Лабораторная работа №6

Задание 1

```
clear;
clc;
%%
Fs = 100000;     % Частота дискретзации
ts = 0 : 1 / Fs : 0.05 - 1 / Fs;     % Временные отсчеты
N = length (ts);     % Количество временных отсчетов
%%
x = cos (2 * pi * 400 * ts);     % Анализируемый действительный
сигнал
X = fft(x);                      % БПФ

fp = 1 : floor(N/2);             % Разделение отсчетов на
положительные и отрицательные
fn = floor(N/2) + 1 : N;

S(fn) = X(fn)*0;                 % (значения функции в первых
коэф-х усилим, во вторых - обнулим)
S(fp) = X(fp)*2;

s = ifft (S);                    % Обратное преобразование
Фурье

%%
figure;
plot ( real (s)), grid on , hold on
plot ( imag (s)), grid on
xlabel ('Время')
ylabel ('Амплитуда')
legend ({ 'Действительная часть - исходный сигнал';'Мнимая
часть - сигнал, сдвинутый по фазе на 90*'})

figure;
plot3(real (s), imag(s), ts);
xlabel ('Исходный сигнал')
ylabel ('Сигнал, сдвинутый по фазе на 90*')
zlabel('Время')
```

Аналитический сигнал это объемный сигнал, проекциями которого являются **исходный** исследуемый сигнал и его **копия**, сдвинутая по фазе на **90 градусов**. Для получения сдвинутого по фазе сигнала, проведем проеобразование Гильберта (не встроенная функция Hilbert, а самописная): проведем БПФ (**10**), разделим отсчеты на положительные и отрицательные (**12-14**), значение в положительных отсчетах умножим на два, в отрицательных обнулим (**16-18**), возьмем ОПФ (обратное преобразование Фурье) (**20**).

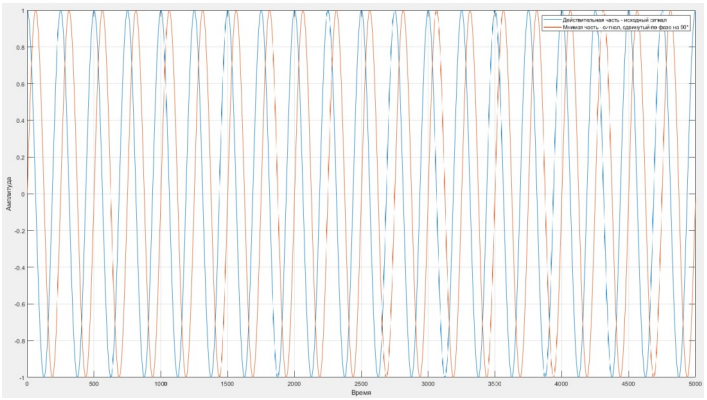


Рисунок 1 – исходный сигнал и его копия, сдвинутая по фазе на 90 градусов

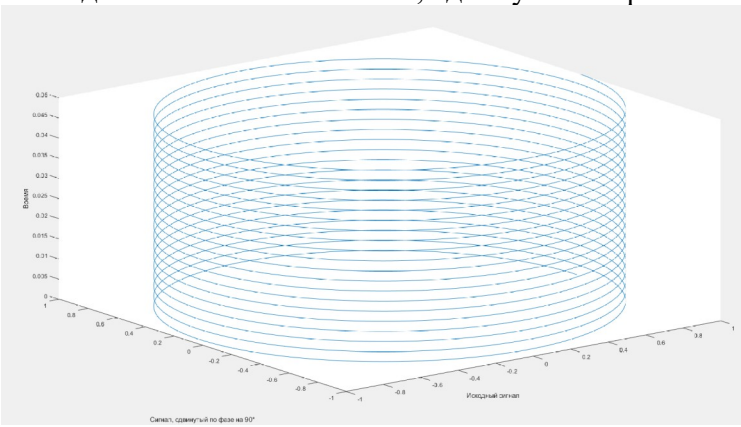


Рисунок 2 – аналитический сигнал

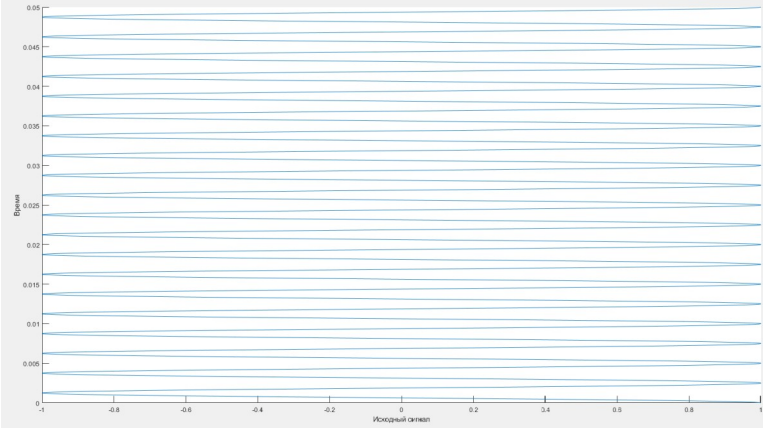


Рисунок 3 – проекция «исходный сигнал-время»

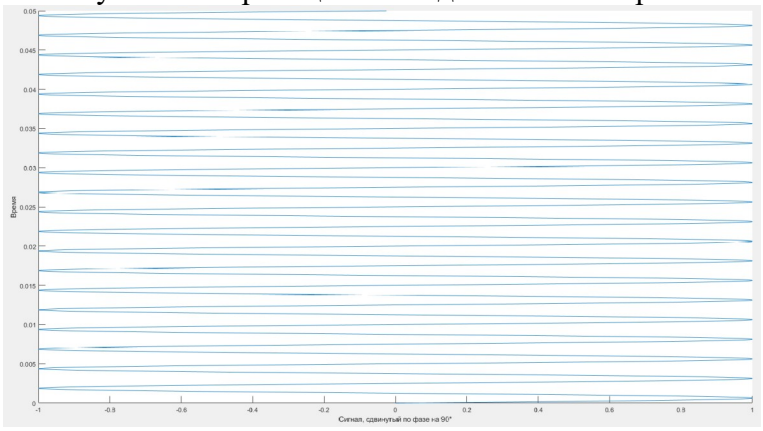


Рисунок 3 – проекция «сигнал, сдвинутый по фазе на 90 градусов-время»

Задание 2

```
clear;
clc;

fs = 10000;
ts = -0.1 : 1/ fs : 0.1 -1/ fs;
N = length (ts);
%%
fc = cos (2* pi *500* ts);
%%
n_for_bit = 200;

code = [1 0.1 1 0.1 1 0.1 1 0.1 1 0.1];

fm = zeros (1,N);
for i=1: length ( code )
   for j= n_for_bit *(i -1) +1: n_for_bit *i
       fm(j) = code (i);
   end
end
%%
x = fc .* fm;

scatterplot ( hilbert (x),n_for_bit , round ( n_for_bit /2)),
grid on

h = hilbert (x);

% компаратор
dem = zeros (1, length (h));
for i=1: length (h)
   if abs(h(i)) >=0.6
       dem (i) = 1;
   else
       dem (i) = 0;
   end
end

figure;
plot (ts ,dem , 'LineWidth' ,2); grid on;
title ('Код демодулированного сигнала');
xlabel ('Время'), ylabel ('Амплитуда');
```

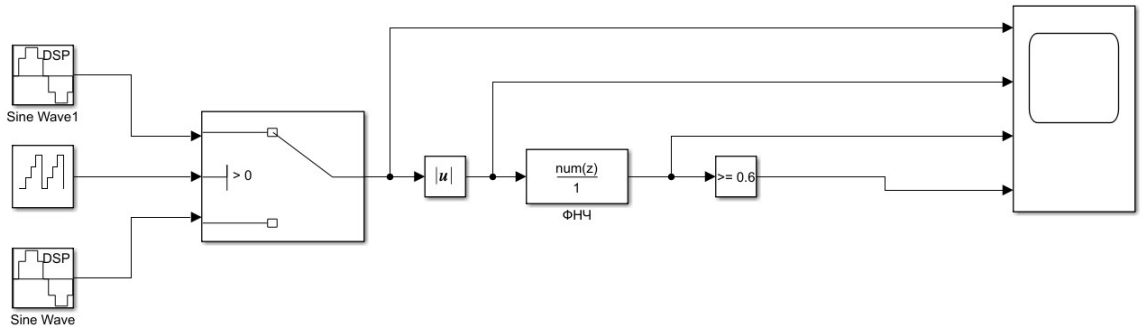


Рисунок 4 – Блок-схема ASK-модулятора в Simulink

Чтобы реализовать амплитудную модуляцию в simulink, был взят переключатель, который по сигналу, приходящему с 1-битного счетчика будет переключать выход с входа 1 на вход 2 и наоборот. К входам подключены два источника синусоид с разными амплитудами. Таким образом, на выходе переключателя будет график, показанный в верхнем поле осциллографа.

Далее, сигнал был пропущен через блок получения модуля (частота сигналов увеличилась вдвое, все отрицательные значения отражены вверх), что показано на 2 поле осциллографа (**сверху вниз**).

После этого, сигнал был пропущен через ФНЧ (3 поле осциллографа) и блок сравнения по условия x > 0.8 (4 поле осциллографа).

Характеристики ФНЧ показаны на рисунке 6, сигнальное созвездие на рисунке 7

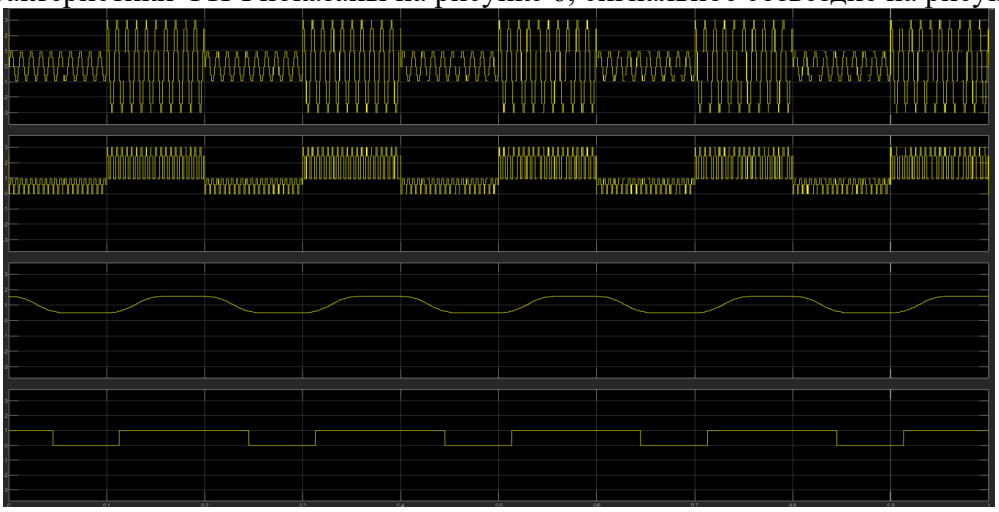


Рисунок 5 – осциллограммы ASK-модулятора на разных этапах

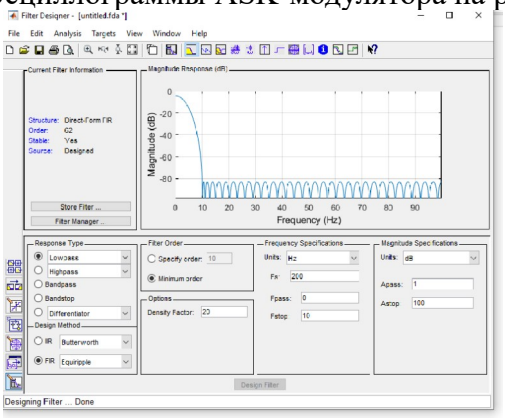


Рисунок 6 – характеристики ФНЧ

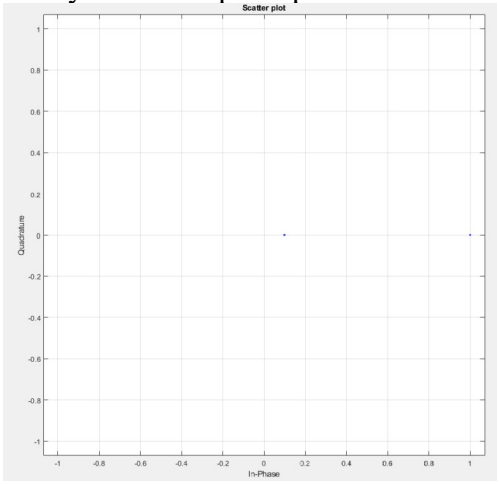


Рисунок 7 – сигнальное созвездие ASK-модулятора

Задание 3

```
% clear;
clc;

fs = 10000;
ts = 0 : 1 / fs : 0.2 -1/ fs;
N = length (ts);
%% несущая частота
fc = 1000;
%% Модуляция
M = 8; % порядок QASK модуляции
Nd = 200; % размер данных
bit_size = N/Nd; % количество отсчётов на бит
data = randi ([0 M-1] ,Nd ,1); % случайные данные

figure;
plot(data);
% формируем массив комплексных чисел
pskdata = pskmod (data , M);
% сигнальное созвездие
sc = scatterplot ( pskdata ); grid on
obj = findobj (sc. Children (1). Children , 'type', 'line');
set (obj , 'MarkerSize', 20)
pskmod = repelem (pskdata , bit_size ).';
% формируем синусный и косинусный сигналы
i = real (pskmod).*cos(2* pi*fc*ts);
q = imag (pskmod).*sin(2* pi*fc*ts);
% суммируем i и q, получаем сигнал, готовый к передаче
y = i+q;
% добавляем шум
y = awgn (y ,20) ;
figure
subplot (3 ,1 ,1)
plot ( real ( pskmod )), grid on
title ('Действительная часть сигнала')
xlabel ('Время'), ylabel ('Real ')
subplot (3 ,1 ,2)
plot ( imag ( pskmod )), grid on
title ('Мнимая часть сигнала')
xlabel ('Время'), ylabel ('Imag')
subplot (3 ,1 ,3)
plot (y), grid on
title ('Модулированный сигнал')
xlabel ('Время'), ylabel ('Амплитуда')

figure;
back = pskdemod (pskdata , M);
plot(back)
```

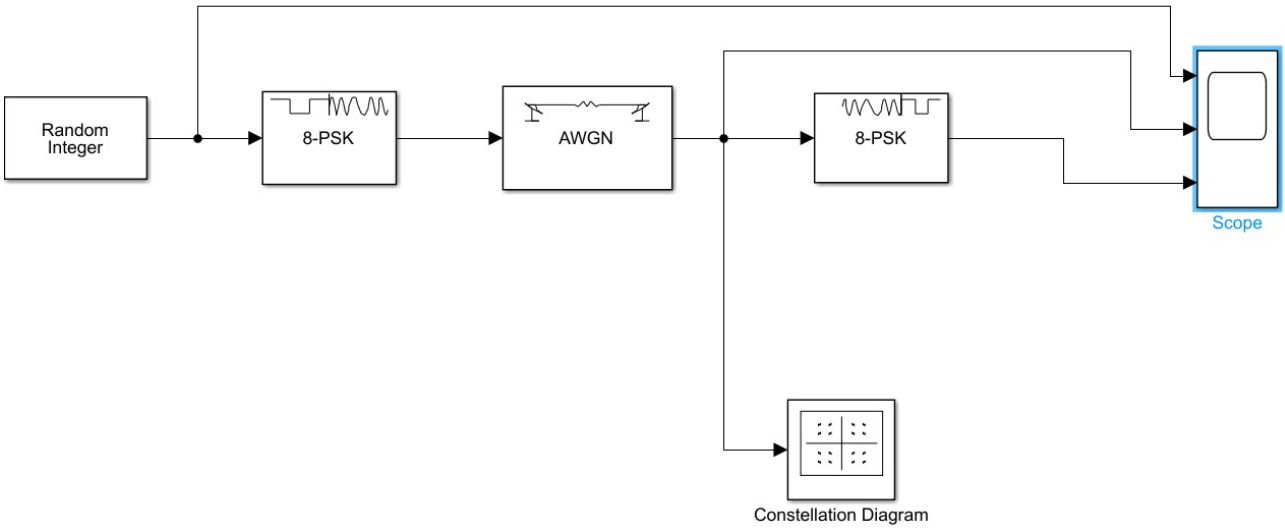


Рисунок 7 – Блок-схема PSK-модулятора в Simulink

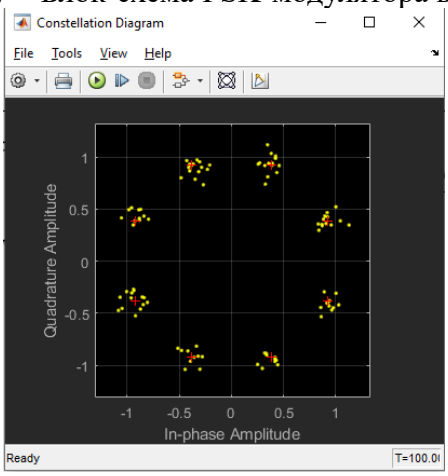


Рисунок 8 – сигнальное созвездие 8-PSK-модулятора

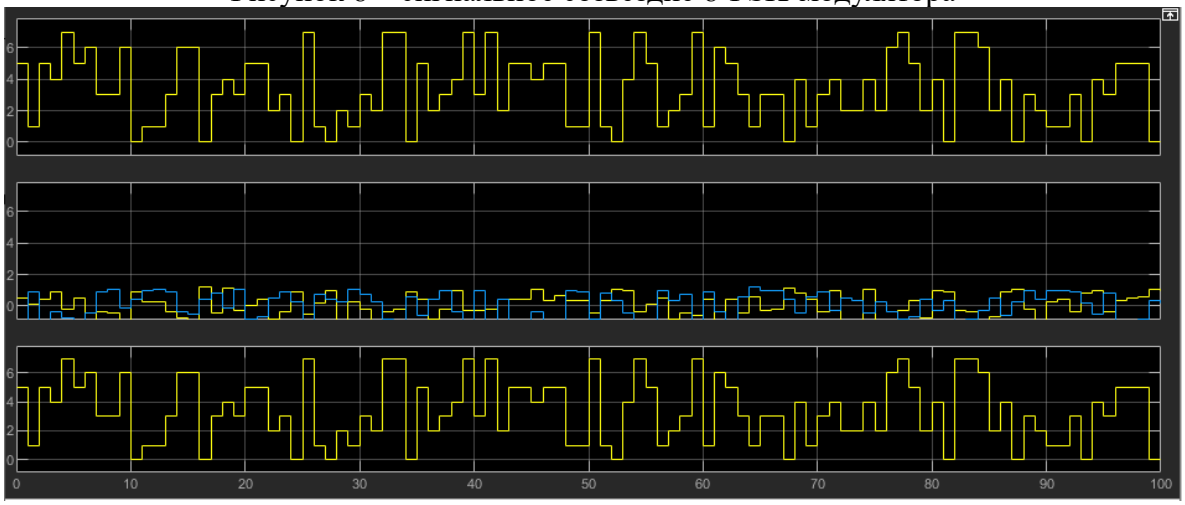


Рисунок 9 – осциллограммы PSK-модулятора на разных этапах

Данные для последующей модуляции и демодуляции генерируются блоком случайных целых чисел. (1 поле осциллограммы сверху-вниз). После этого проводим PSK-модуляцию (2 поле осциллограммы). Добавляем шумы. Величина шумов зависит от величины коэффициента Eb/No в настройках блока шума. Дале демодулируем сигнал и выводим полученный сигнал (3 поле осциллографа).

Скриншоты результатов работы скрипта Matlab:

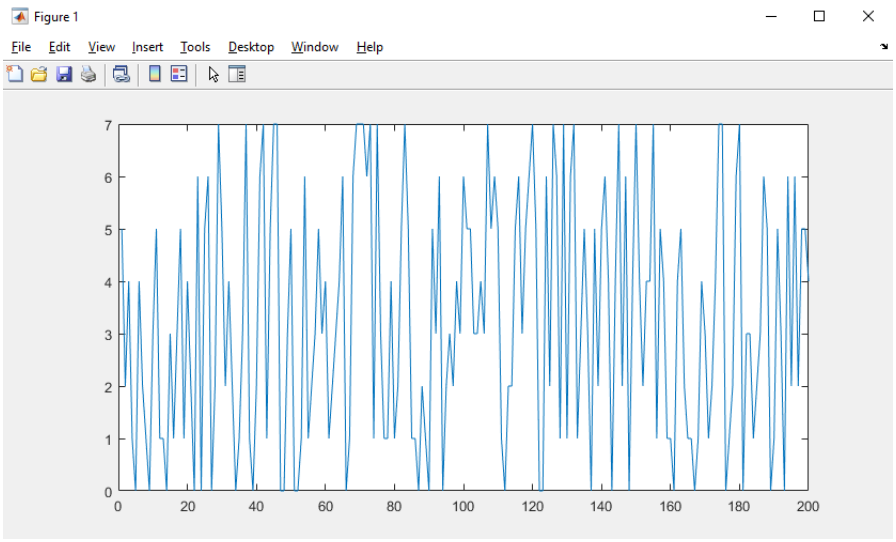


Рисунок 10 – Исходный сигнал

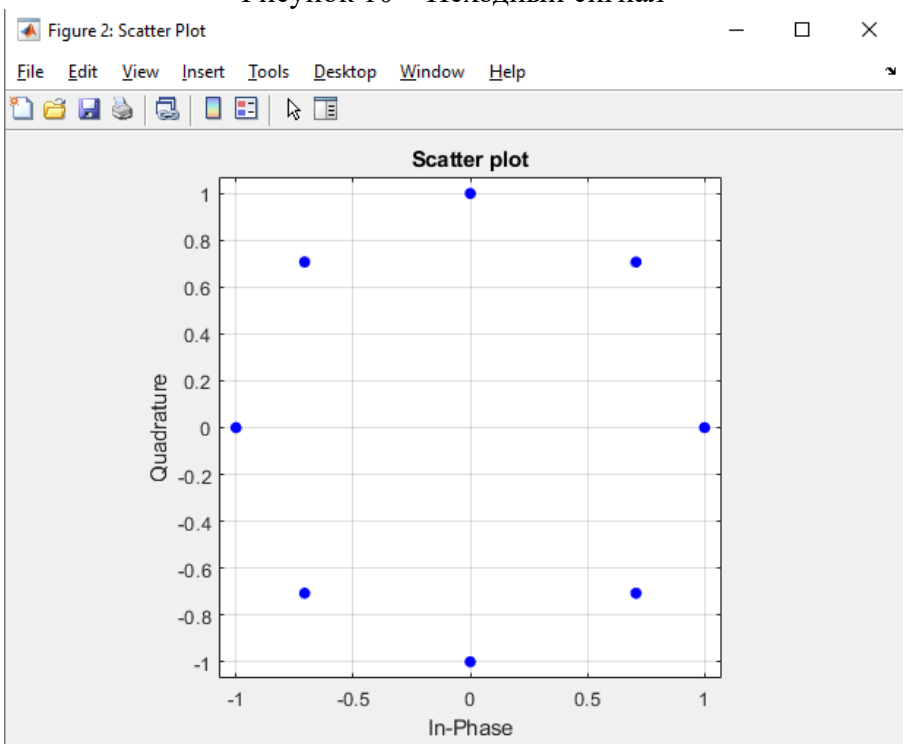


Рисунок 11 – Сигнальное созвездие 8-PSK-модулятора

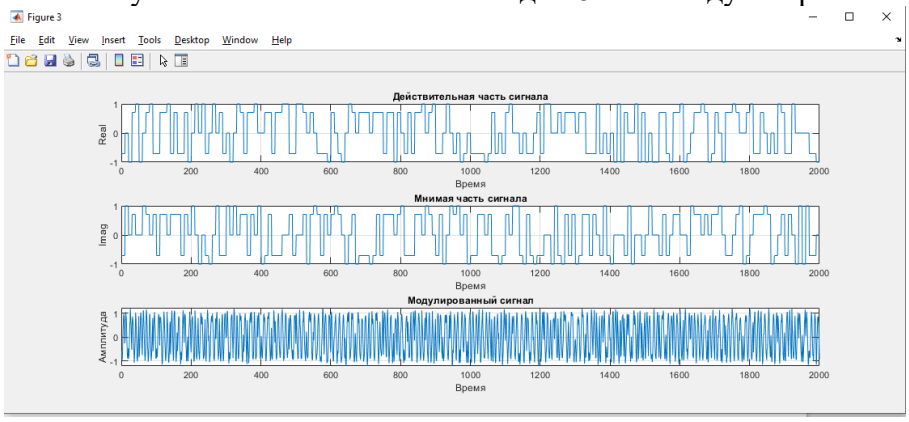


Рисунок 12 – Действительная часть модулированного сигнала (сверху), мнимая часть модулированного сигнала (посередине), модулированный сигнал (снизу)

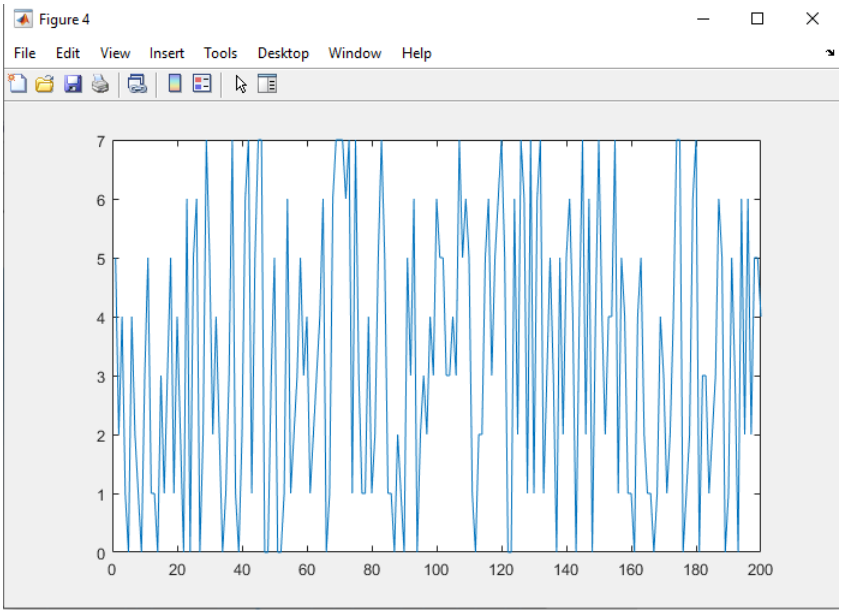


Рисунок 13 – демодулированный сигнал

Задание 4

```
% clear;
clc;

fs = 10000;
ts = 0 : 1 / fs : 0.2 -1/ fs;
N = length (ts);
%% несущая частота
fc = 1000;
%% Модуляция
M = 64; % порядок QASK модуляции
Nd = 200; % размер данных
bit_size = N/Nd; % количество отсчётов на бит
data = randi ([0 M-1] ,Nd ,1); % случайные данные

figure;
plot(data);
% формируем массив комплексных чисел
qdata = qammod (data , M);
% сигнальное созвездие
sc = scatterplot ( qdata ); grid on
obj = findobj (sc. Children (1). Children , 'type', 'line');
set (obj , 'MarkerSize', 20)
qmod = repelem (qdata , bit_size ).';
% формируем синусный и косинусный сигналы
i = real (qmod).*cos(2* pi*fc*ts);
q = imag (qmod).*sin(2* pi*fc*ts);
% суммируем i и q, получаем сигнал, готовый к передаче
y = i+q;
% добавляем шум
y = awgn (y ,20) ;
figure
subplot (3 ,1 ,1)
plot ( real ( qmod )), grid on
```

```
34 title ('Действительная часть сигнала')
35 xlabel ('Время'), ylabel ('Real ')
36 subplot (3 ,1 ,2)
37 plot ( imag ( qmod )), grid on
38 title ('Мнимая часть сигнала')
39 xlabel ('Время'), ylabel ('Imag')
40 subplot (3 ,1 ,3)
41 plot (y), grid on
42 title ('Модулированный сигнал')
43 xlabel ('Время'), ylabel ('Амплитуда')
44
45 figure;
46 back = qamdemod (qdata , M);
47 plot(back)
```

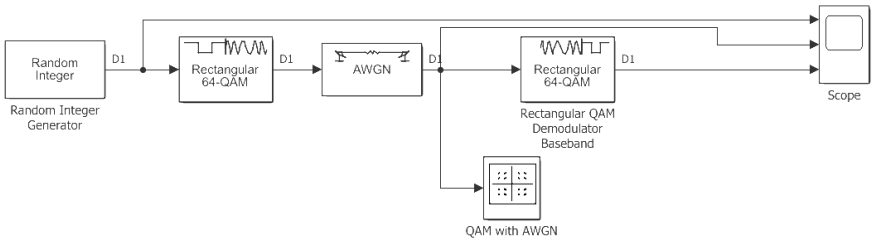


Рисунок 10 – Блок-схема QAM-модулятора в Simulink

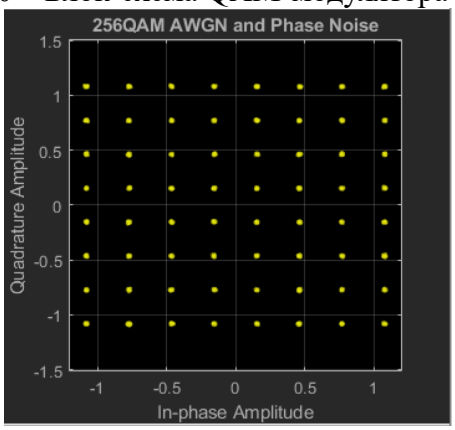


Рисунок 11 – сигнальное созвездие 8-QAM-модулятора

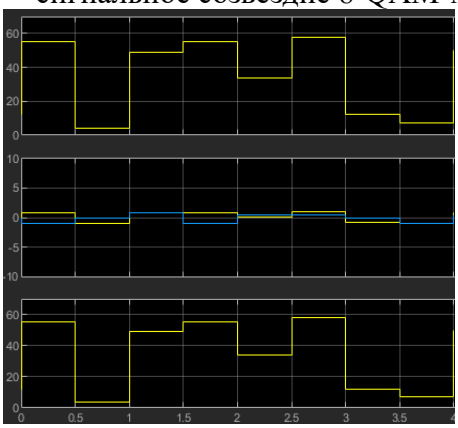


Рисунок 12 – осциллограммы QAM-модулятора на разных этапах

Скриншоты результатов работы скрипта Matlab:

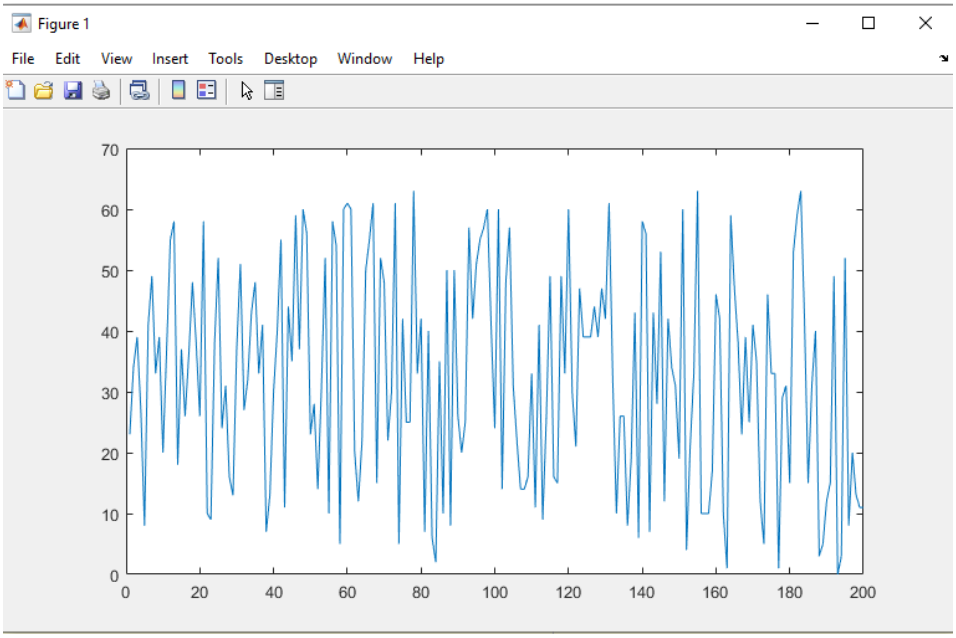


Рисунок 13 – исходный сигнал

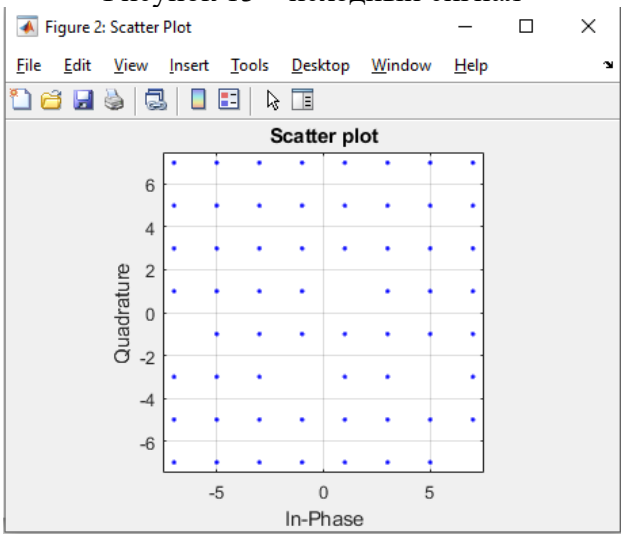


Рисунок 14 – сигнальное созвездие

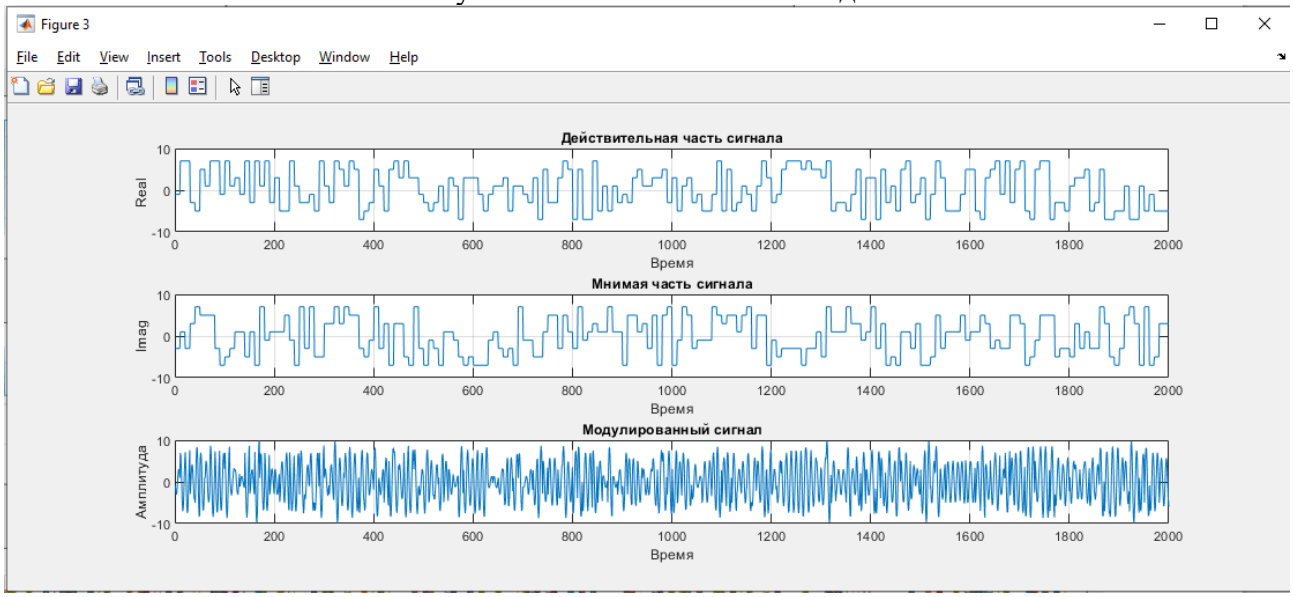


Рисунок 15 – Действительная часть модулированного сигнала (сверху), мнимая часть модулированного сигнала (посередине), модулированный сигнал (снизу)

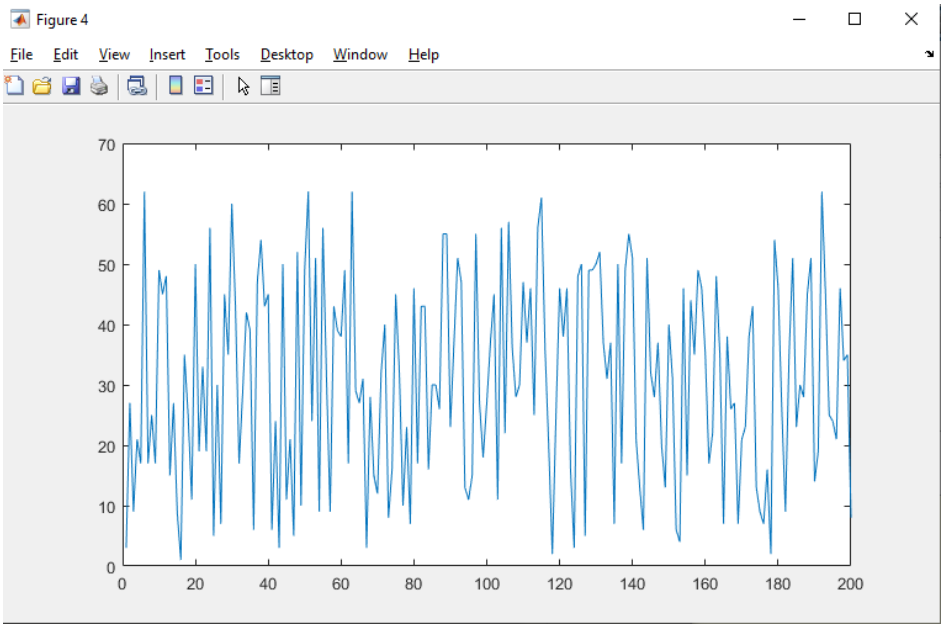


Рисунок 16 – Демодулированный сигнал

Задание 5

```
 1 clear;
 2 clc;
 3
 4 fs = 1000;
 5 ts = 0 : 1/fs : 2 - 1/fs;
 6 N = length (ts);
 7
 8 h = 4;
 9 T = 0.1;
10 f0 = 60;
11 f1 = 100;
12
13n_for_bit = T * fs;
14
15 code = [1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0];
16
17 fm = zeros (1,N);
18 for i=1: length ( code )
19     for j= n_for_bit *(i -1) +1: n_for_bit *i
20         fm(j) = code (i);
21     end
22 end
23
24 x0 = sin(2*pi*f0*ts);
25
26 x1 = sin(2*pi*f1*ts);
27
28 x = zeros (1, N);
29 for i = 1:N
30     if fm(i) == 0
31         x(i) = x0(i);
32     else
33         x(i) = x1(i);
```

```
34     end
35 end
36
37 plot (ts ,x, 'LineWidth' ,0.5) , grid on , hold on
38 plot (ts ,fm , 'LineWidth' ,2) , grid on
39 title ('FSKмодуляция- ')
40 xlabel ('Время'), ylabel ('Амплитуда')
41 legend ({ 'Модулированный сигнал';'Модулирующий сигнал'})
42 y0 = x.* x0;
43 y1 = x.* x1;
44 figure ;
45 subplot (2 ,1 ,1)
46 plot (ts ,y0), grid on
47 title ('Модулированный сигнал, умноженный на f0 ')
48 xlabel ('Время'), ylabel ('Амплитуда')
49 subplot (2 ,1 ,2)
50 plot (ts ,y1), grid on
51 title ('Модулированный сигнал, умноженный на f1 ')
52 xlabel ('Время'), ylabel ('Амплитуда')
53 y = y1 - y0;
54
55 z = filter ( Num ,1,y);
56 figure
57 subplot (2 ,1 ,1)
58 plot (ts ,y), grid on
59 title ('Сигнал y=y0 -y1 ')
60 xlabel ('Время'), ylabel ('Амплитуда')
61 subplot (2 ,1 ,2)
62 plot (ts ,z), grid on
63 title ('Этот же сигнал после ФНЧ')
64 xlabel ('Время'), ylabel ('Амплитуда')
65
66 dem = zeros (1, length (z));
67 for i=1: length (z)
68     if z(i) >=0.1
69         dem (i) = 1;
70     else
71         dem (i) = 0;
72     end
73 end
74 figure
75 plot (ts ,x, 'LineWidth' ,0.5) , grid on , hold on
76 plot (ts ,dem , 'LineWidth' ,2) , grid on
77 xlabel ('Время'), ylabel ('Амплитуда')
78 title ('FSKдемодуляция- ')
79 legend ({ 'Модулированный сигнал';'Демодулированный сигнал'})
```

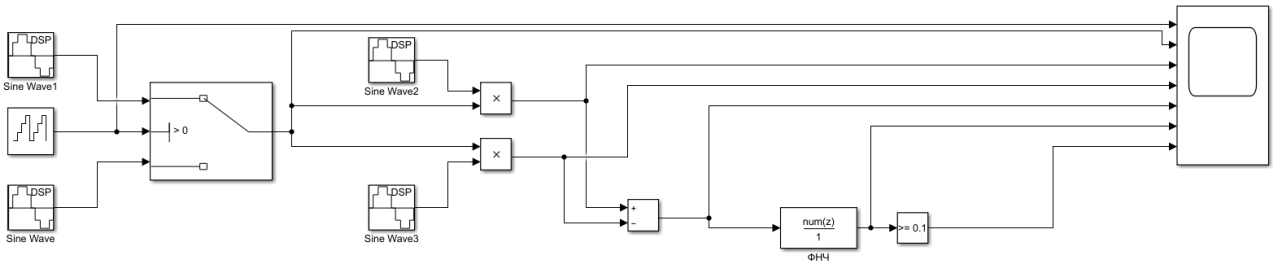


Рисунок 13 – Блок-схема fsk-модулятора в Simulink

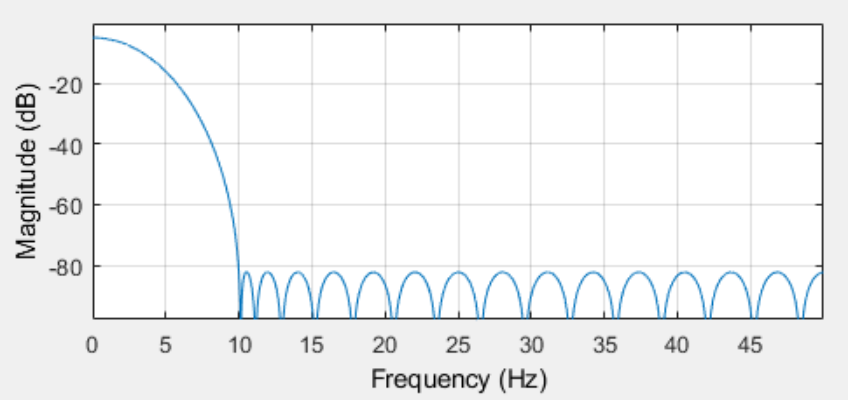


Рисунок 14 – характеристики ФНЧ

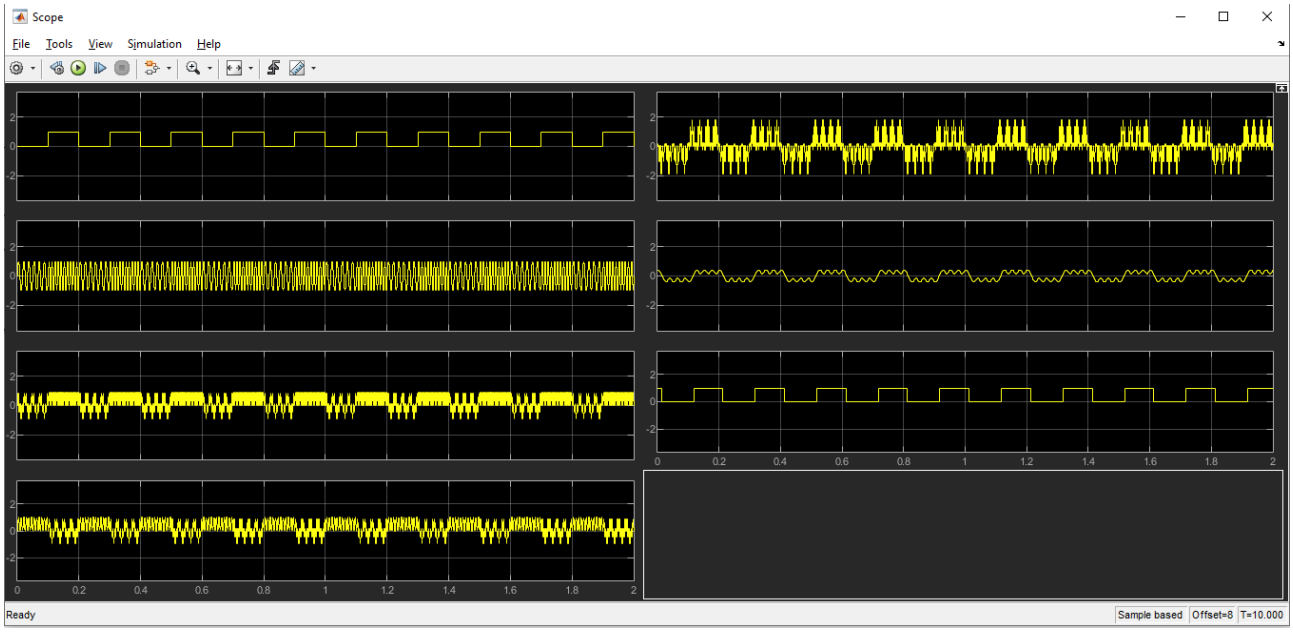
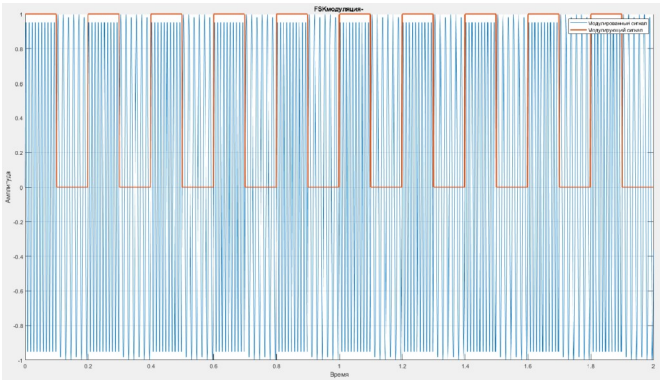
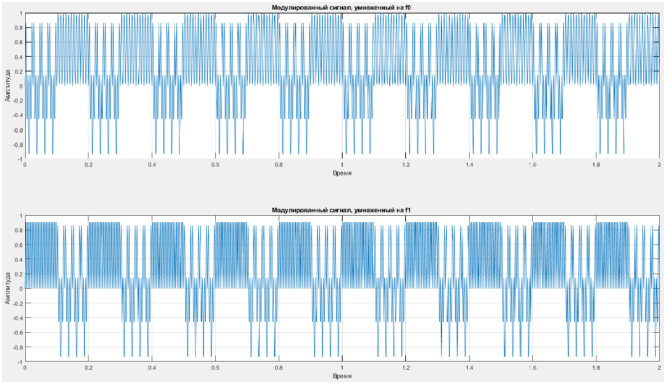


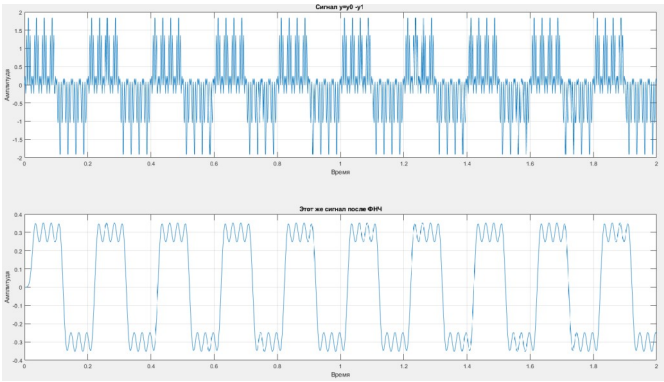
Рисунок 15 – осциллограммы QAM-модулятора на разных этапах

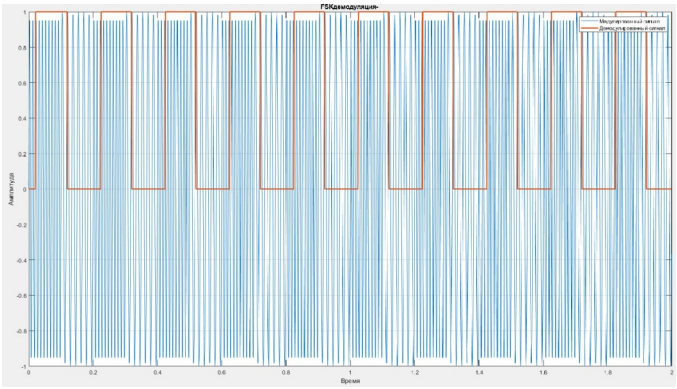
Цифровая последовательность, которую необходимо закодировать, генерируется 1-битовым счетчиком (1 поле осциллографа сверху-вниз слева-направо). Первая половина схемы такая же, как и у ASK-модулятора: происходит переключение выхода с одной синусоиды на другую (2 поле осциллографа сверху-вниз слева-направо). Разница лишь в настройках блока синусоид: в ASK-модуляции у двух сигналов была разная амплитуда, здесь же будет разная частота. После этого домножаем полученный сигнал на две такие же синусоиды (3 и 4 поле осциллографа). После этого вычитаем сигнал 3 поля осциллографа из сигнала 2 поля (5 поле осциллографа). Далее применяем ФНЧ, характеристики которого показаны на рисунке 14. Полученный сигнал показан на 6 поле осциллографа. И пропускаем сигнал через блок сравнения по уровню 0.1 (7 поле осциллографа).

Скриншоты результатов работы скрипта Matlab:


FSKмодуляция-
Модулированный сигнал
Модулирующий сигнал
Амплитуда
Время


Модулированный сигнал, умноженный на f0
Амплитуда
Время
Модулированный сигнал, умноженный на f1
Амплитуда
Время


Сигнал y=y0 -y1
Амплитуда
Время
Этот же сигнал после ФНЧ
Амплитуда
Время


FSKдемодуляция-
Модулированный сигнал
Демодулированный сигнал
Амплитуда
Время