SOFTWARE DEFINED RADIO ALS TEIL EINER INTEGRIERTEN MODULAREN AVIONIK FÜR REMOTELY PILOTED AIRCRAFT SYSTEMS

K. Kainrath, M. Gruber, H. Flühr
FH JOANNEUM, Institut Luftfahrt, Alte Poststraße 149, 8020 Graz, Österreich

Zusammenfassung

Im Forschungsprojekt OMOSA (Open Modular/Open Source Avionics Architecture for Remotely Piloted Aircraft Systems – RPAS) wird versucht, das Prinzip einer Integrierten Modularen Avionik (IMA) für RPAS bzw. UAS (Unmanned Aerial/Aircraft Systems) umzusetzen. Dafür soll quelloffene Hard- und Software zum Einsatz kommen. Für eine in weiterer Folge mögliche Zulassung wird dazu auch ein entsprechendes Redundanzkonzept überlegt und implementiert. Die OMOSA-Architektur umfasst ein Flugsteuerungsmodul, ein Navigationsmodul sowie ein Kommunikationsmodul. Dieses kann einen Command & Control- (C2)- als auch einen Payload-Datenlink integrieren. Je nach Betrieb werden unterschiedliche Anforderungen und Ausfallssicherheiten von der Kommunikationseinheit verlangt.

Wenn man die Kommunikationseinheit als Software Defined Radio (SDR) umsetzt, gewinnt man an Variabilität und Flexibilität und kann so der Ausfallswahrscheinlichkeit entgegenwirken. Man wäre in der Lage, ohne die Hardware zu verändern, Sendefrequenzen, Modulationsarten, Verschlüsselungen sowie Sendeleistungen während des Fluges zu ändern und somit an die jeweiligen Anforderungen anzupassen.

Diese Arbeit beschäftigt sich mit der Erforschung von Einsatzmöglichkeiten quelloffener SDR Module für den Betrieb von RPAS im Sichtbereich (Line Of Sight – LOS) und außerhalb (Beyond Line Of Sight). Dabei sollen auch die rechtlichen Rahmenbedingungen im Auge behalten werden.

1. EINLEITUNG

In Österreich ist der Betrieb von unbemannten Luftfahrzeugen laut Novelle des Luftfahrtgesetzes (LFG) seit dem 1.1.2014 möglich. Die hierfür zuständige Behörde ist die Austro Control [1]. Hierbei wird in zwei Kategorien für die Zulassung unterteilt:

Klasse 1 umfasst Luftfahrzeuge bis max. 150 kg, welche nur mit Sichtkontakt bis zu einer Höhe von max. 150 m verwendet werden dürfen. Zur Klasse 2 zählen alle weiteren Luftfahrzeuge für die keine Sichtverbindung erforderlich ist. Diese werden wie Zivilluftfahrzeuge zertifiziert und zugelassen.

Da es für kleinere Luftfahrtbetriebe kostenintensiv ist, Klasse 1 UAVs (Unmanned Aerial Vehicles) zuzulassen, wäre es wünschenswert, einheitliche und leicht zu zertifizierende Komponenten zur Verfügung zu haben. Im Bereich der Avionik würde dies eine modulare Architektur bedeuten.

Diese Architektur ist an den Aufbau der Integrierten Modularen Avionik, kurz IMA [2], angelehnt. Dabei verfolgt man den Ansatz, die Elektronik eines Flugzeugs mit standardisierter Hard- und Software zu realisieren. Im Forschungsprojekt OMOSA (Open Modular/Open Source Avionics Architecture for Remotely Piloted Aircraft Systems – RPAS) wird so eine Architektur auf die speziellen Rahmenbedingungen von unbemannten Flugsystemen der Klasse 1 erforscht. Verfügbare Echtzeitbetriebssysteme und Hardwaremodule unterschiedlicher Leistungsfähigkeit werden bewertet und mittels einer Backplane mit zuverlässiger Querkommunikation in ein Gesamtsystem integriert. Ziel dieses Vorhabens ist es, eine IMA Plattform für leichte RPAS der Klasse 1 zu entwickeln, die einen geringeren Aufwand für die Implementierung neuer Funktionen erfordert.

2. MODULARE ARCHITEKTUR MIT SOFTWARE DEFINDED RADIO

Für die Umsetzung einer modularen Open Source Architektur wurden unterschiedliche Systeme recherchiert und beurteilt. Am effizientesten in Bezug auf Leistung, Gewicht und Kosten erwiesen sich Einplatinenrechner wie der Raspberry Pi 2 oder der Odroid C1 (BILD 1). Diese kleinen Rechnerplatinen wiegen nur etwa 40 g und verfügen über genügend Rechenleistung. Sie benötigen im Betrieb maximal 4 W. Dabei läuft eine große Palette an Open-Source Software auf ihnen. Neben einigen Linux Distributionen, läuft auch BSD und Windows 10 IoT Core. Weiters besitzen der Raspberry Pi 2 sowie auch der Odroid C1 4 USB 2.0 Ports, eine Ethernet Schnittstelle sowie 40 GPIO Pins, von denen einige z.B. via I²C oder SPI kommunizieren können.

doi:10.25967/370286

BILD 1. Einplatinenrechner Odroid C1 [3]

Um diese Einplatinenrechner für RPAS einsetzen zu können, muss ein entsprechendes Redundanzkonzept überlegt werden. Im Projekt OMOSA wurde daher ein Konzept entwickelt, das einen unabhängigen Fehler im System erkennen kann. So kann im Falle eines Fehlers das System deaktiviert und gleichzeitig auf ein Backup System umgeschalten werden. Die Basisversion dieser Architektur ist in BILD 2 dargestellt. Dieses System besteht aus einem Kontrollrechner (Control), der die Steuerdaten von einem Software Defined Radio (SDR) von der Bodenstation (Ground Control Station – GCS) empfängt und die Telemetriedaten darüber zur GCS sendet. Die Steuerdaten werden in der Control Einheit dekodiert und digital an einen Arduino Mikrocontroller weitergeleitet. Dieser steuert schließlich die Servos an. Eine doppelt ausgeführte Monitor Einheit vergleicht dabei jeweils das Steuersignal (SOLL) mit dem Servofeedback (IST). Treten hier Abweichungen außerhalb der Toleranz auf, wird sofort auf ein Backup System umgeschalten. Dieses besteht aus einem Pixhawk PX4 Autopiloten mit einem Empfänger und integrierten Telemetriesensoren und übernimmt ab dem Fehlerfall die Steuerung der Servos.

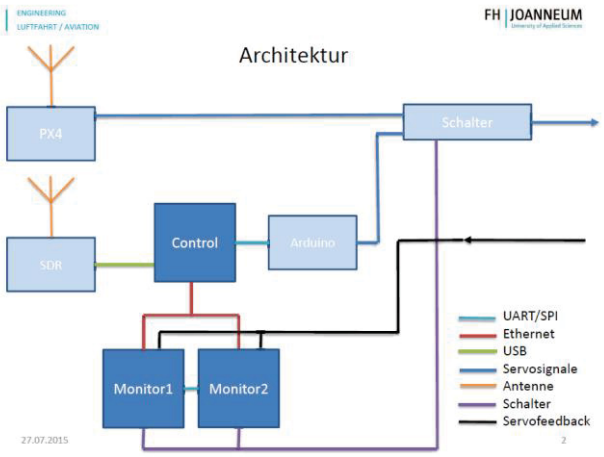

BILD 2. Architektur OMOSA

2.1. Software Defined Radio

Ein klassisches unidirektionales Funk-Übertragungssystem besteht aus einem Sender und einem Empfänger. Ein Basisband Signal, das kann z.B. Stimme bei Telefonie, oder ein Fernsteuersignal eines Modellflugzeuges sein, wird auf ein hochfrequentes (HF) Trägersignal aufmoduliert. Dieses HF Signal wird dann an eine Antenne geleitet und wird dort als elektromagnetische Welle abgestrahlt. Am Empfänger wird die Welle über die Antenne wiederum empfangen und das Signal demoduliert. Um die Übertragung zu verbessern, kann man entsprechende Kodierungen, wie Quellen- oder Kanalkodierung vornehmen und so dem HF Signal eine Spezielle Wellenform geben. Diese Systeme kann man für ihre jeweilige Aufgabe optimieren. Betrachtet man das im OSI Modell [4] erkennt man, dass hier die unteren vier Schichten, die man als Physical Layer bezeichnet, als starre Hardware implementiert sind. Der Unterschied dieses Funk-Übertragungssytems zu einem Software Defined Radio (SDR) [5] besteht nun darin, einige, oder alle dieser Physical Layer Funktionen in Software zu realisieren. Das bedeutet, man kann per Software eine beliebige Wellenform realisieren.

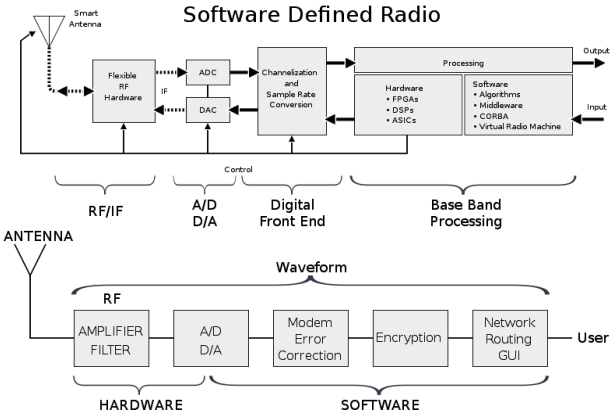

BILD 3. Software Defined Radio Prinzip [6]

Ein SDR besteht meist aus einem FPGA (Field Programmable Gate Array), einem CPLD (Complex Programmable Logic Device) oder einem DSP (Digital Signal Processor), die über Software (oder auch Firmware), das Basisbandsignal kodieren. Das kodierte digitale Signal wird anschließend kanalkodiert und erst dann in ein analoges Signal umgesetzt. Am Ende ist ein flexibles HF Front End, welches das analoge Signal auf die Trägerfrequenz aufmoduliert. So kann man je nach Frequenzbereich des HF Front Ends und der entsprechenden Auflösung der ADU (Analog Digital Umsetzer) und DAU (Digital Analog Umsetzer) unterschiedliche Funk Anwendungen per Software auf ein und derselben Hardware realisieren.
Ein in seiner Funktion eingeschränktes SDR wäre ein sogenanntes Software Controlled Radio (SCR). Hier kann man z.B. nur zwischen bestimmten Trägerfrequenzen wechseln. Ein Mobiltelefon, das per Software zwischen 2G und 3G Empfang wechselt wäre so ein SCR. Sind jedoch auch Kodierungen und/oder Modulation/Demodulation frei konfigurierbar, spricht man von einem SDR [5].

2.2. SDR Komponenten

Durch die kontinuierliche Weiterentwicklung von Software Defined Radios gibt es mittlerweile einige kommerziell verfügbare SDR Komponenten. Wurden vor einigen Jahren noch PC Soundkarten als SDR Empfänger verwendet, gibt es mittlerweile sehr potente und leistbare Open Source

Hardware. Besonders stechen zwei über Kickstarter finanzierte Projekte hervor: HackRF [7] und BladeRF [8]. Weiters empfehlenswert ist die USRP Serie von Ettus [9], die schon sehr lange SDR Komponenten anbieten. Da das HackRF und das BladeRF beide leicht, kosteneffizient, per USB betreibbar sind und einen großen Frequenzbereich abdecken wurden diese Module verwendet. In der folgenden Tabelle sieht man einen Vergleich der Spezifikationen:

	HackRF	BladeRF
Frequenzspektrum	30 MHz – 6 GHz	300 MHz – 3.8 GHz
Bandbreite	20 MHz	28 MHz
Duplex	Half	Full
Sample Size	8 Bit	12 Bit
Sample Rate	20 Msps	40 Msps
Interface (Speed)	USB 2.0 (480 MBit)	USB 3.0 (5 GBit)
Microcontroller	LPC43XX	Cypress FX3
FPGA/CPLD	CPLD	Altera Cyclone IV
Open Source	full	HDL, Code, Schematics

TAB 1. Vergleich HackRF und BladeRF [10]

Ein Vorteil des HackRF ist definitiv das sehr breite Frequenzspektrum, das es abdeckt. Da im HackRF jedoch unterschiedliche Frontend Komponenten arbeiten, steht nicht über den gesamten Frequenzbereich die gleiche Bandbreite zur Verfügung. Zumindest 20 MHz sollte die Bandbreite bei jeder Frequenz betragen. Das Modul kann nur halbduplex arbeiten, das heißt es kann entweder nur Senden, oder nur Empfangen. Das Umschalten sollte im Millisekunden-Bereich machbar sein.

BILD 4. HackRF One [7]

Das BladeRF verwendet nur ein spezielles sehr breitbandiges Frontend. Deshalb kann es die Bandbreite auch über das gesamte Spektrum garantieren. Es arbeitet voll-duplex, kann also gleichzeitig senden und empfangen und hat die besseren AD und DA Umsetzer verbaut. Dafür kann es „nur“ bis 3,8 GHz eingesetzt werden.

BILD 5. BladeRF [8]

Für die Kommunikation mit den SDR Modulen sind besonders die freien ISM Frequenzbänder von Interesse. Zur Auswahl stehen hier 433 MHz, 868 MHz, 2,4 GHz und 5 GHz. In Kürze könnte auch bei etwa 5,1 GHz dedizierter Frequenzbereich speziell für RPAS [11] zur Verfügung stehen.

2.3. SDR Software

Um die SDR Module entsprechend zu nutzen wird das auf Python und C++ basierende Open Source Entwicklungs Toolkit GNURadio [12] verwendet. Es läuft unter Linux und MacOS problemlos, lediglich unter Windows ist die Installation recht komplex. Die Software unterstützt eine große Palette von externer RF Hardware, kann aber auch rein für Simulationen benutzt werden. Es gibt auch eine grafische Benutzeroberfläche, genannt GNURadio Companion, welche die Bedienung erleichtert (BILD 6). Diese Oberfläche funktioniert ähnlich wie MATLAB Simulink. Durch das Hinzufügen, Editieren und Verbinden von Funktionsblöcken entwirft man die gewünschte Funktion.

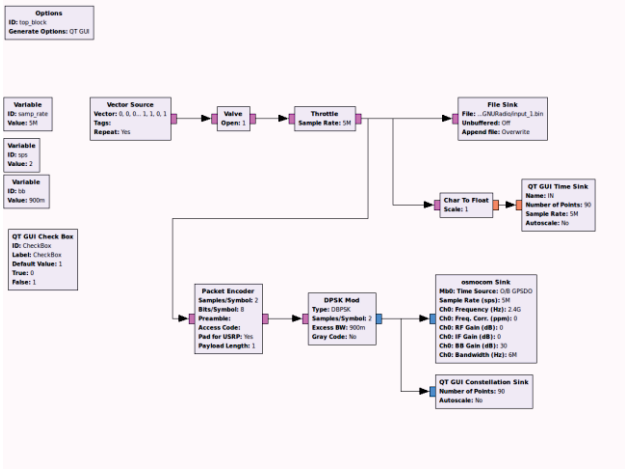

BILD 6. GNURadio Companion Oberfläche [13]

2.4. Abschätzung des Übertragungskanals

Die Übertragung des Funksignals für den Command and Control und den Telemetrie Link erfolgt durch eine direkte Verbindung. Das bedeutet, dass das elektromagnetische (EM) Signal primär eine Freiraumdämpfung (Free Space Path Loss – FSPL) erfährt. Diese wird als inverse

Verstärkung ausgedrückt. Weiters wird auch nicht das Nahfeld einer Antenne betrachtet, sondern rein das Fernfeld, bei dem eine kugelförmige Ausbreitung angenommen wird. Die Ausbreitung der EM Welle im Raum kann dann annäherungsweise durch das Abstandsgesetz (isotroper Kugelstrahler [14]) beschrieben werden. Die Empfangsantenne ist durch ihre Apertur ($A_{W,E}$) frequenzabhängig. BILD 7 zeigt die ideale Freiraumübertragung:

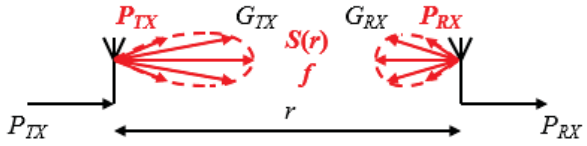

BILD 7. Übertragungskanal [14]

Mathematisch ausgedrückt berechnet sich die mittlere Signalleistungsdichte in Abhängigkeit zum Abstand S(r) zu:

(1) $S(r) = G_{TX}\frac{P_{TX}}{4\pi r^2}$

G_{TX} steht für den Sendergewinn und P_{TX} für die Sendeleistung. Am Empfänger errechnet sich die empfangene Leistung zu:

(2) $P_{RX} = \frac{G_{TX}P_{TX}A_{W,E}}{4\pi r^2}$ wobei

(3) $A_{W,E} = \frac{\lambda^2}{4\pi}G_E$

ist. Wandelt man diese Gleichung um, erhält man den FSPL:

(4) $L_{Free\ Space} = \left(\frac{4\pi r}{\lambda}\right)^2 = L_{F0}$

Umgeformt ausgedrückt kann man die Dämpfung leicht in Dezibel (dB) berechnen:

(5) $L_{F0}(dB) = 20log_{10}(r) + 20log_{10}(f) - 147.55$

Bei idealen, direkten Verbindungen kann die obige Gleichung verwendet werden. Unter Realbedingungen erfährt das Signal jedoch äußere Störeinflüsse wie atmosphärische Absorptionen (Regen, Nebel), Reflexionen z.B. vom Erdboden, Ablenkungen und Zerstreuungen z.B. von Wolken (BILD 8).

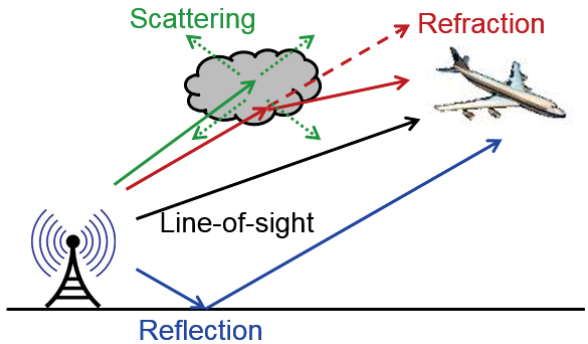

BILD 8. Reale Freiraumübertragung [14]

So kommt es, dass das EM Signal einerseits positiv, aber auch negativ durch die Reflexionen beeinflusst wird. Die reflektierten Signale können das Line of Sight Signal überlagern und können es so einerseits verstärken, andererseits jedoch auch dämpfen. Dieser Vorgang kann durch den 2 Wege Path Loss mathematisch beschrieben werden. Bei einer idealen Reflexion am Grund (r ≈ -1) kann der gesamte Path Loss berechnet und mit dem Free Space Path Loss (FSPL) L_{F0} verglichen werden. Durch die positiven Überlagerungen entstehen Werte rund +6 dB höher als L_{F0}. (Bild 10) Die Minimalwerte würden idealerweise den Wert 0 dB annehmen. Für die Berechnung betrachtet man nun die Skizze in BILD 9:

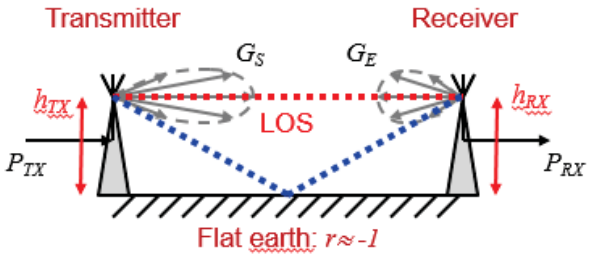

BILD 9. 2 Way Path Loss [14]

Mit der folgenden Formel (6) kann man den 2 Way Path Loss berechnen, wobei h_{TX} und h_{RX} die Höhen des Senders und Empfängers sind. Mit Formel (7) kann man die Asymptote dazu berechnen:

(6) $L_{2-way} = 10log_{10}\left(2\left(\frac{c_0}{4\pi df}\right)^2\left(1 - cos\left(\frac{4\pi h_{TX}h_{RX}f}{dc_0}\right)\right)\right)$

(7) $L_{2-way} = 20log_{10}\left(\frac{h_{TX}h_{RX}}{d^2}\right)$

Stellt man L_{F0} und L_{2-way} samt Asymptote grafisch dar, erhält man für eine Frequenz von 2,4 GHz und eine Sender- und Empfängerhöhe von 3 m folgende Skizze:

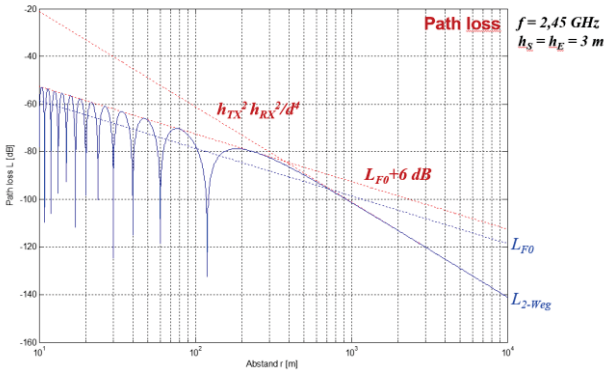

BILD 10.L_{F0}, L_{2-way} und Asymptote [14]

Für die Klasse 1 und eine Distanz von 150 m würde sich eine Dämpfung von etwa 80 dB ergeben. Hier wurden ausschließlich nur die Ausbreitungseigenschaften der EM Welle betrachtet. Man kann die Übertragungsqualität durch Kodierung und Modulation wesentlich verbessern. Trotzdem kann man so eine grundlegende Abschätzung der erforderlichen Sendeleistungen durchführen.

3. SYSTEMAUFBAU UND TESTVERSUCHE

In diesem Abschnitt werden die ersten Testversuche mit dem SDR Modulen sowie der Aufbau der OMOSA Architektur präsentiert. Bevor die Systeme zusammenarbeiten, müssen die einzelnen Funktionen und Szenarien erfolgreich getestet werden

3.1. Aufbau der OMOSA Architektur

Für den Aufbau aus BILD 11 wurden 3 Odroid C1 Einplatinenrechner, ein Arduino UNO, ein Pixhawk PX4 mit Empfänger und ein Switch verwendet. Auf den Odroids läuft ein Lubuntu 14.04. Darauf wurde in C++ die jeweilige Software entwickelt. Der Arduino wurde in der Arduino Oberfläche programmiert. Der PX4 wird mit Standardfirmware bespielt und dient im Fehlerfall als Backupsystem. Dieser Aufbau kann erfolgreich einen unabhängigen Fehler entdecken und dann auf das Backup System mit dem Pixhawk umschalten. Der Aufbau ist in BILD 11 dargestellt:

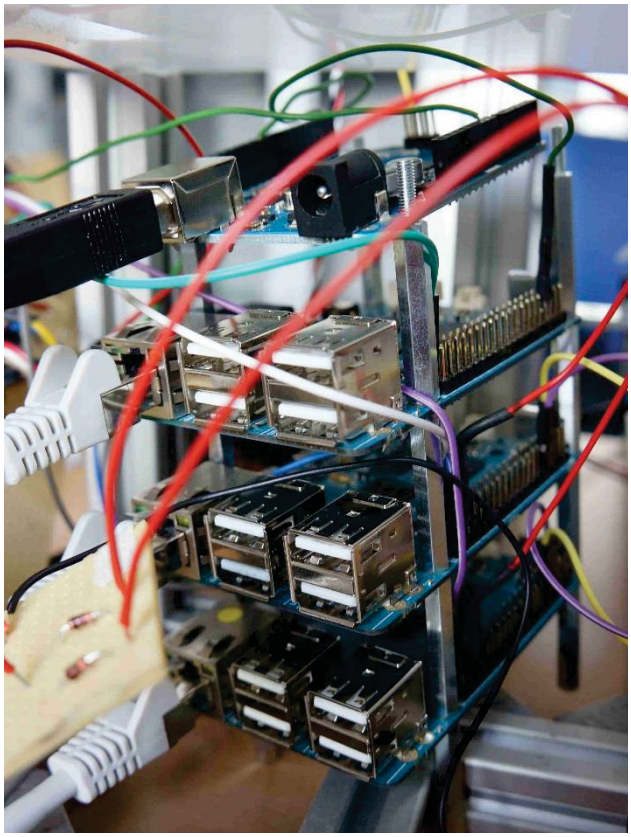

BILD 11.OMOSA Architektur

3.2. SDR Tests mit HackRF One

Da das BladeRF nur bis zu einem Frequenzbereich von 3,8 GHz einsetzbar ist, wurde entschieden, das HackRF zu bevorzugen. Dafür wurde ein erster Testaufbau mit zwei Modulen aufgebaut. Gesteuert wurden sie SDR Module von zwei Linux Rechnern, auf denen ein Ubuntu 14.04 läuft. Es wurde jeweils GNU Radio und alle Treiber installiert. Der Aufbau ist in BILD 12 zu sehen. Für die ersten Tests wurde der Sendeausgang per Kabel und Dämpfungsglied mit dem Empfängereingang verbunden. Im zweiten Schritt wurde über eine Funkverbindung getestet.

BILD 12.SDR Testaufbau

Es wurde ein Sender- und Empfängerskript für eine einfache DBPSK Übertragung bei 2,4 GHz mit einer Bandbreite von 6 MHz geschrieben. Übertragen wurde ein einfacher Vektor aus einer zuvor festgelegten Reihenfolge aus 0 und 1. Dieser wurde immer wieder gesendet. Ein Vergleich des Eingangs mit dem Ausgang zeigte die erfolgreiche Übertragung wie in BILD 13 zu erkennen ist:

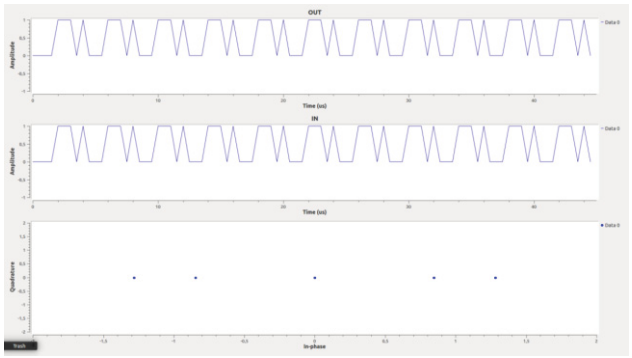

BILD 13.Input und Outputplot des Datenvektors

Abschließend wurde das Sendesignal mit dem Oszilloskop überprüft. Man kann die Phasensprünge der DPSK gut erkennen:

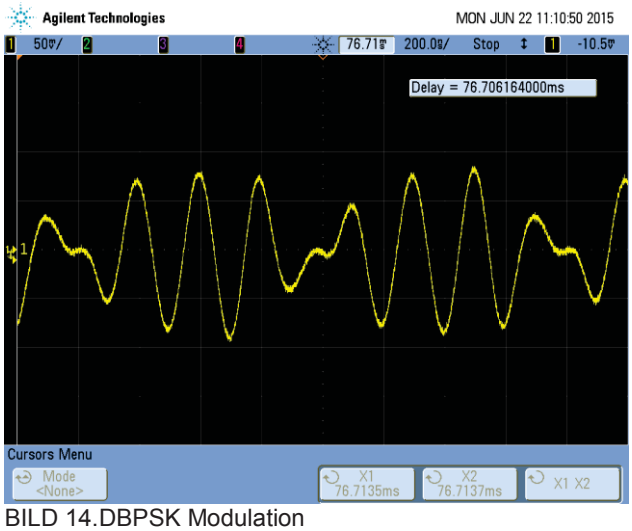

BILD 14.DBPSK Modulation

4. AUSBLICK

Die bisherigen Tests verliefen erfolgreich. Während die OMOSA Architektur soweit fertig ist, müssen für das SDR noch weitere Tests durchgeführt werden. Die Abschätzung des Übertragungskanals muss überprüft werden. Weiters gilt es unterschiedliche Modulationsarten und Kanalkodierungen zu testen. Um die Sicherheit des Command and Control (C2) Links zu gewährleisten, kann eine AES Verschlüsselung eingesetzt werden. [15] Es ist auch erforderlich, eine Reichweitenabschätzung durchzuführen. Im LOS Bereich für die Klasse 1 dürfte die vorhandenen Sendeleistung der SDR Module ausreichend sein. Es muss aber eruiert werden, ob ein optionales zusätzliches Front End die Sicherheit erhöht. Spannend bleibt die Frage, ob und wann ein dediziertes Frequenzband für UAVs [16] festgelegt wird. Der nächste Schritt ist dann das Zusammenführen der beiden Systeme. Optional können noch weitere Komponenten wie etwa ein FMCW Höhenradar [17] oder zusätzliche Navigationssensoren der OMOSA Architektur wiederum modular hinzugefügt werden.
Mit dieser Architektur hat man eine einheitliche modulare und hoch anpassbare Plattform für viele unterschiedliche RPAS Typen der Klasse 1.

Danksagung

Die Autoren bedanken sich hiermit bei der österreichischen Forschungsförderungsgesellschaft (FFG) für die finanzielle Unterstützung. Diese Arbeit wurde durch das Luftfahrtprogramm „TAKE-OFF“ finanziert.

Referenzen

[1] Austro Control, LBTH67, Betrieb von unbemannten Luftfahrzeugen https://www.austrocontrol.at/luftfahrtbehoerde/lizenzen__bewilligungen/flugbewilligungen/unbemannte_lfz

[2] B. Balser, M. Förster, G. Grabowski, Modulare Avionik als Grundlage für Systemdefinition, Systemkonfiguration und Systemkontrolle Control, EADS Deutschland GmbH

[3] Hardkernel, Odroid C1, http://www.hardkernel.com/main/products/prdt_info.php?g_code=G143703355573

[4] Carsten Roppel, Grundlagen der digitalen Kommunikationstechnik,, Fachbuchverlag Leipzig, 2006

[5] Eugene Grayver, Implementing Software Defined Rado, Springer, 2013

[6] Topi Tuukkanen, Software Defined Radio, https://commons.wikimedia.org/wiki/File:SDR_et_WF.JPG

[7] Great Scott Gadgets, HackRF One, Open Source SDR Plattform https://greatscottgadgets.com/hackrf/

[8] Nuand, bladeRF – the USB3.0 superspeed SDR, http://nuand.com/

[9] Ettus Research, A National Instruments Company http://www.ettus.com/

[10] Taylor Killan, SDR Showdown, HackRF vs Blade RF vs USRP http://www.taylorkillian.com/2013/08/sdr-showdown-hackrf-vs-bladerf-vs-usrp.html

[11] BMVIT, Nationale Frequenznutzung http://www.bmvit.gv.at/bmvit/telekommunikation/funk/frequenzverw/natplan/index.html

[12] GNURadio, The Free & Open Source Software Defined Radio http://gnuradio.org/redmine/projects/gnuradio/wiki

[13] Norman-Elvenich, Riedler, Software Defined Radio, Advanced Engineering Project, 2015

[14] Holger Flühr, Flugsicherungstechnik, FH JOANNEUM GesmbH, 2014

[15] Creating Secure C2 Links for UAVs, Global Aerospace Technology Network http://www.intelligent-aerospace.com/articles/2014/02/secure-c2-uav.html

[16] Europäische Kommision, Bestandsaufnahme der Funkfrequenzen, Brüssel 2014 http://www.parlament.gv.at/PAKT/EU/XXV/EU/03/61/EU_36112/imfname_10489511.pdf

[17] Holger Flühr, Avionik und Flugsicherungstechnik, Springer, Auflage 2, 2013