

Лабораторная работа №1 “Настройка базовой станции LoRa”

1 Цель работы

Целью работы является изучение особенностей и характеристик, а также настройка базовой станции LoRa для разворачивания сети LoRaWAN.

2 Краткие теоретические сведения

2.1 Технология модуляции LoRa

Технология модуляции LoRa (Long Range) представляет собой метод модуляции, который обеспечивает значительно большую дальность связи (зону покрытия), чем другие конкурирующие с ним способы. Метод основывается на технологии модуляции с расширенным спектром и вариацией линейной частотной модуляции (Chirp Spread Spectrum, CSS) с интегрированной прямой коррекцией ошибок (Forward Error Correction, FEC).

Технология LoRa значительно повышает чувствительность приемника и, аналогично другим методам модуляции с расширенным спектром, использует всю ширину полосы пропускания канала для передачи сигнала, что делает его устойчивым к канальным шумам и нечувствительным к смещениям, вызванным неточностями в настройке частот при использовании недорогих опорных кварцевых резонаторов.

Технология LoRa позволяет осуществлять демодуляцию сигналов с уровнями на 19,5 дБ ниже уровня шумов, при том что для правильной демодуляции большинству систем с частотной манипуляцией (Frequency Shift Keying, FSK) нужна мощность сигнала как минимум на 8-10 дБ выше уровня шума.

Модуляция LoRa определяет тот физический уровень (Physical

Layer, PHY, иногда его называют слой), который может быть использован с различными протоколами и в различных вариантах сетевой архитектуры, таких как mesh, звезда, точка-точка (point-to-point) и т. п.

2.2 MAC-протокол LoRaWAN

2.2.1 Общие сведения о LoRaWAN

Модуляция LoRa является, как уже говорилось выше, физическим уровнем, а LoRaWAN (Long Range Wide-Area Networks, LoRaWAN) это MAC-протокол для высокочастотных сетей с большим радиусом действия и низким собственным потреблением мощности, который организация LoRa Alliance стандартизировала для маломощных глобальных радиальных сетей (Low Power Wide Area Networks, LPWAN) типа звезда.

Протокол LoRaWAN оптимизирован для малобюджетных сенсоров с работой от батарей и включает в себя различные классы узлов, обеспечивая компромисс между скоростью доставки информации и временем работы устройств при использовании питания от аккумуляторов.

Протокол обеспечивает полную двустороннюю связь, а архитектура (посредством специальных методов шифрования) обеспечивает общую надежность и безопасность всей системы.

Архитектура LoRaWAN также была разработана с целью облегчить обнаружение мобильных объектов для отслеживания активов предприятий, что является одним из наиболее быстро растущих приложений на уровне Интернета вещей (Internet of Things, IoT).

Протокол LoRaWAN разрабатывается для использования в общенациональных сетях крупных операторов связи. С этой целью организация LoRa Alliance стандартизирует свой протокол LoRaWAN с учетом совместимости и взаимодействия со всеми основными глобальными операторами связи.

2.2.2 Возможная скорость передачи данных по протоколу LoRaWAN

Скорость передачи данных по протоколу LoRaWAN в системе LoRa лежит в диапазоне 0,3-11 кбит/с. Для Европы доступен один GFSK-канал (Gaussian Frequency-Shift Keying, GFSK) для передачи информации с потоком данных в 50 кбит/с.

Чтобы продлить срок службы аккумулятора в конечном устройстве и общую пропускную способность сети, сетевой сервер LoRaWAN управляет скоростью передачи данных и радиочастотным выходом каждого конечного устройства по отдельности.

Управление осуществляется с помощью алгоритма адаптивной скорости передачи данных (Adaptive Data Rate, ADR). Это имеет решающее значение для высокой производительности сети и позволяет осуществлять ее необходимую масштабируемость.

Сеть может быть развернута с минимальными инвестициями в ее инфраструктуру и с той ее емкостью, которая необходима для данного конкретного применения. Если развернуто много шлюзов, то технология ADR будет смещать скорость передачи данных в сторону повышения, что обеспечит масштабирование емкости сети в пределах от 6 до 8 раз.

2.3 Шлюзы LoRa

2.3.1 Общие сведения о шлюзах

Шлюзы LoRa предназначены для использования в радиальных звездообразных сетевых архитектурах большого радиуса действия, они используются в системе LoRaWAN.

Из-за свойств технологии LoRa эти шлюзы могут представлять собой многоканальные мультимодемные трансиверы, которые способны выполнять демодуляцию на нескольких каналах одновременно, и даже

одновременную демодуляцию множества сигналов на одном и том же канале.

Эти шлюзы используют иные радиочастотные компоненты, чем те, которые применяются в конечной точке для обеспечения высокой мощности излучения непосредственно радиосигнала.

Шлюзы служат в качестве интерфейса в виде прозрачного моста для передачи сообщений между конечными устройствами и центральным сервером сети.

Шлюзы подключаются к сетевому серверу через стандартные IP-соединения, а конечные устройства используют односкачковую беспроводную связь к одному или нескольким шлюзам.

Все конечные точки связи, как правило, являются двунаправленными, но они также поддерживают функционирование в режиме, обеспечивающем возможность осуществления группового обновления программного обеспечения по радиоканалу или передачу иных массовых сообщений, что позволяет сократить активное время на их передачу.

В зависимости от желаемой их канальной емкости и мест установки доступны разные версии шлюзов, они могут устанавливаться внутри помещений или на вышках.

2.3.2 Базовая станция Вега БС-2.2 от ООО «Вега-Абсолют»

Базовая станция Вега БС-2.2 предназначена для развертывания сети LoRaWAN® на частотах диапазона 863-870 МГц. Питание базовой станции и общение с сервером осуществляется через кабель Ethernet, кроме того общение с сервером может осуществляться через канал 3G. Встроенная антенна GPS, встроенный 3G модем. Рекомендуется использовать антенну 868-01-A10 мощностью 10 дБм. Базовая станция Вега БС-2.2 имеет встроенное ПО.

Более подробную информация о станции можно найти на сайте производителя (<https://iotvega.com/product/bs02-2>).



Рисунок 1 - Базовая станция Вега БС-2.2

3 Задание

Изучить базовую станцию Вега БС-2.2, произвести ее настройку, поставить и настроить сетевой сервер, а также установить приложение для администрирования данного сетевого сервера и произвести его настройку, от той же компании.

Этапы выполнения данной лабораторной работы:

- знакомство с базовой станцией Вега БС-2.2;
- настройка базовой станции Вега БС-2.2;
- сетевой сервер IoT Vega Server, знакомство и настройка;
- приложение для администрирования сетевого сервера IoT Vega Admin Tool, знакомство и настройка.

4 Ход работы

4.1 Знакомство с базовой станцией Вега БС-2.2

Открыв крышку базовой станции, вы увидите управляющую плату, прикрытую оргстеклом, оргстекло необходимо снять, чтобы в дальнейшем

было проще подключать базовую станцию к электрической сети и сети интернет, а также к персональному компьютеру.

На рисунке 2, можно увидеть размещение средств управления и индикации, а также входных и выходных интерфейсов на управляющей плате базовой станции.

Ниже описание каждого из компонентов управляющей платы базовой станции:

- 1: mini-USB порт для подключения к компьютеру;
- 2: USB хост для подключения внешних устройств;
- 3: Запуск интерфейса BS-Dashboard (опция);
- 4: Кнопка перезагрузки базовой станции;
- 5: Сервисные DIP-переключатели;
- 6: Группа индикаторов функционирования различных систем;
- 7: Разъем для micro SD-карты;
- 8: Разъем для Ethernet-кабеля;
- 9: Дополнительный разъем для питания (опция).

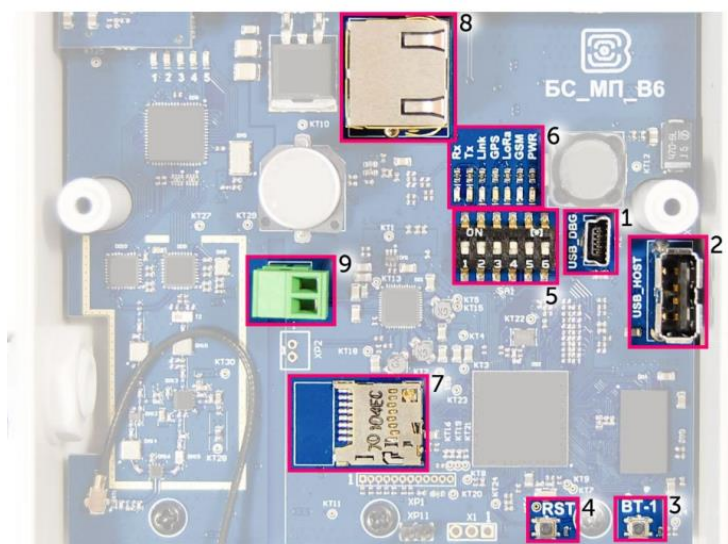


Рисунок 2 - Компоненты управляющей платы базовой станции

Чтобы приступить к работе с базовой станцией, необходимо подробнее разобраться с некоторыми ее компонентами.

Питание базовой станции от электрической сети, а также подключение ее к сети интернет осуществляется при помощи 8-ми жильного сетевого кабеля (витая пара), который может быть обжат по стандартам T568A и T568B, подключаемого в POE-адаптер, идущий в комплекте со станцией, и Ethernet разъем на управляющей плате, соответственно.

За средства управления на плате базовой станции отвечают соответствующие кнопки и переключатели.

Так, на ней располагаются две кнопки, одна из которых отвечает за дальнейшие разработки, а именно запуск части программного обеспечения интерфейса - серверного API «BS-Dashboard», которое отвечает за передачу информации о текущих настройках и получение новых настроек для станции, а также отправки информации об устройстве, по умолчанию API «BS-Dashboard» доступно на порту 3001, 3 кнопка на рисунке 2, другая же за мгновенную перезагрузку базовой станции, 4 кнопка на рисунке 2.

Также на ней имеются сервисные DIP-переключатели, под номером 5 на рисунке 2, функция которых заключается в переключение способа загрузки образа прошивки базовой станции:

- с внутренней памяти;
- с SD-карты;
- через mini-USB с персонального компьютера, однако данный способ используется только в условиях сервиса.

На рисунке 3, указано рабочее положение переключателей, должны быть включены только 3, 4 и 6.

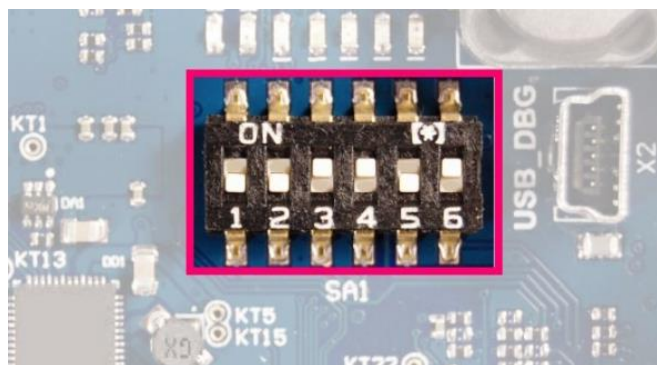


Рисунок 3 - Рабочее положение переключателей

Кроме того на плате расположены светодиодные индикаторы, каждый из которых отвечает за отображение функционирования той или иной системы:

- питание (включено/выключено);
- видимость спутников GPS;
- GSM-модем (включен/выключен);
- функционирование программы обработки сигналов LoRa (Packet forwarder запущен/не запущен);
- наличие активности по Ethernet;
- обмен данными по mini-USB порту.

На рисунке 4, описаны все состояния индикатор.

Индикатор	Цвет	Значение
Rx	Зелёный	<i>Вспыхивает</i> – обмен данными по порту USB_DBG
Tx	Красный	
Link	Зелёный	<i>Вспыхивает</i> – активность по Ethernet <i>Не горит</i> – нет данных от GPS-приёмника
GPS ²	Синий	<i>Вспыхивает</i> – есть данные, но они не валидные и не могут использоваться Packet forwarder
		<i>Горит</i> – местоположение определено
LoRa	Жёлтый	<i>Горит</i> – приложение Packet forwarder запущено
		<i>Не горит</i> – приложение Packet forwarder остановлено
GSM	Зелёный	<i>Горит</i> – GSM-модем включён
		<i>Не горит</i> – GSM-модем отключён
PWR	Красный	<i>Горит</i> – питание базовой станции подключено
		<i>Не горит</i> – питание базовой станции отсутствует

Рисунок 4 - Светодиодные индикаторы

Для улучшения качества получаемого сигнала на внешней стороне

корпуса базовой станции расположен соответствующий разъем для подключения антенны LoRa:

- SMA-разъем;
- N-коннектор.

На рисунке 5, показаны данные разъемы.



Рисунок 5 - Разъемы для подключения антенны LoRa

4.2 Настройка базовой станции Вега БС-2.2

Чтобы начать настройку базовой станции, необходимо подключить ее к персональному компьютеру, для этого есть два способа:

- по последовательному порту;
- по протоколу SSH.

Для этого можно скачать любую свободно распространяемую терминальную программу, например, PuTTY. Ниже будет использована именно данная программа.

В начале, подключение осуществляется при помощи первого способа, так как нам еще не известен ее IP-адрес, он необходим для второго способа.

Для этого, станция соединяется с компьютером кабелем через разъемом mini-USB на ее плате, далее необходимо скачать и установить драйвер для MCP2200 - виртуальный COM-порт, чтобы она могла верно инициализироваться на компьютере.

Увидеть полученный результат можно в диспетчере устройств на компьютере, рисунок 6.

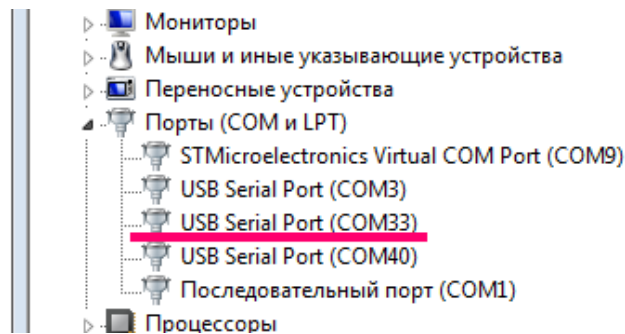


Рисунок 6 - Инициализация базовой станции на персональном компьютере

После удачной инициализации станции на компьютере, необходимо открыть PuTTY, в открывшемся окне, выбрать раздел Session, который отвечает за настройки сессионного подключения, в пунктах Serial line и Speed, а также Connection type, установите следующие параметры, как на рисунке 7, однако у вас может отличаться виртуальный COM-порт. Далее нажмите Open.

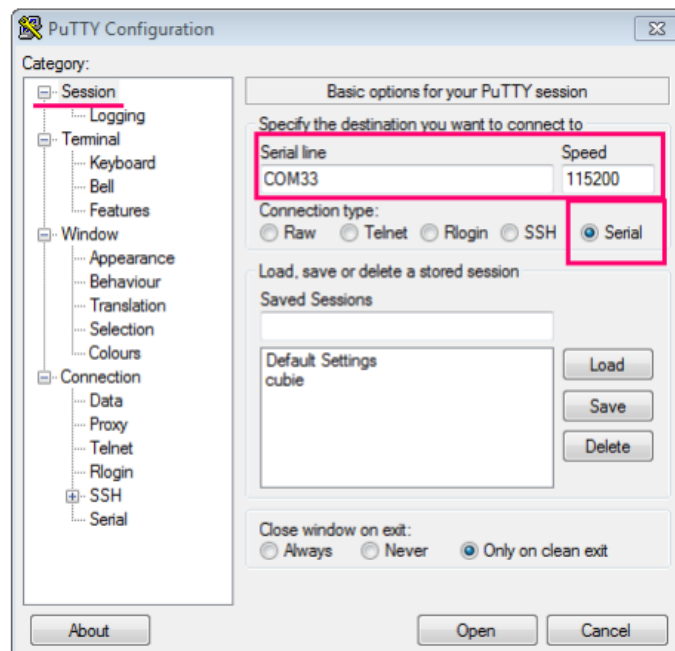


Рисунок 7 - Настройка сессионного подключения по последовательному порту

В открывшемся терминале, необходимо ввести логин и пароль для входа в операционную систему базовой станции - Linux v3.4, по умолчанию:

- login: root;
- password: temppwd.

Затем, чтобы узнать ее IP-адрес, надо ввести команду `ifconfig`, чтобы осуществить настройку при помощи веб-интерфейса, рисунок 8.

```
root@am335x-evm:~/bs-dashboard/manager# ifconfig
eth0      Link encap:Ethernet  HWaddr 34:03:DE:7B:72:80
          inet addr:192.168.1.228  Bcast:192.168.1.255  Mask:255.255.254.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1478151 errors:0 dropped:614 overruns:0 frame:0
          TX packets:103187 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:109611064 (104.5 MiB)  TX bytes:23971656 (22.8 MiB)
          Interrupt:56

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

root@am335x-evm:~/bs-dashboard/manager#
```

Рисунок 8 - Команда `ifconfig`

Данный IP-адрес может быть использован при подключении станции к компьютеру вторым способом, описанным выше. Параметры сессионного подключения показаны на рисунке 9.

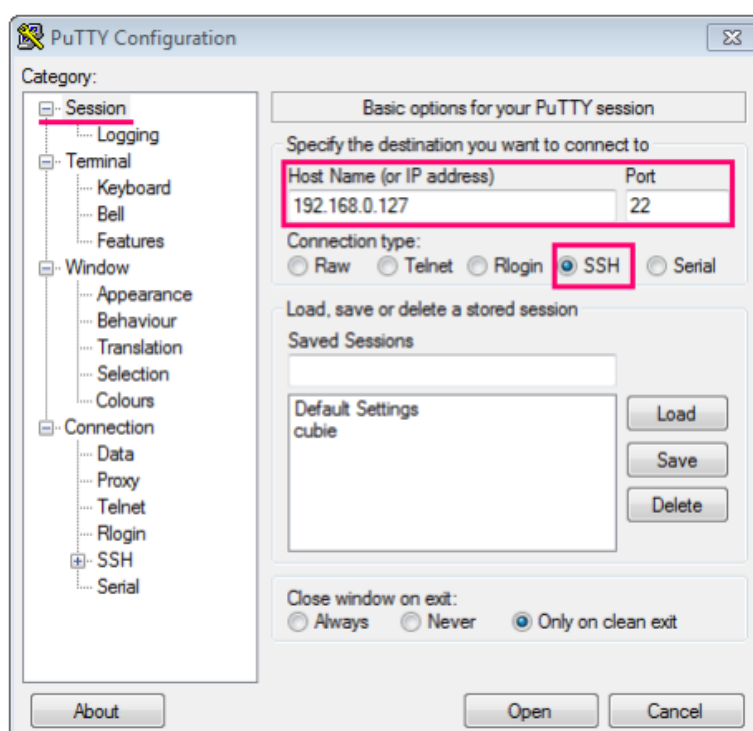


Рисунок 9 - Настройка сессионного подключения по протоколу SSH

Имея IP-адрес станции, можно приступить к ее настройке при помощи клиентского браузерного приложения, работающее с серверным API «BSDashboard», которое предназначено для визуального отображения данных, валидации изменений и отправки изменённых настроек для сохранения на устройстве. Клиентское браузерное приложение доступно на порту 80.

Для доступа к нему, необходимо открыть любой веб-браузер, в окне поиска ввести полученный IP-адрес, по умолчанию, логин и пароль, как при входе в операционную систему станции.

На рисунке 10, представлено окно авторизации в базовую станцию при помощи данного приложения.

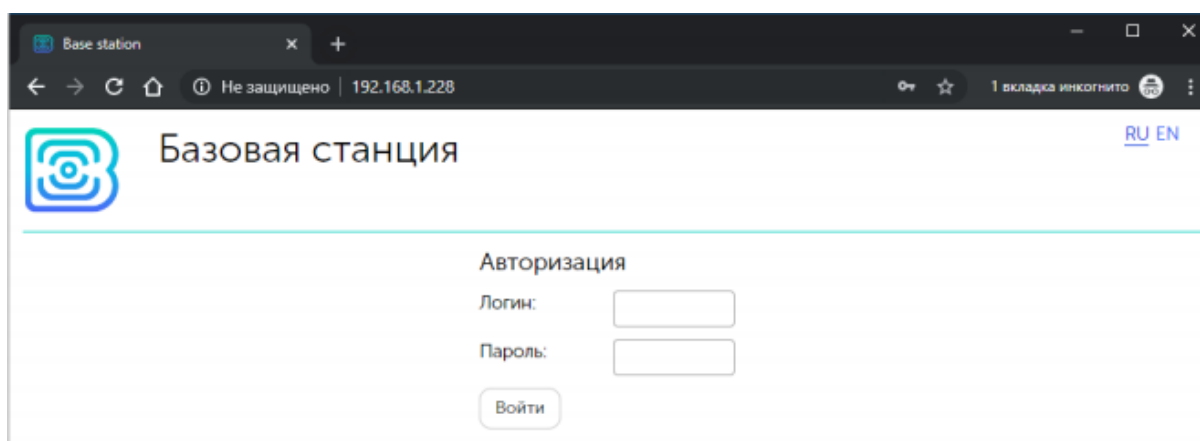


Рисунок 10 - Авторизация в базовую станцию при помощи клиентского браузерного приложения

Первоначально, откроется страница с настройками подключения к серверу LoRaWAN, тут можно прописать будущий IP-адрес сервера, а также выставить нижний и верхний порты, для дальнейшего подключения конечных устройств, также можно увидеть ID нашего шлюза, рисунок 11.

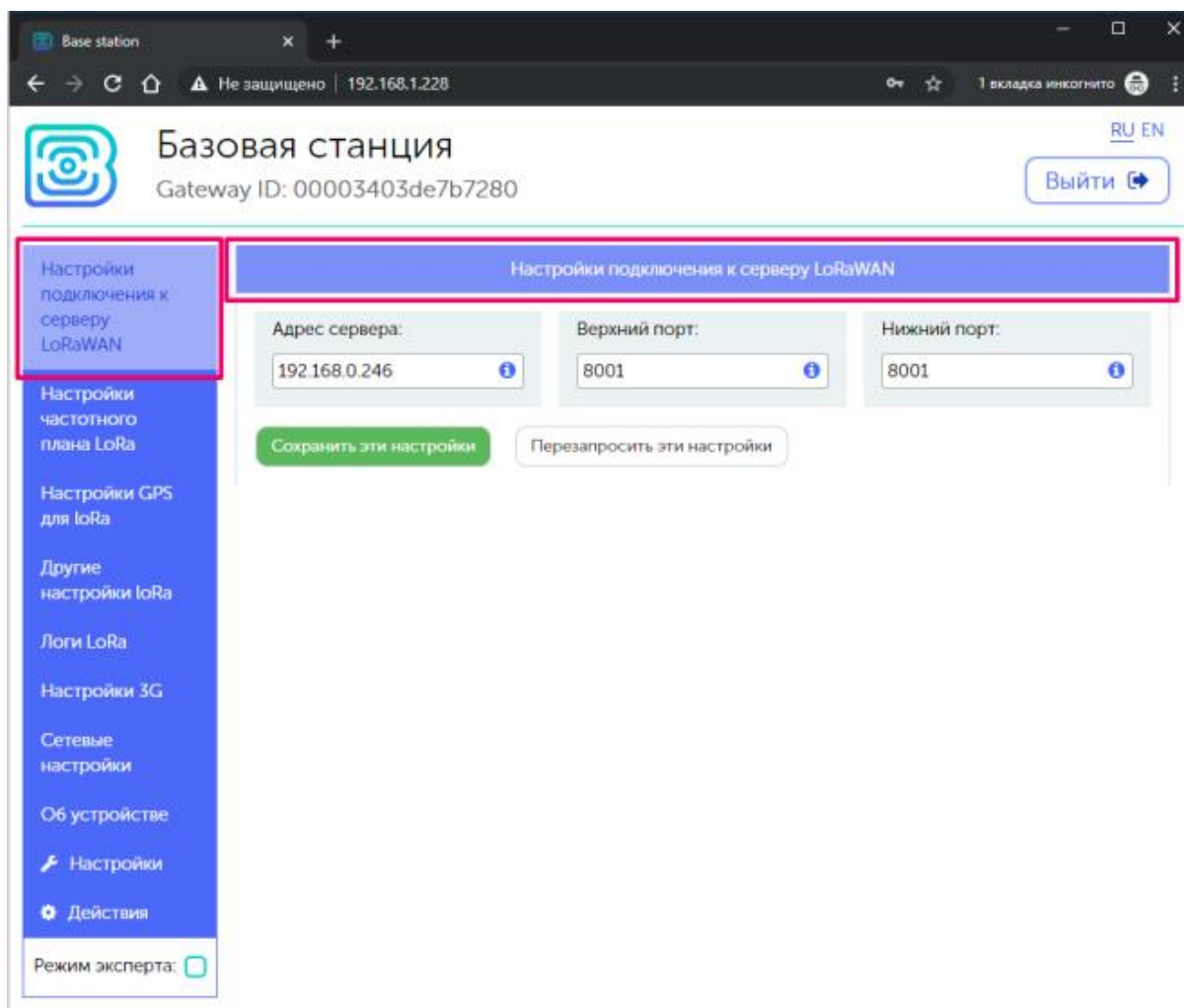


Рисунок 11 - Настройка подключения к серверу LoRaWAN

Ниже будут расписаны некоторые из разделов веб-интерфейса, которые понадобятся для дальнейшей работы:

- настройки частотного плана LoRa: по умолчанию, стоит EU868, он и будет использоваться в дальнейшем, также есть RU868, KZ868 и возможность самим задать данный план;
- настройки GPS для LoRa: здесь указывается точное местоположение, по координатам, базовой станции;
- логи LoRa: в данном разделе отображаются события, которые происходят при работе шлюза, можно выбрать отображение всевозможных событий;

- настройки: можно изменить пароль для входа в веб-интерфейс, а также произвести настройку менеджера, рисунок 12.

Рисунок 12 - Настройки

Остальные разделы веб-интерфейса необходимо изучить и настроить самостоятельно, по мере надобности.

После выполнения данных действий, настройка базовой станции прекращается.

4.3 Сетевой сервер IoT Vega Server, знакомство и настройка

Сетевой сервер IOT Vega Server это инструмент для организации сетей стандарта LoRaWAN® любого масштаба.

Предназначен для управления опорной сетью базовых станций, работающих под управлением ПО Packet forwarder от компании Semtech, приема данных с конечных устройств и передачи их внешним

приложениям, а также передачи данных от внешних приложений на LoRaWAN® устройства.

Более подробную информацию о данном сервере изучите самостоятельно, на сайте производителя.

В начале, необходимо скачать IoT Vega Server v1.2.1 для Windows с официального сайта производителя.

Установка не требуется, в архиве уже лежат готовые файлы сервера. Распакуйте содержимое архива в любое удобное место на персональном компьютере.

После этого, необходимо перейти в папку с сервером, далее надо установить обе библиотеки MSVC C++ 2013, лежащие в папке с тем же названием, дабы сервер корректно работал. После этого можно начинать работу с сервером.

Первым делом, производится настройка сетевого сервера, для этого, в папке IoT Vega Server, надо открыть файл с названием settings.conf с помощью любого текстового редактора, например, блокнот.

Содержимое файла приведено ниже. Красным выделены строки, в которых следует произвести изменения, синим отображены комментарии файла, зеленым отображены комментарии от производителя.

```
# Host connection settings
[host]
# IP-address for UDP connection (gateway connection)
ip=127.0.0.1 – здесь нужно указать IP-адрес компьютера, на котором
будет расположен сервер
# Port for UDP connection (gateway connection)
udpPort=8001 – этот порт используется для подключения базовых
станций к серверу
# Port for TCP (WebSocket) connection
tcpPort=8002 – этот порт используется внешними приложениями, он
пригодится при подключении к серверу через IOT Vega AdminTool
# "path" part of webSocket address
webSocketPath=/
# Flag of using SSL encryption for WebSocket
```

```
useSSL=0
# SSL certificate filename (certificate must be in server's directory)
certFileName=cert.crt
# SSL key filename (key must be in server's directory)
keyFileName=key.key

# LoRaWAN network settings
[lorawan]
# LoRaWAN network identifier (should be random between 1 and 127)
networkID=1 — идентификатор сети следует менять в том случае,
если поблизости организовано более одной сети, в остальных случаях в
этом нет необходимости
# Flag for using Plug-and-Play gateways function.
# If this value is 1, server would automatically append all gateways which
connected to one
usePnPGateway=1

# Super user options
[root] в данном разделе нужно задать логин и пароль для
суперпользователя, эти данные будут использоваться при обращении к
серверу через приложение IOT Vega AdminTool
# Login for super user
root=root
# Password for super user (recommendation: change this password to
your own)
password=123

# Console settings (volume of debug information)
[console]
# Maximum level of console messages that will be shown (levels of
messages represented below)
maxMsgLevel=20
# Maximum level of console messages that will be saved into LOG file
(levels of messages represented below)
maxLogMsgLevel=0 при первоначальной работе с сетью лучше
изменить это значение на 20, чтобы в лог-файл выводилось как можно
больше информации для дальнейших обращений в службу поддержки;
когда сеть стабильно работает и количество устройств возрастает
многократно, есть смысл понизить уровень информативности лог-файла
для более легкого поиска по файлу
# Console message levels:
# errors = 0
# uplink = 1
```



```
# downlink = 2
# warning = 3
# info = 4
# debug = 20
```

External DataBase settings

[external_db] в данном разделе следует производить изменения только в том случае, если вы планируете настраивать работу с внешней базой данных; по умолчанию внешняя база данных отключена, а сервер работает с собственной базой данных

Flag of using external DB

```
useExternalDb=0
```

Type of external DB. Supported only next types:

```
# MYSQL
```

```
# SQLITE
```

```
typeExternalDb=MYSQL
```

Name of external DB (schema's name for MYSQL)

```
nameExternalDb=server
```

IP and port of DB's server ("localhost" is supported)

```
ipExternalDb=127.0.0.1
```

```
portExternalDb=5505
```

User login and password (user should have maximum level of privileges)

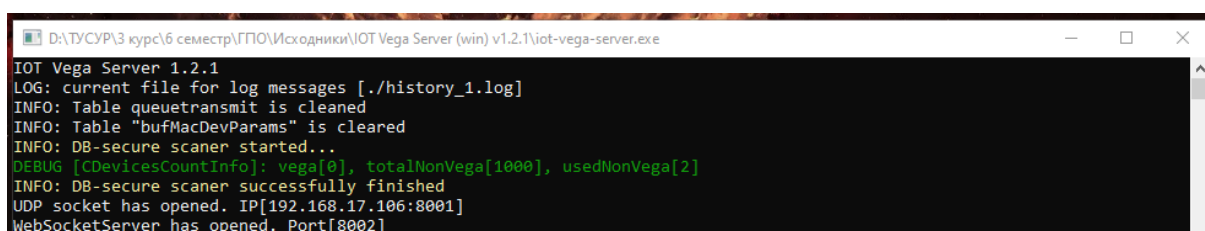
```
userExternalDb=admin
```

```
passwordExternalDb=admin
```

Более подробное описание всех параметров данного файла изучите самостоятельно, на сайте производителя.

После изменения некоторых параметров конфигурационного файла сервера, следует его сохранить и закрыть.

После этого запустите исполняемый файл сервера `iot-vega-server.exe`, в открывшемся консольном приложении будет отображаться вся информация о работе сети LoRaWAN, рисунок 13.



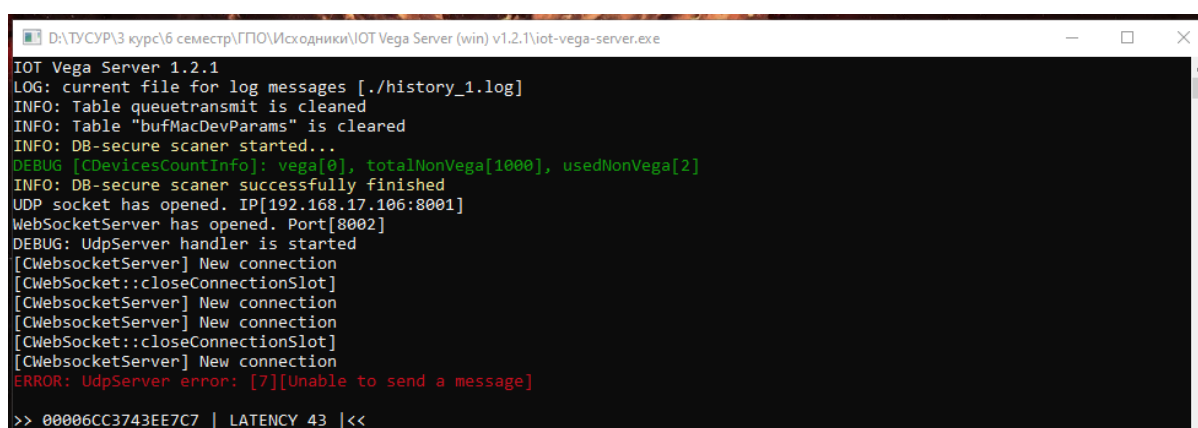
```
D:\TUCUP\3 курс\6 семестр\ГПО\Исходники\IOT Vega Server (win) v1.2.1\iot-vega-server.exe
IOT Vega Server 1.2.1
LOG: current file for log messages [./history_1.log]
INFO: Table queueTransmit is cleaned
INFO: Table "bufMacDevParams" is cleared
INFO: DB-secure scanner started...
DEBUG [CDevicesCountInfo]: vega[0], totalNonVega[1000], usedNonVega[2]
INFO: DB-secure scanner successfully finished
UDP socket has opened. IP[192.168.17.106:8001]
WebSocketServer has opened. Port[8002]
```

Рисунок 13 - Сетевой сервер IoT Vega Server v1.2.1

О корректной работе сервера говорят строки UDP socket has opened и WebSocketServer has opened, а также отсутствие каких-либо сообщений об ошибках.

Закрывать сервер не нужно, он необходим для работы сети.

Так как базовая станция уже была подключена к персональному компьютеру и настроена, она должна подключиться к сетевому серверу автоматически, рисунок 14 - последняя строка в консоли, в ином случае следует ее перезапустить, см. подраздел 4.2.

The image shows a Windows command prompt window titled "D:\ТУСУР\3 курс\6 семестр\ГПО\Исходники\IoT Vega Server (win) v1.2.1\iot-vega-server.exe". The console output displays the following messages:

```
IOT Vega Server 1.2.1
LOG: current file for log messages [./history_1.log]
INFO: Table queueTransmit is cleaned
INFO: Table "bufMacDevParams" is cleaned
INFO: DB-secure scanner started...
DEBUG [CDevicesCountInfo]: vega[0], totalNonVega[1000], usedNonVega[2]
INFO: DB-secure scanner successfully finished
UDP socket has opened. IP[192.168.17.106:8001]
WebSocketServer has opened. Port[8002]
DEBUG: UdpServer handler is started
[CWebSocketServer] New connection
[CWebSocket::closeConnectionSlot]
[CWebSocketServer] New connection
[CWebSocketServer] New connection
[CWebSocket::closeConnectionSlot]
[CWebSocketServer] New connection
ERROR: UdpServer error: [7][Unable to send a message]
>> 00006CC3743EE7C7 | LATENCY 43 |<<
```

Рисунок 14 - Подключение базовой станции Вега БС-2.2 к сетевому серверу IoT Vega Server

На данном этапе, работа с сетевым сервером заканчивается.

4.4 Приложение для администрирования сетевого сервера IoT Vega Admin Tool, знакомство и настройка

Для управления сервером есть удобное приложение IOT Vega Admin Tool с простым дружественным интерфейсом.

Admin Tool открывает перед администратором сервера широкие возможности по управлению сетью LoRaWAN®.

С Admin Tool можно добавлять в сеть новые оконечные устройства LoRaWAN®, просматривать карту сети, контролировать базовые станции, а также управлять правами пользователей.

Более подробную информацию о данном приложении изучите самостоятельно, на сайте производителя.

Первым делом, необходимо скачать приложение Admin Tool v1.1.5, с официального сайта производителя.

Установка не требуется, так как скачанный файл является архивом, в котором уже лежат готовые файлы приложения. Распакуйте содержимое архива в любое удобное место на персональном компьютере.

Перейдите в разархивированную папку с приложением, необходимо произвести его настройку, для этого надо открыть файл config.js с помощью любого текстового редактора, например, блокнот.

Так, в открывшемся окне, имеется всего две строчки, одну из которых надо изменить - обозначена красным цветом, она отвечает за IP-адрес сервера и TCP-порт подключения, которые были указаны в файле settings.conf, при настройке сетевого сервера IoT Vega Server:

```
const address_ws = 'ws://127.0.0.1:8002';  
const demo_user = false;
```

После изменения строки, файл следует сохранить и закрыть.

Далее надо запустить файл index.html с помощью любого браузера. Откроется окно авторизации в IoT Vega Server, где необходимо ввести логин и пароль, указанные в файле settings.conf, при настройке сервера.

Также на этой странице можно убедиться в том, что приложение успешно установило соединение с сервером, для этого надо нажать на шестеренку, появится строка с адресом и портом сервера, указанными в файле настроек config.js, а также значок подключения к серверу - зеленый значок означает успешное подключение, рисунок 15.

Authorization IOT Vega Server

Login

Login

Password

Password

WebSocket address

ws://192.168.0.77:8002

Sign in

Рисунок 15 - Авторизация в IoT Vega Server

Введя логин и пароль, а также убедившись в удачном подключении приложения в серверу, можно нажать кнопку Sign in, после этого приложение будет запущено.

По итогу будет открыта вкладка Home, рисунок 16, на которой отображается карта с местоположением станции и радиусом покрытия сети, также на ней имеется статус сервера и статистика приложения.

Home Devices Gateways Users Exit

BE2A ASCO ADT

Server status

Time	15.04.2021 04:50:32
Time zone	ТОМСК (зима)
Version	1.2.1 [WIN]
Vega devices	0
Third-party devices	2
Third-party devices available	998

Application statistics

Gateways	1
Devices	2
Users	1

Рисунок 16 - Вкладка Home

На вкладке Gateways будет отображаться подключенная к серверу станция, рисунок 17, где имеется имя, ID, статус и задержка данной станции.

		CONNECTED GATEWAYS		
Name	Gateway ID	Active	Latency	
PnP_GW_00006CC3743EE7C7	00006CC3743EE7C7		1	

Рисунок 17 - Подключенная базовая станция Вега БС-2.2 к сетевому серверу IoT Vega Server

Также на этой вкладке можно редактировать уже имеющиеся шлюзы, так и добавлять новые, рисунок 18.

Gateway settings
×

Main settings

Name

PnP_GW_00006CC3743EE7C7

Gateway ID

00006CC3743EE7C7

TX channel

0

Transmit power

14

RX only

Companion Gateway

☐

COMPANION GATEWAY ID

Comment

Comment

Location

Latitude

56.45378

Longitude

84.97771

Altitude

26

Close

Save

Рисунок 18 - Настройки шлюза

Остальные вкладки данного приложения не рассматриваются в данной лабораторной работе, изучите их самостоятельно.

Лабораторная работа №2 “Настройка конечных устройств LoRa”

1 Цель работы

Целью работы является изучение особенностей и настройка конечных устройств LoRa с использованием различных методов активации.

2 Краткие теоретические сведения

Технология модуляции LoRa (Long Range) представляет собой метод модуляции, который обеспечивает значительно бóльшую дальность связи (зону покрытия). Метод основывается на технологии модуляции с расширенным спектром и вариации линейной частотной модуляции с интегрированной прямой коррекцией ошибок.

Протокол LoRaWAN (Long Range Wide-Area Networks, LoRaWAN) это MAC-протокол для высокочастотных сетей с большим радиусом действия и низким собственным потреблением мощности, для маломощных глобальных радиальных сетей (Low Power Wide Area Networks, LPWAN) типа звезда.

В протоколе LoRaWAN существует два метода активации устройств:

- ОТАА, Over-The-Air Activation (требуется пройти процедуру присоединения (join procedure), во время которой вырабатываются сессионные ключи шифрования и адрес DevAddr).
- АВР, Activation By Personalization (не требуется проходить процедуру присоединения, ключи шифрования и адрес DevAddr

записываются в устройство заранее (персонализация устройства))

В результате активации, каждое устройство должно иметь:

- End-device address (DevAddr) — локальный адрес устройства в данной сети [32 бита]. DevAddr состоит из двух полей: NwkID (идентификатор сети, биты 31...25) и NwkAddr (сетевой адрес, биты 24...0);
- Network session key (NwkSKey) — сетевой сессионный ключ [128 бит], используемый для расчета и проверки поля MIC (message integrity code) сообщений при обмене между оконечным устройством и сетевым сервером (Network Server), а также шифрования сообщений MAC-уровня.
- Application session key (AppSKey) — сессионный ключ [128 бит], используемый для шифрования данных на уровне приложения (между оконечным устройством и сервером приложения).

При ABP активации, эти значения вводятся в устройство при его настройке, При OTAA активации эти значения генерируются при присоединении к станции.

При активации OTAA, устройство должно содержать в себе:

- End-device identifier (DevEUI) — уникальный идентификатор, который присваивается устройству в процессе производства [64 бита].
- Application identifier (AppEUI) — уникальный идентификатор приложения [64 бита]
- Application key (AppKey) — ключ [128 бит], который используется в процессе присоединения к сети для получения сессионных ключей NwkSKey и AppSKey.

Конечные устройств в сетях LoRa могут работать в 3 режимах, класс А , класс Б. класс С.

Класс А

Устройства класса А после каждой передачи открывают два коротких временных окна на прием. Интервалы от конца передачи до открытия первого и второго временных окон могут конфигурироваться, но должны быть одинаковыми для всех устройств в данной сети. Устройства класса А являются самыми низкопотребляющими, но для передачи сообщения от сервера к конечному устройству необходимо дождаться следующего исходящего сообщения от этого устройства

Класс Б

Вдобавок к окнам приема, определенным для устройств класса А, устройства класса В открывают дополнительные окна приема по расписанию. Для синхронизации времени открытия дополнительных окон приема шлюзы излучают маячки (beacons). Использование класса В гарантирует, что при опросе конечных устройств задержка отклика не будет превышать определенную величину, определяемую периодом маячков.

Класс С

Устройства класса С находятся в режиме приема практически всё время за исключением промежутков, когда они передают сообщения. Класс С может применяться там, где не нужно изо всех сил экономить энергию (счетчики электрической энергии) или где необходимо опрашивать конечные устройства в произвольные моменты времени

3 Задание

Осуществить настройку конечных устройств LoRa и реализовать их подключение к базовой станции с использованием различных методов активации.

4 Ход работы

В работе рассматриваются устройства от RAKwireless, а именно устройства моделей RAK811 LPWAN Breakout Module и RAK811. Имеется возможность подключить устройства по uart, распиновка представлена на рисунке 1.

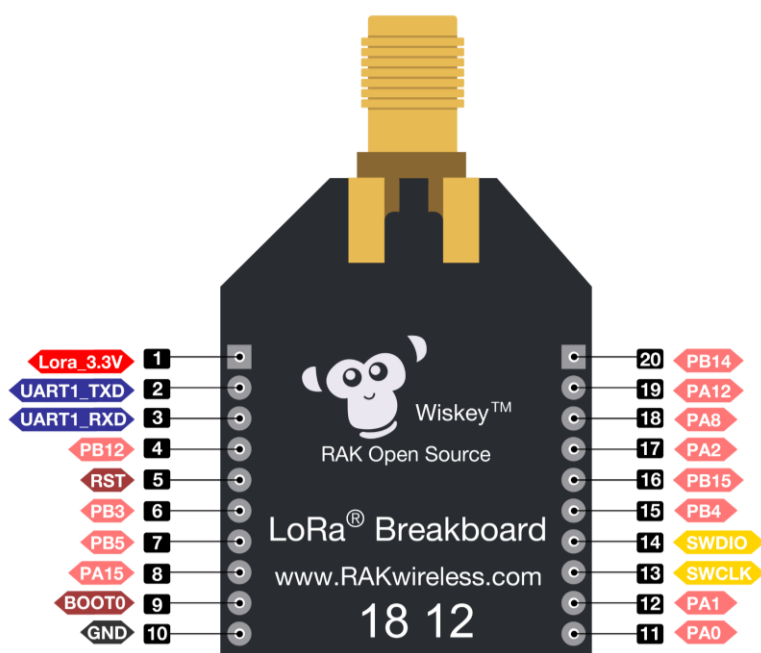


Рисунок 1 - Распиновка модуля

Управление устройством осуществляется посредством отправки AT - команд, для передачи команд можно воспользоваться любой терминальной программой, например Putty, или специальной утилитой от производителя устройств RAK Serial Port Tool, поддерживается только Windows (для скачивания <https://downloads.rakwireless.com/LoRa/Tools/>) . Для работы с использованием любого варианта требуется указать COM-порт устройства и выбрать скорость передачи данных (BaudRate) - 115200.

Рассмотрим отправку команд с использованием RAK Serial Port Tool, окно приложения представлено на рисунке 2 , справа представлен список AT-команд, для отправления команды требуется нажать send напротив команды, также имеется возможность ввода команды руками, для этого нужно ввести команду в окно Sending и нажать Send. Перечень всех AT-команд представлен в “RAK811 Lora AT Command User Guide”. Рассмотрим основные группы AT-команд:

<m>- подразумевает модуль с которым идёт работа (device- работа с устройством, lora - работа с настройками Lora и LoRaP2P для работы с режимом P2P)

<n>- подразумевает параметр или функция с которым идёт работа(подробнее с параметрами можно ознакомиться по ссылке <https://docs.rakwireless.com/Product-Categories/WisDuo/RAK811-Breakout-Board/AT-Command-Manual/#general-at-command>)

Команда чтения - считывает текущую конфигурацию или состояние модуля. Имя команды и список параметров разделены = символом. <m>Параметр отделяется с соответствующим значением <n> с помощью :
. at+get_config=<m>:<n>

Пример at+get_config=lora:status (at+get_config - это команда чтения, мы спрашиваем информацию, lora - это модуль откуда мы спрашиваем информацию, status - что именно мы спрашиваем)

Команда записи : записывает / изменяет текущую конфигурацию модуля. Имя команды и список параметров разделены = символом. <m>Параметр отделяется с соответствующим значением <n> с помощью :
. at+set_config=<m>:<n>

Пример at+set_config=lora:join_mode:1 (at+set_config - это команда записи, lora -это модуль в который мы хотим изменить , join_mode - параметр который мы хотим изменить , :1 - параметр на который мы изменяем)

Операционная команда : некоторые команды не являются командами чтения и записи, но используются для выполнения действия.

Пример `at+send=lora:<port>:<data>`

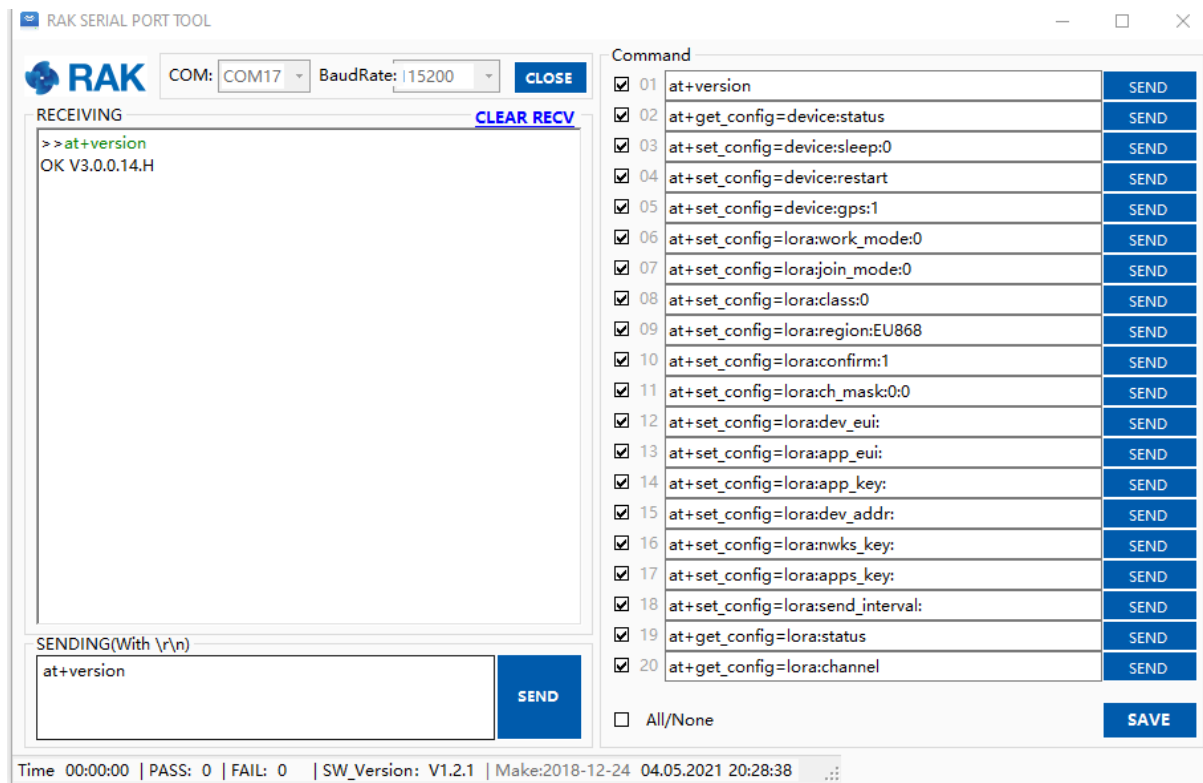


Рисунок 2 - Окно RAK Serial Port Tool

Если использовать программу Putty для настройки, то перед подключением к устройству нужно изменить настройки локального эхо и конца строки как на рисунке 3.

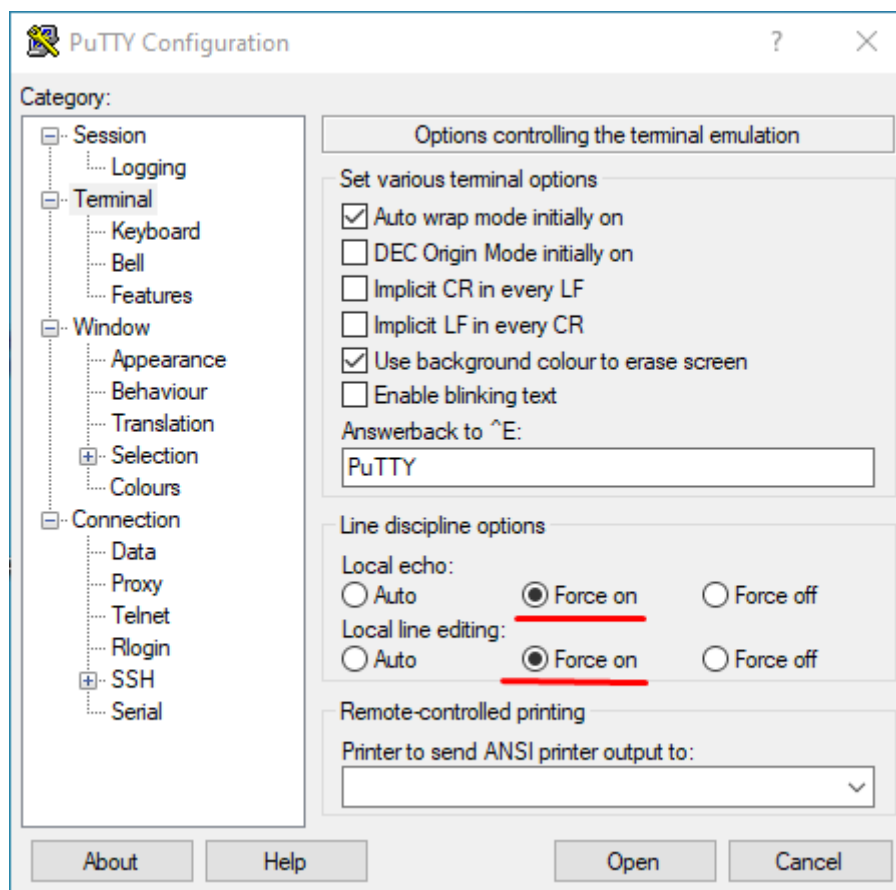


Рисунок 3 - Изменение настроек терминала

После активации подключения, можно в терминале вводить АТ-команды, данный способ менее удобен, так как есть риск ошибиться в вводимой команде. Для просмотра всех at команд можно ввести at+help

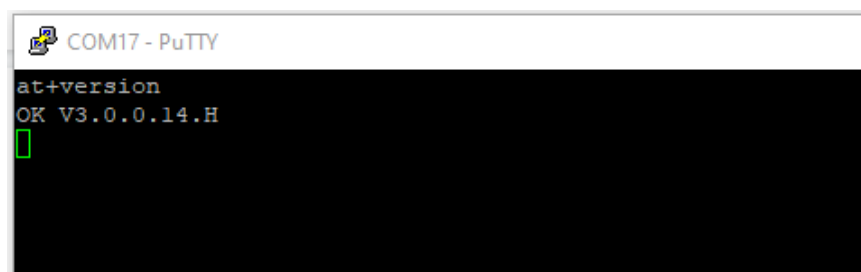


Рисунок 5 - Ввод команды

Если при вводе команды вы получаете ошибки(Error 1 или 2), то требуется обновить прошивку устройства, для этого отправьте команду at+set_config=device:boot(это команда должна перевести устройство в режим обновления) , если команда успешна(получено сообщение <BOOT MODE>) то вы можете обновлять прошивку, если вы снова получаете

ошибки то следует подать постоянное питание на 9 pin(рисунок 6) для перевода устройства в boot mode .



RAK811 LoRa Breakout USB - UART

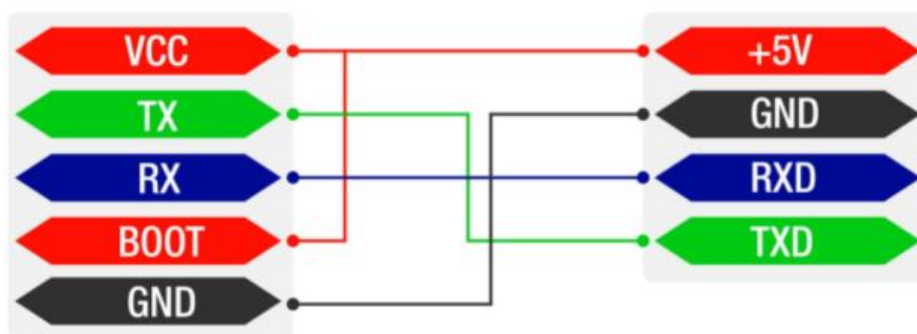


Рисунок 6 - Подключение

Для обновления прошивки на устройстве можно воспользоваться специальным приложением от производителя RAK Device Firmware Upgrade tool (https://downloads.rakwireless.com/LoRa/Tools/RAK_Device_Firmware_Upgrade_tool/). Для обновления в этой программе: сначала нужно выбрать COM-порт к которому подключено устройство, затем выбрать нужную прошивку, требуется файл с расширением .bin (прошивку можно найти на сайте производителя устройства (<https://docs.rakwireless.com/Product-Categories/WisDuo/RAK811-Breakout-Board/Datasheet/#hardware>)), в нашем

случае можно взять прошивку V3.0.0.14.H ,так как она соответствует нужному региону (рисунок 7), или более новую версию с приставкой H) и нажать кнопку upgrade, на рисунке 8 представлен конечный результат обновления .

Module	Region	Frequency
RAK811(L)	Europe	EU433
	China	CN470
RAK811(H)	Europe	EU868
	North America	US915
	Australia	AU915
	Korea	KR920
	Asia	AS923
	India	IN865

Рисунок 7 - Регионы

ВАЖНО!!! Для обновления прошивки, устройство должно находиться в Boot mode.

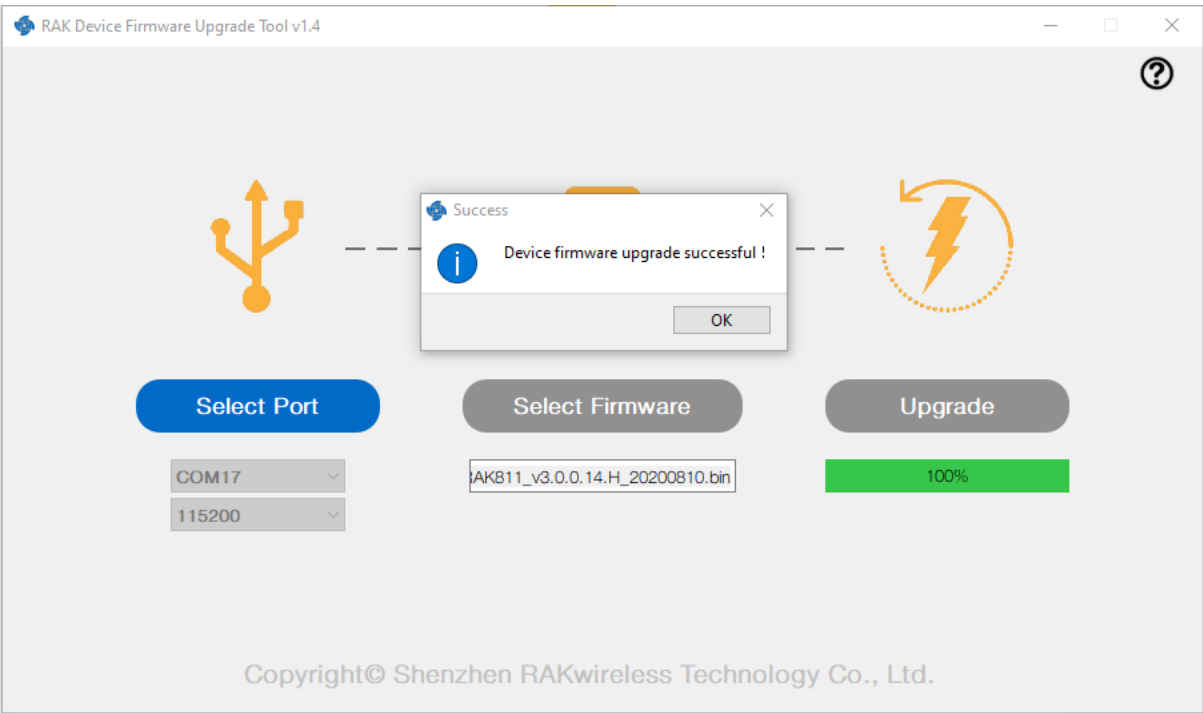


Рисунок 8 - Результат обновления

После обновления прошивки, можно приступить к настройке режима активации устройств . Существует 2 режима активации устройств “ОТАА” и “АВР”.

Рассмотрим сначала АВР активацию. В режиме АВР (Activation by Personalization) в памяти устройства должны быть DevAddr, NwkSKey, AppSKey для подключения к конечной станции.

Параметры DevAddr, NwkSKey, AppSKey вводятся в 16-ричной системе счисления.

Рассмотрим последовательность команд для настройки работы устройства в режиме АВР, сначала нужно выбрать режим активации, далее последовательность команд не важна:

- at+set_config=lora:join_mode:1 (устанавливает режим активации АВР);
- at+set_config=lora:class:0 (0 устанавливает устройство в класс А, 2 устанавливает устройство в класс С);
- at+set_config=lora:region:EU868 (EU868 устанавливает регион EU868, также имеется возможность установить US915, AU915, KR920, AS923, IN865);
- at+set_config=lora:dev_addr:26011af9 (указываем DevAddr);
- at+set_config=lora:nwks_key:c280cb8d1df688bc18601a97025c5488 (указываем NwkSKey);
- at+set_config=lora:apps_key:4d42ec5caf97f03d833cdaf5003f69e1 (указываем AppSKey).

После завершения ввода с помощью команды at+get_config=lora:status посмотрите настройки активации, пример показан на рисунке 9

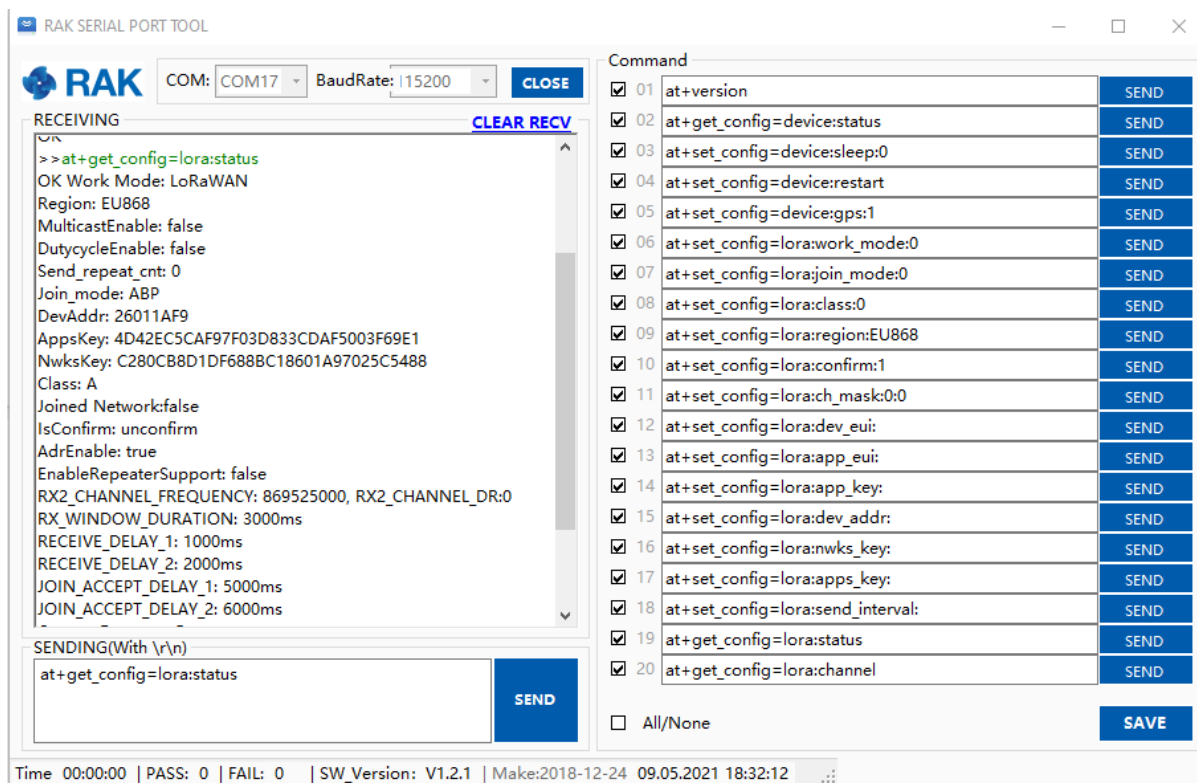


Рисунок 9 - Активация по ABP

Активацию по ABP не рекомендуется использовать, так как она менее защищенная по сравнению с OTAA .

При активации OTAA оконечное устройство должно проходить процедуру присоединения к сети каждый раз, когда сессионная информация (локальный адрес DevAddr, ключи NwkSKey, AppSKey) в устройстве отсутствует или неактуальна. Перед процедурой присоединения в устройство должны быть записаны следующие параметры: DevEUI, AppEUI, AppKey для подключения к станции, с помощью этих значений будет генерироваться ключи для работы со станцией .

End-device identifier (DevEUI) — уникальный идентификатор, который присваивается устройству в процессе производства [64 бита].

Application identifier (AppEUI) — уникальный идентификатор приложения [64 бита]

Application key (AppKey) — ключ [128 бит], который используется в процессе присоединения к сети для получения сессионных ключей NwkSKey и AppSKey.

Параметры DevEUI, AppEUI, AppKey вводятся в 16-ричной системе счисления в устройство пользователем.

Рассмотрим последовательность команд для настройки работы устройства в режиме OTAA:

at+set_config=lora:join_mode:0 (указываем активацию OTAA)

at+set_config=lora:class:0

at+set_config=lora:region:EU868

at+set_config=lora:dev_eui:5e9d1e0857cf25f1 (указываем DevEUI)

at+set_config=lora:app_eui:5e9d1e0857cf25f1 (указываем AppEUI)

at+set_config=lora:app_key:f921d50cd7d02ee3c5e6142154f274b2

(указываем AppKey)

Введите эти команды но со своими значениями DevEUI, AppEUI, AppKey, пример показан на рисунке 10

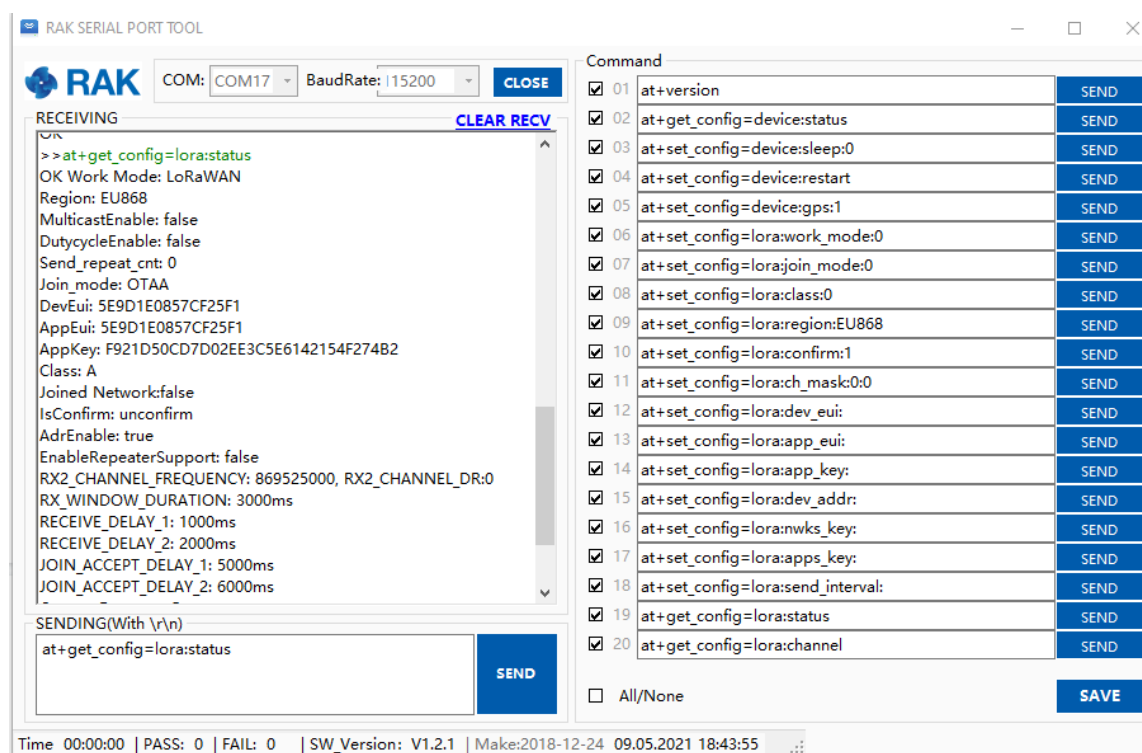


Рисунок 10 - Активация по OTAA

Для перевода устройства в режим сна , нужно ввести команду `at+sleep` , режим сна нужен для уменьшения энергопотребления .

Для перезагрузки устройства существует команда `at+reset=0` или `at+set_config=device:restart`

Команда `at+reload` сбрасывает настройки к начальным значениям

После настройки активации вы можете подключить свое устройство у конечному устройству с помощью команды `at+join`

Для просмотра частотных каналов текущего региона существует команда `at+get_config=lora:channel`

Rak 811 может работать как про протоколу LoRaWAN, так и LoRa P2P, для выбора режима есть команда `at+set_config=lora:work_mode:0` (0 - LoRaWAN, 1 - LoRa P2P)

Для восстановления заводский параметров ОТАА, АВР активации , есть команда `at+set_config=lora:default_parameters`

Для изменения скорости передачи данных существует команда `at+set_config=lora:dr:0` (0 - 250 б/с, 1- 440 б/с, 2 -980 б/с, 3 - 1760 б/с, 4 - 3125 б/с. Чем выше скорость передачи. тем больше вероятность потери сообщения)

Лабораторная работа №3 “Исследование взаимодействия базовой станции и конечного устройства LoRa”

1 Цель работы

Целью работы является исследование взаимодействия базовой станции LoRa и конечных устройств, построение учебной сети.

2 Краткие теоретические сведения

Типовая беспроводная сеть LoRaWAN представляет собой совокупность шлюзов (gateways), рассылающих сообщения между оконечными устройствами (end-devices) и центральным сервером (Network Server, NS), и характеризуется «звездной» топологией «star-of-stars» (Рисунок 1)

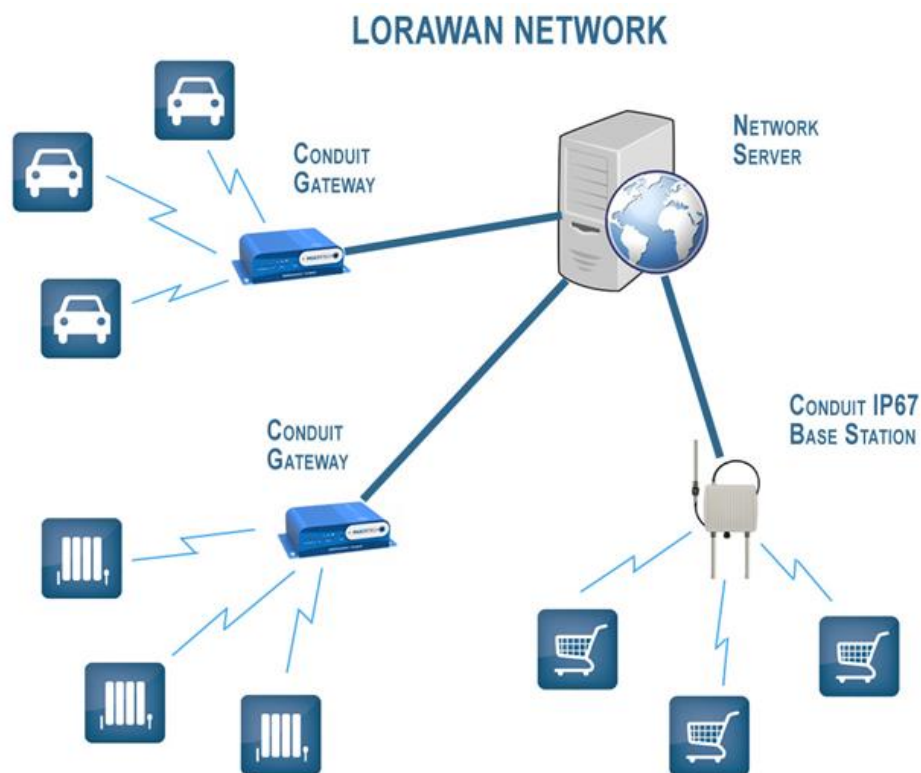


Рисунок 1 - Структура сети LoRaWAN

Связь между шлюзами и центральным сервером осуществляется через стандартные IP-соединения, а между шлюзами и оконечными устройствами — через беспроводные соединения, использующие широкополосную модуляцию LoRa. Связь между шлюзами и оконечными устройствами является двусторонней, но предполагается, что основной объем данных передается от оконечных устройств к шлюзам.

Основные преимущества беспроводных сетей LoRaWAN обусловлены использованием широкополосной модуляции LoRa и безлицензионных диапазонов частот Сети LoRaWAN:

- совместимы с существующими сетями/технологиями беспроводной передачи данных;
- обладают высокой помехоустойчивостью;
- способны обслуживать десятки и сотни тысяч устройств;
- обеспечивают большую зону охвата и малое энергопотребление оконечных устройств.

3 Задание

Реализовать подключение к базовой станции LoRa конечных устройств используя режимы активации OTAA/ABP.

4 Ход работы

Для подключения устройства к станции нужно сначала его добавить, для этого нужно зайти в Vega Admin Tool в вкладку Devices и добавить устройство через Add new device.

Для начала добавим устройство по OTAA активации .

В окна AppEUI и AppKey введите данные, которые были записаны в устройство в предыдущей работе

В настройках Main Settings есть параметры: End-device-name это имя, которое мы хотим присвоить устройству, End-device identifier это идентификатор устройства, обычно он написан на самом устройстве, этот параметр должен быть заполнен в 16-ричной форме и иметь размер [64 бита]), если используется ОТАА активация, то нужно указать тот , что вы указали на конечном устройстве, Если используется ABP то можно указать любой идентификатор, End-device class это выбор класса работы устройства , выбираем класс А , End-device group это выбор группы в котором будет находиться устройство, группа позволяет группировать устройства, упрощая просмотр, это поле можно оставить пустым . Частоту выберите EU868. Пример настроек показан на рисунке 2.

Device settings ×

Activation by personalisation (ABP)

End-device address (devAddr)

DEVADDR

Application session key (AppSKey)

APPSKEY

Network session key (NwkSKey)

NWKSKEY

Over-the-air activation (OTAA)

Application identifier (AppEUI)

5E9D1E0857CF25F3

Application key (AppKey)

F921D50CD7D02EE3C5E6142154F274B3

Main settings

End-device name

Test3

End-device identifier (DevEUI)

5E9D1E0857CF25F3

End-device class

Class A

End-device group

device group

Regional settings

Frequency plan

EU868

Nº	Frequency	Enabled
1	FIXED	<input checked="" type="checkbox"/>
2	FIXED	<input checked="" type="checkbox"/>
3	FIXED	<input checked="" type="checkbox"/>
4	867100000	<input checked="" type="checkbox"/>
5	867300000	<input checked="" type="checkbox"/>
6	867500000	<input checked="" type="checkbox"/>
7	867700000	<input checked="" type="checkbox"/>
8	867900000	<input checked="" type="checkbox"/>

RX2 Frequency, Hz

869525000

☐ Expert settings

Close

Save

Рисунок 2 - Добавление устройства по ОТАА активации

После добавления устройства, оно появится в списке и информацию о нем можно будет посмотреть рисунок 3 .

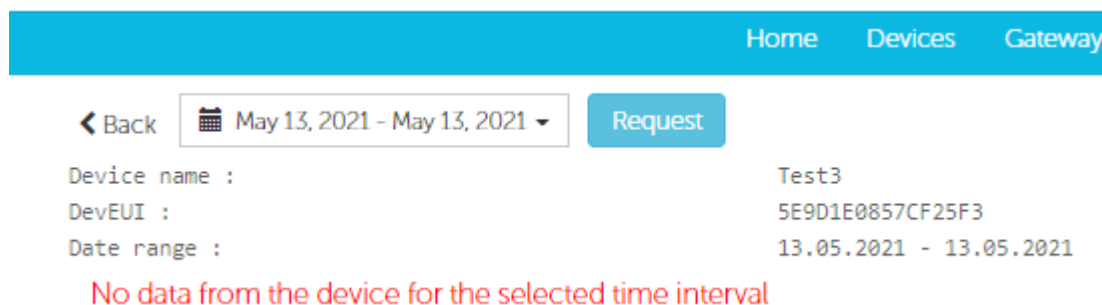


Рисунок 3 - Добавленное устройство

Для того чтобы устройство передавало данные, нужно подключить устройство к станции, для этого отправьте команду `at+join` с конечного устройства (рисунок 4)

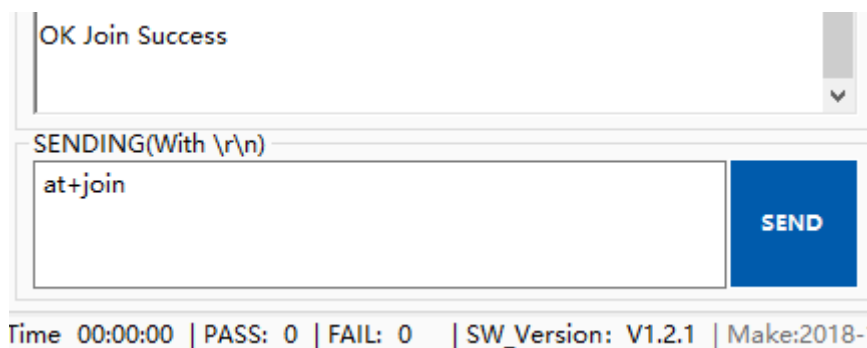


Рисунок 4 - Подключение устройства

После этого на сервере появится информация о подключении устройства, рисунок 5. Также во вкладке устройства можно посмотреть какие данные были приняты и отправлены. После подключения можно увидеть сообщения `Join_REQ` и `Join_ACC`. `Join_REQ` говорит о том, что пришёл запрос на генерации ключей для подключения устройства по ОТАА , а `Join_ACC` ,что сервер отправил ключи для подключение устройства, пример этих запросов можно увидеть на рисунке 6.

```

D:\ТУСУР\3 курс\6 семестр\ГПО\Исходники\IoT Vega Server (win) v1.2\Iot-vega-server.exe
IoT Vega Server 1.2.1
LOG: current file for log messages [./history_1.log]
INFO: Table queueTransmit is cleaned
INFO: Table "bufMacDevParams" is cleaned
INFO: DB-secure scanner started...
DEBUG [CDevicesCountInfo]: vega[0], totalNonVega[1000], usedNonVega[2]
INFO: DB-secure scanner successfully finished
UDP socket has opened. IP[192.168.17.106:8001]
DEBUG: UdpServer handler is started
WebSocketServer has opened. Port[8002]
[CWebSocketServer] New connection
[CWebSocket::closeConnectionSlot]
[CWebSocketServer] New connection
>> 00006CC3743EE7C7 | LATENCY 78 |<<
[CWebSocket::closeConnectionSlot]
[CWebSocketServer] New connection
>> 00006CC3743EE7C7 | LATENCY 1 |<<
>> 00006CC3743EE7C7 | LATENCY 1 |<<
INFO: [5E9D1E0857CF25F3] is NOT validated VEGA device. Limit counter is used: remain [997] free
>> 00006CC3743EE7C7 | LATENCY 0 |<<
>> GW-00006CC3743EE7C7:JOIN_REQ | 5E9D1E0857CF25F3 | 2021-05-13 11:06:11.113 | 867.3 | SF 7 | RSSI: -84 | 9.2 | VA
LIDATED
DEBUG-INFO: [CallBsHandler::tryToTxPacket] GW-00006CC3743EE7C7 is validated
[CrxDeferredPacket] 5e9d1e0857cf25f3 Send now is confirmed via [00006cc3743ee7c7]. Waiting validation
>> 00006CC3743EE7C7 | LATENCY 155 |<<
>> GW-00006CC3743EE7C7:JOIN_ACC | 5E9D1E0857CF25F3 | 2021-05-13 11:06:11.693 | 867.3 | SF 7 | VALIDATED

```

Рисунок 5 - Информация о подключении устройства

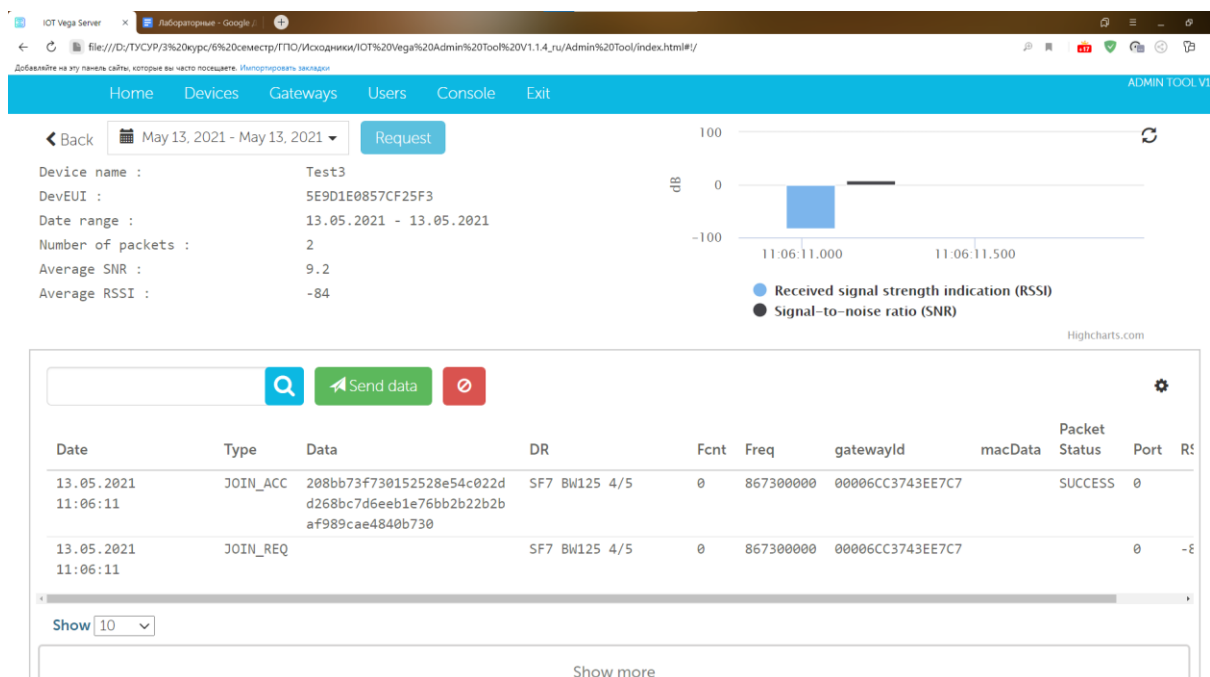


Рисунок 6 - Подключение устройства

После того как устройство было подключено к сети можно отправить данные, для отправки данных нужно использовать команду `at+send=lora:2:1234567890` на конечном устройстве (Сообщение записывается в 16-ричной форме, размер сообщения зависит от скорости передачи данных, так как передача длится определённое окно, то может не хватить времени на большое сообщение) рисунок 7

OK

SENDING(With \r\n)

at+send=lora:2:1234567890

SEND

Time 00:00:00 | PASS: 0 | FAIL: 0 | SW_Version: V1.2.1 | Make:2018-

Рисунок 7 - Отправка данных

На рисунке 8 можно увидеть какие данные были приняты

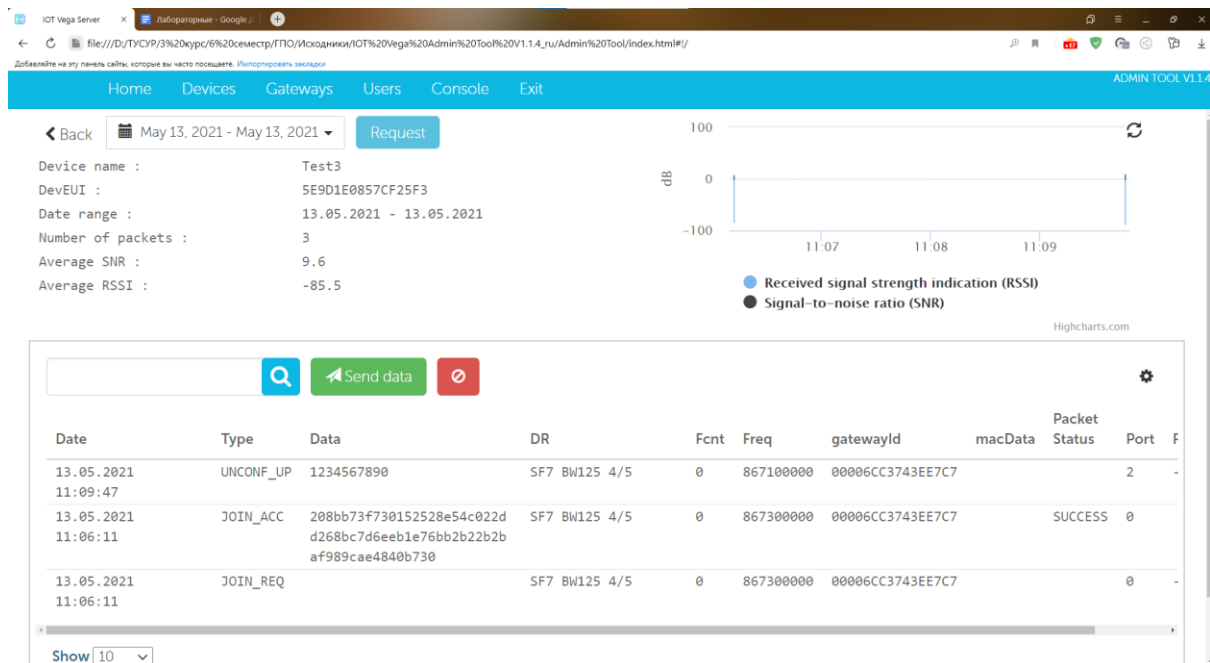


Рисунок 8 - Принятые данные

Теперь подключим устройство используя ABP активацию. Используя Vega Admin Tool перейдите на вкладку Devices и добавьте новое устройство, но теперь укажите End-device address, Application session key, Network session key вашего устройства, частоту выберете EU866, и укажите настройки main settings, пример настроек показан на рисунке 9.

Device settings

Activation by personalisation (ABP)

End-device address (devAddr)

26011AF9

Application session key (AppSKey)

4D42EC5CAF97F03D833CDAF5003F69E1

Network session key (NwkSKey)

C280CB8D1DF688BC18601A97025C5488

Over-the-air activation (OTAA)

Application identifier (AppEUI)

APPEUI

Application key (AppKey)

APPKEY

Main settings

End-device name

Test3

End-device identifier (DevEUI)

5E9D1E0857CF25F3

End-device class

Class A

End-device group

device group

Regional settings

Frequency plan

EU868

Nº	Frequency	Enabled
1	FIXED	<input checked="" type="checkbox"/>
2	FIXED	<input checked="" type="checkbox"/>
3	FIXED	<input checked="" type="checkbox"/>
4	867100000	<input checked="" type="checkbox"/>
5	867300000	<input checked="" type="checkbox"/>
6	867500000	<input checked="" type="checkbox"/>
7	867700000	<input checked="" type="checkbox"/>
8	867900000	<input checked="" type="checkbox"/>

RX2 Frequency, Hz

869525000

☐ Expert settings

Close

Save

Рисунок 9 - Пример активации по ABP

Подключите конечное устройство к станции и отправьте сообщение на сервер (рисунок 10)

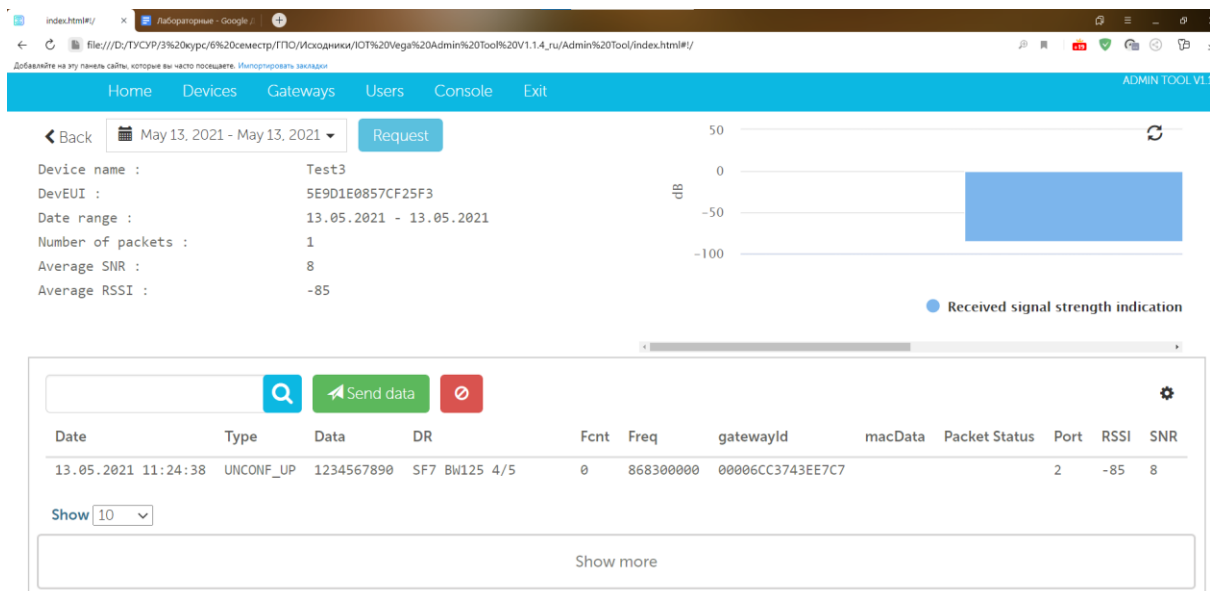


Рисунок 10 - Принятые данные по АВР

Конкретные устройства можно приписывать конкретным пользователям. Для этого требуется зайти в Vega admin tool во вкладку Users и создать нового пользователя (add new user). При создании пользователя, нужно указать: имя пользователя (login), пароль (Password), выбрать роль пользователя (выберете роль user) и в Permissions DevEui введите DevEui или имя устройства (при вводе выпадет список устройств подходящих под ваш запрос)(рисунок 11)

User settings

Main settings

Login

User1

Password

123

Permissions pattern

user

Permissions commands api

Common

☒ View data from devices

☐ Send data to devices

External application

☒ View devices by external app

☒ Add/edit devices by external app

☒ Delete devices by external app

Network management

User privileges

Device access

Selected devices

☒ Send online data

End device uplinks

☐ Send MAC commands

☐ Console enable

Permissions DevEui

TEST3 5E9D1E0857CF25F1

Add devEui

Рисунок 11 - Создание пользователя и выбор устройства

После подключения устройства, можно отправлять данные.

Лабораторная работа №4 “Развертывание сети сбора данных LoRa”

1 Цель работы

Целью работы является построение тестовой сети сбора данных с использованием технологии LoRa.

2 Краткие теоретические сведения

Интернет вещи , для передачи данных используют упрощенные сетевые протоколы. Давайте рассмотрим самые популярные из них.

MQTT или Message Queue Telemetry Transport – это легкий, компактный и открытый протокол обмена данными созданный для передачи данных на удаленных локациях, где требуется небольшой размер кода и есть ограничения по пропускной способности канала.

Основные особенности протокола MQTT:

- Асинхронный протокол
- Компактные сообщения
- Работа в условиях нестабильной связи на линии передачи данных
- Поддержка нескольких уровней качества обслуживания (QoS)
- Легкая интеграция новых устройств

Обмен сообщениями в протоколе MQTT осуществляется между клиентом (client), который может быть издателем или подписчиком (publisher/subscriber) сообщений, и брокером (broker) сообщений (например, Mosquitto MQTT). Издатель отправляет данные на MQTT брокер, указывая в сообщении определенную тему, топик (topic). Подписчики могут получать разные данные от множества издателей в зависимости от подписки на соответствующие топики.

Протокол CoAP предназначен для взаимодействия простых устройств, например датчиков малой мощности, выключателей, клапанов, которые управляются или контролируются удаленно через сеть Интернет.

Такие устройства используются в области Интернета вещей, а порождаемый ими информационный обмен называется межмашинным взаимодействием (M2M). Существенная особенность протокола CoAP – это его совместимость с протоколом HTTP, что обеспечивает при его использовании взаимодействие совокупности устройств IoT, формирующих некую сеть, с всемирной паутиной Интернет.

MQTT-SN спроектирован как можно более похожим на MQTT, но адаптирован к особенностям беспроводной среды передачи данных, таким как низкая пропускная способность, высокая вероятность сбоя в соединениях, короткая длина сообщения и т.п. Также оптимизирован для дешевых устройствах с аккумуляторным питанием и ограниченными ресурсами по обработке и хранения. Основные отличия MQTT-SN от MQTT это уменьшение размера сообщения, в основном, за счет сокращения “служебной” информации, особенно интересна реализация QOS -1 - когда клиент отправляет сообщение без подтверждения о подтверждении доставки, и возможность использование отличного от TCP протокола, можно встретить реализации для UDP, UDP6, ZigBee, LoRaWAN, Bluetooth.

Если рассмотреть функции каждого участника обмена то получается следующее:

- MQTT-SN клиенты (как принимающие так и передающие сообщения) - подключаются к MQTT брокеру, через MQTT-SN шлюзы.
- MQTT-SN шлюз – основная функция двусторонняя "синтаксическая" трансляция MQTT-SN ↔ MQTT.
- MQTT-SN форвардер - если клиентам недоступен шлюз, они могут посылать и принимать сообщения через него.

- MQTT брокер - сервер, своеобразное ядро системы, который тем и занимается что пересылает сообщения.

Работа будет выполняться на Ubuntu 20.04 TLS

3 Задание

Используя модули UMDK-RF, постройте тестовую сеть сбора и передачи данных

4 Ход работы

Для работы будут использоваться модули Umdk-RF (рисунок 1)

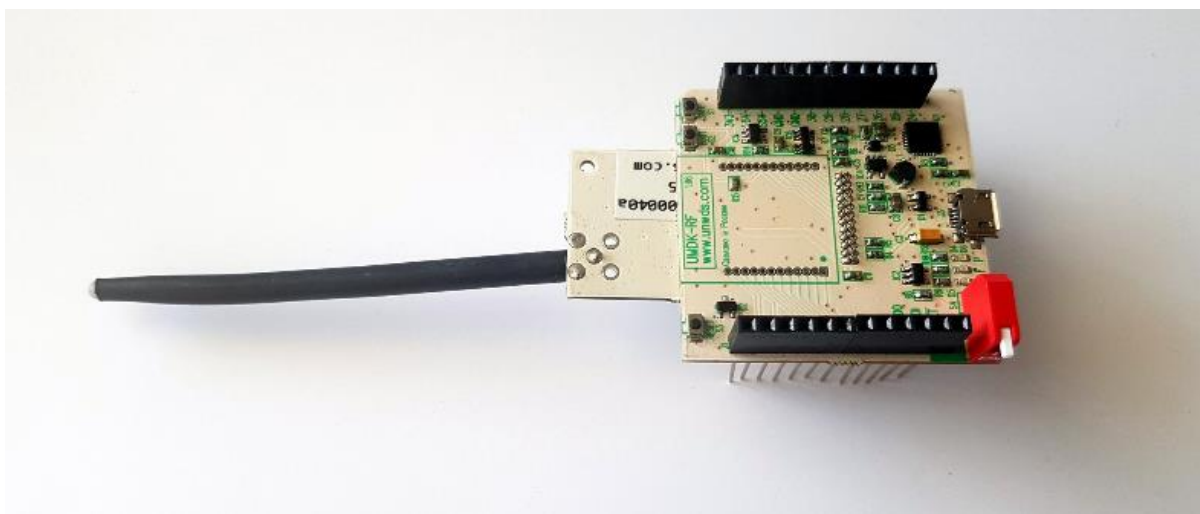


Рисунок 1 - Используемое устройство

Для работы понадобятся минимум 2 таких устройства и датчики, одно устройство будет использоваться как шлюз базовой станции, именно оно будет принимать данные с конечных устройств и осуществлять их регистрацию в сети. Остальные устройства нужно прошить, как конечные устройства. Конечные устройства настроены в режим активации ОТАА , правда с некоторыми особенностями именно поэтому их невозможно подключить к Вега БС. Скачать прошивки для устройств можно по ссылке <https://github.com/unwireddevices/RIOT/releases>. Потребуется два файла:

- loralan-public.hex - файл прошивки конечного устройства;

- loralan-gateway.hex - файл прошивки для базовой станции.

Для выполнения работы потребуется установить нужные пакеты. Приведем список команд с пояснением:

`sudo apt install build-essential` (build-essential нужен для сборки проектов с git)

`sudo apt install git` (для использования проектов с github)

Для обновления прошивки понадобится stm32flash.

`git clone https://github.com/unwireddevices/stm32flash` (копируем проект)

`cd stm32flash` (переходим в папку)

`sudo make` (собираем проект)

`sudo make install` (устанавливаем проект)

`sudo apt install gterm` (устанавливаем терминал для работы с устройствами)

Для обновления устройства, требуется перевести его в режим обновления, это можно сделать нажав `reboot` и нажав `boot` , если вы увидите что зеленый диод начал мигать значит вы сделали правильно. Также можно использовать команду `sudo stm32flash /dev/ttyACM0` (у вас порт может называться по другому) для проверки соединения с устройством(рисунок 2).

```
s@s-VirtualBox:~$ sudo stm32flash /dev/ttyACM0
stm32flash 0.5.2

http://stm32flash.sourceforge.net/

Interface serial_posix: 115200 8E1
Version      : 0x31
Option 1     : 0x00
Option 2     : 0x00
Device ID    : 0x0427 (STM32L1xxxC)
- RAM        : Up to 32KiB (4096b reserved by bootloader)
- Flash      : Up to 256KiB (size first sector: 16x256)
- EEPROM     : Up to 8KiB
- Option RAM : 32b
- System RAM : 8KiB
```

Рисунок 2 - Проверка соединения

Обновление прошивки устройства осуществляется с помощью команды:

```
sudo stm32flash -b 230400 -E -w <имя файла прошивки> -v
/dev/ttyUSB0
```

Обратите внимание что вместо ttyUSB0 у вас будет ваш порт к которому подключено устройство. После загрузки прошивки на конечное устройство и базовую станцию, можно подключиться через терминал к устройству и изучить их настройки (рисунок 3-4)


```

Unwired Range firmware by Unwired Devices
www.unwds.com - info@unwds.com
powered by RIOT - www.riot-os.org
*****
Version: 1.71 (Dec  5 2017 18:59:47)
STM32L151CC 32 MHz (PLL/HSE clock)
32 KB RAM, 256 KB flash, 8 KB EEPROM

[config] Configuration loaded from NVRAM
[device] Initializing...
[ node configuration ]
NOJOIN = no
JOINKEY = 0x....AAAA
EUI64 = 0x807b85902000058a
APPID64 = 0x0000000000000000
REGION = Russia 868
CHANNEL = 1 [869050000]
DATARATE = 4
CLASS = C
MAXRETR = 1

```

Рисунок 3 - Настройки конечного устройства

```

Version: 1.71 (Dec  5 2017 19:00:32)
STM32L151CC 32 MHz (PLL/HSE clock)
32 KB RAM, 256 KB flash, 8 KB EEPROM

[config] Configuration loaded from NVRAM
[device] Initializing...
[ gate configuration ]
JOINKEY = 0x....AAAA
EUI64 = 0x807b859020000598
APPID64 = 0x0000000000000000
REGION = Russia 868
CHANNEL = 1 [869050000]
DATARATE = 4

```

Рисунок 4 - Настройки базовой станции

JOINKEY - ключ шифрования в сети, он исключает присоединение посторонних. В нашем случае ключ очень простой - это АAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA.

EUI64 - 64-битный идентификатор устройства.

REGION и CHANNEL - определяют частоту, на которой работает устройство

DATARATE - скорость передачи данных; для выполнения учебных задач оптимально будет поставить скорость 7 (наивысшую), тогда как для задач реальной жизни больше подойдет наименьшая скорость (0), поскольку в этом случае дальность будет максимальной.

MAXRETR - количество попыток передачи данных. Имеет смысл поставить 5, если много помех и устройства постоянно отсоединяются (вы увидите сообщения об этом в консоли). Тогда уменьшится количество повторных подключений к сети, и в целом система будет работать быстрее. По умолчанию стоит одна попытка - это означает, что устройство отправляет сообщение, ждёт 3 секунды, и если ответ не получен, то оно считает, что связь потеряна и пробует подключиться заново. Соответственно, если MAXRETR = 5, то будет 15 секунд на получение ответа.

Для настройки конечного устройства имеются команды:

set eui64 807b859020000001 (задается идентификатор устройства, для каждого устройства в сети он должен быть уникальным)

set appid64 0000000000000001 (задается код приложения , его можно не менять)

set joinkey 918771929bbab19898c8a787f70a8810 (задается ключ шифрования , он должен быть уникальным)

set dr 5 (задается скорость передачи данных . 0 - наименьшая , 7- наивысшая)

set ch 1 (выбирается канал передачи, 0 это 868850000 и 1 это 869050000)

После ввода требуется сохранить изменения командой save и перезапустить устройство командой reboot

С помощью команды help можно посмотреть весь список команд .

Для просмотра имеющихся модулей существует команда lsmod (рисунок 5)

```
RX: 45 bytes, | RSSI: -63
ls-ed: successfully joined to the network
ls: first RX window expired
ls: staying in RX mode
lsmod
[ available modules ]
[-] gpio (id: 1)
[-] 4btn (id: 2)
[-] gps (id: 3)
[-] lsm6ds3 (id: 4)
[-] lm75 (id: 5)
[-] lmt01 (id: 6)
[-] uart (id: 7)
[-] sht21 (id: 8)
[-] pir (id: 9)
[-] 6adc (id: 10)
[-] lps331 (id: 11)
[-] 4counter (id: 12)
[-] echo (id: 13)
[-] pwm (id: 14)
[+] opt3001 (id: 15)
[-] dali (id: 16)
[-] bme280 (id: 17)
>
```

Рисунок 5 - модули

для включения модуля нужно написать команду `mod id 1`(0 для выключения) где `id` это идентификатор модуля.

На конечных устройствах и базовой станции параметры `JOINKEY`, `REGION` и `CHANNEL` должны быть одинаковые.

Теперь нужно подключить `usb -uart` к базовой станции, Пины `UART1: DIO25 - TX, DIO26 - RX`(помните, что `TX` подключается `RX` и наоборот), землю можно использовать любую. С помощью `usb-uart` мы будем принимать данные с базовой станции.

После того как мы настроили устройства, нужно настроить вывод данных с базовой станции, для этого будем использовать `mosquitto` и `lora-mqtt`. `Mosquitto` - это брокер сообщений с открытым исходным кодом, который реализует протоколы `MQTT`, `lora-mqtt` - специальный сервис для работы с `mqtt` наших устройств.

`Mqtt` понадобится для выведения полученных данных, с конечных устройств, а `mosquito` для вывода данных на компьютере клиента.

Чтобы установить необходимые пакеты требуется выполнить последовательность команд:

apt install software-properties-common -y (устанавливаем скрипты для PPA)

add-apt-repository ppa:ubuntu-toolchain-r/test -y(добавляем ubuntu-toolchain-r/test)

DEBIAN_FRONTEND=noninteractive apt upgrade --allow-downgrades -y (обновляем сценарии для работы и разрешаем откатывать версию пакетов)

apt install gcc-7 g++ - unzip mosquitto mosquitto-clients libmosquitto-dev libc6-dev libssl-dev libc-ares-dev nano -y (устанавливаем mosquito и связанные с ним библиотеки)

sudo		update-alternatives		\
--install	/usr/bin/gcc	gcc	/usr/bin/gcc-7	60 \
--slave	/usr/bin/gcc-ar	gcc-ar	/usr/bin/gcc-ar-7	\
--slave	/usr/bin/gcc-nm	gcc-nm	/usr/bin/gcc-nm-7	\
--slave	/usr/bin/gcc-ranlib	gcc-ranlib	/usr/bin/gcc-ranlib-7	(задаём альтернативы)

update-alternatives --config gcc (обновляем используемые альтернативы)

systemctl enable mosquitto (проверяем сервис mosquitto)

systemctl start mosquitto (запускаем сервис mosquitto)

git clone https://github.com/Трапка/Test.git (клонировать репозиторий)

mv Test lora-mqtt (меняем имя папки для удобства)

cd lora-mqtt

CC=gcc make (собираем проект)

cp bin/mqtt /usr/local/bin/ (копируем программу, чтобы был доступен без пути)

chmod 755 /usr/local/bin/mqtt (выдаем права доступа)

mkdir /etc/lora-mqtt (создаём директорию)

cp dist/openwrt/files/mqtt.conf /etc/lora-mqtt/ (копируем настройки в созданную директорию)

`sed -i s/ttyATH0/ttySAC1/ /etc/lora-mqtt/mqtt.conf` (вместо ttySAC1 вводите порт usb uart, это нужно для определения нужного нам порта)

`cp dist/init.d/mqtt.init /etc/init.d/mqtt` (копируем файл с настройками)

`chmod 755 /etc/init.d/mqtt` (выдаем права)

`cd ..`

`systemctl enable mqtt` (проверяем статус mqtt)

`systemctl start mqtt` (запускаем mqtt)

Для просмотра данных, приходящий на базовую станцию, можно использовать `mosquitto`. Для получения данных нужно подписаться на нужный нам топик, пример `mosquitto_sub -h "ip компьютера" -t "devices/lora/#" -q 1` где:

`-h "ip компьютера"` - Адрес сервера (мы используем адрес компьютера)

`-t "devices/lora/#"` - Название топика, в котором публикуется сообщение

`-q 1` - Качество обслуживания (0 - проверки получения сообщения нет , 1 гарантирует получения сообщения, но возможно дублирование, 2 - гарантирует получение сообщения , без дублирования)

Все модули связи LoRa в нашем случае находятся в субтопике `devices/lora/` . Знак `#` в `devices/lora/#` , означает что мы подпишемся сразу на все субтопики неограниченной глубины, примерно как символ `*` (wildcard). Знак `+` означает спуск только на один уровень вниз. Например, эта команда будет получать данные с датчиков OPT3001 со всех модулей. `mosquitto_sub -h "ip компьютера" -t "devices/lora/+/opt3001" -q 1`

Ip можно посмотреть с помощью `ifconfig`

Если вы все правильно сделали то увидите приходящие сообщения (рисунок 6)

```

s@s-VirtualBox:~$ mosquitto_sub -h "10.0.2.15" -t "devices/lora/#" -q 1
{ "data": { "luminocity": 223 }, "status": { "devEUI" : "807B85902000058B", "rssi": -52, "temperature": 10, "battery": 3200, "date": "2021-05-24T06:43:37.348268Z" }}
{ "data": { "luminocity": 221 }, "status": { "devEUI" : "807B85902000058B", "rssi": -52, "temperature": 10, "battery": 3200, "date": "2021-05-24T06:43:40.421097Z" }}
{ "data": { "luminocity": 227 }, "status": { "devEUI" : "807B85902000058B", "rssi": -54, "temperature": 10, "battery": 3200, "date": "2021-05-24T06:44:44.292699Z" }}
{ "data": { "luminocity": 99 }, "status": { "devEUI" : "807B85902000058B", "rssi": -57, "temperature": 10, "battery": 3200, "date": "2021-05-24T06:45:51.204324Z" }}
{ "data": { "luminocity": 223 }, "status": { "devEUI" : "807B85902000058B", "rssi": -64, "temperature": 10, "battery": 3200, "date": "2021-05-24T06:46:55.108274Z" }}
{ "data": { "luminocity": 216 }, "status": { "devEUI" : "807B85902000058B", "rssi": -53, "temperature": 10, "battery": 3200, "date": "2021-05-24T06:47:57.131297Z" }}

```

Рисунок 6 - Приходящие сообщения

Если у вас не приходят сообщения , то можно с помощью команды `sudo mqtt` и `sudo service mqtt status` посмотреть работоспособность сервиса `mqtt`, или попробовать перезагрузить компьютер

С помощью изменения топика ,можно настраивать вывод данных с конкретных устройств (рисунок 7-8)

```

^Cs@s-VirtualBox:~$ mosquitto_sub -h "10.0.2.15" -t "devices/lora/+/#bme280" -q 1
{ "data": { "temperature": 26.0, "humidity": 31.8, "pressure": 991 }, "status": { "devEUI" : "801581105010b01d", "rssi": -4093, "temperature": 10, "battery": 3200, "date": "2021-05-24T08:31:14.724900Z" }}

```

Рисунок 7 - Получение данных только от bme280

```

VirtualBox:~$ mosquitto_sub -h "10.0.2.15" -t "devices/lora/807B85902000058A/bme280" -q 1
{ "data": { "temperature": 26.4, "humidity": 27.3, "pressure": 991 }, "status": { "devEUI" : "807B85902000058A", "rssi": -78, "temperature": 10, "battery": 3200, "date": "2021-05-24T08:46:39.748369Z" }}

```

Рисунок 8 - Получение данных от датчика на конкретном устройстве