CIE 478 Course Project

Implementation of IEEE 802.11a WLAN Physical Layer (OFDM)

The IEEE 802.11a Wireless LAN standard which operates in the 5 GHz unlicensed band occupying a 20 MHz bandwidth is based on OFDM. The main goal of the project is to implement the OFDM system specified in the 802.11a standard. This document will serve as a guide to the implementation, and will also study the OFDM design and discuss some of the design choices.

In 802.11a, $N = 64$ subcarriers are generated; however, only 48 subcarriers are used for data transmission, 12 zeroed to reduce adjacent channel interference, and 4 used as pilot symbols for channel estimation purposes. The cyclic prefix μ consists of 16 samples, so the total number of samples -associated with each transmitted OFDM symbol- is 80. The modulation types that can be used on the subcarriers are BPSK, QPSK, 16QAM or 64QAM. Forward error correction is applied using convolutional coding with rates 1/3, 2/3, or 3/4. The same modulation and code must be applied for all the subcarriers at any given time. The transmitter gets periodic feedback from the receiver about the packet error rate, and accordingly, it picks an appropriate modulation type and code rate.

Since there are 64 subcarriers distributed evenly over the 20 MHz bandwidth, the bandwidth occupied by each subcarrier is

$$B_N = \frac{20 \times 10^6}{64} = 312.5\ KHz$$

Since $\mu = 16$, and sampling rate $1/T_s = 20\ MHz$, the maximum delay spread for which ISI is eliminated is

$$T_m < \frac{16}{20\ MHz} = 0.8\ \mu sec$$

which corresponds to the delay spread in an indoor environment. Including the OFDM symbol and the cyclic prefix, there are 80 samples per OFDM symbol time, so the symbol time per subchannel is

$$T_N = 80Ts = \frac{80}{20 \times 10^6} = 4\ \mu s$$

The minimum data rate of the system which corresponds to BPSK modulation and code rate 1/2 -taking into account that only 48 subcarriers carry usable data- is

$$R_{min} = 48\ subcarriers \times \frac{1/2\ bit}{1\ coded\ bit} \times \frac{1\ coded\ bit}{1\ subcarrier\ symbol} \times \frac{1\ subcarrier\ symbol}{4\ \mu s}$$

$$= 6\ Mbps$$

The maximum data rate of the system which corresponds to 64QAM modulation and 3/4 code rate is

$$R_{min} = 48\ subcarriers \times \frac{3/4\ bit}{1\ coded\ bit} \times \frac{6\ coded\ bits}{1\ subcarrier\ symbol} \times \frac{1\ subcarrier\ symbol}{4\ \mu s}$$

$$= 54\ Mbps$$

Other combinations of modulation types and code rates provide a wide range of data rates between the two extremes.

System specifications

The block diagram of the OFDM transmitter and receiver is shown figure 1. In this subsection, we will discuss the important details of some of the blocks.

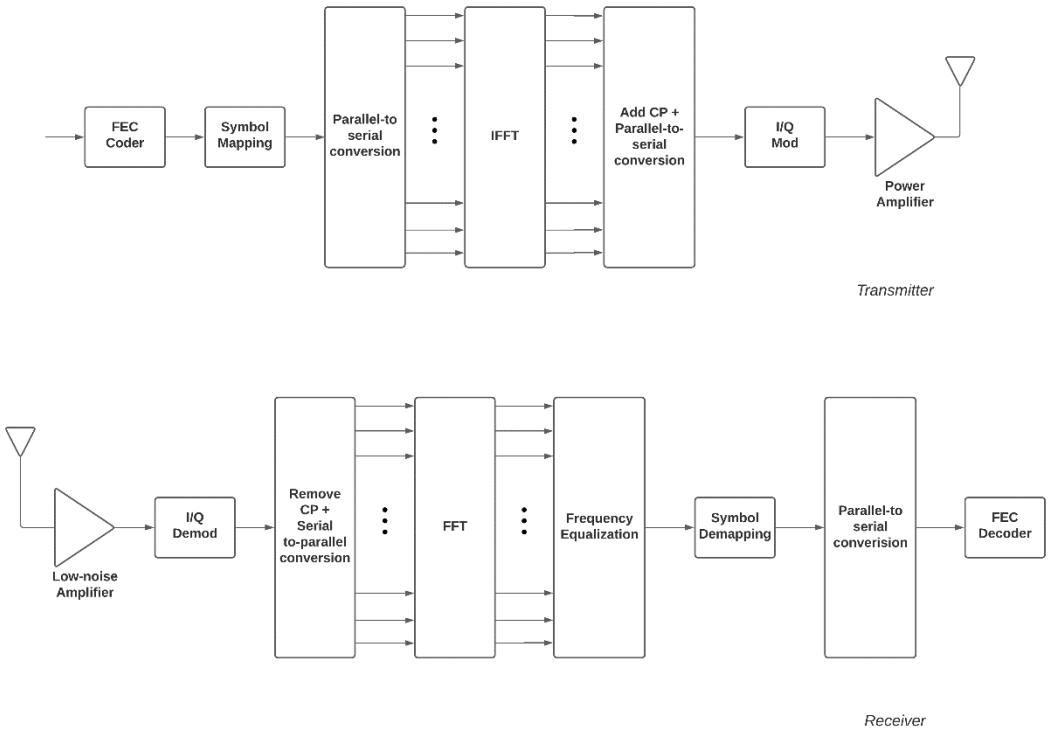

Figure 1 OFDM system block diagram

Forward Error Correction (FEC) Coder

As we already mentioned, 802.11a uses convolutional encoding for forward error correction. The convolutional encoder uses the generator polynomials, $g_0 = 1011011_2$, $g_1 = 1111001_2$, of rate = 1/2. The output bit denoted as A is outputted from the encoder before bit B. The convolutional encoder is illustrated in figure 2. Higher rates are achieved by applying puncturing to the coded data. Puncturing is a procedure for omitting some of

the encoded bits in the transmitter to decrease the number of transmitted bits, which increases the coding rate. At the receiver, before FEC decoding, dummy zero bits are inserted in place of the omitted bits. Decoding is done by the Viterbi algorithm.

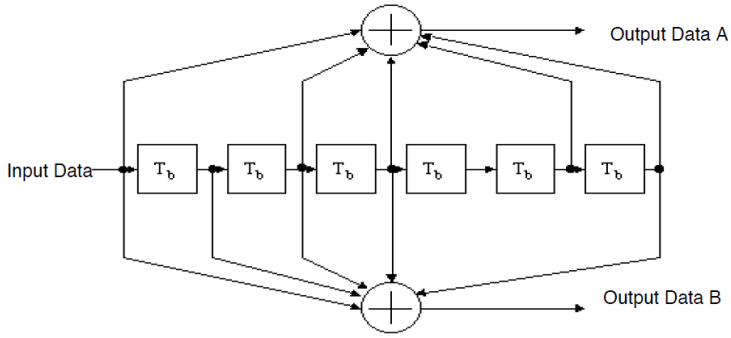

Figure 2 Convolutional encoder (constraint length = 7)

The following example illustrates the puncturing procedure to realize code rates of 2/3 or 3/4 from an output with rate 1/2.

Punctured coding (r = 3/4)

Source data

X_0	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8

Encoded data

A_0	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
B_0	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8

Punctured data

A_0	A_1		A_3	A_4		A_6	A_7	
B_0		B_2	B_3		B_5	B_6		B_8

Sent/received data

A_0	B_0	A_1	B_2	A_3	B_3	A_4	B_5	A_6	B_6	A_7	B_8

Bit inserted data

A_0	A_1	0	A_3	A_4	0	A_6	A_7	0
B_0	0	B_2	B_3	0	B_5	B_6	0	B_8

Decoded data

y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8

Punctured coding (r = 2/3)

Source data

X_0	X_1	X_2	X_3	X_4	X_5

Punctured data

A_0	A_1	A_2	A_3	A_4	A_5
B_0	B_1	B_2	B_3	B_4	B_5

Sent/received data

A_0	B_0	A_1	A_2	B_2	A_3	A_4	B_4	A_5

Bit inserted data

A_0	A_1	A_2	A_3	A_4	A_5
B_0	0	B_2	0	B_4	0

Decoded data

y_0	y_1	y_2	y_3	y_4	y_5

Symbol mapping

The OFDM subcarriers are modulated using BPSK, QPSK, 16QAM, or 64 QAM depending on the desired rate. The bitstream is divided into groups of size 1, 2, 4 or 6; respectively, and mapped to complex numbers representing the constellation points. The mapping is done using gray encoding which is illustrated for 16QAM in figure 3. Please refer to pages 19, 20 and 21 (section 17.3.5.7) in the attached 802.11a standard for the figures of constellation mappings and encoding tables.

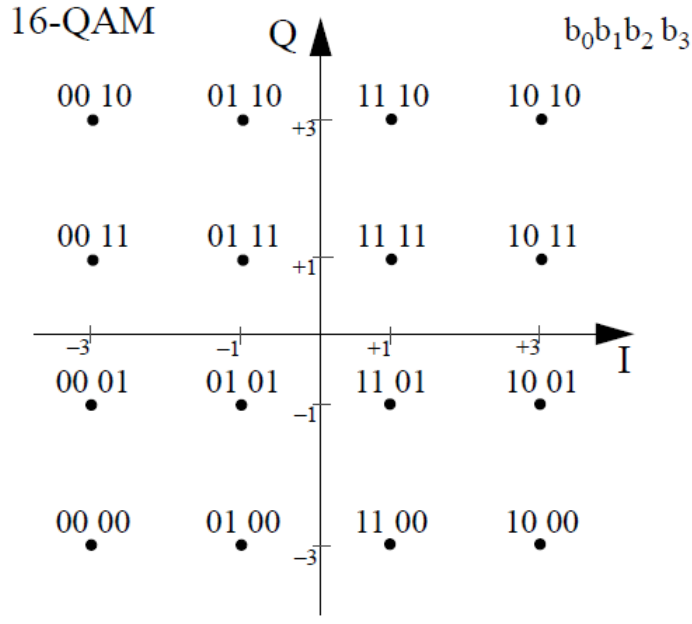

Figure 3 Gray-encoded 16QAM constellation

To achieve the same average power for all modulation types, the output complex symbols are multiplied by the normalization factors in table 1. The output value d of the mapper is formed by the following equation

$$d = K_{mod}(I + jQ)$$

Table 1 Modulation normalization factor

Modulation	$\boldsymbol{K_{mod}}$
BPSK	1
QPSK	$1/\sqrt{2}$
16QAM	$1/\sqrt{10}$
64QAM	$1/\sqrt{42}$

OFDM Modulation (IDFT)

The stream of complex numbers is divided into groups of 48 symbols. The mapping is done using the function

$$M(k) = \begin{cases} k-26, & 0 \le k \le 4 \\ k-25, & 5 \le k \le 17 \\ k-24, & 18 \le k \le 23 \\ k-23, & 24 \le k \le 29 \\ k-22, & 30 \le k \le 42 \\ k-21, & 43 \le k \le 47 \end{cases}$$

which defines a mapping from the logical subcarrier number k ranging from 0 to 47 to a frequency offset index -26 to 26, while skipping the null subcarriers. The pilot symbols are placed at the frequency offsets -21, -7, 7, and 21. The locations of the frequency offset indices are illustrated in figure 4.

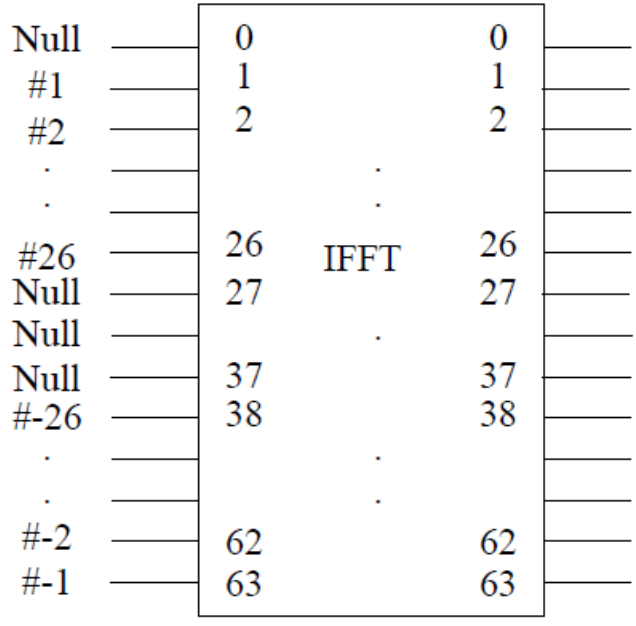

Figure 4 Inputs to the IDFT

The pilots' contribution to the n^{th} OFDM symbol is produced by the following sequence

$$P_{-26,26} = \{0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,$$
$$0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,-1,0,0,0,0,0\}$$

802.11a simplified frame structure

The 802.11a frame has three main fields: Preamble, Signal, and Data. The preamble field is for time and frequency synchronization, and channel estimation, the Signal field is used to exchange control information between the transmitter and the receiver, and the Data field is for the useful information. The Preamble and the Signal fields are discussed in more detail below. The 802.11a frame structure is illustrated in figure 5.

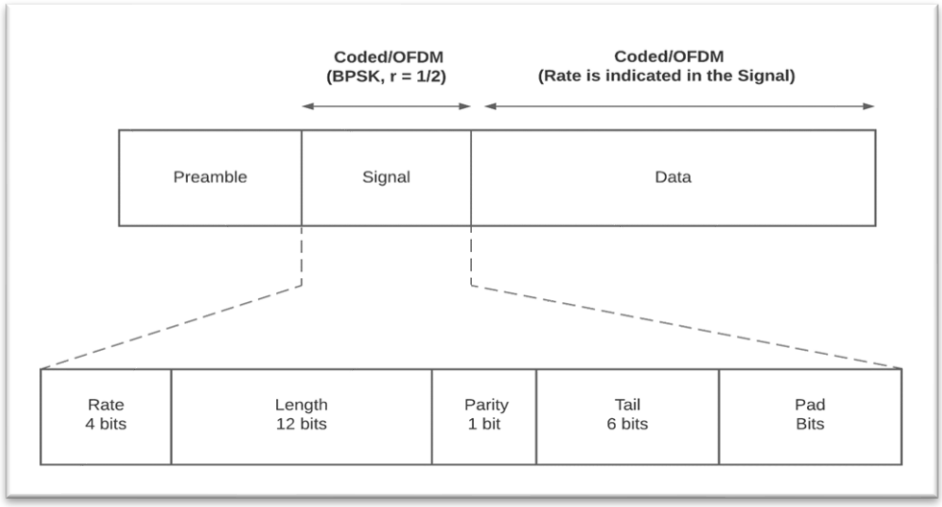

Figure 5 802.11 a simplified frame structure

Preamble

The preamble field is illustrated in figure 6. It consists of two short training symbols used for timing and frequency synchronization, and two long training symbols used for channel estimation.

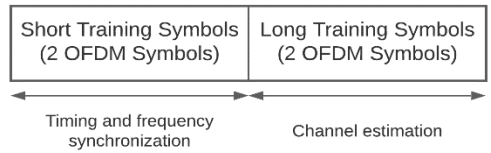

Figure 6 Structure of the preamble field

A short training symbol utilizes only 12 out of the 52 subcarriers with zero value at dc, and is generated using the following sequence

$$S_{-26,26} = \sqrt{\frac{13}{6}} \times \{0, 0, 1+\mathrm{j}, 0, 0, 0, -1-\mathrm{j}, 0, 0, 0, 1+\mathrm{j}, 0, 0, 0, -1-\mathrm{j}, 0, 0, 0, -1-\mathrm{j}, 0, 0, 0, 1+\mathrm{j}, 0, 0, 0, 0,$$

$$0, 0, 0, -1-\mathrm{j}, 0, 0, 0, -1-\mathrm{j}, 0, 0, 0, 1+\mathrm{j}, 0, 0, 0, 1+\mathrm{j}, 0, 0, 0, 1+\mathrm{j}, 0, 0, 0, 1+\mathrm{j}, 0, 0\}$$

The factor $\sqrt{13/6}$ is used to normalize the average power of the resulting OFDM symbol.

A long training symbol utilizes 52 subcarriers with zero value at dc, and is generated using the following sequence

$$L_{-26,26} = \{1, 1, -1, -1, 1, 1, -1, 1, -1, 1, 1, 1, 1, 1, 1, -1, -1, 1, 1, -1, 1, -1, 1, 1, 1, 1, 0,$$
$$1, -1, -1, 1, 1, -1, 1, -1, 1, -1, -1, -1, -1, -1, 1, 1, -1, -1, 1, -1, 1, -1, 1, 1, 1, 1\}$$

Signal

The information in the signal field is critical as it tells the receiver how to decode the data in the frame. Thus, it is transmitted using the most robust modulation and coding (BPSK, r = 1/2). It has five fields: Rate, Length, Parity, Tail, and Pad Bits.

The rate field is used to inform the receiver about the modulation type, and the code rate, so that it can decode the data correctly. The 4 bits (R1-R4) of the Rate field are set according to the values in table 2.

Table 2 Rate field values

Rate (Mbps)	R1-R4	Modulation	Code rate
6	1101	BPSK	1/2
9	1111	BPSK	3/4
12	0101	QPSK	1/2
18	0111	QPSK	3/4
24	1001	16QAM	1/2
36	1011	16QAM	3/4
48	0001	64QAM	2/3
54	0011	64QAM	3/4

The Length field indicates the size of the data in the frame in bytes, Parity is an even parity bit for bits 0 to 16 (Rate and Length fields). The 6 bits of the Tail field are always set to 0 to reset the convolutional decoder to the *zero state* after decoding the control information. This procedure helps in improving the error probability. The number of bits in the frame must be an integer multiple of the number of coded bits per subcarrier. Thus, the pad bits are used to extend the message as needed.

Project requirements

Please read the requirements carefully

A) Floating-point implementation

Your code should be divided into three main modules:

1. A function that creates the 802.11a frame as discussed in the document. The inputs to the function shall be the data, modulation type and code rate.
2. A script/function for the 802.11a OFDM transmitter.
3. A script/function for the 802.11a OFDM receiver.

Your implementation of the 802.11 OFDM transmitter and receiver should support all the modulation types, and code rates in the standard. The OFDM receiver should support both zero-forcing and Weiner equalization. For convolutional coding, you can use the built-in MATLAB module.

B) Fixed-point implementation

This part has the same requirements as part A; however, the design should be done using fixed-point representation. An efficient fixed-point design utilizes as minimum hardware resources as possible for each processing step while maintaining little to no performance degradation.

C) Interleaving and scrambling (Bonus)

In 802.11a, all the coded bits are interleaved to enhance the ability of the convolutional code to correct burst errors which might happen due to deep fades on some of the subcarriers. Interleaving improves the packet error rate performance of the system. Scrambling is done to make the transmitted data unintelligible; it could be as simple as XORing the data with a pseudorandom sequence that is known to both the transmitter and the receiver. The received data is then descrambled at the receiver.

The requirement is to implement the scrambling-descrambling and interleaving-deinterleaving procedures specified in the attached 802.11a standard; sections 17.3.5.4, and 17.3.5.6. You can use the built-in MATLAB modules.

Testing

You shall proceed with the following steps to test your implementation:

1. Read a text file in binary ASCII format.
2. Divide the file into segments of size 1000 bytes.
3. Encapsulate each segment in an 802.11a frame.

4. Forward the frames to the OFDM transmitter, then pass them through a time-invariant frequency-selective wireless channel without white noise.
5. Receive the frames using your implementation of the OFDM receiver.

For equalization, you can use either the ZF equalizer or the Weiner equalizer. The transmitted files should be recovered with no errors.

Simulation of the wireless channel

You can simulate the wireless channel using a time-invariant complex FIR filter. The main requirement is that the delay spread should be less than the maximum delay spread for which ISI is eliminated which we found to be 0.4 μs. Since the sampling time $T_s = 1/20\ Mhz = 0.05\ \mu s$, the maximum size of the filter should be $\frac{0.4\ \mu s}{0.05\ \mu s} = 8\ taps$. In other words, the last received symbol should be delayed with no more than 8 samples to ensure ISI elimination. You can use the following filter whose delay spread is 0.15 μs:

$$h = [0.8208 + 0.2052\text{j}, 0.4104 + 0.1026\text{j}, 0.2052 + 0.2052\text{j}, 0.1026 + 0.1026\text{j}]^T$$

Deliverables and deadlines

Phase 1

In this phase, you should submit your code which supports at least two modulation schemes, and one code rate. Fixed-point implementation is not required in this phase.

Phase 2

In this phase, you should submit the project code which includes the following:

1. Function that creates the 802.11a frame.
2. Floating-point implementation of the OFDM transmitter and receiver.
3. Fixed-point implementation of the OFDM transmitter and receiver.
4. Testing script which includes the following:
    a. Verification that the text files are received correctly for all supported rates. You will find two attached text files: *test_file_1.txt*, and *test_file_2.txt* for testing.
    b. Comparison between the BER performance of the ZF equalizer, and the Weiner equalizer for 64QAM modulation, and 3/4 code rate using the floating-point implementation (both curves on the same figure).

c. Constellation diagram of the received symbols after equalization using the ZF equalizer, and the Weiner equalizer for 64QAM modulation, and 3/4 code rate using the floating-point implementation at any SNR of your choice (one figure for each equalizer).
d. Comparison between the BER performance of the floating-point and fixed-point implementations for 16QAM modulation, and 3/4 code rate (both curves on the same figure).
e. Comparison between the BER performance of all supported rates using the floating-point implementation (all curves on the same figure).
f. Repeat part e using the fixed-point implementation.

You should also **submit the project report** which includes figures of your testing results along with proper comments.

References

[1] A. Goldsmith, "Multicarrier Modulation," in *Wireless Communications*, Cambridge University Press.

[2] IEEE 802.11a-1999: High-speed physical layer in the 5 GHz band, 1999.