Escuela Técnica Superior de Ingenieros Industriales y de Telecomunicación

École d'ingénieurs généraliste dans les domaines des nouvelles technologies

A study of IEEE 802.11ah and its SDR implementation

Degree Final Project

Author: Berta Remírez Moreno
Director: Antonio López
Supervisor: Laurent George

Paris, France

20/06/2016

ABSTRACT

This degree final project studies the standard IEEE 802.11ah and to implement it by Software Defined Radio is used through the program GNU Radio.

First, others standards and technologies are studied, standards and technologies like IEEE 802.15.4, LoRa, SIGFOX and EnOcean. To can do a comparison with the standard IEEE 802.11ah and to can appreciate the improvements that this new standard presents.

Developing the standard IEEE 802.11ah and specifically the physical layer and the MAC layer. The physical layer, this standard uses MIMO-OFDM, the spread spectrum used is a Direct Sequence Spread Spectrum (DSSS) and the channelization in different countries.

The MAC layer has different concepts where they are developed. The MTU, two kind of stations (TIM and Non-TIM), a new parameter (AID) is defined and this leads the number of stations supported is higher. The power saving mode where each station can be between awake state and doze state. The channel access is different for TIM stations and Non-TIM stations. The process that a station has to do the association and authentications and the throughput enhancements.

To implement the standard IEEE 802.11ah, first it is studied and it is analysed IEEE 802.11 a/g/p model done by the program GNU Radio. Moreover, the changes are made to adapt it to IEEE 802.11ah model. When changes are already done, the results of the transmitter and the loopback are analysed. At the transmitter, the results are FFTs of different transmission modes. In the loopback, the results are the different modulations.

Keywords: IEEE 802.11ah, SDR, sub-1GHz.

INDEX

GLOSSARY

3GPP	3rd Generation Partnership Project
16QAM	16 Quadrature Amplitude Modulation
64QAM	64 Quadrature Amplitude Modulation
256QAM	256 Quadrature Amplitude Modulation
ACK	Acknowledgement
ACT	Authentication Control Threshold
AID	Association Identifier
AM	Amplitude Modulation
ASK	Amplitude Shift Keying
BPSK	Binary Phase Shift Keying
BT product	Bandwidth Time product
CAP	Contention Access Period
CFP	Contention Free Period
CP	Cyclic Prefix
CSS	Chirp Spread Spectrum
DSSS	Direct Sequence Spread Spectrum
DTIM	Delivery Traffic Indication Map
FFT	Fast Fourier Transform
FSK	Frequency Shift Keying
GTS	Guaranteed Time Slots
GW	Gateway

HSDPA	High Speed Downlink Packet Access
ID	Identifier
IEEE	Institute of Electrical and Electronics Engineers
IFFT	Inverse Fast Fourier Transform
IoT	Internet of Things
ISM	Industrial Scientific Medical
LAN	Local Area Network
MAC	Medium Access Control
MAN	Metropolitan Area Network
MIMO	Multiple Inputs Multiple Outputs
MTU	Maximum Transmission Unit
OFDM	Orthogonal Frequency Division Multiplexing
O-QPSK	Offset Quadrature Phase Shift Keying
QPSK	Quadrature Phase Shift Keying
PSK	Phase Shift Keying
RA	Resource Allocation
RAW	Restricted Access Window
RF	Radiofrequency
RFID	Radio Frequency Identification
SDR	Software Defined Radio
SISO	Single Input Single Output
STBC	Space Time Block Code

STTC	Space Time Trellis Codec
TGah	Task Group ah
TIM	Traffic Indication Map
Tu	Period of a symbol
UMTS	Universal Mobile Telecommunications System
USRP	Universal Software Radio Peripheral
V-BLAST	Vertical-Bell Laboratories Layered Space-Time
WLAN	Wireless Local Area Network
WPAN	Wireless Personal Area Network

1. INTRODUCTION

IEEE 802.11 Wireless Local Area Network works at 2,4 GHz and 5 GHz bands, is one of the most popular Wireless technologies. These bands are high frequency and this means they have limits for transmission ranges.

Besides, the 2,4 GHz and 5 GHz spectra are saturated due to the excessive utilization. Although a number of solutions based on RFID, ZigBee, Bluetooth, or other WPAN technologies already support low power device communication, their capabilities are limited by the number of devices, throughput, transmission range, [7] etc. Most of these technologies can operate at the same frequency bands than 802.11ah, 868 MHz in Europe and 915 MHz in US. And these technologies have a data rate and coverage lower than IEEE 802.11ah, as it can be seen in the following sections where they are developed.

3GPP, IEEE and other international organizations adopt their technologies to the emerging market of the IoT. For example, IEEE 802.11 (Wi-Fi), used everywhere nowadays, is not suitable for the IoT, since this wireless technology is originally designed to offer high throughput to a limited number of stations located indoor at a short distance between each other and it has a high energy consumption. To meet IoT requirements, IEEE 802 LAN/MAN Standards Committee (LMSC) has formed IEEE 802.11ah Task Group (TGah) to extend the applicability area of 802.11 networks by designing an energy efficient protocol allowing thousands of indoor and outdoor devices to work in the same area. [7]

Moreover, the IEEE 802.11ah Task Group is working in a new IEEE 802.11 WLAN named IEEE 802.11ah standard. This standard is expected to be finished by 2016 and it will use frequencies bands lower than 1GHz. It will be operating sub 1 GHz industrial, scientific and medical (ISM) bands (868 MHz and 915 MHz).

Increasing the spectral efficiency is one of the main concerns in this system design. The Task Group has designed a new physical layer based on the IEEE 802.11ac [6]. To increase the system throughput, the TGah has also designed a Medium Access Control layer, to support more number of stations.

This new standard doesn’t need to maintain compatibility with other 802.11 systems, because this standard operates at different frequency bands. Low-cost and large coverage properties make 802.11ah system attractive for large scale sensor networks [6].

The 802.11ah has enhanced transmission range and large coverage because of the properties of low frequency spectrum.

In this degree final project, first it is studied the existing solution for IoT, like IEEE 802.15.4, LoRa, SigFox and EnOcean. Moreover, it will do a little comparison of these standards and technologies and the standard IEEE 802.11ah. The main objective of this project is studied the new standard IEEE 802.11ah and implement it by SDR. Previously, it is developed the physical and MAC layer of this new standard.

This degree final project is divided in different sections. Section 2 describes the standard IEEE 802.15.4. Section 3 describes LoRa technology. Section 4 provides a brief overview of SigFox technology. EnOcean technology is developed in Section 5. The IEEE 802.11ah standard is developed in Section 6, where it describes sensors networks, backhaul networks, the PHY layer and the MAC layer. Section 7 describes Software Defined Radio, GNU Radio, the model IEEE 802.11 a/g/p and the model IEEE 802.11ah. In the Section 8 is developed the conclusions. In the Section 9 are the references used.

2. IEEE 802.15.4/ZigBee

The IEEE 802.15 standard consists of 3 classes of Wireless Personal Area Network (WPAN). These three classes are distinguished by data rate, battery consumption and Quality of Service (QoS).

This section focuses on the standard IEEE 802.15.4 which is a LR-WPAN (low-rate). It operates in the 2.4 GHz ISM band, it uses three frequency bands, 868 MHz for Europe, 915 MHz for US and 2.4 GHz for worldwide, the first two are important because they are the same as for IEEE 802.11ah. For applications on low-power, low-cost wireless communication and low-bit rate connectivity towards small devices has been designed.

The IEEE 802.15.4 standard supports BPSK modulation at 868 and 915 MHz and O-QPSK modulation at 2.4 GHz. There are three different data rates available because of different physical characteristics in each band. The coverage in this standard is between 10 meters and 75 meters.

Frequency band	868 MHz	915 MHz	2.4 GHz
Data rate	20 Kbps	40 Kbps	250 Kbps

Table 1. Data rates for each frequency band.

The physical layer consists in twenty seven channels, one for 868 MHz, ten for 915 MHz and sixteen for 2.4 GHz.

With regard to the kind of spread spectrum this standard uses direct sequence spread spectrum (DSSS).

IEEE 802.15.4 MAC has two modes, beacon-enabled and beaconless.

Beacon-enabled mode allows splitting of time into multiple active durations with a cluster[1] having exclusive access to the transmission channel during its active duration. The coordinator broadcast a beacon to inform other nodes in the cluster about the beginning of the cluster's active duration. The cluster nodes compete for channel access during their active period using a slotted CSMA/CA algorithm. The devices transmit data during two periods, the first period is the Contention Access Period (CAP) and it is explained just before. The second period is Contention Free Period (CFP) which consists of Guaranteed Time Slots (GTS) allocated to individual devices by the network coordinator. The GTS is used by devices for cyclic data transmission and the coordinator can allocate GTS to a maximum of only seven devices. [27]

[1] consisting of a coordinator and its associated nodes.

In the beaconless operation, there is no division of time into active durations and a node competes for channel access with other nodes in its radio range using an unslotted CSMA/CA algorithm. [1]

The network can assume either a star topology, or cluster tree topology, or mesh topology, or operate in peer-to-peer mode.

There are four frame types:

- Beacon frames are used by the coordinator to describe the channel access mechanism to other nodes.
- Data frames are used to send varying amount of payload (2 -127 bytes).
- ACK frames are used to increase reliability for data frame and control frame transmissions.
- MAC control frames are used to carry out network management functions, such as association to and disassociation from the network.

[1][2][3][27]

3. LoRa

LoRa is a proprietary spread spectrum modulation scheme that is derivative of Chirp Spread Spectrum (CSS). CSS is a spread spectrum technique similar to direct sequence spread spectrum (DSSS). This technique uses wideband linear frequency modulated chirp pulses to encode information. A chirp is a sinusoidal signal whose frequency increases or decreases over time. [24][25]

LoRa implements a variable data rate, utilizing orthogonal spreading factors, which allows the system designer to trade data rate for range or power, so as to optimize network performance in a constant bandwidth.

LoRa is a physical layer implementation and is agnostic with to higher-layer implementations. This allows LoRa to coexist and interoperate with existing network architectures.

LoRa operates in 433 MHz, 868 MHz and 915 MHz ISM bands. It employs a version of CSS with a channel bandwidth of 125 KHz. Data rates are between 0.3 Kbps and 50 Kbps when channel aggregations are employed.

Talking about the spread spectrum, CSS was developed for radar applications in the 1940's. This modulation technique has increased the number of data communications due to its low transmission power requirements and robustness from channel degradation mechanisms such as multipath, fading, Doppler and in-band jamming interferences.

In LoRa modulation the spreading of the spectrum is achieved by generating a chirp signal that continuously varies in frequency. An advantage of this method is that timing and frequency offsets between transmitter and receiver are equivalent, greatly reducing the complexity of the receiver design. The frequency bandwidth of this chirp is equivalent to the spectral bandwidth of the signal.

LoRa modulation has the following properties:

- Bandwidth scalable and frequency scalable. It can use for narrowband frequency hopping and wideband direct sequence applications. LoRa can adapt for either mode of operation.
- LoRa is a constant envelope modulation scheme which means the same low cost and low power high-efficiency PA stages can be reused without modification.
- It is resistant to in-ban and to out-of-band interference mechanisms because of the high Bandwidth Time product (BT product). The time–bandwidth product of a pulse is the product of its temporal duration and spectral width (in frequency space). This parameter

is used to indicate the pulse quality. [26] LoRa symbol can be long, it provides for immunity to pulsed AM interference mechanisms.

- LoRa offers immunity to multipath and fading making it ideal for use in urban environments.
- Doppler shift causes a small frequency shift in the LoRa pulse which introduces a relatively negligible shift in the time axis of the baseband signal. This frequency offset tolerance mitigates the requirement for tight tolerance reference clock sources.
- The star topology is the most common topology. Typically a central coordinator acts as the conduit for all network traffic. Helps to minimize the amount of network traffic.

[4][5][24][25][26]

4. SIGFOX

SIGFOX is an operated telecommunication network, dedicated to the Internet of Things. SIGFOX is seamless and out-of-the box, allowing you to forget about communication and keep focused on the core of your project.

It is a Low-Power Wide-Area (LPWA) network, currently deployed in Western Europe, San Francisco, and with ongoing tests in South America and Asia. Its focus is on energy efficiency it allows you to build connected devices able to last years on a standard battery.

In a matter of communication, the SIGFOX technology allows a bidirectional communication, both from and to the device. The communication is always initiated by the device.

The SIGFOX network is designed for small messages occasionally. It is not appropriate for high-bandwidth usages.

This technology operates on sub 1GHz ISM bands, 868 MHz for Europe and 902 MHz for the United States. It is an ultra-narrow band technology. The modulation used is BPSK and it takes very narrow chunks of spectrum and changes the phase of the carrier radio wave to encode the data. This allows the receiver to only listen in a tiny slice of spectrum which mitigates the effect of noise.

There is no negotiation between the device and a receiving station. The device simply emits in the available frequency band. The signal is detected by the closest base stations, decoded and forwarded to the network backend.

Each message is authenticated using a hash mechanism, and a private key specific to the device. This offers a great protection against replay attacks. The SIGFOX radio protocol also offers a great resistance to interferers.

The message size is 12 bytes and the maximum number of messages that can be sent each day is 140 messages.

[17][18]

5. EnOcean

Patented EnOcean energy harvesting wireless sensor solution is able to generate a signal of astonishing range from an extremely small amount of energy.

In terms of high reliability EnOcean uses the following frequency bands, 868 MHz, 902 MHz and 928 MHz, sub 1 GHz frequency bands.

Multiple telegram transmission with checksum. These telegrams are short (1 ms) for little collision probability enabling a large number of EnOcean sensors operating in a system.

The coverage is up to 30 meters in buildings and is up to 300 meters in open space. Therefore, for range extension a repeater is available.

The communication can be one-way and bidirectional.

The data rate of this technology is 125 Kbps. The modulations used are ASK and FSK.

[19]

6. IEEE 802.11ah

Typically, the number of devices involved in a smart grid is much higher than in traditional 802.11 WLANs and the required transmission range of involved devices is also higher than in traditional 802.11 WLANs. In sub 1 GHz system, the coverage of one-hop transmission can be much higher, thus, allowing to support more devices in a single network. [6]

6.1. Sensors networks

Most of the uses cases are sensing applications such as gas, water and power consumption. An electrical grid is named smart grid.

With an access point, it covers a large number of sensors due to sub 1GHz frequencies. These sensors transmit short packets periodically, this shorts packets increase overhead cause by headers.

The standard has to take into account the energy consumption.

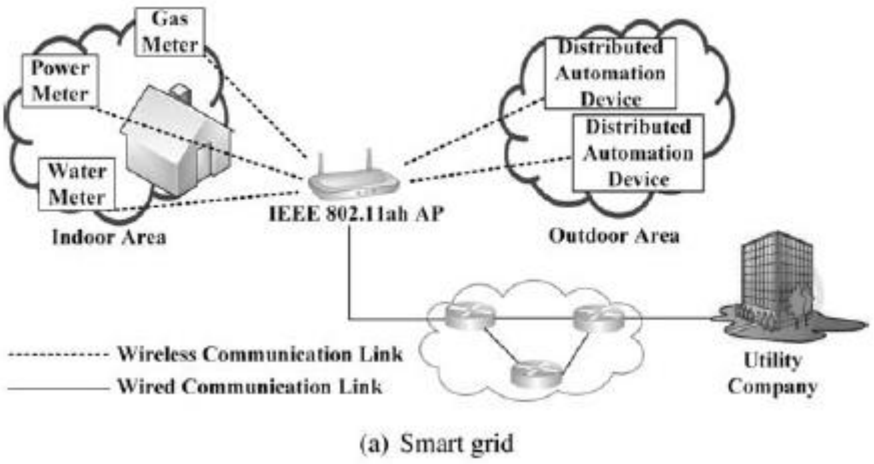


Figure 1 [6]

In this figure, there are sensors like as gas, power and water in indoor area. The access point is in outdoor and a distributed automation devices. The coverage of an access point, which is in an outdoor area, is almost 1 Km.

6.2. Backhaul networks

The other uses cases adopted by IEEE 802.11ah Task Group covers the backhaul connection between sensors and remote server. The large coverage of Sub 1 GHz allows a simple network design to link Sub 1GHz APs together, e.g., as wireless mesh networks.

IEEE 802.11ah can be used to transmit video streaming because IEEE 802.15.4g rates are insufficient for that.

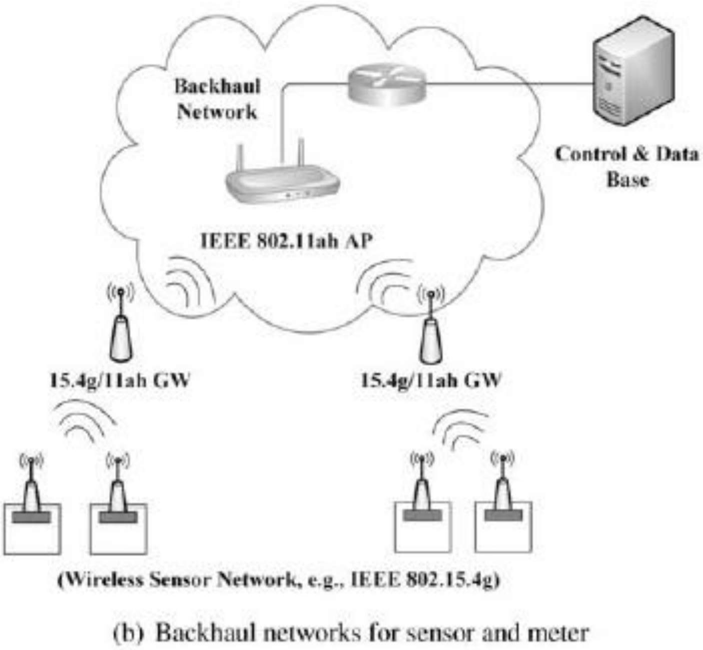

Backhaul Network
Control & Data Base
IEEE 802.11ah AP
15.4g/11ah GW
15.4g/11ah GW
(Wireless Sensor Network, e.g., IEEE 802.15.4g)
(b) Backhaul networks for sensor and meter

Figure 2 [6]

The figure 2 is composed with a wireless backhaul network, there are an access point (AP in the image) and gateways (GW in the image).

6.3. PHY Layer

6.3.1. OFDM

There are two categories. The first one, is the transmission mode of 2 MHz channel bandwidth and the other one, is the transmission mode of 1 MHz channel bandwidth.

For 2 MHz and greater modes (2 MHz, 4 MHz, 8 MHz and 16 MHz) have been adopted techniques like OFDM, which is to use a large number of carriers (2048, 8192), each one modulated QPSK, 16QAM or 64QAM.

The frequency separation is equal than 1/period of symbol (Tu).

In the Y axis, it is located the magnitude. In the X axis, it is located the frequency.

Nulls at frequencies multiples of 1/Tu. (Figure 3)

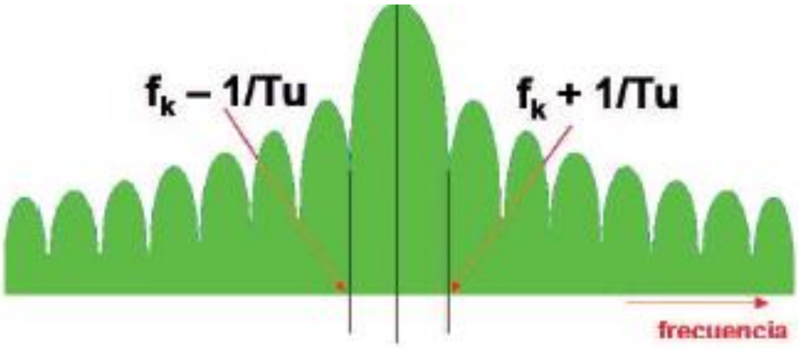


Figure 3[8]

Orthogonality means that data can be transmitted in carrier very close without interfering with each other. As follows, as shown in Figure 4.

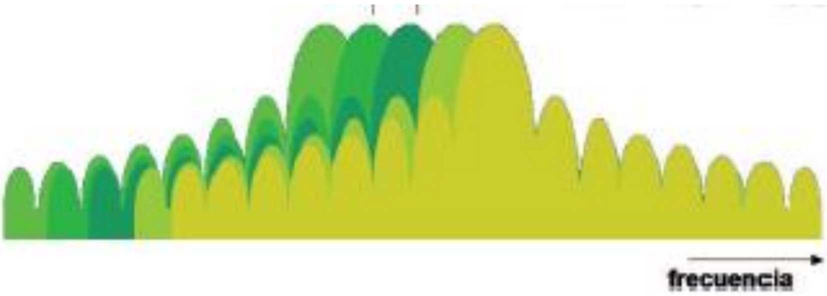


Figure 4 [8]

It is needed a Guard Interval for prevent Intersymbol Interference (ISI). At the transmitter, a time interval is inserted after transmission of each symbol, this is called guard interval. This Guard Interval is a portion of an OFDM symbol containing redundant data. It is used to ensure the different transmission do not interfere between them. The purpose of the guard interval is to introduce immunity to delays, to echoes and reflections to propagation.

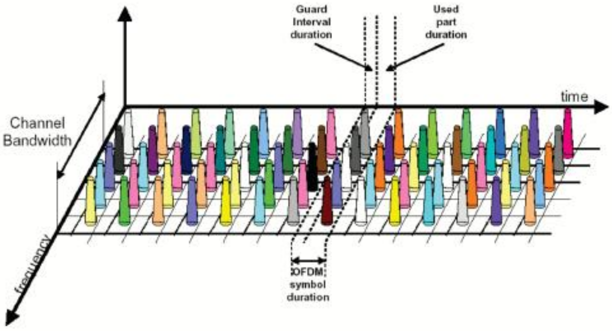


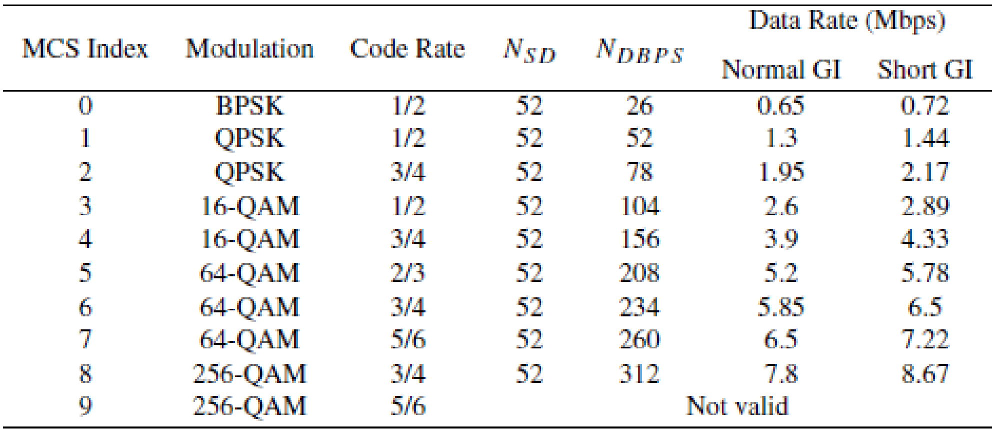
Figure 5 [8]

In 802.11ah, an OFDM symbol duration is ten times of that of 802.11ac.[6] There is the same number of data tones in 2 MHz, 4 MHz, 8 MHz and 16 MHz channels than in 20 MHz, 40 MHz, 80 MHz and 160 MHz channels in 802.11ac.

When it is used 2 MHz transmission mode, to generate an OFDM symbol it is needed 64 Fast Fourier Transform (FFT), there are 64 subcarriers, the number of subcarriers used to transmit data is 52.

About Guard Interval, there is short GI or normal. The shortest is 4 µs and OFDM symbol duration is 36 µs, the normal GI is 8 µs and OFDM symbol duration is 40 µs.

At the following table, it can see that data rate increase as a function of the shortest guard interval.



MCS Index	Modulation	Code Rate	N_{SD}	N_{DBPS}	Data Rate (Mbps)	
					Normal GI	Short GI
0	BPSK	1/2	52	26	0.65	0.72
1	QPSK	1/2	52	52	1.3	1.44
2	QPSK	3/4	52	78	1.95	2.17
3	16-QAM	1/2	52	104	2.6	2.89
4	16-QAM	3/4	52	156	3.9	4.33
5	64-QAM	2/3	52	208	5.2	5.78
6	64-QAM	3/4	52	234	5.85	6.5
7	64-QAM	5/6	52	260	6.5	7.22
8	256-QAM	3/4	52	312	7.8	8.67
9	256-QAM	5/6	Not valid			

Table 2 [6]

When it is used 1 MHz transmission mode, to generate an OFDM symbol it is needed 32 Fast Fourier Transform (FFT), there are 32 subcarriers, the number of subcarriers used to transmit data is 24.

6.3.2. MIMO

Multiple antennas can be used at the transmitter and at the receiver. It uses the advantages presented in spatial diversity, increase the transmission rate. It can be implemented as channel access HSDPA (High Speed Downlink Packet Access), as defined in the UMTS norm.

There are 3 categories:

- The first category that improve the efficiency of power, with techniques like STBC (Space Time Block Codec), STTC (Space Time Trellis Codec)

- The second category that use approximations in layers to increase capacity. An example is V-BLAST (Vertical-Bell Laboratories Layered Space-Time) signals are transmitted on antennas fto improve the transmission rate (normally, it is not reached the total diversity)
- The third category that exploits the channel knowledge at the transmitter (uses the channel information to do a pre and post filtering in the transmitter and receiver, to achieve gain in capacity)

For a MIMO system with N transmitter antennas and M receiver antennas the diversity gain system is d=N·M. [14]

6.3.3. MIMO-OFDM

OFDM can transform a MIMO channel frequency selection, to a set of parallel channels, it reduces the complexity of the receiver.

6.3.3.1. Transmitter

The bitstream is encoded using FEC (Forward Error Correction) technique. When the bitstream is encoded, it is converted to a constellation by a digital modulation, and then, is encoded by a MIMO encoder, where each output corresponds to a symbol stream which will be modulated using OFDM technique and will be transmitted by an antenna. Each symbol is assigned a cyclic prefix (CP) and a preamble.

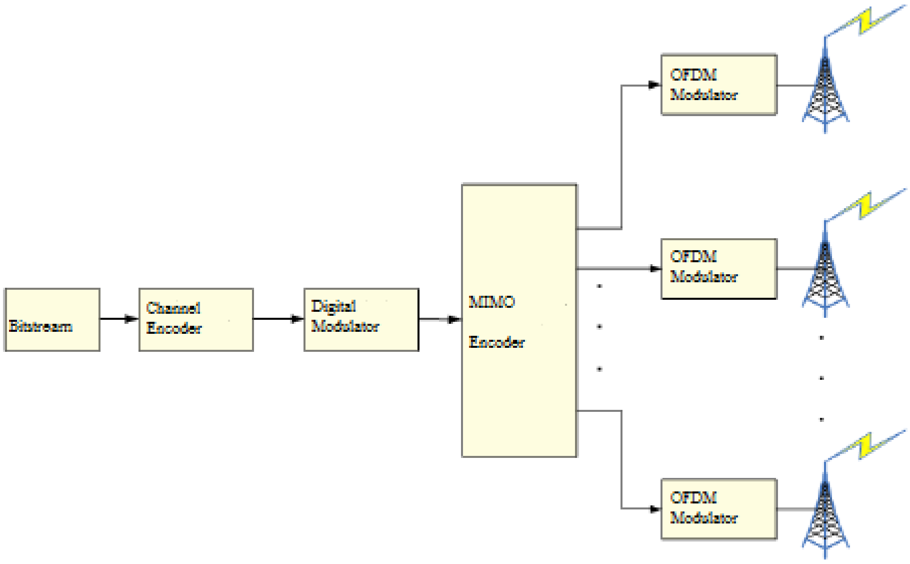


Figure 6 [14]

6.3.3.2. Receiver

The symbols received by the antennas are synchronized in frequency and time with the help of the preamble. The cyclic prefix (CP) and the preamble are extracted, the OFDM symbols are demodulated by the FFT. Then the symbols are decoded with a MIMO decoder, demodulated and decoded. Thus, the bistream is recovered.

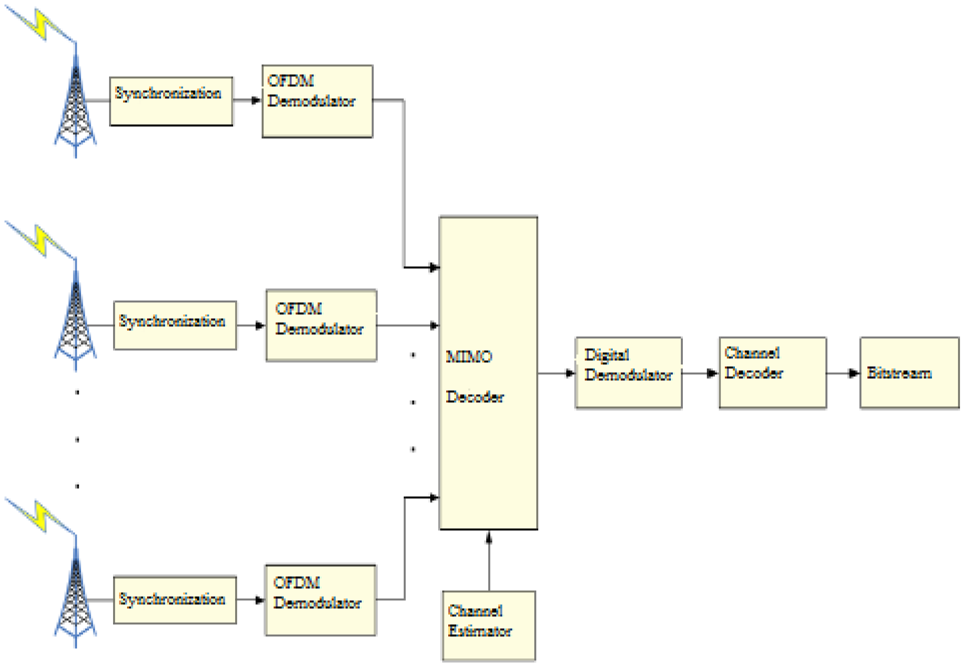


Figure 7 [14]

6.3.4. Direct Sequence Spread Spectrum (DSSS)

Direct Sequence Spread Spectrum is a redundant bit pattern, IEEE 802.11 recommends 11 bits (but the optimum number is 100 [20]), for each of the bits. The sequence must have the same numbers of 1's and 0's (it is the Baker sequence).

The transmitter has to first send the sequence to the receiver, because it is the only way to recover the original signal.

The receiver can reconstruct the information from the received signal by replacing each bit to be transmitted by sequence of 11 bits equivalent, although part of the transmission signal is affected by interferences.

Benefits of using DSSS:

- Resistance to intended or unintended jamming.
- Sharing of a single channel among multiple users.
- Reduced signal/background-noise level hampers interception.

- Determination of relative timing between transmitter and receiver.

[20][21]

6.3.5. Channelization

The channelization is different depending on countries, thus, IEEE 802.11ah has different sub 1 GHz ISM bands for various countries, i.e. United States, South Korea, China, Europe, Japan and Singapore. (Figure 8)

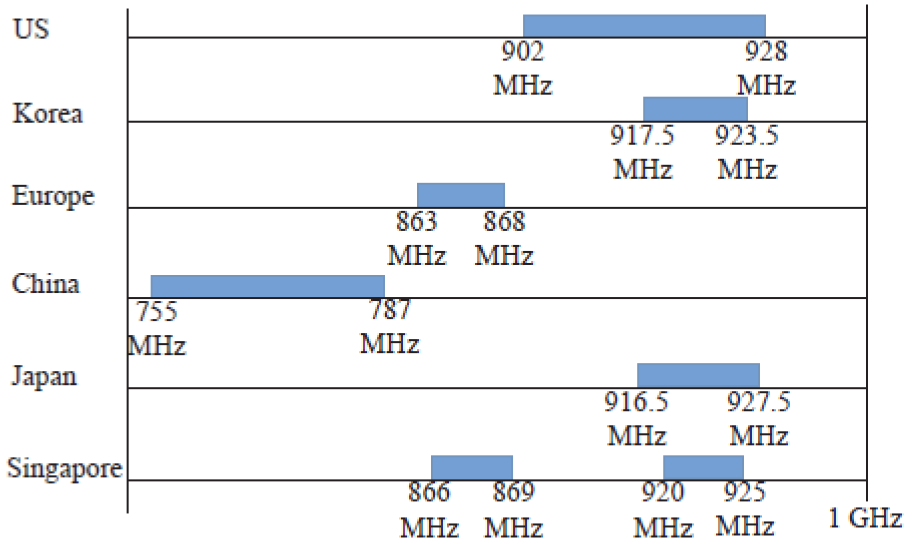


Figure 8 [6]

The US channelization has 26MHz band between 902 MHz to 928 MHz. The number of 1MHz channels is 26, 2 MHz channels are composed of two adjacent 1 MHz channel, and thus, there are 13 channels. 4 MHz channels are composed of two adjacent 2 MHz channel so there are 6 channels. By the same way, there are 3 channels of 8 MHz.. Moreover, there is a 16 MHz channel.

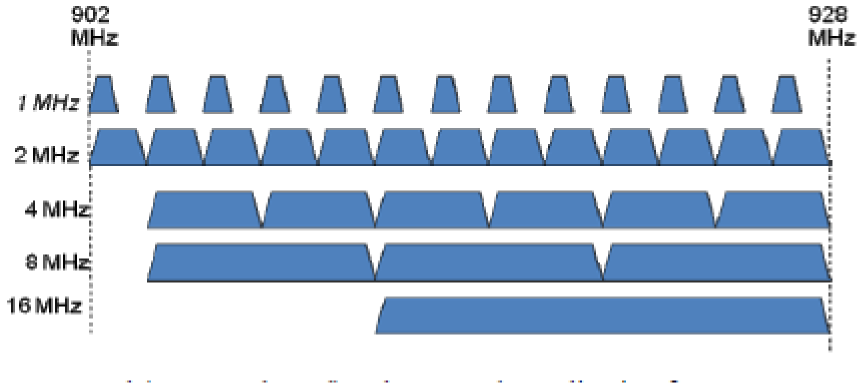


Figure 9. Channelization for US [9]

The South Korea channelization is between 917.5 MHz to 923.5 MHz. The number of 1 MHz channels is 6, the number of 2 MHz channels is 3 and the widest here is 4 MHz channel. There is a 0.5 MHz frequency offset and the reason for this is to reduce possible mutual interference with wireless systems at low frequencies.

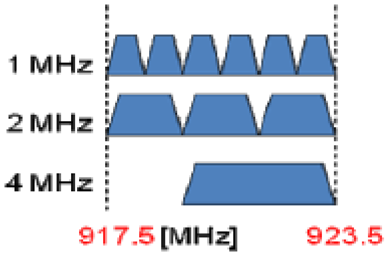


Figure 10. Channelization for Korea [9]

The Europe channelization is between 863 MHz to 868 MHz. The number of 1 MHz channels is 5, the number of 2 MHz channels is 2.

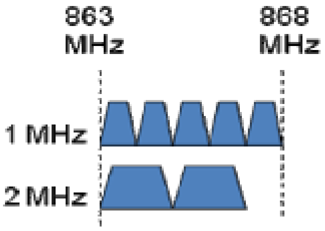


Figure 11. Channelization for Europe [9]

The China channelization starts at 755 MHz and ends at 787 MHz. Between 755 MHz to 779 MHz, the maximum power allowed is 5 mW and between 779 MHz to 787 MHz is 10 mW. The number of 1 MHz channels is 8, the number of 2 MHz channels is 4, the number of 4 MHz channels is 2, the number of 8 MHz channels is 1, and this one is the widest.

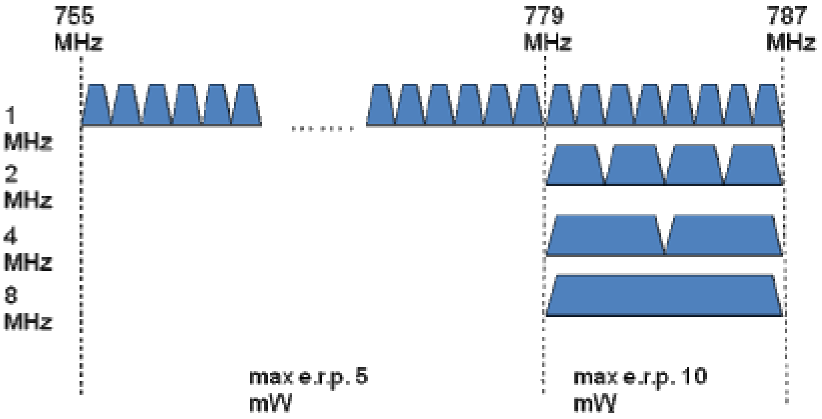


Figure 12. Channelization for China [9]

The Japan channelization is between 916.5 MHz to 927.5 MHz. The number of 1 MHz channels is 11. As in the case of Korea, this channelization has a 0.5 MHz frequency offset. This is due to the fact that Japanese spectrum regulation specifies center frequencies instead of start/stop bands such as in other countries. [9]

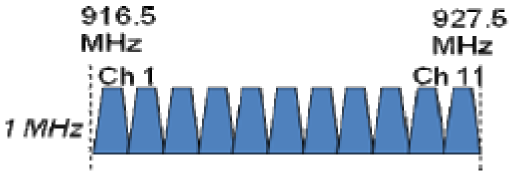


Figure 13. Channelization for Japan [9]

The Singapore channelization starts at 920 MHz and ends at 925 MHz. The number of 1 MHz channels is 5.

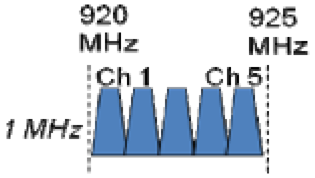


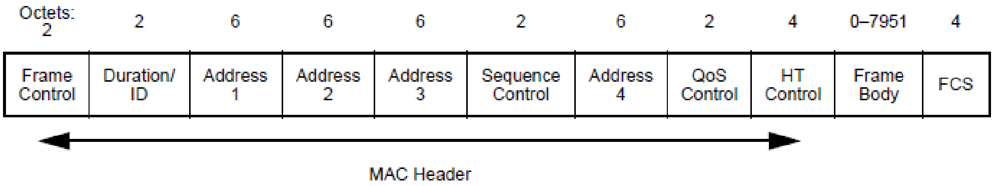
Figure 14. Channelization for Singapore [9]

6.4. MAC Layer

TGah wants to design a MAC layer that maximizes the number of stations supported. The IEEE 802.11ah MAC presents enhancements compared to IEEE 802.11 MAC.

6.4.1. Packet Size

The maximum transmission unit of a WLAN protocol (802.11) is 7991 bytes as it is shown in the following figure.



Octets: 2	2	6	6	6	2	6	2	4	0–7951	4
Frame Control	Duration/ ID	Address 1	Address 2	Address 3	Sequence Control	Address 4	QoS Control	HT Control	Frame Body	FCS

Figure 15. Data frame [16]

6.4.2. TIM Stations

This kind of stations needs to listen to access point beacons to send or receive data. The TIM stations listen to DTIM and TIM beacons to send or receive data. [10]

6.4.3. Non-TIM Stations

This kind of stations doesn't need to listen. Non-TIM stations directly negotiate with the access point to have a transmission time. The following transmissions can be renegotiated or not. Thus, these stations can transmit periodically. For high volume data applications TIM stations are preferred. Non-TIM stations only listen to DTIM beacons to send or receive data. [10]

6.4.4. Support of Many Associated Stations

IEEE 802.11ah supports a large number of stations associated to the same access point, than IEEE 802.11 (2007). To do this, the IEEE 802.11ah TaskGroup defined two parameters:

- Association identifier (AID), it is a unique ID that classifies stations into pages and blocks.

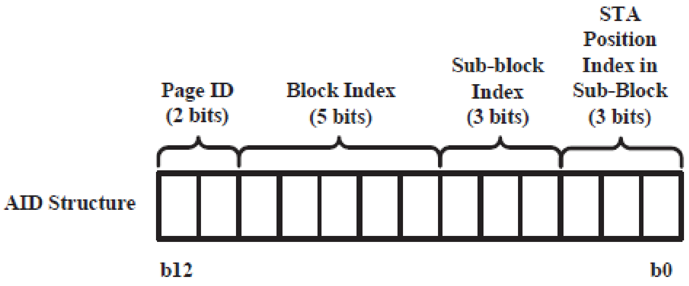


Figure 16. AID structure [6]

- The partial virtual bitmap of Traffic Indication Map (TIM) Information Element (IE), where each bit shows the relevant station's AID.

As shown in the figure 16, an AID structure is formed of 13 bits, so the number of stations supported is 8191 (2^{13}-1).

6.4.5. Power Saving

A station supporting the power saving mode can be in an awake state or in doze state. When a station is in a doze state, the packets received will be in buffers of the access point until the station wakes up. The access point buffers the packets until the station wakes up and requests the delivery of the buffered traffic. A station supporting power saving mode needs to wake up periodically to receive the beacon frame, sent by an access point, thus this way the station can check if there is some packet at the buffer of the access point.

Moreover, the station sends a control frame named Power Saving (PS)-poll frame to the access point to request the delivery of the buffered packets. When the reception of the buffered packets ends, the station can go back to doze state.

If there are a lot of packets in the buffer there can be stations in an awake state for a long time.

To reduce this time and to increase the sleep time, IEEE 802.11ah introduces a system called TIM and page segmentation. There are two types of beacons:

- DTIM (Delivery TIM) beacons are used to signal which TIM groups have pending data, unicast and/or multicast in the access point. The beacon contains information about the restricted access window properties. DTIM beacons indicate the buffering status of group addressed packets.
- TIM beacons indicate the presence of packets destined for each station. Between two consecutive DTIM beacons, there are as many TIM beacons as groups defined.

The station wakes up to receive the DTIM beacon. Each station can recognize which beacon contains the partial virtual bitmap of the page segment that each station belongs to. Then, each station can go back to the doze state, and can wake up when it receives a DTIM beacon.

If no buffered packets are for the station, the station returns to the doze state after the beacon. [7]

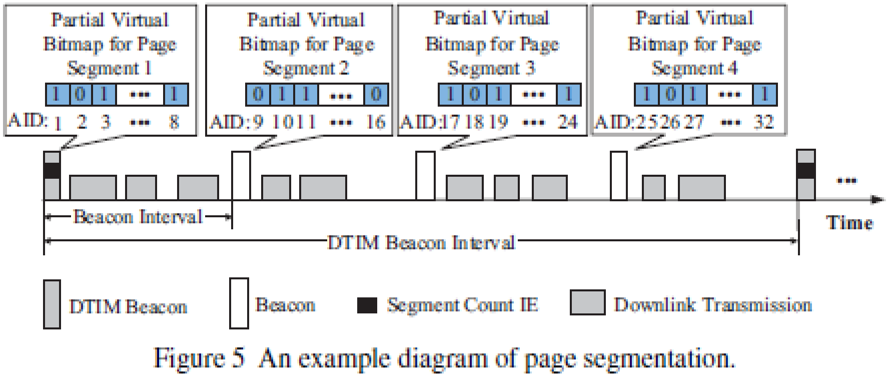


Figure 17. Diagram of page segmentation [6]

6.4.6. Channel Access

There are channel access mechanisms for TIM stations and Non-TIM stations, it is a novelty of 802.11 ah

For non-TIM stations, the access point allows the stations to request buffered traffic or send traffic after the stations wake up. In the case of many stations waking up at the same time, this can provoke delays and collisions at the channel.

To prevent this, an access point allows each station to wake up at a predefined time. For TIM stations a new concept is used called Restricted Access Window (RAW), it is a time duration composed of several time slots. An access point allocates a TIM station some time where the station can receive or transmit packets. RAW can be used for various purposes.

If the time slots were allocated to the stations which are ready for receive or transmit, it could be more efficient. The RA frame is transmitted at the beginning of the RAW and all the stations assigned to that RAW have to wake up and receive it. A RA frame is a frame called Resource Allocation that contains the scheduling information of every individual station.

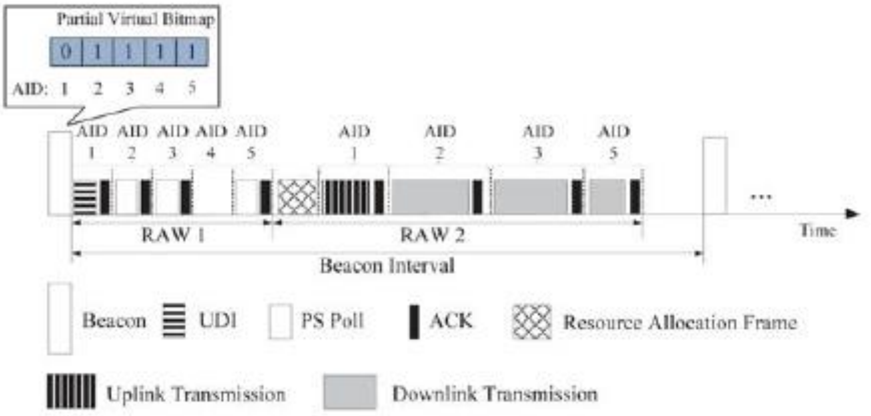


Figure 18. Uplink and downlink data delivery using RAW [6]

6.4.6.1. RAW (Restrict Access Window)

One or more RAW can be assigned in one beacon interval and only the designated stations can access to the channel in one RAW using the distributed coordination function (DCF) or enhanced distributed channel access (EDCA). [12]

A RAW can be divided into RAW slots and each one are assigned to a different station.

When the stations are associated, they can use RAW. This limits the number of associated stations.

[12]

6.4.7. Association and Authentication

IEEE 802.11ah is working on authentication control mechanisms. In every beacon, Authentication Control Threshold (ACT) is selected. The AP can change the ACT dynamically. The station initialized generates an authentication control number randomly in the interval [0, L] (L=1023) [30]. If a station authentication control number is less than the received ACT, the station can associate with the AP.

The station sends an authentication request to the AP. When the station is authenticated, the AP sends an authentication response. The station sends an ACK and an association request as it is authenticated. The association request contains chosen encryption types (if required) and other compatible 802.11 capabilities. If the elements in the association request match with the capabilities of the AP, the AP will create an association ID (AID) for the station and respond with an association response. The station sends an ACK and when the station is associated with the AP, the communication can starts. [12]

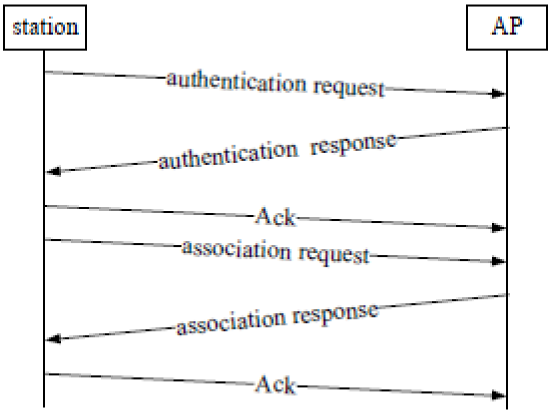


Figure 19 [12]

If there are 6000 stations trying to authenticate at the same time, most of authentication requests won't be received because of collisions. Although there was a response, the access point cannot access the channel to respond.

For 100 stations the time required to finish the authentication procedure can exceed 5 minutes, as it is shown in the following figure.

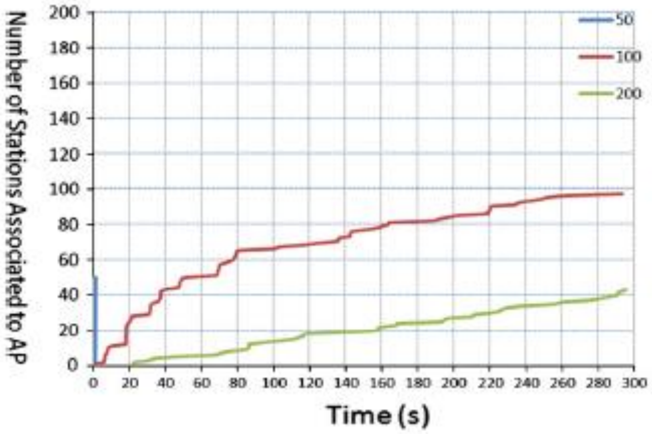


Figure 20 [7]

With the method to adjust Authentication Control Threshold described in [15], 100 stations connect successfully in just few seconds and one minute is enough for 2000 stations to connect.

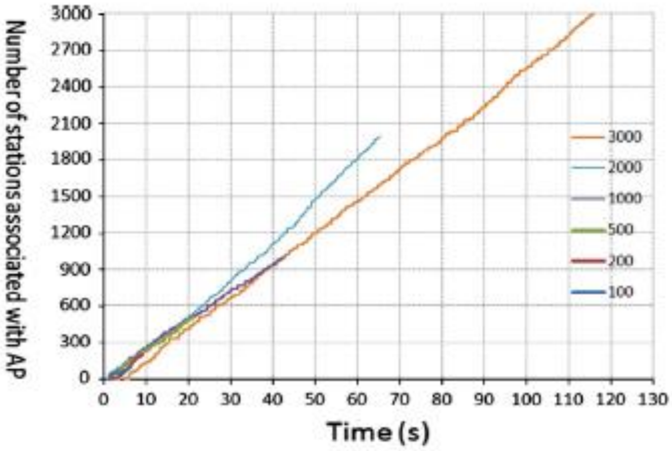


Figure 21 [7]

6.4.8. Throughput Enhancements

IEEE 802.11ah has a low data rates and to improve it, it can do more compact frame formats to reduce overheads.

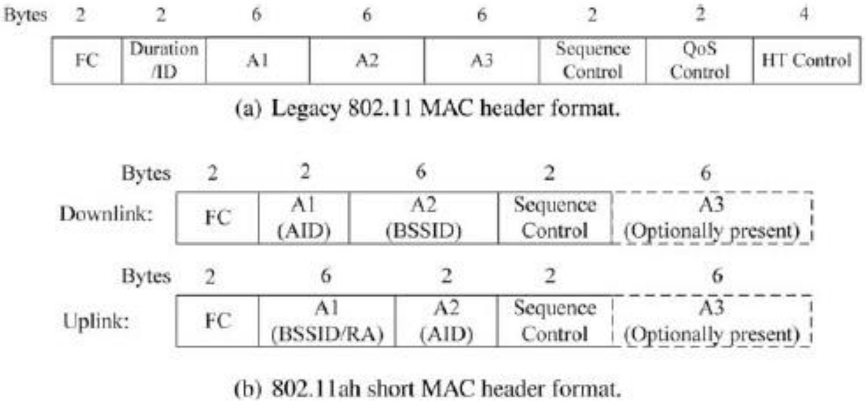


Figure 22 [6]

For the downlink, the address 1 (MAC address) is reduced to 2 bytes. This 2 bytes info is called AID. It is the same case for address 2 for the uplink. The address 2 for the downlink and the address 1 for the uplink are left unchanged. The address 3 is optional. The Frame Control field is used to indicate if the address 3 is present. There is information necessary contained in Quality of Service field (QoS) and High Throughput field (HT). This information is moved to Signal field in the PHY header. This can save 12 bytes.

IEEE 802.11ah has proposed an ACK frame format named Null Data Packet ACK, where the MAC header and FCS field are eliminated.

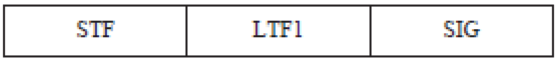


Figure 23. NDP ACK frame [6]

IEEE 802.11ah defines a new mechanism, by which the channel access delay and ACL transmission overhead are eliminated.

These are some mechanisms to improve throughput.

7. Comparison between standards and technologies

Standard/ Technology	Data Rate	Coverage	Spread Spectrum	Modulation
IEEE 802.11 ah	≤ 7.8 Mbps	1 Km (of an access point)	DSSS	MIMO-OFDM
IEEE 802.15.4/ZigBee	≤ 0.25 Mbps	10 m	DSSS	BPSK
LoRa	≤ 50 Kbps	15 – 20 Km	CSS	--
Sigfox	--	--	--	BPSK
EnOcean	125Kbps	30 meters in buildings and 300 meters in open space	--	ASK/FSK

Table 3. Comparison between standards and technologies

IEEE 802.11 standard has higher performances compared to 802.15.4 in terms of transmission range, throughput and so on. [3]

IEEE 802.15.4 network has no or little impact on IEEE 802.11's performance. However, it is potentially vulnerable to interference by other technologies as IEEE 802.11 and Bluetooth.

IEEE 802.15.4 is a LR-WPAN. This standard has different data rates as function of the frequency band. In IEEE 802.11ah, data rates vary depending on the modulation used. Data rates does not vary depending on the frequency band.

In EnOcean, the coverage varies depending if it is between buildings or if it is open space. The coverage in IEEE 802.11ah does not vary.

LoRa is a physical layer implementation. It is derivated of CSS.

IEEE 802.11ah is the only standard among studied using MIMO-OFDM.

The standard IEEE 802.11 ah has the highest data rate among these standards and technologies. Although it has not the larger coverage, the set of all characteristics (i.e. data rate, coverage, spread spectrum and modulation) makes the standard be the most appropriate for those frequencies bands (868 MHz and 915 MHZ).

8. Software Defined Radio

Software Defined Radio (SDR) is a radio communication system where components that have been typically implemented in hardware (e.g. mixers, filters, amplifiers, modulators/demodulators, detector, etc.) are instead implemented by means of software on a personal computer or embedded system.

A basic SDR system may consist of a personal computer equipped with a sound card, or other analog to digital convert, preceded by an adapter radiofrequency (RF). Such a design produces a radio which can receive and transmit different radio protocols (sometimes referred to as waveforms) based solely on the software used.

[22]

8.1. GNU Radio

GNU Radio is a free software development toolkit that provides signal processing blocks to implement software defined radios and signal processing systems. It can be used with external RF hardware to create software defined radios, or without hardware in a simulated environment.

The GNU Radio software provides the framework and tools to build and run software radio or just general signal processing applications. The GNU Radio applications are generally known as 'flow graphs', which are a series of signal processing blocks connected together, thus describing a data flow. As with all software defined radio systems, reconfigurability is a key feature. Instead of using different radios designed for specific but disparate purposes, a single general purpose, radio can be used as a radio front end, and a signal processing software, handles the processing specific to the radio application.

These flow graphs can be written in either C++ or in Python programming language. The GNU Radio infrastructure is written entirely in C++, and many of the user tools are written in Python.

[23]

8.2. Model IEEE 802.11 a/g/p

Through the IEEE 802.11 a/g/p model, it is going to make changes to adapt it to the model IEEE 802.11ah.

8.2.1. Wifi_phy_hier

In this section, it is developed the figure 24. This flow graph is a Hier Block, what it means that it will be a block in other flow graph (in this case, it is a transmitter block). In the first big block (the first block, pad source, it is an input of the Hier Block), the header of the packet is generated. The signal is converted to symbols and it is multiplexed.

Then, the carrier is allocated. Moreover, it is realized the IFFT. Next, the cyclic prefix is included. Then, the last block, after OFDM Cyclic Prefixer block, it is an output of the Hier Block.

In the second big block (the first block, pad source, it is an input of the Hier Block), it is done the complex conjugated and it is done the magnitude. And the moving average is done.

In the third big block, it is done to the signal the synchronization. Next, the signal is converted to a vector.

Then, it is done FFT to the signal, and the symbols are equalized. Next, the signal is decoded. The signal is divided, to one signal, it is realized a MAC decoding. The next block is an output of the Hier Block. And to the other signal, it is converted to stream. The next block is other output of the Hier Block.

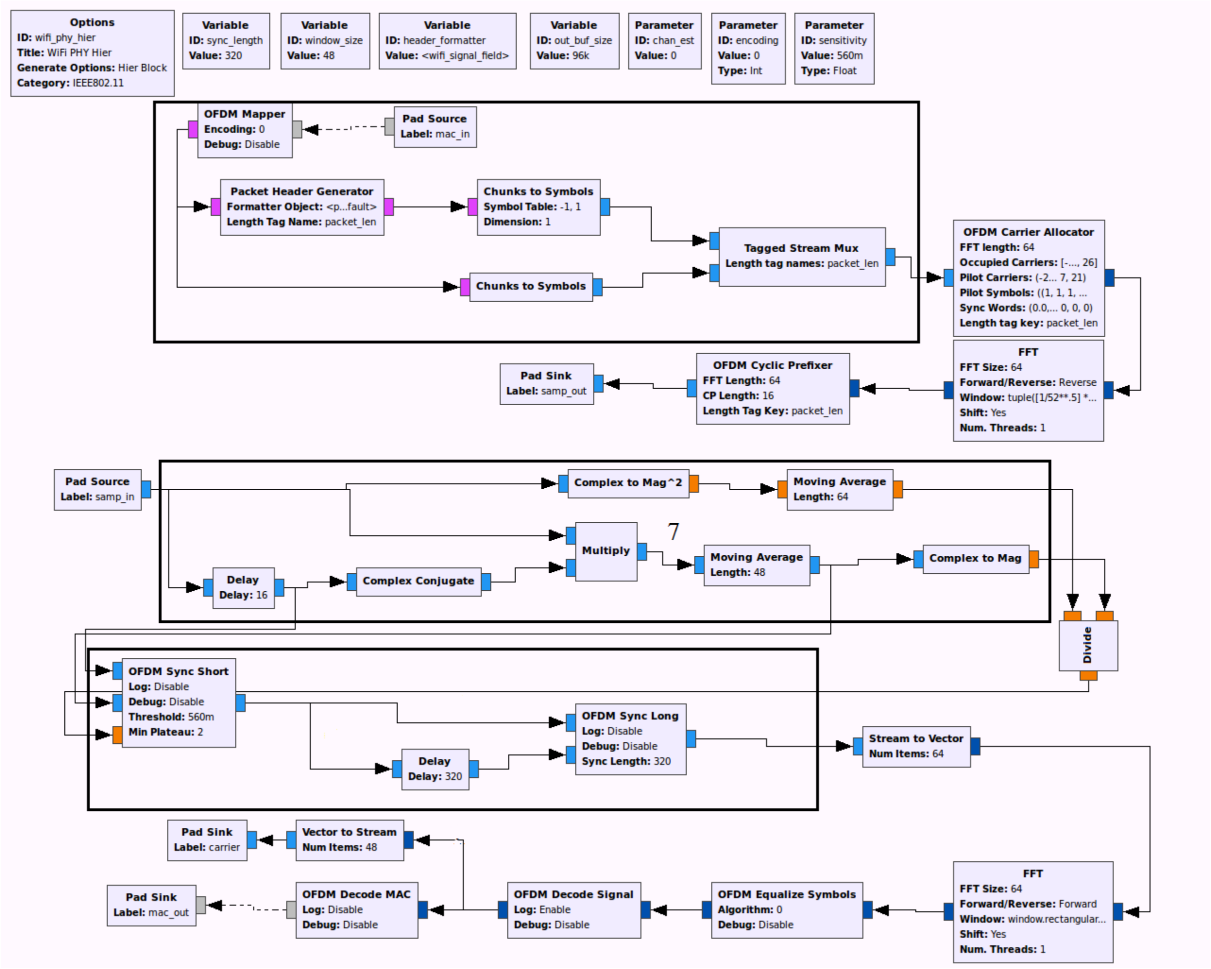
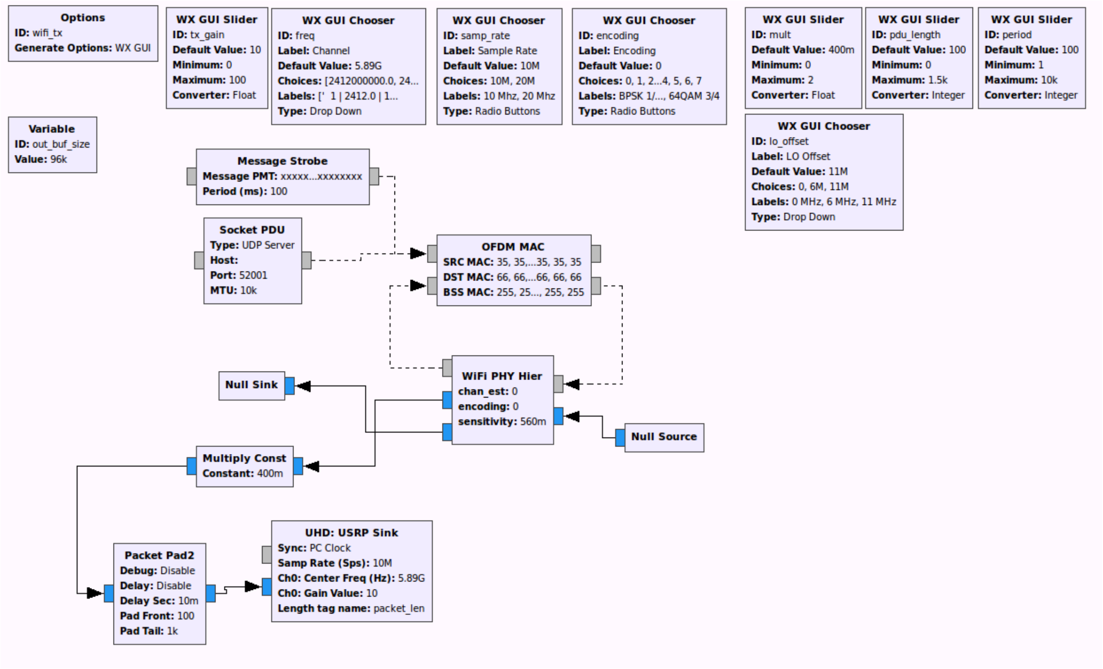


Figure 24. Flow graph of the wifi_phy_hier

8.2.2. Wifi_tx

In this section, it is described the figure 25. An entry of the block OFDM MAC consists of two signals, one is a PMT message with a default period and another is Socket PDU with a MTU of 10000. Another entry of this block comes from the WIFI PHY Hier block (described in the last section).

The output of this block, OFDM MAC, goes to the block WIFI PHY Hier like an entry. There are two entries, one is just described, and another is a null source. The outputs of this block, the first one goes to the OFDM MAC block like is described in the previous paragraph. The second one goes to a null sink. For the third one, the signal is multipling by a constant. This signal goes to the block Packet Pad2, with the debug disable and the delay also. Then, the signal goes to USRP Sink with a sample rate, a center frequency and the gain value.



Options
ID: wifi_tx
Generate Options: WX GUI

WX GUI Slider
ID: tx_gain
Default Value: 10
Minimum: 0
Maximum: 100
Converter: Float

WX GUI Chooser
ID: freq
Label: Channel
Default Value: 5.89G
Choices: [2412000000.0, 24...
Labels: [' 1 | 2412.0 | 1...
Type: Drop Down

WX GUI Chooser
ID: samp_rate
Label: Sample Rate
Default Value: 10M
Choices: 10M, 20M
Labels: 10 Mhz, 20 Mhz
Type: Radio Buttons

WX GUI Chooser
ID: encoding
Label: Encoding
Default Value: 0
Choices: 0, 1, 2...4, 5, 6, 7
Labels: BPSK 1/..., 64QAM 3/4
Type: Radio Buttons

WX GUI Slider
ID: mult
Default Value: 400m
Minimum: 0
Maximum: 2
Converter: Float

WX GUI Slider
ID: pdu_length
Default Value: 100
Minimum: 0
Maximum: 1.5k
Converter: Integer

WX GUI Slider
ID: period
Default Value: 100
Minimum: 1
Maximum: 10k
Converter: Integer

Variable
ID: out_buf_size
Value: 96k

WX GUI Chooser
ID: lo_offset
Label: LO Offset
Default Value: 11M
Choices: 0, 6M, 11M
Labels: 0 MHz, 6 MHz, 11 MHz
Type: Drop Down

Message Strobe
Message PMT: xxxxx...xxxxxxxx
Period (ms): 100

Socket PDU
Type: UDP Server
Host:
Port: 52001
MTU: 10k

OFDM MAC
SRC MAC: 35, 35,...35, 35, 35
DST MAC: 66, 66,...66, 66, 66
BSS MAC: 255, 25..., 255, 255

Null Sink

WiFi PHY Hier
chan_est: 0
encoding: 0
sensitivity: 560m

Null Source

Multiply Const
Constant: 400m

UHD: USRP Sink
Sync: PC Clock
Samp Rate (Sps): 10M
Ch0: Center Freq (Hz): 5.89G
Ch0: Gain Value: 10
Length tag name: packet_len

Packet Pad2
Debug: Disable
Delay: Disable
Delay Sec: 10m
Pad Front: 100
Pad Tail: 1k

Figure 25. Flow graph of the Transmitter

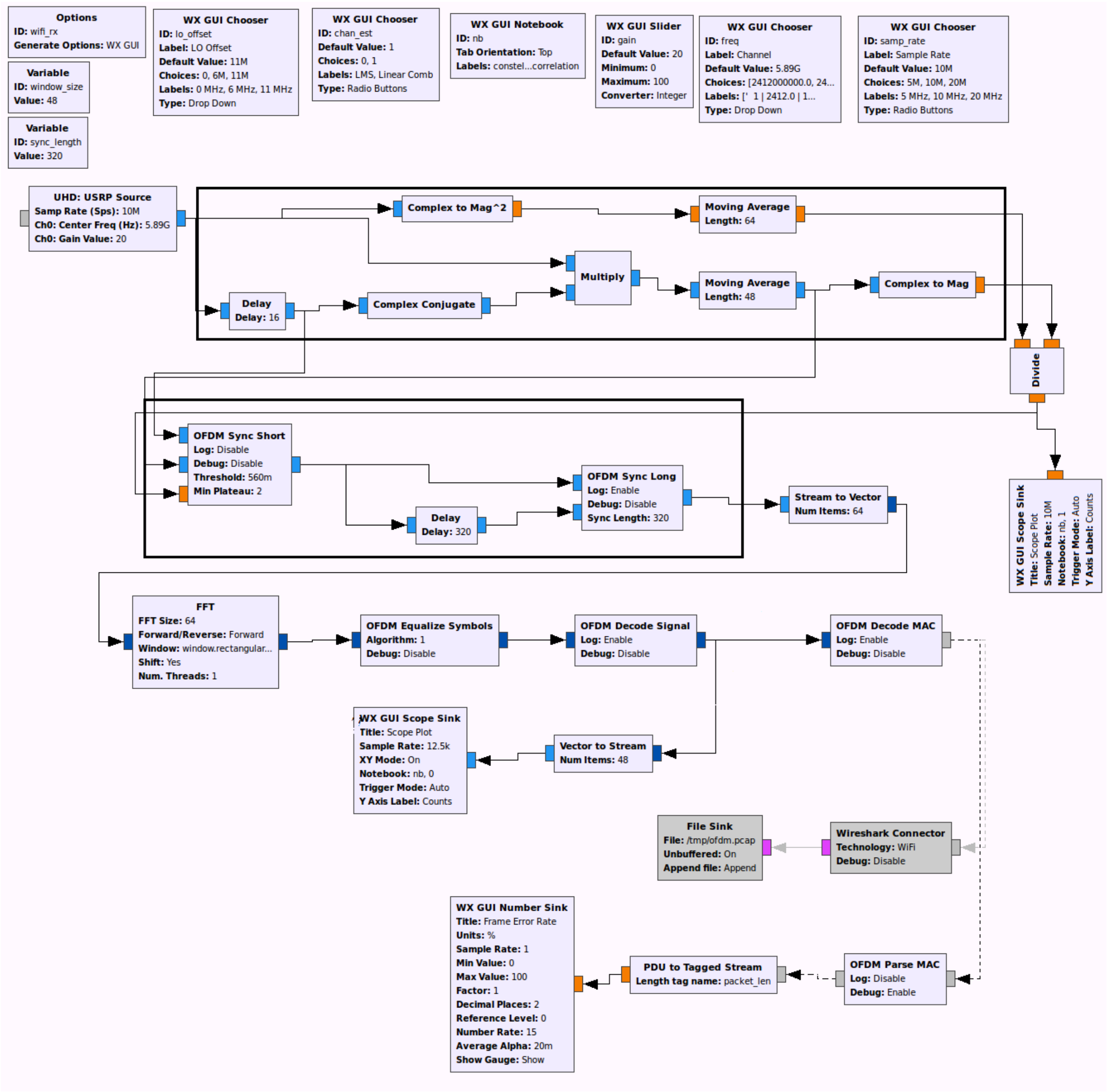
8.2.3. Wifi_rx

In this section, it is developed the figure 26. From the USRP source (Sink with a sample rate, a center frequency and the gain value), there is the first big block, it is done the complex conjugated and it is done the magnitude. And the moving average is done. A signal of this big block goes to a WX GUI Scope Sink.

Next, there is the second big block, it is done to the signal the synchronization. Then, the signal is converted to a vector.

The following step, it is FFT block, so, the FFT is done and the symbols are equalize (OFDM Equalize Symbols). The signal is decoded. The signal is divided, to one signal, it is converted to vector and it is going to a WX GUI Scope Sink.

To the other signal, it is realized a MAC decoding. This signal is divided in two, to one signal, it is going to a Wireshark connector and the, to a file sink. The other signal is going to an OFDM parse MAC, and next, it goes to the block PDU to Tagged Stream and the following block is a WX GUI Number Sink.



Options
ID: wifi_rx
Generate Options: WX GUI

Variable
ID: window_size
Value: 48

Variable
ID: sync_length
Value: 320

WX GUI Chooser
ID: lo_offset
Label: LO Offset
Default Value: 11M
Choices: 0, 6M, 11M
Labels: 0 MHz, 6 MHz, 11 MHz
Type: Drop Down

WX GUI Chooser
ID: chan_est
Default Value: 1
Choices: 0, 1
Labels: LMS, Linear Comb
Type: Radio Buttons

WX GUI Notebook
ID: nb
Tab Orientation: Top
Labels: constel...correlation

WX GUI Slider
ID: gain
Default Value: 20
Minimum: 0
Maximum: 100
Converter: Integer

WX GUI Chooser
ID: freq
Label: Channel
Default Value: 5.89G
Choices: [2412000000.0, 24...
Labels: [' 1 | 2412.0 | 1...
Type: Drop Down

WX GUI Chooser
ID: samp_rate
Label: Sample Rate
Default Value: 10M
Choices: 5M, 10M, 20M
Labels: 5 MHz, 10 MHz, 20 MHz
Type: Radio Buttons

UHD: USRP Source
Samp Rate (Sps): 10M
Ch0: Center Freq (Hz): 5.89G
Ch0: Gain Value: 20

Complex to Mag^2

Moving Average
Length: 64

Multiply

Moving Average
Length: 48

Complex to Mag

Delay
Delay: 16

Complex Conjugate

Divide

OFDM Sync Short
Log: Disable
Debug: Disable
Threshold: 560m
Min Plateau: 2

OFDM Sync Long
Log: Enable
Debug: Disable
Sync Length: 320

Delay
Delay: 320

Stream to Vector
Num Items: 64

WX GUI Scope Sink
Title: Scope Plot
Sample Rate: 10M
Notebook: nb, 1
Trigger Mode: Auto
Y Axis Label: Counts

FFT
FFT Size: 64
Forward/Reverse: Forward
Window: window.rectangular...
Shift: Yes
Num. Threads: 1

OFDM Equalize Symbols
Algorithm: 1
Debug: Disable

OFDM Decode Signal
Log: Enable
Debug: Disable

OFDM Decode MAC
Log: Enable
Debug: Disable

WX GUI Scope Sink
Title: Scope Plot
Sample Rate: 12.5k
XY Mode: On
Notebook: nb, 0
Trigger Mode: Auto
Y Axis Label: Counts

Vector to Stream
Num Items: 48

File Sink
File: /tmp/ofdm.pcap
Unbuffered: On
Append file: Append

Wireshark Connector
Technology: WiFi
Debug: Disable

WX GUI Number Sink
Title: Frame Error Rate
Units: %
Sample Rate: 1
Min Value: 0
Max Value: 100
Factor: 1
Decimal Places: 2
Reference Level: 0
Number Rate: 15
Average Alpha: 20m
Show Gauge: Show

PDU to Tagged Stream
Length tag name: packet_len

OFDM Parse MAC
Log: Disable
Debug: Enable

Figure 26. Flow graph of the Receiver

8.3. Model IEEE 802.11ah

First, the changes made from the model 802.11 a/g/p, it is explained in the previous section, are going to comment. These changes are made to adapt it to the model IEEE 802.11ah.

8.3.1. Common changes in the transmitter and receiver

There are common blocks in the transmitter and receiver. In this case, they are two WX GUI Chooser. The first one, it is the block with the ID is freq. The changes are in Default Value, Choices and Labels. In choices, the centre frequencies, of the channelization for US and Europe (in the section 6.3.5. Channelization), are set in this manner. But as it is explained in the section 6.3.1. OFDM, in the standard IEEE 802.11ah there are two transmission modes there are the transmission mode of 1 MHz channel bandwidth and the transmission mode of 2 MHz and greaters channel bandwidth. Thus, the centre frequencies below to 1 MHz , 2 MHz, 4 MHz. 8 MHz and 16 MHz channel bandwidth.

Choices:

[863500000.0, 864000000.0, 864500000.0, 865500000.0, 866000000.0, 866500000.0, 867500000.0, 902500000.0, 903000000.0, 903500000.0, 904500000.0, 905000000.0, 905500000.0, 906000000.0, 906500000.0, 907000000.0, 907500000.0, 908000000.0, 908500000.0, 909000000.0, 909500000.0, 910000000.0, 910500000.0, 911000000.0, 911500000.0, 912500000.0, 913000000.0, 913500000.0, 914000000.0, 914500000.0, 915000000.0, 915500000.0, 916000000.0, 916500000.0, 917000000.0, 917500000.0, 918000000.0, 918500000.0, 919000000.0, 919500000.0, 920000000.0, 920500000.0, 921000000.0, 921500000.0, 922000000.0, 922500000.0, 923000000.0, 923500000.0, 924000000.0, 924500000.0, 925000000.0, 925500000.0, 926000000.0, 926500000.0, 927000000.0, 927500000.0]

Labels have been changed according to the frequencies and the standard 802.11ah. They have put in this way.

Labels:

[' 1 | 8635.0 | 11ah', ' 2 | 8640.0 | 11ah', ' 3 | 8645.0 | 11ah', ' 4 | 8655.0 | 11ah', ' 5 | 8660.0 | 11ah', ' 6 | 8665.0 | 11ah', ' 7 | 8675.0 | 11ah', ' 8 | 9025.0 | 11ah', ' 9 | 9030.0 | 11ah', ' 10 | 9035.0 | 11ah', ' 11 | 9045.0 | 11ah', ' 12 | 9050.0 | 11ah', ' 13 | 9055.0 | 11ah', ' 14 | 9060.0 | 11ah', ' 15 | 9065.0 | 11ah', ' 16 | 9070.0 | 11ah', ' 17 |

9075.0 | 11ah', ' 18 | 9080.0 | 11ah', ' 19 | 9085.0 | 11ah', ' 20 | 9090.0 | 11ah', ' 21 | 9095.0 | 11ah', ' 22 | 9100.0 | 11ah', ' 23 | 9105.0 | 11ah', ' 24 | 9110.0 | 11ah', ' 25 | 9115.0 | 11ah', ' 26 | 9125.0 | 11ah', ' 27 | 9130.0 | 11ah', ' 28 | 9135.0 | 11ah', ' 29 | 9140.0 | 11ah', ' 30 | 9145.0 | 11ah', ' 31 | 9150.0 | 11ah', ' 32 | 9155.0 | 11ah', ' 33 | 9160.0 | 11ah', ' 34 | 9165.0 | 11ah', ' 35 | 9170.0 | 11ah', ' 36 | 9175.0 | 11ah', ' 37 | 9180.0 | 11ah', ' 38 | 9185.0 | 11ah', ' 39 | 9190.0 | 11ah', ' 40 | 9195.0 | 11ah', ' 41 | 9200.0 | 11ah', ' 42 | 9205.0 | 11ah', ' 43 | 9210.0 | 11ah', ' 44 | 9215.0 | 11ah', ' 45 | 9220.0 | 11ah', ' 46 | 9225.0 | 11ah', ' 47 | 9230.0 | 11ah', ' 48 | 9235.0 | 11ah', ' 49 | 9240.0 | 11ah', ' 50 | 9245.0 | 11ah', ' 51 | 9250.0 | 11ah', ' 52 | 9255.0 | 11ah', ' 53 | 9260.0 | 11ah', ' 54 | 9265.0 | 11ah', ' 55 | 9270.0 | 11ah', ' 56 | 9275.0 | 11ah']

The second change is in the block with the ID is samp_rate. The changes are in Default Value, Choices and Labels. These changes are according to the standard 802.11ah. As is explained in the section 6.3.1. OFDM, there are five options, 1 MHz channel bandwidth, 2 MHz channel bandwidth, 4 MHz channel bandwidth, 8 MHz channel bandwidth and 16 MHz channel bandwidth instead of 10 MHz and 20 MHz that it is according to the standard 802.11 a/g/p. The default value is 2 MHz.

8.3.2. Changes in the transmitter

In the block with the ID is encoding (WX GUI Chooser is the type of the block). The change is to remove a modulation and add two more, according with the Table 2 in the section 6.3.1. OFDM. In the parameter Choices is added the number 8, so now there are nine types of modulations.

Choices: [0, 1, 2, 3, 4, 5, 6, 7, 8]

In Labels, the modulation that is removed is BPSK 3/4 and the two modulations that are added are 64 QAM 5/6 and 256 QAM being as follows.

Labels: ["BPSK 1/2", "QPSK 1/2", "QPSK 3/4", "16QAM 1/2", "16QAM 3/4", "64QAM 2/3", "64QAM 3/4", "64QAM 5/6", "256QAM 3/4"]

In the block Socket PDU, the parameter MTU is changed to 7991 instead of 10000. Because in the section 6.4.1. Packet Size, the size of the MTU appears there.

8.3.3. Changes in wifi_phy_hier

In OFDM Carrier Allocator block is changed the parameter occupied carriers according to [28] and [29]. For the standard IEEE 802.11 a/g the range goes from -26 to 26. For the standard IEEE 802.11ah the range goes from -28 to 28.

Occupied Carriers: (range(-28, -21) + range(-20, -7) + range(-6, 0) + range(1, 7) + range(8, 21) + range(22, 29),)

The remaining changes are in C++ codes.

In the code ofdm_mapper.h, (Desktop/gr-ieee802-11/include/ieee802-11/) the modulation BPSK 3/4 is deleted and the modulations 64QAM 5/6 and 256QAM 3/4 are added in this way:

```
QAM64_5_6 = 7,

QAM256_3_4 =8,
```

In the code chunks_to_symbols_impl.cc (Desktop/gr-ieee802-11/lib/) the modulations 64QAM 5/6 and 256QAM 3/4 as cases, in this manner:

```
case QAM64_5_6:

case QAM256_3_4:

mapping = QAM256;

break;
```

The coordinates of the symbols of 256 QAM modulation are added. First, it is done a code matlab to have the coordinates. The coordinates are the following, in the same way as in the code:

```
const gr_complex chunks_to_symbols_impl::QAM256[256] = {

            gr_complex(-1.1538, -1.1538), gr_complex(-1.1538, -1.0000),

            gr_complex(-1.1538, -0.8462), gr_complex(-1.1538, -0.6923),

            gr_complex(-1.1538, -0.5385), gr_complex(-1.1538, -0.3846),

            gr_complex(-1.1538, -0.2308), gr_complex(-1.1538, -0.0769),

            gr_complex(-1.1538, 1.1538), gr_complex(-1.1538, 1.0000),

            gr_complex(-1.1538, 0.8462), gr_complex(-1.1538, 0.6923),

            gr_complex(-1.1538, 0.5385), gr_complex(-1.1538, 0.3846),
            gr_complex(-1.1538, 0.2308), gr_complex(-1.1538, 0.0769),

            gr_complex(-1.0000, -1.1538), gr_complex(-1.0000, -1.0000),

            gr_complex(-1.0000, -0.8462), gr_complex(-1.0000, -0.6923),

            gr_complex(-1.0000, -0.5385), gr_complex(-1.0000, -0.3846),

            gr_complex(-1.0000, -0.2308), gr_complex(-1.0000, -0.0769),

            gr_complex(-1.0000, 1.1538), gr_complex(-1.0000, 1.0000),

            gr_complex(-1.0000, 0.8462), gr_complex(-1.0000, 0.6923),

            gr_complex(-1.0000, 0.5385), gr_complex(-1.0000, 0.3846),

            gr_complex(-1.0000, 0.2308), gr_complex(-1.0000, 0.0769),

            gr_complex(-0.8462, -1.1538), gr_complex(-0.8462, -1.0000),

            gr_complex(-0.8462, -0.8462), gr_complex(-0.8462, -0.6923),

            gr_complex(-0.8462, -0.5385), gr_complex(-0.8462, -0.3846),

            gr_complex(-0.8462, -0.2308), gr_complex(-0.8462, -0.0769),

            gr_complex(-0.8462, 1.1538), gr_complex(-0.8462, 1.0000),

            gr_complex(-0.8462, 0.8462), gr_complex(-0.8462, 0.6923),

            gr_complex(-0.8462, 0.5385), gr_complex(-0.8462, 0.3846),

            gr_complex(-0.8462, 0.2308), gr_complex(-0.8462, 0.0769),

            gr_complex(-0.6923, -1.1538), gr_complex(-0.6923, -1.0000),

            gr_complex(-0.6923, -0.8462), gr_complex(-0.6923, -0.6923),

            gr_complex(-0.6923, -0.5385), gr_complex(-0.6923, -0.3846),

            gr_complex(-0.6923, -0.2308), gr_complex(-0.6923, -0.0769),

            gr_complex(-0.6923, 1.1538), gr_complex(-0.6923, 1.0000),

            gr_complex(-0.6923, 0.8462), gr_complex(-0.6923, 0.6923),
            gr_complex(-0.6923, 0.5385), gr_complex(-0.6923, 0.3846),
            gr_complex(-0.6923, 0.2308), gr_complex(-0.6923, 0.0769),
            gr_complex(-0.5385, -1.1538), gr_complex(-0.5385, -1.0000),
            gr_complex(-0.5385, -0.8462), gr_complex(-0.5385, -0.6923),
            gr_complex(-0.5385, -0.5385), gr_complex(-0.5385, -0.3846),
            gr_complex(-0.5385, -0.2308), gr_complex(-0.5385, -0.0769),
            gr_complex(-0.5385, 1.1538), gr_complex(-0.5385, 1.0000),
            gr_complex(-0.5385, 0.8462), gr_complex(-0.5385, 0.6923),
            gr_complex(-0.5385, 0.5385), gr_complex(-0.5385, 0.3846),
            gr_complex(-0.5385, 0.2308), gr_complex(-0.5385, 0.0769),
            gr_complex(-0.3846, -1.1538), gr_complex(-0.3846, -1.0000),
            gr_complex(-0.3846, -0.8462), gr_complex(-0.3846, -0.6923),
            gr_complex(-0.3846, -0.5385), gr_complex(-0.3846, -0.3846),
            gr_complex(-0.3846, -0.2308), gr_complex(-0.3846, -0.0769),
            gr_complex(-0.3846, 1.1538), gr_complex(-0.3846, 1.0000),
            gr_complex(-0.3846, 0.8462), gr_complex(-0.3846, 0.6923),
            gr_complex(-0.3846, 0.5385), gr_complex(-0.3846, 0.3846),
            gr_complex(-0.3846, 0.2308), gr_complex(-0.3846, 0.0769),
            gr_complex(-0.2308, -1.1538), gr_complex(-0.2308, -1.0000),
            gr_complex(-0.2308, -0.8462), gr_complex(-0.2308, -0.6923),
            gr_complex(-0.2308, -0.5385), gr_complex(-0.2308, -0.3846),
            gr_complex(-0.2308, -0.2308), gr_complex(-0.2308, -0.0769),
            gr_complex(-0.2308, 1.1538), gr_complex(-0.2308, 1.0000),
            gr_complex(-0.2308, 0.8462), gr_complex(-0.2308, 0.6923),
            gr_complex(-0.2308, 0.5385), gr_complex(-0.2308, 0.3846),
            gr_complex(-0.2308, 0.2308), gr_complex(-0.2308, 0.0769),
            gr_complex(-0.0769, -1.1538), gr_complex(-0.0769, -1.0000),
            gr_complex(-0.0769, -0.8462), gr_complex(-0.0769, -0.6923),
            gr_complex(-0.0769, -0.5385), gr_complex(-0.0769, -0.3846),
            gr_complex(-0.0769, -0.2308), gr_complex(-0.0769, -0.0769),
            gr_complex(-0.0769, 1.1538), gr_complex(-0.0769, 1.0000),
            gr_complex(-0.0769, 0.8462), gr_complex(-0.0769, 0.6923),
            gr_complex(-0.0769, 0.5385), gr_complex(-0.0769, 0.3846),
            gr_complex(-0.0769, 0.2308), gr_complex(-0.0769, 0.0769),
            gr_complex(1.1538, -1.1538), gr_complex(1.1538, -1.0000),
            gr_complex(1.1538, -0.8462), gr_complex(1.1538, -0.6923),
            gr_complex(1.1538, -0.5385), gr_complex(1.1538, -0.3846),
            gr_complex(1.1538, -0.2308), gr_complex(1.1538, -0.0769),
            gr_complex(1.1538, 1.1538), gr_complex(1.1538, 1.0000),
            gr_complex(1.1538, 0.8462), gr_complex(1.1538, 0.6923),
            gr_complex(1.1538, 0.5385), gr_complex(1.1538, 0.3846),
            gr_complex(1.1538, 0.2308), gr_complex(1.1538, 0.0769),
            gr_complex(1.0000, -1.1538), gr_complex(1.0000, -1.0000),
            gr_complex(1.0000, -0.8462), gr_complex(1.0000, -0.6923),
            gr_complex(1.0000, -0.5385), gr_complex(1.0000, -0.3846),
            gr_complex(1.0000, -0.2308), gr_complex(1.0000, -0.0769),
            gr_complex(1.0000, 1.1538), gr_complex(1.0000, 1.0000),

            gr_complex(1.0000, 0.8462), gr_complex(1.0000, 0.6923),

            gr_complex(1.0000, 0.5385), gr_complex(1.0000, 0.3846),

            gr_complex(1.0000, 0.2308), gr_complex(1.0000, 0.0769),

            gr_complex(0.8462, -1.1538), gr_complex(0.8462, -1.0000),

            gr_complex(0.8462, -0.8462), gr_complex(0.8462, -0.6923),

            gr_complex(0.8462, -0.5385), gr_complex(0.8462, -0.3846),

            gr_complex(0.8462, -0.2308), gr_complex(0.8462, -0.0769),

            gr_complex(0.8462, 1.1538), gr_complex(0.8462, 1.0000),

            gr_complex(0.8462, 0.8462), gr_complex(0.8462, 0.6923),

            gr_complex(0.8462, 0.5385), gr_complex(0.8462, 0.3846),

            gr_complex(0.8462, 0.2308), gr_complex(0.8462, 0.0769),

            gr_complex(0.6923, -1.1538), gr_complex(0.6923, -1.0000),

            gr_complex(0.6923, -0.8462), gr_complex(0.6923, -0.6923),

            gr_complex(0.6923, -0.5385), gr_complex(0.6923, -0.3846),

            gr_complex(0.6923, -0.2308), gr_complex(0.6923, -0.0769),

            gr_complex(0.6923, 1.1538), gr_complex(0.6923, 1.0000),

            gr_complex(0.6923, 0.8462), gr_complex(0.6923, 0.6923),

            gr_complex(0.6923, 0.5385), gr_complex(0.6923, 0.3846),

            gr_complex(0.6923, 0.2308), gr_complex(0.6923, 0.0769),

            gr_complex(0.5385, -1.1538), gr_complex(0.5385, -1.0000),

            gr_complex(0.5385, -0.8462), gr_complex(0.5385, -0.6923),

            gr_complex(0.5385, -0.5385), gr_complex(0.5385, -0.3846),
            gr_complex(0.5385, -0.2308), gr_complex(0.5385, -0.0769),
            gr_complex(0.5385, 1.1538), gr_complex(0.5385, 1.0000),
            gr_complex(0.5385, 0.8462), gr_complex(0.5385, 0.6923),
            gr_complex(0.5385, 0.5385), gr_complex(0.5385, 0.3846),
            gr_complex(0.5385, 0.2308), gr_complex(0.5385, 0.0769),
            gr_complex(0.3846, -1.1538), gr_complex(0.3846, -1.0000),
            gr_complex(0.3846, -0.8462), gr_complex(0.3846, -0.6923),
            gr_complex(0.3846, -0.5385), gr_complex(0.3846, -0.3846),
            gr_complex(0.3846, -0.2308), gr_complex(0.3846, -0.0769),
            gr_complex(0.3846, 1.1538), gr_complex(0.3846, 1.0000),
            gr_complex(0.3846, 0.8462), gr_complex(0.3846, 0.6923),
            gr_complex(0.3846, 0.5385), gr_complex(0.3846, 0.3846),
            gr_complex(0.3846, 0.2308), gr_complex(0.3846, 0.0769),
            gr_complex(0.2308, -1.1538), gr_complex(0.2308, -1.0000),
            gr_complex(0.2308, -0.8462), gr_complex(0.2308, -0.6923),
            gr_complex(0.2308, -0.5385), gr_complex(0.2308, -0.3846),
            gr_complex(0.2308, -0.2308), gr_complex(0.2308, -0.0769),
            gr_complex(0.2308, 1.1538), gr_complex(0.2308, 1.0000),
            gr_complex(0.2308, 0.8462), gr_complex(0.2308, 0.6923),
            gr_complex(0.2308, 0.5385), gr_complex(0.2308, 0.3846),
            gr_complex(0.2308, 0.2308), gr_complex(0.2308, 0.0769),
            gr_complex(0.0769, -1.1538), gr_complex(0.0769, -1.0000),
            gr_complex(0.0769, -0.8462), gr_complex(0.0769, -0.6923),
            gr_complex(0.0769, -0.5385), gr_complex(0.0769, -0.3846),
            gr_complex(0.0769, -0.2308), gr_complex(0.0769, -0.0769),
            gr_complex(0.0769, 1.1538), gr_complex(0.0769, 1.0000),
            gr_complex(0.0769, 0.8462), gr_complex(0.0769, 0.6923),
            gr_complex(0.0769, 0.5385), gr_complex(0.0769, 0.3846),
            gr_complex(0.0769, 0.2308), gr_complex(0.0769, 0.0769)};
```

In ofdm_decode_mac.cc, the numbers from 0 to 255 of 256 QAM modulation are added and the case QAM64_5_6 is added too.

```
// qam256
int qam256_bits[] = {
         0,  1,  2,  3,  4,  5,  6,  7,  8,  9,
        10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
        20, 21, 22, 23, 24, 25, 26, 27, 28, 29,
        30, 31, 32, 33, 34, 35, 36, 37, 38, 39,
        40, 41, 42, 43, 44, 45, 46, 47, 48, 49,
        50, 51, 52, 53, 54, 55, 56, 57, 58, 59,
        60, 61, 62, 63, 64, 65, 66, 67, 68, 69,
        70, 71, 72, 73, 74, 75, 76, 77, 78, 79,
        80, 81, 82, 83, 84, 85, 86, 87, 88, 89,
        90, 91, 92, 93, 94, 95, 96, 97, 98, 99,
        100, 101, 102, 103, 104, 105, 106, 107, 108, 109,
        110, 111, 112, 113, 114, 115, 116, 117, 118, 119,
        120, 121, 122, 123, 124, 125, 126, 127, 128, 129,
        130, 131, 132, 133, 134, 135, 136, 137, 138, 139,
        140, 141, 142, 143, 144, 145, 146, 147, 148, 149,
        150, 151, 152, 153, 154, 155, 156, 157, 158, 159,
        160, 161, 162, 163, 164, 165, 166, 167, 168, 169,
        170, 171, 172, 173, 174, 175, 176, 177, 178, 179,
        180, 181, 182, 183, 184, 185, 186, 187, 188, 189,
        190, 191, 192, 193, 194, 195, 196, 197, 198, 199,
        200, 201, 202, 203, 204, 205, 206, 207, 208, 209,
        210, 211, 212, 213, 214, 215, 216, 217, 218, 219,
        220, 221, 222, 223, 224, 225, 226, 227, 228, 229,
        230, 231, 232, 233, 234, 235, 236, 237, 238, 239,
        240, 241, 242, 243, 244, 245, 246, 247, 248, 249,
        250, 251, 252, 253, 254, 255};
qam64.set(cvec(QAM256_D, 256), ivec(qam256_bits, 256));
case QAM256_3_4:
bits = to_vec(qam256.demodulate_bits(symbols));
break;
```

It is necessary to add 256 QAM 3/4 modulation to the puncture matrix that it is written in the code. It had to search information about the puncture matrix to can add one for 64 QAM 5/6. Thus, this puncture matrix is added according to [31].

```
case QAM64_5_6:
puncture_matrix = "1 1 0 1 0; 1 0 1 0 1;";
break;
```

And qam256 variable is declared at the end of the code.

```
Modulator<std::complex<double> > qam256;
```

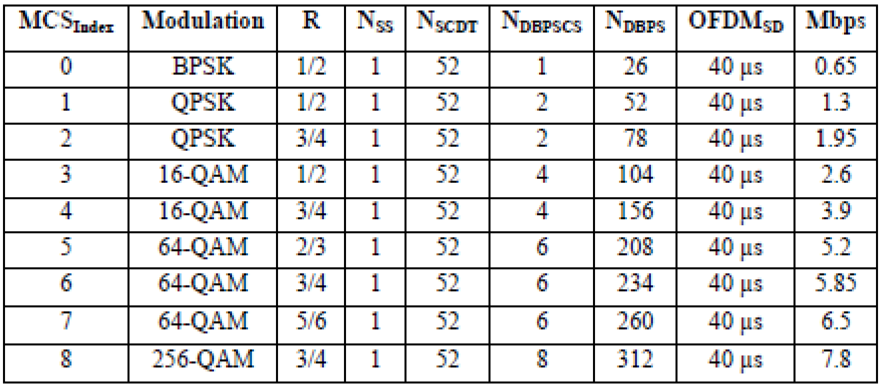
Furthermore the QAM256_D variable has to be declared in other code, in utils.cc (Desktop/gr-ieee802-11/lib/). For declaring this variable, it is necessary to add the coordinates of the symbols of 256 QAM modulation again.

```
const std::complex<double> QAM256_D[256] = {

            std::complex<double>(-1.1538, -1.1538), ….

            …., std::complex<double>(0.0769, 0.0769)};
```

In this code, there are some parameters that they need to be changed because of the standard IEEE 802.11ah. These parameters are N_{BPSC}, N_{CBPS} and N_{DBPS}. The parameter values are achieve according to the following table (columns N_{DBPSCS} and N_{DBPS}) and to this equation N_{DBPS}= r · N_{CBPS} where r is the code rate.



MCS_{Index}	Modulation	R	N_{SS}	N_{SCDT}	N_{DBPSCS}	N_{DBPS}	$OFDM_{SD}$	Mbps
0	BPSK	1/2	1	52	1	26	40 µs	0.65
1	QPSK	1/2	1	52	2	52	40 µs	1.3
2	QPSK	3/4	1	52	2	78	40 µs	1.95
3	16-QAM	1/2	1	52	4	104	40 µs	2.6
4	16-QAM	3/4	1	52	4	156	40 µs	3.9
5	64-QAM	2/3	1	52	6	208	40 µs	5.2
6	64-QAM	3/4	1	52	6	234	40 µs	5.85
7	64-QAM	5/6	1	52	6	260	40 µs	6.5
8	256-QAM	3/4	1	52	8	312	40 µs	7.8

Table 4. [11]

```
case BPSK_1_2:

    n_bpsc = 1;

    n_cbps = 52;

    n_dbps = 26;

    rate_field = 0x0D;

    break;   case QPSK_1_2:

    n_bpsc = 2;

    n_cbps = 104;
    n_dbps = 52;

    rate_field = 0x05;

break;   case QPSK_3_4:

    n_bpsc = 2;

    n_cbps = 104;

    n_dbps = 78;

    rate_field = 0x07;

break;   case QAM16_1_2:

    n_bpsc = 4;

    n_cbps = 208;

    n_dbps = 104;

    rate_field = 0x09;

break;

case QAM16_3_4:

    n_bpsc = 4;

    n_cbps = 208;

    n_dbps = 156;

    rate_field = 0x0B;

break;

case QAM64_2_3:

    n_bpsc = 6;

    n_cbps = 312;

    n_dbps = 208;
    rate_field = 0x01;

break;

case QAM64_3_4:

    n_bpsc = 6;

    n_cbps = 312;

    n_dbps = 234;

    rate_field = 0x03;

break;

 case QAM64_5_6:

    n_bpsc = 6;

    n_cbps = 312;

    n_dbps = 260;

    rate_field = 0x02; [2]

break;

case QAM256_3_4:

    n_bpsc = 8;

    n_cbps = 416;

    n_dbps = 312;

    rate_field = 0x04; [2]

    break;
```

[2] For these two modulations (64 QAM 5/6 and 256 QAM 3/4) there is no information about the rate_field parameter, so it is taken two available values. When the information of this parameter will be updated and if they are different values, this parameter has to be changed.

In this same code (utils.cc), there is a void named puncturing and is added the case QAM256_3_4 in a loop if already programmed. It is necessary to program another loop for the case QAM64_5_6. Cyclic Redundancy Codes (CRCs) are widely used in many wired and wireless standard specifications as a means of error detection. As such, it is only used for error detection, and not error correction. [32]

```
case QAM64_5_6: 3

    if (!(mod = = 0 || mod = = 3 || mod = = 5)){

            *out = in [i];

            out++;

    } break;
```

In the code chunks_to_symbols_impl.h (Desktop/gr-ieee802-11/lib/) the gr_complex QAM256 variable is declared.

```
static const gr_complex QAM256[256];
```

In the code utils.c (Desktop/gr-ieee802-11/lib/) the const std::complex<double> QAM256_D is declared likes this.

```
extern const std::complex<double>  QAM256_D[256];
```

8.3.4. Changes in loopback

In the block with the ID is encoding (WX GUI Chooser is the type of the block). The change is to remove a modulation and add two more, according with the Table 2 in the section 6.3.1. OFDM. In the parameter Choices is added the number 8, so now there are nine types of modulations.

Choices: [0, 1, 2, 3, 4, 5, 6, 7, 8]

[3] For this modulation (64 QAM 5/6) there is no information about the puncturing, so it is taken a CRC code available. When the information of this parameter will be updated and if it is different CRC code, this parameter has to be changed.

In Labels, the modulation that is removed is BPSK 3/4 and the two modulations that are added are 64 QAM 5/6 and 256 QAM being as follows.

Labels: ["BPSK 1/2", "QPSK 1/2", "QPSK 3/4", "16QAM 1/2", "16QAM 3/4", "64QAM 2/3", "64QAM 3/4", "64QAM 5/6", "256QAM 3/4"]

9. Results

9.1. Transmitter

To have results in the transmitter instead of the last block UHD: USRP Sink the last block is a WX GUI FFT Sink. The results are the FFT's of the different channel bandwidth i.e. 1 MHz channel bandwidth, 2 MHz channel bandwidth, 4 MHz channel bandwidth, 8 MHz channel bandwidth and 16 MHz channel bandwidth.

According to the following two images, the center frequency is changed. Especially in the case of 16 MHz channel bandwidth because there is only one channel.

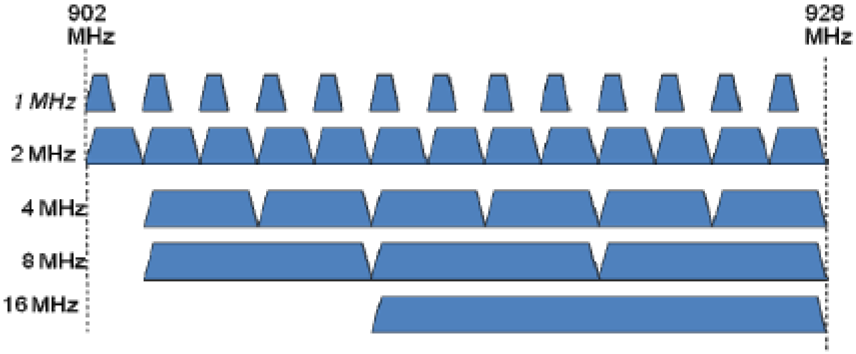


Figure 27. Channelization for US [9]

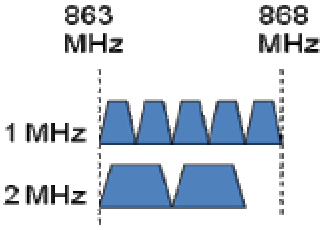


Figure 28. Channelization for Europe [9]

The result of the FFT of the transmission mode of 1 MHz channel is the following. And its center frequency is $f_c = 923.5\ MHz$. It is seen in the figure that the bandwidth is 1 MHz, the frequency starts in 923 MHz and it ends 924 MHz.

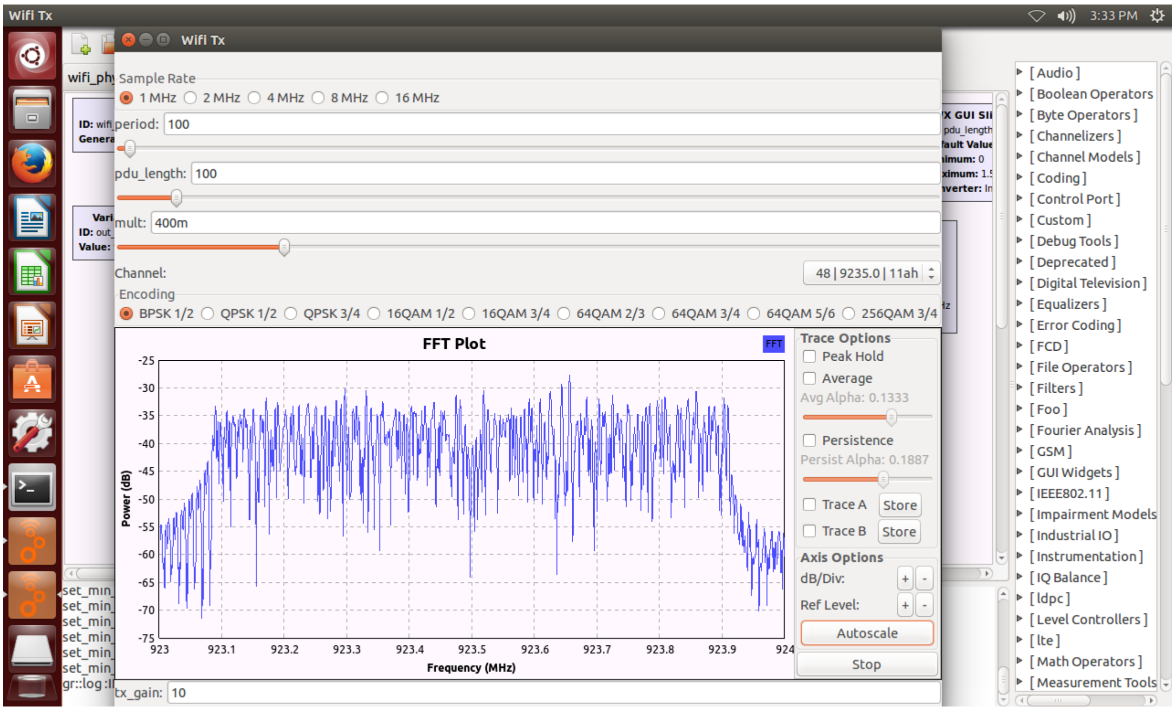

Sample Rate
● 1 MHz ○ 2 MHz ○ 4 MHz ○ 8 MHz ○ 16 MHz
period: 100
pdu_length: 100
mult: 400m
Channel: 48 | 9235.0 | 11ah
Encoding
● BPSK 1/2 ○ QPSK 1/2 ○ QPSK 3/4 ○ 16QAM 1/2 ○ 16QAM 3/4 ○ 64QAM 2/3 ○ 64QAM 3/4 ○ 64QAM 5/6 ○ 256QAM 3/4
Trace Options
□ Peak Hold
□ Average
Avg Alpha: 0.1333
□ Persistence
Persist Alpha: 0.1887
□ Trace A
□ Trace B
Axis Options
dB/Div:
Ref Level:
tx_gain: 10

Figure 29. FFT 1 MHz

The result of the FFT of the transmission mode of 2 MHz is the following. And its center frequency is $f_c = 923\ MHz$. It is seen in the figure that the bandwidth is 1 MHz, the frequency starts in 922 MHz and it ends 924 MHz.

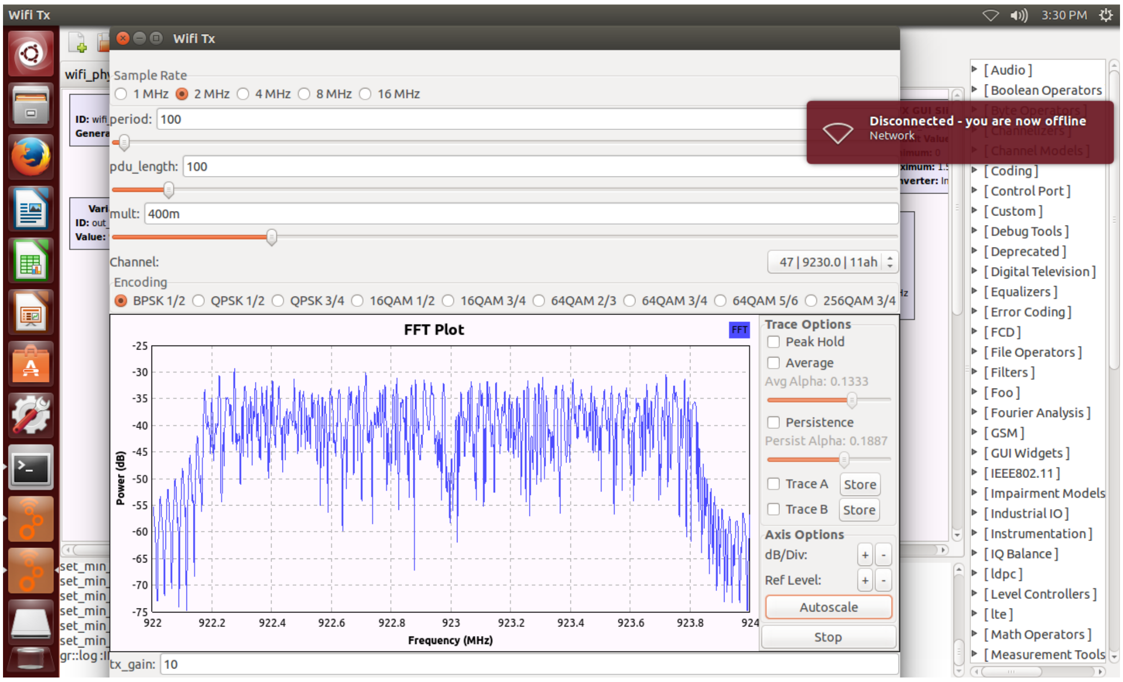
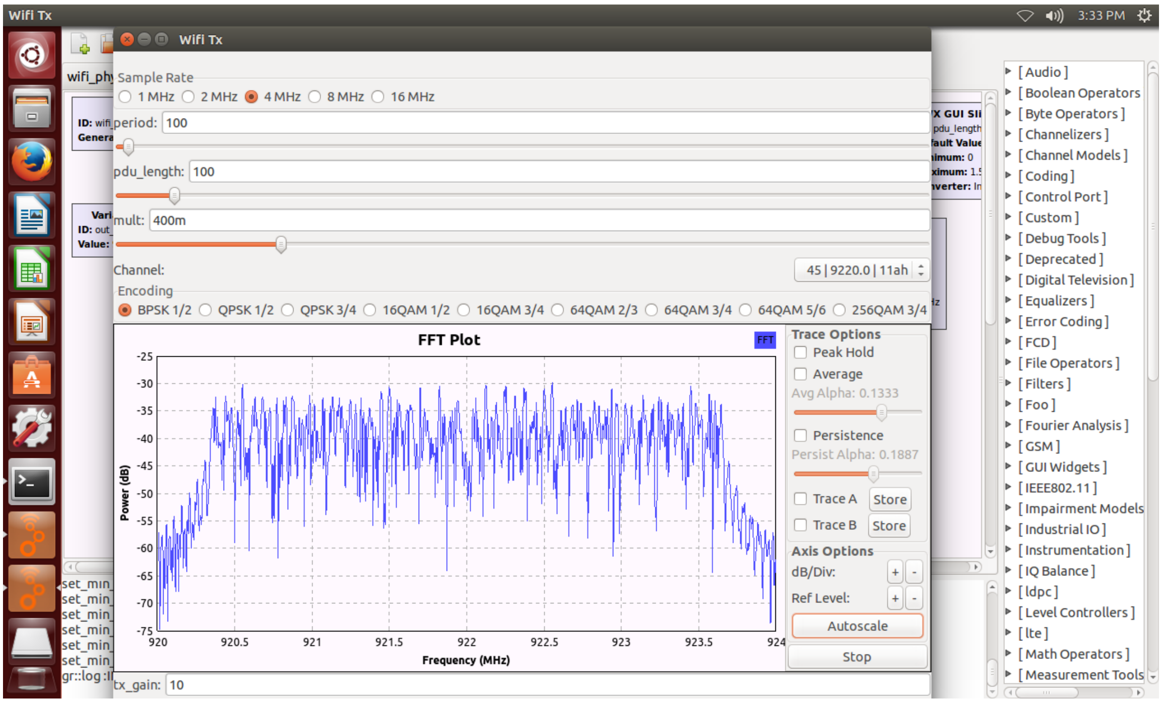

Sample Rate
○ 1 MHz ● 2 MHz ○ 4 MHz ○ 8 MHz ○ 16 MHz
period: 100
pdu_length: 100
mult: 400m
Channel: 47 | 9230.0 | 11ah
Encoding
● BPSK 1/2 ○ QPSK 1/2 ○ QPSK 3/4 ○ 16QAM 1/2 ○ 16QAM 3/4 ○ 64QAM 2/3 ○ 64QAM 3/4 ○ 64QAM 5/6 ○ 256QAM 3/4
Trace Options
□ Peak Hold
□ Average
Avg Alpha: 0.1333
□ Persistence
Persist Alpha: 0.1887
□ Trace A
□ Trace B
Axis Options
dB/Div:
Ref Level:
tx_gain: 10
Disconnected - you are now offline
Network

Figure 30. FFT 2 MHz

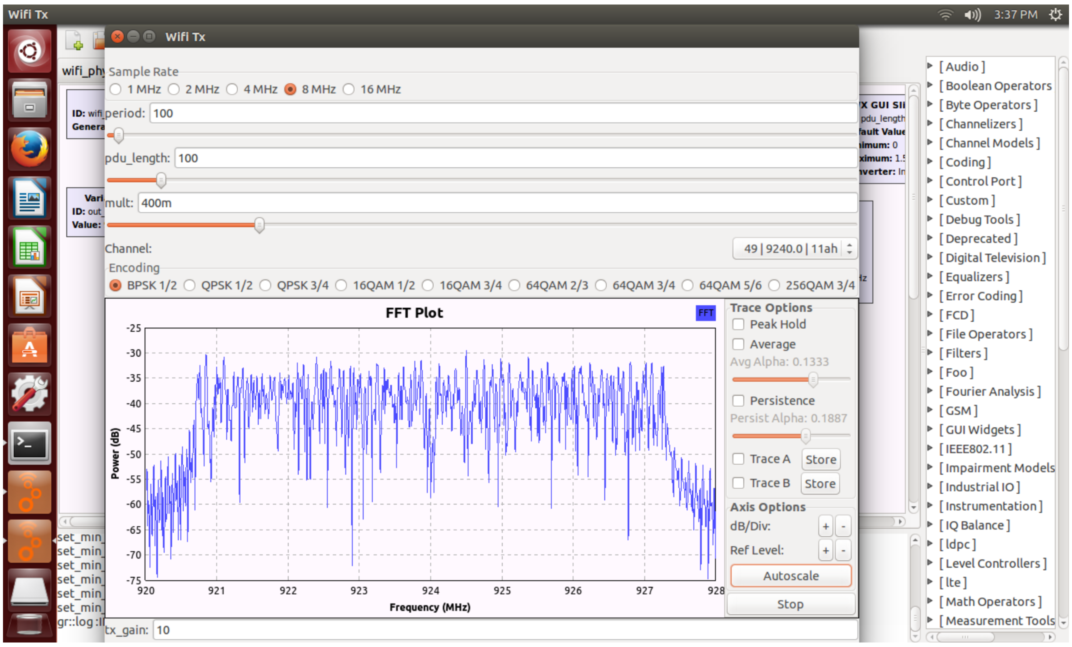
The result of the FFT of the transmission mode of 4 MHz is the following. And its center frequency is $f_c = 922\ MHz$. It is seen in the figure that the bandwidth is 4 MHz, the frequency starts in 920 MHz and it ends 924 MHz.



Sample Rate ○ 1 MHz ○ 2 MHz ● 4 MHz ○ 8 MHz ○ 16 MHz
period: 100
pdu_length: 100
mult: 400m
Channel: 45 | 9220.0 | 11ah
Encoding ● BPSK 1/2 ○ QPSK 1/2 ○ QPSK 3/4 ○ 16QAM 1/2 ○ 16QAM 3/4 ○ 64QAM 2/3 ○ 64QAM 3/4 ○ 64QAM 5/6 ○ 256QAM 3/4
Trace Options
□ Peak Hold
□ Average
Avg Alpha: 0.1333
□ Persistence
Persist Alpha: 0.1887
□ Trace A
□ Trace B
tx_gain: 10

Figure 31. FFT 4 MHz

The result of the FFT of the transmission mode of 8 MHz is the following. And its center frequency is $f_c = 924\ MHz$. It is seen in the figure that the bandwidth is 1 MHz, the frequency starts in 920 MHz and it ends 928 MHz. This is the last channel in the transmission mode of 8 MHz, in this transmission mode there are 3 channels.



Wifi Tx

Sample Rate: ○ 1 MHz ○ 2 MHz ○ 4 MHz ● 8 MHz ○ 16 MHz

period: 100

pdu_length: 100

mult: 400m

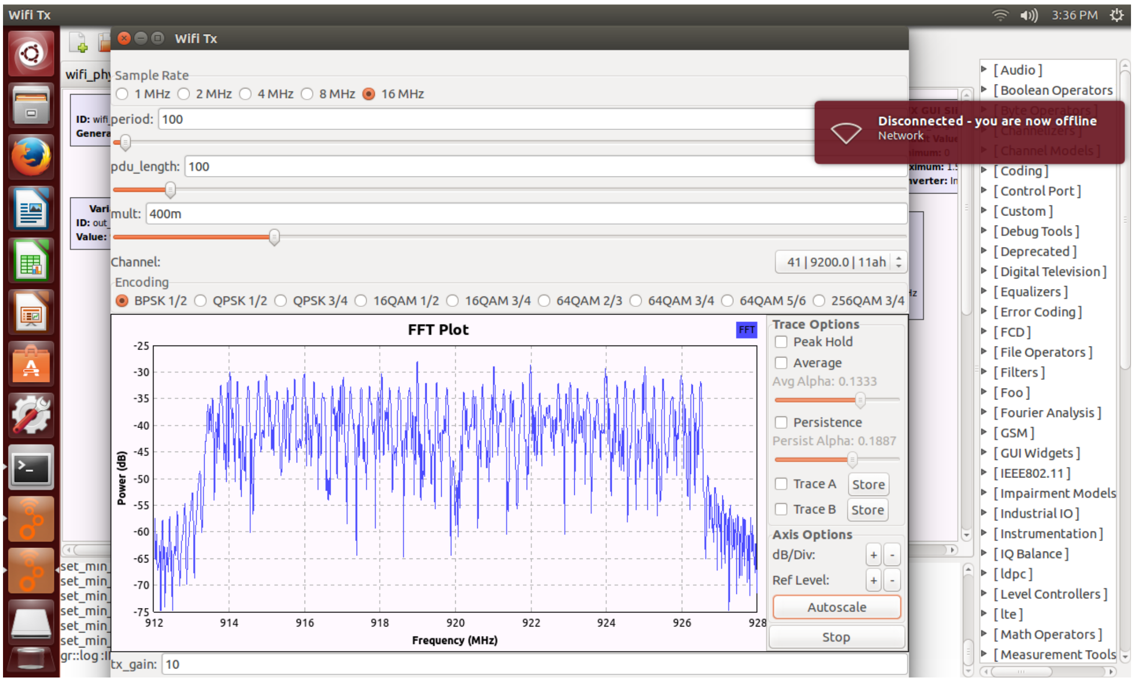
Channel: 49 | 9240.0 | 11ah

Encoding: ● BPSK 1/2 ○ QPSK 1/2 ○ QPSK 3/4 ○ 16QAM 1/2 ○ 16QAM 3/4 ○ 64QAM 2/3 ○ 64QAM 3/4 ○ 64QAM 5/6 ○ 256QAM 3/4

tx_gain: 10

Figure 32. FFT 8 MHz

The result of the FFT of the transmission mode of 16 MHz is the following. And its center frequency is $f_c = 920\ MHz$. It is seen in the figure that the bandwidth is 1 MHz, the frequency starts in 912 MHz and it ends 928 MHz. This is the only channel in the transmission mode of 16 MHz, as seen in the figure of the channelization of US.



Wifi Tx

Sample Rate: ○ 1 MHz ○ 2 MHz ○ 4 MHz ○ 8 MHz ● 16 MHz

period: 100

pdu_length: 100

mult: 400m

Channel: 41 | 9200.0 | 11ah

Encoding: ● BPSK 1/2 ○ QPSK 1/2 ○ QPSK 3/4 ○ 16QAM 1/2 ○ 16QAM 3/4 ○ 64QAM 2/3 ○ 64QAM 3/4 ○ 64QAM 5/6 ○ 256QAM 3/4

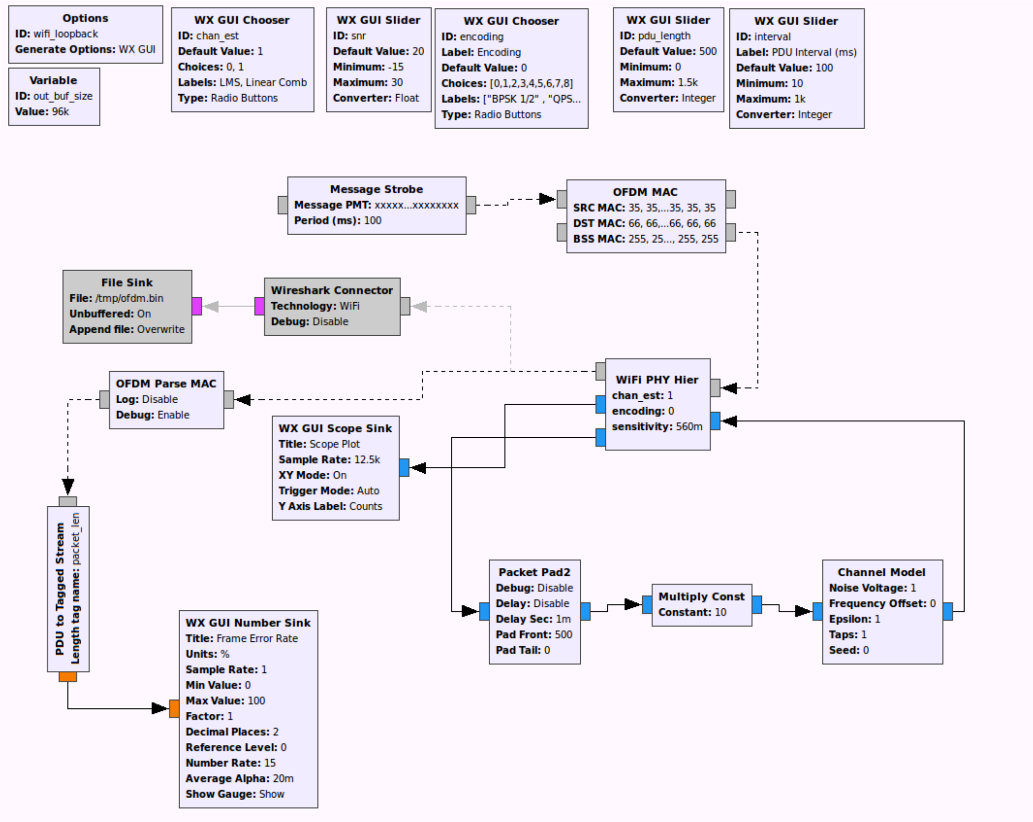
tx_gain: 10

Disconnected - you are now offline
Network

Figure 33 FFT 16 MHz

9.2. Loopback

The results of this flow graph are all the modulations i.e. BPSK 1/2, QPSK 1/2, QPSK 3/4, 16QAM 1/2, 16QAM 3/4, 64QAM 2/3, 64QAM 3/4, 64QAM 5/6 and 256QAM 3/4. First, it is seen the flow graph of the loopback. In the flow graph there are a WX GUI Scope Sink and WX GUI Number Sink. The first one it is to see the shape of the modulations. And, the second one, it is to see the frame error rate.



Options
ID: wifi_loopback
Generate Options: WX GUI

Variable
ID: out_buf_size
Value: 96k

WX GUI Chooser
ID: chan_est
Default Value: 1
Choices: 0, 1
Labels: LMS, Linear Comb
Type: Radio Buttons

WX GUI Slider
ID: snr
Default Value: 20
Minimum: -15
Maximum: 30
Converter: Float

WX GUI Chooser
ID: encoding
Label: Encoding
Default Value: 0
Choices: [0,1,2,3,4,5,6,7,8]
Labels: ["BPSK 1/2" , "QPS...
Type: Radio Buttons

WX GUI Slider
ID: pdu_length
Default Value: 500
Minimum: 0
Maximum: 1.5k
Converter: Integer

WX GUI Slider
ID: interval
Label: PDU Interval (ms)
Default Value: 100
Minimum: 10
Maximum: 1k
Converter: Integer

Message Strobe
Message PMT: xxxxx...xxxxxxxx
Period (ms): 100

OFDM MAC
SRC MAC: 35, 35,...35, 35, 35
DST MAC: 66, 66,...66, 66, 66
BSS MAC: 255, 25..., 255, 255

File Sink
File: /tmp/ofdm.bin
Unbuffered: On
Append file: Overwrite

Wireshark Connector
Technology: WiFi
Debug: Disable

OFDM Parse MAC
Log: Disable
Debug: Enable

WiFi PHY Hier
chan_est: 1
encoding: 0
sensitivity: 560m

WX GUI Scope Sink
Title: Scope Plot
Sample Rate: 12.5k
XY Mode: On
Trigger Mode: Auto
Y Axis Label: Counts

PDU to Tagged Stream
Length tag name: packet_len

WX GUI Number Sink
Title: Frame Error Rate
Units: %
Sample Rate: 1
Min Value: 0
Max Value: 100
Factor: 1
Decimal Places: 2
Reference Level: 0
Number Rate: 15
Average Alpha: 20m
Show Gauge: Show

Packet Pad2
Debug: Disable
Delay: Disable
Delay Sec: 1m
Pad Front: 500
Pad Tail: 0

Multiply Const
Constant: 10

Channel Model
Noise Voltage: 1
Frequency Offset: 0
Epsilon: 1
Taps: 1
Seed: 0

Figure 34. Flow graph of the Loopback

In the following figure, it is seen a BPSK modulation. The code rate used is 1/2. The points are concentrated around -1 and 1.

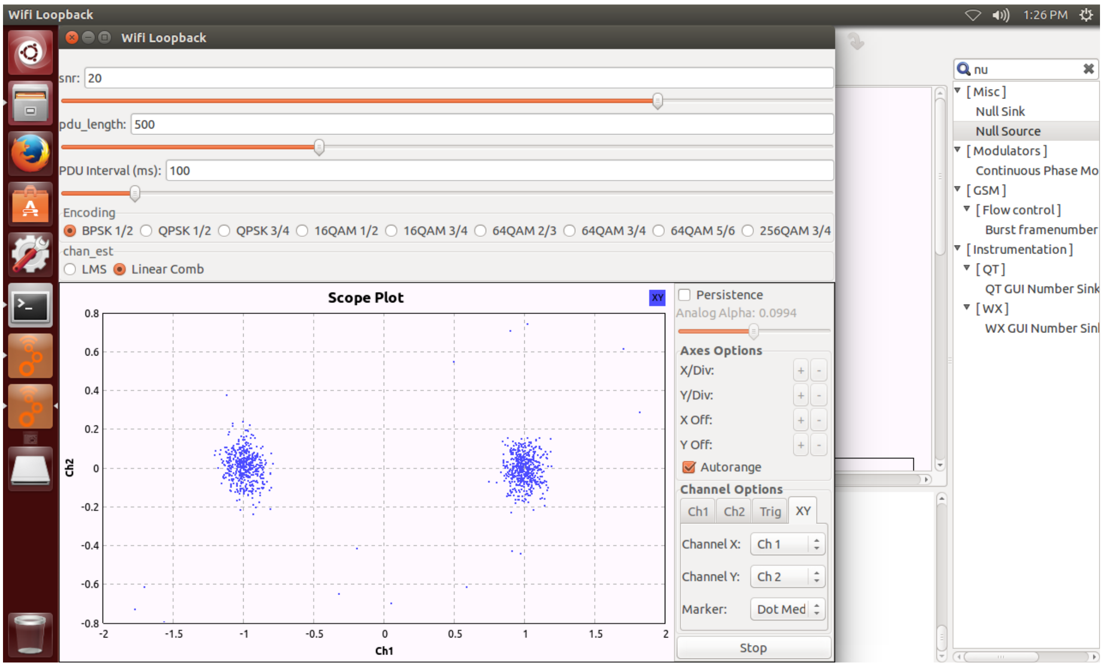

snr: 20
pdu_length: 500
PDU Interval (ms): 100
Encoding
● BPSK 1/2 ○ QPSK 1/2 ○ QPSK 3/4 ○ 16QAM 1/2 ○ 16QAM 3/4 ○ 64QAM 2/3 ○ 64QAM 3/4 ○ 64QAM 5/6 ○ 256QAM 3/4
chan_est
○ LMS ● Linear Comb
□ Persistence
Analog Alpha: 0.0994
☑ Autorange
Channel X: Ch 1
Channel Y: Ch 2
Marker: Dot Med

Figure 35. BPSK 1/2

In the next figure, it is seen a QPSK modulation. The code rate used is 1/2. The points are concentrated in four coordinates. To the x= $\pm\frac{\sqrt{2}}{2}$ and to the y= $\pm\frac{\sqrt{2}}{2}$.

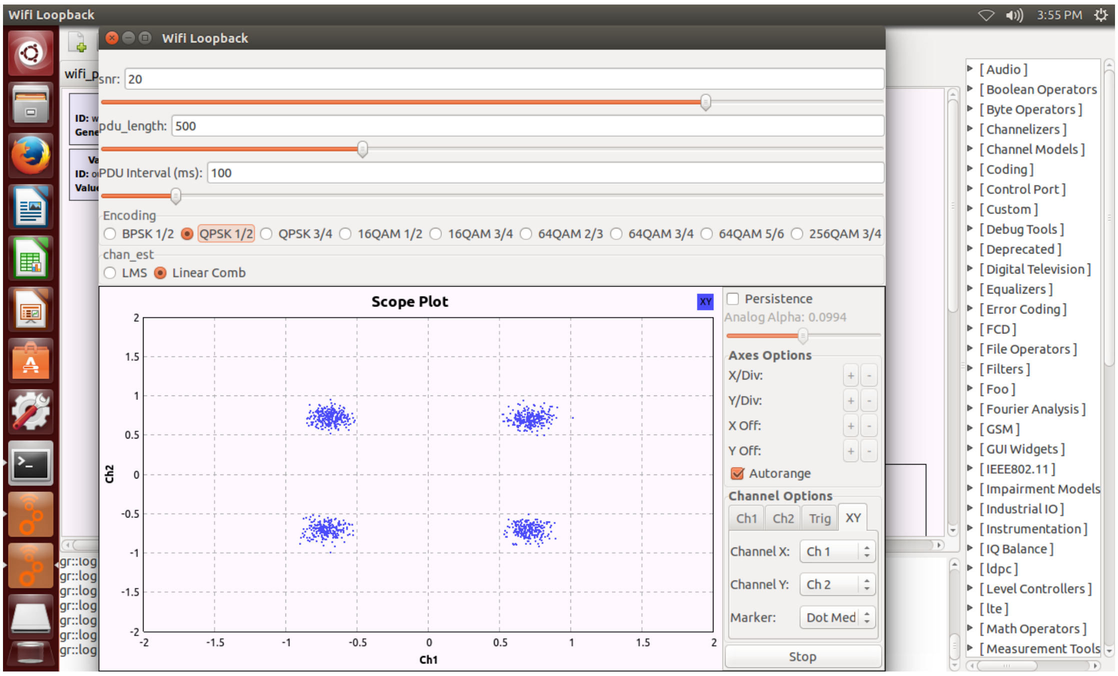
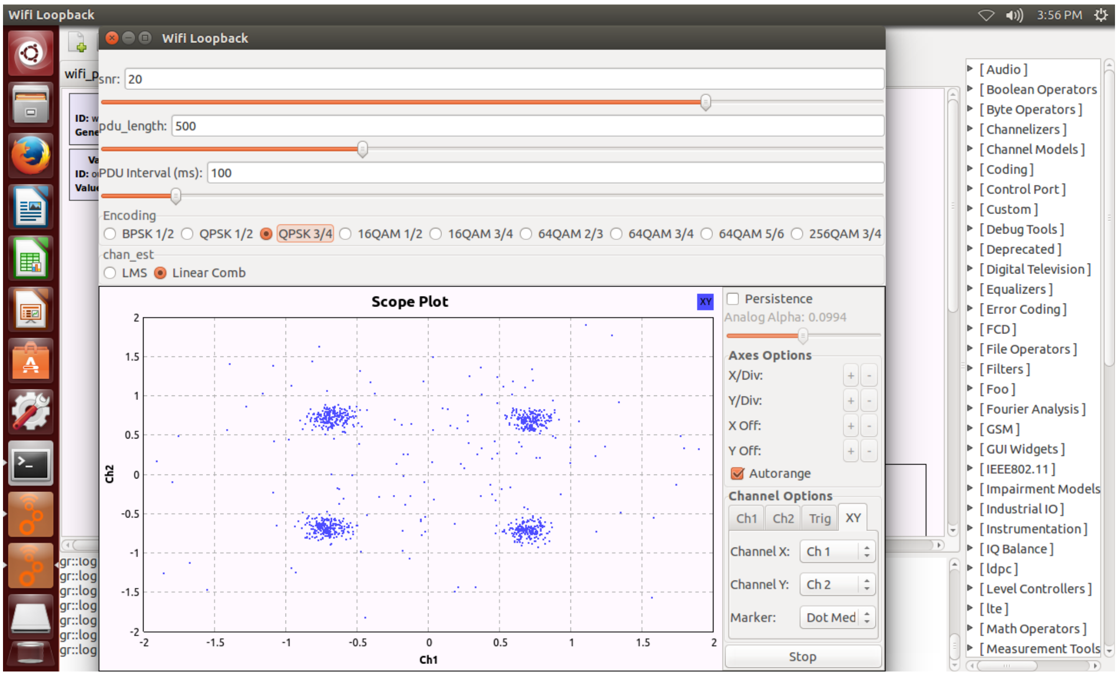

snr: 20
pdu_length: 500
PDU Interval (ms): 100
Encoding
○ BPSK 1/2 ● QPSK 1/2 ○ QPSK 3/4 ○ 16QAM 1/2 ○ 16QAM 3/4 ○ 64QAM 2/3 ○ 64QAM 3/4 ○ 64QAM 5/6 ○ 256QAM 3/4
chan_est
○ LMS ● Linear Comb
□ Persistence
Analog Alpha: 0.0994
☑ Autorange
Channel X: Ch 1
Channel Y: Ch 2
Marker: Dot Med

Figure 36. QPSK 1/2

The following modulation is a QPSK modulation. The code rate is 3/4. The points are concentrated in the same place than QPSK modulation and the code rate is 1/2. The points are more dispersed the puncturing matrix and the puncturing are different.



Wifi Loopback

snr: 20

pdu_length: 500

PDU Interval (ms): 100

Encoding
○ BPSK 1/2 ○ QPSK 1/2 ● QPSK 3/4 ○ 16QAM 1/2 ○ 16QAM 3/4 ○ 64QAM 2/3 ○ 64QAM 3/4 ○ 64QAM 5/6 ○ 256QAM 3/4

chan_est
○ LMS ● Linear Comb

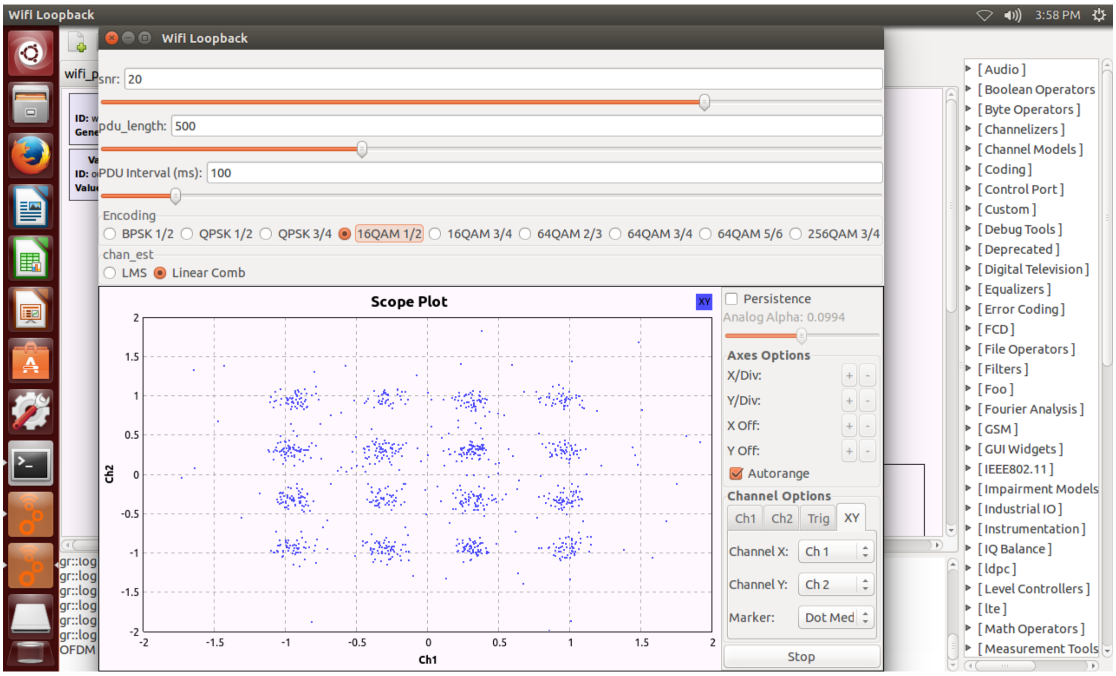
□ Persistence
Analog Alpha: 0.0994

☑ Autorange

Channel X: Ch 1
Channel Y: Ch 2
Marker: Dot Med

Figure 37. QPSK 3/4

The next modulation is a 16 QAM modulation. The code rate is 1/2. The points are concentrated in the sixteen coordinates of the modulation. This coordinates are declared in the C++ codes, chunks_to_symbols.cc and utils.cc.



snr: 20

pdu_length: 500

PDU Interval (ms): 100

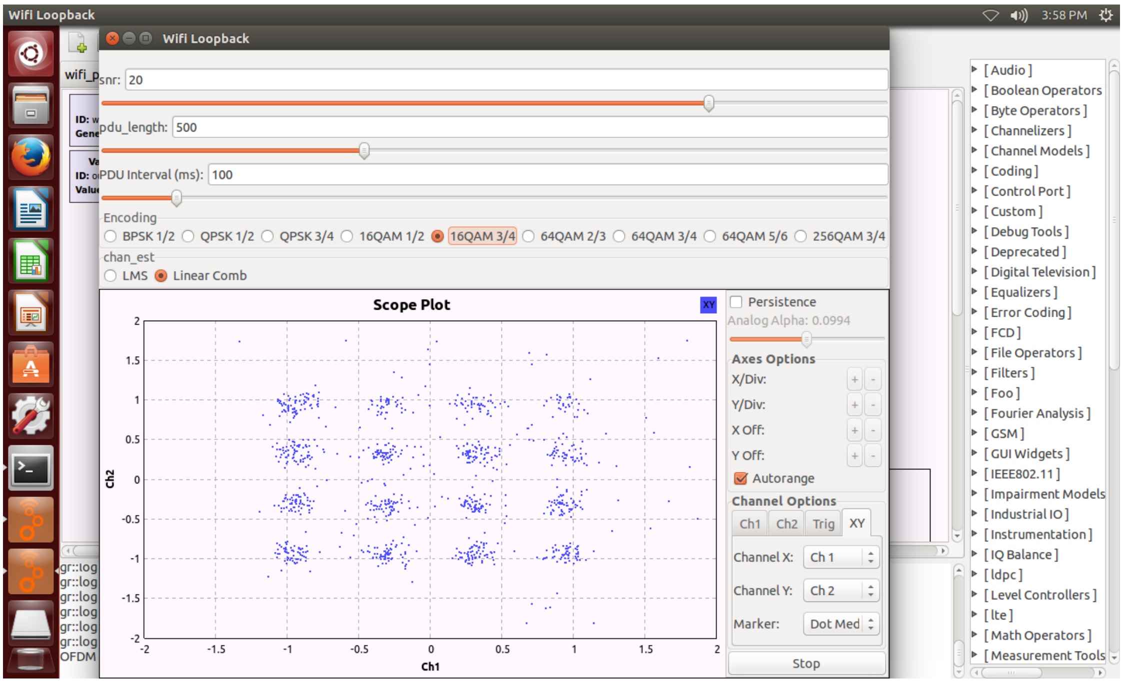
Encoding
○ BPSK 1/2 ○ QPSK 1/2 ○ QPSK 3/4 ● 16QAM 1/2 ○ 16QAM 3/4 ○ 64QAM 2/3 ○ 64QAM 3/4 ○ 64QAM 5/6 ○ 256QAM 3/4

chan_est
○ LMS ● Linear Comb

□ Persistence
Analog Alpha: 0.0994
☑ Autorange
Channel X: Ch 1
Channel Y: Ch 2
Marker: Dot Med

Figure 38. 16 QAM 1/2

The following modulation is a 16 QAM modulation. The code rate is 3/4. The points are concentrated in the same place than 16 QAM modulation and the code rate is 1/2. The points are more dispersed the puncturing matrix and the puncturing are different than for the modulation with code rate 1/2.



snr: 20

pdu_length: 500

PDU Interval (ms): 100

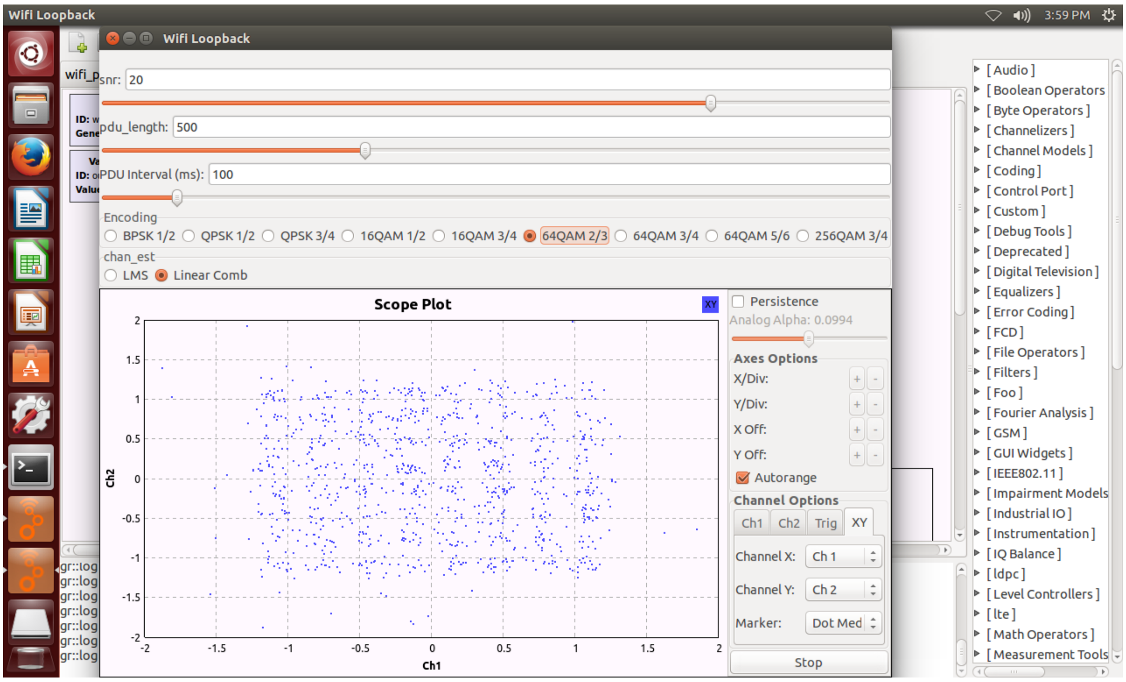
Encoding
○ BPSK 1/2 ○ QPSK 1/2 ○ QPSK 3/4 ○ 16QAM 1/2 ● 16QAM 3/4 ○ 64QAM 2/3 ○ 64QAM 3/4 ○ 64QAM 5/6 ○ 256QAM 3/4

chan_est
○ LMS ● Linear Comb

□ Persistence
Analog Alpha: 0.0994
☑ Autorange
Channel X: Ch 1
Channel Y: Ch 2
Marker: Dot Med

Figure 39. 16 QAM 3/4

The next modulation is a 64 QAM modulation. The code rate is 2/3. The points are concentrated in the sixty four coordinates of the modulation. This coordinates are declared in the C++ codes, chunks_to_symbols.cc and utils.cc.


Wifi Loopback

snr: 20

pdu_length: 500

PDU Interval (ms): 100

Encoding
○ BPSK 1/2 ○ QPSK 1/2 ○ QPSK 3/4 ○ 16QAM 1/2 ○ 16QAM 3/4 ● 64QAM 2/3 ○ 64QAM 3/4 ○ 64QAM 5/6 ○ 256QAM 3/4

chan_est
○ LMS ● Linear Comb

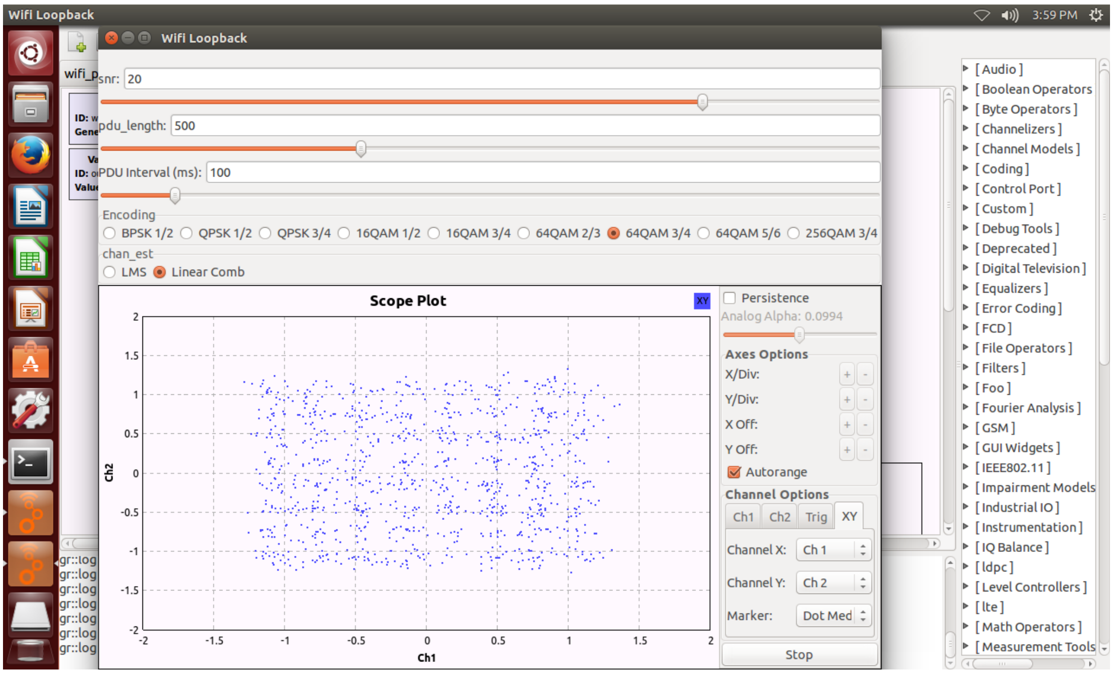
□ Persistence
Analog Alpha: 0.0994

☑ Autorange

Channel X: Ch 1
Channel Y: Ch 2
Marker: Dot Med

Figure 40. 64 QAM 2/3

The following modulation is a 64 QAM modulation. The code rate is 3/4. The points are concentrated in the same place than 64 QAM modulation and the code rate is 2/3. The points are more dispersed the puncturing matrix and the puncturing are different than for the modulation with code rate 2/3.



Wifi Loopback

snr: 20

pdu_length: 500

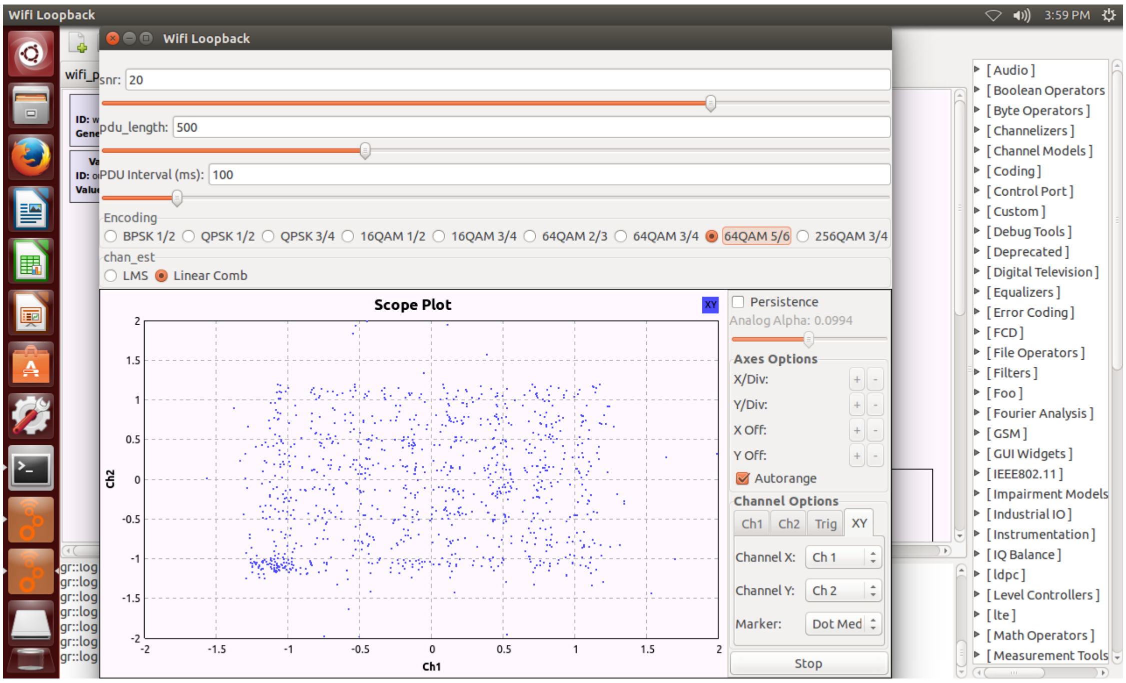
PDU Interval (ms): 100

Encoding
○ BPSK 1/2 ○ QPSK 1/2 ○ QPSK 3/4 ○ 16QAM 1/2 ○ 16QAM 3/4 ○ 64QAM 2/3 ● 64QAM 3/4 ○ 64QAM 5/6 ○ 256QAM 3/4

chan_est
○ LMS ● Linear Comb

Figure 41. 64 QAM 3/4

The following modulation is a 64 QAM modulation. The code rate is 5/6. This modulation is new for this standard. The points are concentrated in the same place than 64 QAM modulation and the code rate is 2/3. The points are more dispersed the puncturing matrix and the puncturing are different than for the modulation with code rate 2/3.



Wifi Loopback

snr: 20

pdu_length: 500

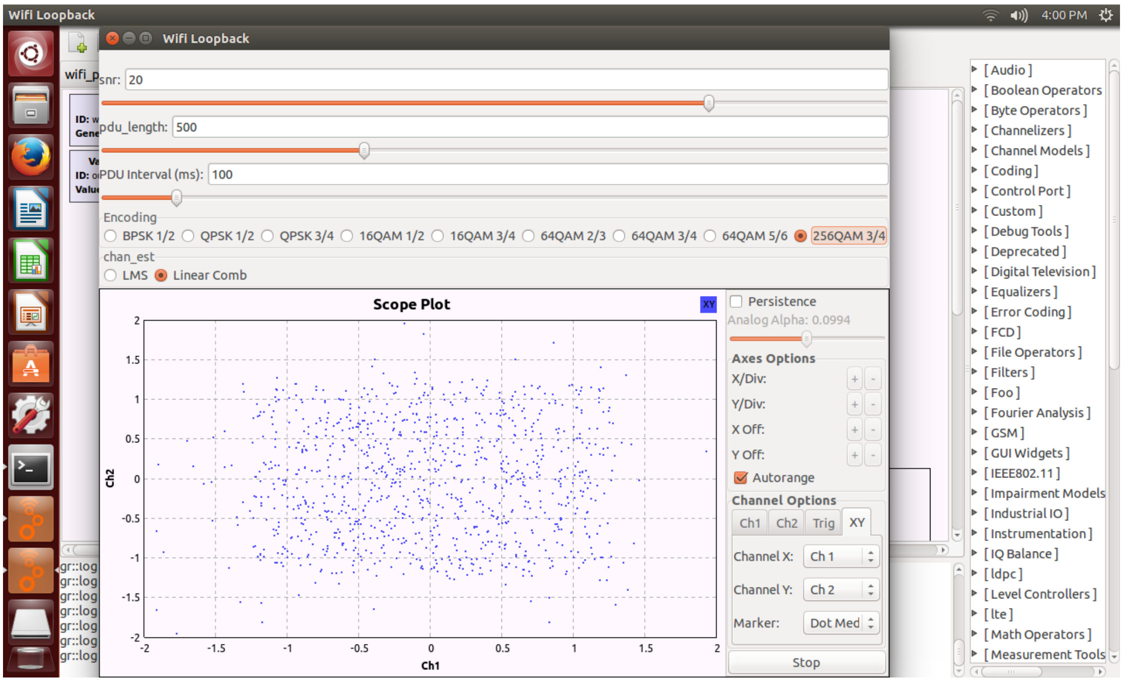
PDU Interval (ms): 100

Encoding
○ BPSK 1/2 ○ QPSK 1/2 ○ QPSK 3/4 ○ 16QAM 1/2 ○ 16QAM 3/4 ○ 64QAM 2/3 ○ 64QAM 3/4 ● 64QAM 5/6 ○ 256QAM 3/4

chan_est
○ LMS ● Linear Comb

Figure 42. 64 QAM 5/6

The next modulation is a 64 QAM modulation. The code rate is 2/3. The points are concentrated in the sixty four coordinates of the modulation. This modulation was not declared in the previous standard (802.11a/g/p). Thus, the coordinates have to be declared in the following C++ codes, chunks_to_symbols.cc and utils.cc.



Wifi Loopback

snr: 20

pdu_length: 500

PDU Interval (ms): 100

Encoding
○ BPSK 1/2 ○ QPSK 1/2 ○ QPSK 3/4 ○ 16QAM 1/2 ○ 16QAM 3/4 ○ 64QAM 2/3 ○ 64QAM 3/4 ○ 64QAM 5/6 ● 256QAM 3/4

chan_est
○ LMS ● Linear Comb

□ Persistence
Analog Alpha: 0.0994

☑ Autorange

Channel X: Ch 1
Channel Y: Ch 2
Marker: Dot Med

Figure 43. 256 QAM 3/4

10. CONCLUSION AND FUTURE WORK

10.1. CONCLUSION

In this degree final project is introduced the standard IEEE 802.11ah. The physical layer design is explained. This standard uses MIMO-OFDM and DSSS. It has a new channelization, it is different depending on countries.

MAC enhancements are developed. It is introduced the TIM stations and the Non-TIM stations. This new standard supports 8191 stations instead of 2007 then IEEE 802.11 supports. It has the power saving. A novelty of IEEE 802.11ah is there is a channel access mechanism for TIM stations and Non-TIM stations. It is described the association and authentication of this standard. There are mechanisms to enhance throughput.

Using a sub 1 GHz spectrum has advantages as large transmission rate (up to 7.8 Mbps), large coverage. The coverage of an access point is 1 Km.

IEEE 802.11 ah has not been standardized yet. It is except to finished this year.

To implement the standard IEEE 802.11ah, it is made changes in the model of 802.11 a/g/p to adapt it to the new standard. This implementation uses Single Input Single Output (SISO).

The biggest change in the C++ codes is to include the modulations 64 QAM 5/6 and 256 QAM 3/4. All the modulations with the same code rate share the puncturing matrix (Desktop/gr-ieee802-11/lib/ofdm_decode_mac.cc) and the puncturing (Desktop/gr-ieee802-11/lib/utils.cc). The C++ code changes in ofdm_mapper.h is this directory Desktop/gr-ieee802-11/include/ieee802-11/. The rest of the C++ codes are in this directory Desktop/gr-ieee802-11/lib/. They are chunks_to_symbols_impl.cc, chunks_to_symbols_impl.h, ofdm_decode_mac.cc, utils.cc and utils.h.

There are changes in all the flow graphs, wifi_hier_phy, transmitter, receiver and loopback. The changes are such as the frequency, the transmission modes, the modulations, the MTU and the occupied carriers in the block OFDM Carrier Allocator.

The results of the transmitter are the FFTs of the different transmission modes. To have the results the last block is the WX GUI FFT Sink block. The results of this flow graph are all the modulations.

10.2. FUTURE WORK

As described in the previous section, this implementation of the standard IEEE 802.11 ah by SDR uses SISO. To have more precise results, it is interesting to implement the standard using MIMO. Because the modulation used in the standard, is MIMO-OFDM.

In the transmitter should be added more inputs in the UHD: USRP Sink block. Thus, to the penultimate block connect all UHD: USRP Sink blocks inputs. In the receiver should be added more outputs in the UHD: USRP Source block, and then, it has to duplicate all the blocks, because there are more signals.

In the loopback, there is a block to model the channel with noise and there is other block to see the Frame Error Rate. When the loopback is running, in the GUI, a part where it should be the Frame Error Rate does not appear. This is a problem that must be solved because in the receiver this part of the Frame Error Rate appears.

11. REFERENCES

[1] B Polepalli, W Xie, D Thangaraja, M Goyal, H Hosseini, Y Bashir. *Impact of IEEE 802.11n Operation on IEEE 802.15.4 Operation*

[2] Marina Petrova, Janne Riihij¨arvi, Petri Mähönen and Saverio Labella. *Performance Study of IEEE 802.15.4 Using Measurements and Simulations*

[3] Kok Seng Ting, Gee Keng Ee, Chee Kyun Ng, Nor Kamariah Noordin and Borhanuddin Mohd. Ali. *The Performance Evaluation of IEEE 802.11 against IEEE 802.15.4 with Low Transmission Power*

[4] AN1200.22 LoRa™ Modulation Basics

[5] George Margelis, Robert Piechocki, Dritan Kaleshi,Paul Thomas. *Low Throughput Networks for the IoT: Lessons Learned From Industrial Implementations*

[6] Weiping Sun, Munhwan Choi and Sunghyun Choi. *IEEE 802.11ah: A Long Range 802.11 WLAN at Sub 1 GHz*

[7] Evgeny Khorov , Andrey Lyakhov, Alexander Krotov, Andrey Guschin. *A survey on IEEE 802.11ah: An enabling networking technology for smart cities*

[8] Alayn Loayssa (UPNA). Televisión Digital interactiva, *Redes de TV DVB*

[9] Stefan Aust, R. R. Venkatesha Prasad, Ignas G. M. M. Niemegeers. *IEEE 802.11ah: Advantages in Standards and Further Challenges for Sub 1 GHz Wi-Fi*

[10] T. Adame, A. Bel, B. Bellalta, J. Barcelo, M. Oliver. *IEEE 802.11ah: The Wi-Fi Approach for M2M Communications*

[11] JULIO CESAR ARAIZA LEON. *Evaluation of IEEE 802.11ah Technology for Wireless Sensor Network Applications*

[12] Pranesh Sthapit, Santosh Subedi, Goo-Rak Kwon, and Jae-Young Pyun. *Performance Analysis of Association Procedure in IEEE 802.11ah*

[13] Jeong-O Seo, Changwon Nam, Sung-Guk Yoon, and Saewoong Bahk. *Group-based Contention in IEEE 802.11ah Networks*

[14] Cesar V. Vargas, Wilson E. Lopez, Carlos F. da Rocha. *Sistemas de Comunicación Inalámbrica MIMO – OFDM*

[15] H. Wang, Supporting Authentication/Association for Large Number of Stations, 2012. *<http://mentor.ieee.org/802.11/dcn/12/11-12-0112-04-00ahsupporting-of-the-authentication-association-for-large-number-of-stations.pptx>*

[16] 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, Page 413

[17] *http://makers.sigfox.com/*

[18] *http://www.link-labs.com/sigfox-vs-lora/*

[19] *https://www.enocean.com/en/technology/radio-technology/*

[20] Espectro ensanchado por secuencia directa – Wikipedia. *https://es.wikipedia.org/wiki/Espectro_ensanchado_por_secuencia_directa*

[21] Direct-sequence spread spectrum – Wikipedia. *https://en.wikipedia.org/wiki/Direct-sequence_spread_spectrum*

[22] Software Defined Radio – Wikipedia. *https://en.wikipedia.org/wiki/Software-defined_radio*

[23] GNU Radio – Wikipedia. *https://en.wikipedia.org/wiki/GNU_Radio*

[24] EEE Standard for Information technology— Telecommunications and information exchange between systems— Local and metropolitan area networks— Specific requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)

[25] Chirp spread spectrum - Wikipedia, the free encyclopedia

[26] RP Photonics Encyclopedia - *https://www.rp-photonics.com/time_bandwidth_product.html*

[27] Syed E. Haque - *Efficient GTS Allocation Schemes for IEEE 802.15.4*

[28] Stefan Herbert AUST. *Advanced Wireless Local Area Networks in the Unlicensed Sub-1GHz ISM-bands*

[29] 802.11ac: A Survival Guide - *http://chimera.labs.oreilly.com/books/1234000001739/ch02.html#phy-section-vht-sig*

[30] Eduard Garcia-Villegas, Elena López-Aguilera (Dept. of Network Engineering) UPC. *IEEE 802.11ah sub 1 GHz WLAN for IoT*

[31] Convolutional code - *https://en.wikipedia.org/wiki/Convolutional_code*

[32] Monisha Ghosh, *Senior Member*, *IEEE* and Frank LaSita, *Member*, *IEEE* - *Puncturing of CRC codes for IEEE 802.11ah*

Appendix A – Initialize the project

To have GNURadio in windows, it is necessary a virtual machine with Linux version Ubuntu (64-bits).

In drive, there are two folders inside the folder named “ieee80211ah”, gr-foo and gr-ieee802-11. Inside of gr-ieee802-11 there is a folder named ‘examples’ where the flow graphs are, i.e. wifi_phy_hier, wifi_tx, wifi_rx and wifi_loopback.

To initialize this, first, you have to copy these two folders on the Desktop (for example) of your Linux machine and then, there are a few steps to follow:

Folder gr-foo: You have to go inside the folder named ‘build’, you have to use the following commands.

```
cd gr-foo

cd build

make

sudo make install
```

Folder gr-ieee802-11: You have to go inside the folder named ‘build’, you have to use the following commands.

```
cd gr- ieee802-11

cd build

make

sudo make install
```

Next, you have to open GNURadio and you have to open the flow graph wifi_phy_hier. Then, you have to click here to create the phyton code and you have to click here to reload. Because this flow graph is a Hier Block, thus, it is a block in other flow graph. In this case, in the transmitter.