Středoškolská technika 2017

Setkání a prezentace prací středoškolských studentů na ČVUT

Robotická platforma

Jakub Kolár

SPŠE Ječná
Ječná 30, Praha 2

Prohlášení

Prohlašuji, že jsem svou maturitní práci vypracoval/a samostatně a použil/a jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů.

Prohlašuji, že tištěná verze a elektronická verze soutěžní práce jsou shodné.

Nemám závažný důvod proti zpřístupňování této práce 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 Praze dne 30.11.2016 ……

Jakub Kolár

Poděkování

Děkuji, konzultantovi práce Ing. Zdeňkovi Velichovi za pomoc a rady ohladně mé práce. Ještě bych rád poděkoval Ing. Miroslavu Ottovi za pomoc, rady a konstruktivní kritiku. Dále pak také Ing. Tomáši Žitnému za jeho cenné rady v oblasti elektroniky.

Anotace

Práce popisuje vývoj a stavbu robotické platformy. Přímo pak využití platformy jako průzkumného robotického vozítka, které lze ovládat pomocí počítačové aplikace. Práce se dělí na tři části: mechanická konstrukce, elektronika a programování.

Klíčová slova

Robot, platforma, modulární robot, vzdálené ovládání

Annotation

This work deals with developing, building and describing of a robotic platform, especially, with using it as a research robotic cart, which is possible to control by the computer application. The work is divided into three parts: mechanical construction, electronics and programming.

Keywords

Robot, platform, modular robot, remote control

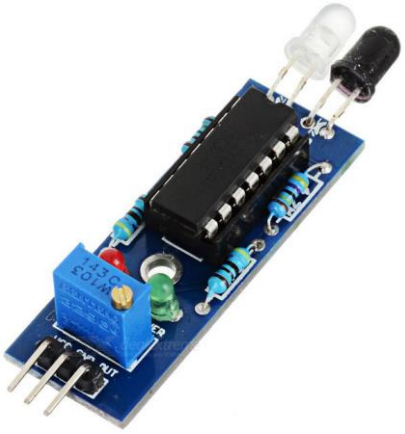
OBSAH

1 Úvod 6
2 Mechanická konstrukce 7
2.1 Návrh 7
2.2 Výroba 8
Motory 8
Kola 9
Akumulátor 9
2.3 Hotová konstrukce 10
3 Elektronika 11
3.1 Co je PWM 11
PWM 11
3.2 Blokové schéma 12
3.3 Schéma a jeho popis 13
LM1084IT 14
74S08 14
3.4 Jednotlivé komponenty a jejich funkce 16
Arduino 17
Raspberry PI 18
H-můstek(L298N) 19
H-můstek(VNH2SP30) 21
3.5 Plošný spoj a jeho výroba 23
4 Programování a síť 25
4.1 Programování platformy Arduino 25
4.2 Instalace systému na Raspberry PI 27
4.3 Programy pro Raspberry PI a jejich spouštění 27
4.4 Programování počítačového rozhraní v jazyce C# 27
4.5 Nastavování sítě 30
5 Závěr 30
6 Seznam zkratek 31
7 Zdroje 31
8 Obrázky 31
9 přílohy 32

1 Úvod

V dnešním světě je k vidění mnoho různých typů, značek a použití robotů. Nastává zde problém, že na každou práci je třeba jiného robota, často bohužel od jiného výrobce a jsou zde problémy se vzájemnou interakcí a komunikací. Nejlepší by tedy bylo udělat robota, který by zvládl všechny práce, například výměnou jednotlivých modulů. Můj robot bohužel není tak dokonalý, aby všechny procesy zastal, ale lze z něho pomocí výměny či přidání modulu udělat něco trochu jiného. Ve Škoda auto mají kupříkladu robotické skladníky, tento robot ovšem již nezvládá nic jiného, je to pouze skladník, který operuje po předem vyhrazeném prostoru. Co kdyby se ovšem objevila potřeba sjet s robotem z jeho dráhy a dálkově ho ovládat, ať už kvůli nestandardní objednávce nebo kontrole objektu (hlídač) ? U takovýchto robotů tato možnost není. Mého robota ale lze bez problémů ovládat přes počítačovou aplikaci. Pomocí jeho kamery je zabezpečen vizuální kontakt (oči), pak už jen záleží na tom co operátor požaduje.

Dalším využitím (které jsem ovšem netestoval) je kupříkladu využití robota jako výše zmíněného transportéru v Iot skladu. Možností dalšího využití je mnoho, a to zejména díky otevřené platformě Arduino. Uživateli (operátor) stačí poté jen zapojit požadovaný senzor a dopsat pro něj podprogram. V rychlosti mě napadá například senzor čáry (Obr. 1), který je použitelný pro mód transportéru. Senzor snímá černou čáru, dle výstupních dat se poté upravuje spínací signál pro motory a robot tak jede po čáře.



Obr.1 Senzor Čáry

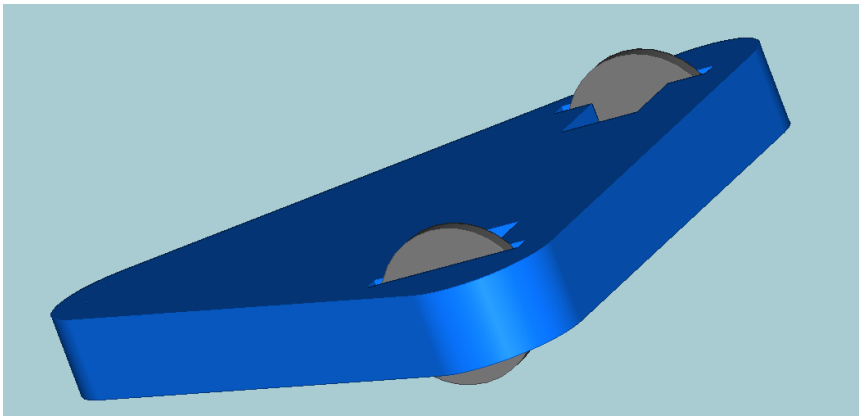
2 MECHANICKÁ KONSTRUKCE

Kapitola Mechanická konstrukce je rozdělena do tří částí :

2.1 Návrh
2.2 Výroba
2.3 Hotová konstrukce

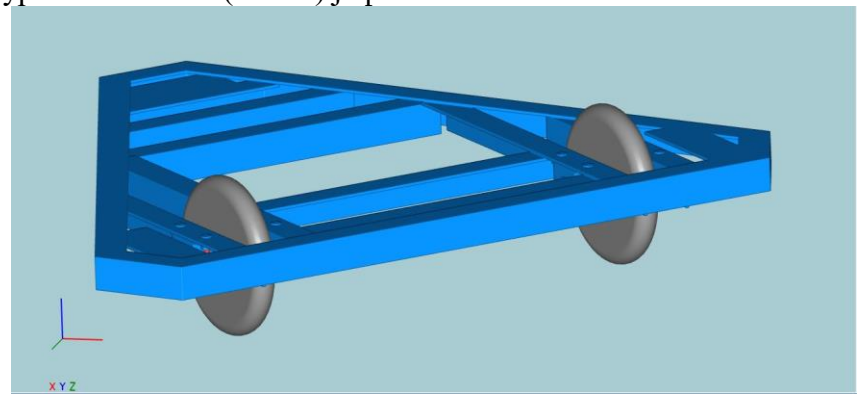
2.1 Návrh

Navržení podoby robota bylo prvním krokem. Tvar podvozku byl zvolen trojúhelníkovitý z důvodu stability celé konstrukce. Trojúhelníkovitý tvar podvozku má narozdíl od čtvercovitého podvozku výhodu v tom, že má tři opěrné body. U čtvercovitého podvozku nemusí jedno kolo přiléhat k zemi a podvozek se může stát nestabilním. Konstrukce má zaoblené hrany, a to z důvodu výrazného snížení rizika zaseknutí robota o zeď. Návrh byl vytvořen v grafickém prostředí VariCAD. Lze také použít programy: AutoCAD, Google Sketch Up...
Pro první vizualizaci podvozku byl vytvořen návrh viz (Obr. 2).



Obr.2 První návrh

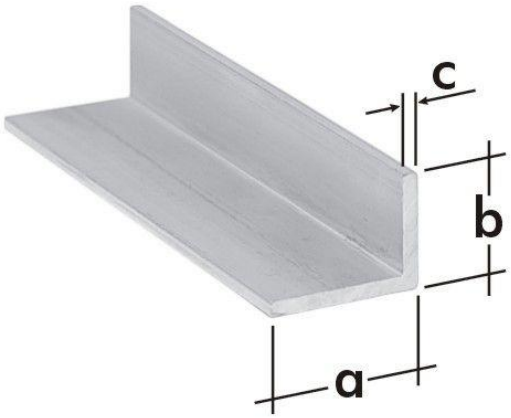
Po vizualizaci jsem vytvořil výrobní výkres (soubor v příloze), dle kterého jsem postupoval při výrobě prototypu. Na obrázku (Obr. 3) je pohled izo v zobrazení 3D.



Obr.3 Hotový model

2.2 Výroba

Jako materiál byl použit hliník ve formě hliníkových L profilů (Obr. 4) o rozměrech : 25 x 25 x 1.5 [mm]. Hliník byl zvolen jako materiál z důvodu jeho vysoké pevnosti a nízké váhy, hliník má také vysokou odolnost vůči korozi. Profily byly dle výkresu naměřeny a následně nařezány. Ke spojování profilů byly použity trhací nýty. Dále byly na rám přinýtovány nosné profily pro ukotvení motorů, zadního kola a držáku akumulátoru.



Obr.4 Al profil

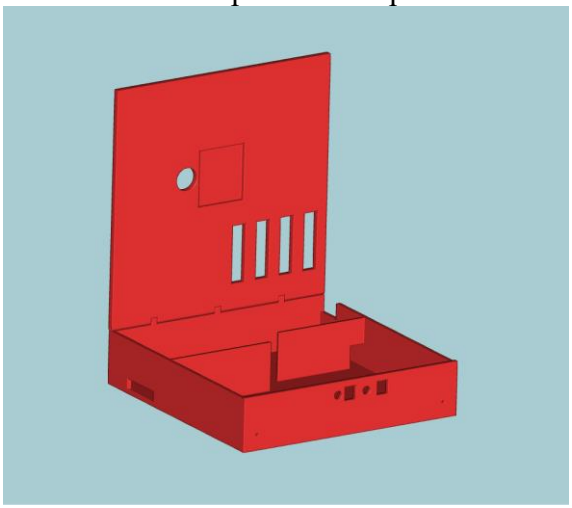
Motory jsou stejnosměrné, zpřevodované převodovkou s převodovým poměrem 1:3 z důvodu vyššího kroutícího momentu a z důvodu snížení otáček. Motory mají pracovní napětí 6 – 8V. Odebíraný proud závisí na zátěži motoru, odebíraný proud při rozjezdu je 2 – 5 A a v jízdním režimu 1 – 3 A.

Obr.5 Motory

Kola (Obr. 6) jsou původně z koloběžky. Plášť je vyroben z polyuretanu a střed kola z ABS plastu. Jejich rozměry jsou: vnitřní průměr kola 20 mm , vnější průměr 200 mm a tloušťka 35 mm. Vnitřní průměr kola byl přizpůsoben použitým hřídelům silonovou vložkou.

Obr.6 Kolo

Akumulátor je standardní gelový, olověný s nominálním napětím 6V a proudovou kapacitou 10Ah. Akumulátor je upevněn pomocí speciálního držáku, rovněž vyrobeného z hliníkových profilů. Z důvodu ochrany elektroniky před vnějším poškozením nebo zkraty je elektronika uložena v plastové krabici (Obr. 7). Plast jako materiál krabice byl zvolen z důvodu jeho izolačních vlastností (nevodič) a dostatečné odolnosti vůči vnějším vlivům. Krabice byla navržena opět v opět v programu VariCAD. Výstupem z programu VariCAD je soubor s příponou .stl, který slouží k vygenerování instrukčního souboru s přípomou .Gcode (Obr. 8) programem Cura. V programu Cura se také dále nastavují parametry 3D tisku jako je teplota podložky, pomocná vrstva výrobku a jiné. V instrukčním souboru s příponou .Gcode jsou instrukce, které řídí chod tepelné tiskárny přímo z hlediska pozicování nástroje. V instrukčním souboru jsou navíc také obsaženy další instrukce .Mcode, ovládající doprovodné funkce: teplotu trysky a chlazení. Z Cury bylo nutné vyexportovat instrukční soubor na SD kartu, kterou jsem následně vložil do slotu 3D tiskárny. Poté proběhlo pouze dodatečné nastavení na displeji tiskárny a následovalo spuštění tisku. Tisk proběhl bez problémů a trval 20:32 hodin.



Obr.7 Elektro krabice

```
;FLAVOR:UltiGCode
;TIME:102595
;MATERIAL:114062
;MATERIAL2:0
;NOZZLE_DIAMETER:0.400000
;NOZZLE_DIAMETER2:0.400000

;Layer count: 328
;LAYER:0
M107
G0 F7200 X191.623 Y70.708 Z0.260
;TYPE:SKIRT
G1 F1200 X191.464 Y71.234 E0.05715
G1 X191.410 Y71.779 E0.11411
G1 X191.463 Y72.317 E0.17033
G1 X191.624 Y72.847 E0.22794
G1 X191.879 Y73.324 E0.28419
G1 X192.226 Y73.747 E0.34109
G1 X192.653 Y74.097 E0.39851
G1 X193.132 Y74.353 E0.45499
G1 X193.657 Y74.512 E0.51204
G1 X194.202 Y74.566 E0.56900
G1 X194.740 Y74.513 E0.62522
G1 X195.269 Y74.353 E0.68270
G1 X195.744 Y74.098 E0.73877
G1 X196.175 Y73.745 E0.79671
G1 X196.518 Y73.328 E0.85286
G1 X196.777 Y72.841 E0.91023
G1 X196.936 Y72.318 E0.96708
G1 X196.990 Y71.774 E1.02393
G1 X196.937 Y71.236 E1.08015
G1 X196.777 Y70.710 E1.13733
G1 X196.519 Y70.225 E1.19446
G1 X196.174 Y69.805 E1.25099
G1 X195.751 Y69.458 E1.30789
G1 X195.264 Y69.199 E1.36526
G1 X194.741 Y69.040 E1.42211
G1 X194.197 Y68.986 E1.47896
G1 X193.659 Y69.039 E1.53518
G1 X193.131 Y69.199 E1.59256
G1 X192.652 Y69.456 E1.64910
```

Obr. 8 Ukázka souboru s příponou .Gcode

2.3 Hotová konstrukce

V této části je již hotová konstrukce.

Obr. 9 Platforma (robot)

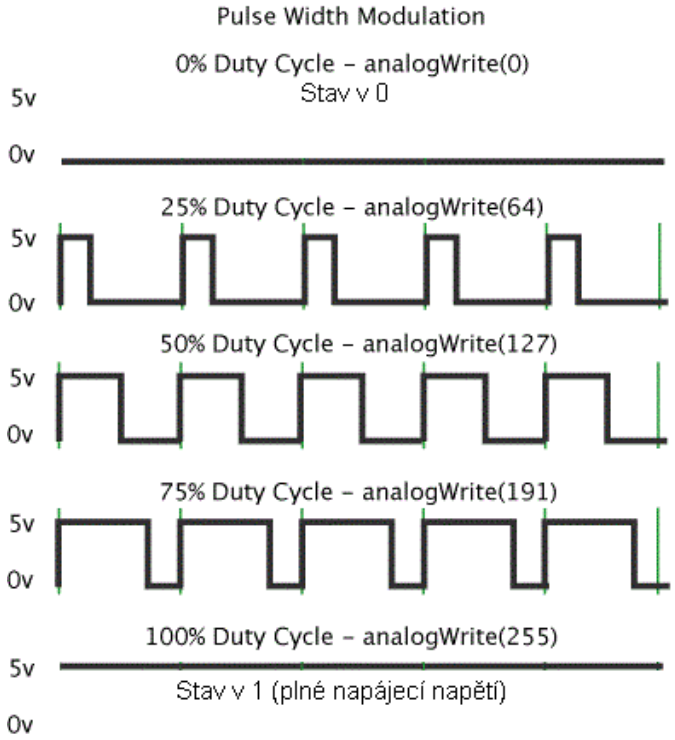
3 ELEKTRONIKA

Kapitola Elektronika je rozdělena do pěti částí :

3.1 Co je PWM
3.2 Blokové schéma
3.3 Schéma a jeho popis
3.4 Jednotlivé komponenty a jejich funkce
3.5 Plošný spoj a jeho výroba

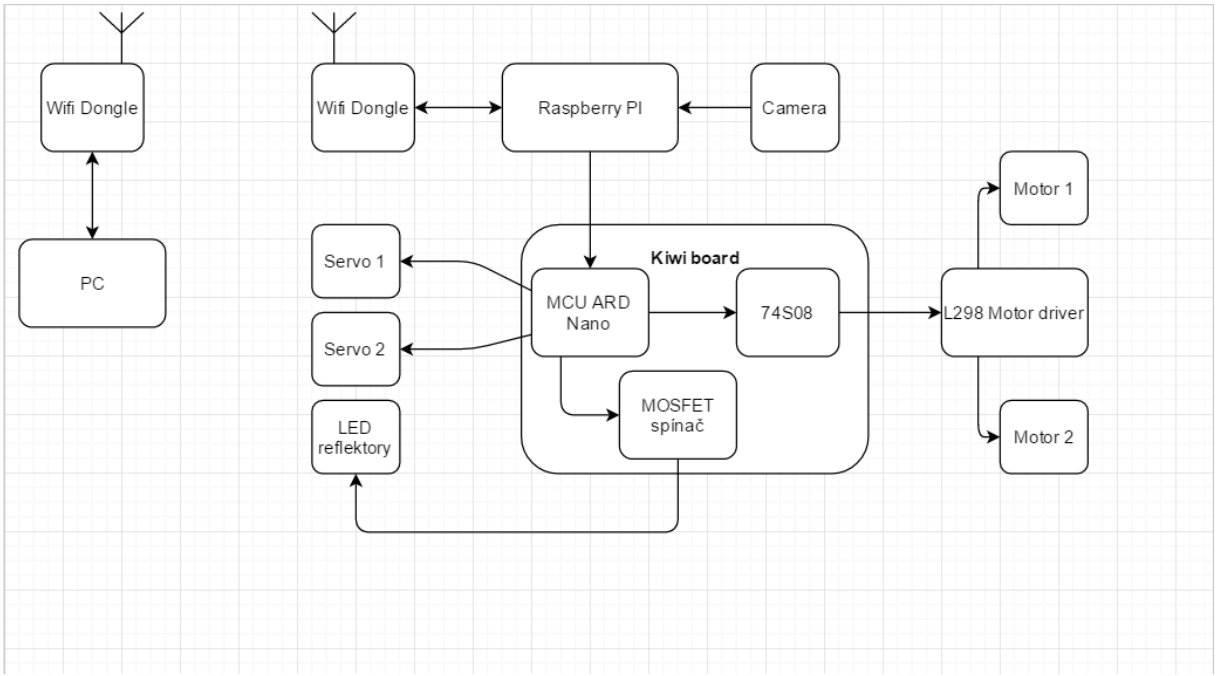
3.1 Co je PWM

PWM – Pulse Width Modulation (Pulzní šířková modulace) je přenos analogové veličiny pomocí střední hodnoty napětí obdélníkového periodického signálu s proměnnou šířkou aktivního impulsu. Vstupnní hodnotou je tzv. střída, která je definována jako poměr šířky aktivního impulsu k šířce celé periody. Střední hodnota takto přenášeného napětí je úměrná střídě, tedy vstupní analogové veličině. Standardně se střída udává v procentech. K získání střední hodnoty musíme průběh integrovat (např. dolní propust). Vzhledem k tomu, že stejnosměrné elektromotory mají sami o sobě integrační charakter, lze je napájet přímo napětím řízeným metodou PWM. To znamená, že otáčky takto řízeného stejnosměrného sériového elektromotoru budou úměrné střídě. V mém případě je PWM modulace řízena 8 bitovým slovem, tzn. že teoreticky dosažitelná přesnost je lepší než 0,5 %. Střídu lze nastavit v rozsahu 1/255 – 254/255, tedy 0,4% - 99,6%.



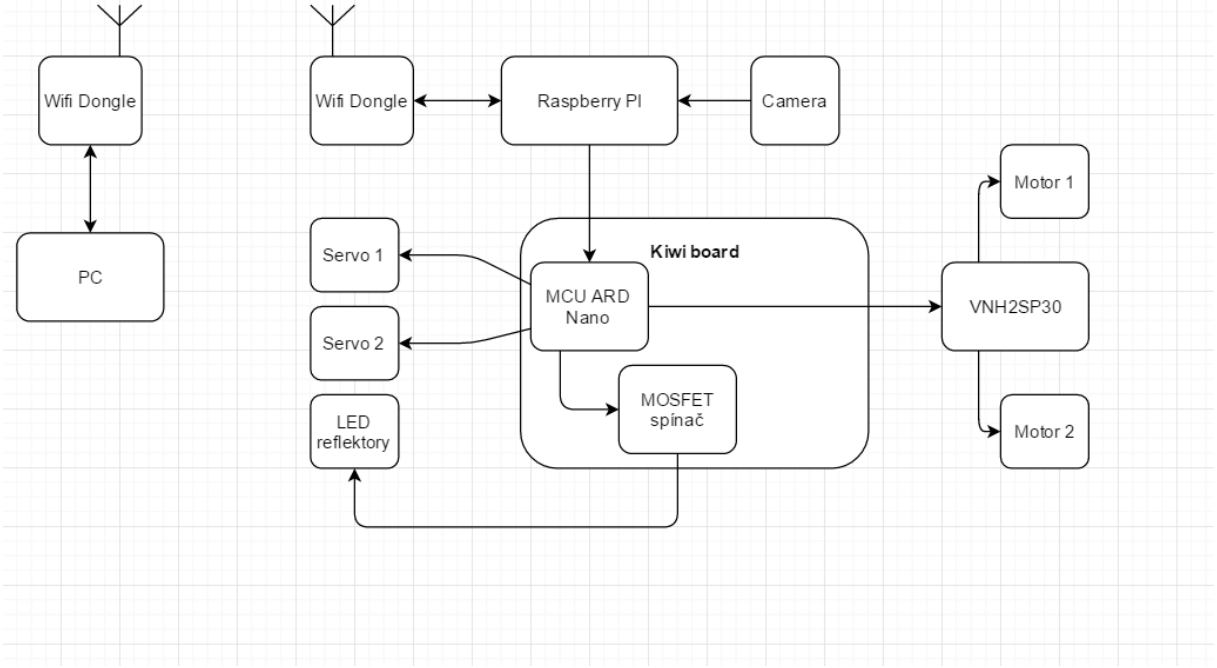
Obr. 10 PWM

3.2 Blokové schéma



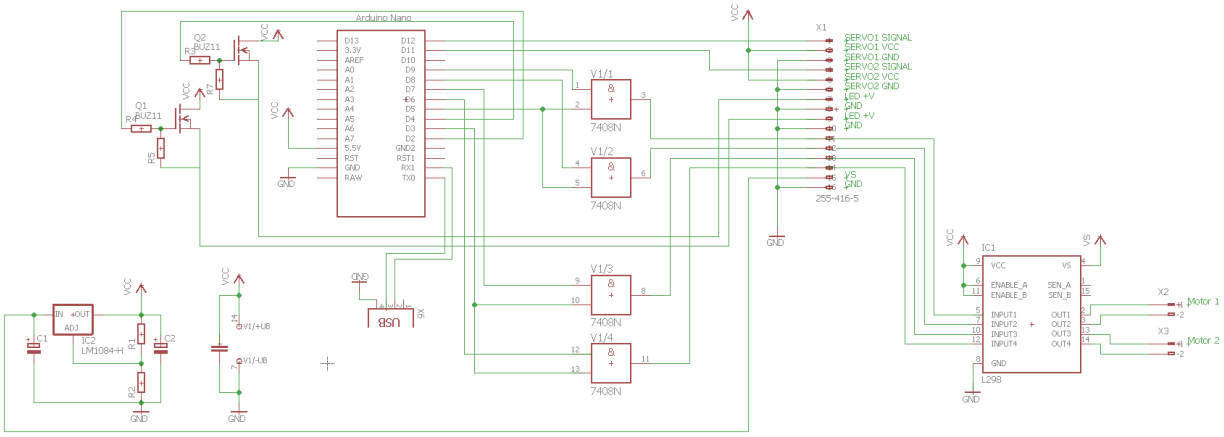
Obr.11 Původní bokové schéma s L298N

Schéma bylo ovšem třeba s příhlédnutím ke změně výkonového můstku inovovat. Podrobnější informace o důvodu změny můstku jsou popsány v kapitole 3.4 Jednotlivé komponenty a jejich funkce.



Obr. 12 Inovované blokové schéma s VNH2SP30

3.3 Schéma a jeho popis



Obr.13 Původní schéma

Celé schéma bylo navrženo v programu Eagle 7.7.0 Standart, určení jednotlivých součástek bylo provedeno dle datasheetů jednotlivých komponent – rezistory (R_7, R_5, R_4, a R_3) z datasheetu pro unipolární tranzistor BUZ11 (Q_1 a Q_2) a rezistory (R_1 a R_2) včetně kondenzátorů (C_1 a C_2) z datasheet stabilizátoru napětí LM1084IT.
Dvojice unipolárních tranzistorů **BUZ11** – tranzistory zde mají funkci spínačů. Jeden z tranzistorů slouží ke spínání LED reflektorů a ten druhý je prozatím připraven pro další možnou potřebu spínání dalších komponent.

Rezistory R_5 a R_7 jsou užity z důvodu zabránění nechtěného otevření tranzistorů, aby se tranzistor nemohl otevřit vlivem statické energie, či naindukovaného napětí. Zvolil jsem max. proudovou ztrátu 1mA a při 5V z Arduina je výpočet následující :

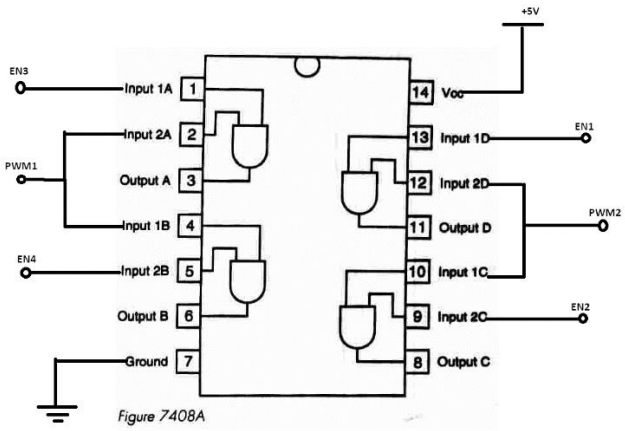
$$\frac{5\ V}{0.001\ A} = 5000\ \Omega$$

Zvoleny byly tedy rezistory nominální hodnoty 5k6 (5600Ω).
Rezistory R_5 a R_7 omezují proud, odebíraný z Arduina. Rezistory byly po stanovení podmínek (max úbytek 1 mV a max odebíraný proud 100 nA) zvoleny v nominální hodnotě 10k (10 000Ω).

$$\frac{1\ mV}{100\ nA} = 10\ 000\ \Omega$$

LM1084IT stabilizátor napětí, který může dodat proud až 5A, zajišťuje napájení všech komponent vyjma H – můstku. Dvojice rezistorů (R_1 a R_2) tvoří napěťový dělič, díky kterému vzniká na referenci takový úbytek napětí, který požadujeme vzhledem ke stabilizovanému napětí. Kondenzátory (C_1 a C_2) jsou zde z důvodu změny napětí zátěže, při změně zátěže by totiž napětí mohlo mírně kolísat. Kondenzátory vyrovnají případné zakolísání napětí a fungují jako lokální zdroje energie (v případě poklesu napětí se začnou vybíjet a tím nahradí případné ztráty).

74S08 logické hradlo (čtyřnásobný dvouvstupový AND), které jsem použil pro řízení směrů. Arduino Nano totiž neumožňuje stabilní použití čtyř výstupů v režimu PWM současně. Použil jsem tedy dva výstupy Arduina v režimu PWM a čtyři směr určující výstupy (Motor 1 dopředu, Motor 1 dozadu,Motor 2 dopředu, Motor 2 dozadu). Výstupy určující směr mohou nabývat dvou stavů 1/0. Stavy těchto výstupů jsou dány výstupními hodnotami z ovládacího Joysticku. Obvod má tedy na své vstupy zavedeny dvě PWM, které jsou připojeny na dva vstupy hradla současně (IN2A a IN1B, IN2D a IN1C) a čtyři výstupy určující směr, viz schéma (Obr. 14). Na vstupech pro napájení obvodu je ještě připojen kondenzátor C_3, který slouží jako lokální zdroj energie, kompenzující případné kolísání napětí v případě změn.

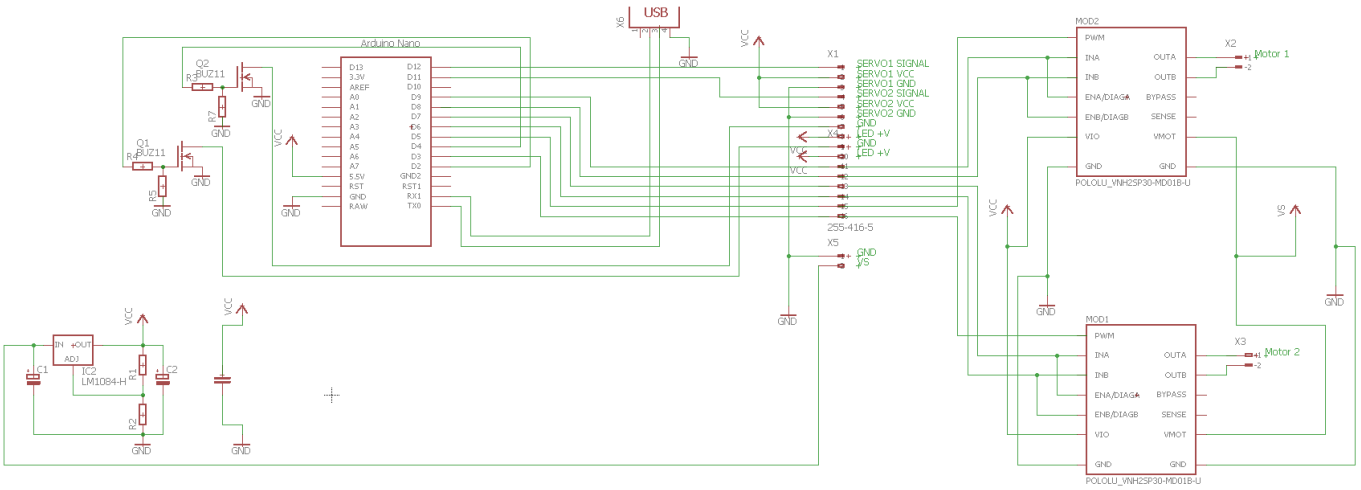


Obr. 14.Hradlo 7408

R1	121Ω
R2	365Ω
R3	10kΩ
R4	10kΩ
R5	5k6Ω
R7	5k6Ω
C1	10µF
C2	10µF
C	100nF
Q1	BUZ 11
Q2	BUZ 11
IC1	L298N
IC2	LM1084IT
V1	74S08
µC	ATmega 328 (Arduino Nano)

Tab. 1 Soupiska součástek

Při testování se vyskytl problém s H – můstkem L298N, nebyl totiž schopen dodávat požadovaný proud, byla tedy nutná změna této součástky za proudově silnější můstek VNH2SP30. Celá úprava a problém jsou popsány níže v kapitole 3.4 Jednotlivé komponenty a jejich funkce.



Obr. 15 Inovované schéma, přispůsobené můstku VNH2SP30

3.4 Jednotlivé komponenty a jejich funkce

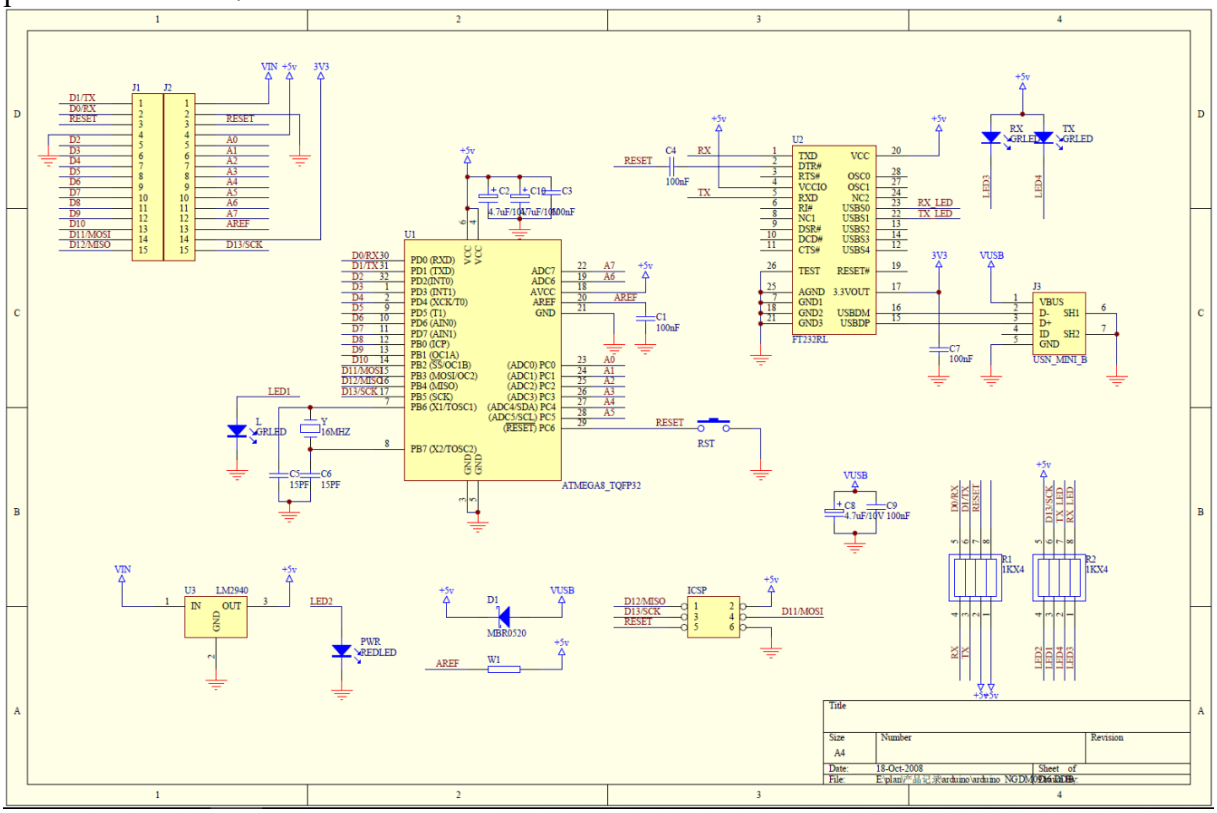
V robotovi jsou využity platformy **Arduino** a **Raspberry PI**, H-můstek(VNH2SP30), dvojice servomotorů v držáku kamery (pan tilt), USB webkamera, USB dongle a akumulátor. **Popis celé cesty:** Přes rozhraní v počítači jsou nastavovány jednotlivé řídící povely, které jsou následně odesílány jako jedna zpráva přes standard WIFI do Raspberry PI, které se chová jako server. Raspberry přijme tato data a přes USB sběrnici je odešle do Arduina. Arduino po přijetí dat rozdělí jednotlivé řídící povely dle příslušného identifikátoru, dle těchto řídících povelů Arduino vykoná požadované funkce tj. nastavení střídy PWM a nastavení řídích výstupů do požadovaných logických stavů. Dále nastavuje také stavy výstupů spojených s tranzistory BUZ11 a výchylku páky servomotorů. Určení výchylky páky servomotorů funguje též na principu PWM. Pro projekt je použito Arduino Nano (ver. ATMEGA 328).

Identifikátor	Číslo	Co řídí
'A'	0 – 255	Motor1 PWM
'B'	0 – 255	Motor2 PWM
'C'	1/0	Motor1 dopředu
'F'	1/0	Motor2 dopředu
'G'	1/0	Motor1 dozadu
'H'	1/0	Motor2 dozadu
'D'	0 – 180	Servomotor1 výchylka
'E'	0 – 180	Servomotor2 výchylka
'L'	1/0	LED reflektory

Tab.2 Protokol

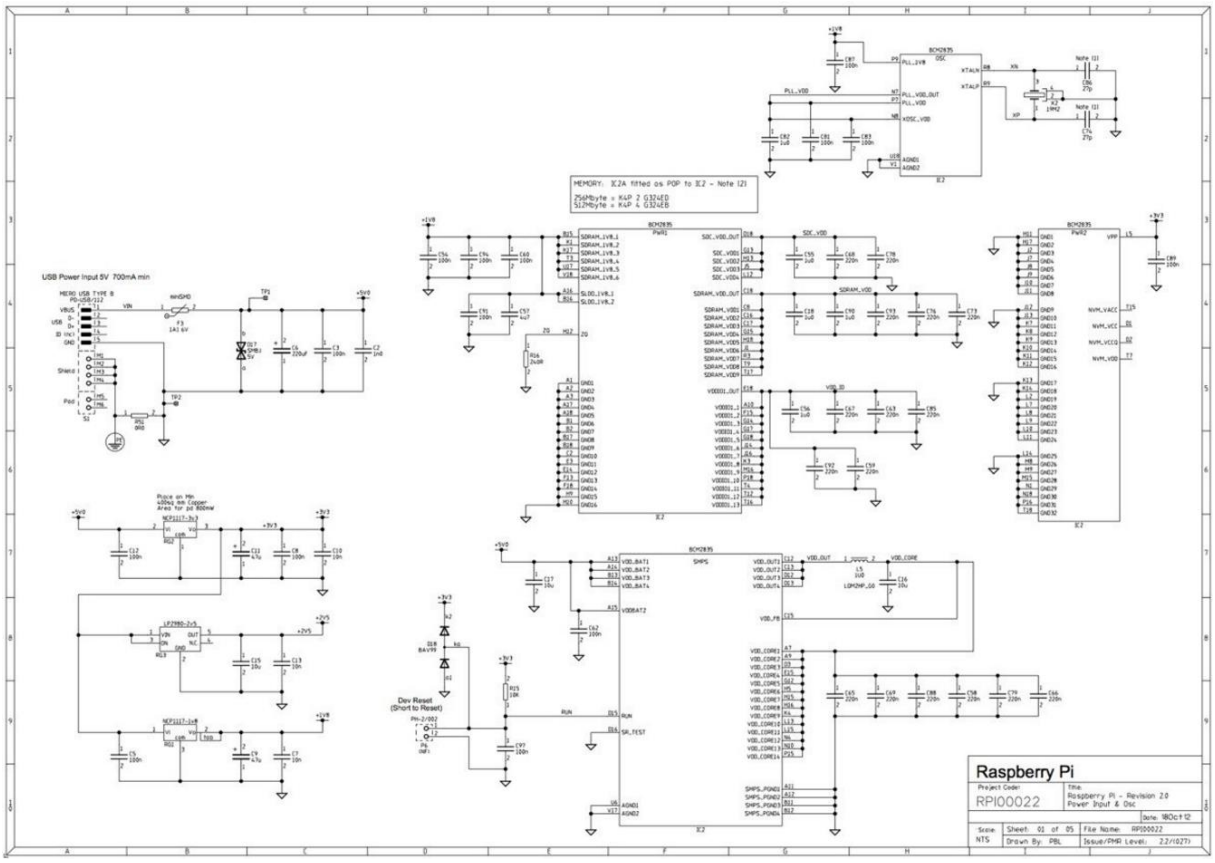
Arduino je řídicí částí pohonu platformy, přijímá datové zprávy z Raspberry, které dle identifikátoru rozdělí na jednotlivé funkce. USB kabel je modifikován přerušením +5V větve, aby Arduino neodebíralo proud z Raspberry PI, ale přímo ze stabilizátoru viz schéma.

Stručný popis platformy Arduino : obsahuje 8 bitový mikrokontrolér a další podpůrné obvody. Každé Arduino má většinu I/O pinů přístupných přes standardizované patice, do kterých se jednoduše připojují další obvody, kterým se ve světě Arduina říká Shieldy. Na desce je několik diod, resetovací tlačítko, konektory pro ICSP programování, napájecí konektor a obvod zprostředkovávající komunikaci po USB sběrnici. Digitální piny je také možné použít na softwarově řízený PWM výstup. Mikrokontrolér má již bootloader (kód, který se po spuštění postará o základní nastavení mikrokontroleru, jako jsou interní časovače, nastavení rozhraní USART a další). Díky tomu se uživatel nemusí starat o detaily a své programy píše v jazyce podobném C/C++.[1]



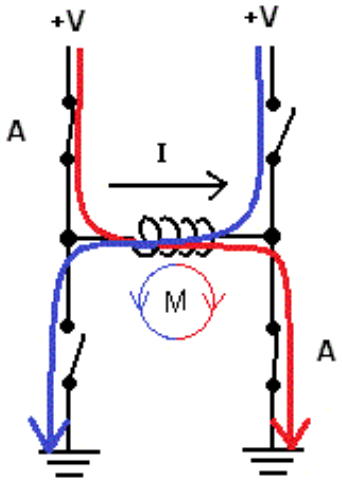
Obr. 16. Schéma Arduino NANO

Raspberry PI je v robotovi použito jako server pro kameru a jako UDP server, který přijímá data z počítače prostřednictvím rozhraní WIFI. Raspberry je stejně jako Arduino napájeno +5V a tudiž i k jeho napájení slouží stabilizátor viz blokové schéma. Raspberry je počítač velikosti platební karty, který má atributy klasického počítače (kromě periferií). Model B1, který byl pro projekt vybrán z důvodu nejnižší ceny, je osazen dvěma USB porty (možnost Hubu), cinch konektorem, HDMI konektorem, Ethernet portem, slotem pro SD kartu a mikro USB konektorem, dále se na něm nachází řada GPIO pinů. Do Raspberry jsou přes USB sběrnici připojena tři zařízení : WIFI dongle, kamera a Arduino.



Obr. 17. Schéma Raspberry PI model 1b

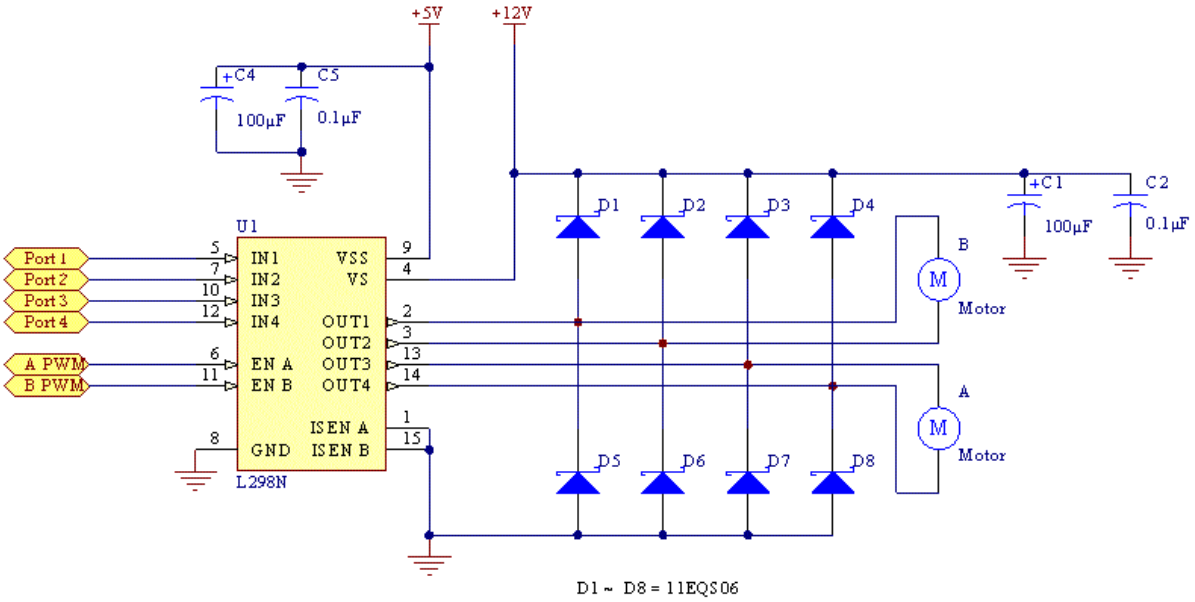
H-můstek(L298N) je použit pro nastavení směru rotace motorů. Tento obvod spíná dle signálu PWM dva stejnosměrné motory, lze jím také ovládat jeden motor krokový. Nejprve popíši vůbec samotný princip H – můstku, poslouží mi pro to ilustrace se čtyřmi spínači (Obr. 18).



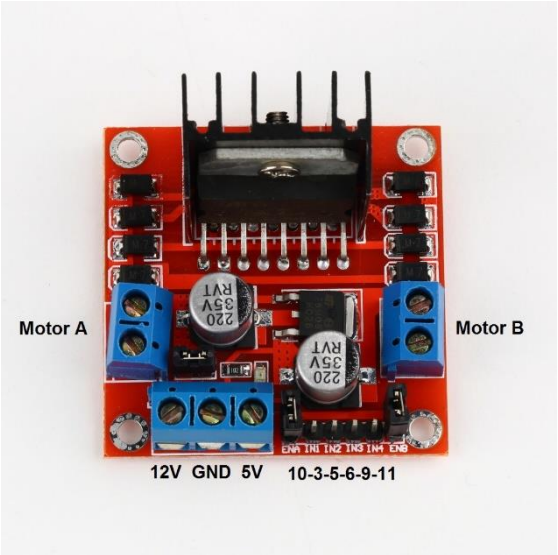
Obr. 18. Princip můstku

Na obrázku (Obr. 18) je znázorněna cívka (motor), kterou prochází proud. Směr proudu je určen spínači, které spínají jednu či druhou diagonálu. V můstku se vždy spíná diagonálně, protože pokud by se seply dva spínače pod sebou, došlo by ke zkratu. Jsou-li sepnuty spínače **A,** proud směřuje dle červené čáry (motor rotuje jedním směrem). Ovšem pokud by se seply protilehlé spínače proud by tekl opačným směrem ve směru modré šipky (motor rotuje reverzně), ještě je důležité dodat, že musíme dvojici, která spínala před tím rozepnout, jinak by také došlo ke zkratu. Vždy tedy spínáme jen jednu diagonálu můstku. Ve skutečném H – můstku řízeném elektrickými signály je to ovšem trochu jinak, vypínače jsou zastoupeny tranzistory, které jsou vybaveny další podpůrnou elektronikou.

Samotný obvod L298N má vstupy pro napájení a pro řízení, pro mé použití 4 řídící vstupy (viz. Blokové schéma obvodu). Ve skutečnosti má obovd 6 vstupů, zbylé dva jsou ENA a ENB, což jsou vstupy pro aktivaci můstků, a to můstku A (motor 1) a můstku B (motor 2). Na mnou zakoupené desce lze ovšem tyto vstupy (*ENA* a *ENB*) zapojit trvale do sepnutého stavu pomocí dvojice jumperů. V obvodu L298N jsou H- můstky dva, pro každý motor zvlášť. Mnou používané 4 vstupy (*IN 1 – 4*) pracují jako dvojice, tzn. dva vstupy na jeden můstek. Vstupy *IN1* a *IN2* jsou v podstatě výše zmíněné „diagonální dvojice“. Přivedením řídícího signálu na vstup *IN1* sepnu jednu dvojici tranzistorů a motor rotuje jedním směrem, při přivedení řídícího signálu na vstup *IN2* to platí přesně opačně. Vstupy *IN3* a *IN4* pracují obdobně. Řídicím signálem je výše popsaná pulzně šířková modulace, kde šířkou pulzu určuji otáčky hřídelů motorů.

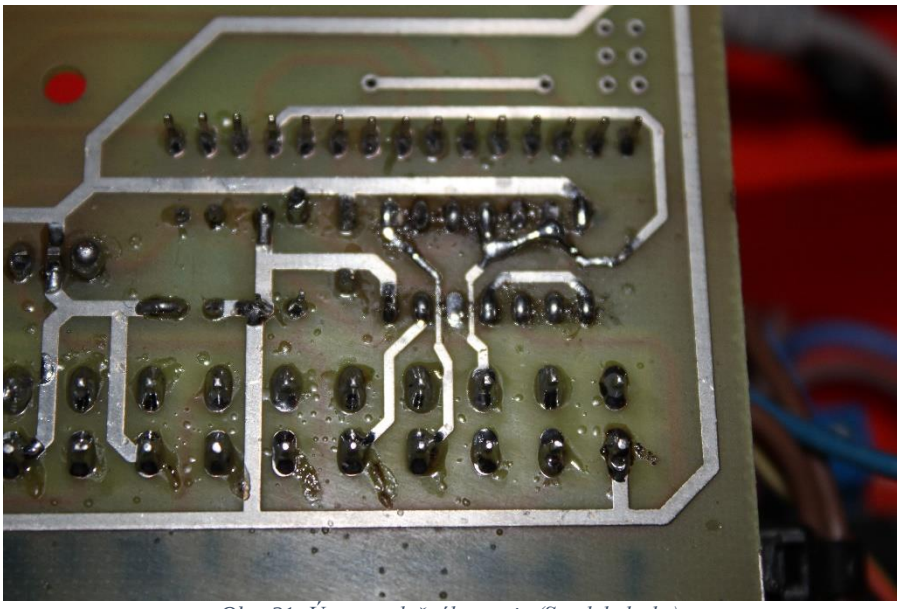


Obr 19. Blokové schéma obvodu s L298N

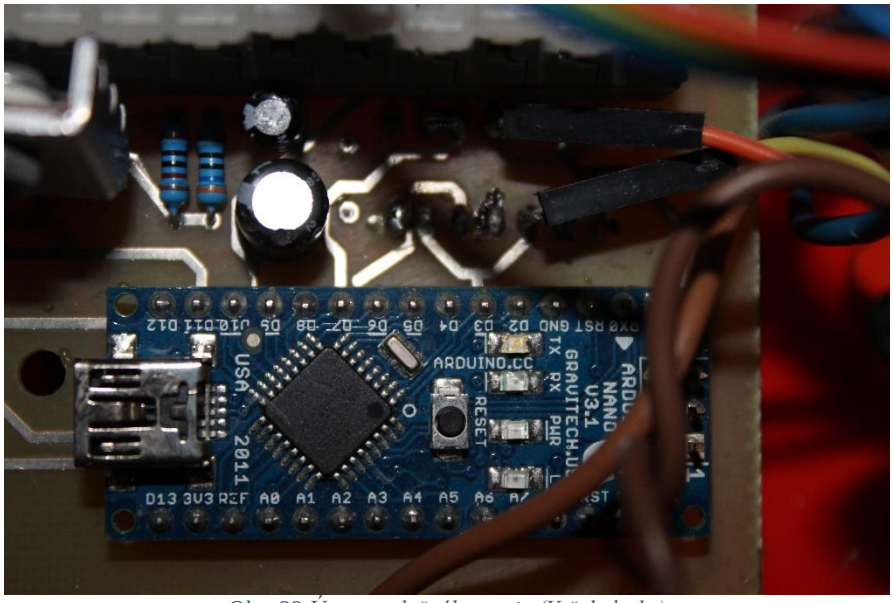


Obr 20. L298Nl

Při testování se bohužel vyskytl problém. Motory potřebovaly ke své činnosti daleko větší proud, než byl schopen H - můstek L298N dodávat, proto bylo třeba koupit nový a silnější H – můstek. Rozhodl jsem se pro **VNH2SP30**, který zvládne špičkově proud až 30A, tento obvod je sofistikovanější než původně zamýšlený (L298N). Obvod má 3 řídicí vstupy namísto 4, další vstupy jednotky nevyužívám. Řídicí vstupy jsou PWM signál a dva dvoustavové vstupy: VPŘED a VZAD (log 1, či log 0). Obvod zároveň umožňuje připojení šesti vodičů (3 vodiče na jeden kanál = PWM, VPŘED, VZAD) a hradlo 74S08 se tím pádem stalo zbytečným. Na plošném spoji jsem musel udělat několik změn, a to přemostit obvod 74S08 a vyvést výstupy PWM signálů (2 vodiče), celá úprava je znázorněna na obrázku č. 21.1 a 21.2.



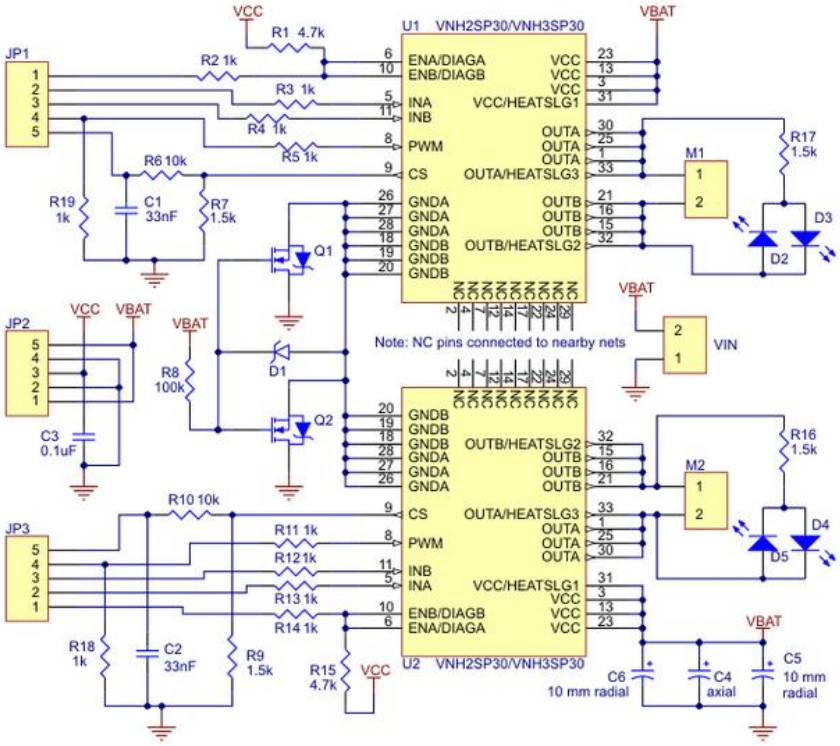
Obr. 21. Úprava plošného spoje (Spodek desky)



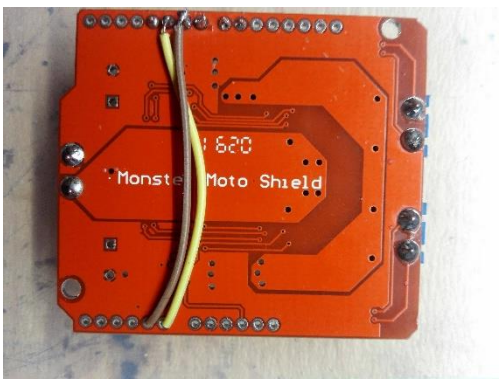
Obr. 22 Úprava plošného spoje (Vršek desky)

H-můstek(VNH2SP30)

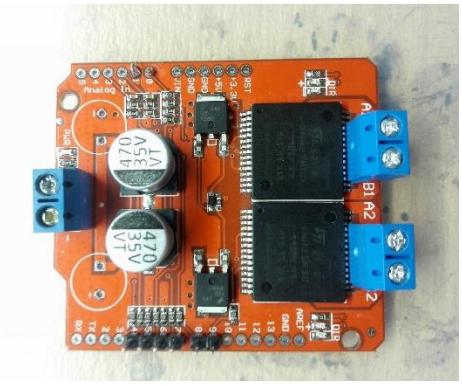
Řídí směr otáčení motorů. Obvod VNH2SP30 je ve výkonovém pouzdře MultiPowerSO-30 a jeho maximální proud je 30A špičkově a 15A kontiuálně, využívám jeho 3 řídící vstupy (PWM, INa, INb), aktivační vstupy mám připojeny ENa na vstupu IN PWM1 a ENb na vstupu IN PWM2. U obvodu byly nutné dvě malé úpravy výše zmíněné připojení aktivačních vstupů na vstupy PWM (Obr. 24) a montáž svorkovnice (Obr. 25), výstupy ck a current sense nepoužívám. Zakoupil jsem již součástku, která obsahuje dva tyto H – můstky a další podpůrné obvody (stabilizátor, ochrané diody...).



Obr. 23. Blokové schéma obvodu VNH2SP30

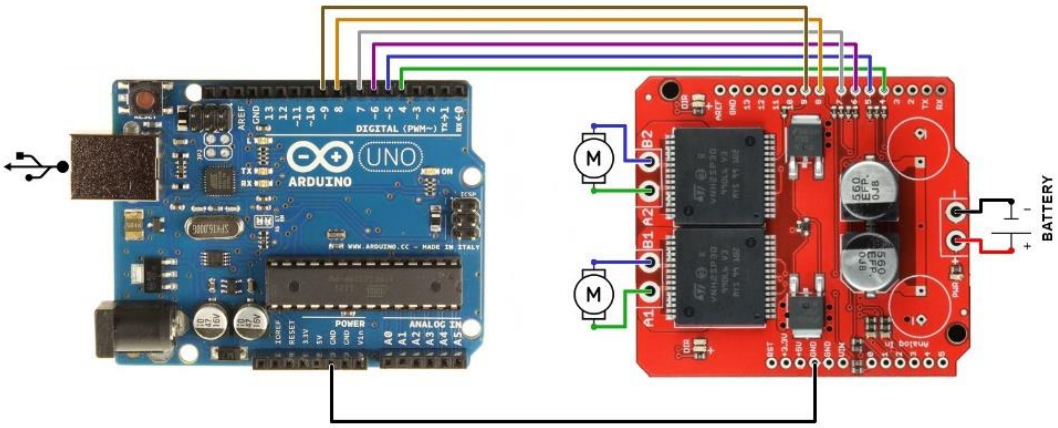


Obr. 24. ENA a ENB



Obr. 25. Svorkovnice

Můstek je s Arduinem propojen pomocí sedmi vodičů, detailnější popis zapojení vodičů mezi Arduinem a H - můstkem je na (Obr. 26). V ilustraci je znázorněno zapojení H - můstku s modelem Arduino Uno. Zapojení pro Arduino Nano zůstává stejné, liší se pouze označením výstupních pinů.



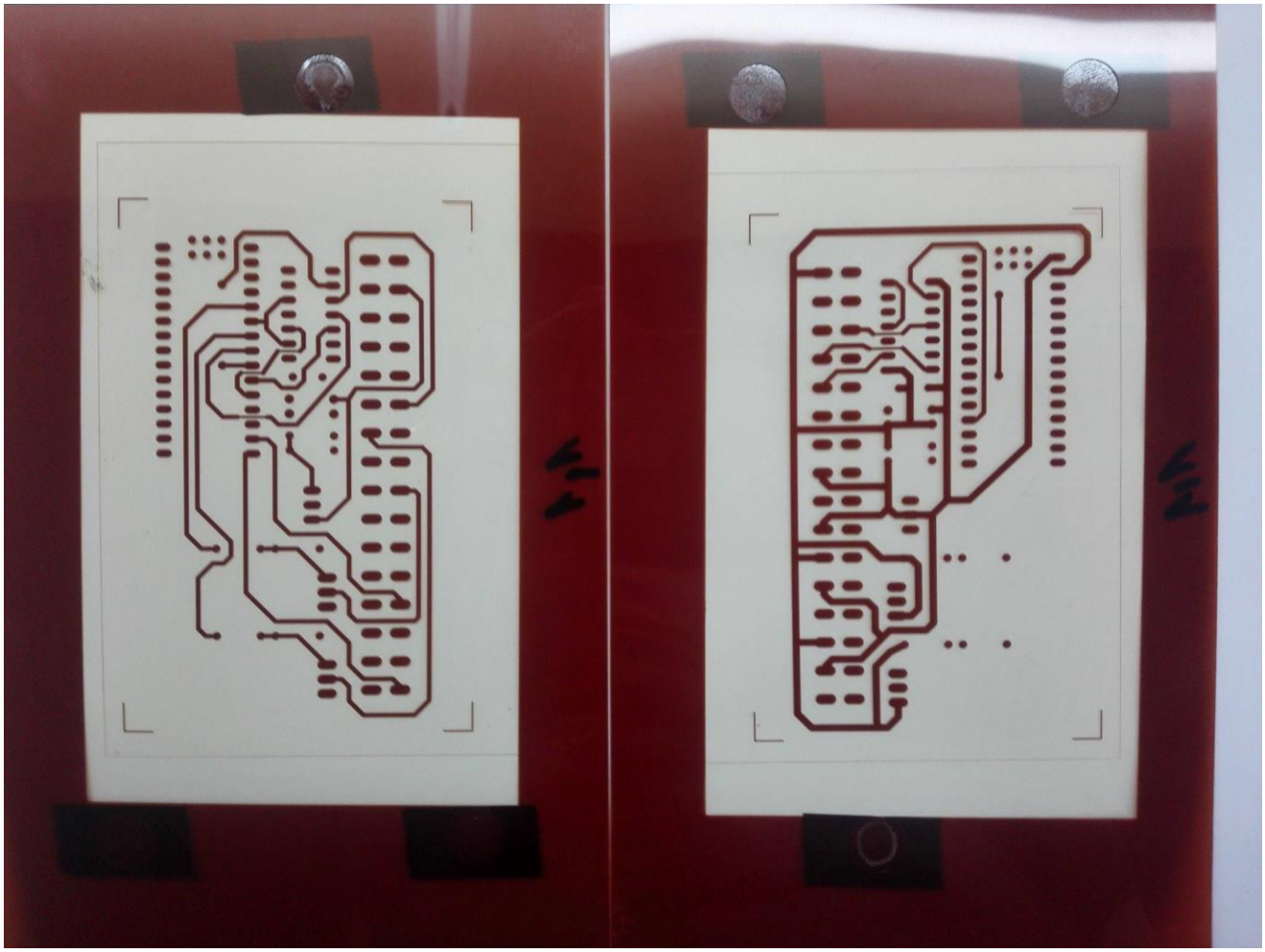
Obr. 26. Zapojení

3.5 Plošný spoj a jeho výroba

Plošný spoj, nebo také DPS, se využívá k upevnění a propojení součástek, namísto propojení pomocí propojovacích vodičů jsou komponenty propojeny pomocí vodivých cest vyleptaných na desce. Jednotlivé cesty vlastně nepředstavují nic jiného než jinou formu výše zmíněných propojovacích vodičů, ale pomocí nich to ve složitějších zapojeních nelze dělat. Propojovací vodiče totiž mají řadu problémů a nevýhod, ta první je zcela evidentní, je skoro nemožné vyznat se v zapojení, kde bude třeba sto a více kabílků, ano, existují serverovny a různé rozvodny, ale tam je to jiný případ. Další problém nastává v případě různých parazitních vlivů, je zde také vyšší možnost zkratů, norozdíl od elegantního řešení, kterým je DPS. Také jsem měl elektroniku popropojovanou pomocí asi třiceti kabílků a už i to byl problém, právě proto jsem se rozhodl pro DPS.

Nejprve jsem na internetu analyzoval informace o různých programech pro tvorbu schémat a DPS a jako nejvhodnější varianta se mi jevil program Eagle 7.7.0 Standart, který jsem využíval už výše při návrhu schémat, které jsem kreslil jako první, poté jsem pomocí zásuvného modulu pro DPS vymodeloval výsledné DPS. Po provedení všech úkonů jsem vygeneroval soubor, který jsem předal výrobci plošného spoje (výrobce plošných spojů), který mi vyrobil fotoprinty (Obr. 27) a následně i desku samotnou. Desku jsem nevyráběl samostatně z důvodu mého špatného vybavení, a také kvůli časové náročnosti. Poté jsem osadil desku součástkami dle jejich umístění a očistil lihem desku od přebytečné kalafuny. Následně jsem ještě desku pro kontrolu pomocí multimetru proměřil, abych měl jistotu spojitosti obvodu a vyvaroval se tak zkratů nebo rozpojenému spoji.

Rád bych ještě stručně popsal výrobu DPS v domácích podmínkách. Je tedy třeba mít Cuprextit (deska potažená spojitou vrstvičkou mědi), fotoprinty (nebo nesmazatelný popisovač v případě menších zapojení) a leptací lázeň (např. roztok $FeCl_3$ – chlorid železitý). V prvním kroku je třeba „zakrýt“ plochy na Cuprextitu, které chceme zachovat (vodivé cesty). V druhém krkou přemístíme takto poznačený Cuprextit do leptací lázně a zhruba tak deset minut (záleží na koncentraci roztoku) ponecháme leptat. Následně Cuprextit vyndáme z lázně a omyjeme vodou, poté vrtačkou či frézkou vyvrtáme do desky na příslušných místech otvory pro vložení „nožiček“ komponentů. Nakonec vše osadíme.



Obr. 27. Fotoprinty

4 PROGRAMOVÁNÍ A SÍŤ

Kapitola Programování a síť je rozdělena do pěti částí :

4.1 Programování platformy Arduino
4.2 Instalace systému na Raspberry PI
4.3 Programy pro Raspberry PI a jejich spouštění
4.4 Program počítačové rozhraní v jazyce C#
4.5 Nastavování sítě

4.1 Programování platformy Arduino

Arduino lze programovat pomocí prostředí přímo od výrobce, které se jmenuje Arduino (mnou používaná verze : 1.6.12), kde lze psát kód v jazyce podobném jazyku C, nebo přímo v Asembleru. Další známější rozhraní je Keil studio, s kterým ovšem nemám moc zkušeností. Mnou používaným se stalo prostředí Arduino a výše zmíněný jazyk podobný jazyku C. Velkou předností prostředí Arduino je velká škála různých knihoven, které jsou použity i v programu a jsou to knihovny pro obsluhu servomotorů a pro sériovou kominukaci. Stažení a instalace prostředí jsou velice jednoduché, prostředí lze stáhnout přímo ze stránek výrobce: https://www.arduino.cc/. Následně se přostředí nainstaluje. Po spuštění to vypadá asi takto (Obr. 27), následně je třeba v kolonce nástroje nastavit typ vývojové desky a její port (musí být připojena k počítači).

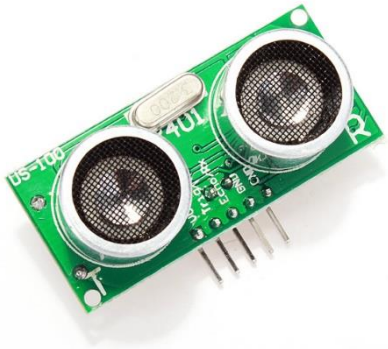
Obr. 28. Prostředí Arduino

Po zhotovení a odzkoušení programu jsem program nahrál do Arduina. Díky již zmíněné široké škále knihoven lze k Arduinu dopojit potřebné senzory, ať už výše zmíněný senzor čáry, nebo senzor vzdálenosti, pohybu, teploty či senzor detekce jedovatých plynů. Senzorů je mnoho, díky tomu lze na robota přidat kupříkladu senzor kvality vzduchu (Obr. 30) a zjišťovat, zda je v budově kvalitní vzduch a zda tam může vstoupit člověk bez ohrožení života. Senzorem PIR (Obr. 29) lze detekovat pohyb, robot tak může vykonávat třeba funkci autonomního hlídače, v kombinaci se senzorem zvuku (Obr. 31) a kamerou lze funkce hlídače ještě vylepšit. Ultrazvukový senzor (Obr. 32) může stejně tak jako senzor čáry sloužit k autonomní jízdě nebo k orientačnímu měření vzdálenosti.

Obr. 29. PIR senzor

Obr. 30. Senzor kvality vzduchu

Obr.31. Senzor zvuku (Analogový mikrofon)



Obr. 32. Ultrazvukový senzor vzdálenosti

4.2 Instalace systému na Raspberry PI

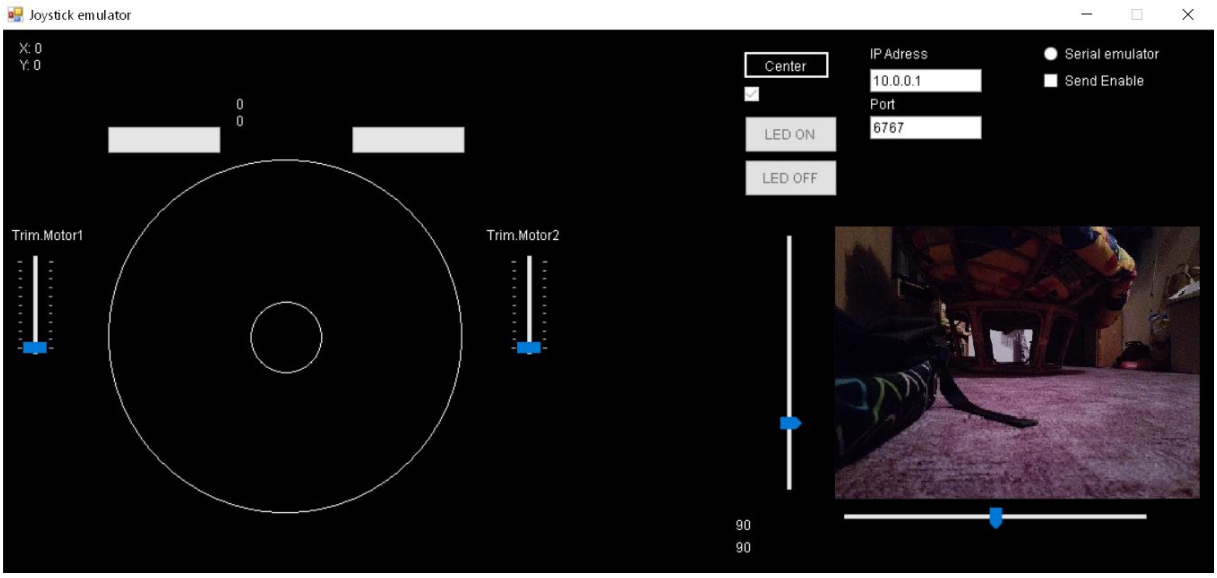
K Raspberry PI je bohužel zapotřebí mít paměťovou kartu, na které je nahrán systém a jeho soubory. Je k tomu také potřeba počítač, přes který se po stažení OS ze stránek výrobce: https://www.raspberrypi.org/, nainstaluje na SD kartu. Operační systém běží na bázi Linuxu (Debian), proto bývá také často označován jako Raspbian. Nakonec jsem SD kartu vložil do slotu počítače a spustil ho zapojením napájení. Raspberry si načetlo data z SD karty a následně se nastavilo.

4.3 Programy pro Raspberry PI a jejich spouštění

Program pro Raspberry je napsán v programovacím jazyce Python, Raspberry dále umí spouštět programy v Javě, C a C++. Program obstarává funkci serveru, který poslouchá na určeném portu. Pro posílání dat do Raspberry využívám UDP protokolu.
UDP protokol je protokol transportní vrstvy orientovaný na zprávy a je zdokumentovaný v IETF RFC. V sadě protokolů Internetu poskytuje UDP velmi jednoduché rozhraní mezi síťovou vrstvou pod a aplikační vrstvou nad. UDP neposkytuje žádné záruky doručení a odesilatelova UDP vrstva si u jednou už odeslaných zpráv neudržuje žádný stav. UDP pouze přidává kontrolní součty a schopnost roztřiďovat UDP pakety mezi více aplikací běžících na stejném počítači. UDP hlavička se skládá jen ze 4 políček, z nichž 2 jsou volitelná. Políčka zdrojového a cílového portu jsou šestnácbitová a identifikují odesílající a přijímající proces. Protože UDP je bezestavový a odesilatel nemusí vyžadovat odpověď, zdrojový port je volitelný. Pokud není použit, zdrojový port by měl být nastaven na nulu. Po číslech portů následuje povinná délka UDP paketu včetně dat v bytech. Minimální hodnota činí 8 bajtů. Zbývající políčko hlavičky tvoří šestnáctibitový kontrolní součet pokrývající hlavičku i data. Tento součet je možné vynechat, ale v praxi se téměř vždy používá.[2]
Program okamžitě po přijetí dat tato data přepošle po USB sběrnici do Arduina. Program server je v Raspberry spouštěn při startu samotného počítače díky funkci Cron, která spoušítí definované programy při bootování samotného počítače. Kamera je obsluhována programem *fswebCam*, který streamuje z kamery a stará se o správnou funkci serveru, na který data (video) zobrazuje, jedná se tedy o software třetí strany, který nebyl modifikován.

4.4 Programování počítačového rozhraní v jazyce C#

Robot je hotový, je tedy funkční hardware i software v Arduinu a RPI, ale jak robota ovládat ? K tomuto účelu jsem se rozhodl naprogramovat ovládací rozhraní v jazyce C#. V kterém bych mohl nastavovat rychlost a směr robota, nejspíše pomocí joysticku myší, nastavovat výchylku servomotorů kamery, rozsvěcet/zhasínat reflektory a vidět obraz z webkamery. Rozhodl jsem se tedy pro tvorbu okenní aplikace (Obr. 32). Jazyk C# jsem zvolil, protože ho znám ze školy a umím v něm vcelku obstojně programovat, také jsem měl již nainstalované Visual studio, které nabízí řadu komponent (text box, track bar…).



Obr. 33. Rozhraní

Počítač posílá data do robota za využití standardu wifi a protokolu UDP. Díky UDP protokolu lze komunikovat s vyšší datovou latencí, ovšem bez zpětné kontroly dat, jako je tomu u TCP/IP protokolu. Data nepotřebuji zpětně kontrolovat, protože i kdyby náhodou jedna datová zpráva nedošla v pořádku, tak v zápětí příjde další. Problém ovšem nastal, když jsem chtěl, aby bylo možné ovládat oba dva motory současně jedním joystickem. Joystick je vytvořen pomocí knihovny **Graphics** (Ohraničení a kruhový kurzor). Při vychýlení kurzoru se mění výstup joysticku (souřadnice). Souřadnice jsou posunuty tak, aby byl střed ohraničené částu počátkem.

Joystick v rozhraní je reprezentován kruhovým ohraničením (větší z kružnic), v kterém se pohybuje s kruhovým kurzorem, jehož výstupem jsou souřadnice [x,y], jak již bylo výše zmíněno. Kruhové ohraničení má průměr 254 pixelů a poloměr tedy 127 pixelů. Střed ohraničení má souřadnice [0,0]. Po vychýlení kruhového kurzoru se odečtou souřadnice [x,y], které jsou využity pro konečný výpočet hodnot, sloužících k nastavení střídy PWM. Ohraničení je dále rozděleno na dvě poloviny dle hodnot na **ose x**, tedy **pro kladnou polovinu osy x (0 – 127)** platí vztahy :

$$Střída\ \textbf{pravého}\ \text{motoru}\ = \left(\frac{y}{\sqrt{(x^2+y^2)}}\right) \times y$$

$$Střída\ \boldsymbol{levého}\ motoru = y$$

Pro zápornou polovinu (-127 – 0) platí vztahy opačně :

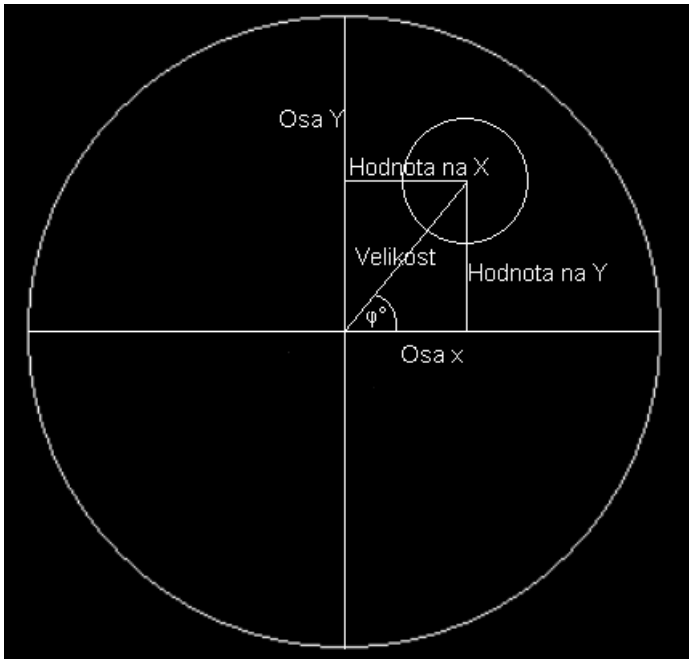
$$Střída\ \boldsymbol{pravého}\ motoru = y$$

$$Střída\ \textbf{levého}\ \text{motoru}\ = \left(\frac{y}{\sqrt{(x^2+y^2)}}\right) \times y$$

V případě, že by hodnota na ose y byla nulová, tak dle polohy (pravá/levá strana) na ose x přiřazuji příslušnému motoru přímo hodnotu odečtenou na ose x, točí se tedy jen jeden motor (plný zatáčecí rejd). Pro výpočty záporné hodnoty nepoužívám, pouze nastavuji 4 směr určující příznaky: C, F, G, H do stavů log. 1/0. Příznaky C a F určují směr otáčení **pravého motoru (doředu/dozadu)**, příznaky G a H určují směr otáčení **levého motoru (doředu/dozadu)**. Tyto příznaky se nastavují dle souřadnic na osy y, následujícím způsobem: **pro kladnou polovinu osy y** (0 – 127) nastavuji příznaky C a G do stavu log. 1 a příznaky F a H do stavu log. 0. **Pro zápornou polovinu osy y** (-127 – 0) nastavuji příznaky F a H do stavu log. 1 a příznaky C a G do stavu log. 0. Příznaky C a F tedy určují zda robot pojede vpřed a příznaky F a H zda pojede vzad.

Původní řešení výpočtu PWM hodnoty pro zpomalovaný motor spočívalo v pouhém odečítání souřadnice na osy x od souřadnice na osy y, toto řešení mělo ovšem jednu slabinu, když byl kruhový kurzor vychýlen, tak že hodnoty na ose x byly větší, než ty na ose y, byla vypočtená hodnota záporná. Původní řešení mě tak omezovalo v používání celé kružnice, výpočet fungoval pouze v jakési kruhové výseči, tedy pro hodnoty, kde byla výchylka na ose y větší než ta na ose x.

Nové řešení spočívá ve využití goniometrické funkce sinus. Funkce sinus totiž nabývá v rozsahu 0° - 90° číselných hodnot 0 – 1, touto číselnou hodnotou se násobí souřadnice z osy y, a výsledek je pak použit, jako hodnota pro nastavení střídy PWM zpomalovaného motoru, druhému motoru nastavuji přímo hodnotu odečtenou na ose y. Tento výpočet provádím ve všech kvadrantech zvlášť, protože řídíme dva motory pro motor 1 máme tedy druhý a třetí kvadrant kruhového ohraničení a pro motor 2 první a čtvrtý kvadrant.



Obr. 34 Kružnice

Ukázka výpočtů při vychýlení kurzoru joysticku k pravému hornímu rohu ohraničení, kde je číselná hodnota z osy y = 100 a úhel = 45° tzn. 100 * sin (45°) = 70,71 hodnota se tedy zmenší a jeden z motorů se bude točit pomaleji (využívám proměnou typu integer, hodnota se tedy převede na celé číslo), druhému z motorů se nastaví přímo hodnota 100. Hodnoty pro motory budou následující: Motor1 = 70, Motor2 = 100.

V případě, že bude joystick vychýlen přímo nahoru budou sořadnice nasledující: y = 127, x = 0, výpočet je tedy takovýto :

$$Hodnota\ pro\ zpomalovaný\ motor = \left(\frac{127}{\sqrt{\left(0^2 + 127^2\right)}}\right) \times 127 = \mathbf{127}$$

Druhému motoru se přiřazuje přímo hodnota odečtená na ose y, a tak budou hodnoty pro motory následující: Motor1 = 127, Motor2 = 127, oba se tedy budou točit stejně rychle.

4.5 Nastavování sítě

Protože počítač komunikuje s Raspberry za použití standardu wifi, bylo třeba použít nějaký přístupový bod (AP). Byl použit starý router, s funkcí bezdrátového AP. Tomuto routeru bylo nastaveno heslo sítě, šifrování a zablokování IP adres pro Raspberry a počítač (aby nedošlo k jejich obsazení) a byl odemčen port pro UDP socket (8080). Dále jsem k síti připojil Raspberry PI a počítač.

5 ZÁVĚR

I když se vyskytlo mnoho problémů, robota se nakonec povedlo uspěšně uvést do funkčního stavu, robot je tedy ovladatelný na dálku a lze přijímat signál z jeho kamery. Díky robotovi jsem si prohloubil své znalosti elektrotechniky a programování. K robotovi jsem také vytvořil webovou stránku https://robotkiwiblog.wordpress.com/ , na které lze nalézt podrobné postupy a tutorialy , stránka by měla pomoct začínajícím robotikům. Do budoucna bych chtěl vyzkoušet přidání vybraných modulů a chtěl bych se, tak pokusit alespoň o částečnou autonomii. Dále bych chtěl nakonfigrovat robota pro ovládání ne pouze z místní sítě. Původní myšlenka stavby robota vznikla ve školním týmu LSD, kde jsme chtěli s robotem provádět o dnech otevřených dveří, abychom tak ukázali naše schopnosti a zatraktivnili školu pro budoucí zájemce o studium. Kvůli nedostatku času jsem, ale tým opustil. Po nějakém čase jsem se dostal k brigádě, kde se naplno zabýváme Průmyslem 4.0 a automatizací a napadlo mě tedy koncipovat robota, jako univerzální podvozek (modul), který by mohl realizovat různé funkce. Pro mě nejzajímavější funkcí by byla asi funkce hlídače, kdy se do terénu posílá robot namísto člověka, z důvodu lidského bezpečí. Dále se mi také velice líbila funkce manipulátoru ve skladu, kterou jsem bohužel nestihl plně realizovat.
Robota by tedy bylo možné použít jako :

- Dálkově ovládaného průvodce na dnech otevřených dveří (přidáním mikrofonu a reproduktoru)
- Dálkově řízený manipulátor (vozící např. krabici se součástkami)
- Průzkumné vozítko
- + Cena robota nepřesahuje 7000 korun

6 SEZNAM ZKRATEK

WIFI	Wireless fidelity – komunikační bezdrátový standard
GPIO	General purpose input/output – vstupy či výstupy
SD	Secure digital – paměťová karta
DPS	Deska plošných spojů
IZO	Typ náhledu v prostředí VariCAD
ABS	Akrylonitrilbutadienstyren - amorfní termoplatický kopolymer
PWM	Pulzně šířková modulace
USB	Universal serial bus
USART	Universal Synchronous Asynchronous Receiver Transmitter
UDP	User datagram protocol – komunikační protokol bez kontroly dat
TCP/IP	Transmission control protocol – komunikační protokol s kontrolou dat
HDMI	High-definition multimedia innterface – multimediální interface

7 ZDROJE

[1] Arduino: Platforma. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 22.1.2017]. Dostupné z: https://cs.wikipedia.org/wiki/Arduino

[2] User Datagram Protocol: Technický přehled. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 22.1.2017]. Dostupné z: https://cs.wikipedia.org/wiki/User_Datagram_Protocol

[3] Datasheet BUZ11. ALLDATASHEET.COM. [cit. 5.2.2017]. Dostupný na WWW: http://pdf1.alldatasheet.com/datasheetpdf/view/22129/STMICROELECTRONICS/BUZ11.html

[4] Datasheet LM1084IT. ALLDATASHEET.COM. [cit. 5.2.2017]. Dostupný na WWW: http://html.alldatasheet.com/html-pdf/8602/NSC/LM1084IT-5.0/43/1/LM1084IT-5.0.html

8 OBRÁZKY

[1] Senzor Čáry, DX [online]. [cit. 4.12.2016]. Dostupný na WWW: http://img.dxcdn.com/productimages/sku_428464_3.jpg

[2] První návrh

[3] Hotový model

[4] Al profil, Kování schránky [online]. [cit. 4.12.2016]. Dostupný na WWW: https://www.kovani-schranky.cz/fotky39391/fotos/_vyrn_2494ob_236_big.jpeg

[5] Motory, MojeRC.cz [online]. [cit. 4.12.2016]. Dostupný na WWW: http://www.mojerc.cz/images/stories/virtuemart/product/tankmotor.jpg

[6] Kolo, Sportisimo [online]. [cit. 4.12.2016]. Dostupný na WWW: https://www.sportisimo.cz/arcore/hyperspeed-8a/102491/?arcore=61

[7] Elektro krabice

[8] Ukázka souboru s příponou .Gcode

[9] Platforma (robot)
[10] PWM, TIMOTHY HIRZEL. Arduino [online]. [cit. 14.12.2016]. Dostupný na WWW: https://www.arduino.cc/en/Tutorial/PWM
[11] Původní bokové schéma s L298N
[12] Inovované blokové schéma s VNH2SP30
[13] Původní schéma
[14] Hradlo 7408, WebTronics™ [online]. [cit. 14.12.2016]. Dostupný na WWW: http://srstansfield.com/electrical-engineering/hyperlinks/logic-ttl-chips/7408.htm
[15] Inovované schéma, přispůsobené můstku VNH2SP30
[16] Schéma Arduino NANO, Dfrobot [online]. [cit. 13.1.2017]. Dostupný na WWW: http://www.dfrobot.com.cn/image/data/DFR0010/Arduino_Nano_Schematic.png
[17] Schéma Raspberry PI model 1b, Adafruit [online]. [cit. 13.1.2017]. Dostupný na WWW: https://blog.adafruit.com/wp-content/uploads/2012/10/adafruit_628.jpg
[18] Princip můstku, SERVO-DRIVE s.r.o. Speciální krokové motory na míru [online]. [cit. 2014-8-16]. Dostupný na www: http://www.servo-drive.com/specialni_krokove_motory_krokove_motory_na_miru.php
[19] Blokové schéma obvodu s L298N, Fritzing [online]. [cit. 2.1.2017]. Dostupný na WWW: http://fritzing.org/projects/working-with-l298n-dc-motor-driver
[20] L298N, Fritzing [online]. [cit. 2.1.2017]. Dostupný na WWW: http://fritzing.org/projects/working-with-l298n-dc-motor-driver
[21] Úprava plošného spoje (Spodek desky)
[22] Úprava plošného spoje (Vršek desky)
[23] Blokové schéma obvodu VNH2SP30, Pololu [online]. [cit. 22.1.2017]. Dostupný na WWW:https://a.pololufiles.com/picture/0J411.1200.png?78fc551ac6ef5b54b366dcd33af0fe0
[24] ENA a ENB
[25] Svorkovnice
[26] Zapojení, Logicware [online]. [cit. 2.1.2017]. Dostupný na WWW: http://www.logicware.com.au/monster-moto-shield-vnh2sp30-stepper-motor-driver
[27] Fotoprinty
[28] Prostředí Arduino
[29] PIR senzor, Robot store [online]. [cit. 2.1.2017]. Dostupný na WWW: http://robotstore.cz/obchod/arduino/pir-senzor-detektor-pohybu-arduino-modul-2/
[30] Senzor kvality vzduchu, Robot store [online]. [cit. 2.1.2017]. Dostupný na WWW: http://robotstore.cz/obchod/arduino/mq-135-sno2-senzor-kvality-vzduchu-arduino-modul/
[31] Senzor zvuku (Analogový mikrofon), Robot store [online]. [cit. 2.1.2017]. Dostupný na WWW: http://robotstore.cz/obchod/arduino/analogovy-mikrofon-detekce-zvuku-arduino-modul/
[32] Ultrazvukový senzor vzdálenosti, Robot store [online]. [cit. 2.1.2017]. Dostupný na WWW: http://robotstore.cz/obchod/arduino/us-100-ultrazvukovy-arduino-modul-mereni-vzdalenosti/
[33] Rozhraní
[34] Kružnice

9 PŘÍLOHY

- Program pro mikrokontrolér Arduino : Kiwi kod.ino
- Pragram pro počítač Raspberry PI : server.py
- Počítačové rozhraní : WpfJoistick.exe (C#)
- Výkresová dokumentace podvozku : ROBOT – Výkresy (soubory .dwb)