DIGCOMT Progress Report

Group 2, Section EK2

Members:

1. Ethan Joseph Dee
2. Jean Lenard Laca
3. Mariel Annika Ong
4. Matthew Peruel
5. Patricio Martin Visda

# Modulation scheme: BPSK

The MATLAB code is attached in this document in the Appendix. Alternatively it may be accessed through this [GitHub Repository](https://github.com/voidTitoMatoy/MATLAB-BPSK-Modulator-Simulation).

# Outline of progress

1. Received approval by professor for **BPSK** bandpass modulation scheme to be implemented in the DIGCOMT and LBYEC3E final projects.
2. Created an outline of the MATLAB simulation live script.
   1. Section 1: Ideal balanced modulator simulation.
   2. Section 2: BPSK complex envelope generator.
   3. Section 3: BPSK complex envelope generator using built-in functions.
3. Wrote Sections 2 and 3 of the code. Compared the performance of both systems using BER computations.
4. Wrote Section 1 of the code. Verified behavior of waveforms based on balanced modulator and BPSK theory.
5. Optimized Section 1 by decreasing the number of samples per bit. More bits can now be used as input to the BPSK system without occupying significant memory. Consequently, there is less stress on MATLAB graphics for presenting the waveforms.

# Data and results

## Section 1: Ideal balanced modulator simulation.

|  |  |
| --- | --- |
|  |  |

## Section 2: BPSK complex envelope generator.

|  |  |
| --- | --- |
|  |  |

## Section 3: BPSK complex envelope generator using built-in functions.

|  |  |
| --- | --- |
|  |  |

# Next steps

1. Present the progress report to the professor.
2. Consult with the professor on which Section is to be used in the final submission, or if both will be presented.
3. Await instructions for the final project report format and further requirements.

# Appendix: Written MATLAB live script and obtained outputs

BPSK Modulator Simulation

**Table of Contents**

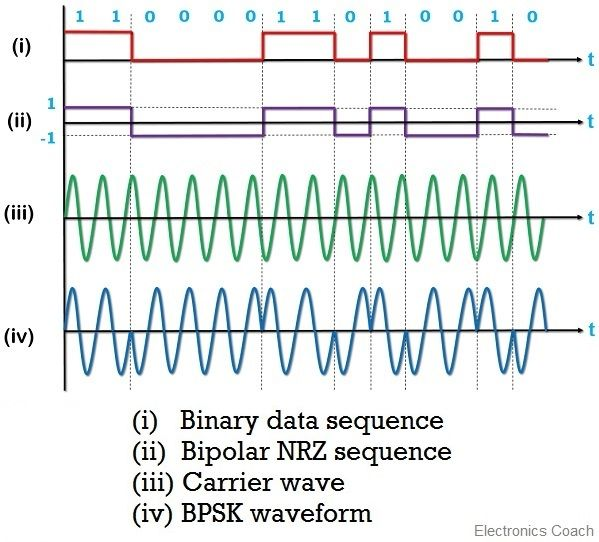
[Theoretical background](#MW_H_2691456D)   
[Ideal balanced modulator simulation](#MW_H_64F1327B)   
 [Carrier wave specifications](#MW_H_0AA15896)   
 [Binary data](#MW_H_477BC38B)   
 [Time vector](#MW_H_60B14702)   
 [Carrier wave](#MW_H_E71F5CB4)   
 [Binary waveform](#MW_H_17EF857F)   
 [NRZ waveform](#MW_H_B116473D)   
 [BPSK modulator output](#MW_H_1C31B7AB)   
 [Waveform plots](#MW_H_F07C7490)   
[BPSK complex envelope generator](#MW_H_6E820128)   
 [Binary data](#MW_H_16036615)   
 [Modulation & Complex envelope](#MW_H_FE690449)   
 [Additive white Gaussian noise](#MW_H_CB67FF5E)   
 [Constellation diagram](#MW_H_75016299)   
 [Demodulation](#MW_H_9CA8D96F)   
 [System performance: Bit error rate](#MW_H_F622C0B4)   
[BPSK complex envelope generator using built-in functions](#MW_H_CF660F93)   
 [BPSK modulation with pskmod()](#MW_H_82547389)   
 [Additive white Gaussian noise](#MW_H_0C876F6A)   
 [Constellation diagram](#MW_H_865DCC0B)   
 [Demodulation](#MW_H_820B93FA)   
 [System performance: Bit error rate](#MW_H_7A93B033)

**Theoretical background**

A diagram of a circuit

Description automatically generated with low confidence

* Above is a BPSK modulator which uses a balanced modulator.
* The carrier wave is a high frequency sinusoidal wave and the binary data input waveform is a square wave of amplitudes +1 V and -1 V corresponding to binary values 1 and 0 respectively.
* The diodes control the "orientation" of the sine wave that is passed through the transformer TR2. When the binary data input is +1 V, the BPSK output is equal to the carrier wave, or has no phase difference. When the binary data input is - 1 V, the BPSK output and the carrier wave will have a 180-degree phase difference, hence the modulation scheme "BPSK".
* The simulation will assume the transformers TR1 and TR2 specifications to be 1:1 and that there is no voltage drop across the diodes. Hence only the phase of the waveforms shall change.
* Below are the expected waveforms (from [What is Phase Shift Keying (PSK)? BPSK modulation, BPSK demodulation, advantages, disadvantages and applications of PSK - Electronics Coach](https://electronicscoach.com/phase-shift-keying.html)):



**Ideal balanced modulator simulation**

**Carrier wave specifications**

* A similar technology, IEEE 802.15. 4 (the wireless standard used by Zigbee) also relies on PSK using two frequency bands: **868–915 MHz with BPSK** and at 2.4 GHz with OQPSK. Both QPSK and 8PSK are widely used in satellite broadcasting. ([Phase-shift keying - Wikipedia](https://en.wikipedia.org/wiki/Phase-shift_keying#:~:text=A%20similar%20technology%252C%20IEEE%20802.15,widely%20used%20in%20satellite%20broadcasting.))

Ac = 10;

fc = 865.5e6;

**Binary data**

n = 16;

bits = randi([0 1], [1 n])

bits = *1×16*

0 0 1 0 1 1 1 1 0 1 0 0 0 ⋯

**Time vector**

spb = 50;

t = linspace(0, (length(bits))/fc, length(bits)\*spb);

**Carrier wave**

fs = fc\*spb;

t\_temp = 0 : 1/fs : length(bits)/fc - 1/fs;

carrier = Ac.\*sin(2.\*pi.\*fc.\*t\_temp);

**Binary waveform**

bin = zeros(1, length(bits)\*spb);

for i = 1 : length(bits)

bin((i-1)\*spb + 1 : i\*spb) = bits(i);

end

**NRZ waveform**

nrz = zeros(1, length(bits)\*spb);

for i = 1 : length(bits)

switch(bits(i))

case 1

nrz((i-1)\*spb + 1 : i\*spb) = 1;

case 0

nrz((i-1)\*spb + 1 : i\*spb) = -1;

end

end

**BPSK modulator output**

fs = fc\*spb;

BPSK = zeros(size(bin));

for i = 1 : length(bits)

switch(bits(i))

case 1

BPSK((i-1)\*spb + 1 : i\*spb) = carrier(1:spb);

case 0

BPSK((i-1)\*spb + 1 : i\*spb) = -carrier(1:spb);

end

end

**Waveform plots**

figure(1)

tiledlayout(4, 1)

sgtitle("BPSK Modulator")

nexttile

plot(t, bin, LineWidth=2)

title("Binary data sequence (" + num2str(n) + " bits)")

xlabel("Time (sec)")

ylabel("Voltage (V)")

xlim([0 max(t)])

ylim([-0.5 1.5])

nexttile

plot(t, nrz, LineWidth=2)

title("Bipolar NRZ sequence")

xlabel("Time (sec)")

ylabel("Voltage (V)")

xlim([0 max(t)])

ylim([-2 2])

nexttile

plot(t, carrier, LineWidth=2)

title("Carrier")

xlabel("Time (sec)")

ylabel("Voltage (V)")

xlim([0 max(t)])

ylim((Ac\*(3/2)).\*[-1 1])

nexttile

plot(t, BPSK, LineWidth=2)

title("BPSK-modulated waveform")

xlabel("Time (sec)")

ylabel("Voltage (V)")

xlim([0 max(t)])

ylim((Ac\*(3/2)).\*[-1 1])

A picture containing text, screenshot, font, diagram

Description automatically generated

figure(2)

tiledlayout(2, 1)

sgtitle("Balanced Modulator Input and Output")

nexttile

plot(t, nrz, LineWidth=2)

title("Bipolar NRZ sequence")

xlabel("Time (sec)")

ylabel("Voltage (V)")

for i = 1 : length(bits)-1, xline(i/fc, LineStyle='--', Color='red', LineWidth=1.5); end

xlim([0 max(t)])

ylim([-2 2])

nexttile

plot(t, BPSK, LineWidth=2)

title("BPSK-modulated waveform")

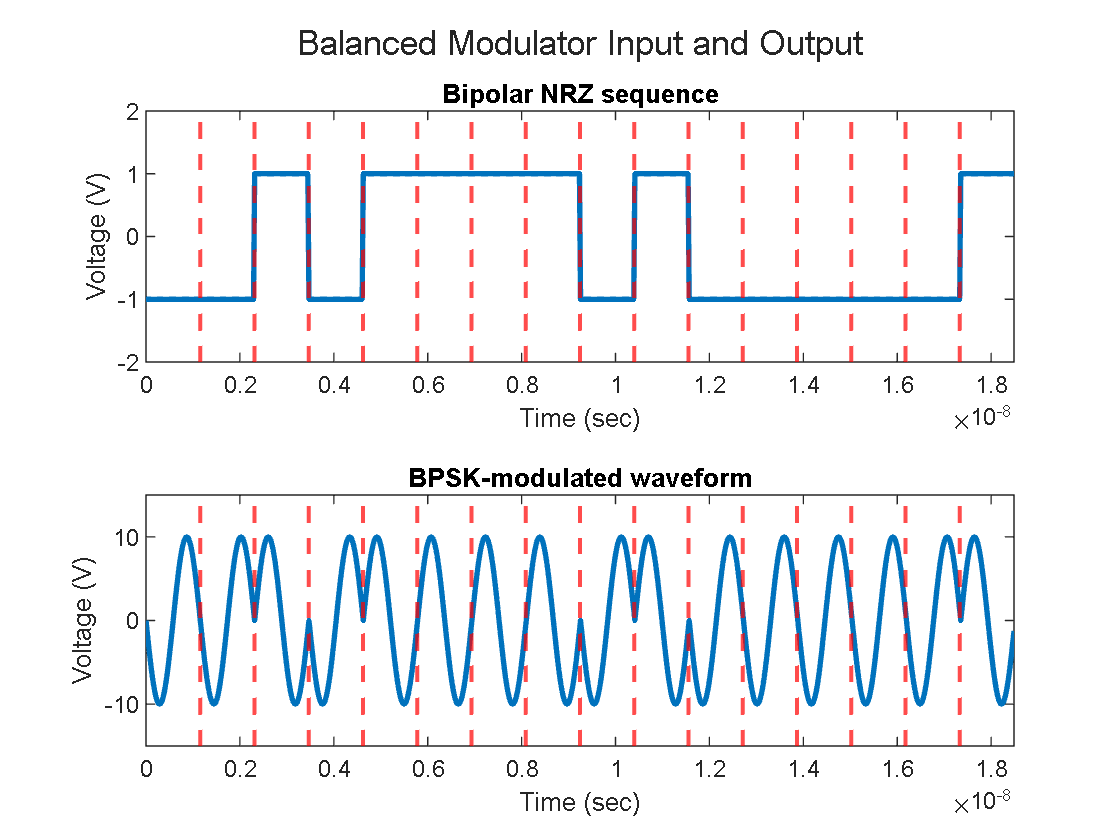
xlabel("Time (sec)")

ylabel("Voltage (V)")

for i = 1 : length(bits)-1, xline(i/fc, LineStyle='--', Color='red', LineWidth=1.5); end

xlim([0 max(t)])

ylim((Ac\*(3/2)).\*[-1 1])



**BPSK complex envelope generator**

**Binary data**

nbits = 1e6;

bits = randi([0 1], 1, nbits)

bits = *1×1000000*

0 0 1 0 1 1 1 1 1 0 1 0 0 ⋯

**Modulation & Complex envelope**

cmplx\_env = zeros(size(bits));

for i = 1 : length(bits)

switch(bits(i))

case 1

cmplx\_env(i) = -1;

case 0

cmplx\_env(i) = 1;

otherwise

error("Error in bit generation")

end

end

cmplx\_env = complex(cmplx\_env)

cmplx\_env = *1×1000000 complex*

1.0000 + 0.0000i 1.0000 + 0.0000i -1.0000 + 0.0000i 1.0000 + 0.0000i ⋯

**Additive white Gaussian noise**

SNR = 5;

noise\_cmplx\_env = awgn(cmplx\_env, SNR)

noise\_cmplx\_env = *1×1000000 complex*

0.4356 - 0.4705i 0.7464 - 0.5819i -0.9059 + 0.0916i 1.2422 + 0.1063i ⋯

**Constellation diagram**

clf

figure

scatterplot(noise\_cmplx\_env)

title("BPSK Constellation Diagram")

grid on

A yellow dot diagram on a black background

Description automatically generated with low confidence

**Demodulation**

recovered\_bits = zeros(size(bits));

for i = 1 : length(noise\_cmplx\_env)

if real(noise\_cmplx\_env(i)) < 0

recovered\_bits(i) = 1;

elseif real(noise\_cmplx\_env(i)) >= 0

% do nothing since array is already zeros

else

error("Error in complex envelope generation")

end

end

recovered\_bits

recovered\_bits = *1×1000000*

0 0 1 0 1 1 1 1 1 0 1 0 0 ⋯

**System performance: Bit error rate**

[number, ratio] = biterr(bits, recovered\_bits);

fprintf("For SNR of " + num2str(SNR) + ", the BPSK modulation system produced " + num2str(number) + ...

" number of bits in error. Resulting BER is " + num2str(ratio\*100) + " percent.")

For SNR of 5, the BPSK modulation system produced 6021 number of bits in error. Resulting BER is 0.6021 percent.

**BPSK complex envelope generator using built-in functions**

**BPSK modulation with pskmod()**

complex\_envelope = pskmod(bits, 2) % uses the same bits from previous section

complex\_envelope = *1×1000000 complex*

1.0000 + 0.0000i 1.0000 + 0.0000i -1.0000 + 0.0000i 1.0000 + 0.0000i ⋯

**Additive white Gaussian noise**

noise\_complex\_envelope = awgn(complex\_envelope, SNR)

noise\_complex\_envelope = *1×1000000 complex*

1.0669 + 0.0969i 0.7725 + 0.7520i -1.9861 + 0.3516i 0.9901 - 0.1286i ⋯

**Constellation diagram**

clf

figure

scatterplot(noise\_cmplx\_env)

title("BPSK Constellation Diagram")

grid on

A yellow dot diagram on a black background

Description automatically generated with low confidence

**Demodulation**

recovered\_bits = pskdemod(noise\_cmplx\_env, 2)

recovered\_bits = *1×1000000*

0 0 1 0 1 1 1 1 1 0 1 0 0 ⋯

**System performance: Bit error rate**

[number, ratio] = biterr(bits, recovered\_bits);

fprintf("For SNR of " + num2str(SNR) + ", the BPSK modulation system produced " + num2str(number) + ...

" number of bits in error. Resulting BER is " + num2str(ratio\*100) + " percent.")

For SNR of 5, the BPSK modulation system produced 6021 number of bits in error. Resulting BER is 0.6021 percent.