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

Bakalářská práce

Realizace experimentální sítě s ZigBee

Plzeň, 2012 Jan Bělohoubek

Prohlášení

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

V Plzni dne 22.4.2012

Jan Bělohoubek

Abstract

Realization of the experimental ZigBee network

This bachelor thesis deals with the usability of ZigBee network in fire suit project realized by KET FEL ZČU. First part of this text is about wireless sensor networks and ZigBee technology in general.

The main part covers realization of the experimental ZigBee network usable in given conditions. In realized network all devices are clearly identified, even though ZigBee stochastic addressing is used. Data can be transferred from intended nodes to coordinator and command may be delivered to specified device.

As hardware platform served RC2400HP modules (based on TI SoC CC2530) placed on development boards. Network status visualization and interface for sending commands is implemented in graphical application for PC.

Poděkování

Rád bych na tomto místě poděkoval vedoucímu své bakalářské práce Ing. Jiřímu Čengerymu, Ph.D. a Ing. Petru Kašparovi, Ph.D. z Katedry technologií a měření FEL ZČU za poskytnuté zázemí a metodické vedení v průběhu celé práce. Dále Ing. Jiřímu Ledvinovi, CSc. z Katedry informatiky a výpočetní techniky FAV ZČU za cenné rady z oblasti senzorických sítí a svým nejbližším za vytvoření podmínek pro psaní práce, a zejména pak za čas věnovaný korektuře textu.

Obsah

1 Úvod **1**

2 Senzorické sítě **2**
2.1 Historie senzorických sítí 2
2.2 Topologie bezdrátových senzorických sítí 4
2.3 Spotřeba prvku senzorické sítě 5
2.4 Specifika napájení 6
2.5 Významní výrobci hardware pro senzorické sítě 6
2.5.1 Atmel Corporation 6
2.5.2 Digi International 6
2.5.3 Freescale Semiconductor Inc 6
2.5.4 Infineon Technologies AG 6
2.5.5 Texas Instruments 7
2.6 Standardy IEEE 802.15 7
2.6.1 802.15.1 8
2.6.2 802.15.3 8
2.6.3 802.15.4 8
2.6.4 802.15.6 8
2.6.5 Protokolové zásobníky založené na 802.15.4 8
2.7 Standardy ZigBee 10
2.7.1 ZigBee RF4CE 10
2.7.2 ZigBee 10

3 Zásobník ZigBee **11**
3.1 Architektura zásobníku ZigBee 11
3.1.1 Síťová vrstva (NWK) 12
3.1.2 Pomocná aplikační podvrstva (APS) 12
3.1.3 Aplikační rámec 12
3.1.4 Objekt zařízení ZigBee a profil zařízení ZigBee 13
3.2 Adresování v ZigBee 13
3.2.1 Přidělení síťové adresy v ZigBee 13

3.2.2 Přidělení síťové adresy v ZigBee PRO 14
3.3 Směrování v ZigBee 14
3.3.1 Cesty v ZigBee 15
3.3.2 Směrovací tabulka 15
3.4 Zabezpečení přenosu 15
3.4.1 Rušení v ZigBee 16
3.5 ZigBee Z-Stack 16

4 Použitý hardware **18**
4.1 TI CC2530 18
4.1.1 Úsporné režimy zařízení CC2530 19
4.2 TI CC2591 19
4.3 Spotřeba RC2400HP 19
4.3.1 Orientační měření spotřeby RC240HP DB - multimetr 20

5 Aplikace pro RC2400HP **21**
5.1 Vývojové prostředky 22
5.1.1 IAR Embeded Workbench 22
5.1.2 SmartRF Flash Programmer 22
5.1.3 Alternativní vývojové prostředky 22
5.2 Charakteristika realizované sítě 23
5.3 Základy práce se Z-Stackem 23
5.4 ARP tabulka 24
5.5 Komunikace s nadřazeným systémem 26
5.5.1 Komunikační protokol 27

6 Obslužná aplikace pro PC **30**
6.1 Vývojové prostředky 30
6.2 Implementace 31

7 Závěr **32**

8 Přehled zkratek **33**

A Příručka aplikace uart-app **I**

B Osciloskopická měření **VI**

C Obsah přiloženého CD **XII**

1 Úvod

V dnešní době se ve stále větší míře používají bezdrátové sítě. Jejich rozmach přímo souvisí s rozvojem senzorických sítí, kterými se zabývá tato práce.

Senzorické sítě se postupem času prosazují ve stále více oblastech lidské činnosti. *Katedra technologií a měření FEL ZČU v Plzni* v současnosti řeší projekt inteligentního hasičského obleku, který bude shromažďovat informace z okolí zasahujícího hasiče a následně je předávat veliteli zásahu. Pro tento účel se výborně hodí senzorické sítě. V projektu se počítá s využitím dvou sítí - jedné typu *BAN*, která bude shromažďovat informace z obleku hasiče a druhé typu *WLAN*, jejímž úkolem bude přenášet zjištěné hodnoty k velitelskému stanovišti. Ve své práci se zabývám druhou z uvedených.

V rámci práce bylo mým úkolem seznámit se s technologiemi bezdrátových senzorických sítí, zejména se standardem *ZigBee*, a dále s hardware dostupným na *KET FEL ZČU v Plzni*. Hlavním cílem práce pak bylo vytvořit funkční demonstrační software, jenž si poradí se specifikem této sítě, tedy s pohyblivostí uzlů.

Softwarové vybavení by mělo umožnit zasílání příkazů na libovolné zařízení v síti a zároveň sběr dat ze sítě. Mělo by být jednoduše rozšiřitelné, aby mohlo být využito jako základ pro další práce na projektu. Síť by měla být schopna, prostřednictvím některého z modulů, interakce s uživatelem - nejlépe prostřednictvím rozhraní na osobním počítači.

2 Senzorické sítě

V současnosti neustále narůstá množství různých senzorů kolem nás. Zvyšující se nároky na bezpečnost a snaha o co největší automatizaci ve všech oborech lidské činnosti posouvají vývoj v této oblasti rychle kupředu. Je třeba měřit stále více veličin, provádět jejich vyhodnocování a na zjištěné skutečnosti patřičně reagovat v co nejkratším možném čase. Často nelze provádět měření, vyhodnocení i reakci na naměřené hodnoty na jednom místě. Z toho důvodu jsou senzory propojovány do sítí různého rozsahu a zjištěná data jsou po nich přenášena k vyhodnocení nebo pro archivaci ke statistickým účelům.

Uzly senzorické sítě mohou být propojeny různým způsobem - například klasickým metalickým vedením. Nejčastěji však bezdrátově pomocí rádiových vln. Lze nalézt i další, avšak méně obvyklé způsoby propojení (např. optické, infračervené nebo ultrazvukové spoje). Senzorické sítě mohou sloužit k propojení senzorů v automobilu, v domácnosti nebo v přírodním parku. Lze je využít i ke shromažďování informací z odběrných míst energií a nahradit tak pracovníka provádějícího odečet v bytech nebo alespoň zefektivnit jeho činnost tím, že nemusí vstupovat přímo do bytu, ale odečet provede pouze tak, že projde s přijímačem kolem domu. V rámci své práce jsem se zabýval možností uplatnění senzorických sítí při sběru dat v prostředí v němž se pohybují zasahující hasiči. Jejich pohybem senzory neustále mění vzájemnou polohu.

2.1 Historie senzorických sítí

Oblast senzorických sítí prochází v současnosti rychlým rozvojem. Jejich vývojem se zabývá velké množství akademiků stejně jako významné průmyslové podniky. O jejich využití se dnes uvažuje na mnoha místech, dříve nemyslitelných.

Typický prvek senzorické sítě se skládá z několika různých součástí. Jejich rozvoj je podmíněn především vývojem v těchto oblastech:

- *mikrokontroléry*
- *zdroje energie*
- *čidla*
- *komunikační jednotky.*

Je všeobecně známo, že oblast výpočetní techniky jako celek prodělala během posledního půlstoletí značné změny. Od reléových elektromechanických

počítačů (nejznámější z nich je asi americký počítač *MARK I*), které vznikaly ještě v období před 2. světovou válkou se vývoj posunul k těm počítačům, které již byly převážně elektronkové. Jasnou nevýhodou všech těchto počítačů byla jejich velikost stejně jako energetická náročnost způsobená použitím elektromechanických prvků a elektronek. Například první československý počítač SAPO[1] zabíral sám několik rozlehlých místností - asi $100m^2$[7]. Následoval příchod tranzistorů a postupná miniaturizace a integrace elektronických prvků. Přímý vliv na senzorické sítě měly až počítače 4. generace s *mikroprocesorem* vyrobeným technologií VLSI[2][3], z něhož připojením periferních obvodů vznikly *mikrokontroléry*, které jsou tradičně ústředním prvkem uzlu senzorické sítě.

Další významnou součástí senzoru v síti je zdroj energie. Nejběžnějším zdrojem energie jsou baterie, a to jak dobíjecí, tak primární články. Ty jsou slabinou v aplikacích vyžadujících bezúdržbovou činnost senzorické sítě po dobu několika let - zejména z důvodu velikosti a kapacity. V současnosti se začíná prosazovat také napájení alternativními metodami, například solárními články, využitím teplotních rozdílů nebo vibrací[8].

Nezbytnou součástí senzoru je čidlo. V dnešní době se hojně využívají čidla na bázi *kapacity*, *fotorezistoru*, *termorezistoru*, *fotodiody* nebo čidla *MEMS*[3].

Další částí senzoru, která umožňuje jeho zapojení do senzorické sítě, je komunikační jednotka. V dnešní době se k propojení senzorů používají zejména rádiové vlny. Používají se hlavně *ISM*[4] pásma 433 MHz a 868 MHz v Evropě, 915MHz v USA a 2.4 GHz. V této oblasti dominují zejména výrobky firmy *Texas Instruments*. Sledovanými parametry jsou v první řadě citlivost při příjmu signálu, výkon vysílače a samozřejmě příkon.

Současně s vývojem hardware pro senzorické sítě se rozvíjí komunikační protokoly. Pro bezdrátové senzorické sítě je organizací IEEE[5] určeno doporučení 802.15.4 pro sítě s nízkým datovým tokem. Tento standard z roku 2003, nad nímž jsou postaveny další standardy, definuje nejnižší vrstvy protokolového zásobníku. Jedním z nich je také standard s názvem *ZigBee*. Podobné sítě vznikaly už na konci 90. let, ale až po zahájení prací na standardu 802.15.4 byla v roce 2002 založena *ZigBee Alliance*. Jedná se o otevřené sdružení, v současnosti více než 200 subjektů, které se věnuje vývoji standardu

[1] *SAPO - SAmočinný POčítač*

[2] *VLSI - Very-Large-Scale Integration* - velmi vysoká integrace

[3] *MEMS - Micro-Electro-Mechanical System* jsou mikro elektro-mechanické prvky vytvořené podobným způsobem jako integrované obvody

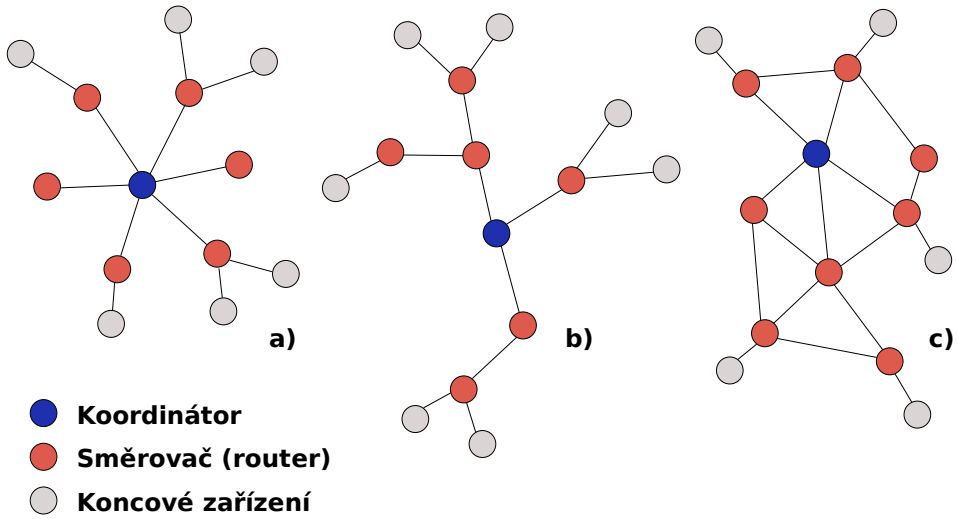
[4] Pásma *ISM - Industrial, Scientific, Medical* jsou volná pásma pro použití v průmyslu, vědě a medicíně

[5] *IEEE - Institute of Electrical and Electronics Engineers*

pro bezdrátové sítě s nízkým datovým tokem [2]. Jeho první verze byla představena v roce 2004, následovala revize v roce 2006. Zatím poslední verze standardu ZigBee byla uvedena roku 2007. Přinesla jeho rozdělení na 2 větve - ta jednodušší se nazývá *ZigBee 2007* nebo jen *ZigBee* a rozšířená verze nese název *ZigBee PRO*. Více viz 2.7 a 3.

Mezníkem v oblasti vývoje senzorických sítí byl projekt *SmartDust* agentury *DARPA*[6] realizovaný mezi lety 1997 a 2001. Jeho cílem bylo vytvořit *mote*, neboli uzel senzorické sítě, velikosti zrnka rýže použitelný pro vojenské účely [18].

2.2 Topologie bezdrátových senzorických sítí



Obrázek 2.1: Síťové topologie: a) hvězdicová topologie b) stromová topologie c) síťová topologie.

Nejtypičtější struktury bezdrátových senzorických sítí jsou na obrázku 2.1. Pro hvězdicovou i stromovou topologii, je typické výlučné postavení jednoho bodu sítě - koordinátoru, bez kterého není síť schopna vůbec fungovat, nebo je její funkčnost značně omezena. Naproti tomu *síťová topologie*[7] potře-

[6] *DARPA - Defense Advanced Research Projects Agency* je americká agentura zodpovědná za vývoj obranných technologií

[7] V češtině se z důvodu neustálené terminologie *síťová topologie* označuje ještě několika dalšími termíny, například termínem *topologie se smyčkami* nebo *volná topologie*. V angličtině se tato topologie nazývá *Mesh topology*

buje koordinátor pouze pro start sítě. Navíc se dokáže vyrovnat i s výpadkem několika uzlů, protože obsahuje redundantní spoje, které umožňují propojení funkčních uzlů i při přerušení některých cest.

Na obrázku 2.1 je vidět několik typů zařízení. Jednak *koordinátor*, který zajišťuje start sítě a také běžně slouží jako monitorovací bod nebo jako koncentrátor dat. Koordinátor v síti obvykle zastává i úlohu *směrovače* - zařízení předávajícího pakety v síti tak, aby došly na určené místo. Poslední zařízení v síti jsou *koncová zařízení*, která mají za úkol zejména sběr dat, případně řídí periferie. Jak již bylo naznačeno, funkce lze kombinovat. Koordinátor tak může sloužit jako směrovač a koncové zařízení, směrovač pak může převzít i funkci koncového zařízení.

2.3 Spotřeba prvku senzorické sítě

Pro senzorické sítě je otázka spotřeby elektrické energie naprosto zásadní. Pokud budeme například chtít čip nepřetržitě provozovat celý rok bez výměny baterií, musíme provést následující výpočet. Určíme optimální čas - tzn. jak dlouho může zařízení být v pracovním režimu [1].

Průměrný odebíraný proud je dán nejméně dvěma složkami (při předpokladu používání pracovního - pr - a úsporného - ur - režimu):

$$I_p = T_{pr} \cdot I_{pr} + T_{ur} \cdot I_{ur} = T_{pr} \cdot I_{pr} + (1 - T_{pr}) \cdot I_{ur}, \tag{2.1}$$

kde T_{pr} je podíl doby, kdy je zařízení v aktivním režimu, I_{pr} je proud, který zařízení v tomto režimu odebírá, T_{ur} je podíl doby, kdy je zařízení v úsporném režimu a I_{ur} je příslušný proud. Pro návrh senzorické sítě je zajímavá zejména hodnota T_{pr}, tedy podíl času, který může zařízení strávit v aktivním režimu, proto jí z rovnice vyjádříme:

$$T_{pr} = \frac{I_p - I_{ur}}{I_{pr} - I_{ur}} \tag{2.2}$$

Za předpokladu napájení čipu ze dvou 1,5 V baterií AAA s kapacitou 750 mAh zapojených v sérii je průměrný roční proud, který jsou baterie schopny dodávat:

$$I_p = \frac{750mAh}{8760h} = 86\mu A$$

Průměrný výkon tedy je:

$$P_p = 3V \cdot 86\mu A = 258\mu W$$

2.4 Specifika napájení

U senzorických sítí se zpravidla snažíme omezit příkon hardwarových modulů na minimum. Vzhledem k povaze senzorických sítí může při probouzení ze spánku dojít k většímu odběru proudu než při samotném užitečném provozu, který může být velmi krátký [1].

2.5 Významní výrobci hardware pro senzorické sítě

2.5.1 Atmel Corporation

Jedná se o jednoho z největších světových výrobců mikrokontrolérů. Mimo jiné se věnuje i výrobě hardware určeného pro bezdrátové senzorické sítě. Kromě vysílačů/přijímačů firma vyrábí čipy osazené mikrokontrolérem s architekturou *AVR*, k nimž poskytuje i softwarovou podporu včetně ZigBee zásobníku.

2.5.2 Digi International

Firma byla založena v roce 1985 v Minnesotě a zabývá se nejen bezdrátovými sítěmi. V současné době firma vyrábí několik typů zařízení pro různé protokoly, včetně vlastního proprietárního. Zařízení označená *XBee® ZB* jsou určena protokolu ZigBee. Digi International vyrábí nejen hardware, ale dodává v podstatě již hotová řešení.

2.5.3 Freescale Semiconductor Inc

Freescale semiconductor se mimo jiné věnuje i výrobě prvků pro bezdrátové senzorické sítě. Současné produkty pro ně jsou označeny *MC132xx*. V méně výkonných čipech jsou použity 8-bitové mikrokontroléry rodiny *HCS08*, ve výkonnějších *ARM7*. Pro své výrobky nabízí proprietární softwarová řešení v podobě objektového kódu i řešení s otevřeným zdrojovým kódem. Firma poskytuje také protokolový zásobník ZigBee *BeeStack*^{TM}.

2.5.4 Infineon Technologies AG

Také německá společnost *Infineon Technologies AG* dodává na trh různé druhy senzorů. Produktová řada určená pro senzorické sítě nese označení *SmartLEWIS*^{TM}. Firma používá mikrokontroléry s architekturou *i8051*.

2.5.5 Texas Instruments

Texas Instruments je v oblasti senzorických sítí známá především jako výrobce komunikačních čipů, které používají i další výrobci hardware. V současné době má firma také širokou nabídku dalších produktů vhodných pro senzorické sítě. Řada produktů určená pro bezdrátové senzorické sítě nese označení *CCxxxx*. V mikrokontrolérech firma využívá modifikaci architektury *i8051* a novější *MSP430*.

Srovnání zařízení firmy Texas Instruments pro nejběžnější ISM pásma

V následující tabulce jsou porovnány některé parametry (přenosová rychlost viz [4]) zvolených referenčních zařízení. Je použita terminologie firmy Texas Instruments[15].

Bezlicenční pásmo	2,4GHz	sub 1 GHz	
		868 MHz	915 MHz
Použití	globálně	Evropa	USA
Přenosová rychlost (povinné) [kb/s]	250	20	40
Dosažitelná rychlost (volitelné) [kb/s]	250	250	250
Softwarové vybavení	Z-Stack	SimpliciTI	
Odběr PM0 - RX mode [mA]	25	17	
Odběr PM0 - TX mode [mA]	34	20/21	
Odběr PM1 [μA]	105	-	
Odběr PM2 [μA]	1	0,5	
Odběr PM3 [μA]	0,4	0,3	

Tabulka 2.1: Srovnání parametrů zařízení TI SoC CC2530 (2,4 GHz) a TI SoC CC1110 (sub 1 GHz).

2.6 Standardy IEEE 802.15

Skupina 802.15 v rámci *Institute of Electrical and Electronics Engineers* se věnuje tvorbě standardů pro *WPAN*[8] [5] (některé lze použít i pro tvorbu *WLAN*[9]). Ty dovolují propojit zařízení jako jsou přenosné počítače, telefony

[8]Zkratka *WPAN* označuje *Wireless Personal Area Network*
[9]Zkratka *WLAN* označuje *Wireless Local Area Network*

a ostatní spotřební elektronika a tím umožní jejich vzájemnou spolupráci. Standardy rodiny 802.15 se používají i pro vytváření senzorických sítí nebo sítí typu *BAN*[10] v medicíně, což je vlastně speciální podmnožina senzorických sítí. V rámci skupiny existují pracovní podskupiny, které vytvářejí konkrétní standardy pro WPAN. V následujícím textu jsou zmíněny nejvýznamnější z nich.

2.6.1 802.15.1

Standard je založen na specifikaci Bluetooth^{TM}v1.1. Frekventovaně se využívá k propojení přenosných zařízení. Definuje zejména MAC rozhraní k fyzické vrstvě.

2.6.2 802.15.3

Standard 802.15.3 je určen pro vysokorychlostní WPAN sítě. Definuje fyzickou a linkovou vrstvu ISO/OSI[11] modelu.

2.6.3 802.15.4

Standard 802.15.4 je určen pro sítě s nízkým tokem dat - je tedy vhodný k použití v senzorických sítích. Definuje fyzickou a linkovou vrstvu ISO/OSI modelu pro globální bezlicenční pásmo - 2,4 GHz a americké, respektive evropské bezlicenční pásmo - 915 MHz, respektive 868 MHz. Definované rychlosti jsou 250 kb/s (2.4 GHz, 16 kanálů), 40 kb/s (915 MHz, 10 kanálů) a 20 kb/s (868 MHz, 1 kanál). Přenosová vzdálenost se pohybuje v rozmezí 10 až 100 metrů v závislosti na výstupním výkonu a prostředí.

2.6.4 802.15.6

Skupina vytváří standard optimalizovaný pro BAN.

2.6.5 Protokolové zásobníky založené na 802.15.4

Existuje celá řada protokolů založených na 802.15.4. Texas Instruments poskytuje implementace několika takových protokolů, které jsou určeny pro zařízení vyráběná touto firmou.

[10] Zkratka *BAN* označuje *Body Area Network*

[11] *ISO/OSI* je referenční vrstvový model pro realizaci protokolů počítačových sítí

Z-Stack

Jedná se o kompletní implementaci protokolu ZigBee 2007 (ZigBee i ZigBee PRO) - odtud i *Z* v názvu tohoto software. Je plně kompatibilní se zařízeními MSP430^{TM} spolu s CC2520 a s SoC CC2530. Podporuje profily (viz 3.1.3) *Smart Energy* a *Home Automation* definované ZigBee Aliancí. Existuje k němu velké množství příkladů a obsáhlé manuály. Je distribuován včetně zdrojových kódů. Doporučené je použití v těchto oblastech:

- automatizace
- řízení spotřeby a výroby energie
- ovládání systémů a zabezpečení v domácnosti
- senzorické sítě
- měřící systémy.

RemoTI

Jedná se o kompletní implementaci protokolu ZigBee RF4CE (určený pro dálková ovládání - *remote control*). Podporuje zařízení CC2530, CC2531 a CC2533. Použitý protokol ZigBee RF4CE ho předurčuje k použití v těchto zařízeních:

- dálkové ovládání
- set-top boxy, televize, apod.

SimpliciTI

Implementuje protokol firmy Texas Instruments. Je určen k realizaci menších sítí ve volných frekvenčních pásmech. Podporuje například zařízení z rodiny MSP430^{TM}, ale také SoC CC11xx a CC25xx). Je distribuován jako zdrojový kód. Název připomíná anglické slovo *simplicity* (jednoduchost). Výrobce doporučuje použití zejména v těchto oblastech:

- senzory a zabezpečení, ovládání osvětlení
- detektory kouře, CO_2 a nebezpečných látek
- automatické měření spotřeby (vodoměr, plynoměr, elektroměr).

TIMAC

Název je zkratka z *Texas Instruments MAC*. Je to zásobník MAC, který je určen k realizaci „point to point“ nebo „point to multipoint“ aplikací. Je poskytován zdarma jako objektový kód.

2.7 Standardy ZigBee

Standardy ZigBee jsou nadstavbou specifikace 802.15.4. Definují vyšší vrstvy zásobníkového modelu ISO/OSI. Jsou určeny primárně k tvorbě sítí typu WPAN, mezi než lze zahrnout i sítě senzorické.

2.7.1 ZigBee RF4CE

Standard ZigBee RF4CE má sloužit k propojení zařízení s dálkovým ovládáním, připojení jednoduchých vstupních zařízení, apod.

2.7.2 ZigBee

Standard ZigBee umožňuje vybudovat síť čítající až několik tisíc zařízení. Současná verze je *ZigBee 2007*. Adresní prostor poskytuje zhruba 65 000 adres. Existují dvě navzájem kompatibilní varianty[12] standardu *ZigBee 2007*, které se označují *ZigBee* a *ZigBee PRO*. Tyto verze se liší způsobem přidělování adres a směrováním, viz 3.2.

ZigBee

Jde o základní variantu protokolu, která je určena pro menší sítě maximálně se stovkami zařízení. Je méně flexibilní a adresování je založeno na stromové topologii.

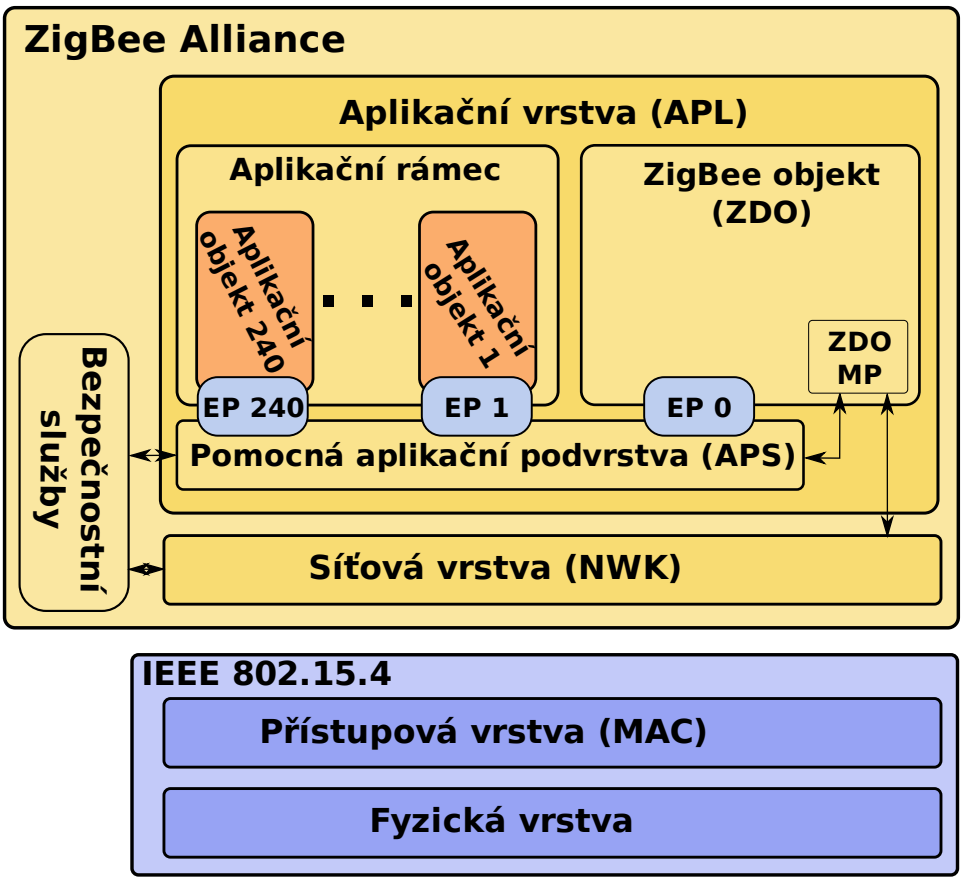
ZigBee PRO

Jedná se o rozšíření předchozí varianty, jehož použití je doporučováno. V této variantě lze provozovat sítě řádově o tisících bodech a relativně velkém vytížení sítě. Oproti předchozí variantě například poskytuje již ve výchozím nastavení fragmentaci dat a mnoho dalších vlastností je optimalizováno [21]. Významný je také jiný způsob přidělování síťové adresy, viz 3.2.

[12] V originále *Feature Sets*

3 Zásobník ZigBee

3.1 Architektura zásobníku ZigBee



Obrázek 3.1: Zásobník ZigBee. Znázorněny jsou pouze některé důležité části.

Zásobník ZigBee je zjednodušeně vyobrazen na obrázku 3.1. Obě vrstvy - fyzická a přístupová jsou definovány standardem IEEE 802.15.4. Vyšší vrstvy zásobníku definovala *ZigBee Alliance*.

Na schématu 3.1 jsou v aplikační vrstvě vyobrazeny *servisní přístupové body* označené *EP*, tzv. *brány*[1]. Ty lze přirovnat k portům v zásobníku TCP/IP. Mohou sloužit k rozlišení a předání příchozích zpráv do správné aplikace.

[1] V originále *endpoint*

Následující text stručně popisuje jednotlivé části zásobníku ZigBee.

3.1.1 Síťová vrstva (NWK)

Síťová vrstva představuje rozhraní mezi *aplikační* a *přístupovou vrstvou*. Tato vrstva umožňuje spuštění a konfiguraci sítě, připojení k síti, směrování, adresování či zabezpečení přenášených dat. Jejím hlavním úkolem je zajistit přenesení aplikačních dat přes síť z jednoho zařízení na jiné. Délka síťové adresy je 16 bitů. Síťová vrstva obsahuje *NIB*[2], v níž jsou uloženy atributy důležité pro správu síťové vrstvy, mimo jiné také přiřazení známých síťových a IEEE adres [20].

3.1.2 Pomocná aplikační podvrstva (APS)

APS [3] tvoří rozhraní mezi aplikacemi (včetně *ZDO*) a síťovou vrstvou. Vyšší vrstvě poskytuje datové služby, jejichž součástí je fragmentace nebo zaručené doručování a služby konfigurační - ty obsahují správu skupin nebo bezpečnost [20].

3.1.3 Aplikační rámec

Aplikační rámec[4] je prostředí, v němž jsou definovány samotné aplikace. Každá z nich je identifikována číslem *brány*. Jelikož jsou některá čísla vyhrazena, může byt definováno maximálně 240 aplikací (1 - 240).

Profil aplikace

Profil aplikace[5] je souhrn vlastností zařízení, které v společně tvoří fungující aplikaci. Specifikace ZigBee uvádí jako příklad termostat na jednom uzlu sítě, který komunikuje s kotlem v jiném uzlu. Dohromady pak tyto dvě zařízení tvoří *profil vytápění* [20]. Profil aplikace je určen 16-ti bitovým identifikátorem [20].

Cluster

Termínem *Cluster* se označuje zpráva aplikační úrovně, která může sloužit jako nosič parametrů. Komunikační protokol na aplikační vrstvě je tedy tvo-

[2]Zkratka *NIB* označuje *Network Information Base*
[3]Zkratka *APS* označuje *Application Support Sub-Layer*
[4]V originále *Application Framework*
[5]V originále *Application Profile*

řen pomocí clusterů. Příkladem může být *ZDP*[6], jehož příkazy jsou umístěny v clusterech. Všechny zprávy jsou tedy clustery, kdy každý je označen 16-ti bitovým identifikátorem, který je unikátní v rámci *profilu aplikace* [20].

3.1.4 Objekt zařízení ZigBee a profil zařízení ZigBee

Výlučné postavení má v aplikační vrstvě *objekt zařízení ZigBee* (*ZDO*)[7]. Ten je zodpovědný za to, že zařízení v síti zastává roli koordinátoru, routeru nebo koncového zařízení [20]. Dále má na starosti některé další detaily související se správou sítě. Jednoduše řečeno: ZDO je aplikace na *bráně* 0 [2], která má na starosti správu sítě a poskytuje rozhraní ke speciálnímu *profilu aplikace* zvanému *ZDP* (s číslem profilu 0x0000). Na rozdíl od ostatních aplikací interaguje nejen s vrstvou APS, ale také s vrstvou NWK.

3.2 Adresování v ZigBee

ZigBee používá dva typy adres - dlouhé 64-ti bitové adresy[8], které jsou určitému zařízení jednorázově přiděleny a 16-ti bitové síťové adresy, které jsou přidělovány dynamicky během připojování zařízení do sítě [11].

3.2.1 Přidělení síťové adresy v ZigBee

Základním adresním schématem ZigBee je „stromové adresování“[9]. Sice snižuje zatížení sítě, ale na druhou stranu vyžaduje přesné a potenciálně omezující nastavení. Nezbytné je několik předkonfigurovaných parametrů:

`MAX_DEPTH` - jedná se o maximální hloubku sítě. Koordinátor má hloubku 1 a sousední uzly směrem od koordinátoru mají vždy hloubku o jednu větší.

`MAX_CHILDREN` - udává maximální počet potomků[10], které může koordinátor nebo router obsluhovat.

`MAX_ROUTERS` - určuje kolik potomků (množství potomků je vyjádřeno parametrem `MAX_CHILDREN`) může být typu router. Zbytek jsou koncová zařízení.

[6] Zkratka *ZDP* označuje *ZigBee Device Profile*
[7] V originále *ZigBee Device Object*
[8] Jedná se o jedinečnou MAC adresu, která jednoznačně identifikuje dané zařízení
[9] V originále *Tree Addressing*
[10] Pojem *potomek* vyjadřuje podřazené zařízení, *rodič* nadřazené zařízení

3.2.2 Přidělení síťové adresy v ZigBee PRO

ZigBee PRO oproti předchozímu způsobu používá metodu náhodného přidělení adresy. Takový způsob sice zvyšuje zatížení sítě při připojení nového zařízení ale zároveň zvyšuje flexibilitu sítě. Pro nově připojivší se zařízení do sítě je jeho rodičem náhodně vygenerována síťová adresa. Toto nové zařízení následně vyšle oznámení o připojení do sítě a všechna ostatní zařízení ověří zda nenastal konflikt adres. V případě konfliktu všechna zařízení, jichž se konflikt týká pro sebe náhodně vygenerují jinou adresu. (Tohoto procesu se neúčastní koncová zařízení - za potomky, kteří jsou koncovými zařízeními jednají jejich rodiče)

3.3 Směrování v ZigBee

V ZigBee se používá směrování *AODV*[11][17], které podporuje jak *unicast* - tedy směrování do konkrétního uzlu, tak i *multicast* - skupinové doručování. Cesty se vytvářejí vždy až ve chvíli kdy jsou skutečně potřebné. V případě použití stromového adresování je cesta pro unicast jednoznačně dána. Následující text popisuje adresování v *ZigBee PRO*.

Vytváření nové cesty je v případě unicastu vyvoláno potřebou odeslání dat z koncového zařízení nebo přímo z některého routeru. Pokud router nezná cestu k požadovanému zařízení, vyšle *broadcast* zprávu *RREQ*[12], kterou zachytí okolní routery. Pokud router, který obdržel *RREQ*, zná cestu k cíli, odpoví zprávou *RREP*[13], prostřednictvím které oznámí odesílateli, že cesta přes něj k cíli existuje. Pokud cestu nezná, spustí stejný mechanismus jako předcházející uzel, pomocí něhož se snaží nalézt cestu k cíli. Když je nalezeno více cest, stává se jejich cena hlavním kritériem při rozhodování o tom, která z nich se použije, viz 3.3.1.

V případě multicastu se vytváří doručovací stromy. Uzel, který se připojuje do multicastové skupiny se snaží nalézt nejkratší cestu k některému ze současných členů skupiny. Pro to je použit velmi podobný mechanismus jako v předcházejícím případě. Jakmile některý z členů skupiny obdrží *RREQ*, který obsahuje adresu skupiny a příznak žádosti připojení do skupiny, odpoví a zpráva *RREP* je pozpátku doručena uzlu, který žádal o připojení do skupiny. Uzel může dostat více odpovědí, a proto, aby nevznikaly redundance, musí po uplynutí timeoutu potvrdit pouze jednu z cest, kterou vybere opět na základě její ceny.

[11] *AODV* neboli *Ad hoc On-Demand Distance Vector Routing* je směrovací protokol pro bezdrátové sítě

[12] *RREQ* neboli *route request*

[13] *RREP* neboli *route reply*

Směrování se účastní výhradně routery, případně koordinátor. Koncová zařízení se bez schopnosti směrování obejdou, protože vždy patří k rodiči, který tuto schopnost mít musí a pro k němu náležející koncová zařízení funkci směrování zprostředkovává. Pro síť je tedy nezbytný minimálně jeden uzel se schopností směrování.

Více o směrování se lze dočíst zejména v [9] a [20].

3.3.1 Cesty v ZigBee

Dokument [20] popisuje také cenu cesty, která se určí jako součet cen všech spojení, ze kterých je cesta složená, tedy:

$$C\{P\} = \sum_{i=0}^{L-1} C\{l_i\}, \tag{3.1}$$

kde $C\{P\}$ je cena cesty, L označuje její délku a $C\{l\}$ cenu spoje. $C\{l\}$ je pak definována takto:

$$C\{l\} = \begin{cases} 7 \\ min(7, round(\frac{1}{p_l^4})) \end{cases}, \tag{3.2}$$

kde p_l určuje pravděpodobnost bezchybného doručení, funkce $round()$ je zaokrouhlení na celá čísla. Z uvedeného vzorce vyplývá, že specifikace umožňuje zvolit jak konstantní ohodnocení spojení tak jeho adaptivní výpočet pomocí spolehlivosti spoje (p_l). Přitom konkrétní postup určení spolehlivosti spoje není také normou přesně definován.

3.3.2 Směrovací tabulka

Obsahují ji pouze koordinátory a routery. V tabulce je umístěna hlavně cílová adresa a další adresa na cestě k cíli. V případě použití *stromového adresování* tabulka obsahuje místo adres koncových zařízení adresy jejich rodičů. Podrobnosti obsahuje dokument [20].

3.4 Zabezpečení přenosu

K odhalení vzniklých chyb se v ZigBee používá cyklický kód. Je použit polynom ve tvaru:

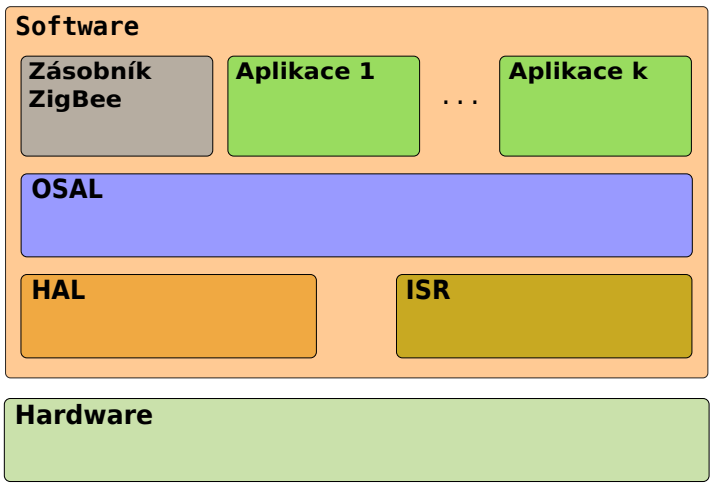
$$P(x) = x^{16} + x^{12} + x^5 + 1$$

Vysílá se pomocí technologie DSSS[14], kdy jsou jednotlivé bity nahrazeny jejich větším počtem. Přenášená data je možné symetricky šifrovat pomocí 128-bitového AES klíče [6].

3.4.1 Rušení v ZigBee

ZigBee dokáže vyřešit problém rušení okolními sítěmi automatickou, centrálně řízenou změnou frekvence [16]. Frekvence 2,4 GHz je velmi hustě využívána (např. od IEEE 802.11), rušení lze jednoduše omezit i volbou kanálů - doporučuje se vyhnout kanálům 1, 6 a 11 [19].

3.5 ZigBee Z-Stack



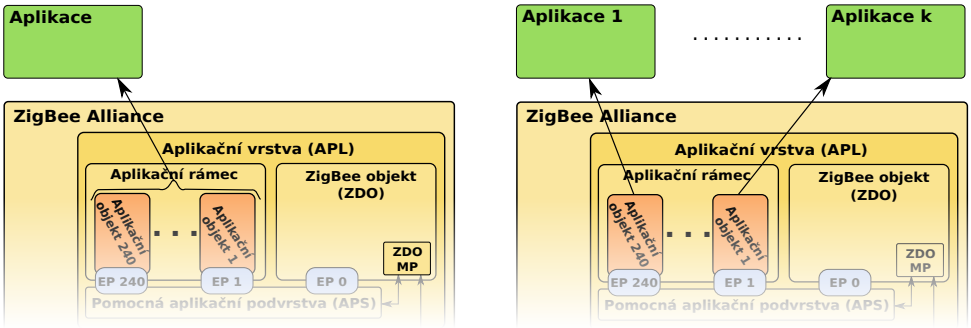
Obrázek 3.2: Hierarchické pojetí jednotlivých částí software umožňuje využití stejného kódu na více hardwarových platformách s menším programátorským úsilím, protože stačí upravit jen nižší vrstvy software.

Z-Stack je implementace zásobníku ZigBee od Texas Instruments. Kromě samotného zásobníku ZigBee, který byl popsán výše, zahrnuje distribuce firmy i jednoduchý operační systém - *OSAL*. Ten implementuje *koopera-*

[14] *DSSS - Direct Sequence Spread Spectrum* je metoda, která zvětšuje redundanci přenášených dat

tivní round-robin[15] algoritmus [12], který spolu s vrstvami *HAL*[16] a *ISR*[17] poskytuje aplikacím základní služby. Hierarchii softwarových vrstev a jejich návaznost na hardware ukazuje obrázek 3.2.

Uživatelská aplikace je běžnou úlohou OSALu. Spolupráce uživatelské aplikace se ZigBee zásobníkem je možná dvěma způsoby - viz obrázek 3.3.



Obrázek 3.3: Zásobník ZigBee a jeho návaznost na uživatelské aplikace. Vlevo varianta s jedinou uživatelskou úlohou pro všechny *aplikační objekty*, vpravo více OSAL úloh přiřazených k *aplikačním objektům*.

Vytvoření jediné aplikace, která obsluhuje všechny *aplikační objekty* je jednodušší, ale v závislosti na její složitosti toto vede k vyšší režii (identifikace události probíhá jak na úrovni ZigBee stacku tak znovu na úrovni aplikace).

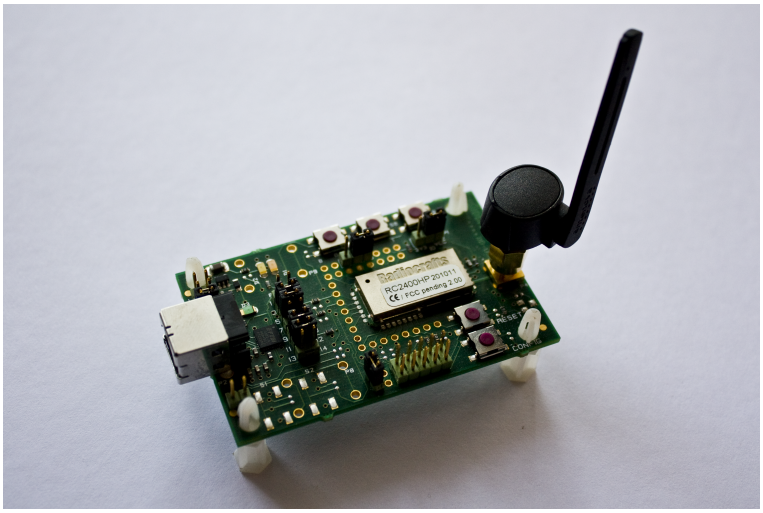
[15] *Kooperativní round-robin* je algoritmus pro střídání úloh, který funguje na principu FIFO, vyžadující, aby běžící úloha sama předala řízení operačnímu systému

[16] *HAL* neboli *Hardware Abstraction Layer* je vrstva sloužící aplikacím pro zjednodušení přístupu k hardware

[17] *ISR* neboli *Interrupt Service Routine* - obsluha přerušení

4 Použitý hardware

Pro realizaci praktické části práce jsem měl k dispozici modul *RC2400HP* osazený na desce plošných spojů, která je na obrázku 4.1. Modul se skládá z *SoC CC2530* a obvodu pro rozšíření dosahu *CC2591* firmy *Texas Instruments*.



Obrázek 4.1: Vývojová deska *RC2400HP DB* firmy Radiocrafts AS použitá při realizaci praktické části práce.

4.1 TI CC2530

CC2530 je zařízení typu *systém na čipu*, které je určeno přímo pro sítě ZigBee. Základem je vylepšený mikrokontrolér *i8051*. Vylepšení spočívá především v množství dostupné paměti, kde je na čipu je dostupných 8 kB RAM. Typ osazený v *RC2400HP* má také největší dodávanou velikost flash paměti, konkrétně 256 kB. Dále je na čipu integrován vysílač/přijímač s maximálním výstupním výkonem 4,5 dBm a citlivostí -97 dBm, jenž je schopen komunikovat až rychlostí 250 kBaudů. Více v [13], [14] a [15].

4.1.1 Úsporné režimy zařízení CC2530

CC2530 PM0

Pracovní režim - hodinový oscilátor (32MHz) zapnutý, regulace napětí zapnuta. Výrobce udává dobu potřebnou pro přechod do režimu RX, respektive TX, 192 μs.

CC2530 PM1

Tento režim se nepoužívá - zapnutý časovač (32.768 kHz), regulace napětí je zapnuta.

CC2530 PM2

Spánek s časovačem - zapnutý časovač (32.768 kHz), regulace napětí vypnuta. Maximální doba spánku - 510 s (8,5 minuty) - je dána velikostí registru. Výrobce udává, že doba potřebná k přechodu do režimu PM0 je 120 μs.

CC2530 PM3

Hluboký spánek - všechny oscilátory i regulace napětí vypnuta. Probuzení je možné pouze pomocí externího přerušení. Doba potřebná k přechodu čipu do režimu PM0 udávaná výrobcem je 120 μs.

4.2 TI CC2591

Je vstupně - výstupní obvod pro rozšíření dosahu zařízení typu CC2530. Po jeho aktivaci se výstupní výkon RC2400HP zvýší až na 20 dBm a citlivost na -99 dBm.

4.3 Spotřeba RC2400HP

V rámci hodnocení použitého hardware jsem provedl i úvahu o možnosti napájení zařízení *RC2400HP* z baterií. Výrobcem udávaná spotřeba zařízení *RC2400HP*, jehož hlavní součástí je CC2530, může špičkově, při vysílání dosáhnou až 218 mA. Pokud budeme využívat výstupní výkon pouze 4 dBm, je spotřeba 34 mA (toho lze dosáhnout i na modelu *RC2400*). Tuto hodnotu můžeme použít jako horní odhad příkonu v pracovním režimu, protože příkon v režimu přijímače je nižší, konkrétně $I_{pr} = 28\ mA$ [10]. Udávaný příkon

v úsporném režimu je $I_{ur} = 1\ \mu A$. Maximální čas, který může zařízení strávit v pracovním režimu, získáme dosazením do vzorce 2.2:

$$T_{pr} = \frac{86 \cdot 10^{-6} - 1 \cdot 10^{-6}}{34 \cdot 10^{-3} - 1 \cdot 10^{-6}} \approx 0,0025,$$

což odpovídá 0,25 % celkového času. Pokud provedeme přepočet, zjistíme následující: aby mohlo být zařízení napájeno z baterií o kapacitě 750 mAh po celý rok, může být v pracovním režimu maximálně po dobu 150 ms každou minutu, což je 9 sekund každou hodinu.

V případě, že vezmeme v úvahu plánované využití čipu jako prvku senzorické sítě na obleku hasiče, lze velmi nepříznivý časový odhad vylepšit. Například při průměrném provozu senzorické sítě 1 hodinu denně je možno vyčerpat vypočtený energetický limit na celý den během této jedné hodiny. Místo 9 sekund dostáváme 24-násobek, tedy celých 216 sekund, které odpovídají více než 3 a půl minutám čistého pracovního času, což je hodnota o poznání příznivější. Zároveň je třeba podotknout, že se, díky shora odhadnuté spotřebě, jedná o dolní odhad doby, po kterou může být zařízení v pracovním režimu.

4.3.1 Orientační měření spotřeby RC240HP DB - multimetr

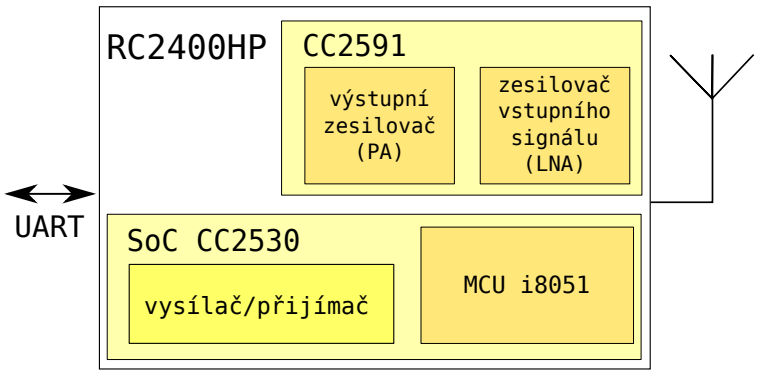
V rámci testování zařízení *RC2400HP* osazeného na desce plošných spojů jsem provedl orientační měření spotřeby zařízení pomocí multimetru. Výsledky měření obsahuje následující tabulka:

Režim RC2400HP	Odběr desky RC2400HP DB
PM0 (`while(1) NOP;`)	7,2 [mA]
PM1	-
PM2	3,9 [mA]
PM3	3,9 [mA]

Tabulka 4.1: Proud odebíraný deskou RC2400HP DB.

5 Aplikace pro RC2400HP

Součástí praktické části práce je aplikace, která běží přímo na modulech *RC2400HP*. Jak jsem již zmínil v předcházející části textu, realizovat takovouto aplikaci je možné více způsoby. Pokud se mluví o programování RC2400HP, jde v podstatě, jak je vidět z obrázku 5.1, o programování jeho hlavní části, tedy obvodu *CC2530*.



Obrázek 5.1: Blokové schéma obvodu RC2400HP.

Samozřejmě lze začít programovat bez použití jakékoli knihovny. Zvolení tohoto přístupu má však zjevné nevýhody. Zejména vyvstává nutnost naprogramovat celý protokolový zásobník včetně bezpečného doručování mezi uzly, přidělování síťové adresy a směrování. Jistěže je teoreticky možné aplikaci přesněji přizpůsobit zadání, avšak tím, že programujeme vše od nuly se pozornost přesouvá k věcem podpůrného charakteru.

Další variantou je využití některého z protokolových zásobníků dodávaných společností Texas Instruments, či volba některé z volně šiřitelných alternativ. Předností tohoto přístupu je zrychlení vývoje díky použití již vytvořeného prověřeného kódu a dále dobrá rozšiřitelnost takové aplikace, která vyžaduje, kvůli zavedení nové funkce, pouze dílčí úpravy.

Volně šiřitelné operační systémy - například *Contiki OS*, ovšem mají určité vývojové zpoždění. Tento operační systém, určený mimo jiné i pro senzorické sítě, má v současné době podporu pouze pro předchůdce čipu *CC2530*, tedy pro *CC2430*. Je nutné podotknout, že oba čipy jsou si velmi podobné. Na druhou stranu nesmí být přehlížen fakt, že se prozatím jedná o podporu pouze experimentální. Z hlediska efektivity vývoje se zdá nejlepší zvolit některý ze zásobníků, které dodává přímo výrobce hardwarových komponent. Výhodou je dobré přizpůsobení konkrétnímu hardware, dále poměrně velké množství

aktivních uživatelů, kteří je používají na stejných nebo velmi podobných zařízeních. Důsledkem je pak již zmíněné zkrácení doby vývoje. U společnosti významu Texas Instruments lze za přednost považovat i samotný fakt, že se jedná o garanta dalšího vývoje, zvláště, pokud jde o software určený pro její vlastní produkty.

Realizovaná aplikace musí plnit jednak účel demonstrační, ale hlavně musí zajistit možnost komunikace s vybraným uzlem v síti a jeho jednoznačnou identifikaci, aby mohla být následně použita jako základ pro další vývoj.

5.1 Vývojové prostředky

Pro procesory architektury *i8051* osazené v zařízeních firmy Texas Instruments, ale i dalších výrobců, existuje několik vývojových prostředků.

5.1.1 IAR Embeded Workbench

Toto kompletní vývojové prostředí včetně překladače a debuggeru patří ke komerčním produktům společnosti *IAR Systems*. Firma Texas Instruments jej doporučuje zejména pro vývoj aplikací pomocí zásobníku Z-Stack, který lze v současné době bez rozsáhlých úprav Z-Stacku realizovat výhradně jen v tomto software. Omezená verze tohoto vývojového prostředí, v níž však nelze pracovat se Z-Stackem, je k dispozici bezplatně. Se Z-Stackem je možno pracovat po zakoupení plné verze, nebo v rámci verze zkušební, jejíž užívání je omezeno na dobu jednoho měsíce. Protože se jedná o standardní, doporučené řešení, zvolil jsem ho jako hlavní vývojový prostředek při své práci na aplikaci pro zařízení RC2400HP. Konkrétně jsem používal *IAR EW Evaluation for 8051* ve verzi *7.60* a vývoj probíhal v operačním systému *Microsoft Windows XP*.

5.1.2 SmartRF Flash Programmer

Volně dostupný produkt Texas Instruments, který slouží, jak již název napovídá, k obsluze programátoru, tedy přímo k programování hardware pomocí PC.

5.1.3 Alternativní vývojové prostředky

Small Device C Compiler - SDCC

Jedná se o překladač jazyka C (nejen) pro procesory architektury *i8051* s otevřeným zdrojovým kódem. Lze jej integrovat do vývojového prostředí Eclipse.

Keil

Tento překladač jazyka C je komerčním produktem firmy ARM (dříve Keil), která poskytuje také vývojové prostředí *Keil µVision*.

5.2 Charakteristika realizované sítě

Jak vyplývá z textu předchozích kapitol i zamýšleného využití sítě, je typ topologie realizované sítě poměrně jednoznačný. Libovolné uspořádání s redundantními spoji, kterému nejlépe odpovídá *síťová topologie*, je ideálním řešením pro pohyblivé uzly.

Pokud vezmeme v úvahu, že uzly se mohou v rámci sítě různé přeskupovat, zjistíme, že ze tří možných typů zařízení mají v takové síti význam pouze typy dva - router a koordinátor. Koncové zařízení bez schopnosti směrování pozbývá významu, protože jediným spojovacím článkem částí sítě s jejím zbytkem se může potenciálně stát kterékoliv zařízení.

Z výše uvedených skutečností je jasné, že nejvhodnějším protokolem pro popsanou síť je *ZigBee PRO*. Tím pádem jsem si zvolil použití protokolového zásobníku ZigBee ve formě dodávané přímo Texas Instruments - použil jsem *Z-Stack 2.3.1*, jehož popisem jsem se zabýval v 3.5.

Pro tuto aplikaci jsem na každém zařízení použil pouze jednu *bránu*. Pokud totiž pomineme demonstrační funkci, plní realizovaná aplikace funkci pouze jedinou a tudíž není zapotřebí jí výrazněji rozčleňovat. Za výchozí bod návrhu posloužila vzorová aplikace *GenericApp* firmy Texas Instruments, dodávaná společně se Z-Stackem. Mnou vytvořená aplikace vznikla výraznou úpravou tohoto příkladu.

Následující odstavce popisují klíčové části programu vytvořeného pro zařízení RC2400HP. Detailní popis API je v programátorské dokumentaci.

5.3 Základy práce se Z-Stackem

Z určitého úhlu pohledu je rozsáhlost Z-Stacku nevýhodou. Seznámení se s jeho koncepcí a osvojení si způsobu práce s ním zůstává, navzdory rozsáhlým manuálům, dokumentaci a vzorovým aplikacím, poměrně časově náročné. Po překonání počátečních komplikací je však práce se Z-Stackem poměrně jednoduchá a především vývoj postupuje rychle kupředu.

Pro ilustraci práce se sítí uvedu krátký úsek kódu s komentáři. Podobný kód je součástí inicializační funkce realizovaného programu.

```
#define GENERICAPP_ENDPOINT           10 /* Číslo brány
                                          používaného aplikací */
...
afAddrType_t GenericApp_DstAddr; /* Cílová adresa dat */
aps_Group_t GenericApp_Group; /* Skupina, jíž bude zařízení
                                 členem */
...
/* Nastavení cílových adres pro zasílaná data */
#if defined(ZDO_COORDINATOR) /* koordinátor bude zasílat zprávy
                                celé skupině zařízení */
  GenericApp_DstAddr.addrMode = (afAddrMode_t)AddrGroup;
  GenericApp_DstAddr.endPoint = GENERICAPP_ENDPOINT;
  GenericApp_DstAddr.addr.shortAddr = GENERICAPP_GROUP_ID;
#else  /* ostatní zařízení budou data posílat koordinátoru */
  GenericApp_DstAddr.addrMode = (afAddrMode_t)Addr16Bit;
  GenericApp_DstAddr.endPoint = GENERICAPP_ENDPOINT;
  GenericApp_DstAddr.addr.shortAddr = 0x0000; /* síťová adresa
                                                 koordinátoru */
#endif
...
/* Přiřazení zařízení do skupiny */
GenericApp_Group.ID = GENERICAPP_GROUP_ID;
osal_memcpy(GenericApp_Group.name, "Skup1", 5);
aps_AddGroup(GENERICAPP_ENDPOINT, &GenericApp_Group);
```

Uvedený příklad ilustruje použití skupinového a unicast adresování. Další možnosti adresování jsou: použití 64-bitové adresy, broadcastu nebo neuvedení žádné adresy. Jestliže adresa není uvedena je určena z *tabulky spojení*, ovšem to muselo být dříve vytvořeno.

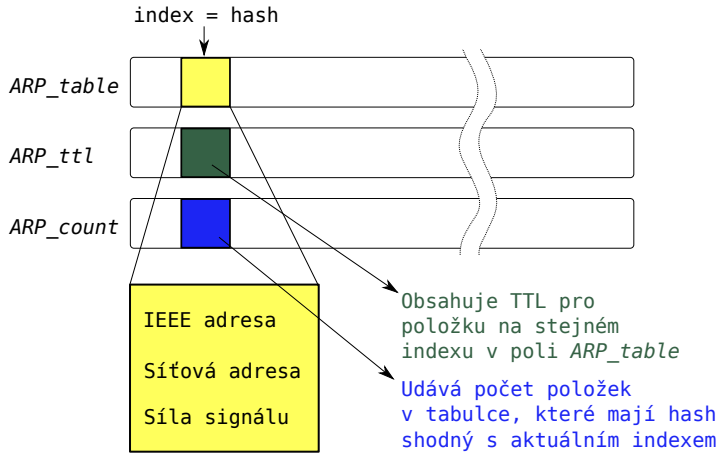
Pro síťovou komunikaci mezi zařízeními jsou použity obyčejné datové rámce, a to prostřednictvím API Z-Stacku. Podrobný popis všech typů rámců lze nalézt v dokumentu [20].

5.4 ARP tabulka

Každé zařízení je reprezentováno unikátní, pevně přidělenou 64-bitovou IEEE adresou a 16-ti bitovou síťovou adresou, která je přidělována dynamicky. Tato se může měnit s každým připojením do sítě.

Aby byla zajištěna jednoznačná identifikace každého zařízení, je třeba spojit síťovou adresu každého zařízení s pevně přiděleným identifikátorem.

Jako takový se samozřejmě nabízí IEEE adresa, která je pro každé zařízení neměnná a unikátní. Zároveň je vhodné shromáždit seznam zařízení, která jsou k síti připojena, protože v síti s pohyblivými uzly se může snadno stát, že se některý uzel dostane mimo dosah sítě, tudíž se stává nedostupným.



Obrázek 5.2: Implementace ARP tabulky včetně pomocných datových struktur.

Z těchto důvodů jsem v aplikační vrstvě na síťovém koordinátoru vytvořil datovou strukturu, kterou jsem nazval *ARP tabulkou*, jelikož se podobá klasické ARP tabulce v IP sítích. Implementace ARP tabulky se nachází v oddělených souborech (*arp.h* a *arp.c*), čímž je zajištěna jednoduchá modifikovatelnost. Maximální velikost tabulky je omezena na 2^{16} prvků a její aktuální velikost definuje konstanta *ARP_L*. Tabulka je tvořena třemi poli, přičemž první, nazvané *ARP_table* obsahuje 11-ti bytové položky, z nichž každá charakterizuje jeden síťový uzel. Položky obsahují síťovou a IEEE adresu uzlu, stejně jako sílu signálu, s níž byla obdržena poslední zpráva o daném zařízení, viz obrázek 5.2.

Následující dvě pole jsou servisního charakteru. První z nich, které se nazývá *ARP_ttl* má prvky o velikosti 1 byte. Definuje jak dlouho je daná položka v ARP tabulce považována za platnou aniž by byla aktualizována. Položky v tomto poli jsou periodicky dekrementovány a při příchodu informační zprávy z daného zařízení je příslušná položka aktualizována na výchozí hodnotu (konstanta *ARP_DEFAULT_TTL*). Tuto hodnotu lze upravit v konfiguračním souboru *arp.h* a společně s periodou časovače rozhoduje o době, která musí uplynout, aby došlo k vyřazení položky z tabulky. Položky s kladnou hodnotou TTL jsou považovány za *platné*, položky s nulovým TTL jsou sice *neplatné*, ale stále jsou přítomny v ARP tabulce. Položky se zápor-

nou hodnotou TTL jsou *neplatné* a mohou být přepsány. Prvek tohoto pole samozřejmě náleží k datům se shodným indexem v poli s daty.

Třetí pole - *ARP_count* - slouží pouze pro zefektivnění práce s ARP tabulkou přičemž obsahuje 2 bytové prvky. Zařízení jsou do tabulky umisťována na základě hashe vypočteného z posledních dvou bytů IEEE adresy pomocí funkce *modulo*. Pole obsahuje na příslušném indexu číslo charakterizující počet položek v tabulce, jejichž hash se rovná danému indexu v tomto poli. Hodnota prvku pole slouží za zastavovací podmínku při prohledávání ARP tabulky a tím urychluje práci s ní, protože ji není potřeba prohledávat celou.

Pro tato pole je v paměti místo vyhrazeno staticky a záleží pouze na programátorovi, aby zvolil dostatečné místo pomocí definice konstanty v hlavičkovém souboru. Statickou alokaci jsem zvolil z důvodu, že jde o bezpečnější a lépe odladitelný způsob, který je pro takovéto zařízení nejvhodnější. Programátor, který by mnou vytvořený software používal, by měl sám zvolit optimální velikost tabulky dle předpokládaného množství zařízení umístěných v síti.

Za předpokladu takového množství zařízení, že by paměť nedostačovala, je možné jednoduše tuto implementaci modifikovat a celou tabulku ukládat pouze na obslužném počítači. O realizaci této možnosti jsem na počátku uvažoval, avšak přiklonil jsem se k univerzálnějšímu řešení - umístění ARP tabulky do paměti čipu. Tento způsob umožňuje případné využití periferií (například tlačítka a LCD display) ve spolupráci s modulem RC2400HP místo obslužného PC.

Tabulku plní data, získávaná ze síťových zařízení tím, že koordinátor je všechna periodicky oslovuje. Na zprávu od koordinátoru zařízení odpoví zprávou, která obsahuje jeho síťovou i IEEE adresu. Tímto způsobem se data dostávají do koordinátoru, který tak zároveň ověřuje dostupnost jednotlivých zařízení. Každé zařízení v síti je díky tomu schopné zjistit, není-li do určité doby osloveno koordinátorem, zda nastal problém s komunikací. Pro oslovování zařízení se využívá multicastu (tzn. koordinátor nemusí každé zařízení oslovovat zvlášť). Pro tento účel by bylo možné použít broadcast, ovšem na rozdíl od něj se multicast uplatní i v případě, kdy by oslovování některých zařízení bylo nežádoucí. Navíc změna na multicast je velmi jednoduše uskutečnitelná bez výrazného zásahu do zdrojových kódů aplikace.

5.5 Komunikace s nadřazeným systémem

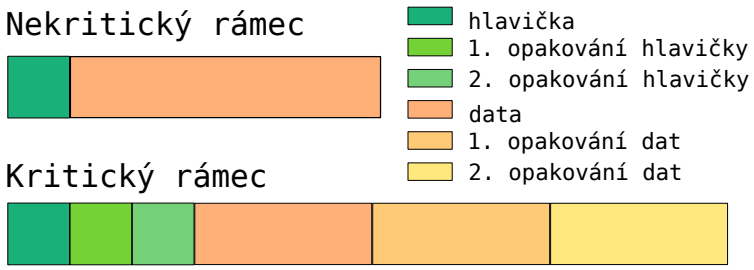
Pro komunikaci čipu s nadřazeným systémem lze použít rozhraní *UART*. Vývojová deska je dokonce vybavena převodníkem *UART na USB*. Ke komunikaci s nadřazeným systémem lze tedy použít USB kabelu, který desku

zároveň napájí. Protože Z-Stack obsahuje API pro obsluhu rozhraní UART, je jeho použití poměrně jednoduché.

Funkci pro komunikaci s nadřazeným systémem jsem z důvodu ladění realizoval jak na routerech tak na koordinátoru, i když praktický význam má pouze na koordinátoru. Na obou typech zařízení si lze pomocí znakově orientovaného protokolu vyžádat základní identifikaci zařízení. Přes UART zasílají oba typy zařízení chybová a informační hlášení. Hlavní význam má komunikace u koordinátoru, který předává nadřazenému systému informace obsažené v ARP tabulce. Nadřazenému systému je umožněno, prostřednictvím koordinátoru, zasílat příkazy jednotlivým zařízením v síti.

5.5.1 Komunikační protokol

Pro komunikaci pomocí rozhraní UART jsem vytvořil jednoduchý, znakově orientovaný protokol. Každý rámec tvoří hlavička a tělo. Jednotlivé rámce od sebe oddělují nulové znaky. Definovány jsou dva druhy rámců - kritický a nekritický, viz obrázek 5.3.



Obrázek 5.3: Kritický a nekritický rámec komunikačního protokolu.

Nekritické rámce

Oba druhy rámců mají velmi podobný charakter. Nekritické jsou však o něco jednodušší. Neobsahují v sobě totiž žádnou redundanci a v podstatě odpovídají výše uvedenému obecnému popisu. Jsou od sebe vzájemně odděleny nejméně jedním nulovým znakem. Tyto rámce by měly být používány v co největší míře, aby se ušetřila šířka pásma. Měly by však být užívány jen v případech, kdy nehrozí, že špatná interpretace nebo ztráta rámce by způsobila chybu. Hlavičky by měly mít takovou vzájemnou *Hammingovu vzdálenost*, aby byl rámec při jednoduché chybě vyhodnocen jako neplatný. Nekritické rámce používám v aplikaci výhradně pro doručování informací směrem do

nadřazeného počítače. Jde zejména o stavové informace, tj. ty o vzniklých chybách (včetně popisu těchto chyb) a podobně.

V případě, že má správnost informace zásadní význam, je vhodné, aby si ji nadřazený systém mohl kdykoliv opakovaně vyžádat a provést tak kontrolu jejím násobným přijetím. Přitom jsem vycházel z předpokladu, že většina informací bude odesílána pouze jednorázově. Tento přístup se ve vytvořené aplikaci uplatňuje tehdy, když systém potřebuje ověřit správnost informací přečtených z ARP tabulky. Ty si může nadřazený systém opakovaně vyžádat. Definovány jsou následující nekritické rámce:

- *UART_PROTO_MESSAGE* - jednoduchá textová informační zpráva, jejíž tělo obsahuje řetězec ASCII znaků.
- *UART_PROTO_ERROR* - jednoduchá textová chybová zpráva, která obsahuje řetězec ASCII znaků představujících popis chyby.
- *UART_PROTO_DEVINFO* - rámec popisuje připojené zařízení. Tělo rámce obsahuje řetězec ASCII znaků ve speciálním formátu popisující dané zařízení.
- *UART_PROTO_ARP* - přenáší informace týkající se ARP tabulky. V těle se ukrývá další hlavička, která obsah rámce blíže specifikuje. Detaily lze vyhledat v programátorské dokumentaci.

Kritické rámce

Dalším typem rámců jsou kritické rámce. Představují obecné příkazy, jejichž ztráta nebo špatná interpretace by mohla vést k chybě. Tyto rámce jsou řešeny samoopravným kódem s trojnásobným opakováním. Rámec má podobný charakter, jako předchozí, nekritická varianta. Začíná hlavičkou, která je dvakrát zopakována, a za níž následuje celé tělo rámce, které se zopakuje ještě dvakrát. Celý rámec je uzavřen nejméně třemi nulami, čímž je zajištěna odolnost proti chybám při přenosu a příjemce následně dokáže rámec opravit bez velké výpočetní náročnosti. Kritické rámce jsou použity výhradně pro komunikaci směrem z nadřazeného systému. Definovány jsou tyto:

- *UART_PROTO_GET_DEVINFO* - příkaz pro vyžádání informací o zařízení.
- *UART_PROTO_GET_ARP_TABLE* - příkaz pro vyžádání celé ARP tabulky.
- *UART_PROTO_CLEAR_ARP_TABLE* - příkaz pro smazání celé ARP tabulky.

- *UART_PROTO_GET_ARP_RECORD* - příkaz pro vyžádání jednoho konkrétního záznamu z ARP tabulky. Obsahuje index do ARP tabulky.
- *UART_PROTO_TOUCH_REMOTE* - toto je demonstrační příkaz, který ukazuje jakým způsobem je možné komunikovat s konkrétním zařízením v síti. Obsahuje index do ARP tabulky. Po úspěšném provedení tohoto příkazu vypíše cílové zařízení na rozhraní UART informaci o tom, že příkaz z koordinátoru dorazil.

6 Obslužná aplikace pro PC

Jak jsem již uvedl v předchozí kapitole, aplikace na modulech RC2400HP dokáže komunikovat s nadřazeným systémem pomocí rozhraní *UART*. Proto jsem se rozhodl realizovat také obslužnou aplikaci pro PC.

Důležitým atributem této aplikace je její uživatelské rozhraní, které může být vytvořeno buď v textové nebo grafické podobě. Vzhledem k tomu, že důležitým prvkem aplikace v modulu je ARP tabulka, bylo vhodné tuto tabulku vizualizovat také na nadřazeném systému. Toto se samozřejmé mnohem lépe a názorněji provádí v grafickém uživatelském prostředí.

Je dobrým zvykem vytvářet multiplatformní aplikace, které jsou schopné provozu na různých platformách a operačních systémech. Multiplatformní aplikace lze psát v mnoha programovacích jazycích. Často používaným jazykem je v tomto ohledu *Java*. Já jsem se rozhodl realizovat aplikaci v jazyce *C++*, pomocí multiplatformní knihovny *Qt*[1], kterou v současnosti vyvíjí firma Nokia. Předností C++ je, že se nejedná o jazyk interpretovaný a navíc, díky práci s knihovnou Qt, lze výsledný program provozovat na velkém množství platforem. Výhodou knihovny Qt je i její bezplatná dostupnost pod licencí *GNU LGPL 2.1*[2].

Drobným nedostatkem knihovny je skutečnost, že přímo neposkytuje API pro práci se sériovým rozhraním na PC. Z tohoto důvodu jsem byl nucen použít rozšiřující třídy *qextserialport*[3], která je však také koncipována multiplatformně. Výsledný program by měl byt tudíž funkční na systémech z rodiny *Microsoft Windows* i na většině *Unixových* systémů, včetně *GNU/Linuxu* nebo *MAC OS X*. Třída qextserialport, použitá ve verzi *1.2-beta*, je dostupná také zdarma za podmínek velmi volné licence *MIT*[4].

6.1 Vývojové prostředky

Protože mou primární pracovní platformou je v současné době 32-bitový operační systém *Debian GNU/Linux 6.0.4, squeeze*, probíhal vývoj primárně právě na tomto systému. Pro vývoj software pomocí knihovny Qt jsou k dispozici standardní prostředky - zejména vývojové prostředí *Qt Creator*, které jsem použil ve verzi *1.3.1*, návrhář uživatelského rozhraní *Qt Designer* a nástroj pro překlad Qt aplikací *Qt Linguist*, oba ve verzi *4.6.3*. Dále standardní

[1]Domovská stránka Qt: *http://qt.nokia.com/*

[2]Stránka projektu GNU s textem licence LGPL: *http://www.gnu.org/licenses/old-licenses/lgpl-2.1.html*

[3]Domovská stránka qextserialport: *http://code.google.com/p/qextserialport/*

[4]Licence třídy qextserialport: *http://code.google.com/p/qextserialport/wiki/License*

nástroje *qmake, GNU make a gcc*, vše z *repositáře*[5] aktuální verze distribuce Debian GNU/Linux.

6.2 Implementace

Jak jsem již uvedl, program, který jsem nazval *uart-app*, disponuje grafickým uživatelským rozhraním a pro práci se sériovým portem počítače využívá třídu *qextserialport*. Část tohoto programu, která implementuje komunikační protokol popsaný v 5.5.1, je z důvodu budoucí modifikovatelnosti uživatelského rozhraní oddělena do zvláštní třídy *Uart*. Výchozím jazykem programu je angličtina, avšak program je standardním způsobe přeložen do češtiny. Proto v případě, kdy je na počítači nastaven jako výchozí jazyk čeština, bude spuštěn v české verzi, jinak bude spuštěna verze anglická. Rozhraní, které vidí uživatel je pouze rozšířením třídy *QMainWindow* a obsahuje dva základní prvky.

Prvním je okno pro zobrazení ARP tabulky. Pokud je k počítači připojen *koordinátor*, zobrazují se v tomto okně zařízení, která jsou obsažena v ARP tabulce koordinátoru. Pokud je připojen *router*, zůstává okno nevyužito. Druhé okno je určeno pro zprávy - veškeré události, které připojené zařízení hlásí nadřazenému systému.

Okno aplikace dále samozřejmě zobrazuje prvky nutné k volbě zařízení, rychlosti komunikace a tlačítka pro připojení, respektive odpojení. Předvolená hodnota rychlosti odpovídá rychlosti aktuálně nastavené v aplikaci v modulu RC2400HP.

Hlavní menu umožňuje kromě ukončení aplikace zasílání příkazů připojenému zařízení a jeho prostřednictvím i zařízením přítomným v ARP tabulce. Detaily obsahuje uživatelská příručka programu, viz A.

Vývoj grafické aplikace nebyl příliš obtížný a použito bylo pouze standardních algoritmů. Výslednou aplikaci jsem následně otestoval jak v systému Debian GNU/Linux 6.0.4 tak v Microsoft Windows XP. Popis vytvořených tříd a funkcí je součástí programátorské dokumentace.

[5] *Repositář* je úložiště používané distribucemi GNU/Linuxu k centralizované distribuci software

7 Závěr

Vzhledem k tomu, že jsem se se zpracovanou problematikou začal seznamovat již téměř před rokem (včetně práce na Projektu 5) podařilo se mi, dle mého názoru, úspěšně splnit všechny body zadání.

Naprogramoval jsem uzly senzorické sítě tak, že výsledná síť je schopná fungovat v požadovaných podmínkách - uzly jsou pohyblivé. Během uplynulé doby jsem nenalezl žádnou aplikaci této technologie, která by měla primárně pracovat podobně jako mnou realizovaná aplikace. Podařilo se ale ověřit, že pro tento účel lze bez větších problémů použít protokol *ZigBee PRO*. Jednodušší protokol *ZigBee* se díky svým omezením, jako je limitovaná hloubka sítě nebo pevný vztah mezi zařízeními *rodič - potomek*, ukázal jako méně vhodný, i když částečně použitelný. Při programování mi velmi pomohlo fórum Texas Instruments, na kterém diskutuje mnoho zkušených vývojářů včetně zaměstnanců této firmy.

Díky implementaci *ARP tabulky* jsem se vypořádal s vlastností protokolu ZigBee, kterou je náhodné přidělování síťové adresy. Tak je možná jednoznačná identifikace každého zařízení v koordinátoru a následně v nadřazeném systému.

Komunikace s nadřazeným systémem je zajištěna pomocí rozhraní *UART*. Multiplatformní aplikace vytvořená pomocí knihovny Qt umožňuje jednoduchou vizualizaci ARP tabulky na PC a dovoluje demonstrovat funkčnost sítě prostřednictvím zasílání příkazů jednotlivým zařízením.

Prokázal jsem, že pomocí protokolu *ZigBee PRO* a použitých zařízení RC2400HP lze vytvořit funkční síť požadovaných vlastností. Na druhou stranu je v tomto modulu je použito MCU architektury i8051, která již není nejnovější. Proto doporučuji více prověřit i možnost použití modulů osazených *MSP430* a *CC2520*.

Osciloskopické měření (viz B) poskytuje informaci pro určení nejmenší možné periody rozesílání výzev v měřeném případě. Bohužel kvůli malému počtu zařízení použitých při měření lze pouze odhadovat situaci ve vetší síti. V rámci dalších prací by bylo vhodné v takové síti podobné měření provést.

Domnívám se, že pokud by v rámci projektu řešeného *Katedrou technologií a měření* byla nakonec zvolen varianta, kterou jsem rozpracoval, bylo by pravděpodobně nutné zaregistrovat u *ZigBee Alliance* nový *aplikační profil*, vzhledem ke specifičnosti této aplikace. Pro dlouhodobější vývoj by bylo zároveň vhodné zakoupit alespoň 1 licenci *IAR EW for 8051*, protože dostupná verze je omezená pouze na 1 měsíc používání. Pak by bylo možné vytvořené softwarové vybavení použít jako základ pro další práci na síti propojující navzájem jednotlivé hasičské obleky.

8 Přehled zkratek

Zkratka	Plné znění
AES	Advanced Encryption Standard
AODV	Ad hoc On-Demand Distance Vector Routing
APS	Application support sublayer
ARP	Address Resolution Protocol
BAN	Body Area Network
DARPA	Defense Advanced Research Projects Agency
DSSS	Direct Sequence Spread Spectrum
HAL	Hardware Abstraction Layer
IEEE	Institute of Electrical and Electronics Engineers
ISR	Interrupt Service Routine
VLSI	Very-Large-Scale Integration
MCU	MicroController Unit
MEMS	Micro-Electro-Mechanical System
ISM	Industrial, Scientific, Medical
ISO	International Organization for Standardization
MAC	Media Access Layer
NIB	Network Information Base
NWK	Network layer
OS	Operating System
OSAL	Operating System Abstraction Layer
OSI	Open Systems Interconnection
PMx	Power Mode x
RREQ	Route Request
RREP	Route Reply
SoC	System on Chip
UART	Universal Asynchronous Receiver Transmitter
WSN	Wireless Sensor Network
WPAN	Wireless Personal Area Network
WLAN	Wireless Local Area Network
ZDO	ZigBee Device Object
ZDP	ZigBee Device Profile

Literatura

[1] Edgar H. CALLAWAY, J.: *Wireless Sensor Networks*. Auerbach publications, 2004, ISBN 0-8493-1823-8.

[2] GISLASON, D.: *ZigBee Wireless Networking*. Elsevier Inc., 2008, ISBN 978-0-7506-8597-9.

[3] HLAVIČKA, J.: *Architektura počítačů*. Vydavatelství ČVUT, 1994, ISBN 80-01-01079-1.

[4] IEEE Computer Society: *IEEE Std 802.15.4^{TM} - 2006* [online]. Dostupné na: `http://standards.ieee.org/getieee802/download/802.15.4-2006.pdf`, 2006.

[5] IEEE Computer Society: *IEEE 802.15 Working Group for WPAN* [online]. Dostupné na: `http://www.ieee802.org/15/about.html`, [cit. 2011-11-26].

[6] KOTON, J., P. ČÍKA a V. KŘIVÁNEK: *Standard nízkorychlostní bezdrátové komunikace ZigBee* [online]. Dostupné na: `http://access.feld.cvut.cz/view.php?cisloclanku=2006032001`, 2006-04-18.

[7] KOVÁŘ, P.: *Samočinný počítač SAPO | Historie počítačů v Československu* [online]. Dostupné na: `http://historiepocitacu.cz/samocinny-pocitac-sapo.html`, 2010.

[8] LEDVINA, J., T. ČOLAKOV, L. DOSTÁLEK, a D. ŠIROKÝ: *Bezdrátové senzorické sítě*. Česká společnost uživatelů otevřených systémů EurOpen.cz, 2009, ISBN 978-80-86583-18-1.

[9] Mobility Management and Networking Laboratory UC Santa Barbara: *Ad hoc On-Demand Distance Vector Routing* [online]. Dostupné na: `http://moment.cs.ucsb.edu/AODV/`, [cit. 2012-03-10].

[10] Radiocrafts AS: *RC2400/RC2400HP Datasheet* [online]. Dostupné na: `http://www.radiocrafts.com/uploads/rc2400_rc2400hp_data_sheet_1_1.pdf`, 2010.

[11] Texas Instruments: *Z-Stack Developer's Guide* [online]. Dostupné na: `http://www.ti.com/tool/z-stack`, 2006 - 2010.

[12] Texas Instruments: *Z-Stack Sample Applications* [online]. Dostupné na: `http://www.ti.com/tool/z-stack`, 2006 - 2010.

[13] Texas Instruments: *CC253x/4x User Guide* [online]. Dostupné na: `http://www.ti.com/product/CC2530`, 2009 - 2010.

[14] Texas Instruments: *CC2530 Datasheet (Rev. B)* [online]. Dostupné na: `http://www.ti.com/product/CC2530`, 2009 - 2011.

[15] Texas Instruments: *Wireless Connectivity Guide* [online]. Dostupné na: `http://www.ti.com/lprf`, 2011.

[16] VOJÁČEK, A.: *ZigBee PRO - nová vylepšená verze bezdrátové komunikace ZigBee* [online]. Dostupné na: `http://automatizace.hw.cz/zigbee-pro-nova-vylepsena-verze-bezdratove-komunikace-zigbee`, 2008-11-02.

[17] Wikipedia: the free encyclopedia: *Ad hoc On-Demand Distance Vector Routing* [online]. Dostupné na: `http://en.wikipedia.org/wiki/AODV`, [cit. 2012-04-12].

[18] Wikipedia: the free encyclopedia: *Smartdust* [online]. Dostupné na: `http://en.wikipedia.org/wiki/Smartdust`, [cit. 2012-04-12].

[19] Wikipedia: the free encyclopedia: *ZigBee / IEEE 802.15.4 Wireless Data Networks* [online]. Dostupné na: `http://en.wikipedia.org/wiki/Electromagnetic_interference_at_2.4_GHz#ZigBee_.2F_IEEE_802.15.4_Wireless_Data_Networks`, [cit. 2012-04-12].

[20] ZigBee Alliance: *ZigBee Specification* [online]. Dostupné na: `http://zigbee.org/Specifications/ZigBee/download.aspx`, 2008-01-17.

[21] ZigBee Alliance: *Features and Benefits* [online]. Dostupné na: `http://docs.zigbee.org/zigbee-docs/dcn/07-5299.pdf`, [cit. 2011-11-26].

A Příručka aplikace uart-app

Překlad zdrojových kódů

Chcete-li provádět úpravy zdrojových kódů aplikace, či ji přeložit pro platformu, pro níž přeložené soubory nejsou dostupné na CD, přečtěte si soubor *README* v adresáři se zdrojovými kódy aplikace uart-app (viz příloha C).

Instalace a spuštění

Microsoft Windows

Instalaci provedete nakopírováním obsahu adresáře *uart-app/bin/windows* (viz C) do vámi zvoleného umístění v počítači. Nyní přejděte do adresáře se spustitelnými soubory a spusťte soubor *uart-app.exe*. Aplikace se spustí v češtině, pokud ji máte nastavenu jako výchozí systémový jazyk. V opačném případě se spustí v angličtině.

GNU/Linux

Instalaci provedete nakopírováním obsahu adresáře *uart-app/bin/linux* (viz C) do vámi zvoleného umístění v počítači. Nyní přejděte do adresáře, se spustitelnými soubory a nastavte práva spuštění:

```
$ cd cesta/ke/spustitelnym/souborum
$ chmod +x uart-app.sh
$ chmod +x uart-app
```

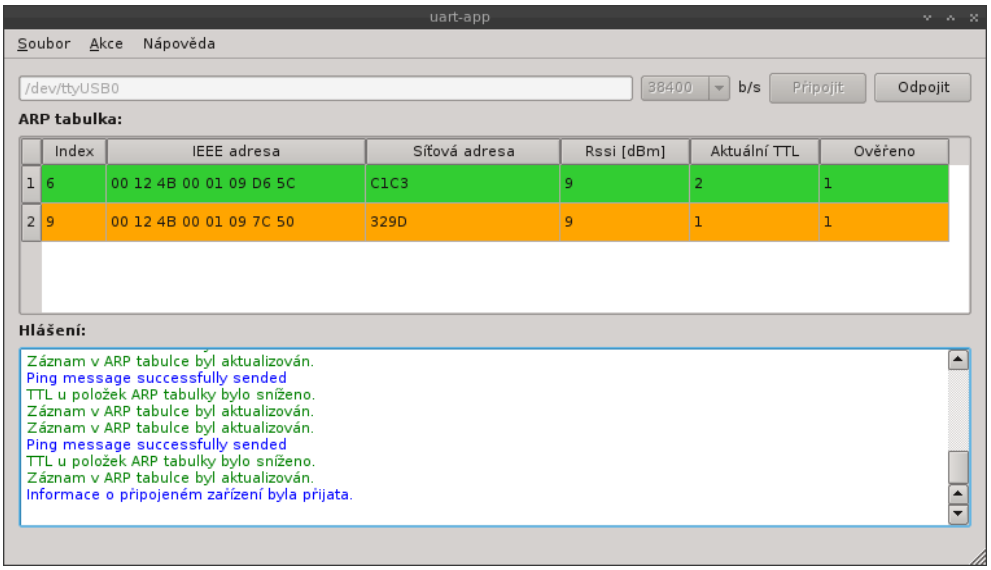
Aplikaci spustíte skriptem *uart-app.sh*:

```
$ ./uart-app.sh
```

Aplikace se spustí v češtině, pokud ji máte nastavenu jako výchozí systémový jazyk. V opačném případě se spustí v angličtině.

Rozvržení prvků v okně aplikace

Okno aplikace je na obrázku A.1. Následuje popis všech prvků okna aplikace uart-app.



Obrázek A.1: Okno aplikace *uart-app* s připojeným koordinátorem, který má 2 položky v ARP tabulce.

Hlavní menu

Hlavní menu slouží k ovládání většiny funkcí aplikace a obsahuje tyto položky:

1. Soubor

Menu soubor obsahuje volby:

- **Připojené zařízení -** zobrazí okno s informacemi o připojeném zařízení
- **Ukončit -** ukončí aplikaci

2. Akce

Pro připojená zařízení, která nejsou koordinátorem má význam pouze první uvedená akce.

- **Získat informace o připojeném zařízení -** vyžádá si zaslání informací od připojeného zařízení, které lze zobrazit pomocí volby *Soubor -> Připojené zařízení*
- **Získat ARP tabulku -** vyžádání si zaslání celé ARP tabulky

- **Vyprázdnit ARP tabulku -** na připojeném zařízení vymaže ARP tabulku
- **Příkazy pro zvolený ARP záznam -** obsahuje příkazy, které se vztahují k jednotlivým záznamům v ARP tabulce:
  - **Poslat testovací příkaz "touch device"-** zvolenému zařízení pošle testovací příkaz "touch device". Výběr zařízení se provádí označením řádku ARP tabulky
  - **Získat ARP záznam -** vyžádá si nové zaslání zvoleného ARP záznamu. Výběr se provádí označením řádku ARP tabulky

3. Nápověda

- **Význam použitých barev -** stiskněte pro zobrazení okna vysvětlujícího význam barev použitých v aplikaci uart-app
- **O tomto programu ... -** zobrazí informace o aplikaci uart-app
- **O Qt ... -** zobrazí informace o Qt

Ovládací prvky pro připojení zařízení

Pole pro zadání jména portu

Sem zadejte jméno portu, ke kterému je zařízení připojeno. Pokud si nevíte rady, pokračujte ve čtení příručky.

Rozbalovací menu pro zvolení komunikační rychlosti

Výběrem jedné z voleb určete rychlost komunikace s připojeným zařízením. Pokud je na připojeném zařízení instalován software, který je distribuován společně s aplikací uart-app, ponechejte výchozí volbu.

Tlačítko připojit

Stisknutím tohoto tlačítka se připojíte k zařízení.

Tlačítko odpojit

Stisknutím tohoto tlačítka se od zařízení odpojíte.

ARP tabulka

Část označená jako ARP tabulka obsahuje obraz ARP tabulky umístěné v zařízení. Pokud k aplikaci není připojen koordinátor, nemá tato část žádný význam. S ARP tabulkou se pracuje pomocí příkazů v menu *Akce*.

Řádek ARP tabulky reprezentuje jeden ARP záznam. Výběr ARP záznamu (respektive síťového zařízení) se provádí označením celého řádku tabulky. Vysvětlení barev použitých v ARP tabulce získáte pomocí *Nápověda -> Význam použitých barev*. Po připojení zařízení je vhodné stáhnout celou ARP tabulku pomocí *Akce -> Získat ARP tabulku*.

Záznam v ARP tabulce obsahuje následující položky:

- **Index -** index v ARP tabulce.
- **IEEE adresa -** 64-bitová IEEE adresa zařízení zobrazená hexadecimálně.
- **Síťová adresa -** 16-ti bitová síťová adresa zařízení zobrazená hexadecimálně.
- **RSSI -** síla signálu, se kterou byla přijata poslední zpráva od daného zařízení.
- **Aktuální TTL -** hodnota TTL záznamu. Tato se postupně snižuje a pokud dosáhne záporné hodnoty, je záznam z tabulky odstraněn.
- **Ověřeno -** počet ověření daného ARP záznamu. Značí kolikrát došlo v nedávné době k ověření ARP záznamu v aplikaci uart-app oproti ARP tabulce v připojeném zařízení - jedná se o ochranu proti chybám při přenosu mezi připojeným zařízením a aplikací uart-app.

Okno hlášení

Obsahuje hlášení ke všem důležitým událostem týkajícím se připojeného zařízení. Události, ke kterým sestavuje hlášení přímo připojené zařízení jsou zobrazené v angličtině, u ostatních zařízení závisí jazyk výpisu na použité lokalizaci aplikace uart-app. Jednotlivá hlášení jsou barevně odlišena. Význam použitých barev naleznete v *Nápověda -> Význam použitých barev*.

Připojení zařízení

Desku *RC2400HP DB* připojte k PC pomocí USB kabelu. Dále postupujte podle použitého operačního systému.

Microsoft Windows

Po připojení zařízení k počítači by se v nástroji *Správce zařízení* měl objevit nový port *COM*. Název tohoto portu, například *COM1*, napište do pole názvu zařízení a stiskněte tlačítko *Připojit*.

GNU/Linux

V dnešní době se soubory zařízení vytvářejí zpravidla automaticky, proto by se po připojení zařízení k počítači měl v adresáři */dev* objevit nový soubor zařízení, například */dev/ttyUSB0*. Jeho název včetně cesty (v našem případě tedy */dev/ttyUSB0*) vepište do pole názvu zařízení a stiskněte tlačítko *Připojit*. Pokud se nelze připojit, zkontrolujte zda máte oprávnění přistupovat k danému zařízení:

```
$ ls -al /dev/ttyUSB0
```

Pokud nemáte dostatečná oprávnění, přístup k zařízení získáte nejsnáze změnou vlastníka souboru. Pod uživatelem *root* proveďte (výraz `uživatel` nahraďte svým uživatelským jménem):

```
$ chown uživatel /dev/ttyUSB0
```

Pokud nebyl soubor zařízení vytvořen automaticky, postupujte dle manuálu vaší linuxové distribuce.

B Osciloskopická měření

Měření pomocí osciloskopu probíhalo na pinu 17 modulu RC2400HP, který je určen k obecnému použití. Byly měřeny velikosti, respektive vzdálenosti, programově vytvořených pulzů.

Bylo provedeno více měření. Hodnoty získané měřeními jsou odečteny přímo z osciloskopu (viz obrázky B.2, B.4, B.8 a B.9). K měření posloužila funkce osciloskopu pro zjištění doby periody a kurzory.

Obrázky B.1, B.3 a B.5 ukazují rozložení pulzů. Měřená hodnota je značena t_x. Obrázky B.6 a B.7 pak znázorňují topologii sítě při daných měřeních.

Následující tabulka ukazuje výsledky měření:

Měření (značení)	Minimum	Maximum	Průměr
Doba startu aplikace (t_s)	277,9 ms	433,6 ms	340,1 ms
Doba odpovědi na zprávu (t_o)	19,7 ms	23,4 ms	21,3 ms
Doba mezi doručením dvou zpráv v topologii 1 (t_{d1})	4,2 ms	11,1 ms	5,7 ms
Doba mezi doručením dvou zpráv v topologii 2 (t_{d2})	5 ms	19,7 ms	11,3 ms

Tabulka B.1: Výsledky měření.

Z tabulky je vidět, že doba nutná pro zpracování jedné přijaté zprávy je menší než 4,2 ms. Vlivem rozmístění uzlů sítě však dochází ke zpožděním. Naměřené hodnoty poskytují dobrou představu o časech v síti tak malých rozměrů jako byla síť měřená. Lze je použít pro přibližný odhad časů v sítích malé velikosti. Za předpokladu, že nárůst času nutného pro doručení všech odpovědí na výzvu koordinátoru je lineární a přímo úměrný počtu skoků, byla by doba, po které budou zpracovány všechny odpovědi, tedy perioda rozesílání výzev:

$$T \geq T_{min},\ T_{min} \approx k \cdot t_{omax} + t_p\ [ms], \tag{B.1}$$

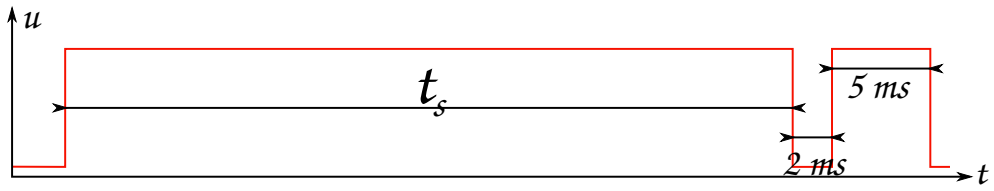
kde k značí maximální hloubku sítě (neboli počet směrovačů), t_p čas nutný pro zpracování jedné odpovědi, t_{omax} čas potřebný pro příchod odpovědi. Po dosazení dostaneme:

$$T_{min} \approx k \cdot 23,4 + 4,2\ [ms]$$

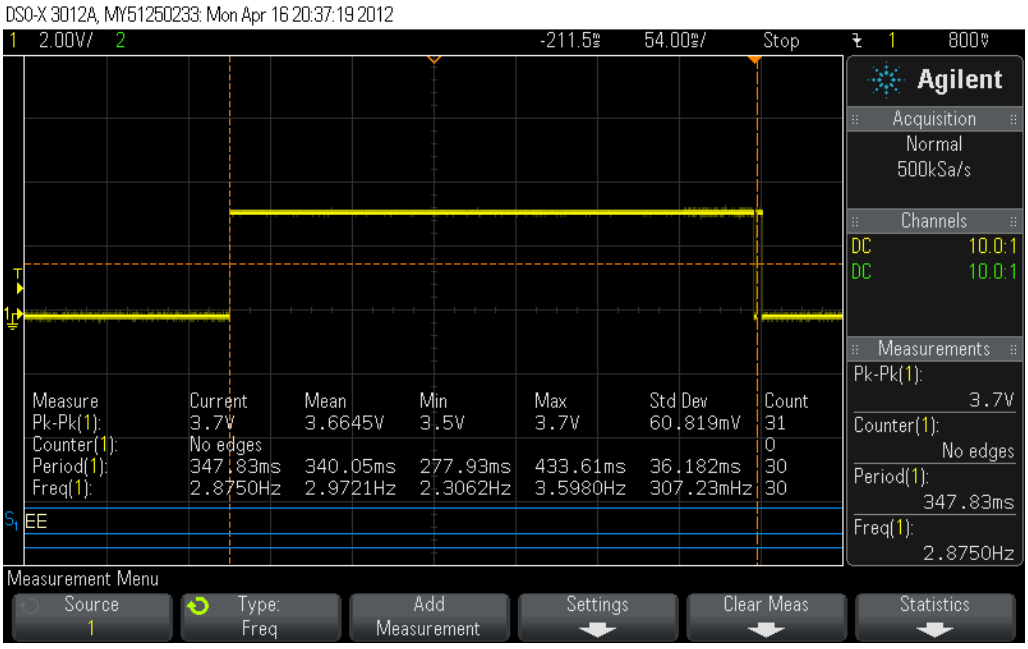
Odhad pro velké sítě bude tímto způsobem pravděpodobně velmi nepřesný. Měření na větší síti by umožnilo přesněji určit rychlost nárůstu času

a zobecnění závěrů. Zároveň lze zachovat metodu použitou při tomto měření, která se ukázala jako dobře realizovatelná.

Měření doby startu aplikace

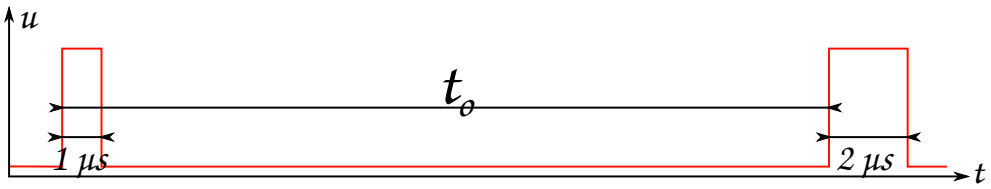


Obrázek B.1: Tvar měřeného pulzu a délka pulzů ohraničujících měřený úsek. Měřena byla doba od resetu zařízení po dokončení inicializační funkce v aplikaci (t_s).

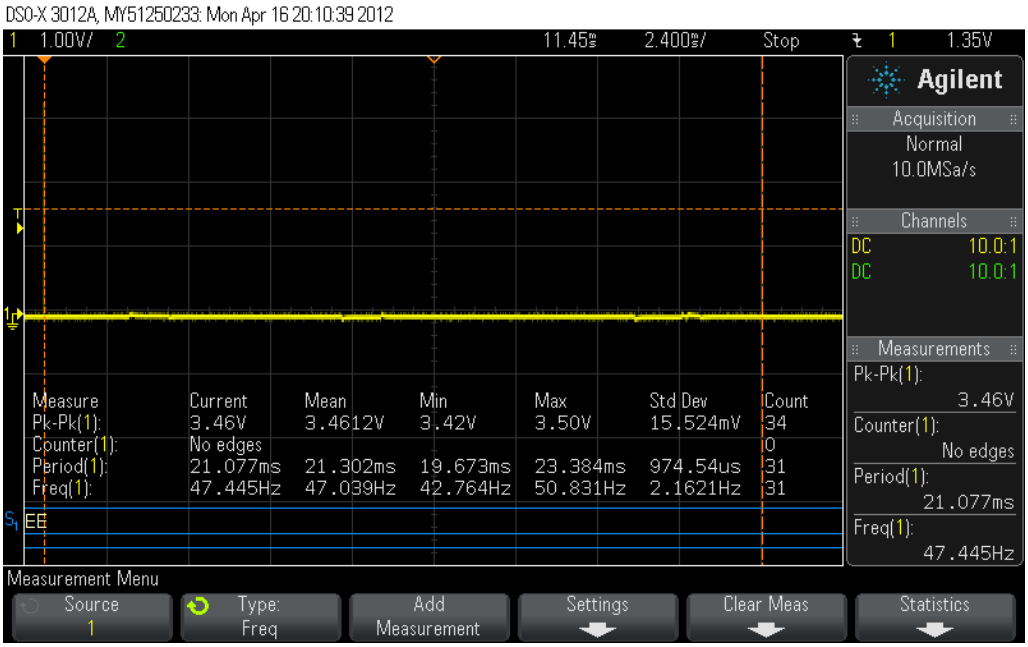


Obrázek B.2: Měření doby startu aplikace v modulu RC2400HP naprogramovaném jako koordinátor.

Měření doby odpovědi

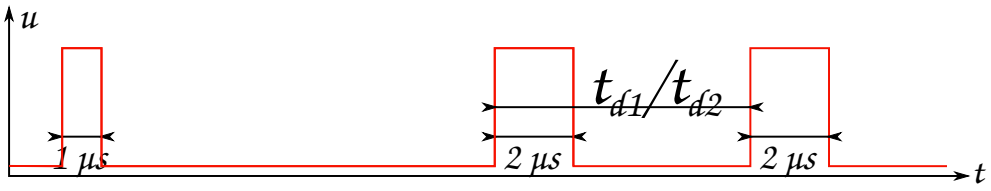


Obrázek B.3: Rozložení pulzů značících okamžik odeslání, respektive přijetí, zprávy aplikací v čase. Měřena byla doba mezi odesláním výzvy a přijetím odpovědi aplikací (t_o).

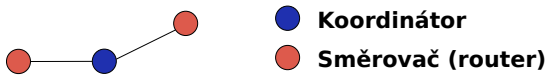


Obrázek B.4: Měření doby mezi odesláním výzvy koordinátoru a doručením odpovědi zpět do koordinátoru (1 zařízení v síti).

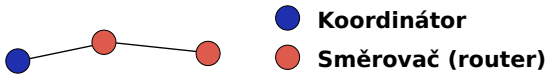
Měření doby mezi doručením dvou zpráv aplikaci



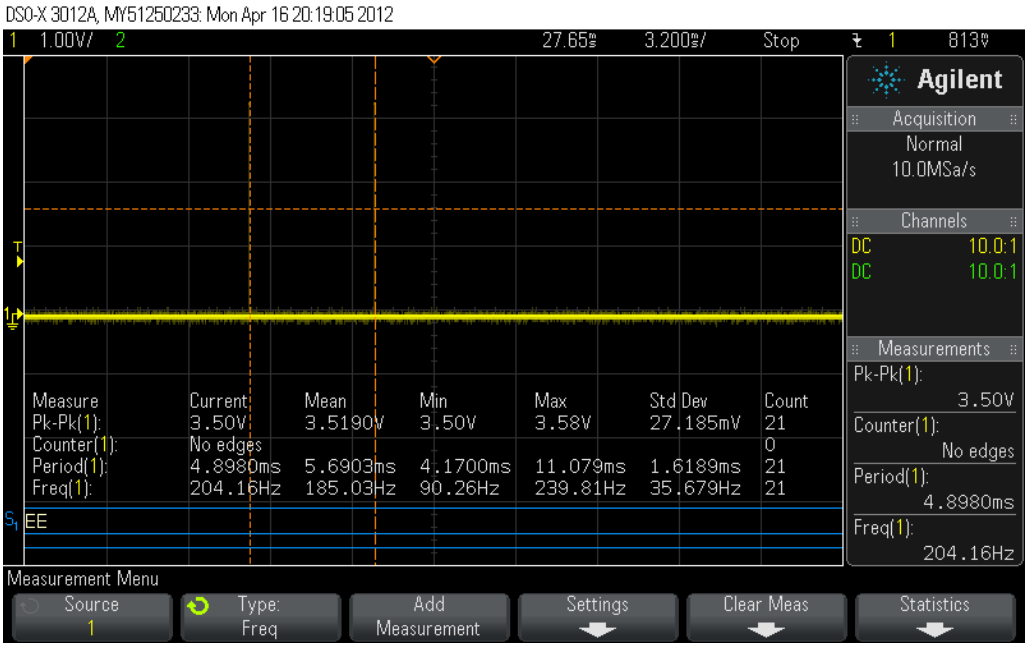
Obrázek B.5: Rozložení pulzů značících přijetí zpráv aplikací. Doba mezi začátkem zpracování dvou po sobě přijatých zpráv (t_{d1} v topologii 1, respektive t_{d2} v topologii 2).



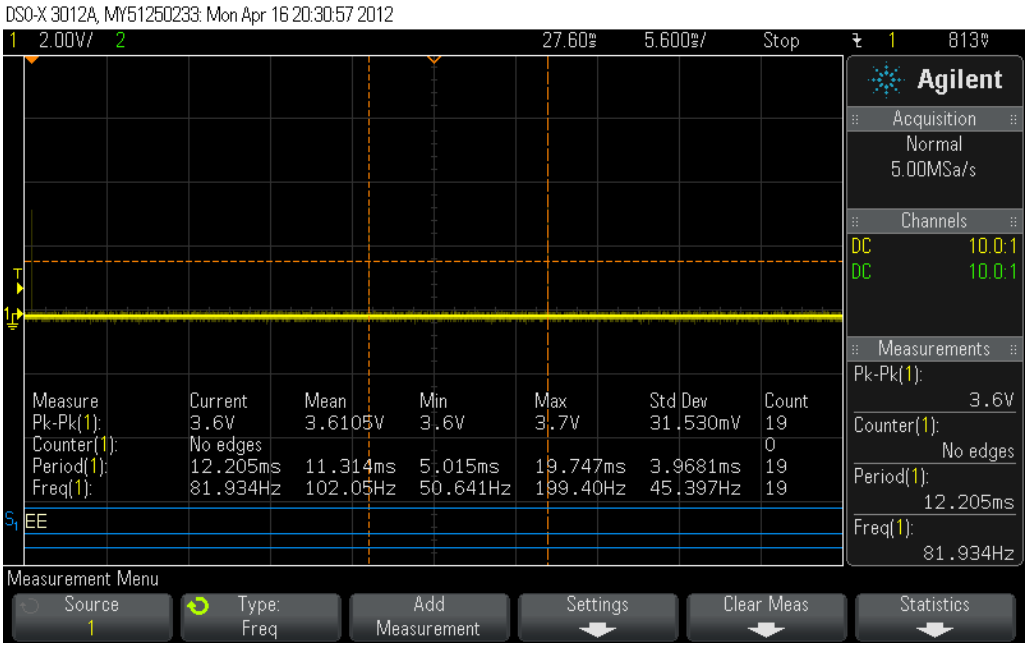
Obrázek B.6: Topologie 1 - dvě zařízení komunikující přímo s koordinátorem.



Obrázek B.7: Topologie 2 - zpráva ze vzdálenějšího směrovače je doručena prostřednictvím bližšího.



Obrázek B.8: Měření doby mezi zpracováním dvou zpráv doručených do koordinátoru od různých zařízení - topologie 1.



Obrázek B.9: Měření doby mezi zpracováním dvou zpráv doručených do koordinátoru různými zařízeními - topologie 2.

C Obsah přiloženého CD

Aplikace pro RC2400HP

- **Zdrojové kódy a programová dokumentace ve formátu HTML:**
  `RC2400HP/Z-Stack/Projects/zstack/Samples/GenericApp/Source`,
  `RC2400HP/Z-Stack/Projects/zstack/Samples/GenericApp/Source/doc`
- **Přeložené soubory:**
  `RC2400HP/hex`
- **Z-Stack 2.3.1 (včetně provedených úprav):**
  `RC2400HP/Z-Stack/`

Obslužná aplikace pro PC (uart-app)

- **Zdrojové kódy a programová dokumentace ve formátu HTML:**
  `uart-app/source`,
  `uart-app/source/doc`
- **Použité knihovny:**
  `uart-app/lib`
- **Spustitelné soubory pro GNU/Linux:**
  `uart-app/bin/linux`
- **Spustitelné soubory pro Microsoft Windows:**
  `uart-app/bin/windows`

Ostatní

- **Videozáznam provozu sítě:**
  `video`
- **Zdrojové soubory textu bakalářské práce:**
  `text-source`
- **Text bakalářské práce:**
  `BP_Belohoubek.pdf`
- **Archiv s obsahem celé bakalářské práce:**
  `BP_Belohoubek.tar`