Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ» (ТУСУР)

Е.В. Рогожников, А. Коновальчиков, А.С. Бокушин

ПРОГРАММНО – ОПРЕДЕЛЯЕМЫЕ РАДИОСИСТЕМЫ

Методические указания для выполнения практических работ и самостоятельной работы для студентов технических направлений подготовки и специальностей

Томск
2026

УДК 681.3.068
ББК 32.973.2
Р 59

Рецензент:
Дмитриев Э., доцент кафедры телекоммуникаций и основ радиотехники ТУСУРа, кандидат технических наук

Р 598 Программно-определяемые радиосистемы: Методические указания для выполнения практических работ и самостоятельной работы для студентов технических направлений подготовки и специальностей / Е.В. Рогожников, А. Коновальчиков, А.С. Бокушин – Томск: Томск. гос. ун-т систем управления и радиоэлектроники, 2026. – 50 с.

Настоящие учебно-методическое пособие содержит указания по выполнению практических работ и самостоятельной работы. Данный практикум имеет целью закрепить и расширить теоретические знания студентов в области цифровой обработки сигналов путем обеспечения работы студентов с реальными сигналами, полученными из радиоэфира, что также позволит применить на практике имеющиеся знания.

Одобрено на заседании кафедры ТОР, протокол № 4 от 26 декабря 2025 г.

УДК 681.3.068
ББК 32.973.2

Оглавление

ВВЕДЕНИЕ

Практикум имеет целью закрепить и расширить теоретические знания студентов в области цифровой обработки сигналов путем обеспечения работы студентов с реальными сигналами, полученными из радиоэфира, что также позволит применить на практике имеющиеся знания. Практикум содержит описание следующих работ:

1) Прием FM сигнала;
2) Введение в Gnu Radio;
3) Прием и обработка сигнала FM-радиостанции в среде MATLAB;
4) Поиск сигнала в эфире (несущая, полоса) и его прием;
5) Формирование опорного сигнала из предоставленной последовательности;
6) Частотная синхронизация;
7) Оценка АЧХ по пилотным поднесущим. Эквалайзирование;
8) Декодирование и восстановление принятого сообщения.

Работы данного перечня выполняются с помощью цифрового приемника RTL-SDR и в средах разработки Gnu Radio, MATLAB. Также в работах используется вспомогательное ПО – SDR#, применяемое для обнаружения и анализа сигналов в эфире и взаимодействия с RTL-SDR. Среды разработки Gnu Radio и MATLAB – это ПО, необходимое для взаимодействия с программно-определяемыми радиосистемами, в том числе RTL-SDR, а также осуществления цифровой обработки сигналов.

Работа № 1

«Знакомство с RTL-SDR прием FM сигнала»

Цель работы: ознакомиться с RTL-SDR, принять FM сигнал.
Задачи работы:

- Знакомство с RTL-SDR;
- Прием FM сигнала.

1. Теоретическая часть

RTL-SDR – это целое семейство дешевых ТВ-тюнеров, способных выполнять функцию SDR-приемника. Они имеют разные названия и бренды, но объединяет их одно – все они построены на чипсете RTL2832. Это микросхема, содержащая два 8-битных АЦП с частотой дискретизации до 3,2 МГц (однако выше 2,8 МГц могут быть потери данных), и интерфейс USB для связи с компьютером. Эта микросхема на входе принимает I- и Q-потоки, которые должны быть получены другой микросхемой. Блок схема RTL-SDR представлена на рисунке 1.1.

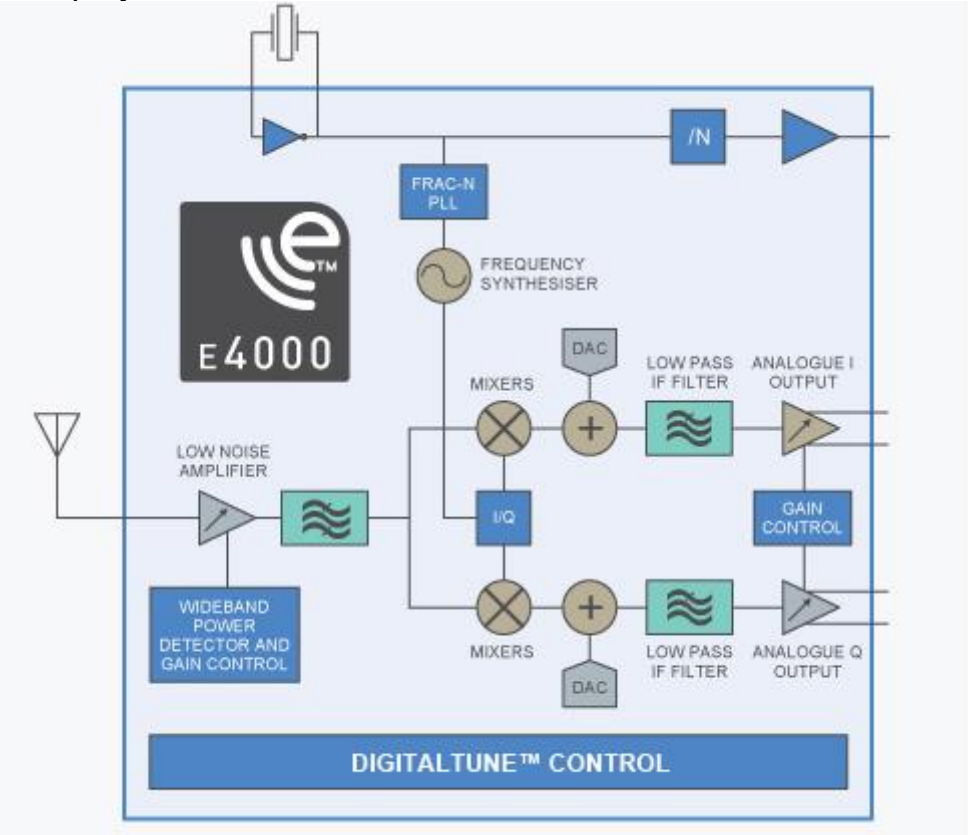


Рисунок 1.1 – Блок схема RTL-SDR

FM радиовещание – это метод радиовещания, который использует частотную модуляцию (Frequency Modulation) (рисунок 1.2) для передачи аудиосигнала по радиоволнам. В этой системе информация кодируется путем изменения частоты несущей волны в соответствии с амплитудой аудиосигнала.

Основные черты FM радиовещания включают:

- Высокое качество звука: FM обычно предоставляет более высокое качество звука по сравнению с AM (амплитудной модуляцией) радиовещанием. Это связано с тем, что FM менее подвержена шумам и интерференциям.

• Стабильность сигнала: FM обладает более стабильным сигналом, чем AM, что делает его более подходящим для передачи музыкальных программ и высококачественных аудиозаписей.

• Широкий диапазон частот: FM вещание может использовать различные частоты для передачи разных станций. В большинстве стран существует много FM-радиостанций, предлагающих разнообразные программы.

FM радиовещание является популярным средством распространения радиопрограмм, музыки и новостей, и его чистое звучание и широкий диапазон частот делают его одним из предпочтительных вариантов для радиослушателей.

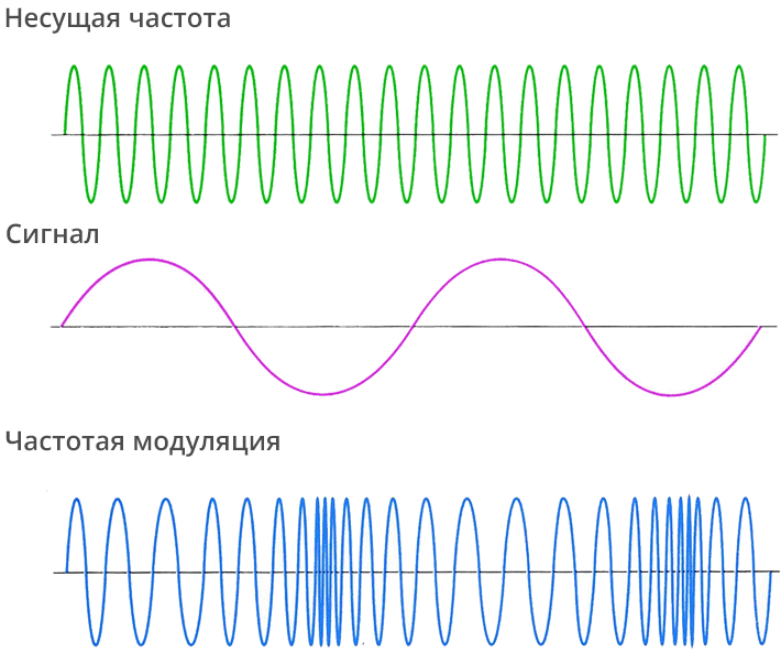


Рисунок 1.2 – Частотная модуляция сигнала

2. Практическая часть

Запустите программу SDRsharp.exe (рисунок 2.1).

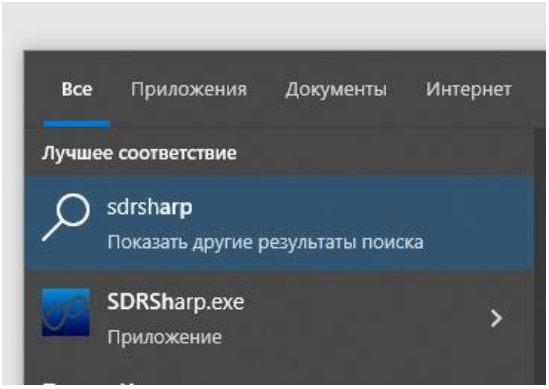


Рисунок 2.1 – SDRSharp в проводнике

При открытии программы вы увидите следующее окно, представленное на рисунке 2.2.

Рисунок 2.2 – Рабочая область программы SDRSharp

В окне source в поле Device выберите Generic RTL2832U OEM, Sample Rate укажите 2,4 MSPS, усиление установите на максимум (рисунок 2.3).

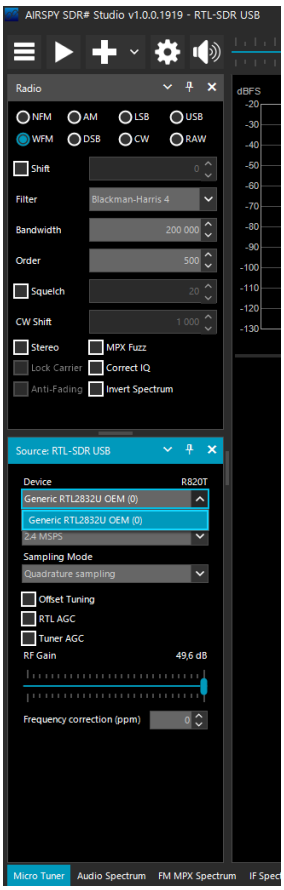


Рисунок 2.3 – Настройка программы SDRSharp

Установите частоту радиостанции, как на рисунке 2.4.

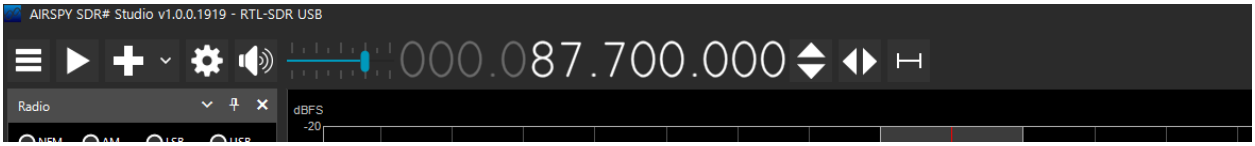


Рисунок 2.4 – Настройка частоты радиостанции

Далее на рисунке 2.5 вы увидите спектр принимаемой полосы частот – верная часть рисунка и спектрограмму – нижняя часть. Так же в наушниках вы должны будите слышать принятый сигнал FM радиостанции.

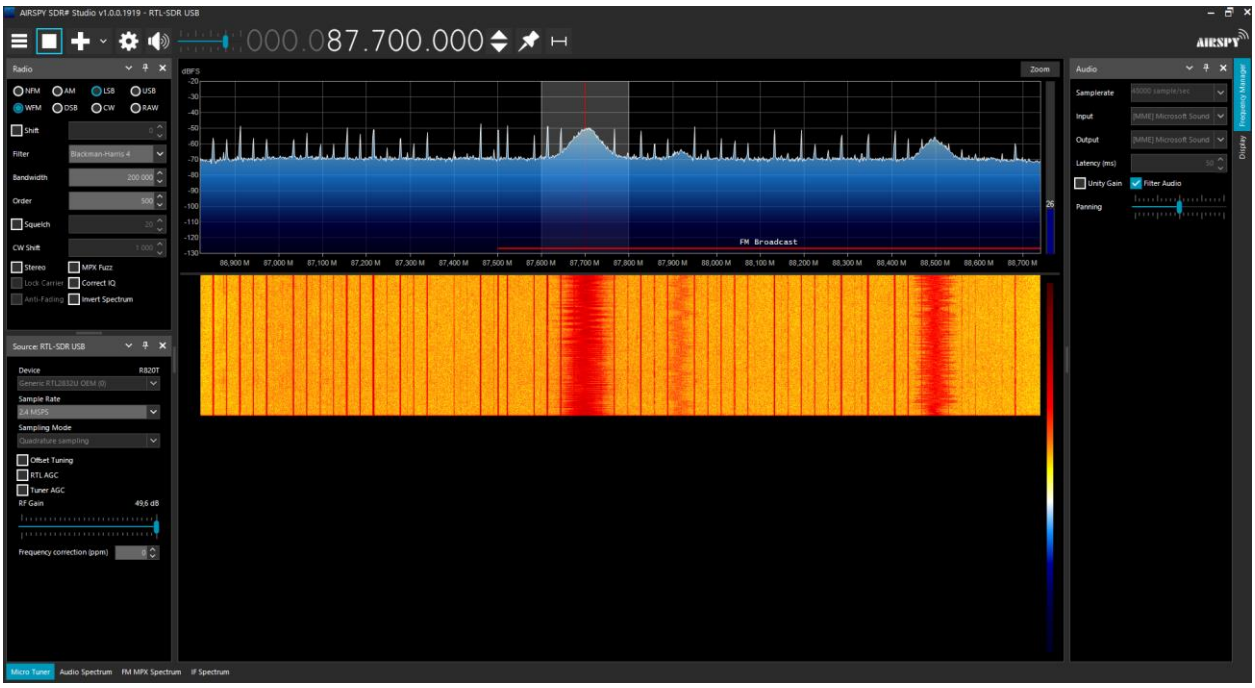


Рисунок 2.5 – Спектр и спектрограмма принимаемого сигнала

3. Самостоятельная часть

Найдите другие частоты FM радиостанций и примите их.

4. Контрольные вопросы к работе

1) Что такое RTLSDR и для чего его используют?
2) Что такое FM модуляция?

5. Требования к оформлению отчета

1) Введение;
2) Входные параметры;
3) Рисунки;
4) Выводы.

Работа № 2

«Введение в Gnu Radio, прием и передача OFDM»

Цель работы: ознакомиться с программным обеспечением Gnu Radio, сформировать и передать OFDM символ

Задачи работы:

1) Ознакомиться с программным обеспечением Gnu Radio;
2) Сформировать и передать OFDM символ.

1. Теоретическая часть

Gnu Radio (GNU Radio Companion или GRC) – это программное обеспечение с открытым исходным кодом, разработано для разработки программного радио и сигнальной обработки на основе открытых стандартов и аппаратного обеспечения.

Основная идея Gnu Radio заключается в том, чтобы предоставить инструменты для создания и выполнения радиосвязи и сигнальной обработки с использованием программного обеспечения, в отличие от традиционных аппаратных средств. Он позволяет пользователям создавать гибкие и адаптивные радиосистемы, используя блоки сигнальной обработки, которые могут быть соединены вместе для создания комплексных радиосистем.

Gnu Radio поддерживает различные типы аппаратного обеспечения, включая универсальные программные радио (USRP) от компании Ettus Research, которые являются основным потоком разработки. Однако, благодаря своей архитектуре, Gnu Radio может быть использован с различными другими устройствами и радиоинтерфейсами.

Gnu Radio также предоставляет графическую среду разработки для создания сигнальных цепей, называемых GNU Radio Companion. С помощью GRC разработчики могут создавать радиосистемы путем перетаскивания и подключения блоков сигнальной обработки, что делает процесс разработки более интуитивным и доступным.

Программное обеспечение Gnu Radio широко используется в исследовательских и академических целях, а также применяется в профессиональной индустрии для разработки радиосистем, радио-сканеров, радиосвязи, радиолокации, радиовещания и других приложений сигнальной обработки.

Одним из основных преимуществ Gnu Radio является его открытая архитектура и подход к разработке, который позволяет пользователям создавать и делиться своими сигнальными цепями и блоками сигнальной обработки. Это повышает доступность и обмен знаниями, улучшает разработку программного радио и способствует инновациям в этой области.

2. Среда разработки Gnu Radio

Для запуска GRC откройте терминал (Ctrl+Alt+T) и введите: gnuradio-companion, как показано на рисунке 2.1.

После выполнения команды в терминале откроется окно программы gnuradio-companion, на котором уже расположены два блока. Блок «Options» в верхнем левом углу используется для установки некоторых общих параметров графа, такие как графический пользовательский интерфейс (GUI) для виджетов и вывода результатов моделирования на дисплей, или размер холста, на котором размещены блоки.

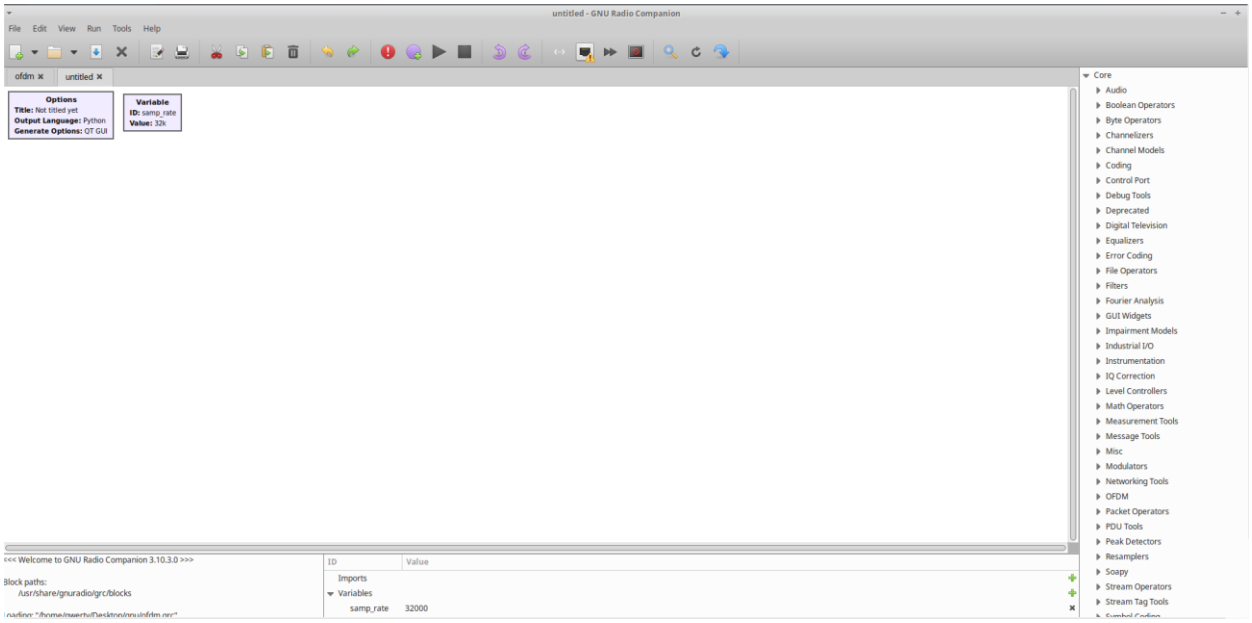
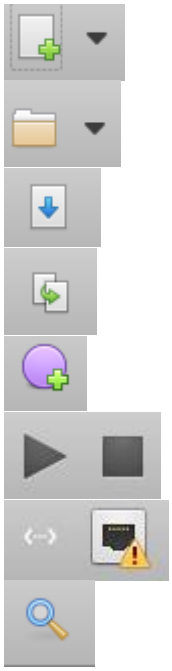


Рисунок 2.1 – Окно программы Gnuradio-companion

Панель меню, которая изображена на рисунке 2.1, состоит из следующих команд:



– Создать новую модель;

– Открыть модель;

– Сохранить модель;

– Команды копирования;

– Компиляция;

– Запуск и остановка моделирования;

– Включение и отключение блока от функционирования модели;

– Поиск блоков по названию.

В нижней части экрана расположено окно терминала, в котором отображается процесс моделирования и описание ошибок, если таковые имеются. Также в нижней части экрана можно увидеть окно, в котором отображаются значения переменных, используемых в модели. Щелкните правой кнопкой мыши на блоке и выберите «Options» (или дважды щелкните на блоке), чтобы увидеть все параметры, которые могут быть настроены. Окно параметров блока «Options» представлено на рисунке 2.2.

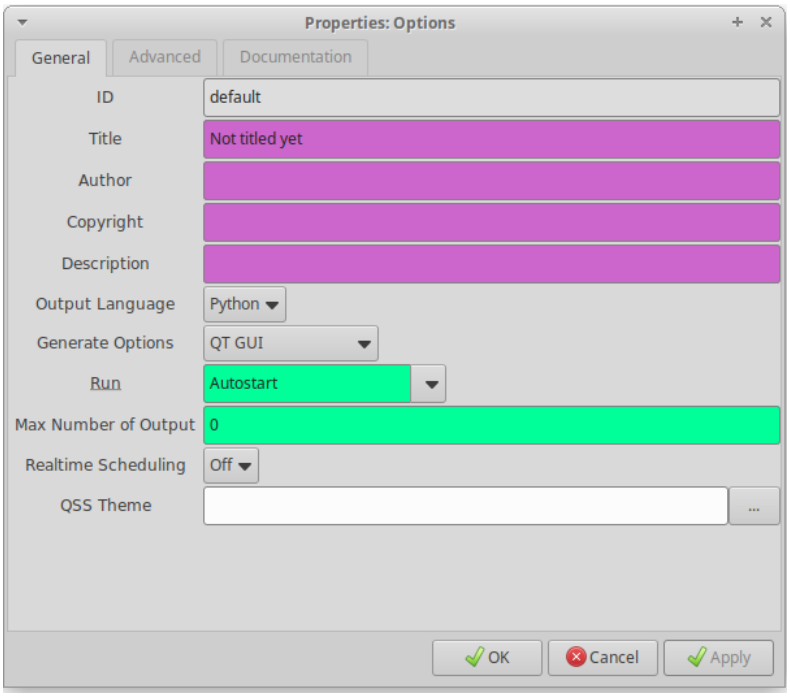


Рисунок 2.2 – Окно параметров блока «Options»

Пока мы оставим настройки по умолчанию без изменений. Ниже расположен блок «Variable» с названием «samp_rate», который используется для установки частоты дискретизации. Блок «Variable» – это блок переменная, в поле «ID» указывается название переменной, а в поле «Value» значение. Если указать в поле какого-либо блока название созданной переменной (например, «samp_rate»), то в поле будет записано значение данной переменной.

В правой части рабочего окна находится список доступных для моделирования блоков, которые разбиты на соответствующие категории (рисунок 2.3).

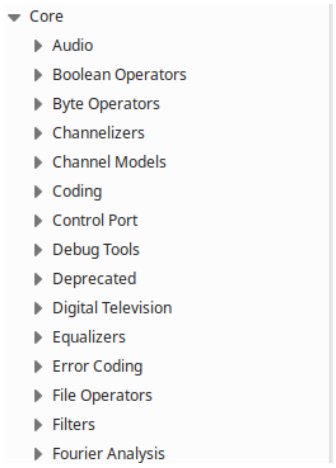


Рисунок 2.3 – Список блоков для моделирования

Нажмите на треугольник рядом с категорией, например, Генераторы сигналов (Waveform Generators), чтобы увидеть, какие блоки доступны в этой категории. Здесь расположены такие часто используемые блоки, как генератор периодических сигналов (Signal Source), источник шума (Noise Source), Генератор случайной последовательности символов (Random Source) и другие. Поиск необходимых блоков можно выполнить, нажав на значок лупы, расположенный на панели меню в верхней части экрана.

Используйте функцию поиска (нажмите на значок лупы в правом верхнем углу), чтобы ввести ключевое слово, например, Source, чтобы увидеть все блоки с «Source» в названии.

Щелкните на «Signal source» и перетащите его в рабочую область, как показано на рисунке 2.4.

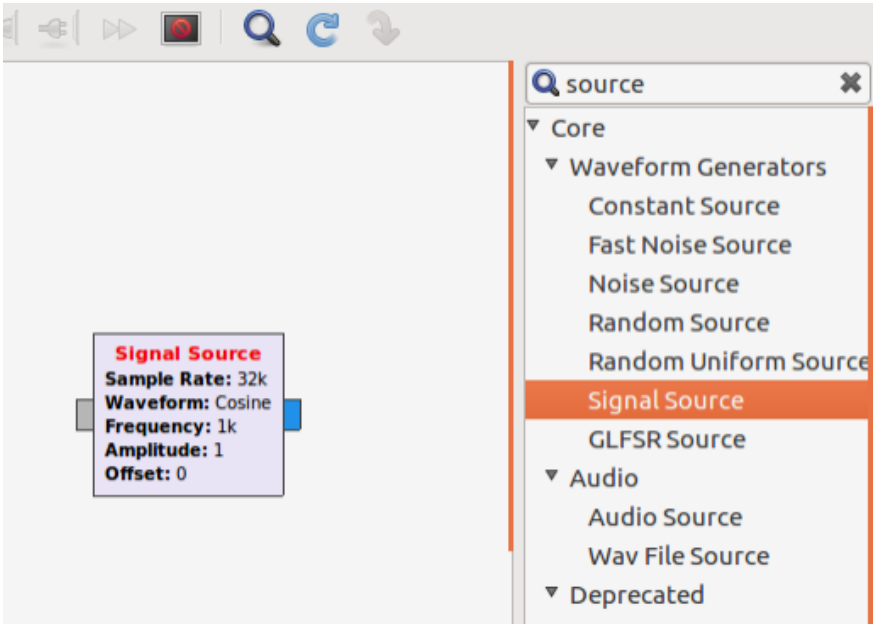


Рисунок 2.4 – Перенос блока в рабочую область программы

Вы также можете вращать блоки, щелкнув правой кнопкой мыши на них, а затем нажав либо «Rotate Counterclockwise» или «Rotate Clockwise». Блоки также могут быть временно отключены, нажав на «Disable», который полезен для отладки.

Обратите внимание на то, что блок «Signal source» имеет два порта, один серый слева и синий справа. Цвет порта указывает на тип данных, генерируемых для выходного порта или типа данных, принятых для входа.

Для того чтобы увидеть используемые в GRC типы данных откройте пункт «types» во вкладке меню «help» (рисунок 2.5). Для того чтобы открыть вкладку «help» в Ubuntu 16.04 необходимо навести курсор на верхнюю часть окна программы.

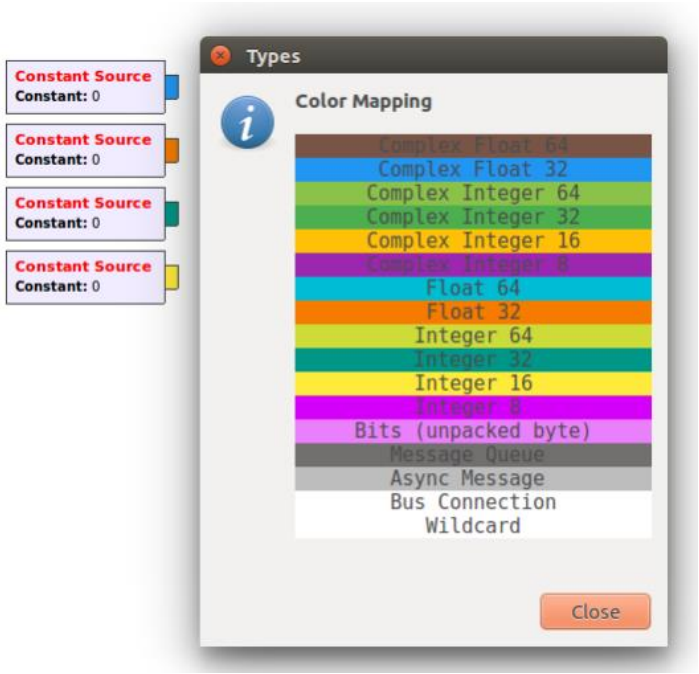


Рисунок 2.5 – Типы данных

GRC использует потоковую модель обработки для работы с большими объемами данных в режиме реального времени, в отличие от традиционной среды обработки массивов (например, Matlab). На практике это означает, что каждый блок обработки сигналов имеет независимый планировщик и работает в своем собственном потоке. То есть если вы работаете с приемо-передающим оборудованием который работает с определенной частотой дискретизации данных (например, 44100 выборок / сек для звукового сигнала, или 10 миллионов выборок / сек для и интерфейса SDR), то и частоту дискретизации моделирования необходимо выставить соответствующую. Но если и источник, и приемник реализуются исключительно в программном обеспечении (например, генератор сигналов или индикация частоты), то будет полезным ограничить скорость обработки данных при конкретной частоте дискретизации вашей модели. Для этого используется блок «Throttle». Его использование значительно снизит нагрузку на процессор и увеличит скорость моделирования.

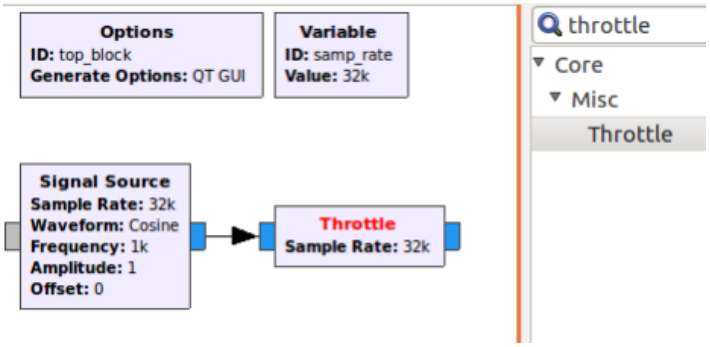


Рисунок 2.6 – Подключение блока «Throttle»

Подключите блок «Throttle» к выходу блока «Signal source» (рисунок 2.6) последовательно кликнув левой кнопкой мыши на порты каждого из блоков.

Примечание! Чтобы блоки работали, оба порта должны использовать данные одного и того же типа (то есть, оба порта должны быть одного и того же цвета).

Если типы данных различны, то стрелка соединения будет красного цвета вместо черного. Обратите также внимание, что надпись на блоке становится красной, указывая, что

с этим блоком что-то не так. Так, же моделирование не может быть запущено, если некоторые порты не подключены.

Чтобы изменить тип выходных данных блока «Signal source» на **float**, дважды щелкните на блоке и в окне «Properties» нажмите на «Complex» в разделе «Output type», затем выберите «Float» (рисунок 2.7).

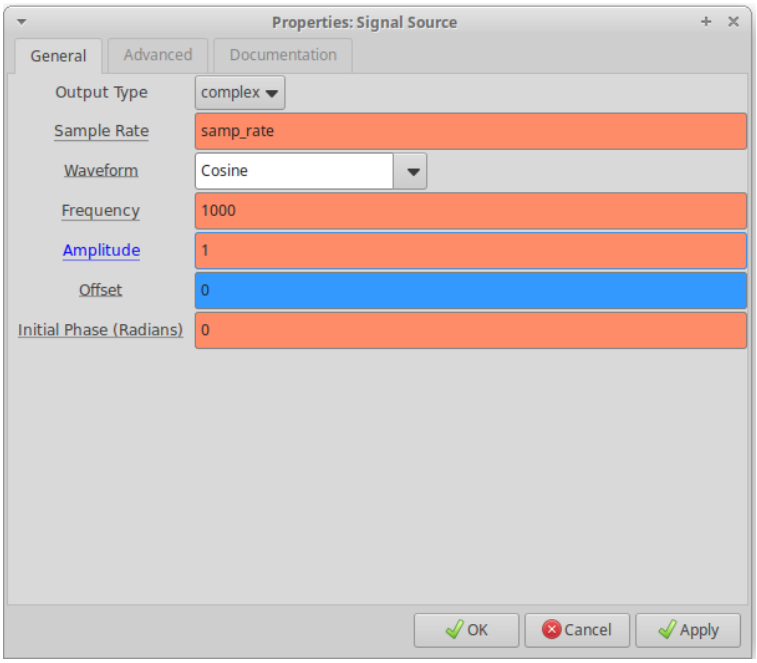


Рисунок 2.7 – Параметры блока «Signal source»

Повторите эти действия для блока «Throttle».

Передача и прием OFDM символа

Соберите модель приемо-передатчика как показано на рисунке 2.8.

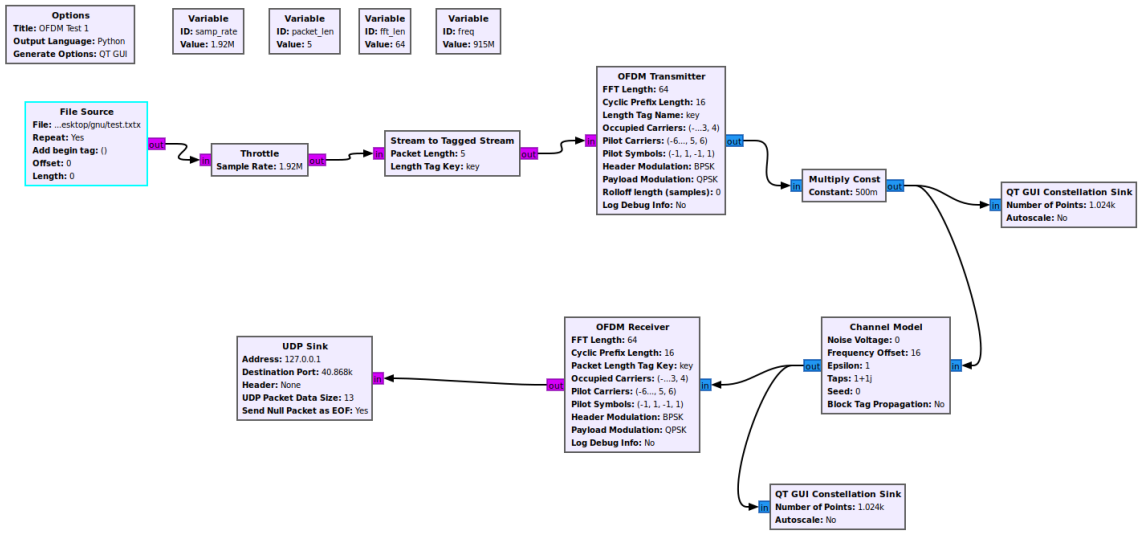


Рисунок 2.8 – Модель OFDM приемо-передатчика

В папке с проектом создайте файл “tx.txt” и запишите в него сообщение с вашим именем и фамилий, а также номером группы на английском языке, например, «Ivanov Ivan group 111». Путь к этому файлу необходимо указать в блоке «File Source», как это показано на рисунке 2.9.

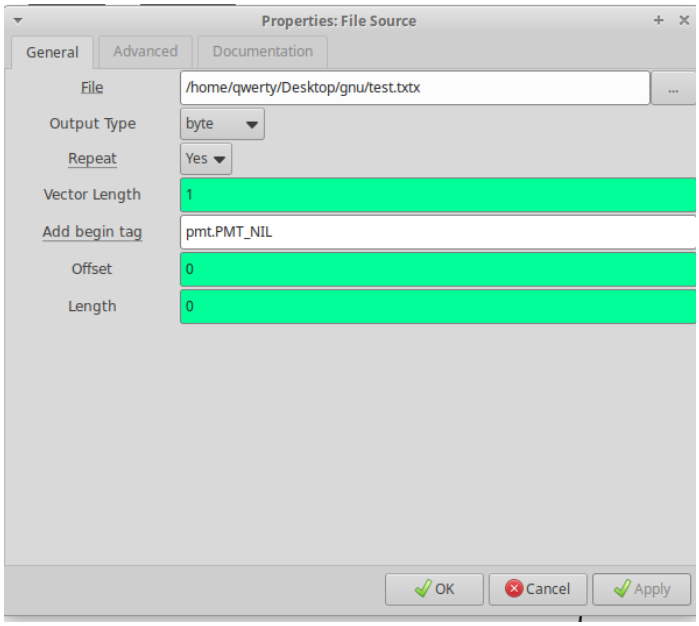


Рисунок 2.9 – настройка блока «File Source»

Параметры блока «OFDM Transmitter» представлены на рисунке 2.10.

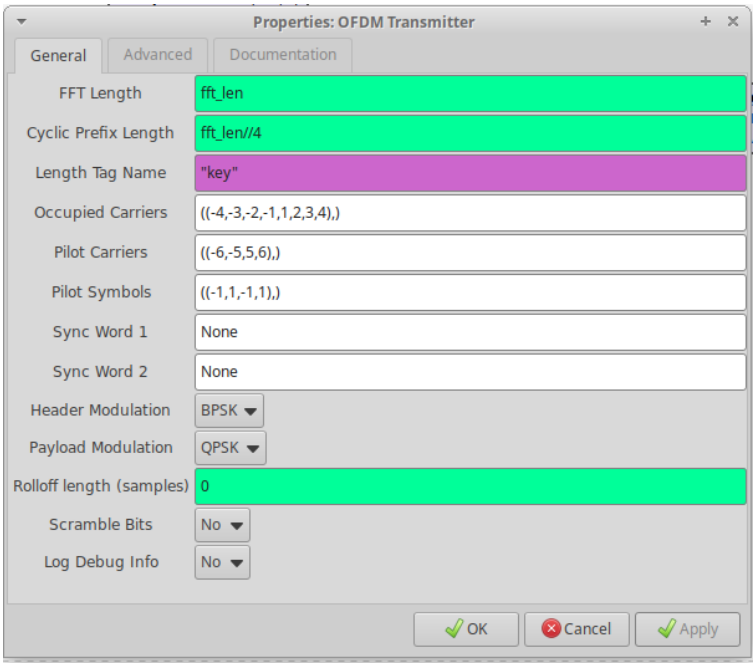


Рисунок 2.10 – настройка блока «OFDM Transmitter»

Параметры блока «OFDM Receiver» представлены на рисунке 2.11.

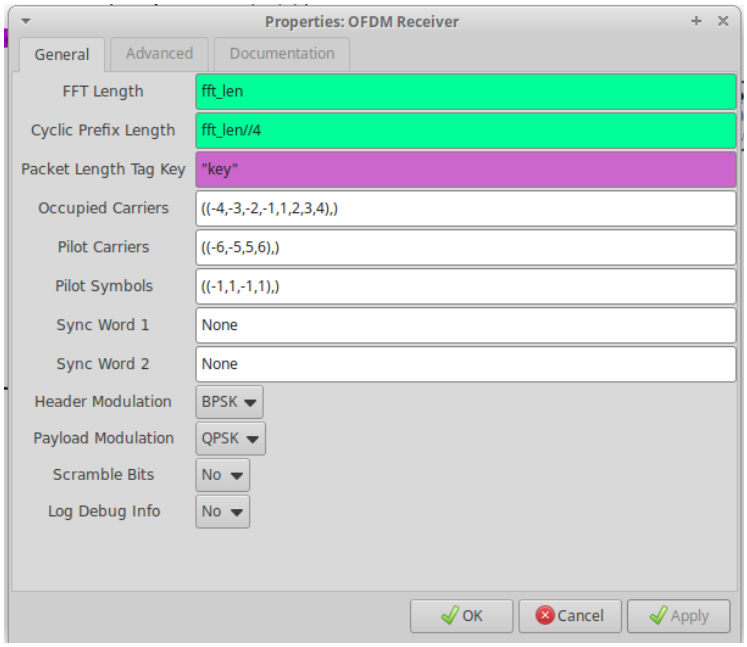


Рисунок 2.11 – Настройка блока «OFDM Receiver»

Передав сообщение по виртуальному каналу модернизируйте схему как показано на рисунке 2.12.

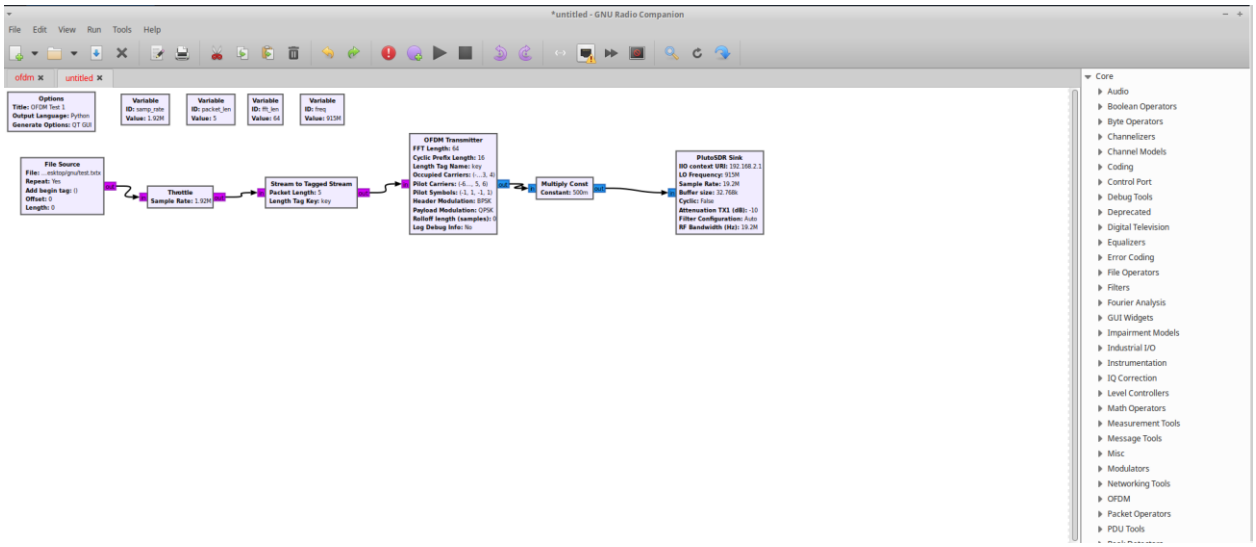


Рисунок 2.12 – Схема передатчика

Передавая сообщение циклично на мониторе преподавателя вы должны будете увидеть передаваемое сообщение.

3. Контрольные вопросы к работе

1) Что такое Gnu Radio и для чего его используют?
2) Что такое OFDM символ?
3) Для чего нужен блок «Throttle»?

4. Требования к оформлению отчета

1) Введение;

2) Входные параметры;
3) Алгоритм OFDM передатчика;
4) Рисунки:
   a. Схема приемо-передатчика
   b. Принятое созвездие;
   c. Схема передатчика
   d. Текст переданного сообщения;
   e. Текст принятого сообщения;
5) Выводы.

Работа № 3

«Прием и обработка сигнала FM-радиостанции в среде MATLAB»

Цель работы: разработать модель приема и обработки сигнала FM-радиостанции в среде MATLAB.

Задачи работы:

1) Обнаружить и принять сигнал FM-радиостанции;
2) Выполнить фильтрации FM-сигнала;
3) Выполнить демодуляцию FM-сигнала;
4) Выделить и воспроизвести аудиосигнал FM-радиостанции.

1. Теоретическая часть

В данной работе необходимо принять и обработать сигнал FM-радиостанции в среде разработки MATLAB при помощи RTL SDR.

В таблице 1.1 указаны входные параметры модели.

Таблица 1.1 – Параметры модели

Параметр	Обозначение	Значение	Пояснения
Полоса сигнала RTL SDR	fs	$2 \cdot 10^6$	Полоса принимаемого сигнала
Несущая частота радиостанции RTL SDR	fc	-	Несущая частота радиостанции
Девиация частоты	wd	$2 \cdot \pi \cdot 75 \cdot 10^3$	Стандартное значение девиации частоты для FM модуляции
Частота среза фильтра	fCutOff	$100 \cdot 10^3$	Необходима для селектирования полезного сигнала из принятого
Коэффициент децимации	decimFactor	10	Коэффициент децимации
Количество семплов принимаемых RTL SDR	SPF	375000	Максимальное количество семплов, единовременно, принимаемых RTL SDR в одном кадре
Количество кадров	NumFrame	20	Количество кадров, принимаемых RTL SDR
Усиление RTL SDR	TunerGain	48	Усиление сигнала на входе приемника
Полоса моно аудио сигнала	fCutOffMono	$15 \cdot 10^3$	Частота среза для выделения моно аудиосигнала
Полоса звуковой карты	fAudio	$48 \cdot 10^3$	Полоса сигнала звуковой карты персонального компьютера

Мгновенная частота сигнала равна производной от его полной фазы. В частотной модуляции информационный сигнал в зависимости от текущей амплитуды или частоты изменяет мгновенную частоту модулированного сигнала:

$$w(t) = w_d \cdot s(t), \tag{1.1}$$

где $w(t)$ – мгновенная частота;

w_d – девиация частоты;

$s(t)$ – информационный сигнал.

Информационный сигнал нормируется к единице, а умножение на нормированной сигнал девиации частоты задает максимальное изменение мгновенной частоты. Чтобы перейти от мгновенной частоты к полной фазе, необходимо вычислить интеграл он мгновенной частоты. Комплексная огибающая ФМ сигнала имеет вид

$$z(t) = \exp\left(j \cdot w_d \cdot \int s(t)\,dt\right). \tag{1.2}$$

В приемнике, после обработки сигнала квадратурным демодулятор останется синфазная и квадратурная компоненты

$$\begin{aligned} A(t) &= \cos\left(w_d \cdot \int s(t)\,dt\right) \\ B(t) &= \sin\left(w_d \cdot \int s(t)\,dt\right) \end{aligned}. \tag{1.3}$$

Для извлечения информационного сигнала необходимо найти арктангенс от данных компонент

$$\arctan\left(\frac{B(t)}{A(t)}\right) = w_d \cdot \int s(t)\,dt\,. \tag{1.4}$$

Арктангенс является периодичной функцией. Выполнение дифференцирования позволит получить информационный сигнал и избавиться от эффекта периодичности.

$$w(t) = \frac{d}{dt}\arctan\left(\frac{B(t)}{A(t)}\right) = \frac{B'(t)A(t) - A'(t)B(t)}{A^2(t)} \cdot \frac{1}{1 + \dfrac{B^2(t)}{A^2(t)}} = \frac{B'(t)A(t) - A'(t)B(t)}{A^2(t) + B^2(t)}. \tag{1.5}$$

2. Ввод параметров модели

В среде разработки MATLAB сформируйте параметры модели, взяв их из таблицы 1. Несущую частоту сигнала определите сами, найдя радиостанцию с наибольшей мощностью передачи в программе SDRSharp.

Следующим шагом необходимо дополнительно рассчитать ряд параметров, которые потребуются дальше, таких, как нормированная частота среза; полоса сигнала после децимации; общее количество, принимаемых при помощи RTL SDR, семплов; общее количество семплов после децимации.

Общее количество принимаемых семплов рассчитывается как максимальное количество семплов, единовременно, принимаемых RTL SDR умноженное на количество кадров принимаемого сигнала:

$$NumSamp = SPF \cdot NumFrame. \tag{2.1}$$

Из принимаемой полосы необходимо осуществить селекцию полезного сигнала с помощью фильтра. Частота среза для фильтра задается в нормированном виде. Нормирование частоты среза выполняется относительно частоты Найквиста, которая равна половине полосы принятого сигнала. Расчет нормированной частоты среза выполняется как удвоенная частота среза фильтра, деленая на полосу сигнала:

$$fCutOffNorm = fCutOff \cdot 2 / fs. \quad (2.2)$$

После селекции сигнала, необходимо выполнить децимацию, чтобы привести сигнал к другой полосе частот. Полоса сигнала после децимации рассчитывается как полоса сигнала, деленная на коэффициент децимации:

$$fsDecim = fs / decimFactor. \quad (2.3)$$

Общее количество семплов после децимации – общее количество принимаемых семплов, деленное на коэффициент децимации:

$$NumSampDecim = NumSamp / decimFactor. \quad (2.4)$$

Задайте входные параметры в соответствие с параметрами из таблицы 1.1.

```
fs = …; % Полоса сигнала
fc = …; % Несущая частота
wd = …; % Девиация частоты
fCutOff = …; % Частота среза фильтра
fCutOffNorm = …; % Частота среза нормированная
decimFactor = …; % Коэффициент децимации
fsDecim = …; % Полоса сигнала после децимации
SPF = …; % Количество принимаемых семплов
NumFrame = …; % Количество кадров
NumSamp = … % Общее кол-во семплов
NumSampDecim = …; % Общее кол-во семплов после децимации
```

После ввода параметров необходимо настроить RTL SDR. Для этого необходимо создать объект класса RTL SRD в MATLAB.

```
rx = comm.SDRRTLReceiver;
```

Затем, обращаясь к свойствам объекта rx осуществите настройку вашего приемника. Свойство класса `CenterFrequency` отвечает за установку несущей частоты. Установите его значение равное значению оцененной вами несущей частоте.

Свойство `SampleRate` отвечает за установку полосы сигнала в которой будет осуществляться прием.

`SamplesPerFrame` отвечает за количество принимаемых отсчетов которое RTL SDR может принять единовременно. Максимальное значение, которое RTL SDR может принимать равно 375000. Важно, что RTL может непрерывно принимать несколько кадров по 375000 отсчетов. Следовательно, если размер принимаемого сигнала выходит за рамки значения 375000, то необходимо просто вести прием в цикле до тех пор, пока вами не будет накоплено необходимое количество отсчетов. О приеме сигнала с RTL SDR чуть позже.

Свойство `EnableTunerAGC` отвечает за включение автоматического усилителя. Установите значение данного свойства равное 0.

Свойство `TunerGain` отвечает за ручную установку усиления. Установите данный параметр равный усилению, которое было установлено в SDRSharp. Этого должно быть достаточно для оптимального уровня принимаемого сигнала. В дальнейших работах этот параметр может меняться в зависимости от помеховой обстановки в канале. Установить усиление можно в диапазоне от 0 до 49,6 дБ.

Свойство `OutputDataType` отвечает за тип данных, которые будут приниматься RTL. Установите тип `'double'`.

```
rx.CenterFrequency = …; % Несущая частота
rx.SampleRate = …; % Полоса сигнала из эфира
rx.SamplesPerFrame = …; % Кол-во семплов
rx.EnableTunerAGC = 0; % Отключение усилителя автоматического усиления
rx.TunerGain = …; % Усиление приемника
rx.OutputDataType = 'double'; % Тип принимаемых данных
```

Заполните параметры приемника, в соответствие с данными приведенными в таблице 1.1.

После заполнения всех параметров можно считать, что RTL SDR настроена и можно переходить к записи данных из эфира. Создайте нулевой массив длинной `NumSamp`, затем создайте цикл, где итерируемая переменная будет изменяться от 1 до количества принимаемых кадров `NumFrame`, в котором будет производится запись данных из эфира при помощи вызова метода объекта `rx()` в ваш нулевой массив с шагом `SPF`. То есть, вы будете принимать несколько кадров длинной 375000 отсчетов, чтобы накопить достаточно информации.

```
rx_sig = zeros(…);
for i …
    rx_sig(…:…) = rx();
end
```

Далее начинается обработка принятого сигнала. Первым этапом обработки необходимо отфильтровать принятый сигнал для выделения полезного сигнала. Воспользуйтесь фильтром нижних частот (ФНЧ) с квадратным окном. В MATLAB для подобных задач используется свертка сигнала с коэффициентами фильтра. Вычислите коэффициенты фильтра при помощи встроенной функции `fir1`, входными аргументами которой является количество коэффициентов фильтра (возьмите равным 200) и нормированная частота среза фильтра. После получения коэффициентов фильтра выполните свертку с принятым сигналом отдельно для синфазной и квадратурной части сигнала.

```
numtaps = 200; % Число коэф фильтра
b = fir1(…, …);
filter_sig_I = conv(… ,b,'same'); % Фильтрация синфазной
filter_sig_Q = conv(…,b,'same'); % Фильтрация квадратурной
```

После выполнения фильтрации, большая часть спектра уже не представляет интереса. Выполните децимацию при помощи встроенной функции `resample` для синфазной и квадратурной части сигнала по отдельности. Входные аргументы функции: сигнал, коэффициент интерполяции, коэффициент децимации. Таким образом при отсутствии интерполяции, коэффициент интерполяции равняется 1.

```
decim_sig_I = resample(…, 1, …);
decim_sig_Q = …
```

Далее демодуляция принятого сигнала. Для начала необходимо выполнить вычисление дифференциалов, которые необходимы для выполнения демодуляции. Выполните вычисление дифференциалов для синфазной и квадратурной части при помощи встроенной функции MATLAB `diff`, аргументом которой является децимированный сигнал,

затем умножьте его на полосу сигнала после децимации и добавьте в начало 0. Добавление нулевого элемента в массив обусловлено тем, что необходимо выровнять размерности массива, т.к., после вычисления дифференциалов длинна массива сократится на 1 элемент.

```
dI_Rx = [0, diff(…)*fsDecim];
dQ_Rx = […];
```

Далее при помощи имеющихся дифференциалов необходимо выполнить демодуляцию. Демодуляция выполняется по следующей формуле:

$$W(t) = \frac{B'(t) \cdot A(t) - B(t) \cdot A'(t)}{A^2(t) + B^2(t)}, \quad (2.5)$$

где $A(t)$ – децимированный сигнал синфазной составляющей;

$A'(t)$ – вычисленные дифференциалы децимированного сигнала синфазной составляющей;

$B(t)$ – децимированный сигнал квадратурной составляющей;

$B'(t)$ – вычисленные дифференциалы децимированного сигнала квадратурной составляющей.

Затем полученный результат необходимо поделить на девиацию частоты, которая указана в таблице 1, чтобы извлечь исходный информационный сигнал.

```
W_t_demod = (dQ_Rx.*… - ….*…)./(….^2 + ….^2);
Sm_demod = ;
```

После того, как была выполнена демодуляция сигнала, из него необходимо выделить аудио сигнал. Выделение аудио сигнала осуществляется также при помощи ФНЧ. В таблице 1.1 представлена полоса аудио сигнала, используемого в FM радиостанциях. Нормируйте данную частоту относительно полосы сигнала после децимации и вычислите коэффициенты фильтра, предназначенного для выделения аудиосигнала. Выполните фильтрацию.

На рисунке 2.1 представлена структура FM кадра, после выполнения демодуляции. На нем видно, что моноканал расположен в полосе от 30 Гц до 15 кГц, там, где обе звуковые дорожки соединены в одну, отсюда и следует величина полосы моноканала. Стерео каналы же располагаются от 23 до 38 кГц и от 38 до 53 кГц.

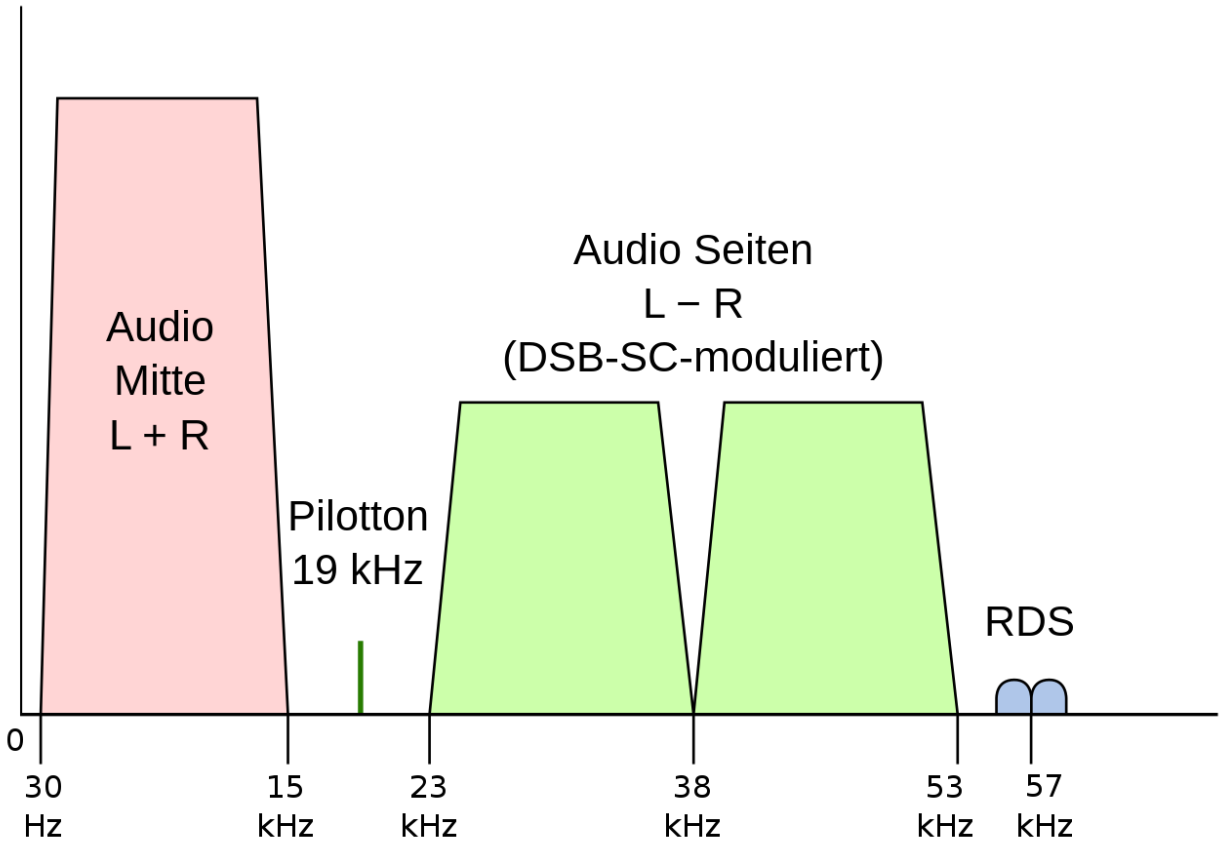


Рисунок 2.1 – Структура FM кадра

Нормирование полосы фильтра для выделения аудиосигнала выполняется относительно частоты Найквиста, которая равна половине полосы сигнала, в данном случае, относительно полосы сигнала после децимации. Следовательно, разделите полосу фильтра на частоту Найквиста. Сформируйте характеристику фильтра и выполните фильтрацию.

```
fcutoff_mono_norm = …/…;
b_mono = fir1(…,…);
Sm_demod_mono = …;
```

Так как сигнал имеет частоту дискретизации в 200 кГц, а частота дискретизации звуковой карты персонального компьютера (ПК) составляет 44/48 кГц, то полученный сигнал необходимо согласовать с звуковой картой ПК. Для этого выполните децимацию сигнала в 200 раз и интерполяцию в 48.

```
sound_demod = resample(…, …, …);
```

Воспроизведите аудио сигнал полученный с FM радиостанции. Для этого воспользуйтесь встроенной функцией MATLAB `sound`, входными аргументами которой являются реальная часть полученного звукового сигнала, умноженная на громкость (укажите равной двойке), а вторым аргументом частота дискретизации звуковой карты ПК.

```
v = 2; % громкость
sound(v*…,…)
```

Вы должны отчетливо слышать музыку или речь. Сохраните сигнал `sound_demod` с расширением mat и приложите его к отчету работы

```
save('ВашеИмя_Фамилия.mat', 'sound_demod')
```

Создайте сетки частот для отображения графиков спектров сигнала из эфира и отфильтрованного сигнала, а также сетку для отображения графиков спектра децимированного и демодулированного сигналов. Это необходимо, чтобы сопоставить графически положение сигналов на определенной частоте, а также обнаружить ошибки в процессе обработки, при их наличии.

```
freq_axis1 = ((0:NumSamp-1)*(fs/NumSamp))./1e3;
freq_axis2 = ((…)*(…))./1e3;
```

Затем постройте следующие графики:

1) Спектр принятого сигнала в логарифмическом масштабе от частоты.

2) Спектр принятого фильтрованного сигнала в логарифмическом масштабе от частоты. Постройте на этом же графике АЧХ ФНЧ в логарифмическом масштабе, применяемого в начале работы. Не забудьте выполнить нормирование АЧХ ФНЧ к спектру сигнала, иначе АЧХ ФНЧ и спектр сигнала будут построены в разных масштабах.

3) Спектр децимированного сигнала в логарифмическом масштабе от частоты

4) Спектр демодулированного сигнала в логарифмическом масштабе от частоты. Постройте на этом же графике АЧХ ФНЧ в логарифмическом масштабе, используемого для выделения аудиосигнала

Используйте функцию freqz для получения АЧХ фильтров. Если вы забыли, как она работает, выделите функцию и нажмите F1, чтобы открыть документацию. На рисунке 2.2 приведен пример необходимых графиков.

```
 [h,f] = freqz(___,n,'whole',fs); % Пример вычисления АЧХ фильтра

figure(1)
subplot(221)
plot(freq_axis1, 10*log10(abs(fft(rx_sig))) )
xlabel('kHz'); title('RX Signal')
subplot(222); hold on
plot(… )
xlabel('kHz'); title('Filtered Signal')
subplot(223)
plot(…)
xlabel('kHz'); title('Decim Signal')
subplot(224); hold on
plot(…)
xlabel('kHz'); title('Demod Signal')
```

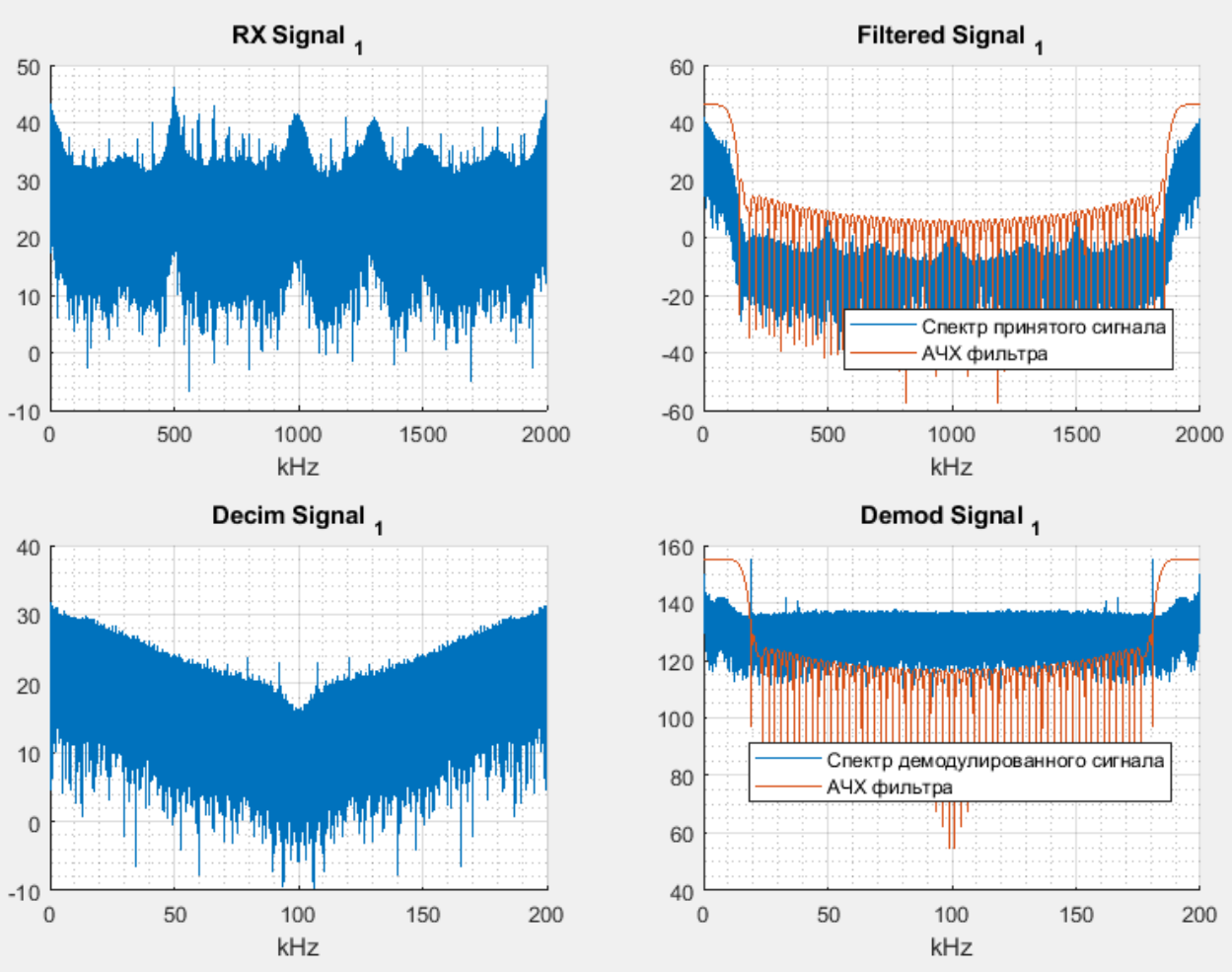


Рисунок 2.2 – Пример графиков

Самостоятельно создайте сетку частот для моно аудиосигнала, приведенного к частоте дискретизации звуковой карты. Постройте в логарифмическом масштабе данный сигнал от вычисленной вами сетки.

3. Примеры контрольных вопросов

1) В какой полосе принимался сигнал из FM эфира?
2) Приведите и поясните структуру спектра FM сигнала в матлабе.
3) Как выполняется демодуляция FM сигнала?

4. Требования к оформлению отчета

1) Введение;
2) Укажите частоту на которой осуществлялся прием FM сигнала;
3) Ход выполнения работы. Алгоритм демодуляции сигнала;
4) Привести рисунок спектра сигнала из SDRsharp, который вы принимаете;
5) Привести в отчете рисунки в логарифмическом масштабе:
   a. Спектр сигнала из эфира от частоты;
   b. Спектр сигнала из эфира после фильтрации от частоты и АЧХ ФНЧ на одном графике;
   c. Спектр сигнала из эфира после фильтрации и децимации от частоты;

d. Спектр демодулированного сигнала от частоты, а также АЧХ ФНЧ для аудиосигнала на одном графике;

e. Спектр аудиосигнала приведенного к частоте звуковой карты от частоты.

6) Выводы по проделанной работе;

7) Листинг программы. Выводимые в результате выполнения кода графики, должны полностью совпадать с графиками в отчете, за исключением формы принимаемых сигналов.

8) Приложите к отчету сохраненный аудиофайл в формате mat.

Работа № 4

«Поиск сигнала в эфире (несущая, полоса) и его прием»

Цель работы: Обнаружить и принять сигнал из эфира.
Задачи работы:
1) При помощи программы SDRSharp обнаружить сигнал в эфире;
2) Определить несущую частоту сигнала;
3) Определить полосу сигнала;
4) Осуществить прием сигнала из эфира.

1. Ход выполнения работы

В данной работе необходимо обнаружить сигнал при помощи SDRSharp и принять OFDM сигнал в среде разработки MATLAB при помощи RTL SDR.

В таблице 1 указаны параметры сигнала, которые необходимо определить при помощи SDRSharp.

Таблица 1.1 – Параметры сигнала

Параметр	Обозначение	Пояснения
Полоса сигнала	fs	Полоса сигнала радиостанции
Несущая частота радиостанции	fc	Несущая частота радиостанции

1.1 Поиск OFDM сигнала при помощи SDRSharp

Откройте SDRSharp и изменяйте несущую частоту в диапазоне от 900 МГЦ до 1,1 ГГц. Перемещаясь по данному диапазону найдите OFDM сигнал в спектре. Установите параметр усиления такой, чтобы OFDM сигнал был четко виден на спектрограмме. Начните с 10 дБ и в зависимости от дальности до передающего устройства увеличивайте этот параметр. Пример установки усиления показан на рисунке 1.1. Внешний вид спектра OFDM сигнала представлен на рисунке 1.2.

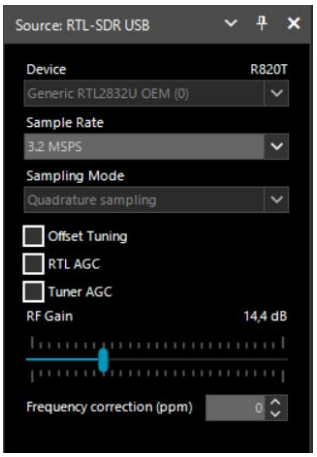


Рисунок 1.1 – Настройка усиления в SDRSharp

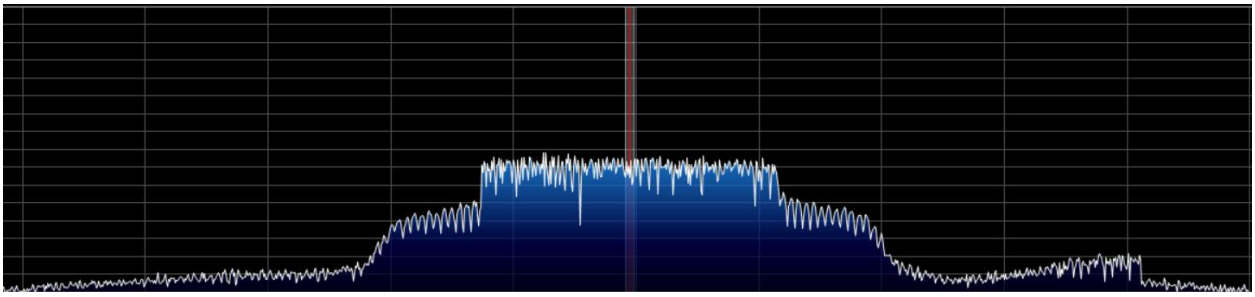


Рисунок 1.2 – Внешний вид спектра OFDM сигнала

После того, как был обнаружен OFDM сигнал точно определите его несущую частоту и полосу. Границы сигнала для определения его полосы указаны на рисунке 1.3.

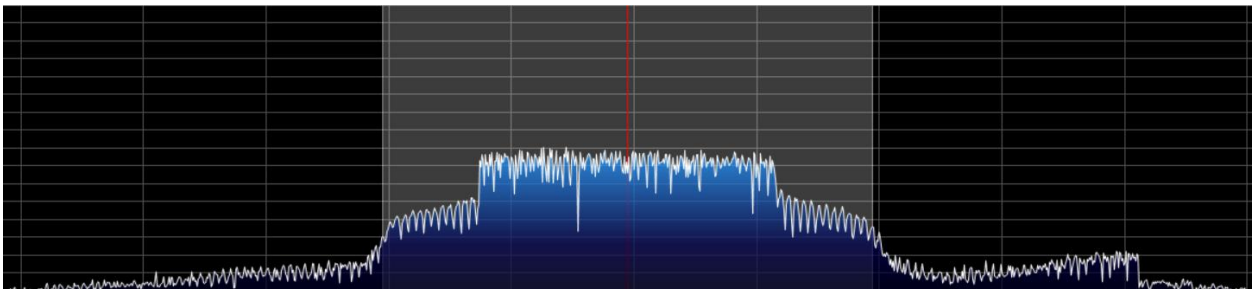


Рисунок 1.3 – Определение полосы сигнала

Затем, зная несущую частоту и полосу сигнала перейдите в MATLAB и выполните настройку RTL SDR приемника в MATLAB.

1.2 Настройка RTL SDR в MATLAB

Первым этапом настройки RTL SDR в MATLAB необходимо создать объект класса ‘comm.SDRRTLReceiver’ отвечающий за инициализацию RTL SDR как приемника и позволяющий выполнять его настройку.

Для создания объекта класса необходимо выполнить следующее действие:

```
rx = comm.SDRRTLReceiver;
```

Затем, обращаясь к свойствам объекта rx осуществите настройку вашего приемника. Свойство класса `CenterFrequency` отвечает за установку несущей частоты. Установите его значение равное значению оцененной вами несущей частоты.

Свойство `SampleRate` отвечает за установку полосы сигнала в которой будет осуществляться прием. Установите его в соответствие с определенной полосой сигнала.

Укажите `SamplesPerFrame` равное 375000.

Установите значение `EnableTunerAGC` равное 0.

Установите параметр `TunerGain` равный усилению, которое было установлено в SDRSharp для того, чтобы выделить OFDM сигнал из спектра.

Установите `OutputDataType` как `'double'`.

```
rx.CenterFrequency = …;
rx. = …;
rx. = …;
rx. = …;
rx. = …;
rx. = 'double';
```

После того, как была выполнена настройка RTL в среде MATLAB можно осуществлять прием сигнала из эфира. Также нужно учесть, что сигнал принимается на 0 частоте, для удобства работы с принятыми сигналами, необходимо выполнить разворот спектра. Для этого необходимо выполнить следующую команду:

```
rx_sig = rx();
rx_sig(2:2:end) = -rx_sig(2:2:end);
```

где `rx_sig` массив в который запишется то количество отсчетов, которое было указано в свойстве `SamplesPerFrame`. Инвертирование каждого 2 отсчета эквивалентно развороту спектра сигнала

После выполнения всех представленных выше действий вами будет получен массив данных в который записан OFDM сигнал из эфира.

К текущей работе необходимо будет возвращаться для того, чтобы считывать сигнал из эфира с новыми параметрами настройки RTL для достижения должного качества принимаемого сигнала и его дальнейшей обработки.

Постройте график принятого сигнала, воспользовавшись функцией plot и взяв от сигнала абсолютные значения.

На рисунке 1.4 и 1.5 представлены графики принятого сигнала. На рисунке 1.4 вы видите несколько последовательно принятых кадров. На рисунке 1.5 показано приближение одного принятого кадра.

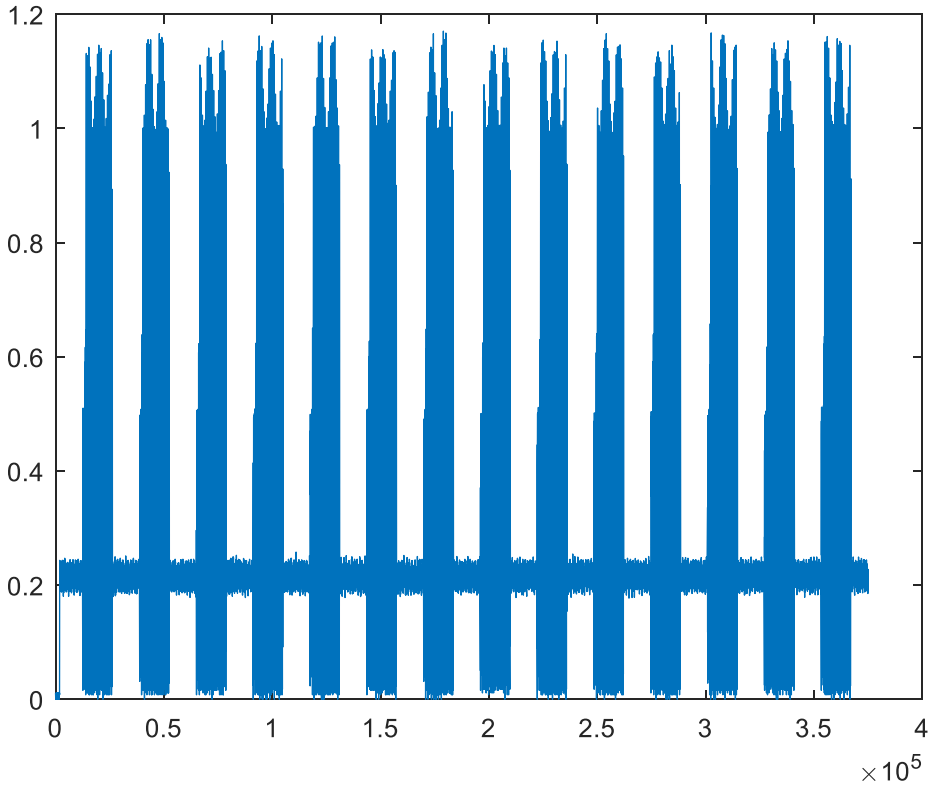


Рисунок 1.4 – Принятый OFDM сигнал из эфира.

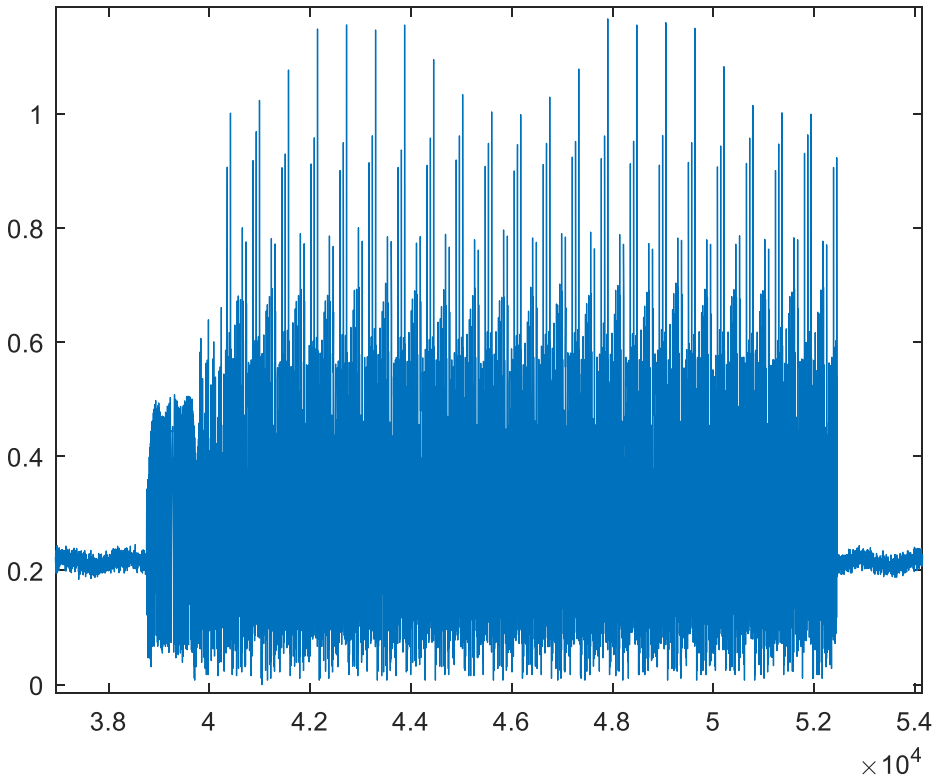


Рисунок 1.5 – Приближение принятого OFDM сигнала из эфира

На рисунке 1.4 видно, что в принятый сигнал при помощи RTL SDR в размере 375 тысяч отсчетов укладывается несколько кадров переданного сигнала. Следовательно, нет необходимости использовать большее количество отсчетов принимаемого сигнала.

На рисунке 1.6 показано начало передаваемого кадра.

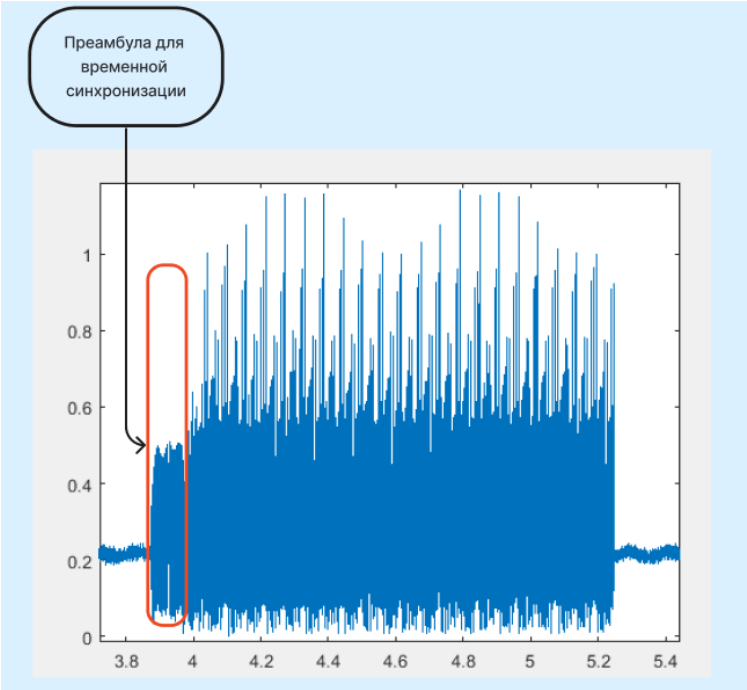


Рисунок 1.6 – Расположение преамбулы в начале OFDM кадра

2. Контрольные вопросы к работе

1) В какой полосе принимался OFDM сигнал из радиоэфира?
2) Какая несущая частота OFDM сигнала?
3) Какое максимальное количество отсчетов за раз записывает RTL SDR?

3. Требования к оформлению отчета

1) Введение;
2) Поиск OFDM сигнала при помощи SDRSharp;
3) Определение входных параметров сигнала;
4) Алгоритм настройки RTL SDR в MATLAB;
5) Рисунки:
   a. Спектр найденного OFDM сигнала в SDRSharp;
   b. График принятого сигнала в MATLAB;
   c. График принятого сигнала с выделением одного кадра MATLAB.
6) Выводы;
7) Листинг программы. Выводимые в результате выполнения кода графики, должны полностью совпадать с графиками в отчете, за исключением формы принимаемых сигналов.

Работа № 5

«Формирование опорного сигнала из предоставленной последовательности. Временная синхронизация принятого сигнала»

Цель работы: разработать модель корреляционного приемника.

Задачи работы:

1) Сформировать опорный сигнал;
2) Вычислить корреляцию между принятым сигналом и опорным;
3) Найти начало кадра;
4) Выделить полезный сигнал из принятого сигнала.

1. Теоретическая часть

Временная синхронизация является одним из основных этапов приема и обработки сигнала. Приемник не знает точное время, когда к нему на вход поступит сигнал, предназначенный для него. Следовательно, приемнику необходимо прослушивать эфир в поисках сигнала, предназначенного для него. Для обнаружения сигнала в эфире используется преамбула.

Преамбула, это своего рода «ключ», который формируется в передатчике и ставится в начало сигнала, а приемник знает этот «ключ» и ищет его в эфире, зная, что за ним будет следовать сигнал, предназначенный для этого приемника.

Еще одно назначение временной синхронизации – обнаружение и компенсации временной задержки из-за многолучевого распространения, где присутствуют переотраженные сигналы, задержанные во времени.

Обнаружение преамбулы в принятом массиве осуществляется при помощи взаимокорреляционной функции (ВКФ). Отсюда следует и название «корреляционный приемник». ВКФ показывает количественный уровень взаимосвязи принимаемого сигнала с опорным сигналом (преамбулой). Полученный уровень называется значением или коэффициентом корреляции. В случае превышения коэффициентом корреляции установленного порога, принимается решение о наличии исходного сигнала в канале передачи. Расчет коэффициента корреляции осуществляется по формуле:

$$1/E_s \int_0^T S(t)S_{on}(t)dt = r, \tag{1.1}$$

где $S(t)$ – принятый сигнал;

$S_{on}(t)$ – опорный сигнал;

r – коэффициент корреляции.

Корреляционная функция – математический инструмент, позволяющий определить наличие связи между двумя величинами и ее силу.

Разделяют две корреляционные функции: автокорреляционная функция (АКФ) и взаимная корреляционная функция (ВКФ). Корреляция позволяет оценить схожесть двух сигналов, что позволяет оценить наличие сигнала в канале и синхронизироваться по времени, а также выполнить частотную синхронизацию, она будет изучена в следующей работе.

АКФ – показывает, как коррелированы две последовательности сигнала, отстоящие друг от друга на N количество отсчетов. Чаще всего применяется для выделения сигнала из шума в канале связи и определения спектральных характеристик сигнала.

ВКФ – показывает насколько сильно коррелируют между собой два сигнала и используется для временной синхронизации.

2. Ход выполнения работы

Входные параметры представлены в таблице 2.1.

Таблица 2.1 – Входные параметры

Параметр	Обозначение	Значение	Пояснение
Коэффициент расширения спектра	SF	10	Берется равным 10 чтобы получить длительность равную 1024 отсчета
Полоса сигнала	BW	1	Принимается равной единице для удобства
Длина кадра	len	13696	Длина принимаемого кадра из эфира

Продолжите написание кода из прошлой работы. В данной работе вам необходимо сформировать опорный сигнал. Опорным сигналом, в качестве преамбулы используется сигнал с линейной частотной модуляцией (ЛЧМ). Сформируйте ЛЧМ сигнал.

$$sigLCM = \exp\left(-1i \cdot 2 \cdot pi \cdot \frac{m * t.^2}{2}\right), \tag{2.1}$$

где m – коэффициент изменения частоты, рассчитывается как:

$$m = BW / Ts, \tag{2.2}$$

где BW – полоса сигнала;

Ts – длительность сигнала, которая рассчитывается как:

$$Ts = \frac{2^{SF}}{BW}, \tag{2.3}$$

где SF – коэффициент расширения спектра

t – полоса времени одного импульса, которая определена:

$$t = 0 : ts : Ts - ts, \tag{2.4}$$

где ts - время дискретизации и равно:

$$ts = \frac{1}{BW}. \tag{2.5}$$

Сформируйте данный сигнал, а затем представьте его в виде вектор-столбца, а также умножьте его на константу 0,0473, для нормирования к уровню преамбулы в принятом сигнале.

```
SF=…;
BW=1;
Ts=…;
ts=…;
m=…;
t=…;
syncSig=…;
```

После того, как вами была сформирована преамбула, расширьте ее длину до размера принятого сигнала, посредством добавления нулей в конец преамбулы до длины сигнала из таблицы 2.1. Учтите, что указанная размерность в таблице 2.1 учитывает, как длину символов, так и длину преамбулы, поэтому необходимо из данной длины вычесть длины преамбулы.

```
syncSigAdvanced = [syncSig;…];
```

Выполните нахождения ВКФ для определения схожести принятого сигнала с сформированной преамбулой. Для этого воспользуйтесь функцией `xcorr`, входными аргументами которой является ваш принятый сигнал и преамбула расширенная до длинны принятого сигнала.

```
[…, …] = xcorr(…,…);
```

Функция возвращает 2 значения. Первое – массив амплитудных значений ВКФ, а второе – массив задержек, с помощью которых вычисляется положение принятого символа.

Постройте график модуля корреляционной функции. Создайте новую фигуру и постройте график при помощи функции `plot` входным аргументом которой будет модуль массива амплитудных ВКФ. Для получения абсолютных значений необходимо воспользоваться функцией `abs` входным аргументом которой является массив (в данном случае массив ВКФ).

```
figure;
plot(…(…));
```

Корреляция должна иметь, примерно следующий вид, представленный на рисунке 2.1.

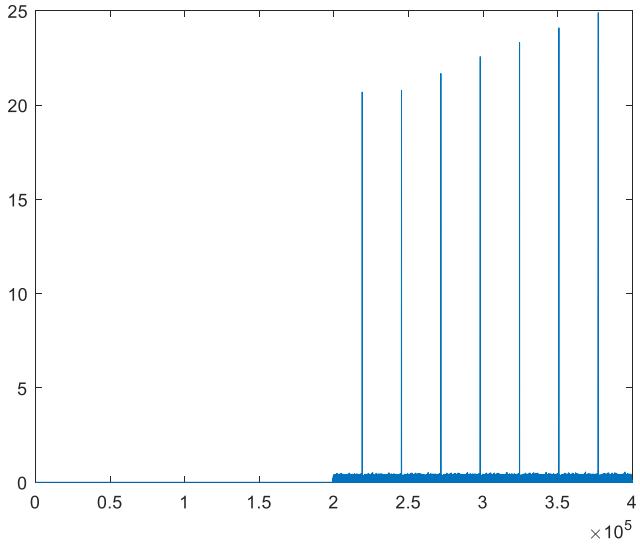


Рисунок 2.1 – График ВКФ

Как видно из рисунка 2.1, наличие нескольких пиков корреляции означает, что принятый массив содержит несколько переданных сигналов, что в рамках текущей работы является достоверным. Так как пиков несколько, последний пик корреляции может иметь максимальное значение. В этом случае приемник может прекратить свою работу до того, как полностью примет сигнал. Тогда необходимо выбрать любой другой пик корреляции, но не последний. Рекомендуется взять первый пик корреляции, за которым следует первый принятый кадр. Нахождение индекса пика корреляции осуществляется путем поиска максимума ВКФ. Для этого необходимо воспользоваться функцией `max` входными аргументами которой являются абсолютные значения ВКФ, а выходными аргументами значение пика корреляции и его номер. Однако вам необходимо задаться границами для того, чтобы искать пик в области первого кадра. Важно! Результирующий массив ВКФ в два

раза длиннее, чем сигнал. Учтите это при поиске начала преамбулы. Задайте границы поиска максимума пика корреляции следующим образом: от длины сигнала расширенной преамбулы с добавлением единицы (ищем пики с первого элемента второй половины) до длинны расширенной преамбулы с добавлением длинны предаваемого кадра. Длину передаваемого кадра можно определить через индексы двух соседних пиков корреляции. Или же взять из таблицы 1. Вы также можете искать пик корреляции используя только половину принятого массива, а вырезать обнаруженный сигнал уже из полного массива. Другой способ заключается в умножении результата корреляции на весовое окно, чтобы выделить пики в начале или центре.

```
 [value, number] = max(…(…(…:…)));
```

Однако, такая манипуляция хоть и позволяет выбрать пик корреляции первого кадра, но вы должны учесть, что необходимо отбросить преамбулу, так как она больше не играет роли. Добавьте к номеру пика корреляции длину нерасширенной преамбулы. Полученное значение является индексом. В массиве задержек (второй выходной аргумент функции `xcorr`) найдите число, размещенное на данном индексе. Полученный результат и будет началом кадра. Вырежьте из принятого массива переданный кадр начиная с найденного номера начала кадра. Длинна кадра указана в таблице 1, за исключением того, что от нее необходимо отнять длину преамбулы, которую убрали на предыдущем этапе.

```
number = number+…;
signalStart = lag(number);
Frame = sigFilter(signalStart+1:…);
```

Тем самым будет выполнена временная синхронизация. В реальных системах связи, на данном этапе осуществляется еще и формирование строб сигналов, для разделения принятого сигнала на OFDM символы, однако для небольшого удобства в модели, сигнал на OFDM символы будет поделен после частотной синхронизации, которая является темой следующей работы.

Постройте график пиков корреляции, Фурье преобразование синхронизированного сигнала в логарифмическом масштабе и временную реализацию реальной части синхронизированного сигнала.

На рисунке 2.2 ниже приведены примеры графиков, которые необходимо построить в ходе выполнения работы.

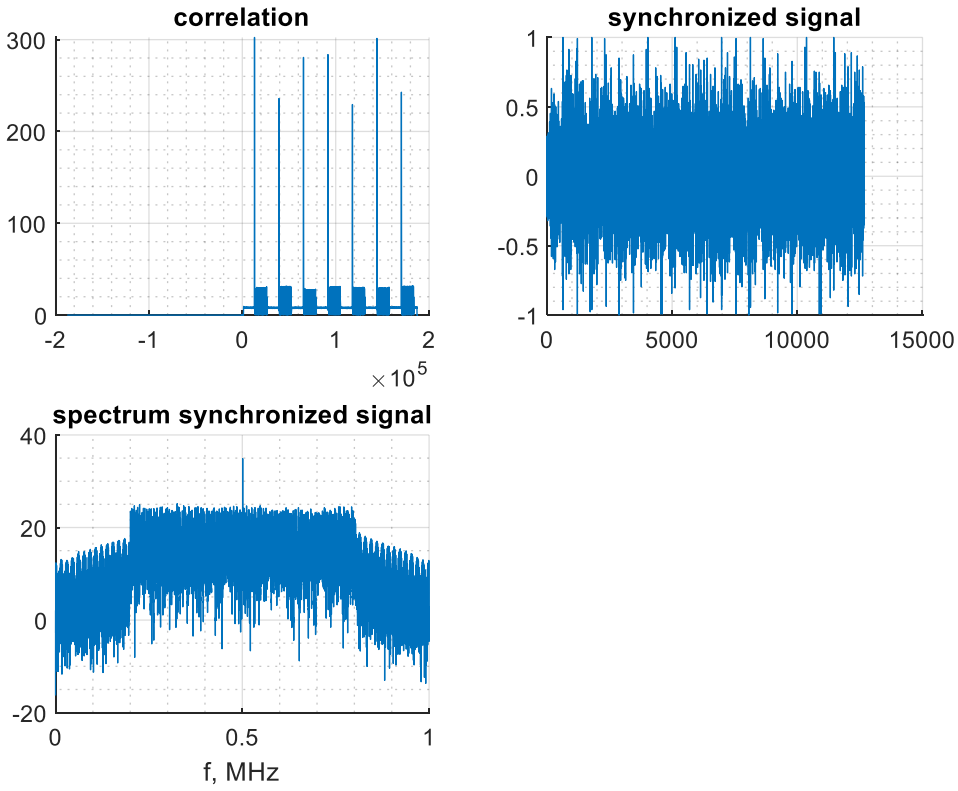


Рисунок 2.1 – Примеры графиков

3. Примеры контрольных вопросов

1) Для чего нужна временная синхронизация?
2) Для чего нужна преамбула в сигнале?

4. Требования к содержанию отчета

1) Введение;
2) Ход выполнения работы;
3) Рисунки:
   a. График корреляции;
   b. График временной реализации синхронизированного принятого сигнала в логарифмическом масштабе;
   c. Спектр синхронизированного принятого сигнала.
4) Выводы;
5) Листинг программы. Выводимые в результате выполнения кода графики, должны полностью совпадать с графиками в отчете, за исключением формы принимаемых сигналов.

Работа № 6

«Частотная синхронизация принятого сигнала»

Цель работы: разработать модель оценки смещения несущей частоты и выполняющую его компенсацию.

Задачи работы:

1) Определить фазовый набег по преамбуле;
2) Определить смещение несущей частоты по фазовому набегу;
3) Компенсировать смещение несущей частоты.

1. Ход выполнения работы

На рисунке 1.1 представлена блок-схема выполняемого алгоритма.

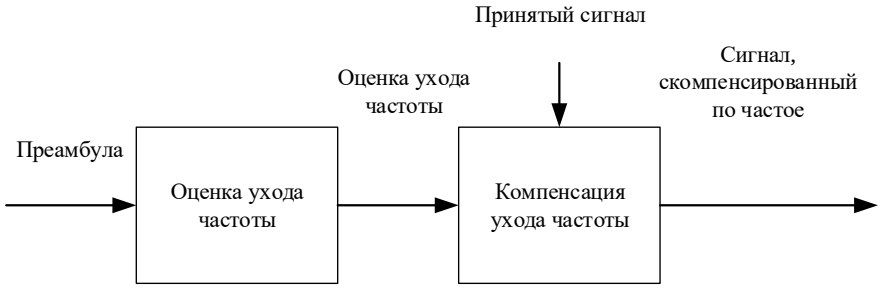


Рисунок 1.1 – Блок схема, реализуемого алгоритма

После того, как была выполнена временная синхронизация сигнала, необходимо выполнить оценку смещения несущей частоты и компенсировать найденное смещение.

Уход несущей частоты возникает из-за несовпадения несущей частоты переданного сигнала и частоты гетеродина в приемнике. Данное рассогласование обусловлено двумя факторами, влияющими на сигнал:

- Разность частот между генераторами передатчика и приемника;
- Эффектом Доплера.

Разность частот в генераторах передатчика и приемника обуславливается тем, что невозможно создать два генератора с абсолютно идентичными параметрами.

Эффект Доплера – изменение частоты сигнала в результате движения передающего или принимающего устройства относительно друг друга. Сдвиг частоты напрямую связан с разностью скорости движения между передающим и принимающим устройством и описывается по формуле 1.1.

$$f = (1 + \frac{\Delta v}{c}) \cdot f_0, \tag{1.1}$$

где f_0 – частота излучаемого сигнала;

Δv – разность скоростей между передающим и принимающим устройством;

c – скорость света;

f – наблюдаемая частота.

В таблице 1.1 указаны параметры OFDM символов, которые понадобятся вам в этой и дальнейших работах.

Таблица 1.1 – Параметры сигнала

Параметр	Обозначение	Значение
Длина OFDM символа	NFFT	512

Продолжение таблицы 1.1

Длина циклического префикса	CPsize	1/8 от длины OFDM символа
Длина защитных интервалов	gi_size	102

Создайте функцию, входными аргументами которой является принятый сигнал после временной синхронизации `Frame`, размер циклического префикса, размер одного OFDM символа и частота дискретизации. Или же можете выполнить реализацию синхронизации в основном коде. При оценке частотного сдвига необходимо найти фазовый набег, который возникает при частотном рассогласовании приемника и передатчика. Для этой цели в принятом сигнале содержится еще одна преамбула, предназначенная для частотной синхронизации. Преамбула в передатчике сформирована таким образом, что содержит внутри себя два одинаковых OFDM символа, длина каждого из которых в 2 раза меньше чем длина обычного OFDM символа. Вам необходимо отделить от преамбулы (первого OFDM символа) циклический префикс и разделить преамбулу на две половины.

```
halfOne = Frame();
halfTwo = Frame();
```

Затем необходимо вычислить корреляцию между полученными половинами, каждая из которых идентична другой. Выполните корреляцию при помощи встроенной функции `xcorr`, первым и вторым аргументом которой являются вторая и первая половина, соответственно. Обязательно используйте именно такой порядок, так как это повлияет на знак найденного ухода фазы. Затем найдите пик корреляции, воспользовавшись встроенной функцией `max` и зафиксируйте его уровень.

```
corrCP = xcorr(…);
value = max(…);
```

Далее необходимо вычислить аргумент пика корреляции `dPhi`, что позволит узнать смещение фазы (в радианах) за длительность одного OFDM символа из второй преамбулы. Нужно учесть, что длина символов в преамбуле в два раза меньше чем в основном сигнале. Для этого необходимо воспользоваться встроенной функцией `angle`, входным аргументом которой является найденное значение уровня корреляции.

Для перехода от смещения фазы к смещению частоты необходимо определить длительность символа (в секундах) `tau`. Разделите длину одного OFDM символа из преамбулы на полосу принятого сигнала, а затем разделите найденный аргумент на полученную длительность умноженную на 2π.

```
dPhi = angle(…);
tau = …/Fs;
calculationFrequency = …/…;
```

Вы получили значение смещения частоты несущей. Далее необходимо вычислить на сколько радиан сдвигается каждый отчет символа во временной области и устранить это смещение. Для этого разделите ранее найденный аргумент на длительность OFDM символа из преамбулы (количество отсчетов). Это значение будет являться оценкой ухода фазы за один отсчет. Введите найденную оценку в `Frame` с противоположным знаком, увеличивая ее значение с каждым отсчетов, согласно свойству преобразования Фурье о частотном сдвиге. Формула для выполнения компенсации имеет вид:

$$Frame(i)' = Frame(i) \cdot \exp(j \cdot i \cdot - f_{\Delta\varphi}), \quad (1.2)$$

где *Frame*(*i*) – *i*-й элемент массива `Frame` до компенсации смещения несущей;

Frame(*i*)' – *i*-й элемент массива `Frame` после компенсации смещения несущей;

$f_{\Delta\varphi}$ – оценка смещения частоты на 1 отсчет;

exp – функция возведения экспоненты в степень.

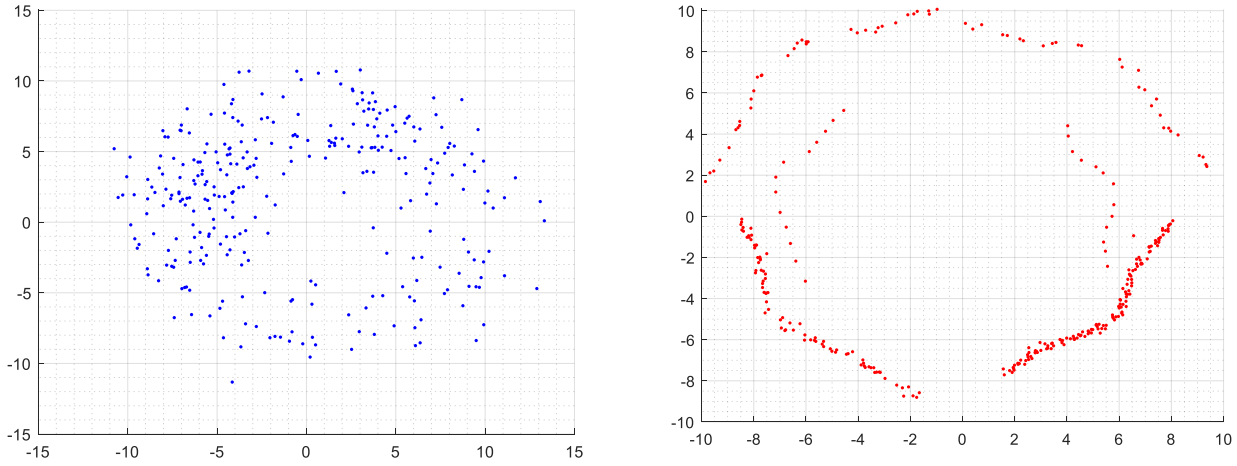
```
dPhiFreq = …/…;

for j=…
    output(j)=…;
end
```

Постройте графики созвездий принятого сигнала до компенсации ухода несущей частоты и после. Для наглядности возьмите последний OFDM символ принятого массива, поскольку за длительность всего сигнала набег фазы будет существенным и созвездие повернется сильнее всего именно в последнем OFDM символе. Для этого выполните вырезание из массива последнего OFDM символа, удалите защитные интервалы и центральную несущую в спектре для выделения только информационных поднесущих и постройте созвездие до частотной синхронизации. Проделайте те же самые операции над сигналом после частотной синхронизации, также постройте созвездие.

```
LastOFDM = fft(Frame(end-NFFT+1:end));
LastOFDMHalf1 = LastOFDM(…);
LastOFDMHalf2 = LastOFDM(…);
graphLastOFDM = […;…];
scatterplot(graphLastOFDM)
```

Пример результата работы частотной синхронизации представлен на рисунке 1.2.



а) б)

Рисунок 1.2 – Сигнальные созвездия сигнала до компенсации смещения несущей частоты (а) и после компенсации (б)

Полученные вами созвездия могут отличаться по причинам того, что канал реален и его условия случайны.

2. Примеры контрольных вопросов

1) Из-за чего возникает сдвиг несущей частоты?

2) К чему приводит сдвиг несущей частоты?
3) Что является результатом компенсации частотного сдвига?

3. Требования к содержанию отчета

1) Введение;
2) Ход работы;
3) Рисунки:
   a. Созвездие до частотной синхронизации;
   b. Созвездие после частотной синхронизации.
4) Выводы;
5) Листинг программы. Выводимые в результате выполнения кода графики, должны полностью совпадать с графиками в отчете, за исключением формы принимаемых сигналов.

Работа № 7

«Оценка АЧХ по пилотным поднесущим. Эквалайзирование»

Цель работы: оценить АЧХ по пилотным поднесущим. Разработать модель эквалайзера, способного выполнять эквалайзирование по пилотным поднесущим, осуществить эквалайзирование принятого сигнала и выполнить его демодуляцию.

Задачи работы:

- Оценить АЧХ канала по пилотным поднесущим;
- Реализовать модель эквалайзера;
- Осуществить эквалайзирование сигнала по пилотным поднесущим;
- Выполнить демодуляцию.

1. Ход выполнения работы

В данной работе необходимо реализовать 2 функции (модуля), отвечающие за оценку АЧХ канала и эквалайзирование сигнала, выполнить демодуляцию.
На рисунке 1.1 представлена блок-схема модулей системы связи, реализуемых в данной работе.

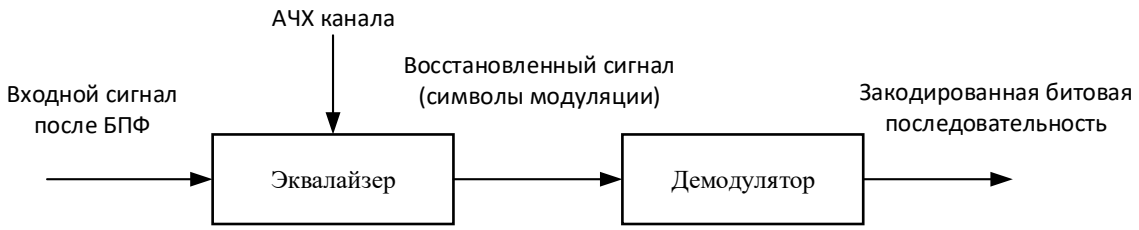


Рисунок 1.1 – Блок-схема, реализуемых модулей

Параметры, необходимые для реализации модулей представлены в таблице 1.

Таблица 1.1 – Параметры модулей

Параметр	Обозначение	Значение	Пояснения
Шаг пилотныйх поднесущих	rsStep	11	Шаг с которым пилотные поднесущие располагаются в OFDM символе (пилотная поднесущая через каждые 10 поднесущих)
Количество информационных поднес1ущих	nInfoSubcarrier	NFFT-2*gi_size	Количество поднесущих в OFDM символе, предназначенных для передачи информации и пилот тонов

1.1 Оценка АЧХ канала по пилотным поднесущим и эквалайзирование

Для начала удалите циклические префиксы из принятой последовательности символов, выполните Фурье преобразование каждого символа, удалите защитные интервалы и центральную поднесущую. Учтите, что самый первый OFDM символ – это преамбула для частотной синхронизации, у которой также имеется циклический префикс, перед выполнением вышеуказанных действий удалите ее вместе с префиксом из сигнала. Количество непосредственно информационных OFDM символов определите самостоятельно,

на основании длительности циклического префикса, длины OFDM символа и длины двух преамбул.

```
rsStep = 11;
nInfoSubcarrier = NFFT-2*gi_size;

num_syms = … ;
ofdm_syms = output(CPsize+NFFT+1:end);

% Вырезание символов
for i=1:num_syms
    ofdm_fft(i,:) = fft(ofdm_syms(…:…));
end

% Удаление защитных интервалов и центральной поднесущей
ofdm_fft(:,257)=…;
ofdm_data = ofdm_fft(:,…);
```

Массив `ofdm_data` представляет собой квадратную матрицу, в строках расположены символы, в столбцах поднесущие, в нем отсутствуют защитные интервалы и центральная поднесущая. Зная шаг пилотных поднесущих и количество поднесущих в OFDM символе сформируйте массив индексов пилотных поднесущих, это понадобится для того, чтобы извлечь пилотные поднесущие из принятого OFDM символа. Для этого заполните массив `rsSc` данными от 1 до `nInfoSubcarrier` с шагом `rsStep+1`.

```
rsSc(:,1) = …:…:…;
```

После того, как был получен массив номеров поднесущих на которых располагаются пилотные сигналы, сформируйте сами пилотные поднесущие. В данной работе, пилотные поднесущие представляют собой BPSK модуляцию, где каждая нечетная пилотная пилотная поднесущая имеет значение 1+0i, а каждая четная пилотная поднесущая – -1+0i. Сформируйте такую знакопеременную последовательность, а затем представьте ее в комплексном виде.

```
RS = zeros(length(rsSc),1);
RS(…) =…;
RS(…) = …;
RS = complex(…);
```

Полученный массив данных в ходе последнего действия и будет являться эталонной последовательностью для оценки АЧХ канала.

Следующим шагом необходимо выделить пилотные поднесущие из принятых символов. Для этого, пользуясь индексами пилотных поднесущих, поместите информацию из ранее обрезанных OFDM символов, размещённую на этих индексах в новый массив.

```
RxRS = ofdm_data(:,rsSc);
```

После того, как были выделены пилотные поднесущие принятого сигнала необходимо оценить АЧХ канала. Поэлементно разделите принятые пилотные поднесущие на эталонные пилотные поднесущие.

```
tft = …./…;
```

Из расположения пилотных поднесущих можно сделать вывод, что оценка АЧХ получена только на данных поднесущих, распределенных с определенным шагом, а не на всех поднесущих OFDM символа. В то время, как для эквалайзирования нам необходима оценка АЧХ на всех поднесущих. Для этого полученную оценку необходимо интерполировать до размеров количества информационных поднесущих OFDM символа. Для интерполирования необходимо разделить оценку на реальную и мнимую часть. Сделайте это, воспользовавшись встроенными функциями MATLAB для выделения синфазной и квадратурной составляющей `real` и `imag`, соответственно.

```
    tftEstimateReal = …;
    tftEstimateImag = …;
```

Выполните интерполяцию каждой составляющей по отдельности. Для этого воспользуйтесь встроенной функцией `interp1`, входными аргументами которой являются индексы размещения пилотных поднесущих, интерполируемый массив, массив с индексами, на которые необходимо поместить оценку, способ интерполяции (используйте `'linear'`), стратегию оценки точек, лежащих вне области определения (укажите `'extrap'`). Соответственно, массив индексов размещения пилотных поднесущих получен вами ранее, интерполируемый массив – полученная АЧХ OFDM символов, массив индексов получаемой оценки – это массив индексов от 1 до количества информационных поднесущих OFDM символа, имеющий вид [1; 2; 3; …; N], где N – количество информационных поднесущих, задайте данную величину самостоятельно. После интерполяции объедините синфазную и квадратурную составляющие в комплексный сигнал при помощи встроенной функции `complex`, входными аргументами которой являются синфазные и квадратурные составляющие, соответственно.

```
for i=1:num_syms
    tftInterpolirReal = interp1(…,…,…, 'linear', 'extrap');
    tftInterpolirImag = interp1(…,…,…, 'linear', 'extrap');
    tftInterpolir(i,:) = complex(…, …);
end
```

Постройте графики, АЧХ до интерполяции и после.

```
figure;
plot(…);
legend('…')
```

На рисунке 1.2 показан пример необходимых графиков.

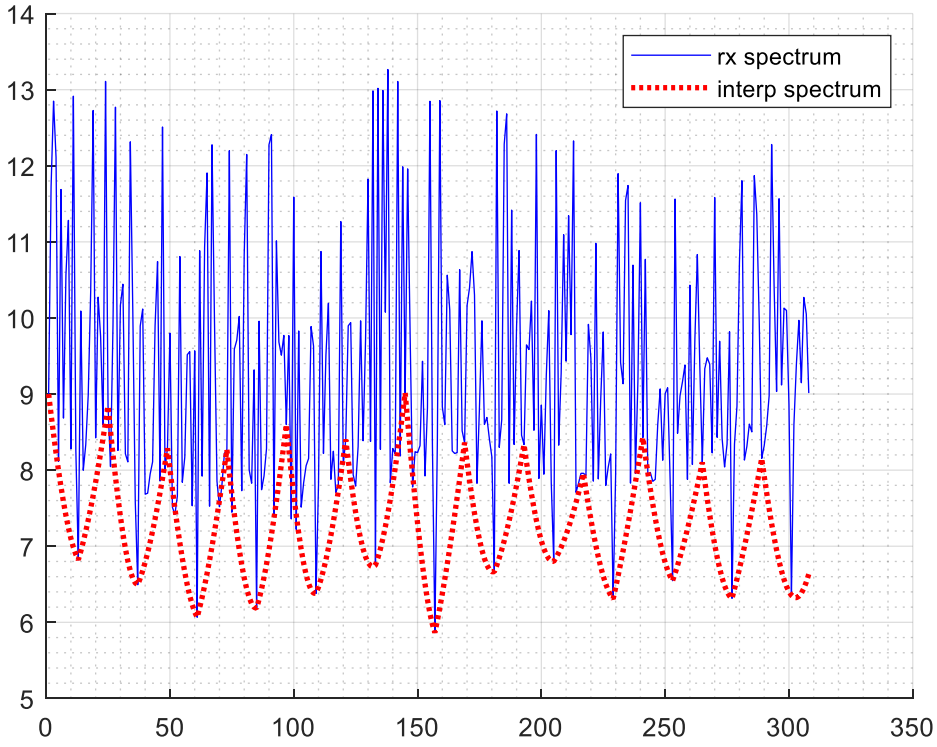


Рисунок 1.2 – Пример интерполяции спектра по пилотным поднесущим

1.2 Эквалайзирование

Воспользуйтесь интерполированным спектром, для восстановления АЧХ каждого из OFDM символов. После чего выделите из всего спектра только информационные поднесущие, пилотные удалите. Так как обрезанный спектр, полученный ранее, представляет собой квадратную матрицу, вам потребуется развернуть восстановленный АЧХ в вектор столбец или же строку, воспользуйтесь функцией reshape.

```
equ_syms = … ./…;
equ_syms(:,…) = …;
equ_syms_resh = … ;
```

Постройте диаграмму созвездия восстановленных данных используя функцию scatterplot, на рисунке 1.3 показан пример полученного созвездия.

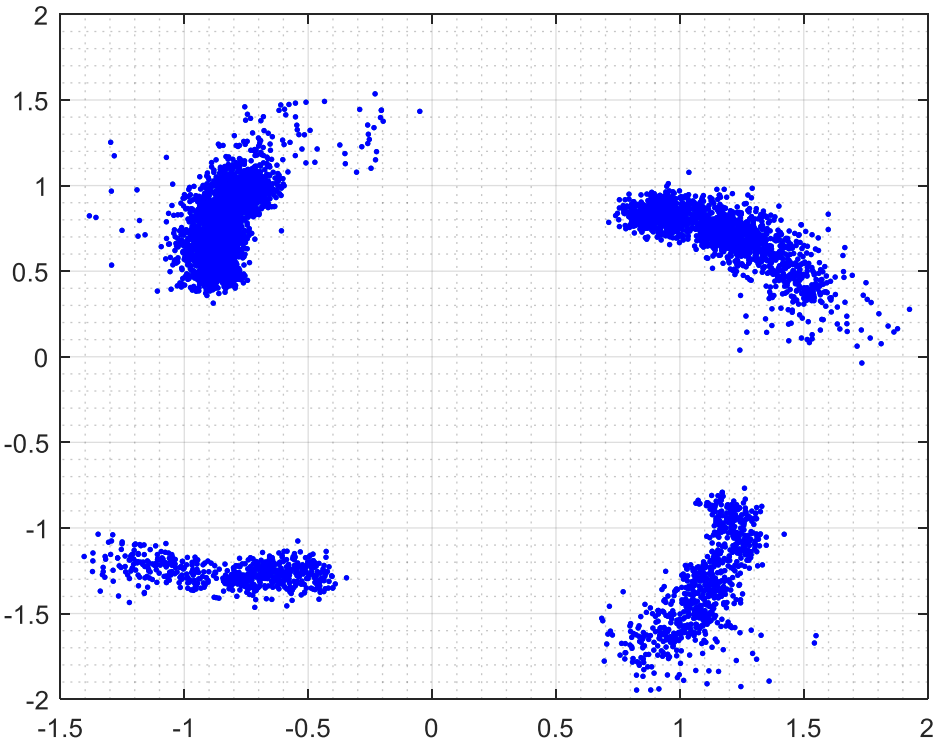


Рисунок 1.3 – Пример диаграммы созвездия после эквалайзера

Самостоятельно постройте диаграмму созвездия до эквалайзирования.

2. Примеры контрольных вопросов

1) Для чего необходимы пилотные поднесущие?
2) Как происходит оценка АЧХ канала по пилотным поднесущим?
3) С какой целью необходимо выполнять оценку АЧХ?

3. Требования к оформлению отчета

1) Введение;
2) Опишите используемый вами алгоритм оценивания АЧХ и эквалайзирования;
3) Рисунки:
   a. АЧХ OFDM символа до и после эквалайзирования;
   b. Созвездия OFDM символа до и после эквалайзирования;
4) Выводы;
5) Листинг программы. Выводимые в результате выполнения кода графики, должны полностью совпадать с графиками в отчете, за исключением формы принимаемых сигналов.

Работа № 8

«Декодирование и восстановление принятого сообщения»

Цель работы: разработать модель декодирования принятого сигнала и модель восстановления текстового сообщения.

Задачи работы:

1) Демодулировать сигнал;
2) Восстановить текстовое сообщение из эфира.

1. Ход выполнения работы

В данной работе необходимо реализовать 2 функции (модуля), отвечающие за демодуляцию и восстановление текстового сообщения, соответственно.

Таблица 1.1 – Параметры модулей

Параметр	Обозначение	Значение	Пояснения
Алфавит текстового сообщения	alphabet	`' abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ.,!'`	Знак «пробел» и латинские буквы нижнего и верхнего регистров, знаки препинания
Количество бит на информационный символ	N	$2^N > 56$	Размер одного знака, бит

Выполните демодуляцию принятого сигнала после того, как вы провели процедуру эквалайзирования. Используйте переменную equ_syms. На каждой из ее строк размещен один OFDM символ, выберите номер строки, соответствующий вашему варианту, это важно, так как каждый символ содержит уникальное сообщение. Тип демодулятора определите на основании полученной диаграммы созвездия. Это битовое сообщение вам необходимо декодировать, чтобы получить текстовое сообщение. Также укажите в функции демодулятора, что она должна возвращать биты.

```
data_demod = qamdemod(…, …,'OutputType','bit').';
```

Для выполнения знакового декодера необходимо реализовать функцию *symbolDecode* входным аргументом *input* которой будет битовая последовательность, полученная после функции декодирования, а выходом функции будет текст сообщения *output*. В функции необходимо задать длину кодового слова *N*, а также сам алфавит *alphabet*. Необходимые параметры представлены в таблице 1.

Размер кодового слова – количество бит на один знак и определяется исходя из соображений возможности закодировать данный знак. Следовательно, при длине алфавита в 56 знаков, необходимо выбрать такой размер кодового слова, чтобы все знаки могли быть

закодированы. Поэтому, необходимо найти такую степень двойки, чтобы неравенство $2^N > 56$ выполнялось.

```
function output = symbolDecode(input)
	%% Параметры символьного декодирования
	N         = …;
	alphabet  = […];
```

После того, как был создан алфавит и задан размер одного кодового слова необходимо закодировать алфавит. Сформируйте массив размерностью *Symb(N,length(alphabet))* и заполнить его при помощи встроенной функции *int2bit*. Третий аргумент функции *int2bit* отвечает за MSb или же LSb репрезентацию битового сообщения, установите его значение равным 0. Таким образом вы получите сопоставление между битовой посылкой знаку, который содержится в ней.

```
% Кодирование алфавита
	Symb = zeros(…,…);
	for i = 0:length(…)-1
		Symb(…,…) = int2bit(…,…,0);
	end
```

После того, как знаковый алфавит был переведен в кодовые слова необходимо декодировать текстовое сообщение из битового массива. Для этого необходимо разделить битовую последовательность на *M* кодовых слов по *N* бит. Воспользуйтесь функцией *reshape*, в аргументы которой необходимо передать входную битовую последовательность *input*, размер кодового слова *N* и указать, что результатом должна вернуться матрица. Для этого третьим аргументом необходимо передать «[]».

```
% Символьное декодирование
	… = reshape(…,…,[]);
```

Далее в цикле `for`, который будет перебирать кодовые слова из принятой последовательности, необходимо сравнить входные данные с массивом кодовых слов алфавита и в случае совпадения кодового слова входного сообщения с кодовым словом алфавита, в новый, результирующий массив, поместить соответствующий знак.

Результатом работы функции должно являться текстовое сообщение.

```
	for i=1:…
		for j=1:…
			if (…(:,i)==…(:,j))
				…(i)=…(j);
			end
		end
	end
end
```

О том, что вами было получено текстовое сообщение необходимо сообщить преподавателю и сравнить принятое сообщение, с переданным.

2. Контрольные вопросы к работе

1) Как рассчитывается длина кодового слова?
2) Для чего необходимо представлять знаки алфавита в виде кодовых слов?

3) Как происходит декодирование знакового сообщения?

3. Требования к оформлению отчета

1) Введение;

2) Приведите структурную схему приемника, который вы реализовали, начиная с 4 работы по текущую. Опишите каждый из реализованных вами блоков приемника. Ход работы;

3) Рисунки;

   а. Текст принятого сообщения;

4) Выводы;

5) Листинг программы. Приведите полный код с 4 работы по текущую. Выводимые в результате выполнения кода графики, должны полностью совпадать с графиками в отчете, за исключением формы принимаемых сигналов.

ЗАКЛЮЧЕНИЕ

Успешно пройденный курс позволяет приобрести навыки программирования и взаимодействия с цифровыми программируемыми приемниками RTL-SDR. Знания и навыки работы со способами взаимодействия с программно-определяемыми радиосистемами посредством использования соответствующего программного обеспечения, такого как: SDR#, Gnu Radio и Matlab. При успешном усвоении курса студентами также будут получены знания и практические навыки работы с такими типами сигналов как FM и OFDM сигналы. Курс дает исчерпывающие знания по взаимодействию с программно-определяемыми радиосистемами, а также позволяет применить на практике полученные ранее навыки цифровой обработки сигналов, что при успешном усвоении материала позволит самостоятельно работать с программно-определяемыми радиосистемами

СПИСОК ЛИТЕРАТУРЫ

1. Документация по MATLAB на русском языке [Электронный ресурс]: перевод статей по MATLAB. URL: https://docs.exponenta.ru/ (дата обращения 07.09.2025).

2. Дьяконов В.П. Matlab. Полный самоучитель./В.П. Дьяконов.- М.: ДМК Пресс, 2012.- 768 с.

3. Щербаков В.С., Руппель А.А., Глушец В.А. Основы моделирования систем автоматического регулирования и электротехнических систем в среде Matlab и Simulink./ В.С. Щербаков, А.А. Руппель, В.А. Глушец.- М.: СибАДИ, 2003. -160 с