UNIVERSITÀ
DEGLI STUDI
DI BRESCIA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Corso di Laurea Magistrale in
Ingegneria Informatica

Tesi di Laurea

Design and Implementation of Tools for the Analysis of MU-MIMO Wi-Fi Networks

Sviluppo e Implementazione di Strumenti per l'Analisi di reti Wi-Fi MU-MIMO

Relatore: Chiar.mo Prof. Francesco Gringoli

Laureando:
Stefano Fontana
Matricola n. 727199

Anno Accademico 2023/2024

Sommario

Questa tesi si propone di sviluppare strumenti software e hardware per l'analisi in tempo reale delle comunicazioni Wi-Fi in cui vengono trasmessi frame Multi-User - Multiple Input and Multiple Output (MU-MIMO).

Questo lavoro cerca di rendere possibile per un ricevitore di terze parti catturare i suddetti frame sfruttando le funzionalità dei chipset Wi-Fi presenti nelle comuni schede di rete per computer, consentendo all'utente di caratterizzare il comportamento della rete.

La tecnologia 802.11ax, comunemente nota anche come 802.11 High Efficiency (HE), consente l'invio e la ricezione di frame multiutente grazie all'uso della modulazione Orthogonal Frequency Division Multiple Access (OFDMA). Questa modulazione, che è anche alla base delle moderne reti cellulari, permette di suddividere la larghezza di banda di comunicazione in intervalli assegnati a varie stazioni riceventi o trasmittenti. Queste trasmissioni vengono poi decodificate dalle schede di rete, che possono essere configurate, in base alle informazioni derivate dal traffico non multiutente e alle informazioni contenute negli header fisici, per decodificare un frame indirizzato a un'altra stazione.

Successivamente, viene valutata la possibilità di modificare il firmware su tali schede per abilitare la decodifica dei frame presenti nel pacchetto, consentendo così all'utente di avere una visione più completa della rete.

Summary

This thesis aims to develop software and hardware tools for the real-time analysis of Wi-Fi communications where Multi-User - Multiple Input and Multiple Output (MU-MIMO) frames are transmitted.

This work attempts to make it possible for a third-party receiver to capture the aforementioned frames by leveraging the functionalities of Wi-Fi chipsets present on common computer network cards, allowing the user to charaterize and observe the network behavior.

The 802.11ax technology, also commonly referred to as 802.11 High Efficiency (HE), allows the sending and receiving of multi-user frames thanks to the use of Orthogonal Frequency Division Multiple Access (OFDMA) modulation. This modulation, which is also the basis of modern cellular networks, allows the communication bandwidth to be divided into intervals assigned to various receiving or transmitting stations. These transmissions are then decoded by network cards, which can be configured, based on information derived from non-multi-user traffic and information contained in physical headers, to decode a frame addressed to another station.

Subsequently, the possibility of modifying the firmware on such cards is evaluated in order to enable the decoding of the frames present in the packet, thereby allowing the user to have a more comprehensive overview of the data traffic.

Contents

1 — Introduction

Wireless network analysis is a critical area of research that underpins the development of advanced technologies such as enhanced scheduling algorithms and efficient resource allocation strategies. As wireless communication standards evolve, the ability to understand network behavior is becoming ever more important. Traditional consumer-grade sniffing tools often fall short in capturing the complexity of modern networks, particularly in high-load scenarios or when dealing with multi-user transmissions.

This work presents an alternative approach to wireless network sniffing, leveraging the capabilities of Broadcom chipsets to overcome the limitations of existing solutions. By utilizing the chipset's four-antenna configuration, our approach enables the capture of multiple spatial stream communications in a more common Access Point (AP) configuration. This enhanced sniffing capability provides a more detailed and accurate representation of network activities, which is crucial for deriving statistically meaningful parameters that can drive further research and development.

Furthermore, we introduce an innovative method for capturing Orthogonal Frequency Division Multiple Access (OFDMA) multi-user frames, an essential technique for the analysis modern wireless networks adhering to the 802.11ax standard, which employ such complex transmission schemes. With this work we aim to provide a sniffing toolset capable of closely representing the network behavior, offering insights into every aspect of the communication.

The modifications and enhancements presented in this work are based on the Nexmon framework, an open-source reverse engineering and patch-

ing tool for Broadcom-based chipsets. This choice not only facilitates the adaptation of our approach to a wide range of devices but also ensures that the research community can freely access and build upon our work. By making these tools openly available, we aim to empower researchers to explore new dimensions of wireless network analysis and contribute to the advancement of the field.

In the following chapter, we will discuss the basic concepts needed to develop such tools, alongside a brief introduction to the 802.11ax standard. Then, we will illustrate the developing process of our sniffing tool, evaluating its performance in various scenarios, and highlighting its potential applications.

2 — Wi-Fi Fundamentals

A Wi-Fi wireless network operates as a layer-two computer communication network, facilitating device communication without requiring direct physical connections. Instead, it utilizes a radio channel to transmit packets between network peers. The typical configuration of a Wi-Fi network, described in Fig. 2.1, comprises multiple Stations (STAs) that connect to a deployed network of APs [1]. These connections may be established over an encrypted or unencrypted channel, based on the security requirements of the network [2].

The presence of multiple APs allows for a bigger area covering, limiting the single AP output power, which results in lower costs for covering places. Multiple APs deployments also need a method of managing roaming of the STAs, either managed by the core network or managed by the wireless devices utilizing signal strength as a metric [3].

Inside the Wi-Fi network, APs serve a critical role in managing the wireless access, orchestrating the communication between connected stations, and potentially forwarding traffic to and from the core network [4], generally housing required network services.

The central management role of the AP allows for an easy management of the connected clients. A group of clients connected to an AP is called Basic Service Set (BSS) [5], whereas the alternative solution of a distributed interconnected network, without a predefined AP, (Wi-Fi *ad hoc* network [4]) implies difficulties in network discovery and routing, as a constantly changing mesh network may introduce latency and unnecessary routing steps [6].

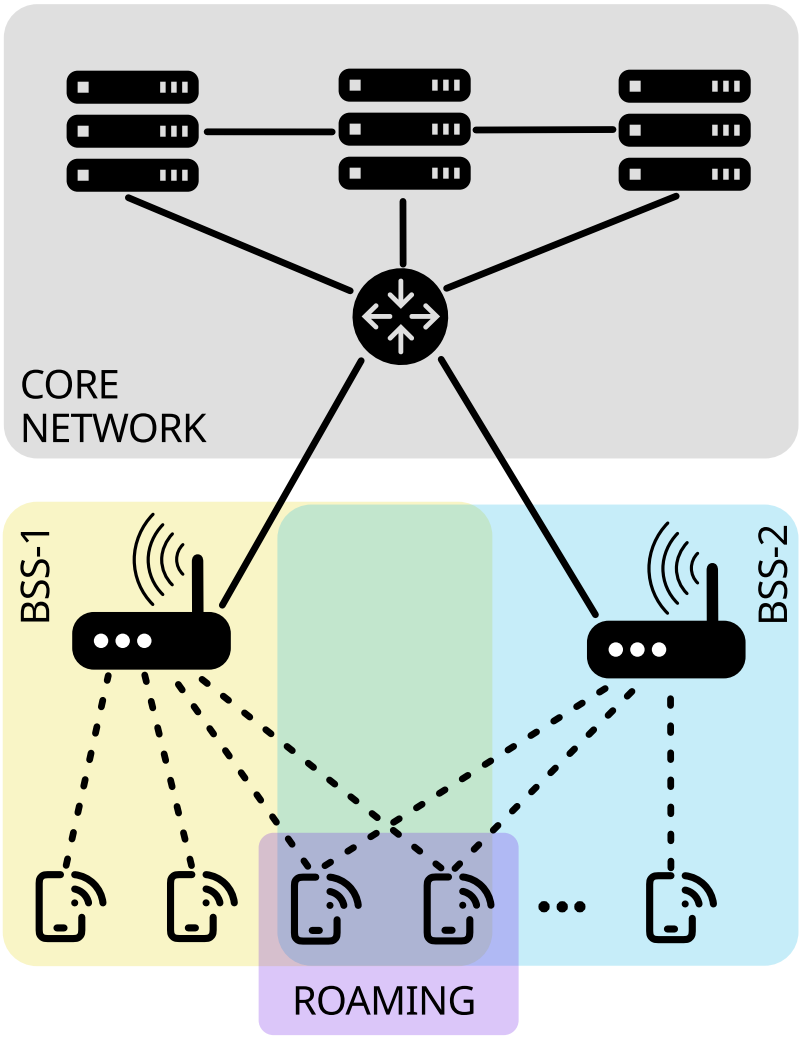


Figure 2.1: Schematic representation of a Wi-Fi network infrastructure with multiple APs and roaming devices in the process of changing BSS.

Radio communication in a Wi-Fi network is achieved through modulated radio transmissions over a designated channel from an available channel pool [4], [7]. A channel is defined by its central frequency f_c, also known as the "carrier frequency", and a bandwidth symmetrically distributed around this central frequency. Nowadays, Wi-Fi utilizes three possible frequency bands: the classic 2.4 GHz, in use since the early releases of the 802.11 standard, the 5 GHz band, and the 6 GHz band later introduced in the 802.11ax release.

The AP plays a crucial role in detecting and inferring the channel state between each connected peer by transmitting and receiving data. This capability allows the AP to select the appropriate transmission power and modulation scheme to maximize the data rate over the channel.

2.1 – Modulation

Every radio communication makes use of modulation in order to transfer meaningful data to the respective peer.

Modulation techniques employ different methods to encode information in a radio transmission over the air. Generally, this modulation is applied over one or more sinusoidal signals that are later transmitted thanks to an antenna and relative circuitry.

A general overview of modulation can be represented by a generic a cosinousoidal signal, defined in its time domain by Equation (2.1).

As per Equation (2.1), the signal possesses defined and misurable features in the time domain.

$$s(t) = A(t)cos(\omega(t)t + \phi(t)) \tag{2.1}$$

We can observe that, for each time instant t the signal has an amplitude $A(t)$, a frequency, related to the $\omega(t)$ angular frequency, and a phase $\phi(t)$.

Most of Wi-Fi modulation schemes act on each subcarrier by modifying phase and amplitude, without modulating its frequency in order not to interfere between subcarriers.

With additional constraints, as defined in Equation (2.2), we can quantize the ranges of the two parameters $A(t)$ and $\phi(t)$ and assign a meaning to them.

In this case, we can define a quantization interval and split the two variables' domains in "buckets" of values representing a symbol. The cartesian product of the two quantized intervals generates the symbol space.

$$\begin{aligned} &A(t) \in [A_{min}, A_{max}) \\ &\phi(t) \in [\phi_{min}\phi_{max}), \quad -\pi \leq \phi_{min} \leq \phi_{max} \leq \pi \end{aligned} \tag{2.2}$$

For example, by defining a quantization interval of dimension N, we can split the amplitude interval in equal buckets such that

$$\begin{aligned} \Delta_A &= \frac{A_{max} - A_{min}}{N} \\ symb = x \iff A(t) &\in \Big[A_{min} + x\Delta_A, A_{min} + (x+1)\Delta_A\Big) \end{aligned} \tag{2.3}$$

The same happens for the phase modulation,

$$\begin{aligned} \Delta_\phi &= \frac{\phi_{max} - \phi_{min}}{N} \\ symb = x \iff \phi(t) &\in \Big[\phi_{min} + x\Delta_\phi, \phi_{min} + (x+1)\Delta_\phi\Big) \end{aligned} \tag{2.4}$$

The two resulting sets of possible symbols can then be used together in

order to provide a N^2 cardinality symbol set.

This is possible because the two variables are orthogonal between each other. Amplitude and phase in the signal are both independent variables, thus are not dependent on each other in the definition of the signal $s(t)$ in Equation (2.1).

The described modulation, namely Quadrature Amplitude Modulation (QAM), is used because of its simple implementation [8]. It is enough to sum two quadrature signals (shifted in phase by $\frac{\pi}{2}$) modulated in amplitude (using Amplitude Shift Keying (ASK) techniques) as described by Equation (2.5).

$$\begin{aligned} s(t) &= I(t)sin(\omega t) + Q(t)cos(\omega t) \\ &= I(t)sin(\omega t) + Q(t)sin\Big(\omega t + \frac{\pi}{2}\Big) \\ &= \sqrt{I(t)^2 + Q(t)^2}sin\left(\omega t + asin\left(\frac{Q(t)}{\sqrt{I(t)^2 + Q(t)^2}}\right)\right) \\ &= \bar{A}sin(\omega t + \bar{\phi}) \end{aligned} \tag{2.5}$$

This allows for a demodulator to exploit the orthogonality between $I(t)$ and $Q(t)$ to decompose the signal simply by multiplying it by a local referenced signal and its 90 degrees phase shift.

The typical graphical representation for QAM-modulated signals is known as "constellation"; an example of its structure is provided in Fig. 2.2. In the general case, the cross product of the symbol spaces in the two axes will give a set of points representing a symbol. In general, a symbol is a defined set of values associated to the modulating quantities that get mapped into a known set of bits.

As per the case of 16-QAM shown in Fig. 2.2, we can associate each

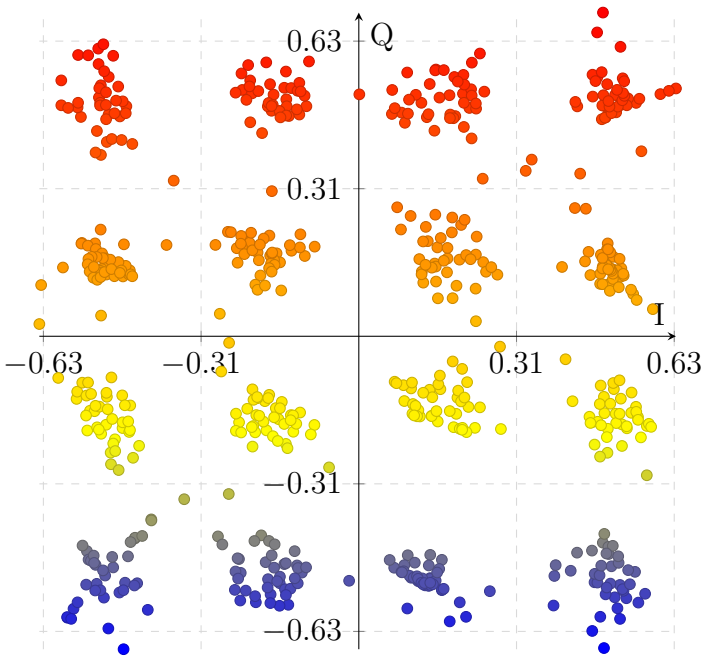


Figure 2.2: 16-QAM Constellation after symbol decoding. This constellation represents the I and Q component of the decoded signal at multiple points in time. The points are not perfectly overlapped due to the presence of noise in the received signal.

symbol to a four bit value that gets decoded after the demodulation.

For easiness of use, the constellation diagram in Fig. 2.2 does not place the points in the Amplitude-Phase cartesian plane. Instead, it defines the axes as the values of the quadrature signal modulator $I(t)$ and $Q(t)$. Because of Equation (2.5), the plot can be read as amplitude and phase modulation on a polar coordinate system.

The modulation and demodulation scheme, reproduced in Fig. 2.3, are, as mentioned before, very simple.

To perform IQ Modulation, a Local Oscillator (LO) is tuned to the expected carrier frequency and its output is fed into two mixers, phased by 90 degrees between the two inputs, used for ASK modulation of the signal. These two mixers will use the I and Q signals to amplitude modulate the

LO signal in order to inject the resulting outputs into a combiner. The output of this is the modulated signal in both amplitude and phase [9].

The receiving side, instead, must recover the clock in some way and synchronize the internal reference (the internal LO) for the demodulation. This makes the reception of a QAM modulation a little harder.

The signal waveforms of the modulator and demodulator are displayed in Fig. 2.4. The figure represents the signals in time as they are modulated and demodulated. The signal starts as a stream of symbols (Fig. 2.4a) which is split in the I and Q components by an adequate metric. The signals in Fig. 2.4b are then used to modulate the two, 90 degrees phased, LO derived signals. The two ASK modulated signals are then added together in Fig. 2.4c and later sent by the radio device.

The demodulator then recovers the I and Q components (Fig. 2.4d) which are perturbed by the high frequency component of the transmitted signal. This perturbation is removed by a low pass filter, as displayed in Fig. 2.4e, and the signals are then averaged inside the symbol window and decoded in the final result, with a reverse IQ-sampling process.

It can be stated that the generic QAM modulation may be a generalization of other types of modulation. For example, by placing

$$A_{min} = A_{max}$$

in Equation (2.2), *de facto* reducing the amplitude symbol space to an empty set, we define a phase modulation scheme, allowing the IQ hardware to perform Binary Phase Shift Keying (BPSK), Quadrature Binary Phase Shift Keying (QBPSK), ASK etc.

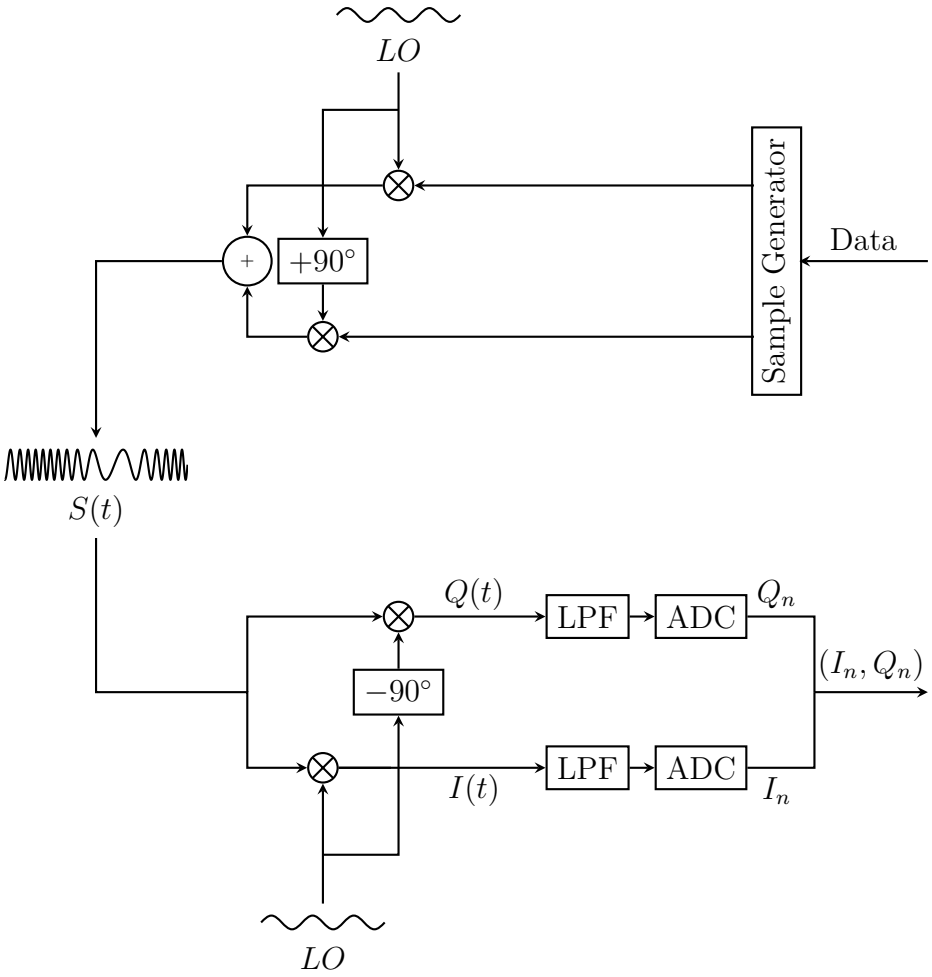
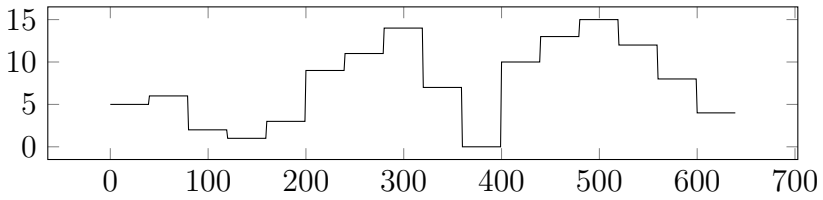
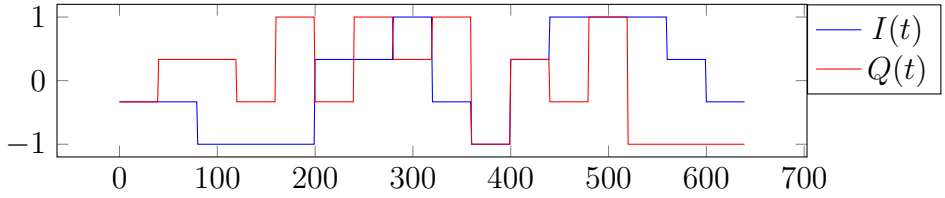


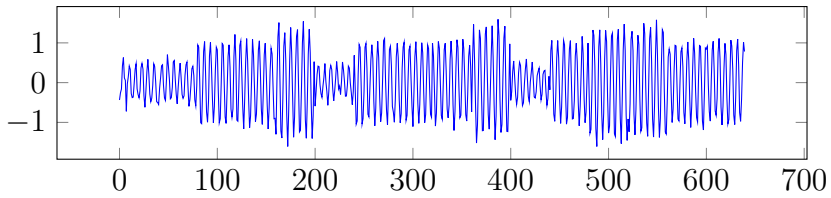
Figure 2.3: Conceptual diagram of the IQ modulation and demodulation process. The signal is first modulated in order to generate the signal $S(t)$ which is sent into the air and later receiver and demodulated by the decoder.



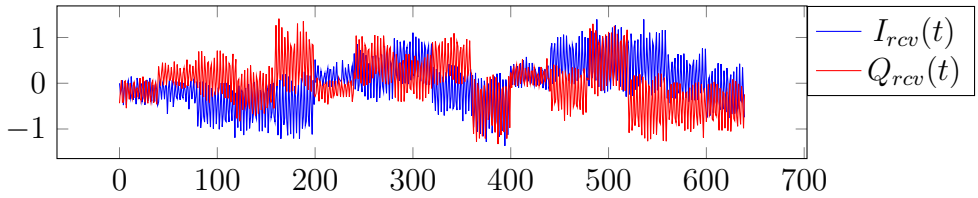
(a). Signal values in time. Each signal value is represented with 4 bits as the 16-QAM modulation required symbol size.



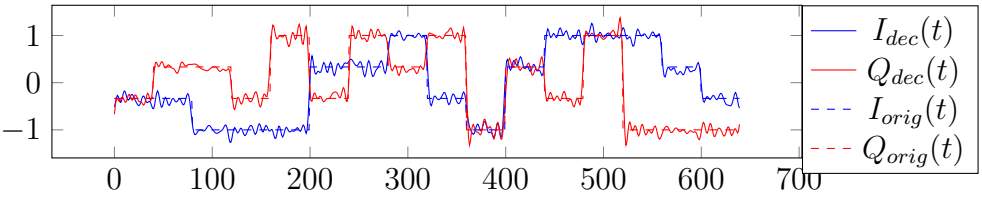
(b). I-Q decomposition of the signal in Fig. 2.4a. The I signal is determined by the two lower bits of the symbol and the Q signal from the two high bits.



(c). Resulting 16-QAM modulated signal.



(d). Receiver side recovered I and Q signals.



(e). Receiver side recovered I and Q after low pass filtering. The two recovered siganls present a small deviation from the original signals.

Figure 2.4: IQ Modulator and demodulator example signals.

Detailing the previous assertion, by forcing

$$\sqrt{I(t)^2 + Q(t)^2} = 1$$

the resulting IQ-modulated signal gets a constant amplitude of one, allowing for the phase variation to be simplified to $asin\Big(Q(t)\Big)$.

2.2 – Physical Header and Equalization

Wi-Fi supports OFDM since it was introduced in 1999 with the 11a amendment. While initially Orthogonal Frequency Division Multiplexing (OFDM) was possible only in the 5GHz band, support was extended to the 2.4 GHz band later in 2003 when 11g was ratified. In this transmission techniques a channel is divided into multiple orthogonal subcarriers representing each a symbol stream. These streams are located at a specific frequency intervals.

Since subcarriers were initially spaced by 312.5 kHz — and in later revisions 76.25 kHz — and based on the mathematical concepts behind OFDM, later discussed in Sect. 2.4.2, only amplitude and phase can be modulated to encode information without interfering with the neighbor data stream.

Because the transmission channel is not perfect, it is indispensable to perform channel equalization on the received data [10], in order to correctly decode the frame when subject to a non-uniform response. Given the usually high bandwidth used by OFDM modulation, there is a high probability that the transmission channel may alter some parts of the spectrum more than others, because of the multipath phenomenon [11]. Because of this necessity, the introduction of the Physical Layer Convergence Protocol (PLCP) follows naturally.

The PLCP, described by Fig. 2.5a for the OFDM based channel modulation, allows for three main scopes: Start of Frame (SOF) detection, Channel State Information (CSI) inference, and demodulation setup.

At first, because Wi-Fi transmits packet data and the Radio Frequency (RF) signal is not continuous, there must be an identifiable sequence to allow the chipset's Base Band (BB) to detect the start of the frame.

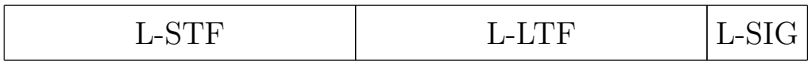
As this step is performed in hardware, the resulting field (Short Training Field (STF)) is used to recover clocks and to synchronize the receiver to the incoming packet [4], [12].

Then, after the STF has been identified and fully received, the second PLCP field is decoded. This field contains a known data sequence to allow the receiving device to extract the CSI affecting the frame [13]. Because, as hypothesis, the channel response does not change, or its change is limited in the frame's life [14] in a non-erratic way, the extracted CSI is then used to perform the equalization needed to decode the rest of the packet.

The CSI extracted from the Long Training Field (LTF) field contains the phase and amplitude response of the channel found by the receiver. Generally, because of multiple spatial streams, these values are represented as a matrix where each element is defined as the channel response between two pairs of antennas (sender and receiver). Thus, the equalization procedure is defined as the pseudo inversion of this matrix to be later applied to the signal, before demodulation.

The third and final utility of the legacy PLCP header is to prepare the receiver to demodulate the rest of the packet.

Because prior to the PLCP no information is transferred to the receiver, the standard must define a fixed modulation for the header transmission to ensure that each device can detect the frame and not interfere. This



(a). 802.11 OFDM PLCP fields.

(b). SIG field structure definition.

Figure 2.5: Legacy PLCP Header structure [4] for OFDM modulated networks.

modulation must satisfy the following conditions:

- It must be slow. Slow modulations are easier to decode and can travel further distances because the power per bit ratio is higher, because the transmission is performed with the same power but takes longer. This enables a distant STAs to detect the beginning of a frame and ensures that collision avoidance methods are effective.

- It must be available to every device supporting the entire standard or the coexistence with it. Generally newer standards allows for older specification devices to be in the same network; this old devices must be able to detect the non-free channel even if they are not able to decode the frame. This usually implies that BPSK is used as defined in legacy standards [4].

The last field of the PLCP is the SIG field, later known as L-SIG, legacy signal field. This field contains information regarding the upcoming data's modulation. Its definition in Fig. 2.5b denotes that the field contains information relative to the transmission, like the total length of the upcoming data and its transmission rate, thus its modulation. These fields, alongside a parity bit to verify the preamble's integrity, allow the receiver to configure its hardware to decode incoming data after the preamble [4].

It's necessary to mention that network management, like association

and authentication, is not performed in the physical layer but is later implemented by the MAC protocol.

Following the PLCP header is the full extent of the data frame sent by the transmitter. Even if the PLCP defines the modulation of the subsequent part of the frame, the standard does not impose the utilization of that modulation for the whole packet length. Taking as an example 802.11ax and 802.11ac, new standards insert an additional implementation-dependent PLCP right after the legacy header previously described. In this subsequent PLCP header, additional modulation information are inserted without breaking the legacy standard, thus granting coexistence between new devices and old ones. This allows for the coexistence of multiple standards in the same network, because older devices are able to detect the presence of a frame in air, given the fact the legacy PLCP described in Fig. 2.5b is present, that later will be dropped because of unknown modulation or format, but the device will be able to interrupt its medium access routine impeding interferences.

2.3 – Channel access

The standard Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA) medium access protocol defines the channel access with the Contention Window (CW) mechanism [15]. It is based upon positive acknowledgement, where a receiver sends back to the transmitter an Acknowledgement (ACK). If the ACK times out, the frame is retransmitted.

Whenever the channel is sensed free by a STA, a Back-Off counter (BO) is initialized with a uniformly random value between 0, included, and the value of the CW excluded. Because channel access is discretized in 9 µs

slots — this means that a transmission can only start at the beginning of a slot —, the STA will count down a BO extracted number of slots before starting a transmission. If another STA has extracted a smaller BO value, all the running BO counters get paused and will resume after the transmission has ended.

If a collision is detected because the ACK is not sent back, then a new BO gets extracted in an interval where the CW gets doubled until its limit of 1024 slots.

For example, in Fig. 2.6, after the transmission of STA2 and the relative ACK and after a Distributed Interframe Space (DIFS), both stations randomly extract a BO value between 0 and 15. Since the BO counter of STA1 expires first, it will start transmit occupying the channel. After the usual ACK and DIFS, since both stations need to transmit a frame, STA1 extracts another BO. This time, both the BO have the same value (since STA2's BO got paused when 4 slots were still to be waited). The transmission will result in a collision, making both the STA timeout in waiting the ACK. The contention window gets doubled, and a new access is performed.

This method of channel access will introduce long latencies with the growth of the number of STAs in the network [16], hence limiting the total network throughput, making the network inefficient because of the time spent in waiting during the CW.

This procedure, as also demonstrated by Fig. 2.6, does not completely prevent collisions. There is still the possibility of extracting the same BO value, because, according to the birthday paradox, in a big network the possibility of collision is pretty high, resulting in an on-air collision and successive retransmission.

In order to limit frame collisions, 802.11 defines a channel booking pro-

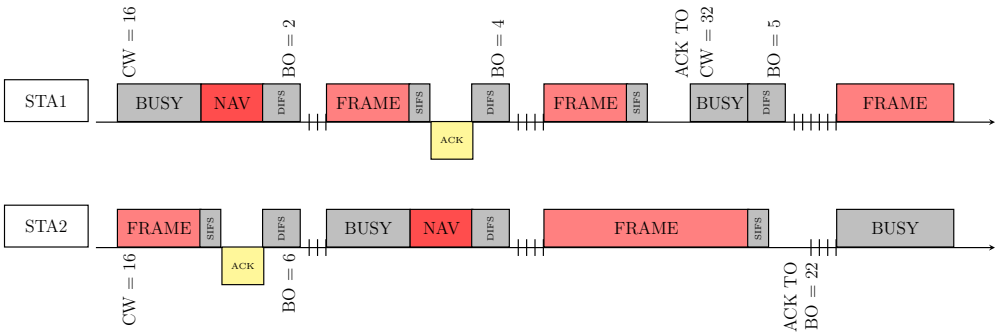


Figure 2.6: Contention window mediation for channel access between two STA.

tocol, mandating the utilization of an active channel protection method. This method, become mandatory since 802.11ac, consists in a two-way request and response with special frames: Request To Send (RTS) and Clear to Send (CTS). Before 802.11ac, the channel access was made with the CW method until a threshold failure count, where the RTS-CTS method was then used. The transmitting device is mandated to send a RTS frame, containing details on the intended destination and the number of slots it will take to transmit the frame, in order to announce the intention of transmission, thus "booking" the channel access. RTS frame structure is represented in **??** [4].

Once the RTS has been received by the destination device, the receiver will respond with a CTS frame after a SIFS, granting priority access to the channel and making collisions nearly impossible. This enables the other STAs to detect an incoming transmission, aborting the possible contention window, improving the raw network performance, and addressing the hidden station problem [17].

It is necessary to specify that the channel will be seen as occupied by other since right after the RTS transmission, as a mechanism of impeding collisions.

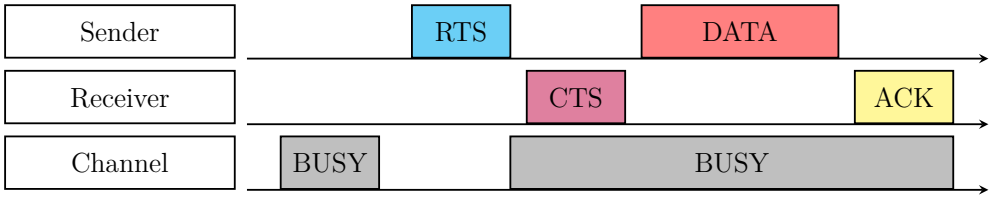


Figure 2.7: RTS-CTS channel access dialogue.

The advantage of RTS and CTS in reducing collisions consist in their raw frame size [18]. Although there can still be collisions in RTS and CTS frames, their small size allows for faster retransmissions and reduced losses. The RTS frame, shown in, consists of a simple Medium Access Control (MAC) Layer control frame totaling at 20 octets of length. The CTS frame, is smaller because it does not contain the whole MAC layer data but will only report the MAC address of the RTS originating device, totaling at 14 octets in size.

The RTS and CTS frames are always sent as a 802.11g/a frames with slowest Modulation Coding Scheme (MCS), maintaining compatibility with older devices in order to not limit coexistence of multiple standards.

Another utilization for CTS is in channel protection. In this case, a transmitting device, like an AP, sends out a CTS frame without previously having received the corresponding RTS in order to clear the channel.

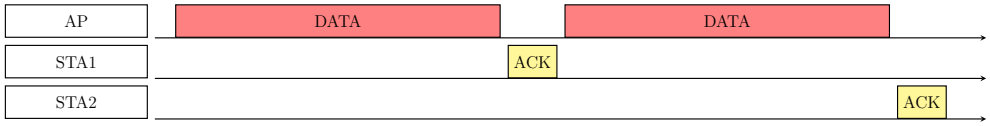
2.4 – Multi-User Multiple Input and Multiple Output

Since 802.11ac — also known as Wi-Fi 5 —, APs have gained the ability of sending packets to multiple stations in a single frame. This technology, known as Multi User Multiple Input Multiple Output (MU-MIMO), is the direct descendant of Multiple Input Multiple Output (MIMO) technology,

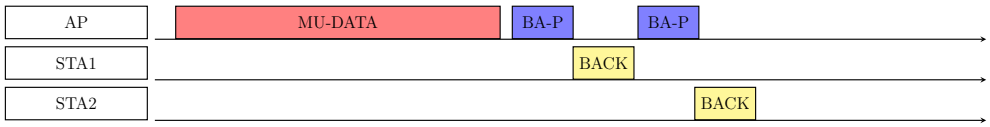
capable of multiple data streams on the same channel thanks to spatial diversity directed to the same STA. This technology is a generalization of MIMO transfers, which are derived by beam-forming techniques, where the signal is "steered" thanks to a computational operation in the digital domain before transmission.

The AP and each STA in the MU-MIMO group are able to recreate the channel response matrix, previously described in Sect. 2.2, and to precode data streams to allow each receiving end to decode its directed stream without interference from the others [19], effectively performing beamforming operations. This is done, in both MIMO and MU-MIMO cases, by exchanging the CSI matrix between the transmitter (generally the AP) and the receiver. By gaining information relative to the channel response, with adequate protocols, the transmitter is able to weigh each data stream with a precoding matrix, whose formulation and construction won't be detailed in this work, to make the receiver capable of recovering its stream from the sum of the others. This method implies the utilization of CSI data to guarantee that, by performing the reverse beamforming operations, the receiver will be able to retrieve the data destined to it.

MU-MIMO, as previously mentioned, is a generalization of MIMO technology, where instead of directing multiple streams to one station increasing the available data rate, multiple streams are directed to multiple STAs. For example, in a system of 3 STA and one AP, the AP may start a MU-MIMO transmission with four streams, considering hypothetically, the AP possesses four antennas, allocating two streams for STA 1 and one stream for the remaining two STAs. Reception of the data by the STA is simply a MIMO reception (if needed). The STAs are not required to know the presence of other MU-MIMO transmissions.



(a). No MU-MIMO channel access and data transmission process description.



(b). MU-MIMO channel access

Figure 2.8: MU-MIMO channel access and data transmission process description.

The advantage of MU-MIMO is that the AP does not have to access the channel multiple times, with the methods described in Sect. 2.3, for each outgoing packet, but it can send all the data in one frame, thus reducing latency and increasing overall throughput [20].

Fig. 2.8a and Fig. 2.8b delineate the two different access methods and highlight the different timings of the transmissions.

CSI data exchange for MU-MIMO in 802.11ac is performed explicitly [21]. Channel measures are performed periodically, by announcing the emission of a Null Data Packet (NDP) with a Null Data Packet Announcment (NDPA), whose structure is shown in Fig. 2.9, containing information regarding the stations that will be involved in the polling procedure. In particular, the frame will report the list of Association Indexes (AIDs) requested and the type of the feedback [22].

The transmission of a NDP, represented by a standard management frame with a known training sequence instead of the usual data, is then followed by a polling procedure where the AP, for each station interested by the NDPA, will request the resulting CSI in compressed format with a poll packet [21] (apart from the first STA which responds after a Short

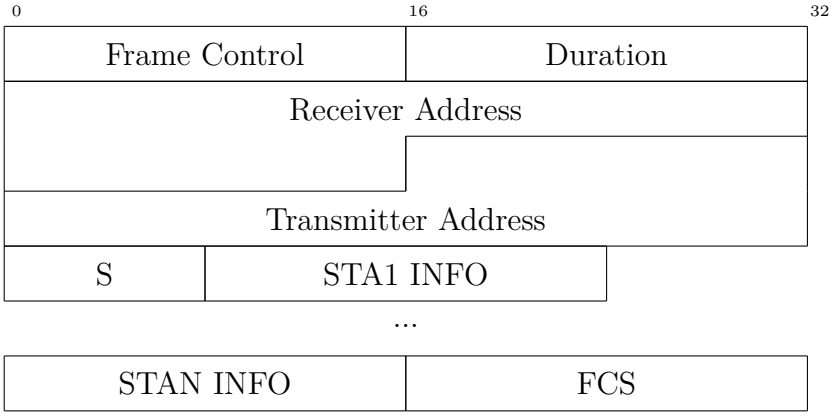


Figure 2.9: NDPA Frame structure [22].

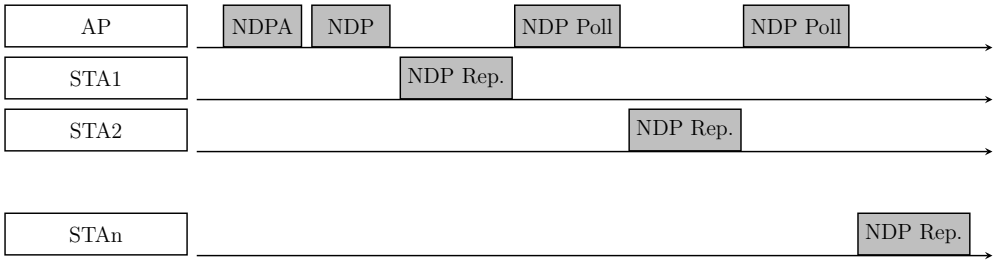


Figure 2.10: 802.11ac sounding procedure. This procedure allows for the reporting of CSI information to the AP.

Interframe Space (SIFS)).

The probing sequence, also known as "sounding procedure", is shown in Fig. 2.10. Each station will answer, whenever the AP polls them, with a compressed beamforming report containing the channel information in a compressed format.

2.4.1 – Limitations

MU-MIMO in 802.11ac comes with several limitations that affect its overall efficiency and applicability in wireless networks. There is no support for MU-MIMO in the uplink direction [23], which requires clients to take turns when transmitting to the AP. This lack of uplink MU-MIMO can lead to inefficiencies, particularly in environments with many clients competing for

bandwidth.

The number of spatial streams allowed in a MU-MIMO transmission is another limitation. The maximum number of spatial streams is given by the amount of channels between the AP and the STA [21]. For example, a four-antennas AP can handle up to four concurrent streams with a four antenna STA. In the MU-MIMO case the same concept applies: with four antennas the AP can serve four single antenna users or two dual antenna users with two streams each.

Another challenge with MU-MIMO in 802.11ac is client compatibility. Not all devices support MU-MIMO, and even those that do must be able to receive the specific number of spatial streams that the AP is transmitting. This limits the effectiveness of MU-MIMO in practical scenarios, as the potential benefits are only realized when both the AP and clients support the necessary number of spatial streams [24].

The performance of MU-MIMO is also highly dependent on the wireless environment. Poor channel conditions, such as interference or a low signal-to-noise ratio (SNR), can significantly degrade MU-MIMO's effectiveness, reducing the throughput gains expected from simultaneous transmissions.

Additionally, the spatial distribution of clients is needed. If clients are too close to each other, or if the AP's beamforming is not accurate, the required channel diversity is compromised, leading to suboptimal performance or even the inability to use the technology.

2.4.2 – Orthogonal Frequency Division Multiple Access

OFDMA is an extension for multi-user communication. This method of multi-user communication is introduced by 802.11ax to compensate for MU-MIMO limitations described in Sect. 2.4.1. With the introduction of

this technology, the number of possible users in a frame increases drastically. Considering the addition of 160MHz channels, the resulting multi-user allocations can reach 128 users, granting a higher overall throughput of the network.

Initially, in order to define OFDMA, some mathematical concepts regarding frequency orthogonality derived from OFDM are needed. OFDM is a frequency multiplexing technique utilized since 802.11a. The advantages of OFDM consist easier equalization and better interference rejection. These advantages overcome the increased hardware manufacturing complications. Given the orthogonality of the subcarriers, they are not interfering with each other by mathematical construction, allowing for a reduced manufacturing cost of the transmitter and receiver because no subcarrier filtering has to be added to the device. Second, by allowing for multiple data carriers, OFDM allows for longer symbol time without data rate plummeting [25], maintaining the channel throughput constant but making the communication more robust by sending more energy per symbol.

The concept of OFDM consists in subdividing the digital signal in multiple parallel streams to be sent on multiple subcarriers. We define a subcarrier as a modulated signal of frequency f_i such that $f_i \in [f_c - \frac{BW}{2}, f_c + \frac{BW}{2}]$, where f_c is the center frequency of the selected channel and BW is the bandwidth.

The problem occurring when transmitting concurrent signal in the spectrum is that they may interfere with each other if not conditioned or placed correctly. This is why orthogonality is important in OFDM and consequently in OFDMA.

It is known that a Fourier transform over an infinite time and pure tone signal will result in an impulse function shifted by the frequency of the

tone. This is not true if the time window is not infinite: this case result in some spread of the peak around the real frequency given by the signal uncertainty.

Given the definition of the Fourier transform for a signal of period T in Equation (2.6),

$$\mathcal{F}\{f(t)\}(n) = \int_{-\frac{T}{2}}^{\frac{T}{2}} f(t)e^{-2\pi\frac{n}{T}ti}dt \tag{2.6}$$

it is possible to theoretically apply it to a single OFDM symbol of duration δ_s. The result is a closed form in the variable k of the value of the Fourier transform of the signal.

Let's assume, without loss of generality, that the symbol resulting from the modulation of the sub-carrier f_i is representable as a sinusoidal function with no phase deviation and given amplitude. The solution of the Fourier transform of the signal, over the symbol duration δ_s, is defined by Equation (2.7) assuming the symmetry of the window that has been reformulated as $[-\frac{\delta}{2}, \frac{\delta}{2}]$.

$$\begin{aligned}\mathcal{F}\{s(t)\} &= \int_{-\frac{\delta}{2}}^{\frac{\delta}{2}} Asin(\omega t)e^{-2\pi\frac{n}{T}ti}dt = \\ &= 2A\frac{\omega T cos(\frac{1}{2}\delta\omega)sin(\frac{\pi\delta n}{T}) - 2\pi sin(\frac{\delta\omega}{2})cos(\frac{\pi\delta n}{T})}{\omega^2T^2 - 4n^2\pi^2}i\end{aligned} \tag{2.7}$$

As we are interested in orthogonality between frequencies, we must solve Equation (2.8).

$$\mathcal{F}\{s(t)\}(f_i) + \mathcal{F}\{s(t)\}(f_i + \Delta f) = \mathcal{F}\{s(t)\}(f_i + \Delta f) \tag{2.8}$$

A valid solution to the system is when Equation (2.9) is satisfied.

$$|\mathcal{F}\{s(t)\}(f_i)| = 0 \tag{2.9}$$

Equation (2.9) also shows why the assumption of the phase shift of the signal to be zero holds. Since we are interested in the modulus of the Fourier transform in the frequency f_i, it is known that Equation (2.10) holds, thus the assumption does not limit the generality of the statements.

$$\Big| \int_{-\frac{T}{2}}^{\frac{T}{2}} Asin(\omega t) e^{-2\pi \frac{n}{T} ti} dt \Big| = \Big| \int_{-\frac{T}{2}}^{\frac{T}{2}} Asin(\omega t + \phi) e^{-2\pi \frac{n}{T} ti} dt \Big| \tag{2.10}$$

It is possible, although this work will not construct a symbolical solution, to determine the solution of Equation (2.9).

The numerical solution found approaches a value $\Delta f = \frac{k}{\delta_s}$, as shown in the plot displayed in Fig. 2.11.

The left and write zeros, barring any rounding errors, are located at $f_i \pm \frac{k}{\delta_s}$ frequency. Thus, by placing sub-carriers at the newly found locations, OFDM will remove any sub-carrier crosstalk without employing special filters or wasting guard bands between the subcarriers.

As previously mentioned, OFDM allows for a more stable channel — if keeping the data rate constant — because each subcarrier splits the amount of data to be sent per symbol in it, allowing for longer symbol time and closer subcarrier distances. Alternatively, by maintaining the same modulation on each subcarrier, it is possible to theoretically multiply the data rate across the channel by increasing its bandwidth. With twice the amount of subcarriers, the total data rate will be doubled.

We can take this concept to the multi-user domain simply by changing the meaning of each sub-carrier stream. Mainly, in OFDM, one single

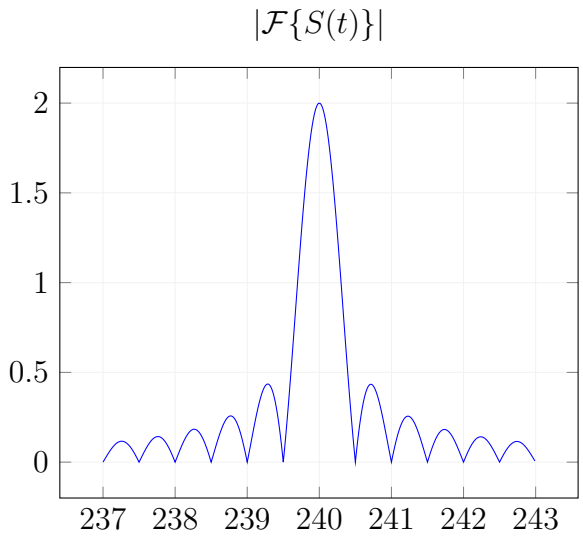


Figure 2.11: Fourier transform of a signal with frequency $f_i = 240Hz$, $\delta_s = 1s$ with relative OFDM zeros.

data stream is split into n_{sub} data sub-streams. The same concept can be extended to multiple input data streams: each data stream can be split on multiple sub-carriers.

The example in Fig. 2.12 assumes two data streams $y_1(t)$ and $y_2(t)$ and a channel with $2n$ subcarriers: the first stream can be mapped to the first n subcarriers and the second to the remaining part.

This concept may be extended to an arbitrary number of subcarriers and arbitrary number of data streams. The main limitation is radio performance during decoding.

The implementation of both OFDMA and OFDM is done thanks to the FFT being doable in hardware. Instead of having multiple IQ Modulators for each subcarrier, it is possible to associate to each symbol in the data stream(s) the respective amplitude and phase for the modulated signal. It is common to use complex numbers to encode both these properties in a single location.

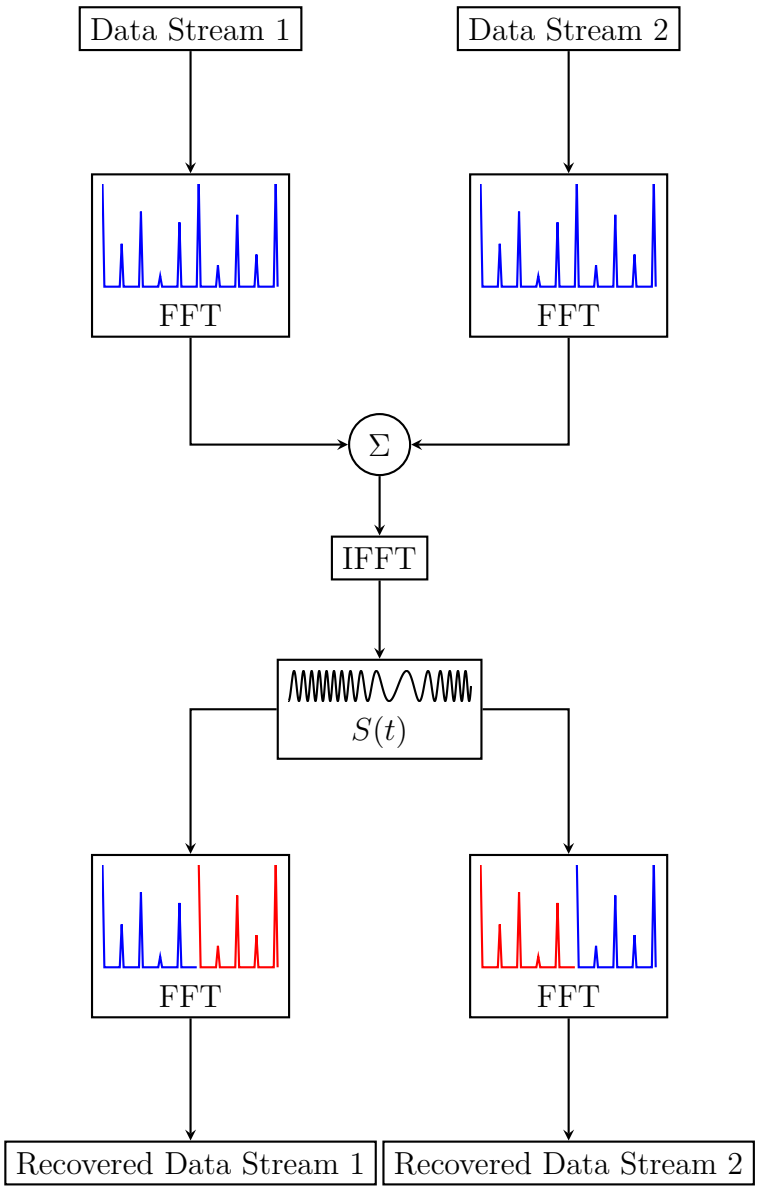


Figure 2.12: OFDMA encoding Fast Fourier Transform (FFT) and multiple decoders. The two symbol streams get encoded into a spectrum representation which is then converted in the time domain thanks to the Inverse FFT function. At the receiver side, each device will decode only the portion of spectrum allocated to it.

Then, we can construct a spectrum with the given frequencies and apply an inverse FFT in order to get signal representation in time. This signal can then be sent through a radio interface and received by multiple clients, which can decode only the part of the spectrum they are interested.

One additional improvement to the transmission efficiency is made possible thanks to the use of OFDMA. The advantage comes thanks to the receiver not having to capture the whole channel bandwidth in order to decode the data stream. Only the assigned portion of the spectrum is needed, thus there is a greater margin for low-power applications.

In general, the use of FFT based methods is preferred due to the ability of selecting and introducing subcarriers without radical changes in the chips. With the ability of new integrated circuits to accelerate FFT and Inverse Fast Fourier Transform (IFFT) calculations, the overall system reached a great optimization and simplicity level. With the addition of orthogonal frequencies that eliminates the needs for a per-subcarrier filter, the system gains impressive spectral and power efficiency due to the removal of losses of the circuit replaced by pure digital computations.

3 — The 802.11ax standard

The successor of 802.11ac is the "802.11 High Efficiency (HE)" protocol, released by IEEE in the year 2021 and rapidly conquering the consumer markets. This update addresses multiple limitations of the previous standard, especially in terms of speed and overall data rate in the network.

It focuses mostly on the radio physical interface, where it introduces new channels in the 6 GHz band for faster transfer rates and new modulation schemes, such as 1024-QAM, capable of more than 1 Gbit*p*s per spatial stream in high bandwidth channels of 160 MHz [7].

Benchmark comparisons denote only a little less than a 40% improvement in terms of raw data rate with respect to the previous standard [26], thanks to the adoption of faster modulation schemes. This happens due to the main focus of the standard having shifted to increasing overall network output and efficiency, which is reported to have improved by 300% alongside a 40% of latency improvement [27] thanks to the new approach at multi-user communications thanks to OFDMA, which represents a technology alignment to the current generation cellular network.

Switching to OFDMA for multi-user communications in 802.11ax, in both uplink and downlink directions, allows up to 72 contemporary data transmissions, excluding MU-MIMO, granting higher spectral efficiency and improved transmission timings and jitter[7], [28].

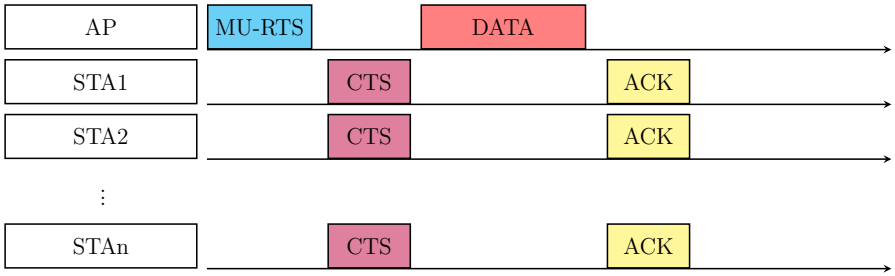
The multi-user access scheme for downlink has been simplified to its limit. Whenever an AP determines, thanks to its scheduler, that a multiuser OFDMA downlink transmission is advantageous, it will simply propagate the OFDMA packet to the radio channel by utilizing a variation of

the RTS-CTS mechanism. Note that in this case, the ACK of each packet (or packet group) is usually sent by each STA (if needed) as an Uplink Orthogonal Frequency Division Multiple Access (UL-OFDMA) packet.

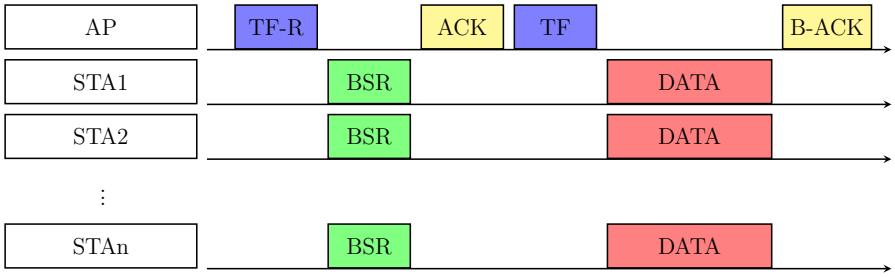
The AP will send a Multi User RTS (MU-RTS) to the involved sta, containing the Resource Unit (RU) allocation for the CTS. Then the data is sent to all the STAs at the same time leveraging OFDMA. As for the CTS, the STAs then send back the ACK with the same RU allocations used in the data frame.

UL-OFDMA is also straightforward: as observable in Fig. 3.1b, whenever the access point's scheduler determines that an uplink multiuser transmission could be beneficial or needed to improve efficiency, like for block ACK of a Downlink Orthogonal Frequency Division Multiple Access (DL-OFDMA) transmission [7], the AP will send a "Trigger Frame" requesting all the involved clients to start sending their uplink data in the RU allocation defined in the trigger frame. The AP can also query the network uplink status with the reception of Buffer State Request (BSRs), obtaining information relative to the uplink situation in order to take decisions. This "Trigger Frame" is then expected to be followed by the uplink transmission of each STA reported in the previously mentioned trigger frame.

This feature requires the AP hardware to be able to decode all the OFDMA transmissions at the same time, giving a great insight of each chipset manufacturer's hardware. Additionally, the previous generation spatial diversity access has not been discontinued: it is possible to send MIMO frames inside each OFDMA RU [7].



(a). DL-OFDMA transmission procedure.



(b). UL-OFDMA transmission procedure.

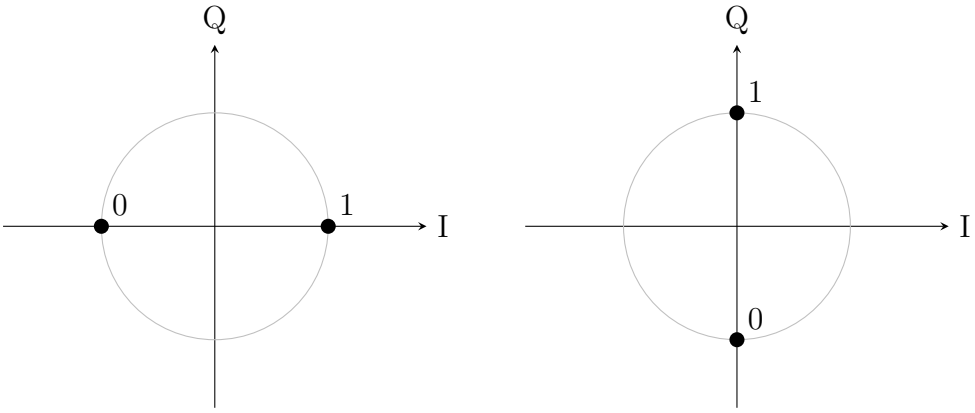
Figure 3.1: Multi user communications in 802.11ax

3.1 – 802.11ax PLCP

Every 802.11 HE communication utilizes information inside the HE preamble described in Fig. 3.3a. This data structure follows the PLCP discussed in Sect. 2.2, and it is encoded, generally, with the maximum available PLCP data rates. Note that the PLCP is still present in order to allow for legacy devices to detect the presence of a frame, thus making coexistence of the new radio protocol with older Wi-Fi apparatus possible.

Frame differentiation between all the various possible types and different standard versions is performed by the receiving device thanks to the modulation of the first two fields in the upcoming data after the legacy PLCP. Particularly, the differentiation is made by computing the average power of the I and Q demodulated signals.

The difference between each type of frame type consists in the mod-



(a). BPSK Constellation **(b).** QBPSK Constellation

Figure 3.2: Difference between BPSK and QBPSK constellation diagrams.

ulation of the first two symbols being BPSK or QBPSK. The difference between the two modulations, described by their constellations in Fig. 3.2, resides in the IQ signal used for modulation which makes the symbols appear as 90 degrees rotated BPSK symbols. By testing the difference of the power for each signal, it is possible to detect if the data is modulated with one of the two methods [29].

HE frames had to use a different method: at first, the identification of a HE frame is performed thanks to the repeated L-SIG (RL-SIG) field, which can be identified by the receiver. In order to differentiate between the four types of HE frames, the length field of the L-SIG is modified to permit a mod 3 calculation, where the result will indicate the effective frame type to be used for interpretation [7].

The generic structure of a HE preamble is defined in Fig. 3.3a. For each frame a HE-SIG-A structure is defined, and it is the first field after the RL-SIG. This field, described in Fig. 3.3b, contains information on the frame coding, bandwidth, and transmission options. These transmission options comprehend coding type used in the packet, channel bandwidth

utilized for the upcoming data, BSS Coloring (later explained in Sect. 3.2) and more features flags.

The HE-SIG-A then contains different fields based upon the frame type. For SU frames, it contains the upcoming MCS and other physical layer flags, whether for MU frames it contains information regarding the upcoming HE-SIG-B modulation and common user modulation properties.

If the frame is a MU frame, after the HE-SIG-A one or more HE-SIG-B (Fig. 3.3c) are inserted. In this case the content of the HE-SIG-A refers to the encoding and modulation of the upcoming HE-SIG-Bs. HE-SIG-B contains a common section, valid for every user in the OFDMA frame, and multiple dedicated sections containing information on the coding, modulation and physical parameters of the respective RU allocation.

The HE-SIG-B common fields contain the RU allocation for the upcoming data, including, if needed, definitions for spatial usage and the STA dedicated section allows the receiver to understand what frame has to be decoded by matching the AID.

The AP can still manage the network when sending control and management frames in the multi-user frames. Special AID are defined to identify broadcast RU allocations (AID 0), data directed to non associated stations (2045) or to define zero data RUs (2046).

The standard states that the receiving STA must decode its assigned AID from every Multi User (MU) frame. If the AID is not present, then the station must check and eventually decode AID 0 to detect broadcast data. Citing the standard directly [7]:

> *"An HE AP with dot11MultiBSSIDImplemented equal to false shall not include in an HE MU PPDU anything other than one or more of the following:*

- *One or more individually addressed RUs, corresponding to the parameter STA_ID equal to the AID(s) of STA(s) associated with the AP, to carry information intended for the STA(s).*
- *A broadcast RU corresponding to parameter STA_ID equal to 0 to carry information intended for the STAs associated with the AP that are not the recipient of an individually addressed RU.*
- *A broadcast RU corresponding to parameter STA_ID equal to 2045 to carry information intended for STAs not associated with the AP.*
- *One or more RUs corresponding to parameter STA_ID equal to 2046 for unassigned RUs."*

where the STA_ID refers to the AID previously defined. It is evident that a STA may decode a broadcast RU if and only if no data was directed to it.

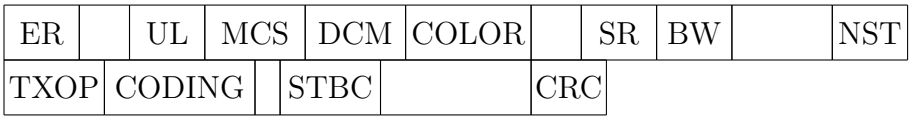
APs are only required to decode multiple RU in the same time frame for UL-OFDMA transmissions.

3.2 – BSS Coloring

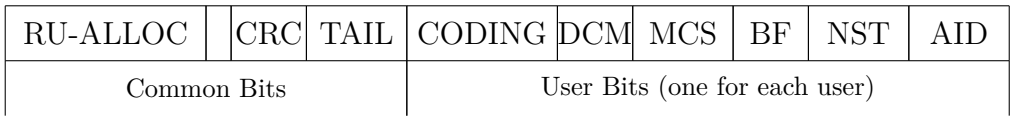
802.11ax includes methods for spatial reuse of frequencies defined as BSS coloring. This AP property allows for the reduction of channels interferences in a multi AP network [7].

The base concept behind coloring is the following. APswill start with a random color assigned to them and included in every beacon. This "color" is simply a tag inserted in the HE-SIG-A to identify the access point emitting

(a). 802.11ax HE Preamble structure.



(b). HE SIG-A field composition for Single User Frame Structure.



(c). HE SIG-B field composition.

Figure 3.3: Full description of an 802.11ax HE PLCP.

it. Each AP will listen to the incoming beacons from other APsin the neighborhood and, if the same color is detected, they will start a color change procedure notifying the connected STAs that a color change will take place in a defined amount of time. This allows for the connected stations to follow the color tag configuration and to keep the connection. STAs use the color tag each time channel access has to be performed, like as described by Fig. 3.4.

In order to perform channel access with CSMA-CA, if a signal is detected with the same BSS color as the one of the associated AP, the channel is considered occupied, and the collision avoidance takes place. If the received signal includes a different color than the BSS Color of the associated AP, instead, if this signal is below a power threshold, the channel is considered free even if a transmission is ongoing, because the two concurrent transmissions will not interfere since, even if received, different colors will prevent further elaboration by the other part [31].

The concept of BSS Coloring is very similar to what has been imple-

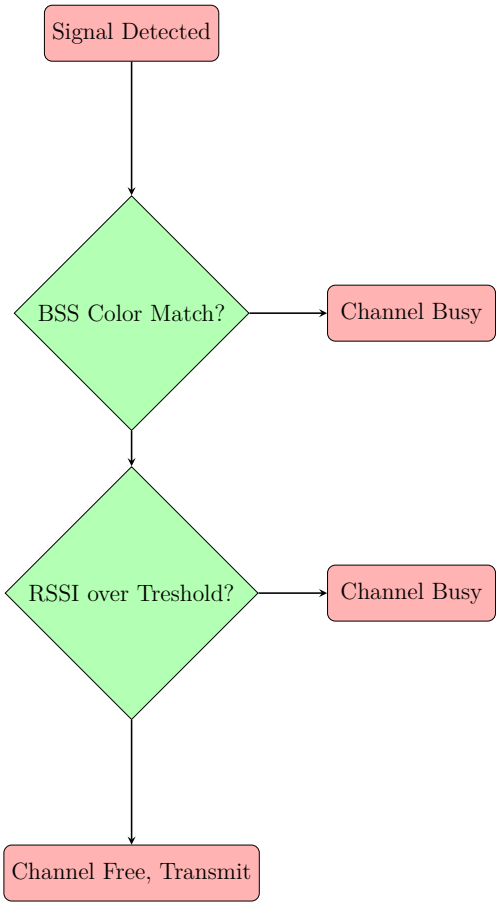


Figure 3.4: BSS Color carrier sensing procedure [30].

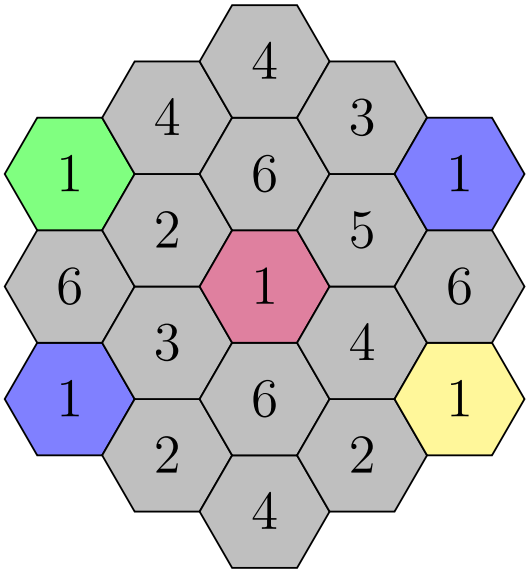


Figure 3.5: BSS Color Distribution. Note the interference radius is increased because distance between BSS with the same channel number and color is maximized.

mented in cellular networks, whose physical layout is shown in Fig. 3.5. Different adjacent cells may use different channels to limit interference. The problem arises when there is a limited amount of usable channels. This condition necessitates the reusal of a channel when not out of reach of the nearest cell using it [32]. The idea is that possible channel interferences are removed by assigning a tag to the transmissions, allowing the devices to ignore unrelated communications.

By coloring, thus associating the messages to the "cell" AP, devices will not be subject to Co-Channel Interference (CCI) and communication will not degrade.

3.3 – Multi User Scheduling

As previously mentioned, multi-user frames in 802.11ax are sent by encoding them in an OFDMA frame. This very spectrum efficient technology

Table 3.1: 802.11ax RU sizes [7].

RU Tones	Data Subcarriers	Pilot Subcarriers
26	24	2
52	48	4
106	402	4
242	234	8
484	468	16
996	980	16

poses the task of deciding how the available bandwidth has to be divided in order to provide effective multiple user transmissions [33], [34].

Even if the idea behind OFDMA is simple, its actual implementation is dependent on a very large set of statistics and transmitter related decisions.

Until 802.11ac every frame was utilizing the full channel bandwidth for modulation. This is considered a spectrum inefficiency because, even considering MU-MIMO, two stations with very different throughput may be grouped together in a multi-user transmission. This hypothetical multi-user transmission would be somewhat efficient for the time the two frames are overlapped, which, given the hypothesis of a fast and a slow STA, consists in a small portion of the frame.

The OFDMA solution, other than allowing for a greater number of concurrent users in a frame, allows for asymmetric allocation of RUs to guarantee spectrum efficiency, allowing the scheduler to reserve an asymmetrical allocation of RUs in order to make the frame times equals, thus minimizing the overall frame duration and network efficiency.

An RU is defined as the group of a given number of tones. The standard defines the possible groupings as shown in Tab. 3.1. The scheduler must be able to assign these RUs while transmitting packets to the STAs. Generally, the scheduler implementation checks how many packets are queued for transmission while preparing the next frame. If enough devices have

data queued and if other system conditions are met, such as number of packet per second the user is transferring, the AP will compose and send a HE_MU frame. Multi-user schedulers must also take into account the average data rate for each device in order to allocate different size RUs as a function of the needed throughput. Another difficulty that has to be included inside the scheduler is the ability to query and to define scheduling patterns for uplink data coming from the devices. Even in this case, it is up to the AP's scheduler to decide the RUs to be allocated in an OFDMA upload.

In OFDMA there is no *a priori* designation for the RUs to be allocated by a client. Each time the scheduler will assign a possibly different RU configuration in the frame for every device, thus making the allocation possibilities not subject to limitation of previous choices. OFDMA is stateless.

Utilizing OFDMA in a crowded network is useful. Latency times get reduced and data flow is not subject to sudden stops because medium access is performed in a crowded environment. In fact, simulations report that, in a pure DL-OFDMA, without OFDMA most packets will be blocked in the MAC First Input First Output queue (FIFO) waiting for a channel access being performed in a crowded environment [28]. Other studies highlight the importance of the scheduling algorithm to make the approach useful. This neat sensitivity to the scheduling process poses difficulties in the development of new chipsets [35].

This problem is also present in cellular networks where this technology has been first implemented in. Solutions are not public because of the algorithms' high monetary value, resulting in lots of different implementations causing a clear difference in performance inside different vendors networks.

4 — Sniffing

We define "sniffing" as the act of intercepting, without altering, some properties of a channel that make the operating subject (the sniffer) able to decode them into symbols representing data with a related meaning.

The concept of sniffing is prominent in the realm of communication networks. More or less every form of communication can be subject to sniffing, as it can be modeled as an eavesdropping attack without — hopefully – malicious intent.

Talking about on-cable networking, the sniffing can happen with the aid of network equipment that exposes a monitoring port, a feature usually used in Intrusion Detection System (IDS) [36]. In the realm of Wi-Fi networking, contrarily, the sniffing can be done without the cooperation of the involved users and devices. This is because the communication channel is shared between all the devices in reach of the signal without the need of cabling and physical access to the network; every radio channel is always accessible, provided that the receiver sensitivity is good enough [37].

The previously mentioned advantages comes with additional intrinsic difficulties: it has already been discussed in Sect. 2.2 how, in order to provide a reliable and fast channel between the network peers, there is the need of equalizing the channel. These features, alongside multi-user technologies, use precoding in order to apply beam-forming and spatial diversity streaming. These operations may result in disturbances — given the fact the receiver and the sniffer are not in the same place, causing the precoding not to maximize correct reception chances — or may limit the total amount of traffic the sniffer is capable of decoding if not in the

immediate neighborhood of the targeted receiver.

In general, it is very improbable to receive every possible frame while sniffing a Wi-Fi network. This is because interference and radio conditioning of the transmitted signal may introduce interference to the sniffer.

4.1 – Brief overview of sniffing history

In the past, where computers were technologically limited, it was easy to sniff traffic if access to the network infrastructure was available. There was no difficulty in intercepting data transiting in a shared network bus given the nature of the communication system, and there was practically no encryption in the data. This made analyzing and understanding the network traffic easier.

With the introduction of switches and point to point networking as we know for 802.3, there have been multiple upcoming difficulties due to the operation of switches, delivering only messages destined to the host (in the L2 network). This made it necessary to develop network attacks in order to make the network apparatus behave like a hub, thus sending every frame to the sniffing host.

4.2 – Sniffing applications

The primary application of sniffing techniques in the IT industry is for IDS. In such configurations, a network switch is often set up to mirror or copy all network traffic to a specific port. This port is connected to a dedicated host, which is deployed specifically for analyzing the network traffic to identify common attack patterns and suspicious activities. By monitoring traffic in this manner, IDS can detect unauthorized access attempts, malware, and

other security breaches in real time, allowing administrators to take swift action [36].

In traditional wired networks, this method is highly effective because the entire data stream passing through the switch can be captured and analyzed. However, Wi-Fi sniffing presents additional challenges. In wireless networks, it's more common to place network analysis systems deeper inside the core network infrastructure rather than on the periphery. This is primarily because IT administrators are often more concerned with threats originating from remote attacks, rather than those coming from local wireless access points. Nonetheless, while Wi-Fi sniffing may be less prevalent in IDS systems due to these limitations, it remains a valuable tool in other contexts.

One significant application of network sniffing, even in wireless environments, is for network performance analysis and troubleshooting. IT professionals use sniffers to diagnose issues with existing network hardware or when deploying new appliances. For example, if a network experiences slowdowns or intermittent connectivity issues, a sniffer can capture and analyze the traffic to pinpoint the problem. This is particularly useful in identifying packet loss, latency issues, or bandwidth bottlenecks.

In addition to troubleshooting, network sniffing is also extensively used in benchmarking network performance. Academic researchers and network engineers often set up controlled environments where they can test different network protocols or technologies. By analyzing the distribution of packet types, they can establish baseline performance metrics. These benchmarks are crucial for understanding how a network will perform under various conditions, such as high traffic loads or in the presence of specific types of traffic (e.g., streaming video versus bulk data transfer).

For example, in a Wi-Fi network, a sniffer might be used to determine the proportion of packets that are management frames versus data frames. Another possibility is to evaluate the number of multi-user frames sent to prove the efficiency of a scheduling algorithm. Understanding this distribution helps in optimizing network configurations for different applications, such as VoIP, which requires low latency and high packet delivery reliability.

Another critical application of network sniffing is for eavesdropping on communication channels to capture illegitimate information transmitted over the network. Malicious actors often use sniffing techniques to steal sensitive data, such as passwords, credit card numbers, or personal information. This type of attack is particularly concerning in public Wi-Fi networks [38] since 2% of the global Wi-Fi BSS are unencrypted, totaling to more than two billion networks. In these open networks, the setup of a Wi-Fi sniffer can be done with minimal effort. Public networks provide an easy target for data thieves.

To counteract these threats, the implementation of secure communication channels and strong encryption protocols is crucial. For example, the use of WPA encryption, namely WPA3 [2], in Wi-Fi networks significantly reduces the risk of data being intercepted by a sniffer. However, despite these measures, the potential for eavesdropping can never be entirely eliminated.

4.3 – Ethical considerations

When discussing sniffing in a Wi-Fi network, both legal and ethical considerations are paramount due to the potential for significant privacy violations and misuse of intercepted data.

Legally, sniffing Wi-Fi networks can often fall into a gray area depending

on the jurisdiction. In many countries, laws make it illegal to intercept or monitor communications without the consent of the parties involved. This means that if someone is capturing data packets from a Wi-Fi network that they do not own or do not have explicit permission to monitor, they could be in violation of the law. For example, in the European Union, the General Data Protection Regulation, also known as GDPR, imposes strict regulations on how personal data is handled and accessed [39]. Unauthorized interception of data through Wi-Fi sniffing could lead to severe penalties under GDPR, including hefty fines. In Italy, information gathered through sniffing of a Wi-Fi network resulting in access to IT systems may incur jail time according to article 615 ter of the penal code *"Accesso abusivo in un sistema informatico o telematico"* which protects against unwanted access to the network [40].

Additionally, certain countries have specific laws governing the use of wireless communications, making it illegal to intercept or decode encrypted communications. Even if a network is unsecured or open, this does not necessarily grant legal permission to intercept and analyze the data transmitted over it. Furthermore, courts have increasingly recognized the expectation of privacy that users have when using wireless networks, which can influence legal interpretations of what constitutes lawful or unlawful sniffing.

Ethically, the act of sniffing Wi-Fi networks presents significant concerns, even if it is technically legal in some situations. Unauthorized sniffing can lead to breaches of confidentiality and trust, as individuals and organizations expect their communications to remain private. This expectation is particularly strong in the context of sensitive information such as personal emails, financial transactions, or private messages. Even in cases where the network is public or open, the ethical principle of respect for

privacy suggests that users' data should not be intercepted and analyzed without their informed consent [40].

Moreover, ethical considerations also extend to the potential for misuse of the information obtained through sniffing. The intercepted data could be used for malicious purposes such as identity theft, corporate espionage, or unauthorized data collection. This raises questions about the intent behind sniffing activities and whether they are conducted for legitimate purposes, such as network diagnostics and security testing, or for harmful and exploitative reasons.

4.4 – Implementation

Almost every network interface has the possibility to be put in a reception mode called "monitor mode". This configuration is usually made by the device driver and instructs the device to send up to the kernel every packet received without performing validation. The device effectively becomes a physical layer interface used only for reception.

The device driver then must create or reconfigure the network interface of the system in order to not drop unexpected packets. Once the driver and the device are configured to deliver every packet to the host, a program to log and analyze the data has to be used.

The most used programs for network packet analysis are "tcpdump" and "Wireshark". The two programs make use of the "Packet Capture Protocol (PCAP)" file format to handle packet identification and reception metadata decoding. This format is widely used because it allows both stream-processing and file storage into a vast variety of analysis programs like IDSs and network tracking.

4.4.1 – The PCAP File format

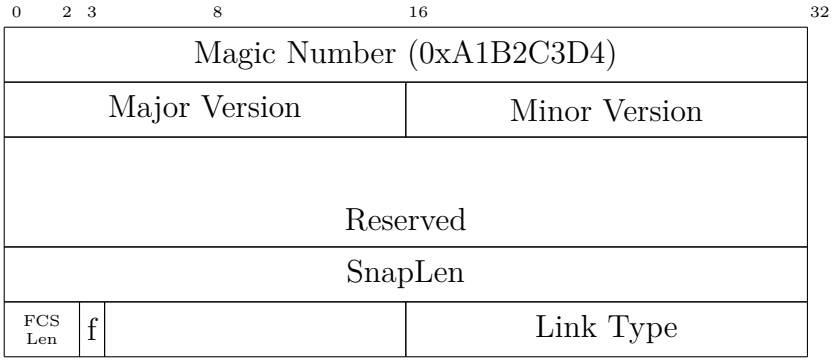
The PCAP format can be considered an inter-application communication interface for the transfer of data packets. It has initially been developed in the late 1980s by a group of researchers at the Berkeley National Laboratory.

The format is defined as a packet streaming protocol divided in multiple packet records headed by the PCAP header [41].

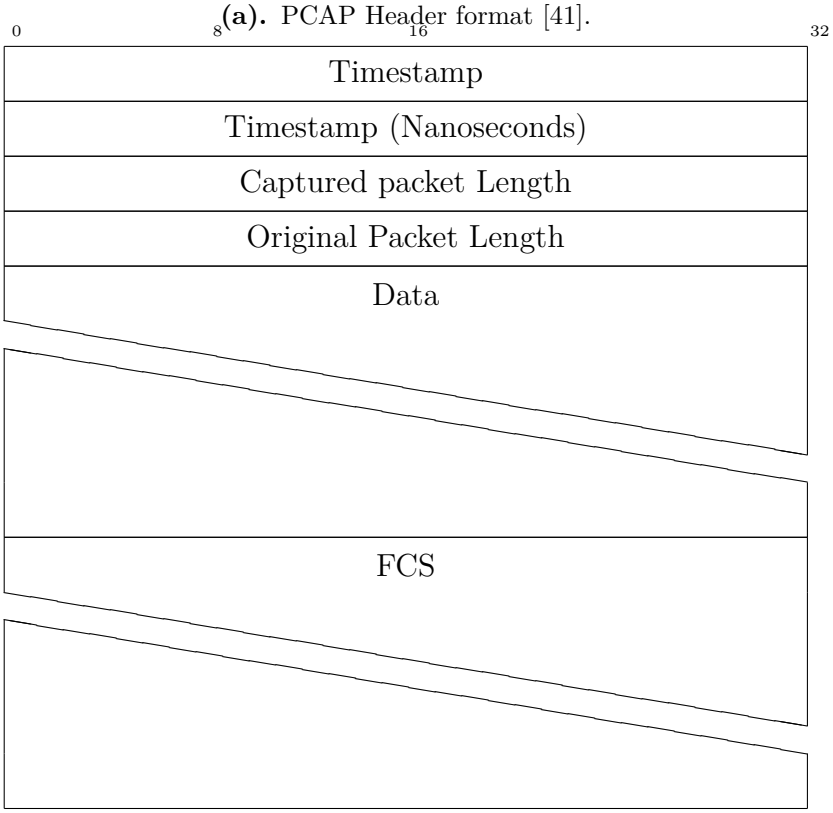
On capture session opening, the capturing program will report at the beginning of the PCAP stream a header, defined in Fig. 4.1a, where the protocol version and some future packet information is stored. Particularly, there is a field, namely SnapLen, to indicate the maximum buffer size the receiver is allocating, thus limiting the maximum number of octets in each packet.

Then there is the possibility to report the presence and the length of the Frame Check Sequence (FCS) (if present) appended to the end of the packet. Frame check sequences are generally used for error-detection of the data contained in the packet and are commonly implemented as Cyclic Redundancy Check (CRC). The "f" bit defines the presence of the FCS, and the value of the field "FCS Len" reports the number of words (16 bit) appended to the packet.

The last field of the header is then a 16-bits identifier to specify the link type to the parsing program. There is a publicly available table of possible values, and each value may require an *ad-hoc* base dissection layer for analysis. Fig. 4.1b shows the single captured packet structure, where the reception timestamp (with respect to the beginning of the stream) is reported with a millisecond or nanosecond precision. Two length fields are included in the header, indicating the captures size and the original



(a). PCAP Header format [41].



(b). PCAP Packet Entry [41].

Figure 4.1: PCAP format definitions.

size received by the device. This is done because usually all the requred informations are stored in the header, and most of the time the body can be discarded.

Note that the dimensions reported does not take into account the FCS.

The simple implementation of the PCAP protocol makes it the first choice when implementing packet analysis scripts. This solution allows for a great interoperability of analysis scripts and programs, because it abstracts their data gathering interface from the device and OS level.

5 — Validation and testing of State-of-the-art technologies

Currently, almost every Wi-Fi chipset vendor allows to enter monitor mode with their products. It's obvious that every vendor will implement this functionality with quirks and details derived by the hardware construction, because this feature is not standardized by any association.

For this work, we started by evaluating the current state of the art in terms of multi-user 802.11ax sniffing. This work will concentrate its efforts in decoding DL-OFDMA frames. While analyzing a network in an high load situation, it can be stated that DL-OFDMA is more common given the improved spectral efficiency and network throughput advantages. We decided, after an extensive evaluation of the possible hardware candidate capabilities, to test Intel's implementation of the 802.11ax chipset: the Intel AX210. This platform has been chosen because of the reported multiuser monitor sniffing capabilities, which are made obvious and explained in the chipset's linux driver, available as open source [42].

Sadly, the presence of an open source driver does not mean that researchers have access to every aspect of the chipset's hardware. The kernel module interacts with the closed source firmware in the Wi-Fi board that carries out all the real time work. [43]

5.1 – Sniffing with Intel AX210

The first experiment made with the Intel AX210 is to test the ability to capture DL-OFDMA frames. Given how the linux driver works, it's evident that the chipset can only decode one AID for each DL-OFDMA frame.

The following sections will report some experiments done with a given setup, providing some comments over the obtained results.

5.1.1 – Setup

For the first setup we tested the AX210 capabilities to monitor DL-OFDMA traffic. The setup for this experiment consisted in:

- AX210 Host, placed in monitor mode
- Asus RT-AX86U (Broadcom BCM4908) in access point mode. This device was set up with DL-OFDMA enabled and was assigned a non-used channel (157/80MHz).
- Two Asus RT-AX86U (Broadcom BCM4908) configured as STA.

The APs consist in a 2 GHz quad-core device with two radio interfaces at 2.4 GHz and 5 GHz. The 5 GHz radio is a separate dongle powered by a Cortex-A7 BCM43684 which has been the main focus of this work.

All the experiments have been made in the 5 GHz band leaving 2.4 GHz analysis for future work.

The layout of the experiment is described in Fig. 5.1.

Experiment #1

The aforementioned Wi-Fi network was configured as an open Basic Service Set Identifier (BSSID) where the two RT-AX86U were configured as

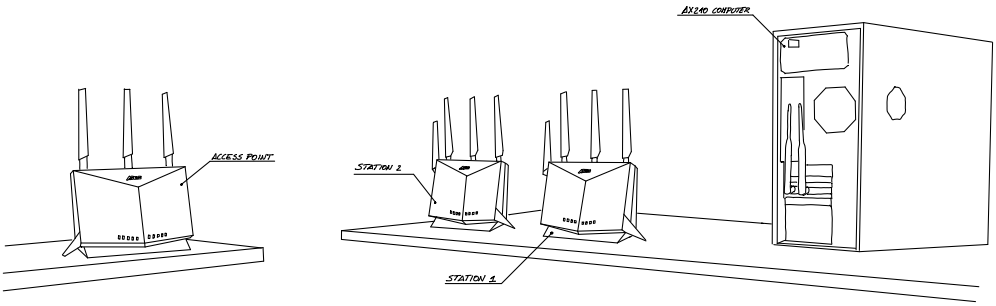


Figure 5.1: Physical location of the hardware in the experiment room. No people were inside the room at the time of the experiment.

stations and were associated to the third RT-AX86U working as an AP. In order to limit packet interferences from other BSS, the network was tuned on channel 157 with 80MHz of bandwidth. This choice ensures without any major doubt that no interference can be registered by the sniffer, thanks to a previous survey which have determined the nearby Wi-Fi network did not use the high part of the 5 GHz. spectrum. Each STA device was equipped with an *iperf* server ready to accept sessions. When the experiment started, the monitoring host was put in monitor mode (see App. A) and two *iperf* sessions were started on the AP to the two STAs.

This setup allows to send frames to the STA without ethernet link limitation from an external host. Since *iperf* generates large frames, we are sure that no particular slowdown for packet generation in user-space is present.

On the monitoring host, the traffic was captured with *tcpdump* and then analyzed on the fly by a custom-made script used to count multiuser frames received thanks to the *radiotap* header prepended by the intel device.

After one minute of recording, the monitoring was interrupted and the results were analyzed. Experiment results reported that no data frame was decoded, thus no packet estimation could be derived, given the fact that

neither the number of stations nor the RU allocation is available in the *radiotap* header.

In order to perform sniffing with the AX210 chipset family, a valid AID must be configured.

Experiment #2

Following the experiment in Sect. 5.1.1.1, without any configuration change, we repeated the test. This time, before the beginning of the experiment, the status of the AP multiuser scheduler was collected, given the fact that we had full access and control over the AP. This resulted in the availability of the connected AID and their association with the MAC address.

The recording produced results similar to the experiment in Sect. 5.1.1.1 in terms of raw frames received, but this time the chipset was able to decode packets from the frames directed to the selected AID. This is represented by the second series in Fig. 5.2 indicating the total number of packets received in a given time interval. We see more packets than frame because Aggregated MAC Service Data Unit (AMSDU) and Aggregated MAC Protocol Data Unit (AMPDU) were enabled in the AP.

Without considering AMSDU and AMPDU, we can estimate the total number of packets in the network as twice the number of frames, given the 50% duty cycle in AID switching.

This is feasible because of the simple network layout and the predictability of the *iperf* traffic. In a real network, this estimate will not be very precise due to variation in the network load.

As a test bench, a traffic capture was run on the AP in order to later match and estimate the number of lost frames. Overall packet loss results are shown in Tab. 5.1.

Figure 5.2: Experiment 2, Number of received packets per time period by the AX210 sniffer.

Table 5.1: Setup 1, Experiment 2, Results

Metric	Value
Total number of packets transferred	2296373
Total Number of packet per AID	1140461
Total Number of packet sniffed	803505
Total Number of packet lost	334391
Percentage of packets sniffed	40.9%
Percentage of packets sniffed for the AID	70.1%

The same experiment has been performed with a 4-NSST configuration reporting no data captured by the AX210 because of its bi-antenna layout.

Experiment #3

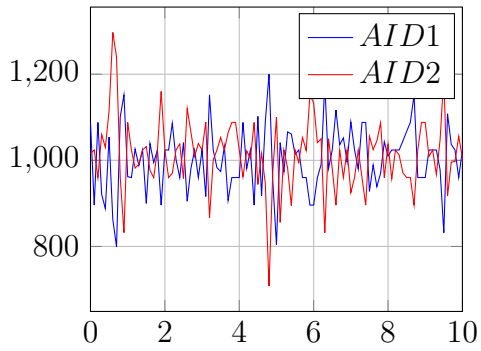
To get a better view of the network packets transmitted on the channel, we repeated the experiment in Sect. 5.1.1.1 and 5.1.1.2 with an additional script cycling through a preconfigured list of AIDs and configuring them as the current target AID for the AX210 every second.

The AID list, as previously stated in Sect. 5.1.1.2, has been obtained thanks to the full access to the AP scheduler information.

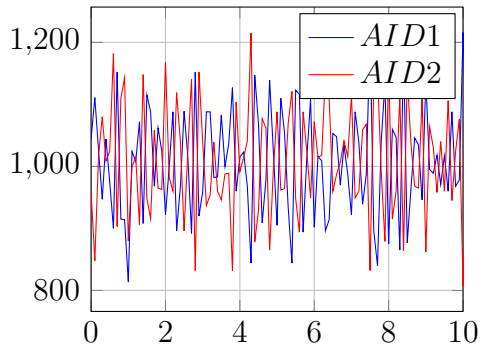
By varying the switching time between the AIDs, we were able to prepare a benchmark of the sniffing capabilities in function of the switching time, evaluating the device jitter and performances.

Fig. 5.3 shows the number of packets captured by the device divided by the AID, across the switching time range of 1ms, 10ms, 100ms and 1s. It is possible to evince the switching nature of the process as the switching interval becomes bigger than the sampling interval, which remains constant at 0.1s.

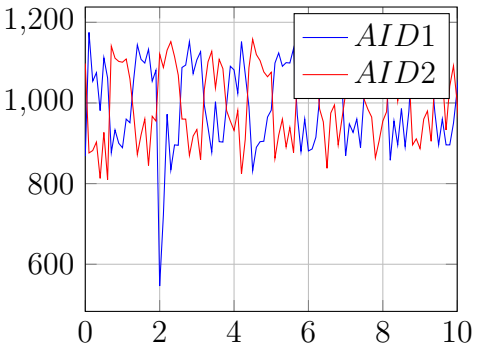
In terms of jitter timing for the switching, it is expected that the histogram of inter-frame timings should present two peaks: one near zero, because of all the frames received in a single switching interval, and one near the value of the switching interval. This is because we should be able to detect at least as many frames with inter-frame interval long as much as the switching interval or more, as the number of switches in the experiment. If during the experiment the total amount of AID switches was a number n and the switch time was t_{switch}, we should expect at least $n-1$ reported inter-frame intervals greater of equal t_{switch}. The more thhese inter-frame



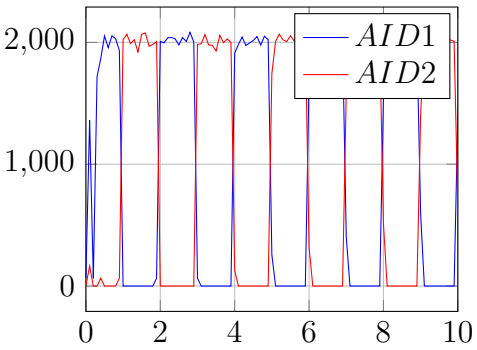
(a). Received packets in time, $\Delta_{switch} = 0.001s$.



(b). Received packets in time, $\Delta_{switch} = 0.01s$.

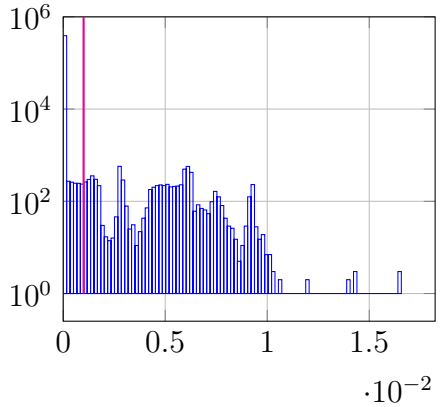


(c). Received packets in time, $\Delta_{switch} = 0.1s$.

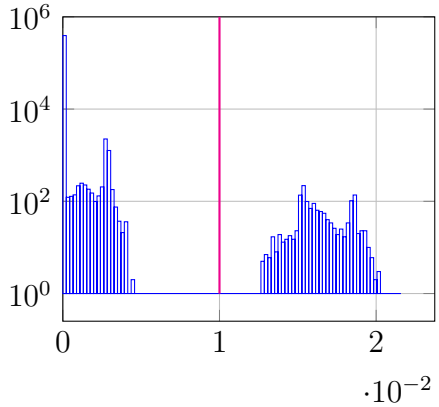


(d). Received packets in time, $\Delta_{switch} = 1s$.

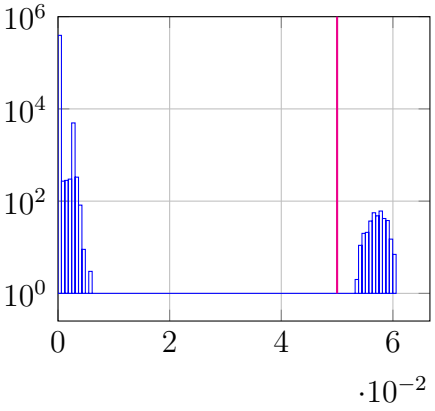
Figure 5.3: Experiment 3, Number of packets sniffed per time interval and AID.



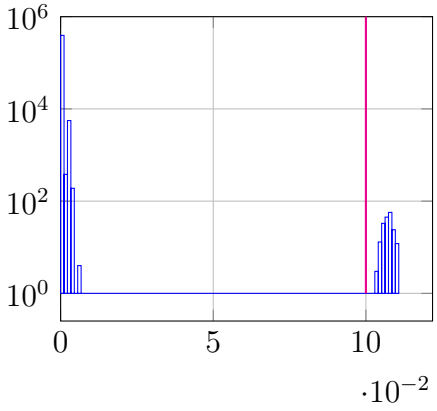
(a). Inter frame interval distribution, $\Delta_{switch} = 0.001s$



(b). Inter frame interval distribution, $\Delta_{switch} = 0.01s$



(c). Inter frame interval distribution, $\Delta_{switch} = 0.05s$



(d). Inter frame interval distribution, $\Delta_{switch} = 0.1s$

Figure 5.4: Experiment 3, Inter-Frame time distruibution per switching time interval.

Table 5.2: Setup 1, Experiment 3, Results.

Metric	Value per Δ_{switch}			
	0.001s	**0.01s**	**0.05s**	**0.10ms**
# of packets transferred	2136617	2320231	2341005	2392980
# of packet for AID1	1079704	1158354	1172402	1200135
# of packet for AID2	1056910	1161875	1168600	1192843
# of packet sniffed	801224	801708	799964	800590
# of packet sniffed for AID1	399376	399622	400420	399382
# of packet sniffed for AID2	401848	402086	399544	401208

times are longer than the switching time, the worse is the traffic sampling granularity, providing less precision in the description of the network load and behavior when burst of traffic are present. Fig. 5.4 demonstrates what previously mentioned exposing the limits of the platform. In fact, for fast switching timings such as 1ms and 10ms, the second peak appears nearly twice as far as expected. The spread around this peak, also, enables us to infer the jitter charateristics of this process.

Because the switching process is driven in user space, it was expected that jitter would have played a big role in the charaterization process.

We were able to also define a limit switching interval for this technique which demonstrates to be in the interval $[100ms - 10ms]$.

Capture statistics of the experiments are represented in Tab. 5.2.

Results

The current state of the art, while allowing for a multi-user OFDMA monitoring by switching the AID during capture, is able to provide only a rough estimate of the network performance with a rough granularity because of the limited performance in switching times.

As shown by experiments in Sect. 5.1.1.3, the fact that AID switching is performed in the user space does not allow for an in depth control over the

hardware, resulting in less precise data with consistent jittering imprecision. Also, the AX210 is one of the few devices allowing for multi-user sniffing in 802.11ax. This new technology allows for possibly an easier sniffing because it does not depend on spatial diversity and precoding in order to decode data, but the decoding hardware is not fully controllable by the user, resulting in difficulties in the sniffing.

The fact that AX210 does not support 5GHz AP mode — in fact it can only be used in a "soft AP configuration only on the 2.4 GHz" — implies that the only product capable of MU sniffing does not support the full AP capabilities needed to decode frames concurrently, needed for UL-OFDMA, increasing the inefficiency of the sniffing process.

5.1.2 – Limitations

The current monitoring setup for 802.11ax multi-user DL-OFDMA presents significant limitations that impact the effectiveness and comprehensiveness of network traffic analysis with the AX210 hardware setup.

Specifically, this setup is constrained by the ability to recover only one frame per multi-user transfer, based on a given Association ID (AID). This restriction inherently leads to the loss of other frames within the same multi-user transmission, thereby preventing a complete capture of all transmitted data. Consequently, this results in an incomplete picture of the network's traffic, as only a subset of the transmitted data can be analyzed.

Moreover, the necessity to change the AID every second to capture traffic from the entire network introduces further complications. This approach is inefficient and prone to missing critical data, as it relies on the manual or automated cycling through AIDs, which may not align with the actual transmission schedule of the AP.

The inability to recover multiple frames simultaneously may obscure important traffic patterns and behaviors, such as load balancing, client-specific issues, or security breaches. The cumulative effect of these limitations is a monitoring setup that may not fully reflect the network's dynamics, potentially leading to misguided conclusions or overlooked issues.

The lack of an easy and usable method to detect and select AIDs exacerbates this issue, as it necessitates either a manual configuration or reliance on the AP's scheduler. Dumping the scheduler data on the AP, while possible, is often impractical and may not provide real-time or near-real-time insight, which is essential for effective monitoring.

These limitations could lead to significant gaps in the captured data, affecting the reliability and accuracy of network performance analysis, security monitoring, and troubleshooting. Also, captured data comes in bursts of one second, and they get more spread over with the increasing number of users in the network, resulting in a less accurate understanding of the networks functioning due to discontinuities in the available data.

6 — Changing approach: Broadcom hardware

As previously stated, this work aims to develop a new set of tools for OFDMA network sniffing.

The hardware choice of the Broadcom BCM46384 comes thanks to the availability of Nexmon, a framework that allows for the complete access to the chipset internals and permits its firmware modification [44].

We must acknowledge that the Broadcom firmware is closed source, but the aforementioned tool is applicable thanks to the great effort in reverse engineering the platform. This tool, with the addition of some reverse engineering of the PHY registers in the chipset, allowed us to gain a deeper understanding of the chipset's structure. Thanks to a multitude of tests and operations such as fuzzing, we were able to adapt the hardware behavior to our scope.

Before handling the effective modifications needed to perform the network analysis and before displaying the newly found behavioral discoveries of the OFDMA hardware, it's necessary to have an understanding of the chipset itself and its architecture.

Please note that the work here presented may be ported to multiple chipsets — up to the next major hardware revision of the chipset — with specific adaptations.

The BCM46384 offers up to four spatial streams on both 2.4GHz and 5GHz bands with support for 802.11ax High efficiency. As previously mentioned in Sect. 5.1.1.1, during the development of these tools, we only

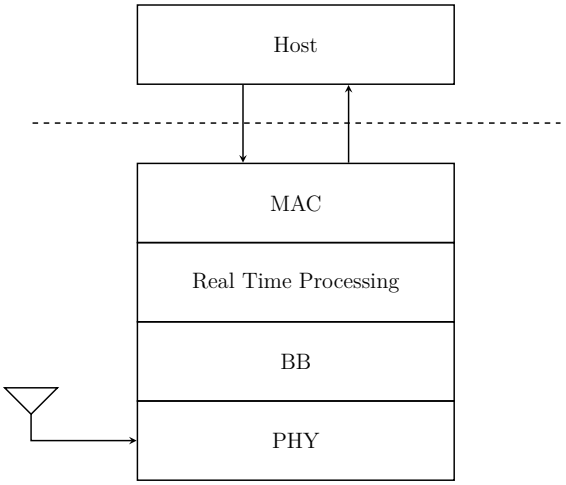


Figure 6.1: Layer organization of the Broadcom hardware and software interface to the user space.

utilized the 5 GHz capabilities.

A high level overview of the platform, described in Fig. 6.1, can be divided in four macro layers handling the different needs for a fast and reliable communication.

The chipset consist in a full-MAC architecture, later described in Sect. 6.2, where every Wi-Fi domain operation is handled in a separate CPU without host interaction.

The base layer, identified as the PHY layer, is the main physical interface to the external world. It includes every component needed to perform the hardware side of the modulation and demodulation, such as signal conditioning and modulation components for OFDM network access and much more. This layer also includes the required DAC and ADC to perform signal synthesis and decoding [45].

The second layer, the BB, contains all the required digital peripherals to perform modulation and demodulation, with the addition of dedicated hardware offload components to allow for faster processing of the 802.11

structures during modulation and demodulation.

Up until now, every layer contains hardware peripherals with no (or very little needed) microcode.

The first interesting layer containing executable code is the Real Time Processing core. There is no official documentation on its architecture other than the name: D11. The D11 core(s) is responsible for the real time processing of the frame, and it can operate on the data even while the Physical Layer (PHY) is still receiving it [45].

The last macro-layer is the MAC. This layer is the responsible for handling "slow" Wi-Fi operations such as handling connection requests. Real time operations such as ACKs and beamforming reports, which must obey strict timing requirements, are handled by the D11 instead.

6.1 – D11 Core

The D11 real time processing core is the main actor for the low-level communication capabilities. It features a 64bit fixed size instruction set with numerous internal registers.

Thanks to previous work made when developing Nexmon, the basics of the internal architecture are now known. The D11 handles frame manipulation with a set of hardware and software instructions and communicates with the upper layers thanks to a shared memory section [44].

Internally, the D11 core presents itself as referenced by Fig. 6.2.

There are a couple of caveats to understand the architecture: initially, lots of decoding and operations on the received frame data are made in hardware. This is evident when reading disassembled code from the distributed firmware in the official driver. There are multiple registers getting populated by the dedicated peripherals with adequate data without the

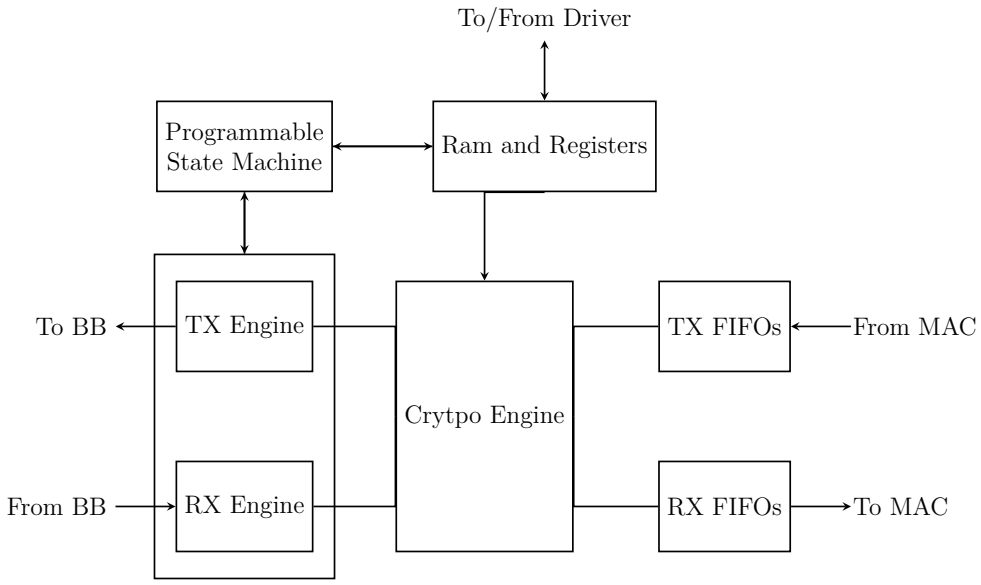


Figure 6.2: D11 internal layout.

need for code to carry out the decoding. This is mirrored by the fact that multiple instructions will behave differently depending on the state the core is in. This implies the presence of a microcode executing in the background optimized for particular tasks.

In general, it is evident that this core is very multipurpose. Its code is loaded at driver initialization and can vary whenever some features are required. For example, the main driver will load a different D11 firmware whenever Bluetooth coexistence is needed or not.

6.2 – MAC Layer

The MAC layer is the high level responsible for non-time-sensitive operations. This layer manages the hardware whenever power limits are enforced and is the main controlling unit of the device. It manages the communication interfaces between the chipset and the host processor and handles generic driver activities.

Up until 802.11n, Broadcom chipsets implemented the MAC layer directly in the host driver, communicating directly to the D11 core on the chipset. This solution guarantees simplicity and cost optimizations, but the ever-increasing computational speed and efficiency required by modern standards limited its application.

The host implementation of the MAC layer, commonly known as Soft MAC, is subject to scheduling jitter and host/OS inefficiency in handling real time data access. This solution was only possible because more operations were delegated to the D11 core running on the hardware, and because of the limited speed required by the standard [45].

Modern chipsets use a different architecture, Full MAC. This architecture introduces an ARM core, usually a Cortex-M4, between the D11 and the host. This core is the responsible for non-real-time operations such as scheduling configuration, AP association, AP mode management operations and interface management.

The presence of this additional layer, introduces communication difficulties between the D11 and the host, which are solved with DMA access discussed in Sect. 6.3, but allows for the possibility of handling Wi-Fi events in low-power hardware consenting the host CPU to sleep until an event or data is received.

6.3 – Data Reception

Data reception begins with the identification and the demodulation of a signal by the physical layer. The signal travels through the layer where the required signal conditioning and amplification occurs. The signal is then sampled by a set of Analog to Digital Converters (ADCs) that allows the BB to handle the newly digitized samples. Here the raw samples are

decoded with the necessary signal processing steps.

For OFDM and OFDMA frames, the Base Band (BB) shall first detect the presence of the frame by identifying and correlating the LTF field. This field is used, as stated in Sect. 2.2, to provide channel equalization and to detect the effective start of the frame. This step is performed in hardware and generates the required CSI data to equalize the signal, compensating for amplitude changes and phase shifts due to channel disturbances and reflections.

Once the frame is detected, FFT and other reception operations are performed to decode the frame, and they are supplied to the D11 for elaboration as bytes. The D11 is also able to read some other hardware-related values decoded by the lower layers with the use of special hardware registers.

The D11 then receives the raw data through a FIFO and starts interpreting them. These operations are performed mostly by a programmable state machine, which reacts to changes faster than code could do. This PSM then controls the various decoding engines implemented in hardware with whom the D11 core interacts with in order to perform the data reception.

Data transmission to the host then happens thanks to the presence of hardware FIFOs. Data is sent up to the ARM core with the help of DMA and ring buffers. Here the Full MAC will apply Ethernet headers as needed and will handle management frames as defined by the standard. Note that data pushback to the host is then performed by DMA or by placing the content directly in the host's physical memory thanks to PCIE DMA access.

6.4 – Data Transmission

For data transmission, data is sent from the host or from the ARM processor (when particular management activities require sending frames) thanks to DMA. The ARM firmware will handle data processing for the packets coming from the host and will place them into a ring buffer that another DMA peripheral will use to send the data to the D11. Here the core handles the most part of multi-user scheduling (if needed) and data processing to form the raw frame which is then sent to the BB for encoding and then out to the channel thanks to the PHY layer.

7 — Tool development

The main focus of this work, as previously mentioned, is to develop a toolset for OFDMA network sniffing.

Thanks to both Nexmon and some GNU General Public License (GPL) Software releases by Asus, we were able to fully dissect and reverse engineer parts of the D11 firmware. Especially, we were able to identify the point where the D11 starts executing while a new frame is being received. The reception flow in the D11 firmware starts after the detection of an incoming packet, thanks to the STF field in the PLCP. The hardware is responsible for the channel setup and for the PLCP reception.

Once the PLCP has been received and has been validated, the D11 can now detect what type of frame has been received.

Then, after the full PLCP reception, the frame is decoded and, based upon its type, dedicated operations are performed, such as sending the frame up to the host or managing the hardware based on information coming through the received packet.

One example of this behavior is the handling of HE Trigger Based frames or UL-OFDMA. In these cases, the D11 will send a packet from the TX FIFO whenever an HE trigger based frame is received containing the required transmission parameters.

7.1 – Understanding the Hardware

As we are interested in the DL-OFDMA decoding hardware a long reverse engineering of the D11 firmware and testing have been performed.

Following the reverse engineering of the disassembled firmware, and thanks to the definitions cross-referenced to the GPL source code release, we are able to make a description of the receiving hardware layout and design.

At first, we discovered that there is a PHY register filtering out the correct color packets while receiving. We suspect that this operation is made in hardware because this allows for the frame decoding to be aborted when a different color than configured is found. This is done to provide better power efficiency and to allow for transmission when the detected power is less than the threshold described in Sect. 3.2.

By writing to the registers `ACPHY_HESigBMyBSS0` and `ACPHY_HESigBMyBSS1`, we can configure what color is being filtered. The color filtering is enabled or disabled by writing a value in the register `ACPHY_HESigParseCtrl`. Specifically, the field to be written is `he_mybss_color_len`. The name explains why there are two different registers containing the color: we can choose the length of the BSS Color vector to be used for matching the color of a packet. This allows the STA to switch colors when roaming without interruption of the incoming traffic.

We can also write a length of zero to disable the color matching filter and receive data from every color. This is useful for sniffing, when the color of the BSS is not known, but it may introduce unwanted traffic that would be later filtered out in an phase of analysis.

The BSS color is available in the Wi-Fi beacon, so it can be configured without difficulty before the sniffing activity. The only downside is that if the color changes, the sniffing will stop. As this is a very situation-dependent configuration, its setup is left available for the user, who can

decide when BSS coloring is needed.

The second set of registers that need modification in order to grant monitoring features are `ACPHY_HESigBStaIdxy`. Here too there are two sets of registers (x equals 0 or 1) and each set contains four different configurable AID (y from 0 to 3). The purpose of these two sets is not clear, since as only the last written bank is effectively enabled.

One strange hardware organization consists in the presence of two equal BSS Color and AID selection register banks. These two banks seem not to control the hardware at the same time, but they seem to be selectable. We suspect this feature is used when roaming between access points. Having two register banks allows configuring the new AID and color while still being connected to the original AP. Then, when the device is required to switch to a new BPSK, the register bank is switched to allow communications to continue without interruption [7].

7.2 – Testing the hardware

In order to support the assumptions of Sect. 7.1, a set of controlled experiments has been run. These experiments utilized the layout reported by Fig. 7.1, where a transmitting Software Defined Radio (SDR) directly connected with 30 dB attenuation to the chipset radio interface. This allowed us to perform reception tests without experiencing signal degradation and interference.

Once the setup has been validated by the transmission of a single user frame generated with the help of MATLAB, a sequence of test frames have then been transmitted to evaluate hardware behavior when HE multi-user frames are received. Particularly, we tested multiple configurations of the AID and BSS color registers while capturing the memory locations respon-

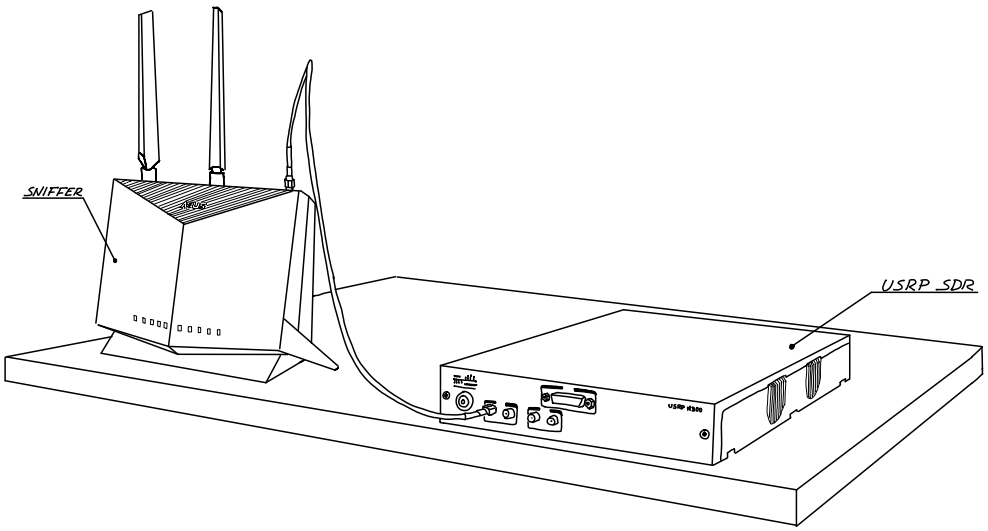


Figure 7.1: SDR reception test configuration. All receiving chains except for the connected one were disabled on the sniffer. Cable connection was preferred because of transmissing power limitations.

sible for decoding the incoming frame.

7.2.1 – AID Register settings

First, experiments were conducted in order to understand how the AID registers were determining how the frame was being decoded.

After having configured the decoding hardware to utilize the register bank 0 with adequate enabling bits in the selection register, we started transmitting an HE_MU frame composed as shown in Fig. 7.2. In Tab. 7.1 we find the outcome of the receptions given the configured AID decoding configuration. Table 7.1 allows to understand that the AID selection registers are used like a priority queue for frame decoding, where the higher index of the list is the most prioritized.

It can be observed that the hardware will receive the packets destined to the first AID found starting from index 03. If an AID is not found then

Table 7.1: Experiment Results for AID configuration registers.

ACPHY_HESigBStaId				**Result**
00	**01**	**02**	**03**	
255	255	255	255	No decoding happening. The AID 255 was not present in the frame, so we expected to not receive any frame.
1	255	255	255	Decoding of frame with AID 1
255	1	255	255	
255	255	1	255	
255	255	255	1	
2	255	255	255	Decoding of frame with AID 2
255	2	255	255	
255	255	2	255	
255	255	255	2	
3	255	255	255	Decoding of frame with AID 3
255	3	255	255	
255	255	3	255	
255	255	255	3	
0	1	255	255	Decoding of frame with AID 1
1	0	255	255	Decoding of frame with AID 0
0	1	2	3	Decoding of frame with AID 3
3	0	2	1	Decoding of frame with AID 1

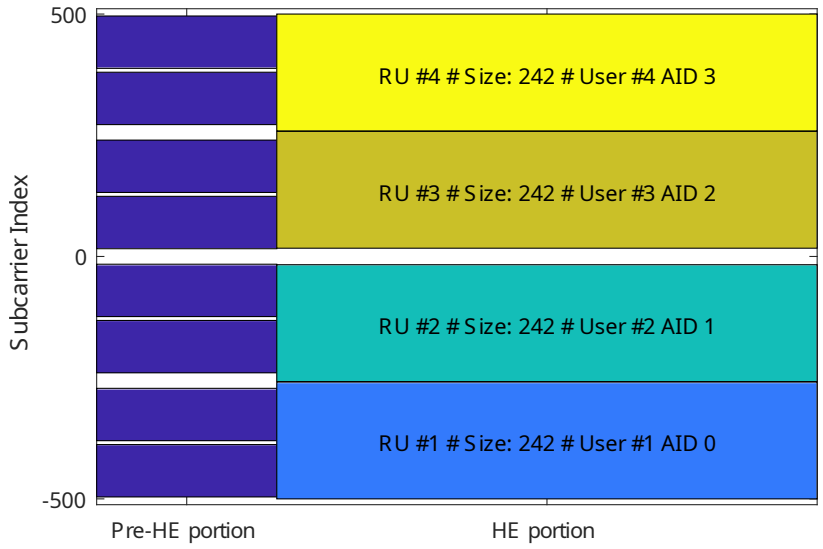


Figure 7.2: RU allocation of the transmitting frame.

the hardware looks for the next one decreasing the index reference.

7.2.2 – Non-Standard Frames

Another experiment was run in order to understand the hardware behavior when non-standard frames were received.

Detailing this frame format, Fig. 7.3 shows an allocation where both RU 1 and RU 2 have been generated with the same AID in the HE-SIG-B but containing different payloads in order to be able to distinguish the allocations.

The outcome of the experiment was that only the first intercepted AID, which was the first RU index available, was being decoded.

7.2.3 – Decoding data

By capturing the hardware registers right after the PLCP detection, we found that the hardware will discard any unmatched HE-SIG-B whose AID

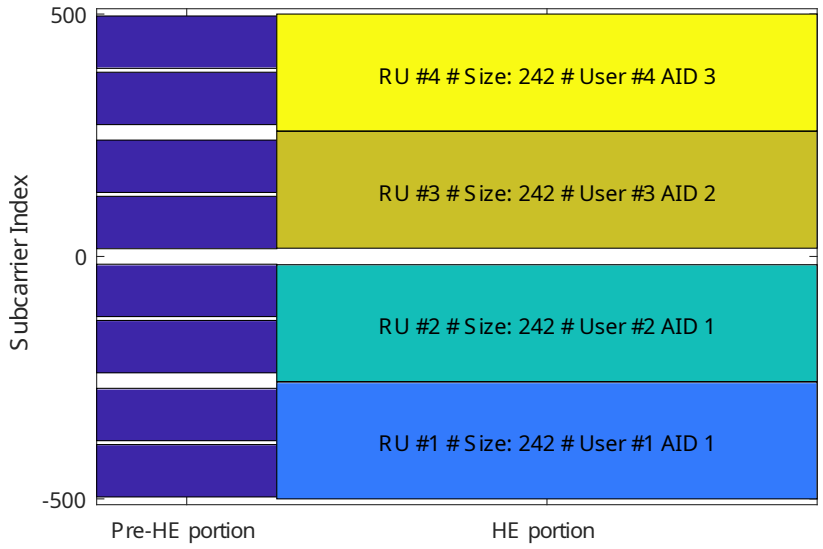


Figure 7.3: RU allocation of the non-standard transmitting frame.

was not in the matching register, as shown by the dumps in Listing 7.1 and Listing 7.2.

The shown content, starting at the address `0x10`, represents the dump of the PLCP registers responsible for matching the various parts of the PLCP data structure while receiving in order to prepare the reception flow based on the type of frame.

The data format of the memory dumps is not fully understood, but with an adequate number of tries, it was possible to deduce the following: knowing that each field is the representation (little endian) of a D11 register, the bytes at the addresses `0x1C` through `0x1F` are the HE-SIG-B. Particularly, the lower 16 bits of the HE-SIG-B are at address 0x1C:1D and the upper 5 bits are encoded in the 0x1D:1F addresses. For example, the SIG-B in Listing 7.1, should look like `00 11 00 03`, consisting in an allocation for AID 3 with 1 spatial stream, no beamforming, LDPC coding and MCS of

```
1  0x00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2  0x10: 0e 00 0b 39 01 01 e0 02 0d 01 82 14 03 00 11 00
3  0x20: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Listing 7.1: Received PLCP memory dump with HE-SIG-B decoded.

```
1  0x00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2  0x10: 0e 00 0b 24 01 01 e0 02 8d 00 82 2c 00 00 00 00
3  0x20: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Listing 7.2: Received PLCP memory dump without HE-SIG-B decoded.

1. Given the fact the experiment was conducted with one transmitting and receiving antenna while using the SDR radio, the decoding corresponds to the real transmission configuration.

It is evident that, in the case of Listing 7.1, where the hardware was configured with the AID equal to 3 (0x03) that the SIG-B has been correctly decoded. Contrary, in Listing 7.2, where the AID was misconfigured on purpose to 255 (0x0f), the SIG-B has not made available by the hardware.

We are now sure the hardware can decode only one HE-SIG-B at a time, and its value will be accessible in the packet header structure utilize by the D11 firmware. Other unrelated SIG-Bs are discarded and are not available in the registers for elaboration.

7.3 – Performance Evaluation

Recreating the experiment in Sect. 5.1.1.2, we focused on the Broadcom sniffer evaluation. The setup saw the addition of the sniffer close to the two STAs, as shown in Fig. 7.4, to make the reception conditions similar to the one at the Intel AX210 board. In this case, We developed a modified firmware in order to be able to keep track of the number of received frames by their types.

In particular, we added a small filtering and counting section in the D11

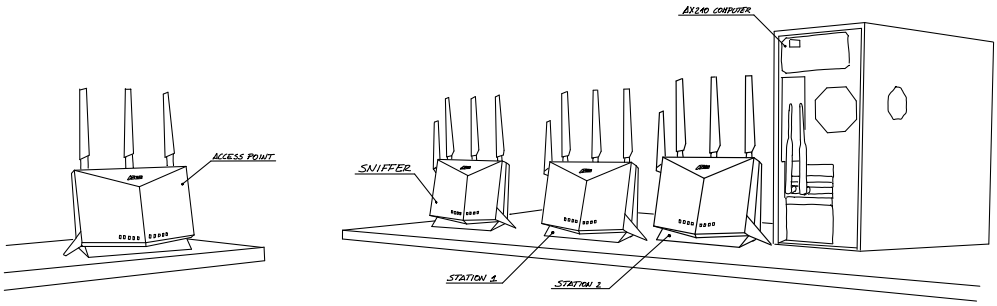


Figure 7.4: Physical location of the hardware in the experiment room. No people were inside the room at the time of the experiment.

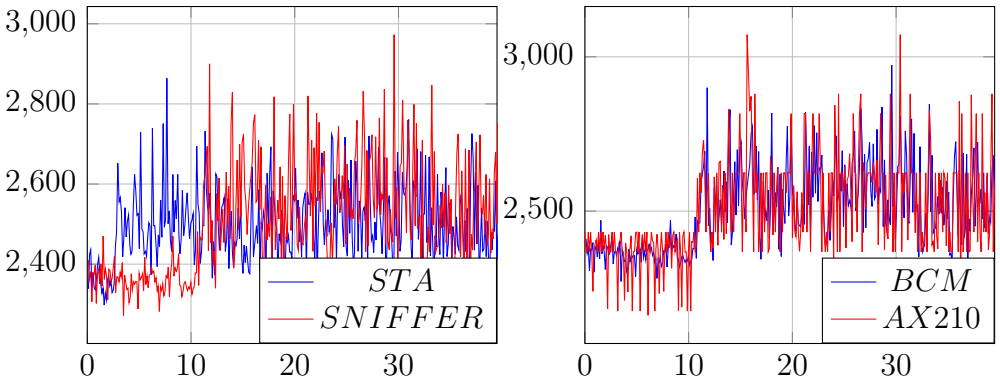
firmware for both the two STAs and the sniffer, right after the reception routine. The code, shown in Listing 7.3, makes easier to understand the process behind this feature: after finishing the reception of a frame we check its receiving address against a configured one in the shared memory, specified by three 16-bit values named `TARGET_MAC1_1`, `TARGET_MAC1_2`, `TARGET_MAC1_3`. Then, since we are intrested in monitoring the *iperf* traffic, we continue counting only if the frame control field of the frame indicates the presence of a data frame. At this point the frame type is encoded in the fourth and fifth bits of `r23` which are shifted and masked out and checked against the values 0 and 1, representing HE_SU and HE_MU frame types respectively and consequently we increment the relative counter.

Experiment results demonstrate that the number of frames received by the sniffing apparatus, configured with the correct BSS Color and the same AID configured for the AX210 sniffer, are very close to each other. In fact, during this experiment, the counter values have been monitored at a constant interval of 0.1s, plotted on the graphs in Fig. 7.5a and Fig. 7.5a.

In Fig. 7.5a the number of received Multi User frames per time interval is plotted. Instead, in figure Fig. 7.5b a comparison between the number of

```
1  L965_rx_complete:
2      [...] ; perform reception routines as FCS checks and
           push the packet in the RXFIFO when able.
3
4      ; filter frame by RA field against memory locations.
5      xjne    TARGET_MAC1_1, SPR_MHP_Addr1_L_ID, done+
6      xjne    TARGET_MAC1_2, SPR_MHP_Addr1_M_ID, done+
7      xjne    TARGET_MAC1_3, SPR_MHP_Addr1_H_ID, done+
8
9      ; filter only data frames
10     and     SPR_MHP_FC_ID, 0xff, SPARE1
11     xjne    SPARE1, 0x88, done+
12
13     ; find out the frame type ((r23 >> 3) & 3)
14     mov     r23, SPARE1
15     sr      SPARE1, 3, SPARE1
16     and     SPARE1, 0x3, SPARE1
17     xjne    SPARE1, 0, next+    ; jump if not Single User
18
19     ; Increment SU Counter (32 bits)
20     xadd.   CONTA1_SU_L, 1, CONTA1_SU_L
21     xaddc   CONTA1_SU_H, 0, CONTA1_SU_H
22     jmp     done+
23 next:
24     ; Increment MU Counter (32 bits)
25     xjne    SPARE1, 1, next+    ; jump if not Multi User
26     xadd.   CONTA1_MU_L, 1, CONTA1_MU_L
27     xaddc   CONTA1_MU_H, 0, CONTA1_MU_H
28     jmp     done+
29 next:
30     ; If the frame type is something else (e.g. Trigger
           Based) we would end up here.
31     ; note: given the filtering of only data frames, we
           should never get here.
32     xadd.   CONTA1_OTHER_L, 1, CONTA1_OTHER_L
33     xaddc   CONTA1_OTHER_H, 0, CONTA1_OTHER_H
34 done:
35     [...] ; continue execution
```

Listing 7.3: Counting routine in the D11 Core firmware for performance evaluation.



(a). STA received frames in time vs. Broadcom sniffed frame count

(b). Intel sniffed frames in time vs. Broadcom sniffed frame count

Figure 7.5: Experiment 1 results for Intel and Broadcom single AID sniffing performances. MCS 6, NSS 2.

frames received by the Broadcom sniffer against the Intel AX210 is made.

It is possible to note that the numbers are very close to each other. In particular the fact the Intel and Broadcom sniffers present practically the same numbers, albeit Intel reports an increased variance in them, positively concludes the performance evaluation of the platform.

In order to force the traffic to be sent as OFDMA, the AP has been set with a fixed MCS and Number of Spatial Streams (NSS) to not incur into traffic generation problems from the user space application.

The same experiment has been reproduced with four spatial streams, confirming the results in Sect. 5.1.1.2, but this time the Broadcom sniffer was able to capture almost every frame, as shown by the comparison in Fig. 7.6

7.4 – Tool Implementation

We discussed in Sect. 5.1.2 the limitations of the AX210 sniffing setup. The main deficit this work aims to improve is the frequency of the receiving AID

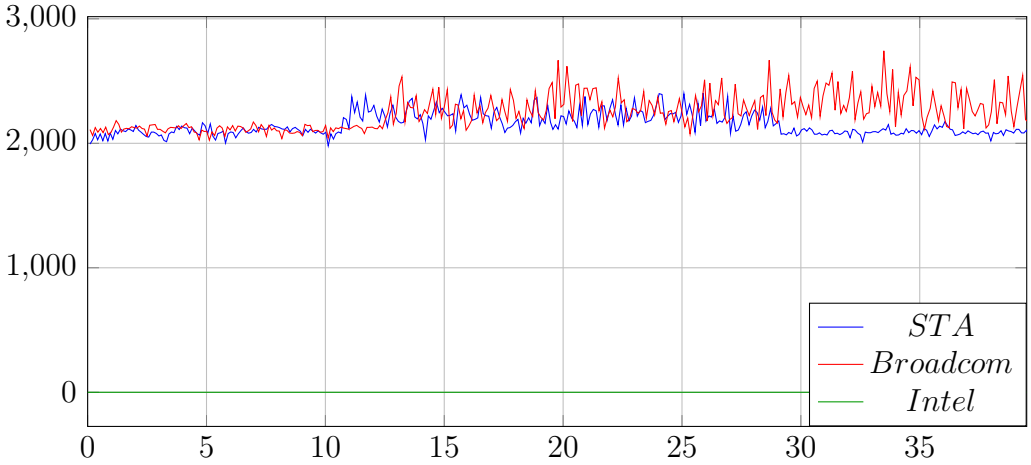


Figure 7.6: Experiment 1 results for Intel and Broadcom single AID sniffing performances against the real values from the STA counters. MCS 3, NSS 4.

switching.

Since the hardware is not allowing nor the full access of the PLCP header and neither multiple AID decoding in a single multi-user frame, our solution to improve sniffing results is to instruct the hardware to switch the listening AID every frame reception.

The implemented sniffing operation, then, follows Algorithm 1, given the configuration parameters that have been set in the D11 shared memory from the host.

The expected shared memory configuration for this modified firmware must contain a table of scanning AID at a predefined address. The table format shall be a length-encoded list of 16 bit values starting at address `0x176c` as described by Fig. 7.7. This list must be preceded by the control byte of the system, containing the activation flag alongside the current list index.

Each time the function for switching the AID is called, the next aid in the list is set in the configuration registers and the `INDEX` field is updated.

Algorithm 1 Sniffing procedure

```
while running do
    recv ← waitframe
    if ¬is_mu(recv) ∨ ¬switching_en then
        PushFrame(recv)
        continue
    end if
    NextAID()                                  ▷ Frame is Multi User
    if sig_b is null then
        continue
    end if
    PushFrame(null)                            ▷ Selected AID present
end while
```

In order to prevent overreading the list, the switching procedure will perform a modulo operation on the index field with the length of the list to automatically roll back to the beginning if the index is pointing over the end of the list.

Thanks to Nexmon utilities, the user space interface for writing into the shared memory is already implemented. Only a small script was written for easiness of use of the system. The script takes the parameters of the AID list from the command line and encodes them in the correct format. Once the data has been formatted, an *ioctl* is performed in order to write the data on the D11 memory [44].

By performing AID switching at every frame, we ensure that the system will not get stuck in case an AID is not transmitted or missing.

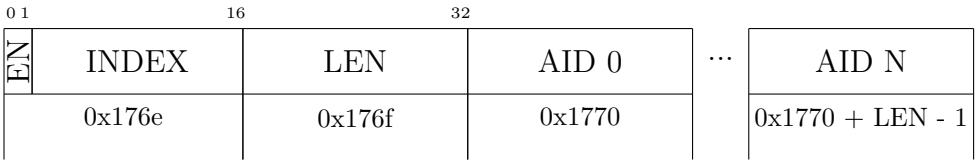


Figure 7.7: Memory representation for the control structure controlling AID switching.

Sadly, because of Sect. 7.1 discoveries, no data regarding the RU allocation can be decoded and interpreted given the lack of HE-SIG-B data in the header available to the firmware.

Another time-based switching mode has been implemented to permit direct confrontation with the banchmarks provided in Sect. 5.1. This mode utilizes the same methodology of Algorithm 1 but, instead of switching at each frame, the change in aid occurs when the D11's internal monotonic counter gets to the time delta configured. The algorithm representing this working mode is defined in Algorithm 2

Algorithm 2 Sniffing procedure

```
while running do
    recv ← waitframe
    if ¬is_mu(recv) ∨ ¬switching_en then
        PushFrame(recv)
        continue
    end if                                   ▷ Frame is Multi User
    if last_time + Δt ≤ GetTime() then
        NextAID()
        last_time ← GetTime()
    end if
    if sig_b is null then
        continue
    end if
    PushFrame(null)                          ▷ Selected AID present
end while
```

7.5 – Results

7.5.1 – Experiment #1

Once the above-mentioned implementation has been inserted into the binary payload the unix driver loads at boot, a new SDR based experiment

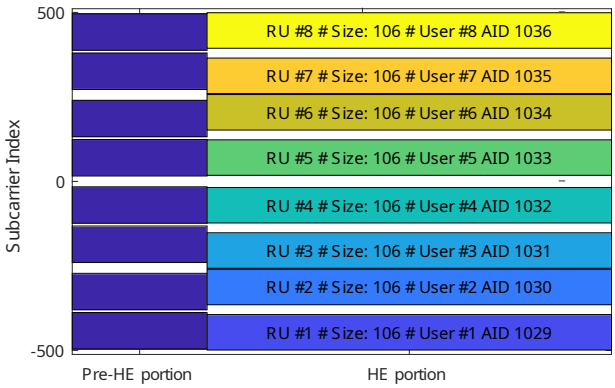


Figure 7.8: RU allocation of the testing frame in transmission.

has been run to check it effectivity.

The OFDMA frame utilized for this experiment, shown in Fig. 7.8, consisted in an eight-user wide tranmission, with AMPDU payload of six packets.

The hardware configuration was the following:

- ENABLE bit: 1
- Staring Index: 0
- List Length: 8
- AIDs: 1029, 1030, 1031, 1032, 1033, 1034, 1035, 1036

Note that in this frame, the presence of AMPDU will introduce in the resulting packet capture multiple packets with the same AID for each received frame. Also, the packets for AID 1032 were sent at MCS 3 instead of MCS 0 like the others. Because of DL-OFDMA frame padding, this sub-frames contain a repetition of packets with the tag `EOT` set to the value 1 [7].

Table 7.2: Receiving MAC address per AID in Experiment #1

AID	MAC Address
1029	24 4b fe 32 11 11
1030	24 4b fe 32 22 22
1031	24 4b fe 32 33 33
1032	24 4b fe 32 44 44
1033	24 4b fe 32 55 55
1034	24 4b fe 32 66 66
1035	24 4b fe 32 77 77
1036	24 4b fe 32 88 88

This removal of the padding is performed in the ARM core, which has been disabled in this execution because the monitor mode will directly send up to the user space any packet received.

Full results of this capture are available in App. B, where it can be observed a cyclic pattern in packet reception, due to the alternation of the AID registers.

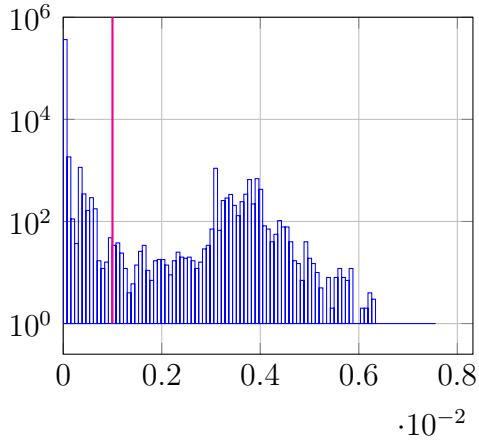
All the packets in each sub-frame were created such that their receiving address (last six bytes of the payload) were reflecting the index of the RU. Table 7.2 shows their association, which may help in decoding data in Tab. B.1.

7.5.2 – Experiment #2

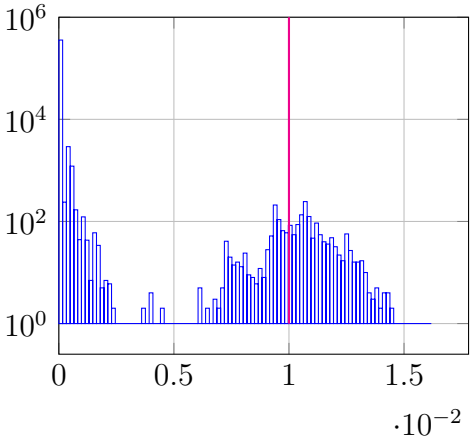
The same experiment previously run in Sect. 5.1.1.3 has been performed with the setup modifications indicated in Sect. 7.5.2 but maintaining the same STA configurations. We found the number of recorded frames to closely match what previously benchmarked.

The sniffer was placed in time-switching mode in order to be compared to the Intel AX210.

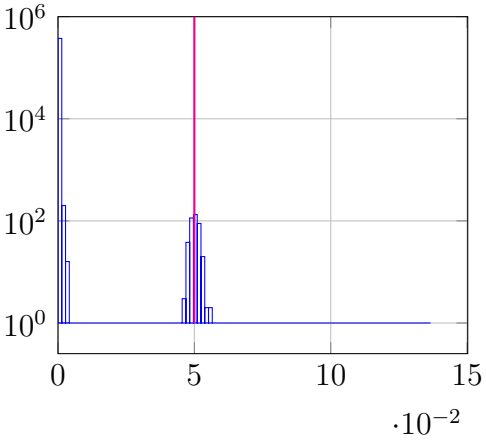
Repeating the experiment analysis, whose graphs are reported in



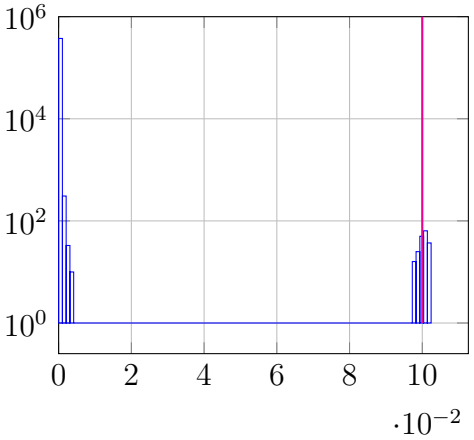
(a). Inter frame interval distribution, $\Delta_{switch} = 0.001s$



(b). Inter frame interval distribution, $\Delta_{switch} = 0.01s$



(c). Inter frame interval distribution, $\Delta_{switch} = 0.05s$



(d). Inter frame interval distribution, $\Delta_{switch} = 0.1s$

Figure 7.9: Inter-Frame time distruibution per switching time interval. Broadcom Sniffer.

Table 7.3: Setup 1, Experiment 3, Results.

Metric	Value per Δ_{switch}			
	0.001s	**0.01s**	**0.05s**	**0.10ms**
# of packets transferred	2330490	2358323	2260490	2206048
# of packet for AID1	1150172	1155346	1057491	1125952
# of packet for AID2	1180316	1202975	1202997	1080094
# of packet sniffed	742250	727628	740587	745918
# of packet sniffed for AID1	366454	363418	364356	370585
# of packet sniffed for AID2	366454	363418	364356	370585

Fig. 7.9, we can observe an improved jitter and switching time precision at even 1ms of switching time. In fact, it is possible to observe a well-defined peak of inter-frame times in the correspondence of the switching period, conferming the better jitter resistance of the system.

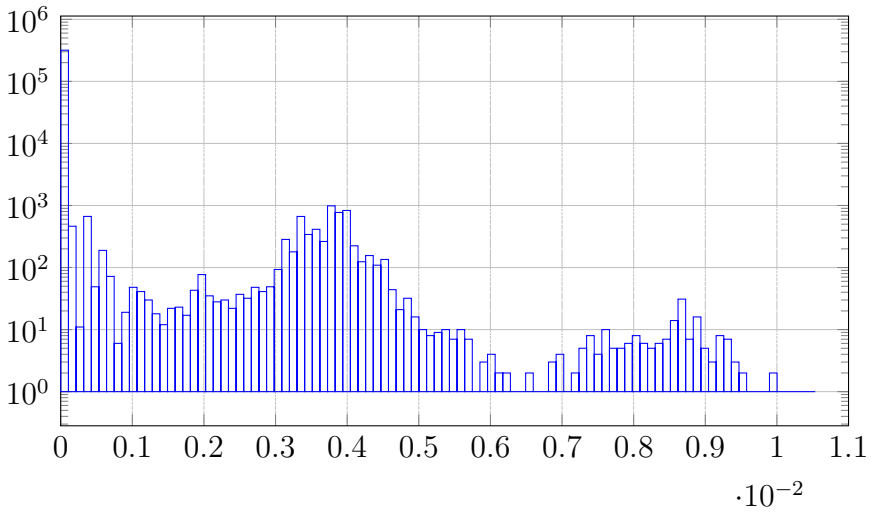
Also, capture statistics of the experiments are represented in Tab. 7.3.

From Tab. 7.3, it is evident that the two systems reach comparable performances in terms of sniffing capabilities. The Broadcom based tool developed while working on this topic, though, highlights better jitter if used in time-switch mode other than allowing for up to four spatial streams communications to be captured.

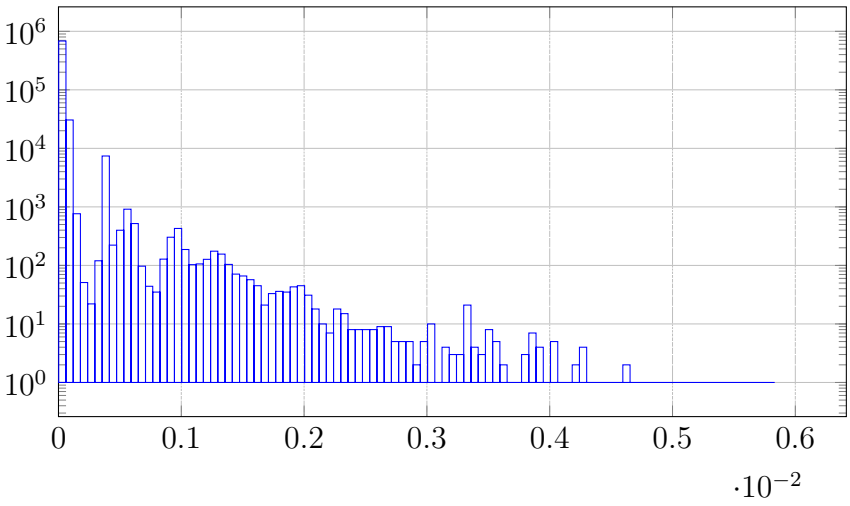
As a final note, an additional step during this experiment has been taken. In order to provide an estimate on the fidelity in representing network behavior, the sniffer was placed in frame-switching mode, utilizing the Algorithm 1.

Fig. 7.10 shows the similarity in inter-frame interval distribution between a capture made with a single recording AID and a capture where the sniffer was configured for switching each frame between two AIDs.

It is possible to observe a similar trend between the two distribution, where the one in Fig. 7.10a shows a slight increase of occurrences of intervals bigger than 5ms.



(a). Frame swithcing mode inter-frame intervals



(b). Single AID reference capture

Figure 7.10: Inter-frame timing behavior with and without switching.

Considering that, with the AP configuration used for this experiment, the maximum frame duration was calculated to be 3ms, the two distribution are closely related. This allows us to state that the system could approximate the network traffic behavior correctly with a resolution in the order of 5ms intervals.

7.6 – AID Discovery

When analyzing a real network with both DL-OFDMA and UL-OFDMA, it is possible to recover present AID informations.

Given the procedure utilized by 802.11ax compatible AP to retrieve data in uplink OFDMA, discussed in Fig. 3.1b, it is possible to gather information about the AID in the network by decoding the trigger frames.

By decoding the trigger frames set by the AP, the sniffer can gather information about the AID in the network and insert them in the sniffing list. Because, usually, the network's connected clients are not changing rapidly, after three to five trigger frames we could be reasonably sure the AID in the network have been already seen. Then, the trigger frame and list update can be kept working while sniffing in order to catch network layout evolution, where AID gets disconnected or added.

Additionally, a timeout can be associated to each AID in order to prune the list of inactive members. In this case, an AID gets discarded whenever no trigger frames reference it for a given period of time or no frames gets received in that interval.

The probability of discarding an active AID, because of the per-frame cycling, tends to zero; given that multi-user traffic happens when multiple data streams are sent to the network. This makes sure that, granted a long enough period of time allowing for a couple of frames per selected AID to

be sent, which usually may not extend over ten milliseconds, no active AID will be discarded by the system.

Updating the list of switching AID needs only a shared memory update increasing or decreasing the index the D11 uses for sniffing, and a small reordering of the list in order to not leave invalid aids in the structure.

For example, adding an AID to the list may simply insert the value at the end of the list and increment its length by one. For removal, the length of the list shall be reduced and the last element (if it is not the one to be eliminated) should overwrite the element being removed. The index of the switching will be updated automatically by the next switching event because the increment with modulus is executed before the change in the registers.

8 — Conclusions and future work

This work has successfully demonstrated the potential to develop a more comprehensive sniffing tool compared to existing solutions. Considering the importance of network sniffing within research environments, e.g. for the development of advanced scheduling algorithms, this enhanced set of tools enables users to observe network activities and derive statistically meaningful parameters for their analysis.

We showcased the capability of the Broadcom sniffer in performing monitoring activities, even in scenarios where other consumer-grade hardware had proven limited. One of the BCM46384's strength lies in its four-antenna configuration, which allows the sniffing of up to four spatial stream communications, achieving a theoretical monitoring throughput exceeding 8Gbps.

Moreover, we developed and tested a novel capturing method for OFDMA multi-user frames, facilitating network research under conditions where high network load is induced. By leveraging the ability to switch AID with each incoming frame directly in the real time processor, we successfully captured multi-user traffic with minimal deviation from real network behavior, as demonstrated in Sect. 7.5.2. Although this method does not capture all the transmitted packets, it enables the detection and understanding of behavioral changes within the network, such as bursts or other peculiar occurrences.

This work represents a significant advancement in the tools available for wireless network analysis by enhancing the instrument's capabilities. As these modifications are based on Nexmon, the reverse engineering and patching tool for Broadcom-based chipsets, the firmware developed here can

be ported to other devices with additional capabilities, thereby further expanding the range of available options. Additionally, Nexmon's open-source distribution ensures unrestricted access for researchers, hence maximizing the potential for new discoveries in the field.

The development of these tools was not without its challenges. As it is common in reverse engineering efforts, considerable energy and time were required to understand the hardware and existing firmware, particularly in the absence of comprehensive documentation. The work presented in this thesis only skims the surface of the effort required to bring this tool to fruition.

Unfortunately, the complete capture of multi-user frames could not be implemented due to hardware limitations. Future research will focus on overcoming these limitations and further enhancing the tools developed in this work.

Finally, a thorough and comprehensive statistical validation of the sniffing methods presented here should be run, which would further broaden the scope for research in this critical area.

Bibliography

[1] R. Mubashar, M. A. Siddique, A. Rehman, A. Asad, and A. Rasool, "Comparative performance analysis of short-range wireless protocols for wireless personal area network", *Iran Journal of Computer Science*, vol. 4, Sep. 2021.

[2] A. Halbouni, L.-Y. Ong, and M.-C. Leow, "Wireless Security Protocols WPA3: A Systematic Literature Review", *IEEE Access*, vol. 11, pp. 112 438–112 450, 2023.

[3] P. Machań and J. Woźniak, "On the fast BSS transition algorithms in the IEEE 802.11 r local area wireless networks P Machań, J Wozniak Telecommunication Systems, 1-8", *Telecommunication Systems*, Jan. 2011.

[4] "IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)*, pp. 1–1076, 2007.

[5] N. Gupta and R. Gupta, "Routing protocols in Mobile Ad-Hoc Networks: An overview", pp. 173–177, 2010.

[6] K. Haseeb, I. Ud Din, N. Islam, A. Altameem, and A. Almogren, "RTS: A Robust and Trusted Scheme for IoT- based Mobile Wireless Mesh Networks", *IEEE Access*, vol. 8, pp. 1–10, Apr. 2020.

[7] "IEEE Standard for Information Technology–Telecommunications and Information Exchange between Systems Local and Metropolitan Area Networks–Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 1: Enhancements for High-Efficiency WLAN", *IEEE Std 802.11ax-2021 (Amendment to IEEE Std 802.11-2020)*, pp. 1–767, 2021.

[8] Y. Shen, R. Bootsman, M. S. Alavi, and L. de Vreede, "A 0.5-3 GHz I/Q Interleaved Direct-Digital RF Modulator with up to 320 MHz Modulation Bandwidth in 40 nm CMOS", in *2020 IEEE Custom Integrated Circuits Conference (CICC)*, 2020, pp. 1–4.

[9] G. Agrawal, P. Sinha, J. Dhar, C. Rao, and R. Jyoti, "Design and Development of Broadband and Compact Size IQ demodulator at 850±112.5 MHz", in *2021 6th International Conference for Convergence in Technology (I2CT)*, 2021, pp. 1–4.

[10] B. Hirosaki, "An Analysis of Automatic Equalizers for Orthogonally Multiplexed QAM Systems", *IEEE Transactions on Communications*, vol. 28, no. 1, pp. 73–83, 1980.

[11] K. Vaigandla, S. R. Allanki, K. Srikanth, Study, and E. Ijmtst, "Study of Modulation Schemes over a Multipath Fading Channels", *International Journal for Modern Trends in Science and Technology*, vol. 7, pp. 34–39, Oct. 2021.

[12] X. Jiao, W. Liu, M. Mehari, M. Aslam, and I. Moerman, "openwifi: a free and open-source IEEE802.11 SDR implementation on SoC", in *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, 2020, pp. 1–2.

[13] F. Gringoli, M. Cominelli, A. Blanco, and J. Widmer, "AX-CSI: Enabling CSI Extraction on Commercial 802.11ax Wi-Fi Platforms", in *Proceedings of the 15th ACM Workshop on Wireless Network Testbeds, Experimental Evaluation & CHaracterization*, ser. WiNTECH '21, New Orleans, LA, USA: Association for Computing Machinery, 2021, pp. 46–53. [Online]. Available: https://doi.org/10.1145/3477086.3480833.

[14] F. Gringoli, M. Schulz, J. Link, and M. Hollick, "Free Your CSI: A Channel State Information Extraction Platform For Modern Wi-Fi Chipsets", in *Proceedings of the 13th International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization*, ser. WiNTECH '19, Los Cabos, Mexico: Association for Computing Machinery, 2019, pp. 21–28. [Online]. Available: https://doi.org/10.1145/3349623.3355477.

[15] R. Laufer and L. Kleinrock, "The Capacity of Wireless CSMA/CA Networks", *IEEE/ACM Transactions on Networking*, vol. 24, no. 3, pp. 1518–1532, 2016.

[16] E. Ziouva and T. Antonakopoulos, "CSMA/CA performance under high traffic conditions: throughput and delay analysis", *Comput. Commun.*, vol. 25, no. 3, pp. 313–321, Feb. 2002. [Online]. Available: https://doi.org/10.1016/S0140-3664(01)00369-3.

[17] P. Chatzimisios, A. Boucouvalas, and V. Vitsas, "Effectiveness of RTS/CTS handshake in IEEE 802.11a Wireless LANs", *Electronics Letters*, vol. 40, pp. 915–916, Aug. 2004.

[18] Y.-J. Cheng, "The impact of RTS/CTS exchange on the performance of multi-rate IEEE 802.11 wireless networks", Ph.D. dissertation, Indiana University South Bend, 2008.

[19] J. Kim and I. Lee, "802.11 WLAN: history and new enabling MIMO techniques for next generation standards", *IEEE Communications Magazine*, vol. 53, no. 3, pp. 134–140, 2015.

[20] B. Bellalta, J. Barcelo, D. Staehle, A. Vinel, and M. Oliver, "On the Performance of Packet Aggregation in IEEE 802.11ac MU-MIMO WLANs", 2012. arXiv: `1204.0643 [cs.NI]`. [Online]. Available: `https://arxiv.org/abs/1204.0643`.

[21] O. Bejarano, E. W. Knightly, and M. Park, "IEEE 802.11ac: from channelization to multi-user MIMO", *IEEE Communications Magazine*, vol. 51, no. 10, pp. 84–90, 2013.

[22] M. S. Gast, *802.11ac: A Survival Guide*. O'Reilly Media, Inc., 2013.

[23] "IEEE Standard for Information technology– Telecommunications and information exchange between systemsLocal and metropolitan area networks– Specific requirements–Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications–Amendment 4: Enhancements for Very High Throughput for Operation in Bands below 6 GHz.", *IEEE Std 802.11ac-2013 (Amendment to IEEE Std 802.11-2012, as amended by IEEE Std 802.11ae-2012, IEEE Std 802.11aa-2012, and IEEE Std 802.11ad-2012)*, pp. 1–425, 2013.

[24] F. Siddiqui, S. Zeadally, and K. Salah, "Gigabit Wireless Networking with IEEE 802.11ac:Technical Overview and Challenges", *Journal of Networks*, vol. 10, Apr. 2015.

[25] T. Hwang, C. Yang, G. Wu, S. Li, and G. Ye Li, "OFDM and Its Wireless Applications: A Survey", *IEEE Transactions on Vehicular Technology*, vol. 58, no. 4, pp. 1673–1694, 2009.

[26] E. Khorov, A. Kiryanov, A. Lyakhov, and G. Bianchi, "A Tutorial on IEEE 802.11ax High Efficiency WLANs", *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 197–216, 2019.

[27] R. Goodwins, *Next-generation 802.11 ax wi-fi: Dense, fast, delayed*, visited: 28/08/2024, 2021. [Online]. Available: `https://www.zdnet.com/home-and-office/networking/next-generation-802-11ax-wi-fi-dense-fast-delayed/`.

[28] S. Avallone, P. Imputato, G. Redieteab, C. Ghosh, and S. Roy, "Will OFDMA Improve the Performance of 802.11 Wifi Networks?", *IEEE Wireless Communications*, vol. 28, no. 3, pp. 100–107, 2021.

[29] E. Perahia and R. Stacey, *Next Generation Wireless LANs: 802.11n and 802.11ac*, 2nd ed. Cambridge University Press, 2013.

[30] M. Natkaniec and N. Bieryt, "An Analysis of the Mixed IEEE 802.11ax Wireless Networks in the 5 GHz Band", *Sensors*, vol. 23, p. 4964, May 2023.

[31] F. Frommel, G. Capdehourat, and B. Rodríguez, "Performance Analysis of Wi-Fi Networks based on IEEE 802.11ax and the Coexistence with Legacy IEEE 802.11n Standard", pp. 492–495, Nov. 2021.

[32] P. Imputato, S. Avallone, M. Smith, D. Nunez, and B. Bellalta, "Beyond Wi-Fi 7: Spatial reuse through multi-AP coordination", *Comput. Netw.*, vol. 239, no. C, Apr. 2024. [Online]. Available: `https://doi.org/10.1016/j.comnet.2023.110160`.

[33] K. Wang and K. Psounis, "Efficient scheduling and resource allocation in 802.11ax multi-user transmissions", *Computer Communications*, vol. 152, Feb. 2020.

[34] D. Magrin, S. Avallone, S. Roy, and M. Zorzi, "Performance Evaluation of 802.11ax OFDMA Through Theoretical Analysis and Simulations", *IEEE Transactions on Wireless Communications*, vol. 22, no. 8, pp. 5070–5083, 2023.

[35] R. Karthik and S. Palaniswamy, "Resource Unit (RU) based OFDMA Scheduling in IEEE 802.11ax system", in *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2018, pp. 1297–1302.

[36] A. A. E. Boukebous, M. I. Fettache, G. Bendiab, and S. Shiaeles, "A Comparative Analysis of Snort 3 and Suricata", in *2023 IEEE IAS Global Conference on Emerging Technologies (GlobConET)*, 2023, pp. 1–6.

[37] R. Singh and S. Kumar, "A Comparative Study of Various Wireless Network Monitoring Tools", in *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*, 2018, pp. 379–384.

[38] WiGLE, *WiGLE: Wireless Network Mapping*, `https://wigle.net/` [Accessed: 18/08/2024], 2001.

[39] Council of European Union, *Council regulation (EU) no 679/2016*, 2016.

[40] Ministero della Giustizia, *Accesso abusivo a un sistema informatico o telematico, Art 615 ter c.p.*

[41] G. Harris, Ed. and M. Richardson, "PCAP Capture File Format", RFC Editor, RFC, Dec. 2021. [Online]. Available: https://www.ietf.org/archive/id/draft-gharris-opsawg-pcap-01.html.

[42] Intel, *iwlwifi*. [Online]. Available: https://github.dev/torvalds/linux/blob/master/drivers/net/wireless/intel/iwlwifi/.

[43] N. Iooss and G. Campana, *Ghost in the Wireless, iwlwifi edition*, 2022.

[44] M. Schulz, D. Wegemer, and M. Hollick, *Nexmon: The C-based Firmware Patching Framework*, version 2.2.2, Sep. 2017. [Online]. Available: https://github.com/seemoo-lab/nexmon.

[45] M. Schulz, "Teaching Your Wireless Card New Tricks: Smartphone Performance and Security Enhancements Through Wi-Fi Firmware Modifications", en, Ph.D. dissertation, Technische Universität Darmstadt, Darmstadt, 2018. [Online]. Available: http://tubiblio.ulb.tu-darmstadt.de/105239/.

List of Figures

List of Tables

List of Acronyms

ACK Acknowledgement

ADC Analog to Digital Converter

AID Association Index

AMPDU Aggregated MAC Protocol Data Unit

AMSDU Aggregated MAC Service Data Unit

AP Access Point

ASK Amplitude Shift Keying

BB Base Band

BO Back-Off counter

BPSK Binary Phase Shift Keying

BSR Buffer State Request

BSS Basic Service Set

BSSID Basic Service Set Identifier

CCI Co-Channel Interference

CRC Cyclic Redundancy Check

CSI Channel State Information

CSMA-CA Carrier Sense Multiple Access with Collision Avoidance

CTS Clear to Send

CW Contention Window

DAC Digital to Analog Converter

DIFS Distributed Interframe Space

DL-OFDMA Downlink Orthogonal Frequency Division Multiple Access

DMA Direct Memory Access

FCS Frame Check Sequence

FFT Fast Fourier Transform

FIFO First Input First Output queue

GDPR Genral Data Protection Regulation

GPL GNU General Public License

HE High Efficiency

IDS Intrusion Detection System

IFFT Inverse Fast Fourier Transform

ISM Industrial, Scientific and Medical

LO Local Oscillator

LTF Long Training Field

MAC Medium Access Control

MCS Modulation Coding Scheme

MIMO Multiple Input Multiple Output

MU Multi User

MU-MIMO Multi User Multiple Input Multiple Output

NDP Null Data Packet

NDPA Null Data Packet Announcment

NSS Number of Spatial Streams

OFDM Orthogonal Frequency Division Multiplexing

OFDMA Orthogonal Frequency Division Multiple Access

PCAP Packet Capture Protocol

PCIe Peripheral Component Interconnect Express

PHY Physical Layer

PLCP Physical Layer Convergence Protocol

QAM Quadrature Amplitude Modulation

QBPSK Quadrature Binary Phase Shift Keying

RF Radio Frequency

RTS Request To Send

RU Resource Unit

SDR Software Defined Radio

SIFS Short Interframe Space

SNR Signal to Noise Ratio

SOF Start of Frame

STA Station

STF Short Training Field

SU Single User

TCP Transmission Control Protocol

UL-OFDMA Uplink Orthogonal Frequency Division Multiple Access

WPA3 Wi-Fi Protected Access version 3

A — AX210 monitor mode

In order to put the AX210 in monitor mode the following steps need to be taken.

```
1  export IFACE="wlp4s0"
2  export IFACE_PCI_ADDRESS="00:04.000"
3  export CHANNEL="157 80MHz"
4  export AID="c"
5  export DEVICE="00:00:00:00:00:00"
6
7  #Ensure the interface is down
8  ip link sed $IFACE down
9
10 #Setup monitor mode
11 iw dev $IFACE set monitor none
12
13 #Bring up the interface and confure the channel
14 ip link sed $IFACE up
15 iw dev $IFACE set channel $CHANNEL
16
17 #Configure an AID to listen at when receiving OFDMA
      frames
18 echo $AID $DEVICE | tee sys/kernel/debug/iwlwifi/
      $IFACE_PCI_ADDRESS/iwlmvm/he_sniffer_params
```

This snippet will configure the chipset for sniffing wifi traffic and, in the event of a OFDMA frame it will try to decode the content for the supplied AID. If a valid mac address is specified in the variable `DEVICE` the chipset will decode UL-OFDMA frames coming after a trigger frame if the corresponding AID has been identified.

B — Extended SDR experiment results

Table B.1: Experiment Results for AID configuration registers

No.	AID	MCS	Frame Counter	Payload
1	0x405	0	0x0000	88 02 00 00 24 4b fe 32 11 11 ...
2	0x405	0	0x1000	88 02 00 00 24 4b fe 32 11 11 ...
3	0x405	0	0x2000	88 02 00 00 24 4b fe 32 11 11 ...
4	0x405	0	0x3000	88 02 00 00 24 4b fe 32 11 11 ...
5	0x405	0	0x4000	88 02 00 00 24 4b fe 32 11 11 ...
6	0x405	0	0x5000	88 02 00 00 24 4b fe 32 11 11 ...
7	0x406	0	0x0000	88 02 00 00 24 4b fe 32 22 22 ...
8	0x406	0	0x1000	88 02 00 00 24 4b fe 32 22 22 ...
9	0x406	0	0x2000	88 02 00 00 24 4b fe 32 22 22 ...
10	0x406	0	0x3000	88 02 00 00 24 4b fe 32 22 22 ...
11	0x406	0	0x4000	88 02 00 00 24 4b fe 32 22 22 ...
12	0x406	0	0x5000	88 02 00 00 24 4b fe 32 22 22 ...
13	0x407	0	0x0000	88 02 00 00 24 4b fe 32 33 33 ...
14	0x407	0	0x1000	88 02 00 00 24 4b fe 32 33 33 ...
15	0x407	0	0x2000	88 02 00 00 24 4b fe 32 33 33 ...
16	0x407	0	0x3000	88 02 00 00 24 4b fe 32 33 33 ...
17	0x407	0	0x4000	88 02 00 00 24 4b fe 32 33 33 ...
18	0x407	0	0x5000	88 02 00 00 24 4b fe 32 33 33 ...
20	0x408	3	0x0000	88 02 00 00 24 4b fe 32 44 44 ...
21	0x408	3	0x1000	88 02 00 00 24 4b fe 32 44 44 ...

22	0x408	3	0x2000	88 02 00 00 24 4b fe 32 44 44 ...
23	0x408	3	0x3000	88 02 00 00 24 4b fe 32 44 44 ...
... 15 more ...				
39	0x408	3	0x1000	88 02 00 00 24 4b fe 32 44 44 ...
40	0x408	3	0x2000	88 02 00 00 24 4b fe 32 44 44 ...
41	0x408	3	0x3000	88 02 00 00 24 4b fe 32 44 44 ...
42	0x408	3	0x4000	88 02 00 00 24 4b fe 32 44 44 ...
43	0x409	0	0x0000	88 02 00 00 24 4b fe 32 55 55 ...
44	0x409	0	0x1000	88 02 00 00 24 4b fe 32 55 55 ...
45	0x409	0	0x2000	88 02 00 00 24 4b fe 32 55 55 ...
46	0x409	0	0x3000	88 02 00 00 24 4b fe 32 55 55 ...
47	0x409	0	0x4000	88 02 00 00 24 4b fe 32 55 55 ...
48	0x409	0	0x5000	88 02 00 00 24 4b fe 32 55 55 ...
49	0x40a	0	0x0000	88 02 00 00 24 4b fe 32 66 66 ...
50	0x40a	0	0x1000	88 02 00 00 24 4b fe 32 66 66 ...
51	0x40a	0	0x2000	88 02 00 00 24 4b fe 32 66 66 ...
52	0x40a	0	0x3000	88 02 00 00 24 4b fe 32 66 66 ...
53	0x40a	0	0x4000	88 02 00 00 24 4b fe 32 66 66 ...
54	0x40a	0	0x5000	88 02 00 00 24 4b fe 32 66 66 ...
55	0x40b	0	0x0000	88 02 00 00 24 4b fe 32 77 77 ...
56	0x40b	0	0x1000	88 02 00 00 24 4b fe 32 77 77 ...
57	0x40b	0	0x2000	88 02 00 00 24 4b fe 32 77 77 ...
58	0x40b	0	0x3000	88 02 00 00 24 4b fe 32 77 77 ...
59	0x40b	0	0x4000	88 02 00 00 24 4b fe 32 77 77 ...
60	0x40b	0	0x5000	88 02 00 00 24 4b fe 32 77 77 ...
61	0x40c	0	0x0000	88 02 00 00 24 4b fe 32 88 88 ...
62	0x40c	0	0x1000	88 02 00 00 24 4b fe 32 88 88 ...

63	0x40c	0	0x2000	88 02 00 00 24 4b fe 32 88 88 ...
64	0x40c	0	0x3000	88 02 00 00 24 4b fe 32 88 88 ...
65	0x40c	0	0x4000	88 02 00 00 24 4b fe 32 88 88 ...
66	0x40c	0	0x5000	88 02 00 00 24 4b fe 32 88 88 ...
...	...	...	...	...

Ringraziamenti

Questo lavoro è frutto di una bellissima intesa nata tra me e il Professor Gringoli. Grazie a lui e al gruppo ANS, ho potuto comprendere a pieno la bellezza della ricerca e della sperimentazione.

Questo percorso è nato ormai un anno fa, quando un gruppo di persone si sono ritrovate in un ufficio per discutere della stesura di un articolo a partire da un lavoro svolto a fine di sostenere un esame. Ringrazio queste persone per avermi permesso di tastare così per la prima volta il mondo accademico "dalla parte opposta", dove l'obbiettivo non è "dimostrare di essere in grado" quanto "apportare un contributo al mondo della ricerca".

Grazie a ciò è nata la voglia di portare oggi questa tesi sperimentale, dopo aver conosciuto il relatore, Francesco Gringoli, ed esssere rimasto estasiato dal suo approccio alla ricerca. Grazie ai suoi insegnamenti e al suo entusiasmo ho potuto scrivere questa tesi. Lo ringrazio in quanto il suo impegno in questo progetto è stato infinitamente più grande di quanto mi aspettassi, essendo sempre stato disponibile anche alle ore più folli della notte.

Ringrazio la mia famiglia per avermi spronato costantemente soprattutto nei momenti di difficoltà, i miei genitori e nonni, che non si sono mai tirati indietro e non sono mai mancati.

Ringrazio tutti i ragazzi che mi sono sempre stati vicino, chi proseguirà la carriera accademica e chi invece ha preferito dedicarsi alla vita lavorativa, per avermi sostenuto e aiutato nel momento del bisogno, oltre che per aver condiviso ogni momento all'interno dell'università.

A chi mi è sempre stato vicino e ha condiviso ogni momento, dentro e

fuori la vita accademica, che mi ha fatto sorridere quando afflitto da ansie e paure: grazie.

Mi rendo conto che da questo momento tutto sta per cambiare: comunque sia, spero che i legami formati con tutti voi possano persistere anche nel futuro più lontano.

- *Stefano*