

Sensor Integration

Studienarbeit

In den Studiengängen Luft- & Raumfahrt elektronik
und Nachrichten- & Kommunikationstechnik
an der DHBW Ravensburg
Campus Friedrichshafen

von

Malte Rahm

Daniel Voggel

Sascha Rupp

08.07.2015

Bearbeitungszeitraum	01.10.2014 bis 19.06.2015
Matrikelnummer	3995993, 9382283 und 6644854
Betreuer	Prof. Dr. Thomas Mannchen

Erklärung

gemäß § 5 (3) der „Studien- und Prüfungsordnung DHBW Technik“ vom 22. September 2011.

Hiermit erkläre ich, dass ich die vorliegende Arbeit mit dem Titel

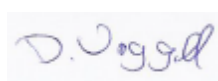
Sensor Integration

selbständig angefertigt, nicht anderweitig zu Prüfungszwecken vorgelegt, keine anderen als die angegebenen Hilfsmittel benutzt und wörtliche sowie sinngemäße Zitate als solche gekennzeichnet habe.

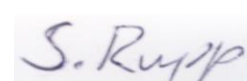
Friedrichshafen, den 08.07.2015



Malte Rahm



Daniel Voggel



Sascha Rupp

Kurzfassung

Kurzfassung

Um ein funktionierendes Unmanned Aerial System (UAS) zu entwickeln und zu betreiben, sind zahlreiche vorwiegend elektronische Subsysteme nötig. Die vorliegende Studienarbeit befasst sich mit den elektronischen Subsystemen eines Multicopters der als Teil eines Schwarms agieren soll.

Dabei wurde unter anderem viel Konzeptarbeit geleistet, um ein schwarmfähiges UAS zu entwickeln. Zu einem UAS gehören in diesem Fall mehrere Multicopter sowie eine Bodenstation. In Kooperation mit den anderen Subteams wurden zunächst grundlegende Architekturkonzepte erstellt. Diese wurden nach und nach detaillierter, sodass zuletzt passendes Equipment ausgewählt und miteinander vernetzt werden konnte. Dabei mussten diverse Komponenten angepasst und entsprechend so konfiguriert werden, dass die Subsysteme auch als Gesamtsystem funktionieren. Da alle Subsysteme Energie benötigen, wurde zusätzlich noch ein Power Management konzeptioniert und realisiert.

Ergebnis dieser Studienarbeit ist eine funktionstüchtige Multicopterplattform, auf die zukünftige Generationen aufbauen können.

Abstract

Abstract

To operate and develop an Unmanned Aerial System (UAS) many subsystems are necessary. This seminar paper deals with the electronic subsystems of a multicopter that shall operate as a member in a swarm.

Thereby a lot of conceptional work has been performed to develop a UAS that can operate as a multi-drone-swarm. In this case the UAS consists of a number of multicopters and a groundstation. At first basic system-architectures were developed in cooperation with the additional subteams. This basic architectures became more and more specific. As a derivation matching equipment was chosen and linked together. To provide the function not only of the subsystems but also of the complete system many components had to be adjusted and respectively configured. Because all subsystems have to be supplied with energy a power management was designed and implemented additionally.

Outcome of this student research project is a fully functional multicopterplatform which can be used as basis for future generations and swarm projects.

Inhaltsverzeichnis

Erklärung.....	II
Kurzfassung.....	III
Abstract.....	IV
Inhaltsverzeichnis	V
Abbildungsverzeichnis	VII
1 Einleitung.....	1
2 Zielsetzung und Aufgabenstellung.....	4
3 Systemarchitektur	5
3.1 Grundidee	5
3.2 Architekturkonzept.....	7
3.3 Architektur des Multicopters	8
3.4 Architektur der Bodenstation	11
4 Subsysteme	12
4.1 Fernsteuersystem.....	12
4.1.1 HoTT	12
4.1.2 MC32	13
4.1.3 GR32.....	14
4.1.4 Lehrer-Schüler-Modus	15
4.2 Ardupilot Mega	16
4.3 Der Beagle Bone Black	18
5 Integration der Baugruppen	19
5.1 BeagleBoneBlack neu aufsetzen.....	19
5.2 Installation des WLAN-Sticks	20
5.2.1 Installation des Treibers des WLAN-Sticks	20
5.2.2 Einrichtung des Netzwerkes.....	21
5.3 Der BeagleBoneBlack als Bridge	21
5.3.1 Installation von Mavproxy.....	22

Inhaltsverzeichnis

5.3.2	<i>Inbetriebnahme von Mavproxy</i>	22
5.4	<i>Implementierung der APM-PCC Schnittstelle auf dem PCC</i>	24
5.4.1	<i>Das MAVLink Protokoll</i>	25
5.4.2	<i>Implementierung</i>	26
5.5	<i>Anschluss des Sonars.....</i>	28
5.6	<i>Implementierung der Bodenstation.....</i>	29
6	Verbesserungen und Ausblick.....	31
6.1	Optimierung der Spannungsversorgung.....	31
6.2	EMV	35
6.3	Flugleistung.....	37
6.4	Flugrechner	38
7	Ergebnis	40
8	Literaturverzeichnis	41

Abbildungsverzeichnis

<i>Abbildung 1: Kommunikationskonzept von Autonomous Flight</i>	<i>5</i>
<i>Abbildung 2: Architekturkonzept für den Multikopter</i>	<i>7</i>
<i>Abbildung 3: Architekturkonzept für die Bodenstation</i>	<i>8</i>
<i>Abbildung 4: Architekturkonzept für den Multikopter</i>	<i>9</i>
<i>Abbildung 5: Architektur der Bodenstation.....</i>	<i>11</i>
<i>Abbildung 6: MC32 HoTT Sender.....</i>	<i>13</i>
<i>Abbildung 7: GR32 HoTT Empfänger.....</i>	<i>14</i>
<i>Abbildung 8: Lehrer-Schüler-Modus des Funkfernsteuersystems</i>	<i>15</i>
<i>Abbildung 9: Platine mit den wichtigsten ICs und Anschlüssen des APM 2.6</i>	<i>16</i>
<i>Abbildung 10: grundlegender Aufbau des APM 2.6.....</i>	<i>17</i>
<i>Abbildung 11: Beagle Bone Black [11]</i>	<i>18</i>
<i>Abbildung 12: Aktionen zum Verbinden von Missionplanner und Mavproxy aus [6].</i>	<i>24</i>
<i>Abbildung 13: Elektrischer Anschluss des Sonars aus [3].....</i>	<i>28</i>
<i>Abbildung 14: Die APM-Parameter für das Sonar</i>	<i>29</i>
<i>Abbildung 15: Betriebsmodi der Bodenstation.....</i>	<i>30</i>
<i>Abbildung 16: Testaufbau für die Spannungsversorgung.....</i>	<i>32</i>
<i>Abbildung 17: Ausgangsspannung des Spannungsreglers unter Last</i>	<i>32</i>
<i>Abbildung 18: Spannungsverlauf bei Anschluss einer Kapazität</i>	<i>33</i>
<i>Abbildung 19: Modifiziertes Spannungsversorgungsmodul</i>	<i>34</i>
<i>Abbildung 20: Spannungsverlauf nach der Modifikation unter gleichen Bedingungen</i>	<i>34</i>
<i>Abbildung 21: Spannungsverlauf nach einbau im Hexacopter</i>	<i>35</i>
<i>Abbildung 22: Aufgespannte Schleifen im Konzept.....</i>	<i>36</i>
<i>Abbildung 23: USB Trennfilter</i>	<i>36</i>
<i>Abbildung 24: Provisorische Umsetzung zur Problemlösung</i>	<i>37</i>
<i>Abbildung 25: Konzept redundanter Flugrechner</i>	<i>39</i>

1 Einleitung

Wie bei vielen anderen wichtigen Erfindungen auch, liegen die Ursprünge der unbemannten Luftfahrt beim Militär. Bereits 1849 kamen österreichische Streitkräfte auf die Idee, Bomben per Ballon nach Venedig zu transportieren und abzuwerfen [1]. Diese zugegebenermaßen rudimentäre Art und Weise der Kriegsführung hat selbstverständlich wenig mit modernen Drohnen zu tun. Die ihr zugrunde liegende Problemstellung und die sich ergebenden Vorteile sind jedoch nicht allzu weit von denen der heutigen Einsatzgebiete entfernt: Unbemannte Flugobjekte sollen den Menschen entlasten oder Aufgaben durchführen, zu denen ein Mensch gar nicht fähig wäre. Ginge es den alpenländischen Invasoren im 19. Jahrhundert noch darum möglichst viele Leben der eigenen Soldaten zu schonen (ein Motiv, das sich auch heute noch wiederfindet, zum Beispiel in den von den US-amerikanischen Streitkräften geführten Drohnenkriegen im Nahen Osten), so lassen sich *Unmanned Aerial Vehicles* (UAVs) heutzutage für eine schier unbegrenzte Anzahl an Anwendungen einsetzen. Am *Massachusetts Institute of Technology* beispielsweise, wurden Personen, die Gäste und Studenten über den Campus führen, durch Drohnen ersetzt, die sich vom Anwender per Smartphone-App rufen lassen, und dann per GPS und Bildverarbeitungssoftware zum Zielort navigieren [2]. Ähnlich futuristisch geht es vor den Küsten der Vereinigten Staaten zu, wo ferngesteuerte Drohnen zum Kampf gegen den Drogenschmuggel eingesetzt werden. Innerhalb der ersten zwei Wochen nach Einführung des Systems konnten über 600 kg Kokain sichergestellt werden [3]. Zu medialer Berühmtheit gelangte Amazons Vorhaben, Pakete innerhalb von maximal 30 Minuten seinen Kunden zuzustellen [4]. Dies soll ebenfalls durch den Einsatz von unbemannten und autonom fliegenden Drohnen erfolgen. Bis zur Markteinführung des Systems wird zwar noch etwas Zeit vergehen, aber der Anblick von selbstständig agierenden Flugobjekten in unseren Städten wird uns, in nicht allzu ferner Zukunft, unter Umständen genau so vertraut sein wie der eines Lieferwagens.

Auch in den Bereichen Forschung, Aufklärung, Natur- und Tierschutz, Journalismus, Search and Rescue, Filmproduktion und Landwirtschaft können durch den Einsatz von UAVs Aufgaben bearbeitet werden, die vom Menschen nicht oder nur mit ungleich größerem Aufwand bewerkstelligt werden könnten.

Einleitung

Die unbemannte Luftfahrt ist also ein verheißungsvolles Gebiet, wenn es darum geht, zukünftige Märkte zu erschließen. Dementsprechend sinnvoll ist es, dass Hochschulen ihre Studierenden frühzeitig an das Thema heranzuführen. Im Wintersemester 2014/15 wurde deshalb am Campus Friedrichshafen der Dualen Hochschule Baden-Württemberg Ravensburg das studentische Projekt Locator ins Leben gerufen. Zehn Studierende arbeiteten an dem Projekt und dokumentieren ihre Ergebnisse in dieser Reihe von Studienarbeiten. Die Bearbeitung fand im Wintersemester 2014/15 und im Sommersemester 2015 statt. Es waren die ersten sechs Monate des Projekts Locator, unterbrochen von drei Monaten Praxisphase. Dieser Zeitraum stellt lediglich den Anfang eines vermutlich mehrjährigen Projektes dar, das mehrere Generationen von Studenten beschäftigen wird. In dieser Anfangsphase wurde noch kein endgültiges Ziel bzw. Anwendungsgebiet definiert. Vorstellbar wären aber Einsätze in der Landvermessung oder experimentelle Flüge zur Entwicklung von Schwarmkonzepten und Erforschung der dahinter stehenden Intelligenz.

Der Plan für die Anfangsphase des Projektes war ein grundsätzliches Verständnis für die unbemannte Luftfahrt zu erlangen. Dazu sollte eine Commercial-Off-The-Shelf-Drohne so modifiziert werden, dass sie einen farbigen Ball komplett autonom verfolgen kann. Da die Ausgangsdrohne für ein solches Unterfangen nicht ausgelegt ist, mussten Änderungen an der Software, Hardware und Struktur vorgenommen werden. Professor Mannchen, der leitende Dozent des Projektes, entschied, dass diese Maßnahmen am besten von einer Kombination von Studierenden aus den Luft- und Raumfahrtssystemen (TLS), der Luft- und Raumfahrtelctronik (TLE) und der Nachrichten- und Kommunikationstechnik (TEN) durchgeführt werden sollen. Von den zehn beteiligten Studierenden gehörten zwei zu TLS, sechs zu TLE und zwei zu TEN. Das Projekt wurde in vier Untergruppen aufgeteilt: Autonomous Flight, Swarm Intelligence, Sensor Integration und Structural Integration. Ursprünglich waren die Aufgaben wie folgt definiert: Autonomous Flight sollte Missionen und Anforderungen an das Flugobjekt definieren und die Systemarchitektur für die Drohnen entwerfen. Die Aufgabe von Swarm Intelligence war es, Konzepte für das Zusammenarbeiten mehrerer Drohnen innerhalb eines Schwarms zu entwickeln. Das Team Sensor Integration befasste sich mit der Integration von den Sensoren und dem Datenlink, sowie dem Batteriemanagement

Einleitung

und der Sensorfusion. Structural Integration, schließlich, befasste sich mit dem mechanischen Aufbau, der Konstruktion und Integration der Drohnen. Die Grenzen zwischen den Aufgaben verschwammen jedoch – speziell am Anfang des Projektes – zwischen den einzelnen Gruppen, da einige Aufgaben mit höherer Priorität bearbeitet werden mussten, oder, da die Bearbeitung bestimmter Aufgaben nicht ohne die Resultate anderer Arbeitspakete erledigt werden konnten, sodass durch die Verlagerung von „Manpower“ bestimmte Aufgaben beschleunigt wurden.

Die einzelnen Kapitel dieser Studienarbeit wurden dabei von den folgenden Personen geschrieben:

Malte Rahm:	3.3; 4.2; 5
Daniel Voggel:	2; 3; 4.1; 4.3; 6; 7
Sascha Rupp:	2; 3; 4.1; 4.2; 4.3; 5.4.2; 6; 7

2 Zielsetzung und Aufgabenstellung

Diese Studienarbeit befasst sich primär mit den Subsystemen des Multicopters.

Aus den Anforderungen der anderen Subteams soll eine Systemarchitektur für fliegendes Gerät sowie eine Bodenstation entwickelt werden, die Komponenten enthält, um den Anforderungen gerecht zu werden. Diese Komponenten sollen ausgewählt, beschafft und anschließend mit einander vernetzt werden.

Da alle Subsysteme Energie benötigen ist außerdem ein Power-Management-Konzept zu erstellen und umzusetzen. Zusätzlich sollen die Datenlinks zwischen den Subsystemen, sowie der Datenlink zur Bodenstation realisiert werden.

3 Systemarchitektur

In diesem Kapitel ist die schrittweise Entwicklung der Systemarchitektur näher ausgeführt.

3.1 Grundidee

Zusammen mit den anderen Subteams wurde folgendes Kommunikationskonzept für das UAV-System erstellt:

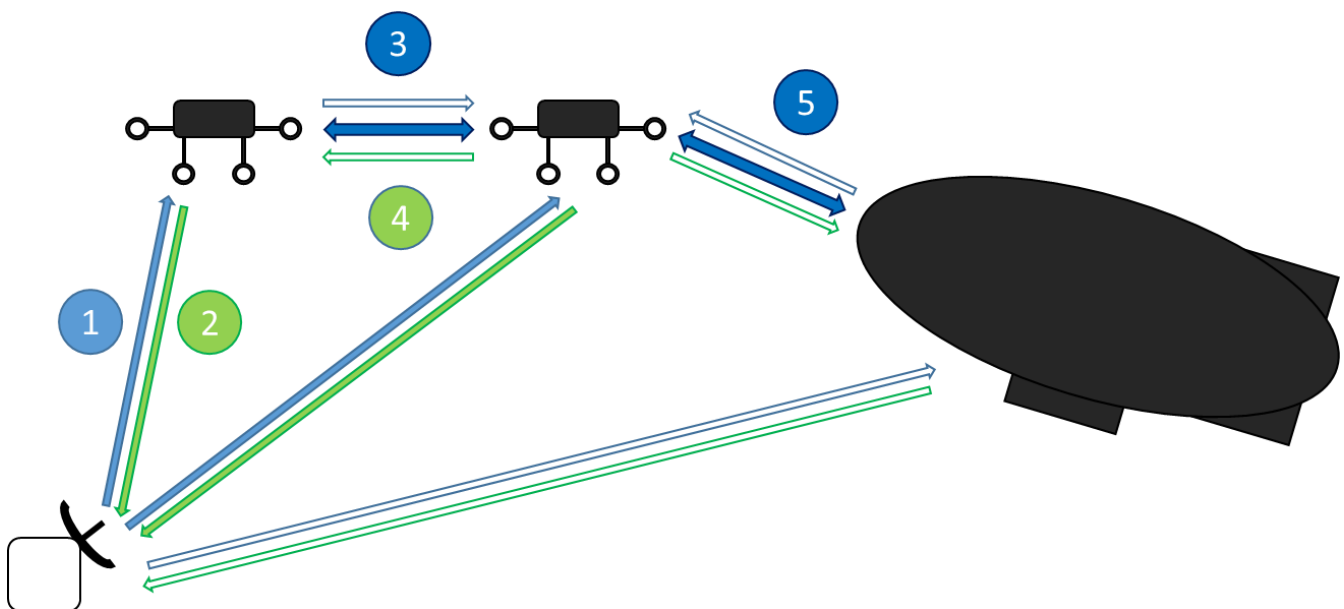


Abbildung 1: Kommunikationskonzept von Autonomous Flight

Wie in Abbildung 1 zu sehen, werden diverse Kommunikationsstrecken benötigt. Diese müssen jeweils auch unterschiedliche Daten übermitteln.

Alle fliegenden UAVs sollen untereinander kommunizieren (3). Um eine Schwarm Intelligenz aufbauen zu können müssen dazu die jeweiligen Positionen und Flugrichtungen übertragen werden. Zusätzlich muss natürlich über diese Schnittstelle die Schwarm Intelligenz an sich ausgetauscht werden. Dazu sollten auch Telemetrie- und Statusdaten zwischen den einzelnen UAVs ausgetauscht werden.

Systemarchitektur

Langfristig soll auch ein asynchroner Schwarm mit unterschiedlichen Teilnehmern aufgebaut werden (5). Auch diese müssen untereinander kommunizieren und Status-, Positions-, Telemetrie- und eventuell Payloaddaten (beispielsweise ein Videostream) austauschen.

Es soll auch die Möglichkeit geschaffen werden, ein UAV als Relaisstation für die Kommunikation von anderen UAVs zu verwenden (4). Auch eine solche Kommunikationsstrecke soll vorgesehen werden.

Zuletzt soll auch eine Bodenstation mit allen UAVs kommunizieren (1). Dabei sollen vor allem die Payload- aber auch alle Telemetrie- und Flugdaten der UAVs auf die Bodenstation übertragen werden. Die Bodenstation soll in der Lage sein, Befehle an die einzelnen UAVs zu übermitteln (2). Dabei sollen unter anderem Flugbefehle kommandiert, als auch die jeweilige Payload gesteuert werden.

Zusätzlich zu dem Kommunikationskonzept wurden vom Subteam „Autonomous Flight“ Missionskonzepte erstellt.

Genauere Informationen sind in der Studienarbeit „Autonomous Flight“ zu finden.

Aus diesen Konzepten leiten sich folgende Anforderungen an die Subsysteme des UAVs und der Bodenstation ab:

1. Hohe Datenrate der Links (ausreichend für Videostream)
2. Leistungsfähiger Rechner (Bildverarbeitung)
3. Trennung von Flug- und Payloadrechner
4. Protokoll für die Datenlinks

3.2 Architekturkonzept

Aus den gegebenen Anforderungen und den bereits vorgegebenen Komponenten wurden folgende in Abbildung 2 und in Abbildung 3 dargestellten Architekturen abgeleitet:

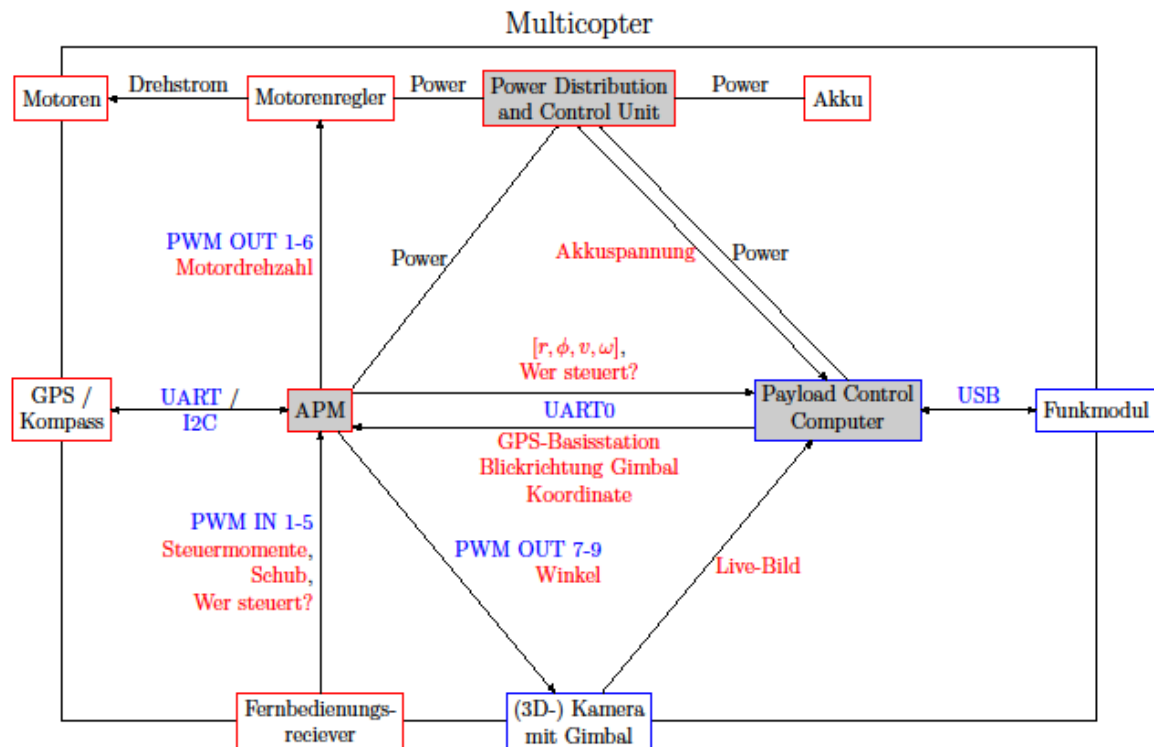


Abbildung 2: Architekturkonzept für den Multikopter

Die in Abbildung 2 mit blauer Schrift gekennzeichneten Objekte stellen physikalische Anschlüsse dar. In roter Schrift sind die zu übermittelnden Daten dargestellt. Die in rot dargestellten Subsysteme sind für die Flugsteuerung zuständig, die in blau gekennzeichneten stellen die Payload dar.

Bodenstation

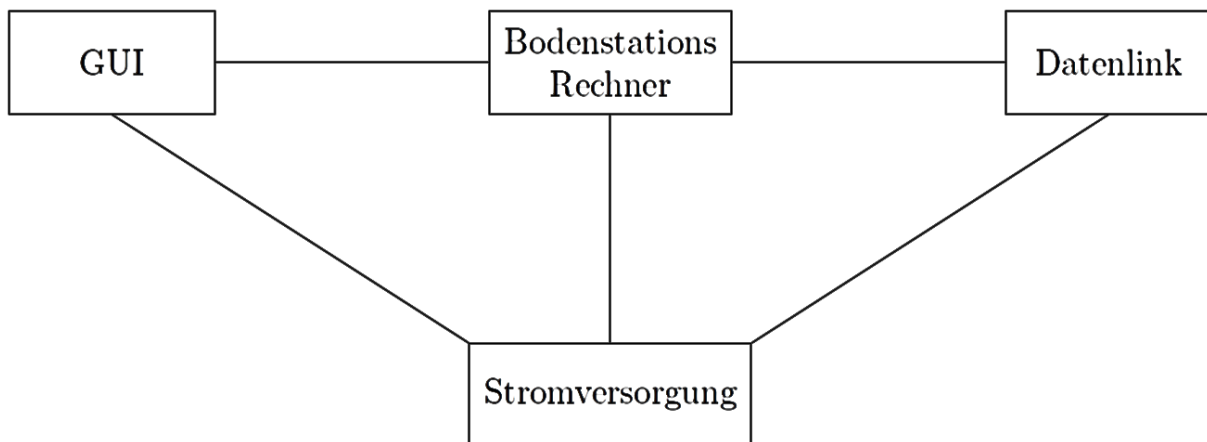


Abbildung 3: Architekturkonzept für die Bodenstation

Durch diese groben Architekturkonzepte wurde Schritt für Schritt die in Kapitel 3.3 und Kapitel 3.4 aufgeführten detaillierten Architekturen entwickelt.

3.3 Architektur des Multicopters

Aus den gegebenen Anforderungen und den bereits vorgegebenen Komponenten wurden folgende in Abbildung 2 dargestellte detailliertere Architektur für die elektrischen Systeme des Multicopters abgeleitet.

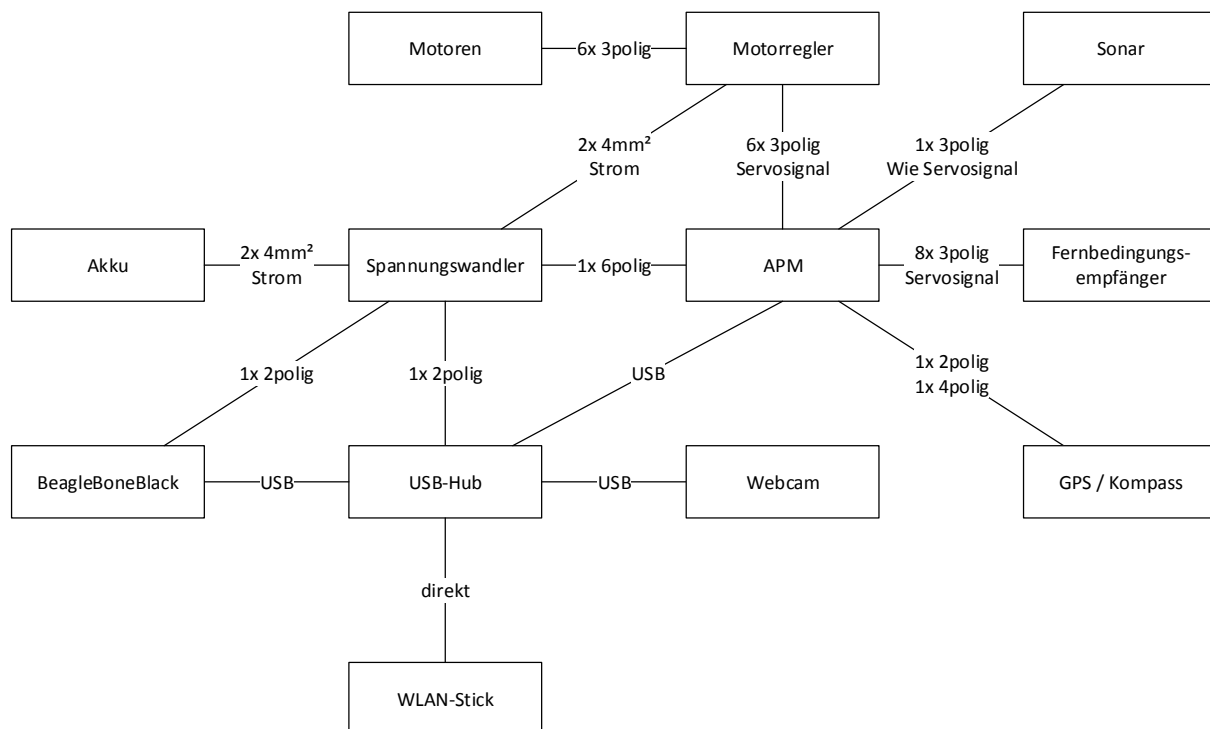


Abbildung 4: Architekturkonzept für den Multikopter

Die Architektur des Multikopters besteht aus vier Hauptkomponenten:

- Payload
- Flugsteuerung
- Antrieb
- Versorgung.

Die Flugsteuerung bildet der Flugsteuerungsrechner APM 2.6 sowie das Funkfernsteuersystem. An den APM werden noch zusätzliche externe Sensoren angeschlossen um alle Anforderungen erfüllen zu können. Unter Anderem ein Sonar-Höhensensor um die genaue Höhe über Grund zu ermitteln, sowie ein kombinierter GPS- und Kompasssensor um die genaue Position und Ausrichtung zu bestimmen.

Als Payload-Rechner wurde der Einplatinencomputer Beagle Bone Black(BBB) ausgewählt. Dieser verfügt über ausreichend Rechenleistung für eine Bildverarbeitung. Außerdem ist er durch das Linux-Betriebssystem sehr gut netzwerkfähig und Schnittstellen lassen sich dadurch leicht implementieren. Zusätzlich verfügt er über diverse General-Purpose-In- und –Outputs, diverse Interfaces wie UART, I²C sowie SPI und

Systemarchitektur

die Möglichkeit, über einen Analog-Digital-Wandler, direkt analoge Signale zu verarbeiten. Dadurch ist er allen bestehenden und zukünftigen Anforderungen gewachsen.

Als erste Payload soll eine USB-Webcam verwendet werden, deren Bilder verarbeitet werden sollen.

Als Schnittstelle zwischen den Fluggeräten und zur Bodenstation wurde WLAN ausgewählt, da dieses ausreichend Bandbreite, sowie Konfigurationsmöglichkeiten für zukünftige Konzepte bietet. Da jedoch das Funkfernsteuersystem im 2,4 GHz-Band arbeitet, wurden 5 GHz-WLAN-Komponenten ausgewählt um Interferenzen zu vermeiden.

Um die Webcam sowie den WLAN-USB-Stick an den BBB anzubinden, ist ein USB-Hub nötig, da der BBB nur einen USB-Port besitzt. Da auch der APM über eine USB-Schnittstelle verfügt kann auch dieser über USB an den BBB angebunden werden, sodass Telemetriedaten sowie Wegpunktkommandos übertragen werden können.

Der Antrieb für den Multikopter besteht im Wesentlichen aus sechs bürstenlosen Ausenläufermotoren und den dazugehörigen Motorreglern. Diese bekommen vom Flugsteuerungsrechner APM jeweils ein PWM-Signal, das die Motorleistung kommandiert. Das PWM-Signal ist bei funkferngesteuerten Fluggeräten häufig verwendetes Interface um Positionen an Stellglieder zu übermitteln. Dabei entspricht eine Pulsbreite von 1 ms dem kleinsten „Ausschlag“ sowie 2 ms dem höchsten. Die Frequenz der PWM liegt zwischen 50 Hz und 480 Hz.

Als Energiespeicher wird ein Lithium-Polymer-Akku mit drei seriellen Zellen, also einer Spannung von 11,1 V und einer Kapazität von 4000mAh verwendet. Die Strombelastbarkeit ist mit 30C, dem 30-fachen der Kapazität/h, also mit maximal 120 A angegeben. Über einen 5 V-Spannungswandler wird die Payload sowie der APM und dessen Sensoren mit Energie versorgt. Zusätzlich ist im Spannungswandler noch eine Batteriespannungs- und Strommessung integriert, die über den APM ausgelesen werden kann.

3.4 Architektur der Bodenstation

Die Bodenstation soll zunächst nur für die Auswertung sowie Kommandierung von Telemetriedaten verwendet werden. Es soll jedoch eine Grundlage geschaffen werden, um auch in Zukunft komplexere Aufgaben zu erfüllen.

Da als Schnittstelle zwischen den Fluggeräten und der Bodenstation WLAN ausgewählt wurde, ist ein WLAN-Router nötig, der als Acces-Point fungiert. Dadurch ist eine Verbindung zwischen Bodenstationscomputer und dem Fluggerät möglich.

Als Software für die Bodenstation wird die von Ardupilot bereitgestellte Software „Mission Planner“ verwendet. Diese Software ist OpenSource als Windows-Anwendung erhältlich und kompatibel zum APM 2.6.

Als Bodenstationsrechner werden Outdoor-Laptops von Dell verwendet. Diese sind auch für die harten Bedingungen von Flugversuchen im Freien ausgelegt. Abbildung 5 zeigt die Komponenten der Bodenstation und deren Verbindungen.

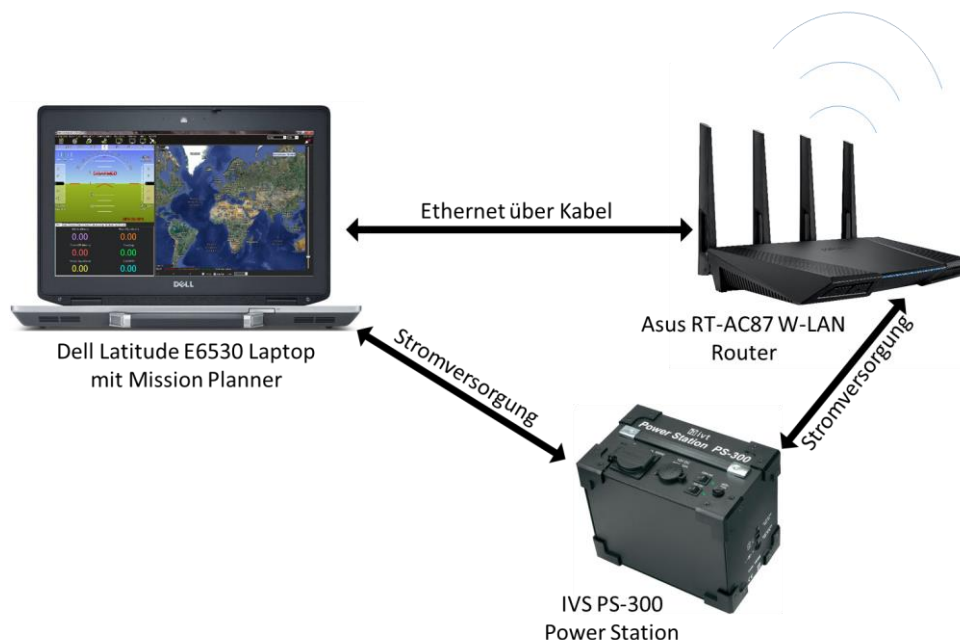


Abbildung 5: Architektur der Bodenstation

4 Subsysteme

Im Folgenden wird nun näher auf die Grundlegenden Komponenten des UAV-Systems eingegangen.

4.1 Fernsteuersystem

Als primäres Funkfernsteuersystem dieser Studienarbeit wird der Sender MC32 HOTT der Firma Graupner in Verbindung mit dem passenden Empfänger GR32 HOTT verwendet. Zusätzlich steht noch eine MX12 ebenfalls vom Hersteller Graupner zur Verfügung. Im Folgenden sollen nun die wesentlichen Funktionen des verwendeten Funkfernsteuersystems näher erläutert werden.

4.1.1 HoTT

Als Hopping Telemetry Transmission (HoTT) wird ein bidirektionales 2,4 GHz-System des Herstellers Graupner bezeichnet. HoTT arbeitet mit bis zu 75 Kanälen im 2,4 GHz-Band und verwendet das Frequenz Hopping Spread Spectrum (FHSS) Frequenzspreizverfahren. Dabei wird in zufälligen Intervallen die Übertragungsfrequenz gewechselt. Dadurch ist eine besonders störungenpfindliche bidirektionale Datenübertragung möglich und es können bis zu 200 Systeme gleichzeitig verwendet werden, ohne dass diese sich gegenseitig stören. FHSS kommt unter anderem auch bei Bluetooth zum Einsatz.

Das HoTT-System ermöglicht ein einfaches Koppeln von Sender und Empfänger. Dabei können sogar mehrere Empfänger gleichzeitig an einen Sender gebunden werden und so bis zu 32 Steuerfunktionen übertragen werden.

Außerdem kann eine sehr geringe Zykluszeit der Steuersignale eingestellt werden. Diese beträgt nur 10 ms, was doppelt so schnell ist wie der Marktstandard. [5]

Das HoTT-System ermöglicht es auch Failsafes zu konfigurieren. Dabei gibt es zwei Konfigurationsmöglichkeiten, die nach einem Verlust der Funkverbindung aktiv werden können und für jede Steuerfunktion individuell eingestellt werden können. Es kann entweder das zuletzt gültige Signal wiederholt werden, oder es wird eine vordefinierte Position ausgegeben.

Subsysteme

Durch die bidirektionale Kommunikation lassen sich auch Telemetriedaten vom Empfänger im Fluggerät zum Piloten am Boden übertragen. [6]

4.1.2 MC32

Die MC32 HoTT ist die aktuell leistungsfähigste Funkfernsteuerung von Graupner mit HoTT als Übertragungstechnik und einer Reichweite von ca. 4000m. Sie kann alle 32 Steuerfunktionen, die von HoTT gleichzeitig übertragen werden können, verwenden. Außerdem können diese Steuerfunktionen über diverse Mischer (für zum Beispiel 8-Klappenflügel bei Flächenflugzeugen, sowie Taumelscheibenmische bei Hubschraubern) gekoppelt werden. Wie in Abbildung 6 zu erkennen, verfügt die MC32 über diverse Bedienelemente und Geber. Diese können frei den einzelnen Steuerfunktionen zugewiesen werden.



Abbildung 6: MC32 HoTT Sender

Zusätzlich können über bis zu 8 Flugphasen jeweils eigene Trimmpositionen sowie die Amplitude der Steuerfunktionen über Dual-Rate konfiguriert werden. Der Verlauf der

Geber-Steller-Verknüpfung kann schrittweise von linear zu exponentiell angepasst werden. Diverse Timer-Funktionen runden den Funktionsumfang ab. Des weiteren können Software-Updates bequem per MicroSD-Karte aufgespielt werden. [7] [8]

4.1.3 GR32

Der GR32 ist das passende Gegenstück für die MC32. Durch die Bewegung des Fluggeräts ändert sich ständig die räumliche Ausrichtung der Antennen. Dies führt bei anderen Empfängern zu einer schlechteren Empfangsleistung. Um dieses Problem zu umgehen, verfügt der GR32, wie in Abbildung 7 zu erkennen ist, über ein Dual-Antennen-System, also über zwei komplett redundante Antennensysteme. Dabei wird dynamisch immer das Antennensystem mit der besseren Empfangsleistung zum Empfangen und Senden von Daten verwendet.



Abbildung 7: GR32 HoTT Empfänger

Mit dem GR32 können bis zu 16 Steuerfunktionen übermittelt werden. Außerdem werden ohne zusätzliche Sensorik bereits Telemetriedaten über Empfängerakkuspannung, Temperatur sowie Signalstärke zur MC32 übermittelt. [8]

4.1.4 Lehrer-Schüler-Modus

Zusätzlich zur MC32 steht in dieser Studienarbeit auch eine MX12 von Graupner zur Verfügung. Da auch diese über das HoTT-System verfügt, lässt sich ein kabelloser Lehrer-Schüler-Betrieb (LS) verwirklichen. Über ein solches LS-System ist es möglich, dass auch unerfahrene Piloten erste Erfahrungen mit funkferngesteuerten Fluggeräten sammeln können ohne dass dabei eine Gefahr entsteht.

Der grundlegende Aufbau eines solchen Lehrer-Schüler-Systems ist in Abbildung 8 dargestellt.



Abbildung 8: Lehrer-Schüler-Modus des Funkfernsteuersystems

Subsysteme

Dabei ist der Lehrer-Sender ständig mit dem Empfänger im Fluggerät gekoppelt. Der Schüler-Sender kommuniziert über HoTT mit dem Lehrer-Sender und übermittelt so die Steuerkommandos über den Lehrer als Relaisstation an das Fluggerät. Macht der Schüler einen Fehler oder ist mit der aktuellen Situation überfordert, kann der Lehrer-Pilot jederzeit wieder die Kontrolle über das Luftfahrzeug übernehmen und so die Kommandos des Schülers überschreiben. Dadurch können gefährliche Situationen verhindert und/oder aufgelöst werden.

4.2 Ardupilot Mega

Der Ardupilot Mega ist eine quelloffene Plattform für UAV's. Er enthält alle wichtigen Sensoren sowie eine Recheneinheit, auf der der Flugregler läuft. Diese Plattform wurde 2007 von der „DIY Drones Community“ entwickelt und basiert wie es der Name schon sagt auf der „Physical-Computing-Plattform“ Arduino. Der aktuellste Stand der Ardupilot-Plattform ist die Version 2.6, welcher in dieser Studienarbeit eingesetzt wurde. Abbildung 9 zeigt die wichtigsten Baugruppen auf der Platine des APM 2.6 [9].

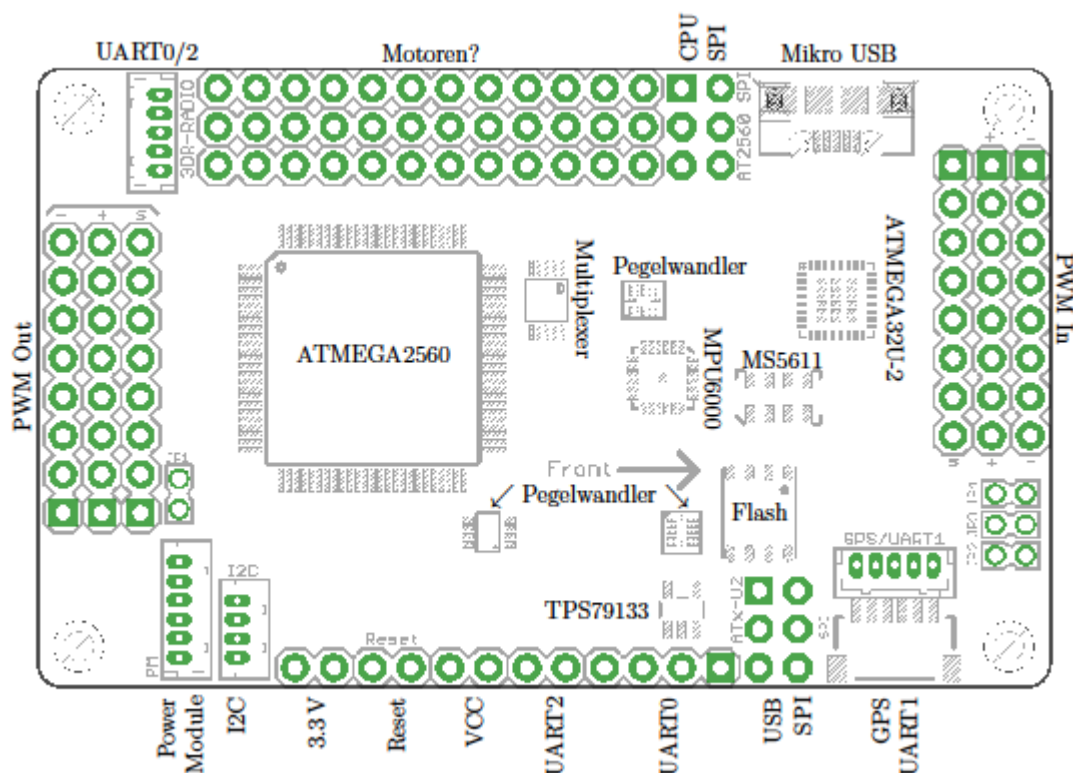


Abbildung 9: Platine mit den wichtigsten ICs und Anschlüssen des APM 2.6

In Abbildung 10 ist das Blockschaltbild des APM's dargestellt. Das Telemetrie-Modul, das GPS-Modul und der Kompass sind nicht im Gehäuse des APM's untergebracht sondern müssen extern angeschlossen werden. Aus dem Luftdruck, welcher vom Barometer aufgenommen wird, wird die Höhe berechnet. Der MPU6000 Chip ist ein Sensormodul, welches die Drehraten und die Beschleunigungen in allen drei Achsen misst. Das GPS-Modul wurde aus Empfangsgründen in ein externes Gehäuse verpackt. In diesem Gehäuse befindet sich auch das Kompassmodul. Dieses wurde dort platziert um die magnetische Beeinflussung durch die Motoren zu minimieren. An die Stiftleiste für die PWM-In kann der Funkfernsteuerungsempfänger angeschlossen werden. An der Stiftleiste für die PWM-Out können die Motorregler und das optionale Gimbal angeschlossen werden.

AMP 2.6

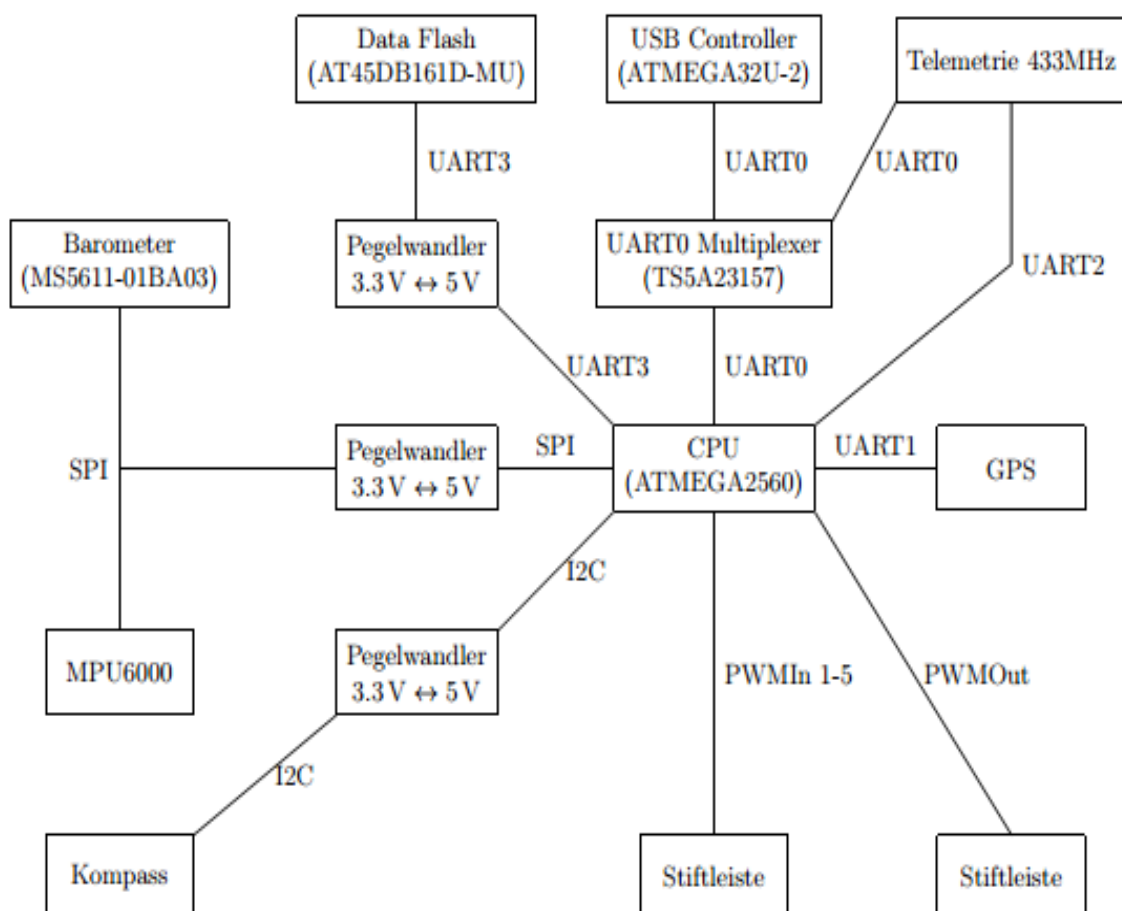


Abbildung 10: grundlegender Aufbau des APM 2.6

4.3 Der Beagle Bone Black

Der Beagle Bone Black ist ein kostengünstiger Einplatinen-Computer, der mit einem Sitara AM3358 ARM Cortex-A8 von Texas Instruments ausgestattet ist. Des Weiteren ist er mit 512MB DDR3 RAM und 4GB eMMC Flash Speicher ausgestattet. Als Schnittstellen besitzt er einen USB-Host an dem USB-Geräte angeschlossen werden können, einen USB-Client über den auf das Beagle Bord zugegriffen werden kann, eine Ethernet-Schnittstelle, eine HDMI Schnittstelle und zwei 46-Pin Anschlussleisten mit mehreren I/O Pins. Auf den Anschlussleisten können auch noch UART-, I2C-, SPI-Schnittstellen konfiguriert werden. Das Beagle Board unterstützt unter anderem die Betriebssysteme Android, Ubuntu und Debian [10]. Durch diese Betriebssysteme lässt sich das Beagle Board problemlos in Netzwerkstrukturen einbinden. Er wird auch über ein Hardware Support Package direkt von Matlab/Simulink unterstützt. Somit ist eine Kompilierung der in Simulink erstellten Software auf das Beagle Board problemlos möglich. Der Beagle Bone Black ist sehr kompakt mit einer Größe von ca. 86 x 53 mm und einem Gewicht von ca. 40 Gramm. Er ist ebenfalls sehr energiesparend mit einer maximalen Leistungsaufnahme von 2,5 Watt, bietet aber dennoch ausreichend Rechenleistung für diese Studienarbeit. [11]

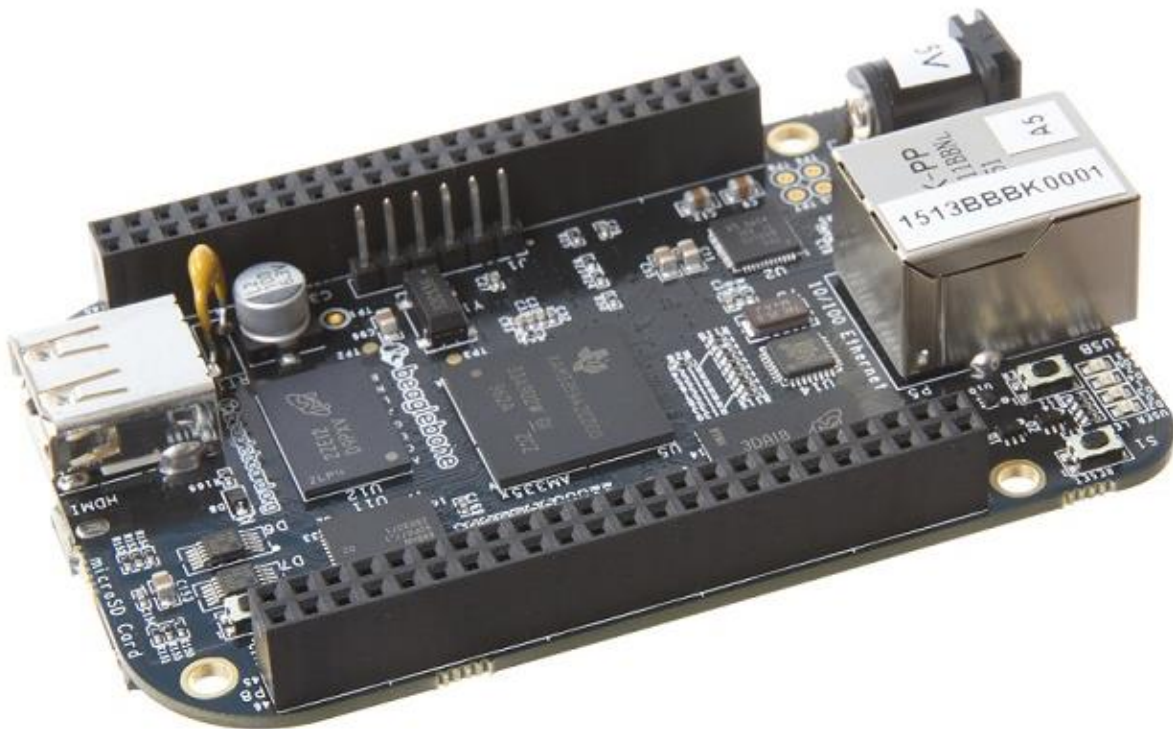


Abbildung 11: Beagle Bone Black [11]

5 Integration der Baugruppen

In diesem Kapitel werden die zwei Verwendungsmöglichkeiten, die während der Studienarbeit entstanden sind sowie deren Einrichtung erläutert. Darüber hinaus wird beschrieben, wie der Beagle Bone Black eingerichtet wird. Dies deckt einerseits eine erneute Installation des Betriebssystems sowie andererseits die Installation des WLAN-Sticks ab.

5.1 *BeagleBoneBlack neu aufsetzen*

Falls Probleme mit dem BeagleBoneBlack auftreten, kann es helfen bei diesem die Firmware neu auf den eMMC zu flaschen. Hierzu wird eine leere microSD-Karte und ein Rechner mit Linux sowie eine Internetverbindung über Ethernet benötigt.

Zuerst muss das aktuelle Debian Image für den BeagleBoneBlack von der Herstellerwebseite heruntergeladen werden. Dann muss das Image auf die microSD-Karte geschrieben werden. Danach muss noch das Beschreiben des eMMC aktiviert werden. Dies geschieht in der Datei „/boot/uEnv.txt“ der FAT-Partition der microSD-Karte. In der Datei muss der folgende Befehl einkommentiert werden:

```
cmdline=init=/opt/scripts/tools/eMMC/init-eMMC-flasher-v3.sh
```

Jetzt folgt das Flashen. Die microSD-Karte in den ausgeschalteten BeagleBoneBlack stecken und diesen dann über USB anschließen. Die LEDs fangen an ein Lauflicht anzuzeigen. Nach einigen Minuten gehen alle LEDs aus. Dann die microSD-Karte entfernen und die USB-Verbindung des BeagleBoneBlack kurz trennen.

Damit ist das eigentliche Flashen beendet. Jetzt empfiehlt es sich, die Pakete auf dem BeagleBoneBlack noch zu aktualisieren. Dazu diesen via Ethernet mit dem Internet verbinden und einmal „apt-get update && apt-get upgrade“ als „root“ ausführen.

Anschließend sollte die benötigte Payloadsoftware wie in Kapitel 5.3.1 beschrieben installiert werden. Außerdem muss noch der Treiber für den WLAN-Stick nach *Installation des WLAN-Sticks*.

5.2 *Installation des WLAN-Sticks*

Die Installation des WLAN-Sticks besteht aus zwei Hauptschritten. Der erste Schritt ist die Installation des Treibers. Der zweite Schritt ist die Einrichtung des Netzwerkes. Für die Installation ist eine kabelgebundene Internetverbindung erforderlich.

5.2.1 Installation des Treibers des WLAN-Sticks

Für die Installation des Treibers muss zuerst noch ein Paket installiert werden. Dieses Paket enthält die Headerdateien zum aktuell installierten Linux-Kernel. Der Befehl lautet wie folgt, wobei natürlich vorher die Paketquellen aktualisiert werden sollten:

```
apt-get install linux-headers-$(uname -r)
```

Nachdem dieses Paket installiert ist, muss der Inhalt eines Ordners komplett in einen anderen kopiert werden, damit der Treiber kompiliert werden kann. Dieser Ordner muss vorher noch erstellt werden. Hierzu werden folgende Befehle benötigt.

```
cd /usr/src/linux-headers-$(uname -r)/arch/
```

```
mkdir armv7l
```

```
cp -R -v arm/* armv7l
```

Danach muss der Originaltreiber von der ASUS-Website auf den BeagleBoneBlack heruntergeladen und auf oberster Ebene entpackt werden. Darauf muss der Treiber aus dem Unterordner driver entpackt werden. Dieser muss dann mit „make“ kompiliert werden. Dies kann auf dem BeagleBoneBlack ein wenig dauern. Nach der Kompilierung muss der Treiber noch an die richtige Stelle kopiert werden. Dies geschieht mit folgendem Befehl:

```
cp 8812au.ko /lib/modules/$(uname -r)/kernel/drivers/net/wireless/
```

Zum Schluss muss der Treiber noch dem Betriebssystem bekannt gemacht werden. Dies geschieht mit „depmod“. Es folgt noch ein Reboot und dann ist der Treiber installiert.

5.2.2 Einrichtung des Netzwerkes

Zuerst muss überprüft werden, ob der WLAN-Stick korrekt erkannt wurde. Dies geschieht mit Hilfe des Befehls „iwonfig“. Hiermit werden alle Netzwerkinterfaces gescannt und dabei geprüft, ob diese drahtlose Schnittstellen sind. Wenn der WLAN-Stick richtig erkannt wurde, sollte in der Ausgabe ein Interface „wlan0“ oder ein Ähnliches wie „wlan1“ erscheinen, das zur Zeit nicht mit einem Netzwerk verbunden ist. Dies ist soweit in Ordnung. Im Folgenden ist jeweils das passende Interface zu ersetzen.

Zur Einrichtung muss dem Betriebssystem bekannt gemacht werden, was es an dem Interface „wlan0“ erwartet und wie die Parameter des WLANs wie SSID und Passwort sind. Dies wird in der Datei „/etc/network/interfaces“ gemacht. Diese muss folgenden Abschnitt beinhalten:

```
# wifi interface wlan0

auto wlan0

allow-hotplug wlan0

iface wlan0 inet dhcp

    wpa-ssid UAS

    wpa-psk DHBW_UAV
```

5.3 *Der BeagleBoneBlack als Bridge*

Für die ersten Flüge und zur Kalibrierung des APM soll der BeagleBoneBlack als Bridge für die Mavlink Nachrichten dienen. Dazu sollen alle Nachrichten von der seriellen Schnittstelle an die WLAN-Schnittstelle weitergeleitet werden, um sie dann über WLAN an die Bodenstation weiterzuleiten. Auf der Bodenstation läuft dann die MissionPlanner Software, mit der der APM kalibriert werden kann und neue Wegpunkt getestet werden können. Die MissionPlanner Software kann die MAVLink-Nachrichten entweder direkt vom APM über die serielle Schnittstelle, die über die USB-Verbindung geführt wird, oder über eine Ethernet/WLAN Schnittstelle entgegenzunehmen und zu senden.

Integration der Baugruppen

Für die gewünschte Aufgabenstellung gibt es schon die fertige Software Mavproxy [12]. Diese Software ist in Python geschrieben und leitet die MAVLink-Nachrichten zwischen einem TCP/IP-Port und einer seriellen Schnittstelle weiter.

Im Folgenden wird die Installation der Software sowie die Inbetriebnahme erläutert, wobei die Anleitung aus [12] als Grundlage dient und auf das System mit Hilfe von [13] angepasst wird.

5.3.1 Installation von Mavproxy

Für die Installation von Mavproxy auf einem BeagleBoneBlack wird eine Internetverbindung benötigt. Die Anleitung ist so geschrieben, dass Grundkenntnisse in Linux und dem Umgang mit Einplatinencomputern benötigt werden.

Zuerst müssen die folgenden Pakete auf dem BeagleBoneBlack aus den Standardpaketquellen installiert werden:

- python-wxgtk2.8
- python-opencv
- python-pip

Dann wird mit Hilfe von „pip“ die Mavproxy Software installiert. Hierzu muss der User „root“ folgenden Befehl ausführen: „pip install mavproxy“

5.3.2 Inbetriebnahme von Mavproxy

Für die Inbetriebnahme von Mavproxy sind einige Vorkehrungen zu treffen. Zuerst muss sichergestellt werden, dass eine Verbindung zwischen BeagleBoneBlack und APM besteht. Dies ist vorzugsweise mit einer USB-Verbindung realisierbar. Diese kann geprüft werden, indem geprüft wird, ob der serielle Port „/dev/ttyACM0“ auf dem BeagleBoneBlack vorhanden ist. Darüber hinaus muss noch die Baudrate bekannt sein, mit der die Nachrichten versendet werden.

Außerdem müssen die IP-Adressen vom BeagleBoneBlack und dem Bodenstationsrechner, auf dem die Missionplannersoftware läuft, bekannt sein. Wenn der Beagle-

Integration der Baugruppen

BoneBlack und der Bodenstationsrechner via USB verbunden sind, gelten die folgenden IP-Adressen. Der BeagleBoneBlack hat die IP-Adresse “192.168.7.2” und der Bodenstationscomputer die IP-Adresse „192.168.7.1“.

Wenn diese Vorkehrungen getroffen sind, kann dann Mavproxy gestartet werden. Hierzu muss eine Konsole mit als „root“ auf dem BeagleBoneBlack zum Beispiel mit Putty gestartet werden. Mit dieser wird dann die Mavproxy Software gestartet. Bei einer USB-Verbindung zwischen dem BeagleBoneBlack und dem APM sowie zwischen BeagleBoneBlack und dem Bodenstationsrechner lautet der Befehl wie folgt:

```
mavproxy.py --master=/dev/ttyACM0,115200  
--out=udp:192.168.7.1:14550
```

Bei diesem Befehl gibt es zwei wichtige Optionen. Die erste ist die Option „—master“. Mit dieser wird eingestellt an welchem Port der APM angeschlossen ist. Dies kann wie im Beispiel der serielle Port, der von der USB-Schnittstelle erzeugt wird, sein oder auch ein beliebiger anderer serielle Port, an dem der APM angeschlossen ist. Es kann sogar ein Ethernet oder WLAN-Port sein, wenn zum Beispiel das Signal von einem APM über mehrere (Relay-)Stationen weitergeleitet werden soll. Dies kann bei Schwarmmissionen im weiteren Verlauf des Projektes nützlich sein. Die zweite wichtige Option ist „—out“. Mit dieser Option wird der Port angegeben an welchen die MAVLink-Nachrichten weitergeleitet werden sollen. Wie auch schon beim Eingangsport sind alle möglichen seriellen sowie Ethernet Ports möglich.

Die Ports werden analog definiert. Bei seriellen Ports wird zuerst der Devicename des seriellen Ports wie z. B. „/dev/ttyACM0“ oder „COM1“, wenn die Mavproxy Software unter Windows läuft, angegeben und dann nach einem Komma die Baudrate des Ports. Bei Ethernet Ports wird zuerst das Protokoll, „udp“ oder „tcp“, dann nach einem Doppelpunkt die IP-Adresse des Empfängers und nach einem weiteren Doppelpunkt die Portnummer, typischerweise 14550, angegeben.

Wenn Mavproxy auf dem BeagleBoneBlack gestartet ist, kann dann der Missionplaner mit dem APM verbunden werden. Dazu muss, wie in Abbildung 12 zu sehen, als Port „UDP“ ausgewählt werden und dann auf „CONNECT“ geklickt werden. Danach muss die Portnummer bestätigt werden. Die Verbindung wird dann aufgebaut.



Abbildung 12: Aktionen zum Verbinden von Missionplanner und Mavproxy aus [6]

5.4 Implementierung der APM-PCC Schnittstelle auf dem PCC

Der Multicopter soll in der Lage sein, eigenständig Wegpunkte anzufliegen, die während des Fluges live bestimmt werden. Die Bestimmung der Wegpunkte findet auf dem PCC statt. Diese müssen dann an den APM übertragen werden, da auf diesem die Flugregelung zum Anfliegen von Wegpunkten stattfindet. Zur Berechnung der Wegpunkte werden verschiedene Daten des Flugzustandes benötigt. Zum Übertragen der Daten wird das MAVLink Protokoll genutzt. Das im Folgenden vorgestellt wird.

5.4.1 Das MAVLink Protokoll

Das MAVLink Protokoll ist ein im UAV-Bereich genutztes Protokoll, dass die Kommunikation zwischen UAV und UAV sowie UAV und GCS ermöglicht. Es ist dazu gedacht Steuerungskommandos und Zustandsdaten aus C-Structs in Paketen über serielle Schnittstellen zu übertragen. Diese können auch Funkstrecken zum Beispiel über xBee-Module enthalten. Durch einen geringen Protokoll Overhead ist es sehr effektiv. Das Protokoll wurde 2009 von Lorenz Meier mit LGPL Lizenz veröffentlicht. In unserem Projekt wird dieses Protokoll leicht außerhalb seines eigentlichen Einsatzgebietes genutzt, nämlich zur Kommunikation innerhalb des Multicopters zwischen APM und PCC. Es wurde extensiv mit dem PX4, dem PIXHAWK, dem APM und der Parrot RT:Drone getestet MAVLinkWebsite. Das Protokoll spezifiziert zur Zeit ca. 150 Nachrichten, die Daten übertragen. Diese sind zusammen mit enum für verschiedene Parameter in einer XML-definiert und werden aus dieser XML mit einem Python-Skript in C-Header umgewandelt.

Aufbau der Datenpakete

Die Beschreibung des Aufbaus der Datenpakete folgt weitestgehend MAVLinkWebsite, MAVLinkForDummies und eigener Recherche im C-Code und der XML-Paketbeschreibung. Der Aufbau eines Datenpaketes wurde von CAN und SAE AE-4 Standards inspiriert. Ein MAVLink Datenpaket hat je nach Nachrichtentyp eine feste Länge zwischen 8 Byte und 263 Byte. Diese setzt sich aus 6 Byte Header, zwischen 0 Byte und 255 Byte Payload und 2 Byte Checksumme zusammen. Die Checksumme wird zur Fehlereerkennung genutzt. In dem Payload-Teil werden die eigentlichen Daten übergeben. Im Header wird spezifiziert wie lang die Nachricht ist, der Nachricht eine durchlaufende Nummer gegeben, von wem die Nachricht kommt und was für eine Nachricht genutzt wird. Das erste Byte des Headers ist immer 0xFE und gibt das Startbyte, mit dem eine Nachricht beginnt. Dann folgt im zweiten Byte die Länge der Nachricht. Das dritte Byte enthält den durchlaufenden Zähler. Das sendende System wird mit dem vierten Byte bezeichnet. Im fünften Byte wird die sendene Komponente spezifiziert. Der Typ der Nachricht wird im sechsten Byte übergeben.

Encodierung der Nachrichten

Die C-Header stellen Funktionen zum Packen und Encodieren der Nachrichten zu Verfügung. Dazu muss in das entsprechende Programm nur der C-Header „common.h“ eingebunden werden.

5.4.2 Implementierung

Die PCC-Software soll in Simulink implementiert werden. Dies sorgt für einen einfachen Datenfluss innerhalb der PCC-Software. Die APM Schnittstelle soll also aus einem Simulink-Subsystem bestehen. In diesem müssen die Parameter zu einer Nachricht encodiert werden und die empfangenen Nachrichten dekodiert werden. Außerdem müssen diese über die serielle Schnittstelle an den APM weitergeleitet werden. Hierzu sollen S-Functions in C und Matlab genutzt werden. Diese ermöglichen es, beim Starten des Simulinkmodells den seriellen Port zu öffnen, bei jedem Durchlauf des Modells diesen auszulesen und beim Beenden den Port wieder zu schließen.

Über eine Internet-Recherche wurden verschiedene Ansätze gefunden, um MAVLink in Simulink zu implementieren. Der in dieser Studienarbeit geforderten Anwendung kommt ein Projekt des Loughborough University Centre for Autonomous Systems [14] recht nah. Jedoch werden dort nur die direkten Sourcen angeboten. Alle include-Dateien werden nicht mitgeliefert. Damit dieser Block jedoch geändert werden kann, sind die include-Dateien nötig. Nach mehreren Recherchen und einigen Modifikationen konnten die passenden include-Dateien erstellt werden. In der aktuellen Version können nun Änderungen an der Datei vorgenommen und später auch über den Matlab-Befehl

```
mex MAVLink_Interface.cpp
```

kompiliert werden. Um den MAVLink-Block zu nutzen muss die *.mdl Datei geöffnet und dann der darin enthaltene Block über Drag and Drop in das gewünschte Simulink-Modell portiert werden. In den Konfigurationsparametern können die Eingangs- und Ausgangssignale eingestellt werden. Ebenfalls kann dort auch der COM-Port konfiguriert werden. Die aktuell bearbeitete Version des MAVLink-Blockes läuft nur unter Windows. Damit dieser Block auch auf dem Beagle Bone Black lauffähig ist, müssen

Integration der Baugruppen

die entsprechenden Windows-Handle Befehle in Linux-Befehle umgewandelt werden. Ebenfalls muss dafür auch die gesamte Kommunikation mit der seriellen Schnittstelle umprogrammiert werden. Um die Kommunikation zwischen Beagle Bone Black und APM2.6 zu testen wurde ein einfacher Block geschrieben, der alle Empfangenen Daten in ein Byte-Array schreibt. Ebenfalls können über diesen Block auch einzelne Bytes an den APM2.6 übertragen werden. Teile dieser S-Function könnten somit wieder in dem MAVLink-Interface verwendet werden. Der letzte Stand des MAVLink-Interfaces konnte leider aus Zeitgründen nicht mehr getestet werden.

5.5 Anschluss des Sonars

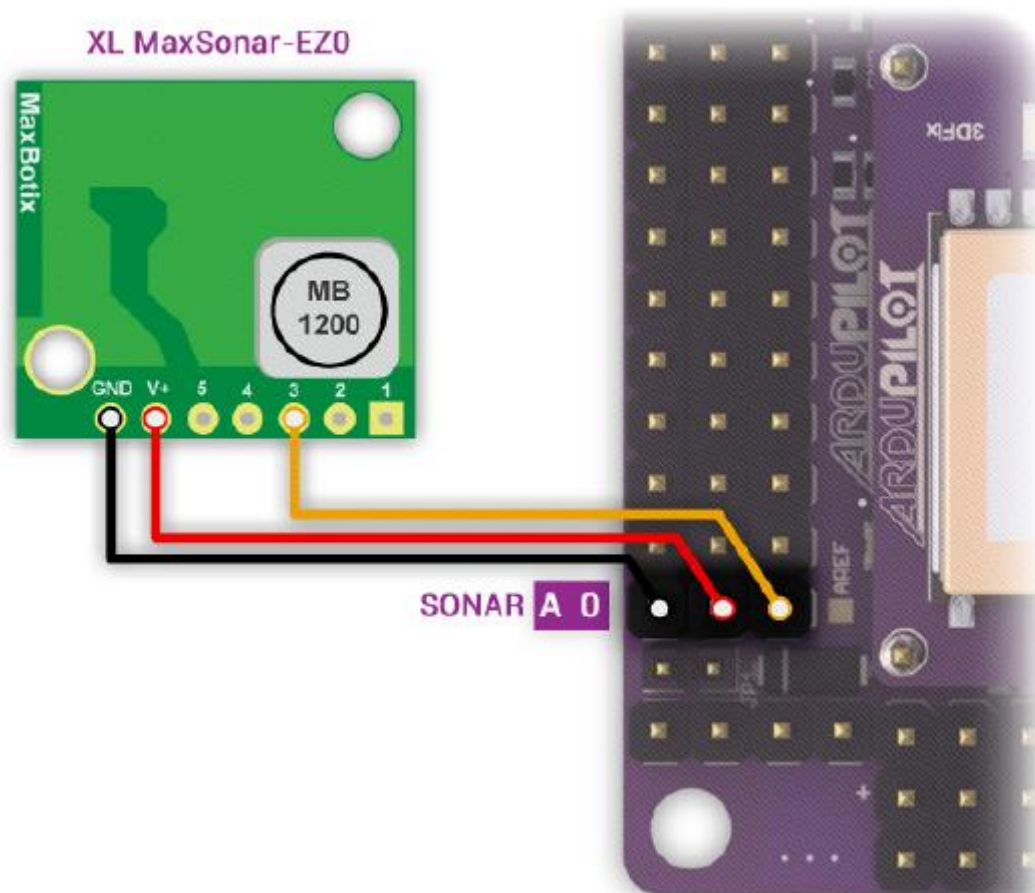


Abbildung 13: Elektrischer Anschluss des Sonars aus [3]

Weil das Sonar ein Zusatzfeature für den APM ist, muss dieses extra konfiguriert und angeschlossen werden. Es wurde ein Sonar vom Typ XL-MaxSonar EZ4 MB1240 von MaxBotix ausgewählt. Dies ist in Ardupilot:AnalogSonar empfohlen. Dort ist auch beschrieben, wie das Sonar angeschlossen werden muss. Dies ist in Abbildung 13 dargestellt, wobei auf den vorgeschlagenen Tiefpass in der Spannungsversorgung verzichtet wurde.

Die Konfiguration des Sonars ist auf dieser Seite nur unzureichend beschrieben, weil die angegebenen Einstellmöglichkeiten nicht vorhanden sind. Wie der APM für das Sonar konfiguriert werden muss, ist in [15] beschrieben, da hier die gleiche Firmware-

Integration der Baugruppen

Version für den APM verwendet wird. Für das Sonar sind 11 Parameter wichtig. Diese beginnen alle mit „RGNFND“. Die Werte der Parameter sind in Abbildung 14 zusammengestellt. Einige der Parameter lassen sich aus dem Datenblatt übernehmen, die anderen geben an, wie und wo das Sonar angeschlossen ist.

Parameter	Wert
RGNFND_FUNCTION	0
RGNFND_GAIN	0,8
RGNFND_MAX_CM	700
RGNFND_MIN_CM	20
RGNFND_OFFSET	0
RGNFND_PIN	0
RGNFND_RMETRIC	1
RGNFND_SCALING	2,04
RGNFND_SETTLE_MS	-
RGNFND_STOP_PIN	-1
RGNFND_TYPE	1

Abbildung 14: Die APM-Parameter für das Sonar

5.6 Implementierung der Bodenstation

Die Bodenstation wird in den ersten Flügen nur für die Telemetrie verwendet. Diese wurde beim Erstflug mit Hilfe des Payloadsubsystem via WLAN übertragen. Beim zweiten Flug diente das Telemetriemodul diesem Zweck.

Als Software für die Bodenstation wird MissionPlanner genutzt. Diese Software ist OpenSource und ist kompatibel zur Flugsteuerungssoftware. Die Software ist nur unter Windows lauffähig, was beim zweiten Flug zu einer etwas komplizierteren Bodenstation führte. Die beiden Telemetripfade sind in Abbildung 15 dargestellt.

Der obere Pfad stellt den ersten Flug dar, der untere Pfad den zweiten Flug. Beide Konzepte sind leider nicht gleichzeitig nutzbar.

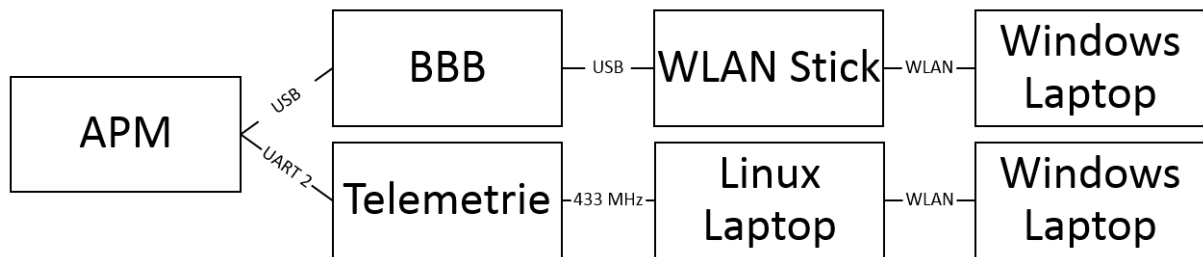


Abbildung 15: Betriebsmodi der Bodenstation

Im unteren Pfad wird das WLAN-Netzwerk vom Linux-Laptop bereitgestellt. Es wird kein externer WLAN-Router benötigt. Im unteren Pfad wird der Linux-Laptop benötigt, weil der Bodenstationsteil der gekauften Telemetrie nicht unter Windows funktioniert, aber unter Linux, da Windows keinen passenden Treiber findet. Auf dem Linux-Laptop läuft dann die gleiche Software, wie beim ersten Flug auf dem Payloadrechner (MavProxy, s. Kapitel 5.3).

6 Verbesserungen und Ausblick

Nach den erfolgreich absolvierten ersten Flugversuchen hat sich gezeigt, dass das entwickelte UAS funktionstüchtig ist. Jedoch wurden auch viele Verbesserungsmöglichkeiten entdeckt. Auf diese soll in diesem Kapitel eingegangen werden und abschließend ein Ausblick auf die Fortführung dieser Studienarbeit gegeben werden.

6.1 Optimierung der Spannungsversorgung

Während dem Erstflug traten Probleme in Punkt 18 der Gerätespezifischen Preflight Checkliste auf. Bei der Abarbeitung dieses Punktes wird die Verteilerplatte mit dem Spannungsregler verbunden. Somit sind nach dieser Handlung die Motorenregler mit Spannung versorgt. Während dem einstecken der Bodenplatte meldete der Beagle Bone Black „Power Button pressed“ und schaltete aus. Nachdem diese Meldung auch nach einem zweiten Versuch wieder auftauchte wurde der Erstflug abgebrochen.

Der komplette Aufbau wurde am nächsten Tag wieder in der ähnlichen Weise aufgebaut. Dabei konnte dieses Verhalten jedoch nicht reproduziert werden. Nach mehrmaligen ein- und ausstecken der Verteilerplatte bootete der APM nun neu. Dieses Verhalten konnte nun in verschiedenen Testreihen reproduziert werden.

Mit 60% Akkuladung konnte dieses Verhalten sogar bei jedem Einstecken der Verteilerplatte beobachtet werden. Ebenfalls verabschiedete sich nun auch der Beagle Bone Black ohne Fehlermeldung. Der Verdacht fiel auf die Spannungsversorgung. Aus diesem Grund wurde diese nun näher untersucht.

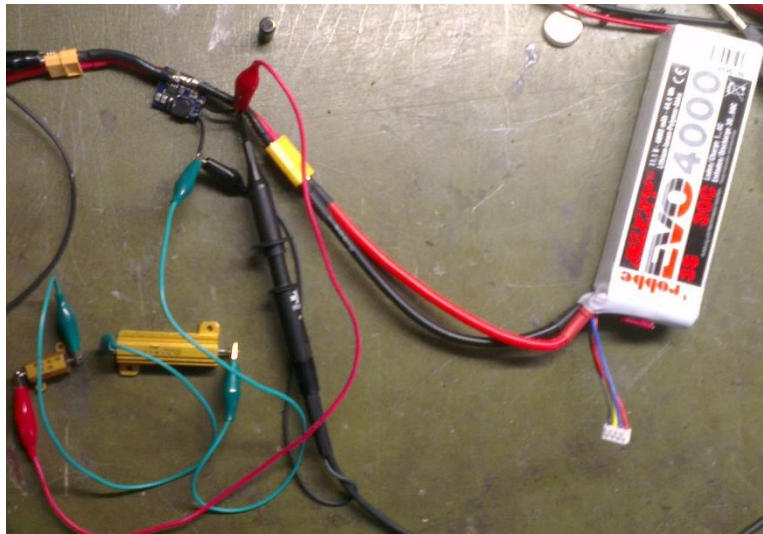


Abbildung 16: Testaufbau für die Spannungsversorgung

Zuerst wurde die Restwelligkeit der Spannungsversorgung mit einer Last von $3\ \Omega$ an der 5 Volt Seite mit dem Versuchsaufbau in Abbildung 16 am Oszilloskop untersucht.

Der Schaltregler arbeitet mit einer Schaltfrequenz von 400 kHz. Teilweise konnten Peaks von bis zu 40 mV gemessen werden (siehe Abbildung 17). Diese Spannung entspricht jedoch allen Anforderungen seitens des Beagle Bones Black und des APM's an die Spannungsversorgung. Somit ist im „Normalbetrieb“ kein Fehlerfall zu erwarten.

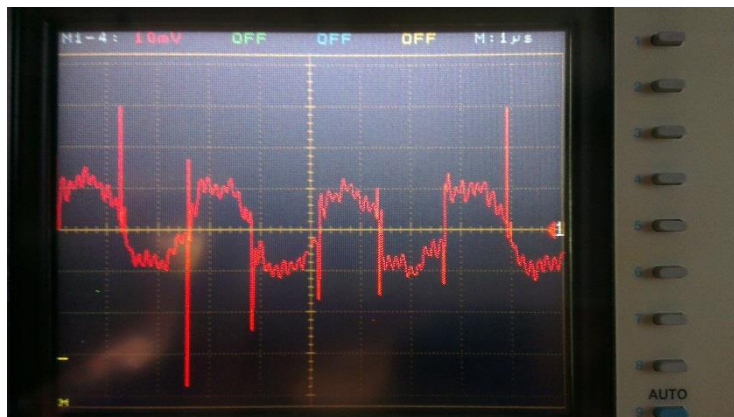


Abbildung 17: Ausgangsspannung des Spannungsreglers unter Last

Verbesserungen und Ausblick

Jeder Motorregler beinhaltet Kondensatoren, die beim Anstecken der Bodenplatte geladen werden. Dieser kurze Strom-Peak könnte auch Auswirkungen auf die 5 V Seite haben. Um dies ebenfalls zu untersuchen wurden zwei 2200 μF parallel geschaltet, entladen und an den Ausgang für die Bodenplatte angeschlossen. Während die 5 V Versorgung belastet war konnte auf dieser keine große Veränderung der Spannung festgestellt werden. Im Leerlauf dagegen konnte ein starker Spannungsspeak gemessen werden (siehe Abbildung 18).

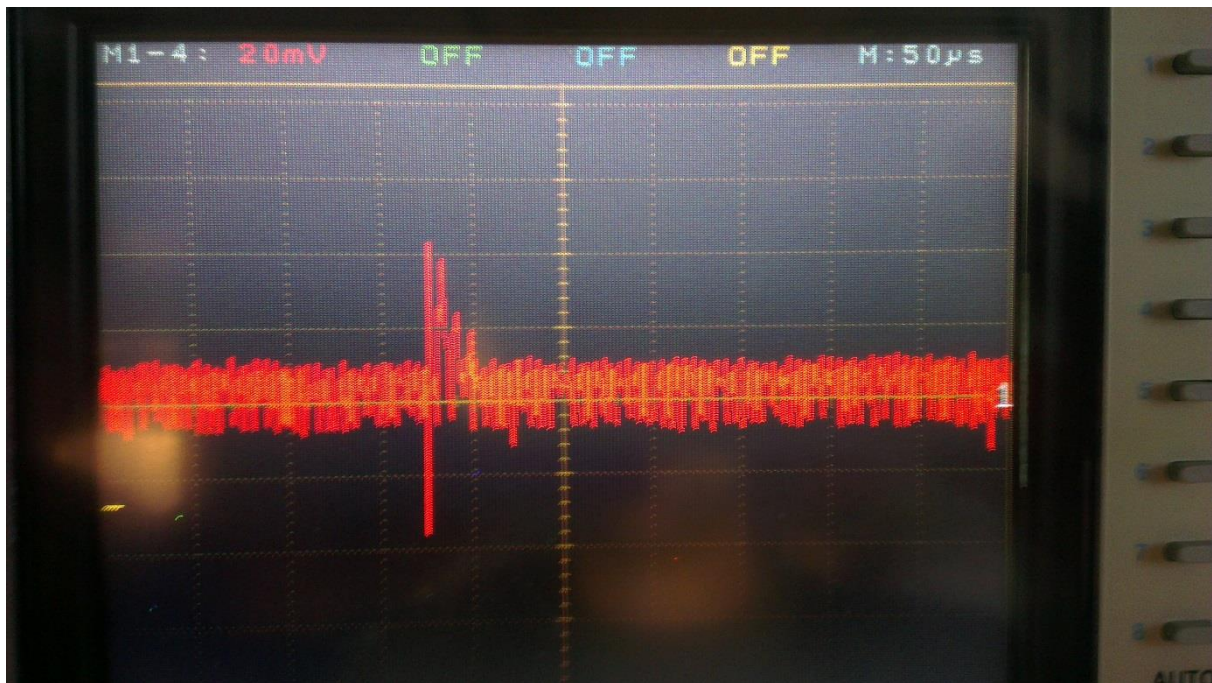


Abbildung 18: Spannungsverlauf bei Anschluss einer Kapazität

Um diesen Peak zu verringern wurde ein 1500 μF , low ESR Kondensator an die 5 V gelötet wie Abbildung 19 entnommen werden kann. Somit konnte der Peak im Versuchsaufbau deutlich reduziert werden. Abbildung 20 zeigt den aufgenommenen Spannungsverlauf nach der Modifikation.



Abbildung 19: Modifiziertes Spannungsversorgungsmodul

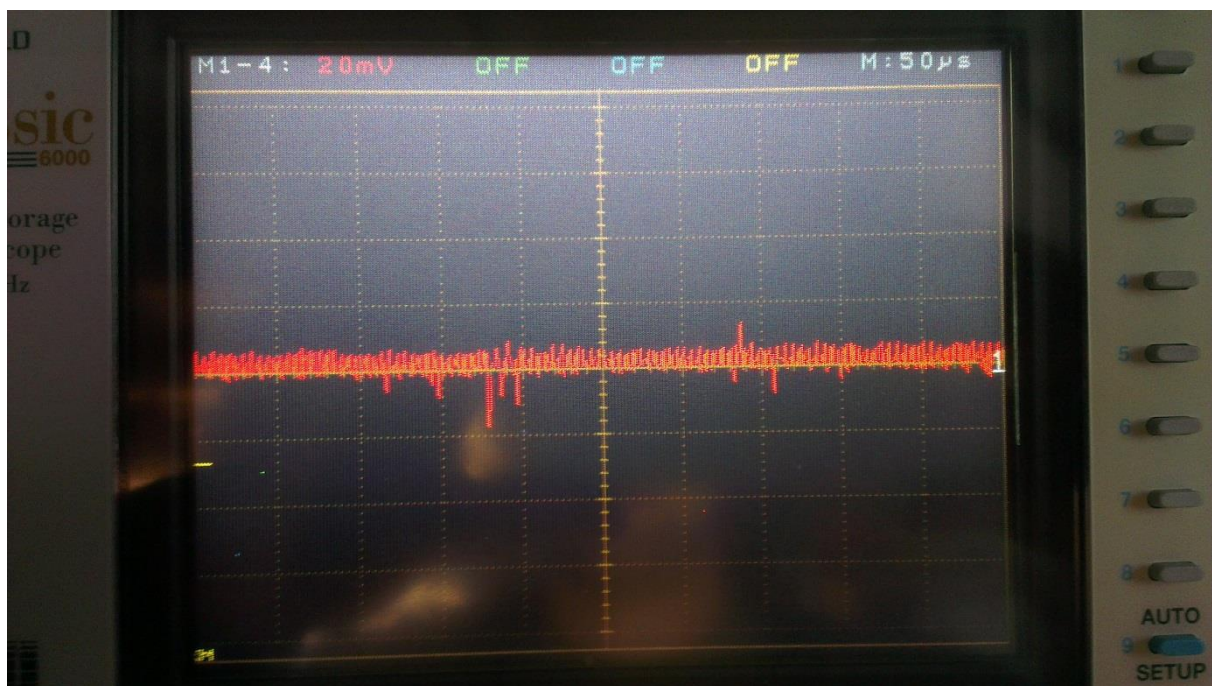


Abbildung 20: Spannungsverlauf nach der Modifikation unter gleichen Bedingungen

Die modifizierte Spannungsversorgung wurde anschließend in den Hexacopter „Boris“ eingebaut und es wurde die 5 V Spannungsversorgung während des Einsteckvorganges der Verteilerplatte mit einem 200 MHz Oszilloskop von HP beobachtet. Dabei konnte der Peak in Abbildung 21 beobachtet werden. Ein Spannungsimpuls dieser

Verbesserungen und Ausblick

Ausprägungsart sollte jedoch keine Auswirkungen auf die Baugruppen mehr haben. Im folgenden Testbetrieb konnten keine weiteren Anomalien festgestellt werden.

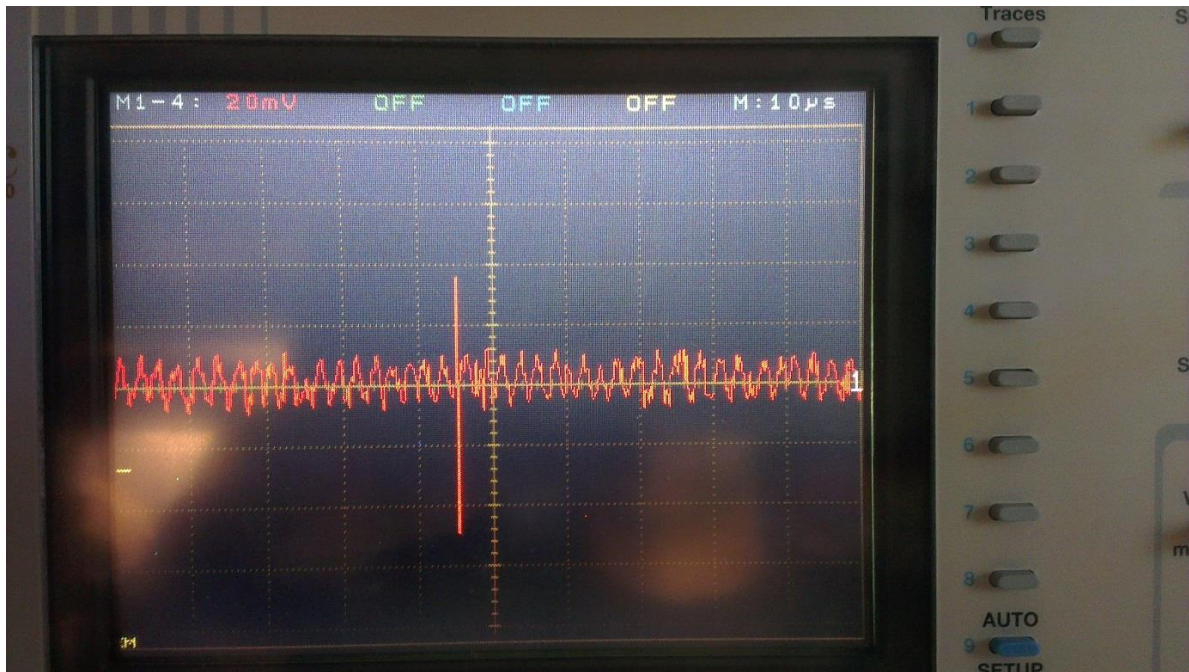


Abbildung 21: Spannungsverlauf nach einbau im Hexacopter

6.2 EMV

Nachdem das Spannungsproblem, welches in Kapitel 6.1 beschrieben wurde behoben war, machte der APM bei der selben Prozedur einen Neustart. Nach ausgiebiger Analyse konnte ein EMV-Problem festgestellt werden. Wie in Abbildung 22 eingezeichnet ist wurde im Konzept keine Aufmerksamkeit auf Schleifen in der Stromversorgung gelegt.

Verbesserungen und Ausblick

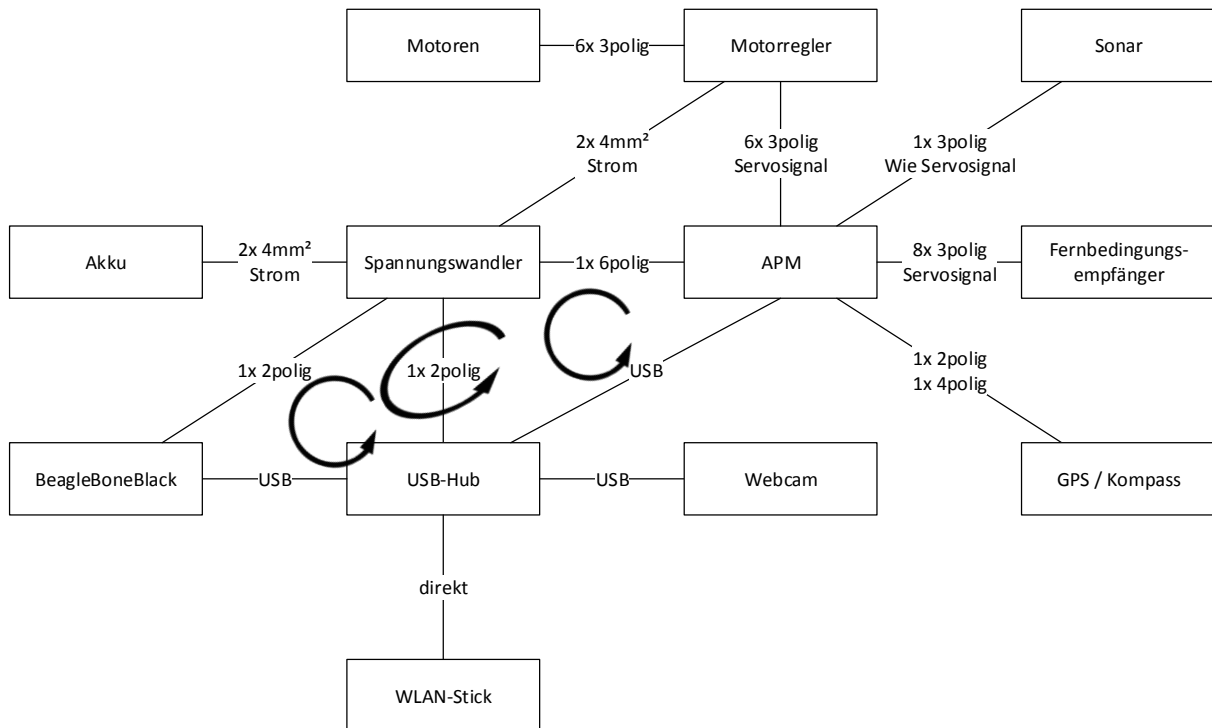


Abbildung 22: Aufgespannte Schleifen im Konzept

Um diese Schleifen zu unterbrechen sind galvanische Trennungen in den USB-Leitungen nötig. Eine galvanische Trennung der Spannungsversorgungen wäre um Größenordnungen aufwendiger. Um USB-Datenverbindungen galvanisch zu trennen können Baugruppen mit dieser Funktion erworben werden. Zwischen Spannungswandler, APM und den Motorreglern existieren keine Schleifen, da die verwendeten Motorregler auf der Servosignal-Seite optisch entkoppelt sind.



Abbildung 23: USB Trennfilter

Verbesserungen und Ausblick

Abbildung 23 zeigt eine solche Baugruppe, mit der die USB-Datenverbindungen galvanisch entkoppelt werden können. Durch einbinden solcher Module in die Datenleitung zwischen Beagle Bone Black und USB-Hub, sowie zwischen USB-Hub und APM können beide Schleifen unterbrochen werden. Für den Zweitflug wurde eine andere Variante der galvanischen Entkopplung gewählt, da eine solche Baugruppe zu diesem Zeitpunkt nicht verfügbar war. Dabei wurden die USB-Leitungen aufgetrennt und ein $470\ \Omega$ Widerstand als Strombegrenzung in die Masseleitung eingebaut. Für die steilflankigen Datensignale wurde noch ein $100\ \text{nF}$ Kondensator parallel zum Widerstand angebracht, damit keine Datenverluste im Flug auftreten. Abbildung 24 zeigt diese Modifikation an der Hardware. Nach diesen Änderungen traten keine Probleme bei der Inbetriebnahmen des Multikopters auf.



Abbildung 24: Provisorische Umsetzung zur Problemlösung

6.3 Flugleistung

Bei den Flugversuchen fiel auf, dass der Multicopter mit allen Subsystemen und der Payload über zu wenig Leistungsreserven verfügt um auch für zukünftigen Missionen und zusätzliche Payload gerüstet zu sein. Daher sollte zunächst die Flugleistung verbessert werden. Dazu wurde zusammen mit dem Struktur-Subteam nach leistungsfähigeren Antriebssträngen gesucht. Nähere Informationen zur Flugleistungssteigerung sind in der Arbeit des Struktur-Subteams zu finden.

6.4 Flugrechner

Es wurde festgestellt, dass die in dieser Arbeit verwendete Hardware, der APM 2.6, in der aktuellsten Softwareversion nicht mehr unterstützt wird. Dies ist dadurch begründet, dass der APM 2.6 nur mit einem 8 Bit Mikrocontroller von Atmel mit einer Taktrate von maximal 16 MHz ausgestattet ist. Dessen Rechenleistung ist für zukünftige Versionen der Ardupilot-Software nicht mehr ausreichend.

Um auch die aktuelle Ardupilot-Software verwenden zu können sollte daher langfristig die Hardware gewechselt werden. Die Ardupilot Software wurde bereits auf zwei weiteren Hardware-Plattformen portiert.

Zum einen existiert die Plattform „Pixhawk“. Diese verfügt über einen ARM Cortex M4F mit einer Taktrate von bis zu 168 MHz. [16] Durch diesen Mikrokontroller ist der Pixhawk wesentlich performanter als der APM2.6. Nähere Informationen sind auf der Pixhawk Homepage zu finden. [17]

Zum anderen die Plattform „PX4“. Diese verfügt über einen etwas leistungsschwächeren ARM-Prozessor als der Pixhawk. Dafür ist der PX4 der kompakteste und leichteste der drei verfügbaren Flugrechner.

Verbesserungen und Ausblick

Des Weiteren zeigte sich, dass die eingestellten Failsafemodes des APM sich gegenseitig beeinflussen können. Damit gewährleistet werden kann, dass der Pilot jederzeit die Kontrolle über das Fluggerät behält, ist in dieser Studienarbeit die Idee eines redundanten Flugrechners entstanden. Als zweiter Flugrechner könnte eine einfache Plattform verwendet werden.

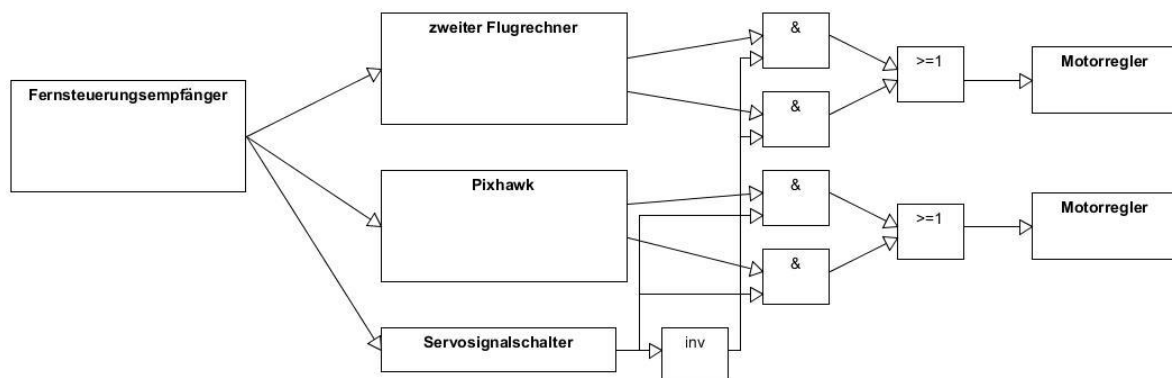


Abbildung 25: Konzept redundanter Flugrechner

Das Ausgangssignal des Fernsteuerungsempfängers könnte dabei auf beide Flugregler gegeben werden. Die Ausgänge der Flugregler könnten über logische Gatter voneinander entkoppelt werden. Ein Signal der Fernsteuerung könnte dann zwischen den Signalen der Flugregler auf die Motorregler hin und her Schalten. Ein schemenhaftes Konzept dieses Aufbaus wird in Abbildung 25 dargestellt.

7 Ergebnis

Abschließend ist die vorliegende Studienarbeit ein Grundstein für zukünftige Generationen von Studienarbeiten. Es wurde viel Konzeptarbeit geleistet, auf die in Zukunft aufgebaut werden kann.

Als finales Resultat dieser Studienarbeit konnte eine funktionierende, flugerprobte Multicopterplattform vorgestellt werden. Dazu wurden diverse Subsysteme konzipiert und passendes Equipment ausgewählt. Diese Subsysteme wurden so vernetzt und konfiguriert, dass die Funktion als Gesamtes gegeben ist. Für dieses flugfähige Gerät wurden alle nötigen Dokumente und Zulassungen erstellt, sodass ein in Deutschland legaler und sicherer Betrieb möglich ist. Der erste Schritt ist somit getan.

In Zukunft kann die vorgestellte Plattform in weiteren Studienarbeiten noch in Richtung Autonomie und Schwarm weiterentwickelt werden.

8 Literaturverzeichnis

- [1] „Finding Dulcinea - Librarian of the Internet,“ [Online]. Available: <http://www.findingdulcinea.com/news/on-this-day/July-August-08/On-this-Day-Austria-Rains-Balloon-Bombs-on-Venice.html>. [Zugriff am 28 06 2015].
- [2] „Vision Systems,“ [Online]. Available: <http://www.vision-systems.com/articles/2013/10/uavs-guide-students-around-mit-campus.html>. [Zugriff am 28 06 2015].
- [3] „Vision Systems,“ [Online]. Available: <http://www.vision-systems.com/articles/2013/07/u-s-coast-guard-makes-first-drug-bust-using-uavs.html>. [Zugriff am 28 06 2015].
- [4] „Amazon,“ Amazon, [Online]. Available: <http://www.amazon.com/b?ie=UTF8&node=8037720011>. [Zugriff am 29 06 2015].
- [5] microcontroller.net, „Modellbauservo Ansteuerung,“ [Online]. Available: http://www.mikrocontroller.net/articles/Modellbauservo_Ansteuerung. [Zugriff am 9 Juli 2015].
- [6] Graupner GmbH & CO. KG, „HoTT Hopping. Telemetry. Transmission,“ 10 Oktober 2010. [Online]. Available: <http://www.graupner.de/de/newsdetail/ea763d32-f660-45d2-bc16-f9c5adb83359>. [Zugriff am 7 Juli 2015].
- [7] Graupner GmbH und Co. KG, „Computer System MC-32HoTT Programmier-Handbuch,“ 22 Januar 2014. [Online]. Available: http://www.graupner.de/mediaroot/files/33032_mc32_HoTT_DE.7.pdf. [Zugriff am 7 Juli 2015].
- [8] Graupner GmbH & Co. KG, „Fernsteuerungen 2,4 GHz mit Telemetrie und Sprachausgabe,“ 2012. [Online]. Available: http://www.graupner.de/filerootdir/catalogs/DZ10607_HoTT_Flyer_2012_screen.pdf. [Zugriff am 7 Juli 2015].

Literaturverzeichnis

- [9] Wikipedia, „Ardupilot,“ [Online]. Available: <https://en.wikipedia.org/wiki/Ardupilot>. [Zugriff am 06 Juli 2015].
- [10] „BeagleBord,“ [Online]. Available: <http://beagleboard.org/BLACK>. [Zugriff am 07 Juli 2015].
- [11] BeagleBoard.org, „BeagleBoneBlack,“ [Online]. Available: http://elinux.org/Beagleboard:BeagleBoneBlack#Revision_C_.28Production_Version.29. [Zugriff am 07 Juli 2015].
- [12] „Communicating with Raspberry Pi via MAVLink,“ [Online]. Available: <http://dev.ardupilot.com/wiki/companion-computers/raspberry-pi-via-mavlink/?lang=de>. [Zugriff am 24 Mai 2015].
- [13] „UAV ground station software package for MAVLink based systems,“ [Online]. Available: <http://tridge.github.io/MAVProxy/>. [Zugriff am 10 Mai 2015].
- [14] Loughborough University Centre for Autonomous Systems, „Lucas-research APMSimulink,“ 27 Dezember 2011. [Online]. Available: <https://code.google.com/p/lucas-research/wiki/APMSimulink>. [Zugriff am 8 Juli 2015].
- [15] „Maxbotix Analog Sonar – Copter3.2,“ [Online]. Available: <http://copter.ardupilot.com/wiki/common-optional-hardware/common-rangefinder-landingpage/common-rangefinder-maxbotix-analog/>. [Zugriff am 24 MAi 2015].
- [16] APM Copter, „Choosing a flight controller,“ [Online]. Available: <http://copter.ardupilot.com/wiki/introduction/choosing-a-flight-controller/>. [Zugriff am 8 Juli 2015].
- [17] „PX4 Autopilot,“ [Online]. Available: <http://pixhawk.org/start>. [Zugriff am 7 Juli 2015].
- [18] QGROUNDCONTROL, „MAVLink Micro Air Vehicle Communication Protocol,“ [Online]. Available: <http://qgroundcontrol.org/mavlink/start>. [Zugriff am 5 Mai 2015].

- [19] S. Balasubramanian, „MavLink Tutorial for Absolute Dummies,“ 2015. [Online]. Available: http://dev.ardupilot.com/wp-content/uploads/sites/6/2015/05/MAVLINK_FOR_DUMMIESPart1_v.1.1.pdf. [Zugriff am 24 Mai 2015].
- [20] „Analog Sonar (AC3.1),“ [Online]. Available: <http://copter.ardupilot.com/wiki/common-optional-hardware/common-rangefinder-landingpage/sonar/>. [Zugriff am 24 Mai 2015].
- [21] „XL-MaxSonar - EZ Series,“ [Online]. Available: http://www.maxbotix.com/documents/XL-MaxSonar-EZ_Datasheet.pdf. [Zugriff am 24 Mai 2015].