



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	2.4.4 Přidávání LoRaWAN zařízení do systému	7
1 Stanovení požadavků systému	2	2.4.5 Odesílání dat ze senzoru	8
2 Realizace gatewaye	3	2.4.6 Dotaz na příznaky	8
2.1 Výběr přenosové technologie	3	2.5 Paměť	9
2.2 Komponenty	3	2.6 Komunikace přes USB	9
2.2.1 NUCLEO-L073RZ	3	2.6.1 Log aplikace	9
2.2.2 RFM95w	4	2.6.2 Konfigurace systému	10
2.2.3 SparkFun Transceiver Breakout - RS485	4	2.7 Koncová zařízení	14
2.3 Implementace LoRaWAN protokolu	5	2.7.1 Zpracování dat jednotlivých typů koncových zařízení	14
2.3.1 Zabezpečení protokolu LoRaWAN	5	2.7.2 Přidávání koncových zařízení ze serveru K4	15
2.4 Implementace protokolu IMA RS485	6	2.8 Zapojení	17
2.4.1 Syntaxe příkazů	6	2.9 Instalace	18
2.4.2 Statusy	6	2.10 Zdrojové soubory projektu	19
2.4.3 ACK neboli potvrzení	7	Literatura	20

Obrázky

1.1 Blokový diagram navrženého systému	2
2.1 Vývojový kit NUCLEO-L073RZ [1]	4
2.2 LoRa transceiver RFM95w [3]	4
2.3 RS485 transceiver [7]	5
2.4 foto zapojení	17

Tabulky

2.1 Syntaxe příkazu pro komunikaci se serverem přes RS485	6
2.2 Příklad sekvence příkazů pro přidávání LoRaWAN zařízení do systému	8
2.3 Nastavení USB terminálu	9
2.4 Defaultní konfigurace systému ..	13
2.5 Typy koncových zařízení	14
2.6 Pinout připojení externích periférií k procesoru	17

■ 0.1 Seznam zkratek

LPWAN	Low Power Wide Area Network
.....	(LAN) Local Area Network
.....	(RF) Radio Frequency

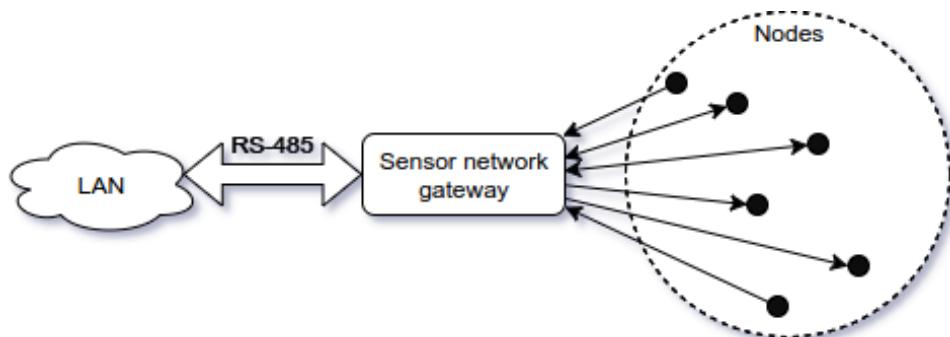
Kapitola 1

Stanovení požadavků systému

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.

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 se PC masterem používá síťový protokol původně navržen pro přístupové systémy.



Obrázek 1.1: Blokový diagram navrženého systému

Kapitola 2

Realizace gatewaye

2.1 Výběr přenosové technologie

Pro tento systém je použita RF technologie LoRa se standardizovaným síťovým protokolem LoRaWAN, ale s požitím pouze jednoho kanálu. Jedenokanálové řešení bylo zvoleno z toho důvodu, že plnohodnotný LoRa transceiver, který přijímá na všech osmi kanálech je příliš drahý (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 gatewayí, zbýtek času mohou být ve stavu nízké spotřeby, což má za důsledek delší životnost baterie. Koncová zařízení od různých výrobců jsou plně kompatibilní s cizí gatewayí, tudíž není problém je implementovat do tohoto systému. Je pouze třeba je překonfigurovat pro vysílání na jednom použitém kanále.

2.2 Komponenty

todo

Celé zařízení se skládá z vývojového kitu NUCLEO-L073RZ, LoRa transceiveru RFM95W a převodník UART na RS485. Kit má kompatibilní pinout s Arduino UNO.

2.2.1 NUCLEO-L073RZ

Kit obsahuje 32-bitový procesor STM32L073RZT6 architektury ARM Cortex-M0+, zaměřující se na nízkou spotřebu. Pro tento projekt je jeho

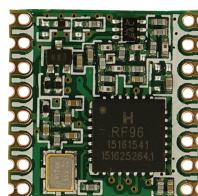
clock nakonfigurován na co nejvyšší hodnotu, tedy 32 MHz. Pořizovací cena kitu přímo na stránce výrobce www.st.com je \$13 [1] [2]. Kit má 2 tlačítka. Černé pro reset a modré pro libovolné použití (user button).



Obrázek 2.1: Vývojový kit NUCLEO-L073RZ [1]

2.2.2 RFM95w

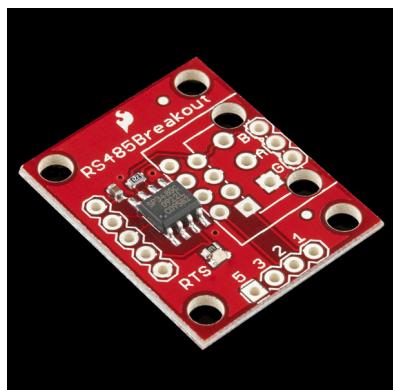
LoRa transceiver od firmy HopeRF založeném na chipu SX1276 od firmy Semtech používá SPI [3]. Pro vývoj zařízení byl využit tento transceiver v tzv. Dragino LoRa Shield [4], který má stejně jako vývojový kit, pinout kompatibilní s Arduino UNO. Pořizovací cena samotného modulu RFM95w je okolo \$7, cena Dragino Shieldu se pohybuje okolo \$22 na ebay.



Obrázek 2.2: LoRa transceiver RFM95w [3]

2.2.3 SparkFun Transceiver Breakout - RS485

Tento převodník převádí RS485 na UART, napěťové úrovně 3.3 V. A je dostupný za cenu okolo \$10 [7].



Obrázek 2.3: RS485 transceiver [7]

2.3 Implementace LoRaWAN protokolu

Pro jednoduchost a nízkou cenu řešení je zde použita technologie LoRaWAN pouze na jednom kanále a jednom SF(Spraying factor). Všechna zařízení v síti tedy musí mít tyto dva parametry nastavené stejně. Defaultně je použito 868.1 MHz a SF7.

Gateway má trvale ve své non-volatile paměti uložené adresy, a typy všech LoRaWAN zařízení v síti. A oba šifrovací klíče NwSKey a AppSKey, které jsou pro celou síť stejné.

Přijme-li gateway LoRaWAN packet, nejprve zkонтroluje zda zná adresu zařízení, pokud ano, přečte z paměti i typ zařízení, packet dešifruje a dekóduje payload, čímž získá konečné hodnoty senzorů, které pak dále pošle přes RS485 rozhraní na server.

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

2.3.1 Zabezpečení protokolu LoRaWAN

Protokol používá AES-128 na 2 způsoby. Pro bezpečné dekódování packetu jsou tedy požity 2 128-bit šifrovací klíče.

Aplikační zabezpečení

Aplikační klíč (Application Session Key) je použit pro zašifrování aplikační zprávy (App message). Tímto je zabráněno přečíst zprávu komukoliv kdo packet příjme

■ Síťové zabezpečení

Síťové zabezpečení je zde aby bylo hackerům zabráněno odesílání duplikovaných paketů nebo paketů s neplatnými daty. LoRaWAN paket obsahuje packet counter počítající od nuly od doby kdy bylo LoRaWAN zařízení spuštěno. Toto umožňuje odhalit duplikování paketu. Poslední 4 byty paketu obsahují MIC (Message Integrity Code), který je získán zašifrováním dat síťovým klíčem (Network Session Key) obsahujících mimo jiné celý payload paketu (včetně packet counter). Toto umožňuje odhalit jakoukoliv manipulaci s daty v paketu [8] [9].

■ 2.4 Implementace protokolu IMA RS485

Každé zařízení na této sběrnici má svoji adresu, která mu je nastavena externě. Server má adresu 0xFF, adresa pro všechny (broadcast) je 0x00 a zařízení v této síti můžou mít adresu libovolnou (krom těchto dvou). V tomto projektu je tento protokol naprogramován v souborech rs485_protocol.h a rs485_protocol.c.

■ 2.4.1 Syntaxe příkazů

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

Tabulka 2.1: Syntaxe příkazu pro komunikaci se serverem přes RS485

Veškeré typy příkazu jsou zadefinované konstanty s předponou CKP_CMD_ v souboru ./Inc/rs485_protocol.h.

Příkazy odeslané ze serveru obsahují navíc synchronizační byte na začátku 0xAA. crc je pro kontrolu XOR přes všechny předchozí byty v celém příkazu kromě synchronizačního bytu.

■ 2.4.2 Statusy

Zařízení má dva možné statusy, offline a online. Status zařízení je odesíán periodicky s typem příkazu 0x10 a jedním bajtem dat označujícím status. Server na tento příkaz neodpovídá.

■ Offline

Je-li zařízení zapnuto, je ve stavu offline. Nemá povolené odesílat data ze senzorů, pouze odpovídá na příkazy serveru a následně dostane příkaz od

serveru pro přechod do stavu online. Odesílání příkazu signalizující tento stav je s periodou 30 s a obsahuje bajt 0xEE.

■ Online

V tomto stavu je povoleno odesílání dat ze senzorů. Odesílání příkazu signalizující tento stav je s periodou 45 s a obsahuje bajt 0x00.

■ 2.4.3 ACK neboli potvrzení

Zařízení odpovídá na každý platný příkaz od serveru ACK. Typ příkazu ACK je 0x06 a v datech je jeden byte, což je typ příkazu na který právě odpovídá potvrzením.

Server odpovídá ACK se stejným typem příkazu 0x06, ale s žádnými daty v příkazu. Délka dat je tedy 0.

■ 2.4.4 Přidávání LoRaWAN zařízení do systému

Pokud server dostane od zařízení příkaz oznamující že je ve stavu offline, nejprve tomu zařízení v RS485 síti pošle seznam adres LoRaWAN modulů v síti a pak ho přepne do stavu online.

Přijímání seznamu adres je tvořeno sekvencí příkazů typu 0x8F.

První Byte dat je counter packetu začínající od nuly, který označuje číslo odeslaného packetu v sekvenci. Na každý tento packet v sekvenci zařízení odpoví ACK příkaz, který se liší od obyčejného ACK příkazu tím, že v datech packetu navíc obsahuje counter packety 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.

LoraWAN používá 4-bytové devAddr. První 4 byty jsou devAddr, páty byte je typ zařízení a ostatní zatím nejsou použity. Momentálně je zaveden pouze jeden typ zařízení a to je RH1S001, pro nějž hodnota tohoto bytu je 0.

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).

V tabulce 2.2 je pro názornost příklad sekvence příkazů. Jak již bylo řečeno, příkazy od serveru lze jednoduše odlišit tím, že vždy začínají bytem 0xAA.

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

Tabulka 2.2: Příklad sekvence příkazů pro přidávání LoRaWAN zařízení do systému

2.4.5 Odesílání dat ze senzoru

Jak již bylo zmíněno, protokol byl navrhnut pro přístupové systémy a nedošlo k žádné úpravě pro tuto odlišnou aplikaci. Data ze senzorů se tedy posílají s použitím stávajícího příkazu "průchod". Typ tohoto příkazu je 0x10. První byte dat označuje typ průchodu, byl zvolen konstantní byte 0xD0. Dále následuje LoRaWAN adresa modulu od kterého byl přijat packet. Dále následují 4 byty dat ze senzoru. Dále 2 byty oznamující čas průchodu, což v tomto projektu není použito a tyto dva byty mají tedy hodnotu 0xFF. A nakonec jsou další 2 byty obsahující data ze senzoru. Celý tento příkaz průchod tedy obsahuje pouze 6 bytů pro data ze senzoru.

Server na tento příkaz odpovídá ACK a má čas na odpověď standardně 3 sekundy, ale tento parametr je nastavitelný. Pokud v tomto timeoutu neodpoví, zařízení příkaz zopakuje a změní typ příkazu na 0x20. Pokud server ani na třetí opakování neodpoví ACK, zařízení se přepne do stavu offline a vymaže frontu příkazů k odeslání.

2.4.6 Dotaz na příznaky

Server se může zeptat s jak dlouhým adresami zařízení pracuje. Je to typ příkazu 0x49 a délka dat je 0. Zařízení na to odpoví ACK, ale navíc je v datech příkazu byte 0x04 a server pak počítá s tím, že pracuje se 64-bit adresami (ve skutečnosti ale používám 32-bitové a zbylé 4 byty jsou pro data).

2.5 Paměť

Konfigurace systému, DevAddr a typ všech zařízení v síti jsou uložena v non-volatile paměti procesoru EEPROM, která je o velikosti 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 systému.

Každé LoRaWan zařízení v síti má v paměti uložené device address (4 byty), typ zařízení (1 byte) a další 3 byty jsou rezervovány. Jedno zařízení zabírá tedy 8 B, takže jich pro jednu síť může být až 760.

2.6 Komunikace přes USB

Komunikace přes USB s PC má za účel konfiguraci systému a Log aplikace. Lze k tomu použít libovolný terminál s nastavením viz tabulka 2.2.

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

Tabulka 2.3: Nastavení USB terminálu

Při komunikaci jsou data oddělována bytem 0x0D, ale je akceptována i sekvence 0x0D 0x0A.

2.6.1 Log aplikace

Gateway odesílá informace přes USB o tom, co právě provádí. Níže je příklad výpisu dat pro případ, že gateway přijme packet. Nejprve jsou vypsána data týkající se LoRaWAN protokolu, DevAddr bylo rozpoznáno, payload byl dešifrován a na základě typu senzoru byl dekódován payload a vyčteny hodnoty senzorů LoRaWAN zařízení. Poslední řádek obsahuje data odeslané přes RS485 na server.

```
Rx -> LoRaWAN, pktCntr: 406
RSSI: -29, SNR: 9, length: 22
Packet rawData: 40 F6 1F 01 26 C0 A1 30 08 D4 5D 93 F0 F0 F6 60 C0
               04 BC BE 4B 24
devAddr: F6 1F 01 26
MHDR: 40
FCtrl: c0
FCnt: 12449
```

```
FPort: 08
MIC: BC BE 4B 24
adaptive data rate: 1
ack: 0
direction: UP
message (encrypted): D4 5D 93 F0 F0 F6 60 C0 04
message (decrypted): 01 44 6C 83 05 00 FF FF 71

Sensor type: RHF1S001
temperature: 27.46 C, humidity: 58 %
period: 10 s, RSSI: -29 dBm, SNR: 9 dB, battery voltage: 2.6 V
Tx -> RS-485: " FF 10 10 0D 00 D0 F6 1F 01 26 BA 0A 3A E3 FF FF 09
1A 96"
```

2.6.2 Konfigurace systému

Pro vstup do configuration setup menu musí být odeslán příkaz "config". Z menu je možné vystoupit kdykoliv příkazem "quit"(bez uložení). Při vstupu do stavu konfigurace systému je pozastavena činnost gatewaye (přijímání LoRaWAN packetů, komunikace se serverem,...). V terminálu je nejprve vypsána kompletní aktuální konfigurace systému a následně menu konfigurace. Uživatel pak vybírá zadáním čísla. 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
server 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
```

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 do menu. Pro uložení nové konfigurace je potřeba v menu vybrat "Save and exit", systém pak následně vypíše které parametry byly změněny.

■ 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 se serverem zahrnuje nastavení adresy tohoto zařízení, adresa serveru a timeout, což je doba čekání na zprávu ACK od serveru po odeslání příkazu obsahujícího data ze senzorů. 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 server address:  
(current: FF)  
FE  
Server address is set to: FE  
  
Enter timeout (seconds)  
(current: 3)  
5  
timeout set to: 5 s
```

■ Config LoRaWAN protocol

Konfigurace LoRaWAN zahrnuje nastavení šifrovacích klíčů pro LoRaWAN protokol. Níže je příklad konfigurace.

```
LoRaWAN protocol configuration:  
Enter NwSKey (16 bytes in HEX)  
(current: FD 90 0D 8C 70 9F 19 24 18 EC FD D4 28 0C AC 47)  
111111112222222333333344444444  
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)  
111111112222222333333344444444  
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 to default configuration**

Po zvolení této možnosti je načtena defaultní konfigurace systému, která obsahuje hodnoty viz tabulka 2.4.

popis	hodnota
RS485 myAddr	0x10
RS485 serverAddr	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 2.4: Defaultní konfigurace systému

2.7 Koncová zařízení

Jelikož používaný protokol ke komunikaci se serverem je datově omezen na 6B na jeden paket, payload senzoru je dekódován v gatewayi a v packetu odeslaném na server jsou pouze nejdůležitější data, která se vejdu do této velikosti. Na základě typu koncového zařízení jsou data v packetu zakódovány, tudíž typ koncového zařízení musí být v gatewayi implementován. Typ koncového zařízení je definován jedním bytem a je uložen v gatewayi i na serveru společně s DevAddr u každého koncového zařízení.

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

Typ zařízení	Hodnota
RHF1S001	0x00
IMA_tempPress	0x01

Tabulka 2.5: Typy koncových zařízení

2.7.1 Zpracování dat jednotlivých typů koncových zařízení

Níže je popsáno pro jednotlivá koncová zařízení jak jsou data uložena ve struktuře, jak jsou data z této struktury zpracována a zobrazena a nakonec jak vybraná data jsou zapsána do výsledného bufferu o délce 6B, který je odeslán na K4 server

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);

```

```

18
19     /* Put the data into 6B long buffer, that is transmitted to the K4 server */
20     buffer [0] = RHF1S001_data.temperature & 0xFF;
21     buffer [1] = RHF1S001_data.temperature >> 8;
22     buffer [2] = RHF1S001_data.humidity;
23     buffer [3] = RHF1S001_data.rssi;
24     buffer [4] = RHF1S001_data.snr;
25     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;

```

2.7.2 Přidávání koncových zařízení ze serveru K4

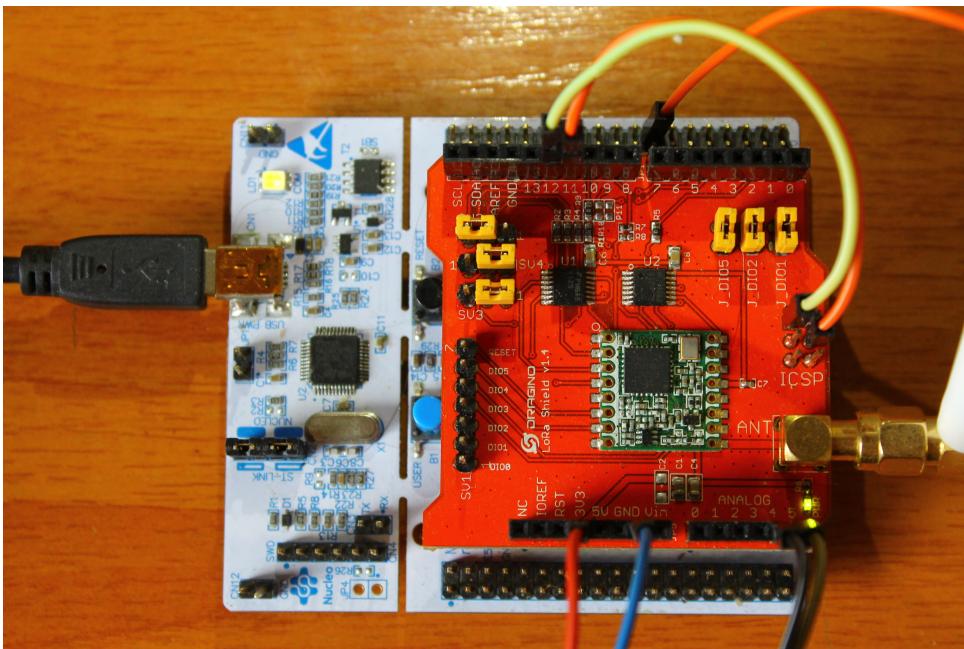
Koncová zařízení sítě se nastavují ze serveru K4 v podobě seznamu offline karet s délkou UID 8B. LoRaWAN device address je dlouhá 4B, jeden byte je navíc použit pro typ koncového zařízení, zbylé 3 byty jsou nuly. Jelikož typ zařízení je uložen v gateway i na serveru. Při odesílání zprávy průchod se tedy už typ zařízení neposílá z důvodu datového omezení v této zprávě (6B). 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.

2.8 Zapojení

LoRa shield [4] je nasazen přímo na vývojový kit. Nukleo kit neobsahuje ISCP konektor, který je součástí pinoutu Arduina a LoRa shield má SPI piny MISO a MOSI přivedený právě na tento konektor. Musí být tedy propojeny externě viz obrázek 2.4. Jumpery na shieldu musí také být stejně jako v obrázku.



Obrázek 2.4: foto zapojení

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

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 2.6: Pinout připojení externích periférií k procesoru

2.9 Instalace

K nahrání zkompilovaného programu do procesoru z PC není potřeba instalovat žádný SW. Kit je potřeba připojit k PC přes USB. V PC se to zobrazí jako flash disk. Zkompilovaný program s koncovkou .binary stačí překopírovat na toto zařízení. Po dobu kopírování souboru bliká na kitu LED1 červená/zelená. Jakmile kopírování skončí, program se spustí. Kit je také možné restartovat černým tlačítkem reset.

Pro uvedení Gatewaye do provozu je nutné se připojít k zařízení přes USB a nastavit všechny parametry viz sekce v tomto dokumentu: "Konfigurace systému".

2.10 Zdrojové soubory projektu

Projekt byl naprogramován v AtolicTrueSTUDIO, což je IDE založené na Eclipse. K programování procesoru byly použity standartní HAL knihovny od firmy ST Microelectronics. Pro šifrování LoRaWAN packetu 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 packet encryption
cmac.h	library for CMAC calculation in LoRaWAN protocol
LinkedList_ByteArray.h ..	Byte array linked list library for stacks
LoRaWAN_packet.h.....	LoRaWAN library for packet data decoding
stm3210xx_hal_conf.h	STM32 HAL library
ByteArray.h.....	Library for Byte array operations
LoRa.h	Library for interfacing LoRa transceiver
main.h.....	Main file
stm3210xx_it.h.....	STM32 HAL library
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_packet.c	source file to the LoRaWAN_packet.h
stm3210xx_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
stm3210xx_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_stm3210xx.c	HAL source file
usb.h	source file to the usb.h

Literatura

- [1] *NUCLEO-L073RZ*. ST Microelectronics [Online]. Available: <https://www.st.com/en/evaluation-tools/nucleo-l073rz.html> [Accessed: 19-Sep-2018].
- [2] *NUCLEO-L073RZ* ARM Mbed. [Online]. Available: <https://os.mbed.com/platforms/ST-Nucleo-L073RZ/> [Accessed: 19-Sep-2018].
- [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 [Accessed: 19-Sep-2018].
- [4] *Lora Shield*. Dragino. [Online]. Available: http://wiki.dragino.com/index.php?title=Lora_Shield [Accessed: 19-Sep-2018].
- [5] *tiny-AES128-C* bitdust. [Online]. Available: <https://github.com/bitdust/tiny-AES128-C> [Accessed: 19-Sep-2018].
- [6] *openpana*. OpenPANA. [Online]. Available: <https://github.com/OpenPANA/openpana> [Accessed: 19-Sep-2018].
- [7] *SparkFun Transceiver Breakout - RS-485* Sparkfun. [Online]. Available: <https://www.sparkfun.com/products/10124> [Accessed: 20-Sep-2018].
- [8] *LoRaWAN Specification*. LoRa Alliance. v1.1. Sparkfun. [Online]. Available: <https://lora-alliance.org/resource-hub/lorawan™-specification-v11> [Accessed: 20-Sep-2018].
- [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-2018].