TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky a mezioborových inženýrských studií

DIPLOMOVÁ PRÁCE

Radiové rozhraní 433 MHz pro sběr dat z měřičů tepla

433 MHz Radio Interface for Data Acquisition from Heat Consumption Meters

Liberec 2006

Jan Růžička

TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky a mezioborových inženýrských studií

Katedra: měření Akademický rok: 2005/2006

ZADÁNÍ DIPLOMOVÉ PRÁCE

pro: **Jana Růžičku**

studijní program: M 2612 – Elektrotechnika a informatika

obor: 3902T005 – Automatické řízení a inženýrská informatika

Vedoucí katedry Vám ve smyslu zákona o vysokých školách č.111/1998 Sb. určuje tuto diplomovou práci:

Název tématu: **Radiové rozhraní 433 MHz pro sběr dat z měřičů tepla**

Zásady pro vypracování:

1. Prostudujte problematiku radiového přenosu na 433,92MHz, možnosti přenosu dat z měřiče tepla MT500L firmy EESA a základy mikroprocesorové techniky.
2. Pro přípravky firmy EESA vytvořte software s ohledem na dosažení co nejdelší vzdálenosti spolehlivého přenosu.
3. Realizujte vlastní radiové rozhraní s využitím procesoru řady '51 a radiomodulu firmy Aurel.
4. Ověřte funkci radiového rozhraní na komunikaci mezi měřičem tepla MT500L a ručním terminálem ESTER02.

+2 CD

KAM/AKI

49 s., 5 o. příl.

dipl., tab.

V 8106 M

Abstrakt

Vývoj zařízení pro přenos malých objemů dat radiem v pásmu 433 MHz s cílem dosažení co největší vzdálenosti. Implementace fyzické a spojové vrstvy modelu ISO-OSI do procesoru řady '51. Použití FM radiomodulu, přenos v jednom kanálu, řízení master–slave.

Anotace

Radiové rozhraní 433 MHz pro sběr dat z měřičů tepla

Cílem diplomové práce je navrhnout a realizovat elektronické zařízení pro bezdrátový sběr dat z měřičů tepla MT500L do ručního terminálu ESTER02 firmy EESA. Rozhraní používá radiovou frekvenci 433,92 MHz, práce se zaměřuje na optimalizaci vzdálenosti a spolehlivosti spojení za použití komerčně nabízeného FM radiomodulu.

Základem hardwaru zařízení je mikropočítač řady 8051 - SILABS C8051F236 a FM radiomodul AUREL XTR-434L. Klíčovou částí práce je návrh firmwaru v jazyce Assembler. V jednočipovém mikropočítači je implementována fyzická a spojová vrstva modelu ISO-OSI.

Práce se z pohledu fyzické vrstvy zabývá specifiky radiového přenosu dat, vzorkováním signálu, kódováním bitů Manchester a dekódováním algoritmem DPLL (Digital Phase Locked Loop), který zajišťuje bitovou synchronizaci.

Z pohledu spojové vrstvy se práce zabývá použitím preambule pro synchronizaci rámců, detekčních (zabezpečovacích) kódů (např. CRC) a samoopravných kódů (např. Hammingův kód). Dále jde o vývoj protokolu pro adresaci různých slave zařízení, navázání a ukončení spojení a zajištění kontinuity přenosu dat při náhodném krátkodobém výpadku příjmu signálu, např. vlivem šumu nebo rušení. Přenos probíhá v jednom kanálu (poloviční-duplex), přístup je řízen metodou master-slave.

Zařízení propojená tímto rozhraním se připojují linkou standardu RS-232C. Implementace rozhraní se snaží o co možná nejvyšší transparentnost z pohledu koncových zařízení. V ideálním případě by nebylo možno poznat, že koncová zařízení nejsou propojena metalickým vedením. V praktické rovině však postačí, když nebude nutné upravovat protokoly nebo časování v koncových zařízeních.

Dalším zaměřením je vytvořit novou verzi hardwaru rádiového rozhraní. Radiové rozhraní RadioSpoj232 je samostatné zařízení. Verze slave je určena pro instalaci k měřiči tepla. Je na výběr z 15-ti adres. Verze master se používá na straně sběru dat. Uživatel volí požadovanou adresu pomocí tlačítka a displeje. Protože v RS232 není zabudován žádný protokol, je možné RadioSpoj232 použít s jakýmkoliv jiným zařízením, které dodrží parametry spoje RS232.

Abstract

433 MHz Radio Interface for Data Acquisition from Heat Consumption Meters

Development of a device for wireless transmition of small data volumes using a 433 MHz radio band, optimised for maximal communication distance and reliability. Implementation of physical and link layer of ISO-OSI model into an 8051 class microcomputer.

The aim of the thesis is to develop and build an electronic device for data acquisition from the heat consumption meter MT500L to the handheld data terminal ESTER02, both made by the EESA Company.

The radio interface works at 433 MHz. Its hardware is based on a commercial available FM radio module and an 8051 class microcomputer.

The main effort is targeted into development of the software for the microcomputer. The software implements physical and link layer of the ISO-OSI reference model. Physical layer concerns with the specifics of the radio transmition, the signal sampling, the use of the Manchester code and the bit synchronisation while decoding the Manchester using DPLL (Digital Phase Lock Loop) algorithm. The link layer deals with the frame synchronisation using the preamble sequence, the error-detecting codes (i.e. CRC), the self-correcting codes, the master/slave channel access, the addressing of slave devices, initiation of a data link and the reliable delivery. The end devices are connected via RS–232 serial interface. The implementation of radio interface tries to be maximally transparent. In the best case the end devices cannot distinguish they are not interconnected using metallic link. The level of transparency in real operation is sufficient, while there is no need to change protocols or timeouts in the end devices.

The other aim is to develop a new printed circuit board for the radio interface. The radio interface called the RadioSpoj232 is now a standalone device. The slave version is intended for installation at the heat consumption meter with choice of 15 addresses. The master version is designed for use at the side of data acquisition. The user selects desired slave address using a button and a display. As there are no protocols in the RS232, the RadioSpoj232 is applicable to any end device, which uses accurate parameters of RS232 link.

Obsah

Seznam příloh

Elektronické přílohy (na přiloženém CD)

1 Úvod

Firma EESA je výrobcem magnetoinduktivních průtokoměrů a měřičů spotřeby tepla. Měřiče tepla nacházejí uplatnění v bytových domech, kde se jimi měří spotřeba tepla pro vytápění a spotřeba teplé užitkové vody. Zpravidla je v celém domě nainstalován pouze jeden měřič tepla pro všechny bytové jednotky. Dodavatel tepelné energie pravidelně sbírá data z měřičů tepla, hlavně za účelem fakturace dodávek.

Sběr dat se tradičně provádí pomocí přenosného počítače, který se připojí kabelem k průtokoměru. Dalším způsobem sběru dat jsou ruční terminály, které se připojují také kabelem, nebo infraportem. Na popud poptávky ze strany zákazníků, kteří si chtěli ušetřit obtíže spojené s odemykáním několikerých dveří v každém místě odečtu spotřeby, přibyla k ručnímu terminálu i varianta s rádiovým rozhraním. Rádiové rozhraní ale nemělo uspokojivý dosah. Proto byla vyvinuta experimentální deska pro nové rádiové rozhraní s výkonnějším procesorem. K realizaci nového rádiového rozhraní ale nedošlo.

Prvním úkolem této diplomové práce bylo vytvořit nový komunikační software do procesoru na experimentální desce. Druhým úkolem bylo navrhnout a realizovat i vlastní verzi hardware rádiového rozhraní.

Hardware rádiových rozhraní používá radiomodul, který přenáší pouze logický signál, nikoliv synchronizovaná data. Veškeré zpracování signálu, synchronizace a zabezpečení se provádí softwarově v procesoru. Kvalita softwaru má tedy klíčový vliv na funkce a vlastnosti rádiového rozhraní.

Software by měl být optimalizován pro dosažení co největší komunikační vzdálenosti a k dosažení spolehlivého přenosu dat, při kterém se přenášená data neztratí ani nepoškodí. Přenosová rychlost není příliš důležitá, protože se přenáší jen velmi malé objemy dat. Realizace komunikačního software předpokládá rozdělení programu do vrstev. Je potřeba implementovat minimálně fyzickou a spojovou vrstvu.

2 Metody telekomunikací

V této kapitole bych chtěl uvést některé principy používané v telekomunikačních zařízeních, které se týkají mé diplomové práce.

2.1 Referenční model OSI

Referenční model OSI (Open System Interconnnection) definuje 7 vrstev protokolů.

1. Fyzická - médium (kabeláž, modulace, rádiové frekvence atd...), kódování média, ovládání hardware vysílače a přijímače.

2. Spojová - definuje rámec, balíčkování dat, ochrana proti chybám.

3. Síťová, 4. Transportní, 5. Relační, 6. Prezentační, 7. Aplikační.

V průmyslových systémech se běžně uplatňují jen fyzická, spojová a aplikační vrstva. Používání ostatních vrstev je zbytečné jak z hlediska omezeného rozměru sítí, tak z hlediska aplikací.

2.2 Bezdrátové médium

K přenosu signálu bez použití metalických vedení se používá elektromagnetické vlnění, které se šíří volně prostorem. Vlnění není uzavřeno do kabelu a proto musí existovat určité dohody, které umožní používání elektromagnetického vlnění v prostoru více účastníky. Nejstarším rozdělením takového prostoru je vymezení frekvenčních pásem jednotlivým uživatelům. Více uživatelů může používat tentýž prostor aniž by se navzájem rušili, protože používají frekvenční multiplex. Rádiová komunikace je zpravidla regulována vládou a jednotlivá frekvenční pásma jsou uživatelům licencována. Komunikační kanál je vždy vymezen do určitého frekvenčního pásma. Nezanedbatelným aspektem je slábnutí signálu se vzdáleností. Dostatečně vzdálené stanice se navzájem neruší. Komunikační kanál je možné sdílet více uživateli ještě v časovém multiplexu a v kódovém mutiplexu. V časovém multiplexu se jednotlivé stanice střídají při vysílání podle určených pravidel. Kódový multiplex je používán při vysílání v rozprostřeném spektru. Zatímco frekvenční multiplex je naprosto univerzální, časový a kódový multiplex je použitelný jen v rámci jedné technologie.

Za bezdrátové médium použitelné fyzickou vrstvou je možné považovat vstup rádiového vysílače a výstup rádiového přijímače.

Komunikační kanál může být založen na rádiových frekvencích nebo i na světelném vlnění. Známé je použití infračervených diod v dálkových ovladačích nebo amatérský spoj z laserových ukazovátek.

Přenášený signál je v porovnání s pracovním kmitočtem vysílače nízkofrekvenční. Pokud žádná stanice nevysílá, je výstupní signál přijímače nedefinovaný, narozdíl od klidového stavu na metalických médiích.

2.2.1 Analogové modulace signálu

Analogová modulace umožňuje přenášet nízkofrekvenční signál v pásmu umístěném ve vysokých kmitočtech. Nejjednodušší modulací je amplitudová modulace (AM). Dále se velmi často používá frekvenční modulace (FM) nebo fázová modulace (PM). Jedná se vždy o modifikaci základní modulační frekvence - nosné frekvence.

Uvažujme nosný a informační signál:

$$\begin{aligned} &\text{nosný signál:} && Y(t) = Y_m \sin(\omega t + \varphi_0) \\ &\text{informační signál:} && u(t), u \in \langle -1;1 \rangle \end{aligned} \tag{2.1}$$

Modulace pak upravují jeden z parametrů nosného signálu:

- Amplitudová modulace (AM): Přenášený signál určuje amplitudu nosného signálu.

$$Y_m = Y_m(t) = A_0(1 + \alpha \cdot u(t)) \tag{2.2}$$

  α je hloubka modulace.

- Frekvenční modulace (FM): Přenášený signál určuje odchylku frekvence signálu od základního kmitočtu.

$$\omega = \omega(t) = \omega_0 + D_F \cdot u(t) \tag{2.3}$$

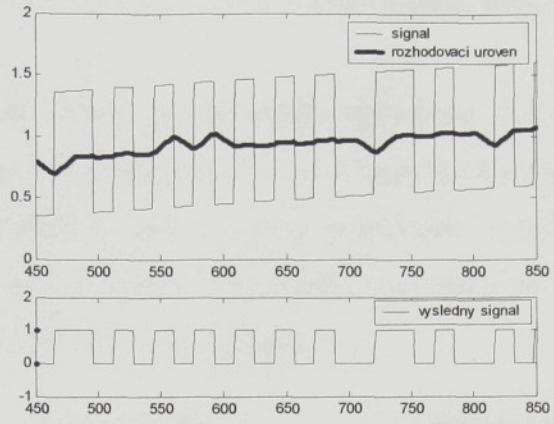
  D_F je modulační zdvih.

- Fázová modulace (PM) : Přenášený signál mění fázi nosného signálu.

$$\varphi_0 = \varphi_0(t) = D_P \cdot u(t) \tag{2.4}$$

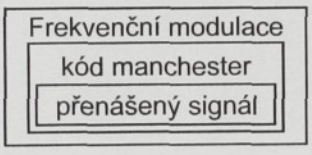
  D_P je největší fázová diference. Vlastnosti fázové modulace jsou velmi podobné vlastnostem frekvenční modulace.

Mohlo by se zdát, že problém přenosu nízkofrekvenčního signálu je vyřešen. Ale díky vlastnostem zvolené modulace nebo nestabilitě analogových obvodů není možné spolehlivě přenést stejnosměrnou složku signálu. Například u amplitudové modulace může signál zeslábnout díky změně vzdálenosti a mohl by být mylně interpretován jako změna úrovně signálu. Pro frekvenční modulaci se obdobně uplatňuje (teplotní) nestabilita oscilátoru. Problém řeší až druhá vnitřní modulace a neustálé kalibrování přijímače podle (střední hodnoty) demodulovaného signálu. Vnitřní modulace bývá již přímo spjata s přenášenými bity a její základní frekvence je odvozena od přenosové rychlosti. Vnitřní modulace se běžně nazývá kódování média.



obrázek 1: Automatické nastavení rozhodovací úrovně

Při přenosu číslicového signálu ve dvou úrovních je vhodné zvolit jako modulaci kód, který má střídu blízkou 1:1. Takovou modulací je kód Manchester.



obrázek 2: Příklad vrstvení modulací

2.3 Kódování média

Ať již je médium jakékoliv, je potřeba definovat způsob, jakým se bude signál po něm přenášený kódovat. Základní kódy jsou binární (používají dvě úrovně signálu). U některých metalických médií se používají kódy ternární, tj. rozlišují se tři úrovně signálu. Hlavním důvodem je odstranění některých nedostatků souvisejících s rušením metalických vedení a možnost opravy chyb. Přenášený signál uvažujeme vždy jako binární.

2.3.1 Informační kapacita kanálu - Shannon-Hartleyova věta

Maximální teoretická informační kapacita kanálu se vypočte podle vzorce

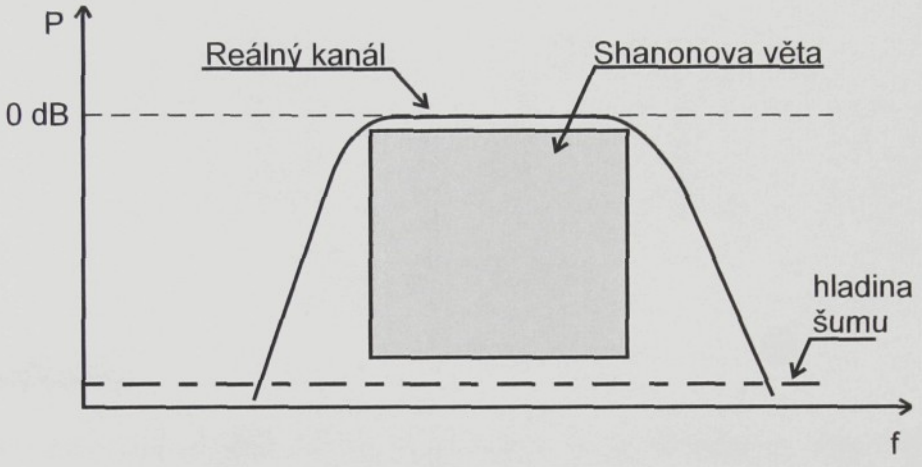
$$C = B\log_2\left(1+\frac{P}{N}\right) \quad [\text{bit/s}], \tag{2.5}$$

kde C je informační kapacita [bit/s], B je šířka pásma [Hz], P je střední výkon signálu a N je střední výkon šumu. [8]

Pokud bychom aplikovali tuto větu na šířku pásma kódu Manchester, kde se frekvence obdélníkového signálu pohybují od $f_{min} = f_{bit}/2$ do $f_{max} = f_{bit}$, je šířka pásma $B = f_{bit}/2$ a tudíž požadovaný poměr signálu a šumu by byl $P/N = 3$, což odpovídá $SNR = 4{,}77$ dB. V praxi je

ale potřeba mít poměr P/N vždy o trochu vyšší a šířka pásma, kterou naše komunikace zabere je také zpravidla větší.

Je zajímavé, že i když bude výkon signálu menší než výkon šumu, je kapacita kanálu stále nenulová. Prakticky dokud je výkon nenulový, je i kapacita kanálu nenulová. Toho využívá princip vysílání v rozprostřeném spektru, který nedostatek vycházející z poměru $P/N < 1$ nahrazuje velmi velikou šířkou pásma. Pro běžné modulace ale musí být výkon signálu alespoň několikanásobně vyšší nežli výkon šumu.



obrázek 3: Reálný kanál a Shannonova věta

Shannon-Hartelyova věta uvažuje šířku pásma jako ostrou hranici a úroveň signálu a šumu jako konstantní v celém rozsahu. Aby bylo kódování efektivní, musí reálný kanál vyplnit co nejlépe. Kódování by zároveň nemělo přesahovat přes hranice dané parametry kanálu.

Podobně definuje kapacitu kanálu Nyquistova věta:

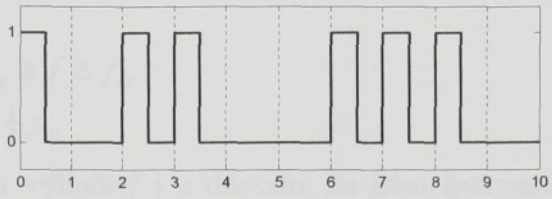
$$C = 2B\log_2(V) \qquad [\text{bit/s}], \qquad (2.6)$$

kde V je počet rozlišitelných úrovní signálu, B je šířka pásma a C je informační kapacita kanálu. Tuto větu lze uplatnit pro kód NRZ i pro Manchester (V=2, => C=2B), nebo pro kód PCM (Pulse Code Modulation), který je běžně používán pro záznam zvuku ($\log_2(V)$ udává počet bitů na vzorek).

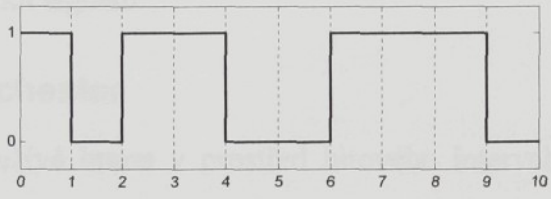
2.3.2 RZ a NRZ

Asi nejjednodušším kódem, se kterým je možné se setkat, je kód NRZ (Non Return to Zero). Hodnotu bitu určuje úroveň signálu v okamžiku vzorkování. Je použit například u RS-232. Druhým známým kódem je RZ (Return to Zero). Hodnotu bitu určuje existence/neexistence pulzu ve sledovaném časovém intervalu. Tyto kódy v sobě přímo nepřenášejí hodinový signál, proto je problematické zajistit synchronizovanost dekodéru. Tato vlastnost se projeví hlavně při přenosu konstantních posloupností (např. "00000000..."). V signálu se neobjevují

žádné hrany a dekodér se nemá s čím synchronizovat. Tento problém se řeší různými algoritmy vkládání bitu (CAN) nebo metodou start-stop bitu (RS-232).



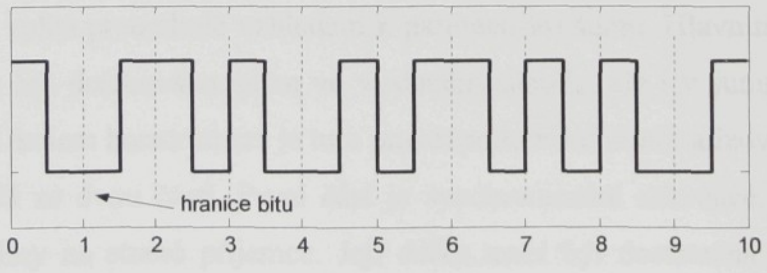
obrázek 4: Kód RZ



obrázek 5: Kód NRZ

2.3.3 Manchester

Kód Manchester narozdíl od kódu NRZ přenáší spolu s daty i hodinový signál. Jde o fázovou modulaci - hodinovému signálu je měněna fáze podle vstupních dat. Změna fáze je o 180° a mění se na hranici bitu. Uprostřed časového intervalu bitu je vždy hrana hodinového signálu. Střída generovaného signálu je 1:1. Data se kódují jako vzestupné nebo sestupné hrany. Různé implementace se liší v definici hodnot hran. Definujme tedy sestupnou hranu (1→0) jako logickou 1 a vzestupnou hranu (0→1) jako logickou 0. Velmi zajímavým vstupním signálem je posloupnost "1 0 1 0 1 0 1 0 ...", protože po zakódování obsahuje signál pouze hodinové hrany. Takového signálu lze s výhodou použít při počáteční synchronizaci přijímače. Naproti tomu nerozpoznatelný signál je pro konstantní posloupnost - nelze zjistit, která hrana je hranicí bitu a která hrana je hodinová.



obrázek 6: Kód manchester - data: 1 0 1 1 0 0 1 1 1 0

V signálu kódu Manchester se hrany objevují velmi pravidelně, v prostřed každého bitu. To umožňuje kvalitně synchronizovat hodiny na straně příjemce. Nevýhodou tohoto kódování

oproti NRZ je potřeba přenášet vyšší frekvence, což v některých případek klade zvýšené nároky na přenosové médium. [1]

Frekvenční rozsahy:

$$\begin{aligned} &\text{pro manchester:} && \tfrac{1}{2} f_{bit} \geq f \geq f_{bit} \\ &\text{pro NRZ:} && f \leq \tfrac{1}{2} f_{bit} \end{aligned} \tag{2.7}$$

Kód Manchester může být výhodný i z důvodu, že jeho frekvenční spektrum je posunuté vzhůru a neobsahuje nízké frekvence. Mohou se pak používat média která nedokáží kvalitně přenést stejnosměrnou složku signálu.

2.3.4 Rozdílový manchester

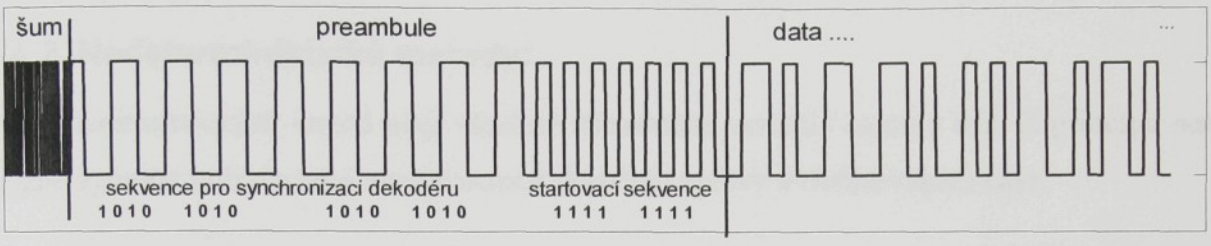
Rozdílový manchester používá hranu v prostřed bitového intervalu pouze k synchronizaci času. Rozhodující je přítomnost nebo nepřítomnost hrany na začátku bitového intervalu. Výhodou oproti obyčejnému manchesteru je zaměnitelnost polarity signálu, protože se nedetekuje směr hrany ale pouze její přítomnost. Odpovídající nesená data k předchozímu obrázku by byla "x001010110" (hodnotu prvního bitu není možné zjistit). [6]

2.3.5 Synchronizace začátku rámce pomocí preambule

V radiové komunikaci neexistuje klidový stav sběrnice známý z komunikace po metalických vedeních. Pokud žádná stanice nevysílá, je v příchozím signálu pouze šum. Tento šum vstupuje do dekodéru, který produkuje náhodnou posloupnost bitů. Jediné parametry, které lze sledovat jsou střída (nebo střední hodnota) a frekvenční spektrum (rozptyl je u binárního signálu zbytečné vyhodnocovat). Díky všudypřítomnému šumu nelze rozpoznat začátek vysílání detekcí hrany. Jedinou možností zjištění začátku vysílání je určit startovací sekvenci - preambuli. Preambule se používá u metalických stejně jako u bezdrátových médií k synchronizaci hodin dekodéru příjemce a k označení začátku přenášených dat. U radiových přenosů je důležitá volba preambule vzhledem k parametrům šumu. Hlavním problémem je, že preambule může být detekována nejen ve vyslaném signálu, ale i v šumu, a to s určitou pravděpodobností. Úkolem konstruktéra je tuto pravděpodobnost minimalizovat.

Preambule se skládá ze dvou částí. První část je synchronizační sekvence. Ta má za úkol synchronizovat hodiny na straně příjemce. Její délka musí být dostatečná k tomu, aby se hodiny příjemce stihnuly synchronizovat se signálem ať byly původně v jakékoliv fázi vzhledem k signálu. V kódu manchester jde typicky o bitovou posloupnost "...10101010", která produkuje obdélníkový signál, jehož hrany jsou hodinovými tiky hodin přenášených v signálu.

Druhou částí je startovací značka. Tato značka musí být jednoznačně rozpoznatelná od jakékoliv části synchronizační posloupnosti. U sítí Ethernet stačí pouze dvoubitová značka "11".



obrázek 7: Preambule

Vzhledem k tomu, že při dekódování šumu vznikají náhodné posloupnosti, je nutné je analyzovat a zvolit preambuli takovou, která bude mít dostatečně nízkou pravděpodobnost výskytu.

2.4 Sdílené médium

Za sdílené médium lze považovat takové, ke kterému může přistupovat více vysílačů. Může to být segment Ethernetové sítě v poloduplexním režimu, sběrnice RS-485, lokální sběrnice počítače nebo kanál rádia. Obsazením média se rozumí vysílání. Jestliže se pokusí dva systémy obsadit médium v jeden čas, dojde ke kolizi. Kolize má za následek zpravidla jen porušení přenášené zprávy, ale u některých systémů může dojít i k poškození hardwaru vysílačů. Kolizní doména je množina systémů (uzlů), které jsou připojeny ke stejnému médiu. Pokud mají spolu uzly v kolizní doméně rozumně komunikovat, musí používat některý z algoritmů sdílení média, tzv. řízení přístupu ke sdílenému médiu.

2.4.1 Deterministické metody:

U deterministických metod je zajišten provoz bez kolizí. Jsou proto vhodné pro sítě kde by při kolizi došlo ke zničení hardwaru. Nevýhodou je (například u rádiového provozu) režie spojená s aplikací metody. K vysílání dochází i když není potřeba přenášet data.

2.4.1.1 Master/Slave

Volně přeloženo nadřizený/podřízený. Podřízený systém smí obsadit médium vysíláním jen je-li dotázán nadřízeným systémem a jen na omezenou dobu. Nadřízený systém smí obsadit médium pokud neočekává odpověď od podřízeného systému. Nadřízený systém bývá zpravidla jeden, počet podřízených systémů není tímto principem omezen.

2.4.1.2 Token passing

Systémy si mezi sebou předávají pověření k vysílaní. Držitel pověření může po určitou dobu vysílat a po vypršení času musí pověření předat dalšímu systému.

2.4.2 Nedeterministické metody:

U nedeterministických metod mají všechny zúčastněné systémy stejnou roli. Z principu není možné vyloučit kolizi a není možné zaručit doručení zprávy v definovaném čase.

2.4.2.1 ALOHA

Jde o nejjednodušší nedeterministickou metodu. Systém může vysílat kdykoliv, jedinou podmínkou je použití kontrolní sekvence, zpravidla CRC, aby bylo možné detekovat porušenou zprávu na straně příjemce. Tato metoda funguje dobře jen na sítích s malým provozem, kde je i malá pravděpodobnost kolize a tím i malá pravděpodobnost nedoručení zprávy. [1]

2.4.2.2 CSMA

Metoda CSMA (Carrier Sense Multiple Access) je vylepšením metody ALOHA o příposlech (carrier sense). Systém sleduje médium jestli není obsazeno jiným vysílačem. V případě že není, začne sám vysílat. Pokud chtějí začít vysílat dva systémy ve stejný čas, oba shodně zjistí, že médium je neobsazené, začnou vysílat a dojde ke kolizi.
Metody ALOHA a CSMA neumí detekovat kolizi a proto nemají možnost opakovat vysílaní zprávy. Kolizi zjistí až příjemce tím, že mu nevyjde kontrolní sekvence. Opakované vysílání zprávy je spíše věcí nadřazeného protokolu.

2.4.2.3 CSMA/CD

Tato metoda rozšiřuje metodu CSMA o detekci kolize (Collision Detection). Lze potom zavést opakování zprávy na úrovni spojové vrstvy OSI modelu. Znovuodeslání zprávy není možné provést okamžitě po detekci kolize, protože by mohlo dojít k další kolizi. Doba čekání před dalším odesláním musí být různá. Algoritmus čekání může být založen na generátoru náhodných čísel nebo na prioritě adres (v případě velmi malého adresního prostoru).

2.4.2.4 CSMA/CA

Metoda detekce kolize s arbitrací (zkratka z Collision Arbitration nebo Collision Detection with Arbitration) patří svými vlastnostmi blíže deterministickým metodám. Lze ji

implementovat jen na síti, kde je čas bitu delší nežli doba šíření signálu po síti. Signál jdoucí po síti je buď dominantní (aktivní úroveň) nebo recesivní (neaktivní úroveň). Arbitrace spočívá v prioritě adres, které se vysílají na začátku zprávy. Pokud systém vysílá recesivní bit adresy, ale zpětně odposlechne bit dominantní, musí se odpojit. Tuto metodu používá sběrnice CAN.

Metody CSMA/CD a CSMA/CA jsou závislé na použitém hardwaru. Nelze je implementovat tam, kde není možné zpětně odposlouchávat vlastní vysílání. Takovým případem je i rádio.

2.5 Kódová zabezpečení

Do rádiové komunikace vždy vstupuje šum. Jeho hodnoty se zvětšují hlavně s rostoucí vzdáleností vysílače a přijímače nebo v souvislosti s umístěním překážek v přímé viditelnosti. V praxi nelze nikdy zaručit spolehlivý přenos. Chyby se vyjadřují pravděpodobností jejich výskytu, tzv. BER (Bit Error Ratio). Vhodným použitím kódů lze pravděpodobnost výskytu chyby velmi snížit. Některé kódy dokážou chyby pouze detekovat, jiné dokáží chyby opravit. Detekční i opravné kódy v sobě nesou určitou redundanci, tzn. že se spolu s užitečnou informací přenáší i další data pro kontrolu nebo pro opravu chyb. Čím lepší schopnosti od kódu požadujeme, tím více redundantní informace bude potřeba přidat. Samoopravné kódy mají obecně větší nároky na nadbytečnou informaci, než kódy detekční.

Samoopravné kódy se používají hlavně při jednosměrné komunikaci (digitální televize nebo rádio). Tam kde je zavedena obousměrná komunikace je výhodnější chyby jen detekovat a chybu opravit pomocí protokolu zopakováním zprávy.

Teorie kódů předpokládá: 1) vždy budeme schopní rozpoznat začátek kódového slova, 2) délka zprávy se nezmění, 3) pozice cifer se nezmění, tj. nemůže dojít k chybě v synchronizaci a 4) chyba je náhodná (opakem je chyba shluková). Tyto předpoklady jsou v rozporu s praxí, protože k poruchám synchronizace v zašuměném signálu dochází běžně a chyby rozhodně nemusejí být náhodné (v průmyslovém prostředí, elektromagnetické rušení jinými spotřebiči). Hlavně z důvodů synchronizace není vhodné používat detekční a opravné kódy na úrovni fyzické vrstvy modelu OSI. Používání samoopravných kódů má smysl až když je účinně zajištěna synchronizace bitu a rámce.

Dodatečným efektem používání detekčního kódu je snížení pravděpodobnosti falešné interpretace šumu jako platné zprávy. Redundantní data jakoby prodlužovaly preambuli. Přesto ale není možné vyloučit takovouto interpretaci a proto doporučuji mít v připojeném zařízení (průtokoměru nebo dataloggeru) takový software, který neumožní měnit důležité

vnitřní stavy zvenčí (konstanty nebo naměřená data). Protože při sběru dat příliš nezáleží na rychlosti, je lepší využít redundantní informaci raději k detekci, nežli k opravě.

2.5.1 Opakovací kód

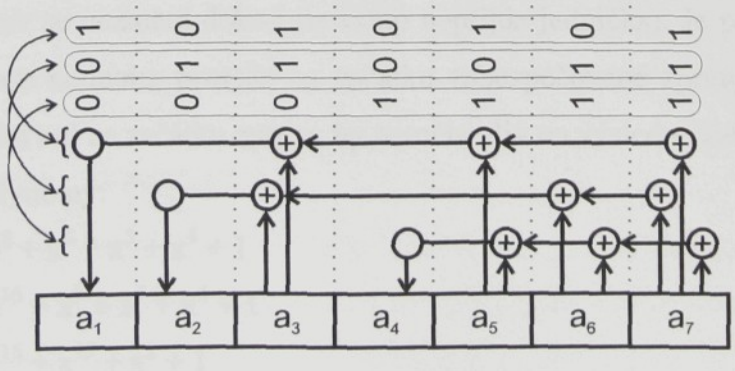
Opakovací kód je asi nejjednodušším detekčním a opravným kódem. Používá dvě slova, která obsahují n-krát zopakovanou původní cifru. Dokáže detekovat n - 1 a opravit $\frac{1}{2}(n-1)$ chyb, kde n je délka kódu. Chybně přijaté slovo lze opravit operací majority, tj. početní převahou. Opravy slova na základě opakovacího kódu se uplatní v dekodéru kódu manchester.

2.5.2 Hammingův kód

Hammingův kód umožňuje detekci až dvou chyb nebo opravu jedné chyby. Je založen na použití vícenásobné parity. Jeho délka je $n = 2^m - 1$, kde m je počet kontrolních bitů. Rozšířením kódu o celkovou paritu vznikne **rozšířený hammingův kód**, který dokáže detekovat až tři chyby nebo opravit jednonásobnou chybu. Tento kód má délku $n = 2^m$, což je výhodné při umisťování do paměti počítače. [2]

Generování Hammingova kódu ukážu na Hammingově (7, 4)-kódu. Tento kód má 4 informační a 3 redundantní (vypočitatelné) bity. Informační bity jsou na místech 3, 5, 6 a 7, kontrolní bity jsou na místech 1, 2 a 4. Bit a_1 je sudou paritou bitů a_3, a_5 a a_7. Jsou to bity, které mají v binárním vyjádření svojí pozice jedničku na místě jednotek (LSB). Bit a_2 je paritou bitů a_3, a_5 a a_7. Tyto bity mají v binárním vyjádření pozice jedničku na místě desítek (pozice uprostřed). Bit a_4 je parita bitů a_5, a_6 a a_7. Jsou to bity, které mají v binárním vyjádření pozice jedničku na místě stovek (MSB). Obdobně by se daly generovat i Hammingovy kódy jiných délek. [7]

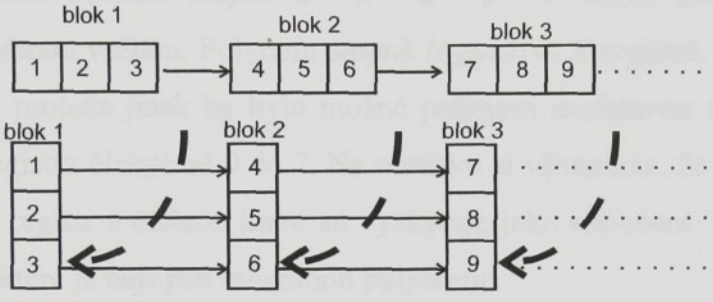
Na obrázku je schéma generování Hammingova (7, 4)-kódu, které by bylo možné snadno převést do podoby číslicového obvodu. Znak ⊕ zde znamená operaci nonekvivalence (XOR).



obrázek 8: Schéma generovaní Hammingova kódu

2.5.3 Oprava shlukových chyb prokládáním bloků

Předpokládejme, že budeme chtít vysílat zprávy o maximální délce 26 bajtů a že budeme chtít opravit shlukovou chybu o délce osmi bitů. Při tom můžeme použít Hammingův kód, který opravuje jednu chybu. Jednoduše se použije 8 bloků (31, 26)-kódu. To ale samo o sobě neumožňuje opravit shlukovou chybu, protože když je porušeno více jak jeden bit v bloku, už nebude možná oprava. Proto se bloky rozdělí po jednom bitu a vysílají se prokládaně. Shluková chyba o délce do 8-mi bitů potom poruší více bloků, které ale půjdou opravit.



obrázek 9: Prokládání bloků

2.5.4 Cyklické kódy a CRC

Teorie cyklických kódů byla rozvinuta v padesátých letech kvůli nenáročnosti jejich implementace do hardwaru. Hardwarově se realizují pomocí posuvných registrů a hradel XOR. Používají se pro počítání kontrolní sekvence CRC (Cyclic Redundancy Check) nebo pro práci s jinými kódy, které jdou převést na cyklické. V teoretické rovině se kódy zapisují v podobě (generujícího) polynomu. Na volbě polynomu závisí všechny vlastnosti kódu. Cyklické kódy se dají použít i pro generování pseudonáhodných sekvencí. Dobrý polynom generuje 2^n - 1 stavů, kde n je délka registru. Jinými slovy to znamená že perioda opakování je maximální možná a že se v posuvném registru vystřídají všechny možné stavy, kterých může dosáhnout. Kritický je stav, ve kterém mají všechny registry hodnotu 0 (protože XOR(0, 0) = 0 - stav se nezmění dokud na vstup nepřijde jednička). Je proto vhodné registr přednastavit na jinou hodnotu, protože na začátku zabezpečované zprávy mohou být nuly, přičemž změna počtu nul na začátku zprávy by neměla vliv na výsledné zabezpečovací slovo. Některé vhodné polynomy:

$$P_{8\text{-bit}} = x^8 + x^6 + x^5 + x^4 + 1$$

$$P_{16\text{-bit}} = x^{16} + x^9 + x^7 + x^4 + 1$$

$$\text{CRC-16} = x^{16} + x^{15} + x^2 + 1$$

$$\text{CRC-CCITT} = x^{16} + x^{12} + x^5 + 1$$

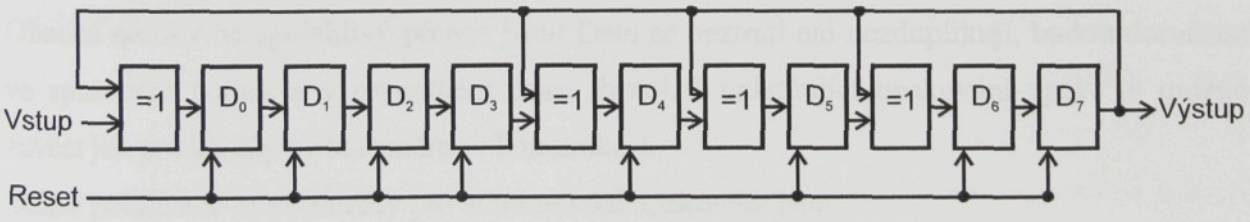
$$\text{CRC-32} = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

Kontrola CRC o délce n dokáže odhalit všechny jednonásobné chyby, lichý počet chyb(jako parita) a shlukové chyby v délce menší nežli n. Pokud je délka zabezpečované zprávy menší než 2^n - 1 bitů (pro $n = 16$ je to 65535) pak detekuje i všechny dvojchyby. Ostatní chyby detekuje s pravděpodobností $P = 1 - \frac{1}{2^n}$ ($P_{16\text{-}bit} = 65535/65536 = 0{,}99998$).

2.5.4.1 Hardwarová realizace CRC

Na příkladu polynomu osmého stupně $x^8 + x^6 + x^5 + x^4 + 1$ ukážu konstrukci posuvného registru s lineární zpětnou vazbou. Polynom stupně N používá N registrů. Členy x^N a 1 ($= x^0$) jsou přítomny vždy, protože jinak by bylo možné polynom zredukovat na polynom nižšího stupně. Jednotlivé registry čísluji od 0 do 7. Na obrázku si všimněme, že výstup hradla XOR je vždy připojen na registr s číslem, které se vyskytuje jako exponent polynomu. Jedinou výjimkou je číslo 8, které je nejvyšší mocninou polynomu.

Před započetím výpočtu kontrolního slova se musí celý registr přednastavit. Na obrázku používám signálu "Reset", ale výhodnější je natavení na jedničky kvůli již výše zmiňované možnosti detekovat chybu v počtu nul na začátku zprávy. Po přednastavení přichází signálem "Vstup" zpráva bit po bitu tak jak je posílána. Když zpráva skončí, přidá se na vstup dalších N (v našem případě 8) bitů hodnoty 0 a na výstupu se zároveň přečte N bitů kontrolního slova. Jiný způsob přečtení kontrolního slova je přímé čtení z registru. Dostaneme pak sice jiné slovo než v prvním případě, ale míra zabezpečení zůstává stejná. Zjištění, zda doručená zpráva je v pořádku, se provádí prostým porovnáním vygenerovaného a přijatého kontrolního slova.



obrázek 10: Zjednodušené schéma hardwarové realizace CRC

2.5.4.2 Softwarová realizace CRC

Celý algoritmus je obdobou hardwarového přístupu. Používá operaci bitového XOR (nonekvivalence) a operace bitových posuvů, které jsou ekvivalentní operace jako násobení nebo celočíselné dělení číslem 2. Je důležité uvědomit si, jak funguje operace XOR. Pokud bychom uvažovali první vstup jako řídící a druhý jako signálový, pak řídící vstup říká, zdali

se na signálu provede negace nebo nikoliv. Algoritmus pak používá bitovou operaci XOR spíše jako selektivní negátor.

Konstanta polynomu se konstruuje obdobně jako v případě schématu hardware. Od nejnižšího bitu se zapisují konstanty polynomu od nejnižší mocniny. Člen v nejvyšší mocnině polynomu se nezapisuje. Používám stejný polynom jako v předchozím případě. Pro jednoduchost neuvádím funkce přednastavení registru ani funkci čtení výsledku.

```
konstanta Polynom= 01110001 b
Proveď 8x:{
 vsupní_Bit <= vstupní_slovo(0)
 posuň vstupní_slovo doprava
 zpětnovazební_Bit <= REGcrc(7)
 posuň REGcrc doleva
 REGcrc(0) <- vstupní_Bit
 Když (zpětnovazební_Bit ==1):{ REGcrc <= REGcrc XOR Polynom }
}
```

2.6 Metody spolehlivého přenosu

Chybovost datového kanálu se vyjadřuje číslem BER (Bit Error Ratio), u rádia se může měnit podle podmínek nebo vlivem provozu jiných stanic na stejné frekvenci. Za všeobecně přijatelné se považuje BER $\leq 10^{-6}$. Někdy se definuje číslo FER (Frame Error Ratio), které vyjadřuje poměrný výskyt závadných rámců. Při nízkém BER nemá cenu zavádět do rámců samoopravné kódy. Naopak detekční kód by měl být použit vždy, aby bylo možné vadné rámce vyloučit.

Služba bez záruky doručení (best-effort) pouze detekuje chyby a vylučuje vadné rámce. Spolehlivá služba musí zavést protokol, který umožní opakování nedoručených rámců. Obecné nároky na spolehlivý přenos jsou: Data se neztratí ani nezduplikují, budou doručena ve správném pořadí a v rozumném čase. Protokol uplatňující opakování zpráv je možné zavést jen pro kanály s obousměrnou komunikací.

Běžně používané mechanizmy pro automatické opakování jsou:

- Čekání na potvrzení každého datagramu (Stop and Wait): Vysílač vyšle jeden rámec a čeká na jeho potvrzení. Pokud potvrzení nepřijde do vypršení času na odpověď, odešle rámec znovu. Přijímač musí uchovávat odeslaný rámec, dokud není potvrzen.
- Opakování od N-tého datagramu: Vysílač vysílá číslované rámce a současně očekává potvrzení. Pokud přijímač zjistí, že dostal rámec, který nenavazuje na poslední přijatý, odešle zpět žádost o zopakování všech rámců počínaje prvním nepřijatým. Pokud přijímač neodpovídá, může vysílač opakovat výzvy k potvrzení. Vysílač musí uchovávat všechny nepotvrzené rámce, přijímač nemusí uchovávat žádný.

- Selektivní opakování: Vysílač vysílá číslované rámce a očekává potvrzení jako v předchozím případě. Když přijímač zjistí, že některý rámec nedorazil, odešle zpět žádost o zopakování chybějících rámců. Přijímač musí uchovávat všechny rámce, které následují po dosud nedoručeném rámci, vysílač uchovává všechny nepotvrzené rámce. Pokud přijímač neodpovídá, může vysílač opakovat výzvy k potvrzení. [1]

2.7 Programovací techniky pro mikropočítače

V této kapitole bych chtěl uvést některé struktury a způsoby pro zefektivnění chodu programu. Výběr technik vychází z použití osmibitového procesoru 8051.

2.7.1 Vyhledávací (Look-up) tabulka

Při dekódování Manchesteru používám operaci majority bitů v bufferu ke zjištění hodnoty přijímaného bitu. Mohl bych vytvořit cyklus, ve kterém otestuji každý bit a případně přičtu jedničku k čítači. Tento postup ale zabírá příliš mnoho času. Daleko rychlejší je rovnou přečíst výsledek z tabulky. Rozměry tabulky jsou ale omezené velikostí paměti. Navíc v osmibitovém procesoru lze adresovat jednoduše jen tabulku do délky 256 bajtů. Takže pro výpočet majority v 32 bitech je nutné zvolit kombinovaný postup, ve kterém se použije tabulka a sčítání. Majorita se pak zjistí porovnáním. Vyhledávací tabulka je tedy zrychlení výpočtu za cenu zabrání paměti.

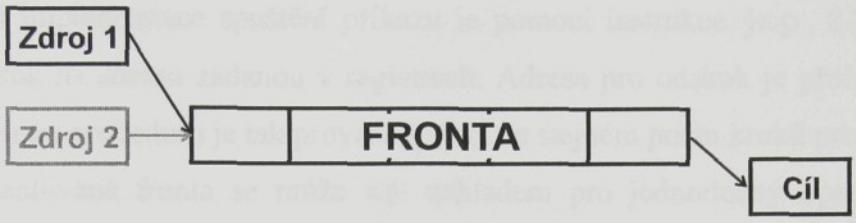
2.7.2 Vyrovnávací paměť FIFO - fronta

Fronta - FIFO paměť - je obecně dobře známá struktura. Lze ji ale implementovat několika způsoby s různým počtem proměnných. Základními proměnnými jsou pointer čtení (Ptr_R) a pointer zápisu (Ptr_W). Základní metody jsou zápis na konec (Add) a čtení ze začátku (Get). Bez přidání dalších proměnných bude muset být vždy alespoň jeden prvek nevyužit. Pokud přidáme proměnnou pro obsazenost paměti (počet zabraných míst, indikace prázdné paměti nebo indikace zcela zaplněné paměti), mohou být použity všechny prvky.

Zajímavým použitím fronty je předávat pomocí ní data mezi režimem obsluhy přerušení (který nastává asynchronně) a přerušitelným programem. Fronta bude takto použitelná jen pro jeden směr a pro jednu úroveň přerušení. Rovněž metody pro přístup k frontě musí být napsány s ohledem na zachování konzistence dat v případě přerušení jejich vykonávání. Operace probíhají ve sledu: 1) test zaplněnosti - 2) čtení/zápis dat - 3) aktualizace pointerů. Funkce přidávání posouvá pouze pointer zápisu a funkce odebrání posouvá jen pointer čtení. Pokud je použita proměnná pro zaplněnost, kterou aktualizují obě metody, musí se měnit

pouze atomickými (=nepřerušitelnými) operacemi, například instrukcemi `inc`, `dec` nebo pomocí dočasného vypnutí všech přerušení.

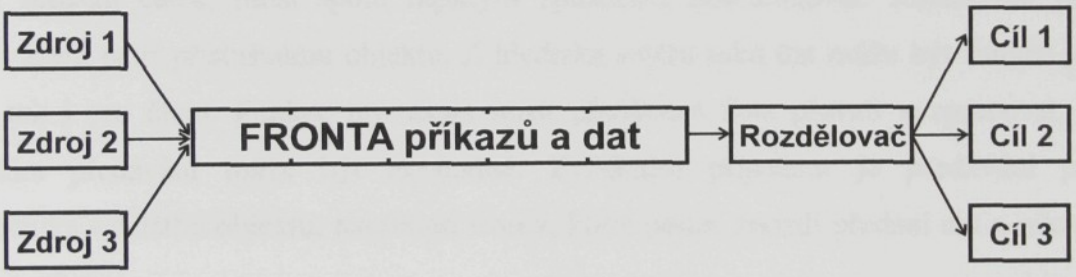
Obyčejná fronta je schopna předávat data jen jednoho významu, a proto lze říct, že může mít jen jeden cíl, který data odebírá a zpracovává. Více cílů by připadalo v úvahu jen u paralelního zpracování stejného typu dat.



obrázek 11: Fronta

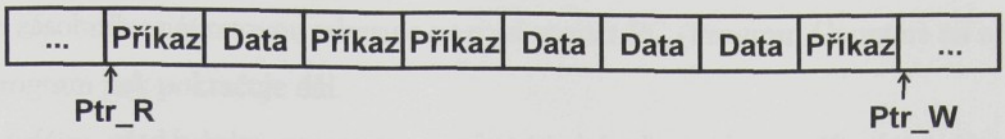
2.7.3 Fronta příkazů a dat

Fronta příkazů a dat je rozšířením fronty o možnost jejího sdílení různými procedurami. Slouží především k vnitřní komunikaci mezi procedurami, které se spouštějí asynchronně. Může jít o předávání dat mezi obsluhou přerušení a procedurami běžícími v programovém cyklu nebo i o plánování úloh procedurami programového cyklu. Data ve frontě nemají jednotnou interpretaci jako v případě obyčejné fronty. Jedinou společnou vlastností je velikost prvku fronty.



obrázek 12: Fronta příkazů a dat

Obsah fronty připomíná instrukční kód mikroprocesoru. První prvek je číslo příkazu (procedury), který se má spustit. Příkaz potom musí z fronty vybrat všechna data, která mu patří. Množství dat náležejících k příkazu není principielně omezeno. Veliký důraz musí být kladen na správné přidávání a odebírání prvků fronty, protože vznik nekonzistence má za následek selhání celého programu (data jiného příkazu by byla interpretována jako číslo příkazu).



obrázek 13: Fronta příkazů a dat

Použití této fronty může ušetřit paměť počítače, protože dochází ke sdílení prostředků. Nejefektivnější implementace spuštění příkazu je pomocí instrukce `jmp @A+DPTR`, která umožňuje odskok na adresu zadanou v registrech. Adresa pro odskok je přečtena z look-up tabulky. Odskok na proceduru je tak proveden vždy ve stejném počtu kroků pro každý příkaz. Takto implementovaná fronta se může stát základem pro jednoduchý operační systém s multitaskingem.

2.7.4 Softwarové rozhraní - Komunikace mezi programovými objekty

Koncipování programu jako množiny objektů je moderní a efektivní přístup. Používáním oddělených objektů se program stává přehlednějším. Každý objekt si řeší svoji část problému a s ostatními objekty musí zachovávat pouze definované rozhraní. Objekty nemusejí být nutně alokovány dynamicky, jak je to známé z platformy osobních počítačů. Jazyk Assembler rozhodně není objektově orientovaným jazykem, ale vhodným rozčleněním do modulů lze dosáhnout rozdělení programu na skupinu staticky alokovaných objektů. Aby tyto objekty tvořily funkční celek, musí spolu nějakým způsobem komunikovat. Rozhraním je vždy funkce, která patří příslušnému objektu. Z hlediska směru toku dat může být funkce jak pro zápis, tak i pro čtení. Funkce pro zápis musí předávaná data převzít a zpracovat, což v okamžiku předávání může být nevhodné. Zvláštním případem je předávání pomocí prostředníka - dalšího objektu, například fronty, který pouze zpozdí předání dat a nikterak je nezpracovává.

2.7.5 Předávání parametrů při volání funkcí v jazyce Assembler

Ve vyšších programovacích jazycích se předávané parametry píši většinou za jméno volané funkce do kulatých závorek, oddělené čárkou a dále se programátor nemusí starat, snad kromě nastavení kompilátoru, kterou z možností předávání bude při překladu preferovat.

Volání procedury nebo funkce je v jazyce Assembler (ale i v jazyce C) totožný pojem. Použije se instrukce `CALL pevná_adresa` a tím je funkce spuštěna. Procesor při zpracování této instrukce uloží návratovou hodnotu na vrchol systémového zásobníku a odskočí na zadanou pevnou adresu. Pro návrat z funkce se použije instrukce `ret`, která

vybere ze zásobníku návratovou adresu a nastaví registr PC (Program Counter) na návratovou adresu. Program pak pokračuje dál.

Nejjednodušším předáváním parametru i návratové hodnoty je použít akumulátor (registr ACC). Takto je možné předat pouze osmibitové číslo, což může být málo, ale ve většině případů to stačí. Další výpomocí může být stavové slovo procesoru PSW, ve kterém se dá využít bit C (Carry flag) nebo další flagy. Takovéto předávání je výhodné hlavně protože programátor nemusí dbát na zálohování dat.

Druhou možností je používat 3. registrovou banku pro předávání parametrů (nultou běžně používá programová smyčka, 1. banku používá přerušení s nízkou prioritou a 2. banku používá přerušení s vysokou prioritou). Je tedy možné předávat až 1 + 8 osmibitových parametrů, navíc operace s registry bývají rychlejší nežli operace v RAM paměti. Pro předávání parametrů lze použít libovolně i staticky alokovanou paměť. Je ale potřeba si dát pozor, aby obsluhy přerušení při volání funkcí používaly jiný paměťový prostor.

Třetí často využívaná možnost je použití systémového zásobníku. V zásobníku lze předat virtuálně neomezený počet parametrů. Problém může nastat s kapacitou zásobníku, s rychlostí přístupu a s nutností vybírat parametry postupně.

Čtvrtou možností je předávat v akumulátoru ukazatel na data. Na platformě 8051 tato technika nemá praktický význam, protože se operační paměť obyčejně alokuje staticky.

Všechny techniky lze využít jak při volání procedury, tak pro návratové hodnoty. V pojetí jazyka C je návratová hodnota pouze v šířce slova procesoru a tedy nejčastěji se pro větší data používá pointer nebo sdílená paměť.

2.7.6 Jednoduchý protokol pro RS232

Po lince RS232 se nejčastěji přenáší sedmibitové (ASCII) nebo osmibitové znaky. Pokud je potřeba odlišit povely od dat, používá se k řízení cílového zařízení buď vyhrazení speciálních funkčních znaků (např. řídící znaky ASCII) nebo zvláštní protokol (např. AT příkazy telefonního modemu).

Pokud je potřeba používat celý osmibitový rozsah pro data, je tu ještě možnost využívat paritní bit jako informační. Je totiž možné nastavit paritu na fixní hodnotu - mark nebo space parity. Takto se dá rozdělit prostor 512-ti znaků na 256 datových a 256 řídících. Terminálové programy běžně nepodporují devítibitový přenos a umožňují jen na výběr z mark nebo space parity. Pokud se parita příchozího znaku neshoduje, je znak smazán. Pro tyto terminály není možné používat příkazy, ale stále mohou přenášet data. Pokud se 9. bit pro datové znaky

nastaví na hodnotu 'mark', může být druhé zařízení nastaveno i na osmibitový provoz, protože interpretuje 9. bit jako stop-bit.

3 Hardware radiového rozhraní

3.1 Systémy firmy EESA

Firma EESA je výrobcem magnetoinduktivních průtokoměrů a měřičů tepla, které se uplatňují v průmyslovém prostředí a pro měření spotřeby teplé užitkové vody a tepla v bytových domech. Jako doplňkový sortiment dodává i programy a ruční terminály ke sběru dat.

3.1.1 Sběr dat z měřiče tepla MT500

Měřič tepla MT500 lze osadit několika druhy komunikačních modulů. Je na výběr z několika druhů rozhraní, mezi něž patří datové sběrnice RS485, M-bus, RS232 nebo analogové výstupy 0 - 10 V nebo 4 - 20 mA. Za účelem sběru dat plně stačí použít sběrnici RS232. Pro komunikaci po RS232 se používá protokol SIMPLE. Jde o velmi jednoduchý protokol, kdy se měřiči tepla zašle dotaz na hodnotu registru a je vrácena aktuální hodnota jako číslo ve formě ASCII řetězce. Data jsou v obou směrech zabezpečena checksumem. Dotaz bývá dlouhý asi 5 bajtů, u návratové hodnoty lze předpokládat délku závislou na přesnosti čísla s plovoucí čárkou, prakticky do 20-ti bajtů. Parametry spojení jsou standardně nastavovány na 9600 baud, 8bit, bez parity, 1 stop bit. [5]

Ke sběru dat se používá buď přenosný osobní počítač s programem VIEW nebo VISIKAL, nebo ruční terminál ESTER. Sběrné zařízení se propojuje s průtokoměrem pomocí kabelu nebo infračerveným portem, sběr dat tedy vyžaduje vstup do prostor s měřičem tepla. Ve speciálních případech lze port RS232 připojit k telefonnímu nebo GSM modemu a sbírat tak data vzdáleně.

3.1.2 Předchozí vývoj radiových rozhraní firmy EESA

Vývoj radiových rozhraní byl vyvolán poptávkou ze strany zákazníků, kteří si chtěli ušetřit obtíže spojené s odemykáním několikerých dveří v každém místě odečtu spotřeby. Prvním radiovým rozhraním uvedeným do praxe bylo zařízení RADKA 1.0 dodávané i jako součást ručního terminálu ESTER. Tento modul má ale poměrně krátký dosah, který byl přisuzován použitému radiomodulu fy. Radiometrix. Proto bylo vyvíjeno další radiové rozhraní RADKA 2.0 osazené nejprve lepším radiomodulem fy. AUREL a později i výkonnějším procesorem fy. Cygnal (dnešní Silabs), pomocí kterého by mohly být implementovány

sofistikovanější algoritmy. Na tomto bodě vývoj ustal z důvodu malého zájmu zákazníků o ruční terminály.

Mě se tedy dostala do ruky deska označená RADKA 2.0 vhodná k osazení procesorem Silabs C8051F236 a radiomodulem AUREL XTR-434 L. Mým prvním úkolem bylo pro tento procesor a tuto desku napsat firmware, který se zaměří na maximalizaci dosahu.

3.2 Rádiové pásmo 433MHz

Pásmo 433 MHz (433,050 ÷ 434,790 MHz) je volně použitelné radiové pásmo pro průmyslové vědecké a lékařské účely. Běžně se používá například pro dálkové ovládání světel v místnostech. Podle Českého telekomunikačního úřadu je provoz takového zařízení povolen s omezeními, z nichž nejdůležitější tři jsou: 1) Vyzářený výkon je menší než 10mW (+10dBm), 2) Klíčovací poměr je menší než 1:10, což znamená, že zařízení smí vysílat maximálně 1/10 času (například 6 minut za hodinu), 3) Povolen je jen přenos dat, zařízení nesmí být použito pro přenos hovoru nebo akustických signálů (bezdrátová sluchátka). Na provoz jsou kladena ještě další omezení uvedená v generální licenci č. GL-30/R/2000 [3].

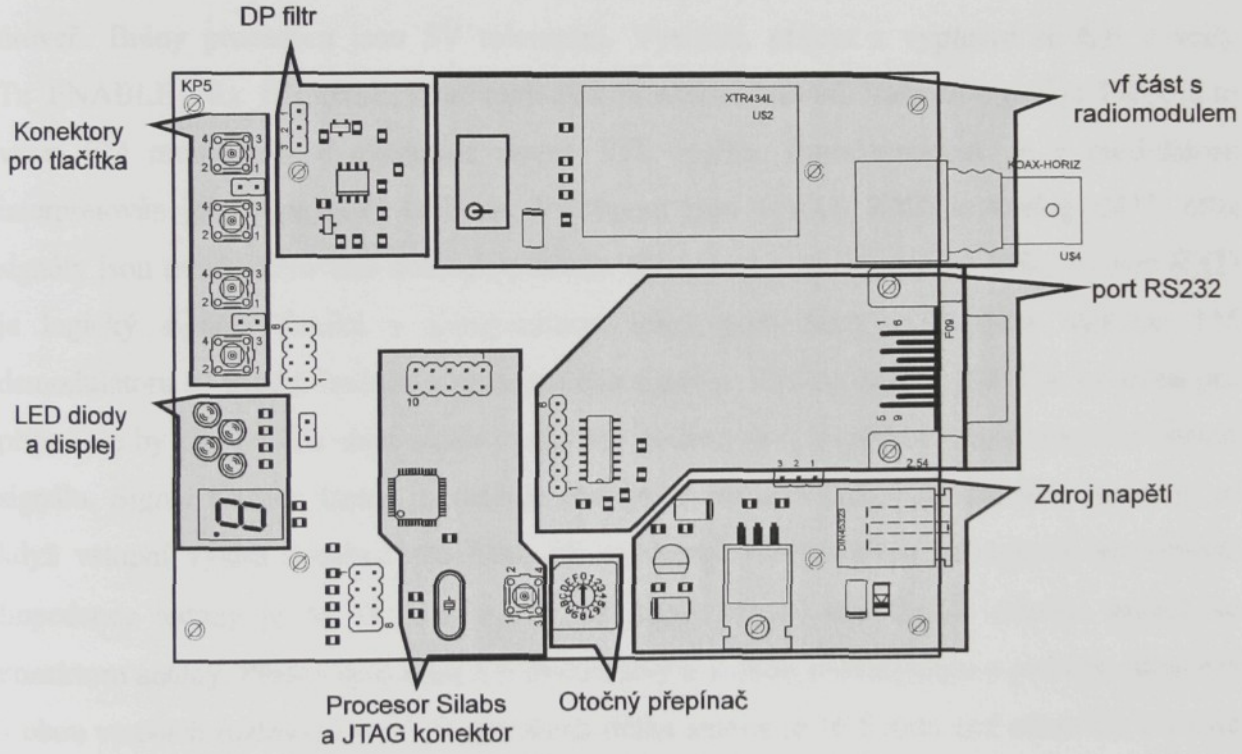
Vývoj zařízení používajících pásmo 433MHz se běžně opírá o radiomoduly zapájitelné do desky vlastního zařízení. Výrobci radiomodulů jsou například italský AUREL nebo britský Radiometrix. Nabízené moduly jsou buď jen vysílač/přijímač pro amplitudovou nebo frekvenční modulaci, nebo radiomodem který nabízí transparentní RS232 v TTL úrovních. Dosah datových radiomodemů je do 200m ve venkovním prostředí. Dosah s jednodušší verzí radiomodulu je dán citlivostí přijímače a pak už je plně v rukou uživatele. Snížením rychlosti lze dosáhnout i vzdáleností nad 400m ve volném prostoru.

Specialitou je výrobek firmy Atmel, která prodává radiomodul a 4-bitový mikroprocesor v jednom pouzdře pro aplikace dálkového ovládání nebo telemetrie.

3.3 Návrh desky plošných spojů

Schéma desky nového rádiového rozhraní navazuje na desku RADKA 2.0 firmy EESA a rozšiřuje její možnosti. Desku, a tím i celé zařízení, jsem nazval RadioSpoj232 v1.0. Celé zařízení není určeno jako konečný produkt, ale jako experimentální deska a vývojový stupeň pro odvození desek do konečných aplikací. Proto deska obsahuje navíc konektory pro všechny vstupně-výstupní brány procesoru, možnost obejití některých obvodů a pozice pro tlačítka, která zatím nejsou softwarem využita. Na procesoru jsem záměrně ponechal nevyužité vývody s funkcí sběrnice SPI.

Rádiové rozhraní jsem se rozhodl pojmout jako transparentní prodloužení portu RS232 pomocí rádia. Nezavádím žádný protokol pro port RS232, je tak možné připojit rozhraní k jakémukoliv zařízení bez nutnosti měnit firmware. Aby bylo možné volit uživatelem z několika cílových stanic, přidal jsem tlačítko pro volbu stanice a sedmisegmentový LED displej pro zobrazení zvolené stanice. Adresa stanice se volí šestnáctipolohovým otočným přepínačem. Port RS232 jsem navrhl s možností hardwarového handshakingu. Desku je možné osadit BNC konektorem pro připojení externí antény. Zařízení je napájeno stejnosměrným napětím v rozmezí 9 ÷ 15 V, například ze síťového napáječe. Desku tištěných spojů jsem přizpůsobil rozměrům krabičky KP5 z nabídky firmy Antonín Atanasovký - A&A. Deska je společná pro stanici master i pro stanice slave. Verze slave se liší v neosazení displejem a tlačítkem, stanice master nepotřebuje otočný přepínač.



obrázek 14: Rozmístění obvodů na desce RadioSpoj232 v1.0

Obvody na desce lze rozdělit do bloků: 1) zdroj napětí +5 V a +3,3 V, 2) procesor Silabs C8051F236 a jeho periferie, 3) vysokofrekvenční část s radiomodulem AUREL XTR-434 L, 4) dolnopropustní filtr na příjmu radiomodulu, 5) port RS232.

3.3.1 Radiomodul AUREL XTR-434 L

Podle výrobce má tento modul vysokou imunitu proti interferencím a velmi krátký čas přepnutí. Modul obsahuje i data slicer, který funguje jako komparátor s dynamicky nastavitelnou rozhodovací úrovní. Proto je vhodné, aby měl přenášený signál střídu 50:50.

Vybrané parametry radiomodulu Aurel XTR-434 L (více viz. katalogový list [4]):

Parametr	**hodnota**	**jednotka**
pracovní frekvence	433,92	MHz
modulační zdvih	± 25	KHz
výkon vf zesilovače	+10	dBm
citlivost vf přijímače	-103	dBm
nf propustné pásmo	2,5 ÷ 25	KHz
čas přepnutí RX ↔ TX	3	ms

Rádiomodul je napájen napětím +5 V, číslicové signály používají napěťové úrovně TTL. Vstupní signály od procesoru posiluji slabými pull-up odpory, abych dosáhl 5V pro vysokou úroveň. Brány procesoru jsou 5V tolerantní. Vysílání, příjem a vypínání se řídí vývody Tx_ENABLE a Rx_ENABLE, které mají aktivní nízkou úroveň. Vstupní signál je TXD, je to vstup FM modulátoru a akceptuje pouze TTL logiku. Pravděpodobně je v modulátoru interpretován jako analogová hodnota. Výstupem jsou vývody RXD a Analog_OUT. Oba signály jsou uvnitř filtrované dolnopropustním filtrem s bodem zlomu 25 KHz. Výstup RXD je logický signál. Vzniká v komparátoru, který porovnává analogovou hodnotu FM demodulátoru se střední hodnotou analogového signálu. Výstup Analog_OUT je vyveden pro případ, že by konstruktér chtěl signál zpracovat podle svého, například rozlišovat více úrovní signálu. Signál Carrier_Detect umožňuje detekovat nosnou rádiového signálu. Aktivuje se když vstupní výkon dosáhne -96 dBm. Při pokusech tento výstup nefungoval spolehlivě. Impedance antény je 50 Ω. Katalogový list dává přesný návod, jak připojit modul ke konektoru antény. Plošný spoj musí být dvouvrstvý a v okolí radiomodulu a pod ním musí být v obou vrstvách rozlévaná měď. Doporučená délka antény je 16,5 mm, což odpovídá čtvrtině vlnové délky.

3.3.2 Procesor Silabs C8051F236

Procesor C8051F236 je jednočipový počítač s jádrem 8051. Jádro má vylepšenou architekturu, která umožňuje narozdíl od klasických procesorů 8051 zpracovat většinu instrukcí v jednom nebo dvou hodinových cyklech, prakticky podle počtu bajtů které instrukce zabírá v paměti. Pro vytvoření programu stačí jakýkoliv překladač pro instrukční soubor MCS-51. Procesor má periferie stejné jako mají procesory 8052 a navíc obsahuje

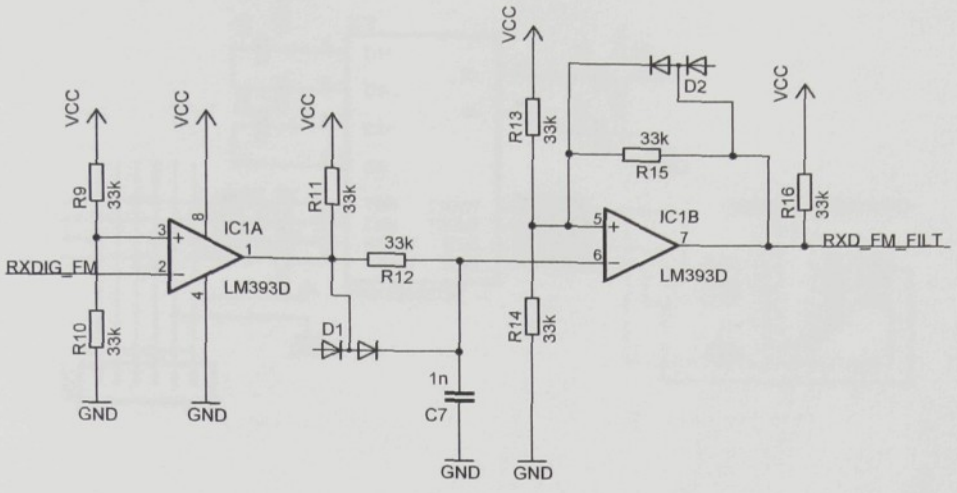
programovatelný obvod WDT, obvod pro sběrnici SPI, dva analogové komparátory, interní zdroj hodinového signálu a monitor napájecího napětí. Paměťově je vybaven 256 bajty RAM, 1024 bajty XRAM a 8kB Flash paměti pro program, která je programovatelná v 512 B blocích i vlastním programem uvnitř procesoru. Programování flash paměti se provádí pomocí rozhraní JTAG, takže je možné měnit firmware přímo v aplikaci, ladit program přímo v aplikaci, včetně krokování, použití breakpointů a přímého prohlížení a změn paměti. Jedinou překážkou v použití může být pouzdro TQFP48 ve kterém je dodáván, protože má velmi malé rozměry, což klade zvýšené technologické nároky při výrobě desek plošných spojů nebo při montáži.

Procesor může být napájen napětím v rozmezí 2.7 ÷ 3.6 V. Vstupně-výstupní brány procesoru fungují volitelně jako push-pull výstup nebo jako otevřený kolektor se slabým pull-up odporem. V případě otevřeného kolektoru jsou brány obousměrné a 5 V tolerantní. Proud procházející jedním vývodem brány je až 20 mA, brána tedy může přímo budit LED diody. Připojením vývodu MONEN na napájecí napětí se aktivuje automatický reset při poklesu napájecího napětí. Procesor umožňuje taktování pomocí externího krystalu do 25 MHz, já používám krystal 11.0592 MHz.

K procesoru je, kromě ostatních zvlášť popisovaných obvodů, připojen osmisegmentový LED displej, tři signalizační LED diody a čtyři tlačítka. Dvě z tlačítek mají paralelně připojen konektor pro externí tlačítko. Tlačítka využívají slabé pull-up odpory integrované ve V/V bráně procesoru a při stisku připojují vodič na zemní potenciál.

3.3.3 Dolnopropustní filtr

Dolnopropustní filtr (na obrázku) je připojen na výstup TXD. První část je komparátor, který porovnává výstupní úroveň signálu s 2,5 V, následuje dolnopropustní RC článek s bodem zlomu přibližně 5 KHz a na výstup RC článku je připojen klopný obvod s hysterezí, který z filtrované hodnoty vytvoří logický signál. Tento filtr odstraní z přijímaného signálu frekvence, které kód kanálu nepoužívá a může tím přispět ke spolehlivosti rozpoznání kódu. Tento obvod je možné vynechat nastavením zkratové propojky. Při pokusech jsem zjistil, že použití tohoto obvodu nemá vliv na celkový dosah. Software totiž provádí podobnou filtraci při rozpoznávání kódu.



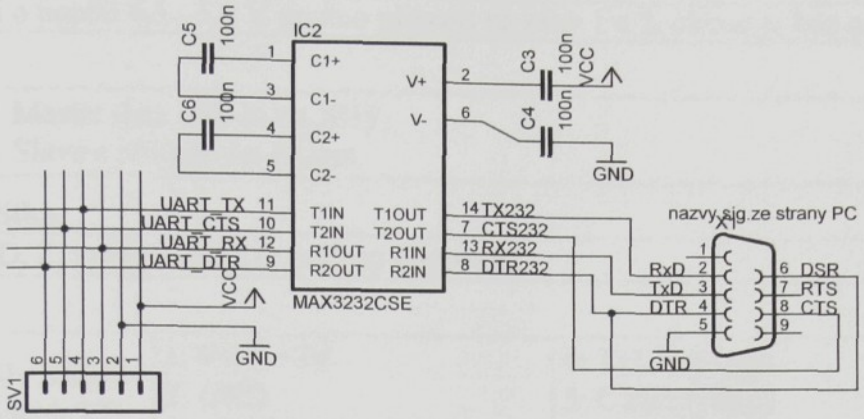
obrázek 15: Schéma dolnopropustního filtru

3.3.4 Port RS232

RadioSpoj232 je z hlediska stran spoje RS232 navrhnut jako DCE - Data Communication Equipment, tedy jako modem. Konektor je 9-pinový D-sub s dutinkami. K zařízením DTE (Data Terminal Equipment - např. počítač) se připojuje pomocí prodlužovacího kabelu. K jiným zařízením DCE se připojuje pomocí kabelu s překříženými signály TxD a RxD.

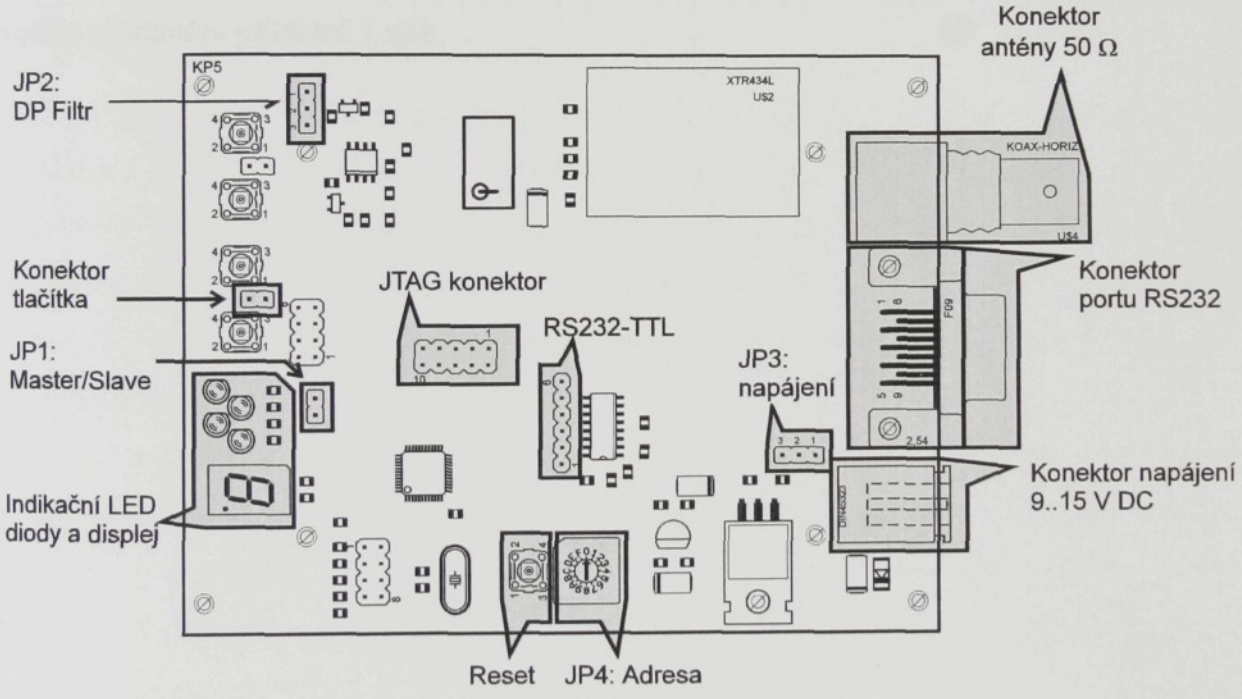
Zapojeny jsou i signály hardwarového handshakingu, který umožňuje řídit spojení a zamezit ztrátě dat při naplnění vyrovnávací paměti. K takovému handshakingu stačí používat signály DTR (terminál je připraven přijímat) a CTS (modem je připraven přijímat). Terminálové programy potřebují pro správnou funkci handshakingu ještě zapojený signál DSR (modem je připraven vysílat). Tento signál stačí propojit se signálem DTR. Signály DTR a DSR jsou propojeny na desce, ale pro uspoření vodičů kabelu je možné tyto signály propojit i na konektoru na druhém konci kabelu. Celkově tak stačí k zařízení přivést 5 vodičů pro plně funkční handshaking a 3 vodiče bez handshakingu. Signál RTS (terminál je připraven vysílat) není zapojen vůbec, protože software nastavuje signál CTS automaticky bez výzvy.

Na desce je navržen navíc konektor se signály RS232 v napěťových úrovních TTL. Může sloužit při vývoji vestavného modulu do ručního terminálu.



obrázek 16: Schéma zapojení portu RS232

3.3.5 Konektory a konfigurační propojky



obrázek 17: Konektory a konfigurační propojky na desce RadioSpoj232v1.0

JP1: Master/Slave

	Otevřený:	Slave
	Zavřený:	Master (bez ohledu na JP4)*

JP2: Dolnopropustní filtr

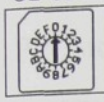
	1-2:	DP filtr zařazen
	2-3:	Bypass

JP3: Napájení

	2-3: Napájení z konektoru (standardně)
	1: GND, 2: +5 V VCC vstup, 3: +5 V výstup stabilizátoru

	Baterii o napětí 4,5 - 5,5 V možno připojit na piny 1 a 2, obvod je bez ochrany!

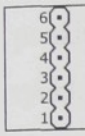
JP4: Adresa

	0: Master (bez ohledu na JP1) 1 - 15: Slave s příslušným číslem

Konektor tlačítka:

	1: GND, 2: vstup tlačítka (s pull-up odporem)

RS232-TTL:

	Signály RS232 v úrovních TTL	1: VCC +5V 2: GND 3: RxD (vstup)	4: TxD (výstup) 5: CTS (výstup) 6: DTR (vstup)

Konektor napájení:

	9 - 15 V, dutý konektor, D = 5,5 mm, d = 2,2 mm

* Pokud je JP1 zavřený (Master) a na JP4 je adresa od 1 do 15, pak bude mít rozhraní roli master a bude se ihned po zapnutí snažit spojit se slave nastaveným na JP4.

Namísto BNC konektoru antény je možno připájet anténu z 16,5 mm dlouhého měděného vodiče o průměru přibližně 1 mm.

4 Software radiového rozhraní

Při koncepci softwaru jsem vycházel z toho, že mám k dispozici dvě desky RADKA 2.0 osazené procesorem Silabs C8051F236 a radiomodulem AUREL XTR-434 L. K taktování procesoru sloužil krystal 11.0592 MHz, přestože procesor může pracovat i na dvojnásobné frekvenci. Po prostudování materiálů jsem usoudil, že nejlepší a téměř jedinou možností je použití kódu Manchester. Požadavky na přenosovou rychlost nebyly vysoké, tak jsem zvolil rychlost 1200 bit za sekundu. Ke zpracování signálu na straně přijímače bylo nutné navzorkovat více vzorků na jeden rozpoznaný bit. Jako bohatě dostačující číslo jsem zvolil převzorkování 32 krát na bit. Provedl jsem pokusy, ve kterých jsem nechal jeden procesor vysílat obdélníkový signál o frekvenci 600 Hz a druhým jsem vzorkovaný signál přenášel do počítače a zaznamenal pro pozdější zpracování. Výsledky byly uspokojivé, obdélníkový signál se přenášel čistě bez poruch i z dvěstěmetrové vzdálenosti. Usoudil jsem tedy, že pro začátek použiji rychlost 1200 bit za sekundu.

Rychlost 1200 bitů za sekundu při převzorkování 32krát vede na vzorkovací frekvenci 38400 vzorků za sekundu. V porovnání s taktováním procesoru vychází na jeden vzorek 288 strojových cyklů, tedy zhruba 150 instrukcí, které může procesor vykonat. Postupem času bylo jasné, že při dekódování kódu Manchester bude potřeba pro rozpoznání přijatého bitu z 32 vzorků čas rozhodně delší nežli vzorkovací interval. Musel jsem proto začít program koncipovat jako systém s multitaskingem.

Představa byla taková, že vzorkování bude probíhat v režimu přerušení od časovače, což zajistí pravidelnost odečtu. Obsluha přerušení bude schopná v krátkém čase odečíst vzorek a uložit jej do FIFO paměti. V programové smyčce, jejíž vykonávání je přerušitelné, poběží program pro dekódování Manchesteru, který bude vybírat vzorky z FIFO paměti vzorků a bude je zpracovávat asynchronně vzhledem k vzorkování. Časově kritické sekce programu budou používat jako paměť registry, protože operace v rámci registrů jsou rychlejší - trvají zpravidla pouze jeden strojový cyklus. Tato koncepce se již zdála reálnější.

Druhým problémem byl způsob komunikace s průtokoměrem. Jednou možností bylo adaptovat se na používaný protokol SIMPLE a začlenit dotaz a odpověď do protokolu, druhou možností bylo vytvořit transparentní rozhraní RS-232 kabel-rádio-kabel. Rozhodl jsem se zobecnit pohled na komunikaci mezi průtokoměrem a sběrným zařízením a vytvořit rozhraní, které se bude navenek chovat jako průchozí RS-232. Připojená zařízení tak budou moct zůstat bez zásahu a komunikace z jejich pohledu bude probíhat stejně jako po kabelu. Jediným omezením oproti kabelu budou fixní parametry komunikace, které jsem v souladu s

manuálem k měřiči tepla MT500 nastavil na 9600 baudů, 8 bit, bez parity. Komunikační rozraní měřičů tepla nepoužívají handshaking. Protože komunikační rychlost měřičů tepla je vyšší nežli datová propustnost mého rádiového rozhraní, bude muset být hlavně přijímací buffer UART portu dostatečně dimenzován vzhledem k velikosti přenášených dat v protokolu SIMPLE.

Zformuloval jsem zadání programu:

- Firmware bude vytvářet transparentní rozhraní RS-232,
- pomocí otočného přepínače bude možné volit z patnácti stanic,
- komunikace bude zajištěná (žádná data by se neměla ztratit) a
- komunikační rychlost bude 1200 bitů za sekundu.

Po vyrobení vlastního radiového rozhraní RadioSpoj232 v1.0 přibyly další požadavky na obsluhu tlačítka a sedmisegmentového displeje, které souvisí s volbou protistrany.

4.1 Program Transceiver 1.0

Plnění zadání vede na použití datových paketů a na vrstvenou architekturu známou z modelu OSI. Program jsem tedy členil do modulů podle vrstev nebo podle tématu. Za programovací jazyk jsem si zvolil Assembler, protože jsem s tímto jazykem měl pro platformu 8051 zkušenosti a také protože vývojové prostředí pro procesory Silabs, které jsem měl k dispozici, umožňuje použít Assembler bez omezení na velikost výsledného kódu (narozdíl od jazyka C). Po zkušenostech nabytých psaním tohoto komunikačního software bych volil raději kombinaci Assembleru a C, Assembler pro obsluhu přerušení, fyzickou vrstvu a jiné časově kritické sekce a jazyk C pro spojovou vrstvu, síťovou vrstvu a jiné procedury, které nejsou spouštěny tak často a obsahují složité struktury podmínek, které se v jazyce Assembler stávají nepřehlednými.

4.2 Úvaha nad signály, periferiemi a výkonem procesoru.

Procesor Silabs C8051F236 obsahuje tři programovatelné šestnáctibitové časovače/čítače. Časovače mohou být použity jako baudrate generátory pro UART port, jako programovatelné časovače s, nebo bez automatického přednastavení, jako čítače externích událostí, nebo jako časovače, jejichž běh je závislý na úrovní externího signálu (gate režim). [9]

Vstupní signál z radiomodulu je podle katalogového listu radiomodulu frekvenčně omezen dolnopropustním filtrem s bodem zlomu na 25kHz. Aby nedocházelo k aliasingu, musím zvolit vzorkovací frekvenci několikrát vyšší, protože úrovně šumu dosáhne výstup filtru rozhodně později než na bodu zlomu a vzorkování musí respektovat Shannonův vzorkovací

teorém $f_s \geq 2f_{max}$. To vede na vzorkovací frekvence ve stovkách kilohertz, které se již blíží taktování procesoru. Řešením je použít jeden hardwarový čítač v gate režimu, který bude odečítat a nulovat vzorkovací rutina. Hodnota vzorku se zjistí porovnáním hodnoty v čítači s polovinou dosažitelného maxima. Hodnota bude reprezentovat střední hodnotu signálu v době mezi odečtem vzorků. Procesor je taktovaný na 11.0592 MHz, stejně tak lze taktovat časovače. Lze předpokládat, že při vzorkování na takto vysoké frekvenci již nebude vznikat aliasing.

Časovač Timer 0 používám v gate režimu pro odečet vstupního signálu. Časovač Timer 1 používám jako taktování portu UART. Časovač Timer 2 lze nastavit na automatické přednastavení (auto-reload) a používám jej k přesnému časování vzorků pro signály radiomodulu.

4.3 Vrstvy a komunikace mezi nimi

Program jsem rozdělil do vrstev Vzorkovač, Manchester, Rámec a Paket. Z pohledu modelu OSI jde pouze o dvě nejnižší vrstvy: fyzickou a spojovou. Spojová vrstva zde přebírá částečně i roli transportní vrstvy, protože obsahuje algoritmy pro opakování vadných paketů a tím vytváří spolehlivý kanál.

Odpovídající vrstva OSI	Vrstva programu	Popis
Fyzická	Vzorkovač	Čtení/zápis vzorků, ovládání radiomodulu
	Manchester	Kodér/dekodér, synchronizace bitů
Spojová MAC	Rámec	Synchronizace znaků a rámců, detekce chyb
Spojová LLC[1]	Paket	Řízení master/slave, adresace, řízení spojení, zotavení z chyb

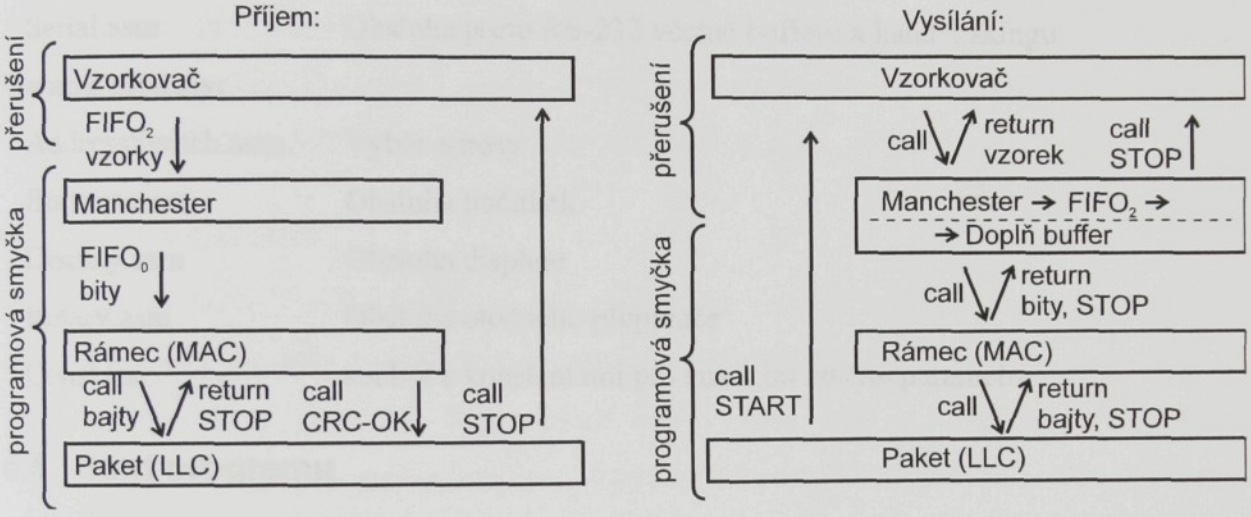
Tyto vrstvy musí mít mezi sebou definována komunikační rozhraní. Dalo by se říci, že čím nižší vrstva, tím více dat musí zpracovat. Při příjmu vzniká na straně vzorkovače nejvíce dat, které dekodér Manchesteru zredukuje dvaatřicetkrát. Dále vrstva rámce najde v nepřetržitém toku bitů rámec, oddělí data od preambule a zabezpečení a ty dále předá vrstvě paketu. Vrstva paketu oddělí hlavičku od zbytku zprávy, zkontroluje adresy, z dalších bitů zjistí zdali může posílat nový paket nebo musí opakovat předešlý a nesenou zprávu předá objektu spravujícímu sériový port RS-232.

Vzorkovač předává data pomocí univerzální FIFO fronty dekodéru manchesteru. Manchester předává bitový tok vrstvě rámce pomocí jiné FIFO fronty. Vrstva rámce již s vrstvou paketu

[1] MAC - Medium Acces Control, LLC - Logical Link Control

komunikuje synchronně pomocí volání funkcí softwarového rozhraní. Dvě FIFO fronty používám proto, aby se minimalizovalo množství dat, které je nutné uchovat, než se zpracují časově náročné funkce ve vrstvě rámce a ve vrstvě paketu. Data se předávají přímo - volaná procedura musí data přijmout.

Na straně odesílání si vrstva paketu odebírá data z objektu portu RS-232 dle potřeby a vytváří v paměti paket včetně hlaviček. Paket zavolá funkci vzorkovače, která zapne vysílání. Vzorkovač se dotáže kodéru Manchesteru na konkrétní vzorek. Kodér Manchesteru obsahuje malou paměť, která se doplňuje v programovém cyklu: Doplňování paměti Manchesteru zavolá funkci rámce, ten podle potřeby volá funkci paketu. Data se tedy předávají na dotaz - volaná procedura musí data poskytnout.



obrázek 18: Schéma komunikace mezi vrstvami programu

4.4 Moduly programu

Program jsem rozčlenil do modulů:

Start programu:

TransceiverMain.asm	Boot-up a reset ostatních modulů
Config.asm	Konfigurace periferií

Operační systém:

tasker.asm	Systémová fronta pro přerušitelný režim, nekonečný cyklus
Timer2Interface.asm	Systémová fronta pro režim obsluhy přerušení
sysTimer.asm	Časovač událostí

Fyzická vrstva:

Timer2.asm	Vzorkovač a obsluha radiomodulu, zdroj hodin OS
ManchesterEnc.asm	Kodér Manchesteru

ManchesterDec.asm	Dekodér Manchesteru
Spojová vrstva MAC:	
FrameEncode.asm	Generuje rámec
FrameDecode.asm	Detektor preambule a zpracování přijatého rámce
CRC16.asm	Rutina generující CRC16, používaná moduly rámce
Spojová vrstva LLC:	
PacketStatus.asm	Řízení master/slave, řízení spojení, adresace
PacketEncode.asm	Vytváření paketu pro odvysílání
PacketDecode.asm	Příjem paketu
Sériový port:	
Serial.asm	Obsluha portu RS-232 včetně bufferu a handshakingu
Ostatní moduly:	
AddressSwitch.asm	Výběr adresy
Buttons.asm	Obsluha tlačítkek
Display.asm	Obsluha displeje
Rotary.asm	Obsluha otočného přepínače
Const.inc	soubor s konstantami pro snadnou změnu parametrů

4.5 Start programu

Kód modulu TransceiverMain.asm je alokován staticky na adresu 0000h, tj. adresu, kde procesor začíná vykonávat program po vyresetování. Definuje odskoky na obsluhu přerušení časovače Timer 2 a portu UART. Spouští modul Config.asm, který provede nastavení všech periferií procesoru a přepne hodinový zdroj z vnitřního na vnější krystalový oscilátor. Poté spustí inicializační rutiny ostatních modulů, které nejsou inicializovány jinak za běhu. Nakonec nastaví globální povolení všech přerušení a provede odskok na nekonečnou smyčku operačního systému v modulu Tasker.asm.

4.6 Operační Systém

Operační systém je jádrem programu. Pomocí něj si některé podprogramy předávají data a umožňuje plánovat opakované spouštění některých podprogramů. Základním modulem je Tasker.asm, který obsahuje nekonečnou smyčku. Nekonečná smyčka běží v přerušitelném režimu a cyklicky vybírá data ze systémových front a spouští příslušné podprogramy. Přednost pro vykonávání má fronta pro režim přerušení (dále $FIFO_2$), protože v ní je největší datový tok a mohla by se snadno přeplnit. Fronta pro přerušitelný režim (dále $FIFO_0$) se

vykonává až když je fronta $FIFO_2$ prázdná. Do fronty $FIFO_0$ mohou zapisovat podprogramy běžící v přerušitelném režimu, fronta $FIFO_2$ je určena pro podprogramy běžící v režimu obsluhy přerušení s vysokou prioritou.

Plánování úloh zajišťuje modul sysTimer.asm, který se spouští pravidelně. Frekvence spouštění je odvozena od vzorkovací frekvence rádia a je to jednou za 256 vzorků. Tato frekvence se dále dělí podle potřeby. Takto je spouštěna například obsluha tlačítek.

Úkolem operačního systému je i ošetření chyb. Chyba v programu nastane, když přeteče některá z front. Chybu přetečení fronty nelze nikterak opravit než resetem. Příslušný podprogram, který tuto chybu zjistí zavolá funkci OS, která zastaví běh programu, rozsvítí příslušnou LED diodu a číslo chyby zobrazí na displeji. Současně se číslo chyby uloží do paměti, která se resetem nevymaže a je možné si ji přečíst pomocí JTAG rozhraní ve vývojovém prostředí. Po krátké chvíli se program vyresetuje. Tento mechanizmus jsem zavedl hlavně kvůli ladění programu, při normálním běhu by se vůbec neměl spustit.

4.7 Fyzická vrstva

Co se týče hardwaru, fyzická vrstva používá kanál rádia, který akceptuje dvouúrovňový signál a jeho výstupem je taktéž dvouúrovňový signál. Signál musí být v přeloženém pásmu a střída by se měla blížit poměru 1:1. Rádio může buď vysílat, nebo přijímat (nebo být úplně vypntuté), je tedy možné zřídit pouze poloduplexní provoz. Rádio pracuje pouze v jednom kanálu - není přeladitelné. Použitá modulace je FM, základní kmitočet je 433.92 MHz. Rádio není schopné detekovat nosný signál, proto je v době, kdy protistrana nevysílá, na výstupu přijímače šum.

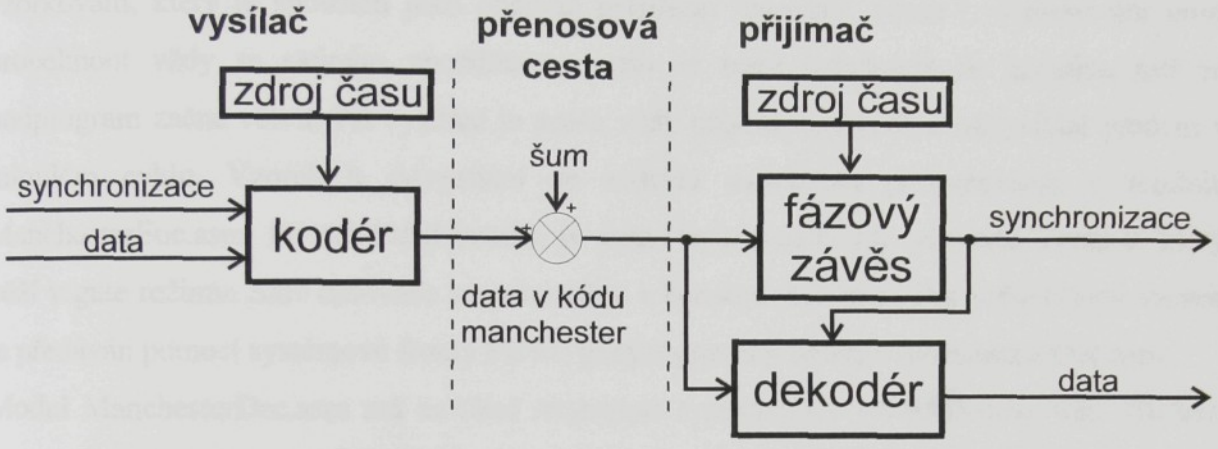
Z pohledu softwaru má fyzická vrstva přeměnit tento hardwarový kanál na tok bitů s volitelným směrem. Rozdělil jsem tento problém na část, která obaluje hardware, a na část, která se zabývá generováním a rozpoznáváním přenášeného signálu.

Použité kódování je kód Manchester, někdy označovaný jako PSK (Phase Shift Keying). Počet vzorků na bit je 32, největší synchronizační krok je o jeden vzorek, spolehlivé synchronizace dosáhne dekodér po šestnácti krocích.

Použitá rychlost je 2500 bit/s. Vzhledem k deklarovanému času přepnutí 3 ms vychází doba přepnutí na 7,5 bitu. Při startu vysílání přidává vysílač navíc 8 bitů, na které navazuje synchronizační část preambule. Doba přepnutí je tímto překlenuta.

4.7.1 Dekódování kódu manchester

Algoritmus jsem odvodil z funkce obvodů DPLL (Digital Phase Lock Loop - digitální fázový závěs, viz. [1]). Kodér kombinuje vstupní data a hodinový signál do přenášeného signálu. Dekodér má vlastní zdroj hodinového signálu, který je řízen podle příchozího signálu. Pokud je hodinový signál dekodéru synchronizovaný s příchozím signálem, je možné z příchozího signálu správně rozpoznat nesená data.



obrázek 19: Schéma přenosu dat v kódu manchester

Algoritmický postup dekódování manchesteru ze vzorkovaného signálu včetně synchronizace:

Algoritmus používá buffer: {"předchozí vzorek", "okénko", "následující vzorek"}

1) Načtou se vzorky do okénka (např. 32 vzorků okénka) a 2 okolní vzorky
2) Vyhodnotí se, zdali se vzorky v okénku podobají více hodnotě log1 nebo log0 a hodnota se zaznamená nebo odešle další vrstvě zpracování.
3) Vyhodnotí se, zdali by se zjištěná hodnota nepodobala více při posunu okénka vpravo(do budoucna, hodiny přijímače byly opožděny za hodinami vysílače), nebo vlevo(do minulosti, hodiny přijímače byly zrychleny před hodinami vysílače).
4) Zapíše se příslušný počet vzorků z konce tohoto bufferu na začátek bufferu (0 až 2 vzorky).
5) Přijme se zbývající počet bitů do bufferu. Cyklus se opakuje od bodu 2).

Za předpokladu kvalitního signálu a malého rozdílu frekvence hodin vysílače a přijímače konverguje okénko k ustálené pozici, kdy hodinová hrana je přesně uprostřed. Počet bitů v okénku by měl být sudý, aby bylo možné okénko rozdělit na dvě stejné poloviny.

Minimální převzorkování (počet bitů v okénku) je 4. Při převzorkování 2 by bylo možné dekódovat signál, ale nezbyl by prostor pro synchronizační posuv. Z praktického hlediska je

ale lepší volit převzorkování větší, protože je potom možné opravovat drobné chyby v signálu. Takovéto opravy se provádějí na bázi opakovacího kódu. [2]

4.7.2 SW moduly fyzické vrstvy

Modul Timer2.asm obsluhuje radiomodul a generuje systémový čas. Modul obsahuje podprogram pro ovládání radiomodulu (vysílání, příjem, vypnuto) a podprogram pro vzorkování, který je spouštěn jako obsluha přerušení časovače Timer 2. Vzorkování musí proběhnout vždy se stejným zpožděním, proto se musí vzorkovat na začátku, než se podprogram začne větvit. Při vysílání je proto vždy připraven vzorek k odvysílání předem v minulém cyklu. Vzorek k odvysílání se získává zavoláním podprogramu v modulu ManchesterEnc.asm. Při odečítání vzorku je nutné vyhodnotit stav časovače Timer 0, který běží v gate režimu. Stav časovače se porovnává s konstantou a výsledný jednobitový vzorek je předáván pomocí systémové fronty $FIFO_2$ podprogramu v modulu ManchesterDec.asm.

Modul ManchesterDec.asm má za úkol rozpoznat z příchozích vzorků bitový tok. Příchozí vzorky nahrává do paměti a když jich je dostatek, rozpozná daný bit výše popsaným algoritmem, včetně bitové synchronizace. Při výpočtu hodnoty bitu používá kombinaci look-up tabulky a součtu. Rozpoznaný bit je předáván pomocí systémové fronty $FIFO_0$ modulu FameDecode.asm ve spojové vrstvě.

Modul ManchesterEnc.asm má za úkol generovat vzorky v kódu Manchester. Podprogram pro výstup (volaný z modulu Timer2.asm) generuje vzorky a v případě nedostatku dat vyšle požadavek na doplnění pomocí systémové fronty $FIFO_2$ podprogramu pro vstup. Data se mezi podprogramy předávají ve sdílené paměti. Podprogram pro vstup při získávání dat přímo volá podprogram v modulu FrameEncode ve spojové vrstvě. Pokud již nejsou k dispozici žádná data k odvysílání, zavolá se rutina ovládání radiomodulu v modulu Timer2.asm a ta přepne radiomodul na příjem.

4.8 Spojová vrstva

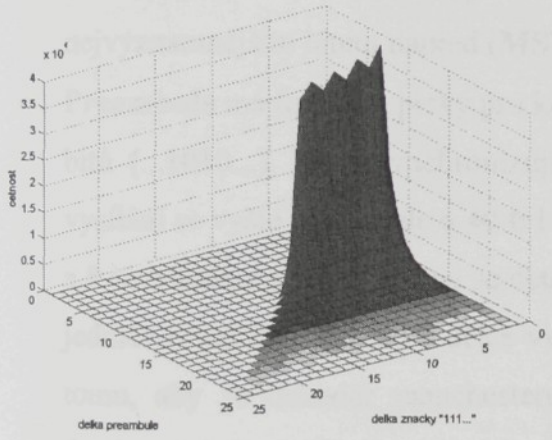
Spojová vrstva má za úkol použít bitový tok, nabízený fyzickou vrstvou, k předávání dat získaných od vyšších vrstev, rozdělených po částech do balíčků. Ve spojové vrstvě se balíčky nazývají rámce. Rámec se skládá z preambule, hlavičkových dat, přenášených dat a kontrolního součtu. Spojová vrstva není úplně nezávislá vzhledem k ostatním vrstvám. Důležitá je volba preambule vzhledem k vlastnostem fyzické vrstvy. Spojovou vrstvu je možné rozdělit na podvrstvy MAC (Medium Acces Control) a LLC (Logical Link Control). Podvrstva MAC definuje rámec a detekuje chyby, podvrstva LLC má na starost filtraci rámců

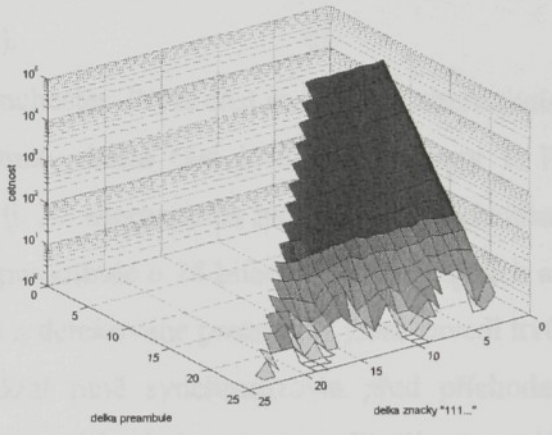
podle adresy, zotavení z chyb (nedoručení rámce) a řízení časového multiplexu. Vzhledem k charakteru aplikace jsem se rozhodl používat řízení master/slave. Master na straně sběrného zařízení, slave na straně měřičů tepla.

4.8.1 Volba vhodné preambule

Preambule má za úkol označit začátek rámce a umožnit synchronizovat hodiny dekodéru Manchesteru. Vzhledem k přítomnosti šumu v době, kdy žádná stanice nevysílá, se musí volit preambule taková, která má pokud možno při svojí délce co nejmenší pravděpodobnost výskytu.

Následující obrázky vznikly analýzou šumu - výstupu dekodéru manchesteru, do kterého vstupoval signál z radiomodulu AUREL XTR-434L. Délka souboru byla $12{,}5.10^6$ bitových vzorků, což reprezentuje 83 minut příjmu.



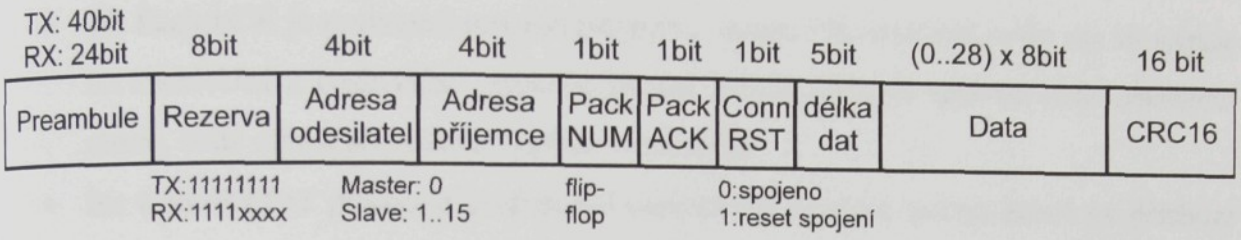


obrázek 20: výskyt preambulí různé délky a formátu **obrázek 21: výskyt preambulí v log. souřadnicích**

Poměrné zastoupení vzorků s hodnotou 0 resp. 1 bylo 54% resp. 46%. Protože mírně převažovaly nuly, je vhodné volit preambuli s převažujícími jedničkami. Sledoval jsem tedy četnost výskytu sekvencí ve formátu "..101010 1111..". Testy četnosti jsem prováděl od celkové délky 8. Z výsledku je patrné, že četnost klesá exponenciálně s délkou sekvence. Vlivem převažujících jedniček v souboru klesá četnost mírně i ve směru zvětšování startovací značky na úkor délky synchronizační sekvence. Dalším jevem, který není příliš dobře patrný z grafu, je mírně vyšší výskyt sekvencí s velmi krátkou částí "11" nebo naopak kde tato část zabírá celou preambuli . Pravděpodobně tedy příliš nezáleží na poměru počtu alternujících a konstantních bitů, bude záležet jen na délce preambule. Zvolil jsem tedy délku preambule 24 bitů, protože se četnost výskytu pohybuje v jednotkách. Počty alternujících a konstantních bitů jsem zvolil jako 16 a 8.

4.8.2 Formát rámce

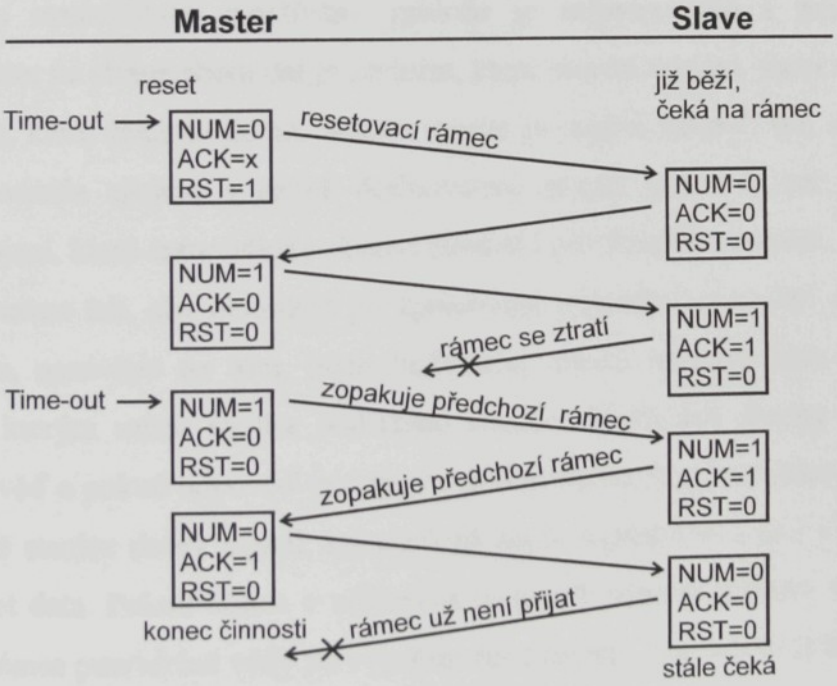
Při úvahách nad definicí vhodného rámce je nutné optimalizovat požadované schopnosti oproti velikosti přídavných dat, která tyto schopnosti zajistí. Požadované vlastnosti jsou: Spolehlivé zabezpečení proti chybám, 16 adres, možnost kompatibility s budoucími verzemi protokolu, rychlá odezva, přenos osmibitových znaků a velikost užitečných dat řádově desítky bajtů v jednom rámci.



obrázek 22: Formát rámce

- Všechna data nesená v rámci jsou seskupena po osmi bitech. Čísla i data jsou řazena nejvýznamnějším bitem napřed (**MSB** first).
- **Preambule** má formát typický pro kód manchester. První část se skládá z alternujících bitů (...1010...), které synchronizují přijímač, druhá část je 8 bitů hodnoty 1. Při vysílání se vyšle preambule o 40-ti bitech, tj. 32 alternujících bitů počínaje hodnotou 1 a 8 bitů hodnoty 1. Při příjmu se detekuje preambule o 24 bitech - 16 alternujících a 8 jedničkových bitů. Rozdíl v délce vysílané a detekované preambule jsem zavedl kvůli tomu, aby se dekodér manchesteru dokázal plně synchronizovat před příchodem detekované části. V případě zvyšování rychlosti by bylo nutné prodloužit alternující část preambule tak, aby byl plně překlenut čas přepnutí radiomodulu. Pro detekci jsem vybral délku 24 bitů, protože z hlediska zpracování je výhodné mít délku dělitelnou osmi a protože u 24 bitů již je pravděpodobnost náhodného výskytu v šumu malá. Pravděpodobnost závadné interpretace rámce dále snižuje povinný test čtyř bitů v rezervovaném slově, test adres (8 bit) a CRC (16 bit). Celkově musí být přesně definováno 52 bitů, aby byl rámec vyhodnocen za platný. Po preambuli následují osmibitové znaky vysílané nejvyšším platným bitem napřed (MSB first).
  Hlavičková data jsem kvůli snazšímu zpracování rozdělil po osmi bitech stejně jako přenášené znaky.
- **Rezervní slovo** jsem použil kvůli možnosti kompatibility s budoucími protokoly. Při vysílání se definuje jako 8 x "1", při příjmu se testuje pouze první polovina. Povinnou část lze označit za skupinu kompatibilních protokolů, nepovinnou část lze označit za verzi protokolu uvnitř skupiny.

- **Adresy** jsou 4bitové. Master má vždy adresu 0, zbytek adresního prostoru je určen pro podřízené stanice. Komunikace probíhá vždy mezi stanicí master a podřízenou stanicí. Vysílání více/všem stanicím není podporováno.
- Bit **PackNUM** je číslo odchozího rámce. Každý nový rámec má opačnou hodnotu tohoto bitu. Při příjmu je každý další rámec se stejným číslem vymazán, dokud nepřijde rámec s opačným číslem.
- Bit **PackACK** je potvrzení správně přijatého rámce. Při odesílání se do něj zkopíruje bit PackNUM z posledního přijatého rámce. Pokud nedojde správné číslo potvrzení rámce, bude rámec při příštím vysílání zopakován.
- Bit **ConnRST**=1 přikazuje protistraně vyresetovat spojení, nebrat zřetel na příchozí bit PackNUM, nastavit vlastní bit PackNUM na počáteční hodnotu a vymazat data uložená ve frontách portu RS-232 i ve strukturách rámce. Stanice která vyšle ConnRST = 1 má za povinnost také vymazat veškerá data. Vytvoří se tím spojení bez ohledu na minulost. Díky této vlastnosti je možné automaticky obnovit spojení po vyresetování jedné ze stanic.



obrázek 23: Potvrzování rámců

- Pole **délka dat** nese informaci o délce užitečných dat v rámci v jednotkách (osmibitových) bajtů. Nabývá hodnot $0 \div 28$. Délka dat je volena záměrně tak, aby celková délka rámce bez preambule a zabezpečení byla kratší než 255 bitů. Pokud je tomu tak, lze očekávat lepší míru zabezpečení od kontrolního součtu CRC.

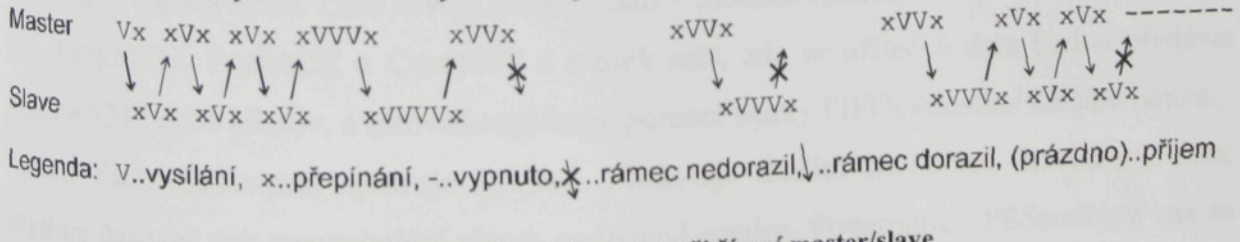
- **Data** jsou přítomna v délce, kterou definuje pole "délka dat." Oproti pevné velikosti rámce to umožňuje zvýšit přenosovou rychlost při přenosu převážně v jednom směru. Rovněž se tím zkracuje odezva v případě krátkých dotazů.
- V poli **CRC16** se přenáší kontrolní kód generovaný polynomem CRC-CCITT uvedeným výše. Vysílá se stav registru po vstupu posledního bitu zprávy. Pokud se na straně přijímače pole CRC16 neshoduje s vypočtenou hodnotou, je obsah celého rámce vymazán bez dalšího zpracování.

V nejdelším rámci je přenášeno 224 bitů užitečných dat a 80 bitů režijních dat. Nejmenší poměr režie je 26 %. Pokud se k tomu přičte čas přepnutí (8 bit navíc), zabírá režie nejméně 28 % času. V případě přenosu objemných dat v jednom směru se přenáší 224 užitečných a 176 režijních bitů, režie tedy činí 44 %. S bitovou rychlostí 2500 bit/s je propustnost 175 znaků/s. Nejkratší čas odezvy v případě jednobajtového dotazu a jednobajtové odezvy je 80 ms.

4.8.3 Řízení Master/Slave

Řízení přístupu master/slave používám, protože je nejjednodušší a pro moji aplikaci vyhovující. Master na straně sběru dat je zařízení, které otevírá spojení, slave na straně měřiče tepla je zařízení, které čeká na dotaz. Stanici master je možné nastavit tak, aby automaticky po spuštění navázala spojení s pevně definovanou stanicí slave. Je tak možné postavit dlouhodobé spojení, které automaticky obnoví činnost i po výpadku napájení.

Program je nastaven tak, aby okamžitě po zpracování přijatého rámce byl odeslán rámec v opačném směru, nezávisle na tom, jestli bude nést nějaká užitečná data. Toto je jediný mechanizmus, kterým může vysílat podřízená stanice. Nadřízená stanice navíc obsahuje čekání na odpověď a pokud odpověď nedorazí v definovaném čase, zopakuje předešlý rámec. Pokud mají obě stanice dobrý signál, komunikují spolu nepřetržitě a to i v případě že není potřeba přenášet data. Pokud rámec v některém ze směrů nemůže dorazit, vysílá nadřazená stanice tentýž rámec pravidelně vždy po vypršení času čekání. Čas čekání je nastaven tak, aby si obě stanice mohly mezi sebou vyměnit nejdelší možné rámce.



obrázek 24: Přidělování kanálu při řízení master/slave

4.8.4 SW moduly spojové vrstvy

Spojovou vrstvu jsem rozdělil na dvě podvrstvy. Nižší podvrstva pracuje s rámcem v čase. Jsou to moduly FrameEncode.asm a FrameDecode.asm. Jde o detekci preambule (rámcová synchronizace) a o znakovou synchronizaci, v případě vysílání jde o generování rámce. Tato nižší vrstva je stále velmi spjatá s fyzickou vrstvou, proto obsahuje i kontrolní kód CRC16, který má snížit riziko interpretace vadného rámce na přijatelnou mez.

Při příjmu dostává modul FrameDecode.asm bitový tok od modulu ManchesterDec.asm. Tento bitový tok prochází posuvným registrem o délce 24 bit. Pokud se obsah registru shoduje se vzorem detekované části preambule, přepne se modul do režimu záznamu. Po každém přijaté osmici bitů předává tento bajt vyšší vrstvě v modulu PacketDecode.asm, který jej uchová v paměti. V návratové hodnotě vrací PacketDecode.asm, zda ještě bude přijímat další bajty, či nikoliv. Rozhoduje se mimo jiné i podle obsahu bajtu s délkou dat. Když přestane PacketDecode.asm přebírat data, přepne se modul FrameDecode do režimu porovnávání kontrolní sekvence. Výsledek porovnání je předán opět modulu PacketDecode, který uložená data vymaže nebo pokračuje ve vyhodnocení. Tento mechanizmus přímého předávání jsem zavedl z důvodu úspory operační paměti.

Při vysílání je modul FrameEncode.asm dotazován modulem ManchesterEnc.asm na výstupní data. Podle vnitřních stavů nejdříve předává preambuli, poté předává data, na která se dotázal modulu PacketEncode.asm a nakonec odešle kontrolní sekvenci, kterou si vygeneroval v průběhu předávání.

Modul CRC16.asm je používán oběma moduly FrameEncode a FrameDecode. Uchovává obsah registru CRC. Protože vysílání a příjem probíhají vždy odděleně, nemůže dojít ke kolizím v používání tohoto modulu.

Vyšší vrstva se zabývá informačním obsahem přenášeného rámce, data která dostane považuje za bezchybně přijatá. Má na starost adresaci, automatické opakování rámce a řízení master/slave.

Modul PacketDecode.asm po naplnění paměti výše zmíněným mechanizmem testuje obsah rezervního bajtu a adres. Dále zavolá podprogram v modulu PacketStatus.asm, který zpracuje bity PackNUM, PackACK a ConnRST a z nich určí, zda se užitečná data budou předávat portu RS232 nebo nikoliv, a zároveň naplánuje pomocí fronty $FIFO_0$ odeslání dalšího rámce.

Modul PacketStatus.asm spravuje spojení. Poskytuje dalším modulům hlavičková data, ošetřuje časovač pro znovuvyslání rámce nadřízené stanice. Podprogram PSSendNow má za

úkol připravit odeslání rámce. Zjišťuje, zda-li se bude odesílat nový rámec nebo se bude opakovat starý. Spouští další podprogramy tak, aby mohlo být zahájeno vysílání.
Modul PacketEncode.asm vytváří obsah nového rámce a poskytuje data modulu FrameEncode.asm.

4.9 Sériový port RS232 (UART)

Modul Serial.asm kompletně obaluje sériový port, který procesor nabízí jako periferii. Umožňuje používat hardwarový handshaking (řízení toku) v obou směrech a automaticky detekuje, zda je handshaking používán. Obsahuje vstupní i výstupní buffer, každý o délce 127 bajtů. Se zbytkem programu komunikuje pouze pomocí podprogramů UARTGet a UARTSend. Modul ovládá sériový port v obsluze přerušení, nastaveného na nízkou prioritu. To umožňuje velmi efektivní funkci. Při používání handshakingu je navíc potřeba pravidelně spouštět podprogram DTRResume, který obnoví spojení v případě zastavení přenosu ze strany DCT. Použité parametry spojení jsou 9600 baud, 8bit, bez parity, 1 stop bit.
Hardware sériového portu je spřažen s časovačem Timer 1. Pokud obvod UART přijme, nebo dokončí odeslání bajtu, spustí se obsluha přerušení sériového portu, která ošetří událost - zaznamená přijatý bajt do paměti nebo pošle další bajt. Hardwarový handshaking již není součástí periferií procesoru a realizuji ho softwarem pomocí dvou bitů vstupně-výstupní brány. Používám pouze linky CTS(Clear To Send - toto zařízení (DCE) může přijímat data) a DTR (Data Terminal Ready - druhé zařízení (DTE) může přijímat data). Linka DSR je v hardwaru desky spojena se signálem DTR a signál RTS je nepotřebný a ignorován. V případě odpojeného kabelu má linka DTR stejnou hodnotu jako OFF - příjem zakázán. Autodetekce probíhá tak, že dokud je DTR ve stavu OFF, předpokládá se, že handshaking není používán. Jakmile je jednou zjištěn stav ON, handshaking začne fungovat. Signál CTS je vysílán vždy.

4.10 Ostatní moduly

Modul Buttons.asm vyhodnocuje stav tlačítek a definuje události. Vyhodnocení tlačítek a obsluha událostí jsou spouštěna pravidelně modulem sysTimer.asm. Stav tlačítek je vzorkován každých 50 ms a zákmity tlačítek jsou filtrovány algoritmem dvou stejných vzorků. Odezva na stisknutí tlačítka je do 100 ms, před dalším stiskem tlačítka musí být tlačítko 100 ms uvolněné.
Modul Display.asm ovládá displej a umožňuje zobrazit na displeji znaky 0-9, A-F.
Modul Rotary vyhodnocuje číslo na otočném šestnáctipolohovém přepínači.

Modul AddressSwitch umožňuje měnit adresu protistrany za běhu jako akci po stisknutí tlačítka. Po změně adresy vymaže všechna zbylá data z mezipaměti.

Modul Const.inc je soubor s konstantami, které se uplatní při překladu programu. Definuje přenosové rychlosti, pojmenování portů a frekvenci použitého krystalu.

4.11 Možnosti pokračování vývoje

Program Transceiver 1.0 je v současné podobě plně funkční firmware pro desky RadioSpoj232 v1.0. Umožňuje vytvořit spojení mezi dvěma stanicemi rychlostí 2500 bit/s. Spojení je spolehlivé (žádná data se neztratí) a vytváří datovou rouru pro port RS232, takže připojená zařízení nemusí měnit své protokoly. Pro komunikaci mezi ručním terminálem a měřičem tepla jsou parametry spojení vyhovující. Přesto existuje několik možností, jakými by vývoj programu mohl pokračovat.

Předně by bylo možné zvyšovat rychlosti. Použitím krystalu 22.1184 MHz a snížením převzorkování by rychlost mohla dosáhnout až 25 kbit/s, což je teoretické maximum pro radiomodul XTR-434 L s kódem Manchester. Rychlost by mohla být volena i automaticky podle ztrátovosti rámců. Ještě vyšších rychlostí by bylo možné dosáhnout s kódem, který vkládá za každé dva informační bity jeden vyvažovací bit. To ale porušuje podmínku nulové střední hodnoty signálu a vede to ke snížení odolnosti proti interferencím a šumu.

Při zavedení větších rámců by se snížil podíl režie a tím zvýšila celková rychlost přenosu objemnějších dat. Zároveň by se ale zvýšila i chybovost rámců a proto by bylo vhodné řídit zároveň bitovou rychlost i délku rámců. Dalším krokem by mohlo být zavedení Hammingova kódu pro opravy chyb. To ale nezvýší dosah, jen sníží chybovost rámců a celkovou rychlost v důsledku přidávání redundantní informace. Hammingův kód by se používal jen za nejhorších podmínek a na nejnižší bitové rychlosti.

Program by bylo možné doplnit o protokol AT příkazů kompatibilního s telefonními modemy, což by umožnilo snadnější integraci do průmyslových aplikací.

Pokud by nepostačoval pouze dvoubodový spoj, bylo by možné za pomoci rádiových rozhraní vytvořit síť s centrálním uzlem. V případě nahrazení přístupové metody master/slave metodou CSMA by bylo možné spojit libovolná dvě rozhraní v rámci dosahu, nebo vytvořit síť s trasováním a dokonce i síť ad-hoc, která automaticky mění trasy v závislosti na dostupnosti stanic.

Na straně zabezpečení dat by bylo možné používat delší CRC kód. V úvahu by připadalo i šifrování přenášených dat, ale v podmínkách průmyslové komunikace je to zbytečné.

Většina uvedených vylepšení by se bezpodmínečně promítla do změny protokolu. Přesto mohou budoucí verze firmware koexistovat zároveň s nynější verzí díky zavedení rezervovaného slova v současném protokolu.

5 Radiové rozhraní RadioSpoj232

Rádiové rozhraní RadioSpoj232 slouží k vytvoření dočasného dvoubodového spoje dvou zařízení, které komunikují pomocí rozhraní RS232. Používá rádiovou frekvenci 433.92 MHz, kterou je v České Republice možné používat bezplatně na základě generální licence Českého telekomunikačního úřadu. Přestože byl RadioSpoj232 navržen pro sběr dat z měřičů tepla a průtokoměrů firmy EESA, funguje i s jinými zařízeními, které dodrží stejné parametry spojení RS232. Maximální propustnost spoje v jednom směru je 175 bajtů za sekundu, nejkratší odezva je okolo 80 ms.

5.1 Technikcké vlastnosti

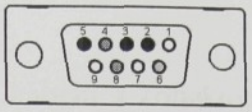
Spojení lze vytvořit spolehlivě na vzdálenost 300 až 400 m volným prostorem. S rostoucí vzdáleností přibývají chyby rámců a spojení je tím pomalejší. Ve vzdálenosti okolo 1000 m dosah končí. Dosah dramaticky snižuje pozice v nepřímé viditelnosti vlivem terénu. Lze ale říct, že do vzdálenosti 100 metrů by se mělo spojení navázat i bez přímé viditelnosti.

Rádiové spojení se vytváří mezi stanicí Master a stanicí Slave. Stanice Slave je určena pro instalaci k měřičům tepla, stanice Master je určena pro připojení na straně sběru dat. Adresa stanice Slave se nastavuje na otočném přepínači. Na stanici Master volí obsluha adresu jedné z patnácti Slave stanic pomocí tlačítka, zvolená adresa se zobrazuje na displeji. Obě stanice signalizují pomocí LED diod vysílání a procházející data. Anténu je možné používat buď vestavěnou, nebo pro zvýšení dosahu externí anténu s BNC konektorem.

Komunikační protokol používá 16 adres. Adresu 0 používá Master, adresy 1 až 15 používají stanice Slave. Pokud je v rámci dosahu stanice Master umístěno více stanic Slave, musí mít každá jinou adresu.

Ke komunikaci mezi dvěma stanicemi se používá jeden rádiový kanál v poloduplexním režimu. K posílání dat se data rozdělují do balíčků s proměnnou délkou. Stanice odpovídají protistraně okamžitě po přijmutí balíčku, takže zabírají celý kanál. Je tedy možné provozovat na jednom místě jen jeden pár rádiových rozhraní. V souvislosti s podmínkami pro používání frekvence 434 MHz by rádiová rozhraní neměla být v provozu déle než 6 minut v hodině. Kvůli provozu jiných zařízení na frekvenci 434 MHz není možné vyloučit náhodnou interakci s jiným rádiovým provozem. Přestože je pravděpodobnost interakce velmi malá, neměla by být k rádiovým rozhraním připojována zařízení, jejichž důležité vnitřní stavy mohou být měněny rozhraním RS232 (výmaz dat, změna konstant měřiče tepla, atd.).

Parametry připojení RS232 jsou 9600 baud, 8 bit, bez parity, 1 stop bit. Konektor na rádiovém rozhraní je typ sub-D s devíti dutinkami. Zapojení konektoru je standardní, pro připojení k počítači se může použít prodlužovací kabel. Nepoužité signály nejsou zapojené.



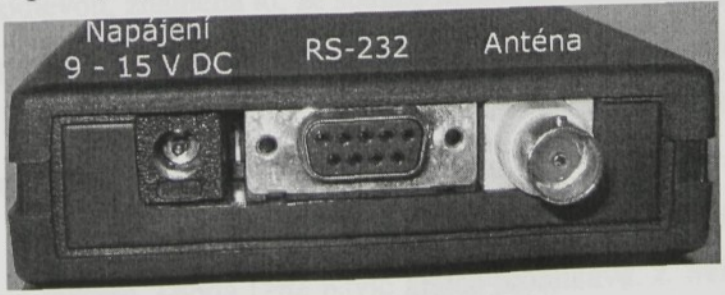
obrázek 25: Konektor RS232

pin	signál	směr	poznámka
5	GND	společná zem	
2	TxD	výstup	
3	RxD	vstup	
4	DTR	vstup	
6	DSR	výstup	spojený s DTR
8	CTS	vstup	

Rádiové rozhraní je možné používat v třívodičovém zapojení nebo v pětivodičovém zapojení se signály DTR a CTS. Používání handshakingu je autodetekováno podle stavu signálu DTR. Po zapojení napájení je signál DTR ignorován do té doby, dokud není zjištěn jeho aktivní stav (terminálový program se připojil k portu). Poté již funguje DTR handshaking normálně až do vypnutí napájení. Signál CTS je vysílán vždy.

Rádiové spojení je pomalejší nežli RS232, z toho vyplývají vlastnosti, které je nutné respektovat. Pro malé bloky dat do 120 bajtů je možné rozhraní používat bez hardwarového handshakingu, pro bloky nad 120 bajtů je handshaking nutný, jinak by docházelo ke ztrátám dat. Maximální velikost balíčku v rádiovém spojení je 28 bajtů. Po přijmutí balíčku jsou data odesílána portem RS232 okamžitě a co nejrychleji s přihlédnutím ke stavu signálů handshakingu. Na terminálu tedy data přibývají skokově. Pokud má koncová aplikace ve svém protokolu nastavený příliš krátký time-out na dokončení bloku, může tato aplikace selhat.

Rozhraní je napájeno síťovým napáječem s napětím 9 ÷ 15 V, nebo ze zásuvky v automobilu. Napáječ musí mít dutý konektor s vnějším průměrem 5,5 mm a vnitřním průměrem 2,2 mm. Rozhraní je chráněno proti přepólování napáječe.



obrázek 26: Konektory rádiového rozhraní RadioSpoj232

5.2 Stanice Slave

Stanice Slave je určena ke připojení k měřiči tepla. Před uvedením do provozu se musí nastavit adresa na otočném přepínači z rozsahu 1 až 15. Pokud je na otočném přepínači nula, je stanice neaktivní.

K měřiči tepla se připojuje pomocí kabelu s konektorem sub-D 9M (9 kolíků). Na straně měřiče tepla se vodiče TxD, RxD a GND připojí do svorkovnice.

5.3 Stanice Master

Stanice Master je určena pro připojení k počítači nebo ručnímu terminálu ESTER02. K ručnímu terminálu se připojuje pomocí speciálního kabelu s konektorem RJ45, k počítači se připojuje prodlužovacím kabelem. Konektor je shodný se stanicí slave.

Stanice Master obsahuje tlačítko pro volbu adresy a sedmisegmentový displej, na kterém se adresa zobrazuje. Po zapojení napájení je na displeji nula a spojení je vypnuté. Krátkým stiskem tlačítka se změní adresa na o jedna vyšší. Krátkými stisky se navolí potřebné číslo Slave stanice a spojení se zahájí asi do jedné sekundy. Dlouhý stisk tlačítka nastavuje nulu a spojení vypíná.

Pokud je potřeba, aby dvě zařízení vytvářela stálý pár bez nutnosti obsluhy stanice Master, použije se otočný přepínač i na stanici Master. Na přepínači se zvolí číslo příslušné Slave stanice a po restartování je adresa nastavena na toto číslo. Funkce tlačítek a displeje přitom zůstává zachována.

Pokud zvolená stanice Slave neodpovídá, snaží se stanice Master pokračovat ve spojení neustále.

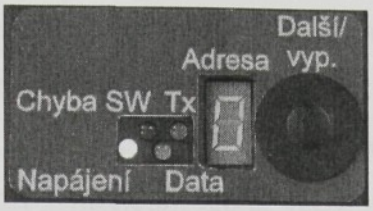
5.4 Význam LED diod

Na vrchním krytu jsou umístěny 4 signalizační diody. Zelená dioda signalizuje zapojené napájení.

Žlutá dioda označená "Tx" signalizuje, že rádiové rozhraní právě vysílá. Pokud dioda Tx bliká krátce s dlouhou prodlevou, znamená to, že stanice Master se snaží komunikovat, ale zvolená stanice Slave neodpovídá. Pokud dioda Tx bliká frekventovaně, pak právě probíhá spojení. Podle poměru dob, kdy dioda svítí a kdy nesvítí lze i odhadnout, kterým směrem proudí data.

Žlutá dioda označená "Data" svítí v případě, že rádiovým zařízením právě prochází data, nebo že jsou nějaká data uložena v mezipaměti. V zapnutém stavu setrvává ještě asi jednu sekundu po zaznamenání komunikace. Dioda Data navíc krátce problikává. Z frekvence problikávání se dá určit, zda je nainstalován správný krystal.

Červená dioda "Chyba SW" signalizuje chybu v software. Za normálních okolností by se neměla nikdy rozsvítit. Pokud se rozsvítí, znamená to přeplnění některé mezipaměti. Zároveň s touto diodou se na displeji zobrazí číslo chyby. Po krátké chvíli se rádiové rozhraní vyresetuje.



obrázek 27: Čelní panel stanice master

5.5 Ověření funkce se systémy firmy EESA

Ve spolupráci s firmou EESA jsem ověřil funkci rádiového rozhraní RadioSpoj232v1.0 na komunikaci mezi měřičem tepla MT500 a terminálem ESTER02, a na komunikaci mezi měřičem tepla a programem View32. Obě aplikace fungovaly bezchybně a ve shodném nastavení jako při komunikaci po kabelu. Nízká rychlost rádiového přenosu se projevila v delší době potřebné k načtení dat.

5.6 Tabulka technických vlastností

Parametr	min.	typ.	max.	jednotka	poznámka
Frekvence		433,92		MHz	
RF výkon			+10	dBm	
Dosah		300	1000	m	
Počet adres		15 + 1			
Bitová rychlost		2500		baud	
Propustnost		175*	224	bajt/s	*maximum v jednom směru
Odezva	80	<250		ms	
Zpoždění	40		375	ms	jen pokud je dobrý signál a mezipaměť prázdná
Mezipaměť		2×127		bajt	
Rychlost RS-232		9600		baud	8bit, bez parity, 1stop bit
Napájecí napětí	9		15	V	
Proud příposlech Slave		35		mA	
Proud spojení Slave		44		mA	
Proud nečinný Master		55		mA	vyšší hodnoty proudu způsobuje LED displej
Proud spojení Master		60	82	mA	

6 Závěr

Pro experimentální desku firmy EESA jsem napsal nový komunikační software, který vytváří rádiové rozhraní pro sériový port RS232. Rozhraní se chová transparentně, tj. data která jsou přečtena z portu RS232 jsou okamžitě přenášena druhé straně. Na portu RS232 se nepoužívá žádný protokol, proto je možné připojit rozhraní k jakémukoliv zařízení, které dodrží stejnou rychlost a parametry spoje RS232. Dále jsem navrhnul hardware vlastního rádiového rozhraní ve dvou verzích. Verze slave patří k měřičům tepla, verze master se použije na straně sběru dat. Od každé verze jsem postavil a oživil jeden kus. Na verzi slave se nastavuje jedna z patnácti adres pomocí otočného přepínače. Na verzi master volí obsluha adresu stanice slave pomocí tlačítka. Aktuální adresa se zobrazuje na displeji.

Software implementuje fyzickou a spojovou vrstvu OSI modelu. Fyzická vrstva používá kód Manchester s rychlostí 2500 bitů za sekundu, rozpoznávání je odolné proti drobným poruchám signálu. Spojová vrstva vytváří spolehlivý kanál. Používá při tom protokol, který má vestavěn algoritmus pro opakování rámce v případě výskytu chyby. Rámce jsou zabezpečeny pomocí šestnáctibitového CRC kódu. V protokolu se také pamatuje na možnost kompatibility s případnými budoucími verzemi protokolu.

Propustnost spoje v jednom směru je 175 bajtů za sekundu, což je pro sběr dat více než dostačující. Spojení se spolehlivě naváže na vzdálenost do 400 metrů v přímé viditelnosti, což je podstatné zlepšení vůči původní verzi rozhraní vyráběné firmou EESA.

Při ověřování funkčnosti rozhraní s měřičem tepla a ručním terminálem proběhlo vše v pořádku. Sběr dat je možné provádět i pomocí počítače s programem View. Rádiová rozhraní je možné používat i mezi dvěma počítači. Pro sběr dat je toto rádiové rozhraní plně dostačující, ale jiná využití jsou limitována nízkou rychlostí spoje.

Literatura

[1] Fairhurst, G.: Communications Engineering course, studijní materiály, on-line z http://www.erg.abdn.ac.uk/users/gorry/course/

[2] Adámek, J.: Kódování, SNTL Praha, 1989

[3] Český telekomunikační úřad: Generální licence č. GL-30/R/2000

[4] Katalogový list radiomodulu AUREL XTR-434 L

[5] Technická dokumentace měřiče tepla MT500, firma EESA, 2004

[6] Hlava J.: Prostředky automatického řízení II, skriptum ČVUT Fakulta strojní, 2000, on-line z http://www.fm.vslib.cz/~krtsub/fm/par/Skripta_PAR.pdf

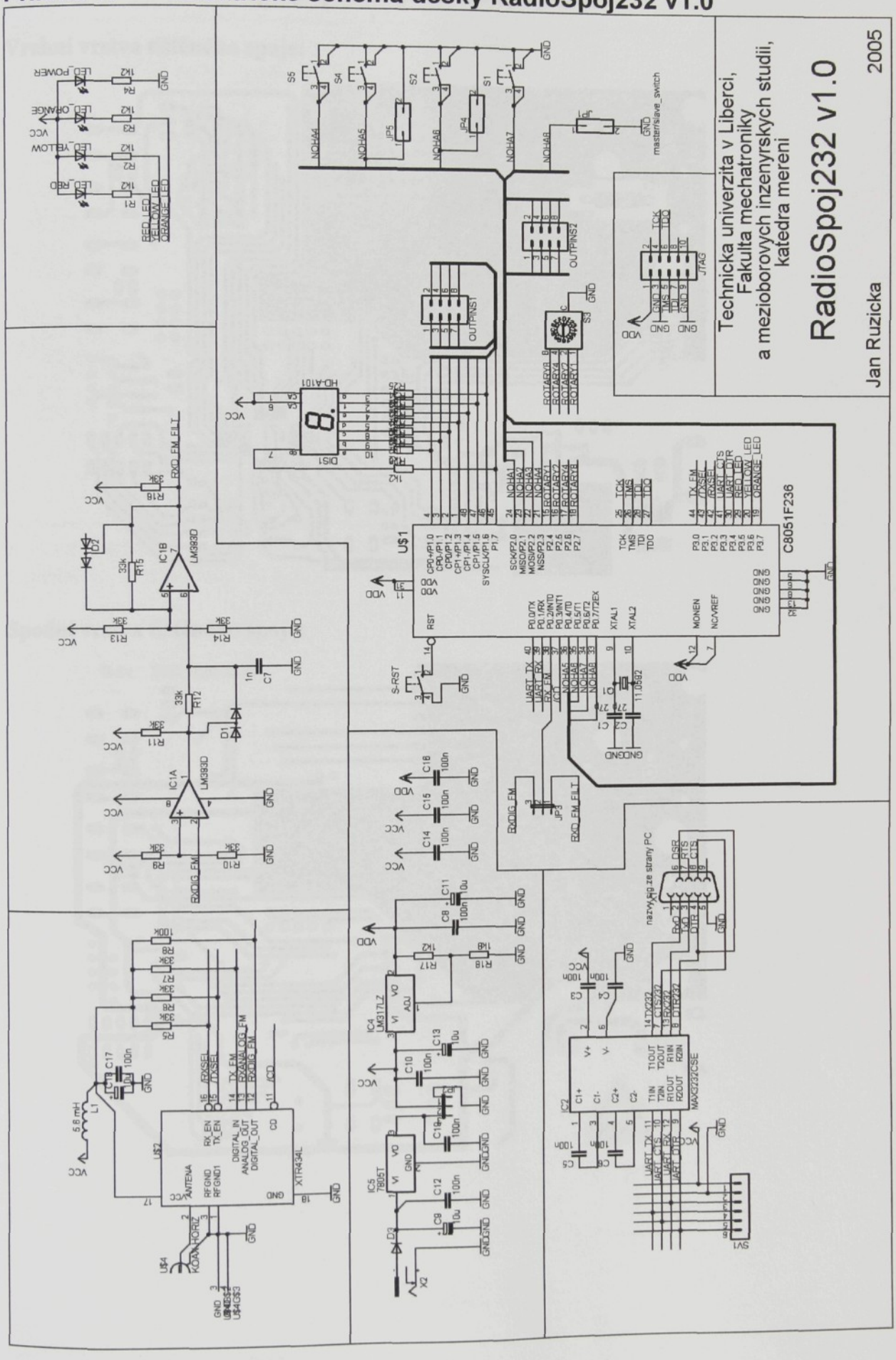
[7] Tůma, J.: Lineární algebra I, kapitola 8 - Samoopravné kódy, texty k přednáškám, 2002, on-line z http://www.karlin.mff.cuni.cz/~tuma/2002/NLinalg8.pdf

[8] Benko, P.: Protokoly a Rozhrania, studijní materiály, on-line z http://alf.fei.tuke.sk/pai/

[9] Katalogový list procesoru Silabs C8051F236

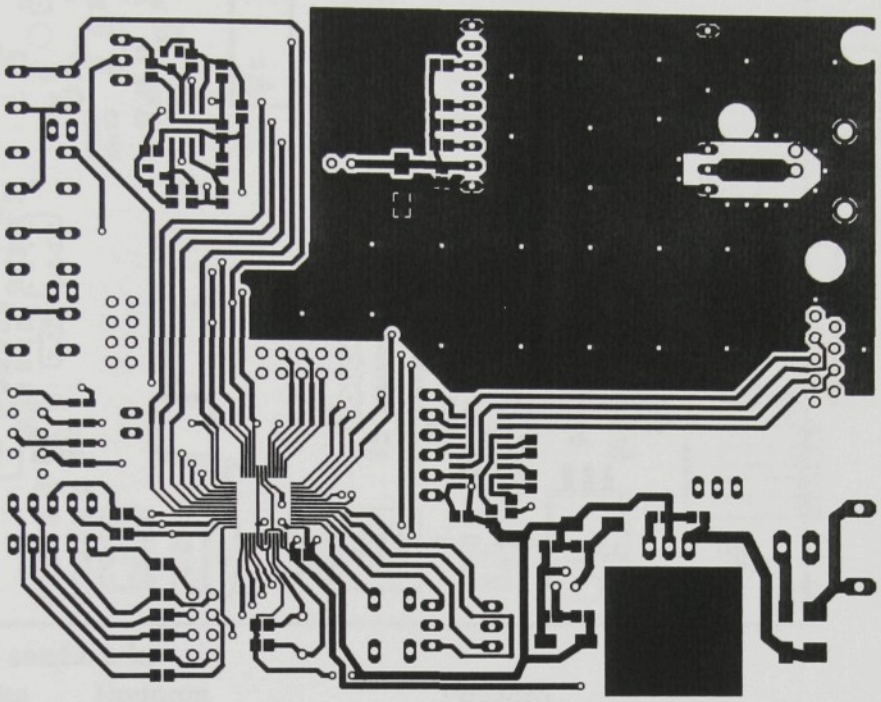
Přílohy

Příloha 1: Elektrické schéma desky RadioSpoj232 v1.0

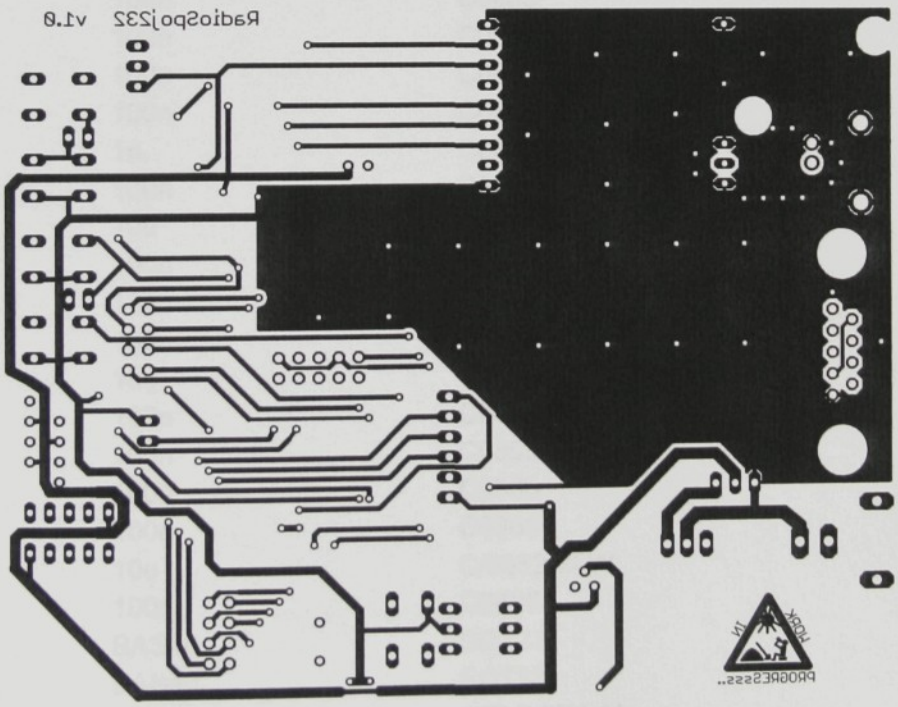


Příloha 2: Tištěný spoj desky RadioSpoj v1.0

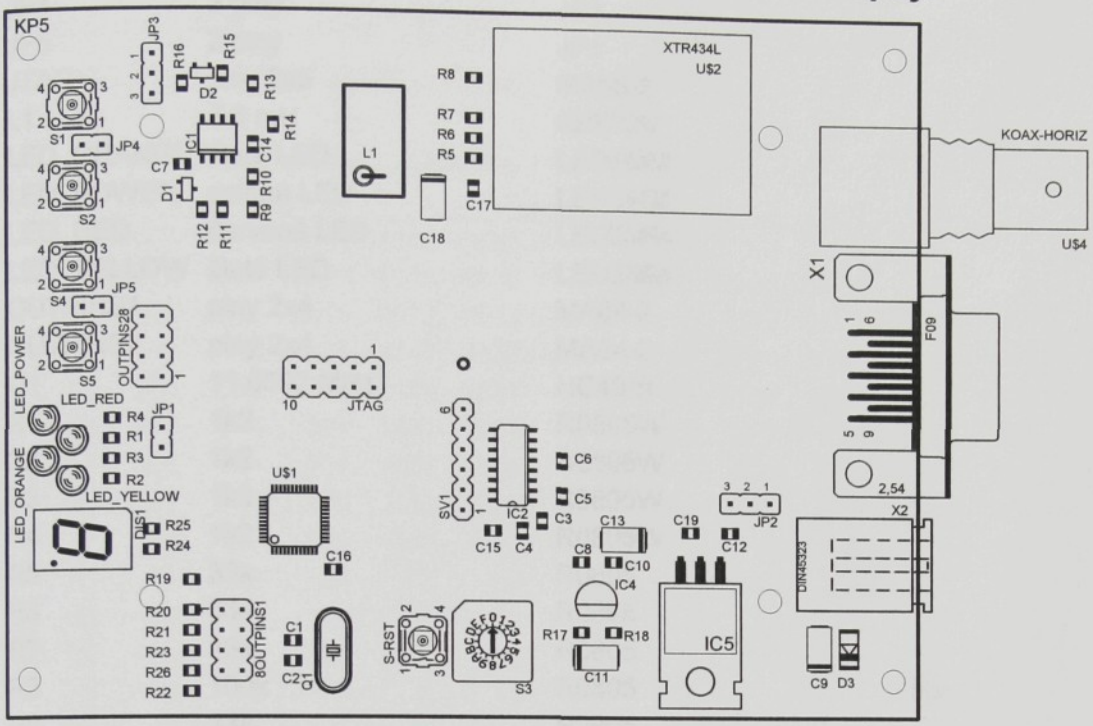
Vrchní vrstva tištěného spoje:



Spodní vrstva tištěného spoje:



Příloha 3: Rozmístění součástek na tištěném spoji



Seznam součástek:

Součástka	Hodnota	Pouzdro
C1	27p	C0805
C2	27p	C0805
C3	100n	C0805
C4	100n	C0805
C5	100n	C0805
C6	100n	C0805
C7	1n	C0805
C8	100n	C0805
C9	10u	C/6032-28W
C10	100n	C0805
C11	10u	C/6032-28W
C12	100n	C0805
C13	10u	C/6032-28W
C14	100n	C0805
C15	100n	C0805
C16	100n	C0805
C17	100n	C0805
C18	10u	C/6032-28W
C19	100n	C0805
D1	BAS40	SOT23
D2	BAS40	SOT23
D3	SM4007	MELF-MLL41
DIS1	KingBright HD-A101	HDSP-A
IC1	LM393D	SO08
IC2	MAX3232CSE	SO16
IC4	LM317LZ	TO92
IC5	7805T	TO220H
JP1	2 piny	JP1
JP2	3 piny v 1 řadě	JP2

JP3	3 piny v 1 řadě	JP2
JP4	2 piny	JP1
JP5	2 piny	JP1
JTAG	piny 2x5	MA05-2
L1	5.6 mH	0207/2V
LED_ORANGE	žlutá LED	LED3MM
LED_POWER	zelená LED	LED3MM
LED_RED	červená LED	LED3MM
LED_YELLOW	žlutá LED	LED3MM
OUTPINS1	piny 2x4	MA04-2
OUTPINS2	piny 2x4	MA04-2
Q1	11.0592 MHz	HC49/S
R1	1k2	R0805W
R2	1k2	R0805W
R3	1k2	R0805W
R4	1k2	R0805W
R5	33k	R0805
R6	33k	R0805
R7	33k	R0805
R8	100k	R0805
R9	33k	R0805
R10	33k	R0805
R11	33k	R0805
R12	33k	R0805
R13	33k	R0805
R14	33k	R0805
R15	33k	R0805
R16	33k	R0805
R17	1k2	R0805
R18	1k8	R0805
R19	1k2	R0805
R20	1k2	R0805
R21	1k2	R0805
R22	1k2	R0805
R23	1k2	R0805
R24	1k2	R0805
R25	1k2	R0805
R26	1k2	R0805
S-RST	mikrotlačítko	B3F-10XX
S1	mikrotlačítko	B3F-10XX
S2	mikrotlačítko	B3F-10XX
S3	otočný přepínač 16 poloh	P103
S4	mikrotlačítko	B3F-10XX
S5	mikrotlačítko	B3F-10XX
SV1	mikrotlačítko	MA06-1
U$1	Silabs C8051F236	TQFP48
U$2	Aurel XTR434L	
U$4	konektor BNC-horizontální	
X1	Konektor RS232 9 dutinek	F09H
X2	Konektor napájení ROKA	737992-5

Příloha 4: Tabulka četnosti výskytu preambulí

Četnost sekvencí formátu ...101010 111111... ve výstupu dekodéru Manchesteru, který dekódoval šum získaný z radiomodulu.

Délka preambule \ Délka konstantní části preambule	1	2	3	4	5	6	7	8	9	10	11	12	13
8	39542	35827	38050	34642	36788	33796	35877	33104	0	0	0	0	0
9	17915	18809	17023	18109	16436	17594	16175	17233	15871	0	0	0	0
10	9325	8536	9011	8077	8613	7914	8412	7779	8326	7545	0	0	0
11	4258	4484	4077	4317	3850	4175	3797	4054	3759	3940	3605	0	0
12	2168	2052	2160	1948	2047	1845	2063	1848	1991	1778	1873	1732	0
13	987	1054	997	1041	924	980	905	1007	918	928	829	906	826
14	505	481	514	474	506	426	482	436	523	425	428	430	427
15	241	256	235	243	221	233	204	247	227	248	203	227	185
16	119	113	124	115	110	105	109	102	135	112	116	109	97
17	54	55	58	67	52	59	51	56	54	67	57	59	46
18	29	29	30	34	26	27	25	29	30	24	34	30	25
19	12	14	16	14	16	12	12	12	14	13	10	17	18
20	6	4	7	6	10	9	6	4	7	6	4	3	11
21	4	1	3	0	3	6	5	2	2	4	1	2	2
22	2	1	1	0	0	2	4	2	1	2	0	1	2
23	2	0	1	0	0	0	2	2	1	1	0	0	1
24	1	0	0	0	0	0	0	2	1	1	0	0	0
25	1	0	0	0	0	0	0	0	1	1	0	0	0

Délka preambule \	14	15	16	17	18	19	20	21	22	23	24	25
8	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0
14	399	0	0	0	0	0	0	0	0	0	0	0
15	204	195	0	0	0	0	0	0	0	0	0	0
16	85	98	97	0	0	0	0	0	0	0	0	0
17	46	39	43	54	0	0	0	0	0	0	0	0
18	20	19	20	23	31	0	0	0	0	0	0	0
19	14	6	8	11	13	18	0	0	0	0	0	0
20	9	4	3	4	8	9	9	0	0	0	0	0
21	4	3	2	3	2	6	4	5	0	0	0	0
22	0	1	1	2	1	2	2	2	3	0	0	0
23	0	0	1	1	1	1	0	1	2	1	0	0
24	0	0	0	1	1	1	0	0	1	1	0	0
25	0	0	0	0	1	1	0	0	0	1	0	0