VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV MIKROELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF MICROELECTRONICS

KONSTRUKCE GPS PŘÍSTROJE

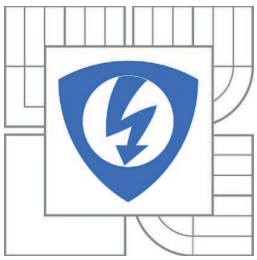
CONSTRUCTION OF GPS DEVICES

DIPLOMOVÁ PRÁCE
MASTER´S THESIS

AUTOR PRÁCE
AUTHOR
Bc. MAREK HORT

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. PAVEL ŠTEFFAN, Ph.D.

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav mikroelektroniky

Diplomová práce

magisterský navazující studijní obor
Mikroelektronika

Student:	Bc. Marek Hort	***ID:***	89398
Ročník:	2	***Akademický rok:***	2009/2010

NÁZEV TÉMATU:

Konstrukce GPS přístroje

POKYNY PRO VYPRACOVÁNÍ:

Navrhněte a sestrojte zařízení, které bude komunikovat s GPS modulem LEA -5s. Souřadnice zjištěné z tohoto modulu uložte do paměti zařízení. Pro realizaci využijte možností platformy FPGA pro maximální urychlení použitých algoritmů. Po propojení zařízení s PC bude možno celou uloženou trasu zobrazit v mapě. Pro demonstraci použijte mapová data volně dostupná na Internetu, případně satelitní snímky poskytované službou Google Earth.

DOPORUČENÁ LITERATURA:

Podle pokynů vedoucího práce

Termín zadání:	8.2.2010	***Termín odevzdání:***	27.5.2010

Vedoucí práce: Ing. Pavel Šteffan, Ph.D.

prof. Ing. Vladislav Musil, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

Licenční smlouva
poskytovaná k výkonu práva užít školní dílo

uzavřená mezi smluvními stranami:

1. Pan/paní

Jméno a příjmení: Bc. Marek Hort

Bytem: Velké Meziříčí, Čechova 34, 594 01

Narozen/a (datum a místo): 17. 7. 1984, Třebíč

(dále jen „autor“)

a

2. Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií

se sídlem Údolní 244/53, 602 00 Brno

jejímž jménem jedná na základě písemného pověření děkanem fakulty:

Prof. Ing. Vladislav Musil, CSc.

(dále jen „nabyvatel“)

Čl. 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
   - □ disertační práce
   - ☒ diplomová práce
   - □ bakalářská práce
   - □ jiná práce, jejíž druh je specifikován jako .....

   (dále jen VŠKP nebo dílo)

Název VŠKP: Konstrukce GPS přístroje

Vedoucí/ školitel VŠKP: Ing. Pavel Šteffan, Ph.D.

Ústav: Ústav mikroelektroniky

Datum obhajoby VŠKP:

VŠKP odevzdal autor nabyvateli v:

- ☒ tištěné formě – počet exemplářů 2
- ☒ elektronické formě – počet exemplářů 2

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizovaní výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti
   - ☒ ihned po uzavření této smlouvy
   - □ 1 rok po uzavření této smlouvy
   - □ 3 roky po uzavření této smlouvy
   - □ 5 let po uzavření této smlouvy
   - □ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: 24. 5. 2010

…….. ……

Nabyvatel Autor

Abstrakt:

Cílem předkládané diplomové práce bylo vytvořit zařízení schopné přijímat navigační data ze systému GPS. Tyto data následně ukládat do vlastní paměti a po připojení k PC je zobrazit na satelitní mapě. Zařízení bylo realizováno za pomocí obvodu FPGA a GPS modulu LEA -5s. Byl vytvořen popis v jazyce VHDL, který byl do obvodu implementován. Součástí VHDL designu byl i popis procesoru PICOBLAZE, který řídí celé zařízení. Pro zobrazení a archivaci dat uložených v zařízení byla vytvořena PC aplikace GPS TRACER. Ta je schopna prostřednictvím GOOGLE MAPS zobrazit uloženou trasu na satelitní mapě. Pro vytvářené zařízení byly navrženy a zhotoveny desky plošných spojů, které byly následně ručně osazeny.

Abstract:

Aim of this Diploma thesis was to create a device capable of receiving navigational data from GPS. These data are subsequently stored in fixed memory and after connection with the PC are displayed it on the satellite map. The device was realized by using FPGA and GPS module LEA-5s. Description was created in the VHDL language, which was implemented into the circuit. The part of VHDL design was description of PICOBLAZE processor that controls whole system. For displaying and archiving data stored in device was created PC application GPS TRACER. It is able to display stored trace on the satellite map by using Google maps server. For created device were designed and manufactured PCBs, which were manually fitted.

Klíčová slova:

GPS modul, FPGA, PICOBLAZE, VHDL, aplikace pro PC, desky plošných spojů.

Keywords:

GPS modul, FPGA, PICOBLAZE, VHDL, PC application, printed circuit boards.

Bibliografická citace díla:

HORT, M. Konstrukce GPS přístroje . Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2010. 79 s. Vedoucí diplomové práce Ing. Pavel Šteffan, Ph.D.

Prohlášení autora o původnosti díla:

Prohlašuji, že jsem tuto vysokoškolskou kvalifikační práci vypracoval samostatně pod vedením vedoucího diplomové práce, s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury. Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne 24. 5. 2010

…….

Poděkování:

Děkuji vedoucímu diplomové práce Ing. Pavlu Šteffanovi, Ph.D. za metodické a cíleně orientované vedení při plnění úkolů realizovaných v průběhu zpracovávání diplomové práce.

OBSAH

1 Úvod

Podnětem pro vznik této práce byla potřeba vlastnit zařízení, jež bude schopno shromažďovat a zpracovávat informace získané během osobních běžeckých nebo cyklistických tréninků. Jednalo se především o určení trasy a časovou závislost rychlosti vlastního pohybu. Na současném trhu existuje spousta zařízení splňujících tyto požadavky. Jejich společnou vlastností je využití systému GPS jako zdroje navigačních dat. Avšak jejich cena se pohybuje řádově v tisících korunách. A to je hlavní důvod vzniku této práce. Vytvořit obdobné zařízení, pracující se systémem GPS, jehož cena nepřesáhne hranici jednoho tisíce korun. Komerčním přístrojům tohoto druhu lze jen stěží konkurovat především v konstrukčních rozměrech, přičemž mnohdy se jedná o zařízení velikosti náramkových hodinek. Toto kritérium však nebylo při návrhu tak zásadní.

Počátek elektronického určování polohy lze nalézt již v období 2. světové války, kdy byly vyvinuty první naváděcí systémy pro letectvo a námořnictvo. V průběhu druhé poloviny minulého století bylo následně vyvinuto několik obdobných systémů, které byly s větší či menší přesností schopny určit polohu. Vždy se však jednalo o systémy pozemního charakteru s omezeným prostorovým rozsahem. Až v době kosmických letů spojených i s vývojem umělých družic Země vzniká první globální navigační systém, dnes označovaný jako GPS.

Historie systému GPS sahá do roku 1973, kdy byly sloučeny dva oddělené výzkumy v oblasti satelitního určení polohy. Jednalo se o systém 621B amerického letectva a TIMATION amerického námořnictva. Od roku 1978 bylo zahájeno postupné vypouštění 18 plánovaných družic, jenž měly být umístěny na zemském orbitu. Tento počet se však záhy ukázal jako nedostatečný a tak byl původní plán navýšen na celkových 24 satelitů. Tohoto počtu bylo dosaženo počátkem roku 1994 a tento stav je označován za plnou operační dostupnost. Pro civilní účely byl však systém stále nedostupný, respektive jeho přesnost byla velice malá. Až polovina roku 2000 přinesla uvolnění systému pro civilní sektor, což je označováno jako zrušení selektivní dostupnosti. Tato skutečnost vyvolala obrovský rozmach obvodů a zařízení pracujících s navigačními údaji. Jedná se především o navigaci v dopravě a turistice, záznamové systémy, lokátory, ale také geodetické přístroje.

2 Teoretická část

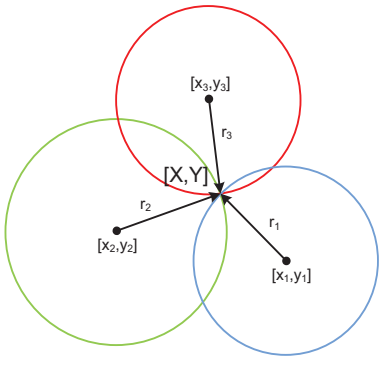
2.1 Globální navigační družicový systém

Obecně lze říct, že globální navigační družicový systém (dále jen GNSS z anglického Global Navigation Satellite System) je systém umožňující kdekoliv na Zemi určit přesnou polohu a čas. Principem zařízení je příjem radiového signálu vysílaného ze skupiny družic, obíhajících po přesných drahách Země. Na tomto principu je založeno několik GNSS systémů, avšak pouze jeden je v současné době plně funkční. Tento systém je často označován jako GPS (Global positioning system).

2.1.1 Obecný princip určení polohy

Pro stanovení neznámé pozice objektu v jedné rovině, například na mapě, je možné využít takzvaného dálkoměrného systému. K určení polohy tímto způsobem je zapotřebí znát pozici tří výchozích bodů a jejich vzdálenost od zjišťované polohy. Obr. 1 znázorňuje trojici kružnic, jejichž počátek je v pozici výchozích bodů a poloměr těchto kružnic je roven jejich vzdálenosti od stanovovaného místa. Výsledný průsečík stanovuje hledanou polohu. Tento způsob je však pouze dvou rozměrný, jelikož výchozí i určovaný bod se nacházejí v jedné rovině. Matematicky lze toto řešení popsat soustavou tří rovnic kružnic se známou polohou jejich středů x_n, y_n a jejich poloměrem r_n [1]. Výpočtem hodnot X a Y lze vyjádřit pozici bodu v jedné rovině.

$$\begin{aligned}(X - x_1)^2 + (Y - y_1)^2 &= {r_1}^2 \\ (X - x_2)^2 + (Y - y_2)^2 &= {r_2}^2 \\ (X - x_3)^2 + (Y - y_3)^2 &= {r_3}^2\end{aligned} \tag{1}$$



Obr. 1: Grafické znázornění stanovení polohy v jedné rovině pomocí průsečíku tří kružnic

Chceme li určit polohu v prostoru, jen nutné znát pozici alespoň tří výchozích bodů, které se nebudou nacházet ve stejné rovině jako určovaný bod. Hledaná pozice je pak dána

průsečíkem plášťů tří koulí, jejichž středy jsou ve výchozích bodech a poloměry jsou vzdáleností výchozích bodů a určovaného místa. Tímto způsobem mohou vzniknout průsečíky dva, ovšem pří vhodné volbě výchozích bodů, lze vždy pouze jeden považovat za reálný. Matematické vyjádření je soustavou rovnic 2. Výpočtem hodnot *X*, *Y*, *Z* z této soustavy tří rovnic lze stanovit pozici určovaného bodu. [1]

$$\begin{aligned}(\mathrm{X}-\mathrm{x}_1)^2+(\mathrm{Y}-\mathrm{y}_1)^2+(\mathrm{Z}-\mathrm{z}_1)^2&=\mathrm{r}_1{}^2\\(\mathrm{X}-\mathrm{x}_2)^2+(\mathrm{Y}-\mathrm{y}_2)^2+(\mathrm{Z}-\mathrm{z}_2)^2&=\mathrm{r}_2{}^2\\(\mathrm{X}-\mathrm{x}_3)^2+(\mathrm{Y}-\mathrm{y}_3)^2+(\mathrm{Z}-\mathrm{z}_3)^2&=\mathrm{r}_3{}^2\end{aligned} \qquad (2)$$

2.1.2 Kódové určení polohy

Tento způsob vychází z popisovaného dálkoměrného systému určení polohy v prostoru. Jako výchozí orientační body slouží družice na oběžné dráze kolem Země, jejichž pozice je přesně stanovena a známá (body *x*, *y*, *z*). Pro určení jejich vzdálenosti je pak využito časového zpoždění radiového signálu, který družice vysílá. Pokud známe časové zpoždění přijímaného signálu a rychlost, jakou se signál šíří, lze snadno určit vzdálenost družice. Je- li rychlost šíření signálu definována jako *v* a zpoždění přijímaného signálu Δt, lze vytvořit soustavu rovnic 3 [1].

$$\begin{aligned}(\mathrm{X}-\mathrm{x}_1)^2+(\mathrm{Y}-\mathrm{y}_1)^2+(\mathrm{Z}-\mathrm{z}_1)^2&=(\mathrm{v}.\Delta\mathrm{t}_1)^2\\(\mathrm{X}-\mathrm{x}_2)^2+(\mathrm{Y}-\mathrm{y}_2)^2+(\mathrm{Z}-\mathrm{z}_2)^2&=(\mathrm{v}.\Delta\mathrm{t}_2)^2\\(\mathrm{X}-\mathrm{x}_3)^2+(\mathrm{Y}-\mathrm{y}_3)^2+(\mathrm{Z}-\mathrm{z}_3)^2&=(\mathrm{v}.\Delta\mathrm{t}_3)^2\end{aligned} \qquad (3)$$

Aby bylo možné stanovit přesné zpoždění přijímaného signálu z jednotlivých družic, je nutné, aby údaj o čase v přijímači byl naprosto synchronní s časem družice. Tato skutečnost je však velice obtížně dosažitelná, proto je čas přijímače také neznámou s označením *T*. Matematické řešení pak vede na soustavu čtyř rovnic 4 s neznámými *X*, *Y*, *Z*, *T*. Pro přesné stanovení polohy v prostoru kódovým mechanizmem systému GNSS je tedy zapotřebí příjem signálu alespoň čtyř satelitů [1].

$$\begin{aligned}(\mathrm{X}-\mathrm{x}_1)^2+(\mathrm{Y}-\mathrm{y}_1)^2+(\mathrm{Z}-\mathrm{z}_1)^2&=[\mathrm{v}.(\mathrm{T}-\mathrm{t}_1)]^2\\(\mathrm{X}-\mathrm{x}_2)^2+(\mathrm{Y}-\mathrm{y}_2)^2+(\mathrm{Z}-\mathrm{z}_2)^2&=[\mathrm{v}.(\mathrm{T}-\mathrm{t}_2)]^2\\(\mathrm{X}-\mathrm{x}_3)^2+(\mathrm{Y}-\mathrm{y}_3)^2+(\mathrm{Z}-\mathrm{z}_3)^2&=[\mathrm{v}.(\mathrm{T}-\mathrm{t}_3)]^2\\(\mathrm{X}-\mathrm{x}_3)^2+(\mathrm{Y}-\mathrm{y}_3)^2+(\mathrm{Z}-\mathrm{z}_3)^2&=[\mathrm{v}.(\mathrm{T}-\mathrm{t}_4)]^2\end{aligned} \qquad (4)$$

2.2 Popis systému GPS

Globální poziční systém byl vyvinut armádou Spojených států amerických a je primárně určen pro vojenské účely. V omezené formě lze tento systém využít i v civilní oblasti, a to především k stanovení vlastní polohy a navigaci v neznámém terénu. Jedná se

o GNSS pracující s kódovým systémem určování polohy. Celý systém se skládá z několika podsystémů, jež vlastní a provozuje armáda Spojených států amerických.

2.2.1 Struktura systému GPS

Systém je dělen na tři základní segmenty

- Kosmický segment
- Kontrolní segment
- Uživatelský segment

Kosmický segment

Do této části GPS systému se řadí satelity umístěné na orbitu Země. Základní konfiguraci tvoří 24 satelitů obíhajících po 6 orbitálních drahách. Ty jsou nakloněné vůči rovníku o 55° a vzájemně posunuty o 60°. Na každém tomto orbitě jsou umístěny 4 satelity. Přibližná výška satelitů nad povrchem Země je 20200 km a jejich oběžná rychlost činí 3,8 km/s. Čas potřebný pro oběh satelitu kolem země je 11 hodin 58 min, což odpovídá jedné polovině hvězdného dne. Satelity jsou vybaveny přesnými atomovými hodinami, anténami pro vysílání navigačních dat, komunikačními obvody pro kalibraci polohy a času, a detektory užívanými pro vojenské účely.

V současné době je na orbit Země umístěno 30 satelitů systému GPS, některé slouží jako záložní, některé jsou součástí vývoje systému a budou postupně nahrazovat stávající satelity.

Kontrolní segment

Tato část systému GPS zajišťuje neustálý dohled nad správnou funkcí a provádí měřicí a kalibrační zákroky na jednotlivých satelitech. Kontrolní segment se skládá z velitelství umístěného na letecké základně v Los Angeles v USA, z řídícího střediska na letecké základně Colorado Springs, z osmnácti monitorovacích středisek a tří povelových stanic. Povelové stanice a monitorovací střediska jsou vždy umístěna společně na jednom místě v co nejtěsnější blízkosti rovníku. Monitorovací stanice trasují jednotlivé satelity a od nich přijaté údaje přenáší do řídícího střediska. Zde jsou tato data zpracovávána a vyhodnocována. Výsledkem jsou korekční zásahy do chodu atomových hodin jednotlivých satelitů, případně korekce polohy na oběžné dráze. Tyto zásahy jsou prováděny s periodou několika hodin.

Uživatelský segment

Základní rozdělení uživatelského segmentu je na vojenskou a civilní skupinu. Vojenská skupina může využít maximálního potenciálu systému a to s přesností na desítky až jednotky centimetrů. Civilní skupina pak přesnost desítek až jednotek metrů.

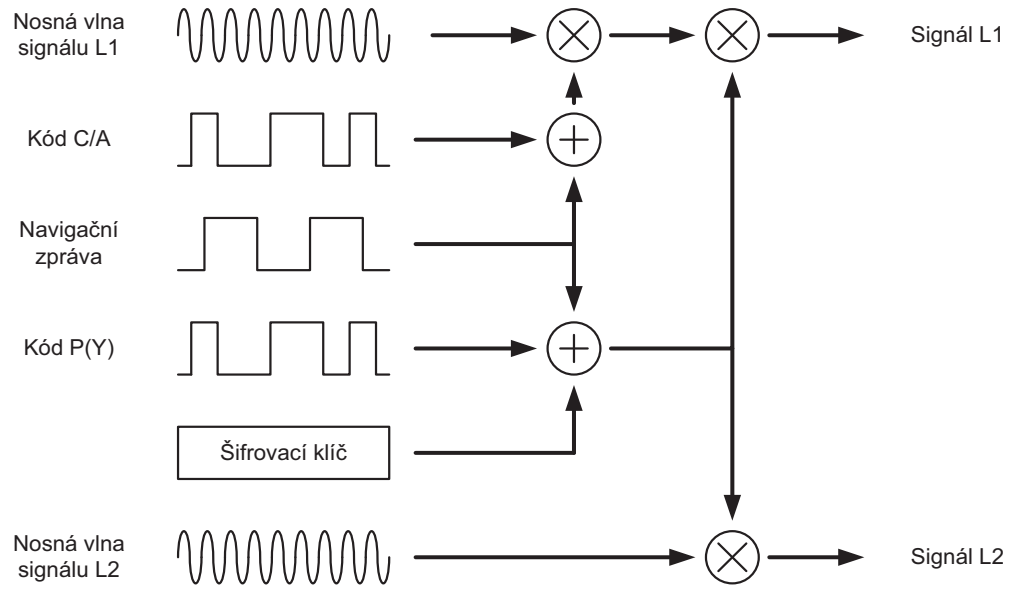
2.2.2 Signály systému GPS

Jak již bylo řečeno, systém GPS je založen na kódovém způsobu určování polohy, kdy je nezbytné znát polohu satelitů a jejich vzdálenost. Informace o vzdálenosti je vypočtena ze vzájemného zpoždění přijatých signálů, poloha je pak zakódována v signálu, který satelit vysílá.

Systému GPS pracuje se dvěma nezávislými signály, civilním a vojenským. Oba tyto signály jsou tvořeny fázovou modulací s binárním klíčováním na základní nosné (označení L1), která má u civilního signálu frekvenci 1575,42 MHz, u vojenského (označenéhoL2) pak 1277,60 MHz. Na tento signál je modulována navigační zpráva. Navigační zpráva obsahuje mimo jiné informace o poloze daného satelitu, předpokládanou dráhu jeho letu a informace o stavu všech satelitů systému GPS. Protože satelity vysílají signály vždy současně na stejné frekvenci, je třeba je vzájemně oddělit. K tomu slouží takzvaný PRN kód (zkratka z anglického Pseudo Random Number – pseudonáhodné číslo), který je pro každou z družic unikátní. Vysílaný signál je tímto kódem upraven tak, že se zdánlivě jeví jako náhodný šum. Po přijetí takto upravených signálů přijímačem a se znalostí jednotlivých PRN kódů je možné signály znovu obnovit a vzájemně oddělit. Systém GSP využívá dva PRN kódy. Civilní kód je označen C/A, vojenský kód P(Y).

C/A kód má délku 1023 bitů, což umožňuje 2^{1023} kombinací. Z tohoto počtu bylo matematicky určeno 37 vzájemně nejméně korelovatelných čísel, tedy je možné je s největší spolehlivostí vzájemně oddělit. C/A kód je vysílán na nosné frekvenci L1.

P(Y) je vojenský PRN kód. Jeho délka je $2{,}35.10^{14}$. Jedná se o šifrovaný kód P, který je možné číst jen za pomoci známého dešifrovacího klíče. Informace o těchto signálech nejsou přístupné, podléhají vojenskému tajemství. P(Y) kód je vysílán na frekvencích L1 i L2. Obr. 2 schematicky znázorňuje tvorbu signálu L1 a L2.



Obr. 2: Schematické znázornění tvorby signálů L1 a L2

2.3 Protokol NMEA 0183

Výstupní data z GPS přijímačů jsou obecně dostupná ve standardizovaných protokolech. Nejrozšířenějším a nejpoužívanějším je standart NMEA 0183. Protokol NMEA 0183 je chráněný licencí a oficiální data k němu nejsou volně dostupná. Lze je však u organizace NMEA zakoupit. Veškeré další údaje o tomto protokolu jsou tedy převzaty z katalogového listu GPS přijímače použitého v této práci [2] nebo odvozeny ze samotného testování zařízení.

Data vysílaná v protokole NMEA – 0183 jsou členěna do jednotlivých vět, které se dělí na vysílací, dotazovací a nastavovací. Tyto věty se skládají z řetězce znaků, přičemž každý znak je reprezentován hexadecimálním číslem odpovídajícím danému znaku v ASCII tabulce [3]. Každá věta je vždy uvozena znakem „*$*“ ($24_{16}$), po němž následuje dvojice znaků (*ii*) identifikujících zařízení, které větu vysílá. Seznam nejčastěji používaných identifikačních dvojic znaků je uveden v tab. 1.

Tab. 1: Druhy nejčastěji používaných vět protokolu NMEA 0183

Zkratka	Typ vysílajícího zařízení
DF	zaměřovač
AG	autopilot
HC	kompas magnetický
HE	kompas gyroskopický
RA	radar
GP	globální poziční systém
VD	senzor rychlosti, systém DOPPLER
SD	akustický hloubkoměr

Následující trojice znaků (*sss*) určuje druh vysílané věty. Věta je vždy ukončena dvojicí znaků <*CR*> ($0D_{16}$) a <*LF*> ($0A_{16}$), které jsou v ASCII tabulce označovány jako návrat na začátek řádku (*carriage return*) a posuv na nový řádek (*line feed*). Jednotlivé údaje obsažené ve větě (*d0, d1*) jsou odděleny znakem „*,*“ ($2C_{16}$). Není li zařízení schopno konkrétní údaj v daném okamžiku určit, je daná pozice v řetězci prázdná. Z tohoto faktu vyplývá, že jediným způsobem jako se v NMEA větě orientovat je sledování znaků „,“ ($2C_{16}$) a dekódování údajů mezi nimi. Pro desetinný oddělovač je použit znak „.“ ($2E_{16}$). Ke kontrole přenosu slouží dvoumístné číslo (*cc*) na konci věty, které je uvozeno znakem „*“ ($2A_{16}$). To je vypočteno exkluzivním součtem všech znaků ve větě mezi znaky „*$*“ a „*“. Maximální délka jedné věty je 80 znaků. Obecný příklad vysílací věty:

$$\$iisss,d0,d1,d2,d3*cc<CR><LF>.$$

Protokol NMEA 0183 stanovuje 78 různých vět, které jsou používány pro konkrétní typy zařízení, ne všechny však souvisí pouze s údaji o navigaci. Jelikož popis všech typů vět by byl velice obsáhlý, jsou v tab. 2 popsány pouze věty používané standardními GPS přijímači. V praktické části této práce je odkazováno na věty typu RMC a GGA, proto je jejich obsah podrobně definován v tab. 3 a tab. 4. Popis zbylých vět je součástí přílohy této práce.

Tab. 2: Druhy vět užívané GPS modulem

Zkratka	Anglické označení	Význam
RMC	Recommended Minimum data	Minimum navigačních dat
VTG	Course over ground and Ground speed	Kurz a rychlost pohybu
GGA	Global positioning system fix data	Navigační data pro stanovený čas
GSA	Dilution Of Precision and Active Satellites	Aktivní satelity a přesnost určení pozice
GSV	Satellites in view	Viditelné satelity
GLL	Geographic Position – Latitude/Longitude	Navigační data, zeměpisná šířka a délka

Věta RMC – minimum navigačních dat

Obecný tvar věty RMC je.

$$\$GPRMC,hhmmss,A,llll.lllll,N,yyyyy.yyyyy,E,s.sss,uu.uu,ddmmrr,md,vmd,*cc<CR><LF>$$

Tab. 3: Popis obsahu věty RMC

Formát	Příklad	Jednotka	Popis
$GPRMC	$GPRMC	-	Uvození věty RMC, vysílač se hlásí jako GP.
hhmmss	122304	h, min, s	Čas určení dat ve větě. Je použit UTC (Coordinated Universal Time - koordinovaný světový čas).
a	A	-	Status určení dat. (A – data v pořádku, V – varování)
llll.lllll	4857.32344	°, ´	Zeměpisná šířka. (48°57,32344')
n	N	-	Ukazatel zeměpisné šířky. (N – severní, S – jižní)
yyyyy.yyyyy	01633.54354	°, ´	Zeměpisná délka. (16°33,54354')
e	E	-	Ukazatel zeměpisné délky. (W – západní, E – východní)
s.sss	3.567	uzel	Rychlost vodorovného pohybu.
uu.uu	17.33	°	Kurz vodorovného pohybu.
ddmmrr	230309	-	Datum určení dat. (23.3.2009)
md	2.4	°	Magnetická deklinace. (Vzájemný rozdíl mezi směry magnetického a zeměpisného pólu).
vmd	E	-	Směr magnetické deklinace. (W – západ, E – východ)
*cc	*25	-	Kontrolní součet.
<CR><LF>	<CR><LF>	-	Konec věty.

Věta GGA – Navigační data pro stanovený čas

Obecný tvar věty GGA:

$$\$GPGGA,hhmmss,llll.lllll,N,yyyyy.yyyyy,E,x,ss,HDOP,nnn.n,m,gg.g,m,dd,i,*cc<CR><LF>$$

Tab. 4: Popis obsahu věty GGA

Formát	Příklad	Jednotka	Popis
$GPGGA	$GPGGA	-	Uvození věty GGA, vysílač se hlásí jako GP.
hhmmss	122304	h, min, s	Čas určení dat ve větě. Je použit UTC (Coordinated Universal Time - koordinovaný světový čas).
llll.lllll	4857.32344	°, ´	Zeměpisná šířka. (48°57,32344')
n	N	-	Ukazatel zeměpisné šířky. (N – severní, S – jižní)
yyyyy.yyyyy	01633.54354	°, ´	Zeměpisná délka. (16°33,54354')
e	E	-	Ukazatel zeměpisné délky. (W – západní, E – východní)
x	1	-	Způsob určení polohy. (0 – neurčeno, 1 – určeno GPS satelity, 2 – korekce DGPS – pozemní referenční stanice)
ss	10	-	Počet viditelných satelitů.
HDOP	2.03	m	Rozptyl přesnosti určení pozice (horizontální) vzhledem k aktuálnímu rozestavení satelitů užitých pro výpočet pozice.
nnn.n	328.6	m	Nadmořská výška.
m	M	-	Jednotka nadmořské výšky.
gg.g	23.5	m	Rozdíl mezi určenou nadmořskou výškou (geoid) a výškou nad elipsoidem (WGS-84).
m	M	.	Jednotka předešlého údaje.
dd	23	s	Stáří korekčních údajů poskytnutých pozemní referenční stanicí.
i	8	-	Identifikační číslo pozemní referenční stanice.
*cc	*63	-	Kontrolní součet.
<CR><LF>	<CR><LF>	-	Konec věty.

2.4 Mikrokontrolér Xilinx PicoBlaze

Jedná se o 8bitový mikrokontrolér, který je možné implementovat do všech obvodů FPGA a CPLD společnosti Xilinx. Existuje několik variant tohoto mikrokontroléru vyhovujících konkrétním řadám obvodů FPGA případně CPLD. Jejich rozdělení a základní vlastnosti jsou uvedeny v tab. 5 [4]. KCPSM je zkratka staršího označení mikrokontroléru PicoBlaze, z anglického „Constant(K) Coded Programmable State Machine“ (původně

„Ken Chapman's Programmable State Machine“) neboli „Programovatelný stavový automat Kena Chapmana“. Ken Chapmen je tvůrcem mikrokontroléru PicoBlaze.

Tab. 5: Rozdělení a základní parametry mikrokontroléru PicoBlaze.

	Verze mikrokontroléru		
Parametr	**KCPSM**	**KCPSM2**	**KCPSM3**
Řady cílových obvodů	Spartan II, Spartan II E, Virtex, Virtex E	Virtex II, Virtex II PRO	Spartan 3, Virtex II, Virtex II PRO
Maximální velikost programu	256 instrukcí	1024 instrukcí	1024 instrukcí
Počet registrů	16	32	16
Počet použitých bloků v cílovém FPGA obvodu	76	84	96
Maximum	16	32	32

Největší výhodou obvodu PicoBlaze je jeho univerzálnost a velikost. V obvodech FPGA zabírá pouze nepatrné procento z celkového využitelného prostoru pro VHDL design. Jeho výkon je pak limitován pouze maximální frekvencí hodinového signálu, přípustnou pro zvolený typ obvodů FPGA. Další předností je fakt, že do obvodu FPGA lze implementovat libovolné množství nezávisle pracujících mikrokontrolérů, omezení zde představuje pouze volný prostor pro VHDL design zvoleného FPGA obvodu. Je tedy možné současně implementovat několik kontrolérů, které ovládají např. jednotlivé periférie, jako jsou LCD, klávesnice nebo UART, a tak výrazně zjednodušit celkový návrh softwaru.

Základní vlastnosti procesoru PicoBlaze:

- 16 8bitových registrů,
- program o délce maximálně 1024 instrukcí,
- 8bitová aritmeticko-logická jednotka,
- 64B paměť dat typu RAM,
- 256 vstupních a 256 výstupních 8bitových portů,
- čas zpracování instrukce - 2 hodinové cykly,

- užití jazyka ASSEMBLER.

Registry

PicoBlaze umožňuje využít 16 registrů (*sX*), označených *s0* až *sF*. Ty jsou využívány při operacích aritmeticko-logické jednotky a pro adresování vstupních a výstupních portů.

Paměť programu

Maximální délka programu je limitována počtem 1024 instrukcí, přičemž každá instrukce má šířku 18 bitů. Paměť programu má tedy velikost 1024x18 bitů.

Aritmeticko logická jednotka

Aritmeticko logické operace se provádí prostřednictvím speciálních registrů *sX* (*s0* až *sF*). ALU umožňuje provádět standardní operace součtu, rozdílu, logické operace AND, OR, XOR, posun a rotaci registrů.

Příznaky

Jsou použity příznaky *ZERO*, *CARRY* a *INTERRUPT_ENABLE*. Příznak *ZERO* je nastaven v případě, že výsledek předešlé operace je roven nule. Příznak *CARRY* je univerzální a závisí na typu předchozí instrukce. *INTERRUPT_ENABLE* je příznakem povolení vnějšího přerušení.

Paměť dat

Je uživatelsky přístupná 8bitová paměť o velikosti 64B s možností přímé i nepřímé adresace. Čtení se provádí instrukcí *FETCH*, zápis instrukcí *STORE*.

Vstupní a výstupní port

Čtení portů se provádí prostřednictvím instrukce *INPUT* uložením dat z portu *IN_PORT* do registru *sX* (*s0* až *sF*). Zápis instrukcí *OUTPUT* přenesením dat z registru *sX* (*s0* až *sF*) na port *OUT_PORT*. Porty jsou 8bitové a rozšíření jejich počtu umožňuje 8bitový výstup *PORT_ID*, který udává adresu vstupních a výstupních portů během instrukcí *INPUT* a *OUTPUT*. Užitím vnější logiky a klopných obvodů je umožněno pracovat až s 256 vstupními a 256 výstupními porty. Pro synchronizaci rozšířených portů obsahuje procesor výstupy *READ_STROBE* a *WRITE_STROBE*. Na těchto výstupech je po dobu provádění instrukce *INPORT* (čtení) respektive *OUTPORT* (zápis) nastavena úroveň log1. Rozšíření vstupních a výstupních portů je popsáno v kapitole 2.4.3.

Čítač instrukcí

Velikost čítače instrukcí je 1024. Při provedení instrukce je automaticky inkrementován a po jeho přetečení se nuluje a začíná čítat znovu. Přímá změna stavu čítače instrukcí není možná. Děje se tak pouze provedením podmíněného a nepodmíněného skoku, voláním podprogramů, přerušením a resetem.

Skoky (JUMP)

Jsou rozděleny na podmíněné a nepodmíněné. Vždy se jedná se o absolutní adresaci bez uložení návratové adresy. Délka skoku v programu není omezena.

Podprogramy

Jsou volány instrukcí *CALL*. Při tomto volání je uložena návratová adresa do *CALL/RETURN* zásobníku. Při následném zpracování instrukce *RETURN* je návratová adresa ze zásobníku *CALL/RETURN* vyzvednuta a zapsána do čítače instrukcí, čímž je proveden skok v programu na místo volání podprogramu. Tento zásobník umožňuje maximálně 32 současně uložených návratových adres, tedy 32násobné vnoření podprogramu.

Přerušení

Je dostupný jeden vstup pro přerušení, který je synchronizován s hodinovým signálem procesoru. Příznak přerušení vyvolá skok na adresu programu ($3FF_{16}$). Zároveň je uložena návratová adresa do *CALL/RETURN* zásobníku a po volání instrukce *RENTURNI* je proveden návrat na původní adresu obdobně jako u podprogramů. Příznak přerušení je nastaven po pěti hodinových cyklech od okamžiku události na vstupu *INTERRUPT* (log1). K hardwarovému mazání vnějšího přerušení slouží výstup *INTERRUPT_ACK*, který se během zpracovávání přerušení nastavuje do úrovně log1. Blíže je popsáno v kapitole 2.4.4.

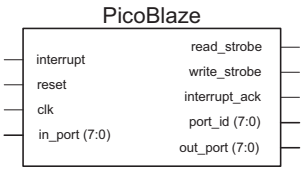
Reset

Je prováděn automaticky po konfiguraci odvodu FPGA nebo přivedením signálu log1 na vstupu RESET. Událost reset provede nulování příznaků *ZERO*, *CARRY*, *ITERUPT_ENABLE*, ukazatele *PC* a zásobníku *CALL/RETURN*. Paměť programu, paměť dat a registry reset neovlivní.

Časování

Procesor pracuje s jedním hodinovým signálem přiváděným na vstup *CLK*. Všechny vnitřní obvody pracují při nástupné hraně tohoto signálu. Jeho frekvence je omezena pouze použitým obvodem FPGA.

2.4.1 Popis komponenty mikrokontroléru PICOBLAZE



Obr. 3: Grafické znázornění komponenty PicoBlaze

Tato komponenta je po registraci volně dostupná na www stránkách společnosti Xilinx [5]. Funkce vstupních a výstupních portů jsou definovány v tab. 6.

Tab. 6: Popis vstupů a výstupů mikrokontroléru PicoBalze.

Port	Směr	Funkce
IN_PORT[7:0]	vstupní	Vstupní 8bitový port
OUT_PORT[7:0]	výstupní	Výstupní 8bitový port
PORT_ID[7:0]	výstupní	Adresový port vstupního a výstupního portu
INTERRUPT	vstupní	Přerušení
RESET	vstupní	Reset
CLK	vstupní	Globální hodinový signál
READ_STROBE	výstupní	Pomocný výstup čtecí instrukce INPORT
WRITE_STROBE	výstupní	Pomocný výstup zápisové instrukce OUTPORT
INTERRUPT_ACK	výstupní	Pomocný výstup přerušení

2.4.2 Assembler

Algoritmus, který má mikroprocesor vykonávat, musí být popsán v programovacím jazyce Assembler. Instrukční sada procesoru PicoBlaze je v tab. 7. Přesná specifikace jednotlivých instrukcí je dostupná v katalogovém listu procesoru. [4]

Tab. 7: Výpis instrukční sady

Instrukce	Popis
JUMP aaa; JUMP Z, aaa; JUMP NZ, aaa; JUMP C,aaa; JUMP NC, aaa	Skokové instrukce (podmíněné, nepodmíněné) na adresu „aaa“, Z – příznak ZERO, C – příznak CARRY.
CALL aaa; CALL Z, aaa; CALL NZ, aaa; CALL C, aaa; CALL NC, aaa	Instrukce volání podprogramů „aaa“ (podmíněné, nepodmíněné), Z – příznak ZERO, C – příznak CARRY.
RETURN; RETURN Z; RETURN NZ RETURN N; RETURN NC;	Návratové instrukce z podprogramů (podmíněné, nepodmíněné), Z – příznak ZERO, C – příznak CARRY.
ADD sX, kk; ADDCY sX, kk; SUB sX, kk; SUBCY sX, kk; COMPARE sX, kk,	Aritmetické instrukce součtu, rozdílu a komparace. sX, – registr, kk – konstanta
LOAD sX, kk; AND sX, kk; OR sX, kk; XOR sX, kk; TEST sX, kk;	Logické instrukce
STORE sX, ss; FETCH sX, ss	Čtení a zápis do paměti dat, ss – adresa v paměti dat
SR0 sX; SR1 sX; SRX sX; SRA sX; RR sX; SL0 sX; SL1 sX; SLX sX; SLA sX; RL sX;	Instrukce rotace posunutí vlevo a vpravo
INPUT sX, pp; OUTPUT sX, pp	Čtení a zápis na vnější porty, pp – adresa portu

U všech instrukcí obsahujících konstanty (*kk, ss, pp*) je možné jejich nahrazení registrem *sY*.

2.4.3 Rozšíření vstupních a výstupních portů

Jak již bylo uvedeno, umožňuje procesor PICOBLAZE pracovat s až 256 vstupními a 256 výstupními 8bitovými porty. K jejich získání je nutné rozšíření entity o logické členy vyhodnocující adresaci vstupních a výstupních portů.

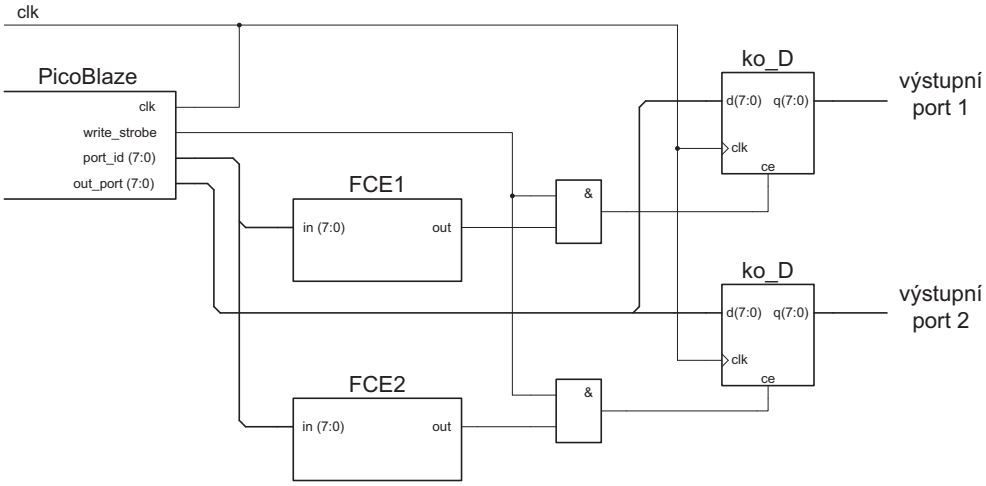
Výstupní port

Obr. 4 znázorňuje rozšíření výstupního portu na 2 nezávislé 8bitové porty. Celý design je synchronizován s hodinovým signálem CLK. Jsou zde použity dva 8bitové klopné obvody D s možností povolení jejich funkce prostřednictvím *CE* (aktivní na úroveň log. 1). Aktivní úroveň je na vstup *CE* přivedena pouze v případě, že výstupem bloku *FCE* daného portu je log. 1 a zároveň je prováděna instrukce *OUTPUT* procesoru. Tato instrukce zajišťuje po dobu své činnosti log1 na výstupu *WRITE_STROBE*. Pokud bude tedy adresa výstupního portu 1 rovna hodnotě 01_{16} musí být výstup bloku *FCE1* tvořen logickým výrazem:

$$out = \overline{in(7)}\ AND\ \overline{in(6)}\ AND\ \overline{in(5)}\ AND\ \overline{in(4)}\ AND\ \overline{in(3)}\ AND\ \overline{in(2)}\ AND\ \overline{in(1)}\ AND\ in(0)$$

Obdobně pro výstupní port 2, jehož adresa má mít hodnotu 02_{16} je výstup *FCE2* tvořen výrazem:

$$out = \overline{in(7)}\ AND\ \overline{in(6)}\ AND\ \overline{in(5)}\ AND\ \overline{in(4)}\ AND\ \overline{in(3)}\ AND\ \overline{in(2)}\ AND\ in(1)\ AND\ \overline{in(0)}$$



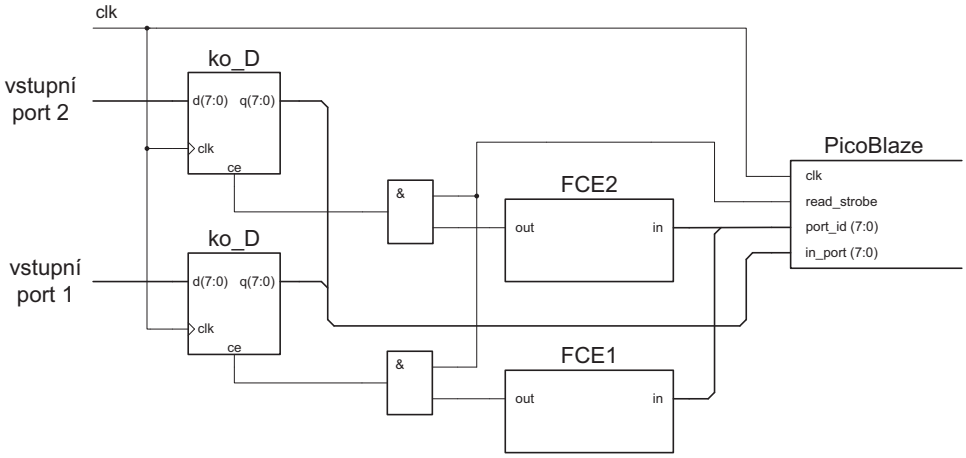
Obr. 4: Rozšíření výstupního portu procesoru

Toto rozšíření portu je možné popsat v jazyce VHDL sekvenčními příkazy tak, jak ukazuje následující výpis procesu *OUTPUT*.

```
output: process (clk)
        begin
            if rising_edge (clk) then
                if write_strobe = '1' then
                    if port_id = x"01" then
                        výstupní_port_1 <= out_port;
                    end if;
                    if port_id = x"02" then
                        výstupní_port_2 <= out_port;
                    end if;
                end if;
            end if;
        end process output;
```

Vstupní port

Obdobná situace jako u výstupního portu je i u vstupního portu. Rozdíl je pouze v užitých portech procesoru. Pro čtení slouží vstupní port *IN_PORT* a pomocným výstupem *READ_STROBE*. Shodná jsou i vnitřní uspořádání bloků *FCE1* a *FCE2* při adresaci vstupního portu 1 hodnotou 01_{16} a vstupního portu 2 hodnotou 02_{16}. Zapojení znázorňuje obr. 5.



Obr. 5: Rozšíření vstupního portu procesoru

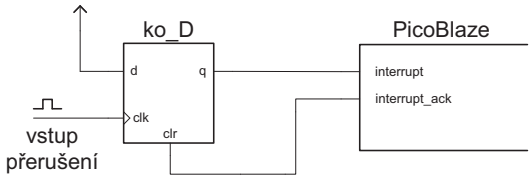
VHDL zápis rozšíření vstupního portu v popsaném uspořádání pomocí procesu *INPUT*.

```
input: process (clk)
        begin
            if rising_edge (clk) then
                if read_strobe = '1' then
                    if port_id = x"01" then
                        in_port <= vstupni_port_1;
                    end if;
                    if port_id = x"02" then
                        in_port <= vstupni_port_2;
                    end if;
                end if;
            end if;
        end process input;
```

2.4.4 Ošetření vstupu vnějšího přerušení

K hardwarovému přerušení běhu programu slouží vstup *INTERRUPT*. Ten je doplněn o výstup *INTERRUPT_ACK*. Přerušení je vyvoláno hodnotou log.1 na vstupu *INTERRUPT* po dobu alespoň 2 period hodinového signálu. Při skoku programu do obslužné procedury

přerušení procesor nastavuje výstup *INTERRUPT_ACK* do úrovně log. 1 a to opět po dobu 2 period hodinového signálu. Toho lze využit k hardwarovému mazání příznaku přerušení po jeho obsloužení a odstranit tak nežádoucí vícenásobné vyvolání přerušení. To by mohlo nastat v případě, že logická úroveň 1 setrvá na vstupu *INTERRUPT* po dobu, delší než jsou 2 periody hodinového signálu.



Obr. 6: Hardwarové ošetření vstupu přerušení.

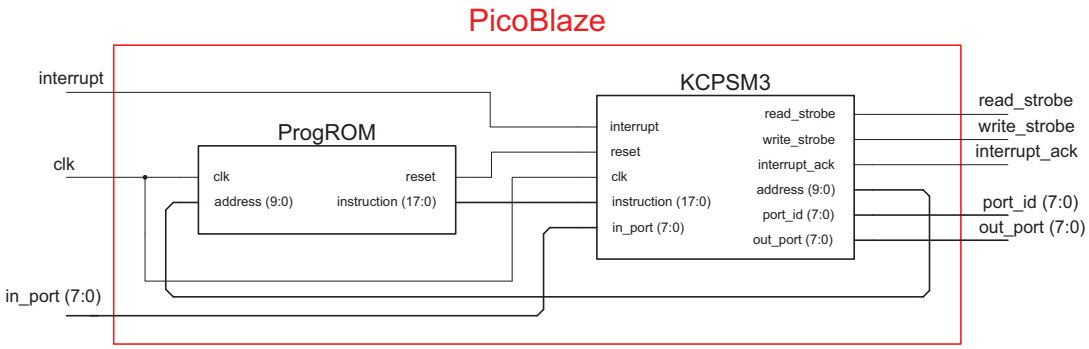
Řešení spočívá ve využití klopného obvodu D s možností nulování výstupu tak, jak je patrno z obr. 6. Vnější signál generující přerušení je přiveden na hodinový vstup tohoto klopného obvodu. Vstup *D* je pak trvale nastaven do úrovně log. 1. Pokud tedy nastane událost vyvolávající přepsání klopného obvodu D, je zároveň vyvoláno přerušení procesoru. Prostřednictvím signálu *INTERRUPT_ACK* je pak výstup klopného obvodu nulován, což zajišťuje vyvolání pouze jednoho přerušení pro jednu událost na vstupu přerušení. Z toho je zřejmé, že volbou klopného obvodu je možné dosáhnout vyvolání přerušení nástupnou nebo sestupnou hranou. Obr. 6 zobrazuje řešení pro volání přerušení nástupnou hranou. Popis v jazyce VHDL je následující.

```
interrupt:  process (vstup_preruseni, interrupt_ack)
            begin
                if rising_edge (vstup_preruseni) and interrupt_ack = '0' then
                    interrupt <= '1';
                end if;
                if interrupt_ack = '1' then
                    interrupt <= '0';
                end if;
            end process input;
```

2.4.5 Generování komponenty pamětí programu

Program pro procesor je fyzicky tvořen komponentou *ProgROM*, což je paměť o velikosti 1024x18b. Ta je spojena s komponentou *KCPSM3* pomocí dvou sběrnic

INSTRUCTION a *ADDRESS* a dvou signálů *RESET* a *CLK*. Toto zapojení je znázorněno na obr. 7.

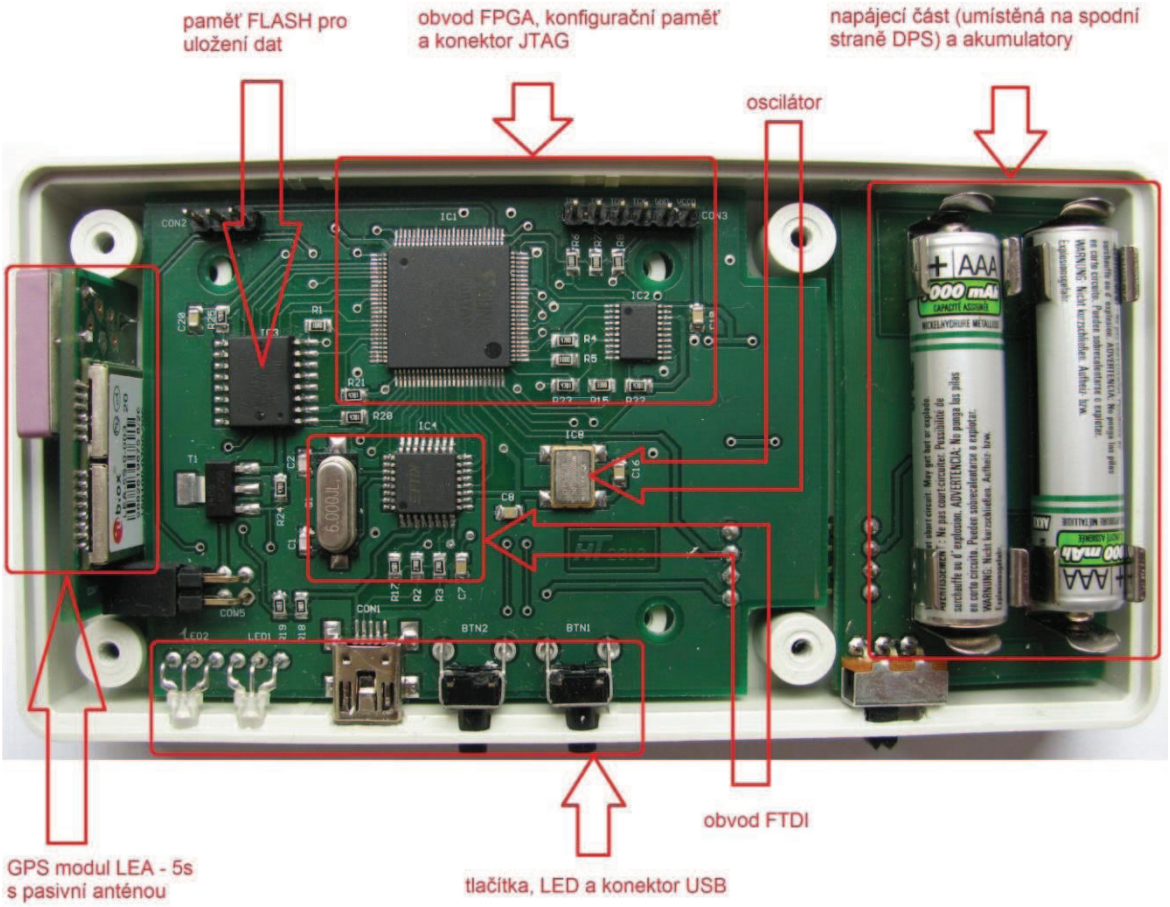


Obr. 7: Spojení jádra procesoru s pamětí programu.

Pro tvorbu programu procesoru PICOBLAZE bylo vytvořeno několik vývojových prostředí. Jedním z nich je i volně poskytovaný program pBlazIDE společnosti Mediatronix [6]. Ten umožňuje nejen tvorbu, ale i simulaci vytvořeného kódu. Pro tvorbu programu byl však použit základní postup doporučovaný společnosti Xilinx. Samotný kód lze napsat v libovolném textovém editoru, je však třeba dodržovat syntaxi jazyka assembler dle katalogového listu [4]. Výsledný soubor je nutné uložit s koncovkou **.psm*. Tento soubor dále slouží jako výchozí zdroj dat pro vygenerování souboru *ProgROM.vhd* obsahující komponentu *ProgROM*. Jejím spojením s komponentou *KCPSM3* vzniká funkční entita procesoru *PICOBLAZE*, obr. 7. Pro samotné vygenerováni souboru *ProgROM.vhd* poskytuje společnost Xilinx aplikaci *KCPSM3.exe* [5]. Při spuštění tohoto programu z adresáře, kde je uložen i soubor **.psm* je automaticky vygenerován soubor obsahující komponentu s pamětí programu.

3 Hardwarové zpracování zařízení

Cílem této práce je návrh a konstrukce zařízení, které je schopno za pomoci systému GPS zaznamenávat vlastní polohu, rychlost pohybu a nadmořskou výšku. Tyto data následně ukládá do vlastní paměti pro pozdější zpracování na PC. Přenos uložených dat je realizován pomocí rozhraní USB. Celé zařízení je řízeno obvodem FPGA a je mobilní, tedy napájeno z baterie. Na obr. 8 je znázorněno celé zařízení s označením jednotlivých funkčních bloků. Zařízení je složeno z tří desek plošných spojů propojených konektory. Tento způsob konstrukce byl zvolen pro lepší polohu antény GPS nacházející se na desce společně s GPS modulem LEA – 5s. Zároveň byla tímto oddělena napájecí část, u které je předpokládán další vývoj, tudíž nebylo vhodné jej osazovat na jednu desku společně s řídící částí obvodu.



Obr. 8: Fotografie celého zařízení osazeného v krabičce s vyznačením jednotlivých funkčních bloků

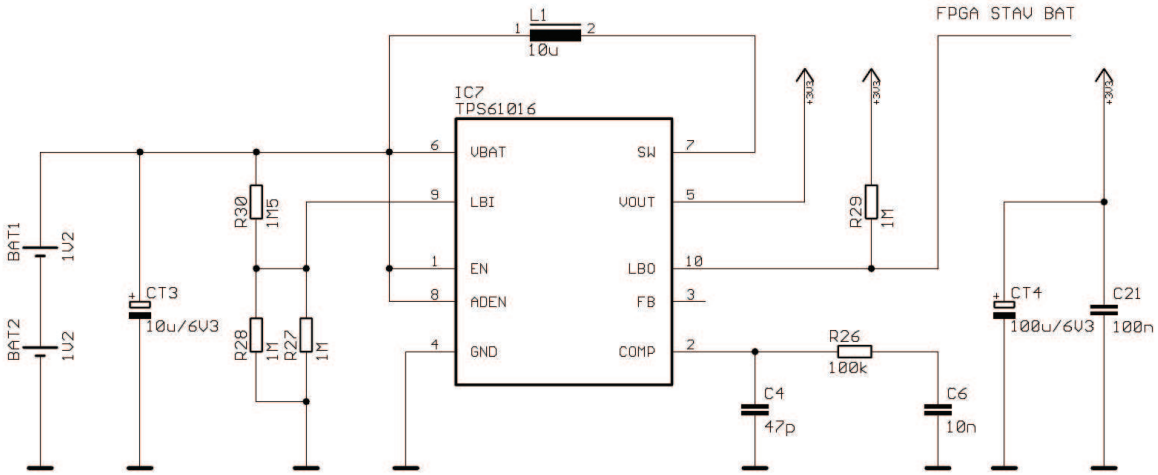
3.1 Napájení zařízení

Zařízení je konstruováno jako mobilní, tedy napájené z baterie. Proto bylo při řešení napájení nutné zajistit nízký proudový odběr celého zařízení vhodnou volbou použitých obvodů. Všechny obvody jsou napájeny napětím +3,3 V. Obvod FPGA pak vyžaduje napětí +3,3 V, +2,5 V a +1,2 V.

Jako primární zdroj napětí byla zvolena sériová kombinace dvou akumulátorů Ni-MH jmenovitého napětí +1,2 V v pouzdře rozměrů AAA. Výsledné napětí +2,4 V je za pomocí měniče zvyšováno na požadovanou hodnotu +3,3 V. Další hodnoty napětí vyžadované obvodem FPGA jsou tvořeny lineárními regulátory.

3.1.1 Napěťový měnič TPS61016

Pro vytvoření stabilizovaného napětí +3,3 V byl zvolen obvod TPS61016 společnosti Texas Instrument [8], který slouží jako napěťová pumpa s napájecím napětím 1,2 V až 3 V a maximálním proudovým odběrem 250 mA pří vstupním napětí větším jak +1,8 V. Bylo zvoleno výrobcem doporučené zapojení, jehož schéma je zobrazeno na obr. 9. Rezistorový dělič *R27, R28 a R30* na vstupní svorce obvodu *LBI* definuje úroveň napětí kladného vstupu vnitřního komparátoru, který ovládá výstupní svorku *LBO*. Na záporné svorce komparátoru je připojeno vnitřní referenční napětí +0,5 V. Pokud tedy poklesne vstupní napětí pod hodnotu +2 V, poklesne přes dělící poměr 1:4 na vstupní svorce *LBI* napětí pod hodnotu +0,5 V. To způsobí překlopení vnitřního komparátoru a na jeho výstupu je úroveň log0. Tento výstup je spojen se svorkou *LBO*. Rezistor *R29* pak definuje úroveň log1 výstupu *LBO* v případě, že vstupní napětí je dostatečné, tedy komparátor je překlopen do log1. Tato signalizace poklesu vstupního napětí je dále vyhodnocována řídícím obvodem FPGA. Cívka *L1* a kompenzační CRC můstek *C4 R26 C6* jsou zapojeny dle katalogového listu obvodu [8], stejně jako vstupní a výstupní kondenzátory *CT3*, *CT4* a *C21*.



Obr. 9: Schéma zapojení napěťového měniče TPS61016

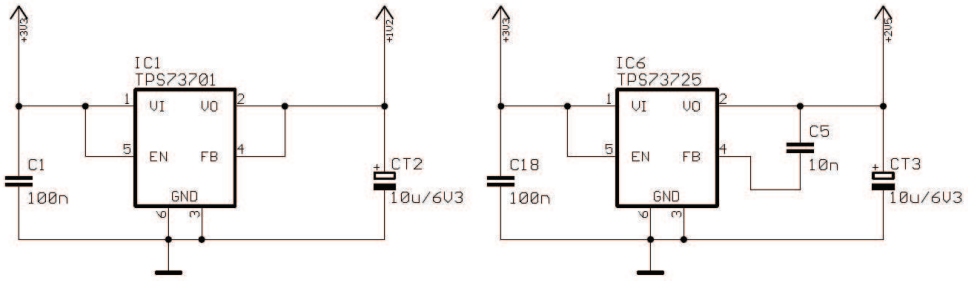
3.1.2 Lineární regulátory napětí TPS73725 a TPS73701

Tyto regulátory napětí jsou využity pro napájení obvodu FPGA. Ten kromě napětí +3,3 V pro výstupní posilovače portů vyžaduje napětí +2,5 V a +1,2 V. Napětí +2,5 V je označováno jako *Vccaux* a slouží pro napájení vnitřních DCM obvodů (digitální hodinový manažer) a pro konfigurační rozhraní JTAG. Proudový odběr tohoto napětí jsou jednotky mA. Proto byl pro vytvoření tohoto napětí zvolen lineární regulátor TPS73725 společnosti Texas Instruments [9]. Ten je schopen při vstupním napětí +3,3 V vytvářet stabilizované napětí +2,5 V s proudovým odběrem do 100 mA.

Napájecí napětí +1,2 V, označované jako *Vccint* je odbodem FPGA využíváno pro napájení vnitřního jádra, logických členů, paměťové části obvodu a k napájení multiplexerů. Proudový odběr této napájecí větve obvodu FPGA jsou jednotky mA. Napětí *Vccint* je tvořeno lineárním regulátorem TPS73701 [9].

Oba lineární regulátory jsou napájeny obvodem TPS61016. Schéma zapojení je zobrazeno na obr. 10. Hodnoty blokovacích a filtračních kondenzátorů *C17*, *C18*, *CT1* a *CT2* byly navrženy dle katalogového listu obvodu [9]. Oba regulátory umožňují připojením vnějšího rezistorového děliče na svorku *FB* měnit libovolně výstupní napětí. Tato funkce však nebyla využita, proto jsou dle doporučení připojeny na výstupní svorky *VO*. Obvod TPS73725 obsahuje pro nastavení výstupního napětí +2,5 V vlastní interní rezistorový dělič, proto je svorka *FB* připojena na výstupní svorku *VO* Přes kondenzátor *C5*, tak aby byl dělicí poměr zachován. Tento kondenzátor slouží pouze pro stabilizaci obvodu. Oba regulátory obsahuje vstupní svorky *EN*, kterými je možné uvést obvod do režimu spánku. Režim spánku je aktivován spojením vstupu *EN* s nulovým potenciálem.

Tato funkce však také není využívána, proto jsou obě svorky připojeny na vstupní napájecí napětí.



Obr. 10: Schéma zapojení lineárních napěťových regulátorů TPS73701 a TPS73725

3.2 Obvod FPGA SPARTAN – 3A

Jako řídící obvod celého zařízení byl zvolen programovatelný logický obvod (FPGA) s označením XC3S50A od společnosti Xilinx [10]. Jedná se o nejmenší typ obvodu FPGA řady Spartan 3 v pouzdře VQFP100. Tento obvod obsahuje 50 000 systémových hradel, tedy 1584 ekvivalentních logických buněk, 3 vnitřní násobičky, 2 digitální hodinové manažery (DCM) a 68 uživatelsky dostupných vstupně výstupních pinů. Popisovaný obvod zároveň plně podporuje implementaci procesoru PicoBlaze.

3.2.1 Konfigurace obvodu FPGA a JTAG rozhraní

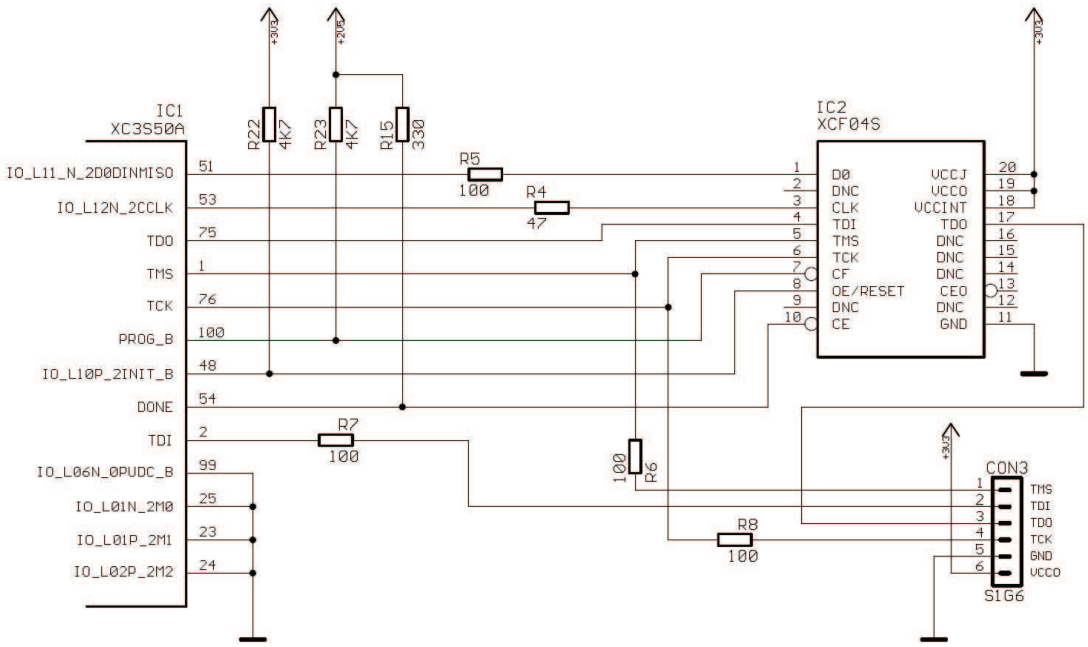
Obvody FPGA obecně provádějí svojí vnitřní konfiguraci při každém připojení napájecího napětí, tedy jejich vnitřní uspořádání je po každém odpojení napájecího napětí ztraceno. Z tohoto důvodu je třeba, aby vytvořený kód popisující konfiguraci obvodu, byl trvale uložen v kompatibilní paměti, ze které bude obvod FPGA tento popis získávat při každém připojení napětí a dle něj se opětovně konfigurovat.

Obvody FPGA řady SPARTAN – 3A umožňují několik odlišných způsobů jak řešit zálohu a vnitřní konfiguraci pomocí externí paměti. Tyto způsoby se liší druhem použité paměti, i způsobem řízení samotné konfigurace. Volba druhu konfigurace se volí pomocí tří vstupních pinů M0, M1 a M2. Kombinace logických úrovní na těchto speciálních vstupech informují obvod o způsobu, jakým bude po připojení napájecího napětí konfigurován. Tab. 8 obsahuje popis konfiguračních způsobů pro danou logickou kombinaci [11].

Tab. 8: Způsoby konfigurace obvodu FPGA

Logická kombinace M2, M1, M0	Příslušný způsob konfigurace obvodu FPGA
<0,0,0>	Sériový mód řízený obvodem FPGA (Master Serial Mode)
<0,0,1>	SPI mód řízený obvodem FPGA (Master SPI Mode)
<0,1,0>	Paralelní mód řízený obvodem FPGA (Master BPI Mode)
<1,1,0>	Paralelní mód řízený externím obvodem (Slave Paralel Mode)
<1,1,1>	Sériový mód řízený externím obvodem (Slave Serial Mode)
<1,0,1>	JTAG konfigurace

Pro tuto práci byl zvolen způsob sériové konfigurace řízené obvodem FPGA. V tomto případě je nutné použít konfigurační paměť společnosti Xilinx z řady XCFxxS, kde *xx* značí velikost paměti. Pro snadnou dostupnost a dostatečnou kapacitu byla vybrána paměť XCF04S [12]. Paměť sama je při zvoleném módu řízena obvodem FPGA, a po sériovém rozhraní odesílá datový tok, který obsahuje uložené konfigurační informace. Samotné programování konfigurační paměti je prováděno pomocí rozhraní JTAG. Zapojení obvodu FPGA, konfigurační paměti XCF04S a rozhraní JTAG je zobrazeno na obr. 11.



Obr. 11: Schéma zapojení konfigurační části obvodu FPGA

Zapojení je realizováno dle doporučení pro napájecí napětí JTAG rozhraní +2,5 V, získaného z katalogového listu popisujícího konfigurační zapojení obvodu FPGA řady SPARTAN – 3A [11].

3.2.2 Zdroj hodinového signálu

Obvody FPGA z pravidla neobsahují vnitřní zdroje hodinového signálu, jako je tomu například u mikrokontrolérů. Použitý obvod SPARTAN 3A není výjimkou. Jako primární zdroj hodinového signálu byl proto v této práci použit oscilátor CFPS – 73 s frekvencí 50 MHz [13]. Jedná se o třístavový HCMOS oscilátor napájený napětím +3,3 V, s maximální zatěžovací kapacitou 15 pF. Stabilita generovaného hodinového signálu je ± 50 ppm a čas nástupné a sestupné hrany je 6 ns. Výstup oscilátoru byl připojen na vstup globálního hodinového signálu FPGA obvodu, označeného *GLCK*. Takto připojený zdroj lze následně využít v celém konfigurovaném FPGA obvodu pro řízení synchronního designu.

3.3 Externí paměť pro záznam GPS dat

Jako externí paměťový člen, sloužící k uchování informací z GPS modulu byla zvolena Sériová FLASH paměť M25P16 od společnosti ST Microelectronic. Jedná se o 16MBitovou paměť využívající komunikační rozhraní SPI. Paměť obsahuje 32 sektorů každý s 256 stránkami. Každá stránka pak obsahuje 256 adresovatelných bytů. V hexadecimálním vyjádření má tedy paměť 200000_{16} bytů. Adresa každého bytu má délku 24bitů v rozsahu 000000_{16} – $1F0000_{16}$. Číst i zapisovat data je možné po jednotlivých bytech nebo stránkách paměti, tedy 256 bytů pomocí jedné instrukce. Obsah paměti je možno mazat sektorově nebo celkově [14].

3.3.1 SPI komunikace

Komunikace prostřednictvím rozhraní SPI probíhá za pomoci čtyř signálových vodičů označených *D*, *Q*, *S* a *CLK*. Vstup s označením *D* je užit jako datový vstup pro příjem instrukcí, výstup *Q* slouží pro vysílání jednotlivých dat z paměti. *CLK* je hodinový vstup dle kterého se paměť synchronizuje s řídícím obvodem, *S* je výběrový vstup a zahajuje nebo ukončuje komunikaci s pamětí Instrukční sada komunikace je detailně popsána v katalogovém listu paměti. Z této sady byly využity pouze některé příkazy, jež jsou uvedeny a popsány v tab. 9. Komunikace je zahájena logickou úrovní 0 na vstupu *S*. Od této chvíle začíná paměť s každou nástupnou hranou hodinového signálu na vstupu *CLK* načítat hodnoty ze vstupu *D*. Úvodních 8 hodnot paměť považuje za instrukci. Podle typu přijaté instrukce jsou následně přijímány další doplňující data /například adresa/ nebo je zahájeno vysílání dat na výstupu *Q*. Zde jsou data platná opět při nástupné hraně hodinového signálu. Ukončení komunikace je provedenou změnou logické úrovně vstupu *S* na hodnotu 1. Samotná komunikace v této práci je pak rozdělena pouze na čtení nebo zápis jednotlivých bytů na požadovanou adresu. Zápisem jednoho bytu se rozumí změna

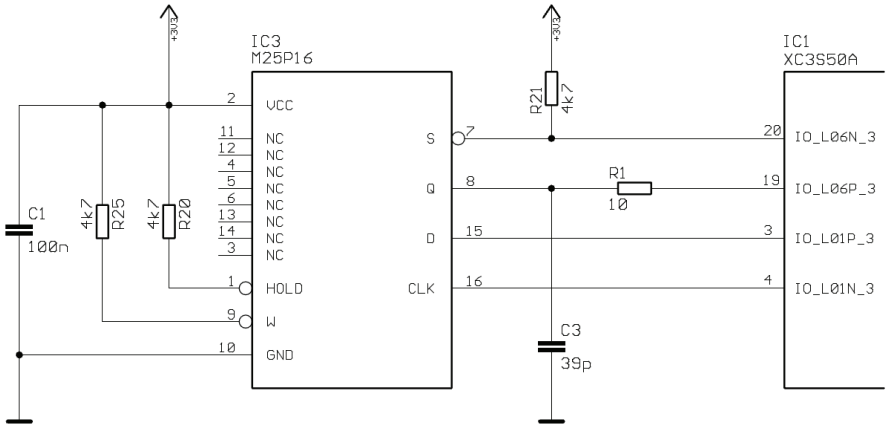
logické úrovně požadovaného bitu z hodnoty 1 na hodnotu 0 nebo zachování stávající hodnoty. Opačný smysl, tedy změna logické úrovně z hodnoty 0 na hodnotu 1 je možné provést pouze příkazem mazání, a to mazáním zvoleného sektoru nebo celé paměti hromadně [14].

Tab. 9: Popis použitých instrukcí při komunikaci s externí FLASH pamětí

Typ instrukce	Kód instrukce	Počet adresových bytů	Počet datových bytů
Povolení zápisu	06_{16}	0	0
Čtení stavového registru	05_{16}	0	1
Čtení bytu	03_{16}	3	1 – ∞
Zápis bytu	02_{16}	3	1 – 256
Mazání sektoru	$D8_{16}$	3	0
Mazání celého obsahu	$C7_{16}$	0	0

3.3.2 Zapojení externí paměti FLASH

Paměť je napájena napětím +3,3 V. Její proudová spotřeba je během aktivity maximálně 15 mA, v klidu pak odběr nepřesahuje 50 μA. Zapojení paměti je zobrazeno na obr. 12. Kromě popsaných vstupů a výstupu *D*, *Q*, *S* a *CLK*, které jsou připojeny na vstupy obvodu FPGA, obsahuje paměť další vstupy *HOLD* a *W*. Tyto vstupy umožňují hardwarové blokování zápisu do paměti a přerušení probíhající komunikace. Obě funkce jsou aktivní přivedením úrovně logické 0, avšak v popisovaném zařízení nejsou využívány. Proto jsou trvale přivedeny na úroveň logické 1 přes rezistory *R20* a *R25*. Rezistor *R1* a kondenzátor *C3* jsou zapojeny jako filtr proti nežádoucím zákmitům při komunikaci, tak jak doporučuje výrobce paměti. Kondenzátor *C1* slouží jako blokovací.



Obr. 12: Zapojení FLASH paměti a spojení s obvodem FPGA

3.4 GPS modul LEA – 5s

Pro stanovení navigačních dat, tedy zeměpisné šířky a délky, nadmořské výšky a rychlosti pohybu je použit GPS modul LEA – 5s. K přenosu dat využívá tento modul sběrnici UART, přičemž jako přenosový protokol je použit NMEA 0183. Parametry UART sběrnice jsou uvedeny v tab. 10. Po připojení napájecího napětí zahájí GPS modul vysílání řetězců RMC, VTG, GGA, GSA, GSV, GLL s frekvencí 1 Hz a to i v případě, že modul není schopen stanovit svoji pozici vlivem nedostatečného signálu. Řetězce však v tomto případě obsahují pouze úvodní, koncové a oddělovací znaky. Znaky nesoucí datové informace jsou vynechány.

Tab. 10: Parametry UART sběrnice GPS modulu

Parametr	**Hodnota**
Přenosová rychlost	9600 $bit.s^{-1}$
Počet datových bitů	8
Typ parity	Žádná
Počet stop bitů	1

Modul LEA – 5s je napájen stejnosměrným napětím +3,3 V na svorce *Vcc*, které musí být schopno zajistit proudovou spotřebu modulu do 100mA. Zároveň modul obsahuje zálohovací napájecí vstup, *V_BCKP*, který slouží jako záložní napájecí vstup pro obvod reálného času (RTC) uvnitř modulu. Tento záložní napájecí vstup zároveň udržuje v paměti modulu data poskytovaná satelity, jež jsou využity k výpočtu polohy. Tím se

výrazně zkracuje čas nutný k určení polohy po opětovném připojení hlavního napájení. To je označováno jako tzv. „teplý start“ a výrobce modulu uvádí, že jeho délka by neměla překročit 1 sekundu. Bez užití záložního napájecího vstupu je určení polohy označováno jako „studený start“. Jeho délka může být v závislosti na síle signálu ze satelitů i několik minut. Záloha obvodu RTC je zajištěna trvalým připojením svorky *V_BCKP* na primární zdroj napětí (akumulátorové baterie).

Pro příjem vysokofrekvenčního signálu ze satelitů můžeme využít pasivní nebo aktivní anténu. Pro dosažení co nejmenších rozměrů a také z finančních důvodů byla vybrána pasivní anténa se ziskem 2 dB. Signál z antény je přiváděn na vstupní svorku *RF_IN*. Pro tu je třeba dodržet impedanční přizpůsobení vstupní svorky modulu 50 Ω při frekvenci signálu 1575 MHz. K výpočtu tvaru a rozměru spoje *RF_IN* vstupu s anténou na desce plošných spojů byl použit návrhový program AppCAD společnosti Agilent [7]. Tento program je schopen při zadání rozměru a druhu základního materiálu, tloušťce měděné fólie, izolační mezeře mezi výplní a zemnící plochou, délce a tloušťce spoje určit impedanci pro zvolenou frekvenci. Zvolené parametry jsou uvedeny v tab. 11. Pro dosažení lepšího příjmu signálu při použití popsané pasivní antény doporučuje výrobce modulu doplnit okolí antény a modulu co největším počtem prokovení mezi zemnícími vrstvami výplně desky plošných spojů.

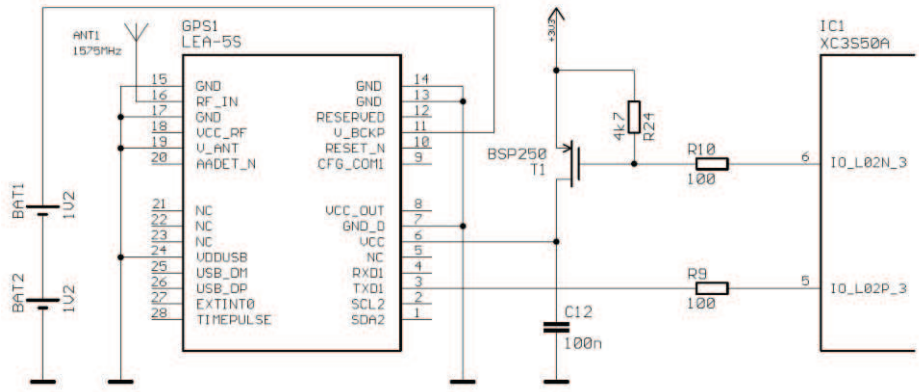
Tab. 11: Vstupní parametry programu AppCAD pro výpočet impedančního přizpůsobení

Parametr	Hodnota
Základní materiál	FR 4
Tloušťka základního materiálu	1,5 mm
Tloušťka měděné fólie	18 μm
Počet vrstev	2
Délka spoje	9,5000 mm
Tloušťka spoje	1,7210 mm
Izolační mezera mezi spojem a výplní	0,4064 mm
Frekvence	1575 MHz

3.4.1 Zapojení GPS přijímače LEA – 5s

GPS přijímač LEA – 5s je zapojen dle standardního doporučení výrobce [2]. Toto zapojení je znázorněno na obr. 13. Napájecí napětí přijímače je spínáno PMOS tranzistorem *T1*, který je ovládán obvodem FPGA. To umožňuje úplné odpojení přijímače

v době, kdy není využíván a tím výrazně snížit proudové zatížení napájecích akumulátorů *BAT1* a *BAT2*. Výstupní svorka *TXD1*, ze které jsou čteny GPS data je propojena s obvodem FPGA přes rezistor *R9*. Zápis do GPS přijímače přes sériovou linku není zařízením využíván, proto je svorka *RXD1* nezapojena.



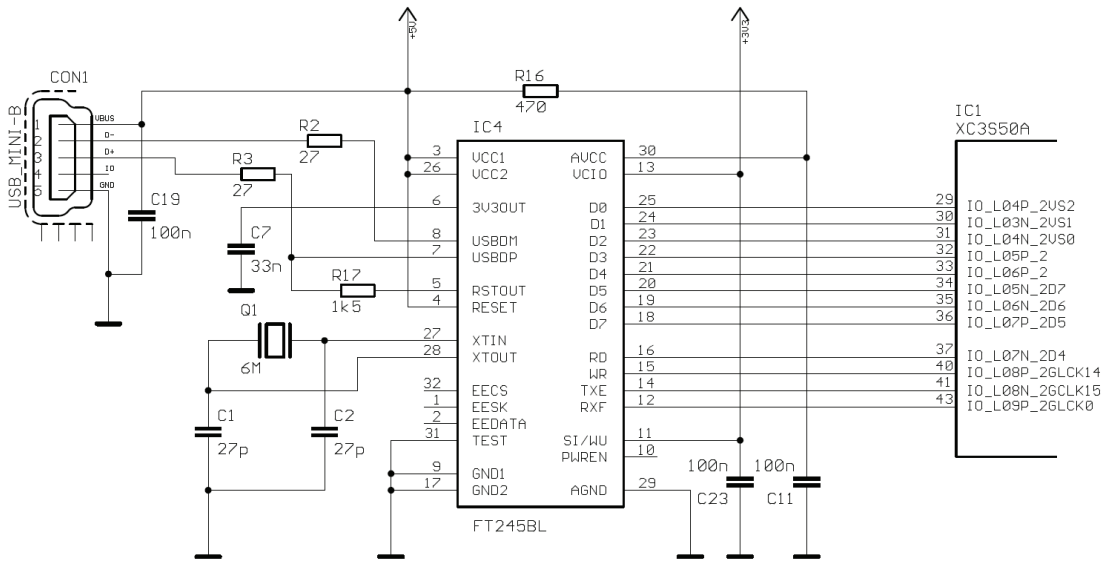
Obr. 13: Schéma zapojení GPS přijímače LEA-5s

3.5 USB komunikace

Data získaná z GPS přijímače a uložená v paměti je nutné přenášet do připojeného PC k dalšímu zpracování. Pro komunikaci zařízení s PC bylo vybráno v současné době nejpoužívanější rozhraní USB.

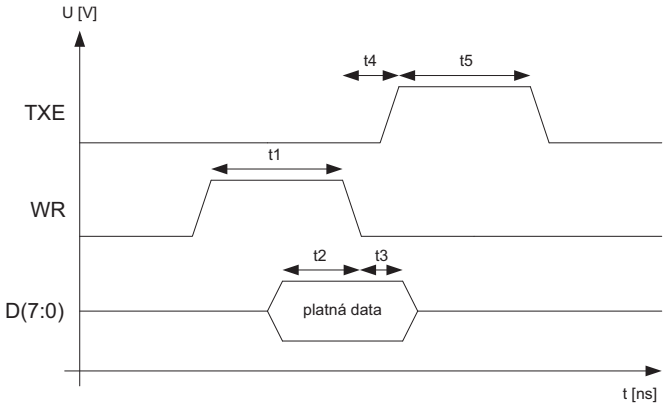
3.5.1 Popis komunikace prostřednictvím obvodu FT245BL

Vlastní komunikaci zajišťuje obvod FT245BL společnosti FTDI [15]. Použité zapojení je zobrazeno na obr. 14. Obvod je propojen s FPGA prostřednictvím sedmi datových vodičů *D0 – D7* a čtyř řídících *RXF*, *TXE*, *RD* a *WR*. Obvod využívá dvou napájecích napětí +3,3 V a +5 V. Napětí +5V slouží pro komunikaci prostřednictvím USB sběrnice a z té je také získáváno. Napětí +3,3 V je pak dodáváno z napěťového měniče TPS61016. Kondenzátory *C11*, *C19* a *C23* jsou blokovací, stejně tak kondenzátor *C7*. Ten stabilizuje vnitřní LDO obvod, který v tomto případě není využit. K chodu vnitřního oscilátoru na požadované frekvenci slouží krystal *Q1* ve spojení s kondenzátory *C1* a *C2*. Rezistory *R2*, *R3*. *R16* a *R17* jsou pak zapojeny dle požadavků výrobce pro zvolený způsob použití obvodu FT245BL. Jako propojovací konektor USB sběrnice je použit typ MINI – B.



Obr. 14: Zapojení obvodu FT245BL a propojení s obvodem FPGA

Tento obvod je po připojení k PC detekován a operační systém s podporou ovladačů tohoto obvodu standardně vytváří virtuální sériový port (VCP). K němu je následně možno libovolným softwarem přistupovat jako ke klasickému sériovému portu a provádět čtení nebo zápis. Komunikace mezi obvodem FT245BL a FPGA je paralelní s šířkou datové sběrnice 8 bitů. Časový diagram zápisu dat do obvodu FT245BL zobrazuje obr. 15. Výchozím okamžikem odesílání dat je úroveň logické 0 na výstupu *TXE*, což signalizuje schopnost přijímat nová data určená k přenosu. Povel k odeslání dat je proveden sestupnou hranou na vstupu *WR*, při níž jsou načtena data ze vstupů *D0 – D7*. Průběh odesílání dat je signalizován logickou úrovní 1 na výstupu *TXE*. Jsou zde vyznačeny časové rozestupy jednotlivých kroků komunikace, které je nutné dodržet. Hodnoty těchto rozestupů jsou uvedeny v tab. 12 Minimální čas nutný k odeslání jednoho bytu dat je dán součtem času *t1*, *t4* a *t5* a má hodnotu 135 ns.

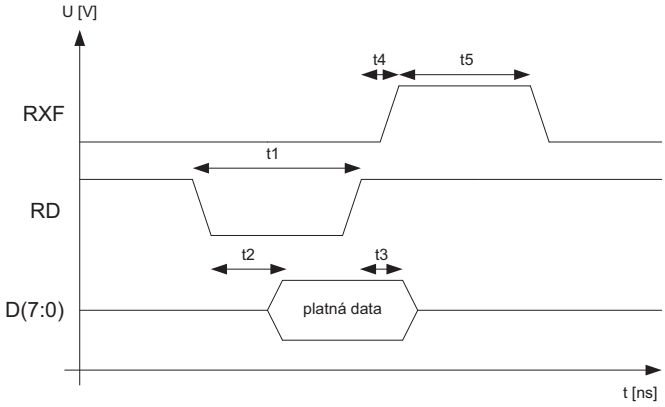


Obr. 15: Časový diagram zápisu dat do obvodu FT245BL

Tab. 12: Hodnoty požadovaných časů při zápisu dat do obvodu FT245BL

Časový interval	Minimální hodnota [ns]	Maximální hodnota [ns]
t1	50	-
t2	20	-
t3	0	-
t4	5	25
t5	80	-

Obdobným způsobem je prováděno také čtení dat, která obvod FT245BL přijme ze sběrnice USB. Časový diagram čtení dat je zobrazen na obr. 16, příslušné hodnoty časových údajů jsou uvedeny v tab. 13. Přijetí nových dat je signalizováno logickou úrovní 0 na výstupu *RXF*. V tomto okamžiku jsou nástupnou hranou na vstupu *RD* tato data přepsána na datový výstup. To je signalizováno logickou úrovní 1 na výstupu *RXF*. Tato úroveň zůstává zachována až do chvíle, kdy jsou přijata nová data.



Obr. 16: Časový diagram průběhu čtení nových dat z obvodu FT245BL

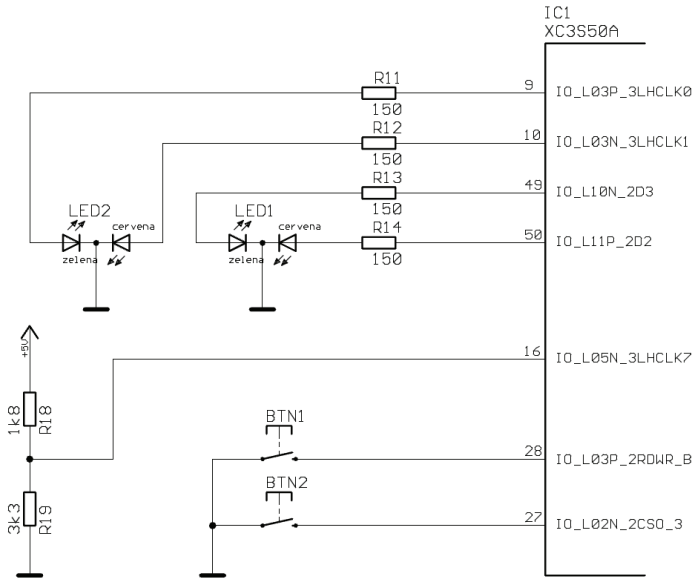
Tab. 13: Hodnoty požadovaných časů při čtení dat z obvodu FT245BL

Časový interval	Minimální hodnota [ns]	Maximální hodnota [ns]
t1	50	-
t2	20	50
t3	0	-
t4	0	25
t5	80	-

3.6 Ovládací prvky zařízení

Základním řídícím členem celého zařízení je napěťový dělič tvořený rezistory *R18* a *R19*. Ten slouží k detekci spojení zařízení s PC prostřednictvím USB rozhraní. V případě spojení je na tento dělič přivedeno napětí +5V z USB rozhraní a dělící poměr vytváří napětí odpovídající logické úrovni 1. Tento stav je detekován řídícím obvodem FPGA, který přepíná zařízení do stavu komunikace. V opačném případě, kdy není na rezistorovém děliči žádný potenciál, což odpovídá úrovni logické 0, je zařízení ve stavu příjmu dat z GPS. Uživatel zařízení je o stavu, v jakém se zařízení nachází, informován dvojící LED diod. Obě tyto diody jsou schopny vytvářet zelené, červené nebo oranžové světlo. Dioda *LED1* informuje o režimech během zaznamenávání dat z GPS modulu, *LED2* informuje o režimech komunikace s PC. Anody diod *LED1* a *LED2* jsou připojeny k výstupům FPGA přes předřadné rezistory *R11 – R14* a jsou aktivní při logické úrovni 1 na příslušném výstupu.

K dalším ovládacím prvkům zařízení patří dvojice tlačítek *BTN1* a *BTN2*, které jsou propojeny přímo s obvodem FPGA. Na těchto vstupech jsou aktivovány vnitřní „pull-up“ rezistory, udržující logickou úroveň 1 v době rozpojení tlačítek. Stisk tlačítka pak na příslušný vstup přivádí úroveň logické 0. Schematické znázornění popsaných ovládacích členů je zobrazeno na obr. 17.



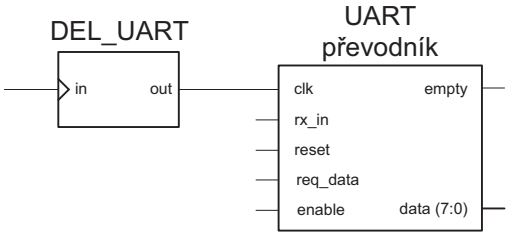
Obr. 17: Schéma zapojení tlačítek, rezistorového děliče a informačních LED diod

4 Design FPGA

Softwarový popis funkce zařízení je rozdělen do dvou částí. První tvoří popis v jazyce VHDL, v němž byla vytvořena komponenta pro čtení sériové linky, rozšiřující porty procesoru PICOBLAZE a filtr zákmitů na tlačítkách. Druhá část je pak tvořena samotným popisem instrukcí procesoru PICOBLAZE v jazyce ASSEMBLER.

4.1 Čtení ze sériové linky GPS přijímače

Pro čtení výstupních dat z GPS modulu, které jsou dostupné prostřednictvím sériové linky, byla vytvořena samostatná komponenta *UART převodník*. Její základní funkcí je načítání sériových dat na vstupu a jejich převedení na paralelní 8bitové výstupní slovo. Uspořádání popisované komponenty je zobrazeno na obr. 18. Komponenta je uzpůsobena pro použitý GPS modul, předpokládá tedy čtení sériové linky s pevně definovanými parametry. Vstupy a výstupy komponenty *UART převodník* jsou popsány v tab. 14.



Obr. 18: Spojení komponent UART převodník a DEL_UART.

Tab. 14: Popis vstupních a výstupních portů komponenty UART převodník.

Port	Směr	Funkce
CLK	vstupní	Hodinový signál.
RX_IN	vstupní	Vstup sériové linky.
RESET	vstupní	Reset.
REQ_DATA	vstupní	Žádost o načtená data, aktivní při úrovni logické 1.
ENABLE	vstupní	Povolení funkce převodníku, aktivní při úrovni logické. 1.
EMPTY	výstupní	Stavu převodníku, úroveň logické 1 nejsou načtena data nebo probíhá jejich načítání, úroveň logické 0 data jsou načtena.
DATA (7:0)	výstupní	Data načtená ze sériové linky.

Převodník je synchronizován na nástupnou hranu hodinového signálu. Kmitočet tohoto signálu má 16krát vyšší frekvenci než je frekvence sériového přenosu. To znamená,

že za dobu trvání jednoho bitu přenosu, hodinový signál provede 16 cyklů. Vzorkování vstupu pak probíhá při 8. nástupné hraně počítané od začátku vysílání každého bitu přenosu. Tím je zajištěno správné vyhodnocení logických úrovní v datovém toku sériové linky. Výpočet požadované frekvence f_P pro přenosovou rychlost a velikost dělicího poměru N děličky *DEL_UART* jsou uvedeny v rovnicích 5 a 6.

Frekvence hodinového signálu pro převodník:

$$f_P = 16 \cdot 9600 \quad [Hz] \tag{5}$$

$$\underline{f_P = 153{,}6 \quad kHz}$$

Výpočet dělícího poměru děličky *DEL_UART*:

$$N = \frac{50 \cdot 10^6}{153{,}6 \cdot 10^3} \quad [-] \tag{6}$$

$$\underline{N \approx 325}$$

Zpětný přepočet výsledné frekvence:

$$f_v = \frac{50 \cdot 10^6}{325} \quad [Hz] \tag{7}$$

$$\underline{f_v = 153{,}846 \quad kHz}$$

Výpočet chyby výsledné frekvence:

$$f_v = \frac{153{,}846 - 153{,}6}{153{,}6} \cdot 100 \quad [\%] \tag{8}$$

$$\underline{f_v \approx 0{,}16 \quad \%}$$

Jelikož je možné vytvořit pouze děličku celým číslem, je vypočtený dělící poměr N zaokrouhlen. To však zanáší chybu do výsledné frekvence f_V, která je děličkou generována. V rovnici 8 je tato chyba procentuelně vyčíslena, její hodnota je 0,16%. Tato chyba se násobí počtem za sebou jdoucích bitů v přenosu. To představuje chybu 1,44 % při příjmu posledního bitu daného bytu. Ovšem díky faktu, že komponenta je vždy znovu synchronizována při každém starut bitu přenosu, nikdy chyba nepřekročí hodnotu 1,44 % a datový přenos je vyhodnocován správně.

4.1.1 Činnost komponenty

Načítání je vždy zahájeno logickou úrovní 0 na vstupu *RX_IN*. V tomto okamžiku je spuštěno čítání hodinového signálu a při jeho 8. nástupné hraně je znovu kontrolována

úroveň na vstupu *RX_IN*. Je li stále v úrovni logické 0, je definitivně považován za start bit datového přenosu a následuje čítání datových bitů dle popsaného pravidla násobné frekvence. Pokud je uloženo 8 bitů, je vyhodnocován stop bit definovaný logickou úrovní 1. V případě, že i tato kontrola proběhne úspěšně, je tento stav signalizován logickou úrovní 1 na výstupu *EMPTY*, čímž je definováno úspěšné načtení dat. Touto změnou log úrovně je vyvoláno přerušení procesoru, který v obslužném programu požádá o přepsání dat na výstup *DATA* pomocí logické úrovně 1 přivedené na vstup převodníku *REQ_DATA*. Pokud jsou data přepsána, logická úroveň na výstupu *EMPTY* se opět vrací do hodnoty 0 a je zahájeno nové načítání datového toku sériové linky. Následné zpracování načtených dat je provedeno rychleji, než při dané přenosové rychlosti sériové linky dojde k přijetí dalšího 8bitového slova. To umožnilo konstrukci tohoto převodníku bez využití vyrovnávací paměti pro případné uložení nezpracovaných dat. Popis komponenty *UART převodník* v jazyce VHDL je součástí přílohy této práce.

4.2 Rozšíření vstupních a výstupních portů procesoru PICOBLAZE

Jelikož funkci celého zařízení obsluhuje procesor PICOBLAZE, byl jeho základní počet jednoho vstupního a jednoho výstupního 8bitového portu nedostačující. Proto byly oba porty rozšířeny popisem v jazyce VHDL. Procesy *OUTPUT* a *INPUT* rozšiřují PICOBLAZE na 8 výstupních, respektive 6 vstupní portů. Přehled těchto portů s popisem jejich funkce je uveden v tab. 15 a tab. 16.

Tab. 15: Popis rozšíření vstupního portu procesoru PICOBLAZE

Číslo portu	Použité bit	Popis	Označení ve VHDL popise
0	bit 0 – 7	Data z komponenty UART převodník	RX_DATA(7:0)
1	bit 7	Vstupní data SPI komunikace	SPI_MISO
2	bit 0	TXE signál obvodu FT245BL	TXE
	bit 1	RXF signál obvodu FT245BL	RXF
3	bit 0	Tlačítko START	INT_BTN_START
	bit 1	Tlačítko STOP	INT_BTN_STOP
	bit 2	Detekce připojení USB konektoru	INT_SW
4	bit 0 – 7	Data přijatá obvodem FT245BL	USB_DATA(7:0)
5	bit 0	Nízké napětí akumulátorů	LOW_BAT

Tab. 16: Popis rozšíření výstupního portu procesoru PICOBLAZE

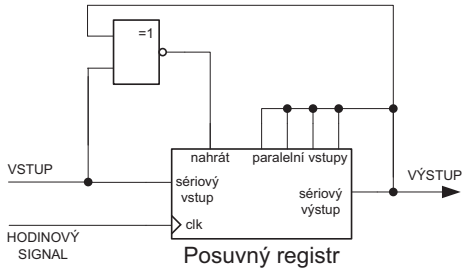
Číslo portu	Použité bit	Popis	Označení ve VHDL popise
0	bit 0	Výstupní data SPI komunikace	SPI_MOSI
1	bit 0	Hodinový signál SPI komunikace	SPI_CLK
	bit 1	Výběrový signál SPI komunikace	SPI_CS
2	bit 0	Žádost o načtení dat z UART převodníku	ULD_RX_DATA
3	bit 0	LED2 zelená	LED_INF(3:0)
	bit 1	LED2 červená	
	bit 2	LED1 červená	
	bit 3	LED1 zelená	
4	bit 0	RD signál obvodu FT245BL	RD
5	bit 0	WR signál obvodu FT245BL	WR
6	bit 0 – 7	Data odesílaná obvodem FT245BL	USB_DATA(7:0)
7	bit 0	Ovládáni tranzistoru T1	GATE
8	bit 0 – 7	Nastavení vysoké impedance portu č. 6	USB_DATA(7:0)

4.3 Vnější přerušení procesoru

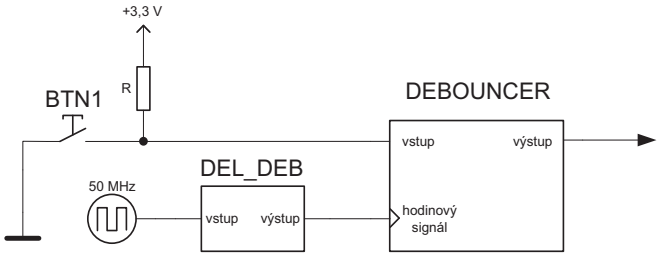
Jak bylo popsáno v teoretické části, je přerušovací vstup procesoru PICOBLAZE ovládán prostřednictvím klopného obvodu D. Jako podnět pro vyvolání přerušení je použit výstupní signál *EMPTY* z komponenty *UART převodník*. Přerušení je vyvoláno sestupnou hranou na tomto signále, což značí načtení dat *UART převodníkem* z GPS přijímače. Obslužný podprogram v procesoru PICOBLAZE pak tuto událost dále zpracovává. Popis přerušení je tvořen procesem *INTERRUPT*.

4.4 Ošetření zákmitů na vstupech tlačítek

K ošetření zákmitů, vznikajících při mechanickém spojení nebo rozpojení kontaktu tlačítka, byl jeho vstup doplněn filtrem zákmitů. Tento filtr je nazýván *DEBOUNCER*. Popis komponenty *DEBOUNCER* byl vytvořen v jazyce VHDL, schematické znázornění je zobrazeno na obr. 19. Aby byl tento filtr, tvořený vhodně zapojeným 4bitovým posuvným registrem, schopen odstranit zákmity z tlačítek v délce trvání několika ms, bylo nutné snížit základní frekvenci hodinového signálu pomocí děličky kmitočtu. Ta je tvořena komponentou *DEL_DEB* a snižuje frekvenci 2^{17} krát. Blokové schéma spojení komponenty *DEBOUNCER*, děličky *DEL_DEB* a tlačítka *BTN1* znázorňuje obr. 20. [16]



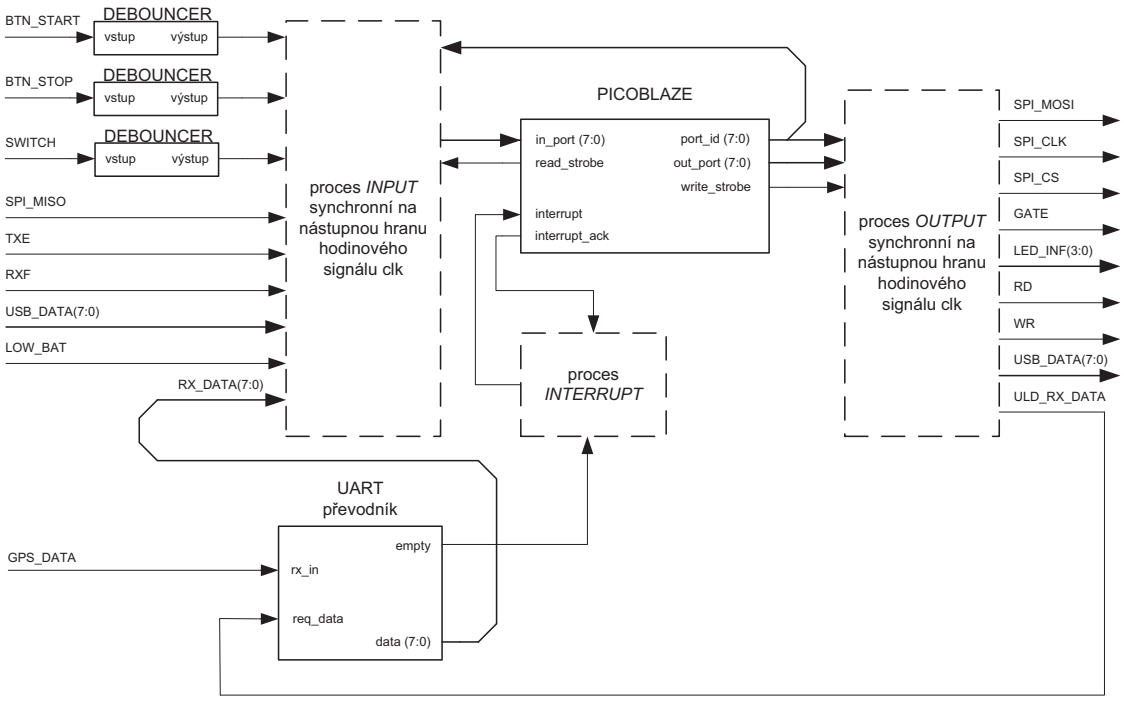
Obr. 19: Vnitřní zapojení filtru zákmitů tlačítka *DEBOUNCER*



Obr. 20: Spojení komponenty *DEBOUNCER*, děličky kmitočtu *DEL_DEB* a tlačítka *BTN1*

4.5 Entita GPS_TRACER

Výsledný VHDL kód popisující funkci celého zařízení tvoří entitu *GPS_TRACER*, která je zobrazena na obr. 21. Pro přehlednost schématu jsou jednotlivé symboly reprezentující příslušné komponenty zjednodušeny. Nejsou znázorněny vstupy hodinového signálu a globální reset, jelikož jsou tyto signály pro všechny komponenty společné. Komponenta *UART převodník* pak neobsahuje povolovací vstup *ENABLE*, který je trvale přiveden na logickou úroveň 1. Bloky ohraničené čárkovanou čarou jsou symbolicky naznačené procesy *INPUT*, *OUTPUT* a *INTERRUPT*.



Obr. 21: Entita *GPS_TRACER*

5 Software PICOBLAZE

Zařízení je možné přepnout do dvou pracovních režimů, ZÁZNAM a KOMUNIKACE. V režimu ZÁZNAM je aktivován GPS modul a jsou z něj přijímána data. V režimu KOMUNIKACE je GPS modul vypnut a je aktivovaná komunikace s PC prostřednictvím USB rozhraní. Režim, ve kterém se zařízení v danou chvíli nachází, je signalizován svícením příslušné LED diody červenou barvou. Volba režimu je prováděna automaticky detekcí připojeného USB kabelu v konektoru zařízení. Zvolený režim je aktivován stiskem tlačítka *BTN_START*, což se projeví změnou barvy LED diody na oranžovou. Stiskem tlačítka *BTN_STOP* se zvolený režim deaktivuje a barva LED se mění zpět na červenou.

V aktivovaném režimu ZÁZNAM je úspěšné určení polohy GPS modulem signalizováno změnou barvy příslušné diody LED na zelenou. Možná ztráta signálu se projeví změnou barvy LED diody zpět na oranžovou. Pokud dojde v tomto režimu k chybnému uložení dat v paměti FLASH nebo je paměť plná, je tento stav signalizován blikáním LED diody červenou barvou.

Režim KOMUNIKACE v aktivním režimu vyhodnocuje data přijímaná z USB rozhraní. Dojde li k úspěšnému navázání komunikace se softwarem v PC, změní příslušná LED dioda svoji barvu na zelenou. Každý následný vykonaný povel, který je přijat z PC, je po úspěšném vykonání signalizován krátkým zhasnutím a opětovným rozsvícením LED diody.

Pokud zařízení zjistí pokles napětí na napájecích akumulátorech pod povolenou hodnotu, jsou režimy deaktivovány a tento stav je signalizován střídavým zapínáním a vypínáním obou LED diod v červené barvě.

Součástí přílohy je vývojový diagram, znázorňující program procesoru PICOBLAZE.

5.1 Režim ZÁZNAM

5.1.1 Vyhodnocování přijímaných dat

Při přepnutí zařízení do režimu ZÁZNAM a jeho aktivaci procesor sepne tranzistor *T1*, který přivede napájecí napětí na GPS modul LEA – 5s. Zároveň je aktivováno povolení přerušení. Procesor následně čeká ve smyčce, kde probíhá kontrola stisku tlačítka *BTN_STOP* a kontrola nízkého napětí na baterii. Pokud jsou načtena data ze sériové linky GPS přijímače, je vyvoláno přerušení a procesor odskakuje na obslužnou proceduru. Zde pomocí výstupu *ULD_RX_DATA* požádá o přepsání přijatých dat na vstup *RX_DATA*, která následně uloží do příslušného registru a obslužnou proceduru ukončuje.

Nyní zahájí procesor identifikaci přijatých dat. Ta je rozdělena do 3 částí, podle stavu, v jakém se v danou chvíli procesor nachází. První je stav, kdy nebyl doposud načten žádný znak, v druhém již probíhá načítání řetězce RMC a ve třetím načítání řetězce GGA.

V prvním stavu se procesor nachází do chvíle, než je načtena za sebou jdoucí kombinace znaků „*$GPRMC*“. Poté procesor přechází do druhého stavu, ve kterém čítá následující přijímané znaky a na 17. pozici od začátku tohoto řetězce kontroluje znak „*A*“. Ten symbolizuje, že GPS přijímač určil polohu a vysílaná data obsahují potřebné informace. Je li tato podmínka splněna, procesor začíná ukládat data z řetězce RMC do vlastní paměti. V opačném případě přechází procesor zpět do prvního stavu.

Pokud je řetězec RMC úspěšně načten, přechází procesor do třetího stavu. Tento stav nejprve sleduje přijetí za sebou jdoucích znaků v kombinaci „*$GPGGA*“. Pokud jsou tyto znaky rozpoznány, začíná čítání znaků v tomto řetězci a ukládání jeho obsahu do paměti. Je li i tento řetězec úspěšně načten, procesor vypíná povolení přerušení a načtená data dále zpracovává. Následně je opět povoleno přerušení a procesor přechází zpět do prvního stavu.

5.1.2 Organizace a obsah ukládaných dat do paměti

V režimu záznam jsou z přijímače GPS načítány řetězce RMC a GGA. Z jejich obsahu jsou vybírána pouze data, důležitá pro určení polohy. Z věty RMC jsou to zeměpisná šířka, zeměpisná délka, rychlost pohybu, datum a čas. Z věty GGA pak pouze nadmořská výška. Ke každému zaznamenávanému údaji je navíc nutné přidat identifikační číslo, jehož hodnota odděluje množiny jednotlivých záznamů a stavový BYTE. Z popisu řetězců RMC a GGA v kapitole 2.3 vyplývá, že by musel mít každý záznam délku 41 BYTE, jelikož každý znak v řetězci má velikost 1 BYTU. Aby bylo možné uložit do paměti více záznamů, bylo nutné zefektivnit způsob ukládání a snížit tak počet bytů nutných pro uložení jednoho záznamu. Vycházelo se z faktu, že všechna ukládaná data mají podobu čísel v rozsahu hodnot 0 – 9, proto je pro uložení jednoho znaku potřebné pouze 4 bitu prostoru v paměti. To umožňuje ukládat do jednoho adresovatelného BYTU vždy dva znaky z řetězce společně. Tímto postupem došlo ke snížení potřebného prostoru pro uložení jednoho záznamu na velikost 23 BYTŮ. Organizace dat jednoho záznamu je popsána v tab. 17.

Tab. 17: Organizace dat jednoho záznamu v paměti

Pořadové číslo	BIT 7 – 4	BIT 3 – 0	Typ dat	POPIS
0	H1	H0	Čas	H1 – desítky hodin H0 – jednotky hodin
1	M1	M0		M1 – desítky minut M0 – jednotky minut
2	S1	S0		S1 – desítky sekund S0 – jednotky sekund
3	L8	L7	Zeměpisná šířka	L8 – desítky stupňů L7 – jednotky stupňů
4	L6	L5		L6 – desítky minut L5 – jednotky minut
5	L4	L3		L4 – desetiny minut L3 – setiny minut
6	L2	L1		L2 – tisíciny minut L1 – desítitisíciny minut
7	L0	–		L0 – stotisíciny minut
8	Y9	Y8	Zeměpisná délka	Y9 – stovku stupňů Y8 – desítky stupňů
9	Y7	Y6		Y7 – jednotky stupňů Y6 – desítky minut
10	Y5	Y4		Y5 – jednotky minut Y4 – desetiny minut
11	Y3	Y2		Y3 – setiny minut Y2 – tisíciny minut
12	Y1	Y0		Y1 – desítitisíciny minut Y0 – stotisíciny minut
13	V3	V2	Rychlost pohybu	V3 – desítky nebo jednotky uzlů V2 – jednotky nebo desetiny uzlů
14	V1	V0		V1 – desetiny nebo setiny uzlů V0 – setiny nebo tisíciny uzlů
15	D1	D0	Datum	D1 – Desítky dnů D0 – jednotky dnů
16	m1	m0		m1 – desítky měsíců m0 – jednotky měsíců
17	R1	R0		R1 – desítky roků R0 – jednotky roků
18	N3	N2	Nadmořská výška	N3 – stovky metrů N2 – desítky metrů
19	N1	N0		N1 – jednotky metrů N0 – desetiny metrů

20	Sb	Stav	Varianta záznamu rychlosti
21	T	Tep	Srdeční tep
22	IT	Identifikační číslo trasy	

Z principu funkce GPS přijímače, který při překročení rychlosti pohybu nad hodnotu 10 uzlů odebere z řetězce hodnoty tisícin a přidá hodnoty desítek uzlů a zároveň posune desetinnou čárku, je nutné tento fakt připojit k záznamu. K tomuto účelu slouží STATUS BYTE, který nabývá hodnoty 0 při variantě záznamu rychlosti do 10 uzlů a hodnoty 1 při jejím překročení. To je nutné pro pozdější správné vyhodnocení záznamu v PC. Byte TEP je rezervován pro možnost pozdějšího rozšíření zařízení o modul měřící tepovou frekvenci srdce. Identifikační číslo trasy pak svojí hodnotou sjednocuje jednotlivé záznamy, aby je bylo v PC možno vzájemně oddělit. Tato hodnota je při smazání paměti FLASH nastavena na 1. Při každé deaktivaci režimu ZÁZNAM, v jehož předchozím aktivním režimu bylo prováděno ukládání do paměti FLASH, je identifikační číslo trasy inkrementováno. Jde tedy o čítač aktivních režimů ZÁZNAM. Poslední tři byty záznamu nejsou, jako jediné, děleny na dvě čtveřice bit, můžou tedy nabývat hodnoty 0 – 255.

Během přenosu dat mezi GPS modulem a procesorem může dojít ke ztrátě nebo narušení přenášených dat, což se projeví nežádoucí hodnotou v přijímaném řetězci. Tato chyba se může projevit kdykoliv během přenosu. V případě odhalení takové chyby by bylo nutné smazat zapsaný BYTE a nahradit jej novými hodnotami. Avšak mazání dat je u použité FLASH paměti možné pouze po jednotlivých sektorech, což by v tomto případě vedlo ke ztrátě velkého množství dat, která byla uložena v paměti před zjištěním případné chyby. Z tohoto důvodu není možné ukládat data přímo do FLASH paměti během jejich příjmu, ale využít pro ukládání paměť dat procesoru RAM. Do této paměti jsou data ukládána v průběhu příjmu a vyhodnocování řetězců z GPS modulu a až pokud dojde k úspěšnému načtení všech potřebných hodnot, tedy kompletaci všech 23 BYTŮ, je jejich obsah překopírován z paměti RAM do paměti FLASH. Obsah paměti RAM je pak znovu využit k tvorbě dalšího záznamu.

5.1.3 Systém ukazatelů

Aby bylo možné využít efektivně celou paměť FLASH, jsou jednotlivé záznamy, tvořené 23 byty, ukládány bezprostředně za sebou. V případě že paměť již obsahuje nějaká data a zařízení je znovu aktivováno, tedy zahájí nový záznam, je nezbytné vědět, kde je konec předešlého záznamu a kde v paměti FLASH je možné nová data uložit. Pro vyřešení

tohoto problému byl vytvořen systém ukazatelů. Ty jsou tvořeny dvěma trojicemi bytů, které v sobě nesou absolutní adresu začátku a konce záznamu dat v paměti FLASH. Pokud je paměť prázdná, mají ukazatelé stejnou hodnotu, tedy 010000_{16}. Z této hodnoty je zřejmé, že k ukládání dat není využit první sektor paměti. Tento fakt je dán nutností ukládat do paměti FLASH i popisované ukazatele. Ty však mění svoji hodnotu s průběhem ukládání dat a je nutné je aktualizovat. To je možné pouze smazáním celého sektoru a novým zapsáním hodnot. Z tohoto důvodu nemohou být data a ukazatele uloženy v totožném sektoru. Aby nedocházelo k častému mazání prvního sektoru paměti, jsou ukazatelé během aktivního režimu uloženy v paměti RAM, kde je s nimi také pracováno. Až během deaktivace režimu jsou aktuální hodnoty ukazatelů uloženy do paměti FLASH, což opět urychluje činnost zařízení.

Ukazatele obsahující adresu kde je možné zahájit nový zápis dat, jsou v programu nazýváni *UKAZ_END* s indexem *0* až *2*. Adresa v nich uložená přímo definuje místo, kde je možné zapsat první byte nového záznamu. Hodnota těchto ukazatelů se mění vždy, když je proveden zápis do paměti FLASH.

Druhá trojice ukazatelů, v programu označená *UKAZ_START 0* až *2*, definuje adresu prvního bytu záznamu dat. Ty ukazatele jsou využívány při přenosu dat z paměti zařízení do PC a to tak, že definují místo, kde začíná záznam dat, která doposud nebyla přenesena do PC. Pokud je příkazem z PC požadováno označit uložená data v paměti FLASH za načtená, jsou hodnoty ukazatelů *UKAZ_START* přepsány hodnotami ukazatelů *UKAZ_END*. V dalším přenose dat do PC pak nejsou stejná data přenášena opětovně, avšak v paměti zařízení jsou stále uložena. Pokud je z nějakého důvodu nutné přenést do PC i data dříve označená jako načtená, provede zařízení, podle příslušného příkazu z PC, přenos všech dat uložených mezi adresou 010000_{16} a adresou uloženou v ukazatelích *UKAZ_END* a to bez ohledu na hodnoty v ukazatelích *UKAZ_START*.

5.2 Režim KOMUNIKACE

V tomto režimu není GPS modul aktivní a zařízení pouze sleduje komunikaci s PC prostřednictvím USB sběrnice. Obslužný software komunikuje se zařízením odesíláním povelů, které jsou uvedeny v tab. 18. Tyto povely jsou tvořeny jedním bytem definované hodnoty. Pro přehlednost v programu je tento povel zároveň definován i jako znak odpovídající dané hodnotě v tabulce ASCII.

Tab. 18: Seznam povelů použitých při komunikaci zařízení a softwaru v PC

Hodnota povelu	Znak v tabulce ASCII	Popis
65	A	Žádost o zaslání všech dat uložených ve FLASH paměti
67	C	Smazání celého obsahu paměti FLASH
68	D	Žádost o zaslání nových dat uložených ve FLASH paměti
69	E	Žádost o zaslání ukazatelů *UKAZ_END*
79	O	Označení stávajících dat v paměti FLASH za načtená
80	P	Žádost o zaslání ukazatelů *UKAZ_START* a *UKAZ_END*

V případě, že zařízení rozpozná některý z uvedených povelů, adekvátně reaguje, případně odesílá definovanou odpověď. Ta se skládá z úvodního znaku, určující typ odpovědi, a následných dat. Úvodní znak je nutné odeslat pro zpětné vyhodnocení přijatých dat v softwaru PC. Za odpovědí s úvodním znakem 3 je vždy dále odeslán druhý a třetí byte. Druhý informuje, o jakou odpověď se jedná a třetí nese informaci o obsahu této odpovědí. Popis úvodních znaků je uveden v tab. 19.

Tab. 19: Seznam úvodních znaků odpovědí zařízení na povely z PC

Hodnota úvodního znaku	Popis
1	Budou odesláni ukazatelé UKAZ_STAR a UKAZ_END
2	Budou odesláni ukazatelé UKAZ_END
3	Bude odeslána odpověď
4	Budou odeslána uložená data

Bezprostředně po úvodním znaku jsou odeslána příslušná data. V případě hodnoty 1 jsou to ukazatelé v pořadí *UKAZ_START0*, *UKAZ_START1*, *UKAZ_START2* a *UKAZ_END0*, *UKAZ_END1*, *UKAZ_END2*. V případě hodnoty 2 jsou to pouze *UKAZ_END0*, *UKAZ_END1*, *UKAZ_END2*. Za znakem s hodnotou 3 následuje dvojice bytů. První z nich nese informaci na jaký druh povelu je odpovídáno, druhý obsahuje samotnou odpověď, která může nabývat hodnot odpovídajícím znakům „*O*“ nebo „*E*“ v tabulce ASCII. Znak „*O*“ signalizuje, že předešlý povel proběhl úspěšně, znak „*E*“ pak signalizuje chybu při provádění povelu. Počáteční znak s hodnotou 4 uvozuje odeslání obsahu paměti a to jak při přenosu nových dat, tak při přenosu celého obsahu paměti. Zaznamenaná data jsou odesílána v pořadí od nejstaršího po nejnovější záznam.

6 Software PC

K tvorbě aplikace v PC, která bude schopna komunikovat se zařízením a dokáže vhodně zpracovávat přijatá data, byl vybrán program DELPHI 2009 společnosti EBARCADERO TECHNOLOGIES [17]. Tvorba této aplikace byla rozdělena do tří oddělených částí. V první byly vytvořeny procedury pro komunikaci se zařízením, v druhé procedury ukládající přijatá data a ve třetí části pak procedury pro zpracování těchto dat.

6.1 Obecný popis aplikace

Základní okno aplikace je zobrazeno na obr. 22. Aplikaci je možno ovládat pomocí lišty hlavního menu, které tvoří několik podskupin tlačítek. V první podskupině *PROGRAM* je obsažena pouze volba ukončení aplikace. Podskupina *NASTAVENÍ* obsahuje tlačítko *KOMUNIKACE*, které otevírá okno pro nastavení přenosu dat mezi aplikací a zařízením. V podskupině tlačítek *PŘENOS* jsou tlačítka uskutečňující samotnou komunikaci. Jsou zde tlačítka *SPÁROVAT ZAŘÍZENÍ*, *ODPOJIT ZAŘÍZENÍ*, *STAŽENÍ NOVÝCH DAT*, *STAŽENÍ VŠECH DAT* a *SMAZÁNÍ VŠECH DAT*. Poslední podskupina *ZPRACOVÁNÍ* umožňuje za pomocí tlačítek *GRAF*, *MAPA* a *VLOŽIT KOMENTÁŘ* uložená data dále zpracovávat a zobrazovat.

Celé okno aplikace je tvořeno komponentou *StringGrid*, která vytvoří tabulku sumarizující uložené záznamy. Ta obsahuje sloupce informující o pořadovém čísle záznamu, datu jeho uložení, časové délce, průměrné rychlosti, maximálním převýšení a o délce trasy. Její součástí je i nepovinný sloupec komentář, určený pro snadnější orientaci v záznamech. První sloupec této tabulky je tvořen čtverci, kterými je možno daný záznam označit. Takto označený záznam je pak použit při zpracovávání.

Při spuštění aplikace, nebo jsou li přijata nová data, je vždy provedena aktualizace této tabulky, což je provedeno výpočty daných hodnot pro každou skupinu záznamů. K tomuto účelu byly využity implementované procedury, které tyto jednoduché výpočty provádějí automaticky. Pouze pro výpočet délky trasy byla vytvořena funkce s názvem *DRÁHA*. Ta je schopna vypočítat vzdálenost dvou bodů zadaných pomocí hodnot zeměpisné šířky a zeměpisné délky. Tuto vzdálenost cyklicky určí pro všechny sousedící body dané skupiny záznamu a následným součtem stanoví celkovou délku trasy. K výpočtu byla zvolena metoda pro stanovení vzdálenosti dvou bodů ležících na kulové ploše. Tato vzdálenost je nazývá ORTODROMA a platí:

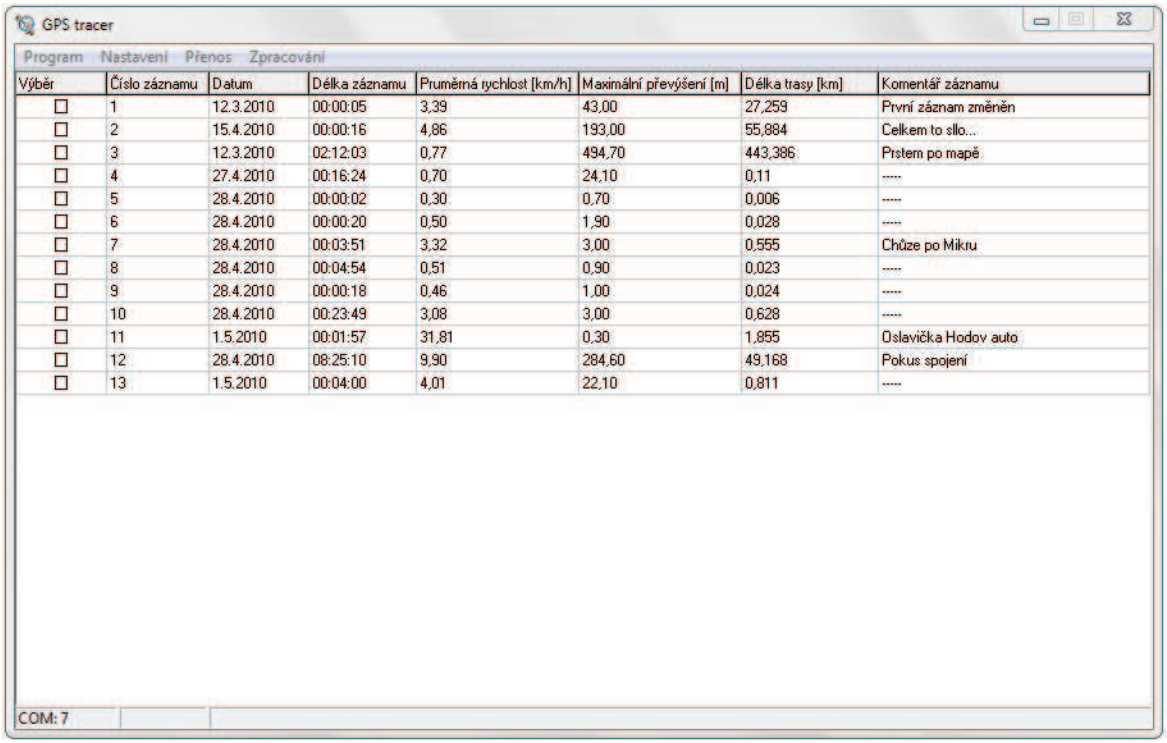
$$o = r \cdot \cos^{-1}(\alpha), \tag{9}$$

kde o je délka ORTODROMY, r je poloměr koule pro kterou je ORTODROMA určována a α je úhel určený vztahem:

$$\alpha = \sin A_{ZŠ} \cdot \sin B_{ZŠ} + \cos A_{ZŠ} \cdot \cos B_{ZŠ} \cdot \cos(B_{ZD} - A_{ZD}), \qquad (10)$$

kde $[A_{ZŠ}, A_{ZD}]$ je výchozí bod a $[B_{ZŠ}, B_{ZD}]$ je cílový bod. Tyto body jsou určeny stupni zeměpisné šířky a délky. Spojení rovnic 9 a 10 je ve funkci *DRÁHA* využito pro přesnější stanovení délky trasy [19].

V dolní části hlavního okna aplikace je umístěn informační panel obsahující dvě položky. V první je zobrazeno číslo vybraného sériového portu, v druhém informace o spojení se zařízením. Úspěšné navázání spojení je signalizováno nápisem *SPÁROVÁNO*. V opačném případě je tato položka prázdná.



Výběr	Číslo záznamu	Datum	Délka záznamu	Pruměrná rychlost [km/h]	Maximální převýšení [m]	Délka trasy [km]	Komentář záznamu
☐	1	12.3.2010	00:00:05	3,39	43,00	27,259	První záznam změněn
☐	2	15.4.2010	00:00:16	4,86	193,00	55,884	Celkem to sllo...
☐	3	12.3.2010	02:12:03	0,77	494,70	443,386	Prstem po mapě
☐	4	27.4.2010	00:16:24	0,70	24,10	0,11	-----
☐	5	28.4.2010	00:00:02	0,30	0,70	0,006	-----
☐	6	28.4.2010	00:00:20	0,50	1,90	0,028	-----
☐	7	28.4.2010	00:03:51	3,32	3,00	0,555	Chůze po Mikru
☐	8	28.4.2010	00:04:54	0,51	0,90	0,023	-----
☐	9	28.4.2010	00:00:18	0,46	1,00	0,024	-----
☐	10	28.4.2010	00:23:49	3,08	3,00	0,628	-----
☐	11	1.5.2010	00:01:57	31,81	0,30	1,855	Oslavička Hodov auto
☐	12	28.4.2010	08:25:10	9,90	284,60	49,168	Pokus spojení
☐	13	1.5.2010	00:04:00	4,01	22,10	0,811	-----

Obr. 22: Hlavní okno aplikace

6.2 Komunikace se zařízením a VCP

V kapitole popisující obvod FT245BL bylo hovořeno o tvorbě virtuálního sériového portu VCP v PC, po připojení tohoto obvodu prostřednictvím USB sběrnice. Vytvořená aplikace musela byt tedy schopna s tímto portem komunikovat a tím komunikovat se samotným zařízením. Parametry portu VCP je možné libovolně měnit v ovládacích panelech operačního systému, avšak aplikace musí tyto parametry při komunikaci dodržovat.

Pro komunikaci se VCP byla použita komponenta *VaComm*, volně dostupná na internetu v balíčku komponent Varian Async32. [18]. Její výhoda oproti obdobným komponentám pro komunikaci se sériovým portem je možnost průběžného ukládání přijímaných dat do „*bufferu*“. Tato rozšiřující funkce komponenty byla v aplikaci využita.

6.2.1 Popis komponenty VaComm

Komponenta *VaComm* pracuje s několika procedurami a funkcemi uskutečňujícími komunikaci se sériovým portem. Aby komponenta byla schopna komunikovat s vytvořeným VCP portem, je nutné nadefinovat parametry tohoto portu. K tomuto účelu je součástí komponenty *VaComm* skupina parametrů uvedených v tab. 20. Jejich správné nastavení je pro komunikaci zásadní.

Tab. 20: Seznam parametrů komponenty VaComm a jejích popis

Parametry komponenty	**Popis**
PORTNUM	Číslo portu, se kterým komponenta komunikuje
BAUDRATE	Přenosová rychlost
DATABITS	Počet datových bitů
STOPBIT	Počet stop bitů
PARITY	Druh parity

Systémové spojení komponenty a sériového portu je provedené voláním procedury *VaComm.Open* pomocí tlačítka *SPÁROVAT ZAŘÍZENÍ* v menu *PŘENOS*. Pokud není port, definovaný parametrem *PORTNUM*, užíván jinou aplikací, volání procedury proběhne úspěšně. V opačném případě je nutné definovat varovné hlášení, informující o nepodařeném spojení. To je prováděno procedurou *HandleException*, která je volána vždy, když selže procedura *VaComm.Open*. Procedura VaComm se od sériového portu odpojuje automaticky při ukončení aplikace, ale je možné ji odpojit také ručně při běhu aplikace. Odpojení je uskutečněno procedurou *VaComm.Close* volanou stiskem tlačítka *ODPOJIT ZAŘÍEZNÍ* v menu *PŘENOS*.

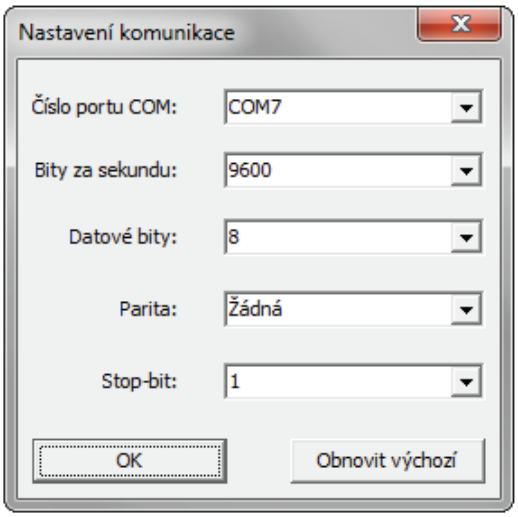
Po úspěšném spojení komponenty je možné zahájit komunikaci. Odesílání dat je realizováno funkcí *VaComm.WriteText* s parametrem typu *AnsiString*. Z popsaného způsobu komunikace je zřejmé, že do zařízení jsou aplikací odesílány povely o délce jediného znaku, tedy parametr typu *AnsiString* má také vždy hodnotu jednoho znaku. Například tedy žádost o zaslání všech dat uložených ve FLASH paměti je provedena voláním funkce ve tvaru *VaComm.WriteText(‘A’)*.

Při přijetí dat je událostí *OnRxChar* komponenty *VaComm* standardně volána příslušná procedura nebo funkce. V ní je pak funkcí *VaComm.ReadChar* možno číst obsah přijatých dat. Událost vrací data uložená v proměnné typu *AnsiString*. Tento způsob příjmu je však možné využít jen do určité délky bezprostředně za sebou odesílaných dat. Pokud jsou data odesílána společně, bez přerušení, je volána událost *OnRxChar* až po ukončení tohoto přenosu. Avšak je li velikost přijímaných dat větší, než je maximální velikost vnitřního „*bufferu*“ komponenty *VaComm*, je tento „*buffer“* přepisován. Velikost tohoto „*bufferu*“ je maximálně 2 kB, což způsobuje ztrátu přijímaných dat, větších než je tato velikost.

Pro tento případ je možné komponentu *VaComm* doplnit komponentou *VaBuffer*. Tímto řešením je odstraněn problém s příjmem objemnějších dat. Komponenta *VaBuffer* může sice využít velikosti pouze 16 kB, což nemusí být opět dostačující, avšak má podstatnou výhodu. Při naplnění jejího „*bufferu*“ je tato událost reflektována voláním události *OnRxBuff* komponenty *VaComm*. Pokud se zajistí přečtení obsahu dat vždy, když je tato událost volána, nedojde ke ztrátě dat, jako tomu bylo v předešlém případě. Tato událost je volána také v okamžiku, kdy dojde k ukončení komunikace, proto není nutné obsah „*bufferu*“ vždy zaplnit. Čtení z „*bufferu*“ komponenty *VaBuffer* je prováděno dvojicí procedur. Nejprve je nutné zjistit, jak velký prostor „*bufferu*“ je využit. K tomuto účelu slouží funkce *VaBuffer.BufUsed*, která vrací velikost v proměnné typu *integer*. Samotné čtení je provedeno voláním funkce *VaBuffer.Read* obsahující parametry, které definují, kde má být obsah uložen a velikost prostoru, který má být z *„bufferu“* vyčten. Pro příklad- voláním funkce *VaBuffer.Read(data,1000)*, bude obsah prvního tisíce hodnot uložených v „*bufferu*“ zkopírován do proměnné s názvem *data*.

6.2.2 Nastavení komponenty VaComm a jeho zálohování

Aby nebylo nutné definovat parametry *VaComm* při každém spuštění aplikace, jsou tyto parametry ukládány do souboru *setting.ini*. Při následném spuštění aplikace je volána procedura *ReadIni*, která nastaví komponentu *VaComm* dle uložených hodnot v tomto souboru. V případě, že je aplikace spouštěna poprvé, nebo dojde li ke smazání souboru *setting.ini*, provede se implicitní nastavení parametrů komponenty *VaComm* a soubor *setting.ini* je znovu vytvořen. Změnu nastavení je možné provést v okně *Nastavení komunikace*, které je aktivováno stiskem tlačítka *KOMUNIKACE* v menu *NASTAVENÍ*. Toto okno je zobrazeno na obr. 23.



Obr. 23: Okno pro nastavení parametrů komponenty *VaComm*

Po nastavení příslušných parametrů a stisku tlačítka *OK*, jsou tyto parametry uloženy v souboru *setting.ini*. Tlačítko *OBNOVIT VÝCHOZÍ* nastavuje implicitní hodnoty používané virtuálním sériovým portem pro číslo portu COM 10.

6.2.3 Způsob příjmu dat ze zařízení

Při volání procedury *VaBuffer.Read* je nejprve načtena velikost obsazení „*bufferu*“. Následně je rozhodnuto pomocí globální proměnné *CONNECT* typu *BOOLENA*, zda se jedná o přenos obsahující data nebo informace. Nabývá li proměnná hodnoty FALSE, jde o informace a obsah „*bufferu*“, je zkopírován do proměnné typu pole bytů s názvem *GPS_POC*. K hodnotám uloženým v tomto typu proměnné je možné přistupovat hromadně nebo po jednotlivých bytech. Nejprve je nutné zjistit, co informace obsahují. K tomu složí úvodní znak přenosu. V případě, že je jeho hodnota rovna 1, následujících šest bytů uložených v proměnné *GPS_POC* jsou ukazatelé záznamu a je z nich okamžitě vypočtena velikost uložených dat. Informace o velikosti jsou uživateli sděleny dialogovým oknem, pomocí něhož může být odeslán příslušný povel pro přenos dat z paměti FLASH. Zároveň je proměnná *CONNECT* typu *BOOLEAN* nastavena to hodnoty *TRUE*, jelikož je po odeslání povelu pro přenos dat očekáván jejich příjem. Jedná li se o informační data jiného charakteru, například potvrzení smazání paměti FLASH, zachovává proměnná *CONNECT* svoji hodnotu na FLASE.

Pokud dochází k samotnému příjmu dat, tedy úvodní znak měl hodnotu 4 a proměnná *CONNECT* nabývá hodnoty *TRUE*, je nejprve při volání procedury *VaComm.OnRxBuff* počítána velikost obsazení „*bufferu*“ *VaBuffer*. Je li jeho velikost shodná s velikostí dat, vypočtených z ukazatelů, jsou data z „*bufferu*“ zkopírována opět do proměnné typu pole

bytů, označené *DATASTREAM*. Tato proměnná je dostatečně velká na to, aby byla schopna uložit i celý obsah paměti FLASH.

Dvojí oddělení informačního a datového přenosu pomocí proměnné *CONNECT* a pomocí úvodních znaků přenosu je v aplikaci nutné, jelikož datový přenos většího rozměru může při opakovaném volání události *VaComm.OnRxBuff* náhodně obsahovat na prvním místě, tedy místě úvodního znaku, hodnotu, která by odpovídala například přijímání ukazatelů. V tomto případě by se však jednalo o data, která by ale aplikace špatně vyhodnotila.

6.3 Ukládání a zpracování dat

Po přijetí dat uložených v paměti FLASH je volána funkce *SaveToDB*, která provede správné roztřídění dat v proměnné *DATASTREAM* a uloží je do databázového souboru. Pro ukládání dat byl vybrán relační databázový systém, využívající pro každou databázi jeden soubor typu *CDS*. Po spuštění je v adresáři aplikace hledán soubor *RUNS.CDS*, pokud jej neobsahuje, aplikace tento soubor vytvoří. Databáze, uložená v souboru *RUNS.CDS*, obsahuje deset sloupců pro ukládání příslušných dat, přičemž každý sloupec má definován svůj typ. Jejich seznam je uveden v tab. 21. Pro komunikaci mezi databázovým souborem a aplikací slouží komponenty *ClientDataSet* a *DataSource*. Jejich provázání a spojení se souborem *RUNS.CDS* umožňuje provádět v databázi čtení, zápis, mazání a filtraci dat pomocí příslušných procedur a funkcí.

Tab. 21: Seznam a popis sloupců databázového souboru

Název sloupce	Typ	Popis
id	INTEGER	Identifikační číslo záznamu
id_beh	INTEGER	Identifikační číslo společné skupiny záznamů
datum	DATE	Datum záznamu
cas	TIME	Čas záznamu
z_sirka	FLOAT	Zeměpisná šířka
z_delka	FLOAT	Zeměpisná délka
n_vyska	FLOAT	Nadmořská výška
rychlost	FLOAT	Rychlost pohybu
Tep	INTEGER	Vyhrazeno pro možnost uložení tepové frekvence
komentar	STRING	Komentář ke skupině záznamů sdružené hodnotou id_beh

Jelikož jsou data ze zařízení od PC přenášena tak, jak jsou uložena v paměti FLASH, je nutné je po přijetí znovu rekonstruovat do podoby, v jaké jsou čtena z GPS modulu. To je činností funkce *SaveToDB*, jejímž výstupem je cyklické zpracování 23bytových skupin dat do řetězců, které je možno jako jediný řádek uložit do databáze. Při rekonstrukci jsou údaje zeměpisné šířky a délky vyjádřeny pouze ve stupních s přesností na sedm desetinných míst. Údaj o rychlosti pohybu je přepočten z uzlů na jednotku kilometry za hodinu vynásobením konstantou 1,852.

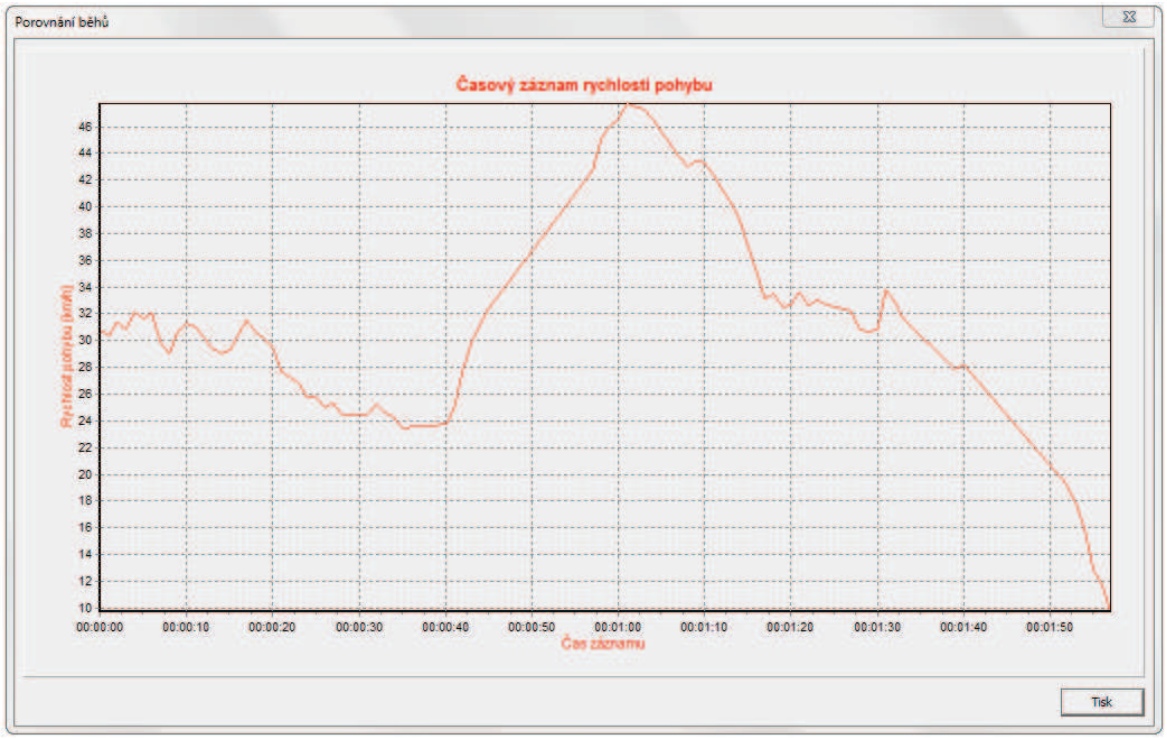
K uložení jednoho rekonstruovaného záznamu do databáze je použita procedura *ClientDataSet.InsertRecord*. Při ukládání každého záznamu je vždy inkrementována hodnota ID, tak aby bylo možné záznamy pomocí této hodnoty řadit. Hodnota *ID_BEH* je vytvářena součtem nejvyšší hodnoty *ID_BEH* uložené v databázi s hodnotou *IT* (identifikační číslo trasy) uloženou u daného záznamu v paměti FLASH. Je li tedy databáze prázdná, odpovídá ukládaná hodnota *ID_BEH* hodnotám *IT* připojeným k záznamu v zařízení. Obsahuje li však již databáze nějaké starší záznamy, jsou hodnoty *ID_BEH* nových záznamů navyšovány o hodnoty *ID_BEH* záznamů již uložených. To zajišťuje unikátnost hodnoty *ID_BEH* pro každou skupinu záznamů bez ohledu na to, proběhne li smazání obsahu paměti FLASH, což mimo jiné také způsobí nulování hodnoty *IT*.

6.4 Graf rychlosti pohybu

Jedním z možných způsobů vyhodnocení zaznamenané trasy je zobrazení rychlosti pohybu v závislosti na čase. K tomuto účelu je použita komponenta *TChart*, která je schopna vykreslovat skupiny bodů definovaných dvojicí hodnot *X Y* do grafu zvolených parametrů. K tvorbě křivky grafu byla vytvořena procedura *DrawGraph* s definovanými vstupními proměnnými *ClientDataSet1* a *SelectedRuns*. Proměnná *ClientDataSet1* definuje, z jaké databáze budou data pro tvorbu grafu použita. Proměnná *SelectRuns* pak stanovuje, který záznam byl zvolen pro vykreslení.

Samotná procedura dle těchto zadaných vstupních proměnných vytvoří série bodů *X Y*, kde hodnotu *X* tvoří čas záznamu a hodnotu *Y* rychlost v daném čase. Aby však bylo možné porovnat jednotlivé zaznamenané trasy v jednom grafu, není údaj na ose *X* udáván v absolutních hodnotách, ale vždy vychází z hodnoty nula. To umožní srovnat dvě zaznamenané trasy, aniž by čas jejich záznamů ovlivnil posunutí na ose *X*. Záznam rychlosti pohybu, pořízený při jízdě automobilem je zobrazen na obr. 24.

V dolní části okna grafu je tlačítko umožňující tisk vykresleného průběhu. Tato funkce je součástí komponenty *TChart* a po stisku tlačítka je volána procedurou *TeePrevi.TeePreview*.



Obr. 24: Okno zobrazující časovou závislost rychlosti pohybu

6.5 Zobrazení mapy prostřednictvím serveru GOOGLE MAPS

Jelikož by tvorba nové komponenty vizualizace trasy na mapě nebyla v prostředí DELPHI jednoduchá a většina již hotových komponent je placena, bylo vybráno řešení využívající v aplikaci webové API GOOGLE MAPS.

Základem vizualizace trasy je komponenta webového prohlížeče *WEBBROWSER*, která se zobrazí po stisku tlačítka *MAPA* v menu *ZPRACOVÁNÍ*. Z databáze jsou následně získány souřadnice jednotlivých bodu zvolené trasy. Všechny body jsou odeslány pomoci metody POST protokolu HTTP, proto je nutné data zakódovat a poslat hlavičku „*application/x-www-form-urlencoded*“, jak je vidět v proceduře *GETMAP* unity UNIT3.PAS, jež je součástí elektronické přílohy. Odeslaný požadavek zpracovává webový server APACHE s běžícím modulem PHP, který má na starosti vlastní zpracování skriptu. Serverový skript pouze kontroluje, zdali POST data byla přijata a následně všechny body spojuje do řetězce, které pak na klientovi představuje JAVASCRIPT objekty jednotlivých bodu cesty. Po zpracování je cela stránka odeslána na klienta a zbytek zpracování a

vykreslení mapy je již prováděno komponentou webového prohlížeče *WEBBROWSER*. Jak je vidět v kódu souboru *INDEX.PHP*, který je součástí elektronické přílohy, v hlavičce vrácené HTML stránky jsou definovány dva JAVASCRIPTY. První načítá celé JAVASCRIPT API ze serveru CODE.GOOGLE.COM. Součástí nějž je také odeslání unikátního klíče pro doménu použitého webserveru, čímž je prováděna kontrola, zda je URL tohoto webserveru zaregistrováno. Tato registrace je na serveru GOOGLE MAPS bezplatná a je díky ní umožněno mapy využívat.

Další JAVASCRIPT je již vlastní využití GOOGLE MAPS API. Tedy je zde definována funkce *INITIALIZE*, která provádí vlastní kreslení mapy, při každém načtení této web stránky (při každém dotazu z komponenty webového prohlížeče *WEBBROWSER*). Nejprve je proveden test kompatibility webového prohlížeče. Následně je vytvořen vlastní JAVACRIPT objekt mapy (v konstruktoru je mu předán element, v němž má mapu vykreslit), kterému je nadefinováno standardní grafické ovládání (šipky, ovládání měřítka a volba typu mapy). Dále je vytvořen a do mapy vložen objekt představující spojnice bodu vykreslované trasy. Tyto body jsou také využity pro nalezení vhodného měřítka mapy a k jejímu vycentrování [20].

Obr. 25 zobrazuje vykreslenou trasu chůze na dvoře areálu Udolní 53, FEKT VUT v Brně.

Obr. 25: Zobrazení zaznamenané trasy pohybu na dvoře areálu Mikroelektroniky FEKT VUT v Brně

7 Závěr

Tato diplomová práce se zabývala návrhem a realizací GPS přístroje schopného kontinuálně zaznamenávat navigační údaje do vlastní paměti. Dále pak tvorbou aplikace v PC, umožňující tyto navigační údaje zobrazovat jako trasu pohybu na mapě.

Pro řízení tohoto zařízení byl zvolen obvod FPGA Spartan – 3A do nějž byl rozhraním JTAG implementován virtuální procesor PICOBLAZE, který celé zařízení ovládá. Program tohoto procesoru byl vytvořen v jazyce Assembler. Aby bylo zařízení mobilní, byla pro napájení zvolena dvojice akumulátorů Ni-MH, které ve spojení s napěťovými měniči zajišťují pracovní napětí všech obvodů. Komunikace mezi zařízením a aplikací v PC je realizována obvodem FTDI prostřednictvím USB rozhraní. Pro příjem dat ze systému GPS byl vybrán modul LEA – 5s, který splňuje požadavky na nízkou spotřebu a malé rozměry. Pro svou činnost vyžadoval pouze připojení pasivní antény definovaných parametrů. K uchování dat v zařízení byla použita sériová FLASH paměť s kapacitou 16 Mbitů, která je schopna komunikovat prostřednictvím SPI rozhraní s rychlostí až 50 MHz.

Při vývoji programu pro procesor PICOBLAZE a design v jazyce VHDL byla využita vývojová deska, osazená obvodem FPGA. K ní byl připojen pouze GPS modul LEA – 5s, který poskytoval potřebná navigační data. Až po vyvinutí a otestování realizovaného programu na vývojové desce, bylo vytvořeno finální zapojení. Dle navrženého schématu byla následně vytvořena trojice desek plošných spojů, které byly ručně osazeny.

Současně s vývojem programu procesoru a návrhem desek plošných spojů byla vytvořena také aplikace pro PC. Ta byla realizována jazykem PASCAL v programu DELPHI 2009. Po přenesení dat z paměti zařízení do PC, je schopna třídit, archivovat a porovnávat záznamy ve vlastním databázovém systému. Dále umožňuje zobrazení zaznamenané trasy pomocí volně dostupných družicových snímků ze serveru GOOGLE MAPS.

Výsledkem této práce je funkční mobilní GPS zařízení splňující požadavky zadání v plném rozsahu. Při testování však byly odhaleny drobné problémy především se silou signálu pro GPS modul, což se projevovalo snížením přesnosti údajů o poloze. Z tohoto důvodu by bylo vhodné v případě budoucího rozšiřování zařízení, doplnit GPS modul vhodnou aktivní anténou nebo doplnit stávající pasivní anténu zesilovačem.

8 Použitá literatura

[1] FRENCH, Gegory. Understanding the GPS. EU : GeoResearch, 1996. 256 s. ISBN 0-9655723-0-7.

[2] LEA-5, NEO-5, TIM-5H u-blox 5 GPS Modules : Hardware Integration Manual [online]: U-blox, 2009 [cit. 2010-05-19]. Dostupné z WWW: <http://www.ublox.com/images/downloads/Product_Docs/LEA-5_NEO-5_TIM-5H_HardwareIntegrationManual(GPS.G5-MS5-09027).pdf>.

[3] ASCII. In Wikipedia : the free encyclopedia [online]. St. Petersburg (Florida) : Wikipedia Foundation, , last modified on [cit. 2010-05-19]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/ASCII>.

[4] PicoBlaze 8-bit Embedded Microcontroller : User Guide [online]. : Xilinx, Inc, 2010 [cit. 2010-05-19]. Dostupné z WWW: <http://www.xilinx.com/support/documentation/ip_documentation/ug129.pdf>.

[5] Xilinx, Inc : PicoBlaze User Resources [online]. 2010 [cit. 2010-05-19]. Dostupné z WWW: <http://www.xilinx.com/ipcenter/processor_central/picoblaze/picoblaze_user_resources.htm#articles>.

[6] Mediatronix : pBlazIDE [online]. 2005 [cit. 2010-05-19]. Dostupné z WWW: <http://www.mediatronix.com/pBlazeIDE.htm>.

[7] AppCAD [online]. 2005 [cit. 2010-05-19]. Dostupné z WWW: <http://www.hp.woodshot.com/>.

[8] HIGH-EFFICIENCY, 1-CELL AND 2-CELL BOOST CONVERTERS [online]: Texas Instruments, 2005 [cit. 2010-05-19]. Dostupné z WWW: <http://focus.ti.com/lit/ds/symlink/tps61016.pdf>.

[9] 1A Low-Dropout Regulator with Reverse Current Protection [online]: Texas Instruments, 2009 [cit. 2010-05-19]. Dostupné z WWW: <http://focus.ti.com/lit/ds/symlink/tps73725.pdf>.

[10] Spartan-3 Generation FPGA User Guide : Extended Spartan-3A, Spartan-3E and Spartan-3 FPGA Families [online]: Xilinx, Inc, 2009 [cit. 2010-05-19]. Dostupné z WWW: <http://www.xilinx.com/support/documentation/user_guides/ug331.pdf>.

[11] Spartan-3 Generation Configuration User Guide : Extended Spartan-3A, Spartan-3E and Spartan-3 FPGA Families [online]: Xilinx, Inc, 2009 [cit. 2010-05-19]. Dostupné z WWW: <http://www.xilinx.com/support/documentation/user_guides/ug332.pdf>.

[12] Platform Flash In-System Programmable Configuration PROMs [online]: Xilinx, Inc, 2009 [cit. 2010-05-19]. Dostupné z WWW: <http://www.xilinx.com/support/documentation/data_sheets/ds123.pdf>.

[13] CFPS-72, -73 SMD CLOCK OSCILLATORS [online]: IQD Frequency Products, 2008 [cit. 2010-05-19]. Dostupné z WWW: <http://www.newark.com/pdfs/datasheets/IQD/CFPS-72_73.pdf>.

[14] M25P16 : 16 Mbit, Low Voltage, Serial Flash Memory With 50 MHz SPI Bus Interface [online]: ST Micorelectronic, 2004 [cit. 2010-05-19]. Dostupné z WWW: <http://www.st.com/stonline/press/news/year2004/p1398m.htm>.

[15] FT245BL USB FIFO [online]: FTDI chip, 2005 [cit. 2010-05-21]. Dostupné z WWW: <http://www.ftdichip.com/Documents/DataSheets/DS_FT245BL.pdf>.

[16] FUJCIK, Lukáš. Ústav mikroelektroniky [online]. 2005 [cit. 2010-05-22]. Návrh digitálních integrovaných obvodů VLSI a jazyk VHDL. Dostupné z WWW: <http://www.umel.feec.vutbr.cz/BNDI/>.

[17] Download Free Trial [online]. 2009 [cit. 2010-05-19]. Embarcadero Technologies, Inc. Dostupné z WWW: <https://downloads.embarcadero.com/free/delphi>.

[18] Sériové rozhraní [online]. 2001 [cit. 2010-05-19]. Builder. Dostupné z WWW: <http://www.builder.cz/data/ASYNC32.ZIP>.

[19] Ortodroma. In Wikipedia : the free encyclopedia [online]. St. Petersburg (Florida) : Wikipedia Foundation, [cit. 2010-05-19]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Ortodroma>.

[20] Google Map API : Map Basic [online]. 2010 [cit. 2010-05-23]. Google CODE. Dostupné z WWW: <http://code.google.com/intl/en/apis/maps/documentation/javascript/v2/introduction.html>.

9 Seznam příloh

Příloha A: Vývojový diagram programu procesoru PICOBLAZE

Příloha B: Celkové schéma zapojení

Příloha C: Desky plošných spojů a osazovací schéma

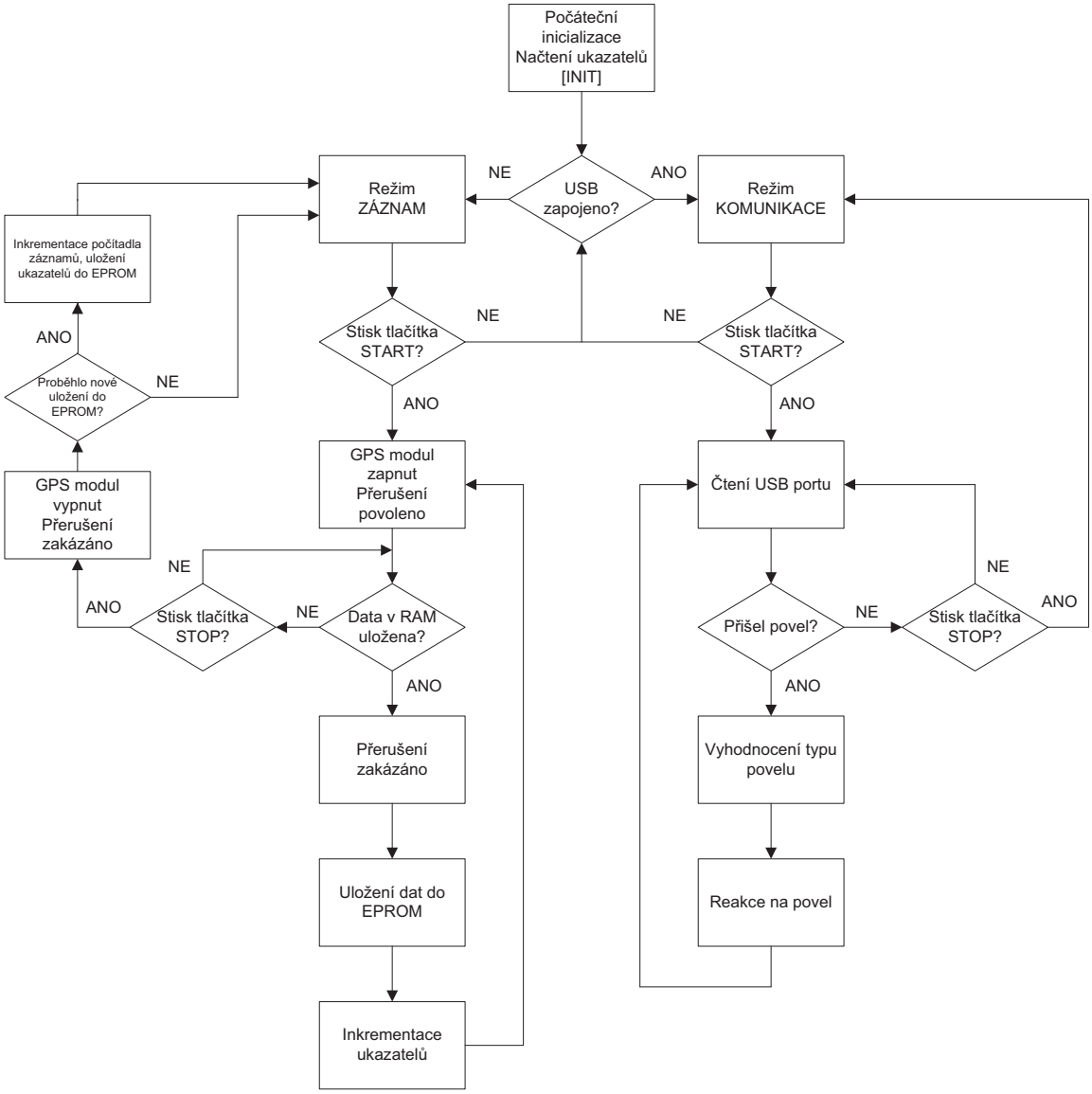
Příloha D: Fotografie osazených desek plošných spojů

Příloha E: Fotografie zařízení

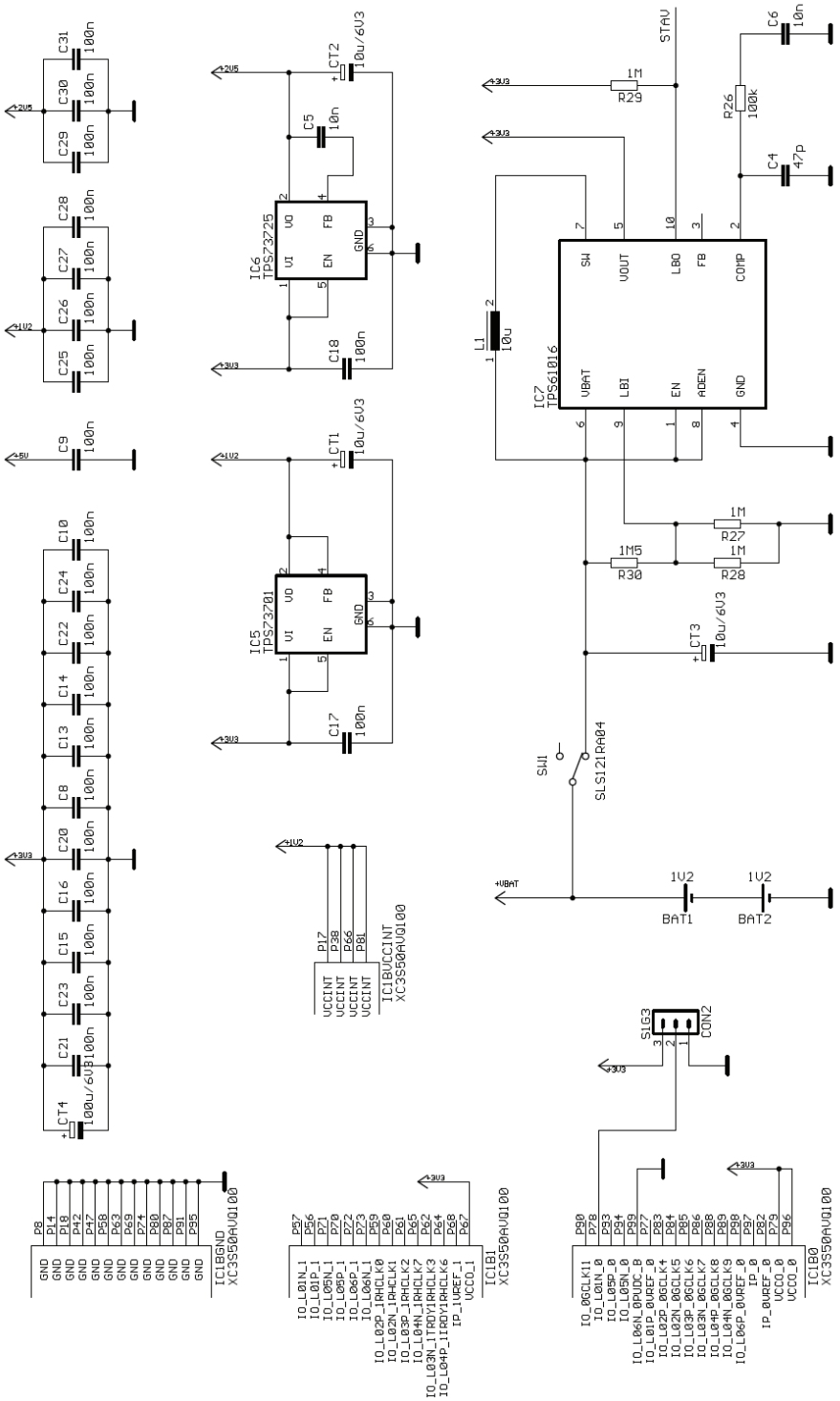
Příloha F: Zaznamenané trasy

10 Přílohy

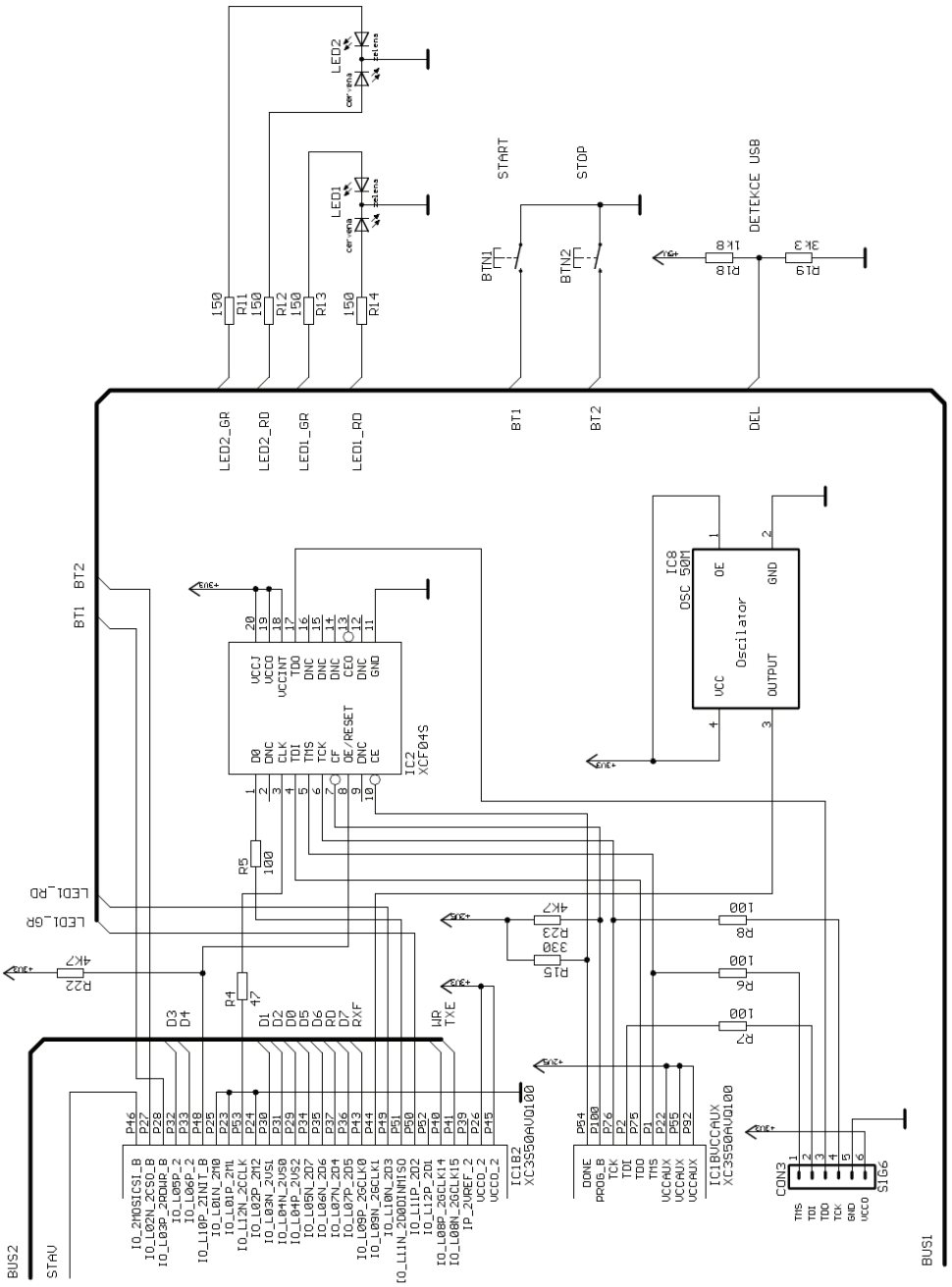
Příloha A: Vývojový diagram programu procesoru PICOBLAZE



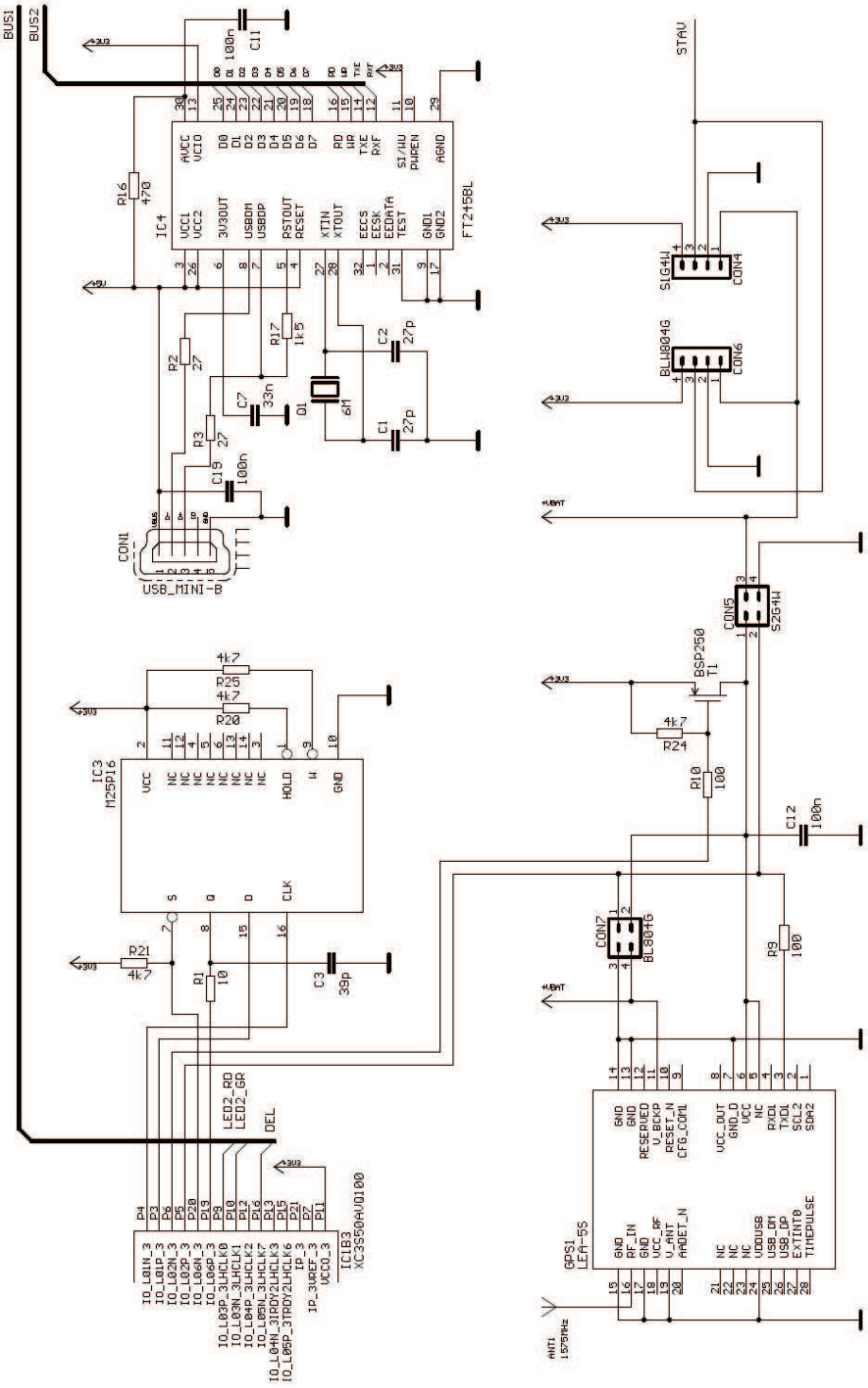
Příloha B: Celkové schéma zapojení



Příloha B 1: Schéma napájecí části zařízení

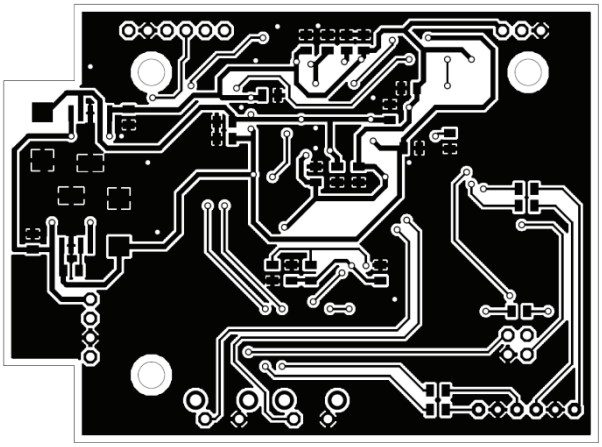


Příloha B 2: Schéma ovládací, programovací části zařízení a oscilátoru

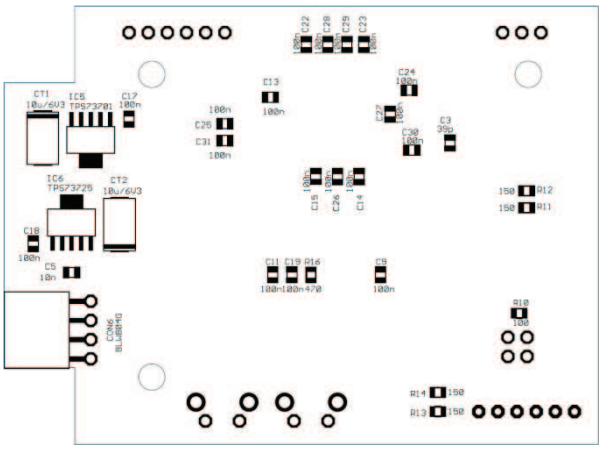


Příloha B 3: Schéma komunikační části, GPS modulu a paměti FLASH

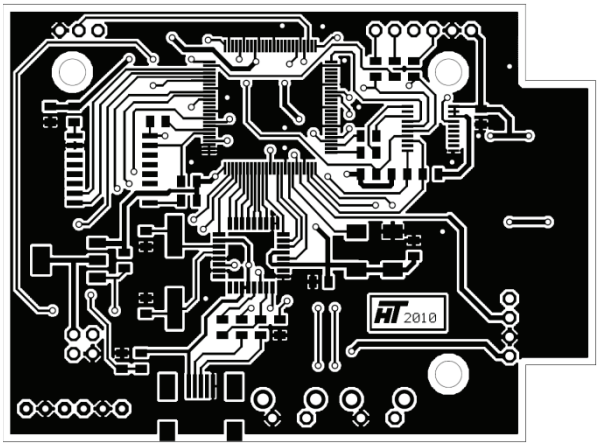
Příloha C: Desky plošných spojů a osazovací schéma



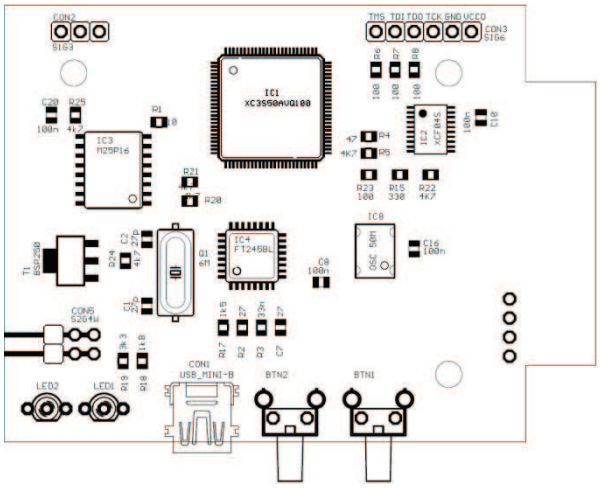
Příloha C 1: Základní deska, spodní strana, zrcadlový pohled, měřítko 1:1



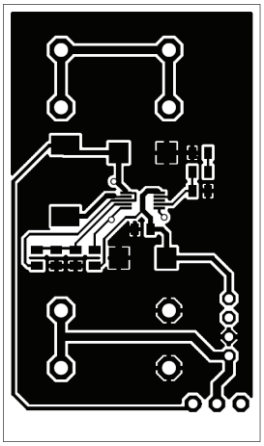
Příloha C 2: Osazení základní desky, spodní strana, zrcadlový pohled, měřítko 1:1



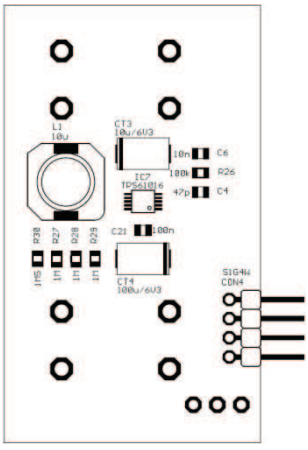
Příloha C 3: Základní deska, horní strana, přímý pohled, měřítko 1:1



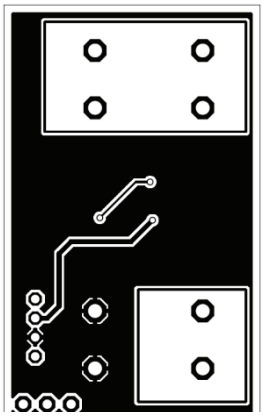
Příloha C 4: Osazení základní desky, horní strana, přímý pohled, měřítko 1:1



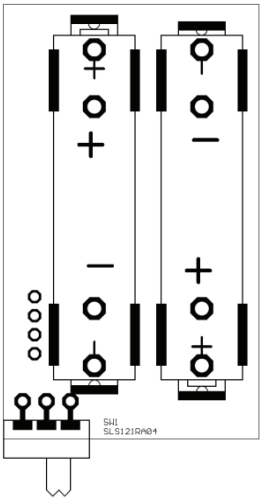
Příloha C 5: Napájecí deska, spodní strana, zrcadlový pohled, měřítko 1:1



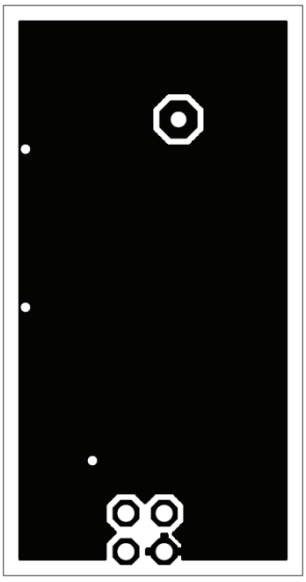
Příloha C 6: Osazení napájecí desky, spodní strana, zrcadlový pohled, měřítko 1:1



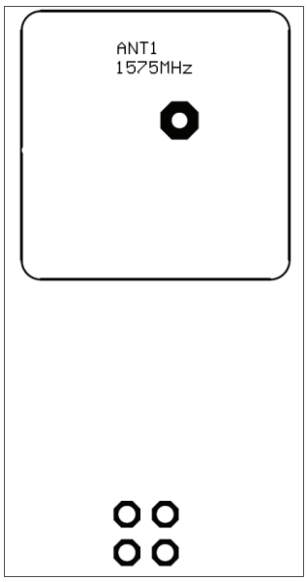
Příloha C 7: Napájecí deska, horní strana, přímý pohled, měřítko 1:1



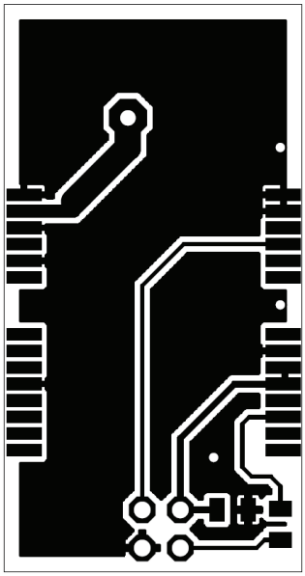
Příloha C 8: Osazení napájecí desky, horní strana, přímý pohled, měřítko 1:1



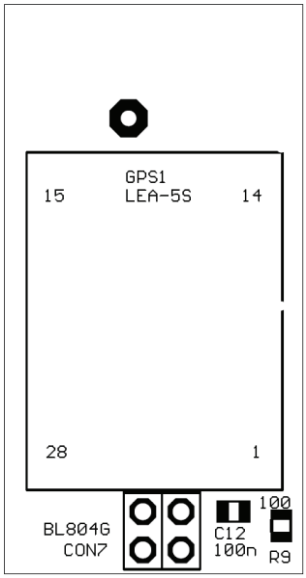
Příloha C 9: Deska GPS modulu, spodní strana, zrcadlový pohled, měřítko 2:1



Příloha C 10: Osazení desky GPS modulu, spodní strana, zrcadlový pohled, měřítko 2:1

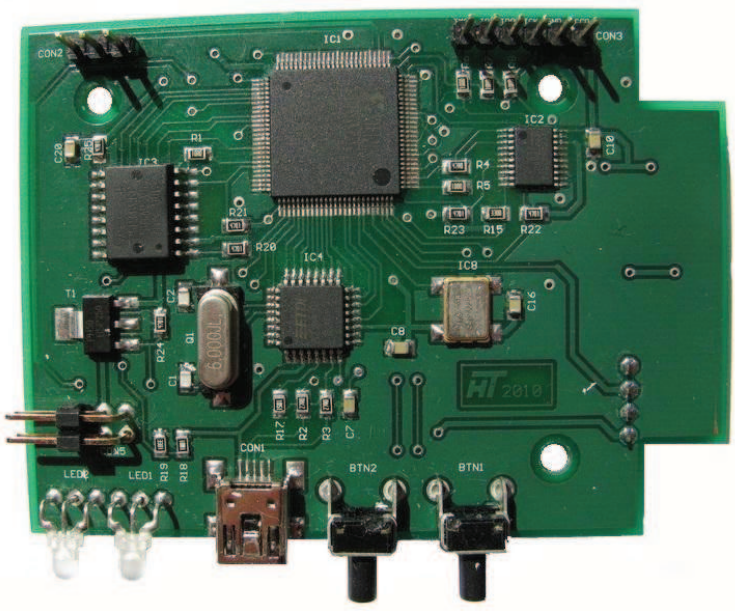


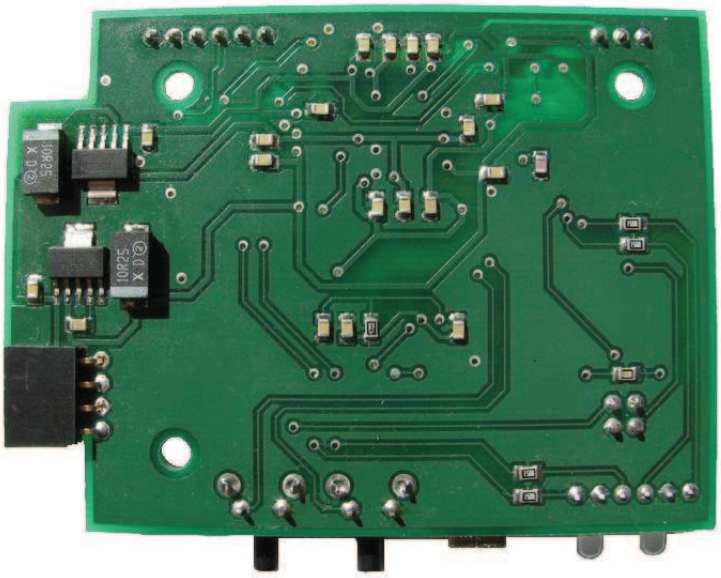
Příloha C 11: Deska GPS modulu, horní strana, přímý pohled, měřítko 2:1

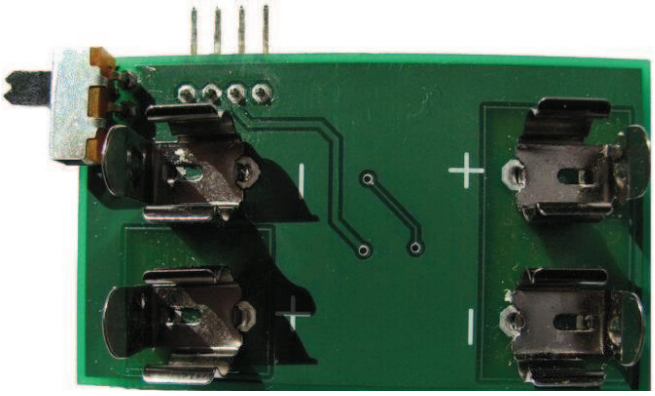


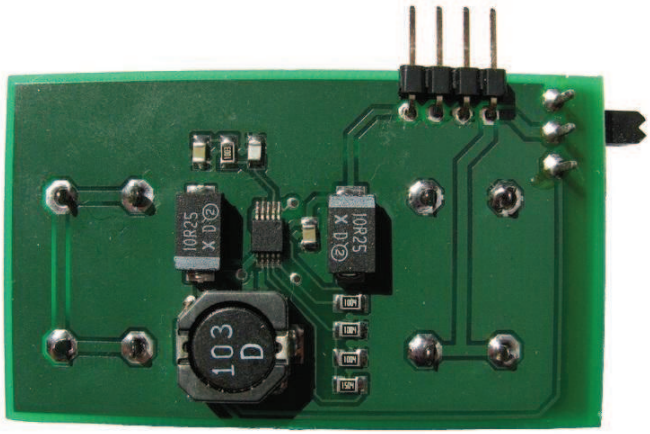
Příloha C 12: Osazení desky GPS modulu, horní strana, přímý pohled, měřítko 2:1

Příloha D: Fotografie osazených desek plošných spojů



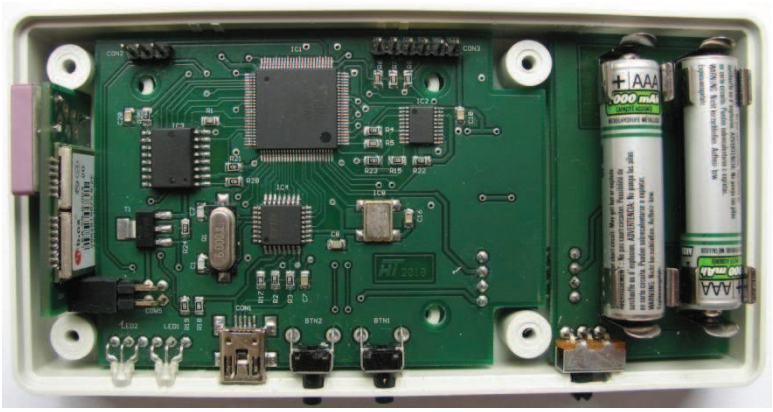





10R25
103
D


ublox
LEA-5S-0-003 20
9882619675 8/26
150100.0700.000

Příloha E: Fotografie zařízení



Příloha F: Zaznamenané trasy