České vysoké učení technické v Praze
Fakulta elektrotechnická

DIPLOMOVÁ PRÁCE

Systém měření klidových proudů ve vozidlech

Praha, 2010 Autor: Zdeněk Hovorka

Prohlášení

Prohlašuji, že jsem svou diplomovou (bakalářskou) práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

V Praze dne _____ _____

podpis

Poděkování

Děkuji především vedoucímu diplomové práce panu Ing. Janu Fischerovi za vedení této práce, za pomoc při problémech a za volnost při tvorbě práce. Dale bych rád poděkoval panu Ing. Jaromíru Kreclovi a celému diagnostickému oddělení ve firmě Škoda auto a.s za podporu a poradenství při tvorbě diplomové práce.

Abstrakt

Cílem diplomové práce je sofwarové vybavení pro ovládání a vyhodnocování dat ze systému na měření klidových proudů v palubní síti automobilu. Část práce se zabývá problematikou měření klidových proudů a dostupnými metodami měření. Dále na základě vybrané metody měření vytvořit systém umožňující měření elektrického proudu a zpětné analýzy naměřených dat. V práci je nastíněna základní práce s potřebnými softwary jako je např. CANoe, CANdb++ nebo Visual Studio 2008 a programovacími jazyky CAPL a C#. Programovací jazyk CAPL je využit pro měření veličin v reálném čase a nastavování systému, který dané veličiny měří. Programovací jazyk C# je využit k vývoji softwaru pro zpětnou analýzu ve formě grafů.

Abstract

The goal of diploma thesis is software equipment for control and data evaluation from system of measurement a quiescent current inside of on-board system an automobile. Part of diploma thesis handle to problems with measurement of quiescent current and accesible methods of measurement. Further on the basis choice method of measurement construct a system enabling measurement of electrical current and backward analyses of measured data. In thesis is foreshadowed basic work with necessary softwares such as e.g. CANoe, CANdb++ or Visual Studio 2008 and programming languages CAPL and C#. Programming laguage CAPL is used for measurement quantities in real-time and pre-setting the system, which this quantities measures. Programming language C# is used to development a software for backward analyse in the form of graphs.

vložit originální zadáníííííííííííííííííí !!!!!

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

Od vzniku prvního automobilu uplynulo více než sto let a za tuto dobu jeho vývoj prošel několika zásadními etapami nejen z hlediska konstrukčních materiálů a mechanizace, ale také z hlediska elektrotechniky a elektroniky. S příchodem 20. let 20. století se standardním vybavením automobilu stalo elektrické osvětlení a elektrická houkačka. Ve 30. letech 20. století přišla na svět elektronková rádia a zdálo se, že na elektrické soustavě automobilu není co zlepšit. Ovšem technický pokrok se nezastavil a s vynálezem tranzistoru se odkryl úplně nový svět elektroniky a mikroelektroniky. Nástup elektroniky v automobilu byl z počátku pozvolný, jelikož pracovní podmínky elektrických a elektronických součástí v motorových vozidlech jsou totiž jedny z nejtěžších. Kladen byl důraz na odolnost proti rozdílným teplotám okolí, proměnnou vlhkost ovzduší nebo působení agresivních plynných a kapalných látek. Rozhodujícím momentem elektronizace automobilu byl přechod na sběrnicovou komunikaci. Aktuálně nejrozšířenější je CAN (Controller Area Network) sběrnicová komunikace. V dnešní době elektronika v automobilu napomáhá nejen ke snižování spotřeby paliva, zvyšování výkonu motoru, ale také ke zvýšení bezpečnosti pasažéru (airbag, ABS,ESP, atd.) nebo ke zvýšení pohodlí a komfortu pasažérů prostřednictvím GPS navigace nebo multimediálních doplňků. Dnes hlavním požadavkem na automobil není jen usnadnění přesunu z bodu A do bodu B, ale zpříjemnění cesty, usnadnění řízení vozidla a hlavně důraz na bezpečnost cestujících.

Při používání elektroniky a vysoké integraci elektronických doplňků vzniká vysoká náročnost na elektrickou energii. Elektrická energie je v automobile omezena kapacitou akumulátoru a ta není nekonečná. Pro představu je průměrný příkon automobilu okolo 3kW a špičkově může dosahovat až 10kW. Z hlediska pohotovostního stavu automobilu není část o klidových proudech zanedbatelná, protože od kvalitního a spolehlivého vozu požadujeme, aby po delším stání na parkovišti jsme k vozu přišli, nastartovali a v klidu

odjeli a neřešili problémy s vybitou baterií. Samozřejmě, že výrobce automobilu zaručuje určitou časovou garanci, po kterou s vozem můžeme bez větších potíží odjet. Aby se tato garantovaná délka stání mohla udržovat na současné hranici nebo prodlužovat, je třeba se zaměřit na měření velikosti odebírané elektrické energie automobilu v klidovém stavu.

Proto je potřeba vyvinout systém, který dokáže klidové proudy měřit s dostačující přesností a následně je zpracovat, zejména při vývoji nových projektů, testování a odstraňování závad. Od systému tedy požadujeme, aby měřená data se dala přenášet a distribuovat v digitální formě a z hlediska krátkodobého časového horizontu se dala pozorovat v reálném čase. Z hlediska dlouhodobého časového horizontu požadujeme, aby naměřená data se ukládala, zpětně z těchto dat se dala vypočítat statistika a systém by měl dát uživateli představu o vývoji měřeného proudu v čase ve formě grafu. Všechny tyto požadavky systém bude splňovat společně s různými doplňujícími funkcemi.

Kapitola 2

Klidové proudy

Klidové proudy je termín, který vyjadřuje velikost odebíraného proudu elektrického obvodu nebo součástky, která nevykonává žádnou činnost(někdy je také nazýván jako pohotovostní proud nebo-li "stanby"proud). Tento proud je obzvláště důležitý při návrhu systému s bateriovým napájením. Většina systémů s bateriovým napájením je více času v klidovém režimu než v aktivním režimu a proto je potřeba na klidové proudy nahlížet s velkou důležitostí. Snaha všech návrhářů je koncipovat takové elektrické obvody a systémy, které mají nízkou hodnotu klidových proudů. Pro představu se hodnoty klidových proudů pohybují od jednotek nA až po stovky mA. Záleží na náročnosti součástek v elektrickém obvodu. Obsah této kapitoly jsem čerpal z [5] a [6].

2.1 Klidové proudy v automobilu

Klidový proud je proud, který je odebíraný spotřebiči v automobilu v pohotovostním režimu do odpojení všech spotřebičů od baterie. Příčina klidového proudu je jednotka, která vypadá zdánlivě nečinná, ale musí reagovat na vnější události. Radiokomunikace - dálková obsluha, výstražný systém proti krádeži, rádio nebo hodiny na palubní přístrojové desce. Nadále je stále více řídicích jednotek, které nejsou přímo zapínané přes zapalovací skříňku vozidla, ale jsou připojeny na baterii a mohou být zapínány nebo vypínány přes sběrnici CAN. Aby tento klidový proud svým odběrem příliš nezatěžoval baterii a tím nebyla ohrožena provozuschopnost vozidla, jsou každou automobilkou vydány směrnice a normy, které dohlíží nad maximálním odběrem klidové energie.

2.1.1 Maximální klidový proud automobilu

Maximální přípustný klidový proud je závislý na garantované délce provozuschopnosti vozidla a velikosti baterie ve vozidle. Pro novější modely automobilů je garantovaná provozuschopnost v délce trvání padesáti dní a pro starší modely je tato provozuschopnost garantována v délce čtyřiceti dní. Přehled maximálních klidových proudů pro danou velikost kapacity baterie je v tabulce 2.1.

Velikost baterie	40 dní	50 dní
36Ah	13,5 mA	10,5 mA
44Ah	16,5 mA	12,8 mA
60Ah	22,5 mA	17,5 mA
61Ah	22,9 mA	17,8 mA
68Ah	25,5 mA	19,8 mA
70Ah	26,3 mA	20,4 mA
72Ah	27,0 mA	21,0 mA
75Ah	28,1 mA	21,9 mA
80Ah	30,0 mA	23,3 mA
82Ah	30,8 mA	23,9 mA
85Ah	31,9 mA	24,8 mA
92Ah	34,5 mA	26,8 mA
95Ah	35,6 mA	27,7 mA

Tabulka 2.1: Tabulka maximálních klidových proudů

Při překročení dovoleného klidového proudu klesá provozuschopnost automobilu. V extrémním případě není nastartování vozidla již dále možné a automobil není provozuschopný.

2.1.2 Výpočet maximálního klidového proudu při zachování provozuschopnosti vozidla

Pro výpočet maximálního klidového proudu, který určuje tabulka maximálních klidových proudů 2.1 je vyžadováno několik kritérií:

- Stav "plná baterie"je definována pro stav nabití baterie na 80% své kapacity.

- Stav "start-způsobilá baterie"(stav baterie, kdy je ještě zaručen start vozidla) je definována na 40% své kapacity.

- Pro doběhový cyklus a samovolné vybíjení baterie se definuje 0,1% své kapacity.Tedy 5% na padesát dní.

Z toho plyne, že pro spotřebu klidovými proudy je k dispozici:

Použitelná kapacita(C_p) = 80% - 40% - 5% = 35% (pro 50 dní)
Použitelná kapacita(C_p) = 80% - 40% - 4% = 36% (pro 40 dní)

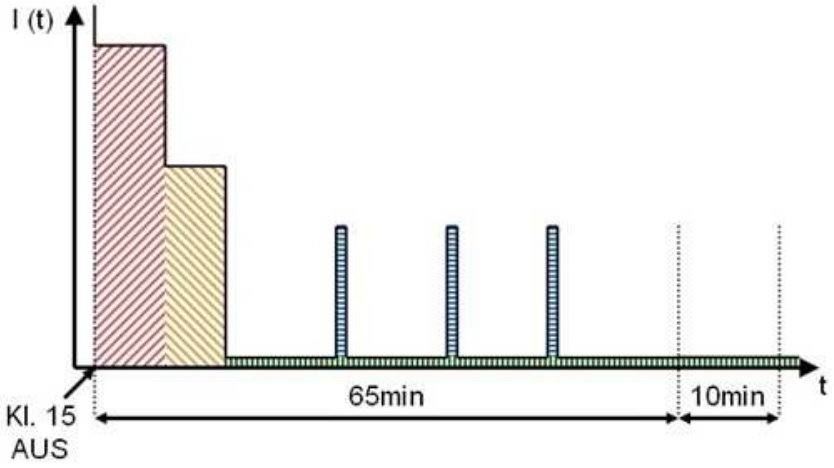
Maximální klidový proud automobilu je dán vzorcem:

$$I_{klid} = \frac{\frac{C_p}{100} \cdot C_{bat}}{D \cdot 24} \tag{2.1}$$

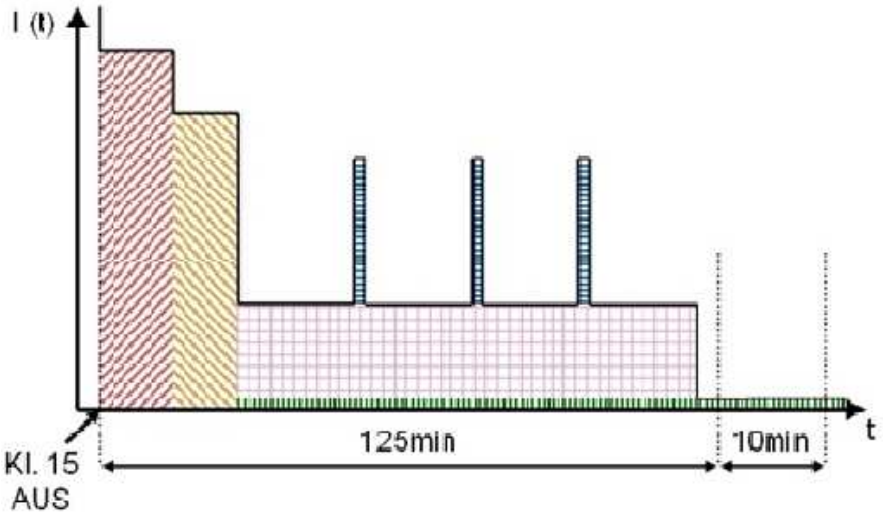
kde C_p je použitelná kapacita v procentech, C_{bat} je celková kapacita baterie v Ah, D je počet dní odstavení automobilu a násobí se 24x pro přepočet na hodiny.

2.1.3 Měření klidových proudů

Celkový klidový proud se měří v bezprostřední blízkosti pólu baterie na jednom ze dvou přívodních vodičů. Pro měření klidových proudů můžeme zvolit jakýkoliv multimetr, speciální bočník nebo bezdotykový měřící přístroj s dostatečnou přesností. U takového měření musíme dbát na to, abychom se vyvarovali krátkodobému odpojení přívodních vodičů baterie, protože by nastal restart všech jednotek automobilu a navýšení odebíraného proudu. Obzvláště důležité při měření klidových proudů je nutnost brát ohled na dobu trvání měření. Ustálení klidového proudu je závislé na úrovní výbavy automobilu a ustálení může probíhat v rozmezí několika desítek minut. Názorný rozdíl odběru proudu ve standardní výbavě nebo v rozšířené výbavě o multimediální prostředky je vidět na obrázcích 2.1 a 2.2.



Obrázek 2.1: Měření se standardní výbavou automobilu

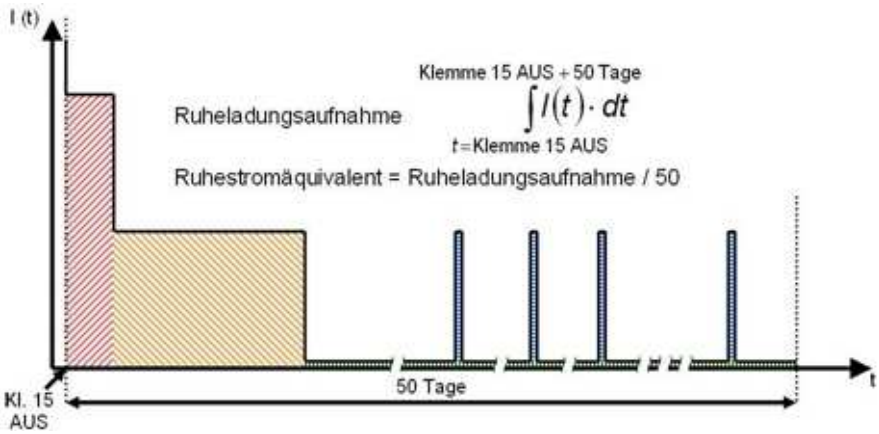


Obrázek 2.2: Měření s nadstandardní výbavou automobilu

S pomocí znalosti průměrných klidových proudů v automobilu lze cíleně vyhledávat závady na vozidle ve formě nadměrného odběru elektrické energie.

2.1.4 Návrh řídicích jednotek z hlediska klidových proudů

Pro klidový proud platí kritéria obsažená v interních směrnicích každého výrobce automobilů. Velikost elektrického náboje se vypočítá jako integrál z odběru proudu za určitý časový úsek. Tedy od vypnutí zapalování, uplynutí padesáti dní a znovu zapnutí zapalování. Klidový proud je tedy roven celkové velikosti elektrického náboje rozdělený do padesáti dní.



Obrázek 2.3: Výpočet klidového proudu

Snahou je docílit u řídicí jednotky klidový proud blížící se 0 mA. Podle interních směrnic je dáno, že jednotky po vypnutí zapalování musí maximálně odebírat 0,1mA. Tedy odebraná energie jedné jednotky za padesát dní nesmí přesáhnout 0,12Ah. Pokud daná jednotka nemůže splnit tyto kritéria musí jí být věnována větší pozornost a musí k ní být vytvořena podrobná dokumentace proč daná jednotka stanovený limit nemůže splnit.

2.1.5 Velikost klidových proudů v závislosti na okolních podmínkách

Jelikož nejde předepsat zákazníkovi, jak má automobil správně uvést do klidového stavu, vzniká mnoho rozdílných situací, které mají vliv na celkovou velikost klidového proudu. Následující možnosti mohou ovlivnit celkovou velikost klidového proudu:

- Zda je klíč zasunutý nebo vysunutý ze zapalovací skříňky
- Zda je vozidlo zamknuté nebo odemknuté

- Jestli je vozidlo zamknuté klíčkem nebo systémem KESSY (bezkontaktní odemykání/zamykání)
- Zda jsou dveře řidiče, spolujezdce a zadní dveře správně zavřené
- Jestli je kufr (výklopná zadní část) vozu otevřený nebo špatně zavřený
- Jestli je kapota motoru otevřená nebo špatně zavřená
- Zda je rádio zapnuté před zapnutím zapalování či nikoliv
- Světlo v interiéru
- Nastavování sedadel do doby startu motoru

Díky násobnosti okolních vlivů se dostane počet možností do řádu 10^6. Je velmi pravděpodobné, že nemůže být otestována každá možnost, která může nastat. Hlavní cíl testování a simulací probíhá pro možnosti, které jsou u zákazníka časté a měření pro danou problematiku probíhá několik dní. Výsledky měření udávají nový směr vývoje řídicích jednotek a jejich náročnost na klidové proudy.

Kapitola 3

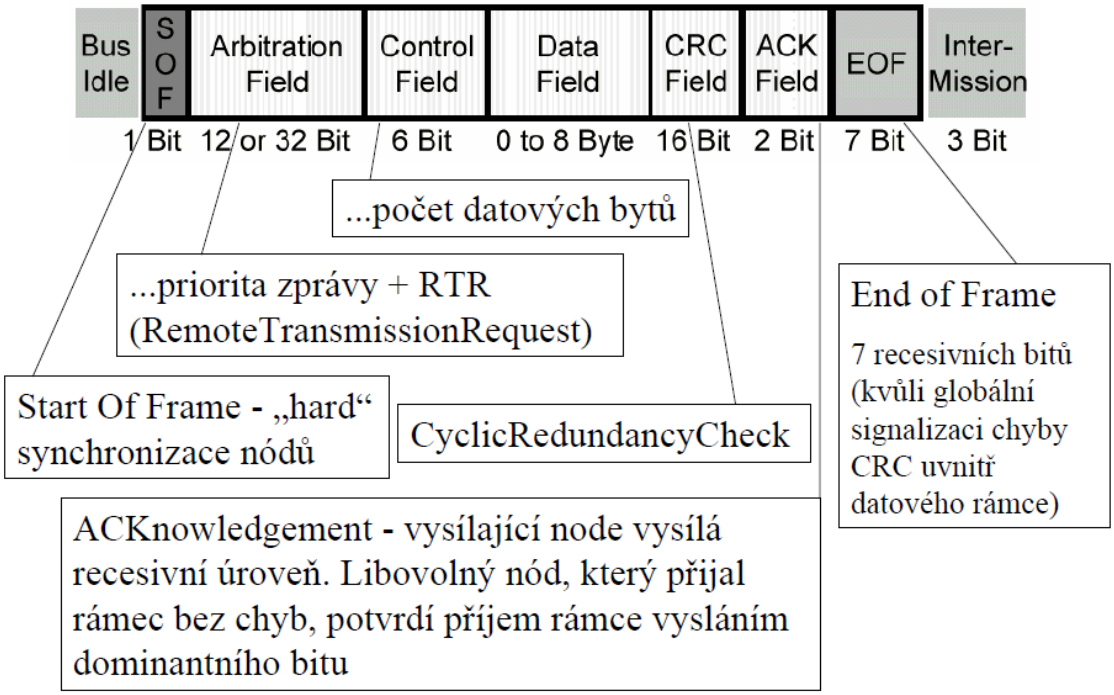
CAN sběrnice

Controller Area Network (CAN) je sériový komunikační protokol, který efektivně podporuje distribuovanou komunikaci v reálném čase s vysokou úrovní zabezpečení. Sběrnice CAN byla původně vyvinuta pro automobilový průmysl. Pro její dobré parametry a vlastnosti je rozšířena v aplikacích, které vyžadují vysokou komunikační rychlost až po aplikace vyžadující finanční dostupnost sběrnicové komunikace nebo multiplexní použití. Zdroje k obsahu této kapitole jsou [4], [3], [2].

3.1 Základy CAN komunikace

Na CAN sběrnici existují dva definované logické stavy. Dominantní stav reprezentuje logická nula a recesivní stav reprezentuje logická jednička. Platí pravidlo, že pokud na sběrnici je současně vysílán dominantní a recesivní bit, pak výsledný stav na sběrnici je dominantní. Informace po sběrnici se posílá ve formě CAN zprávy ve stanoveném formátu, ovšem s proměnnou délkou zprávy v závislosti na délce identifikátoru. Formát CAN zprávy je vidět na obrázku 3.1 Identifikátor zprávy hraje roli při určování priorit CAN zpráv. Čim nižší je identifikátor zprávy, tím vyšší je priorita zprávy.

Pro přidání nového zařízení do CAN sítě nepotřebujeme znát mnoho informací o konfiguraci sítě. To má za následek několik výhod. Systém je flexibilní, protože nevyžaduje žádné výrazné změny v softwaru nebo hardwaru přidaného zařízení. V síti není žádné směrování. Identifikátor pouze určuje o jakou se jedná zprávu a přibližně vypovídá o jeho datovém obsahu vyplývající z protokolu definujícím CAN zprávy. Nikoliv o místě kam má být zpráva doručena nebo odkud byla odeslána. Každé zařízení má filtr identifikátorů,

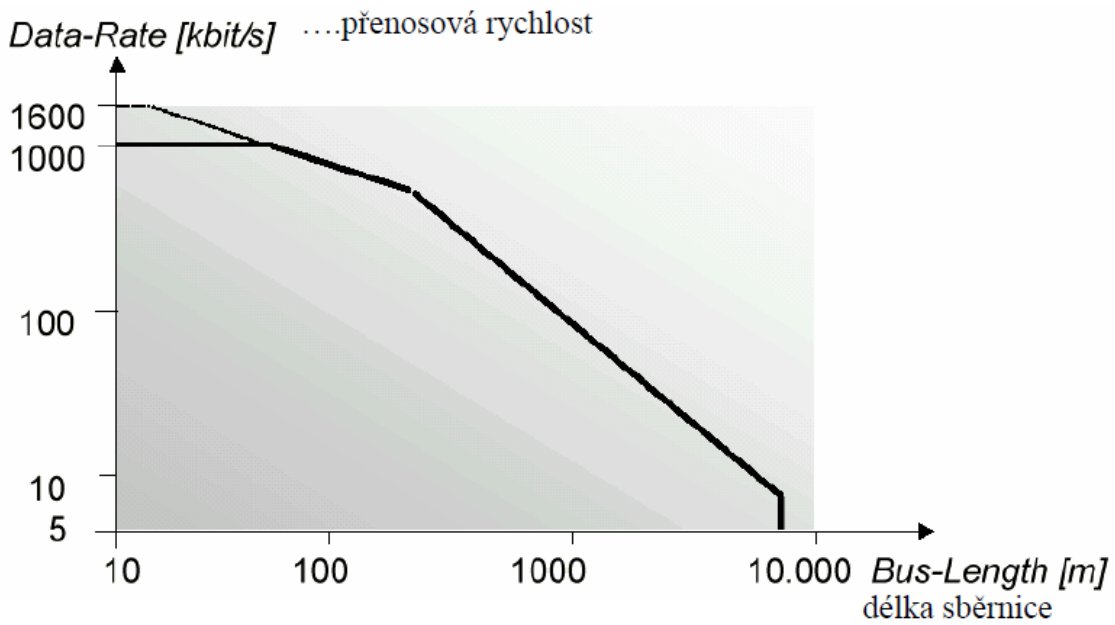


Obrázek 3.1: Struktura CAN zprávy

podle jehož nastavení se rozhodují, jestli zpráva je určena pro dané zařízení nebo ne. Na základě filtrování identifikátorů je možné jednu zprávu zpracovat více zařízeními. Rychlost komunikace je závislá na typu vedení a jeho délce. Na obrázku 3.2 je vidět závislost rychlosti komunikace CAN sběrnice na délce vedení. Obrázek ukazuje rychlost na metalickém vedení. Pro dosažení komunikace na delší vzdálenost lze použít posilovače sběrnice.

Detekce chyb v CAN zprávě má vysokou úroveň účinnosti. Detekuje se, zda nejsou definovány rozdílné chybové rámce, chybu úrovně signálu, využívá se generování 15-ti bitového CRC kódu s Hammingovou vzdáleností 6 a je schopen odhalit 5 chyb v náhodně rozložených bitech, zda nebyl porušen formát zprávy, porušení bit stuffingu(po každých pěti bitech stejné polarity je vložen bit opačné polarity) nebo nepotvrzení přijaté zprávy.

Při komunikační rychlosti 1Mbit/s se zatížením sítě 50 %, doby životnosti zařízení 4000 hodin a s průměrnou délkou zpráv 80 bitů je pravděpodobnost neodhalené chyby menší než 10^{-2}.



Obrázek 3.2: Závislost rychlosti komunikace na délce metalického vedení

3.2 Sběrnicová struktura v automobilu

V předchozí kapitole byly uvedeny základy a princip CAN komunikace a nyní se zaměříme na strukturu sběrnic v automobilu. Podle ISO se dělí CAN sběrnice na dva typy:

- CAN High-speed - ISO 11898 pro komunikační rychlost 125 kbit/s až 1000 kbit/s
- CAN Low-speed = ISO11519-2 pro komunikační rychlost $<$ 125 kbit/s

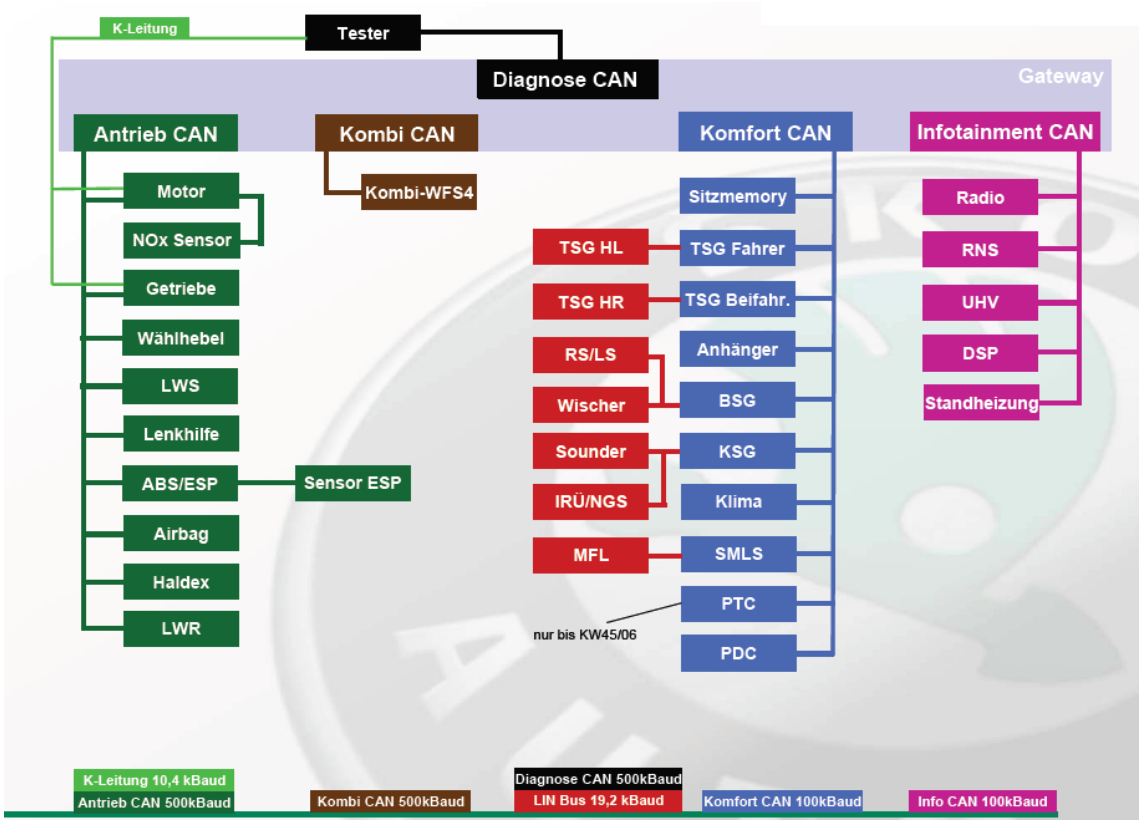
Tato rozdělení vycházejí z komunikační rychlosti sběrnice a následně z přiřazení pro určité aplikace. Řízení spalování směsi nebo regulace brzdného účinku kol jsou rychlé děje, které nejsou člověkem vůbec postřehnutelné, kritický by ovšem byl výsledek pomalého zákroku např. elektroniky ABS nebo ESP. Je zřejmé, že řízení motoru, ABS, automatické převodovky vyžaduje řízení v reálném čase s rychlými odezvami a velkými přenosovými rychlostmi, zatímco elektronika centrálního zamykání, elektrického stahování oken, klimatizace nebo autorádio může mít odezvy pomalejší, neboť výsledné zpoždění je pro lidské smysly stejně nevnímatelné a nepodstatné.

3.2.1 Rozdělení a pojmenování CAN sběrnic v koncernu Volkswagen

Pro potřeby nároků zákazníků, plné funkčnosti vozidla a spolehlivosti sběrnicové komunikace vzniklo v automobilu několik sběrnic mající na starosti komunikaci pouze mezi určitým počtem řídicích jednotek. Toto rozdělení vzniklo na základě náročnosti aplikací na rychlosti sběrnicové komunikace nebo z hlediska přetížení sběrnice a s tím spojené snížení spolehlivosti přenosu dat. Níže je uvedeno rozdělení a pojmenování sběrnic.

- **CAN Antrieb** - odpovídá CAN High-speed podle ISO11898, tedy vysokorychlostní sběrnici určené pro pohonnou část, tj. hlavně pro řídicí jednotku motoru, ABS, automatické převodovky atd.
- **CAN Kombi** - odpovídá CAN High-speed podle ISO11898, tedy vysokorychlostní sběrnici určené pro panel přístrojů. Vysoká rychlost je potřeba kvůli výpočtům mezi jednotkami a velkému toku dat.
- **CAN Diagnose** - odpovídá CAN High-speed podle ISO11898, tedy vysokorychlostní sběrnici určené pro diagnostiku vozu.
- **CAN Komfort** - odpovídá CAN Low-speed podle ISO11519-2, tedy nízkorychlostní sběrnici určené pro komfortní zařízení, tj. řídicí jednotka centrálního zamykání, elektrického stahování oken, alarmu, klimatizace atd.
- **CAN Infotainment** - odpovídá CAN Low-speed podle ISO11519-2, tedy nízkorychlostní sběrnici určené pro informační systémy, tj autorádio, navigační systém, telefon, CD changer atd.
- **LIN (Local Interconnect Network)** - pomalá a levná sběrnice fungující jako podsystém pro CAN sběrnici určená k připojení "inteligentních" senzorů nebo akčních prvků.
- **K-vedení** - sériová sběrnice poskytující doplňující informace jednotkám nebo pro diagnostiku konkrétních jednotek

Sběrnice CAN Komfort a CAN Infotainment jsou svými přenosovými parametry shodné, rozdíl je pouze v připojení odlišných řídících jednotek k těmto sběrnicím. Sběrnice mohou být oddělené nebo sloučené do jedné, záleží na konkrétním projektu.



Obrázek 3.3: Struktura zasíťování automobilu

Na obrázku 3.3 je ukázka zasíťování automobilu Škoda Octavia 2.0 a napojení řídicích jednotek na jednotlivé typy CAN sběrnic. Počet jednotek je závislý na úrovni vybavení nebo typu automobilu. U některých projektů se také liší počet CAN sběrnic. Zejména v případě sběrnic CAN Komfort a CAN Infotainment se u méně vybavených automobilů nebo u automobilů nižší třídy sdružují do jedné CAN sběrnice z důvodu malého vytížení obou sběrnic v případě malého počtu jednotek v automobilu.

Při zavedení několika sběrnic do jednoho vozu, potřebujeme zaručit, aby se informace předávali mezi jednotlivými řídicími jednotkami, které jsou připojeny na různých CAN sběrnicích. Řešením tohoto problému je připojením těchto sběrnic do jednoho uzlu, abychom zajistili předání všech patřičných informací. Tato jednotka se nazývá "Gateway". Tato jednotka je také zobrazena na obrázku 3.3. "Gateway" nemusí být branou jen pro sběrnici CAN, ale může sdružovat více typů sběrnic. Příkladem je použití K-vedení pro diagnostiku některých řídicích jednotek.

Kapitola 4

Metody měření proudu

Při měření proudů v automobilu musíme brát ohled na velikost měřícího zařízení, jelikož prostory, např. v motorové části, jsou velmi omezené a manipulace s velkým měřícím přístrojem v motorové části i mimo ni je limitovaná. Přístup k jednotlivým jednotkám je buď znepřístupněn nebo se k nim velmi těžce dostaneme. Dalším požadavkem při měření proudu v automobilu je zpětná analýza naměřených hodnot z důvodu minimalizace proudů protékajících vozidlem v případě měření trvající i několik hodin nebo dní. Potřebujeme tedy k tomu nějaké další záznamové zařízení, na které se budou ukládat měřené veličiny. Nabízejí se dvě základní varianty měření proudu, které budou popsány v následujících podkapitolách.

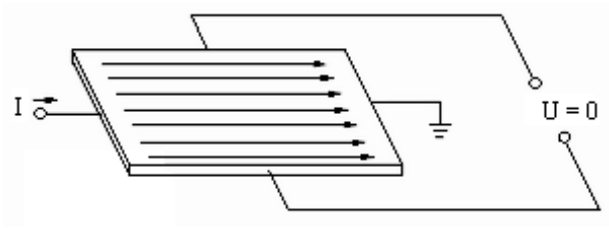
4.1 Měření proudu v obvodu bez jeho rozpojení

Dnešní moderní elektronika nám umožňuje měřit proud bez zásahu do zařízení, tudíž ulehčuje měření, zvyšuje efektivnost a rychlost práce. Využívá se magnetického pole kolem vodiče, kterým protéká proud. S rychlým vývojem technologií polovodičových součástek a objevením tzv. Hallova jevu v roce 1879 fyzikem Edwinem Hallem se k měření proudu v obvodu bez jeho rozpojení převážně požívá Hallova sonda.

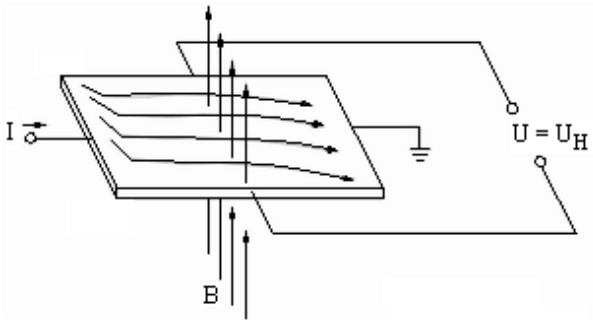
4.1.1 Hallův jev

Jak se píše v [13], Hallův jev spočívá ve vychylování směru toku elektrického proudu v závislosti na velikosti indukce magnetického pole B, které je kolmé na polovodičovou

tenkou destičku, tzv. Hallův element. Výsledkem je generování rozdílového napětí na bočních stranách elementu úměrné právě velikosti působícího magnetického pole či jeho kolmosti vzhledem k destičce.



Obrázek 4.1: Hallův element bez působení magnetického pole



Obrázek 4.2: Hallův element s působením magnetického pole

Pokud tedy Hallův element, jímž protéká konstantní hodnota proudu I_C , není vystaven působení magnetického pole B, je napětí U_H na jeho svorkách nulové, viz. obrázek 4.1. Pokud se objeví v jeho okolí magnetické pole, působí na elementem procházející proud elektronů tzv. Lorenzova síla, která elektrony vychyluje z přímého směru vždy k jedné boční straně destičky silou

$$F = Q(v \times B) \tag{4.1}$$

kde Q je elektrický náboj, v je jeho rychlost a B je indukce působícího magnetického pole. Změní se tak rozložení náboje, kdy na jedné straně je větší koncentrace nosičů náboje než na druhé a tedy obě boční stěny destičky mají rozdílný potenciál. Vzniká tak elektrické pole E a na svorkách Hallova elementu se generuje tzv. Hallovo napětí U_H, viz. obrázek 4.2.

Pokud je proud i magnetické pole konstantní je i napětí neměnné. To je dáno vznikem rovnováhy sil v destičce, kde Lorenzovu sílu kompenzuje opačně orientovaná síla vzniklého elektrického pole

$$F = Q \times E \tag{4.2}$$

Výstupní napětí je tedy proměnné pouze při proměnném proudu či proměnné magnetické indukci B. Velikost Hallova napětí je tedy dána

$$U_H = R_H \cdot I_C \frac{B}{d} \tag{4.3}$$

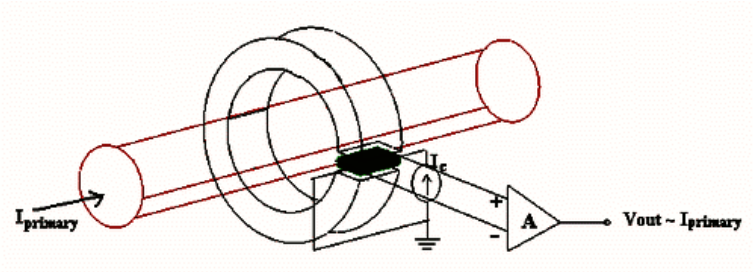
R_H je Halova konstanta, která je závislá na typu použitého polovodiče, d je šířka Hallova elementu. Polarita Hallova napětí je pak závislá na polaritě magnetického pole a procházejícího proudu. Pokud není magnetické pole kolmé na plochu elementu, musíme ještě přidat sinus úhlu odklonění směru magnetického pole od kolmé osy.

4.1.2 Realizace bezkontaktního měření proudu

Již samotný Hallův element je jednoduchý snímač magnetického pole. Pokud jej tedy doplníme vyhodnocovací elektronikou, která nejen zesiluje, upravuje a standardizuje výsledné Hallovo napětí, ale i reguluje a stabilizuje napájecí napětí elementu pro generování konstantního proudu, vznikne kompletní snímač. Udržení konstantního proudu je totiž základní podmínkou pro to, aby se změna Hallova napětí rovnala pouze změně hodnoty indukce magnetického pole.

4.1.2.1 Hallův senzor v otevřené smyčce

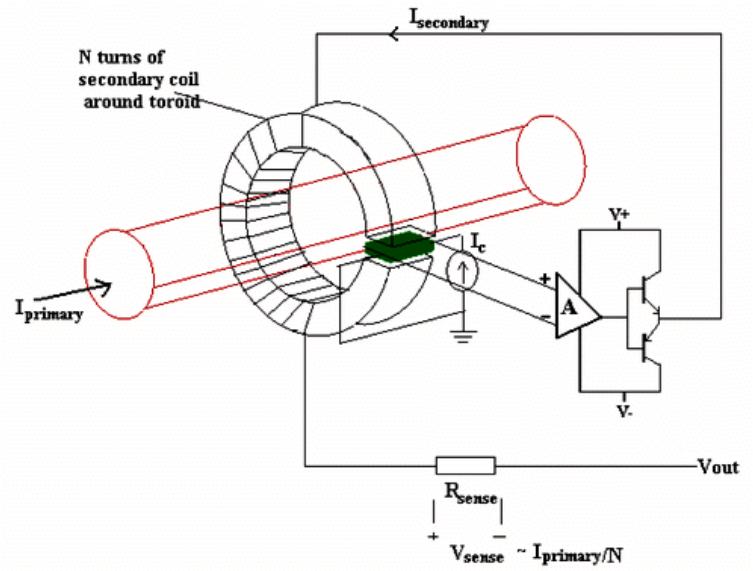
V prvním případě je výhodou velmi snadná realizace měření, kdy se Hallovým senzorem umístěným ve vzduchové mezeře feromagnetického prstence měří intenzita magnetického pole. Prstenec obepíná měřený vodič a usměrňuje magnetické pole právě do místa se senzorem. Ten pak na svém výstupu dává proud, úměrný Hallově napětí. Nevýhodu obvykle bývala menší přesnost měření způsobená nepřesností a offsetem Hallova snímače a známými negativními jevy v feromagnetickém jádře (smyčkové proudy a jen omezeně lineární magnetizační B-H charakteristika). Princip zapojení je vidět na obrázku 4.3.[12]



Obrázek 4.3: Princip měření s Hallovým senzorem v otevřené smyčce

4.1.2.2 Hallův senzor v uzavřené smyčce

Nevýhody měření v otevřené smyčce je možné úplně vyrušit měřením v uzavřené smyčce, kde se zpět magnetizuje feromagnetický prstenec opačným proudem tak, že ve výsledku se negativní jevy vyruší. V prstenci se od magnetického toku vzniklého měřeným proudem odečte magnetický tok generovaný zpětnovazebním proudem regulovaným Hallovým snímačem tak, aby se obě tyto složky vyrovnaly a odečetly. Tím dosáhneme nulové magnetické indukce na Hallově snímači. Výsledkem měření je pak regulační proud v uzavřené smyčce. Tento způsob poskytuje sice velmi vysokou přesnost měření, ale je složitý na realizaci a tedy i drahý. Princip zapojení je opět vidět na obrázku 4.4.[12]



Obrázek 4.4: Princip měření s Hallovým senzorem v uzavřené smyčce

4.2 Měření proudu v obvodu s jeho rozpojením

Při měření proudu metodou rozpojení obvodu je míněno to, že do obvodu vložíme další prvek, který usnadní měření proudu. Při jakémkoli zásahu do obvodu se naruší povaha a vlastnosti obvodu, takže nově vytvořený obvod s vloženým prvkem nebude úplně totožný s původním. Snahou je minimalizovat vliv nežádoucích veličin na měřený obvod. Nežádoucí veličinou může být ztrátový výkon ve formě tepla vytvářeného v přidaném prvku nebo změna povahy součástky vlivem stárnutí pro delší časový horizont.

4.2.1 Princip

Princip metody je prostý. Jedná se vložení rezistoru do obvodu se známou velikostí a měří se velikost úbytku napětí na rezistoru. Výsledná hodnota proudu se vypočítá z Ohmova zákona

$$I = \frac{U}{R} \tag{4.4}$$

kde U je úbytek napětí na rezistoru o známé velikosti R a I je proud protékající rezistorem.

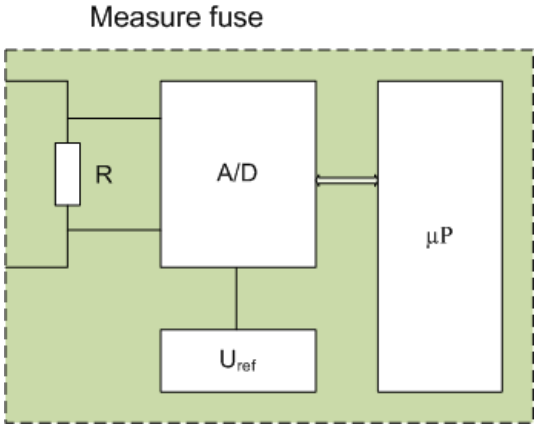
Zmiňovaná metoda je pouze základním zapojením a existuje mnoho modifikací, které dovolují měřit proud ve velkém rozsahu s dostatečnou přesností měření. Pro změnu rozsahu měřeného proudu se paralelně na měřený rezistor připojí tzv. bočník, který zajistí, aby měřícím rezistorem tekl pouze dovolený proud a nedošlo ke zničení měřícího rezistoru. Dále je možné připojit obvod s operačním zesilovačem pro zesílení měřeného signálu nebo analogově - digitální převodník pro další digitální zpracování např. v mikropočítači.

4.2.2 Realizace metody s rozpojením obvodu

Firma Škoda auto a.s. ve spolupráci s externí firmou vyvinula systém pro měření proudů splňující požadované podmínky na měřící systém, který je předmětem této diplomové práce. Tento systém se nazývá Mefuse a je složen ze dvou základních komponent. První z nich je tzv. inteligentní pojistka, která zprostředkovává samotné měření proudu metodou s rozpojením obvodu. Druhou částí je ústředna zpracovávající informace posílané z inteligentní pojistky a zajišťuje komunikaci po CAN sběrnici s nadřazeným systémem nebo mezi jednotlivými ústřednami.

4.2.3 Inteligentní pojistka

Inteligentní pojistka se skládá z měřícího rezistoru z jehož napěťového úbytku se podle Ohmova zákona 4.4 vypočítá velikost proudu protékající obvodem. Úbytek napětí je vzorkován 18-ti bitovým Sigma-Delta A/D převodníkem a výstupní data z A/D převodníku přepočítána na hodnotu proudu v mikropočítači. Součástí inteligentní pojistky je i měření teploty okolí díky integrovanému teplotnímu čidlu na mikroprocesoru. Blokové schéma je na obrázku 4.1.



Obrázek 4.5: Blokové schéma inteligentní pojistky

Inteligentní pojistka existuje v několika provedení s různou nominální hodnotou protékajícího proudu. Maximální rozlišení pojistky je $100\mu A$, relativní přesnost je 1% z měřené hodnoty. Další údaje o inteligentních poojistkách jou uvedeny v tabulce 4.4.

Nominální proud pojistky	5A	10A	20A	30A
Absolutní přesnost	$\pm 100\mu A$	$\pm 200\mu A$	$\pm 400\mu A$	$\pm 600\mu A$
Drift nulové hodnoty	$\pm 100\mu A$	$\pm 200\mu A$	$\pm 400\mu A$	$\pm 600\mu A$

Tabulka 4.1: Přesnost inteligentních pojistek

4.2.4 Ústředna Mefuse

Napájení ústředny je realizováno několika způsoby. Základní napájení je tvořeno externím zdrojem napětí 6-15V. V případě absence externího zdroje je v ústředně integrována baterie, případně je možnost připojit externí bateriové napájení. Samotná

ústředna má několik módů energetického managementu pro optimalizaci co nejdelší výdrže systému při napájení z baterie.

Komunikace mezi senzory a ústřednou probíhá přes galvanicky oddělenou sériovou linku. Počet senzorů na jednu ústřednu je maximálně osm. Komunikace s nadřazeným systémem probíhá přes CAN řadič nebo BlueTooth modul integrované v ústředně. BlueTooth komunikace je využita pro podporu PDA zařízení z důvodů jednoduché a mobilní podpory navrženého systému. CAN sběrnice je využita pro komunikaci mezi jednotlivými ústřednami, s nadřazeným počítačem nebo dataloggerem. Pro CAN komunikaci s ústřednami byl definován CAN protokol se zprávami nastavující režim a způsob měření ústředny nebo naopak, informace posílané z ústředen o jejich stavu a data z aktuálního měření. Veškeré zprávy probíhající mezi ústřednami nebo s nadřazeným počítačem jsou podrobně popsány v Příloze A.

Do modulu je také integrována čtečka paměťových karet na ukládání naměřených dat pro následnou zpětnou analýzu měřených veličin. Ústředna podporuje paměťové karty typu SD o maximální velikosti 2GB. Ukládaná data na SD kartu jsou v následující adresářové struktuře:

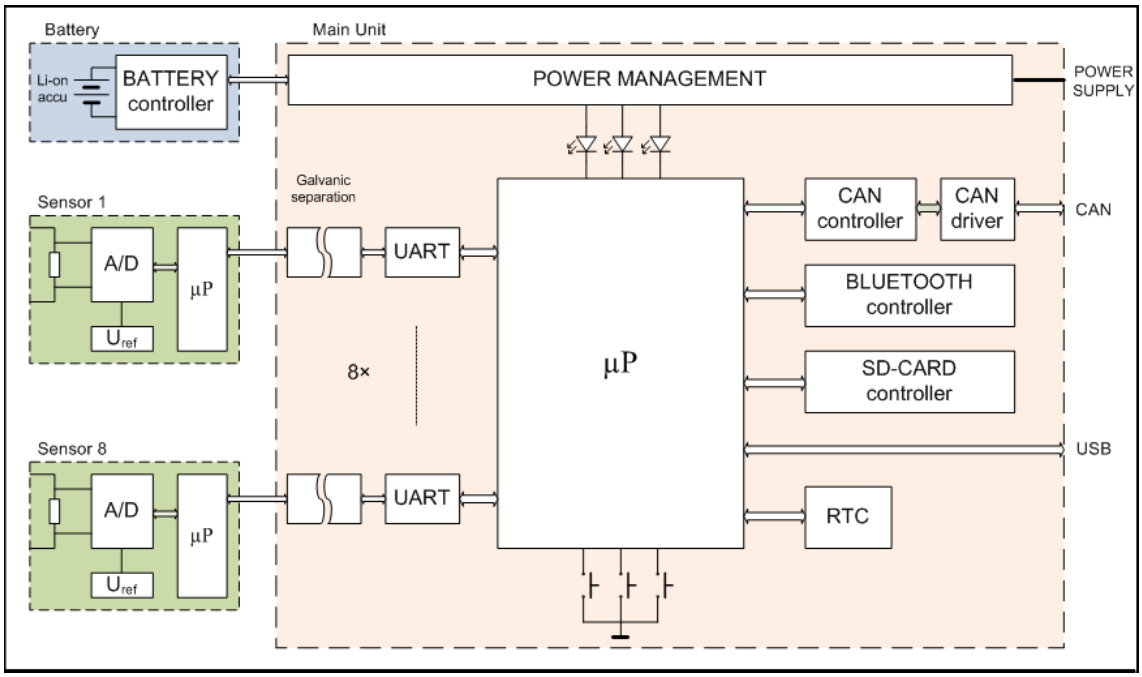
- Settings
- Log_file
  - rrmmddhh
    - config.txt
    - ddhhmmss.bin
    - ddhhmmss.bin
    - ⋮
  - rrmmddhh
  - ⋮

V adresáři "Settings"jsou uloženy konfigurační data a spouštěcí podmínky pro případ nastavení logování přímo z karty. V adresáři "Log_file"jsou ukládány logovaná data. Při každém spuštěném a dokončeném logování je vytvořena nová složka **"rrmmddpp"**, kde rr, mm, dd je rok, měsíc a den počátku logování, pp je pořadí logování v daném dnu, které se při každém novém startu inkrementuje.

Uvnitř složky jednotlivých logovaných dat je vždy umístěn textový soubor "Config.txt", obsahující informace o konfiguraci systému v době, kdy začalo logování. Uvnitř složky "rrmmddhh"jsou dále umístěny binární soubory s vlastními logovanými daty "ddhhmmss.bin", kde dd, hh, mm, ss je den, hodina, minuta, vteřina počátku logování. Logovány jsou CAN zprávy dle nastavení ve zprávě "Logging_Setting". Každá CAN zpráva je uložena v 16 bytech v následujícím formátu:

1. - 5. byte reálný absolutní čas přijetí CAN zprávy
6. - 7. byte identifikátor zprávy
8. - 15. byte datový obsah zprávy
16. byte rezerva

Celé blokové schéma je na obrázku 4.6, kde je vidět základní struktura celého systému Mefuse.



Obrázek 4.6: Blokové schéma systému Mefuse

4.3 Vyhodnocení metod

Obě metody měření proudu, po zvolení dostupných senzorů, mají dostačující přesnost a spolehlivost. Ovšem jedním z hlavních požadavků na měření proudů v automobilu je možnost měření jednotlivých obvodů. V případě metody bez rozpojení obvodu je to problematické, protože přístup k jednotlivým vodičům daného obvodu je značně omezený z důvodu úspory volného místa v automobilu. Nedostatek místa tedy zabraňuje se dostat do míst, kde je daný obvod umístěn. Dalším problémem použití této metody je, že většina elektroinstalace v automobilu je sdružována do svazků a to zabraňuje separaci konkrétního obvodu nebo velké pravděpodobnosti elektromagnetického rušení od ostatních systémů.

Jako vyhovující metodu jsem zvolil metodu s rozpojením obvodu, jelikož splňuje podmínku separovat jednotlivé obvody ať už z místa pojistkové skříně nebo rozpojením určitého svazku elektroinstalace. Další výhodou této metody je již vyvinutý hardware, který umožňuje samotné měření pomocí inteligentní pojistky a měřená data je schopen distribuovat v digitální formě pro jiná elektronická zařízení. Vyhodnocením těchto dvou metod jsem došel k závěru, že metoda s rozpojením obvodu je výhodnější při použití již vyvinutého systému Mefuse.

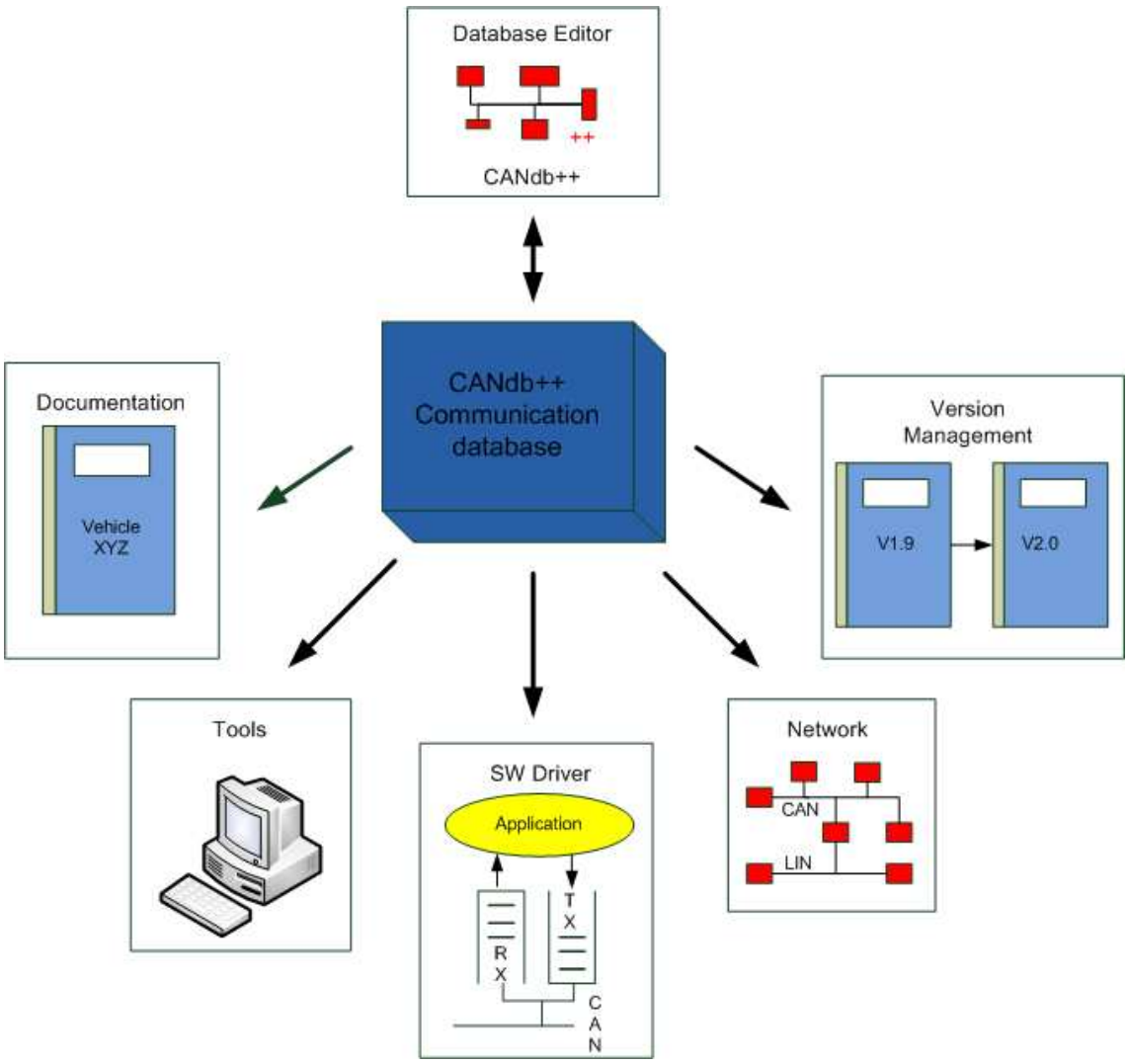
Kapitola 5

Vývoj vyzualizačního software pro měřená data

Hlavními požadavky na vizualizační software systému měření klidových proudů realizovaný měřící ústřednou Mefuse je, aby měřená data byla zobrazována v reálném čase, uživatelsky příjemné nastavování měřících režimů senzorů a měřící ústředny Mefuse. To vše přes CAN sběrnici. Pro zpracování CAN zpráv z ústředny a nastavování ústředny je nejvhodnější zvolit software CANoe. CANoe je univerzální vývojový software pro testování a analýzu CAN komunikace. Sdružuje více podprogramů, které svými funkcemi umožňují vytvořit vizualizační software s danou funkcionalitou a potřebným uživatelským rozhraním. Možnosti jednotlivých podprogramů budou vysvětleny v níže uvedených podkapitolách.

5.1 Software CANdb++ Editor

Veškerá platná data, která jsou posílána po síti CAN sběrnic a jejich vzájemné vztahy jsou spravovány v centrální komunikační databázi. CANdb++ je software pro správu těchto dat a umožňuje vytvářet nebo měnit CAN databázi. Struktura a použití komunikační CAN databáze je vidět na obrázku 3.1.



Obrázek 5.1: Komunikační CAN databáze

5.1.1 Objekty v CANdb++

V CANdb++ je možné definovat několik objektů, které umožňují databázi rozdělit do více projektů a vláken pro snadnější orientaci v databázi a oddělení nezávislých objektů. Možnost vytvoření následujících objektů:

- Vehicle - rozdělení do projektů (kombi,kupé)
- Networks - rozdělení podle jednotlivých CAN sběrnic v daném projektu
- Control units (ECUs) - sdružuje definované Environment variable

- Environment variables - proměnné pro práci s grafickým rozhraním aplikace
- Network nodes - sdružuje CAN zprávy pro řídicí jednotku na sběrnici
- Messages - definice CAN zprávy, její název, obsažené signály, zda se jedná o zprávu Tx nebo Rx
- Signals - definice signálů v CAN zprávě, umožňuje jednodušší dekódování bytu CAN zprávy

Tabulka 5.2 ukazuje všechny možnosti propojení mezi jednotlivými objekty.

	Objekty s nimiž je možno vytvořit spojeni						
Objekty pro propojení	Vehicles	Networks	Control Units (ECUs)	Node Groups	Network Nodes	Messages	Signal Groups
Networks	x						
Control Units (ECUs)	x						
Environment Variables			x				
Node Groups		x					
Network Nodes		x	x	x			
Messages					x		
Message Signals					x		x
Signal Groups						x	
Signals						x	

Obrázek 5.2: Tabulka možného propojení objektů

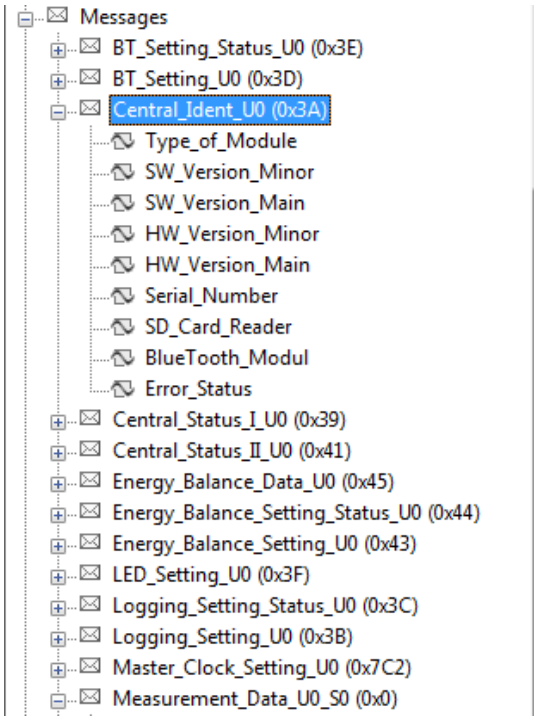
5.1.2 Objekt Signal

Objekt Signal z [9] definuje význam bitů ve zprávě. Při vytváření signálu určujeme jeho jméno, délku signálu v bitech, hodnotový typ (znaménkový, bez znaménka, double, float) a přiřazení signálu k dané zprávě. U signálu je také možné nastavit minimum a maximum hodnoty signálu, multiplikátor, který hodnoty signálu vynásobí, případně

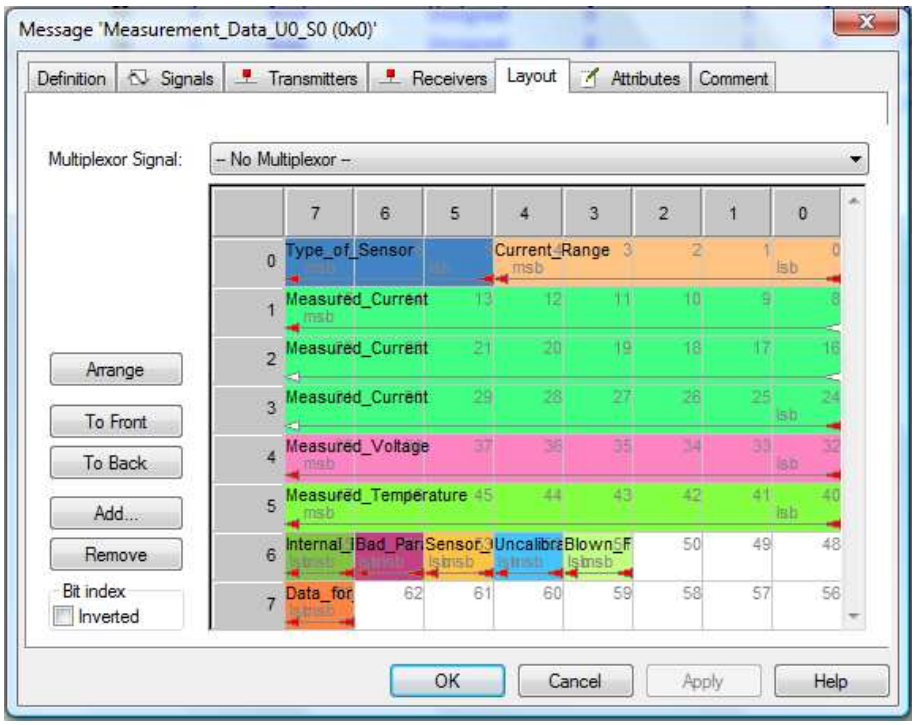
jednotky, ve kterých bude signál reprezentován nebo přiřadit k jednotlivým bitům v signálů informaci o významu bitu. Při použití těchto vlastností dosáhneme snadnější orientace ve zprávě, protože při výpisu záznamu CAN zprávy se zpráva rozloží na signály definující užitečnou informaci z dat ve zprávě. Při přepočtu dat v signálu se již rovnou zobrazují data ve správných jednotkách spolu se značkou o jaké jednotky jde nebo jen poznámka k příslušným bitům signálu.

5.1.3 Objekt Message

Objekt Message z [9] definuje CAN zprávu používanou na CAN sběrnici. Při vytváření CAN zprávy v CANdb++ definujeme parametry typu jméno zprávy, o jaký formát zprávy se jedná, jestli jde o standardní formát nebo o rozšířený formát, identifikátor zprávy a přiřazení signálů do zprávy. Při přiřazování signálů do zprávy se definuje počáteční bit, u kterého signál začíná. Vytvořená zpráva se signály v CANdb++ a rozložení jednotlivých signálů je vidět na obrázcích 5.3 a 5.4.



Obrázek 5.3: CAN zpráva se signály



Obrázek 5.4: Rozložení signálů ve zprávě

5.1.4 CAN zprávy pro ústřednu Mefuse

V tabulce 5.1 jsou uvedeny všechny definované zprávy pro měřící ústřednu Mefuse. Ve sloupci s identifikátory mají značky A0-A3 význam adresy příslušné ústředny. Na jednu CAN sběrnici můžeme tedy připojit až 16 ústředen zároveň. Značky CH0-CH2 reprezentují adresu senzoru u jedné ústředny, kterých může být maximálně 8. Důležité je, aby každá CAN zpráva definovaná pro ústřednu Mefuse měla unikátní identifikátor.

Výsledkem práce se softwarem CANdb++ je vytvoření databáze CAN zpráv podle tabulky 5.1 definující veškerou komunikaci ústředen Mefuse jak mezi jednotlivými ústřednami, tak i s uživatelským rozhraním pro zobrazování dat a nastavování jednotlivých ústředen. Součástí CAN zpráv jsou definované signály a nastavení případných výpočtů s daty v signálech nebo zobrazení poznámek u jednotlivých bitů v signálu. Podrobnější popis všech definovaných zpráv je obsahem Přílohy A.

Název	**Identifikátor**										
	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
Measurement_Data	A3	A2	A1	A0	0	0	0	0	CH2	CH1	CH0
Measurement_Setting	A3	A2	A1	A0	0	0	0	1	CH2	CH1	CH0
Measurement_Setting_Status	A3	A2	A1	A0	0	0	1	0	CH2	CH1	CH0
Sensors_Ident	A3	A2	A1	A0	0	0	1	1	CH2	CH1	CH0
Status_Query	A3	A2	A1	A0	0	1	1	1	0	0	0
Central_Status_I	A3	A2	A1	A0	0	1	1	1	0	0	1
Central_Ident	A3	A2	A1	A0	0	1	1	1	0	1	0
Logging_Setting	A3	A2	A1	A0	0	1	1	1	0	1	1
Logging_Setting_Status	A3	A2	A1	A0	0	1	1	1	1	0	0
BT_Setting	A3	A2	A1	A0	0	1	1	1	1	0	1
BT_Setting_Status	A3	A2	A1	A0	0	1	1	1	1	1	0
LED_Setting	A3	A2	A1	A0	0	1	1	1	1	1	1
Power_Setting	A3	A2	A1	A0	1	0	0	0	0	0	0
Central_Status_II	A3	A2	A1	A0	1	0	0	0	0	0	1
Master_Clock_Setting	1	1	1	1	1	0	0	0	0	1	0
Energy_Balance_Setting	A3	A2	A1	A0	1	0	0	0	0	1	1
Energy_Balance_Setting_Status	A3	A2	A1	A0	1	0	0	0	1	0	0
Energy_Balance_Data	A3	A2	A1	A0	1	0	0	0	1	0	1

Tabulka 5.1: Seznam CAN zpráv pro ústřednu Mefuse

5.2 Software Panel Editor

Panel Editor je software pro tvorbu grafického rozhraní. Základními možnostmi Panel Editoru je vytvoření panelu, na který se vkládají základní grafické prvky jako jsou tlačítka, posuvníky, "textboxy" nebo bitmapové obrázky. Pro funkčnost grafických prvků je třeba v CAN databázi definovat tzv. "Environment variables" a vytvořenou databázi propojit s aktuálním projektem v Panel Editoru. Hodnotový typ "Environment variable" může být typu string, data, float nebo integer. Každému grafickému prvku se musí přiřadit tato proměnná daného hodnotového typu. Podle hodnoty "Environment variable" se mění grafický prvek. Např. při změně proměnné se může měnit barva signalizační kontrolky nebo text v zobrazovacím okně. Stejně tak, jako se podle proměnné může měnit vzhled grafických prvků, tak i funkcí grafického prvku se může měnit hodnota "Environment variable".

5.2.1 Vývoj grafického rozhraní pro měřící systém

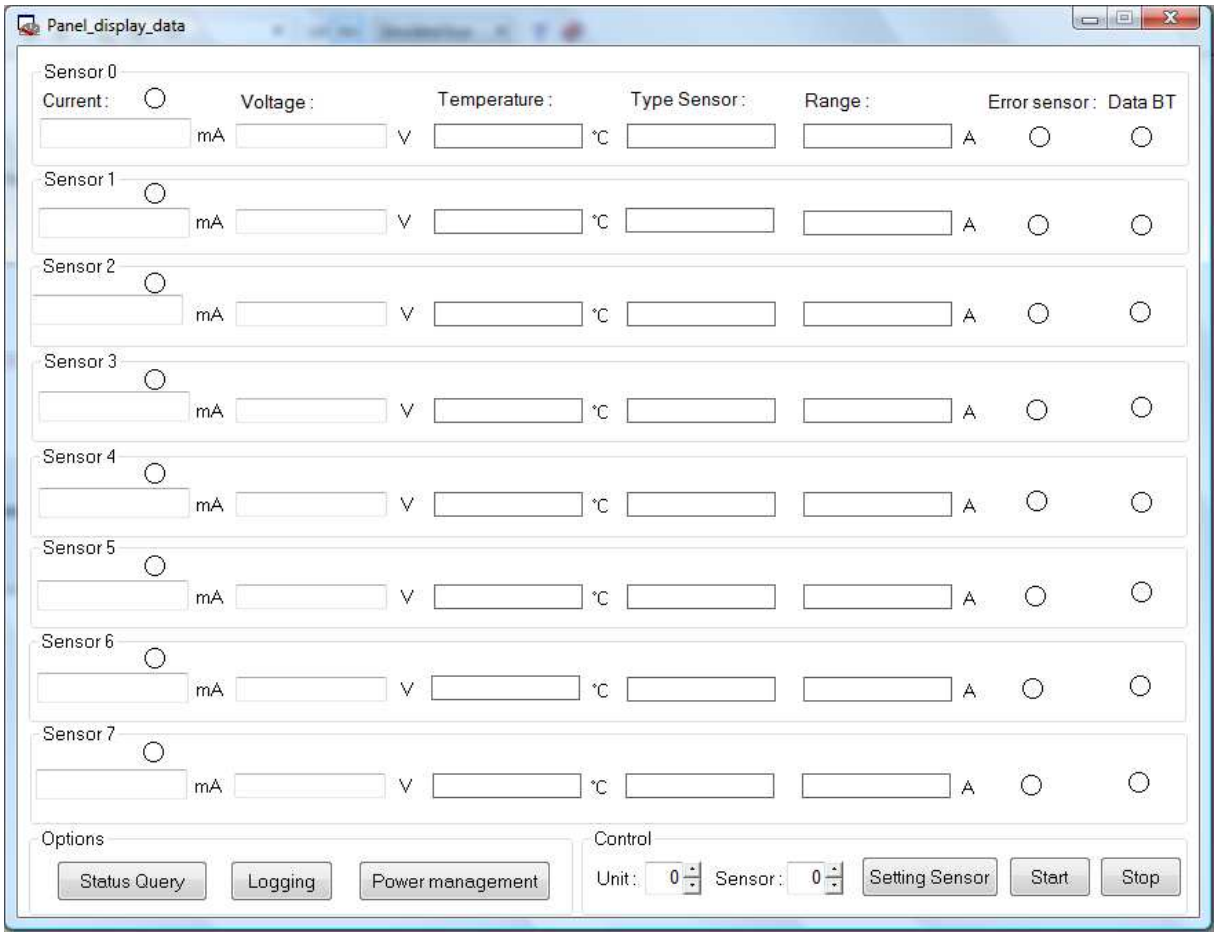
5.2.1.1 Hlavní měřící panel

Při prvním spuštění aplikace se zobrazí hlavní panel vizualizačního prostředí, ukázka je na obrázku 5.5. Hlavním požadavkem základního okna je zobrazování aktuálně měřených dat z ústředny. Proto velkou část hlavního okna zabírají zobrazovací prvky pro každý senzor připojený k ústředně. Přinášejí informaci o velikosti proudu, napětí, teploty, použité inteligentní pojistce, jejím maximálním rozsahu a informaci, jestli senzor funguje správně a zda jsou data ze senzoru vysílána přes komunikační rozhraní BlueTooth. Indikační kontrolka nad zobrazováním velikosti proudu signalizuje příjem zprávy požadovaného typu. Hlavní panel zpracovává pouze zprávu "Measurement_Data", která se liší pouze identifikátorem ústředny a senzoru.

Část hlavního panelu nazvaná "Control" obsluhuje spouštění a vypínání aplikace pomocí tlačítek "Start" a "Stop". Tato část je také určena k nastavení parametrů senzoru v ústředně pomocí tlačítka "Setting Sensor" po předchozím zvolení příslušné ústředny a senzoru vlevo od tlačítka.

Další část hlavního panelu nazvanou "Options" otevírá rozšiřující funkce vizualizačního softwaru. Umožňuje zjistit podrobnější informace o nastavení ústředny po stisku tlačítka "Status Query", nastavení parametrů logování, BT modulu a signalizačních LED diod na senzorech po stisknutí tlačítka "Logging" nebo nastavení energetického hospodaření

ústředny po stisku tlačítka "Power management". Obsluha funkcí podporovaná vizualizačním softwarem bude níže podrobně popsána.



Obrázek 5.5: Hlavní měřící panel

5.2.1.2 Obsluha Status Query

Po obsloužení události stisknutím tlačítka "Staus Query" se na obrazovce objeví nový panel viz. obrázek 5.6. Funkce tohoto panelu je následující. V levém horním rohu je zobrazena část s názvem "Status Query". Tato část po stisku tlačítka "Send" generuje zprávu "Status_Query", která je poslána příslušné ústředně. Označení ústředny je vidět v pravém horním rohu této části. Její hodnota se mění v hlavním panelu v části "Control". Při konfiguraci dotazu na konkrétní ústřednu můžeme volit identifikaci všech senzorů připojených k ústředně, jejich konfiguraci nebo si od ústředny vyžádat zprávy "Central_Ident", "BT_Status", "Central_Status_I", "Central_Status_II", "Logging_Status" nebo

"Energy_Balance_Status"viz. Příloha A. Případnou rekonfiguraci požadavku na ústřednu lze tlačítkem "Reset" smazat aktuální nastavení požadavku a začít znovu.

Po přijmutí zprávy "Central_Ident"se v levém dolním rohu panelu zobrazí informace o ústředně typu ID ústředny, sériové číslo, verze SW a HW, jestli nemá ústředna poruchu nebo zda jsou integrovány v ústředně BlueTooth modul a čtečka paměťových karet.

Po přijmutí zpráv o identifikaci senzorů "Sensor_Ident" a "Measurement_Setting_Status" zjistíme uprostřed panelu informace o připojených senzorech, jestli měří, s jakou periodou a průměrováním nebo jaké veličiny měří a kolik vzorků zbývá do ukončení měření.

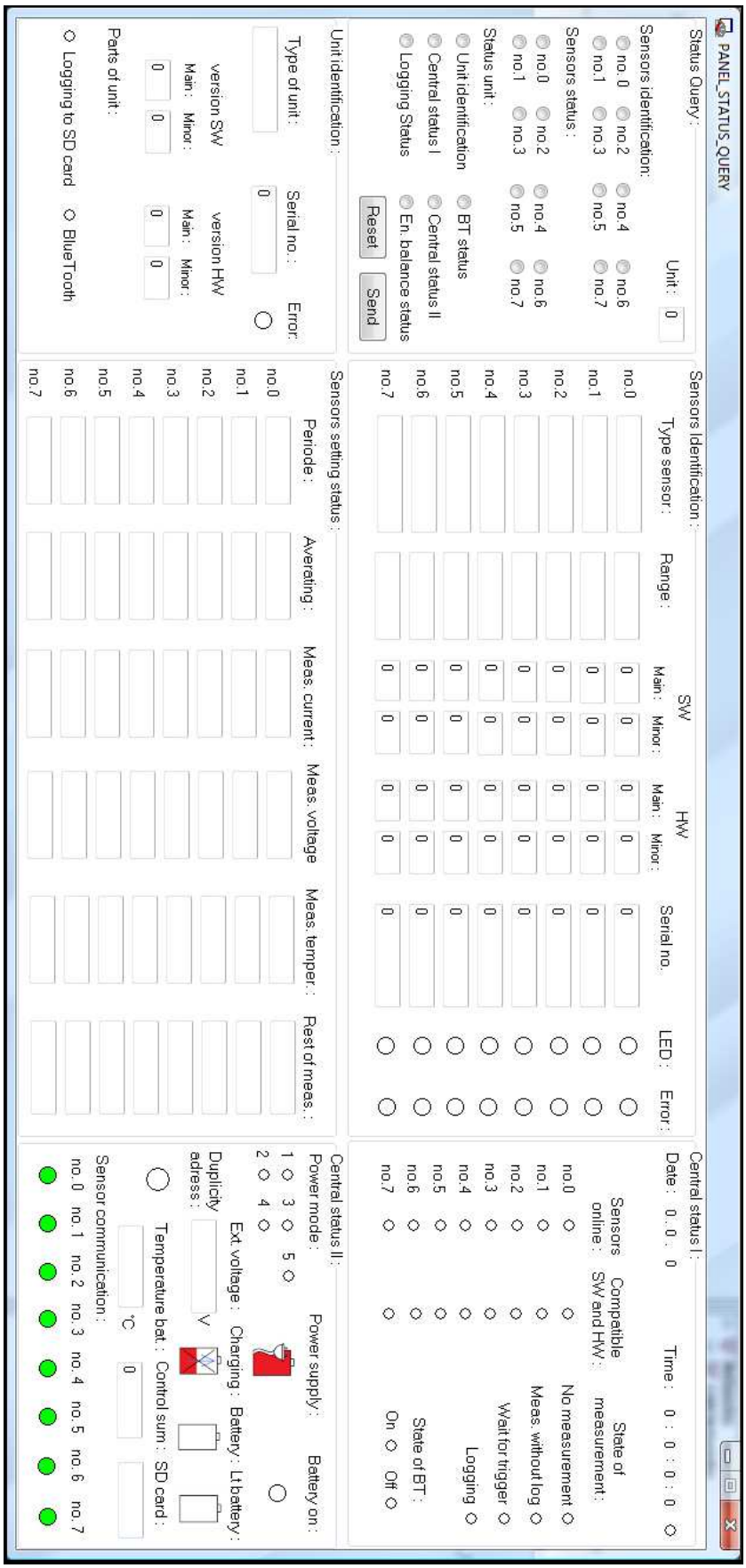
Zbylé části panelu zobrazují informace ze zpráv "Central_Status_I" a "Central_Status_II" o stavu ústředny z hlediska měření, kompatibility senzorů, stavu BlueTooth modulu, mód napájení ústředny nebo stav baterie a zaplnění SD karty.

5.2.1.3 Obsluha Logging

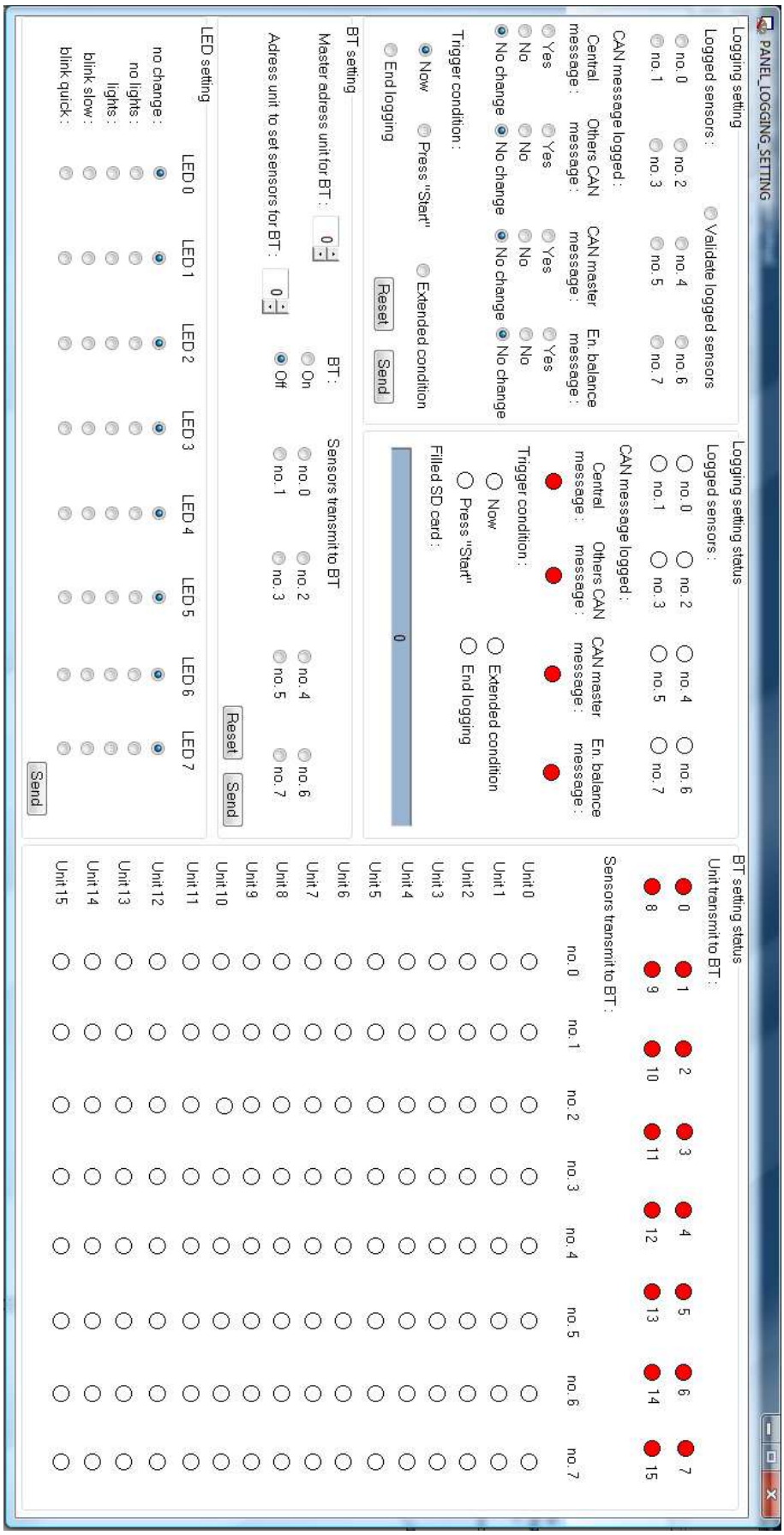
Po obsloužení události tlačítka "Logging" se zobrazí panel "Logging Setting" viz. obrázek 5.7, který umožňuje konfiguraci logovaných dat na SD kartu pomocí zprávy "Logging_Setting" vyslanou do ústředny, konfiguraci dat, která mají být posílána přes BlueTooth modul pomocí zprávy "BT_Setting" a konfiguraci režimu svícení LED diod na senzorech přes zprávu "LED_Setting".

Dále panel poskytuje informace ze zpráv "Logging_Setting_Status" a "BT_Setting_Status". Aby se zobrazili informace z těchto dvou zpráv, je potřeba si je vyžádat od ústředny zaškrtnutím požadavků na tyto zprávy v panelu "Status_Query".

Při vyžádání zprávy o statusu logování dostaneme informace jaká data z jakých senzorů, případně jaké další zprávy se budou ukládat na SD kartu a jaké jsou nastavené spouštěcí podmínky pro ukládání na kartu. V části "BT_Setting_Status" jsme informováni o tom, jaké jednotky mají zapnutý BlueTooth modul, a která data ze senzorů jsou posílána přes BlueTooth modul.



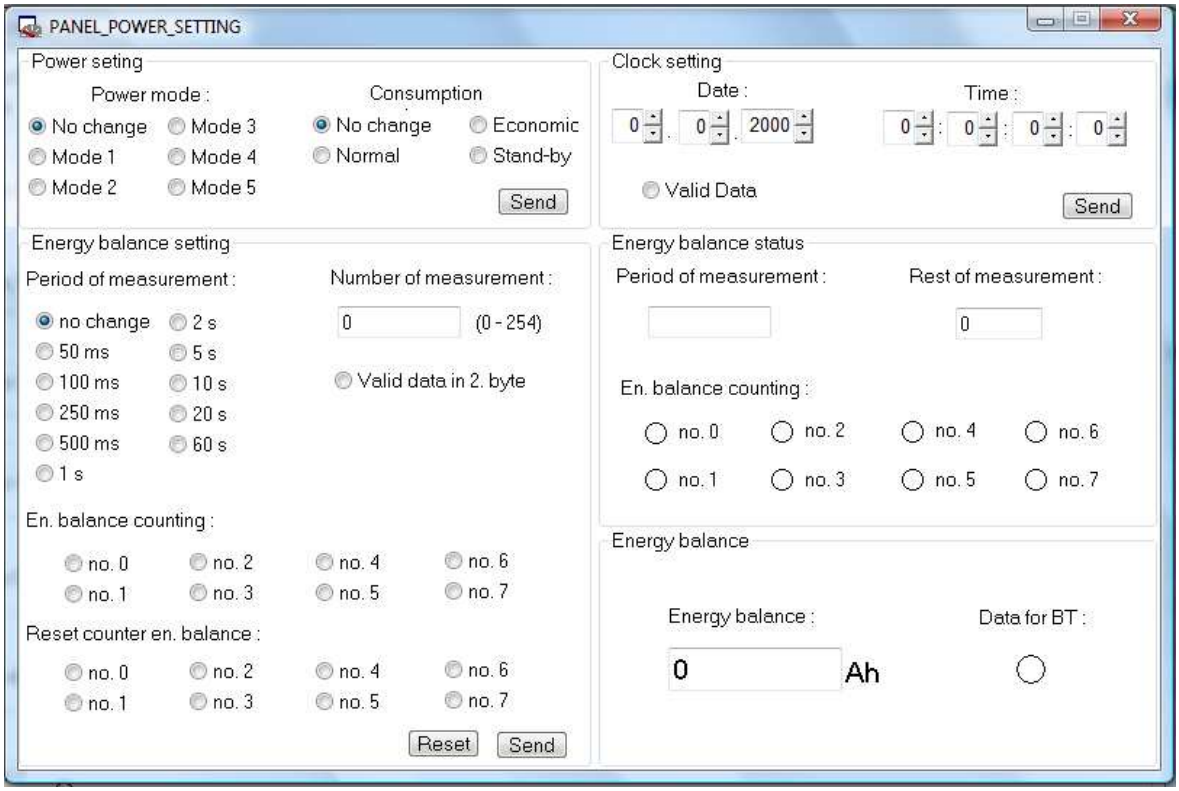
Obrázek 5.6: Panel pro zjištění stavu ústředny



Obrázek 5.7: Nastavení logování ústředny

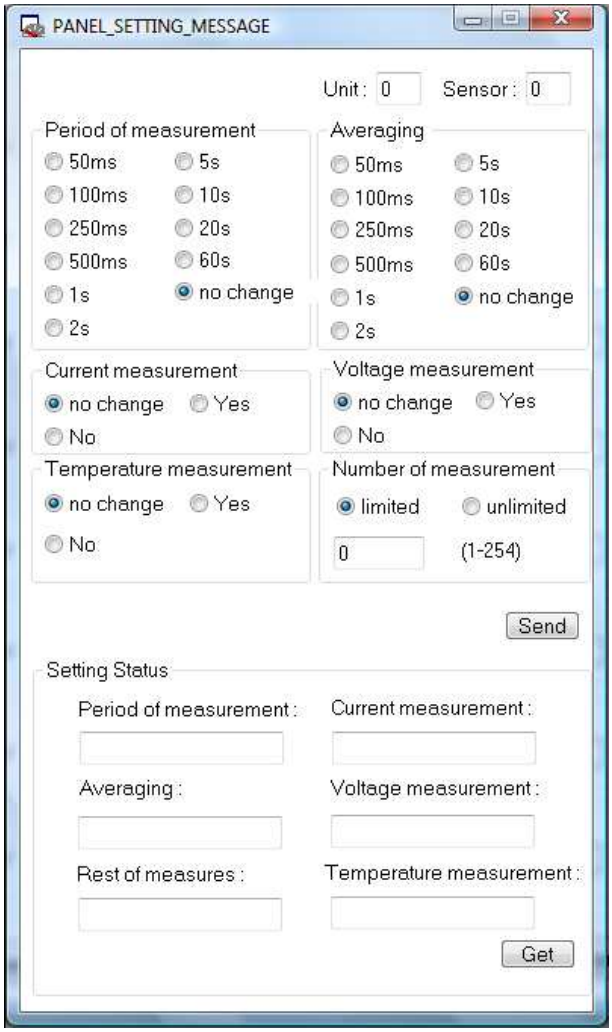
5.2.1.4 Obsluha Power Management

Po stisku tlačítka "Power Management"se objeví panel 5.8 se základním nastavením energetické spotřeby a módu napájení ústředny, zapnutím funkce pro výpočet energetické bilance z měřených veličin a synchronizací hodin v ústředně. Pokud chceme měnit mód napájení nebo změnit režim spotřeby poslouží nám část panelu s názvem "Power Setting", kde se nastavuje až pět módů napájení nebo tři různé způsoby řízení spotřeby elektrické energie. Po stisku tlačítka "Send" se do ústředny vyšle zpráva "Power_Setting" s požadovanými parametry. Pokud chceme počítat energetickou bilanci, využijeme části vpravo dole, kde určujeme s jakou vzorkovací periodou bude počítána energetická bilance, z kolika vzorků chceme bilanci počítat a z jakých senzorů ústředny ji chceme počítat, případně vynulovat počítadlo energetické bilance daného senzoru. Po nastavení se k ústředně vyšle zpráva "Energy_Balance_Setting". Panel také zobrazuje informace o nastavení výpočtu a velikosti energetické bilance po příjmu zpráv "Energy_Balance_Status" a "Energy_Balance_Data". Panel má ještě funkci synchronizaci času v ústředně prostřednictvím zprávy "Master_Clock".



Obrázek 5.8: Nastavení power management ústředny

5.2.1.5 Obsluha Setting sensor



Obrázek 5.9: Nastavení senzoru ústředny

Pro nastavení parametrů měření jednotlivých senzorů je zde panel 5.9, který se zobrazí po stisku tlačítka "Setting_Sensor" v nabídce hlavního panelu. Panel nastavuje senzor, který je vybrán v části "Control"v hlavním panelu a umožňuje nastavit periodu měření, periodu průměrování, jakou veličinu chceme měřit a počet vzorků. Také dokáže zobrazit aktuální konfiguraci vybraného senzoru.

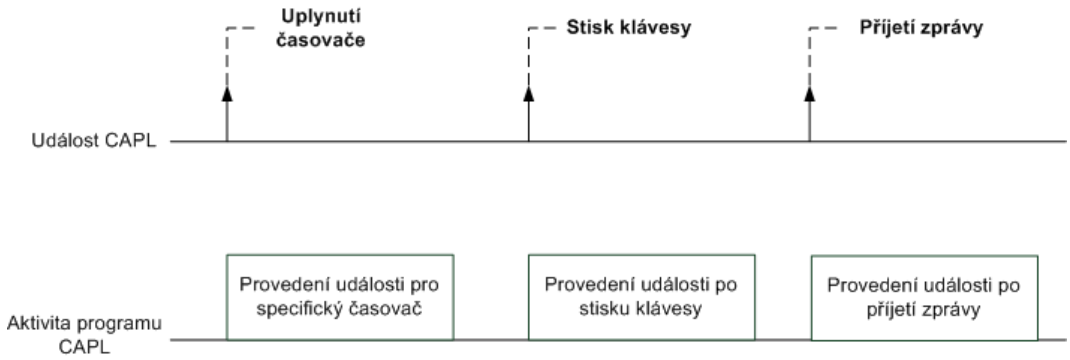
5.3 Programovací jazyk CAPL

Základ jazyka CAPL (CAN Access Programming Language) je velmi podobný programovacímu jazyku C. Programovací jazyk CAPL je jazyk využívající prostředky nástrojů CANoe, CANanalyzer a CANdb++. Spojením jazyka CAPL s prostředím CANoe a CANd++ je možné vytvářet uživatelské aplikace požadovaného chování, které jsou omezené pouze komunikací použitého harwaru nebo rychlostí počítače. CAPL vytváří v prostředí CANoe mnohem efektivnější a srozumitelnější představu o měření a analýze dat. Materiály pro tuto podkapitolu jsem čerpal z [8].

CAPL může být využit v těchto aplikacích:

- Analýza konkrétních CAN zpráv nebo dat
- Anylýza datové vytíženosti sběrnice
- Navržení uživatelského testeru
- Vytvoření simulačního modulu
- Vytvoření modulu pro diagnostiku nebo servisní prostředky
- vytvoření komplexních filtrů zpráv

a mnoho dalších funkcí pro usnadnění práce s CAN sběrnici.



Obrázek 5.10: Zpracování událostí

CAPL je procedurální jazyk řízený událostmi na časovače, změnu I/O (klávesnice, sériový port, paralelní port) nebo události přicházející po CAN sběrnici (zprávy, chyby).

Jedná se o obsloužení události příslušným kódem po vyvolání nějaké změny proměnné nebo příjmu specifické zprávy, případně hned po startu aplikace. Grafické zobrazení obsloužení události je vidět na obrázku 5.10. Nevýhoda jazyka CAPL je absence multitaskingu, tedy, že program nemůže obsloužit více událostí najednou a vykonává je postupně.

5.3.1 Hlavní rozdíly mezi jazykem CAPL a C

Je zde několik významných rozdílů mezi jazykem CAPL a C. Po podrobnějším prostudování [odkaz na lit] zjistíme, že CAPL je poněkud jednodušší než jazyk C a jazykem CAPL nejsou podporovány některé problematické části. V tabulce 5.2 jsou znázorněny některé návrhové vzory, které nejsou podporované jazykem CAPL.

Jelikož je CAPL kompletně řízen událostmi, není v tomto jazyce podporována metoda main(), stejně tak jako hlavičkové soubory nebo preprocesor. CAPL také nepodporuje deklaraci prototypu funkce předtím, než je použita. Také nepoužívá třídy a struktury. Funkce scanf() není podporována, protože CAPL nepodporuje příjem vstupního toku z klávesnice, ale je možné zpracovat vstupní řetězec pomocí "Environment variable" skrze uživatelský panel o němž je psáno v kapitole 5.2.

5.3.2 Datové typy

V tabulce 5.3 jsou definovány datové typy používané v jazyce CAPL. Message, timer a msTimer jsou také považovány za datový typ, protože definují proměnnou symbolizující datové typy, se kterými mohou být prováděny různé operace a mohou být ukládány pro následné použití. Typy bez určení znaménka pouze informují o tom, že nemohou nabývat negativní hodnoty. Naopak znaménkové typy mohou mít kladnou nebo zápornou hodnotu.

Návrhové vzory nepodporované v CAPL	**Rozdíl**	**Alternativní funkce**
main()	nepodporováno	nevyžadováno
hlavičkové soubory	nepodporováno	nevyžadováno
preprocesor	nepodporováno	nevyžadováno
definice makra	nepodporováno	nevyžadováno
vkládání souborů	nepodporováno	nevyžadováno
kompilační podmínky	nepodporováno	nevyžadováno
závislost počítačových platforem	nepodporováno	nevyžadováno
ukazatelé	nepodporováno	nevyžadováno
dynamické lokální proměnné	statické lokální proměnné	ano
void	nevyžadováno	ano
struktura	nepodporováno	nevyžadováno
sjednocení	nepodporováno	nevyžadováno
výčet	nepodporováno	nevyžadováno
define	nepodporováno	nevyžadováno
typedef	nepodporováno	nevyžadováno
sizeof	nepodporováno	nevyžadováno
automatic	nepodporováno	nevyžadováno
external	nepodporováno	nevyžadováno
registr	nepodporováno	nevyžadováno
standardní C knihovna	částečně	ano
string	nepodporováno	ano

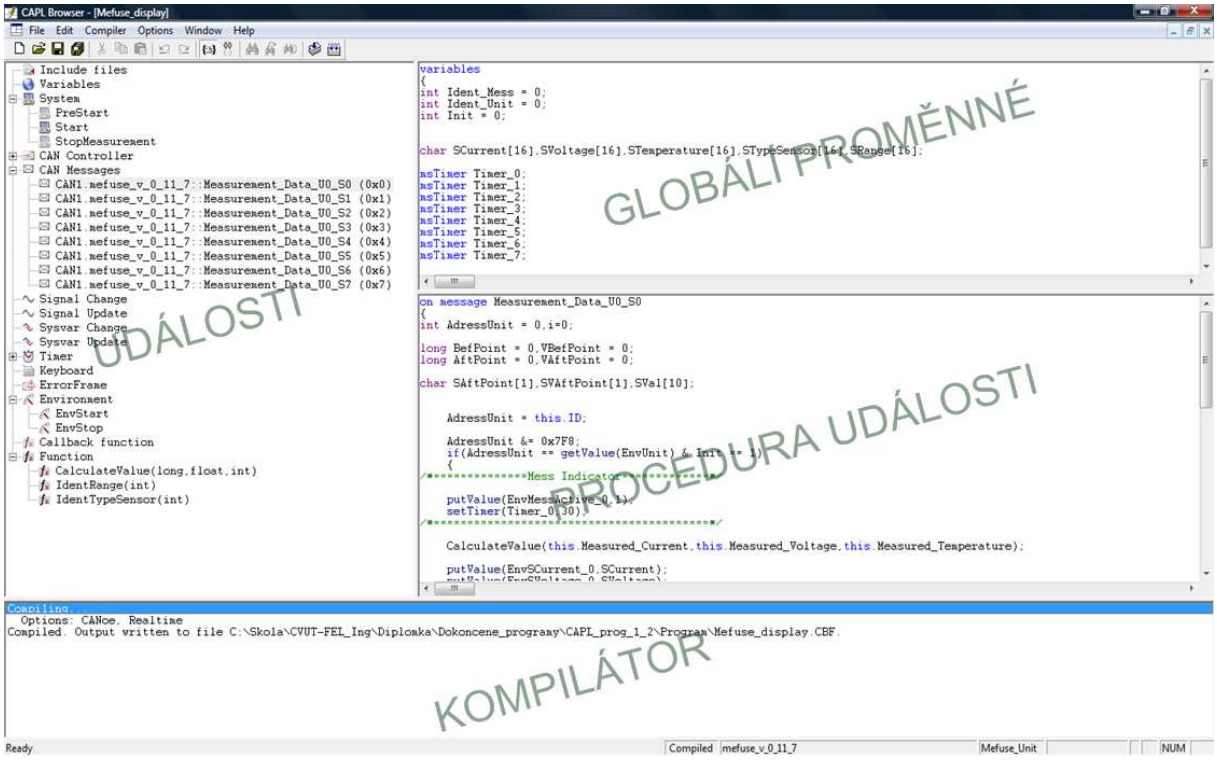
Tabulka 5.2: Hlavní rozdíly mezi jazyky CAPL a C

Datový typ	Popis	Velikost	Hodnota
char	znak	8 bit	neznaménkový
byte	byte	8 bit	neznaménkový
int	celočíselný	16 bit	znaménkový
word	slovo	16 bit	neznaménkový
long	celočíselný	32 bit	znaménkový
dword	slovo	32 bit	neznaménkový
float	plovoucí desetinná čárka	64 bit	znaménkový
double	plovoucí desetinná čárka	64 bit	znaménkový
message	CAN zpráva	—	—
timer	časovač s rozlišením 1s	—	—
msTimer	časovač s rozlišením 1ms	—	—

Tabulka 5.3: Datové typy v jazyce CAPL

5.3.3 Vývojové prostředí CAPL Browser

CAPL Browser v sobě spojuje textový editor a kompiler pro vývoj kódu v programovacím jazyce CAPL. CAPL Browser poskytuje vývoj nového kódu pro danou událost, modifikaci obsluhy stávající události, strukturu všech možných událostí a propojení s vytvořenou databází v CANdb++. Na obrázku 5.11 vidíme základní uspořádání CAPL Browseru. V levém panelu jsou poskytnuty informace o událostech před startem aplikace nebo po ukončení aplikace, události, které obsluhují jednotlivé zprávy definované v CAN databázi nebo funkce vytvořené v prostředí CAPL Browser.
V pravé horní části je panel s deklaracemi všech globálních proměnných potřebných ke správné funkci celé aplikace.
V pravé dolní části je zobrazen funkční kód pro obsloužení dané události vybrané v levé části CAPL Browseru.
Dolní část panelu je kompilátor, který informuje programátora o výskytu chyb v kódu nebo o jeho úspěšném přeložení.



Obrázek 5.11: Prostředí CAPL Browser

5.3.4 Využití CAPL pro vizualizační software

Programovací jazyk CAPL společně s vývojovým prostředím CAPL Browser dávají funkcionalitu celému vizualizačnímu systému. Asociace vytvořené CAN databáze se zprávami a "Environment variables" nám spojuje obsluhu všech příchozích definovaných zpráv a po vykonání příslušného kódu následuje případná změna grafických prvků v jednotlivých panelech vizualizačního systému. Stejně tak i po změně grafických prvků uživatelem daný kód generuje CAN zprávu s definovaným obsahem, která je vyslána po sběrnici do příslušné ústředny a s nastavenými parametry ve zprávě ovlivňuje režim samotné ústředny.

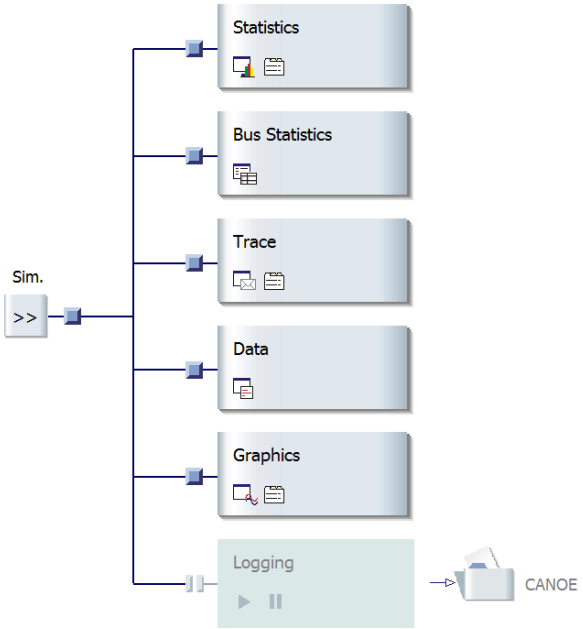
5.4 Software CANoe

Software CANoe je univerzální vývojový prostředek pro testování, simulaci a analýzu dat nejenom na CAN sběrnici, ale podporuje i sběrnice nebo protokoly LIN, MOST, FlexRay, J1939, NMEA2000 a J1587 . Převážně se software používá na analýzu CAN sběrnic v automobilovém průmyslu a tento software je poskytován firmou Vektor Informatik GmbH. Vývojový proces je založený na fázích modelování, které rozlišují tři stádia vývoje

- požadavky analýzy a návrh distribuovaných systému - funkční modelování
- implementaci a simulace sběrnice
- integrace celkového systému

5.4.1 Okno pro nastavení měření

Okno pro nastavení měření ukazuje strukturu procesu a informační tok aktuální konfigurace systému ve formě jednotlivých funkčních bloků, které jsou navzájem propojené. Každý blok reprezentuje jeden z hlavních prostředků softwaru CANoe. Většina těchto prostředků je reprezentována jako samostatné okno.[10]



Obrázek 5.12: Okno nastavení měření

5.4.2 Okno záznamu

Záznamové okno ukazuje veškeré zprávy detekované na sběrnici a k nim doplňující informace. Každá zpráva je zaznamenána s časovým okamžikem její detekce, identifikátorem zprávy, typem zprávy (Rx,Tx), délkou zprávy a obsahem datových bytů. Detekované zprávy mohou být v okně zobrazovány jako seznam všech zpráv se stanoveným časem nebo pomocí filtru identifikátoru, kde jsou zobrazeny pouze zprávy s různým identifikátorem a mění se pouze čas detekce a obsah zpráv.[10]

5.4.3 Okno statistiky zpráv

Statistické okno ukazuje statistické informace o jednotlivých CAN zprávách a průměrný čas mezi vysíláním jednotlivé zprávy. Také zobrazuje histogram jednotlivých CAN zpráv, kde na ose X je identifikátor zprávy a na ose Y je četnost vysílaných zpráv.[10]

5.4.4 Okno statistiky sběrnice

Statistika sběrnice ukazuje informace o celkovém chodu sběrnic jako je maximální a průměrné vytížení sběrnic, celkový počet zpráv za dobu měření. Informuje také o stavu řadiče sběrnice, jestli je pasivní, aktivní, vypnutý nebo zapnutý.[10]

5.4.5 Výstupní okno

Výstupní okno je textový výstup, který informuje o hlavních operacích softwaru CANoe jako je např. začátek a konec měření, zda se jedná o simulaci nebo reálnou komunikaci na sběrnici nebo o kompatibilitě hardwaru. Výstupní okno také slouží jako výstupní konzole programovacího jazyka CAPL při použití funkce write().[10]

5.4.6 Okno simulace

Ukazuje aktuální nastavení jednotlivých vláken vytvořených objektů použitím nástrojů popsaných v předchozích kapitolách. Tyto objekty reprezentují funkční bloky vytvořené prostřednictvím programovacího jazyka CAPL. Ukazuje počet nakonfigurovaných sběrnic a použitou databázi CAN zpráv. V tomto okně je také možnost využití generátoru CAN zpráv pro simulaci vytvořených systémů.[10]

5.4.7 Využití CANoe pro vizualizační systém

Software CANoe je základem pro vytvoření celistvého vizualizačního systému. Prostřednictvím CANoe implementujeme do její konfigurace vytvořené grafické panely pro zobrazování dat a nastavování měřící ústředny, vytvořenou databázi CAN zpráv pro snadnější identifikaci a orientaci v datech jednotlivých zpráv a potřebnou funkcionalitu prostřed nictvím funkčních bloků vytvořených v programovacím jazyce CAPL. Výsledkem celého snažení je komplexní systém pro práci s měřící ústřednou a zobrazování měřených veličin uživateli.

5.5 Spojování a konverze dat ze souboru

Při logování CAN zpráv do souboru je vytvořen soubor "ddhhmmss.bin", do kterého jsou ukládány požadované CAN zprávy v binární podobě v daném formátu. V současné době je ústředna schopna vytvářet soubory o velikosti 32 kB a je potřeba, aby veškerá data z jednoho měření byla v jednom souboru. Proto je dalším úkolem vytvořit skript, který dané soubory bude spojovat do jednoho. Jedním z hlavních požadavků této práce je také to, aby data v binární podobě byla převedena do formátu korespondující s požadovaným formátem softwaru CANoe. Spojená data tedy budou převedena do formátu "*.asc" se správnou strukturou. Soubor ve formátu "*.asc" je důležitý pro zpětnou informaci a analýzu logovaných zpráv v programu CANoe a pro následné použití v softwaru grafického zobrazení naměřených dat. Skripty pro sjednocení binárních souborů a konverze dat jsou napsány v programovacím jazyce C.

5.5.1 Spojování souborů

Funkce skriptu je velice jednoduchá. Při jeho spuštění se objeví konzole a vyzve uživatele k zadání adresáře se soubory "ddhhmmss.bin". Pokud tento adresář existuje následně je vytvořen seznam všech souborů obsažených adresáři. Dále je uživatel vyzván k zadání jména nově vytvořeného souboru s příponou "*.bin". Po vytvoření daného souboru jsou ze souborů v adresáři přesouvána veškerá data v pořadí zapsaném v seznamu všech dostupných souborů. Výstup skriptu je soubor "jmeno_souboru.bin", ve kterém jsou sjednocena všechna data ze souborů obsažených v příslušném adresáři. Pro korektní fungování skriptu jsou ošetřeny výjimky typu neexistujícího adresáře, prázdného adresáře

nebo nesprávného souboru pro spojování.

5.5.2 Konverze dat

Aby mohla být logovaná data z ústředny použita v softwaru CANoe pro zpětnou analýzu a kontrolu správného ukládání požadovaných zpráv bylo nutné vytvořit další skript, který sjednocená data převede do požadovaného formátu "*.asc". Binární soubor je tvořen tak, že každý řádek souboru odpovídá jedné CAN zprávě a zaznamenanému času uložení. Každá CAN zpráva je uložena v 16 bytech specifikovaného formátu, který je patrný z tabulky 5.4.

byte	**bit**	**význam**	**hodnota**	**popis**
1.	D7 - D2	rok	00h - 3Fh	2000 - 2064
	D1 - D0	měsíc	01h - 0Ch	01 - leden ... 0Ch - prosinec
2.	D7 - D6		00h, 0Dh - 0Fh	rezerva
	D5 - D1	den	00h	rezerva
			01h - 1Fh	1. - 31. den v měsíci
	D0	hodina	00h - 17h	00 - 23 hodin
3.	D7-D4		18h - 1Fh	rezerva
	D3 - D0	minuta	00h - 3Bh	00 - 59 minut
4.	D7 - D6		3Ch - 3Fh	rezerva
	D5 - D0	vteřina	00h - 3Bh	00 - 59 vteřin
5.	D7	platnost dat	0	časový údaj není platný
			1	časový údaje je platný
	D6 - D0	setina vteřiny	00h - 63h	0.00 - 0.99 vteřin
			64h - 7Fh	rezerva
6.	D7 - D3	rezerva		rezerva
	D2-D0	ID CAN zprávy	00h-07h	Nejvyšší 3 bity ID (I10-I8)
7.	D7 - D0		00h-FFh	Spodních 8 bitů ID (I7-I0)
8. - 15.	D7 - D0	obsah zprávy	8 bytů	Datový obsah CAN zprávy
16.	D7 - D0	rezerva	00h-FFh	rezerva

Tabulka 5.4: Formát zpráv v binárním souboru

Po spuštění skriptu je z konzole uživatel vyzván k zadání jména souboru, který má být převeden. Dále pak požádá o vytvoření nového souboru "jmeno_souboru.asc", do kterého jsou překonvertována data z binárního souboru. Skript provádí dekódování jednotlivých bytů zprávy a vytváří soubor v následujícím formátu:

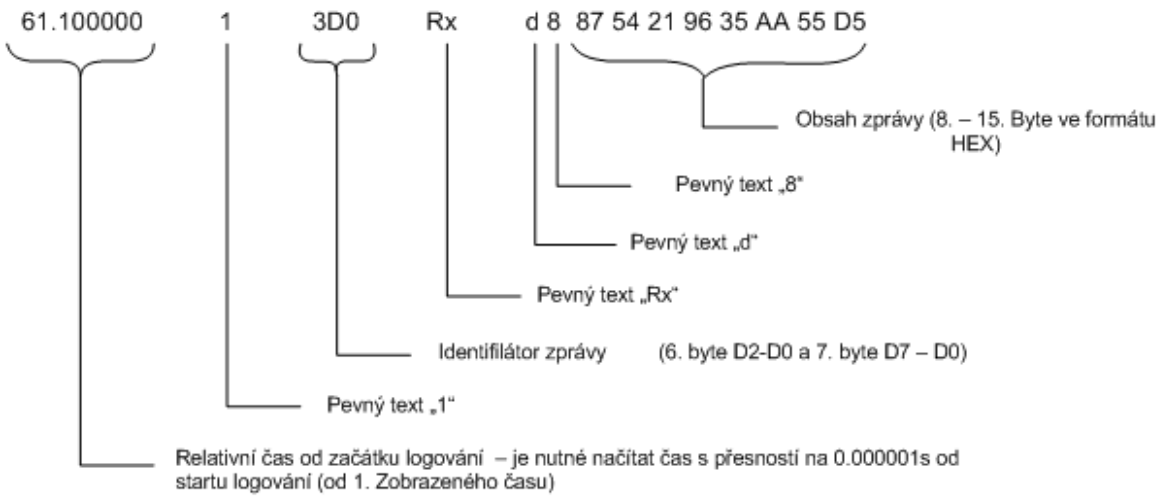
base hex timestamps absolute

```
06.05.2009 12:45
2.650000        1   C2    Rx   d 8 F0 21 85 F6 A5 26 BD 99
30.050000       1   7FB   Rx   d 8 12 34 56 78 96 52 48 AF

06.05.2009 12:46
61.100000       1   3D0   Rx   d 8 87 54 21 96 35 AA 55 D5

06.05.2009 12:50
253.400000      1   3D6   Rx   d 8 87 55 FC 96 35 AA 55 D5
261.950000      1   3D1   Rx   d 8 87 54 21 90 85 AA 5A DF
```

Černý text označuje neměnný text a červený text označuje proměnný text. Popis formátu zobrazení obsahu jednotlivých CAN zpráv je na obrázku 5.13.



Obrázek 5.13: Formát loggovaných zpráv

Kapitola 6

Vývoj software pro grafické zobrazení naměřených dat

Hlavním smyslem softwaru pro grafické zobrazování dat je ten, že poskytuje veškerou informaci o měřených veličinách v průběhu celého měření. Pomocí softwaru jsme schopni analyzovat průběh celého měření a následně po této analýze vyvodit patřičná opatření nebo zlepšení při výskytu jakéhokoli problému v elektrické sítí automobilu. Software zobrazuje časový průběh měřených veličin a poskytuje detailní informaci o signálech z hlediska jejich vývoje v čase. Počet zobrazených signálů je limitován počtem měřících ústředen na CAN sběrnici a počtem připojených senzorů k ústředně. Počet senzorů, které mají být zobrazeny v grafu je pouze na uvážení uživatele a jeho potřeby prověření pouze daných signálů. Celý návrh zobrazení naměřených dat je implementováno v programovacím jazyce C#. Hlavní konstrukce kódu jsem čerpal z [7].

6.1 Návrh grafického vzhledu

6.1.1 Hlavní zobrazovací okno

První představa o grafickém zobrazení naměřených dat byla taková, že v hlavním okně aplikace bude jeden graf, ve kterém budou zobrazeny všechny tři měřené veličiny. Osa X měla jednotné měřítko, které ukazovalo vývoj veličin v čase. Osa Y se skládala ze tří os. Každá z těchto os zobrazovala měřenou veličinu v daných jednotkách. Součástí grafu také byli funkce posuvu grafu v obou osách a "zoom" grafu pomocí horizontálního

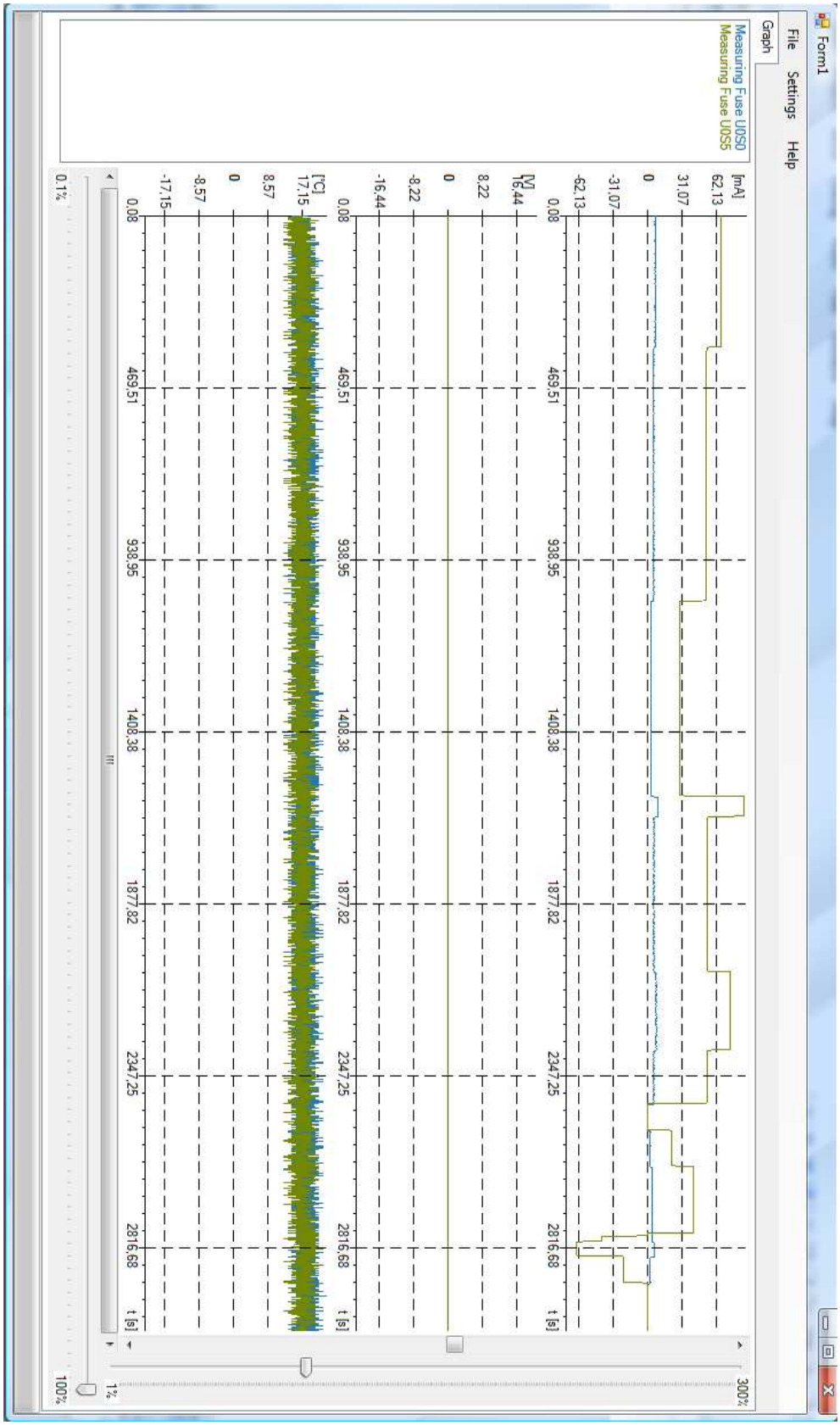
nebo vertikálního posuvníku nebo pomocí kurzoru myši, který svým posunem vymezoval oblast přiblížení.

Tento první návrh grafického zobrazení se neosvědčil jako dostatečně přehledný, jelikož při zobrazení měřených veličin s velkým rozsahem hodnot jedné z nich se ztráceli detaily ostatních veličin a na první pohled nebyli vidět např. krátké špičky, malý nárůst nebo pokles zobrazené veličiny. Dalším problémem grafu byla jeho nepřehlednost při zobrazení dat z více senzoru a špatná orientace mezi jednotlivými signály. Proto jsem tento grafický návrh zamítl, jelikož nesplňoval potřebné požadavky na podrobnou analýzu dat.

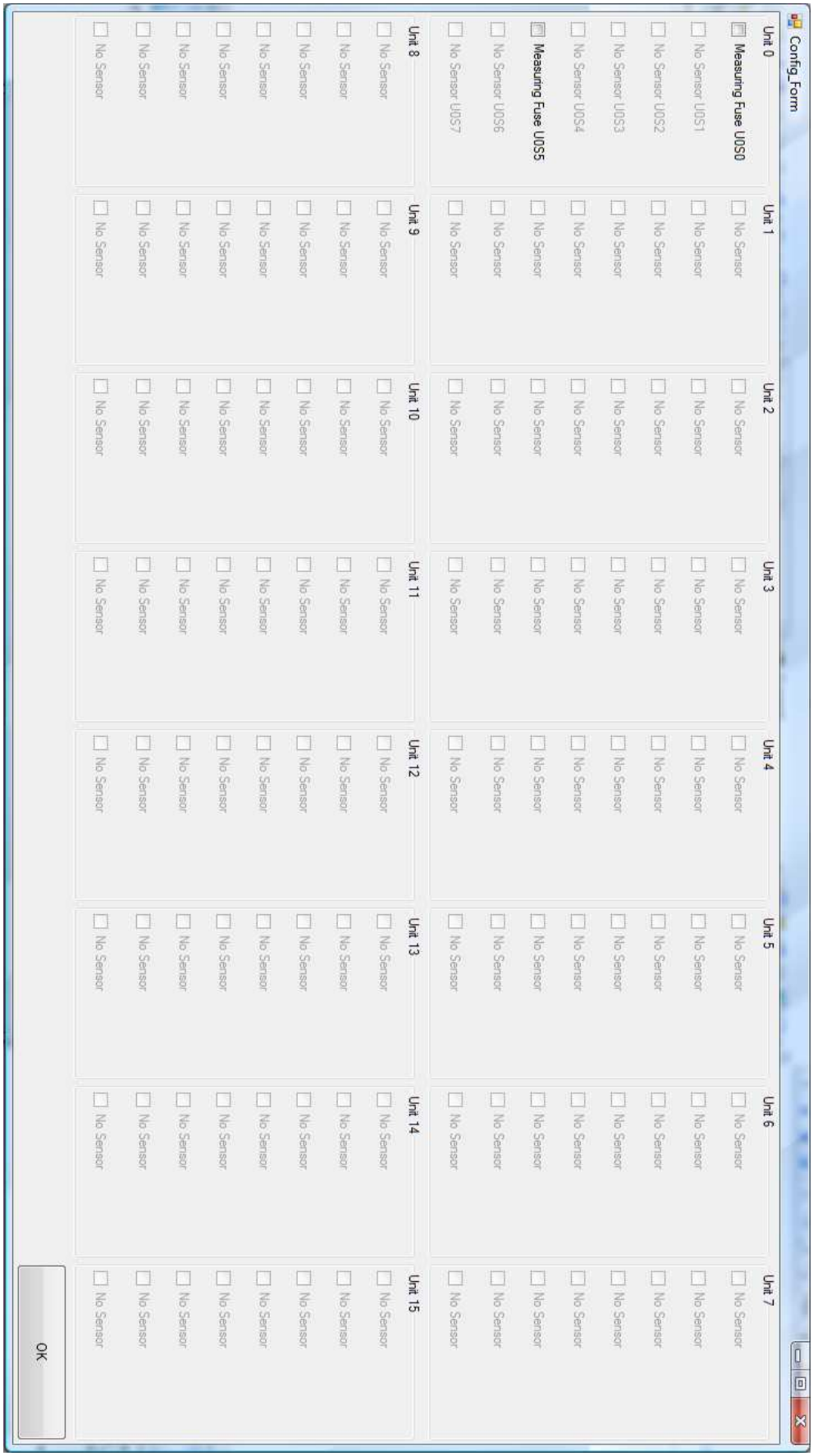
Další možností jak zobrazit měřená data byla separace jednotlivých veličin a každou zobrazit do jednoho grafu. Tím jsem odstranil nedostatek o ztrátě informace měřených veličin, když jedna z nich měla velký rozsah hodnot. Výsledné okno tedy mělo tři na sobě nezávislé grafy, které zobrazovali signály stejné veličiny z různých senzorů. Každý graf tedy měl svoje funkční prvky pro posuv časové osy a "zoom".

Ovšem při testování tohoto grafického zobrazení se ukázalo, že při použití nezávislých funkčních prvků časového posuvu a "zoom" u každého grafu byla ztracena časová synchronizace dat mezi jednotlivými veličinami. Takže když uživatel chtěl detailně zobrazit část průběhu jedné veličiny, např. proudový skok, ztrácel informaci o přesném časovém průběhu ostatních veličin a nemohl určit, zda při změně proudu nenastala i změna teploty nebo napětí.

Finální podoba hlavního grafického zobrazovacího okna má tři grafy, kde každý graf je určen pro jednu veličinu a jsou na sobě nezávislé. Funkční prvky jako je časový posun v ose X a "zoom" jsou pro všechny grafy stejné. Tedy při použití těchto prvků se změny provedou ve všech třech grafech, aby nedošlo ke ztrátě časové synchronizace a při použití "zoom" je poměr zvětšení nebo zmenšení ve všech grafech stejný. Do hlavního okna také byla přidána legenda pro rozpoznání signálů z více senzorů. Finální verze vzhledu hlavního grafického okna je vidět na obrázku 6.1.



Obrázek 6.1: Hlavní zobrazovací okno



Obrázek 6.2: Konfigurační okno senzorů

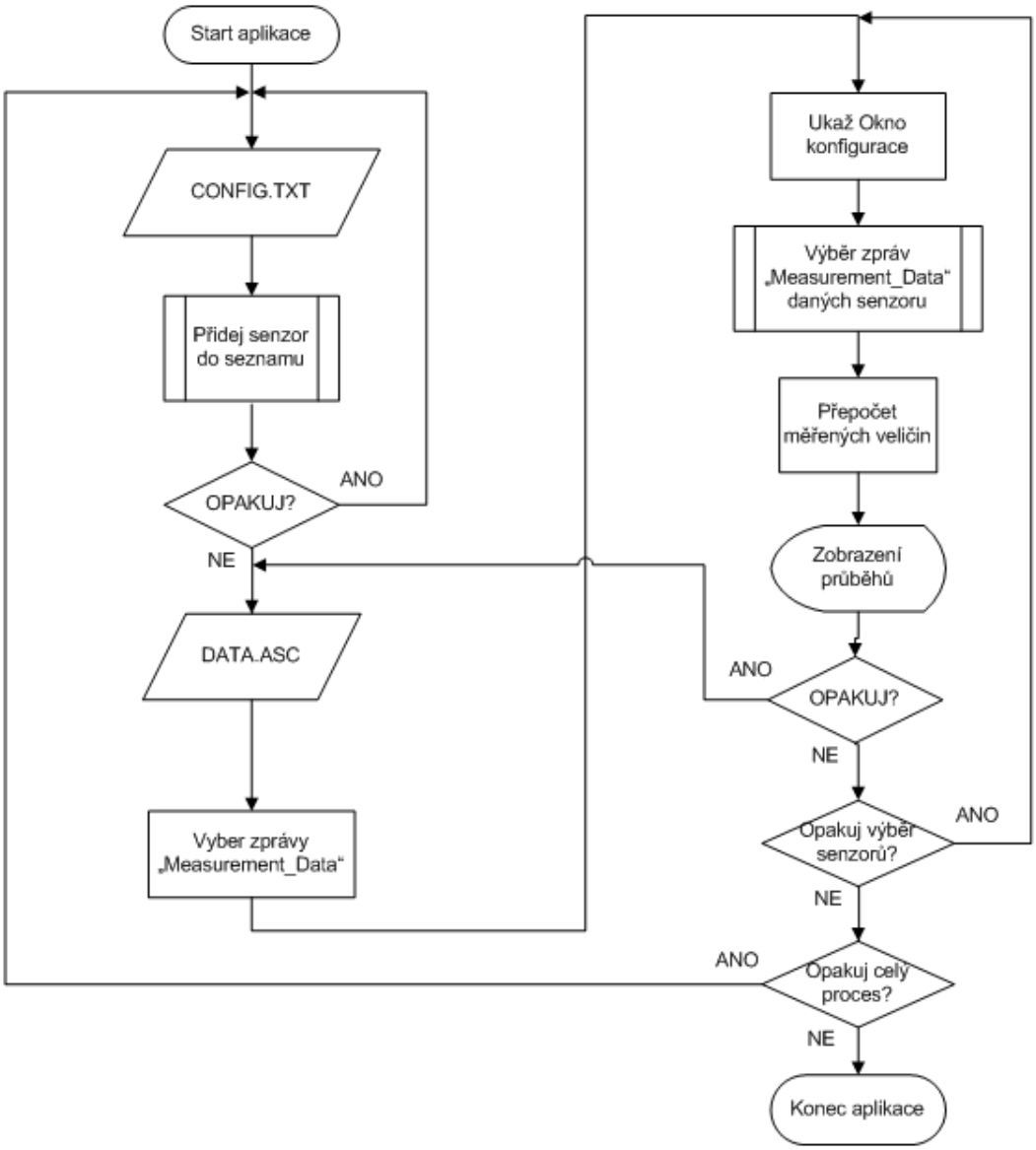
6.1.2 Konfigurační okno

Pro nastavování senzorů, které chce uživatel zobrazit jsem vytvořil konfigurační okno. Toto okno se zobrazí po načtení konfiguračního souboru a po kliknutí na položku *Settings* → *Config sensors* v hlavním menu aplikace. Po vyvolání okna se zobrazí senzory, které jsou vyčteny z konfiguračního souboru. Pokud nejsou obsazeny všechny sloty ústředen, zobrazí se na neobsazené pozici zašedivělý nápis "No Sensor". Hierarchie uspořádání senzorů je vidět na obrázku 6.2 , kde senzory jsou rozděleny do panelů podle ústředny, ke které jsou aktuálně připojeny. Jednotlivé senzory pro zobrazení jejich signálů se vybírají zaškrtnutím příslušného "checkboxu".

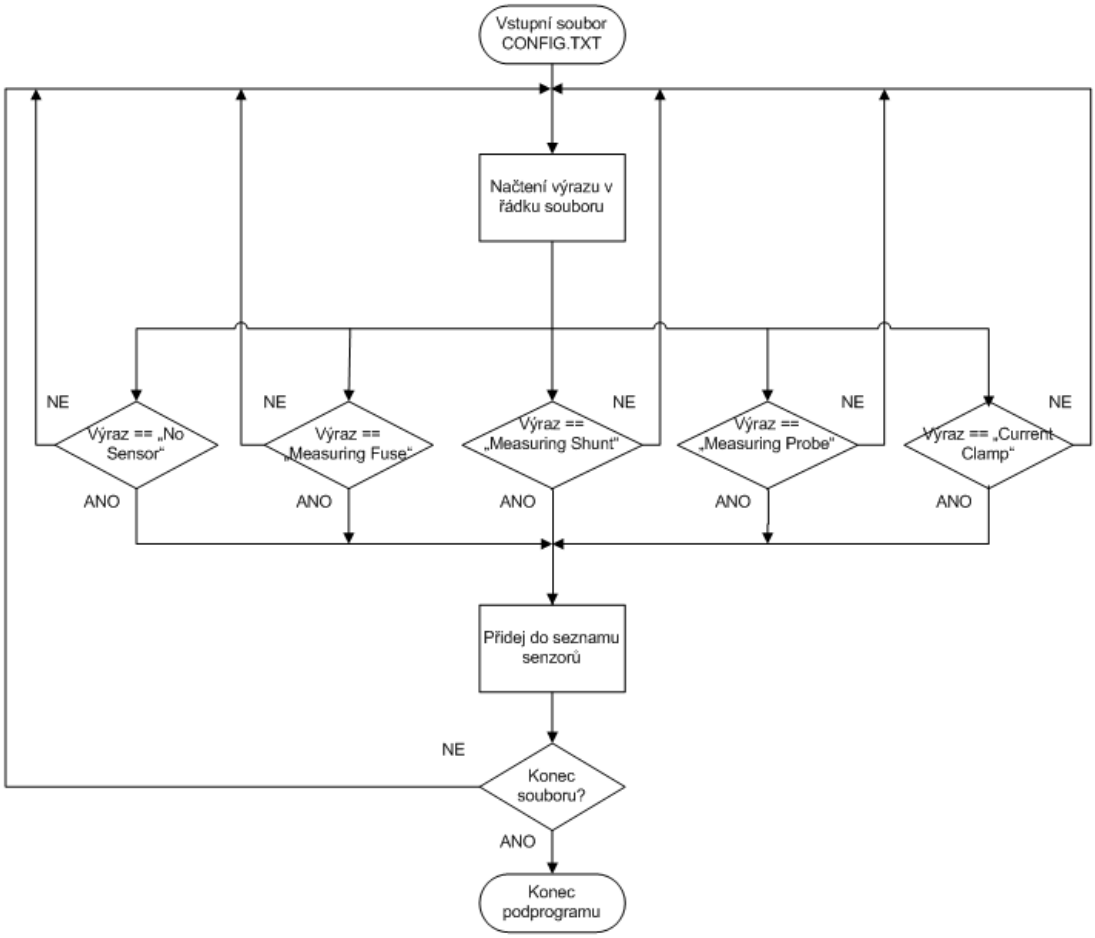
6.2 Implementace

Pro implementaci softwaru pro grafické zobrazení naměřených dat jsem zvolil programovací jazyk C# v prostředí Visual Studio 2008. Principiální vývojový diagram je na obrázku 6.3, dále pak vývojový diagram podprogramu pro zpracování konfiguračního souboru CONFIG.TXT 6.4 a podprogram pro zpracování naměřených dat DATA.ASC 6.5. Projekt se skládá z hlavního formuláře, konfiguračního formuláře a třídy MeasData.

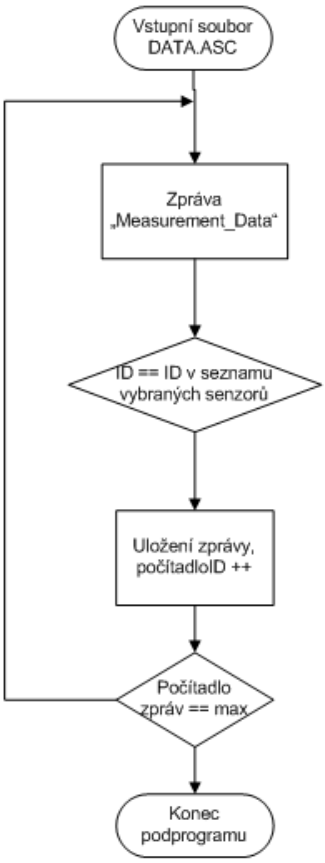
Hlavní a konfigurační formuláře obsluhují grafické prvky v celé aplikaci. Je to obsluha událostí veškerých funkčních grafických prvků jako je ovládání hlavního menu v horní liště hlavního formuláře, obsluha události při zaškrtnutí senzoru v konfiguračním formuláři nebo samotné vykreslování signálů ze senzorů na grafický panel reprezentující graf měřené veličiny. Třídu MeasData podrobně rozeberu v následující podkapitole. Předchozí popis rozdělení struktury aplikace slouží pouze pro její přehlednost a oddělení práce s grafickými prvky od zpracování naměřených dat a konfiguračního souboru.



Obrázek 6.3: Vývojový diagram aplikace



Obrázek 6.4: Vývojový diagram zpracování CONFIG.TXT



Obrázek 6.5: Vývojový diagram zpracování DATA.ASC

6.2.1 Třída MeasData

Třídu MeasData si rozebereme trochu více do hloubky, jelikož je to hlavní třída, která zajišťuje filtraci a zpracování naměřených dat. V každém formuláři je vytvořený objekt této třídy, jehož prostřednictvím se předávají informace pro zpracování celého procesu zobrazení. Pro detekování senzorů, které byly použity při měření nám poslouží konfigurační soubor CONFIG.TXT, který generuje ústředna při každém logování. Z tohoto souboru jsou vyčteny všechny senzory připojené k ústředně. Pro zpracování naměřených dat je využit zkonvertovaný soubor z binárních dat, o kterém jsem se zmínil v kapitole 5.5.2. Správný postup pro korektní fungování aplikace je nejprve načtení konfiguračního souboru, dále pak souboru s naměřenými daty a výběr senzorů pro zobrazení jejich průběhů.

Pomocí metody *ConfigData(string ConfigName)* filtrujeme z konfiguračního souboru seznam všech typů senzorů připojených k ústředně, i včetně informace, že na daném slotu není žádný senzor. Filtrace je prováděna pomocí regulárních výrazů s názvem daného typu senzoru. Pokud se shoduje konkrétní řádek v konfiguračním souboru s názvem senzoru je tento název přidán do seznamu senzorů. Při otevření konfiguračního okna je tento seznam automaticky načten a k položce odpovídající pozici senzoru v ústředně je přiřazen text popisující druh senzoru a slot, do kterého je připojen k ústředně.

Po úspěšném načtení konfiguračního senzoru musíme zvolit cestu k souboru naměřenými daty. Při této události se volá metoda *MeasurementData(string NameData)*. Tato metoda ze souboru s daty filtruje pouze zprávy "Measurement_Data"pomocí identifikátorů jednotlivých zpráv od daných senzorů a tyto zprávy jsou ukládány do seznamu zpráv "ListMessages". Filtrace je zavedena z důvodu možnosti logování i jiných zpráv z ústředny, které pro zobrazení měřených veličin nepotřebujeme a tím by se zbytečně komplikovalo další zpracování dat.

Při úspěšném načtení naměřených dat a po zobrazení konfiguračního okna se zvýrazní ty senzory, které byly načteny z konfiguračního souboru. Při zaškrtnutí libovolného aktivního senzoru je do seznamu "ListActualPaintSensor" přidáno ID senzoru. Toto ID je pevně nastaveno v každém "checkboxu" a identifikuje senzory, které chceme aktuálně vykreslit. Při stisku tlačítka "OK" je volána metoda *ChooseActualData()*.

Metoda *ChooseActualData()* zpracovává pouze ty zprávy odpovídající ID vybraného senzoru. Tato metoda si vytvoří jednorozměrné pole o velikosti odpovídající počtu senzorů pro zobrazení a do každého prvku pole vkládá další dvourozměrné pole se čtyřmi řádky, které reprezentují měřené veličiny čas, napětí, proud a teplota. Délka pole je pak závislá na počtu zpráv "Measurement_Data" pro daný senzor. Jelikož jednotlivé zprávy jsou uloženy v textovém řetězci, je nutné pro aplikování matematických operací řetězec převést do hexadecimálního formátu a pomocí funkce *CountingCurrent(int Value), CountingVoltage(int Value) a CountingTemperature(int Value)* vypočítat přesnou hodnotu měřené veličiny.

Funkce *CountingCurrent(int Value)* počítá hodnotu měřeného proudu podle vzorce

$$I_S = (V_I - K_I) \cdot M_I, [mA] \tag{6.1}$$

kde I_S je vypočtená hodnota proudu, V_I je hodnota získaná z CAN zprávy o maximální velikosti třech bytů, K_I = 800000 hex a M_I = 0,1 je multiplikátor pro přepočet na jednotky mA. Maximální rozsah měřeného proudu je -838.8607 A až 838.8607 A s

maximálním rozlišením $100\mu A$.

Funkce *CountingVoltage(int Value)* slouží k výpočtu měřeného napětí podle vzorce

$$U_s = V_U \cdot M_U, [V] \tag{6.2}$$

kde U_S je vypočtená hodnota napětí,V_U je hodnota získaná z CAN zprávy o maximální velikosti jednoho bytu, $M_U = 0{,}1$ je multiplikátor pro přepočet na jednotkz V. Maximální rozsah měřeného napětí je 0 V až 24 V s maximálním rozlišením 0.1V.

Pro výpočet měřené hodnoty teploty použijeme funkci textitCountingTemperature(int Value), která provádí výpočet podle vzorce

$$T_S = V_T - K_T, [°C] \tag{6.3}$$

kde T_S je vypočtená hodnota teploty, V_T je hodnota získaná z CAN zprávy o maximální velikosti jednoho bytu, $K_T = 80$ hex. Maximální rozsah měřené teploty je -40 °C až 111 °C s maximálním rozlišením 1°C.

Poslední metodou ve třídě MeasData je metoda *Statistic()*, která zjišťuje minimum a maximum měřených veličin ze všech zobrazených senzorů. Je to z důvodů zjištění maximálních rozsahů měřených veličin pro správné nastavení měřítka v grafu.

6.2.2 Funkce podporované softwarem pro grafické zobrazení dat

Hlavní funkcí grafického softwaru je zobrazení časového průběhu naměřených veličin. V hlavním okně jsou funkční prvky, které umožňují časový posun ve vodorovné ose, vertikální posun v ekvivalentních jednotkách odpovídající měřené veličině, použití "zoom" jak v horizontální, tak i ve vertikální ose. "Zoom" je také možné použít prostřednictvím myši tím, že v kterémkoliv grafu zmáčkneme levé tlačítko myši a tahem označíme oblast, kterou chceme zvětšit. Po uvolnění levého tlačítka se označená část přizpůsobí maximální velikosti okna grafu. Po stisku pravého tlačítka myši se zobrazený průběh zmenšuje až do zobrazení celého naměřeného průběhu.

V hlavním menu je položka *Settings*. Tato položka poskytuje načítání konfiguračního souboru, zobrazuje okno s nakonfigurovanými senzory a zapíná nebo vypíná hlavní a vedlejší mřížku grafů. Položka *File* v hlavním menu načítá soubor s daty a ukončuje aplikaci.

Postup pro úspěšné zobrazení měřených veličin je následující. Nejprve v záložce *Settings* → *Open* načteme konfigurační soubor. Následně v záložce *File* → *Open* načteme soubor s daty. Opět otevřeme záložku *Settings* → *Config Sensors* a zaškrtneme senzory, které chceme zobrazit. Po zmáčknutí tlačítka "OK" se zobrazí průběhy ze senzorů a dále při použití funkčních prvků upravujeme vzhled zobrazovaných dat.

Kapitola 7

Sběr dat z reálného automobilu

Na závěr je třeba otestovat systém měření klidových proudů na reálném vozidle. Celý proces testu bude obsahovat komunikaci s ústřednou prostřednictvím vizualizačního softwaru v prostředí CANoe. Jde o přípravu pro správné nastavení ústředny, vzorkovacích frekvencí měřících senzoru a nastavení logování měřených dat na paměťovou SD kartu ústředny. Po ukončení měření přistoupíme k sjednocení vytvořených binárních souborů a konverzi těchto dat do požadovaného formátu pro software CANoe a grafický software. Celý test se skládá ze dvou měření. V prvním případě se jedná o krátkodobé měření odběru proudu výstražných světel automobilu. V druhém případě se jedná o měření odběru proudu při změně stavu automobilu z provozního režimu do režimu pohotovostního a jak velký podíl na velikosti odebíraného proudu má navigační systém. Jedná se o ověření teoretické části o klidových proudech v kapitole 2.1.

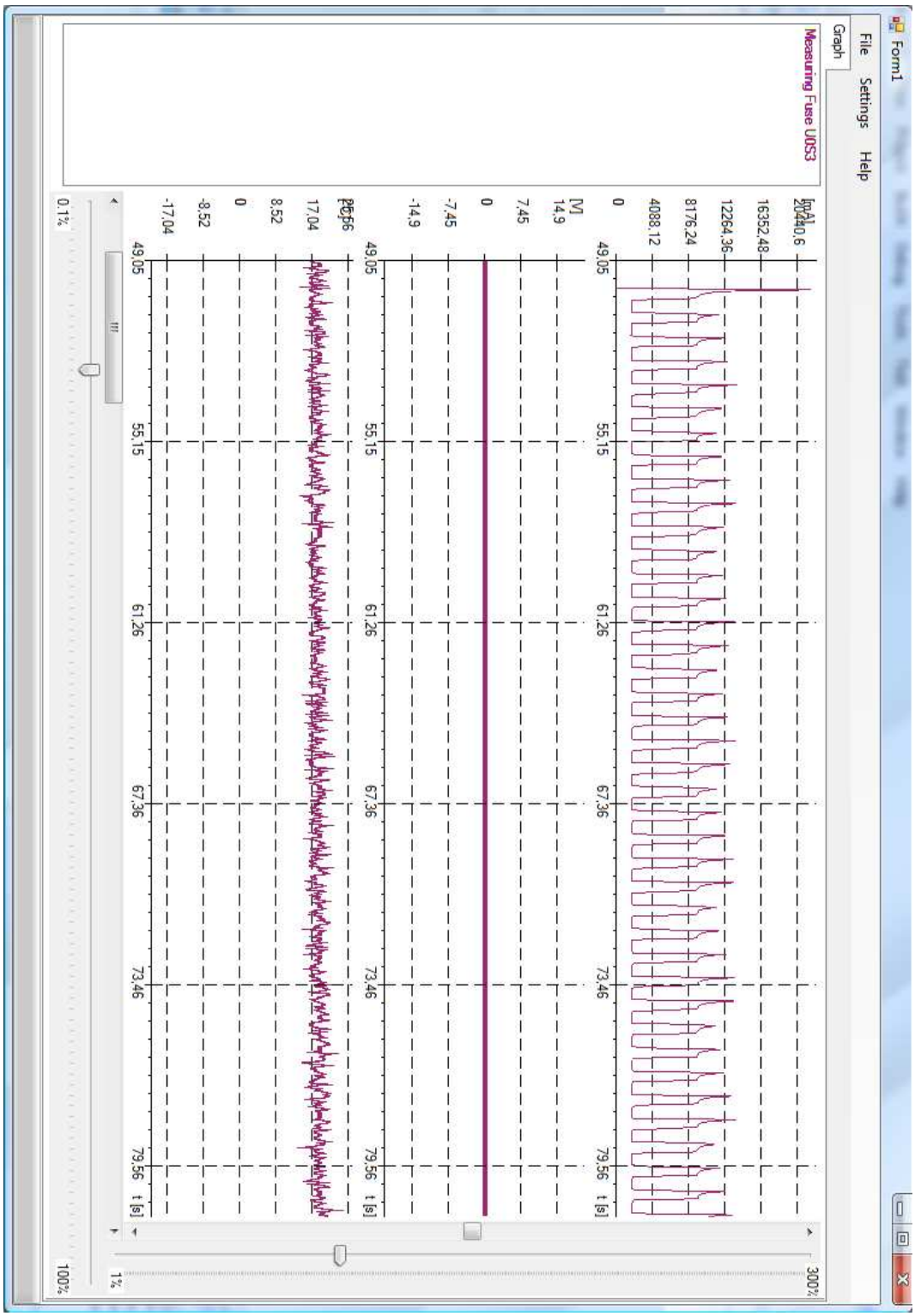
Pro účely testování byl k dispozici automobil Škoda Yeti 2.0 TDI výbavy Experience. Obsahem této výbavy je např. Klimatronic, elektronický imobilizér, multifunkční volant, rozhraní pro mobilní telefon, MAXI DOT (informační panel řidiče) nebo navigační systém.

7.1 Odběr proudu varovnými signalizačními světly

Cílem měření je zjištění velikosti celkového odebíraného proudu při spuštění varovných signalizačních světel, když je automobil v pohotovostním režimu. Zřejmý předpoklad je takový, že světla periodicky blikají a velikost odebíraného proudu bude úměrná frekvenci blikání. Pomocí vizualizačního softwaru jsme nastavili periodu měření na 50 ms a měřená

data se ukládají na paměťovou SD kartu. Z naměřených dat je vidět průběh na obrázku 7.1. Doba trvání celého měření je zhruba 4 minuty.

Z průběhu proudu na obrázku 7.1 vidíme, že při zapnutí výstražné signalizace se hodnota proudu dostala až na 20 A po dobu zhruba 0,1 s. To je způsobeno probuzením a inicializací některých řídicích jednotek. V průběhu blikání signalizačních světel si také všimneme, že ve chvíli, kdy signalizační světla nesvítí není celková odběr nulový, ale pohybuje se okolo hodnoty 1,5 A. To je opět způsobeno aktivními řídícími jednotkami, které mají na starosti službu signalizačních varovných světel.



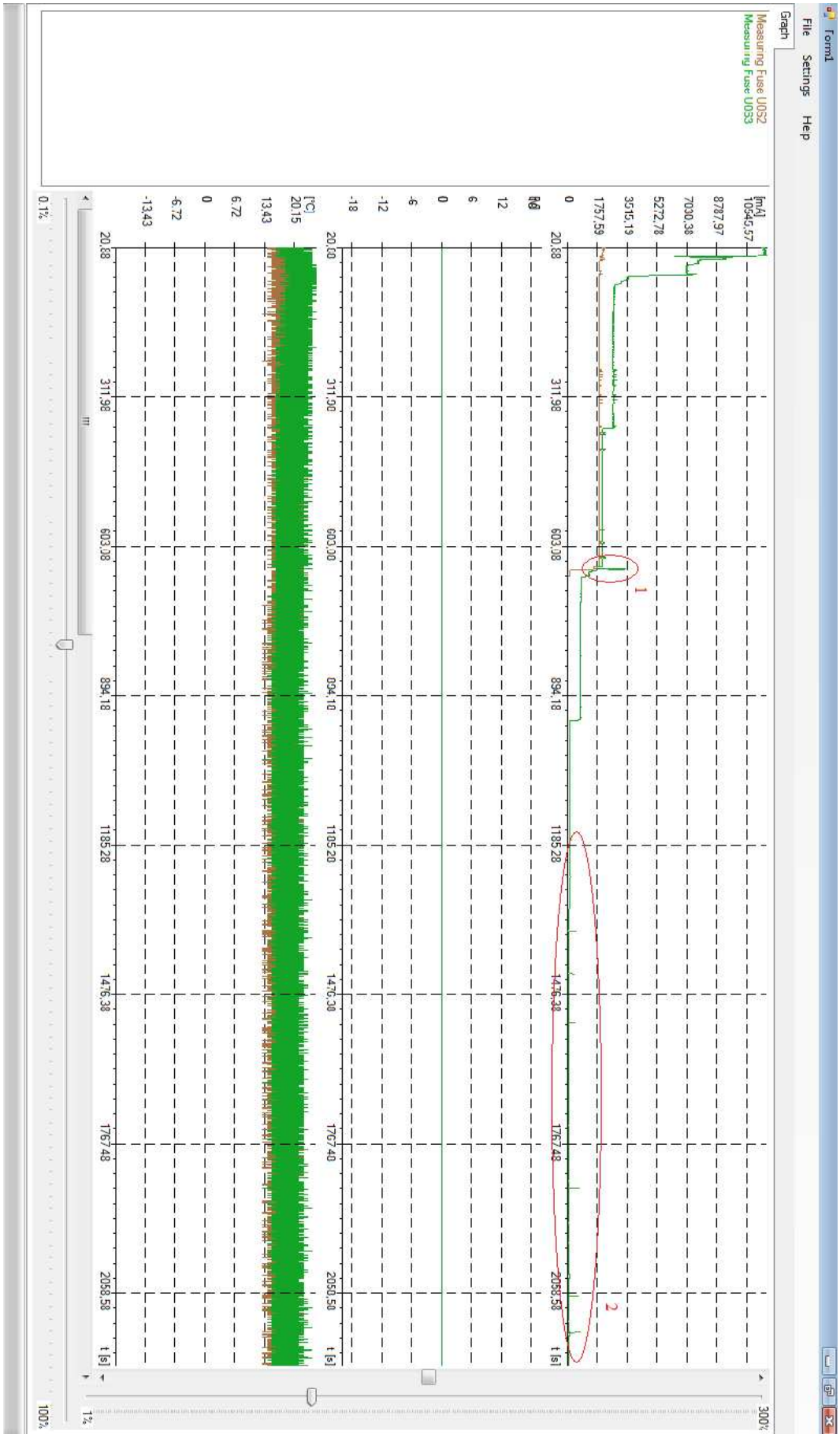
Obrázek 7.1: Průběh proudu varovných signalizačních světel

7.2 Přechod automobilu do pohotovostního stavu

V tomto případě jde o ověření teoretických poznatků z kapitoly 2.1. Opět pomocí vizualizačního softwaru nastavíme parametry ústředny. Parametry jsou shodné jako v předchozím měření s tím rozdílem, že nyní jsou použity dva senzory a druhý senzor měří pouze proud odebíraný navigačním systémem. Po spuštění měření jsme vyndali klič ze zapalovací skříňky a zavřeli dveře, nikoliv zamkli pomocí dálkového ovládání. Výsledný průběh měřeného proudu je vidět na obrázku 7.2.

Ze začátku průběhu měření celkového odběru, značeného zelenou barvou, je vidět zvýšený odběr proudu, což je způsobeno zastrčeným klíčkem v zapalovací skříňce a otevřenými dveřmi u řidiče. Po vyndání klíčku ze zapalování a zabouchnutí dveří se odběr snížil přibližně na 7 A. Následně zhruba po deseti minutách je vidět špička, v grafu označena jako bod 1. To je způsobené částečným probuzením některých řídicích jednotek a kontrolou automobilu před přechodem do pohotovostního režimu. Zhruba po šestnácti minutách je automobil plně v pohotovostním režimu. V označeném bodě 2 na obrázku jsou vidět malé proudové špičky způsobené probouzením řídicích jednotek. Pravidelné probouzení řídicích jednotek může být např. způsobeno systémem detekce deště, kdy při detekci deště jsou zabezpečeny okna vozidla. Po analýze praktických výsledků měření odběru proudu při přechodu automobilu z provozního do pohotovostního režimu můžeme konstatovat, že teoretické předpoklady se shodují s praktickými.

V grafu je také signál označený hnědou barvou. Tento signál ukazuje průběh odběru proudu navigačního systému. Z měření můžeme říci, že po odchodu z vozidla je navigační systém funkční ještě deset minut. Při konečném vypnutí navigačního systému je taky patrný pokles celkového odběru proudu.



Obrázek 7.2: přechod z provozního od pohotovostního režimu

Kapitola 8

Závěr

Z teoretických poznatků o klidových proudech je zřejmé, že tuto problematiku není radno brát na lehkou váhu a účel této práce je více proniknout do této problematiky a osvětlit chyby, které se podílí na vyšším odběru proudu elektronických částí v automobilu.

V první řadě jsem utvořil nadhled nad způsoby a principy měření elektrického proudu v elektrickém obvodu s jeho rozpojením nebo bez jeho rozpojení díky prostudování [11] a [1], s ohledem na prostorové nároky v palubní síti automobilu, dále pak s ohledem na digitalizaci měřených dat a jejich distribuci v digitální formě. Výsledkem tohoto návrhu je tedy princip měření elektrického proudu s rozpojeným obvodem za použití měřícího systému Mefuse a inteligentních senzorů.

Následující část zahrnovala vytvoření databáze CAN zpráv, které napomáhají k lepší identifikaci jednotlivých zpráv a informacím uvnitř zpráv. Celá databáze je vytvořena v softwaru CANdb++. Společně s CAN zprávami je v databázi soubor tzv. "Environment variables", které pomáhají k funkčnosti vizualizačního softwaru aktuálně měřených dat a nastavování režimů a parametrů měřící ústředny Mefuse a senzorů.

Pro vytvoření vizualizačního softwaru bylo potřeba sloučit několik prvků dohromady a tím je grafické rozhraní celého softwaru, funkčnost softwaru prostřednictvím programovacího jazyka CAPL a celé to implementovat do softwaru CANoe, který je základem pro fungování celého systému. Pomocí vizualizačního softwaru je uživatel schopen měřit elektrický proud, elektrické napětí a teplotu v reálném čase. Nastavovat parametry měření a režim měřící ústředny podle požadavků a možností měřícího systému.

Po vytvoření vizualizačního softwaru bylo potřeba naměřená data analyzovat. Jako základní analýza je přehled všech logovaných CAN zpráv v softwaru CANoe, jelikož uložená data jsou v binárním formátu bylo potřeba daná data převést do správného formátu prostřednictvím mnou vytvořeného skriptu v programovacím jazyce C pro soft-

ware CANoe. Pro další analýzu jsem vyvinul software grafického zobrazení dat. Software byl programován v jazyce C# v prostředí Visual Studio 2008. Mezi základní prvky grafického softwaru patří zobrazování signálů měřených veličin ze senzorů v závislosti na čase. Grafy veličin umožňují posuv v horizontální a vertikální ose, "zoom" pomocí grafických prvků nebo myši a případné vypínání nebo zapínání hlavní a vedlejší mřížky.

Sada těchto programů poskytuje způsob měření daných veličin a podrobnou informaci o průběhu veličin. Tento systém pro měření klidových proudů jistě ulehčí práci při odhalování chyb spjatých s palubní sítí automobilu. Jako demonstraci fungování všech programů jsem provedl měření na reálném vozidle, která jsou vidět v kapitole 7

Literatura

[1] Boháček, J.: *Metrologie elektrických veličin*. Praha: Vydavatelství ČVUT, 1994.

[2] Čermák, O.: *11VernetzungSK35 MJ07Octavia II*. Mladá Boleslav, 2006.

[3] Hanzálek, Z.: *Přednášky k předmětu X35DRS*. Praha, 2009.

[4] Jareš, T.: *CAN v automobilu*. Mladá Boleslav, 2000.

[5] Jens Luhmann, U. G.: *Electrical and Electronic Assemblies in Motor Vehicles*. Wolfsburg, 2009.

[6] Norbert Wolf, A. K.: *Energienetzrichtlinie 12V-BordnetzV1.1*. Wolfsburg, 2007.

[7] Sels, C.: *C# a WinForm - programování formulářů Windows*. Praha: Zoner software s.r.o, 2005.

[8] Vector: *Programming with CAPL*. Novi: Vector CANTech, 2004.

[9] Vector: *CANdb++ - User Manual*. Stuttgart: Vector Informatik GmbH, 2006.

[10] Vector: *CANoe Option - User Manual*. Stuttgart: Vector Informatik GmbH, 2007.

[11] Vladimír Haasz, M. S.: *Elektrická měření - Přístroje a metody*. Praha: Vydavatelství ČVUT, 2005.

[12] Vojáček, A.: *Integrované senzory proudu a problematika použití – 1.část*. URL `http://automatizace.hw.cz/integrovane-senzory-proudu-a-problematika-pouzit`

[13] Vojáček, A.: *Magnetické senzory s Hallovým efektem - 1. princip*. URL `http://automatizace.hw.cz/magneticke-senzory-s-hallovym-efektem-1-princip`

Příloha A

Protokol CAN zpráv

byte	bit	význam	hodnota	popis
1.	D7 - D0	typ modulu	01h	měřící ústředna
			00h	rezerva
			02h - 0FFh	rezerva
2.	D7 - D5	verze SW	0h - 7h	hlavní varianta
	D4 - D0		0h - 1Fh	vedlejší varianta
3.	D7 - D5	verze HW	0h - 7h	hlavní varianta
	D4 - D0		0h - 1Fh	vedlejší varianta
4.	D7 - D0	výrobní číslo	00h - 0FFh	vyšší byte
5.	D7 - D0		00h - 0FFh	nižší byte
6.	D7	součásti systému	1	logování na SD kartu
	D6		1	BlueTooth modul
	D5 - D0	rezerva		rezerva
7.	D7	chybový status	0	bez vnitřní chyby
			1	vnitřní chyba ústředny
	D6 - D0		00h - 7Fh	rezerva
8.	D7 - D0	rezerva	00h	rezerva

Tabulka A.1: Central_Indent

byte	bit	význam	hodnota	popis
1.	D7 - D5	typ modulu	0h	žádné čidlo není připojeno
			1h	měřící pojistka
			2h	měřící bočník
			3h	proudová sonda
			4h	napěťová sonda
			5h - 7h	rezerva
	D4 - D0	rozsah pojistky	0h	není relevantní
			1h	3A
			2h	5A
			3h	7.5A
			4h	10A
			5h	15A
			6h	20A
			7h	25A
			8h	30A
			9h	40A
			0Ah	50A
			0Bh	60A
			0Ch	70A
			0Dh	80A
			0Eh	100A
			0Fh	150A
			10h	200A
			11h - 1Fh	rezerva
2. - 4.		naměřený proud	000001h -7FFFFFh	změřený proud -838.8607A až -100uA s rozlišením 100uA, Proud = (hodnota - 800000h) x 100uA
			800000h – FFFFEFh	změřený proud 0A až 838.8591A s rozlišením 100uA, Proud = (hodnota - 800000h) x 100uA
			000000h	překročen rozsah měření v záporných hodnotách
			0FFFFFFh	překročen rozsah měření v kladných hodnotách
			0FFFFFEh	chyba měření proudu
			0FFFFFDh	proud neměřen
			0FFFFF0h – 0FFFFFCh	rezerva
5.		naměřené napětí	00h - 0F0h	naměřené napětí 0 – 24V s rozlišením 0.1V, Napětí = hodnota x 0.1V
			0FFh	překročen rozsah měření
			0FEh	chyba měření napětí
			0FDh	napětí neměřeno
			0F1h - 0FCh	rezerva
6.		naměřená teplota	01h – 07Fh	naměřená teplota -1 ÷ -40 °C s rozlišením 1°C, Teplota = (hodnota – 80h) x 1°C
			80h – EFhh	naměřená teplota 0°C až 111 °C s rozlišením 1°C, Teplota = (hodnota – 80h) x 1°C
			000h	překročen rozsah měření v záporných hodnotách
			0FFh	překročen rozsah měření v kladných hodnotách
			0FEh	chyba měření teploty
			0FDh	teplota neměřena
			0F0h – 0FCh	rezerva
7.	D7 - D0	chybový status čidel	00h - 0FFh	chybový status čidel
8.	D7	data pro BT	1	Tento bit bude nastaven vždy, když od předchozího paketu, který měl tento bit nastaven, uplynulo 0.5s (pro časy měření 50ms až ¡0.5s) nebo pro všechny pakety s periodou 0.5s a vyšší. Pouze zprávy s tímto nastaveným bitem jsou přeposílány na BT
	D6 - D0	rezerva	00h	rezerva

Tabulka A.2: Measurement_Data

byte	bit	význam	hodnota	popis
1.	D7 - D4	perioda měření	01h	50ms
			02h	100ms
			03h	250ms
			04h	500ms
			05h	1s
			06h	2s
			07h	5s
			08h	10s
			09h	20s
			0Ah	60s
			00h	ponechat naposledy nastavenou hodnotu
			0Bh - 0Fh	rezerva
	D3 - D0	průměrování	01h	50ms
			02h	100ms
			03h	250ms
			04h	500ms
			05h	1s
			06h	2s
			07h	5s
			08h	10s
			09h	20s
			0Ah	60s
			00h	ponechat naposledy nastavenou hodnotu
			0Bh - 0Fh	rezerva
2.	D7 - D6	měřená veličina	00h	ponechat naposledy nastavenou hodnotu
			01h	bez měření proudu
			02h	měření proudu
			03h	rezerva
	D5 - D4		00h	ponechat naposledy nastavenou hodnotu
			01h	bez měření teploty
			02h	měření teploty
			03h	rezerva
	D3 - D2		00h	ponechat naposledy nastavenou hodnotu
			01h	bez měření napětí
			02h	měření napětí
			03h	rezerva
	D1 - D0		01h - 03h	rezerva
3.	D7 - D0	počet odměrů	00h	ukončení měření
			01h - 0FEh	1 - 254 odměrů
			0FFh	trvalé měření
4. - 8.	D7 - D0	rezerva	00h	rezerva

Tabulka A.3: Measurement_Setting

byte	bit	význam	hodnota	popis
1.	D7 - D4	perioda měření	01h	50ms
			02h	100ms
			03h	250ms
			04h	500ms
			05h	1s
			06h	2s
			07h	5s
			08h	10s
			09h	20s
			0Ah	60s
			00h	rezerva
			0Bh - 0Fh	rezerva
	D3 - D0	průměrování	01h	50ms
			02h	100ms
			03h	250ms
			04h	500ms
			05h	1s
			06h	2s
			07h	5s
			08h	10s
			09h	20s
			0Ah	60s
			00h	rezerva
			0Bh - 0Fh	rezerva
2.	D7 - D6	měřená veličina	00h	rezerva
			01h	bez měření proudu
			02h	měření proudu
			03h	rezerva
	D5 - D4		00h	rezerva
			01h	bez měření teploty
			02h	měření teploty
			03h	rezerva
	D3 - D2		00h	rezerva
			01h	bez měření napětí
			02h	měření napětí
			03h	rezerva
	D1 - D0		01h - 03h	rezerva
3.	D7 - D0	počet odměrů	00h	ukončení měření
			01h - 0FEh	1 - 254 odměrů
			0FFh	trvalé měření
4. - 8.	D7 - D0	rezerva	00h	rezerva

Tabulka A.4: Measurement_Setting_Status

byte	bit	význam	hodnota	popis
1.	D7 - D5	typ modulu	0h	žádné čidlo není připojeno
			1h	měřící pojistka
			2h	proudový bočník
			3h	proudová sonda
			4h	napěťová sonda
			5h - 7h	rezerva
	D4 - D0	hodnota pojistky /rozsahu	0h	není relevantní
			1h	3A
			2h	5A
			3h	7.5A
			4h	10A
			5h	15A
			6h	20A
			7h	25A
			8h	30A
			9h	40A
			0Ah	50A
			0Bh	60A
			0Ch	70A
			0Dh	80A
			0Eh	100A
			0Fh	150A
			10h	200A
			11h - 1Fh	rezerva
2.	D7 - D5	verze SW	0h - 7h	hlavní varianta
	D4 - D0		0h - 1Fh	vedlejší varianta
3.	D7 - D5	verze HW	0h - 7h	hlavní varianta
	D4 - D0		0h - 1Fh	vedlejší varianta
4.-5.	D7 - D0	výrobní číslo	00h - 0FFh	vyšší byte
	D7 - D0		00h - 0FFh	nižší byte
6.	D7 - D0	stav LED modulu	01h	LED nesvítí
			02h	LED trvale svítí
			03h	LED bliká (50ms svítí, 950ms nesvítí)
			04h	LED bliká (50ms svítí, 50ms nesvítí)
			00h	LED nesvítí
			05h - FFh	rezerva
7.	D7 - D0	chybový status čidel	00h - 0FFh	chybový status čidel
8.	D7 - D0	rezerva	00h	rezerva

Tabulka A.5: Sensor_Ident

byte	bit	význam	hodnota	popis	následující operace
1.	D7	požadavek identifikace senzorů	1	senzor č. 8	Ústředna jednorázově odešle příslušný počet zpráv "Sensor_Ident" dle nastavených bitů
	D6		1	senzor č. 7	
	D5		1	senzor č. 6	
	D4		1	senzor č. 5	
	D3		1	senzor č. 4	
	D2		1	senzor č. 3	
	D1		1	senzor č. 2	
	D0		1	senzor č. 1	
2.	D7	požadavek statusu ústředny	1	identifikace ústředny	Ústředna jednorázově odešle zprávu "Central_Ident"
	D6		1	statusu ústředny	Ústředna jednorázově odešle zprávu "Central_Status_I" nezávisle na jejím periodickém vysílání, které tím není nijak ovlivněno
	D5		1	rozšířený status ústředny	Ústředna jednorázově odešle zprávu "Central_Status_II" nezávisle na jejím periodickém vysílání, které tím není nijak ovlivněno
	D4		1	nastavené logování	Ústředna jednorázově odešle zprávu "Logging_Setting_Status"
	D3		1	nastavené BT	Ústředna jednorázově odešle zprávy "BT_Setting_Status"
	D2		1	nastavení energetické bilance	Ústředna jednorázově odešle zprávu "Energy_Balance_Setting_Status"
	D1 - D0		00 - 03h	rezerva	
3.	D7	dotaz na nastavené parametry měření	1	senzor č. 8	Ústředna jednorázově odešle příslušný počet zpráv "Measurement_Setting_Status" dle nastavených bitů
	D6		1	senzor č. 7	
	D5		1	senzor č. 6	
	D4		1	senzor č. 5	
	D3		1	senzor č. 4	
	D2		1	senzor č. 3	
	D1		1	senzor č. 2	
	D0		1	senzor č. 1	
4.	D7 - D0	dotaz na nastavené spouštěcí podmínky logování		bude definováno později	bude definováno později
5. - 8.	D7 - D0	rezerva	00h - 0FFh	rezerva	

Tabulka A.6: Status_Query

byte	bit	význam	hodnota	popis
1.	D7 - D2	rok	00h - 3Fh	2000 - 2064
	D1 - D0	měsíc	01h - 0Ch	01 - leden ... 0Ch - prosinec (vyšší bity) 01 - leden ... 0Ch - prosinec (nižší bity)
2.	D7-D6		00h, 0Dh - 0Fh	rezerva
	D5-D1	den	00h	rezerva
			01h - 1Fh	1. - 31. den v měsíci
	D0	hodina	00h - 17h	00 - 23 hodin (MSB) 00 - 23 hodin (LSB)
3.	D7-D4		18h - 1Fh	rezerva
	D3-D0	minuta	00h - 3Bh	00 - 59 minut (vyšší bity) 00 - 59 minut (nižší bity)
4.	D7-D6		3Ch - 3Fh	rezerva
	D5-D0	vteřina	00h - 3Bh	00 - 59 vteřin
5.	D7	platnost dat	0	časový údaj není platný
			1	časový údaje je platný
	D6 - D0	setina vteřiny	00h - 63h	0.00 - 0.99 vteřin
			64h - 7Fh	rezerva
6.	D7 - D5	rezerva	00h - 07h	rezerva
	D4 - D3	rezerva	00h	rezerva
			01h	rezerva
			02h	rezerva
			03h	rezerva
	D2 - D1	stav měření	00h	neprobíhá měření
			01h	probíhá měření bez logování
			02h	čekání na trigger
			03h	probíhá logování
	D0	stav BT	0	BT vypnuto
			1	BT zapnuto
7.	D7	připojené senzory k ústředně	1	8. senzor je připojen k ústředně
	D6		1	7. senzor je připojen k ústředně
	D5		1	6. senzor je připojen k ústředně
	D4		1	5. senzor je připojen k ústředně
	D3		1	4. senzor je připojen k ústředně
	D2		1	3. senzor je připojen k ústředně
	D1		1	2. senzor je připojen k ústředně
	D0		1	1. senzor je připojen k ústředně
7.	D7	kompatibilita SW a HW senzorů a ústředny	1	8. senzor má nekompatibilní SW nebo HW
	D6		1	7. senzor má nekompatibilní SW nebo HW
	D5		1	6. senzor má nekompatibilní SW nebo HW
	D4		1	5. senzor má nekompatibilní SW nebo HW
	D3		1	4. senzor má nekompatibilní SW nebo HW
	D2		1	3. senzor má nekompatibilní SW nebo HW
	D1		1	2. senzor má nekompatibilní SW nebo HW
	D0		1	1. senzor má nekompatibilní SW nebo HW

Tabulka A.7: Central_Status_I

byte	bit	význam	hodnota	popis
1.	D7 - D5	aktuální režim napájení	00h	rezerva
			01h	Režim 1
			02h	Režim 2
			03h	Režim 3
			04h	Režim 4
			05h	Režim 5
			06h - 07h	rezerva
	D4	aktuální napájení	0	napájení z ext. Zdroje
			1	napájení z baterie
	D3 - D2		00	neprobíhá dobíjení baterie
			01h	probíhá dobíjení baterie
			02h	neprobíhá dobíjení baterie z důvodu přehřátí
			03h	rezerva
	D1		0	napájení je O.K.
			1	bude následovat vypnutí ústředny kvůli kritickému stavu baterie
	D0	identické adresy	1	v systému se vyskytuje modul, který má stejnou adresu jako tento
2.	D7 - D0	napětí externího napájecího zdroje	00h - 0F0h	naměřené napětí 0 – 24V s rozlišením 0.1V, Napětí = hodnota · 0.1V
			0FFh	překročen rozsah měření
			0FEh	chyba měření napětí
			0F1h - 0FDh	rezerva
3.	D7 - D0	Teplota baterie	01h – 07Fh	naměřená teplota -1 ÷ -40 °C s rozlišením 1°C, Teplota = (hodnota – 80h) · 1°C
			80h – 0EFh	naměřená teplota 0°C až 111 °C s rozlišením 1°C, Teplota = (hodnota – 80h) · 1°C
			000h	překročen rozsah měření v záporných hodnotách
			0FFh	překročen rozsah měření v kladných hodnotách
			0FEh	chyba měření teploty
			0F0h – 0FDh	rezerva
4.	D7	připojená baterie	0	baterie není připojena
			1	baterie je připojena
	D6 - D4	stav baterie	00h - 07h	00 - vybitá . . . 07 - plně nabitá
	D3 - D2	stav lithiové baterie	00h	kritický stav baterie - nutná výměna
			01h	baterie je téměř vybitá - nutná výměna
			02h	baterie OK
			03h	rezerva
	D1 - D0	rezerva	00h - 08h	rezerva
5.	D7 - D3	velikost SD karty	00h	žádná karta
			01h - 03h	rezerva
			04h	32 MB
			05h	64 MB
			06h	128 MB
			07h	256 MB
			08h	512 MB
			09h	1 GB
			0Ah	2 GB
			0Bh	4 GB
			0Ch	8 GB
			0Dh	16 GB
			0Eh	32 GB
			0Fh	64 GB
			10h	128 GB
			11h	256 GB
			12h - 1Fh	rezerva
	D2 - D0	rezerva	00h - 08h	rezerva
6. - 7.	D15 - D0	kontrolní součet konfigurace systému	0000h - 0FFFFh	
8.	D7	výpadek komunikace se senzory	1	7. senzor má výpadky v komunikaci
	D6		1	6. senzor má výpadky v komunikaci
	D5		1	5. senzor má výpadky v komunikaci
	D4		1	4. senzor má výpadky v komunikaci
	D3		1	3. senzor má výpadky v komunikaci
	D2		1	2. senzor má výpadky v komunikaci
	D1		1	1. senzor má výpadky v komunikaci
	D0		1	0. senzor má výpadky v komunikaci

Tabulka A.8: Central_Status_II

byte	bit	význam	hodnota	popis
	D7		1	senzor č. 8
1.	D6	seznam sensorů, které budou logovány	1	senzor č. 7
	D5		1	senzor č. 6
	D4		1	senzor č. 5
	D3		1	senzor č. 4
	D2		1	senzor č. 3
	D1		1	senzor č. 2
	D0		1	senzor č. 1
2.	D7	Platnost dat v 1. byte	0	ponechat naposledy nastavený seznam senzorů, které mají být logovány (na hodnotu 1. bytu nebude brán zřetel)
			1	seznam senzorů pro logování (1. byte) jsou platné
	D6 - D0		00h - 7Fh	rezerva
3.	D7 - D6	seznam ostatních CAN zpráv, které budou logovány	00h	beze změny
			01h	status ústředny nebude logován
			02h	status ústředny bude logován
			03h	rezerva
	D5 - D4		00h	beze změny
			01h	ostatní CAN zprávy ústředny nebude logován
			02h	ostatní CAN zprávy ústředny bude logován
			03h	rezerva
	D3 - D2		00h	beze změny
			01h	všechny CAN zprávy, určené pro danou měřící ústřednu nebudou logovány
			02h	všechny CAN zprávy, určené pro danou měřící ústřednu budou logovány
			03h	rezerva
	D1 - D0		00h	beze změny
			01h	CAN zprávy s daty o vypočtené energetické bilance nebudou logovány
			02h	CAN zprávy s daty o vypočtené energetické bilance budou logovány
			03h	rezerva
4.	D7 - D0	trigrovací podmínky	00h	okamžitý start logování
			01h	start logování po stisku tlačítka "START"
			02h	start logování po splnění rozšiřujících trigrovacích podmínek
			03h - 0FFh	ukončení logování
5. - 8.	D7 - D0	rezerva	00h - 0FFh	rezerva

Tabulka A.9: Logging_Setting

byte	bit	význam	hodnota	popis
1.	D7	seznam sensorů, které budou logovány	1	senzor č. 8
	D6		1	senzor č. 7
	D5		1	senzor č. 6
	D4		1	senzor č. 5
	D3		1	senzor č. 4
	D2		1	senzor č. 3
	D1		1	senzor č. 2
	D0		1	senzor č. 1
2.	D7 - D0	rezerva	00 - 0FFh	rezerva
3.	D7 - D6	seznam ostatních CAN zpráv, které budou logovány	00h	rezerva
			01h	status ústředny nebude logován
			02h	status ústředny bude logován
			03h	rezerva
	D5 - D4		00h	rezerva
			01h	ostatní CAN zprávy ústředny nebude logován
			02h	ostatní CAN zprávy ústředny bude logován
			03h	rezerva
	D3 - D2		00h	rezerva
			01h	všechny CAN zprávy, určené pro danou měřící ústřednu nebudou logovány
			02h	všechny CAN zprávy, určené pro danou měřící ústřednu budou logovány
			03h	rezerva
	D1 - D0		00h	rezerva
			01h	CAN zprávy s daty o vypočtené energetické bilance nebudou logovány
			02h	CAN zprávy s daty o vypočtené energetické bilance budou logovány
			03h	rezerva
4.	D7 - D0	trigrovací podmínky	00h	okamžitý start logování
			01h	start logování po stisku tlačítka "START"
			02h	start logování po splnění rozšiřujících trigrovacích podmínek
			03h - 0FFh	ukončení logování
5.	D7 - D0	zaplnění SD karty	00h	žádná karta
			01h - 65h	stav zaplnění karty [0 - 100%]
			66h - 0FFh	rezerva
6. - 8.	D7 - D0	rezerva	00h - 0FFh	rezerva

Tabulka A.10: Logging_Setting_Status

byte	bit	význam	hodnota	popis
1.	D7 - D4	adresa modulu, jehož senzory budou vysílány na BT	00 - 0Fh	modul 1 - modul 16
	D3 - D0	rezerva	00 - 0Fh	rezerva
2.	D7	seznam naměřených dat od senzorů, které budou vysílány na BT	1	senzor č. 8
	D6		1	senzor č. 7
	D5		1	senzor č. 6
	D4		1	senzor č. 5
	D3		1	senzor č. 4
	D2		1	senzor č. 3
	D1		1	senzor č. 2
	D0		1	senzor č. 1
3.	D7 - D0	rezerva	00h - 0FFh	rezerva
4.	D7	zapnutí / vypnutí BT	1	zapnutí BT modulu
	D6 - D0	rezerva	00h - 7Fh	rezerva
5. - 8.	D7 - D0	rezerva	00h - 0FFh	rezerva

Tabulka A.11: BT_Setting

byte	bit	význam	hodnota	popis
1.	D7 - D4	adresa modulu, jehož senzory budou vysílány na BT	00 - 0Fh	modul 1 - modul 16
	D3 - D0	rezerva	00 - 0Fh	rezerva
2.	D7	seznam naměřených dat od senzorů, které budou vysílány na BT	1	senzor č. 8
	D6		1	senzor č. 7
	D5		1	senzor č. 6
	D4		1	senzor č. 5
	D3		1	senzor č. 4
	D2		1	senzor č. 3
	D1		1	senzor č. 2
	D0		1	senzor č. 1
3.	D7 - D0	rezerva	00h - 0FFh	rezerva
4.	D7	zapnutí / vypnutí BT	1	zapnutí BT modulu
	D6 - D0	rezerva	00h - 7Fh	rezerva
5. - 8.	D7 - D0	rezerva	00h - 0FFh	rezerva

Tabulka A.12: BT_Setting_Status

byte	bit	význam	hodnota	popis
1.	D7 - D0	požadovaný způsob svícení LED v sensoru č.1	00h	LED beze změny
			01h	**LED nesvítí**
			02h	LED trvale svítí
			03h	LED blik (100ms svítí, 900ms nesvítí)
			04h	LED bliká (200ms svítí, 200ms nesvítí)
			05h - FFh	rezerva - beze změny
2.	D7 - D0	požadovaný způsob svícení LED v sensoru č.2	00h	LED beze změny
			01h	**LED nesvítí**
			02h	LED trvale svítí
			03h	LED blik (100ms svítí, 900ms nesvítí)
			04h	LED bliká (200ms svítí, 200ms nesvítí)
			05h - FFh	rezerva - beze změny
3.	D7 - D0	požadovaný způsob svícení LED v sensoru č.3	00h	LED beze změny
			01h	**LED nesvítí**
			02h	LED trvale svítí
			03h	LED blik (100ms svítí, 900ms nesvítí)
			04h	LED bliká (200ms svítí, 200ms nesvítí)
			05h - FFh	rezerva - beze změny
4.	D7 - D0	požadovaný způsob svícení LED v sensoru č.4	00h	LED beze změny
			01h	**LED nesvítí**
			02h	LED trvale svítí
			03h	LED blik (100ms svítí, 900ms nesvítí)
			04h	LED bliká (200ms svítí, 200ms nesvítí)
			05h - FFh	rezerva - beze změny
5.	D7 - D0	požadovaný způsob svícení LED v sensoru č.5	00h	LED beze změny
			01h	**LED nesvítí**
			02h	LED trvale svítí
			03h	LED blik (100ms svítí, 900ms nesvítí)
			04h	LED bliká (200ms svítí, 200ms nesvítí)
			05h - FFh	rezerva - beze změny
6.	D7 - D0	požadovaný způsob svícení LED v sensoru č.6	00h	LED beze změny
			01h	**LED nesvítí**
			02h	LED trvale svítí
			03h	LED blik (100ms svítí, 900ms nesvítí)
			04h	LED bliká (200ms svítí, 200ms nesvítí)
			05h - FFh	rezerva - beze změny
7.	D7 - D0	požadovaný způsob svícení LED v sensoru č.7	00h	LED beze změny
			01h	**LED nesvítí**
			02h	LED trvale svítí
			03h	LED blik (100ms svítí, 900ms nesvítí)
			04h	LED bliká (200ms svítí, 200ms nesvítí)
			05h - FFh	rezerva - beze změny
8.	D7 - D0	požadovaný způsob svícení LED v sensoru č.8	00h	LED beze změny
			01h	**LED nesvítí**
			02h	LED trvale svítí
			03h	LED blik (100ms svítí, 900ms nesvítí)
			04h	LED bliká (200ms svítí, 200ms nesvítí)
			05h - FFh	rezerva - beze změny

Tabulka A.13: LED_Setting

byte	bit	význam	hodnota	popis
1.	D7 - D0	režim napájení	00h	beze změny
			01h	**Režim 1**
			02h	Režim 2
			03h	Režim 3
			04h	Režim 4
			05h	Režim 5
			06h - FFh	rezerva - beze změny
2.	D7 - D5	režim spotřeby	00h	beze změny
			01h	**normální režim**
			02h	úsporný režim
			03h	pohotovostní režim
			04h - 07h	rezerva - beze změny
	D4 - D0	rezerva	00h - 1Fh	rezerva
3. - 8.	D7 - D0	rezerva	00h - 0FFh	rezerva

Tabulka A.14: Power_Setting

byte	bit	význam	hodnota	popis
1.	D7 - D2	rok	00h - 3Fh	2000 - 2064
	D1 - D0	měsíc	01h - 0Ch	01 - leden ... 0Ch - prosinec
2.	D7 - D6		00h, 0Dh - 0Fh	rezerva
	D5 - D1	den	00h	rezerva
			01h - 1Fh	1. - 31. den v měsíci
	D0	hodina	00h - 17h	00 - 23 hodin
3.	D7-D4		18h - 1Fh	rezerva
	D3 - D0	minuta	00h - 3Bh	00 - 59 minut
4.	D7 - D6		3Ch - 3Fh	rezerva
	D5 - D0	vteřina	00h - 3Bh	00 - 59 vteřin
5.	D7	platnost dat	0	časový údaj není platný
			1	časový údaje je platný
	D6 - D0	setina vteřiny	00h - 63h	0.00 - 0.99 vteřin
			64h - 7Fh	rezerva
6. - 8.	D7 - D0	rezerva	00h - 0FFh	rezerva

Tabulka A.15: Master_Clock

byte	bit	význam	hodnota	popis
1.	D7 - D4	perioda měření	01h	50ms
			02h	100ms
			03h	250ms
			04h	500ms
			05h	1s
			06h	2s
			07h	5s
			08h	10s
			09h	20s
			0Ah	60s
			00h	ponechat naposledy nastavenou hodnotu
			0Bh - 0Fh	rezerva
	D3 - D0	rezerva	00h - 0Fh	rezerva
2.	D7 - D0	rezerva	00h - 0FFh	rezerva
3.	D7 - D0	počet odměrů	00h	ukončení měření
			01h - 0FEh	1 - 254 odměrů
			0FFh	trvalé měření
4.	D7 - D0	Senzory, ze kterých se počítá energetická bilance	1	senzor č. 8
			1	senzor č. 7
			1	senzor č. 6
			1	senzor č. 5
			1	senzor č. 4
			1	senzor č. 3
			1	senzor č. 2
			1	senzor č. 1
5.	D7	Platnost dat v 2. byte	0	ponechat naposledy nastavený seznam senzorů pro výpočet energetické bilance (na hodnotu 2. bytu nebude brán zřetel)
			1	seznam senzorů pro výpočet energetické bilance (2. byte) je platný
	D6 - D0		00h - 7Fh	rezerva
6.	D7 - D0	vynulování počitadla energetické bilance	1	senzor č. 8
			1	senzor č. 7
			1	senzor č. 6
			1	senzor č. 5
			1	senzor č. 4
			1	senzor č. 3
			1	senzor č. 2
			1	senzor č. 1
7. - 8.	D7 - D0	rezerva	00h	rezerva

Tabulka A.16: Energy_Balance_Setting

byte	**bit**	**význam**	**hodnota**	**popis**
1.	D7 - D4	perioda měření	01h	50ms
			02h	100ms
			03h	250ms
			04h	500ms
			05h	1s
			06h	2s
			07h	5s
			08h	10s
			09h	20s
			0Ah	60s
			00h	rezerva
			0Bh - 0Fh	rezerva
	D3 - D0	rezerva	00h - 0Fh	rezerva
2.	D7 - D0	rezerva	00h - 0FFh	rezerva
3.	D7 - D0	počet odměrů	00h	měření bylo ukončeno
			01h - 0FEh	1 - 254 odměrů
			0FFh	trvalé měření
5. - 8.	D7 - D0	rezerva	00h	rezerva

Tabulka A.17: Energy_Balance_Setting

byte	**bit**	**význam**	**hodnota**	**popis**
1.	D7 - D0	rezerva	00h - 0FFh	rezerva
2. - 4.	D7 - D0	vypočtená energetická bilance	000001h -7FFFFFh	vypočtená energetická bilance - 763.55Ah až -9.1·10-5 s rozlišením 9.1·10-5, Bilance = (hodnota - 800000h) · 9.1022·10-5
			800000h – FFFFEFh	vypočtená energetická bilance 0 až 763.55Ah s rozlišením 9.1·10-5, Bilance = (hodnota - 800000h) · 9.1022·10-5
			000000h	překročen rozsah měření v záporných hodnotách
			0FFFFFFh	překročen rozsah měření v kladných hodnotách
			0FFFFFEh	chyba výpočtu energetické bilance
			0FFFFFDh	energetická bilance nepočítána
			0FFFFF0h – 0FFFFFCh	rezerva
5. - 7.		rezerva	00h - 0FFh	rezerva
8.	D7	data pro BT	1	Tento bit bude nastaven vždy, když od předchozího paketu, který měl tento bit nastaven, uplynulo 0.5s (pro časy měření 50ms až ¡0.5s) nebo pro všechny pakety s periodou 0.5s a vyšší. Pouze zprávy s tímto nastaveným bitem jsou přeposílány na BT
	D6 - D0	rezerva	00h	rezerva

Tabulka A.18: Energy_Balance_Data

Příloha B

Obsah přiloženého CD

K této práci je přiloženo CD, na kterém jsou uloženy zdrojové kódy.

- Adresář BinToAsc_1_2: Obsahuje zdrojový kód pro konverzi binárních dat do formátu *.asc
- Adresář Link_File_1_1: Obsahuje zdrojový kód pro slučování binárních souborů
- Adresář CAPL_prog_1_2: Obsahuje zdrojové kódy pro vizualizační software
- Adresář Mefuse_Graph_1_3: Obsahuje zdrojový kód softwaru pro grafické zobrazení naměřených dat
- Adresář Diplomova_prace : Obsahuje diplomovou práci v elektronické podobě ve formátu pdf