VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

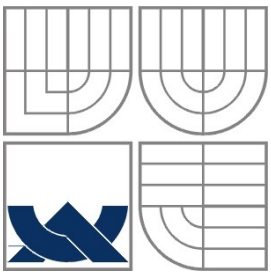
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

SYSTÉM PRO SLEDOVÁNÍ A SPRÁVU SENZOROVÝCH SÍTÍ

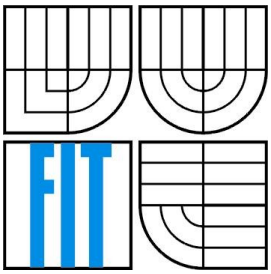
BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE FILIP SEDLÁK
AUTHOR

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

SYSTÉM PRO SLEDOVÁNÍ A SPRÁVU SENZOROVÝCH SÍTÍ

SYSTEM FOR MONITORING AND SUPERVISORY OF SENSOR NETWORKS

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

FILIP SEDLÁK

VEDOUCÍ PRÁCE
SUPERVISOR

Mgr. ROMAN TRCHALÍK

BRNO 2008

Abstrakt

Práce se zabývá metodou správy IP sítí pomocí protokolu SNMP. Dále popisuje principy sítí založených na bezdrátovém standardu ZigBee a zabývá se možností použití technologie SNMP pro správu takových sítí. Praktická část se věnuje analýze a vývoji konkrétního systému pro správu a sledování bezdrátové senzorové sítě skládající se ze ZigBee technologií vybavených vývojových desek PICDEM Z a diskutuje možnosti a problémy užití podobného nástroje při správě obecné sítě zařízení ZigBee.

Klíčová slova

SNMP, správa sítí, SNMP agent, MIB, senzorové sítě, bezdrátové sítě, ZigBee, PICDEM Z

Abstract

The thesis deals with a network management method using SNMP protocol. In the following it describes principles of the networks based on the ZigBee wireless standard and discusses the possibilities of using SNMP technology for management of this kind of networks. The practical part describes analysis and development of a particular system for management of wireless networks made of ZigBee enabled PICDEM Z development boards. Finally it discuses options and problems of using this type of tool to administer a general network of ZigBee devices.

Keywords

SNMP, network management, SNMP agent, MIB, sensor networks, wireless networks, ZigBee, PICDEM Z

Citace

Filip Sedlák: Systém pro správu a sledování senzorových sítí, bakalářská práce, Brno, FIT VUT v Brně, 2008

Název bakalářské práce v jazyce práce

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Mgr. Romana Trchalíka.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Filip Sedlák
14. května 2008

Obsah

Obsah....1
1 Úvod....3
2 Simple Network Management Protocol (SNMP)....4
2.1 Management Information Base (MIB)....4
2.1.1 Typy objektů definovaných SMIv1....5
2.1.2 Struktura MIB modulu....6
2.2 Model manažer-agent....7
2.3 Aplikační protokol SNMPv1....8
2.4 Základní atributy zařízení v IP sítích....9
2.4.1 Příklady atributů síťového uzlu....9
2.4.2 Příklady atributů síťového rozhraní....10
3 Senzorové sítě ZigBee....11
3.1 IEEE 802.15.4....11
3.2 ZigBee....12
3.2.1 Logické typy zařízení....12
3.2.2 Topologie....13
3.2.3 ZigBee profily....14
3.2.4 Adresy uzlů v síti ZigBee....14
3.2.5 Zasílání zpráv....15
3.2.6 Vytvoření sítě....15
3.3 Microchip Stack for ZigBee Protocol....15
4 Návrh....16
4.1 Dostupné nástroje a technické vybavení....16
4.2 Umístění SNMP agenta....17
4.3 SNMP agent....18
4.3.1 Atributy zařízení a sítě ZigBee, MIB agenta....18
4.3.2 Požadavky na agenta....19
4.4 Koordinátor....20
4.5 Koncové uzly....20
5 Implementace....21
5.1 Firmware koordinátora....21
5.1.1 Protokol sériové linky....21
5.1.2 Příjem zpráv....23

5.1.3 Tabulka uzlů....23
5.1.4 Zjišťování atributů koncových uzlů....24
5.1.5 Profil koordinátora....24
5.2 Agent....24
5.2.1 Sériové rozhraní....24
5.2.2 Rozhraní SNMP....25
5.3 Manažer....25
5.4 Problémy a návrhy na vylepšení....26
5.4.1 Návrhy na rozšíření:....26
6 Závěr....27
Literatura....28
Seznam použitých zkratek....29
Seznam příloh....30
Přílohy....31

1 Úvod

Již od prvopočátků komunikačních sítí je proces správy a sledování nezbytnou součástí jejich provozu. S rostoucí složitostí sítí – množstvím připojených zařízení, komplikovanou infrastrukturou a pokročilými mechanismy doručování dat, vznikala potřeba komplexních a dostatečně obecných nástrojů pro zajištění zmíněných úkolů. Dnes, v době celosvětové počítačové sítě rodiny TCP/IP – Internetu, již existuje několik rozšířených platforem a protokolů, z nichž některé mají dokonce povahu otevřeného standardu a tedy širokou podporu zařízení, hotových aplikací a vývojových nástrojů. Takovým standardem je Simple Network Management Protocol (SNMP) – aplikační protokol pro systémy správy a sledování sítí, kterým se práce v této oblasti výhradně zabývá.

Technologický pokrok přinesl vyspělé síťové techniky (bezdrátový přenos, směrování, decentralizace) ze světa IP sítí např. do oblasti jednoduchých řídících a senzorových zařízení. Aplikací tohoto oboru, která zaznamenává v posledních letech rostoucí popularitu a široké uplatnění, jsou bezdrátové senzorové sítě. Otevřeným komunikačním standardem na tomto poli je například ZigBee, určené bezdrátovým ad-hoc sítím autonomních senzorových zařízení, s nízkou cenou a extrémně dlouhou dobou provozu na baterie. Tomuto standardu se práce dále věnuje.

Praktická část popisuje návrh a vývoj konkrétního systému pro sledování a správu senzorové sítě standardu ZigBee pomocí nástrojů postavených na technologii SNMP. Tato sada nástrojů má za úkol sledovat a řídit stav senzorové sítě a jejích jednotlivých prvků z kontextu sítě TCP/IP (Internetu).

Následující kapitola rozebírá protokol SNMP, popisuje jeho jednotlivé části, architekturu typického SNMP systému a jeho úlohu při správě sítě. Nakonec uvádí příklady některých běžně používaných atributů spravovaných zařízení v sítích IP a jejich zařazení v rámci SNMP.

Třetí kapitola v krátkosti popisuje jednotlivé vrstvy protokolu ZigBee, vysvětluje pojmy dále používané v praktické části práce. V závěru se zmiňuje o implementaci protokolu ZigBee firmy Microchip.

Čtvrtá kapitola rozebírá praktickou část práce, fáze návrhu a implementace systému a stav práce. Krátce se věnuje problémům a návrhům na rozšíření.

Závěr shrnuje stav projektu, uvádí přínosy a možnosti dalšího vývoje.

2 Simple Network Management Protocol (SNMP)

SNMP je protokol používaný systémy pro správu sítí rodiny TCP/IP. Byl uveden v roce 1988 jako odpověď na rostoucí potřebu standardizace v této oblasti a dnes je de-facto nejpoužívanějším a nejšířeji podporovaným nástrojem svého druhu. Je tvořen skupinou standardů – protokolem aplikační vrstvy, databázovým modelem a definicemi datových objektů.

Jádrem protokolu je jednoduchá skupina příkazů, které správci umožňují zjišťovat a měnit jednotlivé atributy SNMP-vybaveného zařízení jako jsou stav síťového rozhraní, počet přenesených oktetů, počet kolizí, vytíženost procesoru, paměti, ale i otáčky ventilátoru, teplotu atp. Mimoto definuje asynchronní zprávy generované zejména v případě neočekávaných situací, které mohou na sledovaném zařízení nastat (například selhání některého z rozhraní, vyčerpání paměti RAM apod.)

SNMP nejčastěji slouží ke správě síťových směrovačů, ale díky své rozšiřitelné povaze lze použít pro správu prakticky libovolných IP zařízení, jako jsou Linuxové servery, stanice s operačním systémem Windows, tiskárny, IP telefony, napájecí zdroje atd.

V posledních letech, společně se zaváděním TCP/IP do oboru průmyslové automatizace a řídicí techniky, bývá SNMP poměrně často využíván coby standardní, rozšiřitelný protokol s nízkou režií i v těchto oblastech, pro které nebyl původně zamýšlen. [10]

SNMP se v současnosti vyskytuje ve třech verzích (SNMPv1, SNMPv2 a SNMPv3). Tato práce se v rámci zjednodušení zabývá první verzí (SNMPv1). Změny v následujících verzích nejsou principiální povahy, jedná se zejména o zdokonalení v oblasti autentizace a bezpečnosti.

[1], [2]

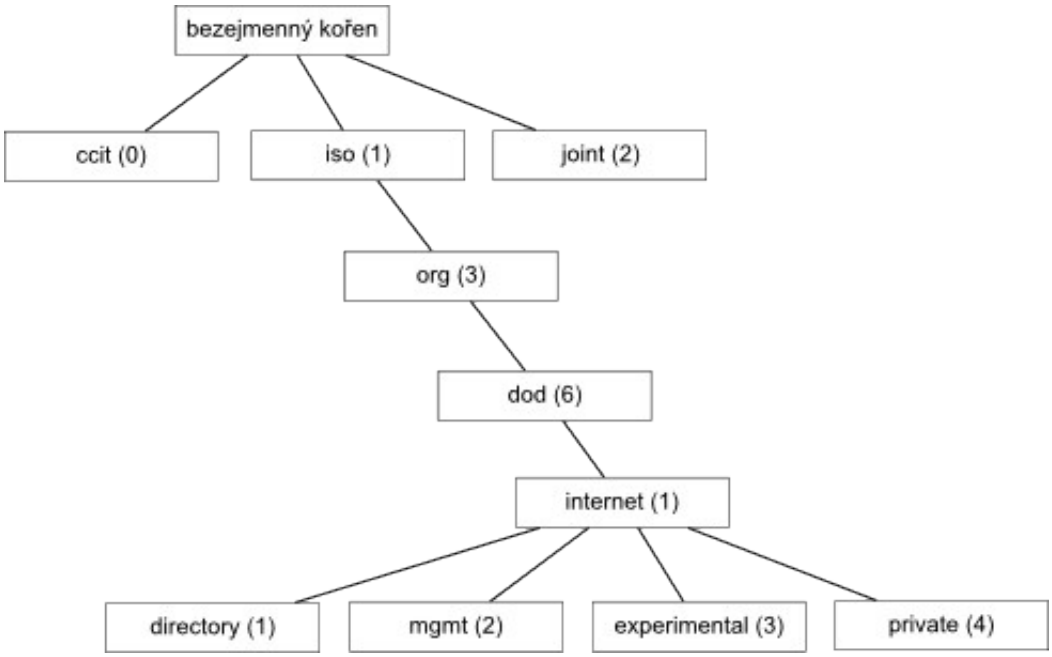
2.1 Management Information Base (MIB)

Protokol SNMP sám o sobě nedefinuje, které atributy (informace) by měl spravovaný systém poskytovat. Místo toho využívá rozšiřitelného modelu, ve kterém jsou dostupné atributy definovány a zařazeny v tzv. databázích spravovaných informací – Management Information Base (MIB) coby tzv. spravované objekty – Managed Object (MO).

Každé zařízení potom implementuje jednu či více MIB databází s popisem informací, jejichž správu umožňuje.

Každá databáze MIB má logickou strukturu stromu, v němž je každý uzel jednoznačně identifikován pomocí tzv. identifikátoru objektu – Object Identifier (OID). Identifikátor objektu, tvořící posloupnost čísel oddělených tečkami (alternativně posloupnost jmen oddělených tečkami), představuje cestu k danému objektu v rámci celosvětové stromové hierarchie. Tento globální hierarchický jmenný prostor s bezejmenným kořenem tvoří standard, jehož vyšší úrovně jsou řízeny jednotlivými organizacemi spravujícími standardy (např. ISO, IETF), identifikátory nižších úrovní jsou pak přidělovány různým organizacím, výrobcům i jednotlivcům. Každá existující schválená MIB má tedy své

místo ve jmenném prostoru určité organizace, o jejíž přidělení musí tvůrce požádat. Kromě toho existují i jmenné prostory pro uživatelské a experimentální využití.



Obr. 2.1: Globální strom jmen

Příkladem standardní MIB, kterou implementuje většina SNMP-vybavených spravovaných zařízení je tzv. MIB-II, která definuje typické informace společné většině IP zařízení, jako jsou informace o rozhraních (rychlost, přijaté oktety, odeslané oktety atd.), ale také informace o zařízení samotném (umístění systému, kontakt atd.).

Jednotlivé databáze MIB (též tzv. MIB moduly) a jejich spravované objekty jsou popisovány standardní textovou formou zvanou Structure of Management Information (SMI), která je podmnožinou abstraktní notace pro popis datových struktur, jejich kódování a přenosu – Abstract Syntax Notation One (ASN.1).

[1], [2], [3]

2.1.1 Typy objektů definovaných SMIv1

Databáze MIB protokolu SNMP verze 1 používají SMI verze 1 (SMIv1), která definuje následující typy spravovaných objektů:

INTEGER – 32-bitové celé číslo, používané také k reprezentaci stavů, příp. pravdivostních hodnot. Například stav rozhraní směrovače může nabývat tří hodnot „up“ (hodnota 1), „down“ (2), „testing“ (3).

OCTET STRING – řetězec 0 až n oktetů (bytů), reprezentující nejčastěji textové řetězce, případně fyzické adresy.

Counter – 32-bitové přirozené číslo reprezentující počet, který je v průběhu sledování opakovaně inkrementován o hodnotu 1. V případě dosažení maximální 32-bitové hodnoty se po další

inkrementaci nastaví opět na 0. V případě restartu by měly být všechny hodnoty tohoto typu vynulovány. Obvykle udává např. počet přijatých a odeslaných oktetů.

OBJECT IDENTIFIER – OID zapsaný jako řetězec čísel oddělených tečkami. Například „.1.3.6.1.3.1“ je OID nacházející se ve jmenném prostoru .iso.org.dod.internet.experimental.

IpAddress – 32-bitová IPv4 adresa.

NetworkAddress – stejný jako IpAddress, ale použití tohoto typu vyjadřuje, že se jedná o jiný typ adresy než IPv4 (vzhledem ke své délce bohužel nelze použít k uložení IPv6 adresy).

Gauge – datový typ podobný typu Counter, avšak navíc s možností dekrementace. Gauge by, na rozdíl od typu Counter, nikdy neměl přesáhnout své maximální a minimální hodnoty. Může udávat např. rychlost rozhraní směrovače.

TimeTicks – 32-bitové přirozené číslo reprezentující čas v jednotkách setin sekund. Obvykle zobrazuje čas, po který je zařízení spuštěno – „uptime“.

Opaque – značí uložení jiného ASN.1 typu jako OCTET STRING.

SEQUENCE – strukturovaný datový typ představující seznam libovolného počtu (0-n) obecně libovolných ASN.1 typů. Používá se k definici konceptuálních řádků tvořících tabulky.

SEQUENCE OF – reprezentuje spravovaný objekt, který se skládá z objektů typu SEQUENCE. Používá se k definici tabulky konceptuálních řádků.

[1], [4]

2.1.2 Struktura MIB modulu

Celý MIB modul je tvořen jedinou základní sekcí skládající se s hlavičky se jménem modulu, ukončovacím symbolem a dalšími podsekcemi v jejím těle.

```
<jméno modulu> DEFINITIONS ::= BEGIN
     <podsekce>
END
```

Tělo modulu se pak typicky skládá ze třech základních částí. První je sekce IMPORTS.

```
IMPORTS
     <identifikátor 1, identifikátor 2, ..., identifikátor n,>
          FROM <název externí MIB>
     <...>;
```

Sekce IMPORTS se skládá obecně z několika klauzulí FROM umožňujících import objektů a identifikátorů OID z externích MIB souborů. Další částí je definice jmen a hierarchie uživatelských identifikátorů, které budou použity ve zbytku modulu.

```
<jméno OID> OBJECT IDENTIFIER ::= {<jméno nadřazeného OID> <číslo
OID>}
```

Každý identifikátor je určen svým jménem, jménem nadřazeného OID a posledním číslem v cestě stromovou strukturou. Poslední částí je definice jednotlivých spravovaných objektů:

```
<jméno objektu> OBJECT-TYPE
     SYNTAX <datový typ>
     ACCESS <práva přístupu>
     STATUS <povinnost implementace>
```

```
DESCRIPTION
      "Textový popis spravovaného objektu"
::= {<jméno nadřazeného OID> <číslo OID>}
```

Každý objekt je popsaný svým jménem, datovým typem, který musí být importovaný, nebo dále definovaný (případ řádků a tabulek) a přístupovými právy:

- **read-only** – hodnotu objektu je možné pouze číst
- **read-write** – hodnotu objektu je možné číst i měnit
- **write-only** – hodnotu objektu je možné pouze měnit
- **not-accessible** – hodnotu nemá smysl ani číst, ani měnit, např. pokud se jedná o tabulku nebo konceptuální řádek – v těchto případech má smysl číst pouze hodnoty jednotlivých buněk

Dále je objektu přiřazen stav, který udává povinnost zařízení implementujícího tuto MIB podporovat tento konkrétní objekt:

- **mandatory** – objekt je povinný
- **optional** – objekt je volitelný
- **obsolete** – objekt je zastaralý, v budoucnosti se dá očekávat jeho vypuštění

Nakonec je objekt jednoznačně zařazen do stromové struktury pomocí jména nadřazeného objektu a svého čísla.

Část konkrétního MIB modulu vytvořeného v rámci praktické části projektu najdete v přílohách (Příloha A).

[1], [4]

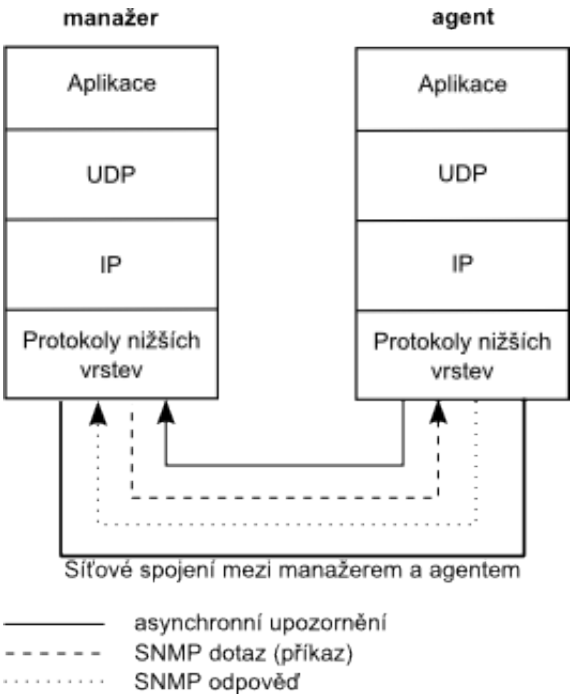
2.2 Model manažer-agent

Obvyklý SNMP systém se skládá minimálně z těchto tří prvků:

1. **Spravované zařízení**, jehož atributy sledujeme (nastavujeme).

2. **Agent**, jež má přístup k atributům spravovaného zařízení (typicky na něm přímo běží jako aplikace, nebo je součástí jádra operačního systému) a zpřístupňuje je jako spravované objekty v rámci implementovaných MIB. Zpracovává dotazy manažera, zasílá na ně odpovědi, případně posílá asynchronní zprávy.

3. **Manažer**, též NMS (Network Management System), zasílá dotazy agentovi, přijímá, zpracovává a prezentuje odpovědi. Volitelně přijímá asynchronní zprávy. Ke své práci obvykle potřebuje znát popis MIB implementovaných agentem.



Obr. 2.2: Model manažer-agent

[1], [2]

2.3 Aplikační protokol SNMPv1

SNMP je jedním z aplikačních protokolů rodiny TCP/IP. Jako protokol transportní vrstvy využívá nespolehlivý, nespojovaný User Datagram Protocol (UDP). Standardním cílovým portem pro komunikaci dotaz-odpověď ve směru manažer-agent je 161, v případě asynchronních zpráv ve směru agent-manažer port 162. Protokol UDP, díky své jednoduchosti, dovoluje použít SNMP často i v okamžicích přetížené a nespolehlivé sítě, na druhou stranu nezajišťuje potvrzení o doručení zpráv. To je přenecháno na protokolu vyšší vrstvy – SNMP, který však tento problém řeší jen u některých typů zpráv.

Zprávy protokolu SNMP jsou stejně jako spravované objekty definovány pomocí abstraktní notace ASN.1. Převod abstraktních objektů do posloupnosti bitů zajišťují pravidla BER (Basic Encoding Rules).

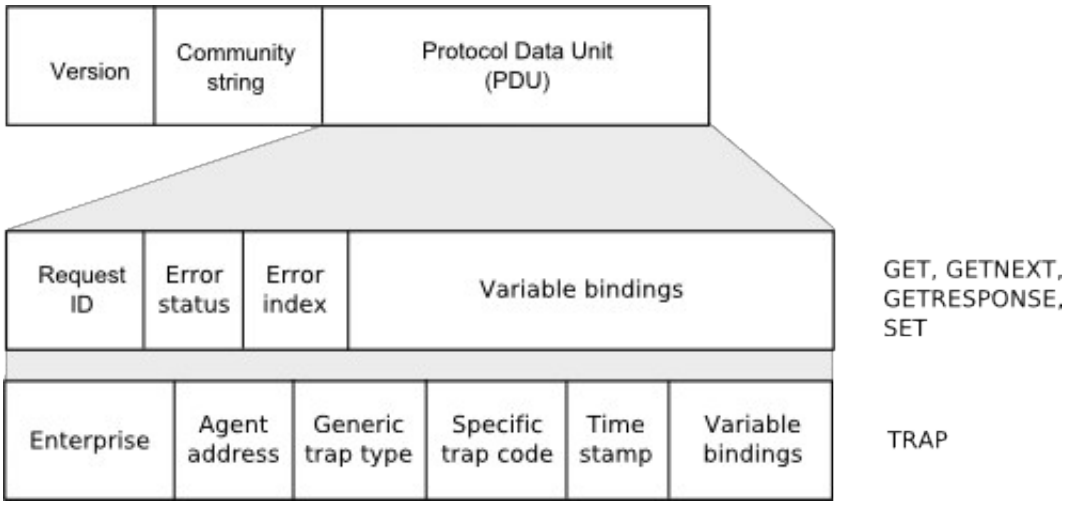
Na nejvyšší úrovní se SNMPv1 zpráva dělí na tři části – identifikace verze (v případě SNMPv1 vždy číslo 0), tzv. community string – řetězec sloužící jako primitivní způsob zabezpečení formou otevřeného hesla a nakonec tělo zprávy – PDU (Protocol Data Unit).

PDU část pak obsahuje některý z příkazů SNMP komunikace, SNMPv1 definuje tyto typy:

- GET REQUEST – požadavek manažera na získání hodnoty MO
- GETNEXT REQUEST – požadavek manažera na získání následující hodnoty v MIB hierarchii, slouží k procházení strukturovaných dat, zejména tabulek

- GET RESPONSE – odpověď agenta na předchozí dva příkazy, obsahující v ideálním případě (nedojde-li k chybě, např. po dotazu na neexistující OID) zjištěnou hodnotu
- SET REQUEST – požadavek manažera na nastavení hodnoty
- TRAP – asynchronní zpráva (bez vyžádání), kterou posílá agent na nastavenou adresu v případě mimořádné události

Tyto zprávy se vždy skládají z kontrolních polí a nakonec ze strukturovaného ASN.1 typu SEQUENCE (tzv. VarBindList), obsahujícího volitelný počet uspořádaných dvojic rovněž typu SEQUENCE (tzv. VarBind), kterou např. u typu SET REQUEST tvoří objekt typu OID identifikující cílový spravovaný objekt a např. objekt typu INTEGER představující nastavovanou číselnou hodnotu.



Obr. 2.3: Struktura SNMP zprávy

2.4 Základní atributy zařízení v IP sítích

Atributy, které má smysl sledovat u zařízení v sítích IP v souvislosti s jejich správou, definuje standard Management Information Base for Network Mangement of TCP/IP-based internets (MIB-II). Objekty této MIB se v globálním stromu jmen nacházejí na cestě .iso.org.dod.internet.mgmt.mib-2. Pro představu uvádím příklady několika často používaných:

[1]

2.4.1 Příklady atributů síťového uzlu

Atributy popisujících uzel jako celek se nachází ve větvi mib-2.host.

Příklady atributů popisujících systém (mib-2.host.hrSystem):

- hrSystemUptime – udává čas od chvíle, kdy byl systém naposled inicializován
- hrSystemDate – udává lokální systémový čas
- hrSystemNumProcesses – informuje o aktuálním počtu uživatelských sezení

Některé atributy popisující datová úložiště (mib-2.host.hrStorage)

- hrMemorySize – udává velikost fyzické paměti (typicky RAM)
- tabulka hrStorageTable s řádky hrStorageEntry – tabulka úložných prostorů
  - hrStorageType – typ úložného prostoru
  - hrStorageSize – kapacita
  - hrStorageUsed – využité místo

Atributy jednotlivých procesorů (mib-2.host.hrDevice.hrProcessorTable):

- hrProcessorEntry.hrProcessorLoad – vytíženost procesoru

[5]

2.4.2 Příklady atributů síťového rozhraní

Informace o síťových rozhraních se nachází na cestě mib-2.interfaces:

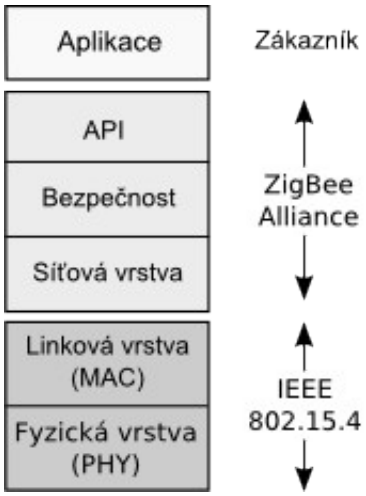
- ifNumber – udává počet síťových rozhraní
- tabulka ifTable s řádky ifEntry popisuje jednotlivá síťová rozhraní
  - ifDesc – textový popis rozhraní
  - ifSpeed – odhad momentální šířky pásma
  - ifPhysAddress – fyzická adresa
  - ifOperStatus – provozní stav rozhraní (1 – up, 2 – down, 3 – testing atd.)
  - ifInOctets – počet přijatých oktetů
  - ifOutOctets – počet odeslaných oktetů
  - ifInErrors – počet přijatých oktetů, které obsahovaly chybu, takže nemohly být doručeny protokolu vyšší vrstvy

[5], [10]

3 Senzorové sítě ZigBee

ZigBee je komunikační standard navržený pro bezdrátové sítě nízkých přenosových rychlostí přenášející malé objemy dat. Díky své jednoduchosti mohou funkci uzlů sítě ZigBee plnit velice jednoduchá (a tedy levná) zařízení s extrémně nízkou spotřebou. Do této kategorie patří například bateriově napájené bezdrátové senzory, používané stále častěji v oblastech průmyslu, spotřební elektroniky a řízení budov.

Standard ZigBee lze popsat několikavrstvým OSI modelem. Na nejnižších vrstvách (fyzické a linkové) je definován standardem IEEE 802.15.4 (na němž mimo ZigBee staví například ještě specifikace MiWi). Další dvě vrstvy (síťová a transportní) jsou definovány organizací ZigBee alliance. Aplikační vrstvu definuje tvůrce konkrétního řešení (zákazník).



Obr. 3.1: ZigBee model

[6], [7]

3.1 IEEE 802.15.4

ZigBee protokol používá specifikaci IEEE 802.15.4 jako fyzickou (PHY, Physical Access Layer) a linkovou (MAC, Media Access Layer) vrstvu. IEEE 802.15.4 definuje tři frekvenční pásma používané v různých částech světa, v rámci jednotlivých pásem jsou definovány tzv. Kanály, pásma se mimo frekvenční rozsahy liší i svými přenosovými rychlostmi.

Frekvenční pásmo	Použití	Kanály	Přenosová rychlost
2.4 GHz	celosvětové	11-28	250 kb/s
915 MHz	Amerika	1-10	40 kb/s
868 MHz	Evropa	0	20 kb/s

Tabulka 1: Frekvenční pásma a kanály definované IEEE 802.15.4

Pro přenos se datový signál moduluje metodou O-QPSK (Offset Quadrature Phase-Shift keying) a vzduchem přenášejí prostřednictvím DSSS (Direct Sequence Spread Spectrum), tedy něco podobného jako v případě technologie WIFI. Pro přístup na kanál se využívá metody CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance and Optional Time Slotting). [6]

Linková (MAC) vrstva definuje způsob komunikace mezi uzly prostřednictvím zpráv (rámců). Maximální délka rámce je 127 bitů včetně 16 bitové CRC hodnoty sloužící k ověření integrity dat. Rámec může být jednoho z těchto typů:

- **Data Frame** – rámec pro přenos užitečné informace
- **Acknowledgement Frame** – rámec pro potvrzení o doručení na úrovni linkové vrstvy
- **MAC Command Frame** – rámec pro přenos konfiguračních a řídících příkazů souvisejících s provozem sítě
- **Beacon Frame** – synchronizační super-rámec používaný pouze v případě řízení přístupu ke společnému médiu technologií časových slotů

IEEE 802.15.4 definuje dva fyzické typy zařízení, lišící se svými schopnostmi, přístupem k úspoře energie a tedy způsobem použití.

Typ zařízení	Poskytované funkce	Zdroj energie	Chování přijímače v době nečinnosti
Full Function Device (FFD)	všechny	elektrická síť	zapnutý
Reduced Function Device (RFD)	omezené	baterie	vypnutý

Tabulka 2: Typy zařízení podle IEEE 802.15.4

[7], [8]

3.2 ZigBee

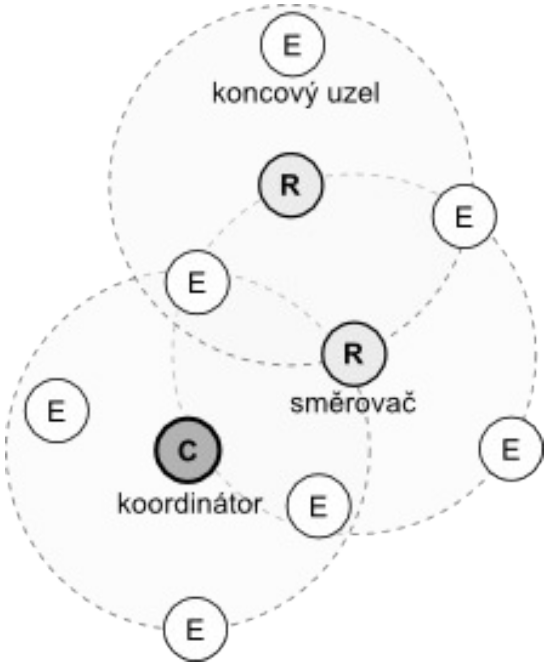
3.2.1 Logické typy zařízení

Standard ZigBee potom dále definuje tři logické typy zařízení nacházející se v bezdrátové síti:

ZigBee zařízení	Požadované IEEE zařízení	Typické použití
Koordinátor	FFD	V síti vždy právě jeden. Zakládá síť, přiděluje adresy a udržuje tabulku vazeb (viz dále).
Směrovač	FFD	Volitelný. Zvětšuje fyzický rozsah bezdrátové sítě.
Koncový uzel	RFD nebo FFD	Senzorové zařízení – provádí sledování, příp. řízení.

Tabulka 3: Logické typy zařízení ZigBee

Minimální síť tvoří koordinátor a množina koncových uzlů.

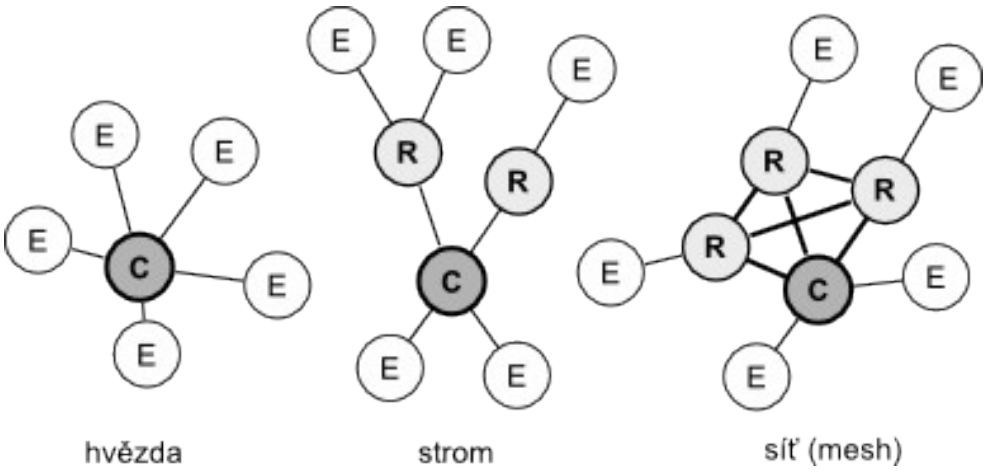


Obr. 3.2: Typická síť ZigBee

[7]

3.2.2 Topologie

ZigBee standard podporuje bezdrátové sítě základních topologií. Nejjednodušší – hvězda, představující koordinátora a skupinu koncových uzlů na něj přímo připojených. Dále topologii typu strom, obsahující alespoň jeden směrovač, ve které vždy existuje pouze jedna směrovací cesta od odesílatele k příjemci. Nakonec topologii typu síť (mesh), ve které spolu směrovače volně komunikují a vytvářejí tak alternativní směrovací cesty. [7]



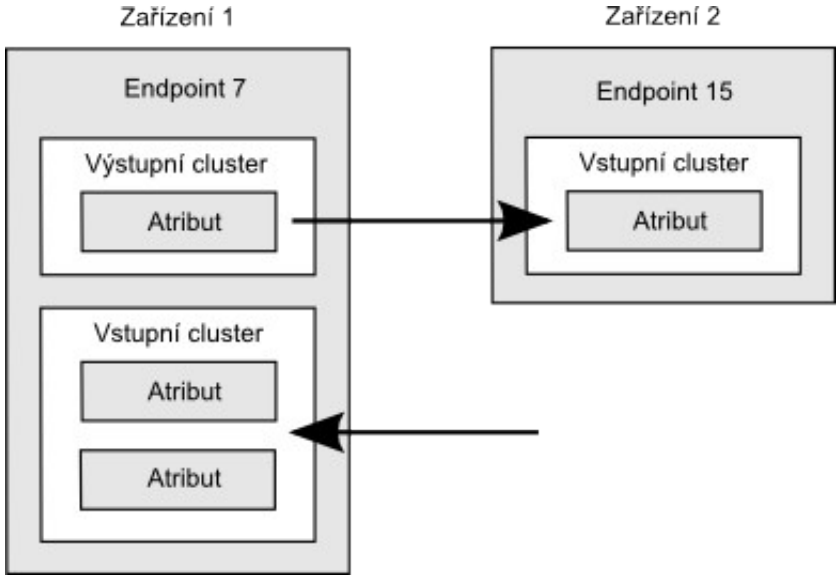
Obr. 3.3: Různé topologie sítě ZigBee

3.2.3 ZigBee profily

ZigBee protokol definuje abstraktní metodu přístupu k jednotlivým hodnotám spravovaným svými uzly (hodnota senzoru, stav tlačítka, napětí generované D-A převodníkem, ale i veškeré informace jako adresa uzlu, logický typ uzlu apod.). Každá hodnota, kterou má smysl přenášet mezi jednotlivými uzly, se v terminologii ZigBee nazývá atribut. Příkladem může být atribut osvětlení OnOff představující 8-bitovou hodnotu s významy 0xFF (zapnuto), 0x00 (vypnuto) a 0xF0 (přepnout stav).

Atributy jsou organizovány do tzv. clusterů – ty mohou být dvojího druhu. Vstupní clustery organizují atributy, jejichž hodnota může být nastavena, výstupní clustery představují proměnné, jejichž hodnota může být cizím uzlem přečtena. Vstupní a výstupní clustery se sdružují do tzv. endpointů (EP), např. endpoint osvětlení, endpoint teploty, endpoint základních informací o uzlu apod. Každé zařízení pak implementuje množinu endpointů.

Popis definující skupiny clusterů a atributů spolu s jejich identifikátory a datovými typy se nazývá profil. Profil nakonec nařizuje, které atributy musí být implementovány povinně a které volitelně. Dvě ZigBee zařízení (různých výrobců) implementující daný profil pak spolu mohou komunikovat na abstraktní úrovni – např. řízení světla a dálkový přepínač.



Obr. 3.4: Architektura profilů

[7], [8]

3.2.4 Adresy uzlů v síti ZigBee

Každé zařízení v síti ZigBee je identifikováno dvěma typy adres. Prvním je globálně unikátní 64-bitová adresa linkové vrstvy (MAC), kterou má uzel zapsanou napevno (jako součást firmwaru). Druhou 16-bitová adresa síťové vrstvy, kterou zařízení obdrží po úspěšném přihlášení do sítě.

[7]

3.2.5 Zasílání zpráv

ZigBee protokol podporuje dva typy adresace zpráv. Prvním jsou tzv. přímé zprávy – umožňující zaslat zprávu přímo (bez účasti koordinátora) libovolnému uzlu. Tento proces pochopitelně zahrnuje zjištění adresy daného koncového zařízení, nebo předpokládá její znalost.

Druhou možností je vytvoření takzvané vazby zařízení (device binding) mezi endpointy (clustery) jednotlivých zařízení. V tom případě pak probíhá zjednodušená komunikace přes koordinátora, který spravuje tabulku vazeb. Komunikující zařízení pak mají mezi sebou vytvořený komunikační kanál a nepotřebují uvádět adresu ani cílový endpoint.

ZigBee protokol definuje dva typy zpráv na aplikační úrovni,

- MSG (Message, zpráva)
- a KVP (Key-Value Pair, pár klíč-hodnota).

Obě zprávy jsou asociovány s ID clusteru, ale zatímco zpráva typu KVP posílá zprávu ve striktním formátu s identifikátorem datového typu, zpráva MSG umožňuje volné formátování svého obsahu, které musí obě komunikující strany podporovat.

[7], [8]

3.2.6 Vytvoření sítě

ZigBee síť je vždy vytvořena koordinátorem. Při spušťění koordinátor hledá další koordinátory ve svém dosahu operující na jeho povolených kanálech. V závislosti na energii jednotlivých kanálů počtu sítí nalezených na daných kanálech vytvoří koordinátor svou vlastní sít a přidělí ji unikátní 16-bitový identifikátor sítě (PAN ID). Jakmile je síť vytvořena, směrovače a koncová zařízení se mohou přihlašovat.

[7]

3.3 Microchip Stack for ZigBee Protocol

Microchip Stack for ZigBee Protocol (MpZBee) je implementace ZigBee standardu pro zařízení postavená na procesorech PIC. Typická struktura takového zařízení je zobrazena níže. MpZBee je úplnou implementací všech vrstev standardu ZigBee verze 1.0, která byla ve své verzi 3.8 schválena jako ZigBee Compliant Platform (ZCP).

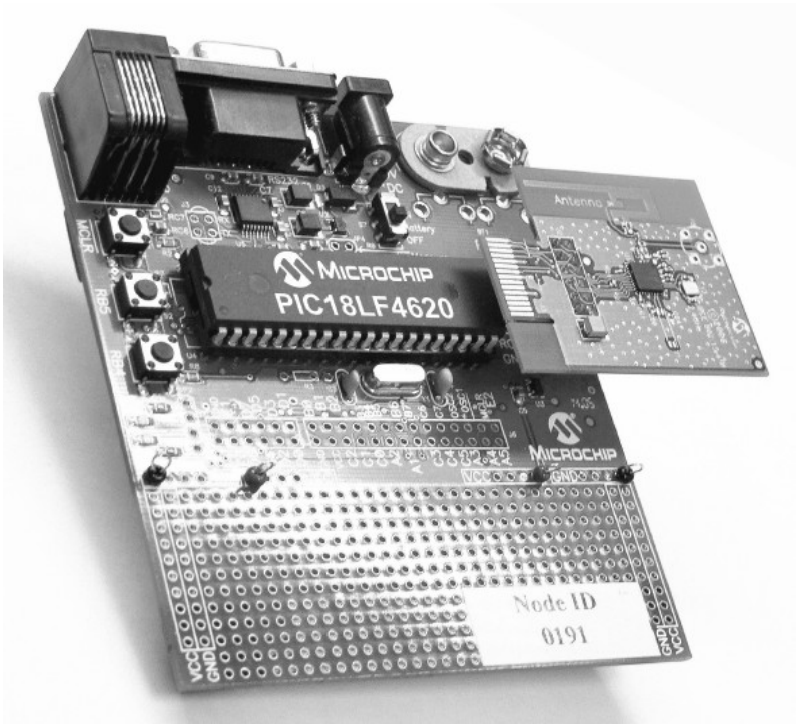
[7]

4 Návrh

Cílem je vytvořit SNMP agenta pro správu atributů ZigBee sítě. ZigBee koordinátor je připojen k PC v síti TCP/IP. Jako manažer je možné použít libovolný SNMP-kompatibilní klient, ale je nutné mu dodat popis MIB.

Je potřeba implementovat software agenta, komunikační protokol mezi koordinátorem a PC, firmware koordinátoru, komunikaci v síti ZigBee a firmware koncových uzlů.

4.1 Dostupné nástroje a technické vybavení



Obr. 4.1: Vývojová deska PICDEM Z

ZigBee síť bude realizována pomocí vývojového balíku fy. Microchip. K dispozici budu mít tři vývojové desky Picdem Z (schopné vykonávat práci jak RFD koncového uzlu, tak FFD koordinátoru či směrovače), bezdrátový analyzátor ZENA a USB programátor obvodů MPLAB ICD 2.

Z pohledu návrhu jsou zajímavá následující technická specifika desky Picdem Z:

- Mikrokontrolér PIC18LF4620 pracující na frekvenci 4 Mhz
- 2,4 GHz RF karta MRF24J40 s vestavěnou anténou
- RS232 konektor umožňující uživatelskou komunikaci desky se sériovým portem počítače
- Možnost napájení jak 9V baterií, tak ze sítě pomocí externího adaptéru (ten bohužel není součástí balíku).

- Tlačítko pro restart mikrokontroléru
- Dvě programovatelná tlačítka
- Dvě programovatelné LED diody
- Teplotní senzor TC77 s rozhraním SPI

Součástí balíku je vývojové prostředí MPLAB IDE s podporou programátoru obvodů a ladicími nástroji. K dispozici jsou dva překladače pro procesory PIC18:

- Překladač assembleru – MPASM
- Překladač jazyka C – C18

[9]

Pro vývoj firmwaru komunikujícího podle standardu ZigBee je k dispozici ZigBee protokol implementovaný v jazyce C (použitá verze) a hotové knihovny k ovládání sériového rozhraní a teplotního senzoru. Dále jsou dostupné ukázkové aplikace pro koordinátor, koncový uzel i router implementující osvětlovací a teplotní profil a demonstrující práci se ZigBee protokolem.

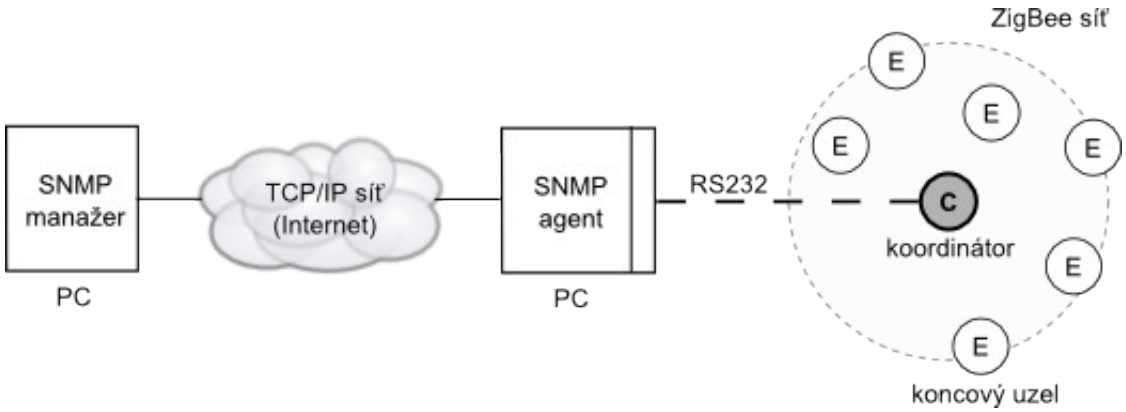
Na základě uvedených informací jsem zvolil vývoj firmwaru pro mikrokontroléry v jazyce C. Komunikace mezi ZigBee sítí a počítačem bude probíhat výhradně prostřednictvím koordinátora po sériové lince.

4.2 Umístění SNMP agenta

Zásadní úvahou při návrhu systému pro správu senzorové sítě ZigBee založeného na protokolu SNMP bylo vhodné umístění agenta.

V sítích TCP/IP se logicky a vcelku bez problémů umisťuje agent přímo na sledované zařízení, takže správa celé sítě pak spočívá v připojení k jednotlivým agentům. Tento přístup přináší výhodu v tom, že pokud se struktura sítě mění – zařízení se připojují a odpojují, veškeré komplikace s tím spojené řeší SNMP manažer a množina spravovaných objektů se u daných typů zařízení nemění – tedy implementace SNMP agenta je bezproblémová (každému OID v databázi mohou být napevno přiřazeny funkce pro získání a změnu hodnoty). S touto představou a s poměrně primitivní, ale hotovou implementací agenta v jazyce C se nabízela možnost přeložit agenta přímo pro jednotlivé ZigBee uzly. Vzhledem k poměrně jednoduché implementaci protokolu SNMPv1 v jazyce C by nebyl problém SNMP agent na dostupných vývojových deskách provozovat. SNMP manažer však komunikuje pomocí protokolu IP, proto by bylo nutné vytvořit most mezi IP a ZigBee sítěmi s mapováním adres ZigBee uzlů na IP adresy a překladem zpráv. Tento přístup se brzy ukázal jako příliš komplikovaný.

Druhá možnost je chápat celou síť ZigBee jako jedno spravované zařízení, implementovat SNMP agent na počítač v IP síti a vybavit ho rozhraním pro komunikaci se ZigBee koordinátorem pomocí vlastního, triviálního protokolu. Drobným problémem se zpočátku zdály být dynamické změny MIB při odpojování a připojování uzlů, ale ty se nakonec ukázaly snadno řešitelné pomocí vhodného návrhu MIB s použitím SNMP tabulek. Tento přístup jsem se po zvážení pro a proti rozhodl realizovat.



Obr. 4.2: Schéma SNMP systému pro správu ZigBee sítě

4.3 SNMP agent

Z důvodů zjednodušení množiny spravovaných objektů a vzhledem k nesnadným možnostem testování směrování v ZigBee síti v laboratorních podmínkách jsem se rozhodl vypustit použití směrovačů z dalšího návrhu – v takto zjednodušené ZigBee síti tedy existuje pouze jeden koordinátor a množina koncových uzlů, které jsou ke koordinátoru přímo připojeny (jedná se o topologii typu hvězda).

4.3.1 Atributy zařízení a sítě ZigBee, MIB agenta

Z atributů, které pro ZigBee síť, koordinátora a koncové uzly připadaly v úvahu jsem vybral dále uvedené. Všechny uzly jsou zařazeny pod OID .iso.org.dod.internet.experimental. U každého atributu uvádím část OID, která je definována v MIB modulu, typ objektu podle SMIv1 a omezení přístupu. Ukázka vytvořeného MIB modulu je součástí příloh (Příloha A).

Atributy sítě (picdemZ.network):

PAN ID (panID), OCTET STRING, pouze čtení
16-bitová adresa sítě, zapsaná v šestnáctkové soustavě formou řetězce
ZigBee kanál (channel), INTEGER, pouze čtení
číslo frekvenčního kanálu (11-26)
Doba od vytvoření (netUptime), TimeTicks, čtení i zápis
udává se v setinách sekundy, zápis umožňuje změnit výchozí hodnotu
Počet koncových uzlů (numNodes), INTEGER, pouze čtení

Atributy koordinátora (picdemZ.coordinator):

IEEE adresa (coordIeeeAddress), OCTET STRING, pouze čtení
Počet přijatých oktetů (coordReceivedOctets), Counter, čtení i zápis

počet bytů přijatých síťovým rozhraním, zápis umožňuje změnit výchozí hodnotu

Počet odeslaných oktetů (coordSentOctets), Counter, čtení i zápis

Atributy koncového uzlu (picdemZ.nodes.node)

Jsou organizovány formou tabulky koncových uzlů:

IEEE adresa (ieeeAddress), OCTET STRING, pouze čtení

64-bitová plná adresa uzlu, zapsána v šestnáctkové soustavě jako řetězec, byty odděleny dvojtečkou

Síťová adresa (shortAddress), OCTET STRING, pouze čtení

Typ zařízení (deviceType), INTEGER, pouze čtení

1 – Reduced Functional Device (RFD), 2 – Fully Functional Device (FFD)

Doba od připojení (uptime), TimeTicks, čtení i zápis

udává se v setinách sekundy, zápis umožňuje změnit výchozí hodnotu

Počet přijatých oktetů (receivedOctets), Counter, čtení i zápis

Počet odeslaných oktetů (sentOctets), Counter, čtení i zápis

Zdroj napájení (powerSource), INTEGER, pouze čtení

1 – síť, 2 – baterie

Světlo (light), INTEGER, čtení i zápis

1 – zapnuto, 2 – vypnuto

Teplota (temperature), OCTET STRING, pouze čtení

desetinné číslo udávající teplotu na senzoru ve °C, zapsané formou řetězce

Zdroje TRAP zpráv:

Spuštění sítě (networkStarted)

Generuje koordinátor po sestavení sítě, signalizace, že se mohou připojovat uzly

Připojení koncového uzlu (nodeJoined)

Generuje koordinátor po připojení uzlu, signalizace změny tabulky uzlů

Odpojení koncového uzlu (nodeLeft)

Generuje koordinátor po odpojení uzlu, signalizace změny tabulky uzlů

Stisknutí tlačítka na vývojové desce (buttonPushed)

Generuje příslušný uzel, součástí zprávy je identifikace uzlu

4.3.2 Požadavky na agenta

- Agent bude implementovat výše popsanou MIB.
- Musí naslouchat na UDP portu 161 a umět zpracovat příchozí příkazy typu GET, GET_NEXT a SET.
- V závislosti na dotázaném OID musí zaslat příslušnou zprávu pomocí RS232 rozhraní koordinátoru a u některých typů zpráv (GET, GET_NEXT) převzít odpověď.

- Zprávy protokolu by měly být vhodně odděleny od textových ladicích výstupů koordinátora na RS232 port – agent by měl být schopen ladicí výstupy vypsat na obrazovku.
- Zprávy přenášené přes RS232 rozhraní budou obsahovat typ, atribut, volitelně identifikaci uzlu (pokud nepůjde o identifikátor koordinátoru) a volitelně data a to minimálně typu 32-bitové číslo (pro přenos SNMP použitých typů INTEGER, TimeTicks a Counter) a řetězec proměnně délky (OCTET STRING).
- Agent musí být schopen na rozhraní RS232 očekávat kromě odpovědí na dotazy i asynchronní zprávy, na které bude reagovat zasláním příslušné SNMP TRAP zprávy manažerovi.
- Agent se musí asynchronně dozvědět o připojení a odpojení nového uzlu a vytvořit (resp. smazat) příslušný řádek v tabulce uzlů.

4.4 Koordinátor

Funkci koordinátora bude provádět vývojová deska Picdem Z s rozhraním RS232, takže bude dobré připojit k počítači pomocí tohoto sériového rozhraní.

Firmware koordinátora musí umět zpracovat výše zmíněné zprávy protokolu komunikace po sériovém rozhraní. Dále musí zvládnout zaslat odpověď, zaslat asynchronní upozornění na mimořádnou událost a umět zaslat ZigBee zprávu v případě, že je dotaz na koncový uzel. Koordinátor musí umět zpracovat asynchronní ZigBee zprávu a poslat trap do RS232.

Koordinátor musí implemetovat endpointy, clustery a atributy nutné k výměně zpráv s koncovými uzly.

V zájmu výkonu bude vhodné ukládat data o uzlech, která se po dobu připojení nemění (jako IEEE adresa, typ zařízení apod.) v koordinátoru.

4.5 Koncové uzly

Koncové uzly mají za úkol čekat na zprávy od koordinátora a zasílat odpovědi, zjišťovat a měnit své hodnoty, při nečekaných událostech (stisknutí tlačítka) zasílat asynchronní upozornění.

Koncové uzly musí, stejně jako koordinátor, implementovat vhodné endpointy, clustery a atributy odpovídající spravovaným atributům agenta.

5 Implementace

Implementace popisuje konkrétní řešení vývoje systému pro správu a sledování senzorové sítě. Skládá se z několika částí: vývoje firmwaru pro ZigBee uzly, a to ve dvou verzích, pro koordinátora a pro koncový uzel. Dále z vývoje agenta se schopností komunikovat se ZigBee koordinátorem po sériové lince a nakonec implementace vhodného nástroje pro získávání a prezentaci SNMP dat – manažera.

5.1 Firmware koordinátora

Při vývoji koordinátora jsem vycházel z demonstrační aplikace TempDemoCoord implementující koordinátora se schopností funkce (volitelné při překladu) device binding i device discovery a obsahující teplotní profil

U koordinátora bylo třeba implementovat asynchronní zachytávání zpráv přicházejících ze sériového portu, jejich dekódování, zjišťování a změna vlastních atributů, případně přeposílání dotazů na připojené koncové uzly a shromažďování statických atributů o koncových uzlech.

5.1.1 Protokol sériové linky

Protokol je binárního charakteru. Každá zpráva obsahuje povinně atribut, kterého se týká (příp. typ TRAP zprávy), následovaný typem zprávy. Následuje povinná délka zprávy (může být i nulová), volitelně index cílového uzlu z tabulky uzlů (viz dále) a volitelně přenášená data.

Hlavička zprávy je dlouhá vždy 1 oktet. 1 bit je vždy nastaven na 1 – slouží k oddělení zprávy od ladicích výpisů, které jsou tvořeny výhradně první polovinou ASCII tabulky a tedy 1. bit mají vždy nastaven na 0. Následujících 5 bitů určuje číslo atributu, kterého se zpráva týká, případně typu TRAP zprávy. Protokol je tedy schopen rozlišit pouze 32 různých atributů, což však se ukázalo jako dostatečný počet. V tabulce je uvedena pro každý použitý atribut (resp. typ TRAP zprávy) hodnota binární hlavičky (x znamená libovolnou hodnotu), symbolický název používaný ve zdrojových kódech aplikace a krátký popis.

Hodnota hlavičky	Symbolický název	Význam
100000xx	COORD_PAN_ID	PAN ID sítě
100001xx	COORD_CHANNEL	ZigBee kanál sítě
100010xx	COORD_UPTIME	Čas od spuštění sítě
100011xx	COORD_NUM_NODES	Počet koncových uzlů sítě
100100xx	COORD_IEEE_ADDRESS	IEEE adresa koordinátora
100101xx	COORD_RECVD_OCTETS	Počet přijatých oktetů koordinátora
100110xx	COORD_SENT_OCTETS	Počet odeslaných oktetů koordinátor

100111xx	COORD_NETSTART_TRAP	Upozornění na spuštění sítě
101000xx	COORD_NODE_JOINED_TRAP	Upozornění na připojení uzlu do sítě
101001xx	COORD_NODE_LEFT_TRAP	Upozornění na odpojení uzlu ze sítě
101010xx	COORD_BUTTON_TRAP	Upozornění na stisknutí tlačítka na koordinátoru
101011xx	NODE_IEEE_ADDRESS	IEEE adresa uzlu
101100xx	NODE_SHORT_ADDRESS	Síťová adresa uzlu
101101xx	NODE_DEVICE_TYPE	Typ zařízení uzlu
101110xx	NODE_UPTIME	Čas od spuštění uzlu
101111xx	NODE_RECVD_OCTETS	Přijaté oktety uzlu
110000xx	NODE_SENT_OCTETS	Odeslané oktety uzlu
110001xx	NODE_POWER_SOURCE	Typ napájení uzlu
110010xx	NODE_LIGHT	Stav diody na uzlu
110011xx	NODE_TEMPERATURE	Teplota na uzlu
110100xx	NODE_BUTTON_TRAP	Upozornění na stisknutí tlačítka na koordinátoru

Tabulka 4: Atributy protokolu sériové linky

Typ atributu dále jednoznačně určuje typ dat zprávy, resp. přítomnost dat ve zprávě jako takových a přítomnost indexu uzlu (vždy první oktet dat).

Druhou částí hlavičky je typ zprávy dlouhý 2 bity. Typ zprávy indikuje fázi komunikace, rozlišuje, zda má být hodnota atributu nastavena či zjištěna a vrácena a odlišuje asynchronní zprávy.

Hodnota hlavičky	**Symbolický název**	**Význam**
1xxxxx00	REQUEST	Žádost o zaslání hodnoty atributu
1xxxxx01	RESPONSE	Odpověď s hodnotou atributu
1xxxxx10	SET	Žádost o nastavení hodnoty atributu
1xxxxx11	TRAP	Asynchronní upozornění na událost

Tabulka 5: Typy zpráv protokolu sériové linky

Zprávu REQUEST zasílá vždy pouze počítač a koordinátor na ni okamžitě odpovídá zprávou RESPONSE (pokud do komunikace nevstoupí zpráva TRAP). Zprávu set posílá opět pouze počítač, ne tuto zprávu nepřichází žádná odpověď ani potvrzení – úspěšné nastavení zprávy může být v případě potřeby ověřeno další zprávou REQUEST. Zprávu TRAP zasílá asynchronně vždy jen koordinátor počítači a může vstoupit do libovolné fáze komunikace.

Další oktet povinně následující za hlavičkou určuje počet oktetů (délku) dat. Data zprávy mohou být tedy dlouhá 0 – 255 Bytů, což se pro potřeby aplikace ukázalo více než dostatečné.

Následující oktety obsahují zakódovaná přenášená data. V případě, že se atribut nachází na některém z koncových uzlů, je prvním oktetem dat vždy index uzlu z tabulky uzlů (viz dále).

1	atribut	typ	délka dat	data	význam
1 bit	5 bitů	2 bity	8 bitů	0 - 255 Bytů	délka

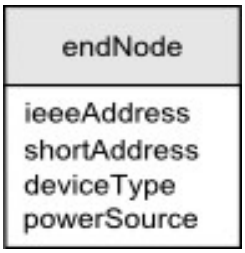
Obr. 5.1: Formát zprávy sériového rozhraní

5.1.2 Příjem zpráv

Použitá výchozí aplikace demonstrovala pouze zasílání dat z desky do počítače. Přijímání dat koordinátorem (která se mohou objevit v nečekaný okamžik), může být zajištěno dvěma způsoby. Buďto cyklickým testováním příznaku o dokončení přijetí Bytu na sériovém portu, nebo, daleko elegantněji pomocí přerušení. Bylo tedy potřeba povolit přerušení při dokončení přijetí bytu a vytvořit jeho obsluhu. Příjem zprávy pracuje jako stavový automat. Pokud je předchozí zpráva zpracována, čeká na hlavičku další zprávy (Byte s hodnotou větší než 127), ostatní Byty ignoruje. V první fázi přijímání zprávy dekóduje hlavičku a naplní proměnné atributu a typu zprávy. V další fázi zjistí a uloží délku dat. Volitelně pak načítá následující Byty do bufferu. V momentě načtení počtu Bytů odpovídajícího délce zprávy je nastaven příznak přijetí zprávy. V čase nečinnosti pak koordinátor zprávu obslouží.

5.1.3 Tabulka uzlů

Řada atributů uzlu se po dobu jeho existence v síti nemění a navíc jsou implicitně zaslány koordinátoru v čase připojení. Mimo to je nutné znát po celý čas běhu aplikace síťové adresy všech koncových uzlů kvůli dotazům. Proto je vhodné uchovávat zmíněné informace přímo v koordinátoru po celý čas připojení uzlu a zbytečně nezatěžovat síť zprávami s opakovaným dotazem (např. při opětovném dotazu na tabulku uzlů SNMP agenta). Informace o uzlech je organizována jako pole struktur. Index v poli struktur koresponduje s indexem řádku v tabulce uzlů agentovy MIB.



Obr. 5.2: Statické informace o koncovém uzlu

5.1.4 Zjišťování atributů koncových uzlů

V případě, že koordinátor obdrží dotaz na atribut některého z uzlů, získá identifikaci uzlu v podobě indexu z tabulky uzlů. Pro zjištění, resp. nastavení daného atributu je třeba zaslat ZigBee zprávu GETACK, resp. SET na příslušný uzel, jehož síťovou adresu zjistíme z tabulky uzlů.

5.1.5 Profil koordinátora

Na základě vybraných atributů bylo třeba v uzlech sítě implementovat následující endpointy a jejich atributy:

- EP_ZDO – tento endpoint implementuje povinně každé zařízení, slouží ke získávání atributů
  - IEEE adresa,
  - síťová adresa,
  - typ zařízení
  - a typ napájení
- EP_SNMP – uživatelský endpoint organizující atributy
  - doba spuštění,
  - počet přijatých oktetů,
  - počet odeslaných oktetů
  - upozornění na stisknuté tlačítko
- EP_LIGHT uživatelský endpoint s atributem
  - světlo
- EP_TEMPERATURE
  - teplota

5.2 Agent

Agent se dělí na dvě hlavní části – část zabezpečující komunikaci s koordinátorem po sériovém portu a část zodpovědná za komunikaci se SNMP manažerem a práci s databází spravovaných objektů.

Jako implementační platformu jsem se rozhodl použít Java SE. Získal jsem tak všechny výhody vyspělého, objektového jazyka Java, zejména rychlost vývoje, výbornou modularitu řešení a lepší možnost budoucího rozšíření, a v neposlední řadě také multiplatformní produkt. Významnými důvody pro tento výběr byly také existující knihovny pro SNMP protokol a komunikaci po sériovém portu.

5.2.1 Sériové rozhraní

Pro implementaci sériového rozhraní jsem použil knihovnu RXTX.org (2.1.7r2). Java sice obsahuje standardní knihovnu pro tyto účely, ale ta není dostupná pro platformu Windows. Knihovna zabezpečuje vytvoření objektů tříd *InputStream* a *OutputStream*, představujících standardní vstupní a

výstupní proud, do kterého je možné zapisovat (resp. číst) data a ta jsou na fyzické úrovni ve správné podobě zapsána na sériový port.

Jádrem této části je třída ***Message*** představující zprávu sériového portu. Tato třída implementuje výše popsaný protokol sériového rozhraní. Dále implementuje metodu *send*, která zapíše správně zakódovanou zprávu do dříve zmíněného výstupního proudu.

Další významnou třídou je ***MessageReader***, jejíž instance běží po celý čas spuštění agenta v samostatném vlákně a provádí blokující čtení dat ze vstupního proudu (sériového portu). Instance čte data Byte po Bytu a čeká na hlavičku zprávy, všechny ostatní Byty chápe jako textové ladicí výpisy koordinátora a jako takové je vypisuje na obrazovku. V momentě detekce zprávy je tato zpráva přečtena, je vytvořen objekt třídy *Message*. Pokud je zpráva typu TRAP, zavolá se příslušná metoda třídy *TrapSender* (popsána dále). Pokud se jedná o upozornění na připojení, či odpojení koncového uzlu, je zavolána příslušná metoda třídy MIB (popsána dále), která vytvoří, resp. zruší řádek v tabulce uzlů. V ostatních případech je zpráva předána jako odpověď čekající metodě send.

5.2.2 Rozhraní SNMP

Pro implementaci sériového rozhraní jsem použil knihovnu JoeSNMP (verze 0.3.4). Tato knihovna bohužel neobsahuje nástroje pro generování šablony zdrojového kódu agenta z popisu MIB modulu, jak je to běžné např. u knihoven balíku net-snmp pro jazyk C. Při malém rozsahu MIB mého agenta však byl tento nedostatek překonatelný. Knihovna implementuje zejména datové typy SNMP a metody pro zakódování zprávy podle pravidel BER.

Pro SNMP rozhraní jsou klíčové tyto třídy:

Třída ***MIB*** implementuje databázi spravovaných objektů formou červeno-černého binárního vyhledávacího stromu. Implementuje metodu pro naplnění MIB objekty a vytváření a rušení konceptuálních řádků tabulky uzlů. Každý spravovaný objekt je představován objektem třídy daného SNMP typu, které jsou v čase vkládání do MIB přepsány metody pro čtení a zápis hodnoty. Tyto metody při zavolání vytváří příslušný dotaz – objekt třídy *Message*, volají metodu *send* a volitelně vrací hodnotu získanou z odpovědi.

Třída ***Agent*** naslouchá na UDP portu 161, očekává a zpracovává SNMP příkazy a pracuje s objekty stromu MIB.

Třída ***TrapSender*** implementuje metodu pro zaslání zprávy TRAP na UDP port 162 nastaveného hosta.

5.3 Manažer

Vzhledem k použití otevřeného protokolu SNMPv1 bylo možné použít libovolný agent podporující tuto verzi. Po většinu času jsem používal multiplatformní produkt iReasoning Mib Browser, který je vzhledem ke svým schopnostem pracovat s popisem MIB modulu, grafickým uživatelským rozhraním, podporou vizuální prezentace SNMP tabulek a přítomností přehledného přijímače TRAP zpráv dobrou volbou.

Samozřejmě je možné použít např. nástroje pro příkazovou řádku ze známého balíku net-snmp, či složitější, zejména komerční systémy s pokročilou prezentací ve formě grafů apod.

5.4 Problémy a návrhy na vylepšení

Koncové uzly se ve chvílích nečinnosti periodicky přepínají do úsporného režimu, ve kterém vypínají síťový adaptér. Po uplynutí periody, během které se zprávy řadí do fronty v koordinátoru, zjišťují, zda na ně nějaké zprávy nečekají a ty případně zpracují.

V navrženém přístupu, kdy jsou atributy jednotlivých uzlů organizovány v SNMP tabulce, může docházet (při výpisu celé tabulky) poměrně často k obnovování všech hodnot tabulky. V situaci, kdy se každá hodnota zjišťuje zvlášť příkazem GETNEXT, čeká se na odpověď koncového uzlu a následující reakci koordinátora na sériovém portu a až poté se stejným způsobem dotazuje následující hodnota, může trvat zjištění tabulky příliš dlouhou dobu. V současném návrhu tedy trvá zjištění jedné hodnoty až celou úspornou periodu. Při běžné délce periody nad 1 sekundu, větším počtu uzlů a sledovaných atributů je už prodleva neúnosná. Tuto prodlevu se podařilo snížit eliminací opakovaných dotazů u statických atributů, ale v případě implementace dalších sledovaných atributů se problém vrátí. S tímto chováním také souvisí fakt, že v momentě načtení budou už některé informace delší dobu neaktuální a není tedy možné získat konzistentní snímek sítě.

Řešením by mohlo být zjištění všech atributů uzlu v rámci jedné zprávy, avšak stále zůstává problém prodlevy u každého připojeného zařízení

5.4.1 Návrhy na rozšíření:

- Implementovat podporu správy vazeb zařízení (device bindings) a zařazení tabulky vazeb do MIB agenta.
- Čistě při správě sítě ZigBee se nepočítá se čtením hodnot senzorů apod., tedy vykonáváním prvotních funkcí koncového zařízení pomocí protokolu SNMP, ty byly v rámci této práce implementovány spíše demonstračně, každý koncový uzel sítě by ale mohl mít navíc sledovatelnou tabulku endpointů, clusterů a atributů s počty zpráv adresovaných danému clusteru apod.
- Navrhnout a implementovat podporu směrovačů v obecné mesh síti ZigBee a jejich správy.
- Implementovat vyšší verzi protokolu SNMP a zaměřit se na otázky bezpečnosti.

6 Závěr

Práce se zabývala analýzou a implementací systému pro správy a sledování obecné ZigBee sítě. Praktická část představovala splnění následujících dílčích úkolů:

- Vytvoření MIB modulu – definice spravovaných objektů ZigBee sítě pomocí SMI,
- implementace SNMP agenta na straně PC v jazyce Java,
- návrh protokolu pro přenos spravovaných informací mezi PC a sítí ZigBee
- a implementace firmwaru vývojových desek PICDEM Z pro koordinátora a koncové uzly sítě ZigBee (formou modifikace hotových příkladů v jazyce C).

Výsledný produkt není zralý k praktickému, ostrému nasazení při správě senzorové sítě, nicméně může velice dobře posloužit jako demonstrace přístupu k této problematice a studijní materiál pro vývojáře podobných produktů.

Řešení na straně senzorů není zcela obecné, aplikaci lze nasadit na procesorech typu PIC a je závislá na implementaci ZigBee protokolu od fy Microchip. Některé části jsou navíc závislé na technických specifikách vývojové desky PICDEM Z. I přesto by se firmware dal s výhodou použít jako základ pro implementaci na jiné platformě umožňující vývoj v jazyce C.

Software na straně počítače je na druhou stranu vysoce modulární, například změna komunikace s koordinátorem ze sériového portu na jinou technologii by nevyžadovala žádné rozsáhlé změny.

Hlavním přínos práce vidím v nutnosti zorientovat se hned v několika okruzích použité problematiky. Bylo nutné nastudovat a zvládnout protokol SNMP, síťovou komunikaci TCP/IP, práci se sériovým portem a s tím související kódování různých typů dat do proudu Bytů. Další velkou kapitolou bylo programování mikrokontroléru PIC v Jazyce C, nezbytné zvládnutí problematiky přerušení a dalších. Nakonec se nebylo možné vyhnout podrobnému nastudování protokolu ZigBee a ovládnutí práce s jeho implementací od fy Microchip. Práce mě obohatila o řadu praktických schopností v uvedených oblastech.

Práce ponechává některé oblasti problematiky, jako je směrování v sítích ZigBee a bezpečnost protokolu SNMP neřešené. Některé další části by bylo možné rozšířit a zdokonalit. Na práci by neměl být problém vhodným způsobem navázat.

Literatura

[1] Mauro, D., Schmidt, K.: Essential SNMP, O'Reilly, červenec 2001.

[2] Wikipedia, The Free Encyclopedia: Simple Network Management Protocol, květen 2008. Dokument dostupný na URL http://en.wikipedia.org/wiki/Simple_Network_Management_Protocol (květen 2008).

[3] Bruey, D.: SNMP: Simple? Network Management Protocol, Rane Corporation, 2005. Dokument dostupný na URL http://www.rane.com/pdf/ranenotes/SNMP%20Simple%20Network%20Management%20Protocol.pdf (květen 2008).

[4] Rose, M., McCloghrie, K.: Structure and Identification of Management Information for TCP/IP-based Internets, květen 1990. Dokument dostupný na URL http://tools.ietf.org/html/rfc1155 (květen 2008).

[5] McCloghrie, K., Rose, M.: Management Information Base for Network Management of TCP/IP-based internets: MIB-II, březen 1991. Dokument dostupný na URL http://tools.ietf.org/html/rfc1213 (květen 2008).

[6] Vojáček, A.: ZigBee - novinka na poli bezdrátové komunikace, www.hw.cz, červen 2005. Dokument dostupný na URL http://hw.cz/Rozhrani/ART1299-ZigBee---novinka-na-poli-bezdratove-komunikace.html (květen 2008).

[7] Microchip Technology: Microchip Stack for the ZigBee™ Protocol, leden 2007. Dokument dostupný na URL http://ww1.microchip.com/downloads/en/AppNotes/00965c.pdf (květen 2008).

[8] ZigBee Alliance: ZigBee Specification v 1.0, leden 2008. Webová stránka www.zigbee.org.

[9] Microchip Technology: PICDEM Z Demonstration Kit User's Guide, leden 2007. Dokument dostupný na URL http://ww1.microchip.com/downloads/en/DeviceDoc/51524b.pdf (květen 2008).

[10] Mrázek, O.: SNMP protokol a jeho využití, www.hw.cz, září 2003. Dokument dostupný na URL http://hw.cz/Produkty/ART957-SNMP-protokol-a-jeho-vyuziti.html (květen 2008).

[11] Case, J., Fedor, M., Schofstall, M., Davin, J.: A Simple Network Management Protocol (SNMP), květen 1990. Dokument je dostupný na URL http://tools.ietf.org/html/rfc1157 (květen 2008).

Seznam použitých zkratek

ASN.1	Abstract Syntax Notation One
CRC	Cyclic Redundancy Check
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
DSSS	Direct Sequence Spread Spectrum
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISO	International Organization for Standardization
MAC	Media Access Control
MIB	Management Information Base
MO	Managed Object
MpZBee	Microchip Stack for the ZigBee Protocol
OID	Object Identifier
O-QPSK	Offset Quadrature Phase-Shift keying
OSI	Open Systems Interconnection
PAN	Personal Area Network
PC	Personal Computer
PHY	Physical Layer
RAM	Random Access Memory
RS232	Recommended Standard 232
SMI	Structure of Management Information
SNMP	Simple Network Management Protocol
TCP/IP	Transmission Control Protocol / Internet Protocol
UDP	User Datagram Protocol
ZCP	ZigBee Compliant Platform

Seznam příloh

Příloha A. MIB modul PICDEMZ-NETWORK-MIB-V1
Příloha B. Instalace, konfigurace a spuštění agenta
Příloha C. DVD se zdrojovými kódy

Přílohy

Příloha A. Vybrané části MIB modulu PICDEMZ-NETWORK-MIB-V1

Modul popisuje vybrané atributy sítě ZigBee a jejích uzlů. Pro demonstrační účely navíc obsahuje některé atributy specifické pro vývojovou desku PICDEM Z. Agent vytvořený v rámci praktické části práce tento modul implementuje, dále je použit pro automatickou konfiguraci manažera. Následují ukázky klíčových sekcí modulu. Kompletní modul je součástí přiloženého CD (Příloha C).

Hlavička modulu a sekce IMPORTS:

```
PICDEMZ-NETWORK-MIB-V1 DEFINITIONS ::= BEGIN
     IMPORTS
          OBJECT-TYPE
               FROM RFC-1212,
          TimeTicks, Counter
               FROM RFC1155-SMI;
```

Zařazení modulu v rámci globálního stromu:

```
     picdemZ OBJECT IDENTIFIER ::= {experimental 1}
     network OBJECT IDENTIFIER ::= {picdemZ 1}
     coordinator OBJECT IDENTIFIER ::= {picdemZ 2}
```

Definice spravovaného objektu *panID*:

```
     -- 1.3.6.1.3.1.1.1.0
     panID OBJECT-TYPE
          SYNTAX OCTET STRING
          ACCESS read-only
          STATUS mandatory
          DESCRIPTION
               "16-bit ZigBee network address"
          ::= {network 1}
```

Definice tabulky koncových uzlů:

```
     -- objekt tabulky
     nodes OBJECT-TYPE
          SYNTAX SEQUENCE OF Node
          ACCESS not-accessible
          STATUS mandatory
          DESCRIPTION
               "Table of currently connected nodes"
          ::= {picdemZ 3}

     -- struktura řádku tabulky
```

```
Node ::=
      SEQUENCE {
            nodeIndex
                  INTEGER,
            ieeeAddress
                  OCTET STRING,
            shortAddress
                  OCTET STRING,
            deviceType
                  INTEGER,
            uptime
                  TimeTicks,
            receivedOctets
                  Counter,
            sentOctets
                  Counter,
            powerSource
                  INTEGER,
            light
                  INTEGER,
            temperature
                  OCTET STRING
      }

-- objekt řádku
node OBJECT-TYPE
      SYNTAX Node
      ACCESS not-accessible
      STATUS mandatory
      DESCRIPTION
            "ZigBee node"
      INDEX {index}

      ::= {nodes 1}
```

Definice dvou zpráv typu TRAP:

```
networkStarted TRAP-TYPE
      ENTERPRISE picdemZ
      DESCRIPTION
            "Network established"
      ::= 1

nodeJoined TRAP-TYPE
      ENTERPRISE picdemZ
      DESCRIPTION
            "New node joined network"
      ::= 2
```

Příloha B. Překlad, konfigurace a spuštění agenta

Zdrojové soubory agenta jsou ve tvaru projektu vývojového prostředí NetBeans IDE 6.1 (http://www.netbeans.org/). V kořenovém adresáři projektu se dále nachází soubor build.xml pro automatický překlad pomocí nástroje Ant (http://ant.apache.org/).

Výsledkem překladu zdrojových souborů agenta je spustitelný JAR archiv snmpzbee.jar v adresáři /dist. Ten ke svému chodu potřebuje dynamickou knihovnu pro práci se sériovým rozhraním daného operačního systému součástí přiloženého DVD je verze pro systém Windows rxtxSerial.dll (aktuální verze pro podporované OS na http://www.rxtx.org/).

Aplikace očekává tři argumenty příkazové řádky:

1. Textový identifikátor sériového portu. Na systémech Windows např. COM1, COM2 apod., na systémech Linux /dev/ttyS0 apod..
2. Síťovou adresu nebo jméno příjemce TRAP zpráv, např. localhost, 192.168.2.15 apod.
3. Číslo cílového UDP portu pro zasílání TRAP zpráv.

Příklad spuštění aplikace:

```
java -jar snmpzbee.jar COM1 localhost 162
```

Příloha C. DVD se zdrojovými kódy

Přiložený nosič obsahuje zdrojové a binární podoby vytvořených aplikací:

- zdrojové kódy agenta a přeložený spustitelný archiv JAR
- zdrojové kódy firmwaru pro koordinátora a koncové uzly včetně použité verze MpZBee
- MIB modul PICDEMZ-NETWORK-MIB-V1 pro konfiguraci SNMP manažerů

Součástí přílohy je i tato technická zpráva.