VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

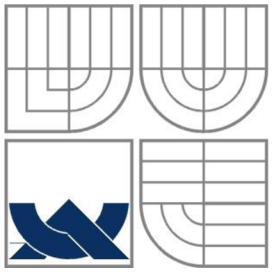
SBĚR DAT A KOMUNIKACE PROTOKOLEM 89

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

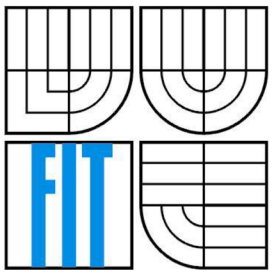
AUTOR PRÁCE
AUTHOR

MARTIN KÖHLER

BRNO 2011



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

SBĚR DAT A KOMUNIKACE PROTOKOLEM 89

DATA COLLECTION AND COMMUNICATION VIA PROTOKOL 89

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

MARTIN KÖHLER

VEDOUCÍ PRÁCE
SUPERVISOR

ING. VLASTIMIL KOŠAŘ

BRNO 2011

Abstrakt

Tato bakalářská práce se zabývá úpravou routeru Edimax BR-6104KP pro potřeby firmy Condata s.r.o. V prvních dvou kapitolách je popis firemních bezdrátových modulů a jejich komunikačního protokolu. Ve třetí kapitole se práce zabývá původním komunikačním řešením, následně pak výběrem jeho náhrady. V další části je popis hardwarových a softwarových modifikací routeru Edimax. V poslední části je popsán návrh a implementace vlastních aplikací. Závěrem práce je zhodnoceno nasazení v reálném prostředí.

Klíčová slova

edimax, br-6104kp, openwrt, midge, amilda, squidge, UCB, protokol 89

Abstract

This Bachelor degree thesis deals with Edimax BR-6104KP router modifications developed for purposes of Condata s.r.o. The first two chapters describe the company‘s wireless modules and their communication protocol. The third chapter deals first with Condata‘s curerent communications set up and subsequently with possible alternatives. The following section describes software and hardware modifications of the router under investigation. The last section describes inovative applications and their implementaion. The thesis is concluded with a real life operation evaluation.

Keywords

edimax, br-6104kp, openwrt, midge, amilda, squidge, UCB, protokol 89

Citace

Köhler Martin: Sběr dat a komunikace Protokolem 89, Brno, bakalářská práce, FIT VUT v Brně, 2011

Sběr dat a komunikace Protokolem 89

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Vlastimila Košaře. Další informace mi poskytl Ing. Jiří Nitsche, ředitel firmy Condata s.r.o. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Jméno Příjmení
Datum

Poděkování

Chtěl bych poděkovat panu Ing. Vlastimilu Košařovi a panu Ing. Jiřímu Nitschemu za to, že mi věnovali svůj čas při konzultacích a odborném vedení práce.

Obsah

Úvod

Téma, které zpracovává tato bakalářská práce je zadané ve spolupráci s firmou Condata s.r.o. Jednou z činností, kterou se firma zabývá, jsou Inteligentní domy. Inteligentní domy jsou aplikace, které s pomocí sítě měřících, akčních a komunikačních prvků, počítače a programového vybavení dokážou rychle, předvídavě a účelně řídit požadované činnosti v domě z jednoho centra. Navíc mohou zpracované údaje přenést na libovolně vzdálené místo.

Tato práce se zaměřuje na popis komunikačního protokolu mezi jednotlivými prvky inteligentního domu. Po seznámení s protokolem se nadále věnuje historii dříve používaných komunikačních prvků a jejich náhradě za novější, které splňují požadavky na moderní zařízení. Do těchto nových zařízení lze přesunout i část řídící logiky, a tím postupně vytvářet systém distribuované inteligence.

V první kapitole práce je popis dostupných modulů, s kterými jsem se během práce setkal. Následující kapitola popisuje komunikační protokol užívaný firmou Condata. Třetí část se věnuje původnímu řešení pro připojení modulů k internetu a důvodům, proč je toto řešení pro firmu nevýhodné. V druhé polovině třetí kapitoly se věnuje výběru řešení nového. Následující dvě kapitoly popisují hardwarové a softwarové modifikace celého zařízení. Šestá a sedmá kapitola pak návrh a implementaci aplikací, které umožní modulům připojení k internetu a sběr dat z těchto modulů. V poslední kapitole je popis, uložení konfigurace zařízení. Závěrem práce popisuje nasazení zařízení v ostrém provozu.

1 Moduly firmy Condata s.r.o.

Firma nabízí moduly různých typů. Jako příklad lze uvést moduly pro měření nejrůznějších elektrických i neelektrických veličin jako například teploty a vlhkosti, dále pak moduly určené pro spínání externích periferií, moduly pro regulaci a podobně. Pro popis modulů a jejich komunikačního protokolu jsem čerpal z interních dokumentů firmy Condata.

Jádrem těchto modulů je procesor firmy Nordic Semiconductor, nRF9E5[1], mezi jehož hlavní výhody patří zejména vestavěný vysílač/přijímač v pásmu 433/868MHz a načítání programu z připojené SPI EEPROM, což umožňuje změnu programu uloženého v externí EEPROM za běhu.

Společnou vlastností všech modulů je možnost bezdrátové komunikace ve volném pásmu 866MHz a velmi snadné přidávání dalších modulů do bezdrátové sítě. Pro komunikaci s počítačem, nebo jiným řídícím prvkem jsou moduly vybaveny linkou RS232-3V. Použitím vhodného převodníku je tedy možné připojit do bezdrátové sítě i PC.

Maximální povolený výkon vysílaček pro datový přenos v pásmu 866MHz je 0,5W. Moduly Condata využívají pouze 10mW, což postačuje s rezervou pro veškerou komunikaci u procesů měření a řízení. Jedinou podmínkou při stavbě bezdrátové sítě je, aby na sebe alespoň dva sousední moduly „viděly“. Maximální počet modulů v jedné síti je 240. Rozsáhlejší objekty je potřeba pokrýt větším množstvím bezdrátových sítí, kde každou z nich je možné připojit k internetu. Počet těchto sítí je prakticky neomezený.

Aby zařízení mohla plně využívat všech možností komunikace, které jsou v síti modulu Condata dostupné, musí spolu komunikovat standardizovaným způsobem pomocí User Control Block (UCB) a po sériové lince protokolem 89, který na sériové lince tvoří pro UCB vnější obálku a slouží pro zabezpečení přenosu po sériové lince pomocí kontrolních cyklických redundantních součtů neboli CRC.

V bezdrátové síti zajišťuje toto zabezpečení samotný nRF9E5, který hardwarově počítá čtyřbytové CRC.

1.1 User Control Block

User Control Block (dále jen UCB) je prostředkem pro univerzální přístup k různým zařízením. Pro komunikaci je zapotřebí dotazované aplikaci odeslat požadavek na provedení operace, popřípadě tento požadavek doplnit potřebnými parametry a je-li zapotřebí, pak i daty. Odpovědí je kód z předem definované množiny stavů, který informuje, jak daná operace skončila. V návratových parametrech a datech se vrací požadovaná informace.

UCB vlastně "normalizuje" komunikaci mezi jednotlivými zařízeními v síti. Pro tato normalizovaná zařízení se používá pojmenování Logická Periferie. Každý program, který zajišťuje

obsluhu libovolné periferie, například teplotního čidla a převádí výsledek na jednotný formát pro další zpracování nadřízeným systémem.

Výhoda normalizované komunikace je zřejmá. V praxi se používá celá řada teplotních čidel. Například čidla termistorová, KTY, PT100, DS18B20, SHT11 a další. Každé čidlo potřebuje jiný hardwarový interface a jinou obsluhu vlastním driverem. Tím, že všechny drivery dodržují striktně rozhraní UCB, je možno v nadřízené vrstvě komunikovat se všemi typy teploměrů naprosto stejně, což s sebou přináší jednoznačné výhody. Struktura UCB je patrná z tabulky 1.1.

Byte	Obsah	Popis
0	odes	adresa odesilatele UCB
1	prij	adresa příjemce UCB
2	LP	adresa logické periferie
3	RQ/ST	číslo v dané logické periferii/informace o stavu požadované operace
4	L_DATA	počet datových bytů obsažených v UCB
5	L_PARAM	počet parametrů v UCB
	PARAM	parametry
	DATA	Data
	crc	CRC celého UCB

Tabulka 1.1: Struktura UCB

1.1.1 Obecná pravidla pro práci s UCB

Aby byl splněn požadavek na univerzálnost USB, musí být pří vývoji dodržena základní pravidla:

- Pokud je LP = 0x00, jedná se o systémovou LP, která slouží pro zjištění stavu zařízení, pro změnu programu za běhu nebo pro restart zařízení, popřípadě další systémové funkce.
- RQ obsluhující operace zápisu mají liché číslo, RQ určené pro čtení hodnoty mají číslo sudé.
- Pokud při zpracování operace došlo k chybě, je nejvyšší bit v RQ/ST nastaven na 1.
- Pokud se v datech přenáší více hodnot, je použito stejné či větší množství parametrů udávající délky jednotlivých částí přenášených v datové části UCB. Pokud se přenáší pouze jedna informace v datové části, je doporučené uvést délku této informace v parametrech, avšak není to vyžadováno.

1.2 Logická periferie

Každé fyzické zařízení se skládá z operačního systému a z několika logických periferií. V každém modulu je naimplementován operační systém, který se stará o přidělování strojového času jednotlivým logickým periferiím.

V nejjednodušším případě u modulů s nRF9E5 je to operační systém reálného času nazývaný FIOS. Tento systém obsahuje čtyři základní logické periferie. Systémová LP = 0x00 slouží ke správě daného fyzického zařízení, v kterém je implementována. Tato LP nabízí možnost zjištění konfigurace zařízení, změnu programu uloženého v externí EEPROM zařízení, přístup k interní XRAM, nebo restart daného zařízení. U pokročilejších zařízení s komplexním operačním systémem je možné prostřednictvím této LP přenášet celé soubory a plnohodnotně pracovat s nainstalovaným operačním systémem.

Dále pak FIOS obsahuje LP = 01, která zajišťuje komunikaci RF, LP = 02, která se stará o obsluhu časovačů a LP = 03, která zajišťuje komunikaci na sériovém kanále.

Vlastní aplikace, například zmíněné měření teploty, zajišťují LP s pořadovým číslem 04 a vyšším. Každá LP může rozpoznávat 64 zápisových a 64 čtecích příkazů a vracet až 64 pomocných stavových hlášení o správně proběhnuvší operaci a 64 unifikovaných hlášení o případné chybě.

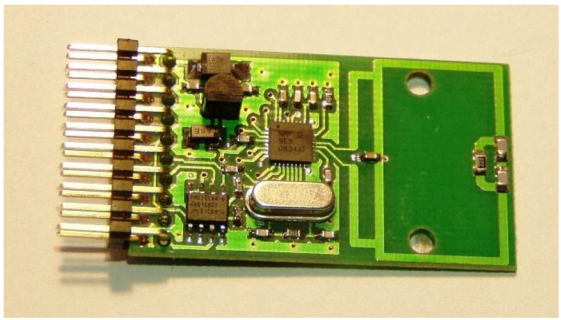
Z výše uvedeného je patrné, že množina RQ a LP poskytuje dostatečný prostor pro realizaci v podstatě jakékoliv aplikace.

1.3 Použité moduly

V rámci práce jsem se seznámil s následujícími moduly, které firma nabízí.

1.3.1 Univerzální modul

Univerzální modul (dále také UM) obsahuje na své desce pouze procesor a paměť EEPROM, piny procesoru jsou vyvedeny na konektor. Podle programu v EEPROM pak může UM zastávat různé funkce. Základní deska UM je na obrázku 1.1. Následující podkapitoly představují příklady možného využití univerzálního modulu.



Obrázek 1.1: Univerzální modul

1.3.1.1 RF AP

RF AP je jednou z aplikací pro univerzální modul. Tento modul může být připojený buď k PC, nebo k jinému zařízení, které potřebuje komunikovat s dalšími bezdrátovými moduly. Obsahuje FIOS se systémovou logickou periferií, dále pak LP zajišťující komunikaci po sériové lince a LP pro přenos RF. Po hardwarové stránce je tento modul připojen k PC přes převodník USB/RS232-3V nebo k jiným zařízením přes převodník na RS485.

1.3.1.2 Ovladač pro kotel

Ovladač pro elektrokotel je modul, který funguje jako termostat a řídí vypínaní a zapínání triakového ovládání elektrokotle. K tomuto modulu jsou připojena dvě čidla DS18B20, kde jedno snímá teplotu topné vody a druhé měří venkovní teplotu pro potřeby případného řízení ekvitermou.

Implicitně se po zapnutí teplota topné vody reguluje na 50°C, pro její změnu je potřeba poslat z nadřízeného počítače UCB s novou teplotou. Toto UCB se posílá na LP = 0x04 a RQ = 0x09. V UCB se musí poslat dva parametry, ve kterých je požadovaná teplota v setinách stupně a v pořadí Lo Hi. Pokud tedy chceme nastavit například teplotu 45°C, musíme v parametrech UCB odeslat 0x94 0x11.

1.3.1.3 Point-to-Point

V Point-to-Point zapojení se používají dva UM, které spolu bezdrátově komunikují. Data přijatá na sériové lince se předávají na druhé zařízení a odesílají se na jeho sériovou linku. Takto lze do bezdrátové sítě zapojit i zařízení, které komunikuje sériově a nemá bezdrátový modul.

O nastavení parametrů sériové linky se stará RQ = 0x03 na LP = 0x04. RQ má jeden povinný parametr, v kterém se nastavují parametry přenosu a jeden nepovinný parametr, ve kterém se nastavuje adresa druhé strany přenosu Point-to-Point spojení.

1.3.2 Termoreg

K regulaci teploty v jednotlivých místnostech se používají moduly termoreg. Tyto moduly ovládají podle nastavené teploty hlavice na radiátorech v místnosti. Místo hlavice může být připojen i jiný aktivní prvek, který může ovládat topení elektrické.

K termoregu je připojeno analogové teplotní čidlo, které snímá teplotu v místnosti. Podle této teploty se buď otevírá, nebo zavírá hlavice, což řídí akční člen této LP. Teplotní čidlo je připojené ke kanálu 0. Po zapnutí se implicitně reguluje na teplotu 23°C. Tuto teplotu je možné změnit stejným UCB, jako u ovladače pro kotel. Zde může být navíc odeslán třetí parametr, který určuje číslo kanálu, ke kterému je připojeno teplotní čidlo. Pokud tento parametr odeslán není, je nastaven kanál 0.

Termoreg umožňuje měřit současně dva analogové teploměry typu KTY. Druhý teploměr je možno využít například pro měření venkovní teploty pro účely nadřazení ekvitermní regulace. Namísto teploměru je možno připojit kontakt pro sledování otevření okna a omezení úniku tepla při větrání.

Ke zjištění teplot na obou kanálech a stavu akčního členu je na LP = 0x04 implementován RQ = 0x04, který má jeden parametr udávající číslo kanálu, ke kterému je připojen teploměr. Tento RQ vrací v odpovědi dva byty dat obsahující teplotu v setinách stupně Celsia v pořadí Lo Hi.

2 Protokol 89

Protokol 89 je jedním z komunikačních protokolů, který používá firma Condata s.r.o. pro zabezpečení komunikace prostřednictvím UCB na sériové lince.

Protokol pracuje systémem dotaz – odpověď, nezávisle na tom, zda se komunikuje mezi dvěma procesy na jednom PC, nebo mezi dvěma zařízeními. Odpovědí na jeden dotaz může být předem určený počet a mohou přijít v předem známem čase. Pokud přijdou později, jsou ignorovány.

Pokud zařízení koná nějakou činnost, která vyžaduje dlouhý čas pro zjištění aktuálních dat, například komunikuje s pomalými periferiemi (některá teplotní čidla), pak data od těchto periferií sbírá do své vyrovnávací paměti. Jakmile se na zařízení obrátí nadřazený prvek dotazující data, tak mu zařízení vrátí aktuální data z této vyrovnávací paměti. Nutnou podmínkou je, aby byla tato data opatřena stavovým bitem, který určuje platnost dat.

Tuto filozofii je nutné dodržovat na všech úrovních komunikace nezávisle na použitých komunikačních prostředcích.

V rámci protokolu je vyhrazeno několik cílových adres. Adresa 0x00 a 0xff slouží jako adresy, kterým rozumí všechna připojená zařízení. Adresy 0x68 a 0x69 se využívají jako servisní adresy, nebo jako adresy pro spojení point-to-point při RF komunikaci. Adresy v intervalu 0xf0 až 0xfe jsou vyhrazeny pro zařízení, která komunikují přes TCP sockety a slouží k připojení dalších zařízení do sítě.

Tabulka 2.1 popisuje hlavičku paketu Protokolu 89. Po hlavičce následuje datová část, která může obsahovat UCB. Datová část je zakončena bytem, obsahujícím kontrolní součet těchto dat.

Byte	Obsah	Popis
0	0x89	signatura a identifikace začátku paketu
1	0x56	
2	src_addr	adresa odesilatele
3	dest_addr	adresa příjemce
4	relay_info	určení způsobu předávání paketu a identifikuje vícenásobné připojení se stejnou src_addr k jednomu zařízení
5	id	id paketu, inkrementuje se při odeslání nového paketu, při odpovědi zůstává stejné
6	packet_type	typ paketu
7	payload_size	velikost datové části paketu (size)
8	crc	CRC hlavičky paketu

Tabulka 2.1: Hlavička protokolu 89

2.1 Parametry komunikace

Protokol 89 lze použít nejen při komunikaci po lince RS-485 a pomocí bezdrátového spojení, ale i při komunikaci přes internet protokolem TCP.

2.1.1 Parametry komunikace přes RS-485

Pro komunikaci na lince RS-485 se využívá přenosové rychlosti 57600Bd. Ve formátu 8 datových bitů, jeden stop bit a žádná parita.

2.1.2 Komunikace přes TCP

Při spojení přes TCP se nekomunikuje přímo mezi dvěma body, ale využívá se prostředník. Tento prostředník se nazývá Datová ústředna (dále jen DU). Datová ústředna je aplikace vyvíjená firmou Condata. A je spuštěna na serveru, který disponuje veřejnou IP adresou a standardně naslouchá na portu 10000 pro Protokol 89. DU mohou být propojeny ve stromové struktuře. Pro tohle propojení se využívá implicitně portu 30000.

Každý bod, který poskytuje služby pro další body, je nutné na DU zaregistrovat. Také každý bod, který chce tyto služby využívat, musí poslat datové ústředně požadavek na registraci cílového bodu. Bod, který pouze využívá služeb bodu jiného, se může registrovat pod svým jménem, ale není to vyžadováno – dostane implicitně přidělené jméno složené z názvu dané DU a pořadí, ve kterém se na DU připojil.

DU naslouchá v rámci Protokolu 89 na adrese 0xfe a obsluhuje Logickou periferii 0x0a. Tato LP rozumí třem RQ. RQ 1 provede registraci zařízení, které o registraci žádá. RQ 2 provede registraci zařízení, s kterým chceme komunikovat a RQ 3 registruje současně vlastní jméno i jméno vzdálené.

2.1.2.1 Registrace bodu poskytujícího služby

Bod, který poskytuje jiným bodům služby, je nutné registrovat na DU pod jedinečným jménem. Pokud tedy máme zařízení, které poskytuje pro ostatní přesný čas, s adresou 0xfd a jménem LX_RTC253, musíme odeslat datové ústředně s adresou 0xfe registrační paket v následujícím tvaru (musí se doplnit CRC):

src	dest	relay	id	type	size	odes	prij	LP	RQ	L_D	L_P	PAR	DATA
0xfd	0xfe	0x00	0x00	0x0e	0x0e	0xfd	0xfe	0x0a	0x01	0x09	0x01	0x09	LX_RTC253

Tímto paketem jsme zajistili, že se zařízení LX_RTC253 registrovalo na DU a ostatní mohou využívat jeho služeb.

2.1.2.2 Propojení s bodem poskytujícím služby

V předchozí podkapitole je popsána registrace zařízení, které poskytuje informace o času. Pokud chceme tyto informace využívat, musíme si dané zařízení zaregistrovat jako protistranu ke komunikaci. To se dělá obdobně s tím rozdílem, že RQ = 0x01 zaměníme za 0x02.

V odpovědi na toto spojení se v datové části UCB vrací hodnota *relay*, kterou musíme pro další komunikaci v protokolu 89 s tímto zařízením vždy nastavit. V odpovědi přichází v UCB status 0x41, pokud se registrace zdařila, nebo jiná hodnota, která odpovídá případné chybě spojení – neexistujícímu protějšku apod.

2.2 Elevátor

Dosah bezdrátové části modulu je díky malému výkonu RF vysílače 10mW omezený. Elevátor je způsob předávání paketů, kterým je možné předávat pakety přes RF na delší vzdálenosti. V tomto režimu slouží každý modul jako opakovač.

Při použití elevátoru, musí být horní dva bity typu paketu v paketu Protokolu 89 nastaveny na logickou 1. Dolní 4 bity určují, kolik je potřeba skoků mezi nejvzdálenějšími zařízeními. Všechna zařízení musí mít nastavený stejný rozsah elevátoru, tzn. údaj o počtu zařízení v bezdrátové síti.

2.3 Timeout

Jak již bylo zmíněno na začátku kapitoly 2, při komunikaci přes Protokol 89 můžeme počítat s tím, že se odpověď vrátí do určité předem známé doby, neboli než vyprší timeout. Platí zde zásada, že timeout se měří pouze v jediném místě, a to tam, kde příslušný požadavek vznikl.

Pokud komunikujeme přes linku RS-485 nebo bezdrátově přímo, nastavujeme timeout standardně na hodnotu 50ms. Pokud komunikujeme pomocí elevátoru, pak timeout = 12ms x rozsah elevátoru x počet skoků.

3 Výběr vhodné platformy

3.1 Původní řešení

V původním řešení byly bezdrátové sítě připojeny k internetu pomocí UM osazeného modulem XPort. Toto řešení mělo několik výhod, ale také spoustu nevýhod, které nakonec převážily.

V roce 2004, kdy se s danou technologií začínalo, byl XPort jedinou možností, jak připojit za přijatelnou cenu osmibitový kontrolér do sítě TCP/IP. XPort bylo možné pomocí protokolu *telnet* jednoduše nakonfigurovat. Na straně TCP/IP nastavit IP adresu a port, ke kterému má být připojen. Na straně sériového kanálu pak stačilo nastavit přenosovou rychlost, počet datových bitů, paritu, a zda využívat hardwarové řízení toku dat.

Přes všechny výhody se ale objevilo několik problémů, které použití znesnadňovaly. Největším problémem bylo, že XPort musel mít napevno zadanou IP adresu, ke které se má připojovat a nepodporoval možnost zadání DNS serverů. Tento nedostatek se projevoval hlavně při změně veřejné IP adresy serveru, na kterém byla spuštěna DU. Po změně veřejné IP bylo nutné všechny XPorty nastavit znovu, což bylo časově a ekonomicky náročné.

Jako druhá nevýhoda se ukázala uzavřenost operačního systému, který byl součástí XPortu. Za vývojovou verzi požadovala firma Lantronix cca 80 000 Kč, což by prodražilo celý systém. Z tohoto důvodu musela být celá režie přihlašování na DU řešena v univerzálním modulu, ke kterému byl XPort připojen.

3.2 Hledání nového řešení

Z předchozích zkušeností se ukázalo, že vývoj vlastního hardware pro připojení k DU není ekonomicky výhodný, a je tedy lepší použít již hotové řešení, které ve výsledku může vyjít levněji. Kritéria pro výběr nového řešení byla následující:

- Cena výsledného zařízení
- Spotřeba zařízení
- Velikost zařízení
- Možnost osadit UM přímo do krabičky dodávané se zařízením

3.2.1 Bifferboard

Bifferboard[2] je miniaturní počítač o rozměrech 68x28x21mm, založený na procesoru ekvivalentním s R8610. Jedná se o 32 bitový RISC procesor kompatibilní s instrukční sadou procesorů 486SX, bez podpory matematického koprocesoru. Procesor je taktovaný na 150MHz, na základní desce je

osazeno 32MB paměti RAM a 8MB paměti FLASH. Dále je na desce vyvedeno 1 USB, 1 Fast Ethernet konektor, sériový port a až 6 GPIO. Spotřeba je 1W. V prodeji je také verze obsahující dva USB porty a čtečku SD/MMC karet. Cena verze s jedním USB portem se pohybuje kolem 1000 Kč.

V Bifferboardu je standardně nainstalováno OpenWRT, to lze ale nahradit libovolným jiným operačním systémem, který je na tomto zařízení podporován. Z operačních systémů, které jsou podporované, lze uvést GNU/Linux a jeho distribuce Gentoo, Slackware a Debian.

Podle udávaných parametrů byl Bifferboard ideálním základem, pro řešení této práce. Vzhledem k jeho rozměrům však nešlo osadit UM přímo do krabičky s Bifferboardem. Další krabička pouze pro UM by zvedla celkové náklady.

3.2.2 Edimax BR-6104KP

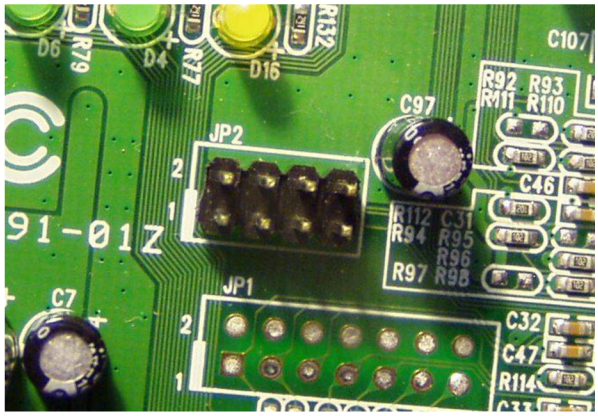
Edimax BR-6104KP[3] (dále uváděný také jako router) je běžně prodávaný router. Cena tohoto routeru Edimaxu se pohybuje kolem 600 Kč. Za danou cenu nabízí 32 bitový procesor Infineon ADM5120P taktovaný na 175MHz, 16MB RAM a 2MB paměti FLASH. Na základní desce jsou vyvedené 2 USB 1.1 porty, 5 Fast Ethernet portů, 1 UART a GPIO ovládající LED indikující stav zařízení.

Edimax BR-6104KP je od výrobce vybaven operačním systémem postaveným na GNU/Linuxu, který lze vyměnit za vlastní, například OpenWrt nebo jiné distribuce zaměřené na tento Edimax.

Po konzultaci s panem Ing. Jiřím Nitschem jsme se nakonec rozhodli pro tento router. K dispozici byly ještě další podobné routery, které umožňovaly instalaci vlastní distribuce, ty však měly mnohonásobně vyšší cenu.

4 Příprava hardware routeru Edimax BR-6104KP

Na základní desce routeru je neosazený konektor JP2. Tento konektor je připojen na UART procesoru ADM5120P[4] na piny 127 a 129. Pin 127 je RxD, pin 129 je TxD. Na obrázku 4.1 je vidět osazený konektor JP2. Na pinu č. 1 je RxD, na pinu č. 2 je napájecí napětí 3,3V, na pinu č. 7 je TxD a na pinu č. 8 je pak GND.

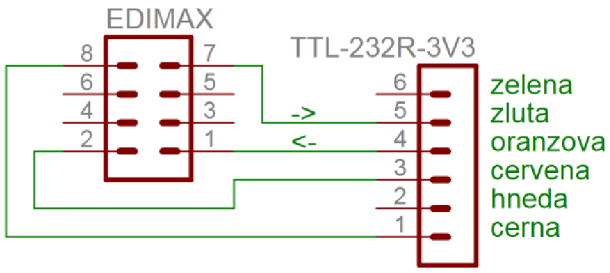


Obrázek 4.1: Osazený konektor JP2

4.1 Připojení k počítači

Aby bylo možné nahrávat a ladit nový software pro tento router, je nutné jej připojit k počítači. Vzhledem k napěťovým úrovním na pinech procesoru není možné připojit router přímo k sériovému portu počítače. Tam se při logické úrovni 0 může vyskytovat napětí až 12V. V případě logické úrovně 1 až -12V. Takto vysoké napětí by mohlo procesor v routeru Edimax zničit, proto je nutné použít vhodný převodník.

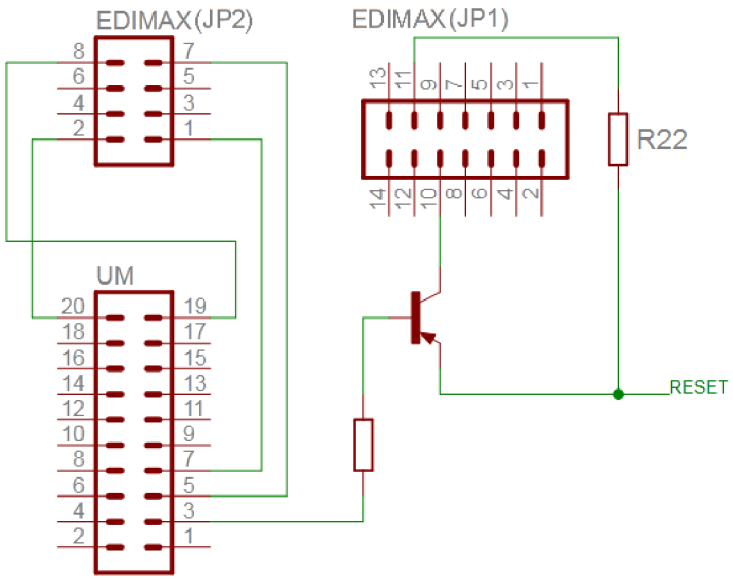
K tomuto účelu jsem měl k dispozici převodník TTL-232R-3V3[5] od FTDI. Jedná se o kabel obsahující převodník USB – UART s 3,3V napěťovými úrovněmi. Na obrázku 4.2 je schéma zapojení redukce mezi kabelem a konektorem JP2 na základní desce routeru Edimax BR-6104KP.



Obrázek 4.2: Schéma zapojení redukce mezi kabelem TTL-232R-3V3 a konektorem JP2 Edimaxu BR-6104KP

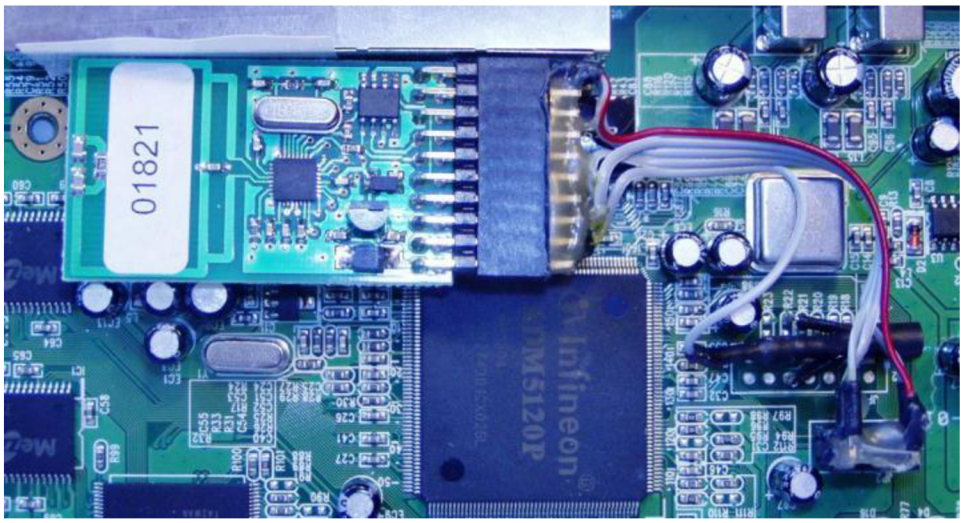
4.2 Připojení univerzálního modulu

Univerzální modul připojený k routeru plní dvě funkce. Zabezpečuje propojení routeru s ostatními moduly v bezdrátové síti a zároveň funguje jako monitor běhu (pro zkrácení watchdog) routeru. Schéma připojení univerzálního modulu je na obrázku 4.3.



Obrázek 4.3: Schéma připojení UM k Edimaxu

Univerzální modul lze napájet buď napětím 5V nebo 3,3V. Vzhledem k vyvedenému napájení 3,3V na konektor JP2 bylo zvoleno toto napájecí napětí. Napěťové úrovně linek RxD a TxD univerzálního modulu jsou shodné s napěťovými úrovněmi na konektoru JP2. Nic tedy nebrání přímému propojení. Na obrázku 4.4 je detail připojení univerzálního modulu.



Obrázek 4.4: Detailní pohled na připojení UM k routeru

4.2.1 Watchdog

Vzhledem k určení celého zařízení do ostrého provozu, konkrétně k monitorování ve Fakultní nemocnici v Olomouci bylo nutné zabezpečit bezproblémový chod a předcházet stavu, kdy dojde k chybě a zařízení přestane odpovídat a bude tedy nutné servisního výjezdu k jeho restartu. Také se ukázalo, že při pomalém náběhu napájení po některých výpadcích dodávky elektrické energie není router schopný korektně naběhnout. Proto bylo nutné přidat watchdog.

Funkce watchdogu byla implementována do univerzálního modulu a plní dvě funkce. Po připojení napájení je na výstupu z modulu po dobu 1s 0V. Tento výstup je přiveden přes rezistor na bázi tranzistoru, který spíná pin RESET procesoru ADM5120.

Druhá část watchdogu sleduje komunikaci mezi aplikacemi v routeru a UM. Restart se provede, pokud za stanovený čas, implicitně 10 minut, neodešle aplikace libovolný dotaz na sériovou linku univerzálnímu modulu.

5 Operační systém pro router Edimax BR-6104KP

Pro Edimax BR-6104KP je k dispozici spousta alternativních operačních systémů. Jako příklad lze uvést několik nejznámějších:

- AMiLDA (Advanced Mini Linux Distribution for ADM5120) [6] – jak již název napovídá, jedná se o distribuci určenou pro zařízení s procesorem ADM5120. Distribuce obsahuje kompletní webové prostředí a dá se použít jako plnohodnotná náhrada vestavěného systému.
- Midge [7] – další z distribucí podporujících procesor ADM5120. Tato distribuce obsahuje základní nástroje jako ssh, telnet, inetd, syslogd, netcat, tftp, wget a další. K instalaci dalších programů obsahuje nástroj ipkg.
- Squidge[8] – tato distribuce se oproti předchozím liší schopností mít souborový systém uložený na USB disku. Lze tak využívat mnohem více programů a disk využít také k ukládání dat. Router s touhle distribucí lze využít jako jednoduchý souborový server.

Každá z uvedených distribucí má své výhody a nevýhody. Jedno však mají společné, všechny staví na OpenWrt, které bude využito pro totu práci.

Konfigurace celého zařízení bude uložena na USB disku, sestavená image s tím tedy musí počítat. Souborový systém na USB disku bude FAT32 z důvodu kompatibility s většinou počítačů, které se nachází v prostředí firmy Condata. Také vlastní aplikace starající se o komunikaci a sběr dat se budou nacházet na USB disku. To pro případ, aby je bylo možné kdykoliv snadno vyměnit a nemusela se kvůli tomu vytvářet nová image celého operačního systému pro zvolený router.

5.1 OpenWrt

OpenWrt [9] je distribuce GNU/Linuxu zaměřená na vestavěná zařízení, jakými mohou být například routery, print servery nebo třeba NAS. Mezi podporovaná zařízení patří produkty od výrobců jako je ASUS, D-Link, Linksys, TP-Link a od mnohých dalších. Hlavní výhodou tohoto systému je možnost sestavit jej dle vlastní potřeby.

Aktuální stabilní verze je *Backfire 10.03*, ta se však pro účely této práce ukázala jako nevhodná. Některé potřebné funkce (podpora GPRS/3G modemu) pro další rozšiřování práce v této verzi nebyly plně funkční nebo byly značně nestabilní. V navazujícím textu bude probíraná vývojová verze, konkrétně *r17794*, která se ukázala jako dostatečně stabilní.

5.1.1 Závislosti nutné pro překlad

Ještě před tím, než se začne se stahováním zdrojových kódů, je potřeba v hostitelském systému doinstalovat některé potřebné balíčky. V Debianu jsou to následující: *subversion, build-essential, asciidoc, binutils, bzip2, fastjar, g++, gcc, gewk, libgtk2.0-dev, intltool, jikes, zlib1g-dev, make, libncurses5-dev, libssl-dev, patch, perl-modules, rsync, ruby, sdcc, unzip, wget, sdcc-nf, gettext* a *xsltproc*.

5.1.2 Stažení zdrojových kódů

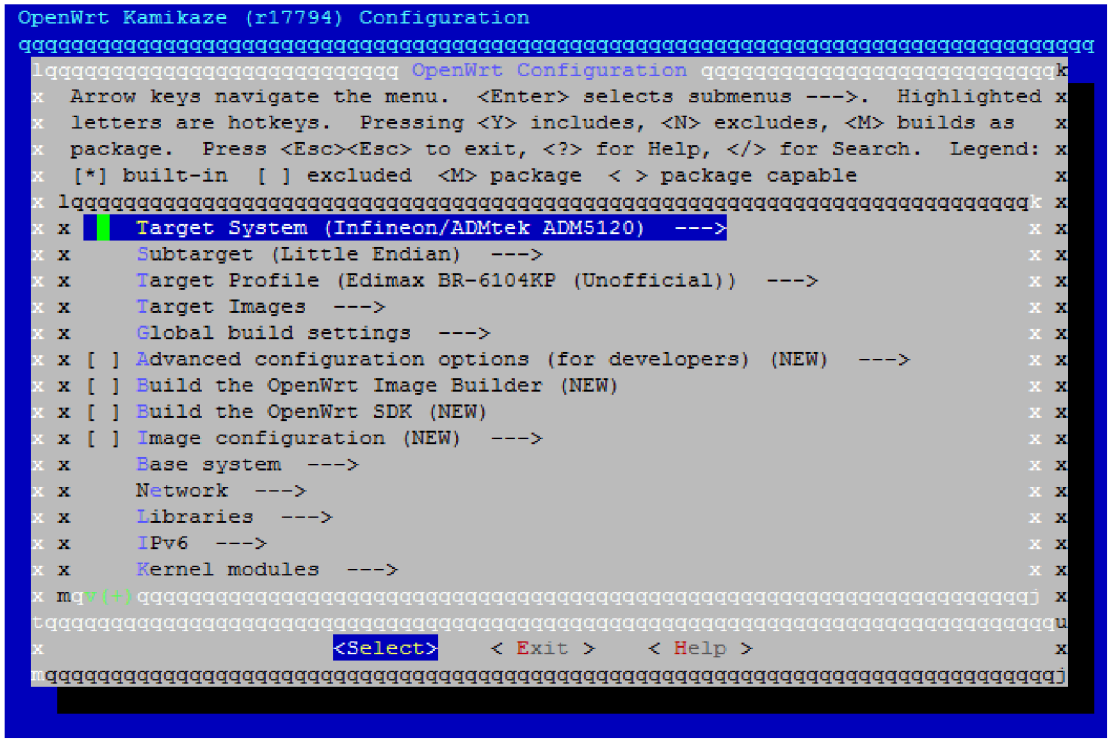
Zdrojové kódy pro sestavení vlastní distribuce OpenWrt je možné stáhnout ze svn.

```
svn co svn://svn.openwrt.org/openwrt/trunk -r17794
```

5.1.3 Sestavení jádra a balíčků

K sestavení nového Linuxového jádra se používá prostředí Buildroot (obrázek 5.1). Je to soubor Makefile a patchů a lze pomocí něj snadno sestavit novou image pro nahrání do routeru.

Po přechodu do adresáře se zdrojovými kódy je možné spustit Buildroot prostředí. To se provede příkazem `make menuconfig`.



Obrázek 5.1: Prostředí Buildroot

V menu *Target System* je potřeba vybrat volbu *Infineon/ADMtek ADM5120* a v *Target Profile Edimax BR-6104KP (Unofficial)*. To zabezpečí překlad pro správnou platformu. V menu *Target*

Images je výběr souborového systému, který se má použít. Na výběr je *jffs2*, které umožňuje připojení souborového systému i pro zápis. Vzhledem k požadované robustnosti je lepší použít *squashfs*, který se dá připojit pouze pro čtení. Nehrozí tak, že by mohlo dojít k porušení struktury dat a tím k nefunkčnosti zařízení.

Aby bylo v budoucnu možné sestavit vlastní image, je nutné vybrat volbu *Build the OpenWrt Image Builder*. Pro kompilaci vlastních aplikací je pak nutné vybrat volbu *Build the OpenWrt SDK*, tím se nainstaluje kompilátor a sada knihoven pro cross kompilaci.

V menu *Base system* je potřeba označit jako vestavěné balíčky *base-files, bosybox, libc, mtd, opkg* následující balíčky stačí označit jako moduly: *hotplug2, uci, udevtrigger, librt* a knihovnu *libpthread.* Tyto balíčky zabezpečí základní funkci routeru.

Ze síťových balíčků jsou potřeba balíčky *ppp* a *chat*. Tyto balíčky umožní připojení zařízení k internetu pomocí GPRS/3G modemu. Balíčky stačí označit jako moduly. O jejich instalaci se pak postará Image Builder.

Z menu *Kernel modules* jsou potřeba balíčky *kmod-usb-core, kmod-usb-adm5120, kmod-usb-storage, kmod-usb-uhci, kmod-usb-serial, kmod-ledtrig-adm5120-switch, kmod-gpio-dev, kmod-leds-gpio, kmod-ppp, kmod-fs-vfat, kmod-fs-ext2, kmod-nls-base, kmod-nls-iso8859-1, kmod-nls-cp437* a *kmod-scsi-core*. Tyto balíčky zajistí podporu pro USB disk se souborovým systémem FAT32 nebo ext2/3, podporu pro připojení GPRS/3G modemu a indikaci komunikace na síťových rozhraních pomocí LED.

V menu *Utilities* je balíček *admswconfig* zajišťující nastavení pěti Fast Ethernet portů do switche a balíček *gpioctl* pro ovládání stavových LED POWER, USB1 a USB2.

Vzhledem k tomu, že k /dev/ttyS0 bude připojen univerzální modul, je nutné přepnout sériovou konzolu na /dev/ttyS1. To se provede editací souboru *./target/linux/adm5120/router_le/config-2.6.28*, kde je potřeba upravit `ttyS0` na řádku začínajícím: `CONFIG_CMDLINE=` na `ttyS1`.

Po výběru balíčků lze sestavit image příkazem `make -j 2 V=99`. Parametr `-j` určuje kolik může současně běžet procesů a parametr `V=99` zapíná režim s detailním výpisem.

5.2 Image Builder

Vygenerovaná image je sice funkční, ale není možné ji nijak konfigurovat a přidávat další balíčky. Proto vznikl nástroj Image Builder. Jedná se o sadu Makefile, které umožňují vygenerovat image znovu. Oproti původnímu sestavení je však možné přidat nebo upravit vlastní soubory a balíčky. V adresáři *./bin* se Image Builder nachází spakovaný v archivu *OpenWrt-ImageBuilder-adm5120-for-Linux-i686.tar.bz2*. Rozbalený Image Builder se pak nachází v adresáři *./build_dir/target-mipsel_uClibc-0.9.30.1/OpenWrt-ImageBuilder-adm5120-for-Linux-i686*.

5.2.1 Příprava souborů image pro Image Builder

Do vlastního adresáře je potřeba nakopírovat soubory, které se mají přidat do výsledné image. Jméno adresáře bude *files*. Do tohoto adresáře je potřeba vytvořit další a to *bin, etc, mnt, tmp* a adresář *usr*. V adresáři *bin* se budou nacházet skripty potřebné pro běh zařízení. Jedním takovým je *checkUSB*, který kontroluje stav USB disku a v případě, že se odpojí a připojí pouze pro čtení, tak jej znovu připojí pro zápis. Při dalším restartu zařízení se pak provede kontrola integrity souborového systému. Tento skript se spouští každou minutu.

Do adresáře *mnt* je potřeba vytvořit adresář *sda1*, kam se připojí USB disk. Na tento adresář je potřeba udělat symbolický odkaz a pojmenovat jej jako */disk*, který bude následně umístěn v kořenu image. Také je nutné vytvořit soubor *.nodisk*, který informuje o nepřítomnosti USB disku a soubor *owcu.log*, kam se bude ukládat protokol chyb, pokud není USB disk připojen.

V adresáři *tmp* stačí vytvořit symbolický odkaz na soubor */disk/owcu.log*. To kvůli zpětné kompatibilitě s některými aplikacemi.

Adresář *usr/sbin* obsahuje nástroj dosfsck, který se využívá ke kontrole souborového systému FAT32 na USB disku.

V adresáři *etc* je základní nastavení celé image. V podadresáři *crontabs* je soubor *root*, který zajišťuje spouštění skriptu *checkUSB* každou minutu. Soubor *10-mount* v podadresáři *hotplug.d/block* zajišťuje kontrolu integrity FAT32 a připojení USB disku do zvoleného adresáře automaticky po jeho připojení nebo po startu zařízení. V podadresáři *init.d* se nachází upravené startovací skripty z původní image. V souboru *done* je zakomentovaná volba přepnutí souborového systému na jffs2. Soubor *network* je upraven tak, aby se po startu systému nedetekoval Wi-Fi karty a nenastavoval switch v procesoru ADM5120. Soubor *ngapps* se postará o spuštění skriptu, který provede další nastavení požadovaných parametrů.

Aby se sériová konzole přepnula na ttyS1, je potřeba upravit v souboru *inittab* řádek `ttyS0::askfirst:/bin/ash - login` na `ttyS1::askfirst:/bin/ash - login`. Tento soubor je pak potřeba nakopírovat do adresáře *files*. K nastavení správného času je potřeba v adresáři *files* vytvořit alespoň prázdný soubor *TZ*. Ještě je možné přidat soubor *banner*, který se bude vypisovat při přihlášení k zařízení.

5.2.2 Generování image nástrojem Image Builder

Nové sestavení image nástrojem Image Builder se spouští příkazem `make` v adresáři nástroje. Je možné zadat několik voleb sestavení. Parametrem *PROFILE* je možné zadat profil zařízení, pro které se má image sestavovat, pro Edimax BR-6104KP je to *BR6104KP*. Parametrem *PACKAGES* lze přidat další balíčky, které se mají do výsledné image zahrnout. Posledním parametrem FILES je možné zadat cestu k souborům, které se přidají do image.

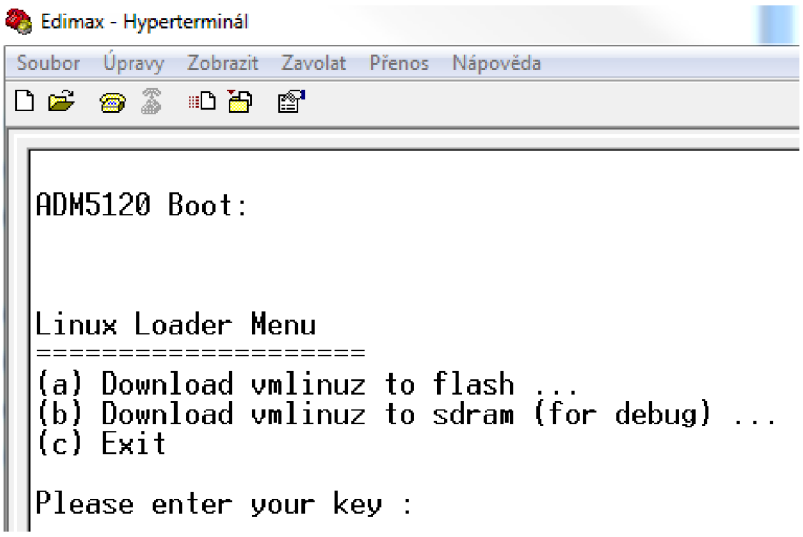
Požadované balíčky pro správnou funkci jsou: *librt, libpthread, admswconfig, kmod-ledtrig-adm5120-switch, gpioctl, kmod-gpio-de,v kmod-leds-gpio, kmod-usb-core, kmod-usb-adm5120, kmod-usb-storage, kmod-usb-uhci, kmod-fs-vfat, kmod-fs-ext2, kmod-scsi-core, libgcc, kmod-nls-base, kmod-nls-cp437, kmod-nls-iso8859-1, kmod-usb-seriál, kmod-ppp, ppp* a *chat*.

Výsledná image je pak v podadresáři *bin* adresáře nástroje Image Builder.

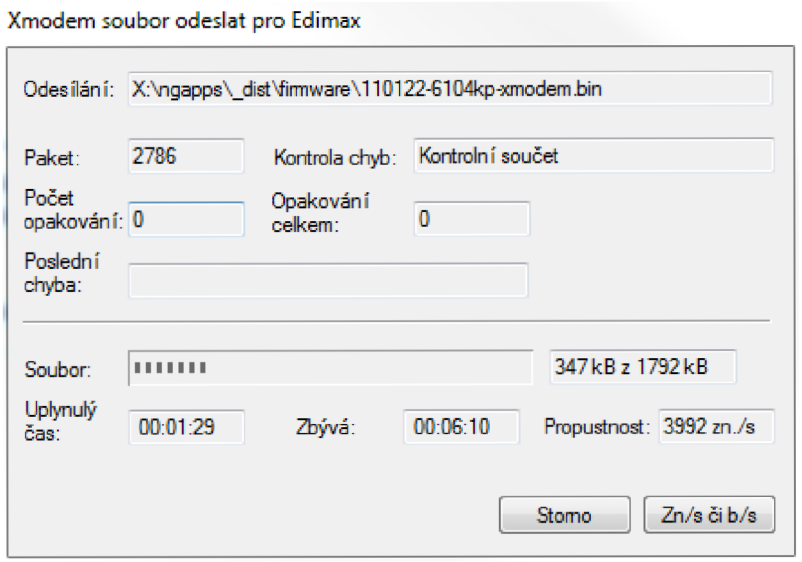
5.3 Nahrání image do Edimaxu BR-6104KP

Nová image se dá do Edimaxu BR-6104KP nahrát dvěma způsoby. Prvním způsobem je použití nástroje *mtd*. Tento nástroj nelze použít, pokud ještě není v routeru OpenWrt. Je tedy nutné využít druhý způsob. Aktualizaci je možné nahrát konektorem JP2 protokolem XMODEM.

Jedním z nástrojů podporujících protokol XMODEM je *Hyperterminál*. Tento program je dodávaný spolu s Microsoft Windows. Bootloader komunikuje přes sériovou linku rychlostí 115 200bps s osmi datovými bity a jedním stop bitem. Na obrázku 5.2 je vidět výzva Linux Loaderu k jeho aktivaci. Ta se provede třemi stlačeními mezerníku. V druhé části je vidět menu Linux Loaderu, stiskem klávesy *c* je možné Loader opustit a router začne bootovat původní image. Stiskem klávesy *b* je možné nahrát novou image do RAM. To je výhodné zejména pro ladění. Pokud je již nová image dostatečně odladěná, je možné ji stiskem klávesy *a* nahrát do FLASH paměti routeru. Na obrázku 5.3 je okno ukazují nahrávání nové image do Edimaxu.



Obrázek 5.2: Linux Loader v Edimaxu BR-6104KP



Obrázek 5.3: Odesílání nové image do Edimaxu BR-6104KP

6 Návrh aplikace

6.1 Konfigurace aplikací

Existuje několik způsobů, jak aplikacím předávat požadované parametry. Jedním z prvních způsobů je předávat parametry aplikaci na příkazovém řádku. Pokud je však těchto parametrů hodně, tak se spouštění aplikace stává velmi složitým. Ne každý si totiž pamatuje nutné parametry pro správný start aplikace.

Mnohem vhodnějším způsobem je mít uloženou konfiguraci v souboru a aplikaci pouze předávat cestu k tomuto souboru. Konfigurační soubor může mít různou strukturu. Mezi nejznámější patří XML a INI. XML je značkovací jazyk, který lze použít pro mnohé účely. Může sloužit pro tvorbu webových stránek, k ukládání dat, nebo také k uložení velmi složitého strukturovaného dokumentu. Z tohoto popisu je jasné, že XML je k danému účelu zbytečně složité.

6.1.1 Soubor INI

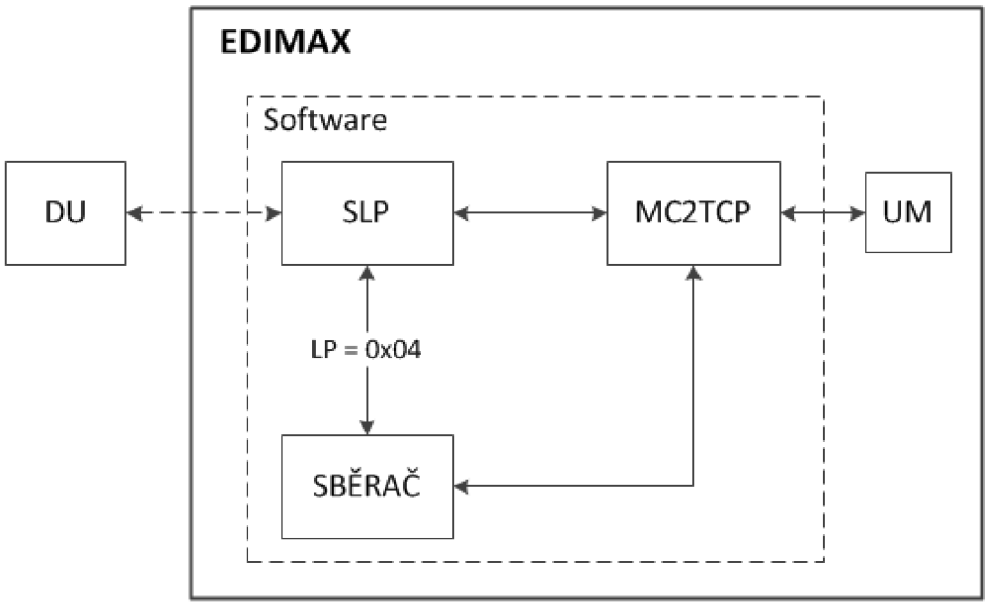
INI soubory [10] byly dříve de-facto standardem pro ukládání konfigurace. Dnes je nahradilo XML, avšak na spoustu aplikací jsou dostačující. INI soubory jsou textové soubory s jednoduchou strukturou.

Přiřazení hodnoty k vlastnosti se provádí jednoduchým řádkem `vlastnost = hodnota`. Nová sekce souboru začíná nadpisem uzavřeným v hranatých závorkách `[sekce]`. Pro větší přehlednost lze použít komentáře, ty se uvozují znakem středníku.

6.2 Struktura aplikací

V prvotních fázích vývoje byla vytvořena aplikace, která pouze nahrazovala zastaralý XPort a přidávala možnost sběru dat. To se v průběhu času ukazovalo jako nevýhodné. Jakékoliv úpravy a přidávání nových funkcí pak znamenalo úpravu celé aplikace. Ve výsledku existovala jedna velká aplikace, která se nedala téměř udržovat a dále upravovat. Z toho důvodu byla zvolena jiná možnost.

Touto možností bylo víc malých a jednoduchých aplikací, které společně obstarají funkce té velké, ale navíc přinesou možnost snadného rozšiřování celého zařízení. Změnu funkce lze zajistit výměnou aplikace za jinou. K přidání dalších funkcí stačí přidat aplikaci novou. Principielní schéma je na obrázku 6.1. Všechny aplikace spolu komunikují přes TCP sockety.



Obrázek 6.1: Principielní schéma aplikací

6.3 Sada základních aplikací

Sada základních aplikací obsahuje aplikace, které jsou nezbytné pro základní funkci a další funkční rozšiřování. Do této základní sady patří aplikace *SLP* a *MC2TCP* – Xport a Fronta.

6.3.1 MC2TCP

Aplikace MC2TCP je aplikace, která zprostředkovává komunikaci mezi univerzálním modulem a mezi ostatními aplikacemi pomocí TCP soketů. Aplikace má dva režimy.

6.3.1.1 XPort

Tento režim aplikace MC2TCP je z důvodu zpětné kompatibility s modulem XPort, kdy se o režii připojování k datové ústředně stará připojený univerzální modul.

Aplikace v tomto režimu vytváří dvě vlákna. První vlákno se připojí na zadanou IP adresu a cílový port. V případě výpadku spojení se snaží spojení obnovit. Vlákno přijímá pakety přicházející z TCP socketu a předává je univerzálnímu modulu. Druhé vlákno otevírá sériový port, na kterém testuje, zda jsou dostupná nějaká data. Pokud data dostupná jsou, tak počká, než se přijme celý paket. Ten pak odešle přes TCP socket datové ústředně. V tomto režimu nemá žádná jiná aplikace k UM přístup.

6.3.1.2 Fronta

V běžném provozu se stává, že kromě aplikace sbírající data, potřebuje k univerzálnímu modulu přistupovat i operátor, nebo další aplikace. Každá taková dotazující se aplikace by pak měla vlastní

číslování pro ID paketu protokolu 89. ID, které by se postupně neinkrementovalo, nebo by se potkávalo s jiným číslování ID, by mohlo způsobit nekonzistenci dat přijatých z univerzálního modulu. Navíc není možné otevřít sériový port souběžně pro více aplikací. Proto je nutné použít frontu požadavků.

Režim fronty paketů obsahuje kongruentní server, který naslouchá na zadaném portu. Pokud se k tomuto portu připojí jiná aplikace a odešle paket protokolu 89, tak se tento paket zařadí do *FIFO* fronty. Spolu s paketem se ukládá informace o deskriptoru daného TCP socketu, z kterého paket přišel.

Druhé vlákno, běžící současně s vlákny naslouchajícího serveru, pravidelně prochází frontu a přijaté pakety po úpravě jejich ID odesílá univerzálnímu modulu. Jakmile je paket odeslán, tak aplikace čeká předem daný čas na odpověď. Pokud odpověď nepřijde, tak volitelně odpoví nadřazenému systému chybovou hláškou. V případě přijaté odpovědi tuto odpověď po úpravě ID odesílá zpět na socket, ze kterého přišel dotaz.

6.3.2 SLP

SLP nebo také server logických periferií je aplikace, pomocí které se dá jednotně komunikovat se všemi samostatnými aplikacemi v routeru a přistupovat k univerzálnímu modulu, a tím i k celé bezdrátové síti tvořené dalšími moduly.

Hlavní funkcí aplikace je registrace na datovou ústřednu. Tuto registraci aplikace provádí opakovaně v pravidelném intervalu. Pokud by se pravidelně neregistrovala, tak by ji datová ústředna po 15 minutách sama odpojila.

SLP se po prvním připojení k datové ústředně snaží připojit i k MC2TCP. To proto, aby mělo přístup k univerzálnímu modulu a mohlo tak předávat pakety určené pro bezdrátovou síť. Takto je zajištěna částečná kompatibilita s původním XPortem s tím rozdílem, že se o registraci na DU stará SLP a nikoliv univerzální modul RF AP.

Aplikace v závislosti na její konfiguraci spustí tolik TCP serverů, kolik je potřeba logických periferií. V konfiguračním souboru má aplikace zadané na jakém TCP portu má server naslouchat a na jakou LP se bude daný socket mapovat. Tyto servery nejsou kongruentní, neumožňují tedy víc připojení k jedné logické periferii.

Všechny příchozí pakety z datové ústředny jsou tříděny podle určitých pravidel. Pokud přijde paket s cílovou adresou v hlavičce paketu protokolu 89 shodnou s nastavenou adresou aplikace, tak se předá dané logické periferii. LP 0 je vestavěná přímo v aplikaci a umožňuje přečtení některých hodnot v aplikaci a přenos souborů. Ostatní LP jsou mapovány ke zvoleným TCP serverům.

Pokud přijde z datové ústředny paket s jinou cílovou adresou než je adresa SLP, tak se tento paket přeposílá aplikaci MC2TCP, která jej pošle připojenému univerzálnímu modulu.

6.3.2.1 Správce souborů

Se soubory na USB disku je v některých případech nutné manipulovat, i když není zařízení fyzicky přítomno a je již nasazeno v ostrém provozu. Takovými případy může být aktualizace aplikací nebo stahování shromážděných dat aplikací *sběrače*. Z tohoto důvodu byl implementován jednoduchý správce souborů. Kromě manipulace se soubory umožňuje i spouštění aplikací. Přehled funkcí a jejich čísel požadavků v rámci UCB je v tabulce 6.1. Detailní popis jednotlivých RQ je pak na přiloženém CD.

RQ	Očekává	Vrací	Popis
0x01	Jméno adresáře		Vytvoření nového adresáře
0x02		Jméno souboru	Udělá výpis adresáře, ten pak uloží do souboru
0x03	Jméno souboru/adresáře		Smaže soubor/adresář
0x04			Odešle soubor ze zařízení
0x05			Odešle souboru do zařízení
0x06		Aktuální cestu	Zjistí aktuální cestu
0x07	Cestu k adresáři		Přepne do zadaného adresáře
0x09	Cestu a jméno aplikace	PID	Spustí zadanou aplikaci
0x0A	PID	Stav aplikace	Kontrola, zda daná aplikace skončila

Tabulka 6.1: RQ pro správce souborů aplikace SLP

6.4 Sběrač

Při provozu v reálných podmínkách se může stát, že se zařízení nepodaří připojit k datové ústředně. Důvody tohoto neúspěchu mohou být různé, ale zařízení se s tím musí umět vyrovnat. Pokud se používal XPort, tak se při výpadku připojení k datové ústředně přišlo o veškerá data. Data totiž sbírala aplikace, která se k zařízení připojovala právě pomocí datové ústředny a data tak v době výpadku nemohla sbírat.

Právě možnost připojit k routeru Edimax BR-6104KP USB disk přinesla řešení tohoto problému. Sbírání dat je tedy možné svěřit aplikaci běžící přímo v zařízení. Přenos dat k dalšímu zpracování je pak v režii nadřazené aplikace, která původně tato data sbírala přímo. Aby aplikace sběrače věděla, jaká data z jakých modulů sbírat, musí dostat seznam těchto zařízení a požadavků na ně.

Pokud se nepodaří spojit s některým modulem zadaným v matrici, tak se po několika neúspěšných pokusech odesílá informační SMS operátorovi.

6.4.1 Matrice sběrače

Matrice sběrače je soubor, který obsahuje seznam zařízení a požadavků pro sběr dat. Matrice sběrače je shodná jako matrice pro původní aplikaci, která shromažďovala data. To umožňuje jednoduchou výměnu původního systému za tento nový.

Matrice je textový soubor, kde každý záznam je na vlastním řádku. Jako oddělovač hodnot se používá středník. Pokud je však středník prvním znakem na řádku, tak se jedná o komentář. Na následujícím řádku je vidět jeden záznam matrice.

```
I;jmeno;;ADR;;;;;;;;;;LP,RQ;P;D;KANAL;;;;VZ;;;;;;;;
```

- První část řádku záznamu určuje typ záznamu. *I* znamená vstup. A znamená, že se výsledná hodnota pro kanál bude počítat pomocí vzorce zadaného v poli *VZ*. Další písmena jsou vyhrazena pro jiné funkce nadřazené aplikace.
- Další část určuje jméno záznamu. Pod tímto názvem se také ukládají informace z datového kanálu daného čidla.
- Čtvrtá část záznamu definuje adresu univerzálního modulu, ke kterému je připojeno čidlo, a s kterým se bude komunikovat.
- Část označená jako *LP,RQ* označuje logickou periferii a typ požadavku, který zvolené čidlo obsluhuje.
- V sekci označené jako *P* a *D* jsou zadané parametry a data oddělená čárkou.
- Sekce KANAL obsahuje seznam čárkou oddělených čísel bytů UCB s odpovědí, které se budou ukládat do zvláštního souboru. Tento soubor potom mohou využít další aplikace.
- VZ je číslo vzorce, který definuje, jak se bude s daty v kanálu zacházet.

V nadřazené aplikaci se využívají i další hodnoty z řádku záznamu. Pro sběrač však nejsou důležité.

6.5 Odesílání SMS

Odesílání SMS bylo implementováno jako samostatná aplikace. To proto, aby se daly odesílat SMS i při událostech, které nejsou vyvolány sběračem. Přidat GSM modem pro odesílání SMS ke každému zařízení by bylo značně neekonomické, z toho důvodu běží na serverech firmy Condata aplikace, která umožňuje odesílat jak placené SMS přes GSM modem, tak i SMS zdarma přes webové brány mobilních operátorů. Aplikace navíc umožňuje odesílání krátkých informačních emailů a na DU se hlásí jako LX_SMS253.

Klientská aplikace v zařízení se k této aplikaci přes DU připojuje. Aplikaci se v parametrech příkazového řádku předávají informace o konfiguračním souboru pro nastavení připojení k DU a se jménem serverové části aplikace. Dále se v parametrech předává typ odesílané zprávy, *sms* pro

placenou SMS, *freesms* pro SMS zdarma přes webovou bránu mobilního operátora a *email* pro odeslání krátkého emailu. Délka textu je omezena velikostí datové části UCB.

V parametrech UCB se aplikaci předává adresát, v datové části potom text zprávy.

7 Implementace

7.1 Programové řešení

Jako programovací jazyk byl zvolen jazyk C. To z toho důvodu, že oproti původně zamýšlenému C++ byly výsledné binární soubory mnohem menší. Jazyk C navíc nabízí ukazatele, které jsou velkým pomocníkem při ukládání složitějších dat do vhodně zvolených datových struktur.

7.2 Knihovny pro práci s Protokolem 89

Před vlastním zahájením prací na aplikacích obstarávajících komunikaci a sběr dat bylo nutné napsat vhodné knihovny, které budou zabezpečovat komunikaci Protokolem 89. Tyto knihovny pak bylo nutné modifikovat jak pro přenosy s DU, tak pro přímou komunikaci s univerzálním modulem připojeným k sériovému portu routeru.

7.2.1 Struktura pro uložení paketu Protokolu 89

Pakety Protokolu 89 je možné ukládat do pole. To však není vhodné. Další vývojáři by tak museli přesně znát pořadí jednotlivých bytů v paketu. V tabulce 7.1 je struktura hlavičky paketu Protokolu 89 a UCB. Maximální součet DATA a PARAM je 249.

Datový typ	**Jméno**	**Datový typ**	**Jméno**	***Popis***
unsigned char	Src			*Odesilatel paketu*
unsigned char	dest			*Příjemce paketu*
unsigned char	relay			*Relay*
unsigned char	ID			*ID*
unsigned char	typ			*Typ paketu*
TUCB	***UCB***	unsigned char	Odes	*Odesilatel UCB*
		unsigned char	Prij	*Příjemce UCB*
		unsigned char	LP	*Logická Periferie*
		unsigned char	RQ_ST	*Request/Status*
		unsigned char	L_DATA	*Délka datových bytů*
		unsigned char	L_PARAM	*Délka parametrů*
		unsigned char	DATA[249]	*Datové byty*
		unsigned char	PARAM[249]	*Parametry*

Tabulka 7.1: Struktura paketu Protokolu 89 a UCB

7.2.2 Funkce pro práci s pakety Protokolu 89

Všechny deklarace funkcí pracujících s pakety Protokolu 89 se nachází v hlavičkových souborech *ucbtcp.h* a *ucbcom.h*. Jak z názvů souborů vyplývá, funkce starající se o TCP komunikaci jsou v souboru *ucbtcp.h*. Funkce pro komunikaci na sériovém portu jsou pak v souboru *ucbcom.h*.

Pro komunikaci přes TCP sockety jsou k dispozici funkce pro odesílání a příjem paketů. Funkce pro odesílání je `int send_p89(int sockfd, const TP89 *P89)`, funkce očekává dva parametry, kde prvním je deskriptor socketu, v druhém parametru je pak očekávána vyplněná struktura paketu, který se má odeslat. Funkce pro příjem paketů jsou dvě, první přijme paket a uloží jej do předem alokované struktury, tato funkce je deklarována jako `int recv_p89(int sockfd, TP89 *P89, int timeout`. Ve třetím parametru funkce očekává dobu v ms, kterou bude čekat na příchozí paket. Druhou funkcí pro příjem paketů je `char *recv_p89_data(int sockfd, int timeout, int *ID, int *len)`. Tato funkce se používá pro příjem datových paketů Protokolu 89. Tyto pakety místo UCB obsahují surová data. To se používá u přenosu dat správce souborů. Obdobně je k dispozici funkce `int send_p89_data` pro odeslání surových dat.

K registraci na datovou ústřednu je k dispozici funkce `int reg_du(int sockfd, unsigned char my_num, char *my_name, unsigned char RQ, char SetTime)`. Pokud je RQ = 1, tak funkce ve svých parametrech očekává adresu a jméno, pod kterým se bude aplikace registrovat na datovou ústřednu. Pokud je RQ = 2, tak očekává jméno vzdálené strany, s kterou bude komunikovat.

Mezi funkce, komunikující přes sériový port, patří funkce `int com_Open(char *port)` a `void com_Close(int fd)`. Tyto funkce se starají o otevření a uzavření zadaného sériového portu. Funkce odesílající a přijímající data protokolem 89 jsou `int send_serial_p89(int fd, const TP89 *P89)` a `int recv_serial_p89(int fd, TP89 *P89, int timeout)`. Funkce jsou shodné s jejich protějšky pro komunikaci přes TCP sockety. Odesílání paketů na sériový port, aby se předešlo restartu routeru watchdogem, řeší funkce `int serial_ping_RQ22(int fd)`, tato funkce odesílá na LP = 0 RQ = 0x22.

Kromě funkcí důležitých pro komunikaci jsou k dispozici i některé podpůrné funkce. Za zmínku stojí `int print_p89(const TP89 *P89)`, která na *stdout* vytiskne zadaný paket Protokolu 89. Pakety Protokolu 89 je možné uložit do souboru, k tomu slouží funkce `int saveP89toFile(char *name, char zn, TP89 *P89, char *text)`. Funkce saveP89toFile je vhodná hlavně pro ladění komunikace. Jinak není důvod pakety ukládat.

Aplikace *sběrač* využívá funkci `int saveRQtoFile(char *name, TP89 *P89_in, TP89 *P89_out, char *text_in, char *text_out)` pro ukládání nasbíraných dat. V prvním parametru funkce očekává zadané jméno a cestu k souboru, kam bude data ukládat. Paket

odeslaný ke zjištění stavu zařízení se předává v parametru P89_out, odpovědní paket pak v P89_in. V případě chyby nebo timeoutu je možné vyplnit textový popis, který se uloží místo UCB, jinak musí být text_in a text_out NULL.

7.3 Konfigurace aplikace

Konfigurace všech aplikací je uložena v INI souborech. Funkce pro jejich zpracování deklaruje hlavičkový soubor *iniparser.h* a *dictionary.h*. Autorem těchto souborů je N. Devillard. Dokumentaci k iniparseru je možné najít na webu [11]. Je také na přiloženém CD.

O načtení konfigurace se v každé aplikaci stará funkce `int get_config(char *ini_file, AppConfig *config)`. Této funkci se v parametrech předává cesta a jméno k INI souboru s konfigurací a ukazatel na alokovanou paměť pro strukturu `AppConfig` k uložení konfigurace. Podle požadovaných vlastností je struktura pro každou aplikaci jiná a deklaruje se v souboru *main.h*.

7.4 Fronta paketů aplikace MC2TCP

Fronta paketů v aplikaci MC2TCP je typu FIFO. Jazyk C však žádný takový datový typ neposkytuje, proto bylo potřeba implementovat jej vlastními silami. Deklarace struktury a funkcí nad touto frontou jsou v souboru *mc2tcp_queue.h*.

Struktura fronty se jmenuje `MC2TCPqueue`. Do jednotlivých prvků fronty se ukládá odchozí a příchozí paket Protokolu 89, stav operace a descriptor TCP socketu, z kterého původní paket přišel. Položka stav operace může nabývat hodnot:

- `MC2TCP_WAIT` – paket je zařazený do fronty nebo se už odeslal, ale ještě nepřišla odpověď
- `MC2TCP_OK` – paket se odeslal a ve stanoveném intervalu přišla odpověď
- `MC2TCP_TO` – paket se odeslal, ale ve stanoveném intervalu nepřišla odpověď
- `MC2TCP_ERR` – při odesílání nebo příjmu odpovědi nastala chyba

Pro inicializaci struktury se používá funkce `int MC2TCPQInit (MC2TCPqueue *)`, jako parametr se předává ukazatel na danou strukturu. Vzhledem k tomu, že k frontě může přistupovat v jeden okamžik více různých vláken, je potřeba ošetřit současný přístup. K tomuto účelu byly použity semafory. Pro usnadnění práce jsou příkazy ovládající semafory zabaleny do funkcí `void MC2TCPQLock (MC2TCPqueue *L)` a `void MC2TCPQUnLock (MC2TCPqueue *L)`. Ta první zamyká frontu pro ostatní vlákna, druhá ji odemyká.

Vložení nového prvku do fronty se provádí funkcí `struct tMC2TCPelem *MC2TCPQInsertLast (MC2TCPqueue *L, TP89 *P89, int socket)`. Funkce vrací

ukazatel na právě vložený prvek. Pro získání prvního paketu uloženého ve frontě je k dispozici funkce `int MC2TCPQFirst (MC2TCPqueue *L)` a funkce `struct tMC2TCPelem *MC2TCPQGet (MC2TCPqueue *L)`, která vrátí aktivní prvek. Jakmile se vrátí odpověď zpátky na vyslaný dotaz nebo nastane nějaká chyba, je potřeba nastavit v záznamu stav (`int MC2TCPQSetStat (MC2TCPqueue *L, int stat)`), v jakém přišla odpověď. Smazání již nepotřebného prvku je zajištěno zavoláním `int MC2TCPQDeleteItem (MC2TCPqueue *L, struct tMC2TCPelem *prvek)`, kde se funkci předá ukazatel na prvek, který je potřeba smazat. Smazání celé fronty a uvolnění paměti zabezpečuje funkce `void MC2TCPQDispose (MC2TCPqueue *)`.

7.5 SLP

Aplikace SLP je rozdělena na několik vláken, kde každé plní svoji funkci. Hlavní vlákno aplikace se stará o spuštění pomocných vláken a o komunikaci s datovou ústřednou. Na začátku tohoto vlákna se načte konfigurace. Pokud je v konfiguraci povoleno předávání příchozích paketů aplikaci MC2TCP, tak se spustí pomocné vlákno `void *mc (void *attr)`. Podle počtu požadovaných obsluhovaných periferií se pak spustí další vlákna, která je budou obsluhovat. Descriptory socketů jednotlivých logických periferií jsou uloženy v poli `lp_array`.

Po spuštění všech požadovaných vláken se SLP připojí k datové ústředně (funkce `open_tcp`) a zaregistruje se na ni (funkce `reg_du`). Následně s nastaveným timeoutem 10ms čeká na příchozí paket z datové ústředny. Po přijetí paketu se kontroluje adresa cíle (`P89_prichozi.dest`). Podle pravidel se rozhodne, jak s paketem naložit. O pakety určené aplikaci SLP s adresou logické periferie 0 se stará funkce `void *lp_0x00(void *attr)`. Popis LP je v kapitole zabývající se návrhem aplikace.

7.5.1 Vlákno pro komunikaci s MC2TCP

Toto vlákno se po svém spuštění připojí k aplikaci MC2TCP a čeká na příchozí pakety z univerzálního modulu. Pokud paket přijde, provede se kontrola platnosti descriptoru socketu pro spojení s DU a paket se na ústřednu odešle.

7.5.2 Vlákno pro obsluhu logických periferií

Vlákno `void *lp_0xXX (void *attr)` spouští po svém startu nekongruentní TCP server a naslouchá na zvoleném portu. Pokud se připojí nějaká aplikace k serveru, provede se registrace LP do pole `lp_array`, hlavní vlákno tak bude vědět, že pakety určené pro tuto LP má posílat právě přes tento TCP server. Pakety přijaté tímto vláknem se automaticky přeposílají na DU.

7.6 Sběrač

Aplikace sběrače po svém startu načítá matrici čidel (`int readMatrice(char *file, Matrice *matrice)`) z přednastaveného souboru a ukládá jí do jednosměrného seznamu popsaného strukturou `Matrice`. Popis jednoho prvku této struktury je v tabulce 7.2.

Datový typ	**Jméno**	***Popis***
int	type	*Typ záznamu (vstup, výstup, vzorec…)*
int	opakovani	*Počet neúspěšných pokusů spojení s modulem v bezdrátové síti*
unsigned char	*name	*Jméno záznamu*
unsigned char	MC	*Adresát paketu*
unsigned char	LP	*Logická periferie*
unsigned char	RQ	*Číslo požadavku*
unsigned char	L_PARAM	*Počet parametrů*
unsigned char	L_DATA	*Počet datových bytů*
unsigned char	*PARAM	*Parametry*
unsigned char	*DATA	*Datové byty*
struct tChannel	*channels	*Nastavení kanálů*
struct tMatriceItem	*next	*Další prvek matrice*
int	vzorec	*Vzorec pro výpočet kanálu (pro type=A)*
struct tVycet	vycet	*Výčet záznamů pro výpočet vzorce*

Tabulka 7.2: Popis prvku matrice čidel

Jakmile sběrač načte matrici a připojí se k aplikacím MC2TCP a SLP, tak ve stanoveném intervalu odesílá dotazy na všechny moduly, jejichž adresa byla zadaná v matrici čidel. Pokud se podaří korektně stáhnout data ze zadaného čidla, tak je uloží funkcí `saveRQtoFile` do zvoleného souboru, pokud jsou navíc povolené kanály, tak uloží hodnotu do kanálu. Pokud se data stáhnout nepodaří, uloží do souboru řádek s hodnotou *TIMEOUT* nebo *ERROR*. V případě tří neúspěšných pokusů (lze nastavit v konfiguraci) o stažení dat z jednoho čidla se odesílá SMS s informací o chybě.

7.6.1 Kanál

Kanál je soubor, do kterého se ukládá aktuální přijatá hodnota dat z příchozího paketu. Tento soubor se přepisuje při každé nové přijaté odpovědi na dotaz, proto je vhodnější tyto soubory umístit do paměti RAM a neukládat je na USB disk.

7.6.2 Vzorec

Pro některé další specifické aplikace (regulace) je potřeba surová data nějakým způsobem upravit. V následujícím seznamu jsou vypsané používané vzorce.

- 1 – dělení 100
- 4 – nalezení maxima ze záznamů zadaných ve výčtu čidel
- 5 – nalezení minima ze záznamů zadaných ve výčtu čidel
- 6 – průměr hodnot z výčtu
- 7 – rozdíl dvou zadaných čidel
- 8 – diference, rozdíl nové a předchozí hodnoty daného čidla

7.6.3 Ovládání sběrače

Sběrač se po svém startu připojuje k SLP, na kterém je spuštěn TCP server pro logickou periferii sběrače. Pokud chce tedy operátor komunikovat se sběračem, tak na DU zaregistruje jako cílový bod SLP a v UCB vyplní právě tuto logickou periferii. K jaké LP se bude sběrač hlásit lze nastavit v konfiguračních souborech.

Logická periferie sběrače obsluhuje pouze jeden typ požadavku (`int lp_novy_soubor(int fd, const TP89 *P89_prichozi, char *datadir)`). Sběrač na tento požadavek vrací velikost, cestu a jméno souboru, do kterého se ukládají data. Odesílaný požadavek obsahuje jeden parametr a jeden datový byte. V datovém bytu se přenáší číslo bezdrátového modulu, o jehož soubor je zájem. Pokud je parametr roven 1, tak se vytvoří nový soubor pro ukládání dat z daného čidla. V opačném případě se data dál ukládají do stejného souboru.

8 Příprava souborů pro USB disk

Startovací skript v routeru počítá s tím, že se v kořenu USB disku nachází spustitelný soubor `ngapps`. Úkolem skriptu `ngapps` je provést dodatečnou inicializaci síťových funkcí routeru a spustit potřebné aplikace.

8.1 Vestavěný switch

Procesor ADM5120 obsahuje vestavěný pěti portový switch. Ten je možné pomocí nástroje `admswconfig` nastavovat a různé porty mapovat k různým síťovým rozhraním v OpenWrt. Jako první je potřeba smazat původní nastavení switche. To je možné následujícím blokem příkazů.

```
for ETH in 0 1 2 3 4  ; do
  admswconfig eth$ETH
done
```

Pro připojení všech portů do jednoho switche a pro jeho připojení k `eth0` je příkaz `admswconfig eth0 01234c`. K nastavení správné funkce indikačních LED je potřeba nastavit správné hodnoty do souborů v adresáři `/sys/class/leds`. Následující příkazy je potřeba zadat pro všechny LED, u kterých chceme indikaci zapnout.

```
echo "port_state" > /sys/class/leds/wan_lnkact/trigger
echo "link_act" > /sys/class/leds/wan_lnkact/port_state
```

8.2 Konfigurace sítě

Konfigurace síťových rozhraní šla nastavit při sestavování image. To by ale neodpovídalo požadavku na snadnou konfiguraci. Z toho důvodu je konfigurace sítě uložena na USB disku. Soubor s konfigurací sítě stačí pomocí skriptu `ngapps` nakopírovat do adresáře `/etc/config` a uložit jej jako `network`.

V tomto souboru je potřeba definovat rozhraní `loopback` a `eth0`. U `eth0` je navíc potřeba nastavit MAC adresu. Vzorový soubor, který nastaví síťové rozhraní z DHCP serveru je k dispozici na přiloženém CD. Po kopii souboru do správného umístění stačí spustit síť příkazem `ifup -a`.

8.3 Cron

Cron je systémový proces, jehož funkcí je spouštět v určitém čase jiné procesy. Toho se dá využít pro plánování událostí. Soubory pro nastavení událostí spouštěných cronem jsou uloženy na USB disku

v adresáři `cron`. Jejich obsah je možné zkopírovat do souboru /etc/crontabs/root [12] příkazem `cat /disk/cron/* >> /etc/crontabs/root`.

8.4 Spuštění aplikací

Spustitelné soubory aplikací se nachází na USB disku v adresáři `bin`, jejich konfigurace v adresáři `config`. Spuštění požadovaných aplikací se provede přidáním jejich startovacích souborů do skriptu `ngapps`.

9 Závěr

Cílem práce bylo vytvořit zařízení, které by nahradilo původní moduly XPort, sloužící pro připojení bezdrátové sítě modulů k datové ústředně a přidalo další možnosti pro rozšiřování sítě modulů. Tvorbě vlastního zařízení předcházelo seznamování se s principy komunikace s moduly firmy Condata a studium původního řešení.

V první verzi bylo zařízení testováno v areálu firmy Condata, zde sloužilo pouze jako náhrada původních modulů XPort. Po dlouhodobém testování a doplnění aplikace sběrače bylo nasazeno koncem roku 2010 v ostrém provozu ve Fakultní nemocnici v Olomouci, kde sbírá informace o teplotách a dveřních kontaktech chladniček a mrazniček. Za dobu běhu se ukázal velký přínos tohoto zařízení, došlo totiž k několika výpadkům připojení mezi zařízením a datovou ústřednou. Data se však oproti původnímu řešení s XPortem neztratila žádná.

Aplikace pro zařízení se dále vyvíjí. Ve fázi testování je aplikace, která je schopná regulovat topení v rodinném domě a převzít tak původní regulační logiku z nadřazené aplikace. Další vyvíjenou aplikací je aplikace, která komunikuje s elektronickou váhou a ukládá naměřené hodnoty.

Zařízení je také připraveno pro připojení k internetu prostřednictvím GSM/3G modemu. K testování tohoto způsobu připojení je k dispozici modem Huawei E1750.

Práce ve mně vzbudila zájem a rozšířila mé povědomí o znalosti vestavěných systémů. Také mi přiblížila klady a zápory tvorby aplikací komunikujících s hardwarem.

Literatura

[1] Katalogový list procesoru Nordic nRF9E5. *Semiconductor Store.* [Online] 2006. [Citace: 20. srpna 2010.] http://www.semiconductorstore.com/pdf/newsite/nordic/nRF9E5.pdf.

[2] Bifferboard. *Bifferboard.* [Online] 2009. [Citace: 18. září 2010.] http://bifferos.bizhat.com.

[3] OpenWrt Wiki Edimax BR-6104KP. *OpenWrt.* [Online] 2009. [Citace: 20. září 2010.] http://wiki.openwrt.org/toh/edimax/br-6104kp.

[4] *ADMtek.* [Online] 2003. [Citace: 30. září 2010.] http://midge.vlad.org.ua/datasheet/ADM5120%20Ver%201.13%2029Oct03.pdf.

[5] *FTDI Chips.* [Online] 2010. [Citace: 30. září 2010.] http://www.ftdichip.com/Support/Documents/DataSheets/Cables/DS_TTL-232R_CABLES.pdf.

[6] *AMiLDA.* [Online] 2008. [Citace: 8. října 2010.] http://www.amilda.org.

[7] *midge - adm5120 linux mini distribution.* [Online] 2008. [Citace: 8. října 2010.] http://midge.vlad.org.ua/wiki/.

[8] *Squidge.* [Online] 2009. [Citace: 9. října 2010.] http://squidge.sourceforge.net/.

[9] *OpenWrt.* [Online] [Citace: 12. října 2010.] http://www.openwrt.com.

[10] INI file. *Wikipedia, the free encyclopedia.* [Online] http://en.wikipedia.org/wiki/INI_file.

[11] *iniParser: stand-alone ini parser library in ANSI C.* [Online] 2007. [Citace: 17. října 2010.] http://ndevilla.free.fr/iniparser/.

[12] man page crontab section 5. *manpagez.* [Online] 2009. [Citace: 25. října 2010.] http://www.manpagez.com/man/5/crontab/.

Seznam použitých zkratek a symbolů

GNU (*GNU's Not Unix*) – sada svobodného software

SPI (*Serial Peripheral Interface*) – typ sériového rozhraní pro připojení periferií

EEPROM (*Electrically Erasable Programmable Read-Only Memory*) – elektricky mazatelná paměť

TCP (*Transmission Control Protocol*) – transportní protokol

XMODEM – jednoduchý protokol pro přenos souborů

LED (*Light-Emitting Diode*) – polovodičová součástka vydávající světlo

SMS (*Short Message Service*) – služba pro posílání krátkých textových zpráv

Příloha A

Na přiloženém CD jsou zdrojové kódy výsledných aplikací a další potřebné soubory k sestavení zařízení.