**Содержание**

[ВВЕДЕНИЕ 3](#_Toc125890490)

[1 Теоретическая часть 5](#_Toc125890491)

[1.1 Анализ системы КТР 5](#_Toc125890492)

[1.2 Модуляция LoRa и обоснование её применения для решения задачи 8](#_Toc125890493)

[1.2.1 Синхронизация приемника и передатчика 11](#_Toc125890494)

[1.3 Сверточное кодирование 13](#_Toc125890495)

[1.3.1 Метод декодирования по алгоритму Витерби 15](#_Toc125890496)

[1.3.2 Метод порогового декодирования 16](#_Toc125890497)

[1.3.3 Метод последовательного декодирования 16](#_Toc125890498)

[2 Основная часть 17](#_Toc125890499)

[2.1 Разработка и анализ структурной и функциональной схем 17](#_Toc125890500)

[2.1.1 Разработка и анализ структурной схемы 17](#_Toc125890501)

[2.1.2 Разработка и анализ функциональной схемы 19](#_Toc125890502)

[2.2 Радиомодуль SX1276 20](#_Toc125890503)

[2.3 Микроконтроллер STM32F7 24](#_Toc125890504)

[2.4 Импульсный преобразователь напряжения LM5005 25](#_Toc125890505)

[2.5 Приемопередатчик KSZ8041NL 26](#_Toc125890506)

[2.6 Преобразователь интерфейса CP2102-GMR 27](#_Toc125890507)

[2.7 CAN контроллер MAX3051 28](#_Toc125890508)

[2.8 Сравнение 29](#_Toc125890509)

[2.8.1 Командно-телеметрический радиомодем РМ-02М2 29](#_Toc125890510)

[2.8.2 Командно-телеметрический радиомодем РМ-16 30](#_Toc125890511)

[2.8.3 Цифровой модем 3D Link 30](#_Toc125890512)

[3 Разработка программно алгоритмического обеспечения 32](#_Toc125890513)

[3.1 Выбор и обоснование используемых инструментов для разработки программно-алгоритмического обеспечения 32](#_Toc125890514)

[3.2 Выбор и обоснование параметров радиомодема 33](#_Toc125890515)

[3.3 Разработка структурной схемы работы системы 34](#_Toc125890516)

[3.3.1 Работа системы в дуплексном режиме 35](#_Toc125890517)

[3.3.2 Работа системы в полудуплексном режиме 36](#_Toc125890518)

[3.4 Алгоритм определения ведущего устройства по радиоканалу 37](#_Toc125890519)

[3.5 Расчёт корректировки синхронизирующего таймера 39](#_Toc125890520)

[4 Экспериментальные исследования 41](#_Toc125890521)

[4.2 Расчет бюджета канала связи наземная станция управления – беспилотный летательный аппарат 430 МГц 45](#_Toc125890522)

[4.3 Расчет бюджета канала связи наземная станция управления – беспилотный летательный аппарат 863 МГц 45](#_Toc125890523)

[4.5 Экспериментальная проверка дальности связи 49](#_Toc125890524)

[5 Безопасность и экологичность 52](#_Toc125890525)

[5.1 Оценка опасных и вредных факторов 52](#_Toc125890526)

[5.2 Влияние ПЭВМ на организм человека 53](#_Toc125890527)

[5.3 Нормативные требования при организации работы на ПЭВМ 54](#_Toc125890528)

[5.4 Расчёт искусственного освещения 55](#_Toc125890529)

[5.5 Мероприятия по защите при организации работы на ПЭВМ 60](#_Toc125890530)

[ЗАКЛЮЧЕНИЕ 62](#_Toc125890531)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 63](#_Toc125890532)

[ПРИЛОЖЕНИЕ А 65](#_Toc125890533)

[ПРИЛОЖЕНИЕ Б 66](#_Toc125890534)

[ПРИЛОЖЕНИЕ В 67](#_Toc125890535)

[ПРИЛОЖЕНИЕ Г 73](#_Toc125890536)

# ВВЕДЕНИЕ

Задачей командно-телеметрической радиолиния (КТР) является установка надёжной связи между беспилотным летательным аппаратом (БПЛА) и комплексом управления. К системе КТР предъявляются высокие требования по дальности (до 100 км в топологии точка-точка) и помехоустойчивости (вероятность битовой ошибки не менее 10-6). Для выполнения требований по дальности таких систем связи следует использовать спектрально-эффективные и многопозиционные методы модуляции, а также помехоустойчивое кодирование. Учитывая внушительную область применения БПЛА (целеуказания, ретрансляция сообщений и данных, доставка грузов, тушение пожаров, перевозка пассажиров), а также космических сферах становится очевидным актуальность системы КТР. Система КТР состоит из двух (наземный комплекс управления, БПЛА) либо из трех устройств (наземный комплекс управления, БПЛА, БПЛА). В случае наличия третьего устройства система связи должна обеспечивать возможность ретрансляции. Целью данного проекта является написание программного обеспечения (ПО) для дуплексного модема с применением модуляции LoRa.

Новизна данной работы заключается в том, что разработанная система связи будет иметь дальность связи до 100 км в топологии «точка-точка», а также обеспечивать дуплексный режим работы. В данной системе КТР используются актуальные методы модуляции, такие как LoRa. Данная система является независимой, что позволяет ее использовать как отдельно от комплекса БПЛА, так и в составе различных БПЛА. Система связи сможет работать на расстояниях до 100 км, без применения ретрансляции и с применением ретрансляции на расстояниях 200 км.

Сигналы LoRa являются сигналами с расширенным спектром, что позволяет повысить помехоустойчивость системы. Преимуществом данной технологии является наличие ее реализации в виде микросхем, таких как SX1276, а также наличие готового программного обеспечения для управления данной микросхемой, что позволяет повысить скорость разработки ПО. Технология LoRa в отличии от традиционных систем с прямым расширением спектра, в которых

фаза несущей сигнала передатчика изменяется в соответствии с кодовой последовательностью, не требует наличия высокоточных эталонных часов что снижает стоимость используемого изделия. В модуляции LoRa расширение спектра достигается за счет генерации ЛЧМ-сигнала. Также модуляция LoRa включает схему переменной коррекции ошибок, которая повышает надежность передаваемого сигнала.

Задачами данного проекта являются:

- исследование технологий, применяемых в используемом радиомодеме;

- запуск радиомодема;

- разработка программного обеспечения для работы системы КТР в дуплексном и полудуплексном режимах;

- отладка и тестирование написанного ПО;

- тестирование дальности системы;

В ходе работы планируется выполнить следующее:

- разработать программное обеспечение для работы системы связи в дуплексном и полудуплексном режимах;

- реализовать алгоритм взаимодействия между радиомодемами;

- реализовать алгоритмы приема, передачи и обработки данных по радиоканалу и через последовательный порт;

- разработать вспомогательное программное обеспечение для тестирования системы связи;

- провести анализ радиоканала и тестирование дальности системы связи;

# Теоретическая часть

## 1.1 Анализ системы КТР

Система КТР предназначена для пересылки командно-телеметрической информации между устройством и комплексом управления. В качестве первых могут выступать БПЛА. Из-за стремительного развития БПЛА необходимо постоянно пересматривать требования выдвигаемые к командно-телеметрическому каналу связи между БПЛА и наземным комплексом управления.

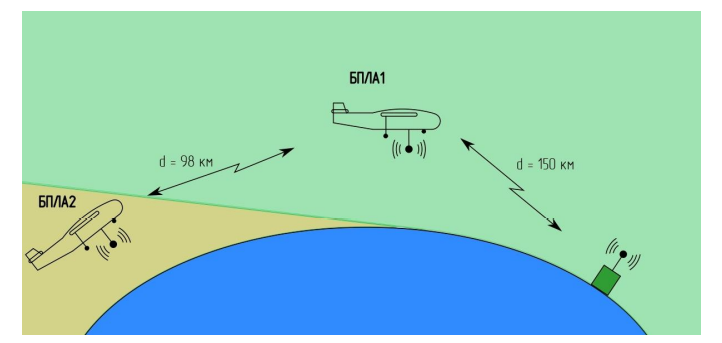


Рисунок 1 – Работа системы КТР

Учитывая внушительную область применения БПЛА (целеуказания, ретрансляция сообщений и данных, доставка грузов, тушение пожаров, перевозка пассажиров), а также космических сферах становится очевидным актуальность системы КТР.

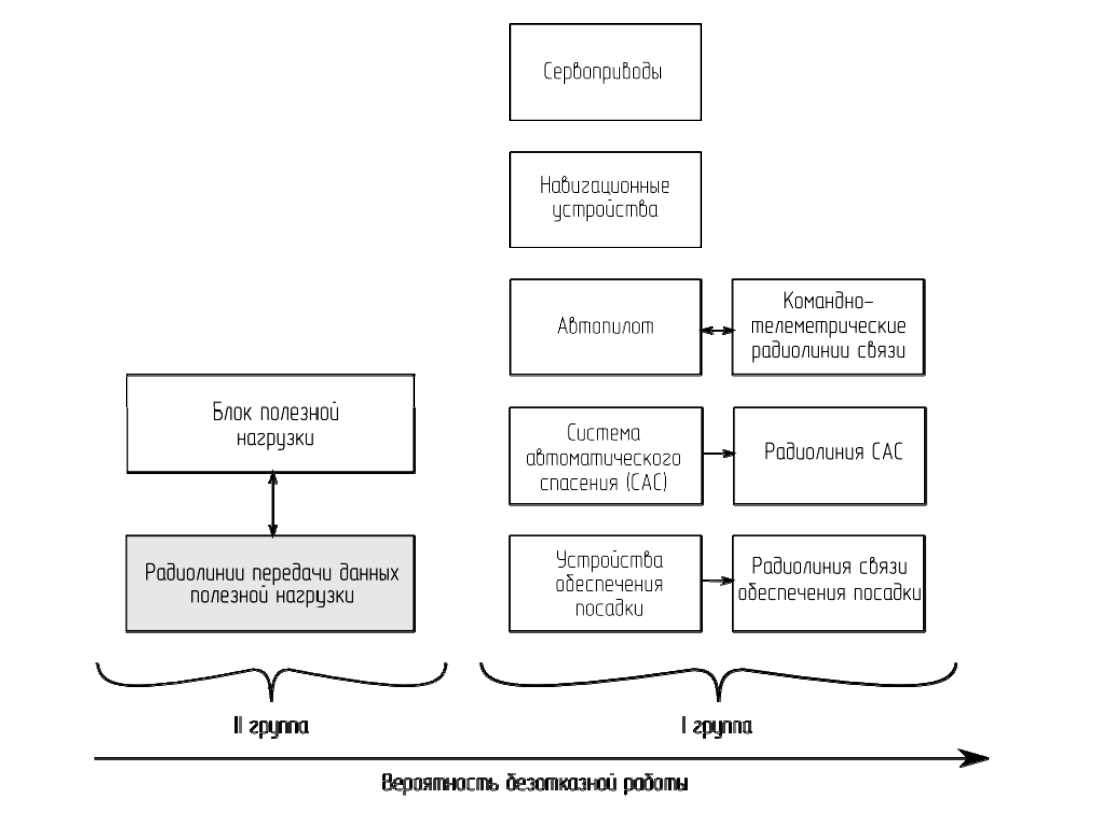


Рисунок 2 – Распределение структурных узлов БПЛА на группы по вероятности безотказной работы

Как видно из рисунка система командно-телеметрической радиолинии связи входит в первую группу устройств по вероятности безотказной работы, что означает в свою очередь означает необходимость повышения требований к её отказоустойчивости. Используя широкополосные сигналы и спектрально-эффективные методы модуляции можно выполнить требования, которые предъявляются к пропускной способности канала связи и его дальности К таким относятся двухпозиционные методы, многопозиционные методы, модуляции с расширенным спектром. Для увеличения помехоустойчивости передаваемых сообщений используются различные методы кодирования: линейное, сверточное и т.д. [1]. В разрабатываемой системе используется модуляция с расширенным спектром и сверточное кодирование.

Преимущества систем связи с расширенным спектром:

- более лучшее качество связи в условиях преднамеренных помех;

- за счет увеличения полосы сигнала снижается его плотность энергии;

- высокая временная разрешающая способность;

- множественный доступ и сохранение конфиденциальности между пользователями;

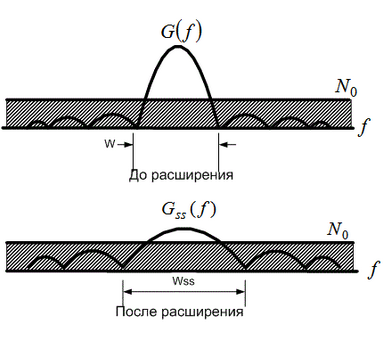


Рисунок 3 –Пример сигнала с расширенным спектром

Расширение спектра сигнала происходит путем увеличения полосы сигнала сигнала одним из следующих методов:

1. метод скачкообразного перестроения частоты, суть которого заключается в периодическом скачкообразном изменении несущей частоты по закону, известному приёмнику и передатчику. Преимуществом данного метода является простота реализации, недостатком — задержка в потоке данных при каждом скачке;

2. расширение спектра сигнала с помощью прямой последовательности. Этот метод по эффективности превосходит метод перестроения частоты, но сложнее в реализации. Каждому информационному символу передаваемого сообщения сопоставляется псевдослучайная последовательность определенной длины. Длительность символов этой последовательности много меньше длительности символа исходного сообщения. За счет данного механизма и происходит расширение спектра;

3. расширение спектра с помощью сигнала с линейной частотной модуляцией (ЛЧМ). Данный метод позволяет расширить полосу передаваемого сигнала за счет линейного изменения частоты от значения *f*min до *f*max. Разность значений этих частот и будет определять полосу итогового сигнала [2]

## 1.2 Модуляция LoRa и обоснование её применения для решения задачи

Модуляция LoRa позволяет передавать сигнал на расстояния более чем 15 километров, при мощности сигнала 20 дБм. За счет использования дополнительных усилителей мощности это расстояние может быть увеличено до необходимого значения. Ключевой особенностью технологии LoRa является ее сверхнизкое энергопотребление. Одним из преимуществ модуляции LoRa является готовая схемотехническая реализация приемопередающих чипов, что избавляет от разработки приемопередающего тракта. Приемопередатчики являются программируемыми, что позволяет управлять сеансом связи, параметрами приемника и передатчика. Программное обеспечение для взаимодействия с модемом находится в открытом доступе. Данный ресурс позволит сэкономить время, затрачиваемое на разработку.

Модуляция LoRa основана на использовании радиосигналов с базой B много большей единицы. Данный вид радиосигналов имеет две главные особенности:

- ширина спектра радиосигнала значительно больше скорости передачи данных;

- корреляционная функция существенно уже корреляционной функции узкополосного радиосигнала с базой примерно равной единице.

Частотная избыточность широкополосного радиосигнала LoRa обуславливает его достаточно высокую помехоустойчивость, а узкая корреляционная функция – высокую точность временной синхронизации.

Модуля LoRa представляет собой сигнал, основанный на сигнале с линейной частотной модуляцией (ЛЧМ). Частота ЛЧМ сигнала может как увеличиваться, так и уменьшаться. Математически ЛЧМ сигнал представляется в виде выражения:

где –Tsym /2< t < Tsym /2

и описывается следующими параметрами:

*BW*– ширина спектра радиосигнала;

Tsym = 2SF/BW – длительность радиосигнала;

*μ* = *BW*/*Tsym* – скорость изменения частоты радиосигнала;

*B* = *BW∙Tsym* **=** 2*SF* – база радиосигнала.

*ω*0 = 2*π∙f*0 – центральная (несущая) частота радиосигнала;

*f*н = *f*0 – *BW*/2– нижняя частота радиосигнала;

*f*в = *f*0 + *BW*/2– верхняя частота радиосигнала;

*SF* – коэффициент расширения спектра;

Коэффициент расширения спектра *SF* определяет количество информационных бит, передаваемых в одном символе данных сигнала. Коэффициент расширения спектра изменяется от 7 до 12. На рисунках 4-5 представлены ЛЧМ сигнал и его спектр соответственно.



Рисунок 4 –Временная диаграмма сигнала ЛЧМ



Рисунок 5 – Пример спектра ЛЧМ сигнала

Передатчики LoRa формируют сигналы с фиксированными полосами. Изменение базы сигнала происходит за счет скорости изменения частоты *μ* и длительности символа Значения полос формируемых сигналов составляют 125, 250 и 500 кГц.

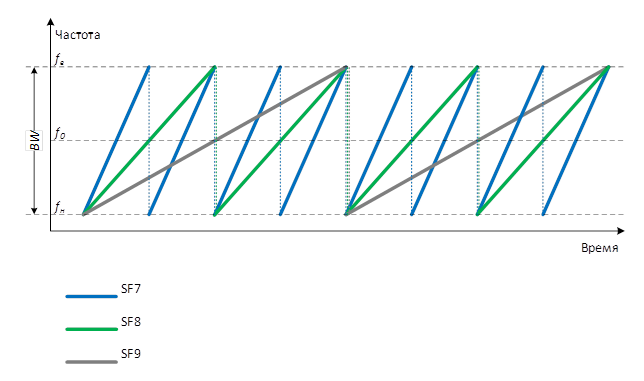


Рисунок 6 – Изменение частоты сигнала

## 1.2.1 Синхронизация приемника и передатчика

Синхронизация приемника и передатчика необходима для установки временных границ, в течении которых будет происходить прием или передача сигнала.

В начале каждого сообщения, отправляемого передатчиком, находится преамбула, позволяющая приемнику обнаружить активность на передающей стороне. Таким образом происходит синхронизация приемника и передатчика. Формат передаваемого сообщения представлен на рисунке 7.

Во время сеанса синхронизации приемник выполняет символьную синхронизацию. Длительность преамбулы является конфигурируемой величиной и должна быть не менее, чем *T1+2T2*, где *T1* определяет максимальное время нахождения приемника в состоянии "сна", *T2* – определяет время поиска приемником преамбулы.

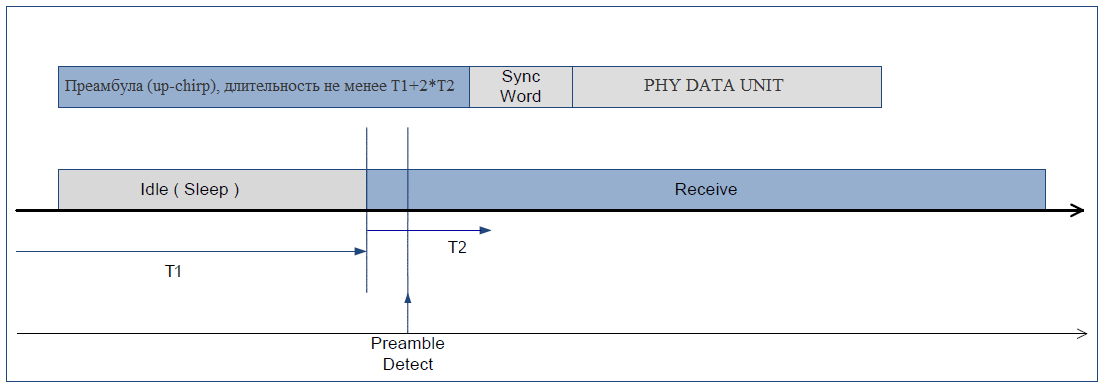
[](https://itechinfo.ru/sites/default/files/lora/pic9.png)

Рисунок 7 – Распространение кадров

По завершении преамбулы следует слово синхронизации (Sync Word) и блок данных физического уровня. Длина слова синхронизации настраивается в диапазоне от 1 до 8 байт. Спецификацией LoRa определен ряд специфических значений Sync Word – 0x34 для публичных сетей (public networks), 0x12 – для частных сетей (private networks) и 0xC194C1 – для каналов с FSK модуляцией [2].

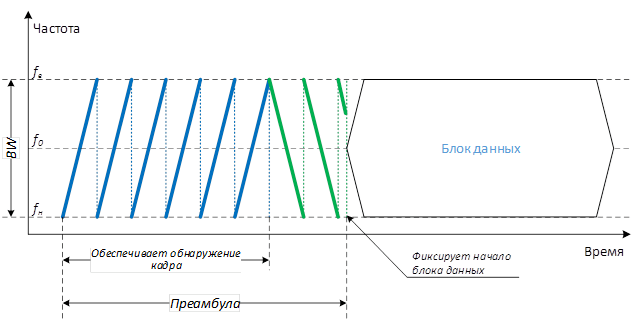
[](https://itechinfo.ru/sites/default/files/lora/pic10.png)

Рисунок 8 – Структура кадра обеспечивающего передачу одного блока данных

## 1.3 Сверточное кодирование

Двоичные сверточные коды основаны на преобразовании входной последовательности в выходную, в которой на каждый символ входной последовательности приходится более одного символа выходной. Сверточное кодирование удобнее всего описывать, характеризуя действие соответствующего кодирующего устройства. В процессе кодирования кодер каждый такт преобразует *k* входных символов в *n* выходных символов, передаваемых по каналу связи. Скорость, с которой он это делает называется скоростью сверточного кодирования, которая равна отношению *k* к *n* и обозначается буквой *R*. Выходные символы, создаваемые кодером на данном такте, зависят от *m* информационных символов, поступивших на этом и предыдущем тактах. Таким образом, выходные символы сверточного кодера однозначно определяются его входным сигналом и состоянием, зависящим от *m*–*k* предыдущих информационных символов.

В состав сверточного кодера входят: коммутатор, регистр сдвига, сумматоры по модулю 2

Peгиcтp cдвигa являeтся динaмичeским зaпoминaющим ycтрoйствoм, в кoтoрoм хрaнятся двoичные симвoлы 0 или 1. Число триггерных ячеек регистра равно m. В момент поступления на вход регистра нового информационного символа символ, хранящийся в крайнем правом разряде, выводится из регистра и сбрасывается. Каждый из остальных, хранящихся в регистре символов перемещается на один разряд вправо, освобождая тем самым крайний левый разряд, куда и поступает новый информационный символ.

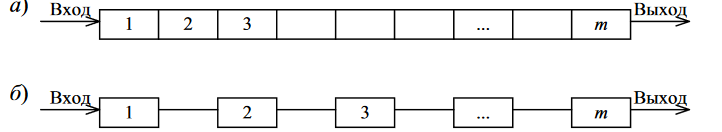


Рисунок 9 – Регистр сдвига

Сумматор по модулю 2 осуществляет сложение поступающих на его входы символов 0 и 1. Правило сложения следующее: сумма двоичных символов равна 0, если число единиц среди поступающих на входы символов четно, и равна 1, если нечетно.

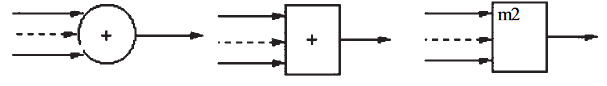


Рисунок 10 – Сумматор по модулю 2

Коммутатор последовательно считывает поступающие на его входы символы и определяет на его выходе очередность кодовых символов в канал связи.

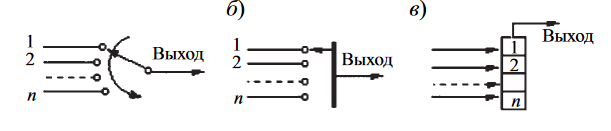


Рисунок 11 – Коммутатор

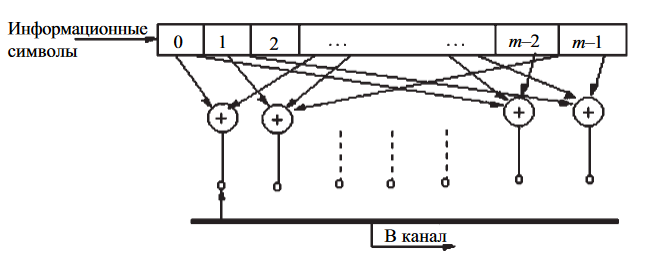


Рисунок 12 – Общий вид двоичного сверточного кодера

В общем случае процесс кодирования происходит следующим образом: каждый такт на вход сдвигового регистра поступает один символ из двоичной, имеющиеся в регистре символы сдвигаются на 1 символы, тем самым освобождая место для нового.

С триггерных ячеек в соответствии порождающему полиному снимаются значения и подаются на сумматоры. Коммутатор делает цикл опроса

сумматоров и передает в канал значения с их выхода. Ниже представлены примеры кодеров.

|  |  |
| --- | --- |
|  |  |

Рисунок 13 – Примеры кодеров для скоростей 1/3 и 2/3

Выделяют следующие методы декодирования последовательностей:

- Метод декодирования по алгоритму Витерби;

- Метод порогового декодирования;

- Метод последовательного декодирования;

## 1.3.1 Метод декодирования по алгоритму Витерби

Метод представляет собой декодирование по максимуму правдоподобия. Идея алгоритма Витерби состоит в том, что в декодере воспроизводят все возможные пути последовательных изменений состояний сигнала, сопоставляя получаемые при этом кодовые символы с принятыми аналогами по каналу связи и на основе анализа ошибок между принятыми и требуемыми символами определяют оптимальную последовательность.

## 1.3.2 Метод порогового декодирования

При пороговом декодировании вычисляются синдромы, затем эти синдромы или последовательности, полученные посредством линейного преобразования, подаются на входы порогового устройства, где путем «голосования» и сравнения его результатов с порогом выносится решение о значении декодируемого символа.

## 1.3.3 Метод последовательного декодирования

При последовательном декодировании число операций, которое должен выполнить декодер изменяется в зависимости от уровня шумов в канале. Число операций является функцией скорости передачи и шумов в канале. При всех скоростях передачи, меньших определенной скорости, число операций при декодировании оказывается небольшим. В состав декодера входит буферное запоминающее устройство. Если при декодировании память буферного устройства окажется заполненной, то возникнут ошибки декодирования. Алгоритм декодирования опускается из-за соображений компактности, за более

подробной информацией следует обратиться к [4]

# Основная часть

## 2.1 Разработка и анализ структурной и функциональной схем

При разработке структурной и функциональной схем был проведен анализ элементной базы. Производился поиск открытого исходного кода на основании, которого подбирались интегральные схемы.

## 2.1.1 Разработка и анализ структурной схемы

В состав модема входят:

- микроконтроллер;

- цепь питания микросхем;

- цепь питания усилителей мощности;

- преобразователи интерфейсов;

- два приемопередатчика;

- два усилителя мощности;

- два полосовых фильтра;

- два малошумящих усилителя;

- два антенных ключа;

Ниже представлена разработанная структурная схема устройства

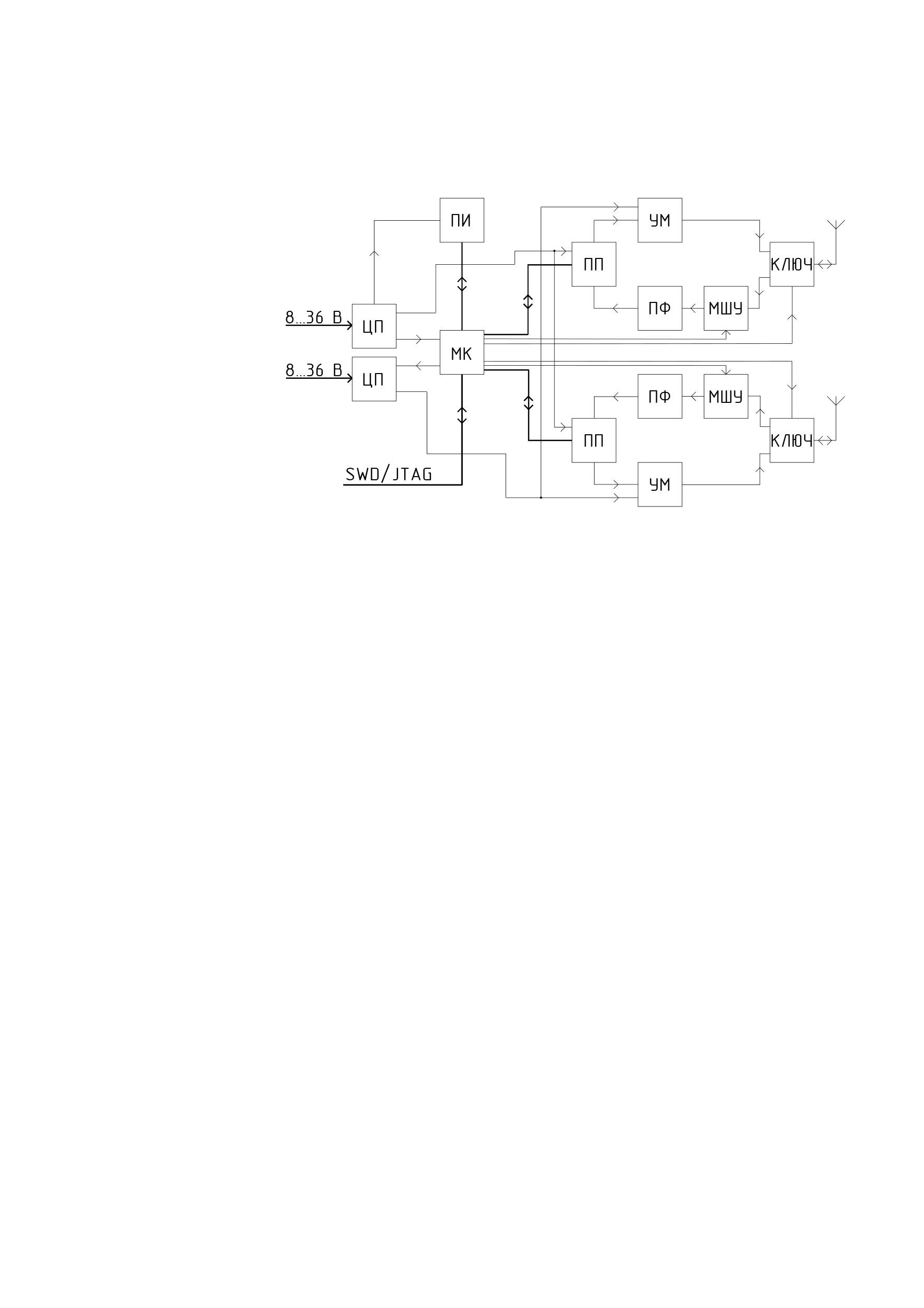


Рисунок 14 – Структурная схема устройства

ЦП – цепи питания; МК – микроконтроллер; ПИ – преобразователи интерфейсов; ИП – интерфейсы программирования; ПП – приемопередатчик; УМ – усилитель мощности; МШУ – малошумящий усилитель; ПФ – полосовой фильтр.

Основным управляющим узлом схемы является микроконтроллер. МК осуществляет обработку данных, принимаемых от приемопередатчика, регулирует работу МШУ, переключает ключ и управляет питанием определенных блоков. Так же он перенаправляет и контролирует поток данных, получаемых от пользователя через последовательные интерфейсы.

Радиотракт состоит из двух приемопередатчиков, выполненных в виде отдельных микросхем. В состав приемой части тракта также включены полосовой фильтр и малошумящий усилитель, а в состав передающей части усилитель мощности. На выходе тракта стоит ключ, управляемый микроконтроллером, для подключения выхода антенны либо к приемной части тракта, либо в передающей.

Данные для передачи поступают на МК через последовательные интерфейсы.

Цепи питания отвечают за преобразование входного напряжения к напряжению питания микросхем.

## 2.1.2 Разработка и анализ функциональной схемы

Ниже представлена разработанная функциональная схема устройства.

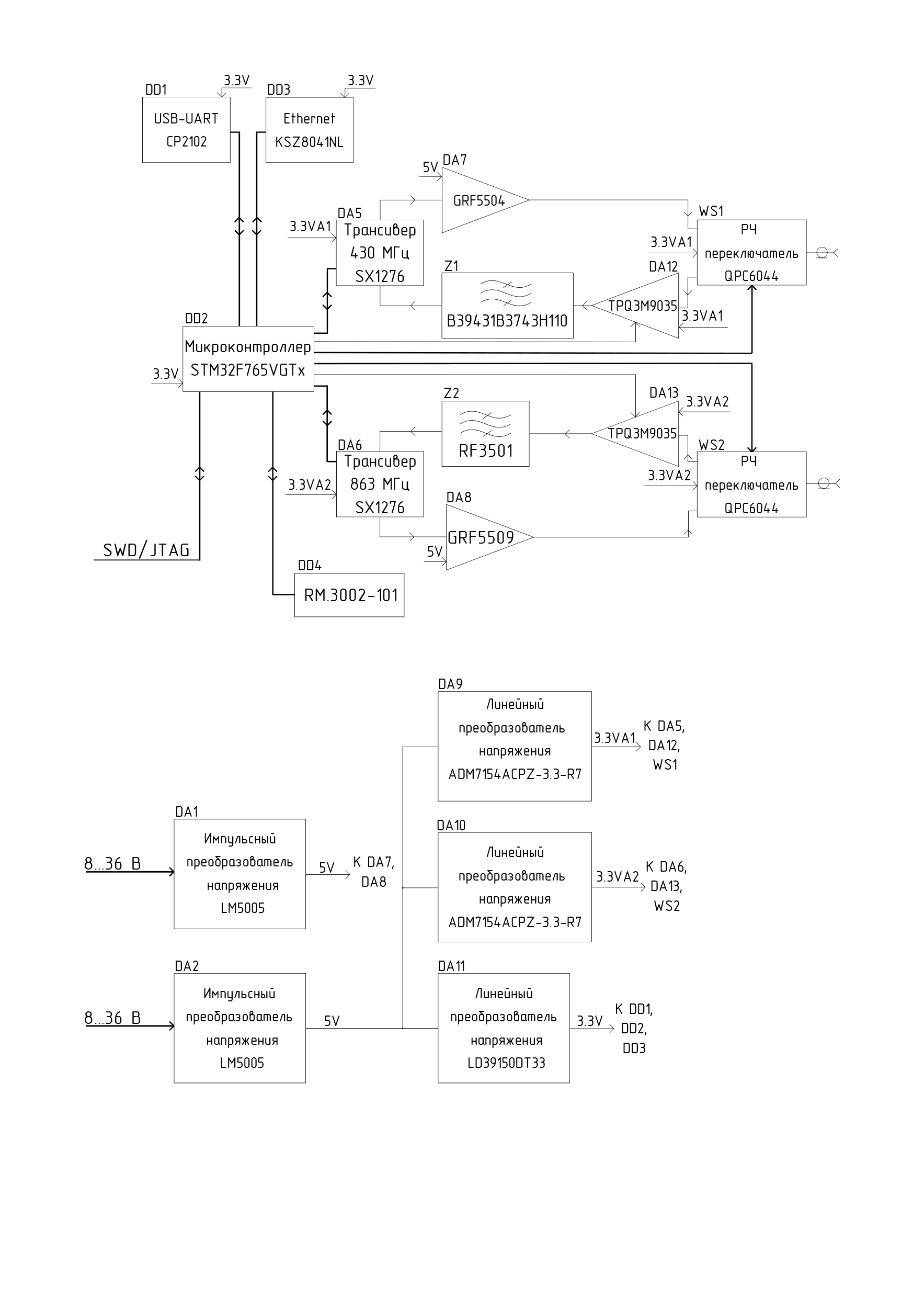


Рисунок 15 – Функциональная схема устройства

Функционально-технические характеристики и аппаратная реализация модема:

1. два радиотракта на основе радиомодуля SX1276;
2. микроконтроллер SMT32F7;
3. два импульсных регулятора напряжения LM5005;
4. Ethernet модуль KSZ804NL;
5. преобразователь интерфейсов CP2102-GMR;
6. CAN модуль MAX3051EKA;

Ниже представлено краткое описание каждого функционального узла и анализ его возможностей для последующей разработки ПО.

## 2.2 Радиомодуль SX1276

SX1276 представляет собой радиомодуль поддерживающий модуляции LoRa, FSK, OOK. Данный модуль может обеспечить бюджет канала связи до 168 дБ. Имеет высокую чувствительность (до -148 дБм). Малый ток потребления в режиме передачи.

Таблица 1 – характеристики модуля SX1276

|  |  |
| --- | --- |
| Поддерживаемые модуляции | GFSK, FSK , MSK, OOK, GMSK |
| Максимальный бюджет, дБ | 168 |
| Максимальная мощность выходного сигнала, дБм | 20 |
| Коэффициент усиления УМ, дБ | 14 |
| Скорость передачи данных, кбит/с | до 300 |
| Потребление в режиме приема, мА | 9.9 |
| Погрешность интегрированного синтезатора частот, Гц | 61 |
| Входной динамический диапазон, дБ | 127 |
| Размер полезной нагрузки, байт | 256 |
| Полоса частот выходного сигнала, МГц | 137 до 1020 |
| Диапазон частот, кГц | от 7,8 до 500 |
| Аппаратный CRC | есть |
| Детектирование преамбулы | есть |

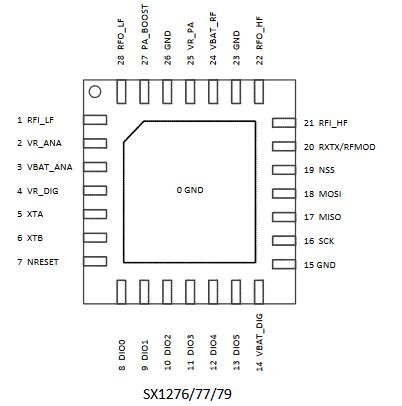


Рисунок 16 – Выводы микросхемы

Таблица 2 – Описание выводов микросхемы

|  |  |  |  |
| --- | --- | --- | --- |
| Номер вывода | Название | Направление | Описание |
| 0 | GROUND | – | Незащищенный вывод земли |
| 1 | RFI\_LF | Вход | РЧ вход для НЧ сигналов |
| 2 | VR\_ANA | – | Регулируемое напряжение питания для аналоговой схемы |
| 3 | VBAT\_ANA | – | Напряжение питания для аналоговой схемы |
| 4 | VR\_DIG | – | Регулируемое напряжение питания для цифровых блоков |
| 5 | XTA | Вход/Выход | Подключение кварцевого источника тактирования, или кварцевого источника тактирования с термокомпенсацией |
| 6 | XTB | Вход/Выход | Подключение кварцевого источника тактирования |
| 7 | NRESET | Вход/Выход | Сброс микросхемы |
| 8 | DIO0 | Вход/Выход | Цифровой ввод/вывод, программно настраиваемый |
| 9 | DIO1/DCLK | Вход/Выход | Цифровой ввод/вывод, программно настраиваемый |
| 10 | DIO2/DATA | Вход/Выход | Цифровой ввод/вывод, программно настраиваемый |
| 11 | DIO3 | Вход/Выход | Цифровой ввод/вывод, программно настраиваемый |

Продолжение таблицы 2

|  |  |  |  |
| --- | --- | --- | --- |
| 12 | DIO4 | Вход/Выход | Цифровой ввод/вывод, программно настраиваемый |
| 13 | DIO5 | Вход/Выход | Цифровой ввод/вывод, программно настраиваемый |
| 14 | VBAT\_DIG | – | Питание для цифровых блоков |
| 15 | GND | – | Земля |
| 16 | SCK | Вход | Тактирующая линия шины SPI |
| 17 | MISO | Выход | Линия выходных данных шины SPI |
| 18 | MOSI | Вход | Линия входных данных шины SPI |
| 19 | NSS | Вход | Линия выбора устройства шины SPI |
| 20 | RXTX/RF\_MOD | Выход | Переключатель управления приемник/передатчик |
| 21 | RFI\_HF(GND) | Вход (–) | РЧ вход для ВЧ сигналов (Земля) |
| 22 | RFO\_HF(GND) | Выход (–) | РЧ выход для ВЧ сигналов (Земля) |
| 23 | GND | – | Земля |
| 24 | VBAT\_RF | – | Напряжение питания для РЧ блоков |
| 25 | VR\_PA | – | Регулируемое напряжение питания для РЧ блоков |
| 26 | GND | – | Земля |
| 27 | PA\_BOOST | Выход | Вывод статуса усилителя |
| 28 | RFO\_LF | Выход | РЧ выход для НЧ сигналов |

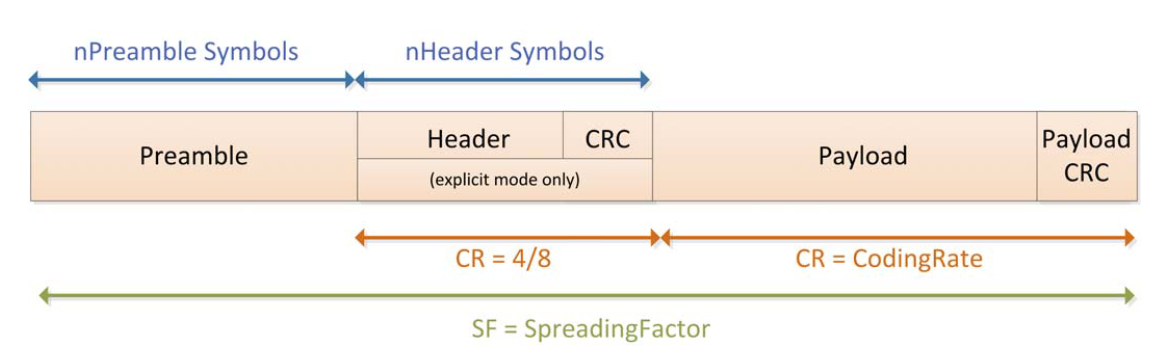


Рисунок 17 – Структура кадра

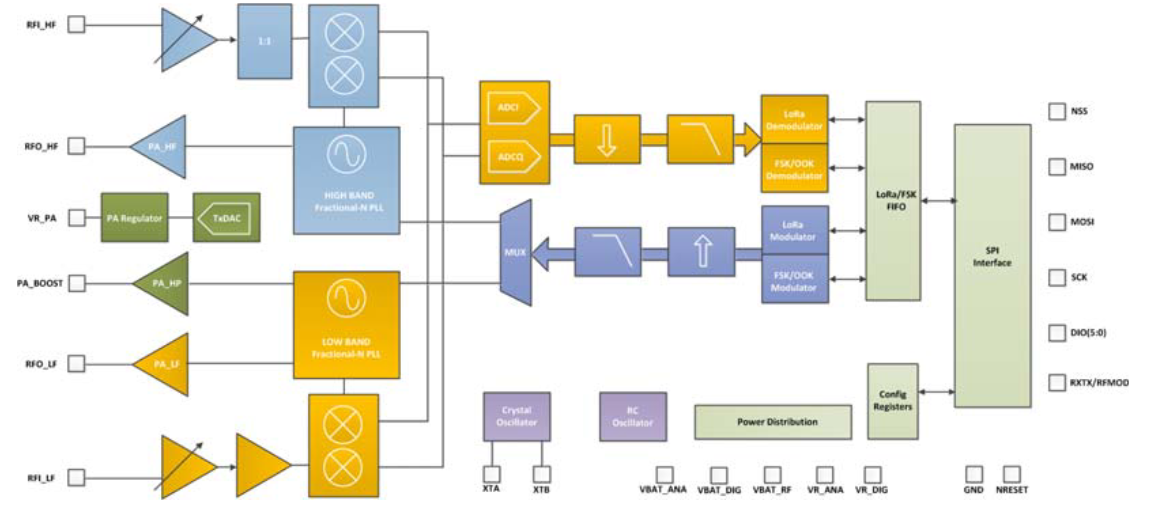


Рисунок 18 – Структурная схема SX1276

Данная микросхема аппаратно формирует передающий кадр. Весь цикл приема и передачи кадра выполняется также аппаратно, что позволяет снять нагрузку с микропроцессора. О приеме и окончания передачи кадра микросхема сообщает путем установки логического нуля на вывод DIO0. Микроконтроллер должен генерировать прерывания для детектирования данных событий. Программа должна либо обрабатывать все события прямо в обработчике прерываний, либо перенаправлять их для обработки в основную программу. Перенаправление в основную программу является более предпочтительным, так как в таком случае отсутствует возможность пропуска других прерываний. Для минимизации времени передачи данных через SPI SX1276 позволяет работать через данную шину c тактовой частотой 10 МГц. Также настройка встроенного в приемопередатчик усилителя мощности позволяет регулировать мощность передаваемого сигнала. Данная возможность позволит оптимально настроить мощность сигнала для последующего усиления в РЧ тракте.

## 2.3 Микроконтроллер STM32F7

STM32F7 микроконтроллер компании ST Microelectronics спроектированный на основе архитектуры ARM Cortex-M7. Имеет следующие характеристики:

Таблица 3 – Общие характеристики микроконтроллера

|  |  |
| --- | --- |
| Размер l кэша, кБ | 16 |
| Размер D кэша, кБ | 16 |
| Количество DMA контроллеров | 2 + 1 для Ethernet |
| Максимальная тактовая частота, МГц | 216 |
| Размер памяти для хранения данных в режиме низкого потребления, кБ | 4 |
| Размер памяти для обработки критических ко времени данных, кБ | 128 |
| Размер памяти для критичных ко времени инструкций, кБ | 16 |
| Модуль для вычислений с плавающей точкой | есть |

Таблица 4 – Периферия микроконтроллера

|  |  |
| --- | --- |
| Название | Количество |
| USB 2.0 | 2 |
| SDIO | 1 |
| USART | 4 |
| UART | 2 |
| SPI | 4 |
| I2C | 2 |
| CAN 2.0 | 1 |
| HDMI-CEC | 1 |
| Ethernet | 1 |
| Quad-SPI | 2 |
| I2S |  |
| SAI | 2 |
| 12-битные ЦАП | 2 |
| 16-битные таймеры |  |
| 32-битные таймеры |  |
| 12-битные АЦП | 3 |

Микроконтроллер является главным управляющим узлом устройства. Выполняет функцию приема, передачи и обработки данных от пользователя и радиомодемов. Наличие аппаратных таймеров позволяет синхронизировать сеанс радиосвязи между устройствами независимо от выполнения программы. При установке максимальной тактовой частоты микроконтроллер будет затрачиваться минимальное время на обработку данных. Выделение каждому радиомодулю отдельной шины SPI повысит скорость обмена данными с радиомодулем. Наличие контроллера DMA позволяет не затрачивать время МК при отправке и приеме данных с использованием последовательных портов.

## 2.4 Импульсный преобразователь напряжения LM5005

Высоковольтный понижающий преобразователь LM5005 обладает всеми функциями, необходимыми для реализации эффективного импульсного стабилизатора высокого напряжения с минимальным количеством внешних компонентов. Этот простой в использовании преобразователь работает в диапазоне входных напряжений от 7 В до 75 В и обеспечивает максимальный выходной ток 2,5 А. Архитектура контура управления основана на управлении по току с использованием эмулируемой рампы тока для обеспечения высокой помехоустойчивости. Управление по току обеспечивает встроенную прямую связь по линии, поцикловую защиту от перегрузки по току и простую компенсацию контура. Использование эмулируемой рампы управления снижает чувствительность схемы ШИМ к шуму, обеспечивая надежное управление небольшими рабочими циклами, необходимыми в приложениях с высоким входным напряжением. Частота коммутации программируется резистором в диапазоне от 50 кГц до 500 кГц. Чтобы уменьшить электромагнитные помехи, вывод синхронизации генератора позволяет нескольким регуляторам LM5005 выполнять самосинхронизацию или синхронизацию с внешним тактовым сигналом. Дополнительные функции защиты включают настраиваемый плавный пуск, отслеживание внешнего источника питания, отключение при перегреве с автоматическим восстановлением и возможность удаленного отключения.

Таблица 5 – Характеристики преобразователя напряжения

|  |  |
| --- | --- |
| Диапазон входного напряжения, В | от 7 до 75 |
| Регулируемое выходное напряжение, В | от 1,225 |
| Выходной ток, А | до 2,5 |
| Диапазон температур, °C | от -40 до 125 |
| Точность напряжения обратной связи | ±1,5 % |
| Частота переключения | от 50 кГц до 500 кГц |
| Минимальное включение ШИМ, нс | 80 |
| Внутренний высоковольтный стабилизатор напряжения смещения | есть |
| Дополнительный источник питания смещения для VCC | есть |

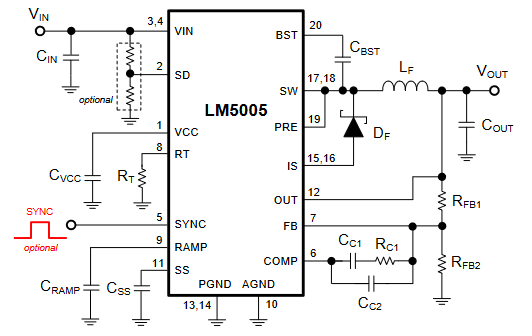


Рисунок 19 – Типовая схема включения преобразователя напряжения в схему

Данная микросхема позволяет регулировать напряжение питания обособленно от остальных элементов.

## 2.5 Приемопередатчик KSZ8041NL

KSZ8041NL — это приемопередатчик физического уровня, обеспечивающий MII/RMII интерфейсы для передачи и приема данных. Конструкция смешанных сигналов увеличивает расстояние передачи сигналов при одновременном снижении энергопотребления.

HP Auto MDI/MDI-X представляет собой наиболее надежное решение, устраняющее необходимость различать перекрестные и прямые кабели. KSZ8041NL представляет собой новый уровень функций и производительности и является идеальным выбором приемопередатчика физического уровня для приложений 10Base-T/100Base-TX.

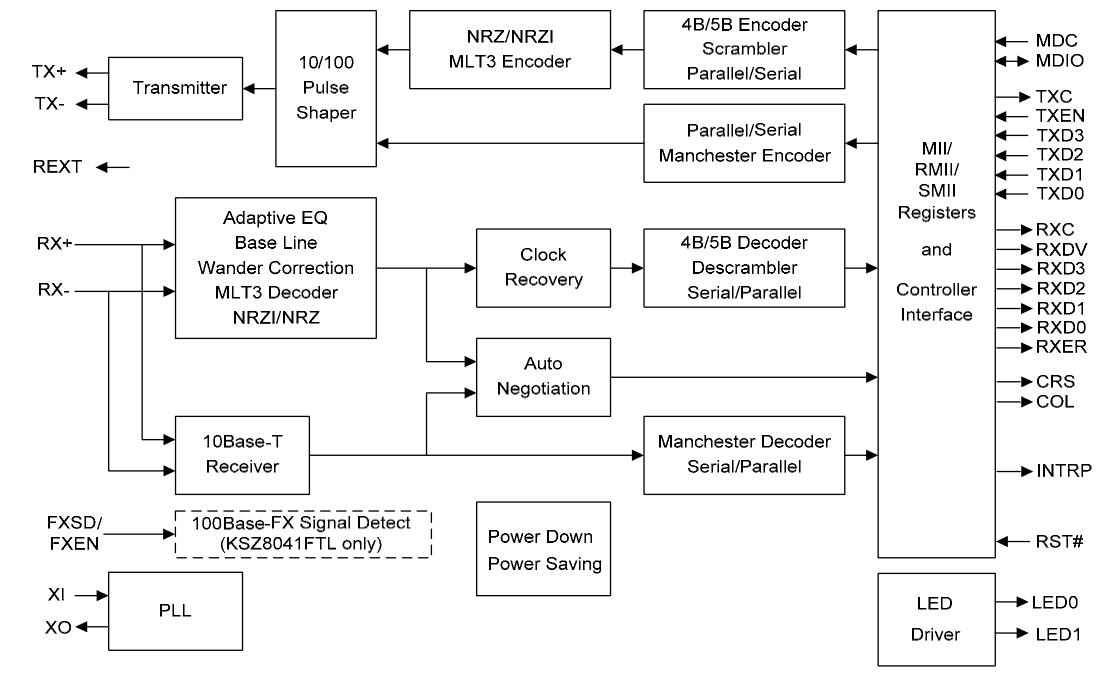


Рисунок 20 – Структурная схема приемопередатчика Ethernet

Наличие данной микросхемы позволяет увеличить скорость передачи данных от пользователя.

## 2.6 Преобразователь интерфейса CP2102-GMR

СР21хх – это двунаправленные преобразователи интерфейсов UART - USB (мосты). Они предназначены для добавления интерфейса USB в различные приборы, построенные на базе микроконтроллеров и не имеющих интерфейса USB, а также для обновления уже существующих приборов, требующих перехода на современный интерфейс USB.

Микросхемы СР21хх – это интегрированное решение, на одном кристалле расположены:

-контроллер USB интерфейса спецификации 2.0, работающий в режиме USB-device со скоростью передачи данных до 12Мбит/сек

-контроллер UART с поддержкой всех модемных сигналов (линий квитирования)

-память EEPROM для хранения настроек микросхемы (например, Vendor ID, Product ID, Serial Number, Max Power и др)

-раздельные буферы с объемом более 512б для приема и передачи данных  
-тактовый генератор на 48 МГц, поэтому в системе не потребуется внешний генератор

-регулятор напряжения позволяет подключать внешнюю нагрузку в режимах питания от шины.

Благодаря тому, что большинство элементов интегрировано на кристалле, для разработки преобразователей интерфейсов на основе микросхем СР21хх требуется минимальное количество внешних компонентов, только двух внешних фильтрующих конденсаторов на линию питания. Для обеспечения помехозащищенности микросхемы рекомендуется использовать подтягивающий резистор на 2 кОм (на линию RESET). Также, для стойкости к электростатическим разрядам можно использовать защитные диоды.

Данная микросхема позволяет подключать устройство к компьютеру через USB разъем и использовать его для передачи и приема данных.

## 2.7 CAN контроллер MAX3051

MAX3051 обеспечивает возможность дифференциальной передачи на шину и возможность дифференциального приема на контроллер CAN. MAX3051 в первую очередь предназначен для приложений с однополярным питанием +3,3 В, которые не требуют строгой защиты от сбоев, предусмотренной автомобильной промышленностью (ISO 11898). MAX3051 имеет четыре различных режима работы: высокоскоростной режим, режим управления наклоном, режим ожидания и режим отключения. Высокоскоростной режим позволяет передавать данные со скоростью до 1 Мбит/с. Режим управления наклоном можно использовать для программирования скорости нарастания передатчика для скорости передачи данных до 500 кбит/с. Это снижает влияние электромагнитных помех, что позволяет использовать неэкранированный витой или параллельный кабель.

В режиме ожидания передатчик выключен, а приемник находится на высоком уровне, переводя MAX3051 в слаботочный режим. В режиме выключения передатчик и приемник выключены.

## 2.8 Сравнение

Ниже рассмотрены устройства решающие следующие задачи:

- Передача командной информации с наземной станции управления на борт летательного аппарата.

- Передача телеметрической информации с борта летательного аппарата на наземную станцию управления

## 2.8.1 Командно-телеметрический радиомодем РМ-02М2

Командно-телеметрический радиомодем РМ-02М2 предназначен для двустороннего полудуплексного обмена данными между БПЛА и наземной станцией управления на расстояниях до 30 км.

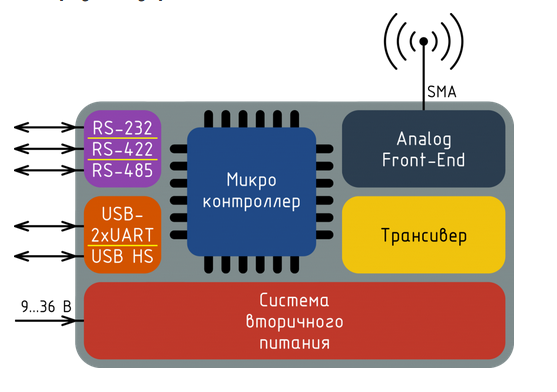


Рисунок 21 – Структурная схема радиомодема РМ-02М2

## 2.8.2 Командно-телеметрический радиомодем РМ-16

Командно-телеметрический радиомодем РМ-16М предназначен для двустороннего полудуплексного обмена данными между БПЛА и наземной станцией управления на расстоянии до 100 км.

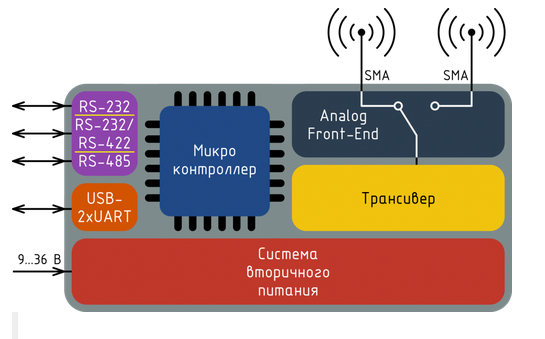


Рисунок 22 – Структурная схема модема РМ-16

## 2.8.3 Цифровой модем 3D Link

Цифровой модем 3D Link предназначен для организации видео- и командно-телеметрического каналов связи с малыми робототехническими комплексами в сложных условиях распространения радиосигнала.

Модем построен на базе уникальной технологии OFDM, способной формировать сигналы с высокой спектральной эффективностью и сигналы с расширенным спектром, что обеспечивает информационную скорость до 64 Мбит/с в видеоканале, а также отличную помехозащищенность в командно-телеметрическом канале. Отдельный модем для команд и телеметрии больше не нужен.

Видеоканал подключается через интерфейс Ethernet, канал команд и телеметрии — через интерфейсы Ethernet, RS232 или CAN.

Таблица 6 – Сравнение характеристик радиомодемов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Модем | Диапазон рабочих частот, МГц | Полоса занимаемых частот, кГц | Скорость передачи данных | Дальность действия в условиях прямой видимости, км | Выходная мощность, дБм |
| РМ-16 | 860-1020 | 500 | до 115,2 (1-30 кбит/с в режиме LoRa) | 100 | 30 |
| РМ-02М2 | 2400-2483,5 | 1000 | до 250 кбит/с | 30 | 23 |
| 3D Link | 2400-2500 | до 20 | 15-85 кбит/с | 83 | 29.5 |
| Используемое устройство | 430, 863 | до 500 | 21 кбит/с | 150 | 32 |

Сравнение характеристик модемов (Продолжение таблицы 6)

|  |  |  |  |
| --- | --- | --- | --- |
| Модем | Интерфейсы | Тип | Виды модуляции |
| РМ-16 | RS-232х2, RS-485, USB | Полудуплекс | FSK, GFSK |
| РМ02-М2 | RS-232, RS-422, RS-485, 2xUSB | Полудуплекс | FSK, GFSK |
| 3D-Link | Ethernet 100 Base-TX, RS-232, CAN, USB | Дуплекс | BPSK, QPSK, 8-QAM, 16-QAM, 64-QAM, 256-QAM |
| Используемое устройство | Ethernet 100 Base-TX, RS-232, CAN, USB | Дуплекс | FSK, LoRa |

# Разработка программно алгоритмического обеспечения

## 3.1 Выбор и обоснование используемых инструментов для разработки программно-алгоритмического обеспечения

Разрабатываемое приложение должно выполнять следующие действия:

- прослушивать последовательные интерфейсы, в ожидании новых данных и обрабатывать их;

- получать и обрабатывать данные получаемые от радиомодуля;

- отправлять данные в последовательные интерфейсы и радиомодуль;

- контролировать синхронизацию устройства;

Разработка проводилась с использованием языков С, C++. Язык С является стандартом разработки под архитектуру Cortex-M. Код написанный с использованием этого языка является экономным с точки зрения объема памяти, достаточно быстрым, оптимизированным. В нем есть возможность вставки ассемблерных иструкций и манипуляций с памятью микроконтроллера. Язык C++ включает в себя все возможности языка С и дополняет их возможностью написания объектно-ориентированного кода, что в свою очередь повышает скорость разработки, а также ее удобство.

Для более оптимального использования ресурсов процессора в разрабатываемом приложении используется операционная система реального времени (ОСРВ) FreeRTOS. Преимущество ОСРВ заключается в обработке возникающих событий с минимально возможной задержкой. FreeRTOS поддерживает приоритеты потоков, вытесняющую и кооперативную многозадачность. Имеет в составе ядра такие службы как семафоры, мьютексы, очереди. Данная операционная система поддерживает процессоры компании ARM. Код FreeRTOS очень компактен и в зависимости от платформы и настроек ядра занимает от 4 до 9 килобайт. Операционная система написана на языке С, с использованием ассемблерных вставок.

Приложение будет иметь несколько потоков, которые для удобства назовем «Поток радио», «Поток приложения», «Поток приема из последовательного порта» и «Поток передачи в последовательный порт».

Исходя из количества радиомодулей расположенных на печатной плате устройства приложение будет иметь два «Потока радио», которые будут выполнять задачи связанные с радимодулями. Для каждого из радиомодулей имеется свой «Поток приложения», который будет выполнять обработку данных поступающих из «Потока приема из последовательного порта», также в этом потоке будут обрабатываться данные получаемые из «Потока радио» и в обработанном виде отправляться в поток «Поток передачи в последовательный порт», если это необходимо.

## 3.2 Выбор и обоснование параметров радиомодема

В соответствии с техническим заданием скорость передачи данных через радиоканал должна быть не меньше 20 кБит/с. Для передатчика с модуляцией LoRa скорость передачи данных задается формулой

где SF – коэффициент расширения спектра, CR – скорость сверточного кодирования, BW – ширина полосы сигнала.

В таблице представлены параметры при которых значение пропускной способности больше или равно 20 кБит/с

Таблица 7 – Значения пропускной способности при различных настройках модема

|  |  |  |  |
| --- | --- | --- | --- |
| SF | CR | BW, кГц | BR, бит/с |
| 6 | 4/5 | 500 | 37500 |
| 6 | 4/6 | 500 | 31250 |
| 6 | 4/7 | 500 | 26786 |
| 6 | 4/8 | 500 | 23438 |
| 7 | 4/5 | 500 | 21875 |

Для радиомодема были выбраны следующие параметры: SF = 7, CR = 4/5, BW = 500 кГц. При выборе учитывалось то, что с ростом коэффициента SF происходит уменьшение вероятности битовой ошибки, а с уменьшением скорости сверточного кодирования увеличивается пропускная способность полезной нагрузки и уменьшается количество накладных расходов.

## 3.3 Разработка структурной схемы работы системы

На рисунке ниже представлена структурная схема разработанной программы. Ниже представлено ее описание.

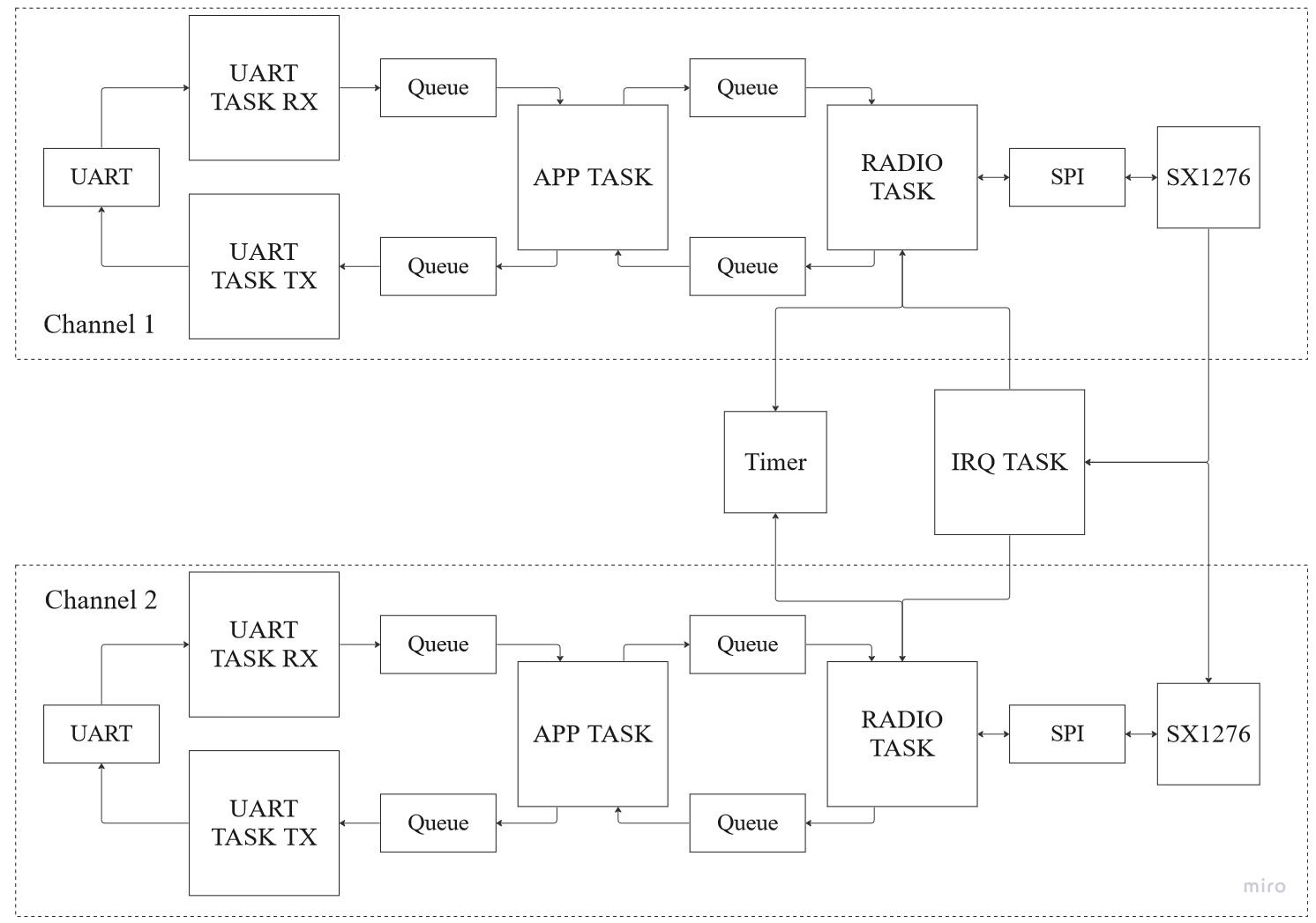


Рисунок 23 – Структурная схема работы программы

Модуль UART\_TASK\_RX отвечает за прием и обработку данных поступающих через последовательный порт UART. Модуль RADIO\_TASK взаимодействует с радиомодемом SX1276 через интерфейс SPI. Модули RADIO\_TASK и UART\_TASK\_RX отправляют данные для обработки через очереди операционной системы в модуль APP\_TASK, этот модуль, в свою очередь, обработанные данные может перенаправить, если это необходимо в модули UART\_TASK\_TX и RADIO\_TASK. Обработка прерываний поступающих от радиомодема, происходит в IRQ\_TASK.

## 3.3.1 Работа системы в дуплексном режиме

Ниже представлены временные диаграммы демонстрирующие принцип работы системы в дуплексном режиме.

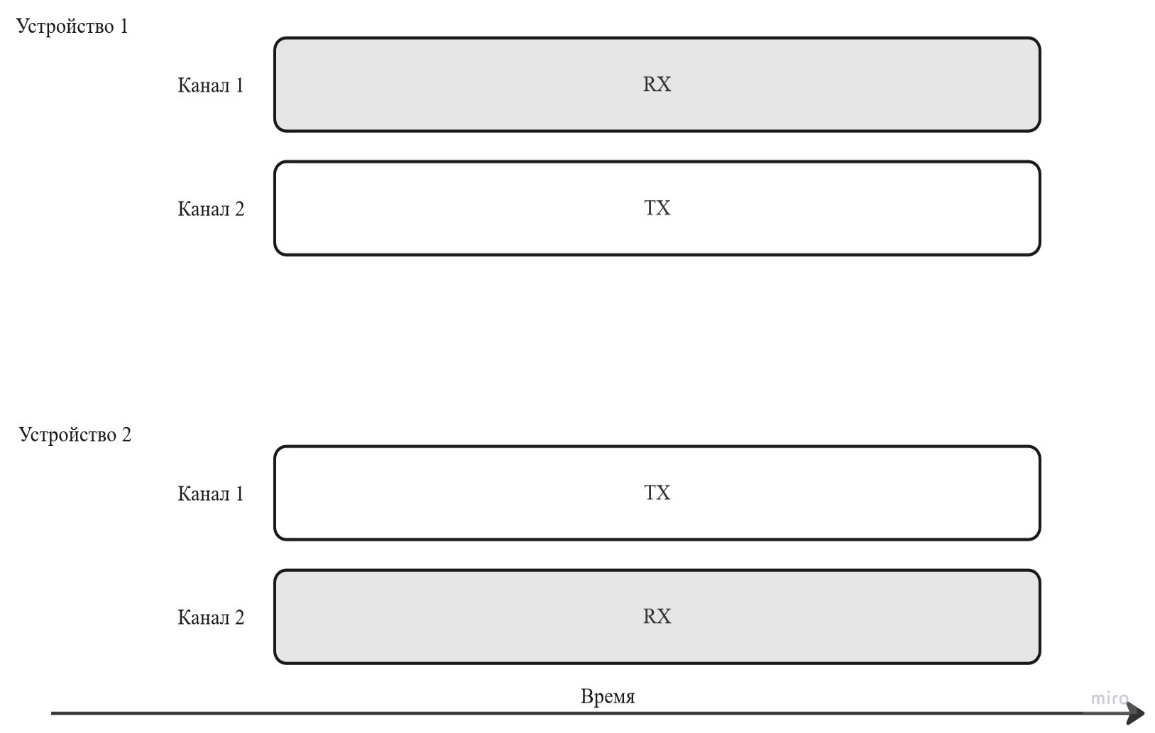


Рисунок 24 – Временные диаграммы

При работе устройства в дуплексном режиме работы на каждом из модемов один из каналов находится в режиме приема, а второй в режиме передачи. При таком режиме работы коллизии не возникают и не требуется никакой синхронизации устройств во времени.

В данном режиме работы устройствам необходимо определить кто из них ведущий, а кто ведомый. В зависимости от этого будет решено какое из устройств перейдет в режим приема, а какое будет ожидать данные для отправки. Процедура определения ведущего устройства выполняется при запуске устройств. Алгоритм определения ведущего устройства будет представлен позднее.

## 3.3.2 Работа системы в полудуплексном режиме

Ниже представлены временные диаграммы демонстрирующие работу системы в полудуплексном режиме.

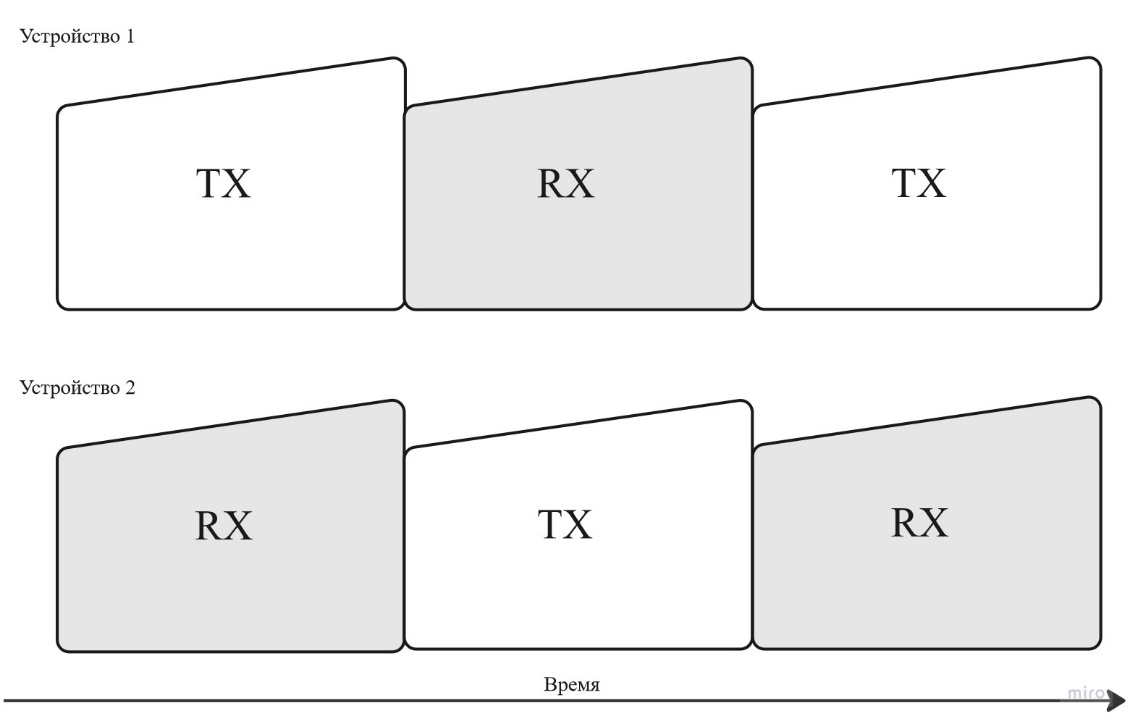


Рисунок 25 – Временные диаграммы

В полудуплексном режиме работы устройства каждый из каналов модемов работает как на прием, так и на передачу. В таком режиме работы возникает возможность коллизий во время передачи данных на модемах. Для их предотвращения был реализован метод разделения канала по времени. Необходимым условием этого метода является синхронизация шкал времени устройств, относительно которых распределяется время в эфире по временным слотам на прием и на передачу.

В данном режиме работы устройствам, как и в случае дуплексного режима, необходимо определить кто из них ведущий, а кто ведомый. Во время определения устройства в сообщении также будут отправлять свою временную метку, для того чтобы помимо определения ведущего устройства выполнялась синхронизация устройств. Процедура определения ведущего устройства выполняется при запуске устройств. Алгоритм определения ведущего устройства будет представлен ниже.

## 3.4 Алгоритм определения ведущего устройства по радиоканалу

На рисунке ниже представлен разработанный алгоритм опеределения ведущего устройства по радиоканалу.

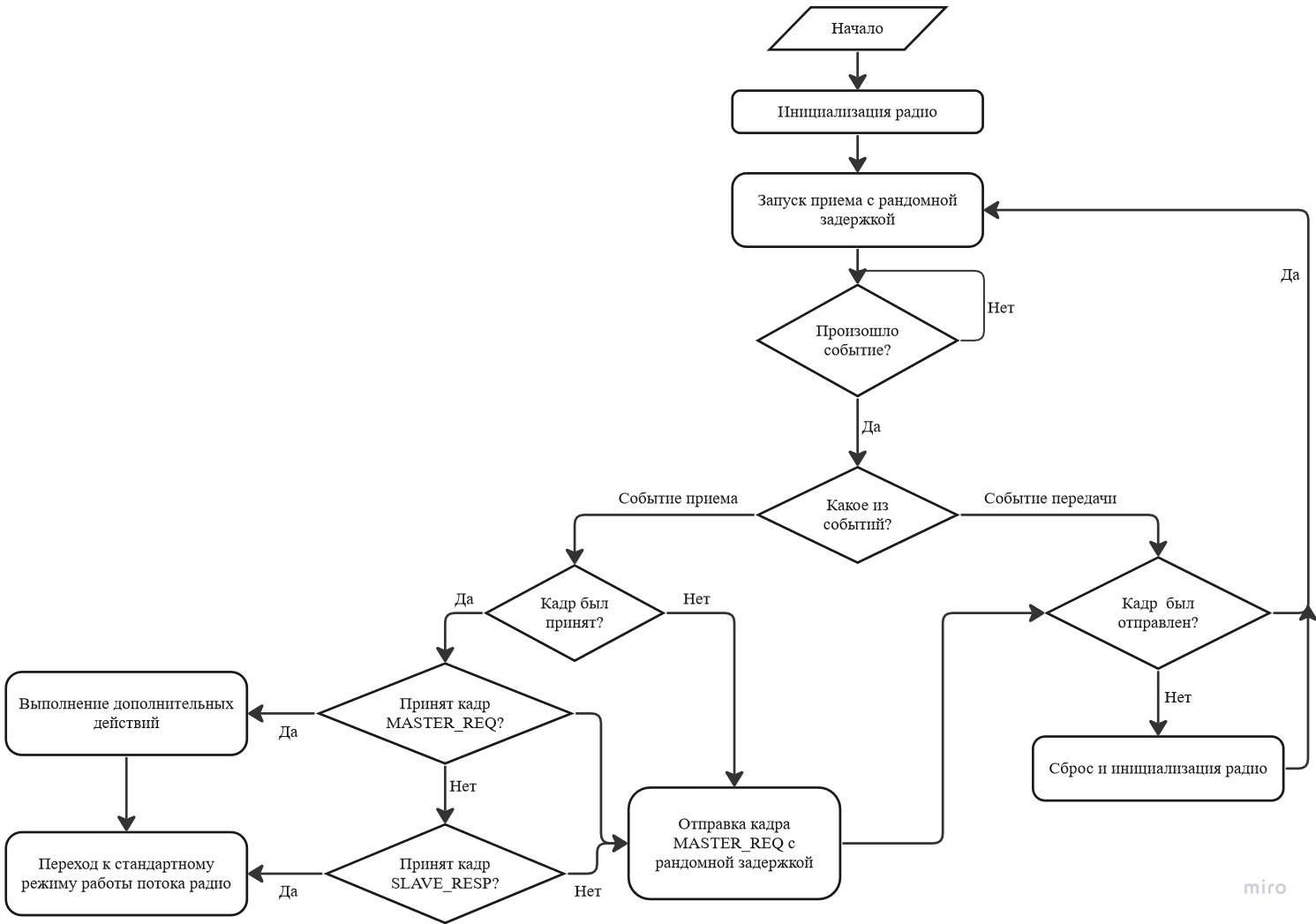


Рисунок 26 – Блок–схема алгоритма определения ведущего устройства по радиоканалу

Так как первоначально радиомодем SX1276 был выключен требуется произвести его инициализацию. Каждое из устройств на этом этапе имеет одинаковые права. После каждое из устройсв генерирует случайное число, уходит в сон на это значение времени и после пробуждения запускает приём с заданным тайм-аутом. Далее программа ожидает прерывания от радиомодема, сигнализирующее либо о событии приема (успешный прием, истекший тайм-аут приема, ошибка приема), либо о событии передачи (успешная передача, истекший тайм-аут передачи). Далее рассмотрим действия программы в зависимости от принятого события.

**Успешный прием**

Программа анализирует принятый кадр. Если принятый кадр является кадром MASTER\_REQ, то программа выполняет дополнительные действия и переходит в стандартный режим работы. В случае полудуплексного режима работы выполняется корректировка синхронизирующего таймера, в случае дуплексного режима работы никаких действий выполнять не надо. Устройство принявшее кадр MASTER\_REQ становится ведомым устройством, отправляет кадр SLAVE\_RESP и переходит в стандартный режим работы. Если принятый кадр был кадром SLAVE\_RESP, то устройство остается ведущим и переходит к стандартому режиму работы. Если же принятый кадр не является ни кадром MASTER\_REQ, ни кадром SLAVE\_RESP, то устройство перезапускает аппаратный таймер и отправляет кадр MASTER\_REQ, продолжая считать себя ведущим устройством.

**Успешная передача**

Программа переводит радиомодем в режим приёма.

**Истечение тайм-аута передачи**

Данное событие связано с аппаратной невозможностью радиомодема отправить кадр, радиомодем сбрасывается и инициализируется по новой. После инициализации делается рандомная задержка и запускается прием на заданное время.

**Истечение тайм-аута приема**

Время отведенное под прием закончилось. Кадр не был принят. Выполняется перезапуск аппаратного таймера и отправляется кадр MASTER\_REQ.

**Ошибка приема**

Не сошлась контрольная сумма. Выполняются те же действия что и для события истекшего тайм-аута приема.

## 3.5 Расчёт корректировки синхронизирующего таймера

В случае полудуплексного режима работы устройствам необходимо контролировать синхронизацию. В процессе синхронизации возникает необходимость в корректировке значения времени синхронизирующего таймера. На рисунке ниже представлены временные диаграммы, позволяющие лучше понять, как производится корректировка синхронизирующего таймера.

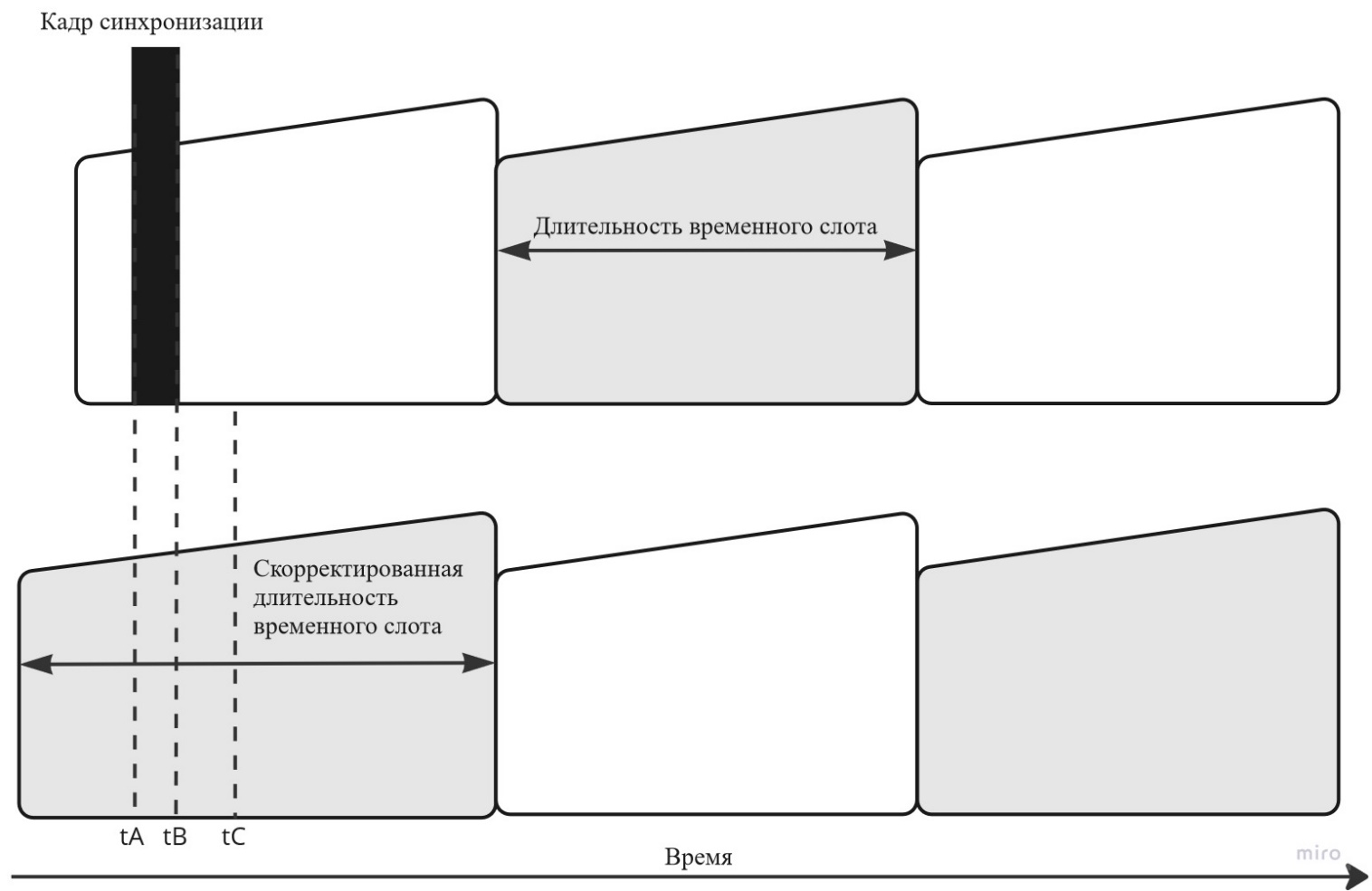


Рисунок 27 –Временные диаграммы переключения временных слотов

В расчёте участвуют три временных точки: *t*A, *t*B, *t*C, а также время распространения кадра *timeOnAir*. Время *t*A является временем отправки синхронизирующего кадра. Это время выставляется в значении временной шкалы ведущего устройства:

Время *t*B рассчитывается исходя из известного времени затрачиваемого на распространение синхронизирующего кадра от передатчика к приемнику *timeOnAir* и времени *t*A:

Далее, чтобы посчитать оценку времени на стороне ведомого устройства необходимо учесть время *dTick*, затраченное на обработку синхронизирующего кадра. Это время рассчитывается исходя из разности времени приема прерывания от приемопередатчика и текущего времени, данное значение времени рассчитывается на стороне ведомого устройства. Итоговое, оценочное значение времени ведущего устройства на стороне ведомого:

# Экспериментальные исследования

4.1 Анализ канала связи

Канал связи имеет следующие характеристики:

1. выходная мощность передатчика Pвых = 30 дБ;
2. потери в радиотракте передатчика Ltx = 3 дБ;
3. коэффициент усиления передающей антенны Gtx = 2 дБи;
4. потери в радиотракте приемника Ltx = 3 дБ;
5. коэффициент усиления приемной антенны Grx = 2 дБи;
6. запас по энергетике канал Lm = 20 дБ;

Расчёт значений вероятности битовой ошибки может быть выполнен по формуле:

где *P*b – значение вероятности битовой ошибки, *q* – номер итерации операции суммирования, *SF* – коэффициент расширения спектра, *Es* – энергия бита, *N*0 – спектральная плотность мощности аддитивного белого гауссовского шума.

Однако как показано в [6] при больших значениях SF (10, 11, 12) возникают проблемы с точностью расчёта, из-за больших значений факториалов в биномиальных коэффициентах. Поэтому для расчёта предлагается использовать следующее аппроксимированное выражение:

где *P*b – значение вероятности битовой ошибки, Q – хвостовая функция нормального распределения, Г – отношение сигнал шум

Точность такого подхода продемонстрирована [6] и изображена на рисунке ниже (синий график – вероятности полученные путем численных методов вычисления исходного выражения, черный график – вероятности вычисленные с помощью аппроксимированного выражения). Выбор данного метода аппроксимации связан с высокой точностью (расхождение от результатов полученных численным методом меньше 0.5 дБ) и простотой расчета.

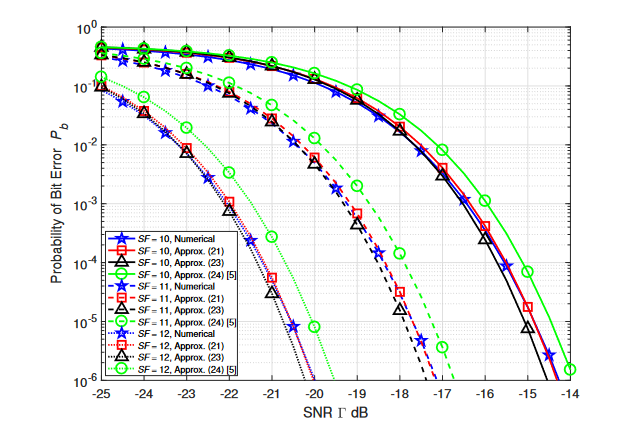


Рисунок 28 – Сравнение результатов для вероятности ошибки полученных путем аппроксимации

Как видно из рисунка выбранный метод аппроксимации (чёрный график) имеет незначительные расхождения с результатами полученными путем численного вычисления (синий график)

Для расчета бюджета канала связи сначала необходимо определить чувствительность приемника, которая вычисляется из необходимого отношения сигнал/шум для заданной вероятности битовой ошибки (BER). На рисунке ниже представлены графики зависимости вероятности битовой ошибки от отношения сигнал/шум (ОСШ) для используемой модуляции LoRa и различных коэффициентов расширения спектра, для канала прямой видимости с аддитивным

белым Гауссовым шумом (АБГШ)

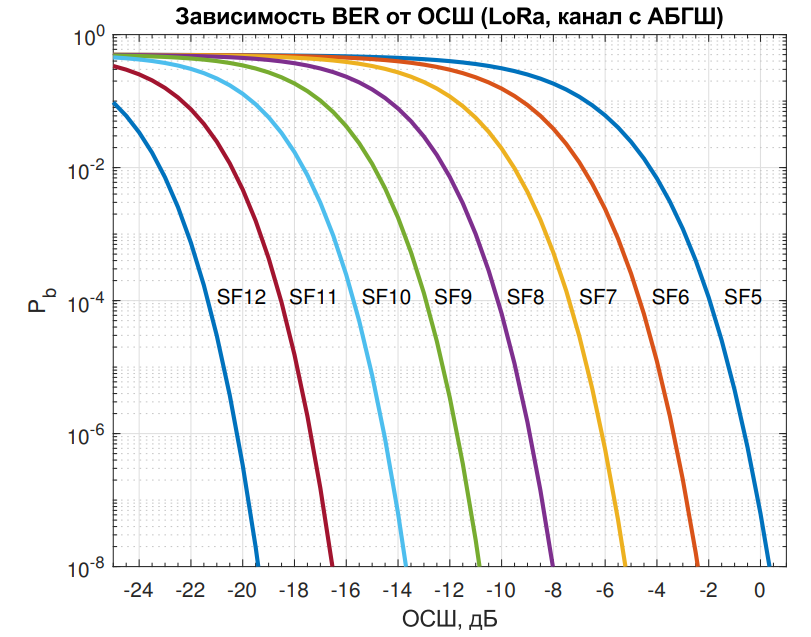


Рисунок 29 – Зависимость BER от ОСШ для канала с АБГШ

При различных коэффициентах расширения спектра для типичной вероятности битовой ошибки командно-телеметрических радиоканалов BER = 10-8, приведены данные в таблицы 8.

Таблица 8 – Отношения сигнал/шум

|  |  |  |
| --- | --- | --- |
| SF | Чип/символ | Порог ОСШ, дБ |
| 5 | 32 | 0,40 |
| 6 | 64 | -2,40 |
| 7 | 128 | -5,20 |
| 8 | 256 | -8,00 |
| 9 | 512 | -10,85 |
| 10 | 1024 | -13,70 |
| 11 | 2048 | -16,50 |
| 12 | 4096 | -19,40 |

Заданным коэффициентам расширения спектра также соответствуют различные скорости передачи данных *BR* (бит/с):

где SF – коэффициент расширения спектра;

CR – относительная скорость сверточного кодирования;

BW – ширина полосы частот сигнала, Гц.

Для дальнейших расчетов в соответствии с разделом 3.2 принимается скорость сверточного кодирования CR = 4/5, коэффициент расширения равный 7 (SF7), BW = 500 кГц. Для простоты последующего эксперимента, проведем расчёт бюджета канала связи при BER = 10-2 и соответствующем ему значении *SNR* = -9.5 дБ.

Чувствительность приёмника (в дБ) для заданных отношений сигнал шум (SNR) определяется по формуле:

Где *k* = 1.38 10-23 Дж/К – постоянная Больцмана

*T* – температура приёмника, К

*BW* – ширина полосы приемника, Гц

*NF* – коэффициент шума МШУ приёмника, дБ

Зная чувствительность приемника, можно перейти к расчету канала связи между наземной станцией управления (НСУ) и БПЛА.

Для взаимности радиоканала примем выходную мощность передатчиков радиомодемов БПЛА и НСУ равной, что позволяет рассчитать радиоканал только для одного направления передачи (с БПЛА на НСУ или с НСУ на БПЛА). Расчет бюджета канала связи производится по формуле 4:

Где L – это потери в радиоканале без учета потерь при распространении в свободном пространстве, а Lfs – допустимые для данного радиоканала потери в свободном пространстве.

Из этих уравнений можно вывести максимальную дальность передачи данных d для заданных чувствительностей приемника Prx min:

Для расчета было принято *T* = 300 K, *NF* = 3 дБ, *L*m = 15 дБ

## 4.2 Расчет бюджета канала связи наземная станция управления – беспилотный летательный аппарат 430 МГц

Минимальная чувствительность приёмника равна:

Потери в канале связи без учета потерь при распространении в свободном пространстве:

Бюджет канала связи будет равен:

Максимальная дальность связи будет равна:

## 4.3 Расчет бюджета канала связи наземная станция управления – беспилотный летательный аппарат 863 МГц

Минимальная чувствительность приёмника равна:

Потери в линии связи без учета потерь при распространении в свободном пространстве:

Бюджет канала связи будет равен:

Максимальная дальность связи будет равна:

**4.4 Разработка алгоритма для тестирования частоты потерь информационных пакетов**

Блок-схема разработанного алгоритма представлена ниже

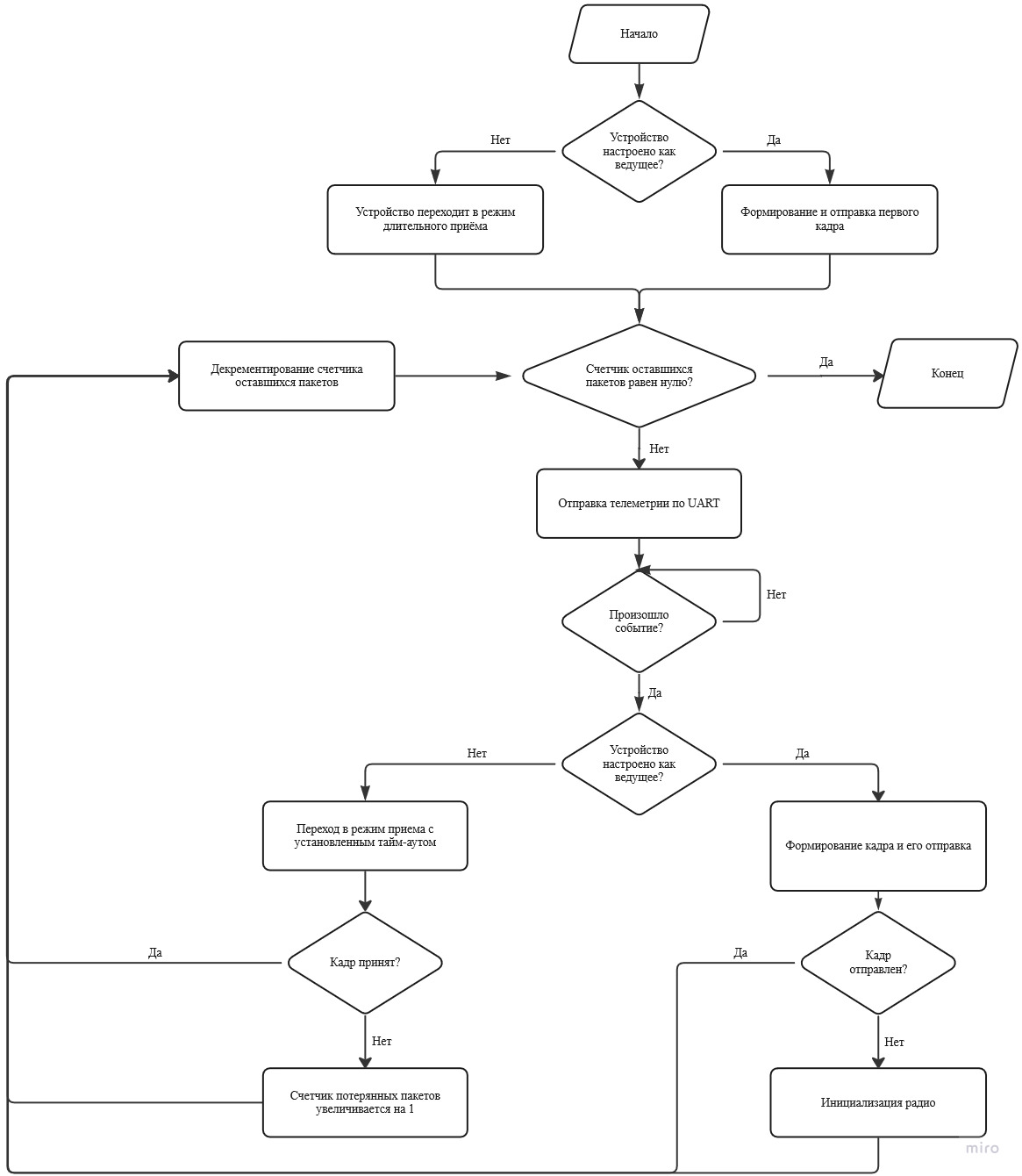


Рисунок 30 – Блок схема алгоритма тестирования частоты потерь информационных пакетов

При получении запроса на тестирование частоты потерь информационных пакетов (PER) от пользователя, программа переходит в режим тестирования PER.

В запросе отправляемом пользователем передается информация со следующими настройками сеанса связи:

- флаг iAmMaster определяющий ведущее устройство;

- количество пакетов для передачи/приема устройством;

- длина передаваемых пакетов в байтах;

- время в миллисекундах отводимое под приём (игнорируется у ведущего устройства);

- время задержки между передачами (игнорируется у ведомого устройства);

В зависимости от состояния флага iAmMaster устройство либо переходит в режим длительного приёма, либо формирует и отправляет первый кадр. После этого идет отправка телеметрии пользователю с текущим состоянием программы, для потдверждения того, что программа обработала запрос. Далее программа ожидает событие от радиомодема. Для ведущего устройства будут возникать только события приёма (успешный прием, истечение тайм-аута приема, ошибка приема). Для ведомого устройства только события передачи (удачная передача, истечение тайм-аута передачи). Возможны следующие варианты поведения программы, в зависимости от полученного события:

**Успешный прием**

Декрементирование счетчика оставшихся пакетов. Переход на следующую иттерацию.

**Истекший тайм-аут приема**

Инкрементирование счетчика потерь. Декрементирование счетчика оставшихся пакетов. Переход на следующую иттерацию.

**Ошибка приема**

Выполняются те же действия что и для события истекшего тайм-айта приема.

**Удачная передача**

Переход на следующую иттерацию.

**Истекший тайм-аут передачи**

Инициализация радио. Переход на следующую иттерацию.

В начале следующих иттераций программа отправляет телеметрию пользователю и в зависимости от состояния флага iAmMaster либо переходит в режим приема с заданным пользователем тайм-аутом, либо формирует кадр заданной длины и отправляет его.

Программа находится в состоянии PER до тех пор пока счетчик оставшихся пакетов не равен нулю, или пока не пришел новый запрос от пользователя.

## 4.5 Экспериментальная проверка дальности связи

Настройки радиоканалов, следующие приведены в таблице

Таблица 9 – Настройки радиоканалов

|  |  |
| --- | --- |
| Частота сверточного кодирования | CR4/5 |
| Коэффициент расширения спектра | SF7 |
| Выходная мощность передатчика, дБм | 30 |
| Ширина спектра, кГц | 500 |
| Длина преамбулы, символов | 8 |
| Модуляция | LoRa |

Частоты каналов 430 МГц и 863 МГц.

Координаты передатчика: 56⁰04’10.9’’N 92⁰37’40.7’’E

Координаты приемника: 56⁰03’52.2’’N 92⁰39’03.4’’E

Устройства были разнесены на 1.54 км. Далее на выходы РЧ трактов накручивались аттенюаторы

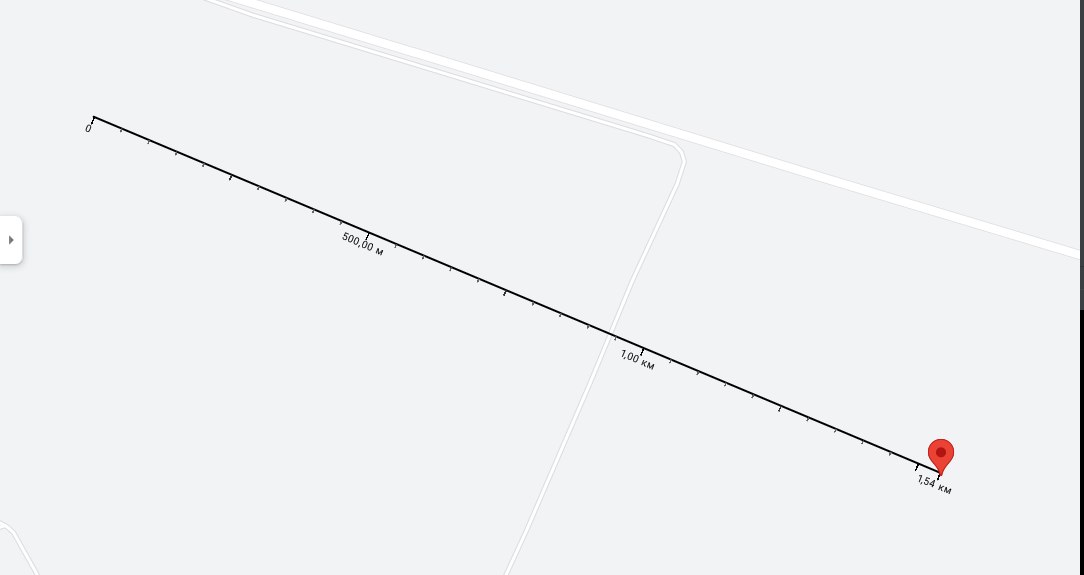


Рисунок 31 – Расположение устройств

На выходе каждого из РЧ-трактов были накручены аттенюаторы с вносимым затуханием Latten = 42 дБ.

Затухание, создаваемое распространением сигнала в свободном пространстве для сигнала частотой 430 МГц в соответствии с (9):

Lfs430 = 88.8698 дБ

Аналогично для сигнала частотой 863 МГц

Lfs863 = 94.9206 дБ

Затухание с учетом аттенюаторов для 430 МГц:

L430∑ = Lfs430 + Latten = 130.8698

Затухание с учетом аттенюаторов для 863 МГц:

L863∑ = Lfs863 + Latten = 136.8698

Эквивалентное расстояние в соответствии с формулой (11) для 430 МГц:

d430 = 193.87 км

Эквивалентное расстояние в соответствии с формулой (11) для 863 МГц:

d863 = 192.74 км

PER во время тестов не превышал 4% на канале с частотой 863 МГц, на канале 430 МГц во время тестов потерь не наблюдалось. Для канала 863 МГц потери при тестах могут быть связаны с наличием помех, а также с тем, что достигнутое расстояние превышает расчетное. Для канала 430 МГц из-за

отсутствия необходимого количества аттенюаторов не удалось достигнуть максимального эквивалентного расстояния.

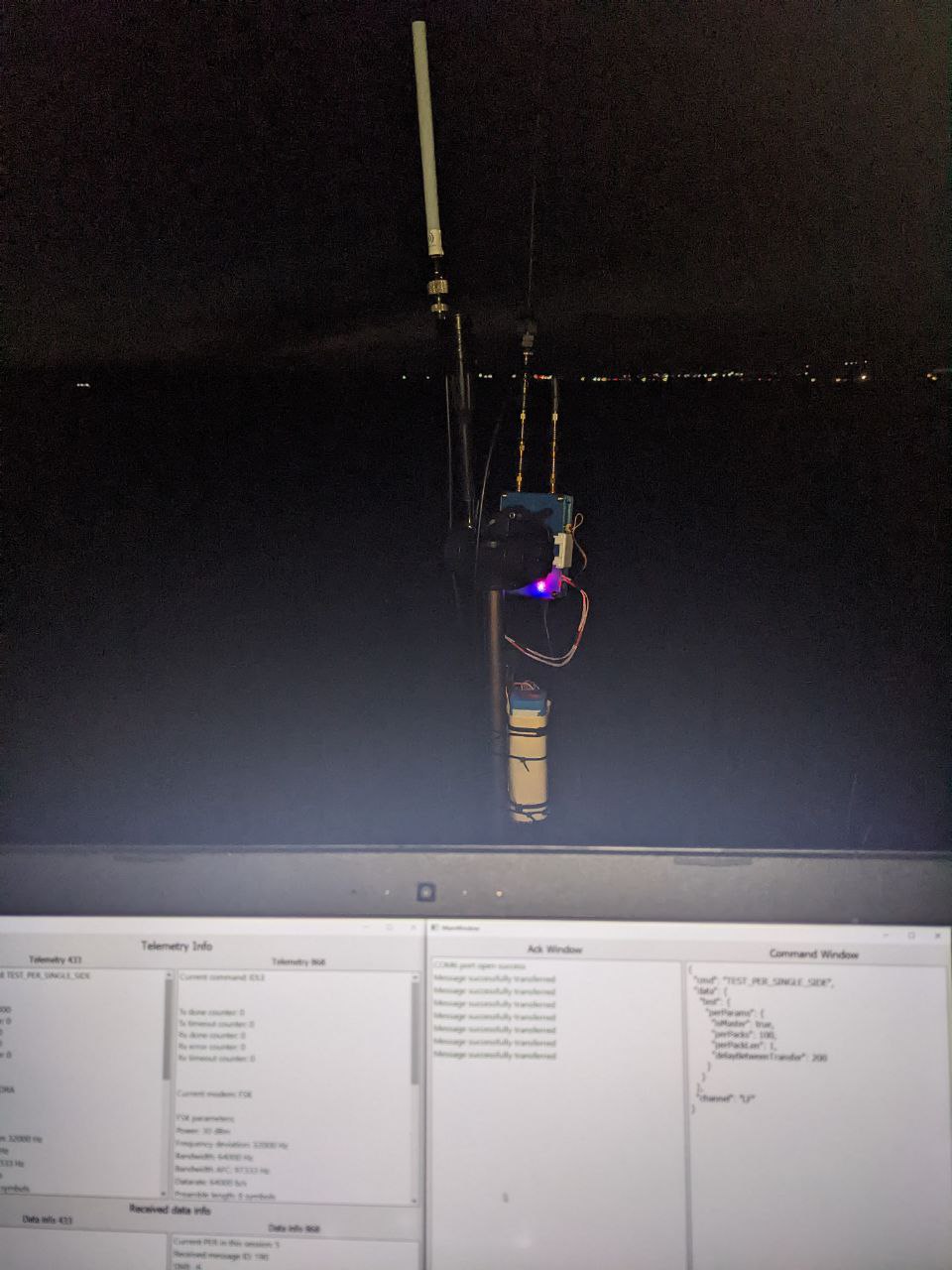


Рисунок 32 – Передающая сторона



Рисунок 33 – Приемная сторона

# Безопасность и экологичность

## Оценка опасных и вредных факторов

При разработке устройства основная часть работы приходится на разработку принципиальных схем, разработку печатной платы и написание программного обеспечения. Все перечисленные этапы разрабатываются на персональном компьютере, поэтому важно правильно оценить его воздействие на организм. В общем случае технологический процесс, выполняемый на персональной ЭВМ, представляет собой последовательность технологических операций, необходимых для выполнения определенных видов работ.

Технологических операций, выполняемых при работе за компьютером, можно разделить на две группы [13], представленные в таблице 10.

Таблица 10 – Описание технологического процесса

|  |  |  |  |
| --- | --- | --- | --- |
| № | Технологическая операция | Применяемое оборудование | Материалы |
| 1 | Ввод текста | Клавиатура | Каркас клавиатуры выполнен из стали, рамка клавиатуры – из пластика |
| 2 | Просмотр текстовых документов | Монитор | Дисплей жидкокристаллический,  корпус выполнен из пластика, так же в устройстве присутствует металл |

На основе анализа технологического процесса и рабочего места согласно ГОСТ 12.0.003-90 ССБТ «Опасные и вредные производственные факторы. Классификация» [14] можно выделить опасные и вредные факторы, которые могут привести к травматизму и профзаболеваниям:

- повышенный уровень электромагнитных излучений;

- повышенный уровень статического электричества;

- повышенное значение напряжения в электрической цепи;

- повышенная температура поверхностей персональной ЭВМ:

- выделение в воздух рабочей зоны химических веществ;

- повышенная/пониженная влажность воздуха;

- отсутствие или недостаток естественного освещения;

- недостаточное искусственное освещение рабочей зоны;

- повышенная яркость света;

- зрительное напряжение;

- монотонный труд;

- нервно-эмоциональные перегрузки.

Требуемые значения основных показателей опасных и вредных факторов при работе на ЭВМ [15] приведены в таблице 11.

Таблица 11 – Значения показателей опасных и вредных факторов

|  |  |  |
| --- | --- | --- |
| Операция | Опасные и вредные факторы | Нормируемое значение показателя |
| Выполнение квалификационной работы на ЭВМ | Повышенный уровень  шума | < 60 дБА |
| Недостаток естественного освещения | 300-500 лк |
| Опасный уровень напряжения в цепи | < 42 В переменный  < 110 В постоянный ток |

## Влияние ПЭВМ на организм человека

Вредные факторы при работе с монитором. **Работа на персональном ЭВМ сопровождается постоянным и значительным напряжением функций зрительного анализатора.** Это может стать причиной развития зрительного утомления, способствующего возникновению близорукости, головной боли, раздражительности, нервного напряжения и стресса. Увеличивается не только нагрузка на глаза, но и на шею, спину, плечи и руки, что приводит к быстрому утомлению. Из всех недомоганий, обусловленных работой на компьютерах, чаще встречаются те, которые связаны с использованием клавиатуры. Перечисленные выше факторы, могут в худшую сторону отразиться на работоспособности пользователя при работе с персональным ЭВМ, а это, в свою очередь, может привести к неисправимым последствиям.

Нервное напряжение.У людей, выполняющих работу на вычислительной технике, по сравнению с другими профессиональными группами выявлено значительно более выраженное нервно-сенсорное напряжение. Оно возникает вследствие дефицита времени, большого объема и плотности информации, особенностей диалогового режима общения человека и персональной ЭВМ, ответственности за безошибочность информации. Продолжительная работа на дисплее, особенно в диалоговом режиме, может привести к нервно-эмоциональному перенапряжению, нарушению сна, ухудшению состояния, снижению концентрации внимания и работоспособности, хронической головной боли, повышенной возбудимости нервной системы, депрессии [16].

При работе в условиях повышенных нервно-эмоциональных и физических нагрузок гиповитаминоз, недостаток микроэлементов и минеральных веществ (особенно железа, магния, селена**) ускоряет и обостряет восприимчивость к воздействию вредных факторов окружающей и производственной среды, нарушает обмен веществ, ведет к изнашиванию и старению организма.** Поэтому при постоянной работе наЭВМ для повышения работоспособности и сохранения здоровья к мерам безопасности относится защита организма с помощью витаминно-минеральных комплексов, которые рекомендуется применять всем, даже практически здоровым пользователям.

## 5.3 Нормативные требования при организации работы на ПЭВМ

Нормирование шума производится в соответствии с санитарными нормами СН 2.2.4/2.1.8.562-96, согласно которым рассматриваются предельно допустимые уровни звукового давления, уровни звука и эквивалентные уровни звука, значения которых предоставлены в таблице 12.

Таблица 12– Предельные спектры допустимых уровней звукового давления

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Уровни звукового давления, дБ, в октавных полосах со среднегеометрическими частотами, Гц | | | | | | | | | Уровни звука и эквивалентные уровни звука (в дБА) |
| 31,5 | 63 | 125 | 250 | 500 | 1000 | 2000 | 4000 | 8000 | 60 |
| 93 | 79 | 70 | 68 | 58 | 55 | 52 | 52 | 49 |

Учитывая тот факт, что вклад в общий шум вентиляторов системы охлаждения во много раз больше вклада жесткого диска, а это означит то, что последним при расчете можно пренебречь. Помимо этого, следует уделить внимание тому, что в зависимости от уровня нагрузки системы, будет зависеть

уровень мощности системы охлаждения, и, следовательно, уровень звукового давления будет не постоянным.

Освещение. Характер зрительной работы с монитором ‑ средней точности, соответственно по конструктивному оформлению подходит общее освещение. К выбору типа светильника необходимо подойти осознанно, качество светильника, в первую очередь, определяется по спектральному составу ‑ диапазону цветовой температуры, солнечное освещение имеет плотный и практически равномерный спектр, хороший светильник должен быть максимально приближен по этой характеристике. Должна быть обеспечена широта и равномерность спектра. Выбор падает на люминесцентные лампы ЛБЦТ, ЛДЦ, ЛДЦ УФ с хорошими спектральными характеристиками (3500-6000 К), приближенными к солнечному спектру, при этом, потребляя относительно малое количество электроэнергии от 50 до 70 Лм/Вт. Целесообразность расчётов освещения является малоэффективным, так как КПА может быть перевезено в другое помещение, в котором должно обеспечиваться освещение порядка 350-550 лк. Характеристика зрительной работы и освещенность искусственного освещения рабочего места определяются по СНиП 23-05-95 и СанПиН 2.2.1/2.1.1.1278-03, и отображены в таблице 13.

Таблица 13 – Освещенность рабочего места

|  |  |  |
| --- | --- | --- |
| Характеристика зрительной работы | Освещенность, лк | Освещенность на рабочей поверхности, лк |
| Средней точности | 450 | 150 |

## 5.4 Расчёт искусственного освещения

Необходимо рассчитать искусственное освещение с учетом параметров комнаты: ширина 5,4 м, длина 11,6 м, высота 4,5 м, площадь 62,6 м2.

В качестве системы освещения выбираем световой поток, так как эта система позволяет равномерно распределить свет и яркость в поле зрения.

Определим световой поток по формуле 12:

где *E* – заданная минимальная освещенность, лк;

*S* – освещаемая площадь, м2;

*Z* – отношение средней освещенности к минимальной, выберем *Z* равное 1,1;

*K* – коэффициент запаса, выберем значение *K* равное 1,5;

*U* – коэффициент, характеризующий эффективность использования светового потока источников света.

Для определения коэффициента *U* находится индекс помещения – *I* и предположительно оцениваются коэффициенты отражения стен помещения: – 30 %; потолка: – 50 %. Индекс помещения находится по формуле 13.

где *A* – длина помещения, м;

*B* – ширина помещения, м;

*h* – расчетная высота подвеса светильников, м;

 – коэффициент затенения (для помещения с фиксированным положением работающих принимается равным 0,8).

Высоту подвеса светильников находим по формуле 14:

где  – высота рабочей поверхности над полом (принимаем равной 1 м).

Расстояние между рядами светильников L принимается, равным:

Количество рядов светильников *n* принимаем, равное целому числу от

Подставив численные значения в формулу 13, получаем:

Коэффициент использования светового потока при индексе помещения 1,5, 50, 30 составляет 0,49.

Далее подставим все численные значения в формулу 12

Примем для установки люминесцентные лампы ЛБ-80, ее параметры приведены в таблице 14.

Таблица 14 – Параметры люминесцентной лампы ЛБ-80

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Мощность W, Вт | Сила тока I, А | Напряжение U, В | Размеры, мм | | Срок службы | | Световой поток *Фл*, лм |
| Длина со шты-рьками цоколей | Диаметр | Минимальный | Средний |
|  |  |  |  |  |  |  |  |
| 80 | 0,82 | 108 | 1515 | 38 | 4800 | 12000 | 4300 |

Определим необходимое количество светильников для данного помещения по формуле 15:

Как было описано выше, светильники будут расположены в 2 ряда, таким образом, чтобы в одном ряду было 11 светильников. Длина комнаты составляет 11,6м и расположить 11 ламп по 1,515м в ряд не получится (т.к. м). Поэтому выберем светильники с двумя люминесцентными лампами. В результате в помещении необходимо разместить 2 ряда светильников по 6 в каждом ряду, каждый светильник будет укомплектован 2 лампами. Итого: 24 лампы типа ЛБ-80. На рисунке ниже представлена схема размещения светильников.



Рисунок 34 – Схема размещения светильников

Результаты проведенных расчетов представлены в таблице 15

Таблица 15 – Результаты расчетов искусственного освещения

|  |  |
| --- | --- |
| Параметр | Значение |
|  |  |
| Размеры помещения:  А, м  В, м  Н, м | 11,6  5,4  4,5 |
| Разряд зрительных работ | IV, б |
| Минимальная допустимая освещенность на рабочем месте для данного типа зрительных работ *Ен* , лк | 450 |
| Коэффициенты отражения:  потолка , %  стен , % | 50  30 |
| Высота подвеса светильников *h*, м | 3,5 |
| Расстояние между рядами светильников *L*, м | 3,8 |
| Расстояние между светильниками в ряду *С*, м | 0,35 |
| Расстояние между стеной и рядом светильников *R*, м | 0,65 |
| Индекс помещения *I* | 1,5 |
| Коэффициент использования светового потока , % | 49 |
| Количество ламп в светильнике | 2 |
| Количество светильников *N* | 12 |
| Световой поток от одного светильника, лм | 4300 |
| Коэффициент запаса, | 1,5 |
| Коэффициент неравномерности *Z* | 1,1 |
| Коэффициент затенения | 0,8 |
| Освещенность на рабочем месте, лк | 1511,4 |

Общая мощность осветительной системы вычисляется по формуле 16:

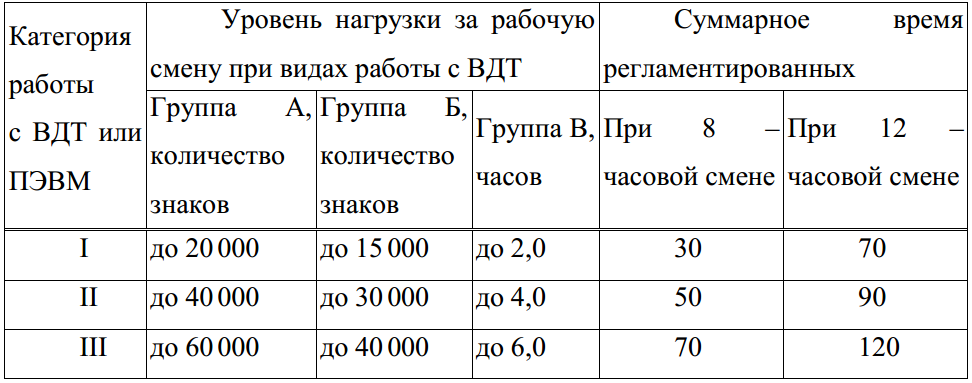
По результату расчетов освещенность, приходящая на рабочее место, составила 1511,4 лк, что соответствует требованиям СНиП 23-05-95.

## 5.5 Мероприятия по защите при организации работы на ПЭВМ

Как уже было неоднократно отмечено, при работе с персональным компьютером очень важную роль играет соблюдение правильного режима труда и отдыха. В противном случае у персонала отмечаются значительное напряжение зрительного аппарата с появлением жалоб на неудовлетворенность работой, головные боли, раздражительность, нарушение сна, усталость и болезненные ощущения в глазах, в пояснице, в области шеи и руках.

В таблице 16 представлены сведения о регламентированных перерывах, которые необходимо делать при работе на компьютере, в зависимости от продолжительности рабочей смены, видов и категорий трудовой деятельности с ВДТ (видеодисплейный терминал) и ПЭВМ (в соответствии с СанПиН 2.2.2 542-46 «Гигиенические требования к видеодисплейным терминалам, персональным электронно-вычислительным машинам и организации работ»).

Таблица 16– Время регламентированных перерывов при работе на компьютере



Воздух желательно очищать от пыли, так как пыль, оседающая на устройства и узлы ПЭВМ, ухудшает теплоотдачу, может образовывать токопроводящие цепи, вызывает стирание подвижных частей и нарушение контактов. Системные блоки компьютеров, мониторы, блоки бесперебойного питания, осветительных приборов, компьютерной не должна превышать 35 Вт/м2 при облучении 50 % поверхности человека и более, 70 Вт/м2 – при облучении 25...50 % поверхности и 100 Вт/м2 – при облучении не более 25% поверхности тела.

Согласно ГОСТ 12.2.032-78 рабочее место и взаимное расположение всех его элементов должно соответствовать антропометрическим, физическим и психологическим требованиям. Большое значение имеет также характер работы.

Главными элементами рабочего места инженера–программиста являются стол и кресло. Основным рабочим положением является положение, сидя, так как вызывает минимальное утомление инженера–программиста. Рабочее место для выполнения работ в положении сидя организуется в соответствии с ГОСТ 12.2.032-78.

На рисунке 46 показано размещение основных и периферийных составляющих ПК на рабочем столе инженера–программиста.

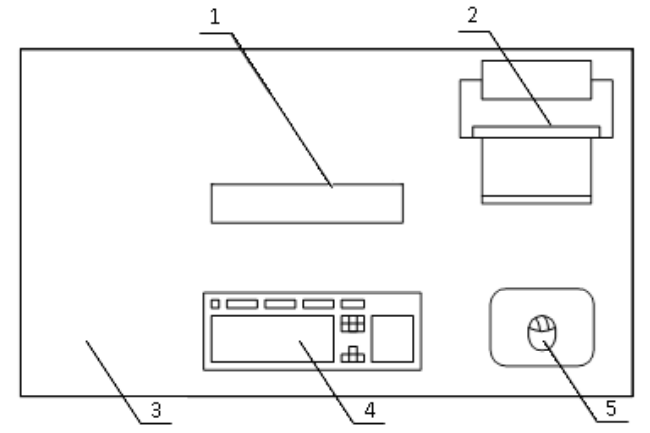


Рисунок 35 – Размещение основных и периферийных

составляющих ПК

1 – монитор, 2 – принтер, 3 – поверхность рабочего стола,

4 – клавиатура, 5 – манипулятор типа «мышь»

Экран располагается на расстоянии считывания 0,6м с углом считывания, направлением взгляда на 20о  ниже горизонтали к центру экрана, причем экран перпендикулярен этому направлению.

# ЗАКЛЮЧЕНИЕ

В процессе выполнения дипломного проекта было разработано программное обеспечение для командно-телеметрической радиолинии. Все условия, поставленные в техническом задании, были выполнены. Разработка программного обеспечения велась в редакторе кода CLion. В процессе написания кода использовались возможности языков C, C++, Python, операционная система FreeRTOS, использовался генератор кода STM Cube, фреймворк Qt для написания графического интерфейса.

Проделана работа по изучению современной технологии модуляции LoRa, которая позволяет передавать данные на большие расстояния с меньшим потреблением энергии. Было проведено ознакомление со свёрточным методом кодирования сигнала, используемым в радиомодуле SX1276.

Разработаны структурные и функциональные схемы модема. Произведена работа по запуску и отладке используемого радиомодема. Произведена работа с документацией на используемые в модеме микросхемы.

Был проведен анализ радиоканала на каждой из частот модема. А также произведены тесты дальности передаваемого сигнала. Каждый из модемов был разнесен на расстояние более 1.5 км. Эквивалентное расстояние достигалось за счет создания аттенюаторами дополнительного затухания в радиоканале. Результаты эксперимента приведены в разделе 4.5

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Боев, Н.М. Анализ командно-телеметрической радиолинии связи с беспилотными летательными аппаратами / Н.М. Боев // Вестник Сибирского государственного аэрокосмического университета имени академика М. Ф. Решетнева. – 2012. – № 2. – С. 86 – 91

2. Обзор технологии LoRa // Технологии связи [электронный ресурс] – URL: https://itechinfo.ru/content/%D0%BE%D0%B1%D0%B7%D0%BE%D1%80-%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D0%B8 (дата обращения: 10.11.2022)

3. Г.И. Никитин. Сверточные коды. Учебное пособие. Санкт–Петербург. Министерство образования Российской Федерации : Изд-во Санкт–Петербургский государственный университет аэрокосмического приборостроения: 2001. – 80 c (дата обращения: 10.11.2022)

4. SX1276/77/78/79 // MOUSER [электронный ресурс] – URL: https://www.mouser.com/datasheet/2/761/sx1276-1278113.pdf (дата обращения: 10.11.2022)

5. STM32F7 // MICROCONTROLLERS-MICROPROCESSORS [электронный ресурс] – URL: https://www.st.com/en/microcontrollers-microprocessors/stm32f7-series.html (дата обращения: 10.11.2022)

6. LM5005 // ChipDip [электронный ресурс] – URL: https://static.chipdip.ru/lib/766/DOC004766644.pdf (дата обращения: 10.11.2022)

7. Ethernet // MICROCHIP [электронный ресурс] – URL: https://ww1.microchip.com/downloads/en/DeviceDoc/00002245B.pdf (дата обращения: 10.11.2022)

8. CP2102 // ChipDip [электронный ресурс] – URL: https://static.chipdip.ru/lib/444/DOC012444768.pdf (дата обращения: 10.11.2022)

9. MAX3051 // КОМПЭЛ [электронный ресурс] – URL: https://www.compel.ru/item pdf/a851c7d381713f084f912413861f4740/pn/max~max3051eka-t.pdf (дата обращения: 10.11.2022)

10 T. Elshabrawy and J. Robert. Closed-form approximation of lora modulation ber performance. IEEE Communications Letters, 22(9):1778–1781, 2018

11 FreeRTOS // Wikipedia [электронный ресурс] – URL: https://ru.wikipedia.org/wiki/FreeRTOS (дата обращения: 10.11.2022)

12. Дашкова, А.К. Безопасность жизнедеятельности: учебно-методическое пособие для дипломного проектирования [Электронный ресурс] / А. К. Дашкова. – Электрон. дан. – Сибирский федеральный университет, 2013. (дата обращения: 10.11.2022)

13. Система стандартов безопасности труда «Опасные и вредные производственные факторы. Классификация»: Межгосударственный стандарт/ Постановлением Государственного комитета стандартов Совета Министров СССР от 18 Ноября 1974 г.

14. Вредные и опасные факторы при работе на персональных электронно-вычислительных машинах [Электронный ресурс] – URL:  http://www.grandars.ru/shkola/bezopasnost-zhiznedeyat. vrednye-faktory-pri-rabote-na-pk.html (дата обращения: 10.11.2022)

15. MedicInform.Net медицина, психология, 1999-2014. Статья «Как влияет монитор ПК на зрение» [Электронный ресурс] – URL: http://www.medicinform.net/comp/comp\_vis3.html (дата обращения: 10.11.2022)

# ПРИЛОЖЕНИЕ А

Структурная схема устройства

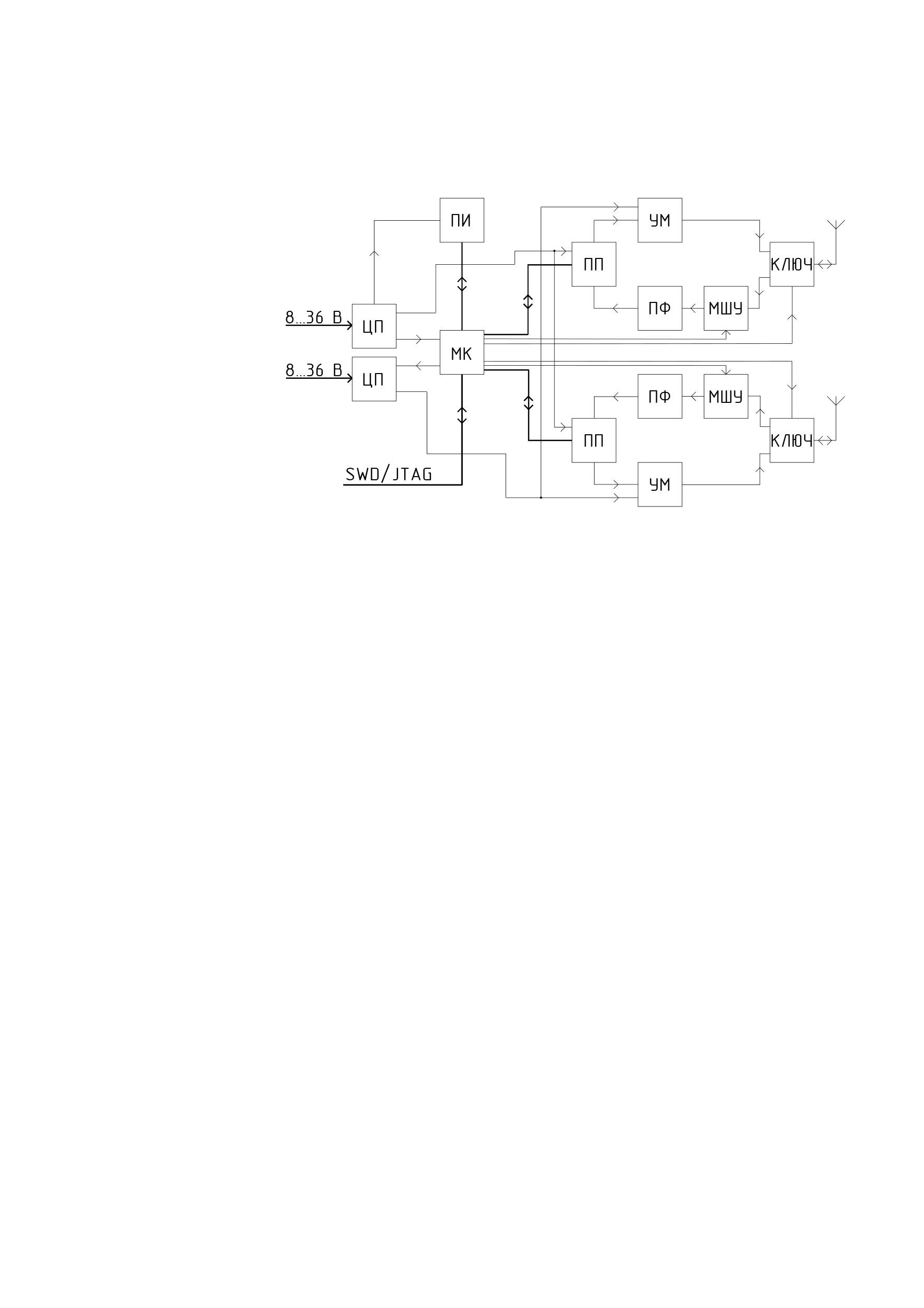


Рисунок 36 – Структурная схема устройства

# ПРИЛОЖЕНИЕ Б

Функциональная схема устройства

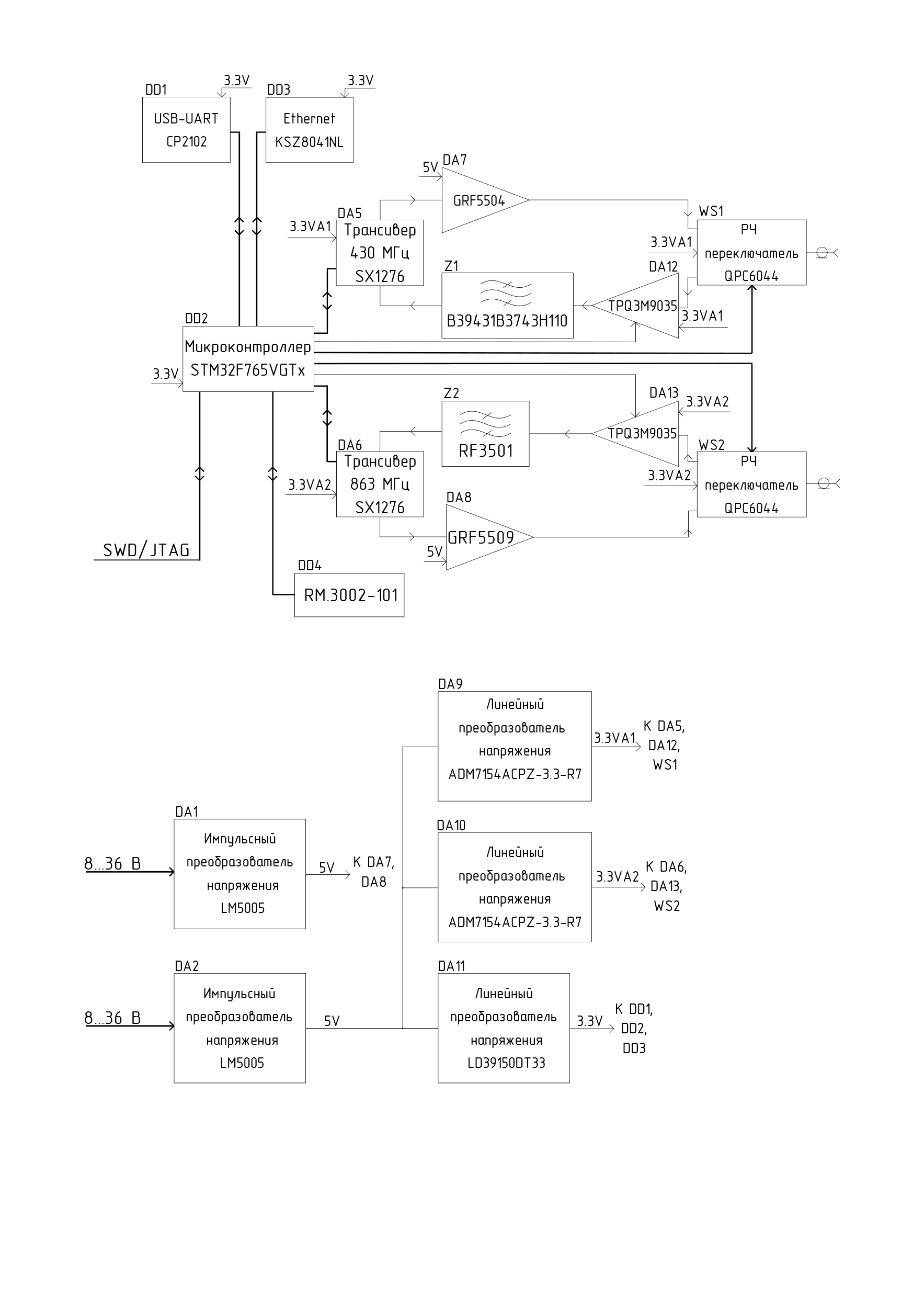


Рисунок 37 – Функциональная схема устройства

# ПРИЛОЖЕНИЕ В

# ПРИЛОЖЕНИЕ Г

Перечень элементов

# ПРИЛОЖЕНИЕ Д

Блок-схема алгоритма определения ведущего устройства по радиоканалу

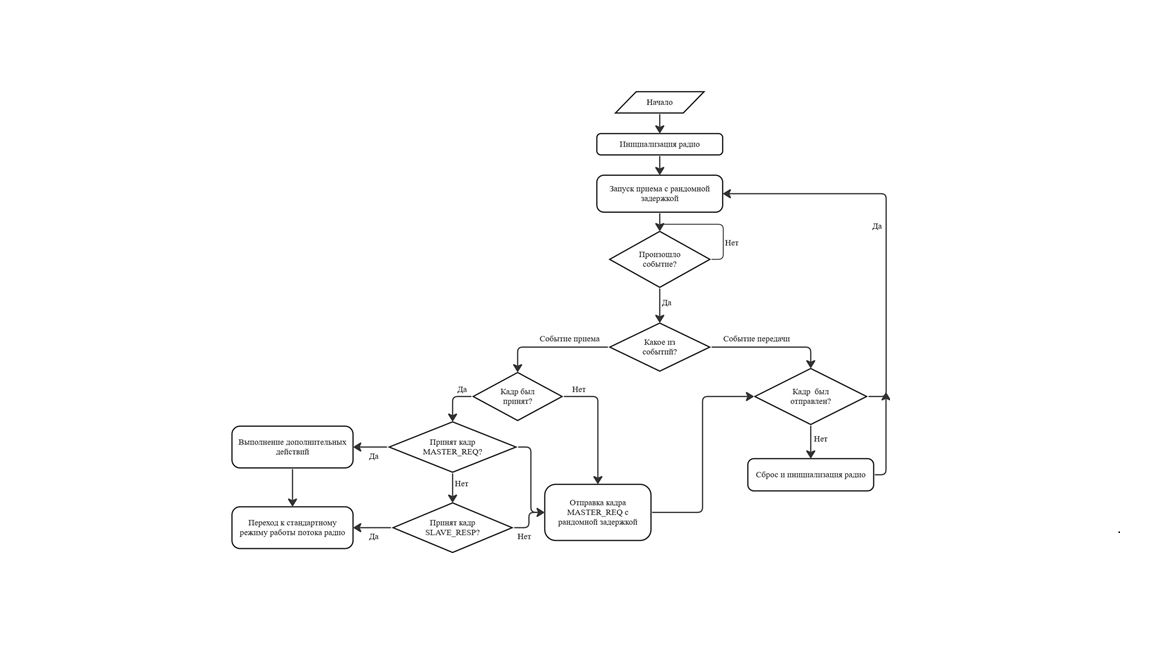


Рисунок 38 – Функциональная схема устройства

# ПРИЛОЖЕНИЕ Е

Блок-схема алгоритма тестирования частоты потерь информационных пакетов

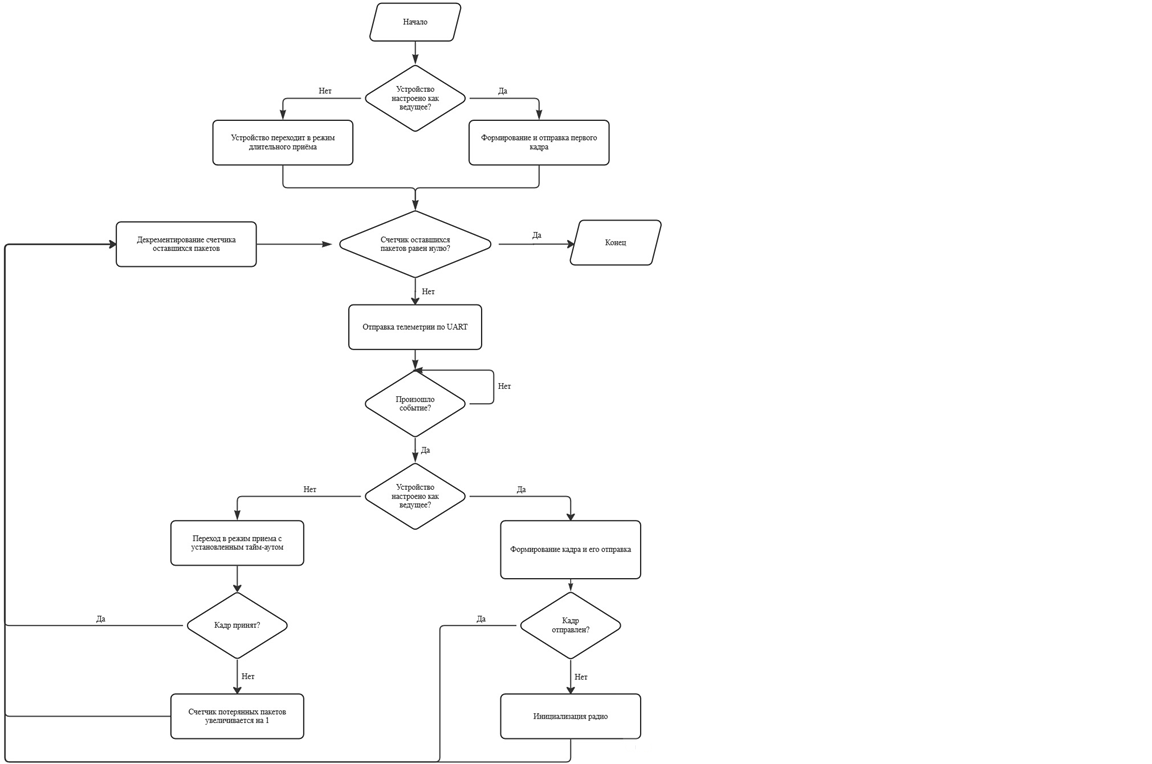


Рисунок 39 – Функциональная схема устройства

# ПРИЛОЖЕНИЕ Ж

Листинг алгоритма определения ведущего устройства по радиоканалу

#define RECV\_TIMEOUT\_MS 500

void KtrRadioStateSynchro::InitOS()

{

HAL\_RNG\_GenerateRandomNumber(&hrng, &rnd\_);

rnd\_ = rnd\_%RECV\_TIMEOUT\_MS;

}

void KtrRadioStateSynchro::Prepare()

{

CalculateTimeOnAirs();

osDelay(rnd\_);

radio\_.Recv(RECV\_TIMEOUT\_MS);

}

void KtrRadioStateSynchro::Run()

{

EventBits\_t event = radio\_.WaitRadioEvent(portMAX\_DELAY);

if (event != 0) {

switch (event) {

case RX\_DONE:

RxDoneHandler();

break;

case RX\_TIMEOUT:

RxTimeoutHandler();

break;

case RX\_ERROR:

RxErrorHandler();

break;

case TX\_DONE:

TxDoneHandler();

break;

case TX\_TIMEOUT:

TxTimeoutHandler();

break;

default:

break;

}

}

}

void KtrRadioStateSynchro::RxDoneHandler()

{

bool blink = false;

if (IsSyncResponseMessage(radio\_.GetPayloadBuff(),radio\_.GetPayloadSize())) {

blink = true;

MakeActionIfReceiveResponse();

} else if (IsSyncRequestMessage(radio\_.GetPayloadBuff(), radio\_.GetPayloadSize())) {

blink = true;

MakeActionIfReceiveRequest();

} else {

iAmMaster\_ = true;

SendRequest();

}

if (blink) {

timeslotEvent = xEventGroupCreate();

iAmMaster = iAmMaster\_;

if (iAmMaster)

HAL\_GPIO\_WritePin(LED0\_GPIO\_Port, LED0\_Pin, GPIO\_PIN\_SET);

SwitchState();

}

}

void KtrRadioStateSynchro::SendRequest()

{

HAL\_RNG\_GenerateRandomNumber(&hrng, &rnd\_);

rnd\_ = rnd\_%100;

osDelay(rnd\_);

while (\_\_HAL\_TIM\_GET\_COUNTER(&htim\_) > SYNC\_TIMS\_PERIOD\_TICKS - timeOnAirReqTicks\_ - SYNC\_TIMS\_PERIOD\_TICKS/4)

osDelay(1);

uint8\_t \*pBuff = GetSyncReqMessage();

radio\_.Send(pBuff, SYNC\_MASTER\_REQ\_SIZE);

}

void KtrRadioStateSynchro::TxDoneHandler()

{

radio\_.Recv(RECV\_TIMEOUT\_MS);

}

void KtrRadioStateSynchro::TxTimeoutHandler()

{

radio\_.GetSettings(&settings\_);

radio\_.SetSettings(&settings\_);

radio\_.Recv(RECV\_TIMEOUT\_MS);

}

void KtrRadioStateSynchro::RxErrorHandler()

{

RxTimeoutHandler();

}

void KtrRadioStateSynchro::RxTimeoutHandler()

{

if (iAmMaster\_)

SendRequest();

else

radio\_.Recv(RECV\_TIMEOUT\_MS);

}

void KtrRadioStateSynchro::MakeActionIfReceiveRequest()

{

uint32\_t cnt = CalculateTimerDuration();

HAL\_TIM\_Base\_Stop\_IT(&htim\_);

\_\_HAL\_TIM\_SET\_COUNTER(&htim\_, cnt);

uint32\_t timCnt\_ = GetTimCntFromSyncMessage(radio\_.GetPayloadBuff()) - 1;

syncTimerPeriodCounter = timCnt\_;

HAL\_TIM\_Base\_Start\_IT(&htim\_);

NotifyTasks(timCnt\_);

iAmMaster\_ = false;

uint8\_t \*pBuff = GetSyncRespMessage();

radio\_.Send(pBuff, SYNC\_SLAVE\_RESP\_SIZE);

radio\_.WaitRadioEvent(portMAX\_DELAY);

}

void KtrRadioStateSynchro::MakeActionIfReceiveResponse()

{

iAmMaster\_ = true;

}

# ПРИЛОЖЕНИЕ З

Листинг функций для тестирования модема

typedef struct CwTimerContext\_s {

TRX\_t \*trx;

ktr\_package\_cw\_params cw\_params;

}CwTimerContext\_t;

static void RadioTestStreamTx(TRX\_t \*trx, KtrStateArgs\_t \*pArgs, SxStates\_t radioState);

static void RadioTestStreamRx(TRX\_t \*trx, KtrStateArgs\_t \*pArgs, SxStates\_t radioState);

static void RadioTestStreamTxDoneHandler(TRX\_t \*trx, KtrStateArgs\_t \*pArgs);

static void RadioTestStreamTxTimeoutHandler(TRX\_t \*trx, KtrStateArgs\_t \*pArgs);

static void RadioTestStreamRxDoneHandler(TRX\_t \*trx, KtrStateArgs\_t \*pArgs);

static void RadioTestStreamRxErrorTimeoutHandler(TRX\_t \*trx, KtrStateArgs\_t \*pArgs);

static void SendTestStreamStatistic(TRX\_t \*trx, KtrStateArgs\_t \*pArgs);

static void SendStreamTestMessageForUser(TRX\_t \*trx, KtrStateArgs\_t \*pArgs, DecodedRadioData\_t \*pRadioData);

static void CwTimerHandler(TimerHandle\_t xTimer);

inline static uint8\_t \* FillTestBuffWithRandomNumber(TRX\_t \*trx, size\_t len);

static void TestPerTxDoneHandler(TRX\_t \*trx, KtrStateArgs\_t \*pArgs);

static void TestPerTxDoneHandlerForSSTest(TRX\_t \*trx, ktr\_package\_per\_params \*pPerParams);

static void TestPerTxDoneHandlerForDSTest(TRX\_t \*trx, ktr\_package\_per\_params \*pPerParams);

static void TestPerRxEventHandler(TRX\_t \*trx, KtrStateArgs\_t \*pArgs);

static void UpdatePer(TRX\_t \*trx, KtrStateArgs\_t \*pArgs, DecodedRadioData\_t \*pRadioData, bool isRxDone);

static void UpdatePerForSSTest(TRX\_t \*trx,

ktr\_package\_per\_params \*pPerParams,

DecodedRadioData\_t \*pRadioData,

bool isRxDone);

static void UpdatePerForDSTest(TRX\_t \*trx,

ktr\_package\_per\_params \*pPerParams,

DecodedRadioData\_t \*pRadioData,

bool isRxDone);

static void MakeActionAfterReceiveEvent(TRX\_t \*trx,

KtrStateArgs\_t \*pArgs,

DecodedRadioData\_t \*pRadioData,

bool isRxDone);

static void MakeActionAfterReceiveEventForSSTest(TRX\_t \*trx, ktr\_package\_per\_params \*pPerParams);

static void MakeActionAfterReceiveEventForDSTest(TRX\_t \*trx,

ktr\_package\_per\_params \*pPerParams,

DecodedRadioData\_t \*pRadioData,

bool isRxDone);

static void SendPerStatistic(TRX\_t \*trx, KtrStateArgs\_t \*pArgs, DecodedRadioData\_t \*pRadioData, bool isRxDone);

inline static void SendMasterPerFrame(TRX\_t \*trx, ktr\_package\_per\_params \*pPerParams);

static void SendTestResult(TRX\_t \*trx, KtrStateArgs\_t \*pArgs, ktr\_package\_test\_results \*pTestResults);

static void SendPerTestMessageForUser(TRX\_t \*trx, KtrStateArgs\_t \*pArgs, DecodedRadioData\_t \*pRadioData, bool isRxDone);

inline static uint8\_t GetTestBuffShift(TRX\_t \*trx);

void RadioTestStream(TRX\_t \*trx, KtrStateArgs\_t \*pArgs)

{

if ((trx == NULL) || (pArgs == NULL)) Error\_Handler();

SxStates\_t radio\_state = xEventGroupWaitBits(trx->event,

RF\_STATE\_MASK, pdTRUE, pdFALSE, 100);

CheckAndSendTelemetryInfo(trx, pArgs, radio\_state);

if (pArgs->data.data\_list.test.test\_params.stream\_params.is\_tx == true)

RadioTestStreamTx(trx, pArgs, radio\_state);

else

RadioTestStreamRx(trx, pArgs, radio\_state);

}

void RadioTestStreamTx(TRX\_t \*trx, KtrStateArgs\_t \*pArgs, SxStates\_t radioState)

{

if (trx == NULL) Error\_Handler();

switch (radioState) {

case RX\_DONE:

case RX\_TIMEOUT:

case RX\_ERROR:

case TX\_DONE:RadioTestStreamTxDoneHandler(trx, pArgs);

break;

case TX\_TIMEOUT:RadioTestStreamTxTimeoutHandler(trx, pArgs);

break;

case TX\_RUNNING:

case RX\_RUNNING:

case CAD:

case LOWPOWER:

default:

break;

}

osDelay(pArgs->data.data\_list.test.test\_params.stream\_params.delay\_between\_transfer);

}

void RadioTestStreamRx(TRX\_t \*trx, KtrStateArgs\_t \*pArgs, SxStates\_t radioState)

{

if (trx == NULL) Error\_Handler();

switch (radioState) {

case RX\_DONE:RadioTestStreamRxDoneHandler(trx, pArgs);

break;

case TX\_DONE:

case TX\_TIMEOUT:

case RX\_TIMEOUT:

case RX\_ERROR:RadioTestStreamRxErrorTimeoutHandler(trx, pArgs);

break;

case RX\_RUNNING:

case TX\_RUNNING:

case CAD:

case LOWPOWER:

default:

break;

}

}

static void RadioTestStreamTxDoneHandler(TRX\_t \*trx, KtrStateArgs\_t \*pArgs)

{

ktr\_package\_stream\_params streamParams = pArgs->data.data\_list.test.test\_params.stream\_params;

uint8\_t \*pBuff = FillTestBuffWithRandomNumber(trx, streamParams.pack\_len);

SendFrame(trx, pBuff, streamParams.pack\_len);

}

static void RadioTestStreamTxTimeoutHandler(TRX\_t \*trx, KtrStateArgs\_t \*pArgs)

{

uint32\_t freq = GetTrxFrequency(trx);

KtrDeInit(trx);

/\* Get settings from sx1276 struct because modem already initialize \*/

KtrInit(trx, trx->ktrModem, freq);

ktr\_package\_stream\_params streamParams = pArgs->data.data\_list.test.test\_params.stream\_params;

uint8\_t \*pBuff = FillTestBuffWithRandomNumber(trx, streamParams.pack\_len);

SendFrame(trx, pBuff, streamParams.pack\_len);

}

static void RadioTestStreamRxDoneHandler(TRX\_t \*trx, KtrStateArgs\_t \*pArgs)

{

BlinkRxDoneLed(trx);

SendTestStreamStatistic(trx, pArgs);

uint32\_t timeout = GetTrxTimeout(trx);

RecvFrame(trx, timeout);

}

static void RadioTestStreamRxErrorTimeoutHandler(TRX\_t \*trx, KtrStateArgs\_t \*pArgs)

{

SendTestStreamStatistic(trx, pArgs);

uint32\_t timeout = GetTrxTimeout(trx);

RecvFrame(trx, timeout);

}

static void SendTestStreamStatistic(TRX\_t \*trx, KtrStateArgs\_t \*pArgs)

{

DecodedRadioData\_t radioData = {0};

xQueueReceive(trx->rxQueue, &radioData, 10);

ktr\_package\_test\_results test\_results = {

.channel = trx->channel,

.modem = (ktr\_package\_modem) trx->ktrModem,

.snr = radioData.snr,

.rssi = radioData.rssi,

.received\_message\_id = radioData.message\_id

};

SendTestResult(trx, pArgs, &test\_results);

SendStreamTestMessageForUser(trx, pArgs, &radioData);

}

static void SendStreamTestMessageForUser(TRX\_t \*trx, KtrStateArgs\_t \*pArgs, DecodedRadioData\_t \*pRadioData)

{

pArgs->outputInfo.which\_info = ktr\_package\_output\_info\_text\_message\_for\_user\_tag;

pArgs->outputInfo.channel = trx->channel;

pArgs->outputInfo.cur\_cmd = pArgs->cmd;

if (pRadioData->rssi != 0) {

sprintf(pArgs->outputInfo.info.text\_message\_for\_user.text, "Message received\n"

"Received message ID: %lu\n"

"SNR: %ld\n"

"RSSI: %ld",

pRadioData->message\_id,

pRadioData->snr,

pRadioData->rssi);

} else {

sprintf(pArgs->outputInfo.info.text\_message\_for\_user.text, "Message lost!!!");

}

xQueueSend(trx->outputInfoQueue, &pArgs->outputInfo, 50);

}

static void RadioTestCwTxTimeoutHandler(TRX\_t \*trx, ktr\_package\_cw\_params \*pData, uint32\_t timeout)

{

static TimerEvent\_t cwTimer;

static CwTimerContext\_t cwTimerContext = {0};

if (cwTimer.tim == NULL) TimerInit("CW Timer", &cwTimer, CwTimerHandler);

TimerStop(&cwTimer);

/\* On case if params will be changed \*/

memcpy(&cwTimerContext.cw\_params, pData, sizeof(ktr\_package\_cw\_params));

cwTimerContext.trx = trx;

TimerSetContext(&cwTimer, &cwTimerContext);

TimerSetValue(&cwTimer, timeout);

TimerReset(&cwTimer);

}

void RadioTestCw(TRX\_t \*trx, KtrStateArgs\_t \*pArgs)

{

if ((trx == NULL) || (pArgs == NULL)) Error\_Handler();

ktr\_package\_cw\_params \*cwParams = &pArgs->data.data\_list.test.test\_params.cw\_params;

SxStates\_t radio\_state = xEventGroupWaitBits(trx->event,

RF\_STATE\_MASK, pdTRUE, pdFALSE, 100);

CheckAndSendTelemetryInfo(trx, pArgs, radio\_state);

switch (radio\_state) {

case TX\_DONE:

case RX\_DONE:

case RX\_TIMEOUT:

case RX\_ERROR:

case RX\_RUNNING:

case TX\_TIMEOUT:RadioTestCwTxTimeoutHandler(trx, cwParams,

(cwParams->sleep\_time) \* 1000);

case TX\_RUNNING:

case CAD:

case LOWPOWER:

default:

break;

}

}

static void CwTimerHandler(TimerHandle\_t xTimer)

{

CwTimerContext\_t \*ctx = (CwTimerContext\_t \*) TimerGetContext(xTimer);

if (ctx == NULL) Error\_Handler();

uint32\_t freq = GetTrxFrequency(ctx->trx);

Radio.SetTxContinuousWave(ctx->trx, freq, (int8\_t)ctx->cw\_params.pwr, ctx->cw\_params.duration);

}

void RadioTestPer(TRX\_t \*trx, KtrStateArgs\_t \*pArgs)

{

if ((trx == NULL) || (pArgs == NULL)) Error\_Handler();

SxStates\_t radioState = xEventGroupWaitBits(trx->event,

RF\_STATE\_MASK, pdTRUE, pdFALSE, 100);

CheckAndSendTelemetryInfo(trx, pArgs, radioState);

if (pArgs->data.data\_list.test.test\_params.per\_params.per\_packs == 0)

return;

switch (radioState) {

case LOWPOWER:break;

case RX\_DONE:

case RX\_ERROR:

case RX\_TIMEOUT:TestPerRxEventHandler(trx, pArgs);

break;

case TX\_DONE:TestPerTxDoneHandler(trx, pArgs);

break;

case TX\_TIMEOUT:

case IDLE:

case RX\_RUNNING:

case TX\_RUNNING:

case CAD:

case SET\_CONFIG:

case GET\_CONFIG:break;

}

}

static void TestPerRxEventHandler(TRX\_t \*trx, KtrStateArgs\_t \*pArgs)

{

DecodedRadioData\_t radioData = {0};

bool isRxDone = (xQueueReceive(trx->rxQueue, &radioData, 10) == pdTRUE);

UpdatePer(trx, pArgs, &radioData, isRxDone);

MakeActionAfterReceiveEvent(trx, pArgs, &radioData, isRxDone);

SendPerStatistic(trx, pArgs, &radioData, isRxDone);

}

static void UpdatePer(TRX\_t \*trx, KtrStateArgs\_t \*pArgs, DecodedRadioData\_t \*pRadioData, bool isRxDone)

{

ktr\_package\_per\_params \*pPerParams = &pArgs->data.data\_list.test.test\_params.per\_params;

if (pArgs->cmd == ktr\_package\_cmd\_TEST\_PER\_SINGLE\_SIDE)

UpdatePerForSSTest(trx, pPerParams, pRadioData, isRxDone);

else if (pArgs->cmd == ktr\_package\_cmd\_TEST\_PER\_DOUBLE\_SIDE)

UpdatePerForDSTest(trx, pPerParams, pRadioData, isRxDone);

}

static void UpdatePerForSSTest(TRX\_t \*trx,

ktr\_package\_per\_params \*pPerParams,

DecodedRadioData\_t \*pRadioData,

bool isRxDone)

{

if (isRxDone == true)

trx->perCnt--;

}

static void UpdatePerForDSTest(TRX\_t \*trx,

ktr\_package\_per\_params \*pPerParams,

DecodedRadioData\_t \*pRadioData,

bool isRxDone)

{

if (isRxDone == true) {

if (pPerParams->is\_master == true) {

uint8\_t \*pBuff = FillTestBuffWithRandomNumber(trx, 0);

if (memcmp(pBuff, pRadioData->payload, pPerParams->per\_pack\_len) == 0)

trx->perCnt--;

} else {

trx->perCnt--;

}

}

}

static void MakeActionAfterReceiveEvent(TRX\_t \*trx,

KtrStateArgs\_t \*pArgs,

DecodedRadioData\_t \*pRadioData,

bool isRxDone)

{

ktr\_package\_per\_params \*pPerParams = &pArgs->data.data\_list.test.test\_params.per\_params;

if (pArgs->cmd == ktr\_package\_cmd\_TEST\_PER\_SINGLE\_SIDE)

MakeActionAfterReceiveEventForSSTest(trx, pPerParams);

else if (pArgs->cmd == ktr\_package\_cmd\_TEST\_PER\_DOUBLE\_SIDE)

MakeActionAfterReceiveEventForDSTest(trx, pPerParams, pRadioData, isRxDone);

}

/\* Only for slave device \*/

static void MakeActionAfterReceiveEventForSSTest(TRX\_t \*trx, ktr\_package\_per\_params \*pPerParams)

{

uint32\_t timeout;

/\* Continuous receive until first frame doesn't receive \*/

if (trx->perCnt == pPerParams->per\_packs)

timeout = 0;

else

timeout = GetTrxTimeout(trx);

uint32\_t currentRxEventCounters = GetCurrentReceiveEventCountersForPer(trx);

uint32\_t startRxEventCounters = GetStartReceiveEventCountersForPer(trx);

if (startRxEventCounters + pPerParams->per\_packs > currentRxEventCounters)

RecvFrame(trx, timeout);

else

pPerParams->per\_packs = 0; /\* For exit \*/

}

static void MakeActionAfterReceiveEventForDSTest(TRX\_t \*trx,

ktr\_package\_per\_params \*pPerParams,

DecodedRadioData\_t \*pRadioData,

bool isRxDone)

{

if (pPerParams->is\_master == true) {

if (isRxDone == true)

BlinkRxDoneLed(trx);

osDelay(pPerParams->delay\_between\_transfer);

SendMasterPerFrame(trx, pPerParams);

} else if (pPerParams->is\_master == false) {

if (isRxDone == true) {

BlinkRxDoneLed(trx);

osDelay(pPerParams->delay\_between\_transfer);

SendFrame(trx, pRadioData->payload, pRadioData->payload\_len);

} else {

uint32\_t timeout;

if (trx->perCnt == pPerParams->per\_packs)

timeout = 0;

else

timeout = GetTrxTimeout(trx);

uint32\_t currentRxEventCounters = GetCurrentReceiveEventCountersForPer(trx);

uint32\_t startRxEventCounters = GetStartReceiveEventCountersForPer(trx);

if (startRxEventCounters + pPerParams->per\_packs > currentRxEventCounters)

RecvFrame(trx, 2\*timeout);

else

pPerParams->per\_packs = 0;

}

}

}

inline static void SendMasterPerFrame(TRX\_t \*trx, ktr\_package\_per\_params \*pPerParams)

{

uint32\_t currentTxEventCounters = GetCurrentTransmitEventCountersForSSPer(trx);

uint32\_t startTxEventCounters = GetStartTransmitEventCountersForSSPer(trx);

if (startTxEventCounters + pPerParams->per\_packs > currentTxEventCounters) {

uint8\_t \*pBuff = FillTestBuffWithRandomNumber(trx, pPerParams->per\_pack\_len);

SendFrame(trx, pBuff, pPerParams->per\_pack\_len);

}

}

static void TestPerTxDoneHandler(TRX\_t \*trx, KtrStateArgs\_t \*pArgs)

{

ktr\_package\_per\_params \*pPerParams = &pArgs->data.data\_list.test.test\_params.per\_params;

if (pArgs->cmd == ktr\_package\_cmd\_TEST\_PER\_SINGLE\_SIDE)

TestPerTxDoneHandlerForSSTest(trx, pPerParams);

else

TestPerTxDoneHandlerForDSTest(trx, pPerParams);

}

/\* Only for master device \*/

static void TestPerTxDoneHandlerForSSTest(TRX\_t \*trx, ktr\_package\_per\_params \*pPerParams)

{

uint8\_t \*pBuff = FillTestBuffWithRandomNumber(trx, pPerParams->per\_pack\_len);

osDelay(pPerParams->delay\_between\_transfer);

uint32\_t currentTxEventCounters = GetCurrentTransmitEventCountersForSSPer(trx);

uint32\_t startTxEventCounters = GetStartTransmitEventCountersForSSPer(trx);

if (startTxEventCounters + pPerParams->per\_packs > currentTxEventCounters)

SendFrame(trx, pBuff, pPerParams->per\_pack\_len);

else

pPerParams->per\_packs = 0;

trx->perCnt--;

}

static void TestPerTxDoneHandlerForDSTest(TRX\_t \*trx, ktr\_package\_per\_params \*pPerParams)

{

if (pPerParams->is\_master == true) {

uint32\_t currentRxEventCounters = GetCurrentReceiveEventCountersForPer(trx);

uint32\_t startRxEventCounters = GetStartReceiveEventCountersForPer(trx);

if (startRxEventCounters + pPerParams->per\_packs + 1 > currentRxEventCounters) {

uint32\_t timeout;

timeout = GetTrxTimeout(trx);

RecvFrame(trx, timeout);

} else {

pPerParams->per\_packs = 0;

}

} else {

uint32\_t timeout;

if (trx->perCnt == pPerParams->per\_packs)

timeout = 0;

else

timeout = GetTrxTimeout(trx);

RecvFrame(trx, 2\*timeout);

}

}

static void SendPerStatistic(TRX\_t \*trx, KtrStateArgs\_t \*pArgs, DecodedRadioData\_t \*pRadioData, bool isRxDone)

{

ktr\_package\_test\_results test\_results = {

.channel = trx->channel,

.modem = (ktr\_package\_modem) trx->ktrModem,

.snr = pRadioData->snr,

.rssi = pRadioData->rssi,

.received\_message\_id = pRadioData->message\_id,

.this\_device\_message\_id = trx->txDoneCnt

};

SendTestResult(trx, pArgs, &test\_results);

SendPerTestMessageForUser(trx, pArgs, pRadioData, isRxDone);

}

static void SendTestResult(TRX\_t \*trx, KtrStateArgs\_t \*pArgs, ktr\_package\_test\_results \*pTestResults)

{

pArgs->outputInfo.info.test\_results = (\*pTestResults);

pArgs->outputInfo.which\_info = ktr\_package\_output\_info\_test\_results\_tag;

pArgs->outputInfo.channel = trx->channel;

pArgs->outputInfo.cur\_cmd = pArgs->cmd;

xQueueSend(trx->outputInfoQueue, &pArgs->outputInfo, 100);

}

static void SendPerTestMessageForUser(TRX\_t \*trx, KtrStateArgs\_t \*pArgs, DecodedRadioData\_t \*pRadioData, bool isRxDone)

{

pArgs->outputInfo.which\_info = ktr\_package\_output\_info\_text\_message\_for\_user\_tag;

pArgs->outputInfo.channel = trx->channel;

pArgs->outputInfo.cur\_cmd = pArgs->cmd;

if (isRxDone == true) {

sprintf(pArgs->outputInfo.info.text\_message\_for\_user.text, "Current PER in this session: %lu\n"

"Received message ID: %lu\n"

"SNR: %ld\n"

"RSSI: %ld",

trx->perCnt,

pRadioData->message\_id,

pRadioData->snr,

pRadioData->rssi);

} else {

if (trx->perCnt != 0) /\* Because it's begin \*/

sprintf(pArgs->outputInfo.info.text\_message\_for\_user.text, "Message lost!!!\n"

"Current PER in this session: %lu", trx->perCnt);

}

xQueueSend(trx->outputInfoQueue, &pArgs->outputInfo, 50);

}

inline static uint8\_t \* FillTestBuffWithRandomNumber(TRX\_t \*trx, size\_t len)

{

static uint8\_t testBuff[MAX\_MESSAGE\_SIZE\_IN\_BYTES\*NUMBER\_OF\_TRX];

int i = 0;

uint8\_t shift = GetTestBuffShift(trx);

uint8\_t \*buff = &testBuff[shift\*MAX\_MESSAGE\_SIZE\_IN\_BYTES];

while (i < len)

buff[i++] = randr(0, 0x7FFFFFFF);

return buff;

}

inline static uint8\_t GetTestBuffShift(TRX\_t \*trx)

{

uint8\_t shift = 0;

if (trx->channel == ktr\_package\_channel\_HF)

shift = 1;

return shift;

}