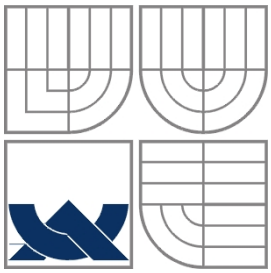
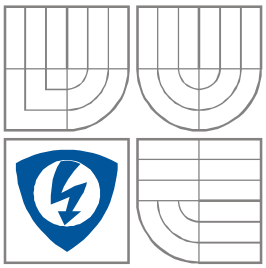


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

LABORATORNÍ PŘÍPRAVEK DEMONSTRUJÍCÍ SBĚRNICI CAN
LABORATORY BOARD DEMONSTRATES CAN BUS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S PROJECT

AUTOR PRÁCE — Vladimír Levek
AUTHOR

VEDOUCÍ PRÁCE — Ing. Tomáš Frýza, Ph.D.
SUPERVISOR

BRNO, 2009

VYSOKÉ UČENÍ
TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

Ústav radioelektroniky

Bakalářská práce

bakalářský studijní obor
Elektronika a sdělovací technika

Student:	Vladimír Levek	***ID:***	73104
Ročník:	3	***Akademický rok:***	2008/2009

NÁZEV TÉMATU:

Laboratorní přípravek demonstrující sběrnici CAN

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte a popište komunikační protokol sběrnice CAN. Sestavte stručný přehled dostupných zařízení a aplikací komunikující pomocí této sběrnice. Navrhněte obvodové zapojení a desku plošného spoje laboratorního přípravku demonstrující sběrnici CAN.

Realizujte a oživte laboratorní přípravek. V jazyce C naprogramujte ukázkovou aplikaci. Navrhněte zadání laboratorní úlohy.

DOPORUČENÁ LITERATURA:

[1] International Organization for Standardization. ISO 11898:2003, Road vehicles - Controller area network (CAN) for high-speed communication. [online]. 1993 – [cit. 7. května 2008]. Dostupné na WWW: http://www.iso.org/iso/iso_catalogue/catalogue_ics/catalogue_detail_ics.htm?csnumber=20380

[2] Freescale Semiconductor. Controller Area Network. [online]. 2008 – [cit. 7. května 2008]. Dostupné na WWW: http://www.freescale.com/webapp/sps/site/homepage.jsp?nodeId=02WcbfNZnL25B4

[3] Atmel Corporation. CAN AVR. [online]. 2008 – [cit. 7. května 2008]. Dostupné na WWW: http://www.atmel.com/dyn/products/devices.asp?family_id=607#1609

Termín zadání:	9.2.2009	***Termín odevzdání:***	5.6.2009

Vedoucí práce: Ing. Tomáš Frýza, Ph.D.

prof. Dr. Ing. Zbyněk Raida
Předseda oborové rady

Abstrakt

Cílem této semestrální práce je navrhnout laboratorní přípravek demonstrující komunikaci pomocí sběrnice CAN. Laboratorní přípravky, řízené mikroprocesorem, budou posílat stavy vstupů a výstupů mezi sebou. Rovněž bude umožněno vysílat a přijímat zprávy pomocí sériového portu RS-232. Protokolem CAN budou posílány tyto informace: teplota, osvětlení, analogový vstup 0 - 5 V, analogový výstup 0 - 5 V, digitální vstupy úrovně TTL a digitální výstupy typu otevřený kolektor.

Abstract

The goal of this thesis, is proposal of the laboratory board demonstrates communication by the help of bus CAN. Laboratory boards, controlled with microprocessor, will be send statuses inputs and outputs with each other. Also it will can be transmit and receive messages by the help of the serial port RS-232. With protocol CAN will be send this information: Temperature, Lighting, Analog input 0 - 5 V, Analog output 0 - 5 V, Digital input level TTL and digital output type open collector.

Klíčová slova

Sběrnice, převodník, budič, řadič, protokol, UART, CAN, laboratorní úloha, procesor.

Keywords

Bus, Converter, Driver, Controller, Protocol, UART, CAN, Laboratory theme, Processor.

Bibliografická citace

LEVEK, V. *Laboratorní přípravek demonstrující sběrnici CAN.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 54 s. Vedoucí bakalářské práce Ing. Tomáš Frýza, Ph.D.

Prohlášení

Prohlašuji, že svou bakalářskou práci na téma Laboratorní přípravek demonstrující sběrnici CAN jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

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

V Brně dne 5. června 2009 .....
podpis autora

Poděkování

Děkuji vedoucímu bakalářské práce Ing. Tomáši Frýzovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce.

V Brně dne 5. června 2009 .....
podpis autora

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod - popis sběrnice CAN 2.0A

1.1 Historie a základní popis sběrnice CAN

V únoru roku 1986 byla představena na kongresu Společnosti automobilových inženýrů SAE (Society of Automotive Engineers) v Detroitu nová sběrnice CAN. Tuto komunikační síť vyvinula (původně jako modifikaci RS485) pro svou potřebu firma **Robert Bosch GmbH**. Sběrnice byla určena pro komunikaci mezi řídícími a výkonovými prvky v automobilech. Prvními uživateli byly automobily z továrny Mercedes-Benz, jejíž vývojoví pracovníci spolu s odborníky firmy Intel se spolupodíleli na jejím vývoji. Sběrnice CAN, která už má dnes mnohem širší uplatnění, je svými vlastnostmi předurčena pro využití nejen v dopravním průmyslu, ale i v dalších oblastech řídící techniky. Jejími hlavními přednostmi jsou:

- vysoká rychlost - až 1Mbit/s (tím je určena spíše pro komunikaci na krátké vzdálenosti)
- sběrnice pracuje bez nutnosti řídící jednotky - všechny uzly na sběrnici jsou rovnocenné
- vysoká spolehlivost a odolnost proti rušení
- propracovaný systém řízení, přístupu ke sběrnici, chybového hlášení, diagnostiky
- prioritní přístup vysílání - důležitá data jsou vysílána přednostně
- spojení pouze dvouvodičovým vedením

V současné době je sběrnice CAN definována normou ISO 11898, která specifikuje **fyzickou vrstvu** (elektrické rozhraní) a **linkovou vrstvu** (datový protokol) prostřednictvím verze CAN 2.0A. Vedle verze CAN 2.0A existuje ještě specifikace CAN 2.0B, která disponuje se standardním a rozšířeným formátem zprávy. Ta vznikla v důsledku rostoucí potřeby přenosu většího objemu dat [9].

Jak už bylo zmíněno, všechny uzly sběrnice CAN spolu komunikují po dvouvodičovém vedení bez nutnosti jakékoliv hlavní řídící jednotky - síť typu *multimaster*. Výhoda spočívá v tom, že pokud je nějaký uzel mimo provoz, sběrnice může pracovat dál. Pokud je sběrnice volná, může kterýkoliv uzel zahájit komunikaci. Vysílací uzel většinou neadresuje zprávu, tedy nevkládá žádnou informaci o cílovém uzlu, kterému je zpráva určena. Proto všechny uzly na sběrnici, pokud nejsou vyřazeny z komunikace, zprávu přijímají, a aktivně či pasivně se podílejí na přenosu všech dat. Vysílaná zpráva je uvozena identifikátorem, který udává význam přenášené zprávy a tím její prioritu, dále pak obsahuje identifikaci vysílače. Následují řídící informace a poté jsou již vysílána data. Ta jsou následována kontrolním rámcem, za kterým přijímače potvrdí (nebo vyvrátí) validitu přenesených dat a tím se komunikace ukončí. Po ukončení komunikace všechny uzly udržují určitou prodlevu na zpracování dat.

1.2 Fyzická vrstva sběrnice CAN

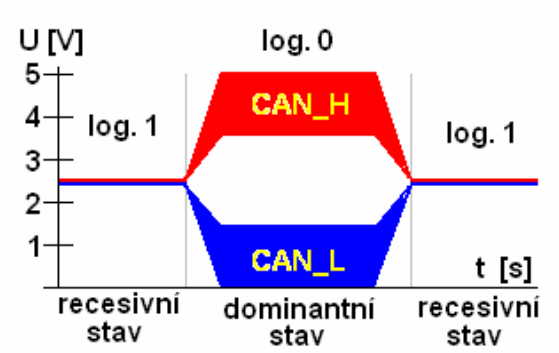
Sběrnice komunikuje po dvouvodičovém vedení. Vodiče jsou označeny **CAN_H** a **CAN_L**. Standard protokolu CAN definuje dva navzájem komplementární stavy bitů na sběrnici a jejich úrovně [1].

- dominantní stav (Dominant state) - min. rozdílové napětí: $V_{diff} = V_{CAN\_H\text{-}CAN\_L} = \mathbf{2\ V}$
- recesivní stav (Recessive state) - rozdílové napětí: $V_{diff} = V_{CAN\_H\text{-}CAN\_L} = \mathbf{0\ V}$

Viz Obr. 1.

Tyto stavy jsou obecné a záleží na konkrétní fyzické realizaci sběrnice. Vždy však musí splňovat podmínky:

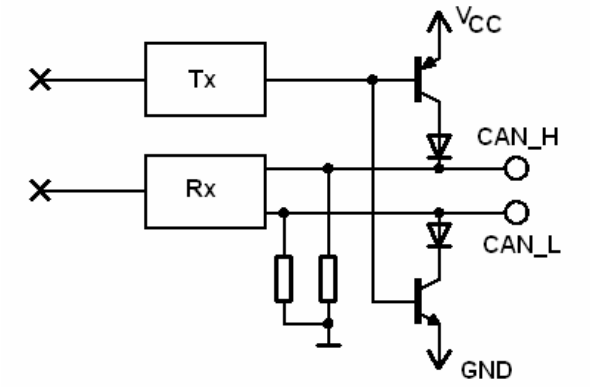
- vysílá-li alespoň jeden uzel dominantní bit, je sběrnice ve stavu dominantním
- vysílají-li všechny uzly recesivní bit, je sběrnice ve stavu recesivním



Obr. 1 *toleranční pásmo napěťových úrovní na sběrnici CAN*

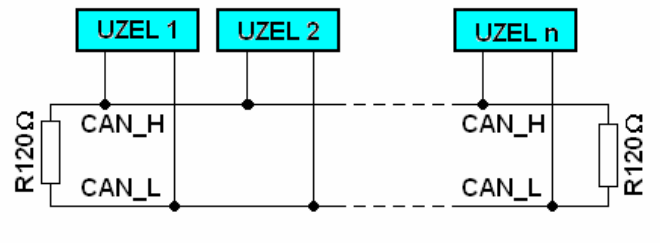
Dominantní stav je většinou realizován upnutím sběrnice dvěma komplementárními tranzistory, které vodič CAN_H připojí na napájecí napětí a vodič CAN_L připojí na nulový bod napájení uzlu. Formálně se pro tento stav všude používá označení: **log. 0**, neboli **stav L.** Viz Obr.2

Recesivní stav je realizován zavřenými tranzistory. Nulový potenciál sběrnice zajišťují shodně vyvážené rezistory, které zajistí, že napětí mezi oběma vodiči nepřesáhne dovolenou mez. Formálně se pro tento stav všude používá označení: **log. 1**, neboli **stav H.** Viz. Obr.2



Obr. 2 *principielní znázornění budiče sběrnice CAN*

Rozložení sběrnice je naznačeno na Obr. 3. Všechny uzly jsou připojeny na dvouvodičové vedení, které je zakončeno na obou stranách rezistory o hodnotě 120 Ω. Tyto eliminují odrazy vzniklé na koncích. Norma ISO 11898 určuje, že maximální počet uzlů na sběrnici může být 30. Je to doporučeno z důvodu zajištění průchodnosti sběrnice. Samozřejmě záleží na rychlosti, četnosti vysílání a objemu přenášených dat. Sběrnice při rychlosti 1 Mbit/s nemá mít délku větší než 40 m. Při rychlosti 50 kbit/s je doporučována maximální délka sběrnice 1 340 m.



Obr. 3 *typické propojení uzlů sběrnice CAN*

Rozložení sběrnice může být i jiné, například kruhové - přístup ke sběrnici je zajištěn z obou stran, to může být vhodné při aplikaci, kde hrozí rozpojení sběrnice. Další možností je hvězdicové řešení sběrnice. Tady hrozí u velmi vzdálených odboček uzlů nebezpečí, že mohou být data zpožděna a systém nemusí spolehlivě pracovat. Maximální vzdálenost srovnatelná s vlnovou délkou odbočky je dána vztahem:

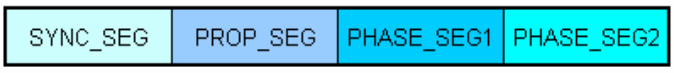
$$\lambda = \frac{c}{f} \quad . \tag{1}$$

Například pro přenos o rychlosti 1 Mbit/s (frekvence přenosu vysílání dat je f = 500 kHz) platí:

$$\lambda = \frac{c}{f} = \frac{3.10^8}{5.10^5} = 0{,}6.10^3 = 600 \text{ m} \quad . \tag{2}$$

Z důvodu udržení časové synchronizace mezi uzly musí mít změny úrovně signálu určitá pravidla. Přenos bitu není kódován žádným kódem (NRZ), musí tedy být vneseny do přenosu bity, které z hlediska zprávy jsou redundantní, ale z hlediska synchronizace přenosu nezbytné.

Doba každého přeneseného bitu je rozdělena do čtyř časových segmentů - viz Obr 4. Během doby SYNC_SEG se očekává hrana signálu (pokud je). Segment PROP_SEG slouží pro kompenzaci doby šíření signálu, mezi segmenty PHASE_SEG1 a PHASE_SEG2 se očekává vzorkování sběrnice [1].



Obr. 4 *časové segmenty doby přenosu jednoho bitu*

1.3 Linková vrstva sběrnice CAN

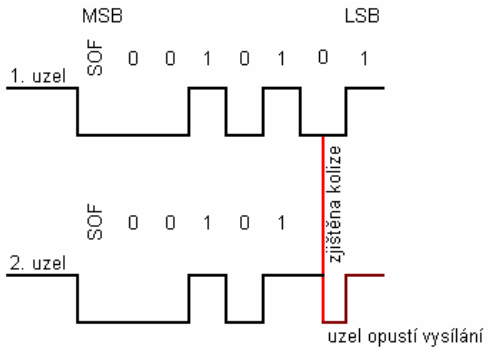
Linková vrstva protokolu CAN je podle modelu ISO rozdělena na dvě části

- podvrstva MAC *(Medium Acces Control)*
- podvrstva LCC *(Logical Link Control)*

Podvrstva MAC má za úkol řídit přístup ke sběrnici, zabezpečit přenos, detekovat chybové přenosy a signalizovat je. Podvrstva LCC má za úkol řídit přenos z hlediska organizace dat, filtrovat zprávy a provádět signalizaci přetížení uzlu.

1.3.1 Řízení přístupu ke sběrnici

Pokud má některý z uzlů připravenou zprávu pro odvysílání, kontroluje sběrnici. V případě, že je volná (Bus Free - recesivní stav - log.1), může zahájit vysílání. Vysílací uzel monitoruje sběrnici, zda je přítomna úroveň, kterou vysílá (to může pouze ve stavu recessive - log. 1). Pokud zjistí kolizi, tak se buď jedná o poruchu sběrnice nebo zahájilo více uzlů vysílání (s přesností jednoho taktu MCU) - Obr. 5. V tom případě uzel s vyšší prioritou (s nižším číslem identifikátoru) pokračuje ve vysílání a uzel s nižší prioritou (vyšším číslem identifikátoru) opustí vysílání a chová se jako přijímač, protože až dosud byla data obou vysílačů stejná. Po odvysílání zprávy se odstoupený uzel pokusí znovu získat přístup na sběrnici. Jelikož jsou data vysílána vždy ve směru MSB → LSB, musí být zajištěno, aby nejméně významná zpráva měla nejvyšší hodnotu identifikátoru a zpráva s nejvyšší prioritou měla nejnižší hodnotu identifikátoru. Pokud je v hodnotě identifikátoru obsažena adresa vysílače, měla by být na pozici LSB (pokud není priorita dána právě prioritou vysílacího uzlu - potom by měla být adresa na pozicích MSB).

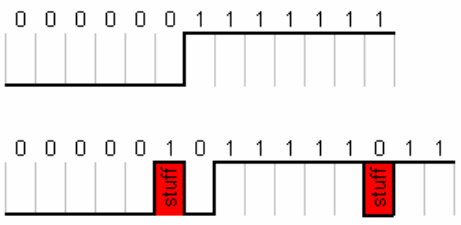


Obr. 5 *princip monitorování sběrnice*

1.3.2 Zabezpečení přenosu

Protokol CAN má několik způsobů zabezpečení přenosu. Všechny mají za úkol obstarat bezproblémový přenos, ale každý z jiného pohledu [1].

- Vkládání bitu *(Bit Stuffing)*. Jelikož nejsou na sběrnici přítomny taktovací signály, musí se přijímače synchronizovat pouze podle datového toku. Data jsou vysílána bez použití synchronizačního kódování - typ NRZ. Synchronizaci napomáhají právě bity stuffing. Ty jsou zobrazeny na obrázku 6. Vysílá-li se na sběrnici pět po sobě jdoucích bitů stejné úrovně, vloží se ihned za ně do zprávy bit opačné úrovně. V přijímači je tento bit detekován a vyřazen z toku dat. V opačném případě detekuje přijímač chybu bitu a signalizuje poruchu. Tyto, z hlediska dat redundantní bity, slouží k synchronizaci přenosu, ale také k průběžné kontrole přenosu.



Obr. 6 *princip vkládání bitu*

- Kód CRC kontrolního součtu *(Cyclic Redundancy Check)*. CRC kód je vložen na konec vysílaných zpráv a je tvořen šestnáctibitovým blokem. Ten je vypočten ze všech dosud odvysílaných bitů podle polynomu: $\mathbf{x^{15}+x^{14}+x^{10}+x^{8}+x^{7}+x^{4}+x^{3}+1}$. Nejvýznamější bit MSB výpočtu se nepřenáší a na pozici LSB je vložen bit ERC v recesivní úrovni (log. 1). Tato data jsou, jak z názvu vyplývá, redundantní a slouží pouze pro kontrolní účely. Pokud se přijímací uzel dopočítá stejného výsledku, pak je s vysokou pravděpodobností přenos v pořádku. Pokud ne, signalizuje přijímač na sběrnici chybu CRC.

- Monitorování sběrnice *(Monitoring)*. Jak již bylo dříve popsáno, vysílač kontroluje stav sběrnice. Pokud detekuje jinou úroveň, jedná se buď o chybu bitu, anebo se vysílače nacházejí na začátku přenosu v bloku řízení přístupu ke sběrnici. Potom ten vysílač, který má nižší prioritu (je to vždy ten, který zjistil kolizi) opustí vysílání, data přijímá, a po ukončení přenosu se znovu pokusí získat přístup ke sběrnici. Jedinou výjimkou, kde je dovolená jiná úroveň na sběrnici než na vysílači, je v bitu ACK (Acknowledge Bit).

- Potvrzení přijaté zprávy *(Acknowledge)*. Všechny přijímací uzly musí na konci datového rámce potvrdit, že zpráva je v pořádku přijatá a že všechny kontrolní a zabezpečovací mechanismy nedetekují chybu. Vysílač vyšle bit ACK v recesivním stavu a očekává od sběrnice stav dominantní (zde je právě rozdíl mezi vysílačem a sběrnicí). Pokud na sběrnici zůstane recesivní stav, pak je buď vysílač na sběrnici sám nebo přijímací uzly nejsou v pořádku.

1.4 Detekce a signalizace chyb sběrnice

Z důvodu zkvalitnění průchodnosti sběrnice je zavedena statistika chybovosti každého uzlu. Obecně platí, že uzel podle počtu chybových přenosů, ať už vysílaných nebo přijímaných, může být odpojen anebo omezen od přístupu ke sběrnici. To se řeší tak, že každý uzel má dvě počitadla chyb. Počitadlo REC (přijatá chybovost) a počitadlo TEC (vyslaná chybovost).V případě vadného přenosu se inkrementuje obsah počitadla, v případě dobrého přenosu se dekrementuje obsah konkrétního počitadla. Podle obsahu počitadel se může uzel nacházet v těchto stavech.

- Aktivní stav (Error Active). Uzel se aktivně podílí na komunikaci a v případě detekce chyby vysílá **aktivní** chybový rámec. Tím poškodí (a ukončí) probíhající přenos.
- Pasivní stav (Error Passive). Uzel má za sebou o 127 chybových přenosů. Normálně se podílí na komunikaci, ale v případě detekce chyby vysílá **pasivní** chybový rámec, čímž neublíží přenášeným datům, ale sám ztratí synchronizaci s probíhajícím přenosem.
- Stav odpojení od sběrnice (Bus off). Uzel má o 255 špatných přenosů více, zřejmě tedy dost "kazil" přenosy a je zcela odstaven od sběrnice.

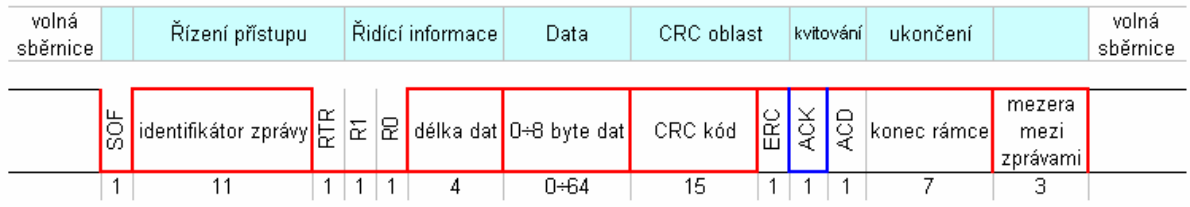
1.4.1 Typy komunikace

Protokol sběrnice CAN rozlišuje čtyři typy komunikace:

- Datový rámec *(Data Frame)*
- Žádost o rámec *(Remote Frame)*
- Chybový rámec *(Error Frame)*
- Rámec přetížení *(Overload Frame)*

1.4.2 Datový rámec

Tento rámec má za úkol standardní vyslání dat na sběrnici. Pokud má uzel připravenu datovou zprávu, detekuje sběrnici zda je volná, pokud ano, může zahájit vysílání dat - viz Obr. 7.

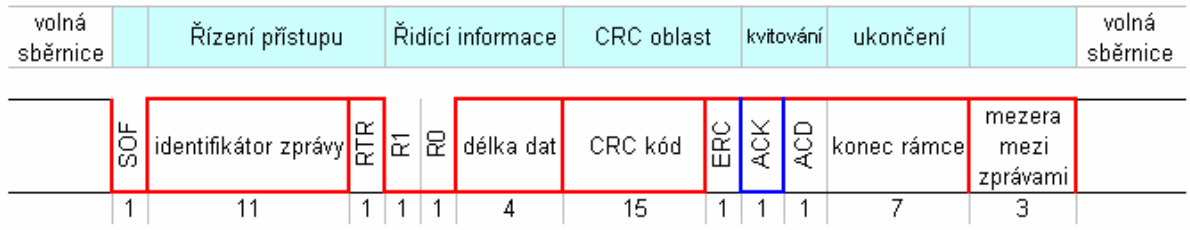


Obr. 7 *schéma datové zprávy sběrnice CAN 2.0A*

Datová zpráva je zahájena Start bitem (SOF), kterým přejde sběrnice z recesivního stavu do dominantního stavu. Následuje blok 11-ti bitů řízení přístupu na sběrnici (Arbitration Field). Tyto bity přinášejí informaci o významu zprávy, prioritu zprávy a adresu vysílače - identifikátor zprávy. Blok řízení přístupu je zakončen bitem RTR (Remote Request). Ten udává, že se jedná o vysílání dat (dominantní stav). Bezprostředně po bloku řízení přístupu následuje pole řídících informací. To je uvozeno dvojicí bitů R_1 a R_0 - ty jsou rezervovány pro další využití - nastavují se do dominantního stavu. Po nich následuje blok informující o počtu přenesených datových bytů. Maximální počet přenesených bytů je osm. Pokud se přenáší pouze jedna binární informace, může být počet datových bytů nulový a tato informace se může vložit do identifikátoru zprávy. Tím se uspoří časový prostor pro ostatní uzly. Po přenesení všech datových bytů je přeneseno 15 bitů CRC kódu. Po tomto bloku je přenesen jeden bit ERC - slouží pouze pro oddělení a vytvoření prodlevy pro zpracování CRC kódu. Bit ERC je v recesivním stavu. Po uplynutí předchozího bitu je vysílač v recesivním stavu a očekává od ostatních uzlů potvrzení správnosti přenosu (Acknowledge) - bit ACK. Ostatní uzly, pokud detekují správný příjem dat, nastaví sběrnici do dominantního stavu. Následuje bit ACD - oddělovač potvrzení, který je opět v recesivním stavu. Celý rámec je zakončen vysláním sedmi bitů v recesivním stavu. Tak je ukončena komunikace a novou komunikaci může zahájit kterýkoliv z uzlů až po uplynutí třech bitů. Tím se získá čas na zpracování získaných dat [1].

1.4.3 Rámec žádosti o data

Vysílání žádosti o rámec je vcelku podobné předcházející, tedy vysílat začíná uzel, který detekuje volnou sběrnici a nachystanou žádost o zaslání dat - viz Obr. 8.



Obr. 8 *schéma zprávy žádosti o data sběrnice CAN 2.0A*

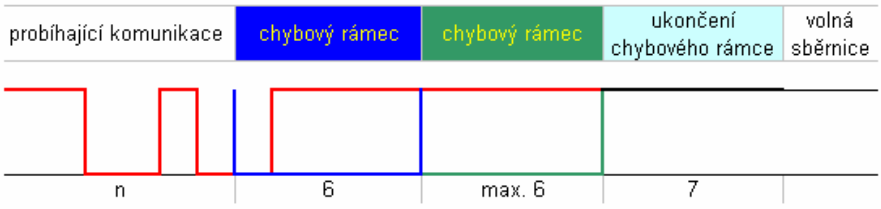
Start je opět zahájen bitem SOF, který je v dominantní úrovni. Následují identifikace zprávy a na konci je bit RTR nastaven do recesivní úrovně. Pokud nastane situace, že dva uzly vysílají se stejnou synchronizací a jeden z nich data chce vysílat a druhý chce o tato data žádat, tak uzel, který o data žádá, bude ve svém recesivním stavu detekovat dominantní stav na sběrnici, opustí vysílání a začne data přijímat. Zbytek rámce je už stejný. Jelikož pole délky dat je nulové, chybí tedy pole vysílaných dat. Rámec se opět standardně ukončí [1].

1.4.4 Rámec chybové zprávy

Chybová zpráva slouží k signalizaci chybného přenosu sběrnice CAN. Pokud některý z přijímacích uzlů detekuje v přenosu chybu, zahájí vysílání chybového rámce. Chyby mohou být zejména tyto:

- chyba bitu
- chyba vkládání bitu (Stuff bit)
- chyba CRC
- chyba rámce

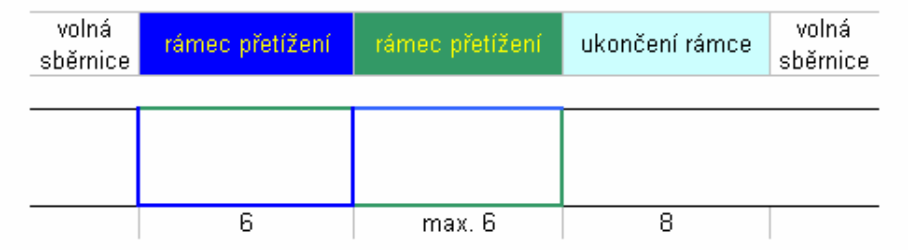
Pokud je uzel detekující chybu v aktivním stavu, zahájí vysílání chybového rámce. Ten spočívá ve vyslání šesti bitů v dominantním stavu - viz Obr.9. Tímto se přenášená zpráva znehodnotí a ostatní aktivní uzly to detekují a rovněž zahájí vysílání chybového rámce. Délka takto zřetězené chybové zprávy může být šest až dvanáct bitů. Pokud je uzel v pasivním stavu, vysílá šest bitů v recesivním stavu. Tím zprávu neznehodnotí. Po odvysílání chybových rámců uzly přejdou do recesivního stavu a detekují sběrnici (i původní vysílač - ten byl jako vysílač "odstaven" a podílí se na sběrnici jako přijímač). Poslední uzel, který ukončí chybový rámec, uvede sběrnici do recesivního stavu. Tak se uzly zasynchronizují, odvysílají sedm bitů recesivního stavu, tím se ukončí chybový rámec a sběrnice je ve stavu "Bus-free" [1].



Obr. 9 *schéma chybového rámce sběrnice CAN 2.0A*

1.4.5 Rámec přetížení

Má velmi podobnou strukturu jako chybový rámec. Je zobrazen na obrázku 10. Nesmí být však umístěn doprostřed probíhající komunikace. Rámcem přetížení signalizuje uzel, že nemůže z hlediska zaneprázdnění jinými instrukcemi data přijímat. Po ukončení standardního rámce může uzel vyslat sběrnici šest bitů v dominantní úrovni. Pokud jsou i jiné uzly zaneprázdněny, mohou během dalších šesti bitů rovněž vysílat příznak přetížení. Maximální počet bitů v poli přetížení může být 12. Po nich následuje osm bitů recesivní úrovně, které zakončí rámec přetížení. Tím se získá relativně dostatek času pro vyřešení interních záležitostí uvnitř uzlu [1].



Obr. 10 *schéma rámce přetížení sběrnice CAN 2.0A*

1.5 Aplikace sběrnice CAN

Komunikační obvody protokolu CAN se obvykle realizují pomocí komerčně dostupných obvodů. Přístupy ke sběrnici se dají realizovat různými způsoby, z nichž každý má své opodstatnění. Kritérium pro volbu použitých obvodů je především rychlost komunikace a množství zpracovávaných dat. Komponenty zajišťující provoz sběrnice můžeme dělit na: budiče, řadiče a kontroléry.

1.5.1 Budiče

Součástky, které fyzicky zajišťují komunikaci po sběrnici, se nazývají budiče. Nemají žádnou inteligenci, pouze vnitřní obvody zajišťující čtení a zápis dat a tím převod úrovně CAN na úroveň TTL. Mohou dále obsahovat určité ochrany proti přetížení, nevhodné manipulaci se sběrnicí, atd. Tyto komponenty většinou mají piny realizující styk se sběrnicí (CAN_H a CAN_L), piny realizující styk s řadičem sběrnice (Tx a Rx) a napájecí piny. Nejpoužívanějšími obvody jsou: Philips PCA82C250 [12], Philips PCA80C251, Temic B10011S, Unitrode UC 5350…..[1].

1.5.2 Řadiče

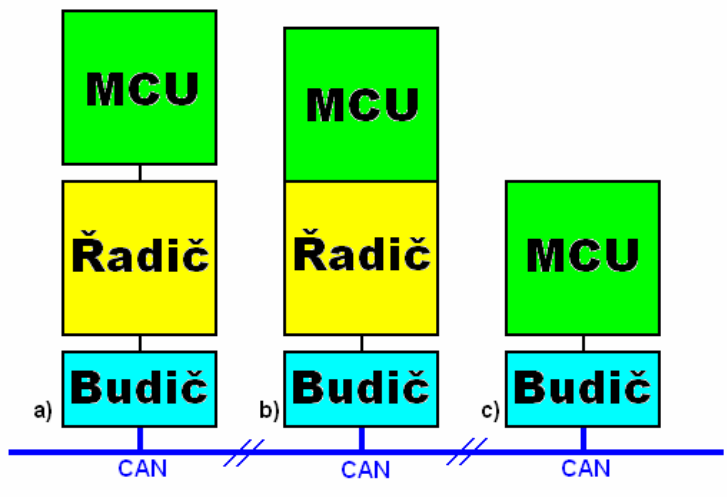
Prvky realizující provoz sběrnice CAN se nazývají řadiče. Kontroléry do nich uloží potřebná data vnějšího charakteru a řadiče z nich vytvoří komunikační protokol CAN. Data která jsou potřeba pro provoz sběrnice, si dopočítá řadič sám. Jsou to kupříkladu: CRC kód, počet datových bytů, Stuff bity, řídící bity, kvitování... Řadiče většinou nepracují synchronně s kontrolérem. Ten do nich uloží hodnoty dat a řadič si sám řídí vlastním hodinovým signálem sběrnici. Tak není kontrolér zaneprázdněn provozem, data pouze odevzdává a přejímá v celku a může tak vykonávat úkony zpracovávající data. Řadiče v sobě většinou mívají zabudovány filtry, pomocí nichž rozhodují, zda přijatá zpráva se předá dál kontroléru či nikoliv (ve sběrnici CAN musí všechny uzly přijímat data a odpovídat na ně vysílači, i když pro ně přijatá data nemají význam). Mezi nejčastěji používané řadiče patří: Philips 80C200, Philips SJA 1000, Intel 82527... [1].

1.5.3 Kontroléry

Kontroléry buď komunikují se sběrnicí pomocí řadičů nebo mají v sobě zabudován řadič sběrnice CAN. V tom případě se spojují se sběrnicí už jenom pomocí budičů. Kontroléry s

integrovanými řadiči jsou např.: Philips 80592, Philps 80598, Motorola MC68HC05X32, Motorola MC68376, Intel 87C196CA/CB, Microchip PIC 18F2480... [1].

Na Obr. 11 jsou ukázány propoje mezi jednotlivými komponenty obsluhujícími sběrnici CAN. Pro aplikace, které jsou náročné na zpracovávaná data, průchodnost dat i na rychlost, je nejvhodnější zapojení a). Mikrokontrolér zpracovává svoje vstupní data, řadič komunikuje se sběrnicí a budič zajišťuje fyzický převod sběrnice. Kontrolér může reagovat na podnět řadiče, např. pomocí přerušení, přijmout data a pokračovat ve své práci. Zapojení b) ušetří jednu součástku, ale kontrolér se už více musí zabývat provozem sběrnice. Moduly CAN v těchto procesorech udělají velkou většinu "provozní" práce, ale přece jenom se musí kontrolér více starat. Zapojení c) je vhodné pouze pro nenáročné aplikace, kde jsou malé nároky na zpracovávaná data, není požadavek maximální rychlosti a kontrolér nemusí provádět náročné výpočty a zpracování dat. MCU je plně zaměstnán provozem sběrnice a data tak může zpracovávat pouze v omezené době. Z důvodu přesné synchronizace sběrnice je rovněž omezeno použití přerušení běhu programu. Nicméně z výukových důvodů jsem si zvolil pro konstrukci přípravku zapojení varianty c). Můžu tak detailně poznat všechny provozní stavy sběrnice CAN a realizovat všechny reakce na stavy sběrnice.



Obr. 11 *aplikace sběrnice CAN 2.0A*

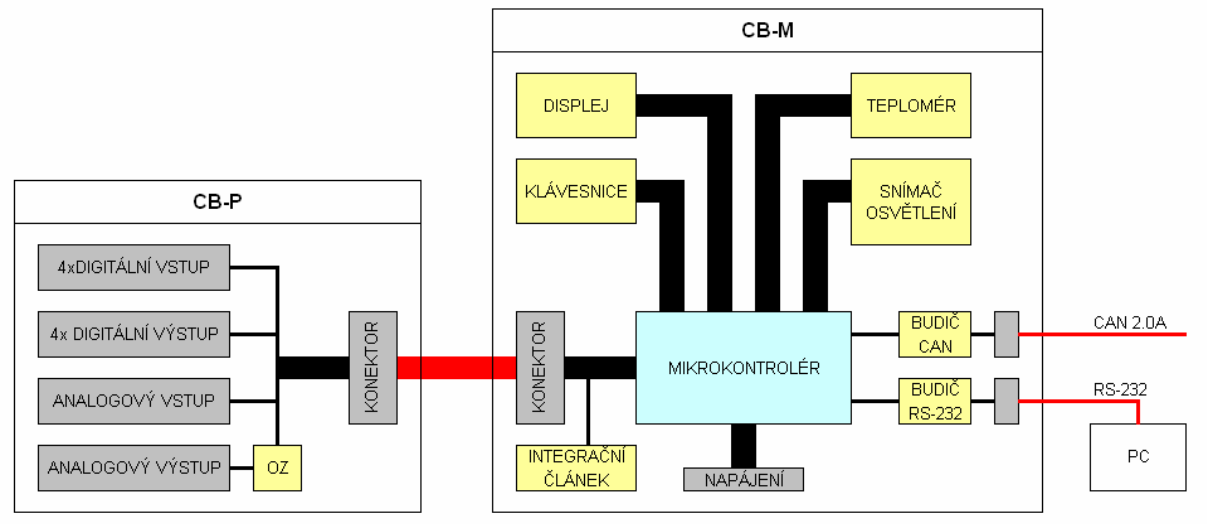
2 Popis přípravku komunikujícího pomocí sběrnice CAN

Laboratorní přípravek je sestaven ze dvou částí. Deska s názvem **CB-M** je hlavní část přípravku. Je schopna komunikace se sběrnicí CAN i se sběrnicí RS-232. Má vstupy a výstupy vyvedeny napřímo na konektor MLW14. Obsahuje tlačítka pro ovládání chodu přípravku a dvouřádkový displej pro zobrazení stavů vlastních periferií. Pro připojení dalších zařízení ke vstupně výstupním vývodům slouží přípravek **CB-P**, který obsahuje ochranné obvody, svorky pro připojení, kontrolní LED diody, zkušební tlačítka a oddělovací OZ.

2.1 Konstrukční popis laboratorního přípravku CB

Laboratorní přípravky CB (CAN Bus) mohou pomocí sběrnice CAN 2.0A přenášet mezi sebou různé typy informací a údajů. Tyto údaje přenášejí pomocí seriové linky RS-232 na PC. Blokové schéma zapojení laboratorního přípravku je na Obr. 12. Přípravek je schopen zobrazit a vyslat tyto informace:

- intenzita osvětlení
- teplota
- analogový vstup 0 ÷ 5V DC
- analogový výstup 0 ÷ 5V DC
- 4x digitální vstup TTL
- 4x digitální výstup O.C.
- diagnostické údaje stavu řadiče



Obr. 12 *blokové schéma laboratorního přípravku*

Laboratorní přípravek je tvořen dvěmi deskami DPS. Řídící deska CB-M a deska periferií CB-P. Hlavní deska DPS (CB-M) obsahuje níže uvedené prvky včetně jejich obvodových zapojení a konektorů:

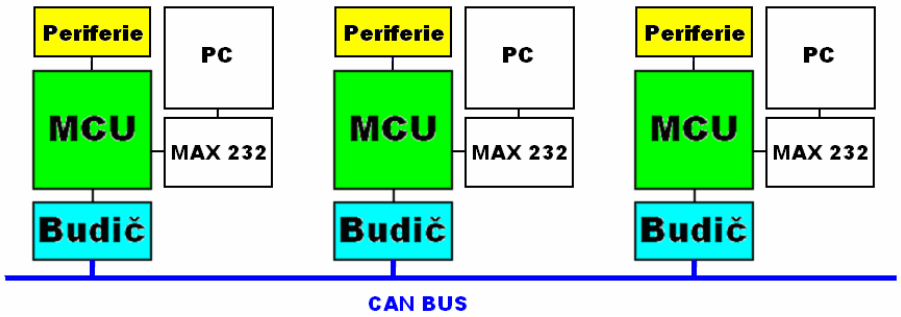
- řadič - ATmega16
- budič sběrnice - PCA82C250
- stabilizátor - 7805 TO220
- snímání osvětlení - fotorezistor VT080
- snímání teploty - snímač I2C - LM75
- displej 16x2 s řadičem - PC1602F
- signalizace provozu sběrnice CAN - LED dioda
- signalizace provozu sběrnice RS-232 - LED dioda
- tlačítka pro ovládání provozu přípravku - 4x P-B1715
- integrační článek PWM D/A převodníku

Deska periferií (CB-P) obsahuje tyto prvky včetně jejich obvodových zapojení:

- operační zesilovač (sledovač signálu K = 1) pro PWM převodník D/A
- ošetřený vstup pro převodník A/D
- 4x ošetřené digitální vstupy s úrovněmi TTL
- 4x výstupy typu otevřený kolektor

Obě desky laboratorního přípravku jsou mezi sebou propojeny pomocí plochého kabelu s konektory MLW14. Toto rozdělení je z důvodu vyšší ochrany vstupů a výstupů laboratorního přípravku. Jednotlivé vstupy a výstupy se mohou vyvést z konektoru základní desky (CB-M) a použít pro jiné aplikace, jsou však neošetřeny a neodděleny. Z důvodu vyšší pravděpodobnosti nevhodného zapojení a následné možnosti zničení obvodů se umístí některé obvody do DPS prostřednictvím patice.

Jednotlivé laboratorní přípravky (CB-M) tvoří uzly sběrnice CAN, které jsou mezi sebou propojeny telefonním kabelem RG-MPFK4 (viz Obr. 13). Ten bude osazen na obou koncích konektorem RJ-14. Maximální počet uzlů je 15 (adresa 0 ÷ F). Adresa každého uzlu se nastavuje softwarově z počítače prostřednictvím hyperterminálu nebo z klávesnice na přípravku. Na desce CB-M je umístěna propojka "Jumper", pomocí které se do obvodu sběrnice zapojí zakončovací rezistor o hodnotě 120 Ω.

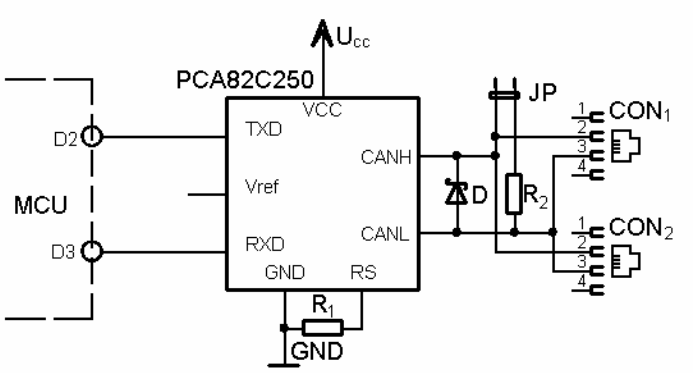


Obr. 13 *schéma propojení přípravků CB*

2.2 Popis dílčích částí obvodů

2.2.1 Budič sběrnice CAN

Budič sběrnice PCA82C250 na Obr. 14 převádí úrovňově i systémově sběrnici CAN na dvoukanálovou sběrnici s úrovněmi TTL. Pin R_{XD} umožňuje přerušení běhu programu (při příjmu), nebo umožňuje verifikaci zapsaných dat (při vysílání). Na pin T_{XD} se budou zapisovat data při vysílání (jinak je vždy v úrovni log.1). Sběrnice je ošetřena transilem o hodnotě U_{RM} = 5,8 V. Rezistor připojený paralelně ke sběrnici pomocí jumperu má hodnotu 120 Ω / 250 mW [11], [12].

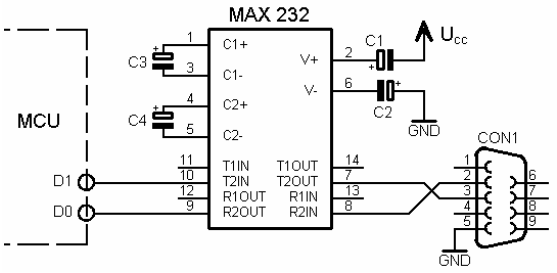


Obr. 14 *schéma zapojení budiče sběrnice*

Jak už bylo popsáno, periferie laboratorního přípravku jsou vyvedeny z desky (CB-P), na které jsou více ochráněny a ošetřeny proti úrovním převyšující kritické hodnoty součástek. Pouze snímání teploty a snímání osvětlení je provedeno přímo na základní desce (CB-M).

2.2.2 Převodník RS-232 / USART

Pro připojení přípravku k PC slouží převodník MAX 232, který převede úrovně RS232 na úrovně TTL a MCU může prostřednictvím jednotky USART a pinů R_{XD} a T_{XD} komunikovat s počítačem - viz Obr. 15. Prostřednictvím PC se vysílají povely a řídí se provoz přípravku. Kondenzátory jsou dle doporučení výrobce o hodnotě 1μF [2], [11], [14].



Obr. 15 *schéma zapojení převodníku RS-232*

2.2.3 Snímání osvětlení

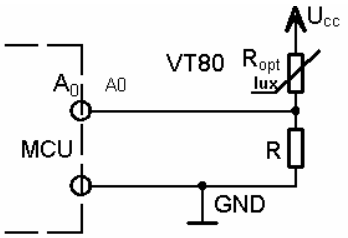
Fotorezistor na obrázku 16 je zapojen do série s obyčejným činným rezistorem a tvoří tak napěťový dělič pro analogový vstup mikrokontroléru. Tento dělič má poměr úměrný intenzitě osvětlení. Pro napěťový dělič platí:

$$U_{\mathrm{R}} = U_{\mathrm{cc}} \frac{R}{R + R_{\mathrm{opt}}} \tag{3}$$

U_{R} je napětí zpracovávané A/D převodníkem, U_{cc} je napájecí napětí, R je hodnota odporu dolní části děliče a R_{opt} je proměnná hodnota odporu v závislosti na osvětlení. Binární hodnota převodu je dána:

$$(\mathrm{ADCH} - \mathrm{ADCL})_2 = 1024 \frac{U_{\mathrm{vst}}}{U_{\mathrm{ref}}} = 1024 \frac{\frac{U_{\mathrm{cc}} R}{R + R_{\mathrm{opt}}}}{U_{\mathrm{cc}}} = 1024 \frac{R}{R + R_{\mathrm{opt}}} \tag{4}$$

Při teoretickém rozkmitu hodnot fotorezistoru R_{opt} = 500 kΩ - 500 Ω (tma - jasné světlo) se při velikosti spodního rezistoru R = 10k bude převod pohybovat mezi 0x014 až 0x3CF. Přibližná hodnota proudu protékající děličem se pohybuje v rozmezí 9,8 μA - 476 μA. To neslibuje příliš přesné výsledky měření, ale vzhledem k nepřesnosti fotorezistoru a jeho logaritmické závislosti odporu na osvětlení, jsou výsledky převodu brány pouze jako orientační. Hodnota osvětlení není vyjádřena v žádných fyzikálních jednotkách, pouze v hexadecimálním vyjádření osmibitového převodu [2], [11], [15].



Obr. 16 *schéma zapojení snímání osvětlení*

2.2.4 Snímání teploty

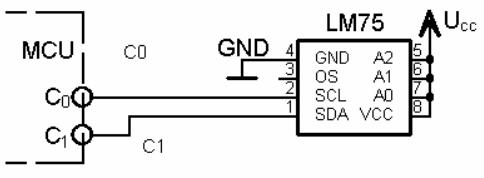
Snímáni teploty je zajištěno obvodem LM75 (Obr. 17). Obvod teplotu nasnímá a pomocí sběrnice I2C (MCU ATMEL ji označují jako sériové rozhraní TWI) komunikuje jako zařízení SLAVE s mikrokontrolérem. Hardwarová adresa zařízení je 1001111. Tato adresa je složená z prefixu daného výrobcem (1001) a volitelné části (111). Protože na sběrnici TWI je umístěn pouze jeden prvek, je lhostejné, jaká se zvolí adresa - tato musí pouze korespondovat se softwarovým zápisem. Sběrnice se bude využívat pouze pro čtení hodnoty teploty - režim Slave, typ Read.

Protokol zjednodušeně vypadá takto:

1.	start	-	(1bit)
2.	adresa	1001111	(7bitů)
3.	bit zápisu na sběrnici R/W	1	(1bit)

4.	potvrzení ACK od LM75	0	(1bit)
5.	čtení dat MSB - LSB	xxxxxxxx	(8bitů)
6.	potvrzení ACK od MCU	0	(1bit)
7.	čtení dat MSB - LSB	xxxxxxxx	(8bitů)
8.	potvrzení NACK od MCU	1	(1bit)
9.	stop	-	(1bit)

Dle Tab. 1 se dá usuzovat, že hodnota kladné teploty je dána hodnotou registru násobeného dvěma. Na pozici LSB značí hodnota 1 rozlišení 1/2°C. Záporné hodnoty bude MCU ignorovat, protože nebudou odpovídat provozním podmínkám ostatních částí laboratorního přípravku [2], [4], [5], [11], [13].



Obr. 17 *schéma zapojení snímání teploty*

Tab. 1 *tabulka příkladů převodu teploty*

Teplota	Bin. hodnota	Hex. hodnota
+125°C	0 1111 1010	0 FA
+25°C	0 0011 0010	0 32
+0,5°C	0 0000 0001	0 01
0°C	0 0000 0000	0 00
-0,5°C	1 1111 1111	1 FF
-25°C	1 1100 1110	1 CE
-40°C	1 1011 0000	1 B0
-55°C	1 1001 0010	1 92

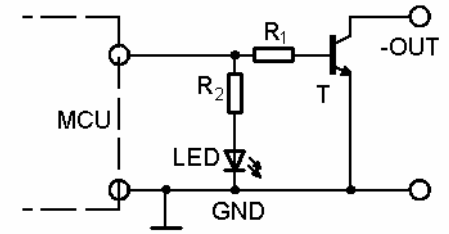
2.2.5 Digitální výstupy

Digitální výstupy jsou na CB-M přítomny na konektoru MLW14 v základní TTL úrovni přímo z výstupu mikrokontroléru. Zapojení ukazuje Tab. 2.

Tab. 2 *tabulka digitálních výstupů*

Číslo výstupu	Pin MLW	Port MCU
1	12	A7
2	11	A6
3	10	C7
4	9	C6

Ochranu a oddělení těchto výstupů realizují obvody na CB-P, které jsou na konci osazeny tranzistorem v zapojení O.C. (otevřený kolektor) - viz Obr. 18. Kontrolní LED diody jsem zapojil na vstup tranzistoru.



Obr. 18 *schéma zapojení digitálních výstupů*

Proud diodou postačuje malý, okolo 5 mA - potom hodnota odporu je dána:

$$R_2 = \frac{U_{cc} - U_{LED}}{I_{LED}} = \frac{4{,}3}{0{,}005} = 860\ \Omega \approx 1\ \text{k}\Omega \tag{5}$$

Protékající proud diodou v propustném stavu je 4,3 mA.

Tranzistor je typu **BC 337-40**. Při I_C = 100 mA je h_{FE} = 300. Maximální zatížení výstupu je I_{CEmax} = 100 mA, při hodnotě P_{tot} = 0,8 W se může provozovat i bez chladiče. Hodnota rezistoru R_1 je dána:

$$R = \frac{(U_{cc} - 0{,}7)h_{FEmin}}{I_{CEmax}} = \frac{4{,}3 * 250}{0{,}1} = 10750\ \Omega \tag{6}$$

Tato hodnota je zbytečně veliká a proud do báze je zbytečně malý (430 μA), proto volím hodnotu rezistoru R_1 = 6k8. Potom celkový proud z pinu MCU je dán:

$$I_{celk} = \frac{U_{cc} - 0{,}7}{R_1} + \frac{Ucc - 0{,}7}{R_2} = (U_{cc} - 0{,}7)\frac{R_1 + R_2}{R_1 R_2} = 4{,}3\frac{78.10^2}{68.10^5} = 4{,}9\ \text{mA} \tag{7}$$

Při současné aktivaci všech výstupů součet vytékajícího proudu nepřesáhne 20mA, což plně vyhovuje mezním podmínkám provozu MCU [4], [11].

2.2.6 Digitální vstupy

Digitální vstupy jsou na CB-M přítomny na konektoru MLW14 v základní TTL úrovni -dle Tab. 3.

Tab. 3 *tabulka digitálních vstupů*

Číslo vstupu	Pin MLW	Port MCU
1	8	C5
2	7	C4
3	6	C3
4	5	C2

Pro ošetření, ochranu a oddělení slouží vstupy na CB-P, které jsou ošetřeny zenerovou diodou a do série je zapojen výkonný rezistor (Obr. 19). Vstupy jsou pro snadnou manipulovatelnost osazeny tlačítky.

Ochranný rezistor má P = 2 W. Jelikož v laboratoři často bývá k dispozici napětí o velikosti U = 24 V, bude se chránit vstup proti náhodnému připojení na toto napětí. Napětí vyšší už bude nad mezní hodnoty.

$$U_R = U_{\max} - U_{ZD} = 26 - 5{,}6 = 20{,}4 \text{ V} \tag{8}$$

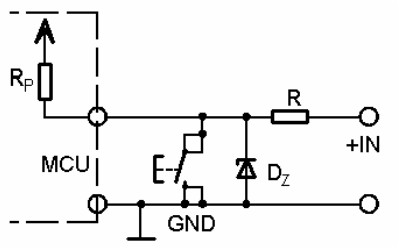
minimální hodnota ochranného rezistoru:

$$R_{\min} = \frac{U_{\max}^2}{P_{tot}} = \frac{20{,}4^2}{2} = 208\ \Omega \tag{9}$$

Volím R = 220 Ω, proud protékající rezistorem je dán:

$$I_R = \frac{U_R}{R} = \frac{U_{\max} - U_{ZD}}{R} = \frac{20{,}4}{220} = 93 \text{ mA} \tag{10}$$

Hodnota rezistoru R = 220 Ω neovlivní činnost vstupu, jelikož pull-up odpory mají hodnotu cca 100 kΩ a proud odebíraný tímto děličem je cca 4μA. Úbytek napětí na tomto rezistoru ve stavu H bude v řádu desítek mikrovoltů [4], [6], [11]. Zenerova dioda je typu: **BZX85V005.6**

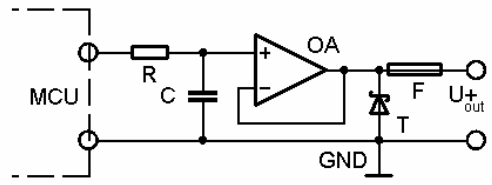


Obr. 19 *schéma zapojení digitálních vstupů*

2.2.7 Analogový výstup

Analogový výstup v rozmezí hodnot 0 - 5 V je tvořen generátorem obdélníku o proměnné střídě, integračním článkem a operačním zesilovačem o přenosu K = 1. Ochrana výstupu je provedena transilem a tavnou pojistkou - viz Obr. 20.

Funkce PWM je jedna z možností čítače TMRx. Při každé shodě čítače s komparátorem se nuluje výstup a při přetečení čítače se výstup nastaví. PWM generátor běží samovolně a nepotřebuje žádnou obsluhu, vyjma vložení vhodné hodnoty do komparátoru.



Obr. 20 *schéma zapojení analogového výstupu*

Průchodem signálu PWM čítače integračním článkem se získá střední hodnota tohoto signálu. Do komparátoru OCRx se nastaví hodnota, která určuje poměr střídy. Čítač je osmibitový, tzn. že hodnota poměru bude v rozmezí 0 - 255. Při napájecím napětí U_{cc} = 5 V bude převodník kvantovat výstup cca krokem 0,02 V. To je zbytečně jemné rozlišení. Postačí, když se výstup bude měnit krokem 0,1 V - tzn. 5x více. Do komparátoru se budou ukládat hodnoty s násobkem 5.

Časová konstanta integračního článku by měla být více jak 10x větší než nejkratší puls čítače PWM. Kmitočet MCU je 16 MHz. Při nastavení předděličky 1/16 se čítač inkrementuje každou 1 μs. Při použití kroku převodníku k = 5, bude nejkratší interval PWM generátoru 5 μs (to bude teoreticky značit hodnotu 0,1 V nebo 4,9 V).

Pro časovou konstantu 8-bitového PWM generátoru platí:

$$\tau = RC \geq \frac{\left(2^8/5\right)10}{f_0 / P_{\text{prescaler}}} = \frac{256/5*10}{16.10^6/16} = \frac{51}{1.10^6} = 51\,\mu\text{s}\ . \tag{11}$$

Pro účely tohoto integračního článku mohou vyhovět tyto parametry R = 2k2 a C = 39n. Časová konstanta bude:

$$\tau = RC = 2{,}2.10^3 * 39.10^{-9} = 86\,\mu\text{s}\,. \tag{12}$$

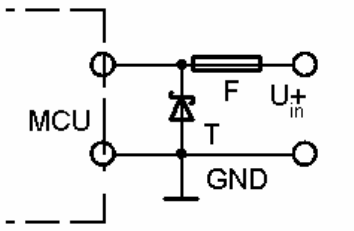
Což vyhovuje podmínce t = 5 μs. Operační zesilovač je zapojen jako sledovač pro oddělení výstupu a následného vyššího zatížení. Na výstupu je paralelně vložen transil typu: **BZW06-5V8-6** se zlomovým napětím U_{RM} = 5,8 V**.** Do série na výstup je vřazena tavná pojistka, která se přetaví při vyšším napětí než je hodnota transilu [2], [6], [11].

2.2.8 Analogový vstup

Analogový vstup na obrázku 21 vede přímo do MCU bez dalších obvodových prvků. Jediným ochranným prvkem je transil v kombinaci s tavnou pojistkou. Tyto součástky mají stejné hodnoty jako u analogového výstupu. Převodník převádí v poměru:

$$(\text{ADCH} - \text{ADCL})_2 = 1024\frac{U_{\text{vst}}}{U_{\text{ref}}} = 1024\frac{U_{\text{vst}}}{5} = 204{,}8 * U_{\text{vst}}\,. \tag{13}$$

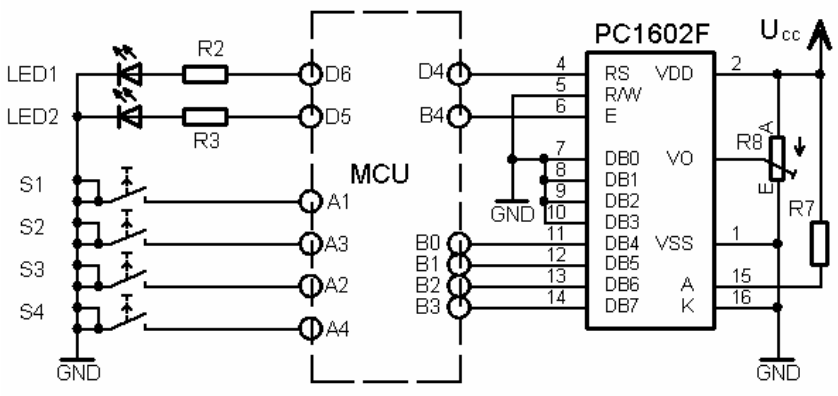
Jelikož napěťová reference bude vzata z napájecího napětí, nebude se jednat o přesný převod, ale spíše o orientační měření. Z toho důvodu není nutné použít pro rozlišení převodu všech 10 bitů A/D převodníku, ale pouze horních 8 bitů [2], [11].



Obr. 21 *schéma zapojení analogového vstupu*

2.2.9 Ovládání přípravku a zobrazování

Obsluha přípravku je prováděna prostřednictvím kláves a displeje s řadičem HITACHI, jak je naznačeno na obrázku 22. Pomocí 4 kláves (Up, Down, Esc, Enter) se bude ovládat provoz přípravku. Klávesy jsou připojeny přímo a ošetření zákmitu je provedeno softwarově. Displej je navržen v podsvětleném provedení o velikosti 2x16. Ovládání displeje jsem z důvodu úspory pinů na kontroléru provedl po čtyřech datových linkách. Ze stejného důvodu není umožněno z displeje číst (pin R/W je natrvalo napojen na log. 0). Provoz sběrnice CAN je signalizován oběma LED diodami. Dioda LED1 zobrazuje vysílání na sběrnici a dioda LED2 zobrazuje snímání sběrnice. Trimr je zapojen pro řízení kontrastu displeje [4], [5].



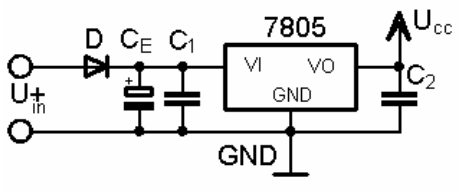
Obr. 22 *schéma zapojení ovládání a zobrazování*

2.2.10 Stabilizace napájení

Stabilizační obvod (Obr. 23) je velmi jednoduchý, sestávající se z usměrňovací diody jako ochrany proti přepólování, stabilizátoru a filtrace. Stabilizátor je typu 78S05 s maximálním odběrem proudu I_{out} = 2 A. Odběr proudu nepřesáhne hodnotu 1 A (viz Tab. 4). Filtrační kondenzátory na vstupu a výstupu jsou umístěny u stabilizátoru a mají hodnotu C =100 nF. Na vstupu je rovněž filtrační elektrolytický kondenzátor o vyšší kapacitě. Jelikož se předpokládá, že se bude napájet odfiltrovaným stejnosměrným napětím, nemusí být filtrační kondenzátor velké kapacity.

Tab. 4 *napájecí parametry přípravku*

Napájecí napětí	12	V
Napájecí proud bez desky CB-P	52,5	mA
Napájecí proud s deskou CB-P	53,3	mA
Celkový příkon s deskou CB-P	639,6	mW



Obr. 23 *schéma zapojení stabilizace napájení*

2.3 Konstrukce přípravku sběrnice CAN

Přípravky **CB-M** a **CB-P** jsou navrženy v editoru EAGLE v jednovrstvém provedení. Napájecí spoje jsou provedeny o šířce 1,27 mm a 0,8 mm. Ostatní signálové propoje jsou šířky 0,4 mm. Součástky jsou aplikovány v povrchovém i klasickém provedení, takže jsou osazeny z obou stran DPS. Obvody realizující spojení s cizími technologiemi jsou zasunuty do patic (ATmega a MAX232). Displej je umístěn na sloupcích nad svorkovnicemi sběrnic z důvodu úspory místa na DPS.

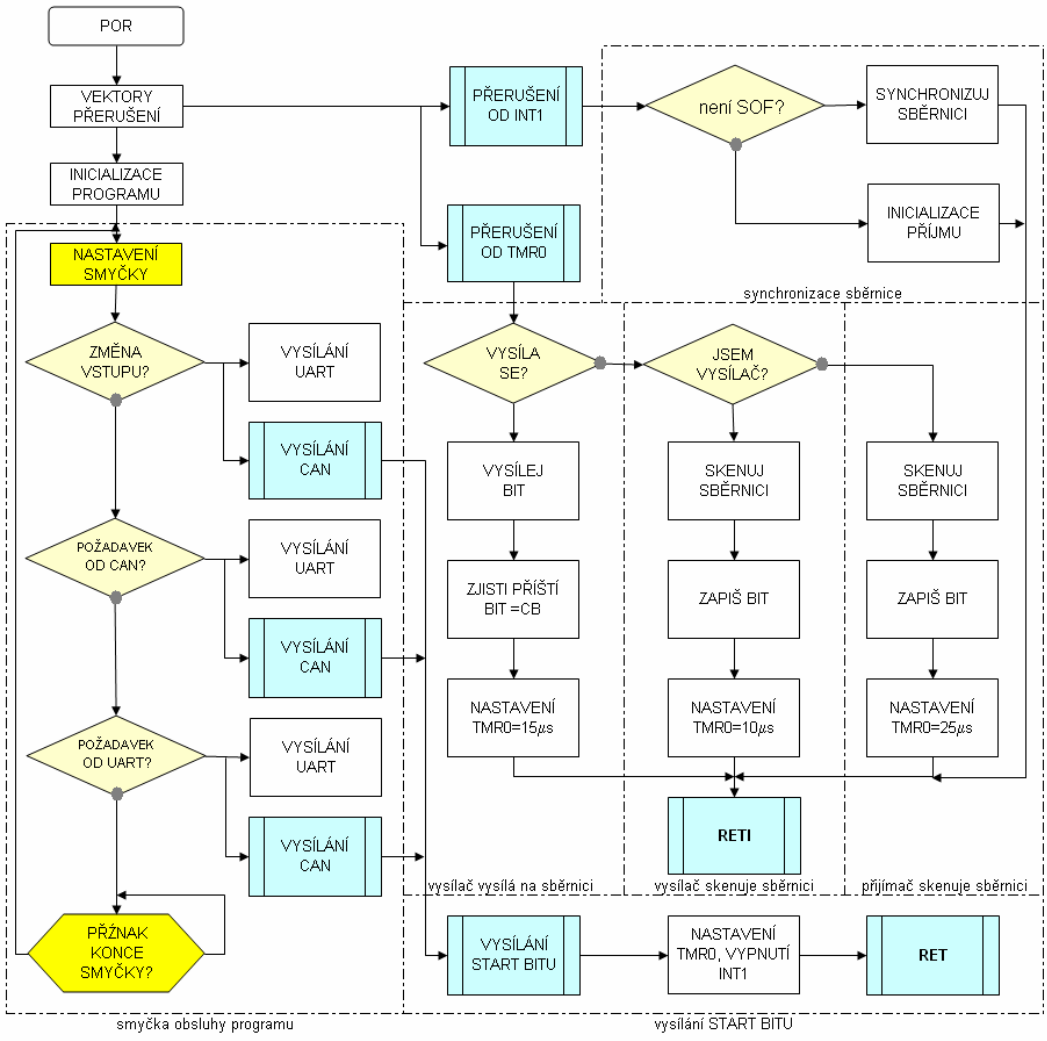
Dokumentace celého přípravku je umístěna v příloze, jedná se o tyto části:

- Schéma přípravku CB-M (hlavní DPS)
- DPS přípravku CB-M ze strany plošného spoje (rozměr:100x90mm)
- Osazování součástek přípravku CB-M - TOP (rozměr:100x90mm)
- Osazování součástek přípravku CB-M - BOTTOM (rozměr:100x90mm)
- Seznam součástek přípravku CB-M
- Schéma přípravku CB-P (DPS-periferie)
- DPS přípravku CB-P ze strany plošného spoje
- Osazování součástek přípravku CB-P ze strany TOP
- Seznam součástek přípravku CB-P

3 Program ovládání laboratorního přípravku

Laboratorní přípravek komunikující pomocí sběrnice CAN je obsluhován pouze jedním programovatelným prvkem - mikrokontrolérem ATmega16. Z toho důvodu je použit pouze jeden program, který je napsán v assembleru. Mikrokontrolér má plně využity všechny vstupně/výstupní porty a pracuje na maximální rychlost, tedy f_{osc} = 16 MHz. Paměť flash má zaplněnou cca ze 33 % a datový prostor RAM je obsazen z 11 %. Celý program jsem volil tak, aby byl univerzální a modul obsluhující sběrnici CAN byl využitelný pro případné další aplikace. Vývojový diagram celého ovládacího programu je na Obr. 24.

pozn.: Na první pohled je zřejmé, že není účelné pracně vytvářet něco, co se dá běžně pořídit. Celý projekt byl směřován na popis a pochopení sběrnice CAN a na vytvoření aplikace, pomocí níž komunikuje laboratorní přípravek. K vytvoření vlastní aplikace obsluhy CAN mě vedly dva důvody. Chtěl jsem se naučit programovat kontroléry ATMEL. Druhý důvod spočíval v tom, že pokud bych použil standardní řadič či mikrokontrolér s modulem CAN, umožnilo by mi to komunikaci mnohem snázeji (a možná i kvalitněji), ale nevyužil bych při tom vědomostí nabytých studiem této sběrnice.



Obr. 24 *zjednodušený diagram programu přípravku*

Běh programu tvoří dva celky. Jednu část tvoří obsluha přípravku. Ta umožňuje snímání stavu vstupů a tlačítek, a vykonání jejich příp. požadavků. Dále umožňuje řešit požadavky od

modulu USART a modulu CAN. Druhou částí programu je modul CAN, který umožňuje inicializaci, provoz a ukončení sběrnice.

Program pro obsluhu laboratorního přípravku využívá tyto moduly mikrokontroléru:

- časovač TMR0 - časování sběrnice CAN
- časovač TMR1 - časování hlavní smyčky programu
- časovač TMR2 - PWM regulátor pro generování výstupního napětí
- přerušení od vstupu INT1 (PORTD,3) - vstupní pin sběrnice CAN
- přerušení CTC časovače TMR0 - časování sběrnice CAN
- modul USART - komunikace s počítačem
- modul TWI - komunikace s digitálním teploměrem
- modul A/D převodník - snímání vstupu a snímání napětí na fotorezistoru
- sériový kanál SPI - ISP programování mikrokontroléru

3.1 Program obsluhy přípravku

Program pro obsluhu přípravku je nejobjemější část celého programu. Jeho vývojový diagram je zobrazen na Obr. 25. Je tvořen inicializačními instrukcemi a hlavní smyčkou. Po úvodní inicializaci jednotlivých modulů, naplnění registrů a inicializaci LCD displeje vyšle program na sériovou linku hlášení o zapnutí uzlu, povolí přerušení a "skočí" do hlavní smyčky. Jelikož obsluha sběrnice CAN je citlivá na synchronizaci, musí být v programu téměř vyloučena obsluha přerušení od jiných podnětů než od sběrnice CAN. Z toho důvodu je použito jediné přerušení v případě příjmu dat po sériové lince. Doba trvání hlavní smyčky je 100 ms z důvodu optimální doby pro zjišťování stavu tlačítek a digitálních vstupů.

Ukončení smyčky se neprovede prostřednictvím přerušení od časovače, nýbrž testováním příznaku přerušení časovače TMR1 - polling. Z toho důvodu může být doba trvání smyčky i delší, což je z hlediska běhu programu nepodstatné.

3.1.1 Popis bloků programu přípravku

Pro větší přehlednost popíšu jednotlivé operace programu z chronologického hlediska:

Inicializace smyčky

Na začátku smyčky se zruší příznak přetečení časovače a časovač se naplní konstantou, která zajistí přetečení časovače za dobu 100 ms.

Snímání stavu tlačítek

Program načte stav tlačítek a porovná jej se stavem, který byl minulou smyčku. Pokud najde shodu, dotazuje se prostřednictvím příznakového bitu, zdali už program vykonal rutinu obsluhy stisku. Jestliže byl stisk tlačítka obsloužen, program žádnou činnost už nekoná a skočí na další pole. Když se obsluha stisku ještě nevykonala, nastaví se příznak a požadavek se vykoná. Jestliže není nic stisknuto, program zruší příznak, vloží do paměti stav tlačítek a skočí na obsluhu dalšího pole. Pokud je stisknuta jiná sekvence než minule, program nevykoná žádnou činnost - ta se vykoná, až bude tlačítko uvolněno a opět dojde k dalšímu stisku libovolného tlačítka.

Jestliže má program vykonat nějaký příkaz od tlačítka, dotazuje se na adresu komunikačního uzlu. Pokud je adresa pro komunikaci totožná s vlastní adresou, neposílá uzel na sběrnici CAN žádnou informaci, ale pouze zobrazuje svůj vlastní stav. V opačném případě zašle na sběrnici požadovaný výsledek. Na sériovou linku se posílají informace i když uzel pouze zobrazuje stavy svých vlastních periferií.

Snímání stavu digitálních vstupů

Program načte stav digitálních vstupů a porovná je se stavem v minulé smyčce. Pokud je shoda, nevykoná žádnou činnost, pokud najde rozdíl, vykoná rutinu obsluhy. Program se opět dotazuje na adresu komunikačního uzlu a na něj pošle informaci o stavu všech vstupů. Pokud je adresa komunikace shodná s vlastní adresou, zobrazí stav vstupů na svoje vlastní výstupy.

Dotaz na požadavek přicházející ze sběrnice CAN

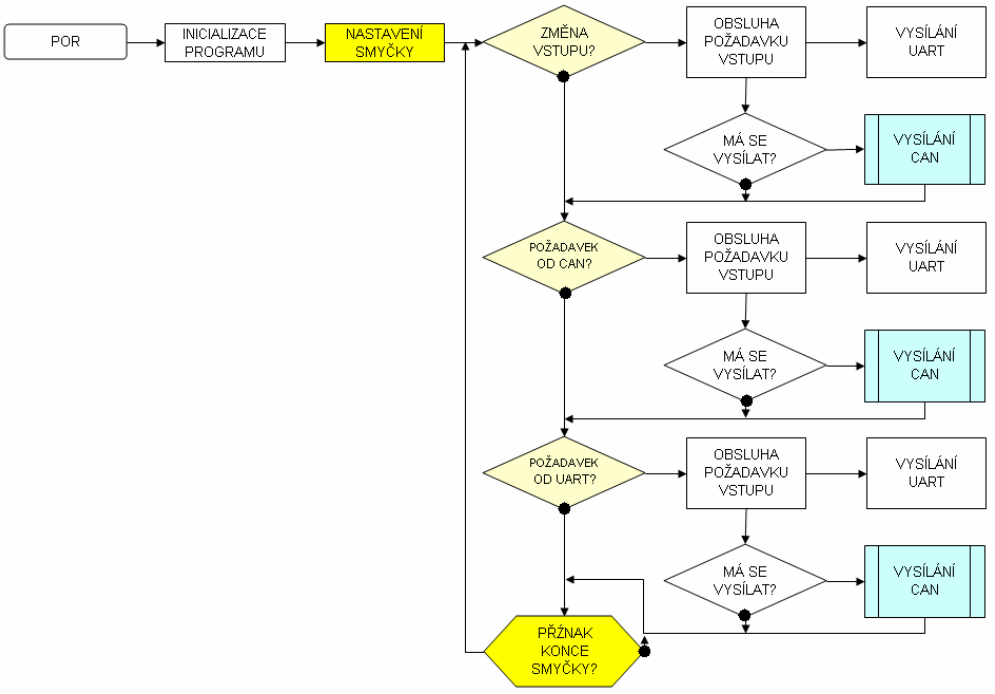
Protože průchody přerušením musí být co možná nejkratší, nemůže program vykonat obsluhu požadavku od sběrnice CAN přímo v obsluze, ale pouze nastaví příznak, že sběrnice přijala data, popřípadě že se nepřenesla korektně. V tomto místě program testuje příznakové bity a pokud je potřeba, vykoná patřičnou obsluhu.

Dotaz na požadavek přicházející z PC

Ze stejného důvodu, jako u obsluhy požadavků sběrnice CAN, se obsluhuje požadavek od sériové linky pomocí příznaku. Pokud je příznak nastaven, vyřeší se požadavek sériové linky právě zde. Při každém vyřešení se samozřejmě příznak požadavku nuluje.

Čekání na příznak přetečení časovače

V tomto místě se obsluha v nekonečné smyčce dotazuje na příznak přerušení od přetečení časovače TMR1. Protože registr příznaků TIFR leží mimo interval prvních 32 registrů, musí se obsluha dotazovat dvoutaktově. Natáhne se obsah I/O registru TIFR do pomocného registru a teprve potom se doptá na stav přetečení. V případě nastavení bitu TOV1, program skočí na návěští začátku smyčky [2], [3], [7], [11].



Obr. 25 *vývojový diagram programu pro obsluhu přípravku*

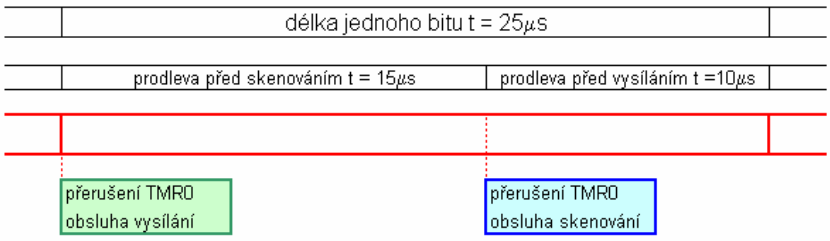
3.2 Program obsluhy sběrnice CAN

Tato část programu není tak náročná na objem dat a instrukcí, ale o to více je náročná na přesnost a propracovanost. Aby byla rychlost sběrnice CAN co možná největší, je nutné, aby byla její obsluha co nejefektivnější. To byl důvod, proč jsem program psal přímo v assembleru a ne pomocí jazyka C. Kmitočet sběrnice jsem chtěl docílit o velikosti 50 kHz. Po dlouhém úsilí jsem se musel této myšlenky vzdát, protože obsluha sběrnice v mnohých místech převyšovala limit, který se musí dodržet pro přesnou synchronizaci. Konečný kmitočet sběrnice CAN je f = **40 kHz**. Tento kmitočet je dostatečně rychlý a přitom poskytuje programu dostatek času pro vykonání ostatních operací.

Přístup k obsluze sběrnice CAN je zprostředkován pomocí přerušení od TMR0, od INT1 a pomocí jednoho podprogramu. Z hlediska provozu se samozřejmě rozlišuje vysílání a příjem protokolu CAN. Pro chod sběrnice jsou určeny dva registry CANREG a CANSTAT, ve kterých jsou příznakové bity. Ty spolu se třemi pomocnými registry a 43 datovými registry slouží k provozu programu obsluhy sběrnice CAN.

3.3 Vysílač sběrnice CAN

Vysílač sběrnice CAN zahajuje činnost vysláním bitu SOF prostřednictvím podprogramu "`can1st`". Ten iniciuje všechny potřebné kroky ke správnému chodu sběrnice a opustí se standardní instrukcí "`ret`". Následně potom je vyvoláno přerušení od časovače (po 15 µs), ve kterém program skenuje sběrnici a testuje shodu s vysílanou hodnotou. Naskenovaný bit je uložen do přijímacích registrů. Další přerušení od časovače (po 10 µs) vyvolá zapsání bitu. Ten se musí určit vždy v předchozím zápisu, aby sběrnice byla přesná. Po vysílání se určí příští bit pro další vyslání - a takto se to cyklicky opakuje (Obr. 27).



Obr. 26 *časové schéma vysíláni jednoho bitu CAN*

Pokud má program obsluhy přípravku potřebu komunikovat, musí se nejprve uložit data do registrů, které se budou přenášet. Jsou to podle Tab. 9 všechny registry kromě `txcancrc` (datových registrů se uloží kolik je potřeba, a do registru `txcand` se zapíše jejich počet). Po zapsání obsluha skočí na podprogram "`can1st`". Tento podprogram je spolu s obsluhou obou přerušení (TMR0 a INT1) vložen v souboru CAN2.inc [2], [3], [7], [11].

Tab. 5 *vysílací registry z datového pole*

CANBIT								
bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
TD5	TD4	-	-	-	-	-	-	txcana
příkaz				adresa uzlu				txcana1
PBIT	RTR	R1=0	R0=0	počet přenášenách dat				txcand
bit7 ÷ bit0								txcand1
bit7 ÷ bit0								txcand1
bit7 ÷ bit0								txcand1
bit7 ÷ bit0								txcand1
bit7 ÷ bit0								txcand1
bit7 ÷ bit0								txcand1
bit7 ÷ bit0								txcand1
bit7 ÷ bit0								txcand1
CRC								txcancrc
CRC								txcancrc1

3.3.1 Inicializace vysílání

Podprogram "`can1st`" nejprve testuje, zda je sběrnice volná a to pomocí příznaku `CANBUS` v registru `CANSTAT`. Pokud je sběrnice volná, uvede ji do dominantního stavu a zahájí tím SOF (Start of Frame). Následně potom podprogram zakáže přerušení od vstupu INT1 a povolí provoz časovače TMR0. Naplní se komparátor hodnotou přerušení, nastaví se všechny potřebné příznaky provozu včetně příznaku příštího vysílacího bitu a nakonec se naplní registry RAM potřebnými hodnotami.

Jsou to tyto registry:

- **`txcanbit`** - registr počtu bitů v přenosovém bytu
- **`txcanrot`** - registr vysouvaní dat pomocí rotace
- **`txpoint`** - pointer na byte, ze kterého se právě vysílá
- **`txstuff`** - rotační registr pro účely výpočtu Stuff Bitu

Po naplnění registrů se opustí podprogram instrukcí "ret".

3.3.2 Vysílač skenuje sběrnici

Podle Obr. 27 po době trvání 15 μs nastane přerušení od časovače TMR0. Dotázáním příznaku `SCAN` v registru `CANREG`, se program přesune na obsluhu skenování sběrnice. Vysílač vysílaná data skenuje a zjišťuje shodu, pokud zjistí kolizi, obslouží ji buď vysláním rámce Error Frame nebo přepnutím do módu přijímače, přestane data vysílat a stává se pouze přijímačem.

Skenování sběrnice vysílačem má tyto bloky:

Uložení hodnot obslužných registrů

Z důvodu okamžité reakce vysílání bitu se uložení pomocných registrů a registru `SREG` do zásobníku (instrukce `push`) odkládá až do místa, kde to synchronizaci sběrnice nevadí. Pro

výkon obsluhy přerušení jsou použity registry POM, POM1 a POM2. Do zásobníku se rovněž uloží hodnota registru SREG.

Skenování

V této části obsluhy se naskenuje stav sběrnice a zkopíruje se na bit T v registru SREG.

Testování pole

V tomto místě program testuje, zdali se nyní obsluhuje bit ACK (důležitý právě pro vysílač), zdali probíhá pole EOF nebo zdali je na sběrnici Error Frame. Zjišťuje se to pomocí příznaků v registru CANSTAT. Jsou to bity: ACKRX, EOFTX, a ERRFR. Po zjištění stavu program skočí na patřičné návěští.

Obsluha Stuff Bitu

Pro zjištění Stuff Bitu je potřeba pomocný rotační registr a příznak v registru CANREG, SBRX. Přijaté bity se rotují přes bit C do registru rxstuff. Pokud nastane po vymaskování registru hodnotou 0x1F stav, že mají hodnotu 0x00 nebo 0x1F, přijatý bit není datový, je pouze synchronizační, neboli - jedná se o Stuff Bit. V tom případě se bit nezapíše, nastaví se příznak přijatého Stuff Bitu a obsluha přejde na testování shody. Pokud není přijatý bit Stuff Bit, obsluha zarotuje nově přijatý bit do registru rxstuff a pokračuje na další pole. Jestliže obsluha zjistí, že minule přijatý bit byl Stuff Bit (pomocí příznaku SBRX), nastaví rotační registr na hodnotu 0xFF (pokud je současně přijatý bit s hodnotou 0) nebo na hodnotu 0x00 (pokud má nynější bit hodnotu 1). Pokud program zjistí chybu Stuff Bitu, skočí na obsluhu této chyby. Jestliže je chyba s úrovní log. 0 (dominantní stav), jedná se pravděpodobně o vysílání Error Frame nebo o vysílání Overload Frame (pokud se chyba nachází v šestém bitu registru rxcana1).

Zápis naskenovaného bitu

Naskenovaný bit se zapíše rotací podobně jako v minulém poli, tentokrát do registru rxcanrot. Zároveň se dekrementuje hodnota registru rxcanbit. Pokud se dosáhne hodnoty 0, přenos bytu se dokončil, registr rxcanrot se uloží na místo určené pointerem rxpoint. Registr rxcanbit se naplní hodnotou 0x08 a program jde najít další byte, do kterého se bude příště ukládat. Při této příležitosti zjišťuje, kolik se přenáší datových bytů, zdali se už přenáší hodnota CRC, jestli je ukončen přenos, zdali se bude přenášet bit ACK a následně potom se bude přenášet EOF. Všechny tyto zjištěné skutečnosti program v tomto bloku ošetří.

Výpočet CRC

Pro výpočet CRC slouží registry rxcrc a registry rxcrc1. Výpočet je proudový, tedy bit po bitu. Když jsem promýšlel postup výpočtu, nikde jsem v literatuře nenarazil na podobný postup. Většinou se počítal CRC až po přijetí celého objemu dat a to pomocí tabulek. Přitom výpočet je velmi jednoduchý. Naskenovaný bit se zarotuje do registru rxcrc1 a přes bit C i do registru rxcrc. Pokud má MSB bit MSB bytu hodnotu 1, obsluha provede operaci XOR s polynomem 0xC599. Jestliže ovšem dotazovaný bit nemá hodnotu 1, nechají se zarotované registry na hodnotě po rotaci a uloží se. Pro účely přenosu se dle protokolu CAN "zahodí" na konci výpočtu MSB bit, oba registry se zarotují doleva a na pozici LSB se dosadí hodnota 1.

Testování shody sběrnice

Poslední obslužná rutina je jednoduchá. Testuje, zda se shoduje vysílací pin `Tx` a přijímací pin `Rx`. Pokud ano, je vše v pořádku, pokud ne, jde se řešit kolize. Kolize se řeší buď vysláním Error Frame, přepnutím vysílače na přijímač nebo hlášením o chybě budiče a následném uvedení do stavu BOFF (Bus Off).

Nastavení ukončení přerušení

V této sekci program nastaví časovač na přerušení po 10 μs a nastaví příznak zrušení skenování (`SCAN=0`). Následně se vyjmou hodnoty registrů uložených do zásobníku a program opustí přerušení instrukcí "`reti`". Maximální doba přerušení 6,5 μs, tzn. že program obsluhy má minimálně 50 instrukcí před následujícím přerušením k dispozici.

3.3.3 Vysílač vysílá na sběrnici

Jak už bylo popsáno, vysílání prvního bitu (SOF - dominantní stav) je zahájeno podprogramem "`can1st`". Ten zajistí nastavení časovače tak, že každých dalších 25 μs vyvolá přerušení pro vysílání. V přesně definovaném a hlavně konstantním čase musí aplikace po přerušení vysílat nový bit. Z toho důvodu modul vysílače CAN zjišťuje ihned po odvysílání současného bitu hodnotu bitu příštího. Ta je uložena v registru `CANREG,` bit `CB`. Zjišťování příštího vysílaného bitu není z časového hlediska konstantní úkon, ale protože se to odehrává až po odvysílání, má aplikace na tuto operaci dostatek času. Ve skutečnosti délka přerušení pro vysílání nepřesáhne hodnotu 6,5 μs. To dává dostatečný prostor pro výpočet operací obslužného programu, neboť další přerušení nastane po 15 μs od zahájení vysílání.

Vysílání na sběrnici vysílačem má tyto bloky:

Testování pole

Zde program testuje, zda obsluhuje provoz EOF nebo zdali neprobíhá přenos Error Frame či Overload Frame. Toto se zjišťuje pomocí příznaků v registru `CANSTAT`. Jsou to bity: `EOFTX, ERRFR a OVERFR`. Bit ACK není v této fázi důležitý, protože vysílač vysílá recesivní úroveň (log. 1). Hodnota ACK se bude zjiťovat až při dalším skenování sběrnice vysílačem.

Testování Stuff Bitu

Pro zjištění Stuff Bitu je potřeba pomocný rotační registr `rxstuff` a příznak v registru `CANREG, SBTX`. Pokud dostaneme po vymaskování registru `rxstuff` hodnotou 0x1F hodnotu 0x00 nebo 0x1F, příště vysílaný bit není datový, je pouze synchronizační, neboli jedná se o Stuff Bit. V tom případě se další bit nezjišťuje, nastaví se příznak vysílaného Stuff Bitu a obsluha přejde na ukončení přerušení. Pokud není přijatý bit Stuff Bit, obsluha programu přejde na další pole.

Zjišťování příštího vysílaného bitu

Bity se vysílají prostřednictvím rotace registru `txcanrot`. Jejich počet v bytu je určen registrem `txcanbit`. Tento registr se dekrementuje a pokud dekrementací dosáhnul nulového výsledku, musí se natáhnout další vysílaný byte do registru `txcanrot`. V tom případě se zjišťuje počet přenášených datových bytů, a také zdali se program nepřesune (příštím vysíláním) do jiného pole (Arbitrační pole - Datové pole - CRC - ACK - EOF). V

každém případě se naplní registr `txcanbit` hodnotou 0x08, nastaví se pointery a registr `txcanrot` se naplní dalšími osmi bity. Příště vysílaný bit se získá rotací registru `txcanrot` vlevo a to z pozice MSB. Získaná hodnota se uloží do registru `CANREG`, bit `CB`.

Obsluha Stuff Bitu

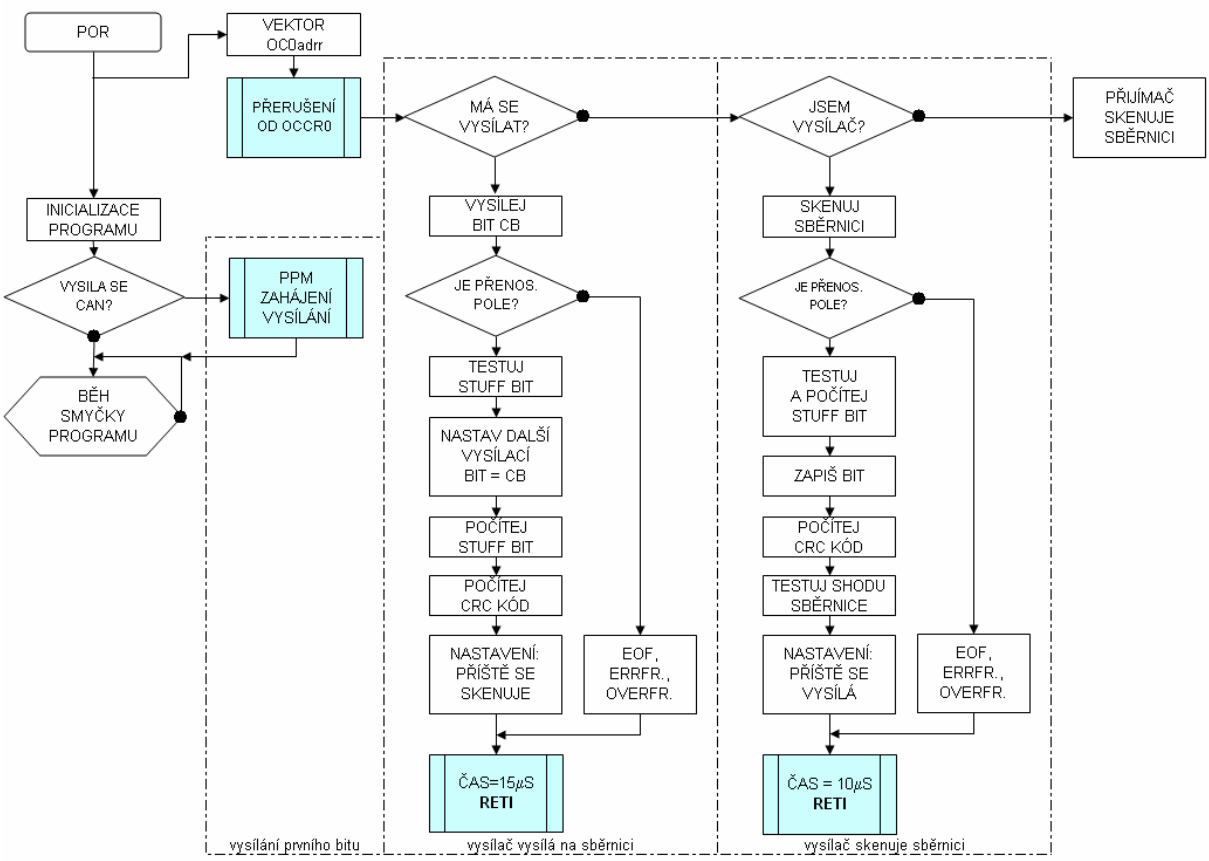
Příští bit (`CB`) se rotuje přes bit C do registru `txstuff`. Nic dalšího v tomto kole program neřeší, pouze pokud zjistí, že minule vysílaný bit byl Stuff Bit (pomocí příznaku `SBTX`), nahraje do registru txstuff hodnotu 0x00 (pokud je bit `CB` v úrovni H) nebo hodnotu 0xFF (pokud je bit `CB` v úrovni L).

Výpočet CRC

Pro výpočet CRC slouží registr `txcancrc` a registr `txcancrc1`. Výpočet je stejný jako v bloku skenování.

Nastavení ukončení přerušení

V této sekci program nastaví časovač na přerušení po 15 µs a nastaví příznak skenování (`SCAN=1`). Následně se vyjmou hodnoty registrů uložených do zásobníku a program opustí přerušení instrukcí "`reti`".



Obr. 27 *diagram obsluhy vysílače*

3.4 Přijímač sběrnice CAN

V klidu je na sběrnici recesivní úroveň (log. 1). Po ukončení vysílání nebo po zahájení programu (POR) je vypnutý modul časovače TMR0 a zapnuto povolení přerušení od vstupu INT1 (Rx). Příjem sběrnice CAN je zahájen upnutím sběrnice do dominantního stavu (log. 0) a tím dojde k inicializaci přerušení od vstupu (viz Obr.29). Pokud se jedná o příjem bitu SOF, dojde současně k nastavení všech potřebných registrů a modulů.

Obecně příjem sběrnice pracuje tak, že dobu snímání bitu řídí časovač TMR0, ale upřesnění času řídí přerušení od pinu Rx. K tomu dochází minimálně jednou za 5 přijatých bitů, tedy za dobu 125 μs. To je doba, po kterou nesmí dojít k časovému ofsetu v intervalu ± 10 μs. Z toho důvodu protokol CAN používá pro synchronizaci Stuff Bity.

3.4.1 Přijímač se synchronizuje

Celý modul synchronizace je zahájen vektorem přerušení od vstupu INT1, na kterém je připojen pin Rx budiče CAN. Průběh je nakreslen na Obr. 28 a zvýrazněn zeleným blokem. Tento proces je velmi krátký a obsahuje tyto bloky:

Nulování časovače

Před tím, než aplikace vynuluje časovač TMR0, uloží se registry `SREG` a `POM` do zásobníku. Nulování probíhá zápisem registru `POM` s nulovou hodnotou do registru TCNT0.

Testování provozu CAN

V tomto bloku testuje aplikace, zda se jedná o vysílání prvního bitu, neboli SOF. Dotazuje se na tento stav prostřednictvím bitu `CANBUS` v registru `CANSTAT`. Pokud se jedná o zahájení vysílání, tento bit je vynulován a program provede obsluhu všech potřebných úkonů.

Jedná se zejména o:

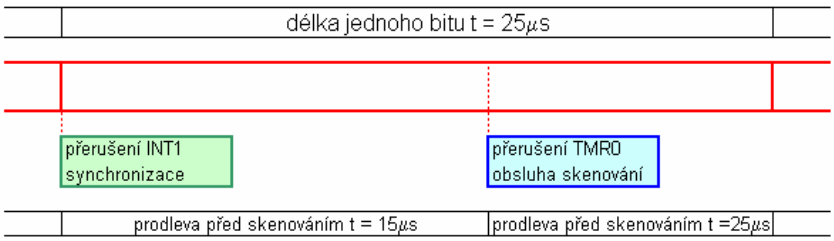
- zapnutí časovače a naplnění hodnotami příštího přerušení
- naplnění stavových registrů `CANSTAT` a `CANREG`
- nastavení registrů `rxcanbit` a `rxcanstuff`
- nulování registrů pro účely výpočtu CRC kódu

Nastavení ukončení přerušení

V tomto bloku se naplní komparátor časovače tak, aby dosáhnul přerušení po 15 μs. Ve skutečnosti se do komparátoru časovače nastaví 14 μs, protože sama obsluha přerušení a uložení registrů do zásobníku probíhá po dobu 1 μs. Nakonec se ze zásobníku vyjmou hodnoty zpět do registrů `SREG` a `POM`.

3.4.2 Přijímač skenuje sběrnici

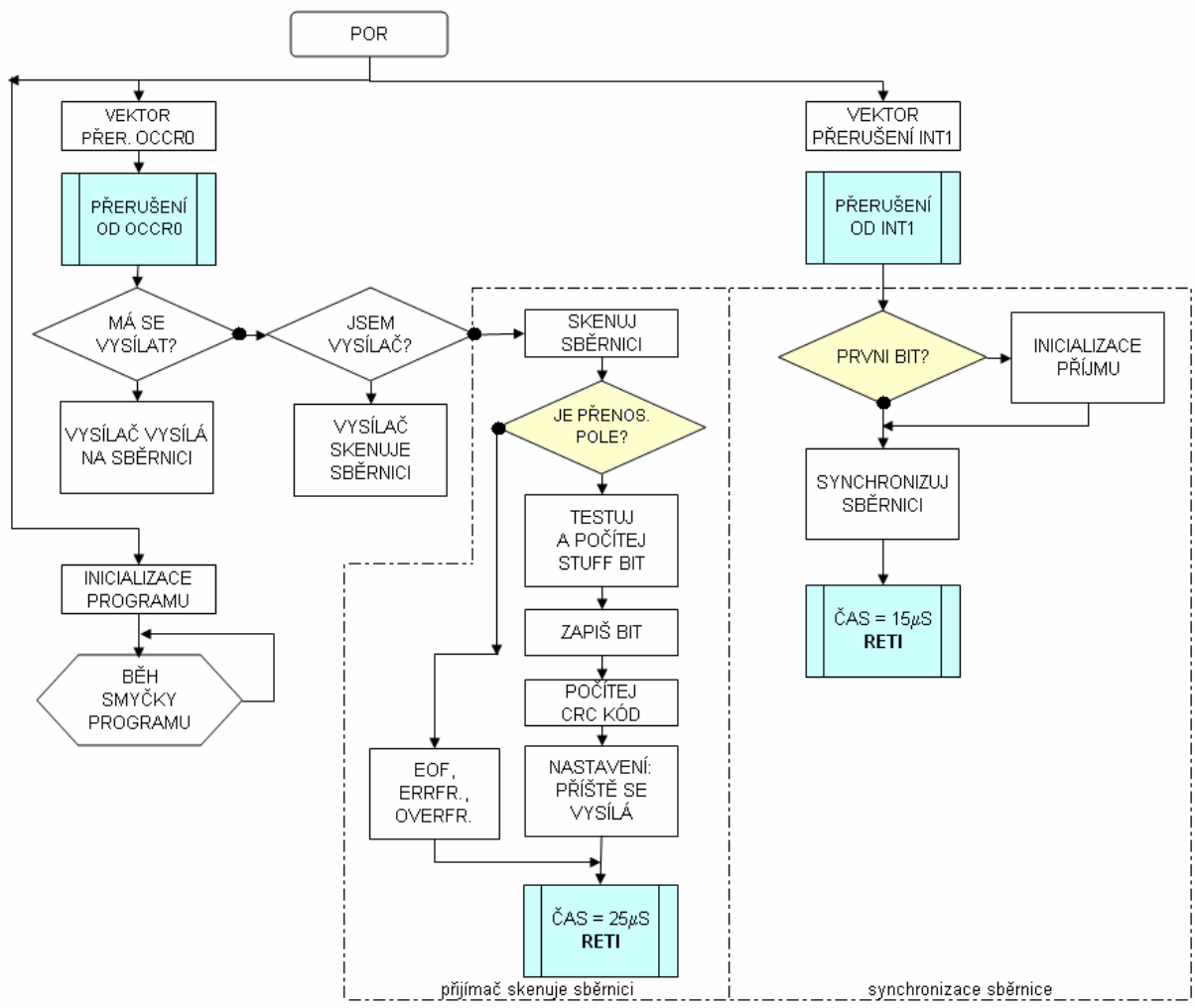
Celý tento modul je velmi podobný modulu skenování sběrnice vysílačem. Liší se pouze v tom, že přijímač netestuje kolize sběrnice, a ve vyhodnocení bitu ACK. Zatímco vysílač testuje, zdali je ACK kladné, přijímač se podílí na vysílání tohoto bitu (log. 0) a tím vysílači potvrdí správnost (nebo neshodu - log. 1) příjmu. Na Obr. 28 je časové rozmezí skenování bitu znázorněno modrým blokem.



Obr. 28 *časové schéma příjmu jednoho bitu CAN*

Procesy příjmu sběrnice CAN:

- uložení hodnot obslužných registrů
- skenování sběrnice
- testování pole
- obsluha Stuff Bitu
- zápis naskenovaného bitu
- výpočet CRC
- nastavení ukončení přerušení



Obr. 29 *diagram obsluhy přerušení vysílače*

4 Popis ovládání laboratorního přípravku

4.1 Popis obsluhy přípravku

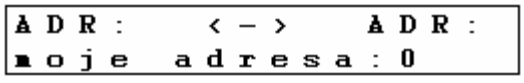
Laboratorní přípravek umožňuje snímat stavy ze svých periferií, popř. přijímat informace z PC pomocí sběrnice RS-232 a tyto informace posílat na další laboratorní přípravek. Jednotlivé přípravky jsou spojeny pomocí dvouvodičové sběrnice CAN a pomocí protokolu CAN 2.0A si předávají data, která následně zobrazí na displeji a pošlou po sériové lince do počítače. V počítači se zobrazují prostřednictvím hyperterminálu. Za účelem komunikace s počítačem používám hyperterminál "HERCULES", který je volně dostupný a je produktem společnosti HW-group (www.HW-group.com). Pomocí sériové komunikace se dá celý přípravek ovládat, aniž by se musely povely zadávat z klávesnice. Pro manuální "místní" ovládání slouží čtyřtlačítková klávesnice s tlačítky: "UP", "DOWN", "ESC" a "OK". Pomocí těchto tlačítek je ovládání intuitivní a slouží pro rolování v menu, popř. pro vnoření či odchodu z podmenu.

4.1.1 Ovládání přípravku pomocí klávesnice

Po zapnutí napájení proběhne inicializace programu, po které přípravek pošle na sériovou linku informaci o zapnutí uzlu. Následně se na displeji objeví "uvítací" nápis:

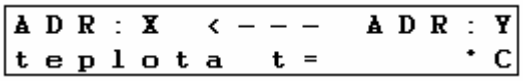
Po stisku libovolného tlačítka se program přesune do hlavního menu, jehož první položka nastavuje vlastní adresu uzlu. Implicitně má adresa přípravku hodnotu 0. Pro lepší orientaci v textu jsou displeje hlavního menu podbarveny modře a displeje podmenu (vnořeného menu) jsou nepodbarveny. Takto je to rozlišeno pouze v návodu .

Pokud hodláme adresu změnit, stiskneme tlačítko "OK" a na přípravku se objeví nápis:

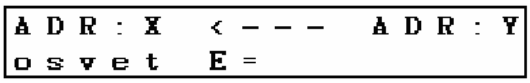


Nyní můžeme pomocí tlačítek "UP" a "DOWN" nastavit adresu vlastního uzlu v rozmezí hodnot 0 - F (decimálně: 0 -15). Volba se dá potvrdit tlačítkem "OK" a tím se program vynoří zpět do hlavního menu. Další položky v menu jsou:

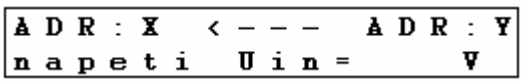
Při stisku tlačítka "OK" se program vnoří do módu zobrazování stavu, při stisknutí tlačítek "UP" a "DOWN" program roluje dále v hlavním menu. Pokud chceme zobrazovat stavy přípravku, stiskneme "OK" a na displeji se objeví nápis:



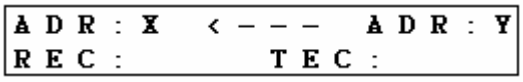
Horní řádek ukazuje, že je nastavena komunikace s uzlem adresy Y. Náš uzel má adresu X. Adresu uzlu, se kterým se má komunikovat, nastavíme v další položce hlavního menu. Implicitně je adresa komunikace opět nastavena na hodnotu 0. Pokud je adresa komunikace a adresa přípravku stejná, program zobrazuje vlastní hodnoty, aniž by komunikoval pomocí sběrnice CAN. Pouze pošle informaci po sériové lince. Pokud hodláme zobrazit teplotu, stiskneme "OK". Jinak se dá v podmenu pohybovat tlačítky "UP" a "DOWN". Další položky menu "ZOBRAZ STAV" jsou:



Program po stisku "OK" ukáže hodnotu osvětlení na přípravku na adrese Y.

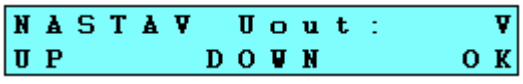


Program po stisku "OK" ukáže hodnotu vstupního napětí převodníku A/D na přípravku na adrese Y.

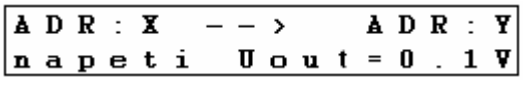


Program po stisku "OK" ukáže hodnotu registrů REC a TEC chybovosti sběrnice CAN přípravku na adrese Y.

Další položka hlavního menu umožňuje nastavit hodnotu výstupního napětí na PWM generátoru. Výstupní napětí je kalibrováno ve voltech a hodnota se pohybuje v rozmezí 0,1 - 4,9 V.

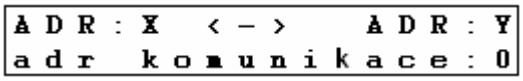


Po stisknutí tlačítka "OK" se program vnoří do menu a implicitně nastaví hodnotu výstupního napětí U_{out} = 0,1 V. Tato hodnota bude nastavena na adrese Y, jak charakterizuje směr šipek na horním řádku. Změnu provádíme pomocí tlačítek "UP" a "DOWN"



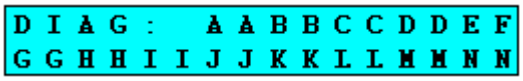
Následující položka hlavního menu nastavuje adresu komunikace. Pokud nehodláme komunikovat s ostatními uzly, ale pouze sledovat stav vlastních periferií přípravku, musíme nastavit adresu komunikace stejnou jako vlastní adresu.

Pokud chceme změnit adresu (implicitně 0), po stisku tlačítka "OK" nám ji program umožní měnit v rozmezí 0 - F pomocí tlačítek "UP" a "DOWN".



Hodnota adresy se potvrdí stiskem klávesy "OK".

Poslední položkou v hlavním menu je informační okno "DIAG" - diagnóza.



Tuto položku jsem vytvořil za účelem ladění aplikace. Při běžném provozu programu bude sloužit pro zobrazení jednotlivých bytů přenosu. Položka "DIAG" (diagnóza) zobrazuje automaticky obsah registrů programu. Pomocí hodnot komunikačních registrů se dá lépe zjistit průběh komunikace CAN a stav některých registrů. Seznam zobrazovaných registrů je v tabulce 6.

Tab. 6 *popis a význam zobrazených registrů*

Pozice na displeji	Název registru	Popis registru
AA	rxuart	první přijatý byte od UART
BB	rxuart+1	druhý přijatý byte od UART
CC	errtwi	identifikace chyby od sběrnice TWI
DD	errorcan	identifikace chyby při provozu CAN
E	ownadr	adresa uzlu
F	adrkom	adresa uzlu s nímž komunikujeme
GG	rxcana1	registr identifikátoru zprávy: v horním niblu je příkaz, v dolním niblu je adresa komunikace
HH	rxcand	registr identifikátoru zprávy: v horním niblu je RTR-0-0, v dolním niblu je počet datových bytů
II	rxcand1	první byte přenášených dat
JJ	rxcrc	MSB registr dat CRC kódu
KK	rxcrc1	LSB registr dat CRC kódu
LL	canreg	příznakový registr provozu CAN
MM	canstat	stavový registr provozu CAN
NN	canst	registr záznamu přenosu jednotlivých polí rámce

Hodnoty digitálních vstupů a výstupů se nenastavují, pouze se kopírují mezi jednotlivými přípravky. Podle nastavené adresy komunikace se zobrazují z daného uzlu digitální vstupy a zasílají se hodnoty našich vlastních vstupů na uzel, s nímž komunikejeme. Znamená to, že

každá aktivace vstupu vyvolá komunikaci na sběrnici CAN, na sériovou linku PC se hodnoty vstupů neposílají.

4.1.2 Ovládání přípravku pomocí počítače

Přípravek lze plnohodnotně ovládat prostřednictvím hyperterminálu. Příkazy, jimiž se přípravek ovládá, jsou zobrazeny v Tab.7. Prostřednictvím sériové komunikace nás přípravek také informuje o provozu a stavech vlastního uzlu, nebo uzlu, s nímž se komunikuje. Seznam jednotlivých povelů je v Tab.8.

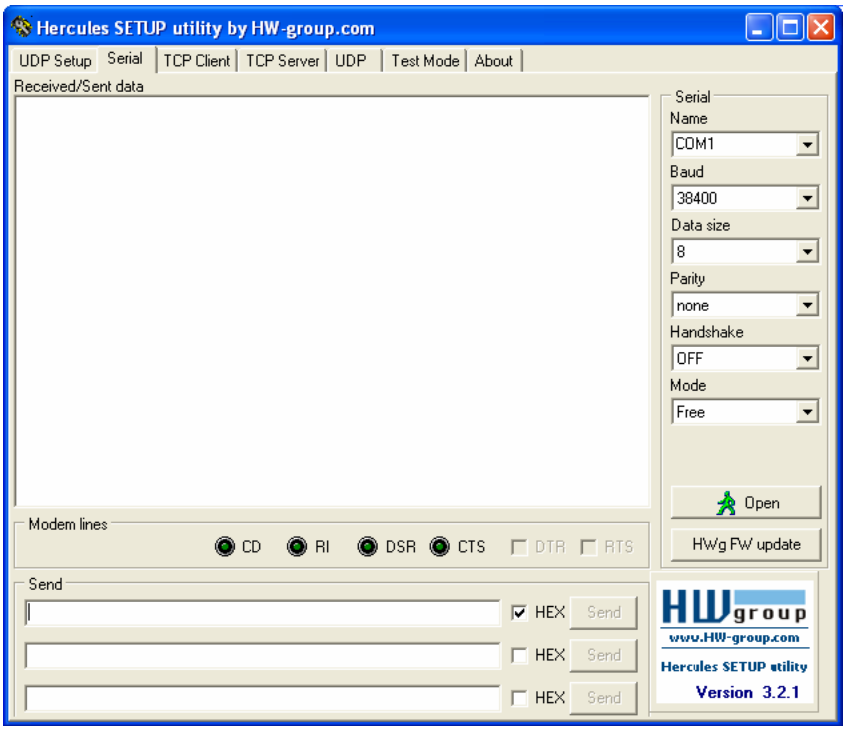
Jedná se o jedno a dvoubytové povely. Každý povel začíná řídícím bytem, za kterým podle potřeby následuje byte s datovou hodnotou. Horní nibl řídícího bytu určuje druh operace, dolní nibl určuje adresu které je povel určen, popř. adresu uzlu, ze kterého informace vychází.

Tab. 7 *povely UART z hyperterminálu do přípravku*

1. Byte	2. Byte	Počet Bytů	Popis
0x			-
1x	-	1	zjisti teplotu na adrese x
2x	-	1	zjisti osvětlení na adrese x
3x	-	1	zjisti vstupní napětí U_{in} na adrese x
4x	-	1	zjisti MSB nibl REC/TEC na adrese x
5x	-	1	zjisti výstupní napětí U_{out} na adrese x
6x	-	1	zjisti stav vstupů na adrese x
7x	-	1	zjisti stav výstupů na adrese x
8x			
9x			
Ax	05 ÷ FE	2	nastav U_{out} na adrese x
Bx	00 ÷ 0F	2	nastav výstupy na adrese x
Cy	-		nastav adresu komunikace y
Dx	-		nastav adresu uzlu (přípravku)
E0	-	1	přenos vyvolávající Error Frame
Fx	00 ÷ FF	0 ÷ 8	přenos x Bytů na adresu uloženou v registru adrkom

Tab. 8 *povely UART z přípravku do hyperterminálu*

1. Byte	2. Byte	Počet Bytů	Popis
0x	00 ÷ FF	2	příjem jednoho Bytu dat z adresy x
1x	tep	2	stav teploty na adrese x
2x	sv	2	stav osvětlení na adrese x
3x	uin	2	stav vstupního napětí U_{in} na adrese x
4x	high(rec); low(tec)	2	stav MSB nibl REC/TEC na adrese x
5x	uout	2	stav výstupního napětí U_{out} na adrese x
6x	digin	2	stav vstupů na adrese x
7x	digout	2	stav výstupů na adrese x
8x			
9x			
Ax			
Bx			
Cy	-	1	adresa komunikace y
Dx	-	1	vlastní adresa x
Ex	-	1	první aktivace uzlu x
Fx	-	1	zapnutí uzlu x



Obr. 30 *dialogové okno hyperterminálu HERKULES*

Z obrázku 30, na kterém je ukázka dialogového okna hyperterminálu, vyplývají komunikační parametry sériové linky. Ty jsou pevně nastaveny v mikrokontroléru. Pro větší přehlednost jsou sestaveny v tabulce 9.

Tab. 9 *parametry komunikace UART*

Parametry UART	Hodnota
přenosová rychlost	38 400 Baudů
počet bitů	8
zabezpečení přenosu	bez parity
délka STOP bitu	2bity

5 Závěr

Bakalářská práce na téma popis a využití sběrnice CAN byla pro mě v době výběru témat zadání velkou neznámou. Při prvním zjišťování a vnořování se do problematiky sběrnice CAN se moje obavy dokonce prohloubily. Na jedné straně nedostatek literatury popisující podrobně tuto problematiku, na straně druhé to byl malý výběr součástek, které umožňují komunikaci s touto sběrnicí. Nyní, v době dokončení bakalářské práce, mám dobrý pocit z výběru tohoto tématu. Jak praxe vždycky ukazuje; čím těžší úkol se splní a čím těžší překážky se překonají, tím toho více utkví v paměti a celkový dojem ze splněného úkolu je o to uspokojivější.

Úkolem bakalářské (a předtím i semestrální) práce bylo pochopení a prostudování sběrnice CAN, návrh laboratorního přípravku a vytvoření programu obsluhující provoz sběrnice. Z těchto úkolů bylo nejtěžším vytvořit program obsluhy. V okamžiku, kdy jsem si stanovil cíl, že provoz sběrnice nebude obsluhovat speciální obvod, ani modul v procesoru, byl tento úkol ještě obtížnějším.

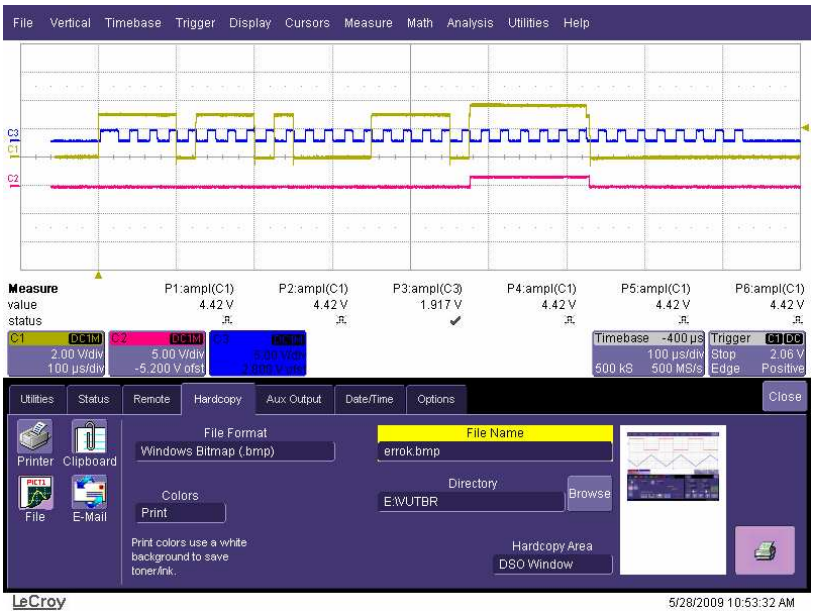
Na hotový program (a tím i na celý přípravek) nahlížím i kritickým pohledem. Pro účely laboratorního přípravku má zbytečně moc funkcí a periferií, které mohou být spíše matoucí. Z laboratorních měření vím, že čím složitější úkol a složitější ovládání přípravků, tím méně zbývá času na pochopení podstaty samotného měření. Všechny podpůrné obvody a zobrazování rozličných veličin (osvětlení desky, teplota na desce, vstupní a výstupní napětí, sepnutí vstupů a výstupů) jsou zřejmě navíc, ale bylo to přidáno k programu z důvodu objevitelské radosti programátora nad možností tolika funkcí jednoho procesoru.

Nejzajímavější částí celé práce bylo měření v laboratoři. Komunikaci mezi dvěma přípravky jsem měřil laboratorním přístrojem LeCroy. Přístroj má možnost zobrazení průběhu i se záznamem čtyř vstupů. Pro měření jsem využil tři sondy. Zobrazení průběhu sběrnice mezi budiči (C1), zobrazení časových sekvencí skenování sběrnice (C3) a zobrazení výstupu na přijímači (C2) - ukazuje obrázek Obr. 31.

Obr. 31 *ukázka průběhu CAN při posílání jednoho datového bytu*

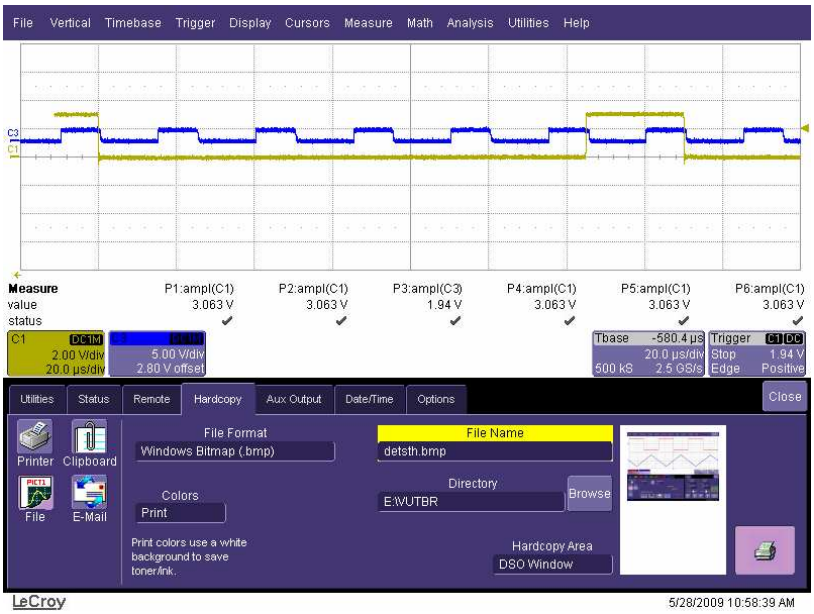
Z předcházejícího snímku je patrné, že před zahájením přenosu pole EOF potvrdí přijímač validitu všech dosavadně přijatých dat bitem ACK - viz červený průběh.

Na dalším obrázku (Obr. 32) je zobrazen průběh vysílání dat, ve kterých je na pozici bitu R1 záměrně vysílán recesivní stav, který přijímač neakceptuje a zahájí vysílání chybového rámce - viz červený průběh.



Obr. 32 *ukázka průběhu CAN při vysílání chybového rámce*

Na dalších ukázkách měření (Obr. 33 a 34) je znázorněn detail vkládacího bitu (Stuff Bit), který slouží k synchronizaci provozu, pokud se vysílá příliš dlouhá sekvence stejných bitů. Provozovaný bit sběrnice je ohraničen sestupnou hranou modrého průběhu.



Obr. 33 *detail vkládacího bitu H*

Obr. 34 *detail vkládacího bitu L*

Ze zobrazených průběhů a z nesčetných testů vyplývá, že přípravky komunikují mezi sebou řádně, bez náznaků nestandardních průběhů, a že vysílaná data jsou na straně přijímače zobrazena.

Vzhledem k tomu, že problematika CAN je dosti obsáhlá, nejsou součástí této práce bezezbytku všechny možné varianty přenosů a reakce na všechny možné stavy, které se na sběrnici mohou vyskytovat. Obsluha těchto ostatních situací, nebo zrychlení sběrnice na maximální možnou míru, by mohla být náplní další práce. Celý program je totiž stavěn tak, aby provoz sběrnice nezahltil obslužné procedury a zbylo dost prostoru pro výpočet a zpracování dat i při maximálně hustém vysílání dat na sběrnici. Teoretická průměrná spotřeba instrukčního času na provoz sběrnice CAN je 29 %. Pokud je stav "Bus Free", tak se nespotřebuje žádný instrukční čas. Náplní další práce by mohlo být vytvoření univerzálního modulu CAN, který by využíval přesně stanovený datový prostor. K tomuto modulu "přijímač/vysílač CAN" by se přistupovalo podle přesně stanovené instruktáže. Takto vytvořený model by mohl být součástí programu procesoru, který by používal časově náročnější úkony a pro svůj provoz by více využíval sběrnici CAN, kterou by mohl mnohem více zatěžovat.

Obslužný program, který je součástí této bakalářské práce, k tomu směřuje.

Seznam použité literatury

[1] Paret, Dominique. Multiplexed Networks for Embedded Systems: CAN, LIN, FlexRay, Safe - by - Wire.... John Wiley & Sons, Ltd. Bognor Regis: 2007. 434 s. ISBN: 978-0-470-03416-3

[2] Matoušek, David. *Práce s mikrokontroléry Atmel ATmega16*. 1. vydání, BEN technická literatura. Praha: 2006. 320s. ISBN 80-7300-174-8

[3] Váňa, Vladimír. *Mikrokontroléry ATMEL AVR-popis procesoru a instrukční soubor.* 1. vydání, BEN-technická literatura. Praha: 2003. 336 s. ISBN 80-7300-083-0

[4] Hrbáček, Jiří. *Komunikace mikrokontroléru s okolím I.* 2. dotisk 1. vydání, BEN - technická literatura. Praha: 1999. 160 s. ISBN 80-86056-42-2

[5] Hrbáček, Jiří. Komunikace mikrokontroléru s okolím II. 1. vydání, BEN-technická literatura. Praha: 2000. 151 s. ISBN 80-86056-73-2

[6] Matoušek, David. Číslicová technika - základy konstruktérské praxe. 1. vydání, BEN - technická literatura. Praha: 2001. 208s. ISBN 80-7300-025-3

[7] PK design. MB-ATmega16/32 v 3.0 Základová deska modulárního vývojového systému MVS [online] 2007 [cit. 13. prosince 2008] Dostupné na WWW: http://www.pk-design.net/Datasheets/Zakladova_deska_ATmega16L_v13.pdf

[8] Novák, Jiří; Kocourek, Petr. Fieldbus. Controller Area Network (CAN) [online] 1998 [cit. 12. listop. 2008] Dostupné na WWW: http://fieldbus.feld.cvut.cz/can/index.html

[9] Taraba, Radek. Aplikování sběrnice CAN. [online] 2004 [cit. 12. listopadu 2008] Dostupné na WWW: http://hw.cz/Rozhrani/ART1173-Aplikovani-sbernice-CAN.html

[10] Závidčák, Miroslav. CAN-popis struktury. [online] 2004 [cit. 12. listopadu 2008] Dostupné na WWW: http://hw.cz/Rozhrani/ART1111-CAN---popis-struktury.html

[11] Atmel Corporation. 8-bit Microcontroller with 16K Bytes ATmega16. [online] 2002 [cit. 12. listopadu 2008] Dostupné na WWW: http://www.datasheetcatalog.org/datasheet/atmel/2466S.pdf

[12] Philips Semiconductors. PCA82C250 CAN controller interface. [online] 2000 [cit.12. listop. 2008] Dostupné na WWW: http://www.datasheetcatalog.org/datasheet/philips/PCA82C250.pdf

[13] ON Semiconductor. LM75 2-Wire Serial Temperature Sensor and Monitor. [online] 2000 [cit. 29. listopadu 2008] Dostupné na WWW: http://www.datasheetcatalog.org/datasheet2/4/099ply5pt9cpu64qsy21y2ekzowy.pdf

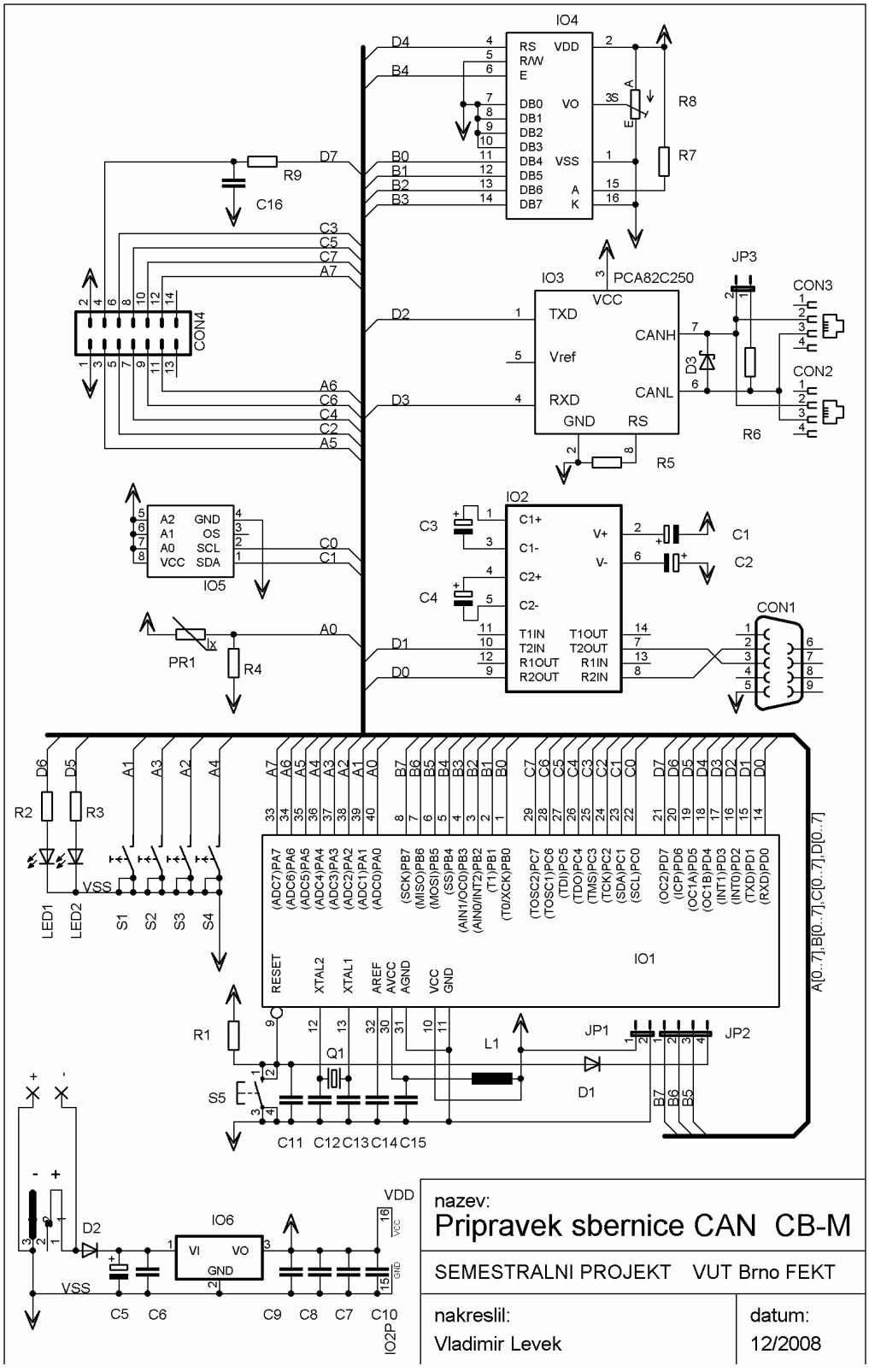
[14] Maxim Integrated Products: MAX232 +5V-Powered, Multichannel RS-232 Driv./Rec. [online] 2006 [cit. 12. listopadu 2008] Dostupné na WWW: http://www.datasheetcatalog.org/datasheet/texasinstruments/max232.pdf

[15] PerkinElmer Optoelectronics. VT80 Photoconductive Cells and Analog Optoisolators. [online] neurčeno [cit. 29. listopadu 2008] Dostupné na WWW: http://optoelectronics.perkinelmer.com/content/datasheets/dts_vt800.pdf

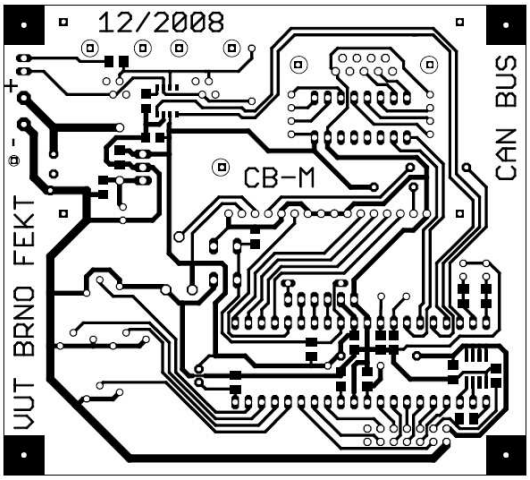
Seznam příloh

Konstrukce laboratorního přípravku

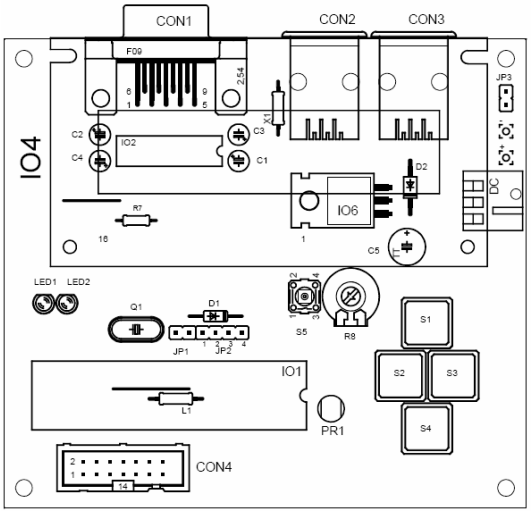
Schéma přípravku CB-M (hlavní DPS)



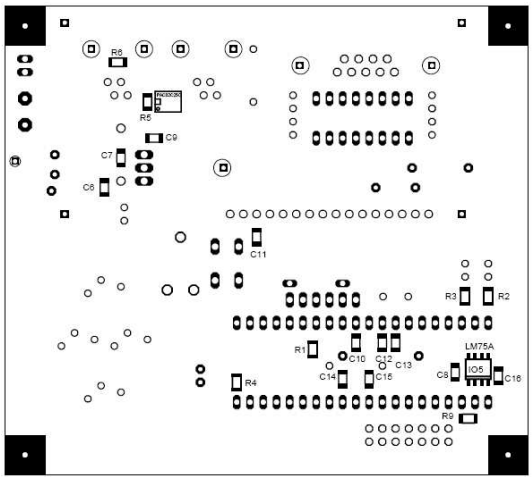
DPS přípravku CB-M ze strany plošného spoje (rozměr:100x90mm)



Osazování součástek přípravku CB-M - TOP (rozměr:100x90mm)



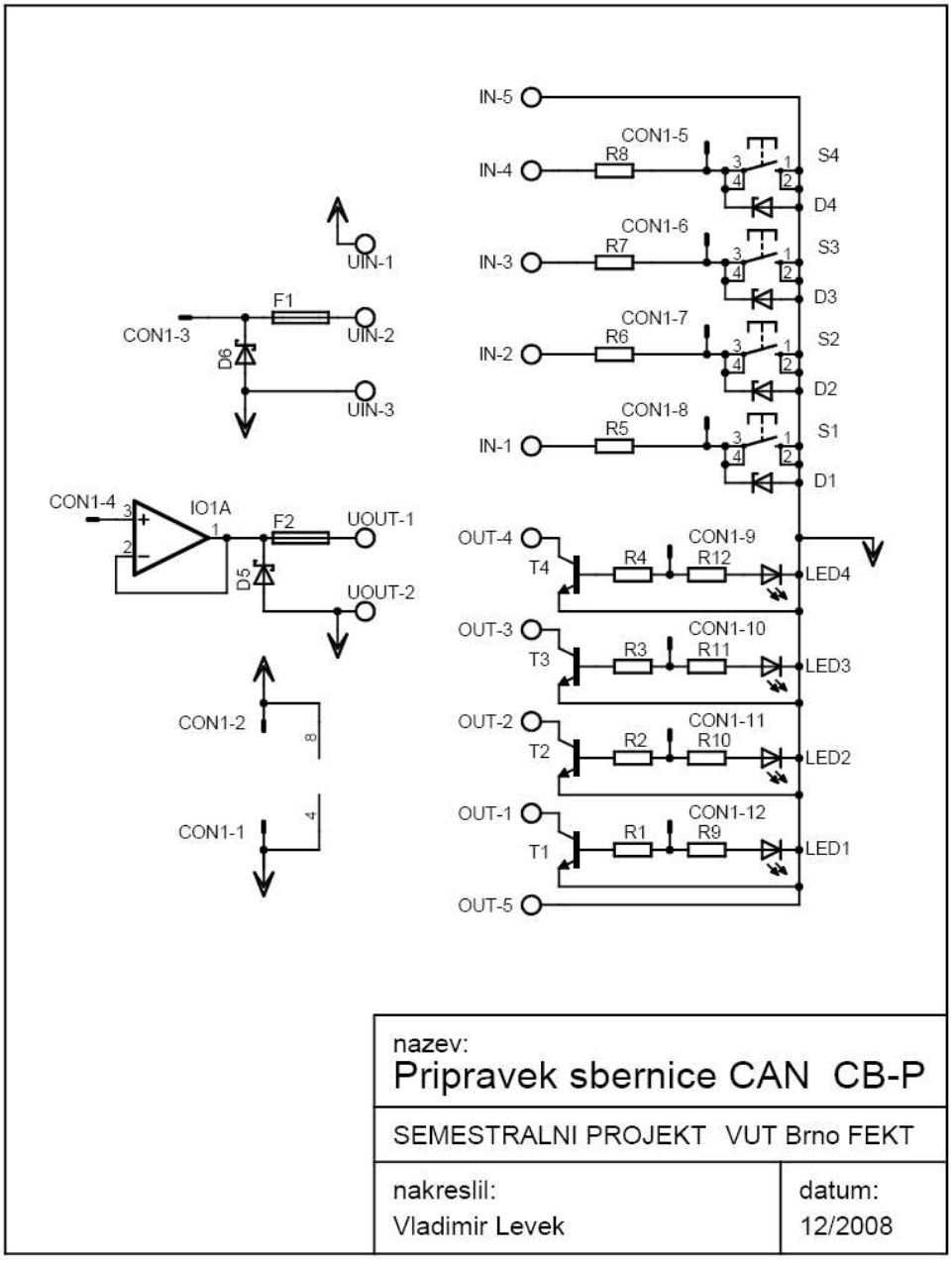
Osazování součástek přípravku CB-M - BOTTOM (rozměr:100x90mm)



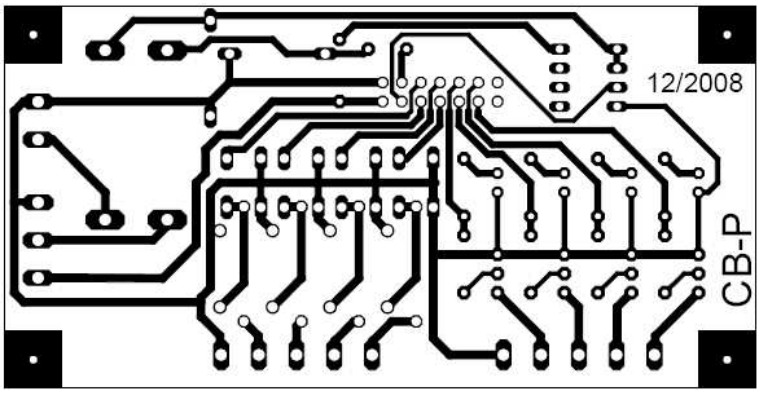
Seznam součástek přípravku CB-M

seznam součástek CB-M		
název	**hodnota**	**popis**
IO1	ATmega16	Mikroprocesor
IO2	MAX232	Převodník úrovně TTL na RS-232
IO3	PCA82C250	Budič sběrnice CAN
IO4	PC1602F	Displej 2x16 s řadičem HITACHI
IO5	LM75A	Snímač teploty I2C
IO6	7805T	Stabilizátor 5V TO220
CON1	CAN9 Z90	Konektor CANON pro port RS-232
CON2	RJ4	Konektor RJ pro sběrnici CAN
CON3	RJ4	Konektor RJ pro sběrnici CAN
CON4	MLW14	Konektor MLW pro připojení periferií
CON5	SCD 016A	Konektor DC pro připojení napájení
S1	PB1715	Mikrospínač pro DSP [UP]
S2	PB1716	Mikrospínač pro DSP [ESC]
S3	PB1717	Mikrospínač pro DSP [ENTER]
S4	PB1718	Mikrospínač pro DSP [DOWN]
S5	PB1720	Mikrospínač pro DSP [RESET]
LED1	LED 3mm	LED dioda vysílání na sběrnici CAN
LED2	LED 3mm	LED dioda skenování sběrnice CAN
JP1	JMP 2/2,54	Napájení ISP programování
JP2	JMP4/2,54	Konektor ISP
JP3	JMP2/2,54	Jumper pro připojení zakončovacího odporu
D1	1N4148	Dioda hradlo pro ISP programování
D2	1N4007	Dioda - ochrana proti přepólování zdroje
D3	BZW06-5V8-6	Transil 5,8V
Q1	Q16MHz	Krystal
L1	TLEC24-100k	Tlumivka pro odfiltrování vnitřního zdroje A/D převodníku
C1	CE 1μF/63 miniaturní	Kondenzátor pro MAX232 - násobič
C2	CE 1μF/63 miniaturní	Kondenzátor pro MAX232 - násobič
C3	CE 1μF/63 miniaturní	Kondenzátor pro MAX232 - násobič
C4	CE 1μF/63 miniaturní	Kondenzátor pro MAX232 - násobič
C5	100nF/63V keramický	Filtrování napájení
C6	100nF/63V keramický	Kondenzátor na vstupu stabilizátoru
C7	100nF/63V keramický	Kondenzátor na výstupu stabilizátoru
C8	100nF/63V keramický	Filtrování napájecích svorek pro měřič teploty
C9	100nF/63V keramický	Filtrování napájecích svorek pro budič CAN
C10	100nF/63V keramický	Filtrování napájecích svorek pro ATmega16
C11	100nF/63V keramický	Filtrování RESET kontaktu
C12	33pF/500V keramický	Kondenzátor pro krystalový vstup
C13	33pF/500V keramický	Kondenzátor pro krystalový vstup
C14	10μF/25V miniaturní	Kondenzátor pro vstup AREF
C15	10μF/25V miniaturní	Kondenzátor pro vstup AVCC
C16	39nF/50V	Kondenzátor pro integrační článek PWM převodníku D/A
R1	10k	Rezistor pro upínání RESETU
R2	1k	Rezistor pro LED1
R3	1k	Rezistor pro LED2
R4	10k	Napěťový dělič fotorezistoru
R5	10k	Rezistor pro budič sběrnice CAN
R6	120R	Zakončovací rezistor sběrnice CAN
R7	10R	Rezistor pro podsvětlení displeje
R8	PT10V 10k	Trimr pro řízení kontrastu displeje
R9	2k2	Rezistor pro integrační článek PWM převodníku D/A
PR1	VT080	Fotorezistor

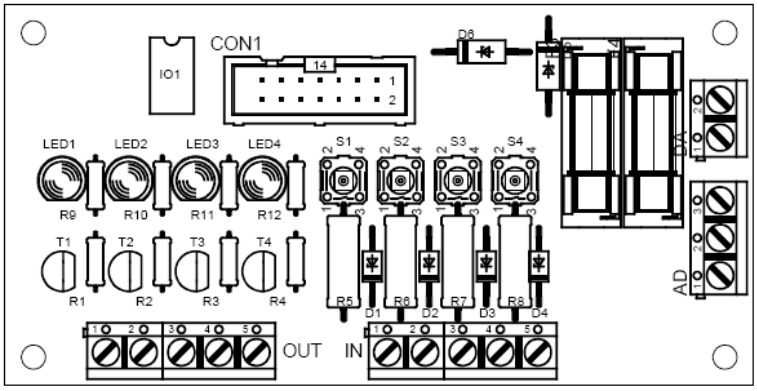
schéma přípravku CB-P (DPS-periferie)



DPS přípravku CB-P ze strany plošného spoje



Osazování součástek přípravku CB-P ze strany TOP



Seznam součástek přípravku CB-P

seznam součástek CB-P		
název	**hodnota**	**popis**
IO1	TL072	Operační zesilovač - sledovač
CON1	MLW14	Konektor MLW pro připojení periferií
IN	ARK103/2+ARK103/3	Konektor - svorkovnice 4x vstup + GND
OUT	ARK103/2+ARK103/3	Konektor - svorkovnice 4x výstup + GND
UIN	ARK103/3	Konektor - svorkovnice 3x vstup pro A/D převodník
UOUT	ARK103/2	Konektor - svorkovnice 2x výstup D/A převodníku
S1	PB1720	Mikrospínač pro DSP [IN1]
S2	PB1720	Mikrospínač pro DSP [IN2]
S3	PB1720	Mikrospínač pro DSP [IN3]
S4	PB1720	Mikrospínač pro DSP [IN4]
LED1	LED 5mm	LED dioda out1
LED2	LED 5mm	LED dioda out2
LED3	LED 5mm	LED dioda out3
LED4	LED 5mm	LED dioda out4
R1	6k8 0,25W 0207	Rezistor do báze tranzistoru typu O.C.
R2	6k8 0,25W 0207	Rezistor do báze tranzistoru typu O.C.
R3	6k8 0,25W 0207	Rezistor do báze tranzistoru typu O.C.
R4	6k8 0,25W 0207	Rezistor do báze tranzistoru typu O.C.
R5	200R 2W 0414	Rezistor pro ochranu vstupu
R6	200R 2W 0414	Rezistor pro ochranu vstupu
R7	200R 2W 0414	Rezistor pro ochranu vstupu
R8	200R 2W 0414	Rezistor pro ochranu vstupu
R9	1k 0,25W 0207	Rezistor pro LED
R10	1k 0,25W 0207	Rezistor pro LED
R11	1k 0,25W 0207	Rezistor pro LED
R12	1k 0,25W 0207	Rezistor pro LED
D1	BZX85V005.6 1,3W DO41	Zenerova dioda pro ochranu vstupu
D2	BZX85V005.6 1,3W DO41	Zenerova dioda pro ochranu vstupu
D3	BZX85V005.6 1,3W DO41	Zenerova dioda pro ochranu vstupu
D4	BZX85V005.6 1,3W DO41	Zenerova dioda pro ochranu vstupu
D5	P6KE6V8A 600W DO15	Transil pro ochranu výstupu
D6	P6KE6V8A 600W DO15	Transil pro ochranu vstupu
F1	KS21SW + F500mA	Tavná pojistka
F2	KS21SW + F500mA	Tavná pojistka

Návrh laboratorní úlohy

LABORATORNÍ CVIČENÍ **Ústav radioelektroniky** FEKT VUT BRNO	*Jméno a příjmení*	*Kód*
	Předmět	*Ročník*
Vyučující	*Měřeno dne*	*Odevzdáno dne*
Název úlohy **Testování protokolu CAN**		*Číslo úlohy*

1. Cíle laboratorní úlohy

Cílem laboratorní úlohy je osvojení a získání znalostí problematiky protokolu CAN, způsobu přenosu, řešení nestandardních stavů a kolizí na sběrnici. Dále je cílem této laboratorní úlohy získání zkušeností při měření přístrojem WaveSurfer Xs-A od firmy LeCroy.

2. Zadání

1. Seznamte se s laboratorním přípravkem CB-M.
2. Proměřte na osciloskopu přenos jednoho bytu dat, zjistěte bitovou rychlost a délku celého rámce zprávy.
3. Na změřeném vzorku najděte jednotlivá pole, najděte rovněž Stuff Bity a zdůvodněte důvod jejich použití. Zjistěte, jaký kód CRC se přenáší a naznačte způsob jeho výpočtu.
4. Vyvolejte chybový rámec, změřte jeho průběh a označte jej na změřeném vzorku.

3. Teoretické poznatky

Protokol sběrnice CAN rozlišuje čtyři typy komunikace:

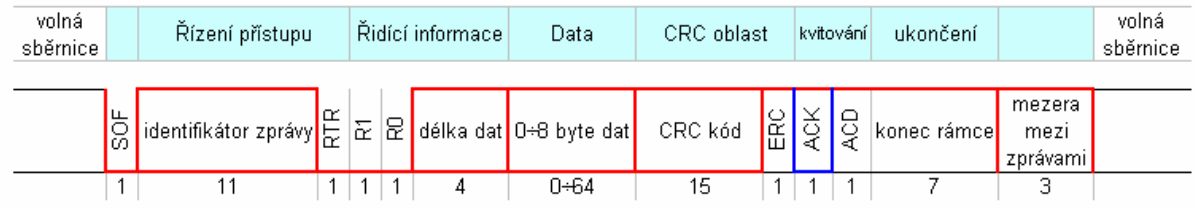
- Datový rámec *(Data Frame)*
- Žádost o rámec *(Remote Frame)*
- Chybový rámec *(Error Frame)*
- Rámec přetížení *(Overload Frame)*

V laboratorním měření budeme proměřovat pouze datový rámec a chybový rámec.

Datový rámec

Tento rámec má za úkol standardní vyslání dat na sběrnici. Pokud má uzel připravenu datovou zprávu, detekuje sběrnici, zda je volná, pokud ano, může zahájit vysílání dat.

Datová zpráva je zahájena Start bitem (SOF), kterým přejde sběrnice z recesivního stavu do stavu dominantního. Následuje blok 11-ti bitů řízení přístupu na sběrnici (Arbitration Field). Tyto bity přinášejí informaci o významu zprávy, prioritu zprávy a adresu vysílače - identifikátor zprávy. Blok řízení přístupu je zakončen bitem RTR (Remote Request). Ten udává, že se jedná o vysílání dat (dominantní stav). Bezprostředně po bloku řízení přístupu následuje pole Řídících informací. To je uvozeno dvojicí bitů R_1 a R_0 - ty jsou rezervovány pro další využití - nastavují se do dominantního stavu. Po nich následuje blok informující o počtu přenesených datových bytů. Maximální počet přenesených bytů je osm. Pokud se přenáší pouze jedna binární informace, může být počet datových bytů nulový a tato informace se může vložit do identifikátoru zprávy. Tím se uspoří časový prostor pro ostatní uzly. Po přenesení všech datových bytů je přeneseno 15 bitů CRC kódu. Po tomto bloku je přenesen jeden bit ERC - slouží pouze pro oddělení a vytvoření prodlevy pro zpracování CRC kódu. Bit ERC je v dominantním stavu. Po uplynutí předchozího bitu je vysílač v recesivním stavu a očekává od ostatních uzlů potvrzení správnosti přenosu (Acknowledge) - bit ACK. Ostatní uzly, pokud detekují správný příjem dat, nastaví sběrnici do dominantního stavu. Následuje bit ACD - oddělovač potvrzení, který je opět v recesivním stavu. Celý rámec je zakončen vysláním sedmi bitů v recesivním stavu. Tak je ukončena komunikace a novou komunikaci může zahájit kterýkoliv z uzlů až po uplynutí třech bitů. Tím se získá čas na zpracování získaných dat.

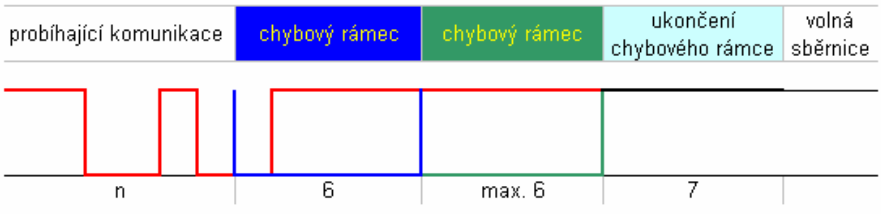


Obrázek 1 *schéma datové zprávy sběrnice CAN 2.0A*

Rámec chybové zprávy

Chybová zpráva slouží k signalizaci chybného přenosu sběrnice CAN. Pokud některý z přijímacích uzlů detekuje v přenosu chybu, zahájí vysílání chybového rámce. Chyby mohou být zejména tyto:

- chyba bitu
- chyba vkládání bitu (Stuff bit)
- chyba CRC
- chyba rámce



Obrázek 2 *schéma chybového rámce sběrnice CAN 2.0A*

Pokud je uzel detekující chybu v aktivním stavu, zahájí vysílání chybového rámce. Ten spočívá ve vyslání šesti bitů v dominantním stavu (viz modrý průběh). Tímto se

přenášená zpráva znehodnotí a ostatní aktivní uzly (viz zelený průběh) to detekují a rovněž zahájí vysílání chybového rámce. Délka takto zřetězené chybové zprávy může být šest až dvanáct bitů. Pokud je uzel v pasivním stavu, vysílá šest bitů v recesivním stavu. Tím zprávu neznehodnotí. Po odvysílání chybových rámců uzly přejdou do recesivního stavu a detekují sběrnici (i původní vysílač - ten byl jako vysílač "odstaven" a podílí se na sběrnici jako přijímač). Poslední uzel, který ukončí chybový rámec, uvede sběrnici do recesivního stavu. Tak se uzly zasynchronizují, odvysílají sedm bitů recesivního stavu, tím se ukončí chybový rámec a sběrnice je ve stavu "Bus-free".

4. Postup měření

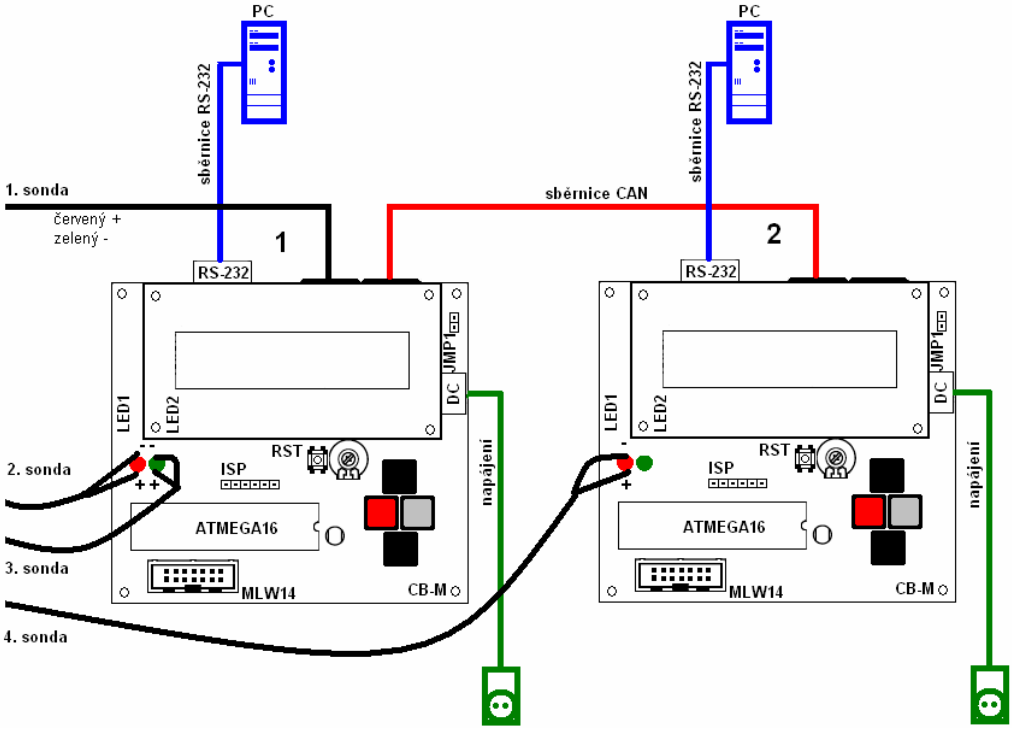
ad 1 Propojte mezi sebou dva laboratorní přípravky, propojte je pomocí sériových kabelů s porty RS-232, zapněte napájení a pomocí ovládání z klávesnice vyzkoušejte přenos. Na každém PC otevřete hyperterminál HERCULES, zadejte v záložce "SERIAL" tyto přenosové parametry:

přenosová rychlost	38 400 Baudů
počet bitů	8
zabezpečení přenosu	bez parity
délka STOP bitu	2 bity

a otevřete sériový port. Za editačním oknem SEND zaškrtněte formát "HEX", totéž nastavte v okně RECEIVED/SEND DATA (pravým tlačítkem myši).

Do editačního okna zapište sekvenci: "`f1 xy`", kde "xy" znamená libovolný byte v hexadecimálním vyjádření (00 - FF). Poté stiskem tlačítka SEND odešlete data do přípravku. Na druhém PC se zobrazí tatáž sekvence ve velkém okně hyperterminálu.

ad 2 Zapněte osciloskop a propojte měřící sondy podle obrázku 3.



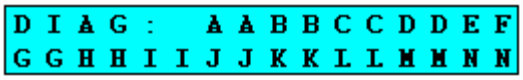
Obrázek 3 *zapojení pracoviště CAN*

Na osciloskopu nastavte potřebné parametry, nastavení vstupů, trigger, časový offset. Na pracovišti č. 1 odešlete z hyperterminálu libovolnou jednobytovou zprávu s prefixem `f1`. Např. pro odeslání dat 0x46 zadejte sekvenci: `f1 46`. Zaznamenanou zprávu uložte a proveďte časovou analýzu. Zjistěte, jakou má rámec délku a jakou bitovou rychlostí jsou data zasílána.

ad 3 První sonda měří úroveň na sběrnici, která je inverzní logické hodnotě (log. 0 má vysokou úroveň amplitudy). Druhá sonda proměřuje vysílání na sběrnici v úrovních 0 - 5 V. Třetí sonda, sloužící pouze pro optické rozdělení průběhu na jednotlivé bity, ukazuje průběh, kdy vysílač skenuje sběrnici (náběžná hrana) a začátek vysílaného bitu (sestupná hrana). Průběh naměřený na sondě č. 4 ukazuje vysílání na sběrnici od přijímače. Zde se projeví bit ACK, kterým přijímač potvrdí platnost přijatých dat. Proveďte označení jednotlivých polí celé zprávy, tedy:

- bit SOF (Start of Frame)
- identifikátor zprávy - na pozici bitu 1 - 4 je uložena adresa uzlu a na pozici bitu 5 - 8 je uložena hodnota 0xF
- bity RTR (log.0), R1,R0 (oba log.0)
- počet datových bytů - zde bude hodnota 0x1
- datové pole - hodnota bude dle zadání na hyperterminálu (druhý odeslaný byt)
- CRC kód - 15 LSB bitů (bit MSB se nepřenáší)
- bity ERC (log. 1), ACK (v případě platného přenosu log. 0 - viz sonda č.4), ACD (log.1)
- pole EOF - upnutí sběrnice na hodnotu log. 1
- všechny vkládací StuffBity - následují vždy po pěti bitech stejné hodnoty a mají opačnou úroveň.

Výpočet CRC kódu proveďte pomocí kontrolního polynomu: 11000101 10011001 (0xC599). Celý binární řetězec začínající SOF (log. 0) a končící posledním datovým bitem LSB zapište za sebe, vznikne tak polynom zprávy. Pokud zpráva obsahuje Stuff Bity, které se nepodílejí na přenosu, nezařazujte je do řetězce polynomu zprávy. Kontrolní polynom zapište pod polynom zprávy a zarovnejte jej zleva. Pokud na pozicích MSB obou polynomů jsou stejné bity (oba log. 1), proveďte operaci XOR nad oběma polynomy a za výsledek dopište zbývající bity polynomu zprávy. Pokud se bity MSB neshodují, neprovádí se žádná operace. Následně potom posuňte kontrolní polynom o jeden bit vpravo a pokud jsou na pozici MSB úrovně log. 1, opět proveďte operaci XOR. Takto posouvejte až do okamžiku, kdy se oba polynomy zarovnají vpravo - neboli provedl se kontrolní součet s posledním přeneseným bitem. Za účelem kontroly se přenáší pouze 15 LSB bitů výsledku CRC. Stiskem černých tlačítek na přípravku se dá narolovat na zobrazení:



kde na pozici GG se zobrazí příkaz zadaný z hyperterminálu (horní G) a adresa uzlu komunikace (dolní G). Na pozici HH se zobrazí počet přenášených bytů (dolní H), na pozici II jsou zobrazena data a na pozici JJ a KK je zobrazen CRC kód včetně bitu ERC.

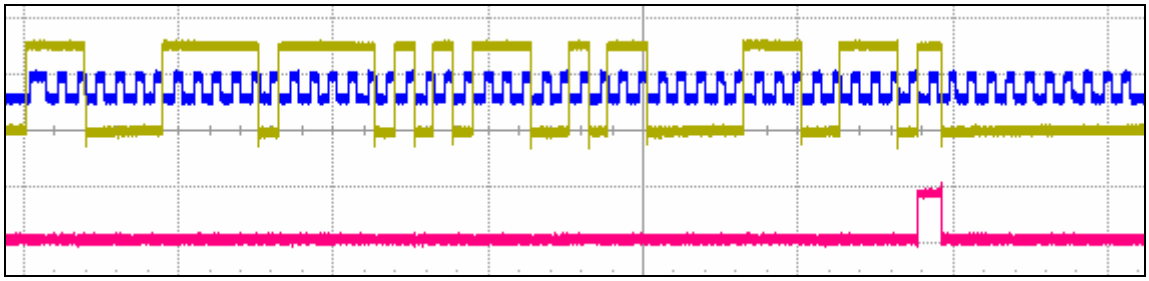
ad 4 Na osciloskopu zapněte čekání na start záznamu. Do hyperterminálu zapište hodnotu E0 a odešlete. Přípravek na tento příkaz reaguje tak, že záměrně odešle zprávu, ve které nejsou všechny bity korektní. Ze záznamu určete, o jakou chybu se jedná a jakým způsobem na ni vysílač reaguje. Celý sled přenesených bitů opět zakreslete a popište jednotlivá pole a významné bity.

5. Vypracování

ad 2

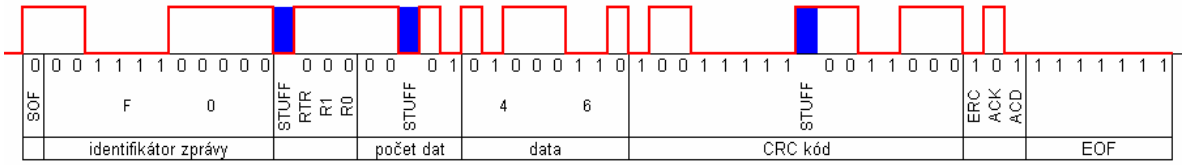
Přenesená data:	0x46
Přenosová rychlost sběrnice:	40 kbit/s
Celková doba přenosu:	1,375 ms

ad 3



Průběh přenosu datové zprávy

žlutý průběh:	úroveň CAN na sběrnici (vyšší amplituda = dominantní stav)
modrý průběh:	náběžná hrana = skenování sběrnice, sestupná hrana = začátek bitu
červený průběh:	vysílání od přijímače



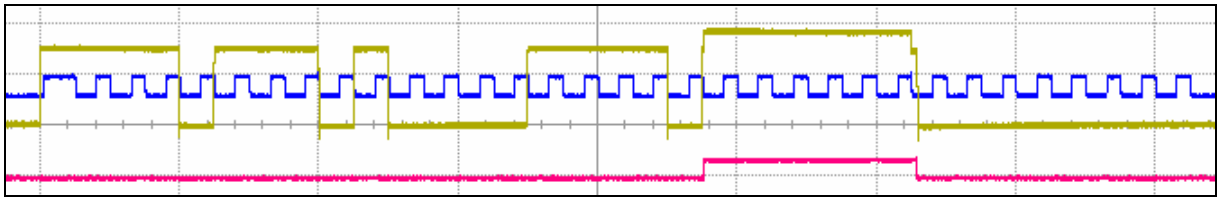
Popis průběhu přenosu datové zprávy

V menu "diagnóza" jsem zjistil, že příkaz má hodnotu F, adresa uzlu má hodnotu 0, posílal se jeden byte o obsahu 0x46, CRC kód včetně bitu ERC má hodnotu 0x9F31.

Výpočet kódu CRC

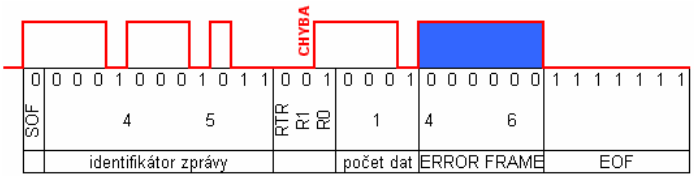
	příkaz				adresa					počet				data							
0	F				0				0				1	4				6			
0 0 0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0
	1	1	0	0	0	1	0	1	1	0	0	1	1	0	0	1					
	0	0	1	1	0	1	0	1	1	0	0	1	1	0	0	0	0	1	0	0	0	1	1	0
			1	1	0	0	0	1	0	1	1	0	0	1	1	0	0	1						
			0	0	0	1	0	0	1	1	1	1	1	1	1	0	0	0	0	0	1	1	0
						1	1	0	0	0	1	0	1	1	0	0	1	1	0	0	1			
						0	1	0	1	1	0	1	0	0	1	0	1	1	0	0	1	1	1	0
								1	1	0	0	0	1	0	1	1	0	0	1	1	0	0	1	
								0	1	1	1	0	0	0	1	0	1	0	0	1	0	1	0	1	0
									1	1	0	0	0	1	0	1	1	0	0	1	1	0	0	1	
									0	0	1	0	0	1	1	1	1	0	0	1	1	0	0	0	0
operace XOR se neprovádí									1	1	0	0	0	1	0	1	1	0	0	1	1	0	0	1	
výsledný CRC									0	1	0	0	1	1	1	1	1	0	0	1	1	0	0	0	

ad 4



Průběh přenosu zprávy obsahující chybový rámec

žlutý průběh:	úroveň CAN na sběrnici (vyšší amplituda = dominantní stav)
modrý průběh:	náběžná hrana = skenování sběrnice, sestupná hrana = začátek bitu
červený průběh:	vysílání od přijímače, který po identifikaci chyby odvysílal chybový rámec



Popis průběhu zprávy z chybovým rámcem

Přijímač identifikoval na pozici bitu R1 recesivní stav a zahájil vysílání chybového rámce. Začátek vysílání je posunut z důvodu, že ke zjištění chyby došlo až po přijetí celého bytu. Po odvysílání chybového rámce přípravek zobrazil na pozici "diagnóza" tyto údaje: příkaz má hodnotu 4, adresa uzlu má hodnotu 5, posílal se jeden byte, jehož obsah se nepřenesl, protože vysílač znehodnotil přenos vysíláním chybového rámce.

Použité laboratorní přístroje:

Měřící přístroj **CB-M**
Laboratorní přípravek **WaveSurfer Xs-A**
Osobní počítač s instalovaným hyperterminálem

6. Závěr

V tomto laboratorním cvičení jsem měřil přenosy pomocí sběrnice CAN. Nejprve jsem zjistil přenosovou rychlost sběrnice 40 kbit/s. V dalším úkole jsem poslal data na přípravek pomocí hyperterminálu. Přípravek data přeposlal protokolem CAN na další uzel, který je zobrazil. Naměřený průběh jsem zaznamenal a provedl jsem analýzu průběhu. Označil jsem jednotlivé pole datového rámce. Rovněž jsem v průběhu identifikoval vkládací bity. V poslední fázi tohoto úkolu jsem provedl výpočet CRC kódu, který se shodoval s naměřeným a zobrazeným stavem. V dalším úkole jsem si zobrazil přenos dat, který byl záměrně upraven tak, aby na příjem reagoval přijímač vysláním chybového rámce. Ten jsem si zobrazil a zjistil jsem, že chyba přenosu vznikla na pozici bitu R1, který měl inverzní úroveň.