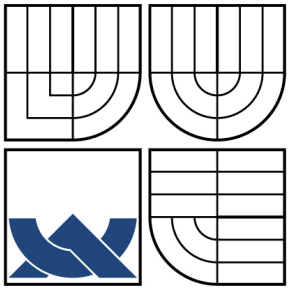
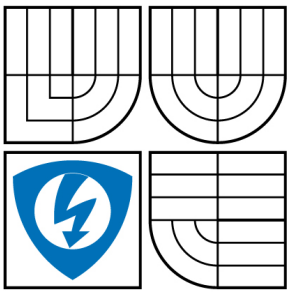


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF CONTROL AND INSTRUMENTATION

ŘÍZENÍ POHYBU VOZÍTKA

SMALL VEHICLE MOTION CONTROL

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

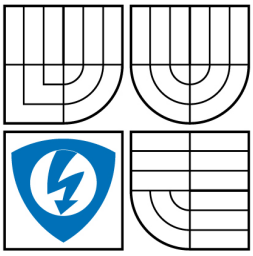
AUTOR PRÁCE — Bc. ZDENĚK ŠTARK

AUTHOR

VEDOUCÍ PRÁCE — Ing. TOMÁŠ MACHO, Ph.D.

SUPERVISOR

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Diplomová práce

magisterský navazující studijní obor
Kybernetika, automatizace a měření

Student:	Bc. Zdeněk Štark	***ID:***	47042
Ročník:	2	***Akademický rok:***	2008/2009

NÁZEV TÉMATU:

Řízení pohybu vozítka

POKYNY PRO VYPRACOVÁNÍ:

1. Seznamte se s problematikou řízení podvozku vozítka pomocí dvou stejnosměrných motorků.
2. Navrhněte hardwarové řešení řízení pohybu vozítka se dvěma stejnosměrnými motorky a kolovým podvozkem.
3. Vypracujte algoritmy, které umožní, aby si vozítko zapamatovalo projetou trajektorii (při ručním řízení) a průjezd trajektorií bylo schopno zopakovat. Ošetřete havarijní stavy (např. vložení překážky do trajektorie).
4. Řešte komunikaci mezi vozítkem a počítačem PC.

DOPORUČENÁ LITERATURA:

Dle vlastního literárního průzkumu a doporučení vedoucího práce.

Termín zadání:	9.2.2009	***Termín odevzdání:***	25.5.2009

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

prof. Ing. Pavel Jura, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Anotace

V rámci diplomové práce bylo navrženo a sestrojeno jednoduché vozítko poháněné diferenciálním podvozkem. Vozítko je ručně řízeno pomocí uživatelského programu z osobního počítače PC. Programové vybavení umožňuje zaznamenat trajektorii projetou vozítkem a následně ji automaticky zopakovat. Řídicí systém vozítka je založen na mikrokontroléru řady ATmega16. Komunikace mezi mikrokontrolérem a počítačem PC probíhá po sběrnici RS232. Vozítko je vybaveno digitálním kompasem a pěticí reflexních čidel, které slouží k detekci překážek v blízkém okolí vozítka. Napájecí jednotku celého vozítka tvoří sedm NiCd článků s kapacitou 500mAh. Pohonná jednotka je napájena přímo z NiCd článků. Napětí 5 V pro napájení řídicí elektroniky je získáváno vytvořeným spínaným zdrojem připojeným k NiCd článkům.

Klíčová slova

Diferenciální podvozek, ATmega16, navigace, reflexní čidlo.

Annotation

In this thesis, a simple small vehicle powered by differential chassis was projected and constructed. The vehicle is controlled manually with the help of user program on personal computer (PC). The software equipment allows recording trajectory run by the small vehicle and then repeating it automatically. The operation system of the small vehicle is based on microcontroller ATmega16 series. Communication between the microcontroller and PC is carried out through RS232 serial bus. The vehicle has digital compass and five reflex sensors that serve for obstruction detection in the closest vicinity of the vehicle. Power unit of the vehicle consists of seven NiCd cells with the capacity of 500 mAh. Propellant unit is powered directly from NiCd cells. Voltage of 5 V for feeding of the control electronics is gained through the set up clasp source of voltage connected to NiCd cells.

Key words

Differential chassis, ATmega16, navigation, reflex sensor.

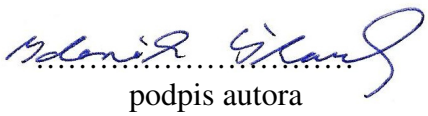
ŠTARK, Z. *Řízení pohybu vozítka.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 44 s. Vedoucí diplomové práce Ing. Tomáš Macho, Ph.D.

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma Řízení pohybu vozítka jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

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

V Brně dne: **25. května 2009**

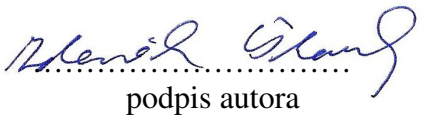


podpis autora

Poděkování

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

V Brně dne: **25. května 2009**



.....
podpis autora

Obsah:

1. ÚVOD

Oblast robotiky sdružuje do sebe mnoho technických oborů. Je to jedna z oblastí, která se v poslední době velice rozvíjí a zahrnuje v sobě veškeré autonomní roboty od obráběcích strojů přes manipulátory až po vesmírné sondy. Velká část robotiky patří vojenským a záchranným složkám, kde se jedná většinou o roboty s malou autonomností a které jsou řízené na dálku operátorem. Robot je schopen se pohybovat v prostředí nebezpečném pro člověka a může operátorovi zprostředkovávat informace (teplota, vlhkost, radiace, obrazový signál) z místa, kde se nachází.

Tuto práci jsem si vybral, protože chci ukázat, že i s jednodušími prostředky je možné vytvořit částečně autonomní vozítko schopné vykonávat požadované operace. Spojením mikroprocesorové a měřicí techniky chci vytvořit jeden celek, který bude splňovat zadaný úkol.

2. NAVIGACE A ZJIŠŤOVÁNÍ POLOHY

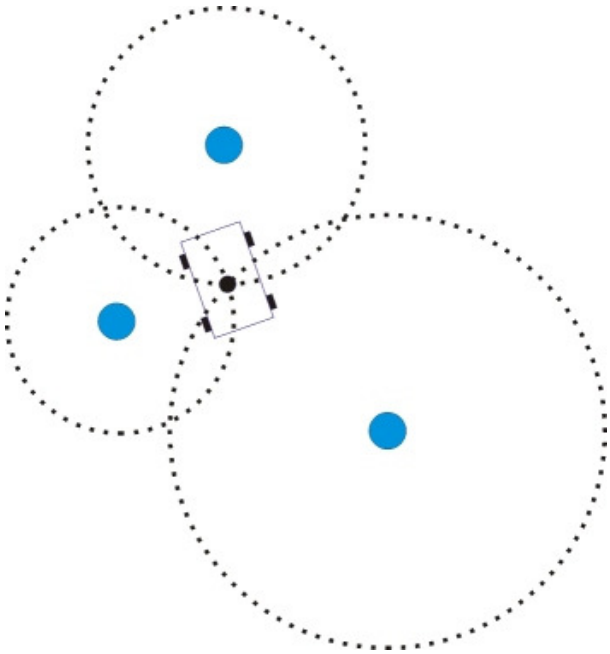
Zjišťování aktuální polohy jakéhokoli robota ať už v prostoru, nebo v ploše je jedna z nejdůležitějších a současně nejsložitějších úkolů robotiky. Je mnoho způsobů, jak polohu robota určit, ale žádná z metod není univerzální a hodí se pouze pro určitou oblast použití. Navíc každá metoda je zatížena jistou chybou a změřená nebo vypočtená poloha robota není nikdy přesná.

2.1 NAVIGACE VÝPOČTEM (DEAD RECKONING)

Tato metoda odhaduje polohu robota ze získaných časových průběhů otáčení jednotlivých motorů, nebo relativních změn (nejčastěji zrychlení). Tato metoda vyžaduje znalost výchozího bodu a úhlu, pod kterým bylo z počátku „vyjeto“. Metoda je velice často používaná u diferenciálních a pásových podvozků. Odometrie je relativně jednoduchá na výpočet, ale oproti ostatním metodám je zatížena velkou integrační chybou.

2.2 TRIANGULAČNÍ METODA

Tato metoda zjištění polohy je založena na měření intenzity signálu od několika vysílačů. Z úrovní jednotlivých signálů je vypočtena přibližná vzdálenost detektoru od vysílače a je „opsána“ pomyslná kružnice. V místě nebo oblasti, kde se jednotlivé kružnice protínají, je pozice robota (viz Obr. 1). Tato metoda ale není primárně schopna určit natočení robota. Tento nedostatek je možné vyřešit například použitím digitálního kompasu nebo směrovým detektorem.



Obr. 1– Triangulační metoda - měření vzdáleností

Další triangulační metoda je založena na měření úhlů jednotlivých detektorů. Otočnou směrovou anténou se zjistí jednotlivé úhly mezi robotem a dvěma různými majáky.

Při použití této metody pro zjišťování polohy je možné na stejných „majácích“ navigovat libovolné množství robotů. Přesnost lokalizace je závislá na přesnosti vyhodnocení intenzity jednotlivých signálů, ale oproti metodě „dead reckoning“ není téměř vůbec zatížena integrační chybou. Pokud se robot pohybuje v prostoru mezi majáky, je určení polohy dostatečně přesné, ale v případě, kdy je robot od majáků vzdálen a poloměry kružnic začínají splývat, tato metoda selhává.

2.3 GPS

Global position systém (GPS) je navigační systém pro určování polohy kdekoliv na zemském povrchu bez ohledu na povětrnostní podmínky. Tento systém je vyvíjen od roku 1973 Ministerstvem obrany Spojených států a v devadesátých letech byl tento systém dostupný i pro civilní účely.

Jádro celého navigačního systému tvoří 24 navigačních družic, obíhajících zeměkouli na šesti různých, velmi přesných drahách, se sklonem 55 stupňů vzhledem k rovníku. Dráhy družic jsou vypočteny tak, aby v daném okamžiku byl v kterémkoliv místě na zeměkouli viditelný nad obzorem dostatečný počet družic, nezbytný k přesnému zaměření.

Přesnost zaměření polohy je dána principem šíření a zpracování přesných časových signálů, které družice nepřetržitě vysílají na kmitočtech okolo 1,57GHz.

GPS pro použití v „InDoor“ robotice nemá velký význam, ale pro navigaci venkovních robotů je velice výhodná.

2.4 SLEDOVÁNÍ VODICÍ ČÁRY

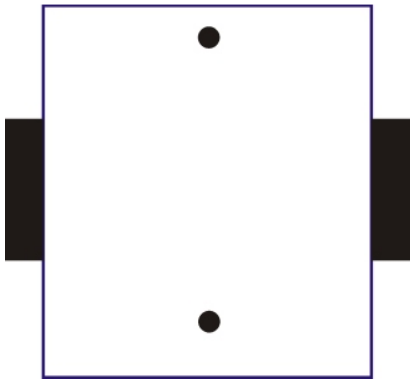
Sledování vodicí čáry (guidepath following) je vysoce spolehlivá metoda pro navigování robota. Robot pomocí optických nebo magnetických senzorů sleduje „vodicí čáru“. Pro svou spolehlivost je tento typ navigace velice často používán v průmyslu (skladové a transportní roboty). V průmyslu je častější magnetická „čára“, neboť barevné pruhy na zemi mohou být v místech znečištěné, nebo dokonce vůbec nerozeznatelné od podkladu.

Navigace a vyhodnocení řízení robota je naprosto stejné u obou typů. Na přídi je několik senzorů a při vychýlení robota, nebo v místech, kde dojde k zatáčení, vyhodnotí vyšší úroveň signálu jedno z krajních čidel a podle toho je proveden zásah do řízení.

3. TYPY PODVOZKŮ

3.1 DIFERENCIÁLNÍ PODVOZEK

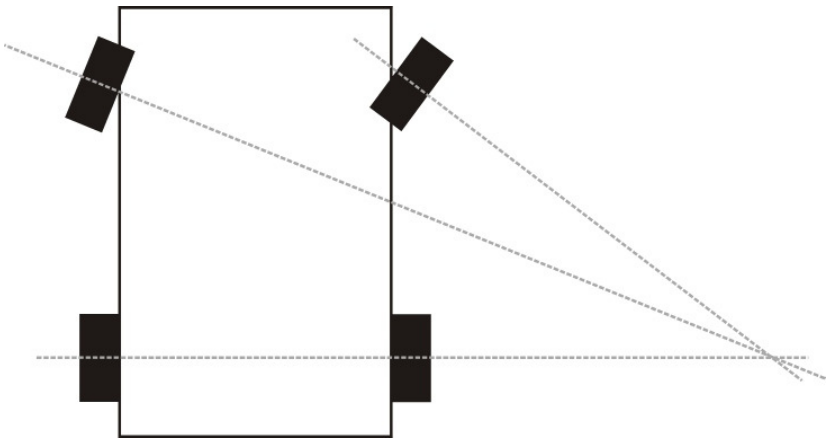
Diferenciální podvozek je nejpoužívanějším typem u „vývojových“ robotů. Skládá se pouze ze dvou hnacích kol, k nimž je ještě použit jeden nebo dva opěrné body pro udržení rovnováhy (viz Obr. 2). U takového podvozku je velice jednoduché vypočítat polohu metodou „dead reckogning“. Přesnost stanovení polohy se odvíjí od chyby při počítání jednotlivých kroků a vůli v převodech.



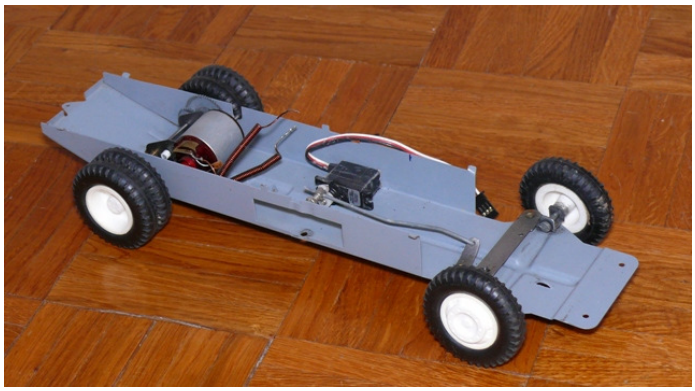
Obr. 2 – Diferenciální podvozek

3.2 ACKERMANŮV PODVOZEK

Tento typ podvozků je použit v automobilovém průmyslu. Skládá se ze dvou náprav, jedna z náprav slouží pro řízení a druhá je hnací, případně je použita jejich kombinace. Navigace takového podvozku je složitější než u diferenciálního, ale Ackermanův podvozek vykazuje vyšší stabilitu a je schopen překonat členitější terén. U konstrukčního uspořádání je pouze jediná velice důležitá podmínka správného natáčení řídicích kol. Při zatáčení musí být úhel kol rozdílný a musí splňovat podmínku, že se osy kol protnou v jednom bodě (viz Obr. 3). Ackermanův podvozek má menší manévrovací schopnosti než předešlý typ.



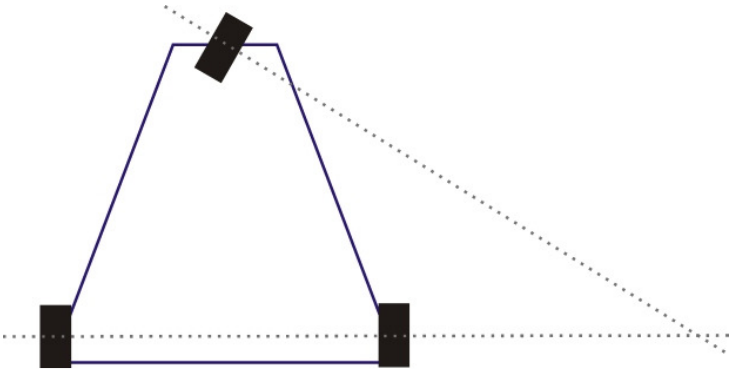
Obr. 3 – Ackermanův podvozek (výchylka řídicí nápravy)



Obr. 4 – Ackermanův podvozek (zadní náprava hnací a přední řídicí)

3.3 TROJKOLOVÝ PODVOZEK

Trojkolový podvozek je principielně totožný s Ackermanovým, ale v případě, že je jedno kolo řídicí a zbylé dvě hnací, odpadá komplikace s různým úhlem natočení přední nápravy, proto je i jednodušší vypočítat polohu takového podvozku z časových průběhů rychlosti a úhlu natočení řídicího kola.



Obr. 5 – Trojkolový podvozek

3.4 PODVOZKY S VŠESMĚROVÝMI KOLY

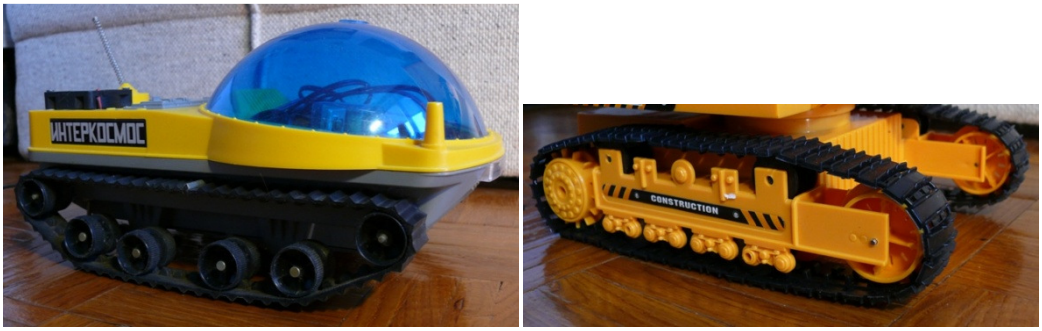
Podvozky s všesměrovými koly se mohou pohybovat libovolným směrem. Uspořádání těchto podvozku bývá nejčastěji v rovnostranném trojúhelníku. Oproti ostatním typům má nejvyšší manévrovací schopnosti, což je vyváženo daleko větší náročností řídících algoritmů.

Obr. 6 – Všesměrové kolo

3.5 PÁSOVÉ PODVOZKY

Pásové podvozky jsou principiálně stejné jako diferenciální podvozky, ale při otáčení dochází ke smýkání pásů, a proto může dojít k nepřesnostem při navigaci pomocí „dead reckoning“.

Oproti všem typům podvozků mají pásové podvozky výborné vlastnosti při jízdě na nerovném povrchu i v terénu s poměrně velkým stoupáním. Jejich výhody jsou vykoupené vyšší spotřebou energie hlavně při zatáčení, kdy dochází ke smýkání pásů.



Obr. 7 – Různé typy pásových podvozků

3.6 KRÁČEJÍCÍ PODVOZKY

Kráčející podvozky jsou nejsložitější a nejkomplikovanější „podvozky“ robotů. Jedná se o napodobeniny pavouků a v případě „dvounohých“ robotů o napodobení člověka. Oproti jejich živým předlohám mají jen několik stupňů volnosti a jejich pohyby jsou velice omezené. Složitost řídících algoritmů a počet stupňů volnosti noh velice ovlivňuje pohyblivost celého robotu. Velkou výhodou tohoto typu podvozku jsou výborné vlastnosti při pohybu ve skalnatém a kopcovitém terénu.

Obr. 8 – Šestinohý typ kráčejícího podvozku - převzato z [6]

4. POHONY PODVOZKŮ

4.1 STEJNOSMĚRNÉ MOTORY

Stejnosměrný motor s permanentním magnetem je nejčastějším typem pro pohon mobilních robotů. Jeho výhodou je velice příznivý poměr výkon/hmotnost a poměrně jednoduché řízení otáček. Výrobci těchto motorů nabízejí širokou škálu nejrůznějších fyzických a výkonnostních provedení. Nevýhodou je složitější a dražší rychlostní a polohové řízení na rozdíl od krokových motorů. Bývá často zdrojem elektromagnetického rušení kvůli jiskření na komutátoru.

Pracovní oblast těchto motorů je v relativně vysokých otáčkách při nízkém momentu, což není příliš výhodné pro pohon robotů, ale při použití převodovky s velkým převodovým poměrem se z tohoto motoru stává velice vhodná pohonná jednotka. Převodovka může být již součástí pouzdra motoru a tvořit s ním tak jeden kompaktní celek. Součástí tohoto celku může být i inkrementální čidlo pro snímání otáček motoru.

Obr. 9 – Stejnosměrné motory Robbe a Graupner

4.2 STŘÍDAVÉ MOTORY

Na rozdíl od stejnosměrných motorů jsou střídavé motory dražší a jejich regulace mnohonásobně složitější (nutnost konstrukce fázových měničů). Jejich nevýhody jsou kompenzované vyšší účinností, nižší hmotností, větším výkonem (při stejné velikosti motoru) a jejich největší výhodou je nepřítomnost mechanického komutátoru. Tím odpadá vznik elektromagnetického rušení a motor je naprosto bezúdržbový.

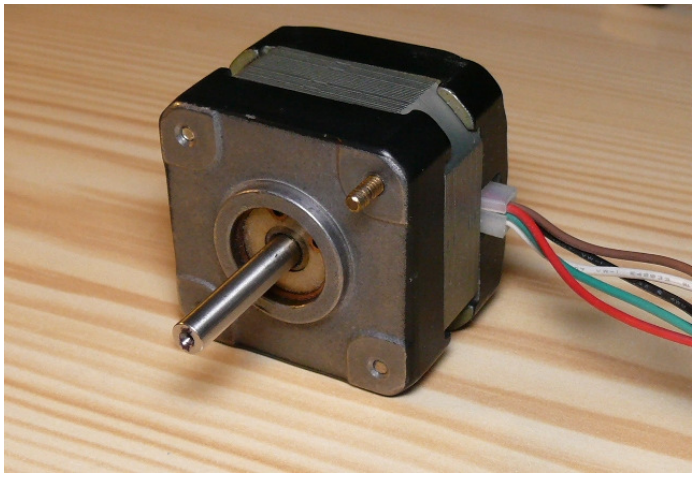
Obr. 10 – Střídavý motor s převodovkou

4.3 KROKOVÉ MOTORY

Krokové motory jsou převážně motory menších výkonů, určené hlavně k přesnému polohování. Jejich chod není spojitý, ale děje se po malých krocích. Velikost kroku závisí na fyzickém uspořádání jeho rotoru a statoru. Velikost kroku je v jednotkách stupňů, což je pro polohování velice přesné, a tedy předurčuje jeho největší využití. Momenty krokových motorů se pohybují v jednotkách N*m, a proto nejsou vhodné jako pohony větších a těžších robotů, ale je možné je použít pro natáčení řídicí nápravy, nebo pro otáčení některých senzorických hlavic robotů (např. kamer, ultrazvukových čidel vzdálenosti apod.).

Podle konstrukce se krokové motory dělí na tyto typy:

- Reluktanční krokové motory
- Krokové motory s permanentními magnety
- Hybridní krokové motory (největší skupina)



Obr. 11 – Krokový motor z disketové mechaniky

4.4 SERVOMOTORY

Servomotory nejsou primárně určené k pohonu robotů, ale k přesnému polohování. Pomocí délky pulzu na stabilní časové základně dojde k přesnému natočení na požadovaný úhel, proto se výborně hodí pro natáčení řídicí nápravy, jak bylo již vidět na Obr. 4. Menší modelářské servomotory se vyrábějí v nejrůznějších velikostních provedeních a výkonech. Typický tah modelářského serva se pohybuje v desítkách newtonů na centimetr.

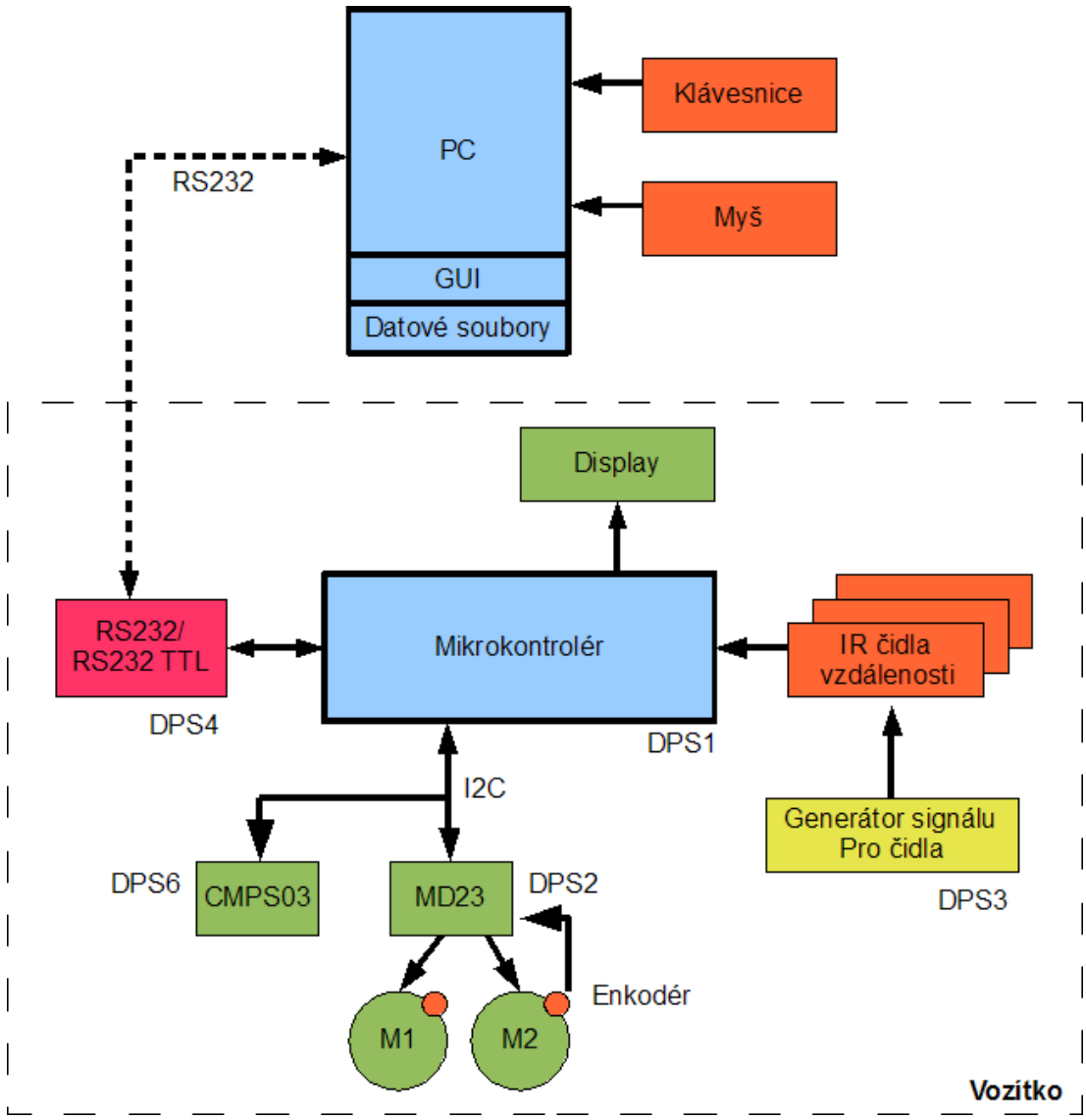
Pokud bychom servo chtěli použít jako pohon, je potřeba v jeho vnitřní elektronice provést několik zásahů, ale musíme počítat s tím, že tento motor není dimenzován na dlouhodobější zatížení.

Obr. 12 – Servo Graupner - převzato z [5]

5. KONCEPCE ŘÍDICÍHO SYSTÉMU VOZÍTKA

Pro řízení vozítka je nutné dopředu znát, jaký podvozek bude použit, jak budou data pro pohyb vozítka interpretována, jak budou měřena a vyhodnocovávána data „nasbíraná“ při jeho pohybu a jak by mělo vozítko reagovat na případné problémy a nežádoucí stavy při jeho pohybu.

Řídicí systém vozítka je rozdělen do několika menších funkčních bloků, které jsou navrhovány samostatně. Viz. blokové schéma na Obr. 13. V osobním počítači je řídící program s grafickým rozhraním pro uživatele, který jej ovládá pomocí klávesnice a myši. Počítač je s vozítkem propojen asynchronní sériovou linkou. Ve vozítku je standard RS232 převeden na TTL úrovně a dále veden do hlavního mikrokontroléru. Mikrokontrolér zajišťuje výměnu dat mezi vozítkem a počítačem, komunikuje s modulem digitálního kompasu a výkonové desky motorů, registruje stavy reflexních čidel a zobrazuje některé důležité hodnoty přímo na displeji vozítka.



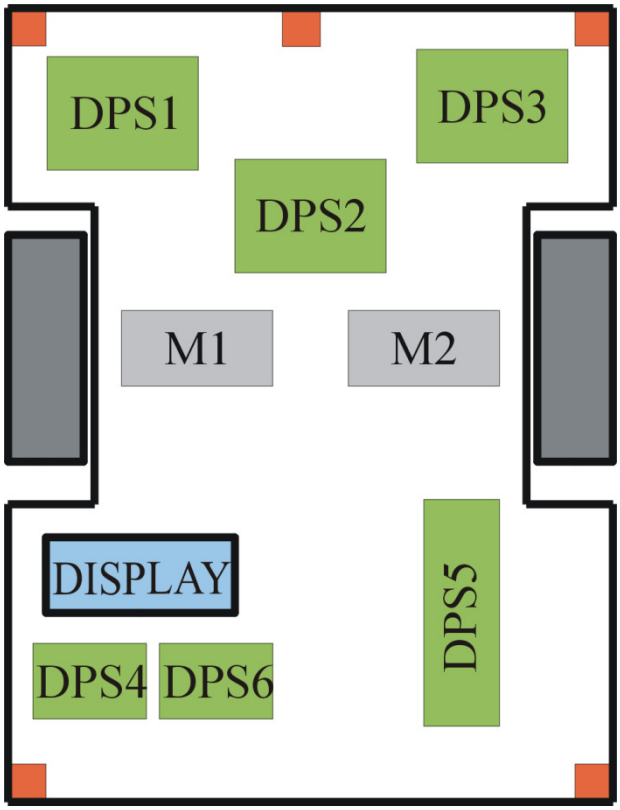
Obr. 13 - Blokové schéma řídícího systému

5.1 PODVOZEK A POHONNÁ JEDNOTKA

Pohonnou jednotku vozítka tvoří dvoukolový diferenciální podvozek s dvojicí stejnosměrných motorů (Obr. 14) s převodovkou 30:1 a enkodérem (360 čtvrt pulzů na otáčku). Motory a ostatní elektronika je připevněna na společné šasi vyrobené z 4mm silného duralového plechu, který zajišťuje dostatečnou pevnost celku a současně má nízkou hmotnost. Konstrukční uspořádání je na Obr. 15. Motory nejsou

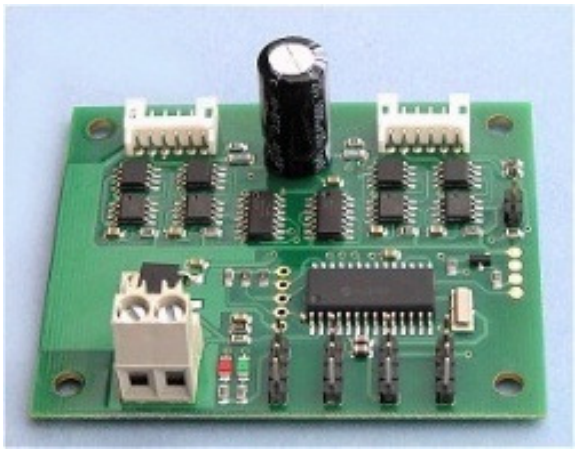
přesně v těžišti podvozku a jako opora slouží jedno volně uložené kolečko v zadní části podvozku. S pohonnými motory úzce souvisí výkonová deska určená pro tyto pohony (viz Obr. 16). Výkonová deska také zpracovává impulzy z enkodérů umístěných na motorech a jednotlivé načítané pulzy ukládá do registrů, které je možné poté vyčíst pomocí řídícího mikrokontroléru. Deska je schopna dále nastavit plynulý náběh otáček, změřit aktuální odebíraný proud motorů, napětí napájecí baterie a má funkci bezpečného zastavení motorů při ztrátě komunikace po sběrnici I2C.

Obr. 14 - Motor EMG30 [7]



Obr. 15 – Konstrukční uspořádání podvozku

DPS1 – Řídící deska s mikrokontrolérem, DSP2 – Spínací výkonová deska pro motory, DSP3 – Generátor a budič signálu pro pětici reflexních čidel, DSP4 – převodník standardu RS232 na úrovně TTL, DPS5 – pulzní pětivoltový zdroj, DPS6 – Modul digitálního kompasu.



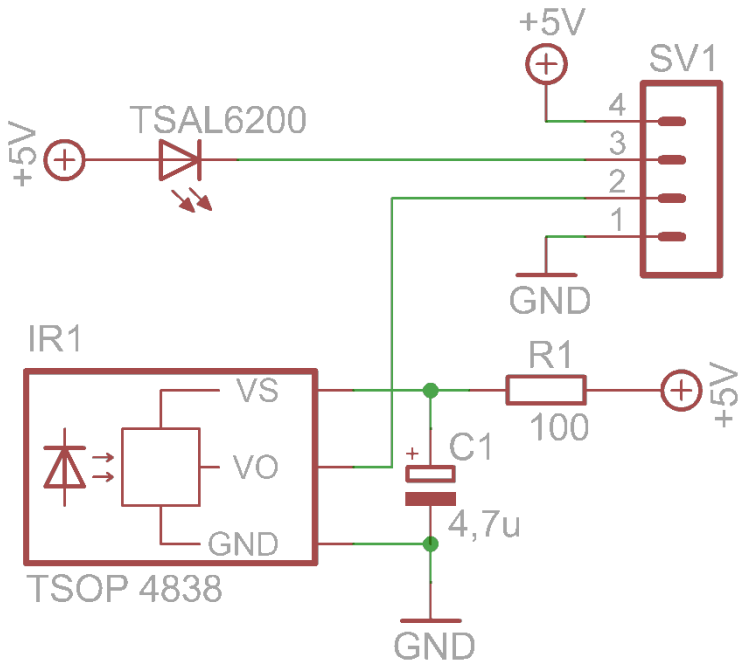
Obr. 16 - Spínací deska motorů MD23 [7]

5.2 SENZORY PŘEKÁŽEK

Při pohybu vozítka může nastat to, že se v jeho trajektorii objeví překážka. Proto je vozítko vybaveno pěticí reflexních infračidel, které slouží pro detekci překážky v jeho blízkém okolí.

Čidla jsem vytvořil z vysílací infradiody a infrapřijímače s vnitřním filtrem, který je nastaven na hodnotu 38kHz přijímaného signálu. Dioda TSAL 6200 tedy vysílá signál o této frekvenci, a pokud je překážka v detekční zóně, dojde k odrazu signálu a k detekci přijímačem. Výrobce diody uvádí vyzařovací úhel 17°, ale dochází k slabému záření i do boků. Proto je dioda vložena do kovového držáku, aby se toto záření odstínilo a nedocházelo k sepnutí senzoru vlivem tohoto bočního záření. Snímač TSOP4838 je vybaven filtrem pro signál, aby nedocházelo k detekci infrapaprsku například od slunečního záření. S použitím těchto prvků čidlo reaguje na překážky bez problémů i při osvícení přímým slunečním zářením. Výstup z čidla je binární (je/není překážka) typu otevřený kolektor a při připojení na řídící mikrokontrolér je nutné použít pull-up odpory (postačí aktivovat integrované pull-up v mikrokontroléru). Elektrické schéma čidla je na Obr. 17. Hodnoty součástek R1 a C1 jsou doporučené výrobcem požitého IR přijímače. Napájení čidla (piny 1 a 4) a signál pro IR diodu (pin 3) jsou vedeny kabelem z desky generující signál, která bude

popsána v následující kapitole 5.3. Pin 2 je propojen přímo na vstupní pin mikrokontroléru. Fyzická realizace čidla je na Obr. 18.



Obr. 17 - Schéma reflexního čidla

Obr. 18 - Reflexní čidlo

5.3 GENERÁTOR SIGNÁLU PRO ČIDLA

Reflexní infra čidla vzdálenosti potřebují pro svoji činnost signál obdélníkového průběhu o frekvenci 38kHz. Tento signál generuje deska DPS3. Jedná se o klasické zapojení astabilního klopného obvodu s integrovaným obvodem NE555. Volbou vnějších součástek je potřeba dosáhnout frekvenci generovaného signálu 38kHz se střídou blížící se 1:1. Výstupní signál z NE555 je výkonově zesílen tak, aby bylo možno přímo budit infradiody v jednotlivých čidlech. Jako zesilující prvek jsem vybral integrovaný obvod ULN 2803. Jedná se o tranzistorové pole s výstupem typu otevřený kolektor schopným spínat proud až 500mA s velkou ostrostí spínací hrany. Z výstupního konektoru desky je možné odebírat jak budící signál pro IRdiody, tak celé čidlo napájet. Schéma desky je na Obr. 19.

K výpočtu hodnot odporů R1 a R2 a kondenzátoru C1 jsem vycházel ze vztahu (5.1), který udává výrobce NE555.

$$\boldsymbol{f = \frac{1,4}{(R1 + 2 * R2) * C}} \quad \boldsymbol{(5.1)}$$

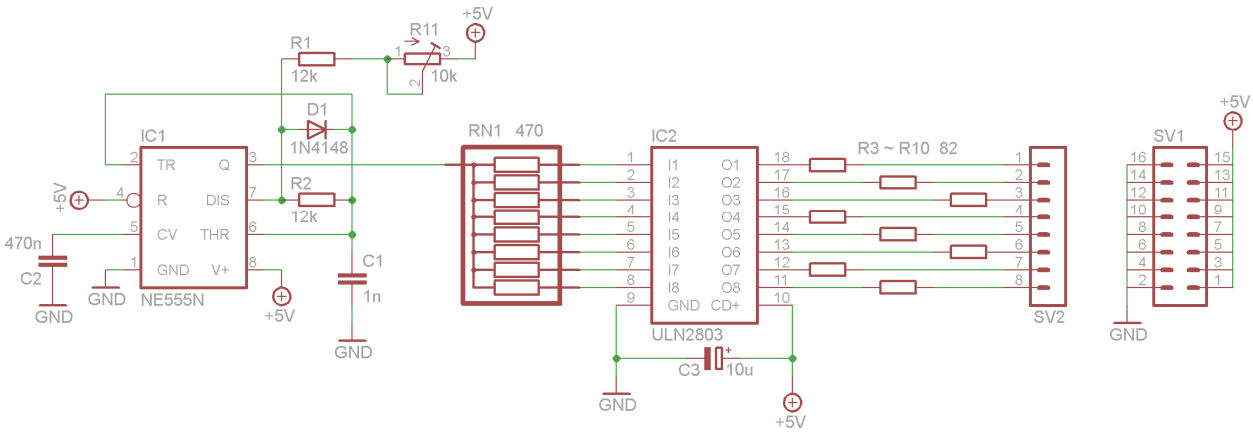
Optimální hodnoty odporů doporučené výrobcem jsou v rozmezí 1kΩ až 100kΩ. Zvolil jsem tedy R2=12kΩ a hodnotu nabíjené kapacity C1=1nF. Při požadované frekvenci f=38kHz jsem tedy osamostatnil jedinou neznámou R1 (5.2) a dosadil do rovnice (5.3).

$$\boldsymbol{R1 = \frac{1,4}{C * f} - 2 * R2} \quad \boldsymbol{(5.2)}$$

$$\boldsymbol{R1 = \frac{1,4}{1 * 10^{-9} * 38 * 10^{3}} - 2 * 12 * 10^{3} = 12,84\ k\Omega} \quad \boldsymbol{(5.3)}$$

Získaná hodnota R1=12,84kΩ neodpovídá žádné hodnotě z řady a vzhledem k výrobní toleranci součástek jsem volil nejnižší bližší hodnotu výrobní řady, tedy R1=12kΩ. Přesná generovaná frekvence je po oživení desky nastavena pomocí trimru R11 s hodnotou 10kΩ. Kondenzátor C2 slouží ke zlepšení stability rozhodovacího komparátoru.

Vypočtené hodnoty součástek sice generují požadovaný kmitočet, ale střída zdaleka neodpovídá poměru 1:1. Je to dáno tím, že kondenzátor C1 se nabíjí přes R1 i R2, ale k vybíjení dojde pouze přes R2. Proto jsou časy nabíjení a vybíjení tolik odlišné (poměr přibližně 2:5). Zlepšení jsem dosáhl zapojením diody D1 paralelně k odporu R2. Vybíjení C1 probíhá stále přes odpor R2 a nabíjení díky diodě pouze přes R1. S touto úpravou jsem dosáhl poměru přibližně 2:3, což je již dostatečné pro správnou funkci čidel a k dosažení větší detekční vzdálenosti.



Obr. 19 - deska pro čidla

5.4 KOMUNIKACE PC ↔ VOZÍTKO

Pro komunikaci mezi počítačem a vozítkem jsem použil sériovou linku, jak bylo již dříve uvedeno na blokovém schématu (Obr. 13). Tato linka je součástí každého mikrokontroléru i osobního počítače, popřípadě je její jednoduchá implementace pomocí USB redukce. RS232 poskytuje dostatečně rychlý a kvalitní přenos dat. Pro přenos dat jsem zvolil standardní nastavení přenosových parametrů a to 9600Bd, 8 bitů, bez parity a jeden stop bit. Toto nastavení je dostatečně rychlé pro komunikaci mezi počítačem a vozítkem a přitom je značně odolné proti rušení na komunikační lince dlouhé přibližně čtyři metry.

Komunikace probíhá ve stanovené znakově orientované sekvenci. Začátek komunikace zahajuje vysílací znak, který jsem zvolil „#“, „@“ a „$“, dále následuje datová sekvence, poté jednoduché zabezpečení a řídící slovo ukončuje znak „*“.

Startovací znaky jsem zvolil tři z důvodu, aby bylo možné snadněji rozpoznat, zdali se jedná o online ruční řízení, sekvenci pro nastavení rychlosti nebo slovo, kterým je definován směr a počet kroků, který je potřeba ujet.

Vozítko odesílá v pravidelných intervalech svůj stav v podobné sekvenci znaků zpátky do počítače. Pravidelnost odesílání z vozítka do počítače může narušit odeslání stavu při nějaké zvláštní události – například detekce překážky.

Při ručním řízení je sekvence velice jednoduchá a skládá se pouze ze čtveřice znaků: startovací znak #, číslo udávající směr, zabezpečovací znak a ukončovací znak. Jednotlivé směry jsem volil takto: 0 – zastavení, 1- vpřed, 2 – vzad, 3 – vlevo a 4 – vpravo.

Zabezpečení odesílaných dat jsem realizoval tak, že jsem vytvořil součet všech hodnot odesílaných znaků v řetězci vyjma startovacího a ukončovacího. A jako zabezpečovací znak je odeslána jen poslední (nejnižší) cifra zmiňovaného součtu.

Další řídící sekvencí je slovo pro nastavení rychlosti složené ze znaku @, hodnoty rychlosti, zabezpečení a z ukončovacího znaku *.

Poslední sekvencí je odeslání předprogramovaného příkazu. Přenos je zahájen znakem $, následuje hodnota směru jako v již uvedeném případě, za tímto znakem je pětimístné číslo reprezentující potřebný počet kroků, zabezpečovací součet a přenos je opět ukončen. Délka sekvence je pevně stanovená, a pokud počet kroků je menší než pětimístné číslo, je zepředu tato hodnota doplněna nulami.

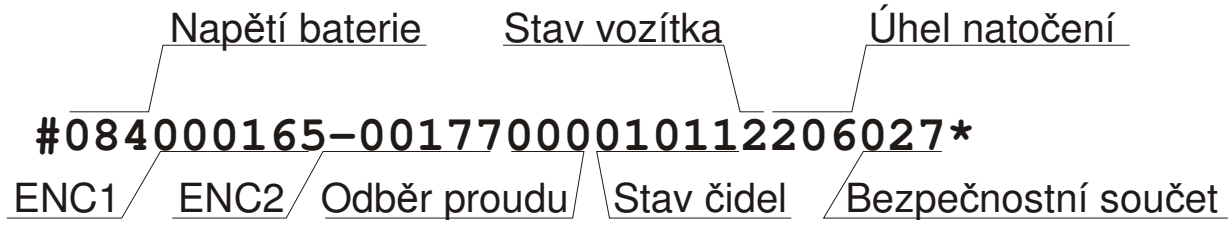
Hodnota počtu kroků může být sice pětimístné číslo, ale maximální hodnota, kterou může nabývat, je 65535. Tato hodnota je maximum datového typu unsigned int vytvořeného spojením dvou osmibitových registrů použitého mikrokontroléru.

Vozítko průběžně informuje počítač o svém aktuálním stavu pravidelným odesíláním stavového slova.

Tuto posloupnost znaků jsem zvolil takto: Za startovacím znakem jsou rezervované tři znaky pro aktuální napětí napájecí baterie ve tvaru desítky, jednotky, desetiny voltů. Dále následuje šest znaků pro počet aktuálních kroků motoru 1 a ihned stejný počet znaků pro čítač kroků motoru 2. Znaků je zvoleno šest, aby byly rozlišitelné kladné a záporné kroky. Dále následují tři znaky odpovídající

proudovému odběru motorů, následně je pět znaků rezervovaných pro reflexní čidla nabývající hodnot 0 nebo 1 (je / není překážka). Po informaci o čidlech je odeslána hodnota jednoho ze tří stavů, ve kterých se vozítko může nacházet. Dále následuje úhel natočení zjištěný modulem kompasu a informační jednotky uzavírá zabezpečovací součet. Aby byla dodržena vždy stejná délka řetězce, jsou všechny hodnoty menší, než je jejich maximální počet, doplněny zepředu nulami.

Ukázka stavového slova je na Obr. 20. Napětí baterie je 8.4V, enkodér prvního motoru od resetu napočítal 165 kroků a druhý -177, motory neodebírají žádný proud, prostřední a levé přední čidlo detekuje překážku. Vozítko je ve stavu 2, což signalizuje zastavení z důvodu překážky. Registr úhlu natočení nabývá hodnoty 206 a zabezpečovací součet tohoto řetězce je 27.



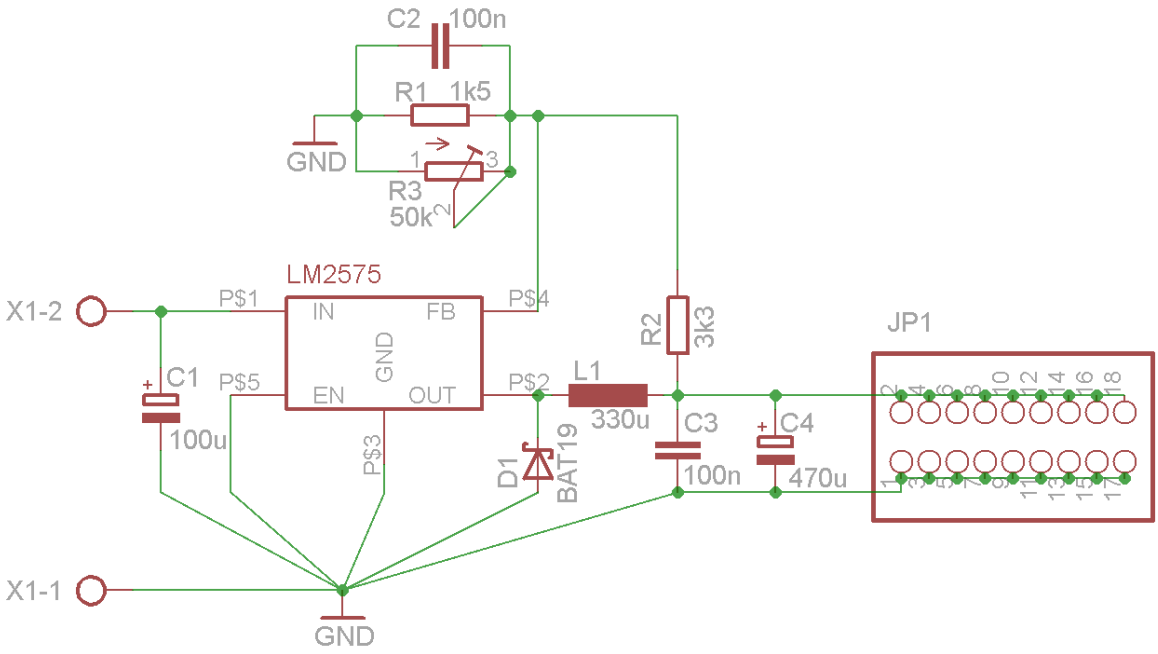
Obr. 20 - Stavové slovo

5.5 NAPÁJENÍ VOZÍTKA

Celé vozítko je napájeno ze sedmi NiCd článků Sanyo N-500AR. Jedná se o sadu článků určenou pro modelářské účely (Obr. 21). Tento accu pack jsem zvolil vzhledem k jeho cenové dostupnosti, poměrně malým rozměrům, odolnosti článků a přijatelnému napájecímu napětí celého vozítka.

Obr. 21 - Napájecí baterie

Většina elektroniky vozítka vyžaduje napájecí napětí 5V. Použitá baterie poskytuje jmenovité napětí 8,4V. Proto jsem navrhl a sestrojil spínaný zdroj napájený baterií, na jehož výstupu je stabilizované a filtrované napětí 5V. Schematické znázornění zdroje je na Obr. 22



Obr. 22 - Schéma pulzního zdroje napětí

Hlavní součástku napájecí desky tvoří monolitický integrovaný obvod LM2575. Jedná se o spínací regulátor napětí s proudovým zatížením 1A a frekvencí spínání 52kHz. Regulovatelné výstupní napětí je v rozsahu 1,23 až 37V za podmínky použití maximálního napájecího napětí 40V. Obvod pro svoji správnou činnost vyžaduje určité rozdílové napětí vstup/výstup. Během testování tohoto obvodu jsem zjistil, že při nastaveném výstupním napětí 5V postačí pro správnou činnost napětí pouze 7V – rozdíl tedy pouze 2V. Při použití sedmičlánkové NiCd baterie se jmenovitým napětím 8,4V je tedy dostatečná rezerva a obvod bude pracovat bez problémů.

Hodnota výstupního napětí je volena kombinací hodnot odporů R1 a R2. Výrobce v katalogovém listu udává vztah pro výpočet výstupního stabilizovaného napětí (vzorec 5.4).

$$U_{OUT} = 1,23 * \left(1 + \frac{R2}{R1}\right) \quad (5.4)$$

Hodnoty odporů zpětnovazebního děliče by dle výrobce měly být v hodnotách jednotek až desítek kiloohmů. Proto jsem zvolil R2=3,3kΩ a podle (5.5) vypočetl hodnotu R1.

$$\boldsymbol{R1 = \frac{1,23 * R2}{U_{OUT} - 1,23} = \frac{1,23 * 3,3 * 10^3}{5 - 1,23} = 1,077\ k\Omega} \quad \boldsymbol{(5.5)}$$

Výslednou hodnotu R1 jsem volil z výrobní řady součástek R1=1,5kΩ. Při použití hodnot R1=1,5kΩ a R2=3,3kΩ by bylo výstupní napětí 3,936V (5.6), proto je k R1 paralelně zapojený trimr s hodnotou 50kΩ k přesnému donastavení na hodnotu 5V.

$$\boldsymbol{U_{OUT} = 1,23 * \left(1 + \frac{3,3 * 10^3}{1,5 * 10^3}\right) = 3,936V} \quad \boldsymbol{(5.6)}$$

Kondenzátory C1 a C4 jsou požadované výrobcem a jsou jim předepsány i jejich doporučené hodnoty. Kondenzátor C3 jsem přidal k lepšímu odrušení výstupního napětí a C2 k odrušení při nastavování přesné hodnoty požadovaného stabilizovaného napětí. Dioda D1 a hodnota L1 jsou doporučeny výrobcem.

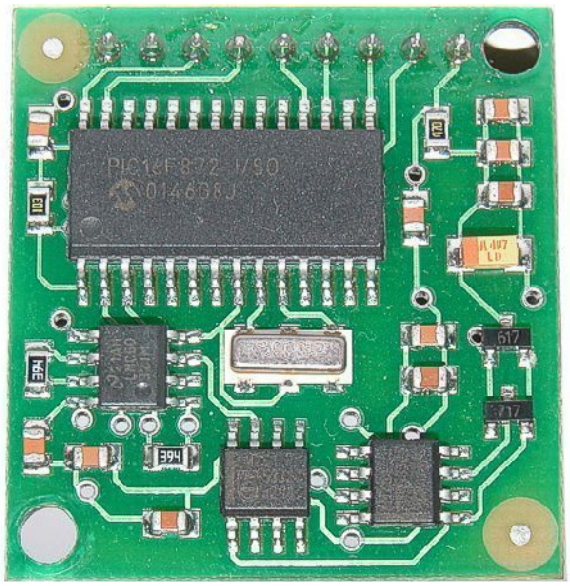
5.6 ČIDLO ÚHLU NATOČENÍ

Vozítko jsem dále vybavil modulem pro snímání úhlu natočení vůči severnímu pólu Země. Pro implementaci jsem se rozhodl z důvodu opakování průjezdu předprogramovanou dráhou. Uživatel sice projede dráhu, kterou vozítko následně zopakuje, ale bod a úhel, pod kterým ze startovní čáry vyrazí, nemusí být pokaždé stejný. Proto jsem se rozhodl pro kontrolu úhlu výjezdu přidat vozítku právě toto čidlo.

Jedná se o celistvý modul s vlastním integrovaným obvodem, který komunikuje s nadřazeným systémem pomocí I2C sběrnice, kterou s výhodou využiji (Obr. 23). Modul má obdobně jako deska pro spínání motorů svoje vnitřní registry, ve kterých jsou jednotlivé podstatné veličiny uloženy. Pro použití vozítka je významný pouze jeden z těchto registrů a to ten, ve kterém je uložena hodnota úhlu

natočení. Registr je osmibitový a úhel je reprezentován proměnnou typu „unsigned char“.

Komunikace s deskou tedy probíhá po již zmíněné sběrnici I2C. Adresa desky je v hexadecimálním vyjádření C0h. Skladba komunikace pro vyčtení hodnoty příslušného registru je analogií komunikace s výkonovou deskou. Komunikací po I2C sběrnici se budu zabývat podrobněji v kapitole 6.3 Funkce pro sběrnici I2C.



Obr. 23 - Modul CMPS03 – převzato z [7]

6. ŘÍDICÍ JEDNOTKA VOZÍTKA

Hlavní řídící jednotku celého vozítka tvoří mikrokontrolér ATmega16. Jedná se o osmibitový nízkopříkonový mikrokontrolér založený na rozšířené AVR RISC architektuře. Vykonavatelnost instrukce v jednom hodinovém cyklu dovoluje tomuto mikrokontroléru výkon 1MIPS při použití 1MHz taktování. Hlavní rysy jsou shrnuty v následujících bodech [8]:

- Instrukční soubor obsahuje 131 instrukcí
- 32 registrů délky 8 bitů
- Čtveřice 8bitových vstupně/výstupních bran
- Maximální hodinový kmitočet 16MHz (maximální výpočetní výkon 16 MIPS)
- Flash paměť programu 16KB
- Datová paměť RAM 1 KB
- Datová EEPROM 512 B
- Dva 8bitové a jeden 16bitový čítač/časovač
- Čtveřice PWM kanálů
- 10bitový A/D převodník
- Jednotky USART, SPI, TWI

Tento mikrokontrolér jsem zvolil, protože jeho architektura a vlastnosti splňují předpokládané požadavky na řídicí systém. Navíc tato řada mikrokontrolérů má malý příkon, velkou spolehlivost, snadné přeprogramování (programování v osazené desce) a s touto řadou mikrokontrolérů mám dobré zkušenosti z jiných aplikací. Typové označení použitého mikrokontroléru je ATmega16-16PU. Jedná se tedy o řadu mega 16 s maximálním možným taktovacím kmitočtem 16MHz industriálního použití (-40°C až 85°C) schváleného pro evropský trh (neobsahující halogenidy). Pro vytvoření aplikace v jazyku C jsem použil vývojové prostředí CodeVisionAVR C od společnosti HP InfoTech [9], [10].

Hlavním úkolem mikrokontroléru je zajistit bezeztrátovou komunikaci mezi nadřazeným počítačem a výkonovou deskou pro pohonné jednotky. Druhotným

úkolem je sledování jeho okolí a hlídání, je-li v jeho dráze nějaká překážka. Terciární a nejméně významná činnost je zobrazování svého stavu na displeji vozítka.

6.1 ŘÍDÍCÍ PROGRAM MIKROKONTROLÉRU

Jak již bylo dříve zmíněno, program je napsán v jazyku C za použití vývojového prostředí CodeVision.

Po zapnutí a náběhu krystalového oscilátoru mikrokontrolér nastaví jednotlivé vstupně výstupní porty do požadovaného stavu (viz. Obr. 24), nastaví jednotlivé čítače/časovače, nastaví přerušení a resetuje všechny globální proměnné. Poté je v programu nastavena krátká čekací smyčka, aby došlo k zapnutí a inicializaci spínací desky motorů.

Po vyčkání je do desky odeslány parametry pro řízení pomocí sběrnice I2C, po které tato deska (DPS2) komunikuje s mikrokontrolérem. Následuje globální aktivace jednotlivých přerušení a program vstupuje do nekonečné smyčky *while(1){}*. V této smyčce se testuje, není-li v okolí vozítka žádná překážka, jaké je přijaté slovo po sériové lince z počítače a zjišťují se hodnoty načítané z enkodérů jednotlivých motorů.

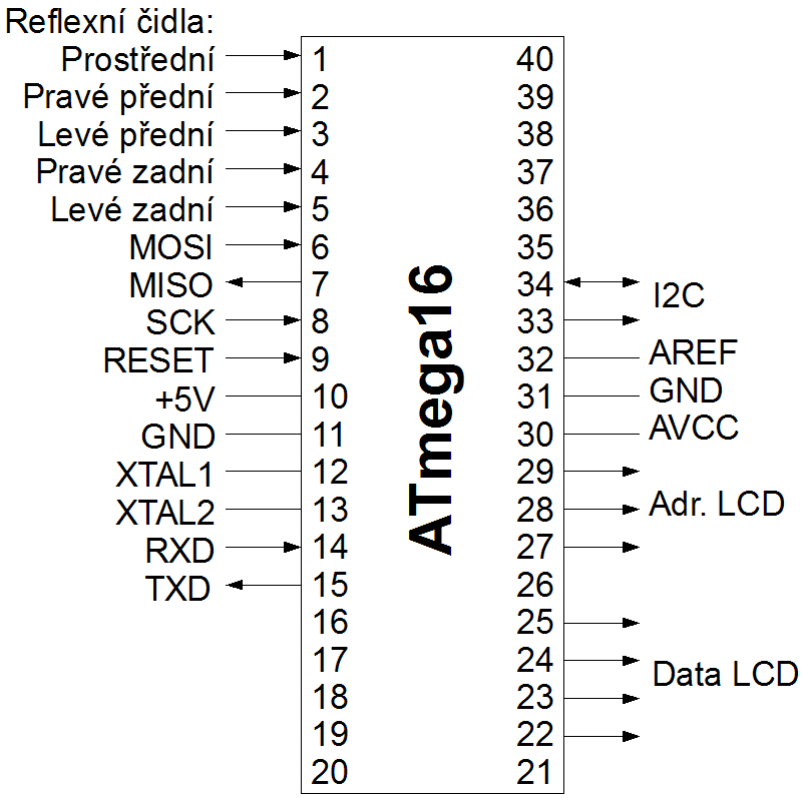
Přijaté znaky po sériové lince se přijímají pomocí přerušení běžící nezávisle na hlavní nekonečné smyčce. Jednotlivé znaky se skládají do řetězce znaků a poté je tento řetězec dále vyhodnocován.

První vyhodnocovací podmínkou je řídící sekvence na předem známý počet kroků, který je nutno ujet. Tato podmínka nejprve testuje tyto parametry: počítadlo přijatých znaků musí být rovno devíti, z toho první znak musí být roven znaku „$“ a poslední „*“. Poté je z přijatých znaků vypočten zabezpečovací součet a porovnán s přijatým výpočtem z počítače. Při souhlasu je již jisté, že přijatá sekvence odpovídá požadovanému příkazu a celý řetězec byl přijat bez chyb, a proto je volána funkce *JedDrahu()* viz. kapitola 6.2 Funkce JedDrahu.

Další testy přijatého řetězce se týkají nastavení rychlosti a směru při ručním řízení. Nejprve je testován počet znaků a to na hodnotu čtyři, následuje test na přijatý první znak. Je-li přijat jako první znak „#” a poslední je „*“, uloží se do globální proměnné *Smer* přijatá hodnota požadovaného směru jízdy. Obdobně je do proměnné

Rychlost uložena požadovaná rychlost motorů při příjmu prvního znaku „@“ udávajícího data pro nastavení rychlosti.

Po těchto testech následuje několikanásobná podmínka závislá na proměnné směru. V této podmínce je vyhodnocen požadovaný směr a je volána příslušná funkce odesílající data po sběrnici I2C do výkonové desky motorů.



Obr. 24 - Využití pinů

6.2 FUNKCE JEDDRAHU

V okamžiku, kdy je vyhodnoceno a zkontrolováno přijaté slovo udávající předprogramovanou dráhu, vstupuje program do funkce *JedDrahu()* a vozítku je nastaven status 1 udávající, že program vozítka vstoupil do této funkce a provádí stanovený příkaz.

V této funkci se nadále pracuje s přijatými znaky tak, že dojde k výpočtu požadovaného počtu kroků do proměnné *Draha*, a podle hodnoty udávající směr je odeslán příkaz do výkonové desky o požadovaném směru.

V okamžiku, kdy se motory rozeběhnou, začínají jednotlivé enkodéry ukládat počet načítaných pulzů do registrů. Tyto registry jsou průběžně čteny a porovnávány s požadovaným počtem kroků.

Při manuálním řízení je možné s vozítkem jezdit při několika odlišných rychlostech. V tomto případě je rychlost pohybu pevně stanovena a to z důvodu setrvačnosti celého vozítka. Pokud by totiž vozítko jelo maximální rychlostí, může nastat problém, že požadovaný počet kroků vlivem velké rychlosti a setrvačnosti „přejede“. Z tohoto důvodu je také do této funkce zahrnuto zpomalení v místě, kdy se vozítko blíží do koncového bodu. V oblasti, kdy vozítku zbývá ujet 250 kroků (cca 20 cm) do cíle, je již rychlost poloviční a 100 kroků před koncem je hodnota rychlosti čtvrtina původní rychlosti. Toto zpomalení zpřesní navigaci pomocí metody dead reckoning.

V této funkci dochází samozřejmě k testování stavu čidel a při detekci překážky je okamžitě zastaveno a nastaven stav 2 – překážka v trajektorii. V opačném případě, kdy je příkaz proveden a vozítko urazilo stanovenou dráhu bez jakýchkoliv komplikací, dojde k nulování počítadel, uvedení vozítka do klidu a nastavení statusu 0, který udává klidový stav, a vozítko vyčkává na další příkazy.

6.3 FUNKCE PRO SBĚRNICI I2C

Pro komunikaci s deskou motorů jsem vytvořil dvě funkce komunikující po synchronní sběrnici I2C vyvinutou firmou Panasonic.

Výkonová deska má již od výroby pevně stanovenou adresu a to B0h v hexadecimálním tvaru. Při zápisu na požadovaný registr této desky začíná komunikace startovací sekvencí, po ní následuje adresa desky, dále adresa požadovaného registru následovaná hodnotou, kterou je potřeba zapsat, a spojení je ukončeno tzv. stop sekvencí. Viz Obr. 25.

START	Adresa 0xB0	A	Adr. registru	A	Data	A	STOP

Obr. 25 - I2C zápis

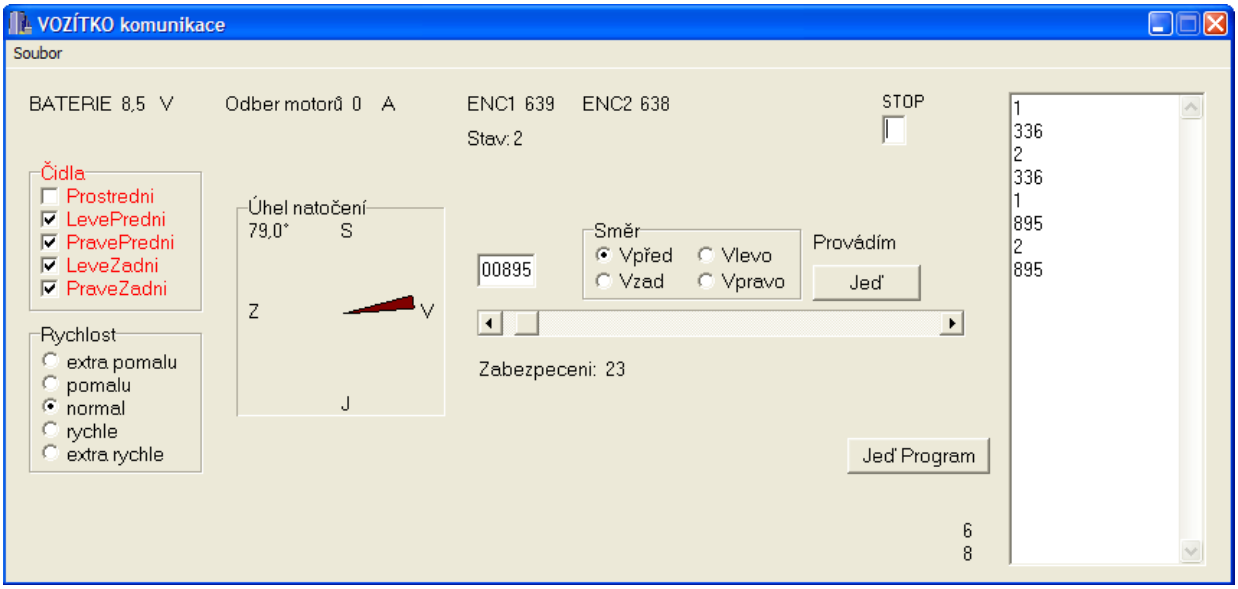
Funkce pro čtení z registrů je poněkud komplikovanější a je nejprve nutno odeslat adresu desky B0h, poté adresu registru, který chceme přečíst, poté je opět zahájen přenos start sekvencí, dále pak je odeslána adresa B1h a po ní teprve následují data uložená v požadovaném registru. Viz. Obr. 26.

START	Adresa 0xB0	A	Adr. registru	A	START	Adresa 0xB1	A	Data	STOP

Obr. 26 - I2C čtení

7. ŘÍDÍCÍ PROGRAM POČÍTAČE

Pro vytvoření řídicího programu počítače jsem zvolil vývojové prostředí Borland C++Builder. Toto prostředí jsem si vybral vzhledem k tomu, že aplikace by měla být pro uživatele přehledná a v tomto prostředí se mi dobře pracuje a vytváří právě takové aplikace. Program musí přehledně zobrazit, co se s vozítkem aktuálně děje, a také uživateli zprostředkovat jeho aktuální nastavení, případně stavy (rychlost, napětí baterie). Program v průběhu vývoje vozítka přešel od jednoduché vývojové verze, která umožňovala pouze manuální řízení a zobrazovala jen několik hodnot z vozítka, do finální podoby. Ta je schopna využít všech možností řízení vozítka, ukládá jednotlivé povely z řízení do souboru a poté je aplikace schopna tyto příkazy opětovně odeslat ze souboru do vozítka. Aplikace dále zobrazuje aktuální napětí baterie, odebíraný proud motorů, stavy jednotlivých čidel, stav vozítka (jede, stojí, provádí příkaz, apod.) a pro manuální řízení přehledně nastavuje rychlost jízdy. Finální podoba aplikace je na Obr. 27.



Obr. 27 - Řídící program

Ihned po spuštění aplikace dojde k otevření sériového kanálu a nastavení jeho přenosových parametrů. K otevření sériového portu dochází pomocí tzv. API funkce. Je vytvořen handle daného portu a poté se na něj přistupuje jako by se zapisovalo do souboru s využitím třídy. Pro nastavení přenosové rychlosti vypadá zápis takto: *Serial->BaudRate=br9600;*.

Srdce uživatelské aplikace tvoří časovač tmrCti, který v pravidelných intervalech vyčítá buffer sériové linky a vyhodnocení přijatých znaků dojde při události OnTimer.

7.1 FUNKCE HLAVNÍHO ČASOVAČE

Uvnitř funkce hlavního časovače dochází k plnění pole znaků přijatými znaky po sériové lince a současně dochází k testování znaků na určitá kritéria. Pokud je přijat jakýkoli jiný znak než „očekávaný“, dojde k resetu počítadla přijatých znaků a znehodnocení pole znaků. Za očekávané znaky se považují pouze ty znaky, ze kterých může být přijaté slovo složeno - tedy startovací a ukončovací znak a znaky číslic.

V okamžiku, kdy proměnná udávající počet přijatých znaků dosáhne hodnoty 31, tedy přijatého celého slova odeslaného vozítkem, dojde k vyhodnocení tohoto přijatého řetězce. Jednotlivé hodnoty ze slova jsou přiřazeny příslušným globálním proměnným, se kterými je dále pracováno v ostatních funkcích, a současně jsou přijaté hodnoty přehledně zobrazeny uživateli.

7.2 FUNKCE PRO ODESLÁNÍ DAT DO VOZÍTKA

V programu počítače jsem vytvořil trojici funkcí pro odesílání dat po sériové lince do vozítka. Každá z funkcí přísluší danému příkazu. Funkce *OdesliData()* slouží k odesílání příkazu při manuálním řízení, *OdesliData2()* pro nastavení rychlosti a *OdesliData3()* pro vyslání řetězce předprogramované jízdy.

7.2.1 Funkce OdesliData()

Jak již bylo výše zmíněno, tato funkce je stěžejní při manuálním ovládání. V okamžiku volání této funkce se na sériovou linku zapisuje startovací znak „#“ následovaný znakem představující směr z globální proměnné, za ním zabezpečovací součet a ukončovací znak „*“. Paritní součet pro jedinou hodnotu je tatáž hodnota, proto dojde k provedení příkazu *Serial->WriteChar(smer)* dvakrát po sobě.

V této funkci je dále zahrnuto zapamatování daného příkazu pro případné pozdější zopakování. Uložení směru jízdy je provedeno zápisem do textového pole příkazem *memoTraj->Lines->Add(smer)*. Počet ujetých kroků se za tuto hodnotu zapíše až po uvolnění klávesy v době, kdy je vozítko uvedeno do klidového stavu.

7.2.2 Funkce OdesliData2()

Tato programová funkce odesílá vozítku požadovanou rychlost. Je to obdoba předchozí funkce *OdesliData()*, ale s tím rozdílem, že startovacím znakem je v tomto případe „@”.

7.2.3 Funkce OdesliData3()

Z trojice funkcí pro odesílání dat je tato funkce programově nejsložitější a jejím úkolem je sestavit a odeslat řetězec příkazu pro předem definovanou jízdu. Požadovaný směr je volen pomocí skupiny radiobuttonů a počet kroků tažením scrollbaru. Příkaz se odešle stiskem tlačítka Jeď.

Pro počet ujetých kroků je v odeslaném řetězci rezervováno pět znaků a hodnota může být v rozmezí hodnot unsigned int. Nižší hodnoty musí být zepředu doplněny nulami, aby byla zachována pokaždé stejná délka. Tato skutečnost je ošetřena na začátku funkce a je zobrazena v příslušném políčku (EditBoxu - edtDraha), kde je hodnota vyjádřena jako text. Po doplnění nulami následuje výpočet kontrolního součtu a je uložen do proměnné *Zabezpeceni*. Poté je již možné vyslat řídicí sekvenci do vozítka. Za startovacím znakem „$“ následuje vyslání hodnoty udávající směr. Dále je nutné vyslat sekvenci znaků reprezentující počet kroků. Tato hodnota je vyjádřena v již zmiňovaném editboxu pomocí textu. Textovou

reprezentaci hodnoty s výhodou využiji a pomocí příkazu *Serial->WriteString(sdtDraha->Text)* dojde k jejímu přesunu do odesílacího bufferu sériové linky. Následuje odeslání kontrolního součtu ukončeného stop znakem „*“.

7.3 FUNKCE ZOBRAZSTAVCIDEL()

Jak již samotný název této funkce napovídá, slouží k zobrazování stavu jednotlivých čidel uživateli. Tato funkce pracuje se znaky 19 až 23 odeslané vozítkem, které reprezentují, zachytilo-li dané čidlo překážku nebo ne. Podle přijatých hodnot je příslušný ChechBox zaškrtnut, nebo ne. Ve funkci následuje podmínka, která testuje výskyt překážky aspoň jednoho z čidel. Pokud tato skutečnost nastane, dochází ke změně barvy textu ze standardní *clWindowText* na *clRed*, které si uživatel rychleji všimne.

7.4 MANUÁLNÍ ŘÍZENÍ ROBOTA

Ruční řízení robota pomocí kurzorových šipek je nepostradatelnou funkcí programu. Okamžik stisku klávesy je testován v jediném editBoxu pomocí události KeyDown daného objektu. Pokud stiskne uživatel klávesu, dojde k testování kódu klávesy, která událost vyvolala pomocí několikanásobného větvení *switch*. Kód kurzorové šipky nahoru (vpřed) je 38, dolů (vzad) 40, vlevo 37 a vpravo 39. Vyhovuje-li podmínka jedné z hodnot, dojde ke změně popisu směru jízdy a zapsání hodnoty směru jízdy do globální proměnné *smer*. Pokud uživatel stiskne jakoukoli jinou klávesu, bere tuto skutečnost podmínka jako příkaz k zastavení. Po testech stisknuté klávesy je volána funkce *OdesliData()*, která byla popsána dříve.

7.5 PŘEDPROGRAMOVANÝ POVEL VOZÍTKA

Předprogramovaným povelem vozítka se rozumí to, že uživatel nastaví směr, jakým má vozítko jet, a počet kroků, který musí urazit. Směr je volen čtveřicí radiobuttonů a délka dráhy pomocí tažení scrollbaru. Stisknutím tlačítka pro povel k jízdě dojde k volání funkce *OdesliData3()*, která byla popsána v kapitole 7.2.3, požadovaný směr a počet kroků je zapsán do MemoBoxu k pozdějšímu uložení.

7.6 ZMĚNA RYCHLOSTI JÍZDY

Pro změnu rychlosti jízdy vozítka je v ovládacím programu skupinka pěti RadioButtonů. Požadovanou rychlost tedy uživatel volí kliknutím na jeden z buttonů. Do globální proměnné rychlost je uložena požadovaná hodnota získaná z tzv. ItemIndexu skupiny buttonů a je volána funkce *OdesliData2()*. K odeslání o změně rychlosti dojde i v případě, že uživatel klikne na rychlost, která je již nastavena.

7.7 PRÁCE SE ZAPAMATOVANÝMI PŘÍKAZY

Během ručního řízení, nebo předprogramované jízdy, dochází k ukládání jednotlivých příkazů do MemoBoxu v podobě textu. Tento sled příkazů je možné pomocí roletové nabídky *Soubor->Ulož* uložit k pozdějšímu načtení, dále je možné z již existujícího souboru tyto příkazy načíst *Soubor->Otevři* nebo existující program úplně vymazat a začít s prázdným MemoBoxem pomocí *Soubor->Nový*.

7.7.1 Uložení souboru

Pro uložení jsem využil standardního knihovního dialogu SaveDialog. Nejdůležitější metoda tohoto dialogu je *Execute*, kterou je dialog vyvolán a která nabývá hodnoty true, je-li dialog uzavřen stiskem tlačítka OK. Proto je tato návratová hodnota testována a podprogram uložení postupuje ve vykonávání dále. V opačném případě bude dialog ukončen a žádný další zásah se s daty nestane.

Během ukládání souboru mohou nastat dvě odlišné varianty. První možností je, že soubor, který uživatel hodlá uložit, ještě neexistuje. V tom případě je vytvořen soubor a data jsou do něj uložena. Následuje uvolnění souboru z paměti uzavřením jeho handle.

Druhým případem při ukládání je existence shodného názvu souboru, kdy dojde k tvrdému přepsání původních dat v souboru novými a opět je paměť uvolněna.

7.7.2 Načtení uložených dat

Pro otevření souboru a načtení dat z něj existuje v použitém vývojovém prostředí také dialog, a to OpenDialog. Obdobně jako u SaveDialogu je nutné volat Execute příslušného dialogu a testovat návratovou hodnotu. V případě, že je navráceno „true“, pokusí se program soubor s požadovaným jménem otevřít, vyčíst z něj veškerá data a ty následně zapsat do prázdného MemoBoxu k další prácí.

7.8 JÍZDA VOZÍTKA PO PŘEDEM ZADANÉ TRAJEKTORII

Tato funkce je další významná vlastnost řídícího programu, která pracuje se zapamatovanými příkazy v hlavním MemoBoxu programu. Uživatel manuálním řízením nebo pomocí příkazů projel s vozítkem požadovanou trajektorii, jejíž jednotlivé povely byly ukládány. A právě pomocí těchto povelů je vozítko schopné celou dráhu zopakovat.

Jednotlivé příkazy jsou ukládané jako text v řádcích pod sebou. Na lichém řádku je vždy směr, kterým má vozítko jet, a na sudém řádku je počet požadovaných kroků. Jeden sudý a jeden lichý řádek tvoří společně jeden příkaz.

V okamžiku stisku tlačítka „Jeď dráhu“ dojde k nastavení proměnné *JedeProgram* na hodnotu jedna. Tato proměnná slouží jako příznak pro program, že uživatel zvolil zopakovat od začátku celou dráhu.

Následuje přečtení hodnoty prvního řádku udávajícího směr a uložení hodnoty do globální proměnné *smer*. Následně řídící program vyčte hodnotu druhého řádku odpovídající počtu kroků a na tuto hodnotu nastaví ScrollBar, kterým uživatel volil počet kroků při předprogramovaném příkazu (Viz. kapitola 7.5 Předprogramovaný povel vozítka). V globální proměnné je požadovaný směr, ScrollBar udává počet kroků, postačí tedy pouze zavolat funkci *OdesliData3()*, která příkaz odešle do vozítka.

V tomto okamžiku začíná mikrokontrolér zpracovávat a vykonávat příkaz o pohybu. Stav vozítka se mění z 0-klid na 1-je v pohybu a řídící program v počítači pracuje pouze ve smyčce hlavního časovače.

V této smyčce přijímá a zobrazuje data z vozítka a vyčkává, až vozítko dokončí odeslaný příkaz. Když vozítko změní svůj stav na klidový a další příkazy pro jízdu jsou ještě neprovedeny, dojde opět k přečtení dalších dvou řádků a je volána funkce *OdesliData3()*. Toto se opakuje do chvíle, než jsou odeslané všechny příkazy do vozítka, nebo dokud jedno z čidel vozítka nezareaguje na vloženou překážku a dojde k zastavení.

8. ZÁVĚR

V rámci své diplomové práce jsem navrhl a realizoval jednoduché vozítko poháněné dvoukolovým diferenciálním podvozkem s jedním opěrným bodem. Toto vozítko je propojeno s osobním počítačem a je z něj řízeno.

Koncepci hardwaru jsem zvolil blokovou tak, aby jednotlivé části tvořily samostatné celky (deska mikrokontroléru, napájení, generátor signálu pro čidla).

Vytvořená reflexní čidla detekují při normální rychlosti pohybu překážku v dostatečném předstihu tak, aby vozítko zastavilo a nedošlo k jeho kontaktu s překážkou. Reakční pole čidel nepokrývá celé okolí vozítka, a pokud bude překážka velmi úzká a vozítko k ní pojede mimo jeho středovou osu, může nastat kolize. Takovouto překážku může tvořit například noha židle.

Zvolená napájecí baterie ze sedmi NiCd článků s kapacitou 500mAh vystačí vozítku více jak na hodinu provozu. Klidový odběr celé elektroniky je 190mA. Při pohybu je proudový odběr z napájecí baterie navýšen o odběr motorů, který ovšem díky malé hmotnosti vozítka nepřesáhne 250mA.

Komunikace mezi vozítkem a počítačem se děje prostřednictvím rozhraní RS232 za použití tří vodičového spojení. Jako propojovací kabel jsem vybral stíněnou dvojlinku používanou například u stereofonních sluchátek. Až do vozítka je zachován standard RS232 daný CCITT a až u mikrokontroléru je tento standard převeden na úrovně TTL. Tím je sníženo riziko znehodnocení komunikace vozítka a počítače. Komunikace byla bezeztrátová i při vložení komunikační linky do silného elektromagnetického pole vytvořeného vodičem, kterým protéká stejnosměrný proud 170A.

Použitý modul digitálního kompasu bez problémů udává správné hodnoty i v blízkosti vodičů elektrického napětí, avšak při přiblížení se k silnému magnetickému poli ztrácí směr.

U finální verze jsem nezaznamenal žádné nežádoucí stavy vytvořeného softwaru mikrokontroléru a ani ovládacího programu v počítači.

Grafické rozhraní jsem se snažil vytvořit tak, aby bylo co nejvíce přehledné a intuitivní.

9. SEZNAMY

9.1 SEZNAM LITERATURY:

[1] URL: <http://www.colorado.edu/geography/gcraft/notes/gps/gps_f.html:> [rev. 2000-1-5]

[2] URL: <http://www.skyfly.cz/gpspraxe> [rev. 2004-9-25]

[3] URL: <http://www.rctek.com/handling/ackerman_steering_principle.html> [cit. 2007-10-4]

[4] URL: <http://cs.wikipedia.org/wiki/Robot> [rev. 2009-3-11]

[5] URL: <http://www.garupner.com>

[6] URL: <http://www.hobbyrobot.cz>

[7] URL: <http://www.snailinstruments.com>

[8] MATOUŠEK, David. Práce s mikrokontroléry ATMEL AVR 4.díl, Praha: BEN, 2006. ISBN 80-7300-174-8

[9] URL: <http://www.hpinfotech.com>

[10] VÁŇA, Vladimír. Mikrokontroléry ATMEL AVR – programování v jazyce C, Praha: BEN, 2003. ISBN 80-7300-102-0

9.2 SEZNAM OBRÁZKŮ:

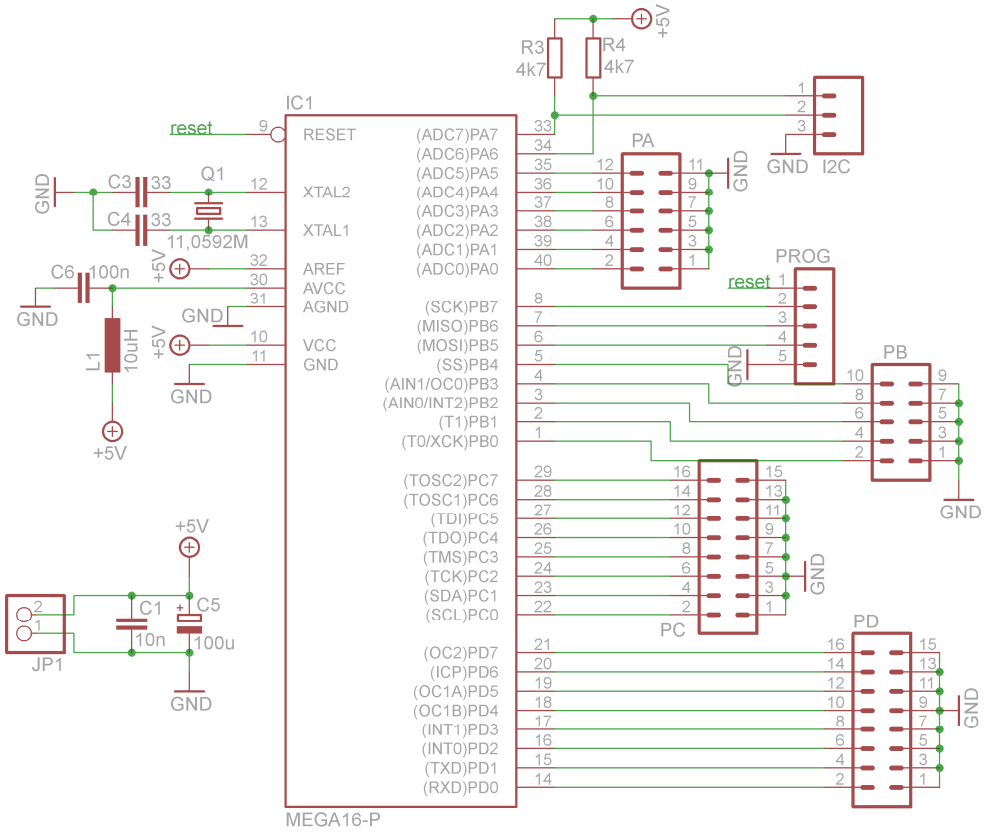
9.3 SEZNAM PŘÍLOH

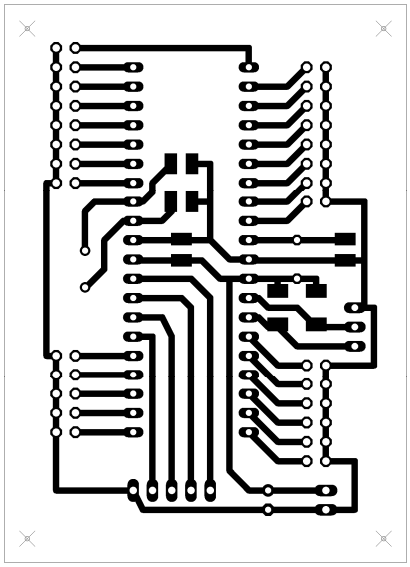
Příloha A – Deska mikrokontroléru

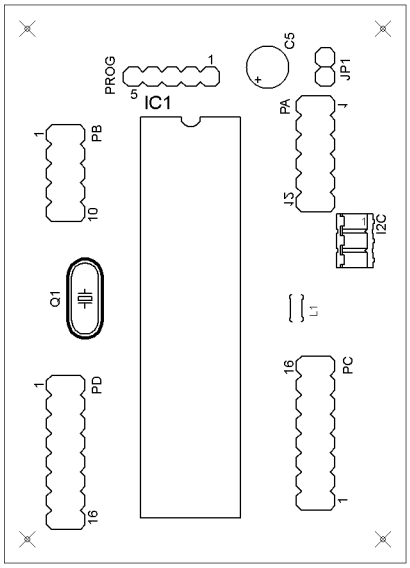
Příloha B – Deska generátoru signálu pro čidla

Příloha C – Deska napájení

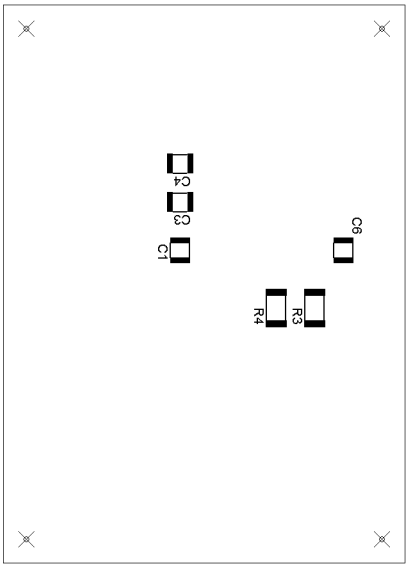
Příloha A – Deska mikrokontroléru







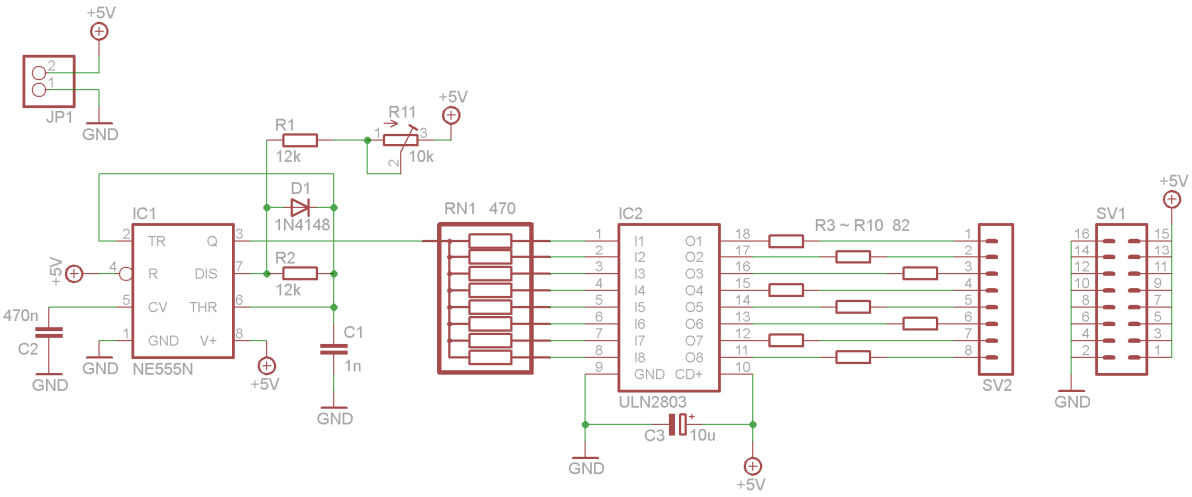
Pohled ze strany součástek

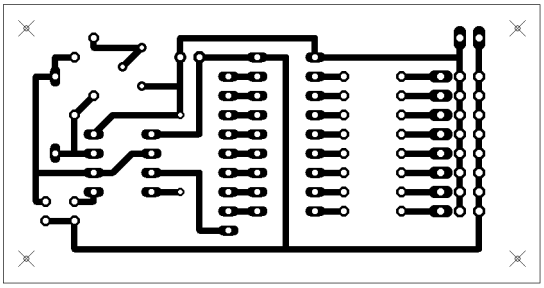


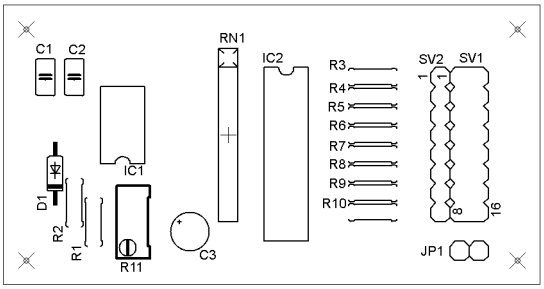
Pohled ze strany spojů

Označení	Součástka	Hodnota	Typ	Výrobce
C1	kondenzátor	10n	1210	Hitano
C3	kondenzátor	33	1210	Hitano
C4	kondenzátor	33	1210	Hitano
C5	kondenzátor	100µ	1210	Hitano
C6	kondenzátor	100n	1210	Hitano
I2C	konektor		PSH02-03PG	Dodavatel GM electronic
IC1	integrovaný obvod		MEGA16-PU24	Atmel
JP1	konektor		S1G2	Dodavatel GM electronic
L1	indukčnost	10µ	09P-100K	Fastron
PA	konektor		S2G16	Dodavatel GM electronic
PB	konektor		S2G16	Dodavatel GM electronic
PC	konektor		S2G16	Dodavatel GM electronic
PD	konektor		S2G16	Dodavatel GM electronic
PROG	konektor		S2G16	Dodavatel GM electronic
Q1	krystal	11,0592M	HC-49U	Vanlong
R3	rezistor	4k7	CR-50S 1/2W	CYM
R4	rezistor	4k7	CR-50S 1/2W	CYM

Příloha B – Deska generátoru signálu pro čidla



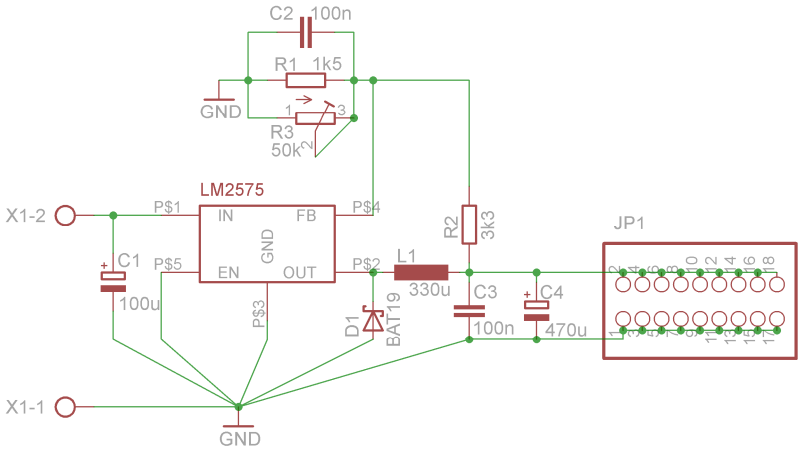


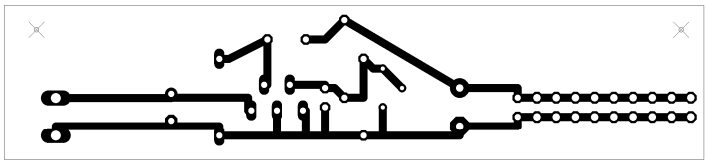


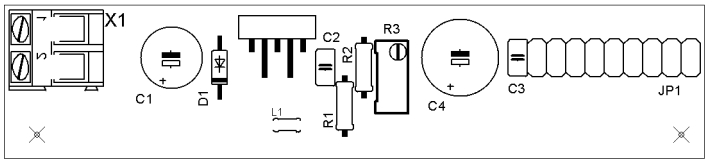
Pohled ze strany součástek

Označení	Součástka	Hodnota	Typ	Výrobce
C1	kondenzátor	1n	0050731102	BCE Sud Passive Component
C2	kondenzátor	470n	0050731472	BCE Sud Passive Component
C3	kondenzátor	10µ	E 10M/25V	Rubycon
D1	rychlá spínací dioda		1N4148	Semtech electonic LTD.
IC1	integrovaný obvod		NE555N	National Semiconductor
IC2	integrovaný obvod		ULN2803A	Toshiba
JP1	konektor		S1G2	Dodavatel GM electronic
R1	rezistor	12k	CR-50S 1/2W	CYM
R2	rezistor	12k	CR-50S 1/2W	CYM
R3	rezistor	82	CR-50S 1/2W	CYM
R4	rezistor	82	CR-50S 1/2W	CYM
R5	rezistor	82	CR-50S 1/2W	CYM
R6	rezistor	82	CR-50S 1/2W	CYM
R7	rezistor	82	CR-50S 1/2W	CYM
R8	rezistor	82	CR-50S 1/2W	CYM
R9	rezistor	82	CR-50S 1/2W	CYM
R10	rezistor	82	CR-50S 1/2W	CYM
R11	trimr	10k	64Y10K	Dodavatel GM electronic
RN1	rezistorová síť	470	RR 8x470A	Dodavatel GM electronic
SV1	konektor		S2G16	Dodavatel GM electronic
SV2	konektor		S1G8	Dodavatel GM electronic

Příloha C – Deska napájení







Pohled ze strany součástek

Označení	Součástka	Hodnota	Typ	Výrobce
C1	kondenzátor	100µ	E 100M/100V	Rubycon
C2	kondenzátor	100n	0050731104	BCE Sud Passive Component
C3	kondenzátor	100n	0050731104	BCE Sud Passive Component
C4	kondenzátor	470µ	E 470M/100V	Rubycon
D1	schottky dioda		BAT19	SGS Thomson
JP1	konektor		S2G18	Dodavatel GM electronic
L1	indukčnost	330µ	09P-330K	Fastron
R1	rezistor	1k5	CR-50S 1/2W	CYM
R2	rezistor	3k3	CR-50S 1/2W	CYM
R3	trimr	50k	64Y50K	Dodavatel GM electronic
U1	integrovaný obvod		LM2575 TO220	National Semiconductor
X1	konektor		ARK710I/2	PTR Messtechnik