VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
FACULTY OF MECHANICAL ENGINEERING

ÚSTAV AUTOMATIZACE A INFORMATIKY
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

NÁVRH VHODNÝCH HW A SW KOMPONENT PRO DÁLKOVÝ PŘENOS DAT

DESIGN OF SUITABLE HW AND SW COMPONENTS FOR REMOTE DATA TRANSMISSION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. Daniel Pietrowicz
AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. Zdeněk Němec, CSc.
SUPERVISOR

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání diplomové práce

Ústav:	Ústav automatizace a informatiky
Student:	**Bc. Daniel Pietrowicz**
Studijní program:	Strojírenství
Studijní obor:	Aplikovaná informatika a řízení
Vedoucí práce:	**doc. Ing, Zdeněk Němec, CSc.**
Akademický rok:	2020/21

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Návrh vhodných HW a SW komponent pro dálkový přenos dat

Stručná charakteristika problematiky úkolu:

1. Rešerše problematiky vhodného a energeticky nenáročného dálkového přenosu dat.
2. Analýza dnešních možných HW a SW řešení.
3. Návrh vlastní sestavy přístrojové a programové pro splnění daných podmínek.
4. Realizace hlavní části nebo úplného řešení a odzkoušení činnosti.
5. Zhodnocení výhod, nevýhod, možná řešení a cenové dostupnosti na trhu.

Cíle diplomové práce:

Problematika problému souvisí s provozem mimo napájecí síť, tedy bez závislosti na ní a včetně dlouhodobé funkcionality. Zároveň se hledí na cenovou dostupnost daného řešení. Cílem je navrhnout přístroj, který bude měřit denně úroveň elektrického napětí plynového potrubí a tato data posílat dále ke zpracování a vyhodnocení operátorem.

Seznam doporučené literatury:

MONK, Simon. Programming Arduino: getting started with sketches. New York: McGraw-Hill, C2012. ISBN 978-0071784221.

SigFox. Sigfox: Technical Overview. May, 2017.

LIBERG, Olof, Marten SUNDBERG, Y.-P. Eric WANG, Johan BERGMAN a Joachim SACHS, [2018]. Cellular Internet of things: technologies, standards, and performance. San Diego, CA, United States: Academic Press, an Imprint of Elsevier. ISBN 978-012-8124-581.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2020/21

V Brně, dne

L. S.

doc. Ing. Radomil Matoušek, Ph.D.	doc. Ing. Jaroslav Katolický, Ph.D.
ředitel ústavu	děkan fakulty

ABSTRAKT

Tato práce je zaměřena na návrh zařízení dálkového přenosu dat bez přístupu k síťovému napájení. Přístroj bude měřit napětí plynového potrubí a tuto informaci dále posílat operátorovi ke zpracování. Dále se práce zaobírá rozborem dnešních technologií pro dálkový přenos a na základě vybrané technologie bude navrhnuto zařízení. Podle vybraných komponent bude navrženo jeho napájení s cílem dlouhodobé funkcionality a nepřetržitého provozu.

ABSTRACT

This thesis is focused on the design of remote data transmission equipment, without access to mains power. The device will measure the voltage of the gas pipeline and send this information to the operator for processing. Analysis of today's technologies for long-distance transmission and based on the selected technology, the device will be designed. According to selected components, its power supply will be designed with the aim of long-term functionality and continuous operation.

KLÍČOVÁ SLOVA

Dálkový přenos, lithiová baterie, mikropočítač, GPRS modul, vysílač, Sigfox, NB-IoT, LoRa, napětí, Arduino Nano, klon

KEYWORDS

Remote transmission, lithium battery, microcomputer, GPRS module, transmitter, Sigfox, NB-IoT, LoRa, voltage, Arduino Nano, clone

ÚSTAV AUTOMATIZACE A INFORMATIKY

2021

BIBLIOGRAFICKÁ CITACE

PIETROWICZ, Daniel. *Návrh vhodných HW a SW komponent pro dálkový přenos dat*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav automatizace a informatiky, 2021, 75 s. Diplomová práce. Vedoucí práce: doc. Ing. Zdeněk Němec, CSc.

PODĚKOVÁNÍ

Chtěl bych vyjádřit vděk svému vedoucímu práce, doc. Ing. Zdeňku Němcovi, CSc., za ochotu a vstřícnost. Dále bych chtěl poděkovat své rodině a blízkým za morální podporu.

ČESTNÉ PROHLÁŠENÍ

Prohlašuji, že, že tato práce je mým původním dílem, vypracoval jsem ji samostatně pod vedením vedoucího práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury.

Jako autor uvedené práce dále prohlašuji, že v souvislosti s vytvořením této práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následku porušení ustanovení § 11 a následujících autorského zákona c. 121/2000 Sb., včetně možných trestně právních důsledků.

V Brně dne 22. 5. 2021 .....

Daniel Pietrowicz

OBSAH

1 ÚVOD

Automatizace a informatika se v této době každým dnem posouvá o kus dopředu a je využívána v našem každodenním životě. Usnadňuje nám přístup k informacím, jejich šíření a přenosu.

Na tento popud bude v této práci navrženo zařízení, které by mohlo nahradit každoroční měření napětí v plynovém potrubí. Zaměstnanec tedy nebude muset dojet na místo a změřit manuálně danou informaci.

Tento údaj je sice lehko dosažitelný, ale je velmi potřebný. Jedná se o katodickou protikorozní ochranu. Napětí potrubí vůči zemi se musí udržovat pod hranicí -1 V. Toto napětí je dodáváno ze stanic umístěných po celé délce potrubí. Jsou od sebe vzdáleny průměrně 40-60 km, mezi těmito stanicemi se nachází měřící body. Zařízení bude umístěno přibližně ve středu vzdáleností dvou sousedních stanic. Tomuto bodu říkáme bod kritický, zde je napětí nejblíže k nule.

V první řadě se tato práce bude soustředit na výběr vhodného typu dálkového přenosu. V tomto oboru je technologií mnoho, liší se samozřejmě energetickou náročností, vzdáleností, na kterou jsou schopna vysílat, způsobem komunikace a řadou dalších parametrů.

Preferovaná technologie bude ta, jež nebude energeticky náročná a zvládne přenos malého objemu dat na velkou vzdálenost. Tímto vypadává například Wi-Fi a Bluetooth. Podmínky splňuje například komunikace Goupe Spécial Mobile, zkratka GSM, GPRS nebo LTE. Mezi těmi méně známými je technologie LoRa, LoRaWan, Sigfox a NB-IoT. Vybranou modulací se stává Sigfox pro své pokrytí v České Republice, nízkou spotřebu energie a velký dosah mimo zastavěnou oblast.

Další důležitý krok pro návrh tohoto zařízení je výběr vhodného mikrokontroléru pro zpracování a odeslání dat. Tím se stává pro svou rozšířenost a cenu Arduino Nano. Aby bylo zařízení energeticky nenáročné, bude potřeba jej dimenzovat jak pomocí programu tak i úpravou samotné desky. Arduino je open-source, tím umožňuje vznik neoriginálních verzí desek, nazýváme je klony. Jsou z pravidla levnější, ale jak se ukáže během návrhu zařízení, nejsou plně vhodné k účelu minimálního odběru proudu.

Jako napájení komponent je vybrána lithiová baterie. Ta vyčnívá svou provozní teplotou, která se pohybuje i níže než -40 °C. Tento parametr je podstatný, kritické body nejsou nijak vytápěny a provoz bude celoroční. Konkrétně primární článek z chemické sloučeniny lithium-thionyl chloridu je dobrý pro svou vybíjecí charakteristiku a kapacitu až 19 000 mAh.

Arduino IDE je použito jako vývojové prostředí, je určeno pro jazyk C a C++. Díky rozšířenosti v celém světě má velkou podporu třetích stran, proto máme k dispozici velké množství knihoven pro různé mikrokontroléry, jejich funkce a periferie, s nimiž můžeme pracovat.

2 TECHNOLOGIE DÁLKOVÉHO PŘENOSU

Tato kapitola je zaměřena na stručný souhrn dnešních technologií dálkového přenosu dat a popis vybrané modulace Sigfox. Ve 21. století je výběr široký, nejznámější a nejpoužívanější je GSM, další jsou například LoRa a Sigfox. Řadí se zde samozřejmě i Bluetooth a Wi-fi, ale ty jsou vhodnější pro větší objem dat a hlavně kratší vzdálenosti, řadově desítky metrů (10 až 100 m).

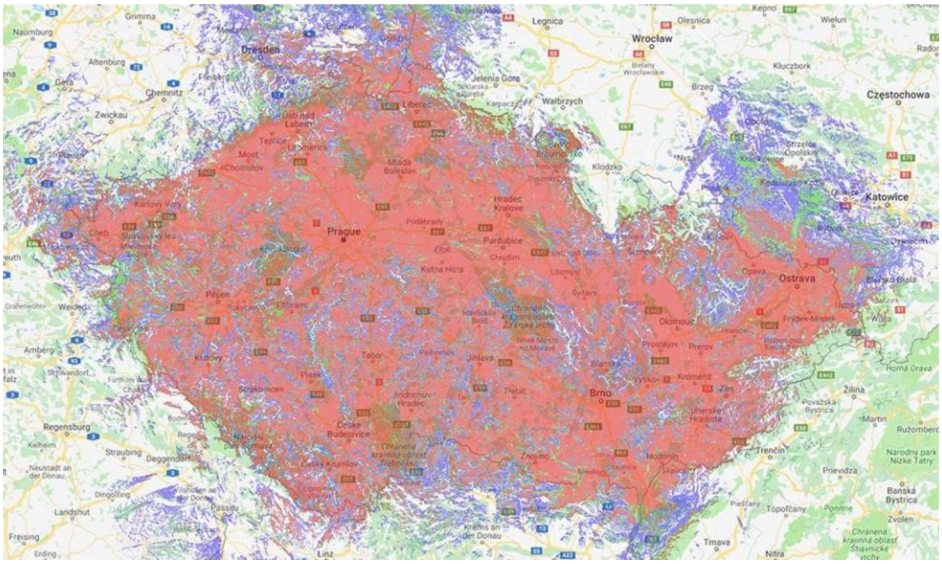
2.1 Sigfox

Sigfox je nejstarší ze všech komunikačních nízko energetických přenosů. Díky tomu je široce používána a odladěna. Patří francouzské společnosti stejného názvu jako tato technologie.

Dosah je ovlivněn okolím, pohybuje se mezi jednotkami kilometrů v zastavěné oblasti a desítkami kilometrů v nezastavěných oblastech, maximum je až 50 km. Další výhodou je také odolnost vůči rušení nebo nízká pořizovací cena potřebných modulů. [1]

Kvalita signálu se odvíjí dle počtu přijímacích stanic v dosahu. Čím více stanic, tím lepší šance na úspěšný přenos dat.

V České republice je jeden z poskytovatelů Sigfox Czech Republic. Na svém webu uvádějí pokrytí 94 % území naší republiky. Zasahuje i do míst, kde není dosah GSM signálu. Na obrázku níže je mapa pokrytí z portálu Sigfox.



Obrázek 2.1 Pokrytí Sigfox v Česku [2]

Díky velkému počtu přijímacích stanic je téměř 100 % úspěšnost přenosu. Při výpadku jedné stanice lze odeslanou zprávu doručit přes jinou.

Úspora energie je dosažena díky dvou klíčových aspektů. První je velikost samotných zpráv a omezení počtu zaslaných dat za den.

Velikost zprávy se pohybuje mezi 1 až 12 bajty. Lze zaslat i zprávu o velikosti 0 bajtů, jde pouze o ping ze serveru do modemu a naopak. Objem dat ve zprávě není nijak převratný, nicméně je vhodný pro řadu aplikací, kde je klíčová výdrž baterie.

Omezení je 140 zpráv ze zařízení na IoT server denně. Čímž se zajistí služba pro větší spektrum zařízení. Síť není příliš vytížena.

2.1.1 SFM10R1 Sigfox modem

Na obrázku níže je modem pro komunikaci v síti Sigfox. Komunikuje přes sériovou linku (UART rozhraní) a umožňuje připojení například k mikrokontroléru Arduino. Napájecí napětí je od 1.8 V do 3.6 V, proudový odběr při odesílání nepřesáhne 80 mA a při příjmu zpráv se pohybuje okolo 15 mA. Další důležitý parametr je odběr v režimu spánku, ten je 2 µA při napětí 3.3 V. Díky těmto vlastnostem je velice nízko energetický náročný a proto se hodí pro bateriový provoz, výdrž zařízení bez servisního zásahu by mohla být v rozmezí 5 až 15 let. Jeho cena je na našem území přibližně 360,- Kč. Důležitá je také provozní teplota od -30°C do +85°C, vhodné na venkovní provoz. [1]

Obrázek 2.2 Sigfox 868 MHz modul [1]

Jiné kompatibilní chipsety jsou například firmy Silicin Labs, SI446X nebo firmy Semtech, SX1272, existuje jich poměrně hodně. Důležitý parametr je obvod na zasílání a příjem radiových frekvencí a jednočipový počítač s flash pamětí alespoň 5 kB až 10 kB pro obdržená data.

Data zasíláme a přijímáme pomocí sériové linky mezi modemem a mikropočítačem. Celkem je potřeba zapojit 4 piny a pro lepší dosah anténu. Konfigurace má rychlost 9600 Bd, 8 datových bitů, 1 stop bit, žádnou paritu a kontrolu toku dat. Toto rozhraní umožňuje externí řízení a konfiguraci. Vše je dosaženo pomocí AT příkazů. [1]

2.2 GSM

GSM, Goupe Spécial Mobile je velmi rozšířený digitální buňkový globální systém pro mobilní komunikaci. Je budován jako otevřený celoevropský standard. Otevřel dveře takzvanému mezinárodnímu roamingu. [3]

Základem tohoto způsobu komunikace je karta SIM (ang. Subscriber Identity Module). Tento modul nese identifikační údaje účastníka (IMSI, ang. International Mobile Subscriber Identity), zařízení, a také spoustu dalších důležitých informací. Mezi ně patří například ověřovací klíč, telefonní seznam účastníka, identifikační číslo modulu, data o předplacených službách a další. [3]

Mobilní přístroj lze používat pouze s aktivovanou SIM kartou poskytovatelem služby. Existují i výjimky jako je tísňové volání. Tento přenos je kódován a šifrován, značně znemožňuje odposlech datového toku informací. [3]

Obrázek 2.3 SIM karta pro mobilní přístroj

Základní technologie GSM funguje v pásmu 900 MHz, poptávka vedla k vývoji více variant. Nyní existují tří standardy:

- GSM 900
- GSM 1800
- GSM 1900

Číslo v názvu označuje pásmo, ve kterém standard funguje. Jedná se o 900 MHz, 1800 MHz a 1900 MHz. Varianty 1800 a 1900 jsou také označovány jako digitální komunikační systém (DCS, ang. Digital Communication Systém). [3]

Systém GSM umožňuje poskytování hlavně telekomunikačních a přenosových služeb. Mezi ně patří především [3]:

- Telefonie (i tísňové volání)
- Přenos krátkých textových zpráv, tzv. SMS (Short Message Services). Ty mohou obsahovat až 160 znaků
- Záznamová služba
- E-mail
- Bankovní služby

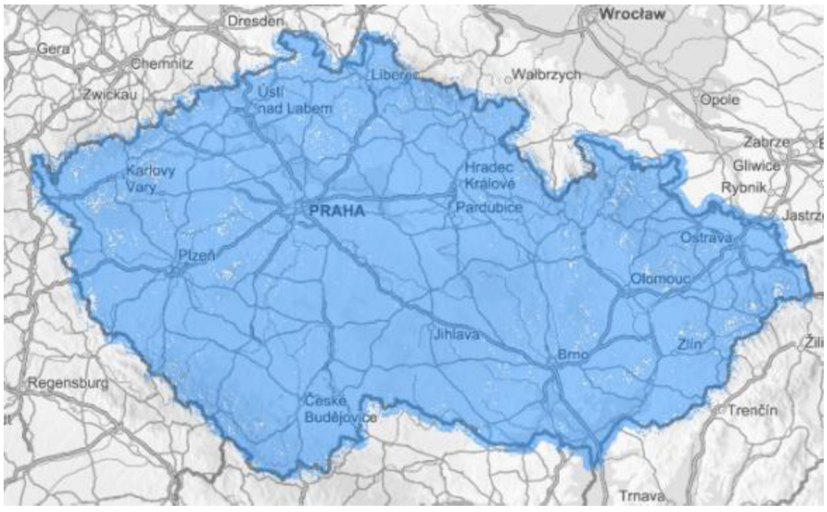
2.2.1 SIM 800L GSM modem

Typické mobilní stanice se skládají z vysílací a přijímací části, řídícího mikroprocesoru, SIM karty a řady možných periferií jako je Mikrofon, klávesnice, displej a další.

Obrázek 2.4 GSM modul SIM800L V2.0 [4]

Na obrázku výše je GSM modul. Tento modul je vhodný pro mikropočítače rodiny Arduino. Zdroj pro tento komponent musí být v rozmezí 4.9 - 5.2 V a alespoň 3 A. Při menším zdroji lze doplnit o kondenzátor na vstup. Během vysílání je vysoký špičkový odběr energie. Na českém trhu se tato periferie pohybuje v rozmezí 250 až 550 Kč. [5]

Dalším velice důležitým aspektem k zohlednění patří pokrytí. To je nejlépe znázorněno na mapách. Liší se daným poskytovatelem dané služby. V České republice je poskytovatelů mnoho. Mezi známé patří T-Mobile, O2 a Vodafone.



Obrázek 2.5 T-Mobile - Pokrytí GSM (2G) [6]

Na obrázku 2.5 je zobrazena mapa České republiky s pokrytím od operátora T - Mobile. Téměř celá republika je pokryta signálem GSM (2G).

2.3 LoRa

Technologie LoRa (ang. Long Range) je patentovaná modulace (US7791415 /2008, EP2763321 /2013). Vyvinula jí původně firma Cycleo a později tuto firmu odkoupila společnost Semtech. Díky svému potenciálu v dálkových přenosech a internetu věcí (IoT) vznikl spolek LoRa Aliance s velkým počtem členů. [7]

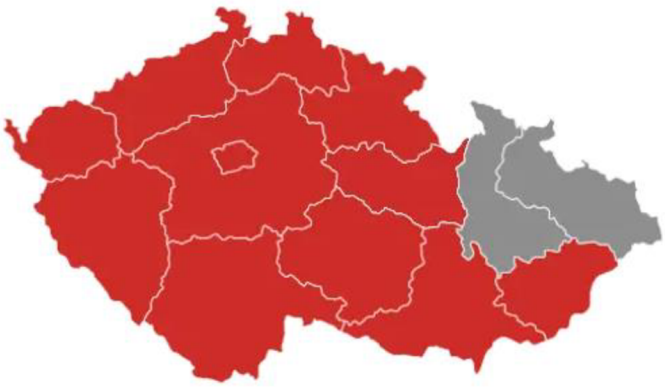
Jelikož jde o patentovanou modulaci, hardwarová implementace není veřejnosti dostupná. Její klíčová vlastnost vyplývá z jejího názvu, velký dosah. Pohybuje se okolo desítek kilometrů. Mimo tuto vlastnost vyniká také svou nízkou energetickou náročností. S jejím dosahem a energetickou náročností souvisí nízký objem posílaných dat a nízká frekvence posílaných dat.

Používá bezlicenční frekvenční pásma ISM. Podobně jako u GSM, se tato pásma liší státem, kde je LoRa používána. Nejčastější kmitočty jsou [8]:

- 915 MHz
- 868 MHz
- 434 MHz
- 315 MHz

V Evropských státech se používá kmitočet 868 MHz. Existují tři důležité parametry, a to rozprostírací faktor (ang. Spreading Factor), šířka pásma (Bandwidth) a kódový poměr (Coding Rate). S rostoucím rozprostíracím faktorem roste úspěšnost zachytit a dekódovat slabší signál, číselně se pohybují mezi -5 dB až -25 dB. Šířka pásma má vliv na datovou rychlost a citlivost přijímače. Musí se řídit normami vysílání daného regionu. Je konfigurovatelná ve spektru od 7.8 kHz do 500 kHz. Pomocí kódového poměru měníme délku zaslaných dat. S vyšším poměrem roste i oprava chyb přenosu při zpětném dekódování. Specifikovaný pro tuto modulaci jsou poměry 4/5 až 4/8. Nejlepší odolnost proti chybám má 4/8. [8]

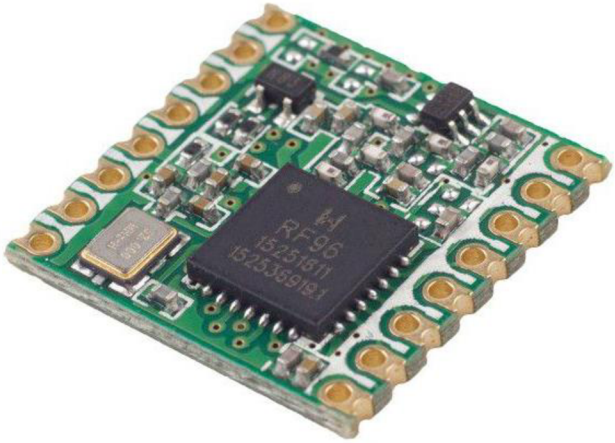
Pro komunikaci je nutno mít vysílač a přijímač dat. Kde vzdálenost těchto dvou zařízení se pohybuje řádově v jednotkách až desítkách kilometrů. Existuje i český poskytovatel této služby jménem Starnet. Na obrázku číslo 4 je vidět dosavadní pokrytí touto modulací.



Obrázek 2.6 LoRaWAN - pokrytí komunikací firmou Starnet [9]

2.3.1 RFM95 LoRa modem

K realizaci této komunikace je zapotřebí RFM9x modul pro mikroprocesory Arduino, Raspberry Pi a spoustu dalších. Verze RFM95 podporuje evropskou frekvenci 868 MHz.



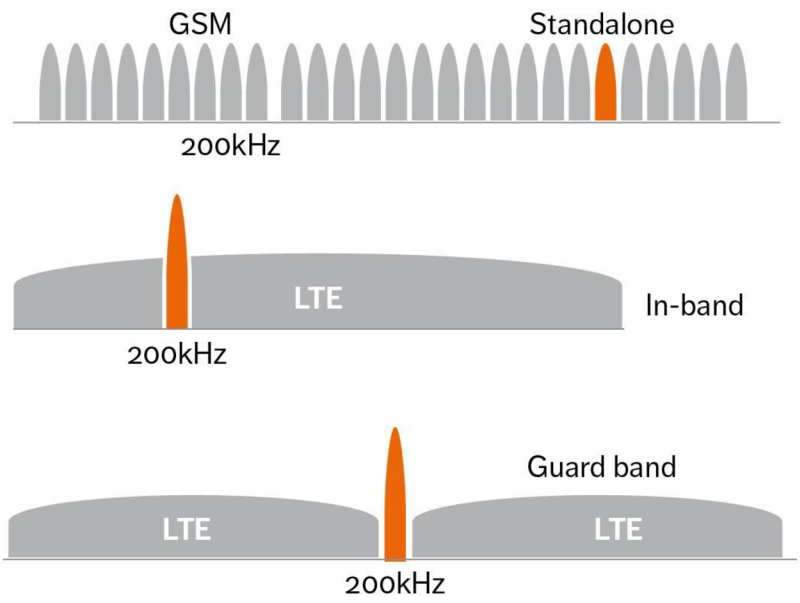
Obrázek 2.7 RFM95 modul s podporou LoRa modulace [10]

Jeden z důležitých parametrů je tedy špičkový proudový odběr, ten je zde velice nízky, cca. 100.3 mA, v režimu spánku je odběr přibližně 200 nA. Cena i s anténou se na českém trhu pohybuje kolem 900,- Kč. Dosah se pohybuje ve vzdálenosti kolem 2 km. [10]

Výdaje na roční provoz pro providera Starnet se líší počtem přijatých a odeslaných zpráv. Pro jedno zařízení pro tuto aplikaci je velice dostačující nejnižší tarif, čili 100 přijatých a 5 odeslaných zpráv, cena je 229,- Kč.

2.4 NB-IoT

NB je zkratka pro Narrow Band, čili úzké pásmo. Podobně jako u předchozích technologií je i tato bezdrátová. Její velké plus je možnost provozu v pásmech GSM a LTE, jejím poskytovatelem jsou v České Republice například Vodafone nebo T-mobile.



Obrázek 2.8 NB-IoT pásmo [11]

Na výše uvedeném obrázku je vidět několik způsobů implementace. První je paralelní běh zároveň s GSM, čímž se dovolí rychlejší nasazení po celém světě. Druhá metoda je pomocí flexibilní části spektra pro LTE. Tato možnost dodává největší pásmo a celkovou cenovou dostupnost řešení. Poslední je ochranné pásmo LTE. Zde je cíl minimalizovat rušení, kde LTE a NB-IoT musí fungovat souběžně. [11]

Obrázek 2.9 NB-IoT - Vodafone pokrytí [12]

Pokrytí se líše dle poskytovatele, obrázek 9 ukazuje Vodafone, jenž by měl pokrýt celou republiku. Také bude záležet na geografické členitosti, kvalita signálu bude lepší v okolí větších měst než ve vysokohorských oblastech.

2.4.1 MKR NB 1500 NB-IoT modem

Jeden z nejznámějších modulů je založen na mikroprocesoru rodiny Arduino. Nazývá se MKR NB 1500. Na českém trhu se pohybuje s cenou okolo 2300,- Kč.

Obrázek 2.10 NB-IoT - MKR NB 1500 [13]

Jedná se o kompletní zařízení, chybí pak vybrat vhodnou baterii a dokoupit SIM kartu. Oproti dříve zmíněným IoT modulům, není potřeba dokoupit vhodný mikroprocesor ke zpracování dat, tomu odpovídá cena.

Provozní napětí je 3,3 V, minimální proudový odběr není výrobcem přesně stanoven. Důvod je složení ze dvou základních komponent, procesor je 32 bitový SAMD21 a jako rádiový modul je použit SARA-R410M-02B. Maximální odběr pro Narrow Band je 140 mA. [14]

3 MIKROKONTRÓLERY

Kapitola se věnuje vybranému Arduinu Nano V3.0 s čipem ATmega328P, porovnání s klonem a jiným alternativám mikropočítačů. Mezi ty řadíme například rodinu Raspberry Pi, ARM a mnoho dalších.

3.1 Arduino

Jedná se o open-source platformu hardwaru a softwaru. V překladu to znamená, že vše o deskách Arduino je volně přístupné. Od zapojení komponent po programové prostředky a vývojové prostředí. Toto dovoluje i vznik takzvaných klonů, které jsou zpravidla levnější. Nicméně se během testování v praktické části této práce ukáže, že k účelu nízké spotřeby energie se klon příliš nehodí.

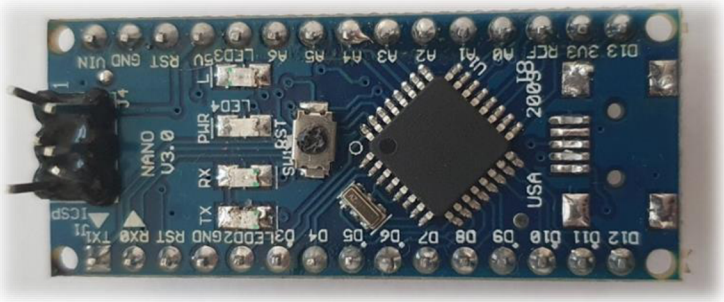
Je založeno na procesorech ATmega firmy Atmel a využívají programovací jazyk C++, ale taky jiné funkce a metody.

3.1.1 Originál nebo klon

Hlavní rozdíl mezi originálním a kopií Arduina je cena za jednočipový počítač. Klon se pohybuje s cenou i 3krát až 4krát níž, záleží na jeho kvalitě. Nicméně kvalita je kolikrát na velmi dobré úrovni, co se týče plošných spojů a osazování součástkami. Součástky už pak nemusí být naprosto totožné, ale funkčnost je téměř stejná.

Při koupi klonů je zásadní nevýhoda ta, že k němu není žádná oficiální dokumentace. Firemní podklady k originálu obvykle obsahují přes 300 stran informací. V nich je nespočet údajů od popisu registrů, vnitřních schémat po desítky charakteristik chování za různých teplot a jiných externích vlivů.

Mimo cenu má klon minimálně ještě jednu výhodu, a tou je dostupnost na českém trhu. Originální čip není běžně dostupný ani na portálech zabývajících se konkrétně Arduiny. Lze jej po delší době hledání nalézt, ale klon je téměř na každém rohu, a hned více druhů.



Obrázek 3.1 Arduino Nano V3.0 Klon

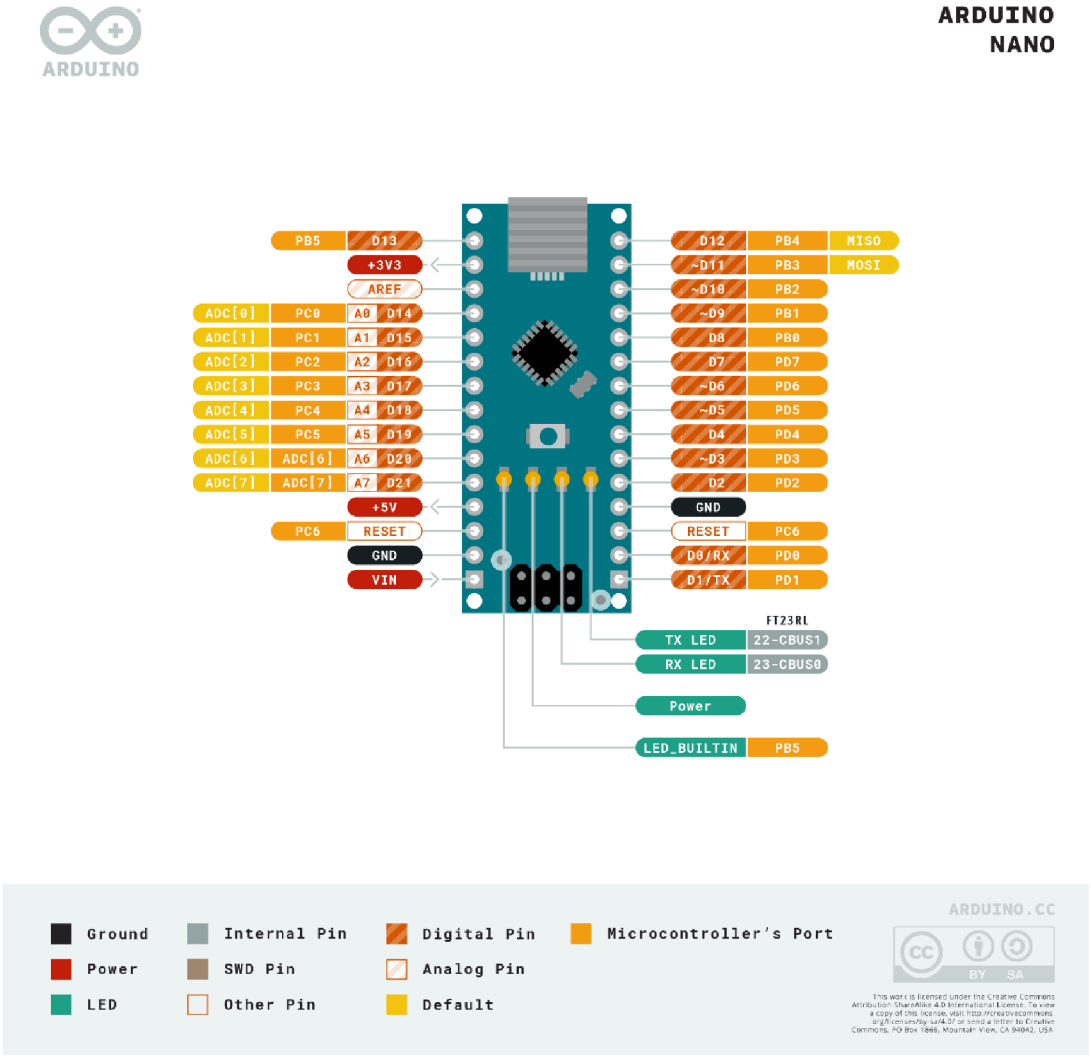
3.1.2 Arduino Nano

Nano je miniaturní, ale kompletní deska. Jeho jádro je mikroprocesor ATmega328 s architekturou AVR. AVR je označení 8 bitových a někdy 32 bitových čipů s harvardskou architekturou. Zkratka vzešla ze jmen tvůrců a RISC strategie. [15]

Základní rozlišení je fyzicky rozdělená paměť instrukcí a dat. Paměti mohou mít zcela jiný typ. V případě ATmega328 má 32 kB FLASH na program a 1 kB EEPROM na data. Obě paměti se při odpojení napájení nesmažou. [15]

RISC značí procesor s redukovaným setem instrukcí. Cílem je zjednodušit a optimalizovat výkon. Základní rysy jsou například řetězení instrukcí, víceúčelové registry a hardwarová implementace mikroinstrukcí. [16]

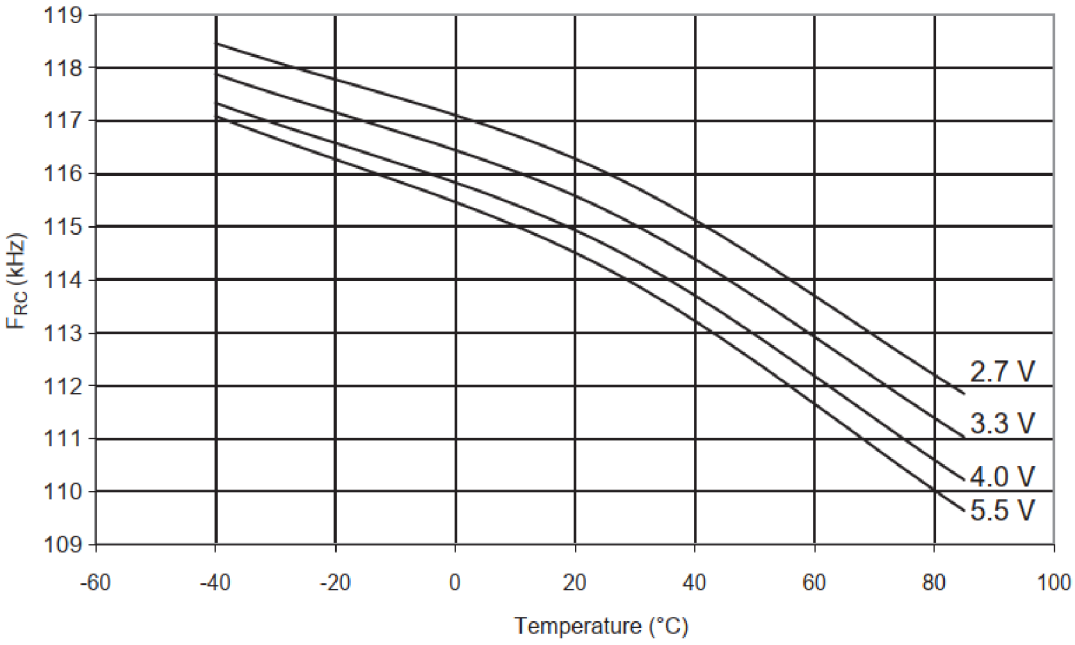
Operační napětí je 5 V, používá krystal o frekvenci 16 MHz. Tato rychlost je snížitelná za pomocí prescale registru a tím se také sníží požadavek na napájecí napětí. Díky regulátoru napětí je možno napájet desku s větším napěťovým zdrojem. Má 22 digitálních a 8 analogových vstupů a výstupů, každý z nich lze použít obousměrně. Definice použití je zapsána v programu. [15]



Obrázek 3.2 Arduino Nano Piny [17]

Pro účely návrhu zařízení bude potřeba využít dva digitální piny ke komunikaci s modulem do sítě internetu věcí. Analogový vstup ke změření napětí plynového potrubí a napájecí piny.

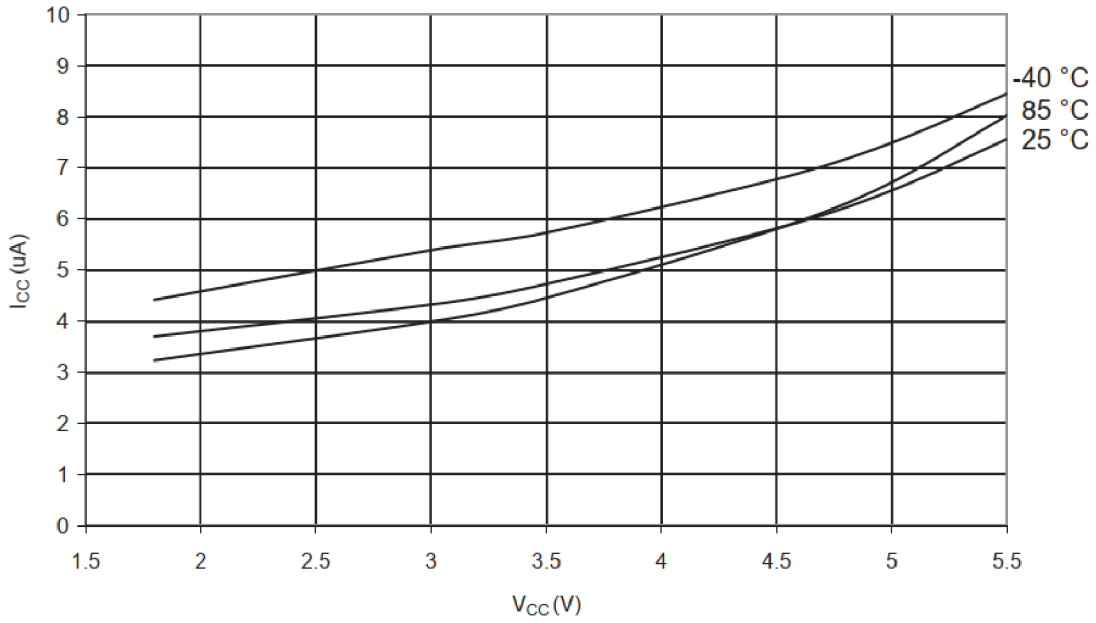
Důležitá komponenta je watchdog časovač, který má za účel resetování procesoru, jestliže dojde k jeho zacyklení nebo zastavení. V praktické části bude použit k probuzení mikroprocesoru z režimu spánku. Na tento časovač má vliv jak teplota, tak napájecí napětí. Závislost je uvedena ve firemní dokumentaci podle obrázku níže.



Obrázek 3.3 ATmega328P Watchdog [15]

Na výsledném zařízení bude použito napájecí napětí 3.6 V z lithiového článku. A teplota by se měla pohybovat v rozpětí -20°C až +40°C, což podle výše uvedené závislosti je rozdíl 117 kHz a 114 kHz, přibližně 3 kHz. Lze tedy říci, že frekvence posílání dat se bude lišit v závislosti na venkovní teplotě a největší rozdíl je tedy mezi zimním a letním obdobím.

Proudový odběr má ATmega328 také dobře popsán. Jako u většiny elektrických komponent má na něj zásadní vliv napájecí napětí a venkovní teplota. Graf na další straně znázorňuje jejich účinky v režimu spánku s watchdog metodou probuzení.



Obrázek 3.4 ATmega328P Power-Down Current [15]

V grafu je pro zmíněný procesor spotřeba energie při napětí 3.6 V kolem 5 až 6 µA. Při napájení z baterie by měl splnit požadovaný cíl dlouhodobého provozu. Samozřejmě vliv mají také i jeho další komponenty, které můžeme libovolně vypnout či zapnout pomocí PRR registru.

Tabulka 3.1 ATmega328P - PRR registr [15]

PRR bit	Napětí 3 V Frekvence = 4 MHz [µA]	Napětí 5 V Frekvence = 8 MHz [µA]
PRUSART0	22.17	100.25
PRTWI	46.55	199.25
PRTIM2	50.79	224.25
PRTIM1	41.25	176.25
PRTIM0	14.28	61.13
PRSPI	43.84	186.5
PRADC	61.8	295.38

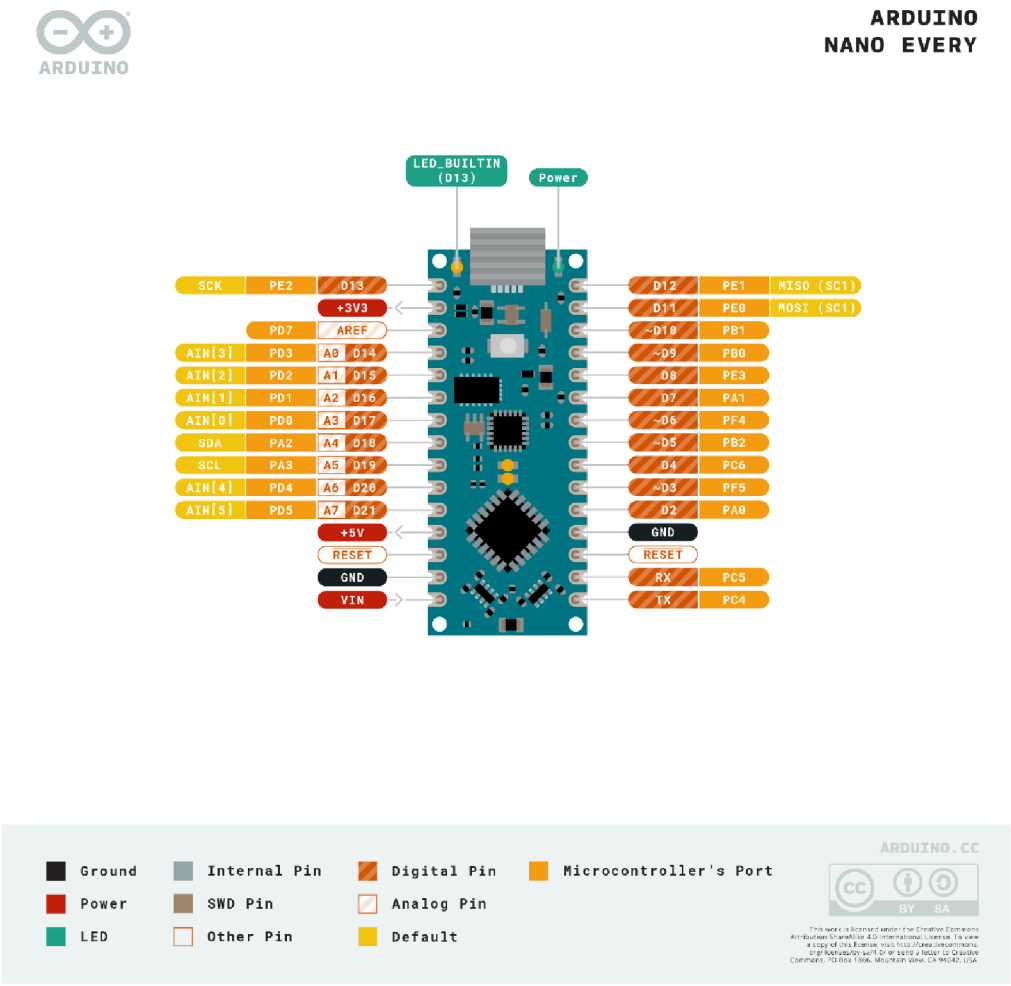
PRR registr poskytuje metodu odpojení krystalu od individuálních periferií za účelem snížení spotřeby elektrické energie. Deaktivovat můžeme například analogově digitální převodník, SPI sběrnici, sériovou komunikaci, watchdog a tři obecné časovače.

Jeho cena se na českém trhu pohybuje od 600,- Kč. Na oficiálním internetovém obchodě lze zakoupit za 20 €.

3.1.3 Arduino Nano Every

Nano Every je novější verze dříve zmíněné desky. Tato však obsahuje procesor ATmega4809. Poskytuje prostor pro větší projekty. Jeho programová paměť je 48 kB,

datová paměť je pouze 256 B. Operační data se ukládají do SRAM o velikosti 6 kB, ta je třikrát větší než u ATmega328P. [18]



Obrázek 3.5 Arduino Nano Every Piny [19]

Počet digitálních a analogových pinů je stejný, avšak v tomto případě lze využít všechny digitální piny jako externí přerušení. Používá krystal o frekvenci 20 MHz. Rozměrově je stejně velký jako dříve uvedené Nano, 45 mm na 18 mm. Na internetovém obchodě jej lze zakoupit ve dvou verzích, verze s uzpůsobenými kontakty do nepájivého pole stojí 11 €.

3.2 Raspberry Pi

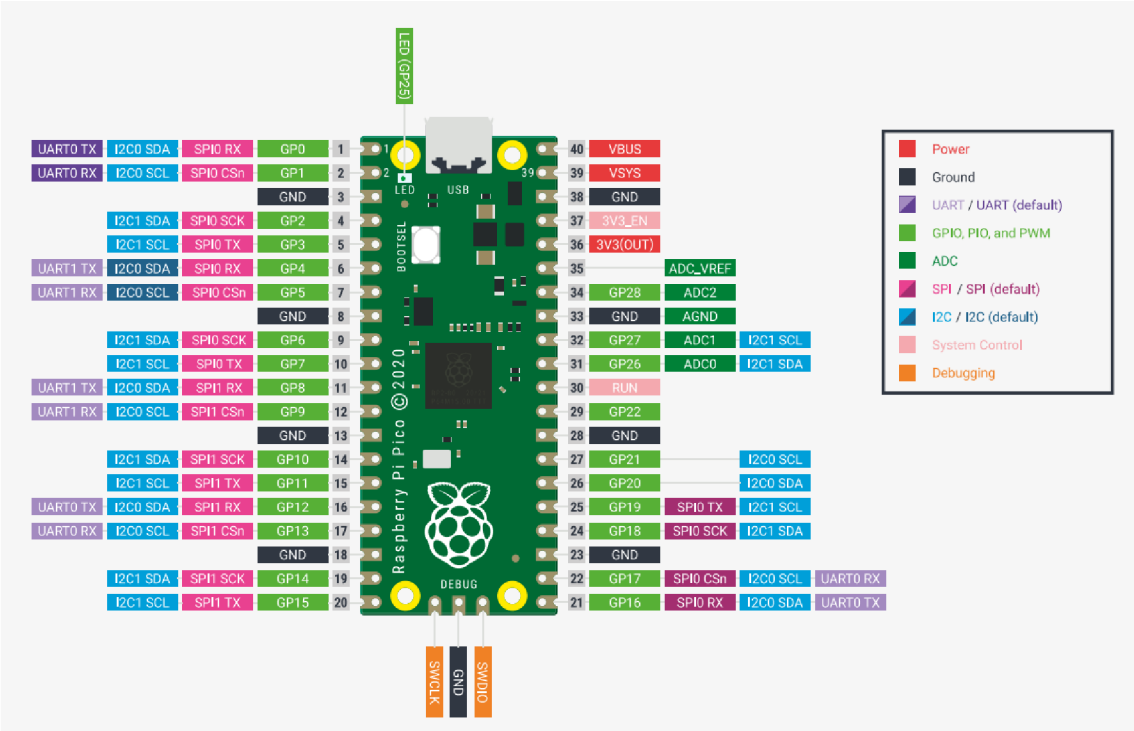
Raspberry Pi je založeno na procesorech typu ARM, zkratka ARM znamená pokročilý RISC stroj. Naproti Arduina je možné nahrát i jednoduchý operační systém a lze tento jednočipový počítač využívat podobně jako stolní desktop, nejpreferovanější systém je Raspbian vyvinut výhradně pro Raspberry Pi. Existují i jiné Linuxové distribuce.

Díky své přednosti výkonu s malou spotřebou se procesory ARM používají i v telefonech.

Původní ARM procesor byl taktovaný na 700 MHz, což je výkonnostně neporovnatelné s ATmega328, který podporuje maximálně frekvenci 20 MHz. Je možné nejen řídit aplikaci pomocí digitálních a analogových vstupů, ale také přímo vyvíjet program k řízení strojů. Deska má na sobě navíc grafický procesor, obrazový výstup, zvukový výstup, USB pro připojení dalších periferií a mnoho dalších komponent, které Arduina běžně nemají.

3.2.1 Raspberry Pi Pico

Tento model ze všech ostatních nejvíce konkuruje Arduinu Nano. Rozměrově je o pár milimetrů větší, 21 mm na 51 mm. Má dvou jádrový procesor ARM Cortex-M0, maximální možná frekvence je 133 MHz. Operační napětí je od 1.8 V do 5.5 V. [20]



Obrázek 3.6 Raspberry Pico Piny [20]

Pico podporuje režim spánku, bohužel kvůli jeho výkonu i během uspání má relativně velkou spotřebu energie. Tabulka na další straně z firemní dokumentace ukazuje spotřebu samotné desky v řádech jednotek miliampér, kdyžto u ATmega328 se pohybujeme v řádech desítek mikroampér.

Tabulka 3.2 Raspberry Pico – Spánek [20]

Měření	Proud při napětí 5 V [mA]		
	Teplota [°C]		
	-25	25	85
1	1.35	1.3	1.81
2	1.53	1.39	1.92
3	1.4	1.32	1.92
Průměr	1.4	1.3	1.9

Má vše potřebné, aby zvládl měřit hodnotu napětí a posílal jí do IoT sítě. Jeho přednost je výkon, ten však není v této aplikaci potřebný. S vyšším výkonem je také spjatá vyšší spotřeba energie, proto se nehodí na tuto aplikaci.

4 BATERIE

V této kapitole jsou sepsány základní informace o primárních bateriích. Jedná se o elektrický zdroj. Baterie se liší od akumulátoru důležitou vlastností, a tou je dobíjení, baterie nelze po vybití nabít.

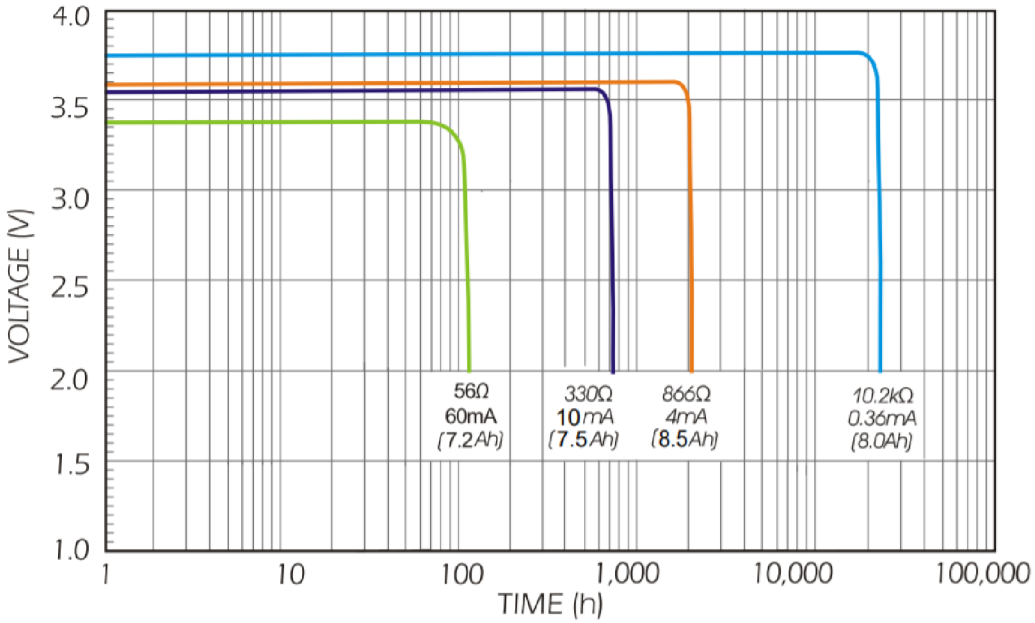
Do důležitých charakteristik pro výběr baterie řadíme teplotní rozsah během provozu. Je podstatné, aby baterie byla schopna funkce i za nízké teploty v zimě, nebo vyšší teploty v létě. Jelikož míříme i na dlouhodobou funkci, je životnost také významná. Baterie je spotřební zboží, výrobní záruka se dává okolo 6 měsíců. Vliv na její životnost má spotřeba zařízení, které napájí a také vnější podmínky. Obecně je známo, že během nízkých teplot klesá kapacita baterie na 80% své původní hodnoty. Jako poslední parametr pro tuto aplikaci lze zařadit vybíjecí charakteristiku. Některé jsou schopny si udržet hodnotu napětí téměř ke konci svého života a jiné naopak lineárně ztrácí svou hodnotu v jistých mezích. [21]

4.1 Lithiový článek ER 26500

Anoda je buď přímo z lithia, nebo z jeho sloučeniny. Napětí se podle složení pohybuje od 1.5 V do 3.7 V. Předností článku je životnost, používají se například v kardiostimulátorech, hodinkách a řada dalších. Katodu obvykle tvoříme z oxidu manganičitého.

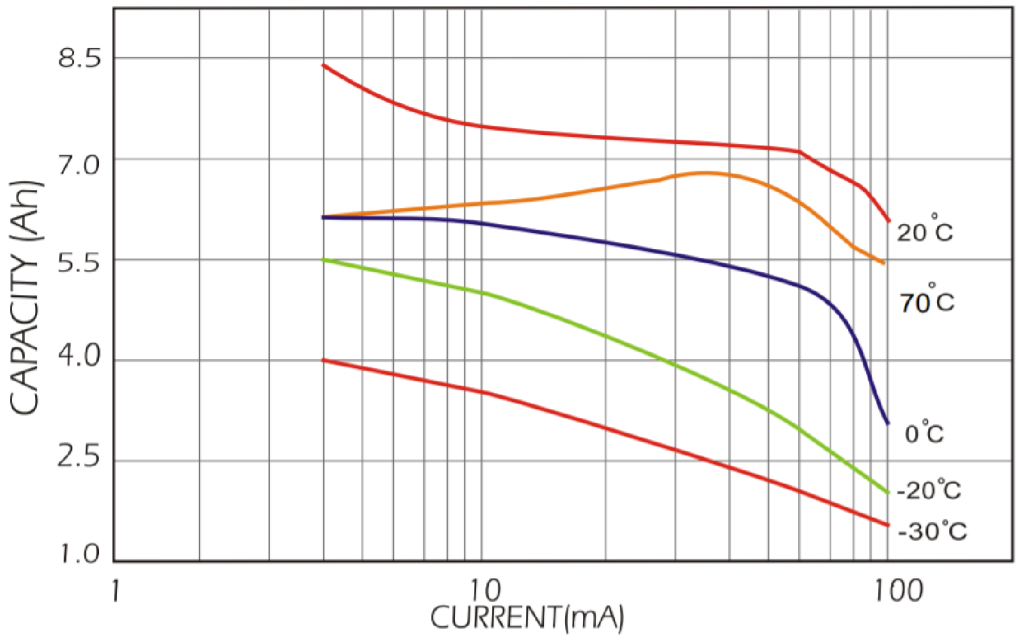
Obrázek 4.1 EVE lithiový článek ER 26500 [22]

Na obrázku je článek z lithium-thionyl chloridu ER 26500. Nominální napětí této baterie je 3.6 V, kapacita je 8500 mAh. Maximální dodávaný proud je 150 mA a cena se pohybuje od 180,- Kč do 350,- Kč, záleží na výrobci. Důležitý je mimo zmíněné parametry také dovolený provoz za teplot od -65 °C do 85 °C. [22]



Obrázek 4.2 ER26500 Vybíjecí křivky [22]

Graf na obrázku 4.2 znázorňuje vybíjecí křivky pro zmíněnou baterii. Na ose y je napětí a na ose x čas v hodinách. Nejpodstatnější je zachování konstantního napětí po téměř celou dobu vybíjení. Abychom dosáhli alespoň doby provozu alespoň dvou let, což je přibližně 17500 hodin, je potřeba dosáhnout maximální průměrné spotřeby 0.45 mA.

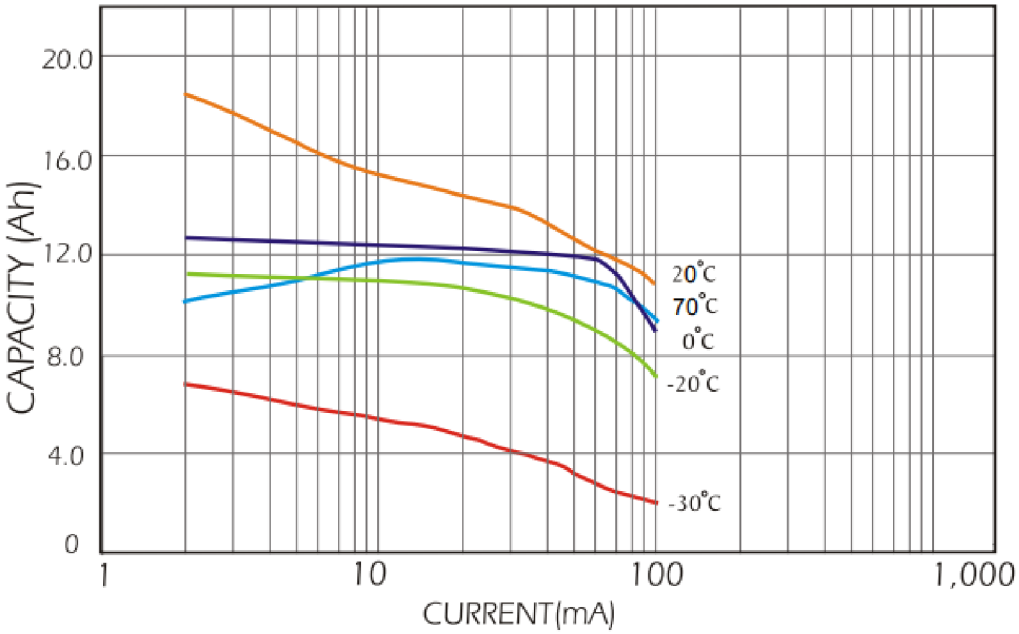


Obrázek 4.3 ER26500 Závislost kapacity na teplotě [22]

Graf znázorněný výše popisuje chování kapacity baterie v závislosti na teplotě a vybíjecím proudu. Průměrný odběr by neměl být vyšší než 0.45 mA, nicméně graf začíná až na 4 mA. Ideální podmínky jsou za pokojové teploty a nejhorší naopak v zimním období, kdy teplota může klesnout až na -30 °C. Kapacita v tomto případě klesne o více než 50 % své původní hodnoty.

4.2 Lithiový článek ER 34615

Chemické složení, provozní teplota a výstupní napětí je stejné jako u předchozího typu. Hlavní rozdíl v tomto případě je kapacita 19000 mAh. Vliv to má také na maximální výstupní proud 100 mA. Další rozdíl je váha, rozměr a cena baterie. Váha a rozměr nejsou zásadní. Cena je od 270,- Kč přímo na internetovém obchodu výrobce. [23]



Obrázek 4.4 ER34615 Závislost kapacity na teplotě [23]

Z grafu můžeme orientačně vyčíst, že při nízké teplotě -20 °C je kapacita stále vyšší, než u předchozího typu v plně nabitém stavu. Za pokojových podmínek by k provozu minimálně dvou let měl být průměrný odběr proudu maximálně 1 mA.

Vybíjecí charakteristiku není potřeba uvádět, jelikož je stejné složení. Rovněž si zachovává úroveň napětí 3.6 V až do téměř úplného vybití.

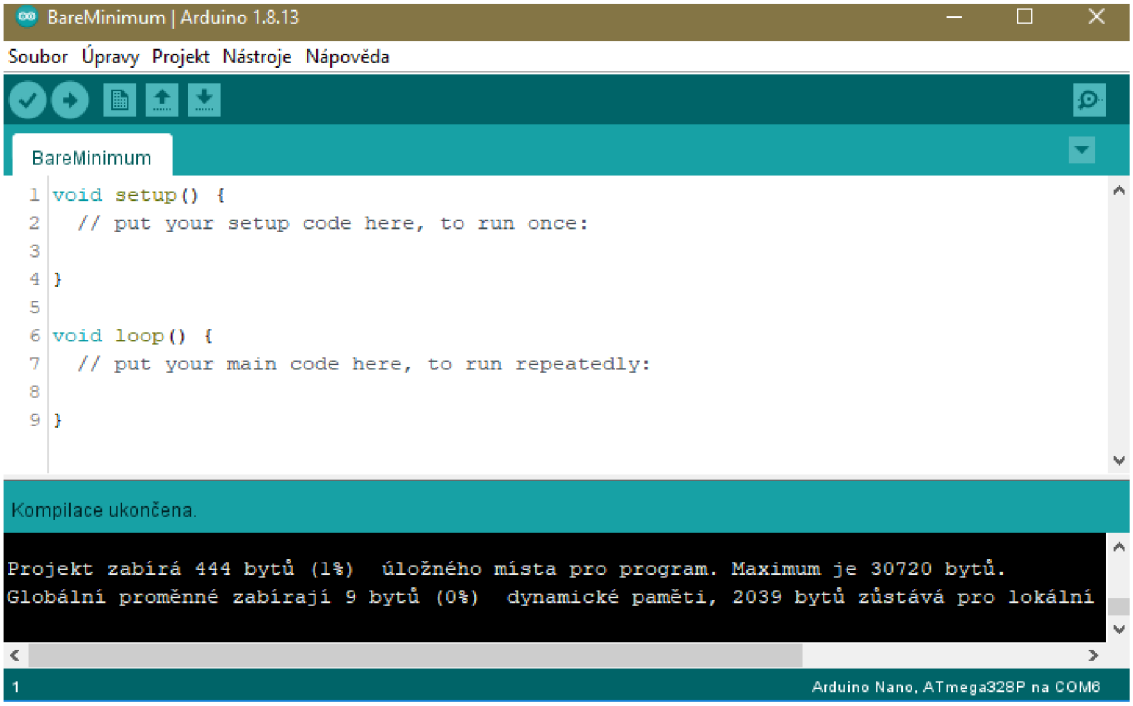
Jedná se o největší možnou kapacitu lithiové baterie. Zvýšit ji můžeme pomocí paralelního zapojení článků, celková výdrž je pak součtem kapacit. Velká nevýhoda je však nerovnoměrné vybíjení, které nelze přesně určit. Pokud bychom chtěli zvýšit napětí, pak lze baterie zapojit do série. V tom případě se kapacita zachová stejná, ale napětí se sčítá. V obou případech je nevýhoda stejná a je nutno použít stejný typ baterie.

5 VÝVOJOVÉ PROSTŘEDÍ ARDUINO IDE

Arduino IDE je zkratkou pro Integrated Development Environment, v překladu integrované vývojové prostředí. Stejně jako Arduino desky je i IDE open-source, k dispozici je jádro tohoto prostředí. [24]

Kompatibilita je zajištěna se třemi základními operačními systémy, Windows, Linux a macOS. Také jej lze používat na ARM procesorech. Využívá funkcí v jazyce C a C++.

5.1 Vývojové prostředí



Obrázek 5.1 Arduino IDE - Vývojové prostředí

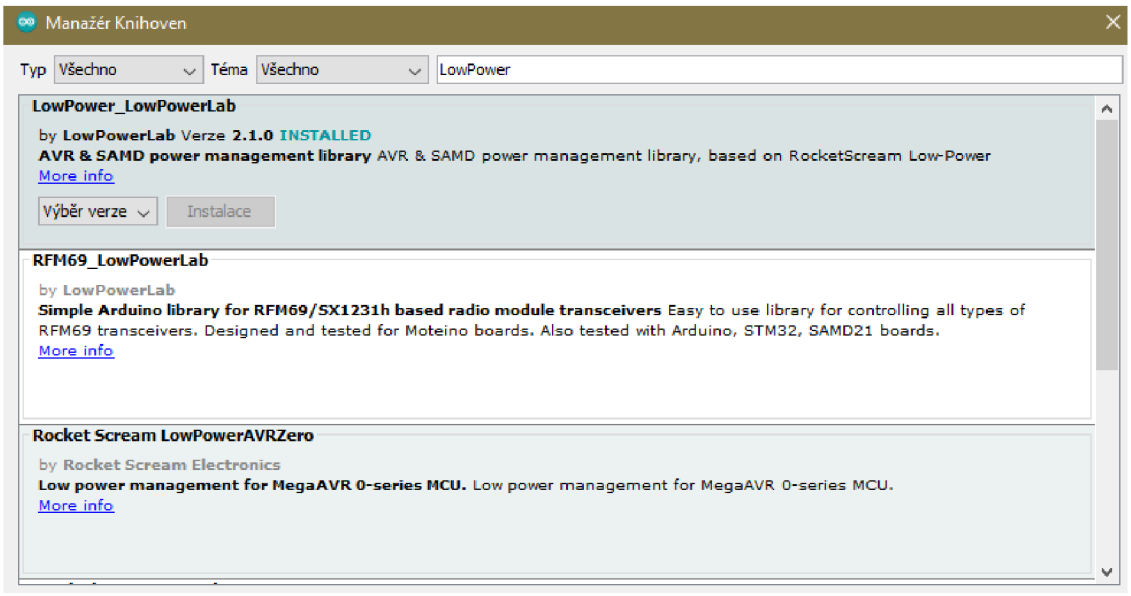
Na obrázku je zobrazeno okno prostředí IDE s příkladovým programem zvaným úplné minimum. V popisu okna je obsažena i používaná verze 1.8.13, která je odnoží programu Processing IDE. Pod popisem jsou nástroje, jenž umožní uzpůsobení a nastavení potřebných parametrů. Následují nejpoužívanější příkazy, mezi ty patří kompilace, nahrání do mikroprocesoru, vytvoření nového projektu, uložení nebo otevření. Mimo Arduino desky můžeme programovat i ARM, například výše uvedený Raspberry Pi Pico. Dále můžeme vidět část okna s programem, který je nutný pro vybranou desku. Desku vybíráme přes záložku nástroje a po zvolení se zobrazí v pravém dolním rohu i s komunikačním portem, ke kterému je mikropočítač připojen. V poslední řadě je statusový výpis. V něm je vidět úspěšná kompilace a níže jsou informace o velikosti, případně o chybě, která neumožní úspěšné nahrání.

5.2 Knihovny

V tomto projektu budou klíčové tři knihovny. První je nutná k vytvoření sériové komunikaci mezi Arduinem a Sigfoxem. Druhá je důležitá pro práci s řetězci, jenž budeme posílat z mikrokontroléru do IoT sítě. Poslední knihovna je nutná k uvedení mikrokontroléru do režimu spánku a minimalizace jeho spotřeby.

- SoftwareSerial.h
- Stdlib.h
- LowPower.h

První dvě zmíněné jsou již obsaženy v IDE prostředí. Třetí knihovnu je potřeba stáhnout za pomocí nástroje Manažera knihoven.



Obrázek 5.2 Arduino IDE - Manažer knihoven

V manažeru nalezneme nespočet knihoven. Každá se hodí na jinou aplikaci nebo jiný procesor. Existuje i knihovna pro komunikaci se Sigfox modemem, ale díky lehké aplikaci nebude potřeba.

Kód k připojení knihoven vypadá následovně.

```
1:    #include <LowPower.h>
2:    #include <SoftwareSerial.h>
3:    #include <stdlib.h>
```

Každý řádek připojuje jednu. Stačí nám mřížka, příkaz include a obalení názvu knihovny do znaků menší a větší než.

5.3 Jazyk C a C++

Jazyk C byl vytvořen v sedmdesátých letech s operačním systémem Unix. Řadí se mezi nízko úrovňový a jednoduchý programovací jazyk. Jeho nedílnou součástí je překladač, který převádí napsaný program v textovém editoru na strojový kód. Jedním z nejznámějších překladačů je GCC, GNU Compiler Collection. Dnes se GCC zabývá kompilací i C++ a jiných jazyků. [25]

C++ staví na základě jazyka C. Podporuje navíc objektově orientované programování. Je rozšířen o objekty a s tím spojené zapouzdření, dědičnost a další vlastnosti objektů. [25]

V této aplikaci budou použity základní stavební kameny jazyka C++, jako jsou globální a lokální proměnné, funkce a cykly. Rozdíl mezi globální a lokální proměnnou už vyplývá z názvu. Ke globální proměnné můžeme přistupovat kdekoliv v programu, jen je nutné hlídat práci s ní. Je nežádoucí, aby byla globální proměnná měněna na dvou místech v programu zároveň, týká se převážně větších aplikací. Lokální proměnné jsou vytvořeny buď přímo ve funkcích, nebo například v cyklech. Vytvoří se na začátku zavolání funkce a po dokončení se zpátky smaže a uvolní místo v paměti. V konečné verzi programu v kapitole 6.4 je globální proměnná definována a inicializována na řádcích 7 až 13 a lokální je na řádku 25 a 31. Rozdíl mezi definicí a inicializací je ten, že při definici pouze vytvoříme a nezapisujeme do ní žádnou hodnotu. Při inicializaci se proměnná vytvoří a zároveň se do ní zapíše zvolená hodnota.

S hodnotou proměnné souvisí její datový typ, existuje obrovská řada datových typů. Mezi základní patří například boolean, který má binární hodnotu 0 nebo 1, celé číslo se znaménkem nebo bez znaménka, desetinné číslo a řetězce. Celočíselná hodnota bude použita pro čtení analogového vstupu Arduina, kdy nám vstup bude vracet hodnotu 0 až 1023, jelikož je desetibitový. Následně tuto hodnotu převedeme na číslo s desetinnou čárkou tak, aby odpovídala hodnotě ve voltech. A poté je v konečné verzi programu provedeno přetypování hodnoty ve voltech na řetězec, který bude formulován do podoby AT příkazu pro Sigfox modul.

Cykly mohou mít tři podoby, v první je podmínka opakovaného spuštění ověřena na konci. V druhé podobně se podmínka ověřuje před započetím cyklu. Poslední cyklus má tři parametry, inicializace, podmínka a příkaz před začátkem nového opakování. [25]

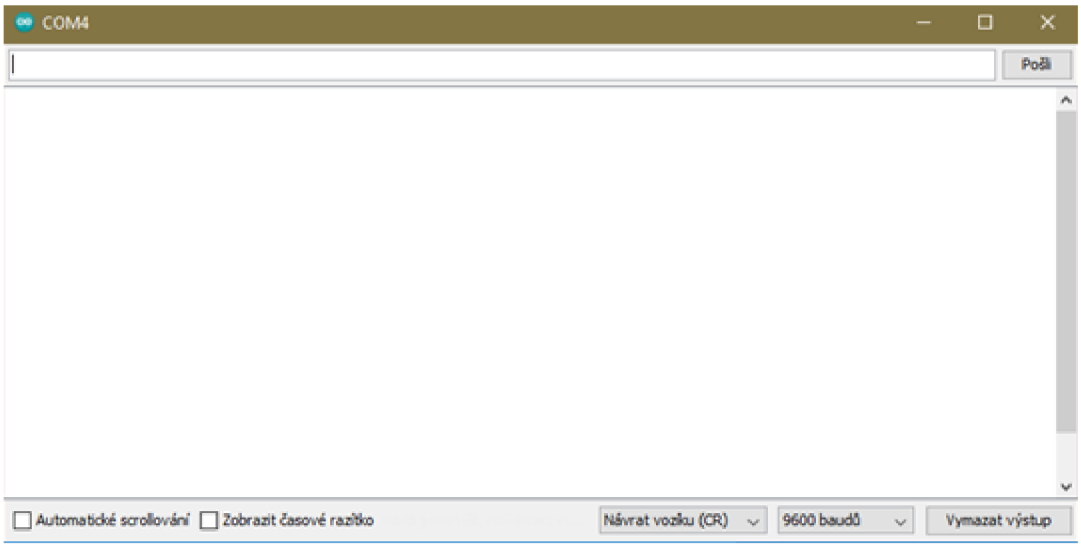
- Do – While (podmínka na konci)
- While (podmínka na začátku)
- For (inicializace, podmínka na začátku, příkaz)

Funkce můžeme rozdělit na návratové a bez návratové. Návratová funkce má definovaný datový typ, který bude vracet po zavolání. Opak je tedy funkce, která nemá určený datový typ a nevrací žádnou hodnotu při ukončení. Další důležitá vlastnost jsou vstupní parametry, které mohou, ale nemusí být při volání funkce vyplněny. Nutné je zachování jejich posloupnosti.

5.4 Sériový monitor

IDE má nástroj zvaný sériový monitor. Jeho funkce je mimo jiné také vhodná pro testování zařízení. Obdobou tohoto nástroje je například program hyperterminal, který umožňuje zobrazení přenosu na sériové lince mezi počítačem a mikrokontrolérem.

Můžeme v něm nastavit rychlost přenosu a ukončovací znak. Abychom docílili zasílání dat na sériovou linku z mikrokontroléru, je zapotřebí implementace knihovny SoftwareSerial pro Arudino.



Obrázek 5.3 Arudino IDE - Sériový monitor

Popis okna nese název kanálu COM4, který monitorujeme. První řádek slouží k zaslání dat z počítače na sériovou linku, větší okno zobrazuje přijatá data. Nastavit můžeme automatické scrollování nebo zobrazení časového razítka přijatých dat. Podstatná konfigurace spočívá v ukončovacím znaku a přenosové rychlosti.

Ukončovací znak může být nový řádek, návrat vozíku, oba dva nebo žádný. Rychlost přenosu limituje procesor a jeho frekvence. Více popisují obrázky 6.9 až 6.11 v následující kapitole.

6 NÁVRH ZAŘÍZENÍ

Tato kapitola se věnuje testování základních komponent. Mezi ty řadíme klon a originál Arduino Nano V3.0 a Sigfox modul WISOL SFM10R1. Tyto součástky jsou propojeny sériovou linkou. Po zprovoznění se dále provádí měření odběru proudu na testovacím programu a určení doby životnosti.

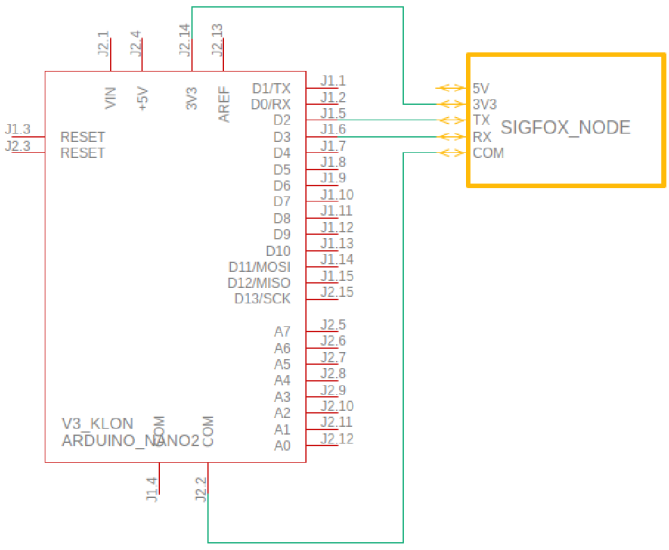
6.1 Testování funkčnosti

Klon mikrokontroléru je připojen k modulu pouze čtyřmi vodiči. Dva jsou k napájení a dva slouží ke komunikaci přes sériové rozhraní UART. Sigfox napájíme z Arduina pomocí vývodů na napětí 3.3 V a zemí. Sériová linka je připojena mezi GPIO klonu D2, D3 a modulu Tx, Rx.

Důvod proč jsou zvoleny GPIO místo Rx a Tx klonu je ten, že během testu jsou Rx a Tx připojeny přes mikro USB do PC. Během testu se pošle příkaz „AT“ z počítače na Sigfox a ten nám vrátí buď chybu, nebo „OK“.

Chyba by mohla být způsobena špatným propojením, napájením, hardwarovým problémem nebo programem. V programu je důležité dát pozor na konfiguraci rychlosti komunikace a správně zvolit piny (D2 jako Rx a D3 jako Tx) vůči zapojení.

Na obrázku níže je schéma, které je použito. Napájení je poskytováno z USB mezi PC a Arduinem.



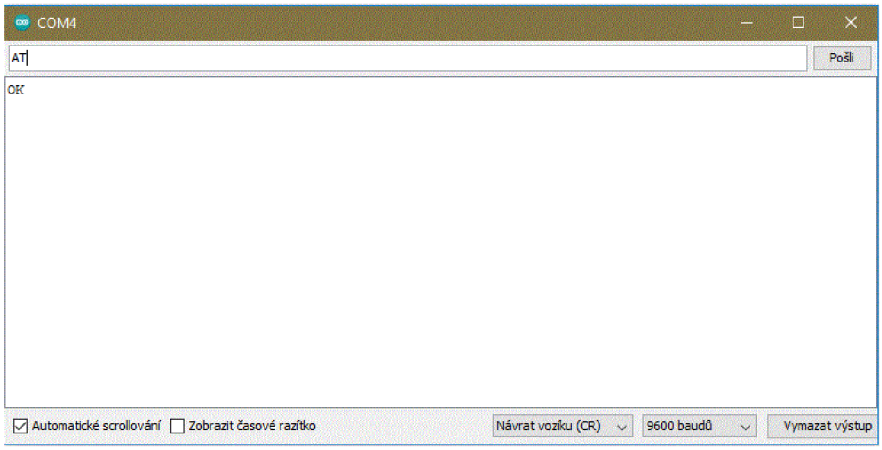
Obrázek 6.1 Základní zapojení - Arduino a Sigfox

Základní program ke zkoušce je uveden na další straně. Je velice jednoduchý. V první řadě připojíme knihovnu k definici sériové komunikace, přiřazení pinů a pomocí těchto náležitostí je definována linka mezi klonem a modulem.

Následuje funkce *setup*. Ta se provede pouze jednou po nahrání nebo zapnutí. Pro jednoduchost testu je potřeba dva řádky. První zahajuje komunikaci mezi PC a Arduinem a druhý mezi Arduinem a Sigfox. Další funkce je zvaná *loop*, ta pak běží nepřetržitě od doby nahrání programu. Podmínky pak jen řeší základní věc, a to pokud přijde informace na jednu linku, přepošle ji na druhou a naopak. Díky tomu jsme schopni poslat příkaz „AT“ z počítače a obdržet zpátky odpověď dálkové komunikace.

```
1:    #include <SoftwareSerial.h>
2:    #define TX 3                  //GPIO 2
3:    #define RX 2                  //GPIO 3
4:    SoftwareSerial Sigfox(RX, TX);
5:    void setup() {
6:      Serial.begin(9600);
7:      Sigfox.begin(9600);
8:    }
9:    void loop() {
10:   //Přeposílání dat z PC do Sigfox Modulu a obraceně
11:     if (Sigfox.available()) {
12:       Serial.write(Sigfox.read());
13:     }
14:     if (Serial.available()) {
15:       Sigfox.write(Serial.read());
16:     }
17:   }
```

Jakmile úspěšně nahrajeme program, můžeme začít se zkouškou. V prostřední Arduino IDE spustíme Serial Monitor a nastavíme podle obrázku níže. Klíčové je nastavit ukončovací znak na „Návrat vozíku“ (Carrige Return – CR) a rychlost jako v programu, 9600 baudů. Poté můžeme zaslat příkaz „AT“, pokud se vrátí hodnota „OK“, je všechno v pořádku.

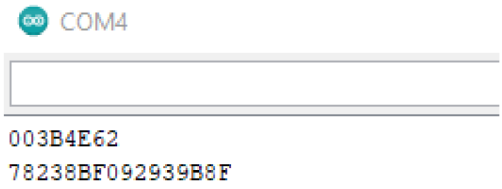


Obrázek 6.2 Serial monitor - Základní test

6.2 Test komunikace s IoT serverem

Po úspěšném otestování komponent je další krok registrace našeho konkrétního Sigfox modulu. Ten na sobě má identifikační číslo (ID) a autorizační kód portu (Porting Authorization Code). Modul použitý v této práci obsahoval v balení navíc anténu, připojovací drátky a především roční licenci na provoz, která se spustí po zaregistrování.

Čísla ID a PAC byly také v obsahu balení, nicméně to není vždy ten případ. Tyto údaje můžeme získat také pomocí předchozího programu. A to díky příkazům „AT$I=10“ a „AT$I=11“. První nám vrátí identifikační číslo a druhý příkaz pošle zpátky hodnotu autorizační. Na obrázku níže je výsledek z tohoto konkrétního zařízení.



Obrázek 6.3 Výpis ID a PAC pomocí AT příkazů

Pro naši potřebu je tato metoda dostačující. Údaje potřebujeme jen na registraci a aktivaci. Existují i jiné způsoby, například využití knihovny *Sigfox.h.* Má v sobě řadu funkcí a dvě z nich vrací právě potřebná data, jsou to *SigFox.ID()* a *SigFox.PAC().*

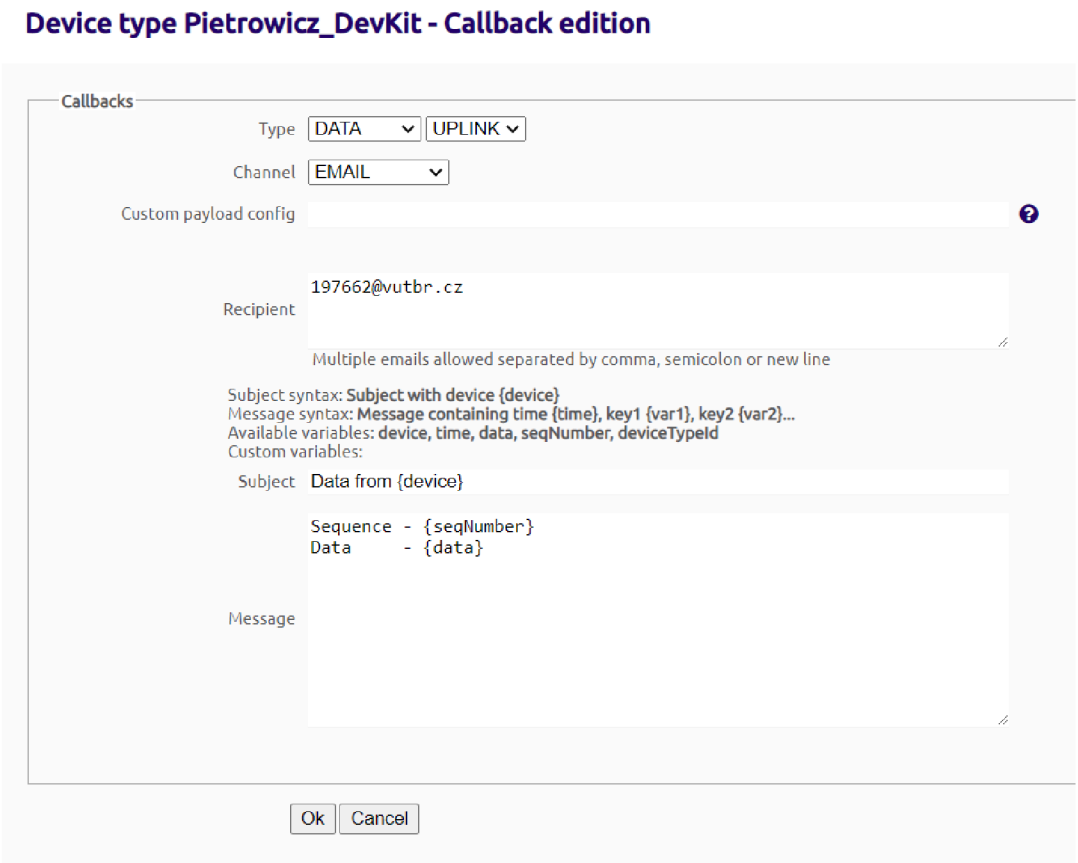
Proces registrace vyžaduje mimo výše zmíněná data také údaje o uživateli. Zařízení lze přenést později na jiného uživatele. Backend portál pro přenášená data přináší řadu údajů a funkcí pro provoz.

Jedna z důležitých je funkce callback, která umožňuje přenést data z Sigfox serveru na jiný server, případně zaslat na email. Stejný způsob přeposlání informací je i pro aktuální události, tím je například chyba v kontinuitě zpráv, obnova smlouvy, přetečení kapacity zpráv za den a mnoho dalších.

6.2.1 Callback z IoT cloudu

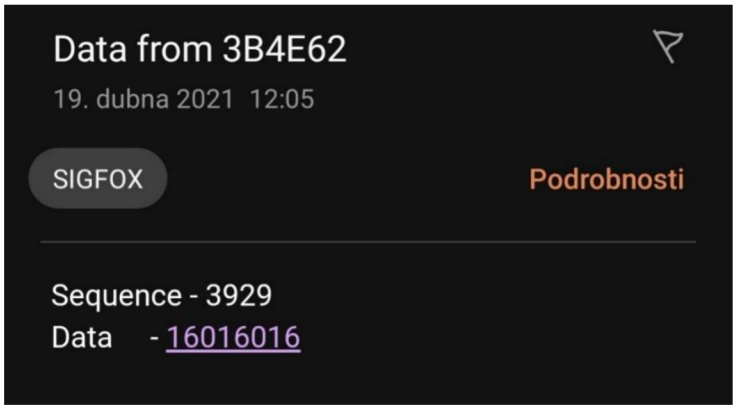
Vytvoření přeposlání dat z IoT je relativně snadné. V první řadě se nastaví, zda chceme data, údaje o chybě nebo servisní informace. Vybrané údaje můžeme poslat buď na email, server nebo pomocí odkazu, také je možné je rovnou poslat dekódované.

Na další straně je obrázek, který popisuje nastavení k zaslání dat na email. V tomto případě není potřeba nic rozkódovat, zařízení nám bude posílat informaci již v podobě celého čísla. Do předmětu emailu je vloženo jméno zařízení, které lze kdykoliv změnit. Zpráva obsahuje na prvním řádku číslo sekvence a na dalším řádku přijatý rámec dat.



Obrázek 6.4 IoT Callback – Email

Vytvoření pak potvrdíme pomocí tlačítka „OK“ na stránce a tím máme zkoušku připravenou. Teď server čeká na data a ta nám pak přepošle pomocí našeho vybraného způsobu. Metod lze mít několik, můžeme je kdykoliv zapnout, upravit nebo smazat.



Obrázek 6.5 Přijatá data na emailu

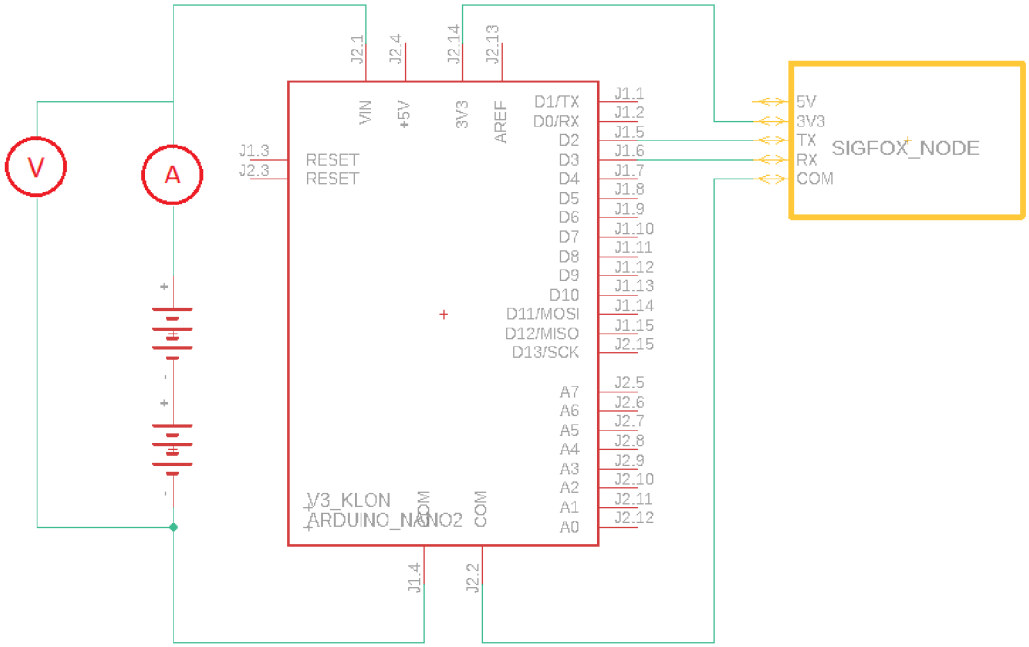
Na výše uvedeném obrázku je přijatý email ze Sigfox sítě, odeslaný z navrženého zařízení.

6.3 Minimalizace odběru proudu

Zde se budeme věnovat softwarové minimalizaci proudové spotřeby a zkusíme teoreticky určit dobu funkčnosti zařízení. Cílem je maximalizovat výdrž baterie. V první řadě se provede měření bez úprav pro získání referenčního bodu. Poté budou prováděny změny za výše zmíněným účelem, dále porovnání a propojení těchto úprav.

Jako měřicí přístroj pro proud je použit multimetr výrobce UNI-T, model UT61A. Ten je schopen měřit jak mili tak i mirko ampéry. Druhý přístroj, jenž je použitý jako voltmetr je opět výrobce UNI-T, ale model UT107. Dvě lithiové baterie v sérii jsou použity jako zdroj, každý článek má 3.6 V a 8500 mAh. Čili napájecí napětí je přibližně 7.2 V a kapacita se nesčítá, zůstává na 8500 mAh.

Níže je schéma zapojení, na kterém se měření provádí. Jedná se o velice jednoduché zapojení. Ampérmetr je zapojen do série za baterie a na vstup Arduina. Voltmetr je připojen, aby měřil napětí na svorkách Arduina. Propojení mezi Sigfoxem a mikrokontrolérem zůstává stejné, jak bylo znázorněno v kapitole 6.1.



Obrázek 6.6 Schéma měřícího zapojení

6.3.1 Spotřeba energie bez minimalizace

V tomto bodě je cílem získat proud, který odebírá zařízení během odesílání dat a odběr v prostoru mezi odesíláním dat. Pro odchozí data je zvoleno čtení hodnoty na analogovém vstupu 0 (A0). K tomuto vstupu nebude připojeno nic, jedná se nám pouze o získání dat, naformátovaní a samotný přenos.

Tento program je použit pro testování:

```
1:    #include <LowPower.h>
2:    #include <SoftwareSerial.h>
3:    #include <stdlib.h>
4:    #define TX 3
5:    #define RX 2
6:    SoftwareSerial SigFox(RX, TX);
7:    bool stav = true;
8:    String sSentVoltVal = "";
9:    String sVoltRaw = "";
10:   Int VoltRdPin = A0;
11:   int VoltValRaw = 0;
12:   void setup() {}
13:   void loop() {
14:   int i = 0;
15:     while(stav==true){
16:   // Spánek 16 sekund
17:           LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
18:   //Měření a převod na zprávu
19:           VoltValRaw = analogRead(VoltRdPin);
20:           sVoltRaw = String(VoltValRaw, DEC);
21:           sSentVoltVal = String("AT$SF=00000" + sVoltRaw);
22:   // Odesílání dat do sítě:
23:           SigFox.begin(9600*2);
24:           delay(500/2);
25:           SigFox.println(sSentVoltVal);
26:           delay(500/2);
27:           SigFox.end();
28:   // Ukončovací podmínka:
29:           i=i+1;
30:           if(i>=2){
31:                   stav = false;
32:                   delay(100);
33:   }}}
```

Má celkem 33 řádků, v první řadě jsou připojeny potřebné knihovny, definice komunikačních vstupů/výstupů pro Sigfox modul a vytvoření sériové komunikace. V dalším bodě je vytvořeno několik důležitých globálních proměnných.

V cyklicky volající funkci je pak jádro celého programu. Po připojení k napětí začne Arduino čekat 30 sekund, poté změří hodnotu na analogovém vstupu, převede ji na řetězec a nakonec do podoby příkazu AT$SF. Podoba „AT$SF=00000“ příkazu není

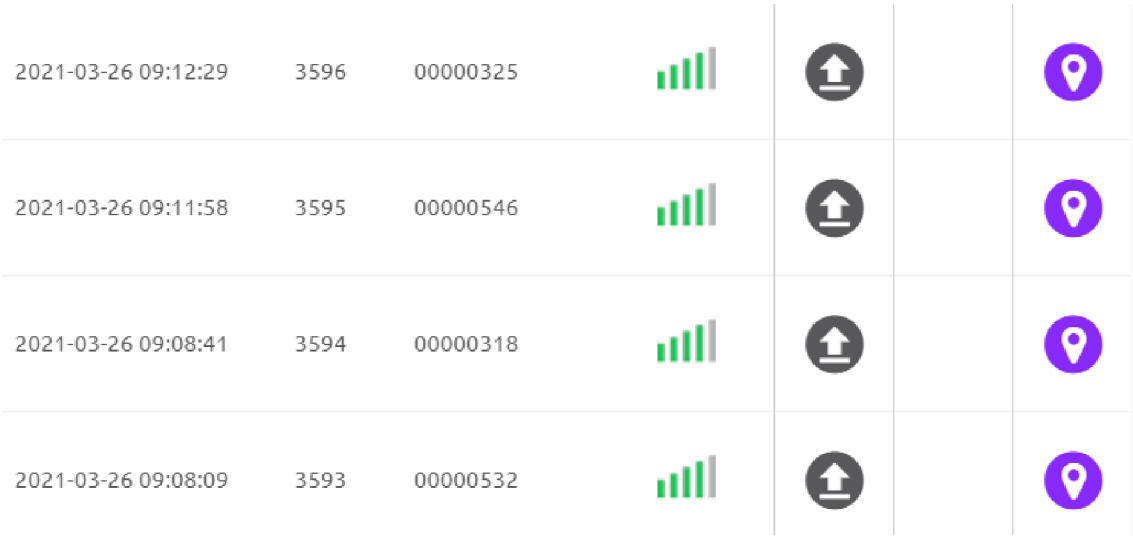
náhodná ale ani dokonalá, pro tuto potřebu testování dostačující. Analogový vstup je desetibitový, to znamená 2^{10} možností. Čili se zapíše hodnota 0 až 1023, tyto čísla popisují rozsah 0 až 5 V. Po chvíli testování vyšlo najevo, že pokud na vstup není připojeno nic, čte se hodnota v rozmezí 200 až 600.

Příkaz SF pak čeká 8 čísel jako rámec odesílaných dat. Proto je použito pět nul a pak připojen řetězec s přečtenou hodnotou, dohromady vždy 8 čísel. Kdyby měl rámec jiný počet cifer, došlo by k chybě a data by nebyla přenesena do Sigfox sítě.

Jakmile je získán finální řetězec, dojde k zapnutí modulu komunikace příkazem *begin(9600)*. Před posláním dat je prodleva 500 ms, je z důvodu inicializace před zahájením přenosu. Zapíšeme příkaz s daty na sériovou linku. Opět počkáme 500 ms, aby byl dostatek času na odeslání dat, a nakonec modul uspíme.

Na závěr je použita ukončovací podmínka, která má za úkol hlídat, aby nedošlo k přenosu více než dvakrát po zapnutí zařízení. Počet rámců, které můžeme odeslat do sítě, je omezen na 140 za den. Kdyby nebyla uvedena ukončovací podmínka a posílání dat by probíhalo co 30 sekund, vyčerpali bychom tento tarif přibližně za 70 minut.

Výsledná přijatá data mají tuto podobu na IoT serveru:



Obrázek 6.7 Přijatá data na IoT serveru

Měření proudu a napětí bylo provedeno 10krát. Výsledné hodnoty jsou v tabulce na další straně.

Tabulka 6.1 Měření - Referenční bod

Číslo zprávy	Proud - delay	Napětí - delay	Proud - odesílání	Napětí - odesílání
[-]	[mA]	[V]	[mA]	[V]
1	42.3	6.8	90.1	6.67
2	43	6.6	89.5	6.7
3	41.8	6.65	89.8	6.71
4	42	6.65	90.2	6.6
5	41.7	6.65	89.3	6.5
6	41.6	6.64	88.2	6.45
7	42.1	6.63	91.1	6.65
8	41	6.76	90.3	6.6
9	40.9	6.7	90.2	6.65
10	40.8	6.76	90.8	6.7

Průměrná hodnota proudu v režimu delay se pohybovala kolem hodnoty 41,72 mAh. V režimu přenosu se maximální hodnota odběru dostala na 91,1 mAh.

Pokud budeme uvažovat, že režim čekání je primární příčina spotřeby a odesílání dat by se dělo pouze jednou za 24 hodin. Můžeme krátký odběr během odesílání zanedbat. Baterie o velikosti 8500 mAh by se vyčerpala za těchto podmínek za 204 hodin, což je přibližně 8,5 dne.

6.3.2 Snížení spotřeby pomocí software úpravy

Spotřebu energie lze snížit i díky úpravě programu pro Arduino. Konkrétně využití režimu spánku nebo zmenšení frekvence krystalu samotného čipu.

Režimů spánku je hned několik, například IDLE, Power-Down, Power-Save a pár dalších. Abychom se do jednoho z výše uvedených režimů dostali, je potřeba změnit hodnotu SMCR (Sleep Mode Control Register) registru. Jeho struktura vypadá následovně:

Tabulka 6.2 Sleep mode registr [15]

Bit	7	6	5	4	3	2	1	0
0x33 (0x53)	-	-	-	-	SM2	SM1	SM0	SE
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Důležité bity jsou 3 až 0, čili SM2, SM1, SM0 a SE. Bit SE, sleep enable, musí být nastaven na logickou jedničku, abychom umožnili jakýkoliv režim spánku. Podle firemní dokumentace se doporučuje nastavit tento bit těsně před zahájením spánku a vynulovat hned po probuzení čipu.

Výběr režimu spánku se volí dle SMx bitů, v tabulce 6.3 je jejich nastavení.

Tabulka 6.3 Sleep mode registr - Režimy Spánku [15]

SM2	SM1	SM0	Mode
0	0	0	Idle
0	0	1	ADC Noise Reduction
0	1	0	Power-Down
0	1	1	Power-Save
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Standby
1	1	1	Extended Standby

Za pomocí již definované knihovny *LowPower* a její úpravy na ATmega328P lze se vyhnout přímým úpravám registru a využit již definované funkce ke vstupu a probuzení ze spánku.

Jednoduchý testovací program pak vypadá takto:

```
1:    #include <LowPower.h>
2:    bool stav = true;
3:    int  i=0;
4:    void setup() {}
5:    void loop() {
6:    while(stav==true){
7:          LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
8:          i=i+1;
9:          if(i>=2){
10:               stav = false;
11:   }}}
```

Jedno úskalí je, že se nelze v této sestavě komponent dostat až na nejnižší možný režim spánku. K probuzení je potřeba využít přerušení z digitálního vstupu a to v této konfiguraci nelze dosáhnout. Probuzení bude provedeno za pomocí přerušení z Watchdog časovače.

Po připojení knihovny *LowPower.h* je potřeba jen zavolat funkci powerDown(). Vybrané vstupní parametry vychází z popisu knihovny pro ATmega328P na vybraném klonu Arduino Nano V3. Parametr SLEEP_8S uvádí, že Watchdog pošle přerušení, a tedy probuzení po 8 vteřinách, to je nejvyšší možná nastavitelná hodnota. Současně se vypne ADC (Analogově/Digitální převodník) a BOD (Brown-Out detection). BOD je součást čipu, která by v režimu spánku brala zbytečně velkou část proudu. Je to obvod, který sleduje úroveň napájení a porovnává ji s fixní spouštěcí hodnotou.

Pro porovnání odběru ve spánku, v přechodu mezi spánky a mimo spánek je použita ukončovací podmínka. Program dvakrát po sobě spustí cyklus spánku na 8 sekund a pak čip neprovádí nic. Naměřené hodnoty jsou v tabulce 6.4.

Tabulka 6.4 Měření - Power-Down režim

Číslo měření	Proud - Idle	Proud - Spánek
[-]	[mA]	[mA]
1	41.2	23.9
2	41.3	23.8
3	40.8	23.9
4	41.6	24
5	40.9	23.9
6	40.9	23.8
7	41.2	24
8	41.1	23.8
9	41.1	24
10	41.3	23.9

Spotřeba energie byla snížena díky spánku přibližně o 40%. V době přechodu mezi prvním a druhým spánkem nedošlo k navýšení odběru, čili po dobu 16 vteřin byl průměrný odběr přibližně 24 mA. K vyčerpání baterie by došlo za necelých 15 dní. Pokud se neprováděla žádná instrukce, byl odběr totožný jako u měření referenčního bodu ve chvíli mimo odesílání dat.

Další úspory energie lze dosáhnout nižší taktovací frekvencí. Ta však ovlivní chování synchronních periferií, na tomto zařízení konkrétně sériovou komunikaci se SigFox modulem.

Změnu frekvence provedeme přepsáním hodnoty v registru CLKPR (Clock Prescale Registr), ten se skládá z 8 bitů.

Tabulka 6.5 CLKPR registr [15]

Bit	7	6	5	4	3	2	1	0
0x61	CLKPCE	-	-	-	CLKPS3	CLSPS2	CLSPS1	CLSPS0
Read/Write	R/W	R	R	R	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bit zvaný CLKPCE, clock prescale change enable, musí mít vždy hodnotu jedna vždy ve chvíli, kdy provádíme změnu. V dalších čtyřech cyklech je nutné zapsat hodnotu na CLKPSx bity, vynuluje se pak sám.

CLKPSx bity popisují dělící faktor frekvence. Může do nich být zapsáno za běhu programu, aby vyhovovala dané aplikaci. Obecně zvýšením dělícího faktoru se zmenšuje výpočetní rychlost čipu, ta však nemá zásadní vliv na tuto aplikaci.

Klon má krystal s frekvencí 16 MHz. Dělící faktor lze zvýšit až na hodnotu 256. Nejnižší dosažitelná hodnota je 62,5 kHz.

Tabulka 6.6 CLKPR - Dělící faktor a výsledná frekvence [15]

CLKPS3	CLKPS2	CLKPS1	CLKPS0	Dělící faktor [-]	Frekvence [MHz]
0	0	0	0	1	16
0	0	0	1	2	8
0	0	1	0	4	4
0	0	1	1	8	2
0	1	0	0	16	1
0	1	0	1	32	0.5
0	1	1	0	64	0.25
0	1	1	1	128	0.125
1	0	0	0	256	0.0625

Testovací program pak má jen vybrané frekvence.

```
1:      bool stav = true;
2:      void setup() {
3:              CLKPR = 0b10000000;     // Change Enable
4:              CLKPR = 0b00000000;     // Dělící faktor 1
5:      }
6:      void loop() {
7:      while(stav==true){
8:              delay(8000);
9:              CLKPR = 0b10000000;
10:             CLKPR = 0b00000001;   // Dělící faktor 2
11:             delay(8000/2);
12:             CLKPR = 0b10000000;
13:             CLKPR = 0b00000010;   // Dělící faktor 4
14:             delay(8000/4);
15:             CLKPR = 0b10000000;
16:             CLKPR = 0b00000100;   // Dělící faktor 16
17:             delay(8000/16);
18:             CLKPR = 0b10000000;
19:             CLKPR = 0b00000110;   // Dělící faktor 64
20:             delay(8000/64);
21:             CLKPR = 0b10000000;
22:             CLKPR = 0b00001000;   // Dělící faktor 256
23:             delay(8000/256);
24:              if(stav==true){              // Ukončovací podmínka
25:                      stav = false;
26:                      CLKPR = 0b10000000;
27:                      CLKPR = 0b00000000;}}}
```

V programu na přechozí straně dochází k postupným změnám. Frekvence se sníží, provede se prodleva, nastavena vždy na 8 sekund, a po provedení poslední změny se nastaví zpět do výchozího režimu.

Použité dělící faktory jsou 2, 4, 16, 64 a 256. Dopad nižší frekvence je pozorovatelný na delay funkci. Vždy je potřeba patřičně snížit také její hodnotu. Jestliže by zůstala vždy původní hodnota, násobila by se faktorem zpomalení. V případě 256 by doba čekání byla 256 * 8 sekund, což je přibližně 30 minut. Měření by bylo zbytečně příliš dlouhé.

Ukončovací podmínka je rozšířena o návrat do výchozího režimu 16 MHz. V několika článcích a fórech byl zmíněn problém s příliš nízkou frekvencí a čip se pak stal neprogramovatelným. Komunikace mezi programovacím rozhraním a čipem není uzpůsobena na tak nízkou výpočetní rychlost.

Tabulka 6.7 Měření - Snížení frekvence

Číslo měření	Proud - faktor 2	Proud - faktor 4	Proud - faktor 16	Proud - faktor 64	Proud - faktor 256
[-]	[mA]	[mA]	[mA]	[mA]	[mA]
1	37.5	35.4	32.7	31.9	31.7
2	37.5	35.3	32.9	32.1	31.7
3	37.4	35.8	33	32	31.8
4	37.8	35.7	32.7	31.8	31.9
5	37.6	35.6	32.8	32.1	31.7
6	37.5	35.8	32.7	31.9	31.8
7	37.6	35.4	32.9	31.9	31.9
8	37.5	35.6	32.9	32.1	31.7
9	37.5	35.5	32.8	32	31.8
10	37.5	35.6	32.9	31.9	31.8

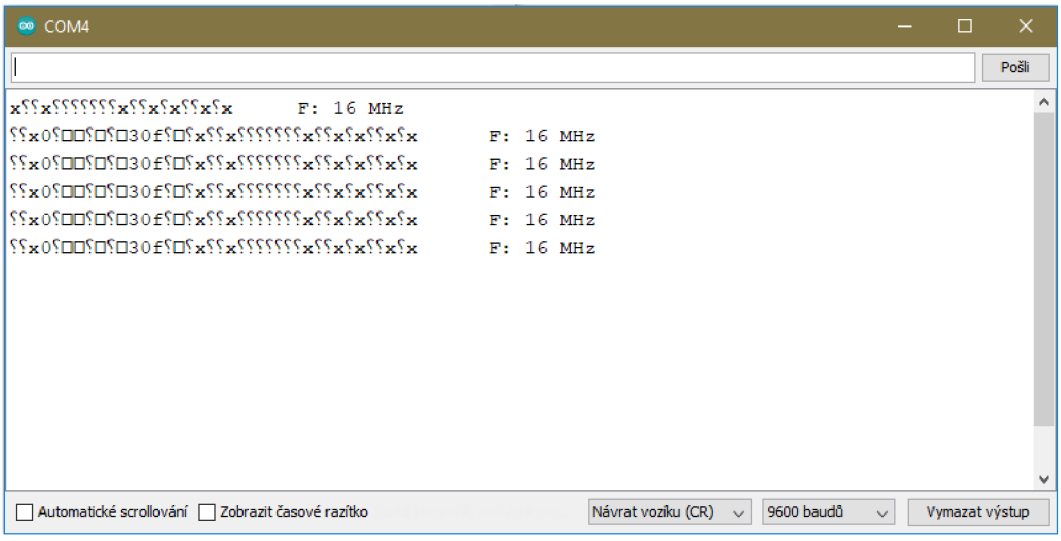
Naměřené hodnoty ukazují, že pomalejší čip je méně náročný na spotřebu energie. Rozdíl mezi faktorem 16 a 256 se pohybuje kolem 1 mA. A rozdíl mezi faktorem 1 (referenční bod) a 256 je přibližně kolem 10 mA. Úspora je tedy 20-25 %.

V dalším kroku je potřebné zjistit ovlivnění komunikace po sérové lince, jelikož Sigfox modul je definovaný pevně na rychlost přenosu 9600 Baudů. Dopad a řešení otestujeme mezi PC a Arduinem.

Testovaný faktor bude 1, 2 a 4, což je v podobě frekvence 16 MHz, 8 MHz a 4 MHz. Přenosová rychlost bude zatím ponechána na 9600 Baudech. Kód je ponechán zjednodušený. V *setup* funkci se pouze zahájí start komunikace. V cyklické funkci se mění rychlost čipu a odesílá zpráva s informací o zvolené frekvenci. Mezi každým odesláním je prodleva 1 sekunda. Mezi zprávou a změnou je prodleva 50 ms, aby došlo ke změně a čip byl během přenosu opravdu ve správné rychlosti. Program je uveden na další straně.

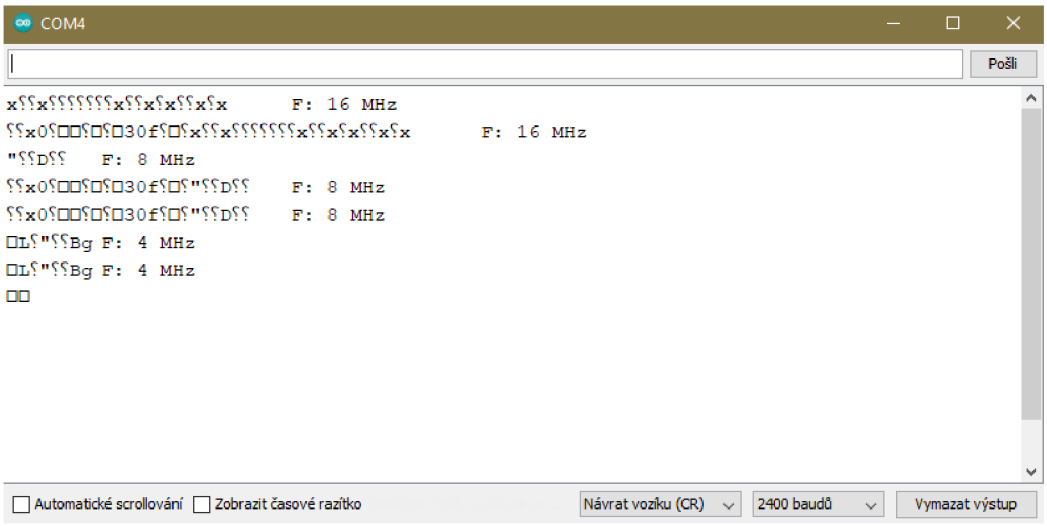
```
1:      #include <SoftwareSerial.h>
2:      void setup() {
3:              Serial.begin(9600);
4:      }
5:      void loop() {
6:              CLKPR = 0b10000000;
7:              CLKPR = 0b00000000;
8:              delay(50);
9:              Serial.write("\tF: 16 MHz\n");
10:             delay(1000);
11:             CLKPR = 0b10000000;
12:             CLKPR = 0b00000001;
13:             delay(50/2);
14:             Serial.write("\tF: 8 MHz\n");
15:             delay(1000/2);
16:             CLKPR = 0b10000000;
17:             CLKPR = 0b00000010;
18:             delay(50/4);
19:             Serial.write("\tF: 4 MHz\n");
20:             delay(1000/4);}
```

Po nahrání do Arduina otevřeme Sériový monitor s rychlostí 9600 Baudů, ukončovací znak CR (Návrat vozíku) a zkontrolujeme výstup. Přijaté řetězce uvedeny na obrázku níže.



Obrázek 6.8 Sériový monitor s nižší frekvencí čipu

Správně odeslaný řetězec je pouze s faktorem 1. Zpomalené dva nejsou čitelné. Abychom získali beze změny programu další dva, je potřeba změnit pouze rychlost v monitoru. A to při faktoru 2 na 4800 Baudů a při faktoru 4 na 2400 Baudů. Toto je provedeno postupně a výstup pak vypadá podle obrázku na další straně.

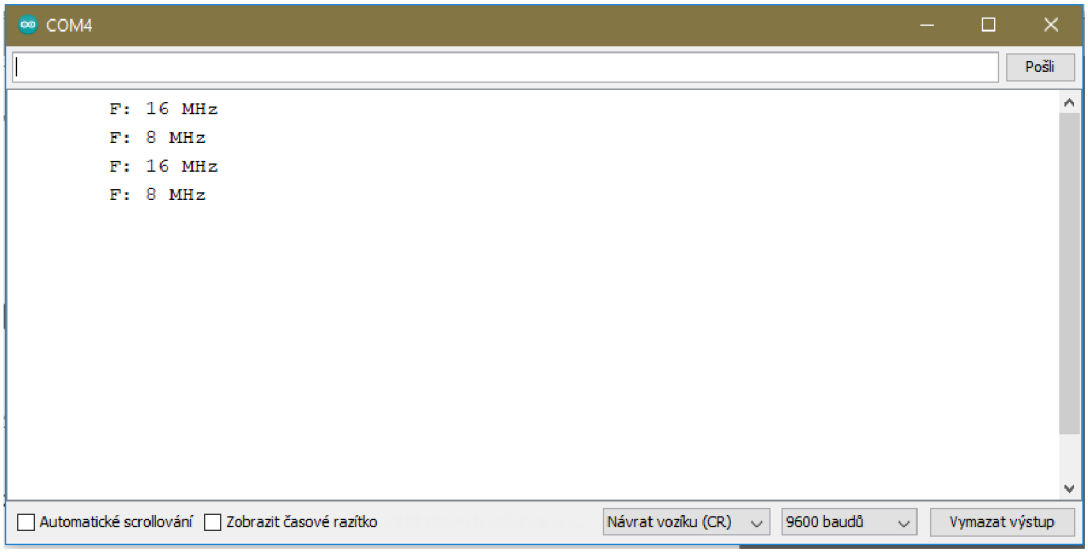


Obrázek 6.9 Sériový monitor zpomalení čipu, změna rychlosti přijmu

Protože nemůžeme změnit rychlost na Sigfox modulu, je potřeba upravit program tak, aby odesílání bylo vždy nastaveno na 9600 Baudů. Nová verze programu je rozšířena o změnu rychlosti sériové linky, rychlost násobíme použitým faktorem u jednotlivé zprávy. Aby byla změna použita, je třeba vždy ukončit a poté zahájit komunikaci.

```
1:    #include <SoftwareSerial.h>
2:    void setup() {}
3:    void loop() {
4:    Serial.begin(9600);
5:    CLKPR = 0b10000000;
6:    CLKPR = 0b00000000;
7:    delay(50);
8:    Serial.write("\tF: 16 MHz\n");
9:    delay(1000);
10:   Serial.end();
11:   Serial.begin(9600*2);
12:   CLKPR = 0b10000000;
13:   CLKPR = 0b00000001;
14:   delay(50/2);
15:   Serial.write("\tF: 8 MHz\n");
16:   delay(1000/2);
17:   Serial.end();
18:   Serial.begin(9600*4);
19:   CLKPR = 0b10000000;
20:   CLKPR = 0b00000010;
21:   delay(50/4);
22:   Serial.write("\tF: 4 MHz\n");
23:   delay(1000/4);
24:   Serial.end();  }     // Ukončení komunikace
```

Výstup v monitoru pak ponecháme na 9600 Baudech. Na sériové lince vidíme, že řešení z části funguje.



Obrázek 6.10 Výstup sériové linky po úpravě

Posílají se řetězce pouze s faktorem 1 a 2. Faktor 4 má přenos nastaven na 9600 * 4 Baudů, což je 38400 Baudů. To je příliš veliká rychlost na zpomalený čip o frekvenci 4 MHz. Jinými slovy Arduino je pomalé a vyšší faktor než 2 se nehodí k implementaci Sigfox. Úspora energie bude s tímto limitem přibližně 10 %, okolo 4 mA.

Další krok je zjištění vlivu snížení výpočetní rychlosti na spotřebu energie v režimu spánku. Použijeme tady jedinou možnost, tou je dělící faktor 2. Spánek je ponechán v původní podobě. Program je uveden níže.

```
1:    #include <LowPower.h>
2:    bool stav = true;
3:    int  i = 0;
4:    void setup() {
5:    CLKPR = 0b10000000;
6:    CLKPR = 0b00000001;
7:    }
8:    void loop() {
9:    while(stav==true){
10:         LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
11:         i=i+1;
12:         if(i>4)
13:              stav = false;
14:   }}
```

Frekvence čipu je upravena hned ve funkci *setup* a v cyklické je opakovaně spínán spánek na 8 sekund. Spánek je spínán 5krát po sobě, celkem na 40 sekund.

Naměřené hodnoty ve spánku a mimo něj jsou zapsány v tabulce 6.7.

Tabulka 6.8 Měření - Propojení spánku a nižší rychlosti čipu

Číslo měření	Proud - Idle	Proud - Spánek
[-]	[mA]	[mA]
1	37.5	23.05
2	36.9	23.04
3	37.2	23.05
4	37.1	23.03
5	37.3	23.05
6	37.5	23.04
7	37.5	23.04
8	37	23.05
9	36.9	23.04
10	37.5	23.03

Proud mimo spánek je obdobný jako předtím, zde podle očekávání není žádná změna. V režimu spánku jsme dosáhli snížení o téměř 1 mA.

Poslední softwarová úprava se týká Sigfox čipu, ten samotný lze také uvést do spánku a obdobně jako Arduino, má dva režimy. První lze zpět probudit odesláním po sériové lince „\n“, tzv. break a druhý způsob je resetováním připojení k napájení. Nicméně první režim má odběr pouze 2 µA, na kapacitní životnost baterie nemá významný vliv. Pro úsporný režim stačí poslat z Arduino AT příkaz „AT$P=1“ nebo „AT$P=2“ k volbě typu spánku. [1]

6.3.3 Implementace softwarových úprav

V této podkapitole budou implementovány úpravy programu z předchozích stránek do referenční verze programu a následné změření rozdílů spotřeby.

```
1:  #include <LowPower.h>
2:  #include <SoftwareSerial.h>
3:  #include <stdlib.h>
4:  #define TX 3                    // GPIO 2
5:  #define RX 2                    // GPIO 3
6:  SoftwareSerial SigFox(RX, TX);
7:  bool stav = true;
8:  String sSentVoltVal = "";
9:  String sVoltRaw    = "";
10: int VoltRdPin  = A0;
11: int VoltValRaw = 0;
```

```
12:  void setup() {
13:  CLKPR = 0b10000000;
14:  CLKPR = 0b00000001;
15:  SigFox.begin(9600*2);
16:  SigFox.println("AT$P=1");
17:  SigFox.end();
18:  }
19:  void loop() {
20:  int i = 0;
21:  while(stav==true){
22:          LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
23:          LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
24:  // Měření a převod na zprávu:
25:          VoltValRaw = analogRead(VoltRdPin);
26:          sVoltRaw = String(VoltValRaw, DEC);
27:          sSentVoltVal = String("AT$SF=00000" + sVoltRaw);
28:  // Odesílání dat do sítě:
29:          SigFox.begin(9600*2);
30:          SigFox.print("\n");
31:          delay(500/2);
32:          SigFox.println(sSentVoltVal);
33:          delay(500/2);
34:          SigFox.println("AT$P=1");
35:          SigFox.end();
36:  // Ukončovací podmínka: Dvě odeslání po startu zařízení, poté konec
37:          i=i+1;
38:          if(i>=2){
39:                  stav = false;
40:                  delay(100);
41:  }}}
```

Původní verze byla rozšířena o knihovnu LowPower, abychom byli schopni zavolat funkci spánku. Následně bylo přidáno přetaktování na 8 MHz, což lehce sníží spotřebu běhu programu i ve spánku a rovněž klesne úroveň napájecího napětí. Čili je možnost jej snížit a s tím také klesne celková spotřeba. Jako poslední je pak nahrazení prodlevy mezi zprávami spánkem na celkem 16 vteřin. Zároveň nesmíme zapomenout zvýšit rychlost přenosu po sérové lince pro Sigfox a naopak snížit veškeré prodlevy. Vše dle použitého dělícího faktoru 2.

A jako poslední úprava je přidání uspání Sigfox, to je provedeno po startu zařízení v *setup* funkci, řádek 15-17. Probuzení je pak voláno na řádku 30 a opakované volání spánku pak po dokončení vysílání na řádku 34.

Obrázek 6.11 Měření - Úprava referenční verze

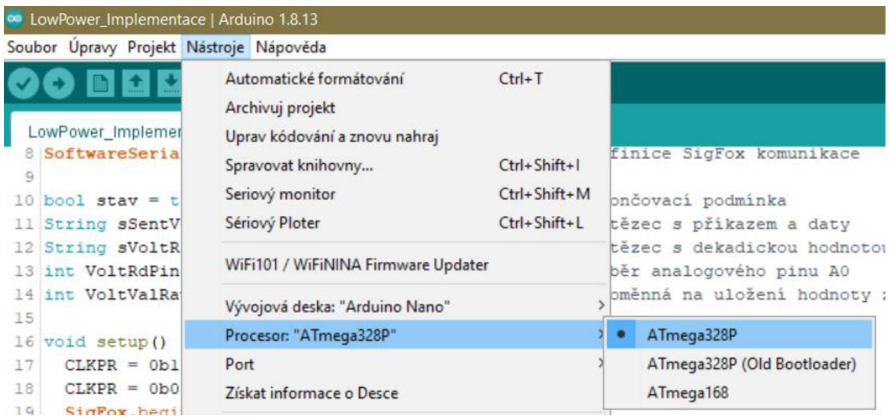
Číslo zprávy	Proud - spánek	Napětí - spánek	Proud - odesílání	Napětí - odesílání
[-]	[mA]	[V]	[mA]	[V]
1	22.1	6.84	86.6	6.73
2	22.09	6.9	87.5	6.81
3	22.08	6.91	86.6	6.7
4	22.07	6.9	86.8	6.69
5	22.08	6.91	87.2	6.71
6	21.9	6.88	86.7	6.72
7	22	6.86	87	6.66
8	22	6.88	87.8	6.68
9	22	6.68	87.4	6.7
10	22.1	6.76	86.9	6.71

Úrovně napětí zůstaly stejné, zde by se také nemělo nic změnit. Proud klesl tedy jak pro prodlevu mezi zprávami tak i při odesílání zpráv do Sigfox sítě. Prodleva má nyní okolo 22 mA, úspora je přibližně 19 mA. Při zdroji s 8500 mAh jsme získali navíc celkem 7 dní provozu. Celkem tedy 16 dní, pokud zanedbáme vteřinové odesílání dat za jeden den. Procentuálně se úspora pohybuje na 50 % z původní referenční hodnoty.

Ani z daleka jsme se nepřiblížili cíli provozu minimálně dvou let. V tomto případě se bude jednat o problém s deskou Arduina. Na tomto problému bude v následující kapitole vidět rozdíl mezi klonem a originálním čipem. Mimo jiné klon používá ATmega328, kdežto originál má ATmega328P. Zásadní rozdíl je v samotné spotřebě energie, jinak jsou plně zaměnitelné.

6.3.4 Snížení spotřeby pomocí úpravy hardware komponent

V první řadě dojde k výměně klonu za originální čip a zkoušky běhu upravené verze referenčního programu, zda dojde k odesílání dat. Jediná úprava v programovacím prostředí je změna procesoru. Klon měl původně tzv. old bootloader, nově se v Arduino IDE nastaví ATmega328P, více na následujícím obrázku. Jiné změny nejsou potřeba.



Obrázek 6.12 Změna procesoru

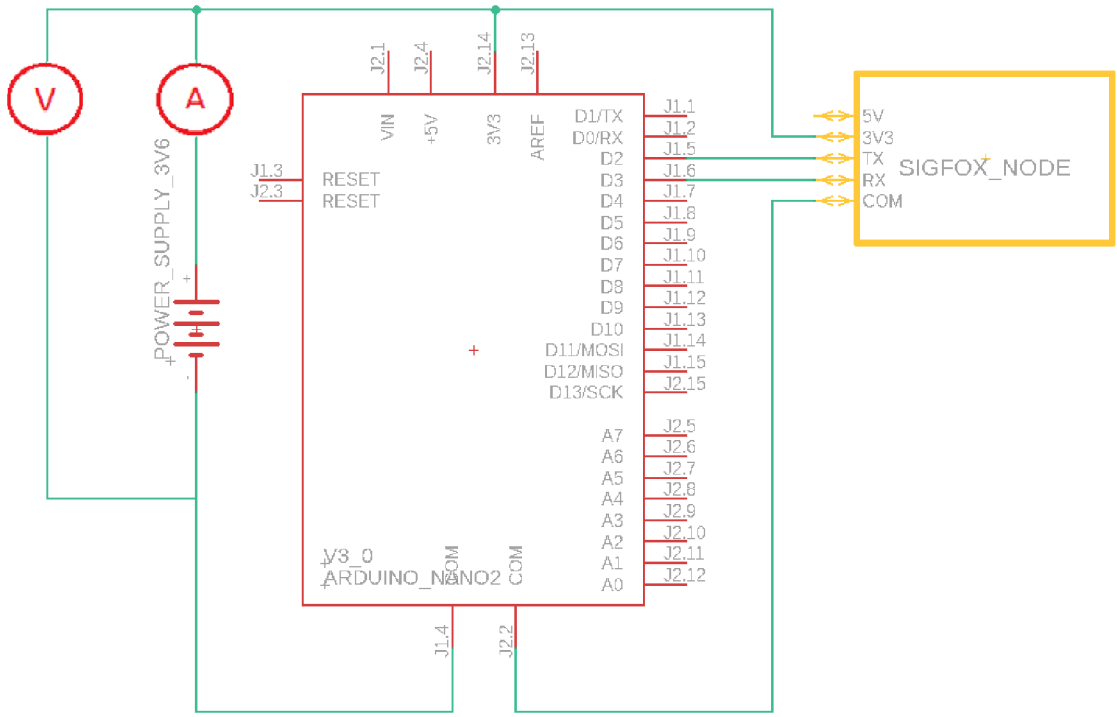
Po nahrání na originální desku a záměně za klon v zapojení vypadá spotřeba zařízení o něco lépe.

Tabulka 6.9 Měření - Originál Arduino místo klonu

Číslo zprávy	Proud - spánek	Napětí - spánek	Proud - odesílání	Napětí - odesílání
[-]	[mA]	[V]	[mA]	[V]
1	18.51	6.82	81.2	6.65
2	18.6	6.8	80.9	6.71
3	18.48	6.78	81	6.68
4	18.65	6.75	80.5	6.67
5	18.55	6.88	80.6	6.7
6	18.61	6.86	81.5	6.7
7	18.45	6.76	82	6.7
8	18.68	6.75	81.4	6.65
9	18.55	6.68	80.6	6.73
10	18.49	6.76	80.7	6.69

Lze vidět znatelné snížení spotřeby o přibližně 4.5 mA jak při spánku, tak i odesílání dat. Nicméně nám zatím originál získal navíc pouze 3 dny provozu navíc, celkem tedy 19 dní.

Největší úsporu získáme díky omezení zdroje pouze na jeden článek o napětí 3.6 V. To je možné hlavně díky snížení rychlosti čipu. Jelikož jsou v původním zapojení baterie přivedené k pinu VCC, který vede do regulátoru napětí, je třeba připojit k novému pinu 3V3, nepotřebujeme dále měnit 7.2 V na 5 V. Zapojení s měřicími přístroji je uvedeno na další stránce.



Obrázek 6.13 Zapojení s jedním článkem a originálem Arduino

V této upravené sestavě má zařízení velice nízkou spotřebu, odběr v úsporném režimu klesl téměř 4 krát oproti původnímu zapojení. Posílání dat kleslo o 30 mA. Nyní zařízení vydrží teoreticky 80 dní provozu.

Tabulka 6.10 Měření - Originál Arduino, jeden 3.6 V článek

Číslo zprávy	Proud - spánek	Napětí - spánek	Proud - odesílání	Napětí - odesílání
[-]	[mA]	[V]	[mA]	[V]
1	4.31	3.65	50.2	3.64
2	4.35	3.66	51.2	3.64
3	4.25	3.6	50.4	3.65
4	4.26	3.64	50.5	3.63
5	4.25	3.63	51.1	3.65
6	4.25	3.64	52	3.66
7	4.32	3.64	52.1	3.64
8	4.3	3.64	50.5	3.65
9	4.3	3.65	51.1	3.65
10	4.31	3.66	51.1	3.64

Samotné Arduino má komponenty navíc, jež zbytečně značně spotřebovávají energii. Jednou z nich je například dioda, která svítí nepřetržitě a nelze ji odpojit pomocí softwarové úpravy, je napojena přímo ke zdroji napětí. Při odstranění je velké riziko jejího poškození vlivem tepla, ale bude-li sejmuta trvale, poškození provozu vadit nebude.

Další součástka je regulátor napětí, který se sice díky připojení na pin 3V3 obchází, ale i přesto má proudový odběr.

Díly lze odstranit za pomocí pájky nebo horkovzdušné pistole. Vybraná metoda v tomto případě je horkovzdušná pistol, pokryje větší plochu a značně usnadní odebrání diody a regulátoru. K sejmutí je vhodné použít také pinzetu.

Výsledek lze vidět v tabulce 6.10 níže.

Tabulka 6.11 Měření - Odebrání LED a regulátoru z Arduina

Číslo zprávy	Proud - spánek bez LED	Proud - odesílání bez LED	Proud - spánek bez LED a regulátoru	Proud - odesílání bez LED a regulátoru
[-]	[mA]	[mA]	[µA]	[mA]
1	2.21	50.5	65.1	49.8
2	2.23	50.4	64.2	48.9
3	2.24	50.4	63.2	49.9
4	2.19	51.2	60.1	50.1
5	2.18	50.9	59.5	49.7
6	2.2	50.5	64.3	49.8
7	2.21	50.3	60.1	49.8
8	2.24	50.1	60.5	49.6
9	2.25	50.1	59.6	49.7
10	2.18	50.1	59.7	49.5

Pro porovnání byla prvně sejmuta LED dioda a poté i regulátor. Na první dojem působí, že odebrání regulátoru způsobilo nárůst proudového odběru. Ale v pravé části tabulky je menší jednotka ve sloupci pro úsporný režim.

Dioda měla spotřebu přibližně 2 mA, jelikož svítila nepřetržitě během zapnutí napájení zařízení. Regulátor měl podobný dopad jako dioda. Celkově se snížila spotřeba na velice nízkou hodnotu, a tím se prokázala vhodnost vybraných komponent.

S touto spotřebou je zařízení teoreticky schopno fungovat s lithiovým článkem s kapacitou 8500 mAh po dobu více než 15 let. Snížení spotřeby se zdařilo, zbývá navrhnout vhodný program, který bude ve finálním zařízení.

6.4 Konečná verze programu

Zařízení bude posílat data minimálně jednou za den. Žádaný údaj, napětí v plynovém potrubí vůči zemi, bude měřen 3 krát během dne v intervalu přibližně 8 hodin. Jakmile zařízení provede třetí měření, odešle data do Sigfox sítě v podobě minimální, průměrné a maximální změřené hodnoty. Posílaná data již budou převedena na volty, bez desetinného čísla.

Pro výdrž baterie bude zařízení pouze data posílat, nebude možné poslat zprávu opačně s požadavkem na změření a poslání aktuální hodnoty, interval měření nemusí být přesně dán, postačí orientační čas 8 hodin. Program, jenž bude použitý v zařízení, vypadá následovně.

```
1:    #include <LowPower.h>
2:    #include <SoftwareSerial.h>
3:    #include <stdlib.h>
4:    #define TX 3
5:    #define RX 2
6:    SoftwareSerial SigFox(RX, TX);
7:    String sSentVals   = "";
8:    String sMin        = "";
9:    String sMax        = "";
10:   String sAvg        = "";
11:   int VoltRdPin  = A0;
12:   int VoltValRaw = 0;
13:   int Volt_raw[3];
14:   void setup() {
15:   CLKPR = 0b10000000;
16:   CLKPR = 0b00000001;    // Dělící faktor 2
17:   SigFox.begin(9600*2);
18:   delay(50/2);
19:   SigFox.println("AT$SF=00000000");
20:   delay(500/2);
21:   SigFox.println("AT$P=1");
22:   SigFox.end();
23:   }
24:   void Sleep_8h(){
25:   int i = 0;
26:   while(i<3600){  // ( 8h * 60 min * 60s / 8s) = 3600
27:         LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
28:         i = i + 1;
29:   }}
30:   void Min_Max_Avg(){
31:   int i=0,volt_min=Volt_raw[0],volt_max=Volt_raw[0],volt_avg = 0;
32:   while(i<3){
33:         volt_avg = volt_avg + Volt_raw[i];
34:         if(Volt_raw[i]<volt_min) volt_min = Volt_raw[i];
35:         if(Volt_raw[i]>volt_max) volt_max = Volt_raw[i];
36:         i = i + 1;
37:   }
```

```
38:     sMin = String((float(volt_min)/1023)*2.9,1);
39:     sMax = String((float(volt_max)/1023)*2.9,1);
40:     sAvg = String(((float(volt_avg)/3)/1023)*2.9,1);
41:     }
42:     void loop(){
43:     int i = 0;
44:     while(i<3){
45:             Sleep_8h();
46:             Volt_raw[i] = analogRead(VoltRdPin);
47:             i=i+1;
48:     }
49:     Min_Max_Avg();
50:     sSentVals = String("AT$SF="+sMin+"0"+sMax+"0"+sAvg);
51:     sSentVals.replace(".","");
52:             SigFox.begin(9600*2);
53:             SigFox.print("\n");
54:             delay(500/2);
55:             SigFox.println(sSentVals);
56:             delay(500/2);
57:             SigFox.println("AT$P=1");
58:             SigFox.end();
59:     }
```

Celý program se vešel na celkem 59 řádků. V úvodu jsou typicky připojeny knihovny, definované a inicializované globální proměnné a definování sériové komunikace pro Sigfox modem. Funkce *setup* obsahuje přetaktování na 8 MHz, odeslání kontrolní zprávy a uvedení modemu do spánku. Následuje funkce *Sleep_8h*, v ní je proveden spánek na čas přibližně 8 hodin. Vzorec pro výpočet opakování je jednoduchý.

1. $(8h * 60\,\text{min} * 60\text{sec})/8s = 3600\ opakování$

Vynásobíme čas, kterého chceme dosáhnout a vydělíme intervaly spánku. Celkem se provede spánek 3600krát po osmi vteřinách. Jako další je funkce k vyhledání minima, maxima a výpočtu průměru pro naše tři změřené hodnoty. V ní jsou rovněž hodnoty přetypované na proměnnou řetězce pro usnadnění poslání. Jelikož máme původně hodnotu v datovém typu celého čísla, je třeba jí převést na desetinné číslo a hned na řetězec.

Jako poslední je hlavní cyklická funkce. V ní je prvně volán spánek na 8 hodin, poté změření hodnoty na analogovém vstupu. Tyto kroky se provedou třikrát, po třetím spánku a naměření pokračuje volání funkce pro určení minima, maxima a průměru. Výsledek se pak zapíše do podoby příkazu AT. Jelikož v řetězci stále visí desetinná čárka, je potřeba všechny odstranit pomocí nahrazení znaku desetinné čárky za prázdný znak.

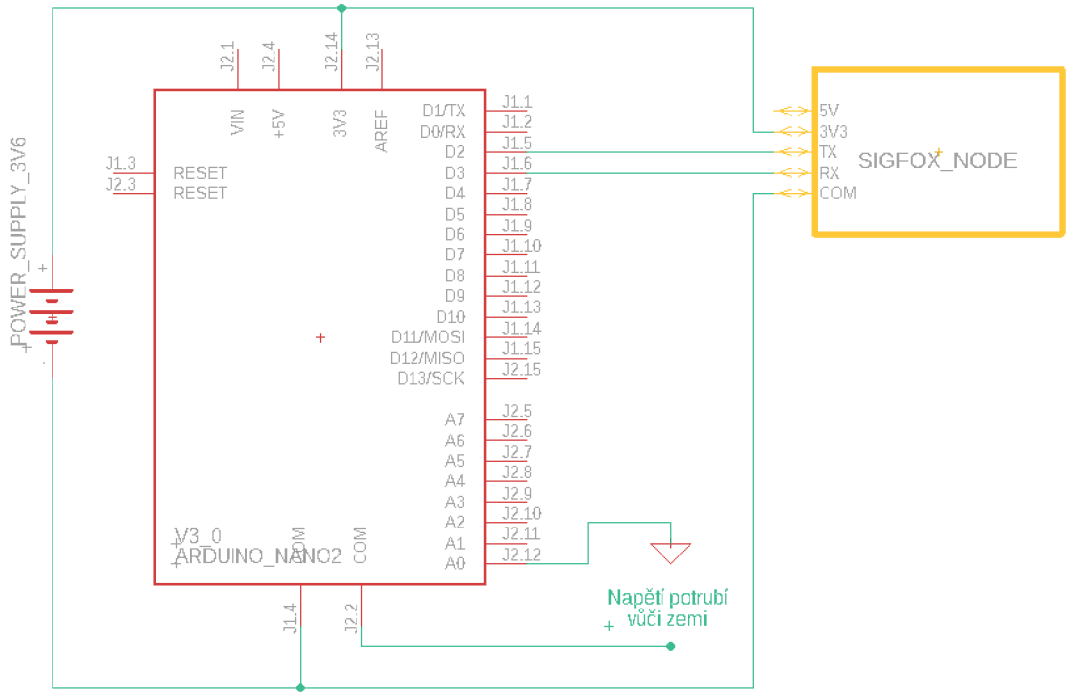
Jakmile se všechny zmíněné kroky provedou, dojde k poslání dat do Sigfox sítě. Poté se celá cyklická funkce opakuje znovu.

Čili Arduino po připojení k napájení pošle kontrolní zprávu s nulovými daty a pak se provede měření, přibližně po prvních 24 hodinách přijdou první data ke zpracování. Další data pak chodí co 24 hodin.

6.5 Konečné zapojení komponent

K sestavení stačí dvě hlavní komponenty, Arduino Nano a Sigfox modul. Jako napájecí zdroj je použit lithiový článek o kapacitě 8500 mAh a napětí 3.6 V. Všechny zmíněné součásti jsou vhodné až do -30 °C, jsou ideální pro venkovní použití.

Další součástky jsou primárně k usnadnění zapojení a sestavení zařízení. Patří mezi ně terminál pro Arduino, držák na baterii a samozřejmě anténa.



Obrázek 6.14 Finální sestava HW

Z desky Arduina bylo nutné odstranit dvě věci. První byla Power LED dioda, která svítila nepřetržitě při napájení zařízení, a regulátor napětí.

Tabulka 6.12 Měření - Konečná podoba zařízení

Číslo měření [-]	Proud - spánek [µA]	Proud - odesílání [mA]
1	65.1	50.1
2	65.3	49.8
3	65.4	49.8
4	64.5	50.1
5	64.9	50.2
6	66	50.1
7	64.8	50.1
8	64.8	48.5
9	65.1	48.5
10	65.1	50.1

Změřené hodnoty téměř odpovídají hodnotám na testovacím programu. Proud během odesílání stěží přesáhne 50 mAh a ve spánku si zařízení bere pod 70 µA. K výpočtu průměrné spotřeby během 24 hodin byl použit tento vzorec.

2. $$\frac{86398s*0{,}07mA+2s*50mA}{86400} = 0{,}0712mA$$

Celková teoretická výdrž na jedné baterii je počítána vzorcem níže.

3. $$\frac{8500mAh}{0{,}0712mA} = 119382h$$

Zařízení by mělo vydržet celkem 119 382 hodin provozu, což je přibližně 5 000 dní nebo 13.5 let. Jedná se pouze o teoretickou hodnotu, vliv zimy snižuje celkovou kapacitu baterie, zároveň sníží i proudový odběr a napětí lithiového článku.

Výše uvedené kroky byly také odzkoušeny i na klonu Arduina. Nicméně spotřeba neklesla pod 10 mA, klon se tedy ukázal jako nepoužitelný v tomto případě. Bylo by nutné využít fotovoltaiku a akumulátor nebo jiný dlouhodobě udržitelný zdroj.

7 ZÁVĚR

Navržené zařízení splnilo základní požadovanou podmínku, a to dlouhodobý dálkový přenos dat. Jeho návrh s sebou přináší řadu výhod, ale také nevýhod.

Mezi výhody lze zařadit malý počet nutných komponent a jejich společnou vlastnost provozní teploty až do -40 °C, čímž se hodí na celoroční použití ve venkovním prostoru. Nemluvě o tom, že Sigfox má také lepší dosah venku a v nezastavěné oblasti. Další velkou výhodou je samozřejmě spotřeba energie a s tím spjatá výdrž zařízení na jedné baterii, teoreticky až 13 let. Také cenu zařízení lze považovat za přívětivou. Cena za hlavní hardware komponenty je přibližně 1500,- Kč. Zařízení také přináší možnost automatizace celé sítě. Nahradí člověka při měření údajů a zároveň je možné díky tomuto kroku i zavedení vhodné logiky řízení ke změně úrovně napětí v potrubí.

Tabulka 7.1 Cena HW komponent

Součástka	Cena [Kč]
Arudino Nano V3.0	659,-
Sigfox modul	529,-
Baterie ER 26500	250,-
Celkem	1438,-

Do nevýhod možno zařadit práci s deskou Arduina. Bylo z ní potřeba odstranit minimálně dvě součástky, LED a regulátor napětí, aby byl dosažen cíl nízké spotřeby. Také bylo potřeba velké opatrnosti kvůli malému rozměru mikrokontroléru a kvůli riziku poškození vlivem tepla při práci s horkovzdušnou pistolí. Mimo tuto skutečnost je také nevýhoda ta, že zařízení kvůli šetření baterie není schopno fungovat obousměrně. V případě přijímání informací má podle datasheetu minimální spotřebu 15 mA. Nelze na tomto návrhu poslat žádost o aktuální hodnotě ani změnit interval posílání dat. V tabulce níže je teoretický výpočet doby funkce pro různé intervaly posílání dat.

Tabulka 7.2 Různé intervaly posílání dat a výdrž baterie

Interval posílání	Průměrná spotřeba [mAh]	Počet let provozu
1x týdně	0.07	13.8
1x denně	0.071	13.6
3x denně	0.073	13.2
každou hodinu	0.098	9.9

Rozdíl v životnosti baterie u zasílání zpráv jednou týdně a 3krát denně je pouze 6 měsíců, tak zásadní vliv to není. Nicméně posílání dat každou hodinu už vezme z celkové životnosti téměř 3 roky. Poslední zásadní nevýhoda je měřitelný rozsah, ten je díky nižšímu napájecímu napětí pouze 0 až -3 V. Toto je sice dostačující, ale lepší by

bylo mít rozsah i na pozitivní stranu, čili +3 až -3 V. V případě změření hodnoty buď pod, nebo nad zvolený interval, zařízení pošle pouze osm nul.

Jiné řešení by mohlo být nahrazení baterie za akumulátor a přidání fotovoltaické desky. Zde by byla pouze obava o dlouhodobý celoroční provoz u akumulátoru, který není stavěný do nízkých teplot. Také by bylo možné i použít jiný přístup do IoT sítě, existuje řada konkurentů vůči Sigfox, například LoraWAN, NB-IoT, GSM a jiné.

Veškerá dříve provedená měření a testování se odehrála za pokojové teploty v domácím prostředí. Zařízení po zapouzdření půjde do testovacího provozu.

Obrázek 7.1 Navržené zařízení

Je zde i tenká spojitost s mou bakalářskou prací, která se zabývala vlivem nepřesností vzorkovací periody na frekvenční analýze. Nepřesnost při měření stejnosměrného napětí nehraje žádnou roli, nicméně bychom mohli aplikovat frekvenční analýzu na přenos v použitém pásmu 868 MHz. Bylo by možné z ní zjistit vliv okolního rušení na výsledné kvalitě a dosahu přenosu. I když jde o malý balík dat, tím spíše by mohlo dojít k jeho neúspěšnému doručení do IoT sítě. Mimo to se práce lehce zabývala i digitalizací a analogovými převodníky, bez kterých bychom nebyli schopni změřit požadovanou hodnotu napětí. [26]

8 SEZNAM POUŽITÉ LITERATURY

[1] LPWAN Co., Ltd. [online katalogový list]. LPWAN SigFox node. ©2016 [cit. 6.4.2021]. Dostupné z: https://www.lpwan.cz/LPWAN_sigfox_node_datasheet_v1.pdf

[2] Sigfox. In: *www.sigfox.com* [online]. SigFox coverage. [cit. 6.4.2021]. Dostupné z: https://www.sigfox.com/en/coverage

[3] Ing. Bc. Ivan Pravda, Ph.D. Nové trendy v elektronických komunikacích, Mobilní a bezdrátové sítě [cit. 15. 2. 2021]. Dostupné z: https://publi.cz/books/236/Impresum.html

[4] AUTOR NEUVEDEN. In: *www.hadex.cz* [online] [cit. 15. 2. 2021]. Dostupný na: https://www.hadex.cz/img/zbozi/m374a.jpg

[5] AUTOR NEUVEDEN. SIM800L Hardware Design V1.00 [online katalogový list]. Copyright © Shanghai SIMcom Wireless Solutions Ltd. 2013. [cit. 15. 2. 2021]. Dostupné z: https://www.hadex.cz/spec/m374a.pdf

[6] T-Mobile. In: *www.t-mobile.cz* [online]. [cit. 16. 2. 2021]. Dostupné z: https://www.t-mobile.cz/podpora/mapa-pokryti

[7] Lora-Alliance. In: *lora-alliance.org* [online]. Members list. lora-alliance.[cit. 16. 2. 2021]. Dostupné z: https://lora-alliance.org/member-directory/

[8] MIKULÁŠEK, M. Bezdrátová senzorická síť využívající LoRa technologii. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2018. 79 s. Vedoucí bakalářské práce doc. Ing. Vladislav Škorpil, CSc.

[9] Starnet. In: *www.starnet.cz* [online]. [cit. 16. 2. 2021]. Dostupné z: https://www.starnet.cz/wp-content/themes/starnetcz/static/img/pokryti_mapa.png

[10] HOPERF. [online katalogový list]. RFM95W/96W/98W .[cit. 16. 2. 2021]. Dostupné z: https://www.hoperf.com/data/upload/portal/20190801/RFM95W-V2.0.pdf

[11] Ericsson Technology. In: *www.ericsson.com* [online]. NB-IOT: A Sustainable Technology for Connecting. [cit. 16. 2. 2021]. Dostupné z: https://www.ericsson.com/49ead5/assets/local/reports-papers/ericsson-technology-review/docs/2016/etr-narrowband-iot.pdf

[12] Vodafone. In: *www.vodafone.cz* [online]. [cit. 16. 2. 2021]. Dostupné z: https://www.vodafone.cz/mapa-pokryti/

[13] Arduino. In: *www.arduino.cc* [online]. [cit. 16. 2. 2021]. Dostupné z: https://store-cdn.arduino.cc/uni/catalog/product/cache/1/image/500x375/f8876a31b63532bbba4e781c30024a0a/a/b/abx00019_extra_2.jpg

[14] Microchip. [online katalogový list]. SAM D21 family. ©2018 [cit. 6.4.2021]. Dostupné z: https://content.arduino.cc/assets/mkr-microchip_samd21_family_full_datasheet-ds40001882d.pdf

[15] Atmel Corporation [online katalogový list]. ATmega48PA/88PA/168PA/328P. ©2009 [cit. 6.4.2021]. Dostupné z: https://www.alldatasheet.com/datasheet-pdf/pdf/392244/ATMEL/ATMEGA328P.html

[16] Ing. Petr Olivka. Procesory CISC a RISC. [cit. 25. 4. 2021]. Dostupné z: https://poli.cs.vsb.cz/edu/arp/down/procrisc.pdf

[17] Arduino. In: *www.arduino.cc* [online]. [cit. 25. 4. 2021]. Dostupné z: https://content.arduino.cc/assets/Pinout-NANO_latest.pdf

[18] Atmel Corporation [online katalogový list]. ATmega809/1609/3209/4809. ©2019 [cit. 6.4.2021]. Dostupné z: https://content.arduino.cc/assets/Nano-Every_processor-48-pin-Data-Sheet-megaAVR-0-series-DS40002016B.pdf

[19] Arduino. In: *www.arduino.cc* [online]. [cit. 25. 4. 2021]. Dostupné z: https://content.arduino.cc/assets/Pinout-NANOevery_latest.pdf

[20] Raspberry Pi (Trading) Ltd. [online katalogový list]. Raspberry Pi Pico Datasheet. ©2020 [cit. 6.4.2021]. Dostupné z: https://datasheets.raspberrypi.org/pico/pico-datasheet.pdf

[21] KNOTEK, Vojtěch. Vliv teploty na parametry baterií. Praha, 209, Diplomová práce. České vysoké učení technické v Praze, Fakulta elektrotechnická, Katedra elektrotechnologie.

[22] EVE Energy CO. Ltd [online katalogový list]. ER26500. ©2018 [cit. 6.4.2021]. Dostupné z: https://www.tme.eu/Document/14f163cc8e8ed894a3ac7bbc3434f54a/ER26500.pdf

[23] EVE Energy CO. Ltd [online katalogový list]. ER34615. [cit. 6.4.2021]. Dostupné z: https://www.tme.eu/Document/20ac92d761024d233b7424b5d7dfc8c7/ER34615.pdf

[24] Arduino. In: *github.com* [online]. Arduino IDE. [cit. 6.4.2021]. Dostupné z: github.com/arduino/Arduino

[25] Němec, Jan. In: *www.linuxsoft.cz* [online]. Úvod do C/C++. [cit. 6.4.2021]. Dostupné z: https://web.archive.org/web/20070524034128/http://www.linuxsoft.cz/article.php?id_article=370

[26] PIETROWICZ, Daniel. Vliv nepřesnosti vzorkovací periody na frekvenční analýzu [online]. Brno, 2019 [cit. 2021-05-02]. Dostupné z: https://www.vutbr.cz/studenti/zav-prace/detail/117086. Bakalářská práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav automatizace a informatiky. Vedoucí práce Daniel Zuth.

9 SEZNAM ZKRATEK, SYMBOLŮ, OBRÁZKŮ A TABULEK

10 SEZNAM PŘÍLOH

Příloha A: CD

1. Text práce ve formátu PDF
2. Finální zdrojový kód z prostřední Arudino IDE
3. Finální schéma zapojení z prostředí Eagle