätzlich zu den Problemen der Wellenausbreitung, welche im folgenden Kapitel erläutert werden, existieren keine isotropen Emitter für Ultraschallwellen und viele Materialien

1 Der Drehratensensor ändert den Wert der Drehrate obwohl tatsächlich keine Drehung erfolgt.
2 International Conference on Indoor Positioning and Indoor Navigation 2010 Zürich

sind für diese undurchlässig. Deshalb ist ein Ultraschallsignal immer gerichtet. In [HAR99] ist gezeigt, dass dieses Problem sehr deutliche Auswirkungen im Praxiseinsatz, in Form von „toten“ Bereichen in der Nähe von Hindernissen oder Wänden, hat. In diesen Bereichen ist keine Lokalisierung möglich. Auch die Art der Befestigung der „*bat units“* an den zu lokalisierenden Objekten hat Einfluss auf die Präzision der Abstandsmessungen. Es muss dabei darauf geachtet werden, dass der Ultraschallemitter nicht abgedeckt und möglichst gut, auf die in der Decke installierten Detektoren der Basisstationen, ausgerichtet ist.

1.1.4 Fehlerquellen der Messmethoden

Elektromagnetische Wellen breiten sich bei Sichtverbindung zwischen Sender und Empfänger *(LoS, line of sight)*, geradlinig aus. Dabei schwächt sich die Leistungsdichte der Wellen mit steigender Entfernung ab. Sind Hindernisse in der *LoS* vorhanden, wird einerseits die Ausbreitungsrichtung der Wellen durch verschiedene Effekte beeinflusst, andererseits ist die Dämpfung, der die Wellen durch das Hindernis unterliegen, höher als in freier Luft. Die Stärke der Dämpfung ist dabei abhängig vom Material des Hindernisses. Im folgenden wird der Einfluss von Hindernissen auf die verschiedenen Messmethoden erläutert.

Da gerade im Innenraumbereich viele verschiedene Hindernisse auf dem Weg vom Sender zum Empfänger als Übertragungsmedium dienen können, muss die Dämpfung welche das Signal auf diesem Weg erfährt, für jedes einzelne durchquerte Hindernis berechnet werden. Diese Hindernisse besitzen keine homogene Materialstruktur. So ist zum Beispiel der Aufbau einer typischen Wand aus Mauerwerk in verschiedene Schichten, wie Farbe, Tapete, Putz, Mauerwerk und Mörtel unterteilt, welche die Wellen unterschiedlich stark dämpfen. Dadurch ist der Dämpfungsfaktor der einzelnen Hindernisse nicht berechenbar und kann nur experimentell angenähert werden. Es ist deshalb, selbst bei bekannter Anzahl und Materialbeschaffenheit der Hindernisse, kaum berechenbar wie hoch die Leistungsdichte der Wellen beim Empfänger sein wird. Bei der Verwendung der *RSSI* – Methode wird von Freiraumausbreitung ausgegangen, sodass die Abstandsmessungen bei vorhandenen Hindernissen in

der Regel einen zu langen Abstand ergeben werden. Es ist weiterhin möglich, dass die Wellen, durch die im folgenden näher beschriebenen Effekte, einen längeren Ausbreitungsweg benutzten als den geradlinigen. Die Abstände zwischen Sender und Empfängerknoten, werden in diesem Fall zu groß berechnet werden, da die Wellen durch den längeren Weg stärker gedämpft eintreffen als erwartet. Der Einfluss von Hindernissen kann dazu führen, dass die Leistungsdichte, an einem vom Sender weiter entfernten Punkt, höher ist als an einem Punkt welcher sich näher am Sender befindet (Vgl. [LOP10] Kapitel 1.2), was auf den nachfolgend erläuterten *multipath* Effekt zurückzuführen ist. Die Abweichung von *RSSI* basierten Abstandsmessmethoden ist also von den Umgebungsparametern abhängig, welche sich, zum Beispiel in einem Bürogebäude, durch veränderliche Hindernisse (Person, Türen und Fenster), ständig ändern können.

Bei laufzeitbasierten Messmethoden wie *RToF* und *TDoA* hat die Dämpfung nur untergeordneten Einfluss auf die Abweichung der Messwerte vom wahren Wert. Es ist von größerer Bedeutung, welchem Weg die elektromagnetischen Wellen vom Sender zum Empfänger folgen. Dabei bestimmen nach [SCH03] die folgenden Effekte, abhängig von der Wellenlänge, den Ausbreitungsweg:

- Reflexion
- Beugung
- Brechung
- Streuung

An Hindernissen, welche deutlich größer sind als die Wellenlänge, werden die Wellen **reflektiert**, das heißt ein Teil der Wellen wird im selben Winkel vom Hindernis zurückgeworfen, in welchem sie aufgetroffen sind. Ein weiterer Effekt welcher die Richtung der Wellen verändert ist die **Beugung**. An einem scharfkantigen Hindernis werden die Wellen, durch Wechselwirkungen mit den vom Hindernis beeinflussten Anteilen der Wellen, leicht in Ihrer Richtung abgelenkt. Treten die Wellen in eine Grenzschicht zwischen verschiedenen Übertragungsmedien ein, so werden sie, durch die verschiedenen Ausbreitungsgeschwindigkeiten der Wellen in den Übertragungsmedien, leicht in ihrer Richtung abgelenkt, dieser Effekt wird **Brechung** ge-

nannt. Aus jedem dieser Effekte resultiert ein Ausbreitungsweg, welcher länger ist als der tatsächliche Abstand zwischen Sender und Empfänger. Da nur ein Teil der Wellen reflektiert, gebeugt oder gebrochen wird, werden die Wellen stärker gedämpft als bei einem gleich langen Ausbreitungsweg in freier Luft. Die **Streuung** von Wellen ist besonders stark, wenn die Objekte im Übertragungsweg deutlich kleiner sind als die Wellenlänge. Elektromagnetische Wellen werden durch Streuung stark gedämpft. Weitere Effekte sind die Abschattung des Empfängers und Blockierung des Senders durch für elektromagnetische Wellen undurchdringliche Hindernisse.Diese Effekte führen dazu das keine Messungen mehr durchgeführt werden können, da die vom Sender ausgesandten Wellen den Empfänger nicht mehr erreichen.

In der Realität breiten sich Wellen zwischen Sender und Empfänger, außer bei *LoS* Bedingungen, nicht geradlinig aus sondern unterliegen mehrfach den vorgenannten Effekten, welche die Laufzeit und die Leistungsdichte beeinflussen. Man spricht dann von *multipath* Ausbreitung, bei welcher aus einem vom Sender ausgesandten Signal mehrere zu unterschiedlichen Zeiten beim Empfänger eintreffende, unterschiedlich gedämpfte Signale entstehen. Die unterschiedlichen Signale überlagern sich beim Empfänger und Verändern das Ausgangssignal in Frequenz, Phasenlage und Amplitude, was auch die Zerstörung der im Signal enthaltenen Information zur Folge haben kann. Die *multipath* Ausbreitung beeinflusst die Ergebnisse von Laufzeitmessungen und Leistungsdichtemessung erheblich, da beim Empfänger des Signals nicht reproduziert werden kann ob das Signal den kürzesten Weg zwischen Sender und Empfängerknoten genommen hat oder nicht. Ein Ansatz, dem Problem der *multipath* Ausbreitung zu Begegnen ist die Benutzung der *UWB* Technologie. Bei dieser Technologie ist das Signal so kurz, dass sich das Ursprungssignal nicht mit den *multipath* Signalen überlagert. Dadurch kann das Signal welches den direkten Weg genommen hat, von den anderen später eintreffenden unterschieden werden.

Wann von unbeeinflusster Ausbreitung der Wellen ausgegangen werden kann, hängt von der Wellenlänge des Signals und dem Abstand von Sender- und Empfängerknoten ab. Es ist zu Beachten das es sich bei der *LoS* nicht um eine gedachte Linie handelt, sondern um eine Zone, welche frei von Hindernissen sein muss. Diese Zone

ist von dem französischen Ingenieur Augustine Jean Fresnel, durch Experimente mit Licht, erforscht worden. Er hat festgestellt, dass reflektierte Wellen das Signal beim Empfänger nur dann in ihrer Leistungsdichte beeinflussen, wenn der Umweg, welchen die Wellen durch die Reflexion genommen haben, nicht mehr als eine halbe Wellenlänge größer ist als der direkte Weg. Diese sogenannte 1. fresnel'schen Zone ist, abhängig von der Wellenlänge λ des Signals und dem Abstand von Sender und Empfängerknoten d, durch folgende Gleichung definiert:

$$r = \sqrt{\frac{\lambda \cdot d_S \cdot d_E}{d}} \tag{1.10}$$

Mit dieser Gleichung lässt sich der Radius r der 1. fresnel'schen Zone in Abhängigkeit der Entfernung von Sender und Empfängerknoten d_S, d_E berechnen.

Des weiteren unterliegt das empfangene Signal Veränderungen durch, in der heutigen Zeit allgegenwärtige, Störungen durch andere Sender in der Umgebung mit gleicher oder harmonischer Frequenz. Diese Veränderungen wirken sich ähnlich wie *multipath* Effekte auf das ursprüngliche Signal aus, sollten aber durch geeignete Medienzugriffsverfahren erkannt und verhindert werden können.

Zusammenfassend ist festzustellen, dass die Abweichung der Abstandsbestimmung aller Methoden in erster Linie keine Messabweichungen sind, denn der Weg welcher gemessen wurde entspricht dem, welchen die Wellen zurückgelegt haben. Dieser gemessene Weg ist aber nur im Fall von *LoS* der Abstand zwischen Sender und Empfängerknoten. Es ist deshalb notwendig Signale welche sich nicht entlang des direkten Weges zum Empfängerknoten ausgebreitet haben zu erkennen. Im Kapitel Analyse und Evaluierung dieser Diplomarbeit wird untersucht ob es anhand der Streuung der Werte der *RToF* – Messung, dem *RSSI* Wert und der Verbindungsqualität *(LQI)* möglich ist *LoS* Signale von anderen zu unterscheiden.

2 Laufzeitmesssystem

Einleitend wurde dargestellt, dass die Qualität der Ergebnisse von Laufzeitmesssystemen stark von der Präzision der ihnen zugrundeliegenden Messsysteme abhängt. Beispielhaft wurde gezeigt, dass sich durch den Einsatz spezialisierter Messhardware die Präzision der Messsysteme verbessern lässt. Die fortschreitende Entwicklung der Mikroprozessoren und digitaler Transceiver macht die neuste Generation von Sensorknoten so leistungsfähig, dass es möglich wird mit Standardhardware gute Messsysteme entwickeln zu können. Der Fokus dieser Arbeit besteht in der Entwicklung, experimentellen Untersuchung und Evaluierung, eines solchen Messsystems. Es soll abschließend dargestellt werden, welche Ergebnisse mit einem solchen Messsystem zu erreichen sind. Im folgenden Abschnitt werden die Hardware und Softwareanteile beschrieben, welche als Grundlage für die Entwicklung des Laufzeitmesssystems dienen.

2.1 Hardwareplattform

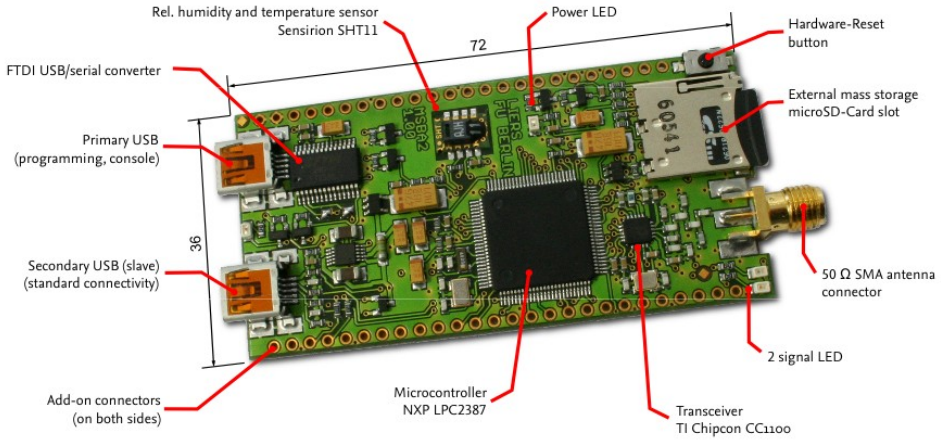

Abbildung 2.1 Die verwendete MSB-A2 Hardwareplattform.(Abb. aus [BAA08])

Die zugrundeliegende Hardwareplattform ist die neueste Generation einer Serie von Sensorknoten, welche an der Freien Universität Berlin entwickelt wurden. Abbildung 2.1 zeigt die *MSB-A2* Plattform, die sich vor allem durch die hohe Taktrate des eingesetzten Mikrocontrollers *NXP LPC2387* [NXP08] für Laufzeitmessung auszeichnet. Weitere für diese Arbeit besonders relevante Bestandteile der Plattform sind, der Transceiver *CC1100* von Texas Instruments [TEX09], die vorhandene *MCI*[1] Schnittstelle und die flexible und leichtgewichtige Softwareplattform. Die Grundlage für diese Softwareplattform wurde von Heiko Will in seiner Diplomarbeit gelegt [WIL08]. Besonders hervorzuheben ist die Flexibilität der *MSB-A2* Plattform. Diese erleichtert, durch die an den Seiten der Platine bereitgestellten Anschlüsse zu den Ein- und Ausgängen des Mikrocontrollers, die Forschungsarbeit erheblich. So sind Messungen mit Oszillographen oder Logikanalysatoren meist ohne vorherige Lötarbeiten möglich. Des weiteren konnten Hardwareänderungen, welche für die timergestützte Messung in dieser Arbeit nötig waren, schnell und unkompliziert durchgeführt werden. Ein Derivat dieser Plattform ist eine Erweiterung, welche im Rahmen des *BMBF* geförderten Projekts *AVS-Extrem* an der Freien Universität Berlin durchgeführt wurde [DEZ10]. Es wurden zusätzlich zur bestehenden Hardware ein Beschleu-

1 Multimedia card interface: Schnittstelle für SD/MMC Karte

nigungssensor (*SMB380*, Bosch) und ein *GPS* Empfänger (*FSA03*, Falcom) in die Plattform integriert. Weiterhin ist die Hardwareänderung, welche für die timergestützte Laufzeitmessung notwendig ist, im Design berücksichtigt, sodass der Einsatz und die Weiterentwicklung, des im Rahmen dieser Diplomarbeit entstandenen Laufzeitmesssystems, im Projekt *AVSExtrem* und folgenden Projekten möglich ist. Die Programmierung und die Standardeingabe und -Ausgabe erfolgt über die serielle Schnittstelle des Mikrocontrollers, welche über den *FTDI FT232RL* über *USB* zur Verfügung gestellt wird.

2.1.1 Der LPC2387 Mikrocontroller

Der *LPC2387* Mikrocontroller der Firma *NXP* basiert auf einem *ARM7TDMI-S* Kern und verfügt über *512 kB* großen Flashspeicher und *98 kB SRAM* und kann bis zu einer Taktfrequenz von *72 MHz* betrieben werden. Der SRAM teilt sich in einen *64 kB* großen ununterbrochenen Block als Arbeitsspeicher und zwei *16 kB* große Blöcke als Pufferspeicher für die USB - und Ethernet – Schnittstelle. Ein weiterer *2 kB* großer Block kann getrennt vom Rest des Mikrocontrollers zusammen mit der Echtzeituhr mit Strom versorgt werden. Diese separate Stromversorgung ist nicht vorgesehen, so dass dieser Speicherblock seinen Inhalt ohne die Stromversorgung zwar verliert, aber Daten resetfest darin abgelegt werden können. Die Schnittstellen und die Peripherie des LPC 2387 sind in Tabelle 2.1 aufgelistet.

	LPC 2387
Ethernet	DMA[1]
USB 2.0	Device / Host; DMA
UART	4 Stück mit eigenem FIFO
CAN	2 Kanäle
SPI	2 einer geteilt mit SSP
SSP	2, GPDMA
I2C	2
I2S	1, GPDMA
MCI	1, GPDMA
GPIO[2] Pins	70
10-bit ADC/DAC	1 jeweils
Real Time Clock	Separate Versorgung und Taktung
PWM	3 Phasen

Tabelle 2.1 Schnittstellen und Peripherie des LPC 2387 Mikrocontrollers

Es werden 4 Hardwaretimer zur Verfügung gestellt welche im Kapitel Hardwaretimer des LPC2387 wegen ihrer Bedeutung für diese Arbeit genauer beschrieben werden. Als weitere Peripherie ist noch der WatchDog Timer (WDT) zu nennen. Zu Debugzwecken besitzt der Mikrocontroller ein *JTAG* Interface. Er beherrscht drei verschiedene Stromsparmodi, in welchen verschiedene interne Peripherien sukzessive abgeschaltet werden. Die Reaktivierung des Mikrocontrollers kann über vier externe Eingänge oder die *GPIO* Pins an Anschluss Null oder Zwei erfolgen. Weiterhin können Unterbrechungsanforderungen der Echtzeituhr oder der *USB*- und Ethernetschnittstellen die Reaktivierung aus dem Stromsparmodus veranlassen. Jede Peripherie besitzt einen eigenen Teiler für den Mikrocontrollertakt und kann so anforderungsgerecht und stromsparend betrieben werden und bei Nicht-Benutzung einzeln deaktiviert werden. Die Nutzung einer *PLL*[3], statt der direkten Taktung über einen Oszillator, ermöglicht eine Veränderung der Taktfrequenz des Mikrocontrollers im Betrieb

1 Direct Memory Access: Die jeweileige Peripherie verfügt über einen DMA – Kontroller, sodass eine Datenübertragung in oder aus dem Arbeitsspeicher ohne Belastung des Mikrocontrollerkerns durch Interrupts erfolg.
2 General purpose input output: frei programmierbare digitale Ein- und Ausgänge des Mikrocontrollers
3 Phase-Locked Loop: Phasenregelschleife dient hier zur Frequenzsynthese

und könnte auch dazu eingesetzt werden den Mikrocontroller mit demselben Oszillator zu betreiben wie den Transceiver (siehe dazu Kapitel Fazit).

2.1.1.1 MCI Schnittstelle

Im Verlauf der unten erläuterten Experimente und Messungen wurden Daten gesammelt, deren Menge die Größe des Arbeitsspeichers übersteigt. Deshalb wurde es notwendig ein anders Medium zur Datenspeicherung zu wählen. Eine Möglichkeit wäre die Ausgabe der Daten über die UART – Schnittstelle und anschließende Speicherung auf einem leistungsfähigeren Computer gewesen. Dieser Ansatz wurde wegen fehlender Flexibilität, in Hinblick auf Messungen zwischen bis zu 500 Meter entfernten Sensorknoten, verworfen. Es wurde eine SD – Karte als Datensenke ausgewählt. Da die bestehende Implementierung sowohl des MCI – Treibers als auch die des Filesystemtreibers nicht effizient und stabil genug ausgeführt war, musste im Rahmen dieser Arbeit an dieser Stelle nachgebessert werden. Dazu wurde eine bestehende Implementierung[1] eines Fat - Filesystems und dazugehörigen MCI – Treibers an die verwendete Softwareplattform angepasst. Dabei wurde die Verwendung des GPDMA für die Schreib - und Leseoperationen implementiert und somit die Belastung des Mikrocontrollers mit Interrupts gesenkt und die Performanz gesteigert. In der vorherigen Implementierung wurde alle *64 Byte* eine Unterbrechungsanforderung ausgelöst, wohingegen nun über DMA bis zu 512 Bytes ohne Unterbrechung des Mikrocontrollers übertragen werden können.

2.1.1.2 Hardwaretimer des LPC2387

Die im LPC2387 vorhandenen vier 32 Bit Hardwaretimer können jeweils mit unterschiedlichen Teilern und Frequenzen, bis zur Taktfrequenz des Mikrocontrollers, betrieben werden und sind als Timer oder Counter konfigurierbar [NXP09]. Jeder Hardwaretimer besitzt zwei Aufnahmeanschlüsse (*capture*) zur Messung externer Ereignisse. Die Messung kann frei konfigurierbar, auf die steigende, fallende oder beide Flanken erfolgen. Weiterhin stehen für die Hardwaretimer mehrere Ausgänge zur Verfügung um Timerzustände extern zu signalisieren.

1 Chan`s FatFs siehe http://elm-chan.org/fsw/ff/00index_e.html (06.11.2010)

Die Hardwaretimer zählen Zyklen ihrer eigenen peripheren oder einer externen Taktquelle über einen einstellbaren Teiler. Es können auf Grundlage von vier verschiedenen definierbaren Timerwerten verschiedene Aktionen, wie zum Beispiel Unterbrechungsanforderungen an den Mikrocontroller oder Schaltung von externen Anschlüssen, ausgelöst werden. Dabei muss bei zeitkritischen Anwendungen beachtet werden, dass die Signalisierung des Timerzustandes über Unterbrechungsanforderungen, wie im Kapitel 2.1.1.3 beschrieben, Verzögerungen unterliegen kann. Die grundlegende Konfiguration der Auflösung der Hardwaretimer erfolgt über das entsprechende *PCKLSELx* – Register. Hier kann entschieden werden ob die periphere Uhr (*PCLK*) für den Hardwaretimer mit vollem, halben, dem viertel oder dem achtel Takt des Mikrocontrollers betrieben wird. Nachfolgend kann, mit Hilfe des *prescaler counter* – Registers des Hardwaretimers *(TxPC*), eine weitere Unterteilung der *PCLK* erfolgen. Dieses Register zählt bis zum im Register *TxPR* eingestellten Wert jeden Takt der *PCLK*. Bei Erreichen des Wertes im Register *TxPR* wird das Hardwaretimer-Register *TxTC* inkrementiert. Dabei ist zu beachten, dass das *TxTC* Register *32 Bit* breit ist und nach 2^{32} Hardwaretimer-Zyklen überläuft und wieder bei Null zu zählen beginnt. Dieser Überlauf kann über eine Unterbrechungsanforderung signalisiert werden und tritt in der, in dieser Arbeit verwendeten, höchsten Auflösung nach ca. *59,65 s* auf. Die maximale Auflösung der Hardwaretimer ist die volle Taktfrequenz des Mikrocontrollers und entspricht auf der verwendeten Plattform *13,88 ns* pro Takt. Der Aufnahmemodus des Hardwaretimers kann über das *capture control register* (*TxCCR*) eingestellt werden. Ändert im Aufnahmemodus der konfigurierte Anschluss seinen Zustand, so wird, ohne Belastung des Mikrocontrollers, der aktuelle Wert des *TxTC* Registers in das entsprechende *capture register* (*TxCRx*) übertragen. Das Speichern des Timerwertes kann über eine Unterbrechungsanforderung signalisiert werden. Es ist dem Umstand Rechnung zu tragen, dass das *TxCRx* Register bei jedem Ereignis überschrieben wird und deshalb entsprechend zeitnah ausgelesen werden muss. Für die Aufnahmeanschlüsse muss beachtet werden, dass der Hardwaretimer zwei steigende Flanken seiner peripheren Uhr benötigt um eine Zustandsänderung des anliegenden Signals zu erkennen, was eine Verringerung der zeitlichen Auflösung zur Folge hat.

2.1.1.3 Unterbrechungsanforderungen im LPC2387

Unterbrechungsanforderungen (engl. *interrupt request* kurz *IRQ*) dienen zur asynchronen Behandlung von externen, peripheren oder unerwarteten (zum Beispiel ein *data abort)* Ereignissen. Dazu wird der laufende Programmkontext bis zur Abarbeitung der sogenannten Interrup – Service - Routine (*ISR*), welche das Ereignis behandeln soll, pausiert. Sie dienen der effizienteren Programmierung von komplexen Programmen, welche viele verschiedene Ereignisse auswerten sollen. Es ist nicht mehr notwendig im Programm selbst die Zustandsänderungen der verschiedenen Anschlüsse zu verfolgen, dies erfolgt im Mikrocontroller über eine eigene Hardware den *vectored interrupt controller* (*VIC*). Der *VIC* des *LPC2387* verfügt über *32* Kanäle, welche priorisiert und mit zwei unterschiedlichen Typen von Unterbrechungsanforderungen behandelt werden können. Diese beiden Typen *IRQ* und *fast interrupt request* (*FIQ*) unterscheiden sich in der Art und Weise ihrer Abarbeitung. Alle *FIQ* besitzen die selbe höchste Priorität. Detektiert der *VIC* eine Zustandsänderung so wird die Ausführung des Programmes unterbrochen, der Zustand der Prozessorstatusregister gespeichert und zur definierten Adresse für die jeweilige *ISR* gesprungen. Die Adresse der *IRQ ISR* ist im Flash-Speicher des *LPC2387* an Adresse *0x00000018* zu hinterlegen und die der *FIQ ISR* an der darauffolgenden. Nach Ausführung des dort abgelegten *ISR* – Programmcodes werden die gespeicherten Statusregister zurückgeschrieben und der normale Programmablauf fortgesetzt. Der Unterschied zwischen *IRQ* und *FIQ* besteht in der Art des Sicherns der Programmstatusregister. Dabei wird die Abarbeitung eines *FIQ* durch die Verwendung von Schattenregistern, welche anstelle der Register für den normalen Programmablauf benutzt werden, beschleunigt. Somit spart man im Gegensatz zur Bearbeitung von *IRQ* teure Kopieroperationen und es vergehen weniger Taktzyklen bis zum Beginn der Abarbeitung der *ISR*. Da der *VIC* alle Kanäle logisch *ODER* – verknüpft, muss im Falle eines *FIQ* die Quelle der Unterbrechungsanforderung durch Auslesen des entsprechenden Registers (*VICFIQStatus*) ermittelt werden oder wenn es sich um einen *IRQ* handelt die Startadresse der *IRQ ISR* aus dem entsprechenden Register des *VIC* (*VICAddress*) ausgelesen werden. Weiterhin ist zur besseren Implementierbarkeit von Echtzeiteigenschaften, eine Unterbrechung einer *ISR* durch eine höher priorisierte Unterbrechungsanforderung in der *ARM7TDMI-S* – Architektur vorgesehen.

2.1.2 Der CC1100 Transceiver

Der *CC1100* ist ein stromsparender Transceiver, welcher unterhalb von *1 GHz* im *ISM*[1]- und im *SRD*[2]-Band betrieben werden kann [TEX09]. Es werden Datenübertragungsraten von bis zu *500 kBaud* unterstützt. Der analoge Teil des Transceivers beherrscht die Frequenzmodulationsverfahren *2-FSK*, *gaussian frequency shift keying (GFSK)* und *minimum shift keying (MSK)* sowie die Amplitudenmodulationsverfahren *on off keying (OOK)* und *amplitude shift keying (ASK)*. Um eine große Empfangsempfindlichkeit zu erreichen ist eine automatische Verstärkungskontrolle eingebaut (*AGC*). Mit der automatischen Trägerfrequenzkompensation(*FOC)* kann auf hochpräzise Oszillatoren verzichtet werden, was der Eignung des Transceivers für *WSNs* zugute kommt. Die *FOC* stellt dabei, in gewissen Grenzen die Trägerfrequenz des Empfängers auf die des ankommenden Funksignals ein. Die maximale Sendeleistung des Transceivers beträgt bis zu *10 mW* ohne externe Nachverstärkung des Signals, dabei liegt der Strombedarf, nach Herstellerangabe, bei ca. *31 mA*. Der *CC1100* erleichtert mit seiner komplexen digitalen Nachverarbeitung die Implementierung von paketorientierter Funkkommunikation für Sensorknoten. Die Merkmale reichen von Paketerkennungs- und Überprüfungsfunktionen, wie *CRC*-, Adress- und *sync word*[3]-Überprüfung, bis zur digitalen Bereitstellung von Funkqualitätsparametern. So wird die aktuelle Leistungsdichte in digitaler Form als *RSSI* zur Verfügung gestellt. Dieser und die Werte *preamble quality indicator(PQI)* und *link quality indicator(LQI)* können zur Bewertung der Empfangssituation, wie in Kapitel Digitale Signalverarbeitung beschrieben, ausgewertet werden. Der Transceiver unterstützt Stromsparmodi, in welchen ein Empfang von Funkpaketen immer noch möglich ist (*wake on radio*). Dazu wird der *CC1100* zyklisch, für ein kurzes Zeitintervall, aus dem Stromsparmodus in den Empfangsmodus versetzt. Die Kommunikation mit dem Mikrocontroller erfolgt, auf der in dieser Arbeit verwendeten Plattform, per *SPI* und über zwei der drei vorhandenen Interruptleitungen. Diese Interruptleitungen, die so-

1 *industrial, scientific and medical*: verschiedene Frequenzbereiche (siehe ITU-R 5138 und 5150)
2 *short range device*: Frequenzbereiche für Funkkommunikation über kurze Distanzen. In Europa 863 Mhz bis 870 Mhz
3 Das sogenannte *sync word* ist eine 16 Bit (auch 32 Bit durch Wiederholung) lange frei konfigurierbare Zeichenkette, welche automatisch vor den Paketdaten gesendet wird.

genannten *GDOx* Pins, können dynamisch zur Signalisierung verschiedener Betriebszustände des *CC1100* konfiguriert werden. Die in dieser Arbeit verwendete Konfiguration ist aus Tabelle 2.2 ersichtlich.

	GDO 0	*GDO 1*	*GDO 2*
Einstellungswert	*0x0E*	*0x2E*	*0x06*
Beschreibung	*carrier sense high* wenn *RSSI* über dem eingestellten Schwellwert.	Hochohmig (*3-state*) da geteilter Ausgang mit *SPI – slaveout*.	*high* wenn *sync word* erfolgreich empfangen oder gesendet wurde. *low* am Ende des Paketes (*RX* und *TX*) oder im Fehlerfall.
Verbundener Pin am *LPC2387* (*MSBA2*)	*P0.27*	*P1.23*	*P0.28* und *P0.23 (CAP3.0)*
Verbundener Pin am *LPC2387* (*AVSEXTREM*)	*P2.6*	*P0.8*	*P0.28* und *P0.23 (CAP3.0)*

Tabelle 2.2 Konfiguration der GDO x Pins

Der *CC1100* Transceiver besitzt getrennte Empfangs- und Sende-Puffer mit einer Größe von *64 Byte*, welche durch den unterstützten Bursttransfer[1] schnell gelesen oder gefüllt werden können. Weiterhin sind schnelle Wechsel der Trägerfrequenz möglich, was zum Beispiel die Implementierung von Frequenzsprungverfahren ermöglicht, aber auch in Kombination mit dem RSSI zum schnellen Auffinden eines freien Funkkanals benutzt werden kann. Um die im Kapitel Analyse und Evaluierung gezogenen Schlüsse nachvollziehbar zu machen, werden im Folgenden die analoge und digitale Signalverarbeitung näher betrachtet und die Ermittlung der Werte *RSSI*, *LQI* und *PQI* erklärt.

2.1.2.1 Analoge Signalverarbeitung

In dem in Abbildung 2.2 gezeigten Blockschaltbild ist im oberen Teil der Empfangsteil, ein *low IF* – Receiver, und im unteren der Sendeteil des *CC1100* abgebildet. Die in diesem Kapitel beschrieben analogen Schaltungsteile des Transceivers sind in der Abbildung durch Weiß gefüllte Blöcke gekennzeichnet. Der Empfangsteil beginnt

1 In diesem Übertragungsmodus ist es möglich den gesamten Inhalt des Sende- oder Empfangs-Puffers mit einem einzigen SPI – Kommando auszulesen.

bei den von der Antenne kommenden Anschlüssen *RF_P* (positiv) und *RF_N* (negativ) an welchen das empfangene Hochfrequenzsignal (*RF*: *radio frequency*) in den Transceiver gelangt. Bei diesem ankommenden Signal handelt es sich eher um ein Signalgemisch, aus welchem nun erst durch Filterung und Verstärkung das gewünschte Signal extrahiert werden muss. Dazu wird das noch sehr schwache Hochfrequenzsignal über eine Verstärkerstufe (*LNA*: *low noise amplifier*) geführt, welche durch rauscharme Verstärkung einerseits und ihr Bandpassverhalten anderseits, den Pegel der gewünschten Anteile um die Trägerfrequenz herum anhebt.

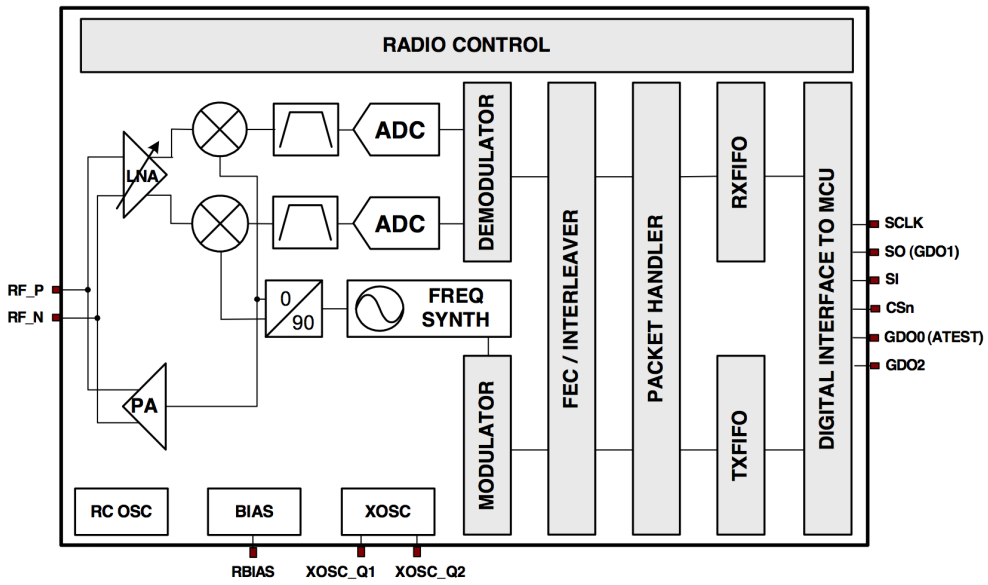

Abbildung 2.2 CC1100 Transceiver vereinfachtes Blockschaltbild [TEX09]

Nach dem *LNA* wird das verstärkte noch immer hochfrequente Signal in zwei identische Teile geteilt und zwei Mischern zugeführt. In diesen wird das Signal mit einem lokal im Frequenzsynthesizer erzeugten Signal bzw. seiner *90 Grad* gedrehte Kopie, jeweils zu einer deutlich niedrigeren sogenannten Zwischenfrequenz (*intermediate frequency, IF*, Faktor *1000* niedriger) gemischt. Die entstandenen Signale werden Inphase-(*I*) und Quadratursignal(*Q*, aus dem mischen mit dem *90 Grad* verschobenen Signal) genannt und lassen sich als Real- und Imaginärteil des komplexen Basisbandsignals[1] interpretieren. Die Umwandlung in die Zwischenfrequenz ermöglicht nachfolgend die Benutzung fest auf eine Frequenz abgestimmter Verstärker und Filter um

1 Basisband ist der Frequenzbereich in welchem das reine Nutzsignal definiert ist. Es wird für Frequenzmodulation durch die Bitfrequenz und den Modulationsindex, also den Abstand der Symbolfrequenzen, bestimmt.

eine hohe Selektivität zu erhalten. Selektivität ist dabei die Fähigkeit, das gewünschte Signal von Signalen, welche im Spektrum benachbart liegen, zu trennen [LAS04]. Der Signalweg für das Inphase- und das Quadratursignal verläuft über einen aktiven Bandpass, welcher für alle möglichen Trägerfrequenzen der Kanäle durchlässig ist. Danach folgt ein nicht dargestellten Verstärker(*variable gain amplifier, VGA*) für die automatische bzw. manuelle Verstärkungsanpassung (*automatic gain control, AGC*). Die *AGC* ist ein Regelkreis, welcher den Verstärkungsfaktor des *VGA* im analogen Teil über den, in der nach dem Analog–Digital–Wandler(*ADC*) folgenden digitalen Signalverarbeitung, ermittelten *RSSI* einstellt. So ist der Dynamikbereich des Empfängers zu vergrößern, denn die nach dem *VGA* folgenden Module, wie der *ADC* sind auf einen festen Pegelbereich eingestellt und würden bei zu hohen Pegel in die Sättigung getrieben bzw. bei zu niedrigem Pegel zu schlechte Werte für die nachfolgende digitale Demodulation liefern. Einen ähnlichen Regelkreis zwischen analogen und digitalen Teil bildet die Trägerfrequenzkalibrierung (*frequency offset calibration, FOC*), und dient der Anpassung der Empfangsteils auf die Trägerfrequenz des empfangenen Signals, welche sich durch Oszillatorungenauigkeiten des Senders ergeben. Dazu wird die im digitalen Teil des Empfängers ermittelte Bitfrequenz mit dem Frequenzsynthesizer abgeglichen. Die Digitalisierung des Signals kann nun, unter Einhaltung des Abtasttheorems, im niedrigen *MHz* - Bereich erfolgen. Die weitere Signalverarbeitung erfolgt digital. Der Sendeteil ist von Texas Instruments nicht genau beschrieben, funktioniert aber im Prinzip in der nachfolgend Dargestellten Weise. Nachdem das zu sendende Signal mit Hilfe eines digitalen Modulators im *IF* - Band erzeugt wurde, wird es mit einem Digital-Analog-Wandler in ein analoges Signal umgewandelt. Eventuell enthaltene unerwünschte Frequenzen, welche im *DAC* entstanden sind, werden über Tiefpassfilter weitestgehend eliminiert. Danach erfolgt die Mischung des Nutzsignals mit der Trägerfrequenz, ähnlich dem beschriebenen Verfahren in Empfangsrichtung. Das entstandene hochfrequente Signal kann nun über den, wiederum per Register steuerbaren, Leistungsverstärker (*power amplifier, PA*) über die an *RF_P* und *RF_N* verbundene Antenne, dem Übertragungsmedium zugeführt werden.

2.1.2.2 Digitale Signalverarbeitung

Die Module der digitalen Signalverarbeitung sind in Abbildung 2.2 durch die hellgrau hinterlegten Blöcke definiert. In Abbildung 2.3 wird das Paketformat des

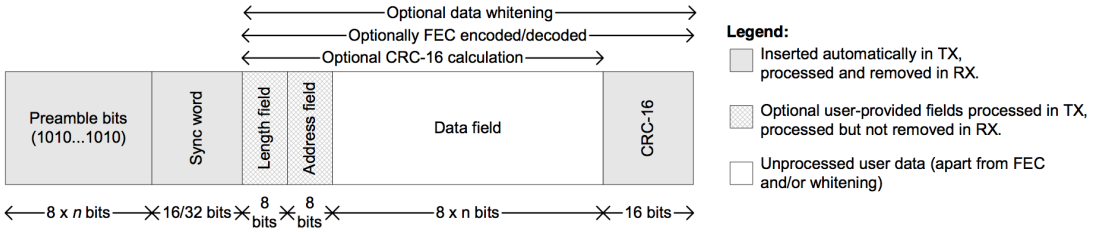

Abbildung 2.3 Paketformat des CC1100 [TEX09]

CC1100 dargestellt, dabei ist ersichtlich, dass die Menge der Daten, welche ein Paket enthalten kann, nicht festgelegt ist. Die digitale Signalverarbeitung erfolgt nun nach [TEX09] auf nachfolgend erläuterte Weise.

Der Transceiver unterstützt feste und variable Paketgrößen bis hin zum kontinuierlichen Datenstrom. Dabei ist bei fester Paketgröße eine Filterung der Pakete bezüglich ihrer Größe einstellbar. Der Filtermechanismus verwirft zu kleine Pakete. Bei zu großen Paketen wird der Empfangsmodus verlassen, wenn die eingestellte Anzahl an Bits empfangen wurde. Bei variabler Paketgröße wird direkt vor den Daten ein *length field* - Byte mit der Anzahl der Datenbytes gesendet, welches der Empfänger zuerst empfängt und damit den Paketlängenfilter initialisiert.

Eine Datenübertragung startet mit der Präambel *(preamble)*, einer Folge von Nullen und Einsen, welche den Empfänger auf eine kommende Datenübertragung vorbereiten soll. Die Präambel wird, wie in Abbildung 2.2 dargestellt, im Empfangsweg nach der analogen Signalverarbeitung im Empfänger vom digitalen *Demodulator* verarbeitet. Im Demodulator erfolgt die Kanalfilterung, FOC, AGC und die Ermittlung des RSSI wie folgt. Da die Präambel aus ständig wechselnden Bits besteht, ist es möglich anhand dieses Signals die Länge eines einzelnen Bits und somit die Bit-Frequenz zu ermitteln, wenn die erwartete Datenrate bekannt ist. Daraus lässt sich die Abweichung von der Trägerfrequenz des empfangenen Signals ableiten und die FOC implementieren, welche über den Frequenzsynthesizer diese Abweichung kompensieren kann. In ähnlicher Weise ist die AGC implementiert, mit dem Unterschied,

dass nicht die Frequenz sondern die max Amplitude der einzelnen Bits als Grundlage für den Regelkreis genutzt wird. Ein weiterer Wert, der *PQI*, wird auf Basis des Wechsels der Bits der Präambel wie folgt erzeugt. Immer wenn sich das momentane Bit vom vorherigen unterscheidet, wird der *PQI* inkrementiert und im andern Fall wird 8 vom PQI abgezogen. In der Konfiguration des Transceivers wird ein Schwellwert (*preamble quality threshold*, *PQT*) für den *PQI* festgelegt. Bei erreichen dieses Schwellwertes beginnt die Byte Synchronisation auf Grundlage des nach der *Präambel* übertragenen *sync word*(siehe Abbildung 2.3). Dieses *sync word* ist eine vom Benutzer frei wählbare zwei Byte lange Bitfolge, anhand derer die Byte – Grenzen im Datenstrom erkannt werden. Die Übertragung des *sync word* kann wiederholt werden um mit einer vier Byte langen Bitfolge die Erkennung zu verbessern. Nun können die nachfolgenden Bytes des Paketes bearbeitet werden, welche optional per Whitening[1] möglichst gleichspannungsfrei und per *interleaving*[2] und *forward error correction* (*FEC*) störunanfälliger gemacht werden können. Diese Daten, welchen eine *CRC16* Checksumme angefügt werden kann, werden nun in der Paketverarbeitung (*packet handler,* siehe Abbildung 2.2) ausgewertet. Dabei wird wie oben beschrieben die Länge des Paketes behandelt, sowie die Auswertung des *adressbytes* vorgenommen. Ob Pakete weitergeleitet werden oder anhand des *adressbytes* ausgefiltert werden ist konfigurierbar. Die Filterung erfolgt anhand der im Transceiver eingestellten Adresse und einer konfigurierbaren *broadcast* – Adresse. Nach der Paketverarbeitungsstufe werden die Daten, das *addressbyte* und das Längenbyte im *RX* – Puffer zur Verfügung gestellt und können über *SPI* ausgelesen werden. Zusätzlich ist es möglich, den ermittelten *RSSI* und den *link quality index*(*LQI*) an die Daten anfügen zu lassen. Der *LQI* wird aus den Zeichen welche nach dem *sync word* folgen generiert und ist ein Indikator dafür wie schwierig die Demodulation der Zeichen im Vergleich zur idealen Konstellation war. Dabei repräsentiert ein hoher Wert des *LQI* eine gute Verbindungsqualität.

Der Sendeweg ist äquivalent zum Empfangsweg, wobei die Werte *RSSI, LQI, PQI* in diesem Fall nicht bestimmt werden. Es werden die zuvor erläuterten Stufen in umgekehrter Reihenfolge durchlaufen und deshalb nicht im einzelnen nochmals erklärt.

1 Die Daten werden mit einer Pseudorandom (PN9) Bitfolge verodert.
2 Datenvermischung um die FEC zu unterstützen, welche vereinzelte Bitfehler korrigieren kann.

2.2 Softwareplattform

Die Programmierung erfolgt für weite Teile dieser Arbeit in der Programmiersprache *C*. Für zeitkritische Abläufe, wie die Routinen für Unterbrechungsanforderungen, wurde auf Assembler zurückgegriffen. Als Compiler ist der quelloffene *gnu c compiler* (gcc)[1] eingesetzt worden. Dieser ist Teil der, vom *FeuerWhere*-Betriebssystemkern[2] (Kernel) vorgegebenen, Entwicklungsumgebung. Es war notwendig die gesamte Entwicklungsumgebung für das MacOS X Betriebssystem anzupassen und den *arm-elf-gcc* Crosscompiler neu zu übersetzten. Die Programmierung der einzelnen Sensorknoten erfolgt per *UART0* Schnittstelle, welche über den *USB*/Seriell Umsetzer Chip (*FT232RL*) der Firma *FTDI* Ltd.[3] per *USB* an den Entwicklungsrechner angeschlossen werden konnte. Des weiteren ist im Verlauf der Arbeit die *JTAG* – Schnittstelle[4] in Betrieb genommen worden. Grundlage dafür war die Übersetzung des Debugprogramms *OpenOCD*[5] für *MacOS X*.

2.2.1 Das FeuerWhere Betriebssystem

Der, im Rahmen des *FeuerWhere* Projektes, an der Feien Universität Berlin entwickelte Kernel basiert unter anderem auf den Diplomarbeiten von Heiko Will [WIL08] und Kasper Schleiser und ist die Grundlage des in dieser Arbeit eingesetzten Betriebssystems. Hierbei handelt es sich um einen preemptiven Mikrokernel, welcher speziell für den Einsatz in drahtlosen Sensornetzen entwickelt wurde. Dabei sind die wesentlichen Merkmale, welche einen stromsparenden Betrieb der Hardware ermöglichen, einerseits die sogenannten *tickless timer* und andererseits die preemptive Ablaufsteuerung (s*cheduling*) von Prozessen. Die *tickless timer* Architektur ermöglicht, durch die Nutzung mehrerer Hardwaretimer des *LPC2387* Mikrocontrollers, eine

1 Vgl. http://www.gnu.org/software/gcc/
2 Vgl. API Feuerwhere kernel http://cst.mi.fu-berlin.de/projects/scatterweb/documentation/feuerware/trunk/api/main.html
3 Vgl. http://www.ftdichip.com/
4 Joint Test Action Group, IEEE1142.1 spezifiziert neben Verfahren zum Testen und Debuggen elektronische Hardware auch Schnittstellen.
5 Vgl. http://openocd.berlios.de/web/

strom- und rechenzeiteffiziente Implementierung von Timerfunktionalität, ohne aktiv zyklisch das Ablaufen von Timern überprüfen zu müssen. Die preemptive Prozesssteuerung ermöglicht es, quasi parallel Programmteile in Prozessen ablaufen zu lassen, ohne dass es nötig ist während der Laufzeit eines Prozesses, den Status von anderen Prozessen zu überprüfen. Bei diesem Schedulingverfahren, welches Rechenzeit einspart und Unterbrechungen wichtiger Prozesse verhindert, läuft immer der Prozess mit der höchsten Priorität ab. Dieser Prozess kann nur durch eine Unterbrechungsanforderung der Hardware oder einen noch höherrangigen Prozess unterbrochen werden. Prozesse niedrigerer Priorität werden erst gestartet wenn der höherrangige pausiert und die Rechenzeit abgibt. Damit ist es möglich Echtzeiteigenschaften zu implementieren. Zum Kernel gehören zusätzlich Teile welche Interprozesskommunikation ermöglichen und Zugriffssperren (*mutex*) implementieren.

Ein weiterer Teil des Betriebsystems ist die quelloffene *newlib* - Bibliothek[1], die in Verbindung mit der *gnu mpfr* Bibliothek[2], mathematische Funktionen unterstützt welche Fließkommaberechnungen ermöglichen. Weiterhin unterstützt die *newlib* – Bibliothek Operationen für Zeichenketten und allgemeine Ein- und Ausgabefunktionen.

Neben dem schon im Kapitel 2.1.1.1 erwähnten *MCI*-Treiber ist der *CC1100*-Treiber, welcher von Thomas Hillebrandt im Rahmen seiner Diplomarbeit [HIL07] entwickelt wurde, für diese Diplomarbeit von Bedeutung. So konnten die ersten Versuche auf Basis dieses Treibers stattfinden, wobei sich im späteren Verlauf zeigte, dass nur die Hardwarenahen Anteile den Anforderungen des Laufzeitmesssystems genügten. Es wurden daher die Schnittstelle zur *SPI* – Kommunikation und die Registerdefinitionen in den Headerdateien, für die im Rahmen dieser Diplomarbeit entstandene Software wiederverwendet.

2.2.1.1 Unterbrechungsanforderungen im Kernel

Vom Betriebssystemkern werden nur die im Kapitel 2.1.1.3 erläuterten *IRQ* behandelt. Die Behandlung von *FIQs* wurde im Rahmen der experimentellen Analyse der

1 Vgl. http://sourceware.org/newlib/
2 Eine Bibliothek für Fließkommaberechnungen mit korrekter Rundung. Vgl. http://www.mpfr.org/

Interrupt – Latenzen nachimplementiert (Vgl. 3.1.1). Die Routinen zur Behandlung von Unterbrechungsanforderungen sind in Assembler programmiert und erfordern im Falle der *IRQ* – Routine eine aufwendige Unterstützung der Prozessablaufsteuerung. Das beinhaltet einen rechenaufwendigen Kontextwechsel, wobei unter anderem alle Prozessorstatusregister, vor dem Eintritt in die *IRQ* - Routine, im dem dem aktuell aktiven Prozess zugeordneten Stack gespeichert werden müssen [WIL08]. Danach kann aus dem *VIC* die entsprechende, dem *IRQ* Kanal zugeordnete, Routine aufgerufen werden. Diese Routinen sind in *C* programmiert, wobei im Falle eines *GPIO* – Interrupts eine weitere Suche der dem Port zugeordneten Routine erfolgt. Durch diese *ISR* – Implementierung vergeht eine Zeit von *10 µs* bis zur ersten Instruktion der eigentlichen Interrupt – Routine. Da am Ende der *IRQ* – Routine nach dem Prozess mit der höchsten Priorität gesucht wird und dieser mit dem Zurückschreiben der Prozessorstatusregister aus dessen Stack aktiviert wird, vergehen weitere *10 µs* bis die erste Instruktion des Prozesses zur Ausführung kommt. Diese Wartezeit kann nicht von anderen Interrupts unterbrochen werden, da *nested interrupts* im Kernel nicht unterstützt werden. Um eine konstante Latenz für Unterbrechungsanforderungen zu garantieren, welche für Laufzeitmessungen unbedingt notwendig ist, wäre es notwendig *nested interrupts* zu implementieren. Darauf wurde, aufgrund der Ergebnisse der Analyse der Interrupt – Latenz, im Rahmen dieser Diplomarbeit verzichtet. Unterbrechungsanforderungen werden hier nur für laufzeitunkritische Kommunikation mit dem *CC1100* – Transceiver verwendet.

2.3 Messhilfsmittel

Um die Messwerte, welche mit den Sensorknoten ermittelt wurden, zu überprüfen wurden digitale Oszilloskope und ein Logikanalysator benutzt. Die Messung von Signalen im zweistelligen Megahertzbereich stellt erhöhte Anforderungen an die Messhilfsmittel. Deshalb wurde es im Verlauf der Arbeit notwendig bestimmte Messungen in der Bundesanstalt für Materialforschung und -prüfung (*BAM*)[1] durchzuführen.

1 BAM, Ansprechpartner Enrico Köppe, Unter den Eichen 87, 12205 Berlin, enrico.koeppe@bam.de

Vergleichsmessungen mit Messhilfsmitteln unterliegen gewissen Beschränkungen, sodass nicht alle im realen Umfeld durchgeführten Messungen überprüft werden können. Eine Beschränkung ist der Abstand der Sensorknoten, welcher durch die Länge der Messleitungen des jeweiligen Messgerätes beschränkt ist. Es hätte die Möglichkeit bestanden, Dämpfungsglieder zur Simulation verschiedener Sensorknotenabstände einzusetzen, jedoch sind diese nicht für jeden beliebigen Abstand vorhanden. Weiterhin lässt die Simulation aller Abstände und somit die Überprüfung aller Messungen keinen großen Zugewinn an Information über das Verhalten des Laufzeitmesssystems erwarten und wurde deshalb nicht durchgeführt.

Alle Vergleichsmessungen mit Messhilfsmitteln wurden mit einem Sensorabstand von einem Meter durchgeführt. Die Ausgangsverstärkung des CC1100 wurde niedrig gewählt, um ein Übersteuern des Empfangsteils des Empfängersensorknotens zu verhindern und einen sicheren Empfang zu gewährleisten. Um die Messleitungen sicher anschließen zu können, wurden an den entsprechenden Pins kurze Litzen angelötet. Folgende Leitungen am *CC1100* [TEX09] und am *LPC2387* [NXP08] wurden jeweils am Sender- und Empfängersensorknoten gemessen :

- CC1100 GDO0 Pin: signalisiert in der benutzten Konfiguration den *air free* Zustand
- CC1100 GDO2 Pin: signalisiert in der benutzen Konfiguration das Senden oder Empfangen eines Paketes
- LPC2387 Pin P0.0: dient zur Überprüfung der Interruptverzögerung / Jitter
- LPC2387 Pin P0.23: Aufzeichnungsport des Hardwaretimers

2.3.1 Digitale Oszillographen

Für alle Messungen wurde das *S4622A* der Firma Agilent, ein digitales Zweikanaloszilloskop mit einer minimalen zeitlichen Auflösung von *10 ns*. Die maximale Abtastrate beträgt *200 MS*[1] pro Sekunde. Dieses Oszilloskop wurde zur Überprüfung der Messwerte der Sensorknoten und zur Analyse der Interruptverzögerung eingesetzt.

1 MS: MegaSample 1MS entspricht 1 Millionen Messwerten

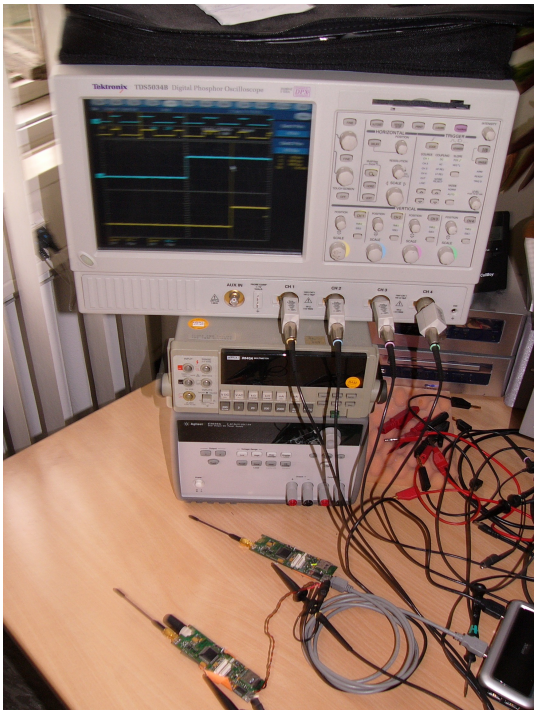

Abbildung 2.4 Messungen am Sensorknoten mit dem Tektronix Oszillographen

Die maximale Abtastrate des *S4622A* lässt keine zweifache Überabtastung der vom *LPC2387* erzeugbaren Signale zu. Somit ist, mit diesem Gerät, entsprechend dem Nyquist – Shannon Abtasttheorem [SHA49], keine genaue und fehlerfreie Messung solch hochfrequenter Signale möglich. Deshalb wurden die Messwerte, welche mit den *S4622A* ermittelt wurden, mit dem *TDS5034B* der Firma *Tektronix* überprüft. Dieses in Abbildung 2.4 gezeigte Vierkanaloszilloskop ist Eigentum der *BAM* und stand nur kurze Zeit zur Verfügung. Es ist in der Lage mit einer maximalen Abtastrate von *5 GS*[1] pro Sekunde aufzuzeichnen und kann Zustandsänderungen mit einer zeitlichen Auflösung von bis zu *200 ps* erfassen, und ist damit hinreichend genau für alle auftretenden Messungen.

2.3.2 Logikanalysator

Um längere Intervalle, wie das Senden mehrerer hintereinander folgender Funkpakete messen zu können, wurde ein Logikanalysator der Firma Intronix eingesetzt. Das Gerät verfügt über 32 Kanäle und besitzt eine minimale zeitliche Auflösung von *4 ns*.

1 GS: GigaSample 1GS entspricht 1 Milliarde Messwerte

Durch die relativ hohe zeitliche Auflösung des Gerätes, konnten auch mit diesem Gerät die mit dem Oszilloskop der Firma Agilent ermittelten Messwerte überprüft werden.

2.3.3 Funkspektrumanalysator „*mspect*“

Um während der Messungen, welche im Rahmen dieser Diplomarbeit durchgeführt wurden, den benutzten Frequenzbereich überwachen zu können, wurde eine Sensorknotensoftware entwickelt, die es erlaubt direkt auf den Sensorknoten des Laufzeitmesssystems die momentane Auslastung des Mediums zu beurteilen. Die Implementierung erfolgte auf der Grundlage einer *design note* der Firma *Texas Instruments* [TEX10]. Es können alle Trägerfrequenzen, welche der *CC1100* Transceiver unterstützt, überwacht werden. Dazu wird für jeden Kanal über ein bestimmtes Zeitintervall der *RSSI* Wert gemittelt und ausgegeben. Da der Transceiver schmalbandig ausgelegt ist und eine hohe Umschaltzeit zwischen den Kanälen für die Kalibrierung benötigt wird, sollte die Anzahl der überwachten Kanäle gering gehalten werden. Diese Software steht im *FeuerWhere – SVN Repository*[1] unter dem Projektnamen *mspect* zur Verfügung.

2.4 Technische Begrenzungen

Abschließend, für das Kapitel 2 , wird aufgrund der zuvor beschriebenen Hardware- und Software-Eigenschaften der benutzten Plattform, die theoretisch erreichbare Messgenauigkeit erörtert.

Die Messgenauigkeit bezieht sich, im Rahmen dieser Arbeit, hauptsächlich auf die zeitliche Abweichung bei der Detektion einer steigenden oder fallenden Flanke eines Rechtecksignals.

1 Repository location: https://svn.mi.fu-berlin.de/msb

Die größte Abweichung tritt durch die unterschiedlich lange zeitliche Latenz bei der Detektion eines Signals auf. Die Detektion erfolgt auf Grundlage des Pegels, welcher an einem Eingang des *LPC2387* anliegt. Um die steigende oder fallende Flanke zu detektieren, wird der Pegel am Eingang gemessen. Danach wird dieser mit dem Pegel aus dem vorherigen Takt verglichen. Es ist aber nicht möglich den, Zeitpunkt der Pegeländerung, innerhalb eines Taktes zu bestimmen.

Wenn Signalquelle und -Senke zeitsynchron betrieben werden ist der Zeitpunkt der Pegeländerung immer konstant, da ähnlich wie bei der Detektion des Pegelwechsels auch die Generierung dessen fest an den Taktzyklus gebunden ist und deshalb immer zum gleichen festen Zeitpunkt geschieht. Dadurch entsteht keine Abweichung bei der Detektion des Pegelwechsels. Werden Signalempfänger und -Sender asynchron betrieben, ist der Zeitpunkt des Pegelwechsels zufällig verteilt. Daher beträgt die Abweichung, bei der Detektion des Pegelwechsels, maximal eine Taktlänge des Signalempfängers.

2.4.1 Messabweichungen des CC1100

Die Präzision mit welcher der Zeitpunkt bestimmt werden kann, wann ein Funksignal eingetroffen ist, hängt von der Frequenz ab mit welcher das Funksignal abgetastet wird. Da, wie in 2.1.2.2 dargestellt, die gesamte Signalverarbeitung auf Bitebene im digitalem Teil des *CC1100* Transceivers erfolgt, wird für die folgende Betrachtung die Taktfrequenz des Transceivers, also *26 MHz* angenommen. Da Sende- und Empfänger-Transceiver asynchron arbeiten, beträgt die maximale Abweichung *38,46 ns*. Das entspricht bei einer Abstandsmessung nach Gleichung (1.4) einem Fehler von *11,53 m*. Die Abweichung des Oszillators, welche maximal *40 ppm*[1] betragen darf[TEX09], ist so gering, dass sie für diese Arbeit nicht ins Gewicht fällt, da die gesamte Messung weniger als drei Millisekunden in Anspruch nimmt. In dieser Arbeit erfolgt die Abstandsmessung nach der *RToF* - Methode (1.1.1.2), deshalb tritt der oben genannte der Fehler zweimal auf, und beträgt rund *23 m*. Das Zeitintervall

1 parts per million: 1ppm = 0,001%

welches von eintreffen des Signals an der Antenne bis zur Weiterverarbeitung im digitalen Teil des Receivers definiert ist, auch *propagation delay*, wird durch die Schaltzeiten der einzelnen Komponenten im Analogteil bestimmt. Dieses Intervall ist nicht dokumentiert, fügt aber den Messwerten der Abstandsbestimmungen ein Offset hinzu, welches bei der Auswertung berücksichtigt werden muss. Für Senderichtung muss das *propagation delay* in gleicher Weise beachtet werden. Da es, im Rahmen dieser Diplomarbeit keine Möglichkeit gibt, dieses durch direkte Messungen zu bestimmen, wurde es experimentell ermittelt.

Anzumerken ist hierbei, dass die Abtastrate des Analog-Digital-Wandlers des CC1100 für die Bestimmung der Messauflösung zu Grunde gelegt werden müsste. Diese ist, wie das *propagation* delay, nicht bekannt und konnte auch nicht durch Nachfragen beim Hersteller „Texas Instruments“ ermittelt werden.

2.4.2 Messabweichungen des LPC2387

Der *LPC2387* Mikrocontroller ist mit *72 MHz* getaktet und besitzt wie der *CC1100* Transceiver einen eigenen Oszillator. Da der Mikrocontroller die asynchron auftretenden Signale detektiert ist hier eine Abweichung von *13,89 ns* zu berücksichtigen. Dies entspricht bei der Abstandsmessung einer Abweichung von rund *4,16 m*. Wie im vorherigen Kapitel begründet, bleibt die Abweichung des Oszillators unberücksichtigt.

Diese Abweichung kann sich, durch die Verzögerung von Unterbrechungsanforderungen, welche im Kapitel 2.1.1.3 beschrieben wurden, noch vergrößern. Wie in Kaptitel Analyse der Messgenauigkeit dargestellt, konnte experimentell ermittelt werden, dass diese Abweichung groß ist und Schwankungen unterliegt. Deshalb wurde eine andere Methode zur Zeitmessung eingesetzt, welche die Abweichung nicht vergrößert.

Bei der Abstandsbestimmung mit der *RToF* - Methode wird vier mal der Zeitpunkt des Auftretens eines Signals vom Transceiver mit dem Mikrocontroller bestimmt

(siehe Kapitel 3). Dabei ist jedes mal die Abweichung des Mikrocontrollers bei der Zeitpunktbestimmung zu beachten. Im schlechtesten Fall addieren sich diese Abweichungen mit den Abweichungen des Transceivers. Daraus ergibt sich, im schlechtesten Fall, eine Messabweichung von fast $40\,m$. Das eine Messung mit diesem System sinnvoll ist, basiert auf der Annahme, dass sich durch stochastische Verfahren, diese Abweichung reduzieren lässt, da alle Abweichungen zufällig und gleich verteilt auftreten können.

3 Messungen

Aufgrund der in den vorherigen Kapiteln gemachten theoretischen Überlegungen wurden die aufgestellten Thesen experimentell untersucht. Es galt zu überprüfen, ob die durch die Plattform vorgegebenen Beschränkungen überwunden werden können. Kernthese ist es, dass sich durch wiederholte Messungen an ein und dem selben Ort die Abweichungen, welche unter anderem durch die niedrige Taktfrequenz der benutzten Hardware auftreten, durch stochastische Methoden ausgleichen lassen. Dazu ist es notwendig mehrere Messungen, bestehend aus einem Anforderungspaket und einem Antwortpaket, durchzuführen ohne dabei mögliche andere Funkkommunikation zu stören. Es ist notwendig ein geeignetes Medienzugriffsverfahren zu implementieren. Es soll für jede Messung das Zeitintervall, vom Senden des Anforderungspakets bis zum Eintreffen des Antwortpakets, bestimmt werden. Dafür war es notwendig, möglichst noch im *CC1100* – Transceiver festzustellen, ob ein Funkpaket gesendet wurde oder ob ein Paket eingetroffen ist. Dazu besteht, wie im Kapitel *Der CC1100 Transceiver* beschrieben, die Möglichkeit über einen Ausgangspin des Transceivers den Paketstatus zu signalisieren. Im Sendemodus signalisiert eine steigende Flanke an diesem Ausgang den erfolgreichen Versand des s*ync words* und eine fallende Flanke signalisiert den erfolgreichen Versand des gesamten Pakets. Im Empfangsmodus signalisiert eine steigende Flanke den Empfang und die erfolgreiche Dekodierung eines *sync words*, welches mit dem im Transceiver programmierten übereinstimmt. Das heißt, es konnte der Beginn eines Pakets empfangen werden, welches zu hoher Wahrscheinlichkeit, an diesen Sensorknoten adressiert ist. Dieses Signal ist die Basis der Laufzeitmessung und wird sowohl beim anfordernden als auch beim antwortenden Sensorknoten wie folgt ausgewertet.

Nachdem der Mikrocontroller des anfordernden Sensorknotens die Paketdaten zum Senden an den Transceiver übertragen hat, wartet der Mikrocontroller auf die steigende Flanke um den Beginn der Zeitmessung (*ToF, time of flight*) einzuleiten.

Danach wird der Transceiver in den Empfangsmodus umgeschaltet und der Mikrocontroller wartet wiederum auf eine steigende Flanke, welche den Empfang eines richtigen *sync words* signalisiert. Bei Detektion dieser wird die Zeitmessung gestoppt. Um die Bearbeitungszeit (*TCP*, *time to compute packet*) im antwortenden Sensorknoten zu bestimmen wird auf ähnliche Weise verfahren. Hier beginnt die Zeitmessung beim Empfang des *sync words* des Anforderungspaketes und endet beim Versenden des *sync words* der Antwort. Da nun das Antwortpaket nicht mehr modifiziert werden kann, erfolgt die Übertragung der Bearbeitungszeit im nächsten Antwortpaket. Weil somit für das letzte Paket einer Messreihe kein Wert für die Bearbeitungszeit im antwortenden Knoten zur Verfügung steht, muss dieses verworfen werden. Dieser Messzyklus ist in Abbildung 3.1 noch einmal veranschaulicht.

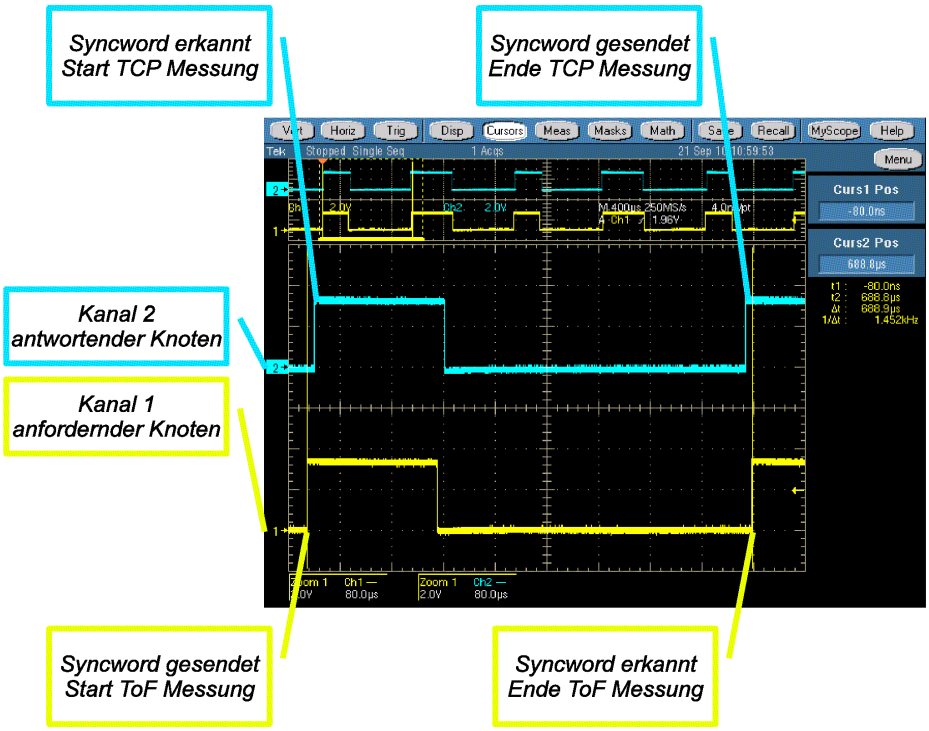

Abbildung 3.1 Zeigt das oszillographierte Bild des Signalverlaufs am GDO2 Pin des anfordernden und antwortenden Sensorknoten während einer Messung. Im oberen Bereich ist am Signalverlauf zu erkennen, dass kontinuierlich Pakete versendet werden. Weiterhin ist anhand der Länge des jeweiligen High-Pegels der Pakettyp zu identifizieren. Das Antwortpaket enthält mehr Nutzdaten, somit liegt im Signalverlauf länger ein High-Pegel an. Deshalb wurde beim oszillographieren auf die Pulsweite des Anforderungspakets getriggert.

Es ist zu beachten, dass der erfolgreiche Empfang des s*ync words* keine ausreichende Sicherheit über den Inhalt des nachfolgendem Pakets gibt. Deshalb prüft der antwor-

tende Knoten nach vollständigem Empfang des Paketes ,ob es sich um ein an ihn adressiertes gültiges Anforderungspaket handelt. Anderenfalls erfolgt keine Antwort und der Sensorknoten schaltet wieder in den Empfangsmodus um. Stellt der anfordernde Knoten fest, dass es sich bei dem empfangenen Paket nicht um das erwartete Antwortpaket, erkennbar an einer eindeutigen Sequenznummer und der Absenderadresse handelt, wird die Zeitmessung wiederaufgenommen. Kann innerhalb eines bestimmten Zeitintervalls kein gültiges Antwortpaket empfangen werden, startet der anfordernde Knoten die Messung mit der Wiederholung des Anforderungspakets neu. Ist ein gültiges Antwortpaket eingegangen werden die Messwerte gespeichert und ein neuer Messzyklus beginnt mit dem Senden eines Anforderungspaketes mit inkrementierter Sequenznummer.

Um eine geeignete Methode zur Erfassung der Start- und Endzeitpunkte, der *ToF* und *TCP* Messungen zu ermitteln, folgt im nächsten Kapitel eine Analyse der Messgenauigkeit der verschiedenen Signaldetektionsmethoden der *LPC2387* Mikrocontrollers. Darauf folgend wird die für die Messungen verwendete Konfiguration des *CC1100* Transceivers beschrieben und das verwendete Paketformat erläutert. Danach folgen die Beschreibung der Experimente zur Abstandsbestimmung und Lokalisierung.

3.1 Analyse der Messgenauigkeit

Da aus der Dokumentation des *LPC2387* [NXP09] nicht hervorgeht ob das Zeitintervall der Verzögerung von Unterbrechungsanforderungen immer eine konstante Größe hat, ist es notwendig dieses mithilfe von Messungen zu analysieren. Ist das Zeitintervall nicht konstant, so spricht man von Jitter, der die Größe der Schwankungen des Zeitintervalls bezeichnet. Dazu werden im folgenden drei Messungen beschrieben, welche den Jitter für *IRQ*, *FIQ* und die Genauigkeit des Hardwaretimers bestimmen.

3.1.1 Messung der Interrupt-Jitters

Zur Messung des Jitters ist eine, asynchron zum Mikrocontroller-Takt betriebene, Signalquelle notwendig. Diese ist mit dem *CC1100* gegeben, da dieser einen eigenen Taktgeber besitzt. Zur Erzeugung eines Signals wird zyklisch das selbe Paket versendet. Wie im Kapitel 2.1.2 beschrieben, signalisiert der Transceiver am *GDO2*-Pin den Status des Paketversands (*IOCFG2* Register auf Wert *0x06* [TEX09]), welcher mit Pin *P0.28* des *LPC2387* verbunden ist. Der Signalverlauf am *GDO2*-Pin wird mit dem *LPC2387* verfolgt und löst bei einer steigenden Flanke des Signals einen Interrupt aus. Um das Zeitintervall, welches vom Eintreten der Interrupt-Bedingung (steigende Flanke) bis zum Ausführen der ersten Instruktionen der *ISR* definiert sei, messen zu können, wird in den ersten Instruktionen der *ISR* der *Pin P0.11* des Mikrocontrollers auf High-Potenzial gesetzt und danach zurückgesetzt. Dazu folgendes Codefragment der *ISR* für *FIQ*:

```
fk_cpu_fiq_isr:

/* set gpiopin 0.11 */
ldr     r0,=0x3FFFC018 /* FIO0SET */
ldr     r1,=0x00000800 /* BIT11 */
str     r1, [r0,#0x00] /* write to adr from r0 value from r1*/

/* clr gpiopin 0.11 */
ldr     r0,=0x3FFFC01C /* FIO0CLR */
ldr     r1,=0x00000800 /* BIT11 */
str     r1, [r0,#0x00] /* write to adr from r0 value from r1 */

/* clear all ints*/
ldr     r0,=0xE002808C /* Interrupt Clear GPIO 0 */
ldr     r1,=0xFFFFFFFF
str     r1, [r0,#0x00]

ldr     r0,=0xE00280AC /* Interrupt Clear GPIO 2 */
ldr     r1,=0xFFFFFFFF
str     r1, [r0,#0x00]

mov     r0,=0xFFFFF01C /* VIC interrupt clear */
str     r1, [r0,#0x00]

/* jump back to task */
subs pc, lr, #4;
```

Die Messung erfolgt mit dem oben vorgestellten Oszilloskop der Firma Agilent, wobei Kanal eins des Oszilloskopes an den *GDO2*-Pin und Kanal zwei an den Pin *P0.11* angeschlossen wird. Als Trigger-Bedingung ist eine steigende Flanke auf Kanal eins gesetzt. Mit dieser Konfiguration wurden jeweils einzeln *IRQ* und *FIQ* oszillographiert, wobei während der Messungen, keine anderen Interrupt-Quellen aktiv waren. Um eine bessere oszillographische Darstellung zu erhalten, wurde das Display nicht nach jeder Einzelmessung gelöscht. Das in Abbildung 3.2 gezeigte oszillographierte Bild stellt den Jitter des *FIQ* dar. Es ist ein zufällig verteilter Jitter (steigende Flanken zwischen Cursor *X1* und *X2*) von *90 ns* zu verzeichnen, dies entspricht *6,5* Mikrocontroller-Takten. Dazu kommt ein periodischer Jitter von *230 ns*, welcher durch die seltener auftretenden aber scharf abgegrenzten Flanken (ca. *670 ns* nach Trigger) zu erkennen ist. Für den *IRQ* sind ähnliche Werte ablesbar, wie in Abbildung 3.3 zu erkennen ist.

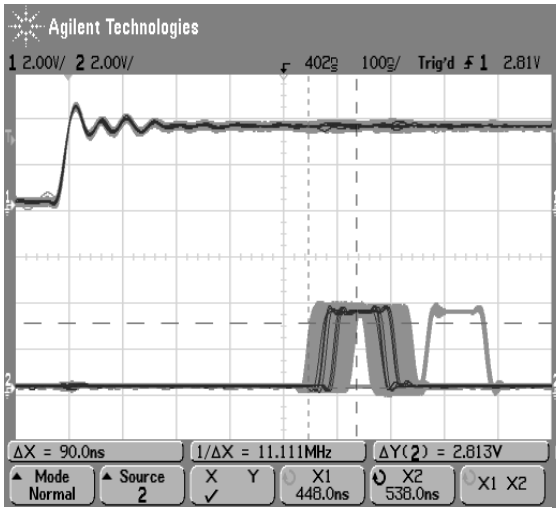

Abbildung 3.2 Messung Jitter FIQ

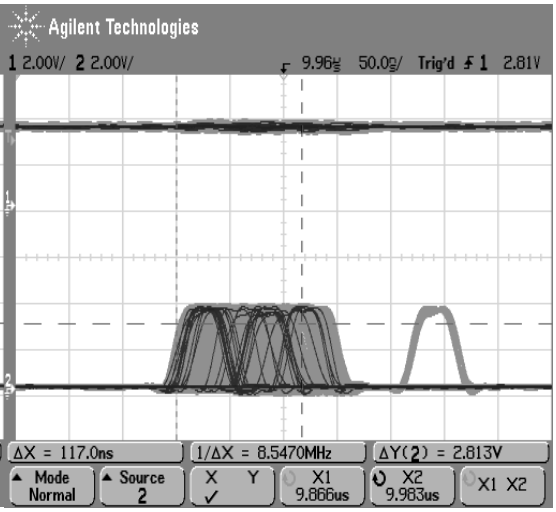

Abbildung 3.3 Messung Jitter IRQ

Es ist weiterhin abzulesen, dass das Zeitintervall in welchem ein *high* - Potenzial am Pin *P0.11* anliegt nur halb so groß ist, wie im Falle des *FIQ* (die Impulse in Abbildung 3.3 sind kürzer). Diese Differenz ist damit zu begründen, dass die *IRQ ISR* in *C* und nicht wie die *FIQ ISR* in Assembler geschrieben ist. Der *C*–Kompilierer hat an dieser Stelle das Laden der Adressen und der Werte in die Register vorgezogen, und führt das Setzen und das Zurücksetzten des Pin *P0.11* direkt hintereinander aus. Die *FIQ ISR* führt nach dem Setzen des Pins *P0.11* zwei Instruktionen zum Laden der Register aus, bevor Pin *P0.11* zurückgesetzt wird. Über die Ursache der Ähnlichkeit des Jitters von *FIQ* und *IRQ* kann man nur Vermutungen anstellen. Eine Möglichkeit

wäre es, dass der Jitter nicht aus dem *VIC* resultiert, sondern vielmehr eine Folge des Zusammenschalten aller interruptfähigen *GPIO*[1] Anschlüsse des *LPC2387* auf einen *VIC*-Kanal (*EXTINT3*) ist. Es ist, mit der in dieser Arbeit zur Verfügung stehenden Hardwareplattform nicht möglich die anderen drei nicht geteilten externen Interrupt-Leitung(*EXTINT0-2*) zu analysieren, da diese vom SD-Karten-Interface belegt sind und nicht nach außen geführt werden. Somit lässt sich die aufgestellte These nicht untermauern.

Zusammenfassend ist festzustellen, dass eine Abstandsmessung mit Interrupts auf dem *LPC2387* keine hohe Genauigkeit erwarten lässt, da der ermittelte Jitter eine Abweichung von *27 m* zur Folge hat, wenn man nur den zufällig verteilten Jitter betrachtet. Betrachtet man auch den periodisch auftretenden Jitter, so erhöht sich die Abweichung auf fast *70 m*. Es muss eine andere Möglichkeit gefunden werden, Zeitmessungen mit einem asynchronen externen Signal möglichst präzise steuern zu können. Dazu wird im nachfolgenden Kapitel die Genauigkeit der Aufnahmefunktion des Hardwaretimers untersucht.

3.1.2 Genauigkeit des Hardwaretimers

Um den Signalverlauf des *GDO2* Pin des *CC1100* Transceivers mit dem Aufnahmeanschluss des *LPC2387* Mikrocontrollers verfolgen zu können, wurde auf der *MSBA2*-Plattform eine Hardwareänderung notwendig, welche in das Design der *AVSExtrem* - Plattform übernommen worden ist. Diese Änderung verbindet zusätzlich den *GDO2* - Pin des *CC1100* mit dem ersten Kanal des dritten Hardwaretimers (*CAP3.0*) an Pin *P0.23* des Mikrocontrollers [TEX08a].

Um die Genauigkeit der Aufnahmefunktion des Hardwaretimers zu ermitteln wird, wie im vorherigen Kapitel, ein externes asynchrones Signal benötigt, welches möglichst jitterfrei vorliegt und wiederholt erzeugt werden kann. Dazu wird der *GDO2*-Pin des Transceivers so konfiguriert, dass an diesem ein, vom Oszillator des

1 General Purpose Input Output: digitale Ein oder Ausgänge zur beliebigen Verwendung. Es können digitale Signale bis zur halben Taktfrequenz des Mikrocontrollers ausgegeben und aufgenommen werden.

CC1100 abgeleitetes, Rechtecksignal anliegt (*IOCFG2* Register auf Wert *0x39* [TEX09]). Die Frequenz des Signals beträgt *1,08 MHz*, es stehen damit *66* Mikrocontrollertaktzyklen zum Speichern der einzelnen Messwerte zur Verfügung. Der Hardwaretimer zählt, wie im folgenden Codefragment ersichtlich, mit jeden Takt des Mikrocontrollers und schreibt seinen Wert bei einer detektierten steigenden Flanke in das Capture-Register (*T3CR0*):

```
// configure timer 3 for capture on CAP 3.0

// CCLK to PCLK no divider
PCLKSEL1 = (PCLKSEL1 & ~(BIT14 | BIT15)) | (1 << 14);
PCONP |= PCTIM3;          // power on

PINSEL1 |= BIT14 + BIT15; // configure PIN 0.23 as CAP3.0
T3CCR = 0x1;              // capture on rising edge
//T3CCR = 0x2;            // capture on falling edge
//T3CCR = 0x3;            // capture on both edge

T3TCR = 1;                // counter reset
T3MCR = 0;                // disable interrupt
T3EMR = 0;                // no external match output.
T3PR = 0;                 // no prescaler
T3TC = 0;                 // reset counter
T3IR = 0;                 // ack old interrupts

T3MR0 = 0;                // clear match registers
T3MR1 = 0;
T3MR2 = 0;
T3MR3 = 0;
```

Um die einzelnen Messwerte zu speichern wird im Hauptprogramm der Status des Pin *P0.28* zyklisch abgefragt. Nachdem eine steigende Flanke detektiert werden konnte, wird der Wert des Registers *T3CR0* als Startwert in einem Array gespeichert. Auf dem selben Weg wird der Stop-Wert, bei der nächsten steigenden Flanke, ermittelt und gespeichert. Am Ende der Messung wird die Liste aller Messwerte über *UART0* ausgegeben und mit *matlab* weiterverarbeitet.

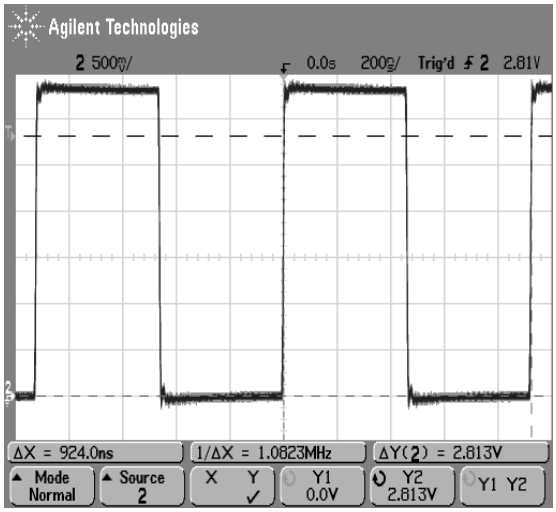

Abbildung 3.4 Signal am GDO2 Pin des CC1100 bei Konfiguration CCLK/24

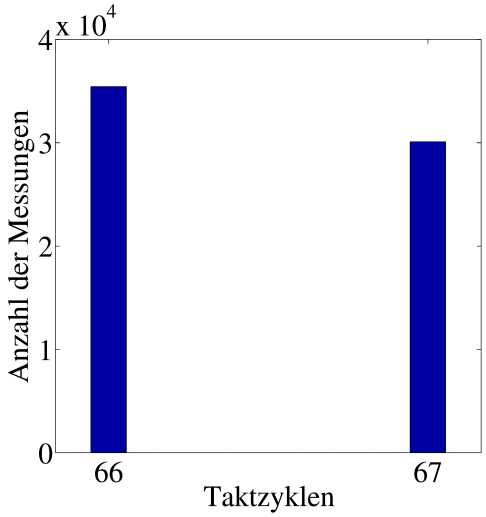

Abbildung 3.5 Histogramm Jitter des Hardwaretimers im Aufnahmemodus

Abbildung 3.4 Zeigt das ursprüngliche Signal am *GDO2* – Pin der Transceivers. Im in Abbildung 3.5 dargestellten Histogramm ist zu erkennen, dass annähernd gleich verteilt Werte zwischen *66* und *67* Takten über *65536* Messwerte gemessen wurden. Das entspricht *1,09 MHz* bzw. *1,07 MHz*. Dieser Jitter über einen Mikrocontrollertakt ist, wie in Kapitel 2.4.2 erläutert, durch die zwei asynchron zueinander laufenden Taktgeber zu erklären.

In einer weiteren Messung wurde das Paketstatussignal am *GDO2*-Pin des *CC1100* Transceivers untersucht, welches einen periodischen Jitter von *26 MHz* aufweist. Die Abbildung 3.6 zeigt im oberen Bereich das vollständige Signal und darunter vergrößert den Jitter der fallenden Flanke. Getriggert wurde bei der oszillographischen Messung auf die steigende Flanke des Signals. Um dieses Signal mit dem *LPC2387* messen zu können wurde, zusätzlich zur steigenden Flanke, der Hardwaretimerwert zur fallenden Flanke gespeichert (Register *T3CCR = 0x3*). Das Vorgehen ist äquivalent zur ersten Messung, wobei *1000* Messwerte aufgezeichnet wurden.

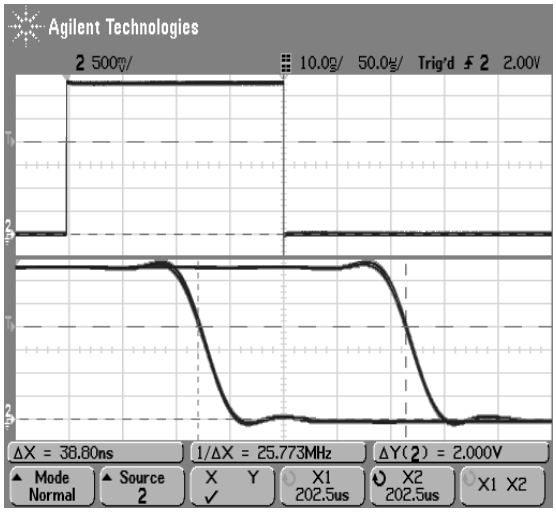

Abbildung 3.6 Jitter GDO2 des CC1100 Signal beim Paketversand

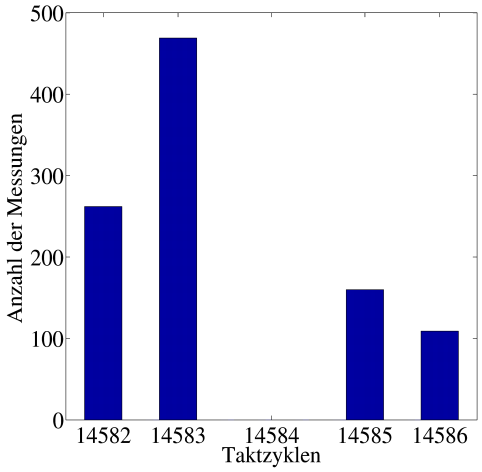

Abbildung 3.7 Histogramm Jitter Hardwaretimer Aufnahmemodus: Paket senden

Im Histogramm der Messwerte des Hardwaretimers(Abbildung 3.7) entspricht der kleinste Messwert *202,542 µs* und der größte Werte *202,583 µs*. Es sind beide fallenden Flanken, dargestellt im unteren Bereich in Abbildung 3.6, aus den Messwerten zu erkennen, wobei der Abstand der Flanken in den Messwerten zwei Taktzyklen beträgt. Das entspricht einer Jitterfrequenz von *24 Mhz* und weicht damit weniger als die bekannte Messtoleranz von einem Takt bzw. *13,9 ns* vom Oszillographen-Messwert ab.

Zusammenfassend konnte, anhand der Messungen, die Aufnahmefunktion des Hardwaretimers als die genaueste, vom Mikrocontroller unterstützte, Messmethode ermittelt werden. Es ist bei dieser Methode eine Abweichung von *13,9 ns* zu beachten, das entspricht bei der Abstandsmessungen einer Abweichung von rund vier Metern. Deshalb wird im folgenden für alle Messungen die Aufnahmefunktion des Hardwaretimers der *LPC2387* benutzt.

3.1.3 Analyse der Transceiver Konfiguration

Die folgenden Messungen sollen den Einfluss der Konfigurationen des Transceivers, auf den Jitter der Laufzeitmessung ermitteln. Dazu werden verschiedene Modulationsverfahren mit den in Tabelle 3.1 aufgelisteten Konfigurationen experimentell un-

tersucht. Diese Messungen wurden im Innenhof des Instituts für Informatik durchgeführt, wobei zwei Knoten in einem Abstand von *10* Metern aufgestellt worden sind. Es wurden nacheinander *500* Laufzeitmessungen mit der *RToF* - Methode unter Verwendung der verschiedenen Konfigurationen durchgeführt, ohne die Position der Knoten zu verändern.

Messung	Modulationsverfahren	*FOC_LIMIT*	*AGC_FREEZE*
1	MSK 400 kbps	0x01, FOC an	0x00: normal Operation
2	MSK 400 kbps	0x00: FOC an	0x01: freeze after *sync word*
3	MSK 400 kbps	0x01: FOC aus	0x00: normal Operation
4	MSK 400 kbps	0x00: FOC aus	0x01: freeze after *sync word*
5	OOK 250 kbps	-	0x00: normal Operation
6	OOK 250 kbps	-	0x01: freeze after *sync word*

Tabelle 3.1 Liste der CC1100 Konfiguration aller Messungen zur Bestimmung des Jitters von Laufzeitmessungen.

Als Modulationsverfahren wurden neben dem *minimum shift keying (MSK)* auch das für den Transceiver einfacher zu demodulierende *on off keying (OOK)* untersucht. Die Regelkreise welche Einfuss auf die analoge Signalverarbeitung im Transceiver haben und somit direkt das Zeitverhalten der Biterkennung beeinflussen könnten, sind die *automatic gain control* (*AGC*) und die *frequency offset calibration* (*FOC*) (siehe 2.1.2). Das Verhalten dieser beiden Regelkreise wird über die Registerwerte *AGCCTRL0* Feld *AGC_FREEZE* und *FOCCFG* Feld *FOC_LIMIT* im *CC1100* Transceiver gesteuert [TEX09]. Weiterhin besteht die Möglichkeit den Transceiver automatisch auf die Datenrate (Bitfrequenz) des empfangenen Signals zu kalibrieren (Register *BSCFG 0x1A*). In Vorversuchen wurde festgestellt, dass mit vollständig deaktivierter *AGC* (*AGC_FREEZE 0x10 / 0x11*) und auch bei eingeschalteter Bitsynchronisation (*BS_LIMIT 0x01 bis 0x03*), eine Kommunikation der Sensorknoten kaum möglich war. Deshalb wurden diese Konfigurationen nicht untersucht. Für das *OOK* steht keine Trägerfrequenzkalibrierung zur Verfügung und wurde in den Messungen somit für *OOK* nicht betrachtet. Die gesamte Konfiguration für *OOK* und für *MSK* ist im Anhang A verfügbar. Es wurde bei diesen Messungen eine modifizierte Version

der im Kapitel 3.2 vorgestellten Implementierung der Laufzeitmessung benutzt. Die Modifikationen beschränken sich auf eine dynamisch zur Laufzeit veränderbare Konfiguration des Transceivers.

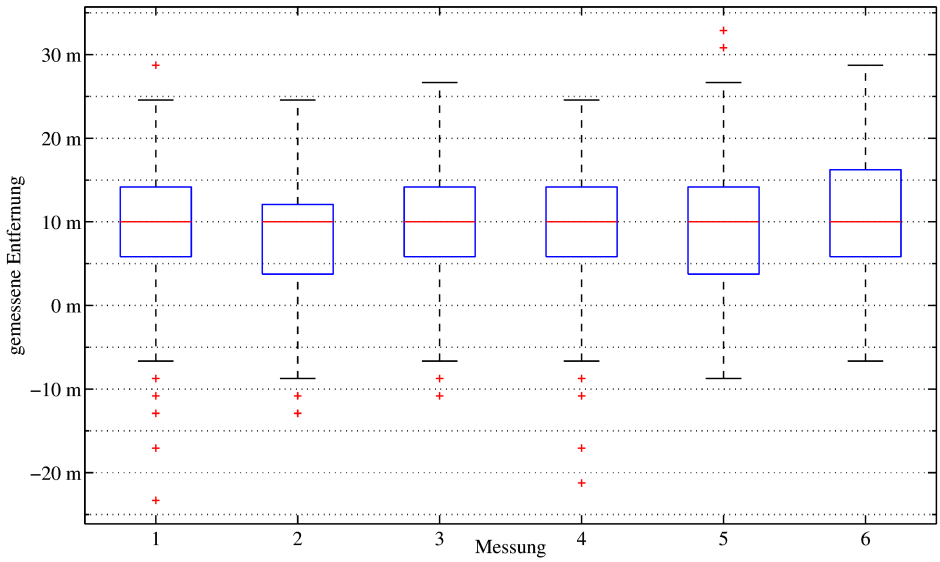

Abbildung 3.8 Boxplot der Ergebnisse der Konfigurationsmessungen 1 bis 6.

Der Boxplot in Abbildung 3.8 zeigt den Jitter der Laufzeitmessungen, wobei auf der X – Achse die Messungen anhand ihrer Nummerierung (siehe Tabelle 3.1) aufgetragen sind. Die *whisker* Länge beträgt 1,5 und es wurden 500 Messwerte betrachtet. Da der Jitter verglichen werden soll, wurden die Messwerte auf der Basis ihres Medians zur besseren Vergleichbarkeit Offset bereinigt, sodass der Median bei allen Messungen bei *10 m* liegt. Es ist nur ein marginaler Unterschied des Interquartilsabstandes zwischen den Messwerten von den Messungen mit *MSK* und *OOK* zu verzeichnen. Deshalb kann man davon ausgehen, dass die untersuchten Parameter hier messbaren Einfluss auf die Abweichung der Laufzeitmessungen haben. Deshalb wird die im bestehenden *FeuerWhere* - Hardwaretreiber benutzte Konfiguration eingesetzt, wodurch im realen System der Transceiver nicht, vor und nach jeder Laufzeitmessung, umkonfiguriert werden muss.

3.2 Implementierung

Die Implementierung der Laufzeitmessung erfolgte für die folgenden Experimente in minimalistischer Weise, um Unterbrechungen der Laufzeitmessungen, durch andere Prozesse des Kernels oder durch Hardwaretreiber, möglichst zu vermeiden. Es wurde die nicht benötigte Hardware abgeschaltet und auch die im Mikrocontroller integrierte unbenutzte Peripherie wurde deaktiviert. Die Senderichtung und Datenaufzeichnung des anfordernden Sensorknotens wird im Hauptprozess implementiert, wobei während einer Messung Kontextwechsel und Interrupts unterbunden wurden. Der einzige Interrupt, welcher aktiviert bleibt, ist die Unterbrechungsanforderung vom *CC1100* Transceiver, welche durch den Paketempfang ausgelöst wird. Dieser Interrupt wird durch eine *ISR* behandelt, welche den Paketempfang und die Beendigung der Laufzeitmessung implementiert. Die Implementierung des antwortenden Sensorknoten erfolgt in der *ISR*, des Interrupts, welchen der *CC1100* für die Paketstatussignalisierung verwendet. In der Tabelle 3.2 sind alle verwendeten Kernelmodule und Hardwaretreiber und eine Beschreibung ihrer Verwendung im Laufzeitmesssystem aufgelistet.

Modul / Hardwaretreiber	Beschreibung
ktimer	Laufzeitmessung / Wartezustände
gpio	Interruptverarbeitung der digitalen E/A
realtime clock	Zeitstempel Datenaufzeichnung
cc1100 spi Treiber	Zugriff auf den Transceiver
mci / fat Treiber	Datenaufzeichnung
newlib	Standard E/A, *math* Bibliothek

Tabelle 3.2 Liste der für das Laufzeitmesssystem verwendeten Kernelmodule und Hardwaretreiber

Beide Sensorknotentypen initialisieren zu Beginn die verwendeten Kernelmodule und Hardwaretreiber und laden die entsprechende Konfiguration (Anhang A *MSK 400 kbps*), in den *CC1100* Transceiver. Dabei wurde der oben genannte *SPI* Treiber für den *CC1100* verwendet, welcher das Lesen und Schreiben von Registern imple-

mentiert. Danach erfolgt die Initialisierung des Hardwaretimers, wie im Kapitel 3.1.2 beschrieben wurde, zur Detektion des am *GDO2* – Pins anliegenden Paketstatussignals. Der weitere Programmablauf unterscheidet sich für den anfordernden und antwortenden Sensorknoten, deshalb wird dessen Implementierung in den folgenden Kapiteln beschrieben.

3.2.1 Implementierung des anfordernden Sensorknoten

Der anfordernde Sensorknoten tritt nach der Initialisierungsphase in die Hauptschleife des Programms ein, in welcher das Versenden der Anforderungspakete und die Datenaufzeichnung implementiert wurde. Der Programmablauf wird anhand des folgenden Zustandsautomaten erläutert.

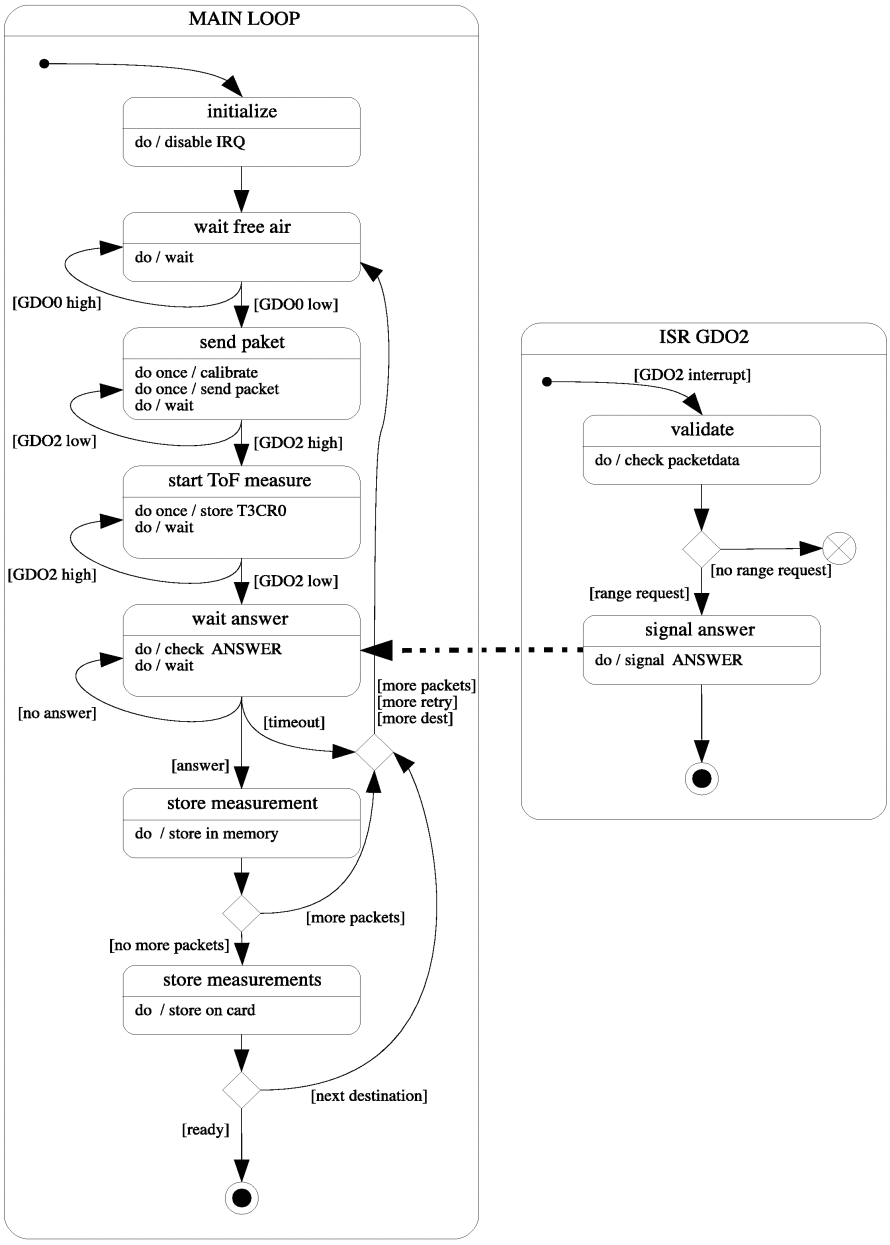

Abbildung 3.9 Zustandsdiagramm anfordernder Knoten

Vor dem Beginn der Senderoutine wird der *GDO2* - Interrupt im Mikrocontroller deaktiviert, da die auch für die Senderichtung der Paketstatus vom Transceiver signalisiert wird, und dann die *ISR* für den Paketempfang aktiviert werden würde. Dadurch würde der Programmablauf für mindestens *40 µs* unnötig unterbrochen. Im ersten Zustand wird vom Transceiver das Medium auf andere Kommunikation überprüft. Ist der Übertragungskanal frei so folgt direkt im Anschluss das Senden des Anforderungspaketes, ohne das die im Medienzugriffsverfahren des *MicroMesh* – Stacks vor-

geschriebene *backoff* – Zeit eingehalten wird. Dadurch wird die Übertragung der Pakete des Laufzeitmesssystems bevorzugt erfolgen. Danach wird der Transceiver in den *IDLE* - Zustand versetzt, was im Falle einer notwendigen Neukalibrierung der *PLL* bis zu 800 Mikrosekunden in Anspruch nehmen kann. Die Einstellungen zur Neukalibibrierung der *PLL* sind so gewählt worden, dass der Übergang des Transceiver vom Sende- in den Empfangsmodus, schnell möglich ist. Dazu wurde die automatische Neukalibrierung der *PLL* im Register *MCSM0 0x18* mit dem Wert *FS_AUTOCAL = 0x10* auf den Übergang vom Sendemodus in den Leerlauf eingestellt. Nachdem die Paketdaten, welche im Kapitel 3.2.3 spezifiziert werden, an den Transceiver übertragen wurden, wird dieser in den Sendemodus geschaltet. Nach rund *90 µs* beginnt der Transceiver mit der Versendung der Paketdaten und signalisiert, mit einer steigenden Flanke am *GDO2* – Pin, wann das *sync word* gesendet worden ist. Diese Flanke wird vom Hardwaretimer detektiert und daraufhin der aktuelle Wert des Hardwaretimers in das Register *T3CR0* gespeichert. Da im Programmablauf ebenfalls auf eine *high* - Pegel am *GDO2* – Pin gewartet wird, kann der Zeitstempel für den Start der Laufzeitmessung, direkt nach dem Schreiben des Registers *T3CR0* ausgelesen und im Arbeitsspeicher abgelegt werden. Wird der Erfolgreiche Versand des Pakets signalisiert, so wird der Interrupt des *GDO2* – Pins aktiviert und auf das Antwortpaket gewartet. Im Fehlerfall wird das Senden wiederholt. Der *CC1100* Transceiver wurde im Register *MCSM1* mit dem Wert *TXOFF_MODE = 0x11* so konfiguriert, dass dieser automatisch in den Empfangsmodus versetzt wird sobald das Anforderungspaket versendet wurde. Trifft das Antwortpaket nicht innerhalb eines bestimmten Zeitintervalls ein, so wird davon ausgegangen dass entweder die Anforderung oder die Antwort verloren gegangen ist. Daraufhin wird das Anforderungspaket wiederholt, wobei generell nach einer beschränkten Anzahl von Wiederholungen die Messung zu diesem Ziel aufgegeben wird. Trifft ein Paket ein, so erfolgt eine Signalisierung durch eine steigende Flanke am *GDO2* – Pin, ausgelöst durch die erfolgreiche Demodulation des letzten Bits des *sync word*. Der Hardwaretimer triggert auf diese Flanke und speichert den aktuellen Timerwert in das *T3CR0* Register. Der Paketempfang erfolgt in der *ISR* des *GDO2* – Pins, welche durch eine fallende Flanke (Paketende) an diesem Pin gestartet wird. In dieser wird, nach erfolgreicher Verifikation des Antwortpaketes, die Laufzeitmessung beendet. Die Messwerte welche im Paket enthalten sind und der Wert der des *T3CR0* Registers werden im Ar-

beitsspeicher abgelegt, und es erfolgt die Signalisierung des Empfangs des Antwortpaketes an die Hauptschleife, welche nach Beendigung der *ISR* fortgesetzt wird. Der Transceiver wechselt, aufgrund des Wertes *RXOFF_MODE = 0x00* im Register *MCSM1,* automatisch in den Leerlauf und führt dabei eine Kalibrierung der *PLL* durch. Damit ist die Messung beendet und es erfolgen, je nach Einstellung weitere Messungen zum Zielsensorknoten. Sind alle Messungen durchgeführt worden, so werden die gesammelten Messdaten auf der SD - Karte gespeichert, und es können die nächsten Messungen zum nächsten Knoten erfolgen.

3.2.2 Implementierung des antwortenden Sensorknoten

Die Implementierung des antwortenden Sensorknoten besteht nur aus der Interrupt Routine, welche durch eine fallende Flanke am *GDO2*-Pin aktiviert wird. Nach der Initialisierung versetzt das Hauptprogramm den Transceiver in den Empfangsmodus. Der Hardwaretimer wird in der selben Weise benutzt, wie in der oben beschriebenen Implementierung des anfordernden Sensorknotens. Nach erfolgreicher Überprüfung der Gültigkeit des Anforderungspakets, wird die enthaltene Sequenznummer sofort an den Absender zurückgesandt. Dabei werden die Werte *RSSI* und *LQI*, welche vom Transceiver für jedes empfangene Paket an dieses angehängt werden, und die letzte *TCP* als weitere Daten dem Antwortpaket angefügt. Die *TCP* wird aufgrund der gespeicherten Werte des Hardwaretimers bestimmt und im nächsten Antwortpaket übermittelt (Vgl. Abbildung 3.1).

3.2.3 Paketformat

Da die Laufzeitmessung im Umfeld des Instituts für Informatik an der Freien Universität Berlin durchgeführt wurden, war es notwendig das verwendete Paketformat an des dort vielfach verwendete Paketformat anzupassen. Die Anpassung erfolgt in so weit, dass Pakete welche vom Laufzeitmesssystem versendet werden, von anderen Sensorknoten erkannt werden und verworfen werden können. Auch muss das Laufzeitmesssystem Funkpakete welche von anderen Knoten gesendet werden erkennen und ausfiltern. Dazu wurde die ersten vier Byte - Felder des Paketformates des *Mi-*

croMesh[1] – Protokolls übernommen [WIL08]. Diese sind die vom Transceiver vorgegebenen Felder *length* und *address* und die im *MicroMesh* – Protokoll verwendeten Felder *sourceaddress* und *protocol*. Dies Felder werden gemäß der Vorgaben des *MicroMesh* Protokolls gefüllt, wobei für die Protokollnummer eine bislang nicht benutzte Nummer verwendet wurde, um wie oben beschrieben die Kommunikation aller den *MicroMesh* – Protokollstack benutzenden Sensorknoten nicht zu stören. Die folgende Abbildung zeigt das Paketformat welches für die Anforderungspakete verwendet wird.

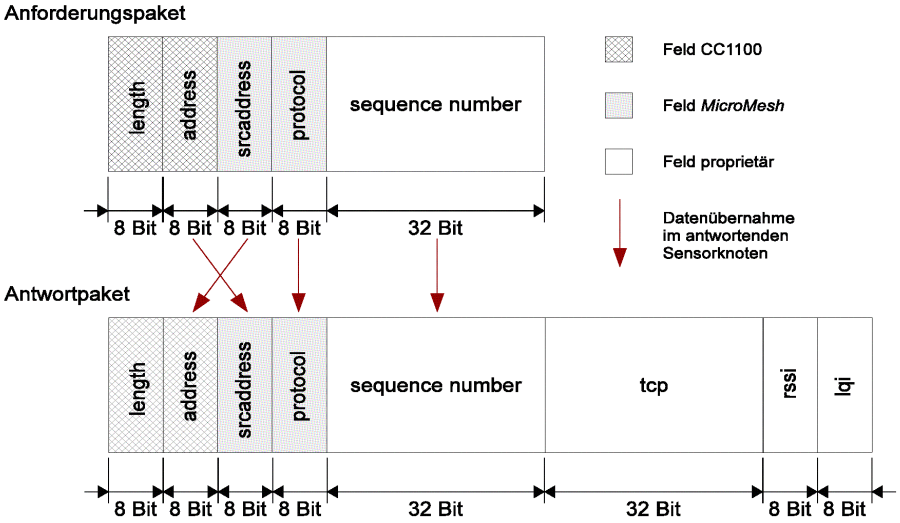

Abbildung 3.10 verwendete Paketformate für des Anforderungs- und das Antwortpaket der Laufzeitmessungen

Das *sequencenumber* – Feld wird als *32 – Bit unsigned Integer* interpretiert und dient der Verifikation des Antwortpaketes. Dazu sendet der antwortende Knoten die empfangene Sequenznummer im Antwortpaket zurück. Da der anfordernde Knoten nach dem Senden des Anforderungspaketes auf ein Antwortpaket mit der richtigen Sequenznummer und Absenderadresse wartet, ist die Zuordnung eindeutig. Die Größe des Feldes *sequencenumber* wird zur Protokollierung, auf dem sonst *stateless* arbeitenden antwortenden Sensorknoten, benötigt. Im realen Einsatz des Laufzeitmesssystems kann dieses Feld verkleinert werden. Für das Antwortpaket ist ein anderes Paketformat notwendig, da hier weitere Informationen zum anfordernden Sensorknoten übertragen werden müssen. Wie in Abbildung 3.10 gezeigt wird, zusätzlich zu den

1 *MicroMesh* ist ein Funkprotokollstack, der von Heiko Will und Thomas Hillebrandt an der Freien Universität Berlin entwickelt wurde.

Feldern des Anforderungspaketes, die Bearbeitungszeit, welche für das vorherige Paket benötigt wurde, im vier Byte großen Feld *tcp* übertragen (*unsigned integer*). Die weiteren ein Byte großen Felder *rssi* und *lqi* beinhalten die, vom Transceiver beim Empfangen des Anforderungspakets ermittelten Werte für den *RSSI* und den *LQI*. Diese Werte sollen vom anfordernden Sensorknoten gespeichert werden und müssen deshalb zu diesem übertragen werden.

3.3 Experimentelle Untersuchung

Zur Ermittlung der Abweichung, welche das im Rahmen dieser Diplomarbeit entstandene Laufzeitmesssystem bei der Abstandsmessung aufweist, wird dieses unter verschiedenen Bedingungen experimentell untersucht. Dazu werden im folgenden drei Versuchsmessungen beschrieben. Das erste Experiment dient der Untersuchung des Systems unter möglichst idealen Bedingungen. Dazu wurden Messungen im Freifeld durchgeführt. Ziel war es, neben der experimentellen Bestimmung der Abweichungen der Abstandsbestimmungen, das *propagation delay* im Transceiver unter realen Bedingungen zu bestimmen, welches von der *TCP* – Messungen nicht erfasst werden kann. Weiterhin wurden zwei Experimente im Innenraumbereich durchgeführt, wobei zuerst eine Messung mit direkter Sichtverbindung Aufschluss über das Verhalten des Systems innerhalb von Gebäuden geben und den darauf folgenden Lokalisierungsversuch vorbereiten soll. Der Lokalisierungsversuch dient der Untersuchung der Nutzbarkeit des Systems im inneren von Gebäuden.

3.3.1 Planung und Durchführung des Freifeldexperimentes

Der Messaufbau für das Freifeldexperiment wird auf Basis der, in Kapitel 1.1.4 beschriebenen, Grundvoraussetzungen für eine unbeeinflusste Ausbreitung der elektromagnetischen Wellen geplant. Ziel des Experimentes ist es die Abweichungen der Messwerte der Laufzeitmessung zu bestimmen. Weiterhin soll, dass in Kapitel 2.4.1 angesprochene, *propagation delay* des Transceivers analysiert werden. Um möglichst ideale Bedingungen für die Messungen garantieren zu können, wurde neben einer

Abbildung 3.11 Aufbau des Freifeldexperimentes nahe Güterfelde südlich von Berlin

Umgebung, welche wenig störende andere Funkkommunikation erwarten lässt, besonderes Augenmerk auf die Aufstellung der vier an der Messung beteiligten Sensorknoten gelegt. Da der Durchmesser der 1. fresnelsche Zone mit steigender Entfernung wächst, ist die maximale Messentfernung durch die Höhe der Sensorknoten über dem Erdboden festgelegt. In diesem Experiment wurden, wie in Abbildung 3.11 zu erkennen, über *4 m* hohe Masten benutzt. Der Abstand *d* zwischen den Sensorknoten, bei welchen man von Freiraumdämpfung ausgehen kann, kann über die, umgestellte Gleichung der 1. fresnelschen Zone (1.10), bestimmt werden.

$$d = \frac{(2 \cdot h_{MAST})}{\lambda}$$

$$\frac{(2 \cdot 4.2\,m)^2}{0{,}3454\,m} = \mathbf{204{,}29\,m} \qquad (3.1)$$

Die Höhe der Masten wird in h_{MAST} angegeben und λ ist die Wellenlänge der eingestellten Trägerfrequenz des Transceivers. Die Trägerfrequenz beträgt während dieser Messung *868 Mhz* somit beträgt die Wellenlänge λ rund *35 cm*. Es ergibt sich eine maximale Messentfernung von rund *200* Metern. Damit die Messungen nicht durch Funkkommunikation, anderer in der Umgebung befindlicher Geräte gestört werden

konnte, wurde ein abgemähtes Feld südlich von Berlin als Ort für die Messungen ausgewählt. Um sicher zu gehen, dass sich kein anderer Funksender in der Umgebung befunden hat, wurden mit einem fünften Sensorknoten, welcher, mit einer im Rahmen dieser Diplomarbeit entwickelten Software *mspect*, als Spektrumanalysator programmiert war, während der gesamten Dauer des Experimentes die benutzten Frequenzen überwacht (Vgl. 2.3.3).

Es wurden drei antwortende Sensorknoten auf dem selben Mast auf der gleichen Höhe am Ankerpunkt aufgestellt. Auf einem zweiten Mast war ein anfordernder Sensorknoten montiert. Die Sensorknoten befanden sich, um vor Umwelteinflüssen geschützt zu sein, in einem Kunststoffgehäuse, wobei die Antennen durch ein Bohrung im Deckel am höchsten Punkt der Konstruktion ins Freie geführt wurden. Die Stromversorgung wurde durch *6 AAA* Zellen, pro Sensorknoten, ausgeführt. Die Steuerung der Messung erfolgten durch einen am anfordernden Sensorknoten angeschlossenen Laptop. Die Einzelmessungen fanden hintereinander über eine Dauer von rund fünf Minuten statt. Dabei wurde der anfordernde Sensorknoten der Reihenfolge nach an den in der Karte (Abbildung 3.11) markierten Punkten 1 – 10 in einer Entfernung von *10 m* bis *250 m* zum Ankerpunkt aufgestellt, wobei die Abstände mit einem *25* Metermaßband abgemessen wurden. Nach jeder Messung wurden die Datenaufzeichnungen, welche im folgenden Kapitel näher spezifiziert werden, von der *SD*-Karten auf die Festplatte des Laptops gesichert. Obwohl bei einem Abstand von *250 m* nicht mehr von Freiraumdämpfung ausgegangen werden kann, wurde die Messung zur Reichweitenbemessung durchgeführt.

3.3.1.1 Datenaufzeichnung während des Freifeldexperimentes

Da in den folgenden Experimenten die Daten der Messungen nicht im Arbeitsspeicher gehalten werden können, werden diese auf die *SD* – Karte ausgelagert. Dabei wurden jeweils *300* Werte im Arbeitsspeicher zwischengespeichert, obwohl der *MCI* – Treiber einen *512 Byte* (Blockgröße) großen Cache besitzt. Somit ist sichergestellt, das die Übertragung der Funkpakete unterbrechungsfrei erfolgt. Es werden pro Messpunkt drei Dateien angelegt, in welchen die Daten jeweils für einen antwortenden Sensorknoten gespeichert werden. Der Dateiname setzt sich wie folgt zusammen:

```
range<reale Entfernung>to<Nummer des Knotens>
```

Die nummerischen Daten werden zum späteren Nachbearbeiten im ASCII Format, Komma - separiert gespeichert. Dabei wird pro Messwert genau eine Zeile verwendet. Tabelle 3.3 zeigt welche Werte gespeichert werden, deren Datentyp und wie diese später zu interpretieren sind.

	TOF,	TCP,	RSSI,	RSSIACK,	LQI,	LQIACK,
Bedeutung	Wert der Zeitmessung	Bearbeitungszeit im Empfängerknoten	Wert des Antwortpaketes	Wert des Anforderungspaketes	Wert des Antwortpaketes	Wert des Anforderungspaktes
Max. Wert	uint32_t	uint32_t	uint8_t	uint8_t	uint8_t	uint8_t
Interpretation	In Takten	In Takten	Dezimal, ohne RSSI-Offset	Dezimal, ohne RSSI-Offset	-	-

Tabelle 3.3 Format der Datenaufzeichnung des messenden Sensorknotens

3.3.2 Planung und Durchführung des Innenraumexperimentes

Abbildung 3.12 Anfordernder Sensorknoten im Versuchsaufbau des Innenraumexperimentes

Das Ziel dieses Experimentes war es, die Auswirkungen von Einflüssen einer nicht idealen Umgebung auf das Laufzeitmesssystem zu analysieren. Es wurden Messungen im Flur des oberen Stockwerkes des Institutes für Informatik durchgeführt, wobei sich in der direkten Sichtlinie keine Hindernisse befanden. Durch die Deckenhöhe kann bei einer Entfernung von bis zu rund *10* Metern von Freiraumdämpfung ausgegangen werden, wenn sich keine Personen im Flur zwischen den Sensorknoten befinden (Gleichung (3.1)). Es wurden verschiedene Messpunkte in einer Entfernung von bis zu *30 m* mit einem Maßband vermessen. Es wurden dieselben Masten aus dem Freifeldexperiment verwendet, wobei deren Höhe auf *2 m* verringert wurde. Um Erkenntnisse über die optimale Höhe für einen Sensorknoten zu erlangen, wurden die drei Sensorknoten am Ankermast in unterschiedlicher Höhe angebracht. Sensorknoten *2* wurde mittig am Mast in rund *1,30 m* Höhe angebracht, um einen maximalen Abstand zur Decke und zum Boden zu gewährleisten. Sensorknoten *4* wurde an der Spitze des Mastes, rund *2,10 m* hoch, und Sensorknoten *3* am Fuß des Mastes, rund

50 cm hoch, angebracht. Der anfordernde Sensorknoten wurde an einem zweiten Mast ebenfalls mittig in einer Höhe von *1,30 m* befestigt. Um einen maximalen Abstand von den Wänden zu erhalten, wurden die beiden Masten in der Mitte des Flures aufgestellt. Es wurde mit der Messung am *30 m* Messpunkt begonnen. Weitere Messungen erfolgten, in *5 m* Schritten in Richtung des Ankers. Ab einem Abstand von *5 m* wurde die Entfernung meterweise reduziert. Alle weitere Punkte wie die Stromversorgung, die Datenaufzeichnung und die Überwachung des Experiments mit dem Spektrumanalysator wurden vom Freifeldexperiment übernommen.

3.3.3 Lokalisierungsexperiment

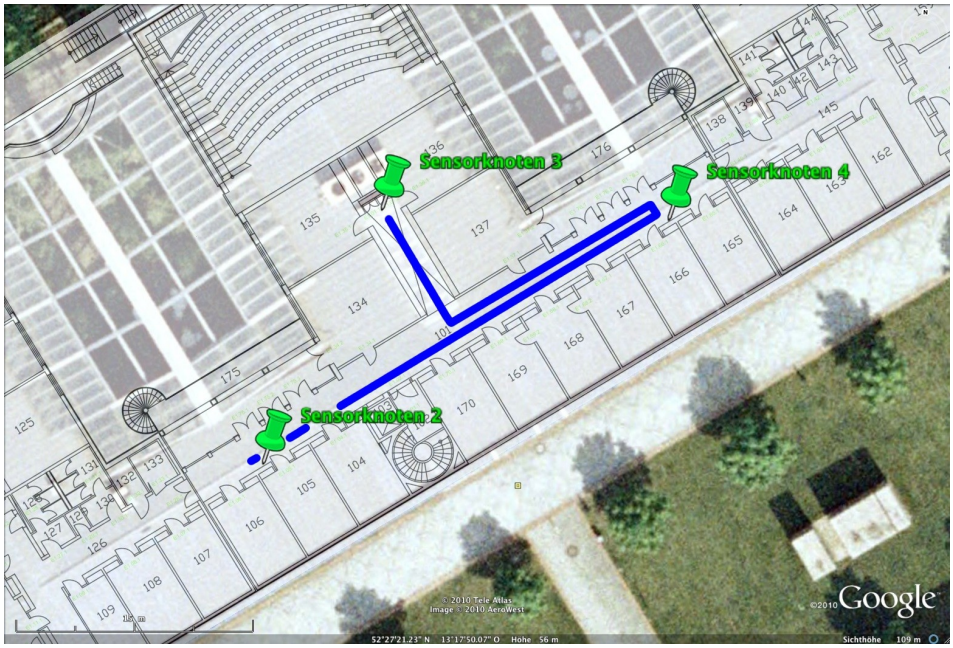

Abbildung 3.13 Position der Ankerknoten beim Lokalisierungsversuch

Ziel dieses Experimentes war es, das entwickelte Laufzeitmesssystem in einem praxisnahen Versuchsaufbau bezüglich der Einsetzbarkeit in einem Echtzeitlokalisierungssystem zu analysieren und mit einem kommerziellen Produkt zu vergleichen. Als Vergleichsobjekt diente der *NanoPAN* [NAN09] Transceiver, ein kommerzielles Produkt welches speziell für Laufzeitmessungen entwickelt wurde. Dieser Transceiver ist auf der Hardwareplattform des *FeuerWhere* Projektes verfügbar (Vgl. [FEU10]). Diese Hardwareplattform basiert, wie die in dieser Arbeit verwendete

Hardwareplattform, ebenfalls auf dem *LPC2387* Mikrocontroller, enthält aber zusätzlich neben dem *NanoPAN* Transceiver noch weitere Komponenten.

Es soll in diesem Experiment eine sich bewegende Person unter Innenraumbedingungen lokalisiert werden. Der Aufbau des Experimentes folgt dabei den Erkenntnissen aus dem Innenraumexperiment, sodass die als Anker dienenden Sensorknoten mit dem *CC1101* Transceiver in *50 cm* Höhe im Obergeschoss des Institutes für Informatik wie in Abbildung 3.13 verzeichnet ausgebracht wurden. Die Anker des Vergleichssystems mit dem *NanoPAN* Transceiver werden an den selben Punkten, den Erfahrungswerten aus dem *FeuerWhere* Projekt folgend, in rund *1,5 m* Höhe an den, aus den vorangegangenen Experimenten bekannten, Kunststoffmasten befestigt aufgestellt. Es wurde mit beiden Hardwareplattformen die selbe Strecke nacheinander in langsamen Tempo abgeschritten, wobei jeweils der zu lokalisierende Sensorknoten in der Hand auf Brusthöhe(*1,3 m*) getragen wurde. Die Messstrecke ist in Abbildung 3.13 durch den blau eingezeichneten Pfad verdeutlicht und beginnt bei Sensorknoten *3*.

Da sich Personen normalerweise mit einer maximalen Geschwindigkeit von *1,5 m* pro Sekunde durch ein Gebäude bewegen und ein *„RToF“* - Messzyklus rund *3 µs* benötigt, hat die Person nach *74* Messungen zu allen drei Ankerknoten ihre Position um *1 m* verändert. Es war deshalb notwendig die Programmierung des anfordernden Sensorknoten so zu modifizieren, dass hintereinander alle drei Ankerknoten abgefragt werden konnten. Weiterhin sind die gewonnenen Erkenntnisse der beiden vorangegangenen Experimente, zur Korrektur der Einzelmessungen verwendet worden (siehe Kapitel 4.2). Die Datenaufzeichnung und Überwachung des Experiments erfolgte wie in den vorangegangenen Experimenten beschrieben. Die für die Abstandsmessungen mit dem *NanoPAN* eingesetzte Software, fragt über den von der Firma *Nanotron* bereitgestellten Hardwaretreiber, zyklisch die Abstände zu den Ankerknoten ab und speichert diese in einer Datei auf der *SD* - Karte.

4 Analyse und Evaluierung

In diesem Kapitel werden die Ergebnisse der drei Experimente analysiert und bewertet. Dazu erfolgt ein Vergleich der Messwerte und ihrer Abweichungen mit einer selbstdurchgeführten *RSSI* basierten Abstandsbestimmung. In weiteren Schritten soll überprüft werden ob mit einem hybriden Ansatz, also mit der Verknüpfung der Werte aus den Laufzeitmessungen mit den Messwerten von *LQI* und *RSSI*, genauere Abstandsapproximationen, durch Filterung von nicht *LoS* Signalen, möglich sind. Abschließend soll die Nutzbarkeit des Systems an Parametern wie benötigte Paketanzahl und daraus folgender Energieverbrauch, sowie dem Speicher- und Rechenzeitbedarf, bewertet werden.

4.1 Ergebnisse des Freifeldexperimentes

Dieser Abschnitt stellt die Messergebnisse des Freifeldexperimentes vor, dabei soll neben der Analyse der Messabweichung auch die Offset – Bestimmung erfolgen. Dazu wurden für die *10* Messpunkte insgesamt über *430.000* Einzelmesswerte ausgewertet. Diese Messwerte wurden zur Auswertung in *matlab* weiterverarbeitet. Dazu war es notwendig für die einzelnen Plots eigene Funktionen in *matlab* zur programmieren. Wenn nicht anders benannt, wird in diesem Kapitel von Metern als Längenmaß und Taktzyklen des Mikrocontrollers *(72* Megahertz*)* als Zeitmaß ausgegangen.

4.1.1 Offset - Bestimmung

Wie in Kapitel 2.1.2 erläutert, wird zur Signalverarbeitung im Transceiver eine bestimmte Zeit benötigt, welche nicht genau dokumentiert ist. Der einzige Hinweis ist in [NAM07] im Kapitel 3.1.1 zu finden, wo ein *propagation delay* von *4* Mikrose-

kunden beschrieben wird. Um das bei jeder Abstandsmessung abzurechnende Offset bestimmen zu können, wurden die Ergebnisse der Messung ausgewählt, welche die geringste Streuung für alle drei antwortenden Sensorknoten aufweisen. Dies ist beim *10 m* Messpunkt der Fall. Die ausgewählten Messwerte werden in folgendem Histogramm dargestellt, wobei die oberen und unteren 10 Prozent der Messwerte zur Filterung von Ausreißern nicht einbezogen wurden. Die dargestellten Werte sind die Messwerte der *ToF* Messung, von welchen die korrespondierenden Messwerte für die *TCP* Messung abgezogen wurden.

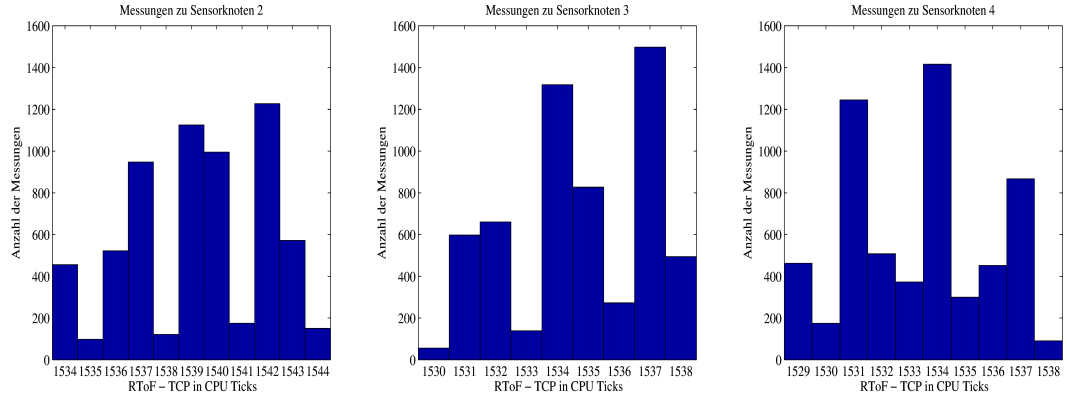

Abbildung 4.1 Histogramm zur Analyse des propagation delay

In allen drei Histogrammen sind jeweils drei Spitzen zur erkennen, sodass wenn die Taktfrequenz des Transceivers in die Betrachtung einbezogen wird, die Werte um drei Taktzyklen des Transceivers schwanken. Es wird, aufgrund der Drift der vier asynchron zueinander laufenden Taktgeber des Laufzeitmesssystems, von einer Gleichverteilung des Auftretens der steigenden Flanke am *GDO2* Pin Transceivers ausgegangen. Deshalb entspricht der Median zwischen diesen Spitzen dem wahren Messwert. Die Ausbreitungszeit, des den direkten Weg von Sender zu Empfänger, laufenden Signals beträgt bei *10 m* Abstand *76,61 ns,* was rund 5 Taktzyklen des Mikrocontrollers entspricht. Zieht man diese vom Median der Messwerte am *10 m* Messpunkt ab, so ergeben sich für die einzelnen antwortenden Sensorknoten die in der folgenden Tabelle aufgelisteten Werte für das Offset. Diese Werte werden auch für die folgenden Plots verwendet.

	Sensorknoten 2	Sensorknoten 3	Sensorknoten 4
Offset	1536	1530	1528

Tabelle 4.1 Offsets der Sensorknoten für das Freifeldexperiment

Die Streuung der einzelnen Offsets ist groß und bedeutet, dass der Transceiver des Sensorknotens *4* das Funksignal rund drei Takte früher dekodieren konnte, als der Transceiver des Sensorknotens *2*. Aufgrund der hier vorliegenden Daten lassen sich keine Schlüsse über die Ursache dieser Differenz ziehen. Vermutlich spielen Fertigungsschwankungen, welche sich auf die Signalqualität auswirken eine Rolle, sodass die Demodulation unterschiedlich lange Zeit in Anspruch nehmen kann. Diese Vermutung kann durch die Tatsache untermauert werden, dass auch die *LQI* und *RSSI* Werte bei verschiedenen Sensorknoten, bei gleicher Konfiguration und Position sehr unterschiedlich sind.

Die ermittelten Offsets lassen sich in vier Teile aufteilen. Ein Teil entfällt auf das Senden und ein Teil auf das Empfangen eines Paketes, wobei für eine Einzelmessung zwei Pakete versendet werden. Es sind durchschnittlich *5,32 µs* Offset für das *propagation delay* zu berücksichtigen. Ob dieser Wert konstant ist, werden die Analysen des Offsets für die im Innenraumbereich gesammelten Daten ergeben (siehe 4.2).

4.1.2 Abweichungen der Abstandsmessung

Für die Umrechnung der für die Abstandsmessung verwenden Einzeldaten, (Vgl. 3.3.1.1) von Zeitstempeln in Strecken, wurde folgende Gleichung benutzt:

$$d = TOF - TCP - Offset \tag{4.1}$$

In den nachfolgend dargestellten Diagrammen werden die Messwerte, welche an den einzelnen Messpunkten aufgenommen wurden, entsprechend ihres realen Abstandes als Abszisse und anhand des Medians der Messwerte als Ordinate aufgetragen. Wobei für jeden Messpunkt die Abweichungen der Einzelmessungen als Boxplot mit einer *whisker* - Länge des *1,5*-fachen des Interquartilsabstands dargestellt werden. Vor der Darstellung wurden die Daten gefiltert, sodass nur Werte, welche nicht mehr als *300 m* vom Median aller Messwerte abweichen, berücksichtigt wurden. Diese Filterung basiert auf der Annahme das der maximale mögliche Abstand zum Ankerpunkt *300 m* beträgt, und kann auch in Echtzeit auf dem Sensorknoten durchgeführt werden.

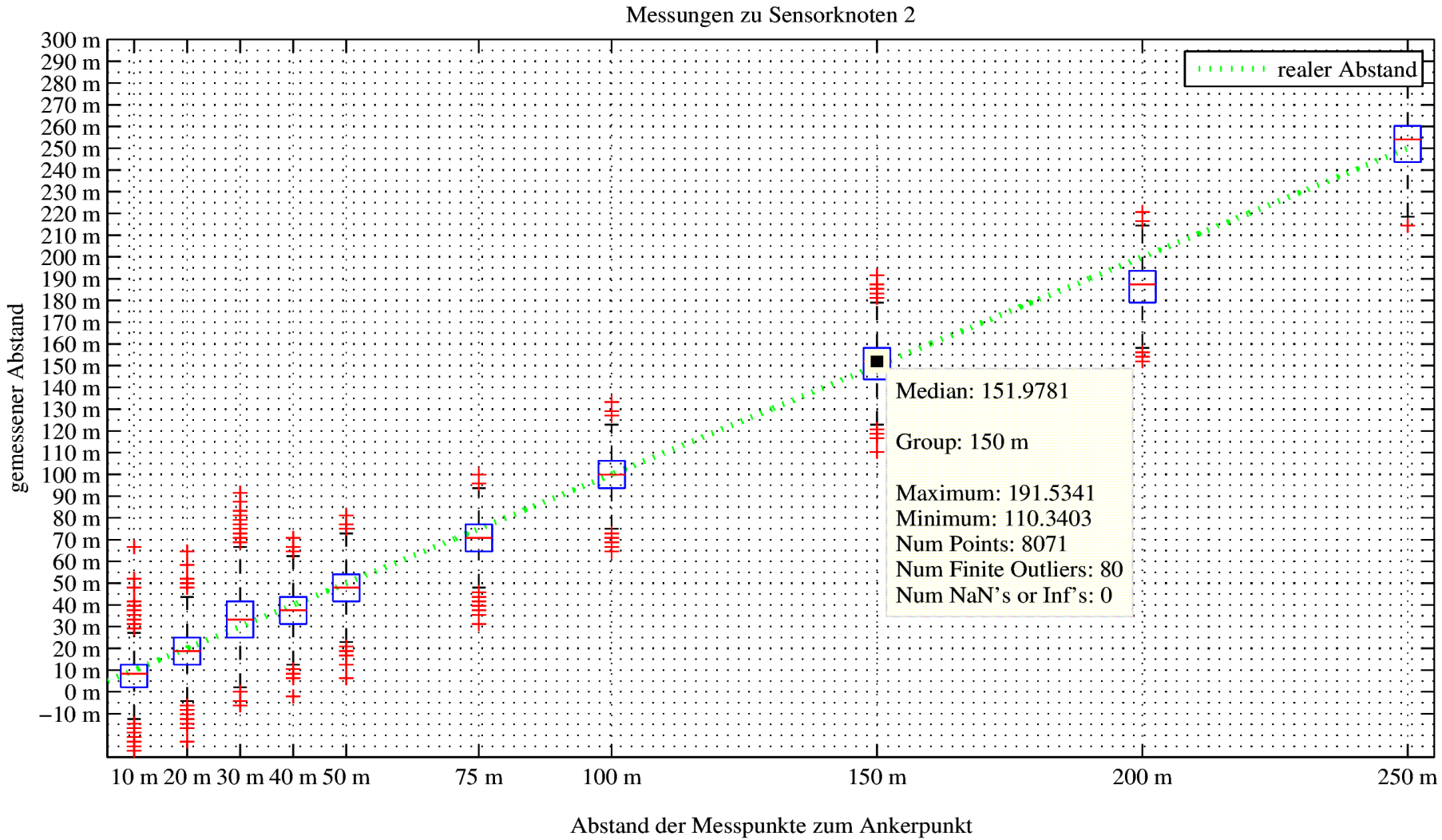

Abbildung 4.2 Messergebnisse der Abstandsmessung zum Sensorknoten 2 aus dem Freifeldexperiment.

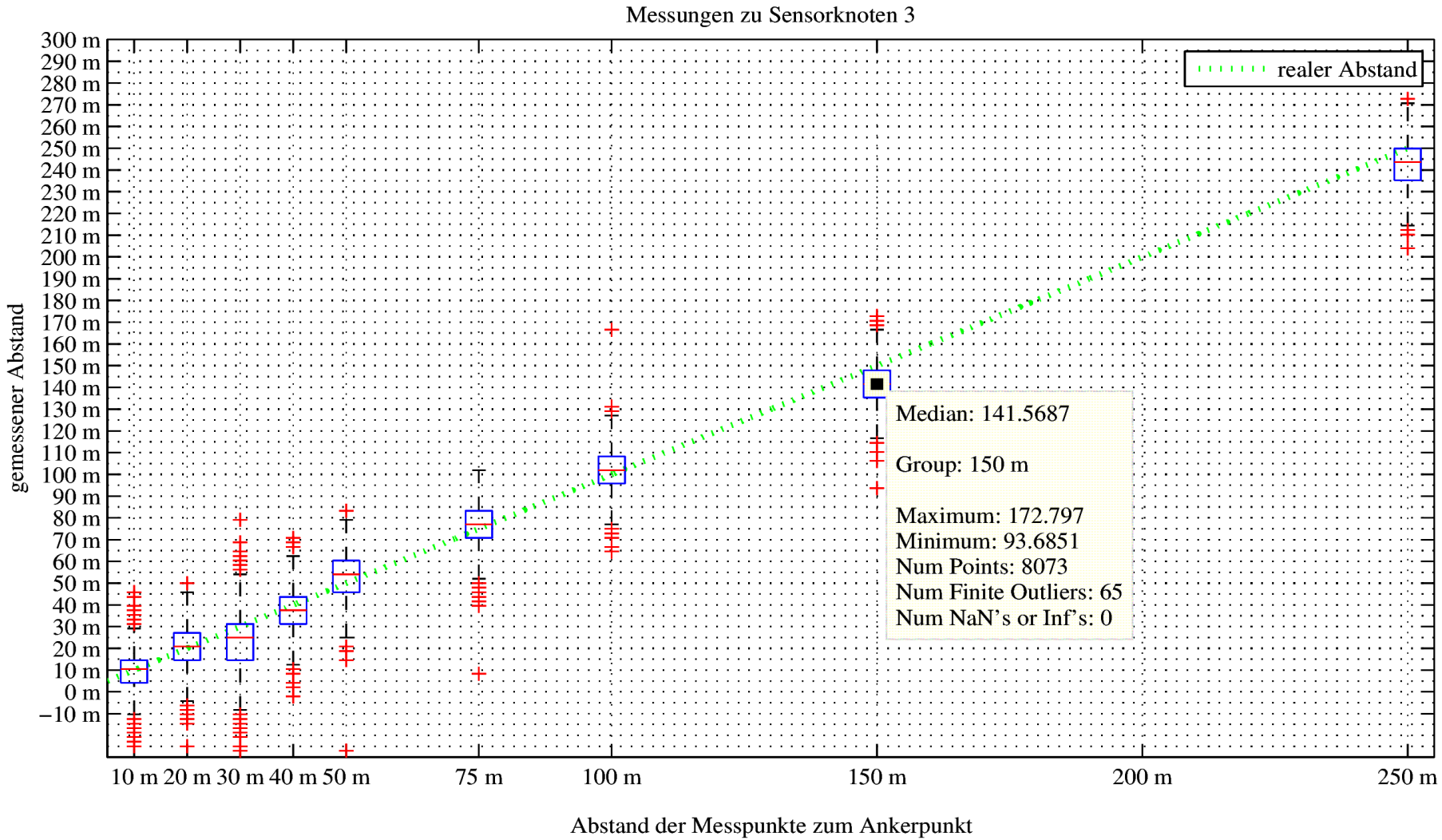

Abbildung 4.3 Messergebnisse der Abstandsmessung zum Sensorknoten 3 aus dem Freifeldexperiment.

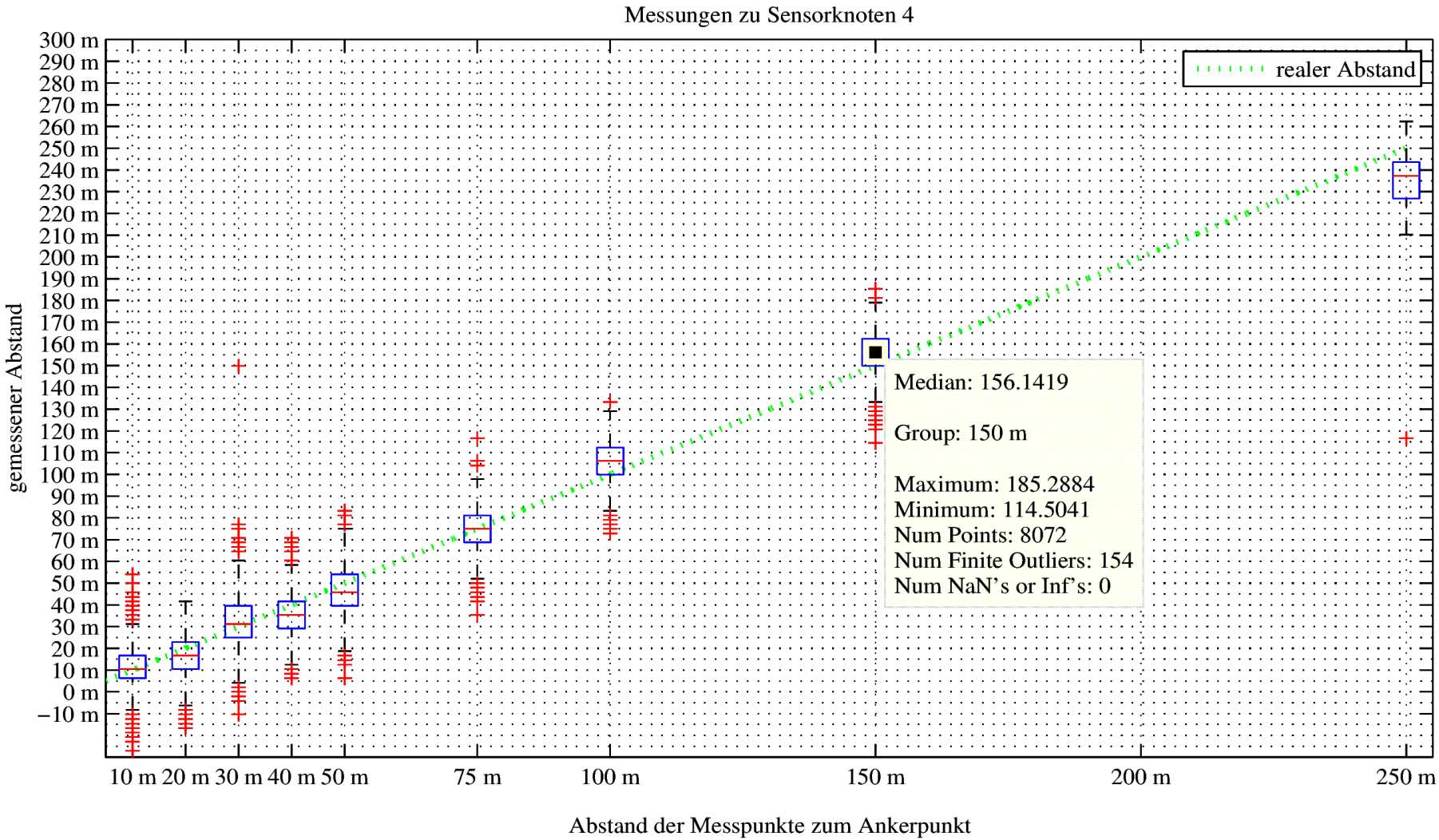

Abbildung 4.4 Messergebnisse der Abstandsmessung zum Sensorknoten 4 aus dem Freifeldexperiment.

In den Diagrammen für die Sensorknoten drei und vier fehlen die Boxplots für den *200 m* Messpunkt, da an diesem Punkt keine Funkkommunikation zu den beiden Sensorknoten möglich war. Die absolute Abweichung über alle Messungen beträgt bezogen auf die Messstrecke *5 %* und bei *90 %* der Messpunkte *2,6 %*. Diese Abweichung liegt im Bereich zwischen einem Mikrocontroller- und einem Transceiver – Takt. Die Anzahl der Messwerte ist im oberen und unteren Quartil annähernd gleich und die Streuung dieser Werte beträgt rund *10 m*, wobei in Einzelfällen die Messwerte im Bereich von *20 m* variierten. Die Verteilung der Werte innerhalb des Interquartilsabstandes ist dabei, wie im Diagramm in Abbildung 4.1 bereits dargestellt, durch die Abhängigkeit der Messwerte vom Transceivertakt als kammartig zu charakterisieren. Innerhalb der *whisker* beträgt die Streuung der Messwerte *40 m* bis *60 m*, also bis zu sechs Taktzyklen des Transceivers. Extreme Ausreißer sind nur vereinzelt zu beobachten und die Menge der milden Ausreißer ist mit rund *1,6 %* bezogen auf die Anzahl der Messwerte gering.

4.1.3 Vergleich mit der Leistungsdichtemessung

Für jedes gesendete Paket wurden neben den Daten welche für die Laufzeitmessung relevant sind, zusätzlich für jedes Paket die *RSSI* Messwerte gespeichert. Diese werden wie im Kapitel 1.1.1.1 erläutert in Abstandwerte umgerechnet, wobei hier von Freiraumdämpfung ausgegangen wurde. Nach der Umrechnung werden *RSSI* Messwerte, wie im vorangegangen Kapitel anhand des Messpunktes als Abszisse und ihrem Median als Ordinate im Diagrammen als Boxplot aufgetragen. Das Diagramm 4.5 zeigt nur Messwerte bis zum *100 m* Messpunkt, da die *RSSI* Messwerte an den weiteren Messpunkten keine besseren Resultate zeigen und sich in der Streuung nicht von den Abgebildeten unterscheiden.

Die Abstandsapproximationen aus den *RSSI* Messwerten nähern sich im besten Fall an den realen Abstand in dem Maße an, in welchem die gemessenen Abstände der *RToF*-Messung maximal abweichen. Die Streuung der *RSSI* Messwerte ist deutlich größer als bei den Messwerten der *RToF* - Messung. Weiterhin ist ersichtlich, dass die Dämpfung der Leistungsdichte nicht kontinuierlich abnimmt, sondern beispielsweise am *100 m* Messpunkt im Vergleich zum *50 m* Messpunkt immer niedriger ist.

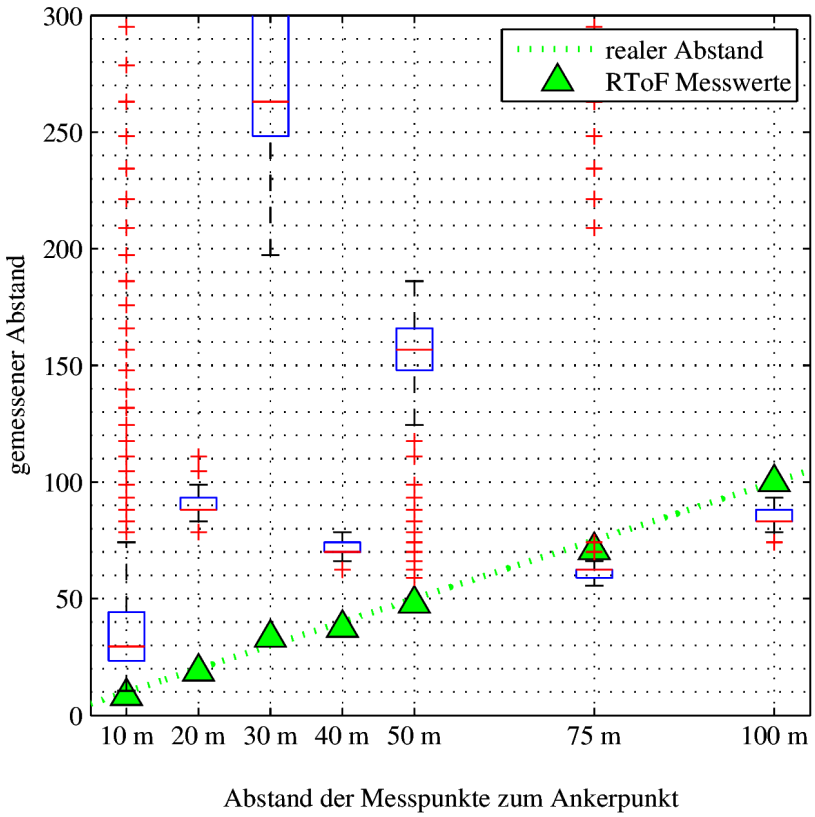

Abbildung 4.5 Vergleich der RSSI und RToF Messwerte des Freifeldexperimentes

Der Grund für die Abweichungen liegt vermutlich in der Bestimmungsmethode des *RSSI* im Transceiver, da die Messungen in einer Umgebung, in der von Freiraumdämpfung ausgegangen werden kann, durchgeführt wurden. Eine Vermutung ist, dass die *AGC* und die *FOC* in manchen Situationen dazu führen, dass das absolute Maximum des Pegels im Empfangspfad des Transceivers, nicht als Teil des Signals gewertet, sondern als Störgröße ausgefiltert wird. Um diese Fehlinterpretation der Leistungsdichte des empfangenen Funksignals zu verhindern, könnten alle automatischen Anpassungen abgeschaltet werden, was eine stark verminderte Reichweite der Transceiver zur Folge hat.

4.1.4 Analyse der Abhängigkeit von der Paketanzahl

In der Auswertung im Kapitel 4.1.2 sind für jeden Messpunkt Einzelwerte von über *8000* Paketen zur Bestimmung des Abstandes ausgewertet worden. Diese Paketanzahl ist für den praktischen Einsatz zu hoch, da damit der Energiebedarf für die Ab-

standsapproximation die verfügbaren Energieressourcen eines Sensorknotens übersteigt. Außerdem würden die Messungen für eine Lokalisierung von bewegten Objekten oder Personen zu viel Zeit in Anspruch nehmen, sodass sich nach einer Messung zu einem Lokalisierungsanker das Objekt schon weiterbewegt hat und deshalb keine verwertbaren Ergebnisse für Messungen zu den anderen Ankerknoten möglich sind. In diesem Kapitel wird deshalb analysiert, in welchem Zusammenhang die Paketanzahl mit den Abweichungen der Abstandsmessung stehen. Dazu werden für das folgende Diagramm (4.6) die Messwerte der Laufzeitmessungen des Freifeldexperimentes zum Sensorknoten zwei verwendet. Dabei wird die Anzahl der verwendeten, beginnend bei zwei Paketen, schrittweise gesteigert, wobei keine gesonderte Auswahl der Messwerte getroffen wird, sondern beim ersten ermittelten Messwert begonnen wird. Es wurden für die Darstellung die Messwerte von zwei Messpunkten ausgewählt. Die Auswahl der Messpunkte erfolgte nach der Größe der Abweichung und der Streuung der Messwerte für den jeweiligen Messpunkt. Für den ersten Messpunkt wurde der *100 m* Messpunkt ausgewählt, da für diesen der berechnete Abstand gering abweicht und die Streuung der Werte durchschnittlich ist. Am *30 m* Messpunkt, welcher als zweites ausgewählt wurde, weicht der berechnete Abstand rund *3,4 m* vom realen Wert ab und die Streuung ist überdurchschnittlich hoch. Es wird für jeden Messpunkt eine Kurve im Diagramm aufgetragen, wobei die Abszisse von der Paketanzahl und die Ordinate von der Abweichung des Medians der berechneten Abstände bestimmt wurde.

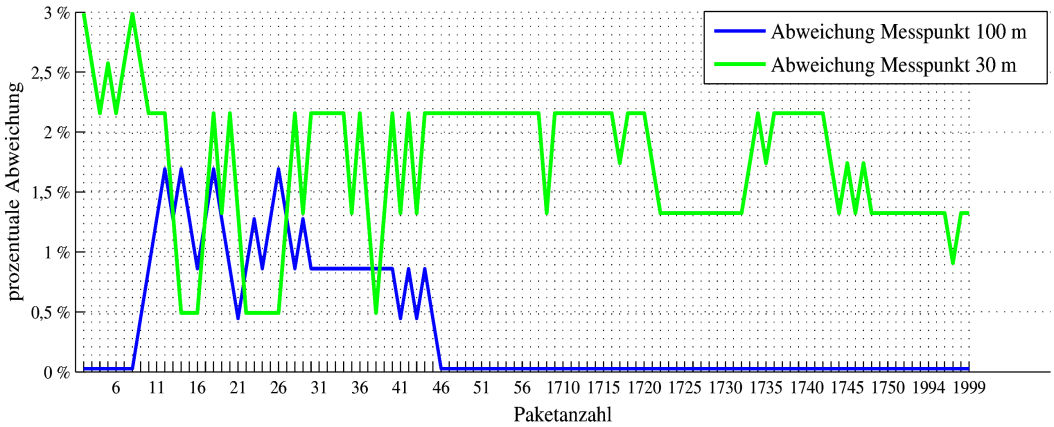

Abbildung 4.6 Abhängigkeit der Abweichungen der Laufzeitmessung von der Paketanzahl

Bei der Betrachtung des Diagrammes ist zu beachten, dass die Abszissenachse nicht kontinuierlich abgebildet ist. Da sich die Werte für die fehlenden Abszissenwerte

nicht verändert haben, wurde das Diagramm zur besseren Lesbarkeit angepasst. Es ist zu erkennen, dass sich im besten Fall nach rund 50 Einzelmessungen Abstandwerte mit geringer Abweichung errechnen lassen. Dagegen ist im schlechten Fall ein Festlegung der besten Paketanzahl schwierig, da sich die Abweichung mit steigender Paketanzahl vergrößern kann. Setzt man für eine möglichst genaue Abstandsapproximation die minimale Einzelmessungsanzahl auf die im besten Fall ausreichenden 50 Messungen an, so währe die benötigte Zeit mit *0,5 s* für eine Lokalisierungsmessung zu drei Ankern ausreichend gering (Vgl. Kapitel 3.3.3).

4.2 Ergebnisse des Innenraumexperimentes

Zur Auswertung der Messwerte des Innenraumexperimentes wurden die Werte in gleicher Weise wie in der Auswertung des Freifeldexperimentes (Vgl. 4.1.2) bearbeitet. Die in Kapitel 4.1.1 bestimmten Offsets konnten nicht auf das Innenraumexperiment übertragen werden, deshalb wurden hier für den *20 m* Messpunkt, neue Offsets bei gleicher Vorgehensweise bestimmt. In Tabelle 4.2 sind die ermittelten Werte zur Kompensation des *propagation delay* aufgelistet.

	Sensorknoten 2	Sensorknoten 3	Sensorknoten 4
Offset	1538	1536	1528

Tabelle 4.2 Offsets der Sensorknoten für das Innenraumexperiment

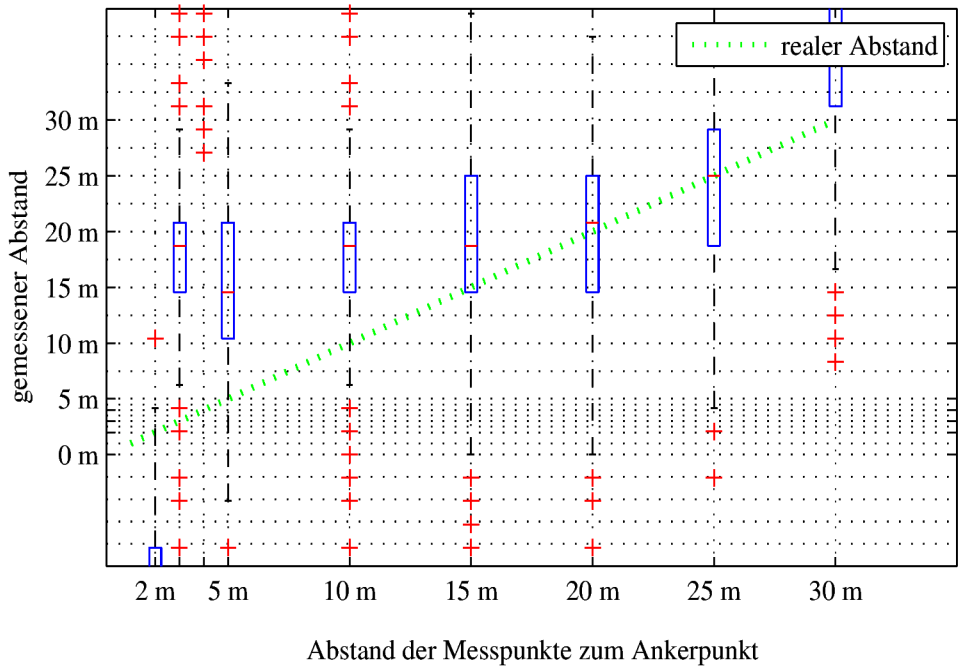

Abbildung 4.7 Messergebnisse der Abstandsmessung zum Sensorknoten 2 aus dem Innenraumexperiment.

Das Diagramm 4.7 zeigt die Abweichungen und die Streuung der Messwerte an den einzelnen Messpunkten für die Abstandsmessungen zum Sensorknoten *2*. Die Abweichung der Messwerte ist im Vergleich zu den Messwerten aus dem Freifeldexperiment hoch, deshalb wird auf die Vorstellung von prozentualer Abweichung verzichtet. Obwohl die Messwerte der Messpunkte bei 15 m, 20 m und 25 m eine Abweichung aufweisen, welche innerhalb eines Mikrocontrollertaktes liegt, ist ohne die zusätzlichen Informationen, über die Zuordnung der Messwerte zu den Messpunkten, keine nachträgliche Zuordnung möglich.

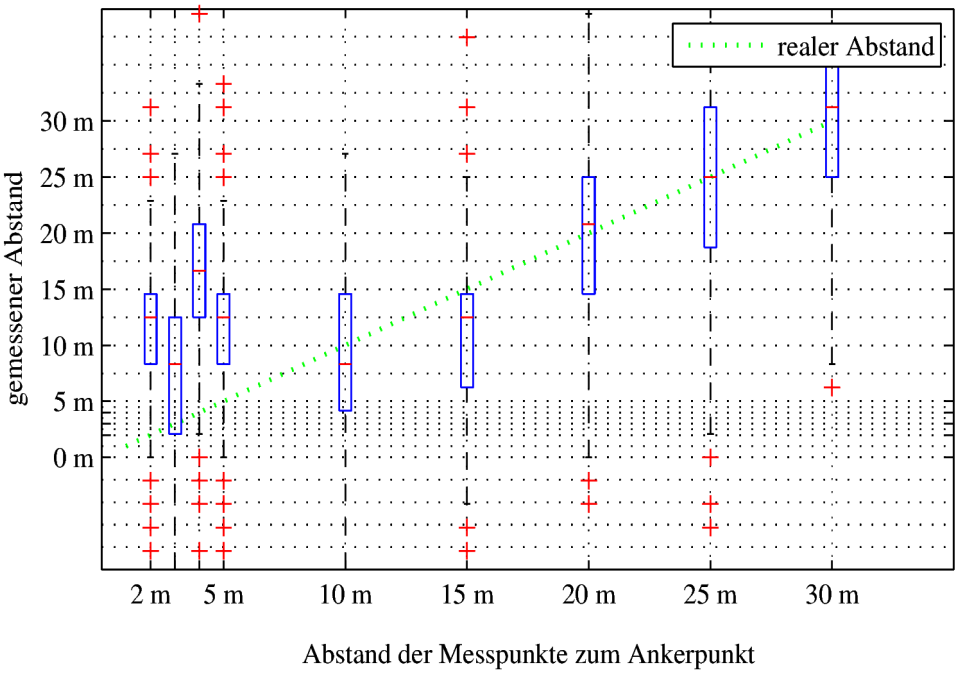

Abbildung 4.8 Messergebnisse der Abstandsmessung zum Sensorknoten 3 aus dem Innenraumexperiment.

Die Messwerte der Abstandmessungen zu Sensorknoten 3 weisen, wie in Abbildung 4.8 ersichtlich, eine deutliche geringere Abweichung auf, sie liegt bei den Messungen über *10 m* Abstand innerhalb von vier Metern. Die Streuung der Messwerte für alle Messpunkte beträgt durchschnittlich im Interquartilsabstand *10 m*. Innerhalb der *whisker* streuen die Messwerte, im Vergleich mit dem Freifeldexperiment, mit rund *40 m* weniger. Ausreißer außerhalb der *whisker* streuen um rund *50 m*.

Auf die Darstellung der Messergebnisse aus den Abstandsmessungen zu Sensorknoten 4 wird, aufgrund mangelnder Verwertbarkeit, verzichtet. Die Messergebnisse weichen aufgrund der schlechten Sendeposition des Sensorknoten 4 nahe der Decke stark von den realen Abständen ab, da die zerklüftete Struktur der Decke, welche zum Beispiel durch die Lampenkästen entsteht, die Ausbreitung der elektromagnetischen Wellen negativ beeinflusst. Deshalb ist es nur möglich stark reflektierte Signale zu erhalten, was dazu führt, dass keine verwertbare Abstandsbestimmung möglich ist.

4.2.1 Analyse der Messergebnisse im Nahbereich

Bei allen Messergebnissen dieses Experimentes sind große Abweichungen bei einem Abstand von unter *10 m* zu verzeichnen. Der Grund dafür liegt in Ausgangsverstärkung des Transceivers. Diese ist auf den maximalen Wert eingestellt um, gerade im Innenraumbereich, auch in abgeschatteten Bereichen noch eine Verbindung zwischen den Sensorknoten zu ermöglichen. Durch die hohe Ausgangsleistung des sendenden Transceivers ist die Leistungsdichte im nahen Umfeld dessen sehr hoch. Dadurch wird der Eingangspegel im empfangenden Transceiver so hoch, dass die *AGC* diesen nicht mehr kompensieren. Der *ADC* wird deshalb für das *LOS* Signal in die Sättigung getrieben (Vgl. 2.1.2). Der Demodulator erhält dadurch nur annähernd maximale Spannungswert vom *ADC* und kann kein Signal auswerten. Die reflektierten Anteile des Funksignals, welche im Innenraumbereich immer auftreten, besitzen eine niedrigere Leistungsdichte, somit sind diese besser demodulierbar. Da diese Signalanteile später eintreffen als das *LOS* – Signal, ist auch die gemessene Laufzeit höher und somit der gemessene Abstand. Um dieses Problem zu lösen, wurde zu einem frühen Zeitpunkt der Diplomarbeit mit einer automatischen Anpassung der Sendeleistung beider beteiligter Sensorknoten experimentiert. Auf diesem Wege konnte keine Lösung gefunden werden, da die implementierte Anpassung zu viel Zeit in Anspruch genommen hat, weil dazu mindestens *10* Funkpakete zwischen den Sensorknoten ausgetauscht werden mussten und somit die benötigte Zeit pro Einzelmessung zu hoch für eine Lokalisierung von bewegten Objekten, gewesen wäre.

Durch die Einführung des *CC1101*, einer weitgehend kompatiblen Weiterentwicklung des *CC1100* Transeivers, auf der *AVSExtrem* - Hardwareplattform, konnte ein bis dahin undokumentiertes Leistungsmerkmal genutzt werden. In der technischen Notiz [JEN09] wird eine Methode beschrieben, welche es erlaubt die Dämpfung im Empfangsteil manuell zu verändern. Dazu wird bereits während des Empfangs der Präambel der *RSSI* – Wert ausgelesen und ausgewertet. Noch während des Empfangs der Präambel kann nun, über die Bits vier und fünf des Registers *FIFOTHR(0x03)* im *CC1101* Transceiver, die Dämpfung in drei Stufen (*6dB, 12dB, 18dB*) manuell nach-

reguliert werden. Dadurch ist es möglich den resultierenden Pegel des *LOS* Signals im Empfangsweg, soweit abzusenken, dass dieses Demoduliert werden kann. Dieser Vorgang ist auf den Empfänger begrenzt und der Sender kann weiterhin mit voller Sendeleistung betrieben werden. Durch die Nutzung dieses neuen Leistungsmerkmals des Tranceivers steigt Präzision der Abstandsbestimmung, gerade für die Lokalisierung im Inneren von Gebäuden, da der Abstand zu den Ankern hier gering sein kann. Diese Anpassung für den Nahbereichsempfang wurde für den Lokalisierungsversuch nachimplementiert.

4.2.2 Vergleich mit der Leistungsdichtemessung

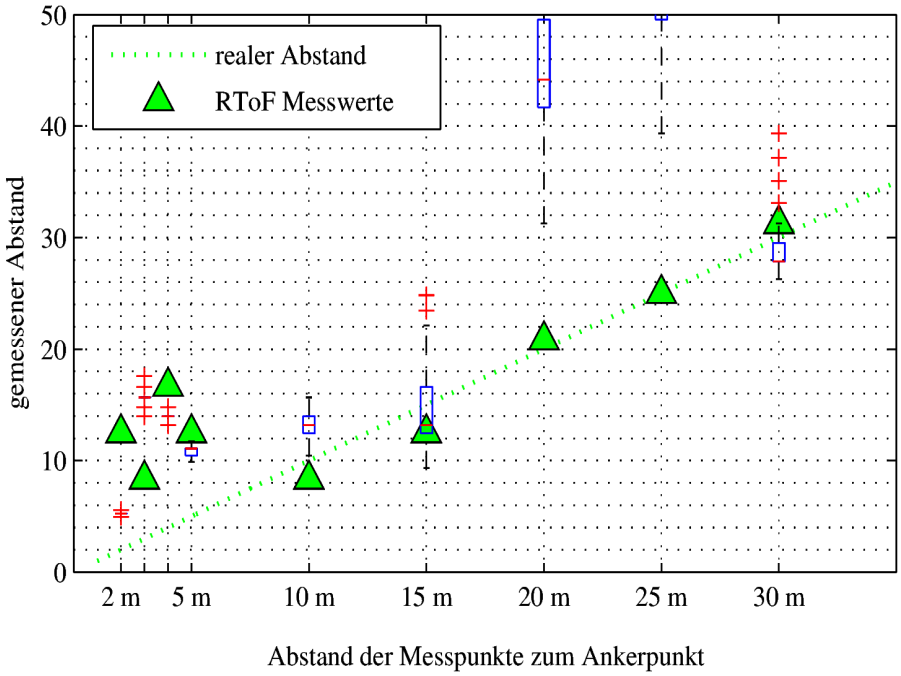

Abbildung 4.9 Vergleich der RSSI Werte und der RToF Messwerte des Innenraumexperimentes

Abbildung 4.9 zeigt ein Diagramm, dass wie in Kapitel 4.1.3 die Abweichung der aus den *RSSI* Messwerten errechneten Abstände mit den Abständen welche aus der *RToF* – Messung vergleichbar macht. Es ist ersichtlich, dass die Abstände aus der *RSSI*-Messung bei der Mehrzahl der Messpunkte lediglich um *5 m* von den *RtoF*-Messungen abweichen. In zwei Fällen, an den Messpunkten bei *20 m* und bei *25 m*, ist die Abweichung mit *20 m* groß. Charakteristisch ist hierbei, dass die Streuung an diesen Messpunkten höher ist als bei den anderen Messpunkten. Im Nahbereich werden die Messwerte aus den *RToF* Messungen durch die *RSSI* Messwerte bestätigt, sodass wie

in Kapitel Fehlerquellen der Messmethoden beschrieben, hier die reflektierten Signalanteile gemessen wurden und nicht das *LOS* Signal.

4.2.3 Hybride Methoden im realen Umfeld

Es ist erwiesen, dass sich, mit den beiden Abstandsmessmethoden Leistungsdichtemessung und Laufzeitmessung, nur die Länge des Weges, welchen die elektromagnetischen Wellen vom Sender zum Empfänger zurückgelegt haben, bestimmen lässt, nicht aber der tatsächliche Abstand. Es wurde gezeigt, dass die *RToF*-Messung im Mittel weniger von Reflexionen und anderen Störungen beeinflusst wird, sodass zum Teil stark voneinander abweichende Messergebnisse mit beiden Methoden für eine Einzelmessung zu beobachten waren. Nutzt man die Messergebnisse beider Messmethoden in der in [BAH10] vorgestellten Weise, so weicht das Resultat stärker vom realen Wert ab, als wenn nur der Wert der Messmethode betrachtet würde, welcher die geringere Abweichung hat (Vgl. Abbildung 4.9, Messpunkt bei *25 m*). Deshalb wird hier eine andere hybride Methode vorgestellt, welche auf Grundlage aller für eine Einzelmessung verfügbarer Parameter entscheidet, ob ein bestimmtes Einzelmessergebnis den wahren Abstand zwischen Sender- und Empfängerknoten repräsentieren kann oder nicht. Dazu werden alle vom Transceiver zur Verfügung gestellten Informationen über die Signalqualität *(LQI)* und die Signalstärke *(RSSI)* verwendet, um die Messwerte der *RToF*-Messung zu bewerten.

Die Nachbearbeitung der Einzelmesswerte erfolgt in zwei Stufen. In der ersten Stufe werden die Einzelmesswerte anhand des *LQI* Wertes korrigiert und gefiltert. In der zweiten Stufe wird die aus dem *RSSI* Wert errechnete Entfernung mit der Entfernung aus der *RToF* – Messung korreliert. Der *RToF* -Messwert repräsentiert die Laufzeit zweier Pakete, welche unterschiedliche Wege vom Sende- zum Empfängerknoten gewählt haben können. Deshalb wird der *RToF* - Messwert in zwei gleiche Teile aufgeteilt. Dadurch wird der Abstandsmesswert für das Anforderungspaket und das Antwortpaket getrennt nachbearbeitet. Dabei ist es möglich, dass am Ende der Nachbearbeitung die Messwerte für beide Pakete, für keines oder nur für eines der beiden Pakete als Messergebnis berücksichtigt werden.

Ziel der ersten Stufe ist es die Streuung der *RToF* – Messwerte zu verringern. Dazu gilt es zu beachten das die Streuung der Messwerte nicht nur durch *multipath* Effekte hervorgerufen wird, sondern der Transceiver selbst bei augenscheinlich konstanten Bedingungen, wie sie beim Freifeldexperiment herrschten, unterschiedliche Zeitintervalle zur Decodierung des Funkpaketes benötigte. Dieser Umstand ist auf ein variables *propagation delay* des Transceivers zurückzuführen. Da die benötigte Zeit zum decodieren des Funkpaketes von der Signalqualität abhängt, wurden die Einzelmessergebnisse anhand des *LQI* Messwertes nachbearbeitet. Dazu wird über die letzten *50* Pakete der am häufigsten auftretende *LQI* Messwert bestimmt. Dieser Modus der *LQI* - Messwerte ist die Basis der Korrektur der *RToF* – Messwerte. Es wird davon ausgegangen, dass wenn der aktuelle *LQI* – Messwert unterhalb des momentanen Modus liegt, die Zeit zum decodieren des Funkpaketes höher ist, deshalb wird der Abstand zu groß berechnet. Ist der *LQI* – Messwert entsprechend höher als der Modus, so ist diese Zeit niedriger, und folglich wird der Abstand zu klein berechnet. Um diese Zeitdifferenzen zu kompensieren, wird die prozentuale Abweichung des aktuellen *LQI* Wertes vom Modus der letzten *50 LQI* Werte für den aktuellen *RToF* Messwert übernommen. Entspricht beispielsweise der aktuelle *LQI* Wert *93 %* des Modus, so wird für die Abstandsberechnung nur *93 %* der Zeit aus der *RToF* – Messung verwendet. Somit wird die in diesem Beispiel längere Zeit für die Demodulierung, hervorgerufen durch die schlechtere Signalqualität, für die Abstandsbrechung, kompensiert. Weiterhin wird nach der Korrektur eine Auswahl, der für die Abstandsberechnungen verwendeten Einzelmessergebnisse, getroffen. Dabei werden Einzelmessergebnisse verworfen, bei welchen eine, im Vergleich zum Modus der letzten *50 LQI* Werte, besonders schlechte oder auch eine besonders gute Signalqualität gemessen wurde. Dadurch wird erreicht, dass Funkpakete welche durch *multipath* Effekte beeinflusst wurden, also nicht den direkten Weg zwischen Sende- und Empfängerknoten gewählt haben, ausgefiltert werden. Es soll dadurch die Abweichung, welche die Laufzeitmessung in Innenräumen aufweist, korrigiert werden.

Die zweite Stufe der Nachbearbeitung korrigiert auf der Grundlage des *RSSI* Wertes die *RToF* – Einzelmesswerte wie folgt. Zuerst wird mit der Gleichung für die Freiraumdämpfung (1.2) der resultierende Abstand für den aktuellen *RSSI* Wert bestimmt. Danach wird der Mittelwert zwischen dem aus der ersten Nachbearbeitungs-

stufe erhaltenen Abstand der *RToF* – Messung und dem Abstand aus dem *RSSI* Wert gebildet. Da bei der Berechnung des Abstandes auf Grundlage des *RSSI* - Wertes von Freiraumdämpfung ausgegangen wurde, wird dieser mit steigender Entfernung, gerade in Innenräumen, zu groß. Deshalb wird dieser Abstand bei der Mittelwertberechnung mit steigender Größe prozentual geringer bewertet. Ist der Abstand aus der *RToF* – Messung deutlich größer als der Abstand welcher aus dem *RSSI* – Wert resultiert, so wird diese Einzelmessung als nicht plausibel verworfen, da es sich bei dieser Konstellation nicht um eine Messung handelt, welche den Abstand zwischen Sender- und Empfängerknoten repräsentiert.

In einem abschließenden Schritt werden die Einzelmessungen zur Eliminierung von Ausreißern mit einem *running median* gefiltert.

4.3 Auswertung des Lokalisierungsversuchs

Die Auswertung des Lokalisierungsversuchs erfolgte mit einer von Thomas Hillebrandt entwickelten Anwendung, welche im Rahmen des *FeuerWhere* Projektes zur graphischen Überprüfung der entwickelten Lokalisierungsalgorithmen dient. Dazu wurden, die auf den SD – Karten gespeicherten Abstände, zu den Ankerknoten eingelesen und im Falle der mit dem *NanoPAN* Transceiver ermittelten Abstandsdaten mit einem Medianfilter vor-verarbeitet. Für die mit dem *CC1101* Transceiver ermittelten Abstandsdaten wurde eine solche Filterung schon auf dem lokalisierenden Sensorknoten durchgeführt. Deshalb werden diese Daten in der Anwendung nicht nachbearbeitet. Danach erfolgte für jedes Abstands – Tripel, welches drei gültige Messwerte enthält, die Berechnung der relative Position zu den Ankerknoten. Ungültige Messwerte entstehen durch Einzelmessversuche, in welchen keine Funkverbindung zum Ankerknoten hergestellt werden konnte. Für die Berechnung der Position wurde ein bislang unveröffentlichter geometrischer Lokalisierungsalgorithmus von Thomas Hillebrandt verwendet, welcher auf Grundlage von approximierten Kreisschnittpunkten auch bei zu kurz gemessenen Abständen gute Ergebnisse liefert. Die ermittelten Positionen werden anhand der zuvor festgelegten Ankerpositionen in die in die An-

wendung geladene Karte als blauer Kreis eingezeichnet. In den folgenden beiden Abbildungen sind Bildschirmphotographien der Programmausgabe beider Versuche des Lokalisierungexperimentes dargestellt, wobei der im Versuch zurückgelegte Pfad als grüne Linie markiert ist. Abbildung 4.10 Zeigt die Ergebnisse der Messungen mit dem *CC1101* Transceiver.

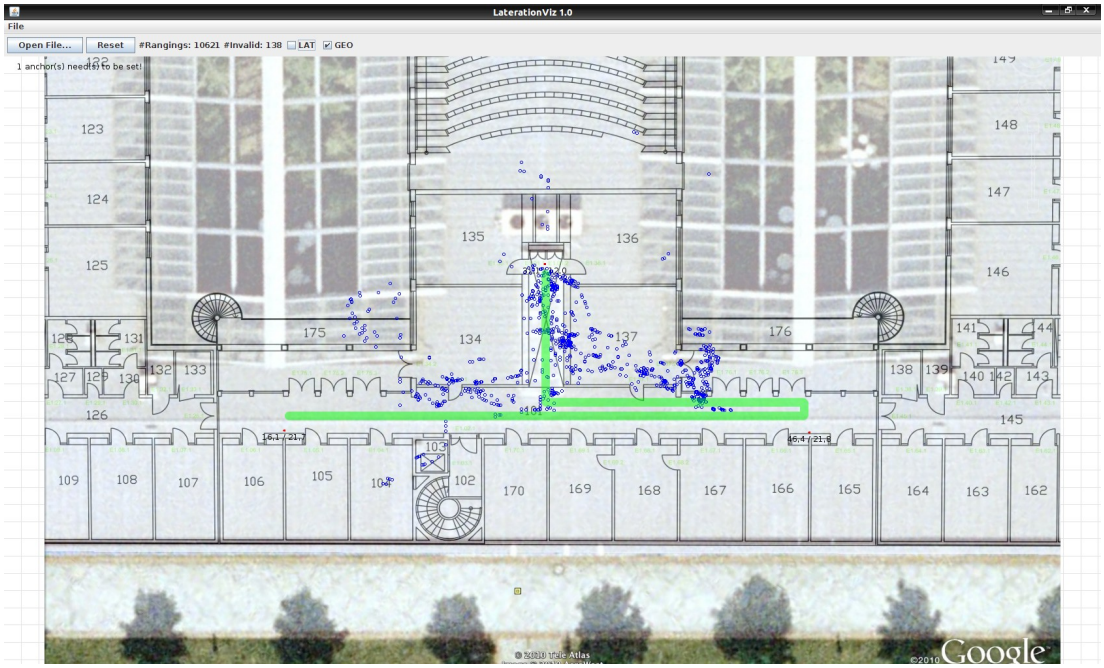

Abbildung 4.10 Ergebnis des Lokalisierungsexperimentes mit dem CC1101 Transceiver.

In diesem Versuch liegt die Abweichung vom real gelaufen Pfad zwischen *2,5 m* und *5 m*, wobei die Streuung der Positionen mit rund *5 m,* gemessen an der tatsächlichen Messauflösung der verwendeten Hardware, ebenfalls gering ausfällt. Es ist möglich, anhand der aufgezeichneten Punkte den gelaufenen Pfad zu rekapitulieren. Es wurden während des Versuchs über *10000* Messungen zu drei Ankerknoten durchgeführt, von denen nur *138* Einzelmessungen ungültig waren. Weiterhin konnte bei langsamer, nacheinander erfolgender Einzeichnung der Einzelpositionen durch die Anwendung festgestellt werden, dass ermittelte aufeinanderfolgende Einzelpositionen nicht mehr als *5 m* von einander entfernt waren. Es ist ersichtlich, dass bei vielen Messergebnissen die Abstände zu kurz berechnet wurden, was eine Überkorrektur vermuten lässt.

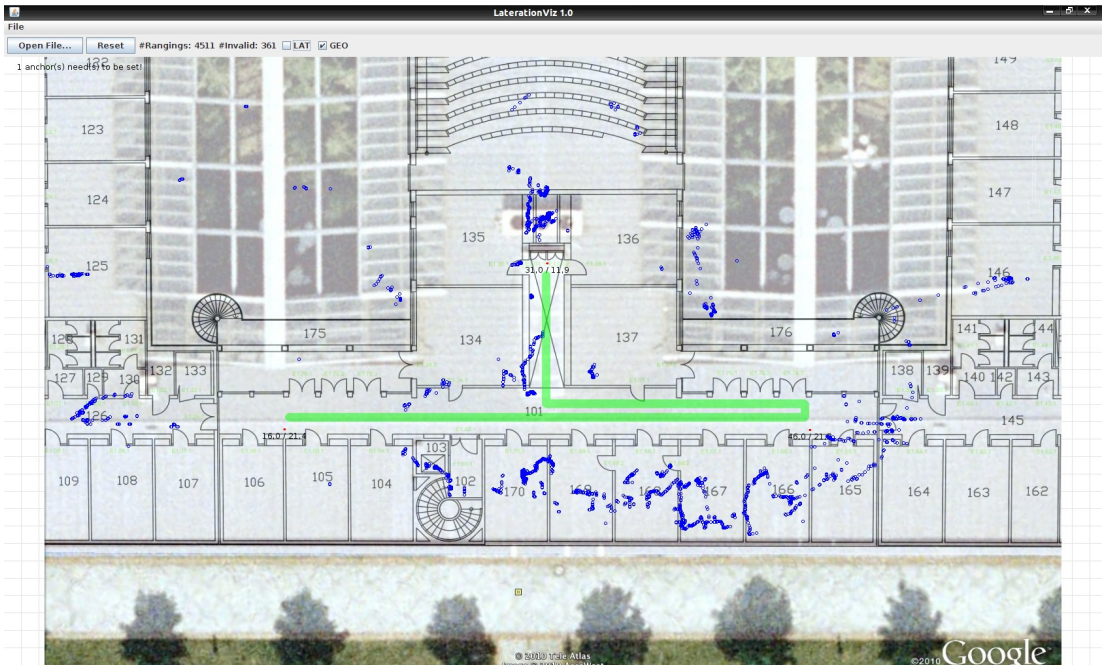

Abbildung 4.11 Ergebnis des Lokalisierungsexperimentes mit dem NanoPAN Transceiver.

Abbildung 4.11 zeigt die Ergebnisse der Messungen mit dem *NanoPAN* Transceiver. In diesen Versuch konnten, bei annähernd gleicher Versuchsdauer rund *4500* Messungen durchgeführt werden, wobei bei rund *2000* Messungen mindestens ein Ankerknoten nicht erreichbar war. Die Abweichung und Streuung der Abstandswerte ist im Vergleich zum in dieser Arbeit entwickelten Laufzeitmesssystem größer. Betrachtet man wiederum die einzelnen Messpunkte nacheinander, so sind große Sprünge zwischen den Einzelmessergebnissen zu verzeichnen, welche mitunter auch über *10 m* entgegen der Bewegungsrichtung erfolgen können. Die errechnete Position auf dem Weg vom Sensorknoten *4* zum Sensorknoten *2* springt beispielsweise abwechselnd von den Räumen in Bewegungsrichtung links des Flurs und dem in Bewegungsrichtung rechts liegenden Innenhof hin und her. In der Mehrzahl der Einzelmessungen wird der Abstand zu den Ankern zu lang berechnet, was lässt auf eine mangelhafte Erkennung und Kompensation von indirekten Messungen schließen lässt. In einigen Fällen wird der Abstand auch zu kurz berechnet, was auf ein auch bei diesem Transceiver nicht konstantes *propagation delay* schließen lässt. Die Rekapitulation des gelaufenen Pfades ist, aufgrund der großen Streuung, anhand der Messergebnisse des *NanoPAN* Transceivers deutlich schwieriger als bei den Ergebnissen welche mit dem *CC1101* Transceiver erzielt wurden.

4.4 Energiebedarf der Laufzeitmessung

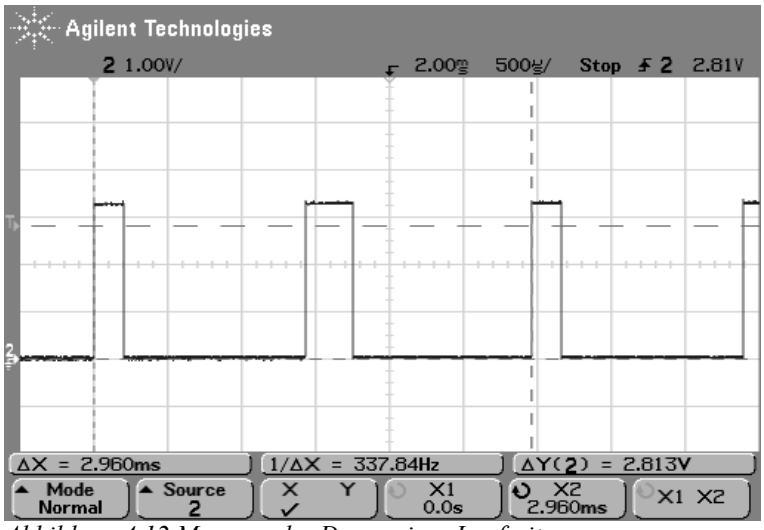

Abbildung 4.12 Messung der Dauer einer Laufzeitmessung

Der maximale Energieverbrauch Q_{ges} lässt sich für eine Einzelmessung, aufgrund der in den Datenblättern des Mikrocontrollers [NXP08] und des Transceivers [TEX09] angegebenen Strombedarfswerte und der Dauer der Sende- und Empfangszyklen während der Messung, näherungsweise ermitteln. Für die Messung der Zeitdauer der Sende- und Empfangszyklen wurde eine typische Laufzeitmessung am *GDO2* – Pin des anfordernden Sensorknotens, oszillographiert. In Abbildung 4.12 ist diese Messung dargestellt, wobei die kurzen Intervalle, an denen ein hoher Pegel anlag, die Zeit des Sendens des Anforderungspaketes mit einer Nutzdatenmenge von *8 Byte*, zeigen. Das entsprechend längere Intervall bestimmt die Empfangszeit des Antwortpaketes mit einer Nutzdatenmenge von *14 Bytes*. Da der antwortende Sensorknoten einen höheren Stromverbrauch, aufgrund der größeren Datenmenge welche im Antwortpaket versendet werden muss, hat, wird dieser zur Berechnung des maximalen Energieverbrauchs herangezogen.

Der Mikrocontroller ist während der gesamten Messung, wegen der benötigten Hardwaretimer, voll aktiv, wobei der Transceiver bei beiden an der Messung beteiligten Sensorknoten bis auf die kurze *PLL* – Kalibrierungszeitt t_{cal} und die Sendezeit t_{tx} im Empfangsmodus ist.

$$Q_{CPU} = I_{CPU} \cdot t_{ges} \tag{4.2}$$

$$Q_{CC1100} = I_{CC1100RX} \cdot (t_{ges} - t_{tx} - t_{cal}) + I_{CC1100CAL} \cdot t_{cal} + I_{CC1100TX} \cdot t_{tx} \tag{4.3}$$

$$Q_{ges} = Q_{CPU} + Q_{CC1100} \tag{4.4}$$

Setzt man den Strombedarf der einzelnen Modi des Transceivers und den des Mikrocontrollers und die benötigten Zeiten, welche mit dem Oszillographen ermittelt wurden (Vgl. Abbildung 4.12), ein so ergibt sich folgender Wert für den Energieverbrauch einer einzelnen Messung:

$$Q_{CPU} = 63\,mA \cdot 2{,}96\,ms = \mathbf{0{,}18648\,mAs} \tag{4.5}$$

$$\begin{aligned} Q_{CC1100} &= 16{,}4\,mA \cdot (2{,}96\,ms - 0{,}32\,ms - 1{,}23\,ms) \\ &+ 8{,}2\,mA \cdot 1{,}23\,ms + 31{,}1\,mA \cdot 0{,}32\,ms = \mathbf{0{,}0432\,mAs} \end{aligned} \tag{4.6}$$

$$Q_{ges} = \mathbf{0{,}2279\,mAs} \tag{4.7}$$

Somit kann man, eine Batteriekapazität von *2000 mAh* vorausgesetzt, über *200 000* Laufzeitmessungen mit drei Ankern und *50* Einzelmessungen pro Messung durchführen.

5 Fazit

In dieser Arbeit wurde die Entwicklung und Analyse eines Abstandmesssystems auf der Basis eines preiswerten Serientransceivers und eines Mikrocontrollers mit hoher Taktfrequenz vorgestellt. Dieses Messsystem stütz sich auf hybride Methoden. Damit können die Abweichungen, welche unter schwierigen Bedingungen, die zum Beispiel in Innenräumen auftreten minimiert werden.

Einleitend wurden die Methoden, die zur Abstandsbestimmung zwischen Sensorknoten dienen, erläutert und deren physikalischen Grenzen aufgezeigt. Im Kapitel 2 wurden die für das Laufzeitmesssystem verwendeten Hard- und Softwarekomponenten beschrieben und die theoretisch erreichbare Präzision erörtert. Es erfolgte, wie im dritten Kapitel erläutert, eine messtechnische Analyse des Zeitverhaltens der Hard- und Softwarekomponenten und darauf aufbauend die Implementierung des Laufzeitmesssystems. Es wurden zur Analyse und Evaluierung des entwickelten Laufzeitmesssystems mehrere Experimente in unterschiedlichen Umgebungen bis hin zu einem praxisnahen Lokalisierungsversuch durchgeführt. Der Aufbau und die Durchführung dieser Experimente wurden im dritten Kapitel beschrieben. Abschließend wurden die aus den Experimenten gewonnenen Daten vorgestellt und analysiert und im Fall des Lokalisierungsexperimentes mit den Ergebnissen eines kommerziellen Produktes verglichen.

Es konnte gezeigt werden, dass mit stochastischen Methoden die theoretisch mögliche Auflösung, welche durch die verwendeten Hardwarekomponenten vorgegeben wird, verbessert werden kann. Auf dieser Grundlage konnte im Freifeld bei Verwendung der *RToF* – Methode der Abstand zwischen zwei Sensorknoten mit einer Abweichung von maximal *5 %* und in *90 %* der Fälle mit *2,6 %,* bezogen auf die Länge der Messstrecke bestimmt werden. Weiterhin konnte im Vergleich mit dem *NanoPAN*

Transceiver bewiesen werden, dass es mit preiswerten Standardhardwarekomponenten möglich ist, unter Verwendung von hybriden Messmethoden, eine geringere Abweichung und Streuung bei der Lokalisierung im Innenraumbereich zu erreichen als sie teurere kommerzielle Produkte aufweisen. Es wurde die minimal benötigte Anzahl an Funkpaketen zur Abstandsbestimmung analysiert. Für die Lokalisierung von statischen Objekten im Freifeld liegt diese bei *50* Funkpaketen. Die maximale Entfernung ist dabei nur durch die Funkreichweite des Sensorknotens beschränkt. Bezogen auf die Energieressourcen eines Sensorknotens ist die benötigte Anzahl an Funkpaketen realisierbar.

Die Skalierbarkeit des Systems ist aufgrund der benötigten Paketanzahl bei Echtzeitlokalisierung von bewegten Objekten im Innenraumbereich gering. Dieser Sachverhalt ist bei allen auf der *RToF* – Messmethode basierenden Systemen gegeben, da bei der Echtzeitlokalisierung das Übertragungsmedium dauerhaft belegt ist.

Da die verschiedenen *CC110x* Transceiver durch Fertigungsschwankungen ein unterschiedliches *propagation delay* aufweisen, bietet die Untersuchung der in dieser Arbeit vorgestellten Implementierung mit anderen Transceivern Stoff für weitere Arbeiten. So kann die Verwendung eines höher getakteten Transceivers, welcher eine automatische Beantwortung von Anfragepaketen unterstützt, die Abweichungen der Abstandsbestimmung minimieren. Weiterhin kann die Verwendung des selben Oszillators für Transceiver und Mikrocontroller die Präzision der Zeitmessungen verbessern. Ein entsprechendes Design für eine Hardwareplattform ist am Institut bereits entstanden, konnte aber im Rahmen dieser Arbeit nicht mehr untersucht werden. Zur Verbesserung der Skalierbarkeit ist die Änderung der vorgestellten Implementierung zur *TDoA* – Methode möglich. Die bestehende Implementierung müsste um eine präzise Zeitsynchronisation der als Anker dienenden Sensorknoten erweitert werden. Zur Realisierung der *TDoA* - Methode tauschen Ankerknoten und der zu lokalisierende Sensorknoten die Rollen, so dass ein Ankerknoten die Messung initiiert und der zu lokalisierende Sensorknoten auf die Anfrage per *broadcast* antwortet. Diese Antwort kann von allen in Reichweite befindlichen Ankern zur Berechnung der Position verwendet werden, so dass diese keine erneute Anfrage initiieren müssen. Dadurch

kann die benötigte Gesamtpaketanzahl verringert werden und es können mehr Sensorknoten pro Ankerknoten als mit der *RToF* – Messung lokalisiert werden.

Die Kombination aller Informationen, welche der Transceiver zur Signalqualität und -Stärke bereitstellt zu einer hybriden Messmethode, hat gezeigt dass auch in Innenräumen die Approximation der Entfernung zwischen den Sensorknoten möglich ist. Es könnten durch weitere Analysen der Signalparameter in weiterführenden Arbeiten bessere Methoden zur Erkennung von indirekten Abstandsmessungen entwickelt werden. Damit würde die Approximation der Entfernung präziser werden. Abschließend kann festgestellt werden, dass die zu Beginn der Arbeit erwartete Präzision der Abstandsbestimmung übertroffen werden konnte.

Literaturverzeichnis

[BAA08] Baar, M., Will, H., Blywis, B., Hillebrandt, T., Liers, A., Wittenburg, G., Schiller, J., The ScatterWeb MSB-A2 Platform for Wireless Sensor Networks, 2008,

[BAH10] A. Bahillo, S. Mazuelas, R. M. Lorenzo, et al., Hybrid RSS-RTT Localization Scheme for Indoor Wireless Networks, EURASIP Journal on Advances in Signal Processing, 2010

[CHI03] Charles Chien, Location Position System For Relay Assisted Tracking, US Patent 6,512,478, 2003

[DEZ10] Norman Dziengel, Marco Ziegert, Zakaria Kasmi, Frederik Hermans, Stephan Adler, Georg Wittenburg, and Jochen Schiller, A Platform for Distributed Event Detection in Wireless Sensor Networks, First International Workshop on Networks of Cooperating Objects (CONET 2010), 2010

[FEU10] Freie Universität Berlin, FeuerWhere – Tracking Fire Fighters., Oktober, 2010, http://cst.mi.fu-berlin.de/projects/FeuerWhere

[FUJ09] Fujiwara, Ryosuke and Mizugaki, Kenichi and Nakagawa, Tatsuo and Maeda, Daisuke and Miyazaki, Masayuki, TOA/TDOA hybrid relative positioning system using UWB-IR, , 2009

[GRO07] Ralf Grossmann, Jan Blumenthal, Frank Golatowsk, Dirk Timmermann, Localization in Zigbee-based Sensor Networks, 1st European ZigBee Developer´s Conference (EuZDC), 2007

[HAR99] Andy Harter, Andy Hopper, Pete Steggles, Andy Ward, Paul Webster, The Anatomy of a Context-Aware Application, Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking, 1999

[HIL07] Thomas Hillebrandt, Untersuchung und Simulation des Zeit- und Energieverhaltens eines MSB430-H Sensornetzwerkes, 2007

[JEN09] Mathias Jensen, Henrik Vatnar, DN010 -- Close-in Reception with CC1101 (Rev. B), 2009, http://focus.ti.com/lit/an/swra147b/swra147b.pdf

[KÖP10] Enrico Köppe, Heiko Will, Achim Liers, Jochen Schiller, Tracking Persons with an Autarkic Radio-Based Multi-Sensor System, IPIN, 2010

[LAS04] Laskar, Joy; Matinpour, Babak; Chakraborty, Sudipto, Modern Receiver Front-ends, 2004

[LOP10] Freddy López Villafuerte, Localization of Wireless Sensor Nodes Based on Local Network Density, 2010

[NAK08] Tatsuo Nakagawa, Masayuki Miyazaki, Goichi Ono, Ryosuke Fujiwara, Takayasu Norimatsu, Takahide Terada, Akira Maeki, Yuji Ogata, Shinsuke Kobayashi, Noboru Koshizuka, and Ken Sakamura, 1-cc Computer using UWB-IR for Wireless Sensor Network, Proceedings of the 2008 Asia and South Pacific Design Automation Conference, 2008

[NAM07] Siri Namtvedt, DN506 -- GDO Pin Usage, 2007, http://focus.ti.com/lit/an/swra121a/swra121a.pdf

[NAN09] Nanotron Technologies GmbH, nanoPAN 5375 RF Module - High Performance 2.4 GHz Transceiver, 2009, http://www.nanotron.com/EN/pdf/Factsheet_nanoPAN_5375_RF_Module.pdf

[NXP08] NXP Semiconductors, LPC2387 Single-chip 16-bit/32-bit MCU; 512 kB flash with ISP/IAP, Ethernet, USB 2.0 device/host/OTG, CAN, and 10-bit ADC/DAC, 2008, http://www.nxp.com/documents/data_sheet/LPC2387.pdf

[NXP09] NXP Semiconductors, UM10211 LPC23XX User manual (Rev. 03), 2009, http://ics.nxp.com/support/documents/microcontrollers/pdf/user.manual.lpc23xx.pdf

[PRI03] N. B. Priyantha, H. Balakrishnan, E. Demaine, S. Teller, Anchor-Free Distributed Localization in Sensor Networks,MIT Laboratory for Computer Science, 2003

[SCH03] Jochen H. Schiller, Mobile Communications (Second Edition), 2003

[SHA08] Sharma, P. Krishnan, S. Zhang Guoping, A multi-cell UWB-IR localization system for robot navigation, Radio and Wireless Symposium, 2008 IEEE , 2008

[SHA49] Claude Elwood Shannon, Communication in the Presence of Noise, Proc. IRE, Vol. 37, No. 1 (Jan. 1949), , Nachdruck in: Proc. IEEE, Vol. 86, No. 2, (Feb. 1998), 1949

[TEX08] Texas Instruments, DN018 -- Range Measurements in an Open Field Environment (Rev. A), 2008,

[TEX08a] NXP Semiconductors, LPC2387 Single-chip 16-bit/32-bit MCU; 512 kB flash with ISP/IAP, Ethernet, USB 2.0 device/host/OTG, CAN, and 10-bit ADC/DAC, 2008, http://www.nxp.com/documents/data_sheet/LPC2387.pdf

[TEX09] Texas Instruments, Single-Chip Low Cost Low Power RF-Transceiver (Rev. D), 2009, http://focus.ti.com/lit/ds/symlink/cc1100.pdf

[TEX10] Texas Instruments, DN505 -- RSSI interpretation and timing (Rev. D), 2010, http://focus.ti.com/lit/an/swra114d/swra114d.pdf

[WIB09] Sigit Basuki Wibowo, Martin Klepal, Dirk Pesch, Time of Flight Ranging using Off-the-self IEEE802.11 WiFi Tags, POCA 2009: Positioning and Context Awareness International Conference Antwerp, Belgium, 2009

[WID08] Widyawan, Martin Klepal, Ste´phane Beauregard, A novel backtracking particle filter for pattern matching indoor localization, Proceedings of the first ACM international workshop on Mobile entity localization and tracking in GPS-less environments, 2008

[WIL08] Heiko Will, Entwurf und Implementierung einer Multi-Plattform Sensornetz-Architektur, 2008

Anhang A Konfigurationen des CC1100 Transceivers

Konfiguration: MSK, 400 kbps

```
uint8_t cc1100_conf[] =
{
        0x06, // IOCFG2
        0x2E, // IOCFG1
        0x0E, // IOCFG0
        0x0F, // FIFOTHR
        0x9B, // SYNC1
        0xAD, // SYNC0
        0x3D, // PKTLEN
        0x06, // PKTCTRL1
        0x45, // PKTCTRL0
        0xFF, // ADDR
        0x00, // CHANNR
        0x0B, // FSCTRL1
        0x00, // FSCTRL0
        0x21, // FREQ2
        0x71, // FREQ1
        0x7A, // FREQ0
        0x2D, // MDMCFG4
        0xF8, // MDMCFG3
        0x73, // MDMCFG2
        0x42, // MDMCFG1
        0xF8, // MDMCFG0
        0x00, // DEVIATN
        0x07, // MCSM2
        0x03, // MCSM1
        0x18, // MCSM0
        0x1D, // FOCCFG
        0x1C, // BSCFG
        0xC0, // AGCCTRL2
        0x49, // AGCCTRL1
        0xB2, // AGCCTRL0
        0x87, // WOREVT1
        0x6B, // WOREVT0
        0xF8, // WORCTRL
        0xB6, // FREND1
        0x10, // FREND0
        0xEA, // FSCAL3
        0x2A, // FSCAL2
        0x00, // FSCAL1
        0x1F, // FSCAL0
        0x00 // padding to 4 bytes
};
```

Konfiguration: OOK, 200 kbps

```
char cc1100_conf_ook[] =
{
        0x06, // IOCFG2
        0x2E, // IOCFG1
        0x0E, // IOCFG0
        0x0F, // FIFOTHR
        0x9B, // SYNC1
        0xAD, // SYNC0
        0x3D, // PKTLEN
        0x06, // PKTCTRL1
        0x45, // PKTCTRL0
        0xFF, // ADDR
        0x0B, // FSCTRL1
        0x00, // FSCTRL0
        0x21, // FREQ2
        0x71, // FREQ1
        0x7A, // FREQ0
        0x2C, // MDMCFG4
        0xF8, // MDMCFG3
        0x33, // MDMCFG2
        0x42, // MDMCFG1
        0xF8, // MDMCFG0
        0x00, // DEVIATN
        0x07, // MCSM2
        0x03, // MCSM1
        0x18, // MCSM0
        0x1D, // FOCCFG
        0x1C, // BSCFG
        0x04, // AGCCTRL2 OOK 0x03 to 0x07
        0x00, // AGCCTRL1,OOK 0x00
        0x92, // AGCCTRL0 OOK 0x91 0x92
        0x87, // WOREVT1
        0x6B, // WOREVT0
        0xF8, // WORCTRL
        0xB6, // FREND1
        0x11, // FREND0
        0xEA, // FSCAL3
        0x2A, // FSCAL2
        0x00, // FSCAL1
        0x1F, // FSCAL0
        0x00 // padding to 4 bytes
};
```