Středoškolská technika 2014

Setkání a prezentace prací středoškolských studentů na ČVUT

Závodní elektrokolka řízená procesorem se zabezpečovacím systémem

Jiří Pouzar, Michal Batelka, Jakub Kaplan

Střední průmyslová škola dopravní, a.s.
Plzeňská 298/217a, Praha 5 - Motol

Abstrakt

Práce se zabývá vytvořením závodní elektrokolky řízené procesorem se zabezpečovacím systémem. Protože je však práce rozsáhlá, je zde jen hlouběji předveden zabezpečovací systém pracující s GPS a GSM, ochrana před krádeží a případné vyhledání polohy objektu prostřednictvím vysílaných zpráv. Ostatní části jako je řízení rychlosti pomocí PWM, měnič 36V/12 a 8V, snímače proudu a napětí včetně integrátoru a též ovládání jsou zde jen zmiňovány.

Obsah

1.0 Zadání práce

Práce může být realizována na libovolném mikrokontroleru, případně mikropočítači. Její funkcí bude zabezpečení a identifikace polohy pohyblivého objektu (auto, motorky aj.), na požadavek poslat sms zprávu na předvolené číslo mobilního telefonu, měřit a předávat buď pomocí Bluetoo, nebo záznamem do paměti data (U, I, ot^{-1}). Práce musí být realizována, nikoliv jen navržena a být plně funkční. V případě, že bude použita přídavná paměť, pak je třeba přidat i možnost snímání a mazání dat. Musí umět realizovat následující funkce:

1. Určit polohu zařízení pomocí GPS
2. Umět pracovat s bránou GSM, tj, vyslat polohu zařízení, které bylo získáno pomocí GPS
3. Měřit proud do I_{max} = 65 A
4. Měřit napětí do U_{max} = 40 V
5. Měřit otáčky motoru do 3000 ot^{-1}
6. Podle převodů, průměru kola vypočítat rychlost pohybu v m/s, případně km/h
7. Podle Vaší volby máte možnost si vybrat následující řešení:
   a) Průběžné vysílání naměřených dat, kde je potřeba navrhnout a vyrobit i funkční přijímač, ze kterého mohou být data načtena do PC.
   b) Ukládání dat do paměti a umožnění přenosu obsahu do PC včetně možnosti mazání obsahu pomocí tlačítka či jiného bezpečného zdroje signálu.

Poznámky k jednotlivým bodům:
Mikrokontrolérový systém může být i víceprocesorový, protože by možná svou rychlostí nestačil na požadované činnosti

1. Je třeba zjistit rychlost (frekvenci) a délku přijímaného telegramu z družice, který lze použít pro synchronizaci měření.
2. Při zaslání čísla e-mailem na zařízení, bude muset systém vyslat poslední údaj získaný z GPS.
3. Pro měření proudu použít bočník s odporem 0,01Ω, a protože bude motor řízen PWM či pulsně (změnou f), pak je třeba použít integrační článek a operační zesilovač, který provede úpravu rozsahu výstupního signálu (v tomto případě je to proudu úměrné napětí). Zpracování bude provedeno pomocí A/D převodníku v mikrokontroléru.
4. To samé bude potřeba pro měření napětí, které bude pulsní též, tzn. použít integrátor včetně OZ, pomocí kterého lze nastavit odpovídající výstupní signál. Zpracování bude provedeno pomocí A/D převodníku v mikrokontroléru.
5. Bude potřeba realizovat inkrementální snímač otáček motoru. Zatím neznáme konstrukci, takže kde a jak bude umístěn, bude rozhodnuto později. Jako vlastní snímač použijte optickou IR závoru, kde její výstup bude připojen k systému.
6. Není třeba komentář, stačí trocha matematiky
7. Je třeba si vybrat. Doporučení - použijte co nejjednodušší řešení, což je použití paměti o kapacitě cca 128 až 256 kB.

2.0 Úvod

Práce se zabývá vytvořením zařízení, které bude namontované na elektromobilu. Má plnit funkci sledovacího zařízení, zabezpečovacího zařízení, černé skříňky a také vyhodnocovat některé důležité informace pro správný chod elektromobilu.

Jelikož elektromobil, pro který je zařízení vyrobeno, jsme fyzicky neměli od počátku ve svých rukou, museli jsme spolupracovat se spoustou lidí. Hlavní požadavky na naše zařízení nebyly moc náročné, co se týče rozměrů, měli jsme volné ruce. Napájení jsme museli přizpůsobit pomocí měniče, pracujícího v rozsahu 30 – 42 V (což je napájení motoru z akumulátoru) na 8 a 12 V.

Jedna z konkrétních funkcí našeho zařízení je sledovat polohu GPS a ukládat ji do paměti. Pokud je aktivováno zabezpečení a u elektromobilu se změní poloha o více než 50 metrů, vyhodnotí se to jako krádež a následně bude poslána SMS zpráva skrze GSM bránu. Ve zprávě bude aktuální poloha vozidla, a pokud majitel zavolá zpět na toto číslo, pošle se mu okamžitě nová zpráva s aktuálními souřadnicemi. Co se hardwarové části maturity týče, museli jsme vypracovat měření otáček motoru, přes které následně budeme počítat rychlost vozidla. Práci jsme si rozdělili do 3 částí. První částí byla práce s GPS + programování čítače otáček a převodování meřeného napětí a proudu. Druhá část byla práce s GSM bránou, neboli modulem PGSM a navrhnutí desky pro moduly PGSM a PGPS. Třetí část byla čistě hardwarová a to sestrojení regulátoru pro napájení modulů a mikropočitače, snímač otáček, filtr bočníku pro měření proudu, který pak přes zesilovač půjde na vstup Arduina.

Elektrokolka

3.0 Koncepce řešení

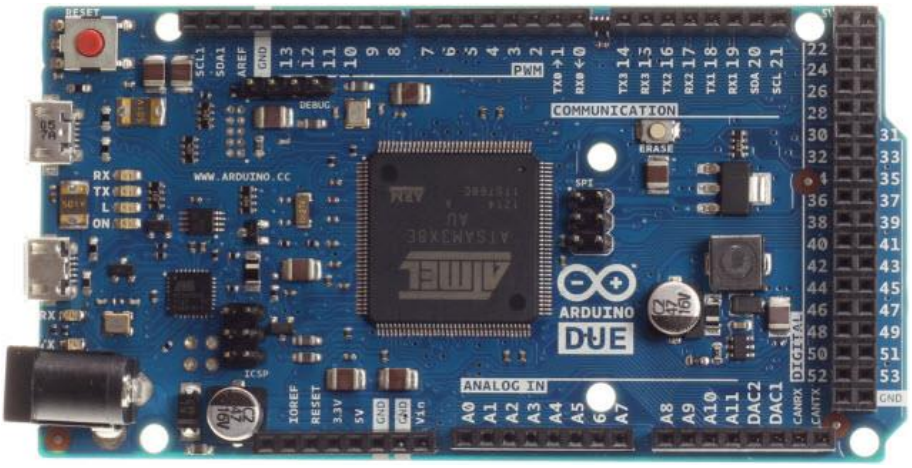
3.1 Použitá zařízení

3.1.1 Arduino

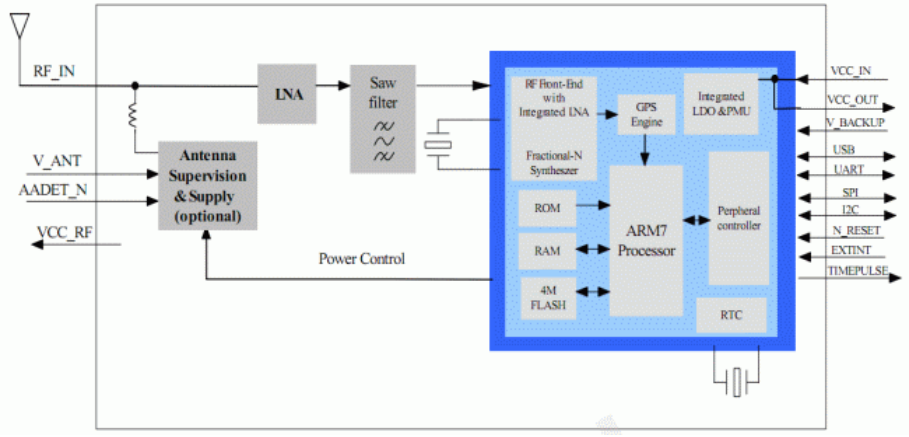
Arduino je open-source platforma pro snadný návrh a vývoj elektronických programovatelných zařízení založená na mikrokontroleru od fy ATMEL a grafickém vývojovém prostředí, které vychází z prostředí Processing.
Projekt Arduino si od počátku zakládá na své tvůrčí otevřenosti a vstřícnosti. Jsou k dispozici všechny zdrojové soubory a oficiální stránky, kde jsou napsané návody i vysvětlení. Výhodou zařízení Arduino je jednoduché připojení k počítači a programování pomoci jazyku odvozeného z Wiringu. Arduino Due, které jsme použili je bratříčkem nejznámějšího a nejpoužívanějšího Arduina Uno s mikroprocesorem ATMega328. Jedná se o první Arduino založené na 32 bitové architektuře, které je vybavené už procesorem Cortex M3 s označením AT SAM 3 vysokou taktovací rychlostí 84 MHz a nabídne nám vyšší výkon a lepší možnosti konektivity. Dále je vybaveno microUSB konektorem a po připojení k PC se hlásí jako sériový port, dvanáct ADC s 12bitovým rozlišením třeba pro audio a dva rovněž 12bitové DAC, pak čtyři COM, rozhraní CAN, dvanáct PWM kanálů, dva I2C a podpora protokolu Android ADK 2012. Deska je určena pro vstupní napětí 7 - 12 V a obsahuje 512 kB paměti Flash, 64 + 32 kB SRAM a také debugovací konektory JTAG / SWD.

Pro tuto práci jsme si vybrali vývojovou desku založenou na SAM3X8E ARM Cortex-M3, zvanou Arduino Due. Volba Arduino Due byla také podpořena tím, že jsme potřebovali velké množství vstupních a výstupních pinů. Jelikož je Arduino Due na trhu relativně krátce, většina knihoven, shieldů na něm nefungují. Což bychom viděli jako jediné mínus.

Co vlastně shieldy jsou? Když se chceme na stolním PC připojit k Wifi, většinou nemáme jinou možnost než koupit Wifi kartu. Když chceme poslouchat kvalitní hudbu, musíme připojit kvalitní zvukovou kartu. A stejně tak to má Arduino. Pokud například chceme, aby naše Arduino umělo sledovat GPS polohu, použijeme GPS shield, který nám mnohé usnadní. Je to součástka dělaná přesně tak, aby na Arduino pasovala. Stačí ji nasadit a nemusíme se starat o nějaké zapojování atd. Poté už je vše jen o programování. Bohužel touto cestou jsme se my vydat nemohli, jelikož bychom se nevešli do rozpočtu.



Obr. 1 Arduino DUE - deska



Obr. 2 Arduino DUE- schéma 1

3.1.2 PGPS-1 modul

Tento modul slouží k realizaci GPS. Moduly PGPS umožňují určení přesné polohy, nadmořské výšky, rychlosti a času. Pro naši práci jsme použili jeden z řady nových modulů, určených pro vývoj elektronických zařízení a malosériovou výrobu. PGPS je špičkový produkt L10 od společnosti Quectel, který jsme zakoupili v Pandatronu. Vynikající vlastností našeho modulu je obnovovací frekvence, která je až 5 Hz. Další vynikající vlastností je citlivost, která dosahuje maximální hodnoty až 165 dBm, rychlé určení polohy, rozsah provozních teplot: -40 až +85 °C, studený start: <35s, horný start: <1s, Vlastní spotřeba v aktivním režimu jen 38 mA, přesnost určení polohy: 3,0 m a jediné napájecí napětí 3,0 V až 4,3 V. Uvedené parametry platí pro L10, který je součástí PGPS-1. Je vidět, že modul svou citlivostí spadá do kategorie GPS přijímačů pro obecné použití. Výhodou je rovněž široký rozsah provozních teplot, který umožňuje jeho použití i v nepříznivých podmínkách či zařízeních, pracujících ve venkovním prostředí. K modulu jsme vybrali GPS anténu 2J431 s SMA konektorem. Pro naše vozidlo zjišťuje potřebné souřadnice polohy, čas a směr, přitom je jednoduchý a relativně levný. K modulu jsme vybrali anténu typu 2J431 (viz obrázek).

Obr. 3 PGPS-1 Modul

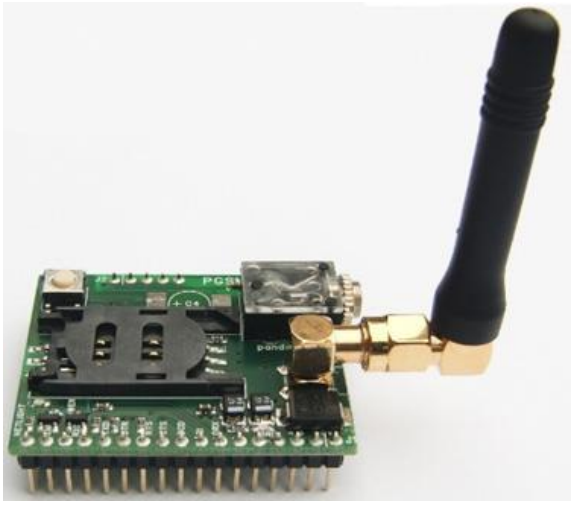
Obr. 4 Aktivní anténa 2J431

3.1.3 PGSM-1 modul

Jedná se o vývojový modul od firmy Pandatron, jenž je určen pro bezdrátovou komunikaci v GSM sítích. Modul PGSM je určen především pro vývoj, nebo malosériovou výrobu. Je navržen tak, aby práce s ním byla co nejjednodušší. Podporuje i hlasovou komunikaci, což v našem případě není potřeba. Přesto jsme vybrali tento modul. Na trhu je velký výběr, ale největším konkurentem byl modul od firmy Telit. Rozhodli jsme se pro Quectel protože jde o levnější variantu, která splní to samé.

Základem PGSM je kompletní GSM/GPRS modul M10 společnosti Quectel, což je stejný výrobce jako u druhého použitého modulu L10 PGPS-1.

Stabilizované napájecí napětí modulu PGSM se přivádí na první piny konektoru (pinové lišty) J1 a mělo by mít hodnotu maximálně 5V.



Obr. 5 Modul PGSM-1 s anténou 2J010

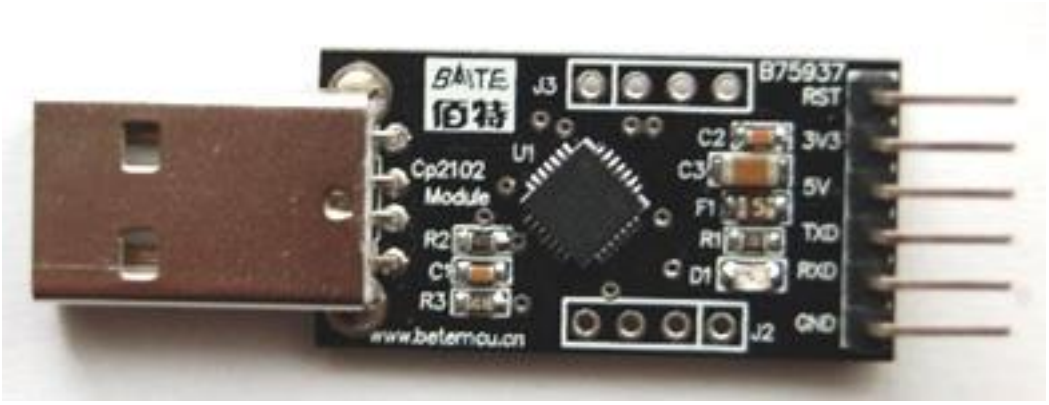
Obr. 6 Modul PGSM-1 pohled shora.

Na obrázku můžeme vidět Modul PGSM-1, rozmístění pinů, pouzdro pro SIM kartu. Pro naše potřeby budou stačit pouze 4 piny a totiž GND, 5V, RXD, TXD.

Na modulu si můžeme všimnout jacku 3,5mm který slouží pro hlasovou komunikaci. Modul se ovládá AT příkazy, kterým se budu věnovat níže v práci.

3.1.4 Modul CP2102

Pro první komunikaci jsme se rozhodli použít modul CP2102, pomocí kterého se dá realizovat sériový port z USB portu našeho PC. Základem modulu je pouze jediný integrovaný obvod, zmíněný CP2102. Ten je napájen přímo z USB portu počítače, prostřednictvím USB A konektoru. Na opačném konci modulu jsou dostupné piny se standardní roztečí 2,55 mm, které jsou kompatibilní například i s kontaktními poli. Dostupné jsou zde základní signály, jako je RXD a TXD, stejně jako vyvedené napájecí napětí +5V, +3,3V, GND a dokonce i RST.



Obr. 7 Modul CP2102 (převodník USB-UART)

3.1.5 Externí paměť EEPROM 24L256

EEPROM 256 Kbit, která jsou schopná pracovat v širokém rozsahu napětí (1.7V až 5.5V). Byla vyvinuta pro nízko výkonové aplikace, jako jsou osobní komunikace a sběry dat. Toto zařízení má také schopnost page zápisu až 64 bajtů dat a je schopné jak náhodné tak sekvenční čtení až k hranici 256 kB . Funkční adresy linky umožňují až osm zařízení na stejné sběrnici, pro až 2 Mbit adresního prostoru. K dispozici jsme měli 2 čipy.

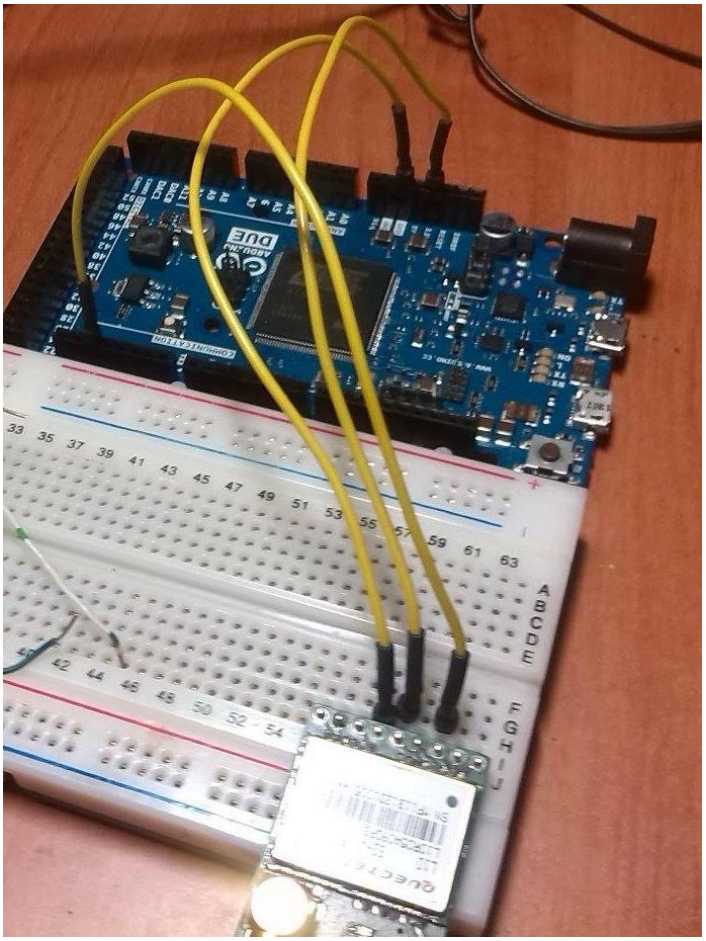
Obr. 8 Paměť 24LC256

3.2 Realizace

3.2.1 Seznámení se s Arduinem

Nejprve jsme se museli seznámit s Arduinem, jak s ním pracovat, jak ho zapojit a jak programovat. Prvním kódem, kterým jsme otestovali funkčnost zařízení a seznámení se s programovacím jazykem, který je podle autorů jazyk podobný C++, který známe, bylo rozblikání LED diody. Celkem rychle jsme se seznámili a mohli jsme se pustit do naší práce.

Začali jsme tím, že jsme propojili Arduino s GPS modulem zatím přes nepájivé pole a mohli jsme začít programovat. Dalším kódem jsme zjistili, že naše GPS zařízení zachytává formát NMEA vět z družice.



Obr. 9 Propojení GPS modulu s Arduinem DUE

V NMEA-0183 formátu jsou data posílána po řádcích. Každý řádek začíná znakem '$', následuje dvojpísmenná zkratka zařízení (GP = GPS) a dále trojpísmenný kód určující formát zprávy. Každý řádek pak končí hvězdičkou a hexadecimálně zapsaným kontrolním součtem (XOR všech znaků na řádku mezi '$' a '*'). Délka řádku je omezena na maximálně 80 znaků a jednotlivé položky jsou od sebe odděleny čárkami. Ze všech možných formátů NMEA vět nás zajímala GPRMC věta (základní informace o pozici).

3.2.2 RMC

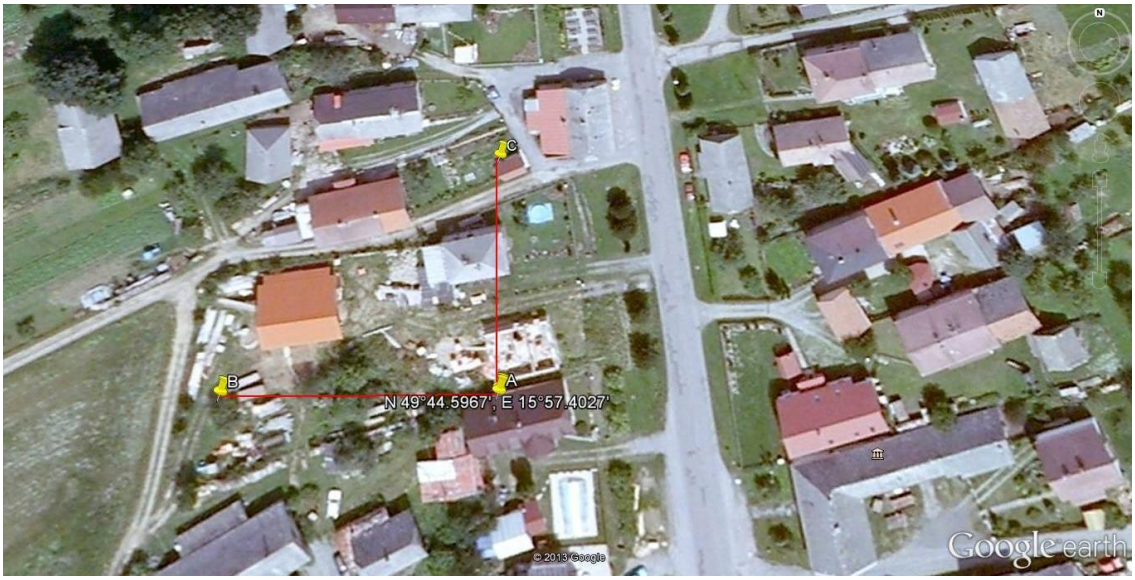
RMC je zkratka pro ***Recommended minimum specific GPS/Transit data*** a je to tedy jakési doporučené minimum, které by měla poskytovat většina GPS zařízení. RMC věty mohou vypadat takto:

$GPRMC,181038.0,A,5006.3171,N,01425.6622,E,0.00,42.00,150114,,*37

- 181038.0 - určuje čas, konkrétně 18:10:38.0 UTC
- A - status (A=active, V=void) platná či neplatná pozice
- 5006.3171,N - latitude: 50 stupňů, 6.3171 minut, severní šířky
- 01425.6622,E - longitude: 14 stupňů, 25.6622 minut, východní délky
- 0.00 - rychlost v námořních uzlech
- 42.00 - azimut ve stupních
- 150114 - datum 15. ledna 2014
- chybí záznam o magnetickém rozptylu
- a jeho směru (W nebo E)

Pro moji práci jsme potřebovali pouze 4 údaje: čas, severní šířku, východní délku a datum.

Déle jsme potřebovali zjistit, kdy bylo auto odcizeno, takže jsme si museli stanovit odchylku, o kolik se může zařízení posunout, aniž by se aktivovalo (tříkolka byla odcizena). Stanovili jsme si 50 metrů, protože v budově nám ukazoval GPS modul odchylku až 25 metrů.



Obr. 10 Zjištění 50 metrů na Google earth

Abychom nemuseli složitě počítat kolik minut, vteřin je pro českou republiku 50 metrů, rozhodli jsme se ulehčit si práci a použít program Google earth. Tímto jsme zjistili odchylku jak severní šířky, tak východní délky.

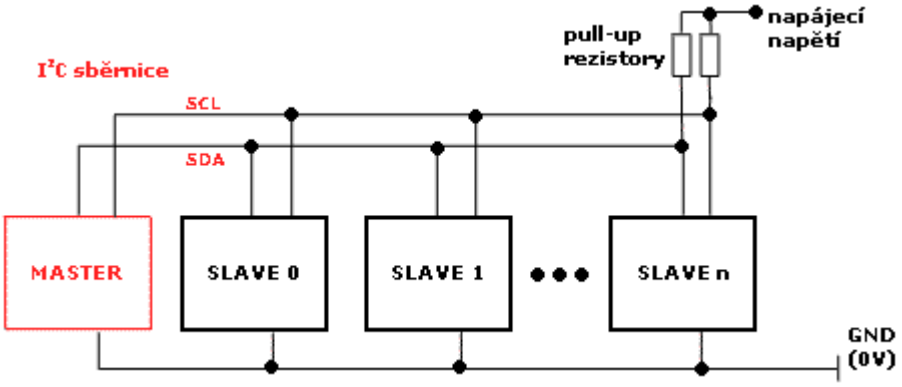
Jako další úkol jsem měl ukládat tyto hodnoty do paměti. Bohužel nastaly problémy.

Zjistili jsme, že Arduino Due nemá kompatibilních mnoho knihoven a není kompatibilní s žádným doposud existujícím shieldem. On totiž pracuje s napájením 3,3 V, zatímco ostatní Arduina pracují s 5 V. Ani knihovka EEPROM, kterou jsme potřebovali, nebyla kompatibilní s naším zařízením. Museli jsme tedy použít vnější paměť 24LC256. S ní jsme museli komunikovat přes I²C.

3.2.3 I²C

I²C je interní datová sběrnice sloužící pro komunikaci a přenos dat mezi jednotlivými integrovanými obvody většinou v rámci jednoho zařízení. Hlavní výhodou je, že obousměrný přenos probíhá pouze po dvou vodičích - "data SDA (serial data)" a "hodiny SCL (serial clock)". Na jednu sběrnici může být připojeno více integrovaných obvodů.

Možné propojení jednotlivých integrovaných obvodů ukazuje následující obrázek.



Obr. 11 I²C – možné připojení

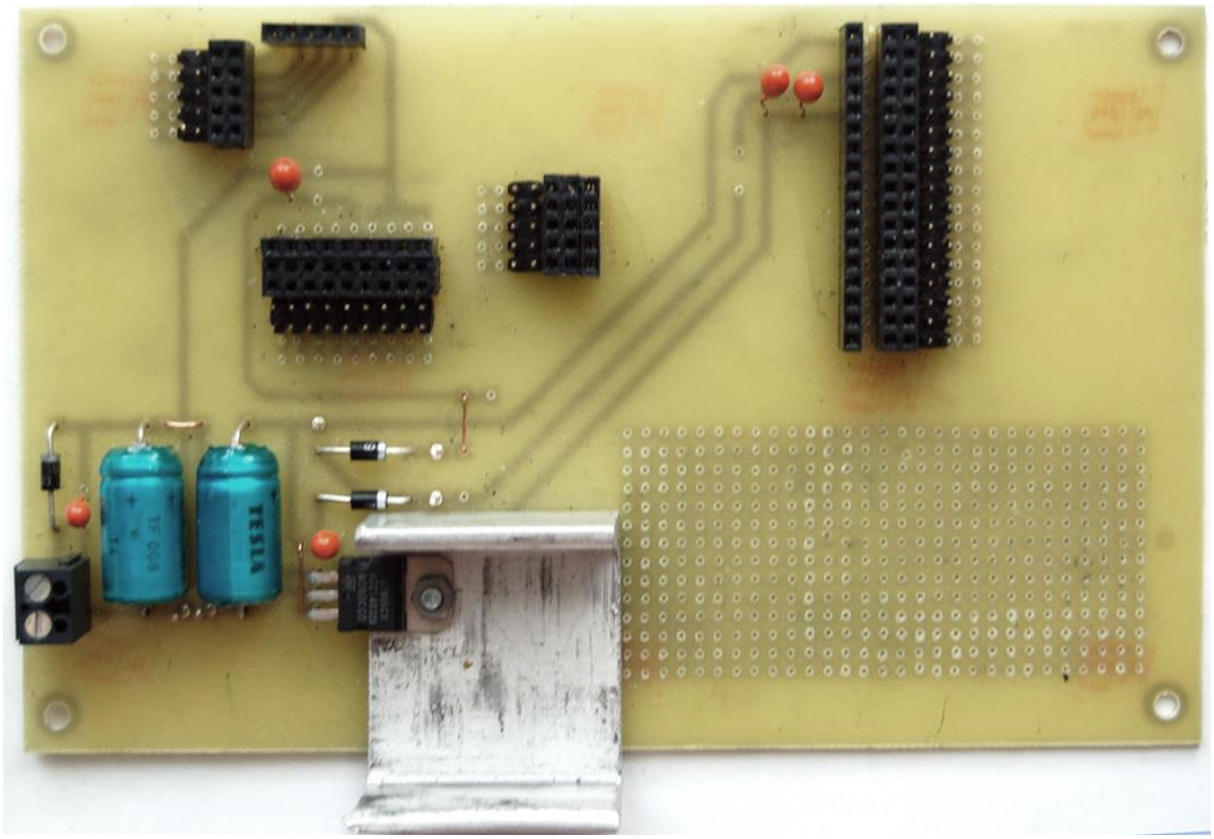
Jeden z integrovaných obvodů (většinou mikrokontrolér) je nastaven jako MASTER a všechny ostatní obvody jsou SLAVE. Master při jakémkoli přenosu generuje hodinový signál na vodiči SCL. Když jeden čip vysílá, přijímají všechny ostatní a pouze podle adresy určují, zda jsou data určena jim. Čip, který chce vyslat/přijmout data musí nejprve definovat adresu čipu, s kterým chce komunikovat a zda půjde o příjem nebo vysílání - tedy o čtení nebo zápis. To určuje R/W (read/write) bit, který je součástí adresy.

3.2.4 První plošný spoj

Při prvních pokusech jsme používali nepájivé pole, abychom si usnadnili práci a abychom nemuseli pájet. Poté, co jsme se seznámili a naučili pracovat se zařízením, jsme museli navrhnout desku, která měla sloužit jak pro práci s GPS, tak i GSM a na které lze lépe testovat naše zařízení, vzhledem k jeho špatným zkušenostem s nepájivými poli.

Jedná se o desku, na kterou se umísťují oba dva moduly, jak PGSM tak PGPS. Na desce je realizováno napájení pro oba dva moduly. Jsou to hodnoty 5V, 4,3V a 3,6V. Tyto hodnoty jsou realizovány díky úbytkům na polovodičových diodách. První testování desky

neproběhlo nejlépe. Díky laboratornímu zdroji jsme si všimli, že proud tekoucí přípravkem se nám exponenciálně zvětšoval. Tudíž jsme přípravek odpojili, a vzhledem k chování přípravku jsme si odůvodnili, že to bude chyba nějakého kondenzátoru. Jednalo se o banální chybu, a to otočenou polaritu u jednoho z kondenzátorů. Kondenzátor jsme odpájeli, otočili a vše fungovalo tak, jak má.



Obr. 12 První plošný spoj

Pro konečnou verzi jsme si museli navrhnout další plošný spoj, který by měl být co nejmenší a měl by být pro GPS modul, GSM bránu a pro paměti.

3.2.5 Komunikace přes CP 2102

Pro první komunikaci jsme se rozhodli použít modul CP2102, pomocí kterého realizuji sériový port z USB portu svého PC. Obvod CP2102 je napájen přímo z USB portu počítače, prostřednictvím USB A konektoru. Na opačném konci modulu jsou dostupné piny se standardní roztečí 2,55 mm, které jsou kompatibilní například i s kontaktními poli. Dostupné jsou zde základní signály, jako je RXD a TXD, stejně jako vyvedené napájecí napětí +5V, +3,3V, GND a dokonce i RST.

3.2.6 Práce s AT příkazy

Hardwarovou stránku jsme tedy měli vyřešenu, nyní jsme potřebovali nějaký terminál, přes který budu posílat data z PC přes Modul CP2102 až na PGSM modul. Zkoušeli jsme více programů, ale nakonec byl vybrán program pro Arduino, který relativně nově umožňuje připojit sériový monitor (viz obrázek). Nyní jsem se poprvé setkal s AT příkazy. AT příkazy jsou příkazy v ASCII kódu, které se pro GSM běžně používají. Mají svou ustálenou podobu. Tudíž jsem si vyhledal několik příkazů, které ve svém programu budu potřebovat.
Modul se dá aktivovat dvěma způsoby, a to přivedením logické jedničky na pin POWER_ON, nebo použít tlačítko přímo na modulu stlačením alespoň na 5 sekund.

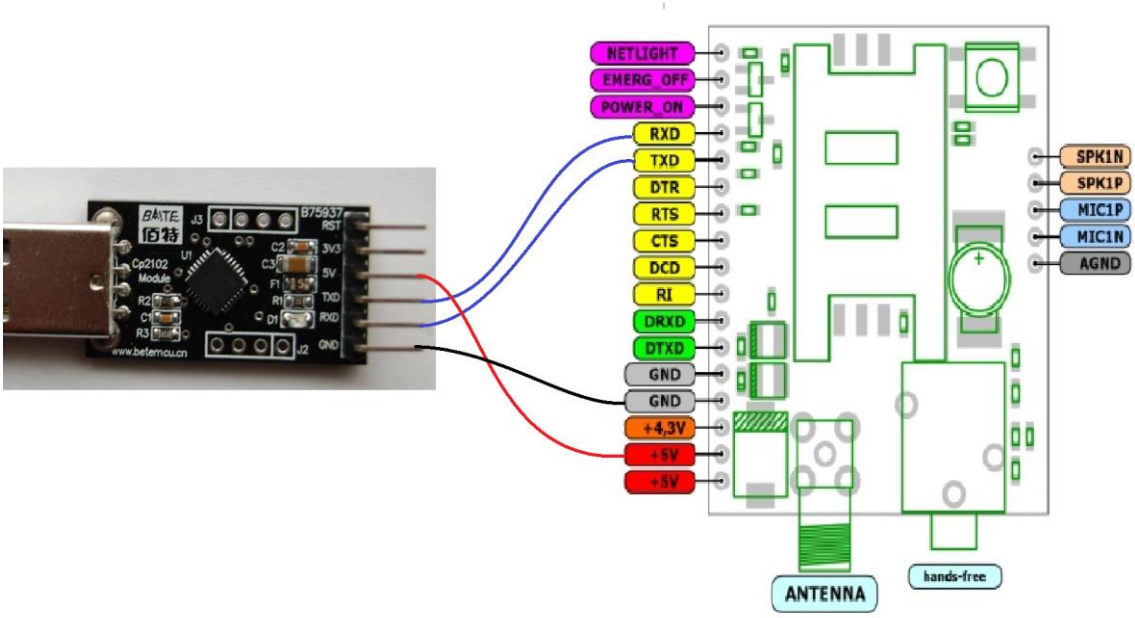
Jeden z nejjednodušších příkazů je samotné AT, na což by modul měl odpovědět OK, pokud je správně zapojený a vše je v pořádku. Dále budeme používat příkaz ATH, který pokládá buď příchozí, nebo odchozí hovor. Na druhou stranu pro zvednutí příchozího hovoru se používá ATA. Mezi další důležité bychom zařadili příkaz AT+CMGF, který přepíná GSM do různých módů. Konkrétně byl zvolen mód 1, který přepne GSM do SMS módu (tedy příkaz AT+CMFG=1). Vytočení telefonního čísla provedeme pokynem:

ATD00420XXXXXXXXX.

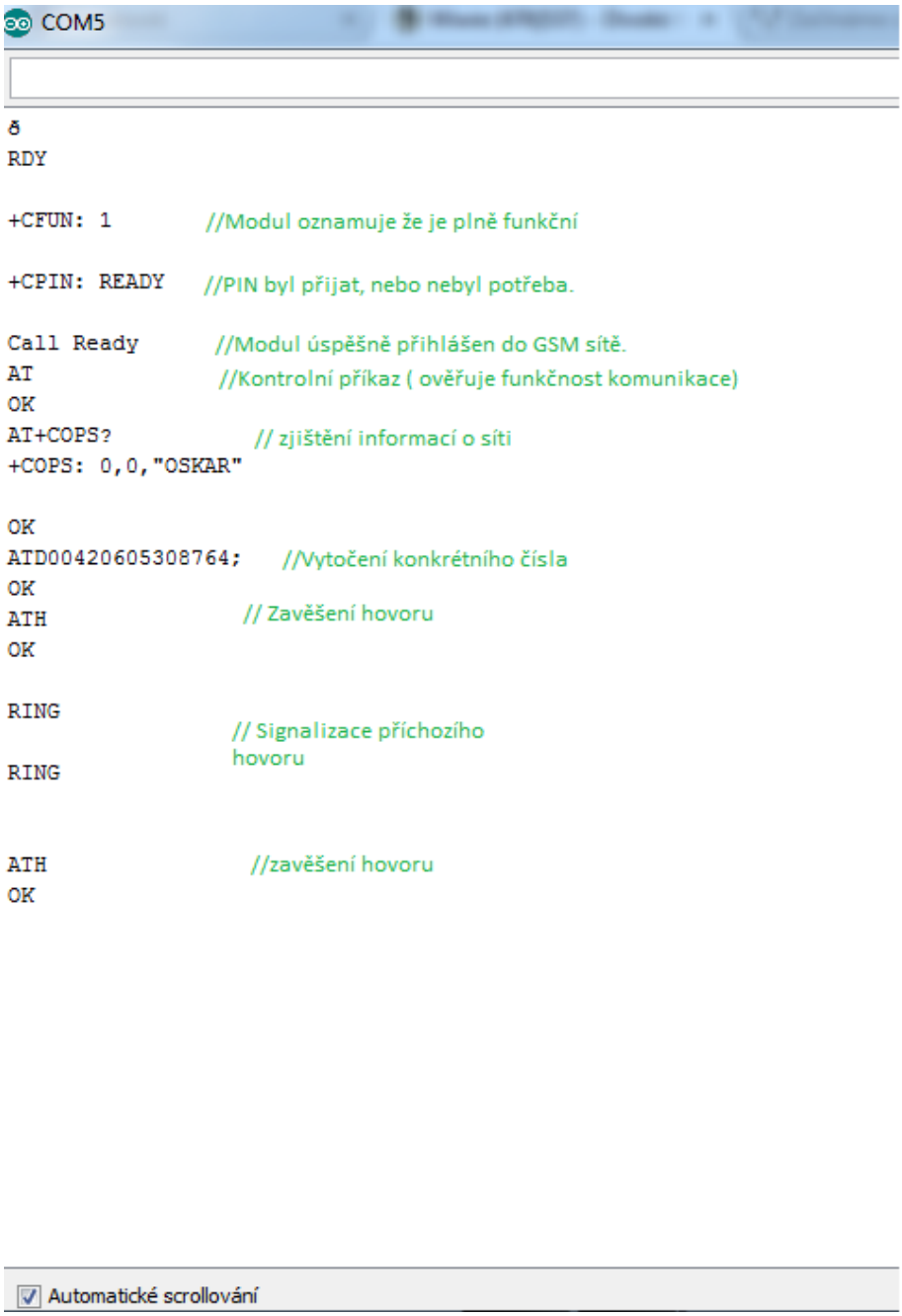
(za X dosazujeme telefonní číslo, před ním vidíme předvolbu).

Jak bylo výše zmíněno, modul na každý správně zadaný příkaz odpovídá jednoduše OK, což lze vidět v obrázku níže. Nyní si ukážeme jak propojit modul CP 2102 s modulem PGSM.

Pro naši práci budou stačit 4 piny, +5V, GND, RXD, TXD. Piny RXD a TXD slouží k sériové komunikaci a propojují se křížem. Tudíž pin RXD z modulu PGSM vede na pin TXD na modulu CP 2102. Níže se tedy podívejme na zapojení.



Obr. 13 Schéma zapojení modulu CP2102 a PGSM modulu



Obr. 14 Sériový monitor připojen přes program pro Arduino 1.0.5 - přímá komunikace s GSM modulem

Poté, co bylo vše správně zapojeno, byla realizována komunikace. Vyznat se v AT příkazech nebylo jednoduché, ale nakonec jsme si v nich udělali přehled. Tak jako každý telefon při zapnutí, i modul PGSM požaduje zadat pin. Ten byl SIM kartě zrušen, jelikož při testech bychom ho museli zadávat nespočetněkrát. Nicméně pokud bude pin potřeba zadat, řešitelné pomocí AT příkazu +CPIN:PIN_KÓD.

Při prvním zapojení se vyskytly první obtíže s prací. Modul neodpovídal, ačkoliv vše bylo zapojeno správně. Poté bylo zjištěno, že byl chybně nainstalován ovladač na modul CP 2102, tudíž reinstalace ovladače vše vyřešila. Poté byly prostudovány AT příkazy a celkově chování modulu PGSM.

Na obrázku 14 je uvedeno několik prvních použitých příkazů, přímo ze sériového monitoru. Obrázek je z programu Arduino 1.0.5, jedná se o nejnovější verzi programu.

3.2.7 Výsledný plošný spoj pro GPS a GSM

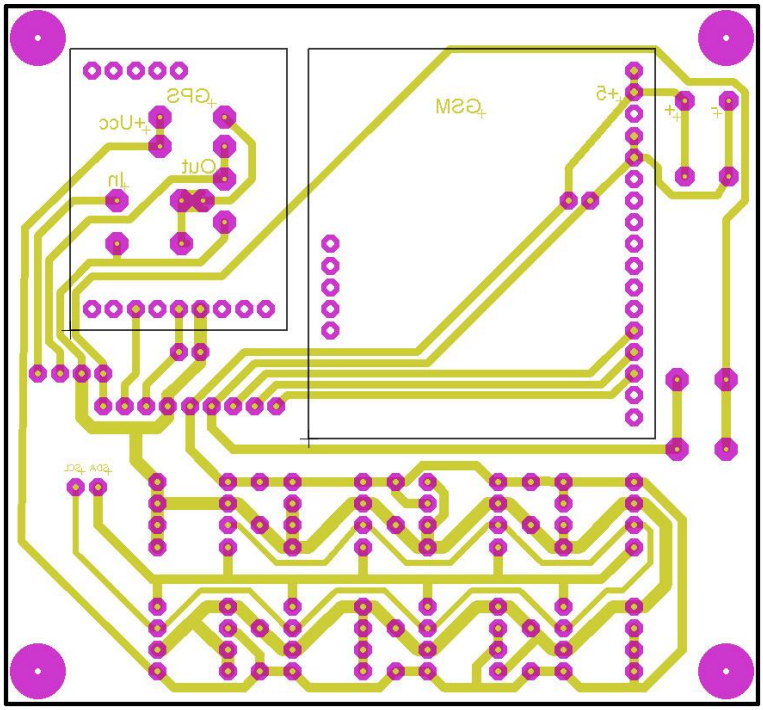
Nyní již byla práce funkční a museli jsme vytvořit plošný spoj pro oba dva moduly (PGSM, PGPS). Byly dva požadavky pro tento plošný spoj, a to aby byl co nejmenší, nejjednodušší. Proto jsme řešili napájení, zda má být na plošném spoji, či nikoliv. Zjistili jsme si odběry proudu modulů a ujistili jsme se, že nám bude stačit, když moduly budeme „živit“ přímo z Arduina.

Na desce plošného spoje můžeme vidět místo pro oba dva moduly a také ve spodní části desky je řešená paměť EEPROM. To byla jedna z dalších komplikací. Prve jsme předpokládali, že použijeme vnitřní EEPROM paměť Arduina DUE pro ukládání dat.

Bohužel, jelikož je Arduino DUE relativně novým procesorem, zatím nemá žádnou podporu pro tuto paměť. Tudíž jsme se rozhodli použít 24LC256, což je 32kB EEPROM paměť, která má 3 piny pro adresy. Z toho plyne, že jich je možné zapojit až 8.

Počet potřebných chipů jsme si lehce odvodili. Požadavek na zařízení byl, aby byl schopný ukládat data každou desetinu vteřiny, po dobu deseti minut minimálně. Tudíž budeme ukládat 6000x. Každé uložení do paměti by mělo obsahovat 1B pro rychlost, 1B pro naměřený proud, 2B pro napětí a přibližně 12B pro polohu. Tímto se dostaneme na zhruba 100kB. Proto jsme zvolili 8 pamětí – 8* 32kB = 256kB.

Na desce jsou vyvedeny z této paměti 2 vodiče, jedná se o komunikaci I^2C, jeden vodič realizuje frekvenci a druhý je pro komunikaci.



Obr. 15 Finální plošný spoj pro moduly PGSM + PGPS a 8 pamětí 24LC256

3.2.8 Program pro zpracování a vyhodnocení informací

Ze začátku byly menší potíže s programováním, jelikož jsme programovali zařízení s Arduinem poprvé. Brzy jsme to však zvládli. Jak vidíte, opravdu se podobá tento programovací jazyk jazyku C++, takže to pro nás byla veliká výhoda.

Zdrojový kód

```
#include <Wire.h> //I2C knihovna
 int SpravnyRadek=0;
 int pricitani=0;
 int pricitani1=0;
 int poprve=0;
 char retezec[300] = " ";
 char retezecGPRMC[7] = "$GPRMC";
 char data[10];
 char cas[10];
 char SSirka[9];
 char SSirka_prvni[9];
 char VDelka[10];
 char VDelka_prvni[10];
 char datum [6];
 bool poplach = false;
 bool prvni_smycka_S = false;
 bool prvni_smycka_V = false;
 int num = 0;
 int poc=0;
 int pocitani=0;
 bool jenJednouZaImpulz;
 unsigned long threshold;
 bool first=false;
 int timeTosend=1;
 int count=0;
 char tel_cislo[]="+420739043488";
 float RRychlost =0;

void eepromWriteBytes( int device, unsigned int address, byte* data1, byte length ) {
Wire.beginTransmission(device);
Wire.write((int)highByte(address));
Wire.write((int)lowByte(address));
byte r;
for ( r = 0; r < length; r++)
Wire.write(data1[r]);
Wire.endTransmission();
```

```
}
byte eepromReadByte( int device, unsigned int address ) {
byte rdata = 0xFF;
Wire.beginTransmission(device);
Wire.write((int)highByte(address));
Wire.write((int)lowByte(address));
Wire.endTransmission();
Wire.requestFrom(device,1);
if (Wire.available()) rdata = Wire.read();
return rdata;

}


void setup() {

  Serial1.begin(9600);
  Serial.begin(9600);
  Serial.println("AT+CMGF=1");
  Wire.begin();
  pinMode(48,OUTPUT);
  pinMode(41,INPUT);
  pinMode(30,INPUT);

  prvni_smycka_S=true;
  prvni_smycka_V = true;
  jenJednouZaImpulz = false;
  threshold = millis() + 100;

  for (int i=0;i<300;i++)
  {
     retezec[i]=' ';     //vyprázdění řeteztec před použitím
  }
}
void loop()
{


 if (digitalRead(48)== HIGH)
 {

  if(Serial1.available())
  {
    digitalWrite(13, HIGH);
```

```
Serial.read();

int c = Serial1.read();       // čtení přijímaných dat

retezec[pricitani]=c;       // ukládání dat do řetězce
pricitani++;
SpravnyRadek = 0;

 if (c == 13)
 {
    for (int i=0;i<7;i++)
    {
      if (retezecGPRMC[i-1]==retezec[i])      //zjištění, zda jsme zachytili správnou větu
      {
        SpravnyRadek++;
      }
    }

  pricitani1=0;
  if(SpravnyRadek==6)
  {
   for (int i=0;i<300;i++)
   {
    if (retezec[i] == ',')      //rozdělení údajů podle ","
    {
      data[pricitani1] = i;
      pricitani1++;
    }
   }

  int cas_pocet =0;
  int sirka_pocet =0;
  int sirka_prvni_pocet=0;
  int vdelka_pocet =0;
  int vdelka_pocet_prvni =0;
  int datum_pocet =0;
   for (int i=0;i<10;i++)
   {
    switch(i)
    {
     case 0 :
        for (int j=data[i];j<(data[i+1]-1);j++)
        {
          cas[cas_pocet]=retezec[j+1];       //ukladaní dat čas
```

```
            cas_pocet++;
          }
        break;
        case 2 :
          for (int j=data[i];j<(data[i+1]-1);j++)
          {

           if(prvni_smycka_S)
           {
            SSirka_prvni[sirka_prvni_pocet] =retezec[j+1];  // severní šířka (první cyklus)
            sirka_prvni_pocet++;
           }
           else
           {
            SSirka[sirka_pocet]=retezec[j+1];                  // severní šířka
            sirka_pocet++;
           }
          }
          prvni_smycka_S = false;
        break;
        case 4 :
          for (int j=data[i];j<(data[i+1]-1);j++)
          {
           if(prvni_smycka_V)
           {
            VDelka_prvni[vdelka_pocet_prvni] =retezec[j+1];  //východní délka (první
cyklus)
            vdelka_pocet_prvni++;
           }
           else
           {
            VDelka[vdelka_pocet]=retezec[j+1];                  //východní délka
            vdelka_pocet++;
           }
          }
         prvni_smycka_V = false;
        break;
        case 8 :
          for (int j=data[i];j<(data[i+1]-1);j++)
          {
           datum[datum_pocet]=retezec[j+1];        //datum
           datum_pocet++;
          }
        break;
```

```
          }

         Serial.write("\n");
         poprve++;

       }

        for (int i=0; i<9;i++)
        {
         if (i==6)
         {
          if (SSirka_prvni[i]!=SSirka[i])
          {
           if ((SSirka_prvni[i]- SSirka[i])>= 2 || (SSirka_prvni[i]- SSirka[i])<= -2 )  //odchylka
šířky
           {
            if (first == true)
              poplach = true;
           }
          }
         }

        }
        for (int i=0;i<10;i++)
        {
         if (i== 7)
         {
          if (VDelka_prvni[i] != VDelka[i])
          {
           if(VDelka_prvni[i]- VDelka[i]>= 3 || VDelka_prvni[i]- VDelka[i]<= -3)
//odchylka délka
           {
            if (first == true)
             poplach = true;
           }
          }
        } }

       Serial.write ("cas:");
      for (int i=0; i<9; i++)              // výpis (slouží k odzkoušení)
      {
       Serial.write(VDelka_prvni[i]);
      }
```

```
 Serial.write ("cas1:");
for (int i=0; i<9; i++)              // výpis (slouží k odzkoušení)
{
  Serial.write(VDelka[i]);
}


 if (first == true)
 {
 if ( poplach== true)
 {
   while(count<timeTosend)
    {

      Serial.print("AT+CMGS=\"");
      Serial.print(tel_cislo);
      Serial.println("\"");

      while(Serial.read()!='>');
      {
        Serial.print("text k odeslání");
        //Sleep(500);
        Serial.write(0x1A);
        Serial.write(0x0D);
        Serial.write(0x0A);

      }
     count++;
    }

 if (poplach == true)
 Serial.write("poplach");


for (int i=0; i<300;i++)
    {

       delay(10);
    }
 }
 }

}
 pricitani =0;
```

```
      for (int i=0;i<300;i++)
            {
              retezec[i]=' ';
            }

   }
    first = true;
    poplach = false;

 }
}

else if (digitalRead(48)==LOW)
{

 int vysledna =0;

 if(((digitalRead(41)== HIGH)) && (jenJednouZaImpulz== false))
 {
  pocitani++;
  jenJednouZaImpulz = true;
 }
 if((digitalRead(41)== LOW))
  jenJednouZaImpulz = false;

   if (threshold <= millis())
   {
     RRychlost = (pocitani*61)/100;
 //   Serial.println (pocitani);
     pocitani=0;
     threshold += 100;
   }


 float Odpor = 0.005;
 float Proud;
 int interval=50;
 int mereni = analogRead(A0);
 float volty = mereni * (5.0 / 1023.0);
 Proud = volty/Odpor/9;
 delay(950);
```

```
String str;
char cstr[]= ("retezec");
char somedata_1[10];
char RESET0VANI_vnejsi_Pameti[]=("mazani");
  str = String(num);
  str.toCharArray(cstr,30);

  String str1;
  char proud_char[]= ("retezec");
  str1 = String(Proud);
  str1.toCharArray(proud_char,30);

  String str2;
  char napeti_char[]= ("retezec");
  str2 = String(volty);
  str2.toCharArray(napeti_char,30);

  String str3;
  char Rychlost[]= ("retezec");
  str3 = String(RRychlost);
  str3.toCharArray(Rychlost,30);



char carka[] =("/");

char myBigArray[300];
myBigArray[0] = '\0';
strcat(myBigArray, cstr);
strcat(myBigArray, carka);
strcat(myBigArray, Rychlost);
strcat(myBigArray, carka);
strcat(myBigArray, proud_char);
strcat(myBigArray, carka);
strcat(myBigArray, napeti_char);


 num++;

 for(int i=0;i<299;i++)
 {
  somedata_1[i] = cas[i];
 }
```

```
char somedata_2[] = "data z eeprom 2";


eepromWriteBytes(0x50, 0, (byte *)myBigArray, sizeof(myBigArray));
delay(10);

eepromWriteBytes(0x51, 0, (byte *)myBigArray, sizeof(myBigArray));
delay(10);
int addr = 0; //prvni adresa
byte b = 0;

do
{
b = eepromReadByte(0x50, addr);
if (b!=0) Serial.print((char)b);
addr++;
} while (b!=0);
delay(10);
addr = 0;
b = 0;
do
{
b = eepromReadByte(0x51, addr);
if (b!=0) Serial.print((char)b);
addr++;
} while (b!=0);
Serial.println(" ");
delay(20);

if (digitalRead(30)== LOW)
{
  Serial.write("mazani");
  for (int i=0;i<interval;i++)
  Serial.write("0\n");num=0;
  Serial.write("vymazano\n");
}

 }
 else Serial.println("0");
}
```

3.2.9 Program pro řízení komunikace

Tato část se zabývá pouze komunikací s GSM bránou. Ale na rozdíl od GPS brány, která pouze posílá data do Arduina, jsme museli komunikovat i opačně. Proto je zde předveden první úsek kódu, který se zabývá odesláním polohy v případě krádeže. Tato část kódu je dále vložena v předchozím řetězci, jenž vyhodnocuje, zda bylo vozidlo ukradeno. Programování v tomto jazyce není složité. Jsou zde 3 různé bloky, které si ukážeme na prázdném bloku v programu Arduino 1.0.5.

```
// Před funkcí setup je místo na deklarování globálních proměnných. Které bude program znát jak v setupu, tak ve smyčce neboli loop.

void setup()
 {
// část kódu, která se spustí pouze jednou a to při zapnutí arduina.

}

void loop()
{
 // Smyčka, která se bude opakovat neustále, dokud se arduino nevypne.

}
```

Zdrojový kód

```
int pocet_odeslani=1;      // Pomocná proměnná, slouží k nastavení množství SMS.
int pricitani=0;                  // Porovnáváme s počet_odeslání.
char tel_cislo[]="+420725043806";     // Nastavení telefonního čísla pro odesílání SMS
void setup()
{
Serial.begin(9600);             // Zapnutí sériové komunikace s rychlostí 9600 baudů.
delay(2000);                          // Zdržení o 2 sekundy, pro práci procesoru.
Serial.println("AT+CMGF=1");              // Nastavení GSM brány do režimu SMS.
delay(200);                           // vyčkání 0,2 Sekundy.
}
void loop()
{
while(pricitani<pocet_odeslani)       // Podmínka, aby se SMS neodesílala dokola.
{
  delay(1500);                        // Zdržení o 1,5 sekundy.
  Serial.print("AT+CMGS=\"");       // Poslání na sériovou linku AT příkazu na odeslání SMS
```

```
 Serial.print(tel_cislo);                // Předání modulu GSM telefonního čísla
 Serial.println("\"");                   // Ukončení příkazu SMS.
  while(Serial.read()!='>');             // Čtení sériové linky, dokud se neobjeví znak„>“
   {
     Serial.print("text k odeslání");  // Zapisování textu který bude v SMS.
     delay(500);                         // Zdržení o 0,5 sekundy.
     Serial.write(0x1A);                 // Potvrzení odeslání SMS
     Serial.write(0x0D);
     Serial.write(0x0A);

     delay(5000);                        // Vyčkání 5 sekund.
    }
    pricitani++;                         // k pricitani se přičte jednička.
 }
}
```

Na programování nebylo nic extra složitého, největší zádrhel byl u potvrzení odeslání SMS (jedná se o blok kódu, zvýrazněn červeně). Našli jsme však, že se realizuje pomocí zkratky CTRL+Z. Naše první kroky byly, že jsme se snažili zjistit jak CTRL+Z realizovat hexadecimálně. Na což jsme přišli, že můžeme vyjádřit jako 1A. Bohužel, i když jsme na sériovou linku poslali 1A, SMS se neodeslala. Jako další krok nás napadlo udělat kontrolní program, který pouze odešle jednoduchý AT příkaz na GSM bránu a bude ukládat vše, co se odeslalo a přijalo, a následně jej vypisovat. Díky tomu jsme přišli na to, že za každým příkazem jsou 2 ASCII znaky. A to CARRIAGE RETURN (0D) a LINE FEED (0A). Tudíž jsme tyto dva znaky také poslali na sériovou linku a vše fungovalo tak, jak má.

3.2.10 Další řešené části

Bylo řešeno řízení elektromotoru pomocí „bezeztrátového řízení PWM“, včetně dalších náležitostí, jako je měnič napětí 36V/8 a 12 V, snímač otáček motoru, snímání pulsního proudu (až 60 A pomocí bočníku) a jeho převod na střední hodnotu a jiné záležitosti včetně celého ovládání. Tyto části jsou zde jen takto zmiňovány, protože rozsah práce byl velký, zveřejňujeme jen výše uvedenou část.

4.0 Závěr

Přestože se jednalo o náročnou práci, řešení se podařilo a je po dílčích částech ověřeno. Vzhledem k času není však zatím ověřena jen elektromagnetická kompatibilita.

Jelikož byla nutnost navrhnutí plošného spoje pro moduly, museli jsme se naučit pracovat v programu Eagle.

Také s platformou Arduino jsme pracovali poprvé, museli jsme se naučit programovací jazyk, který Arduino obsluhuje. Práce v týmu byla velmi poučná, avšak chvílemi i dosti náročná. Každopádně dále jsme se naučili pracovat s AT příkazy a též řadu věcí o silnoproudé

elektrotechnice, která byla pro nás prakticky úplně nová, neboť s proudy kolem 60 A jsme se doposud nesetkali.

5.0 Použité zdroje

http://www.arduino.cc

http://www.linuxsoft.cz/article.php?id_article=1881

http://www.root.cz/clanky/arduino-jak-pro-nej-zacit-programovat/

www.pandatron.cz

http://pandatron.cz/?2868&pgps_-_gps_moduly_pro_vyvoj

http://www.quectel.com/

http://www.josefnav.cz/

http://www.dhservis.cz/dalsi/at_prikazy.htm

http://kellycontroller.com/kds24050e50a12v-24v-mini-brushed-controller-p-72.html

Skripta – Embedded systémy - Hrázský - 2012

Kybernetika – Hrázský – 2012

Dílenská technika – Hrázský – 2012

Učební texty Elektronika – Knapp 2012