Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Zařízení pro bezdrátový sběr dat

Plzeň 2012 Lukáš Kopáček

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 21. června 2012 Lukáš Kopáček

Poděkování

Tímto bych chtěl poděkovat vedoucímu práce Ing. Jiřímu Ledvinovi CSc. z Katedry informatiky a výpočetní techniky FAV ZČU za jeho vřelý přístup, cenné rady a poskytnuté materiály. Dále bych rád poděkoval svým nejbližším za čas strávený nad korekturou práce.

Anotace

Tato bakalářská práce je zaměřena na návrh komunikačního protokolu pro lineární bezdrátovou senzorickou síť, dovolující přenášet data z koncových uzlů sítě do monitorovací stanice. Návrh je podřízen minimální spotřebě koncových uzlů. Navržený systém je realizován a aplikován na měření teploty.

Klíčová slova

Bezdrátové senzorické sítě, IEEE 802.15.4, komunikační protokol, CC2431.

Abstract

Devices for wireless data collection

This balcheors thesis focuses on the design of the communication protocol for wireless sensor network with linear topology. Network allows data transfer from the terminal nodes to the monitoring station. The proposal is subordinated to a minimum power consumption of end nodes. The proposed system is implemented and applied to the temperature measurement.

Keywords

LR-WPAN, low power listening, wireless sensor network, communication protocol, CC2431, IEEE 802.15.4.

Obsah

[1] Veškeré informace o architektuře jsem čerpal z manuálu k procesoru viz [1].
[2] Veškeré informace o protokolu IEEE 802.15.4 jsem čerpal z normy viz [3].

1 Úvod

S rostoucím zájmem o kvalitní snímání parametrů prostředí roste zájem o bezdrátové senzorické sítě. Hlavní výhody bezdrátových senzorických sítí jsou nízká spotřeba, dlouhá životnost, nízké pořizovací náklady a snadná instalace jednotlivých uzlů.

Cílem této práce je navrhnout a realizovat komunikační protokol pro bezdrátový sběr dat a ověřit, je-li možné vytvořit programové vybavení pro bezdrátovou senzorickou síť při použití volně šiřitelného překladače jazyka C.

Předpokladem pro návrh systému je lineární topologie. Hlavním kritériem pro návrh protokolu je minimální spotřeba uzlu.

2 Teoretická část

2.1 Architektura procesoru CC2431 [1]

Procesor CC2430 se vyrábí s flash pamětí o kapacitě 32, 64 nebo 128 KB. Tento procesor byl navržen pro implementaci protokolu IEEE 802.15.4 [3] a ZigBee® aplikace. Umožňuje vytvořit síť s velmi malými náklady na její uzly. CC2430 kombinuje výkon předešlého procesoru CC2420 s průmyslovým standardem procesorů 8051 (32/64/128 KB flash paměti, 8 KB RAM a mnoho dalších výhod). CC2430 je doporučován pro systémy, které vyžadují velmi nízkou spotřebu energie. Nízké spotřeby je docíleno různými operačními režimy a krátkými časy přepínání mezi nimi.

2.1.1 Procesor

Procesor CC2431 obsahuje rozšířené 8 bitové jádro průmyslového standardního jádra procesoru 8051. Toto rozšířené jádro používá standardní instrukční sadu procesoru 8051, ale instrukce jsou prováděny rychleji než ve standardním jádře 8051. Toho je docíleno jedním hodinovým cyklem na jeden instrukční cyklus na rozdíl od standardu, kde připadá 12 hodinových cyklů na jeden instrukční cyklus. Pokud je to možné, je instrukční cyklus srovnaný se čtením z paměti, takže jsou jednobajtové instrukce provedeny v jediném hodinovém cyklu. Navíc je rozšířené jádro 8051 vybaveno druhým registrem pro nepřímý přístup do datové paměti a rozšířenou jednotkou pro zpracování až 18 přerušení. Jádro je zpětně kompatibilní s průmyslovým standardem 8051, ale u programů používajících časové zpoždění pomocí smyček je nutné tyto smyčky upravit. Také programy používající periferní jednotky nemusejí pracovat správně, kvůli změně ovládání periferií oproti standardu 8051.

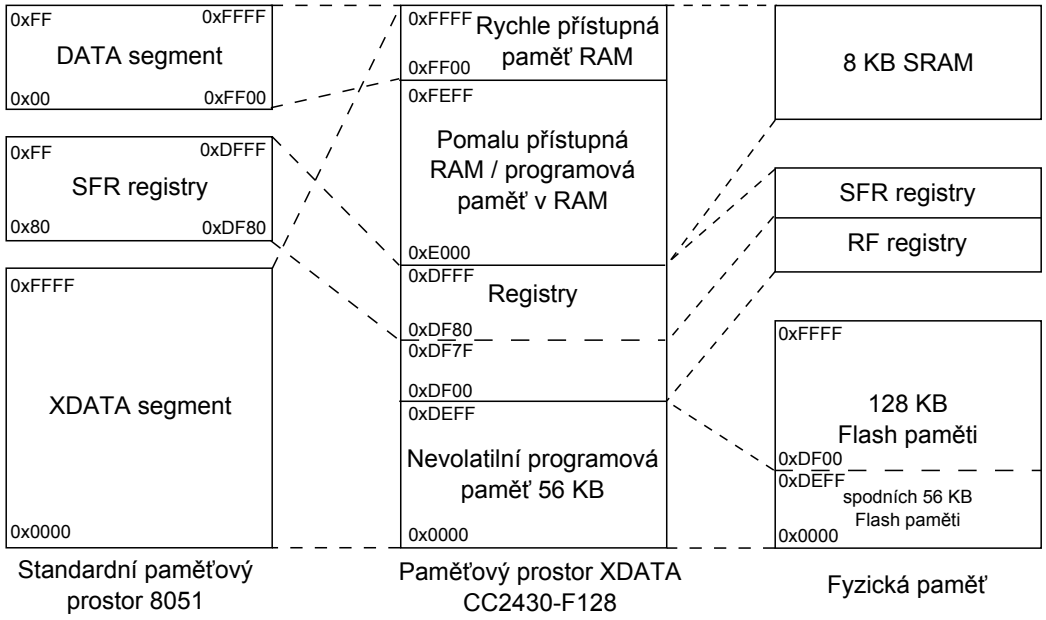
2.1.2 Paměť

Procerosr 8051 má oddělené paměťové prostory pro program a pro data. Architektura procesoru 8051 má 4 odlišné paměťové prostory.

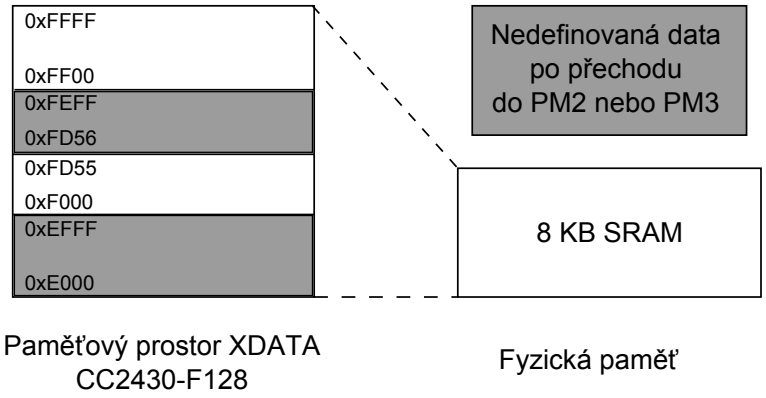
XDATA

Paměťový prostor pro čtení i zápis. Velikost tohoto paměťového prostoru je 64 KB. Na rozdíl od přístupu k segmentu DATA má XDATA společnou

[1] Veškeré informace o architektuře jsem čerpal z manuálu k procesoru viz [1].



Obrázek 2.1: Paměťový prostor XDATA



Obrázek 2.2: Uchovávaný paměťový prostor XDATA

sběrnici s jádrem procesoru a paměťovým prostorem CODE, tudíž nemůže souběžně probíhat prefetch instrukcí ze segmentu CODE.

Mapa paměťového prostoru XDATA je na obrázku 2.1. Pro zařízení s flash pamětí větší než 32 KB je do paměťového prostoru XDATA namapováno pouze spodních 56 KB flash paměti od adresy `0x0000` do adresy `0xDEFF`. Ve verzi s 32 KB flash paměti je mapováno pouze 32 KB a to od adresy `0x0000` do adresy `0x7FFF`. Přístup k neimplementované oblasti paměti (od adresy `0x7FFF` do adresy `0xDEFF`) vrací nedefinované výsledky.

Pro všechna provedení procesoru CC2430 je 8 KB SRAM umístěno v adresním rozsahu `0xE000-0xFFFF`. SFR registry jsou mapovány na adresy `0xDF80-0xDFFF` a jsou také stejné pro všechny verze procesoru CC2430. Do zbylé části v rozsahu adres `0xDF00-0xDF7F` paměťového prostoru jsou mapovány RF registry pro řízení vysílacího modulu.

Mapování flash paměti, SRAM a registrů do prostoru XDATA umožňuje DMA řadiči a procesoru přístup do všech fyzických pamětí z jednoho unifikovaného adresního prostoru. Změna banky, popsaná v části pro kódovou paměť, nemá vliv na obsah 24 KB v prostoru nad 32 KB namapované flash paměti. Jedním z důsledků tohoto mapování je i fakt, že první adresa použitelné SRAM začíná na adrese `0xE000`, toto musí vzít překladače v úvahu.

V úsporných režimech PM2 a PM3 jsou uchovány horní 4 KB SRAM, t. j. paměť v rozsahu `0xF000-0xFFFF` mimo adresy `0xFD56-0xFEFF`, jak je zobrazeno na obrázku 2.2. Obsah si uchovávají i registry procesoru, periferní registry a RF registry. Přepínání mezi režimy spotřeby se jeví vzhledem k softwaru jako transparentní. Výjimku tvoří watchdog timer, který se po vstupu do PM2 nebo PM3 vynuluje a registry TXFIFO/RXFIFO, jejichž obsah také není uchován při vstupu do PM2 nebo PM3.

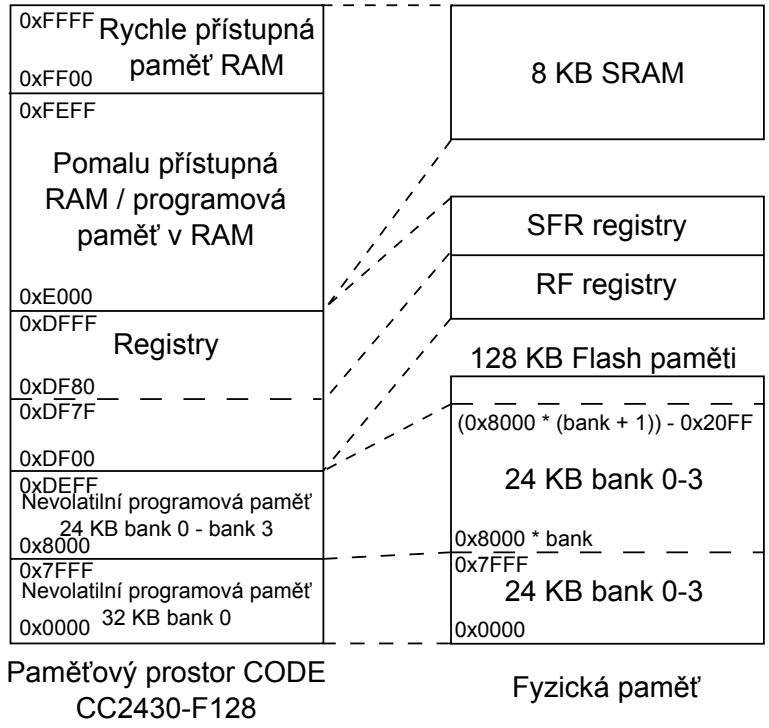
Procesor CC2430 poskytuje registr `MPAGE`, který je používán během instrukcí `MOVX A,@Ri` a `MOVX @Ri,A`. V registru `MPAGE` je uloženo 8 nejvýznamnějších bitů adresy, přičemž registr `Ri` obsahuje 8 nejnižších bitů adresy.

CODE

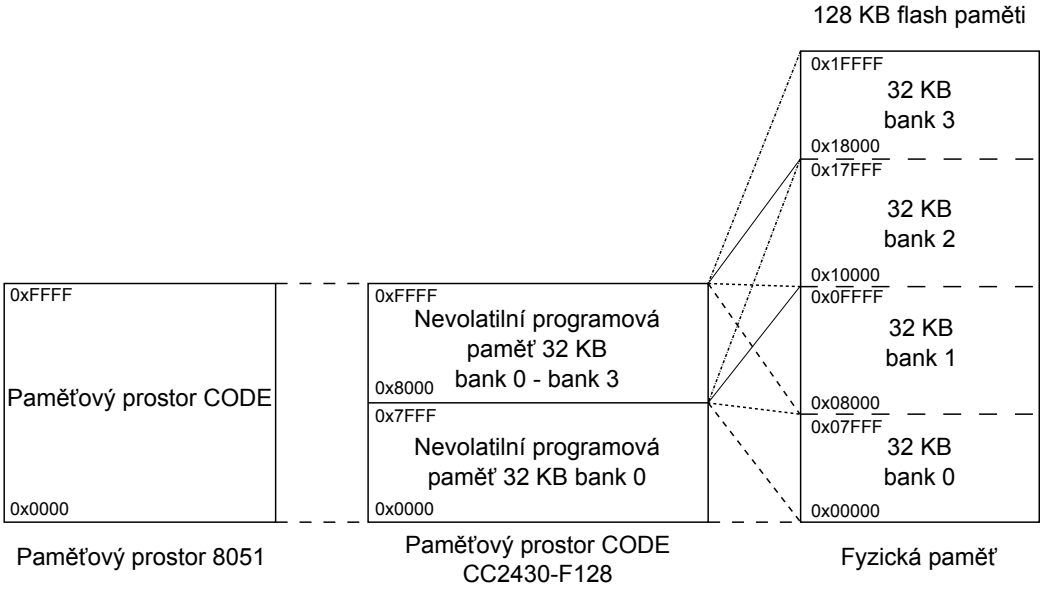
Prostor programové paměti používá buď unifikované (viz obrázek 2.3) nebo neunifikované (viz obrázek 2.4) mapování paměti na fyzickou paměť.

Unifikované mapování je stejné jako mapování XDATA. V tomto případě je pro flash paměti nad 32 KB mapováno do paměťového prostoru CODE jen 56 KB. Obsah horních 24 KB řídí zvolený bank (viz obrázek 2.3). Při použití unifikovaného mapování paměťového prostoru CODE neobsahují adresy větší než `0xDEFF` data z flash paměti.

Pro zařízení s rozsahem 128 KB flash paměti se používá banking pro paměťový prostor CODE. Horních 32 KB paměťového prostoru CODE je



Obrázek 2.3: Paměťový prostor CODE s unifikovaným mapováním



Obrázek 2.4: Paměťový prostor CODE s neunifikovaným mapováním

mapováno na jeden ze 4 bloků flash paměti (bank), dolních 32 KB paměťového prostoru CODE je nastálo mapováno do dolních 32 KB flash paměti. Nastavení bloku flash paměti (bank) je řízeno nastavením bitů `MAP` v registru `FMAP`. Rozdělení flash paměti na bloky a jejich mapování je na obrázku 2.4.

Při použití unifikovaného mapování paměťového prostoru CODE zároveň s volbou bloku dat (banking) je k dispozici jen 24 KB z vybraného bloku (bank) viz obrázek 2.3.

DATA

Paměťový prostor DATA je přímo nebo nepřímo adresovatelný paměťový prostor pro čtení i zápis přístupný v jednom instrukčním cyklu.

8 bitový adresní rozsah paměťového prostoru DATA je mapován do horních 256 bajtů v SRAM. Tato oblast je také dostupná přes unifikovaně mapovaný paměťový prostor CODE a přes paměťový prostor XDATA na adresním rozsahu `0xFF00-0xFFFF`.

Spodních 128 B může být adresováno přímo i nepřímo, horních 128 B může být adresováno jen nepřímo.

SFR

Paměťový prostor SFR je paměťový prostor pro čtení i zápis přístupný v jednom instrukčním cyklu. Tento paměťový prostor má rozsah 128 B.

Do adresního prostoru SFR je mapováno 128 registrů propojených s hardwarem. Tyto registry jsou také přístupné přes adresní prostor XDATA na rozsahu adres `0xDF80-0xDFFF`. Některé registry specifické pro procesor jsou umístěny v jádře procesoru a mohou tak být použity pouze přes paměťový prostor SFR.

Mnoho SFR registrů je shodných se standardem 8051, ale procesor CC2430 obsahuje navíc SFR registry, které ovládají periferie, které nejsou ve standardu 8051. Tyto registry jsou používány pro komunikaci s periferní jednotkou a radiovým modulem.

RFR

Tyto registry se všechny vztahují k řízení a konfiguraci radiového modulu a jsou přístupné pouze přes paměťový prostor XDATA na adresách `0xDF00-0xDF7F` viz obrázek 2.1.

2.1.3 Přerušovací systém

Procesor CC2431 má 18 zdrojů přerušení. Každý zdroj má svou vlastní značku (request flag) umístěnou v prostoru SFR registrů. Každé přerušení je vyvolané odpovídající značkou (flag) a může být samostatně zakázáno nebo povoleno. Přerušení jsou seskupovány a jednotlivým skupinám lze přidělit prioritu přerušení.

2.1.4 Periferie

Časovače

Časovač 1 Nezávislý 16 bitový časovač, který podporuje typické funkce čítače/časovače, jako jsou vstupní zachycení, výstupní porovnání a funkce PWM. Časovač má 3 nezávislé zachycovací/porovnávací kanály a používá jeden pin ze vstupních/výstupních portů na kanál.

Vlastnosti čítače/časovače 1:

- Tři zachycovací/porovnávací (capture/compare) kanály.
- Zachycení na náběžnou, doběžnou nebo na obě hrany vstupního signálu.
- Nastavení, vynulování nebo přepnutí porovnávacího výstupu.
- Volně běžící (free-running), modulo nebo čítání nahoru/dolů (up/down) módy čítače.
- Předdělička kmitočtu hodinového signálu (možnosti dělit 1, 8, 32, 128).
- Přerušení je generováno při každé události (zachycení/porovnání a koncová hodnota čítače).
- Přepínač DMA přenosu.

Přehled registrů:

- T1CNTH obsahuje horní bajt šestnáctibitové hodnoty čítače.
- T1CNTL obsahuje dolní bajt šestnáctibitové hodnoty čítače.
- T1CTL řídící a stavový registr čítače/časovače 1.
- T1CCTLx řídící registr čítače/časovače 1 kanálu "x"(0,1,2).
- T1CCxH horní bajt zachytávané/porovnávané hodnoty na kanálu "x".
- T1CCxL dolní bajt zachytávané/porovnávané hodnoty na kanálu "x".

Časovač 2 Časovač 2 poskytuje kromě standardních možností navíc možnost startu na náběžnou hranu hodin sleep timeru, možnost automatického přepočtu hodnoty po vzbuzení a je provázán s radiovým koprocesorem.

Časovač 3,4 Časovače 3 a 4 poskytují podobné možnosti jako časovač 1 s několika omezeními.

Vstupní/výstupní porty

Mikroprocesor CC2430 má 21 digitálních vstupních/výstupních vývodů, které lze nakonfigurovat jako hlavní vstupní/výstupní vývody nebo jako periferní vývody připojené k ADC převodníku, čítačům/časovačům nebo k sériové lince. Použití těchto vývodů je plně konfigurovatelné přes konfigurační registry. Možnosti připojení periferií na dané porty jsou zobrazeny v tabulce 40 na straně 80 manuálu (datasheet) k procesoru CC2430 [1]. Podle této tabulky a našich požadavků se nastavují příslušné registry PERCFG, ADCCFG, PxSEL, PxDIR, PxINP, PxIFG, PICTL a P1IEN, kde "x"vyjadřuje možnosti 0,1,2. Porty P0, P1 a P2 jsou bitově adresovatelné.

Přehled registrů:

- PERCFG nastavuje umístění vstupů/výstupů daných periferií na výstupních portech.
- ADCCFG nastavuje vstupy ACD převodníku.
- PxSEL nastavuje funkci vývodů (periferní nebo vstupní/výstupní).
- PxDIR nastavuje jestli daný vývod bude vstupní nebo výstupní.
- PxINP nastavuje jestli dané vývody budou dvoustavové nebo třístavové (P2 nemá možnost třístavového výstupu).
- PxIFG obsahuje příznaky přerušení (interrupt flags).
- PICTL nastavuje konfiguraci přerušení od vstupních/výstupních portů.
- P1IEN obsahuje masku přerušení pro port P1.

Seriové rozhraní

Mikroprocesor CC2430 má dvě sériové linky (USART0 a USART1), které umožňují klasický synchronní (SPI) a asynchronní přenos (UART). Obě sériové linky mají stejnou funkci a jsou vyvedeny na samostatné vývody.

Přehled registrů:

- UxCSR ovládací a stavový registr sériové linky "x"(0,1).
- UxUCR řídící registr pro asynchronní přenos UART "x"(0,1).
- UxGCR řídící registr pro SPI a pro nastavení rychlosti přenosu.
- UxDBUF vysílací/přijímací buffer.
- UxBAUD registr pro nastavení rychlosti přenosu.

Analogově digitální převodník

Analogově digitální převodník (ADC) podporuje až dvanáctibitovou analogově-digitální konverzi. ADC obsahuje analogový mutiplexer až s osmi samostatně konfigurovatelnými kanály, generátorem referenčního napětí a možností zápisu výsledků konverze přes DMA do paměti. Vstupy lze také nakonfigurovat jako rozdílové (diferenciální).

Vlastnosti analogově digitálního převodníku:

- Volitelný dělící poměr, který také určuje rozlišení (7 až 12 bitů).
- Osm samostatných vstupních kanálů (diferenciální).
- Referenční napětí volitelné jako interní, externí ukončené na jednom výstupu, externí diferenciální nebo AVDD_SOC[8].
- Umožňuje generování přerušení.
- DMA přepínače při ukončení převodu.
- Vstup teplotního čidla na kanálu 14.
- Možnost měření baterie na kanálu 15.

Přehled registrů:

- ADCL obsahuje dolní bajt výsledku konverze.
- ADCH obsahuje horní bajt výsledku konverze.
- ADCCONx řídící registry ADC převodníku.

[8] AVDD_SOC je vývod 20 na pouzdře procesoru.

2.1.5 Komunikační procesor

Jádro vysílače je založené na vysílacím procesoru CC2420. Detekce začátku rámce (SFD) vyvolává přerušení a přijatá data jsou uchována v registru `RXFIFO`, který má omezenou kapacitu na 128 B. Je z něj možné číst přes registr `RFD` umístěný v paměťovém prostoru SFR. Při čtení z registru `RFD` se čte přijímací buffer (`RXFIFO`), při zápisu do registru `RFD` se zapisuje do vysílacího bufferu (`TXFIFO`).

Komunikační procesor obsahuje hardwarovou podporu pro kontrolu CRC zabezpečovacího polynomu. RSSI a korelační hodnoty jsou umístěny na konci rámce. Digitální část komunikačního procesoru obsahuje podporu pro manipulaci s rámcem, rozpoznání adresy, ukládání dat do vyrovnávací paměti, jednotku pro CSMA-CA a jednotku pro zabezpečení na přístupové vrstvě.

Pro řízení komunikačního procesoru se používá speciální instrukční sada. Tato instrukční sada obsahuje jednobajtové instrukce, z nichž každá ovládá nějakou fukci komunikačního procesoru. Tyto instrukce je třeba použít pro zapnutí frekvenčního syntetizátoru, příjmu, vysílání a ostatních funkcí. Operační kód těchto instrukcí se zapisuje do registru `RFST` v oblasti SFR.

RF registry jsou mapovány do paměťového prostoru XDATA na adresách `0xDF00-0xDF7F` a slouží k nastavení komunikačního procesoru a k zjišťování stavových informací o něm.

Komunikační procesor generuje dva druhy přerušení. Při přetečení registru `TXFIFO` nebo podtečení registru `RXFIFO` generuje přerušení `RFERR` (přerušení 0). Přerušení `RF` (přerušení 12) generuje podle nastavení registru `RFIM` podle zvoleného příznaku přerušení (interrupt flag).

2.1.6 Registry

Procesor CC2430 má speciální funkční registry (SFR) a registry pro ovládání radiového modulu (RFR). Speciální funkční registry leží v rozsahu adres `0xDF80-0xDFFF` a jsou přístupné přímo nebo přes paměťový prostor XDATA. Registry pro ovládání radiového modulu na adrese `0xDF00-0xDF7F` jsou přístupné jen z paměťového prostoru XDATA.

2.2 Prostředky pro programování procesoru CC2431 v jazyce C

Na trhu existuje mnoho překladačů, které jsou schopny vygenerovat ze zdrojového kódu v jazyce C příslušný soubor pro naprogramování procesoru. Tyto

překladače jsou často komerční, jejich volně šiřitelná verze obsahuje zpravidla omezení vygenerovaného kódu a verze bez omezení velikosti jsou velice drahé.

Proto jsem byl požádán, abych z experimentálních důvodů zvolil pro překlad zdrojových souborů volně šiřitelný překladač SDCC a ověřil tak, jde-li napsat programové vybavení pro procesor CC2430 bez použití komerčního softwaru. Tento překladač je schopen vygenerovat potřebný soubor pro naprogramování procesoru a pro jeho používání není potřeba žádné licence, licenčního klíče nebo registrace. Navíc existuje k tomuto překladači také plug-in do vývojového prostředí Eclipse. Překladač podporuje volbu bloku kódové paměti (banking). Značnou nevýhodou je, že vývojové prostředí komerčního překladače má hardwarovou podporu v podobě rozhraní pro ladění programu přímo v zařízení.

2.3 Protokol IEEE 802.15.4 [10]

2.3.1 Fyzická vrstva

Fyzická vrstva je zodpovědná za tyto funkce:

- aktivace a deaktivace radiového vysílače;
- detekce vysílání na aktuálním kanálu;
- LQI pro přijaté pakety;
- detekce prázdného kanálu (CCA) pro CSMA-CA;
- nastavení frekvence kanálu;
- příjem a vysílání dat.

Frekvenční rozsahy a rychlosti přenosu

Specifikace IEEE 802.15.4-2003 [3] definuje na fyzické úrovni tři frekvenční pásma viz tabulka 2.1.

[10] Veškeré informace o protokolu IEEE 802.15.4 jsem čerpal z normy viz [3].

Fyzická úroveň [MHz]	Frekvenční rozsah [MHz]	Počet kanálů	Modulace	Rychlost přenosu [kb/s]
868/915	868-868,6	1	BPSK	20
	902-928	10	BPSK	40
2450	2400-2483,5	16	O-QPSK	250

Tabulka 2.1: Frekvenční rozsahy a přenosové rychlosti

Přiřazení kanálů a jejich číslování

Celkem 27 kanálů číslovaných od 0 do 26 je k dispozici ve třech frekvenčních pásmech. 16 kanálů je k dispozici v pásmu 2450 MHz, 10 v pásmu 915 MHz a 1 v pásmu 868 MHz. Střední frekvence F_c těchto kanálů je definována následovně:

$$F_c = 868,3[MHz], k = 0 \tag{2.1}$$

$$F_c = 906 + 2(k-1)[MHz], k = 1, 2, ..., 10 \tag{2.2}$$

$$F_c = 2405 + 5(k-11)[MHz], k = 11, 12, ..., 26 \tag{2.3}$$

kde k je číslo kanálu.

Každá fyzická vrstva podporuje všechny kanály povolené předpisy pro oblast, kde zařízení pracuje.

Obecný formát paketu na fyzické vrstvě

Oktety: 4	1	1		proměnná délka
Preambule	Značka začátku rámce (SFD)	Délka rámce (7 bitů)	rezerva (1 bit)	Fyzická servisní datová jednotka (PSDU)
Synchronizační hlavička (SHR)		Hlavička fyzické úrovně		Obsah rámce fyzické úrovně

Obrázek 2.5: Formát paketu na fyzické vrstvě

Z formátu paketu fyzické vrstvy plyne maximální délka paketu a to 127 oktetů.

2.3.2 Přístupová vrstva

Přístupová vrstva (MAC) zpracovává přístup k fyzické vrstvě a je zodpovědná za tyto úlohy:

- generování síťových signálů (beacon), pokud je zařízení koordinátor;
- synchronizaci na síťové signály (beacon);
- podporu připojení k PAN a odpojení od PAN;
- podporu zabezpečení zařízení;
- využívání CSMA-CA mechanismu pro přístup ke komunikačnímu kanálu;
- manipulaci a udržování GTS mechanismu;
- zprostředkování spolehlivé linky mezi dvěma MAC úrovněmi.

Typy rámců na přístupové vrstvě

Každý rámec na přístupové vrstvě musí obsahovat hlavičku přístupové vrstvy (MHR), obsah (payload) a konec rámce na přístupové vrstvě (MFR).

Hlavička přístupové vrstvy obsahuje údaje pro řízení rámce, sekvenční číslo a adresní informace. Obsah přenášený rámcem má proměnnou délku a obsahuje specifické informace pro daný typ rámce. Potvrzovací rámce nepřenášejí žádný obsah. Konec rámce na přístupové vrstvě obsahuje sekvenci pro zjištění chyb při přenosu (FCS).

Následující rámce přístupové vrstvy jsou popsány jako sled polí v určitém pořadí. Všechny formáty rámců přístupové vrstvy jsou zobrazeny v pořadí, v jakém jsou předávány fyzické vrstvě zleva doprava, kde vlevo je bit přenášený v čase jako první. Bity v každé části jsou číslované od 0 (nejméně významné) až do $k - 1$ (nejvýznamější), kde k je počet bitů. Pole, která jsou delší než jeden oktet jsou odeslány fyzické vrstvě od nejnižšího oktetu po nejvyšší.

Základní formát rámce Formát rámce přístupové vrstvy je složen z hlavičky (MHR), obsahu (payload) a konce (MFR). Pole hlavičky má fixní pořadí, avšak adresní pole nemusí být v každém rámci. Základní formát rámce přístupové vrstvy je na obrázku 2.6.

Formáty jednotlivých rámců Na obrázcích 2.7 až 2.10 jsou zobrazeny formáty jednotlivých rámců.

<table>
<tr><td>Oktety: 2</td><td>1</td><td>0/2</td><td>0/2/8</td><td>0/2</td><td>0/2/8</td><td>proměnné</td><td>2</td></tr>
<tr><td rowspan="2">Řízení rámce (frame control)</td><td rowspan="2">Sekvenční číslo</td><td>Identifikátor cílové sítě (destination PAN identifier)</td><td>Cílová adresa</td><td>Identifikátor zdrojové sítě (source PAN identifier)</td><td>Zdrojová adresa</td><td rowspan="2">Obsah rámce</td><td rowspan="2">Kontrolní sekvence rámce (FCS)</td></tr>
<tr><td colspan="4">Adresní pole</td></tr>
<tr><td colspan="6">Hlavička přístupové vrstvy (MHR)</td><td>Obsah rámce přístupové vrstvy</td><td>Konec rámce přístupové vrstvy (MFR)</td></tr>
</table>

Obrázek 2.6: Základní formát rámce přístupové vrstvy

<table>
<tr><td>Oktety: 2</td><td>1</td><td>4/10</td><td>0/2/8</td><td>0/2</td><td>0/2/8</td><td>proměnné</td><td>2</td></tr>
<tr><td>Řízení rámce (frame control)</td><td>Sekvenční číslo</td><td>Adresní pole</td><td>Specifikace superframe</td><td>Pole GTS</td><td>Pole čekání na adresu (pending address fields)</td><td>Obsah beacon rámce</td><td>Kontrolní sekvence rámce (FCS)</td></tr>
<tr><td colspan="3">Hlavička přístupové vrstvy (MHR)</td><td colspan="4">Obsah rámce přístupové vrstvy</td><td>Konec rámce přístupové vrstvy (MFR)</td></tr>
</table>

Obrázek 2.7: Formát beacon rámce přístupové vrstvy

<table>
<tr><td>Oktety: 2</td><td>1</td><td>4/10</td><td>proměnné</td><td>2</td></tr>
<tr><td>Řízení rámce (frame control)</td><td>Sekvenční číslo</td><td>Adresní pole</td><td>Obsah datového rámce</td><td>Kontrolní sekvence rámce (FCS)</td></tr>
<tr><td colspan="3">Hlavička přístupové vrstvy (MHR)</td><td>Obsah rámce přístupové vrstvy</td><td>Konec rámce přístupové vrstvy (MFR)</td></tr>
</table>

Obrázek 2.8: Formát datového rámce přístupové vrstvy

Oktety: 2	1	2
Řízení rámce (frame control)	Sekvenční číslo	Kontrolní sekvence rámce (FCS)
Hlavička přístupové vrstvy (MHR)		Konec rámce přístupové vrstvy (MFR)

Obrázek 2.9: Formát potvrzovacího rámce přístupové vrstvy

Oktety: 2	1	4/10	1	proměnné	2
Řízení rámce (frame control)	Sekvenční číslo	Adresní pole	Identifikáto řídícího rámce	Obsah řídícího rámce	Kontrolní sekvence rámce (FCS)
Hlavička přístupové vrstvy (MHR)			Obsah rámce přístupové vrstvy		Konec rámce přístupové vrstvy (MFR)

Obrázek 2.10: Formát řídícího rámce přístupové vrstvy

Způsoby adresování na přístupové vrstvě

Na přístupové vrstvě je možné adresovat jak uvnitř sítě (intra PAN) tak mezi sítěmi. Adresa sítě obsahuje dva bajty. Pro adresaci všech sítí slouží tzv. všeobecná adresa sítě (broadcast) ve tvaru `0xFFFF`.

Stejným způsobem jsou adresována i jednotlivá zařízení ve stejné síti. Adresa zařízení může být dvoubajtová nebo osmibajtová.

2.4 Implementace protokolu IEEE 802.15.4 na procesoru CC2431

Procesor CC2431 má hardwarovou podporu rozpoznání adresy, možnost automatického potvrzování přijatého rámce, hardwarovou podporu šifrování/dešifrování AES, CSMA-CA strobovací procesor, možnost volby vysílacího kanálu, intenzity vysílání a hardwarovou podporu pro měření RSSI/LQI.

2.5 Přehled komunikačních protokolů pro bezdrátovou senzorickou síť

Obecně lze komunikační protokoly přístupové úrovně rozdělit na tři skupiny viz [2].

První skupinou jsou plánované protokoly označované jako TDMA (Time Division Multiple Access). U těchto protokolů má každý uzel přidělený časový slot pro vysílání a příjem. Uzel se vzbudí jen v okamžiku, kdy má komunikovat. Uzel sítě musí znát časové sloty sousedních uzlů, aby s nimi mohl komunikovat. Tento typ protokolů poskytuje dobré výsledky, ale vyžaduje dobrou synchronizaci mezi uzly. Navíc uzly spotřebují určitou část energie na vytvoření a udržování časové synchronizace.

Druhou skupinou jsou protokoly se společnou aktivní periodou. Tyto protokoly organizují uzly sítě tak, aby se přepnuly do režimu nízké spotřeby a zpět ve stejný čas. Pro odeslání dat musí uzel čekat na další periodu pro vzbuzení. Tato skupina protokolů také vyžaduje synchronizaci. Tato synchronizace však nemusí být tak přesná jako u plánovaných TDMA protokolů. Uzly používající protokol se společnou aktivní periodou také spotřebovávají energii na vytvoření a udržování okamžiků pro přechod do režimu nízké spotřeby a zpět.

Třetí skupina jsou asynchronní protokoly přístupové úrovně. V tomto případě není potřeba žádná synchronizace, protože každý uzel přechází do

režimu nízké spotřeby a zpět nezávisle na ostatních uzlech sítě. Tyto protokoly implementují rozdílné mechanismy ke zjištění, je-li příjemce aktivní. Tyto protokoly mohou být rozděleny na protokoly vzorkující preambuli a na protokoly, u kterých přenos dat vyvolá příjemce.

3 Návrh systému

Cílem této práce je vyvinout uzel pro dlouhodobé snímání parametrů prostředí (teplota, tlak, vlhkost, ...). Skupina takových uzlů musí být schopna vytvořit lineární síť a doručovat naměřená data do řídící stanice. Předpoklad pro návrh systému je lineární umístění uzlů, ze kterého návrh vychází. Hlavním kritériem pro návrh tohoto systému je minimální spotřeba uzlu.

3.1 Fyzická úroveň

Jako protokol fyzické úrovně je použit protokol IEEE 802.15.4-2003 [3], určený pro LR-WPAN.

3.2 Přístupová úroveň

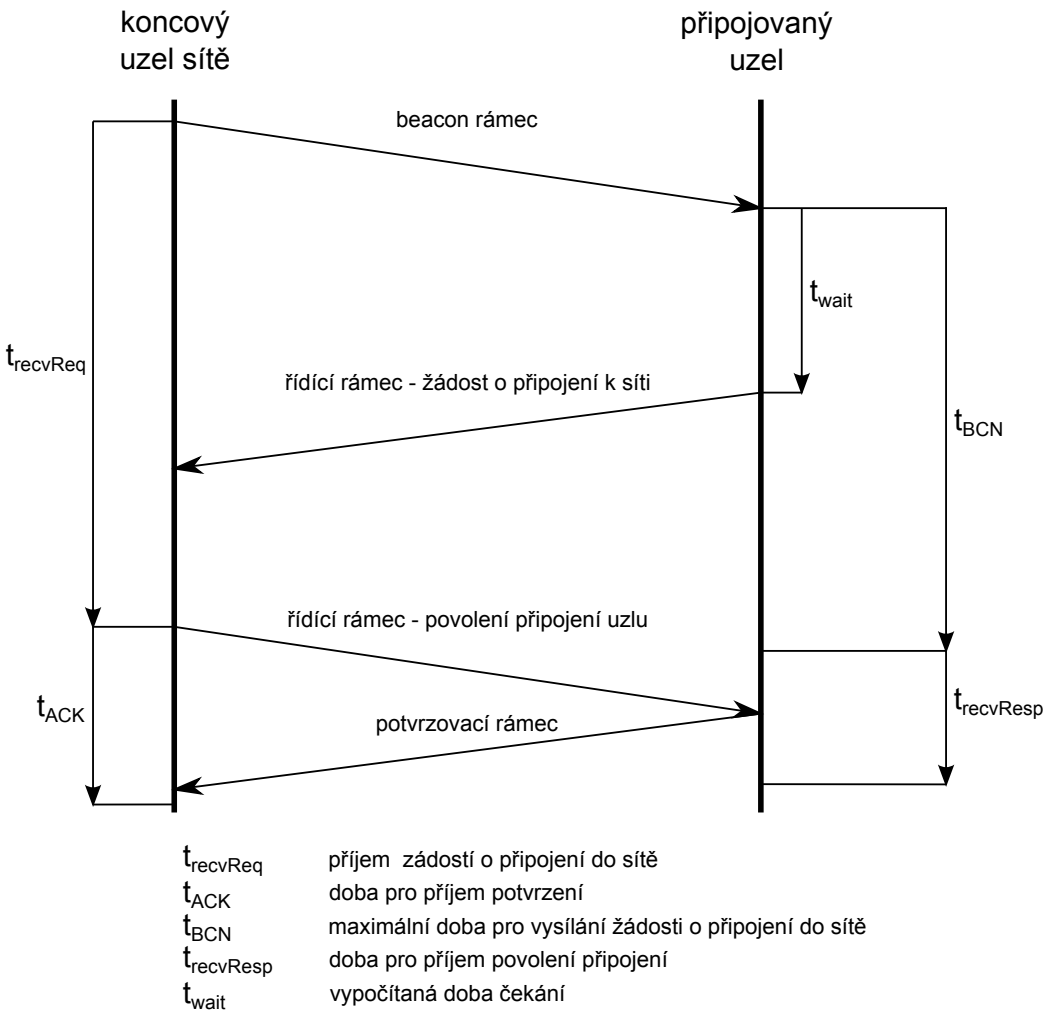
Na přístupové úrovni je potřeba navrhnout komunikační protokol a definovat hlavičky přenášených rámců. Vzhledem k minimalizaci spotřeby a předpokladu lineární sítě jsem se rozhodl pro návrh protokolu typu TDMA (viz 2.5).

3.2.1 Návrh komunikačního protokolu

Návrh si rozdělím na tři části: přenos dat a řídících rámců, připojení uzlu do sítě a synchronizaci času. Každý uzel sítě bude mít přidělen svůj časový slot (okamžik), kdy proběhne přenos dat směrem ke kořenu, přenos řídících rámců směrem od kořenu a připojení nových uzlů do sítě. Během tohoto časového slotu také proběhne synchronizace uzlu.

Připojení uzlu do sítě

Pro synchronizaci uzlu, který se chce připojit do sítě slouží beacon rámce. Koncový uzel v síti (nejdále od kořenu) vyšle beacon rámec a čeká na odpověď po definovanou dobu. Při návrhu jsem dobu čekání na odpověď $t_{recvReq}$ počítal dle vztahu 3.1, kde t_{CMDreq} je doba vysílání žádosti o připojení do sítě. Po dobu $t_{recvReq}$ mohou okolní uzly posílat uzlu v síti řídící rámce s požadavkem na připojení do sítě. Uzel v síti si pamatuje jen poslední požadavek, na který po uplynutí doby $t_{recvReq}$ odpoví kladně řídícím rámcem a následně čeká na potvrzení odeslaného řídícího rámce. Pokud potvrzení přijde, je v síti



Obrázek 3.1: Připojení uzlu do sítě

nový uzel, který se stává koncovým (je nejdále od kořene). Pokud potvrzení nepřijde, nebudou pakety od tohoto uzlu přijaty.

$$t_{recvReq} = (255 + (max.\ počet\ pokusů)) \cdot t_{CMDreq} \tag{3.1}$$

Z pohledu uzlu, který se chce připojit do sítě je situace následující. Uzel přijímá rámce až do doby, kdy přijme beacon rámec. Tento rámec v sobě nese i změřenou kvalitu signálu (RSSI), podle které uzel odvodí kvalitu linky (LQI) jako číslo v rozsahu jednoho bajtu $(0 - 255)$. Toto číslo použije pro nastavení nižšího bajtu své aktuální adresy ($\texttt{0x0000} + LQI$). Vyšší bajt své aktuální adresy nastaví na počet neúspěšných pokusů o připojení. K číslu LQI přičte počet neúspěšných pokusů o připojení do sítě a počká dobu t_{wait} definovanou podle vztahu 3.2.

$$t_{wait} = (LQI + (počet\ pokusů)) \cdot t_{pkt} \tag{3.2}$$

Po uplynutí této doby vyšle řídící rámec s požadavkem na připojení do sítě. Dále čeká na uplynutí doby pro zasílání požadavků t_{BCN}, poté začne přijímat řídící rámce pro přijetí do sítě. Pokud do uplynutí doby $t_{recvResp}$ řídící rámec nepřijde, uzel není přijat do sítě. Pokud přijde řídící rámec, odešle uzel potvrzovací rámec. Při ztrátě potvrzovacího rámce nepřijme uzel v síti žádný datový rámec a neodpoví na něj potvrzením.

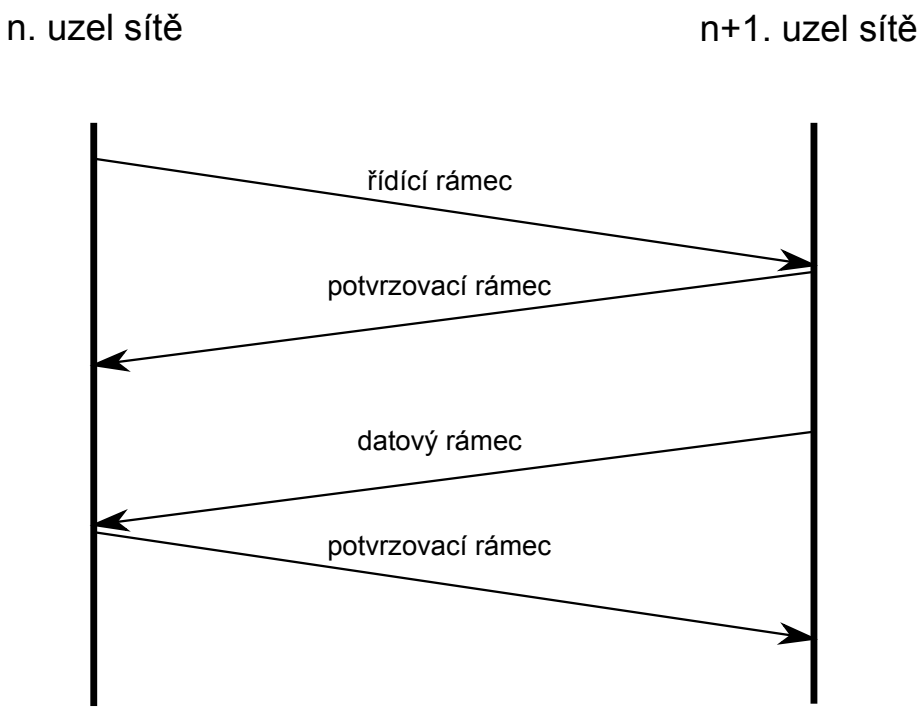
Přenos dat a řídících rámců

Uzel blíže ke kořeni pošle vzdálenějšímu uzlu řídící rámec. Vzdálenější uzel musí řídící rámec potvrdit potvrzovacím rámcem a za tímto potvrzovacím rámcem může vyslat datový rámec směrem ke kořeni. Datový rámec musí uzel blíže ke kořeni potvrdit potvrzovacím rámcem viz obrázek 3.2. Pokud nepřijde potvrzení, počká odesílatel na další časový slot, ve kterém se pokusí přenést data znovu.

Synchronizace času

Jelikož i krystalové oscilátory, podle nichž určuje uzel sítě okamžik vzbuzení, mají určitou toleranci, je potřeba při vzájemné komunikaci dvou a více uzlů tyto oscilátory naladit tak, aby bylo možné oba uzly vzbudit ve stejný okamžik. Tím se eliminuje zbytečné naslouchání na minimum a minimalizuje se tak spotřeba energie. Pro synchronizaci uzlů v síti je v této práci navržen následující algoritmus.

Všechny uzly sítě mají *tabulku událostí* a *časovou značku*. Tabulka událostí má tři sloupce a čtyři řádky. V každém sloupci je údaj o aktuálním



Obrázek 3.2: Přenos dat a řídících rámců v síti

reálném čase, údaj o hodnotě časovače (sleep timeru), údaj o maximální délce události v reálném čase a údaj o maximální délce události v počtu tiků časovače. Jednotlivé sloupce představují časové sloty (časová okénka). V tomto případě budou časové sloty tři: slot pro měření, slot pro odeslání dat a slot pro přijmutí dat. Časová značka obsahuje údaj o reálném čase a k němu příslušnou hodnotu časovače (sleep timeru). Vždy při přechodu do režimu nízké spotřeby se do časovače (sleep timeru) načte hodnota časového okamžiku. Tato hodnota se uloží do časové značky i s příslušným reálným časem. Přepočítá se původní hodnota v tabulce a zařízení se uvede do režimu nízké spotřeby.

Synchronizace probíhá vždy při předání řídícího rámce. Pro realizaci synchronizace je potřeba zavést *statickou korekci* a *dynamickou korekci*.

Statická korekce posune všechny hodnoty časovače (sleep timeru) v tabulce událostí o stejnou hodnotu. Tento typ korekce proběhne jen jednou po připojení uzlu do sítě.

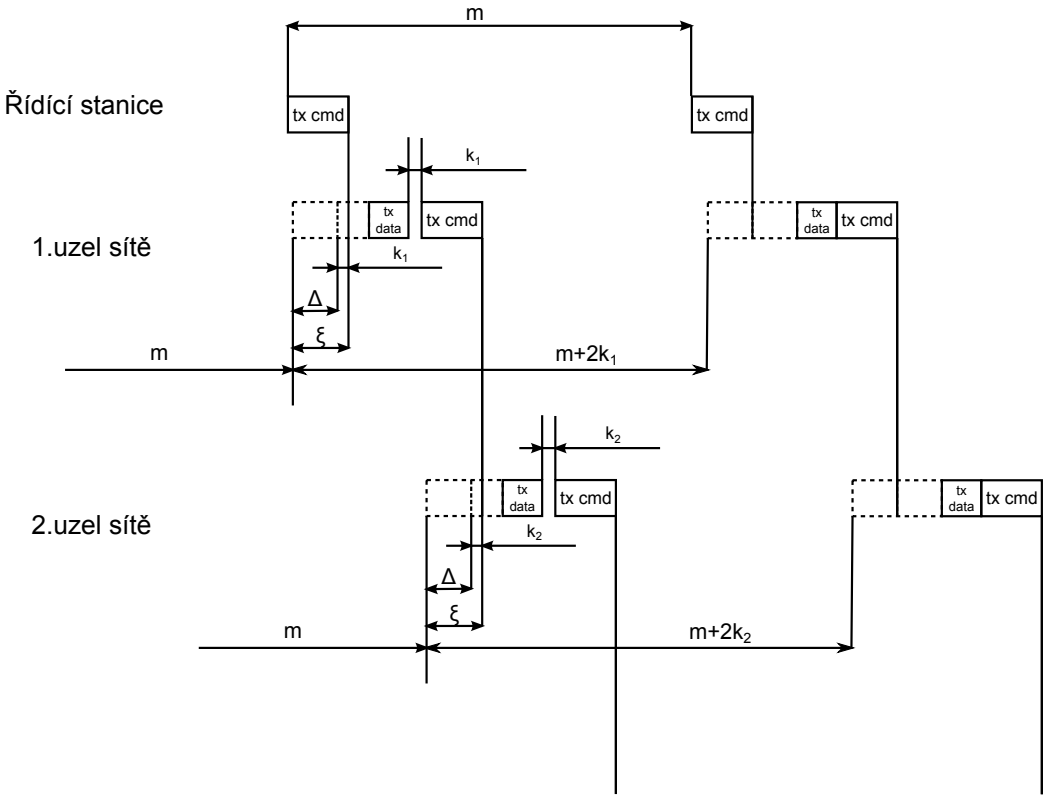
Dynamická korekce zprostředkuje naladění oscilátoru podle uzlu, který vyslal řídící rámec následujícím způsobem. V prvním kroku si zapamatuje hodnotu dynamické korekce k_n, posune hodnoty časovače pro stávající periodu o k_n a hodnoty časovače pro následující periodu o hodnotu $2 \cdot k_n$. Délku posledního časového slotu v ticích časovače o k_n prodlouží. V každém dalším kroku záleží na předchozí hodnotě k_n. Je-li předchozí hodnota k_n nulová, proběhne pouze posunutí událostí o aktuální k_n resp. $2 \cdot k_n$. Je-li předchozí hodnota k_n nenulová proběhne posunutí událostí o aktuální hodnotu k_n resp. $2 \cdot k_n$ a také prodloužení posledního slotu o aktuální hodnotu k_n.

Hodnota k_n, kde n představuje identifikátor uzlu je získána ze vztahu 3.3, kde ξ představuje dobu čekání na řídící rámec v ticích časovače a Δ představuje požadovanou dobu čekání na řídící rámec v ticích časovače.

$$k_n = \xi - \Delta \tag{3.3}$$

Ihned po připojení uzlu do sítě proběhne statická korekce, která posune události tak, aby při naprosto stejné frekvenci oscilátorů začal připojený uzel přijímat řídící rámce právě o Δ dříve. V další periodě již uzel z okamžiku přijmutí řídícího rámce odvodí hodnotu k_n a podle této hodnoty provede dynamickou korekci viz obrázek 3.3. Při ustáleném stavu zůstane uzel několik period v synchronizaci, tudíž bude předchozí hodnota k_n nulová. Díky nulové hodnotě k_n dojde k doladění pouze posunutím všech událostí, neboť jsou již oscilátory synchronizované s přesností na jeden tik časovače.

Tímto algoritmem a také skutečností, že všechny krystalové oscilátory uzlů jsou ve stejné toleranci je dosažena synchronizace podle kořenového uzlu sítě.



Obrázek 3.3: Ukázka mechanismu dynamické korekce

3.2.2 Formáty rámců na přístupové úrovni

Formáty rámců jsou popsány v normě IEEE 802.15.4-2003 [3]. Kvůli hardwarové podpoře rozpoznávání adresy je nutné dodržet typy rámců a strukturu jejich hlaviček.

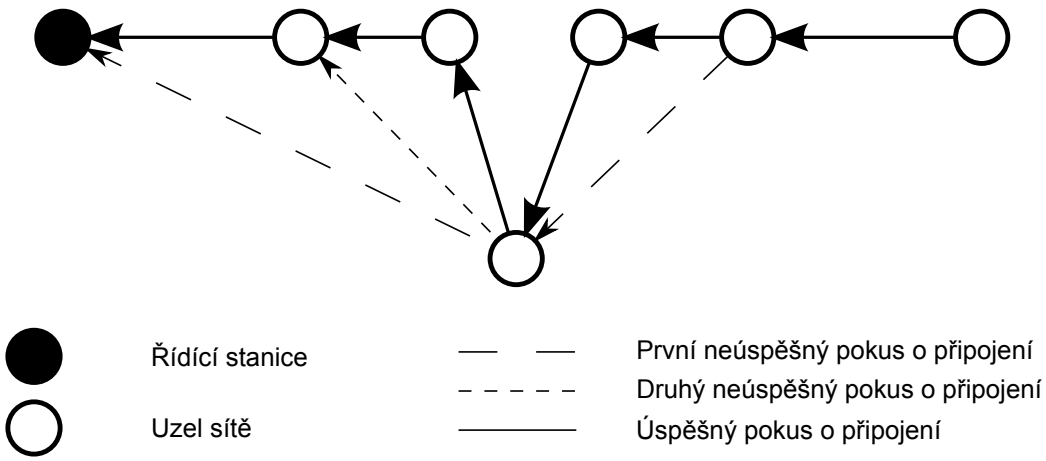
3.3 Síťová úroveň

Vzhledem k předpokladu lineární sítě je značně zjednodušeno směrování v této síti. Každý uzel má jen dva sousední uzly, jeden ve směru ke kořenu a jeden ve směru od kořenu. Data jsou přenášena mezi sousedními uzly směrem ke kořenu, řídicí rámce směrem od kořenu. Zjednoduší se také adresování uzlů na síťové úrovni. Proto je v síťové úrovni použita adresace z úrovně přístupové.

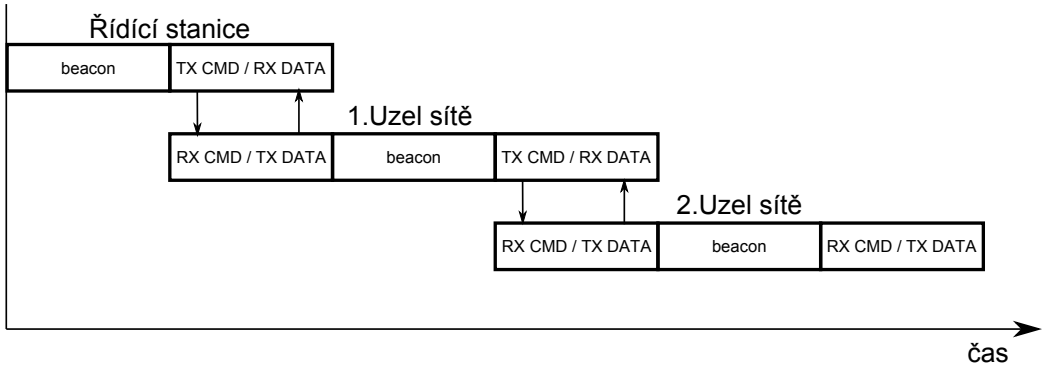
Na síťové úrovni je šest úloh, kterými se obsluhují nižší vrstvy: inicializace síťové vrstvy, vyslání a příjem řídícího rámce, vyslání a příjem datového rámce, připojení dalšího uzlu do sítě. Inicializaci síťové vrstvy představuje připojení uzlu do sítě. Pro sestavení lineární senzorické sítě byl navržen a realizován algoritmus popsaný v sekci 3.2.1.

Na začátku jsou všechny uzly zapnuty a čekají na beacon rámec. Řídící stanice vyšle beacon rámec a do uplynutí doby $t_{recvReq}$ přijímá žádosti o připojení do sítě. Každý uzel, který přijme tento beacon rámec si ze síly signálu odvodí kvalitu linky (LQI) a na tuto hodnotu nastaví nejnižší bajt své adresy. Vyšší bajt adresy nastaví na počet neúspěšných pokusů o připojení. Hodnotu LQI použije spolu s počtem neúspěšných pokusů k určení okamžiku vysílání. Čím lepší kvalita linky a čím více neúspěšných pokusů o připojení do sítě, tím déle vyšle uzel žádost o připojení do sítě. Uzel v síti si pamatuje jen poslední přijatou žádost, na tu odpoví povolením s adresou vítězného uzlu a čeká na potvrzení o přijetí povolení.

Tento algoritmus není příliš výpočetně náročný a je schopen vytvořit lineární síť i v některých speciálních případech obecné topologie. Na obrázku 3.4 je příklad sestavení lineární sítě na jednom speciálním případu topologie. V tomto případě by algoritmus bez mechanismu přidávání priority odmítnutých uzlů jeden uzel vynechal, nebo by pro uzel ve spodní části obrázku vybral cestu z posledního uzlu. Tím by se výrazně zvýšily nároky na energetickou náročnost při přenosu dat.



Obrázek 3.4: Ukázka sestavení lineární sítě



Obrázek 3.5: Časový diagram přenosu dat sítí

3.4 Transportní úroveň

Tato úroveň slouží k uložení dat přijmutých od sousedních uzlů a jejich předávání v síti. Na této vrstvě se ukládají dosud neodeslaná data, řídící rámce od kořenového uzlu sítě a data přijatá od uzlu vzdálenějšího od kořenu, slouží tudíž jako komunikační buffer.

Komunikace je řízena spouštěním jednotlivých úloh z aplikační vrstvy. Na obrázku 3.5 je znázorněn časový diagram přenosu dat a řídících rámců. Důsledkem tohoto časování je fakt, že si lineární síť uzlů můžeme představit jako posuvný registr, který každou periodu odešle jeden blok dat k řídícímu uzlu sítě, ale při vyslání řídícího rámce z kořenového uzlu se tento rámec rozšíří mezi všechny uzly během jedné periody.

3.5 Aplikační úroveň

Funkce aplikační úrovně byla rozdělena do čtyř úloh. Jedná se o měření parametrů prostředí, připojení uzlu do sítě, komunikace s uzlem ve směru ke kořenu, komunikace s uzlem ve směru od kořenu.

3.5.1 Měření parametrů prostředí

Měřených parametrů prostředí může být několik, v tomto případě bylo dle zadání zvoleno měření teploty. Aby bylo možné pro řešení úlohy využít existujících modulů, bez nutnosti připojit externí součástky, bylo pro měření teploty využito teplotní čidlo zabudované v procesoru. Aktuální teplotu představuje teplota jádra procesoru, kterou získávám z analogově digitálního převodníku. Tuto hodnotu uložím do komunikačního bufferu, aby ji transportní vrstva mohla odeslat ke kořenu.

3.5.2 Připojení uzlu do sítě

Připojení uzlu do sítě je součástí inicializace transportní úrovně, která následně inicializuje všechny nižší úrovně a spustí příslušné funkce pro připojení uzlu do sítě.

3.5.3 Komunikace s uzlem ve směru ke kořenu

V této úloze se nejprve přijme řídící rámec od uzlu blíže ke kořenu, zároveň se při přijmutí řídícího rámce provede statická resp. dynamická korekce a poté se vyšlou data z bufferu. Je-li uzel koncový, proběhne beacon fáze. V beacon fázi umožní koncový uzel dalším uzlům připojit se do sítě.

3.5.4 Komunikace s uzlem ve směru od kořenu

Tato úloha slouží ke zprostředkování komunikace mezi sousedním uzlem ve směru od kořenu a kořenem. Směrem ke kořenu se přenáší data, směrem od kořenu řídící informace. Nejdříve je vyslán řídící rámec a poté je přijat datový rámec.

4 Realizace navrženého systému

4.1 Fyzická úroveň

Na fyzické úrovni jsem vytvořil funkce pro inicializaci fyzické vrstvy, vyslání dat, příjem dat, zastavení příjmu dat a pro zjištění LQI.

Pro inicializaci fyzické vrstvy je potřeba předat funkci číslo kanálu, na kterém se bude vysílat nebo přijímat a vysílací výkon. Tato funkce také nastaví hardwarovou podporu rozpoznávání adresy.

Funkce pro vyslání dat má dva parametry, délku zprávy a ukazatel na pole znaků představujících zprávu. Tato funkce zapíše do bufferu vysílacího koprocesoru vysílaná data přes registr `RFD`. Formát vyslaného rámce odpovídá obrázku 2.5.

Příjem dat probíhá také přes registr `RFD`. Pokud je tento registr čten, vrací data z přijímacího bufferu, pokud je do něj zapisováno, data se zapisují do vysílacího bufferu. Čtení dat probíhá v obsluze přerušení. Přerušení je vyvoláno hardwarovým rozpoznáním adresy. Při 16 bitové cílové adrese je tato adresa porovnávána s registrem `SHORTADDRH:SHORTADDRL`, při 64 bitové adrese je porovnávána s registry `IEEE_ADDR0 - IEEE_ADDR7`. Toto platí za předpokladu, že identifikátor cílové sítě obsažený v adresním poli přijatého rámce souhlasí s registry `PANIDH:PANIDL`. Hardwarové rozpoznávání adresy podporuje také boroadcast adresy, kde jsou všechny bity nastaveny na jedničku.

Aby nebyla funkce přijímání dat blokující, je třeba ukončit přijímání dat. To se děje prostřednictvím funkce pro zastavení příjmu dat, která nastaví globální proměnnou `extStopRecv`. Tato proměnná zprostředkuje konec čekání na data z fyzické vrstvy.

Hodnotu LQI počítám podle vztahu 4.1. Konstanty ve vztahu byly zjištěny experimentálně.

$$LQI = (RSSI - 45) \cdot 4 + 360. \tag{4.1}$$

4.2 Přístupová úroveň

Na přístupové úrovni jsem vytvořil funkce pro vysílání a příjem řídících rámců, beacon rámců, datových rámců a potvrzovacích rámců. Pro identifikaci rámce slouží nejnižší tři bity v prvním bajtu pole frame control, které je v hlavičce každého rámce přístupové vrstvy.

Funkce pro příjem periodicky přijímají rámce s cílovou adresou uzlu (díky hardwarové podpoře rozeznávání adresy). Každá tato funkce použije jako ná-

vratovou hodnotu obsah rámce jen pokud je rámec správného typu. Pokud neodpovídá typ rámce, funkce přijímá další rámec. Toto lze zastavit jen zavoláním příslušné funkce pro přerušení přijímání paketů.

Funkce pro vysílání rámců vysílají rámce s hlavičkou podle obrázku 2.6 a s předaným obsahem.

Synchronizaci času jsem implementoval podle navrženého algoritmu viz 3.2.1. Vytvořil jsem funkce pro statickou a dynamickou korekci a funkci pro přechod do režimu nízké spotřeby.

4.3 Síťová úroveň

Na síťové úrovni jsem vytvořil funkce pro připojení uzlu do sítě, vyslání a přijmutí řídícího a datového rámce a funkci pro připojení dalšího uzlu do sítě. Funkce pracují na základě protokolu stop&wait.

Funkce pro vysílání daných rámců vyšlou příslušný rámec a pokud nepřijmou do určité doby potvrzovací rámec, považuje se rámec za nepřenesený, což funkce signalizuje nulovou návratovou hodnotou.

Funkce pro příjem daných rámců přijímají po nastavenou dobu rámec daného typu. Pokud rámec v časovém intervalu přijmou, pošlou obratem potvrzovací rámec a jako návratovou hodnotu použijí ukazatel na obsah rámce. Pokud daný rámec v časovém rozmezí nepřijde vrací funkce ukazatel na hodnotu 0 (null pointer).

Připojení uzlu do sítě je zprostředkováno funkcí pro inicializaci síťové úrovně. Tato funkce bezprostředně souvisí s funkcí pro přidání uzlu do sítě. Na levé straně obrázku 3.1 pracuje funkce pro přidání uzlu do sítě, na pravé straně funkce pro připojení uzlu do sítě. Koncový uzel sítě čeká na beacon rámec, ve kterém je obsažen aktuální čas. Jakmile uzel přijme tento rámec, spustí funkci pro synchronizaci a vypočítá si hodnoty nadcházejících časových okamžiků a k nim příslušné hodnoty sleep timeru. Po vyslání beacon rámce čeká uzel v síti dobu $t_{recvReq}$, po kterou přijímá žádosti o připojení do sítě. Stejnou má maximální čekací dobu i uzel, který se chce k síti připojit.

$$t_{recvReq} = \frac{(m \cdot n)8}{p} = \frac{(300 \cdot 18) \cdot 8}{250000} = 172,8ms \tag{4.2}$$

Hodnota m značí počet počet časových okamžiků, kdy je možno vyslat žádost o připojení do sítě. Tuto hodnotu jsem zvolil 300, aby byl dostatečný prostor pro uzly, kterým se již několikrát nepodařilo připojit a chtějí se připojit k uzlu, od kterého mají $LQI = 255$.

Hodnota n značí počet bajtů na jednu žádost o připojení do sítě. V této hodnotě je zahrnuto 11B na hlavičku a obsah přístupové vrstvy, 4B preambule, 1B značka začátku rámce a 2B CRC zabezpečovací polynom.

velikost:	1 B	4 B	2B
obsah:	id uzlu	časová známka	naměřená hodnota

Obrázek 4.1: Formát přenášených datových bloků

Hodnota p je přenosová rychlost viz tabulka 2.1.

Připojovaný uzel čeká stejnou dobu. Po uplynutí nastaveného časového okamžiku vyšle koncový uzel sítě povolení k připojení do sítě a čeká na potvrzení dobu t_{ACK}.

$$t_{ACK} = \frac{11 \cdot 8}{250000} = 0,352ms \tag{4.3}$$

Mezitím čeká připojovaný uzel dobu $t_{recvResp}$.

$$t_{recvResp} = \frac{20 \cdot 8}{250000} = 0,64ms \tag{4.4}$$

4.4 Transportní úroveň

Na transportní úrovni jsou vytvořeny funkce pro inicializaci této úrovně řídící stanicí, pro inicializaci uzlem sítě, funkci pro poslání dat a funkci pro přijmutí dat.

4.4.1 Formát přenášených dat

Data z uzlu do řídící stanice obsahují 1B identifikátor uzlu, 4B časovou známku a 2 B naměřených dat, jak je zobrazeno na obrázku 4.1.

4.4.2 Výpočet komunikačního bufferu

Na transportní úrovni je realizován buffer. Tento buffer je zvlášť potřebný na uzlech blízko kořenovému uzlu, neboť při doručování dat jsou tyto uzly nejvíce zatíženy, protože musí uchovávat při bezproblémovém přenosu jeden blok dat od každého uzlu. Při velikosti 7 B na jeden blok dat by při 255 uzlech sítě musel uzel u kořene odeslat $7 \cdot 255 = 1785B$, což při maximální délce obsahu datového rámce

$$N_{dataPayload} = 127 - 9 - 2 = 116B \tag{4.5}$$

(9 B hlavička a 2 B CRC zabezpečovací sekvence) není možné odeslat v jednom rámci.

Tento problém je vyřešen následovně. Uzel nejdříve přijme řídící rámec od uzlu směrem ke kořenu a potom vyšle směrem ke kořenu datový rámec, ve kterém podle potřeby nastaví příznak zřetězení. Uzel ve směru ke kořenu sleduje tento příznak a dokud je tento příznak v aktivní úrovni, přijímá data. Tímto mechanismem mohou uzly využít čas pro beacon fázi, kterou již nerealizují, neboť nejsou koncovými uzly. Uzel tak může ke kořenu přenést

$$M = \left\lfloor \frac{N_{BCN}}{N_{DATApkt} + N_{ACK}} \right\rfloor = \left\lfloor \frac{300 \cdot 18 + 20 + 11}{132 + 11} \right\rfloor = 37 \qquad (4.6)$$

paketů navíc. Jestliže každý paket může obsahovat 116 B pak je maximální počet přenesených bloků dat zaokrouhlený na celá čísla $\left\lfloor \frac{37 \cdot 116}{7} \right\rfloor = 613$. Při předpokladu maximálního počtu uzlů 255 je toto číslo dostačující.

Z počtu bajtů, které je možné přenést za jednu periodu, vyplývá velikost komunikačního bufferu.

$$N_{bfr} = (M + 1) \cdot N_{dataPayload} = (37 + 1) \cdot 116 = 4408B \qquad (4.7)$$

Tento buffer je realizován jako kruhový a při jeho naplnění nejsou další datové rámce potvrzovány.

4.4.3 Popis funkcí

Funkce pro zaslání dat přijme řídící rámec, který použije jako svou návratovou hodnotu. V případě, že nedojde k příjmu rámce, vrací tato funkce ukazatel na null. Po příjmu řídícího rámce se funkce stará o vyprázdnění bufferu. Dokud jsou data v bufferu a uzel ve směru ke kořenu potvrzuje vyslaná data, posílá funkce datové rámce s příslušným příznakem zřetězení. Poslední přenášená data obsahují příznak zřetězení v neaktivní úrovni, tím je signalizováno uzlu ve směru ke kořenu, že je tento datový rámec poslední.

Funkce pro přijmutí dat vyšle řídící rámec uzlu a po jeho potvrzení přijímá datové rámce dokud obsahují příznak zřetězení a je dostatek volného místa v komunikačním bufferu.

Inicializace transportní úrovně v případě uzlu probíhá cyklickou inicializací síťové vrstvy dokud se uzel nepřipojí do sítě, nebo dokud nevyčerpá zadaný počet pokusů. Po úspěšném připojení do sítě si uzel nastaví identifikátor. Pokud uzel již má nastaven identifikátor, nezmění si jej a nechá si původní.

Inicializace transportní úrovně v případě řídící stanice probíhá opakujícími se pokusy o přidání uzlu sítě. Tomuto uzlu předává v beacon rámci informaci o aktuálním čase. Informaci o aktuálním čase po každém pokusu stanice přepočítá tak, aby byla vždy aktuální. V řídícím rámci pro přijetí uzlu do sítě pošle stanice uzlu informaci o kanálu pro předávání dat.

4.5 Aplikační úroveň

4.5.1 Řídící stanice

Pro řídící stanici je vzhledem k časové synchronizaci důležitý pouze okamžik pro příjem dat a okamžik pro měření parametrů prostředí. Okamžik pro odeslání dat je nahrazen komunikací s počítačem přes sériovou linku.

Nejdříve musí stanice dostat informaci o aktuálním čase, aby si tak mohla vytvořit časovou značku, podle které určí aktuální čas. Získanou časovou značku použije funkce pro inicializaci transportní vrstvy.

Po inicializaci transportní vrstvy se již pravidelně střídají události pro příjem dat, měření parametrů prostředí a komunikaci s počítačem.

4.5.2 Uzel sítě

Uzel sítě nejprve inicializuje transportní úroveň. Po úspěšné inicializaci opakuje tři události: odeslání dat ke kořenu, přijmutí dat od vzdálenějšího uzlu a měření parametrů prostředí.

Po změření parametrů prostředí zapíše do komunikačního bufferu blok dat se svým identifikátorem, naměřenou hodnotou a časem, kdy byla hodnota naměřena. Tato data budou spolu s ostatními přenesena směrem ke kořenu v příslušný časový okamžik daný synchronizací sítě.

5 Závěr

Seznámil jsem se s implementací protokolu IEEE802.15.4 na procesoru CC2431, s architekturou tohoto procesoru a s volně šiřitelným překladačem SDCC. Překladač SDCC jsem použil pro překlad vytvořených zdrojových souborů v jazyce C. Navrhl a realizoval jsem protokol pro bezdrátovou senzorickou síť.

Protokol zabezpečí díky navrženému a realizovanému algoritmu vytvoření lineární sítě i v některých speciálních případech obecné topologie. Navržený protokol je typu TDMA a vyžaduje tak přesnou časovou synchronizaci všech uzlů sítě.

Pro synchronizaci byl navržen a realizován výpočetně jednoduchý algoritmus, který zabezpečil synchronizaci libovolně rozsáhlé lineární sítě po druhé periodě vysílání.

Bohužel jsem i přes veškerou snahu nebyl schopen odladit programové vybavení tak, aby síť pracovala dlouhodobě. Uzly sítě vytvoří lineární síť a doručují data do monitorovací stanice, ale po náhodném časovém intervalu, v řádu minut až desítek minut, se tato síť rozpadne. Jádro tohoto problému patrně tkví ve správě paměti. Jelikož překladač SDCC nemá podporu pro debugování přímo v zařízení a chyba se projevuje až po několika minutách, je velice obtížné a časově náročné chybu odhalit.

I přes tyto nedokonalosti má uzel sítě pracující podle navrženého protokolu poměrně nízkou spotřebu energie (viz příloha A.1).

Za největší přínos této práce považuji algoritmus pro synchronizaci uzlů sítě, navržený komunikační protokol a také skutečnost, že lze vytvořit programové vybavení pro bezdrátovou senzorickou síť bez použití komerčního překladače.

6 Přehled zkratek

Zkratka	Význam	Vysvětlení zkratky
AES	advanced encryption standard	standard pro symetrické šifrování
ADC	analog to digital converter	analogově digitální převodník
BPSK	binary phase-shift keying	způsob modulace fázovým posunem
CCA	clear channel assessment	zjištění, je-li komunikační kanál volný
CRC	cyclic redundancy check	polynom pro odhalení chyb ve zprávě
CSMA-CA	carrier sense multiple access with collision avoidance	metoda vícenásobného přístupu s naslouchánim nosné
DMA	direct memory access	přímý přístup do paměti
FCS	frame check sequence	sekvence bitů pro ověření správnosti rámce
GTS	guaranteed time slot	časový slot, ve kterém je zaručeno, že nebude žádná jiná stanice vysílat
LQI	link quality indication	indikátor kvality spojení
LR-WPAN	low rate wireless personal area network	bezdrátová síť s nízkou spotřebou
MAC	medium access control	přístupová vrstva
MHR	MAC header	hlavička přístupové vrstvy
MFR	MAC footer	konec rámce přístupové vrstvy
O-QPSK	offset quadrature phase-shift keying	způsob modulace fázovým posunem
PAN	personal area network	druh sítě
PSDU	physical service data unit	data přenášená rámcem na fyzické úrovni
RAM	random access memory	paměť s přímým přístupem
RSSI	received signal strength indication	síla přijímaného signálu
SDCC	small device C compiler	volně šiřitelný překladač jazyka C (viz `http://sdcc.sourceforge.net/`)

Zkratka	Význam	Vysvětlení zkratky
SFD	start of frame delimiter	značka začátku rámce na fyzické úrovni
SFR	special function registers	registry speciálních funkcí
SHR	synchronization header	synchronizační hlavička
SPI	serial peripheral interface	sériové periferní rozhraní
TDMA	time division multiple access	plánovaný komunikační protokol
UART	universal asynchronous receiver / transmitter	rozhraní pro asynchronní sériovou komunikaci
USART	universal synchronous / asynchronous receiver and transmitter	zařízení, které je schopno komunikovat jako SPI i jako UART

Literatura

[1] *A True System-on-Chip solution for 2.4 GHz IEEE 802.15.4 / ZigBee ®*. Texas Instruments, Post Office Box 655303, Dallas, Texas 75265, 2008.

[2] C.Cano et al.: Low energy operation in WSNs: A survey of preamble sampling MAC protocols. *Comput. Netw.*, doi:10.1016/j.comnet.2011.06.022.

[3] IEEE Computer Society: *Part 15.4: wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (LR-WPANs)*. Institute of Electrical and Electronics Engineers, New York, 2003, ISBN 07-381-3686-7.

A Přílohy

A.1 Měření spotřeby

Spotřeba uzlu sítě byla měřena podle schématu na obrázku A.1. Zařízení bylo napájeno napětím $U = 3,3V$ (Ch1) z laboratorního zdroje. Toto napětí bylo konstantní, není tudíž zobrazeno ve výsledných osciloskopických měřeních. Odpor R má hodnotu $4,3\Omega$. Jelikož jsem věnoval maximum času ladění programu, je výsledná naměřená spotřeba vyšší než by bylo nutné. Během měření byla aktivní led dioda pro snžší indikaci stavu zařízení.

A.1.1 Výpočet spotřeby uzlu uvnitř sítě

Na obrázku A.2 je zobrazen průběh odběru zařízení během jedné periody. Nyní se zaměřím na jednotlivé úseky (sloty).

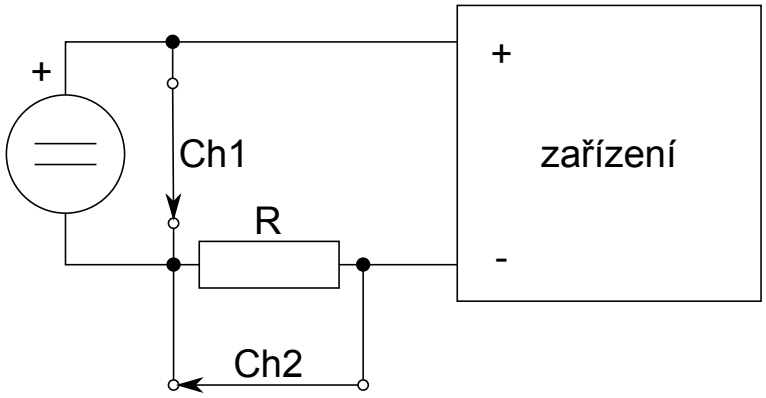
Výpočet spotřeby slotu pro měření parametrů prostředí

Podrobný průběh odběru zařízení při měření parametrů prostředí je na obrázku A.3. Spotřebu uzlu při tomto slotu jsem vypočítal podle vztahu A.1.

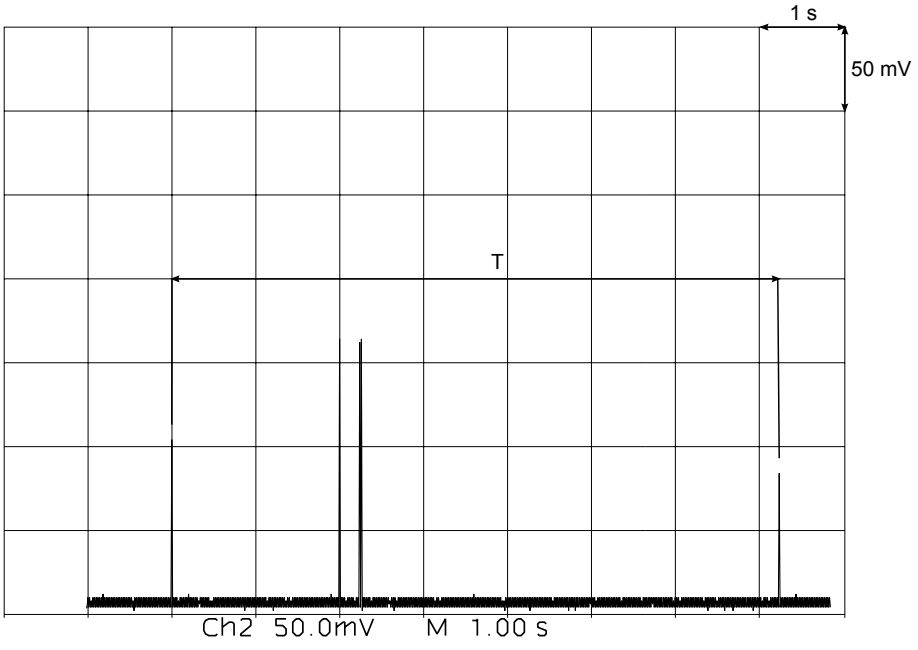
$$\begin{aligned} P_m &= I_{ADC} \cdot t_{ADC} + I_{phy\_init} \cdot t_{phy\_init} = \\ &= \frac{U_{ADC}}{R} \cdot t_{ADC} + \frac{U_{phy\_init}}{R} \cdot t_{phy\_init} = \\ &= \frac{0,252}{4,3} \cdot 50 \cdot 10^{-6} + \frac{0,086}{4,3} \cdot 650 \cdot 10^{-6} = \\ &= 2,930 \cdot 10^{-6} + 13 \cdot 10^{-6} = 15,930 \cdot 10^{-6}\ As \end{aligned} \tag{A.1}$$

Jelikož jsem při přípravě uzlů pro měření zapomněl ve zdrojovém kódu odstranit inicializaci fyzické vrstvy, je tato spotřeba znatelně vyšší. Při vynechání inicializace fyzické vrstvy by měla být spotřeba definována vztahem A.2.

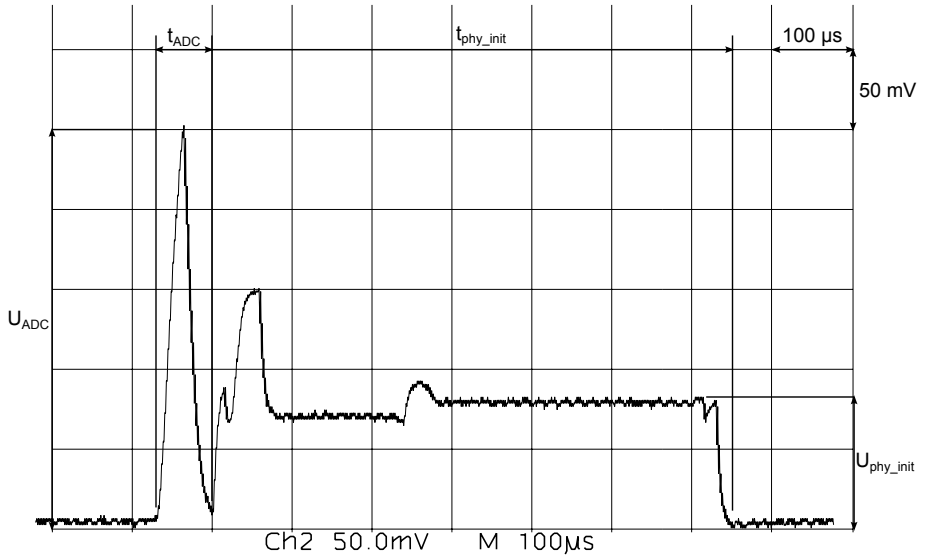
$$\begin{aligned} P_{mO} &= I_{ADC} \cdot t_{ADC} = \frac{U_{ADC}}{R} \cdot t_{ADC} = \\ &= \frac{0,252}{4,3} \cdot 50 \cdot 10^{-6} = 2,930 \cdot 10^{-6}\ As \end{aligned} \tag{A.2}$$



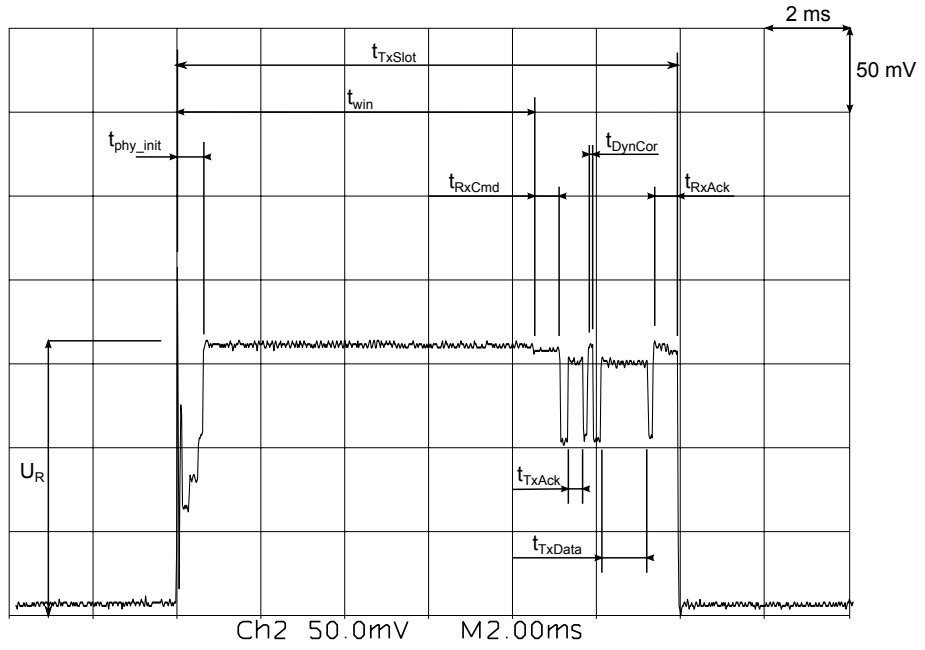
Obrázek A.1: Schéma pro měření spotřeby uzlu



Obrázek A.2: Odběrová charakteristika uzlu v síti během periody vysílání



Obrázek A.3: Odběrová charakteristika uzlu v síti při měřícím slotu



Obrázek A.4: Odběrová charakteristika uzlu v síti při slotu pro odeslání dat

Výpočet spotřeby slotu pro odeslání dat

Podrobný průběh odběru zařízení při odeslání dat je na obrázku A.4. Spotřebu uzlu při tomto slotu jsem vypočítal podle vztahu A.3

$$P_{TxData} = I_R \cdot (t_{TxSlot}) = \frac{U_R}{R} \cdot t_{TxSlot}$$

$$= \frac{0,162}{4,3} \cdot 12 \cdot 10^{-3} = 4,520 \cdot 10^{-4}\ As \tag{A.3}$$

Spotřebu uzlu v tomto slotu znatelně ovlivňuje doba t_{win}. Velikost doby t_{win} je potřeba snižovat s ohledem na synchronizaci uzlů sítě. Vzhledem k navrženému algoritmu synchronizace by měla velikost doby vypočítaná podle vztahu A.4 dostačovat.

$$t_{winO} = T_{ST} \cdot 50 = 30,517578125 \cdot 10^{-6} \cdot 50 = 1,526 \cdot 10^{-3}\ s \tag{A.4}$$

Tím se sníží spotřeba na hodnotu definovanou vztahem A.5.

$$P_{TxDataO} = I_R \cdot (t_{TxSlot}) = \frac{U_R}{R} \cdot (t_{TxSlot} - t_{win} + t_{winO}) =$$

$$= \frac{0,162}{4,3} \cdot (12 \cdot 10^{-3} - 9,155 \cdot 10^{-3} + 1,526 \cdot 10^{-3}) = 1,647 \cdot 10^{-4}\ As \tag{A.5}$$

Takové spotřeby v tomto slotu může ovšem dosáhnout pouze uzel, který je předposlední. Spotřeba uzlu totiž závisí na velikosti přenášených dat. Uzly blíže ke kořeni přenášejí kromě svých dat také data všech následujících uzlů. Délka předávání dat by neměla přesáhnout dobu t_{BCN}. Spotřeba uzlu, který je aktivní také v čase pro beacon fázi je definovaná vztahem A.6.

$$P_{TxDataOMax} = P_{TxData} + I_R \cdot (t_{BCN}) = P_{TxDataMax} + \frac{U_R}{R} \cdot t_{BCN} =$$

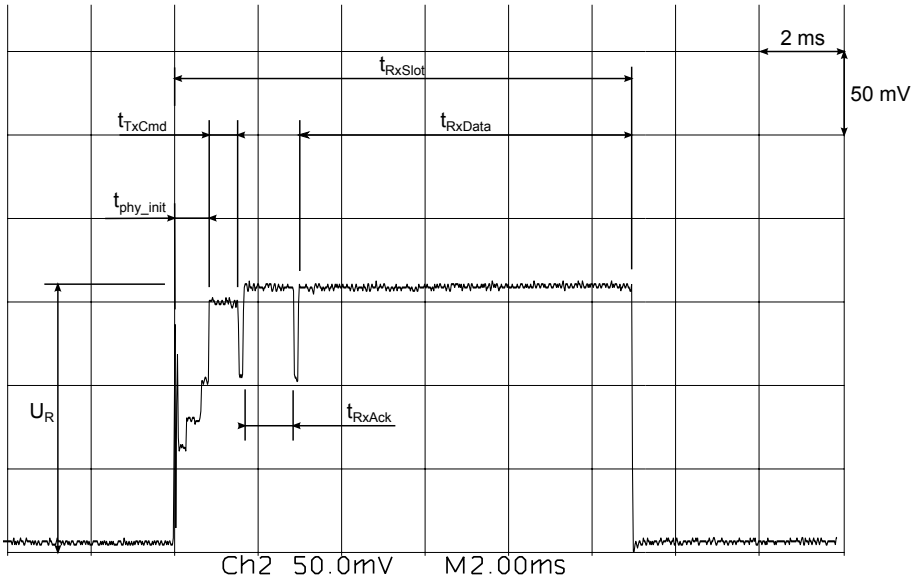
$$= 1,647 \cdot 10^{-4} + \frac{0,162}{4,3} \cdot 180 \cdot 10^{-3} = 6,782 \cdot 10^{-3}\ As \tag{A.6}$$

Výpočet spotřeby slotu pro přijmutí dat

Podrobný průběh odběru zařízení při přijmutí dat je na obrázku A.5. Spotřebu uzlu při tomto slotu jsem vypočítal podle vztahu A.7.

$$P_{RxData} = I_R \cdot (t_{RxSlot}) = \frac{U_R}{R} \cdot t_{RxSlot} =$$

$$= \frac{0,162}{4,3} \cdot 11 \cdot 10^{-3} = 4,144 \cdot 10^{-4}\ As \tag{A.7}$$



Obrázek A.5: Odběrová charakteristika uzlu v síti při slotu pro přijmutí dat

Na spotřebě P_{RxData} se projevila skutečnost, že další uzel sítě již nekomunikoval. Spotřeba tohoto slotu může být maximálně $P_{RxDataMax}$ definovaná vztahem A.8.

$$P_{RxDataMax} = I_R \cdot (t_{RxSlot}) = \frac{U_R}{R} \cdot (t_{phy\_init} + t_{TxCmd} + t_{RxAck} + t_{BCN} =$$

$$= \frac{0,162}{4,3} \cdot (0,8 \cdot 10^{-3} + 0,8 \cdot 10^{-3} + 1,3 \cdot 10^{-3} + 180 \cdot 10^{-3}) = 6,891 \cdot 10^{-3}\ As \quad \text{(A.8)}$$

Spotřeba při tomto slotu taktéž závisí na velikosti a počtu datových rámců.

Výpočet celkové spotřeby uzlu v síti

Měřený uzel sítě měl na jednu periodu spotřebu danou vztahem A.9.

$$P_{celk} = P_m + P_{TxData} + P_{RxData} =$$

$$= 15,930 \cdot 10^{-6} + 4,520 \cdot 10^{-4} + 4,144 \cdot 10^{-4} = 8,8233 \cdot 10^{-4}\ As \quad \text{(A.9)}$$

Při spotřebě P_{celk} by uzel potřeboval na jeden rok provozu baterie o celkové kapacitě 1064,630 mAh (viz vztah A.10).

$$K_{celk} = \frac{365 \cdot 24 \cdot 60 \cdot 60}{T} \cdot P_{celk} = \frac{365 \cdot 24 \cdot 60 \cdot 60}{7,26} \cdot 8,8233 \cdot 10^{-4} =$$

$$= 3832,667\ As = 1064,630\ mAh \tag{A.10}$$

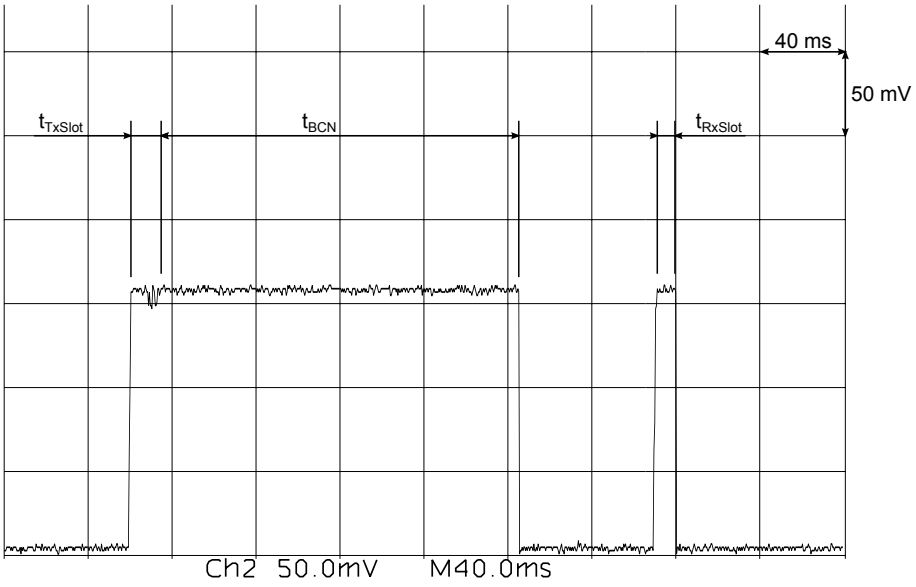
Při výše popsané minimalizaci spotřeby by měl uzel spotřebu v rozmezí daném vztahem A.11. Je nutné podotknout, že maximální spotřeby by uzel nemohl dosahoval dlouhodobě ani při teoretickém počtu uzlů 255. Údaj $P_{celkMax}$ představuje stav při maximálním zatížení a využití maximálního možného průtoku dat. Při běžném provozu bez chybného přenosu by byla spotřeba uzlu lineárně závislá na počtu uzlů přes něj připojených.

$$P_{celkMin} = P_{mO} + P_{TxDataO} + P_{RxData} =$$
$$= 2,930 \cdot 10^{-6} + 1,647 \cdot 10^{-4} + 4,144 \cdot 10^{-4} = 5,8203 \cdot 10^{-4}\ As$$
$$P_{celkMax} = P_{mO} + P_{TxDataOMax} + P_{RxDataMax} =$$
$$= 2,930 \cdot 10^{-6} + 6,782 \cdot 10^{-3} + 6,891 \cdot 10^{-3} = 136,7593 \cdot 10^{-4}\ As \tag{A.11}$$

Při minimální spotřebě a periodě 7,26 s by uzel sítě potřeboval na jeden rok provozu baterie o celkové kapacitě 702,284 mAh (viz vztah A.12).

$$K_{celkMin} = \frac{365 \cdot 24 \cdot 60 \cdot 60}{T} \cdot P_{celkMin} = \frac{365 \cdot 24 \cdot 60 \cdot 60}{7,26} \cdot 5,8203 \cdot 10^{-4} =$$

$$= 2528,223\ As = 702,284\ mAh \tag{A.12}$$



Obrázek A.6: Odběrová charakteristika koncového uzlu při slotu pro odeslání dat

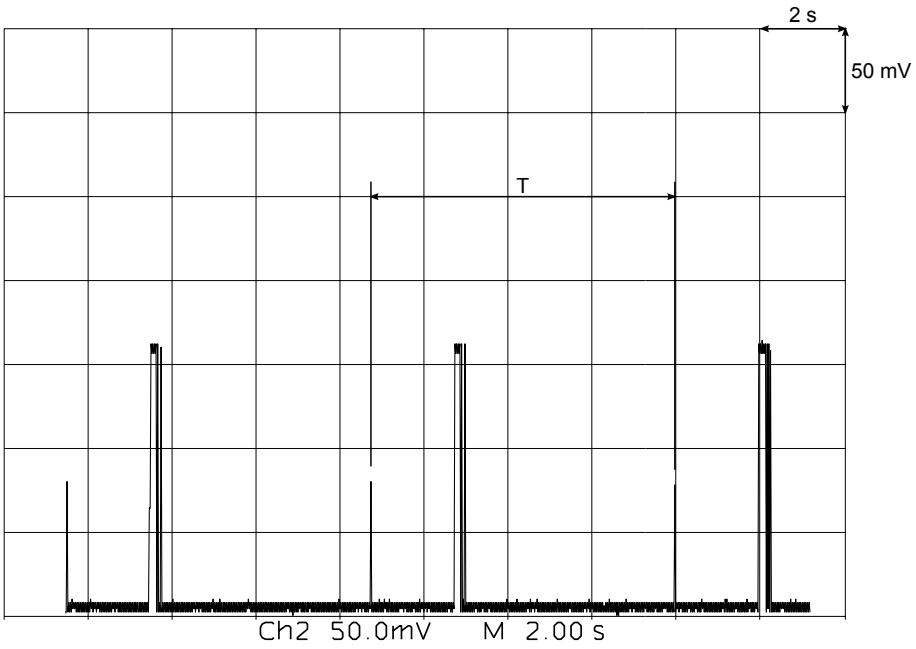
A.1.2 Výpočet spotřeby koncového uzlu sítě

Ihned po odeslání dat vyšle koncový uzel sítě beacon rámec a čeká, jestli se bude chtít další uzel připojit. Velkou část spotřeby tvoří aktivní naslouchání, proto by měl poslední uzel sítě větší nároky na spotřebu. Tuto nepříjemnost by bylo možné vyřešit buď zasíláním beacon rámce jednou za určitý počet period, nebo vysíláním beacon rámce jen během několika prvních period. Bez takového mechanismu má koncový uzel spotřebu danou vztahem A.13.

$$P_{celkKon} = P_{mO} + P_{TxDataO} + (I_R \cdot (t_{BCN}) =$$
$$= P_{mO} + P_{TxDataO} + (\frac{U_R}{R} \cdot t_{BCN}) =$$
$$= 2,930 \cdot 10^{-6} + 1,647 \cdot 10^{-4} + (\frac{0,162}{4,3} \cdot 180 \cdot 10^{-3}) = 69,490 \cdot 10^{-4}\ As \quad \text{(A.13)}$$

Při takové spotřebě by koncový uzel sítě potřeboval na rok provozu baterie o celkové kapacitě 8384,744 mAh (viz vztah A.14), což je značně nevýhodné.

$$K_{celk} = \frac{365 \cdot 24 \cdot 60 \cdot 60}{T} \cdot P_{celkKon} = \frac{365 \cdot 24 \cdot 60 \cdot 60}{7,26} \cdot 69,490 \cdot 10^{-4} =$$
$$= 30185,078\ As = 8384,744\ mAh \quad \text{(A.14)}$$



Obrázek A.7: Odběrová charakteristika koncového uzlu během periody vysílání

A.1.3 Zhodnocení výsledků měření

Spotřeba uzlu je závislá na počtu následujících uzlů. Tuto nepříjemnou vlastnost lze řešit například odesíláním dat jen v případě, že nastane změna měřené veličiny. Koncový uzel sítě má poměrně velkou spotřebu energie, kterou lze snížit (viz A.1.2).