MASINDE MULIRO UNIVERSITY OF SCIENCE & TECHNOLOGY
EXPERIMENT 1 AMPLITUDE SHIFT KEYING

OBJECTIVE

The objective of this lab is to generate and plot the wave form for Binary Amplitude Shift Keying (ASK) signal using MATLAB for a stream of bits.

THEORY

Amplitude Shift Keying (ASK) is the digital modulation technique in which the amplitude of the carrier signal is varied in accordance with the modulating binary data in order to create a high frequency signal. Both frequency and phase remain constant while the amplitude changes. In ASK, the amplitude of the carrier assumes one of the two amplitudes dependent on the logic states of the input bit stream. This modulated signal can be expressed as:

$$x_e(t) = \begin{cases} 0 & symbol = 0 \\ Acos\omega_c t & symbol = 1 \end{cases}$$

ASK in the context of digital signal communications is a modulation process, which imparts to a sinusoid two or more discrete amplitude levels. These are related to the number of levels adopted by the digital message. For a binary message sequence there are two levels, one of which is typically zero. Thus the modulated waveform consists of bursts of a sinusoid. Figure 1 illustrates a binary ASK signal, together with the binary sequence which initiated it.

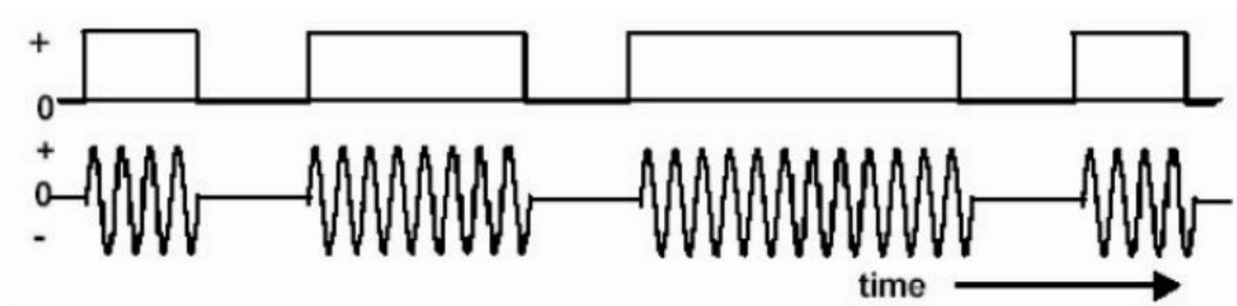


Figure 1: Amplitude Shift Modulation of a binary sequence {1 0 1 1 0 1 1 1 0 1}

EXPERIMENT

1. Start a new script file in Matlab and create the file shown below.

```
clear;
clc;
b = input('Enter the Bit stream \n '); %b = [0 1 0 1 1 1 0];
n = length(b); t = 0:.01:n;
x = 1:1:(n+1)*100;
for i = 1:n
for j = i:.1:i+1 bw(x(i*100:(i+1)*100)) = b(i);
end
end
bw = bw(100:end);
sint = sin(2*pi*t);
st = bw.*sint;
subplot(3,1,1)
plot(t,bw); grid on ;
axis([0 n -2 +2])
subplot(3,1,2)
plot(t,sint); grid on ;
axis([0 n -2 +2])
```

```
subplot(3,1,3)
plot(t,st); grid on ;
axis([0 n -2 +2])
```

<u>REPORT</u>

1. Write a flow chart of the program and run it with input bit stream: [0 1 0 1 1 1 0].
2. Repeat with input beat-stream: [1 0 1 1 0 0 0 1]
3. Using the Snipping tool, paste in your report the various graphs
4. Draw the flow-chart of the program.

EXPERIMENT 2 - FREQUENCY SHIFT KEYING

OBJECTIVE

The objective of this lab is to plot the wave form for Binary Frequency Shift Keying (BFSK) signal using MATLAB for a stream of bits.

THEORY

In frequency-shift keying, the signals transmitted for marks (binary ones) and spaces (binary zeros) are respectively.

$$x_e(t) = \begin{cases} A cos\omega_{c1} t & symbol = 0 \\ A cos\omega_{c2} t & symbol = 1 \end{cases}$$

PROCEDURE

Create the code listed below in Matlab.

```
clear;
clc;
b = input('Enter the Bit stream \n '); %b = [0 1 0 1 1 1 0]
 n = length(b);
t = 0:.01:n;
x = 1:1:(n+1)*100;
for i = 1:n
 if (b(i) == 0) b_p(i) = -1; else b_p(i) = 1;
end
for j = i:.1:i+1
 bw(x(i*100:(i+1)*100)) = b_p(i);
end
end
bw = bw(100:end);
wo = 2*(2*pi*t);
W = 1*(2*pi*t);
sinHt = sin(wo+W);
sinLt = sin(wo-W);
st = sin(wo+(bw).*W);
subplot(4,1,1)
plot(t,bw) grid on ;
axis([0 n -2 +2])
subplot(4,1,2)
plot(t,sinHt) grid on ;
axis([0 n -2 +2])
subplot(4,1,3)
plot(t,sinLt) grid on ;
axis([0 n -2 +2])
subplot(4,1,4)
plot(t,st) grid on ;
axis([0 n -2 +2]) Fs=1;
figure %pburg(st,10)
periodogram(st)
```

REPORT

1. Write a flow chart of the program and run it with input bit stream: [0 1 0 0 1 1 1 0]
2. Repeat with input beat-stream: [1 0 1 1 0 0 0 1]
3. Using the Snipping tool, paste in your report the various graphs
4. Draw the flow-chart of the program.

EXPERIMENT 3 – MINIMUM SHIFT KEYING (MSK)

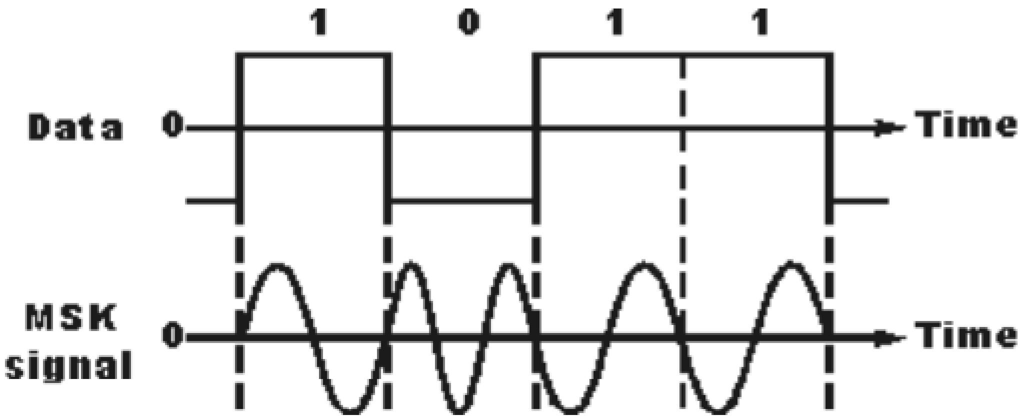
PURPOSE

The purpose of this lab is to generate a Minimum Shift Keying (MSK) signal using Matlab and to display the results.

THEORY

Minimum shift keying (MSK) is a type of continuous-phase frequency shift keying. MSK encodes each bit as a half-wave sinusoid which results in a constant-modulus signal which reduces the problems caused by non-linear distortion. As a result, MSK is sometimes referred to as a continuous phase frequency shift keying.

MSK offers advantages in terms of spectral efficiency when compared to other similar modes, and it also enables power amplifiers to operate in saturaton enabling them to provide high levels of efficiency.



PROCEDURE

Create the following code in Matlab.

```
clear;
clc;
b = input('Enter the Bit stream \n '); %b = [0 1 0 1 1 1 0];
n = length(b);
t = 0:.01:n;
x = 1:1:(n+2)*100; for i = 1:n
if (b(i) == 0)
 b_p(i) = -1;
else
b_p(i) = 1;
end
for j = i:.1:i+1
bw(x(i*100:(i+1)*100)) = b_p(i);
 if (mod(i,2) == 0)
 bow(x(i*100:(i+1)*100)) = b_p(i);
bow(x((i+1)*100:(i+2)*100)) = b_p(i);
else
bew(x(i*100:(i+1)*100)) = b_p(i);
bew(x((i+1)*100:(i+2)*100)) = b_p(i);
end
if (mod(n,2)~= 0)
bow(x(n*100:(n+1)*100)) = -1;
bow(x((n+1)*100:(n+2)*100)) = -1;
 end
end
end
bw = bw(100:end);
bew = bew(100:(n+1)*100);
bow = bow(200:(n+2)*100);
wot = 2*pi*t*(5/4);
```

```
Wt = 2*pi*t/(4*1);
st = bow.*sin(wot+(bew.*bow).*Wt);
subplot(4,1,1)
plot(t,bw)
axis([0 n -2 +2])
subplot(4,1,2)
plot(t,bow)
axis([0 n -2 +2])
subplot(4,1,3)
plot(t,bew)
axis([0 n -2 +2])
subplot(4,1,4)
plot(t,st)
axis([0 n -2 +2])
Fs=5/4;  %figure %periodogram(st)
S = fft(st,65);
PSS = S.* conj(S) / 65;
f = 1000*(-16:16)/65;
figure
plot(f,PSS(1:33))
```

REPORT

1. Write a flow chart of the program and run it with input bit stream: [0 1 0 0 1 1 1 0]
2. Repeat with input beat-stream: [1 0 1 1 0 0 0 1]
3. Using the Snipping tool, paste in your report the various graphs

EXPERIMENT 4 – PHASE SHIFT KEYING

PURPOSE

The purpose of this lab is to generate Phase Shift Keying (PSK) signal using Matlab and to display the results.

THEORY

PROCEDURE

Create the following code in Matlab.

```
clear;
clc;
b = input('Enter the Bit stream \n ');  %b = [0 1 0 1 1 1 0]
n = length(b);
t = 0:.01:n; x = 1:1:(n+1)*100; for i = 1:n
if (b(i) == 0)
 b_p(i) = -1;
else
b_p(i) = 1;
end
for j = i:.1:i+1
bw(x(i*100:(i+1)*100)) = b_p(i);
end
end
bw = bw(100:end);
sint = sin(2*pi*t);
st = bw.*sint;
subplot(3,1,1)
plot(t,bw)
axis([0 n -2 +2])
subplot(3,1,2)
plot(t,sint)
axis([0 n -2 +2])
subplot(3,1,3)
plot(t,st)
axis([0 n -2 +2])
```

PROCEDURE

Run the Matlab program above with input data stream: [0 1 0 0 1 1 1 0]

REPORT

1. Write a flow chart of the program and run it with input bit stream: [0 1 0 0 1 1 1 0]
2. Repeat with input beat-stream: [1 0 1 1 0 0 0 1]
3. Using the Snipping tool, paste in your report the various graphs

EXPERIMENT 6 – 8 QAM

PURPOSE

The purpose of this lab is to generate Quadrature Amplitude Modulation (QAM) signal using Matlab and to display the results.

THEORY

PROCEDURE

Create the following code in Matlab.

```
clc
close all
m=8;
k=log2(m);
n=9e3; nsamp=1;
x=randint(n,1);
stem(x(1:20),'filled');
title('bit sequence');
xlabel('bit index'); ylabel('bit amplitude');
xsym=bi2de(reshape(x,k,length(x)/k).','left-msb');
 figure;
stem(xsym(1:10));
title('symbol plot');
xlabel('symbol index');
ylabel('symbol amplitude');
y=modulate(modem.qammod(m),xsym);
ytx=y;
ebno=10;
snr=ebno+10*log(k)-10*log10(nsamp);
yn=awgn(ytx,snr);
yrx=yn;
scatterplot(y);
 scatterplot(yrx,30);
```

REPORT

1. Create the Matlab code above
2. Draw and explain the flow chart of the code

EXPERIMENT 7 – QUADRATURE PHASE SHIFT KEYING

```
clear;
clc;
b = input('Enter the Bit stream \n '); %b = [0 1 0 1 1 1 0];
n = length(b);
t = 0:.01:n;
x = 1:1:(n+2)*100;
for i = 1:n
if (b(i) == 0)
b_p(i) = -1;
else
b_p(i) = 1;
end
for j = i:.1:i+1
bw(x(i*100:(i+1)*100)) = b_p(i);
if (mod(i,2) == 0) bow(x(i*100:(i+1)*100)) = b_p(i);
bow(x((i+1)*100:(i+2)*100)) = b_p(i);
else
bew(x(i*100:(i+1)*100)) = b_p(i);
bew(x((i+1)*100:(i+2)*100)) = b_p(i);
end if (mod(n,2)~= 0) bow(x(n*100:(n+1)*100)) = -1;
bow(x((n+1)*100:(n+2)*100)) = -1;
end
end
end %be = b_p(1:2:end);
    %bo = b_p(2:2:end);
bw = bw(100:end);
bew = bew(100:(n+1)*100);
bow = bow(200:(n+2)*100);
cost = cos(2*pi*t);
sint = sin(2*pi*t);
st = bew.*cost+bow.*sint;
subplot(4,1,1)
plot(t,bw)
axis([0 n -2 +2])
subplot(4,1,2)
plot(t,bow) grid on ;
axis([0 n -2 +2])
subplot(4,1,3)
plot(t,bew);
axis([0 n -2 +2])
subplot(4,1,4)
plot(t,st)
 axis([0 n -2 +2])
```

EXPERIMENT 8 – QPSK WITH FADING CHANNELS

PROCEDURE

Create the Matlab code below

```
-------
clear all;
close all;
format long;
bit_count = 10000;
Eb_No = -3: 1: 30;
SNR = Eb_No + 10*log10(2);
for aa = 1: 1: length(SNR) T_Errors = 0;
T_bits = 0;
while T_Errors < 100 uncoded_bits = round(rand(1,bit_count));
B1 = uncoded_bits(1:2:end);
B2 = uncoded_bits(2:2:end);
qpsk_sig = ((B1==0).*(B2==0)*(exp(i*pi/4))+(B1==0).*(B2==1) *(exp(3*i*pi/4))+(B1==1).*(B2==1)*(exp(5*i*pi/4))
+(B1==1).*(B2==0)*(exp(7*i*pi/4))); ray = sqrt(0.5*((randn(1,length(qpsk_sig))).^2+(randn(1,length(qpsk_sig))).^2));
 rx = qpsk_sig.*ray;
N0 = 1/10^(SNR(aa)/10); rx = rx + sqrt(N0/2)*(randn(1,length(qpsk_sig))+i*randn(1,length(qpsk_sig)));
rx = rx./ray; B4 = (real(rx)

B4 = (real(rx)
B3 = (imag(rx)
uncoded_bits_rx = zeros(1,2*length(rx));
uncoded_bits_rx(1:2:end) = B3;
uncoded_bits_rx(2:2:end) = B4;
diff = uncoded_bits - uncoded_bits_rx;
 T_Errors = T_Errors + sum(abs(diff));
T_bits = T_bits + length(uncoded_bits);

end
figure;
clf; plot(real(rx),imag(rx),'o'); % Scatter Plot
title(['constellation of received symbols for SNR = ', num2str(SNR(aa))]);
xlabel('Inphase Component');
ylabel('Quadrature Component');

disp(sprintf('bit error probability = %f',BER(aa)));
end

figure(1);
semilogy(SNR,BER,'or'); hold on;
xlabel('SNR (dB)');
ylabel('BER');
title('SNR Vs BER plot for QPSK Modulation in Rayleigh Channel');

figure(1);
EbN0Lin = 10.^(Eb_No/10);
theoryBerRay = 0.5.*(1-sqrt(EbN0Lin./(EbN0Lin+1)));
semilogy(SNR,theoryBerRay); grid on;
```

```
figure(1);
theoryBerAWGN = 0.5*erfc(sqrt(10.^(Eb_No/10)));
semilogy(SNR,theoryBerAWGN,'g-+'); grid on;
legend('Simulated', 'Theoretical Raylegh', 'Theroretical AWGN');
axis([SNR(1,1) SNR(end-3) 0.00001 1]);
```

EXPERIMENT 9 – QAM WITH AWGN FADING CHANNEL

Create the following Matlab code.

```
M = 16;
SNR = 1:2:20;
bitRate = 10000;
ch = rayleighchan(1/bitRate, 10, 0 ,0);
ch.ResetBeforeFiltering = 0;
tx = randint(5000,1,M);
hmod=qammod(tx,M);
gmod = qamdemod(hmod,M);
fad = abs(filter(ch, ones(size(hmod))));
fadedSig = fad.*hmod;
BER = ones(size(SNR)); for n = 1:length(SNR) rxSig = awgn(fadedSig,SNR(n),'measured');
rx = qamdemod(rxSig,M);
[nErrors, BER(n)] = biterr(tx,rx);
end
semilogy(SNR,BER,'r-*');
xlabel('SNR (dB)');
 ylabel('BER'); grid on
```