Úvod

Pre tento projekt sme sa rozhodli preto, že nás už dávno zaujímali rádiovo ovládané vozidlá, lenže tieto nemajú dosť veľký praktický potenciál, keďže je potrebné byť blízko vozidla a treba na nich stále vidieť.

Dnešné rádiovo ovládané vozidlá sú stále vytvárané na rovnakom princípe, takže je efektívne vytvoriť niečo unikátne a pospájať viac technológií do jedného vozidla. Našou hlavnou prioritou bolo spraviť pripojenie pomocou internetu na rozdiel od infračerveného, bluetooth, rádiového alebo podobného pripojenia s relatívne krátkym dosahom.

Internet je technológia s veľkým potenciálom. Predpoklad je, že aj pokiaľ bude niekedy zastaraný, nahradí sa niečím kompatibilným a náš projekt bude rovnako použiteľný s malými zmenami.

Keďže dnešné rádiovo ovládané vozidlá sa napriek popularizácii takého nápadu v rôznych filmoch prakticky vôbec nevyrábajú s kamerou podporujúcou sledovanie v reálnom čase, pričom pripojenie na veľkú vzdialenosť vyžaduje možnosť vidieť pred vozidlo, našou druhou prioritou je, aby toto vozidlo kameru malo. Neskôr sme plánovali pridať aj ďalšie moduly ako napríklad mikrofón, reproduktor, teplomer a snímač vzdialenosti od prekážky.

Naša vďaka za informačnú, praktickú aj morálnu pomoc patrí hlavne našim učiteľom, ktorí nám s realizáciou projektu priamo pomohli, ale aj našim ostatným spolužiakom. Za čas a financie potrebné k realizácii tohto projektu ďakujeme škole SOŠ Hviezdoslavova 5. Okrem toho ďakujeme aj všetkým autorom voľných programov a knižníc, ktoré sme využili v našom projekte a ich snahu podporujeme vydaním nášho projektu pod GNU GPL v3+ licenciou, pretože vo svete bez voľných diel by sme vôbec nemali podmienky k vytvoreniu tohto projektu.

1 Diaľkovo ovládané vozidlá

Skratka RC je od anglických slov Radio Control [Rádiovo Ovládané]. Bežne sa ľudia stretnú s hračkárskymi RC vozidlami dostupnými v obchodoch, no všeobecne má ich využitie omnoho väčší potenciál. Zatiaľ sa využívajú zväčša vo firmách, kde je potrebné zautomatizovať, respektíve uľahčiť určité činnosti alebo na projekty, kde nie je bezpečné, alebo kvôli veľkosti vozidla možné, aby vozidlá zvnútra riadil človek.

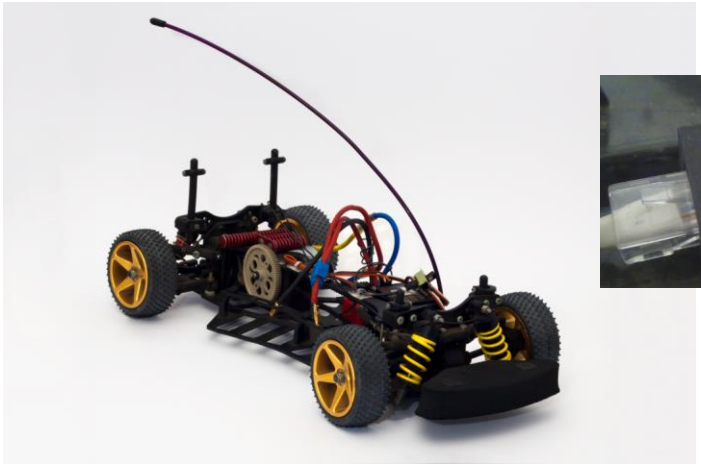
Napriek tomu, že IEEE 802.11 (WiFi) je rádiový bezdrôtový štandard, bolo by príliš zjednodušené, keby sme sa limitovali na rádiovo ovládané vozidlá. Technológie nemajú limit, ale prinajmenšom stále platí, že toto vozidlo sa môže ovládať aj priamo cez IEEE 802.3 (Ethernet), t.j. káblom za použitia úplne tých istých metód prenosu dát. Stále sú ale aj iné metódy bezdrôtového prenosu, napríklad infračerveným svetlom alebo laserom. Tieto síce nie sú v dnešnej dobe praktické, no v budúcnosti sa môžu objaviť nové metódy prenosu signálu, ktoré zatiaľ nepoznáme a vôbec nebudú používať rádiový prenos signálu.

Obrázok 1: Oficiálne logo WiFi štandardu (Public Domain)

Rádiové frekvencie sa merajú v Hertzoch. Hračkárske vozidlá majú zväčša frekvencie okolo 2.4 GHz. IEEE 802.11 (WiFi) štandardy majú frekvencie:

802.11b, 802.11g, 802.11n-2.4 → 2.4 – 2.5 GHz

802.11a, 802.11n → 4.915 – 5.825 GHz



Obrázok 2: Obyčajné RC (rádiovo ovládané) vozidlo

Obrázok 3: WiFi router mini použitý v našom projekte

1.1 Využitie, výhody a nevýhody

Takéto vozidlo na diaľkové ovládanie môže mať rôzne využitie, napríklad:

- **Zábava** – preteky medzi hráčmi alebo preskúmavanie neznámeho územia.
- **Bezpečnosť** – sledovanie miestnosti alebo domu pomocou kamery umožňujúcej sledovanie v reálnom čase s možnosťou zmeny polohy sledovania.
- **Výučba** – použitie ako učebná pomôcka na trénovanie programovania, stavby vozidla (konštrukcie a hardwaru) a praktické znázornenie jeho funkcií.
- **Práca** – prenášanie predmetov alebo priamo vykonávanie určitých činností buď vo firme, ktorá má pre vozidlo určitý cieľ, alebo viacúčelovo v domácnosti.
- **Armáda** – infiltrácia a špionáž určitej oblasti alebo obrana, vysokorizikové činnosti, kde nie je vhodné riskovať ľudské životy.

Ďalším príkladom je kombinácia zábavy a komerčného projektu – prenajímanie vozidiel na ovládanie cez internet do vopred pripraveného prostredia – preteky alebo preskúmavanie v reálnom svete na rozdiel od virtuálnej hry.

Výhodami sú vysoká bezpečnosť ľudí obsluhujúcich vozidlo a ich komfort pri práci (ovládanie z domu alebo služobnej kancelárie). Oproti autonómnym vozidlám bez kamery je možnosť spoľahlivejšej spätnej väzby ohľadom úspešnosti vykonanej práce, respektíve možnosť zistiť, aké neočakávané okolnosti nastali a riešiť ich individuálne.

Hlavnou nevýhodou je síce cena – o tom by sa ale dalo diskutovať, keďže firmy, ktoré potrebujú vozidlá na konkrétne činnosti, môžu nakúpiť veľa nakonfigurovaných vozidiel pozostávajúcich z rovnakých komponentov za veľkoobchodné ceny, pričom bežný používateľ, ktorý chce iba zopár kusov na trhu, kde nie je o nákup žiaden záujem, musí zaplatiť plnú cenu za všetky komponenty bez dostatočného výberu medzi konkurenčnými firmami a dokonca si musí vozidlo poskladať a naprogramovať sám. Tejto skupine ľudí ale náš projekt pomôže aspoň vo výbere vhodných komponentov a v programovej časti.

2 Cieľ a riešenie projektu

Cieľom nášho projektu bolo vytvoriť vozidlo schopné pohybu, ktoré je riadené cez Ethernet (IEEE 802.3 štandard, pripojenie káblom) alebo priamo cez WiFi (IEEE 802.11 štandard, pripojenie bezdrôtovo) pomocou programu kompatibilného aspoň s počítačom (napríklad cez Java klient), potenciálne aj mobilným telefónom alebo bezdrôtovým ovládačom.

Obrázok 4: Ilustrácia vozidla ovládaného cez internet

Toto vozidlo nám má umožniť na diaľku vykonávať funkcie:

- jazdenie všetkými štyrmi smermi s nastaviteľnou rýchlosťou,
- spomalenie v prípade prekážky,
- výstup z web kamery na vozidle,
- otáčanie web kamery,
- zvukový vstup a výstup,
- odosielanie stavu o teplote v okolí vozidla.

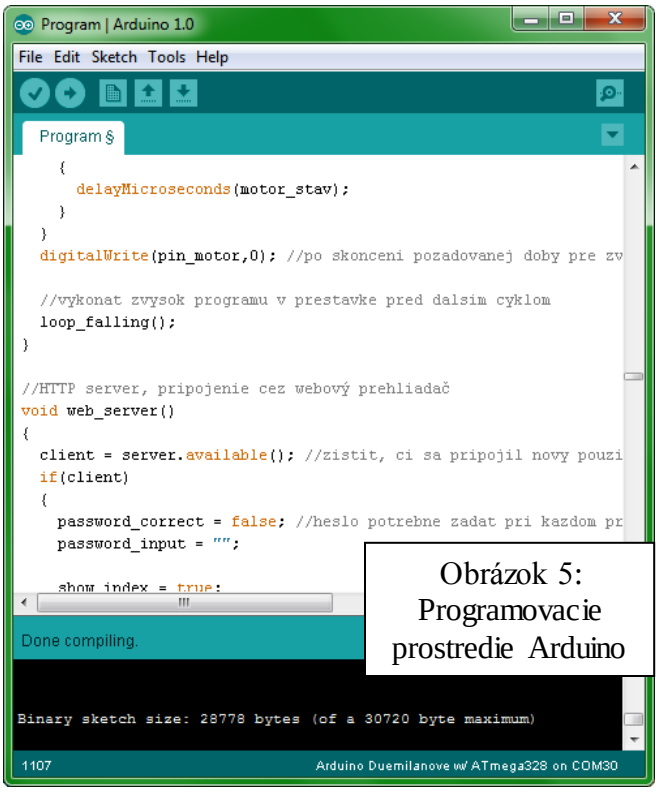
Riešenie projektu vyžaduje nezávislé vytvorenie dvoch častí práce: softvérovej (programovej) časti a hardvérovej (fyzickej) časti. Pri našom projekte sa ale zameriavame skôr na programovú časť, čo znamená, že hardvér bude pozostávať iba z existujúcich komponentov, ktoré zakúpime a pospájame do funkčnej podoby a nie z ich tvorby. Napriek tomu je ale potrebné plne chápať, na akom princípe jednotlivé komponenty fungujú a poznať, čo je potrebné na dosiahnutie požadovaného cieľa.

3 Programovanie vozidla

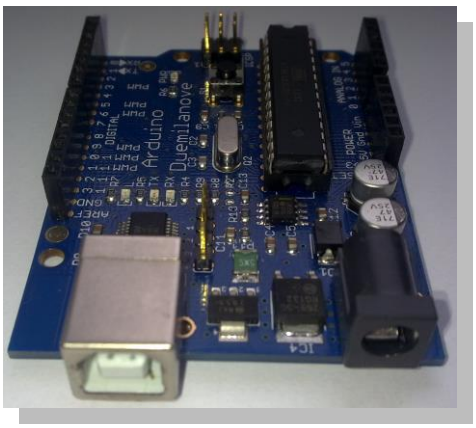
Riadenie vozidla zabezpečuje mikroprocesor ATMega328, ktorý sa programuje a prepája pomocou dosky Arduino Duemilanove (obr. č. 6). Táto doska pomocou USB portu komunikuje s počítačom, ktorý do nej cez open-source platformu Arduino posiela nový skompilovaný program (spracovaný zo zdrojového kódu na spustiteľný kód pre mikroprocesor). Po nahratí programu funguje nezávisle na počítači a po dodaní potrebného napájacieho napätia začne bežať program za použitia vstupno-výstupných portov Arduina a tým pádom aj jeho Ethernet Shieldu s Ethernet konektorom.

Programovací jazyk prostredia Arduino (obr. č. 5) je založený na C/C++. Celý program je k dispozícii na stiahnutie z oficiálnej stránky (zdroj č. 1). Boli využité oficiálne knižnice Arduina, ktoré sú predinštalované automaticky, ale aj neoficiálne knižnice:

- **SoftTimer** (GNU licencia) – umožňuje viacerým cyklom bežať nezávisle na sebe, vďaka čomu sa eliminuje nedostatok toho, že jeden cyklus pauzou (funkciou delay) znemožní fungovanie druhého. My sme ju ale využili na pravidelné časované spúšťanie funkcie pre hlavný PWM servo motor. Táto knižnica potrebuje tiež stiahnutú knižnicu **PciManager** (GNU licencia).
- **OneWire** (odporúčaná z oficiálnej stránky) – umožňuje čítanie dát z digitálneho teplomera pomocou jedného vodiča.



Obrázok 5:
Programovacie prostredie Arduino



Obrázok 6:
Doska Arduino Duemilanove s mikroprocesorom ATMega328

Knižnice je potrebné priložiť do zložky Dokumenty daného používateľa, napríklad pri používateľskom mene Steve pod OS Windows 7 na disku C: je adresa:

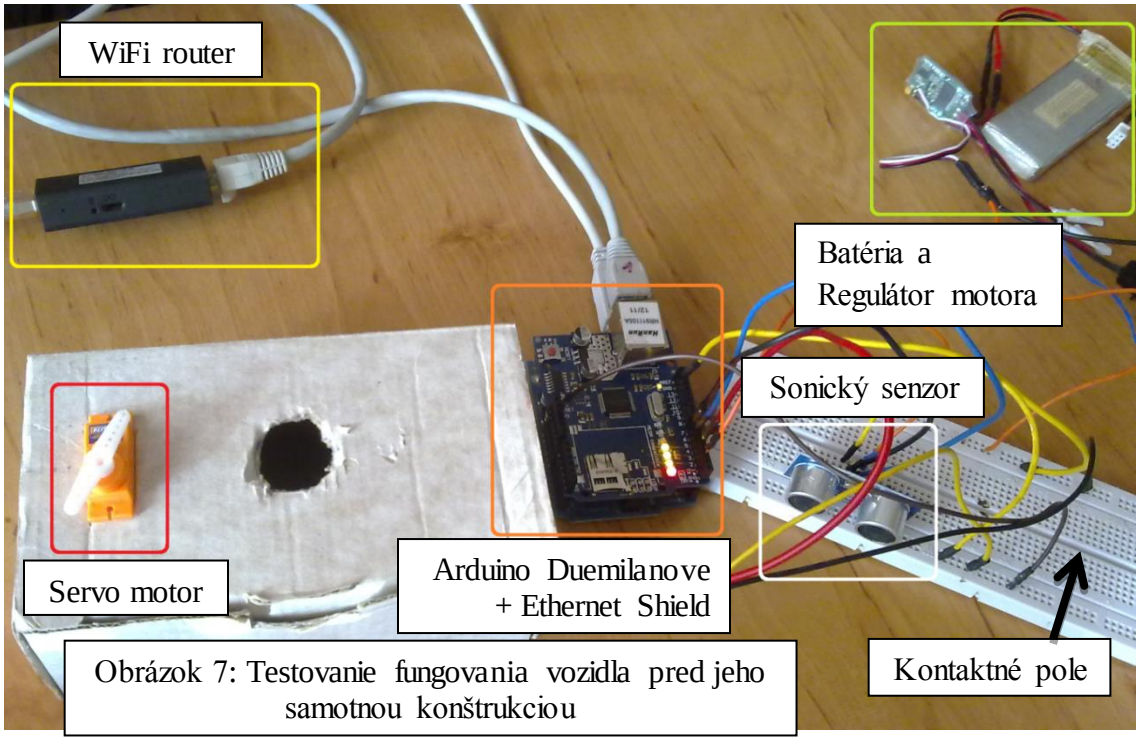
C:\Users\Steve\Documents\Arduino\libraries

Knižnice sa prikladajú v zdrojovom kóde na začiatku programu:

```
//Oficiálne knižnica Arduina pre Ethernet
#include <SPI.h>
#include <Ethernet.h>
#include <EthernetUdp.h>

//Oficiálna knižnica Arduina pre servo motory
#include <Servo.h>

//Oficiálna knižnica Arduina pre teplomer
#include <OneWire.h>

//Neoficiálna knižnica na časovanie funkcií
#include <SoftTimer.h>
```

3.1 Testovanie vozidla počas programovania

Ovládanie modulov prebiehalo za použitia kontaktného poľa. Všetky jednotlivé moduly boli pripájané do Arduina a testovali sa jednotlivo, ešte pred ich integráciou do samotného vozidla. Buď sa testovali moduly, ktoré sa neskôr použili priamo, alebo iba nejaké kompatibilné, ktoré sa ovládajú rovnako (napríklad bol použitý iný model PWM servo motora, ktorý veľkosťou a tvarom viac vyhovoval konštrukcii vozidla). (obr. č. 7)



Obrázok 7: Testovanie fungovania vozidla pred jeho samotnou konštrukciou

4 Programová časť vozidla

Program je z logickej časti rozdelený na viaceré cykly, ktoré sa pravidelne opakujú. Na začiatku sa raz spustí funkcia **setup**, časť jeho obsahu ako príklad:

```
void setup()
{
  Ethernet.begin(mac,ip);  //statická IP adresa

  server.begin();
  udp.begin(udp_port);

  pinMode(pin_motor,OUTPUT);
  riadenie.attach(pin_riadenie);

  SoftTimer.add(&motor_pohyb_task);
}
```

Po skončení funkcie setup sa začne dookola spúšťať funkcia **motor_pohyb**. Táto je definovaná ako **Task motor_pohyb_task(20, motor_pohyb);** (objekt knižnice SoftTimer.h), nastavená na opakovanie každých 20 milisekúnd a inicializovaná ako **SoftTimer.add(&motor_pohyb_task);** pod setupom.

Z týchto 20 milisekúnd sa vždy pre motor využijú maximálne 2 a zvyšných 18 sa prenechá funkcii hlavného cyklu programu s názvom **loop_falling**. Tým dosiahneme, že tieto funkcie môžu voľne využiť celých 18 milisekúnd procesorového času, napríklad aj funkciami delay bez negatívneho dopadu na fungovanie PWM motora.

4.1 Hlavný cyklus

Hlavným cyklom je funkcia, ktorá sa volá **loop_falling** a má nasledovné úlohy:

1. Kontrola stavu batérie, posielanie signálu pre regulátor v prípade prekročenia maximálnej alebo poklesnutia pod minimálne povolené napätie.
2. Nameranie novej hodnoty z teplomera a jej uloženie do premennej, ktorá sa dá prečítať napríklad cez webové rozhranie.
3. Nameranie vzdialenosti sonického senzora v smere, ktorým je aktuálne vozidlo otočené. Uložená hodnota sa využije v iných funkciách.
4. Zabezpečenie funkčnosti webového servera na základe TCP protokolu a v prípade požiadavky od používateľa odoslanie požadovanej odpovede.

5. Zabezpečenie funkčnosti UDP servera, ktorý síce vykonáva skoro to isté, čo TCP webový server, lenže prenos je iba jednosmerný s dôrazom na maximálnu rýchlosť prenosu.
6. Vykonanie výpočtov pre riadenie motorov, ako napríklad postupné zrýchľovanie, alebo bezpečnostné zastavenie vozidla po strate spojenia s používateľom.

Program je rozdelený na tieto podprogramy za účelom ľahkej diagnostiky a zmeny, poprípade neskoršieho rozšírenia programu. Pre odstránenie určitej funkcie vozidla stačí v tejto časti zdrojového kódu programu zmazať danú časť a zvyšok programu bude stále fungovať. Taktiež je to hlavným miestom, kde sa stačí vrátiť, pokiaľ programátor nevie, ako sa nazýva akákoľvek hľadaná funkcia.

Úlohy 1, 2 a 3 sú zložené iba z jednoduchých činností ako načítavanie hodnoty napätia na vstupných pinoch Arduina a ich následné uloženie do potrebnej premennej, respektíve nastavenie napätia na výstupných pinoch. Zvyšné časti sú popísané nižšie.

4.2 Komunikácia TCP a UDP protokolom

Z používateľského uhla pohľadu je medzi týmito protokolmi rozdiel hlavne vo forme, akou sa s nimi dá komunikovať so serverom. TCP server používa webovú stránku, ktorú sa dá otvoriť priamo pomocou internetového prehliadača. UDP server vyžaduje od používateľa spustenie aplikácie, ktorá zaznamenáva stlačenie kláves na klávesnici počítača a posiela ich veľkou rýchlosťou dookola na udržanie spojenia.

4.2.1 Formát prenášaných údajov

Jednotlivé príkazy sa z prehliadača (pri TCP protokole) a z aplikácie klienta (pri UDP protokole) posielajú vo forme sieťových packetov. TCP využíva podobnú metódu ako obyčajné webové servery – správa z prehliadača sa skladá z textu „GET /“, za čím nasleduje konkrétny príkaz, ktorý Arduino spracuje. Odpovie aj v prípade, keď je toto celý príkaz – v prehliadači sa zobrazí úvodná stránka nazývaná index. Prehliadač okrem toho obvykle pošle aj ďalšie údaje (verzia podporovaného protokolu, používaný operačný systém a iné), tieto sú nám ale zbytočné a okrem GET príkazu ich ignorujeme. UDP funguje podobne, okrem toho, že sa vynechá text „GET /“ a posiela sa iba samotný príkaz.

Syntax príkazu sme spravili na podobnom princípe, aký používajú PHP webové servery: znak ‚&‘ sa nachádza pred každým príkazom (s výnimkou prvého, pred ktorým je znak ‚?‘), vďaka čomu sa dá naraz poslať viac než 1 príkaz v každom sieťovom packete. Príkaz sa skladá z jeho názvu a parametra, podľa ktorého sa vykoná, medzi ktorými je znak ‚=‘. Syntax príkazu na prihlásenie pomocou hesla „heslo“ a otočenie predných kolies smerom doľava teda je „?password=heslo&cmd=a“.

4.2.2 Ovládanie vozidla

Obidva servery v prípade požiadavky na pohyb motorov využijú funkciu **ovladanie**. Táto na základe vstupného znaku vykoná potrebnú činnosť, napríklad:

```
case 'w':
if(motor_smer==DOZADU)regulator_fix=1;
if
(motor_zrychlovanie_i==0||motor_smer==DOZADU)
{
  motor_smer=DOPREDU;
  motor_zrychlovanie_i=1;
}
motor_wait_i=1;
break;
```

```
case 'a':
riadenie.write(VLAVO);
riadenie_wait_i=1;
break;


case 'd':
riadenie.write(VPRAVO);
riadenie_wait_i=1;
break;
```

Plynulé zrýchľovanie motora používa nasledovný algoritmus: pokiaľ sa buď ešte nezrýchľuje smerom dopredu, alebo sa už jazdí smerom dozadu, premenné sa nastavia na zrýchľovanie smerom dopredu od najmenšej rýchlosti. Tieto premenné sa využijú v neskoršej časti programu funkciou **pocitanie_servo**, tu je potrebné ich iba nastaviť. Pohyb dozadu funguje podobne, iba s inými premennými.

Priame spustenie konštantnej rýchlosti motora sa dosiahne tým, že namiesto využitia premennej **motor_zrychlovanie_i** sa okrem zmeny **motor_smer** nastaví aj **motor_stav**.

Otáčanie predných kolies je jednoduchšie. Objekt typu Servo, ktorý sme nazvali **riadenie** obsahuje funkciu na zmenu stavu: **riadenie.write** s hodnotou parametra od 0 po 180. Túto funkciu stačí zavolať priamo a pracuje v pozadí – nespôsobuje, že by program musel čakať na dokončenie otáčania.

Automatické zastavenie riešia premenné **motor_wait_i** a **riadenie_wait_i** rovnako, ako premenné pri zrýchľovaní. Po ich nastavení na hodnotu 1 bude zvyšok riešiť funkcia **pocitanie_servo**.

4.2.3 Webová stránka

Táto sa nachádza iba pod TCP protokolom. Okrem prijímania a spracovávania TCP packetov a vykonávania požadovaných príkazov zároveň odpovie klientovi vo forme HTML stránky. Pripojenie priamo (na IP adresu vozidla) otvorí index.html – hlavnú stránku, ktorá obsahuje linky a tlačidlá ovládajúce jednotlivé príkazy. Okrem toho obsahuje iframe, cez ktorý sa dá sledovať výstup z web kamery.

4.3 Pohyb motora

Funkcia **motor_pohyb** je síce hlavnou (a technicky jedinou) funkciou spúšťanou v cykle priamo a ostatné funkcie sú spúšťané z nej, no hierarchicky je pod funkciou loop_falling, keďže robí iba jednu činnosť: PWM signál pre motor.

Spúšťa sa každých 20 milisekúnd kvôli princípu PWM (impulzová šírková modulácia) – regulátor motora musí pravidelne dostávať signál o dĺžke 20 milisekúnd, z čoho určitú dobu najprv prichádza signál 1 a zvyšok je 0. Tento signál môže mať aj trochu väčšiu alebo menšiu dĺžku, stanovili sme 20 ms preto, aby ostalo dosť priestoru pre prácu ostatných modulov. Dĺžka signálu 1 určuje rýchlosť pohybu motora. Keďže bol použitý regulátor umožňujúci aj pohyb dozadu (reverzný pohon), používajú sa nasledovné hodnoty signálu 1 (v mikrosekundách):

- **1750 – 2000** = signál dopredu
- **1500** = žiaden pohyb (tento signál musí prichádzať, inak regulátor nie je pripravený na žiaden pohyb, takže nestačí vôbec neposielať signál 1)
- **1000 – 1350** = pohyb dozadu

Signál 0 je vypočítaný takto: dĺžka celého impulzu mínus dĺžka signálu 1. Programovo je to zabezpečené tým, že funkcia sa spúšťa pravidelne každých 20 milisekúnd a najprv na požadovanú dobu posiela signál 1, potom do konca svojho cyklu posiela signál 0. Pre túto funkciu sú pripravené konštanty:

```
#define MIN_DOPREDU   1750
#define MAX_DOPREDU  2000
#define STOP              1500
#define MIN_DOZADU    1350
#define MAX_DOZADU   1000
```

Okrem toho funkcia zabezpečuje aj obmedzenie maximálnej rýchlosti v prípade prekážky. Premenné, ktoré určujú maximálne hodnoty rýchlosti sa nazývajú **motor_sonic_limit_dopredu** a **motor_sonic_limit_dozadu**. Toto sa vykonáva iba v prípade, že používateľ pomocou príkazu cez webový server limit nevypol.

4.3.1 Servo motor riadiaci predné kolesá

Vďaka oficiálnej knižnici **Servo.h** stačí definovať nový objekt, v tomto prípade **Servo riadenie;** a inicializovať ho v setupe pomocou **riadenie.attach(pin_riadenie);**. Funkcia na zmenu stavu je **riadenie.write** s hodnotou parametra od 0 po 180. Z dôvodu prehľadnosti sú tieto nahradené konštantami:

```
#define  ALT_VLAVO    120
#define  VLAVO        160
#define  STRED         90
#define  VPRAVO        30
#define  ALT_VPRAVO    60
```

4.4 Riadenie motorov

Pre riadenie motorov sme museli pravidelne prepočítavať pomocné premenné automatického riadenia. Funkcia pre túto činnosť je nazvaná **pocitanie_servo**. Skladá sa z nasledovných úloh:

- zastavenie motora po uplynutí doby latencie,
- vycentrovanie predných kolies po uplynutí doby latencie, ale iba v prípade, že je toto pomocou webového servera aktivované,
- zrýchľovanie motora smerom vpred alebo vzad.

Prvá a druhá činnosť majú za úlohu vypočítať, či prešla požadovaná doba bez signálu od používateľa a ak áno, nastaviť stav na základný. Premenné, ktoré to sledujú sa volajú **motor_wait_i** a **riadenie_wait_i**. Pokiaľ sa v akejkoľvek časti programu nastavia na hodnotu 1 a vyššiu, spustí sa odpočet, čo v praxi znamená, že postupným vykonávaním cyklu **loop_falling** sa hodnota bude stále zvyšovať o 1. Po dosiahnutí hodnoty **motor_wait** a **riadenie_wait**, ktoré sa inak dajú nazvať latencia, respektíve maximálna hodnota pred resetom, sa motor zastaví a riadenie vycentruje. Za tým sa hodnoty nastavia na nulu, čím sa odpočet zastaví.

Zrýchľovanie používa premennú **motor_zrychlovanie_i**, ktorá sa podobným princípom zvyšuje, až kým nedosiahne hodnotu **motor_doba_zrychlovania**. Po jej

dosiahnutí sa rýchlosť motora – premenná **motor_stav** nastaví na **MAX_DOPREDU** alebo **MAX_DOZADU**. Výpočet rýchlosti počas zrýchľovania je v programe nasledovný:

- **Dopredu:** „*motor_zrychlovanie_i / motor_doba_zrychlovania * (MAX_DOPREDU - MIN_DOPREDU) + MIN_DOPREDU*“
- **Dozadu:** „*MIN_DOZADU - motor_zrychlovanie_i / motor_doba_zrychlovania * (MIN_DOZADU - MAX_DOZADU)*“

4.5 Hierarchia funkcií programovej časti vozidla

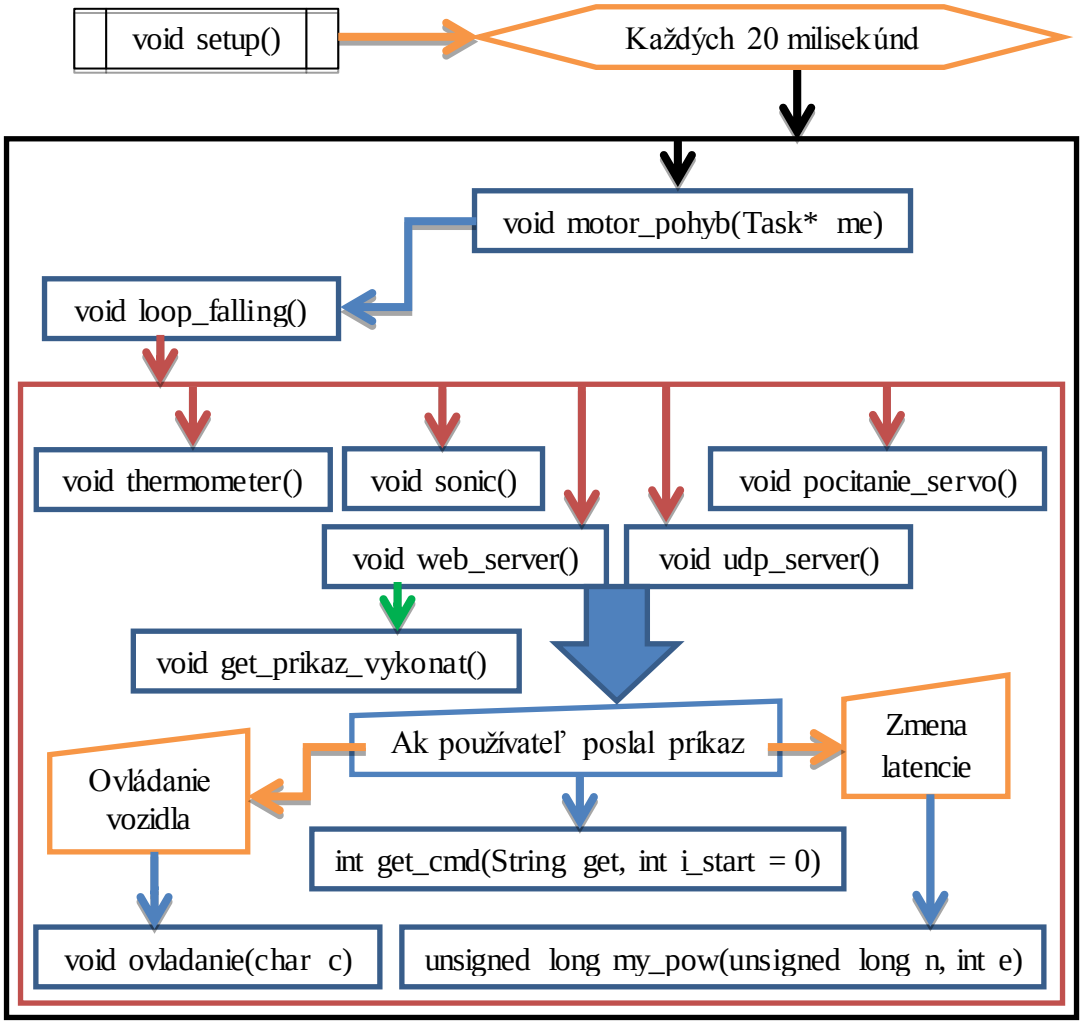


Schéma 1: Teoretická hierarchia zdrojového kódu Arduina

5 Programová časť strany klienta

Ovládanie vozidla môže prebiehať z akéhokoľvek kompatibilného zariadenia, ktoré umožňuje komunikáciu pomocou internetu. Buď sa dá použiť jedna z našich metód, alebo vytvoriť inú aplikáciu, ktorá bude odosielať tie isté údaje pomocou rovnakého protokolu. Pre pripojenie sme vytvorili 2 alternatívy.

5.1 Komunikácia s TCP serverom Arduina

Pripojenie na webový server Arduina nepotrebuje (na počítači bežného používateľa) žiadne špeciálne aplikácie, iba štandardný internetový prehliadač. Po otvorení určitej HTTP adresy webový server okrem odoslania odpovede prehliadaču vykoná požadovaný príkaz vo vozidle. Z pohľadu používateľa je potrebné iba zadať do prehliadača sieťovú IP adresu vozidla a ovládať vozidlo klikaním na HTML stránke.

Nový stav sa po určitej dobe vráti do pôvodného stavu – napríklad stav pohybu hlavného motora sa z pohybu vpred prepne na zastavený, hlavne z dôvodu bezpečnosti, keby používateľ nestihol včas poslať príkaz na zastavenie. Toto rozhranie umožňuje aj nastaviť novú hodnotu latencie (dobu čakania pred zastavením), ako aj veľa ďalších nastavení. Tieto nastavenia platia aj pre UDP server (teda dá sa používať obe naraz).

Táto metóda nie je dostatočne spoľahlivá, keďže sme využili iba obyčajné GET príkazy HTML protokolu bez akýchkoľvek doplnkov, akým by bol napríklad JavaScript. Hlavnou nevýhodou je, že používateľ musí posielať príkaz stále nanovo dostatočne rýchlo, pokiaľ nechce, aby sa stav prepol na pôvodný. Pôvodne sme toto chceli riešiť automatickým refreshom (obnovením stránky) na strane prehliadača, lenže spoľahlivosť, rýchlosť a kompatibilita s rôznymi prehliadačmi nebola dostatočná, taktiež použité Arduino nestíha odosielať odpoveď dostatočne rýchlo a pri častých HTTP požiadavkách sa zvykne reštartovať. To síce trvá iba zlomok sekundy, ale znepríjemňuje zážitok z ovládania a potrebnú funkčnosť. Preto túto možnosť síce necháme ako druhú (zastaranú) alternatívu, ale hlavná metóda ovládania bude pomocou UDP protokolu cez aplikáciu klienta.

5.2 Java klient komunikujúci UDP protokolom

Tento klient bol vytvorený pomocou programovacieho prostredia Processing (obr. č. 8), ktoré na rozdiel od Arduina neprogramuje ATMega328 mikročip, ale umožňuje vytvárať aplikácie pre počítač. Vďaka podpore jazyka Java kompatibilita ostane zachovaná aj medzi viacerými operačnými systémami. Je možné vytvoriť aj JavaScript a Android aplikácie, lenže kvôli nedostatku skúseností v tejto oblasti sme sa rozhodli vytvoriť iba Java klienta.

Obrázok 8: Programovacie prostredie Processing

Celý program je k dispozícii na stiahnutie z oficiálnej stránky (zdroj č. 2). Boli využité oficiálne, automaticky predinštalované knižnice ale aj neoficiálne knižnice:

- **UDP** – umožňuje sieťovú komunikáciu pomocou UDP protokolu.
- **G4P** – pridá podporu objektov akými sú textové polia, popisy a tlačidlá. Táto knižnica sa dá skombinovať s nástrojmi G4PTools.

Okrem knižníc boli použité nástroje (rozšírenia programovacieho prostredia):

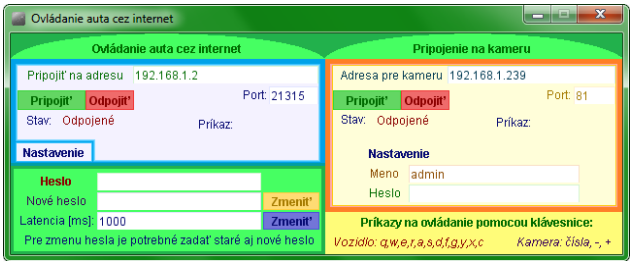
- **G4PTools** – pridajú do Processing rozhrania možnosť návrhu grafického rozhrania pre používateľa (GUI).

Knižnice je potrebné priložiť do zložky Dokumenty daného používateľa, napríklad pri používateľskom mene Steve pod OS Windows 7 na disku C: je adresa: *„C:\Users\Steve\Documents\Processing\libraries“*.

Pre nástroje je cesta: *„C:\Users\Steve\Documents\Processing\tools“*.

Knižnice sa prikladajú v zdrojovom kóde na začiatku programu:

```
// Neoficiálna G4P knižnica pre grafické rozhranie
import g4p_controls.*;

// Neoficiálna UDP sieťová knižnica
import hypermedia.net.*;

// Oficiálna TCP sieťová knižnica
import processing.net.*;
```



Obrázok 9: Java aplikácia na ovládanie vozidla cez UDP protokol a otáčanie kamery

6 Pripojenie vozidla do LAN siete

Cieľom nášho projektu je, aby bolo možné vozidlo ovládať cez Internet, no v praxi to z dôvodu princípu fungovania TCP/IP protokolov znamená možnosť ovládania cez akúkoľvek pripojenú sieť. Všetko je to zabezpečené správnou voľbou metód routovania. Toto je ale na rozdiel od hardvérovej a softvérovej časti úplne nezávislá téma, pretože sieť sa rieši mimo vozidla a nie je súčasťou tohto projektu. Bez plného prístupu ku administrácii routera danej siete na väčšine sietí nebude možné pripojenie na vozidlo cez internet, iba zo zariadení nachádzajúcich sa na danej sieti.

Arduino je pomocou Ethernet Shieldu pripojené priamym Twisted Pair Ethernet káblom do WiFi routera. V zdrojovom kóde Arduina je nastavená IP adresa staticky na **192.168.1.2**. V prípade, že by bolo niekedy potrebné zmeniť ju, je potrebné prepísať zdrojový kód, skompilovať a poslať ho do Arduina nanovo. Taktiež je možnosť automatického získania IP adresy pomocou DHCP protokolu, stačí pod funkciou **setup** nad riadkom **Ethernet.begin(mac,ip);** odkomentovať (vymazať znaky „//“ na začiatku) riadok **if(Ethernet.begin(mac)==0)**.

WiFi router sme nastavili na IP adresu a port **192.168.1.1:80**. Pre prihlásenie sa do administrácie routera stačí túto adresu po pripojení do jeho WiFi siete zadať ako adresu do prehliadača a prihlásiť sa so správnymi údajmi. Aktuálne heslá z dôvodu bezpečnosti v dokumentácii neuvádzame. Keby ich každý poznal, boli by zbytočné a aj tak sa musia pravidelne meniť. Všetky komponenty sa dajú resetovať na výrobné nastavenia v prípade zabudnutia údajov. V prípade „tvrdého resetu“ routera sa jeho adresa nastaví na **192.168.100.1:80**. Názov WiFi siete sa nastaví na „**MIFI_580F50**“, heslo od siete bude „**12345678**“ a prihlasovacie meno aj heslo do administrácie budú „**admin**“. IP adresu je potrebné po resete prestaviť naspäť na našu adresu pre správnu funkčnosť (alebo podľa vlastného uváženia prestaviť všetky ostatné komponenty). Ak v administrácií routera nastavíme, aby sa pripojil do už existujúcej LAN, pomocou funkcie **Virtual Server** je potrebné nastaviť presmerovanie všetkých vnútorných IP adries na jednu vonkajšiu, z pohľadu druhej LAN siete teda vnútornú IP adresu.

WiFi IP kamera má nastavenú IP adresu a port **192.168.1.239:81**. Táto je nastavená automaticky aj po „tvrdom resete“ (podržaní tlačidla na reset po dobu 1 minúty), vtedy sa ale treba pripojiť káblom (rovnakým ako Arduino s routerom) a prihlasovacie meno sa nastaví na „**admin**“, heslo bude prázdne. Jediné, čo treba spraviť, je v nastaveniach otvoriť Network – Wi-Fi a pripojiť kameru na sieť Arduina.

6.1 Pripojenie do siete internetu

Podmienkou je, aby ISP (poskytovateľ internetovej služby) poskytoval pre danú počítačovú sieť, do ktorej je pomocou WiFi vozidlo pripojené, verejnú a voči prístupu zvonku neblokovanú IP adresu. Toto býva štandardom pri bežnom pripojení, no existujú výnimky. Taktiež nesmie blokovať porty.

Jediná vec, ktorú treba nastaviť, je presmerovanie portov z vnútornej siete do vonkajšej siete (internetu). Vozidlo má vnútornú IP adresu **192.168.1.2** a pracuje na dvoch portoch, **80** pre webový server a **21315** pre UDP server na prijímanie príkazov od Java klienta spomenutého v bode 5.2. IP kamera má vnútornú IP adresu **192.168.1.239** a používa jedine port **81** pre webový server, pomocou ktorého sa sleduje obraz.

Cieľom je tieto tri porty presmerovať na spoločnú verejnú IP adresu, ktorú dodal ISP. Postup ale nie je jednoznačný, pretože každá sieť používa odlišný model routera na prepojenie vnútornej siete s vonkajšou sieťou (internetom). Ak to router nepodporuje, ovládanie cez internet v danej sieti nebude možné. Najčastejšie je postup nasledovný:

- do prehliadača treba zadať IP adresu routera, často je to „192.168.1.1“,
- keď si prehliadač vypýta prihlasovacie údaje, je potrebné zadať správne údaje, často je meno aj heslo „admin“ (základné údaje bývajú v dokumentácii routera),
- v administrácii by sa mala nachádzať položka „Port forwarding“,
- po jednom treba priradiť všetky vnútorné adresy a porty k vonkajším portom.

Tabuľka 1: Štruktúra odporúčaných hodnôt pre presmerovanie portov

Vnútorná adresa	Vnútorný port	Vonkajší port
192.168.1.2	80	80
192.168.1.2	21315	21315
192.168.1.239	81	81

Ak je router pripojený do inej LAN siete, namiesto týchto IP sa použije IP routera v danej LAN. Týmto dosiahneme, že pripojením sa na vonkajšiu IP adresu cez daný port, router presmeruje požiadavku do vnútornej adresy pod rovnakým portom. Adresu vlastnej verejnej IP je možné zistiť vyhľadaním internetovej stránky slúžiacej na to pomocou vyhľadávača, no keď sa nechceme spoliehať na neznáme služby, môžeme sa opýtať na technickej podpore nášho poskytovateľa internetu, alebo v prípade kvalitného routera nájsť si túto adresu pod administráciou. Túto adresu zadáme buď do prehliadača, alebo do UDP klientskej Java aplikácie a môžeme ovládať vozidlo.

7 Technická časť vozidla

Samotné vozidlo sa skladá z modulov zabezpečujúcich napájanie, požadovaný pohyb motorov, sieťových zariadení a prepájacích káblov. Ochranu vnútorných súčiastok pred manipuláciou a poškodením vonkajšími vplyvmi, taktiež lepší vzhľad auta zabezpečuje obal vozidla. Bolo ho ale potrebné upraviť, pretože všetky komponenty sa dovnútra pôvodne nezmestili. Niektoré komponenty sú namontované z vonkajšej strany obalu. Regulátor motora je pripevnený suchým zipsom a kamera originálnou skrutkou.

Pre napájanie vozidla bolo použité bezpečné napätie vo výške 7V a ďalej vyregulované na stabilizovaných 5V. Vzhľadom k uvedeným skutočnostiam boli dodržané elektrotechnické bezpečnostné predpisy.

7.1 Centrálne napájanie vozidla

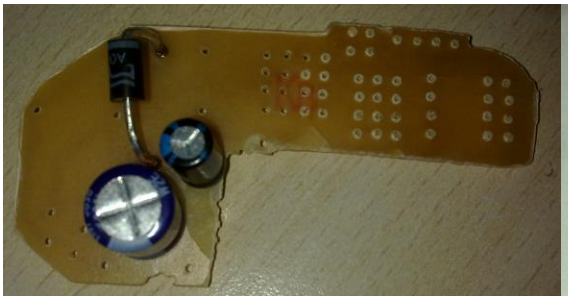
Vozidlo je napájané 7V batériou, ktorá sa bežne využíva v modelárstve. Ako ochranu pred nesprávnym zapojením sme použili diódu na kladnom póle batérie. Meranie napätia na batérii funguje tak, že sme ju pripojili na delič napätia a vývod sme pripojili do Arduina na analógový pin **A0**. Pomocou pinu 9 z Arduina pripájame feedback do regulátora batérie, ktorý zastaví dodávanie napätia ostatným komponentom v prípade, že program vyhodnotí stav batérie ako nepovolený. Namiesto batérie je možné aj pripojiť adaptér dodávajúci 5V.

Pre transformáciu vstupného napätia z batérie na 5V sme vytvorili vlastný plošný spoj, ktorý má za úlohu ako regulovanie napätia, tak aj zjednodušenie vstupov a výstupov. Sú na ňom nasledujúce súčiastky:

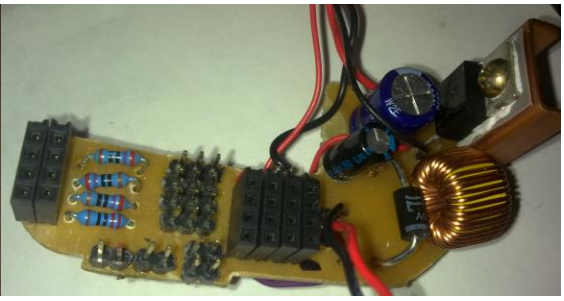
- cievka so 100 závitmi o indukčnosti 90mH,
- usmerňovacia dióda,
- elektrolytické kondenzátory 100uF 16V a 1000uF 16V,
- 4 odpory pre napájanie LED diódy,
- ICE menič napätia zo 7V až 30V na 5V/3A,
- chladič pre ICE menič napätia.

Postup výroby plošného spoja bol nasledovný: odmerali sme voľné miesto v obale vozidla a navrhli sme plošný spoj, ktorý má požadovanú veľkosť a tvar. Najprv sme navrhli náhradnú schému na papier, ktorú sme neskôr vypracovali vo voľne dostupnom programe EAGLE (zdroj č. 3). Schému regulátora sme našli na webovej

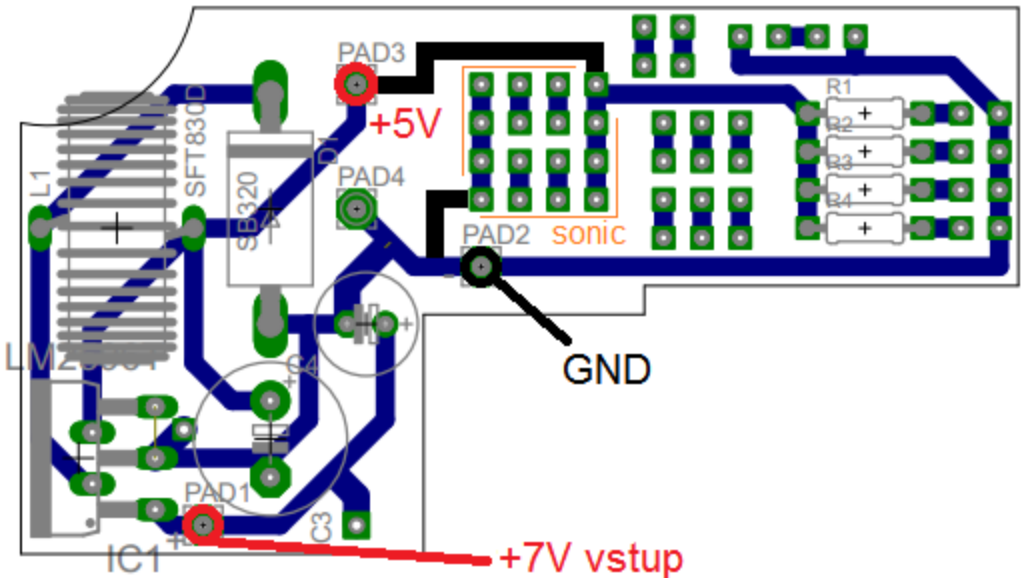
stránke výrobcu (zdroj č. 4) a prepájacie kontakty sme navrhli podľa potreby. Hotovú schému sme vytlačili na fóliu. Pomocou nej sme fotocestou vyleptali potrebné spoje na plošný spoj. Tu sme dodatočne navŕtali diery pre nacínovanie súčiastok po ich zapojení.



Obrázok 10: Nedokončený plošný spoj bez súčiastok



Obrázok 11: Dokončený plošný spoj pred pripojením



Obrázok 12: Schéma plošného spoja na zmenu napätia z 7V na 5V, mierka 2:1

7.2 Pripojenie ostatných komponentov

Ako riadenie predných kolies sme použili modelárske servo strednej veľkosti, ktoré napájame pomocou 5V z plošného spoja predchádzajúcej schémy. (obr. č. 12) Signál dostáva z Arduina cez pin 3. Ovládanie predných kolies funguje pomocou oceľových sponiek, pretože sme podľa požiadaviek vozidla museli zabezpečiť menší uhol otáčania kolies, inak by hrozilo ich vylomenie. Softvérovo sa toto vyriešiť nedalo, keďže Arduino po reštarte chvíľu posiela do všetkých pinov náhodné hodnoty napätia.

Pohon zadných kolies zabezpečuje jednosmerný motor z tlačiarne. Prevod motora na pohon zadných kolies sme vyriešili pomocou ozubených kolies. Použili sme

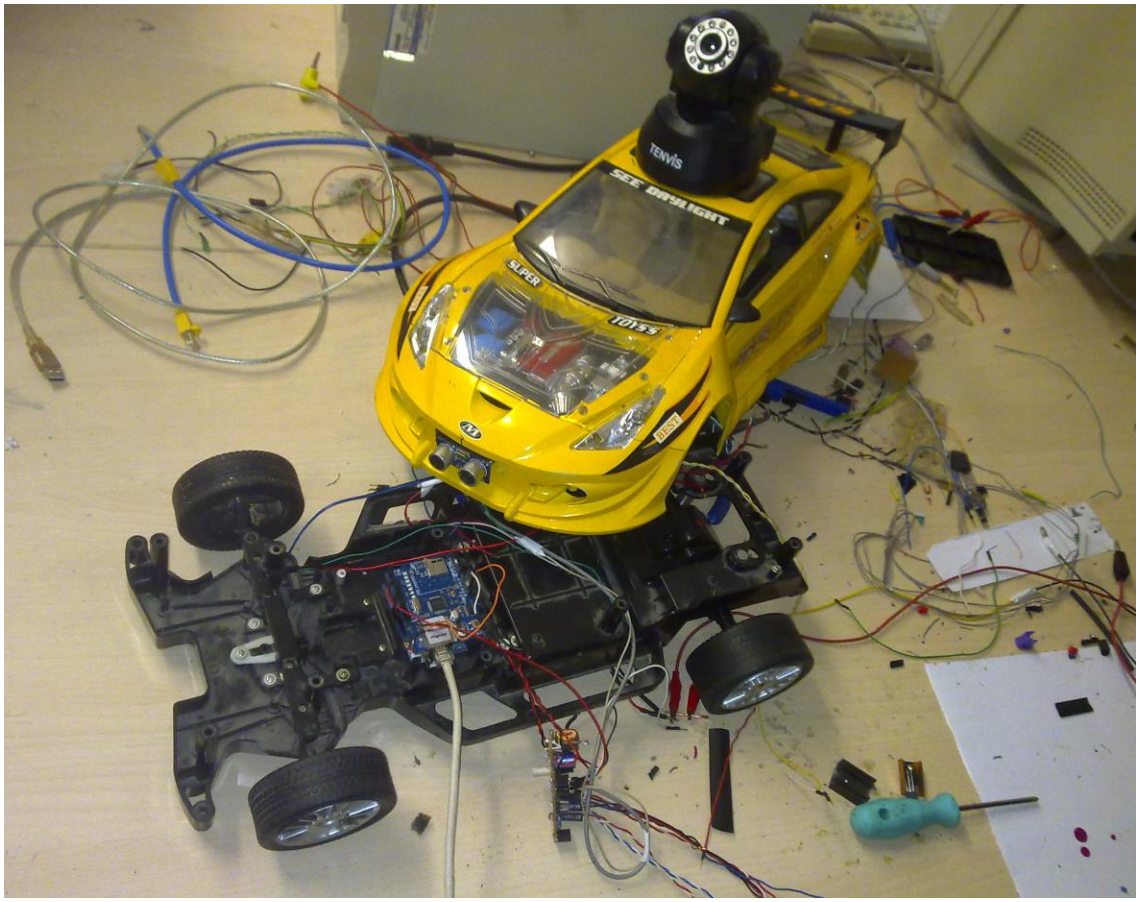
súčiastky z pokazenej tlačiarne. Napájanie neprebieha priamo z batérie, ale cez modelársky regulátor s parametrami 4V až 8,4V a 20A, ktorý zároveň riadi motor pomocou signálu z Arduina cez pin 2.

Router požaduje napájacie napätie 5V, ktoré sme mu dodali pomocou USB kábla, z ktorého sú piny zapojené do nášho plošného spoja a tým pádom dodávané tiež z batérie. Do Ethernet Shieldu Arduina je pripojený priamym Twisted Pair Ethernet káblom.

Kamera je tiež napájaná z plošného spoja, ale pripája sa bezdrôtovo, s výnimkou prvotnej konfigurácie z počítača. Vtedy sa použije rovnaký kábel, akým sa prepája router s Arduinom.

Pre sonické, respektíve ultrazvukové senzory sme vytvorili konkrétne miesto na plošnom spoji. Napájanie a uzemnenie je dodávané priamo, no piny nazvané **echo** a **trigger** sú vyvedené do káblov, ktoré zapájame do nasledujúcich pinov: echo dopredu 4, trigger dopredu 5, echo dozadu 6, trigger dozadu 7.

Teplomer zapájame stredným, dátovým konektorom do Arduino pinu 8, napájanie a zem je tiež z plošného spoja rovnako ako pri ostatných komponentoch.



Obrázok 13: Skladanie finálneho vozidla

7.3 Cenová kalkulácia

Tabuľka 2: Zoznam všetkých použitých komponentov hardvérovej časti a ich cena

Názov komponentu	Cena	Dátum nákupu
Hračkárske autíčko sériovej výroby	86.00€	5.10.2010
Batéria LiPo 2S	7.60€	10.2.2010
Regulátor otáčok motora	5.60€	20.10.2013
Motor na pohon auta	4.20€	15.9.2013
Servo motor	2.70€	10.1.2014
WiFi IP kamera	48.00€	18.12.2013
Ultrasonické senzory 2ks	2.78€	23.1.2014
Digitálny teplomer DS18B20	2.16€	15.10.2013
Arduino Duemilanove s mikroprocesorom ATMega328	9.33€	12.5.2013
Ethernet Shield pre Arduino	7.80€	10.10.2013
WiFi Router mini	5.30€	15.10.2013
Rezistory 220Ω 4ks	0.40€	18.9.2013
Kremíková dióda 1N5408	0.04€	18.9.2013
Elektrolytický kondenzátor 100µF/16V	0.03€	18.9.2013
Elektrolytický kondenzátor 2200µF/16V	0.13€	18.9.2013
Cievka 100mH	0.80€	18.9.2013
Stabilizátor napätia LM2576HV	4.40€	18.9.2013
Doska plošného spoja 7x4 cm	1.12€	18.9.2013
Spolu	**188.39€**	

8 Výsledky práce a diskusia

Výsledkom práce je funkčné vozidlo so všetkými navzájom spolupracujúcimi komponentmi po hardvérovej stránke a program pre Arduino s programom pre počítač po softvérovej stránke. Podľa očakávaní sa podarilo vyriešiť sieťové prepojenie počítača s vozidlom cez WiFi v lokálnej, vnútornej sieti a aj routovanie packetov cez internet, vonkajšiu sieť.

Zistili sme aj rozličné nedostatky vozidla. Internetové spojenie máva výpadky, kamera nie vždy stíha posielať údaje plynule a ovládanie vozidla býva oneskorené. Arduino s ATMega328 mikroprocesorom má tiež limity čo sa týka rýchlosti a spoľahlivosti, pokiaľ od neho požadujeme viacej paralelných úloh, ako načítavanie hodnôt zo senzorov, prenos údajov cez TCP a UDP servery, meranie vzdialenosti od prekážky a teploty v okolí, pohyb motorov a výpočty v pozadí. Pamäť sme síce nevyužili naplno, ale pri ďalšom rozšírení programu už bude potrebné mikroprocesor vymeniť za lepší.

Počas vypracovávania projektu sme porovnávali viacero alternatívnych metód postupu, napríklad napájanie sme plánovali riešiť napäťovým regulátorom 7805, ktorý je žiaľ ale iba na 1, maximálne 2 ampére, ale zistili sme, že kamera má špičky napájania a preto sme to riešili úplne novým zapojením so stabilizovaným regulátorom LM2576HV, ktorý dokáže dodať až 3 ampére.

Pokiaľ by sme v budúcnosti plánovali tento projekt vylepšiť, stále je na to dosť priestoru. Mikroprocesor sa dá vymeniť za výkonnejší, batéria za výkonnejšiu, ktorá má dlhšiu výdrž, kamera za drahší a menší model a WiFi router za kvalitnejší s väčším dosahom. V súčasnosti použitý router má nízku citlivosť a preto sa ťažko chytá WiFi signál, čo má za následok, že aj keď router v určitej LAN sieti pokryje router vo vozidle, tento mu nedokáže odpovedať a spojenie je nespoľahlivé.

9 Závery práce

Našou snahou bolo vytvoriť vozidlo využiteľné ako na praktické činnosti, tak aj zábavu, lenže prakticky vykonateľná úžitková práca a zážitok z ovládania nesplnili naše idealistické očakávania. Ovládanie je síce pri vhodnom pokrytí WiFi sieťou dostatočne rýchle, ale napriek tomu prakticky nie je možné dosiahnuť zážitok porovnateľný s počítačovými hrami, ani účinne vykonávať činnosti ako preprava predmetov.

S tým sme ale počítali, cieľom nebolo vytvoriť prvý prototyp ako dokonalý. Naopak, vďaka tomuto projektu sme nazbierali veľa rozličných skúseností z oblasti praktického programovania reálnych zariadení, spolu s informáciami o rozličných metódach prístupu k tejto problematike. Napriek tomu, že sme sa rozhodli pre niektoré konkrétne voľby, najprv bolo potrebné skúšať viacero metód a stotožniť sa s nimi. Väčšina získaných informácií nám môže pomôcť aj v úplne nezávislých projektoch.

Projekt pozostával okrem programovania z výberu vhodného obalu pre vozidlo, do ktorého sa museli zmestiť všetky potrebné časti a jeho následných modifikácií. Hardvérové záležitosti, ako napájanie, so sebou niesli rôzne problémy, pretože veľa modulov potrebovalo aj rozličné napätie a aj zabezpečenie dostatočného prúdu. Toto musela vedieť zvládnuť jedna batéria, ktorú je potrebné správne regulovať, ale neskôr sme pridali aj možnosť napájania z elektrickej siete pomocou adaptéra. Porovnali sme rôzne moduly a objednali sme vhodnú WiFi IP kameru, malý WiFi router a Ethernet Shield, ktoré bolo treba správne nastaviť pre spoluprácu v sieti.

Hlavná časť práce, program, bola ale časovo omnoho náročnejšia. Vďaka rozsiahlej dokumentácii a rozličným knižniciam sme dokázali pomocou mikroprocesora ovládať aj zložitejšie činnosti, ale väčšina programu napriek tomu pozostávala z vlastných algoritmov na ovládanie vozidla na základe konkrétnych sieťových packetov. Navrhli sme syntax príkazov a metódu ich spracovania. Na strane Arduina spúšťame UDP, ale aj TCP server, ktorý obsahuje HTML stránku, cez ktorú sa nastavujú aj vedľajšie parametre okrem jazdenia, vrátane merania teploty a vzdialenosti od prekážky. Klientska aplikácia je úplne nezávislý Java program, vytvorený cez Processing rozhranie.

10 Zhrnutie

Cieľom práce bolo vytvoriť a naprogramovať vozidlo ovládané pomocou počítačovej siete, z ktorej sa komunikácia následne pomocou routera presmeruje do siete internetu. Toto vozidlo má disponovať kamerou na prenos obrazu v reálnom čase spolu s ďalšími komponentmi ako sonický senzor na spomalenie pri detekcii prekážky, digitálny teplomer, mikrofón a reproduktor. Ovládanie vozidla prebieha pomocou počítača alebo kompatibilného zariadenia pomocou TCP protokolu cez webovú stránku alebo UDP protokolu cez Java klienta.

Technická aj softvérová časť vozidla bola vypracovaná nezávisle, pričom dôraz práce bol na softvérovú časť. Komponenty boli iba nakúpené a nie vytvárané, technická časť pozostávala hlavne z prepojenia komponentov a zabezpečenia napájania z batérie. Programovanie mikroprocesora prebiehalo v C++ kompatibilnom jazyku. Tento projekt ukazuje možnosť praktického prepojenia teoretickej informatiky s reálnymi zariadeniami mimo počítača.

11 Resume

The aim of our project was to create and program a vehicle controlled via computer network, from which the communication is subsequently forwarded using a router to the network of internet. This vehicle has to have a camera that allows video transmission in real time, along with other components like sonic sensor to slow down in case of detection of an obstacle, digital thermometer, microphone and a loudspeaker. Operation of the vehicle takes place through a computer or a compatible device using TCP protocol through web page or UDP protocol through Java client.

Hardware and software part of the vehicle was developed independently, while the emphasis was on the software part. Components were only purchased, we did not create them. The hardware part primarily consisted of interconnecting components and providing supply of electricity from a battery. Programming of the microprocessor took place in C++ compatible language. This project shows the possibility of practical usage of theoretical informatics combined with real devices outside the computer.

Použitá literatúra

1 Arduino. 2005. [online]. 2013, [cit. 2013-12-16]. Dostupné na internete: <http://arduino.cc/>

2 Processing. 2004. [online]. 2013, [cit. 2013-12-16]. Dostupné na internete: <http://processing.org/>

3 CadSoft EAGLE PCB Design Software. 2011. [online]. 2013, [cit. 2013-12-16]. Dostupné na internete: <https://www.cadsoftusa.com/download-eagle/>

4 Step-Down (Buck) Switching Voltage Regulators with LM2576/LM2576HV. 2013. [online]. 2013, [cit. 2013-12-16]. Dostupné na internete: <http://www.hobby-hour.com/electronics/lm2576-step-down-switching-regulator.php>

5 RC Race Car SST2000. 2010. [online]. 2013, [cit. 2013-12-16]. Dostupné na internete: <https://en.wikipedia.org/wiki/File:RC_Race_Car_SST2000.jpg>

6 IEEE Standards associations, IEEE 802.11. 2012. [online]. 2013, [cit. 2013-12-16]. Dostupné na internete: <http://standards.ieee.org/getieee802/download/802.11-2012.pdf>

7 STOFFREGEN, Paul. Arduino knižnica OneWire. 2013. [online]. 2013, [cit. 2013-12-16]. Dostupné na internete: <http://playground.arduino.cc/Learning/OneWire>

8 Arduino knižnica SoftTimer. 2013. [online]. 2013, [cit. 2013-12-16]. Dostupné na internete: <http://code.google.com/p/arduino-softtimer/>

9 Arduino knižnica PciManager. 2013. [online]. 2013, [cit. 2013-12-16]. Dostupné na internete: <https://code.google.com/p/arduino-pcimanager/>

10 COUSOT, Stephane. Processing library UDP. 2014. [online]. 2014, [cit. 2014-02-10]. Dostupné na internete: <http://ubaa.net/shared/processing/udp/>

11 LAGER, Peter. Processing library G4P. 2014. [online]. 2014, [cit. 2014-02-10]. Dostupné na internete: <http://www.lagers.org.uk/g4p/>

12 LAGER, Peter. Processing knižnica G4PTools. 2014. [online]. 2014, [cit. 2014-02-10]. Dostupné na internete: <http://www.lagers.org.uk/g4ptool/>

13 Wi-Fi Alliance, Wi-Fi Logo. 2014. [online]. 2014, [cit. 2014-02-13]. Dostupné na internete: <https://upload.wikimedia.org/wikipedia/commons/thumb/3/32/Wi-Fi_Logo.svg/304px-Wi-Fi_Logo.svg.png>