МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра РС

ОТЧЕТ

по преддипломной практике

Тема: Программный модуль обработки сигнала стандарта DMR

Студент гр. 8182 _____ Боржонов А.И.

Руководитель _____ Андреева О.М.

Санкт-Петербург
2024

ЗАДАНИЕ

НА УЧЕБНУЮ ПРАКТИКУ

Студент Боржонов А.И.

Группа 8182

Тема практики: Программный модуль обработки сигнала стандарта DMR

Задание на практику:

Реализовать алгоритм обработки сигнала стандарта DMR *с целью обнаружения и идентификации источников радиоизлучения*.

Требования: Язык программирования C++, реализация с использованием объектно-ориентированного стиля программирования, быстродействие алгоритма не хуже чем 1:1.

Сроки прохождения практики: 05.02.2024 – 15.05.2024

Дата сдачи отчета: 00.00.2000

Дата защиты отчета: 00.00.2000

Студент _____ Боржонов А.И.

Руководитель _____ Андреева О.М.

АННОТАЦИЯ

В отчете представлено подробное описание алгоритма обработки сигнала стандарта DMR и его реализации на C++ с использованием ООП-парадигмы.

К результатам преддипломной практики относится программное обеспечение высокой производительности.

SUMMARY

The report provides a detailed description of the optimal signal processing algorithm of the DMR standard and its implementation in C++ using the OOP paradigm.

The results of the work include high-performance software.

СОДЕРЖАНИЕ

	Введение	5
1.	Алгоритм обработки сигнала	6
1.1.	Устройство пеленгования	6
1.2.	Основные этапы алгоритма	7
1.3.	Обнаружение	9
1.4.	Демодуляция	11
1.5.	Декодирование	11
1.6.	Верификация и тестирование алгоритма	14
1.7.	Быстродействие алгоритма	16
	Заключение	17
	Список использованных источников	18

ВВЕДЕНИЕ

Целью данной преддипломной практики является реализация алгоритма обработки сигнала стандарта DMR. Алгоритм выполняет обработку принятого сигнала с целью обнаружения и идентификации источников радиоизлучения (ИРИ).

Для достижения поставленной цели сформированы следующие задачи:

1. Изучение стандарта DMR Air Interface;
2. Изучение основных этапов формирования сигнала физического уровня;
3. Разработка алгоритма;
4. Реализация алгоритма на языке программирования C++;
5. Верификация алгоритма;
6. Тестирование программного модуля на устройстве;

1. АЛГОРИТМ ОБРАБОТКИ СИГНАЛА

1.1. Основные этапы алгоритма

Основываясь на принципах построения сигнала стандарта DMR, предложен и реализован следующий алгоритм обработки сигнала:

- Первым этапом является накопление отсчетов сигнала до минимального количества необходимого для обработки. Накопление осуществляется с использованием класса *FrameCyclicBuffer*. Данный класс является частью коммерческой библиотеки ООО «СТЦ». Входными и выходными данными для коллектора являются комплексные отсчеты сигнала;
- Вторым этапом является коррекция частотной ошибки, вызванной эффектом Доплера и неточностью установки центральной частоты ПУ. Коррекция осуществляется с использованием класса *DmrFreqErrorCompensator*. Входными и выходными данными являются комплексные отсчеты сигнала;
- Третьим этапом является обнаружение синхрогруппы сигнала DMR в потоке данных. Обнаружение осуществляется с использованием класса *DmrPreamble*. Входными данными комплексные отсчеты сигнала, выходными – массив позиций синхрогруппы в наборе входных данных;
- Четвертым этапом является демодуляция сигнала. Демодуляция осуществляется с использованием класса *DmrDemodulator*. Входными данными являются комплексные отсчеты сигнала, выходными – набор демодулированных символов;
- Пятым этапом является декодирование системной информации об устройстве. Декодирование осуществляется с использованием класса *DmrDeviceInfoExtractor*. Входными данными является набор демодулированных символов и позиции начала кадра, выходными – массив, состоящий из пар: позиция начала кадра и системная информация об устройстве;

- Последним этапом алгоритма является формирование результата работы алгоритма. При формировании результата учитывается был или не был обнаружен сигнал и был обнаруженный сигнал декодирован или нет. В случае обнаружения сигнала формируется пара значений: позиция начала кадра в полученном сигнале и тип обнаруженной синхрогруппы. В случае декодирования к уже имеющейся структуре добавляется системная информация.

Описанный алгоритм был реализован как метод *process()* класса *DmrCore*, содержание данного класса является коммерческой тайной ООО «СТЦ». Результатом работы метода являются два возможных состояния:

- *detected* (обнаружено) – получение данного состояния возможно только в случае, когда сигнал был обнаружен.
- *notDetected* (не обнаружено) – получение данного состояния возможно только в случае, когда сигнал не был обнаружен.

На рисунке 1.1 изображена блок-схема алгоритма.

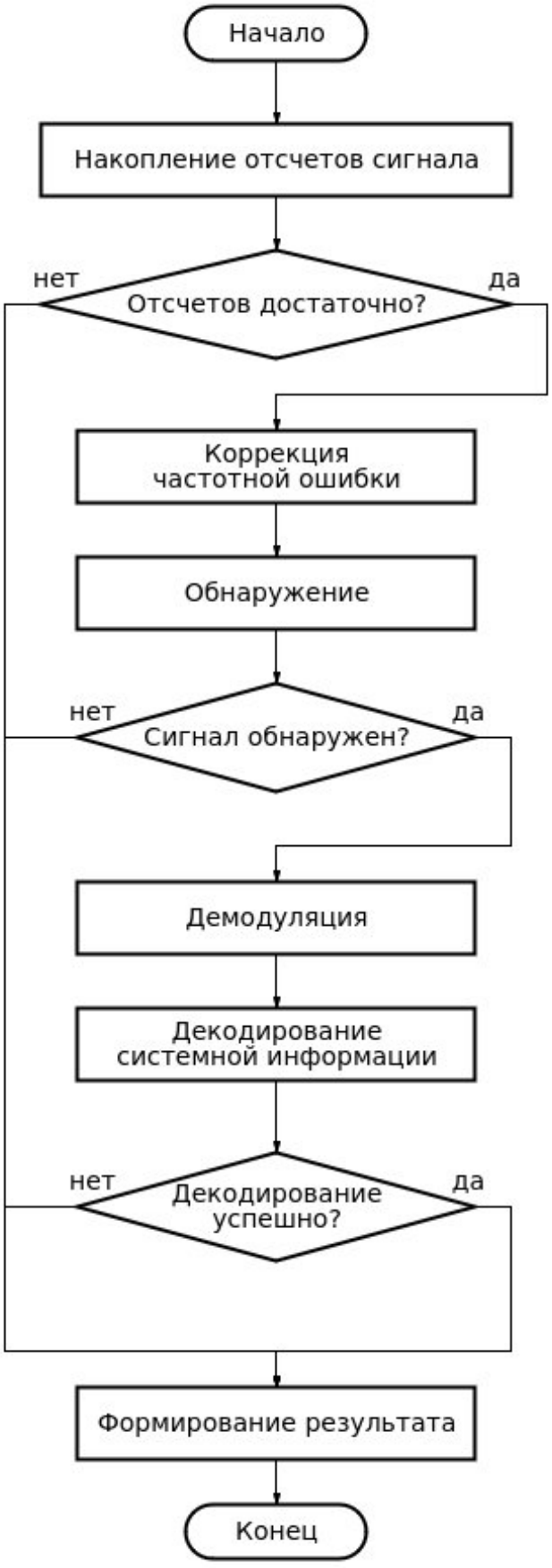

Рисунок 1.1 – Блок-схема алгоритма обработки сигнала.

1.2. Обнаружение

Алгоритм обнаружения реализован следующим образом:

- Первым этапом алгоритма обнаружения является расчет корреляционных функций принятого сигнала с 4-мя сигналами синхрогрупп (BS Data, BS Voice, MS Data, MS Voice).
- Вторым этапом является нормировка полученных корреляционных функций.
- На третьем этапе происходит поиск максимумов всех нормированных корреляционных функций. Далее выбирается наибольший из четырех максимумов, на основе этого выбора, определяется какой тип синхрогруппы присутствует в сигнале.
- Четвертым этапом является сравнение максимума корреляции с порогом обнаружения. В случае превышения порога выносится решение о наличии сигнала, если же порог не был превышен, выносится решение об отсутствии сигнала.
- В случае обнаружения сигнала, пятым этапом является вычисление позиции начала кадра.
- На последнем этапе происходит формирование результата. В случае обнаружения сигнала алгоритм возвращает значение *true* и в контейнер с результатами заносится структура, содержащая в себе позицию начала кадра и тип синхрогруппы. В случае если сигнал не был обнаружен алгоритм возвращает значение *false*.

Описанный алгоритм реализован как метод *findDmr()* класса *DmrPreamble*. Класс *DmrPreamble* содержит в себе четыре вектора комплексных отсчетов, представляющих собой сигналы синхрогрупп и контейнер с результатами обнаружения. Для извлечения результатов обнаружения в классе *DmrPreamble* предусмотрен метод *getResults()*. Реализация класса *DmrDemodulator* является коммерческой тайной ООО «СТЦ».

Блок-схема алгоритма декодирования представлена на рисунке 1.2.

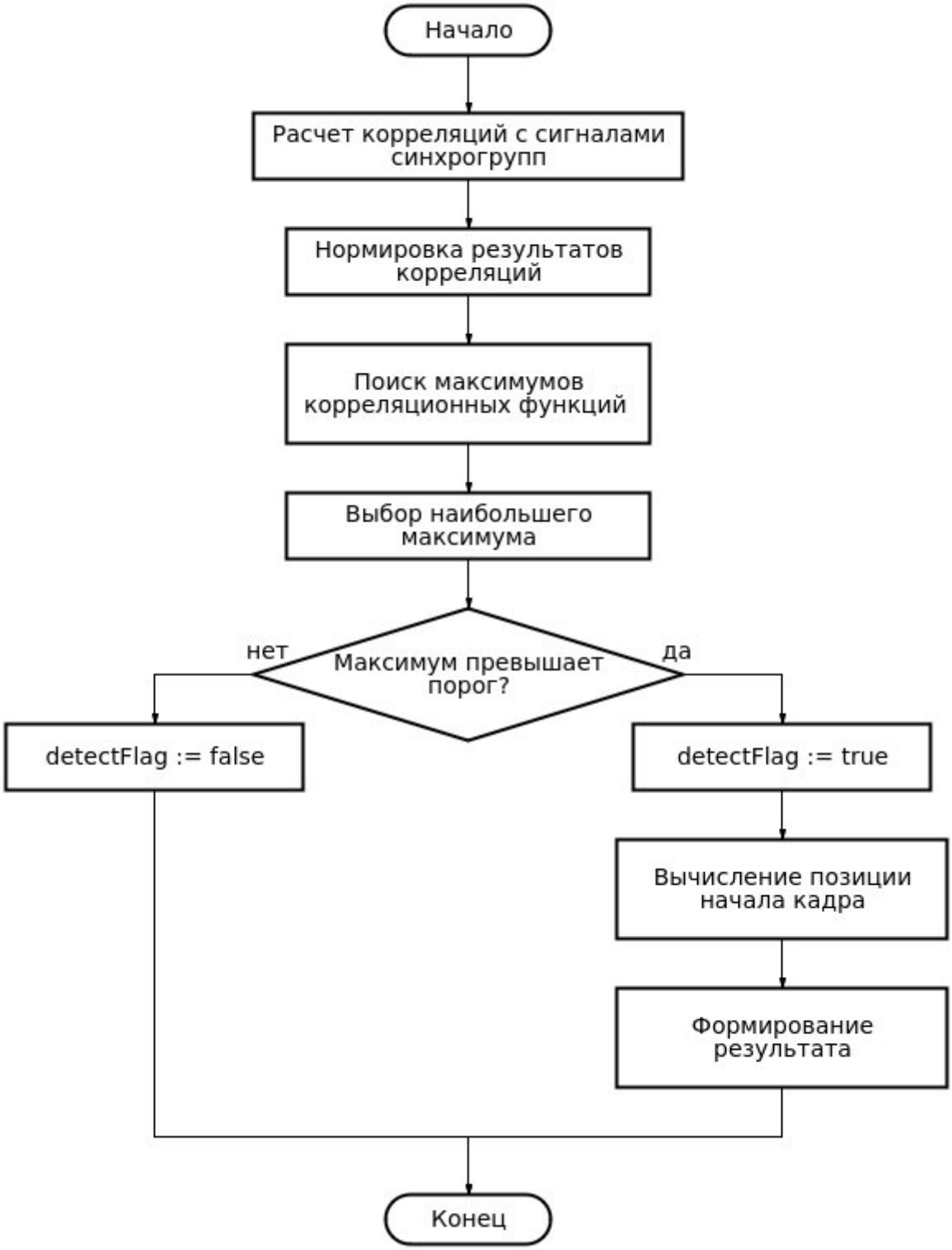

Рисунок 1.2 – Блок-схема алгоритма обнаружения.

1.3. Демодуляция

Алгоритм демодуляции предполагает использование четырех полосовых фильтров с центральными частотами, которые соответствуют модуляции 4FSK, используемой в стандарте DMR. Для каждого отсчета выбирается тот символ, фильтр которого дал наибольший выход. АЧХ фильтров представлена на рисунке 1.3.

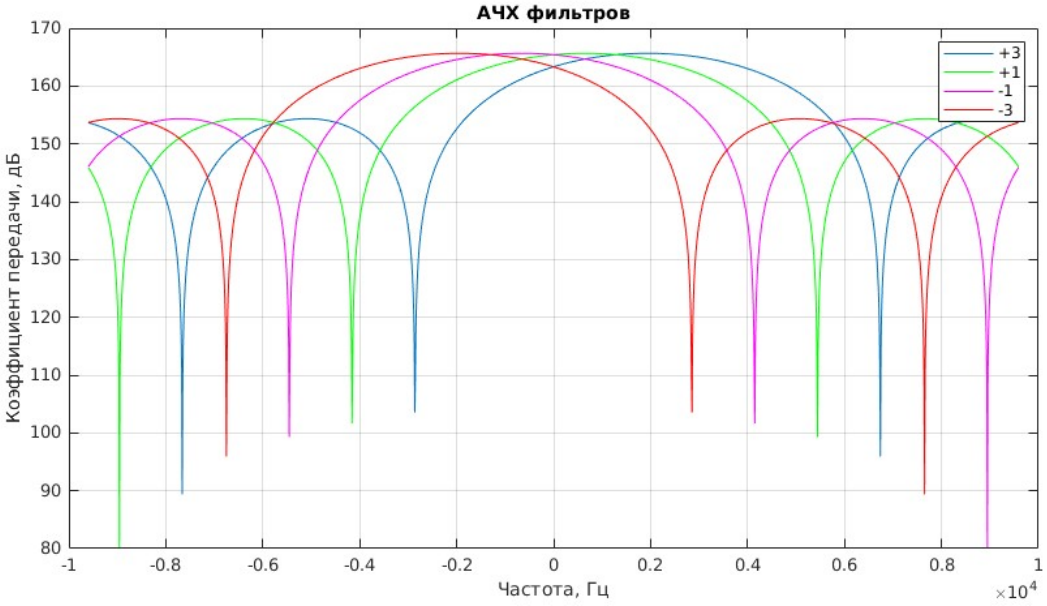

Рисунок 1.3 – АЧХ фильтров демодулятора.

Описанный алгоритм реализован как метод *demodulate()* класса *DmrDemodulator*. Класс *DmrDemodulator* содержит в себе четыре вектора комплексных отсчетов, представляющих собой импульсные характеристики фильтров и вектор символов. Реализация класса *DmrDemodulator* является коммерческой тайной ООО «СТЦ».

1.4. Декодирование

Алгоритм декодирования реализован следующим образом:

- Первым этапом алгоритма декодирования является преобразование демодулированных символов в биты.
- Вторым этапом происходит декодирование поля «Тип слота». Для декодирования поля «Тип слота» используется декодер кода Рида-Соломона, реализация данного декодера является

коммерческой тайной ООО «СТЦ». Из поля тип слота извлекаются информационные элементы «Цветовой код» и «Тип данных».

- Третьим этапом осуществляется деперемежение информационных бит. Класс деперемежитель *DmrDeinterleaver* написан на основании алгоритма перемежения описанного в стандарте DMR.
- Четвертым этапом является декодирование кода BPTC (196, 96). Декодирование осуществляется с использованием декодера кода Хэмминга, реализация данного декодера является коммерческой тайной ООО «СТЦ».
- На пятом этапе происходит проверка CRC. Перед проверкой CRC к последним 24 битам информационного сообщения (битам четности) применяется специальная CRC маска, она выбирается на основе информационного элемента «Тип данных». В случае, если элемент «Тип данных» принимает значение Idle (Пустой), происходит формирование результата декодирования.
- Шестым этапом является получение системной информации. В зависимости от элемента «Тип данных» применяются разные варианты преобразования декодированных бит в системную информацию. Для преобразования системной информации используется класс *DmrParcer*.

Описанный алгоритм реализован как метод *decodeData()* класса *DmrDeviceInfoExtractor*. Класс *DmrDeviceInfoExtractor* содержит в себе четыре объекта декодеров (декодер кода Хэмминга (13, 9), декодер кода Хэмминга (15, 11), декодер кода Голея (20,8) и декодер кода Рида-Соломона (12,9)), один объект класса *DmrDeinterleaver* и один объект класса *DmrParcer*. Реализация класса *DmrDeviceInfoExtractor* является коммерческой тайной ООО «СТЦ».

Блок-схема алгоритма декодирования представлена на рисунке 1.4.

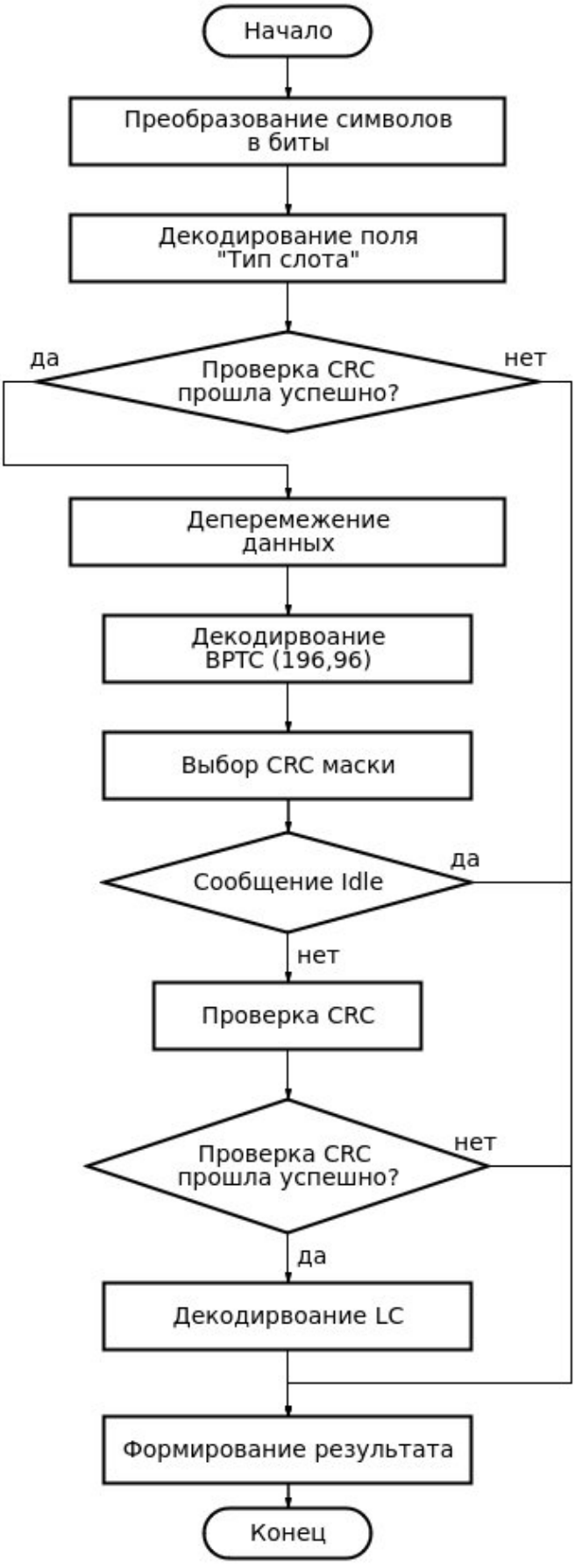

Рисунок 1.4 – Блок-схема алгоритма декодирования.

1.5. Верификация и тестирование алгоритма

Верификация алгоритмов обнаружения сигнала и декодирования, описанных в п. 1.3 и 1.5 соответственно, осуществлялась с использованием ПО MATLAB. Алгоритм декодирования и все функции, описанные в нем, были верифицированы ООО «СТЦ».

Для верификации алгоритма обнаружения в MATLAB был сгенерирован тестовый сигнал соответствующий одному сообщению LC.

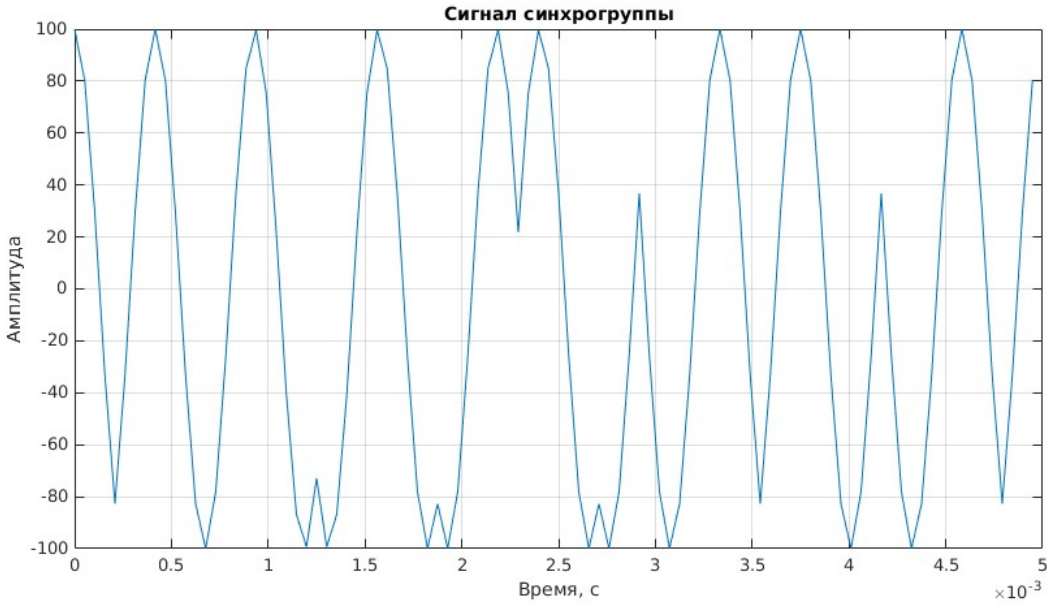

Рисунок 1.5 – Сигнал синхрогруппы (BS Data).

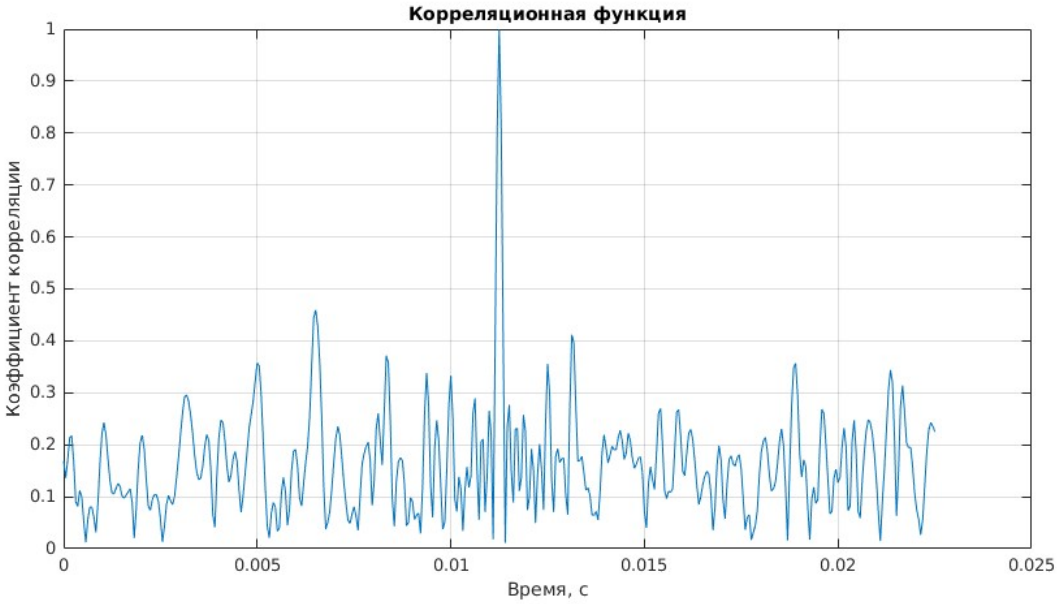

Рисунок 1.6 – Нормированная автокорреляционная функция сигнала синхрогруппы (BS Data).

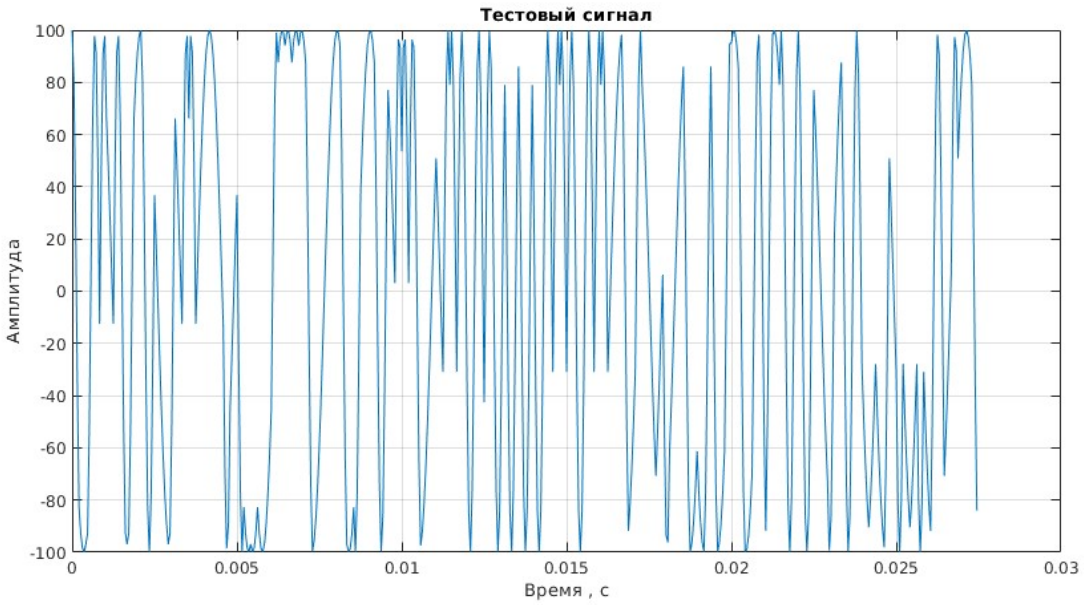

Рисунок 1.7 – Тестовый сигнал.

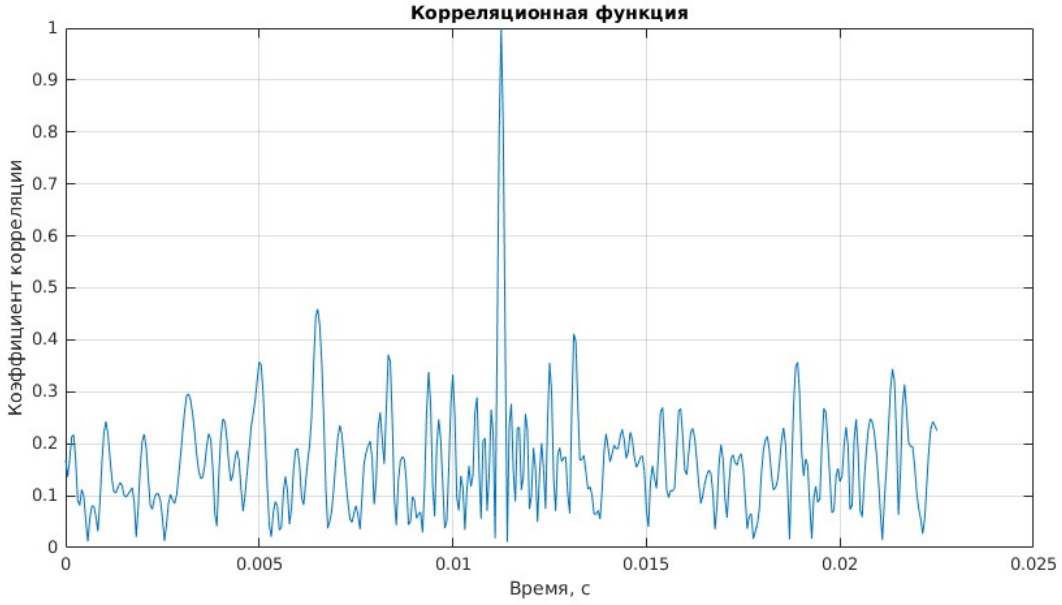

Рисунок 1.8 – Нормированная корреляционная функция тестового сигнала с сигналом синхрогруппы (BS Data).

Выше приведены рисунки сигнала синхрогруппы (BS Data) (рис. 1.5), нормированной автокорреляционной функции сигнала синхрогруппы (рис. 1.6), тестового сигнала сгенерированного в MATLAB (рис. 1.7) и нормированной корреляционной функции тестового сигнала и сигнала синхрогруппы (рис. 1.8), полученные в MATLAB.

Верификация демодуляции и алгоритма декодирования была произведена путем сравнения демодулированных и декодированных бит с

соответствующими битами тестового сигнала. При верификации кода на C++ был использован аналогичный тестовый сигнал, в результате были получены аналогичные выходные данные.

Верификация алгоритмов демодуляции и декодирования проводилась путем сравнения символов и битов, полученных после демодуляции и декодирования с соответствующими символами и битами тестового сигнала. Аналогичные тестовые сигналы использовались при верификации кода на C++, в результате были получены аналогичные выходные данные.

1.7. Быстродействие алгоритма

Для оценки быстродействия быстродействия алгоритма на вход программного модуля был подан тестовый сигнал размером 5120 отсчетов с частотой дискретизации 19200 Гц, что составляет 267 мс. Требования к быстродействию алгоритма выражены следующим условием: отношение времени обработки подаваемого сигнала ко времени самого сигнала не должно превышать 1:1.

Длительность обработки и отдельных её этапов представлены в таблице 1.1.

Таблица 1.1 – Быстродействие алгоритма.

Этап	Время, мкс
Коррекция частотной ошибки	19
Обнаружение	27
Демодуляция	21
Декодирование	5
Полный цикл	1725

Полученные результаты удовлетворяют заданным требованиям к быстродействию алгоритма, т.к. отношение времени обработки подаваемого сигнала ко времени самого сигнала равно 1:153.

ЗАКЛЮЧЕНИЕ

В результате прохождения практики был реализован алгоритм обработки сигнала стандарта DMR. Алгоритм был внедрен в программный модуль реализованный на языке программирования C++ с использованием объектно-ориентированного стиля программирования, представляющий собой высокопроизводительное, кроссплатформенное программное обеспечение, реализующее обнаружение и демодуляцию сигнала, а так же идентификацию устройств стандарта DMR.

В ходе прохождения практики были решены следующие задачи:

- Изучение стандарта DMR Air Interface;
- Изучение основных этапов формирования сигнала физического уровня;
- Разработка алгоритма;
- Реализация алгоритма на языке программирования C++;
- Верификация алгоритма;
- Тестирование программного модуля на устройстве;

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Федеральный закон «О связи»: текст с изменениями на Ф32 2021 год. – Москва: Эксмо, 2021. – 112 с. – (Актуальное законодательство).
2. Сергиенко А.Б. Цифровая связь: учеб. пособие. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2012. 164 с.
3. Прокис Дж. Цифровая связь / под ред. Д.Д. Кловского М.: Радио и связь, 2000. 800 с.
4. Скляр Б. Цифровая связь. Теоретические основы и практическое применение. Изд. 2-е / под ред. А.В. Назаренко М.: Издательский дом "Вильямс", 2003. 1104 с.
5. Радиомониторинг – задачи, методы, средства / Под ред. А.М. Рембовского. 2-е изд., перераб. И доп. – М.: Горячая линия-Телеком, 2021. – 624 с.: ил.
6. Справочник по радиоконтролю. МСЭ 2002. Женева. 2004. 584 с.
7. Проблемы поиска сигналов системами радиоэлектронной борьбы // Иностранная печать. Сер. ТСР. 1998. №9. С. 25-32.
8. Рембовский А.М. Задачи и структура средств автоматизированного радиоконтроля // Специальная техника. 2003. С. 2-7.
9. Алиев Д.С., Иванов А.В., Иванов А.В.,. Анализ конструкций современных пеленгаторных антенн. «Воздушно-космические силы. Теория и практика» № 1, март 2017.
10. Electromagnetic compatibility and Radio spectrum Matters; Digital Mobile Radio Systems; Part 1: DMR Air Interface protocol.
11. Ипатов В. Широкополосные системы и кодовое разделение сигналов. Принципы и приложения. Москва: Техносфера, 2007. – 488с.
12. Волков В.Ю. Моделирование и обработка сигналов и полей в радиотехнических задачах: учеб. пособие. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2021. 122 с.