802.11p - Umsetzung

Dominik Bayerl

2. Umsetzung

Im Rahmen des Projekts wurden die zwei grundsätzlichen Funktionen umgesetzt, die zur Nutzung der Hardware für weitere Experimente erforderlich sind. Dies ist einerseits die Erweiterung des 802.11-Reference-Designs auf den 802.11p-Standard und andererseits die Implementierung einer Ethernet-Schnittstelle zum Empfang und Senden von Daten über einen angeschlossenen Computer. Auf beide Funktionen soll im Folgenden kurz eingegangen werden.

2.1. WARP Reference Design für 802.11p

Auf die grundsätzliche Funktionsweise von *802.11p* wurde bereits in **TODO: Referenz 802.11p** eingegangen. Dabei wurde deutlich, dass der Standard sehr ähnlich zum bereits implementierten *802.11a* ist (insbesondere die Orthogonal Frequency-Division Multiplexing (OFDM)-Waveform) und sich vor allem in zwei wesentlichen Merkmalen, den Channel-Frequenzen und der Channel-Bandbreite unterscheidet. Es bietet sich daher an, die Implementierung auf Basis des vorhandenen Frameworks vorzunehmen.

2.1.1. Channel-Frequenzen

Für das Teilnehmer-Multiplexing in WLAN-Funksystemen werden üblicherweise Channels verwendet, d.h. es können mehrere getrennte Funknetze dadurch unabhängig voneinander existieren, indem sie verschiedene Channels und dadurch verschiedene Frequenzen für die Kommunikation nutzen. Im 802.11p Standard sind acht verschiedene Channel-Typen spezifiziert, die für unterschiedliche Aufgaben reserviert sind. Table 1 gibt einen Überblick über die spezifizierten Kanäle [1].

Tabelle 1: 802.11p Channels.

Channel Type	Center frequency	IEEE 802.11 channel number	Channel spacing	Default data rate
G5-CCH	5900 MHz	180	10 MHz	6 Mbps
G5-SCH2	5890 MHz	178	10 MHz	12 Mbps
G5-SCH1	5880 MHz	176	10 MHz	6 Mbps
G5-SCH3	5870 MHz	174	10 MHz	6 Mbps
G5-SCH4	5860 MHz	172	10 MHz	6 Mbps
G5-SCH5	5850 MHz	182	10 MHz	6 Mbps
G5-SCH6	5910 MHz	184	10 MHz	6 Mbps
G5-SCH7	nach IEEE 802.11, 5470 MHz*to*5725 MHz	94 bis 145	verschiedene	abhängig von der Bandbreite

Es wird deutlich, dass eine Erweiterung des verfügbaren Frequenzbandes von 802.11a (5180 MHz*to*5825 MHz) auf 802.11p (5850 MHz*to*5925 MHz) notwendig ist.

Die Radio Frequency (RF)-Frequenz wird auf dem Mango WARPv3 Board durch den RF-Transceiver MAX2829 (siehe **TODO: Referenz Hardware**) erzeugt. Dieser kann via Serial Peripheral Interface (SPI) durch die Low-CPU des Field Programmable Gate Array (FPGA) konfiguriert werden [4]. Zur Einstellung der Center-Frequenz des Transceivers sind dabei insbesondere die Register *Band Select and PLL*, *Integer-Divider Ratio* und *Fractional-Divider Ratio* wichtig. Über das *Band Select* Register wird das Frequenzband (5 GHz) ausgewählt und durch den Vorteiler (engl. Divider) wird die Grundfrequenz des Oszillators durch einen rationalen Teiler (Ganzzahl und Fraktion) auf den gewünschten Wert abgeleitet. Zur Anpassung der verfügbaren Frequenzen ist daher eine Änderung der möglichen Register-Werte des RF-Transceivers notwendig.

Im WARP Reference Design erfolgt die Konfiguration des Transceivers durch den radio_controller IP Core. Änderungen der Konfiguration erfolgen also am elegantesten im Treiber des Peripherals. Konkret bedeutet dies, dass in der Datei `edk/pcores/radio_controller.c` Änderungen für drei Lookup-Tables `rc_tuningParams_5GHz_freqs`, `rc_tuningParams_5GHz_reg3` und `rc_tuningParams_5GHz_reg4` notwendig sind, nämlich müssen die Register-Werte für die hinzugefügten Channels hinterlegt werden. Die Berechnung der Werte kann händisch nach [4] oder durch das beiligende Python-Skript erfolgen.

Anschließend müssen die zusätzlichen Kanäle zur Verwendung "freigeschalten" werden. Dies erfolgt in der Software der Low-CPU.

2.1.2. Channel Bandbreite

Die verwendete Bandbreite des Kanals hängt direkt von der gewählten Sampling-Rate des verwendeten Analog Digital Converter (ADC)/Digital Analog Converter (DAC)-Wandlers AD9963 ab. Dieser ist ebenfalls über die SPI-Schnittstelle durch die Low-CPU konfigurierbar, die Implementierung erfolgt über den w3_ad_controller Core.

Für 802.11p werden vorrangig Kanäle der Bandbreite 10 MHz verwendet. Diese Bandbreite ist bereits im 802.11 Reference Design implementiert, muss jedoch in der Software der Low-CPU durch einen Aufruf der Funktion `set_phy_samp_rate()` aktiviert werden. Der 10 MHz-Modus wird dabei durch die Konstante `PHY_10M` ausgewählt.

Im 802.11p Reference Design wird ist dies im *wlan_mac_low_11p*-Projekt implementiert.

2.2. Ethernet-Schnittstelle

Für das Projekt wurde beschlossen, dass die Umsetzung von Funktionalität möglichst auf einem normalen Computer erfolgen soll. Die Gründe dafür sind, dass dort bereits eine Vielzahl von spezialisierten Tools (u.a. Wireshark und PCAP) vorhanden sind, deren Implementierung den Rahmen des Projekts bei weitem sprengen würde. Zusätzlich ermöglicht wird es dadurch einfacher ermöglicht, Fehler in der Software zu debuggen und 3rd-party Komponenten (wie beispielsweise MATLAB) anzubinden.

Die Realisierung dieser Design-Ziele erfolgt durch die Instrumentierung des WARP v3 Board über eome Ethernet-Schnittstelle. Darüber können sowohl Daten des WLAN-Kanals empfangen und an einen Computer weitergeleitet, als auch Daten von einem normalen Rechner als 802.11p Frames gesendet werden.

2.2.1. Hardware

Das WARPv3 Board besitzt zwei 1 Gbps-Ethernet-Interfaces. Dabei ist Interface **B** durch das Reference-Design reserviert, um darüber Experimente steuern zu können[3]. Die Schnittstelle zur Datenübertragung wurde deshalb auf Interface **A** realisiert. Diese kann über ein normales RJ45 Ethernet-Kabel mit einem beliebigen Rechner verbunden werden.

2.2.2. RFtap

Die Übertragung der WLAN-Frames über eine Ethernet-Schnittstelle ist nur möglich, wenn diese vorher in ein entsprechendes Transport-Protokoll verpackt werden. Dies ist dem Umstand geschuldet, dass 802.11-Frames keine gültigen Ethernet(-II)-Frames sind und umgekehrt. Würde das WARP-Board die empfangenen 802.11 Frames vollständig identisch auf die Ethernet-Schnittstelle übertragen, werden die Frames von der Netzwerkkarte des angeschlossenen Rechners verworfen und erreichen dessen Betriebssystem bzw. Anwendungen erst gar nicht.

Es wurden zwei verschiedene Protokolle zur Übertragung von 802.11 Frames über die Ethernet-Schnittstelle evaluiert: - radiotap, das spezialisiert ist auf "[...] 802.11 frame injection and reception"[5] - RFtap, ein Protokoll "[...] designed to provide Radio Frequency (RF) metadata about packets"[6]

Für beide Protokolle existiert eine gute Unterstützung in bestehenden Netzwerk-Analyse Tools wie Wireshark. Im Rahmen des Projekts wurde das *RFtap* Protokoll in die High-CPU Software implementiert. Die Gründe dafür sind die einfachere Implementierung gegenüber radiotap und die erweiterte Funktionalität. Da ein RFtap-Frame verschiedenste Payload-Pakete verpacken kann, ist es unter anderem auch möglich, damit radiotap-Frames zu übertragen. RFtap kann folglich als Obermenge von radiotap angesehen werden. Zusätzlich ist es mit RFtap Möglich, die empfangenen Pakete um weitere Informationen (insbesondere physikalische Parameter) zu annotieren. fig. 1 zeigt schematisch den Aufbau eines RFtap-Frames.

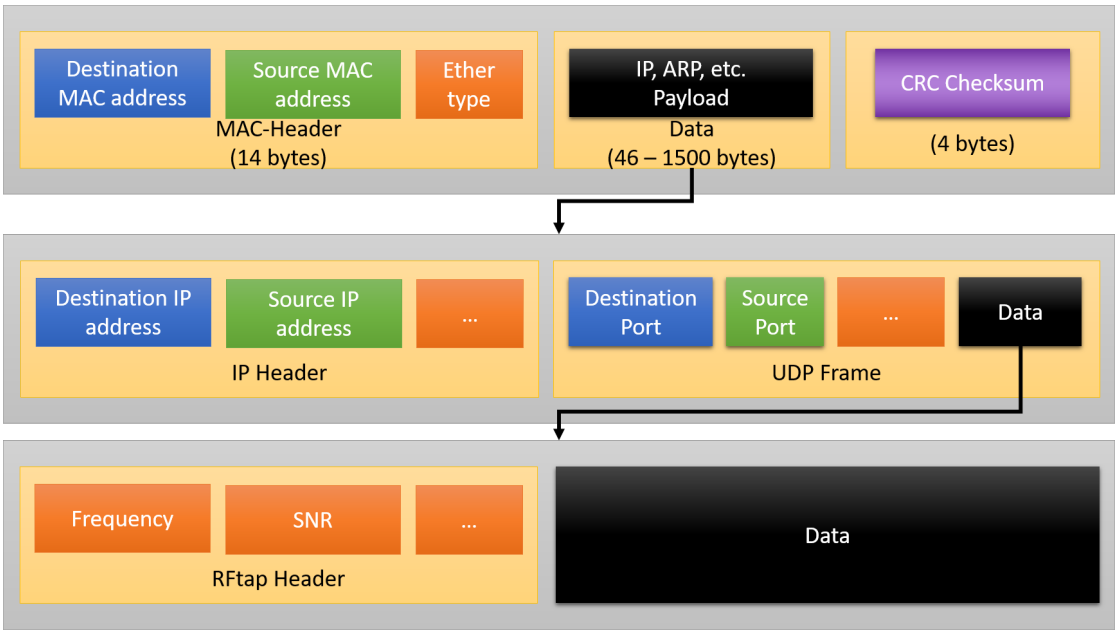

Abbildung 1: RFtap Frame Aufbau.

Für die direkte Verbindung zwischen dem WARPv3 Board und einem Computer sind die verwendeten MAC- und IP-Adressen unkritisch, da auf den Interfaces in diesem Fall keine Filterung stattfindet. In der derzeitigen Implementierung sind die Felder daher leer (Wert 0). Für künftige Projekte wäre es denkbar, diese Informationen sinnvoll zu befüllen. Dies ermöglicht beispielsweise ein Routing von RFtap-Frames über gewöhnliche, kommerzielle Netzwerk-Hardware.

In Wireshark sind RFtap-Dissectors für UDP-Frames auf Destination-Port **52001** implementiert. Erkannt werden die Frames durch die Magic Numbers `0x52 0x46 0x74 0x61` (ascii: RFta) zu Beginn des Frames. Es folgt die Länge (unsigned integer, 2 Byte) des RFtap-Headers (ohne Datenteil!) in 32-bit Words und ein Flags-Bitfield (2 Byte), das die nachfolgenden Header-Flags spezifiziert[7]. Dabei können (aufgrund des Längenfelds) beliebige zusätzliche Felder an das Ende des Headers angefügt werden, die dann jedoch nicht durch einen Dissector abgedeckt werden können. Der RFtap-Frame endet mit dem RF-Payload. Besonders interessant ist die Angabe des Data Link Type (DLT)-Felds, da dadurch der Typ der Nutzdaten spezifiziert wird. Bei korrekter Angabe nutzt Wireshark dann automatisch den richtigen Dissector um die Payload zu analysieren (beispielsweise LINKTYPE_IEEE802_11, IEEE 802.11 wireless LAN Frame [2]).

Das Senden von Frames von einem Rechner erfolgt analog, in umgekehrter Reihenfolge.

Literatur

[1] *Draft ETSI EN 302 663 V1.2.0*. EN 302 663. V1.2.0. ETSI. Nov. 2012.

[2] *link-layer header types | tcpdump/libpcap public repository*. 2017. URL: http://www.tcpdump.org/linktypes.html.

[3] Inc. Mango Communications. *Mango 802.11 Reference Design Experiments Framework Documentation*. 2017. URL: https://warpproject.org/docs/mango-wlan-exp/.

[4] *max2828/max2829 single-/dual-band 802.11a/b/g world-band transceiver*. 2004. URL: https://datasheets.maximintegrated.com/en/ds/MAX2828-MAX2829.pdf.

[5] *radiotap*. URL: http://www.radiotap.org/.

[6] *rftap*. URL: https://rftap.github.io/.

[7] *specifications*. URL: https://rftap.github.io/specifications/.

Abkürzungsverzeichnis