

České  
vysoké  
učení technické  
v Praze

F3

Fakulta elektrotechnická  
Katedra telekomunikační techniky

## Low power wireless sensor network

**Tomáš Hyhlík**

Vedoucí: Ing. Bc. Marek Neruda, Ph.D  
Školitel–specialista: Ing. Bc. Lukáš Vojtěch, Ph.D  
Obor: Elektronika a komunikace  
Studijní program: Elektronika  
Říjen 2019

**Poděkování** | **Prohlášení**

## Abstrakt

## Abstract

**Klíčová slova:**

**Vedoucí:** Ing. Bc. Marek Neruda, Ph.D

**Keywords:**

**Title translation:** Low power wireless  
sensor network

## Obsah

0.1 Seznam zkratek .....	1	4.4.1 Syntaxe příkazů .....	10
<b>1 Účel LPWSN v kontextu současného stavu ve světě</b>	<b>2</b>	4.4.2 Adresace zařízení v síti .....	10
<b>2 Identifikace problému, který LPWSN řeší</b>	<b>3</b>	4.4.3 Statusy .....	10
<b>3 Stanovení požadavků návrhu zařízení</b>	<b>4</b>	4.4.4 Přidávání LoRaWAN zařízení do systému .....	11
3.1 Přístupové systémy .....	4	4.4.5 Odesílání dat z koncových zařízení .....	12
3.2 Implementace IoT do přístupového systému firmy IMA .....	5	4.4.6 Potvrzení .....	12
<b>4 Realizace zařízení</b>	<b>6</b>	4.4.7 Dotaz na příznaky .....	12
4.1 Výběr přenosové technologie ....	6	4.5 Komunikace přes USB .....	13
4.2 Výběr komponent .....	6	4.5.1 Log aplikace .....	13
4.2.1 Microcontroller .....	6	4.5.2 Konfigurace systému .....	14
4.2.2 LoRa transceiver .....	7	4.6 Koncová zařízení .....	17
4.2.3 RS485 transceiver .....	8	4.6.1 Zpracování dat jednotlivých typů koncových zařízení .....	17
4.3 Implementace LoRaWAN sítě ...	9	4.6.2 Přidávání koncových zařízení ze serveru K4 .....	19
4.4 Implementace komunikačního protokolu v síti IMA_RS485 .....	9	4.7 Využití non-volatile paměti gatewaye .....	19
		4.8 Zapojení .....	19
		4.9 Naprogramování .....	20

4.9.1 Zdrojové soubory projektu . . . 20

4.9.2 Nahrání programu do MCU . . 21

**Literatura**

**22**

## Obrázky

3.1 Blokový diagram funkce gatewaye	4
3.2 Příklad architektury přístupového systému [10] .....	5
4.1 Vývojový kit NUCLEO-L073RZ [1]	7
4.2 LoRa transceiver RFM95w [3] ...	7
4.3 RS485 transceiver [7] .....	8
4.4 foto zapojení .....	20

## Tabulky

4.1 Fyzické vlastnosti IMA_RS485 sítě	9
4.2 Syntaxe příkazu pro komunikaci v síti IMA_RS485 .....	10
4.3 Příklad sekvence příkazů odesílaných mezi masterem a slavem během předávání seznamu LoRaWAN device address koncových zařízení .	11
4.4 Příklad dat příkazu "průchod"odeslaného z gatewaye k masteru, obsahující data z koncových zařízení .....	12
4.5 Nastavení USB terminálu .....	13
4.6 Defaultní konfigurace systému ..	17
4.7 Typy koncových zařízení .....	17
4.8 Pinout připojení externích periférií k procesoru .....	20

## 0.1 Seznam zkratek

<b>AppSKey</b> .....	Application Session Key
<b>CPU</b> .....	Central Processing Unit
<b>CR</b> .....	Carriage Return
<b>CRC</b> .....	Cyclic Redundancy Check
<b>IoT</b> .....	Internet of Things
<b>LAN</b> .....	Local Area Network
<b>LF</b> .....	Line Feed
<b>LPWAN</b> .....	Low Power Wide Area Network
<b>LPWSN</b> .....	Low Power Wireless Sensor Network
<b>MCU</b> .....	Micro Controller Unit
<b>NwkSKey</b> .....	Network Session Key
<b>RF</b> .....	Radio Frequency
<b>SF</b> .....	Spreading Factor

## Kapitola 1

**Účel LPWSN v kontextu současného stavu  
ve světě**

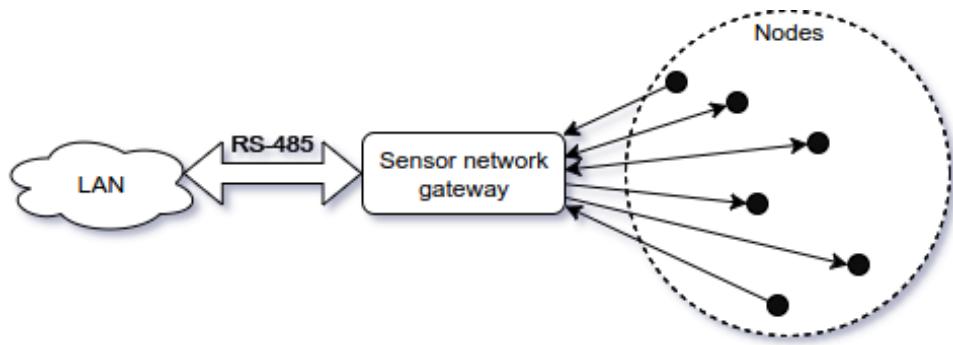
## Kapitola 2

Identifikace problému, který LPWSN řeší

## Kapitola 3

### Stanovení požadavků návrhu zařízení

Cílem tohoto projektu je návrh, realizace a otestování gatewaye, která shromažďuje data z bezdrátových koncových zařízení a přeposílá je přes RS485 LAN na PC master, který je dále přeposílá na IMA K4 server, kde jsou data zpracovávány.



Obrázek 3.1: Blokový diagram funkce gatewaye

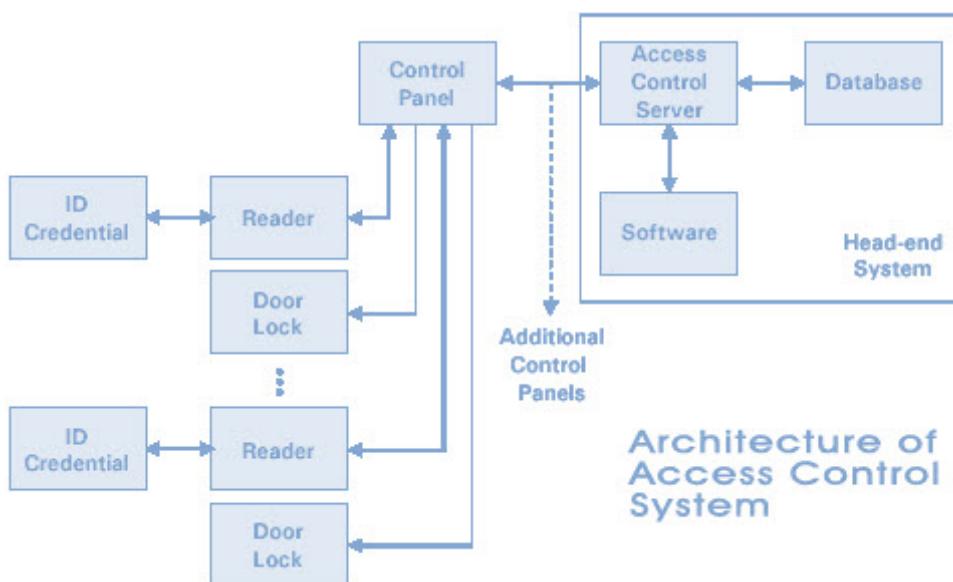
Předpokládá se, že koncová zařízení jsou senzory nebo aktuátory napájeny z baterie, tudíž pro jejich dlouhodobou životnost je kladen důraz na nízkou spotřebu vybrané bezdrátové technologie.

Drátová síť RS485, přes kterou gateway komunikuje s PC masterem používá síťový protokol původně navržen pro přístupové systémy. Rošiřování vlastností tohoto protokolu by znamenalo mnoho komplikací, cílem je tedy implementace IoT tak, aniž by bylo nutné protokol rozšiřovat.

#### 3.1 Přístupové systémy

Přístupové systémy jsou elektronické systémy řídící skrze síť přístup uživatelů do budov či objektů na základě ověření jejich identity [10]. V obrázku 3.2 je znázorněn příklad infrastruktury přístupového systému.

### 3.2. Implementace IoT do přístupového systému firmy IMA



Obrázek 3.2: Příklad architektury přístupového systému [10]

todo: popsát jednotlivé bloky obrazku

## 3.2 Implementace IoT do přístupového systému firmy IMA

V použitém systému firmy IMA je control panel nazýván PC Master, který komunikuje s čtečkami proprietárním protokolem přes rozhranní RS485. Účelem tohoto projektu je tedy nahrad

## Kapitola 4

### Realizace zařízení

Tato kapitola popisuje návrh gateweye neboli centrálního bodu LPWSN.

#### ■ 4.1 Výběr přenosové technologie

Pro tento systém je použita RF technologie LoRa se standardizovaným síťovým protokolem LoRaWAN, ale s požitím pouze jednoho kanálu. todo: popsat v cem se muj navrh lisi od standardu Jedenokanálové řešení bylo zvoleno z toho důvodu, že plnohodnotný LoRa transceiver, který přijímá na všech osmi kanálech je příliš drahy (přibližně desetinásobná cena) a složitý k implementaci, zatímco v tomto projektu je kláden důraz na cenu a jednoduchost řešení.

Vybraná technologie používá topologii typu hvězda, tedy koncová zařízení komunikují přímo s gateway, zbytek času mohou být ve stavu nízké spotřeby, což má za důsledek delší životnost baterie. Koncová zařízení od různých výrobců jsou plně kompatibilní s cizí gateway, tudíž není problém je implementovat do tohoto systému. Je pouze třeba je překonfigurovat pro vysílání na jednom použitém kanále.

#### ■ 4.2 Výběr komponent

##### ■ 4.2.1 Microcontroller

Pro toto zařízení je zvolen mikrocontroller STM32L073RZ se zaměřením na nízkou spotřebu, jelikož je levný, má dostačující vlastnosti a je dostupný ve formě vývojového kitu NUCLEO-L073RZ který byl použit pro vývoj zařízení. Mezi hlavní vlastnosti patří [1]:

- Architektura ARM Cortex-M0+ 32-bit RISC
- Interní Flash paměť 192 KB
- Interní SRAM paměť 20 KB
- Interní EEPROM paměť 6 KB
- Až 32 MHz CPU
- 2X SPI, 3x I2C, 4x USART, LIN, ADC

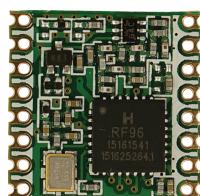
Pořizovací cena kitu přímo na stránce výrobce [www.st.com](http://www.st.com) je \$13 [1] [2].



**Obrázek 4.1:** Vývojový kit NUCLEO-L073RZ [1]

### 4.2.2 LoRa transceiver

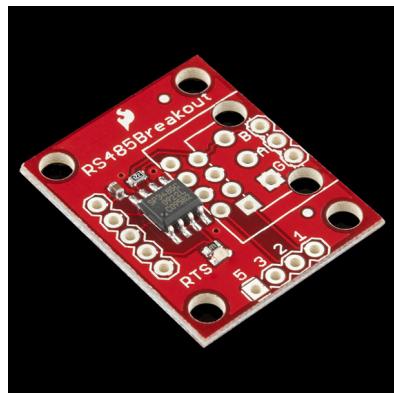
Lora transceiver čip doposud vyrábí pouze Semtech, pro použití v Evropském pásmu je určen typ SX1276. V tomto návrhu je použita deska RFM95w od firmy HopeRF s integrovaným čipem SX1276 [3]. Pro vývoj zařízení byl využit tento transceiver v tzv. Dragino LoRa Shield [4], který má stejně jako použitý vývojový kit, pinout kompatibilní s Arduino UNO. Pořizovací cena samotného transceiveru RFM95w je okolo \$7, cena Dragino Shieldu se pohybuje okolo \$22 na ebay.



**Obrázek 4.2:** LoRa transceiver RFM95w [3]

### 4.2.3 RS485 transceiver

SparkFun Transceiver Breakout - RS485 převádí rozhranní UART na RS485, pří vstupním napětí 3.3 V. A je dostupný za cenu okolo \$10 [7].



Obrázek 4.3: RS485 transceiver [7]

## 4.3 Implementace LoRaWAN sítě

Jednokanálové použití technologie LoRa umožňuje použít transceiver navržený pro koncová zařízení, kterým jsou pakety kontinuálně odposlouchávány na jednom nastaveném kanále a SF. Tyto dva parametry jsou nakonfigurovány na všech koncových zařízeních v síti.

Použitý protokol IMA\_RS485, původně navržen pro přístupové systémy, umožňuje posílat data pouze pomocí příkazu "průchod", který má kapacitu pro data z koncového zařízení pouze 6 B. Systém je proto navržen neobvyklým způsobem. Přijme-li gateway LoRaWAN paket, nejprve zkонтroluje zda zná adresu zařízení, pokud ano, přečte z paměti i typ zařízení, paket dešifruje a dekóduje payload, címž získá konečné hodnoty senzorů, které pak dále pošle přes RS485 rozhraní na master. LoRaWAN device address a typ každého zařízení v síti je uložena v EEPROM (non-volatile) paměti gatewaye a jsou nastavována na K4 serveru. Všechna zařízení mají nastavené stejné šifrovací klíče a gateway je má uložené v EEPROM.

Pro tento projekt byla vyvinuta knihovna pro dekódování payloadu na základě dokumentů [8] [9].

## 4.4 Implementace komunikačního protokolu v síti IMA\_RS485

V tomto projektu je komunikace protokolu IMA\_RS485 naprogramována v souborech rs485\_protocol.h a rs485\_protocol.c. Jedná se o kolizní protokol v síti, kde je připojen jeden master, a jeden nebo více slaveů řízených masterem.

Baud rate	9600
Data bits	8
Parity	none
Stop bits	1

**Tabulka 4.1:** Fyzické vlastnosti IMA\_RS485 sítě

### 4.4.1 Syntaxe příkazů

Komunikace v síti probíhá formou příkazů, které mají specifikovanou syntaxi v tabulce 4.2.

popis	adresa příjemce	adresa odesílatele	typ příkazu	délka dat	data	crc
počet bytů	1	1	1	2	délka dat	1

**Tabulka 4.2:** Syntaxe příkazu pro komunikaci v síti IMA\_RS485

Typy příkazu jsou zadefinované konstanty s předponou CKP\_CMD\_ v souboru ./Inc/rs485\_protocol.h. Příkazy odeslané masterem obsahují navíc synchronizační byte na začátku 0xAA. CRC je pro kontrolu XOR přes všechny předchozí byty v celém příkazu kromě synchronizačního bytu.

### 4.4.2 Adresace zařízení v síti

Každé zařízení na této sběrnici má svoji adresu, která mu je nastavena externě. Master má adresu 0xFF, adresa pro všechny (broadcast) je 0x00 a zařízení v této síti můžou mít adresu libovolnou (kromě těchto dvou) nesmí zde však být připojena 2 zařízení s nastavenou stejnou adresou.

### 4.4.3 Statusy

Slave má dva možné statusy, offline a online, což značí, zda je zařízení v aktivním režimu, tudíž má povolení odesílat příkaz průchod (V případě této gatewaye příkaz průchod obsahuje data ze koncového zařízení). Slave odesílá příkaz obsahující informaci o jeho statusu periodicky s typem příkazu 0x10 a jedním bytem dat označujícím status. Master na tento příkaz odpoví pouze pokud mění status slaveu. Pokud je slave online a master přestane komunikovat, zařízení slave se samo přepne na status offline.

#### Offline

Je-li zařízení zapnuto, je ve stavu offline. Nemá povoleno odesílat příkaz průchod obsahující data z koncových zařízení, pouze odpovídá na příkazy masteru. Odesílání příkazu signalizující tento stav je s periodou 10 s a obsahuje byte 0xEE.

#### Online

Odesílání příkazu signalizující tento stav je s periodou 45 s a obsahuje byte 0x00. V tomto stavu je povoleno odesílání příkazu průchod.

#### 4.4.4 Přidávání LoRaWAN zařízení do systému

Pokud master přijme od slavea příkaz oznamující že je ve stavu offline, nejprve slaveu pošle seznam LoRaWAN adres všech známých koncových zařízení a následně slavea přepne do stavu online. Přijímání seznamu adres je realizováno sekvencí příkazů typu 0x8F.

Níže v tabulce 4.3 je příklad sekvence příkazů odesílaných mezi masterem a slavem během předávání seznamu LoRaWAN device address koncových zařízení, kde slave má adresu 0x10 a master standardně 0xFF. Jak již bylo řečeno, příkazy od masteru lze jednoduše odlišit tím, že vždy začínají bytem 0xAA.

příkaz	data
master: start	AA 10 FF 8F 02 00 00 00 62
slave: ACK	FF 10 06 02 00 8F 00 64
master: data	AA 10 FF 8F 21 00 01 B1 C4 12 00 00 00 00 00 B2 C4 12 00 00 00 00 00 B3 C4 12 00 00 00 00 00 B4 C4 12 00 00 00 00 00 44
slave: ACK	FF 10 06 02 00 8F 01 65
master: data	AA 10 FF 8F 19 00 02 B5 C4 12 00 00 00 00 00 B6 C4 12 00 00 00 00 00 F6 1F 01 26 00 00 00 00 B6
slave: ACK	FF 10 06 02 00 8F 02 66
master: konec	AA 10 FF 8F 04 00 03 FF 2A 57 E5
slave: ACK	FF 10 06 02 00 8F 03 67

**Tabulka 4.3:** Příklad sekvence příkazů odesílaných mezi masterem a slavem během předávání seznamu LoRaWAN device address koncových zařízení

LoraWAN protokol používá 4-bytové adresy koncových zařízení. Adresy předávány touto sekvencí jsou dlouhé 8-bytové. První 4 byty je tedy LoRaWAN device address, pátý byte je typ zařízení a zbylé 3 byty jsou nevyužity, jejichž použití je možné v případě změn či rozšířování vlastností systému.

První Byte dat je counter paketu začínající od nuly, který označuje číslo odeslaného paketu v sekvenci. Na každý tento paket v sekvenci slave odpovídá ACK příkaz, který se liší od obyčejného ACK příkazu tím, že v datech paketu navíc obsahuje counter pakety v sekvenci. První příkaz této sekvence má délku dat 2 byty, které mají hodnotu 0x00 přičemž první je counter. Další příkazy hned za counter bytem obsahují několik osmibytových adres, jejichž počet je různý. Příkaz ukončující tuto sekvenci příkazů má délku 4 byty, což je tedy counter, 0xFF a 2 byty CRC přes všechny odeslané adresy (nepodstatné, tudíž ho nepoužívám).

### 4.4.5 Odesílání dat z koncových zařízení

Jak již bylo zmíněno, protokol byl navrhnut pro přístupové systémy, tudíž data z koncových zařízení jsou odesílána příkazem "průchod", jehož typ je 0x10 a kapacita na data z koncového zařízení je pouze 6 B.

První byte dat označuje typ průchodu, byl zvolen konstantní byte 0xD0. Dále následuje LoRaWAN adresa koncového zařízení od kterého byl paket přijat. Dále následují 4 byty dat ze senzoru, další 2 byty signalizující čas průchodu, což v tomto projektu není použito a tyto dva byty mají vždy hodnotu 0xFF. A nakonec jsou další 2 byty obsahující data ze senzoru.

Příklad příkazu: FF 1F 10 0D 00 D0 F6 1F 01 29 AD 0A 5A 27 FF FF DE 09 E1.

Data příkazu jsou níže rozepsána v tabulce 4.4.

typ průchodu	LoRaWAN device address	data (4B)	cas	data (2B)
D0	F6 1F 01 29	AD 0A 5A 27	FF FF	DE 09

**Tabulka 4.4:** Příklad dat příkazu "průchod" odeslaného z gatewaye k masteru, obsahující data z koncových zařízení

Master na příkaz "průchod" odpovídá příkazem ACK. Slave na tuto odpověď čeká standardně 3 sekundy, ale tento parametr je nastavitelný. Pokud v tomto timeoutu master neodpoví, slave příkaz zopakuje a změní typ příkazu na 0x20. Pokud master ani na třetí opakování neodpoví ACK, slave se přepne do stavu offline a vymaže frontu příkazů "průchod" k odeslání.

### 4.4.6 Potvrzení

Slave odpovídá na každý příkaz od masteru ACK. Typ příkazu ACK je 0x06 a data příkazu obsahují jeden byte signalizující typ příkazu na který je právě odpovídáno potvrzením. Master odpovídá ACK se stejným typem příkazu 0x06, ale s žádnými daty příkazu.

### 4.4.7 Dotaz na příznaky

Master se může zeptat s jak dlouhými adresami slave pracuje s typem příkazu 0x49. Slave na to odpovídá ACK s tím, že v datech příkazu je navíc byte 0x04. Master pak počítá s tím, že slave pracuje se 64-bit adresami (ve skutečnosti ale používá 32-bitové a zbylé 4 byty v příkazu průchod jsou pro data z koncového zařízení).

## 4.5 Komunikace přes USB

Gateway má implementovanou komunikaci přes USB, což má za účel konfiguraci systému a logování. K připojení přes USB lze použít PC s aplikací terminálu s nastavením viz tabulka 4.5.

Baud rate	115200
Data bits	8
Parity	none
Stop bits	1
Flow control	none

**Tabulka 4.5:** Nastavení USB terminálu

Při komunikaci jsou data standardně oddělována bytem CR (carriage return) 0x0D, ale je akceptována i sekvence CR LF (Line Feed), tedy 0x0D 0x0A.

### 4.5.1 Log aplikace

Gateway loguje informace o proběhlých událostech přes USB. Níže je příklad výpisu dat pro případ, že gateway přjala LoRaWAN paket z koncového zařízení v síti.

Nejprve jsou vypsána data týkající se LoRaWAN protokolu, zašifrovaný i dešifrovaný payload, typ zařízení a konečné informace dekódované z payloadu. Řádek začínající předponou "Tx -> RS-485:" obsahuje data odeslané k masteru přes RS485 síť a následující řádek začínající předponou "Rx -> RS-485:" obsahuje odpověď od masteru.

---

```
Rx -> LoRaWAN, pktCntr: 6
RSSI: -51, SNR: 9, length: 22

Message type: Unconfirmed Data Up
Packet rawData: "40F61F0128C0D62508D970CB071595D115BAC68F6663"
Device Address: "F61F0128"
FCnt: 9686
message (encrypted): "D970CB071595D115BA"
MHDR: 40; FCtrl: C0; FPort: 08; MIC: "C68F6663"
adaptive data rate: true; ack: false
message HEX (decrypted): "013566779600FFFFAF"

Sensor type: RHF1S001
temperature: 23.30 C, humidity: 52 %
period: 300 s, RSSI: -51 dBm, SNR: 9 dB, battery voltage: 3.2 V
Tx -> RS-485: "FF1F100D00D0F61F01281A0934CDFFF09202E"
```

---

Rx -> RS-485: "AA1FFF060000E6"  
ACK

---

### 4.5.2 Konfigurace systému

Konfigurace gatewaye se provádí opět přes USB port. Je do ní vstoupeno odesláním příkazu "config", následuje vypsání současného stavu konfigurace a dále je vypsáno konfigurační menu, kde uživatel vybere jednotlivý bod menu zadáním jeho čísla na začátku řádku. Níže je zobrazen příklad výpisu po vstupu do konfigurace.

---

-----Entering configuration setup-----

System configuration:

\*\*\* LoRa channel:  
channel: 0 (868.1 Mhz)  
SF7

\*\*\* RS485 channel:  
my address: 10  
master address: FF  
timeout: 3 s

\*\*\* LoRaWAN keys:  
NwSKey: FD 90 0D 8C 70 9F 19 24 18 EC FD D4 28 0C AC 47  
AppSKey: 68 9F D0 AC 7A 0F 95 58 B1 19 A0 16 17 F4 16 33

Config menu:  
1 -> Config LoRa channel  
2 -> Config RS485 channel  
3 -> Config LoRaWAN protocol  
4 -> Print all LoRaWAN devices  
5 -> Erase all LoRaWAN devices  
6 -> Restore to default configuration  
7 -> Exit without save  
8 -> Save and exit

---

Z konfigurace je možné vystoupit kdykoliv bez uložení změn příkazem "quit". Při vstoupení do konfigurace je pozastavena činnost gatewaye, komunikace s koncovými zařízeními LoRaWAN sítě a komunikace s masterem v síti RS485 nejsou aktivní. Jsou zde tedy 3 stavy konfigurace, přičemž je možné vždy jednotlivá nastavení přeskakovat odesláním "prázdného příkazu" 0x0D (v terminálu obvykle stačí stisknout Enter). Systém při konfiguraci vždy vypíše jaká data mají být zadána v jakém tvaru a zároveň současnou hodnotu měněného parametru. Zadaná data uživatelem jsou vždy zkонтrolována zda

splňují požadovaný tvar. Pokud ne, uživatel je o tom informován a vyzván k dalšímu pokusu. Po provedení konfigurace následuje vždy návrat zpět do hlavního menu. Pro uložení nové konfigurace je potřeba v menu vybrat "Save and exit", gateway pak následně vypíše které parametry byly změněny a provede restart.

### **■ Config LoRa channel**

Konfigurace LoRa RF kanálu zahrnuje nastavení SF a frekvenční pásmo. Níže je příklad konfigurace.

---

```
LoRa channel configuration:
Enter SF number (7-12)
(current: 7)
8
SF8 set.

Enter LoRa channel number (0-7)
ch0 is 868.1 Mhz
ch1 is 868.3 Mhz
ch2 is 868.5 Mhz
ch3 is 867.1 Mhz
ch4 is 867.3 Mhz
ch5 is 867.5 Mhz
ch6 is 867.7 Mhz
ch7 is 869.0 Mhz
(current: 0)
1
channel 1 set.
```

---

### **■ Config RS485 channel**

Konfigurace RS485 kanálu pro komunikaci s masterem zahrnuje nastavení adresy tohoto zařízení, adresa masteru a timeout, což je doba čekání na potvrzení od masteru po odeslání příkazu "průchod". Níže je příklad konfigurace.

---

```
RS485 channel configuration:
Enter address of this device, FF and 00 are reserved.
(current: 10)
11
Address of this device is set to: 11

Enter master address:
(current: FF)
FE
Master address is set to: FE

Enter timeout (seconds)
```

```
(current: 3)
5
timeout set to: 5 s
```

---

### ■ Config LoRaWAN protocol

Konfigurace LoRaWAN protokolu zahrnuje nastavení šifrovacích klíčů NwkSKey a AppSKey. Níže je příklad konfigurace.

```
LoRaWAN protocol configuration:
Enter NwkSKey (16 bytes in HEX)
(current: FD 90 0D 8C 70 9F 19 24 18 EC FD D4 28 0C AC 47)
11111112222222333333344444444
NwSKey set to: 11 11 11 11 22 22 22 22 33 33 33 33 44 44 44 44

Enter AppSKey (16 bytes in HEX)
(current: 68 9F D0 AC 7A 0F 95 58 B1 19 A0 16 17 F4 16 33)
11111112222222333333344444444
AppSKey set to: 11 11 11 11 22 22 22 22 33 33 33 33 44 44 44 44
```

---

### ■ Print all LoRaWAN devices

Vypíše všechna LoRaWAN zařízení uložená v paměti. Níže je příklad.

```
number.....0:
Device Address: B1 C4 12 00
Device Type: RH1S001
number.....1:
Device Address: B2 C4 12 00
Device Type: RH1S001
number.....2:
Device Address: B3 C4 12 00
Device Type: RH1S001
number.....3:
Device Address: B4 C4 12 00
Device Type: IMA_tempPress
number.....4:
Device Address: B5 C4 12 00
Device Type: IMA_tempPress
```

---

### ■ Restore default configuration

Po zvolení této možnosti je načtena defaultní konfigurace systému, která obsahuje hodnoty viz tabulka 4.6. Tyto defaultní hodnoty jsou nastaveny v programu a slouží především pro testovací účely.

popis	hodnota
RS485 myAddr	0x10
RS485 MasterAddr	0xFF
RS485 timeout	3
LoRa SF	SF7
LoRa channel	0 (868.1 Mhz)
NwSKey	FD 90 0D 8C 70 9F 19 24 18 EC FD D4 28 0C AC 47
AppSKey	68 9F D0 AC 7A 0F 95 58 B1 19 A0 16 17 F4 16 33

**Tabulka 4.6:** Defaultní konfigurace systému

## 4.6 Koncová zařízení

Jelikož používaný protokol ke komunikaci s masterem je omezen na pouhých 6 B na jeden paket, payload koncových zařízení je dekódován v gatewayi a v paketu odeslaném na master jsou pouze vybraná nejdůležitější data, která se vejdu do této velikosti. Prote společně s LoRaWAN device address koncového zařízení je v gatewayi uložen i typ zařízení zadefinován jedním bytem a na základě typu zařízení gateway rozpozná jak dekódovat payload.

Momentálně jsou podporovány dva typy koncových zařízení, dle potřeby je možné rozšířit FW gatewaye o další typy koncových zařízení.

Typ zařízení	Hodnota
RHF1S001	0x00
IMA_tempPress	0x01

**Tabulka 4.7:** Typy koncových zařízení

### 4.6.1 Zpracování dat jednotlivých typů koncových zařízení

Níže je popsáno pro jednotlivá podporovaná koncová zařízení jak jsou data uložena v datové struktuře, jak jsou data z této struktury zpracována a zobrazena a nakonec jak vybraná data jsou zapsána do výsledného bufferu o délce 6 B, který je odeslán do masteru příkazem "průchod".

#### RHF1S001

Senzor od firmy RisingHF měří teplotu a vlhkost.

```

1  /* RHF1S001 data structure */
2  typedef struct {
3      int16_t temperature;
4      uint8_t humidity;
5      uint16_t period;
6      int8_t rssi ;
7      int8_t snr;
```

```

8     uint8_t battery;
9 } RHF1S001_data_t;
10
11 /* Print the data from the structure */
12 printf("temperature: %d.%d C, ", RHF1S001_data.temperature / 100,
13     RHF1S001_data.temperature % 100);
14 printf("humidity: %d %%\n", RHF1S001_data.humidity);
15 printf("period: %d s, ", (int)RHF1S001_data.period);
16 printf("RSSI: %d dBm, ", RHF1S001_data.rssi);
17 printf("SNR: %d dB, ", RHF1S001_data.snr);
18 printf("battery voltage: %d.%d V\r\n", RHF1S001_data.battery/10,
19     RHF1S001_data.battery % 10);
20
21 /* Put the data into 6B long buffer, that is transmitted to the master */
22 buffer [0] = RHF1S001_data.temperature & 0xFF;
23 buffer [1] = RHF1S001_data.temperature >> 8;
24 buffer [2] = RHF1S001_data.humidity;
25 buffer [3] = RHF1S001_data.rssi;
26 buffer [4] = RHF1S001_data.snr;
27 buffer [5] = RHF1S001_data.battery;

```

## IMA\_tempPress

Senzor vytvořený ve firmě IMA, měřící teplotu a tlak.

```

1 /* IMA_tempPress data structure */
2 typedef struct {
3     int16_t temperature;
4     uint16_t pressure;
5     int8_t rssi ;
6     int8_t snr;
7 } IMA_tempPress_data_t;
8
9 /* print the data from the structure */
10 printf("temperature: %d.%d C, ", IMA_tempPress_data.temperature / 100,
11     IMA_tempPress_data.temperature % 100);
12 printf("pressure: %d.%d Pa\r\n", IMA_tempPress_data.pressure/10,
13     IMA_tempPress_data.pressure % 10);
14 printf("RSSI: %d dBm, SNR: %d dB\r\n", IMA_tempPress_data.rssi,
15     IMA_tempPress_data.snr);
16
17 /* Put the data into 6B long buffer, that is transmitted to the K4 server */
18 buffer [0] = IMA_tempPress_data.temperature & 0xFF;
19 buffer [1] = IMA_tempPress_data.temperature >> 8;
20 buffer [2] = IMA_tempPress_data.pressure & 0xFF;
21 buffer [3] = IMA_tempPress_data.pressure >> 8;
22 buffer [4] = IMA_tempPress_data.rssi;
23 buffer [5] = IMA_tempPress_data.snr;

```

### 4.6.2 Přidávání koncových zařízení ze serveru K4

Koncová zařízení síť se nastavují ze serveru K4 v podobě seznamu offline karet s délkou UID 8 B. LoRaWAN device address je dlouhá 4 B, jeden byte je navíc použit pro typ koncového zařízení, zbylé 3 byty jsou nuly. Jelikož typ zařízení je uložen v gatewayi i na K4 serveru. Při odesílání příkazu průchod se tedy už typ zařízení neposílá z důvodu datového omezení tohoto příkazu. Na serveru K4 se UID nastavuje jako dekadické číslo. Níže je příklad vytvoření výsledného čísla obsahující DevAddr a typ zařízení, které se zadává do K4 serveru.

#### Příklad

Pro případ, kde typ zařízení je 01 a DevAddr AABBCCDD (little endian) výsledné číslo v hexadecimální podobě je 01DDCCBAA. Následně se překládá do decimalní podoby, výsledné číslo k zadání do K4 serveru je tedy 8016149418.

## 4.7 Využití non-volatile paměti gatewaye

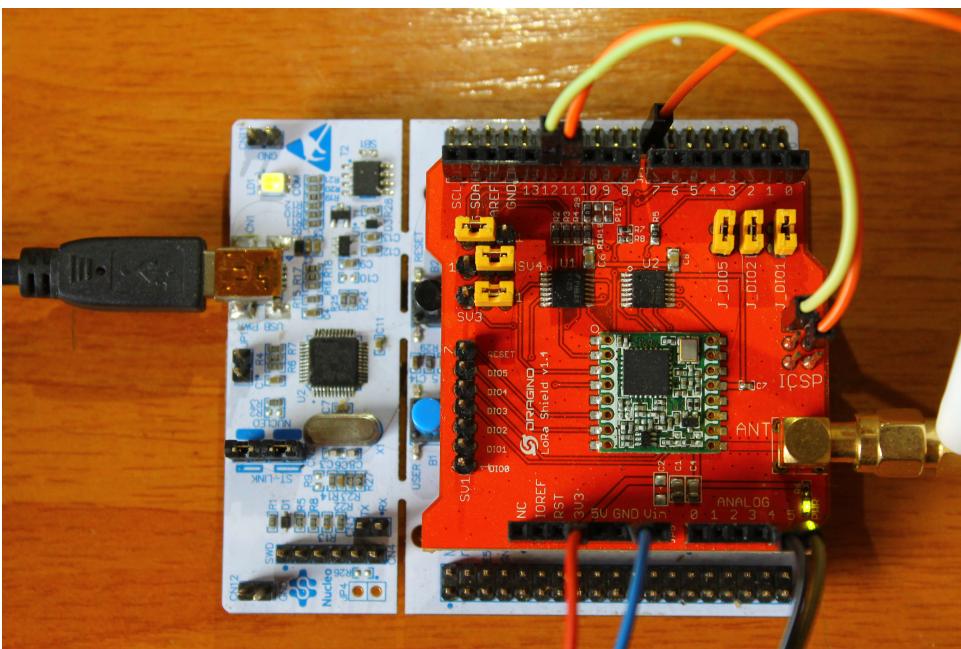
Konfigurace a adresy s typy všech koncových zařízení v LoRaWAN síti jsou uloženy v non-volatile paměti EEPROM gatewaye o kapacitě 6144 B. Paměť je tedy rozdělená tak, že od adresy 0 až po 6080 je prostor pro ukládání LoRaWAN zařízení a od 6080 až po 6144 je prostor pro ukládání konfigurace gatewaye.

Každé LoRaWAN zařízení v síti má v paměti uložené LoRaWAN device address (4 byty), typ zařízení (1 byte) a další 3 byty jsou rezervovány. Jedno koncové zařízení v paměti tedy zabírá 8 B, takže gateway má kapacitu paměti pro až 760 koncových zařízení.

## 4.8 Zapojení

LoRa shield [4] je nasazen přímo na vývojový kit Nukleo. Kit neobsahuje ISCP konektor, který je součástí pinoutu Arduino UNO a LoRa shield má SPI piny MISO a MOSI přivedeny právě na tento konektor. Musí být tedy propojeny externě viz obrázek 4.4. Jumpery na Dragino LoRa shieldu musí také být stejně jako v obrázku.

Pro komunikaci s LoRa transceiverem je tedy použito SPI1, pro komunikaci přes USB je použito USART2 a pro komunikaci přes RS485 je použito UART1.



Obrázek 4.4: foto zapojení

Periférie	Název pinu	Pin procesoru
RS485 transceiver	RX	PC1
	TX	PC0
	RTS	PB1
LoRa transceiver	CS	PB6
	CLK	PA5
	MISO	PA6
	MOSI	PA7
	RST	PC7
	DIO0	PA10

Tabulka 4.8: Pinout připojení externích periférií k procesoru

## 4.9 Naprogramování

K naprogramování MCU byla použita HAL knihovna a inicializační nástroj STM32CubeMX poskytnuté výrobcem, tedy ST Microelectronics. Zdrojové soubory programu byly vyvíjeny v textovém editoru VS-Code, ke komplikaci zdrojových souborů byl použit kompilátor arm-none-eabi-gcc a jako pomocný nástroj makefile skript.

### 4.9.1 Zdrojové soubory projektu

Pro šifrování LoRaWAN paketu byla použita knihovna AES-128, dostupná na githubu [5] a knihovna OpenPANA také dostupná z githubu [6]. Níže je

seznam zdrojových souborů.

Drivers .....	STM32 Drivers
Inc .....	Headers
aes.h .....	AES-128 library for LoRaWAN paket encryption
cmac.h .....	library for CMAC calculation in LoRaWAN protocol
LinkedList_ByteArray.h ..	Byte array linked list library for stacks
LoRaWAN_paket.h.....	LoRaWAN library for paket data decoding
stm32l0xx_hal_conf.h.....	HAL initialization of peripherals
ByteArray.h.....	Library for Byte array operations
LoRa.h .....	Library for interfacing LoRa transceiver
main.h .....	Main file
stm32l0xx_it.h .....	HAL initialization of peripherals
EEPROM.h.....	Library for eeprom operations
LoRa_sensors.h.....	Library for decoding data from payload
rs485_protocol.h .....	Library for RS485 IMA protocol
usb.h ...	Library for USB communication and system configuration
Src.....	Sources
aes.c .....	source file to the aes.h
aes.c .....	source file to the cmac.h
LinkedList_ByteArray.c	source file to the LinkedList_ByteArray.h
LoRaWAN_paket.c .....	source file to the LoRaWAN_paket.h
stm32l0xx_hal_msp.c .....	HAL source file
ByteArray.c .....	source file to the ByteArray.h
LoRa.c .....	source file to the LoRa.h
main.c .....	main source file
stm32l0xx_it.c .....	HAL source file
EEPROM.c .....	source file to the EEPROM.h
LoRa_sensors.c .....	source file to the LoRa_sensors.h
rs485_protocol.c .....	source file to the rs485_protocol.h
system_stm32l0xx.c .....	HAL source file
usb.h .....	source file to the usb.h

## 4.9.2 Nahrání programu do MCU

Výstupem komplikace je soubor s koncovkou .binary, který je nahrán do MCU. K tomuto nahrání není potřeba žádný speciální SW nebo HW. Stačí kit připojit k PC přes USB, v PC se kit zobrazí jako flash disk. Zkomplikovaný program s koncovkou .binary stačí překopírovat na toto zařízení. Po dobu kopírování souboru bliká na kitu LED1 červená/zelená. Jakmile kopírování skončí, program na kitu je spuštěn, případně je možné kit resetovat černým tlačítkem reset. Pro uvedení Gateweye do provozu je nutné se připojit k zařízení přes USB a nastavit všechny parametry viz sekce 4.5.2.

## Literatura

- [1] *NUCLEO-L073RZ*. ST Microelectronics [Online]. Available: <https://www.st.com/en/evaluation-tools/nucleo-l073rz.html> [Accessed: 20-Sep-2019].
- [2] *NUCLEO-L073RZ* ARM Mbed. [Online]. Available: <https://os.mbed.com/platforms/ST-Nucleo-L073RZ/> [Accessed: 20-Sep-2019].
- [3] *RFM95/96/97/98(W) - Low Power Long Range Transceiver Module*. HopeRF electronic. V1.0. [Online]. Available: [http://wiki.dragino.com/index.php?title=Lora\\_Shield](http://wiki.dragino.com/index.php?title=Lora_Shield) [Accessed: 20-Sep-2019].
- [4] *Lora Shield*. Dragino. [Online]. Available: [http://wiki.dragino.com/index.php?title=Lora\\_Shield](http://wiki.dragino.com/index.php?title=Lora_Shield) [Accessed: 20-Sep-2019].
- [5] *tiny-AES128-C* bitdust. [Online]. Available: <https://github.com/bitdust/tiny-AES128-C> [Accessed: 20-Sep-2019].
- [6] *openpana*. OpenPANA. [Online]. Available: <https://github.com/OpenPANA/openpana> [Accessed: 20-Sep-2019].
- [7] *SparkFun Transceiver Breakout - RS-485* Sparkfun. [Online]. Available: <https://www.sparkfun.com/products/10124> [Accessed: 20-Sep-2019].
- [8] *LoRaWAN Specification*. LoRa Alliance. v1.1. Sparkfun. [Online]. Available: <https://lora-alliance.org/resource-hub/lorawan-specification-v11> [Accessed: 20-Sep-2019].
- [9] Robert Miller. *LoRa Security Building a Secure LoRa Solution*. MWR Labs Whitepaper. [Online]. Available: <https://labs.mwrinfosecurity.com/assets/BlogFiles/mwri-LoRa-security-guide-1.2-2016-03-22.pdf> [Accessed: 20-Sep-2019].
- [10] [Online]. Available: <https://www.elprocus.com/understanding-about-types-of-access-control-systems/> [Accessed: 9-Sep-2019].