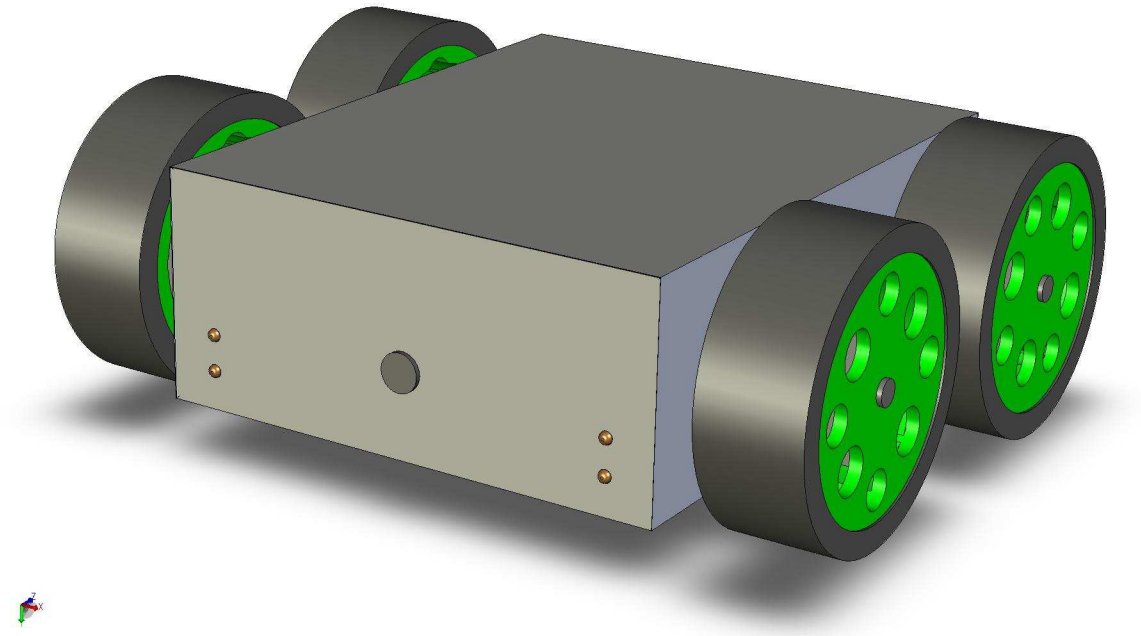
**Středoškolská technika 2009**

**Setkání a prezentace prací**
**středoškolských studentů na ČVUT**

# ROB 2 – ČTYŘKOLOVÝ ROBOT

**Jan Hanč**

Integrovaná střední škola Nová Paka
Kumburská 1136, Nová Paka



**Anotace:**
Tato práce řeší návrh a výrobu jednoduchého, čtyřkolového robota řízeného primárně mikrokontrolérem Atmega16, sekundárně pak ovládacím softwarem v počítači.
Čtyřkolového robota jsem zvolil proto, že je celkem jednoduchý na konstrukci, snadná je jeho odometrie a ovládání a posledním důvodem je, že vypadá celkem zajímavě.

**Jeho základní funkce:**

1) Robot se umí samostatně pohybovat po čáře
2) Jezdí sám náhodně v prostoru, aniž by narážel
3) Je možnost ho vzdáleně řídit skrze počítač
4) Zasílat odometrická data do počítače (stav čidel, natočení serv, apod.)

**Základní součásti:**

1) Mechanická konstrukce robota
2) Řídící elektronika
3) Servomechanismy

*Pozn.: Viz. Příloha č. 1*

**1) Mechanická konstrukce robota**

S ohledem na to, že vlastním několik RC modelů a mám proto k dispozici použitá kola pro modely v měřítku 1/8 (průměr 12cm) zvolil jsem pro robota šasi kvádrového tvaru s rozměry 210x250x95mm (š,h,v). Protože naše škola vlastní tříosou CNC frézku rozhodl jsem se šasi vyrobit z reklamního ABS plastu, jehož šíře je 7mm. Veškeré výkresy jsem navrhl v programu GALAAD.
Výrobní výkresy viz. Příloha č.5

**2) Řídící elektronika:**

**Mikrokontrolér Atmega16**

Jádrem (řídící jednotkou) celého robota je 8-bitový mikrokontrolér Atmega16 v TQFP pouzdře, který je přímo napojen na všechny ostatní součásti robota. Tento mikrokontrolér jsem zvolil z důvodu jeho všestrannosti a jeho snadného naprogramování ve vývojovém prostředí BASCOM, které úzce vychází z jazyka Basic.

**Mikrokontrolér obsahuje tyto pro konstrukci robota důležité součásti:**

8 proporcionálních A/D převodníku (funkčnost je zaručena pouze u pouzdra TQFP)
vlastní USART jednotku pro komunikaci v TTL logice
4 vstupně/výstupní porty po 8 pinech
PWM modulární jádro
Vnitřní napěťovou referenci 1,12V
16 KB vnitřní paměti pro řídící program
512 B EEPROM
Vnitřní kalibrovaný taktovací oscilátor (až do 8Mhz)

**A/D převodník (Analogově digitální převodník):**

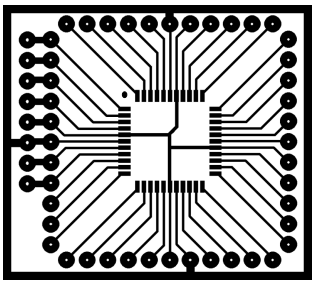
**Je** elektronická součástka určená pro převod spojitého (neboli analogového) signálu na signál digitální. Většina převodníků pracuje s napětím, které převádí na číselnou hodnotu.

**USART jednotka:**

Je to interní součástka mikrokontroléru, která mu umožní v případě převedení napěťových úrovní (např. pomocí i.o. MAX232) komunikovat s počítačem přes sériový port bez zdlouhavého programování softwarového USART rozhraní.

**Osazení:**

Protože tento mikrokontrolér je s tímto pouzdrem velice malý (SMD montáž), musel jsem pro něj vyrobit patici, nejjednodušším řešením bylo navrhnout v programu malý tištěný spoj, který se pak dá snadno osadit na jakoukoli desku plošných spojů. Toto jádro je dostatečně kompaktní a s minimem externích součástek může pracovat samostatně.



**Základní modul:**

Pro osazení modulu s mikrokontrolérem jsem navrhl základní modul, ten je tvořen paticí pro mikrokontrolér, resetovacím obvodem, výkonovým tranzistorem, LED diodou a taktovacím krystalem.

Tento modul zajišťuje propojení jednotlivých částí robota do funkčního celku. Resetovací obvod pravidelně kontroluje vstupní (napájecí) napětí a podle poměru kapacit kondenzátorů C1 a C4 pravidelně resetuje mikrokontrolér (v tomto případě jsem zvolil poměr 1:1 a tím se čas restartování mikrokontroléru pohybuje okolo jedné hodiny). Rezistor $R_1$ slouží spolu s rezistorem R2 k zabránění zkratu při resetu mikrokontroléru tlačítkem S1 či obvodem 7705. Port PB6 je připojen k pinu 4 (RESET) a je určen pro softwarové restartování mikrokontroléru. Pokud jej přepneme do stavu log. nuly mikrokontrolér se restartuje. Protože A/D převodník mikrokontroléru musí být také napájen a vstupní napětí do něj musí být co „nejhladší“, připojíme ho na napájení přes indukčnost L1. Ustalovací kondenzátor není třeba, protože ho již mikrokontrolér obsahuje. Výkonový tranzistor je svou bází připojen na výstup mikrokontroléru PB7 , který prostřednictvím něho spíná silovou zátěž (čidla), která by jinak při své trvalé činnosti zbytečně zatěžovala napájecí zdroj. Na výstup PB4 je připojena LED dioda, která zobrazuje aktuální stav mikrokontroléru. Poslední podstatnou součástí tohoto modulu je taktovací krystal, ten je podle vnitřního nastavení registrů mikrokontroléru buďto nezbytný nebo naopak nezapojený. Paralelně k němu na zem jsou zapojeny filtrační kondenzátory, které zabraňují šíření oscilací dále do obvodu. Piny PD0 a PD1 jsou použity jako RX a TX (vysílání a příjem) dat z mikroprocesoru a jsou spojeny dvoužilovým kabelem s 12 a 14 v modulu XTR-903. Řídící serva jsou zapojena do pinů 1 a 2 zdířky SV2.

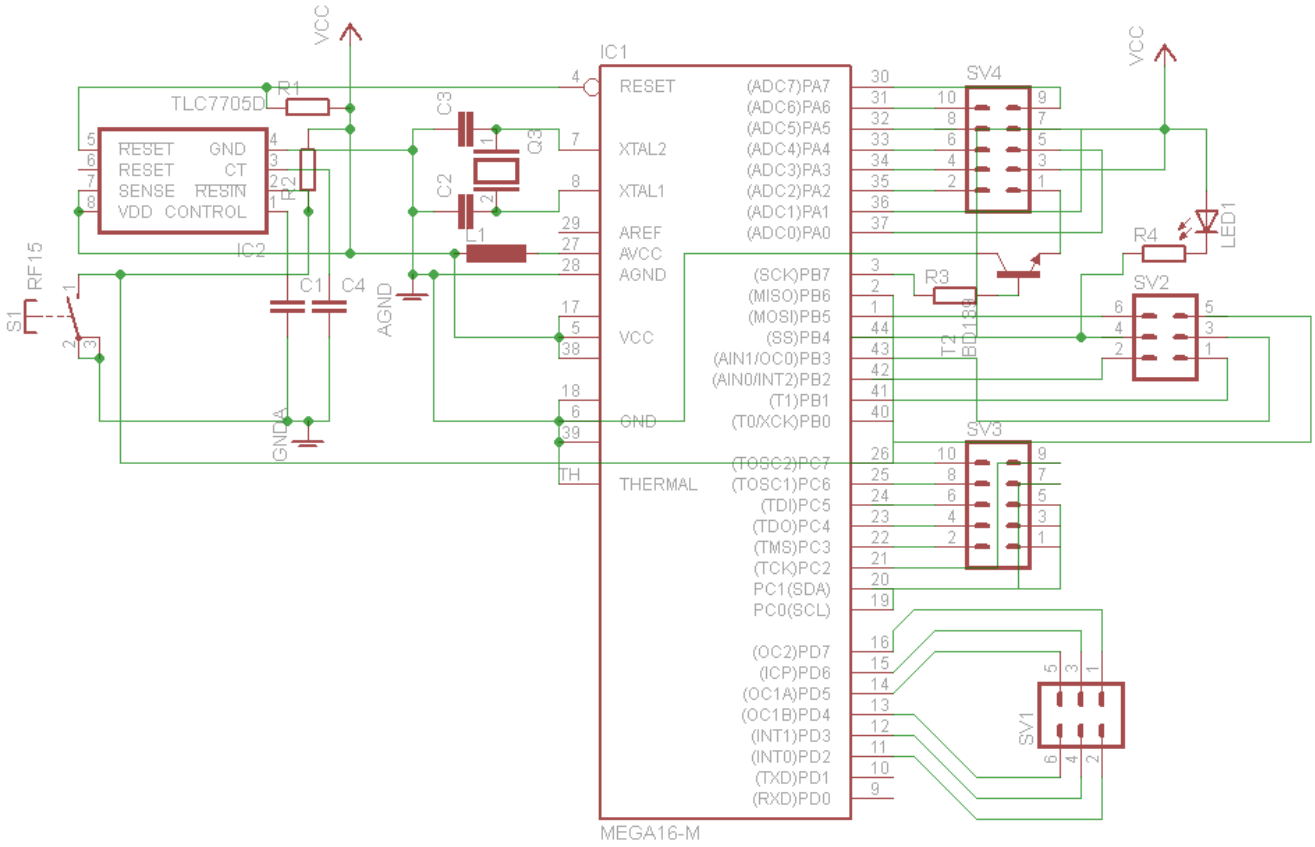



*Schéma č.1*

**Zapojení odrazových čidel:**
LED diody jsou zapojeny přes předřadné rezistory přímo na svorky SV1-1, SV1-2. Svorka SV1-1 je připojena přímo na napájecí napětí a svorka SV1-2 na výstup mikrokontroléru PB7 (ten když sepne, připojí svorku SV1-2 k zemi a tím začne protékat obvodem proud – čidla se aktivují). Fototranzistory jsou připojeny kolektory přes trimry na kladné napájecí napětí. Jejich emitory jsou připojeny přímo na výstupy SV5, které přímo vedou do portů PA v mikrokontroléru. LED diody svým svitem budí tranzistory a tím mění jejich přechodový odpor. Tím pak vzniká větší či menší úbytek na jejich přechodu, který je snímán a převáděn A/D převodníkem.

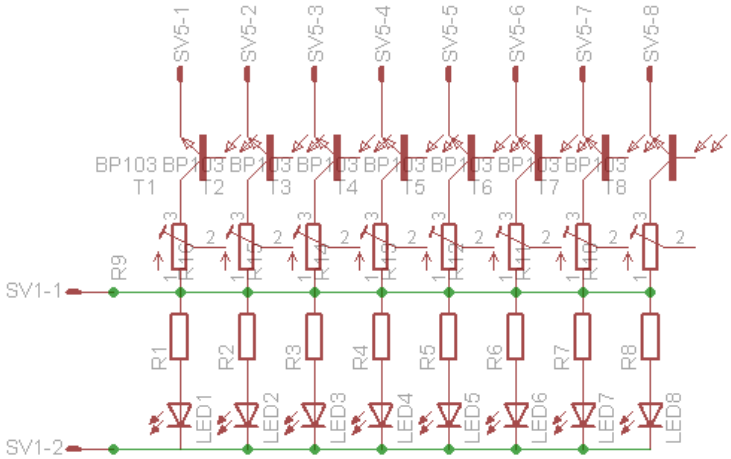


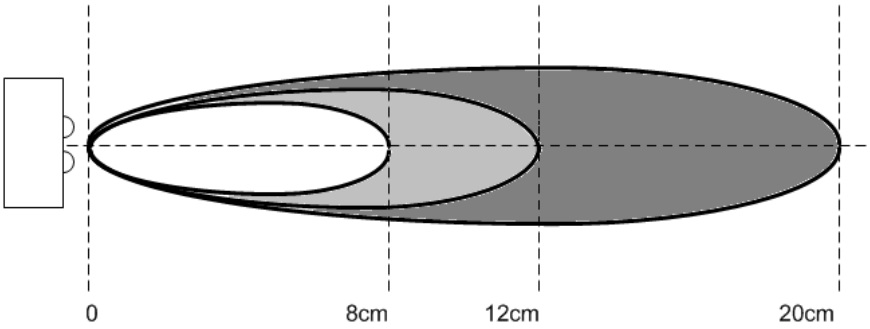

*Schéma č.2*

**Princip odrazového čidla s použitím IR prvků:**
IR Dioda vysílá kontinuální IR signál (svítí). Tento signál se odráží od neprůhledných materiálů zpět do snímače tedy do IR fototranzistoru, ten podle „množství dopadajícího světla (fotonů) mění svojí propustnost kolektor – editor. Tranzistor je zapojen jako prvek děliče a proto je možné na jeho výstupech kolektor – editor měřit úbytek napětí, který je přímo úměrný vzdálenosti, barvě, bohužel ale také na intenzitě okolního světla. Měřením jsem došel k závěru, že mezní vzdálenost reakce čidla je 8cm. V této vzdálenosti je překážka spolehlivě zjištěna což vyplívá z následující tabulky.

*Tabulka č.1*

| Číslo měření | Vzdálenost předmětu (cm) | Barva předmětu | Úbytek na FT (V) | Po A/D převodu | Použitý sek. Rezistor |
|---|---|---|---|---|---|
| 1. | 50 | Černá | 0,15 | 31 | 312KΩ |
| 2. | 50 | Bílá | 0,45 | 92 | 312KΩ |
| 3. | 30 | Černá | 0,75 | 154 | 312KΩ |
| 4. | 33 | Černá | 0,77 | 158 | 312KΩ |
| 5. | 15 | Černá | 1,42 | 291 | 312KΩ |
| 6. | 15 | Bílá | 1,95 | 399 | 312KΩ |
| 7. | 10 | Černá | 1,75 | 358 | 312KΩ |
| 8. | 8 | Černá | 2,67 | 547 | 312KΩ |
| 9. | 6 | Černá | 1,22 | 250 | 312KΩ |
| 10. | 4 | Černá | 0,46 | 94 | 312KΩ |
| 11. | 3 | Černá | 0,06 | 12 | 51KΩ |
| 12. | 3 | Bílá | 3,62 | 741 | 51KΩ |

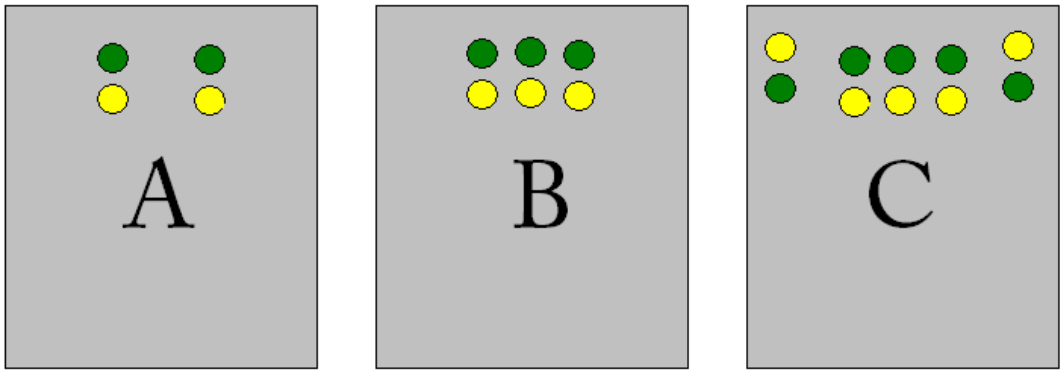
**Přesnost určení vzdálenosti:**
Protože intenzita vyzařovaného paprsku IR diodou je ovlivňována vzdáleností a okolními vlivy, přesnost čidla klesá s narůstající vzdáleností jak je vidět na obrázku níže.




*Pozn.:*
*Bílá zóna je okolí, ve kterém určí čidlo vzdálenost s přesností na desetiny centimetru.*
*Světle šedá zóna, ve které čidlo určí vzdálenost s přesností jednotek centimetrů.*
*Tmavě šedá zóna je okolí, ve kterém čidlo určí vzdálenost v přesnosti na desítky centimetrů.*

**Čidlo čáry:**
Aby robot mohl sledovat čáru nakreslenou na zemi, musí mít k tomuto účelu uzpůsobená čidla. Pro tento účel jsem zvolil kombinaci fototranzistoru a infradiody pracující na 940nm
Pro opravdu kvalitní znázornění polohy čáry je zapotřebí alespoň 3 těchto čidel. Já jsem z důvodu jednoduchosti a jednoduššího technického provedení zvolil variantu pouze s dvěma čidly…levá a pravá strana čáry. Tímto se celý proces značně zjednodušil. Pokud je řídící program napsán správně, tak by měl být i zcela funkční.

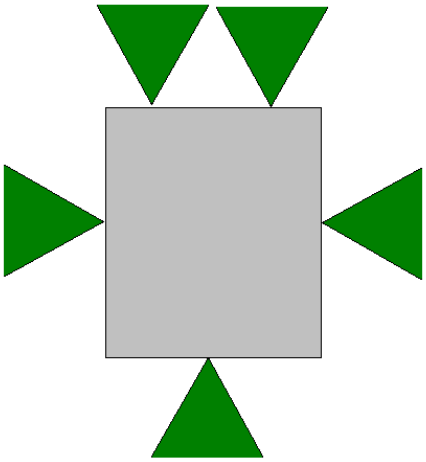
**Příklady rozmístění čárových čidel:**




*A) Nejjednodušší technika snímání čáry – tzv. jízda opilce na kole*
*B) Již docela kvalitní rozpoznání přesné pozice čáry, před každou jízdou nutná kalibrace šířky čáry*
*C) Nejlepší snímání čáry, nutná kalibrace šíře čáry, možnost orientace v bludišti, náročné na součástky*

**Čidlo překážky:**
Toto čidlo je konstruováno stejným způsobem, ale na rozdíl od čidel čáry se u těchto čidel mění vzdálenost, barva nemá při větších vzdálenostech takový vliv (snad kromě černé). Měřením (1 až 9 v tabulce č.1) jsem dospěl k závěru, že mezní vzdálenost je 8cm. Na tuto vzdálenost je mikrokontrolér schopen určit stoprocentně povahu překážky.

*Pozn.: Na obrázku níže je znázornění pokrytí robota čidly, zelená barva ukazuje místa kde čidlo rozpozná překážku.*



**Zdroje napětí:**
Jako zdroj energie jsem použil články z repasovaných notebookových baterií. Protože pohonná serva mají velký proudový odběr (celkem okolo 3A podle jejich zatížení) a z toho vyplívající velké rušení, zvolil jsem použití dvou nezávislých zdrojů spojených v jednom bodě záporným pólem jejich napájecích baterií.(Tímto odpadá nutné použití signálové země pro řízení serv.)

První zdroj je určen pro napájení serv nebo jiných silových (s odběrem od 500mA) spotřebičů umístěných v robotu. Jako zdroj napětí jsou využity čtyři články 5,5V z repasovaných baterii notebooků, napájecí napětí je tedy přibližně 11-12V, maximální proud který je baterie schopna vyvinout je cca 16A, což je pro tuto aplikaci více než dostačující. Jako stabilizační a usměrňovací část je použit stabilizátor 78H05. Tento stabilizátor je velice jednoduchý na zapojení, protože se spokojí se třemi elektrolytickými kondenzátory. Jelikož jsem potřeboval pro serva napájecí napětí alespoň 5,5V (Z důvodu vyšší síly serva), zapojil jsem mezi zem vstupu a zem stabilizátoru diodu 1N4007, která svým úbytkem zvýší výstupní napětí o 0,7V tj. na 5,6V. Svorky SV1 slouží jako vstupní, na svorky SV2 a SV3 se připojují serva a na svorky SV4 se připojuje signál z mikrokontroléru pro řízení serv.

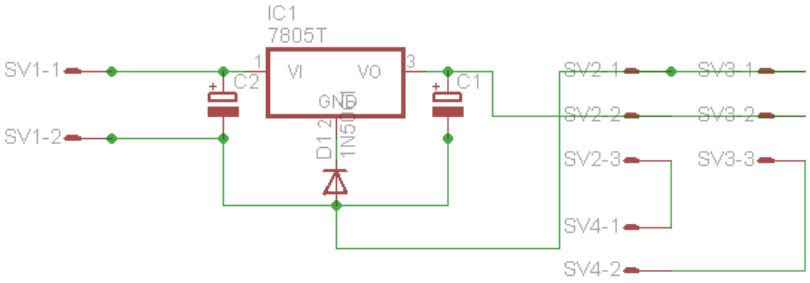



*Schéma č.3*

Druhým zdrojem je zdroj pro vysílací část a samozřejmě pro mikrokontrolér. Jako napájecí médium jsem použil 3 články 3,7V řazené seriově. Tímto jsem se dostal na napájecí napětí cca 11V. Tento zdroj je tvořen dvěma stabilizátory 7805. Dvěma stabilizátory proto, že je třeba oddělit mikrokontrolér od vysílací části, nebo také z důvodu připojení dalšího mikrokontroléru v režimu SPI, pro který je nutné spojení zemí pouze v jednom bodě.

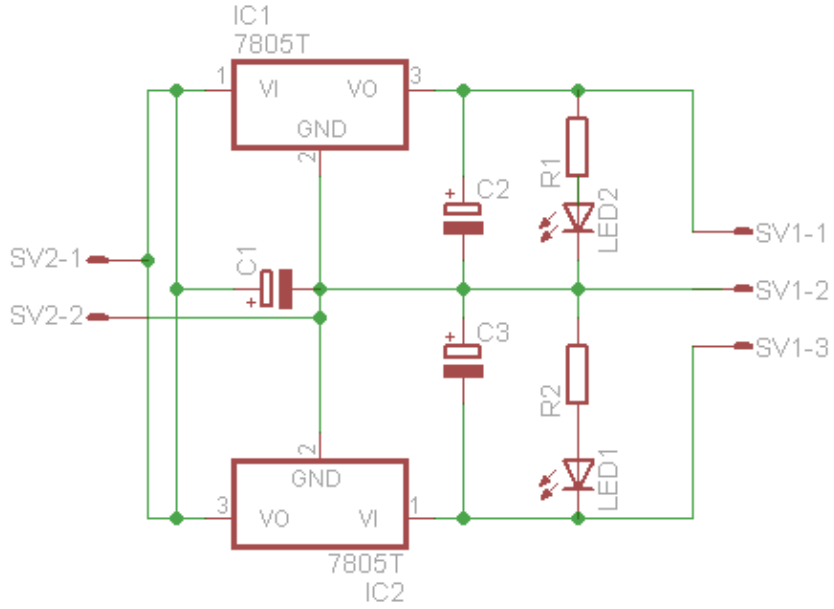
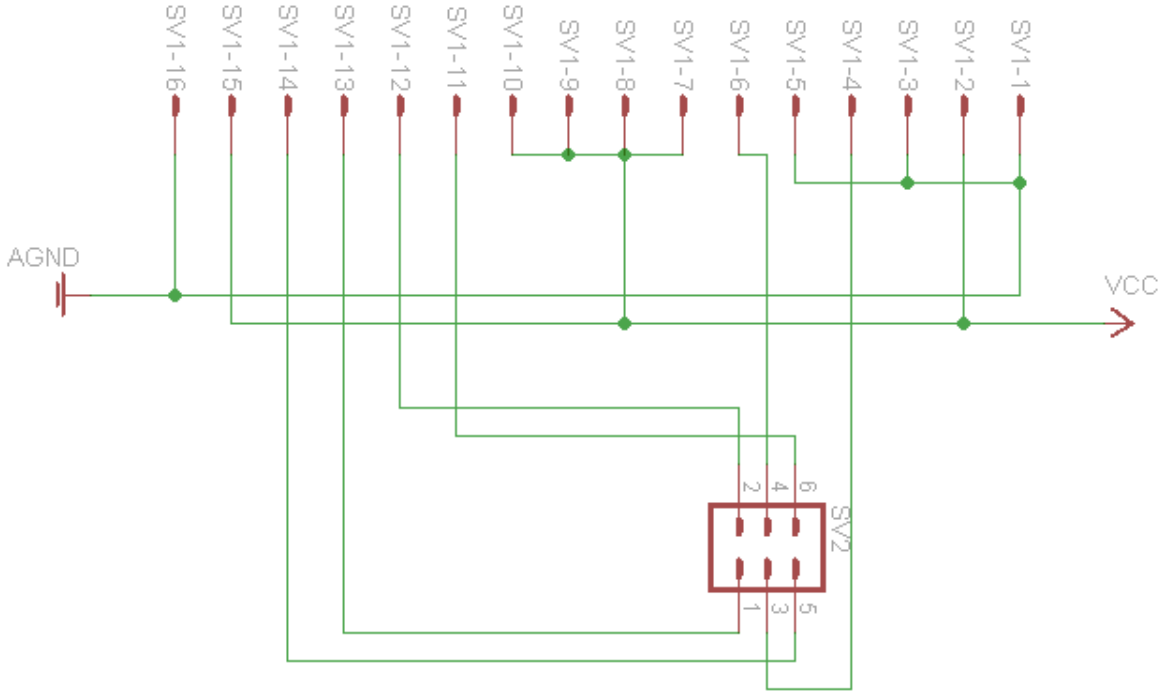



*Schéma č.4*

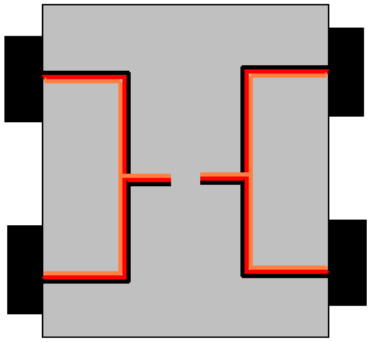
**Displej:**
Zakoupil jsem šestnáctiznakový displej se čtyřmi řádky jako hotový modul s vlastní řídící elektronickou (MC1604). Pro jeho řízení jsem použil port PB v mikrokontroléru.

Propojení s mikrokontrolérem:

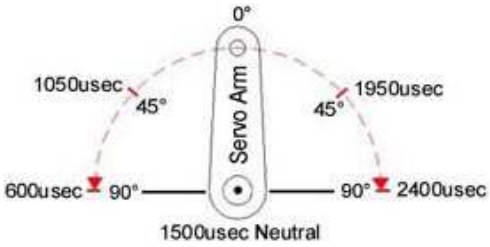



Svorky SV1 jsou svorky řadiče displeje, svorky SV2 jsou pak výstupy z portu D mikrokontroléru.

**Pohon:**
Jako pohon pro robota jsem zvolil čtyři modelářská serva Hitec 422 (tah 4kg/cm), což bylo po několika pokusech očividně málo a proto jsem v další fázi tato serva vyměnil za silnější serva Hextronik HX12K (tah 12kg/cm). U robota je potřebné ovládat vždy celou stranu (náhon obou kol) stejnoměrně, stejným směrem a současně. Proto jsem spojil na každé straně řídící vodiče (u většiny serv oranžové nebo žluté) a signál vysílaný z mikrokontroléru jde tedy přímo do obou serv bez nutnosti generovat dva signály.



**Teorie řízení serva:**
Do každého serva vede třížilový kablík a jednotlivé žíly jsou v barvě černá, červená a žlutá. Černá je nula - GND , červená je Vcc - 5-6V . Žlutá (někdy žlutooranžová) je ovládání serva. Sem musí mikrokontrolér nebo RC přijímač modelářské soupravy posílat v pravidelném intervalu 50Hz (každých 20ms) kladný pulz, na jehož šířce závisí natočení serva. V drtivé většině případů je při délce pulzu 1500µs servo v neutrální pozici (střed). Se zkracováním pulzu až k 600µs se servo natáčí vlevo, s prodlužováním až k 2400µs pak vpravo. Je dobré vědět, že každé servo má jiné krajní hodnoty a není vůbec dobré, pokud je budeme překračovat. Mechanika serva je pak nepříjemně zatěžovaná kroutícím momentem serva a může se v takové situaci lehce poškodit. Právě proto je nutné nastavit tyto hodnoty pro každé servo zvlášť.



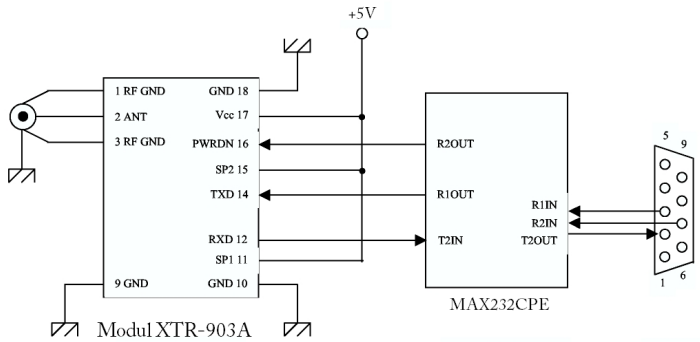

*Pozn.: Na obrázku výše je zobrazen stav natočení serva vzhledem k délce řídícího pulsu*

V dnešní době se dá drtivá většina modelářských serv upravit tak, aby se mohlo otáčet kontinuálně, neboli stále dokola. Úprava spočívá v odstranění vymezovacího sloupku na výstupním hřídeli serva a samozřejmě na vyřazení vyrovnávacího potenciometru.

*Pozn.: Viz příloha č.2*

**Komunikace:**
Komunikace mezi mikrokontrolérem a počítačem probíhá mezi dvěma totožnými moduly Aurel XTR-903A, ty pracují na frekvenci okolo 414MHz. Tato frekvence je zatím málo rušena a tyto zařízení tak mají velký dosah (cca 150m) a spolehlivost.

Jako první jsem musel navrhnout část pro počítač, jelikož je nutné každý modul před jeho funkčním zapojením naprogramovat a naladit.




Podle obecného schématu jsem navrhl desku plošných spojů. Piny PS1 a SP2 se konfigurují komunikační rychlost těchto modulů. Já jsem pro svoji potřebu zvolil rychlost 9600bd, i když modul dokáže pracovat s rychlostí 19200bd. Bohužel pak ale ztrácí potřebnou stabilitu a kvalitu spojení. To se zdařilo a modul jsem bez problémů osadil, další metou bylo naprogramování obou dvou modulů přes tento interface. Tady jsem se setkal s prvním problémem. Modul se mnou nekomunikoval. Nakonec jsem z „nouze“ použil program z knížky o elektronice B.Kainka, kterému se říká Terminál. Ten umí posílat jednotlivé bity bez prodlevy a tím se podaří zpřístupnit servisní menu modulu.

**Popis konfigurace modulů XTR-903A:**

Pro spuštění servisního módu je nutné poslat „+++“ a to bez prodlevy či přídavných znaků.
Poté již mikrokontrolér ovládáte příkazy ATS1-16, ATCC, ATWR.

Ukázka komunikace:
PC => +++
MODUL => OK
PC => ATS2=5
MODUL => OK
PC => ATWR
MODUL => OK
PC => ATCC
MODUL => OK

*Pozn.: Servisní komunikace je tímto ukončena a začne komunikace mezi moduly, tedy standardní sériový přenos*

**Popis softwarové funkce mikrokontroléru (veškeré tyto funkce jsou přímo závislé na konkrétním programu, který je v mikrokontroléru nahrán):**

Mikrokontrolér se v tomto robotu stará o řízení servomotorů, snímání hodnot z odrazových čidel, komunikaci s osobním počítačem a jeho řídícím softwarem.

Mikrokontrolér se nejdříve zinicializuje, poté zkontroluje, zda je připojen transceiver (modul XTR-903A). Pokud ano, pokusí se začít komunikaci s počítačem, vyšle startovací bit (10), a pokud je počítač a řídící software připraven, odešle počítač startovací bit (10) zpět. Když tato kontrola spojení proběhla úspěšně, mikrokontrolér se přepne do tzv. poslouchacího režimu, kdy čeká na příkaz z řídícího softwaru. Obdrží-li nějaký bit, porovná ho v databázi a pokud jej v ní najde, provede odpovídající příkaz.

Seznam řídících kódů:
55 – Zapni režim Čára
56 – Zapni režim Automat
57 – Zapni režim Ruční řízení
58 – Ukaž stav čidel
59 – Servisní režim
60 – Kalibrace čidel
255 – Startovací bit
254 – Ukonči úkon a vrať se zpět do základu
253 - Pauza
252 – Přepni se do režimu spánku
251 – Překážka vpředu
250 – Překážka ze stran

**Princip režimu Čára (55)**
Pokud je robot do tohoto režimu přepnut, zapne všechna odrazová čidla, zkontroluje zda je baterie dostatečně nabitá a vyčká, dokud neobdrží startovací bit z počítače. Když obdrží startovací bit začne vysílat signály do řídících serv (tyto signály musí mít reverzní charakter, aby se robot nezačal točit dokolečka). Touto rychlostí pokračuje, dokud některým z čárových čidel nenajede na čáru, ta protože je tmavá a má malou odrazivost způsobí požadovanou reakci a mikrokontrolér zareaguje zpomalením či zastavením protilehlých serv. Pokud však robot najede svými čidly na rovnoběžnou čáru (obě čidla jsou nad čárou), mikrokontrolér vyšle signál pro zastavení serv a společně s ním také signál pro počítač, zda má pokračovat. Dostane-li pauzovací bit (253), bude pokračovat, jestliže dostane cokoli jiného, režim čáry ukončí.

**Princip režimu Automat (56)**
Je-li robot přepnut do režimu automat, zapne všechna odrazová čidla, zkontroluje baterii a vyčká, dokud neobdrží startovací bit(255). Když tento bit obdrží, pošle sekundárnímu mikrokontroléru přes softwarový UART bit 56 (pokud je zapojen, když sekundární mikrokontrolér zapojen není, generuje tato čísla sám), čímž u něj spustí generování náhodných čísel pomocí funkce Random (generování náhodných čísel) v pravidelných intervalech 50ms. Ten opakovaně posílá tyto bity řídícímu mikrokontroléru, který v intervalech stanovených svojí funkcí Random snímá tyto hodnoty a s nimi pak řídí jednotlivá serva. Takto se celý proces náhodně opakuje. Před každou smyčkou mikrokontrolér zkontroluje stav odrazových čidel, zda není v blízkosti robota nějaká překážka. Pokud ano nastaví serva opačným směrem než je aktivované čidlo. Sepnou-li ale obě dvě čidla přední,

zastaví se a čeká, dokud není překážka odstraněna. Když mikrokontrolér obdrží stopbit (254) okamžitě práci ukončí a zašle sekundárnímu mikrokontroléru softwarovým UARTem stopbit. Tím se oba dva mikrokontroléry přepnou do klidového režimu. Spojením dvou mikrokontrolérů je dosaženo opravdu náhodného pohybu robota, protože funkce random čerpá z výrobních a provozních parametru konkrétního čipu (pokud je mi známo, tak ze sériového čísla a strojového času mikrokontroléru).

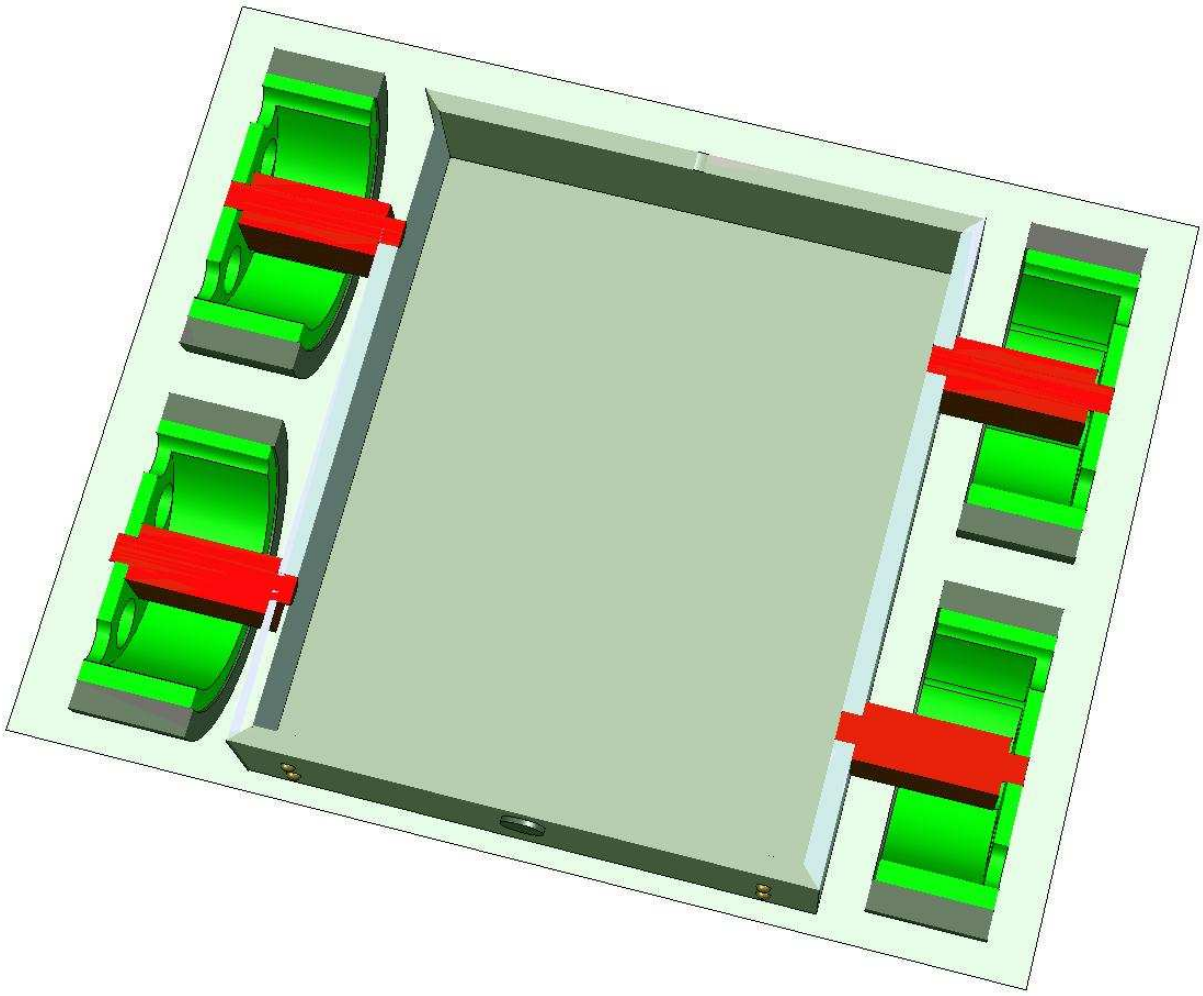
**Princip režimu Ruční řízení (57)**
Dalším módem, který tento robot umí je manuální řízení. To je provedeno tak, že robot stále čeká na bit 57, když je jím tento bit přijat, čeká na další 2 bity, které jsou již převedeny přímo na hodnoty řídící pohybová serva. Takto pokračuje stále dokola, dokud neobdrží stopbit. Na začátku každé smyčky kontroluje stav jednotlivých serv.

Příklad komunikace počítače:

| | |
|---|---|
| ROBOT | 10 // pokus o komunikaci |
| PC | 10 // odpověď na tento pokus |
| ROBOT | čeká… |
| PC | 55 // příkaz o přepnutí do režimu čára |
| ROBOT | 255 // překážka vpředu |
| PC | 253 // pokračuj |
| ROBOT | 253 // pokračovat? (najel jsem na čáru) |
| PC | 253 // pokračuj |
| ROBOT | 253 // pokračovat? (najel jsem na čáru) |
| ROBOT | čeká… |
| PC | 254 // ukonči tento režim |

**Řídící program v počítači:**
Tento řídící program jsem navrhl pro zjednodušení práce s robotem, jediné co tenhle program dělá že v pravidelných intervalech 25ms odesílá řídící kódy které byly již zmíněny výše.
*Ukázka programu viz. Příloha č. 3*

**Příloha č.1**



*Pozn.: Tento obrázek je rendrovaný v počítači a je pouze přibližný*
*Zeleně jsou zobrazena kola*
*Červenou barvou řídící servomotory*
*Šedou barvou je znázorněno šasi.*

**Příloha č.2**
**Postup úpravy serva pro kontinuální pohyb:**

1. Servo Hitec HS-422 před úpravou


2. Servo zespodu, povolíme všechny čtyři šrouby a vytáhneme je


3. Zespodu odpadne spodní víko, uvidíme desku elektroniky. Zde se nebude nic dělat, otočíme servo a sejmeme vrchní díl

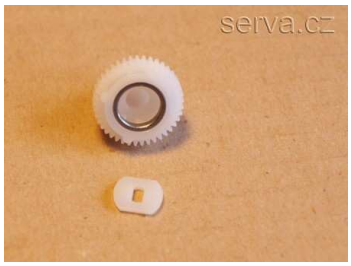

4. Odkryté servo, vidíme "dobře promazané" převody. Při manipulaci se snažíme pokud možno co nejméně setřít mazací gel.


5. Sundáme poslední (největší) ozubené kolo převodovky.


6. Vyjmeme z něho ocelový kroužek (vnější prstenec kluzného ložiska)

7. Vyjmeme plastový unašeč potenciometru


8. Vložíme zpět ocelový kroužek. **Tím** jsme docílili toho, že servo hlavní osa serva není mechanicky spojená s osou potenciometru

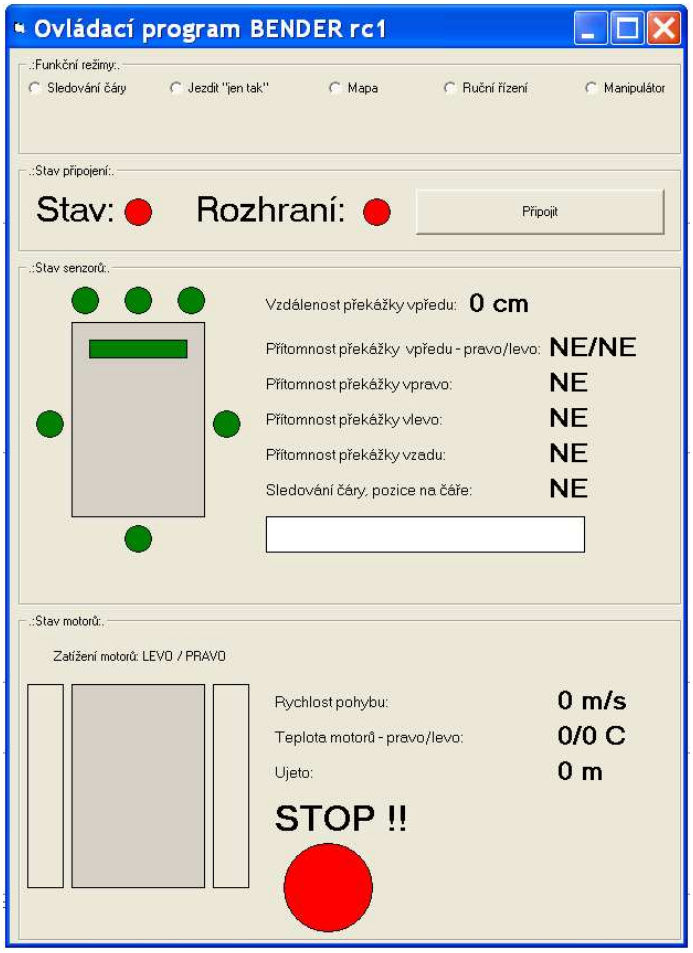



9. Z ozubeného kola odstraníme ostrým nožem zarážku, viz obrázek, vpravo kolečko po úpravě


10. Nyní bychom servo měli zapojit, a délku pulzu nastavit na 1500us (střed). Osou potenciometru jemně doladíme tak, aby se motor serva zastavil.


11. Opatrně nasadíme zbytek převodovky, servo zakrytujeme a máme hotovo, ze serva je hnací motor

**Příloha č. 3**


Ovládací program BENDER rc1
.:Funkční režimy:.
Sledování čáry
Jezdit "jen tak"
Mapa
Ruční řízení
Manipulátor
.:Stav připojení:.
Stav:
Rozhraní:
Připojit
.:Stav senzorů:.
Vzdálenost překážky vpředu: 0 cm
Přítomnost překážky vpředu - pravo/levo: NE/NE
Přítomnost překážky vpravo: NE
Přítomnost překážky vlevo: NE
Přítomnost překážky vzadu: NE
Sledování čáry, pozice na čáře: NE
.:Stav motorů:.
Zatížení motorů: LEVO / PRAVO
Rychlost pohybu: 0 m/s
Teplota motorů - pravo/levo: 0/0 C
Ujeto: 0 m
STOP !!

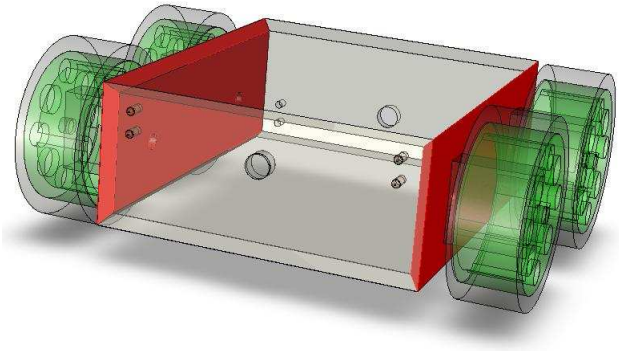
**Příloha č. 4**
**SEZNAM POUŽITÝCH SOUČÁSTEK**

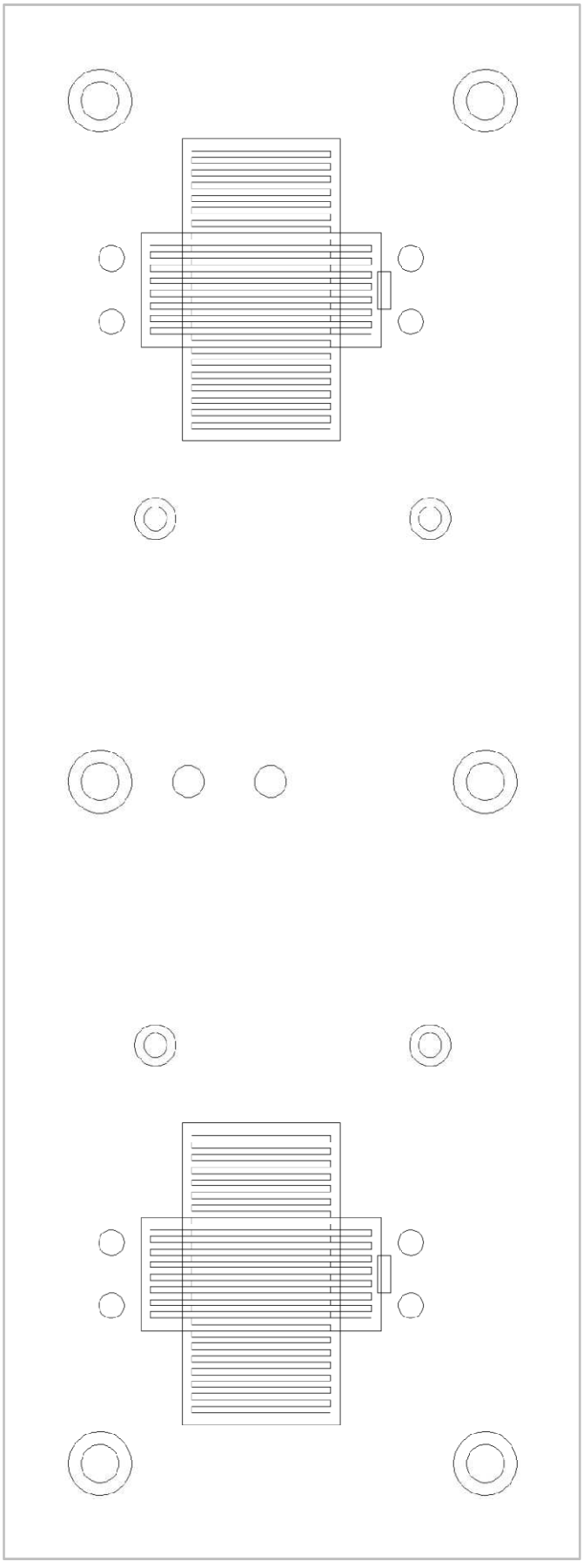
Schéma č. 1
C1 – kapkový kondenzátor 100n
C2 – kapkový kondenzátor 22p
C3 – kapkový kondenzátor 22p
C4 – kapkový kondenzátor 100n
R1 – Rezistor 0,1W 10k
R2 – Rezistor 0,1W 10k
R3 – Rezistor 0,1W 4k4
R4 – Rezistor 0,1W 2k2
T2 – Tranzistor BD139
Q3 – Krystal 1Mhz – 16Mhz (Pozor!!!! Při zapojení krystalu 16Mhz použít Datasheet)
LED1 – LED dioda 3mm červená
IC1 – ATMega16 16PU

Schéma č.2
LED (1 – 8) – IRS5 (IR dioda 940nm)
R(1-8) – Rezistor 47R
R(9-16) – Plochý odporový trimr 1M
T(1-8) – IRE5 – Fototranzistor 850-940nm

# VÝROBNÍ VÝKRESY:

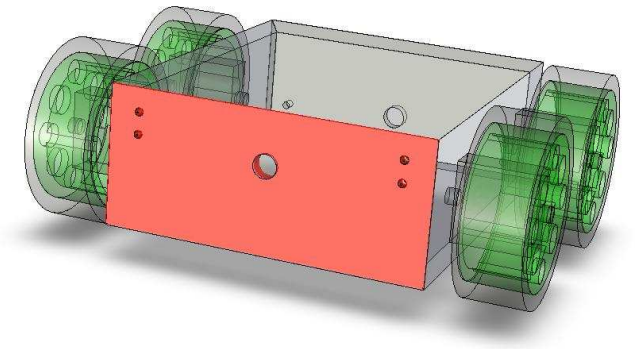
***Výkres boční strany robota:***

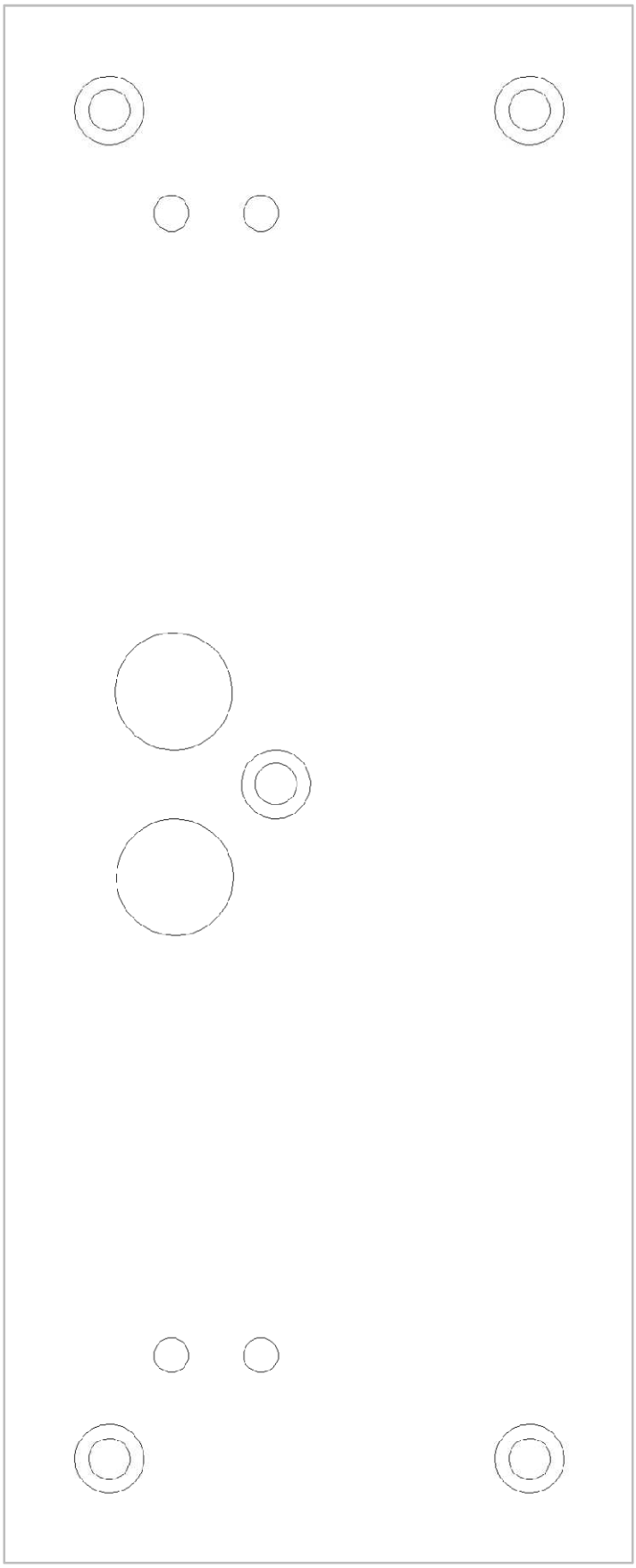




# VÝROBNÍ VÝKRESY:

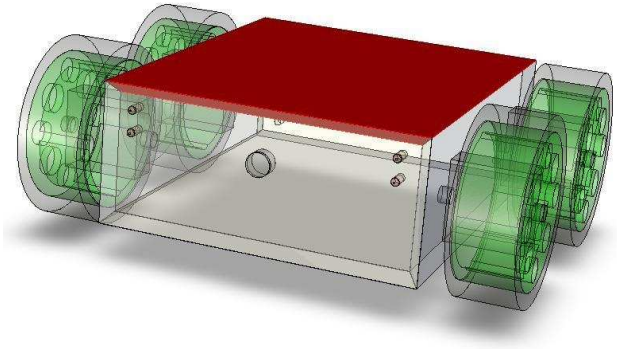
***Výkres přední strany robota:***

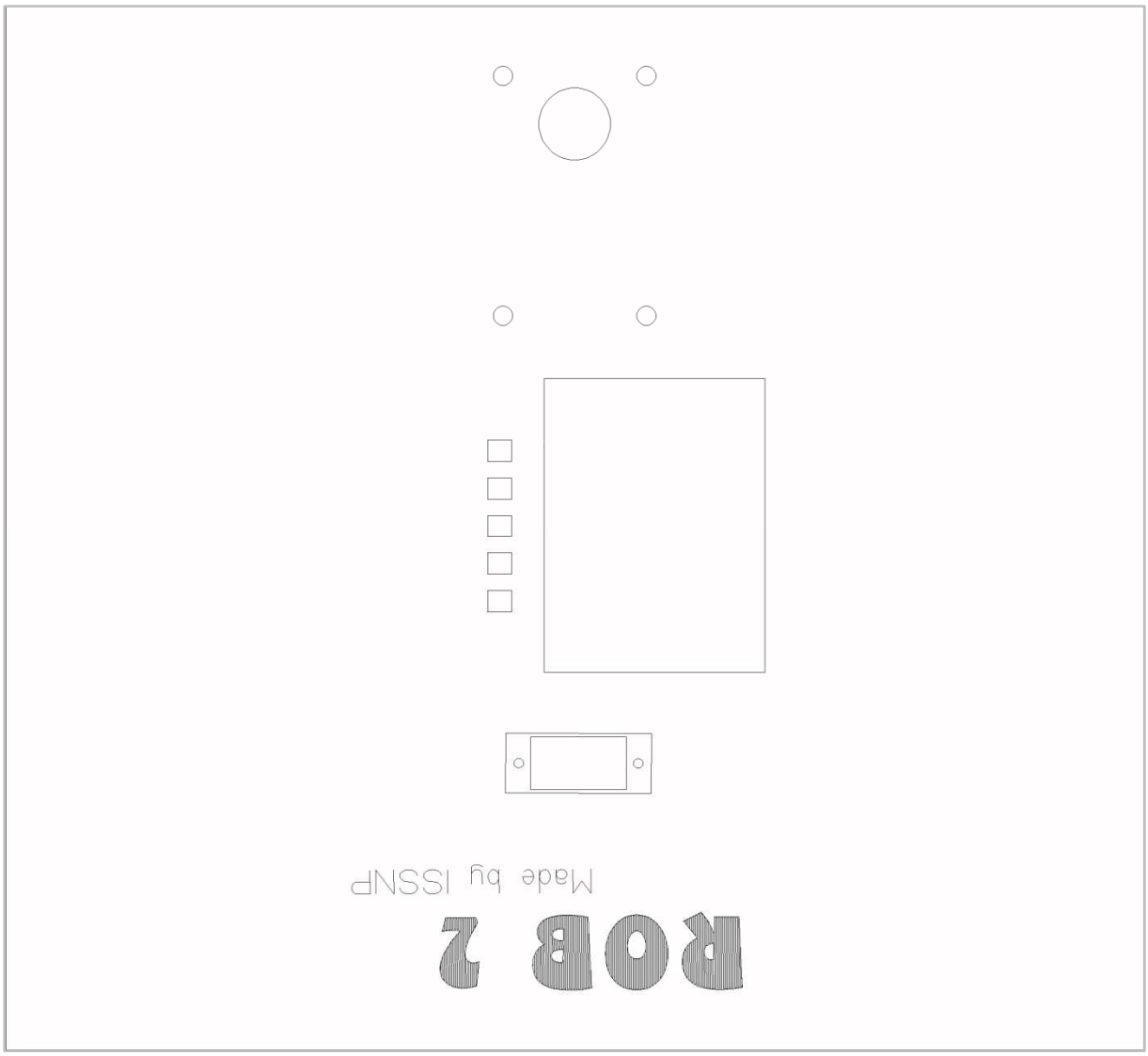




# VÝROBNÍ VÝKRESY:

***Výkres přední strany robota:***





## Příloha č. 5

### *Ukázka firmware řídícího mikrokontroléru v.1.02:*

```
$regfile = "m16def.dat"
$baud = 9600

Config Adc = Single , Prescaler = Auto , Reference = Avcc
Config Servos = 2 , Servo1 = Portb.5 , Servo2 = Portb.6 , Reload = 20

Config Portb = Output
Config Portc = Input
Config Portd = Output
Enable Interrupts
Dim Mcidlopredek1 As Eram Byte At 1
Dim Kcidlopredek1 As Eram Byte At 2
Dim Mcidlopredek2 As Eram Byte At 3
Dim Kcidlopredek2 As Eram Byte At 4
Dim Mcidlozadek As Eram Byte At 5
Dim Kcidlozadek As Eram Byte At 6
Dim Mcidlocara1 As Eram Byte At 7
Dim Kcidlocara1 As Eram Byte At 8
Dim Mcidlocara2 As Eram Byte At 9
Dim Kcidlocara2 As Eram Byte At 10
Dim Mcidlolevo As Eram Byte At 11
Dim Kcidlolevo As Eram Byte At 12
Dim Mcidlopravo As Eram Byte At 13
Dim Kcidlopravo As Eram Byte At 14
Dim S1s As Eram Byte At 15
Dim S2s As Eram Byte At 16
Dim I As Byte
Dim B As String * 15
Dim Temp As String * 15
Dim C As Byte
Dim D As Byte
Dim Servo1 As Byte
Dim Servo2 As Byte
Dim Servo3 As Byte
Dim Servo4 As Byte
Dim Servo5 As Byte
Dim Servo6 As Byte
Dim Temps As Integer
Dim Cidlopredek1 As Integer
Dim Cidlopredek2 As Integer
Dim Cidlozadek As Integer
Dim Cidlocara1 As Integer
Dim Cidlocara2 As Integer
Dim Cidlolevo As Integer
Dim Cidlopravo As Integer
Dim Sd As Byte
Dim Menu As Byte
Start Adc
Cursor Off
Cls
Lcd "    **ROB 2*     "
Servo(1) = 75
Servo(2) = 75
Do
Gosub Prectiadc
```

```
If Ischarwaiting() = 1 Then
   Sd = Inkey()
   End If
Lowerline
Lcd "      MENU      "

Select Case Sd
Case 55
Locate 3 , 1
Lcd "      Cara"
Gosub Cara
Case 56
Locate 3 , 1
Lcd "  Volna jizda"
Gosub Jizda
Case 57
Locate 3 , 1
Lcd "      Ruka"
Gosub Ruka
Case 58
Gosub Mapa
Case 59
Gosub Resetovat
Case 60
Locate 3 , 1
Lcd "    Stav cidel"
Gosub Sc
Case 61
Gosub Automat
Case 62
Locate 3 , 1
Lcd "   Kalibrace cidel"
Gosub Kalibracec
Case Else
End Select
Loop

Prectiadc:
Cidlopredek1 = Getadc(3)
Cidlopredek2 = Getadc(7)
Cidlozadek = Getadc(6)
Cidlolevo = Getadc(4)
Cidlopravo = Getadc(1)
Cidlocara1 = Getadc(0)
Cidlocara2 = Getadc(2)
Return

Zastav:
Servo(1) = 75
Servo(2) = 75
Return

Cara:
Servo(1) = 120
Servo(2) = 30
Do
Gosub Prectiadc
If Cidlocara1 < 350 And Cidlocara2 < 250 Then
Servo(1) = 75
Servo(2) = 75
```

```
Locate 4 , 1
Lcd "    Konec - STOJIM"
Return
End If
If Cidlocara1 < 350 Then
Servo(1) = 75
Servo(2) = 30
Else
Servo(1) = 120
Servo(2) = 30
End If
D = Kcidlocara2
If Cidlocara2 < 350 Then
Servo(1) = 120
Servo(2) = 75
Else
Servo(1) = 120
Servo(2) = 30
End If

Loop

Jizda:
Do
Inputbin Sd
If Sd = 56 Then
Inputbin Servo1
Inputbin Servo2
Servo(1) = 75 + Servo(1)
Servo(2) = 75 - Servo(2)
Else
Gosub Zastav
End If
Loop

Spat:
Powerdown
Return

Ruka:
Return

Mapa:
Return

Resetovat:
Return

Sc:
Do
If Ischarwaiting() = 1 Then
   Sd = Inkey()
   End If
   Locate 4 , 1
   Lcd "       "
   Locate 4 , 1
   Select Case Sd
   Case 0
   Lcd Getadc(0)
   Waitms 20
```

```
    Case 1
    Lcd Getadc(1)
    Waitms 20
    Case 2
    Lcd Getadc(2)
    Waitms 20
    Case 3
    Lcd Getadc(3)
    Waitms 20
    Case 4
    Lcd Getadc(4)
    Waitms 20
    Case 5
    Lcd Getadc(5)
    Waitms 20
    Case 6
    Lcd Getadc(6)
    Waitms 20
    Case 7
    Lcd Getadc(7)
    Waitms 20
    End Select
    Waitms 180
Loop

Automat:

Return

Kalibracec:
While Sd < 78
Inputbin Sd
If Sd = 77 Then
Gosub Prectiadc
Print Cidlopredek1
Print Cidlopredek2
Print Cidlozadek
Print Cidlolevo
Print Cidlopravo
Print Cidlocara1
Print Cidlocara2
End If
Select Case Sd
Case 63
Inputbin C
Mcidlopredek1 = C
Case 64
Inputbin C
Kcidlopredek1 = C
Case 65
Inputbin C
Mcidlopredek2 = C
Case 66
Inputbin C
Kcidlopredek2 = C
Case 67
Inputbin C
Mcidlozadek = C
Case 68
Inputbin C
```

```
Kcidlozadek = C
Case 69
Inputbin C
Mcidlocara1 = C
Case 70
Inputbin C
Kcidlocara1 = C
Case 71
Inputbin C
Mcidlocara2 = C
Case 72
Inputbin C
Kcidlocara2 = C
Case 73
Inputbin C
Mcidlolevo = C
Case 74
Inputbin C
Kcidlolevo = C
Case 75
Inputbin C
Mcidlopravo = C
Case 76
Inputbin C
Kcidlopravo = C
Case Else
End Select
Wend
Printbin 255
```