

**Министерство образования и науки РФ**

Федеральное государственное бюджетное образовательное  
учреждение высшего профессионального образования  
**ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ  
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)**

Радиотехнический факультет (РТФ)

Кафедра средств радиосвязи (СРС)

**Кологривов В.А.**

**ИССЛЕДОВАНИЕ  $\pi/4$ \_QPSK МОДЕМА  
(РЕАЛИЗАЦИЯ С ФАЗОВЫМ КОДЕРОМ)**

***Учебно-методическое пособие  
по лабораторной работе  
для студентов радиотехнических специальностей***

**2012**

**Кологривов В.А.**

**“Исследование  $\text{Pi}/4\text{-QPSK}$  модема (реализация с фазовым кодером)”**: Учебно-методическое пособие по лабораторной работе для студентов радиотехнических специальностей. – Томск: ТУСУР. Образовательный портал, 2012. – 38 с.

Учебно-методическое пособие содержит описание функциональной модели  **$\text{Pi}/4\text{-QPSK}$**  модема с фазовым кодером выполненной в среде функционального моделирования **Simulink** системы для инженерных и научных расчетов **MatLab**.

В пособии приведены сведения из теории квадратурной фазовой модуляции и демодуляции, краткая характеристика пакета **Simulink** системы **MatLab**, подробное описание виртуального лабораторного макета и используемых блоков библиотеки **Simulink**, а также требования к экспериментальному исследованию и контрольные вопросы, ответы на которые необходимы для успешной защиты лабораторной работы.

Пособие предназначено для студентов радиотехнических специальностей по направлениям: «Радиотехника», «Телекоммуникации» и др.

## АННОТАЦИЯ

Лабораторная работа “**Исследование  $\Pi/4$ \_QPSK модема (реализация с фазовым кодером)**” посвящена экспериментальному исследованию модема (модулятора-демодулятора)  **$\Pi/4$**  квадратурной фазовой манипуляции (с фазовым кодером) цифровых систем радиосвязи с использованием пакета функционального моделирования **Simulink**.

В описании сформулирована цель лабораторной работы, приведены необходимые сведения из теории квадратурной фазовой модуляции и демодуляции, краткая характеристика пакета **Simulink** системы **MatLab**, подробное описание виртуального лабораторного макета и используемых блоков библиотеки **Simulink**, а также требования к экспериментальному исследованию и контрольные вопросы, ответы на которые необходимы для успешной защиты лабораторной работы.

Исследование модема  **$\Pi/4$**  квадратурной фазовой манипуляции (с фазовым кодером) проводится по функциональной схеме, построенной на основе блоков базовых разделов библиотеки **Simulink**.

**СОДЕРЖАНИЕ**

<b>1. Цель работы. Краткие сведения из теории</b>	<b>5</b>
<b>2. Краткое описание пакета Simulink</b>	<b>8</b>
<b>2.1. Общая характеристика пакета Simulink</b>	<b>8</b>
<b>2.2. Запуск и работа с пакетом Simulink</b>	<b>9</b>
<b>3. Описание лабораторного макета</b>	<b>10</b>
<b>4. Описание используемых блоков библиотеки Simulink</b>	<b>22</b>
<b>5. Экспериментальное задание</b>	<b>36</b>
<b>6. Контрольные вопросы</b>	<b>37</b>
<b>Список использованных источников</b>	<b>37</b>

## 1. ЦЕЛЬ РАБОТЫ. КРАТКИЕ СВЕДЕНИЯ ИЗ ТЕОРИИ

**Цель работы:** Изучить структуру и принцип работы модема  $\text{Pi}/4$  квадратурной фазовой манипуляции (с фазовым кодером), используемой при реализации модуляторов и демодуляторов с кодовым принципом модуляции и демодуляции фазовых состояний несущей частоты.

### Теоретическая часть

Вариант функциональной **Sim**-модели модема **Pi/4\_QPSK** (**Pi/4\_Quadrature Phase Shift Key**) манипуляции (модуляции) с кодовым принципом модуляции и демодуляции фазы несущей частоты приведен на рисунке 1.1.

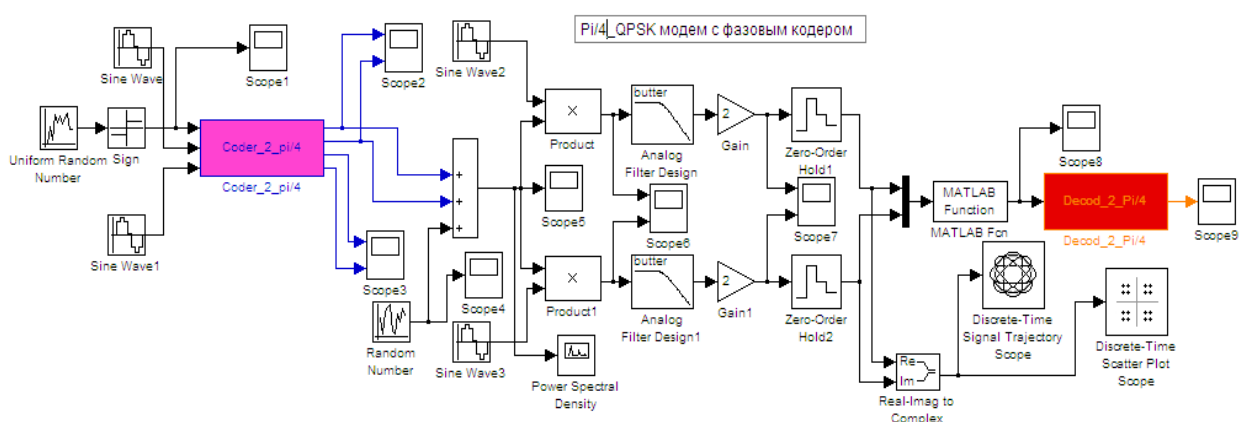


Рисунок 1.1 – Sim-модель модема  $\text{Pi}/4\_QPSK$  манипуляции с кодовым принципом модуляции и демодуляции фазовых состояний несущей частоты

Отличие данного  $\text{Pi}/4\_QPSK$  модулятора от *варианта реализации QPSK модулятора с фазовым кодером* заключается в том, что здесь входной поток битовых импульсов группируется в *чередующиеся нечетные и четные дибиты*. Дибиты, с учетом четности, преобразуются вначале в импульсы по амплитуде пропорциональные фазовому состоянию  $\varphi_k$ , а затем в квадратурные модулирующие импульсы пропорциональные  $\cos(\varphi_k)$  и  $\sin(\varphi_k)$ , совпадающие по длительности с дибитом. Этап фазовой модуляции реализован в данном случае, путем умножения текущих значений квадратурных составляющих несущей частоты  $\cos(\omega \cdot t_k)$  и  $\sin(\omega \cdot t_k)$ , соответственно на значения  $\cos(\varphi_k)$  и  $\sin(\varphi_k)$ , определяемые текущим фазовым состоянием. В канал поступает сумма фазоманипулированных квадратурных составляющих несущей частоты, что соответствует аналитическому представлению фазоманипулированного сигнала [1]

$$\cos(\omega \cdot t_k) \cdot \cos(\varphi_k) + \sin(\omega \cdot t_k) \cdot \sin(\varphi_k) = \cos(\omega \cdot t_k - \varphi_k).$$

Квадратурная фазовая манипуляция может быть реализован и в виде функциональной схемы (см. рисунок 1.2.).

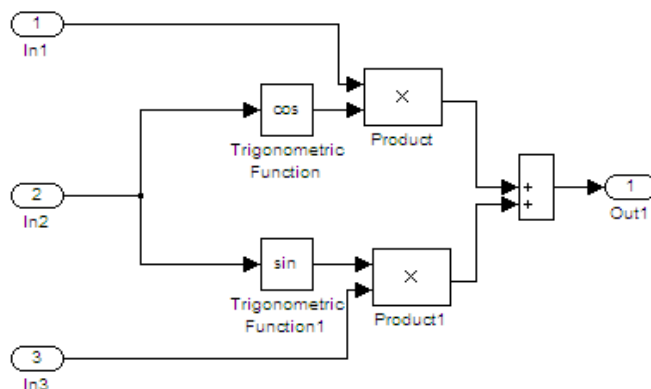


Рисунок 1.2 – Функциональная схема квадратурного фазового манипулятора

Здесь на входы **In1** и **In3** поступают квадратурные колебания несущей частоты  $\cos(\omega \cdot t_k)$  и  $\sin(\omega \cdot t_k)$ , а на вход **In2** поступают модулирующие значения фазового состояния несущей  $\varphi_k$ . С выхода сумматора в канал передачи поступает сумма фазоманипулированных квадратурных составляющих, что соответствует фазовой манипуляции колебаний несущей частоты.

Описанный принцип работы квадратурного фазового **PSK** манипулятора с фазовым кодером используется при реализации (**QPSK**, **OQPSK**, **Pi/4\_QPSK**, **MSK**, **GMSK**) модуляторов, функционирующих по принципу кодирования фазовых состояний несущей частоты. Данный принцип построения модуляторов легко переносится на общий случай **M**-арной фазовой манипуляции **8\_PSK**, **16\_PSK**, **32\_PSK**, **64\_PSK** и так далее.

**Модуляция типа Pi/4\_QPSK** основана на разбиении исходной последовательности импульсов на нечетные и четные дибиты [2] и их отдельном согласованном кодировании квадратурными  $\pi/2$  фазовыми сдвигами, смещенными на  $\pi/4$  (см. рисунок 1.3.).

Для обеспечения равного кодового расстояния между составляющими сигнального пространства при кодировании используется код Грея. В результате **Pi/4\_QPSK** модуляции реализуется восемь состояний фаз, четыре фазовых состояния для нечетных символов (дибит) и четыре для четных символов (дибит). Таким образом, конкретный дибит определяет лишь четыре из восьми состояний фаз. Соответствие нечетных и четных дибит и фазовых состояний с учетом согласованного кодирования по Грею приведено в таблице.

В результате чередования нечетных и четных дибит и их отдельном кодировании оказываются возможными фазовые переходы лишь в пределах  $\pm\pi/4$  и  $\pm 3 \cdot \pi/4$ . Напомним, что в **QPSK** и **OQPSK** модуляциях

максимальные значения скачков фазы составляют соответственно  $\pm\pi$  и  $\pm\pi/2$ .

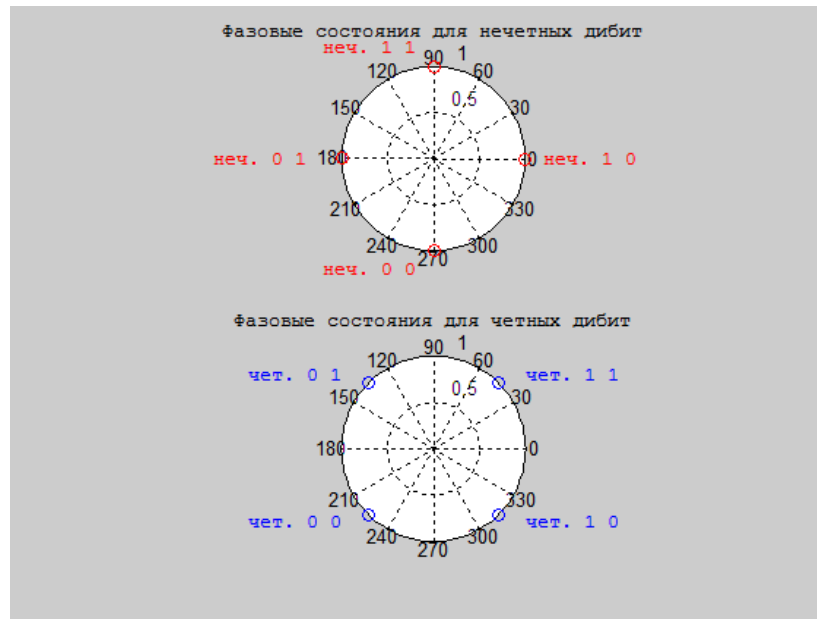


Рисунок 1.3 – Диаграммы фазовых состояний для нечетных и четных дибит Pi/4\_QPSK модуляции

Таблица кодирования нечетных и четных дибит при Pi/4\_QPSK

	Модулирующие символы (дибиты) и соответствующие им фазовые состояния			
	0_0	0_1	1_0	1_1
Нечетн.	$-2 \cdot \pi/4$	$4 \cdot \pi/4$	$0 \cdot \pi/4$	$2 \cdot \pi/4$
Четн.	$-3 \cdot \pi/4$	$3 \cdot \pi/4$	$-1 \cdot \pi/4$	$1 \cdot \pi/4$

Таким образом, при **Pi/4\_QPSK** модуляции кратковременный провал уровня сигнала в узкополосном канале оказывается меньшим, чем при **QPSK** модуляции, но большим, чем при **OQPSK** модуляции, что частично способствует повышению компактности спектра сигнала, соответственно, увеличению скорости убывания внеполосного излучения и возможности повышения **КПД** используемых усилителей мощности.

**Демодуляция** принятого квадратурно-фазоманипулированного сигнала на несущей частоте в данном случае осуществляется путем разветвления его на квадратурные каналы и подачи на первые входы умножителей. На вторые входы умножителей поступают синхронизированные опорные косинусоидальные  $\cos(\omega \cdot t_k)$  и синусоидальные  $\sin(\omega \cdot t_k)$  колебания несущей частоты. Фильтры нижних частот (**ФНЧ**), стоящие на выходах умножителей, отфильтровывают составляющие на второй гармонике несущей частоты и выделяют изменения постоянной составляющей,

пропорциональные  $\cos(\varphi_k)$  и  $\sin(\varphi_k)$ , позволяющие определить  $\varphi_k$  и декодировать соответствующие дибиты.

Математическая запись процесса демодуляции может быть представлена в виде

$$\begin{aligned} \cos(\omega \cdot t_k) \cdot \cos(\omega \cdot t_k \mp \varphi_s) &= \cos(\omega \cdot t_k) \cdot \langle \cos(\omega \cdot t_k) \cdot \cos(\varphi_k) \pm \sin(\omega \cdot t_k) \cdot \sin(\varphi_k) \rangle = \\ &= \frac{1}{2} \cdot \cos(\varphi_k) \cdot \langle 1 + \cos(2 \cdot \omega \cdot t_k) \rangle \mp \frac{1}{2} \cdot \sin(\varphi_k) \cdot \sin(2 \cdot \omega \cdot t_k) \Rightarrow \frac{1}{2} \cdot \cos(\varphi_k); \end{aligned}$$

$$\begin{aligned} \sin(\omega \cdot t_k) \cdot \cos(\omega \cdot t_k \mp \varphi_s) &= \sin(\omega \cdot t_k) \cdot \langle \cos(\omega \cdot t_k) \cdot \cos(\varphi_k) \pm \sin(\omega \cdot t_k) \cdot \sin(\varphi_k) \rangle = \\ &= \frac{1}{2} \cdot \sin(\varphi_k) \cdot \langle 1 - \cos(2 \cdot \omega \cdot t_k) \rangle \mp \frac{1}{2} \cdot \cos(\varphi_k) \cdot \sin(2 \cdot \omega \cdot t_k) \Rightarrow \frac{1}{2} \cdot \sin(\varphi_k). \end{aligned}$$

Здесь вторые слагаемые в формулах тригонометрических преобразований на выходах **ФНЧ** дают нулевой вклад за счет фильтрации. Математически это означает, что интегралы от тригонометрических функций на интервале дибита дают нули

$$\int_0^{2 \cdot T} \cos(2 \cdot \omega \cdot t_k) dt_k = 0, \quad \int_0^{2 \cdot T} \sin(2 \cdot \omega \cdot t_k) dt_k = 0.$$

В результате сигналы на выходах демодулятора пропорциональны модулирующим импульсам квадратурных каналов  $d_i = \cos(\varphi_k)$ ,  $d_q = \sin(\varphi_k)$ . Блок декодера, считывая квадратурные составляющие демодулятора, восстанавливает исходные переданные значения фазовых состояний и соответствующие  $k$ -биты, то есть исходную последовательность импульсов.

Таким образом, нами дано математическое описание работы варианта модема квадратурной фазовой манипуляции (с фазовым кодером), используемой при реализации (**QPSK, OQPSK, Pi/4\_QPSK, MSK, GMSK**) модуляторов и демодуляторов с кодовым принципом модуляции и демодуляции фазовых состояний несущей частоты.

## 2 КРАТКОЕ ОПИСАНИЕ ПАКЕТА SIMULINK

### 2.1 Общая характеристика пакета Simulink


Пакет **Simulink** разрабатывается компанией **Mathworks** ([www.mathworks.com](http://www.mathworks.com)) и распространяется в составе системы для инженерных и научных расчетов **MatLab**. Пакет основан на графическом интерфейсе и является типичным средством визуально-ориентированного



программирования. Он обладает обширной библиотекой готовых блоков с модифицируемыми параметрами для построения моделей рассматриваемых систем и наглядными средствами визуализации результатов моделирования [3 - 6].

## 2.2 Запуск и работа с пакетом Simulink

Для запуска системы **Simulink** необходимо предварительно выполнить запуск системы **MatLab**. После открытия командного окна системы **MatLab** нужно запустить систему **Simulink**. Это можно сделать одним из трех способов:

- нажать кнопку  (**Simulink**) на панели инструментов системы **MatLab**;
- в строке командного окна **MatLab** напечатать **Simulink** и нажать клавишу **Enter**;
- выполнить опцию **Open** в меню **File** и открыть файл модели (**mdl**-файл).

Последний способ предпочтителен при запуске уже готовой и отлаженной модели, когда требуется лишь провести моделирование и не нужно добавлять новые блоки в модель. При применении двух первых способов открывается окно обозревателя библиотеки блоков (**Simulink Library Browser**).

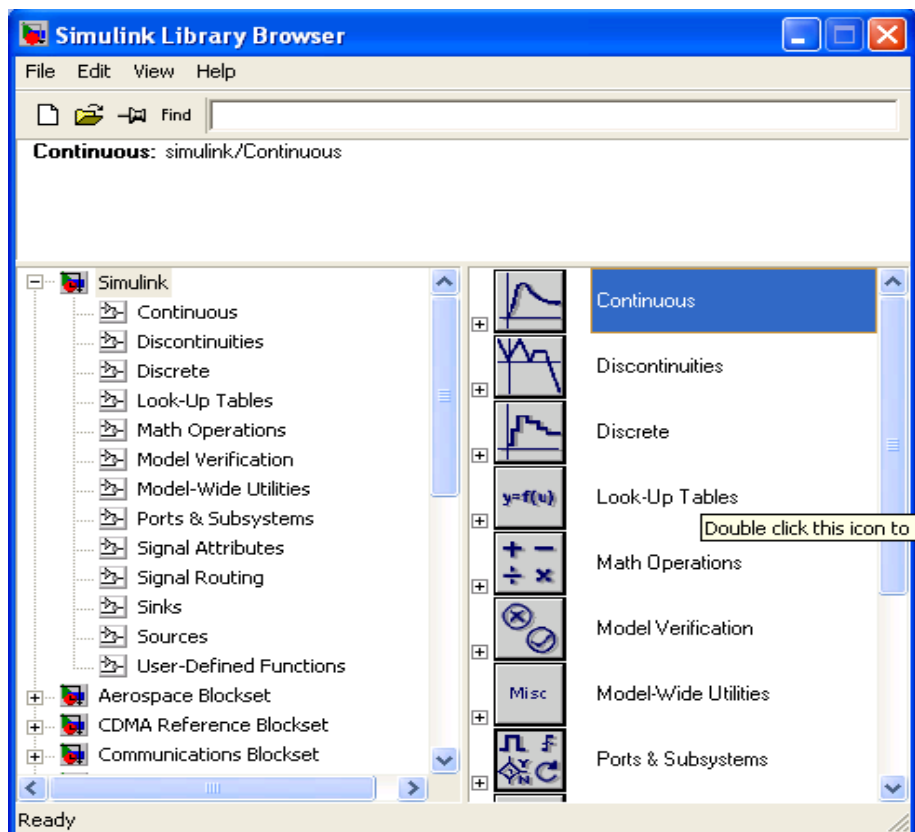


Рисунок 2.1. – Библиотека блоков **Simulink Library Browser**

На рисунке 2.1 выведена библиотека системы **Simulink** (в левой части окна) и показаны ее разделы (в правой части окна). Основная библиотека системы содержит следующие разделы:

- **Continuous** – блоки аналоговых элементов;
- **Discontinuous** – блоки нелинейных элементов;
- **Discrete** – блоки дискретных элементов;
- **Look-Up Tables** – блоки таблиц;
- **Math Operations** – блоки элементов, определяющие математические операции;
- **Model Verification** – блоки проверки свойств сигнала;
- **Model-Wide Utilities** – раздел дополнительных утилит;
- **Port&Subsystems** – порты и подсистемы;
- **Signal Attributes** – блоки задания свойств сигналов;
- **Signal Routing** – блоки маршрутизации сигналов;
- **Sinks** – блоки приема и отображения сигналов;
- **Sources** – блоки источников сигнала;
- **User-Defined Function** – функции, определяемые пользователем.

Список разделов библиотеки представлен в виде дерева, и правила работы с ним являются общими для списков такого вида: пиктограмма свернутого узла дерева содержит символ «+», а пиктограмма развернутого – символ «-».

Для того чтобы развернуть или свернуть узел дерева, достаточно щелкнуть на его пиктограмме левой клавишей мыши (**ЛКМ**). При выборе соответствующего раздела библиотеки его содержимое отображается в правой части окна.

При работе элементы разделов библиотек "**перетаскивают**" в рабочую область удержанием **ЛКМ** на соответствующих изображениях. Для соединения элементов достаточно указать курсором мыши на начало соединения и затем при нажатии левой кнопки мыши протянуть соединение в его конец.

При двойном щелчке **ЛКМ** на выделенном блоке всплывает меню, в котором задаются параметры блоков.

Работа **Simulink** происходит на фоне открытого окна системы **MatLab**, закрытие которого приведет к выходу из **Simulink**.

### 3. ОПИСАНИЕ ЛАБОРАТОРНОГО МАКЕТА

Приведем краткое описание работы модема **Pi/4\_QPSK** манипуляции с кодовым принципом модуляции и демодуляции фазовых состояний несущей частоты на основе **Sim**-модели представленной на рисунке 1.1.

Входной поток данных реализуется блоком источника случайного сигнала с равномерным распределением (**Uniform Random Number**), на выходе которого включен блок определения знака сигнала (**Sign**). Изменяя

параметры блока источника **Initial seed** и **Sample time**, задаем вид случайной последовательности и длительность импульсов.

С выхода схемы формирования входной импульсной последовательности сигнал поступает на кодер-формирователь квадратурных составляющих модулированной несущей фазового манипулятора (блок **Coder\_2\_Pi/4** см. рисунок 3.1).

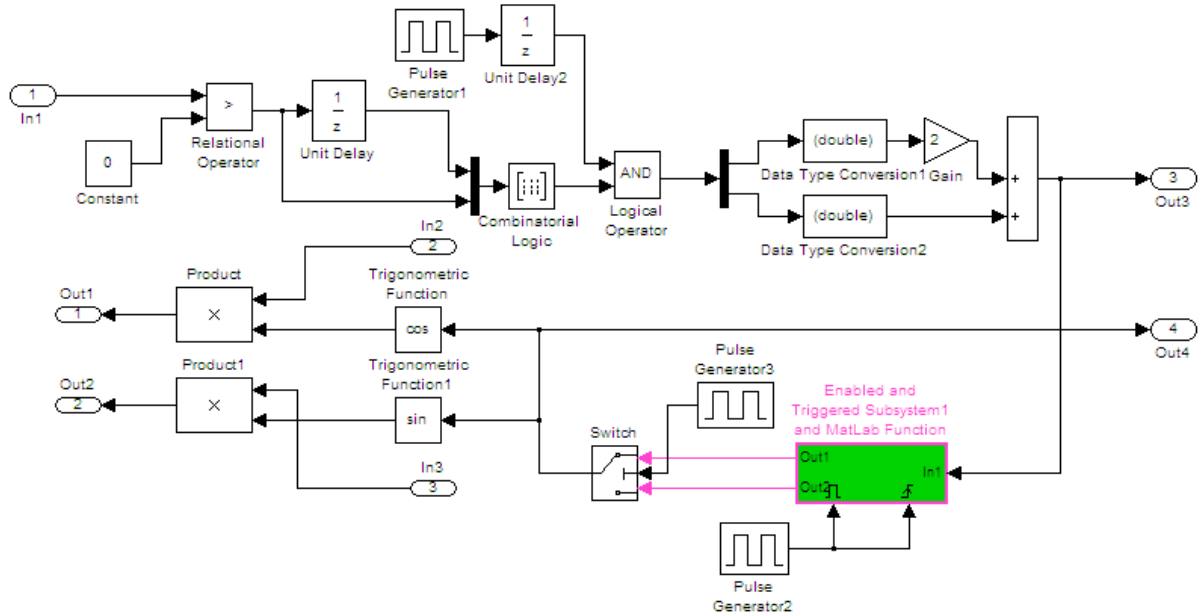


Рисунок 3.1 – Sim-модель составного блока **Coder\_2\_Pi/4** кодера-формирователя квадратурных составляющих модулированной несущей фазового манипулятора

Вход составного блока кодера-формирователя оформлен входным портом **In1**. В начале блока кодера-формирователя включен преобразователь двухполярной импульсной последовательности в однополярную битовую последовательность импульсов, на основе блока операции сравнения (**Relational Operator**) – в данном случае с **Constant=0**.

С выхода кодера-преобразователя однополярная битовая последовательность импульсов подается на преобразователь код-амплитуда, включающий преобразователь битовой последовательности в дибиты и преобразователь дибит в амплитуду. Преобразование битовой последовательности импульсов в векторное представление дибитов, реализованно путем подачи текущего и задержанного на один такт предыдущего импульсов на блок мультиплексора (**Mux**). С выхода мультиплексора векторное представление дибитовых импульсов подается на блок комбинаторной логики (**Combinatorial Logic**), который в каждый шаг по времени преобразует входной векторный сигнал, в соответствии с таблицей истинности, в его логическое (двоичное) представление. Далее, для выборки состояния текущего дибита, двоичное представление дибита подается на один из входов логического блока (**Logical Operator**) с

операцией **И (AND)**. На второй вход логического блока поступают прямоугольные импульсы с опорного генератора (**Pulse Generator1**) с периодом следования равным длительности дибита и скважностью **2**.

С выхода логического блока двоичный логический вектор поступает на преобразователь состояния логического вектора в импульсы пропорциональные по амплитуде десятичному представлению дибита. Входным блоком преобразователя векторного (последовательного) представления логического дибита является демультиплексор (**Demux**), который переводит векторное представление двоичного дибита в параллельное (поразрядное) двоичное представление. После этого каждый разряд двоичного представления дибита подается на блок преобразования типа данных (**Data Type Conversion**), умножается на  $2^k$ , где  $k$  - номер разряда и поступают на вход сумматора (**Sum**).

С выхода преобразователя код-амплитуда импульсы, пропорциональные по амплитуде десятичному представлению дибита и длительностью в один такт, подаются на вход преобразователя-расширителя входных импульсов в *импульсы пропорциональные значению текущего фазового состояния и длительностью равной дибиту*. Преобразователь-расширитель представлен **ET-подсистемой (Enabled and Triggered Subsystem)** с двумя встроенными блоками задания **M-функций (MatLab Fnc)** и (**MatLab Fnc1**). Входной импульс подается на основной вход подсистемы. На **E-** и **T-**входы подсистемы одновременно подается задержанная на один такт импульсная последовательность с опорного генератора (**Pulse Generator2**) с периодом следования равным длительности дибита и скважностью **2**. Встроенные **M-функции Cod\_4\_PSK\_Pi\_4\_1** и **Cod\_4\_PSK\_Pi\_4\_2**, по правилам “нечет” “чет”, сопоставляют амплитудам входного импульса текущее фазовое состояние в соответствии с кодом Грея (см. таблицу). Наличие заданных импульсов на **E-** и **T-**входах обеспечивает расширение формируемого модулирующего импульса до длительности с бита до дибита.

С выходов преобразователя-расширителя, сформированные по правилам “нечет” “чет”, модулирующие импульсы пропорциональные текущему фазовому состоянию  $\varphi_k$  поступают на формирователь “**Нечет-Чет**”, реализованного переключателем (**Switch**), управляемого импульсами опорного генератора (**Pulse Generator3**) с периодом следования равным длительности дибита и скважностью **2**.

С выхода формирователя “**Нечет-Чет**” модулирующие импульсы  $\varphi_k$  через блоки (**Trigonometric Function**) и (**Trigonometric Function1**) в виде импульсов пропорциональных значениям  $\cos(\varphi_k)$  и  $\sin(\varphi_k)$  поступают на первые входы умножителей (**Product** и **Product1**) квадратурного фазового модулятора несущей частоты. На вторые входы умножителей, через порты **In2** и **In3**, поступают сигналы с внешних опорных генераторов несущей частоты, изменяющиеся по законам  $\cos(\omega \cdot t_k)$  и  $\sin(\omega \cdot t_k)$  (блоки **Sine Wave**

и **Sine Wave1**) (см. рисунок 1.1). Квадратурные компоненты модулированной несущей частоты через выходные порты **Out1** и **Out2** поступают на внешний блок сумматора (**Sum**), реализующего заключительный этап модуляции. В результате с выхода сумматора получаем фазоманипулированный (**Pi/4\_QPSK**) сигнал.

Для контроля и визуализации работы составного блока кодера-формирователя оформлены выходные порты **Out3** и **Out4** с выходов преобразователя код-амплитуда и формирователя “Нечет-Чет”.

Внешний узел суммирования квадратурного модулятора (блок **Sum**) одновременно с генератором случайного сигнала с нормальным распределением (**Random Number**), подключенным к третьему входу, имитирует канал передачи сигнала с шумами (см. рисунок 1.1).

Сигнал с имитатора канала передачи подается на вход демодулятора, реализованного двумя умножителями (**Product** и **Product1**), на первые входы которых подается принятый сигнал, а на вторые входы умножителей поступают сигналы с опорных генераторов несущей частоты, изменяющиеся по законам  $\cos(\omega \cdot t_k)$  и  $\sin(\omega \cdot t_k)$  (блоки **Sine Wave2** и **Sine Wave3**). С выходов умножителей сигналы поступают на **ФНЧ** (блоки **Analog Filter Design** и **Analog Filter Design1**), которые отфильтровывают высокочастотные составляющие сигналов умножителей, а низкочастотные составляющие усиливаются блоками **Gain** и **Gain1**. Усиленные низкочастотные составляющие спектра сигналов с **ФНЧ**, пропорциональные принятым квадратурным модулирующим импульсам подаются для предварительного сглаживания на необязательные блоки экстраполяторов нулевого порядка (**Zero-Order Hold1** и **Zero-Order Hold2**).

Далее, сглаженные квадратурные модулирующие импульсы через мультиплексор (блок **Mux**), поступают на блок принятия решений, реализованный программно. С выхода мультиплексора квадратурные составляющие в векторной форме поступают на блок задания **М-функции** (**MatLab Fnc**). Встроенная **М-функция** **Phase\_Pi\_4\_4PSK** реализует вычисление текущих значений фазы по искаженным шумами канала значениям квадратурных составляющих модулирующих импульсов.

Принятые значения фазовых состояний фазоманипулированной несущей подаются на составной блок восстановления исходной последовательности битовых импульсов **Decod\_4psk\_i\_q\_pi\_4**, реализованный программно-аппаратно (см. рисунок 3.2).

Составной блок восстановления исходной последовательности битовых импульсов по принятым фазовым состояниям несущей реализован на основе блока **Е-подсистемы** (**Enabled Subsystem**) со встроенным блоком задания **М-функции** (**MatLab Fnc**) и управляемого опорным генератором прямоугольных импульсов (**Pulse Generator1**) через блок определения момента пересечения порогового значения (**Hit Crossing**). Встроенная **М-функция** **Decod\_4psk\_i\_q\_pi\_4** реализует декодирование фазовых состояний в последовательность допустимых значений нечетных и четных дибит.

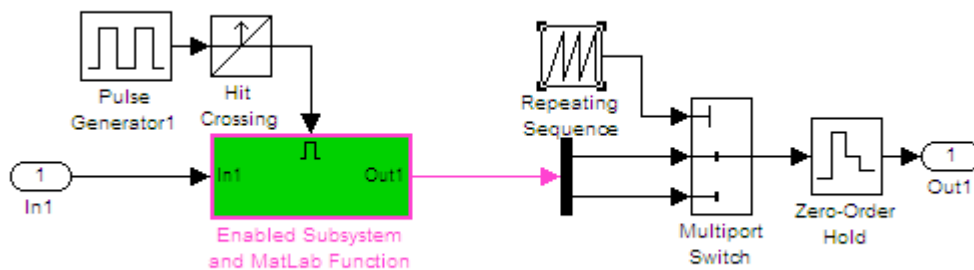


Рисунок 3.2 – Sim-модель составного блока Decod\_4psk\_i\_q\_pi\_4 восстановления исходной последовательности битовых импульсов по принятым фазовым состояниям несущей

Схема восстановления исходной битовой последовательности импульсов по восстановленным дибитам реализована на основе блока демультимплексора (**Demux**) и двухпортового переключателя (**Multiport Switch**) управляемого генератором пилообразного напряжения (**Repeating Sequence**). На выходе схемы восстановления для исключения нежелательных скачков включен блок экстраполятора нулевого порядка (**Zero-Order Hold**). Схемой восстановления дибиты переводятся демультимплексором в параллельное (поразрядное) представление. Значения младшего и старшего разрядов текущего дибита подаются на входные порты двухпортового переключателя, которые управляющим генератором пилообразного напряжения, поочередно на длительность бита, коммутируются на выход. На выходе экстраполятора в результате появляется восстановленная исходная битовая последовательность импульсов.

Схема восстановления исходной последовательности импульсов может быть реализована на основе простого переключателя (**Switch**), управляемого опорным генератором прямоугольных импульсов (**Pulse Generator**) с периодом равным биту и скважностью 2.

Для визуализации работы **Pi/4\_QPSK** модема с кодовым принципом модуляции и демодуляции фазовых состояний приведены осциллограммы блоков осциллографов (**Scope**), подключенных к узловым точкам функциональной модели, и, отображающие основные этапы преобразования сигнала в модуляторе и демодуляторе. Приведены осциллограммы блока отображения спектральной плотности мощности (**Power Spectral Density**) из подраздела **Additional Sinks** раздела **Simulink Extras** библиотеки **Simulink** [5], визуализирующие спектральные характеристики мощности **Pi/4\_QPSK** сигнала. Кроме того, представлены диаграммы рассеяния и переходов фазовых состояний сигнала за счет влияния полосы пропускания и помех тракта передачи, отображаемые блоками **Discrete-Time Scatter Plot Scope** и **Discrete-Time Signal Trajectory Scope** из подраздела **Comm Sinks** раздела **Communication Blockset** библиотеки **Simulink** [6].

На рисунке 3.3 приведена осциллограмма входной импульсной последовательности, сформированной источником случайного сигнала с равномерным распределением и блоком определения знака сигнала.

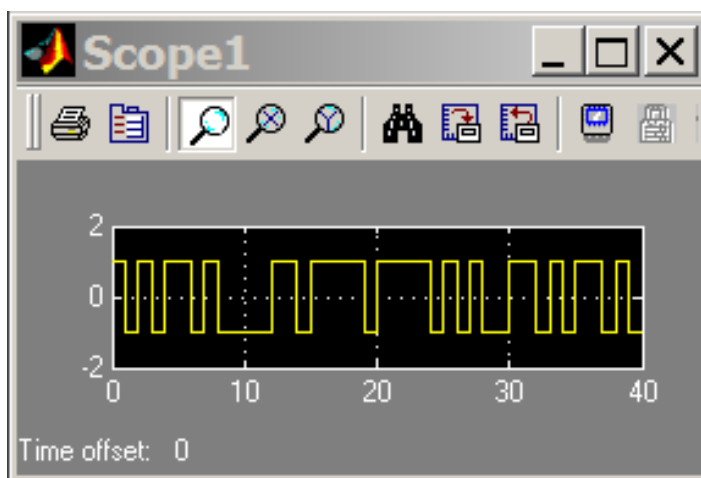


Рисунок 3.3 – Осциллограмма входной импульсной последовательности

На рисунке 3.4 приведены осциллограммы квадратурных составляющих манипулированной несущей частоты модулятора  $\text{Pi}/4$ \_QPSK модулятора, сформированные кодером-формирователем (блок **Coder\_2\_Pi/4**).

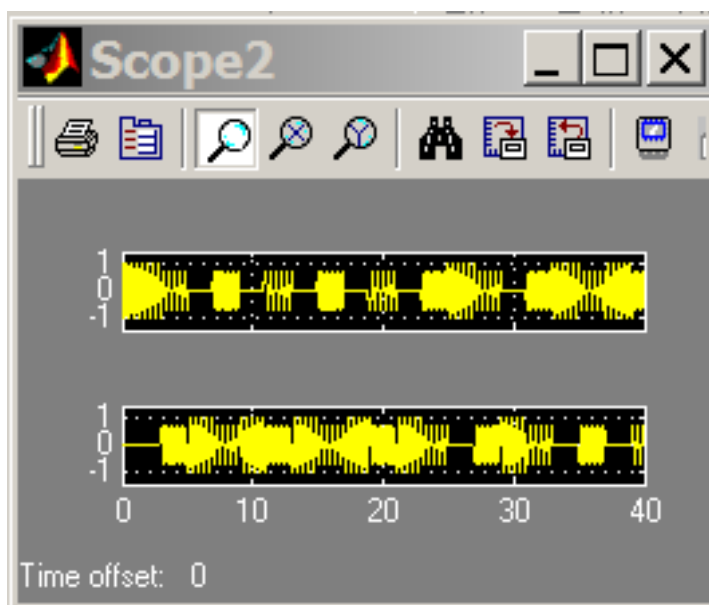


Рисунок 3.4 – Осциллограммы квадратурных составляющих манипулированной несущей частоты модулятора  $\text{Pi}/4$ \_QPSK

На рисунке 3.5 приведены осциллограммы выходов преобразователя код-амплитуда и формирователя “Нечет-Чет” составного блока кодера-формирователя.

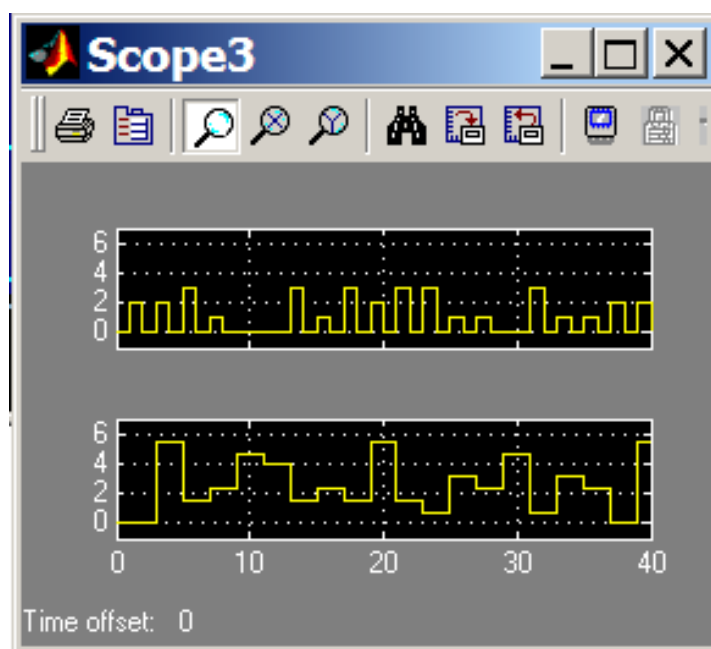


Рисунок 3.5 – Осциллограммы выхода преобразователя код-амплитуда в виде импульсов битовой длительности и модулирующие импульсы фазовых состояний длительностью определяемой дибитом

На рисунке 3.6 приведена осциллограмма случайной импульсной последовательности имитатора шумов канала передачи блока **Random Number** (источника случайного сигнала с нормальным распределением уровня).

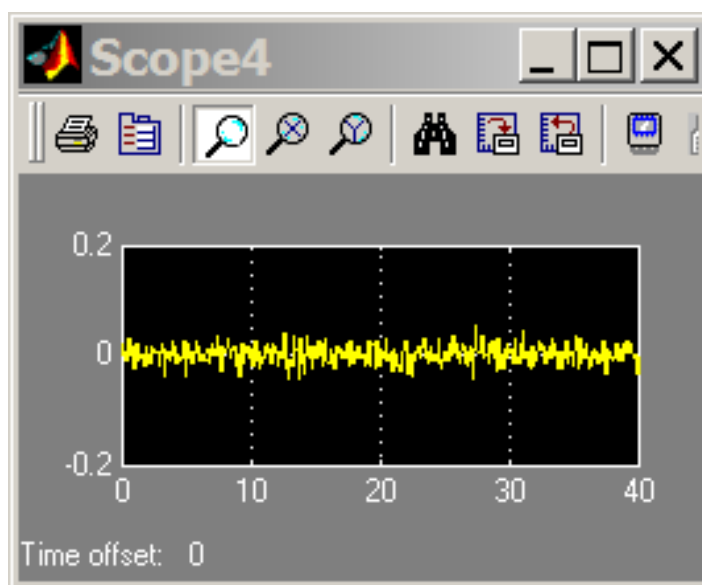


Рисунок 3.6 – Осциллограмма случайной импульсной последовательности имитатора шумов канала передачи (блока Random Number)

На рисунке 3.7. приведена осциллограмма принятого фазоманипулированного **Pi/4\_QPSK** сигнала на несущей частоте.



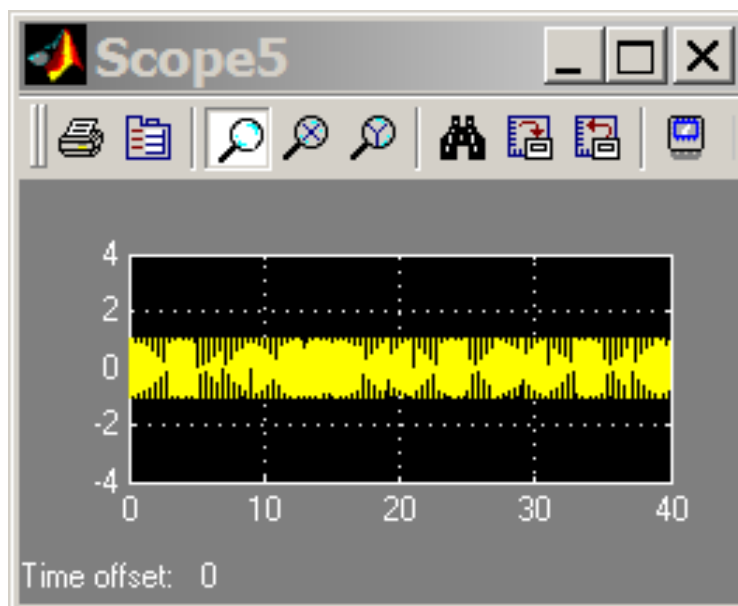


Рисунок 3.7 – Осциллограмма принятого фазоманипулированного  $\text{Pi}/4\text{-QPSK}$  сигнала на несущей частоте

На рисунке 3.8 приведены осциллограммы принятых квадратурных составляющих несущей  $\text{Pi}/4\text{-QPSK}$  модулированного сигнала на входах ФНЧ демодулятора.

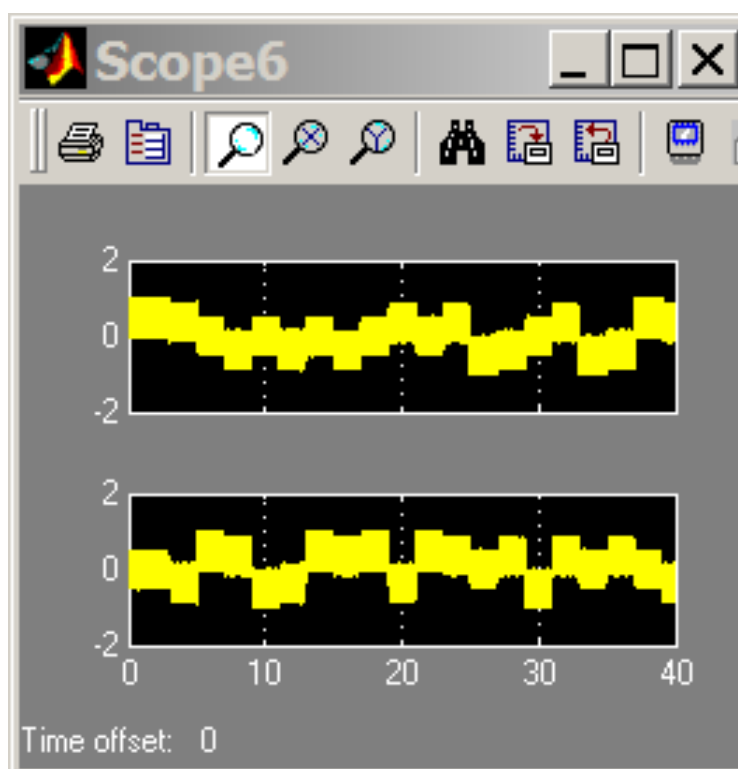


Рисунок 3.8 – Осциллограммы квадратурных составляющих несущей сигнала на входах ФНЧ демодулятора.

На рисунке 3.9 приведены осциллограммы квадратурных составляющих демодулированных импульсов на выходах ФНЧ демодулятора, соответствующих принятым модулирующим импульсам квадратурных каналов модулятора  $\text{Pi}/4\_QPSK$ .

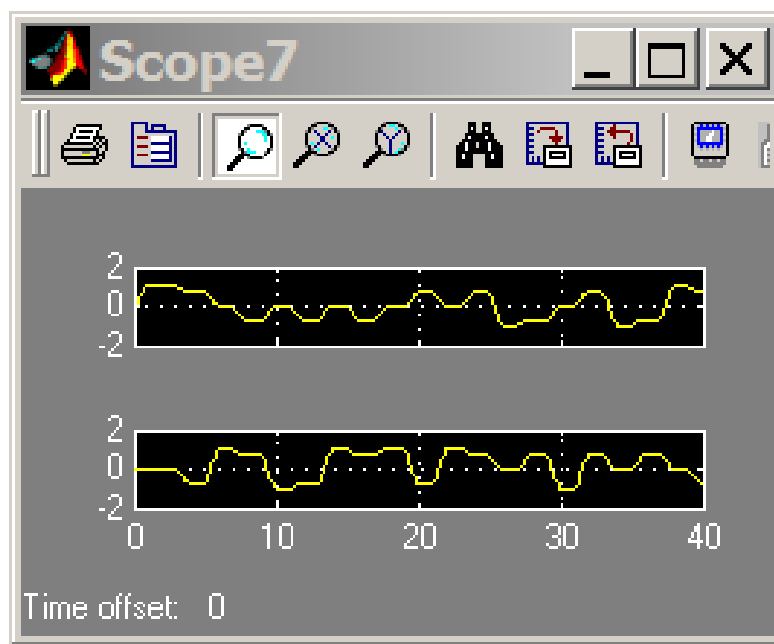


Рисунок 3.9 – Осциллограммы квадратурных составляющих демодулированных импульсов на выходах ФНЧ демодулятора.

На рисунке 3.10 приведены осциллограммы принятых, демодулированных и восстановленных схемой принятия решений модулирующих импульсов отражающих фазовые состояния несущей сигнала.

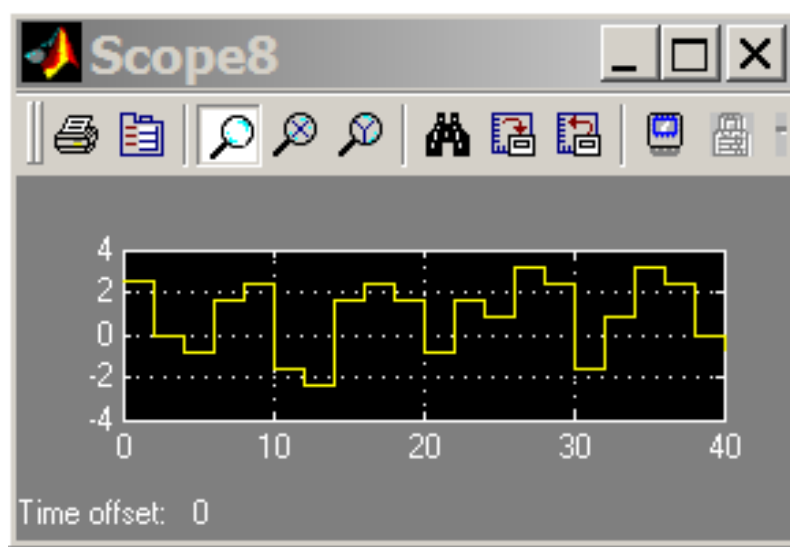


Рисунок 3.10 – Осциллограммы принятых, демодулированных и восстановленных схемой принятия решений фазовых состояний несущей сигнала

На рисунке 3.11 приведена осциллограмма принятого сигнала на выходе схемы восстановления исходной битовой последовательности импульсов по принятым фазовым состояниям несущей сигнала.

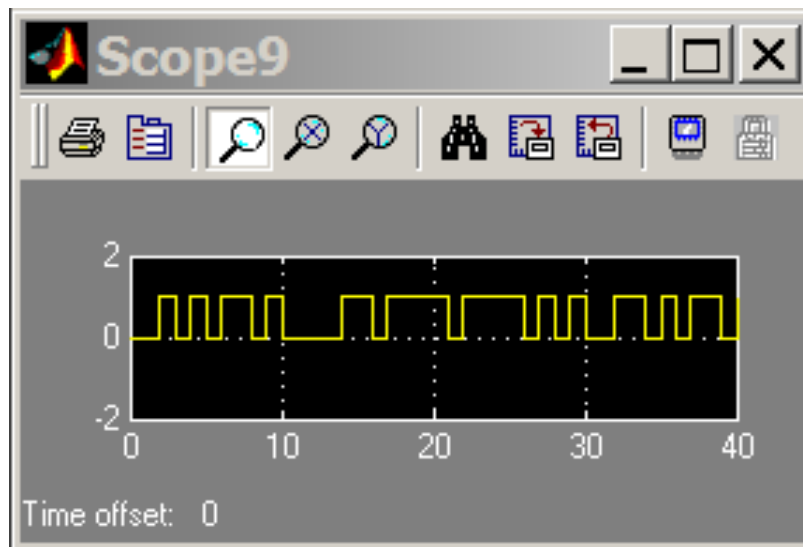


Рисунок 3.11 – Осциллограммы принятого сигнала на выходе схемы восстановления исходной битовой последовательности импульсов

На рисунке 3.12 приведены осциллограммы блока отображения спектральной плотности мощности (**Power Spectral Density**) из подраздела **Additional** раздела **Sinks Simulink Extras** библиотеки **Simulink** [5]. Окно анализатора спектра содержит три графика: - временной зависимости входного сигнала; - амплитудно-частотной зависимости спектральной плотности мощности входного сигнала; фазо-частотной зависимости спектра мощности входного сигнала. Осциллограммы блока **Power Spectral Density** позволяют оценить эффективность использования полосы пропускания исследуемого типа фазовой манипуляции, в данном случае **Pi/4\_QPSK** модуляции.

Как известно ширина полосы пропускания необходимая для обеспечения цифровой связи определяется длиной битовой посылки. Чем короче битовый импульс, тем шире полоса частот, занимаемая его спектром, в частности, основным лепестком. Кроме того, с точки зрения интерференционных помех важна скорость спада уровня боковых лепестков спектральной характеристики. Ширина основного лепестка и скорость спада уровня боковых лепестков существенно зависят от формы импульса. Наиболее узкий основной лепесток имеет прямоугольная форма импульса, однако при этом скорость затухания уровня боковых лепестков мала и пропорциональна  $1/f$  (для спектральной плотности  $1/f^2$ ). При косинусоидальной форме импульса скорость спада уровня боковых лепестков пропорциональна  $1/f^2$  (для спектральной плотности  $1/f^4$ ), однако ширина бокового лепестка примерно в **1.5** раза шире, чем у

прямоугольного импульса. Для обеспечения компромисса между шириной бокового лепестка и скоростью спада уровня боковых лепестков косинусоидальные импульсы нормируют гауссовским фильтром, что используется в **GMSK** модуляции.

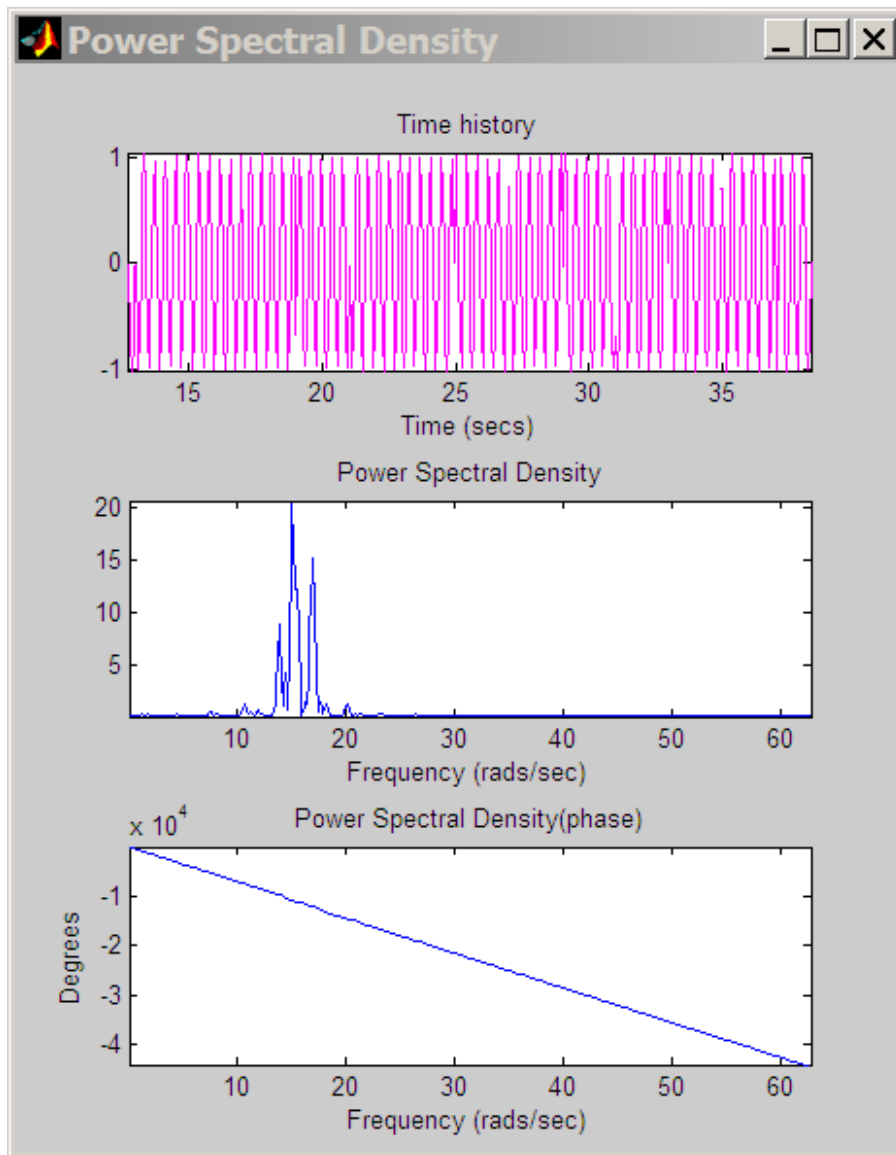


Рисунок 3.12 – Осциллограммы блока отображения спектральной плотности мощности (Power Spectral Density)

Основным достоинством квадратурной фазовой манипуляции является *удвоение* длительности модулирующих импульсов квадратурных каналов, то есть снижение требуемой скорости передачи битов *вдвое*, а, следовательно, сокращение требуемой полосы пропускания канала *вдвое*.

При  **$\pi/4$ \_QPSK** модуляции вследствие чередования нечетных и четных дибит и их отдельном кодировании максимальные фазовые переходы оказываются равными  $\pm 3 \cdot \pi/4$ . В **QPSK** и **OQPSK** модуляциях максимальные значения скачков фазы составляют соответственно  $\pm \pi$  и  $\pm \pi/2$ .

Таким образом, при  $\text{Pi}/4\_QPSK$  модуляции кратковременный провал уровня сигнала в узкополосном канале оказывается меньшим, чем при  $QPSK$  модуляции, но большим, чем при  $OQPSK$  модуляции, что частично способствует повышению компактности спектра сигнала, увеличению скорости убывания внеполосного излучения и возможности повышения КПД используемых усилителей мощности.

На рисунках 3.13 - 3.14 приведены диаграммы рассеяния и перехода фазовых состояний  $\text{Pi}/4\_QPSK$  сигнала за счет влияния полосы пропускания и помех тракта передачи, отображаемые блоками **Discrete-Time Scatter Plot Scope** и **Discrete-Time Signal Trajectory Scope** из подраздела **Comm Sinks** раздела **Communication Blockset** библиотеки **Simulink** [6].

Диаграммы блоков **Discrete-Time Scatter Plot Scope** и **Discrete-Time Signal Trajectory Scope** позволяют оценить влияние метода модуляции, полосы пропускания и шумов канала передачи на рассеяние и переходы фазовых состояний принятого сигнала.

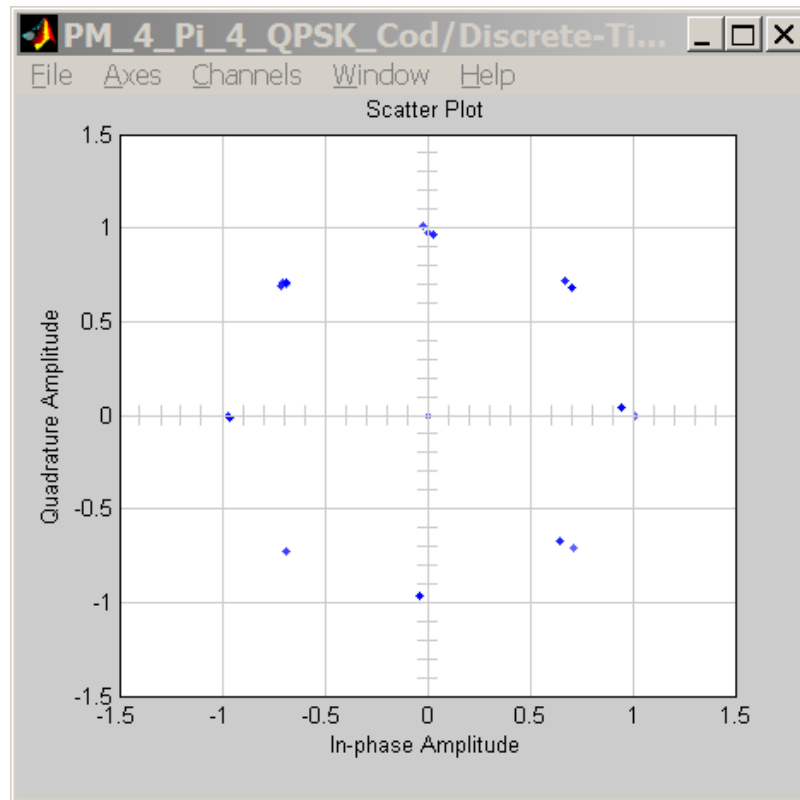


Рисунок 3.13 – Диаграмма рассеяния фазовых состояний сигнала за счет влияния метода модуляции, полосы пропускания и помех тракта передачи, отображаемая блоком **Discrete-Time Scatter Plot Scope**

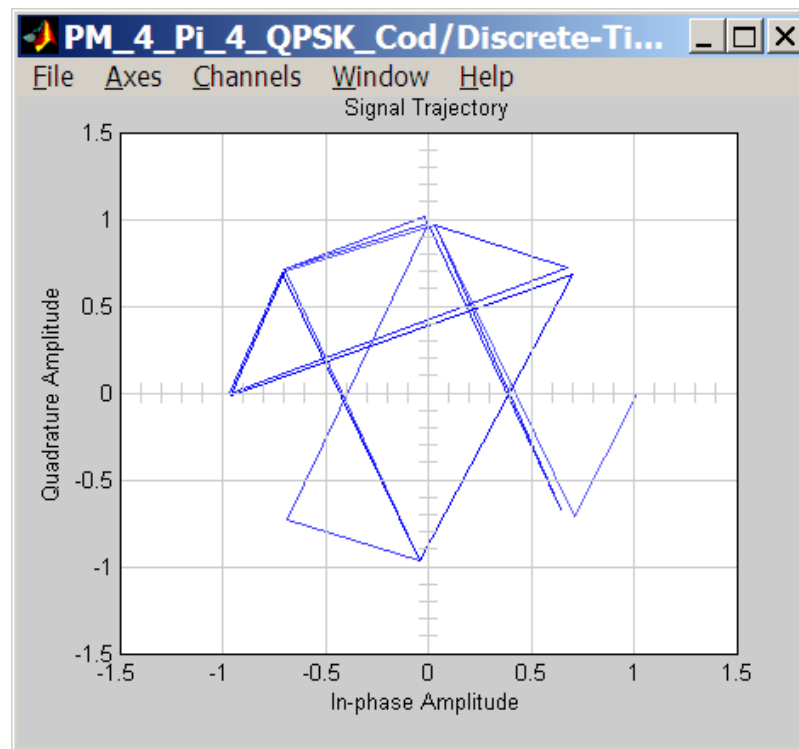


Рисунок 3.14 – Диаграмма траекторий фазовых состояний сигнала за счет влияния метода модуляции, полосы пропускания и помех тракта передачи, отображаемая блоком Discrete-Time Signal Trajectory Scope

#### 4. ОПИСАНИЕ ИСПОЛЬЗУЕМЫХ БЛОКОВ БИБЛИОТЕКИ SIMULINK

Ниже описаны основные блоки базовых разделов библиотеки **Simulink** [3, 4, 5], используемые в функциональной схеме классического варианта модема квадратурной фазовой манипуляции, с естественным принципом модуляции и демодуляции фазовых состояний несущей частоты.



**Uniform Random Number** – источник случайного дискретного сигнала с нормальным распределением. *Назначение:* формирование случайного сигнала с равномерным распределением уровня. *Параметры блока:* **Minimum** – минимальный уровень сигнала; **Maximum** – максимальный уровень сигнала; **Initial seed** – начальное значение генератора случайного сигнала; **Sample time** – такт дискретности.



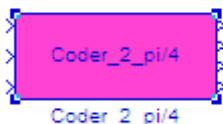
**Sign** – блок определения знака сигнала. *Назначение:* определяет знак входного сигнала, при этом, если  $x$  - входной сигнал, то сигнал на выходе определяется выражением

$$\text{sign}(x) = \begin{cases} -1 & \text{при } x < 0, \\ 0 & \text{при } x = 0, \\ 1 & \text{при } x > 0. \end{cases}$$

*Параметры блока:* флажок - **Enable zero crossing detection** позволяет фиксировать прохождение сигнала через нулевой уровень.



**Scope** – блок осциллографа. *Назначение:* построение графиков исследуемых сигналов как функций времени. Открытие окна осциллографа производится двойным щелчком ЛКМ на пиктограмме блока. В случае векторного сигнала каждая компонента вектора отображается отдельным цветом. Настройка окна осциллографа выполняется с помощью панелей инструментов, позволяющих: осуществить печать содержимого окна осциллографа; установить *параметры*, в частности, **Number of axes** - число входов осциллографа, **Time range** – отображаемый временной интервал и другие; изменить масштабы графиков; установить и сохранить настройки; перевести в плавающий режим и так далее.



**Coder\_2\_pi/4** – составной блок фазового кодера и **Pi/4\_QPSK** модулятора из пользовательской библиотеки **Library Communication**. *Назначение:* кодирование нечетных и четных дибит входной импульсной последовательности модулирующими импульсами фазовых состояний и формирование квадратурных составляющих модулированной несущей фазового манипулятора. Входные и выходные порты, служащие для соединения с внешними блоками, пронумерованы сверху – вниз. Первый входной порт предназначен для ввода двухполярной информационной последовательности импульсов. Второй и третий входные порты служат для подключения колебаний несущей частоты изменяющихся, соответственно, по законам  $\cos(\omega \cdot t_k)$  и  $\sin(\omega \cdot t_k)$ . Первый и второй выходные порты служат выводами квадратурных составляющих модулированной несущей для последующего суммирования с целью завершения этапа фазовой манипуляции. Третий и четвертый выходные порты служат для подключения блоков осциллографа **Scope** с целью визуализации работы внутренних блоков преобразователя код-амплитуда и формирователя “Нечет-Чет”.



**Inport** – блок входного порта. *Назначение:* создает входной порт для подсистемы или выполняет считывание из рабочей области системы **MatLab** в модель. *Параметры блока:* **Port number**- номер порта; **Port dimensions**- размерность входного порта, если этот параметр равен **-1**, то размерность входного сигнала определяются автоматически; **Sample time**-

такт дискретности (при задании значения параметра равного **-1** такт дискретности наследуется от предшествующего блока).; **Show additional parameters** – показать дополнительные параметры, в нашем случае не используется.

Использование блока **Inport** в подсистемах. Блоки **Inport** подсистемы образуют ее входы. Сигнал, подаваемый на входной порт **Inport**, передается внутрь подсистемы, название входного порта на изображении подсистемы показывается соответствующей меткой.

При создании подсистем и добавлении блока **Inport** в подсистему пакет **Simulink** использует следующие правила:

При создании подсистемы с помощью команды **Edit/Create subsystem** входные порты создаются автоматически и нумеруются, начиная с единицы.

При добавлении в подсистему нового блока **Inport** ему присваивается следующий по порядку номер.

Удаление блока **Inport** приводит к перенумерации остальных портов, оставляя ее непрерывной.

Если последовательность номеров портов окажется разрывной, то при выполнении моделирования **Simulink** выдает сообщение об ошибке с остановкой вычислений. В этом случае следует вручную переименовать порты таким образом, чтобы последовательная нумерация не нарушалась.

Использование блока **Inport** в моделях верхнего уровня. Входной порт в системе верхнего уровня используется для передачи сигнала из рабочей области **MatLab** в модель. Для передачи сигнала из рабочего пространства **MatLab** требуется не только установить в модели входные порты, но и выполнить установку параметров ввода на вкладке **Workspase I/O** окна диалога **Simulation parameters** (должен быть установлен флажок для параметра **Inport** и задано имя переменной, содержащей входные данные). Тип вводимых данных: **Array**- массив; **Structure**- структура; **Structure with time**- структура с полем “Время” (задается на той же вкладке).



**Outport**- блок выходного порта. *Назначение:* создает выходной порт для подсистемы или для модели верхнего уровня иерархии для передачи сигнала в рабочую область системы **MatLab**. *Параметры блока:* **Port number**- номер порта; **Output when disabled**- вид сигнала на выходе подсистемы, в случае если подсистема выключена. Используется для подсистем управляемых внешним сигналом и может принимать следующие значения из списка: **held**- выходной сигнал подсистемы равен последнему рассчитанному значению, **reset**- выходной сигнал подсистемы равен значению определяемую параметром **Initial output**; **Initial output**- начальное значение сигнала на выходе подсистемы до начала ее работы или система находится в отключенном состоянии (используется для подсистем управляемых внешним сигналом).

Использование блока **Outport** в подсистемах. Блоки **Outport** подсистемы образуют ее выходы. Сигнал, подаваемый на выходной порт



**Output**, передается в модель или подсистему верхнего уровня. Название выходного порта на изображении подсистемы показывается соответствующей меткой.

При создании подсистем и добавлении блока **Output** в подсистему пакет **Simulink** использует следующие правила:

При создании подсистемы с помощью команды **Edit/Create subsystem** выходные порты создаются автоматически и нумеруются, начиная с единицы.

При добавлении в подсистему нового блока **Output** ему присваивается следующий по порядку номер.

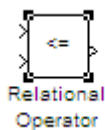
Удаление блока **Output** приводит к перенумерации остальных портов, оставляя ее непрерывной.

Если последовательность номеров портов окажется разрывной, то при выполнении моделирования **Simulink** выдает сообщение об ошибке с остановкой вычислений. В этом случае следует вручную переименовать порты таким образом, чтобы последовательная нумерация не нарушалась.

В том случае, если подсистема управляется извне, для выходных портов можно задать вид выходного сигнала для тех интервалов времени, когда подсистема заблокирована.

Использование блока **Output** в моделях верхнего уровня. Входной порт в системе верхнего уровня используется для передачи сигнала в рабочую область **MatLab** или для обеспечения связи функций анализа с выходами модели. Для передачи сигнала в рабочее пространство **MatLab** требуется не только установить в модели выходные порты, но и выполнить установку параметров ввода на вкладке **Workspace I/O** окна диалога **Simulation parameters** (должен быть установлен флажок для параметра **Output** и задано имя переменной для сохранения данных). Тип сохраняемых данных: **Array**- массив; **Structure**- структура; **Structure with time**- структура с полем “Время” (задается на той же вкладке).

Блок **Output** может использоваться также для связи модели с М-функциями анализа, например: **linmod** или **trim**.



**Relational Operator** – блок выполнения операций отношения. *Назначение:* сравнение текущих значений входных сигналов поступающих на входы. *Параметры блока:* **Relational Operator** – тип операции отношения выбираемый из списка:

== - тождественно равно;

≠ - не равно;

< - меньше;

≤ - меньше или равно;

≥ - больше или равно;

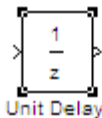
> - больше.

Флажок - **Show additional parameters** – показать дополнительные параметры – в нашем случае не используется.



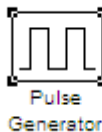
**Constant** – источник постоянного сигнала.

*Назначение:* задает сигнал постоянного уровня. *Параметры блока:* **Constant value** – постоянная величина, значение которой может быть задано действительным или комплексным числом, вычисляемым выражением, вектором или массивом; флажок **Interpret vector parameters as 1 - D** – интерпретировать вектор как массив скаляров; флажок **Show additional parameters** – показать дополнительные параметры, в нашем случае не используется.



**Unit delay** – блок дискретной задержки. *Назначение:*

выполняет задержку дискретного сигнала на заданный такт дискретности. *Параметры блока:* **Initial conditions** – начальное значение выходного сигнала; **Sample time** – такт дискретности (при задании значения параметра равного **-1** такт дискретности наследуется от предшествующего блока).



**Pulse generator** – блок источника импульсного сигнала.

*Назначение:* формирование сигнала в форме прямоугольных импульсов. *Параметры блока:* **Pulse Type** – способ формирования сигнала, может принимать два значения: **Time-based** – по текущему времени; **Sample-based** – по величине такта дискретности и количеству шагов моделирования. Вид окна параметров зависит от выбранного способа формирования сигнала. **Amplitude** – амплитуда; **Period** – период, задается в секундах при способе **Time-based** или количеством тактов при способе **Sample-based**; **Pulse width** – ширина импульса, задается в процентах от периода при способе **Time-based** или количеством тактов при способе **Sample-based**; **Phase delay** – фазовая задержка, задается в секундах при способе **Time-based** или количеством тактов при способе **Sample-based**; **Sample time** – такт дискретности; флажок **Interpret vector parameters as 1 - D** – интерпретировать вектор как массив скаляров.



**Mux** – блок мультиплексора. *Назначение:*

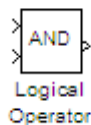
объединяет входные сигналы в вектор. *Параметры блока:* **Number of Inputs** – количество входов; **Display option** – способ отображения, выбирается из списка: **bar** – вертикальный узкий прямоугольник черного цвета; **signals** – прямоугольник с белым фоном и отображением меток входных сигналов;

**none** – прямоугольник с белым фоном без отображения меток входных сигналов.



**Combinatorial Logic** – блок комбинаторной логики.

*Назначение:* преобразует входной векторный сигнал в соответствии с таблицей истинности. Таблица истинности представляет собой список возможных выходных значений блока. Каждому состоянию входного векторного сигнала соответствует определенное логическое состояние выходного сигнала. *Параметры блока:* **Truth table** – таблица истинности. Так для возможных значений входного вектора, соответствующего дибитам, таблица истинности имеет вид [00; 01; 10; 11].



**Logical Operation-** блок выполнения логических операций.

*Назначение:* реализует одну из базовых логических операций. *Параметры блока:* **Operator-** вид реализуемой логической операции – выбирается из списка:

**AND-** логическое умножение (операция логическое **И**), **OR-** логическое сложение (операция логическое **ИЛИ**), **NAND-** операция **И-НЕ**, **NOR-** операция **ИЛИ-НЕ**, **XOR-** операция сложения **по модулю 2** (операция **ИСКЛЮЧАЮЩЕЕ ИЛИ**), **NOT-** логическое отрицание (логическое **НЕ**); **Number of input ports-** количество входных портов; Флажок **Show additional parameters** – показать дополнительные параметры (в нашем случае не используется); Флажок **Require all inputs to have same data type-** установить одинаковый тип входных данных; **Output data type mode-** выбор типа выходных данных из списка: **Boolean** (двоичный), **Logical** (логический), **Specify via dialog** (задаваемый дополнительным списком). В последнем случае появится окно списка **Output data type-** тип выходных данных.

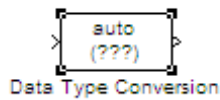
Входные сигналы могут быть как действительного, так и логического типа (**Boolean**). Выходным сигналом блока является **1**, если результат вычисления логической операции есть **ИСТИНА**, и **0**, если результат – **ЛОЖЬ**.



**Demux** – блок демультиплексора. *Назначение:*

разделение входного векторного сигнала на составляющие (последовательного представления в параллельное). *Параметры блока:* **Number of output** – количество выходов; **Display option** – способ отображения выбирается из списка: **bar** – вертикальный узкий прямоугольник черного цвета; **none** – прямоугольник с белым фоном без отображения меток входных сигналов. Флажок **Bus Selection Mode** – режим

разделения векторных сигналов в шине, используется для разделения сигналов, объединенных в шину.



**Data Type Conversion** – блок преобразователя типа сигнала. *Назначение:* преобразует тип входного сигнала. *Параметры блока:* **Data type** – тип данных выходного сигнала, может принимать значения из списка: **auto; double; single; int8; int16; int32/ uint8; uint16; uint32; Boolean**. **Saturate on integer** – подавлять переполнение целого. При установленном флажке ограничение сигналов целого типа выполняется корректно. Значение **auto** параметра **Data type** используется при наследовании выходным сигналом типа входного сигнала. Входной сигнал блока может быть действительным или комплексным, в случае последнего выходной сигнал также комплексный. Входные сигналы могут быть в скалярной, векторной и матричной формах.

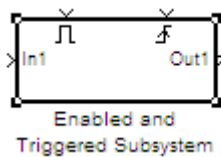


**Gain** – блок усилителя. *Назначение:* блок **Gain** умножает входной сигнал на постоянный коэффициент; *Параметры блока:* **Multiplication** – способ выполнения операции, значение параметра выбирается из списка: **Element-wise  $K*u$**  – поэлементный; **Matrix  $K*u$**  – матричный, коэффициент усиления является левосторонним оператором; **Matrix  $u*K$**  – матричный, коэффициент усиления является правосторонним оператором; **Matrix  $K*u$  ( $u$ -вектор)** – векторный, коэффициент усиления является левосторонним оператором. Флажок **Show additional parameters** – показать дополнительные параметры, при выставленном флажке отображаются окна списков **Parameter data type mode, Output data type mode**. **Saturate on integer** – подавлять переполнение целого. При установленном флажке ограничение сигналов целого типа выполняется корректно.

Блоки **Gain** и **Matrix Gain** по сути есть один и тот же блок, но с разными начальными установками параметра **Multiplication**. Для векторного входного сигнала  $u$  и параметра **Gain** в виде вектора-строки  $K$ , при способе выполнения операции **Element-wise  $K*u$**  получаем вектор покомпонентного умножения векторов  $K*u$ , а при способе выполнения операции **Matrix  $K*u$**  получаем скалярное произведение векторов  $K*u$ . Параметр блока **Gain** может быть положительным или отрицательным, как больше, так и меньше единицы. Коэффициент усиления можно задавать в виде скаляра, матрицы, вектора или вычисляемого выражения. Если входной сигнал действительного типа, а коэффициент усиления комплексный, то выходной сигнал будет комплексным. В случае отличия типа входного сигнала от типа коэффициента усиления **Simulink** пытается выполнить приведение типа усиления к типу входного сигнала и выдает сообщение об ошибке, если такое приведение невозможно.



**Sum** – блок сумматора. *Назначение:* вычисление алгебраической суммы текущих значений входных сигналов. *Параметры блока:* **Icon shape** – форма блока, выбирается из списка: **round** – круг; **rectangular** – прямоугольник. **List of sign** – список знаков из набора: + - плюс; - - минус, | - делитель. Флажок **Show additional parameters** – показать дополнительные параметры, при выставленном флажке отображаются окна списка **Output data type mode**, в нашем случае не используется. Количество входов и соответствующие им операции определяются списком знаков **List of sign**. При этом метки входов обозначаются соответствующими знаками. В списке **List of sign** можно также указать число входов, при этом все входы будут суммирующими.



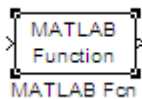
**Enabled and Triggered Subsystem** – блок **ET**-подсистемы. Исходный шаблон блока содержит в своем составе блоки: **Triggered** – триггер; **Enable** – блок управления; входной и выходной порты и линию связи между ними. После того как блок подсистемы скопирован из библиотеки в окно модели, он становится доступным для редактирования. Блок может содержать в своем составе другие блоки и подсистемы. Число входных и выходных портов определяется конфигурацией подсистемы и устанавливается пользователем. *Назначение:* **ET**-подсистема включается фронтом сигнала, поступающего на **T**-вход системы при наличии положительного сигнала на **E**-входе подсистемы. Так же как и **Triggered Subsystem**, эта подсистема выполняет вычисления только на том шаге моделирования, где произошло изменение управляющего сигнала на **T**-входе. Параметр **States when enabling** блока **Enable** не оказывает влияния на работу **ET**-подсистемы. Оба управляющих сигнала могут быть векторными.

Блок **Triggered** – триггер. *Параметры блока:* **Show port labels** – показать метки портов; **Treat as atomic unit** – считать подсистему неделимой; **Read/Write Permissions** – разрешить чтение и запись, допустимы 3 опции: **ReadWrite** – чтение и запись; **Read Only** – только чтение; **NoReadOrWrite** – ни чтения ни записи. **Name of error callback function** – имя функции ответного вызова. Блок **Triggered: Trigger type** – тип триггера, выбирается из списка: **rising** – активизация подсистемы положительным фронтом; **falling** – активизация подсистемы отрицательным фронтом; **either** – активизация подсистемы как положительным, так и отрицательным фронтом; **function call** – активизация подсистемы определяется логикой работы вызываемой **S**-функции. Флажок **Show output port** – показать выходной порт, при установленном флажке на пиктограмме блока **Trigger** появляется дополнительный выходной порт, сигнал с которого может быть использован для управления блоками внутри подсистемы. Флажок **Enable zero crossing**

**detection** – предписывает фиксировать прохождение сигнала через нулевой уровень.

Блок **Enable** – управляющий вход. *Параметры блока:* **States when enabling** – состояние при запуске, задает состояние подсистемы при каждом запуске и выбирается из списка: **held** – использовать предыдущее активное состояние; **reset** – использовать начальное (исходное состояние). **Show output port** – показать выходной порт, при установленном флажке на пиктограмме блока **Enable** появляется дополнительный выходной порт, сигнал с которого может быть использован для управления блоками внутри подсистемы. Флажок **Enable zero crossing detection** – предписывает фиксировать прохождение сигнала через нулевой уровень.

Пользователь может автоматически создать блок **Subsystem**. Для этого необходимо выделить с помощью мыши нужный фрагмент модели и выполнить команду **Create Subsystem** из меню **Edit** в окне модели. Выделенный фрагмент будет помещен в подсистему, а входы и выходы будут снабжены соответствующими портами. В дальнейшем, если это необходимо, можно объявить этот блок неделимым, изменив параметры, или сделать условно управляемым, добавив управляющие входы. Отменить группировку блоков в подсистему можно командой **Undo**.



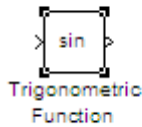
**MatLab Fnc** – блок задания М-функции.

*Назначение:* задание выражения (алгоритма) в стиле языка программирования системы **MatLab** или имени М-функции (М-файла). Входные и выходные параметры М-функции могут быть векторными. Входной сигнал в выражении обозначается **u**, если он скаляр, для вектора необходимо указывать индекс компонента в круглых скобках. Необходимо иметь в виду, что М-функция или выражение выполняются на каждом шаге модельного времени, передаются на выход и недоступны на следующем шаге времени. Имя М-функции или М-файла не должно совпадать с именем модели (**mdl**-файлом). *Параметры блока:* **MatLab function** – имя М-функции без параметров или выражение на языке **MatLab**; **Output dimension** – размерность выходного сигнала, значение равно (-1) предписывает автоматическое определение размерности; **Output signal type** – тип выходного сигнала, выбирается из списка: **real** – действительный сигнал; **complex** – комплексный сигнал; **auto** – автоматическое определение типа сигнала. Флажок **Collapse 2 – D result to 1 – D** – разрешает преобразование двумерного выходного сигнала в одномерный сигнал.

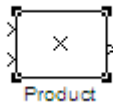


**Switch** – блок переключателя. *Назначение:* переключение входных сигналов по сигналу управления. *Параметры блока:* **Criteria for passing first input** – условие прохождения сигнала с первого входа, значение выбирается из списка: **u2>=Threshold** – сигнал управления

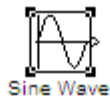
больше или равен пороговому значению;  $u_2 > \text{Threshold}$  – сигнал управления больше порогового значения;  $u_2 \neq \text{Threshold}$  – сигнал управления не равен пороговому значению. **Threshold** – порог; флажок **Show additional parameters** – показать дополнительные параметры. При выставленном флажке отображаются дополнительные окна списков, в нашем случае флажок не используется.



**Trigonometric Function** – тригонометрическая функция. *Назначение:* выдает выбранную тригонометрическую функцию. *Параметры блока:* **Function** – вид вычисляемой функции. Вид функции выбирается из списка: **sin, cos, tan, asin, acos, atan, atan2, sinh, cosh, tanh**. **Output signal type** – тип выходного сигнала. Тип выходного сигнала выбирается из списка: auto – автоматическое определение типа, real – действительный сигнал, complex – комплексный сигнал. При векторном или матричном входном сигнале блок выполняет поэлементное вычисление заданной функции.



**Product** – блок умножения и деления. *Назначение:* вычисление произведения текущих значений сигналов. *Параметры блока:* **Number of inputs** – количество входов, может задаваться как число или как список знаков. В списке знаков можно использовать знаки: \* – умножить и / – разделить. **Multiplication** – способ выполнения операции, может принимать значения из списка: **Element-wise** – поэлементный; **Matrix** – матричный. Флажок **Show additional parameters** – показать дополнительные параметры. При выставленном флажке отображается окно списка **Output data type mode**, в нашем случае флажок не используется.



**Sine Wave** – блок источника синусоидального сигнала. *Назначение:* формирование синусоидального сигнала с заданной частотой, амплитудой, фазой и смещением. *Параметры блока:* **Sine Type** – способ формирования сигнала реализуется двумя алгоритмами: **Time-based** – по текущему времени (для аналоговых систем) или по значению сигнала на предыдущем шаге и величине такта дискретности (для дискретных систем); **Sample-based** – по величине такта дискретности и количеству расчетных шагов на один период синусоидального сигнала. Вид окна задания параметров меняется в зависимости от выбранного способа формирования синусоидального сигнала.

Выходной сигнал источника в режиме **Time-based (по текущему значению времени для аналоговых систем)** определяется выражением

$$y = \text{Amplitude} \cdot \sin(\text{frequency} \cdot \text{time} + \text{phase}) + \text{bias} .$$

*Параметры блока в режиме **Time-based**: **Amplitude*** – амплитуда; **Bias** – постоянная составляющего сигнала (смещение); **Frequency (rads/sec)** – частота (рад/с); **Phase (rads)** – начальная фаза (рад); **Sample time** – такт дискретности, используется для получения дискретной выборки из непрерывного сигнала и может принимать следующие значения: **0** (по умолчанию) – используется при моделировании непрерывных систем; **> 0** (положительное значение) – задается при моделировании дискретных систем; **-1** (минус один) – такт дискретности устанавливается таким же, как и в предшествующем блоке. Данный параметр может задаваться для многих блоков библиотеки **Simulink**. Флажок **Interpreted vector parameters as 1 – D** – интерпретировать вектор как массив скаляров. Для очень больших значений времени точность вычисления значений сигнала падает.

Выходной сигнал источника в режиме **Time-based (по текущему значению времени для дискретных систем)** определяется матричным выражением

$$\begin{bmatrix} \sin(t + \Delta t) \\ \cos(t + \Delta t) \end{bmatrix} = \begin{bmatrix} \cos(\Delta t) & \sin(\Delta t) \\ -\sin(\Delta t) & \cos(\Delta t) \end{bmatrix} \cdot \begin{bmatrix} \sin(t) \\ \cos(t) \end{bmatrix},$$

где  $\Delta t$  – постоянная величина, равная значению **Sample time**. В данном режиме ошибка округления для больших значений времени меньше, чем в предыдущем режиме.

Выходной сигнал источника в режиме **Time-based (по количеству тактов на период для дискретных систем)** определяется выражением

$$y = A \cdot \sin(2 \cdot \pi \cdot k \cdot f \cdot T + \varphi) + b = A \cdot \sin(2 \cdot \pi \cdot k + l) / N + b,$$

где  $A$  – амплитуда сигнала;  $f$  – частота сигнала в Гц;  $T$  – такт дискретности;  $N$  – количество тактов в секунду;  $k$  – номер текущего шага,  $k = 0, \dots, N - 1$ ;  $\varphi$  – начальная фаза сигнала;  $l$  – начальная фаза, заданная количеством тактов;  $b$  – постоянная составляющая (смещение) сигнала.

*Параметры блока в режиме **Sample-based**: **Amplitude*** – амплитуда; **Bias** – постоянная составляющего сигнала (смещение); **Samples per period** – количество тактов на один период синусоидального сигнала:

$$N = \text{Samples per second} = 1 / (f \cdot T);$$

$$p = \text{Samples per period} = 2 \cdot \pi \cdot N.$$

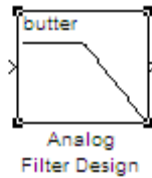
**Number of offset samples** – начальная фаза сигнала, задается количеством тактов дискретности  $l = \varphi \cdot p / (2 \cdot \pi)$ . **Sample time** – такт дискретности. Флажок **Interpret vector parameters as 1 - D** – интерпретировать вектор как одномерный. В данном режиме ошибка округления не накапливается, поскольку **Simulink** начинает отсчет номера текущего шага с нуля для каждого периода.



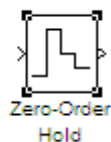
**Random Number** – источник случайного сигнала с нормальным распределением уровня. *Назначение:* формирование сигнала с



равномерным распределением уровня. Параметры блока: **Mean** – среднее значение; **Variance** – дисперсия; **Initial seed** – Начальное значение генератора случайного сигнала; **Sample time** – такт дискретности; флажок **Interpreted vector parameters as 1 – D** – интерпретировать вектор как массив скаляров. В нашем случае блок используется для моделирования шумов канала передачи данных.



**Analog Filter Design** – блок аналогового фильтра заданного метода проектирования и типа из подраздела **Filter Design**; подраздела **Filtering**, раздела **DSP Blockset** библиотеки **Simulink** [4]. *Назначение:* аналоговая фильтрация низкочастотных составляющих спектра входного сигнала. *Параметры блока:* **Design method** – метод проектирования, выбирается из списка: **Butterworth** – фильтр Баттерворта; **Chebyshev I** – фильтр Чебышева 1-го рода; **Chebyshev II** – фильтр Чебышева 2-го рода; **Elliptic** – фильтр эллиптический; **Bessel** – фильтр Бесселя. **Filter type** – тип фильтра, выбирается из списка: **Lowpass** – нижних частот; **Highpass** – верхних частот; **Bandpass** – полосно-пропускающий; **Bandstop** – полосно-заграждающий. Далее для каждого метода проектирования и типа фильтра выдается свой список параметров. Так для фильтра Баттерворта типа нижних частот параметрами являются: **Filter order** – порядок фильтра; **Passband edge frequency (rads/sec)** – нижняя граничная частота (радиан в секунду). Для других методик проектирования и типов фильтров определяемые параметры очевидны.

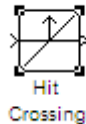


**Zero-Order Hold** – экстраполятор нулевого порядка. *Назначение:* экстраполяция входного сигнала на интервале дискретизации. Блок фиксирует значение входного сигнала в начале интервала дискретизации и поддерживает на выходе это значение до окончания интервала дискретизации. Затем выходной сигнал изменяется скачком до величины входного сигнала на следующем шаге дискретизации. *Параметры блока:* **Sample time** – такт дискретности. Блок экстраполятора нулевого порядка может использоваться также для согласования работы дискретных блоков, имеющих разные такты дискретности.



**Decod\_4psk\_i\_q\_pi\_4** – составной блок восстановления исходной последовательности битовых импульсов по принятым фазовым состояниям несущей из пользовательской библиотеки **Library Communication**. *Назначение:* восстановления исходной

последовательности битовых импульсов по принятым фазовым состояниям несущей. Вначале фазовые состояния несущей декодируются нечетными и четными дибитами, а затем схемой на основе мультипортового переключателя, управляемого генератором пилообразного напряжения преобразуются в последовательность битовых импульсов. Входной и выходной порты служат для соединения с внешними блоками. Входной порт предназначен для ввода демодулированных фазовых состояний несущей. С выходного порта восстановленная исходная последовательность битовых импульсов поступает на выход модема.



**Hit Crossing** -блок определения момента пересечения порогового значения. *Назначение:* определяет момент времени, когда входной сигнал пересекает заданное пороговое значение. *Параметры блока:* **Hit Crossing offset**- порог. Значение уровня, момент пересечения которого входным сигналом необходимо идентифицировать; **Hit Crossing direction**- направление пересечения, выбирается из списка: **rising**-возрастание, **failing**-убывание, **either**-оба направления; Флажок **Show output port**- показать выходной порт. При снятом флажке момент пересечения сигналом порогового уровня определяется, но выходной сигнал блоком не генерируется. Флажок **Enable zero crossing detection**- фиксировать момент прохождения сигнала через нулевой уровень. В момент пересечения сигналом порогового уровня блок вырабатывает единичный сигнал длительностью в один такт дискретности.

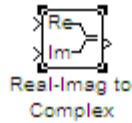


**Repeating Sequence** – источник периодического сигнала. *Назначение:* формирование заданного пользователем периодического сигнала. *Параметры блока:* **Time values** – вектор значений времени; **Output values** – вектор значений сигнала. Блок выполняет линейную интерполяцию выходного сигнала для моментов времени не совпадающих со значениями, заданными вектором **Time values**.

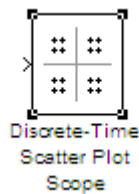


**Multiport Switch** – блок многовходового переключателя. *Назначение:* выполняет переключение входных сигналов на выход по сигналу управления, задающему номер активного входного порта. *Параметры блока:* **Number of inputs** – количество входов; флажок **Show additional parameters** – показать дополнительные параметры, в нашем случае не используется. Блок **Multiport Switch** пропускает на выход сигнал с того входного порта, номер которого равен текущему значению

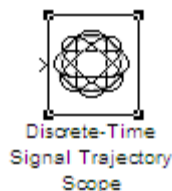
управляющего сигнала. Если управляющий сигнал не является сигналом целого типа, то блок **Multiport Switch** производит округление значения в соответствии со способом, выбранным в графе дополнительного параметра **Round integer calculations toward**.



**Real-Imag to Complex** – блок вычисления комплексного числа по его действительной и мнимой частям. *Назначение:* вычисление комплексного значения по его проекциям на действительную и мнимую оси. *Параметры блока:* **Input** – входной сигнал, значение выбирается из списка: **Real** – действительная часть; **Image** – мнимая часть; **RealAndImage** – действительная и мнимая часть. **Image part** – мнимая часть, параметр доступен, если **Input** определен как **Real**. **Real part** – действительная часть, параметр доступен, если **Input** определен как **Image**. Входные сигналы могут быть скалярными, векторными или матричными. Параметры **Real part** и **Image part** при векторном или матричном входном сигнале должны задаваться как векторы или матрицы.



**Discrete-Time Scatter Plot Scope** – блок отображения диаграммы рассеяния фазовых состояний сигнала из подраздела **Comm Sinks** раздела **Communication Blockset** библиотеки **Simulink** [6]. *Назначение:* отображение диаграммы рассеяния фазовых состояний за счет влияния полосы пропускания и помех тракта передачи. *Параметры блока:* Флажок **Show Plotting Properties** – показать графические установки; **Samples per Symbol** – шаг периода символа; **Offset (samples)** – смещение шагов; **Points displayed** – число отсчетов сигнала, начиная с которого отображается диаграмма; **New points per display** – число отсчетов при обновлении отображения; флажки **Show Rendering Properties**, **Show Axes Properties**, **Show Figure Properties** – показать свойства отображения, осей и фигуры в нашем случае не используются.



**Discrete-Time Signal Trajectory Scope** – блок отображения диаграммы переходов фазовых состояний сигнала из подраздела **Comm Sinks** раздела **Communication Blockset** библиотеки **Simulink** [6]. *Назначение:* отображение диаграммы рассеяния фазовых состояний за счет влияния полосы пропускания и помех тракта передачи. *Параметры блока:* Флажок **Show Plotting Properties** – показать графические

установки; **Samples per Symbol** – шаг периода символа; **Offset (samples)** – смещение шагов; **Points displayed** – число отсчетов сигнала, начиная с которого отображается диаграмма; **New points per display** – число отсчетов при обновлении отображения; флажки **Show Rendering Properties, Show Axes Properties, Show Figure Properties** – показать свойства отображения, осей и фигуры в нашем случае не используются.

В нашем случае диаграммы блоков **Discrete-Time Scatter Plot Scope** и **Discrete-Time Signal Trajectory Scope** позволяют оценить влияние метода модуляции, полосы пропускания и шумов канала передачи на фазовую диаграмму состояний принятого сигнала.

## 5. ЭКСПЕРИМЕНТАЛЬНОЕ ЗАДАНИЕ

1. Собрать **Sim**-модель для исследования **Pi/4\_QPSK** модема с кодовым принципом модуляции и демодуляции фазовых состояний несущей частоты, в соответствии с рисунком 1.1.

2. Выставить параметры блоков **Sim**-модели, согласованные с исходными параметрами блока источника случайного сигнала с равномерным распределением (**Uniform Random Number**), например: **Minimum = -1; Maximum = 1; Initial seed = 11; Sample time = 1**.

3. Пронаблюдать и зафиксировать основные осциллограммы, иллюстрирующие работу модема квадратурной **Pi/4\_QPSK** манипуляции.

4. Изменить параметр **Initial seed** блока источника случайного сигнала с равномерным распределением (**Uniform Random Number**) и пронаблюдать работу модема **Pi/4\_PSK** при другой входной последовательности.

5. Изменить параметр **Sample time** на (**0.5** или **2**) блока источника случайного сигнала с равномерным распределением (**Uniform Random Number**), выставить согласованные параметры остальных блоков **Sim**-модели, пронаблюдать работу **Pi/4\_QPSK** модема и сделать выводы.

6. Модернизировать функциональную схему фазового кодера, исключив блок **Combinatorial Logic** и преобразователь код-амплитуда, заменив последний **ET**-подсистемой (**Enabled and Triggered Subsystem**) с двумя новыми встроенными блоками задания *M-функций* (**MatLab Fnc**) и (**MatLab Fnc1**).

7. Преобразовать схему восстановления исходной битовой последовательности импульсов по принятым фазовым состояниям несущей, реализованную на основе двухпортового переключателя (**Multiport Switch**), управляемого генератора пилообразного напряжения (**Repeating Sequence**), в схему на основе простого переключателя (**Switch**), управляемого опорным генератором прямоугольных импульсов (**Pulse Generator**) с периодом равным дибиту и скважностью **2**.

## 6 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое квадратурная модуляция и перечислить основные ее преимущества?
2. Объясните работу **Pi/4\_QPSK модулятора** модема с кодовым принципом модуляции и демодуляции фазовых состояний несущей частоты при помощи комплексной формы записи сигнала.
3. Объясните работу **Pi/4\_QPSK демодулятора** модема с кодовым принципом модуляции и демодуляции фазовых состояний несущей частоты при помощи комплексной формы записи сигнала.
4. Чем обусловлено использование ортогональных несущих  $\cos(\omega \cdot t)$  и  $\sin(\omega \cdot t)$  в **Pi/4\_QPSK** модуляторе и демодуляторе?
5. Объясните принцип работы формирователя квадратурных модулирующих импульсов фазового манипулятора.
6. Объясните принцип работы схемы восстановления исходной битовой последовательности импульсов по принятым фазовым состояниям несущей.
7. Какую функцию в схеме выполняет блок **Enabled and Triggered Subsystem** со встроенными блоками **MatLab Function**?
8. Какую функцию в схеме выполняют блоки **Analog Filter Design**?
9. Опишите изменения в спектре сигнала при расширении его длительности во временной области в два раза.
10. Как влияет увеличение длительности бита в квадратурных каналах в два раза на скорость передачи информации?
11. В чем преимущества манипуляции **Pi/4\_QPSK** перед **OQPSK** манипуляцией?
12. Чем отличается модулятор **Pi/4\_QPSK** с фазовым кодером от модулятора без кодера?
13. Изобразите функциональную схему квадратурного фазового модулятора исследуемого **Pi/4\_QPSK** модема.
14. Изобразите функциональную схему квадратурного фазового демодулятора исследуемого **Pi/4\_QPSK** модема.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Скляр Б. Цифровая связь. Теоретические основы и практическое применение, 2-е изд., исправл.: Пер. с англ. – М.: Изд. дом “Вильямс”, 2003.- 1104 с.
2. Рихтер С.Г. Цифровое радиовещание. – Горячая линия-Телеком, 2004.- 352 с.
3. Гультаев А.К. **MatLab 5.3**. Имитационное моделирование в среде **Windows**: Практическое пособие. – СПб.: КОРОНА Принт, 2001.- 400 с.
4. Черных И.В. **Simulink**: среда создания инженерных приложений. / Под общ. ред. В.Г. Потемкина – М.: ДИАЛОГ-МИФИ, 2003.- 496 с.

5. Дьяконов В.П. **MatLab 6.5 SP1/7 + Simulink 5/6**. Основы применения. Сер. Библиотека профессионала. - М.: СОЛОН-Пресс, 2005.- 800 с.

6. Дьяконов В.П. **MatLab 6.5 SP1/7 + Simulink 5/6** в математике и моделировании. Сер. Библиотека профессионала. - М.: СОЛОН-Пресс, 2005.- 576 с.