Středoškolská technika 2015

Setkání a prezentace prací středoškolských studentů na ČVUT

S-Bot – robot pro sledování barevné čáry

Jiří Šrámek, Karel Trnečka

SPŠ el.it Dobruška
Čs.odboje 670, Dobruška

STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor č. 10 - Elektrotechnika, elektronika a telekomunikace

S-Bot – robot pro sledování barevné čáry

S-Bot - The Coloured line following robot

Autoři: Jiří Šrámek, Karel Trnečka
Škola: Střední škola elektrotechniky a informačních technologií, Dobruška
Konzultant: Ing. Jiří Vintera

Dobruška, 2015

Poděkování

Rádi bychom poděkovali panu Miroslavu Máchovi za cenné rady ohledně programování PIC32, dále panu Ing Richterovi, Ph. D. za rady ohledně PSD regulátoru a panu Ing. Jiřímu Vinterovi za tipy ohledně softwaru a návrhu DPS, bez kterých by tato práce nemohla vzniknout.

Prohlášení

Prohlašuji, že jsem svou práci vypracoval samostatně, použil jsem pouze podklady (literaturu, SW atd.) uvedené v přiloženém seznamu a postup při zpracování a dalším nakládání
s prací je v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) v platném
znění.

V Dobrušce, dne 31. ledna 2015
Podpis: .....

Anotace

Cílem práce bylo sestavení robota vybaveného rozhraním Wi-Fi, díky čemuž je možné dálkové ovládání robota a odesílání telemetrie do PC. Robot je schopen sledování černé čáry na bílém podkladu s možností reagovat na různě zbarvené úseky, např. zrychlením či zpomalením.

Anotation

The aim of this work was to build a robot equipped with a Wi-Fi interface thus enabling to remote control the robot and send telemetric data to PC. The robot can track a black line on a white background with the option to respond to colored sections, such as acceleration or deceleration.

1 Úvod

Robotika je velmi rozsáhlý obor zabývající se konstrukcí robotů, jejich designem, ale i jejich využitím. Využití nachází ve firmách zabývajících se automobilovou výrobou (svářecí technika), v armádě, ale i ve vesmírném inženýrství, tak v běžném životě (robotické vysavače). S robotikou se setkáme také v různých zájmových organizacích, kde se řeší například roboti sledující čáru, přes chodící roboty, po komplikovaná řešení s více mikrokontroléry na různých létajících přístrojích (např. quadcoptery).

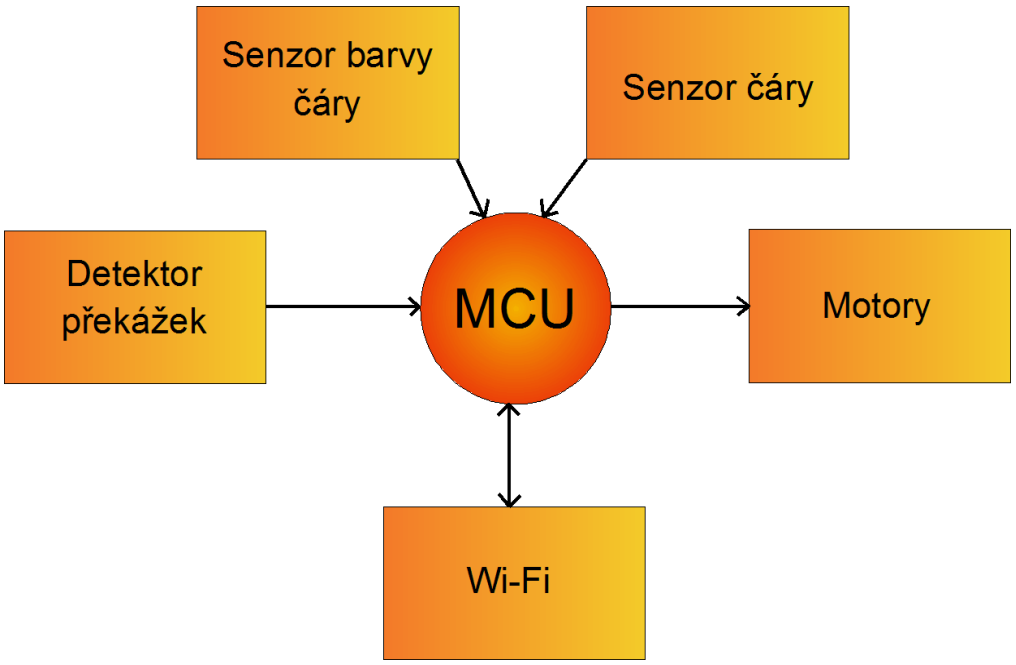
Existuje velké množství soutěží po celém světě zabývajících se soutěžením mezi roboty. Mezi nejznámější soutěže patří např. minisumo, sledování čáry. Mezi pokročilé týmově soutěže patří například robotický fotbal.

Světově nejznámnější soutěž v robotice je VEX Robotics World Championship. Jedna z nejpopulárnějších soutěží, které se konají v našem okolí, je ISTROBOT Bratislava, kde se soutěží v kategoriích linefollower, micromouse nebo ketchup house.

Náš robot může soutěžit v kategorii linefollower, s možností reagovat na různé barvy úseků trati.

2 Teoretický rozbor

Vzhledem k tomu, co požadujeme, aby robot uměl, budou potřeba hardwarové bloky patrné z *obrázku 1*.



Obrázek 1: Potřebné hardwarové bloky

2.1 Způsob pohonu robota

Je mnoho možností pohonu robota, např. diferenciální řízení robota, čtyřkolový robot, případně kráčející robot. Kráčející robot je pro sledování čáry nevhodný, protože i kdyby dosáhl potřebné rychlosti (což je náročné při sestavení v domácích podmínkách), tak by se čára obtížně sledovala vzhledem k složitému uchycení senzorů a houpání robota za chůze. Se čtyřkolovým robotem (typ řízení podobný automobilu) by bylo možné sledovat čáru, dosáhne potřebné rychlosti a s umístěním senzorů by také nebyl problém, avšak vyřešení způsobu zatáčení v prudkých úsecích čáry by bylo nejspíše obtížné. Naproti tomu se diferenciální řízení robota pro sledování čáry nejspíše hodí ze všeho nejvíce, jelikož robot dokáže zatočit prakticky na místě (podobně jako tank) a dosáhne také potřebné rychlosti.

2.2 Snímání pozice čáry

Pro snímání pozice čáry existuje také mnoho postupů, např. použití kamery nebo pomocí IR reflexních optozávor. Pokud bychom použili pro snímání čáry kameru, byl by pak robot univerzální, bylo by možné sledovat i jiné čáry než černé, případně i detekce překážek a jejich barvy, nebo také hledání předmětu v bludišti. Toto řešení by však bylo nákladné a také složité jej implementovat v mikrokontroléru z důvodu potřebných algoritmů pro zpracování obrazu, které by si také vyžádaly velký výpočetní výkon mikrokontroléru. Pokud bychom použili pro snímání pozice čáry IR reflexní optozávory, byly by dvě možnosti jak z nich určovat množství odraženého světla, a to číst digitálně z vhodné kombinace fototranzistoru a rezistoru, zapojených jako dělič napětí, nebo z této kombinace číst analogovou hodnotu napětí pomocí A/D převodníku, ať už externím nebo integrovaným v mikrokontroléru. Pokud bychom využili čtení digitální, bylo by třeba navrhnout rezistor děliče tak, aby senzor snímal s potřebnou citlivostí, tzn. aby při černé čáře byl výstup log. 0 a při bílé log. 1 (nebo obráceně). Oproti tomu čtení A/D převodníkem tento problém částečně eliminuje, jelikož hranice mezi černou a bílou čárou nemusí být mezi log. 1 a log 0, ale kdekoliv, protože v případě 10bitového A/D převodníku při napájecím a referenčním napětí 3,3V je možné určit výstupní napětí ze senzoru po krocích 3,22mV. Mezi klady tohoto způsobu také patří možnost snímání čáry, i když není pod celým senzorem, protože se na jeho výstupu objeví napětí, které je svou velikostí mezi výstupním napětím černé čáry a bílé. Citlivost jednoho senzoru však stále závisí na vhodně zvoleném rezistoru, ale již se nejedná o tak velký problém. Pozici čáry je možné snímat různým počtem senzorů, například i použití dvou senzorů na krajích robota by bylo možné, za předpokladu, že by robot jel stále rovně a až by se na jednom ze senzorů objevila čára, zatočil by. Takový robot by však nesledoval čáru plynule, proto je třeba použít více senzorů, ideální počet bude závislý na šířce robota, šířce čáry a rozestupu mezi senzory.

2.3 Snímání barvy čáry

Barvu čáry je možné snímat dvěma způsoby, analogově nebo digitálně. Analogové snímání barvy čáry by bylo možné pomocí jednoho fotorezistoru, který by byl obklopen barevnými diodami LED (červená, zelená, modrá). Takovýto senzor by však reagoval pomalu a nebyl by moc přesný. Barvu čáry lze také snímat digitálně, pomocí specializovaných integrovaných obvodů, které dokáží rychle převést míru osvětlení fototranzistorů s barevnými filtry např. na frekvenci. Toto dokáží např. Obvody TCS3200D nebo TCS3210 od firmy TAOS.

2.4 Reakce na pozici čáry

Když robot zjistí pozici čáry, měl by na ni nějakým způsobem reagovat. Existují jednoduché způsoby, které však nezaručí plynulou jízdu. Např. máme-li robota se třemi senzory, je možné na principu stavového automatu naprogramovat, když bude čára pod prostředním senzorem - jeď rovně, pod levým - jeď vlevo, pod pravým - jeď vpravo. Jedná se o velmi jednoduchou implementaci sledovače čáry, ve výsledku robot čáru sledovat bude, ale zdaleka ne plynule. Jako lepší se zdá použití PSD regulátoru, který se v programu implementuje jako rovnice, jejímž výsledkem je plnění (PWM) pro levý a pravý motor. Plynulé sledování čáry je však v tomto případě podmíněno správným nastavením proporcionální, sumační a diferenční konstanty. Těmi se v rovnici regulátoru násobí proporcionální, sumační a diferenční složka. Výsledná rovnice regulátoru v obecné podobě poté vypadá takto:

$$PID = k_p P + k_i I + k_d D \qquad (1)$$

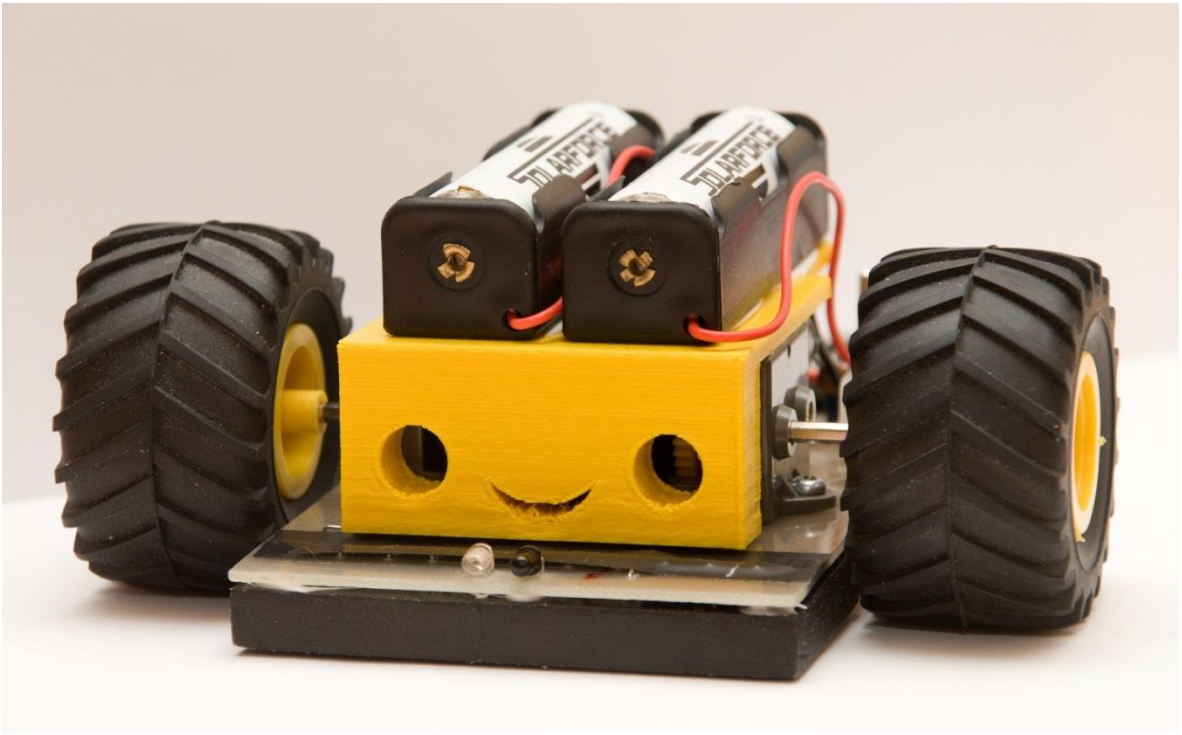
Tyto konstanty je možné nastavit zkusmo nebo pomocí nějaké metody, např. Ziegler-Nicholsovy. Tato metoda spočívá v nastavení konstant k_d a k_i na nulu a následném zvětšování konstanty k_p, regulátor nedosáhne konečného zisku k_u, kdy regulátor začne kmitat s konstantní periodou T_u. Z hodnot T_u a k_u lze poté vypočítat konstanty k_p, k_i a k_d podle vzorců:

$$k_p = 0{,}6k_u$$
$$k_i = \frac{1{,}2k_p}{T_u}$$
$$k_d = \frac{k_p T_u}{8} \qquad (2)$$

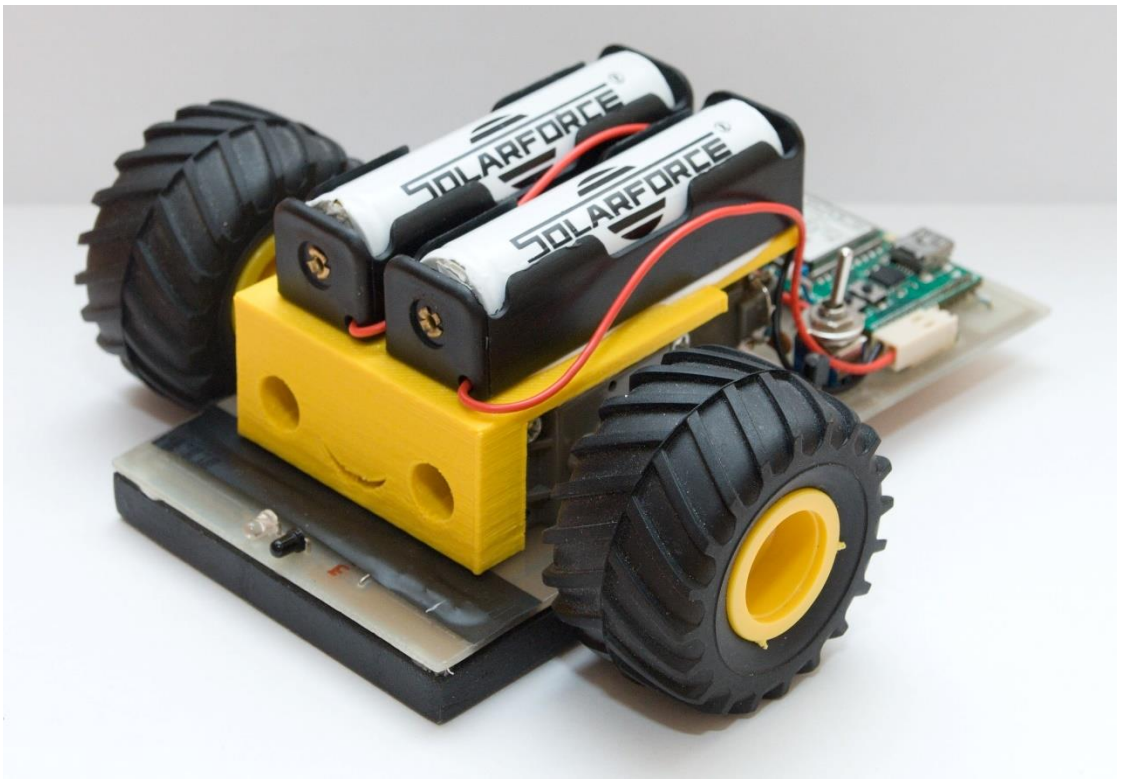
3 Mechanická konstrukce robota

Jako základ robota jsme použili desku plošných spojů, na které je našroubována převodovka TM70168 od firmy TAMIYA s dvěma 3 voltovými motory. Pro přenos točivé síly na povrch jsou použita kola TM70096 o průměru 50mm a šířce 30mm. Převodovka je sestavena pro převod 114.7:1, při kterém se kola točí rychlostí 115ot/min, čemuž odpovídá maximální rychlost robota asi 30cm/s. Vzadu na spodní straně robota je umístěna kulička jako třetí opěrný bod. Na senzorech čáry a senzoru barvy je upevněn kryt, který slouží k odstínění okolního světla. Ten je vyhotoven z plastu o tloušťce 8mm, do kterého jsou vyvrtány otvory pro senzory a drážka. Na

motorech jsou uchyceny dva držáky pro Li-Ion akumulátory 18650. Celkové rozměry robota jsou 145x140x60mm. Robot je na *obrázcích 2* a *3*.



Obrázek 2: Pohled na robota zepředu

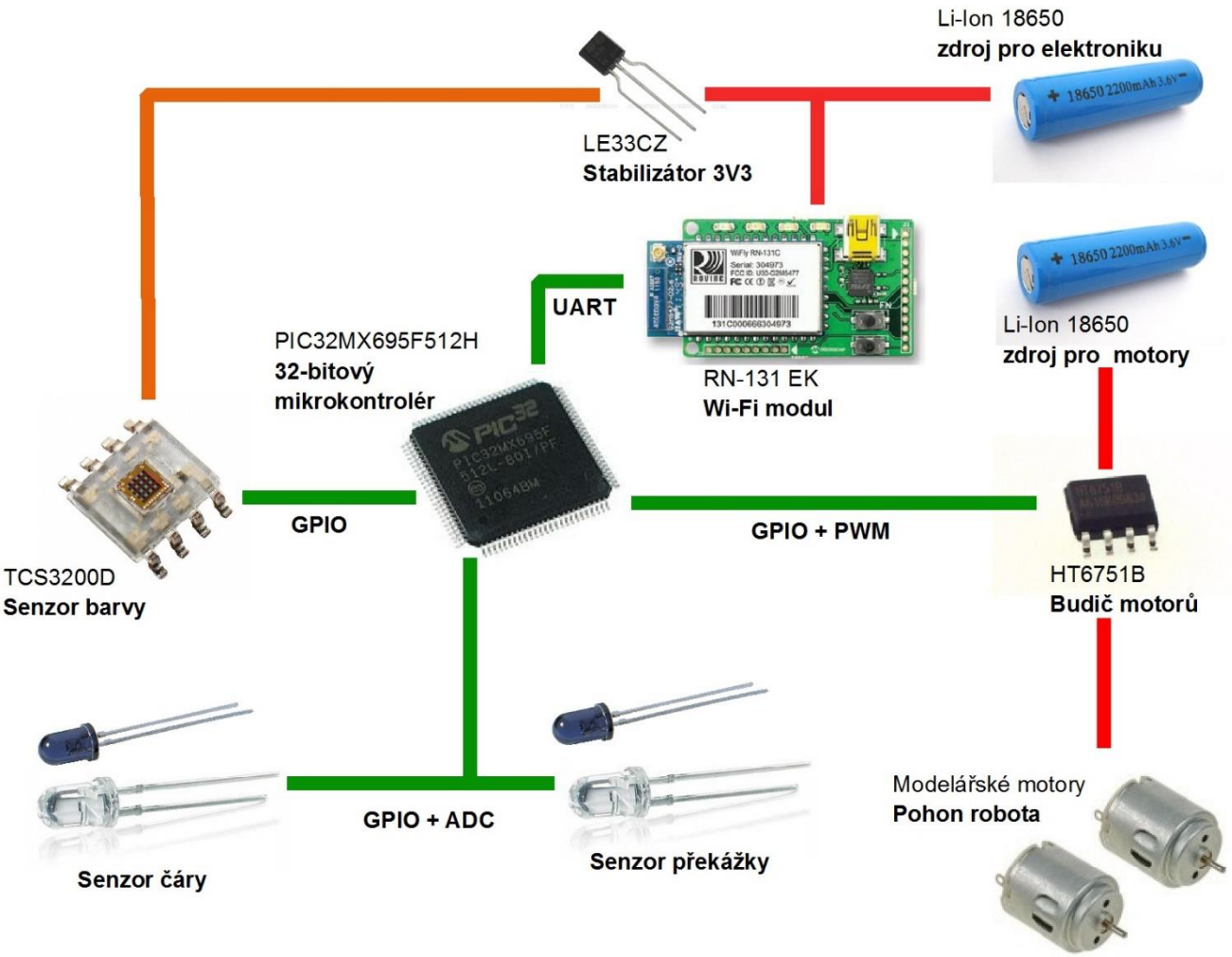


Obrázek 3: Pohled na robota z vrchu

4 Elektronika robota

4.1 Blokové schéma

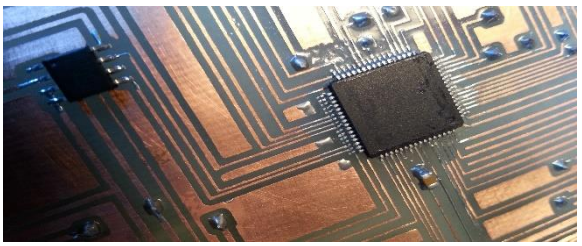
Výsledný návrh hardwaru robota je na *obrázku 4*.



Obrázek 4: Blokové schéma robota

Srdcem zapojení bude mikrokontrolér PIC32, k němu budou pomocí GPIO a ADC připojeny senzory čáry a překážek. GPIO se také využijí pro komunikaci se senzorem barvy. Wi-Fi modul se připojí na sběrnici UART. Motory se budou ovládat pomocí GPIO a PWM. O napájení celého robota se budou starat dva Li-Ion akumulátory a stejný počet stabilizátorů 3,3V. Jeden pro Wi-Fi modul z důvodu velkého odběru (je již na desce modulu) a druhý pro zbytek elektroniky. Motory se budou z důvodu rušení napájet z druhého Li-Ion akumulátoru.

4.2 Deska plošných spojů



Obrázek 5: Připájený PIC32

Deska je navržena jako jednostranná, pro montáž součástek technologií SMT. Všechny součástky jsou použity velikosti 0805, aby bylo možné desku osadit v domácích podmínkách. Integrované obvody jsou kromě mikrokontroléru, který je v pouzdře TQFP64, v pouzdrech SOIC. V desce jsou vyvrtány díry pro uchycení převodovky (3,5mm) a Wi-Fi modulu (2mm).

Návrhy schématu i desky byly vytvořeny v programu EAGLE. Deska byla vyrobena v domácích podmínkách a obsahuje množství propojek na druhé straně. Proto by bylo vhodné použít desku oboustrannou s prokovy a nechat si ji vyrobit u specializované firmy, toho jsme však nevyužili z důvodu vysoké ceny takto vyrobené DPS. Deska je na straně spojů chráněna izolačním lakem určeným na osazené DPS.

Motiv desky, schéma zapojení 3V3 části a výkonové části je z důvodu velikosti a čitelnosti v příloze.

4.3 Důležité komponenty na desce

Mikrokontrolér

Obrázek 6: PIC32 v pouzdru TQFP64 [10]

V konstrukci je použit 32-bitový mikrokontrolér PIC32MX695F512H [2] od firmy Microchip Technology Inc. [1]. Mikrokontrolér je postaven na Harvardské architektuře, proto má oddělenou paměť dat a programu. Jádro mikrokontroléru je typu MIPS32 M4k. Kompletní blokové schéma mikrokontroléru naleznete v příloze.

Mikrokontrolér byl vybrán zejména kvůli svému výpočetnímu výkonu 105DMIPS při frekvenci 80MHz, který se využije při výpočtu regulační hodnoty PSD regulátoru a tím zajistí rychlou reakci na změnu pozice čáry. Dále obsahuje velkou paměť typu flash pro uložení programu (512kB) a také RAM (128kB). Mezi jeho přednosti také patří rychlý 10-bitový A/D převodník, který je schopen přečíst 1M vzorků za vteřinu. Mikrokontrolér obsahuje standardní periferie jako 16-bitové časovače/čítače, SPI a I2C sběrnice, ale také USB a Ethernet, které se v této aplikaci neuplatňují.

Wi-Fi modul

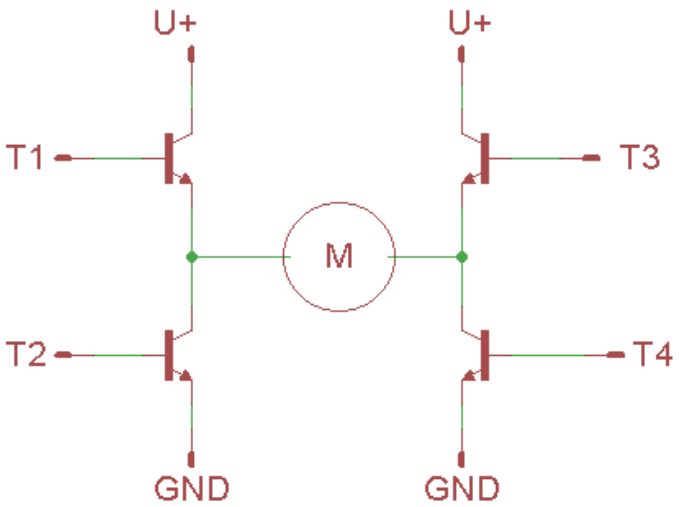
Pro komunikaci s počítačem jsme se rozhodli použít modul Wi-Fi 802.11 b/g/n RN-131 [3] taktéž od firmy Microchip Technology Inc. Tento Wi-Fi modul byl vybrán zejména

kvůli jednoduchému nastavení pomocí ASCII příkazů a komunikaci pomocí rozhraní UART. Tento modul je schopen i samostatné činnosti díky integrovanému procesoru typu ARM, na kterém běží TCP/IP stack a také umožňuje komunikaci s periferiemi připojenými ke GPIO vývodům modulu, případně také čtení analogových hodnot napětí z analogových vstupů modulu. Modul má vlastní stabilizátor napětí *Obrázek 7: Wi-Fi modul [9]*

3,3V, takže je připojen přímo na akumulátor, který je určený pro napájení elektroniky. Použitý Wi-Fi modul je na *obrázku 6*.

Ovládání motorů

Pro spínání DC motorů jsou použity 2 můstky HT6751B [4]. Ty jsme vybrali namísto oblíbeného a rozšířeného L293D z toho důvodu, že fungují spolehlivě a s malými ztrátami už od 3V, a to nejen jejich elektronika, ale i výkonová část. Při pokusech s L293D jsme zjistili, že jeho elektronika funguje spolehlivě také při 3V, ale na výkonové části jsou velké ztráty. Dva můstky HT6751B jsou každý ovládán pomocí 2 GPIO pinů *Obrázek 8: HT6751B [8]* mikrokontroléru pro ovládání směru otáčení a 1 pinu s podporou PWM, která je generovaná modulem OutputCompare/PWM za pomoci 16-bitového časovače TMR2. Tím je zajištěna možnost plynulého nastavení rychlostí motorů v rozsahu hodnot 0 až 12500 při frekvenci PWM asi 800Hz. Schéma základního můstku pro řízení motorů je na *obrázku 8* :



Obrázek 9: Obecný můstek pro řízení motorů

Můstek umožňuje jízdu vpřed i vzad. Při jízdě vpřed se sepnou tranzistory T1 a T4, čímž se na levou svorku motoru přivede napájecí napětí a pravá se uzemní. Motor se točí ve směru hodinových ručiček. Při jízdě vzad se sepnou tranzistory T2 a T3, tím se na pravou svorku motoru přivede napájecí napětí, zatímco se levá svorka uzemní. Motor se točí proti směru hodinových ručiček.

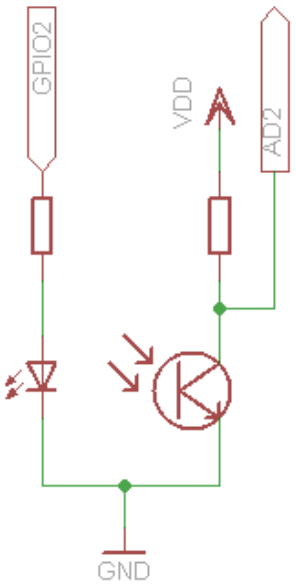
Napájecí obvody

Napájení robota je z důvodu rušení, vznikajícího pravděpodobně při spínání motorů, rozděleno na dvě části, základem každé části je Li-Ion akumulátor typu 18650 s napětím 3,7V a kapacitou 2Ah. K prvnímu akumulátoru jsou připojeny pouze výkonové části můstku pro řízení motorů, odběr z tohoto akumulátoru bývá průměrně asi 0,6A, avšak špičky sahají až k 1,5A. Druhý akumulátor napájí přímo Wi-Fi modul, který má na své desce vlastní stabilizátor 3,3V, a zároveň pomocí stabilizátoru s nízkým úbytkem LE33CZ se napájí stabilními 3,3V elektronika robota. Dva stabilizátory, jeden pro elektroniku a druhý pro Wi-Fi, byly zvoleny z důvodu nedostatečného výstupního proudu LE33CZ, který je roven 100mA. Wi-Fi modul si ve špičce při vysílání může vzít až 200mA. Robot je schopen na tyto dva plně nabité akumulátory jezdit asi 2 hodiny.

4.4 Senzory

Detektor překážek

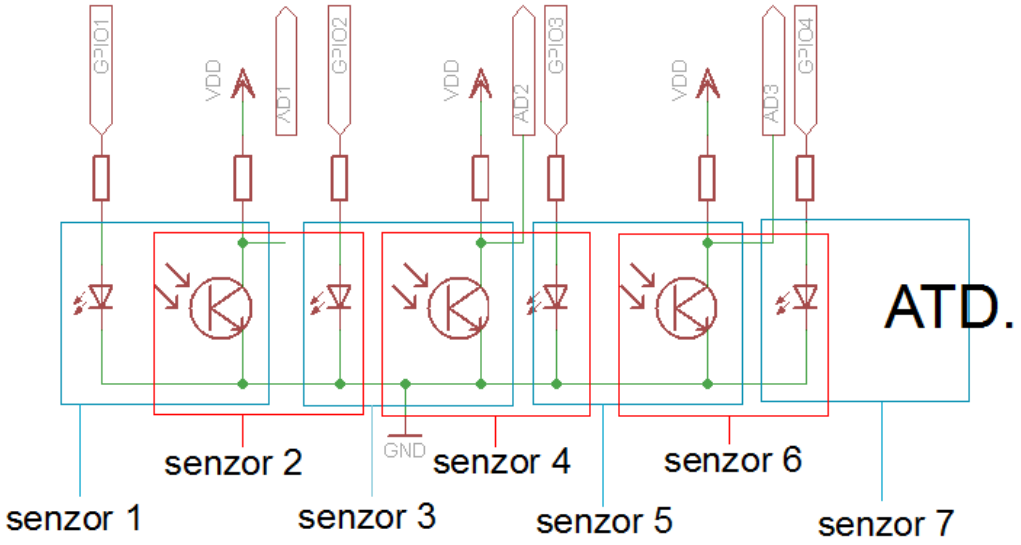
Jako detektor překážek je použita jednoduchá soustava IR fototranzistoru a IR LED. Vzdálenost od překážky mikrokontrolér získává pomocí A/D převodníku. Při takto sestaveném detektoru je možné spolehlivě detekovat překážky do asi 10cm od robota, poté se již hodně projevuje rušení okolního světla, hlavně zářivek. Tento senzor je ale pro většinu věcí pro robota dostačující. Schéma senzoru je vidět na *obrázku 9*:



Obrázek 10: Zapojení detektoru překážek

Senzor polohy čáry

Senzor čáry je sestaven z diskrétních součástek, šesti IR LED a pěti IR fototranzistorů o průměru 3mm. Ty jsou umístěny střídavě vedle sebe tak, jak je vyznačeno na *obrázku 11*.



Obrázek 11: Zapojení senzoru polohy čáry

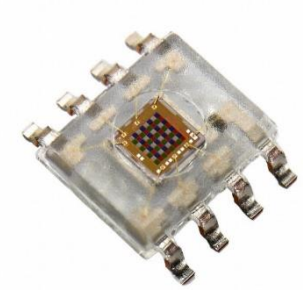
LED řídí mikrokontrolér pomocí GPIO, hodnoty odraženého světla čte A/D převodník.

Pokud je **senzor nad černou čárou**, čára světlo z LED pohltí a do fototranzistoru se téměř nic neodrazí, a proto je na výstupu senzoru napětí blízké napájecímu.

Umístíme-li **senzor nad čáru bílou**, světlo z LED se odrazí do fototranzistoru, obvodem začne procházet proud a napětí na výstupu se zmenší. Díky tomu, že výstupní napětí senzorů čteme pomocí A/D převodníku, máme lepší kontrolu nad pozicí čáry a je tu i možnost přizpůsobit robota k sledování méně kontrastní čáry pouze úpravou programu. Pokud bychom zjišťovali pouze logickou 0 nebo 1 na výstupech senzoru, tuto možnost bychom neměli.

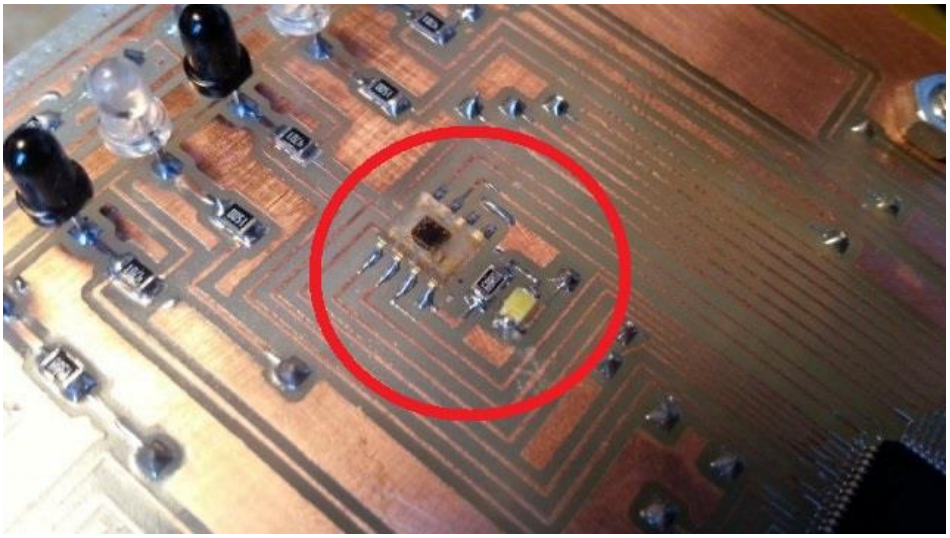
Senzor barvy čáry

Pro snímání barvy čáry jsme použili digitální RGB snímač TAOS TCS3200D [5]. Ten je umístěn na středu robota, hned za senzorem čáry. V senzoru je pole 8x8 fotodiod, s nichž 16 má zelený filtr, 16 modrý, 16 červený a 16 je bez filtru. Senzor na výstupu generuje signál o frekvenci, která odpovídá množství světla, které dopadlo na jeho fotodiody. Základní frekvence se nastaví pomocí 2 vstupů senzoru, stejně jako barevný filtr, který modul použije (další 2 vstupy). Výstup modulu lze vypnout přivedením log. 0 na vstup /OE.



Obrázek 12: TCS3200D [7]

Pro správné snímání barvy je potřeba přisvětlení, které je realizováno jednou bílou SMD LED umístěné poblíž senzoru (*obrázek 13*).



Obrázek 13: Detail senzoru barvy a přisvětlovací LED

5 Programování robota

Mikrokontolér programujeme v jazyce C, za použití vývojového prostředí MPLAB X a kompilátoru XC32. Program se do mikrokontroléru nahrává pomocí programátoru PICkit3 přes rozhraní ICSP. PICkit3 je vyráběn firmou Microchip Technology Inc, umožňuje programování všech mikrokontrolérů firmy Microchip Technology Inc, nastavení napájecího napětí pro programovaný mikrokontrolér v rozmezí 2 až 5V. Podporuje také ladění programů, přičemž je možné pozastavit periférie při dosažení breakpointu. Dále umožňuje naprogramování mikrokontroléru i bez nutnosti PC, jelikož lze program pro nahrání uložit do programátoru, který jej po připojení napájení a stisku tlačítka nahraje. Firmware programátoru lze také aktualizovat z PC. Tento programátor je velmi malý a je určen pro vývoj programů, pro komerční využití je třeba využít programátory ICD2 nebo ICD3, které jsou rychlejší a umožňují více možností ladění a mnoho dalších pokročilých vlastností.

Základní části programu, starající se o inicializaci a následnou obsluhu periferií, jsou z větší části vyřešeny přímým zápisem do konfiguračních registrů mikrokontroléru, pro zbytek (např. nastavení periferie OutputCompare pro generování PWM) jsou použita makra z knihovny *plib.h*, která velmi usnadňují práci. Příklad makra pro inicializaci jednotky OutputCompare1:

```
#define OpenOC1( config, value1, value2) ( OC1RS = (value1), OC1R = (value2), OC1CON = (config) )
```

5.1 Použité sběrnice

UART - Wi-Fi

Jednotka UART (Universal Asynchronous Receiver/ Transmitter) je nastavena na rychlost 115200kbps. Využívá se pro komunikaci s modulem Wi-Fi, jeho nastavování a odesílání dat do aplikace pro PC. Pro práci s jednotkou UART2 jsme vytvořili tyto funkce:

- *void WiFi_Init (void)* - nastaví jednotku UART2 a povolí přerušení při přijetí dat
- *void SendU2 (char in_c)* - odešle znak *in_c*

  Pro odesílání dat je z důvodu lepších možností formátování využita funkce:

- *printf (const char *, ...)*

Aby bylo možné odesílat data pomocí této funkce přes jednotku UART2, je nutné přesměrovat vypisování znaku na jednotku UART2, čehož jsme docílili napsáním vlastní funkce:

```
int write(int handle, void *buffer, unsigned int len) {
    int i;
    switch (handle) {
        case 0:
        case 1:
        case 2:
        default:
            for (i = len; i; --i) {
                SendU2(*(char*) buffer++);
            }
    }
    return (len);
}
```

Pokud tuto funkci kompilátor najde v uživatelském programu, použije ji namísto výchozí verze z knihoven.

5.2 Řízení motorů - PWM

Jednotky OutputCompare 1, 2, 3 a 5 jsou využity pro hardwarové generování signálu PWM, za pomoci čítače/časovače TMR2, který generuje signál o frekvenci 800Hz. Výsledný signál PWM je využit pro řízení rychlostí levého a pravého motoru, přivedením na můstky HT6751B. Jednotkami 1 a 5 se řídí pravý motor, levý motor obsluhují jednotky 2 a 3. Funkce můstků je zřejmá z *tabulky 1*.

Tabulka 1: Signály pro řízení motorů

	Levý motor		**Pravý motor**	
	IN1 - OC2	**IN2 - OC3**	**IN1 - OC5**	**IN2 - OC1**
Vpřed	PWM	0	0	PWM
Vzad	0	PWM	PWM	0
Odpojen	0	0	0	0
Zabrzdit	1	1	1	1

IN1, IN2 jsou vstupy jednotlivých můstků, vstup IN3 je napevno uzemněn na plošném spoji.

OC1, OC2, OC3 a OC5 jsou jednotky OutputCompare, ke kterým jsou připojeny vstupy můstků.

Vzhledem k tomu, že je na všech těchto pinech generován signál PWM, jsou stálé logické 0 nebo 1 vytvořeny nastavením střídy PWM na 0, respektive 100%.

Střídu signálu PWM je možno nastavit v 12800 krocích, což odpovídá 0-100%. Vlastní signál PWM je vytvořen vložením požadované hodnoty střídy z proměnné do registru OCxRS (x je číslo modulu). To se děje v obsluze přerušení od TMR2, zároveň jednotka OC přesune tuto hodnotu z OCxRS do primárního registru OCxR. Hned jak se TMR2 vynuluje, OC nastaví daný výstup do logické 1. Jakmile TMR2 dosáhne hodnoty OCxR, jednotka daný výstup vynuluje. TMR2 čítá až do hodnoty uložené ve svém preload registru PR2 (PR2=perioda-1; v našem případě 12800-1=12799), poté se TMR2 vynuluje, čímž vygeneruje přerušení a celý cyklus se opakuje. Číslo 12800 bylo zvoleno jako kompromis mezi rozlišením a frekvencí signálu PWM.

Pro ovládání motorů jsme vytvořili tyto funkce:

- *void LMotor_SetPercent(char direction, int speed_percent)*

  -nastaví rychlost a směr otáčení levého motoru (rychost v procentech 0-100)

- *void RMotor_SetPercent(char direction, int speed_percent)*

  - nastaví rychlost a směr otáčení pravého motoru (rychost v procentech 0-100)

- *void LMotor_Set(char direction, int speed)*

  - nastaví rychlost a směr otáčení levého motoru (rychost 0-12800)

- *void RMotor_Set(char direction, int speed)*

  - nastaví rychlost a směr otáčení pravého motoru (rychlost 0-12800)

Speciální funkce:

- *void SetMotors_LineFollowing(int psd, int speed)*

  - nastaví rychlost motoru podle výstupu z regulátoru psd, se střední rychlostí speed

5.3 Sledování čáry

Převodník A/D

10-bitový A/D převodník je využíván pro čtení ze senzoru překážek na přední straně robota a senzorů čáry na spodního straně robota. Základní rutiny pro inicializaci a čtení analogových vstupů jsou umístěny v souboru pi32_adc.c.

Senzor čáry

Robot detekuje čáru pomocí kombinace IR fototranzistorů (T) a IR diod (D) (detaily konstrukce viz. 4.4 Senzory - - Senzor polohy čáry). Blokové uspořádání je patrné z *obrázku 14*.

Obrázek 14: Blokové uspořádání senzoru čáry

Nejprve se rozsvítí LED D1, A/D převodníkem se přečte T1, následně zhasne D1, rozsvítí se D2 a znovu se přečte T1. Dioda D2 zhasne a čtení pokračuje stejně s fototranzistorem T2, až se přečtou všechny senzory. Toto uspořádání je vhodné z důvodu nízkých nákladů (malé množství potřebných součástek), ale také umožňuje zjistit pozici čáry v deseti bodech.

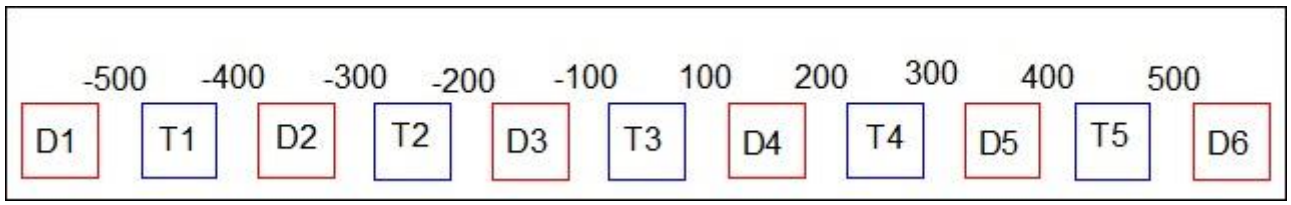
Zjišťování čáry následně pokračuje připočítáním kalibrace, která byla provedena přečtením 100000 hodnot na bílé tabuli a následným zprůměrováním hodnot pro každý z deseti bodů.

Samotné připočítání kalibrace probíhá tak, že se od přečtených hodnot odečtou kalibrační hodnoty, výsledek se následně vynásobí výrazem *1023/(1023-kalibrační_hodnota)*, abychom výsledné číslo i po odečtení kalibrace měli v rozsahu 0-1023. To je patrné z části kódu:

```
void Add_Calibration(int source[10], int destination[10]) {
	int z;
	for (z = 0; z < 10; z++) {//pripocitani kalibrace
		if (source[z] < LineConf[z]) source[z] = LineConf[z];
		destination[z] = (source[z] - LineConf[z])*(float) (1023 / (1023 - LineConf[z]));
	}
}
```

V *LineConf [10]* jsou uloženy hodnoty kalibrace.

Každý senzor přiřazenou svoji "chybu", na *obrázku 15* je pohled shora na senzory.



Obrázek 15: Každému senzoru je přiřazena chyba

Tyto chyby jsou uloženy v poli *errors[10]*.

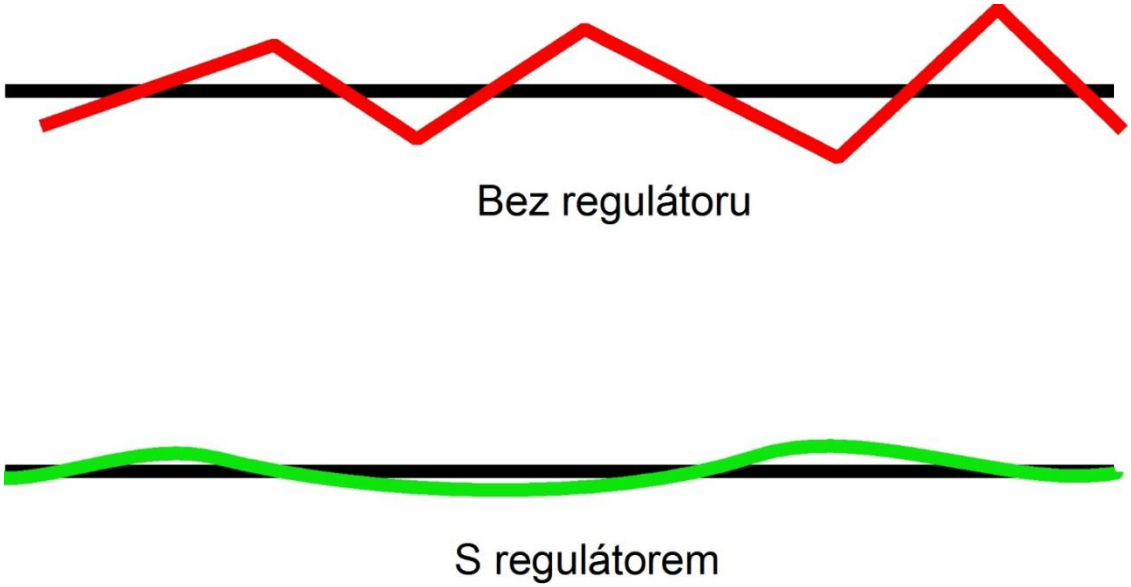
Pro vylepšení rozlišení nevyhledáváme senzor s nejnižším odrazem a jeho chyba by byla poloha čáry, ale využíváme vážený průměr. Ten spočívá v součtu součinů přečtených hodnot senzorů a jejich chyb a následném vydělení součtem přečtených hodnot. To je patrné z části kódu:

```
int GetLine_Pos(int line[10]) {
	int i;
	long polohy = 0;
	long hodnoty = 0;
	for (i = 0; i < 10; i++) {
		polohy += line[i] * errors[i];
	}
	for (i = 0; i < 10; i++) {
		hodnoty += line[i];
	}
	if (hodnoty == 0) return 0; //nelze dělit nulou
	return (polohy / hodnoty); //(součet součinů váhy a hodnoty senzoru) vydělený součtem
hodnot senzoru
}
```

Tím dosáhneme rozlišení 1000 bodů (-500 až 500), čímž může být sledování čáry daleko přesnější a plynulejší.

PSD regulátor

Regulátor PSD se používá proto, aby při sledování čáry byla dráha robota co nejplynulejší a nejkratší. Rozdíl mezi sledováním čáry s pomocí tohoto regulátoru a bez něj je patrný z *obrázku 16*.



Obrázek 16: Sledování čáry bez použití regulátoru a při použití regulátoru PSD

Při sledování čáry je cílem robota, aby čára ležela vždy pod jeho středem. Aby toho dosáhl, je výsledná odchylka od čáry (= chyba - pro detaily výpočtu této chyby viz Senzor čáry) matematicky zpracována v rovnicích PSD regulátoru. V programu je to vyjádřeno funkcí *PSD_Regulator*:

```
int PSD_Regulator(int error) {
      int delta = error - PrevError;
      sumaE += error;
      PrevError = error;
      return ((p * error) + (s * sumaE) + (d * delta));
}
```

- *error* je odchylka od čáry
- *PrevError* je minulá chyba
- *SumaE* je součet všech předchozích chyb
- *delta* je rozdíl aktuální chyby a minulé chyby
- *p*, *s*, *d* jsou konstanty PSD regulátoru

Konstanty p, s a d je nutné pro správnou funkci regulátoru vyladit, teprve pak bude robot sledovat čáru naprosto plynule. Toto nastavování konstant PSD regulátoru lze nastavovat přes rozhraní Wi-Fi díky aplikaci pro počítač.

Reakce na výstup PSD regulátoru

Podle výsledku PSD regulátoru se nastaví rychlosti levého a pravého motoru:

```
void SetMotors_LineFollowing(int psd, int speed) {
      if (psd < -speed) psd = -speed;
      if (psd > speed) psd = speed;
      LMotor_Set(FORWARD, speed + psd);
      RMotor_Set(FORWARD, speed - psd);
}
```

5.4 Rozlišení barvy čáry

Pro snímání barvy využíváme digitální RGB senzor TAOS TCS3200D a přisvětlovací LED (detaily konstrukce viz. 4.4.3 Senzor barvy čáry). Senzor se nastavuje pomocí 5 vývodů mikrokontroléru, proto jsme vytvořili tato makra:

- *ColorSensor_Led (x)*
- *ColorSensor_OE (x)*

a tyto funkce:

- *void ColorSensor_Init ()*
- *void ColorSensor_SetFreq (char freq)*
- *void ColorSensor_SetFilter (char filter)*

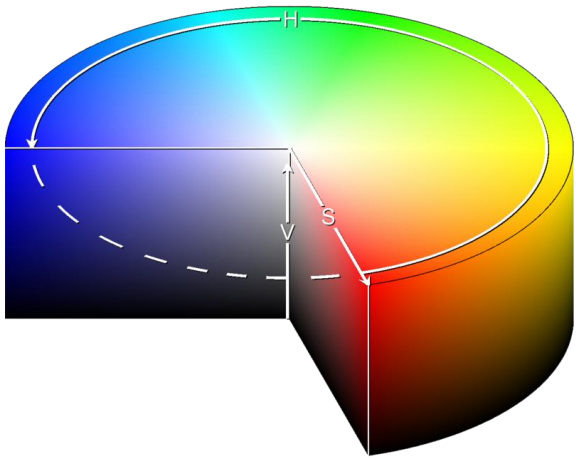
Pro lepší přehlednost kódu jsou hodnotám pro nastavení filtrů a frekvencí přiřazeny textové ekvivalenty, např. *#define COLOR_FILTER_BLUE 0b10*

Samotné čtení barvy je realizováno na pozadí, provádí se ve dvou přerušeních. Jedno přerušení, s nižší prioritou, je vyvoláno vždy při vzestupné hraně signálu z výstupu senzoru. V obsluze přerušení se pouze inkrementuje proměnná, obsahující počet přerušení. Druhé přerušení s vyšší prioritou je vyvoláno časovačem/čítačem TMR3. V obsluze tohoto přerušení se nejprve uloží proměnná obsahující počet 1. přerušení do odpovídající buňky pole c*olorRGBW[4]*, nastaví se následující filtr z pole *_filtersCS[4]* a celý děj se znovu opakuje.

Určení barvy je ve spektru RGB obtížné, proto si program nejprve tyto hodnoty převede do spektra HSV funkcí *void RGBtoHSV(int rgbw[3], colorHSV_t *hsv).* Pro snazší uložení dat spektra HSV (odstín, sytost, jas) a přehlednost kódu využíváme strukturu *colorHSV_t*:

```
typedef struct {
	int H;
	float S;
	float V;
} colorHSV_t;
```

Ve spektru HSV je mnohem jednodušší určení jedné ze tří základních barev, jelikož ty poznáme podle odstínu (H) a černou podle jasu (V) (*obrázek 17*).



Obrázek 17: Spektrum HSV [6]

Robot dokáže spolehlivě rozpoznat červenou, modrou, zelenou a černou barvu. Pokud je jas nižší než 0,3, barva je černá. Je-li odstín vždy +-40 od základní barvy, výsledkem je ta základní barva (např. zelená H=120, robot vyhodnotí barvu jako zelenou, pokud je H v rozmezí 80-160)

5.5 Senzor překážek

Detekce překážek je vyřešena jednoduchým senzorem, který se skládá z jednoho IR fototranzistoru a IR diody (detaily konstrukce viz. 4.4.1 Detektor překážek). Nejprve se zhasne IR LED, A/D převodníkem se přečte hodnota okolního světla, poté se rozsvítí IR LED a přečte se znovu hodnota osvětlení, následně IR LED zhasne. Protože je IR fototranzistor zapojen mezi GND a analogový vstup, výsledkem měření bude rozdíl hodnoty okolního světla a hodnoty osvětlení IR LED. Čím blíže bude překážka, tím vyšší číslo bude výsledkem. Tento způsob je spíše indikací, než měřením vzdálenosti, ale pro naše využití je zcela dostačující. Problémy mohou pouze způsobovat zářivky, kvůli kterým jsme byli nuceni upravit vzdálenost detekce překážky na asi 5cm. Původně jsme zamýšleli využití senzoru Sharp GP2Y0A21YK0F, ten ale nepracuje při napětí 3,3V.

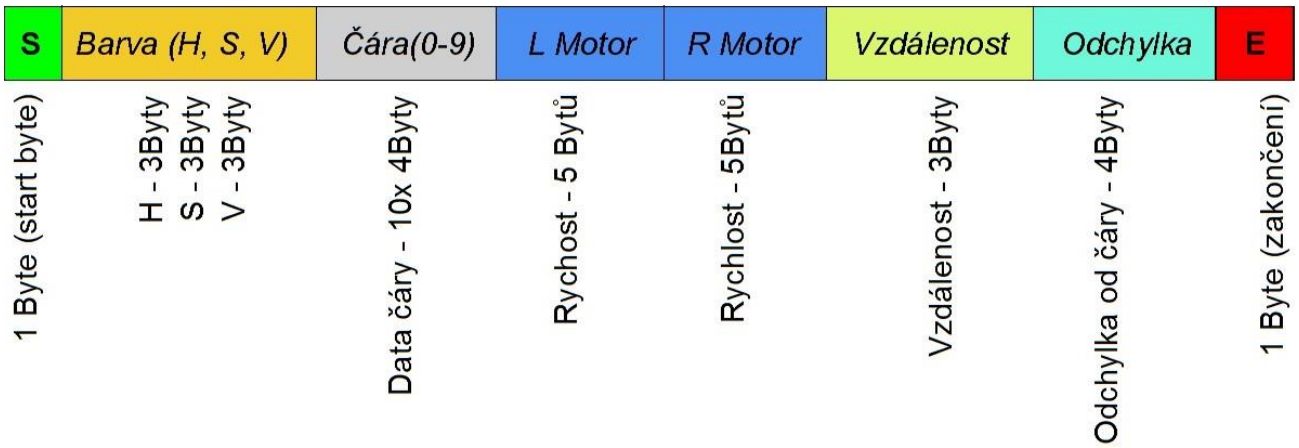
5.6 Aplikace pro PC

Odesílaná data

Pro odesílání dat do PC jsme vytvořili funkci Send_Telemetric:

- *void Send_Telemetric(colorHSV_t clr, int line[10], int LMotor_Speed, int RMotor_Speed, int Front_distance, int line_err)*

Jeden paket obsahuje data ze senzoru barvy, čáry, rychlosti levého a pravého motoru, vzdálenost od překážky a aktuální odchylku od čáry (*obrázek 18*).

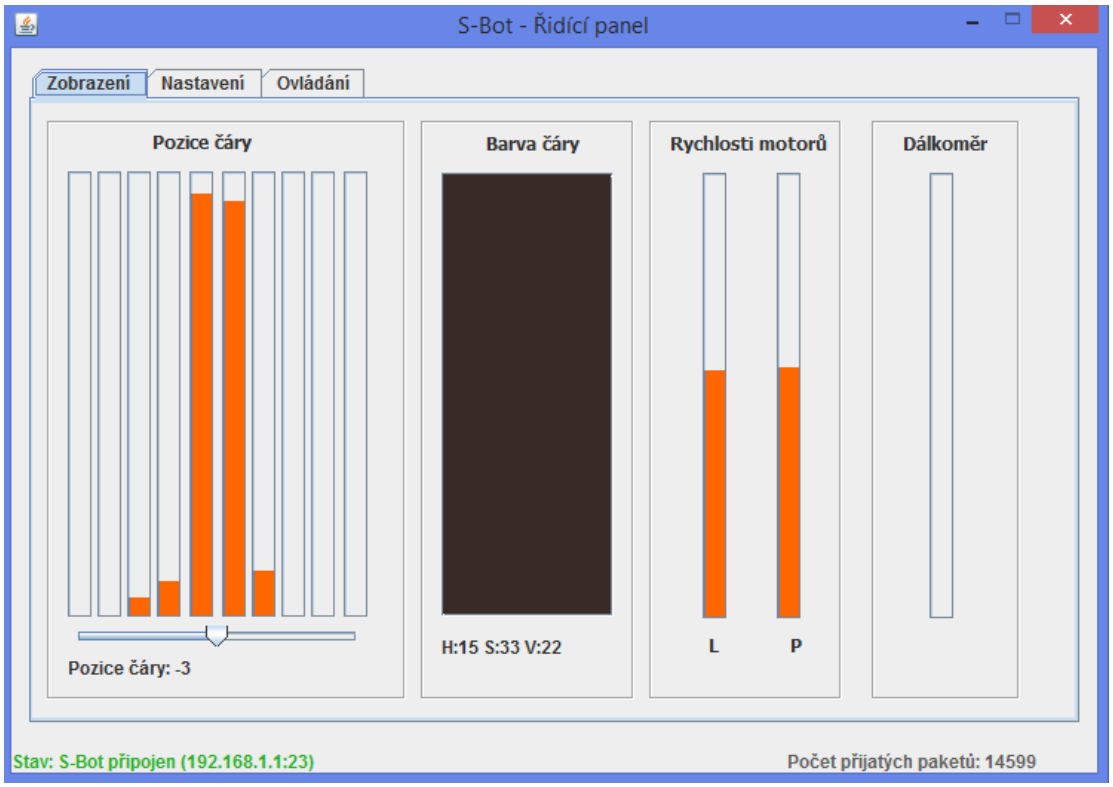


Obrázek 18: Paket komunikace mezi robotem a PC

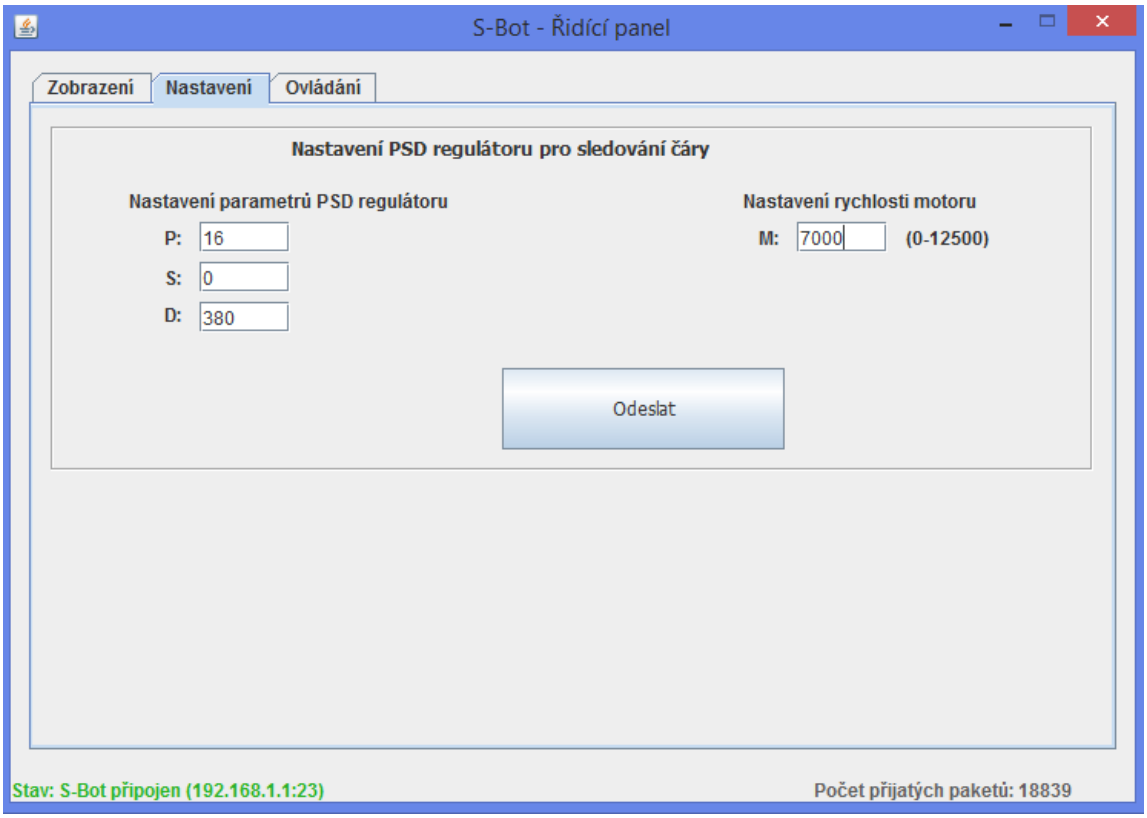
Popis aplikace

Aplikace pro PC je napsána v jazyce Java za použití vývojového prostředí NetBeans. Komunikace probíhá protokolem TCP-IP na portu 23, počítač se připojí do sítě vytvořené robotem. Toto připojení zaručeně funguje s interní anténou na vzdálenost 15m přes zdi, při větší vzdálenosti již dochází k chybám v komunikaci, což by nejspíše bylo možné eliminovat přidáním kontrolního součtu.

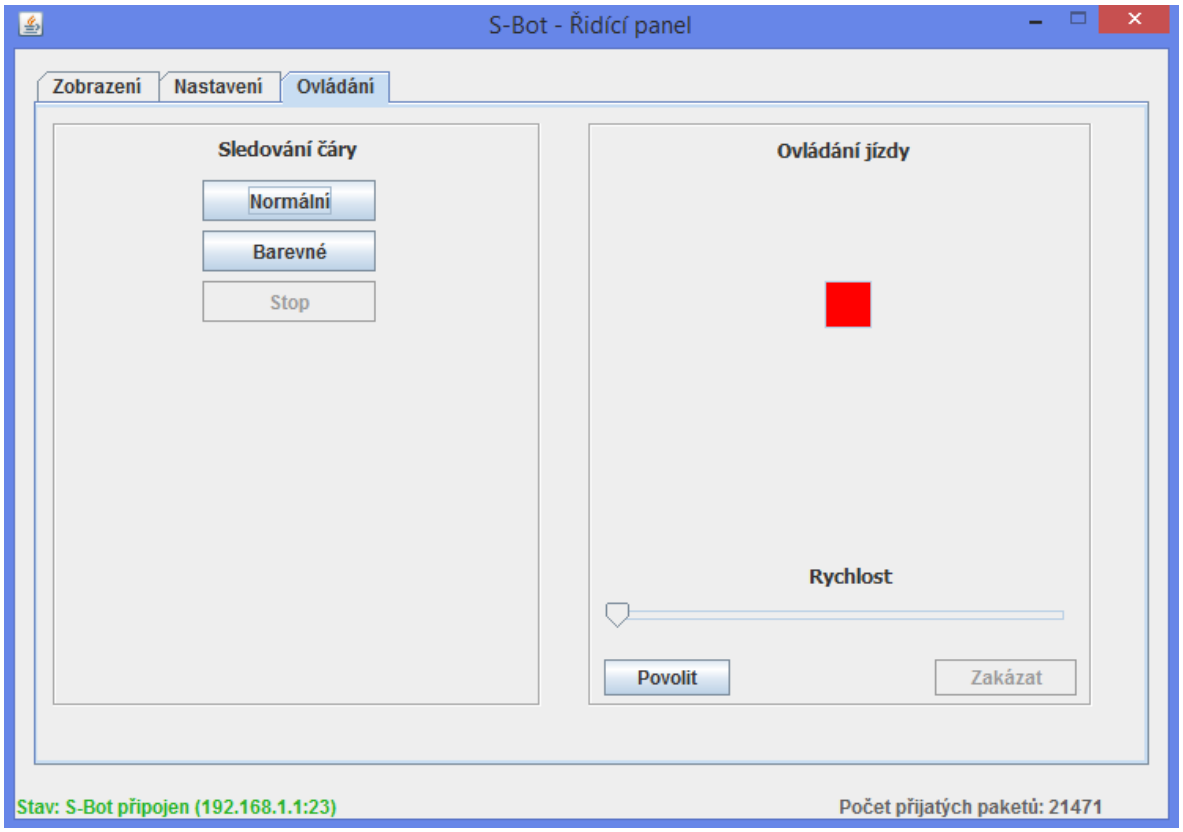
Aplikace umožňuje v reálném čase monitorovat senzory čáry, barvy čáry, překážek a rychlosti motorů (*obrázek 19*). Dále je možné dálkové ladění konstant PSD regulátoru a nastavování rychlosti pro sledování čáry (*obrázek 20*). Pomocí programu můžeme také přepínat mezi režimy klasického sledování čáry nebo sledováním barevné čáry (modrá - zrychlí, červená - zpomalí, černá - normální rychlost), případně jej i ovládat pomocí šipek na klávesnici s možností kontroly rychlosti a zastavení při nalezení překážky (*obrázek 21*).



Obrázek 19: Monitorování senzorů a rychlostí motorů v reálném čase pomocí PC aplikace



Obrázek 20: Nastavování robota přes Wi-Fi pomocí PC aplikace



Obrázek 21: Nastavení režimu sledování čáry a dálkové ovládání robota pomocí šipek na klávesnici PC

6 Závěr

Podařilo se nám zkonstruovat i naprogramovat robota schopného komunikace pomocí Wi-Fi a sledování barevné čáry. Při sestavování robota, jeho programování i řešení problémů, např. rušení od motorů, jsme získali mnoho nových zkušeností. Tato práce nám také pomohla se zdokonalit v programování mikrokontrolérů PIC a také se naučit programování v jazyce Java pro PC.

Nyní je PSD regulátor vyladěn pro 2/3 maximální rychlosti, další ladění je již obtížné. Robot spolehlivě rozpozná modrou a červenou barvu, zelenou určí s nižší přesností, proto není využita. Určování polohy čáry je přesné, díky výpočtu váženého průměru (podrobněji viz 5.3 Senzor čáry) ji určí v 1000 bodech.

Robota bychom chtěli i nadále vylepšovat, například optimalizovat velikost odesílaných dat (nyní odesílá 52B), přidání kontrolního součtu pro omezení chyb při přenosu, ale také na robota osadit nové snímače, jako akcelerometr či elektronický kompas, aby bylo možné se lépe orientovat v prostoru. Také bychom chtěli upravit robota pro hledání cesty v bludišti.

7 Seznam obrázků

8 Seznam tabulek

9 Seznam vzorců

10 Seznam příloh

Složka PC – Program pro PC napsaný v Javě

Složka PCB – Motiv a schema DPS

Složka Schéma – Blokové schema MCU

Složka Videa – Videa sledování barevné čáry a rychlého sledování černé čáry

11 Literatura

[1] MICROCHIP TECHNOLOGY INCORPORATED. *Microchip Technology Inc.* [online]. 1998, 2013 [cit. 2015-03-06]. Dostupné z: http://www.microchip.com/

[2] MICROCHIP TECHNOLOGY INCORPORATED. *PIC32MX5XX/6XX/7XX* [online]. U.S.A.: Microchip Technology Incorporated, 2009, 2013 [cit. 2015-03-06]. ISBN 978-1-162077-125-9. Dostupné z: http://ww1.microchip.com/downloads/en/DeviceDoc/61156H.pdf

[3] ROVING NETWORKS INCORPORATED. *RN - 131G & RN - 131C 802.11 b/g Wireless LAN Module* [online]. Version 3.2r. 2012, 9.4.2012 [cit. 2015-03-13]. Dostupné z: http://ww1.microchip.com/downloads/en/DeviceDoc/rn-131-ds-v3.2r.pdf

[4] HOLTEK SEMICONDUCTOR INC. *HT6751A/HT6751B* [online]. Rev 1.10, 2008, 2012 [cit. 2015-03-13]. Dostupné z: http://www.holtek.com.tw/pdf/consumer/ht6751v110.pdf

[5] TEXAS ADVANCED OPTOELECTRONIC SOLUTIONS INCORPORATED. *TCS3200, TCS3210* [online]. 2009 [cit. 2015-03-13]. Dostupné z: http://www.mouser.com/catalog/specsheets/TCS3200-E11.pdf

[6] (3UCKY(3ALL. HSV cylinder. In: . *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2015-03-06]. Dostupné z: http://cs.wikipedia.org/wiki/HSV#mediaviewer/File:HSV_cylinder.png

[7] TCS3200D. DIGI-KEY CORPORATION. *Digi-Key* [online]. 1995, 2015 [cit. 2015-03-06]. Dostupné z: http://media.digikey.com/Photos/AMS-Taos%20USA%20Photos/TCS3200D.jpg

[8] HT6751B. SNAIL SHOP. *Snail Shop* [online]. [cit. 2015-03-06]. Dostupné z: http://www.snailshop.cz/1926-large_default/ht6751b.jpg

[9] MICROCHIP RN-131-EK EVAL KIT. A PREMIER FARNELL COMPANY. *A Premier Farnell Company* [online]. 2015 [cit. 2015-03-06]. Dostupné z: http://uk.farnell.com/productimages/standard/en_GB/2281721-40.jpg

[10] PIC32MX695F512L. FUTURLEC. *Futurlec* [online]. 2015 [cit. 2015-03-06]. Dostupné z: http://www.futurlec.com/Pictures/Package/PIC32MX695F512L.jpg