VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

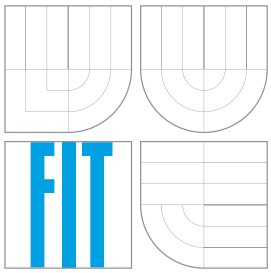
ZÁZNAM A VIZUALIZACE PARAMETRŮ OSMIVÁLCOVÉHO ZÁŽEHOVÉHO MOTORU

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE JÁN KUBIŠ
AUTHOR

BRNO 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

ZÁZNAM A VIZUALIZACE PARAMETRŮ OSMIVÁLCOVÉHO ZÁŽEHOVÉHO MOTORU

COMBUSTION ENGINE POCKET PC DIAGNOSTIC TERMINAL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JÁN KUBIŠ
AUTHOR

VEDOUCÍ PRÁCE — Ing. RICHARD RŮŽIČKA, Ph.D.
SUPERVISOR

BRNO 2007

Zadání bakalářské práce/4110/2006/xkubis05

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačových systémů Akademický rok 2006/2007

Zadání bakalářské práce

Řešitel: **Kubiš Ján**
Obor: Informační technologie
Téma: **Diagnostický terminál spalovacího motoru na Pocket PC**
Kategorie: Uživatelská rozhraní

Pokyny:

1. Seznamte se s programováním Pocket PC s operačním systémem MS Windows 2003 (CE).
2. Prostudujte způsob, jakým vysílá diagnostická data po sériové lince řídicí jednotka spalovacího motoru GEMS REM 8CZ. Využijte poznatky shrnuté v bakalářské práci z roku 2004/05.
3. Navrhněte jednoduchý program, který by pod OS Win 2003 snímal diagnostická data ze sériového portu a zobrazoval je na displeji Pocket PC.
4. Program realizujte a ověřte jeho funkčnost.

Literatura:

- Krell, B. E.: Pocket PC Developer's Guide, Mc Graw Hill/Osborne 2002
- Kabátek, P.: Záznam a vizualizace parametrů osmiválcového zážehového motoru, bakalářská práce, FIT VUT v Brně, 2005

Při obhajobě semestrální části projektu je požadováno:

1. Seznamte se s programováním Pocket PC s operačním systémem MS Windows 2003 (CE).
2. Prostudujte způsob, jakým vysílá diagnostická data po sériové lince řídicí jednotka spalovacího motoru GEMS REM 8CZ. Využijte poznatky shrnuté v bakalářské práci z roku 2004/05.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním paměťovém médiu (disketa, CD-ROM), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Růžička Richard, Ing., Ph.D.**, UPSY FIT VUT
Datum zadání: 1. listopadu 2006
Datum odevzdání: 15. května 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačových systémů a sítí
612 66 Brno, Božetěchova 2

L.S.

doc. Ing. Zdeněk Kotásek, CSc.
vedoucí ústavu

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. Pan

Jméno a příjmení: **Ján Kubiš**
Id studenta: 83961
Bytem: Centrum II 84/30, 018 41 Dubnica nad Váhom
Narozen: 27. 12. 1979, Žilina
(dále jen "autor")

a

2. Vysoké učení technické v Brně

Fakulta informačních technologií
se sídlem Božetěchova 2/1, 612 66 Brno, IČO 00216305
jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....
(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
bakalářská práce

Název VŠKP: Diagnostický terminál spalovacího motoru na Pocket PC
Vedoucí/školitel VŠKP: Růžička Richard, Ing., Ph.D.
Ústav: Ústav počítačových systémů
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:

tištěné formě počet exemplářů: 1
elektronické formě počet exemplářů: 2 (1 ve skladu dokumentů, 1 na CD)

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti:
   - ☑ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

.....	.....
Nabyvatel	Autor

Abstrakt

Moja bakalárska práca, v ktorej som mal vytvoriť programový systém pracujúci pod operačním systémom Windows CE 2003, sa zaoberá príjmaním a spracovaním dat, ktoré posiela riadiaca jednotka osmivalcového zážehového motoru Tatra T613/4i. Program je schopný príjmať jednotlivé tabuľky, ktoré riadiaca jednotka vysiela. Veličiny jednotlivých tabuliek je možné zobrazovať vo forme textových a číselných dat. Ďalej systém tiež vie zobrazovať chybné hlásenia, ktoré je riadiaca jednotka schopná posielať.

Klíčová slova

Sériová linka sériové komunikačné rozhranie RS232

T613/4i typ motoru montovaný do automobilov značky Tatra

Windows CE 2003 operačný systém z dielne firmy Microsoft.

Riadiacá jednotka Mikropočítač, malý jednoduchý počítač obsahující procesor, pamäť a vstupno/výstupné rozhranie

Abstract

My bachelor work, in which am had create programmatic system working under operating system Windows CE 2003 with under-used to be revenue and data processing, which sending control unit of engine Tatra T613/4i. Programme work with receive single table, which kontrol unit sending. Quantity single table is possible represent in practice textual or numeral data. System also can represent error report, which kontrol unit can sending.

Keywords

Serial line serial computer communication.s interface RS232

T613/4i type of engine prefab into the cars of Tatra

Windows CE 2003 operating system with workshop firm Microsoft

Control unit Microcomputer, small simple computer containing procesor, memory and in/out interface.

Citace

Ján Kubiš: Záznam a vizualizace parametrů
osmiválcového zážehového motoru, bakalářská práce, Brno, FIT VUT v Brně, 2007

Záznam a vizualizace parametrů osmiválcového zážehového motoru

Prehlásenie

Prehlasujem, že som túto bakalárskú prácu vypracoval samostatne pod vedením pána Ing. Richard Růžička

.....

Ján Kubiš

14. května 2007

Obsah

Kapitola 1

Úvod

Od doby, kedy bol uvedený do premávky prvý automobil, prešlo veľa rokov. Za túto dobu došlo k veľkým zmenám a vývoju nových technológií a taktiež aj k vývoju automobilového priemyslu. Keď bol chod motoru v automobiloch riadený na základe mechanických zariadení už sa upustilo a dnes v podstate nenájdete vyrobený automobil, ktorého motor by nebol riadený riadiacou jednotkou. Týmto spôsobom riadenia chodu motoru bolo dosiahnuté hlavne zvýšenie spoľahlivosti, zvýšenie výkonu motoru pri znižujúcej sa spotrebe, zníženia škodlivých emisií a jednoduchšia práca pri detekovaní chyb motora. Týchto výsledkov by sme u starších motorov založených na mechanickom riadení nikdy nedosiahli.

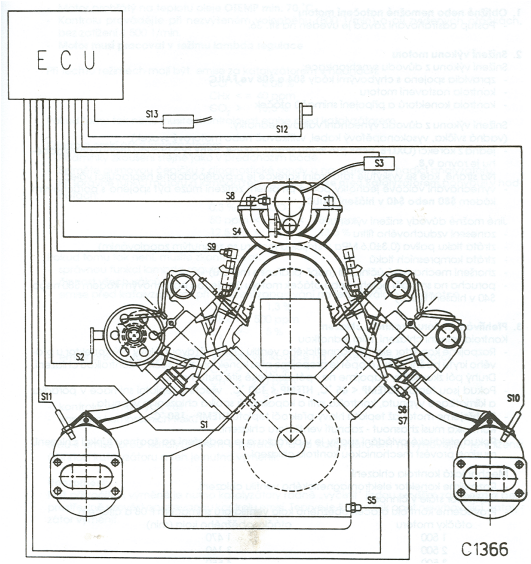
A práve vďaka tomuto je kladený veľký dôraz pri stavbe automobilu na vývoj a správne naprogramovanie riadiacej jednotky, ktorá sa stáva srdcom každého moderného automobilu. Riadiacá jednotka je vlastne taký mikropočítač (malý počítač), ktorý príjma vstupné data, tieto data spracuje a na ich základe vytvorí výstupné data, podľa ktorých je riadený chod motoru. Vstupné data sú do jednotky zasielané od rôznych čidiel a snímačov (teploty - hlav,oleja,vzduchu, otáčky -motoru .....). Riadiaca jednotka má tiež svoju vnútornú pamäť, v ktorej sú uložené tabuľky s prednastavenými hodnotami - napr. pre predstih, voľnobeh ..... Riadiaca jednotka na základe vstupných hodôt vyberie výstupnú hodnotu z danej tabuľky a pošle ju na svoje výstupné konektory, ktoré sú spojené s rôznymi, väčšinou elektricky ovládanými ventilmi, ktoré priamo ovládajú chod motoru.

Konkrétne sa jedná o motor Tatra T613-4i s dvoma riadenými katalyzátormi. Základná dodávka paliva, predstih zápalu a veľa ďalších funkcií je riadeno počítačom firmy GEMS. Všetky data motora (ako napr. tabuľky volumentrickej účinnosti, predstihu zápalu a všetká kalibrácia) sú uložené v pamäti počítača. Motor je vybavený sekvenčným osmibodovým vstrekovaním paliva. Zmiešavací pomer je riadení pomocou dvoch kyslíkových snímačov (lambda sond) nezávisle pre každú stranu valcov. Druhou stránkou veci je mať možnosť prístupu a kontroly veličín, na ktorých základe je riadený chod motoru.

A tento problém sa stal tématom mojej bakalárskej práce - ,,Záznam a vyzulizácia parametrov osemvalcového spaľovacieho motora".

Ide o to, že riadiacá jednotka posiela data zakódované do určitého kódu a pre neznalého

uživateľa, by tieto data nič neznamenali, pretože by nevedel, čo znamenajú. Takže úlohou bolo dajakým spôsobom vytvoriť program, ktorý by zobrazoval data, ktoré posiela riadiaca jednotka tak, aby aj neznalý užívateľ ľahko pochopil čo posielané data znamenajú. Toto všetko je umožnené práve vďaka poslednej uvedenej vlastnosti počitača GEMS a to: komunikácia s PC po sériovej linke. Táto vlastnosť riadiacej jednotky GEMS je dôležitá hlavne pre overenie správneho chodu motoru - je využívaná hlavne v autorizovaných servisoch a to nie len pre vozy TATRA. To znamená, že je dôležité sa touto problematikou zaoberať.



Obrázok 1.1: Schéma umiestnenia snímačov na motore

Kapitola 2

Riadiaca jednotka

Jedná sa o jednotku určenú pre riadenie osemvalcového vidlicového motoru značky Tatra - v našom prípade typu t613/4i. Riadiaca jednotka REM8 je od českej firmy General Engine Management Systems Ltd. Informácie o tejto riadiacej jednotke a o spôsobe zasielania dat cez sériové rozhranie je popisané [5], kde je aj zdroj informácií použitý k tejto téme.

2.1 Základné vlastnosti a funkcie:

1. Nastavenie optimálného predstihu zápaľu vo všetkých režimoch
2. Riadenie otáčok voľnobehu
3. Ovládanie chladiaceho ventilátoru
4. Riadenie regenerácie uhlikového filtru - odlučovač benzínových pár
5. Paľubná diagnostika s výstupom na kontrolku
6. Komunikacia s PC po sériovej linke

2.2 Vstupy riadiacej jednotky

- Otáčky: Snímač Honeywell(na princípe Hallovho javu) sníma dvanásť rovnako veľkých zubov rozmiestnených po 30° na zotrvačníku.
- Poloha motoru: Snímač Honeywell (na princípe Hallovho javu) sníma osem zubov rozmiestnených po 60° na ľavom výfukovom rozvodovom kolese a slúži pre synchronizáciu vstrekovania a zapaľovania. Jeden zub je väčší ako ostatných osem, od neho sa synchronizuje vstrekovanie pri štarte.
- Tlak v sacom potrubí (MAP - Manifold Absolute Pressure): tenzometrické čidlo (Generál Motors) pre snímanie absolutného tlaku v sacom potrubí, vytvára napätie 0-5V úmerne tlkaku 0-104kPa.

- Barometrický tlak (BARO): rovnaký snímač ako pri zisťovaní tlaku v sacom potrubí, ale zaznamenáva tlak okolného vzduchu pre korekciu množstva vstrekovacieho paliva.
- Teplota hláv (HMT - Head Meta Temperature): dva snímače GANZ (380 060 360 01) na hlavách 3. a 4. valca dávajú napätie 0-5V. Nízke napätie indikuje vysokú teplotu a naopak. Pomocou odporov sú rozsahy snímačov posunuté tak, že napätie 5V - 0V odpovedá u jedného snímača teplotnému rozsahu 4 až 205°C a u druhého -50 až 100°C.
- Teplota oleja (OTEMP): Automobilový termistor s rozsahom -40 až 130°C.
- Teplota paliva (FTEMP): Automobilový termistor montovaný v potrubí rozvodu paliva ku vstrekovacím ventilom.
- Teplota vzduchu (ATEMP): Automobilový termistor montovaný v komore sania vzduchu.
- Napätie batérie (BATT): Doba vstreku a doba nasítenia zapaľovacej cievky sú upravované pri zmenách napätia.
- Lambda sondy: Snímače NTK s vyhrievanou titanovou vrstvou sú inštalované v prúde výfukových plynov pred katalyzátorom (jeden pre každú stranu motora) a dávajú napätie 0-1V. Pod 0,5V ide o chudú zmes a nad 0,5V ide o bohatú zmes.
- Poloha škrtiacej klapky (TPS - Throttle Position): Otočný potenciometer dáva napätie 0-5V, škrtiaca klapka je priamo mechanicky spojená s pedálom akcelerátoru, hodnota TPS je teda tiež mierou zošľapnutia pedálu akcelerátora.
- Snímač klimatizácie (ACIN - Air Conditioning Input): Spojením na kostru signalizuje požiadavok na zapnutie klimatizácie. Riadiaca jednotka zapne spojku kompresoru klimatizácie a zvýši voľnobeh, aby kompenzovala pokles voľnobežných otáčok vplyvom väčšieho zaťaženia motora.
- Spínač funkcie čerpadiel (PSPIN - Power Steering Pump Input): Zapnutie na kostru signalizuje činnosť čerpadla posilovača riadenia, riadiaca jednotka zvýší voľnobeh.
- Rýchlosť vozidla (VSS - Vehicle Speed Signal): Riadiaca jednotka počíta inpulzy po dobu pol sekundy.
- Chyba ukostrenia motora (GNDERR): Rozdieľ napätí medzi blokom motora a rámom (ktorý sa mení s odberom pomocných zariadení) slúži ku korekcii vstupu teploty hláv.

2.3 Výtupy riadiacej jednotky

- Zapaľovanie: jediný výstup riadi elektronický spínač zapaľovania (LUCAS).

- Vstrekovače: osem výstupov pre sekvenčné otváranie vstrekovacích ventilov (LUCAS).
- Výstupy ovládania voľnobehu (IAC - Idle Air Control): Komplementárné pulzy 100Hz s šírkovou moduláciou pre budenie obojsmerného pohonu obtokového ventilu Bosh. Ventil riadi množstvo vzduchu pri voľnobehu.
- Regenerácia uhlíkového filtra (CCP - Carbon Canister Purge): Šírkovo modulovanými pulzami 16Hz sa otvára elektromagnetický ventil.
- Palivové čerpadlo: Po zapnutí zapaľovania sa prostredníctvom tochto výstupu spustí na 15s relé elektronického palivového čerpadla, aby došlo k zaplaveniu palivovej sústavy.
- Chladiací ventilátor: Týmto výstupom sa ovláda cez relé, elektromagnetický ventil hydrospojky chladiacého ventilátoru.
- Spojka kompresoru klimatizácie: Týmto výstupom sa ovláda relé elektromagnetickej spojky kompresoru klimatizacie
- Kontrolka palubnej diagnostiky/výstup sériovych dat do PC.

Kapitola 3

Komunikácia PC s riadiacou jednotkou REMS od GEMS

3.1 Komunikácia riadiacej jednotky s PC vo dvoch režimoch

1. Prvý režim: Stane sa aktívnym pokiaľ pripojite PC k riadiacej jednotke a následne zapnete zapaľovanie vozidla. V tomto režime riadiaca jednotka vysiela jedinú tabuľku a to tabuľku D. Je to tabuľka, ktorú riadiaca jednotka vysiela automaticky po zapojení sériového káblu bez nutnosti vysielania akéhokoľvek riadiacého znaku. Tabuľka D obsahuje základné údaje o stave motoru. Presné informácie o obsahu jednotlivých tabuliek budú uvedené neskôr.

2. Druhý režim: Stane sa aktívnym ak budeme postupovať následovne

   - Zapnúť zapaľovanie motoru
   - Pripojiť riadiacu jednotku k PC
   - Poslaním z PC do riadiacej jednotky dva znaky ,,yy"

   V tomto režime je jednotka schopná posielať na sériovú linku sedem rôznych tabuliek. Tabuľky sa budú označovať následovne: R, W, E, O, X, Y.

   Okamžite po zaslaní znakov ,,yy" do riadiacej jednotky, začne riadiaca jednotka posielať na sériovú linku tabuľku R. Aby riadiaca jednotka začala odosielať jednu z uvedených tabuliek, je potreba zaslať do riadiacej jednotky jeden znak označujúci požadovanú tabuľku, napr. pre tabuľku E by to bol znak ,,E". Čiže jednotlivé odosielajúce znaky, označujú jednotlivé tabuľky. Každá tabuľka osahuje šestnásť hodnôt, ktoré riadiaca jednotka posiela na sériovú linku dokola zasebou. Bližšie špecifikácie jednotlivých tabuliek a ích veličín ktoré obsahujú, bude upresnená v neskorších kapitolách.

3.2 Parametre sériovej linky:

Komunikácia medzi PC a riadiacou jednotkou prebieha po sériovej linke. Pre ustanovenie spojenia a správný priebeh komunikácie, je nutné pred zahájením nastaviť potrebné parametre sériovej linky. Ďalej je nutné brať do úvahy hardwarové parametre PC a riadiacej jednotky - aby napríklad nebola nastavená väčšia rýchlosť prenosu, akou je riadiaca jednotka schopná komunikovať.

Medzi parametre sériovej linky patria:

- Rýchlosť prenosu - riadiaca jednotka vyžaduje 9600 bitov/s
- Počet naraz prenesených bitov - v našom prípade 8
- Parita - riadiaca jednotka vyžaduje nepárnu paritu
- Počet stop bitov - v našom prípade je nastavené na 1 stop bit
- Riadenie - hardflow alebo softflow - v našom prípade ani jedno - oba nastavené na false

3.3 Data posielané riadiacou jednotkou:

Riadiaca jednotka posiela po sériovej linke data po znakoch. Každú hodnotu alebo názov posiela v sekvencii siedmich znakoch. Riadiaca jednotka pracuje tak, že posiela data po sériovej linke jeden za druhým, ako keby tlačila na papier. Preto prvé dva znaky(ich ordinálna hodnota) zo siedmich, tvoria súradný systém. Prvé dva znaky (ich ordinálna hodnota) značia kam sa hodnota alebo jej názov veličiny na stránke vytlačia. Pre rozlúštenie kódovania, je teda nezbytné mať k dispozícií vzor posielaných dat od každej tabuľky. Tieto a nižšie popísané informácie sú čerpané z [5].

3.4 Postup pre dekódovanie dat posielaných riadiacou jednotkou:

Postup pre dekódovanie prijatých dat je pre každú tabuľku zhodný. Vo vzorkách dat pre každú tabuľku je nutné nájsť názvy 16 veličín (každá ma maximálne 5 znakov - napr. speed, prval, tps ...) a od konca každej veličiny prejisť o 7 znakov naspäť, kde pré dva znaky z našich 7-mich určujú súradnicový systém. Tieto znaky si pre každú veličinu zapamätáme tak, ako išli po sebe. Prvý znak z týchto dvoch znakov určuje riadok a druhý stĺpec, samozrejme znak neurčuje súradnice ale jeho ordinálne číslo.

Teraz vieme, akým spôsobom rozkódovať zobrazenie názvu veličín z dat, ktoré riadiaca jednotka posiela. Ďalšiu dôležitú vec, ako získať z posielaných dát hodnoty k daným veličinám,

si preberieme teraz. Dôležitý je pre nás fakt, že každá hodnota k danej veličine sa nachádza na jednom riadku. K hodnote druhého znaku súradnicového označenia veličiny, čiže stĺpca, je nutné pripočítať hodnotu 6, a máme súradnice hodnoty danej veličiny. Toto prevedieme so všetkými dvojicami znakov pre každú veličinu zvlášť. Týmto získame pre každú veličinu všetkých tabuliek novú dvojcu znakov, ktorých ordinálne vyjadrenie určuje súradnícový systém, kam sa na stránku vypíšu hodnoty jednotlivých veličín.

Týmto máme vyriešnú obrovskú čať problému, pretože všetky znaky, ktoré riadiaca jednotka posiela, môžeme ignorovať, sú pre nás nezaujímavé. Zaujímajú nas len tie dva znaky, po ktorých riadiaca jednotka posiela veličinu a po nich k danej veličine patriacu hodnotu. Kedže dvojic znakov je šestnáť a tabuliek osem, znaky sú posielané vždy práve pre jednu tabuľku. To znamená, že všetky tabuľky využívajú šestnáť rovnakých dvojíc znakov - len ku každej hodnote dvojic znakov tabuľka priradí rôznu veličinu a jej hodnotu. V praxi to znamená, že ak náš ešte neznámi algoritmus, narazí v postupnosti prijatých dat na jednu zo šestnástich dvojic znakov, tak už dopredu vie čo bude nasledovať v ďalších piatich znakoch (veličina alebo príslušná hodnota k danej veličine) reprezentovanej dvojcou znakov a tabuľkou, ktorú riadiaca jednotka práve posiela na sériovu linku.

3.5 Príklad zobrazenia tabuľky:

3.5.1 Tabuľka D:

SPEED	0	TPS	100
PRVAL	127	HTEMP	82
SPEED	0	SPEED	0
ATEMP	25	FAIL2	$A0
OTEMP	125	FAIL3	$02
FAIL	$0A	FAIL4	$00
msecL	5	msecR	5
FMOD	.0	SMOD	.0

3.6 Zoznam dvojic znakov a ich ordinalných čísiel

V stĺpci ,,veličina" sú dva špecifické znaky, po ktorých v prúde dat na sériovej linke následuje postupnosť piatich znakov, ktoré označujú názov veličiny. V stĺpci ,,hodnota" sa nachádzajú dva špecifické znaky, po ktorých sa v prúde dat na sériovej linke vyskytuje postupnosť piatich znakov, ktoré označujú hodnotu danej veličiny. Ďalšie stĺpce označené podľa typu tabuľky, reprezentujú všetky tabuľky, ktoré je riadiaca jednotka schopná vysielať. Nachadzajú sa vnich názvy šestnástich veličín k daným dvojciam špecifických znakov.

Význam daných veličín bude uvedený v ďalších kapitolách.

Veličina	Hodnota	D	R	W	E	O	Q	X	Y
m! (32,33)	m. (32,39)	speed	speed	batt	speed	speed	tpsad	tpsad	speed
!! (33,33)	!. (33,39)	prval	prval	bvolt	prval	prval	tps	adair	prval
m. (32,46)	m4 (32,52)	tps	tps	gnder	ttflg	tps	admap	adoil	oxbf2
!. (33,46)	!4 (33,52)	htemp	htemp	htemp	htemp	htemp	press	rfuel	oxflg
m; (32,59)	mA (32,65)	fmod	fmod	adoxl	atemp	atemp	adbar	rhmt1	time
!; (33,59)	!A (33,65)	smod	smod	adoxr	otemp	otemp	baro	rhmt2	vss
#$ (35,36)	#* (35,42)	speed	otemp	fail1	adoxl	idset	adoil	adoxr	ttflg
#6 (35,54)	#¡ (35,60)	speed	vss	faila	adoxr	idlms	rhmt1	adgnd	stat5
$$ (36,36)	$* (36,42)	atemp	ve(f)	fail2	oxler	stat9	otemp	adoxl	stat1
$6 (36,54)	$¡ (36,60)	fail2	igadv	failb	oxrer	igadv	shmt1	vscnt	stat6
%$ (37,36)	%*(37,42)	otemp	oxlfb	fail3	oxlfb	oxlfb	adair	admap	stat2
%6 (37,54)	%¡(37,60)	fail3	oxrfb	failc	oxrfb	oxrfb	rhmt2	digin	stat7
&$ (38,36)	&* (38,42)	fail1	blfbk	fail4	blfbk	blfbk	atemp	adbar	stat3
&6 (38,54)	&¡ (38,60)	fail4	brfbk	faild	brfbk	brfbk	shmt2	speed	stat8
.$ (39,36)	.* (39,42)	msecl	msecl	stat5	oxflg	msecl	rfuel	batt	stat4
.6 (39,54)	.¡ (39,60)	msecr	msecr	stat9	oxbf2	msecr	ftemp	nic	stat9

m=medzera, v zátvorke je zobrazené ordinálne vyjadrenie, pod jednotlivými písmenami je názov veličiny v tabuľke, ktorú reprezentuje dané písmeno

3.7 Príklady dat vysielaných riadiacou jednotkou pre jednotlivé tabuľky:

3.7.1 Tabuľka D:

```
%$OTEMP&* $0A'* 5#< 0$< $A0%< $02&< $00'< 5 ' #* 0$* 25%* 125&* $0A'*
5#6SPEED$< $A0%< $02&< $00'< 5 ' 0!' 127 4 100!4 82 A .0!A .0#* 0$* 25%*
125&* $0A'* 5#< 0$6FAIL2%< $02&< $00'< 5 ' 0!' 128 4 100!4 82 A .0!A .0#*
0$* 25%* 125&* $0A'* 5#< 0$< $A0%6FAIL3&< $00'< 5 ' 0!' 127 4 100!4 82 A
.0!A .0#* 0$* 25%* 125&* $0A'* 5#< 0$< $A0%< $02&6FAIL4'< 5 ' 0!' 128 4
100!4 82 A .0!A .0#* 0$* 25%* 125&* $0A'* 5#< 0$< $A0%< $02&< $00'6msecR '
0!' 128 4 100!4 82 A .0!A .0#* 0$* 25%* 125&* $0A'* 5#< 0$< $A0%< $02&<
$00'< 5 !SPEED!' 127 4 100!4 82 A .0!A .0#* 0$* 25%* 125&* $0A'* 5#< 0$<
$A0%< $02&< $00'< 5 ' 0!!PRVAL 4 100!4 82 A .0!A .0#* 0$* 25%* 125&* $0A'*
5#< 0$< $A0%< $02&< $00'< 5 ' 0!' 127 .TPS !4 82 A .0!A .0#* 0$* 25%* 125&*
$0A'* 5#< 0$< $A0%< $02&< $00'< 5 ' 0!' 127 4 100!.HTEMP A .0!A .0#* 0$*
25%* 125&* $0A'* 5#< 0$< $A0%< $02&< $00'< 5 ' 0!' 127 4 100!4 82 ;FMOD !A
.0#* 0$* 25%* 125&* $0A'* 5#< 0$< $A0%< $02&< $00'< 5 ' 0!' 127 4 100!4 82
```

```
A .0!;SMOD #$SPEED$* 25%* 125&* $0A'* 5#< 0$< $A0%< $02&< $00'< 5 ' 0!'
127
```

3.7.2 Tabuľka R:

```
0$< .0%< .0&< .0'< 5 ' 0!' 127 4 100!4 82 A .0!A .0#* 125$* 0%$OXLFB&* .0'*
5#< 0$< .0%< .0&< .0'< 5 ' 0!' 127 4 100!4 82 A .0!A .0#* 125$* 0%*
.0&$blfbk'* 5#< 0$< .0%< .0&< .0'< 5 ' 0!' 127 4 100!4 82 A .0!A .0#* 125$*
0%* .0&* .0'$msecL#< 0$< .0%< .0&< .0'< 5 ' 0!' 127 4 100!4 82 A .0!A .0#*
125$* 0%* .0&* .0'* 5#6VSS $< .0%< .0&< .0'< 5 ' 0!' 127 4 100!4 82 A .0!A
.0#* 125$* 0%* .0&* .0'* 5#< 0$6IGADV%< .0&< .0'< 5 ' 0!' 127 4 100!4 82 A
.0!A .0#* 125$* 0%* .0&* .0'* 5#< 0$< .0%6OXRFB&< .0'< 5 ' 0!' 127 4 100!4
82 A .0!A .0#* 125$* 0%* .0&* .0'* 5#< 0$< .0%< .0&6brfbk'< 5 ' 0!' 127 4
100!4 82 A .0!A .0#* 125$* 0%* .0&* .0'* 5#< 0$< .0%< .0&< .0'6msecR ' 0!'
127 4 100!4 82 A .0!A .0#* 125$* 0%* .0&* .0'* 5#< 0$< .0%< .0&< .0'< 5
!SPEED!' 127 4 100!4 82 A .0!A .0#* 125$* 0%* .0&* .0'* 5#< 0$< .0%< .0&<
.0'< 5 ' 0!!PRVAL 4 100!4 82 A .0!A .0#* 125$* 0%* .0&* .0'* 5#< 0$< .0%<
.0&< .0'< 5 ' 0!' 127
```

3.7.3 Tabuľka W:

```
' 12.2 4- 128!4 82 A 0!A 0#* $00$* $A0%* $02&* $00'* $07#< $0A$<
$A0%6FAILC&< $00'< $02 ' 131!' 12.2 4- 128!4 82 A 0!A 0#* $00$* $A0%* $02&*
$00'* $07#< $0A$< $A0%< $02&6FAILD'< $02 ' 131!' 12.2 4- 128!4 82 A 0!A 0#*
$00$* $A0%* $02&* $00'* $07#< $0A$< $A0%< $02&< $00'6STAT9 ' 131!' 12.2 4- 128!4
82 A 0!A 0#* $00$* $A0%* $02&* $00'* $07#< $0A$< $A0%< $02&< $00'< $02 !
BATT!' 12.2 4- 128!4 82 A 0!A 0#* $00$* $A0%* $02&* $00'* $07#< $0A$< $A0%<
$02&< $00'< $02 ' 131!!BVOLT 4- 128!4 82 A 0!A 0#* $00$* $A0%* $02&* $00'*
$07#< $0A$< $A0%< $02&< $00'< $02 ' 131!' 12.2 .GNDER!4 82 A 0!A 0#* $00$*
$A0%* $02&* $00'* $07#< $0A$< $A0%< $02&< $00'< $02 ' 131!' 12.2 4- 128!.HTEMP A
0!A 0#* $00$* $A0%* $02&* $00'* $07#< $0A$< $A0%< $02&< $00'< $02 ' 131!' 12.2
4- 128!4 82 ;ADOXL!A 0#* $00$* $A0%* $02&* $00'* $07#< $0A$< $A0%< $02&<
$00'< $02 ' 131!' 12.2 4- 128!4
```

Tu môžme vidieť data troch tabuliek, ktoré riadiaca jednotka posiela. Data posielané sériovou linkou sú pre ostatné tabuľky, skoro totožné. Líšia sa len v názvoch veličín a ich hodnotách, odpovedajúcim daných dvojíc znakov. Pokiaľ si pozreme jednu z tabuliek R, D, W a data, ktoré sme si vypísali a budeme postupovať podľa už vyššie zmieneného postupu, získame potrebné veličiny a ich hodnoty z daných, vyšie vypísaných dat. Ako príklad som vypísal nami hladané data tabuľky D 3.5.1 z dat posielaných sériovou linkou .

3.8 Význam jednotlivých veličín

Význam väčšiny z nižšie vypísaných hodnôt veličín je na prvý pohľad jasný a není treba ho ďalej upresňovať. Akurát význam porúch a registrácia stavu motora nie je na prvý pohľad jasný, a preto jeho hodnoty $00 až $FF je treba vysvetliť a prípadne dekódovať.

SPEED	0 až 7500 ot./min.	Otáčky motoru
TPS	0 až 100 %	Poloha škrtiacej klapky
TPSAD	0 až 255	Poloha škrtiacej klapky - hodnota z A/D prevodníku
PRVAL	0 až 130	Tlak v sacom potrubí - hodnota čítača
PRESS	/2 kPa	Tlak v sacom potrubí
ADMAP	0 až 255	Tlak v sacom potrubí - hodnota z A/D prevodníku
BARO	/2 kPa	Barometrický tlak
ADBAR	0 až 255	Barometrický tlak - hodnota z A/D prevodníku
FMOD	-50 až 50 %	Korekcia paliva z klávesnice
SMOD	-50 až 50 %	Korekcia predstihu
VE	0 až 255	Tabuľková hodnota pre palivo
VE(f)	0 až 255	Zkorigovaná hodnota pre palivo
ADV1	0 až 60 °	Tabuľkový predstih
IGADV	0 až 60 °	Práve použitý predstih
ATEMP	-50 až 205 °C	Teplota nasávaného vzduchu
ADAIR	0 až 255	Teplota nasávaného vzduchu - hodnota z A/D prevodníku
OTEMP	-50 až 205 °C	Teplota oleja
ADOIL	0 až 255	Teplota oleja - hodnota z A/D prevodníku
FTEMP	-50 až 205 °C	Teplota paliva
RFUEL	0 až 255	Teplota paliva - hodnota z A/D prevodníku
SHMT1	-50 až 205 °C	Teplota hlavy č. 3
RHMT1	0 až 255	Teplota hlavy č. 3 - hodnota z A/D prevodníku
SHMT2	-50 až 205 °C	Teplota hlavy č. 4
RHMT2	0 až 255	Teplota hlavy č. 4 - hodnota z A/D prevodníku
HTEMP	-50 až 205 °C	Výsledná teplota hlavy
BVOLT	V	Elektrické napätie v systéme
BATT	0 až 255	Elektrické napätie v systéme - hodnota z A/D prevodníku
ADOXL	0 až 255	Lavá lambda sonda - hodnota z A/D prevodníku
ADOXR	0 až 255	Pravá lambda sonda - hodnota z A/D prevodníku
OXLER		Zpracované hodnoty z A/D prevodníku, ľavá lambda sonda
OXRER		Zpracované hodnoty z A/D prevodníku, pravá lambda sonda

OXLFB	-10 až 10 %	Korekcia od ľavej lambda sondy
OXRFB	-10 až 10 %	Korekcia od pravej lambda sondy
BLFBK	-10 až 10 %	Korekcia uložená v pamäti - ľavá
BRFBK	-10 až 10 %	Korekcia uložená v pamäti - pravá
MSECL	/10 ms	Doba vstreku - ľavá
MSECR	/10 ms	Doba vstreku - pravá
VSS	0 až 255	Rýchlosť automobilu
IDSET	$00 až $FF	Požadované volnobežné otáčky
IDLMS	0 až 255	Poloha volnobežného ventilu
STAT1		Registrácia stavu motoru
STAT2		Registrácia stavu motoru
STAT3		Registrácia stavu motoru
STAT4		Registrácia stavu motoru
STAT5		Registrácia stavu motoru
STAT6		Registrácia stavu motoru
STAT7		Registrácia stavu motoru
STAT8		Registrácia stavu motoru
STAT9		Registrácia stavu motoru
TTFLG		Registrácia stavu motoru
GNDER		Chyba ukostrenia
FAIL1	$00 až $FF	Indikované poruchy
FAIL2	$00 až $FF	Indikované poruchy
FAIL3	$00 až $FF	Indikované poruchy
FAIL4	$00 až $FF	Indikované poruchy
FAILA	$00 až $FF	V prevádzke vyskytnuté poruchy
FAILB	$00 až $FF	V prevádzke vyskytnuté poruchy
FAILC	$00 až $FF	V prevádzke vyskytnuté poruchy
FAILD	$00 až $FF	V prevádzke vyskytnuté poruchy
OXFLG	$00 až $FF	V prevádzke vyskytnuté poruchy
OXBF2	$00 až $FF	V prevádzke vyskytnuté poruchy

Kapitola 4

Význam hodnôt veličín indikujúcich poruchy

4.1 Dekódovanie závad

Chybový kód je dvojznakový(00 až FF) v šestnástkovej sústave. Prvý znak definuje súčet prvých štyroch číslic, a druhý znak definuje súčet druhých štyroch číslic v šestnástkovej sústave. V šestnástkovej sústave sú čísla 1 až 9 vyjadrené číslami, a čísla 10 až 15 vyjadrené písmenami A až F. Napríklad kód $A5 obsiahuje poruchy $20, $80, $04, $01 - 2+8+0+0 = A, 0+0+4+1 = 5 to je $A5.

4.2 Riadiaca jednotka vysiela dva typy chybových hlásení:

1. Chyby vyskytujúce sa v okamžiku kontroly fail1 až fail3.
2. Chyby registrované v priebehu chodu motora failA až failC (nemusí súhlasiť s nasledujúcim)
   Tieto chyby po odstránení príčiny ich vzniku môžme vymazať s pamäte riadiacej jednotky, pomocou vytiahnutia poistky č.1 na dobu cca 30s (vedľa hlavného relé v motorovom priestore).

4.3 Význam hodnôt

Fail1:	(FailA)
$80	Chyba v kontrolnom súčte
$40	
$20	Hodnota tlaku v sacom potrubí je vysoká
$10	Hodnota tlaku v sacom potrubí je nízká
$08	Napätie na snímači teploty vzduchu je vysoké
$04	Napätie na snímači teploty vzduchu je nízké
$02	TPS hodnota je príliš vysoká
$01	TPS hodnota je príliš nízká

Fail2:	(FailB)
$80	Napätie na snímači teploty paliva je vysoké
$40	Napätie na snímači teploty paliva je nízké
$20	Napätie na snímači teploty oleja je vysoké
$10	Napätie na snímači teploty oleja je nízké
$08	Napätie na snímači teploty hlavy 1 je vysoké
$04	Napätie na snímači teploty hlavy 1 je nízké
$02	Napätie na snímači teploty hlavy 2 je vysoké
$01	Napätie na snímači teploty hlavy 2 je nízké

Fail3:	(FailC)
$80	Pravá lambda sonda chudá po 3 s
$40	Ľavá lambda sonda chudá po 3 s
$20	Vada na pravé lambda sonde
$10	Vada na ľavej lambda sonde
$08	Žiadny synchronizační impulz po 3 impulzoch zo zotrvačníku
$04	Žiadny impulz zo zotrvačníku po synchronizačnom impulze
$02	Baro hodnota je vysoká
$01	Baro hodnota je nízká

Fail4:	(FailD)
$04	Palubné napätie prekročilo 20 V
$02	Palubné napätie prekročilo 20 V
$01	Palubné napätie prekročilo 20 V

Podľa predchádzajúcich špecifikácií sme si mohli vytvoriť predstavu o tom, čo všetko je riadiaca jednotka schopná posielať za údaje cez sériové pripojenie. Niektoré z veličín sa opakujú vo viacerých tabuľkách. Dôvod toho je ten, že každá tabuľka je určená pre odlišnú

kontrolu správnej funkcie motora a často opakujúcej sa veličiny, ako napríklad speed - otáčky, htemp - teplota hláv, sú pre správnu kontrolu nepostrádateľné.

4.4 Význam jednotlivých tabuliek:

D - Základné údaje

O - Obrazovka pre kontrolu voľnobehu

R - Základná obrazovka

Q - Obrazovka pre kontrolu snímačov

W - Obrazovka chyb

X - Obrazovka hodnôt A/D prevodníku a čítačov

E - Obrazovka kontroly lambda regulácie

Y - Obrazovka registrácie stavu motora

Kapitola 5

Vývoj aplikácie

5.1 Vývojové prostredie

Čo je pre mňa dôležité pri výbere vývojového prostredia, je programovací jazyk, v ktorom mám danú aplikáciu napísať a operačný systém, na ktorom daná aplikácia má pracovať. Oba tieto prvky hrajú základnú úlohu pri výbere vývojového prostredia. Na výsledný program je požiadavka, aby bol schopný pracovať na operačnom systéme Windows CE 2003, a aby bol napísaní v jazyku C. Z týchto podmienok som usúdil, že najlepšie vývojové prostredie je Embedded Visual C++ [4]. Toto vývojové prostredie podporuje jazyky C a C++. Ďalej to bol program ActiveSync na pripájanie a komunikáciu medzi PPC (Pocket PC)a desktopovým PC. A na testovanie vyvíjanej aplikácie počas jej písania som použil program SDK (software development kit) [4], ktorý dokáže simulovať operačný systém Windows CE 2003 a natívne spolupracuje s Embedded Visual C++ a ActiveSync [4]. Tieto vývojové aplikácie pochádzajú z firmy Microsoft [4].

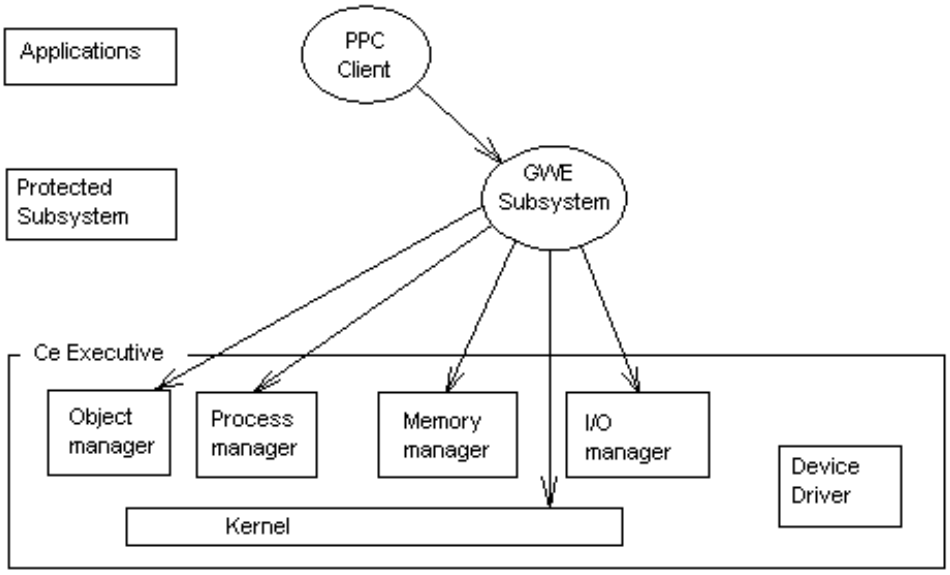
5.2 Operačný systém Windows CE 2003

Windows CE je malý, modulárný 32-bitový operačný systém, navrhnutý pre prácu na zariadeniach s podporou malej pamäte. Windows CE je veľmi podobný veľkému desktopovému operačnému systému Windows NT. Obsahuje väčšinu z užívateľského rozhrania Windows NT, čiže užívatelia sú už s nim dobre oboznámený, a to uľahčuje prácu vývojárom. Windows CE zariadenia majú úložisko dát riešené ako pamäť ROM (random access memory) a RAM (read-only memory), kde operačný systém Windows CE je uložený v ROM. Pre mňa ako vyvojára má veľký význam to, že Windows CE je takisto ako aj Windows NT postavený na Win32 API a to znamená, že vývojár, ktorý je oboznámený s programovaním na štandartných windows platformách ako napríklad Windows NT, môže začať programovať aplikácie s malým oboznámením do programovania vo Windows CE. Rozdiel oproti štandartnému desktopovému Windows NT je napríklad v podpore modulov pre programovanie aplikácií. Týchto funkcií je očosi menej ako vo Windows NT. Napríklad

kde Windows NT má podporu pre kreslenie grafických tvarov (bezier krivky, kruh ...), Windows CE už túto podporu nemá. Je to kôli šetreniu kapacity pamäte. Tieto funkcie musí vývojar nahradiť vlastnými alebo prispôsobiť aplikáciu podporovaným funkciám.

5.2.1 Architektúra Windows CE

- The kernel
- The Graphics, Windowing, and Event Subsystem (GWES)
- The OEM Adaptation Layer (OAL)
- The device driver layer
- The communication APIs
- Custom Shells and Internet Explorer



Obrázok 5.1: Windows CE Architecture

5.3 Vlastný vývoj systému

Postup pri začatí vyvíjania danej aplikácie bol nasledovný. V prvom rade som si nainštaloval Embedded Visual C++ [4], ten si vyžiadal nainštalovať ActiveSync [4] pre komunikáciu s

PPC. Po spustení Visual C++ som zistil, že operačný systém, na ktorom potrebujem vyvíjať aplikáciu nie je podporovaný. Samozrejme, ako pri väčšine som nazrel na internet, na stránkach msdn som našiel podporu pre potrebný systém (SDK for Windows Mobile 2003-based Pocket PCs) [4]. Takže potrebné aplikácie pre vývoj máme a môžme začať písať zdrojový kód našej aplikácie. Začal som u jednoduchých veciach ako napríklad vytvorenie jednoduchého okna a napísať doň jednouchý text napr ,,Hello world". Ešte pred skompilovaním je dobre si skontrolovať či je dobre nastavný systém, pre ktorý sa má aplikácia kompilovať. Týmto jednoduchým programom som vlastne otestoval potrebné aplikácie na vývoj.

Ďalším mojim krokom, bolo navrhnúť aspoň základnú štruktúru písania kódu aplikácie. Kedže máme náš jazyk zadaný ako C, nemôžeme vytvárať triedy a moduly ako napr. v C++. Takže som si poradil ináč, kedže bude v aplikácií potrebné použiť globálne premenné, ktoré budú uchovávať informácie potrebné počas celého behu aplikácií, som si vytvoril súbor, v ktorom ich mám nadeklarované a vytvorené funkcie na prístup k nim. Čiže nakoniec náš súbor vypadá ako trieda, ktorá má svoje privátne premenné a funkcie
(moduly), pomocou ktorých k nim pristupuje. Ďalším súborom, ktorý som si vytvoril bol súbor pre pomocné funkcie ako napr. pre správne vytvorenie zložitejších datových štruktúr. Je vňom napr. vytvorenie nového fontu písma, ďalej nastavenie práce so sériovým rozhraním atď.. Tie dôležitejšie budú prebrané v ďalších kapitolach. Ďalej mám už len základné súbory na spracovanie windowsových správ, ciže CALLBACK funkcie.

Tak, súborovú štruktúru aplikácie máme hotovú. Môžme začať písať kód našej aplikácie. Ako prvé, čo som si zvolil a naprogramoval, bolo vytvorenie sériového pripojenia. Kedže Windows CE ako aj naša aplikácia beži na Win32API, našiel som si potrebnú funkciu na internetových stránkach msdn, kde bola pekne popísaná a aj vysvetlená. Podrobnejšie si otvorenie a nastavenie seriového portu (jeho funkcie a datové štruktúry) popíšeme neskôr. Tieto informácie som čerpal zo zdrojov [5] [1] [1].

Po zistení, že komunikácia po sériovej linke funguje, som mohol začať s filtrovaním potrebných dat od nami nepotrebných. Túto úlohu zaisťujem pomocou jedného z dvoch timerov. Ten sa spúšťa po 50ms a vňom prebieha for ciklus 60x, čiže daný limit seriovej komunikácie (9600bit/s) bude splnení. Ďalší, druhý taimer používam pre vykreslovanie hodnôt na plátno (okno) aplikácie. Ten sa spúšťa už len každých 150ms, čo by malo stačiť na zobrazovanie hodnôt z riadiacej jednotky.

5.4 Vytvorenie sériového pripojenia

V tejto sekcii preberiem podrobnejšie vytvorenie spojenia medzi zariadeniami za požitia sériového rozhrania. Sekcia je rozdelená podľa programového použitia : otvorenie portu,

nastavenie portu, zápis a čítanie dat a uzavretie portu. Win API používa pre prácu s vstupno výstupným sériovým zariadením tie isté funkcie ako pre prácu so súbormi. Čo nám dosť zjednoduší prácu. Čiže napríklad pre otvorenie sériového portu stačí zadať do funkcie CreateFile() názov nami zvoleného portu, a port sa otvorí pre príjmanie a odosielanie.

5.4.1 Otvorenie portu

Na otvorenie sériového portu sa používa už vyžšie zmienená funkcia CreateFile. Ako hlavný parameter je názov portu. Tento názov mám nastavený na štandartný ako napr. COM0 - COM9. Pretože užívateľ si môže vytvoriť vlastný draiver na seriové rozhranie, je nutné aby mu dal štandartný názov COM0 - COM9, aby ho mohol používať v tejto aplikácií.

```
// Otvorenie sériového portu
  hPort = CreateFile  (lpszPortName,  // Odkaz na názov portu
                       GENERIC_READ | GENERIC_WRITE,
                                      // prístup (čítanie/zápis)
                       0,             // zdielanie
                       NULL,
                       OPEN_EXISTING, // Spôsob otvorenia
                       0,             // Windows CE očakava 0
                       NULL);
```

Funkcia vráti ERROR_FILE_NOT_FOUND ak sa port nepodaril otvoriť. Ak sa podarí otvoriť port, funkcia vráti ukazateľa na daný port.

5.4.2 Nastavenie portu

Po úspešnom otvorení portu na sériové rozhranie, je dobré ho nastaviť na požadované parametre. Pretože defaultné nastavenie nám nemusí vyhovovať. Postup na nastavenie otvoreného portu je jednoduché špecifikovanie datovej štruktúry DCB (PortDCB) a nasledné použitie fukcie SetCommState, ktorá nastaví daný port na nami špecifikovanú štruktúru. Takisto na získanie nastavenia je funkcia GetCommState, čiže pred zmenou nastavenia si môžme nahrať aktuálne nastavenie.

```
// Štruktúra nastavenia sériového portu
PortDCB.BaudRate = 9600;              // Nastavenie rýchlosti prenosu
PortDCB.fBinary = TRUE;               // Binárny mod, nekontroluje sa EOF
PortDCB.fParity = TRUE;               // Kontrola paryty
PortDCB.fOutxCtsFlow = FALSE;
PortDCB.fOutxDsrFlow = FALSE;
PortDCB.fDtrControl = DTR_CONTROL_ENABLE;
PortDCB.fDsrSensitivity = FALSE;
```

```
PortDCB.fTXContinueOnXoff = TRUE;     // Špecifikuje chovanie prenosu
PortDCB.fOutX = FALSE;
PortDCB.fInX = FALSE;
PortDCB.fErrorChar = FALSE;
PortDCB.fNull = FALSE;
PortDCB.fRtsControl = RTS_CONTROL_ENABLE;
PortDCB.fAbortOnError = FALSE;
PortDCB.ByteSize = 8;                 // Počet bit/byte
PortDCB.Parity = NOPARITY;
PortDCB.StopBits = ONESTOPBIT;
```

Po vytvorení datovej štruktúry DCB, ju nahráme pomocou SetCommState(hPort, &PortDCB). Kde hPort je ukazateľ na nami otvorený port a PortDCB je štruktúra DCB, podľa ktorej sa daný port nastaví. Ak sa nám podarí nahrať naše nastavenie, funkcie SetCommState vráti nenulovú hodnotu, ináč nulu.

5.4.3 Nastavenie časovania

Po nastavení COM portu, musíme nastaviť časovanie na komunikáciu s portom. Časovanie sa tiež nastavuje pomocou datovej štruktúry a to COMMTIMEOUTS, ak túto datovú štruktúru nenakonfigurujeme, nastaví sa defaultná konfigurácia. Tá nám s veľkov pravdepodobnosťou nebude vyhovovať, preto pri tvorbe každej aplikácie by sa mala nadefinovať. Na získanie a nastavenie časovania pri komunikácií, sú nám k dispozícií funkcie GetCommTimeouts, SetCommTimeouts. Čiže pri konfigurácií si môžme načítať aktuálnu konfiguráciu a zmeniť len potrebné položky a znova nahrať. Časové hodnoty sú v milisekundách.

```
// max. čas medzi prijatím dvoch znakov
timeouts.ReadIntervalTimeout=0;
// násobok, špecifikujúci totálny čas na jedno čítanie
timeouts.ReadTotalTimeoutConstant=1;
// konštanta v ms, ktorá určuje totálny čas na čítanie
timeouts.ReadTotalTimeoutMultiplier=1;
// násobok, špecifikújúci totálny čas na jeden zápis
timeouts.WriteTotalTimeoutConstant=1;
// konštanta v ms, ktorá určuje totalný čas na zápis
timeouts.WriteTotalTimeoutMultiplier=1;
```

Funkcia pre nahratie našeho nastavenia SetCommTimeouts vráti pri úspešnom nahraní nášho nastavenia nenulovú hodnotu, ináč nulu.

5.4.4 Príjmanie a odosielanie dat

Príjmanie a odosielanie dat je podobné ako zapisovanie alebo čítanie zo súborov. Aj funkcie sa používajú tie isté, WriteFile a ReadFile. Akurát namiesto názvu súboru, vložíme ako parameter ukazovateľ na otvorený port. Parametre týchto funkcií sú väčšine vývojárov známe, preto ich tu nebudem rozoberať. Takisto vracajúca nenulová hodnota značí úspech a nulová hodnota neúspech.

5.4.5 Uzavretie sériového portu

Náš otvorený port, pri ukončení práce treba uvólniť. Ak necháme port ďalej po ukončení otvorený, nebude sa dať ďalej používať. Preto má uvolnenie dôležitý význam. Vo vyvíjanej aplikácií je potreba uvolňovať port pri dvoch udalostiach, a to pri ukončení aplikácie a pri zmene nastavenia portu. Na uzavretie portu sa používa funkcia CloseHandle. Pri úspechu/neúspechu vracia štandartné hodnoty nenulová/nulová, pre zistenie typu chyby treba použiť funkciu GetLastError, ktorá vráti kód chyby

5.5 Filtrovanie dat zo sériovej linky

Oddelenie požadovaných dat (názvy veličín a ich hodnoty) z posielaných dat sériovou linkou sa prevádza vo funkcii taimeru. Taimer je nastavený na 50ms. Vo funkcii timeru je ciklus for, ktorý sa vykonáva 60x, spolu s nastavením timeru na 50ms, by funkcia mala stíhať komuniakáciu 9600 bit/s ($20 * 60 * 8 = 9600$).

```
for(i=0; i < 60; i++)
  {
    if(Wait5)
    {
      for(j=0; j<5; j++)
      {
        value[j] = tolower(readFromCom());
      }
      if( Typ5 == false)
      {
        VarOK = checkVariable5(value, poz);
      } else if(Typ5 == true && VarOK)
      {
        insertValue(value, poz);
        Typ5 = false;
      }
    }
```

```
    buf = readFromCom();
    if( buf > 40)
    {
      typ1 = typ2;
      typ2 = buf;

      ibuf = checkVariable(typ1, typ2);
      if( (ibuf >=0 && ibuf < 16) ||
          ((Typ5 = checkValue(typ1, typ2, poz)) != false) )
      {
        poz = ibuf;
        Wait5 = true;
      }
      else
      {
        Wait5 = false;
      }
    }
  }
```

Funkcia časovača je rozdelená do dvoch častí.

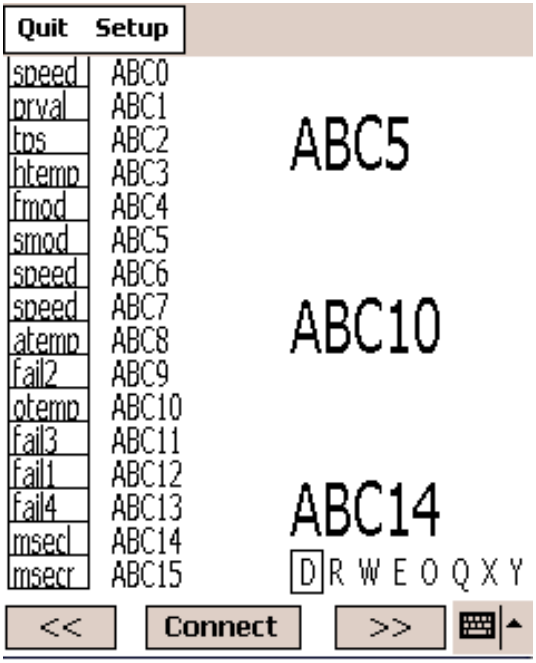
Prvá časť sa vykoná, ak boli nájdené dva špecifické znaky. Následne sa načítajú znaky (5 znakov), ktoré obsahujú buď názov veličiny alebo jej hodnotu. Pokiaľ premenná Typ5 je false, jedná sa o názov veličiny. Tá sa skontroluje pomocou funkcie checkVariable5, kde sa porovná so správnym názvom, ktoré je uložené v poli. Ak porovnanie uspelo, nastaví sa hodnota premennej VarOK na true. Pri načítaní dvojíc znakov, ktoré udávajú súradnice hodnoty veličiny sa nastavia premenné VarOK, Typ5 a Wait5. Tým sa povolí načítanie hodnoty do poľa.

Druhá časť funkcie časovača nastavuje premenné Typ5, Wait5 a ibuf, podľa načítaných špecifických znakov (dvojci znakov). Kontrola dvojce znakov prebieha pomocou funkcie checkVariable, ktorá vráti aj pozíciu daných znakov v tabuľke. Skontrolovanie či dvojca znakov znamená hodnotu veličiny sa prevádza pomocou funkcie checkValue, nastaví sa premenná Typ5 na true. Čiže máme jeden časovač pre všetky tabuľky, ktorý nam odfiltruje potrebné data (hodnoty a názvy) z dát posielaných sériovou linkou.

5.6 Grafické prostredie aplikácie

Uživateľské grafické rozhranie je navrhnuté čo najjednoduchšie. Obsahuje základne ovládacie prvky na zmenu príjmaných hodnôt. Tlačítko ,,Connection", pomocou ktorého začne príjmať iné tabuľky ako D (zaslanie znakov ,,yy"). Ďalej sú tam tlačítka ,,<<'' ,,>>'' ktoré slúžia

na zmenu príjmaných tabuliek. Väčinou je tvorená pomocou funkcií DrawText a DrawRectangle. Tlačítka sú tvorené pomocou funkcií CreateWindow. Tieto funkcie sú štandartné a vývojárom známe, čiže ďalšie rozpisovanie neni nutné. Na výpis názvov a hodnôt veličín som si vytvoril vlastné fonty. Jedná sa znova o vytvorenie a správne naplnenie datovej štruktúry, a to štruktúry LOGFONT. Táto štruktúra je len logická a nedá sa použiť na vykreslovanie textu. Na vytvorenie použiteľného fontu nám slúži funkcia CreateFontIndirect, a na zmenu fontu používam funkciu initiateText. Text sa na plátno aplikácie vykresluje pomocou časovača, ktorý prevádza vyplnenie plátna bielou farbou a následné vykreslenie hodnôt. Časovač sa spúšťa jedenkrát za 150ms. Obrazok 5.2 zobrazuje okno aplikácie v stave príjmania hodnôt z tabuľky D, text `ABC<číslo>` znázorňuje príjmanú hodnotu. Večšina týchto informácií je čerpaná z [2] [1].



Obrázok 5.2: Schéma umiestnenia snímačov na motore

Kapitola 6

Záver

Úlohou mojej bakalárskej práce bolo vytvoriť aplikáciu na zobrazovanie údajov zo sériovej linky. Myslím si, že túto úlohu som splnil a aplikácia spĺňa základné požiadavky mojej práce. Napriek tomu by do terajšej verzii aplikácie bolo dobré doprogramovať dalšie funkcie pre prácu s datami, ako napríklad zobrazovanie grafov, hlásení o chybe a prekročených medzných hodnôt. Tiež by bolo dobré pri vykreslovaní hodnôt do plátna použiť double-buffer a nie timer. Napriek týmto nedostatkom program splňuje dané kritéria bakalárskej práce.

Počas mojej práce som sa zoznámil s programovaním aplikácií na platformu Windows CE 2003, konkrétne vytváraním užívateľského prostredia a to nie len pomocou grafického prostredia vo vývojovej aplikácii eMbedded Visual C++ ale tiež pomocou funkcií CreateWindow. Pri programovaní aplikácie som na internete našiel program Message Cracker Wizard for Win32 SDK Developers, ktorý vytvára pre jednotlivé windowsácke správy, makra a tým sprehľadňuje ich správu. Čiže po použití som mal vo funkcii správy windosových správ jeden jednoduchý switch, v ktorom som mal makra vytvorené už zmieneným programom. Pri vývoji som použil zdroje z internetu [3] [4] [2] a knižné publikácie [5] [1].

Taktiež si myslím, že v dnešnej dobe automobilov s motormi riadenými mikroprocesormi je vývoj a študovanie takýchto programových systémov je nezbytný a v budúcnosti sa stanú pri kontrólach alebo opravách, nielen motorov, hlavnýn zdrojom informácií pre automechanikov. S nástupom nových technológií ako je wi-fi alebo blutooth nebudú už data posielané do PC len po sériovej linke, ale aj prostredníctvom práve týchto technológií. Preto bude pre vývojarov takýchto programových systémov dôležité štúdium kumunikačných protokolov, ktoré tieto nové technológie využívajú.

Kapitola 7

Príloha - uživateľská príručka

Po spustení aplikácie uvidíte na obrazovke oznámenie ,,Not Connect" a tri tlačítka ,,<<'' ,,Connect'' ,,>>'', ktoré zatiaľ nefungujú. Keď sa premiestnime na obrazovku nastavenia, pomocou tlačítka ,,Setup" v lište aplikácií, dostaneme sa do okna nastavenia, v tomto okne je len combobox pre výber sériového portu. Po výbere a otvorení sériového portu, stlačte tlačítko Done. Aplikácia sa vráti do prvého okna, kde už sú vypísané názvy aj hodnoty veličín a takisto označenie tabuliek. V tomto stave aplikácia už príjma hodnoty veličín z tabuľky D. Pre príjmanie iných tabuliek musíte použiť tlačítko Connect, ktoré nám nastaví aplikáciu pre príjmanie tabuľky R. Pre príjmanie ostatných tabuliek okrem tabuľky D použite tlačítka ,,<<'' ,,>>''. V pravom dolnom rohu nad tlačítkami sú zobrazené názvy tabuliek a aktuálna tabuľka je zvýraznená. Všetkých šestnásť názvov a hodnôt veličín zobrazených v ľavej časti aplikácie, slúži pre výber hlavného zobrazenia. V hlavnom zobrazení sú len tri hodnoty, ktoré sú zobrazené vo väčšom fonte.

Literatura

[1] Ph.D. Bruce E. Krell. *Pocket PC Developer's Guide*. Osborne, 2002. ISBN 0-07-213150-0.

[2] Robert Burdick. *Essential Windows CE Application Programming*. Wiley Computer Publishing, John Wiley & Sons, Inc., 1999. ISBN 0471327476.

[3] Microsoft. Msdn library, serial communications in win32. `http://msdn.microsoft.com`.

[4] Microsoft. Vývojové aplikácie. `http://www.microsoft.com`.

[5] Kopřivnice Tatra a.s. *Dílenská příručka pro opravy a seřizování motorů Tatra 613-4i KAT*. Tatra a.s., Kopřivnice, 1994.