TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky, informatiky a mezioborových studií

Studijní program:B2612-Elektrotechnika a informatika

Studijní obor:2612R011/90-Elektronické informační a řídící systémy

Robot s inteligencí pro projíždění bludištěm

The robot with the intelligence to pass through a maze

Bakalářská práce

Autor: Karel Höll

Vedoucí práce: ing. Zbyněk Mader Ph.D.

V Liberci 30.4.2011

Prohlášení

Byl jsem seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č121/2000 Sb. právu autorském, zejména § 60 – školní dílo.

Beru na vědomí ,že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu použití , jsem si vědom povinnosti informovat o této skutečnosti TUL , v tomto případě má TUL právo ode mne požadovat úhradu nákladů , které vynaložila na vytvoření díla , až do jejich skutečné výše

Bakalářskou prácí jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím bakalářské práce a konzultantem.

V Liberci 30.4.2011

Podpis

Poděkování

Tímto bych chtěl poděkovat svému konzultantovi ing. Zbyňku Maderovi PhD za mnoho cenných rad při řešení mé práce.Dále bych chtěl poděkovat Doc. Ing. Liboru Tůmovi CSc vedoucímu mého bakalářského semináře za rady k finálnímu provedení mé práce a její prezentaci

Abstrakt

Práce se zabývá vyhodnocením použitých konstrukcí robota z několika otevřených zdrojů a navrhnutí nové konstrukce s použitím řídící desky Arduino založené na čipu Atmega 128. Konstrukce je navržena pro snadné opakování pro použití jako výukový robot k dalšímu programování. Naprogramování řídící desky v prostředí Wiring a jazyce Wiring. Robot je naprogramován pro autonomní jízdu v bludišti a nalezení východu.

Abstract

The work deals with evaluation of the robot designs from several open sources and propose a new design using a control board based on the Arduino chip ATmega 128. The structure is designed for easy retrieval for use as teaching the robot to the next programming. Programming the control board in Wiring environment and Wiring language . The robot is programmed for autonomous driving in a maze and find the exit.

Obsah:

Úvod:

Jako první část celé práce následovalo zhodnocení použití robotických platforem a systémů a to i z hlediska dostupnosti ale i nákladnosti. Celou práci jsem směřoval především v nalezení jiného hardwarového řešení konstrukce a to hlavně z hlediska použití čidel a jejich funkce.Již v počátku práce jsem se musel rozhodnout jestli řešení bude založeno na simulaci bludiště pouhým sledováním čáry nebo plně funkční malý robot s prostorovou orientací.Při hodnocení jsem prostudoval velké množsví řešení z různých zdrojů, především pak s použitím internetu.Velké množství řešení, především univerzitních robotů, jsou ale bohužel mnohapatrové složité konstrukce, které nekorespondují se zadáním aby celý robot byl na platformě jednoho microchipu.Dále jsem vyloučil komerční roboty u kterých by smyslem celé práce bylo pouhé nalezení algoritmu cesty bludištěm.Proto jsem tedy navrhnul celou platformu robota z relativně dostupných komponentů s tím že celá složitost se spíše přibližovala komerčně prodávaným robotům.Při výběru řídící desky pak sehrál největší úlohu fakt, že deska je programovaná v jazyce vycházející ze syntaxe jazyka C. V samotné práci byl kladen důraz na vlastní invenci k řešení problému a k možnosti reprodukovat řešení nikoliv profesionály ale studenty.

1.Zhodnocení dostupných řešení:

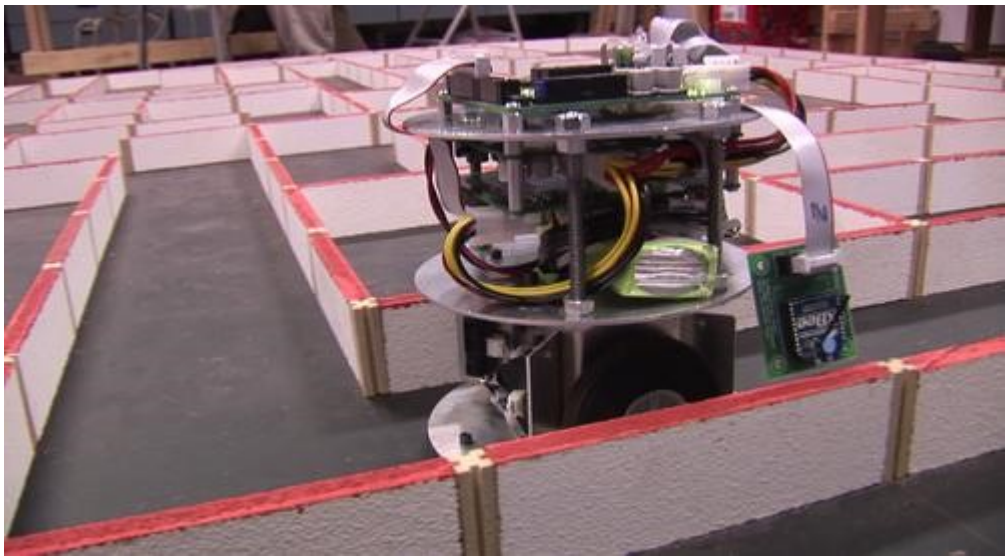
Při návrhu vlastního řešení konstrukce robota bylo nutné nejprve zhodnotit jiné konstrukce.Při hledání jiných konstrukcí a to i univerzitních je optimální cestou vyhledání údajů na internetu. Dále bylo nutné vybrat elementární koncepci robota. Dle dostupných informací se používají dvě koncepce.První koncepcí je simulace bludiště pouhou černou čárou na bílém podkladu kdy řešení robota nenabízí mnoho možností.Všechny řešení jsou víceméně totožné a spočívají v optickém snímání černé čáry několika čidly v řadě těsně nad podložkou např komerčne proádvaný robot Pololu 3pi (*Obrázek 1.*)

Obr.1 *Robot Pololu 3pi*

Jako vhodnější se proto zdála druhá koncepce spočívající v prostorové orientaci robota.I tato koncepce má velké množství řešení a to především z hlediska použitých čidel a celkové složitosti řízeni.Jako příklad poměrně složitého robota je možno ukázat robota ze San Diego Jacobs school of Engineering (*Obrázek 2.*). Zde je vidět už poměrně složitá konstrukce řídící desky , která omezuje reprodukci robota pro výukové účely samotnými studenty. Z těchto důvodů byla zvolena konstrukce založená na desce s jedním mikroprocesorem a samotná konstrukce se navrhovala především na množství a typ čidel. Při výběru robotů založených na koncepci řízení jedním mikroprocesorem je možné použít několik komerčně prodávaných robotů ale i stavebnici např LEGO Mindstorm. Všechny tyto možnosti jsou ale omezeny konstrukcí a použitými čidly.A jako zásadní se jeví např v případě LEGA i způsob programování i když je dnes možné i

LEGO programovat v jazyce C.

Dále bylo nutné vybrat a zhodnotit typ a výrobce senzorů na orientaci v prostoru. V tomto případě je na výběr z několika možností. Při orientaci v bludišti byly předem vyloučeny senzory na orientaci



Obr.2 *Robot San Diego Jacobs school of Engineering*

na bázi GPS, kompasu a orientace pomocí zaměřování lokálníchv ysílačů. Výběr se tedy omezuje na použití ultrazvukových senzorů např SRF05 (*Obrázek 3.*) který je možno použít i na krátké vzdálenosti, ale využití těchto senzorů je spíše ve větších vzdálenostech.

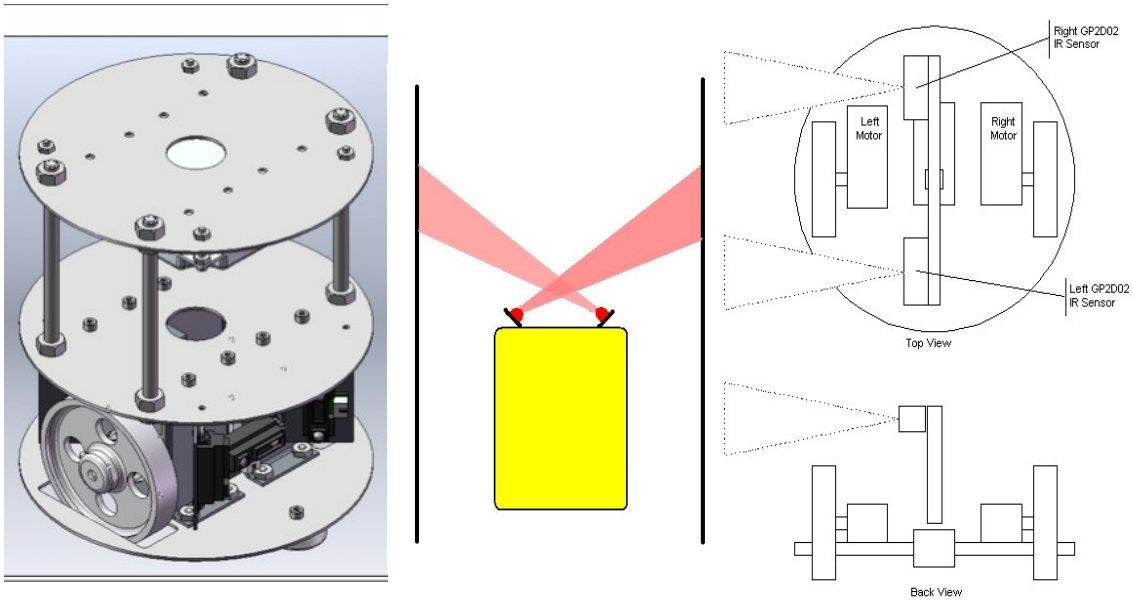
Obr.3 *Ultrazvukový senzor SRF05*

Další z možností pro prostorovou orientaci je použití optických IR senzorů. Z řešení nalezených na internetu velká část robotů používá čidla typu GP2Y0A41 (*Obrázek 4.*) od fy Sharp,kde jako výstup se používá měření napětí na 10 bitovém AD převodníku. Jako nevýhoda těchto senzorů je jejich mírně vyšší cena a nepřesnost měření pod 4 cm, což může být v úzkém bludišti komplikace.

Obr.4 *IR senzor GP2Y0A41 Sharp*

Je možno najít několik robotů se zajímavou konstrukcí založenou na těchto čidlech např konstrukci

J.Wurzbacha a A.Forencicha využívající 6 IR senzorů nebo konstrukci uveřejněnou na webu ieee.ucsd.edu využívající 2 IR senzory popřípadě konstrukce M. Linnena uveřejněná na webu www.parex.com (*Obrázek 5.*)

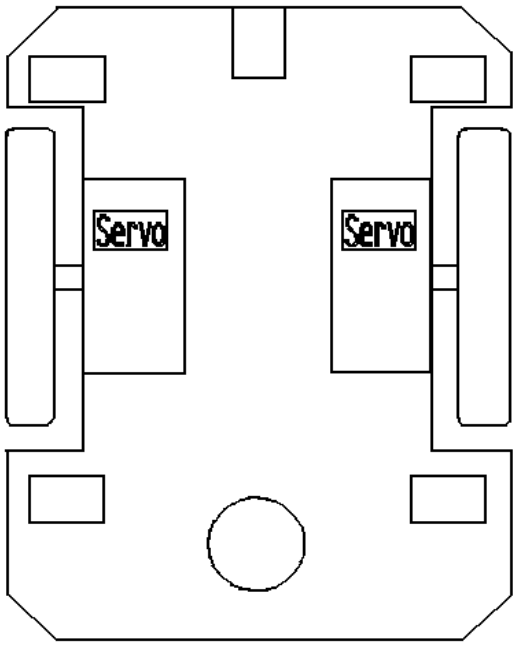


Obr.5 *Použití IR čidel fy Sharp odečítající napětí*

Zvolené řešení této práce zkouší jít trochu jiným směrem a to použití dvou typů IR senzorů a to GPY20D805 a 810 i tyto senzory jsou od firmy Sharp , která vyrábí nejdostupnější senzory pro malou robotiku. Senzor typ 805 spíná při vzdálenosti 5 cm a to s poměrně velkou přesností a senzor 810 na vzdálenost 10 cm.

2.1 Kostra robota:

Pro navržení a zhotovení kostry robota byl vybrán systém s diferenčně řízenými koly s podpěrou všesměrovou kuličkou , které umožnuje nejjednoduší systém řízení a to především s ohledem na otočení na malém prostoru a také oproti pásové platformě menší spotřebu energie.Vzhledem k použití robota pouze v mistnosti na hladkém povrchu je toto řešení plně dostačující.Kostra je zhotovena z PVC , které je mezi deskami z hliníku a je řešena jako dvoupatrová propojená hliníkovými profily. Spodní část nese pohonná serva, samotná čidla a napájení, vrchní deska nese řídící desku s mikropocesorem(*Obrázek 6.*).Celý robot je navržen jako mírně asymetrický pro použití pouze jednoho všesměrového kola a lepší umístění napájecích baterií. Mírnou komplikací při tomto návrhu je ne úplně ideální diferenční zatáčení.



Obr.6 *Nákres spodní desky robota*

2.2 Pohon robota:

Pro pohon robota bylo nutné zvažit několik možností, a to stejnosměrnými motorky nebo modelářskými servy.Protože robot se bude pohybovat pouze na rovině a na hladké podložce nebylo nutné dimenzovat pohon na vysoké zatížení. Při použití stejnosměrných motorků by bylo nutno doplnit robota ještě o H- můstek a proto byl vybrán pohon

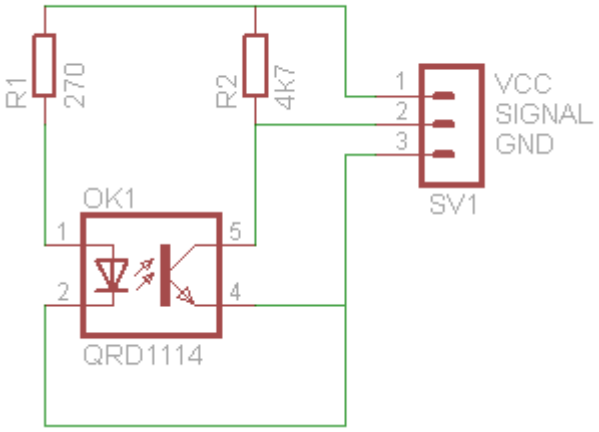
upravenými modelářskými servy od firmy Hitec HS-422. Serva je nutno upravit na kontinuální otáčení mechanickou úpravou převodovky(Odstraněním zarážky)a odpojením zpětné vazby potenciometru.Serva jsou ovládána PWM ,které řídící deska umožnuje(*Obrázek 7.*).

Obr.7 *Servo HS-422*

Serva HS-422 odebírají dle katalogu proud 180mA a dávají moment 3,3kg/cm při napájení 4,8V. Dle katalogu Hitec má servo rychlost 0,21sec/60° při 4,8V jednoduchým výpočtem tedy dostaneme 1,26sec/ot a z tohoto vypočteme 48 ot/min. Při průměru kola d=6,6cm dostaneme rychlost pohybu 993,6 cm/min popř 16,56 cm/sec.
Pro samotný pohon byla použita pogumovaná kola SW 65 které jsou vyrobena k přímé montáži na unašeče serv Hitec.

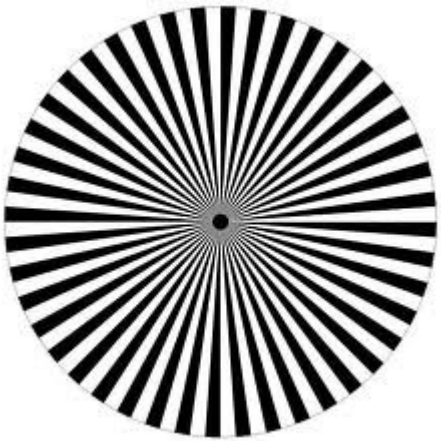
2.3 Incrementální encoder

Vzhledem k nutnosti přesného řízení zatáčení v prostoru bludište bylo nutné zhotovit dva inkrementální encodery. Encoder je vyroben z IR reflexního čidla QRD1114 skládajícího se z IR diody a fototranzistoru.Toto čidlo nahrazuje dříve často používané CNY 70.Jako zapojení bylo použito naprosto jednoduché zapojení s pouhými dvěma rezistory omezujícími procházející proud(*Obrázek 8.*).



Obr.8 *Zapojení čidla QRD1114*

Pro odrazivou plochu byla použita matná bílá samolepící folie s natištěnými 44 černými výseči.Takto jemný rastr je naprostro dostačující pro čítání pulzů a řízení zatáčení na základě těchto pulzů(*Obrázek 9.*)Mírná nevýhoda je dílenské zpracování které u pohonných kole může obsahovat vůli a IR čidlo je nutné mít do vzdálenosti cca 1mm.Pro případné použítí v celkové lokalizaci je zde možnost generovat chyby.Pro ještě přesnější snímání by bylo nutné použít zapouzdřené encoderu kde už je cenový nárůst.



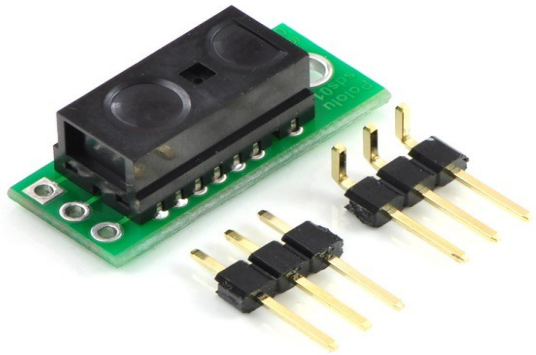
Obr.9 *Rozlišení incrementálního encoderu*

2.4 Napájení

Pro napájení celého robota i včetně řízení byl zvolen jeden napájecí modelářský pack 8,4V 1300mAh sestavený z článků GP NiMh umožňující odběr proudu po dlouhou dobu.

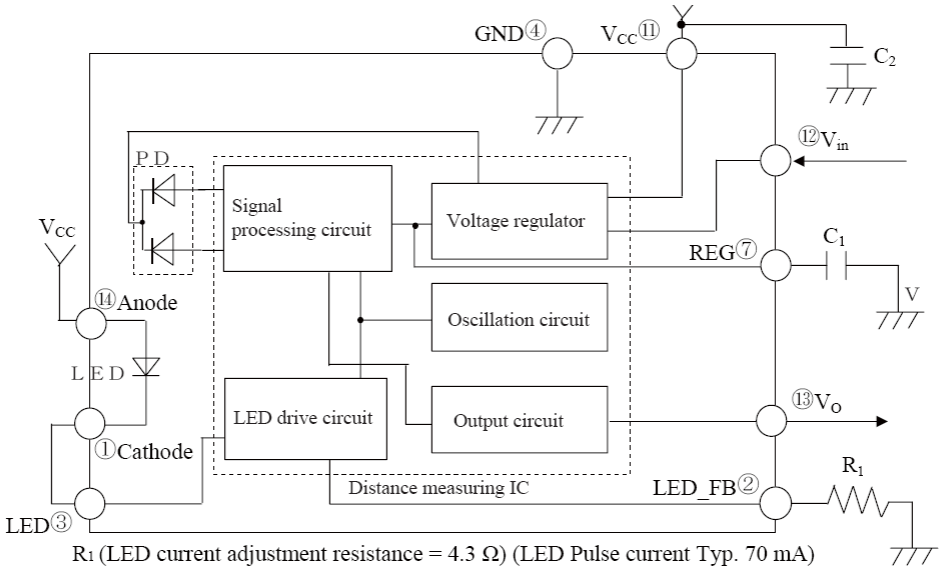
2.5 IR senzory

Pro samotnou orientaci v prostoru a to především na detekci překážek ale i rozhodování byly použity IR senzory fyrmy Sharp a to GPY20D805 a 810(*Obrázek 10.*).Použití těchto senzorů bylo pro celou práci naprosto zásadní jelikož od jejich použitía rozvržení se řídí i celý algoritmus naprogramování hledání východu z bludiště.Pro zjednodušení byla použita i základní deska Pololu s SMD odpory a kondenzátory a zároveň sloužící také k mechanickému ukotvení IR senzoru na robota.



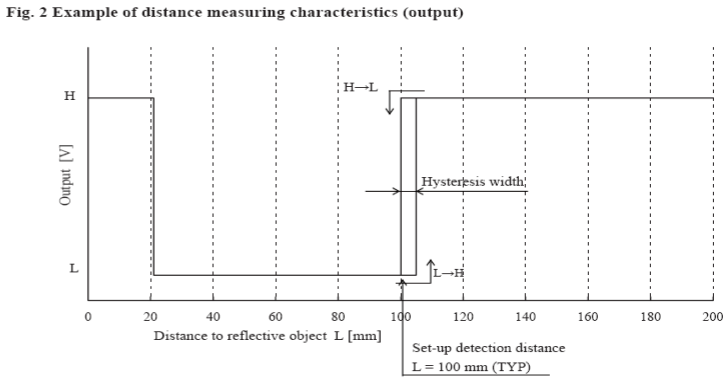
Obr.10 *IR senzor Sharp GPY20D805 již namontovaní na desce příslušenství*

Ze zapojení je patrné že senzor již obsahuje mikroprocesor generující pulzy o dané frekvenci na IR diodu a zároveň vyhodnocují na fotodiodě odraz dle vzdálenosti. (*Obrázek 11.*).Jako velkou výhodu je možné považovat při použití tohoto senzoru fakt, že mikroprocesor základní řídící desky není zatěžován výpočtem pulzů při použití samotných IR diod což by pro 5 senzorů znamenalo nezanedbatelnou komplikaci už vzhledem k výpočetnímu času a jako další výhoda je přímý digitalní výstup kdy výstup **LOW** nám dává signál o překážce v dané vzdálenosti.Tento stav je již snadněji interpretován řídícím mikroprocesorem kdy nám postačuje pouhých 5 digitálních vstupů.A



Obr.11 *Schema zapojení senzoru GPY20D805*

A jako poslední důležitou věc je možné u těchto senzorů považovat hysterezi při překlápění senzoru.(*Obrázek 12.*)Samotné použití těchto senzorů navíc nezatěžuje proudovým odběrem 10mA.



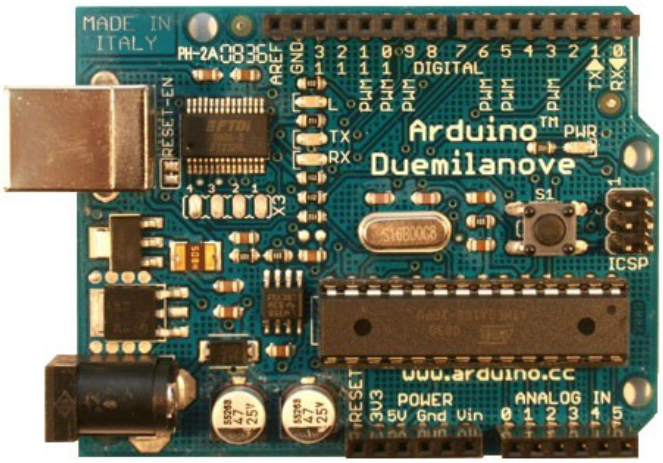
Obr.12 *Hystereze senzoru GPY20D81O*

řídící desku. Z návrhu základní desky(*Obrázek 6.*) je patrné rozmístění jednotlivých senzorů.V přední polovině robota byly použity senzoty GPY20D805 pro překlápění na vzdálenosti odrazu 5 cm.Tyto senzory byly použity jako kolizní čidla a to čelní pro lokalizaci přímé překážky a boční pro korekci směru v bludišti, kde pro tento účel se

zdálo použití těchto senzorů vchodnější než mechanické spínače kolize.Zadní senzory GPY20D810 překlápějící na vzdálenosti odrazu 10 cm jsou použity jako logické pro samotné rozhodování cesty v bludišti.Celá tato koncepce vzhledem k hledání cesty v náhodném bludišti bez přesné lokalizace cíle byla zvolena jako optimální, umožnující poměrně blízké sledování překážky ve vzdálenosti mezi 5-10 cm.

2.6 Řídící jednotka

V souladu se specifikací bakalářské práce a to naprogramovat celé ovládání robota pomocí mikroprocesoru bylo nutné provést výběr daného typu.Vzhledem k možné replikovatelnosti jako výukového robota studenty bylo nutné přihlédnout při výběru ke složitosti zapojení a také k výběru programovacího jazyka.Vzhledem k možnosti sestavovat tohoto robota i bez nutnosti zhotovování desek plošných spojů bylo rozhodnuto použít nekterou s výukových desek obsahující přímo mikroprocesor. Několik možností bylo s procesory Picaxe ale programování je zde realizováno pomocí jazyka Basic popř vývojovými diagramy.Další možností bylo použití desky pro procesor BasicAtom což je také interpreter jazyka Basic. Deska Orangutan LV-168 zde byla jako nepříznivá vyhodnocena cena. Desky obtížněji zajistitelné nebyly vůbec uvažovány. Jako nejpříznivější byla vybrána deska Arduino Duemilanove a to z důvodů velkého množství pinů a programovaní v jazyce syntaxe C které je pro studenty nejvhodnější.(*Obrázek 13.*)

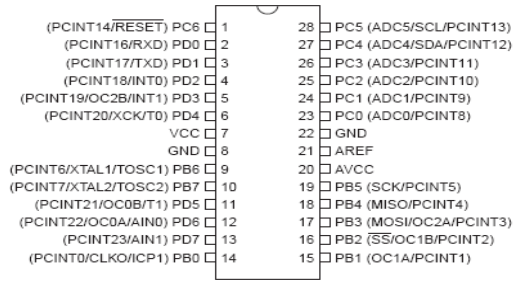


Obr.13 *Deska Arduino Duemilanove*

Deska je osazena mikroprocesorem Atmel Atmega168. Vývojová deska obsahuje 14 I/O pinů z kterých je 6 možno použít na generování PWM. 6 analogových vstupů

obsahujíci 10 bitový AD převodník. Celá deska pracuje na taktu 16MHz. Procesor obsahuje 16kB paměti Flash (z toho 2kB zabírá nainstalovaný bootloader). Dále mikroprocesor obsahuje 1kB SRAM paměti a je možno použít ještě 512 bytů EEPROM.ále je možno použít dva I/O piny pro seriovou linku jako Tx/Rx. Dále je možno využít externí přerušení na I/O pinech 2 a 3. Tato možnost byla využita pro zajištění bezproblémového řízení zatáčení pomocí encoderů.Dále deska obsahuje stabilizátor napětí na 5V. Z možností které nebyly využity je ještě možno zmínit komunikaci po I2C a SPI popřípadě možnost připojit vlastního referenčního napětí pro AD převodníky.Celkově ale deska obsahuje minimum přídavných obvodů a z hlediska rozměrů je pro použití v malých autonomních robotech vhodná.

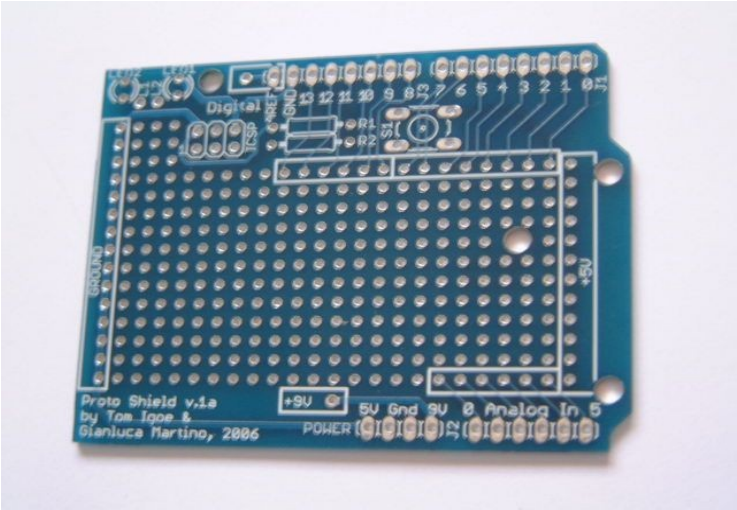
Celá deska je programována jako ISP přes USB rozhraní.Součástí desky je chip FT 232 pro převod úrovní.(*Obrázek 14.*)



Obr.14 *Atmega 168 zapojení pinů*

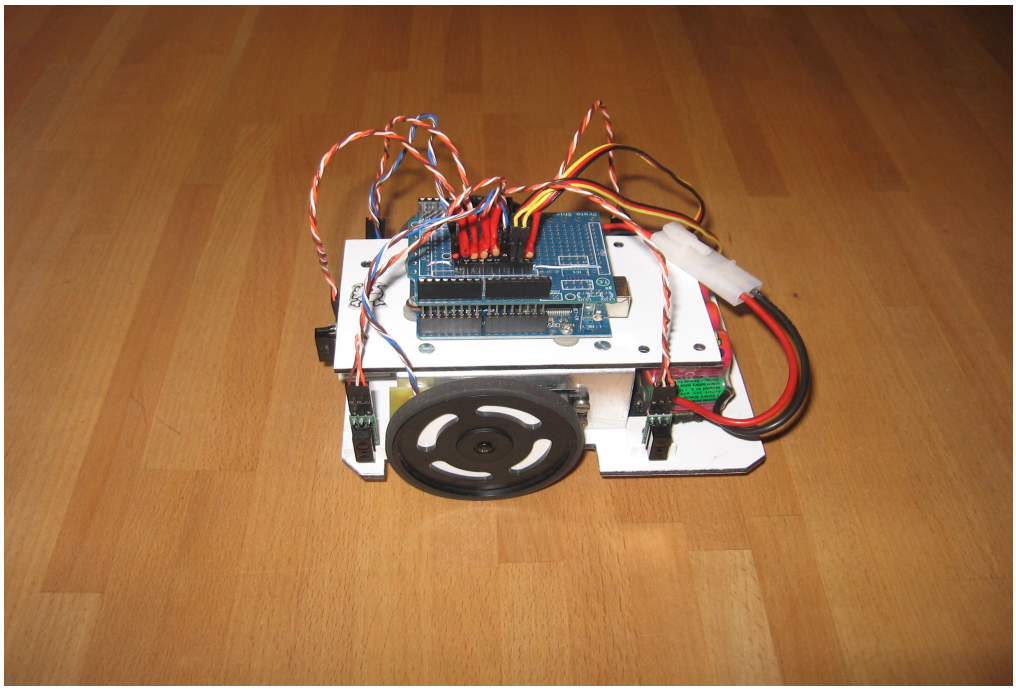
Celá platforma Arduino vznikla jako open source elektrotechnická platforma právě pro snadné používání hardware a software a je možno použít kompletní desku popřípadě nekterý z jejich mnoha klonů nebo si vyrobit vlastni dle Eagle files které jsou volně k dispozici nahrát si bootloader programovacího jazyka do vlastních mikroprocesorů.Celé schema zapojení Arduina viz příloha.

Jako rozvod napájení a GND byla použita originální deska ProtoShield sloužící pouze pro rozvedení signálů napájení a GND pro 3pinových kabelech.Tento Shield je pak použit s propojkami pinů a rozvod napájení a GND je proveden drátem(*Obrázek 15.*)



Obr.15 *ProtoShield*

Hotovou konstrukci robota je možno vidět na připojeném obrázku(*Obrázek 16.*). Zde je vidět rozložení a propojení všech senzorů i mechanickou konstrukci řídící desky.



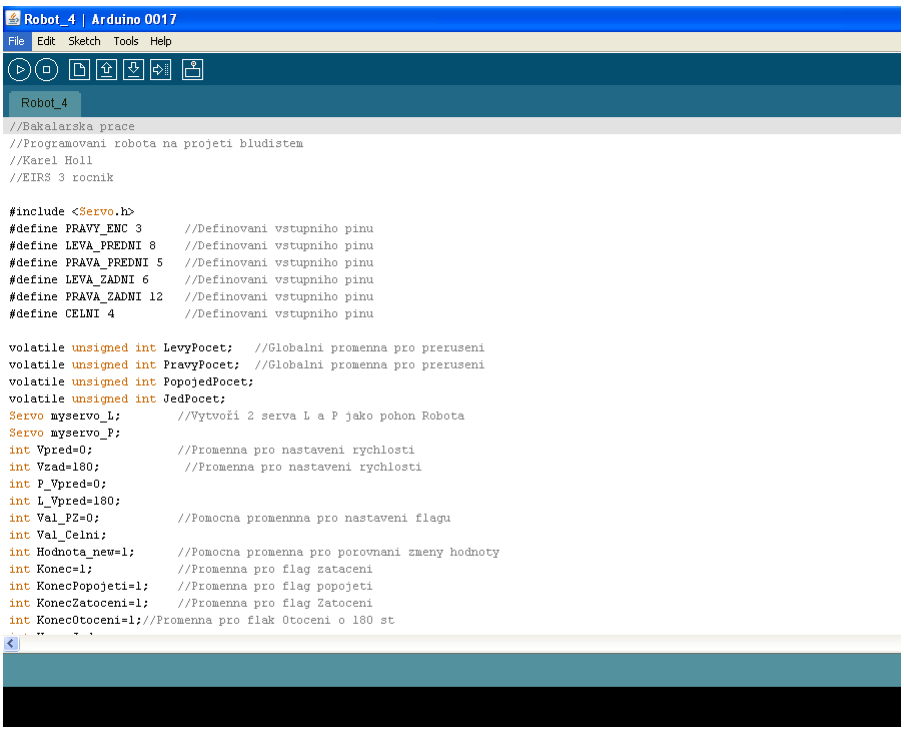
Obr.16 *Konstrukce celého robota*

3.Softwarová část

Programování Arduina je v jazyce Wiring. Tento jazyk je open source jazyk syntaxe jazyka C . Platforma Wiring byla vytvořena Hernando Barragánem na Univerzitě de Los Andes v Kolumbii. V současnosti je již platforma vyvýjena mezinárodně a je široce podporována na internetu.

3.1 Vývojové prostředí

Platforma Wiring obsahuje i vývojové prostředí pro samotné psaní programů a nahrávaní do desky Arduino.Vývojové prostředí je jednoduché ale obsahuje například Serial monitor pro komunikaci po USB lince. Výhodné je pro zobrazování výstupů.Tento monitor je nutné využívat pro odladění programů a to především pro zobrazování jednotlivých proměnných.(*Obrázek 17.*)



Obr.17 *Vývojové prostředí jazyka Wiring*

Verify/Compile
Pomocí tohoto tlačítka kompilujeme vytvářený program.Slouží jako kontrola syntaxe

Stop
Stop seriové komunikace

New
Vytváří nový 'Sketch' neboli program.

Open
Otevírání uložených programů.

Save
Ukládání programů

Upload to I/O Board
Nahrání programu do zvoleného boardu

Serial Monitor
Otevírá nové okno pro seriovou komunikaci s Arduinem.

3.2 Konstrukce programu

Celý blok programu se dělí do několika sekcí .Konstrukce programu vychází z jazyka C a proto i má i podobnou konstrukci. Hned v úvodu je možno pomocí příkazu **#include** zadat použití knihovny.V této práci je pro zjednodušení použita knihovny pro ovládání serva.První část slouží jako deklarační pro deklarování konstant, proměnných. U proměných je možno stejně jako u klasického jazyka C rovnou přiřazovat hodnotu. Každý příkaz musí být ukončený středníkem **';'** .

```
#include <Servo.h>
#define PRAVY_ENC 3      //Definovani vstupniho pinu
#define LEVA_PREDNI 8    //Definovani vstupniho pinu
#define PRAVA_PREDNI 5   //Definovani vstupniho pinu
#define LEVA_ZADNI 6     //Definovani vstupniho pinu
#define PRAVA_ZADNI 12   //Definovani vstupniho pinu
#define CELNI 4          //Definovani vstupniho pinu
volatile unsigned int LevyPocet;   //Globalni promenna pro preruseni
```

```
volatile unsigned int PravyPocet;  //Globalni promenna pro preruseni
volatile unsigned int PopojedPocet;
volatile unsigned int JedPocet;
Servo myservo_L;        //Vytvoří 2 serva L a P jako pohon Robota
Servo myservo_P;
```

Na ukázce kodu je naprosto jasné použití knihovny a definování konstatnt, které je výhodné právě pro naprogramování pinů.Je zde také vidět definování proměnných.Samotný jazyk Wiring má velké množství knihoven které můžeme vložit do programu např pro komunikaci po seriové lince, pro přístup EEPROM,Ethernet apod.

Použitá knihovna **<servo.h>** řeší pomocí několika příkazů ovládání serva bez nutnosti počítání PWM pomocí časovačů mikroprocesoru .

```
myservo_L.attach(10);
myservo_P.attach(11);
```

Pomocí těchto příkazů inicializujeme jednotlivá piny jako výstup pro servo.

```
myservo_P.write(P_Vpred);
myservo_L.write(L_Vpred);
```

Pomocí těchto příkazů zadáváme hodnoty pro jednotlivá serva zde v rozsahu 0-180° pro směr a rychlost otáčení

```
myservo_L.detach();
myservo_P.detach();
```

Pomocí těchto příkazů jednotlivá serva vypínáme a to především zdůvodů, že i při nastavené nulové rychlosti serva pocukávají.

V další části programu pomocí příkazu **void setup() se** nastavují vlastnosti jednotlivých I/O pinů zda budou sloužit jako vstupní nebo výstupní a zároveň se nastavují další hodnoty pro mikroprocesor jako obsluha vnějšího přerušení, která je zde použita pro čítání inkrementálního encoderu a dále se zde nastavují hodnoty pro komunikaci se Serial minitorem .V tomto bloku je možno také naprogramovat některé události které

mají být provedeny pouze jednou při inicializaci.

```
void setup()
{
  Serial.begin(9600);
  attachInterrupt(1, PravyEncoderEvent,FALLING);     //Nastavi vnejsi preruseni, falling
od 1 do 0, a obsluhu preruseni
  pinMode(PRAVY_ENC,INPUT);    //Nastavi pin cidla na vstup
  pinMode(CELNI,INPUT);        //Nastavi pin cidla na vstup
  pinMode(LEVA_PREDNI,INPUT);  //Nastavi pin cidla na vstup
  pinMode(PRAVA_PREDNI,INPUT);  //Nastavi pin cidla na vstup
  pinMode(LEVA_ZADNI,INPUT);      //Nastavi pin cidla na vstup
  pinMode(PRAVA_ZADNI,INPUT);   //Nastavi pin cidla na vstup
}
```

Celý hlavní blok potom pracuje ve smyčce dané příkazem **void loop()** Tato smyčka pracuje jako trvalá a vykonává celý hlavní program.Samotný jazyk Wiring obsahuje klasické vyhodnocovací konstrukce jako **if/else, for, while.** Pro naprogramování hlavního programu byla použita především základní konstrukce **if/else.** Celá tato konstrukce byla použita pro porovní vstupů jednotlivých kolizních a vyhodnocovacích čidel.

```
void loop()
{
    if (digitalRead(PRAVA_ZADNI)==LOW)       //Testuje prave zadni cidlo pro pravidlo
prave ruky
        {
            if(digitalRead(CELNI)==HIGH)            //Testuje celni kolizni cidlo pokud neni
prekazka
                {
                    if(digitalRead(PRAVA_PREDNI)==LOW)   //Testuje prave predni kolizni
cidlo na upravu rychlosti
                    {
                    P_Vpred=85;                    //Snizuje rychlost leveho kola ke korekci smeru
                    }
```

```
                else
                {
                    P_Vpred=0;                    // pokud je cidlo HIGH rychlost se vraci na
puvodni
                }
               if(digitalRead(LEVA_PREDNI)==LOW)   //Testuje leve predni kolizni cidlo
na upravu rychlosti
                {
                     L_Vpred=100;                 //Snizuje rychlost praveho kola na upravu
rychlosti
                }
                else
                {
                  L_Vpred=180;
                }
                myservo_L.attach(10);
                myservo_P.attach(11);
                myservo_P.write(P_Vpred);
                  myservo_L.write(L_Vpred);             //Pokud je prekazka i vlevo- zatoci
zpet o 180 st
            }
           else                //Pokud je na celnim koliznim cidle prekazka vetvime program
dale
            {
                  if(digitalRead(LEVA_ZADNI)==HIGH)        //Testuje leve zadni cidlo a
pokud  je HIGH robot pouze zatoci doleva
                {
                PravyPocet=0;                     //Vynuluje counter
                ZatocDoleva();                    //Zavola podprogram na zatoceni doleva
                }
                else                          //Pokud je cidlo LOW robot zatoci o 180 st
                {
```

```
            LevyPocet=0;                    //Vynuluje counter
            ZatocZpet();                    //Zatoci o 180 st
            }
        }
    }
    else   //Pokud je  prave zadni cidlo HIGH a tedy volno zatoci robot a popojede o dany
pocet tiku
 }
```

Zde je ukázáno testování jednotlivých vstupů pomocí příkazu **if/else .** Po načtení hodnoty pomocí příkazu **digitalRead()** ze vstupních pinů porovnáváme hodnoty zda dosahují **HIGH** nebo **LOW** a detekují překážku před sebou popřípadě na straně ve vzdálenosti 5 cm.V případě detekce překážky je proměnná na hodnotě **LOW.**

Pro rozhodování směru jízdy jsou použity vstupní piny které načítají pomocí příkazu **digitalRead()** ze zadních čidel, které indikují překážku ve vzdálenosti 10 cm. Na základě vyhodnocení nejdříve pravého zadního čidla zatáčí robot doprava a využíváme tak algoritmus pravé ruky.V případě že je pravé zadní čidlo **HIGH.** Pokud je pravé zadní čidlo **LOW** sleduje robot překážku a pomocí předních čidel koriguje směr jizdy.V případě že pravé přední popřípadě levé přední čidlo načte mikroporocesor v hodnotě **LOW** upravují se pomocí příkazu **myservo_L.write()** popř **myservo_P.write()** jednotlivé rychlosti.Zde je nutno uvést, že při použití upravených serv je korekce slabá pouze v malém rozsahu

.Pro vyhodnocování je možno použít i klasickou konstrukci **switch/case** pro celé vektory , kdy načtení všech vstupních pinů je uložen do vektorů,ale ta zde nebyla použita.

Jako nezbytné pro celý program je používání podprogramů pro opakující se příkazy.Zde především pro zatáčení a popojíždění kdy podprogramy využívají vnější přerušení.

Použití fukncí je možno použít se zadáním návratové hodnoty nebo bez ní.Zde byl použit příkaz bez návratové hodnoty sloužící pro pouhé vykonání příkazu.

```
void ZatocDoleva()
//Podprogram na zatoceni doleva o 90stupnu
 {
 for(int i=0;i<17;)                    //Pocet tiku na otoceni
   {
    i=PravyPocet;                  //Prirazeni hodnoty
    myservo_L.attach(10);
    myservo_P.attach(11);
    myservo_P.write(Vzad);
    myservo_L.write(Vzad);
   }
   myservo_L.detach();
   myservo_P.detach();
   PravyPocet=0;               // Vynuluje counter
 }
```

Zde je vidět část kodu podprogramu, který se vykonává bez předávání parametrů a bez návratové hodnoty.Tento kod ale využívá přerušení od inkrementálního encoderu pro čítání jednotlivých tiků a nikoliv softwarovou inkrementaci. Tímto způsobem lze výhodně dosáhnout přesných zatočení podle počtu tiků v převodu na jednotlivé stupně encoderu.Samozřejmě ale tato funkčnost je zaručena na pevné podložce kde nedochází ke smyku a tím pádem zkreslení.V podprogramu bylo nutno použít i nastavování vlastních uživatelských flagů pro nastavení ukončení zatáčení.

```
void PravyEncoderEvent()      // Vyhodnocuje preruseni od praveho encoderu
{
 if (digitalRead(PRAVY_ENC) == LOW)
  {
   PravyPocet++;    //Incrementuje counter
   LevyPocet++;     //Incrementuje druhy pomocny counter
```

```
    PopojedPocet++;
    JedPocet++;
  }
}
```

Zde je vidět podprogram pro obsluhu vnějšího přerušení který nastavuje jednotlivé proměnné pomocí kterých inkrementujeme podprogramy pro zatáčení a popojíždění. Pro požívání vnějšího přerušení jsou u desky Arduino vyhraženy digitální piny 2 a 3. Obsluha přerušení se nastavuje v inicializační části programu pomocí příkazu **attachInterrupt(přerušení,funkce,mód).** Zde určíme pomocí kterého podprogramu bude každé ze dvou přerušení ošetřeno a pomocí módu si můžeme vybrat zda přerušení bude reagovat.Mód **low** určuje, že mikroprocesor bude reagovat pokud vstupní pin bude **LOW.** Pro mód **change** čítá mikroprocesor při jakékoliv změně. A pro módy **rising** popř **falling** dochází k čítání při změně z **LOW** na **HIGH** a naopak.

Jako další zde neuvyužitá možnost je pomocí podprogramu pro ošetření vnějšího přerušení možnost počítat celkovou ujetou vzdálenost .

Závěr:

Podle mého názoru byl cíl této práce splněn. Byla vytvořena nová funkční hardwarová koncepce. Pomocí jazyka Wiring byl celý robot naprogramován. Celá koncepce se po testech ukázala jako plně funkční, i když nebylo možno plně porovnat tohoto robota s ostatními platformami, což je možno pouze na závodech.Zde by bylo možno porovnat úspěšnost i v porovnání s ostaními roboty.Domnívám se, že tato platforma umožnuje použití jako výukového robota s možností dalšího rozšíření např o Bluetooth moduly pro komunikaci a možností napž mapování terénu. S ohledem na jednoduchost procesoru a čidel je jeho replikovatelnost snadná. Mírným problémem při vývoji složitějších algoritmů se může ukázat vývojové prostředí, které neumožnuję složitější ladění.

Použitá deska Arduino je velmi široce rozšířená především pro výukové účely a ve spojení s jazykem vycházejícím ze syntaxe C je možno uvažovat i nad alším použití ve výuce.

Seznam použité literatury:

[1] NOVÁK Petr. ***Mobilní roboty:Pohony,senzory,řízení*** *BEN-technická literatura Praha 2005*

URL:

www.serva.cz: Jiří Ježerský-úprava serva HS-422 na 360 stupňové otáčení

www.arduino.cc: Reference

www.societyofrobots.com/member_tutorials/node/94

ucsdnews.ucsd.edu

www.fairchildsemi.com-zapojení čidla QRD1114

www.sharpsma.com/optoelectronics- optická čidla vzdáleností

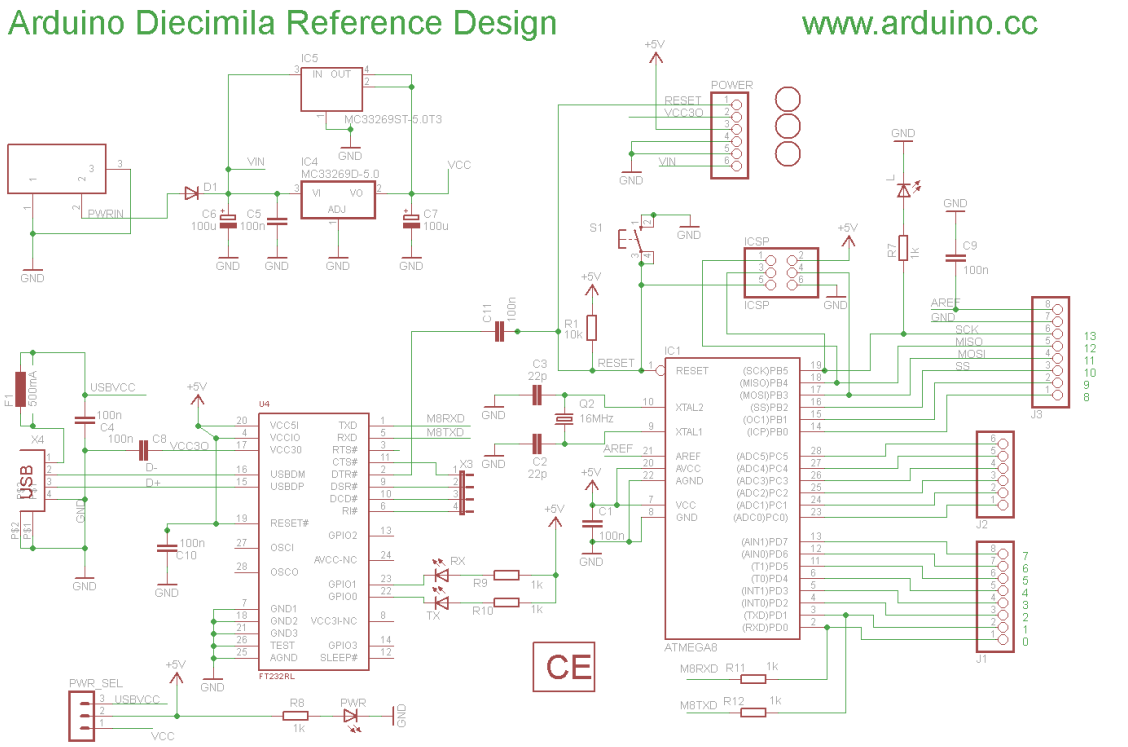
www.atmel.com

www.hitec.com

www.parex.org

psurobotics.org/wiki – použití encoderu

Schema desky Arduino:



Fotodokumentace: