Zigbee Standard (IEEE 802.15.11)

Course Instructor : Dr. Ahmed Eltarss
TAs : Eng. Heba selim & Eng. Mariam Kandil

Mohamed Saleh 201-902-216
Nada Ismail, 202-001-387
Sama Yousef 202-000-819

8th January, 2024

A Introduction

In ever-evolving landscape of wireless communication, the Zigbee standard has became a valuable protocol for enabling reliable, low-power, and cost-effective IoT networks (WPANs) across various applications. The physical layer of the Zigbee standarad plays is the main focus in the project.

This report aims to delve into the intricacies of the Zigbee standard's physical layer by utilizing two powerful tools: Matlab for simulation and LabView for implementation. Matlab, with in mathmatical modelling and simulation, provides a way to test and analyze the algorithms that of Zigbee's physical layer.

On the flip side, LabView offers a unique platform that blends the software's flexibility with the hardware's tangibility, allowing us to implemment and validate the physical layer's protocols in real-world scenarois.

B Uses of Zigbee

Zigbee is a standards-based wireless technology developed to enable low-cost, low-power wireless machine-to-machine (M2M) and internet of things (IoT) networks. Its main applications span across a wide array of fields. Here are some of the prominent uses:

B.1 Home Automation

Zigbee enables smart devices to communicate with each other in residential settings, including lighting controls, security systems, thermostats, and other home appliances.

B.2 Wireless Sensor Networks

It is widely used for environmental monitoring through sensor networks, where deploying wiring would be challenging or expensive.

B.3 Industrial Control Systems

In industrial settings, Zigbee provides a solution for controlling and monitoring machinery, managing energy use, and automating routine tasks.

B.4 Embedded Sensing

Miniaturized devices for embedded sensing in various contexts such as infrastructure health monitoring or agricultural conditions often rely on Zigbee.

B.5 Medical Data Collection

For non-invasive medical devices that collect patient data, Zigbee offers a way to transmit this data wirelessly and securely to healthcare providers.

B.6 Smoke and Intruder Warning

Security systems employ Zigbee to connect alarms and sensors for real-time alerting in case of smoke detection or unauthorized entry.

B.7 Building Automation

Large buildings can utilize Zigbee for coordinated control systems including HVAC, lighting, and access controls. Remote Wireless Microphone Configuration: Audio systems in large venues or for professional setups may use Zigbee for configuring and controlling wireless microphones.

C History of Zigbee

The Zigbee standard was conceived in the late 1990's, when a group of companies formed the Zigbee Alliance. The technology is built upon the IEEE 802.15.4 standard, which was relaesed in 2003 and defines the layers associated with the physical radio signal and a low-level control of the radio itself. Zigbee adds network, security, and application software layers on top of this, providing a framework for IoT networks.

The Zigbee specification was finalized in Decemeber 2004 and since then, Zigbee has evolved through different versions, improving the robustness, scalability, and ease of use of the network. One of the most significant updates to the Zigbee standard was Zigbee 3.0, which aimed to unify the various Zigbee wireless standards to create a more harmonized market and consumer experience.

Zigbee's low power consumption and mesh networking capabilities, allowing for devices to interconnect and extend the range of the network through each other, have made it particularly attractive for a various of IoT applications. Zigbee continues to grow in popularity as the IoT market expands, with new devices regularly being brought into the Zigbee ecosystem.

D Physical Layers Specifications

We have two modes of operation in Zigbee one QPSK and the other on BPSK on three bands.
Physical Range: 10-100 meters

D.1 868/902 MHz band

868MHz is used in north America while 902 is used in Europe
Pulse Shaping filter: Raised Cosine
Modulation type: BPSK

D.2 2.4 GHz band

2.4 GHz is used worldwide.
Pulse Shaping filter: Raised Cosine
Modulation type: BPSK

Since the 2.4GHz frequency range is an unlicensed frequency range, it could be used worldwide but since QPSK is used to optimize the the Bandwidth at 2.4 there is higher probabiltiy of error.

E Matlab Simulation

Parameters: Setting carrier frequencies, energy of a single bit and signal to noise ratio.
Load data: loading image using its path.
Transmitters: Implementing BPSK and QPSK transmitter.
Channel: Implementing the channels AWGN, Rayleigh, and rician.
Receivers: Implementing BPSK and QPSK receiver.
Saving data: Saving all output images.

```
%% parameters
fc1 = 915; % carrier frequency for the first band of operation
fc2 = 2450; % carrier frequency for the second band of operation
img_path = "cameraman.tif";
Eb = 1; % energy of a single bit
Tb1 = 1/fc1; % bit duration for the first band of operation (assumed to equal the carrier
    period)
Tb2 = 1/fc2; % bit duration for the second band of operation (assumed to equal carrier
    period)
snr_db = 1; % signal to noise ratio expressed in db;

%% Load data

% load image from path
```

```
[img, bits_img, img_size] = load_image(img_path);


%% Transmitters

% BPSK transmitter
tx_BPSK = pskmod(bits_img, 2, InputType="bit");
scatterplot(tx_BPSK);
title('Transmitted BPSK symbols')

% QPSK transmitter
tx_QPSK = pskmod(bits_img, 4, InputType="bit");
scatterplot(tx_QPSK)
title('transmitted QPSK symbols')



%% channel

% AWGN channel
rx_BPSK = awgn(tx_BPSK, snr_db);
rx_QPSK = awgn(tx_QPSK, snr_db);


%rayleigh channel
rayleighchan = comm.RayleighChannel(...
                "SampleRate", fc2, ...
                'PathDelays', [0 1e-20], ...
                'AveragePathGains', [4 6], ...
                'NormalizePathGains', true, ...
                'MaximumDopplerShift', 0.01, ...
                'RandomStream', 'mt19937ar with seed', ...
                'Seed', 22, ...
                'PathGainsOutputPort', false);

rayleighchan.ChannelFiltering=true;
rx_BPSK_rey = rayleighchan(tx_BPSK);
rx_QPSK_rey = rayleighchan(tx_QPSK);

%rician channel
ricianchan = comm.RicianChannel(...
                "SampleRate", fc2, ...
                'PathDelays', [0 1e-3], ...
                'AveragePathGains', [2 1], ...
                'MaximumDopplerShift', 0.01, ...
                'PathGainsOutputPort', false);

ricianchan.ChannelFiltering=true;
rx_BPSK_rice = ricianchan(tx_BPSK);
rx_QPSK_rice = ricianchan(tx_QPSK);

%% Receivers

% BPSK receiver
scatterplot(rx_BPSK)
title('Received noisy BPSK symbols')
bits_img_BPSK = pskdemod(rx_BPSK, 2, OutputType='bit');

scatterplot(rx_BPSK_rey)
title('Received noisy reyleigh BPSK symbols')
bits_img_BPSK_rey=pskdemod(rx_BPSK_rey, 2, OutputType='bit');

scatterplot(rx_BPSK_rice)
title('Received noisy rician BPSK symbols')
bits_img_BPSK_rice=pskdemod(rx_BPSK_rice, 2, OutputType='bit');

% QPSK receiver
scatterplot(rx_QPSK)
title('Received noisy QPSK symbols')
bits_img_QPSK = pskdemod(rx_QPSK, 4, OutputType='bit');

scatterplot(rx_QPSK_rey)
title('Received noisy reyleigh QPSK symbols')
```

```
bits_img_QPSK_rey=pskdemod(rx_QPSK_rey, 4, OutputType='bit');

scatterplot(rx_QPSK_rice)
title('Received noisy rician QPSK symbols')
bits_img_QPSK_rice=pskdemod(rx_QPSK_rice, 4, OutputType='bit');

%% Save data

img_BPSK = save_image(bits_img_BPSK, img_size, 'output_bpsk.jpg');
img_BPSK_rey = save_image(bits_img_BPSK_rey, img_size, 'output_bpsk_rey.jpg');
img_BPSK_rice = save_image(bits_img_BPSK_rice, img_size, 'output_bpsk_rice.jpg');

img_QPSK = save_image(bits_img_QPSK, img_size, 'output_qpsk.jpg');
img_QPSK_rey = save_image(bits_img_QPSK_rey, img_size,'output_qpsk_rey.jpg');
img_QPSK_rice = save_image(bits_img_QPSK_rice, img_size,'output_qpsk_rice.jpg');

%% Graph BER vs. SNR

SNR = -10:0.01:50;
BER_BPSK = [];
BER_QPSK = [];

for i = 1:length(SNR)

    % pass through channel
    rx_BPSK = awgn(tx_BPSK, SNR(i));
    rx_QPSK = awgn(tx_QPSK, SNR(i));

    % demodulate
    data_BPSK = pskdemod(rx_BPSK, 2);
    data_QPSK = pskdemod(rx_QPSK, 4, OutputType='bit');

    % calculate BER
    [x, ber_BPSK] = symerr(bits_img, data_BPSK);
    [x, ber_QPSK] = symerr(bits_img, data_QPSK);

    % append to BER arrays
    BER_BPSK = [BER_BPSK ber_BPSK];
    BER_QPSK = [BER_QPSK ber_QPSK];
end

% Plot BER vs. SNR of BPSK
figure;

subplot(1, 2, 1);
semilogy(SNR, BER_BPSK)
title('SNR vs. BER of BPSK (Simulated)');
xlabel('SNR (dB)')
ylabel('BER')

subplot(1, 2, 2);
semilogy(SNR, 0.5 * erfc(sqrt(10.^(SNR / 10))));
title('SNR vs. BER of BPSK (Theoretical)');
xlabel('SNR (dB)')
ylabel('BER')


% Plot BER vs. SNR of QPSK
figure;

subplot(1, 2, 1);
semilogy(SNR, BER_QPSK)
title('SNR vs. BER of QPSK (Simualted)')
xlabel('SNR (dB)')
ylabel('BER')


subplot(1, 2, 2);
semilogy(SNR, 0.5 * erfc(sqrt(10.^(SNR / 10))));
title('SNR vs. BER of QPSK (Theoretical)')
xlabel('SNR (dB)')
ylabel('BER')

```

```
%% user-defined functions

function modulated_bpsk = modulate_bpsk(bits, Eb)
    % bits: bit sream to be modulated
    % Eb: energy of a single bit

    modulated_bpsk = pskmod(bits, 2)*Eb;
end

function img = save_image(bits_img, img_size, img_path)

    % from bit stream to bit characters
    bits_img = char(bits_img + '0');

    % convert the image from a stream of bits into matrix form
    img = reshape(bin2dec(reshape(bits_img, [], 8)), img_size(1), img_size(2));

    % convert pixel values to uint8
    img = uint8(img);

    % save image
    imwrite(img, img_path);

end

function [img, bits_img, img_size] = load_image(img_path)

    % read image from path
    img = imread(img_path);

    % turn the image into greyscale, if it is not in grey scale
    if length(size(img)) > 2
        img = rgb2gray(img);
    end

    % store image size
    img_size = size(img);

    % represent the image as a stream of bit characters
    bits_img = reshape(dec2bin(img), [], 1);

    % Convert the bit characters into bit stream
    bits_img = bits_img - '0';

end
```

F Noise Channels

When examining the influence of noise in communications channels, it is crucial to consider the effects of different noise models on the performance of the Zigbee standard, which operates in the industrial, scientific, and medical (ISM) bands and is designed for low-power, low-data-rate, and close-proximity wireless personal area networks (WPANs). Here's a structured overview of how three common noise types—AWGN, Rayleigh, and Rician—impact Zigbee communication.

F.1 Additive White Gaussian Noise (AWGN)

The AWGN model represents a type of noise characterized by a wide frequency band with a flat spectral density (white noise), and with amplitude values that follow a Gaussian distribution. In a Zigbee communication system, AWGN would be analogous to a constant background noise, affecting all frequencies equally. As a result, it can degrade the performance of the signal-to-noise ratio (SNR), leading to higher bit error rates (BER). However, Zigbee's modulation schemes, such as Offset Quadrature Phase-Shift Keying (O-QPSK), are relatively robust against Gaussian noise, and forward error correction (FEC) can be employed to help mitigate this noise.

Figure 1: Image with BPSK modulation and AWGN noise

Figure 2: Image with QPSK modulation and AWGN noise

F.2 Rayleigh Fading

Rayleigh fading describes the signal degradation that happens when a transmitted signal encounters multiple indirect paths before arriving at the receiver, with none of these paths being dominant. This can be the case in urban environments full of obstructions where non-line-of-sight (NLOS) conditions predominate. The impact on Zigbee devices can be significant because the varying signal strength can result in intermittent drops in connectivity or increased error rates. Zigbee networks can adapt to some extent to these conditions through mesh network protocols, which allow for the re-routing of messages through alternate pathways when certain links become unreliable due to fading.

Figure 3: Image with BPSK modulation and Rayleigh Fading noise

Figure 4: Image with QPSK modulation and Rayleigh Fading noise

F.3 Rician Fading

Rician fading is a model that applies to scenarios where a line-of-sight (LOS) path exists along with multiple reflected and scattered paths. This might be more typical in mixed-environment conditions where Zigbee nodes have fewer obstructions between them. Rician fading causes fluctuations in the signal amplitude and phase, which can lead to a varying BER in the received data. Zigbee's channel hopping feature and the ability to assess link quality can aid in mitigating the adverse effects of Rician fading by choosing better channels and routes for data transmission.

Figure 5: Image with BPSK modulation and Rician Fading noise

Figure 6: Image with QPSK modulation and Rician Fading noise

In summary, the impact of AWGN, Rayleigh, and Rician noises on a Zigbee standard is mainly on the reliability and quality of the communication link, affecting factors such as BER, SNR, throughput, and overall network performance.There are higher Error rate in the QPSK than BPSK. Zigbee's design includes several mechanisms to combat these issues, ensuring reliable operation despite the presence of channel noise and fading. Quantitative assessment typically involves simulation or empirical studies to measure the impact on BER and SNR, often leading to the deployment of additional network-level strategies to maintain communication integrity.

G BER vs SNR

the SNR value used in our simulation was 1 which means there is as much signal as there is noise to simulate the actual use cases for Zigbee where low power comsumption is a must. However, you will find the BER vs SNR graph for multiple BERs.

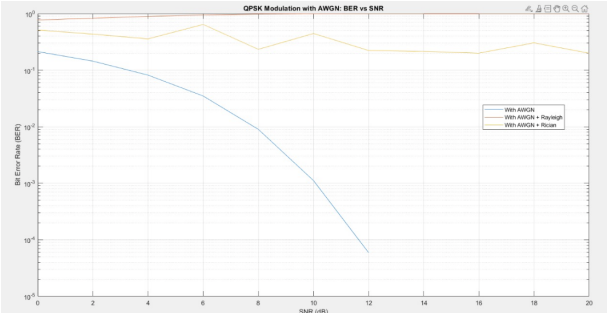

Figure 7: Bit Error Rate vs Signal to Noise Ratio

H LabView implementation

We used the only full duplex port on the NI-USRP to send and receive

H.1 USRP Transmitter

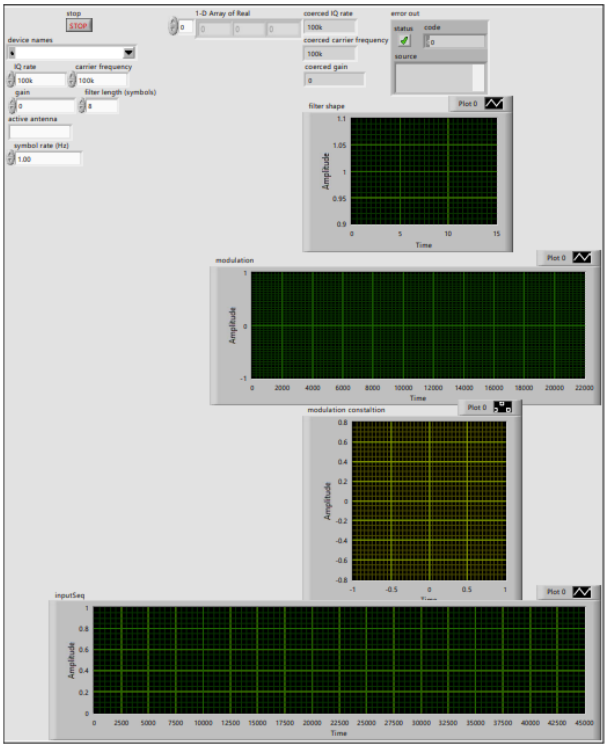

Figure 8: Transmitter Front Panel

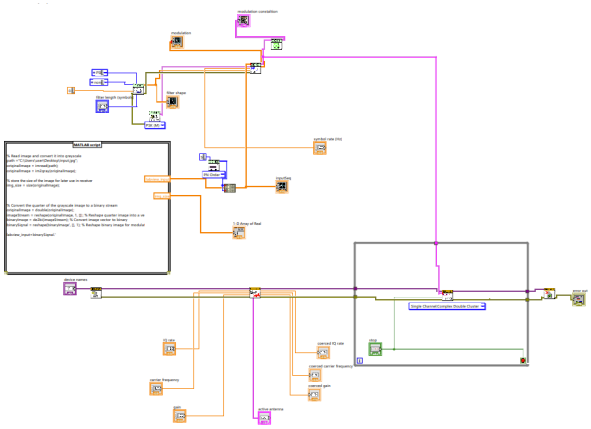

Figure 9: Transmitter block diagram

H.2 USRP Receiver

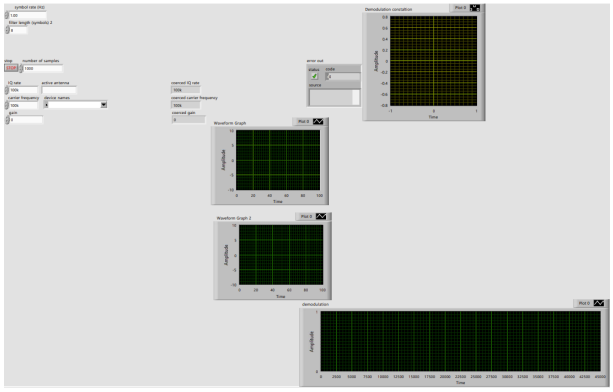

Figure 10: Receiver Front Panel

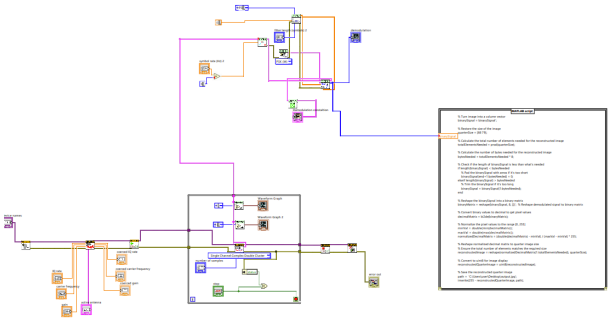

Figure 11: Receiver block diagram

H.3 USRP transceiver

for simulation purposes we have simulated the entire system on LabView.

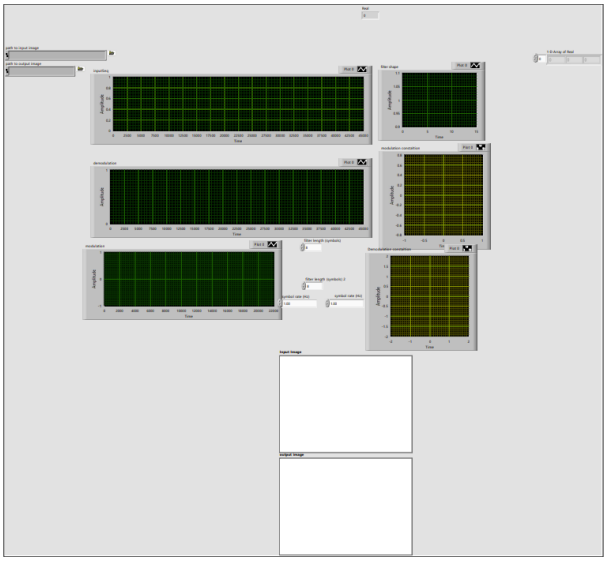

Figure 12: transceiver Front Panel

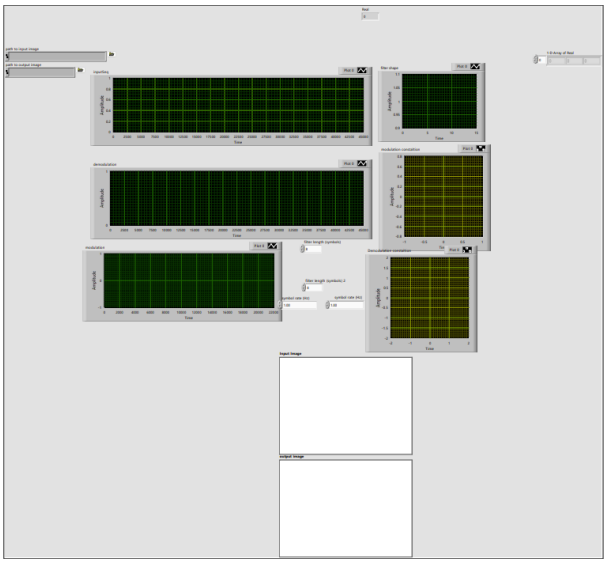

Figure 13: transceiver block diagram part 1

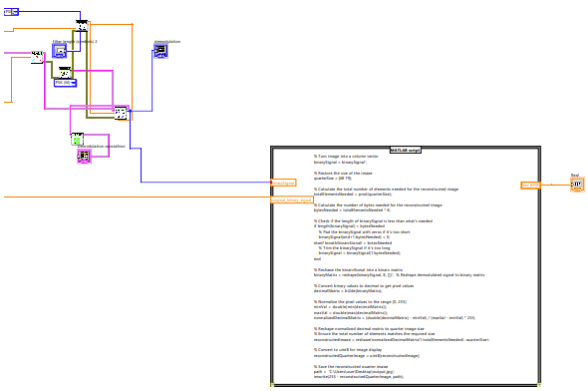

Figure 14: transceiver block diagram part 2