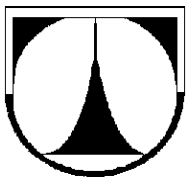


TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky a mezioborových inženýrských studií

Katedra softwarového inženýrství

DIPLOMOVÁ PRÁCE

Realizace uzlu s protokolem CAN

A node with CAN protocol realization

Liberec 2003 **Lubomír Novák**

Anotace:

Cílem diplomové práce bylo navrhnout uzel sběrnice CAN vybavený řídícím mikroprocesorem na bázi x51 a vybraným řadičem sběrnice CAN. Uzel měl být konfigurovatelný pomocí standardního PC vybaveného operačním systémem Microsoft Windows prostřednictvím sériové linky RS232. Zobrazování přijatých zpráv mělo být řešeno pomocí LED diod nebo LCD displeje. Uzel měl mít dále možnost snímat fyzikální veličiny pomocí napěťového AD převodníku.

Uzel byl realizován s řídícím mikroprocesorem Atmel AT89C52, řadičem sběrnice CAN Intel AN82527 a budičem této sběrnice Philips PCA82C250. Pro snímání hodnot napětí byl použit AD převodník Analog Devices AD7890-10. Řídící program mikrokontroléru uzlu byl napsán v jazyce C ve vývojovém prostředí µVision2 od firmy Keil Electronic. Aplikace pro nastavování vlastností uzlu byla vytvořena v prostředí Borland Delphi 5.0 od firmy Inprise. Pro ověření návrhu byl vytvořen program v jazyce C pro mikropočítač C167CR v prostředí µVision2.

Abstract:

The aim of this Diploma Thesis was to design a CAN bus node equipped with a controlling microprocessor based on x51 and a selected CAN bus controller. The node was required to be configurable by means of standard PC equipment of the operating system Microsoft Windows and the RS232 line. Notification of messages received was required to be solved by means of LEDs or an LCD display. Furthemore, the node was required to have the ability to scan physical quantities by means of an AD converter.

The node was realized with the controlling microprocessor Atmel AT89C52, CAN bus controller Intel AN82527 and bus driver Philips PCA82C250. To scan voltage values, the Analog Devices AD7890-10 AD converter was used. The monitoring program for the microprocessor node was written in the language C in the development system µVision2 by Keil Electronic. The application to set the node's properties was developed in the environment Borland Delphi 5.0 by Inprise Corporation. To verify the project, a C program for the microcomputer C167CR was developed in the µVision2 environment.

Prohlášení

Byl jsem seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 o právu autorském, zejména § 60 (školní dílo).

Beru na vědomí, že TUL má právo na uzavření licenční smlouvy o užití mé DP a prohlašuji, že **s o u h l a s í m** s případným využitím mé diplomové práce (prodej, zapůjčení apod.)

Jsem si vědom toho, že užít své diplomové práce či poskytnout licenci k jejímu využití mohu jen se souhlasem TUL, která má právo ode mne požadovat přiměřený příspěvek na úhradu nákladů, vynaložených univerzitou na vytvoření díla (až do jejich skutečné výše).

Diplomovou práci jsem vypracoval samostatně s použitím uvedené literatury na základě konzultací s vedoucím diplomové práce a konzultantem.

Datum: 23.5.2003

Podpis:

1. Obsah

2. Předmluva

Rád bych na tomto místě poděkoval několika lidem, bez jejichž podpory by tato diplomová práce pravděpodobně nevznikla.

Na prvním místě bych rád poděkoval vedoucímu diplomové práce, Ing. Josefu Grosmanovi, za cenné rady při konzultacích, celkovou trpělivost a materiální i morální podporu během tvorby této práce.

Dále bych chtěl poděkovat Ing. Radimu Vondrovi za rady při návrhu desky plošných spojů, také za rychlé a kvalitní zajištění výroby této desky.

Na závěr bych rád poděkoval všem blízkým lidem, kteří se mnou trpělivě trávili chvíle beznaděje, kdy jsem pouze bezmocně bušil hlavou do stolu.

3. Úvod

Prudký vývoj automatizační techniky v několika posledních desetiletích vykazuje stále vyšší požadavky na vzájemnou výměnu informací, jednak mezi jednotlivými řídícími systémy a jednak interakci s uživatelem těchto systémů, tj. s člověkem. Za tímto účelem bylo již vyvinuto mnoho různých, vzájemně nekompatibilních, komunikačních systémů a lze předpokládat, že tato oblast lidské činnosti bude i v budoucnu nadále prudce expandovat.

Při vývoji každého takového systému patří mezi základní požadavky jeho snadná realizovatelnost, odolnost proti negativnímu působení vnějších vlivů, bezpečnost a v neposlední řadě také efektivní rychlost přenosu dat. Jako jeden z nejzdařilejších vyvinutých systémů pro výměnu informací mezi řídícími systémy lze bezesporu nazvat protokol CAN, vyvinutý v polovině 80. let minulého století firmou Bosch v Německu, který splňuje požadavky jednoduché realizovatelnosti a vysoké odolnosti, při současném zachování bezpečného přenosu dat vysokou rychlostí.

Tato diplomová práce volně navazuje na ročníkový projekt vypsaný v akademickém roce 2001/2002 na katedře softwarového inženýrství, který byl realizován řešitelskou dvojicí P. Semenec, L. Novák. V tomto projektu byl realizován jednoduchý uzel sběrnice CAN, který pomocí řídícího mikroprocesoru Atmel AT89C51 a dále pomocí řadiče sběrnice CAN Intel AN82527 vysílal a přijímal data ze sběrnice CAN. Možnosti tohoto uzlu však byly značně omezené. Identifikátor zpráv bylo možno částečně měnit pomocí DIP přepínače umístěného na desce, vysílány a přijímány byly pouze datové zprávy o délce jednoho bajtu, přičemž přijímané zprávy byly zobrazovány pomocí LED diod. Přenosová rychlost, kterou bylo možno použít, byla pevně nastavena. Vysílané zprávy byly v podstatě generované matematické funkce, mezi nimiž mohl uživatel přepínat pomocí dalšího DIP přepínače.

Návrh uzlu sběrnice CAN popsaný v této diplomové práci řeší většinu nedostatků uzlu popsaného v předchozím odstavci, přičemž přejato bylo pouze hardwarové jádro předchozího návrhu. Pomocí aplikace, která je navržena pro operační systém Microsoft Windows, lze připojením uzlu k sériovému portu osobního počítače konfigurovat prostřednictvím rozhraní RS232 většinu podstatných vlastností uzlu jako je přenosová rychlost, identifikátory jednotlivých objektů zpráv, masky atd. Spíše z historických a vývojových důvodů zůstala tomuto uzlu zachována možnost zobrazení příchozích zpráv na LED diody, pro hlavní zobrazování přibyl dvouřádkový LCD displej. Uzel je dále vybaven maticovou klávesnicí, pomocí níž má uživatel možnost volby zobrazované zprávy, přičemž displej umožňuje zobrazovat plnou délku dat získaných ze sběrnice CAN, tj. 8 bajtů. Poslední možností

je snímání hodnot napětí pomocí sériového AD převodníku z až osmi vzájemně nezávislých zdrojů napětí a to v rozsahu ± 10 V. Takto získané hodnoty napětí jsou z části cyklicky vysílány na sběrnici CAN a z části je hodnota příslušných napěťových vstupů odvysílána pouze na žádost od jiného uzlu.

4. Popis použitých sběrnic

4.1 CAN

4.1.1 Historie protokolu CAN

Během vývoje se podařilo výrobcům a konstruktérům vytvořit celou řadu komunikačních protokolů k různým způsobům určení. V polovině 80. let minulého století se díky masivnímu vlivu německých automobilek začal však čím dál více prosazovat protokol CAN (Controller Area Network), vyvinutý firmou Robert Bosch, Stuttgart. Protokol CAN byl původně navržen pro distribuované řízení systémů automobilu v reálném čase s přenosovou rychlostí až 1Mbit/s a s vysokým zabezpečením přenosu proti chybám. Jedná se o protokol typu multimaster, kde každý uzel sběrnice může být master a řídit tak chování jiných uzlů. Není tedy nutné řídit celou síť z jednoho nadřazeného uzlu, což přináší zjednodušení řízení a zvyšuje spolehlivost, protože pokud dojde k poruše jednoho uzlu, zbytek sítě pracuje bez problémů dál. Tato první verze pro použití v automobilech byla firmou Bosch zveřejněna ve specifikaci CAN v. 1.2. Záhy se protokol CAN začal rozšiřovat i do dalších odvětví průmyslové výroby, např. robotika, sběr dat z distribuovaných měřících systémů, zabezpečovací zařízení atd. Proto firma Bosch zveřejnila v roce 1991 novou specifikaci, označovanou jako CAN v. 2.0. Ta obsahovala jednak původní verzi 1.2, nově označovanou jako CAN 2.0 A, s tzv. standardním 11 bitovým identifikátorem a současně novou verzi označovanou jako CAN 2.0 B, s rozšířeným 29 bitovým identifikátorem. O dva roky později, tj. v roce 1993 byl norma CAN v. 2.0 vydána jako standard ANSI / ISO 11898. To již nic nebránilo prudké expanzi téměř do všech odvětví průmyslové činnosti a k získání dominantního masivního rozšíření na trhu, dnes je sběrnice CAN zastoupena asi v 85 % z celkového použití průmyslových sběrnic. Na základě protokolu CAN se od 90. let vyvíjejí další protokoly, např. CANOpen, DeviceNet a SDA (Smart Distributed System).

4.1.2 Srovnání CAN s referenčním modelem ISO / OSI

V referenčním modelu ISO / OSI pro přenos dat v otevřených systémec je definováno sedm komunikačních vrstev, z nichž každá může komunikovat s vrstvou max. stejné úrovně prostřednictvím vrstev nižších, blíže k hardwaru. Význam jednotlivých vrstev je uveden v tab. 1. Počet těchto vrstev se u jednotlivých standardů však v praxi často redukuje. Ve specifikaci protokolu CAN jsou definovány pouze dvě nejnižší vrstvy, tj. fyzická a linková. Další čtyři vrstvy jsou nedefinovány a je použita až vrstva aplikační. V této oblasti

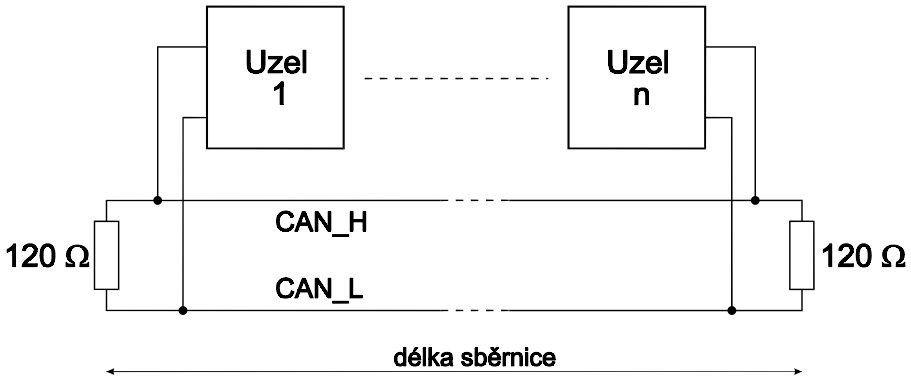
doposud neexistuje žádná norma upravující vlastnosti aplikační vrstvy protokolu CAN, existuje pouze několik vzájemně nekompatibilních standardů, např. CAL (CAN Application Layer), či již dříve zmiňovaný DeviceNet.

Číslo vrstvy	Název vrstvy	Funkce vrstvy
1	fyzická (Physical Layer)	definuje elektrické, mechanické, funkční a procedurální parametry síťového uzlu (např. RS485, RS422, CAN aj.)
2	linková (Data Link Layer)	definuje pravidla přenosu dat mezi dvěma uzly, tj. délku bloků dat, způsob zabezpečení dat, rozpoznávání chyb, opakování přenosů, pravidla přístupu na sběrnici atd.
3	síťová (Network Layer)	definuje pravidla pro přenos dat mezi podsítěmi
4	transportní (Transport Layer)	zajišťuje bezchybný přenos dat mezi koncovými systémy ; dále zajišťuje např. segmentaci delších přenášených zpráv na maximální povolenou délku paketů
5	relační (Session Layer)	řídí konverzaci dvou partnerských systémů (započetí, ukončení, synchronizační body)
6	prezentační (Presentation Layer)	provádí konverzi dat mezi obecným formátem definovaným protokolem a formátem definovaným pro daný koncový systém
7	aplikační (Application Layer)	poskytuje rozhraní k uživatelskému programovému vybavení, tzn., že uživateli jsou k dispozici definované funkce např. pro přenos souborů, emulaci terminálů, elektronickou poštu atd.

Tab. 1 Komunikační vrstvy referenčního modelu ISO/OSI

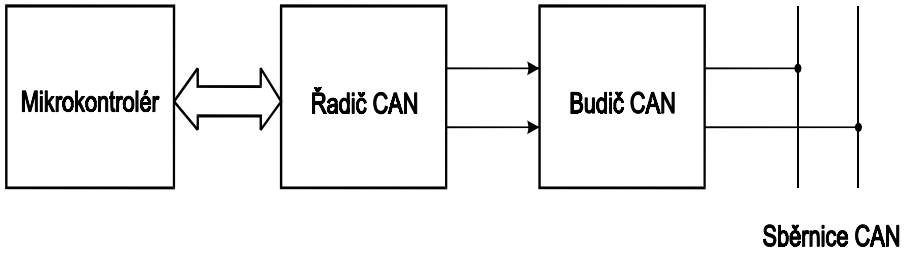
4.1.3 Fyzická vrstva

Sběrnice CAN je sběrnice sériová. Fyzicky je tvořena dvěma vodiči, které se nazývají páteř. Tyto vodiče jsou označovány ve schématech jako CAN H (High) a CAN L (Low). Vzájemně jsou oba vodiče na svých koncích spojeny rezistory s hodnotami odporu 120 Ω. Aby bylo dosaženo vyšší odolnosti proti elektromagnetickému rušení, jsou tyto vodiče nejčastěji v provedení kroucené dvoulinky (TP, Twisted Pair) nebo stíněných vodičů. Jednotlivá zařízení, označovaná též jako uzly, se potom ke sběrnici připojují pomocí odboček. Díky tomuto uspořádání má potom každé zařízení přímý přístup ke všem ostatním uzlům, bez ohledu na pořadí připojení ke sběrnici. Délka každé odbočky by měla být do 1 metru, fyzicky sousední odbočky z páteře by měly být vzájemně vzdáleny alespoň 10 cm. Teoreticky je počet uzlů připojitelných ke sběrnici neomezený, v praxi se často redukuje s ohledem na statické a dynamické parametry sběrnice max. na 30 připojených zařízení. Uspořádání sběrnice a připojení jednotlivých uzlů k páteři je znázorněno na obr. 1.



Obr. 1 Připojení uzlů CAN k páteřní síti

Každý uzel sběrnice CAN je zařízení, sestávající se z několika částí. Jednak z řídícího a komunikujícího zařízení, obvykle jednočipového mikropočítače (mikrokontroléru), dále z řadiče sběrnice CAN (CAN Controller), který řídí příjem a vysílání zpráv, nakonec z vysílače resp. přijímače sběrnice CAN, též označovaného jako budič sběrnice (CAN Transceiver, CAN Controller Interface). Takovéto uspořádaní se nazývá jako samostatný řadič CAN (Stand Alone). Někdy bývá mikrokontrolér, řadič a budič sběrnice CAN integrován do jednoho zařízení (pouzdra). Toto uspořádání se potom označuje jako integrované. Blokové schéma uzlu je znázorněno na obr. 2.



Obr. 2 Blokové schéma uzlu CAN

Každý vysílač může nabývat dvou stavů: aktivního, označovaného též dominant a pasivního, označovaného recessive. Spínač na sběrnici CAN H připíná napětí + 5 V, spínač na sběrnici CAN L připíná 0 V. Oba tyto spínače jsou s otevřeným kolektorem. Proto také může libovolný vysílač změnit stav sběrnice ze stavu recessive na stav dominant. Jsou-li vysílače všech uzlů ve stavu recessive, je celá sběrnice též ve stavu recessive. Je-li vysílač alespoň jednoho uzlu v hodnotě dominant, potom je celá sběrnice ve stavu dominant. Tato metoda je označována jako wired AND. Přehled možných stavů sběrnice je uveden v tab. 2.

Logická úroveň	CAN_H	CAN_L	Stav sběrnice
0	+ 5 V	0 V	dominant
1	pasivní	pasivní	recessive

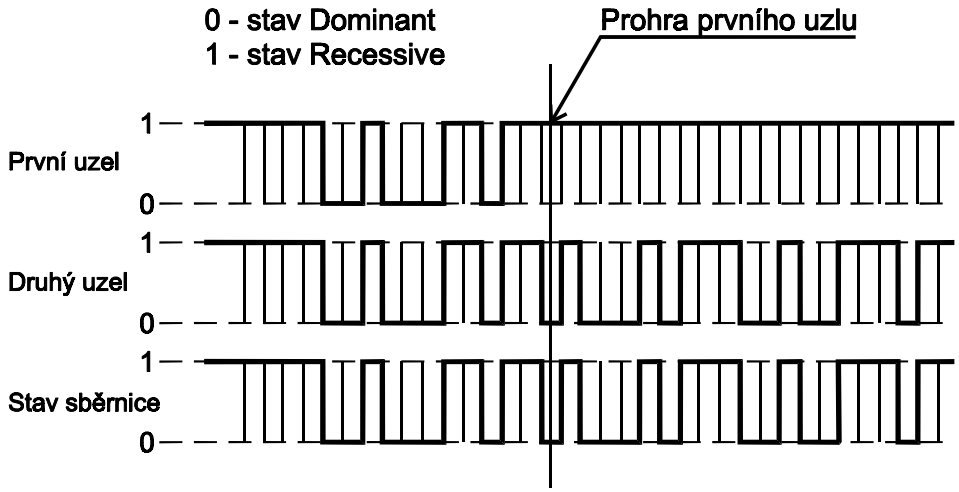
Tab. 2 Stavy sběrnice CAN

4.1.4 Linková vrstva

Linková vrstva protokolu CAN se skládá ze dvou podvrstev, MAC a LLC. První z nich má na starost přístup k médiu MAC (Medium Access Control). Úkolem je provádět kódování dat, vkládat doplňkové bity do komunikace (Stuffing/Destuffing), řídit přístup všech uzlů k médiu s rozlišením priorit zpráv, detekce chyb, hlášení chyb a potvrzování správně přijatých zpráv. Druhou podvrstvou je LLC (Logical Link Control), která filtruje přijaté zprávy (Acceptance Filtering) a hlásí přetížení (Overload Notification).

Informace se po sběrnici CAN šíří ve formě zpráv (Messages). Každé zařízení může vysílat libovolný počet zpráv a to jak ve formě cyklického vysílání nebo jako odpověď na konkrétní podnět od libovolné další jednotky, tzv. vyžádání zprávy (Remote Request). Metoda přístupu ke sběrnici je náhodná CSMA-CD (Carrier Sence Media Acces – Collision Detection), resp. náhodná s rozlišením kolize CSMA-CR (Carrier Sence Media Acces – Colision Resolution). Při vysílání zprávy na sběrnici může nastat několik případů. Pokud je sběrnice volná, libovolný uzel může zahájit vysílání své zprávy. Jestliže je sběrnice již

obsazena vysílajícím uzlem, ostatní uzly jsou ve stavu příjmu a čekají na ukončení vysílání. Nejzajímavější situace však nastává v okamžiku, kdy chce více uzlů současně zahájit vysílání. V tu chvíli totiž dochází ke sporu o sběrnici, který se řeší arbitrací. Každá zpráva vysílaná po sběrnici CAN má totiž svůj jednoznačný identifikátor, pomocí něhož se určuje priorita vysílané zprávy. Pokud tedy dojde ke sporu o sběrnici, všechny uzly začnou porovnávat svůj identifikátor se stavem sběrnice a pokud mají identifikátor s nižší prioritou, odpadají a stávají se přijímači. Vítězem sporu o sběrnici se stává uzel vysílající zprávu s identifikátorem nejvyšší priority, tedy vlastně s nejnižší číselnou hodnotou. Nevýhoda tohoto systému je zřejmá. Je-li na sběrnici připojeno příliš mnoho často vysílajících uzlů, dochází často ke kolizím, následné arbitraci a celková efektivita datové propustnosti klesá. Arbitrace mezi dvěma uzly v porovnání se stavem sběrnice je názorně vidět na obr. 3.



Obr. 3 Arbitrace mezi dvěma uzly

Ve specifikaci protokolu CAN jsou definovány celkem 4 druhy zpráv:

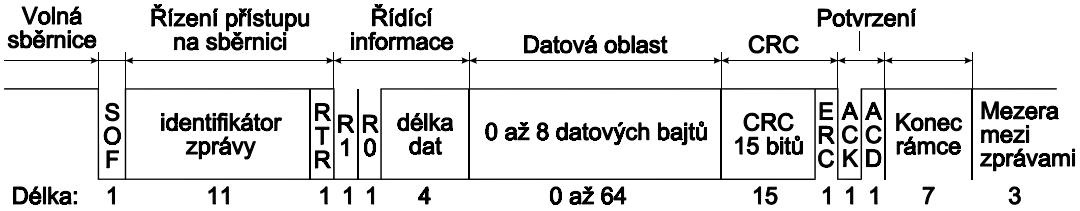
1. Datová zpráva (Data Frame) – základní prvek komunikace mezi uzly, identifikátor zprávy odpovídá vysílajícímu uzlu. Ve zprávě může být obsaženo až 8 datových bajtů. Pro jednoduché příkazy může být datová část prázdná, protože význam příkazu je patrný pouze z odvysílaného identifikátoru.
2. Žádost o data (Remote Frame) – uzel žádá některého z dalších o zaslání požadovaných dat. Datová část zprávy neobsahuje žádná data, identifikátor zprávy je shodný s identifikátorem uzlu, po kterém jsou data požadována. Odpovědí na takovouto žádost jsou data od uzlu, po kterém byla požadována.

3. Chybová zpráva (Error Frame) – tuto zprávu vysílá uzel, který zjistil chybu při příjmu zprávy.
4. Zpráva o přetížení (Overload Frame) – tato zpráva je odvysílána uzlem, který ještě nestihl zpracovat předchozí zprávu, další vysílání je potom oddáleno. Formát této zprávy je stejný jako zprávy chybové.

Zprávy uvedené v bodech 1. a 2. jsou také nazývány jako zprávy významové, zprávy uvedené v bodech 3. a 4. jsou označovány jako režijní.

Ve specifikaci CAN v 2.0 jsou uvedeny dva typy zpráv, verze 2.0 A se standardním 11 bitovým identifikátorem a verze 2.0 B s rozšířeným 29 bitovým identifikátorem. Struktura vysílaných zpráv se v jednotlivých verzích liší.

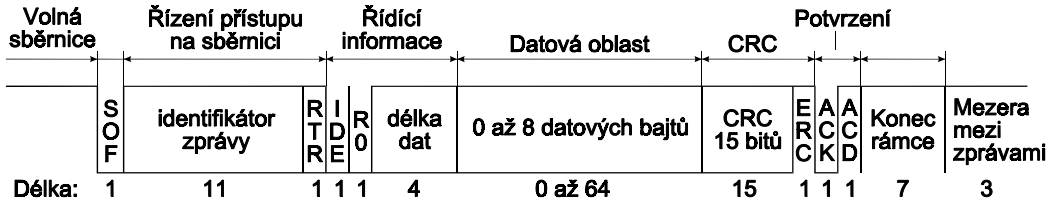
Struktura zprávy ve formátu CAN 2.0A je názorně vidět na obr. 4.



Obr. 4 Struktura zprávy ve formátu CAN 2.0A

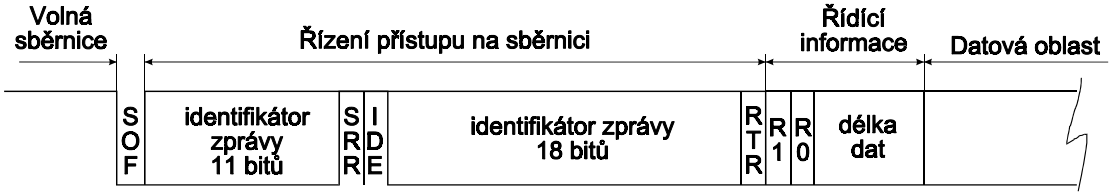
- SOF (Start Of Frame) - 1 bit, označení začátku zprávy
- Identifikátor zprávy - 11 bitů, udává význam přenášené zprávy
- RTR bit (Remote Transmission Request) - 1 bit, udává, zda jde o datovou zprávu (hodnota dominant) nebo o žádost o data (hodnota recessive)
- R1 - rezervovaný bit- R0 - rezervovaný bit
- Délka dat (Data Length) - 4 bity, udává počet přenášených datových bajtů ve zprávě
- Datová oblast - 0 až 8 datových bajtů
- CRC kód - 15 bitů, cyklický redundantní kód založený na bázi polynomů pro kontrolu správnosti přenosu
- ERC (End CRC) - 1 bit, ukončení CRC
- ACK - 1 bit, bit potvrzení vyhrazený pro přijímací uzly, v něm mohou potvrzovat správnost přijatých dat
- ACD - 1 bit, oddělovač potvrzení
- Konec zprávy (End Of Frame) - 7 bitů, pole vyhrazené pro přijímací uzly, v něm mohou informovat vysílací uzel o chybně přijatých datech
- Mezera mezi zprávami (Interframe Space) - 3 bity, odděluje zprávy

Zpráva ve formátu CAN 2.0 B může nabývat dvou podob. Jednak formát 2.0 B podporuje standardní 11 bitový identifikátor a jednak rozšířený 29 bitový identifikátor. Zpráva ve formátu CAN 2.0 B se standardním 11 bitovým identifikátorem je znázorněna na obr. 5.



Obr. 5 Struktura zprávy ve formátu CAN 2.0B s 11 bitovým identifikátorem

Jak je patrné při porovnání obr. 5 a obr. 4, jediným rozdílem je použití bitu IDE (Identifier Extension) informujícího o tom, zda je použit standardní nebo rozšířený formát zpráv. Tento bit se ve verzi 2.0 A nazývá R1 a je nevyužit. Zpráva ve verzi CAN 2.0 B při použití rozšířeného identifikátoru je na obr. 6.



Obr. 6 Struktura zprávy ve formátu CAN 2.0B s 29 bitovým identifikátorem

Použitý 29 bitový identifikátor je rozdělen do dvou částí. V první části je obsaženo 11 bitů identifikátoru jako u standardní verze, zbylých 18 bitů identifikátoru je v druhé části. Bit RTR (Remote Request) je přesunut až za druhou část identifikátoru, na jeho místo je vložen bit SRR (Substitute Remote Request), který má vždy hodnotu recessive. Díky tomu jsou při arbitraci vždy zvýhodněny zprávy se standardním 11 bitovým identifikátorem.

Kromě již zmíněných zabezpečovacích prostředků používá protokol CAN ještě následující:

- Vkládání bitů (Bit Stuffing) – vložení jednoho bitu opačné hodnoty do posloupnosti alespoň pěti po sobě jdoucích bitů stejné logické hodnoty. Toto se používá pro potřeby synchronizace.
- Monitorování sběrnice vysílačem – vysílač porovnává požadovanou úroveň signálu na sběrnici se skutečným stavem

- Hlášení chyby – přijímač sestavuje chybové hlášení složené z Error Flag, což je 6 po sobě jdoucích bitů hodnoty dominant a z Error Delimiter, což je 8 po sobě jdoucích bitů recessive

Pokud je vše v pořádku, odvysílá příslušná přijímací jednotka potvrzení o správnosti dat, které přijme vysílající uzel. Každý uzel CAN proto navíc bývá vybaven mechanizmem, který zajistí odpojení uzlu v případě, že by se sám v důsledku poruchy stal zdrojem příliš mnoha chybových zpráv, aby zbytečně nezahlcoval sběrnici přílišným počtem chybových zpráv, event. nesmyslných údajů.

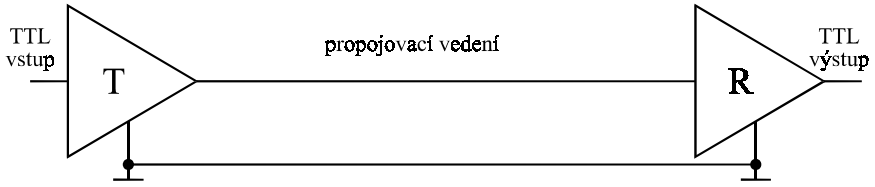
4.1.5 CAN v normách

V normách, které definují protokol CAN, tj. normy ISO 11898 a ISO 11519, se definuje protokol CAN na dva typy sběrnic. Na tzv. „rychlý CAN“ (High Speed) s přenosovou rychlostí od 125 kBaud do 1 MBaud s délkou vedení do 30 m a na tzv. „pomalý CAN“ (Low Speed) s přenosovou rychlostí menší nežli 125 kBaud a s délkou vedení do 1000 m. V této druhé verzi by navíc měla být podle normy sběrnice CAN funkční i v případě přerušení jednoho z vodičů. Další normy, které vznikly pro protokol CAN jsou např. u výrobců osobních automobilů velice různé, liší se podle výrobce a obecně jsou nekompatibilní. Jedinou normu, kterou lze za normu považovat, vydala americká Asociace automobilových inženýrů SAE, resp. její skupina elektrizace a elektronizace nákladních vozidel a autobusů a vydala ji pod označením J 1939. Tento dokument má přibližně sedm průběžně doplňovaných a upřesňovaných částí.

4.2 Sběrnice RS232C

4.2.1 Fyzická vrstva

Sériové komunikační rozhraní podle TIA/EIA (Telecommunications Industry Association / Electronic Industries Association) 232 F je schématicky znázorněno na obr. 7.

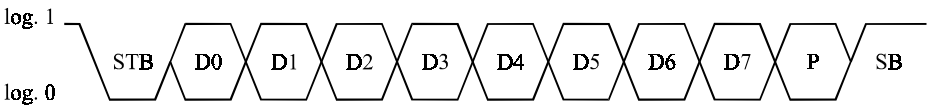


Obr. 7 Fyzická vrstva rozhraní RS232

Propojovací vedení není kompatibilní s úrovněmi TTL. Logická nula na výstupu vysílače (T, Transmitter) v hodnotách +5 V až +15 V, logická jedna na výstupu vysílače -5 V až -15 V. Na vstupu přijímače (R, Receiver) jsou podle normy TIA/EIA přípustné hodnoty pro logickou nulu jsou od +3 V, pro logickou jedničku od -3 V, přičemž šumová imunita musí být minimálně 2 V. Maximální délka vedení RS232C je 15 metrů, přičemž limitujícím faktorem není délka vedení, ale zatěžovací kapacita vysílače, která smí být maximálně 2500 pF. Při použití víceželového kabelu dochází k přeslechům, z toho důvodu je rychlost změny na signálových vodičích normou TIA/EIA 232F omezena na 30 V/μs, doba průchodu zakázaným pásmem však nesmí překročit 4 % doby přenosu jednoho bitu. Teoretická rychlost přenosu dat získaná z těchto podmínek by mohla být 200 kbit/s, prakticky se však podle normy používají přenosové rychlosti ležící ve standardní řadě, tj. 50, 75, 110, 150, 300, 600, 1200, 2400, 4800, 9600 a 19200 bit/s. Při překročení rychlostí, které stanovuje norma, se potom nejčastěji používají rychlosti 28800, 38400, 57600, 115200 bit/s.

4.2.2 Linková vrstva

Formát přenosu dat po rozhraní TIA/EIA 232F je znázorněn na obr. 8.



Obr. 8 Formát přenosu dat po sběrnici RS232

Pokud je vedení v klidovém stavu, je na něm hodnota logické jedničky. Vysílání je zahájeno přivedením jednoho start bitu (STB), která má hodnotu logické nuly. Poté následuje 5 až 8 datových bitů a jeden bit paritní. Tento paritní bit může mít několik významů:

- Sudá parita - bit nastaven tak, aby celkový počet jedničkových bitů ve vyslaném slově byl sudé číslo
- Lichá parita - celkový počet jedničkových bitů je liché číslo
- Nulová parita (Space Parity) - bit je vždy logická nula
- Jedničková parita (Mark Parity) - bit je vždy logická jednička
- Žádná parita - paritní byt je nepoužit

Celá sekvence je ukončena tzv. stop bitem/-y (SB), který má hodnotu logické jedničky, délku 1, 1.5 nebo 2 bity a slouží k oddělení jednotlivých slov vysílaných po sběrnici.

4.2.3 Signály rozhraní TIA / EIA 232 F

- SG (Signal Ground) - signálová zem
- TxD (Transmitted Data) - vysílaná data
- RxD (Received Data) - přijímaná data
- RTS (Request to Send) - požadavek na vysílání, vysílač tímto přijímači oznamuje, že je připraven k vysílání
- CTS (Clear to Send) - pohotovost přijímače, přijímač vysílači oznamuje svoji připravenost k přijímaní dat
- DSR (Data Set Ready) - signál přijímače oznamující jeho připojení a připravenost
- DTR (Data Terminal Ready) - signál pohotovosti vysílače k příjmu nebo k vysílání
- DCD (Data Carrier Detect) - detekování nosné
- RI (Ring Indicator) - indikace volání

5. Hardwarová realizace

Při hardwarovém návrhu bylo dbáno na jednoduchost a funkčnost celého zařízení, napěťovou a rychlostní kompatibilitu použitých obvodů a pokud možno co nejlogičtější uspořádání z hlediska pozdějšího snadného programování řídícího programu. Neméně důležitými kritérii byla cena použitých součástek a hlavně jejich dostupnost na našem trhu.

5.1 Stručný popis použitých komponent

5.1.1 Řídící mikrokontrolér Atmel AT89C52

Mikroprocesor Atmel AT89C52 je nízkopříkonový, vysoce výkonný procesor vyrobený technologií CMOS, vybavený 8 kB programové paměti typu Flash a plně programově i svými vývody kompatibilní s průmyslovým standardem 80C51 a 80C52. Flash paměť může být programována v režimu ISP (In System Programming) nebo konvenčními programátory, počet cyklů mazání/zápis je minimálně 1000. Frekvence připojeného oscilátoru může být v rozsahu 0 Hz až 24 MHz, obvod obsahuje tříúrovňový zámek programové paměti (Three-level Program Memory Lock). Procesor dále obsahuje 256 B paměti typu RAM, 32 vstupně-výstupních linek uspořádaných do čtyř osmibitových portů, 3 na sobě vzájemně nezávislé 16 bitové čítače/časovače a programovatelný sériový kanál. Je umožněno až osm zdrojů přerušení a energeticky úsporné režimy při nečinnosti (Low-power Idle, Power-down Mode) . Procesor AT89C52 je vyráběn v pouzdrech PQFP/TQFP, PDIP a PLCC. Více informací lze nalézt v publikaci [14] a v navazující literatuře.

5.1.2 Řadič sběrnice CAN Intel AN82527

Řadič sériové komunikace AN82527 je zařízení s vysokou integrací součástek, které umožňuje sériovou komunikaci pomocí protokolu CAN. Plně podporuje normu CAN ve specifikaci 2.0, umožňuje využívat standardní i rozšířený identifikátor pro datové rámce zpráv i pro rámce žádostí o data. Je vybaven možností programovatelné masky pro standardní i rozšířený formát. Obsahuje celkem 15 objektů zpráv, z nichž 14 umožňuje vysílat i přijímat data, patnáctý objekt zpráv je určen pouze pro příjem dat a je vybaven vlastní programovatelnou maskou. Existuje více možností propojení tohoto řadiče s řídícími procesory: 8 bitové multiplexované, 16 bitové multiplexované, 8 bitové synchronní nemultiplexované, 8 bitové asynchronní nemultiplexované a sériové propojení. U tohoto řadiče lze programovat přenosovou rychlost, dále je vybaven programovatelným zdrojem hodinového signálu, má flexibilní strukturu přerušení a konfigurovatelný vstupní komparátor.

Navíc je rozšířen o dva vstupně/výstupní 8 bitové porty. Řadič Intel AN82527 je vyráběn v pouzdrech QFP44 a PLCC44. Více informací lze nalézt v publikacích [9],[10],[18].

5.1.3 Budič sběrnice CAN Philips PCA82C250

Obvod PCA82C250 je rozhraní mezi řadičem sběrnice CAN a fyzickou sběrnicí. Převádí rozdílné úrovně logických signálů při práci se sběrnicí CAN. Budič je plně kompatibilní s normou ISO 11898, zvládá přenosové rychlosti až do 1 MBaud, vodiče sběrnice jsou chráněny proti poruchám ve vybavení automobilu, je redukována možnost rádiové interference (Radio Frequency Interference) a interference s elektromagnetickým rušením (Electro Magnetic Interference). Obvod má tepelnou ochranu a nízkou spotřebu v pasivním režimu. Nezapojený uzel nemá možnost rušit sběrnici a současně může být na sběrnici připojeno až 110 uzlů CAN. Více informací lze nalézt v publikacích [7] a [8].

5.1.4 AD převodník Analog Devices AD7890-10

AD převodník AD7890-10 je dvanáctibitový, osmikanálový, vysokorychlostní převodník s dobou konverze 5,9 μs. Vstupní napěťový rozsah každého kanálu je ± 10 V a všechny analogové vstupní kanály jsou jednotlivě zakončeny. Je povolen oddělený přístup k multiplexeru a AD převodníku, na čipu je integrován obvod referenčního napětí 2,5 V. Výstup z převodníku je sériový, vysokorychlostní a flexibilní. Je použito jednoduché napájení, převodník má malou spotřebu energie (max. 50 mW) a umožňuje přepnutí do režimu s nízkou spotřebou (Power-Down mode), při kterém má spotřebu cca. 75 μW.

5.1.5 LCD displej LMO16L

LMO16L je LCD displej umožňující zobrazení až dvou řádků textu po šestnácti znacích. Je vybaven vestavěným řadičem HD44780 s vnitřním oscilátorem, 80 bajty paměti typu RAM pro operaci s až 80 znaky, předdefinovanou tabulkou 160 znaků a možností definování až 8 vlastních znaků uživatelem, všechny znaky v rozlišení 5 x 7 bodů. Displej umožňuje 4 bitové nebo 8 bitové datové propojení s řídícím mikroprocesorem, další tři signály jsou potřeba jako řídící. Speciální řídící příkazy pro mazání displeje, přesun kurzoru na počátek, vypínání a zapínání displeje a kurzoru, posun znaků atd. jsou vyjádřeny jako číselné instrukce.

5.1.6 Maticová klávesnice Velleman

Tato klávesnice se 16 tlačítky v maticovém uspořádání 4 x 4 umožňuje připojení všech 16 tlačítek k pouhým osmi bitům jednoho portu. Vývody symbolizují čtyři řádky a čtyři

sloupce tlačítek. Klávesnice je určena pro napětí max. 24 V při proudovém zatížení 20 mA, maximální přechodový odpor jednotlivých spínačů je 200 Ω a životnost je garantována na 1 mil. stisků každého tlačítka.

5.1.7 Převodník úrovní TTL ⇔ RS232 MAX232CPE

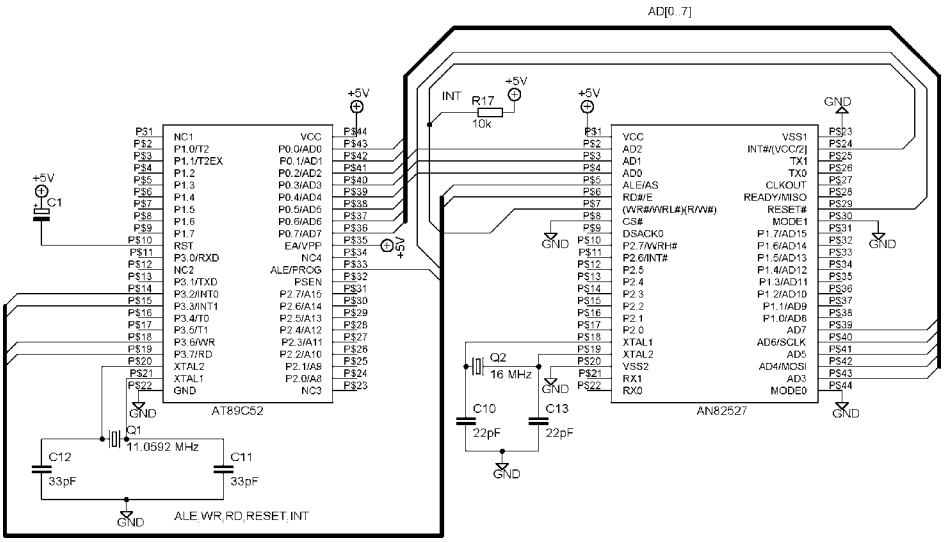
MAX232 je rozhraní pro převod signálů na úrovně RS232 pro obvody, kde není dostupné napětí ± 12 V. Obvod obsahuje DC-DC měnič založený na principu nábojové pumpy, kdy si za pomoci externích kondenzátorů vyrábí napětí + 10 V a - 10 V. Jelikož je schopen pracovat již od velmi malých napětí (cca. 0,2 V) je možné jej využít také pro napájení nenáročných obvodů z baterií.

5.2 Propojení jednotlivých součástek

Veškeré obrázky použité v této kapitole byly nakresleny v návrhovém systému Eagle od německé firmy CadSoft Computer GmbH . Celkové schéma zapojení realizovaného uzlu CAN je možno nalézt jako přílohu v kapitole 10.5.

5.2.1 Atmel AT89C52 - Intel AN82527

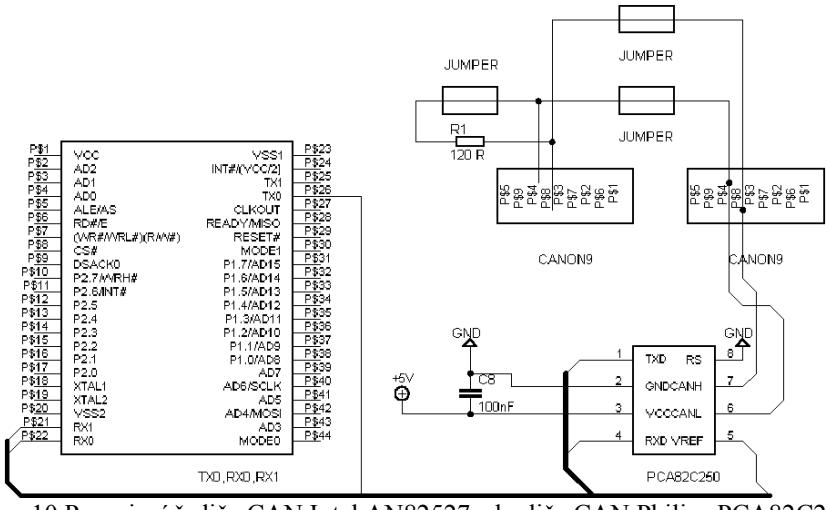
Propojení řídícího mikrokontroléru Atmel AT89C52 s řadičem sběrnice CAN Intel AN82527 je v 8 bitovém multiplexovaném režimu, kdy řadič CAN je připojen jako externí paměť o velikosti 256 bajtů pomocí signálů DB0 ÷ DB7, což je alternativní funkce portu P0 mikroprocesoru AT89C52. K tomuto propojení ještě náleží řídící signály $\overline{WR}$ (P3.6), $\overline{RD}$ (P3.7) a ALE (Adress Latch Enable). Signály $\overline{WR}$ a $\overline{RD}$ slouží pro informování, zda jsou data zapisována nebo čtena, signál ALE dává řadiči CAN informaci o tom, zda hodnota na portu P0 představuje adresu či data. Vzájemné propojení se ještě sestává ze signálu $\overline{INT}$ (Interrupt), který generuje Intel AN82527, přivedeného jako zdroj vnějšího přerušení na vstup mikrokontroléru $\overline{INT0}$ (P3.2) a signálu $\overline{RESET}$ (P3.3), pomocí něhož může mikrokontrolér mazat nastavení řadiče CAN. Dále jsou připojeny obvody oscilátorů obou obvodů a napájecí vývody. U AT89C52 je navíc přítomno napájení + 5 V u vývodu EA/$\overline{VPP}$ (volba spouštění programu z vnitřní paměti Flash) a resetovací obvod u vývodu RST. Připojení napájení GND k vývodu $\overline{CS}$ (Chip Select) řadiče CAN aktivuje tento obvod pro permanentní užívání, napájení GND na vývodech MODE0 a MODE1 volí 8 bitový multiplexovaný režim komunikace, napájení GND na vývodu V_{SS1} značí digitální zem obvodu a napájení GND na V_{SS2} představuje zemnění pro analogový komparátor. Schéma propojení je uvedeno na obr. 9.



Obr. 9 Propojení mikrokonroléru Atmel AT89C52 a řadiče CAN Intel AN82527

5.2.2 Intel AN82527 - Philips PCA82C250

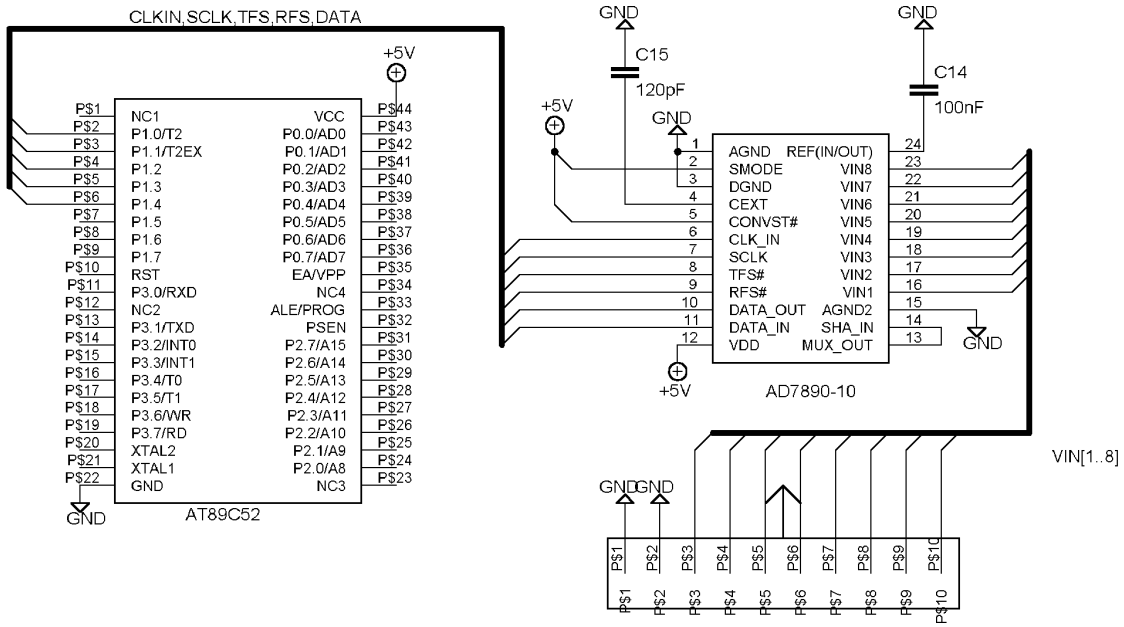
Propojení mezi budičem sběrnice CAN a jejím řadičem je realizováno pomocí signálů TX0 (vysílaná data), RX0 (přijímaná data) a RX1 (použito jako referenční napětí). Signál TX1 řadiče CAN není využit. Přítomno je i napájení budiče CAN a jeho vývody na sběrnici CAN_H a CAN_L, které jsou připojeny na piny č. 8 a 4 konektoru Canon9-M (vidlice). Dále jsou použity dvě zkratovací propojky (JUMPER) pomocí nichž lze nastavit propojení dvou konektorů Canon9-M a tím i pokračování sběrnice CAN. Třetí zkratovací propojka připojuje zakončovací rezistor s hodnotou odporu 120 Ω pro případ, že by uzel byl připojen na konci sběrnice. Popsané zapojení je na obr. 10.



Obr. 10 Propojení řadiče CAN Intel AN82527 a budiče CAN Philips PCA82C250

5.2.3 Atmel AT89C52 - Analog Devices AD7890-10

Propojení mikrokontroléru s AD převodníkem je realizováno pomocí pěti signálů připojených k některým pinům portu P1 mikroprocesoru. Pin P1.0 je použit jako zdroj hodinového signálu (CLKIN), P1.1 slouží jako signál $\overline{\text{TFS}}$ (Transmit Frame Synchronization Pulse), P1.2 má funkci signálu $\overline{\text{RFS}}$ (Receive Frame Synchronization Pulse), P1.3 je využit jako signál DATA (společný pro vývody DATA IN a DATA OUT AD převodníku) a P1.4 je zdrojem asynchronního hodinového signálu SCLK (Serial Clock Input). Analogové vstupy jsou připojeny na šroubovací svorkovnici. Dále jsou zobrazeny symboly napájení + 5 V (VDD) a GND (AGND, AGND2, DGND). Vývody převodníku MUX OUT (Multiplexer Output) a SHA IN (Track/Hold Input) jsou propojeny. Pin pro připojení externí reference REF IN/OUT je přes kondenzátor 100 nF uzemněn, dále je připojen kondenzátor 120 pF k vývodu CEXT, který určuje délku vnitřního pulzu pro zpoždění prostřednictvím externího antialiasingového filtru nebo dalších obvodů. Napájení + 5 V je též připojeno k vývodu $\overline{\text{CONVST}}$ (Conversion Start), který umožňujě hardwarové spuštění převodu. Vzhledem k tomu, že konverze je spouštěna programově, je tento vývod připojením + 5 V vyřazen ze své činnosti. Schéma zapojení je na obr. 11.

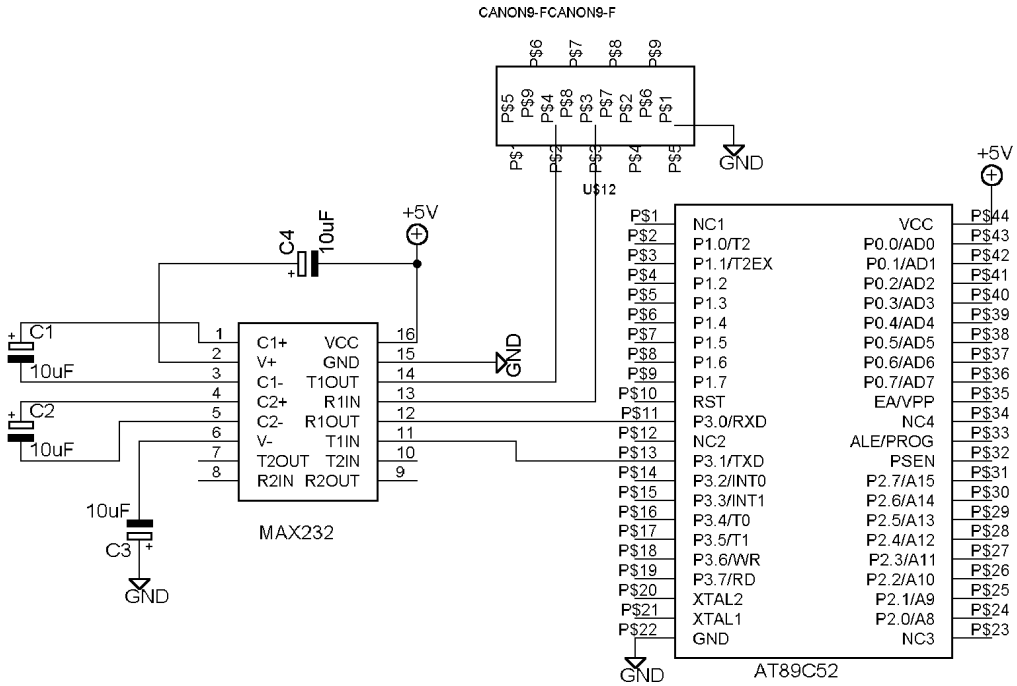


Obr. 11 Propojení mikrokontroléru Atmel AT89C52 a AD převodníku AD7890-10

5.2.4 Atmel AT89C52 - MAX232CPE

MAX232 je k mikroprocesoru připojen pomocí dvou vodičů. U mikroprocesoru jsou použity piny RxD (P3.0) a TxD (P3.1). Vstup resp. výstup dat na sériovou linku je z pinů obvodu MAX232 T1OUT (TxD) a R1IN (RxD) připojené na piny č. 3 resp. č. 2 konektoru

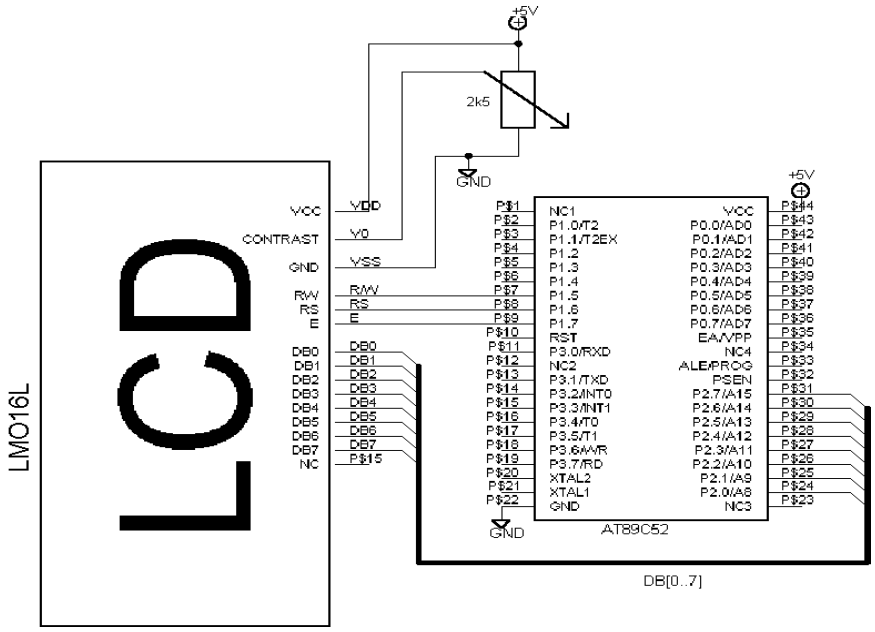
Canon9-F (zásuvka). Opačný konektor vzhledem k použitým konektorů pro připojení na sběrnici CAN byl zvolen z důvodu nemožnosti jejich záměny uživatelem. Dále je zobrazeno připojení napájení a externích kondenzátorů pro obvod MAX232. Schéma propojení je na obr. 12.



Obr. 12 Propojení mikrokontroléru Atmel AT89C52 a převodníku MAX232CPE

5.2.5 Atmel AT89C52 - LMO16L

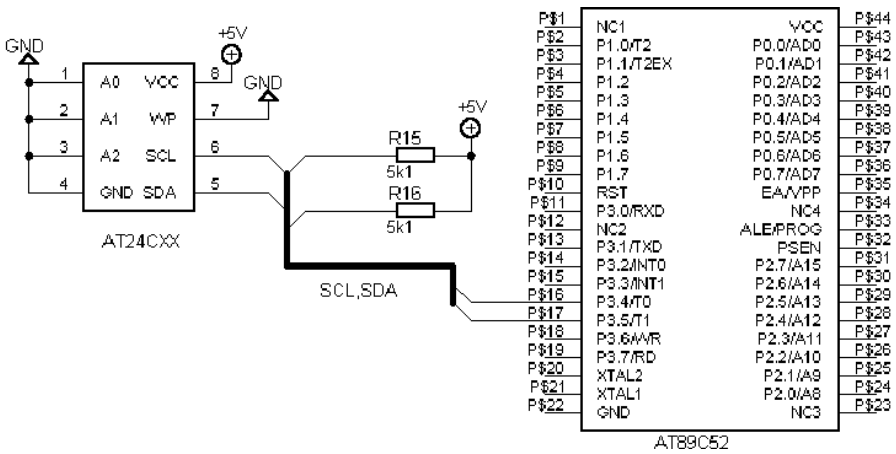
Datové propojení mezi mikroprocesorem a displejem je 8 bitové, displej je připojen k portu P2 mikroprocesoru. Pro řídící signály jsou použity zbylé tři piny portu P1, konkrétně P1.7 pro signál E (Enabled), P1.6 pro signál RS (Register Select) a P1.5 pro signál R/W (Read/Write Select Signal). Dále jsou zobrazeny symboly napájení + 5 V (VDD) a GND (VSS). Dále je připojen obvod pro nastavování kontrastu displeje, sestávající se z trimru s nominální hodnotou odporu 2,5 kΩ, jehož jezdec je připojen na vývod V0 a zbylé dva vývody jsou připojeny mezi napájecí napětí + 5 V a GND. Schéma propojení displeje s mikroprocesorem je na obr. 13.



Obr. 13 Propojení mikrokontoléru Atmel AT89C52 a LCD displeje LMO16L

5.2.6 Atmel AT89C52 - Atmel AT24Cxx

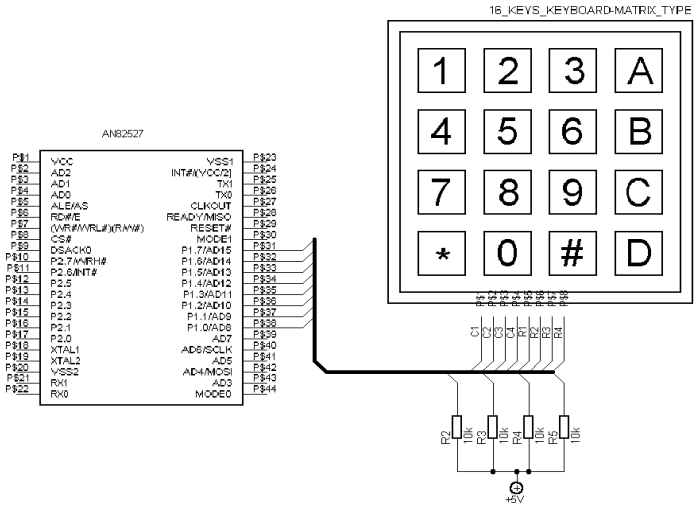
V hardwarovém návrhu je počítáno s připojením sériové paměti EEPROM, do které by se ukládala uživatelská konfigurace. Paměť typu EEPROM je z řady 24Cxx a s mikroprocesorem je propojena dvouvodičově pomocí sběrnice I^2C. Propojení je realizováno signálem SCL připojeným na pin P3.4 a signálem SDA připojeným na pin P3.5. Paměť má pevně zvolenou adresu 00 H zařízení na sběrnici I^2C pomocí napájení GND na vstupech A2, A1, A0. Připojením pinu WP (Write Protect) na GND dojde k odstranění ochrany proti zápisu. Na schématu je zobrazeno napájení + 5 V (VCC) a GND (GND). Schéma propojení je na obr. 14.



Obr. 14 Propojení mikrokontroléru Atmel AT89C52 a paměti EEPROM Atmel AT24C04 sběrnicí I^2C

5.2.7 Intel AN82527 - maticová klávesnice Velleman

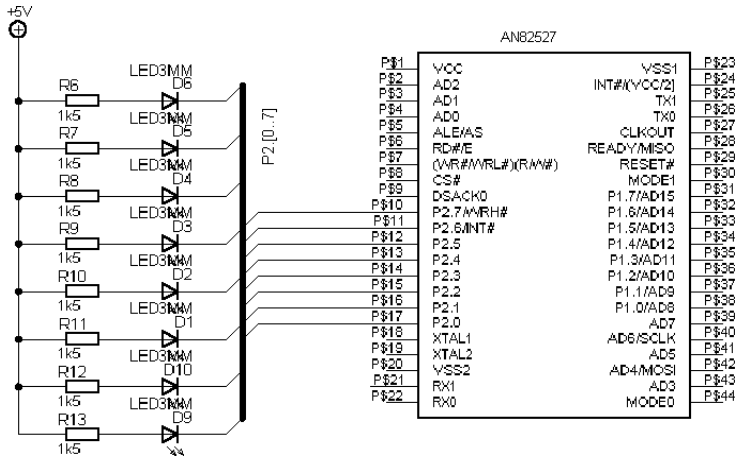
Maticová klávesnice je připojena k rozšiřujícímu portu P1 řadiče CAN. Výstupy klávesnice reprezentující jednotlivé řádky jsou připojeny k nižší polovině portu a jsou brány jako vstupy, protože k nim je současně připojeno přes rezistory s hodnotou odporu 10 kΩ napájecí napětí + 5 V. Vývody klávesnice představující jednotlivé sloupce klávesnice jsou připojeny na horní polovinu portu a jsou brány jako výstupy. Připojení klávesnice k portu je zobrazeno na obr. 15.



Obr. 15 Propojení maticové klávesnice a řadiče CAN Intel AN82527

5.2.8 Intel AN82527 - LED diody

Osm LED diod je připojeno k rozšiřujícímu portu P2 řadiče CAN. Nízkopříkonové LED diody ∅ 3 mm, odebírající proud cca. 2 mA jsou k jednotlivým bitům portu připojeny napájecím napětím + 5 V přes rezistory s hodnotou odporu 1,5 kΩ. Nízký odběr LED diod je navržen s ohledem na co možná nejnižší proudové zatížení portu. Připojení LED diod je znázorněno na obr. 16.



Obr. 16 Propojení LED diod a řadiče CAN Intel AN82527

5.2.9 Návrh desky plošných spojů

Pro návrh desky plošných spojů (DPS) byl použit návrhový systém Eagle 4.09r2 od německé firmy CadSoft Computer GmbH, která nabízí na svých internetových stránkách http://www.cadsoft.de zdarma tento produkt ke stažení pro výukové účely. Tato freeware verze má však omezení v podobě prostoru na DPS, na kterém lze navrhovat. Z tohoto důvodu a z důvodu množství rozměrných součástek (klávesnice, LCD, konektory) bylo zvoleno uspořádání do dvou vrstev, kde veškeré aktivní a pasivní součástky (vyjma konektorů a napájecích obvodů) jsou umístěny pod klávesnicí a klávesnice s displejem jsou k desce přišroubovány pomocí distančních sloupků. Plošný spoj je oboustranný, s vodivými průchodkami, šířka vodivých spojů je 0,3 mm. Vyroben byl technologií rozlévané mědi, přičemž šířka izolační mezery mezi spoji a plochou rozlité mědi je minimálně 1,2 mm a plochy rozlité mědi jsou uzemněny. Schéma zapojení a návrh DPS v programu Eagle je možné najít na přiloženém CD v adresáři */Eagle/DPS*. Pro vytvoření schématu zapojení a následně desky plošných spojů bylo zapotřebí nakreslit v programu Eagle vlastní knihovny součástek. Tyto knihovny lze též nalézt na přiloženém CD v adresáři */Eagle/lib*.

Při vývoji zařízení bylo použito univerzální desky plošných spojů, kdy spoje jsou realizovány pomocí izolovaných vodičů napájených na vývody jednotlivých součástek. Fotografie vývojové desky lze nalézt na přiloženém CD v adresáři */Universal*.

6. Software

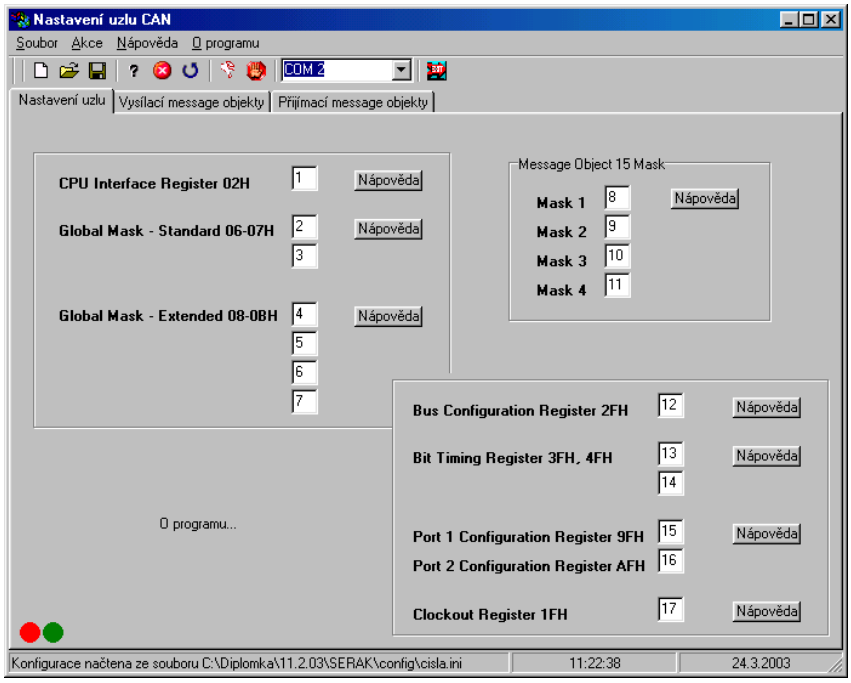
6.1 Aplikace pro PC k nastavování vlastností uzlu CAN

6.1.1 Popis aplikace

Tato aplikace, nazývaná též CAN Setup v. 1.2, je určena k uživatelskému nastavování podstatných vlastností uzlu CAN prostřednictvím sériového rozhraní RS232, kterým je standardně vybaven každý osobní počítač kompatibilní s IBM PC. Nastavování vlastností uzlu znamená vlastně předávání zakódovaných informací, jež předtím uživatel zadal v jemu srozumitelné podobě, prostřednictvím sériové linky řídícímu mikrokontroléru uzlu, který je poté předá řadiči sběrnice CAN a tím ovlivní chování uzlu vzhledem ke sběrnici CAN. Úkolem aplikace tedy je přijímat od uživatele data, umožnit mu s těmito daty pracovat, přijaté údaje vhodně zakódovat, předat tento zakódovaný obsah prostřednictvím sériové linky řídícímu mikrokontroléru uzlu, ověřit správnost doručení dat a v případě chyby provést nápravu, event. informovat uživatele o provozních stavech resp. poruchách uzlu. Zdrojový kód aplikace je na přiloženém CD v adresáři */Source/CAN_Setup*.

6.1.2 Vzhled a ovládání aplikace

CAN Setup v. 1.2 byl vytvořen ve vývojovém prostředí Borland Delphi 5 od firmy Inprise Corporation pro operační systémy Microsoft Windows 95/98/NT/ME/2000/XP, program je plně lokalizován do českého jazyka. Vzhled aplikace je na obr. 17.



Obr. 17 Vzhled aplikace CAN Setup v. 1.2

Pro správnou funkčnost a pohodlnou práci je vyžadováno nastavené rozlišení obrazovky minimálně 800 x 600 bodů, nainstalovaný a asociovaný prohlížeč html stránek, dále funkční a volný sériový port.

První částí pro ovládání aplikace je tzv. roletové menu, ovladatelné jednak z klávesnice a jednak myší. Toto menu obsahuje položky „Soubor“, „Akce“, „Nápověda“, „O programu“. Roleta „Soubor“ (zkratková klávesa Alt+S) obsahuje položky „Vymazat nastavení“ (Ctrl+N), „Otevřít nastavení“ (Ctrl+O) a „Uložit nastavení“ (Ctrl+S). Při volbě „Vymazat nastavení“ se z editačních polí nenávratně vymažou veškeré uživatelem zadané informace. Volba „Otevřít nastavení“ uživateli zobrazí dialogové okno pro otevření souboru s již uloženou konfigurací. Soubory s konfigurací mají příponu *.ini, kterou nelze změnit. Výchozí adresář s konfiguracemi je nastaven na */config*, lze jej však v případě potřeby změnit. Poslední položkou v menu „Soubor“ je „Uložit nastavení“. Tato funkce umožňuje uložit uživateli svou stávající konfiguraci do souboru pro pozdější použití. Pro práci s příponou souboru a výchozím adresářem platí to samé co v případě příkazu „Otevřít nastavení“. Poslední položkou v nabídce „Soubor“ je možnost „Ukončení programu“ (Ctrl+K). Tato funkce ukončí program bez jakéhokoliv dalšího varování.

Druhou částí roletového menu je položka „Akce“ (Alt+A) obsahující podmenu nabízející „Inicializace uzlu“ (Ctrl+Alt+I), „Reset uzlu“ (Ctrl+Alt+R), „Nastav vše“ (Ctrl+Alt+N), „Spuštění uzlu“ (Ctrl+Alt+S) a „Zastavení uzlu“ (Ctrl+Alt+Z). Pokud není správně zvolen a dostupný sériový port, je celá nabídka neaktivní a uživateli nepřístupná, protože slouží ke komunikaci s připojeným uzlem CAN. Příkaz „Inicializace uzlu“ ověřuje, zda je zmíněný uzel správně připojen a je připraven přijímat data a příkazy prostřednictvím sériové linky. Pokud tomu tak je, aplikace vydá informační hlášení „Inicializace provedena, status OK“. V případě chybného propojení nebo nefunkčnosti uzlu aplikace uživateli zobrazí hlášení chybové „Chyba komunikace“. Toto chybové hlášení je společné pro chybové stavy všech neuskutečněných příkazů z nabídky „Akce“ s vyjímkou příkazu „Nastav vše“. Položka „Reset uzlu“ předá mikrokontroléru uzlu příkaz ke zresetování připojeného řadiče, vykonání příkazu je uživatel informován ve stavovém řádku (viz. dále popis stavového řádku), v opačném případě již výše zmíněným chybovým hlášením. Příkaz „Nastav vše“ spustí kódování informací z editačních polí, jejich předávání po sériové lince mikrokontroléru, ověření správnosti doručených dat a jejich případné opravy. O průběhu procesu je uživatel informován procentuálním ukazatelem průběhu, který se zobrazí na první záložce „Nastavení uzlu“ (viz. dále popis záložek). Po skončení procesu tento ukazatel zmizí. V případě správného průběhu nastavování vydá na jeho konci aplikace informační hlášení „Kompletní

nastavení provedeno“. V případě výskytu více než 10 chyb za sebou během přenosu vydá hlášení chybové „Chyba komunikace, nepodařilo se navázat spolehlivé spojení“. V tomto případě se doporučuje provést reset celého uzlu a zkontrolovat funkčnost propojení. Položka „Spuštění uzlu“ předá uzlu příkaz pro vykonávání řídícího programu uloženého v mikrokontroléru uzlu. O tomto spuštění je uživatel informován jednak ve stavovém řádku (viz. dále popis stavového řádku) a také rozsvícením zeleného indikačního prvku umístěného v levém dolním rohu na první záložce „Nastavení uzlu“. Posledním příkazem v tomto podmenu je příkaz „Zastavení uzlu“, který zastaví řídící program mikrokontroléru, zhasne indikátor běhu tohoto programu, rozsvítí vedle umístěný indikátor zastavení uzlu a informuje uživatele ve stavovém řádku.

Další položkou v roletovém menu je položka „Nápověda“ (Alt+N), která obsahuje odkazy na všechny pojmy používané v souvislosti s nastavováním registrů řadiče sběrnice CAN. Pokud je v operačním systému správně nainstalován a asociován prohlížeč souborů s příponou *.htm, aplikace předá systému příkaz na spuštění tohoto prohlížeče, přičemž parametrem je soubor ve formátu html. Uživateli se v prohlížeči zobrazí příslušná část nápovědy v kontextu s volbou nápovědy k danému problému. Soubory s nápovědou jsou vzájemně provázány odkazy, není tedy nutné pro každou položku nápovědy vždy využívat menu „Nápověda“, mezi jednotlivými položkami nápovědy lze potom přepínat již v prohlížeči za předpokladu, že nainstalovaný prohlížeč tuto funkci podporuje.

Poslední položkou roletového menu je nabídka „O programu“ (Alt+O). Tato nabídka neobsahuje žádné podmenu, při této volbě se otevře nové okno, v němž se zobrazí některé informace o univerzitě, fakultě, programu a jeho tvůrci. Aby mohl uživatel CAN Setup v. 1.2 dále pracovat, je nutné toto okno nejprve zavřít.

Další možnosti ovládání jsou umístěny pod roletovým menu na tzv. tlačítkové liště pro rychlý přístup k většině funkcí aplikace. Jednotlivé typy tlačítek jsou sdruženy do skupin, tyto skupiny jsou od sebe odděleny separátory. Význam všech tlačítek je zobrazen jednak jejich grafickým motivem - ikonou, která je shodná s použitým symbolem u odpovídající funkce v roletovém menu a dále zobrazením tzv. plovoucí nápovědy, která se zobrazí, pokud kurzor myši na okamžik zastaví nad daným tlačítkem. Do první skupiny patří tlačítka příslušející funkčností k roletové nabídce „Soubor“, tlačítka mají funkce „Vymazat nastavení“, „Otevřít nastavení“, „Uložit nastavení“. Další je skupina pěti tlačítek funkčně odpovídajících položkám roletového menu „Akce“, tj. „Inicializace uzlu“, „Reset uzlu“, „Nastav vše“, „Spuštění uzlu“ a „Zastavení uzlu“. Stejně jako nabídka „Akce“, tak i těchto pět zmíněných tlačítek je při špatně zvoleném, nefunkčním či obsazeném sériovém portu

neaktivních, uživateli nepřístupných. Další možností na tlačítkové liště je volba sériového portu. Tato volba se nikde jinde v aplikaci nenachází a je bezpodmínečně nutná pro její správnou činnost. Po spuštění programu je v tomto poli zobrazena výzva „Zvolte sériový port“. V nabídce jsou obsaženy položky „COM 1“ až „COM 4“, protože většina IBM PC kompatibilních počítačů může být vybavena až čtyřmi sériovými porty. Pokud uživatel správně zvolí přístupný a funkční sériový port je informován ve stavovém řádku a zároveň jsou mu zpřístupněny položka "Akce" v roletovém menu a této položce odpovídající tlačítka na tlačítkové liště. V případě neúspěšné volby je zobrazeno chybové hlášení „COM X se nepodařilo otevřít“ a zmíněné funkce aplikace nejsou dále přístupné.

Posledním tlačítkem na liště je volba „Konec programu“, která má funkci jako stejnojmenná položka v nabídce „Soubor“.

Další částí programu jsou tzv. záložky. Jedná se o karty, mezi kterými lze přepínat, každá karta obsahuje další údaje. Tyto záložky mají názvy „Nastavení uzlu“, „Vysílací message objekty“ a „Přijímací message objekty“. Všechny tyto záložky obsahují editační pole s popisy, tlačítka pro kontextovou nápovědu a záložka „Nastavení uzlu“ navíc obsahuje ještě indikaci běhu programu v připojeném uzlu CAN, zobrazuje se na ní procentuální ukazatel průběhu při kompletním nastavování uzlu a text, na který pokud se klikne myší, dojde k zobrazení informací obdobně jako při použití položky roletového menu „O programu“. Editační pole odpovídají registrům řadiče sběrnice CAN, viz. příloha 10.2. Do každého editačního pole lze data zadávat jak klávesnicí, tak i myší. Při zadávání z klávesnice jsou akceptovány pouze znaky, které představují čísla v šestnáctkové soustavě, tj. znaky 0 až 9, A až F. U písmen A až F je jedno, zda uživatel zadává velká či malá písmena, automaticky jsou změněna na velká. Délka řetězce, který lze zapsat do každého editačního pole, je omezena na dva znaky, což odpovídá šestnáctkovému vyjádření čísla délky jeden bajt (8 bitů), maximální hodnota, kterou lze tedy zadat je FF, což dekadicky odpovídá hodnotě 255. Více znaků nelze do editačního pole zadat, při pokusu o zadání třetího znaku se předchozí dvojice automaticky přepíše. Pomocí myši lze za použití pravého tlačítka také vkládat text ze schránky. V tomto případě lze do editačního pole vložit jakýkoliv text. V případě vložení nestandardního řetězce nedovolí aplikace uživateli toto editační pole opustit do té doby, než je text v něm změněn na korektní formát.

Posledním prvkem použitým v aplikaci je stavový řádek, nacházející se na spodní straně okna. Tento stavový řádek je rozdělen do tří částí. V první části je uživatel informován o poslední akci, kterou provedl, resp. o jejím výsledku. V druhé části se v sekundových

intervalech zobrazuje aktualizovaný systémový čas a v poslední, třetí části, se zobrazuje systémový datum.

6.1.3 Komponenta pro sériové rozhraní

Protože Borland Delphi 5.0 ve své instalaci nenabízejí žádnou komponentu pro práci se sériovými porty, naskýtaly se dvě možnosti. První spočívala v naprogramování vlastní komponenty, což není cílem diplomové práce, a druhá, kdy bylo třeba hledat, zda neexistují nějaké veřejně publikované komponenty na Internetu. Podařilo se nalézt celkem dvě tyto volně šířitelné komponenty s názvy QCCom32 a CommPort. Vzhledem k tomu, že druhou jmenovanou se nepodařilo korektně nainstalovat do prostředí Delphi, padla volba na již zmíněnou komponentu QCCom32, jejímž autorem je Scott Pinkham, Bozeman MT, scottpinkham@yahoo.com.

Možnosti této komponenty se ukázaly jako plně dostačující. Komponenta umožňuje volbu libovolného sériového portu, nastavení jeho přenosové rychlosti, možnost využití hardwarového handshake, zobrazovat chyby vzniklé při používání sériového portu, nastavit si čekání na konkrétní řetězec ukončující libovolnou posloupnost znaků, dobu čekání na tuto posloupnost a proceduru, která se vykoná v případě, že tato posloupnost v daném čase nepřijde. Pro práci s bufferem sériového kanálu jsou definovány funkce čtení, zápisu, mazání, vyzvednutí, testování zda je port právě používán, počet přijatých znaků v bufferu. Jedinou nevýhodou byla funkce čekání na určitý konec posloupnosti, která se vykonala pouze pokud ona posloupnost v daném časovém horizontu nedorazila a nebyla současně definována žádná jiná, která by se naopak při tomto příjmu vykonala. Další nevýhodou bylo, že chybová hlášení byla plně lokalizována do anglického jazyka, což bylo v rozporu s koncepcí CAN Setup v. 1.2, který je plně lokalizován pro české prostředí. Proto došlo k drobnému zásahu do této knihovny a část chybových resp. varovných hlášení byla přeložena do češtiny, konkrétně se jednalo o chybová hlášení při otevírání portu.

6.1.4 Komunikace mezi PC a CAN Node v. 2.0

Komunikace probíhá po sériovém rozhraní RS232C v režimu Master - Slave, kde funkci Master přebírá PC, rychlostí 9600 Baud, poloduplexně, oběma směry, hardwarový handshake není použit. Přenášené slovo obsahuje jeden start bit, 8 datových bitů, žádnou paritu a jeden stop bit. Vzájemná komunikace probíhá na principu tzv. echo protokolu, kdy Master vždy čeká na odezvu Slave, tuto odezvu vyhodnotí a poté pokračuje v komunikaci. Tuto komunikaci lze rozdělit do dvou částí. První část tvoří krátké,

jednobajtové řídící příkazy pro připojený uzel CAN, druhou část reprezentuje vlastní předávání uživatelem definovaných hodnot mikrokontroléru uzlu CAN a jejich následná aplikace do registrů řadiče CAN.

6.1.5 Řídící příkazy

Tyto krátké jednobajtové příkazy reprezentují funkce „Inicializace uzlu“, „Reset uzlu“, „Spuštění uzlu“, „Zastavení uzlu“ a jsou reprezentovány znakem, jehož ASCII hodnota je uvedena v tab. 3. Vyslání příkazu je podmíněno spuštěním této akce použitím některého z příslušných ovládacích prvků aplikace popsaných v kapitole 6.1.2. O vykonání tohoto příkazu připojený uzel CAN informuje PC rovněž jednobajtovým číslem, avšak odlišným od příkazu, na který reagoval. Přehled řídících příkazů a odpovědí na ně je uveden v tab. 3.

Příkaz	Kód příkazu (hexa)	Odpověď (hexa)
Inicializace uzlu	E5	AA
Reset uzlu	CA	80
Spuštění uzlu	99	86
Zastavení uzlu	96	88

Tab. 3 Kódy řídících příkazů a odpovědí uzlu

6.1.6 Zakódování údajů z editačních polí

Uživatel má k dispozici celkem 92 editačních polí, které reprezentují registry řadiče CAN. Ne všechny je však nezbytně nutné využít, některé mohou tedy zůstat nevyplněny. I tato informace musí být při předávání údajů zahrnuta. Není možné tyto údaje vynechat, protože zároveň je nutné dodržet pevnou délku předávaného bloku dat, kde každé editační pole, resp. hodnota reprezentující toto pole, má své pevné místo. Pokud by byla prázdná editační pole vynechána, registry řadiče CAN by dostaly úplně jiné hodnoty než ty, které jim náleží. Zároveň je nutné respektovat, že datová paměť použitého mikrokontroléru AT89C52 je pouhých 256 bajtů, takže odvysílání celé posloupnosti již zmíněných 92 bajtů a dalších doplňujících informací najednou by zabralo téměř polovinu této paměti a nemusela by být dostačující pro další proměnné, které používá řídící program mikrokontroléru. Poslední podmínkou je, že ačkoliv nepoužité editační pole sice obsahuje prázdný řetězec, nelze tento prázdný řetězec reprezentovat po převodu prázdným znakem, vyslána musí být vždy

konkrétní hodnota s tím, že doplňující informace o tom, zda tato hodnota v řetězci není určena k použití je uložena někde jinde.

Při vyvolání akce „Nastav vše“ je postupně probíráno všech 92 editačních polí, textový řetězec reprezentující jejich hodnotu je předáván funkci, která jej převede na odpovídající číslo a zároveň je volána funkce, která sestavuje informace o platnosti do tzv. bajtů platnosti, které ukládá rovněž do pole. Na každých osm editačních polí tedy připadá jeden bajt platnosti, kde platnost je vyjádřena hodnotou bitu na pozici odpovídají pozici editačního pole, s krokem vždy po osmi editačních polích. Pokud je tedy editační pole platné, tj. že obsahuje neprázdný řetězec, je tento řetězec převeden na číslo, uložen do pole hodnot a na pozici odpovídající bitu platnosti v bajtu platnosti je uložena hodnota logické jedničky. V případě, že je editační pole prázdné, do pole hodnot je přidána nepoužitelná hodnota 66 hexa a bit v bajtu platnosti příslušející tomuto editačnímu poli je nastaven na hodnotu logická nula. Mechanizmus tvorby bajtů platnosti je uveden na obr. 18.

Editační pole č.	9	10	11	12	13	14	15	16
Obsah edit. pole (řetězec)	C9		88	76		AB	EC	
Hodnota edit. pole (hexa)	C9	66	88	76	66	AB	EC	66
Bit v bajtu platnosti č.2	1	0	1	1	0	1	1	0
Hodn. bajtu platnosti č. 2	B6 H							

Obr. 18 Vytvoření bajtu platnosti

6.1.7 Průběh nastavování

V kapitole 6.1.6 byl popsán algoritmus, jak z jednotlivých editačních polí dojde k převedení na číselné hodnoty, tedy vytvoření pole hodnot a pole bajtů platnosti. Jak bylo dále zmíněno, datová paměť mikrokontroléru je pouhých 256 bajtů, existuje tedy reálný požadavek na přenesení celé skupiny informací po částech, aby nedošlo ke zbytečnému zaplnění této paměti.

Komunikace tedy probíhá po částech, tzv. paketech. Každý paket má délku deset bajtů. První bajt je vždy pevně daný, obsahuje znak s ASCII hodnotou CC hexa. Mikrokontrolér uzlu CAN je tím informován o tom, že budou následovat další hodnoty, které bude třeba ukládat, mikrokontrolér se tedy přepne do režimu přijímání řetězce. Po úvodním znaku následuje osm znaků, které reprezentují ASCII hodnoty z pole hodnot a na posledním desátém místě je bajt platnosti. Schéma paketu je na obr. 19.

CC	data 1	data 2	data 3	data 4	data 5	data 6	data 7	data 8	platnost

Obr. 19 Struktura paketu s nastavujícími daty

Master, v tomto případě tedy PC, odešle tento první paket. V okamžiku, kdy je odeslán, přepne do režimu čekání. V té době již mikrokontrolér uzlu CAN celý tento paket posílá zpět. V případě, že tento paket dorazí zpět včas a v podobě, ve které byl vyslán, vysílá PC signál, který povoluje uzlu CAN přijaté hodnoty použít. Tento signál je reprezentován jednobajtovou hodnotou 55 hexa a potom se opět PC přepne do režimu čekání. Uzel CAN začne získané hodnoty zpracovávat, po zpracování si vyžádá další přísun hodnot jednobajtovým příkazem reprezentovaným hodnotou 66 hexa a pokud má PC další hodnoty k dispozici, uzlu je poskytne. Pokud však paket nedorazí zpět včas, nebo dojde k chybě při přenosu, PC svolení k použití nepošle, namísto toho posílá celý paket znovu. Pokud dojde k deseti chybám během přenosu, je nastavování přerušeno a uživateli se zobrazí hlášení o chybě. Takto se tedy postupně odešle všech 92 uživatelských hodnot. Datová část posledního dvanáctého paketu by však byla neúplná, dojde tedy k doplnění čtyřmi neplatnými hodnotami 66 hexa. Komunikaci při nastavování názorně ukazuje záznam uvedený v příloze 10.1.

6.1.8 Ukládání konfigurace do souboru

V kapitole 6.1.2 je zmíněna možnost uložení vytvořené konfigurace do souboru. Toto ukládání probíhá tak, že uživateli se po volbě „Uložit nastavení“ zobrazí standardní dialogové okno pro ukládání souboru, ve kterém uživatel zvolí adresář, do kterého chce soubor uložit a jméno souboru. Přípona za jménem souboru je pevně nastavena na *.ini.

Vlastní soubor je vytvořen jako soubor textový, bez jakékoliv hlavičky či kódování. Postupně jsou v pevném a neměnném pořadí po řádcích do tohoto souboru zapisovány hodnoty z editačních polí. V případě, že je editační pole prázdné, je do souboru uložen prázdný řádek. Nabízí se tedy další možnost, pokud uživatel zná přesné pořadí uložených hodnot, editovat tento soubor v libovolném textovém editoru, aniž by byl nucen spouštět aplikaci.

Načítání konfigurace ze souboru je řešeno postupem přesně opačným. Po volbě „Otevřít nastavení“ se uživateli zobrazí standardní dialogové okno pro otvírání souborů, ze kterého uživatel vybere konkrétní jméno souboru. Souborová maska je napevno nastavena na *.ini. Poté aplikace otevře zvolený soubor a po řádcích dochází k načítání a nastavování příslušných editačních polí.

6.2 Řídící program mikrokontroléru uzlu CAN

6.2.1 Funkce řídícího programu

Řídící program je uložen v programovatelné paměti mikrokontroléru AT89C52. Je napsán v jazyce C, v prostředí μVision2 od německé firmy Keil Elektronik. Do binárního kódu srozumitelného mikrokontroléru byl přeložen překladačem integrovaným do výše zmíněného vývojového prostředí. Do paměti mikrokontroléru je výsledný binární kód nahrán prostřednictvím programátoru ATMega od firmy MITE Hradec Králové.

Mezi základní funkce řídícího programu patří přijímání příkazů z nadřazeného systému (osobní počítač) prostřednictvím sériového kanálu, snímání hodnot z jednotlivých kanálů AD převodníku, vysílání těchto hodnot prostřednictvím řadiče sběrnice CAN, příjem zpráv ze sběrnice CAN a jejich zobrazování na LCD displej resp. LED diody. Pro větší přehlednost je zdrojový kód programu rozdělen do několika modulů. Kompletní zdrojový kód je uložen na přiloženém CD v adresáři */Source/CAN_Node*.

6.2.2 Nastavení použitých SFR

Pro správnou funkci řídícího programu je nejprve nutné nastavit napevno některé SFR (Speciální Funkční Registry) řídícího mikrokontroléru AT89C52.

Nejprve nastavení časovačů. Mikroprocesor AT89C52 obsahuje celkem tři na sobě vzájemně nezávislé čítače/časovače (č/č). U těchto č/č je nutné zvolit jejich pracovní mód, tj. zda budou pracovat jako osmibitové nebo šestnáctibitové, vypočíst časové konstanty, rozhodnout o tom, zda budou použity jako čítače nebo časovače.

Pro volbu pracovního módu čítače/časovače 0 a 1 je určen registr TMOD (adresa RAM mikrokontroléru 89 H). Složení registru je na obr. 20.

				časovač 1			
GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0

				časovač 0			

Obr. 20 Struktura registru TMOD

- GATE (hradlo) – řízení hradlování. Pokud je GATE = 1, potom č/č čítá při vstupu $\overline{INT1}$ = 1 resp. $\overline{INT0}$ = 1 a bitu TR1 = 1 resp. TR0 = 1, kde TR1 resp. TR0 jsou bity registru TCON. V tomto režimu je tedy čítač/časovač 0 resp. 1 ovládán nejen programově, ale současně také i pomocí vnějšího signálu přivedeného na vstup $\overline{INT1}$ resp. $\overline{INT0}$, kde tyto vstupy jsou alternativními funkcemi I/O pinů portu 3, konkrétně pinů P3.2 a P3.3. Je-li GATE = 0 je čítač/ časovač řízen pouze programově bitem TR0 = 1 resp. TR1 = 1.

- $C/\overline{T}$ (Counter/Timer) - volba čítač/časovač. Pokud je $C/\overline{T}$ = 0 potom je zvolen režim časovače, kdy je na čítací vstup přiváděn pouze hodinový signál mikroprocesoru vydělený vnitřním děličem kmitočtu hodnotou 12, tzn. 1/12 frekvence hodinového signálu. Je-li $C/\overline{T}$ = 1, je zvolen režim čítače vnějších událostí, který je zaznamenáván na vstupním I/O pinu procesoru (P3.2 resp. P3.3).
- M0, M1 – kombinací těchto bitů dojde ke zvolení pracovního módu čítače/časovače. Tyto módy jsou uvedeny v tab. 4.

M0	M1	mód
0	0	0
0	1	1
1	0	2
1	1	3

Tab. 4 Módy č/č

  - ♦ Mód 0 – čítač/časovač pracuje jako osmibitový, tvořen je registrem TH1 resp. TH0, kterému je předřazen pětibitový registr TL1 resp. TL0. Pokud dojde k přetečení čítače, je nastaven bit TF1 resp. TF0, který může být zdrojem přerušení.
  - ♦ Mód 1 – je obdobný módu 0 s tím rozdílem, že čítač/časovač je šestnáctibitový a je tvořen dvěma registry TH0 a TL0, resp. TH1 a TL1.
  - ♦ Mód 2 – v módu 2 pracuje čítač/časovač jako osmibitový s automatickým přednastavením. Obsah čítače je obsažen v registru TL0 resp. TL1, předvolba v registru TH0 resp. TH1. Pokud dojde k přetečení v registru TL0 resp. TL1, je do registru TL0 resp. TL1 automaticky uložena hodnota TH0 resp. TH1.
  - ♦ Mód 3 – v módu 3 je čítač/časovač 0 rozdělen na dva samostatné osmibitové čítače TH0 a TL0. Čítač TL0 využívá signály GATE, $C/\overline{T}$, TR0, $\overline{INT0}$ a TF0. Čítač TH0 je nastaven jako časovač a je ovládán pouze bitem TR1. Při přetečení se nastavuje pouze příznak TF1. Pracuje-li č/č 0 v módu 3, může být č/č 1 využit pouze pro generování rychlosti sériového kanálu event. využit se současným zákazem přerušení od tohoto č/č.

Pro potřeby řídícího programu byl zvolen pro oba č/č zvolen režim časovače, pro časovač 0 mód 2, pro časovač 1 mód 1. Časovač 0 je použit jako zdroj přerušení pro práci s AD převodníkem, časovač 1 je použit pro generování rychlosti sériového kanálu, popis je uveden

dále v části věnující se nastavení sériového kanálu. Nastavení je v programu realizováno příkazem:

```
TMOD = 0x21;
```

Tento příkaz lze nalézt ve funkci *main* obsažené v modulu *main.c*.

Rychlost časovače 0 je nastavena na 16 Hz, což umožňuje během jedné sekundy přečíst hodnoty ze všech analogových vstupů AD převodníku. Pro registry předvolby TH0 a TL0 platí tedy vztah:

$$[\mathrm{TH0,TL0}] = 65536 - \frac{f_{OSC}}{12 \cdot f_{T0}} = 65536 - \frac{11{,}0592 \cdot 10^6}{12 \cdot 16} = 7936 \approx \underline{\underline{[1F,00]H}}$$

Nastavení této předvolby je v programu realizováno voláním funkce *tinit0* z funkce *main*, obě tyto funkce jsou obsaženy v modulu *main.c*.

```
void tinit0 (void)
{
     TH0 = 0x1F;
     TL0 = 0x00;
}
```

Řízení čítačů/časovačů je prováděno registrem TCON (Timer/Counter Control). Jedná se o bitově adresovatelný osmibitový registr na adrese RAM 88 H. Rozložení jednotlivých bitů je znázorněno na obr. 21.

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Obr. 21 Struktura registru TCON

- TF1, TF0 (Timer Flag 0, 1) – příznakový bit přetečení čítače/časovače je automaticky hardwarově nastaven při přechodu z maximální hodnoty do nuly. V případě přechodu do obslužné rutiny přerušení je tento bit automaticky vynulován.
- TR1, TR0 (Timer Running 1, 0)– programově ovládaný bit spouštění a zastavování příslušného čítače/časovače. Pokud je v registru TMOD bit GATE = 1, potom o spuštění čítače/časovače rozhoduje vstupní signál $\overline{\mathrm{INT1}}$ resp. $\overline{\mathrm{INT0}}$.
- IE1, IE0 (Interrupt Enable 1, 0) – přijetí vnějšího přerušení. Bit IT1 resp. IT0 je nastaven při sestupné hraně nebo logické úrovni 0 na vstupu vnějšího přerušení $\overline{\mathrm{INT1}}$ resp. $\overline{\mathrm{INT0}}$

v závislosti na nastavení příslušného konfiguračního bitu IT1, IT0. Při přechodu do obslužné rutiny přerušení je tento bit automaticky vynulován.

- IT1, IT0 – konfigurace aktivace vnějšího přerušení. Pokud je IT1 = 1 resp. IT0 = 1 je žádost o přerušení aktivována sestupnou hranou signálu vnějšího přerušení $\overline{\text{INT1}}$ resp. $\overline{\text{INT0}}$. Je-li IT1 = 0 resp. IT0 = 0, je žádost o přerušení aktivována úrovní signálu $\overline{\text{INT1}}$ resp. $\overline{\text{INT0}}$ logická nula. Pokud je vnější signál v této úrovni delší dobu, může být přerušení vyvoláno několikrát za sebou.

Jelikož registr TCON je, jak již bylo výše zmíněno, bitově adresovatelný, v programu se spouštění a zastavování příslušného čítače/časovače a další povely řídí nastavováním konkrétního bitu v registru TCON.

Čítač/časovač 2 má k dispozici vlastní konfigurační registr s názvem T2CON (adresa RAM C8 H). Rozložení bitů v tomto registru je znázorněno na obr. 22.

TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/$\overline{\text{T2}}$	CP/$\overline{\text{RL2}}$

Obr. 22 Struktura registru T2CON

- TF2 (Timer Flag 2)– bit příznaku přetečení čítače/časovače 2. Bit je hardwarově nastaven při přechodu obsahu čítače/časovače z maximální hodnoty do nuly. Je nutné jej vynulovat programově. Pokud bude nastaven bit RCLK nebo TCLK, nedojde k nastavení příznaku TF2 při přetečení.
- EXF2 (External Flag 2)– vnější příznak čítače/časovače 2. Příznak se nastaví, pokud je EXEN2 v logické jedničce a pokud je detekována sestupná hrana na vývodu T2EX, což je alternativní funkce pinu P1.1. Pokud je povoleno přerušení od čítače/časovače 2, potom příznak EXF2 způsobí jeho vyvolání, přičemž se v příslušném podprogramu přerušení musí nulovat programově.
- RCLK (Receive Clocking) – povolovací bit synchronizace příjmu od sériového kanálu. Pokud je bit nastaven, je přenosová rychlost pro příjem dat ze sériové linky generována čítačem/časovačem 2. Je-li bit vynulován, k synchronizaci se použije čítač/časovač 1.
- TCLK (Transmit Clocking) – povolovací bit synchronizace vysílání sériového kanálu. Význam tohoto bitu pro vysílání po sériové lince je obdobný významu bitu RCLK.
- EXEN2 (External Enable 2) – bit povolující vnější ovlivnění čítače/časovače 2. Pokud je EXEN2 = 1 a čítač/časovač 2 není využit pro generování rychlosti sériového kanálu,

potom zachycení nové hodnoty nebo přednastavení čítače/časovače 2 nastane při sestupné hraně signálu na vývodu T2EX. Jestliže je EXEN2 = 0, potom jsou ignorovány události na vývodu T2EX.

- TR2 (Timer Running 2) – spouštění/zastavování čítače/časovače 2. Pokud TR2 = 1, je čítač/časovač spuštěn, je-li TR2 = 0, je zastaven.
- C/$\overline{T2}$ (Counter/Timer) - volba režimu, zda se bude č/č 2 požívat jako čítač nebo jako časovač. Pokud je C/$\overline{T2}$ = 1 je nastaven režim čítače vnějších událostí a události jsou sledovány při sestupné hraně signálu na vývodu T2, je-li C/$\overline{T2}$ = 0, jedná se o režim časovače.
- CP/$\overline{RL2}$ (Capture/Reload) - výběr režimu zachycení (Capture) nebo přednastavení (Reload)

Pracovní režim čítače, tj. použití bitů RCLK, TCLK a CP/$\overline{RL2}$ se volí podle tab. 5.

RCLK+TCLK	CP/$\overline{RL2}$	režim
0	0	16 bitový s aut. přednastavením
0	1	16 bitový záchytný
1	X	generátor přenosové rychlosti sér. kanálu

Tab. 5 Režimy č/č 2

Dalším registrem, který určuje vlastnosti č/č 2 je registr s názvem T2MOD (adresa RAM C9 H). Rozložení jednotlivých bitů v tomto registru je znázorněno na obr. 23.

-	-	-	-	-	-	T2OE	DCEN

Obr. 23 Struktura registru T2MOD

- T20E (Timer 2 Output Enable) - nastavením tohoto bitu lze využít alternativní funkci pinu P1.0 (T2), kdy tento pin bude použit jako výstupní, na kterém bude generován hodinový signál s programovatelnou frekvencí (Programable Clock Out).
- DCEN (Down Counter Enable) - pokud DCEN = 1, potom č/č čítá směrem nahoru i dolů, přičemž směr čítání je řízen stavem vstupního pinu T2EX (P1.1), T2EX = 1 znamená čítání nahoru, T2EX = 0 způsobí čítání dolů. Jestliže je DCEN = 0, č/č 2 čítá směrem nahoru.

Čítač/časovač 2 je využit jako zdroj hodinového signálu pro AD převodník s frekvencí 2,7 Mhz, je 16 bitový s automatickým přednastavením čítající nahoru, přičemž není použit jako generátor rychlosti sériového kanálu a nelze jej ovlivnit vnějším signálem. Pro registry předvolby RCAP2H a RCAP2L platí vztah:

$$[\text{RCAP2H}, \text{RCAP2L}] = 65535 - \frac{f_{OSC}}{4 \cdot f_V} = 65535 - \frac{11{,}0592 \cdot 10^6}{4 \cdot 2{,}7 \cdot 10^6} = \underline{\underline{[\text{FF}, \text{FE}]\text{H}}}$$

Nastavení časovače 2 je ve funkci *tinit2*, která je umístěna v modulu *main.c* a volána z funkce *main*.

```
void tinit2 (void)
{
      CLKIN=1;
      T2CON=0x04;
      T2MOD=0x02;
      RCAP2H=0xFF;
      RCAP2L=0xFD;
}
```

Pro práci se sériovým kanálem slouží registr SBUF (Serial Data Buffer). Ten je společný pro vysílání i pro příjem. Sériová linka může pracovat ve čtyřech módech, příslušný mód se volí nastavením registru SCON (Serial Control). Rozložení bitů v registru SCON je znázorněno na obr. 24.

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

Obr. 24 Struktura registru SCON

- SM1, SM0 (Serial Mode 1, 0) – kombinací těchto bitů se volí jeden ze čtyř módů sériového kanálu. Přehled těchto módů je uveden v tab. 6.

SM1	SM0	Mód	Přenos	Přenosová rychlost
0	0	0	synchronní, osmibitový	f_{OSC} / 12
0	1	1	osmibitový UART	časovač 1
1	0	2	devítibitový UART	f_{OSC} / 32, f_{OSC} / 64
1	1	3	devítibitový UART	časovač 1

Tab. 6 Módy sériového kanálu

- SM2 (Serial Mode 2) – bit, který povoluje vytvoření víceprocesorové komunikace v módu 2 a 3. Pokud je bit SM2 nastaven na hodnotu log. 1, potom nemůže být vyvoláno přerušení od příznaku čtení dat RI. V Módu 1 se může bit SM2 využít ke kontrole platnosti stop bitu. V módu 0 se bit SM2 nevyužívá.
- REN (Read Enable) – programově nastavitelný bit povolení příjmu, je-li REN = 1, je příjem povolen.
- TB8 – devátý datový bit při vysílání. Používá se v módech 2 a 3, nastavuje a nuluje se programově.
- RB8 – devátý datový bit pro příjem, využívá se v módech 2 a 3. V módu 1, pokud je nastaveno SM2 = 0, obsahuje RB8 přijatý stop bit. V módu 0 se RB8 nevyužívá.
- TI – příznak prázdného vysílacího registru. K jeho nastavení dochází při odvysílání osmého datového bitu v módu 0, event. při odvysílání stop bitu v ostatních módech. Příznak TI může být zdrojem žádosti o přerušení společně s příznakem RI a proto nemůže být při přechodu do příslušné rutiny přerušení nulován hardwarově, z důvodů rozlišení zdroje přerušení v přerušovací rutině, musí být příznak nulován programově.
- RI – příznak naplnění přijímacího registru. Je nastaven v okamžiku přijmutí osmého datového bitu v módu 0, uprostřed přijímání stop bitu v módu 1 nebo uprostřed přijímaní bitu RB8 v módech 2 a 3. Obdobně jako příznak TI není možno RI nulovat obvodově z důvodu rozlišení zdroje přerušení.

Pro komunikaci s nadřazeným osobním počítačem bez použití paritního bitu je použitelný pouze mód 1. Jedná se o osmibitový asynchronní režim s programovatelnou přenosovou rychlostí. Data jsou přijímána vstupem RxD, vysílána jsou výstupem TxD. Vysílá/přijímá se celkem 10 bitů, první je vždy nulový stop bit, poté následuje osm datových bitů a nakonec je odvysílán desátý jedničkový stop bit. Přenosová rychlost je v tomto režimu řízena četností přetečení čítače/časovače 1 a bitem SMOD v registru PCON. Tento bit zdvojnásobuje přenosovou rychlost určenou hodnotou čítače/časovače 1. Pro použitou přenosovou rychlost 9600 Baud platí při použití krystalu 11,0592 MHz pro předvolbu časovače 1 vztah:

$$TH1 = 256 - \frac{2^{SMOD} \cdot f_{OSC}}{384 \cdot rychlost} = 256 - \frac{2^0 \cdot 11{,}0592 \cdot 10^6}{384 \cdot 9600} = 253 \approx \underline{\underline{FD\,H}}$$

Nastavení časovače 1 je provedeno voláním funkce *tinit1* umístěné v modulu *main.c*.

```
void tinit1 (void)
{
     TH1 = 0xFD;
}
```

Nastavení sériového kanálu je provedeno voláním funkce *serial* ležící v modulu *main.c*.

```
void serial (void)
{
     SCON = 0x50;
     PCON = 0x00;
     REN = 1;
}
```

Poslední částí použitých SFR registrů jsou registry sloužící k nastavení přerušovacího systému mikroprocesoru AT89C52. Tento mikroprocesor má k dispozici celkem šest zdrojů přerušení, jejichž přehled je uveden v tab. 7. Priorita těchto zdrojů přerušení je uvedena sestupně, tzn. nejvyšší prioritu má vnější přerušení 0 a nejnižší přerušení od čítače/časovače 2.

Zdroj přerušení		Vektor přerušení
Příznak	Typ	
IE0	Vnější přerušení 0	0003 H
TF0	Čítač/časovač 0	000B H
IE1	Vnější přerušení 1	0013 H
TF1	Čítač/časovač 1	001B H
RI + TI	Sériový kanál	0023 H
TF2 + EXF2	Čítač/časovač 2	002B H

Tab. 7 Zdroje přerušení mikrokontroléru Atmel AT89C52

Registr pro povolení jednotlivých přerušení se nazývá IE (Interrupt Enable) a leží na adrese RAM A8 H. Rozložení jednotlivých bitů v registru je uvedeno na obr. 24.

EA	-	ET2	ES	ET1	EX1	ET0	EX0

Obr. 24 Struktura registru IE

- EA (Enable All) - globální povolení resp. zakázání používání přerušovacího systému. Pokud EA = 1, přerušovací systém je zapnut, jestliže EA = 0, systém přerušení je vypnut.
- ET2, ET1, ET0 (Enable Timer 2, 1, 0) - povolení resp. zakázání přerušení od č/č 2, 1, 0.
- ES (Enable Serial) - povolení resp. zakázání přerušení od sériového kanálu.
- EX1, EX0 (Enable External 1, 0) - povolení resp. zakázání přerušení od vnějšího zdroje 1, 0.

Pevně nastavenou prioritu jednotlivých zdrojů přerušení lze měnit pomocí registru IP (Interrupt Priority) na adrese RAM B8 H. Pokud je příslušný bit registru nastaven na hodnotu jedna, je tento zdroj přepnut do vyšší úrovně priority, pokud je nastaven na hodnotu nula, zůstává v platnosti jeho umístění v hierarchii standardní úrovně. Rozložení jednotlivých bitů v registru IP je znázorněno na obr. 25.

-	-	PT2	PS	PT1	PX1	PT0	PX0

Obr. 25 Struktura registru IP

- PT2, PT1, PT0 (Priority Timer 2, 1, 0) - zvýšení resp. ponechání úrovně priority přerušení od č/č 2, 1, 0.
- PS (Priority Serial) - zvýšení resp. ponechání úrovně priority přerušení od sériového kanálu.
- PX1, PX0 (Priority External 1, 0) - zvýšení resp. ponechání úrovně priority přerušení od externího zdroje přerušení 1, 0.

V programu jsou využívána přerušení od sériové linky a časovače 0. Při použití časovače 1 jako generátoru přenosové rychlosti sériového kanálu a časovače 2 jako zdroje hodinového signálu pro AD převodník, jsou přerušení od těchto časovačů zakázána. Externí zdroje přerušení jsou taktéž zakázány, protože nejsou využívány. Oproti standardní hierarchii jednotlivých zdrojů přerušení byla zvýšena priorita přerušení od sériového kanálu, protože během nastavování při současně spuštěném časovači 0 by mohlo dojít ke zvýšenému výskytu chyb během procesu nastavování a komunikace s nadřazeným osobním počítačem. Nastavení přerušovacího systému je v programu provedeno příkazy:

```
IE = 0x92;
IP = 0x10;
```

Tyto příkazy jsou obsaženy ve funkci *main* nacházející se v modulu *main.c*.

6.2.3 Operace s AD převodníkem

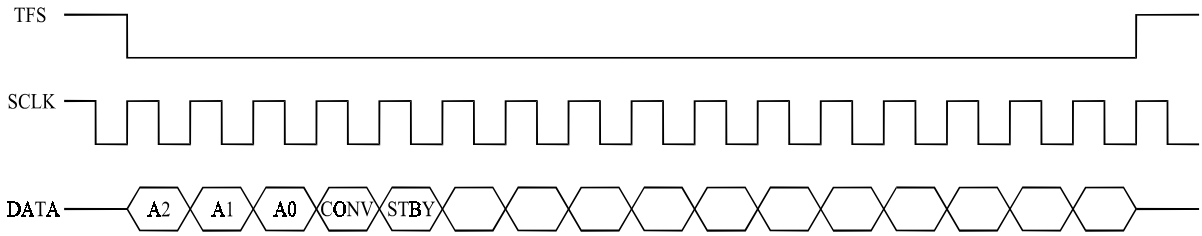
Pro komunikaci s AD převodníkem je použito celkem pět pinů portu P1, přičemž programově jsou ovládány pouze čtyři z nich, pátý je ovládán hardwarově, jedná se o generátor hodinového signálu pro AD převodník, kterým je alternativní funkce pinu P1.0 nazvaná T2. Princip činnosti je popsán v předchozí kapitole 6.2.2. Zbývající čtyři piny jsou ovládány programově a jejich názvy odpovídají signálům AD převodníku se kterými pracují. Piny portu jsou pojmenovány pomocí příkazu sbit.

```
sbit CLKIN = P1^0;
sbit TFS = P1^1;
sbit RFS = P1^2;
sbit DATA = P1^3;
sbit SCLK = P1^4;
```

Vlastní práce, tj. čtení hodnot z převodníku a zápis do převodníku probíhá v rutině přerušení od časovače 0. Jak již bylo zmíněno, frekvence tohoto časovače je 16 Hz, což při osmi analogových vstupech zaručuje přečtení všech těchto vstupů během jedné sekundy. Proces probíhá tak, že nejprve je do převodníku vyslána zpráva, obsahující číslo kanálu, ze kterého chceme číst hodnotu napětí a příkaz pro zahájení konverze. Při dalším příchodu do rutiny přerušení je nastavením řídících signálů zahájeno čtení dat, které převodník vysílá, přičemž rychlost vysílání je řízena asynchronním hodinovým signálem SCLK. Po přijetí kompletní zprávy od převodníku, je tato hodnota předána funkci, zajišťující její odvysílání po sběrnici CAN. Zároveň dochází k přepnutí na vyšší analogový vstup, aby při dalším cyklu mohlo dojít k přepnutí na vyšší analogový kanál a zahájení konverze na něm. Postupně jsou tedy procházeny analogové vstupy od vstupu č. 0 až do vstupu č. 7. Po přečtení vstupu č. 7 se opět přejde na vstup č. 0 a celý cyklus se opakuje.

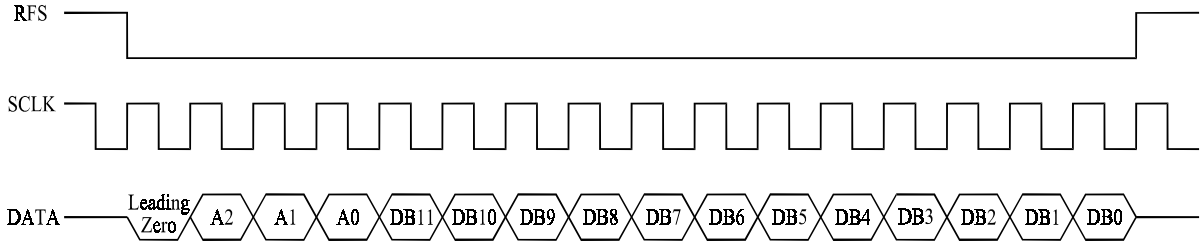
Zápis dat do převodníku probíhá následovně. Nejprve je připravena zpráva, která bude převodníku předána. Tato zpráva má celkem 16 bitů. První tři bity představují adresu analogového kanálu převodníku (bity A2,A1,A0), jehož hodnota napětí má být přečtena, tedy binárně vyjádřené číslo od 0 do 7. Čtvrtý bit (CONV) představuje příkaz pro zahájení konverze, pokud má tento bit hodnotu 1, je spuštěna konverze převodníku. Pátý bit (STBY) udává, jestli se má převodník po vykonání konverze zastavit. Na zbývajících 11 bitech nezáleží, jsou vyplněny nulami. Zapisování do převodníku je zahájeno nastavení bitu (řídícího signálu) TFS do logické nuly. Potom je bit DATA nastaven na logickou úroveň nejvyššího bitu připravené zprávy (A2). Následuje shození signálu SCLK z hodnoty logická jedna

na hodnotu logické nuly, čili sestupná hrana signálu. V tomto okamžiku převodník příjme hodnotu signálu DATA. Následuje posun zprávy doleva o jeden bit, přechod signálu SCLK z logické nuly do logické jedničky (vzestupná hrana), přiřazení hodnoty aktuálně nejvyššího bitu zprávy (A1) pinu DATA a opětovné nastavení sestupné hrany signálu SCLK. Celý tento proces se opakuje až do odvysílání všech 16 bitů, které zpráva obsahuje, stále však s hodnotou signálu TFS logická nula. Ukončení zápisu je provedeno navrácením signálu TFS na hodnotu logické jedničky. Celý mechanizmus zápisu je na obr. 26.



Obr. 26 Průběh zápisu dat do AD převodníku AD7890-10

Čtení dat z převodníku začíná náběžnou hranou signálu SCLK a shozením signálu RFS do logické nuly. Potom je vždy při sestupné hraně signálu SCLK zaznamenávána hodnota na pinu DATA. Hodnoty jsou postupně zasouvány zleva do proměnné napětí, která reprezentuje zprávu vyslanou převodníkem, jež obsahuje na prvním bitu úvodní nulu (Leading Zero), potom na třech bitech adresu analogového kanálu, ze kterého byla hodnota napětí získána, a následuje dvanáct datových bitů, reprezentujících změřené napětí, počínaje bitem s nejvyšším významem (DB11). Přijímání zprávy z převodníku je ukončeno nastavením signálu RFS na hodnotu logické jedničky. Postup čtení je přehledně uveden na obr. 27.



Obr. 27 Průběh čtení dat z AD převodníku AD7890-10

Ve zprávě, kterou vyslal převodník je zakódována hodnota napětí (DB11..DB0) na příslušném analogovém vstupu (A2..A0). Jedná se o binární reprezentaci změřené hodnoty napětí v rozsahu dvanácti bitů. Význam zakódované hodnoty napětí je vysvětlen v tab. 8.

Analogový vstup	Číslicové výstupní přechodné kódování
+FSR/2 - 1 LSB (9,995117 V)	011111111110…011111111111
+FSR/2 - 2 LSB (9,990234 V)	011111111101…011111111110
+FSR/2 - 3 LSB (9,985352 V)	011111111100…011111111101
AGND + 1 LSB (0,004883 V)	000000000000…000000000001
AGND (0,000000 V)	111111111111…000000000000
AGND - 1 LSB (-0,004883 V)	111111111110…111111111111
-FSR/2 + 3 LSB (-9,985352 V)	100000000010…100000000011
-FSR/2 + 2 LSB (-9,990234 V)	100000000001…100000000010
-FSR/2 + 1 LSB (-9,995117 V)	100000000000…100000000001

Tab. 8 Kódování hodnot napětí AD převodníkem AD7890-10

FSR (Full Scale Range = rozsah převodníku) = 20 V

N (počet bitů) = 12

$$\mathrm{LSB}\left(\text{rozlišovací úroveň}\right) = \frac{\mathrm{FSR}}{2^{N}} = \frac{20}{2^{12}} = 4{,}883\ \mathrm{mV}$$

Nejvyšší bit (DB11) tedy udává znaménko, pokud je roven jedné, napětí je záporné, pokud je nulový, napětí je kladné. Zbývajících 11 bitů (DB10…DB0) vyjadřuje při kladném napětí binární hodnotu napětí (pro přepočet potřeba vynásobit LSB), při záporném napětí je použita hodnota dvojkového doplňku.

6.2.4 Vysílání zpráv na sběrnici CAN

Veškeré operace související s řadičem CAN Intel AN82527 se vybavují pomocí příkazu PBYTE, který je v modulu *main.c* definován hned na počátku:

```
#define PBYTE ((unsigned char *)0x30000L)
```

Jedná se o definici makra pro zápis do externí paměti RAM. Řadič CAN obsahuje celkem 256 registrů, jejichž stav řídí vlastnosti uzlu, do kterých se ukládají přijatá data, data určená k vyslání, řídící a stavové registry. Mapa adres je uvedena v příloze 10.2.

Řadič CAN nabízí celkem 15 objektů zpráv (Message Objects, MO), kde každý tento objekt zpráv se vnějšímu pozorovateli může zdát jako samostatný uzel CAN, protože má svůj vlastní identifikátor pod kterým vystupuje při komunikaci. Těchto 15 objektů zpráv bylo přehledně rozděleno na objekty vysílací a na objekty přijímací. Protože AD převodník má

celkem osm analogových vstupů (V_{IN1}…V_{IN8}), tyto vstupy byly přiřazeny objektům zpráv s odpovídajícím číslem (MO1…MO8) a je pro správnou funkci uzlu nutné, aby byly uživatelem nastaveny jako vysílací. Také je nutné, aby délka vysílaných zpráv byla nastavena na dva bajty, poněvadž zpráva od AD převodníku obsahuje celkem šestnáct bitů, tedy dva bajty.

Každému objektu zpráv přísluší celkem patnáct registrů sloužících pro jeho řízení, určení identifikátoru, konfiguraci a data. Struktura objektu zpráv je zobrazena na obr. 28, přičemž bázové adresy jednotlivých objektů zpráv jsou uvedeny v příloze 10.2.

Adresa registru		Název registru
bázová adresa	+ 0	Control 0
	+ 1	Control 1
	+ 2	Arbitration 0
	+ 3	Arbitration 1
	+ 4	Arbitration 2
	+ 5	Arbitration 3
	+ 6	Mess. Conf.
	+ 7	Data 0
	+ 8	Data 1
	+ 9	Data 2
	+ 10	Data 3
	+ 11	Data 4
	+ 12	Data 5
	+ 13	Data 6
	+ 14	Data 7

Obr. 28 Struktura objektu zpráv

- Control 0, Control 1 – řídící registry objektu zpráv
- Arbitration 0…3 – registry určené pro identifikátor objektu zpráv
- Mess. Conf. – nastavení objektu zpráv
- Data 0…7 – registry určené pro práci s vysílanými resp. přijímanými daty

Bližší komentář týkající se konkrétního obsahu jednotlivých registrů lze nalézt v příloze 10.3 nebo jako nápovědu k programu CAN Setup v. 1.2, který je na přiloženém CD v adresáři */CAN_Setup*.

Postup vyslání zprávy je následující. V okamžiky, kdy je v rutině přerušení od časovače 0 získána nová hodnota napětí, zavolá se funkce *Transmit2CAN*, která nejprve příslušnému objektu zpráv nastaví v registru Control1 příznak zadávání nových dat (CPUUpd, CPU Update). Potom jsou do registrů Data0 (nižší bajt zprávy od převodníku) a Data1 (vyšší bajt zprávy od převodníku) uloženy získané hodnoty. Následovně je smazán příznak CPUUpd a je nastaven příznak požadavku vyslání na sběrnici (TxRqst, Transmit Request). Tento příznak je po úspěšném odvysílání poté automaticky řadičem smazán. Tím došlo k odvysílání zprávy.

Postup uvedený v předchozím odstavci je však platný pouze pro objekty zpráv 4 až 8, u kterých dochází k cyklickému vysílání. První čtyři objekty zpráv odpovídající analogovým vstupům AD převodníku V_{IN1}…V_{IN4} jsou odvysílána pouze na žádost jiného uzlu (Remote Request). Postup přípravy zprávy je pro tyto objekty popsán v předchozím odstavci. Jediným rozdílem je to, že těmto objektům není nastaven příznak TxRqst, čímž nedojde k jejich okamžitému odvysílání. Pokud dorazí žádost o data, nastaví se řadiči příznak RmtPnd, řadič CAN automaticky tato data odešle z příslušných datových registrů. Příznak RmtPnd je po odvysílání zprávy řadičem automaticky smazán.

6.2.5 Příjem zpráv ze sběrnice CAN

Z celkem patnácti objektů zpráv, které jsou k dispozici zbývá pro příjem zpráv sedm objektů, pro příjem jsou tedy určeny objekty s čísly od devíti do patnácti. Zde se také ukazuje výhoda tohoto přehledného uspořádání, kde spodních osm objektů zpráv je určeno pro vysílání a horních sedm pro příjem, poněvadž objekt zpráv 15 je poněkud specifický tím, že je určen pouze k přijímání dat. Pro správnou funkci je tedy nutné, aby uživatel nastavil tyto objekty zpráv (MO9…MO15) jako přijímací.

K vlastnímu přijímání dochází v nekonečné smyčce hlavního programu, kdy je neustále dokola testován registr Control1 příslušného objektu zpráv na příznak přijetí nových zpráv (NewDat, New Data). Pokud je tento příznak nastaven, data jsou přečtena pro další zpracování a příznak NewDat je smazán.

6.2.6 Zobrazování přijatých dat na LCD displej

Vlastní komunikace s řadičem displeje probíhá tak, že jsou mu posílány ASCII hodnoty jednotlivých znaků a za pomoci řídících signálů displej rozpoznává zda jsou posílána data nebo instrukce. Tento displej nabízí dva režimy, čtyřbitový a osmibitový. V kapitole 5.2.5 (připojení LCD) je uvedeno, že displej je zapojen v osmibitovém režimu. Toto propojení je jednodušší a rychlejší z toho důvodu, že data nemusí být posílána ve dvou krocích, nejprve dolní čtyři bity a poté horní čtyři bity. Ke komunikaci ještě přísluší tři řídící signály, kterými je displej ovládán. Tyto signály jsou nazývány RS, RW, E a symbolicky jsou přiřazeny bitům portu P1.5 (RS), P1.6 (RW) a P1.7 (E). Veškeré funkce pro práci s displejem jsou v modulu *lcd.c*, kterému navíc přísluší hlavičkový soubor *lcd.h*. Tyto soubory je možné najít na přiloženém CD v adresáři */Source/CAN_Node*. Vlastní zápis na LCD provádí funkce *writeLCD,* které je vždy předán kód znaku, který chceme displeji předat. Tato funkce nejprve nastaví řídící signál E (Enable), jímž displej aktivuje. Potom je na P2 nastaven kód znaku a zavolána funkce *delayLCD*, která dává displeji čas potřebný k přijetí a zpracování dat. Nakonec je řídící signál E vynulován.

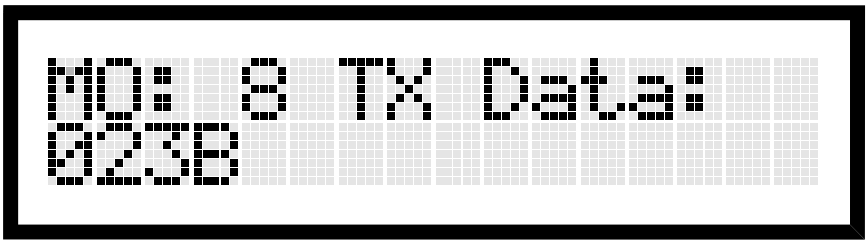
Aby bylo možno na LCD displej cokoliv zobrazovat je třeba jej nejprve inicializovat. Tuto inicializaci má za úkol funkce *initLCD,* která je volána po spuštění programu. Nejprve šestkrát zavolá funkci *delayLCD*. Zmíněné opatření je proto, aby spolehlivě došlo k ustálení hodnot napětí, což je nezbytně nutné pro správnou funkci displeje. Poté následují inicializační příkazy. Nejprve je zapsána pomocí funkce *writeLCD* hodnota 38 H. Tato hodnota reprezentuje funkční nastavení na osmibitový režim komunikace, režim zobrazování jako dvouřádkový (Dual-Line) a režim zobrazování znaků v matici 5 x 7 bodů. Druhým zaslaným příkazem je hodnota 0C H, jež displej zapíná, vypíná blikání kurzoru a kurzor samotný. Třetím a zároveň posledním inicializačním příkazem je hodnota 06 H určující automatické posouvání znaků doleva a inkrementaci adresového čítače AC (Address Counter). V tomto okamžiku je displej schopen přijímat a zobrazovat znaky. Více informací o nastavování displeje je možné najít v publikaci [1] uvedené v kapitole 9.

Výpis textů na displej po znacích by bylo značně nevýhodné, proto je definována funkce *WriteTextLCD* do níž jako parametry vstupují ukazatel na textový řetězec, který chceme vypsat, a délka tohoto řetězce. Funkce potom postupně řetězec prochází a po jednotlivých znacích jej předává funkci *writeLCD* zajišťující vlastní předání řadiči displeje.

Kromě již zmíněných funkcí jsou definovány ještě další, jako funkce pro přepnutí na nový řádek displeje, tj. funkce *NewLineLCD* (zapisuje hodnotu C0 H), funkce pro smazání

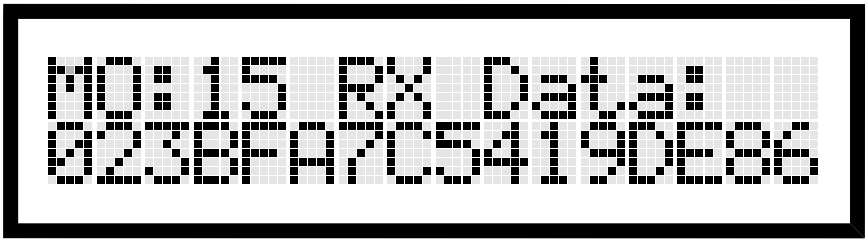
displeje nazvaná *clearLCD* (posílá hodnotu 01 H) a ještě funkce pro navrácení kurzoru na začátek řádku *CursorHomeLCD* (předává symbol 02 H).

Vlastní vypisování přijatých resp. vysílaných zpráv probíhá pomocí funkce *porovnani* resp. *porovnani2*. Tyto funkce zjistí od kterého objektu zpráv požadavek na vypsání pochází a zda je pomocí klávesnice povoleno vypisování na displej právě tohoto objektu zpráv. V jednom okamžiku lze totiž zobrazovat pouze údaje o jednom objektu zpráv. Bližší popis funkcí pro volbu zobrazování pomocí klávesnice je popsán v kapitole 6.2.8. Toto opatření bylo přijato z důvodu malého počtu znaků, které lze na displej najednou vypsat, tj.2 řádky po 16 znacích. V případě vypsání dat z vysílacího objektu zpráv vypadá text na displeji jak je naznačeno na obr. 29.



Obr. 29 Výpis vysílaných zpráv na LCD displej

Zkratkou „MO“ je míněn objekt zpráv, za dvojtečkou je uvedeno číslo objektu zpráv. Výraz „TX“ naznačuje, že tento objekt je nastaven jako vysílací a řádek ukončuje text „Data:“. Vysílaná data jsou zobrazena v hexadecimálním formátu. Tento formát byl zvolen spíše kvůli přijímacím objektům. Zatímco vysílaná data jsou dvoubajtová, přijímaná data mohou být až osmibajtová, což představuje šestnáct znaků a přijatá data se tak přesně vejdou na šestnáct míst druhého řádku. Další odlišností při zobrazování přijatých dat je nahrazení textu „TX“ textem „RX“, což symbolizuje přijatá data. Možný vzhled displeje při zobrazování přijatých dat je na obr. 30.



Obr. 30 Výpis přijatých dat na LCD displej

Aby bylo možné zobrazit přijatá data na displej, musí být nejprve provedena jejich konverze z podoby hexadecimálního čísla na text. Tuto konverzi provádí funkce *zobrazeni* resp. *zobrazeni2*. Využívá se vlastnosti ASCII tabulky, kde znaky pro čísla a písmena jsou posunuta pouze o konstantu. Z toho důvodu stačí číslům z intervalu 0…9 přičíst hodnotu 48, k číslům A…F hodnotu 55 a takto získané číslo odeslat k vypsání.

6.2.7 Zobrazování přijatých dat na LED diody

Vzhledem k nepříliš vysokým možnostem zobrazování zpráv na osm LED diod je tato funkce spíše doplňková, byla však důležitá během vývoje a testování zařízení. Vzhledem k tomu, že diody jsou připojeny k rozšiřujícímu portu P2 řadiče CAN a tento port by v původním návrhu zařízení zůstal jinak nevyužit, byly tyto LED diody ponechány. V případě volby zobrazování objektu zpráv 14 nebo 15, je na LED diody zobrazován nejnižší bajt obdržené zprávy. Vzhledem k tomu, že u obou portů řadiče CAN se musí nastavit zda jsou vstupní nebo výstupní, je pro správnou funkci nutné, aby uživatel nastavil konfigurační registr P2CONF tohoto portu na hodnotu FF H, která určuje tento port jako výstupní. Více informací o nastavování portů lze nalézt v příloze 10.3.7 nebo v nápovědě programu CAN Setup v. 1.2.

6.2.8 Volba zobrazované zprávy klávesnicí

Jak již bylo zmíněno dříve, je přehledné zobrazení všech údajů současně na displej nemožné a zároveň při rychlém střídání zobrazovaných informací by docházelo ke špatnému zachycení uživatelem. Proto je uživateli k dispozici klávesnice se šestnácti tlačítky pro volbu aktuálního zobrazování zprávy. Význam jednotlivých tlačítek je uveden v tab. 9.

Tlačítko	Volba zobrazení	Tlačítko	Volba zobrazení
0	zastavení zobrazování	8	Message Object 8
1	Message Object 1	9	Message Object 9
2	Message Object 2	A	Message Object 10
3	Message Object 3	B	Message Object 11
4	Message Object 4	C	Message Object 12
5	Message Object 5	D	Message Object 13
6	Message Object 6	#	Message Object 14
7	Message Object 7	*	Message Object 15

Tab. 9 Význam tlačítek klávesnice pro volbu zobrazované zprávy

Všechny funkce pro obsluhu klávesnice jsou v modulu *keypad.c*, kterému přísluší hlavičkový soubor *keypad.h*. Klávesnice je připojena k rozšiřujícímu portu P1 řadiče CAN a to tak, že spodní čtyři bity tohoto portu musí být definovány jako vstupní a horní čtyři bity jako výstupní. Proto je nutné, aby uživatel pro správnou funkci klávesnice nastavil konfigurační registr tohoto portu na hodnotu F0 H.

V nekonečné smyčce hlavního programu je volána funkce *read_keypad*. Ta testuje port a v případě, že je stisknuté některé tlačítko, vrátí jeho hodnotu. Mechanismus testování spočívá v tom, že na vstupních bitech portu je hardwarově nastavena hodnota logické jedničky. Na nejnižší bit výstupní části je přivedena logická nula, která se v případě stisknutého tlačítka v tomto sloupci přenese na vstupní část port, což je zaznamenáno a testování se ukončí. V opačném případě se logická nula posune o jeden krok doleva. Tento postup proběhne pro všechny čtyři výstupní bity, čímž dojde k otestování všech tlačítek. Celý cyklus v případě nestisknutého tlačítka je přehledně zobrazen v tab. 10

Krok testu	Stav registru P1OUT	Test
1	1110XXXX	sloupec 1
2	1101XXXX	sloupec 2
3	1011XXXX	sloupec 3
4	0111XXXX	sloupec 4

Tab. 10 Testování stisku tlačítka klávesnice

Díky maticovému uspořádání klávesnice má tedy každé tlačítko svůj kód. Tento kód je poté funkcí *decode_keypad* převeden na číslo od 0 do 15 a uložen do proměnné, která odpovídá zobrazovanému objektu zpráv.

6.2.9 Nastavování vlastností uzlu pomocí sériové linky

V kapitole 6.1.7 je popsán průběh nastavování vlastností uzlu po sériové lince z hlediska nadřazeného systému, tj. osobního počítače a použitý echo protokol. Na straně uzlu CAN probíhá proces s použitím generování přerušení od sériového kanálu. Toto přerušení může mít však dva zdroje, od přijatého bajtu a od odvysílaného bajtu, proto je potřeba při vstupu do rutiny přerušení nejprve rozlišit zdroj přerušení. Rutina přerušení je umístěna v modulu *main.c*.

V případě, že přerušení je generováno příchozím bajtem, je tento bajt porovnáván s množinou řídících příkazů uvedených v tab. 11. Pokud je příkaz rozeznán, odpovídající

kroky jsou provedeny a zároveň je zpět posláno potvrzení o vykonání příkazu. Seznam příkazů, jejich význam, provedená akce a odpověď uzlu jsou uvedeny v tab. 11.

Příkaz	Kód příkazu (hexa)	Akce	Kód odpovědi (hexa)	Odpověď
Inicializace uzlu	E5	-	AA	Uzel připojen
Reset uzlu	CA	*resetuzlu ();*	80	Uzel resetován
Spuštění uzlu	99	TR0=1; run=1;	86	Program spuštěn
Zastavení uzlu	96	TR0=0;run=0;	88	Program zastaven
Začátek řetězce	CC	nastavovani=1;	-	Obdržený řetězec
Svolení k použití dat	55	*nastav ();*	66	Žádost od další data

Tab. 11 Řídící příkazy, akce a odpovědi

V případě příkazu inicializace uzlu se jedná pouze o odpověď. Význam tohoto příkazu spočívá pouze v otestování správného a funkčního propojení mezi uzlem CAN a PC. Při příkazu k resetu uzlu je zavolána funkce *resetuzlu*, která na jednu dobu časovače 0, tj. na cca. 62,5 ms aktivuje signál reset pro řadič CAN. Pokud dorazí příkaz ke spuštění uzlu, je spuštěn časovač 0 a pomocné proměnné *run* je přiřazena hodnota 1. Tato pomocná proměnná slouží pro vykonávání hlavního programu v nekonečné smyčce. Instrukce k zastavení uzlu se chová přesně opačně nežli předchozí příkaz, zastaví časovač 0 a proměnné *run* přiřadí hodnotu 0. Dalším příkazem v pořadí je oznámení o tom, že bude následovat nastavovací posloupnost. V tomto případě bude devět dalších příchozích hodnot ukládáno do bufferu. Ve chvíli, kdy dorazí poslední desátá hodnota, je celý tento buffer posílán v nezměněném pořadí zpět. Odpovědí by mělo být svolení k použití právě odvysílaných hodnot. Při příchodu tohoto svolení je zavolána funkce *nastav*, která se znalostí pořadí přijatého paketu provede podle posledního bajtu v řetězci, označovaného jako bajt platnosti, dekódování celého řetězce a předání získaných hodnot příslušným registrům řadiče CAN. Na konci nastavovací funkce je vyslána žádost o další data.

Přerušení vygenerované od odvysílaných dat má význam pouze při zpětném vysílání obdrženého řetězce, kdy je odvysílán další bajt.

Pokud po sériové lince dorazí nějaký jiný příkaz, než jaký je uveden ve výčtu v tab. 11, je tento příkaz ignorován a není na něj reagováno.

6.2.10 Chování uzlu po zapnutí

Po připojení uzlu na napájecí napětí se nejprve rozsvítí LED dioda signalizující připojené napájení. Na první řádek displeje se vypíše nápis „TU v Liberci, FM“ a na druhý řádek se vypíše „CAN Node v. 2.0“. Poté je z paměti programu nahrána předvolená konfigurace pro řadič Intel AN82527. Po nahrání této konfigurace již je uzel připraven plnit všechny své funkce. Přenosová rychlost je 100 kBaud, vzorkovací bod 75 %. Přednastavená konfigurace uzlu je uvedena v tab. 12.

Objekt zpráv	Identifikátor [hexa]	Směr	Hodnota	Délka dat [B]
1	7F8	vysílání (RMT)	V_{IN1}	2
2	7F9	vysílání (RMT)	V_{IN2}	2
3	7FA	vysílání (RMT)	V_{IN3}	2
4	7FB	vysílání (RMT)	V_{IN4}	2
5	7FC	vysílání (cyklicky)	V_{IN5}	2
6	7FD	vysílání (cyklicky)	V_{IN6}	2
7	7FE	vysílání (cyklicky)	V_{IN7}	2
8	7FF	vysílání (cyklicky)	V_{IN8}	2
9	3F8	příjem		2
10	3F9	příjem		2
11	3FA	příjem		2
12	3FB	příjem		2
13	3FC	příjem		2
14	3FD	příjem		1
15	3FE	příjem		1

Tab. 12 Přednastavená konfigurace uzlu CAN

7. Ověření realizace pomocí mikropočítače C167CR

7.1 Stručný popis vývojového systému C167CR

Systém, jehož jádrem je mikroprocesor Infineon C167CR-LM, je výrobkem anglické firmy Hitex Ltd. Tento mikropočítač je rozšířen deskou EVA167 vybavenou uživatelskými vstupy a výstupy. K dispozici je LCD displej Hitachi se šestnácti znaky ve dvou řádcích, šestnáctitlačítková maticová klávesnice, pět potenciometrů, deset LED diod, osm DIL přepínačů pro volbu adresového prostoru paměti, vstupní a výstupní analogový filtr a miniaturní reproduktor pro generovaní zvuků. Pro ověření činnosti navrženého uzlu CAN je nejdůležitější přítomnost řadiče sběrnice CAN s až patnácti objekty zpráv podporující všechny funkce potřebné pro práci se sběrnicí CAN. Posledním důležitým obvodem je již známý budič sběrnice CAN Philips PCA82C250.

7.2 Program pro C167CR

Mikropočítač může být programován jak v jazyce C, tak v assembleru. Pro větší přehlednost byl zvolen jazyk C. Zdrojový text programu lze nalézt na přiloženém CD v adresáři */Source/c167*.

Výměna dat po sběrnici CAN probíhá rychlostí 100 kBaud při vzorkovacím bodu 75 % a použití standardního 11 bitového identifikátoru. Posílány jsou jednobajtové a dvoubajtové zprávy. Ke sběrnici CAN je připojen jeden mikropočítač C167CR a CAN Node v. 2.0.

7.2.1 Nastavení vlastností řadiče CAN

Nastavování vlastností probíhá v modulu *can.c*. Nejprve je nastavena přenosová rychlost a vzorkovací bod.

```
#define BRP         9
#define TSEG1       5
#define TSEG2       2
#define SJW         2
```

Poté jsou jednotlivým objektům zpráv přiřazeny identifikátory, objekty jsou nastaveny jako vysílací resp. přijímací a jsou validovány. Toto nastavení probíhá ve funkci *initialise_CAN* umístěné v modulu *can.c*. Nastavení jednotlivých objektů ukazuje tab. 13.

Objekt zpráv	Identifikátor [hexa]	Funkce
1	3F8	vysílání 2 B hodnoty od potenciometru 0
2	3F9	vysílání 2 B hodnoty od potenciometru 1
3	3FA	vysílání 2 B hodnoty od potenciometru 2
4	3FB	vysílání 2 B hodnoty od potenciometru 3
5	3FC	vysílání 2 B hodnoty od potenciometru 4
6	3FD	vysílání 1 B generované hodnoty (inkr. 0 ÷ 255)
7	3FE	vysílání 1 B generované hodnoty (Knight-Rider efekt)
8	7F8	vysílání požadavku a příjem 2 B hodnoty (V_{IN1})
9	7F9	vysílání požadavku a příjem 2 B hodnoty (V_{IN2})
10	7FA	vysílání požadavku a příjem 2 B hodnoty (V_{IN3})
11	7FB	vysílání požadavku a příjem 2 B hodnoty (V_{IN4})
12	7FC	příjem 2 B hodnoty (V_{IN5})
13	7FD	příjem 2 B hodnoty (V_{IN6})
14	7FE	příjem 2 B hodnoty (V_{IN7})
15	7FF	příjem 2 B hodnoty (V_{IN8})

Tab. 13 Struktura uspořádání objektů zpráv v testovacím programu

7.2.2 Vysílání dat na sběrnici CAN

Vysílání je zajištěno v rutině přerušení od časovače 1. V této rutině jsou získány hodnoty natočení potenciometrů z jednotlivých kanálů AD převodníku, tyto hodnoty jsou přiřazeny příslušným objektům zpráv a odvysílány. Dále jsou generovány hodnoty délky 1 B. První takto generovanou hodnotou je inkrementace čísla v rozsahu 0 ÷ 255, vždy s krokem po jedné v každém spuštění přerušovací rutiny. Tato hodnota je vždy odvysílána. Při dosažení čísla 255 se začíná načítat znovu od nuly. Druhou generovanou hodnotou je tzv. Knight-Rider efekt, kdy jednobajtové číslo obsahuje pouze jednu log. jedničku a ta je posouvána zprava doleva, při dosažení nejvyššího bitu se obrátí směr posouvání až do dosažení nejnižšího bitu.

V rutině přerušení ještě dochází k vysílání žádostí o data od přijímacích objektů zpráv 8 až 12. V každém proběhnutí rutiny přerušení je odvysílána žádost jednoho objektu, vysílání se cyklicky opakuje.

7.2.3 Příjem zpráv ze sběrnice CAN

Příjem zpráv probíhá v nekonečné smyčce funkce *main* umístěné v modulu *main.c*. Všechny přijímací objekty jsou testovány na příznak nových dat, pokud je tento příznak

nastaven, jsou příslušná data vyzvednuta. Vzhledem k tomu, že je známa struktura přijatých dat, je z těchto dat nejprve získána hodnota analogového vstupu desky CAN Node v. 2.0 a poté dochází k přepočtu získané hodnoty na napětí. Takto přepočtená data jsou zobrazena na displej. Informace jež má uživatel k dispozici označují objekt zpráv, od kterého data pochází, číslo analogového vstupu desky CAN Node v. 2.0 a získanou hodnotu napětí.

7.3 Vyhodnocení návrhu

Při testování návrhu nebyly pozorovány jakékoliv problémy s komunikací po sběrnici CAN, vysílaná i přijímaná data byla zobrazována správně. Překvapením však byla práce AD převodníku AD7890-10 na desce CAN Node v. 2.0. Tento převodník vykazoval hodnotu napětí cca. 1,4 V i v případě, že jeho vstupy byly plovoucí, tj. bez připojení na zdroj napětí. Pokud byl zdroj napětí připojen, změřená hodnota odpovídala řádově danému napětí, byla však velice rozkmitaná a to v řádu nejnižších cca. 4 - 5 bitů, což odpovídá hodnotě napětí až 0,16 V. Vzhledem k tomu, že z obdržených dat na straně C167CR byla vždy správně získána i adresa analogového kanálu převodníku AD7890-10, špatné získávání hodnot z převodníku bylo vyloučeno, chyba tedy pravděpodobně souvisela se špatnou činností zmíněného převodníku.

Na straně CAN Node v. 2.0 byla všechna data přijatá data zobrazována korektně, došlo i k otestování přijetí zprávy o délce dat 8 B, kdy zobrazení takto přijaté hodnoty zabralo celý druhý řádek displeje. Zprávy přijímané objekty zpráv 14 a 15 byly zobrazovány jak na displej tak na LED diody (inkrementace, Knight-Rider efekt) a tyto hodnoty si vzájemně vždy odpovídaly. Přepínání zobrazení jednotlivých zpráv pomocí klávesnice také fungovalo bez jakýchkoliv problémů.

Změna parametrů uzlu CAN z PC prostřednictvím linky RS232 byla testována záměnami identifikátorů jednotlivých objektů zpráv, přičemž uživatelsky zvolené nastavení vždy odpovídalo následné funkci uzlu.

8. Závěr

8.1 Chybovost během nastavování vlastností uzlu prostřednictvím RS232

Naskýtá se otázka, jestli není řešení pomocí echo protokolu popsaného v předchozích kapitolách 6.1.5, 6.1.6 a 6.1.7 příliš zbytečné a komplikované, zda-li by nebylo jednodušší a rychlejší data do uzlu CAN pouze posílat bez zpětné kontroly. I když komponenta Timer, v jejíž události OnTimer probíhá výměna dat, běží na vlastním vlákně (Thread), operační systém Microsoft Windows umožní přístup k bufferu sériového kanálu až ve chvíli, kdy má k dispozici dostatek volných systémových prostředků. Dochází-li tedy ke krátkodobému špičkovému vytěžování systému, u staršího méně výkonného počítače s procesorem Intel Pentium 100 stačilo v rychlém sledu několikrát po sobě minimalizovat a maximalizovat okno libovolné aplikace, systém Windows zabrání komponentě QCCom32 v přístupu k bufferu sériového portu a tím pádem komponenta načte špatnou hodnotu, nedojde k vyzvednutí bufferu. To je samozřejmě vyhodnoceno jako chyba. Pokud nebyl systém vytěžován žádným dalším způsobem, proběhla výměna desetibajtového paketu bez chyby řádově při desetitisících přenosech tam i zpět a to i při rušivých vlivech okolních elektrospotřebičů. Toto číslo by samozřejmě bylo mnohem méně příznivé, pokud by měl propojovací sériový kabel větší délku, nežli je délka použitá, tj. cca. 1,5 m nebo při použití vyšších přenosových rychlostí než stávajících 9600 Baud. Z těchto důvodů se tedy jeví použití echo protokolu jako opodstatněné, zejména na méně výkonných počítačích.

8.2 Možná optimalizace hardwarového návrhu

Řadič sběrnice CAN Intel AN82527 nabízí více způsobů propojení s řídícím mikrokontrolérem, viz. kapitola 5.1.2. Zvolené osmibitové multiplexované propojení je vhodným kompromisem mezi počtem obsazených pinů a rychlostí, kterou lze k datům přistupovat. Pokud by bylo nutné snížit počet propojovacích vodičů a tím i obsazených pinů mikrokontroléru, lze ještě využít sériové propojení pomocí tří vodičů, snížila by se však celková rychlost získávání dat z řadiče CAN.

Možná úspora čtyř pinů portu P2 mikroprocesoru se nabízí při zapojení displeje ve 4 bitovém režimu. Vzhledem k tomu, že v původním návrhu uzlu se pro takto uspořené piny nenašla žádná funkce, bylo ponecháno 8 bitové propojení s displejem, protože je dvakrát rychlejší nežli 4 bitové.

8.3 Hierarchie řídícího programu

Nabízí se otázka, jestli by nebylo možné nalézt lepší hierachické uspořádání řídícího programu mikrokontroléru. Současné uspořádání, kdy vysílání je zařízeno v rutině vygenerovaného přerušení od časovače 0 a příjem ve smyčce hlavního programu, má vyšší prioritu vysílání zpráv, příjem může být kdykoliv přerušen právě časovačem 0. Zároveň je příjem zpráv pomalejší, protože je nutné neustále cyklicky číst příznak nových dat v příslušných registrech řadiče CAN. Možným řešením tohoto problému by bylo využití signálu externího přerušení generovaného řadičem CAN. V tuto chvíli by se však změnilo pořadí priorit a vyšší prioritu by v programu mělo čtení přijatých zpráv. Protože však v návrhu bylo počítáno přednostně s vysíláním zpráv, nebylo toto řešení uznáno za vhodné. Dalším možným řešením by ještě bylo přepnout přerušení od časovače 0 do režimu zvýšené priority. To však naráží na skutečnost, že v tomto režimu již je přerušení od sériového kanálu, tedy časovač 0 by měl vyšší prioritu nežli sériový kanál a během procesu nastavování by mohlo docházet ke zvyšování počtu chyb. Tyto problémy by bylo možné odstranit tím způsobem, že během nastavování by byly zakázány všechny zdroje přerušení (externí, časovač 0), s vyjímkou přerušení od sériového kanálu.

8.4 Ošetření správného snímání hodnot napětí

V kapitole 7.3 je zmíněna špatná činnost převodníku AD7890-10, kdy na vstupech převodníku je změřeno napětí i v případě, že je tento vstup plovoucí a současně změřené napětí je velice proměnlivé. Po podrobném prostudování manuálu k tomu to převodníku byl vznesen dotaz přímo na jeho výrobce. Technická podpora firmy Analog Devices se k problému vyjádřila s tím, že se jedná o digitální šum. E-mail s dotazem a s odpovědí technické podpory lze nalézt v příloze 10.4.

8.5 Rekapitulace zadání a dosažených výsledků

V tab. 14 jsou uvedeny požadavky na uzel CAN tak, jak byly specifikovány v zadání diplomové práce a stručně rekapitulovány výsledky, kterých bylo dosaženo. Znaménko „+“ resp. „-“ u každého z výsledků určuje jeho pozitivní resp. negativní význam.

Požadavek	Výsledek
snímání hodnot AD převodníkem	- nepřesné měření hodnot napětí vlivem digitálního šumu + možnost zobrazování snímaných hodnot na LCD displeji + možnost přepínání zobrazování snímaných hodnot na LCD displeji pomocí klávesnice
cyklické vysílání získaných hodnot na sběrnici CAN	+ cyklické vysílání rozšířeno o možnost odvysílání pouze na požadavek u daných analogových vstupů
zobrazování přijatých zpráv na LCD displej event. LED diody	+ možnost zobrazování přijatých zpráv na LCD displej i LED diody současně + možnost přepínání zobrazování příchozích zpráv pomocí klávesnice
nastavování vlastností uzlu prostřednictvím sériové linky RS232	+ nastavování a kontrola nastavených parametrů + řízení činnosti uzlu z PC

Tab. 14 Rekapitulace dosažených výsledků

8.6 Možná zlepšení návrhu, další vývoj

Odstranění negativních vlivů digitálního šumu je podle techniků firmy Analog Devices možné připojením externího Anti-aliasing filtru mezi vývody převodníku MUX OUT a SHA IN. Mělo by se v podstatě jednat o pasivní dolnofrekvenční propust, která by odstranila digitální šum. Dále výrobce doporučuje použití kvalitnějšího zdroje hodinového signálu, nežli je použitý generátor mikroprocesoru AT89C52. Toto vše by mohlo být v dalších návrzích zahrnuto.

Podstatnou nevýhodou uzlu je, že po jeho spuštění je použita napevno přednastavená konfigurace se kterou byl uzel testován a případnou změnu této konfigurace lze dodatečně zajistit nastavením prostřednictvím sériové linky. Aby bylo dosaženo vyšší univerzálnosti, bylo by zapotřebí, aby uzel zaznamenal naposledy použitou konfiguraci a tu při dalším spuštění automaticky použil. V hardwarovém návrhu uzlu je s touto alternativou počítáno, do obvodového řešení je zahrnuta energeticky nezávislá sériová paměť typu EEPROM AT24C04, do které by bylo možno tuto konfiguraci ukládat.

Použití sériové linky RS232 pouze k nastavování vlastností uzlu by mohlo být rozšířeno o předávání přijatých zpráv ze sběrnice CAN prostřednictvím této linky osobnímu počítači a jejich následnou vizualizaci. Takovéto vylepšení by však muselo řešit problém s maximální rychlostí sběrnice CAN, což je 1 MBaud, a maximální rychlostí sériové linky u standardního PC, která je 115,2 kBaud.

Možným uspořením vývodů mikroprocesoru AT89C52 uvedeným v kapitole 8.2 by mohlo dojít k vybavení uzlu interfacem k dalším rozhraním, např. RS485. V takovémto případě by uzel CAN mohl sloužit jako jakási křižovatka mezi těmito rozhraními, přičemž by opět musel být řešen problém s rozdílnými přenosovými rychlostmi. Lze však použít ještě mnoho dalších moderních technologií, např. pro dálkové řízení připojit na sériovou linku GSM modul, pomocí něhož by mohla být data a následně i technologické procesy zpracovávány dálkově prostřednictvím mobilní telefonní sítě GSM, což je v současné době jeden z nejmodernějších nasazovaných způsobů řízení.

9. Literatura

[1] SHARP Microelectronics of the Americas: Dot-Matrix LCD Units (with built-in controllers), SHARP Microelectronics of the Americas 1999

[2] Intel Corporation: 82527 Serial Communications Controller Architectural Overview, Intel Corporation 1996

[3] Fuksa, M.: Alespoň něco o 8051…, http://8051.zde.cz

[4] ELBAS, s.r.o.: Sběrnice CAN ve vozidlech, http://www.elbas.cz

[5] Analog Devices, Inc.: LC^{2}MOS 8-Channel, 12-bit Seriál, Data Acquisition Systém AD7890, Analog Devices, Inc. 2001

[6] KEIL ELEKTRONIK GmbH: C-Compiler-51 User's Guide 9.88, KEIL ELEKTRONIK GmbH 1988

[7] Philips Electronics N.V.: PCA82C250 - CAN controller interface - Data Sheet, Philips Electronics N.V. 2000

[8] Philips Electronics N.V.: PCA82C250 / 251 CAN Transceiver - Application Note, Philips Electronics N.V. 1996

[9] Intel Corporation: Interfacing an MCS-51 Microcontroller to an 82527 CAN Controller - Application Note, Intel Corporation 1995

[10] Intel Corporation: 82527 Serial Communications Controller - Controller Area Network Protocol, Intel Corporation 1995

[11] Hitex (UK) Ltd.: An Introduction To C On The C166 Family, Hitex (UK) Ltd. 1997

[12] Bartůněk, I.: CAN – Controller-Area-Network, Automatizace 4/1996

[13] Bartůněk, I.: Distribuované moduly CAN pro průmyslové aplikace, Automatizace 9/1996

[14] Atmel Corporation: 8-bit Microcontroller with 8K Bytes Flash AT89C52, Atmel Corporation 1999

[15] Hlava, J.: Prostředky automatického řízení, nepublikované přednášky

[16] Grosman, J.: Řídící počítačové systémy, nepublikované přednášky

[17] Semenec, P., Novák, L.: Realizace jednoduchého uzlu s protokolem CAN, závěrečná zpráva ročníkového projektu, TUL FM KSI 2002

[18] Intel Corporation: 82527 Serial Communications Controller Architectural Overview, Intel Corporation 1995

[19] Hitex (UK) Ltd.: Evaluation And Training Board 167IO User Guide, Hitex (UK) Ltd. 1997

10.Přílohy

10.1 Záznam komunikace mezi PC a uzlem CAN

```
Pole hodnot: ´ˆ˜¯˘·¸°   ••••••••••
!"#$%&'()0123456789@ABCDEFGHIPQRSTUVWXY`abcdefghipqrstuvwxy• ,•„…†‡•‰ ‘’BBB

Pole platností: ••••••••••••

PC: •´ˆ˜¯˘·¸°•
CAN: •´ˆ˜¯˘·¸°•
PC: U
CAN: f
PC: • ••••••••
CAN: •      ••••••••
PC: U
CAN: f
PC: •••• !"#$•
CAN: •••• !"#$•
PC: U
CAN: f
PC: •%&'()012•
CAN: •%&'()012•
PC: U
CAN: f
PC: •3456789@•
CAN: •3456789@•
PC: U
CAN: f
PC: •ABCDEFGH•
CAN: •ABCDEFGH•
PC: U
CAN: f
PC: •IPQRSTUV•
CAN: •IPQRSTUV•
PC: U
CAN: f
PC: •WXY`abcd•
CAN: •WXY`abcd•
PC: U
CAN: f
PC: •efghipqr•
CAN: •efghipqr•
PC: U
CAN: f
PC: •stuvwxy••
CAN: •stuvwxy••
PC: U
CAN: f
PC: • ,•„…†‡••
CAN: • ,•„…†‡••
PC: U
CAN: f
PC: •‰ ‘’BBBB•
CAN: •‰ ‘’BBBB•
PC: U
CAN: f
```

10.2 Mapa adres řadiče CAN Intel AN82527

00H	Control Register
01H	Status Register
02H	CPU Interface Reg.
03H	Reserved
04-05H	High Speed Read
06-07H	Global Mask - Standard
08-0BH	Global Mask –Extended
0C-0FH	Message 15 Mask
10-1EH	**Message 1**
1FH	CLKOUT Register
20-2EH	**Message 2**
2FH	Bus Config. Register
30-3EH	**Message 3**
3FH	Bit Timing Reg. 0
40-4EH	**Message 4**
4FH	Bit Timing Reg. 1
50-5EH	**Message 5**
5FH	Interrupt Register
60-6EH	**Message 6**
6FH	Reserved
70-7EH	**Message 7**
7FH	Reserved
80-8EH	**Message 8**
8FH	Reserved
90-9EH	**Message 9**
9FH	P1CONF
A0-AEH	**Message 10**
AFH	P2CONF
B0-BEH	**Message 11**
BFH	P1IN
C0-CEH	**Message 12**
CFH	P2IN
D0-DEH	**Message 13**
DFH	P1OUT
E0-EEH	Message 14
EFH	P2OUT
F0-FEH	**Message 15**
FFH	Serial Reset Address

10.3 Význam jednotlivých editačních polí v aplikaci CAN Setup v. 1.2

10.3.1 CPU Interface Register (02H)

7	6	5	4	3	2	1	0
RstST	DSC	DMC	PwD	Sleep	MUX	0	CEn
r	rw	rw	rw	rw	rw	rw	rw

RstST Hardware Reset Status

log. 1 Hardwarový reset řadiče je aktivní (RESET # v log. 0). Dokud je reset aktivní, není možný přístup k 82527.

log. 0 Normální režim. CPU se musí ujistit, že je tento bit nulový před prvním přístupem k 82527 po resetu.

Tento bit je zapisován obvodem 82527.

DSC Divide System Clock (SCLK). SCLK nemůže přesáhnout hodnotu 10 Mhz.

log. 1 Hodinový signál systému, SCLK, odpovídá hodnotě XTAL/2.

log. 0 Hodinový signál systému odpovídá hodnotě XTAL.

Tento bit je zapisován CPU.

DMC Divide Memory Clock (MCLK). Hodinový signál paměti nemůže přesáhnout hodnotu 8 Mhz.

log. 1 Hodinový signál paměti odpovídá hodnotě SCLK/2.

log. 0 Hodinový signál odpovídá hodnotě SCLK.

Tento bit je zapisován CPU.

PwD Power Down Mode enable

Sleep Sleep Mode enable

PwD	Sleep	
0	0	Obojí, tj. Power Down a Sleep Mode je neaktivní
1	0	Power Down mód je aktivní
0	1	Sleep mód je aktivní

MUX Multiplex pro ISO nízkorychlostní fyzickou vrstvu

Pokud je $V_{CC}/2$ užit k implementaci základní CAN fyzické vrstvy, pin 24 přivádí napěťový výstup $V_{CC}/2$ a pin 11 je výstup přerušení přiváděného do CPU. Jinak je signál přerušení dostupný pouze na pinu 24. $V_{CC}/2$ je dostupné pouze během normálních operací a při Sleep módu, není dostupné během módu Power Down.

Poznámka:

DcR1 bit (adresa 2FH) musí být nastaven pro povolení $V_{CC}/2$ na pinu č. 24.

log. 1 ISO nízkorychlostní fyzická vrstva aktivní: pin 24 = $V_{CC}/2$, pin 11 = INT#.

log. 0 Normální operace: pin 24 = INT#, pin 11 = P2.6

Tento bit je zapisován CPU.

Reserved Bit 1

log. 1 Tato hodnota nemusí být programována uživatelem.

log. 0 Log. 0 musí být vždy zapsána na tento bit

CEn Clockout enable

log. 1 Hodinový signál je zapnut (defaultní nastavení po resetu).

log. 0 Hodinový signál je vypnut.

10.3.2 Global Mask – Standard Register (06 – 07H)

06H

7	6	5	4	3	2	1	0
ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
rw	rw	rw	rw	rw	rw	rw	rw

07H

7	6	5	4	3	2	1	0
ID20	ID19	ID18	rezervováno				
rw	rw	rw					

Rezervované bity jsou čteny jako log. 1.

Nastavená hodnota standardní globální masky po hardwarovém resetu je nezměněna.

Standardní globální maska je aplikována pouze na zprávy používající standardní identifikátor CAN nebo objekty zpráv s nastaveným XTD bitem na log. 0. Toto vyžaduje též volání

akceptace filtrování zpráv povolených uživatelem nastavením globální masky nebo „don't care" jakýchkoliv bitů identifikátoru příchozí zprávy. Tato maska je programovatelná pro povolení využívání specifické maskovací strategie definované uživatelem.

Hodnota log. 0 znamená „don't care" nebo akceptování log. 0 nebo log. 1 na dané pozici bitu. Hodnota log. 1 znamená, že příchozí bit musí souhlasit identicky s příslušným bitem identifikátoru zprávy.

10.3.3 Global Mask – Extended Register (08 – 0BH)

08H

7	6	5	4	3	2	1	0
ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
rw	rw	rw	rw	rw	rw	rw	rw

09H

7	6	5	4	3	2	1	0
ID20	ID19	ID18	ID17	ID16	ID15	ID14	ID13
rw	rw	rw	rw	rw	rw	rw	rw

0AH

7	6	5	4	3	2	1	0
ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5
rw	rw	rw	rw	rw	rw	rw	rw

0BH

7	6	5	4	3	2	1	0
ID4	ID3	ID2	ID1	ID0	rezervováno		
rw	rw	rw	rw	rw	rw		

Rezervované bity jsou čteny jako 000
Ostatní – viz. Global Mask Standard.

10.3.4 Message 15 Mask Register (0C – 0FH)

0CH

7	6	5	4	3	2	1	0
ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
rw	rw	rw	rw	rw	rw	rw	rw

0DH

7	6	5	4	3	2	1	0
ID20	ID19	ID18	ID17	ID16	ID15	ID14	ID13
rw	rw	rw	rw	rw	rw	rw	rw

0EH

7	6	5	4	3	2	1	0
ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5
rw	rw	rw	rw	rw	rw	rw	rw

0FH

7	6	5	4	3	2	1	0
ID4	ID3	ID2	ID1	ID0	rezervováno		
rw	rw	rw	rw	r			

Význam viz. Global Mask Extended.

10.3.5 Bus Configuration Register (2FH)

2FH

7	6	5	4	3	2	1	0
0	CoBy	Pol	0	DcT1	0	DcR1	DcR0
rw	rw	rw	rw	rw	rw	rw	rw

Rezervované bity 7, 4, 2

log. 1 Tato hodnota nemusí být programována uživatelem.

log. 0 Log. 0 musí být vždy zapsána na tento bit.

CoBy Comparator Bypass

log. 1 Vstupní komparátor je odstaven a vstup RX0 je nastaven jako platný vstup sběrnice (DcR0 musí být nastaven na log. 0)

log. 0 Normální operace: RX0 a RX1 jsou vstupy do vstupního komparátoru (defaultní po hardwarovém resetu).

Pol Polarity

log. 1 Pokud je vstupní komparátor odstaven, potom je log. 1 interpretována jako hodnota dominant a log. 0 je hodnota recessive na vstupu RX0.

log. 0 Jestliže je vstupní komparátor odstaven, log. 1 je interpretována jako hodnota recessive a log. 0 je dominant bit na vstupu RX0 (defaultní po hardwarovém resetu).

DcT1 Disconnect TX1 output

log. 1 Zakáže ovladač výstupu TX1. Tento mód je pro použití jednoduchého rozhraní sběrnice nebo v případě diferenciální sběrnice, když vodiče sběrnice jsou zkratovány.

log. 0 Povoluje ovladač výstupu TX1 (defaultní po hardwarovém resetu).

DcR1 Disconnect RX1 input

log. 1 RX1 je zakázán a vstup RX1 je odpojen od invertujícího vstupu komparátoru a je nahrazen referenčním napětím $V_{CC}/2$.

log. 0 RX1 je povolen a vstup RX1 je připojen k invertujícímu vstupu komparátoru (defaultní po hardwarovém resetu).

DcR0 Disconnect RX0 input

log. 1 RX0 je zakázán a vstup RX0 je odpojen od neinvertujícího vstupu komparátoru a nahrazen referenčním napětím $V_{CC}/2$. Bit MUX v CPU Interface Register (02H) musí být nastaven na log. 1 pro aktivování referenčního napětí $V_{CC}/2$.

log. 0 RX0 je povolen a vstup RX0 je připojen k neinvertujícímu vstupu vstupního komparátoru (defaultní po hardwarovém resetu).

Hodnota registru po hardwarovém resetu je 00H.

10.3.6 Bit Timing Registers (3FH, 4FH)

Registry časování bitu jsou použity k definování frekvence sběrnice CAN, vzorkovacího bodu v čase jednoho bitu a režimu synchronizace.

Bit Timing Register 0 (3FH)

3FH

7	6	5	4	3	2	1	0
SJW		BRP					
rw		rw					

SJW (Re)Synchronization Jump Width

Platné programované hodnoty jsou 0 – 3. SJW definuje maximální počet časových kvant o které může být zkrácen nebo prodloužen čas bitu během jedné synchronizace. Aktuální interpretace této hodnoty hardwarem je použití jiné než naprogramované hodnoty.

BRP Baud Rate Prescaler

Platné programované hodnoty jsou 0 – 63. BRP programuje délku jednoho časového kvanta: $tq = t_{SCLK} \times (BRP + 1)$, kde t_{SCLK} je perioda hodinového signálu systému. Defaultní hodnota registru časování bitu 0 po hardwarovém resetu je nezměněná.

Bit Timing Register 1 (4FH)

4FH

7	6	5	4	3	2	1	0
Spl	TSEG2			TSEG1			
rw	rw			rw			

Spl Sampling Mode

Sampling Mode = log. 0 může rozlišit v rychlejších přenosových rozsazích, pokud sampling mode = log. 1 je mnohem imunnější k šumovým špičkám na sběrnici CAN.

log. 1 Tři vzorky jsou použity k určení platné hodnoty bitu s využitím majoritní logiky. Sběrnice CAN je vzorkována třikrát během doby jednoho bitu.

log. 0 Jeden vzorek je použit k určení platné hodnoty bitu. Sběrnice CAN je vzorkována jednou během doby jednoho bitu.

TSEG1 Time Segment 1

Platná programované hodnota jsou 2 – 15. TSEG1 je časový segment před vzorkovacím bodem. Aktuální interpretace této hodnoty hardwarem je jiná než hodnota programovaná uživatelem.

TSEG2 Time Segment 2

Platná programované hodnota jsou 1 – 7. TSEG2 je časový segment po vzorkovacím bodu. Aktuální interpretace této hodnoty hardwarem je jiná než hodnota programovaná uživatelem.

10.3.7 Port 1,2 Configuration Register (9FH,AFH)

9FH

7	6	5	4	3	2	1	0
P1CONF 0-7							
rw							

P1CONF 0 – 7 (9FH)

Port 1 vstup/výstup konfigurační bity

log. 1 Pin portu je definován jako push-pull výstup

log. 0 Pin portu je definován jako vstup s vysokou impedancí.

Defaultní hodnota po hardwarovém resetu je 00H.

P2CONF (AFH)

AFH

7	6	5	4	3	2	1	0
P2CONF 0-7							
rw							

P2CONF 0 – 7

Port 2 vstup/výstup konfigurační bity

log. 1 Pin portu je definován jako push-pull výstup

log. 0 Pin portu je definován jako vstup s vysokou impedancí.

Defaultní hodnota po hardwarovém resetu je 00H.

10.3.8 CLKOUT (Clockout) Register (1FH)

1FH

7	6	5	4	3	2	1	0
0	0	SL1	SL0	CD_V			
r	r	rw	rw	rw			

Registr CLKOUT řídí kmitočet signálu CLKOUT stejně jako předvolený rozsah. Defaultní frekvence signálu CLKOUT závisí na módu propojení s CPU. Pro módy 0, 1 a sériový mód je typická frekvence XTAL. Pro módy 2 a 3 je typický kmitočet XTAL/2.

CD_V	CLKOUT
0	XTAL
1	XTAL/2
10	XTAL/3
11	XTAL/4
100	XTAL/5
101	XTAL/6
110	XTAL/7
111	XTAL/8
1000	XTAL/9
1001	XTAL/10
1010	XTAL/11
1011	XTAL/12
1100	XTAL/13
1101	XTAL/14
1110	XTAL/15
1111	rezervováno

SL1	SL0	CLKOUT rozsahy
0	0	CLKOUT > 24 MHz
0	1	16 MHz < CLKOUT ≤ 24 MHz
1	0	8 MHz < CLKOUT ≤ 16 MHz
1	1	8 MHz < CLKOUT

Defaultní hodnota po hardwarovém resetu je 00H pro módy 0, 1 a sériový a 01H pro módy 2,3.

10.3.9 Arbitration 0, 1, 2, 3 Registers

Bázovou adresu arbitračního registru náležící příslušnému objektu zpráv naleznete v mapě adres.

Arbitrace 0 (bázová adresa + 2)

7	6	5	4	3	2	1	0
ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
rw	rw	rw	rw	rw	rw	rw	rw

Arbitrace 1 (bázová adresa + 3)

7	6	5	4	3	2	1	0
ID20	ID19	ID18	ID17	ID16	ID15	ID14	ID13
rw	rw	rw	rw	rw	rw	rw	rw

Arbitrace 2 (bázová adresa + 4)

7	6	5	4	3	2	1	0
ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5
rw	rw	rw	rw	rw	rw	rw	rw

Arbitrace 0 (bázová adresa + 5)

7	6	5	4	3	2	1	0
ID4	ID3	ID2	ID1	ID0	rezervováno		
rw	rw	rw	rw	rw	r		

Rezervované bity jsou čteny jako 000.

ID0 – ID28 Identifikátor zprávy

ID0 – ID28 Identifikátor pro rozšířený formát rámce.

ID18 – ID28 Identifikátor pro standardní formát rámce.

10.3.10 Message Configuration Register (bázová adresa + 6)

MessConf (bázová adresa + 6)

7	6	5	4	3	2	1	0
DLC				Dir	Xtd	rezervováno	
rw				rw	rw	r	

DLC Data Length Code

Platné programované hodnoty jsou 0 – 8. Kód délky dat objektu zprávy je zapsán hodnotou odpovídající délce dat.

Dir Direction

log. 1 Směr = vysílání. Když je nastaven bit TXRqst, zpráva bude odvysílána.

log. 0 Směr = příjem. Když je nastaven bit TXRqst, rámec žádosti bude odvysílán. Když je zpráva přijata s platným identifikátorem, zpráva bude uložena v objektu zpráv.

Xtd Extended or standard identifier

log. 1 Tento objekt zpráv užívá 29 bitový identifikátor.

log. 0 Tento objekt zpráv užívá 11 bitový identifikátor.

10.4 E-mail od technické podpory firmy Analog Devices

From: lubomir.novak@centrum.cz [mailto:lubomir.novak@centrum.cz]

Sent: 09 April 2003 11:37

Posted To: Euro.linear backup

Conversation: Standard Linear Technical Support

Subject: Standard Linear Technical Support

problem:

Hi

I have a problem with this ADC. In my apllication is connected to I/O pins of 8-bit microprocessor AT89C52. When are the analog inputs floating, on this inputs are voltages approximately 1,4 V. When i connect the source of constant voltage, the measured value is jumping (circa 5 lower bits of digital value) .

Description of my circuitry:

* AGND(1,15),DGND(3): 0V[GND]
* SMODE(2): +5V (slave mode)
* Cext (4): through 120 pF capacitor connected to 0V
* CONVST (5): +5 V
* CLK IN (6): pin P1.0 of AT89C52 (Clock generator)
* TFS (8), RFS (9), SCLK (7) : I/O different pins of AT89C52
* DATA IN (11), DATA OUT (10): both connected to I/O pin of AT89C52
* Vdd (12): + 5 V
* MUX OUT (13) is connected to SHA IN (14)
* REF OUT/REF IN (24): decoupled to 0V with a 100 nF ceramic disc capacitor.

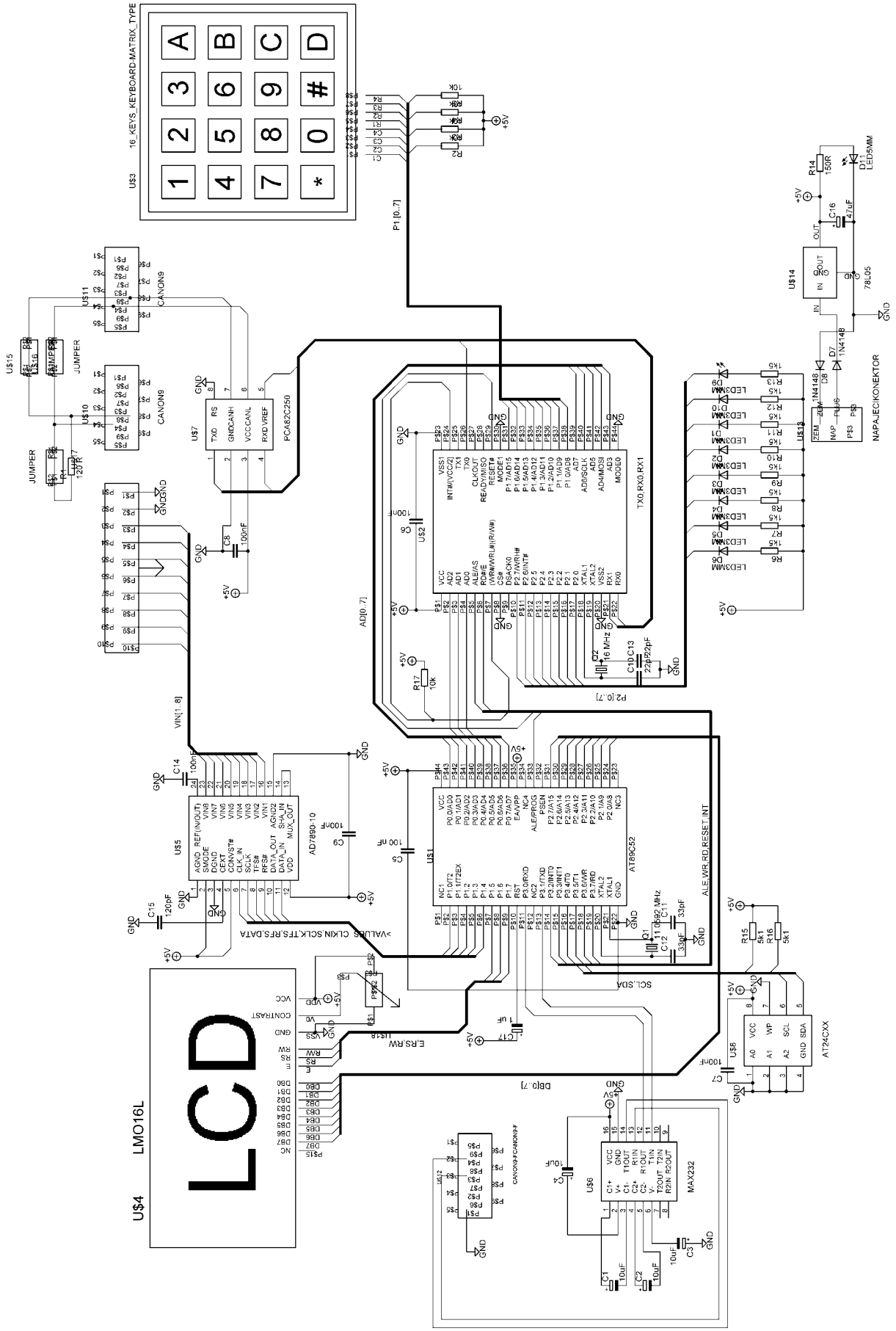
I hope, that You know the answer on my problem.
Goodbye.

email : lubomir.novak@centrum.cz
name : Lubomir Novak
company : Technical University of Liberec
phone : +42 048-5355 x360
fax :
support_location : CZECH REPUBLIC
number : AD7890-10

This sounds like digital noise being seen by the analogue inputs.Your basic setup looks ok to me except that you appear to have no anti-aliasing filter . Even a simple passive lowpass filter will help to remove any higher frequency digital noise that may be picked up along your analog input PCB tracks.

A common problem is that the logic outputs from a microcontroller can contain a large amount of jitter (random variation in the exact timing of the rising/falling edges of the clock signal). A poor quality sampling clock can result in increased noise in the ADC conversion result. You'll need to evaluate whether this is a problem in your system. You may have to test this by temporarily applying a clean clock from a known low jitter source. I am copying this email to the AD7890 factory applications engineer in case he has any further comments
Kind regards
Chris

10.5 Kompletní schéma CAN Node v. 2.0



10.6 Stručný návod k použití desky CAN Node v. 2.0

D0…D7 - signalizační LED diody

T – trimr regulující intenzitu jasu LCD displeje

VIN1…VIN8 - napěťové vstupy AD převodníku, lze připojit napětí ± 10 V proti svorce GND

GND – svorky pro připojení nulového napětí

J1 – zkratovací propojka propojující signál CAN_H na konektor CN2 (pin č. 8)

J2 – zkratovací propojka propojující signál CAN_L na konektor CN2 (pin č. 4)

J3 – zkratovací propojka připojující zakončovací odpor 120 Ω mezi CAN_H a CAN_L

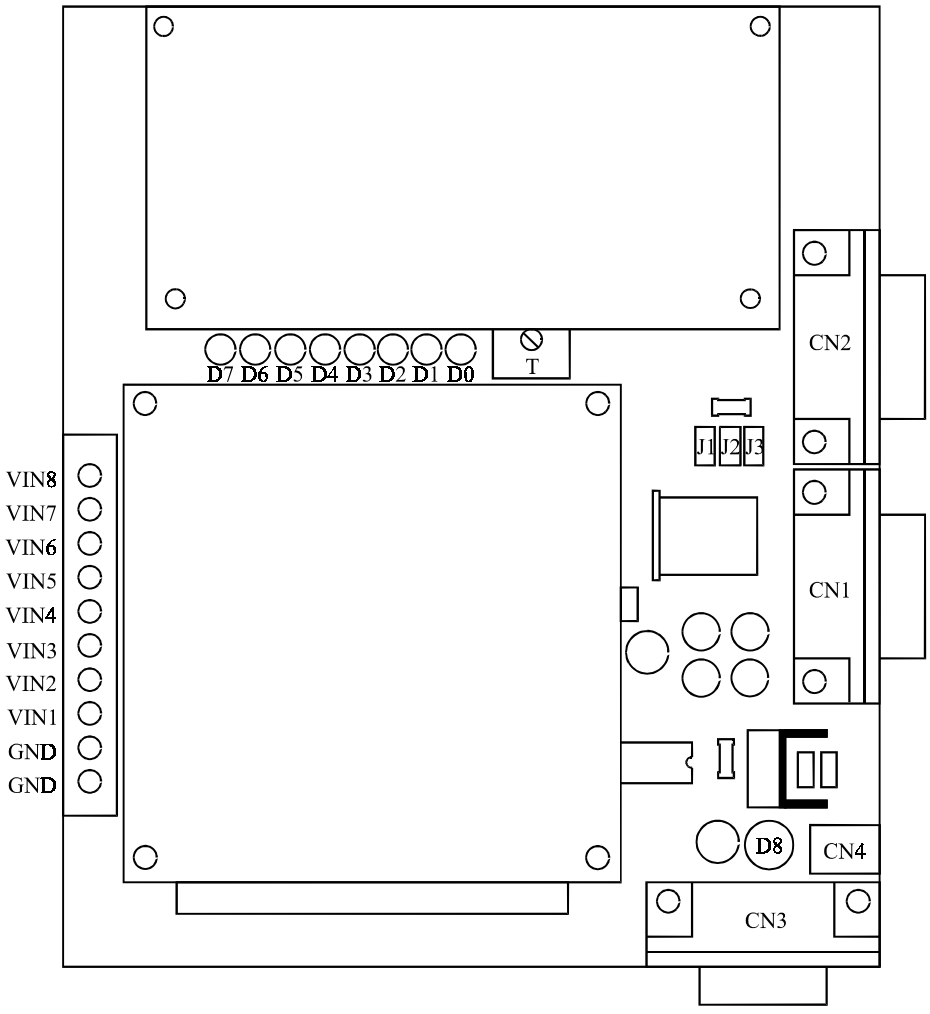
CN1 – primární konektor pro připojení na sběrnici CAN přiloženým kabelem

CN2 – sekundární konektor pro připojení na sběrnici CAN přiloženým kabelem

CN3 – konektor pro propojení s PC pomocí sériového rozhraní RS232 přiloženým kabelem

CN4 – konektor pro napájení přiloženým napájecím zdrojem

D8 – signalizace zapnutí



10.7 Software na CD-ROM