Разработка протокола беспроводной самоорганизующейся сенсорной сети.

**Содержание**

[1. Особенности протокола. 3](#_Toc105534354)

[2. Состав и топология сети. 5](#_Toc105534355)

[3. Аппаратная реализация узлов сети Протокола. 6](#_Toc105534356)

[4. Структура пакетов протокола 13](#_Toc105534357)

[Общая структура заголовка 13](#_Toc105534358)

[Описание полей заголовка 13](#_Toc105534359)

[5. Типы пакетов 15](#_Toc105534360)

[00 -“Я потенциальный роутер” 15](#_Toc105534361)

[01-“Я - узел” 16](#_Toc105534362)

[02-“Я выбрал роутер” 17](#_Toc105534363)

[03-“Я роутер” 18](#_Toc105534364)

[04-“Опрос устройств” 19](#_Toc105534365)

[05-“Ответ от устройства UNO” 20](#_Toc105534366)

[06-“Ответ от роутера MANY“ 21](#_Toc105534367)

[6. Описание работы протокола 22](#_Toc105534368)

[Этап построения сети 22](#_Toc105534369)

[Передача данных. 30](#_Toc105534370)

[Механизм управления доступом к эфиру. 31](#_Toc105534371)

[7. Программная реализация узлов сети Протокола. 36](#_Toc105534372)

[Описание модели информационной системы, использующей Протокол. 36](#_Toc105534373)

[Используемый язык программирования 38](#_Toc105534374)

[Общая структура и функции программного модуля Протокола 38](#_Toc105534375)

[Роль и место программного модуля Протокола в составе общей программы узла. 39](#_Toc105534376)

[Архитектура программного модуля Протокола. 41](#_Toc105534377)

# **Особенности протокола.**

Протокол беспроводной самоорганизующейся сенсорной сети (далее Протокол) предназначен для организации обмена небольших порций данных для сбора информации с датчиков, входящих в состав узлов сети, а также для отправки команд управления для различных исполнительных механизмов, управляемых узлами сети.

Первой ключей особенностью разрабатываемого протокола является покрытие значительной территории за счёт применения процедуры эстафетной передачи сообщения через промежуточные узлы сенсорной сети.

Вторая ключевая особенность состоит в возможности автоматической перестройки сети, в случае если узлы по каким-то причинам поменяли своё положение относительно друг друга. Эта особенность при определённых условиях в теории может обеспечивать обмен информацией между мобильными узлами сети. В текущей версии протокола из-за особенностей алгоритма перестроения сети (при изменении топологии перестраивается вся сеть) сеть может эффективно обслуживать только объекты с низкой мобильностью, т.е. когда значительные изменения топологии происходят не слишком часто. Однако и такой вариант функционирования сети открывает множество возможностей для автономных систем мониторинга и управления в производстве, горнодобывающей промышленности, сельском хозяйстве, логистике (складские процессы), ЖКХ, обеспечении безопасности охраняемых объектов, предотвращении чрезвычайных ситуаций техногенного и природного характера, мероприятий экологической направленности и т.д.

Неактивный узел

Оконечное устройство

Роутер

Выход из строя роутера

Роутер-шлюз

Направления передачи данных (зона прямой радиовидимости)

Рисунок 1. Пример механизма самоорганизации сети при отказе узла (роутера).

Третья ключевая особенность представлена в механизмах обеспечения доставки пакетов по сети. Почти каждый передаваемый пакет проходит процедуру квитирования (подтверждения доставки). Однако в качестве квитанции служит не ответный пакет от принимаемого узла, а пакет на передачу следующему узлу. Т.к. радиоканал вещает во все стороны, то он должен достичь как следующего получателя в эстафете, так и предыдущего отправителя.

Отправка сообщения в радиоэфир



Приём сообщения с соседнего узла и регистрация доставки отправленного сообщения

Игнорирование сообщения

Приём сообщения для ретрансляции

Роутеры, через которые лежит путь сообщения

Оконечные устройства

Роутер-шлюз



Рисунок 2. Схема механизма квитирования Протокола.

Протокол строит сеть по топологии типа “дерево”. Среди беспроводных протоколов связи данная топология встречается реже нежели “звездообразная” или ячеистая (mesh). Это связано с такими недостатками топологии, как малая связность сети (ограниченное число связей, а следовательно резервных путей), отказ корневых точек приводит к отказу целого сектора (ветки), которая находится за ним, слабая возможность балансировки нагрузки (корневые узлы первых уровней неизбежно будут прогонять через себя весь трафик от узлов, расположенных ниже по ветке) и др.

# **Состав и топология сети.**

Устройства, работающие на базе Протокола, в процессе взаимодействия между собой образуют сеть с древовидной топологией. Сеть состоит из 3 типов узлов:

1. Роутер
2. Роутер-шлюз
3. Оконечное устройство

*Роутер –* промежуточный узел сети, выполняющий функции оконечного устройства и ретранслятора одновременно. Переправляет полученные пакеты узлу-получателю или следующему узлу, через которого лежит путь к получателю.

*Роутер-шлюз* – корневой узел сети, от которого строится сеть. Агрегирует полученные от устройств сети данные и направляет их на сервер через IP-шлюз. Инициирует процедуры построения сети и сбора данных с устройств сети.

*Оконечное устройство* – узел сети, функции которого ограничены передачей данных с датчиков (приложений), по запросу.

**Типы узлов сети**

Оконечное устройство

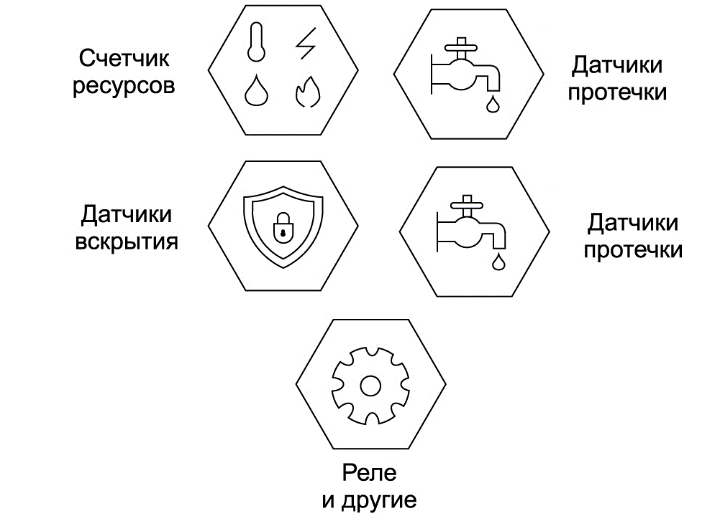
Роутер-шлюз

Роутер

Рисунок 3. Топология сети, работающей на базе Протокола.

# **Аппаратная реализация узлов сети Протокола.**

Каждый узел сети, работающей на базе протокола, должен состоять из платы управления на базе микроконтроллера, поддерживающей стандартный набор команд языка C, радио модуля и источника/потребителя информации (периферия), которыми, как правило в подобных сетях выступают датчики и исполнительные механизмы (Рисунок4).



Протокол связи

получение данных с датчика

Получение/отправка данных в эфир

Основная логика

(управление отправкой, предварительная обработка данных, формирование сообщения и др.)



**Модуль связи**

**Периферия**

**Плата управления (контроллер)**

Рисунок 4. Состав узла сети Протокола.

Возможно также исполнение в виде единой платформы беспроводной связи по типу CC1310 SimpleLink™ Ultra-Low-Power Sub-1 GHz Wireless MCU, в котором имеется полноценная операционная система TI-RTOS. Преимуществом данного модуля являются реализованная в самом модуле функция, облегчающие реализацию множественного доступа к среде передачи. Такими функциями являются LBT (Listen before talk) и CCA (clear channel assist). Помимо этого, реализован удобный интерфейс вывод значения уровня принимаемого радиосигнала (RSSI), поддерживаются различные схемы модуляции (OOK, FSK и MSK). Основным недостатком такого решения является высокая сложность развёртывания и настройки узла при подключении узла к сети, работающему на базе протокола. Разработка под данную платформу требует установку специальной среды разработки от TI, а также использование специального SDK, которое реализует взаимодействие с ядром TI-RTOS, на базе которой работает данная плата. Данное обстоятельство делает эту плату слабо пригодной для кроссплатформенной разработки.



Рисунок 5. Модуль SimpleLink™ Sub-1 GHz CC1310 wireless MCU LaunchPad.

Именно поэтому в качестве аппаратного тестового модуля решили выбрать платформу Arduino UNO. Разработка под данную платформу не требует написания большого количества кода для взаимодействия с микроконтроллером. Все необходимые инструкции изящно упакованы в максимально простые в понимании и написании структуры кода. Это позволяет максимально отделить код, реализующий работу протокола, от остальной части приложения.

Arduino Uno контроллер построен на ATmega328. Платформа имеет 14 цифровых вход/выходов (6 из которых могут использоваться как выходы ШИМ), 6 аналоговых входов, кварцевый генератор 16 МГц, разъем USB, силовой разъем, разъем ICSP и кнопку перезагрузки. Для работы необходимо подключить платформу к компьютеру посредством кабеля USB, либо подать питание при помощи адаптера AC/DC или батареи.

Arduino Uno может получать питание через подключение USB или от внешнего источника питания. Источник питания выбирается автоматически.

Внешнее питание (не USB) может подаваться через преобразователь напряжения AC/DC (блок питания) или аккумуляторной батареей. Преобразователь напряжения подключается посредством разъема 2.1 мм с центральным положительным полюсом. Провода от батареи подключаются к выводам Gnd и Vin разъема питания.

Платформа может работать при внешнем питании от 6 В до 20 В. При напряжении питания ниже 7 В, вывод 5V может выдавать менее 5 В, при этом платформа может работать нестабильно. При использовании напряжения выше 12 В регулятор напряжения может перегреться и повредить плату. Рекомендуемый диапазон от 7 В до 12 В.

Выводы питания:

* **VIN**. Вход используется для подачи питания от внешнего источника (в отсутствие 5 В от разъема USB или другого регулируемого источника питания). Подача напряжения питания происходит через данный вывод.
* **5V**. Регулируемый источник напряжения, используемый для питания микроконтроллера и компонентов на плате. Питание может подаваться от вывода VIN через регулятор напряжения, или от разъема USB, или другого регулируемого источника напряжения 5 В.
* **3V3**. Напряжение на выводе 3.3 В генерируемое встроенным регулятором на плате. Максимальное потребление тока 50 мА.
* **GND**. Выводы заземления.

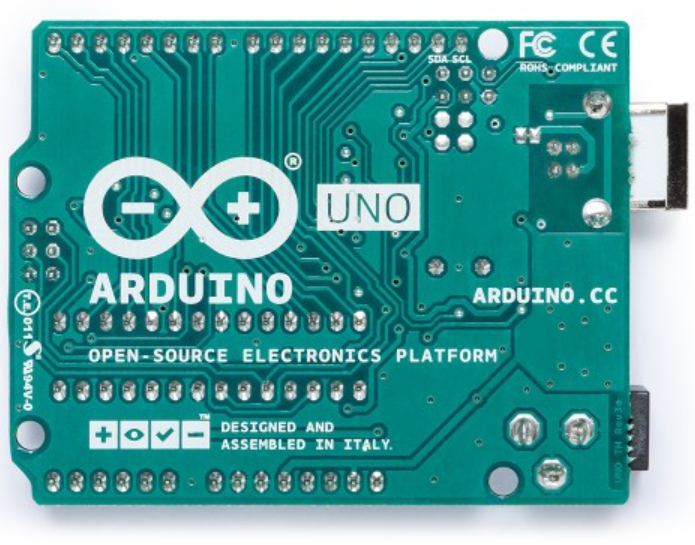
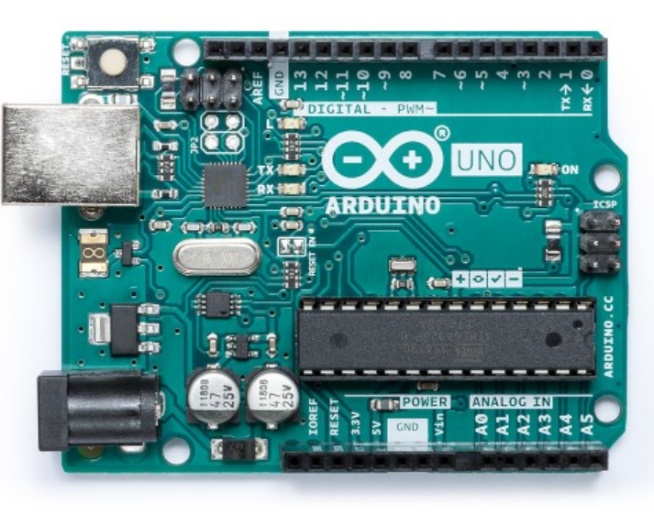


Рисунок 6. Внешний вид модуля Arduino UNO.

В качестве приёмопередатчика используется модуль HC-12. Модуль представляет собой печатную плату с необходимыми элементами размером 27,8 х 14,4 мм. Беспроводной модуль имеет возможность подключения антенны двумя способами: подключение антенны через разъем IPEX20279-001E-03 (ANT1) или просто припаять кусочек провода (к контакту 6) (ANT2) или другой разъем (контакты 7,8 - земля). Для подключения к другим устройства модуль имеет пятиконтактный разъем (PLS/PBS), где 1 – плюс питания, 2 – земля, 3 – RxD, 4 – TxD, 5 – SET. Вывод 5 SET подтянут 10 кОм резистором к плюсу питания. Этот вывод необходим для активации настроек при помощи AT команд. Активация происходит при подключении вывода к низкому уровню, то есть к земле.

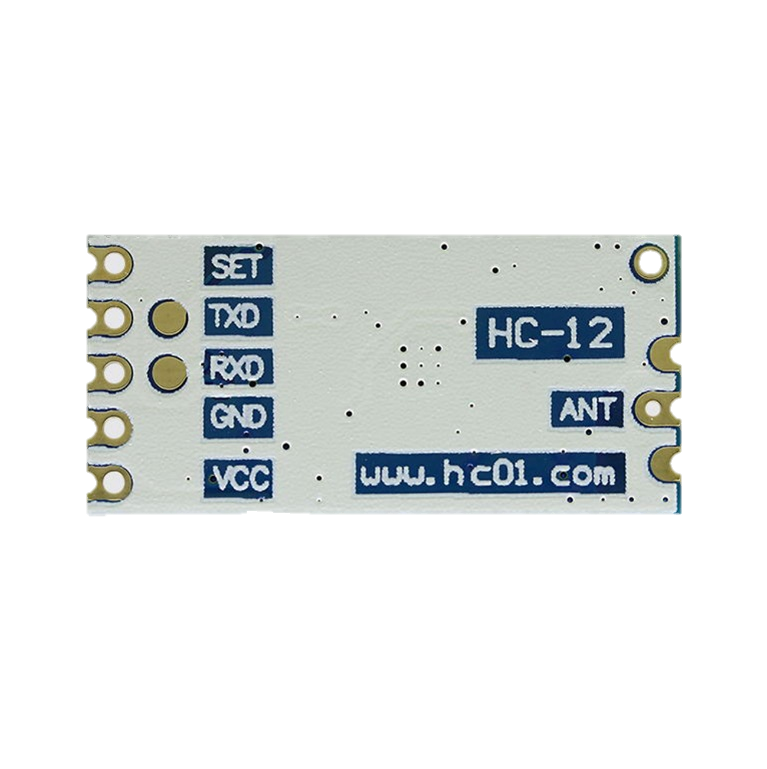
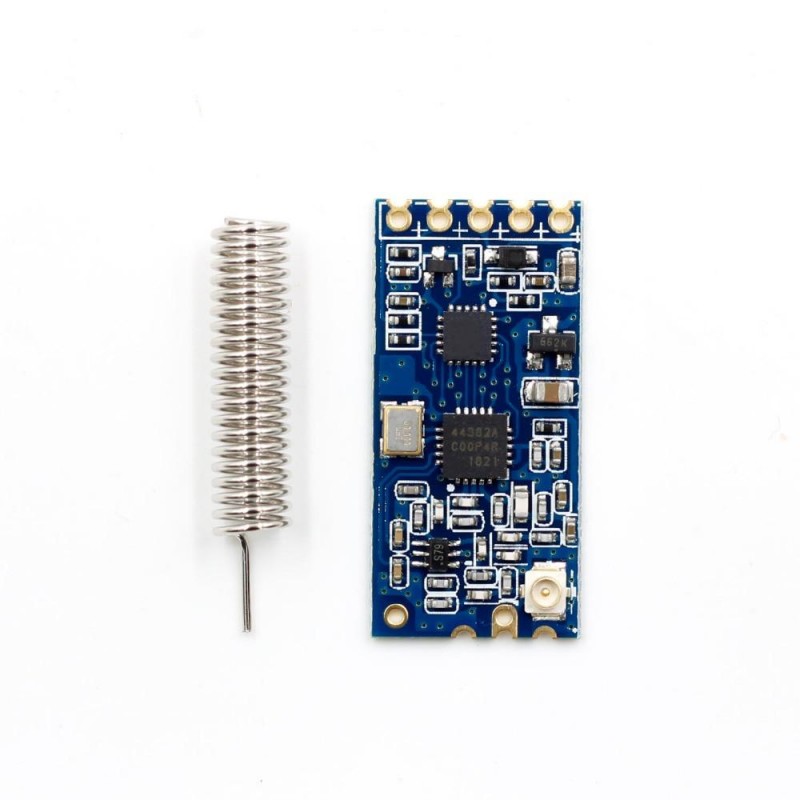


Рисунок 7. Радио приёмопередатчик HC-12.

Для удобства использования функционал платы можно расширить бесчинным множеством дополнительных модулей. Например, в тестируемой установки для того, чтобы иметь возможность видеть основные показатели протокола связи, к тестовой платформе был подключён специальный дисплей SSD1306 , на который планируется выводить сообщения о наиболее значимых событиях: включение питания, изменение статуса, приём пакета, оправка пакета, показания датчиков и др. Вывод данных на экран осуществляется через специальную библиотеку от Arduino, которая очень сильно упрощает работу с ним. Вид тестовой платформы для разработки и тестирования реализации протокола представлен ниже ()



Рисунок 8. Аппаратная реализация тестовой платформы для передачи данных в сенсорной сети, построенной на базе протокола.

Функциональная схема подключения различных модулей тестовой установки представлена ниже (см. Рисунок 8).

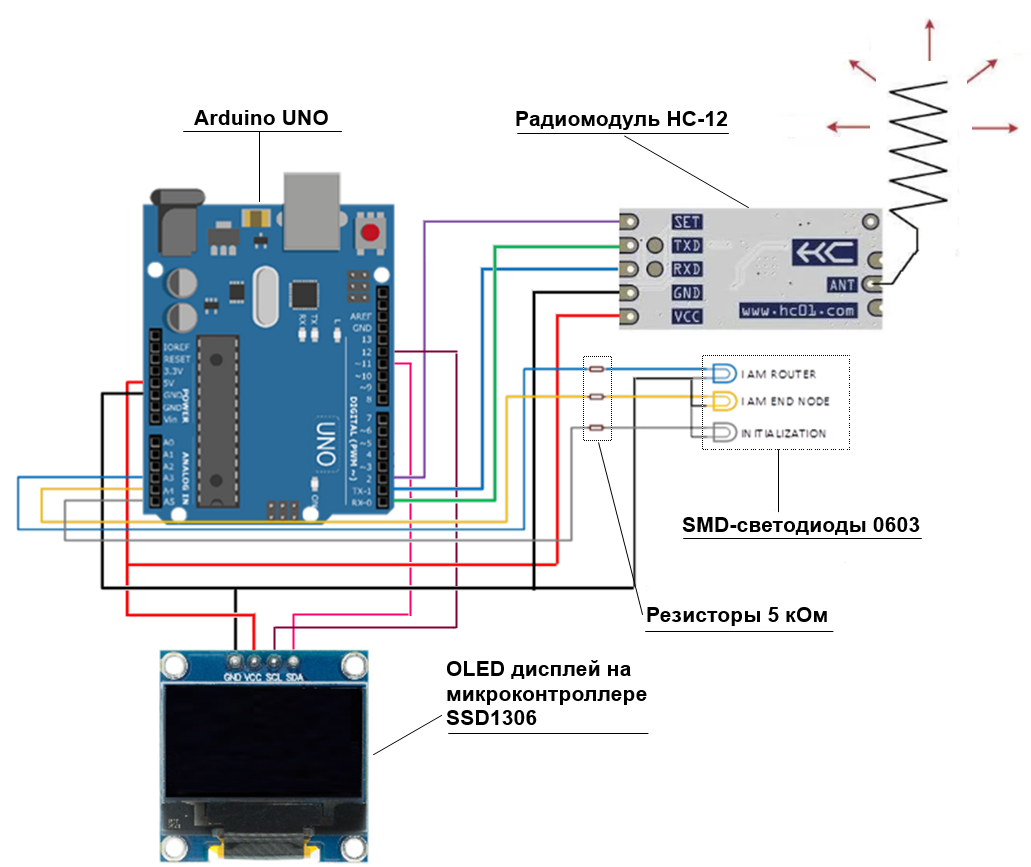


Рисунок 9. Состав тестовой платформы узла сенсорной сети и порядок подключения её составных элементов.

Резисторы на входе перед светодиодами подбираются исходя из характеристик светодиодов. В приведённой ранее на рис тестовой платформы используются резисторы с номиналом 5.1 кОм +- 5% в связке с smd-диодами 0603. Питание на светодиоды подаётся с управляемых аналоговых выводов платы Arduino UNO. OLED-дисплей SSD1306 питается от источника напряжения 5В.

Вывод данных на дисплей осуществляется путём подачи необходимых команд на контроллер SSD1306 через интерфейс I2C. Для этого достаточно подключить выводы интерфейса I2C – SDA (шина последовательных данных) и SCL (шина тактирования) к любым цифровым выводам на плате Arduino UNO. В представленной установке эти шины подключены к цифровым выводам 11 (SDA) и 12 (SCL). Управление OLED экраном на программном уровне происходит с помощью библиотеки OLED\_I2C.

Радиомодуль HC12 питается от напряжения 5В. На входы через входы Rx и Tx тестовая платформа передаёт и принимает радиосигналы с эфира. Приём и передача данных с этих выводов происходит по TTL-логике. Взаимодействие с платой Arduino обычно производится через интерфейс COM-to-TTL. Контроллер выдаёт данные по протоколу COM на внутренний преобразователь COM-TTL, который строится как правило на микросхеме MAX232. На выходе этой схемы выдаётся сигнал в форме TTL-логики. На плате Arduino UNO имеются специальные выводы с внутреннего COM-TTL преобразователя на выделенные цифровые выводы, которые промаркированы RX и TX. Также они соответствуют цифровым выводам 1 (Tx) и 0 (Rx). Вывод SET предназначен для перевода радиомодуля режим настройки конфигурации. Для возможности управления процессом переключения между режимами через Arduino необходимо подключить вывод SET на радиомодуле HC12 с любым цифровым управляемым выводом на плате Arduino UNO. В представленной выше тестовой платформе вывод SET подключен к цифровому выводу 3.

Для удобства подключения, монтажа и тестирования подключение каждого из модулей происходит через платы прототипирования и макетные платы.

Ниже приведена таблица, в которой отражены все необходимые составные элементы для сборки одной тестовой платформы (1 узла сети протокола).

Таблица 1. Состав тестовой платформы узла сенсорной сети.

|  |  |  |
| --- | --- | --- |
| Тип составленного элемента платформы | Модель | кол-во |
| Радиопередатчик | HC12 | 1 |
| Светодиоды | SMD 0603 | 3 |
| Резисторы | любой с номиналом 5 КОм | 3 |
| Плата прототипирования | UNO ProtoShield | 1 |
| Двухсторонняя печатная плата | любая печатная плата 2 см на 8 см | 1 |
| OLED-модуль | SSD1306 | 1 |
| Источник автономного питания | любая батарея 5В | 1 |
| Комплект соединительный проводов | - | в зависимости от исполнения |
| Антенна | антенна 433 МГц | 1 |

# **Структура пакетов протокола**

Протокол подразумевает 7 основных типов пакетов. Пакет состоит из заголовка и полезной нагрузки. Заголовок имеет фиксированную длину – 28 байт. Структура заголовка едина для всех пакетов, передаваемых по сети. Каждому типу пакета соответствует определённый набор значений служебных полей, содержащихся в заголовке.

Под полезной нагрузкой подразумевается область пакета, содержащая пользовательские данные. Пользовательские данные – это сведения, которые протокол не использует для продвижения пакетов по сети. В качестве пользователей сети могут выступать как сторонние сервисы (например сервис сбора данных о температуре теплоносителей), так и внутренние сервисы, поддерживающие работу самой сети (например сервис маршрутизации). Длинна полезной нагрузки может варьироваться от 0 до 99 байт. Максимальная длинна пакета – 128 байт.

## **Общая структура заголовка**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *№ байта* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| № поля | **1** | **2** | **3** | | | | **4** | | | | **5** | | **6** | **7** | **8** | **9** | **10** | **11** | **12** | | | | **13** | | | | **14** | |

## **Описание полей заголовка**

1. **Начало пакета** – первый байт пакета содержит символ $ (0x24);
2. **Тип пакета** – содержит код, который однозначно определяет тип пакета. В зависимости от типа пакета принимает значения 0-6 (0x00 – 0x06);
3. **Адрес узла отправителя** – содержит адрес узла отправителя, состоящий из 4 байт.
4. **Адрес узла получателя** – содержит адрес узла получателя, состоящий из 4 байт.
5. **Время** – содержит текущее время от начала сеанса связи в квантах.
6. **Уровень** – Порядковый номер роутера, относительно шлюза. Роутеры находятся на одном уровне, если число узлов, вдоль которых лежит путь сообщения от указанных роутеров до роутера-шлюза одинаково.
7. **Сессия** – содержит код сессии. Под сессией понимается процесс функционирования сети, от начала построения сети, до её перестроения**.** Перестройка сети происходит в двух случаях: принудительно или при повторном включении питания роутера-шлюза. В течении одной сессии значение соответствующего поля в заголовке пакетов остаётся неизменным.
8. **Сеанс** – содержит код сеанса. Под сеансом подразумевается процесс опроса сети на предмет полезных данных (данные с датчиков или результаты работы приложений устройств сети). Код представляет собой порядковый номер процесса опроса сети на предмет полезных данных и служит для сбора статистики.
9. **Роль модуля** – содержит код, определяющий роль устройства, передающего пакет (“0x01” -роутер, “0x02” - альтернативный роутер,”0x00” – оконечное устройство).
10. **Номер пакета** – содержит порядковый номер пакета, позволяющий отбрасывать свои пакеты при широковещательной рассылке команды на сбор данных с устройств сети. Пакеты типа “00” –“Я потенциальный номер всегда имеют номер “0x00”, все последующие пакеты генерируются с номером на 1 больше предыдущего. При смене сессии нумерация пакетов начинается заново.
11. **TTL –** значение счётчика жизни пакета. Может использоваться для определения направления движения пакета
12. **Адрес следующего промежуточного узла получателя** – содержит адрес узла, находящегося в зоне радиовидимости, который является получателем или промежуточным узлом на пути к получателю. Состоит из 4 байт.
13. **Адрес предыдущего промежуточного узла отправителя** – содержит адрес промежуточного узла, от которого поступил пакет. Необходим для подтверждения предыдущим узлом факта получения пакета. Состоит из 4 байт.
14. **Резерв** – резервное поле.

# **Типы пакетов**

## **00 -“Я потенциальный роутер”**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *№ байта* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| № поля | **$** | **00** | **<адрес потенциального роутера>** | | | | **00 00 00 00** | | | | **<время отправки>** | | **<уровень>** | **<сессия>** | **00** | **<роль модуля>** | **00** | **01** | **55 55 55 55** | | | | **00 00 00 00** | | | | **55 55** | | **#** |

**$** - начало пакета

**<адрес потенциального роутера> -** устройство, которое хочет объявить себя роутером формирует пакет 00 –“Я потенциальный роутер” и помещает в поле 3 - “адрес узла отправителя” свой адрес.

**<время отправки>** - время отправки в квантах. Каждый потенциальный роутер при первичной отправке данного пакета сбрасывает значение данного поля.

**<уровень>** - В пакете типа 00 -“Я потенциальный роутер” помещается значение уровня. Уровень определяется исходя из значения, которое потенциальный роутер получил от устройства, которое ранее потенциальный роутер выбрал в качестве своего роутера.

**<сессия>** - номер сессии. Присваивается при инициации процедуры построения или перестроения сети.

**<роль модуля>** - код роли модуля. В пакете типа 00 - “Я потенциальный роутер” может принимать только значения 0x00 и 0x01. Значение 0x01 соответствует роли “роутер”, 0x00 – роли “устройство”. В данном типе пакета значение “0x01” может помещаться только исходным роутером-шлюзом, поскольку его роль всегда “роутер”. В остальных случаях потенциальные роутеры по умолчанию имеют роль “устройство” в данном пакете.

**# -** конец пакета

Пакет типа “00” - “Я потенциальный роутер” состоит только из заголовка и символа конца пакета – ‘#’.

## **01-“Я - узел”**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Заголовок пакета* | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| *№ байта* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| № поля | **$** | **01** | **<адрес узла>** | | | | **00 00 00 00** | | | | **<время отправки>** | | **<уровень>** | **<сессия>** | **00** | **00** | **<№ пакета>** | **01** | **55 55 55 55** | | | | **00 00 00 00** | | | | **55 55** | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Полезная нагрузка пакета* | | | | | | | | | | |
| *№ байта* | *29* | *30* | *31* | *32* | *33* | *34* | *35* | *36* | *……….* | *N < = 128* |
| № поля | **<адрес потенциального роутера 1>** | | | | **<адрес потенциального роутера 2>** | | | | **<адреса других обнаруженных потенциальных роутеров>** | **#** |

**$** - начало пакета

**<адрес узла> -** устройство в ответ на пакет 00 - “Я потенциальный роутер” (0x00) формирует пакет 01 -“Я узел” (0x01) и помещает в поле 3 - “адрес узла отправителя” свой адрес.

**<время отправки>** - время отправки в секундах. За основу берётся время, полученное с прошлого пакета. Пред отправкой его увеличивают время, которое прошло между получением последнего пакета и отправкой ответного пакета.

**<уровень>** -уровень определяется исходя из значения, которое устройство получило от потенциального роутера. Определяет удалённость узла от Роутера-шлюза.

**<сессия>** - номер сессии определяется из соответствующего поля входящего пакета 00 - “Я потенциальный роутер” (0x00).

**<№ пакета>** - номер пакета устанавливается относительно номера последнего отправленного пакета. При отправке пакета 00 – “Я потенциальный роутер” номер сбрасывается.

**<адрес потенциального роутера>** - помещает адреса устройств, от которых получил пакет 00 - “Я потенциальный роутер” (0x00). В зависимости от числа пакетов, принятых от разных роутеров число полей с адресами, занимающих пространство полезной нагрузки пакетов, может варьироваться от 1 до 24. Т.е. теоретически один узел может уведомить соседние устройства о наличии прямого соединения 24 потенциальными роутерами.

**# -** конец пакета

## **02-“Я выбрал роутер”**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Заголовок пакета* | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| *№ байта* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| № поля | **$** | **02** | **<адрес узла>** | | | | **00 00 00 00** | | | | **<время отправки>** | | **<уровень>** | **<сессия>** | **00** | **00** | **<№ пакета>** | **01** | **55 55 55 55** | | | | **00 00 00 00** | | | | **55 55** | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Полезная нагрузка пакета* | | | | | | | | | |
| *№ байта* | *29* | *30* | *31* | *32* | *33* | *34* | *35* | *36* | *37* |
| № поля | **<адрес основного роутера>** | | | | **<адрес резервного роутера>** | | | | **#** |

**$** - начало пакета

**<адрес узла> -** устройство формирует пакет 02 - “Я выбрал роутер” (0x02) и помещает в поле 3 “адрес узла отправителя” свой адрес.

**<время отправки>** - время отправки в секундах. За основу берётся время, полученное с прошлого пакета. Пред отправкой его увеличивают время, которое прошло между получением последнего пакета и отправкой ответного пакета.

**<уровень>** - уровень определяется исходя из значения, которое устройство получило от потенциального роутера.

**<сессия>** - номер сессии определяется из соответствующего поля входящего пакета 00 - “Я потенциальный роутер” (0x00).

**<№ пакета>** - номер пакета устанавливается относительно номера последнего отправленного пакета. При отправке пакета 00 – “Я потенциальный роутер” номер сбрасывается.

**<адрес основного роутера>** - помещает адрес устройства, которое узел выбрал себе в качестве основного роутера. Именно по этому адресу в дальнейшем будут отправляться пакеты от своих оконечных устройств и низлежащих роутеров.

**<адрес резервного роутера>** - помещает адрес устройства, которое узел выбрал себе в качестве резервного роутера. На данный адрес будут отправляться пакеты, которые должны были быть отправлены на основной роутер, в случае если устройство потеряет связь с основным роутером (например, не принял сообщение от следующего роутера, который отправляет сообщение вверх следующему роутеру). В том случае, если резервный роутер не может выбрать резервный роутер (это возможно, когда устройство приняло сообщение только от одного потенциального роутера, и выбрать резервный), данное поле заполняется балластом типа “00 00 00 00".

**# -** конец пакета.

Пакет 02 - “Я выбрал роутер” в силу своей специфики имеет фиксированный размер пакета в 37 байт, полезная нагрузка которого занимает 9 байт.

## **03-“Я роутер”**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Заголовок пакета* | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| *№ байта* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| № поля | **$** | **03** | **<адрес роутера>** | | | | **00 00 00 00** | | | | **<время отправки>** | | **<уровень>** | **<сессия>** | **00** | **01** | **<№ пакета>** | **01** | **55 55 55 55** | | | | **00 00 00 00** | | | | **55 55** | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Полезная нагрузка пакета* | | | | | | | | | | |
| *№ байта* | *29* | *30* | *31* | *32* | *33* | *34* | *35* | *36* | *……….* | *N < = 128* |
| № поля | **<адрес оконечного устройства 1>** | | | | **<адрес оконечного устройства 2>** | | | | **<адреса других оконечных устройств, которые выбрали данный узел в качестве своего роутера>** | **#** |

**$** - начало пакета

**<адрес роутера> -** устройство, которое другие устройства выбрали в качестве роутера, объявляет в сеть о том, что он теперь роутер, отправляя в сеть пакет 03 – “Я роутер”. В поле заголовка 3 - адрес узла отправителя он помещает свой адрес.

**<время отправки>** - время отправки в секундах. За основу берётся время, полученное с прошлого пакета. Пред отправкой его увеличивают время, которое прошло между получением последнего пакета и отправкой ответного пакета, либо на промежуток времени между приёмом и отправкой ретранслируемого пакета.

**<уровень>** - уровень был получен роутером от роутера более высокого уровня, на этапе, когда данный узел выбирал свой роутер. Исключением является роутер-шлюз, который по умолчанию имеет уровень 0.

**<сессия>** - номер сессии был получен аналогично уровню.

**<№ пакета>** - номер пакета устанавливается относительно номера последнего отправленного пакета. Исключением из данного правила является пакет типа 00– “Я потенциальный роутер”. Он всегда имеет значение 00. От него начинается счёт пакетов.

**<адрес оконечного устройства>** - помещает адреса узлов, от которых он получил пакет 02 – “Я выбрал роутер”, где в полезной нагрузке был указан его адрес, как адрес основного роутера.

**# -** конец пакета

## **04-“Опрос устройств”**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Заголовок пакета* | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| *№ байта* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| № поля | **$** | **04** | **<адрес роутера- шлюза>** | | | | **00 00 00 00** | | | | **<время отправки>** | | **00** | **<сессия>** | **<сеанс>** | **00** | **<№ пакета>** | **1e** | **ff ff ff ff** | | | | **<признак подтвреждения>** | | | | **55 55** | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Полезная нагрузка пакета* | | | | | | | | | | |
| *№ байта* | *29* | *30* | *31* | *32* | *33* | *34* | *35* | *36* | *……….* | *N < = 128* |
| № поля | **<адрес подчинённого устройства 1>** | | | | **<адрес подчинённого устройства 2>** | | | | **<адреса других подчинённых устройств, которые выбрали данный узел в качестве своего роутера>** | **#** |

**$** - начало пакета

**<время отправки>** - время отправки в секундах. За основу берётся время, полученное с прошлого пакета. Пред отправкой его увеличивают время, которое прошло между получением последнего пакета и отправкой ответного пакета.

**<сессия>** - номер сессии присваивается при инициализации сети. Смена номера сессии, говорит устройству, что необходимо заново определить свою роль в сети и заново провести операции по построению сети.

**<сеанс> -** помещается номер сеанса связи. Номер сеанса меняется при инициации роутером шлюзом очередного опроса сети.

**<№ пакета>** - номер пакета устанавливается относительно номера последнего отправленного пакета. Исключением из данного правила является пакет типа 00– “Я потенциальный роутер”. Он всегда имеет значение 00. От него начинается счёт пакетов.

**<адрес оконечного устройства>** - помещает адреса своих оконечных (низлежащих) устройств.

**# -** конец пакета

## **05-“Ответ от устройства UNO”**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Заголовок пакета* | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| *№ байта* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| № поля | **$** | **05** | **<адрес оконечного устройства>** | | | | **<адрес роутера>** | | | | **<время отправки>** | | **<уровень>** | **<сессия>** | **<сеанс>** | **00** | **<№ пакета>** | **01** | **55 55 55 55** | | | | **00 00 00 00** | | | | **55 55** | |

|  |  |  |  |
| --- | --- | --- | --- |
| *Полезная нагрузка пакета* | | | |
| *№ байта* | *29* | *……….* | *N < = 128* |
| № поля | **<Данные от устройства (приложения)>** | | **#** |

**$** - начало пакета

**<адрес оконечного устройства>** - адрес устройства, которое сформировало и отправила пакет в сеть.

**<адрес роутера>** - адрес роутера, которому предназначен пакет. Оконечное устройство отправляет пакет 05 “ Ответ от устройства UNO” только тому устройству, которое приходится ему основным или резервным роутером.

**<время отправки>** - время отправки в секундах. За основу берётся время, полученное с прошлого пакета. Пред отправкой его увеличивают время, которое прошло между получением последнего пакета и отправкой ответного пакета.

**<уровень>** - уровень, на котором располагается устройство.

**<сессия>** - номер сессии присваивается при инициализации сети. Смена номера сессии, говорит устройству, что необходимо заново определить свою роль в сети и заново провести операции по построению сети.

**<сеанс> -** помещается номер сеанса связи. Номер сеанса меняется при инициации роутером шлюзом очередного опроса сети.

**<№ пакета>** - номер пакета устанавливается относительно номера последнего отправленного пакета. Исключением из данного правила является пакет типа 00– “Я потенциальный роутер”. Он всегда имеет значение 00. От него начинается счёт пакетов.

**<Данные от устройства (приложения)>** - приложение на устройстве либо вспомогательное устройство, подключенное к устройству – узлу сети, помещает в данное поле свои данные, которые адресованы роутеру шлюзу

**# -** конец пакета

## **06-“Ответ от роутера MANY“**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Заголовок пакета* | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| *№ байта* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| № поля | **$** | **06** | **<адрес исходного отправителя>** | | | | **<адрес конечного получателя>** | | | | **<время отправки>** | | **<уровень>** | **<сессия>** | **<сеанс>** | **01 (02)** | **<№ пакета>** | **01** | **<Адрес следующего роутера>** | | | | **<Адрес предыдущего роутера>** | | | | **55 55** | |

|  |  |  |  |
| --- | --- | --- | --- |
| *Полезная нагрузка пакета* | | | |
| *№ байта* | *29* | *……….* | *N < = 128* |
| № поля | **<Данные от устройств>** | | **#** |

**$** - начало пакета

**<адрес исходного отправителя>** - адрес роутера, сформировавшего пакет MANY.

**<адрес конечного получателя>** - адрес роутера шлюза.

**<время отправки>** - время отправки в секундах. За основу берётся время, полученное с прошлого пакета. Пред отправкой его увеличивают время, которое прошло между получением последнего пакета и отправкой ответного пакета.

**<уровень>** - уровень, на котором располагается устройство.

**<сессия>** - номер сессии присваивается при инициализации сети. Смена номера сессии, говорит устройству, что необходимо заново определить свою роль в сети и заново провести операции по построению сети.

**<сеанс> -** помещается номер сеанса связи. Номер сеанса меняется при инициации роутером шлюзом очередного опроса сети.

**<№ пакета>** - номер пакета устанавливается относительно номера последнего отправленного пакета. Исключением из данного правила является пакет типа 00 – “Я потенциальный роутер”. Он всегда имеет значение 00. От него начинается счёт пакетов.

**<Данные от устройств>** - роутер при в ответ на отправленный пакет 04 – “Опрос устройств”, принимает от своих устройств сообщения 05 –“ Ответ от устройства UNO”. Из них он вытаскивает данные полезной нагрузки и последовательно помещает в полезную нагрузку своего пакета 06 -“ Ответ от роутера MANY” данные из полученного пакета в формате *адрес устройства + данные.*

**# -** конец пакета.

# **Описание работы протокола**

Работу протокола можно разделить на два этапа: построение сети и передача данных.

## **Этап построения сети**

На первом этапе размещённые случайным образом устройства выстраиваются в топологию типа “дерево”. Отправной точкой построения сети является устройство, которое носит название “Роутер-шлюз”. Его главная отличительная особенность – заранее предустановленная роль “Роутер". Остальные устройства до начала построения сети имеют роль “Оконечное устройство” и находятся в состоянии “Сон”. Состояние “Сон” — это такой режим работы устройства, при котором оно ничего не вещает в сеть, до момента поступления на него специального пакета. До этого момента устройство только прослушивает эфир.

На рамках построения сети каждое устройство, за исключением шлюза, проходит две стадии построения сети. На первой из них устройство определяет свой основной и резервный роутеры. На второй стадии оно определяет свою роль в сети и список подчинённых ему устройств.

?

Потенциальный роутер 1

Потенциальный роутер 2

Неопределённое устройство (сон)

Неопределённое устройство (сон)

2. Определение основного и резервного роутеров

1. Объявление себя Потенциальным роутером

2.Ожидание решений устройств относительно выбора его роутером

1. Пробуждение путём приёма пакетов от потенциального роутеров

I

II

Устройство

Рисунок 10. Стадии, на которые разделён этап работы протокола по построению сети.

На этапе построения сети два устройства общаются друг с другом, находясь при этом на различных стадиях этапа построения сети. Каждое устройство обязательно должно пройти обе стадии этапа построения сети. Исключением из этого правила является роутер-шлюз.

II

I

II

I

Неопределённые “пробуждённые”

устройства

Потенциальный роутер

Стадии этапа построения сети (выполняется 2 стадия)

II

I

Стадии этапа построения сети (выполняется 1 стадия)

II

I

Рисунок 11. Соотношение стадий построения сети потенциального роутера и неопределённого устройства.

Процесс построения сети начинается с отправки потенциальным роутером в эфир пакета типа 00 – “Я потенциальный роутер”. Устройства, расположенные в зоне прямой радиовидимости и имеющие статус неопределённости, принимают данный пакет и извлекают из него адрес отправителя. На основании полученного адреса устройство регистрирует потенциальный роутер в специальном списке, который носит название “Список потенциальных роутеров”. Регистрация происходит путём создания записи, состоящей из адреса потенциального роутера и счётчика, в этом списке. Счётчик представляет собой переменную, которая хранит число соседних устройств, находящихся в зоне прямой радиовидимости относительно данного потенциального роутера.

Регистрация потенциальных роутеров происходит только на основании пакета 00 – “Я потенциальный роутер”.

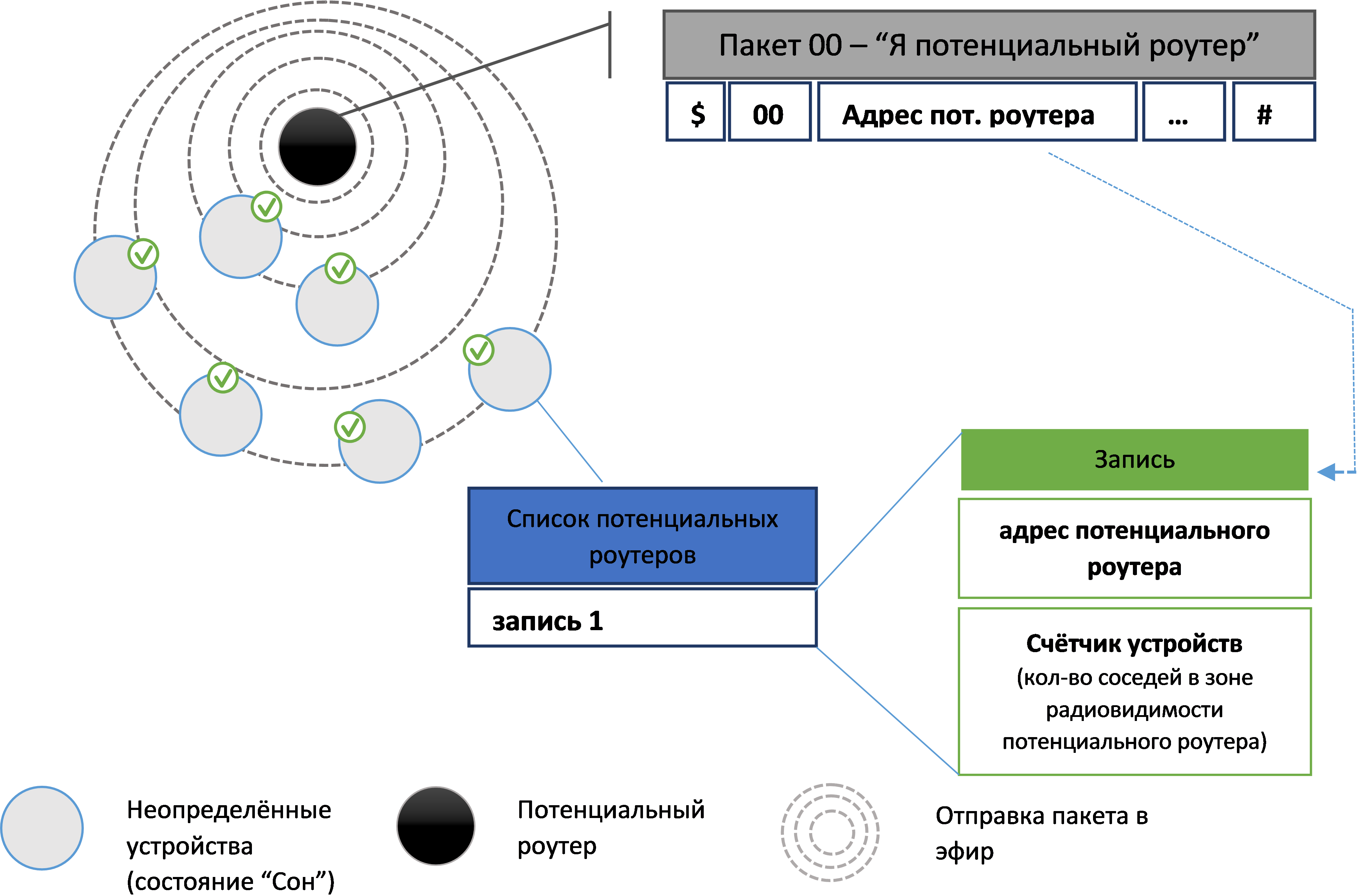


Рисунок 12. Отправка потенциальным роутером пакета 00-"Я потенциальный роутер".

В ответ на пакет 00 – “Я потенциальный роутер” устройства формирует пакет типа 01- “Я узел”, в поле полезной нагрузки которого помещаются адреса потенциальных роутеров, от которых ранее были приняты пакеты типа 00 – “Я потенциальный роутер”. На первом этапе, когда в качестве потенциального роутера выступает роутер-шлюз, устройство поместит только адрес роутера-шлюза. Сформированный пакет отправляется в радиоэфир.

Соседние устройства, за исключением потенциальных роутеров, принимают данный пакет и извлекают из него адреса потенциальных роутеров. Каждое из них производит поиск записей, содержащих извлечённые адреса в своих списках. Если такие записи будут найдены, то значение счётчика устройств в этих записях инкрементируется.

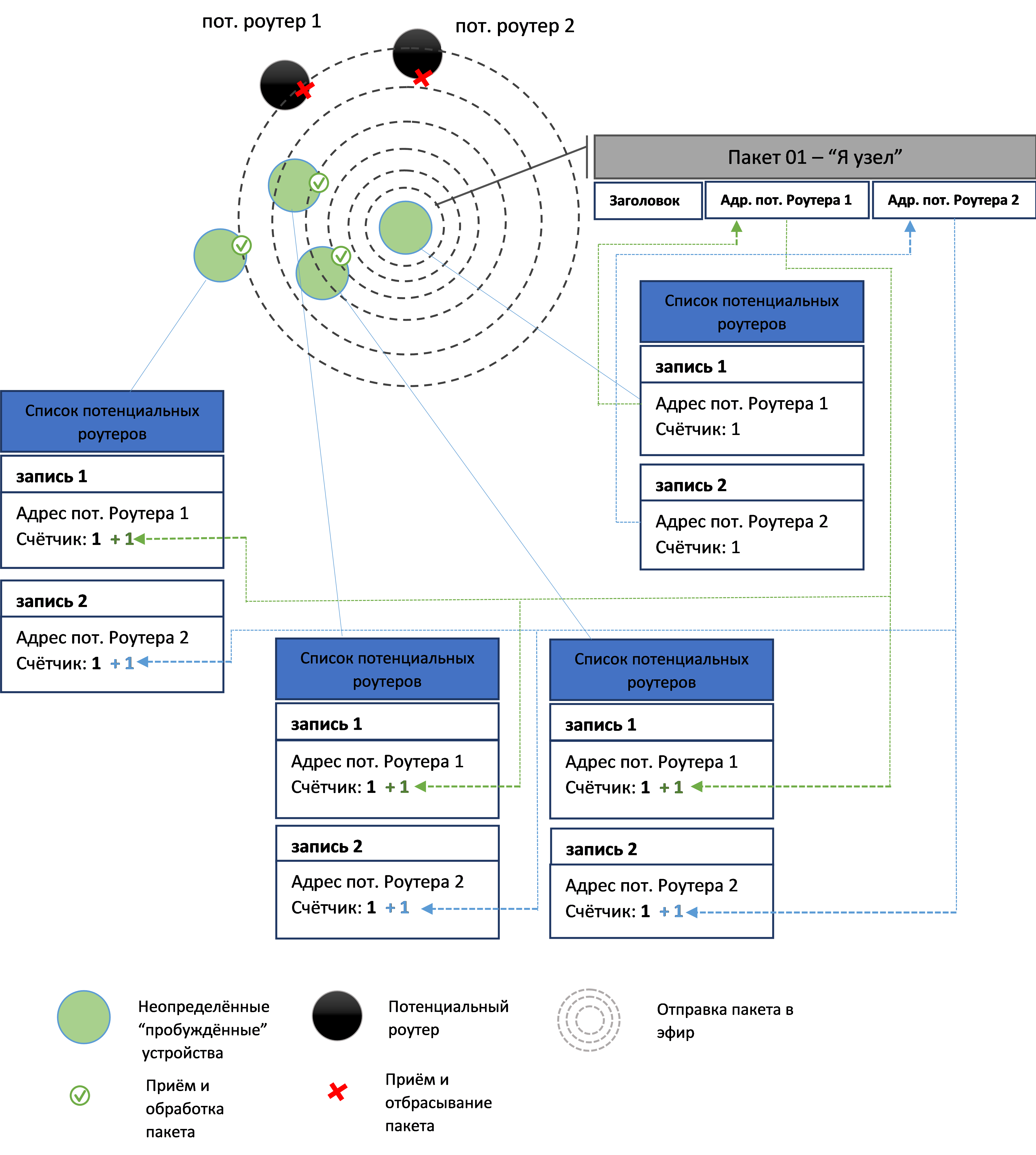


Рисунок 13. Отправка устройством пакета 01-"Я узел".

Через некоторый промежуток времени (предполагается, что все соседние устройства уже “пробудились”, получив пакет 00 – “Я потенциальный роутер”, и заявили о себе и своём местоположении в сети относительно одного из ближайших потенциальных роутеров путём отправки в радиоэфир пакета 01-“Я узел”) каждое “пробудившееся” устройство анализирует свои записи с целью определить основной и резервный роутер. Выбор производится на основании 2 параметров:

1. Количество соседних устройств, находящихся в зоне радиовидимости данного роутера;
2. Уровень принимаемого сигнала (RSSI).

При наличии в списке потенциальных роутеров неопределённых устройств двух и более записей, основным роутером будет выбран тот потенциальный роутер, в область радиовидимости которого входит наибольшее количество соседних неопределённых устройств. Количество соседних устройств, входящих в зону радиовидимости потенциального роутера, хранится в специальной переменной-счётчике. Она является частью записи списка потенциальных роутеров.

пот. роутер 2

пот. роутер 1

**запись 1**

Список потенциальных роутеров

Адрес пот. Роутера 1

Счётчик: 4

**запись 2**

Адрес пот. Роутера 2

Счётчик: 4

**запись 1**

Список потенциальных роутеров

Адрес пот. Роутера 2

Счётчик: 3

**запись 1**

Список потенциальных роутеров

Адрес пот. Роутера 1

Счётчик: 4

**запись 2**

Адрес пот. Роутера 2

Счётчик: 3

Неопределённые “пробуждённые”

устройства

Потенциальный роутер

Область радиосвязи потенциального роутера 2

Область радиосвязи потенциального роутера 1

Выбор устройством потенциального роутера в качестве своего роутера

Рисунок 14. Выбор устройством роутера.

В случае, если два потенциальных роутера имеют равное количество устройств, то выбор падёт на тот потенциальный роутер, уровень сигнала (RSSI) от которого имеет наименьшее значение в пределах некоторого разумного порога (для каждого приёмопередатчика данный параметр будет разным, в зависимости от чувствительности приёмника). Данное решение позволяет в ряде случаев сократить число сеансов радиовещания устройств, путём сокращения количества промежуточных узлов передачи пакета по сети.

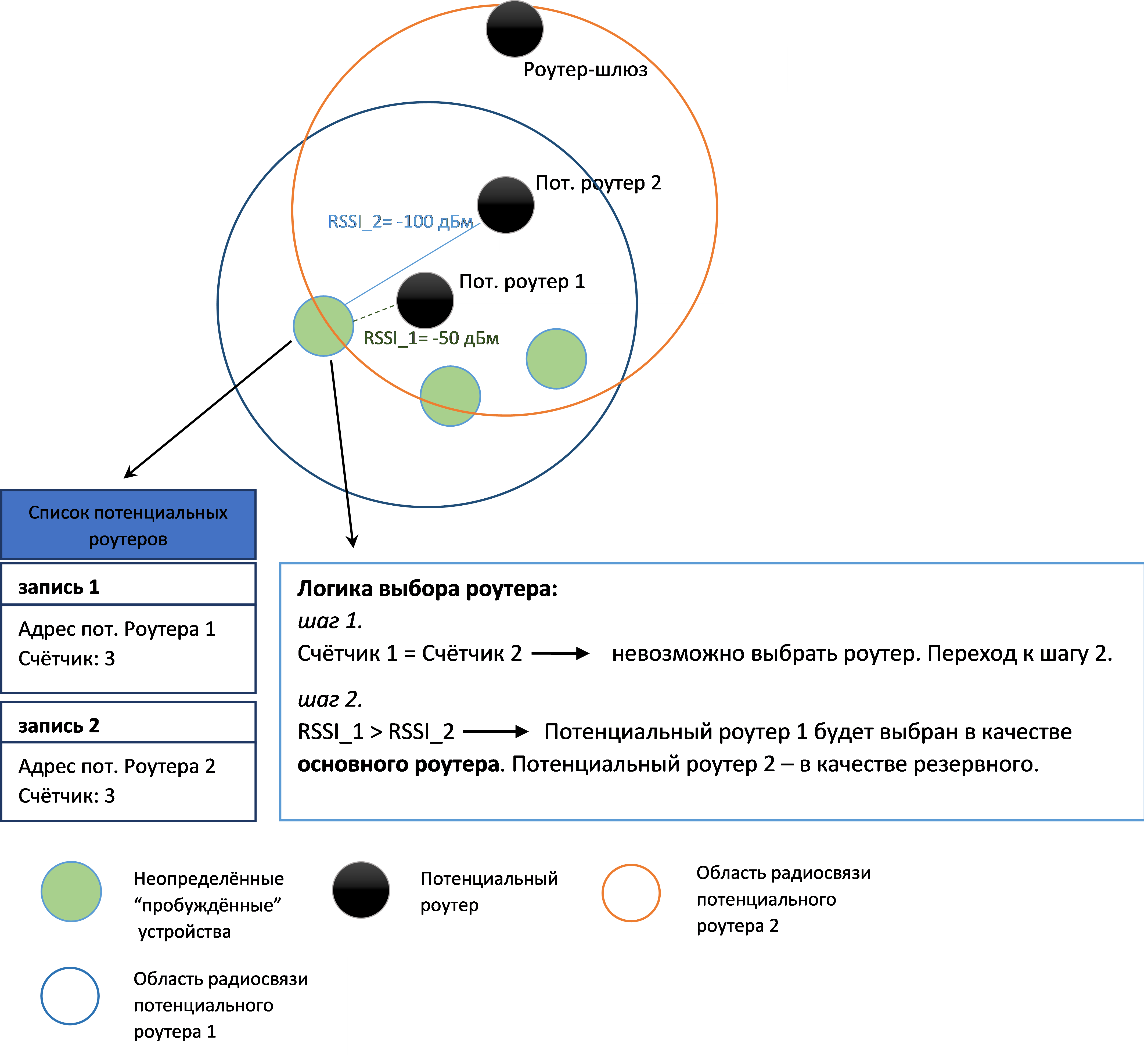


Рисунок 15. Выбор роутера на основании RSSI.

После того, как основной роутер определён, устройство приступает к выбору резервного роутера. Процедура выбора резервного роутера из числа потенциальных, производится аналогичным образом, что и выбор основного.

Определив для себя основной и резервный роутеры, устройство помещает их адреса в полезную нагрузку пакета 02 – “Я выбрал роутер” и отправляет этот пакет в эфир.

Данный пакет принимается и обрабатывается всеми потенциальными роутерами, находящимися в зоне радио досягаемости. Если в полезной нагрузке потенциальный роутер обнаружит свой адрес, то устройство, отправившее этот пакет, будет добавлено в списки подчинённых устройств роутера. Потенциальный роутер имеет два списка подчинённых устройств:

1. Список подчинённых устройств, которые выбрали данный потенциальный роутер в качестве основного роутера;
2. Список подчинённых устройств, которые выбрали данный потенциальный роутер в качестве резервного роутера.

пот. роутер 1

Список подчинённых устройств резерва

*Адр. осн. роутера*

*Адр. рез. роутера*

Список подчинённых устройств

Адрес подчинённого устройства 1

Адрес подчинённого устройства 2

Адрес подчинённого устройства N

…

Адрес подчинённого рез. устройства 1

Адрес подчинённого рез. устройства 2

Адрес подчинённого рез. устройства N

…

**Заголовок**

Пакет 02 – “Я выбрал роутер”

пот. роутер 2

Неопределённые “пробуждённые”

устройства

Потенциальный роутер

Отправка пакета в эфир

Приём и обработка пакета

Приём и отбрасывание пакета

Рисунок 16. Обработка пакета 02-"Я выбрал роутер", сформированный устройством, потенциальными роутерами путём добавления адреса устройства в свои таблицы.

В зависимости от того, в каком поле пакета 02 – “Я выбрал узел”, потенциальный роутер обнаружит свой адрес, в соответствующий список и будет помещён адрес устройства, отправившего пакет. В случае, если устройство при формировании пакета 02 – “Я выбрал узел” содержит только адрес одного потенциального роутера, то в поле полезной нагрузки пакета на месте адреса резервного роутера будет записан балласт (00 00 00 00).

После некоторого промежутка времени ожидания (устанавливается опционально) потенциальный роутер, формирует пакет 03 – “Я роутер”, в полезную нагрузку которого помещает адреса всех устройств, от которых он получил пакет 02 – “Я выбрал роутер” со своим адресом в соответствующем поле (*<адрес основного роутера>*). Данные для пакета берутся из “списка подчинённых устройств”. Сформированный пакет потенциальный роутер вещает в эфир. С его помощью устройства определяют, был ли доставлен их пакет 02 – “Я выбрал роутер” потенциальному роутеру-получателю.

В случае, если по каким-либо причинам пакет 02 – “Я выбрал роутер” не был доставлен до потенциального роутера, то устройство это обнаружит, обработав пакет 03 – “Я роутер” (в нём не будет его адреса). После этого будет предпринята повторная попытка отправки недоставленного пакета. По умолчанию число повторных отправок – 2. Через некоторое время потенциальный роутер повторно сформирует и отправит в эфир пакет 03 – “Я роутер”, на основании которого устройство сможет определить успешно ли был передан повторный пакет 02 – “Я выбрал роутер”.

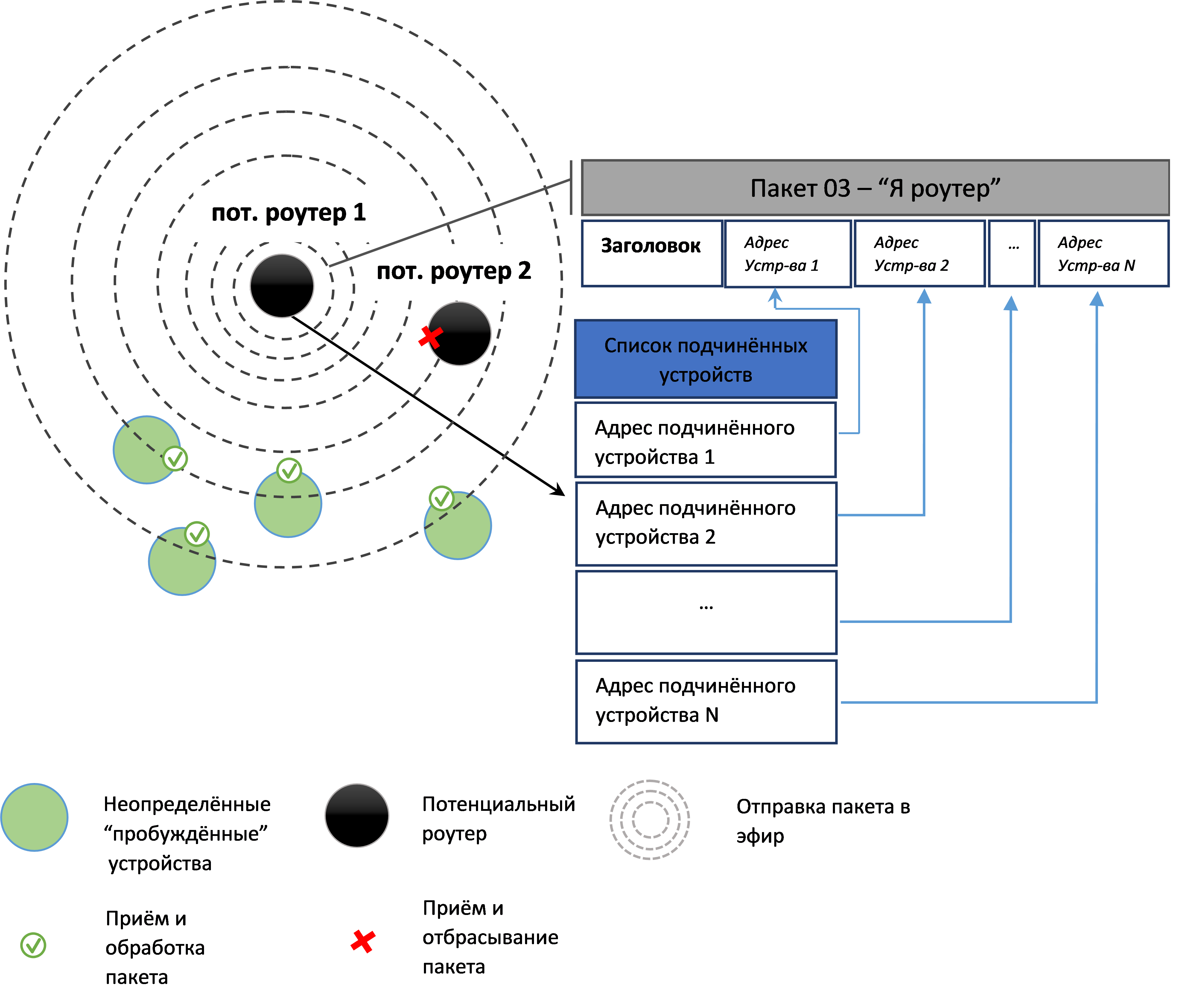


Рисунок 17. Формирование и отправка потенциальным роутером пакета 03-"Я роутер"

После того, как потенциальный роутер, отправил в эфир установленное количество раз пакет 03 – “Я роутер”, процесс построения сети для данного устройства завершается. Его роль сменяется с “Оконечного устройства” на “Роутер”.

С этого момента роутер готов собирать данные со своих подчинённых устройств, отправлять их роутеру шлюзу, а также продвигать пакеты по сети через себя от других роутеров.

Таким образом устройство, которое рассматривалось в качестве потенциального роутера завершает процесс построения сети и переходит в рабочий режим, ожидая пакеты 04-“Опрос устройств”, 05-“Ответ от устройства UNO”, 06-“Ответ от роутера MANY”, а также пакет 00-“Я потенциальный роутер” с номером сессии отличном от сохранённого в устройстве. Последний пакет даёт команду устройству заново инициировать процесс построения сети на своём участке. Все остальные пакеты игнорируются.

В то же время его подчинённые устройства (неопределённые устройства), переходят ко второй стадии построения сети на своём участке и объявляют себя потенциальными роутерами. Последовательность действий новых потенциальных роутеров идентична той, что выполнялась их роутерами ранее.

По результатам второй стадии потенциальные роутеры либо примут роль роутера, либо останутся оконечными устройствами. Необходимым условием установления роли роутера является наличие хотя бы одного устройства, которое объявит потенциальный роутер своим роутером. Если этого не происходит, то в данном направлении сеть замыкается, а потенциальный роутер утверждается в роли “оконечного устройства”.

## **Передача данных.**

На этапе передачи данных происходит периодический сбор данных, генерируемых устройствами сети. Последовательность операций в сети, при которой данные с устройств по запросу поступают на роутер-шлюз, называется “сеансом передачи данных”. Работа сети на этапе передачи данных в таком случае представляет собой сменяющие друг друга сеансы передачи данных.

1 этап – Построение сети

I стадия

II Стадия

2 этап – Передача данных

Сеанс N

Сеанс 1

Сеанс 2

…

Отправка запрашиваемых данных на ближайший узел сбора данных (роутер)

Агрегация данных

Продвижение данных (ретрансляция)

Накопление и обработка принятых данных

Инициация опроса сети

**Сеанс передачи данных**

Рисунок 18. Функциональная схема работы Протокола по этапам.

Этап передачи данных длится до тех пор, пока на устройство не перестанет подаваться питание, либо роутер шлюз не инициирует заново процедуру построения сети, отправив в сеть пакет 00-“Я потенциальный роутер”.

Процесс передачи данных начинается с отправки роутером-шлюзом пакета типа 04-“Опрос устройств”, полезная нагрузка которого содержит адреса всех подчинённых ему устройств. Данный

(Продолжение в разработке)

## **Механизм управления доступом к эфиру.**

В процессе функционирования сети узлы взаимодействуют друг с другом путём передачи сообщений через радиоэфир, который представляет собой разделяемую среду. В связи с этим возникает проблема совместного использования радиоэфира узлами сети, расположенных в одной зоне. Данная проблема решается на втором уровне семиуровневой модели взаимодействия открытых систем (L2 OSI). В радиомодулях, используемых в сенсорных сетях, применяются следующие протоколы доступа к среде: ALOHA, LBT, FDMA, TDMA. К сожалению, значительная часть радиомодулей производятся без внедрения этих механизмов. Вследствие чего Протокол должен предусматривать свой встроенных механизм распределения доступа между несколькими устройствами, использующих общую среду передачи.

Механизм управления доступом к среде предполагает разделение эфира на временные каналы. При необходимости передать данные через эфир устройство генерирует псевдослучайное целое число в диапазоне (1…N), где N - количество выделенных каналов. Полученное число будет номером временного канала, которое устройство займёт для передачи конкретного пакета. На основании номера канала рассчитывается время отправки.

Перед расчётом времени отправки пакета устройство определяет начало периода передачи, который выпал на текущий момент времени (T0 тек – см. Рисунок 19). После, на основании параметров канала (bстарт, bпер, bзад и Vпер) и выбранного случайным образом номера канала, вычисляется необходимая временная задержка от начала периода передачи до выбранного канала ΔTi (Рисунок 20). На основании ΔTi и T0 тек рассчитывается время отправки пакета (Рисунок 19). При этом может возникнуть ситуация, при которой рассчитанное время отправки может быть меньше текущего времени на устройстве в момент расчёта (Рисунок 19 – а). Чтобы данная ситуация не привела к ошибкам при передаче, в механизме предусмотрена проверка на предмет того, что рассчитанное время отправки ещё не наступило, т.е. время отправки больше, текущего ([T0 тек + ΔTi ] > tтек  - Рисунок 19 – б). Если данное условие выполняется, то рассчитанное время отправки записывается с запись очереди пакета на отправку, в противном случае производится пересчёт времени отправки на следующий период передачи. При этом вычисляется время начала следующего периода T0 тек+1 и на основании ранее вычисленной временной задержки канала относительно начала периода передачи определяется новое время отправки пакета (Рисунок 19 – а).

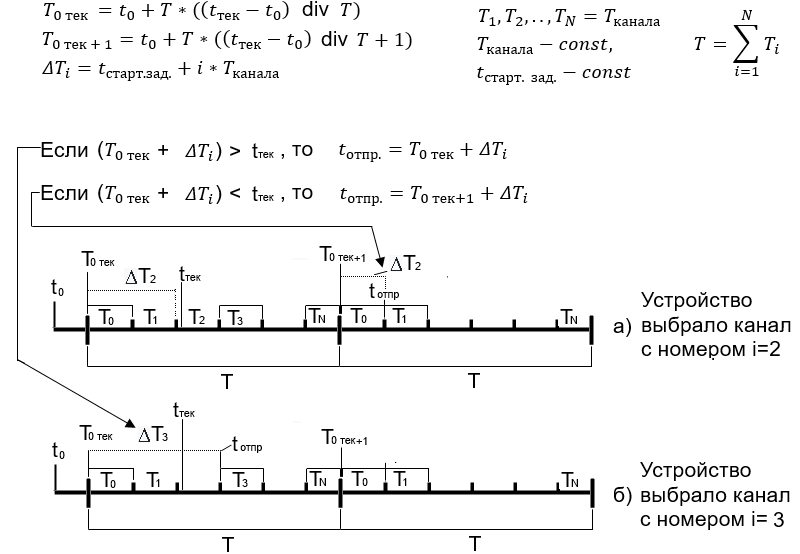
ч

Рисунок 19. Схема определения времени отправки при распределении доступа к радиоэфиру путём деления канала на временные каналы: а) случай, когда текущее время больше времени начала выбранного канала в текущем периоде б) случай, когда текущее время меньше времени начала временного канала в текущем периоде передачи.

**Пояснения** (Рисунок 19)



Рисунок 20.Распределение временных каналов Протокола. Вычисление временного смещения i-ого временного канала ΔTi.

**Пояснения** (Рисунок 20)

Множитель 1000 стоит для того, чтобы значение было в миллисекундах [мс]

# **Программная реализация узлов сети Протокола.**

## **Описание модели информационной системы, использующей Протокол.**

В подавляющей части случаев информационные системы, использующие в своём составе сенсорные сети, представляют собой системы мониторинга, системы управления или их объединение. Основной целью таких систем является сбор параметров окружающей среды или отдельных объектов, а также воздействие на них. Органами чувств и манипуляции выступают различного рода датчики, реле, исполнительные механизмы и роботы. Органы управления состоят из серверов и контроллеров. На серверах происходит обработка, накопление и хранение данных, а также управление системой в целом. Контроллеры представляют собой отдельную микропроцессорную плату, к которой непосредственно подключаются исполнительные механизмы, датчики, приёмопередатчики и другие вспомогательные устройства. Контролеры выполняют сбор, предварительную обработку и отправку серверу информации, поступающей с датчиков и исполнительных механизмов, а также выполняют задачи, которые поступают от соответствующего сервера посредством подключённых к нему устройств (включение/отключение сигнализации, поворот механизма, замыкание реле, регуляция давление теплоносителя на основании внешней температуры и др.).

Примерная архитектура информационных систем мониторинга/управления представлена ниже (*Рисунок 21*). С точки зрения архитектурной парадигмы наиболее подходящей является агентно-ориентированную архитектура, которая состоит из двух уровней: 1) Уровень управления 2) Агентский уровень.

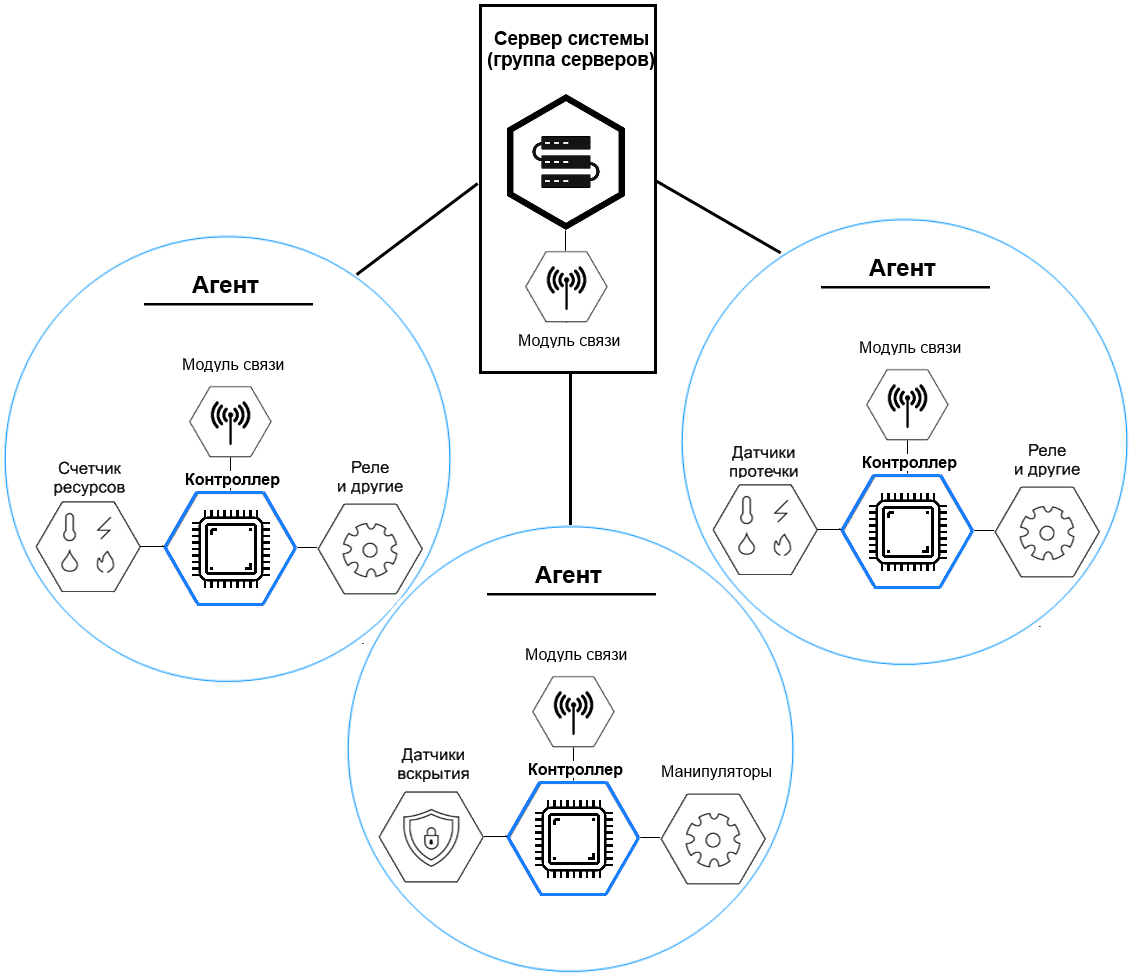


Рисунок 21. Архитектура информационных систем управления и мониторинга на базе сенсорных сетей.

Уровень управления представлен управляющим сервером (*Рисунок 21 – блок ‘Сервер-системы’*), задачами которого является сбор и обработка информации о наблюдаемых средах/объектах, а также манипуляции этими средами/объектами путём постановки соответствующих задач агентам.

Агентский уровень представлен конечными узлами-агентами (*Рисунок 21 – блоки ‘Агент’*), которые с помощью подключённых к ним устройств по команде, поступившей с сервера управления, либо исходя из встроенных инструкций, выполняет мониторинг/манипуляции с над объектами/средами.

Очевидно, что с точки зрения, системной архитектуры приложения, её топология будет представлять собой звезду (*Рисунок 22*). Сервер управления имеет непосредственную связь со своими Агентами, а их взаимодействие происходит путём приёма и передачи сообщений, формат которых определяется самой системой.

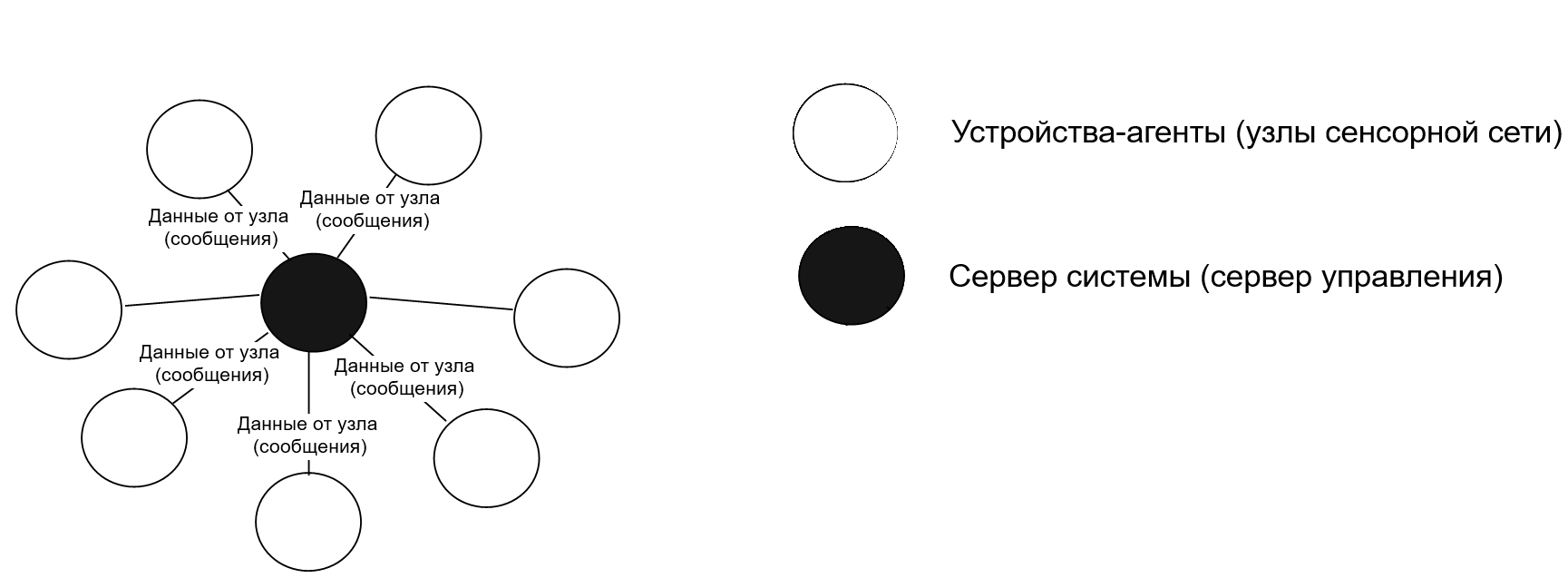


Рисунок 22. Логическая топология компонентов Информационной системы.

Физическая топология будет определяться топологией сети, которую информационная система будет использовать в качестве средства коммуникации между своими компонентами. Ниже приведена физическая топология информационной системы, построенной на базе сети Протокола.

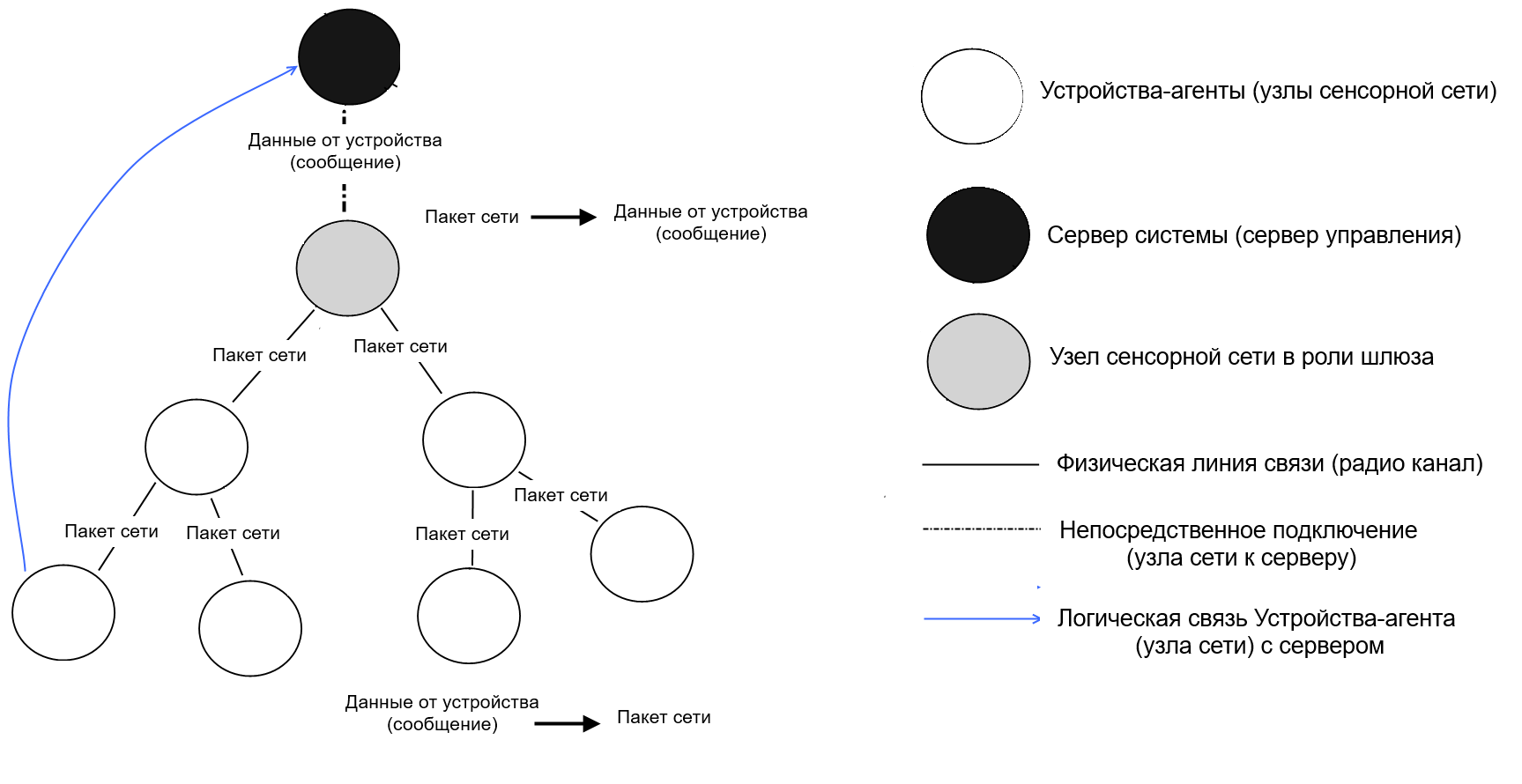


Рисунок 23. Физическая топология Информационной системы, использующая сети.

Не сложно заметить, что физическая топология значительно отличается от логической. Данное различие рамках одной системы указывает на независимость логических связей системы от физической организации связи устройств, из которых состоит система. Такой подход к проектированию даёт разработчикам возможность использовать в качестве сетевой инфраструктуры любую технологию связи, удовлетворяющей требования информационной систем. При этом нет необходимости сильно погружаться в устройство технологии связи для её внедрения в проект. Достаточно включить в проект необходимый программный модуль и пропускать все сообщения через него. Все промежуточные операции от поступления сообщения на вход программного модуля связи, его путешествие по сети в виде пакетов и выход через такой же программный модуль в виде исходного сообщения на стороне сервера, ложатся на плечи самого модуля. Для разработчика и узлов сети эти операции выглядят прозрачно, т.е. сообщения они получают, как если бы они напрямую были подключены к отправителю сообщения.

## **Используемый язык программирования**

Для разработки программной реализации Протокола было решено использовать в качестве языка программирования “чистый” Си (стандарт ISO/IEC 9899:2011). Данное решение было принято в целях обеспечения максимальной совместимости на различных аппаратных платформах.

## **Общая структура и функции программного модуля Протокола**

Сама реализация Протокола предполагается в виде отдельного программного модуля или библиотеки, которые можно будет встроить в любой проект, написанный на Языках Cи и C++. В зону ответственности программного модуля Протокола (далее Программный модуль) входит:

* преобразование байтовых массивов в структуры (парсинг пакетов);
* валидация пакетов на уровне структуры и логики;
* реакция устройства на внутренние и внешние события сети, построенной на базе Протокола);
* формирование новых пакетов для передачи в сеть;
* управлением доступом к среде;
* построение таблиц маршрутизации;
* подключение к сети протокола и обеспечение автоматического встраивания в сеть.

С точки зрения приложения Программный модуль будет выглядеть, как некоторая функция, которая принимает на вход либо принятый из сети пакет, либо данные, которые требуется передать на сервер. На выходе модуля приложение должно получить пакет на отправку в сеть, либо специальный пакет “пустышку”, получив который приложение не будет ничего отправлять в эфир (Рисунок 21). Следует отметить, что работа Программного модуля не предусматривает взаимодействие с радиоустройствами (напрямую или через отдельный соответствующий программный модуль).

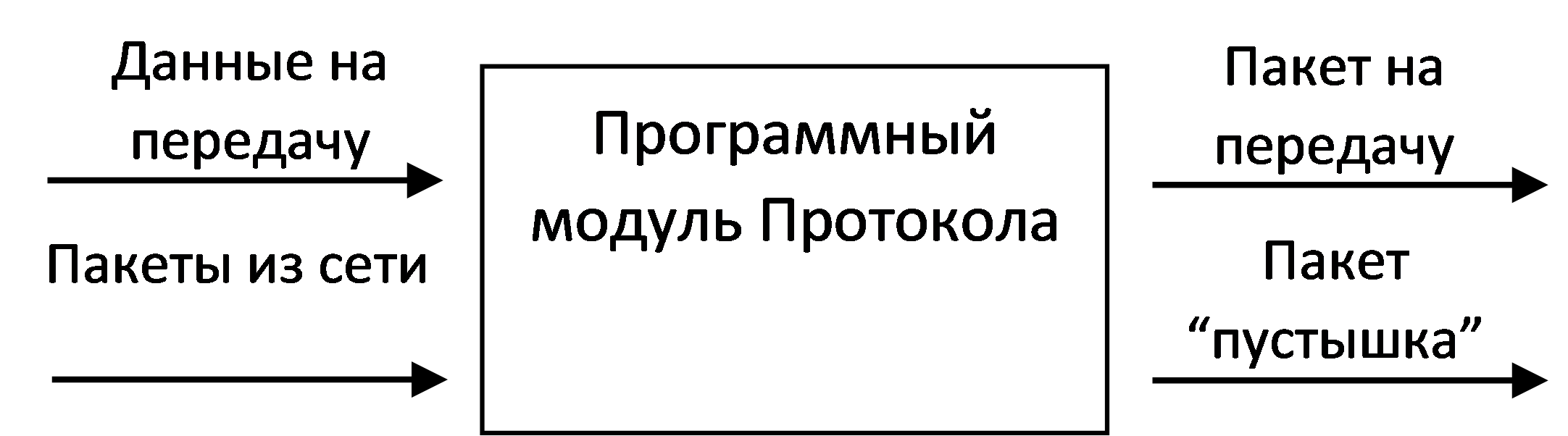


Рисунок 24. Входы и выходы основной функции программного модуля Протокола.

## **Роль и место программного модуля Протокола в составе общей программы узла.**

Управление и связь с приёмопередатчиками осуществляется приложением через отдельные специальные программные модули/библиотеки. Также их часто называют драйверами устройств. (*Рисунок 25 – ‘Модуль для взаимодействия с радиопередатчиком’*). Их задачей является общение с устройством на уровне команд, которые определены в логике работы самого устройства. Связь с такими устройствами обычно осуществляется через интерфейсы I2C, SPI или RS232. Также всё чаще появляются устройства, использующие USB.

Аналогичным образом устроена работа контроллера с остальными периферийными устройствами (различные датчики, счётчики, исполнительные механизмы, реле, дисплеи, динамики и др.). В зависимости от типа устройства взаимодействие контроллера с периферией происходит через интерфейсы I2C, SPI, RS232, USB для “умных устройств” (имеющие в своём составе свой микропроцессор), и подачей аналоговых или цифровых сигналов напрямую устройствам для остальных (динамики, реле, автоматы, простые датчики) (*Рисунок 25 – ‘Модуль работы с периферией’ и ‘Периферия’*). В рамках программной реализации это также обеспечивается специализированным программным модулем (драйвером) для работы с тем или иным устройством (*Рисунок 25* *– ‘Модуль работы с периферией’*).

Оставшиеся функции, такие как управление устройствами (запрос данных, подача команд устройствам) и обработка информации (приём, запись, преобразование, накопление, обновление и др.) происходит в основном цикле программы, который также именуется ‘основной логикой’ программы (*Рисунок 25 – ‘Основная логика’)*. Данные функции основная логика производит путём вызова специальных функций, которые могут быть как частью этой логики, так и входить в состав отдельных программных модулей, присоединённых к проекту. Алгоритмы, заложенные в основной логике, определяют порядок и условия вызова этих функций для выполнения задач, возникающих в ходе работы узла в рамках информационной системы (*Рисунок 25* – *Программная часть узла*).

Модуль для взаимодействия с сенсорной сетью является отдельным специальным модулем, который обеспечивает набор функций для работы с сенсорной сетью, построенной в соответствии с разрабатываемым Протоколом. Обобщённая структура модуля и перечень его функций перечислен выше *(*см. раздел *Общая структура и функции программного модуля Протокола).* По своей сути программный модуль Протокола можно представить как некий драйвер, подобно модулям работы с периферийными устройствами. В этом случае в качестве условного устройства выступает сенсорная сеть, с которой модуль Протокола взаимодействует посредством сообщений установленного формата в установленном Протоколом порядке. С точки зрения основной логики программы данный модуль обеспечивает доставку и приём пакетов между текущим устройством и сервером информационной системы. При этом за сам радиоканал модуль Протокола не использует. Все пакеты из сети он получает от основной логики Программной реализации узла при его вызове последним.

Таким образом программная реализация узла сенсорной сети в рамках информационной системы мониторинга и управления строится вокруг основной логики, которая на основании заложенных в ней алгоритмах осуществляет решение специфических для этого узла задач. Для взаимодействия со средой (съём информации о среде, воздействия на среду/объекты) основная логика вызывает функции из программного модуля для работы с устройствами периферии. Для взаимодействия с радиосредой используется модуль для работы с соответствующим устройством радиосвязи. Обеспечением связи с другими устройствами в рамках единой сети, на базе имеющегося радиоканала занимается модуль Протокола, который ведёт учёт как соседних устройств, так и устройств, с которыми связь возможна только через другие устройства. Помимо функций эксплуатации сети, модуль Протокола также и сам предоставляет услуги сети, узлом которой является. К ним относятся обязательные функции ретрансляции пакетов от других узлов сети через себя в сторону сервера, а также пакетов от сервера в сторону других узлов сети. Очевидно, что для функционирования сенсорной сети, на каждом устройстве этой сети должен работать программный модуль Протокола. Узел сети, располагающийся между сервером и сетью в соответствии с Протоколом должен исполнять роль Корневого роутера шлюза, который агрегирует на себе все пакеты из сети, извлекает из них сообщения и трансформирует их в формате, установленном приложением сервера. Кроме того, корневой узел-шлюз должен иметь средства коммуникации с сервером. Это может быть как прямое подключение, так и посредством иных инфокоммуникационных технологий (через локальную компьютерную сеть Ethernet, через глобальную сеть Интернет по TCP/UDP/HTTP и т.д.).

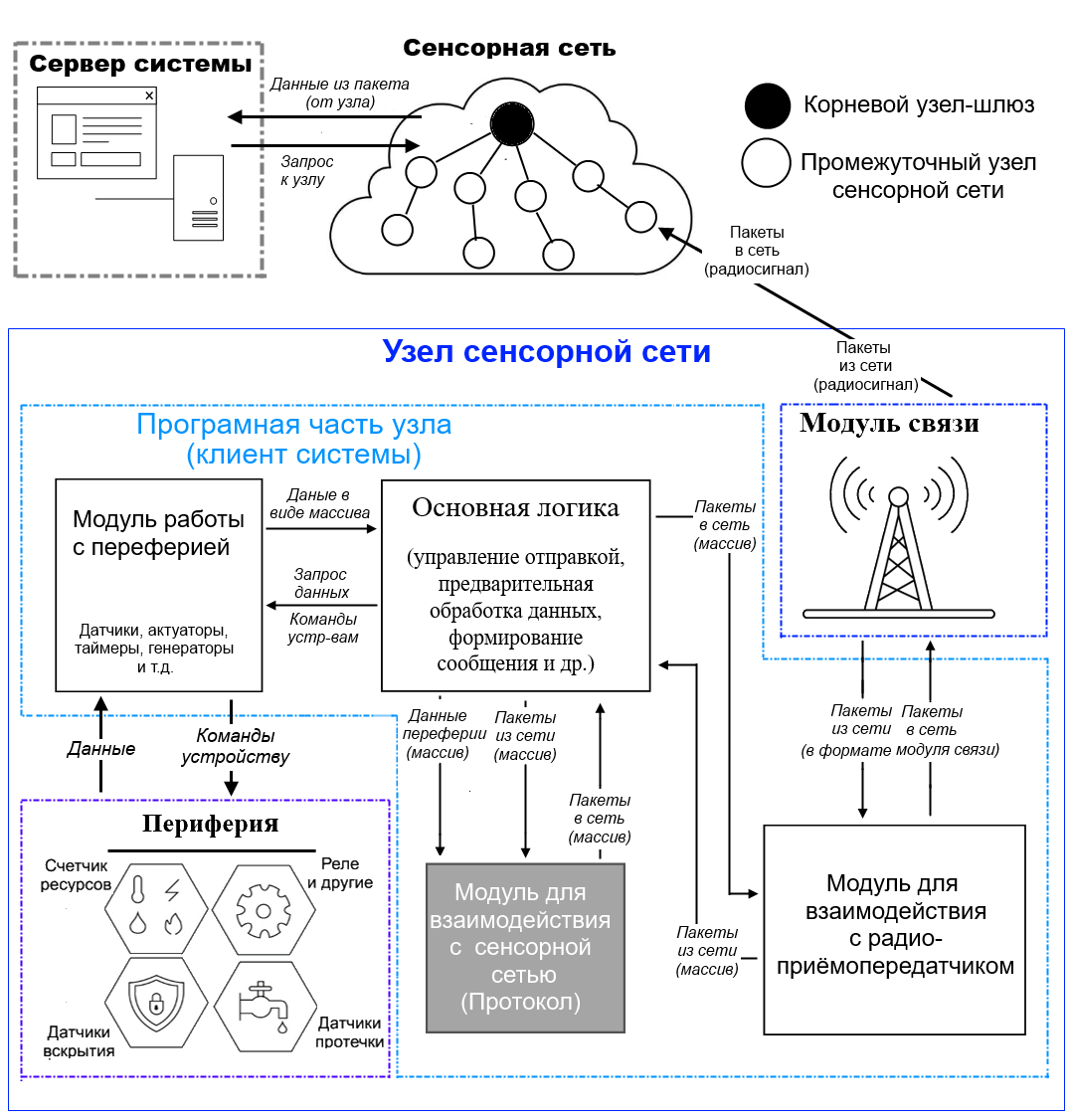


Рисунок 25. Общая схема информационной системы, построенной на базе сенсорных сетей.

## **Архитектура программного модуля Протокола.**

Работа программного модуля строится вокруг основной функции, которая носит название “Менеджер пакетов”. Именно её вызывает основная логика вышестоящей программы. На вход эта функция принимает следующие данные:

* Сообщение, которое необходимо отправить на сервер, либо пакет, принятый из сети в виде символьного/байтового массива;
* Указатель на область памяти, выделенную основной логикой под рабочие таблицы Протокола;
* Данные об уровне принятого сигнала при приёме последнего пакета (не обязательно).

Выходом данной функции, как было отмечено ранее в разделе “*Общая структура и функции программного модуля Протокола”,* является символьный/байтовый массив, который содержит пакет на отправку или пакет “пустышка”. Пакет “пустышка” служит сигналом для функции, вызвавшей “Менеджер пакетов”, что в эфир на данном этапе ничего отправлять в сеть ничего не требуется.

Для выполнения возложенных на программный модуль Протокола задач “Менеджер пакетов” поэтапно и в зависимости от ситуации вызывал свои вспомогательные функции. В соответствии с архитектурой программный модуль Протокола содержит следующие вспомогательные функции:

*Парсер заголовков входящих пакетов (ParseHeader)* – вспомогательная функция, извлекающая из байтового массива поля заголовка пакета, и помещает их в специальную структуру *Packet* (*Рисунок 26 – Парсер пакета*);

*Валидатор входящих пакетов (Validator)* – вспомогательная функция, выполняющая проверку входящего пакета на целостность (наличие символа начала и конца пакета, размер пакета и т.д.), а также на соответствие типа входящего пакета с разрешённым набором типов, установленных для текущего статуса, в котором пребывает устройство (*Рисунок 26 – Валидатор пакетов*)*;*

*Селектор обработчика (****switch****(pack.\_typepacket)) –* блок исполнительного кода функции *“Менеджер пакета”,* который определяет, какой *“Обработчик входящего пакета”* вызвать, исходя из типа пакета (*Рисунок 26 – Селектор обработчика*);

*Блок обработчиков входящих пакетов (packet\_Handler\_ xx) –* набор вспомогательных функций для обработки для обработки входящих пакетов, обновления сведений о сети на основе информации полученной из обработанного пакета (*Рисунок 26 – Обработчики пакетов*);

*Контроллер состояний узла сети/устройства (StatusController) –* вспомогательная функция, определяющая условия и порядок перехода устройства в то или иной состояние (*Рисунок 26 – Контроллер состояний*);

*Селектор фабрики (****switch****(ram->Status)) –* блокисполнительного кода функции *“Менеджер пакетов”*, который на основании текущего статуса устройства определяет, какую фабрику из “*Блока* *фабрик исходящих пакетов”* вызвать для формирования исходящего пакета. Если же переданному на вход селектора статуса нет соответствующей фабрики, то пакет не формируется;

*Блок фабрик исходящих пакетов (packet\_Factory\_xx)* – набор вспомогательных функций, отвечающих за формирование исходящих (ответных) пакетов, посредством которых устройство взаимодействует с сетью, работающей на базе Протокола. Каждая функция в результате своей работы возвращает пакет в виде байтового массива (*Рисунок 26 – Фабрики пакетов*);

*Менеджер очередей исходящих пакетов (QueneManager) –* вспомогательная функция, определяющая время отправки исходящих пакетов. Время отправки определяется задержкой, которая генерируется по правилам Протокола. Также в задачи данной функции входит определение пакета на отправку, стоящих в очереди, либо отказ от отправки пакета (*Рисунок 26* – *Менеджер пакетов).*

Для работы протокола в памяти создаются специальные поля, которые хранят различные данные, необходимые для работы Протокола. В рамках архитектуры совокупность всех этих полей носит название “Память устройства” (*Рисунок 26 – Память устройства*). Память устройства содержит следующие блоки полей:

*Таблица маршрутизации (****RouteUnit*** *pRouterList[]) –* списокзаписей маршрутизации, которые содержат всю необходимо информацию для формирования маршрутов на этапе построения сети, а также для выбора направления отправки пакета (*Рисунок 26 – Таблица маршрутизации*);

*Статусы устройства (****enum*** *STATE) –* данный блок полей в программном модуле Протокола представлен особым типом структуры – “перечисление”. Данное перечисление хранит коды всех состояний (стадий) устройства в рамках жизненного цикла Протокола (*Рисунок 26 – Статусы устройства*);

*Параметры устройства –* под параметрами понимается набор константных и переменных полей, которые устанавливают такие параметры как максимально возможное число подключённых устройств, размер таблицы маршрутизации, число максимально допустимых подчинённых устройств, максимальный размер пакета, величины задержек и др. (*Рисунок 26* *– Параметры устройства* );

*Буфер очереди пакетов на отправку (****qUnit*** *QUENE[]) –* список записей (*qUnit*), необходимых для организации очереди пакетов на отправку в эфир;

*Список квитирования –* список отправленных пакетов, для которых требуется подтверждение по правилам Протокола;

*Буфер агрегации пакетов –* список записей, содержащих проходящие пакеты для агрегации.

Помимо указанных полей и структур имеются и другие специфические структуры. Подробнее с ними можно ознакомиться как в комментариях листинга кода проекта, так и в приложении к данному документу.



Рисунок 26. Архитектура проекта программной реализации Протокола (программный модуль протокола).

## **Описание процесса выполнения программного модуля протокола.**

Работа основной функции протокола (Менеджер пакетов) предполагает, что её вызов основной логикой происходит циклично. Причём время одного цикла программы в как минимум в десятки раз выше скорости приёма пакета Следовательно при описании реализации для работы в рамках Протокола необходимо, чтобы после выполнения модуля протокола последующие функции, вызываемые в основном цикле программы не оказывали такой задержки, при которой принятый радио модулем пакеты могли бы быть не выгружены должным образом.

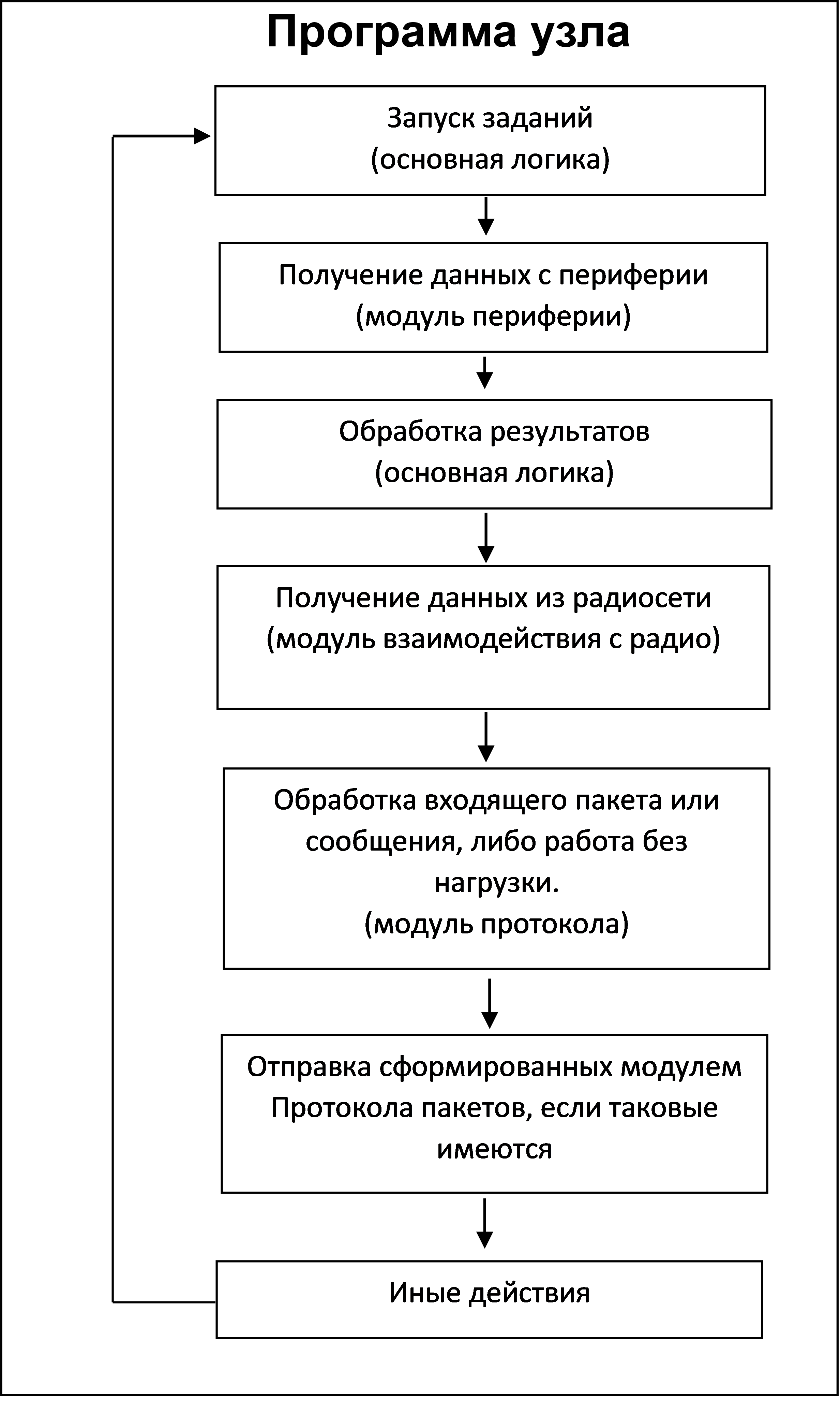


Рисунок 27. Примерная последовательность действий программы узла сети Протокола.

Работа программного модуля Протокола начинается с обработки массива, который передаётся при вызове его основной функции (*Менеджер пакетов*) в качестве аргумента вместе с другими входными данными. Обработка массива выполняется *Парсером пакета.* По результатам его работы модуль определяет, что именно поступило в виде массива, и заполняет соответствующие структуры (“Входящий пакет”,” Сообщение на сервер”, “Пустой массив”). Далее эстафету принимает вспомогательная функция *Валидатор пакета*. Она проверяет считанные из массива в структуру входящего пакета данные на предмет целостности, соответствия формату и статусу устройства. Если структура пуста, т.е. во входном массиве не содержался пакет, то функция немедленно возвращает управление вызвавшей её программе. Аналогичным образом *Валидатор пакета* поступает в случае, если обнаруживает, что принятый пакет не удовлетворяет условиям проверки. После проверки вызывается *Селектор обработчика*, который в зависимости от типа поступившего пакета вызывает соответствующую функцию обработчика пакета. Обработчик пакета исходя из типа обрабатываемого пакета производит изменение данных в соответствующих полях Протокола. Обновив поля, отражающие состояние сети