

Embedded systém pro GPS navigaci

Embedded System for GPS Navigation

Juraj Štefan

Bakalářská práce
2014



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2013/2014

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Juraj Štefan**
Osobní číslo: **A11169**
Studijní program: **B3902 Inženýrská informatika**
Studijní obor: **Informační a řídicí technologie**
Forma studia: **prezenční**

Téma práce: **Embedded systém pro GPS navigaci**

Zásady pro vypracování:

1. Vytvořte rešerši na téma GPS navigace robotických systémů.
2. Navrhněte aplikaci pro jednočipový počítač s operačním systémem linux pro komunikaci s GPS moduly na základě protokolu NMEA.
3. Aplikaci doplňte o možnosti navigace a zaznamenávání aktuální polohy na SD kartu.
4. Definujte protokol pro externí komunikaci s aplikací pomocí sériové linky.
5. Vytvořte desktopový interface pro komunikaci s aplikací vytvořenou v jednočipovém počítači.
6. Všechny součásti práce publikujte pod licencí GPLv2.

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: tištěná/elektronická

Seznam odborné literatury:

1. PRATA, Stephen. Mistrovství v C++. 3. aktualiz. vyd. Překlad Boris Sokol. Brno: Computer Press, 2007, 1119 s. ISBN 978-80-251-1749-1.
2. BARR, Michael. Programming embedded systems in C and C. 1st ed. Sebastopol, Calif.: O'Reilly, c1999, xvii, 174 p. ISBN 15-659-2354-5.
3. MATLOFF, Norman S. The art of debugging with GDB, DDD, and Eclipse. San Francisco: No Starch Press, c2008, xiv, 264 s. ISBN 978-1-59327-174-9.
4. MASTERS, Jon a Richard BLUM. Linux profesionálně: programování aplikací. Vyd. 1. Brno: Zoner Press, 2008, 539 s. ISBN 978-80-86815-71-8.
5. RAPANT, Petr. Družicové polohové systémy. Vyd. 1. Ostrava: Vysoká škola báňská - Technická univerzita, 2002. ISBN 80-248-0124-8.

Vedoucí bakalářské práce:

Ing. Michal Pikner
Ústav řízení procesů

Datum zadání bakalářské práce:

28. února 2014

Termín odevzdání bakalářské práce:

13. června 2014

Ve Zlíně dne 28. února 2014



prof. Ing. Vladimír Vašek, CSc.
děkan



prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

ABSTRAKT

Táto práca pojednáva o vytvorení GPS navigácie pre robotické systémy, ktorá bude schopná prijímať dáta z GPS prijímača, obsahujúce aktuálnu polohu, a na základe nich a vytvorenej trasy diktovať smer a uhol pootočenia.

Teoretická časť rozoberá spôsoby navigácie pomocou družíc, usmerňovanie pohybu, vysvetľuje fungovanie systému GPS a popisuje vety protokolu NMEA. Na konci rozoberá základné poznatky o embedded systémoch.

Obsahom praktickej časti je popis použitého hardvéru. Hlavnou témou tejto časti je návrh systému. Popisuje návrh a vývoj desktopovej aplikácie, rovnako aj aplikácie GPS navigácie.

Kľúčové slova: poloha, navigácia, GPS, embedded systém

ABSTRACT

This thesis discusses the creation of GPS navigation for robotic systems that will be able to receive data containing the current position from the GPS receiver, and based on that and the route, to dictate the direction and angle of rotation.

The theoretical part discusses ways of satellite navigation, direction of movement, explains the operation of the GPS system and describes the NMEA protocol sentences. At the end is discussed basic knowledge about embedded systems.

Practical part contains a description of the used hardware. The main theme of this part is the design of the system. There is described the design and development of desktop application, as well as GPS navigation application.

Keywords: location, navigation, GPS, embedded system

PodĎakovanie

V prvom rade veľkú vďaku si zaslúži pán Ing. Peter Janků za všetky rady a pripomienky, ktoré mi boli nápomocné pri vypracovaní tejto práce, a tiež za vytvorenie podmienok potrebných k vypracovaniu tejto práce. Vďaku si zaslúži aj vedúci práce, Ing. Michal Pikner, a celý tím doktorandov z laboratória VTP/B237, ktorí mi svojimi cennými radami pomáhali pri vypracovaní. V neposlednom rade by som rád poďakoval mojej rodine a priateľom, ktorí mi boli celý čas oporou.

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....
podpis diplomanta

OBSAH

ÚVOD.....	9
I TEORETICKÁ ČASŤ.....	10
1 URČOVANIE POLOHY A NAVIGÁCIA POMOCOU DRUŽÍC	11
1.1 PRINCÍPY RÁDIOVÉHO URČOVANIA POLOHY	11
1.1.1 Určovanie polohy uhlomerným meraním	11
1.1.2 Určovanie polohy na základe dopplerovského merania.....	12
1.1.3 Určovanie polohy na základe merania vzdialeností.....	12
1.1.4 Určovanie polohy na základe fázového merania.....	13
2 URČOVANIE SMERU	15
3 GPS	16
3.1 OBECNÁ ŠTRUKTÚRA DRUŽICOVÝCH POLOHOVÝCH SYSTÉMOV.	16
3.1.1 Kozmický segment.....	16
3.1.2 Riadiaci segment	16
3.1.3 Užívateľský segment	17
3.2 SIGNÁLY VYSIELANÉ DRUŽICAMI	17
3.2.1 Základná frekvencia	18
3.2.2 C/A-kód.....	18
3.2.3 P-kód	19
3.2.4 Y-kód.....	19
3.2.5 Navigačná správa	19
3.3 URČOVANIE POLOHY	20
3.3.1 Kódové meranie	21
3.4 PRESNOSŤ MERANIA	21
3.5 GPS PRIJÍMAČ	22
4 NMEA 0183.....	24
4.1 OBECNÝ FORMÁT	24
4.1.1 GSA.....	24
4.1.2 RMC.....	25
4.1.3 GSV.....	25
4.1.4 GGA	26
5 EMBEDDED SYSTÉMY.....	27
5.1 REAL-TIME OPERAČNÉ SYSTÉMY	27
5.2 EMBEDDED OPERAČNÉ SYSTÉMY	27
5.2.1 Embedded Linux	27
5.3 SOFTVÉR PRE EMBEDDED SYSTÉMY	27
5.3.1 Krížová kompilácia	27
5.3.2 Debuggovacie nástroje	28
II PRAKTICKÁ ČASŤ	29
6 POUŽITÝ HARDVÉR.....	30
6.1 GPS MODUL	30
6.2 MIKROPOČÍTAČ COLIBRI T30.....	31
7 NÁVRH SYSTÉMU	33

7.1	DESKTOPOVÁ APLIKÁCIA	33
7.1.1	Použité technológie	33
7.1.1.1	Volanie funkcií napísaných v Javascripte v Java kóde.....	35
7.1.1.2	Volanie funkcií napísaných v Jave v JavaScript kóde.....	35
7.1.2	Užívateľské rozhranie	36
7.1.2.1	Práca so súbormi	37
7.1.2.2	Tabuľka súradníc a jej obsluha	37
7.1.2.3	Rozhranie pre vkladanie značiek	38
7.1.2.4	Rozhranie pre sériovú komunikáciu	38
7.1.2.5	Mapa	38
7.2	GPS NAVIGÁCIA.....	38
7.2.1	Štruktúry knižnice nav.h	38
7.2.1.1	Štruktúra navigation.....	38
7.2.2	Funkcie knižnice nav.h.....	39
7.2.2.1	Funkcia na_radiany()	39
7.2.2.2	Funkcia na_stupne().....	39
7.2.2.3	Funkcia distance().....	39
7.2.2.4	Funkcia azimuth()	40
7.2.2.5	Funkcia nmeaPOS_to_angle_in_deg()	40
7.2.2.6	Funkcia set_port()	40
7.2.2.7	Funkcia set_mode().....	40
7.2.2.8	Funkcia mode_addtrack()	41
7.2.2.9	Funkcia mode_nav()	41
7.3	DEMONŠTRÁCIA SYSTÉMU.....	42
7.3.1	Proces tvorby trasy a jej odoslanie mikropočítaču.....	42
7.3.2	Proces navigácie.....	44
ZÁVER		47
ZOZNAM POUŽITEJ LITERATÚRY		48
ZOZNAM POUŽITÝCH SYMBOLOV A SKRATIEK.....		50
ZOZNAM OBRÁZKOV		52
ZOZNAM TABULIEK		53
ZOZNAM ZDROJOVÝCH KÓDOV		54
ZOZNAM PRÍLOH.....		55

ÚVOD

Medzi robotické systémy sa už dávno neradia len roboty v priemysle, ktoré svoje polohu takmer nemenia alebo sa pohybujú len po presne určenej trase, ale patria sem napríklad aj roboty slúžiace hlavne pre vedu, výskum a armádne účely. Tieto vykonávajú obhliadku neznámeho terénu alebo aj obhliadku známeho terénu, ktorý je napríklad nebezpečný alebo neprístupný pre človeka, alebo iné mobilné roboty. Pri týchto druhoch robotických systémov vzniká potreba určovania polohy a navigácie v priestore.

Cieľom práce je návrh a vývoj aplikácie v jazyku C pre mikropočítač, so systémom Linux, ktorá bude slúžiť ako GPS navigácia pre robotické systémy. Práca popisuje štruktúru a funkcie knižnice nav.h a ich použitie v aplikácii. Aplikácia bude schopná komunikovať s GPS modulom na základe protokolu NMEA 0183. Aby bolo možné navrhnuť a vyobraziť trasu na mape, je vyvinutá desktopová aplikácia v programovacom jazyku Java, ktorá je schopná posilať vytvorenú trasu cez sériovú linku do mikropočítača. Vyvinuté aplikácie budú ako celok tvoriť jednoduché rozhranie, ktoré bude možné ďalej integrovať do robotických systémov.

I. TEORETICKÁ ČASŤ

1 URČOVANIE POLOHY A NAVIGÁCIA POMOCOU DRUŽÍC

K určovaniu polohy a k navigácii možno využívať rôzne fyzikálne princípy a na nich založené systémy.

Rádiové navigačné systémy sú tvorené sieťou vysieláčov vysielajúcich navigačné signály a užívateľskými zariadeniami, ktoré na základe spracovania a vyhodnotenia prijímaných signálov určujú aktuálnu polohu objektu. V prípade družicových navigačných systémov sú vysieláče reprezentované družicami a užívateľské zariadenia GPS prijímačmi. [1]

1.1 Princípy rádiového určovania polohy

Pri určovaní polohy pomocou rádiových signálov možno využiť niektorú z nasledujúcich metód:

- metóda uhlomerná,
- metóda dopplerovská,
- metóda diaľkomerná,
- metóda založená na meraní fázy nosnej vlny.

Tieto metódy umožňujú určovať polohu prijímača v dvojrozmernom prípadne aj v trojrozmernom priestore. Základným predpokladom je, že pozorovateľ pozná presnú polohu vysieláča. [1]

1.1.1 Určovanie polohy uhlomerným meraním

Metóda určovania polohy uhlomerným meraním je založená na princípe, kedy z bodu, ktorého chceme určiť polohu, zmeriame pomocou smerovej antény:

- buď azimuty k niekoľkým vysieláčom umiesteným na povrchu Zeme,
- alebo elevačné uhly k niekoľkým družiciam,
- prípadne elevačné uhly opakovane k jednej družici, ale s časovými odstupmi.

Pri pozemných vysieláčoch sa poloha určí jednoducho. Do mapy vynesieme u každého vysieláča priamku s odpovedajúcim nameraným azimutom. Tieto priamky sa pretnú v mieste navigačného prijímača.

V prípade družíc je postup o niečo zložitejší. Najprv musíme vypočítať polohu družice v okamihu merania elevačného uhla. Spojnica tohto bodu so stredom zemegule definuje os

kužela s vrcholom v mieste družice, jeho plášť je tvorený všetkými priamkami, ktoré prechádzajú družicou pod meraným elevačným uhlom. Ak skonštruujeme takéto kužele pre všetky merania a nájdeme ich priesečnice so zemským povrchom, v ktorom sa hľadaný bod nachádza, tak sa všetky tieto priesečnice pretnú práve v hľadanom bode. [1]

1.1.2 Určovanie polohy na základe dopplerovského merania

Táto metóda je využívaná predovšetkým pri meraní rádiových signálov vysielaných družicami. K určovaniu polohy bodu využíva Dopplerovho posunu (zmena frekvencie signálu vysielaného pohybujúcim sa objektom). Družica vysielala signál o známej konštantnej frekvencii f_p . Na tomto signáli sú prenášané v pevnom časovom intervale časové značky a ďalej parametre o obežnej dráhe družice, umožňujúce prijímaču vypočítať presnú polohu družice v dobe merania. Prijímač je na základe frekvencie prijímaného signálu f_p , časových značiek, parametrov obežnej dráhy družice a referenčného signálu o frekvencii f_0 generovaného priamo v prijímači, schopný pomocou opakovaných meraní vykonávaných vždy medzi dvoma časovými značkami vypočítať teoreticky až trojrozmernú polohu meraného bodu. [1]

1.1.3 Určovanie polohy na základe merania vzdialeností

Meria sa vzdialenosť medzi bodom, ktorého poloha sa zisťuje a vysielateľom, a táto vzdialenosť sa určí z doby šírenia signálu od vysielateľa k navigačnému prijímaču, prípadne z časového rozdielu medzi príchodom signálu z niekoľkých dvojíc vysielateľov.

V prípade globálnych družicových navigačných systémov sa vzdialenosť medzi prijímačom a družicami určuje prvým z dvoch vyššie uvedených spôsobov. Prijímač určuje čas t_{di} , ktorý potrebuje signál k tomu, aby dorazil z navigačnej družice, nachádzajúcej sa v mieste o súradniciach (x_i, y_i, z_i) , do miesta merania o súradniciach (X, Y, Z) , rýchlosťou šírenia rádiových vĺn (rovné rýchlosti svetla c). Aby sme mohli určiť polohu neznámeho bodu, stačí ak zmeriame vzdialenosti od troch navigačných družíc a vyriešime sústavy troch rovníc o troch neznámych. [1]

$$t_{d1} \cdot c = \sqrt{(x_1 - X)^2 + (y_1 - Y)^2 + (z_1 - Z)^2}$$

$$t_{d2} \cdot c = \sqrt{(x_2 - X)^2 + (y_2 - Y)^2 + (z_2 - Z)^2}$$

$$t_{d3} \cdot c = \sqrt{(x_3 - X)^2 + (y_3 - Y)^2 + (z_3 - Z)^2}$$

Je nevyhnutné, aby sa diaľkomerné signály jednotlivých družíc dali od seba odlíšiť.

Signály sa môžu od seba odlišovať troma spôsobmi:

- odlišnosť na základe frekvencie nosnej vlny, kedy každá zo súčasne viditeľných družíc používa pre prenos signálov nosnú vlnu s inou frekvenciou,
- odlišnosť na základe kódu, kedy všetky družice vysielajú na nosnej vlne s rovnakou frekvenciou, ale diaľkomerný kód je pre každú družicu iný,
- odlišnosť na základe doby vysielania, kedy všetky družice vysielajú rovnaký kód na rovnakej frekvencii, avšak v presne definovaných úsekoch.

Diaľkomerné kódy sú spravidla tvorené tzv. pseudonáhodnými signálmi, ktoré majú jednu vlastnosť, že ak porovnávame dva odlišné diaľkomerné kódy, je výsledný signál veľmi slabý, zatiaľ čo ak porovnávame dva rovnaké fázovo posunuté kódy, je výsledný signál výrazne silnejší. Pseudonáhodné signály sú periodické, generujú sa podľa určitých algoritmov.

Diaľkomerné navigačné systémy môžu byť aktívne alebo pasívne.

Aktívne systémy pracujú tak, že každý užívateľ je vybavený tzv. odpovedačom. Riadiaca stanica systému vyšle prostredníctvom navigačných družíc identifikačnú značku tohto prijímača. Keď prijímač značku rozpozná, odošle svoju odpoveď, ktorá je cez navigačné družice poslaná do riadiacej stanice, ktorá z vyhodnotenia oneskorenia odpovedí prijatých rôznymi družicami a zo znalosti polohy družíc v okamihu prijatia vypočíta polohu prijímača.

Pasívne systémy pracujú tak, že navigačné družice vysielajú diaľkomerné signály spolu s časovými značkami a údajmi o obežných dráhach družíc. Poloha sa zisťuje tak, že prijímač meria časový interval t_{di} , ktorý uplynie medzi odoslaním a z tohto intervalu je schopný dopočítať vzdialenosť k družiciam, a teda aj polohu prijímača. [1]

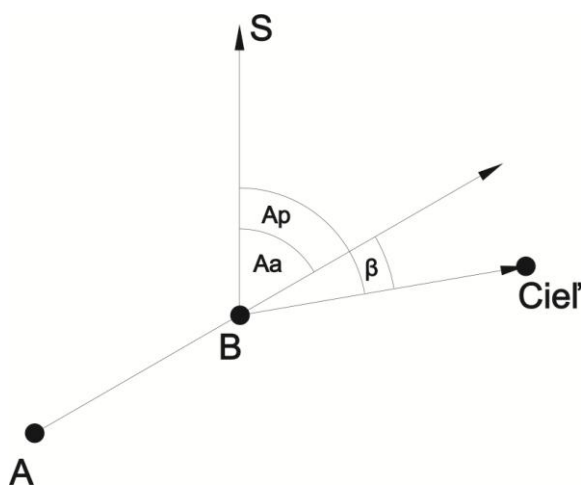
1.1.4 Určovanie polohy na základe fázového merania

Pri určovaní polohy na základe fázového merania sa vychádza z toho, že ak spočítame počet vlnových dĺžok rádiovej vlny, ktoré sa nachádzajú medzi prijímačom a vysielateľom v okamihu merania, môžeme jednoduchým vynásobením vlnovou dĺžkou prijímanej rádiovej vlny zistiť skutočnú vzdialenosť vysielateľ – prijímač. Aj keď je táto metóda jednoduchá, jej realizácia je obťažnejšia. V praxi nie je problém určiť desiatinnú časť vlny,

ale problémem je určit celočíselný počet vln. Tento počet sa obvykle označuje ako neurčitost'. [1]

2 URČOVANIE SMERU

Určovanie smeru je ilustrované na obrázku *Obrázok 1*. Smer možno určiť z rozdielu polôh tak, že spočítame azimut aktuálny Aa a požadovaný Ap . Azimut predstavuje uhol odklonu od severu. Ak stojíme, nevieme určiť smer, ale v momente keď sa pohneme z bodu A do bodu B, je smer pohybu určený. Tento smer je vyjadrený ako azimut Aa . Ap predstavuje azimut, ktorý by sme mali mať ak chceme prísť do cieľu. Rozdiel medzi Aa a Ap je uhol β , o ktorý ak sa otočíme v smere hodinových ručičiek, a budeme si držať azimut Ap , pridáme do cieľa.



Obrázok 1 Určovanie smeru

Pri určovaní smeru možno využiť aj elektronický kompas. V tomto prípade smer určujeme tak, že zistíme aktuálny azimut, teda ako sme otočený k severu, a azimut medzi aktuálnou polohou a cieľom. Rozdiel týchto dvoch uhlov udáva o koľko sa treba otočiť, aby sme od okamihu otočenia o uhol tvoriaci tento rozdiel pri priamočiarom pohybe dosiahli cieľu.

3 GPS

GPS je globálny družicový systém, ktorý bol vyvinutý Ministerstvom obrany USA v roku 1973, s cieľom spĺňať požiadavky vojenských jednotiek na presné určovanie polohy, rýchlosti a času v jednotnom systéme kdekoľvek a kedykoľvek na Zemi. Neskôr bol americkým kongresom sprístupnený pre civilné obyvateľstvo.

GPS je založený na princípe jednosmerného dĺžkomeru. Meranou veličinou je doba šírenia rádiového signálu z družicovej antény k anténe prijímača. Nameraný čas sa prepočítava na vzdialenosť. Jeho výhodou je okamžité a pomerne presné určenie polohy ľubovoľného počtu i rýchlo pohybujúcich sa objektov. Signály GPS sú tiež odolné voči rušivým vplyvom. [3]

3.1 Obecná štruktúra družicových polohových systémov.

Družicové polohové systémy sú tvorené troma segmentmi:

- kozmický,
- riadiaci,
- používateľský.

3.1.1 Kozmický segment

Kozmický segment je tvorený sústavou družíc obiehajúcich po známych, presne definovaných a určených obežných dráhach tak, aby bolo vždy viditeľných najmenej 5 až 8 družíc. Tento segment je definovaný:

- typom obežných dráh,
- výškou, sklonom a počtom obežných dráh,
- počtom a rozmiestnením družíc na obežných dráhach. [1]

3.1.2 Riadiaci segment

Úlohy riadiaceho segmentu sú:

- nepretržité monitorovanie a riadenie družicového systému,
- určovanie systémového času,
- predpovedanie dráhy družíc a chod hodín na družiciach

- pravidelné obnovovanie navigačnej správy každej družice.

V riadiacom segmente sa rozlišujú tri typy staníc:

- monitorovacie stanice – monitorujú signály vysielané družicami kozmického segmentu a prenášajú ich do centra,
- hlavná riadiaca stanica – spracováva signály z monitorovacích staníc, prevádza modelovanie chovania kozmického segmentu, určuje parametre obežných dráh a korekčných parametrov, hodín na družiciach a výsledky predáva na stanice určené pre komunikáciu s družicami,
- stanice pre komunikáciu s družicami – slúžia k prenášaniam novo určených parametrov obežných dráh a korekčných parametrov atómových hodín umiestených na družiciach a taktiež slúži k ich ovládaniu. [1] [2]

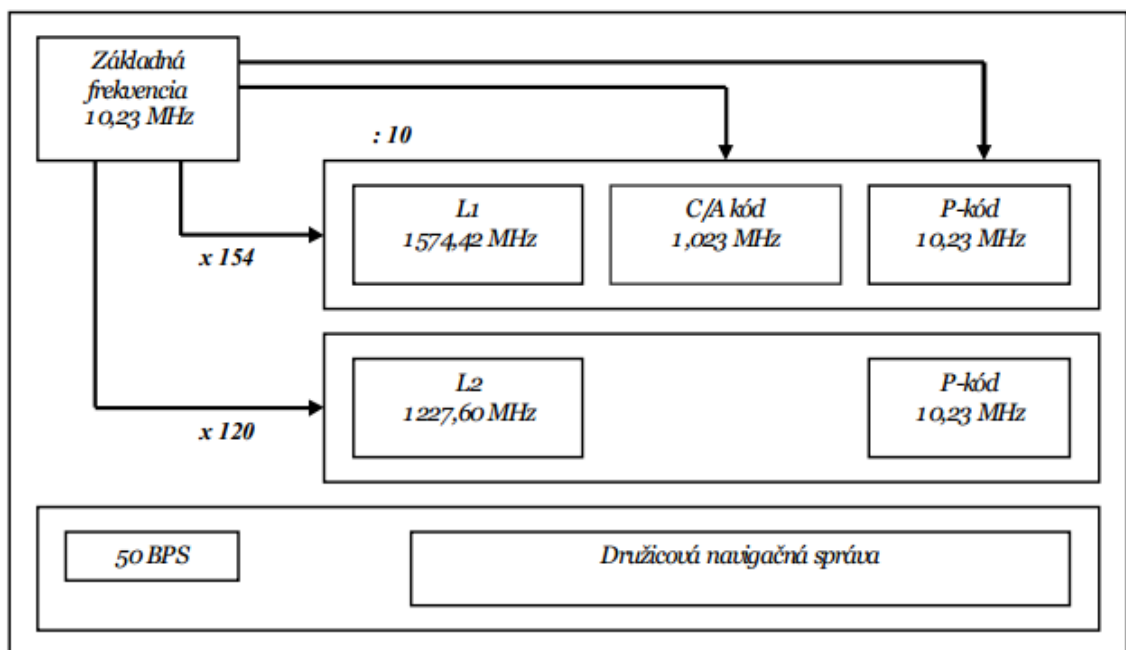
3.1.3 Užívateľský segment

Tvoria ho prijímacie zariadenia ako napr.:

- navigačné prijímače na pozemnú, námornú, leteckú a inú navigáciu,
- geodetické prijímače a to jednofrekvenčné a dvojfrekvenčné prístroje, RTK systémy a pod,
- prijímače na časovú synchronizáciu pre potreby astronomických meraní a telekomunikačných zariadení. [2]

3.2 Signály vysielané družicami

Každý signál vyslaný družicou GPS je kombináciou nosnej vlny, diaľkomerného kódu a navigačnej správy. Vysielaný signál, je vytváraný v celej rade krokov. Vychádza sa z faktu, že všetky zložky signálu sú odvodzované násobením a delením základnej frekvencie $f_0 = 10,23 \text{ MHz}$. [1]



Obrázok 2 Schéma odvodzovania frekvencií jednotlivých signálov [3]

Celočíselným násobením základnej frekvencie f_0 sú vytvorené dve nosné vlnenia v L-pásme rádiových vln označené ako L1 a L2, pričom pre ich frekvencie platí

$$f_1 = 154 \times f_0 \text{ a } f_2 = 120 \times f_0. [3]$$

Frekvencia L1 je modulovaná dvoma meracími kódmi tzv. pseudonáhodnými šumami, označovanými aj P-kód, a C/A-kódmi. Druhá frekvencia L2 je modulovaná P-kódom, resp. jeho šifrovaným variantom Y-kódom. Okrem C/A a P-kódu je oboma nosnými frekvenciami prenášaný ešte binárny kód, obsahujúci navigačnú správu, ktorý je zakódovaný pomocou fázových posunov nosných vln. [1] [3]

3.2.1 Základná frekvencia

Družice GPS odvodzujú frekvencie všetkých svojich signálov od základnej frekvencie, ktorá je odvodená z frekvencie atómových hodín a jej presná hodnota je nastavená tak, aby boli eliminované relativistické efekty, spôsobené pohybom družíc. [3][1]

3.2.2 C/A-kód

Vysiela sa na nosnej frekvencii L1 a jedná sa o sekvenciu 1023 núl a jednotiek. Každá družica má pridelený svoj vlastný C/A kód. Využitie tohto kódu je hlavne pre navigáciu a časovú synchronizáciu. [3]

3.2.3 P-kód

P-kód má dĺžku $2,3547 \cdot 10^{14}$ bitov, no celková dĺžka kódu je rozdelená na menšie segmenty o dĺžke $6,1871 \cdot 10^{12}$ bitov. Každý družici je priradený jeden segment. P-kódom sa modulujú obe frekvencie L1 aj L2. Umožňuje merať vzdialenosť medzi užívateľom a družicou s vyššou presnosťou, a to najmä z dôvodu použitia rýchlejšieho a dlhšie kódu, pričom dochádza k väčšiemu frekvenčnému rozprestretiu a zvyšuje sa presnosť merania. [1][2]

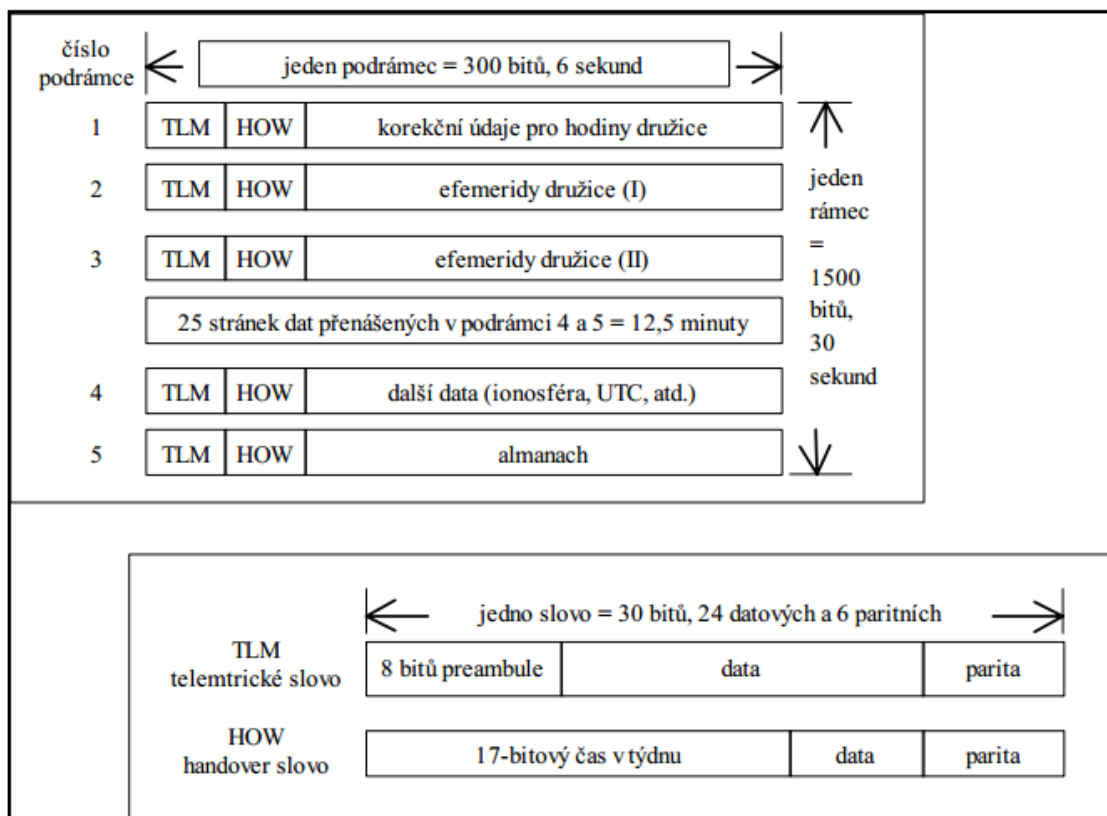
3.2.4 Y-kód

Je možné ho považovať za šifrovaný P-kód a rovnice pre jeho dešifrovanie sú tajné. [3]

3.2.5 Navigačná správa

Navigačná správa je súbor údajov vysielaných každou družicou. Obsahom navigačnej správy sú parametre hodín na palube družice, dráhové parametre družice, zdravotný stav družice, jej pozícia a rôzne korekčné údaje. Správu tvorí tzv. rámec dlhý 1 500 bitov, ktorý je zložený z piatich blokov:

- 1. blok obsahuje poradové číslo týždňa GPS, koeficienty kvadratického polynómu, slúžiaceho pre korekciu hodín a parametre indikujúce stav družice,
- 2. a 3. blok obsahuje vysielané dráhové parametre danej družice. Údaje o pozíciách (efemeridy) sú dekodované prijímačmi,
- Obsahom 4. a 5. bloku sú údaje o stave ionosféry, informácie o čase UTC a tzv. almanach.



Obrázok 3 Štruktúra navigačnej správy [1]

Efemeridy popisujú obežnú dráhu družíc pre krátky úsek ich orbity. GPS prijímač obyčajne prijme nové efemeridy každú hodinu, alebo môže používať staré dáta až po dobu štyroch hodín bez výrazného ovplyvnenia merania.

Almanachy predstavujú približné parametre obežných dráh všetkých družíc. Ak sú známe aktuálne almanachy, je možné znížiť dobu potrebnú pre naštartovanie GPS prijímača a získanie signálu. Prijímač tieto približné parametre dráh využíva pre prednastavenie približných pozícií družíc a Dopplerovského posunu nosných frekvencií každej družice. [3][2]

3.3 Určovanie polohy

Družicové polohové systémy sú budované ako pasívne diaľkomerné systémy, tzn. že prijímač určuje svoju vzdialenosť k niekoľkým družiciam navigačného systému a svoju polohu tak stanovuje pretínaním. Určovanie vzdialeností prijímača od družíc možno dosiahnuť na základe:

- kódového merania,
- fázového merania,

- dopplerovského merania. [1]

Opísaný bude len princíp kódového merania, lebo princípy fázového a dopplerovského merania boli popísané v predchádzajúcej kapitole.

3.3.1 Kódové meranie

Predstavuje základné meranie pomocou systému GPS. Princíp si ukážeme na jednoduchom príklade, kedy na vstupe prijímača sa objaví len signál z jednej družice a bez akéhokoľvek šumu. Prijímaná nosná vlna L1, ktorá je modulovaná C/A kódom, je prevedená na signál s nižšou frekvenciou, a tá je potom zmiešavaná s C/A kódom generovaným priamo v prijímači. V prijímači generovaný kód nie je synchronný s časom GPS a teda s C/A kódom generovaným družicou, a to z dôvodu, že hodiny v prijímači sú riadené len kremíkovým kryštálom a majú teda výrazne nižšiu stabilitu. Ďalej aj preto, lebo prijímaný signál je proti času GPS posunutý o časový interval, potrebný k urazeniu vzdialenosti od družice k prijímaču. Prijímač musí nájsť pre neznámu družicu odpovedajúci diaľkomerný kód a následne musí postupným posúvaním prijímačom generovanej sekvencie dosiahnuť plnej zhody oboch kódov. Keď sa dosiahne synchronizácie, vyruší sa navzájom C/A kód generovaný družicou a prijímačom. Na výstupe vstupného modulu prijímača sa objaví len nosná vlna modulovaná navigačnou správou, ktorú je možno ďalej spracovať.

Vzhľadom k tomu, že družica vysiela jednotlivé sekvencie C/A kódu v presne stanovených časových okamihoch, je možné z prijatého C/A kódu a navigačnej správy určiť presný čas odoslania signálu. Rozdiel medzi časom odoslania sekvencie C/A kódu a časom prijatia prijímačom je rovná času šírenia signálu od družice k prijímaču. Z tohto časového rozdielu je možné spočítať jeho vynásobením rýchlosťou šírenia rádiových vln pseudovzdialenosť. [1]

3.4 Presnosť merania

Určenie 3D polohy objektu pomocou GPS sa dá vysvetliť tak, že sa nachádza v priesečníku guľových plôch, ktorých polomer je daný meranými vzdialenosťami. K dosiahnutiu vysokej presnosti určenia polohy je teda dôležité, aby sme využívali čo najväčšieho počtu viditeľných družíc.

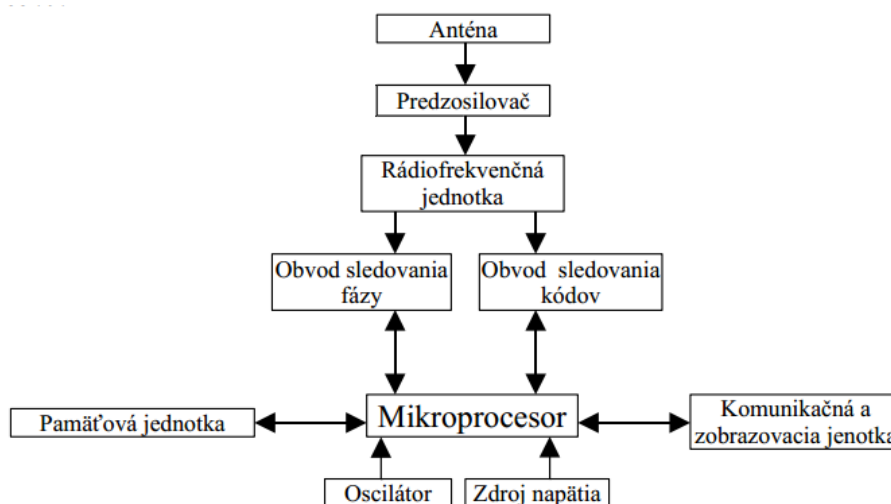
Presnosť polohy stanovená GPS sa môže pohybovať od 100m do niekoľko mm v závislosti od použitého prístroja, spôsobu merania a spracovania. Vplyvy jednotlivých zdrojov chýb sú ukázané v tabuľke *Tabuľka 1*. [3]

Tabuľka 1 Zdroje chýb a ich príspevok k celkovej chybe [3]

Zdroj chýb	Max. veľkosť chyby
Družicové hodiny	40 m
Efemeridy družíc	15 m
Obežná dráha	5 m
Selektívna dostupnosť (S/A)	10 m
Vplyv ionosféry	12 m
Vplyv troposféry	3 m
PRN šum	1 m
Šum prijímača	2 m
Odrazené signály	2 m

3.5 GPS prijímač

Základná schéma GPS prijímača je ukázaná na obrázku *Obrázok 4*.



Obrázok 4 Základná schéma GPS prijímača [2]

Anténa GPS prijímača premieňa energiu elektromagnetických vĺn prichádzajúcich z družice na elektrický prúd. Signál z antény je veľmi slabý a z tohto dôvodu je anténa doplnená nízkošumovým predzosilňovačom, ktorý zvyšuje hladinu signálu predtým, než sa cez koaxiálny kábel dostane do prijímača. Úlohou rádiofrekvenčnej jednotky je zmeniť vysokú frekvenciu nosných vĺn signálu na nižšiu, medzifrekvenčnú, z dôvodu jednoduchšieho spracovania. Referenčný signál je generovaný kremenným oscilátorom. Obvody sledovania kódov a fázy merajú prijatý signál, ktorý sa mení s časom a mení sa aj jeho frekvencia. V spätnoväzobných obvodoch sa priebežne porovnáva signál generovaný v prijímači so signálom prijatým z družice s cieľom vytvorenia chybového signálu ako ich

rozdielu. Minimalizáciou tohto chybového signálu sa hľadá replika družicového signálu posunutého v čase a frekvencii.

Obvod sledovania kódov sa používa pri kódových technikách na priradenie postupnosti pseudonáhodného kódu (C/A-kódu alebo P-kódu), ktorý je vyslaný družicou a prijatý prijímačom, k identickému kódu generovaného prijímačom. Tieto dva signály sa v korelátore postupne posúvajú v čase, vzájomne sa porovnávajú a sleduje sa pri tom výstupná hodnota korelátora, ktorá je maximálna v prípade zhody signálov. Čas, ktorý je potrebný na dosiahnutie vzájomného priradenia signálov, je rovný času potrebnému na šírenie signálu od družice k prijímaču. Po identifikácii pseudonáhodných kódov sa tieto odstránia z prijatého signálu a spracovanie pokračuje v obvode sledovania fázy. Obsah navigačnej správy sa demoduluje priradením fázy prijatého filtrovaného medzifrekvenčného signálu k fáze signálu generovaného prijímačom.

Mikroprocesor slúži na riadenie celkovej činnosti prijímača. Pracuje na digitálnej báze, preto musí byť prijatý analógový signál prevedený na digitálny. Mikroprocesor vypočítava časový interval z kódového merania a následne určuje pseudovzdialenosti, počíta celé cykly a určuje zlomkové časti merania fázy nosnej vlny. Mikroprocesor ďalej dekoduje navigačnú správu a určuje okamžitú polohu pomocou pseudovzdialeností. Okrem toho dekoduje almanach a vypočítava polohu družíc v horizontálnom súradnicovom systéme so začiatkom v mieste prijímača. Mikroprocesor umožňuje aj operácie, ako sú výpočty rýchlosti pri pohybujúcom sa prijímači, výpočty parametrov pre navigáciu, určenie okamžitých hodnôt veličín DOP.

Súčasťou komunikačnej a zobrazovacej jednotky môže byť displej a sériový port pre komunikáciu. [3]

4 NMEA 0183

NMEA – Nation Marine Electronic Association (národná asociácia pre námornú elektroniku) vytvorila štandard, ktorý definuje rozhranie medzi elektronickými zariadeniami používanými pre lodnú dopravu. Tento štandard umožňuje výmenu informácií medzi počítačom a rôznymi zariadeniami, v našom prípade GPS prijímačom. Prostredníctvom protokolu predáva GPS prijímač informácie o pozícií. [4]

4.1 Obecný formát

V NMEA 0183 formáte sú dáta posielané po riadkoch. Každý riadok začína znakom '\$', nasleduje skratka zariadenia, označovaná ako tt, ďalej kód z troch písmen určujúci formát správy sss. Nasledujú dátové položky oddelené čiarkami a po nich kontrolný súčet. Veta je ukončená znakmi <CR><LF>. Význam jednotlivých dátových položiek je definovaný pre konkrétny typ vety. Ak nie je dátová položka k dispozícii, zostane dátové pole prázdne. [4][5]

Obecný formát:

\$ttsss,d1,d2,...<CR><LF>

Najpoužívanejšie vety sú opísané v nasledujúcich častiach.

4.1.1 GSA

Veta GSA informuje o počte dimenzií, o aktívnych satelitoch a o odchýlkach od presnosti DOP. Satelity sú popísané podľa identifikátoru. [5]

Príklad:

\$GPGSA,A,3,29,26,22,09,07,05,04,,,,,1.7,1.0,1.4*30

Tabuľka 2 Popis vety GSA [4]

	Formát	Príklad	Komentár
1	c	A	Prepínanie medzi N-rozmernými módy (A = automatické, M = Manuálne)
2	dd	3	Počet dimenzií N (1=D, 2=2D, 3=3D)
3-14	dd	29	ID satelitu použiteľného pre výpočet
15	d.d	1.7	PDOP (Position Dilution Of Precision) v metroch
16	d.d	1.0	HPDOP (Horizontal Dilution Of Precision) v metroch
17	d.d	1.4	VDOP (Vertical Dilution Of Precision) v metroch
18	*xx	30	Kontrolný súčet

4.1.2 RMC

Výraz RMC poskytuje minimálne doporučené informácie pre navigáciu, čím sa myslí zemepisná šírka a dĺžka, kurz pohybu. Ďalej obsahuje informácie o aktuálnom čase o rýchlosti v uzloch a dátume. [5]

Príklad:

\$GPRMC,170138.615,A,4912.2525,N,01635.0378,E,0.04,16.43,280705,*,*32

Tabuľka 3 Popis vety RMC [4]

	Formát	Príklad	Komentár
1	hhmmss.sss	192031.241	Čas (UTC)
2	c	A	Status (A=OK, V = varovanie)
3	ddmm.mmmm	4818.3485	Zemepisná šírka
4	c	N	Indikátor sever/juh (N = sever, S = juh)
5	ddmm.mmmm	1820.4873	Zemepisná dĺžka
6	c	E	Indikátor východ/západ (E = východ, W = západ)
7	d.d	0.04	Vodorovná rýchlosť v uzloch
8	d.d	16.43	Kurz pohybu v stupňoch
9	ddmmyy	181014	Dátum
10	d.d	3.2	Magnetická deklinácia v stupňoch
11	c	E	Indikátor východ/západ (E = východ, W = západ)
12	*xx	32	Kontrolný súčet

4.1.3 GSV

Veta GSV informuje o počte viditeľných satelitov a o každom z nich poskytujú informácie ako identifikačné číslo družice, uhlová výška, azimut a odstup signál - šum. Množstvo údajov závisí od počtu viditeľných družíc. Jedna veta môže obsahovať maximálne 80 znakov, čo vystačí na maximálne štyri družice. Informácie preto bývajú rozdelené od niekoľkých viet. [5]

Príklad:

\$GPGSV,3,1,11,09,84,297,41,05,48,256,45,07,38,059,41,26,22,178,41*74

\$GPGSV,3,2,11,24,13,063,00,14,12,324,00,30,12,251,00,22,12,286,38*78

\$GPGSV,3,3,11,29,10,173,35,04,09,105,30,18,06,254,00*46

Tabuľka 4 Popis vety GSV [4]

	Formát	Príklad	Komentár
1	d	3	Celkový počet viet
2	d	1	Číslo aktuálnej vety
3	dd	11	Počet viditeľných družíc
4	dd	09	Identifikačné číslo družice
5	dd	84	Uhlová výška, kde sa daná družica nachádza
6	ddd	297	Azimut, kde sa daná družica nachádza
7	dd	41	Odstup signál od šumu. Ak je tento údaj rovný nule, nemožno satelit použiť k výpočtu polohy
8	Od počtu vid družíc môžu nasledovať ďalšie štvorice (4-7)
9	*xx	74	Kontrolný súčet

4.1.4 GGA

Výraz GGA poskytuje informácie o zemepisnej šírke a dĺžke, o počte viditeľných satelitov, o výške antény nad geoidom, o čase poslednej aktualizácie DGPS a iné. [5].

Príklad:

\$GPGGA,170139.615,4912.2526,N,01635.0378,E,1,07,1.0,357.5,M,43.5,M,0.0,0000*7D

Tabuľka 5 Popis vety GGA [4]

	Formát	Príklad	Komentár
1	hhmmss.sss	192031.241	Čas (UTC)
2	ddmm.mmmm	4818.3485	Zemepisná šírka
3	c	N	Indikátor sever/juh (N = sever, S = juh)
4	ddmm.mmmm	1820.4873	Zemepisná dĺžka
5	c	E	Indikátor východ/západ (E = východ, W = západ)
6	d	1	Indikátor kvality: 0 - nebolo možné určiť pozíciu 1 - pozícia úspešne určená 2 - pozícia úspešne určená (diferenčné GPS)
7	dd	07	Počet viditeľných satelitov 00 - 12
8	d.d	1.0	Vplyv rozostavania družíc na určenie polohy HDOP
9	d.d	357.5	Výška antény nad geoidom
10	c	M	Jednotka pre prechádzajúci údaj (M = meter)
11	d.d	43.5	Geoidial separation. Rozdiel medzi WGS-84 zemským elipsoidom a strednou úrovňou mora (geoidom).
12	c	M	Jednotka vzdialenosti pre prechádzajúcu položku
13	d.d	0.0	Čas od poslednej aktualizácie DGPS v sekundách.
14	dddd	0000	Identifikačné číslo referenčnej stanice DGPS.
15	*xx	7D	Kontrolný súčet

5 EMBEDDED SYSTÉMY

Embedded systém je kombináciou počítačového hardvéru a softvéru, a prípadne aj prídavných mechanických alebo iných častí, vytvorený pre špecifickú funkciu. Príkladom môže byť mikrovlnná rúra, práčka, roboty a iné. [10]

5.1 Real-time operačné systémy

Operačný systém reálneho času je operačný systém, ktorý má časove obmedzenie. Inak povedané, RTOS je schopný vykonať činnosť v určitý čas, teda činnosti majú termín, do kedy majú byť vykonané. [10]

5.2 Embedded operačné systémy

Tieto operačné systémy sú navrhnuté tak, aby boli kompaktné, spoľahlivé a aby efektívne využívali poskytnuté zdroje, pričom neposkytujú mnoho funkcií poskytovaných ne-embedded systémami. Hardvér, na ktorom obvykle beží embedded operačný systém je limitovaný zdrojmi ako RAM a ROM, preto tieto systémy sú špecifické pre konkrétny hardvér, čo znamená, že vďaka dostupným prostriedkom tieto operačné systémy pokrývajú špecifické úlohy. Mnohé z týchto operačných systémov poskytuje aj beh aplikácií v reálnom čase. [15]

5.2.1 Embedded Linux

Operačný systém Linux bol pôvodne určený pre osobné počítače. S vývojom hardvéru sa začal objavovať aj na embedded systémoch. Linux je opensource operačný systém s monolitickým jadrom. Linux je otvorená platforma, tým pádom je zadarmo, čo je dôvodom aj jeho rozmachu. [14]

5.3 Softvér pre embedded systémy

Aplikácie pre embedded systémy sa môžu písať v množstve jazykov, vo veľkej miere však v jazykoch C alebo C++. Ako pri vývoji každej aplikácie aj tu sa stretávame s pojmami kompilácia a debugovanie (ladenie).

5.3.1 Krížová kompilácia

Väčšina softvéru pre Linux je vyvíjaná na rovnakom type zariadenia, na ktorom nakoniec pobeží dokončená aplikácia. U vývoja softvéru pre embedded systémy to už nie je také

jednoduché. Musí sa totiž pracovať so zariadenia rôznych druhov, pričom každé z nich môže vyžadovať Linux, ktorý beží na úplne inej architektúre. Tieto zariadenia nemajú väčšinou dostatočný výkon procesoru a dostupný úložný priestor, aby sa softvér mohol vyvíjať na danom zariadení. Preto sa využíva takzvaná krížová kompilácia. Krížová kompilácia predstavuje proces vývoja softwaru na inej platforme, ako bude finálny softvér bežať. [9]

5.3.2 Debuggovacie nástroje

Ladiacich nástrojov sa využíva hlavne vtedy, keď chceme nájsť chybu v programe a tú následne odstrániť. Pri ladení sa používajú breakpointy, ktoré slúžia na zastavenie vykonávaného programu. Breakpointy môžu byť aj podmienené, kedy nastane prerušenie len za určitých podmienok, napríklad pri nejakej hodnote premennej.

Debuggery poskytujú programátorovi nástroje ako sledovanie priebehu programu, sledovanie aktuálnych hodnôt premenných, pozastavenie, opakované spustenie, nastavenie breakpointov, a iné.

Pri programovaní embedded systémov sa často využíva dvoch spôsobov debuggovania.

Jedným spôsobom je, že debuggovaný program beží na cieľovom HW a debugger na inom počítači. Používa sa väčšinou pri embedded systémoch nižšej úrovne.

Druhý spôsobom je ladenie, pri ktorom debugger aj debuggovaný program bežia priamo na danom zariadení a užívateľ sa k tomuto zariadeniu vzdialene pripojuje. Tento spôsob sa používa pri embedded systémoch vyššej úrovne, ako napríklad s Linuxom.

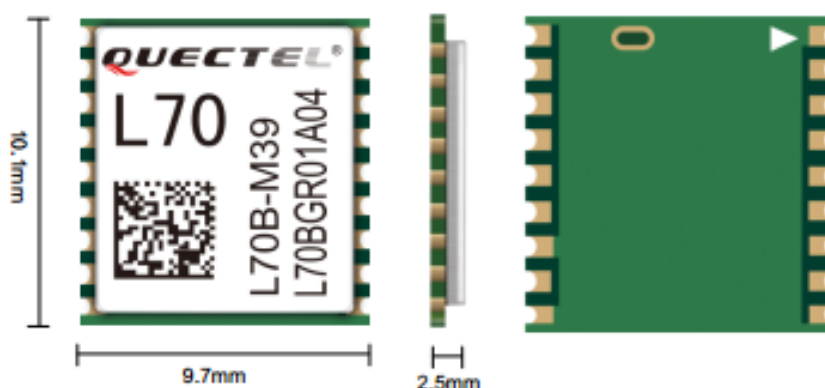
Najčastejšie sa pre ladenie používa ladiaci nástroj GDB alebo grafické vývojové prostredie Eclipse, ktoré využíva externé ladiace nástroje typu GDB. [11] [12]

II. PRAKTICKÁ ČÁST

6 POUŽITÝ HARDVÉR

6.1 GPS MODUL

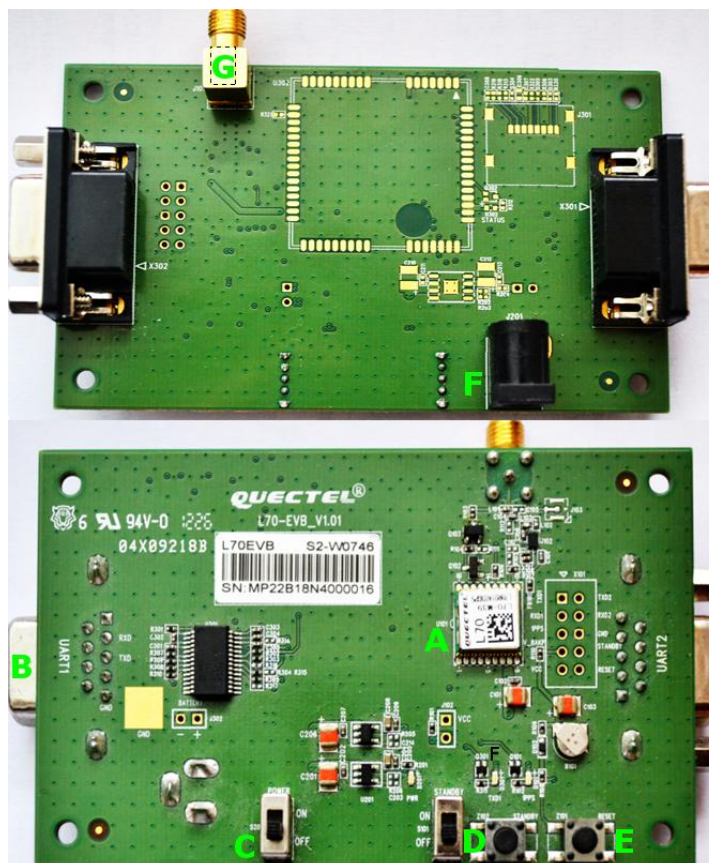
V rámci práce bol použitý GPS modul L70 (viď. *Obrázok 5*) od firmy Quectel, ktorý sa vyznačuje vysokým výkonom, nízkou spotrebou a rýchlym určením polohy. Využíva technológiu AGPS nazývanú aj EASY, vďaka ktorej môže počítať a predpovedať orbity automaticky použitím efemeridov uložených v internej flash pamäti, a tak môže modul rýchlo vykonať určenie polohy dokonca aj v interiéroch, kde je slabší signál. AlwaysLocate technológia umožňuje GPS prijímaču adaptívne vypínať a zapínať čas, s cieľom dosiahnutia kompromisu medzi presnosťou merania a spotrebou energie v závislosti na prostredí a pohybových podmienkach. V neposlednom rade funkcia LOCUS umožňuje modulu zapisovať informácie o polohe do internej pamäte v 15-sekundových intervaloch a to po dobu viac než 16 hodín. [6]



Obrázok 5 GPS modul L70 a jeho rozmery [6]

Modul L70 je osadený na EVB doske (viď *Obrázok 6*). Doska obsahuje tieto komponenty:

- A - GPS modul L70
- B - Sériový port
- C - Vypínač
- D - Tlačidlo a prepínač do standby režimu
- E - Tlačidlo pre reštart
- F - Napájanie
- G - Konektor pre pripojenie antény



Obrázok 6 GPS modul L70 na EVB doske

6.2 Mikropočítač Colibri T30

Aplikácia bola testovaná mikropočítači Colibri T30 (Obrázok 7) od firmy Toradex. Jeho špecifikácie sú uvedené v tabuľke *Tabuľka 6*. Na mikropočítači bol nainštalovaný operačný systém Linux.

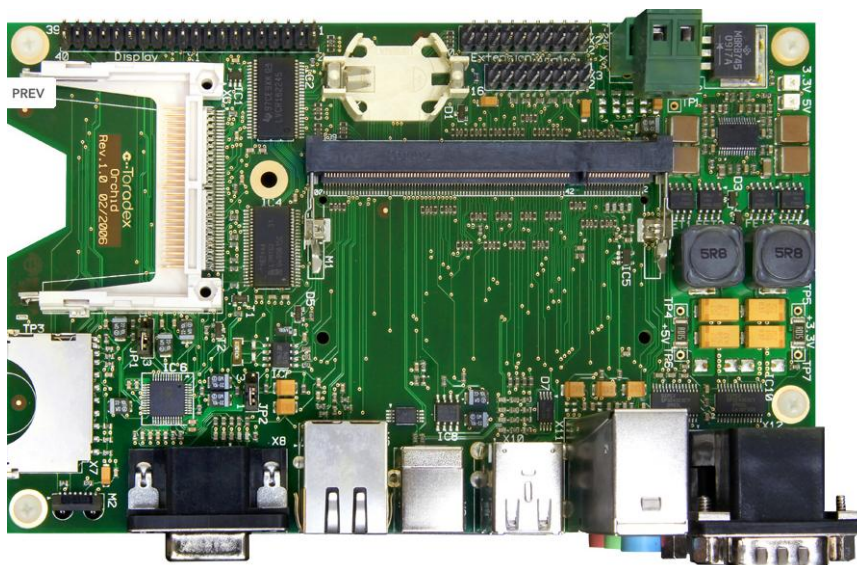


Obrázok 7 Mikropočítač Colibri T30 [7]

Tabuľka 6 Špecifikácie Colibri T30 [7]

CPU	
Názov CPU	NVIDIA Tegra 3
Typ CPU	ARM Cortex-A9
Počet jadier	4
Frekvencia procesoru	1,4 GHz
L1 inštrukčná cache	32 KB pre jadro
L1 dátová cache	32 KB pre jadro
L2 cache	1 MB zdieľaný
Pamäť	
Ram	1 GB DDR3L (32 Bit)
Flash	2 GB eMMC (8 Bit)

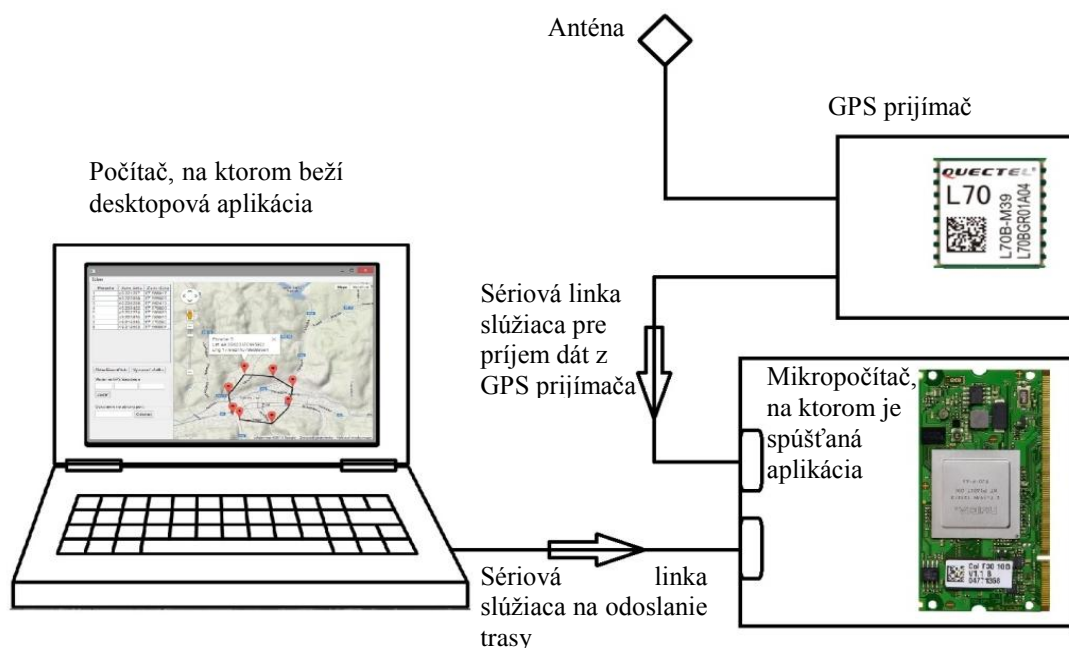
Mikropočítač je osadený na doske Orchid carrier board (Obrázok 8). Doska obsahuje dva sériové RS232 porty, jeden RJ45 port, čítačku SD pamäťových kariet, dva USB porty, VGA video výstup a iné rozhrania pre vstup a výstup. [7]



Obrázok 8 Orchid carrier board [7]

7 NÁVRH SYSTÉMU

Systém sa skladá z dvoch základných častí, a to desktopovej aplikácie a aplikácie, bežiacej na mikropočítači, ktorá využíva funkcie knižnice *nav.h*, ktorej vytvorenie bolo tiež súčasťou tejto práce. Celkový návrh systému je vyobrazený na obrázku *Obrázok 9*. Počítač slúži ako systém, ktorý sprostredkúva beh desktopovej aplikácie. Medzi ním a mikropočítačom je sériový kábel, po ktorom sa odosiela vytvorená trasa. Namiesto sériového káblu môže byť použitý aj USB kábel. Po nahratí trasy do mikropočítača možno komunikáciu medzi mikropočítačom a počítačom ukončiť. Na mikropočítači beží aplikácia navigácie, príma dáta z GPS prijímača, ktorý je pripojený k mikropočítaču opäť cez sériový kábel. S prijímačom je ešte spojená anténa, ktorá prijíma informácie potrebné pre určenie polohy vo forme elektromagnetického vlnenia.



Obrázok 9 Návrh systému

7.1 Desktopová aplikácia

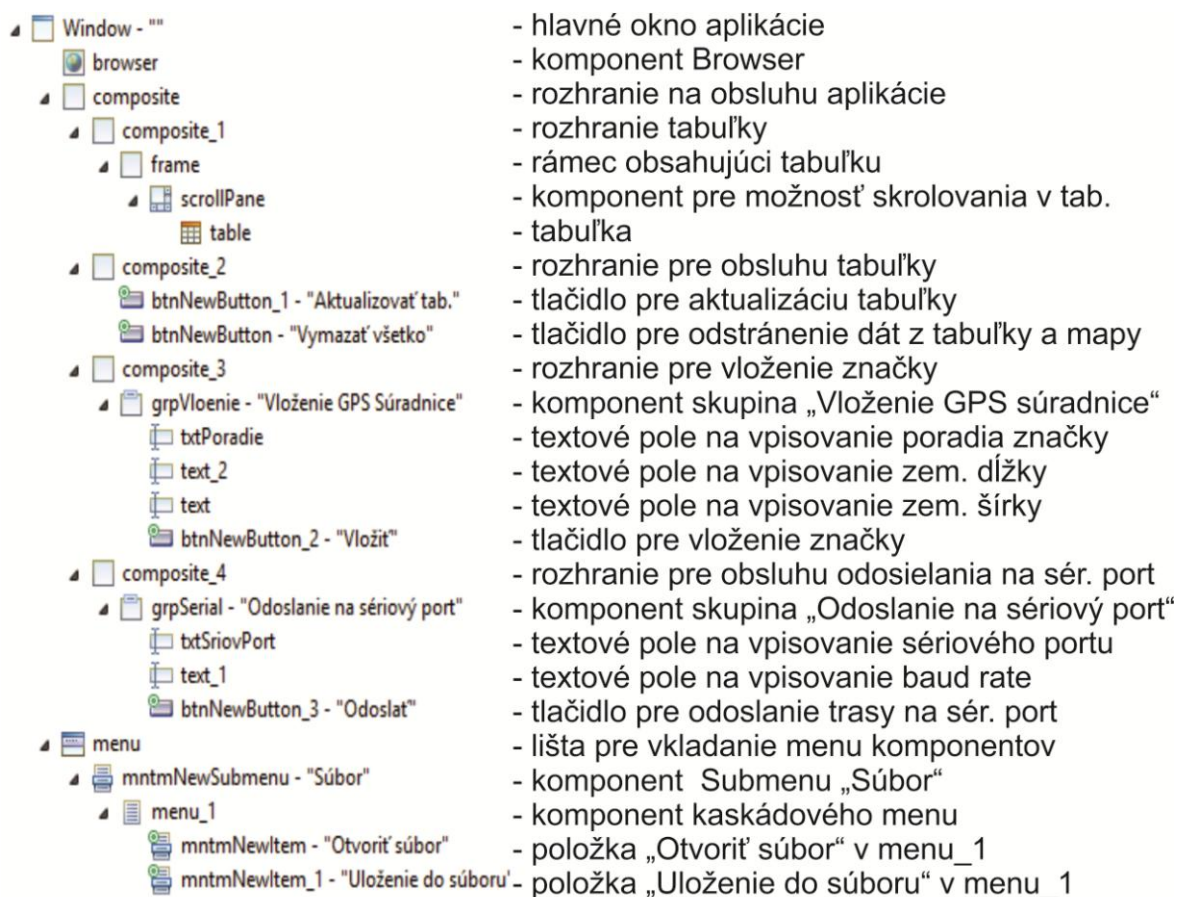
Desktopové rozhranie slúži na komunikáciu s mikropočítačom. Možno v ňom navoliť trasu a tú následne poslať cez sériovú linku do mikropočítača.

7.1.1 Použité technológie

Aplikácia je napísaná v jazyku Java, a ako bolo spomenuté, s mikropočítačom komunikuje prostredníctvom sériovej komunikácie. Pre vytvorenie GUI bola použitá knižnica

grafických užívateľských prvkov SWT a z časti aj knižnica SWING, pre potrebu vytvorenie tabuľky, keďže možnosti jTable sú rozsiahlejšie ako možnosti elementu Table v knižnici SWT. Pre sériovú komunikáciu sú využívané funkcie knižnice JSSC, ktorá je voľne dostupná pod licenciou GNU Lesser GPLv3.

Ukážka štruktúry komponentov a ich popis je na obrázku *Obrázok 10*.



Obrázok 10 Štruktúra komponentov a ich popis

Dominantným prvkom aplikácie je element z knižnice SWT browser, ktorý načítava zadanú URL adresu, v tomto prípade súbor index.html, ktorý obsahuje kód napísaný v jazykoch HTML a JavaScript (ukážka kódu v JavaScripte *Zdrojový kód 1*), a je v ňom vytvorené webové rozhranie s mapami od firmy Google. Java aplikácia volá JavaScript funkcie, ktoré slúžia na vytvorenie a mazanie značiek na mape, a naopak v JavaScript kóde sú volané funkcie, ktoré pracujú s tabuľkou.

```
function addMarker(location, update) {
    path = poly.getPath();
    path.push(location);
    var marker = new google.maps.Marker({
        position: location,
        map: map
    });

    markers.push(marker);
    if (update == false)
        theJavaFunction(location.lat(), location.lng());

    google.maps.event.addListener(marker, 'rightclick', function () {
        poly.setMap(null);
        theDeleteFunction(markers.lastIndexOf(marker));
        markers.splice(markers.lastIndexOf(marker), 1);
        this.setMap(null);
        poly = new google.maps.Polyline(polyOptions);
        poly.setMap(map);
        path = poly.getPath();
        for (var i = 0; i < markers.length; i++) {
            path.push(markers[i].getPosition());
        }
    });

    var infowindow = new google.maps.InfoWindow({
        content: 'Poradie: ' + (markers.lastIndexOf(marker) + 1) + '<br>Lat: ' +
            location.lat() + '<br>Lng: ' + location.lng()
    });
    google.maps.event.addListener(marker, 'click', function () {
        infowindow.open(map, marker);
    });
}
```

Zdrojový kód 1 Ukážka kódu v jazyku JavaScript – funkcia pre vloženie značky

7.1.1.1 Volanie funkcií napísaných v Javascripte v Java kóde

Aby sa prejavila zmena v tabuľke aj na mape, je potreba volať funkcie v JavaScripte v kóde aplikácie napísanej v Jave. V ukážke kódu *Zdrojový kód 2* je vytvorený *Browser*. Trieda *Browser* obsahuje metódu *evaluate()*, ktorá môže volať JavaScript funkcie.

```
//vytvorenie browsera v ktorom bude index.html zobrazeny (index.html sluzi na zobrazenie mapy a
pracu z google maps API)
final Browser browser = new Browser(Window, SWT.NONE);
browser.setTouchEnabled(true);
browser.setLayoutData(BorderLayout.CENTER);
browser.setUrl("index.html");
browser.evaluate("update("+model.getValueAt(i, 1)+","+model.getValueAt(i, 2)+");");
```

Zdrojový kód 2 Ukážka volania JavaScript funkcie v Java

7.1.1.2 Volanie funkcií napísaných v Jave v JavaScript kóde

Aby naopak zas bolo možné po pridaní alebo vymazaní značky na mape pridať alebo vymazať značku aj v tabuľke, muselo byť použité riešenie na zavolanie funkcií v jazyku Java aplikáciou napísanou v JavaScripte. V ukážke kódu *Zdrojový kód 3* je časť kódu, kde je ukážka vytvorenia triedy *CustomFunction*, ktorá je potomkom triedy *BrowserFunction*.

Metóda *function()* je volaná kedykoľvek, keď v JavaScript kóde je volaná funkcia *theJavaFunction*. Potom v kóde *Zdrojový kód 4* je ukázané volanie Javovskej funkcie v Javascript kóde.

```
new CustomFunction (browser, "theJavaFunction", Window);
class CustomFunction extends BrowserFunction {
    Shell shell;
    CustomFunction (Browser browser, String name, Shell shell) {
        super (browser, name);
        this.shell = shell;
    }
    public Object function (Object[] arguments) {
        double lat = ((Double) arguments[0]).doubleValue();
        double lng = ((Double) arguments[1]).doubleValue()-
        360*Math.floor(((Double) arguments[1]).doubleValue()/360);
        DefaultTableModel model = (DefaultTableModel) table.getModel();
        model.addRow(new Object[] {new Integer(model.getRowCount()+1),lat,lng});
        return null;
    }
}
```

Zdrojový kód 3 Kód v Jave potrebný pre volanie funkcie v Jave v JavaScript kóde

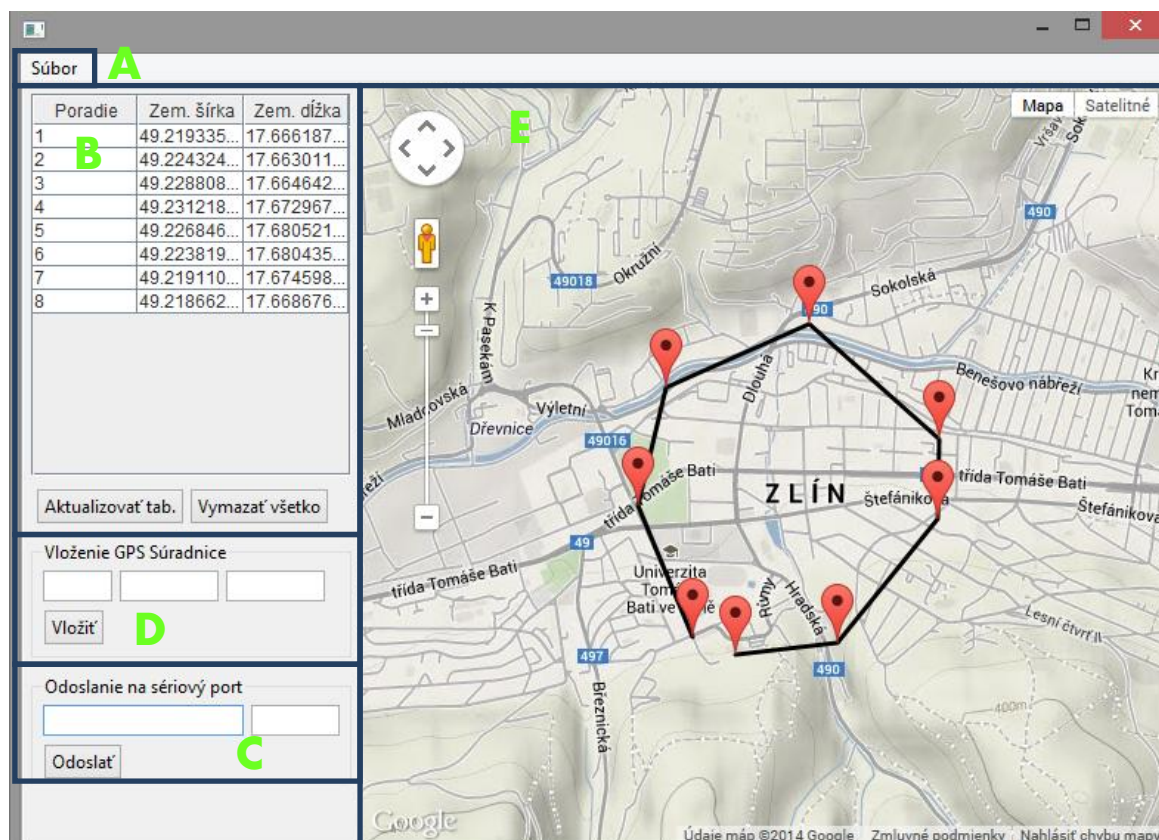
```
theJavaFunction(location.lat(), location.lng());
```

Zdrojový kód 4 Ukážka volania Java funkcie

7.1.2 Užívateľské rozhranie

GUI aplikácie (*Obrázok 11*) pozostáva z piatich základných častí:

- A - Práca so súbormi
- B - Tabuľka súradníc a práca s ňou
- C - Rozhranie pre sériovú komunikáciu
- D - Rozhranie pre vkladanie značiek
- E - Mapa



Obrázok 11 Uživatelské rozhranie desktopovej aplikácie

7.1.2.1 Práca so súbormi

Aplikácia umožňuje načítavanie a ukladanie do súborov s koncovkou .txt. Aby aplikácia vedela rozoznať dáta musí byť text v súbore vo formáte:

START,1,LAT1,LON1,2,LAT2,LON2,3,.....,N,LATN,LONN,END

Po načítaní zo súboru sa trasa uloží do tabuľky. V tomto istom formáte sú aj ukladané zemepisné súradnice trasy do súboru a posielané na sériový port.

7.1.2.2 Tabuľka súradníc a jej obsluha

Tabuľka obsahuje tri stĺpce: "Poradie", "Zem. šírka", "Zem. dĺžka". Po stlačení tlačidla „Aktualizovať tab.“ sa pri zmene dát v tabuľke, prejaví zmena aj na mape. Tlačidlom „Vymazať všetko“ sa vymažú všetky údaje z tabuľky aj značky z mapy.

7.1.2.3 Rozhranie pre vkladanie značiek

Rozhranie „Vloženie GPS súradnice“ obsahuje tri textové polia, do ktorých sa vpisuje poradie, zemepisná šírka a zemepisná dĺžka a táto nová súradnica sa po stlačení tlačidla "Vložiť" vloží do tabuľky a do mapy.

7.1.2.4 Rozhranie pre sériovú komunikáciu

Rozhranie obsahuje dve textové polia. Jedno pre sériový port, druhé pre nastavenie baud rate. Pre odoslanie trasy na sériovú linku treba do prvého textového poľa napísať, na ktorý sériový port dáta odoslať a po stlačení tlačidla „Odoslať“ sa dáta pošlú. Hodnota baud rate nie je povinná, prednastavená je na hodnotu 9600.

7.1.2.5 Mapa

Mapa slúži rovnako pre vizualizáciu trasy, ako aj pre vytvorenie jej návrhu. Pri kliknutí ľavým tlačidlom myši sa pridá nová značka na mapu v mieste kliknutia. Značky sú pospájané čiarami, pre lepšiu predstavu trasy. Pre informácie o značke stačí kliknúť pravým tlačidlom na požadovanú značku a objaví sa nad ňou okno s informáciami o poradí, zemepisnej šírke a dĺžke značky. Pri kliknutí ľavým tlačidlom na značku sa značka odstráni z mapy aj z tabuľky.

7.2 GPS navigácia

Hlavným bodom práce bolo vytvorenie návrhu a vývoj aplikácie slúžiacej ako navigácia pre robotické systémy. Aplikácia je napísaná v jazyku C. Navigácia pozostáva z viacero vytvorených funkcií, a tie sú obsiahnuté v knižnici *nav.h*. Funkcia *main()* v aplikácii volá funkcie knižnice *nav.h*. V nasledujúcich častiach sú popísané jednotlivé časti knižnice *nav.h* a ich funkcionality. Pre rozbor dát prijímaných z GPS prijímača bola použitá knižnica NMEA library, ktorá je voľne publikovaná pod licenciou GNU Lesser GPLv2.1.

7.2.1 Štruktúry knižnice *nav.h*

Knižnica *nav.h* obsahuje jedinú štruktúru a to štruktúru *navigation*.

7.2.1.1 Štruktúra *navigation*

Členmi štruktúry sú *strana* a *uhol*. Premenná *strana* je vlastne smer do ktorého sa otočiť, teda doľava či doprava a premenná *uhol* je vlastne uhol, o ktorý sa treba otočiť. [8]

```
typedef struct navigation {  
    char* strana;  
    double uhol;  
} navigation;
```

Zdrojový kód 5 Štruktúra navigation

7.2.2 Funkcie knižnice nav.h

7.2.2.1 Funkcia *na_radiany()*

Funkcia *na_radiany()* slúži na prevod uhla vyjadreného v stupňoch na uhol vyjadrený v radiánoch. Vstupným parametrom funkcie je uhol *uholStp* vyjadrený v stupňoch a výstupom je ten istý uhol, vyjadrený v radiánoch.

```
double na_radiany(double uholStp);
```

Zdrojový kód 6 Prototyp funkcie na_radiany()

7.2.2.2 Funkcia *na_stupne()*

Funkcia *na_stupne()* slúži na prevod uhla vyjadreného v radiánoch na uhol vyjadrený stupňoch. Vstupným parametrom je uhol *uholRad* vyjadrený v radiánoch a výstupom je ten istý uhol vyjadrený v stupňoch.

```
double na_stupne(double uholRad);
```

Zdrojový kód 7 Prototyp funkcie na_stupne()

7.2.2.3 Funkcia *distance()*

Funkcia *distance()* slúži na výpočet vzdialenosti medzi zemepisnými súradnicami, podľa vzorca:

$$\text{dist} = \text{acos}(\cos(\text{lat1}) * \cos(\text{lat2}) * \cos(\text{lon1} - \text{lon2}) + \sin(\text{lat1}) * \sin(\text{lat2}))) * ((3.141592653589793 * 6371000)/180 \quad [13]$$

Funkcia najprv prevedie zemepisnú šírku a dĺžku pozícií na radiány, pretože goniometrické funkcie knižnice *math.h* pracujú s uhlami v radiánoch. Vstupnými parametrami funkcie sú zemepisné súradnice *pos1* a *pos2* a výstupom je vzdialenosť medzi nimi vyjadrená v metroch.


```
double distance(nmeaPOS pos1, nmeaPOS pos2);
```

Zdrojový kód 8 Prototyp funkcie distance()

7.2.2.4 Funkcia *azimut()*

Funkcia *azimut()* slúži na výpočet azimutu podľa vzorca:

$$\text{Azimut} = \text{atan2}(\cos(\text{lat1}) * \sin(\text{lat2}) - \sin(\text{lat1}) * \cos(\text{lat2}) * \cos(\text{lon2} - \text{lon1}), \sin(\text{lon2} - \text{lon1}) * \cos(\text{lat2})) * 180) / 3.141592653589793 \text{ [13]}$$

Rovnako ako aj funkcia *distance()*, tiež prepočítava zemepisné súradnice na radiány. Vstupnými parametrami funkcie sú pozície *pos1* a *pos2* a výstupom je azimut.

```
double vypocitaj_azimut(nmeaPOS pos1, nmeaPOS pos2);
```

Zdrojový kód 9 Prototyp funkcie azimut()

7.2.2.5 Funkcia *nmeaPOS_to_angle_in_deg()*

Funkcia *nmeaPOS_to_angle_in_deg()* slúži na prevod zemepisnej šírky a dĺžky z tvaru ddmm.mmm, ktorý vracia funkcia *nmea_parse()* z knižnice *parser.h* do tvaru dd.dddd. Vstupným parametrom funkcie je zemepisná súradnica *position* a výstupom je súradnica popísaná zem. šírkou a dĺžkou v tvare dd.dddd. [8]

```
nmeaPOS nmeaPOS_to_angle_in_deg(nmeaPOS position);
```

Zdrojový kód 10 Prototyp funkcie nmeaPOS_to_angle_in_deg()

7.2.2.6 Funkcia *set_port()*

Funkcia *set_port()* slúži na nastavenie sériového portu. Nastavenie sa prevádza pomocou volania funkcie *system()*, ktorá je schopná volať systémové príkazy. Vstupným parametrom funkcie je sériový port *port*.

```
void set_port(char *port);
```

Zdrojový kód 11 Prototyp funkcie set_port()

7.2.2.7 Funkcia *set_mode()*

Funkcia *set_mode()* slúži na výber medzi režimami. Vstupnými parametrami funkcie sú *cntpar* a *params*, čo sú v podstate vstupné parametre funkcie *main()*, a výstupom je

premenná typu *int*, ktorej hodnota je 0, ak je zadáný neznámy parameter, 1, ak je zvolený režim *nav* alebo 2, ak je zvolený režim *addtrack*.

```
int set_mode(int cntpar, char *params[]);
```

Zdrojový kód 12 Prototyp funkcie set_mode()

7.2.2.8 Funkcia *mode_addtrack()*

Funkcia *mode_addtrack()* slúži na uloženie prijatej trasy zo sériovej linky do textového súboru. Sériový port a názov textového súboru sú vstupnými parametrami. Vstupnými parametrami funkcie sú *cntpar* a *params*, čo sú v podstate vstupné parametre funkcie *main()*.

```
void mode_addtrack(int cntpar, char *params[]);
```

Zdrojový kód 13 Prototyp funkcie mode_addtrack()

7.2.2.9 Funkcia *mode_nav()*

Funkcia *mode_nav()* (ukážka kódu je v prílohe Príloha I) slúži ako navigácia. V slučke *while()* sa do premennej *posTmp* typu *nmeaPOS* ukladajú aktuálne súradnice, ak sa prijímajú skutočné dáta z GPS prijímača alebo body nasimulovanej trasy. Následne počíta vzdialenosť medzi *posActual* a *posTmp* a zisťuje, či posledná zmena bola väčšia aspoň ako pol metra. Ak áno, predchádzajúca poloha *posPrevious* sa prepíše doterajšou aktuálnou a aktuálna poloha *posActual* sa prepíše práve nameranou *posTmp*. Vypočíta sa aktuálny azimut a azimut, ktorého by sme sa mali držať, ak chceme dosiahnuť cieľ. Z rozdielu týchto azimutov sa vypočíta uhol. Podľa tohto uhla sa určí, do ktorej strany by sme sa mali otočiť a o aký uhol. V premennej *acc* je hodnota, ktorá je normálne prednastavená na 1, čiže 1 m, ale možno ju nastaviť pri spúšťaní aplikácie, ak zadáme napr. *-acc 2*, nastaví sa hodnota premennej *acc* na 2. A táto hodnota sa chápe ako maximálna vzdialenosť medzi aktuálnou pozíciou a nasledujúcim záchytným bodom, pri ktorej sa považuje záchytný bod za dosiahnutý. Ak je záchytný bod dosiahnutý, inkrementuje sa hodnota premennej *j*, pomocou ktorej sa pohybujeme v poli záchytných bodov *destinacie[]*. Ak je záchytný bod posledný v poli, slučka *while()* sa preruší príkazom *break*. Funkcia informuje o priebehu navigácie a tieto isté informácie ukladá do logovacieho textového súboru, ktorý je zadávaný za parametrom *-flog*. Vstupnými parametrami funkcie sú *cntpar* a *params*, čo sú v podstate vstupné parametre funkcie *main()*. [8]

```
void mode_nav(int cntpar, char *params[]);
```

Zdrojový kód 14 Prototyp funkcie mode_nav()

7.3 Demonštrácia systému

Celý systém je navrhnutý tak, aby mohla navigácia pracovať sama bez nutnosti bezprostredného nahrávania trasy pred navigáciou. Pre toto riešenie boli navrhnuté dva režimy. Prvý režim umožňuje uloženie trasy do súboru na pamäťové miesto v mikropočítači. Druhý režim je samotná navigácia, ktorá využíva túto trasu na navigovanie. Navigácia môže pracovať aj v testovacom režime, kedy namiesto skutočných hodnôt polohy z GPS prijímača, pracuje s nasimulovanou trasou uloženou v textovom súbore.

V zdrojovom kóde *Zdrojový kód 15* je ukázané použitie funkcií z knižnice *nav.h* v hlavnom programe vo funkcii *main()*. Program sa rozhoduje na základe vstupných parametrov (*Tabuľka 7*), ktoré sa predávajú ako parametre funkcie *set_mode* a tá podľa nich vráti hodnotu, podľa ktorej sa rozhodne, ktorý z dvoch režimov sa spustí.

```
int main(int argc, char *argv[])
{
    int mode = set_mode(argc, argv);
    if(mode == 2) mode_addtrack(argc, argv);
    else if(mode == 1) mode_nav(argc, argv);
    return 0;
}
```

Zdrojový kód 15 Ukážka použitia funkcií z knižnice nav.h

Tabuľka 7 Popis vstupných parametrov

parameter	Popis
-mode	Výber režimu z dvoch možností "nav" alebo "addtrack"
-ftrack	Súbor, kde sa v režime "addtrack" uloží trasa
-srctrack	Zdroj trasy
-flog	Logovací súbor – dostupný len v režime "nav"
-acc	Presnosť v metroch – dostupný len v režim "nav", nepovinný parameter

7.3.1 Proces tvorby trasy a jej odoslanie mikropočítaču

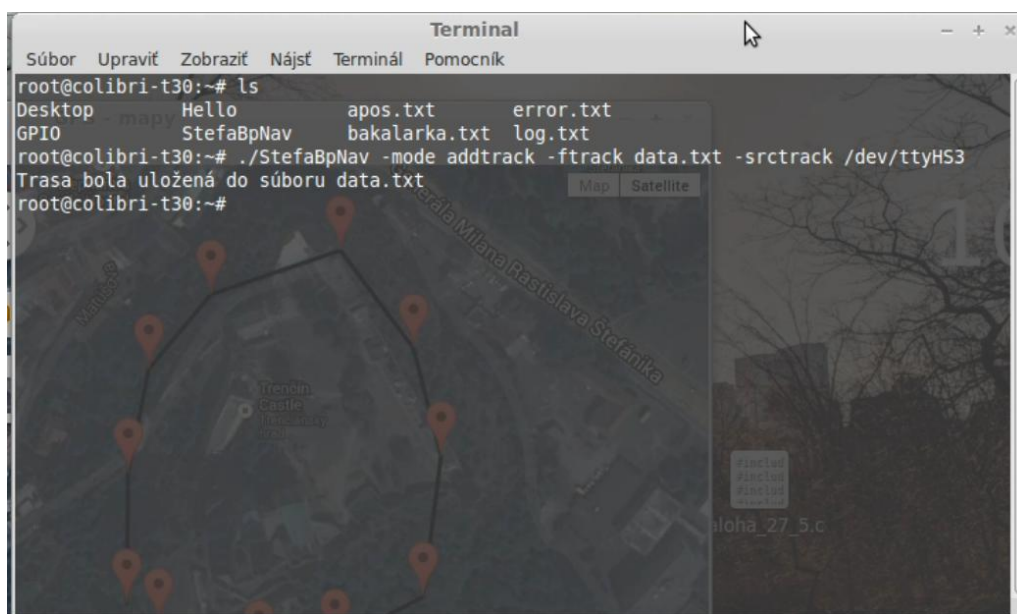
Trasa sa vytvorí pomocou desktopovej aplikácie, buď postupným pridávaním značiek kliknutím na mapu alebo pridávaním značiek pomocou rozhrania „Vloženie GPS pozície“.

Trasu možno aj načítať zo súboru s už vytvorenou trasou. Vytvorená trasa sa pošle na sériový port, ktorý zadáme do poľa v rozhraní „Odoslanie na sériový port“.

Ďalej napríklad pomocou protokolu SSH sa pripojíme cez terminál k mikropočítaču a zadáme príkaz:

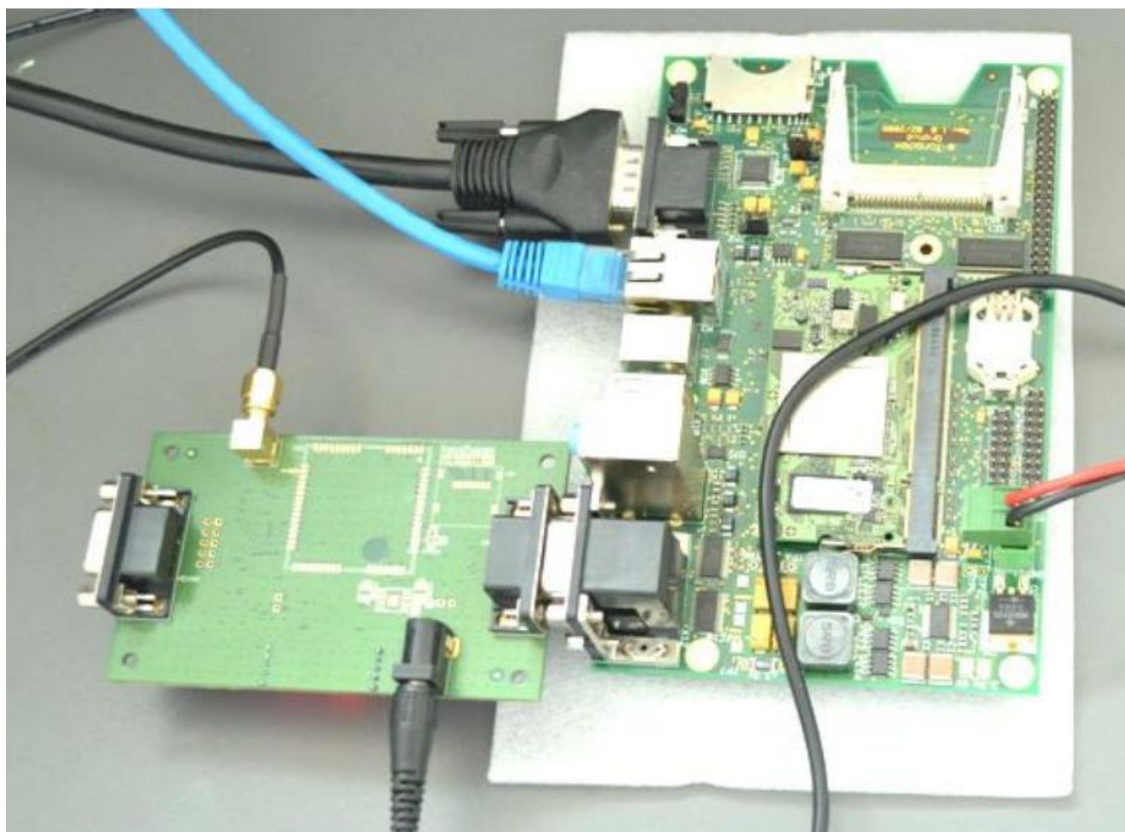
```
./StefaBpNav -mode addtrack -ftrack [txt súbor] -srctrack [seriový port]
```

Príklad zadania príkazu je na obrázku *Obrázok 12*. Do súboru *data.txt* sa uložia dáta prijaté zo sériového portu */dev/ttyHS3*. Ak sa odosielanie podarilo, vypíše sa na terminál „Trasa bola uložená do súboru data.txt“.



Obrázok 12 Ukážka režimu addtrack

7.3.2 Proces navigácie

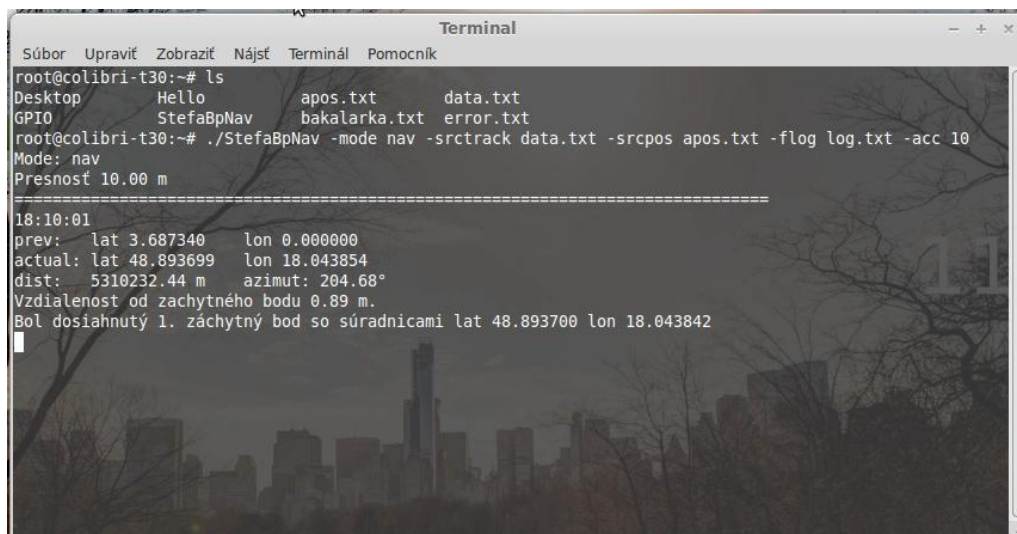


Obrázok 13 Ukážka reálneho zapojenia systému pri procese navigácie

Pre spustenie navigácie zadáme príkaz:

```
./StefaBpNav -mode nav -srtrack [txt súbor/ sériový port] -srcpos [seriový port] -flog  
[txt súbor] -acc [číslo]
```

Príklad zadania príkazu je na obrázku *Obrázok 14*. Aplikácia zo súboru *data.txt* berie súradnice trasy, po ktorej by sme sa mali pohybovať a zo súboru *apos.txt* berie súradnice nasimulovanej trasy pohybu. Pre získanie súradníc reálnej polohy sa pripojí GPS prijímač a za *-srcpos* sa napíše sériový port, na ktorý je prijímač pripojený. Do súboru *log.txt* sú zaznamenávané rovnaké výpisy, aké sú popisované nižšie. Parameter *-acc* je nastavený na 10, čo znamená presnosť je 10 metrov.



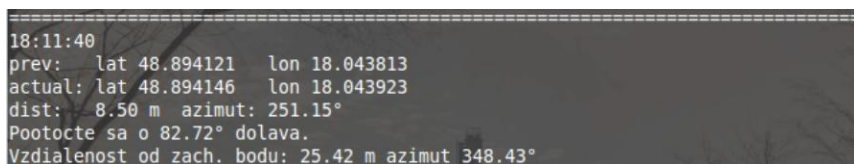
```
Terminal
Súbor Upraviť Zobrazíť Nájsť Terminál Pomocník
root@colibri-t30:~# ls
Desktop      Hello        apos.txt     data.txt
GPIO         StefaBpNav   bakalarka.txt error.txt
root@colibri-t30:~# ./StefaBpNav -mode nav -srctrack data.txt -srcpos apos.txt -flog log.txt -acc 10
Mode: nav
Presnosť 10.00 m

=====
18:10:01
prev: lat 3.687340 lon 0.000000
actual: lat 48.893699 lon 18.043854
dist: 5310232.44 m azimut: 204.68°
Vzdialenosť od zachytného bodu 0.89 m.
Bol dosiahnutý 1. záchytný bod so súradnicami lat 48.893700 lon 18.043842
```

Obrázok 14 Ukážka režimu nav

Aplikácia hneď po spustení začína navigovať. Rozlišujeme tri typy výpisov. Každý výpis informuje o aktuálnom čase, aktuálnej a predchádzajúcej polohe, vzdialenosti medzi nimi a aktuálnom azimute.

Prvý typ zápisu informuje o tom, do ktorej strany a o aký uhol sa treba otočiť, aby sme dosiahli cieľ. Ďalej informuje o vzdialenosti od záchytného bodu a o azimute medzi aktuálnou pozíciou a záchytným bodom (Obrázok 15).

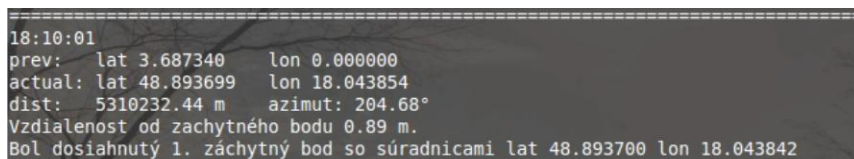


```
=====
18:11:40
prev: lat 48.894121 lon 18.043813
actual: lat 48.894146 lon 18.043923
dist: 8.50 m azimut: 251.15°
Pootoče sa o 82.72° dolava.
Vzdialenosť od zach. bodu: 25.42 m azimut 348.43°
```

Obrázok 15 Výpis navigovania

Ďalšie dva typy výpisov sú skoro rovnaké a informujú o dosiahnutí záchytného bodu.

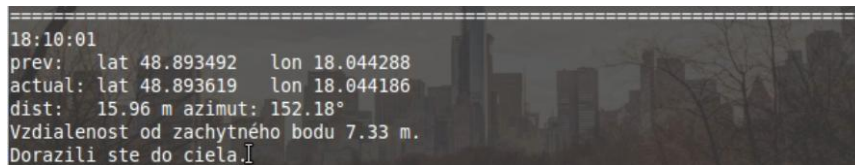
Ak je dosiahnutý záchytný bod, ktorý nie je posledný, vypíše sa napr. „Bol dosiahnutý 1. záchytný bod so súradnicami lat 48.893700 lon 18.043842“, čiže sa vypíšu informácie, ktorý bod v poradí bol dosiahnutý a jeho poloha (Obrázok 16).



```
=====
18:10:01
prev: lat 3.687340 lon 0.000000
actual: lat 48.893699 lon 18.043854
dist: 5310232.44 m azimut: 204.68°
Vzdialenosť od zachytného bodu 0.89 m.
Bol dosiahnutý 1. záchytný bod so súradnicami lat 48.893700 lon 18.043842
```

Obrázok 16 Výpis o dosiahnutí bodu trasy

Ak je dosiahnutý záchytný bod posledný, vypíše sa formulka „Dorazili ste do cieľa“ (Obrázok 17). 1

A screenshot of a terminal window with a dark background and light-colored text. The text displays route completion statistics, including a timestamp, previous and actual coordinates, distance, azimuth, and a confirmation message. The background of the terminal shows a faint city skyline.

```
18:10:01
prev:   lat 48.893492   lon 18.044288
actual: lat 48.893619   lon 18.044186
dist:   15.96 m  azimuth: 152.18°
Vzdialenosť od záchytného bodu 7.33 m.
Dorazili ste do cieľa.█
```

Obrázok 17 Výpis o dosiahnutí posledného bodu trasy

ZÁVER

V teoretickej časti boli rozobrané spôsoby určenia polohy pomocou družíc. Ďalej je vysvetlený spôsob fungovania systému GPS, jeho obecná štruktúra, kódy vysielané družicami, spôsoby určenia polohy a nepresnosť pri určovaní. V teoretickej časti bol ešte popísaný protokol NMEA 0183, prostredníctvom ktorého predáva GPS prijímač informácie o pozícii. V závere teoretickej časti boli popísané embedded systémy a pojmy ako krížový prekladač a vzdialené ladenie, s ktorými sa prichádza do kontaktu pri vývoji aplikácií pre embedded systémy.

Praktická časť sa zaoberala použitým hardvérom a návrhom embedded systému pre GPS navigáciu slúžiacej robotickým systémom.

Systém je navrhnutý tak, aby bolo možné vytvorenú trasu odoslať na sériovú linku, na ktorú bude pripojený mikropočítač s bežiacou aplikáciou. Aplikácia bola napísaná v jazyku C. Mikropočítač túto správu prijme a uloží do súboru a ďalej už nie je nutná komunikácia s desktopom. Trasu možno vytvoriť a odoslať cez desktopovú aplikáciu, ktorej vytvorenie bolo súčasťou tejto práce a bola napísaná v jazyku Java. V režime navigácie potrebuje aplikácia pre svoj beh prijatú vytvorenú trasu a zdroj aktuálnej polohy, teda GPS prijímač. Aplikácia porovnáva aktuálnu polohu s nasledujúcou pozíciou trasy a snaží sa usmerniť pohybovaný objekt na miesto určenia pomocou informácií o uhle a smeru pootočenia tak, aby bol toto miesto určenia dosiahnuté. Aplikácia informuje aj o aktuálnej a predchádzajúcej polohe a vzdialenosti medzi nimi a aktuálnom azimute. Ďalej podáva informácie o vzdialenosti od záchytného bodu a odporúčanom azimute. Jedným z bodov bolo, aby sa tieto informácie ukladali na pamäťovú kartu. Toto bolo vyriešené tak, že tieto informácie ukladá do textového súboru zadaného pri spustení navigácie, ktorého umiestenie môže byť aj na pamäťovej karte.

Aplikácia bola spúšťaná na mikropočítači Colibri T30 s procesorom nVidia Tegra 3. Na mikropočítači bol operačný systém Linux. O zisťovanie polohy sa staral GPS modul L70 od firmy Quectel.

Výsledky práce sú dostupné pod licenciou GPLv2.

ZOZNAM POUŽITEJ LITERATURY

- [1] RAPANT, Petr. *Družicové polohové systémy*. Ostrava: Vysoká škola báňská - Technická univerzita, 2002, 197 s. ISBN 80-248-0124-8.
- [2] *Globálne navigačné systémy* [online]. 2005 [cit. 2014-05-18]. ISBN 80-8070-542-9. Dostupné z: http://svf.utc.sk/kgd/skripta/Globalne_navigacne_systemy.pdf
- [3] *Družicové navigačné systémy* [online]. 2009 [cit. 2014-05-18]. ISBN 978-80-89282-31-9. Dostupné z: <http://mafiajara.stranky.cc/4.%20rocnik/druzicove%20navigacne%20systemy/dns.pdf>
- [4] DLOUHÝ, Martin. GPS. *Robotika.cz* [online]. 2006 [cit. 2014-05-15]. Dostupné z: <http://robotika.cz/guide/gps/cs>
- [5] MARTÍNEK, Jan. GPS a komunikační protokol NMEA. *AbcLinuxu* [online]. 2006 [cit. 2014-05-15]. Dostupné z: <http://www.abclinuxu.cz/clanky/ruzne/gps-a-komunikacni-protokol-nmea-3-dekodovani-dat>
- [6] *Quectel L70*. [cit. 2014-06-04]. Dostupné z: http://www.quectel.com/UploadFile/Product/Quectel_L70_GPS_Specification_V2.1.pdf
- [7] NVIDIA Tegra 3 Computer on Module - Colibri T30. *Toradex* [online]. © 2014 [cit. 2014-06-09]. Dostupné z: <http://www.toradex.com/products/colibri-arm-computer-modules/nvidia-tegra-3-computer-module>
- [8] PRATA, Stephen. *Mistrovství v C++*. 3. aktualiz. vyd. Překlad Boris Sokol. Brno: Computer Press, 2007, 1119 s. ISBN 978-80-251-1749-1.
- [9] MASTERS, Jon a Richard BLUM. *Linux profesionálně: programování aplikací*. Vyd. 1. Brno: Zoner Press, 2008, 539 s. ISBN 978-80-86815-71-8.
- [10] BARR, Michael. *Programming embedded systems in C and C++*. 1st ed. Sebastopol, Calif.: O'Reilly, c1999, xvii, 174 p. ISBN 15-659-2354-5.
- [11] MATLOFF, Norman S. *The art of debugging with GDB, DDD, and Eclipse*. San Francisco: No Starch Press, c2008, xiv, 264 s. ISBN.

- [12] Ladění (programování). In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2014-06-06]. Dostupné z:
[http://cs.wikipedia.org/wiki/Lad%C4%9Bn%C3%AD_\(programov%C3%A1n%C3%AD\)](http://cs.wikipedia.org/wiki/Lad%C4%9Bn%C3%AD_(programov%C3%A1n%C3%AD))
- [13] Distance. *SunEarthTools.com* [online]. © 2009-2014 [cit. 2014-06-06].
Dostupné z: <http://www.sunearthtools.com/tools/distance.php>
- [14] DOŠEK, Roman. *OS Linux na platformě ARM*. Zlín, 2013. Bakalářská práce. Univerzita Tomáše Bati ve Zlíně.
- [15] Embedded operating system. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2014-06-09].
Dostupné z: http://en.wikipedia.org/wiki/Embedded_operating_system

ZOZNAM POUŽITÝCH SYMBOLOV A SKRATIEK

GPS	Global navigation system
NMEA	National Marine Electronics Association
GGA	Global Positioning System Fix Data
RMC	Recommended minimum data for gps - minimálne informácie potrebné pre GPS
GSV	GPS Satellites in view - videné satelity
GSA	GPS DOP and active satellites - DOP a aktívne satelity
UTC	Coordinated Universal Time - koordinovaný svetový čas
PRN	Pseudo-random noise - pseudonáhodný šum
DOP	Dilution of precision - zníženie presnosti
DGPS	Differential GPS - difrenčné GPS
RTOS	Real-time operating system - operačný systém reálneho času
RAM	Random Access Memory - typ pamäte
ROM	Read Only Memory - typ pamäte
GDB	GNU Debugger - typ ladiaceho nástroja
AGPS	Assisted GPS
EVB	Evaluation Board
CPU	Central Processor Unit - centrálna procesorová jednotka
ARM	Advanced RISC Machine - architektúra procesorov
eMMC	MultiMediaCard - druh pamäťovej karty
SD karta	Secure Digital karta - druh pamäťovej karty
USB	Universal serial bus - univerzálna sériová zbernica
RS232	Sériový port
RJ45	Elektrický konektor využívaný v počítačových sieťach
VGA	Video Graphics Array - štandard pre zobrazovaciu techniku

GUI	Graphical User Interface - grafické uživatelské prostředí
GNU	General Public Licence
URL	Uniform Resource Locator - jednotná adresa zdroja
SSH	Security Shell - sieťový protokol

ZOZNAM OBRÁZKOV

<i>Obrázok 1 Určovanie smeru</i>	<i>15</i>
<i>Obrázok 2 Schéma odvodzovania frekvencií jednotlivých signálov [3]</i>	<i>18</i>
<i>Obrázok 3 Štruktúra navigačnej správy [1]</i>	<i>20</i>
<i>Obrázok 4 Základná schéma GPS prijímača [2]</i>	<i>22</i>
<i>Obrázok 5 GPS modul L70 a jeho rozmery [6]</i>	<i>30</i>
<i>Obrázok 6 GPS modul L70 na EVB doske</i>	<i>31</i>
<i>Obrázok 7 Mikropočítač Colibry T30 [7]</i>	<i>31</i>
<i>Obrázok 8 Orchid carrier board [7]</i>	<i>32</i>
<i>Obrázok 9 Návrh systému</i>	<i>33</i>
<i>Obrázok 10 Štruktúra komponentov a ich popis</i>	<i>34</i>
<i>Obrázok 11 Užívateľské rozhranie desktopovej aplikácie</i>	<i>37</i>
<i>Obrázok 12 Ukážka režimu addtrack</i>	<i>43</i>
<i>Obrázok 13 Ukážka reálneho zapojenia systému pri procese navigácie</i>	<i>44</i>
<i>Obrázok 14 Ukážka režimu nav</i>	<i>45</i>
<i>Obrázok 15 Výpis navigovania</i>	<i>45</i>
<i>Obrázok 16 Výpis o dosiahnutí bodu trasy</i>	<i>45</i>
<i>Obrázok 17 Výpis o dosiahnutí posledného bodu trasy</i>	<i>46</i>

ZOZNAM TABULIEK

<i>Tabuľka 1 Zdroje chýb a ich príspevok k celkovej chybe [3]</i>	22
<i>Tabuľka 2 Popis vety GSA [4]</i>	24
<i>Tabuľka 3 Popis vety RMC [4]</i>	25
<i>Tabuľka 4 Popis vety GSV [4]</i>	26
<i>Tabuľka 5 Popis vety GGA [4]</i>	26
<i>Tabuľka 6 Špecifikácie Colibri T30 [7]</i>	32
<i>Tabuľka 7 Popis vstupných parametrov</i>	42

ZOZNAM ZDROJOVÝCH KÓDOV

<i>Zdrojový kód 1 Ukážka kódu v jazyku JavaScript – funkcia pre vloženie značky</i>	<i>35</i>
<i>Zdrojový kód 2 Ukážka volania JavaScript funkcie v Java</i>	<i>35</i>
<i>Zdrojový kód 3 Kód v Jave potrebný pre volanie funkcie v Jave v JavaScript kóde</i>	<i>36</i>
<i>Zdrojový kód 4 Ukážka volania Java funkcie</i>	<i>36</i>
<i>Zdrojový kód 5 Štruktúra navigation</i>	<i>39</i>
<i>Zdrojový kód 6 Prototyp funkcie na _radiany()</i>	<i>39</i>
<i>Zdrojový kód 7 Prototyp funkcie na _stupne()</i>	<i>39</i>
<i>Zdrojový kód 8 Prototyp funkcie distance()</i>	<i>40</i>
<i>Zdrojový kód 9 Prototyp funkcie azimuth()</i>	<i>40</i>
<i>Zdrojový kód 10 Prototyp funkcie nmeaPOS_to_angle_in_deg()</i>	<i>40</i>
<i>Zdrojový kód 11 Prototyp funkcie set_port()</i>	<i>40</i>
<i>Zdrojový kód 12 Prototyp funkcie set_mode()</i>	<i>41</i>
<i>Zdrojový kód 13 Prototyp funkcie mode_addtrack()</i>	<i>41</i>
<i>Zdrojový kód 14 Prototyp funkcie mode_nav()</i>	<i>42</i>
<i>Zdrojový kód 15 Ukážka použitia funkcií z knižnice nav.h</i>	<i>42</i>

ZOZNAM PRÍLOH

Príloha P I: Ukážka zdrojového kódu funkcie `mode_nav()`

PRÍLOHA P I: UKÁŽKA ZDROJOVÉHO KÓDU FUNKCIE MODE_NAV()

```
int mode_nav(int cntpar, char *params[])
{
    nmea_parser_init(&parser);
    nmea_zero_INFO(&info);
    while(1) {
        if(strstr(nmea_data, ".txt") == 0) {
            size = read(fd2, &buff, sizeof buff); //čítanie zo seriovej linky
            nmea_parse(&parser, &buff[0], size, &info);
            posTmp.lat = nmeaPOS_to_angle_in_deg(info.position).lat;
            posTmp.lon = nmeaPOS_to_angle_in_deg(info.position).lon;
        } else {
            posTmp = poloha[i];
        }

        if(distance(posActual, posTmp) > 0.5) {
            posPrevious = posActual;
            posActual = posTmp;
        }

        azimuthActual = vypocitaj_azimut(posPrevious, posActual);
        azimuthRequired = vypocitaj_azimut(posActual, destinacie[j]);
        nav1.uhol = azimuthRequired - azimuthActual;

        if(nav1.uhol > 180 || (nav1.uhol < 0 && nav1.uhol > -180)) {
            nav1.strana = "dolava";
        } else {
            nav1.strana = "doprava";
        }

        if(nav1.uhol < 0) {
            nav1.uhol = fabs(nav1.uhol);
        }

        if(nav1.uhol > 180) {
            nav1.uhol = 360 - nav1.uhol;
        }
        if(distance(posActual, destinacie[j]) < acc) {
            printf("\nVzdialenosť od zachytaného bodu %.2f m.",
                distance(posActual, destinacie[j]));

            if(j+1 == cntpdest) {
                printf("\nDorazili ste do cieľa. \n");
                break;
            } else {
                printf("\nBol dosiahnutý %d. zachytaný bod so súradnicami lat %f
                    lon %f\n", j+1, destinacie[j].lat, destinacie[j].lon);
            }
            j++;
        } else {
            printf("\nPootočte sa o %.2f° %s."
                "\nVzdialenosť od zach. bodu: %.2f m azimut %.2f°\n",
                nav1.uhol, nav1.strana, distance(posActual, destinacie[j]),
                vypocitaj_azimut(posActual, destinacie[j]));
        }
        i++;
        if(strstr(nmea_data, ".txt") == 0) {}
        else {
            if(poloha[i].lat == 0) {
                break;
            }
        }
    }
    fclose(fileLog);
    fclose(fileActualPos);
    nmea_parser_destroy(&parser);
    return;
}
```