Abstrakt

Předložená bakalářská práce se zabývá návrhem a realizací hardware a software testovacího zařízení slave pro sběrnice LIN.

Testovacího zařízení slave je určeno pro testování a vývoj zařízení typu master. Pro realizaci testovacího zařízení slave byl použit osmibitový mikrokontrolér firmy Freescale typu MC9S08DZ60 a LIN transceiver MC33661.

Navržené testovací zařízení slave podporuje protokol LIN verze 1.3. Zařízení komunikuje s nadřazeným PC pomocí linky RS232. Na tuto linku zařízení vysílá informace o přenosu a je schopné zpracovávat příkazy posílané přes linku RS232.

Klíčová slova

Testovací zařízení slave pro sběrnici LIN, mikrokontrolér MC9S08DZ60.

Abstract

Submitted bachelor thesis occupies by implementation and realization hardware and software of testing node for LIN bus.
Testing slave node is used for testing and making master node. For realization slave node was chosen 8 bit microcontroller from company Freescale type MC9S08DZ60 and LIN transceiver MC33661.
Created testing slave node supports LIN version 1.3. Node communicates with superior computer through RS232 bus. On this bus node sends information about transfer and it is able to process commands which are sending through RS232 bus.

Keywords

Testing slave node for LIN bus, microcontroller MC9S08DZ60.

Bibliografická citace

DYČKA, O. *Testovací zařízení typu slave pro sběrnici LIN*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 55 s. Vedoucí bakalářské práce Ing. Tomáš Macho, Ph.D.

Prohlášení

„Prohlašuji, že svou bakalářskou práci na téma "*Testovací zařízení typu slave pro sběrnici LIN*" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury
na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.“

Poděkování

Chtěl bych poděkovat všem lidem, bez nichž by nemohla tato práce vzniknout, především svému vedoucímu práce Ing. Tomáši Machovi, Ph.D., za vedení v průběhu realizace testovacího zařízení a poznámky k práci. Dále bych chtěl poděkovat rodině, zejména za jejich pomoc a podporu při mém studiu.

OBSAH

SEZNAM OBRÁZKŮ, TABULEK A GRAFŮ

Obrázky:

Tabulky:

Grafy:

1. ÚVOD

V dnešní době můžeme pozorovat, že automatizace proniká do všech zařízení, kterými se člověk obklopuje. Platí staré pravidlo, že člověk je od přírody tvor pohodlný. A proto rádi používáme zařízení, která většinu věcí udělají za nás. Všechny pokroky v řízení nemají pouze za cíl zvětšit pohodlí člověka, ale také přispívají k bezpečnosti a efektivitě celého pracovního procesu. Velký pokrok vývoje automatizace můžeme zaznamenat mimo jiné v automobilovém průmyslu. Lidé cestují velké vzdálenosti za prací či zábavou, a tak se automobil v rodině stává potřebným prostředkem. Proto automobilové společnosti vyvíjejí nejrůznější vylepšení v oblasti řízení motoru, zvětšení pohodlí a v neposlední řadě se zabývají ekologickými alternativami pohonů. Moderní vůz je obklopen množstvím snímačů a regulátorů, které vytváří systémy, které jsou pro nás dnes samozřejmostí. Například pro bezpečnostní systémy ABS, ASR, ESP, nebo moderní řízení vstřikování paliva Common Rail či ovládání klimatizace, zrcátek i sedadel. Tyto systémy mají přehled o všech funkcích v automobilu, a proto se dají dobře využít v servisech, kde se řídící jednotky jednotlivých systémů propojí s diagnostickým zařízením, takže hledání závady je snazší. V současné době je v automobilech hojně užívaná sběrnice CAN a LIN. Každá sběrnice má své výhody, a proto jsou standarty zdokonalovány nebo také vytvářeny nové, jako například sběrnice Flexray.

Předložená práce se zabývá problematikou návrhu a realizace testovacího slave zařízení sběrnice LIN. Tato sběrnice byla vyvinuta pro automobilový průmysl pro řízení jednoduchých zařízení. Komunikace mezi zařízeními je typu master – slave (tedy pán a otrok). Přenos po sběrnici je řízen masterem, který vytváří hlavičku zprávy a jednotlivé slave jednotky hlavičku zpracují a vykonají požadovanou akci. Byly vyvinuty tři verze sběrnice – 1.3, 2.0 a poslední verze 2.1.

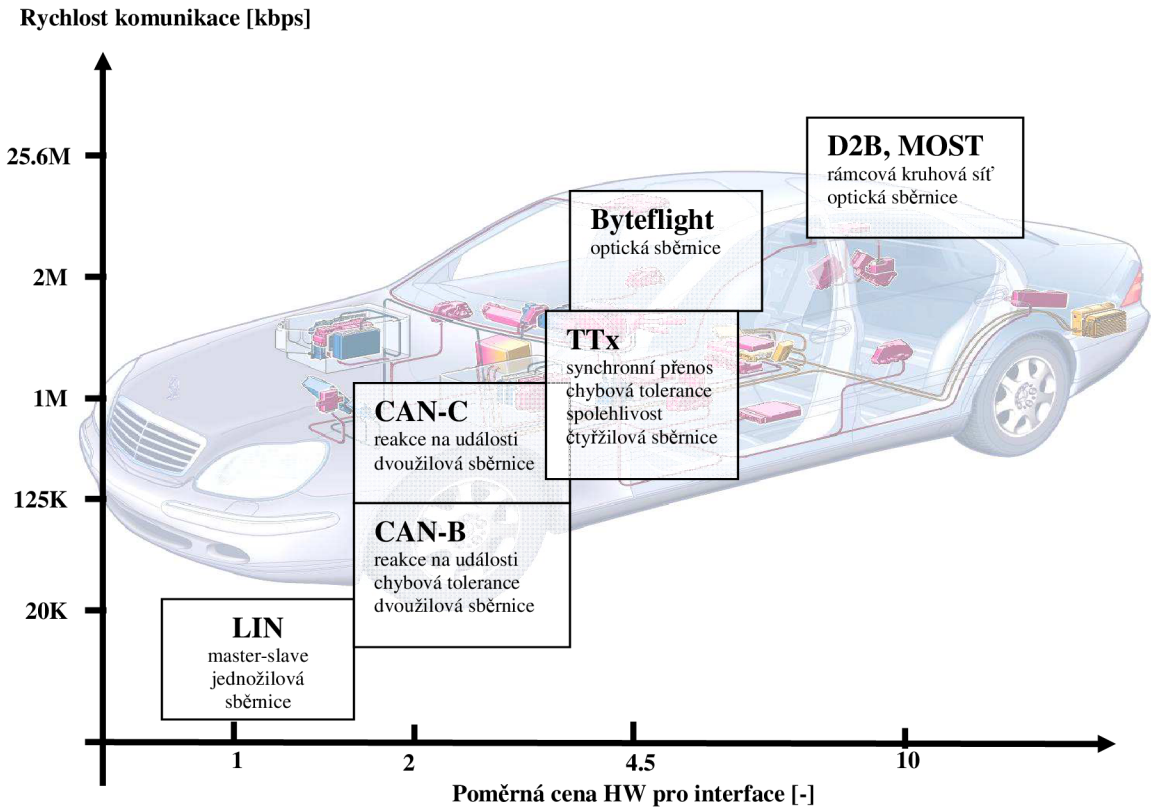
Pro realizaci byl použit procesor řady HCS08 MC9S08DZ60 firmy FREESCALE, předního výrobce polovodičových součástek. Zařízení slouží k testování funkcí a ladění zařízení master.

2. SBĚRNICE LIN

2.1 VLASTNOSTI SBĚRNICE LIN

Local Interconnect Network (LIN) je sběrnice komunikující po jednom vodiči. Byla navržena pro automobilový průmysl skupinou LIN Consortium, která byla založena v roce 1998 a seskupovala pět automobilek – Audi AG, BMW AG, DaimlerChrysler AG, Volkswagen AG, Volvo Car Corporation a firmy Motorola,Inc. a Volcano Communication Technologies AB. Standart LIN byl dokončen 2. 2. 2000. V dnešní době se používá spolu s dalšími standarty především v automobilovém průmyslu. Tato sběrnice si buduje místo vedle standartu CAN hlavně pro svoji jednoduchost. Sběrnice byla navržena jako nízkorychlostní komunikace, která ovládá zařízení v časech okolo 100 ms. Sběrnice LIN nemá nahradit sběrnici CAN, respektive Flexray, ale doplnit je v aplikacích tam, kde by použití sběrnice CAN bylo finančně neefektivní. Velká výhoda sběrnice je nejenom v připojení jedním vodičem, ale v celé své jednoduchosti – nemá velké požadavky na hardware a protokol nemusí být striktně kompletní, takže můžeme implementovat jen část protokolu, kterou potřebujeme. Ovšem i za jednoduchost se platí. Komunikace po jednom vodiči je náchylná na elektromagnetické rušení daleko více než kroucená dvoužilová kabeláž pro CAN. Samotný standart LIN nedosahuje tak velké rychlosti jako CAN, a proto má na starosti ovládání jednodušších systémů zajišťující naše pohodlí v autě, než přesné rychlé řízení například vstřikovací jednotky v motoru. Proto se tato sběrnice používá pro ovládání zrcátek, ovládání elektricky stahovaných oken, polohování sedadel, ovládání klimatizace a další aplikace, které nás obklopují v moderních automobilech a u nichž nepotřebujeme rychlou odezvu.

V grafu 1 je znázorněn přehled některých sběrnicí používaných v automobilovém průmyslu a jejich poměrnou cenou za interface k rychlosti sběrnice. Cena zahrnuje finance vynaložené za realizaci požadovaného hardwaru pro daný interface.



Graf 1 Závislost rychlosti přenosu sběrnicí na ceně HW pro interface

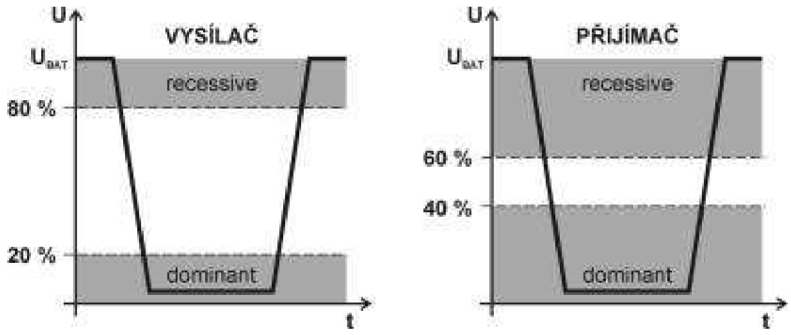
Sběrnice LIN je otevřený standart, a proto může najít své uplatnění i mimo automobilový průmysl. Podporu sběrnice nalezneme u firem Freescale, Texas Instruments i Microchip. Například firma Freescale vyrábí procesory s podporou sběrnice, obvody fyzického interface a vytváří příklady využití sběrnice pro řízení například robota se čtyřmi servomotory (dokument AN2470.pdf [7]), nebo ukázkový program pro ovládání hodin pomocí sběrnice LIN (dokument AN2103.pdf [6]).

Maximální délka sběrnice je 40 m. Maximální rychlost přenosu je 20 kbps. Sběrnice je typu master/slave, kde jedno zařízení master řídí komunikaci se zařízeními slave. Z toho plyne, že přenos je multiplexní – více zařízení sdílí informace po jednom vodiči. Komunikace je obousměrná a může být připojeno až 16 zařízení typu slave k jednomu řídícímu masteru. Přenos je asynchronní. Proto protokol obsahuje v hlavičce synchronizační impulsy, během kterých se jednotlivá zařízení zasynchronizují. Toto je vlastnost, která přidává na jednoduchosti fyzického provedení sběrnice, ale snižuje její celkovou maximální rychlost. Výhoda LINu je, že není požadován přesný hodinový kmitočet u zařízení slave právě proto, že v protokolu je implementována synchronizace masteru při každé komunikaci se slave. Proto slave nepotřebuje přesný krystal pro generování časových impulsů, ale stačí pro generování hodin oscilátor RC. Povolená odchylka frekvence oscilátoru zařízení slave od přesného hodinového signálu masteru je ±14%. Toto je další výhoda, která ve výsledku přispívá k nízké ceně kompletní sběrnice. V protokolu je také implementovaný sleep režim. Ten umožňuje uspat zařízení, čímž dojde ke snížení příkonu zařízení slave. Sleep režimy ovládá master, jednotlivé jednotky uspává a probouzí.

2.2 FYZICKÁ VRSTVA

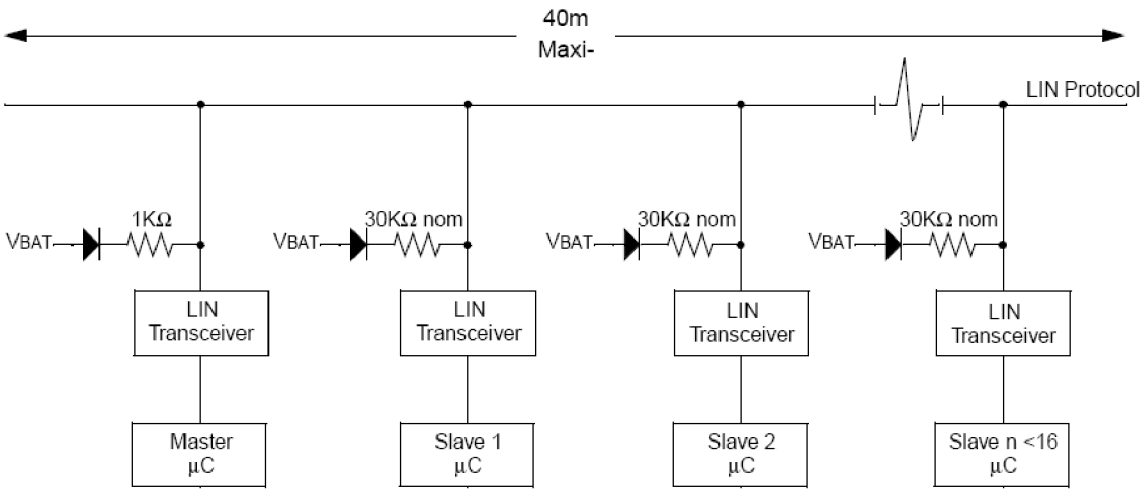
Fyzická vrstva LIN vychází z normy ISO 9141, která definuje povinnou fyzickou vrstvu pro komunikaci s emisními systémy. Ovšem tato norma musela být přizpůsobena vlastnostem sběrnice LIN. Jak již bylo zmíněno, jednožilové vedení způsobuje problémy ohledně rušení. Sběrnice nesmí rušit okolí – náběžné a sestupné hrany nesmí být ostré a také jsou upraveny rozhodovací úrovně, aby malá změna napětí baterie nezpůsobila nefunkčnost celého zařízení. Jednotlivé rozhodovací úrovně jsou od sebe odděleny pásmem necitlivosti, do kterého by se logika neměla nikdy dostat. Úroveň logické nuly se nazývá dominant a úroveň logické jedničky se jmenuje recessive.

Na obrázku 1 jsou zobrazeny jednotlivé úrovně napětí odpovídající logickým hodnotám.



Obr. 1 Napěťové úrovně logických stavů

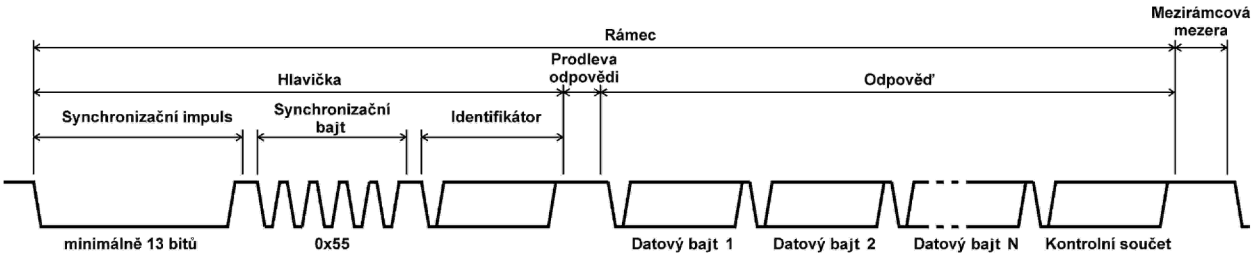
Fyzická vrstva je navržena na napětí automobilových baterií v rozmezí 9V - 18V. Jednotlivé rozhodující napěťové úrovně jsou vztaženy relativně k napětí v automobilu. Elektronika na sběrnici je testována a musí vydržet maximální napětí 40V a provozní teplotu -40 ^{0}C až 125 ^{0}C. Mikrokontroléry jsou odděleny od sběrnice vrstvou driver/reciver, aby mohli pracovat na svém typickém napětí 5V. Jednotka master se připojuje ke sběrnici přes odpor 1kΩ a jednotka slave přes odpor 20 kΩ – 47 kΩ (typicky 30 kΩ). Rozvržení obvodů na sběrnici je zobrazeno na obrázku 2.



Obr. 2 Struktura připojených zařízení na sběrnici

2.3 RÁMEC ZPRÁVY

Formát zprávy je jednotný a používá se k adresaci zařízení, synchronizaci i přenosu dat mezi zařízeními. Obrázek 3 popisuje v časové oblasti význam jednotlivých pulsů.



Obr. 3 Formát zprávy protokolu LIN

Základní rámec zprávy se skládá z hlavičky (header) a odpovědi (response). Hlavičku vysílá pouze master a obsahuje data pro inicializaci celého přenosu. Je v ní definována velikost dat, a kterému zařízení je to adresováno.

Hlavička obsahuje následující části:

Synchronizační puls – break field

Na začátku hlavičky je synchronizační puls o minimální délce 13 bitů. Jakmile se na lince objeví break, zařízení slave čekají na příjem zprávy. Často slave jednotky s RC oscilátorem usínají a tím ztrácí synchronizaci s masterem. Proto je break dlouhý 13 bitů, aby i uspané nesynchronizované jednotky byly schopny tento signál detekovat.

Synchronizační pole

Poté hlavička obsahuje synchronizační pole. Synchronizační pole má hodnotu 0x55 a během těchto impulsů se jednotky slave přizpůsobí taktu přenosu vysílaného masterem.

Chráněné identifikační pole – Protected identificator field (PID)

Rozložení významových bitů v bajtu PID je zobrazeno na obrázku 4. Bity ID0 – ID5 jsou bity identifikátoru a doplňkové bity P0 a P1 mají význam kontrolních bitů v PID.

Obr. 4 Význam jednotlivých bitů v chráněném identifikačním poli

PID obsahuje identifikátor, který v sobě nese informaci o příjemci a velikosti datové zprávy. Identifikátor je složen z 6 bitů, takže teoreticky máme k dispozici 64 identifikátorů použitelných v přenosu. Identifikátory jsou rozděleny do jednotlivých sekcí podle jejich použití v tabulce 1:

Hodnota identifikátoru		Význam a použití identifikátoru
Dekadicky	Hexadecimálně	
0 – 59	0x00 - 0x3B	použito pro přenos obecných dat
60 a 61	0x3C - 0x3D	použito pro přenos diagnostických dat
62	0x3E	rezervováno pro uživatelem definovaný rámec
63	0x3F	rezervováno pro budoucí využití

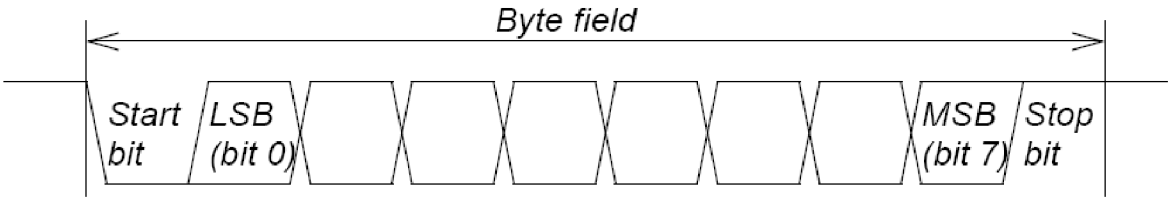
Tab. 1 Použití identifikátorů

Na konci PID bajtu se nachází kontrolní bity P0 a P1. Paritní bity zajišťují základní kontrolu o správnosti přenosu identifikátoru. Parita je vypočtena podle vzorce 1 jako exkluzivní součet jednotlivých bitů identifikátoru:

$$P0 = ID0 \oplus ID1 \oplus ID2 \oplus ID4$$
$$P1 = \neg(ID1 \oplus ID3 \oplus ID4 \oplus ID5) \quad (1)$$

Odpověď – datová část

Datová část může být vysílána masterem i slavem. Data jsou posílána po bajtech ve standartu UART, jak je to ukázáno na obrázku 5. Jedinou výjimku tvoří synchronizační puls o délce 13 bitů.



Obr. 5 Struktura dat v jednom bajtu

Start bit je prezentován jako logická nula – dominant a stop bit je logická jednička – recessive. Jako první je posílán LSB a na konci MSB.

Počet bajtů v datové oblasti definuje identifikátor. Můžeme přenášet 2, 4 a 8 bajtů a závislost počtu bajtů na identifikátoru je v tabulce 2.

ID5	ID4	Počet datových bajtů
0	0	2
0	1	2
1	0	4
1	1	8

Tab. 2 Počet datových bajtů definovaných bity identifikátoru

Od verze LIN 2.0 můžeme vysílat bajtová pole. Pak je uložen LSB v prvním bajtu a v posledním MSB (little endian).

Kontrolní součet – checksum

Rámec zprávy je zakončen kontrolním součtem dat. Klasický kontrolní součet obsahuje 8 invertovaných bitů i s carry, což se používalo u verze LIN 1.3. Od verze 2.0 se používá rozšířený kontrolní součet, který obsahuje i chráněný identifikátor. Tím tento kontrolní součet pokrývá kontrolu přenosu dat i identifikátoru.

2.4 KOMPATIBILITA SBĚRNICE LIN

Sběrnice LIN se neustále vyvíjí a v současné době je aktuální verze 2.1. Předchozí verze byly 2.0 a 1.3 a v následující části je uveden popis rozdílů a kompatibility jednotlivých verzí sběrnice.

2.4.1 Hlavní rozdíly mezi verzí LIN 1.3 a 2.0

- byla povolena bajtová pole o maximální velikosti 8 bajtů
- byly odstraněny signálové skupiny a nahrazeny bajtovým polem
- specifikace obsahuje automatickou detekci rychlosti přenosu

- rozšířený kontrolní součet obsahuje i chráněné pole identifikátorů
- identifikátory jsou definovány dynamicky a jednotlivé funkční významy identifikátorů jsou přidělovány masterem

2.4.2 Hlavní rozdíly mezi verzí LIN 2.0 a 2.1

- byl odstraněn identifikátor zprávy pro jednotku slave
- byl odstraněn rámec assign frame id a byl přidán assign frame id range
- byla přidána konfigurace pro uložení

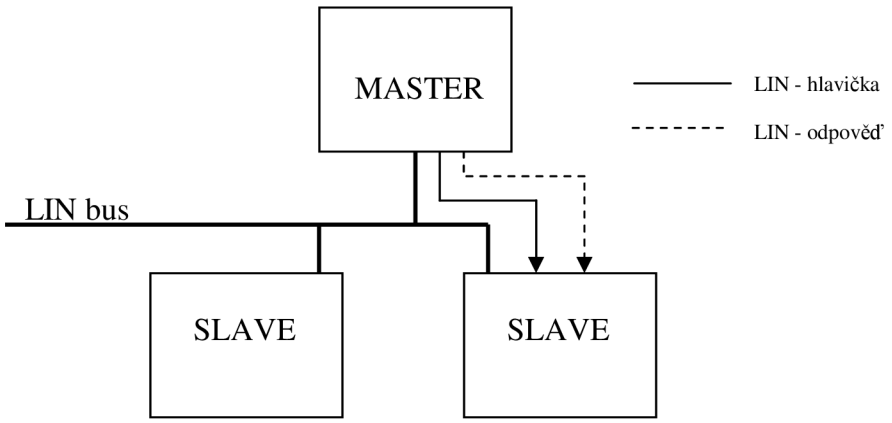
2.4.3 Kompatibilita verze LIN 2.1 s předchozími

Základní přenos zůstává zachován. Jednotka master verze 2.1 umí obsluhovat slave 2.1 i 1.3. Master pak může vyžadovat po slave verze 1.3 rozšířený kontrolní součet, detekci automatické volby rychlosti přenosu a diagnostické změny. Nemůžeme použít jednotku slave 2.1 ovládanou materem 1.3 z důvodů rozšířeného kontrolního součtu. Jednotka master verze 2.1 může ovládat jednotku slave 2.0, jestliže obsahuje funkce jako master 2.0, například funkci assign frame id.

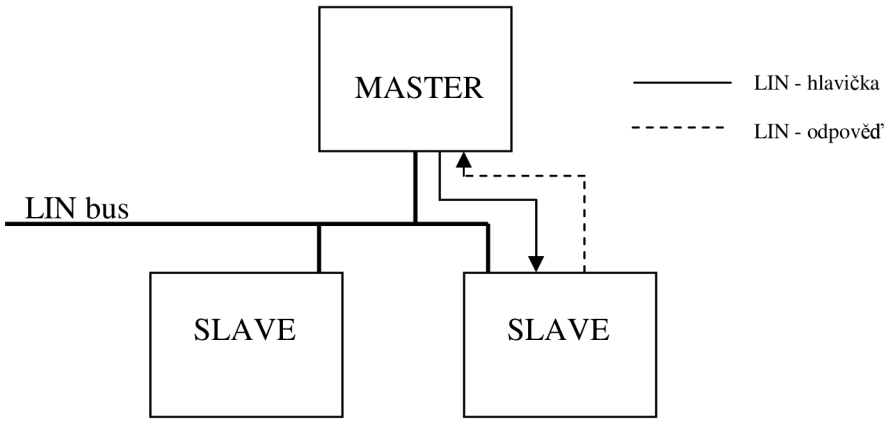
Fyzická vrstva u specifikace 2.1 je zpětně kompatibilní s verzí 1.3 i 2.0.

2.5 KOMUNIKACE PO SBĚRNICI LIN

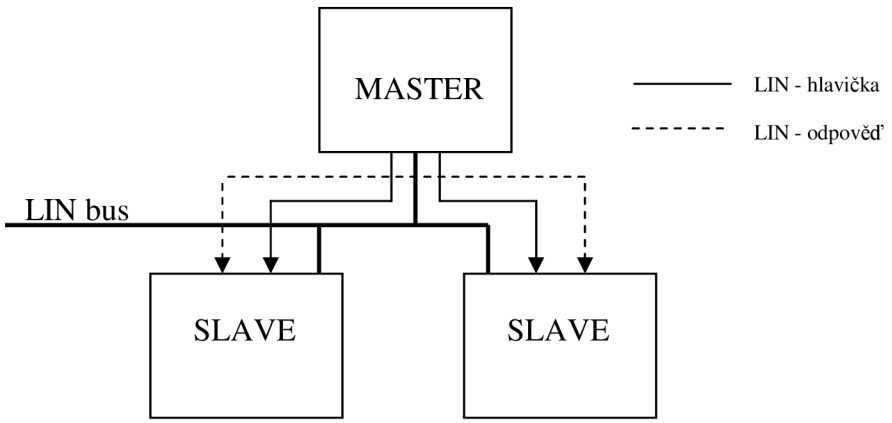
Na obrázku 6 až 8 jsou znázorněny komunikace master - slave, slave - master a slave - slave. Při komunikaci master - slave posílá master hlavičku i odpověď zařízení slave (obrázek 6). Při komunikaci slave - master slave čeká na hlavičku od mastera a poté vysílá svoji odpověď (obrázek 7). Verze protokolu LIN 2.0 dovoluje, aby spolu komunikovali jednotky slave mezi sebou, aniž by data procházela přes master. Master pouze posílá hlavičky zprávy pro inicializaci přenosu dat (obrázek 8).



Obr. 6 Komunikace master -> slave



Obr. 7 Komunikace slave -> master



Obr. 8 Komunikace slave -> slave

Nadřízená úloha je vysílána masterem a ten vytváří hlavičku, kde je nadefinován typ zprávy a příjemce. Jednotlivé podřazené úlohy se odehrávají v zařízení slave. Když jednotka slave obdrží hlavičku, zpracuje ji a má možnost odpovědět, nebo vykonat činnost požadovanou masterem.

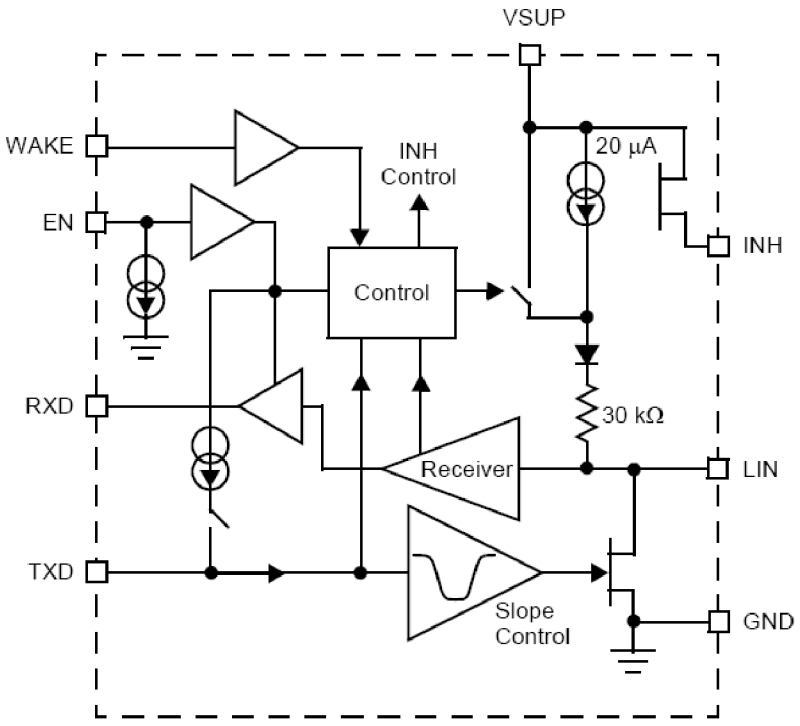
2.5.1 Jednotlivé funkce zařízení master a slave:

Master:
- vysílá hlavičku – definuje rychlost, synchronizuje ostatní jednotky, vytváří identifikátory
- monitoruje a potvrzuje data pomocí kontrolních součtů
- přepíná podřízené jednotky do úsporných sleep módů a probouzí je
- reaguje na probuzení ze sleep módů

Slave:
- čeká na synchronizační impuls
- synchronizuje se podle synchronizačních pulzů
- podle identifikátoru provádí:
  - čekání
  - vysílání
  - přijímání

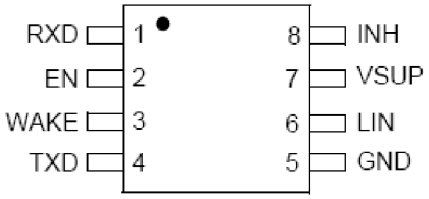
3. OBVODY FYZICKÉHO INTERFACE LIN

Pro realizaci jsem si vybral obvod od firmy Freescale MC33661. Tento obvod je fyzický interface sběrnice LIN verze 2.0. Je optimalizován pro rychlosti 10 kbps a 20 kbps. Je možnost využít i rychlosti 100 kbps pro testovací a programovací módy. Tento obvod má vstupní obvody optimalizované pro maximální rozsah napětí -18 V až 40 V, ale typické napětí je 6 – 18 V. Obvod podporuje sleep režim, při kterém odebírá pouhých 8 µA. Uvnitř obvodu je vnitřní odpor 30 kΩ, takže na vstup zařízení typu slave není potřeba zapojovat externí odpor. Pouze pokud chceme využít zařízení jako master, musíme připojit externí odpor 1 kΩ. Je zde také oddělení driver/reciver, a proto je kompatibilní se vstupy 5 V a 3,3 V. Blokový diagram vnitřního zapojení je na obrázku 9.



Obr. 9 Zjednodušené vnitřní schéma obvodu MC 33661

Na obrázku 10 je zobrazené pouzdro MC33661 a jednotlivé vývody.



Obr. 10 Označení vývodů u pouzdra obvodu MC33661

Vývod VSUP (napájení)

Napájení obvodu je chráněno diodou proti otočení polarity napájení. Nesmí být přerušeno připojení na zem - signál GND. Při přerušení napájení není obvod schopen reagovat na stav recessive (logická jednička) na sběrnici.

Vývod LIN

Přes vývod LIN se připojuje k obvodu jednovodičová sběrnice LIN. Jak bylo již řečeno, obvod podporuje rychlosti 20 kbps (normal slew rate) a 10 kbps (slow slew rate). Pomalejší rychlost 10 kbps je vhodné využít tam, kde nepožadujeme velkou rychlost a tím zmenšíme elektromagnetické rušení vysílané do okolí.

Vývod TXD (datový vstup)

Tento vstup slouží k posílání dat na sběrnici. Když je na vstupu TXD logická nula, tak výstup LIN je také logická nula, a když je signál TXD v logické jedničce, tak je výstup LIN vypnut. Rychlost přenosu (normal nebo slow slew rate) se nastavuje pomocí vstupu EN.

Vývod RXD (datový výstup)

Tento vývod slouží ke čtení dat ze sběrnice. Logická jednička na výstupu RXD odpovídá logické jedničce na sběrnici LIN (recessive) a logická nula na výstupu odpovídá logické nule na sběrnici (dominant). Logická nula je fixní, ale logická jednička je podle napětí na EN 3,3 V nebo 5 V. V režimu sleep je výstup RXD přepnut do režimu vysoké impedance, takže obvod nezatěžuje sběrnici. Jakmile přijde signál na probuzení (Wake up), výstup přejde do logické nuly.

Vývod EN (enable input terminal)

Přes tento vstup se nastavuje interface a definuje se pracovní napětí výstupu (3,3 V nebo 5 V). Je-li vstup v jedničce (normal mode), výstup LIN je na pinu RXD a vstup je TXD, a oba jsou aktivní. Napětí logické jedničky definuje napětí pro RXD. Sleep režim se aktivuje tím, že EN = LOW (logická nula) a RXD = HIGH (logická jednička).

Vývod INH (inhibit output terminal)

Tento výstup má dvě funkce. Může být použitý jako externí přepínatelný napěťový regulátor, nebo je zde připojen externí rezistor na sběrnici v režimu master.

Vývod WAKE (wake up)

Tento vstup slouží k probouzení zařízení ze sleep módu. Většinou se ovládá přes externí přepínač a napětí na vstupu je poloviční jako napájecí napětí. Tento vstup by neměl být nezapojen. Pro zakázání sleep módu se tento signál připojuje na zem.

3.1.1 Provozní a přechodné módy obvodu MC33661

Provozní režimy jsou dva – normální mód a sleep režim. V normálním režimu lze ovlivnit elektromagnetické rušení pomocí rychlosti přenosu na LIN výstupu normal a slow mode. Přechodné režimy jsou také dva. Probouzecí režim (awake mode), který probouzí zařízení do normálního režimu, a pomalý čekací režim (wait slow mode), který nastává před přechodem do pomalého normálního režimu.

Normální mód (normal mode)

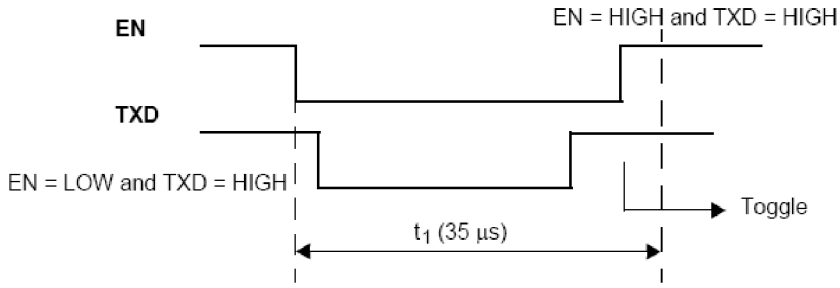
V tomto režimu je obvod kompatibilní se specifikací protokolu LIN a je schopný provozovat přenos dat po sběrnici s rychlostí 1 až 20 kbps. Režim nastává po sleep režimu nastavením TXD na logickou jedničku a přepnutí na vstupu EN z logické nuly do jedničky. Jestliže chceme přejít do pomalého módu, musíme nejdříve nastavit zařízení do režimu sleep. Normální mód má tři režimy rychlosti – pomalou, normální a rychlou (slow, normal a fast).

Slow mode

Jak již bylo řečeno, rychlost přenosu se pohybuje od 1 kbps do 10 kbps a zmenšuje rušení oproti normální rychlosti. Tento režim se nastaví, když po sleep režimu je na vstupu TXD logická nula a EN je nastaven z logické nuly do jedničky. Přechod do normální rychlosti se opět musí provádět přepnutím do sleep režimu.

Fast mode

Tento režim je až desetkrát rychlejší než maximální rychlost standartu LIN. Dovoluje komunikovat s rychlostí větší než 100 kbps a je určen pro testování elektronické ovládací jednotky (ECU) a stahování programu pro mikrokontrolér. V tomto režimu je možné použít přídavný rezistor, aby přizpůsobil časovou konstantu vnějšího RC obvodu přenosové rychlosti rychlého režimu. Tento režim může být vyvolán z režimu normální i pomalé rychlosti. Je vyvolán speciální sekvencí (toggle function), ukázanou na obrázku 11.



Obr. 11 Přepínací funkce pro nastavení fast režimu

Jestliže proběhne nastavení vstupů do kombinace podle obrázku 12 v čase maximálně 35 µs, komunikace se přepne do rychlého režimu. Tento fast režim se ukončí buď provedením přepínací funkce, nebo krátkým nastavením vstupu EN do logické nuly po dobu menší než 5 µs. Po skončení rychlého režimu se komunikace vrátí do původního režimu, ze kterého byl rychlý režim vyvolán, bez nutnosti přepnout jednotku do režimu sleep.

Sleep režim

V tomto režimu obvod nekomunikuje po sběrnici LIN a je přepnut do úsporného. Probuzení se provádí pomocí signálu EN a WAKE. Obvod MC33661 obsahuje vnitřní zdroj proudu 20 μA, který chrání obvod před proudovým přetížením.

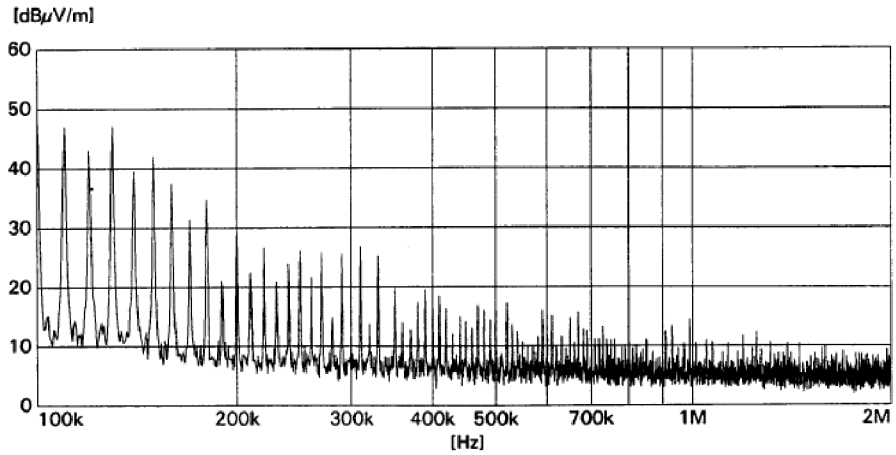
Probouzení (awake mode)

Při narůstání proudového odběru se obvod automaticky přepne do přechodného módu probouzení. Obvod nastaví INH do logické jedničky a RXD do logické nuly. Poté se mikrokontrolér nastaví do rychlostního stavu normal nebo slow, podle nastavených vstupů TXD a EN. Zařízení se může probudit třemi událostmi:

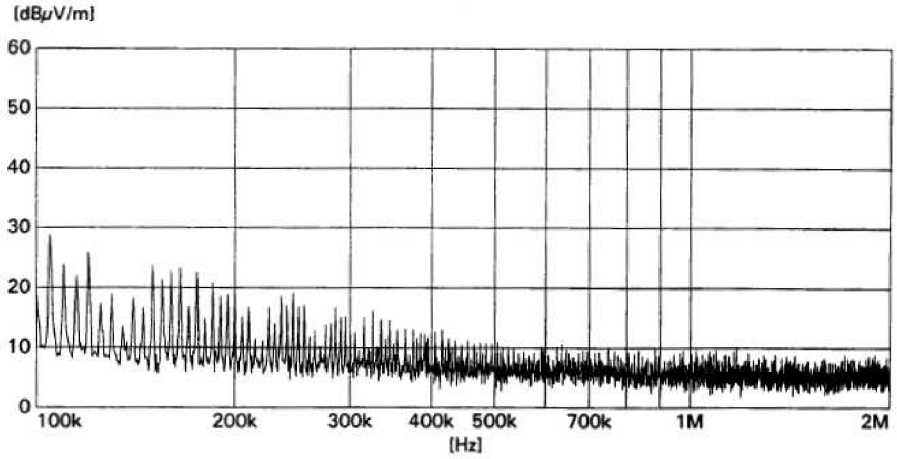
- probuzení po sběrnici LIN
- vnitřní probuzení přes vstup EN
- probuzení signálem WAKE

3.1.2 Elektromagnetická kompatibilita

Sběrnice LIN se používá v průmyslu spolu s dalšími standarty, a proto je velmi důležité, aby jedna komunikace nerušila druhou. Samotný standart LIN byl navržen pro malé vyzařované elektromagnetické rušení. Proto i obvod MC33661 je testován pro elektromagnetickou kompatibilitu. Na následujících dvou obrázcích 12 a 13 můžeme vidět rozdíl v amplitudě vyzařovaného rušení pro použití normální a pomalé rychlosti.

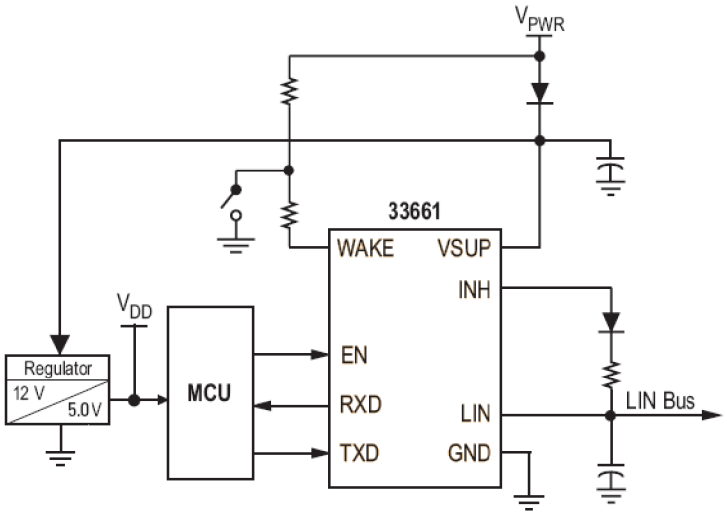


Obr. 12 Elektromagnetické rušení při normální rychlosti přenosu



Obr. 13 Elektromagnetické rušení při pomalé rychlosti přenosu

Při praktické realizaci sběrnice LIN pomocí obvodu MC33661 je zobrazeno blokové zapojení na obrázku 14.



Obr. 14 Typické zapojení obvodu MC33661

3.2 KONCEPCE ZAŘÍZENÍ SLAVE

Při realizaci můžeme vybírat obvody ze širokého sortimentu, které jednotlivé firmy nabízí. Můžeme vybrat obvod, který vytváří fyzický interface LIN, a k němu vybereme mikroprocesor. Nebo využijeme mikroprocesory, které obsahují podporu interface LIN na čipu. V tabulce 3 je výčet některých produktů jednotlivých firem, které vytváří interface LIN.

Výrobce	Typ	Popis
Freescale	MC33661	LIN interface 2.0
	MC33399	LIN interface 1.3
Microchip	MCP201	LIN interface + napěťový regulátor
Texas Instruments	TPIC1021	LIN interface 2.0

Tab. 3 Přehled výrobců a obvodů interface LIN

Všechny tyto obvody mají obdobné vlastnosti, a proto na výběru obvodu nezáleží, jelikož jsou obvody od různých výrobců mezi sebou kompatibilní.

Procesor vybíráme vzhledem k danému využití v aplikaci. Řady mikroprocesorů od předních výrobců, které je vhodné použít pro LIN, jsou následující:

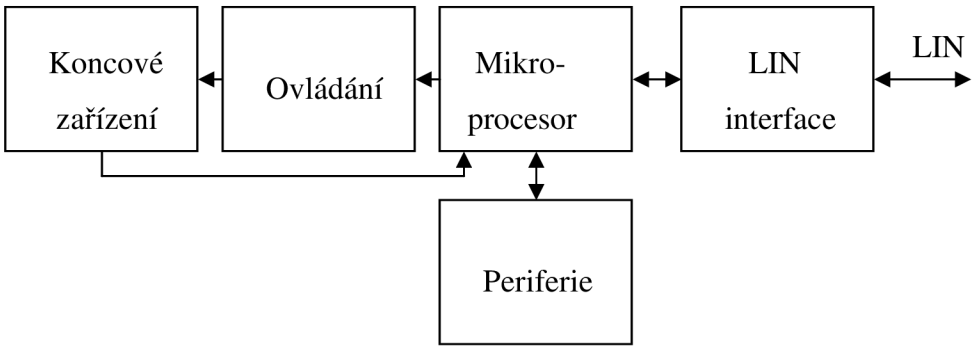
- Microchip PIC16x
- Freescale HC12, HCS12X, HCS08
- Texas Instruments TMS470

Jednotlivé modely mají široký výběr periferií integrovaných na čipu a tím vytvářejí velké kombinace využití. Základní požadavek na procesor je sériová linka optimalizovaná pro LIN. Většina mikroprocesorů určená pro automobilový průmysl obsahují interface CAN a některé i interface LIN (například procesor PIC16C432 od firmy Microchip). Pro jednotku master volíme procesory rychlejší, které zvládnou ovládat komunikaci po sběrnici. Proto použijeme procesor šestnáctibitový a ne pomalý osmibitový. Jednotka master je náročnější, a proto můžeme vyžadovat rozšíření vnitřní paměť implementovanou na čipu o paměť externí. Procesor pro jednotku slave vybíráme dle činnosti zařízení. Většinou požadujeme integrovaný D/A převodník a další periferie nutné pro komunikaci s koncovým zařízením, ovládaném prostřednictvím sběrnice LIN. U jednotek slave nevyžadujeme rychlost, a proto je možné v některých aplikacích použít osmibitové mikrokontroléry. Příklad typů mikrokontrolérů od firmy Freescale řady HCS12 pro jednotku slave je zobrazen v tabulce 4.

LIN Slave MCUs				
Device	ROM	FLASH	RAM	Features
68HC908JK3	–	4K	128	Timer, PWM, ATD
68HC908JL3	–	4K	128	Timer, PWM, ATD
68HC908JK1	–	1.5K	128	Timer, PWM, ATD
68HC08AB16	16K	–	512	Timer, PWM, ATD, SCI, SPI
68HC908EY8	–	8K	256	Timers, ATD, SPI, Enhanced SCI

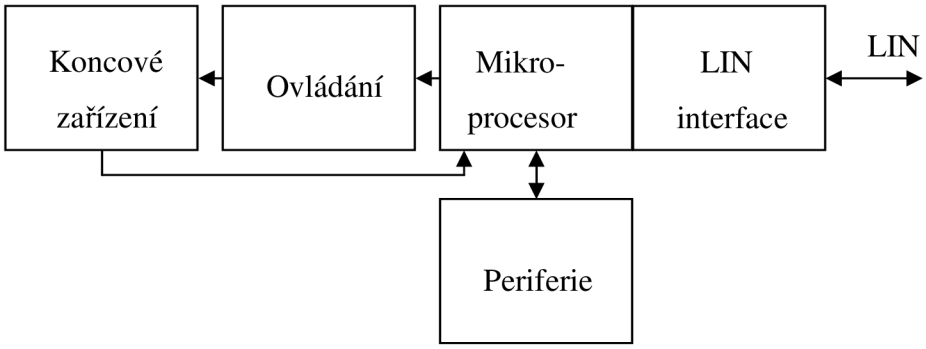
Tab. 4 Příklad obvodů pro jednotku slave řady HCS12

Návrh slave zařízení se odvíjí od použitého procesoru, jeho periférií a propojení výstupů pro ovládání koncového zařízení. Obecné slave zařízení plní ovládací funkci v řídícím procesu, kdy zpracovává informace získané od mastera, a podle nich ovládá koncové zařízení. Nejčastěji slave zařízení obsahuje libovolný procesor, který je doplněn o externí obvod LIN transceiver. Blokové schéma návrhu takového zařízení je zobrazeno na obrázku 15.



Obr. 15 Blokové schéma slave zařízení s procesorem a LIN transceiverem

Můžeme se setkat i se zařízením slave s procesorem obsahující interface LIN na čipu. Potom je blokové schéma zařízení slave zařízení jako na obrázku 16.



Obr. 16 Blokové schéma slave zařízení s procesorem obsahující LIN transceiver

4. MIKROPROCESOR HCS12

Tento procesor je šestnáctibitový, akumulátorově orientovaný a má dva osmibitové akumulátory A a B, které dokáže spojit v jeden šestnáctibitový akumulátor D. I adresové registry IX a IY jsou šestnáctibitové, takže procesor může adresovat více paměti a pracovat s většími čísly v akumulátoru. Procesor jsem vybral pro testovací účely zařízení master. Použil jsem k tomu školní přípravek s procesorem MC9S12DP256B.

Procesor je v 112 pinovém pouzdře a přípravek má vyvedené všechny výstupy z procesoru. Obsahuje také externí oscilátor 16 MHz. Procesor disponuje 256 kB flash paměti pro program, 12 kB RAM a 4 kB EEPROM. Procesor obsahuje široké možnosti v připojení – 5 linek CAN 2.0 A, B, 2 linky SCI, 3 linky SPI, linku IIC a deseti bitový převodník A/D pro zpracování analogových signálů.

Procesor jsem programoval přes BDM režim (background debug mode). Tento režim dovolí nahlížet na proměnné v procesu na běžícím procesoru. Režim je velice výhodné používat pro ladění různých aplikací, kdy programátor má přehled a kontrolu nad proměnnými v programu a tím eliminuje nežádoucí stavy programu. Použil jsem programátor firmy P&E Multilink, který byl připojen přes paralelní port LPT a má přímou podporu z programu CodeWarrior verze 5.5.1272.

5. MIKROPROCESOR HCS08

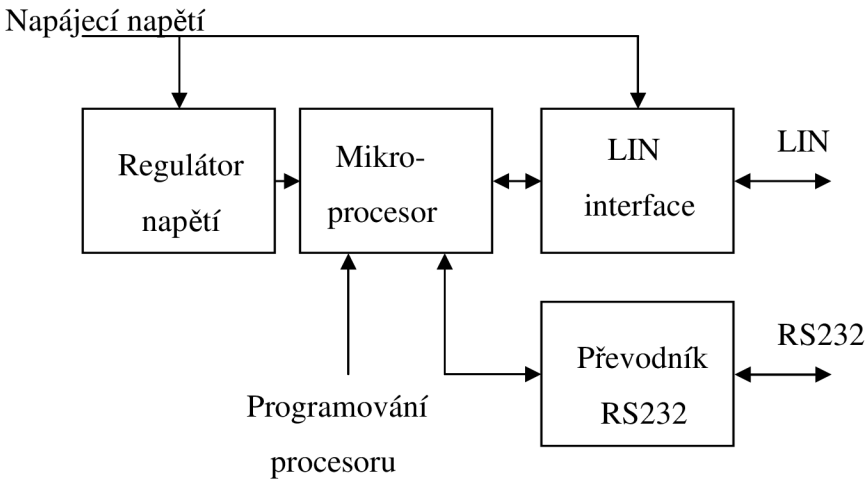
Pro slave zařízení jsem vybral jednoduchý osmibitový procesor z rodiny HCS08. Pro jednoduché úkony slave zařízení je procesor HCS08 dostačující a není nutné jednoduchou aplikaci řídit 16-ti bitovým procesorem řady HCS12. Procesor řady HCS08 obsahuje jeden osmibitový akumulátor pro výpočetní operace a šestnáctibitový adresový registr složený ze dvou osmibitových dílčích adresových registrů X a Y.

Firma Freescale se věnuje automobilovému průmyslu a nabízí několik řad, které obsahují periferie využitelné právě v tomto průmyslu. Vybral jsem si řadu mikroprocesorů DZ, konkrétně model MC9S08DZ60. Použitý procesor je v 48 pinovém pouzdře, obsahuje dvě sériové linky a podporu sériové linky pro LIN 2.0. Podpora znamená, že linka má programovatelnou rychlost přenosu, přerušení pro detekci 13 bitového breaku a přerušení pro detekci náběžné nebo sestupné hrany. Procesor obsahuje interní oscilátor až 16 MHz (externí až 40 MHz) a disponuje 60kB pamětí a 4 kB pamětí RAM. Dále obsahuje 12 bitový 24 kanálový převodník a podporu sběrnice CAN, SPI i IIC. Procesor jsem programoval přes režim BDM programátorem P&E Multilink, připojený přes USB z prostředí CodeWariror verze 5.9.0 pro HCS08 procesory.

6. REALIZACE HARDWARE

6.1 SLAVE

Koncepce zařízení vychází z blokového schématu zobrazeného na obrázku 15. Realizoval jsem testovací zařízení, které neovládá žádné koncové zařízení, ale pouze monitoruje LIN komunikaci a podává zprávu o jejím průběhu na linku RS232. Blokové schéma realizovaného zařízení je zobrazeno na obrázku 17.



Obr. 17 Blokové schéma realizovaného testovacího slave zařízení

6.1.1 Regulátor napětí

Jako stabilizovaný zdroj byl pro zařízení slave použit obvod 7805. Tento obvod dovoluje připojit na vstup napětí až 35V. Toto vstupní napětí je také využito jako napájení LIN transceiveru, který dovoluje maximálně 40V. Celkové maximální dovolené napájecí napětí pro slave zařízení je 35V.

Výstupní napětí regulátoru je 5V při libovolném vstupním napětí po maximální dovolené napětí.

6.1.2 Mikroprocesor

Použitý mikroprocesor je MC9S08DZ60. Procesor disponuje 60 kB pamětí programu, což je pro jednoduchou obsluhu slave zařízení dostačující. Procesor obsahuje interní oscilátor o maximální frekvenci 16 MHz, který používá teplotní stabilizaci. Tím je zaručen přesný kmitočet a není potřeba zařízení osazovat externím krystalem.

6.1.3 Lin transceiver

Tuto funkci u realizovaného slave zařízení zajišťuje obvod MC33661. Obvod je připojen na SCI1 procesoru přes vývody RxD a TxD. Na bránu F procesoru je připojen vstup EN, který povoluje LIN transceiver.

6.1.4 Převodník RS232

Testovací zařízení obsahuje připojení na linku RS232 a je připojeno na linku SCI2 procesoru. Po přijetí LIN zprávy zařízení na tuto linku vysílá informace o přenosu. Pomocí této linky lze také procesor ovládat a tím ladit slave zařízení. Převodník napěťových úrovní z logických úrovní 0 – 5 V na úrovně sběrnice RS232 je realizován obvodem MAX232. Tento obvod obsahuje vstupy a výstupy pro dvě nezávislé sériové linky. Obsahuje napěťovou pumpu realizovanou externími kondenzátory, která vytváří napětí 10V.

6.1.5 Návrh desky plošných spojů

Návrh schématu a desky plošných spojů pro testovací slave zařízení byl vytvořen v programu EAGLE. Příloha A obsahuje schéma zapojení, příloha B zobrazuje vrchní stranu desky a příloha C spodní stranu desky. Příloha D obsahuje rozložení součástek na desce plošných spojů.

Zařízení obsahuje přepínač LIN slave switch, který po propojení vytváří z slave zařízení. Jestliže přepínač není propojen, je na desce místo pro odpor 1 kΩ a diodu, která definuje obvod MC33661 jako master.

6.2 MASTER

Pro testovací účely jsem postavil zařízení typu master. Toto zařízení je založeno na procesoru HCS12. Na výstup sériové linky SCI0 jsem připojil obvod pro fyzický interface LIN MC33661, který obsahuje externí odpor a diodu pro definování masteru.

7. IMPLEMENTACE SOFTWARE

7.1 SLAVE

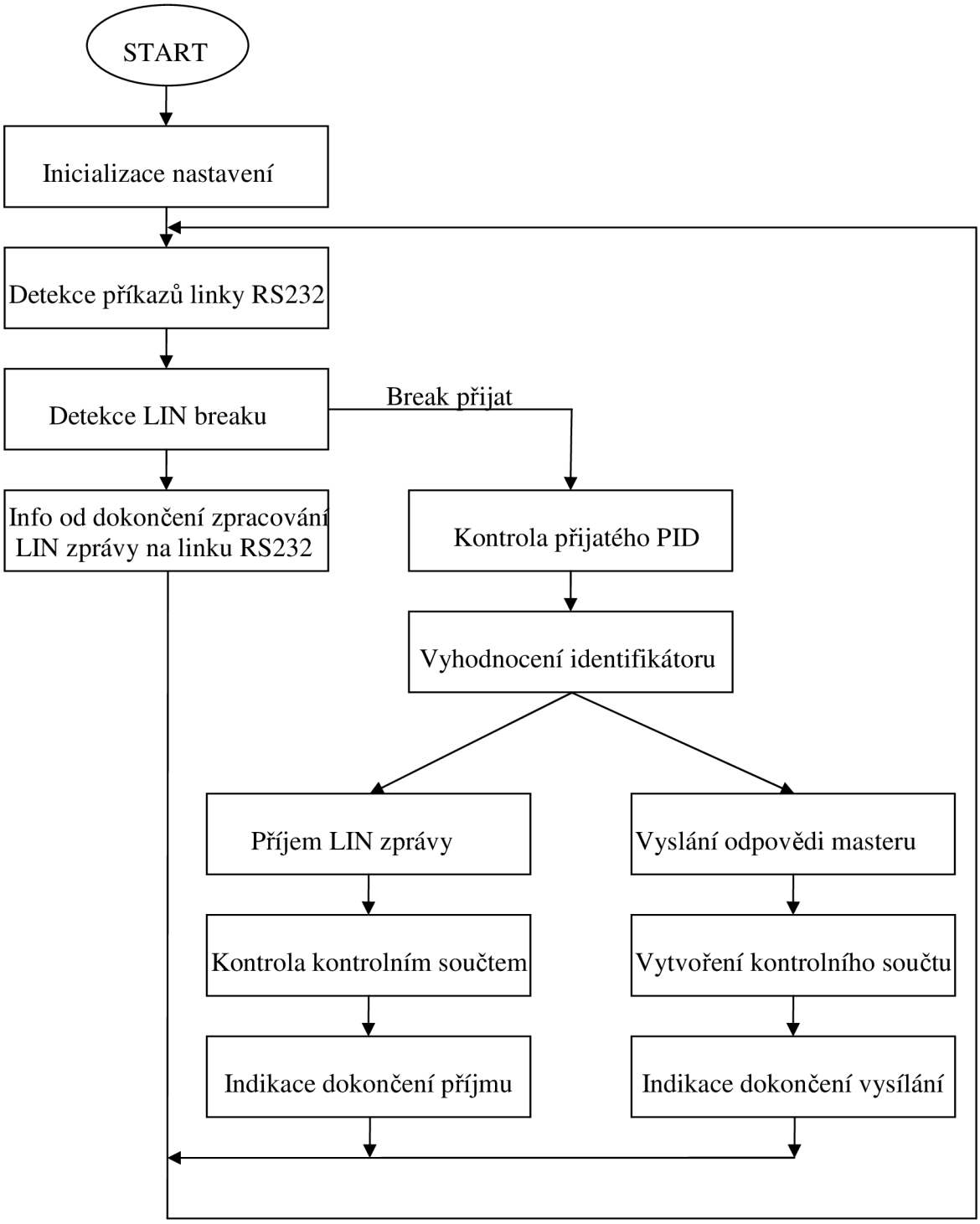
7.1.1 Funkce testovacího slave zařízení

Zařízení neustále čeká na příchod dat na obou linkách SCI. Přijde-li nějaký požadavek z počítače přes linku RS232, procesor jej zpracuje. Tím se dá testovací zařízení ovládat přes linku RS232. Při příchodu zprávy na LIN sběrnici zařízení detekuje break a zkontroluje, zda přišel synchronizační puls 0x55. Zařízení dále přijme a zkontroluje PID. Poté podle identifikátoru zařízení provede požadovanou akci. Po dokončení zpracování identifikátoru zařízení pošle na linku RS232 zprávu o průběhu zpracování zprávy. Při příjmu zprávy zařízení vypíše [PID] [LIN data] [kontrolní součet] [O;B], kde O znamená přenos v pořádku (OK) a B je znakem pro chybu (bad). Při odpovědi zařízení slave masterovi se na linku RS232 pošle zpráva [“Data sent“] [PID] [kontrolní součet] [O;B]. V praxi by zařízení obsahovalo při chybě nějaké resetovací úkony, aby chyba byla odstraněna. Testovací zařízení je ale postaveno právě pro odlaďování chyb a není požadováno, aby se při každé chybě zařízení resetovalo.

Program je napsán v ANSI C v prostředí CodeWarrior verze 5.9.0. Toto prostředí mimo jiné obsahuje hlavičkové soubory všech procesorů řady HCS08. Hlavičkový soubor definuje registry periferií v procesoru.

Program obsahuje část volání hlavičkových souborů od výrobce, či uživatelsky definovaných. Následuje inicializační nastavení, například SCI, a povolení přerušení `EnableInterrupts;`. Na konci je nekonečná smyčka, kterou procesor vykonává stále dokola. V hlavní smyčce program vyčkává na příchod LIN zprávy. Smyčka musí obsahovat instrukci `__RESET_WATCHDOG();`. Watchdog je „hlídací pes“, který kontroluje, zdali se program nedostal do prázdné smyčky. Jestliže nedostane zprávu o běhu programu, provede restart a tím prázdnou smyčku procesor opustí. Proto za normálního běhu programu musíme potvrdit správný chod resetováním watchdogu.

7.1.2 Hlavní funkce



Obr 18 Stavový diagram slave zařízení

Hlavní funkce obsahuje tělo programu slave zařízení. Stavový diagram úloh je zobrazen na obrázku 18.

Na začátku hlavního programu se nachází inicializační nastavení. Následuje oživení LIN transceiveru posláním logické jedničky na signál EN `PTFD=0xFF;`. Poté program skončí v nekonečné smyčce, kde kontroluje jednotlivé příznaky. Příznak příchodu LIN právy je iniciován příchodem breaku a detekován proměnnou `breakdetect`. Program pak reaguje na příjem dat na SCI. Při příjmu znaku totiž dojde k nerovnosti mezi počítadlem zpracovaných bajtů a počítadlem přijatých bajtů. Proto je následující část uvnitř podmínky `if (progrxc!=bufrxc)`. Jestliže po detekci breaku přijde synchronizační bajt 0x55, pokračuje dále zpracování zprávy povolením proměnné `linenable`. Jak již bylo zmíněno, jednotlivé kroky se provádí při příchodu znaku. Proto se v této části na lince nachází PID, které se uloží na první pozici pole `msg`, které bude obsahovat kompletní LIN zprávu. Dále se přijatý PID zkontroluje pomocí funkce `chechpid`. Z přijatého PID se vymaskuje ID tak, že se vynulují první dva nejvýznamnější bity PID `id=pid&0x3F;`. Zpracovávání LIN zprávy v hlavní funkci končí zavoláním funkce pro zpracování ID, tedy `switchid`. Poslední příkaz při splnění podmínky o přijatém znaku je vytvoření kruhového bufferu o velikosti 32 `progrxc = (progrxc+1)%32;`.

Dále hlavní funkce obsahuje zpracování dokončení LIN zprávy. Při dokončení příjmu zprávy se nastaví `done` na hodnotu jedna. V tomto případě se pošle na linku RS232 diagnostická zpráva, kde se zobrazí data a indikace správnosti přenosu. Vypsaná data jsou ve formátu [PID] [LIN data] [kontrolní součet] [O;B]. Na konci se vypíše O (OK) při správném přenosu bez příznaku `errorflag`. Jestliže se ale v průběhu zpracování nastaví do jedničky, pak se na konci vypisovaného pole objeví B (bad). V případě, že je dokončen výpis na sběrnici LIN, nastaví se `done` na hodnotu dva. Na linku RS232 se pak vypíše údaj, že data byla odeslána „Data sent“. K tomu se připojí informace o přenosu ve tvaru [PID] [kontrolní součet] [O;B]. Po jakémkoli dokončení (`done` je rovno 1 nebo 2) je potřeba uvést program pro detekci breaku do původního stavu. Proto se zpátky povolí detekce breaku instrukcemi `SCI1S2_LBKDE=1;`

a `SCI1BDH_LBKDIE=1;`. Tím je program znovu připraven pro detekování zprávy od mastera.

Program umožňuje komunikaci také přes rozhraní SCI2 resp. na něj připojenou sběrnici RS232. To lze využít pro ladění a testování zařízení. Hlavní funkce obsahuje podmínku podobnou pro příchod znaku, jako je tomu u SCI1 `if (progrxccom!=bufrxccom)`. Následuje opět definování kruhového bufferu o velikosti 32 `(progrxccom+1)%32;`. Uvnitř podmínky lze přiřazovat různým vstupním datům na RS232 jednotlivé akce procesoru. Naprogramování funkce například pro znak „a“ je uvedena `if (bufferrxcom[progrxccom]=='a')` a poté následuje libovolná funkce, kterou procesor vykoná vždy, když z počítače přes COM port přijde znak „a“. Pokud například potřebujeme otestovat sběrnici LIN tak, že master správně nevysílá break, synchronizační puls či PID, můžeme to nahradit. Funkci pro „a“ naprogramujeme `id=0x3D; switchid();` a procesor pak nebude ctít LIN zprávu a vyšle bez dotázání data. Tím se dá například otestovat problém hardwarové propojení, když od mastera jsou přijatá data nečitelná.

Inicializace procesoru

Procesor využívá interní oscilátor. Oscilátor má mnoho nastavení, různé variace a kombinace pro pestré využití procesoru. Při implicitním nastavení oscilátoru je frekvence na sběrnici poloviční frekvenci oscilátoru. Interní oscilátor je nastaven instrukcí `MCGTRM=0xA6;` na 16 MHz.

Slave zařízení používá pro komunikaci dvě sériové linky. Na SCI1 je připojen LIN transceiver a je tedy využita pro komunikaci po sběrnici LIN. SCI2 je vyvedeno na obvod MAX232 a přes tuto linku komunikuje mikroprocesor s počítačem přes COM port.

`SCI1C1=0;` - nastavení rámce zprávy na SCI
- obsahuje jeden start a stop bit, 8 datových bajtů a žádnou paritu

`SCI1C2=0x6C;` - povolení funkce přijímače a vysílače (RE,TIE) a povolení přerušení od vysílače i přijímače (RIE, TIE)

`SCI1BDL=52;` - nastavení přenosové rychlosti 9600 baudů (bitů za sekundu) pro frekvenci 8 MHz na sběrnici
- výpočet je podle vzorce

$$BDL = \frac{f_{BUS}}{baudrate \times 16} = \frac{8 \times 10^6}{9600 \times 16} = 52 \qquad (2)$$

`SCI1BDH=0;` - vyšší bajt přenosové rychlosti není využit

Nastavení druhé sériové linky SCI2 je stejné, jako v případě SCI1, takže komunikace po RS232 je nastavena na 9600 baudů. Další nastavení se týká funkcí sběrnice LIN, konkrétně povolení detekce breaku. Break je potřeba nastavit na 13 bitů. Je to speciální signál, který neobsahuje start ani stop bit a trvá právě 13 bitů. K detekci je tedy potřeba využít nadstavby pro SCI od výrobce, protože 13 bitů v normálním režimu bez start a stop bitu nelze detekovat. Nastavení breaku na 13 bitů je instrukcí `SCI1S2_BRK13=1;`. Nejprve je potřeba povolit detekci breaku pomocí `SCI1S2_LBKDE=1;`. Protože příchod breaku je asynchronní akce, je potřeba obsluhovat tuto událost v přerušení. Povolení přerušení při detekci breaku je v rozšířené části nevyužitého vyššího bajtu pro nastavování rychlosti SCI nastaveno instrukcí `SCI1BDH_LBKDIE=1;`.

Obvod MC33661 obsahuje signál EN, který ovládá LIN transceiver. Pro normální komunikaci je potřeba, aby tento vývod byl v logické jedničce. Zařízení je navrženo tak, že EN výstup z procesoru je na portu F. Abychom mohli použít bránu jako výstup, je nejdříve nutné nastavit směr dat (data direction) – vstup či výstup. Je-li brána výstup, je potřeba nastavit bit do logické jedničky příkazem `PTFDD=0xFF;`. Poté můžeme bránu využít jako výstup a `PTFD=0xFF;` pošle na bránu F samé jedničky a tím i na vstup EN obvodu MC33661.

7.1.3 Hlavičkový soubor linapi.h

Tento hlavičkový soubor obsahuje deklarace funkcí, které využívá procesor ke komunikaci po sběrnici LIN. Jednotlivé funkce jsou definovány v souboru linapi.c a jsou volány z hlavní funkce programu.

Obsluha přerušení

Procesor MC9S08DZ60 má rozdělené přerušení od sériové linky do tří vektorů. Pro sériovou linku SCI1 je vektor 18 (0xFFDA, 0xFFDB), který se vyvolá při dokončeném vysílání. Vektor 17 (0xFFDC, 0xFFDD) zahrnuje události při příchodu znaku. Kromě detekce příjmu znaku na linku je tento vektor vyvolán při detekci breaku. Vektor 16 (0xFFDA, 0xFFDB) je vyvolán při datové chybě na SCI. Zde lze nastavit detekci chyb zprávy (framing error) nebo například chybu parity.

Vektor 17

Tento vektor slouží pro obsluhu při příjmu znaku. Funkce je definována instrukcí `void interrupt 17 sci1_rx(void)`. Zásada je při vstupu do obsluhy přerušení zakázat ostatní přerušení. Proto při vstupu do obslužné rutiny je první instrukce `SCI1C2=0;`. Při vyvolání přerušení se musí zjistit zdroj přerušení. Proto pomocí status registru SCI zjistím, zda je nastaven příznak přijetí znaku podmínkou `if((SCI1S1 & 0x20)!=0)`. Je-li tomu tak, naplním kruhový buffer bufferrx znakem, který přišel na linku.

Jak již bylo zmíněno, tento vektor přerušení je vyvolán i při detekci breaku. Tato akce se vyhodnotí podmínkou `if (SCI1S2_LBKDIF==1)`. Pokud je zdroj přerušení právě přijatý break, je nutné vynulovat flag zapsáním jedničky do něj. Pro detekci provedení této větve jsem si zvolil proměnnou `breakdetect`, která se nastaví do jedničky, a tato proměnná je dále vyhodnocována v programu. Poté se zakáže přerušení a detekce breaku.

Vektor 18

Vektor se vyvolá při vyprázdnění vysílacího bufferu SCI, tedy když jsou data odeslána. Struktura obslužné funkce je obdobná, jako v případě obsluhy přerušení pro přijatý znak. Po zakázání přerušení následuje vyhodnocení zdroje přerušení podmínkou `if((SCI1S1 & 0x80)!=0)`. Pokud je linka volná, vypíše se obsah kruhového bufferu. Koncová podmínka pro výpis je porovnání dvou počítadel – počet bajtů vypisovaného bufferu (`buftxc`) a počet dat připravených pro výpis z programu (`progtxc`). Při dokončení výpisu se nastaví indikace dokončení `txint=1;`. Podle této proměnné je na konci obsluhy přerušení rozhodnutí, jaká přerušení se zpátky povolí. Když byla data vypsána všechna (`txint==1`), povolí se pouze přerušení pro čtení z linky. Jestliže výpis na linku ještě nebyl dokončen (`txint==0`), povolí se přerušení pro čtení i zápis na SCI.

Funkce write

Tato funkce obsluhuje výpis znaků na linku. Do bufferu pro výpis se nakopírují data, která mají být vyslána. Funkce povolí přerušení a vymaže flag pro detekci dokončení přenosu dat. Funkce obsahuje časovou smyčku, která zabraňuje přepsáním dat, když ještě nejsou vypsána pomocí podmínky `do {} while (txint!=1);`.

Funkce checkpid

Úkol funkce je kontrolovat přijatý PID (Protect IDentificator), který je vysílán masterem a obsahuje identifikátor a kontrolní bity pro ochranu. Na začátku funkce se shiftováním (posunem bitech v registru) nastaví proměnné id0 – id5, p0 a p1 podle hodnot odpovídajících bitů. Po shiftování se vymaskuje poslední bit a instrukce potom vypadá takto: `id5=(pid>>5)&0x01;`. Podle vzorce (1) se vypočítají nové hodnoty p0new a p1new. S negací ve výpočtu p1new opět souvisí maskování čísla tak, aby se projevil v čísle pouze poslední bit. Na konci funkce je podmínka, která porovnává p0 a p1 přečtená z PID a p0new a p1new vypočítané. Když dojde k odlišnostem, nastaví se chybový flag `errorflag`.

Funkce checksum

Další důležitá funkce, která provádí kontrolu kontrolního součtu vygenerovaného masterem a tím kontroluje platnost přijatých dat. Kontrolní součet je rozšířený a je v něm zahrnut i PID. Součet je prosté sčítání všech vstupních dat a při příznaku carry (přetečení) se suma zvýší o jedničku. Pro detekci přetečení je vhodné použít proměnnou jako unsigned (bez znaménka). Tím dojde k navýšení proměnné o jeden bit, který se dá použít pro vyhodnocování přetečení. Jestliže dojde k přetečení a inkrementování sumy o jedničku, je potřeba číslo sumy vymaskovat na 8 bitů a vyšší bity vynulovat. Po dokončení součtu je vyhodnocení správnosti kontrolního součtu vytvořena pomocí negace. Princip spočívá v tom, že vypočítaný součet se zneguje, a přičte se k němu přijatý součet vygenerovaný masterem. Pokud totiž sečteme číslo s jeho negací, musíme dostat číslo 0xFF. Proto pokud součet takto vytvořených sum neodpovídá číslu 0xFF, je nastaven `errorflag`.

Funkce createchecksum

Když je zařízení slave požádáno masterem o vyslání dat, je potřeba vytvořit kontrolní součet. Funkce má stejné početní jádro jako funkce `checksum`. Výsledná suma se pak odešle na konci dat v odpovědi.

Funkce switchid

Hlavní jádro slave zařízení, kde jsou definovány jednotlivé identifikátory a k nim jsou přiřazeny funkce. Sestavené testovací slave zařízení obsahuje dva identifikátory – 0x3D pro odpověď masterovi a 0x3C pro příjem zprávy. Zde se dají definovat nové identifikátory zpracovávané slavem. Od verze LIN 2.0 je tento styl definování pevných identifikátorů nahrazen dynamickým přidělováním ID.

Zpracování identifikátoru ID 0x3C

Tento identifikátor znamená čtení dat z linky. Zpráva v poli `msg` již na první pozici obsahuje PID. Proto funkce pro tento identifikátor čte znaky z linky a zapisuje jej na další pozice od jeničky. Funkce obsahuje smyčku stejnou, jaká je v hlavní

funkci pro čtení dat z linky. Předem je známá délka vysílaného pole v proměnné `size`. Smyčka přečte jeden znak navíc za daty, a to je kontrolní součet. Ten se uloží na konec pole celé zprávy. Až smyčka tedy přijme všechny znaky včetně kontrolního součtu, jsou data ověřena funkcí `checksum`. Funkce vytvoří kontrolní součet z dat i PID a zkontroluje správnost výpočtu s přijatým kontrolním součtem, nacházejícím se na konci zprávy. Po dokončení kontroly se nastaví příznak `done=1;`, který slouží k detekci dokončení příjmu a kontroly dat. V hlavní části programu se pak tento příznak testuje. Příznak slouží ke komunikaci s uživatelem pomocí RS232 rozhraní, kam procesor pošle zprávu o přenosu LIN zprávy.

Zpracování identifikátoru ID 0x3D

Identifikátoru je přiřazen výpis dat na linku. Testovací zařízení vysílá zprávu „LIN test“. Buffer pro výpis na LIN se naplní daty z pole `msg`, opět posunutého o jedničku, protože na první pozici je přijatý PID. Následuje vyvolání funkce `createchecksum`, která vytvoří kontrolní součet pro vysílaná data včetně PID. Výsledek kontrolního součtu se pak uloží na poslední pozici a celé pole se vypíše na linku SCI a tím přes LIN transceiver na sběrnici LIN. Po dokončení akce následuje opět identifikace dokončení přenosu pomocí `done=2;`.

7.1.4 Hlavičkový soubor com.h

Hlavičkový soubor obsahuje deklarace funkcí pro vysílání znaků na SCI2 a tím přes obvod MAX232 na linku RS232. Funkce jsou definovány v souboru com.c a jsou založené na stejných principech, jako je tomu u funkcí pro výpis a čtení z linky v linapi.h. Hlavičkový soubor obsahuje funkce `writecom` a obsluhy přerušení. Funkce `writecom` je totožná s funkcí `write`, ale spolupracuje s obsluhou přerušení SCI2. Tato obsluha je také totožná s obsluhou přerušení pro SCI1, změnily se pouze vektory přerušení. Pro příjem znaků se vyvolá vektor 20 a při vyprázdnění bufferu pro data vypisované na linku se vyvolá vektor přerušení 21.

7.2 MASTER

Aby bylo možno naprogramovat a otestovat slave zařízení, sestavil jsem zařízení master. Master je realizován na procesoru řady HCS12 a naprogramován v prostředí CodeWarior v jazyce C.

Zařízení master je založeno na stejných hlavičkových souborech a funkcích, jako je zařízení slave.

7.2.1 Hlavičkový soubor linapi.h

Soubor obsahuje deklarace totožných funkcí jako slave zařízení: `write`, `checksum`, `createchecksum` a `switchid`. Rozdíl ve funkci `switchid` je pouze ten, že má prohozené významy identifikátorů, protože význam ID čtení pro slave znamená zápis dat pro mastera a obráceně.

Obsluha přerušení

Funkce je založená na stejném principu, jak již bylo popsáno u slave zařízení. Rozdíl je v tom, že procesor MC9S12DP256B má pro přerušení od SCI pouze jeden vektor, a to 20. Po zakázání ostatních přerušení je podmínka pro detekci zdroje přerušení – zda jde o indikaci prázdného bufferu či přijatého znaku z linky. Podmínka `if((SCI0SR1 & 0x20)! =0)` obsluhuje příjem znaků a podmínka `else if((SCI0SR1 & 0x80)! =0)` obsluhuje vysílání znaků na linku. Úkol mastera je též vysílat break. V inicializačním nastavení v hlavní funkci je break nastaven na 13 bitů instrukcí `SCI0SR2 = 0x4;`. Vysílání breaku je nutné korektně ukončit a to je zajištěno právě v obsluze přerušení. Break je sice vysílání nuly na výstup, ale stejně jako vysílání dat je ukončen vyvoláním přerušení. To zajišťuje podmínka `else if (SCI0CR2_SBK==0)`. Jestliže tedy break byl vyslán a není žádoucí posílat další, vynuluje se příznak breaku zapsáním jedničky do něj `SCI0CR2_SBK=1;`. Funkce ještě nastaví proměnnou `brkend`, která je pak zkoumána v hlavní funkci a vytváří zámek pro nechtěný výpis dat na linku v průběhu výpisu breaku. Obsluha přerušení je pak korektně ukončena zpětným obnovením povolením správného přerušení, jako je tomu v případě slave zařízení.

Funkce createpid

Výhradní funkcí mastera je vytvářen PID, tedy chráněné pole identifikátoru. Funkce vychází z početního základu funkce `checkpid`, kde je potřeba přidat vypočítané dva bity do PID. Po výpočtu se bity shiftováním posunou na správnou pozici `p1=p1new<<7;`, `p0=p0new<<6;` a přidají se do PID proměnné `pid=p0|p1|id;`. Tím je PID vytvořen a může se vyslat slave zařízením.

Funkce createmsg

Stěžejní funkce mastera. Funkce vytváří LIN zprávu pomocí předchozích funkcí. Na začátku se vyšle break SCI0CR2_SBK=1;. Následuje časový zámek `do {} while (brkend=0);`, který chrání přepsání linky v době vysílání breaku. Na linku se vyšle synchronizační puls 0x55 a pomocí funkce `createpid` vytvořený PID. Na konci se zavolá `switchid`, která vykoná definované úkony podle zadaného identifikátoru `id`.

7.2.2 Hlavní funkce

Na začátku hlavní funkce se opět nachází inicializační nastavení, které nastavuje linku SCI0, na kterou je připojen LIN transceiver MC33661. Signál EN je připojen na bránu B, a proto je zde také nastaveno, že brána B je výstup `DDRB=0xFF;`. Hlavní nekonečná smyčka pak obsahuje vygenerování LIN zprávy. Pro vyslání zprávy je potřeba naplnit zprávu daty (pro testovací účely „12345678“), určit identifikátor `id=0x3C;` a zavolat funkci `createmsg();`. Program pak vygeneruje kompletní LIN zprávu včetně kontrolních dat.

8. ZÁVĚR

Cíl této práce bylo vytvořit testovací slave zařízení sběrnice LIN. Zařízení je realizováno na jednoduchém osmibitovém procesoru řady HCS08, typu MC9S08DZ60, jež je určen zejména pro automobilový průmysl. Zařízení je implementováno na desce plošných spojů. Vytvořil jsem programové vybavení pro zařízení slave v prostředí CodeWarrior verze 5.9.0.

Zařízení zpracovává LIN zprávu podle standartu 1.3. Po příjmu je zpráva poslána na linku RS232, kde obsluha obdrží informace o LIN zprávě – přijatý identifikátor, data, kontrolní součet a indikaci, že přenos proběhl správně.

Zařízení je schopno zpracovávat příkazy přes linku RS232. Testovací zařízení tak může vykonávat funkce, které si nevyžádá master. To umožňuje odladit potenciální problém na sběrnici LIN.

V budoucnu lze zařízení rozšířit o kompatibilitu s vyšším standardem LIN 2.1 a to například podporou automatické přenosové rychlosti, dynamického přidělování identifikátorů, či plnou podporu slep režimu.

9. SEZNAM POUŽITÉ LITERATURY

[1] *LIN Specification Package Revision 2.1*, © LIN Consortium, 2006 (dokument LIN-Spec_Pac2_1.pdf z www.lin-subbus.org (po registraci))

[2] *Local Area Network (LIN) Enhanced Physical Interface With Selectable Slew Rate*, Freescale Semiconductor, Rev 4.0 02/2005 (www.freescale.com/files/analog/doc/data_sheet/MC33661.pdf)

[3] *LIN Protocol Implementation Using PICmicro® MCUs,* Microchip Technology Inc., 2000 (http://ww1.microchip.com/downloads/en/AppNotes/00729a.pdf)

[4] *MC9S08DZ60 Data Sheet*, Freescale Semiconductor, Rev. 3, 10/2007 (www.freescale.com/files/microcontrollers/doc/data_sheet/MC9S08DZ60.pdf)

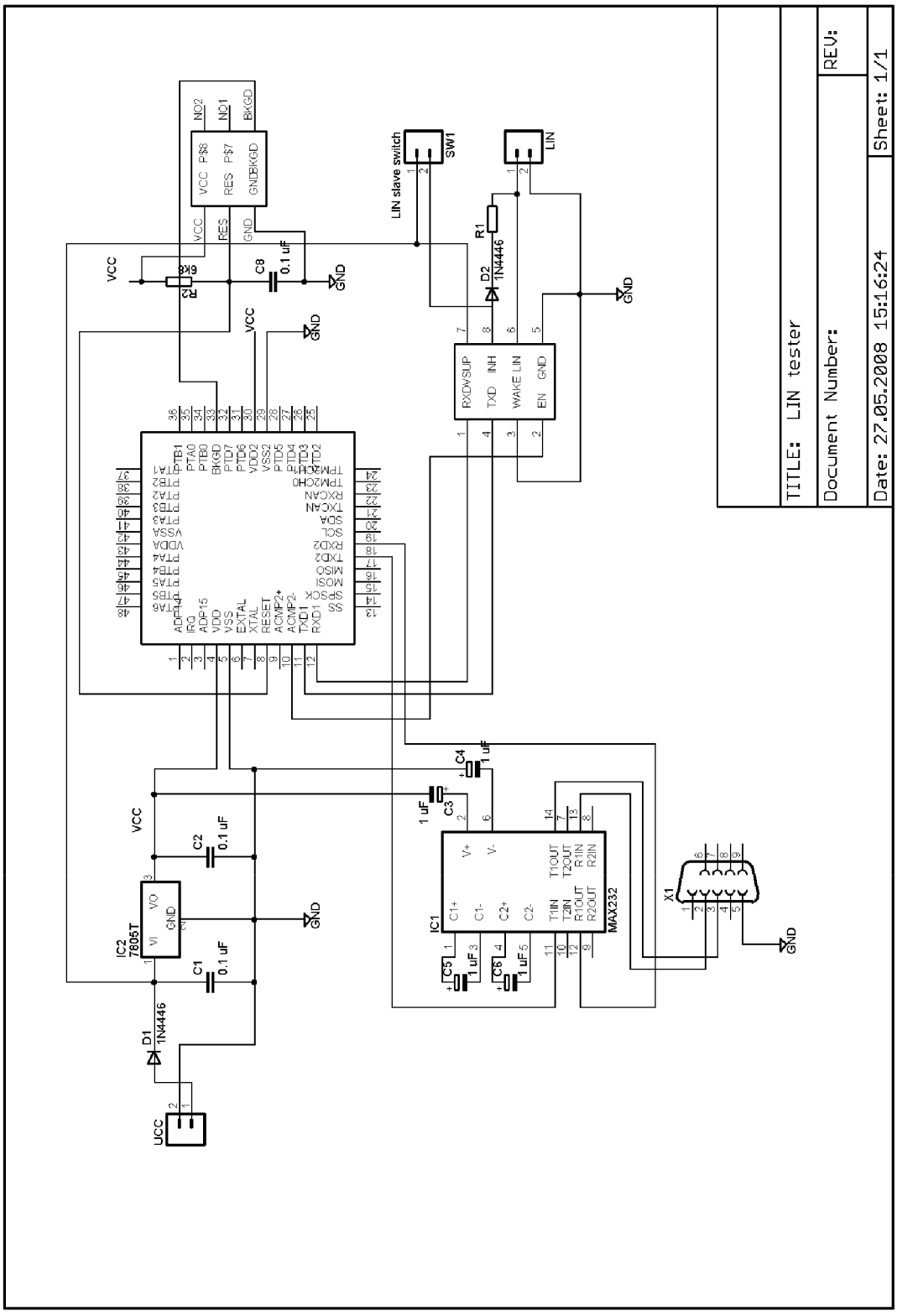
[5] *MC9S12DP256B Data sheet*, Freescale Semiconductor, Rev. 2.15, Jan 11, 2005 (www.freescale.com/files/microcontrollers/doc/data_sheet/9S12DP256BDGV2.pdf)

[6] *Local Interconnect Network (LIN) Demonstration*, Freescale Semiconductor, 2004 (www.freescale.com/files/microcontrollers/doc/app_note/AN2103.pdf)

[7] *MC68HC908EY16 Controlled Robot Using the LIN Bus*, Freescale Semiconductor, Rev. 1, 01/2007 (www.freescale.com/files/microcontrollers/doc/app_note/AN2470.pdf)

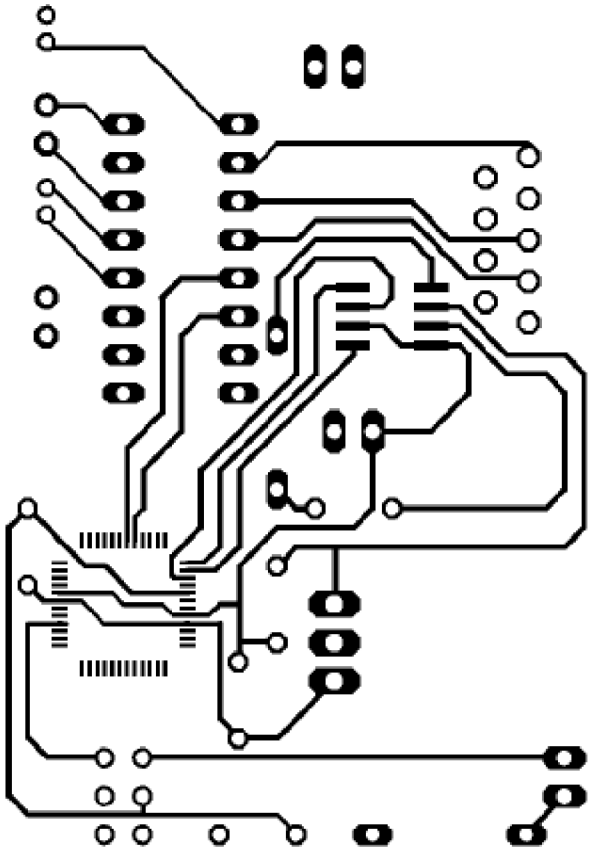
10. SEZNAM PŘÍLOH

Příloha A – Schéma zapojení realizovaného slave zařízení



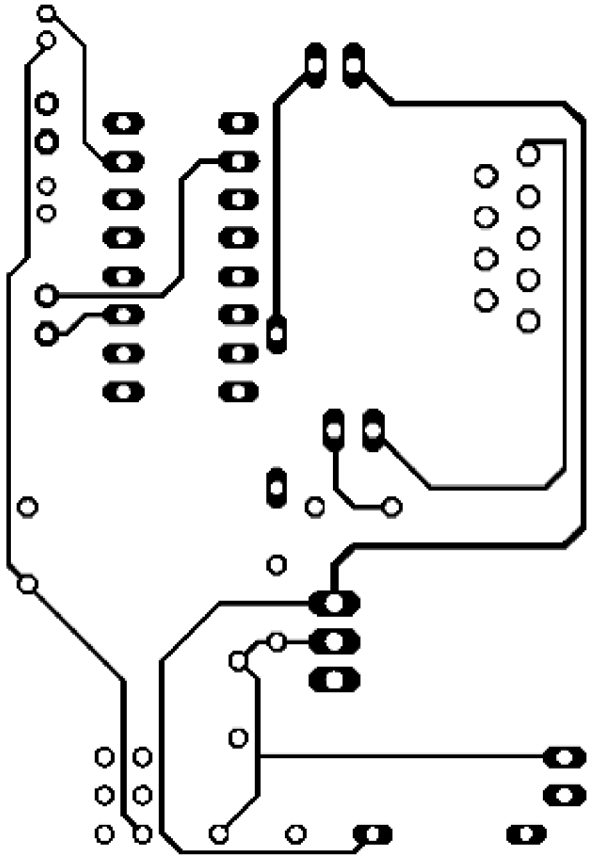
Příloha B – Návrh desky plošných spojů – strana součástek

v měřítku 2:1

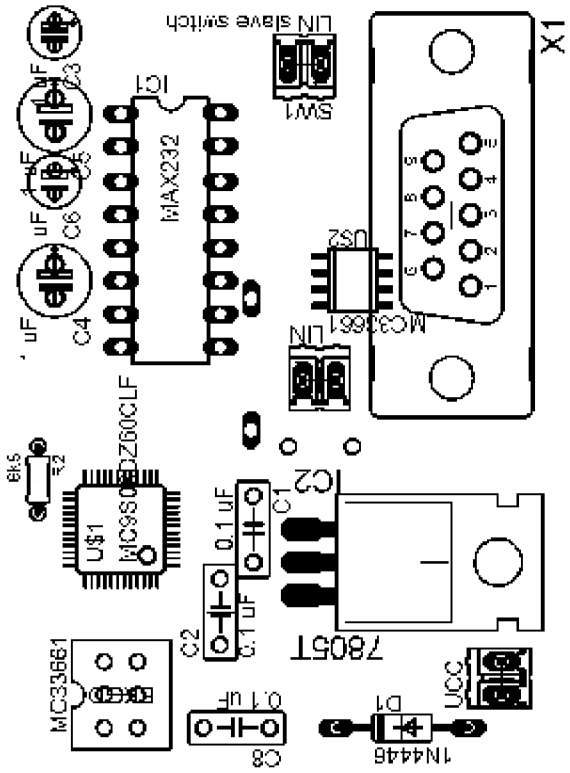


Příloha C – Návrh desky plošných spojů – spodní strana

v měřítku 2:1



Příloha D – Návrh desky plošných spojů – rozložení součástek v měřítku 2:1



Příloha E – Obsah přiloženého CD

CD:\BP prace

DYCKABP.pdf	- text bakalářské práce
ANOTACE.pdf	- text obsahuje abstrakt (CZ, ENG) a klíčová slova (CZ, ENG)

CD:\LIN software

\Master	- složka obsahuje projekt v prostředí CodeWarrior pro mastera
\Slave	- složka obsahuje projekt v prostředí CodeWarrior pro slave

CD:\Literatura

LIN-Spec_Pac2_1.pdf	- literatura [1]
MC33661.pdf	- literatura [2]
00729a.pdf	- literatura [3]
MC9S08DZ60.pdf	- literatura [4]
9S12DP256BDGV2.pdf	- literatura [5]
AN2103.pdf	- literatura [6]
AN2470.pdf	- literatura [7]

CD:\Prilohy

prilohaA.jpg	- schéma zapojení slave zařízení
prilohaB.jpg	- návrh desky plošných spojů – pohled od součástek
prilohaC.jpg	- návrh desky plošných spojů – pohled na spodní stranu
prilohaD.jpg	- rozvržení součástek na desce
\EAGLE	- složka obsahuje projekt návrhu desky