Сети и системы передачи данных

Алексюк А.О., Мурашко Д.С.

2 іюня 2014 г.

Оглавление

Глава 2

Система верстки TEX и расширение LATEX

2.1 Цель работы

Изучение принципов работы TEXи создание первого отчета.

2.2 Ход работы

Создание документа Для создания нового документа проще всего воспользоваться инструментом "Быстрый старт"в TeXMaker. С помощью этого инструмента можно выбрать класс документа, размер бумаги, кодировку и т.д. С тем же успехом можно создать простой текстовый файл и вручную задать эти параметры с помощью директив documentclass и usepackage.

Компиляция в командной строке Для сборки документа нужно выполнить команду latex report.tex, где report.tex - путь к собираемому файлу. В результате сборки появится файл с расширением dvi, который можно просмотреть, например, в xdvi. Для получения PDF необходимо либо воспользоваться dvips, а затем ghostscript, либо сразу запускать pdflatex вместо latex.

То же самое можно сделать прямо в TeXMaker через меню "Инструменты".

Нумерованный список

1. Элемент списка

2. блаблабла

Формулы

$$\int_2^\infty blablabla$$

$$D = \frac{n^2}{\phi}$$

$$c^2 = a^2 + b^2\ \Omega + \omega$$

- $\sum\limits_{n=0}^{N-1} a_0 q^n = a_0 \dfrac{q^N - 1}{q - 1}$
- $\Phi_{\text{ст}} = \Phi_{\text{д}}(f) \dfrac{1}{T_2} \Phi_{\text{П}}(f) = \dfrac{\sin(\pi f T_2)}{\pi f T_2} \sum\limits_{k=-\infty}^{\infty} \Phi(f - k f_2)$

2.3 Вывод

LaTeX — довольно удобный инструмент для создания документов. Из положительных сторон можно отметить простой набор формул и красивый шрифт по умолчанию. Проблем с синтаксисом LaTeX не возникло.

Глава 4

Визуализация сигналов во временной и частотной области

4.1 Цель работы

Познакомиться со средствами генерации сигналов и визуализации их спектров.

4.2 Постановка задачи

В командном окне MATLAB и в среде Simulink промоделировать чистый синусоидальный сигнал, а также синусоидальный сигнал с шумом. Получить их спектры.

4.3 Теоретическая часть

Преобразование Фурье в интегральной форме:

$$\hat{f}(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x) e^{-ix\omega}\, dx$$

Дискретное преобразование Фурье:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn} \qquad k = 0, \ldots, N-1$$

Обратное преобразование:

$$x_n = \frac{1}{N}\sum_{k=0}^{N-1} X_k e^{\frac{2\pi i}{N}kn} \qquad n = 0, \ldots, N-1.$$

4.4 Ход работы

4.4.1 Код Matlab

```
close all;
clear all;
x = 0:0.01:4*pi;
f0 = 5;
%исходный сигнал
y = sin(2*pi*f0*x);
plot(x(1:200),y(1:200))
grid
%спектр исходного сигнала
figure
spectrum = abs(fft(y,128));
norm_spectrum = spectrum.*conj(spectrum)/512;
f=100*(0:127)/512;
plot(f, norm_spectrum(1:128))
axis([0 max(f) 0 10])
grid
```

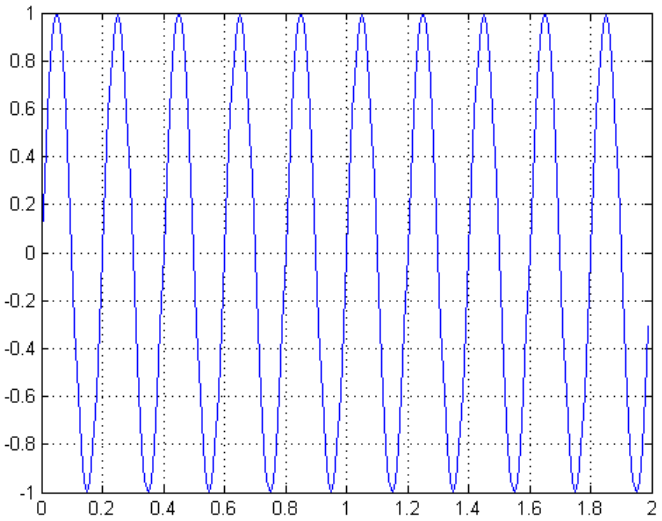

Рис. 4.1: Синусоида

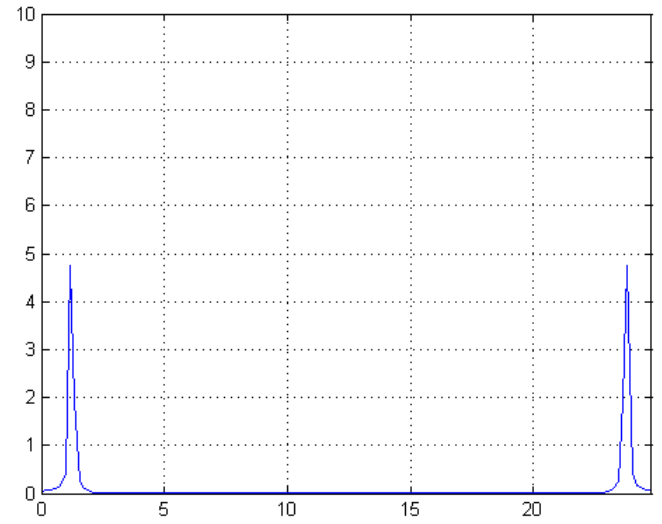

Рис. 4.2: Спектр

4.4.2 Simulink

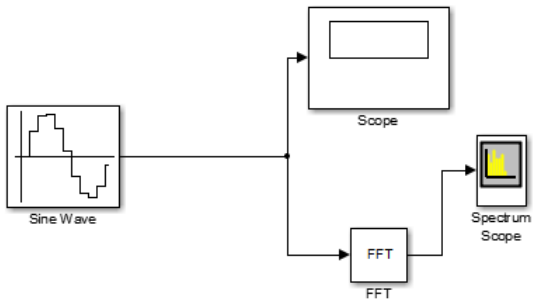

Рис. 4.3: Проект Simulink

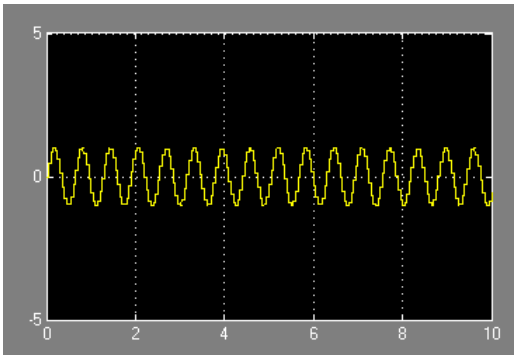

Рис. 4.4: Синус

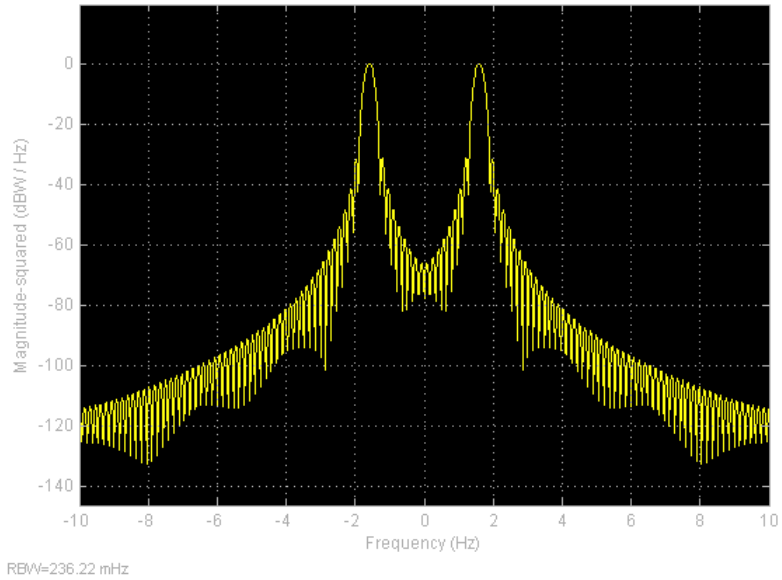

Рис. 4.5: Спектр

4.5 Вывод

В ходе работы были исследованы два важных нюанса, касающиеся применения преобразования Фурье в цифровых системах.

Во-первых, если бы входной сигнал был бесконечным, то его спектр представлял бы собой бесконечно короткий импульс, с бесконечно большой амплитудой и единичной площадью, однако на практике из-за ограниченной длины сигнала (бесконечный сигнал был как бы умножен на прямоугольный импульс конечной длины) произошла свертка со спектром прямоугольного окна.

Во-вторых, так как сигнал был дискретизирован, он умножается на решетку дельта-импульсов. Спектр решетки дельта-импульсов - также решетка дельта-импульсов, и спектр нашего сигнала сворачивается также с решеткой дельта-импульсов. Из-за этого импульс повторяется бесконечное число раз с периодом равным частоте дискретизации (на графике видно два импульса).

Глава 5

Спектры простых сигналов

5.1 Цель работы

Получить представление о свойствах спектров.

5.2 Постановка задачи

В командном окне MATLAB и в среде Simulink промоделировать следующие тестовые сигналы:

- Полигармонический сигнал
- Прямоугольный импульсный сигнал
- Треугольный импульсный сигнал

Получить спектров этих сигналов.

5.3 Теоретическая часть

- Полигармонический сигнал y(t)=$\sum_{n=0}^{N-1} cos(nt)$
- Прямоугольный импульсный сигнал y(t)=$\prod(t, Ti)$
- Треугольный импульсный сигнал y(t)=$\Delta(t, Ti)$

5.4 Ход работы

5.4.1 Код Matlab

```
close all;
clear all;
x = 0:0.01:4*pi;
f0 = 5;
%исходный сигнал
%for n=1:length(x)
%   y(n) = cos(2*pi*f0*n*x(n));
%end;
for n=1:length(x)
   if mod(n,20) < 10
        y(n) = 0;
   else
        y(n) = 1;
   end;
end;
figure
plot(x(1:200),y(1:200))
axis([-0.5 2.5 -0.5 1.5])
y = conv(y,y);
for n=1:length(x)
    y(n) = y(n)/n;
end;
figure
plot(x(1:200),y(1:200))
grid
%спектр исходного сигнала
figure
spectrum = fft(y,512);
norm_spectrum = spectrum.*conj(spectrum)/512;
f=100*(0:255)/512;
plot(f, norm_spectrum(1:256))
axis([0 max(f) 0 1000])
grid

clear all;

t = 0:00.1:4*pi;
```

```
N = 100;
y = 0;

% Полигармонический сигнал
y = sin(pi*t)+sin(3*pi*t)+sin(pi*0.3*t);
plot(t,y)
grid

figure
spectrum = fft(y,512);
norm_spectrum = spectrum.*conj(spectrum)/512;
f=100*(0:255)/512;
plot(f, norm_spectrum(1:256))
axis([0 max(f) 0 10])
grid

% Треугольный сигнал
figure
y2 = conv(square(t,20),square(t,20));
plot(t(1:100),y2(1:100));
grid

figure
spectrum = fft(y2,512);
norm_spectrum = spectrum.*conj(spectrum)/512;
f2=100*(0:255)/512;
plot(f2, norm_spectrum(1:256)/1000)
axis([0 max(f2) 0 50])
grid
```

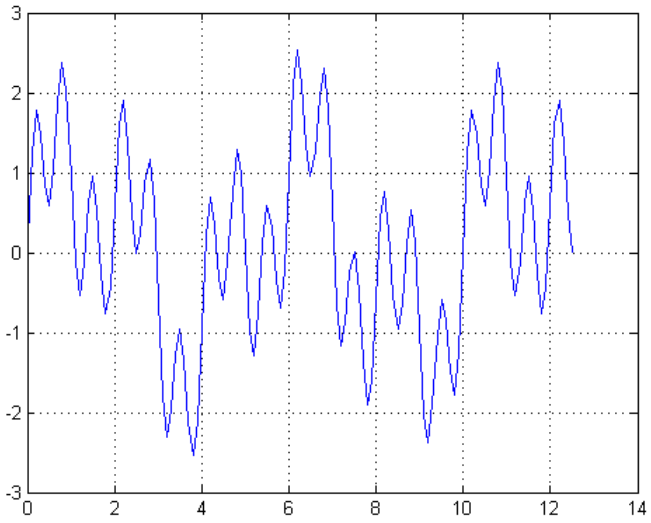

Рис. 5.1: Полигармонический сигнал

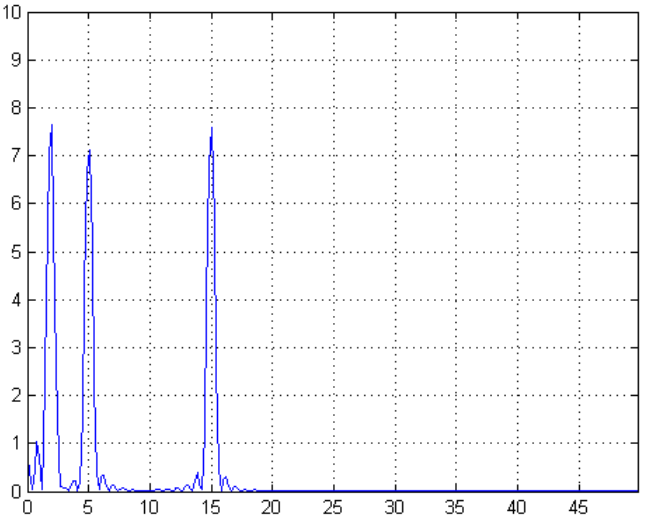

Рис. 5.2: Спектр gолигармонического сигнала

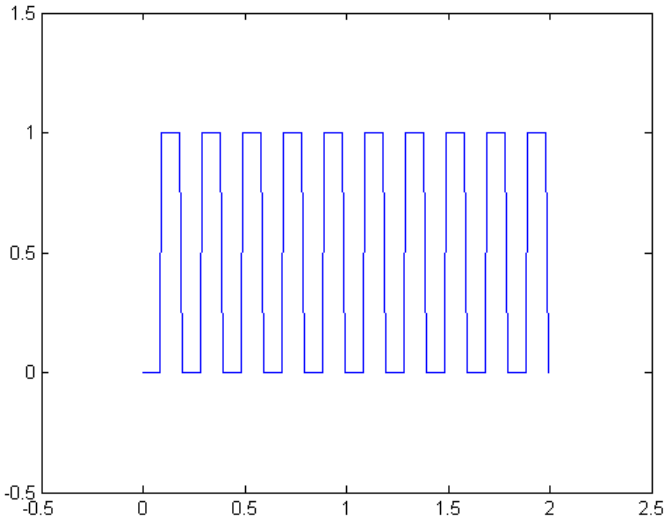

Рис. 5.3: Прямоугольный сигнал

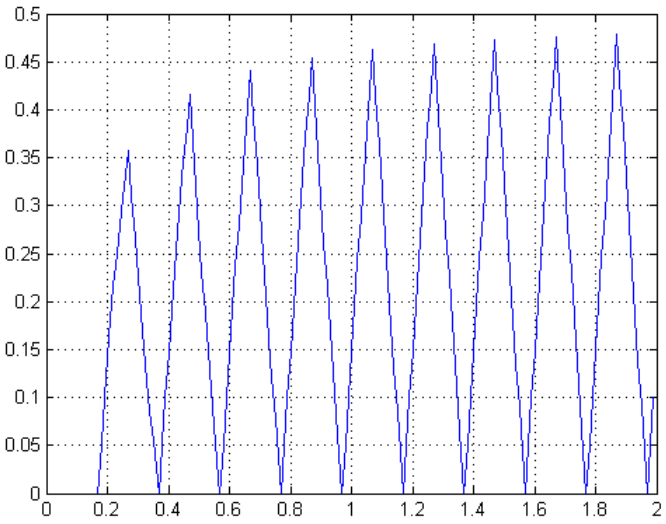

Рис. 5.4: Спектр прямоугольного сигнала

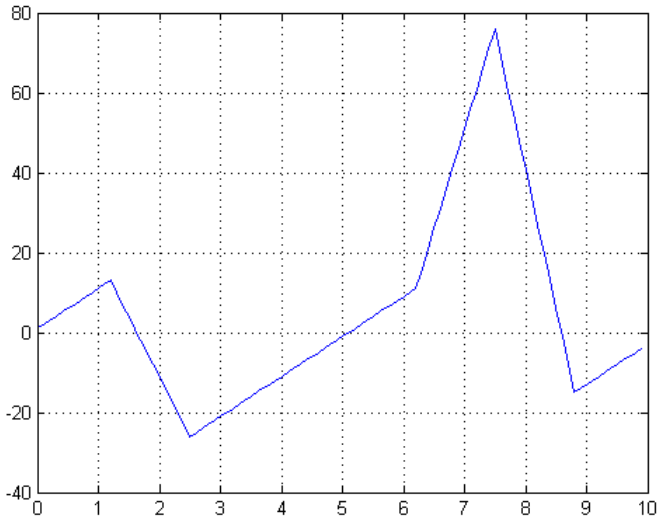

Рис. 5.5: Треугольный сигнал

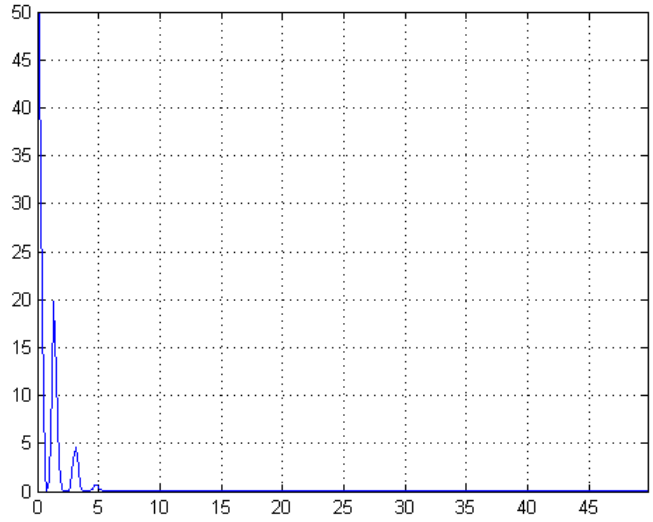

Рис. 5.6: Спектр треугольного сигнала

5.4.2 Simulink

В ходе моделирования сигналов в Simulink были собраны следующие схемы и получены соответствующие графики:

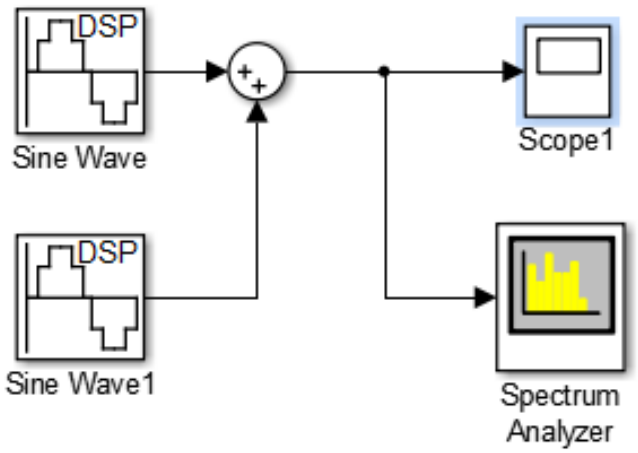

Рис. 5.7: Полигармонический сигнал (Simulink)

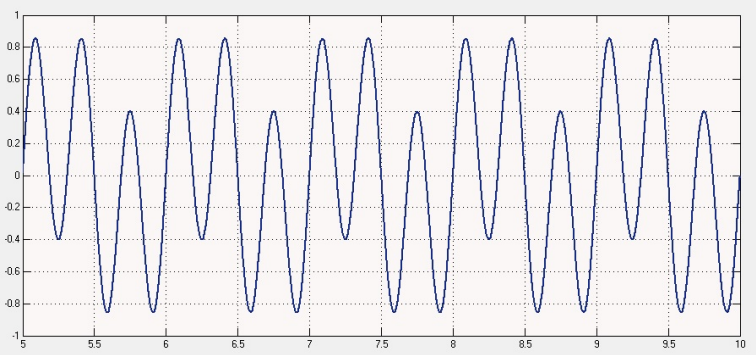

Рис. 5.8: Полигармонический сигнал

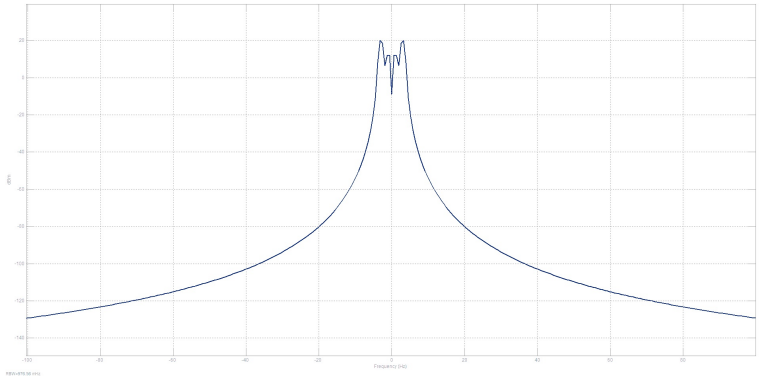

Рис. 5.9: Спектр полигармонического сигнала

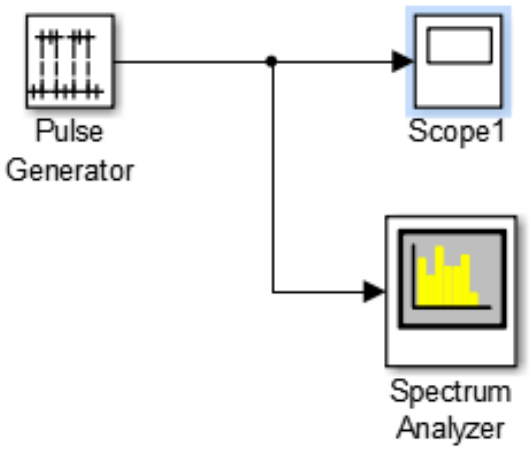

Рис. 5.10: Прямоугольный импульсный сигнал (Simulink)

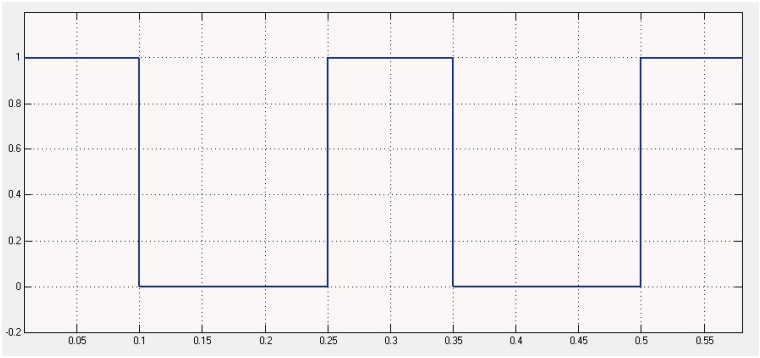

Рис. 5.11: Прямоугольный импульсный сигнал

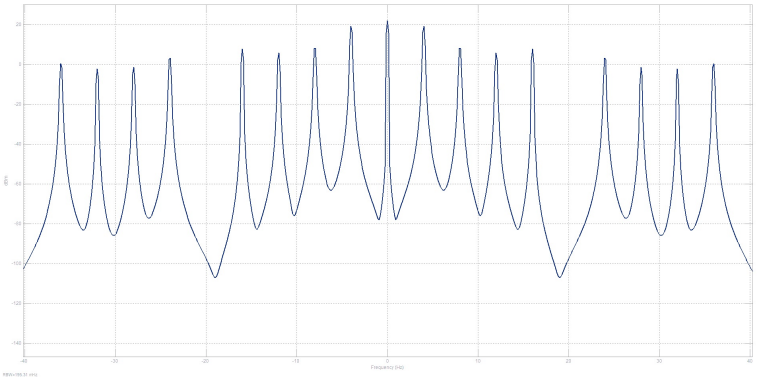

Рис. 5.12: Спектр прямоугольного сигнала

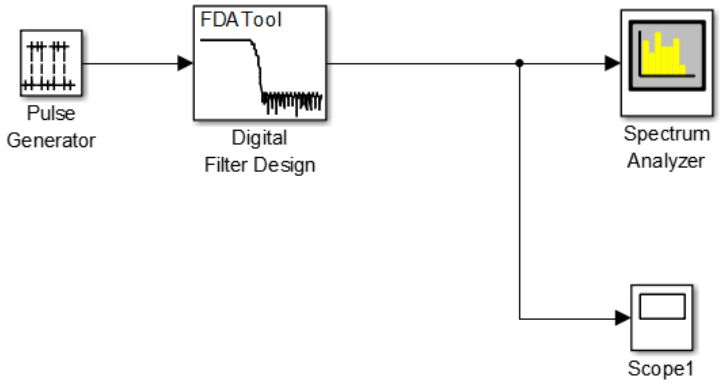

Рис. 5.13: Треугольный импульсный сигнал (Simulink)

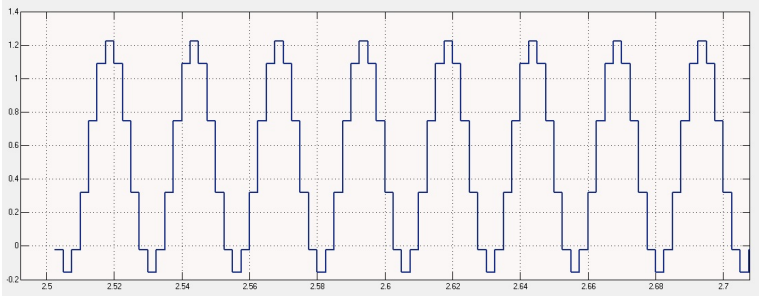

Рис. 5.14: Треугольный импульсный сигнал

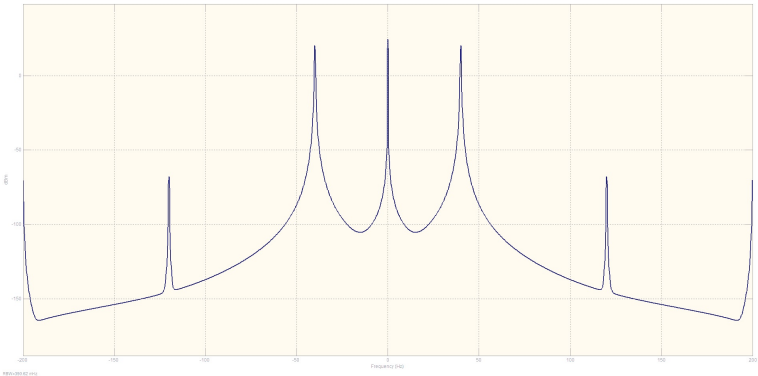

Рис. 5.15: Спектр треугольного сигнала

5.5 Выводы

В результате проделанной работы было проведено моделирование в среде MatLAB и Simulink полигармонического, прямоугольного и треугольного импульсных сигналаов, а также получены их спектры. В Simulink для получения генератора треугольного сигнала используется генератор прямоугольных импульсов каскадно с фильтром с прямоугольным окном. Это обоснуется тем, что свертка двух прямоугольных импульсов в результате дает треугольный.

Глава 6

Линейная фильтрация

6.1 Цель работы

Изучить воздействие ФНЧ на тестовый сигнал с шумом.

6.2 Постановка задачи

Сгенерировать гармонический сигнал с шумом и синтезировать ФНЧ. Получить сигнал во временной и частотной областях до и после фильтрации. Сделать выводы о воздействии ФНЧ на спектр сигнала.

6.3 Ход работы

Линейный фильтр - динамическая система, применяющая некий линейный оператор ко входному сигналу для выделения или подавления определенных частот сигнала и других функций по обработке входного сигнала. Фильтр Баттерворта - один из типов электронных фильтров. Фильтры этого класса отличаются от других методом проектирования. Фильтр Баттерворта проектируется так, чтобы его амплитудно-частотная характеристика была максимально гладкой на частотах полосы пропускания. В командном окне Matlab сгенерируем гармонический сигнал без шума и с шумом, а также спектры сигнала с шумом и без шума.

6.3.1 Код Matlab

```
x = 0:0.01:4*pi;
```

```
f=100*(0:255)/512;
figure
noise=rand(size(x));
y = sin(2*pi*x);
y_noisy = y+0.3*noise;
%Построение сигнала без шума:
plot(x(1:200),y(1:200))
grid
%синтез ФНЧ Баттерворта
[B,A] = butter(16,0.99);
B=B./sum(B);
A=A./sum(A);
%обработка сигнала ФНЧ
figure
y_filtered = conv(y_noisy,[B,A]);
%Построение сигнала с шумом:
plot(x(1:200),y_filtered(1:200))
grid
figure
%Построение спектра сигнала с шумом
noisy_spectrum = fft(y_noisy,512);
norm_noisy_spectrum = noisy_spectrum.*conj(noisy_spectrum)/512;
%Построение нормирова спектра сигнала с шумом:
plot(f,norm_noisy_spectrum(1:256))
axis([0 max(f) 0 2])
grid
figure
%Спектр сигнала без шума
spectrum = fft(y_filtered,512);
norm_filtered_spectrum=spectrum.*conj(spectrum)/512;
%Построение нормированого спектра сигнала без шума:
plot(f,norm_filtered_spectrum(1:256))
axis([0 max(f) 0 2])
grid
```

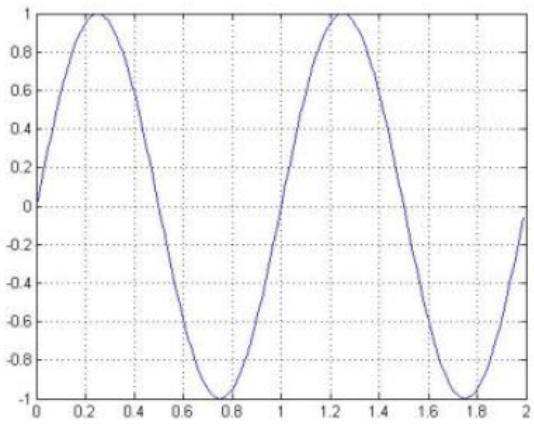

Рис. 6.1: Сигнал без шума

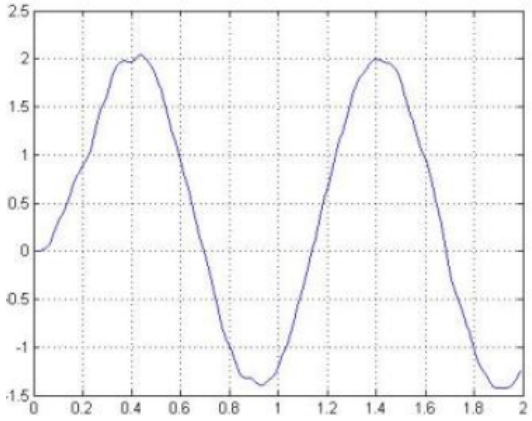

Рис. 6.2: Сигнал с шумом

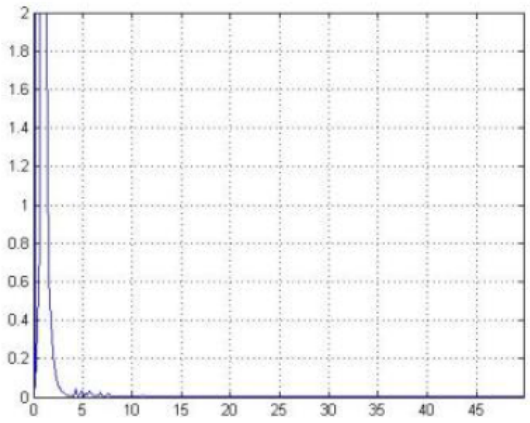

Рис. 6.3: Спектр сигнала без шума

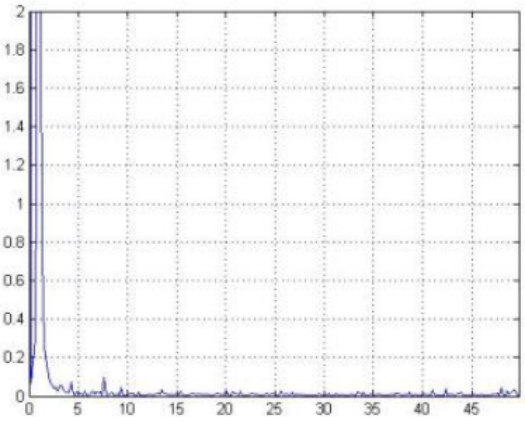

Рис. 6.4: Спектр сигнала с шумом

6.3.2 Simulink

Для создания модели в Simulink используем блок Discrete FIR Filter раздела Discrete главной библиотеки и блок Digital Filter design из Signal Processing Blockset/Filtering/Filter Designs.

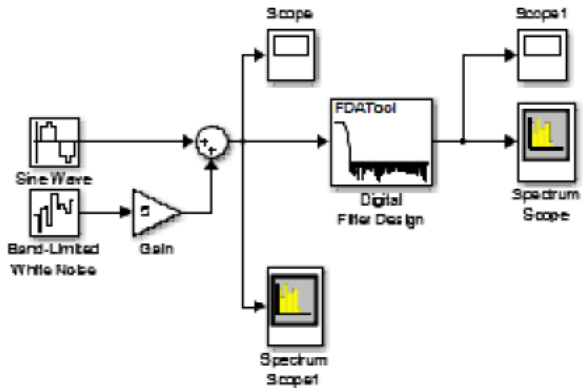

Рис. 6.5: Схема модели

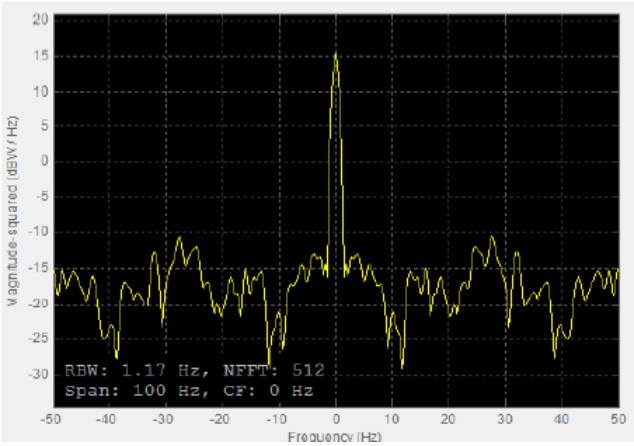

Рис. 6.6: Спектр до фильтрации

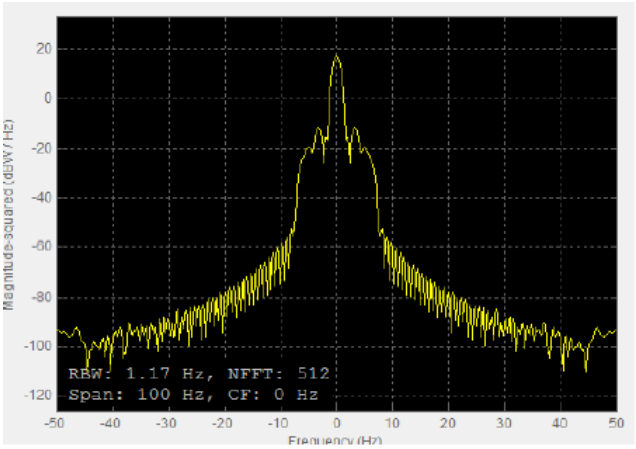

Рис. 6.7: Спектр после фильтрации

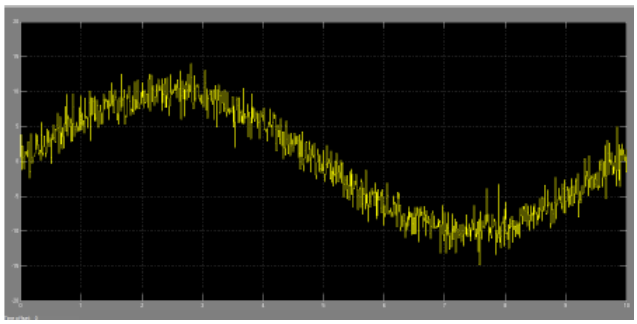

Рис. 6.8: Сигнал без шума

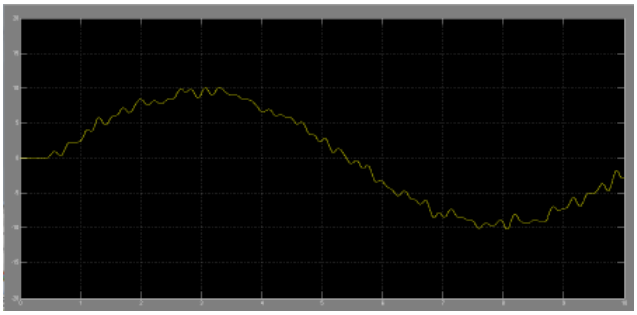

Рис. 6.9: Сигнал с шумом

6.4 Вывод

Фильтр нижних частот - вид фильтра, пропускающий частотный спектр сигнала ниже некоторой частоты (частоты среза), и подавляющий частоты сигнала выше этой частоты. Степень подавления каждой частоты зависит от вида и порядка фильтра. Идеальный ФНЧ полностью подавляет все частоты входного сигнала выше частоты среза и пропускает без изменений все частоты ниже частоты среза.

В лабораторной работе мы произвели удаление шума линейным фильтром. Однако важно понимать, что линейные фильтры способны удалять только шум с частотой, отличающейся от частоты сигнала.

Глава 7

Аналоговая модуляция

7.1 Цель работы

Изучение амплитудной модуляции/демодуляции сигнала.

7.2 Ход работы

7.2.1 Теоретическая часть

* $S(t)$ — информационный сигнал, $|S(t)| < 1$,
* $U_c(t)$ — несущее колебание.

Тогда амплитудно-модулированный сигнал $U_{\text{am}}(t)$ может быть записан следующим образом:

$$U_{\text{am}}(t) = U_c(t)[1 + mS(t)](1)$$

Здесь m — некоторая константа, называемая коэффициентом модуляции. Формула описывает несущий сигнал $U_c(t)$, модулированный по амплитуде сигналом $S(t)$ с коэффициентом модуляции m.

7.2.2 Код Matlab

```
close all;
clear all;

x = 0:0.1:8*3.14;
f0 = 0.1;
```

```
% Исходный сигнал
y = sin(2*pi*f0*x);
figure
plot(x(1:250),y(1:250))
grid

% Спектр исходного сигнала
spectrum = fft(y, 512);
norm_spectrum = spectrum.*conj(spectrum)/512;
f = 100*(0:255)/512;
figure
plot(f, norm_spectrum(1:256))
axis([0 max(f) 0 70])
grid

% Амплитудная модуляция
Fc = 10*f0;
Fs = 100*f0;
U = ammod(y, Fc, Fs, 0, 1);
figure
plot(x(1:250), U(1:250))
grid

 % Спектр модулированного сигнала
 u_spectrum = fft(U, 512);
 norm_u_spectrum = u_spectrum.*conj(u_spectrum)/512;
 figure
 plot(f, norm_u_spectrum(1:256))
 axis([0 max(f) 0 70])
 grid

% Амплитудная модуляция с подавлением несущей
Fc = 10*f0;
Fs = 100*f0;
U = ammod(y, Fc, Fs);
figure
plot(x(1:250), U(1:250))
grid

% Спектр модулированного сигнала
u_spectrum = fft(U, 512);
```

```
norm_u_spectrum = u_spectrum.*conj(u_spectrum)/512;
figure
plot(f, norm_u_spectrum(1:256))
axis([0 max(f) 0 70])
grid

% Однополосная модуляция
Fc = 10*f0;
Fs = 100*f0;
U = ssbmod(y, Fc, Fs, [], 'upper');
figure
plot(x(1:250), U(1:250))
grid

% Спектр модулированного сигнала
u_spectrum = fft(U, 512);
norm_u_spectrum = u_spectrum.*conj(u_spectrum)/512;
figure
plot(f, norm_u_spectrum(1:256))
axis([0 max(f) 0 70])
grid

%Синхронное детектирование
[b, a] = butter(10, Fc*2/Fs);
z = ssbdemod(U, Fc, Fs, 0, b, a);
figure
plot(x(1:250), z(1:250))
grid

% Спектр демодулированного сигнала
du_spectrum = fft(U, 512);
norm_du_spectrum = du_spectrum.*conj(du_spectrum)/512;
figure
plot(f, norm_du_spectrum(1:256))
axis([0 max(f) 0 70])
grid

% Рассчет КПД модуляции
M = 0.5;
n = M^2/(M^2+2)
```

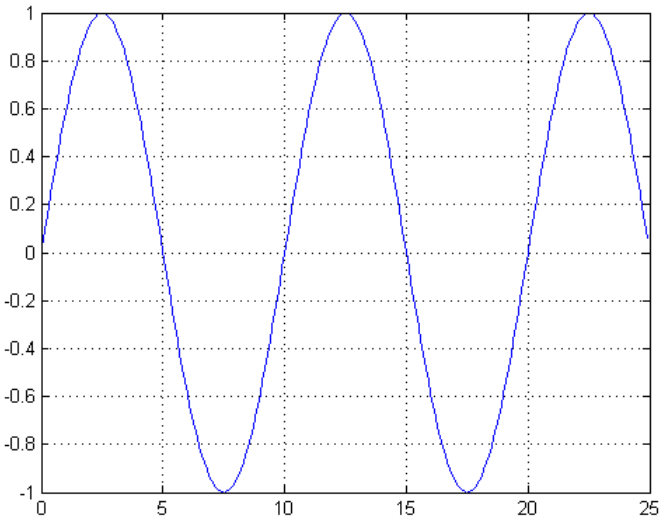

Рис. 7.1: Исходный сигнал

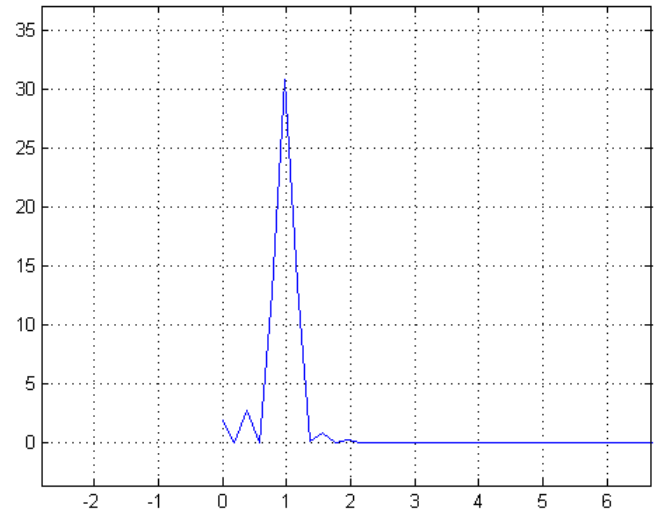

Рис. 7.2: Спектр исходного сигнала

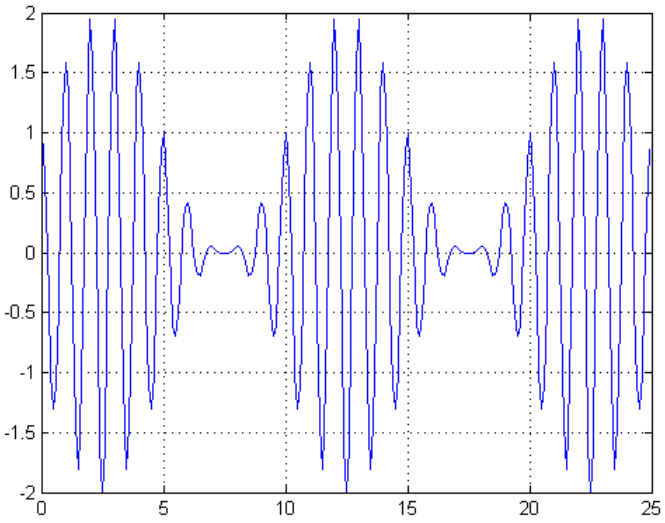

Рис. 7.3: Амплитудная модуляция

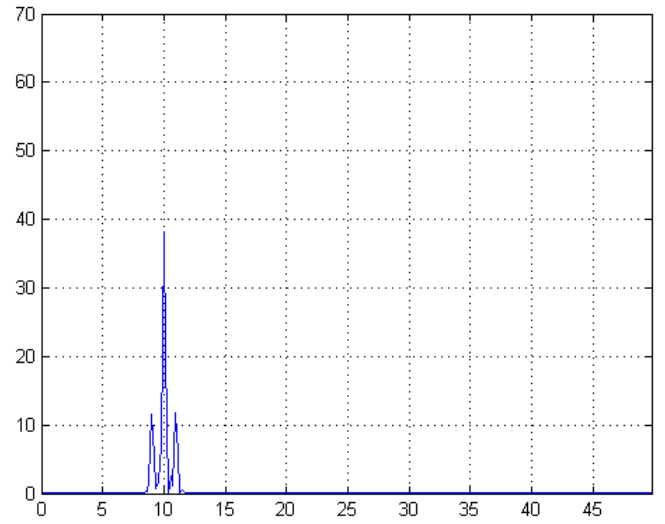

Рис. 7.4: Спектр модулированного сигнала

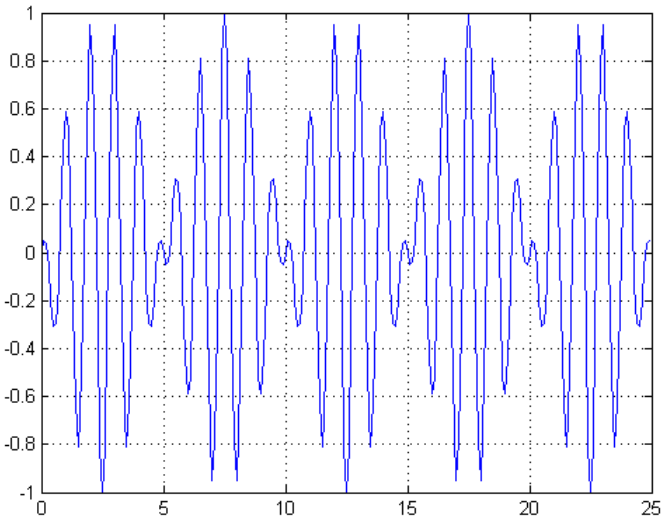

Рис. 7.5: Амплитудная модуляция с подавлением несущей

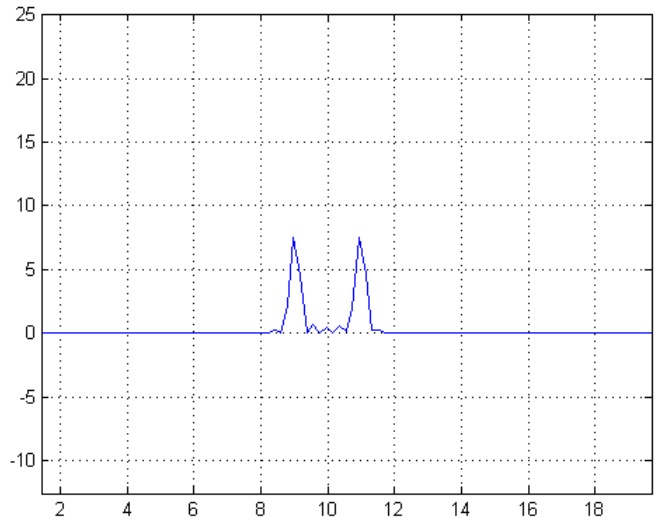

Рис. 7.6: Спектр модулированного сигнала

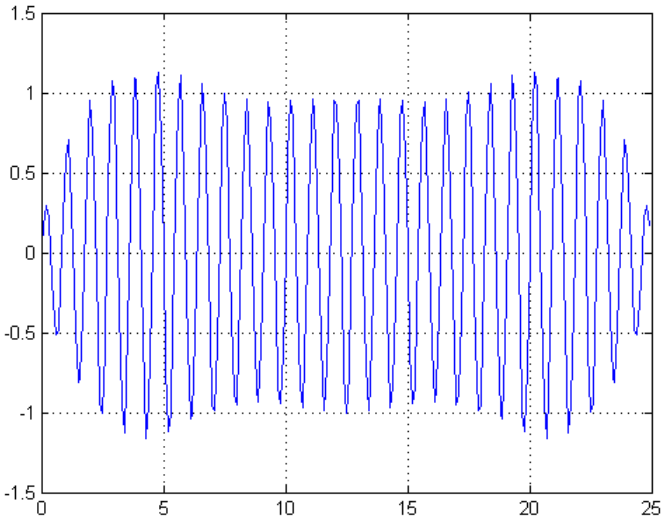

Рис. 7.7: Однополосная модуляция

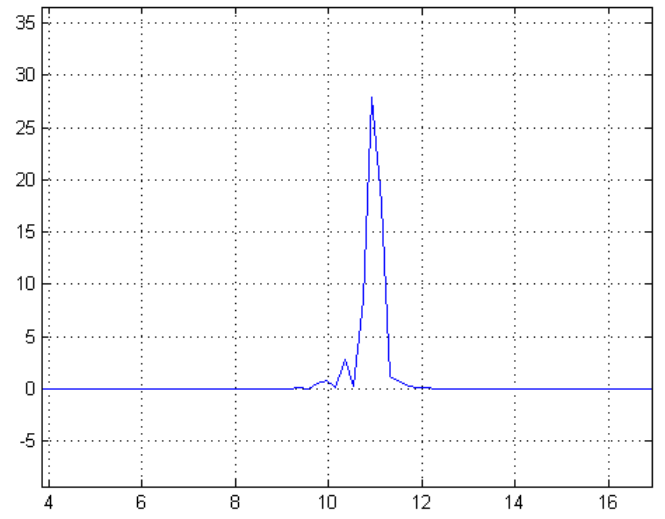

Рис. 7.8: Спектр модулированного сигнала

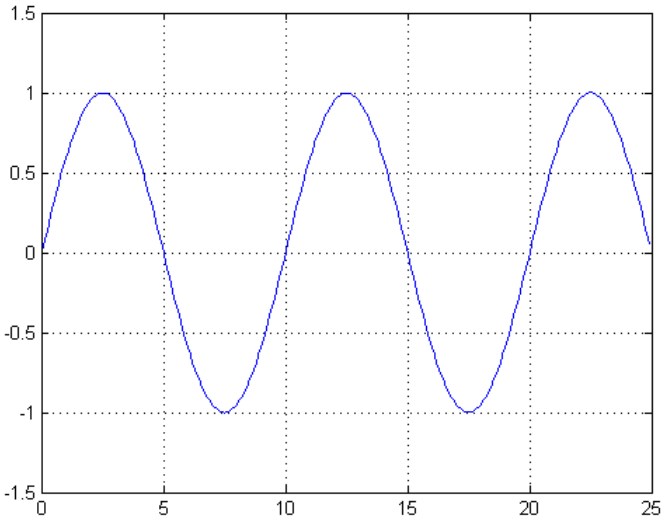

Рис. 7.9: Синхронное детектирование

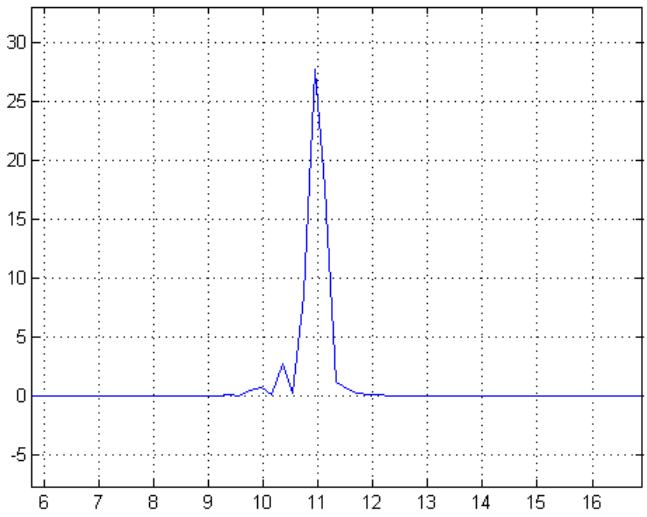

Рис. 7.10: Спектр модулированного сигнала

7.2.3 Simulink

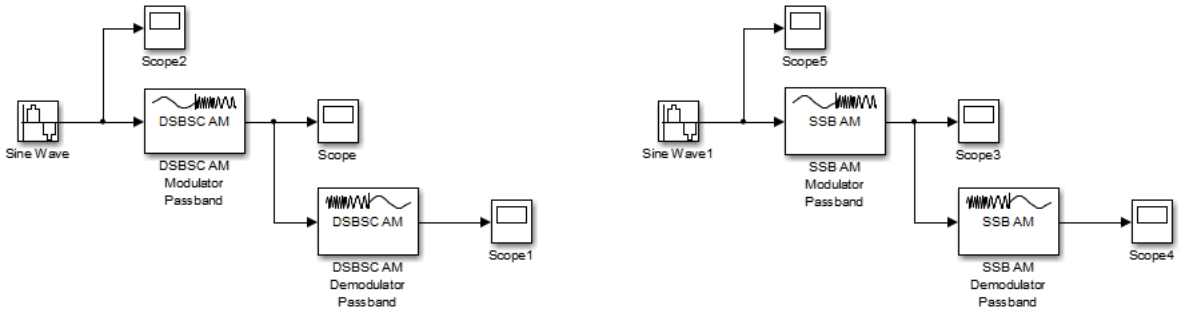

Рис. 7.11: Схема модели

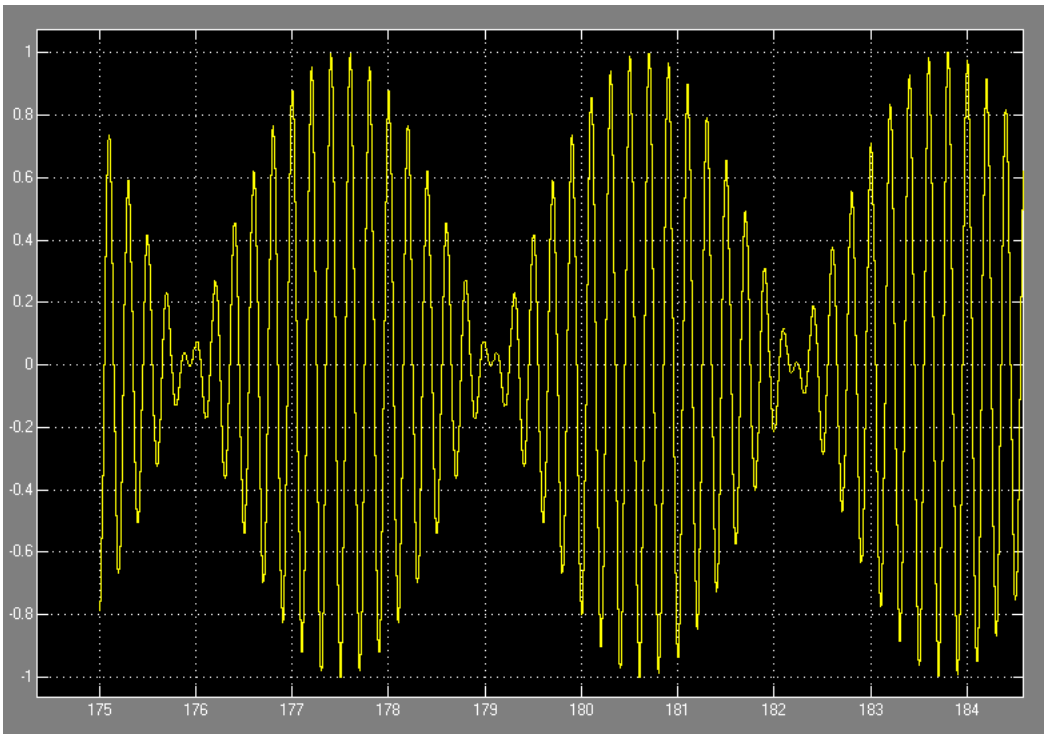

Рис. 7.12: Модулированный сигнал

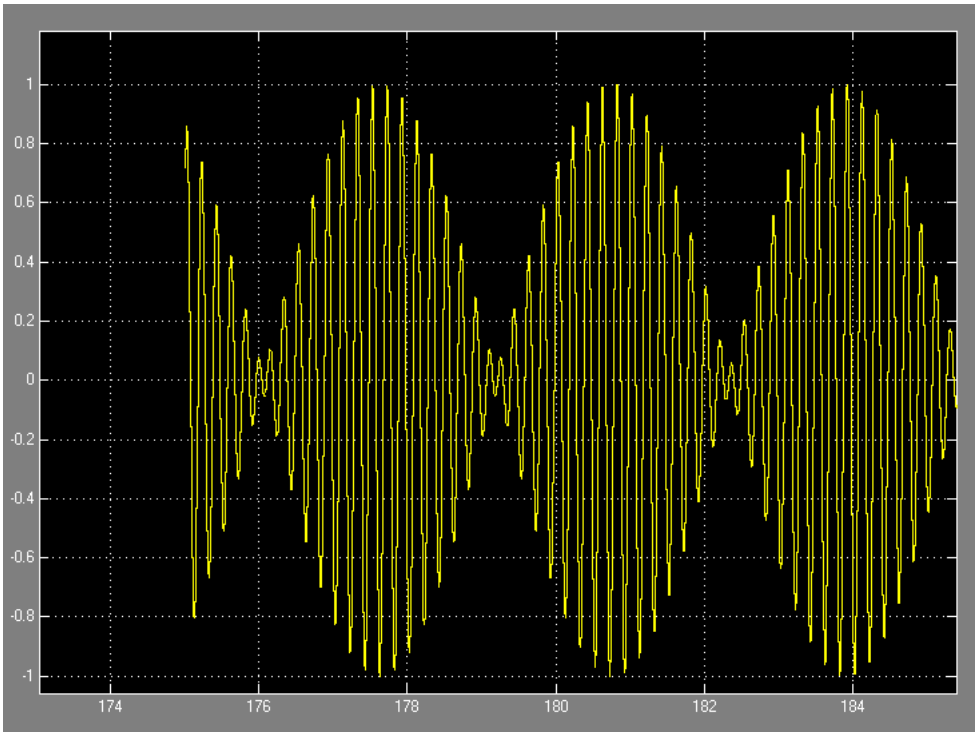

Рис. 7.13: Демодулированный сигнал

7.3 Вывод

Основная проблема амплитудной модуляции - она менее защищена от помех по сравнению с частотной. В настоящее время амплитудная модуляция применяется в основном только для радиовещания на сравнительно низких частотах (не выше коротких волн) и для передачи изображения в телевизионном вещании. Это обусловлено низким КПД использования энергии модулированных сигналов.

АМ - сигнал представляет собой произведение информационной огибающей U(t) и гармонического колебания ее заполнения с более высокими частотами. Значение М должно находиться в пределах от 0 до 1 для всех гармоник модулирующего сигнала. Малую глубину модуляции для основных гармоник модулирующего сигнала(М «1) применять нецелесообразно, т.к. при этом мощность передаваемого информационного сигнала будет много меньше мощности несущего колебания, и мощность передатчика используется неэкономично. Для однотональной модуляции начальная фаза модулирующего колебания для верхней боковой частоты складывается с начальной фазой несущей, для нижней - вычитаются из фазы несущей. Физическая ширина спектра модулированного сигнала в

два раза больше ширины спектра сигнала модуляции, так как модулирующее колебание перемещается в область частоты ω_0 и расщепляется на два колебания, симметричные относительно частоты ω_0, что будет видно в спектре.

$$u(t) = U_m cos(\omega_0 t) + (U_m * M/2) cos[(\omega_0 + \Omega)t] + (U_m * M/2) cos[(\omega_0 - \Omega)t]$$

При балансной модуляции производится перемножение двух сигналов - модулирующего и несущего, при котором происходит подавление несущего колебания, соответственно, КПД модуляции становится равным 100%.

Глава 8

Частотная и фазовая модуляция

8.1 Постановка задачи

1. Сгенерировать однотональный сигнал низкой частоты.
2. Выполнитьфазовую модуляцию/демодуляцию сигнала по закону $u(t) = (U_m \cos(\omega t = ks(t))$, используя встроенную функцию MatLab pmmod, pmdemod.
3. Получить спектр модулированного сигнала.
4. Выполнить частотную модуляцию/демодуляцию по закону

$$u(t) = U_m \cos(\omega_0 t + k \int_0^t s(t)dt = \phi_0) \tag{8.1}$$

используя встроенные функции Matlab fmmod, fmdemod.

8.2 Теоретическая часть

Частотная модуляция — вид аналоговой модуляции, при котором информационный сигнал управляет частотой несущего колебания. По сравнению с амплитудной модуляцией здесь амплитуда остаётся постоянной.

Фазовая модуляция — один из видов модуляции колебаний, при которой фаза несущего колебания управляется информационным сигналом. Фазомодулированный сигнал s(t) имеет следующий вид:

$$s(t) = g(t) \sin[2\pi f_c t + \varphi(t)], \tag{8.2}$$

где $g(t)$ — огибающая сигнала; $\phi(t)$ является модулирующим сигналом; f_c — частота несущего сигнала; t — время.

По характеристикам фазовая модуляция близка к частотной модуляции. В случае синусоидального модулирующего (информационного) сигнала, результаты частотной и фазовой модуляции совпадают.

8.3 Ход работы

8.3.1 Код Matlab

```
x = 0:0.01:2;
u = cos(2*pi*x);

figure;
subplot(3,1,1);
plot(x,u);
grid;
am = fmmod(u,2,30,1);
subplot(3,1,2);
plot(x,am);
grid;
modulatedSpectr = fft(am,512);
normSpectrum = modulatedSpectr.*conj(modulatedSpectr)/512;
f = 100*(-256:255)/512;
subplot(3,1,3);
plot(f,normSpectrum);
grid;
axis([min(f) max(f) 0 max(normSpectrum)]);

figure;
subplot(3,1,1);
plot(x,u);
grid;
am = pmmod(u,5,1000,pi/2);
subplot(3,1,2);
plot(x,am);
grid;
modulatedSpectr = fft(am,512);
normSpectrum = modulatedSpectr.*conj(modulatedSpectr)/512;
f = 100*(-256:255)/512;
subplot(3,1,3);
plot(f,normSpectrum);
```

```
grid;
axis([min(f) max(f) 0 max(normSpectrum)]);
```

8.4 Результаты работы

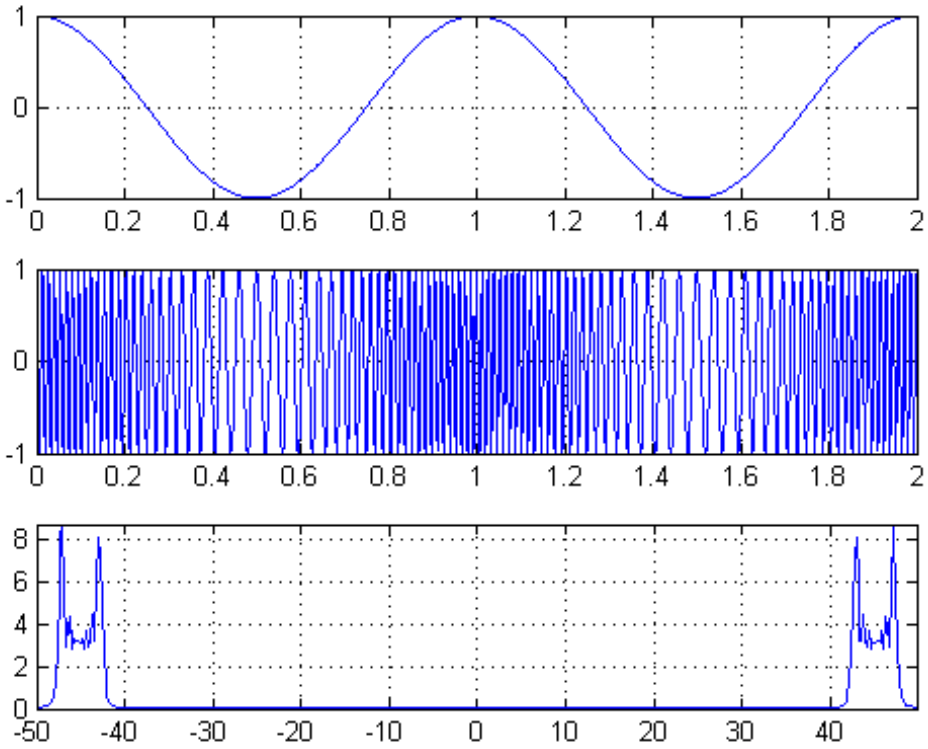

Рис. 8.1: Частотная модуляция сигнала

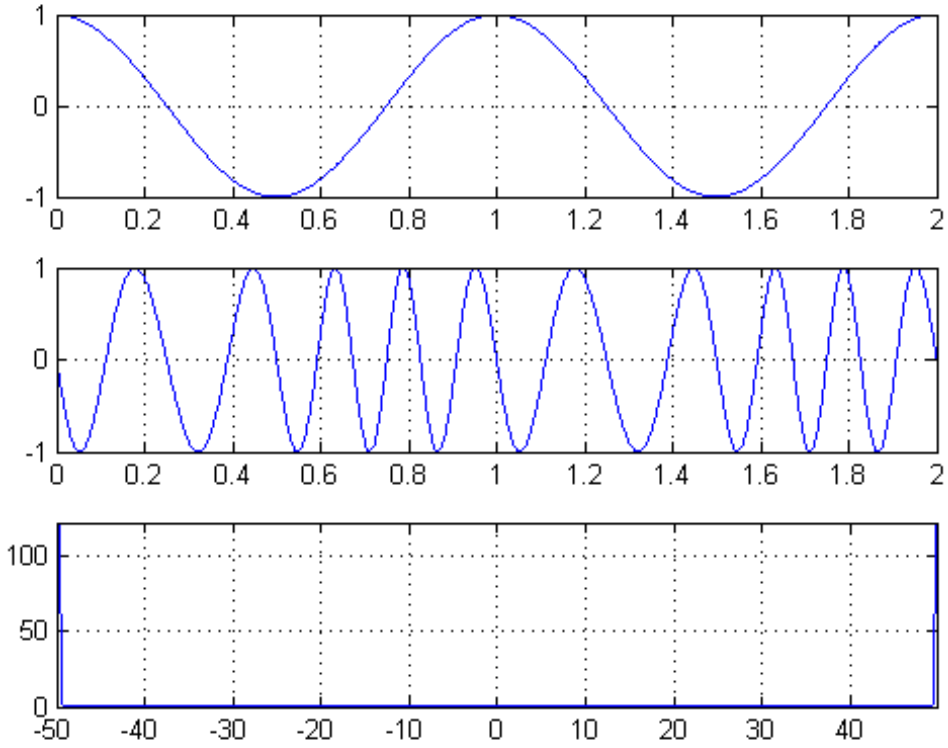

Рис. 8.2: Фазовая модуляция сигнала

Смоделируем ход работы в среде Simulink:

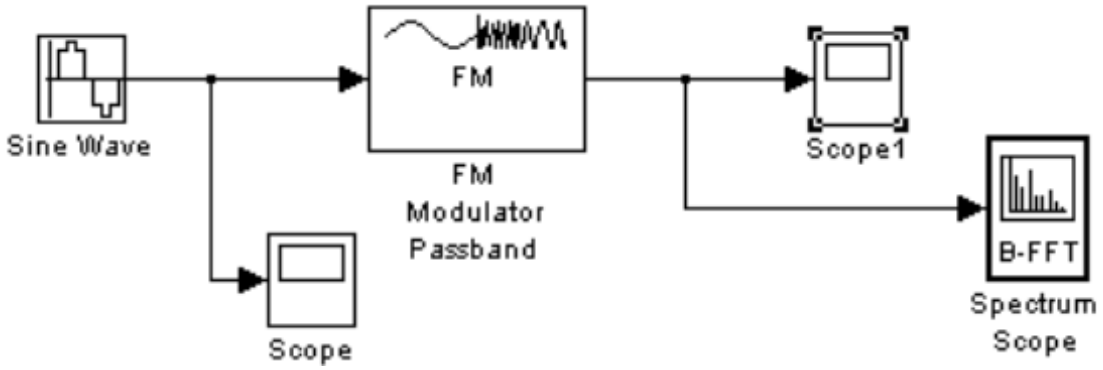

Рис. 8.3: Модель для частотной модуляции

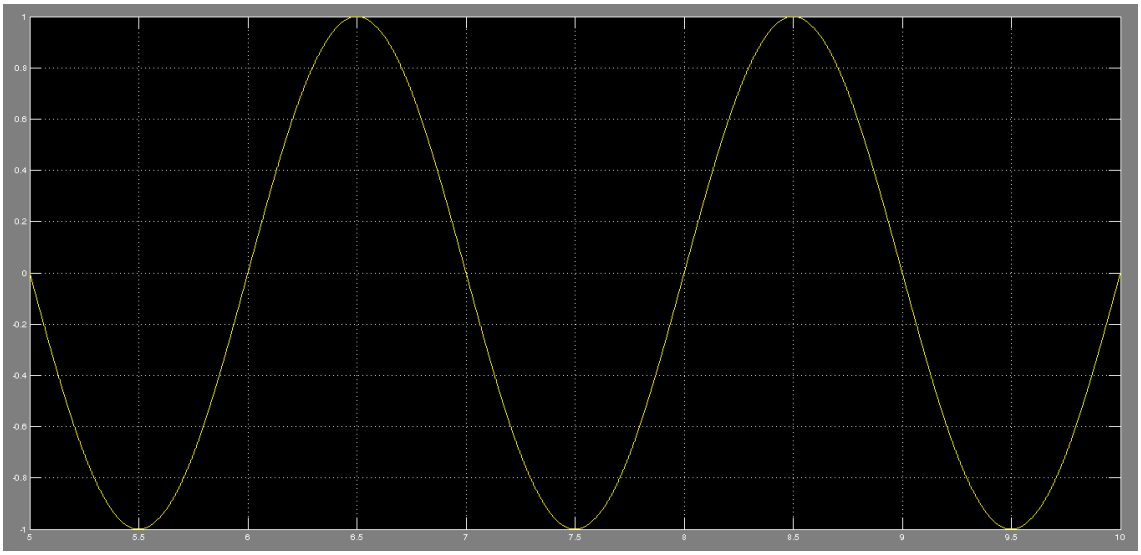

Рис. 8.4: Исходный сигнал

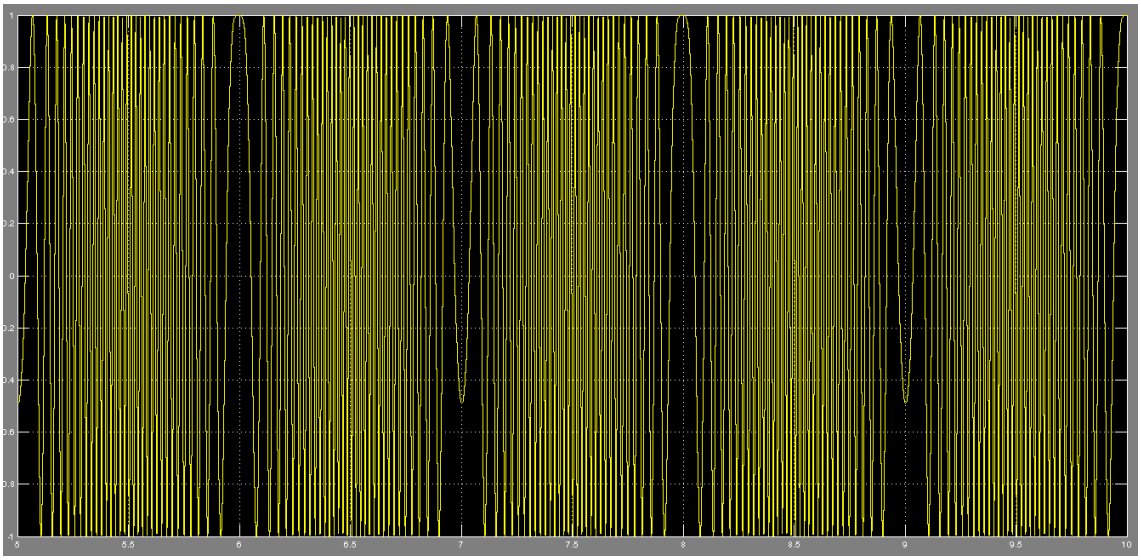

Рис. 8.5: Моделированный сигнал

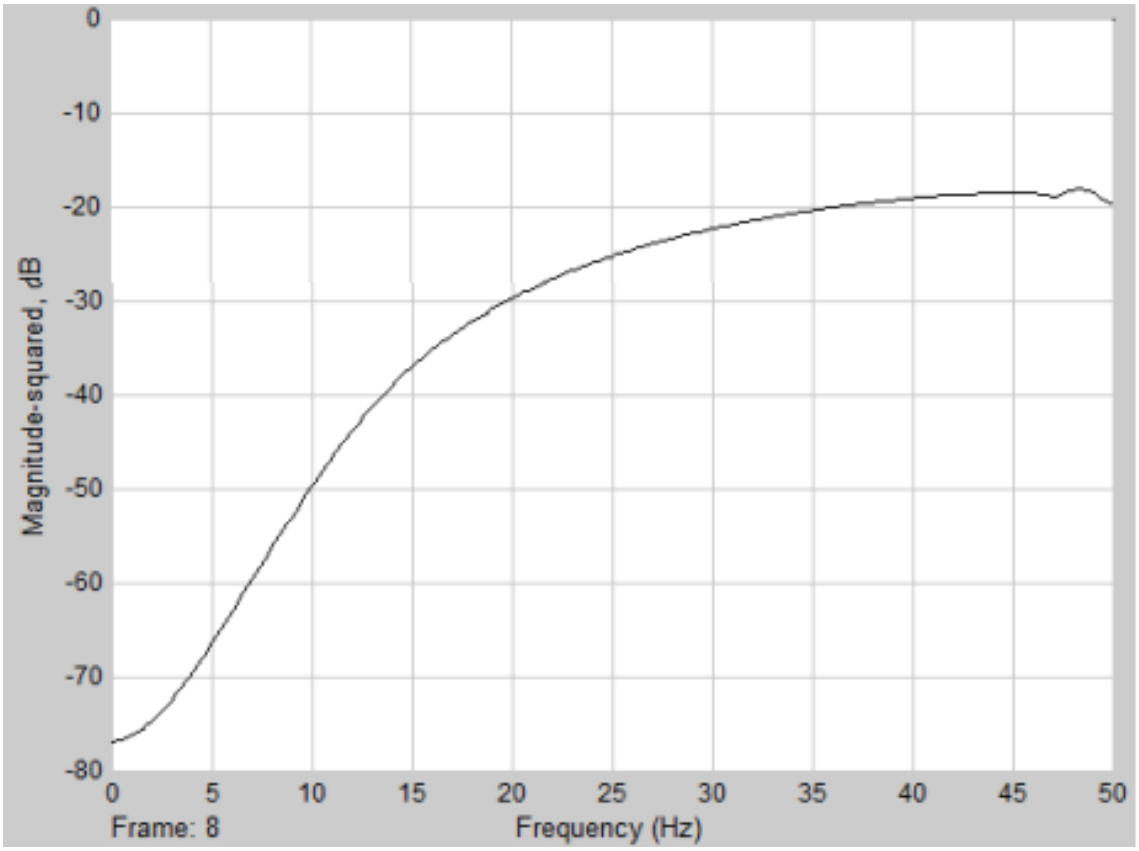

Рис. 8.6: Спектр моделированного сигнала

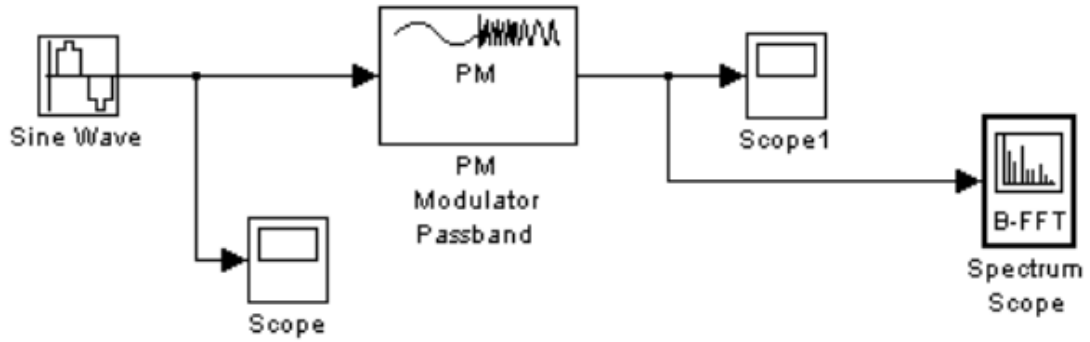

Рис. 8.7: Модель для фазовой модуляции

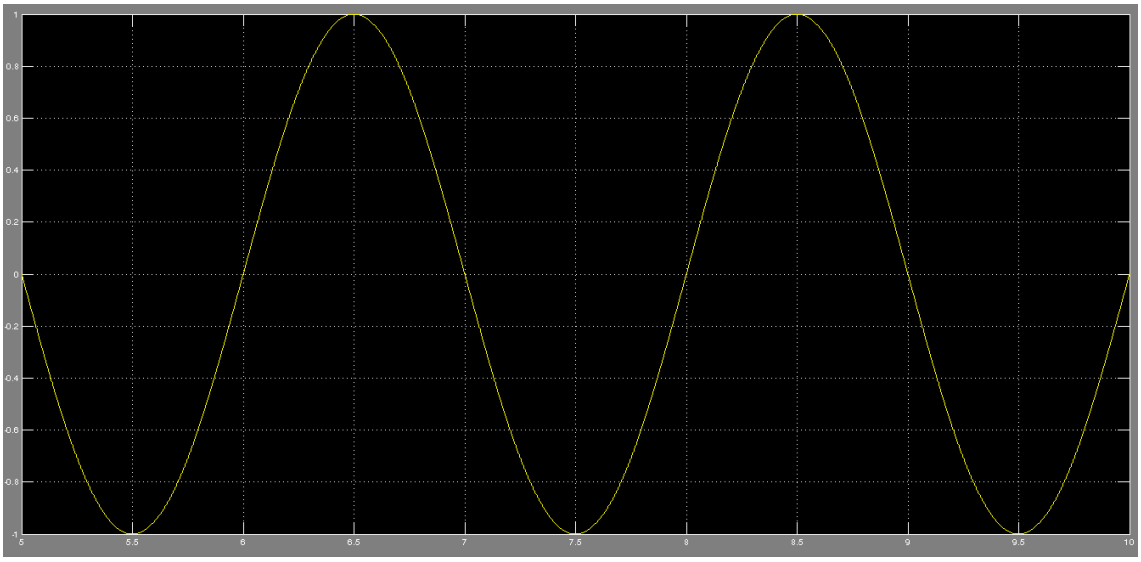

Рис. 8.8: Исходный сигнал

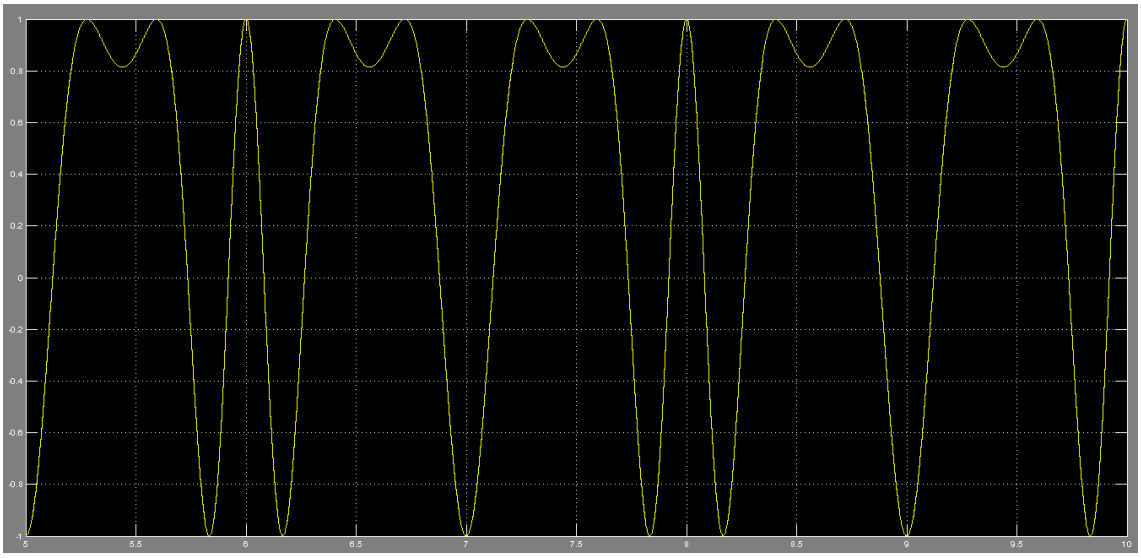

Рис. 8.9: Моделированный сигнал

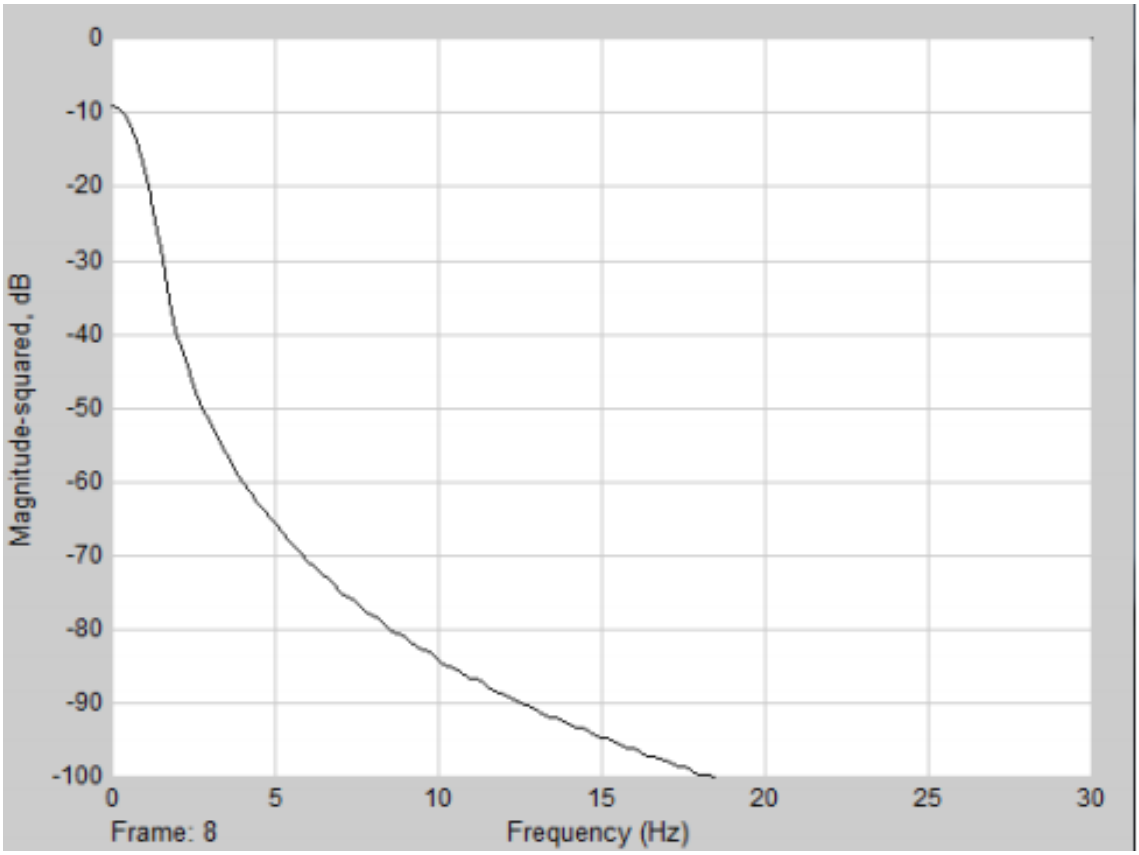

Рис. 8.10: Спектр моделированного сигнала

Выполним демодуляцию, в том числе с помощью блока захвата фазы (фазовой автоподстройки частоты) Phase-Locked Loop.

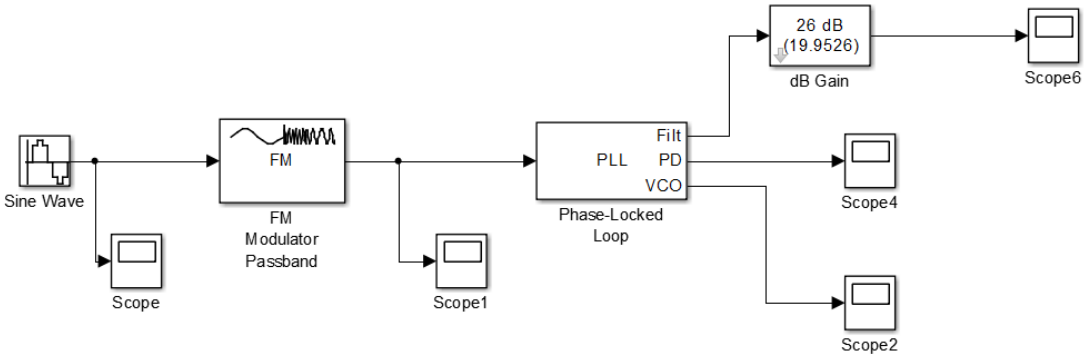

Рис. 8.11: Phase-Locked Loop

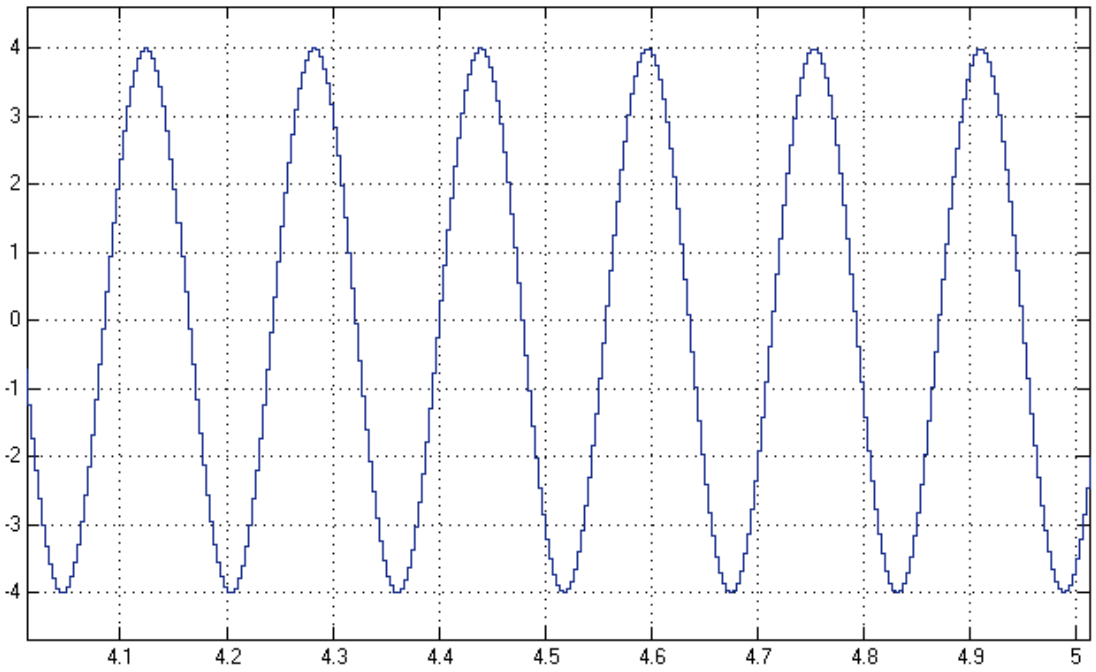

Рис. 8.12: Исходный сигнал

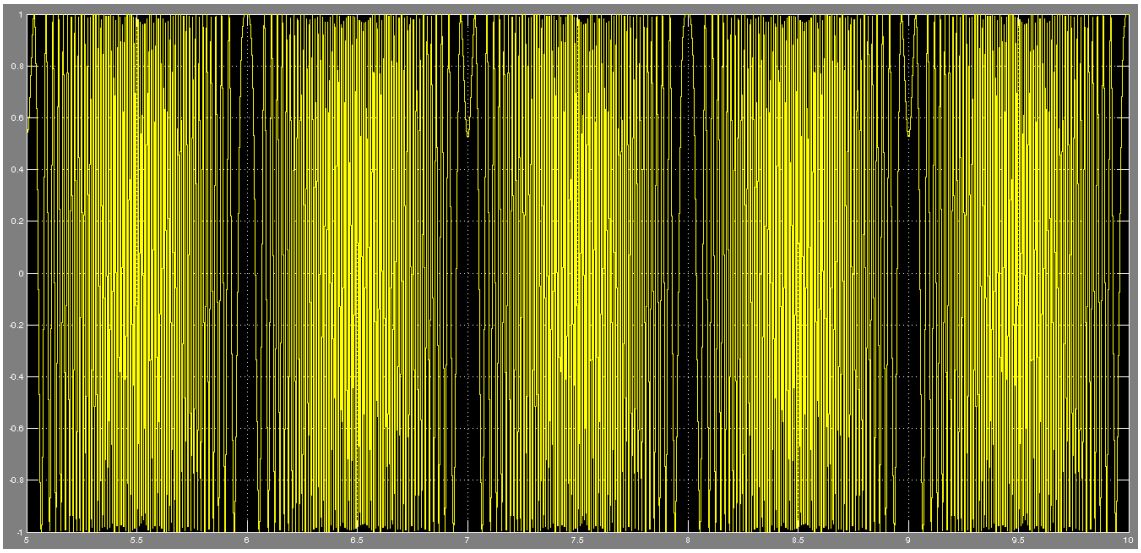

Рис. 8.13: Моделированный сигнал

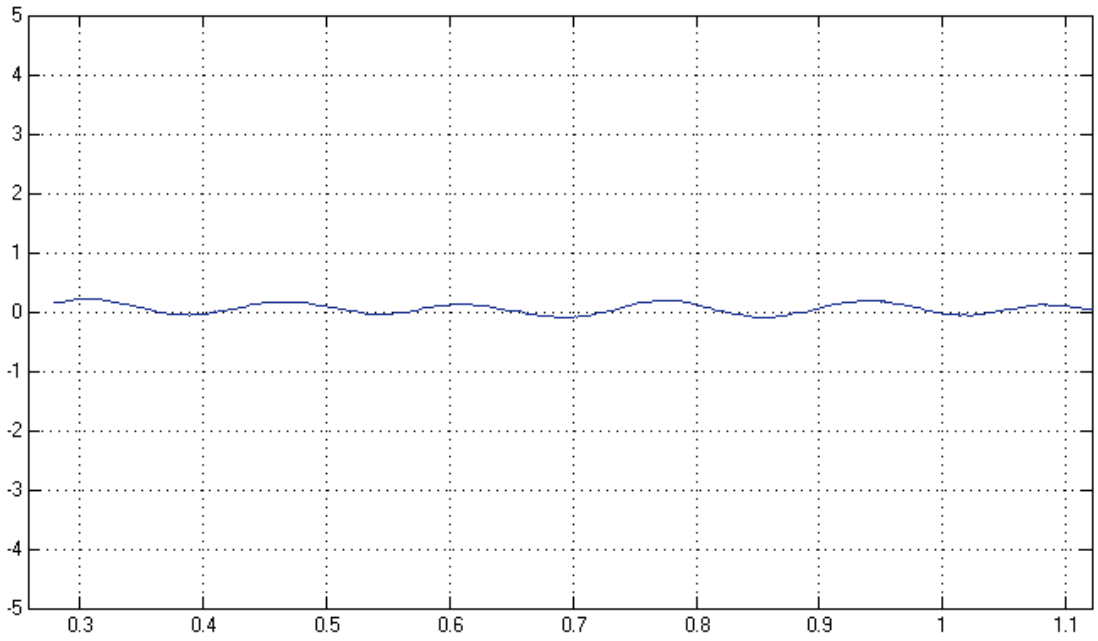

Рис. 8.14: Сигнал на выходе ФНЧ

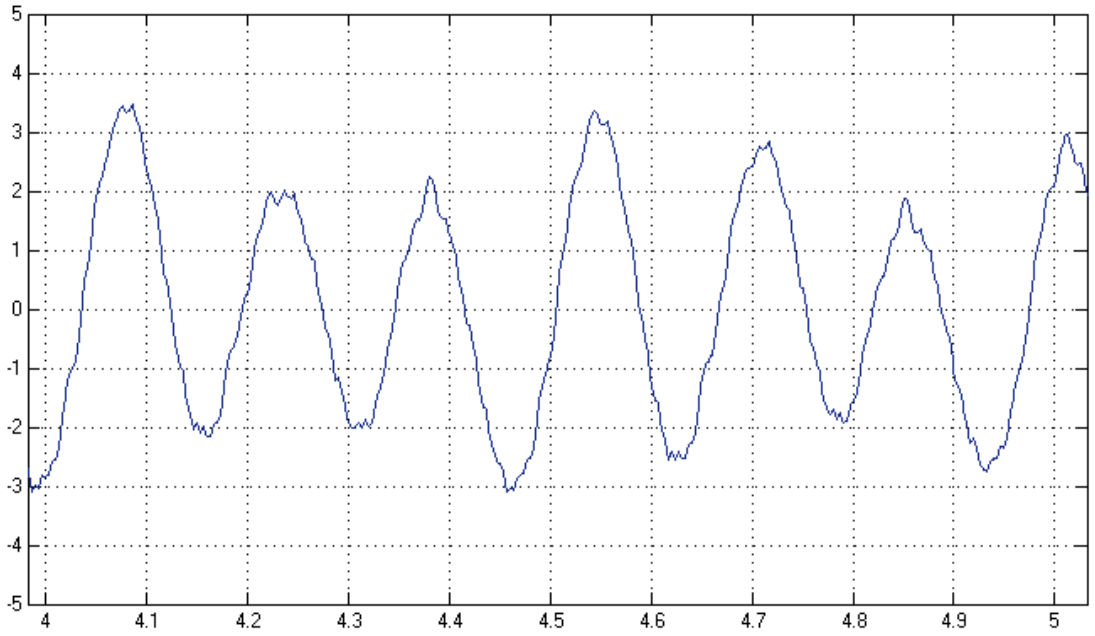

Рис. 8.15: Сигнал на выходе ФНЧ с усилением на 26 dB (в 20 раз)

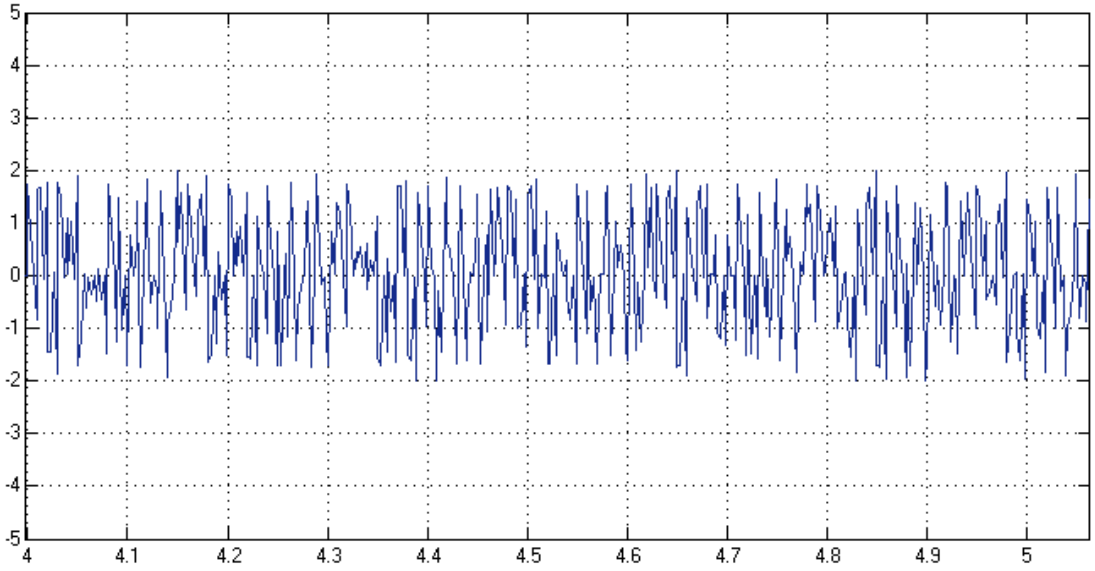

Рис. 8.16: Сигнал на выходе фазового детектора

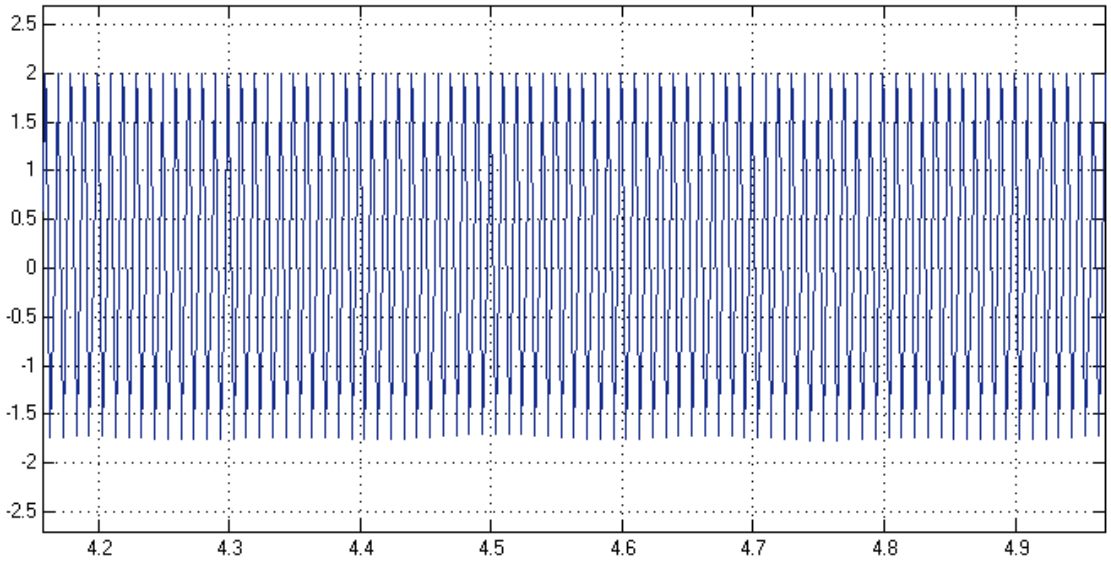

Рис. 8.17: Сигнал на выходе ГУН

8.5 Вывод

В результате выполнения данной работы были выполнены частотная и фазовая модуляция/демодуляция, а также чатотная демодуляция с помощью блока захвата фазы. Частотная и фазовая модуляция тесно взаимосвязаны, поскольку обе они влияют на аргумент функции cos. Поэтому эти два вида модуляции имеют общее название — угловая модуляция. Сигнал с угловой модуляцией имеет вид колебания, начальная фаза которого зависит от времени:

$$s_(t) = A_0 cos(\omega 0t + j(t)). \tag{8.3}$$

В случае фазовой модуляции частота несущей зависит от производной модулируемого сигнала, а в случае частотной - от самого его значения.

Для демодуляции использовалась петля ФАПЧ, состоящая из перемножителя (используемого в качестве фазового детектора), фильтра нижних частот и генератора, управляемого напряжением (ГУН).

Глава 9

Цифровая модуляция

9.1 Постановка задачи

1. Получить сигналы BPSK, PSK, OQPSK, genQAM, MSK, M-FSK модуляторов.
2. Построить их сигналы
3. Провестисравнение изученных методов модуляции цифровых сигналов.

9.2 Теоретическая часть

Сущность цифровой модуляции заключается в том, что передаваемый непрерывный сигнал дискретизируется во времени, квантуется по уровню и полученные отсчеты, следующие в дискретные моменты времени, преобразуются в кодовые комбинации. Полученной последовательностью кодовых видеосигналов модулируется высокочастотный сигнал-переносчик. Существует 3 основных вида манипуляции сигналов: амплитудная (Amplitude-shift keying (ASK)), частотная (Frequency-shift keying (FSK)) и фазовая (Phase-shift keying (PSK)).

1. Амплитудная манипуляция (ASK) — изменение сигнала, при котором скачкообразно меняется амплитуда несущего колебания.
2. При частотной манипуляции (FSK) значениям «0» и «1» информационной последовательности соответствуют определённые частоты синусоидального сигнала при неизменной амплитуде. Частотная

манипуляция весьма помехоустойчива. Однако при частотной манипуляции неэкономно расходуется ресурс полосы частот телефонного канала. Поэтому этот вид модуляции применяется в низкоскоростных протоколах, позволяющих осуществлять связь по каналам с низким отношением сигнал/шум.

3. Фазовая манипуляция (PSK) — один из видов фазовой модуляции, при которой фаза несущего колебания меняется скачкообразно в зависимости от информационного сообщения.

Этот набор манипуляций определяется основными характеристиками, которыми обладает любой сигнал. Для моделирования модуляции цифрового сигнала в ходе лаботаторной работы предлагается использовать следующий набор функций:

1. Функция randerr предназначена для формирования ошибок в цифровом сигнале. Она дает матрицу, в каждой строке которой имеется заданное число случайно расположенных ненулевых элементов.

2. Для оценки помехоустойчивости системы связи необходимо произвести сравнение исходного (передаваемого) сообщения с сообщением, полученным в результате приема, и определить число ошибок, возникших в процессе передачи, а также вероятность ошибки. Эти действия выполняются функциями symerr и biterr, первая из которых подсчитывает число несовпадающих символах в двух сообщениях, а вторая — число несовпадающих битов в двоичных представлениях этих символов. Кроме числа ошибок, обе функции могут возвращать долю ошибок в общем числе символов (битов) и индикаторы мест возникновения ошибок.

3. Последние две функции данной группы предназначены для графического отображения сигналов с квадратурной манипуляцией. Функция eyediagram выводит так называемую глазковую диаграмму, а функция scatterplot — диаграмму рассеяния.

9.3 Ход работы

9.3.1 Код matlab

```
%BPSK modulation
h = modem.pskmod('M', 2);
g = modem.pskdemod('M', 2);
```

```
msg = randint(10,1,2);
modSignal = modulate(h,msg);
errSignal = (randerr(1,10, 3) ./ 30)';
modSignal = modSignal + errSignal;
demodSignal = demodulate(g,modSignal);
scatterplot(modSignal);
eyediagram(modSignal,10);
scatterplot(demodSignal);
eyediagram(demodSignal,10);

%PSK modulation
h = modem.pskmod('M', 8);
g = modem.pskdemod('M', 8);
msg = randint(10,1,8);
modSignal = modulate(h,msg);
errSignal = (randerr(1,10, 3) ./ 30)';
modSignal = modSignal + errSignal;
demodSignal = demodulate(g,modSignal);
scatterplot(modSignal);
eyediagram(modSignal,10);
scatterplot(demodSignal);
eyediagram(demodSignal,10);

%OQPSK modulation
h = modem.oqpskmod;
g = modem.oqpskdemod;
msg = randint(200,1,4);
modSignal = modulate(h,msg);
errSignal = (randerr(1,400, 100) ./ 30)';
modSignal = modSignal + errSignal;
demodSignal = demodulate(g,modSignal);
scatterplot(modSignal);
eyediagram(modSignal,10);
scatterplot(demodSignal);
eyediagram(demodSignal,10);

%GENQAM modulation
M = 5;
h = modem.genqammod('Constellation', exp(1i*2*pi*[0:M-1]/M));
g = modem.genqamdemod('Constellation', exp(1i*2*pi*[0:M-1]/M));
msg = randint(10,1,8);
```

```
modSignal = modulate(h,msg);
errSignal = (randerr(1,10, 3) ./ 30)';
modSignal = modSignal + errSignal;
demodSignal = demodulate(g,modSignal);
scatterplot(modSignal);
eyediagram(modSignal,10);
scatterplot(demodSignal);
eyediagram(demodSignal,10);

%MSK modulation
h = modem.mskmod('SamplesPerSymbol', 10);
g = modem.mskdemod('SamplesPerSymbol', 10);
msg = randint(10,1,2);
modSignal = modulate(h, msg);
errSignal = (randerr(1,100, 3) ./ 30)';
modSignal = modSignal + errSignal;
demodSignal = demodulate(g, modSignal);
scatterplot(modSignal);
eyediagram(modSignal,10);
scatterplot(demodSignal);
eyediagram(demodSignal,10);
```

9.4 Результаты работы

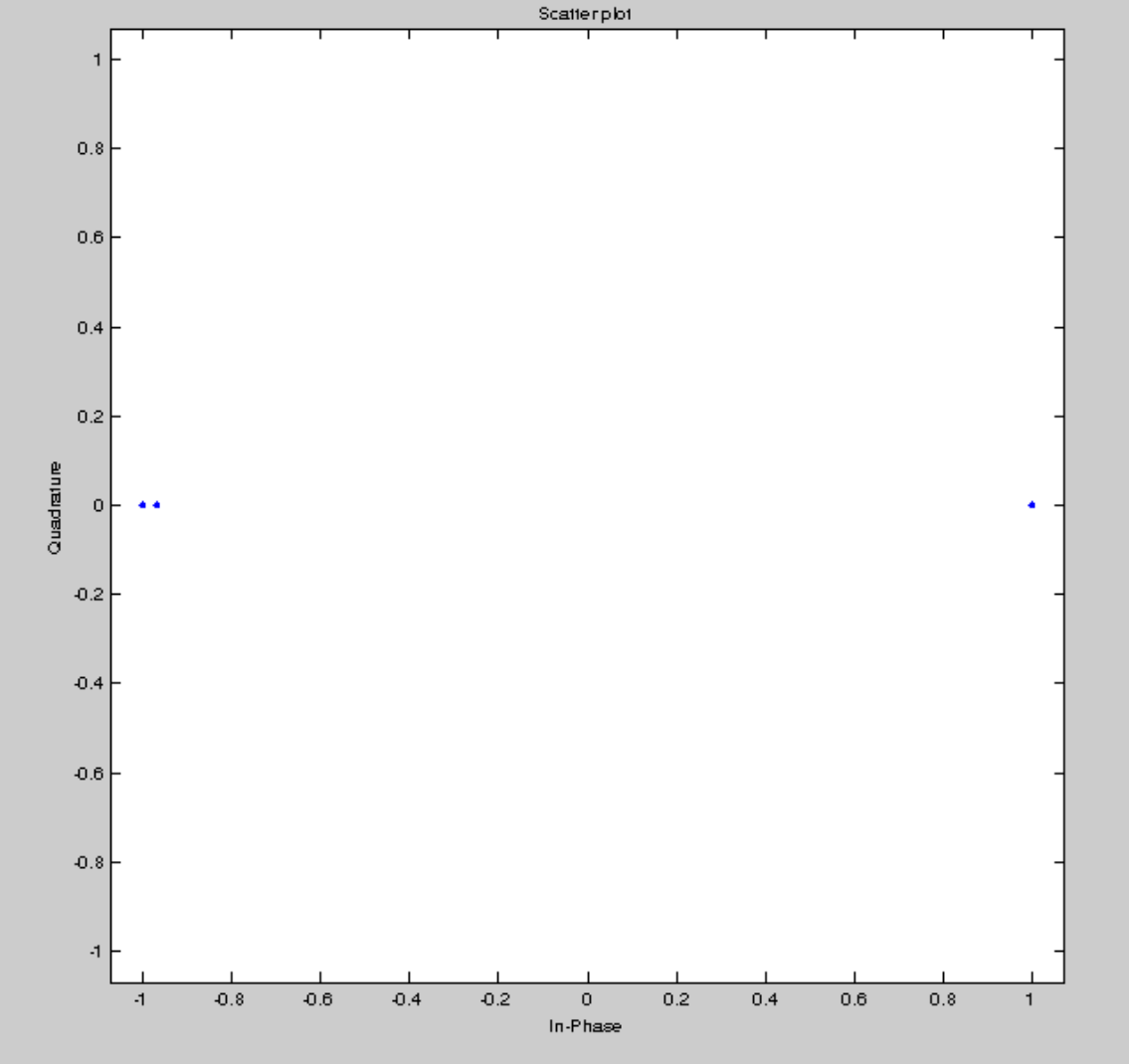

Рис. 9.1: Сигнальное созвездие BPSK-модуляции

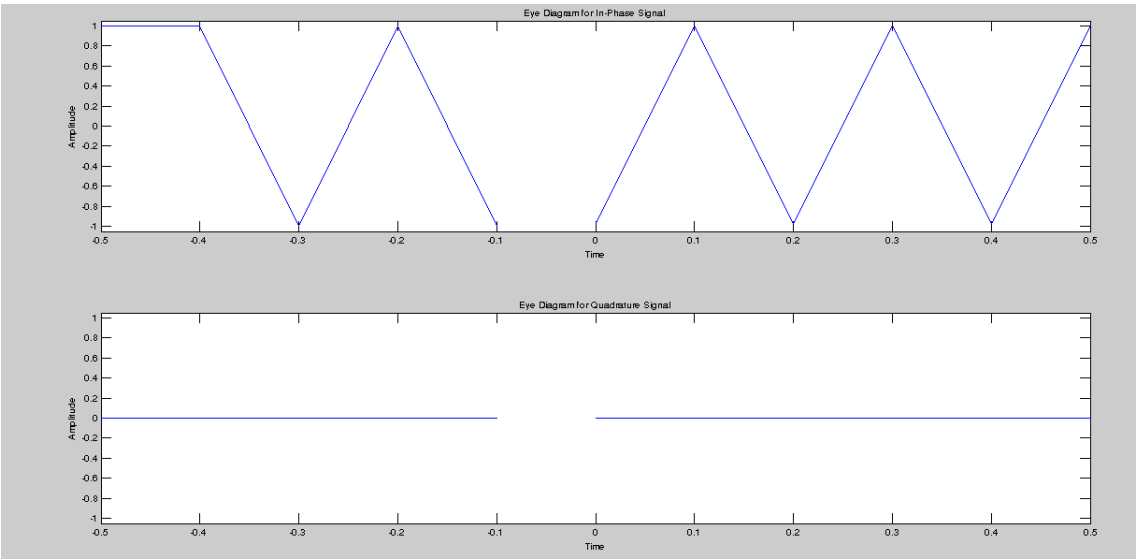

Рис. 9.2: Глазковая диаграмма BPSK-модуляции

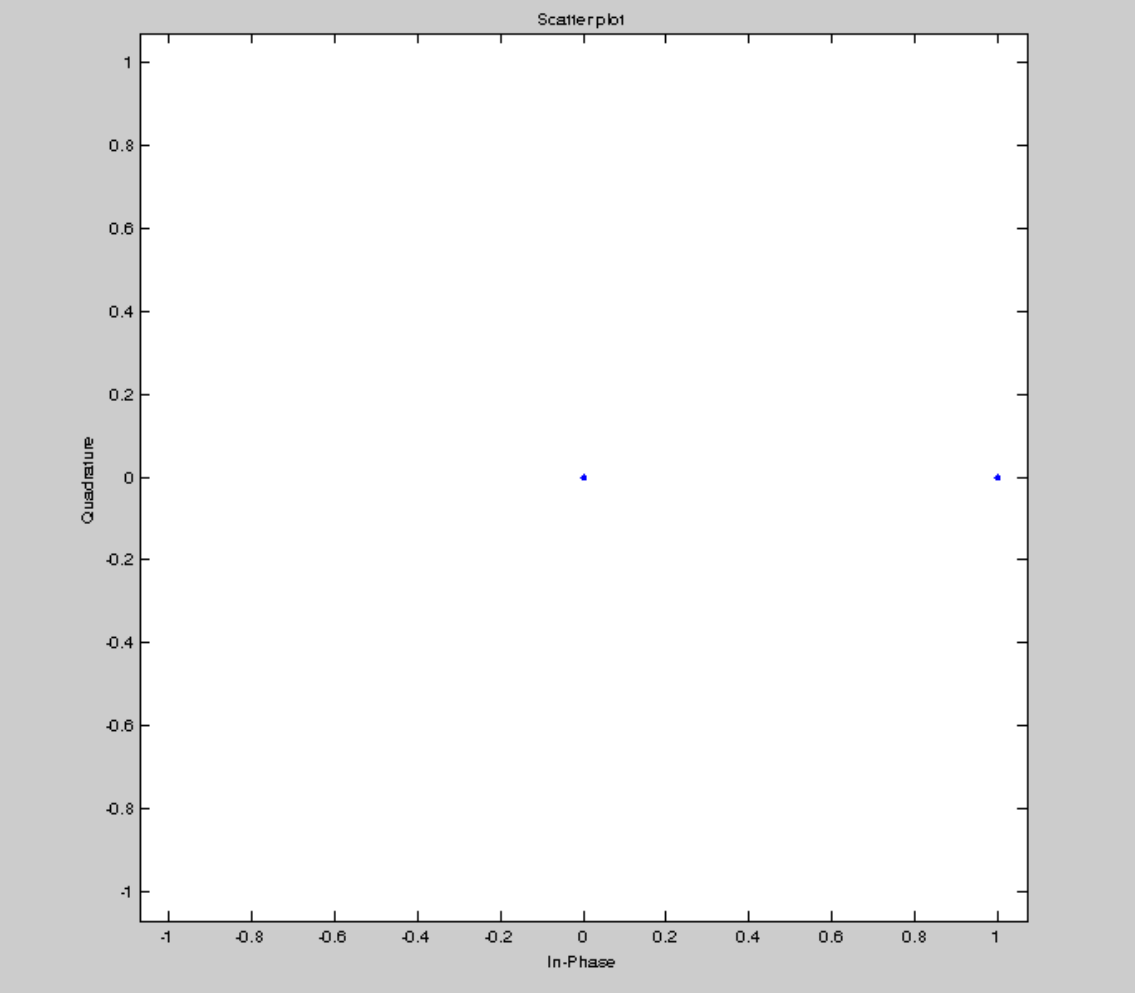

Рис. 9.3: Сигнальное созвездие BPSK-демодуляции

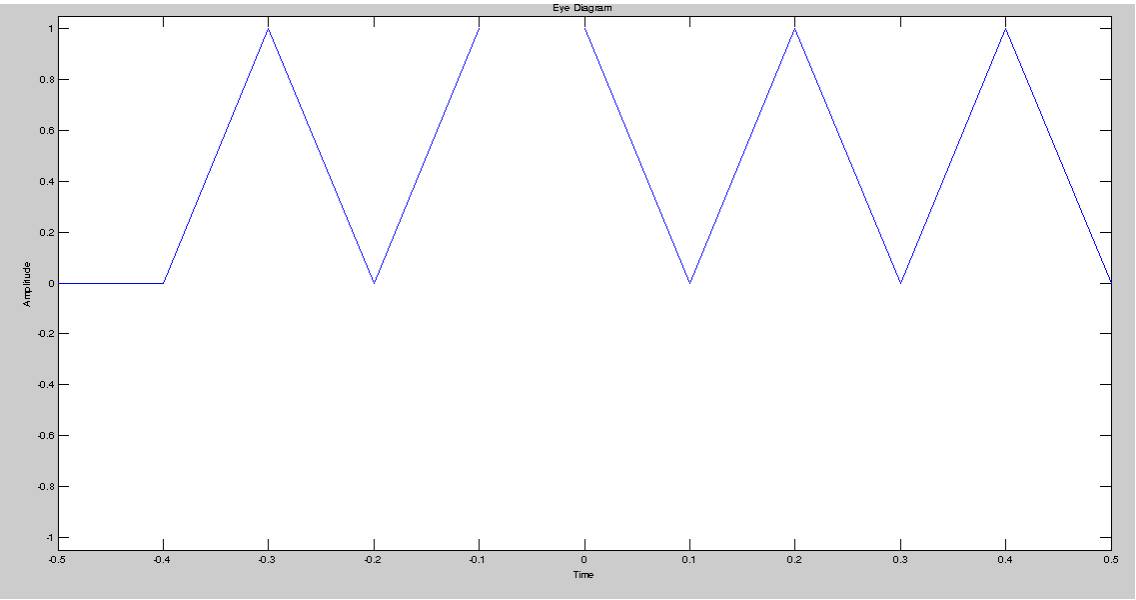

Рис. 9.4: Глазковая диаграмма BPSK-демодуляции

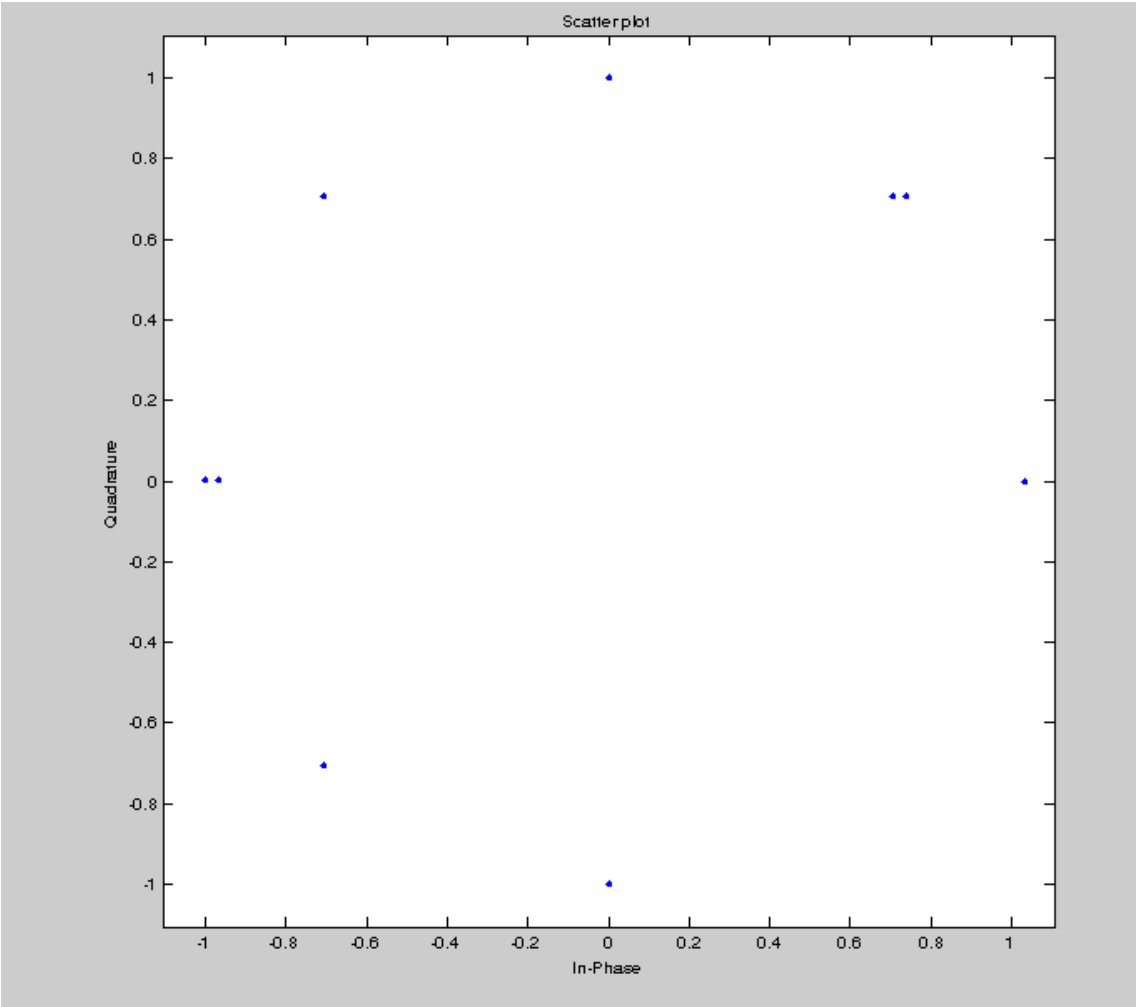

Рис. 9.5: Сигнальное созвездие PSK-модуляции

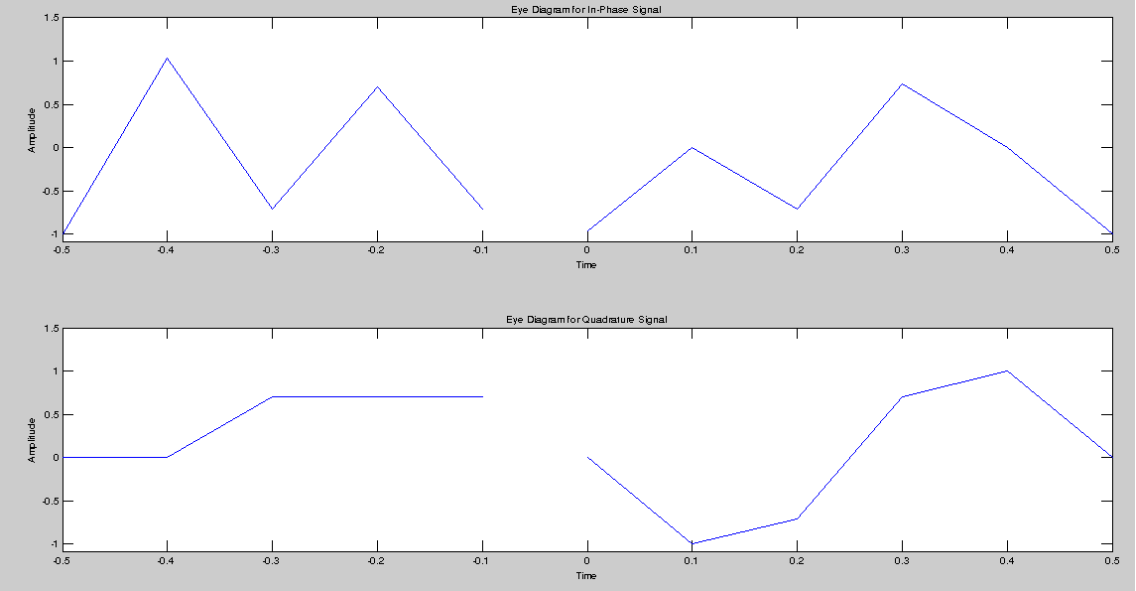

Рис. 9.6: Глазковая диаграмма PSK-модуляции

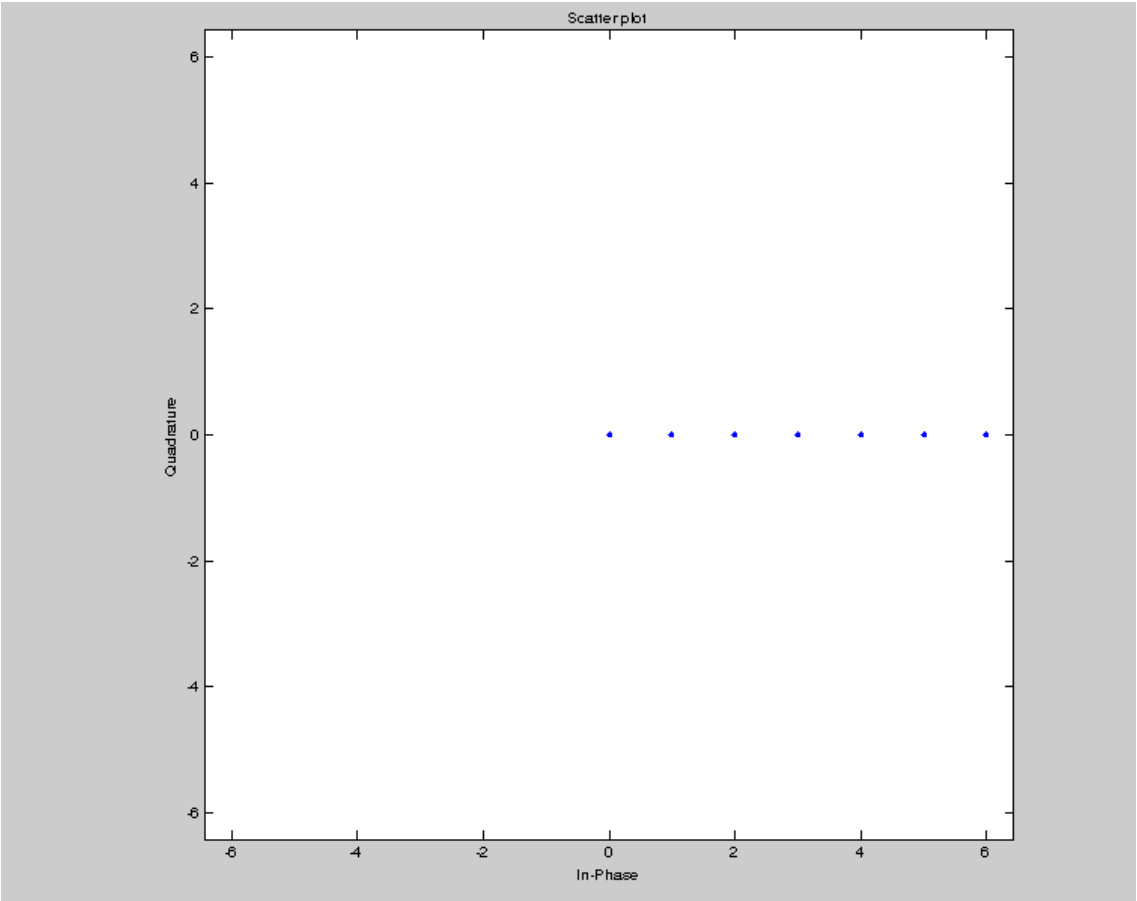

Рис. 9.7: Сигнальное созвездие PSK-демодуляции

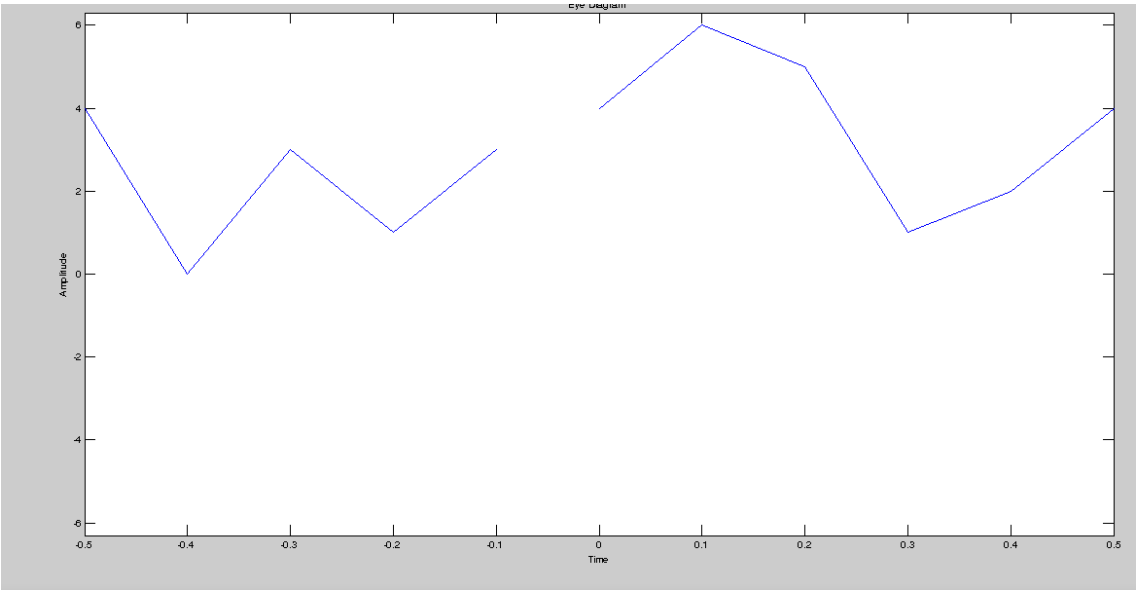

Рис. 9.8: Глазковая диаграмма PSK-демодуляции

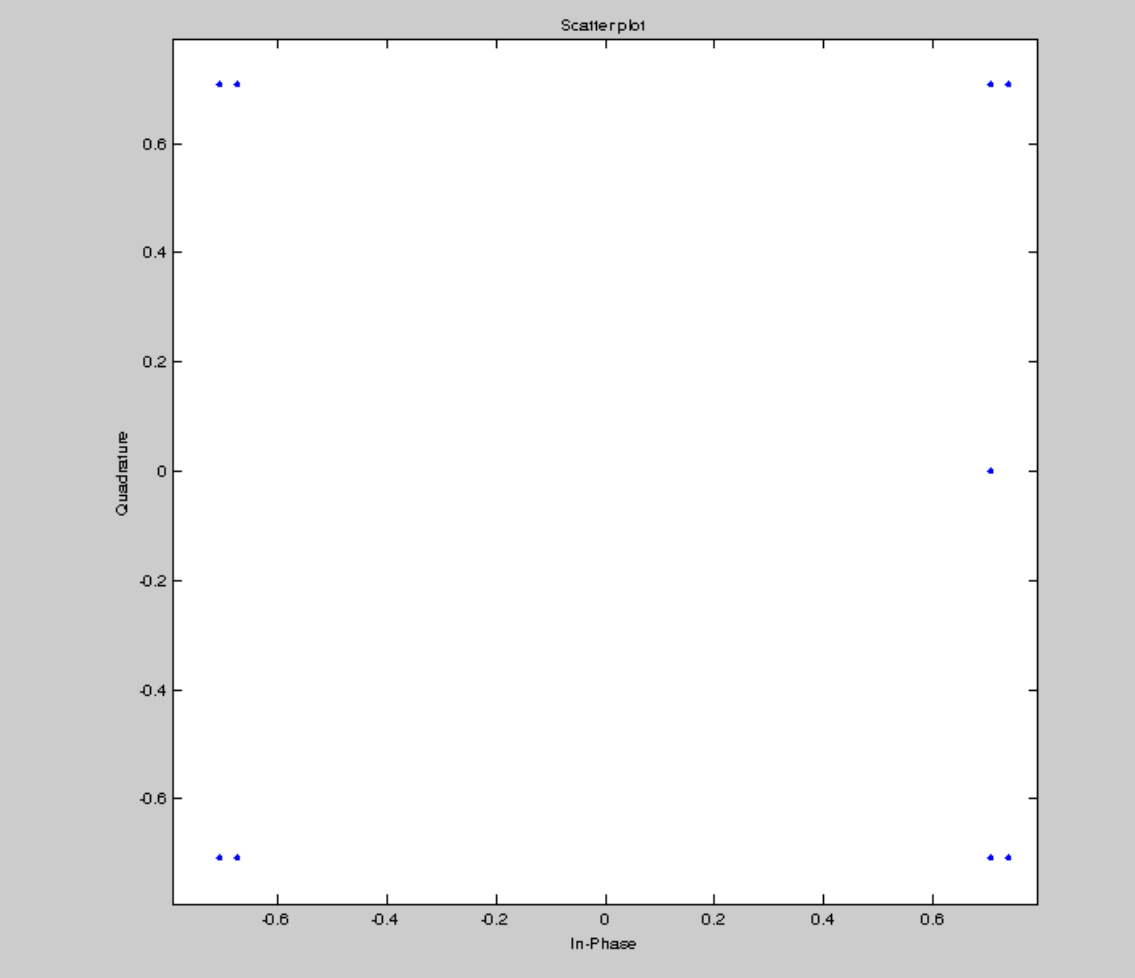

Рис. 9.9: Сигнальное созвездие OQPSK-модуляции

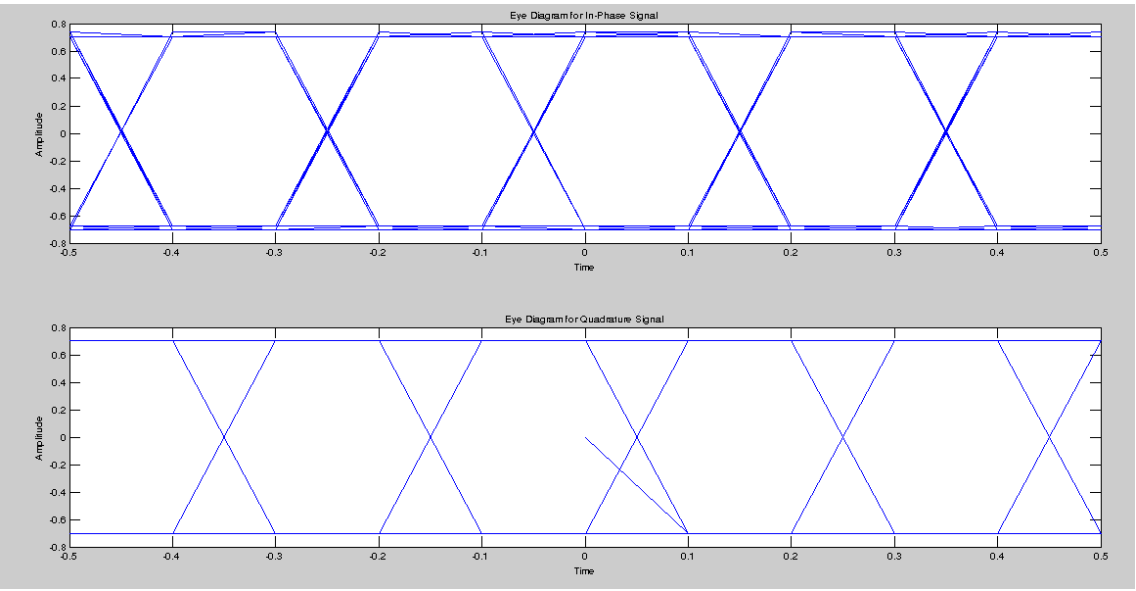

Рис. 9.10: Глазковая диаграмма OQPSK-модуляции

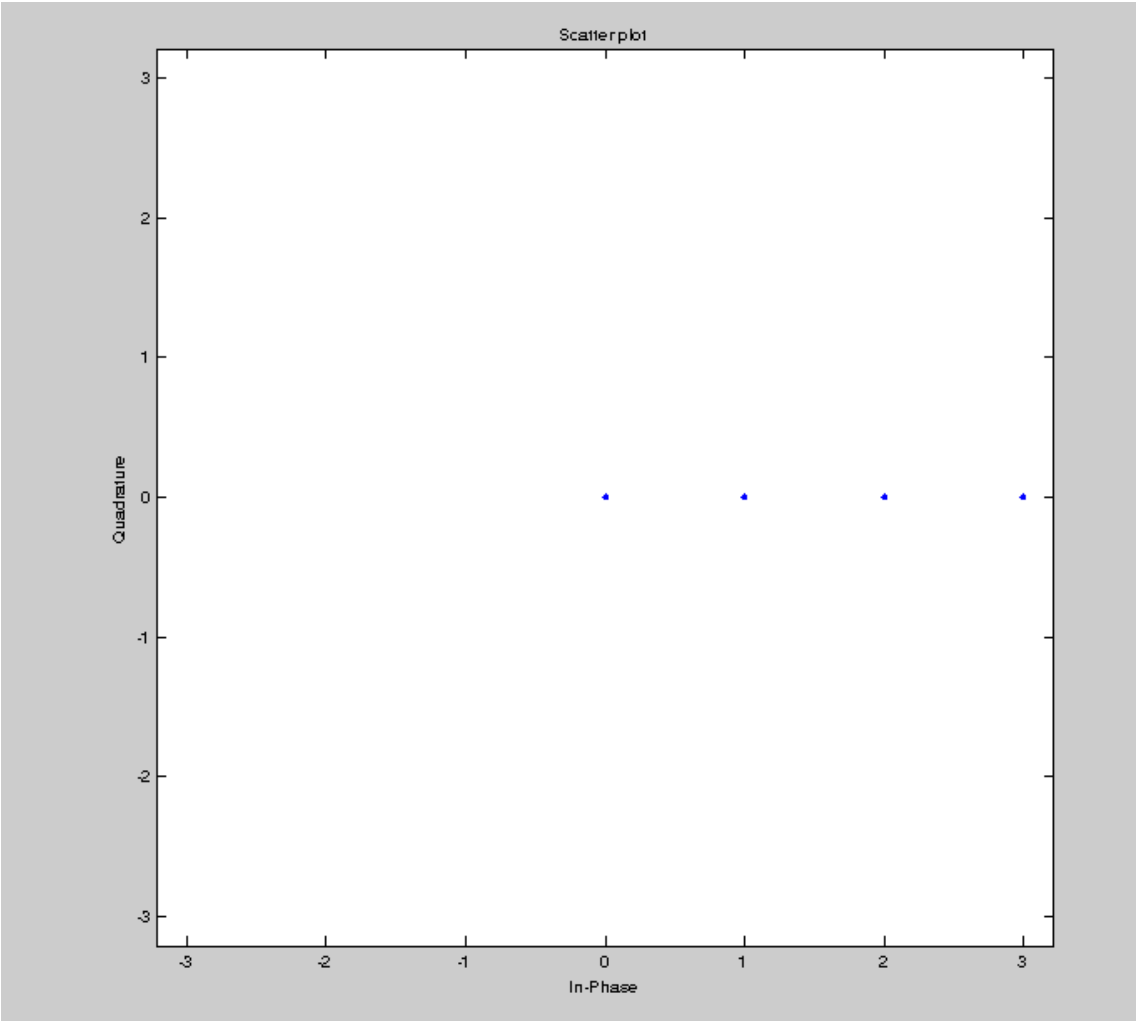

Рис. 9.11: Сигнальное созвездие OQPSK-демодуляции

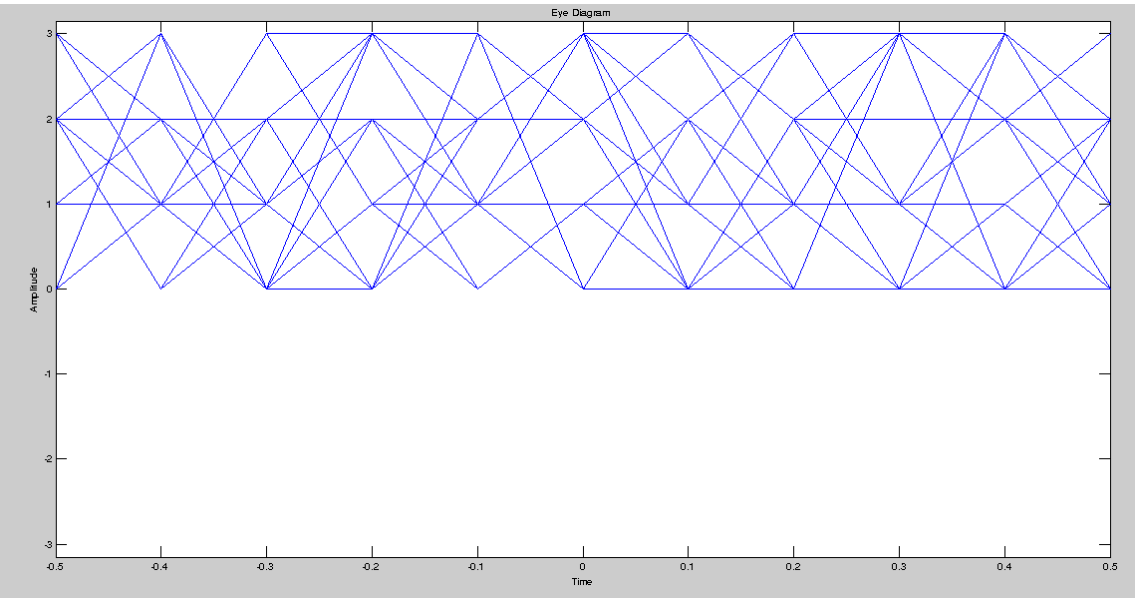

Рис. 9.12: Глазковая диаграмма OQPSK-демодуляции

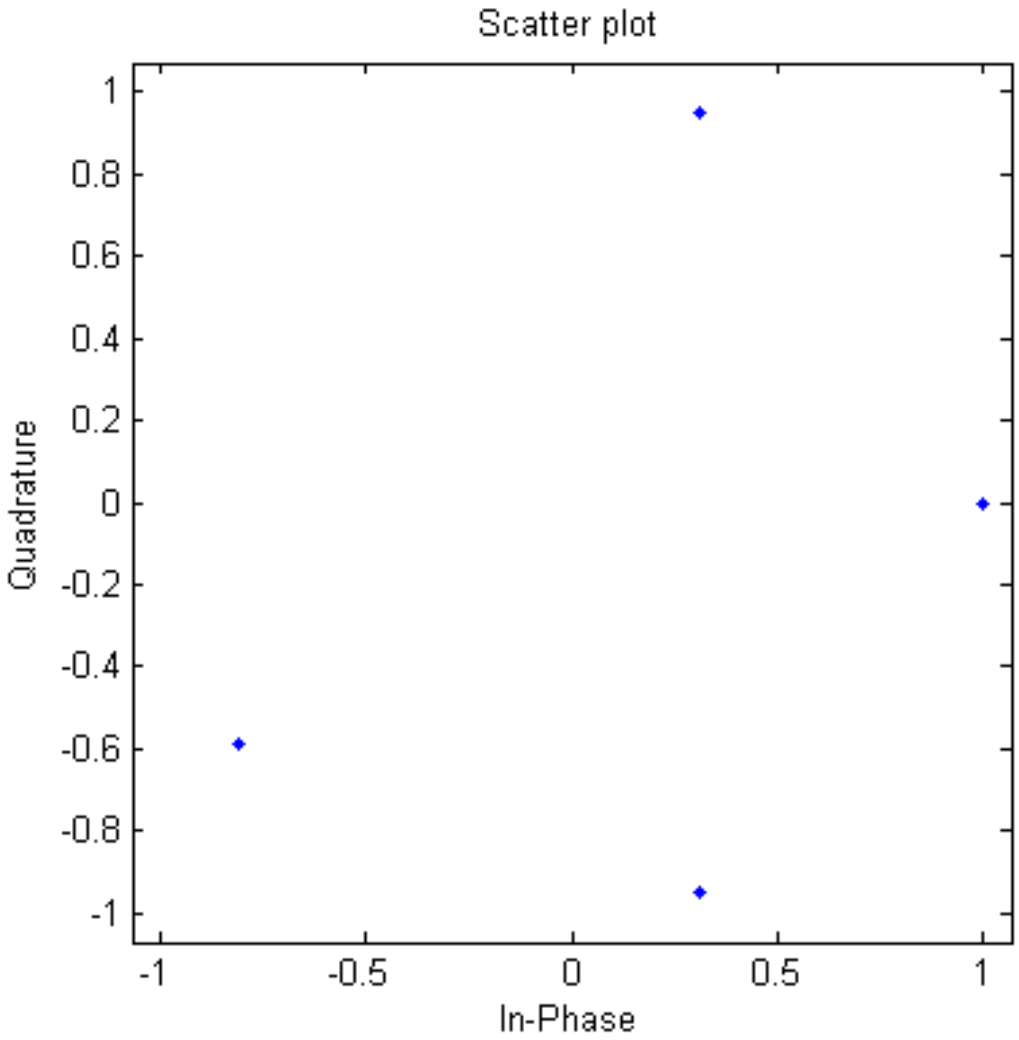

Рис. 9.13: Сигнальное созвездие genQAM-модуляции

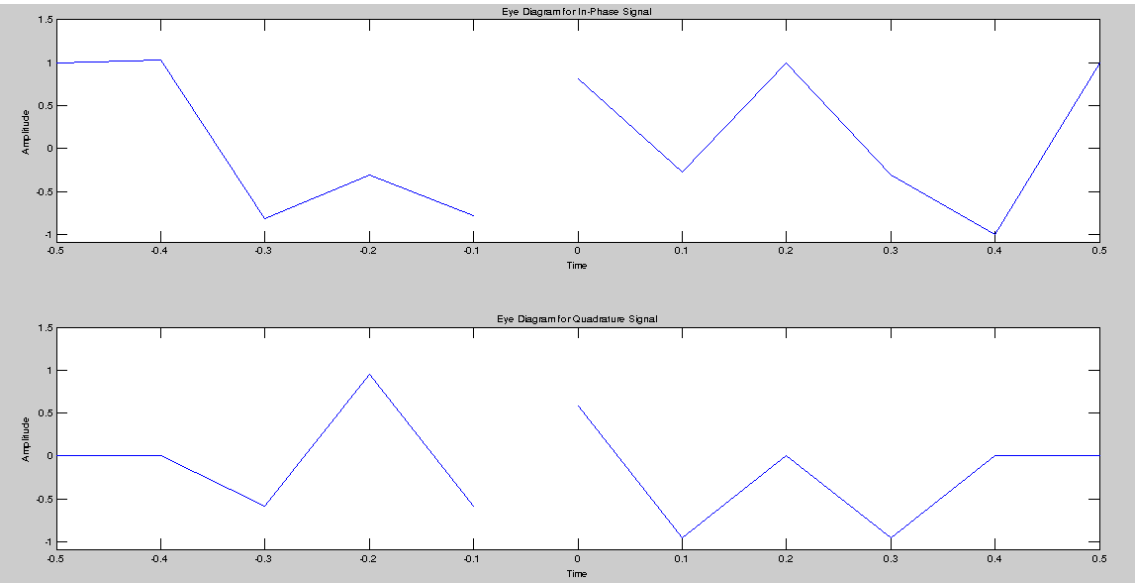

Рис. 9.14: Глазковая диаграмма genQAM-модуляции

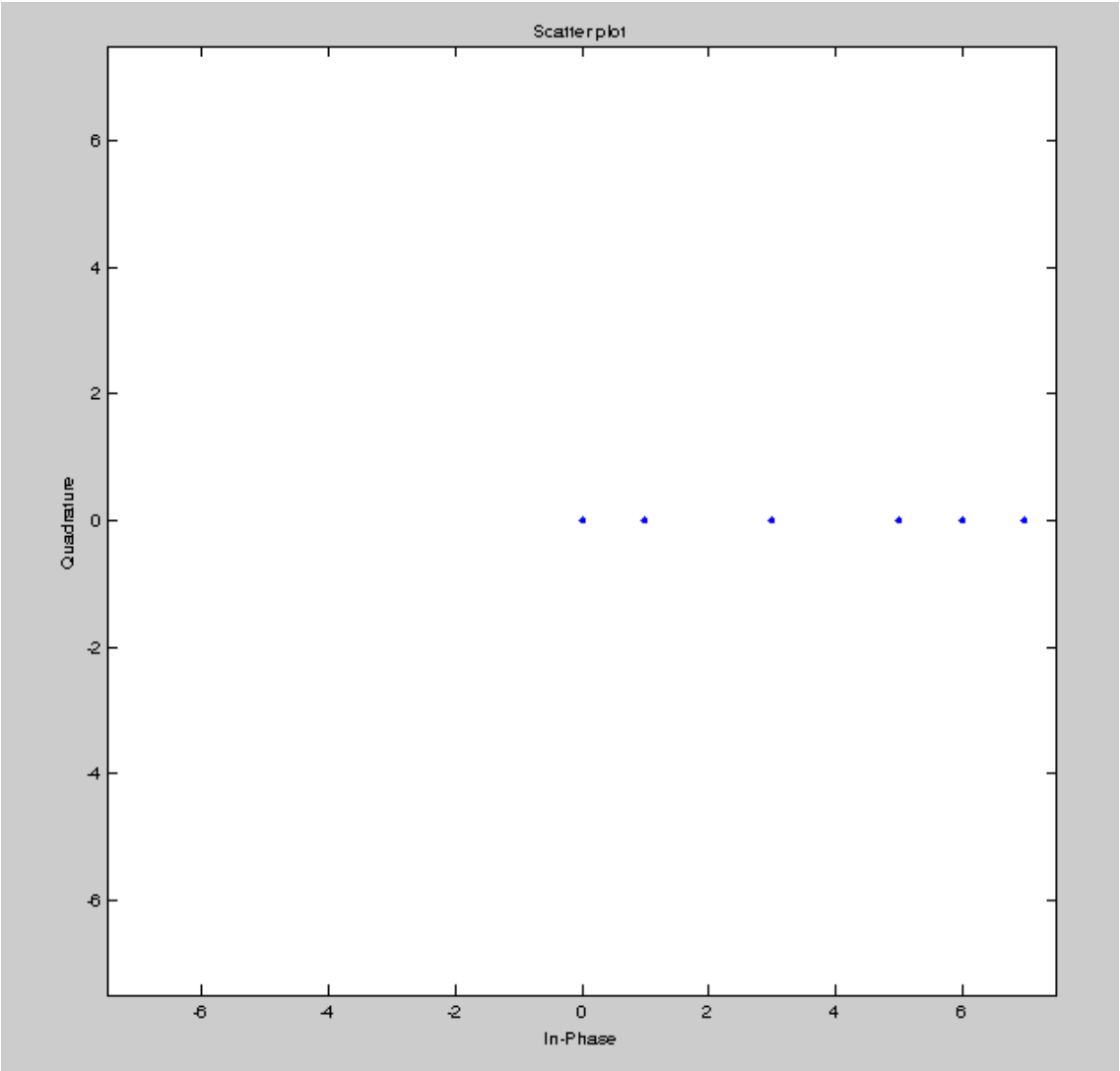

Рис. 9.15: Сигнальное созвездие genQAM-демодуляции

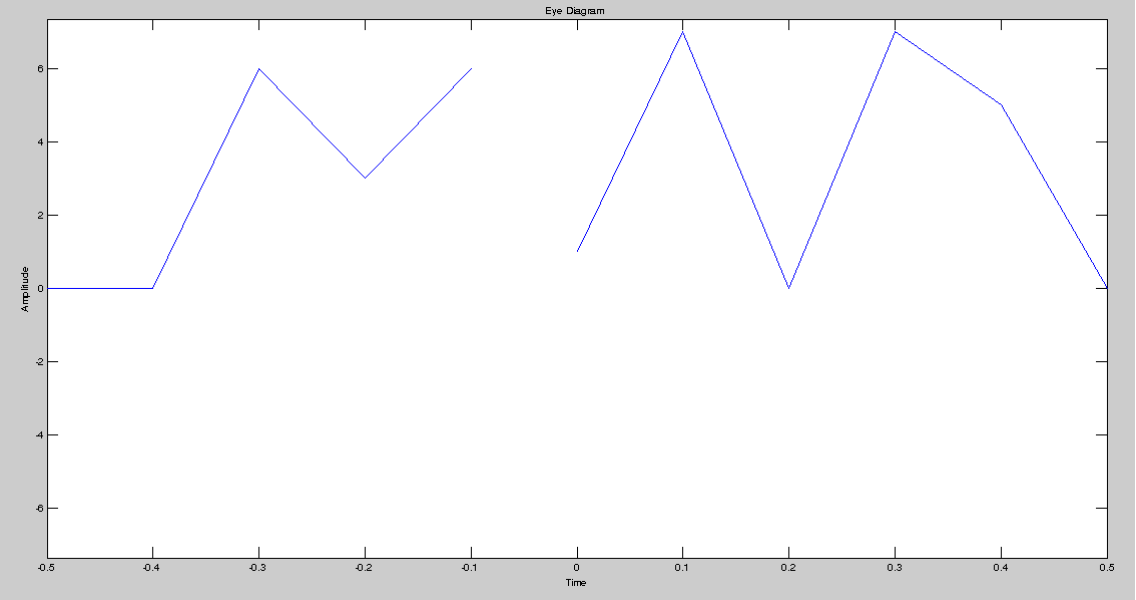

Рис. 9.16: Глазковая диаграмма genQAM-демодуляции

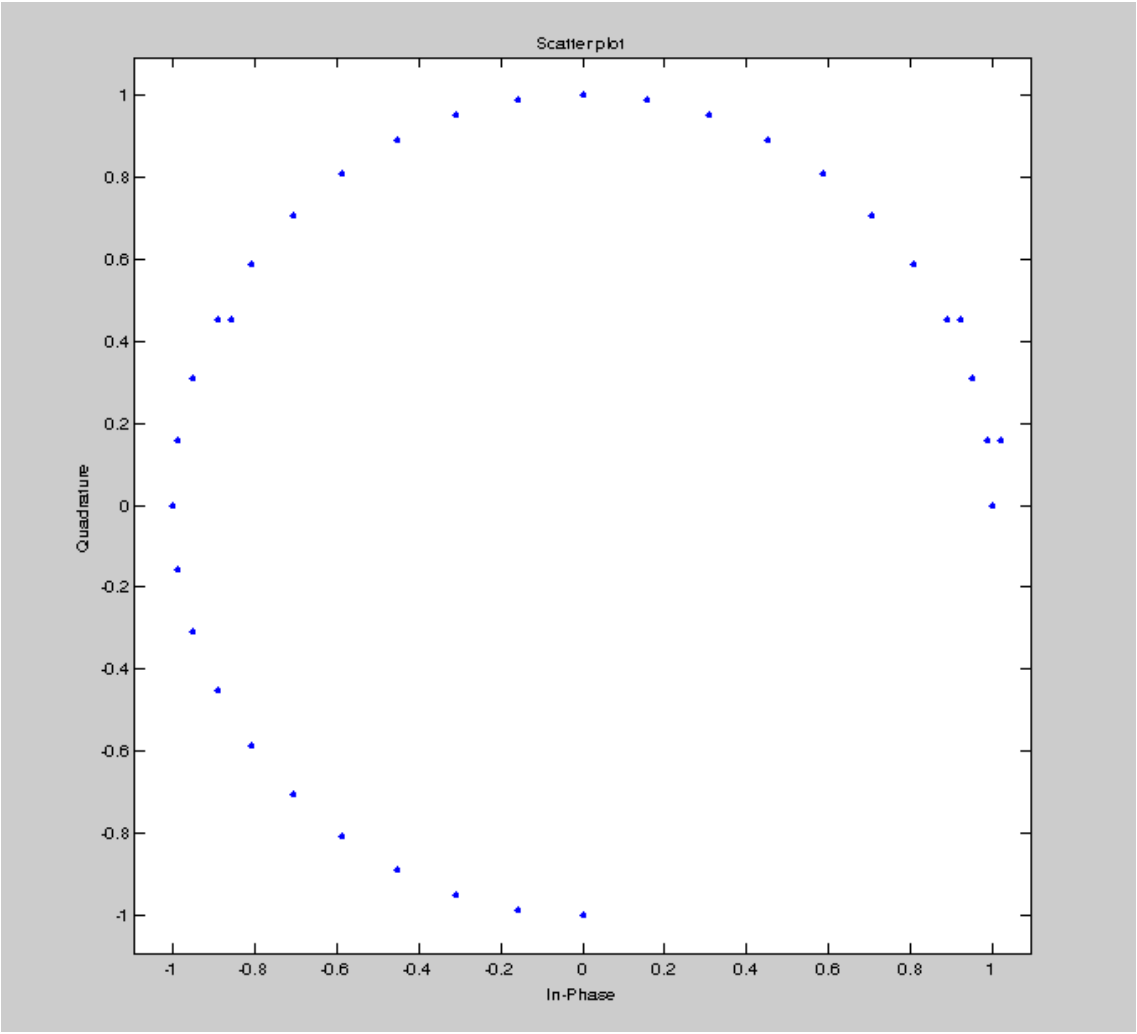

Рис. 9.17: Сигнальное созвездие MSK-модуляции

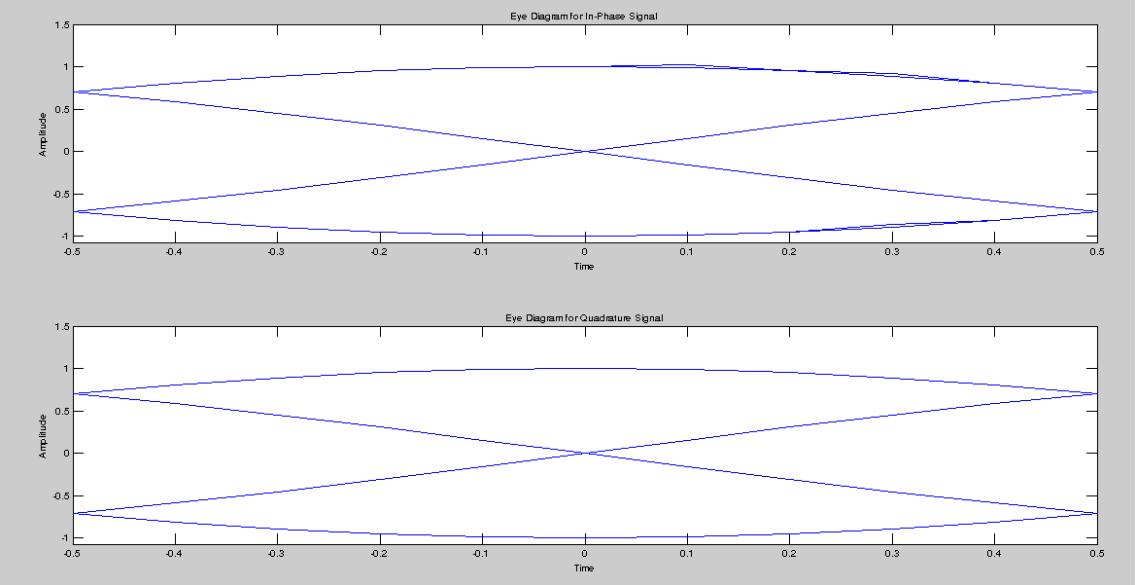

Рис. 9.18: Глазковая диаграмма MSK-модуляции

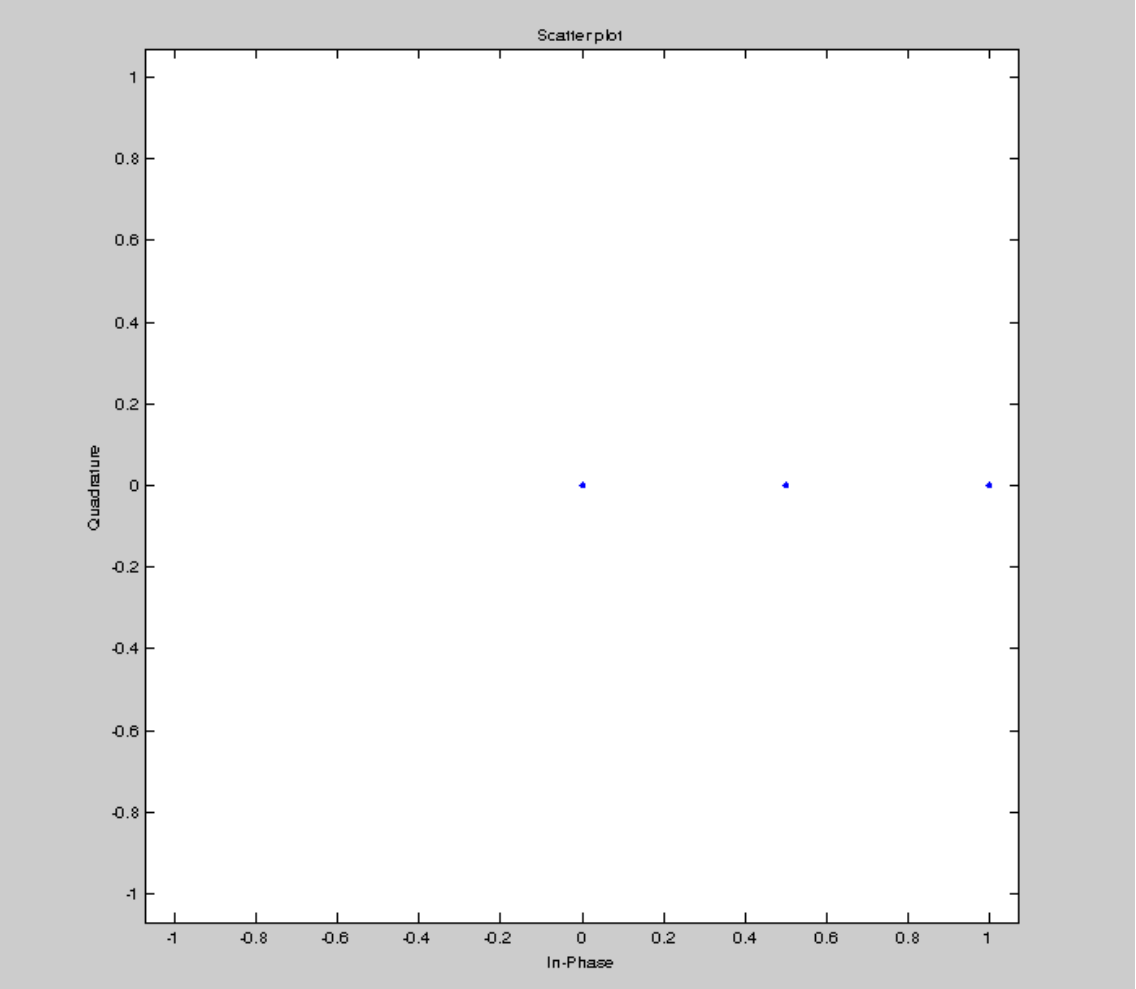

Рис. 9.19: Сигнальное созвездие MSK-демодуляции

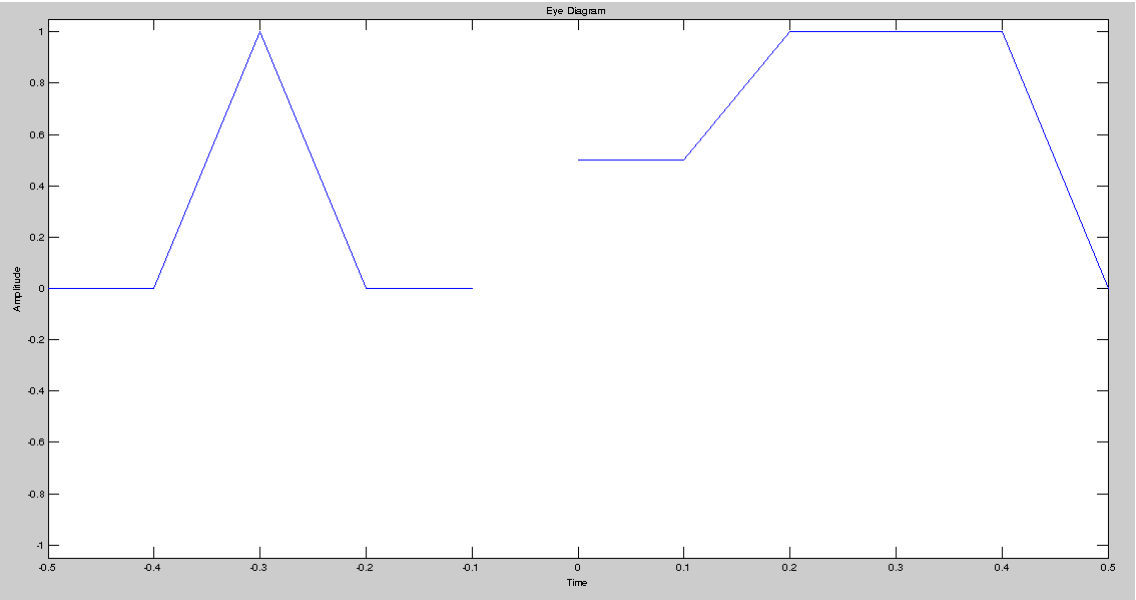

Рис. 9.20: Глазковая диаграмма MSK-демодуляции

Смоделируем проделаные опыты при помощи simulink

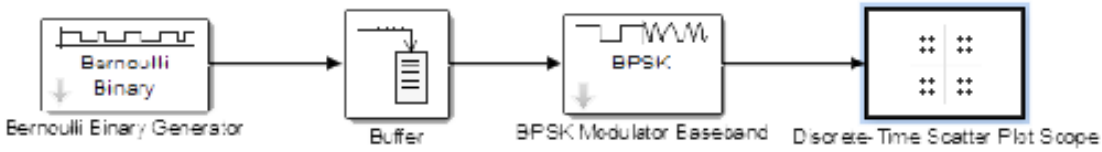

Рис. 9.21: Модель BPSK-модуляции

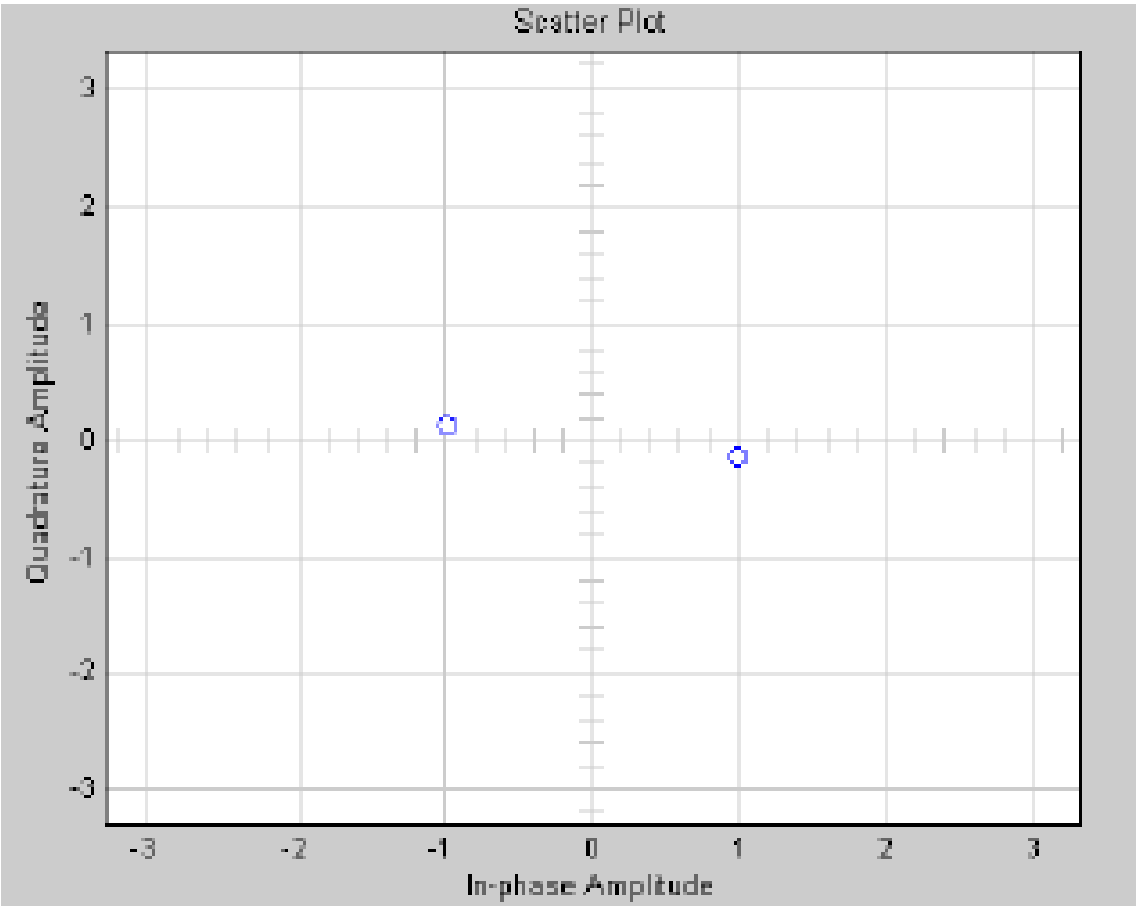

Рис. 9.22: Сигнальное созвездие BPSK-модуляции

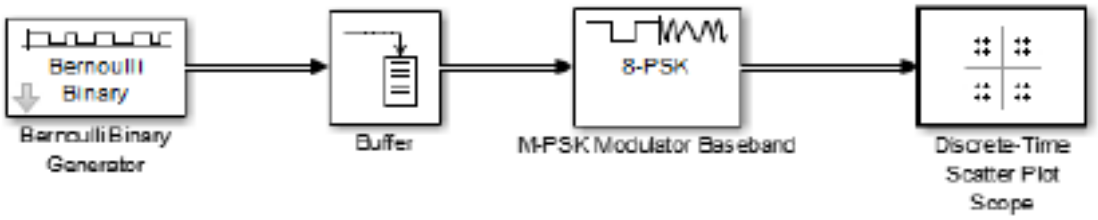

Рис. 9.23: Модель PSK-модуляции

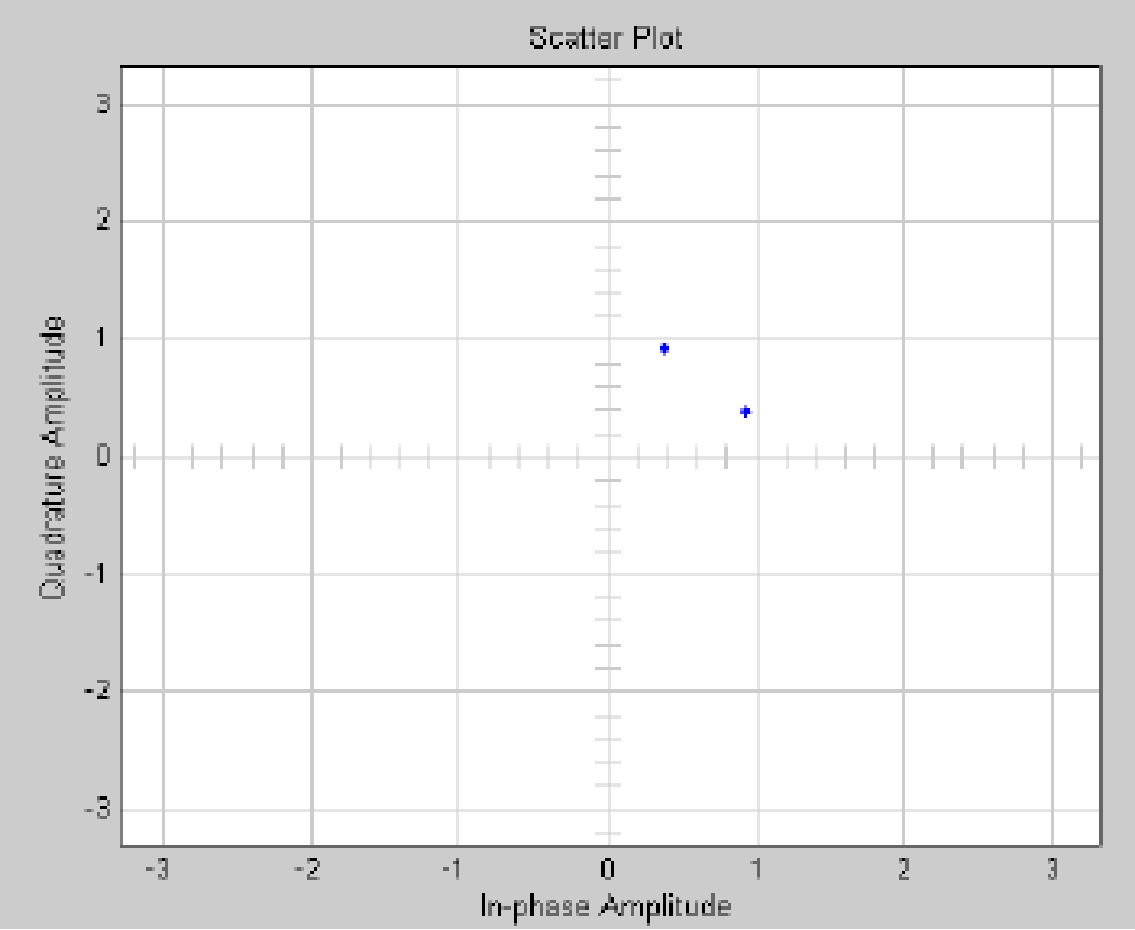

Рис. 9.24: Сигнальное созвездие PSK-модуляции.

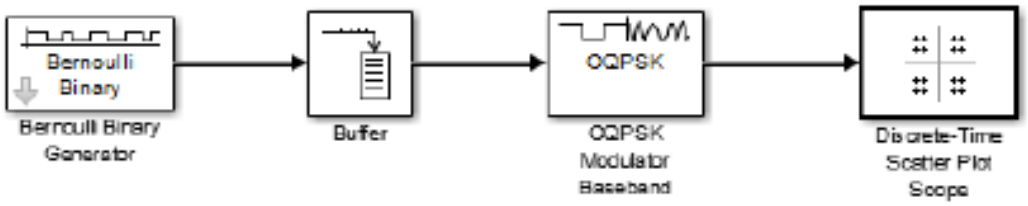

Рис. 9.25: Модель OQPSK-модуляции

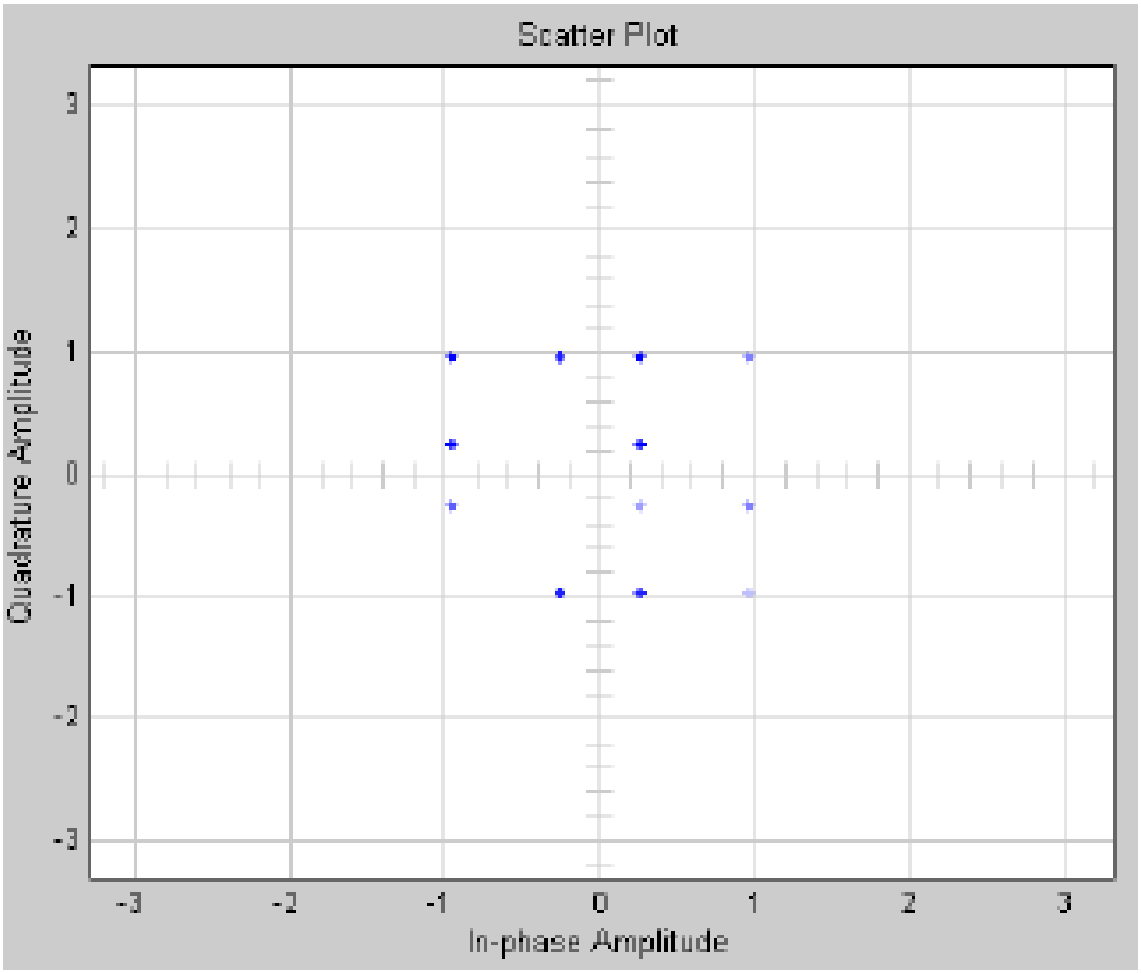

Рис. 9.26: Сигнальное созвездие OQPSK-модуляции.

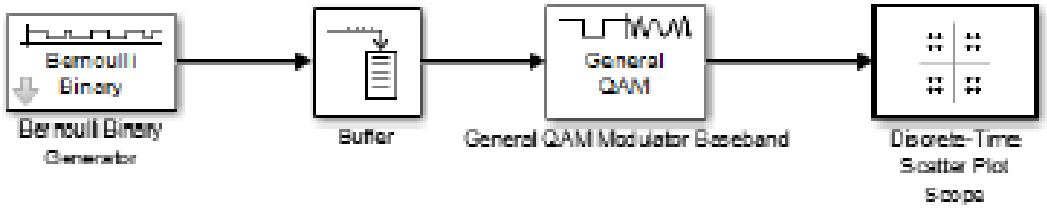

Рис. 9.27: Модель genQAM-модуляции

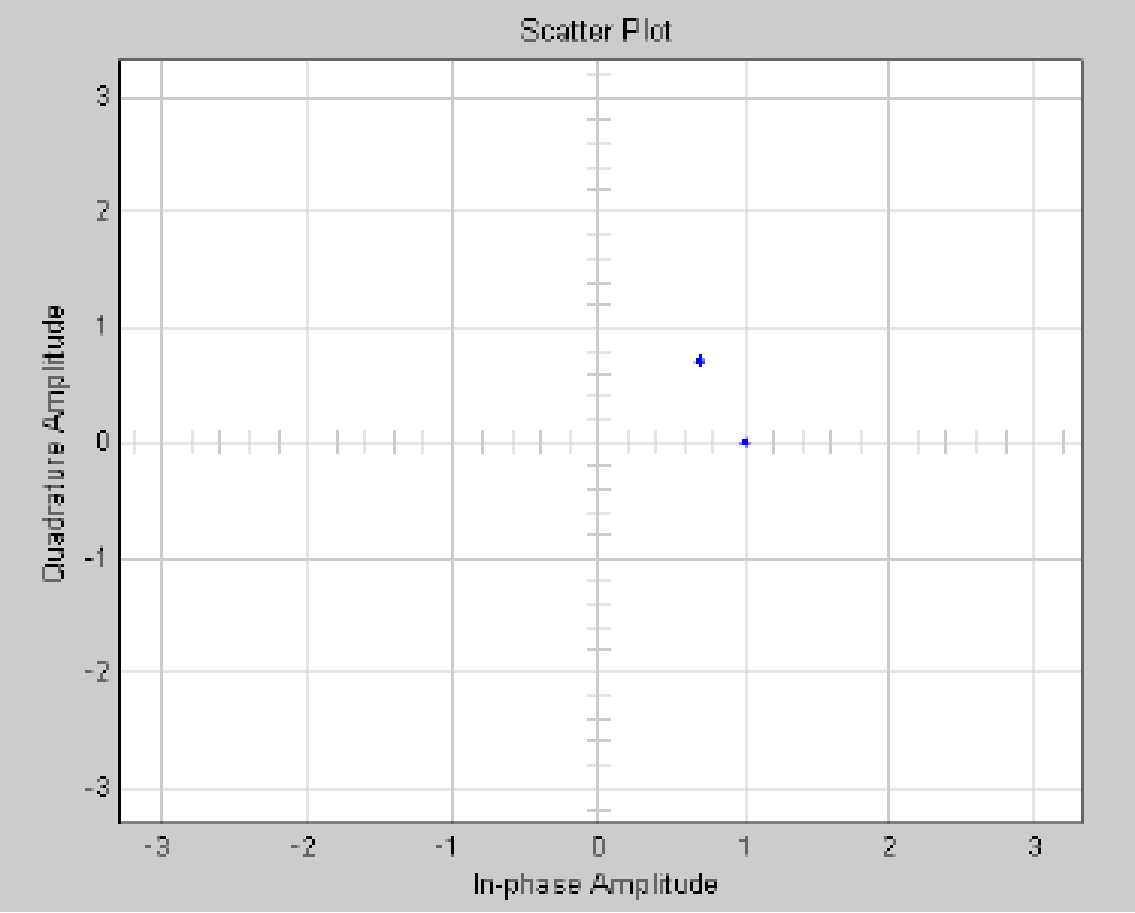

Рис. 9.28: Сигнальное созвездие genQAM-модуляции

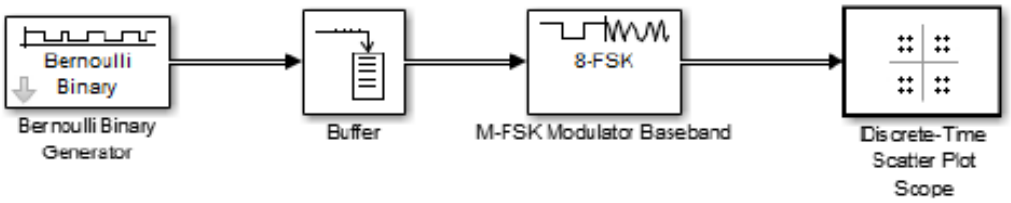

Рис. 9.29: Модель MSK-модуляции

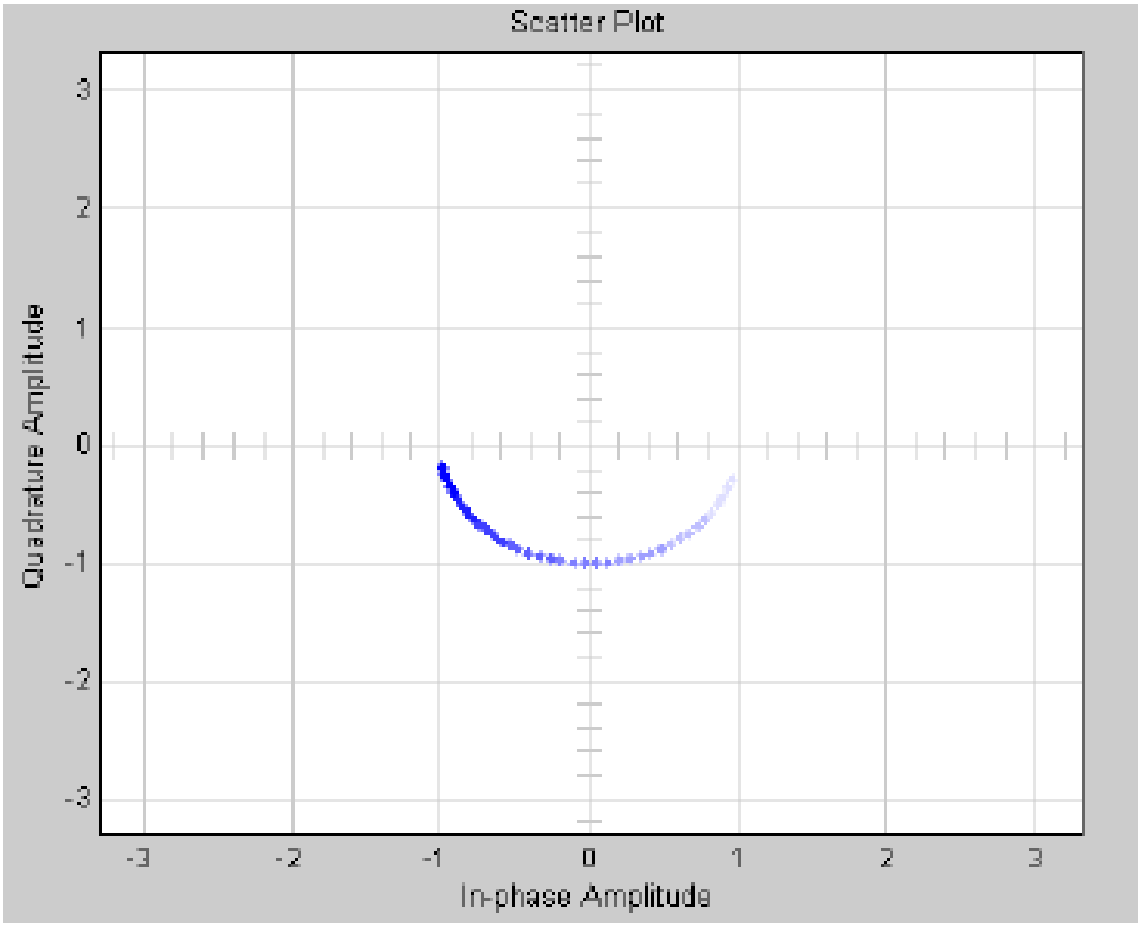

Рис. 9.30: Сигнальное созвездие MSK-модуляции.

9.5 Вывод

Уровень модуляции определяет количество состояний несущей, используемых для передачи информации. Чем выше этот уровень, тем большими скоростными возможностями и меньшей помехоустойчивостью модуляция обладает. Число бит, передаваемых одним состоянием, определяется как $LogN$, где N — уровень модуляции. Таким образом, чем выше уровень модуляции, тем больше данных мы можем передать (или потерять) за единицу времени. Так же уровень модуляции напрямую зависит от приемника сигнала: чем более точный приемник, тем больший уровень модуляции мы можем задать.