VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

DETEKCE DRONU V PROSTORU

DETECTION OF A DRONE IN SPACE

SEMESTRÁLNÍ PROJEKT
TERM PROJECT

AUTOR PRÁCE **Bc. ŠTĚPÁN RYDLO**
AUTHOR

VEDOUCÍ PRÁCE **prof. Ing., Dipl.-Ing. MARTIN DRAHANSKÝ, Ph.D.**
SUPERVISOR

BRNO 2019

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav inteligentních systémů (UITS) Akademický rok 2018/2019

Zadání diplomové práce

Student: **Rydlo Štěpán, Bc.**
Program: Informační technologie Obor: Inteligentní systémy
Název: **Detekce dronu v prostoru**
Detection of a Drone in Space
Kategorie: Zpracování obrazu

Zadání:

1. Seznamte se s možnostmi lokalizace objektů v prostoru, způsobu radiové komunikace pro lokalizaci a RTL-SDR zařízeními.
2. Popište možnosti detekce dronu a navrhněte vlastní lokalizační systém pro určení polohy dronu.
3. Implementujte navržený lokalizační systém.
4. Určete přesnost vytvořeného lokalizačního systému.
5. Zhodnoťte svojí práci a její přínos, diskutujte možnosti rozšíření.

Literatura:

- Kai B., Denis M.A. *A Software-Defined GPS and Gallieo Receiver*, 2007, ISBN 978-0-8176-4540-3.
- Stewart R.W. et al. *Software defined radio using MATLAB & Simulink and the RTL-SDR*. Strathclyde Academic Media, 2015.
- Tuttlebee W.H.W. (ed.). *Software defined radio: enabling technologies*. John Wiley & Sons, 2003.

Při obhajobě semestrální části projektu je požadováno:

- Body 1 a 2

Podrobné závazné pokyny pro vypracování práce viz http://www.fit.vutbr.cz/info/szz/

Vedoucí práce: **Drahanský Martin, prof. Ing., Dipl.-Ing., Ph.D.**
Vedoucí ústavu: Hanáček Petr, doc. Dr. Ing.
Datum zadání: 1. listopadu 2018
Datum odevzdání: 22. května 2019
Datum schválení: 1. listopadu 2018

Abstrakt

Tato práce se zabývá vytvořením lokalizačního systému za použití softwarově definovaného rádia. Cílem práce je vytvořit vlastní lokalizační systém, který bude nezávislý na již existujících systémech. Pro vytvoření lokalizačního systému je použito zařízení ADALM-PLUTO, které umožňuje vysílat a přijímat radiový signál. Obsahem práce je popis několika možností, jak lokalizační systém vytvořit a popis způsobu jejich komunikace.

Abstract

This master thesis purposes is create localization system using software defined radio. The purpose of this thesis is to create new localization system, which will be independent of existing systems. To create a localization system, we will use ADALM-PLUTO device to send and receive radio signals. This work contains a decription of serval possibilities how to create the localization system and description of their comunication.

Klíčová slova

lokalizační systém, TDOA, ADALM-PLUTO, modulace, zlaté sekvence, multikorelátor

Keywords

localization system, TDOA, ADALM-PLUTO, modulation, gold code, multicorrelator

Citace

RYDLO, Štěpán. *Detekce dronu v prostoru*. Brno, 2019. Semestrální projekt. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce prof. Ing., Dipl.-Ing. Martin Drahanský, Ph.D.

Detekce dronu v prostoru

Prohlášení

Prohlašuji, že jsem tuto semestrální práci vypracoval samostatně pod vedením pana prof. Ing. Dipl.-Ing. Martina Drahanského, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Štěpán Rydlo
16. května 2019

Poděkování

Tímto bych rád poděkoval panu prof. Ing. Dipl.-Ing. Martinu Drahanskému, Ph.D. za vedení semestrálního projektu.

Obsah

Seznam obrázků

Kapitola 1

Úvod

V dnešní době existuje řada systémů pro lokalizaci. Pomocí lokalizace určujeme polohu objektu v určitém prostoru. Lokalizační systémy jsou v základě děleny podle dostupnosti na lokální nebo globální. Globální lokalizační systémy už podle názvu mají větší dostupnost. Mezi globální lokalizační systémy patří satelitní navigační systémy označované jako GNSS (Global Navigation Satellite System). Mezi tyto navigační systémy patří například GPS, GLONASS a v budoucnu i Galileo.

Globální lokalizační systémy jsou dostupné po celém světě. Avšak v případě použití těchto systémů jsme plně závislí na provozovateli vysílačů, které nám dodávají lokalizační data. Dalším problémem je přesnost a dostupnost systémů v uzavřeném prostoru (hala, dům, ...). Z důvodu závislosti na provozovatelích GNSS nebo nedostupnosti vznikají lokální lokalizační systémy, které jsou nezávislé a dostupné na místech, kde je potřebujeme.

Cílem práce je navrhnout a popsat vlastní lokalizační systém, který nám umožní vypočítat polohu objektu. V prvních částech práce se podívám na možnosti, jakými je možné určit polohu objektu v prostoru. V další částí se pak zaměřím na metodu TDOA pro určení polohy. Zaměřím se na to jakým způsobem se vypočítává poloha objektu, ale také stanovit jakých vlastností lokalizační systém musí dosahovat. Jednou z takovýchto vlastností je, aby všechny vysílače vysílali ve stejný čas, proto je třeba dosáhnout mezi nimi určité synchronizace. Na závěr první části se pak zaměřím na způsob komunikace mezi zařízeními a jakým způsobem tuto komunikaci lze zajistit.

V následující kapitole definuji vlastní lokalizační systém, který nebude nijak závislí na již existujících. Součástí lokalizačního systému musí být definován způsob komunikace. Jako způsob komunikace byla vybrána radiová komunikace za použití zlaté sekvence, která bude tvořit lokalizační data. Zlatá sekvence bude vysílána pomoci digitální modulace BPSK. Na základě této sekvence se potom budou synchronizovat všechna zařízení v lokalizačním systému a možnost lokalizace, tedy zda bude objekt určovat vlastní polohu nebo systém bude určovat polohu objektu. Obě tyto možnosti mohou být použity zároveň.

V čtvrté, předposlední části práce, se podívám na samotnou implementaci lokalizačního systému. Pro vysílání a přijímání signálu použiji zařízení ADALM-PLUTO, které umožňuje přijímat a vysílat signál zároveň. Zpracování signálu se bude provádět pomoci programu, metodou softwarově definovaného rádia. V první části samotné implementace se zaměřím na možnosti ovládání zařízení a také v jakém formátu vzorků je definovaný samotný signál. Následně se zaměřím na implementací aplikace pro vysílání dat, která musí provádět digitální modulaci signálu a vysílat dokola jednu stanovenu sekvenci. Druhou z aplikací bude přijímač, který musí zpracovávat přijímaná data a na jejich základě určit časový rozdíl mezi

jednotlivými sekvencemi. V poslední části kapitoly se pak zaměřím na přeložení a spuštění aplikace na samotném zařízení.

Na závěr práce se zaměřím na stanovení přesnosti lokalizačního systému. V této části určím přesnost vytvořené implementace. Přesnost lokalizačního systému lze určit teoreticky nebo pomoci testování v prostoru. Při testování přesnosti v prostoru se dají použít další metody pro zlepšení, jako je například multilaterace. Při teoretickém stanovení přesnosti musím přesnost vypočítat na základě vzorkovací frekvence zařízení a rychlosti šíření signálu v prostoru.

Kapitola 2

Technika lokalizace objektů

Lokalizace objektů je postup, při kterém vypočítáme polohu objektu anebo polohu pozorovatele v závislosti na určitých orientačních bodech. V historii se jako orientační body používali různé přírodní anebo uměle vytvořené body, ale také magnetické kompasy. Tyto možnosti lokalizace umožňovaly orientaci pouze ve známém prostředí. Prvním z větších pokroků lokalizace bylo použití astrolábu (později sextant) [50], který sloužil pro výpočet polohy pomoci hvězd.

Sextant využívá pro určení polohy metodu triangulace, kde zjišťoval svou polohu na základě úhlu hvězd. Pozorovatel také potřeboval znát přesný čas. Bez znalosti přesného času bylo možné stanovit pouze zeměpisnou šířku [49]. Sextant používá k určení přesného úhlu hvězd zrcátko, které se naklání na stanovené stupnici. Odraz ze zrcátka se odráží do průhledného sklíčka, kde se poloha hvězdy porovnává s horizontem. V případě námořní lokalizace tak horizont tvořil úhel nula stupňů tedy vodorovnou plochu. Po zjištění úhlu několika hvězd bylo možné určit přibližnou pozici a směr. Tato metoda však nedosahovala velké přesnosti a musel být dobrý výhled na hvězdy.

Velkého pokroku lokalizaci dosáhla v první polovině 20. století, kdy pro lokalizaci bylo využito radiového signálu. Použití signálu zvyšovalo nejenom přesnost měření, ale i pokrytí určitého lokalizačního systému. Radiové lokalizace používají několik různě umístěných vysílačů. Ze začátku se používaly pozemní vysílače a však v dnešní době pro globální lokalizaci se využívá družic, které obíhají kolem země [38]. Na některé metody výpočtu polohy a možnosti bezdrátové komunikace se podívám v následujících částech.

2.1 Možnosti lokalizace objektu

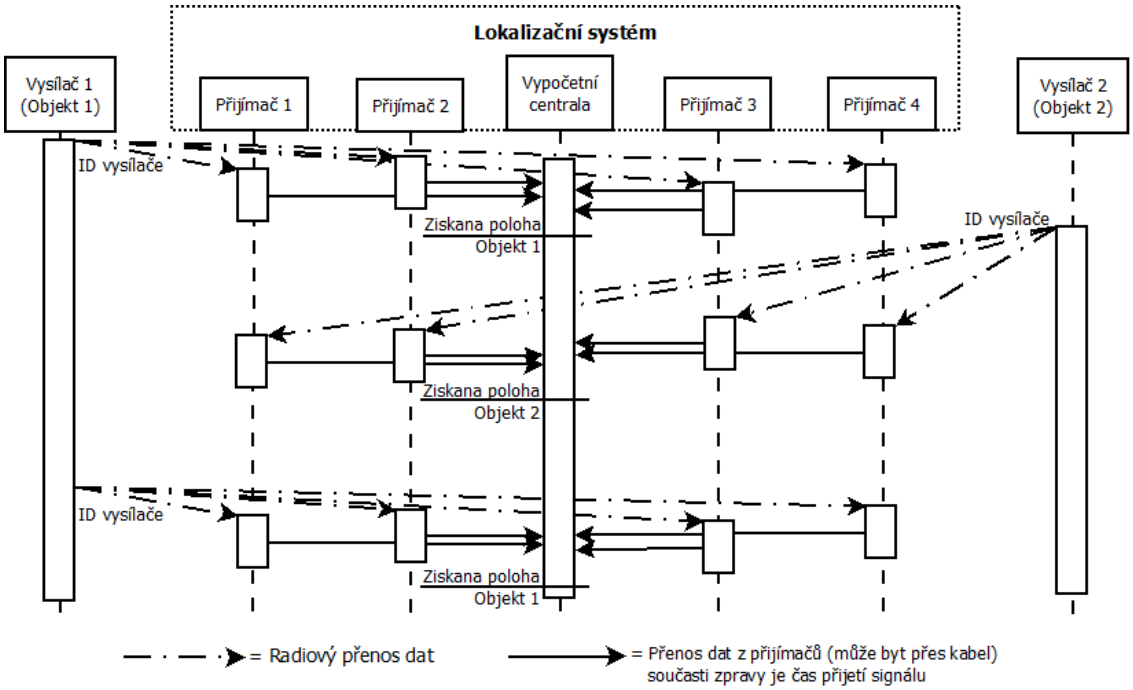
Možností určení polohy objektu pomoci radiových vln existuje mnoho. Jedna z možností rozdělení lokalizačních systémů je na aktivní a pasivní [49]. Aktivní lokalizační systémy používají pro určení polohy vlastních zdrojů elektromagnetického vlnění. Tyto systémy jsou tvořené z vysílačů a přijímačů. Oproti tomu pasivní lokalizační systémy jsou tvořeny pouze přijímači.

2.1.1 Pasivní lokalizační systémy

Pasivní lokalizační systémy jsou systémy, které zjišťují polohu objektů na základě přijímání radiových vln [35]. Do této skupiny patří lokalizační systémy, které se skládají pouze z několika přijímačů. Pasivní lokalizační systémy pouze přijímají signál, ale žádný signál ne-

vysílají. Vysílače přijímají stejný signál a na základě časových rozdílů určují polohu objektu [47].

Signál, který přijímače přijímají je tvořen vysílači, které nejsou součástí lokalizačního systému. Sledovat můžeme objekty, které signál vysílají. Na základě časových rozdílů signálu na jednotlivých přijímačích systému následně můžeme spočítat polohu objektu. Všechny přijímače musí mít stejný a přesný čas, aby bylo možné určit přesnou polohu sledovaného objektu. Musíme však vědět jaký signál a na jaké frekvenci objekt vysílá. Použití této metody umožňuje sledovat objekt a určit jeho polohu. Objekt není součástí lokalizačního systému a proto se jedná o pasivní lokalizační systém. Tyto lokalizační systémy se používají například pro určení polohy letadel [2]. Průběh určení polohy objektu je zobrazen pomoci sekvenčního diagramu na obrázku 2.1. Důležitou vlastností je, že všechny přijímače musí mít přesný čas, na jehož základě určuje přijímač čas přijetí signálu od objektu.



Obrázek 2.1: Sekvenční diagram pasivního lokalizačního systému.

Druhým ze způsobu určení polohy objektu je za pomoci odražených radiových vln. Pozorujeme odraz radiového signálu, který vytváří jiné vysílače. Tyto vysílače nemusí primárně sloužit pro určení polohy a mohou to být vysílače televizního nebo radiového signálu [5]. Určujeme pozici na základě odražených radiových vln, proto sledovaný objekt musí mít větší plochu, od které se tyto vlny odrážejí.

2.1.2 Aktivní lokalizační systémy

Aktivní lokalizační systémy na rozdíl od pasivních se skládají z vysílačů, ale i přijímačů. U těchto systémů existuje více možností, kde může systém určovat polohu objektu, ale i objekt může si určit sám svou polohu na základě informací ze systému. Do této skupiny systémů patří i systémy GNSS, které umožňují provádět oboustranné metody lokalizace. Systém tak může určit polohu objektu, ale i objekt můžeme určit vlastní polohu pomoci tohoto systému.

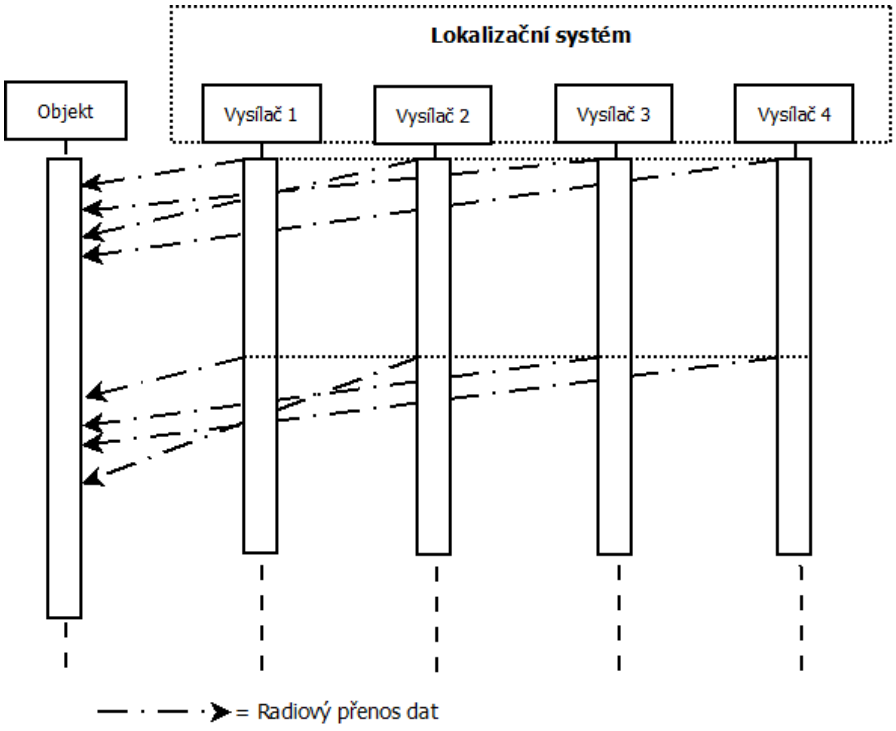
Centralizovaný výpočet polohy

U první z možností, kde systém určuje polohu objektu, musí objekt být vybaven vysílačem a přijímačem. Objekt vyšle signál, podle kterého určí lokalizační systém jeho polohu. Do této chvíle se systém podobá pasivnímu lokalizačnímu systému popsaného pomoci sekvenčního diagramu na obrázku 2.1. Nicméně, objekt chce znát svou polohu, a tak lokalizační systém mu tuto informaci o poloze musí poslat. V tuto chvíli už se lokalizační systém stává aktivním. Lokalizační systém posílá polohu objektu skrz přijímače.

Výhodou tohoto způsobu lokalizace je, že jak objekt, tak i lokalizační systém znají polohu objektu. Nevýhodou však je, že mezi odesláním signálu a přijetím výsledné polohy tvoří časové zpoždění a objekt tak získává svou starší pozici. Druhou z nevýhod je, že jedno zařízení, které je součástí systému, vypočítává polohu pro všechny objekty, jenž žádají o svou polohu. Je zde tak omezená kapacita co se týče počtu objektů, pro které můžeme určit polohu.

Výpočet polohy na základě signálu vysílačů

Druhou možností je, že objekt určí vlastní polohu na základě informací od jednotlivých vysílačů lokalizačního systému. Lokalizační systém tak tvoří vysílače, které znají vlastní polohu a vysílají dokola sekvenci, díky které lze oddělit signál jednotlivých vysílačů od sebe. Pro tuto identifikaci používají vysílače pseudonáhodné sekvence (zlatá, Barkerova,...). Příkladem tak může být systém GPS, kde družice vysílají zlaté pseudonáhodné sekvence (Gold code) [4]. Průběh určení polohy tímto způsobem je zobrazen na obrázku 2.2.



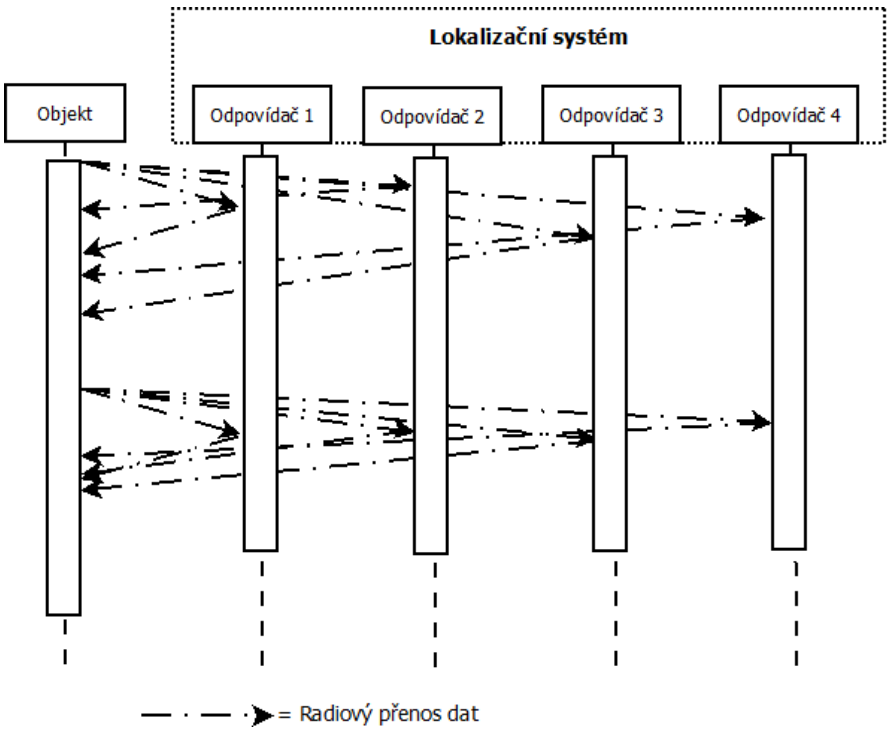
Obrázek 2.2: Sekvenční diagram aktivního lokalizačního systému pomoci vysílačů.

U tohoto způsobu lokalizace však musí být všechny vysílače synchronizované. Musí vysílat počátek své sekvence ve stejnou chvíli jako všechny ostatní vysílače. Výhodou tohoto

lokalizačního systému je, že v systému může být neomezené množství objektů, které vypočítávají vlastní polohu.

Výpočet polohy na základě odpovědi vysílačů

Poslední z možností aktivního lokalizačního systémů, který si popíšeme, je možnost implementace vysílačů, tedy orientačních bodů, jako odpovídače. Jedná se o způsob, kdy objekt zašle signál, který lze označit jako dotaz. Odpovídače mu na tento signál odpoví se stejným zpožděním. Všechny odpovídače proto musí mít stejné zpoždění mezi přijetím signálu a odpovědí. Příklad takového časového rozdílu je vidět na obrázku 2.3.



Obrázek 2.3: Sekvenční diagram aktivního lokalizačního systému pomoci odpovídačů.

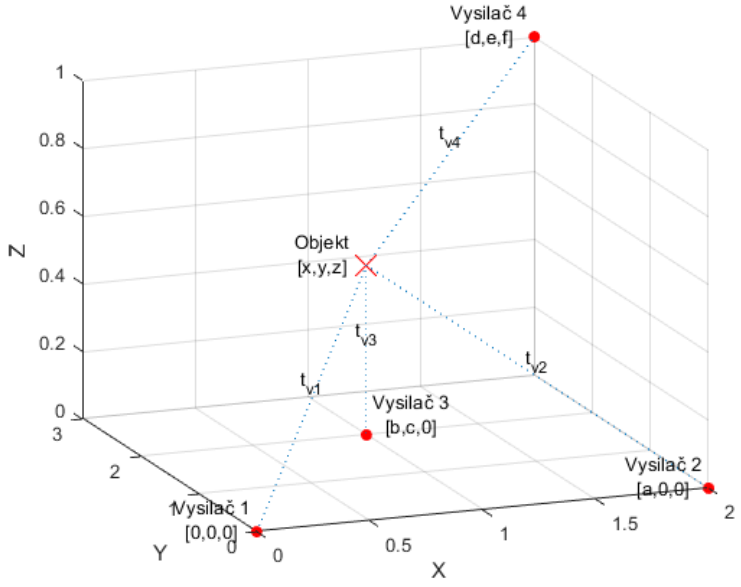
Výhodou této možnosti je, že odpovídače nemusí být nijak synchronizovány. Synchronizaci zajišťuje objekt, který posílá první zprávu. Zpráva se sice dostane ke každému odpovídači v jinou chvíli, ale objekt určuje svou polohu na základě odpovědí. Zde se tedy musí časové zpoždění jednotlivých odpovědí případně upravit, aby se počítalo s cestou signálu tam a zase zpět.

2.2 Metody výpočtu polohy

V dalších částech této kapitoly se podíváme na rovnice a možnosti výpočtu polohy. Ve všech příkladech předchozí kapitoly lze použít metodu TDOA (Time Difference of Arrival) pro výpočet polohy [35] [45]. Dále se také podíváme na přesnost určení polohy, ale i na možnosti, jak přesnost zvýšit a co k jejímu zvýšení potřebujeme.

2.2.1 Výpočet polohy

Výpočet polohy probíhá na základě lokalizačních dat z vysílačů. Poloha jednoho objektu může být definována jinak pro různé lokalizační systémy. Každý z lokalizačních systémů může mít vlastní souřadnicový systém. V této části se podívám obecně na výpočet polohy ve 3D prostoru [37]. Nebude nás tedy zajímat jenom poloha zařízení, ale i jeho výška. Výpočet polohy ve 2D prostoru je popsán v mé bakalářské práci [40]. Jako příklad pro určení výpočtu polohy použijeme rozložení vysílačů a umístění bodu, které je zobrazeno na obrázku 2.4.



Obrázek 2.4: Příklad rozmístění vysílačů pro výpočet polohy ve 3D prostoru.

Pro výpočet polohy ve 3D prostoru použijeme některé z rovnice, které určují hodnotu x a y ve 2D prostoru. V těchto rovnicích se pouze změní označení proměnných L a R pro čas, kdy budeme v tomto případě používat označení t_1 a t_2. Hodnoty těchto proměnných určují časové rozdíly mezi signálem z vysílačů a zůstanou stanoveny stejně, tak jak určují rovnice 2.1.

$$\begin{aligned} t_1 &= t_{v2} - t_{v1} \\ t_2 &= t_{v3} - t_{v1} \\ t_3 &= t_{v4} - t_{v1} \end{aligned} \tag{2.1}$$

Pro výpočet souřadnic x a y zůstanou rovnice stejné jako je tomu ve 2D prostoru. Tedy $x = A + Bk$ a $y = C + Dk$ 2.2, kde jsou pouze přejmenovány proměnné L a R.

$$B = -\frac{L}{a}, \qquad A = \frac{a^2 - {t_1}^2}{2a} = \frac{a + t_1 L}{2} \tag{2.2}$$

$$C = -\frac{c^2 + b^2 - 2Ab - {t_2}^2}{2c}, \qquad D = \frac{-Bb - t_2}{c} \tag{2.3}$$

Poslední rovnicí, kterou je třeba upravit ze 2D prostoru do 3D prostoru je Pythagorova věta 2.4.

$$k = \sqrt{x^2 + y^2 + z^2} \tag{2.4}$$

Následně už vytvořím a vypočítám rovnici pro získání souřadnice v ose z 2.5.

$$\begin{aligned}
t_3 &= \sqrt{(x-d)^2 + (y-e)^2 + (y-f)^2} - \sqrt{x^2 + y^2 + z^2} && \Rightarrow \\
t_3 + k &= \sqrt{(x-d)^2 + (y-e)^2 + (y-f)^2} && \Rightarrow \\
(t_3 + k)^2 &= (x-d)^2 + (y-e)^2 + (y-f)^2 && \Rightarrow \\
{t_3}^2 + 2t_3k + k^2 &= x^2 - 2xd + d^2 + y^2 - 2ye + e^2 + z^2 - 2zf + f^2 && \Rightarrow \\
{t_3}^2 + 2t_3k + k^2 &= k^2 - 2xd + d^2 - 2ye + e^2 - 2zf + f^2 && \Rightarrow \\
{t_3}^2 + 2t_3k &= -2Ad - 2Bdk + d^2 - 2Ce - 2Dek + e^2 - 2zf + f^2 && \Rightarrow \\
z &= \frac{-2Ad - 2Bdk + d^2 - 2Ce - 2Dek + e^2 + f^2 - {t_3}^2 - 2t_3k}{2f}
\end{aligned} \tag{2.5}$$

Jelikož neznáme proměnné k a chceme zjistit z, která určuje polohu objektu je dobré pro další použití upravit rovnici do následující podoby $z = E + Fk$, kde:

$$E = \frac{d^2 + e^2 + f^2 + {t_3}^2 - 2Ad - 2Ce}{2f}, \qquad F = \frac{-t_3 - Bd - De}{f} \tag{2.6}$$

Nyní už máme definované všechny rovnice pro výpočet souřadnic objektu, ve kterých však zůstává jedna neznámá proměnná k. Pro určení této proměnné použijeme Pythagorovu větu 2.4, kde za x, y, z dosadíme námi stanovené rovnice 2.2 a 2.6. Výsledkem tak je kvadratická rovnice 2.7, jejíž výsledek nám určí hodnotu k, kterou následně dosadíme do rovnic pro x, y, z.

$$\begin{aligned}
k^2 &= x^2 + y^2 + z^2 && \Rightarrow \\
k^2 &= (A + Bk)^2 + (C + Dk)^2 + (E + Fk)^2 && \Rightarrow \\
k^2 &= A^2 + 2ABk + B^2k^2 + C^2 + 2CDk + D^2k^2 + E^2 + 2EFk + F^2k^2 && \Rightarrow \\
0 &= A^2 + 2ABk + B^2k^2 + C^2 + 2CDk + D^2k^2 + E^2 + 2EFk + F^2k^2 - k^2 && \Rightarrow \\
0 &= k^2(B^2 + D^2 + F^2 - 1) + k(2AB + 2CD + 2EF) + (A^2 + C^2 + E^2)
\end{aligned} \tag{2.7}$$

2.2.2 Rozložení orientačních bodů

Jako orientační body označujeme vysílače nebo odpovídače, které znají svou polohu a nemění ji. Všechny vysílače by neměly být umístěny v jedné rovině, aby bylo možné zjistit přesnou v pozici. V případě umístnění všech vysílačů v jedné rovině jsou výsledkem dvě pozice, které odpovídají časovým zpožděním.

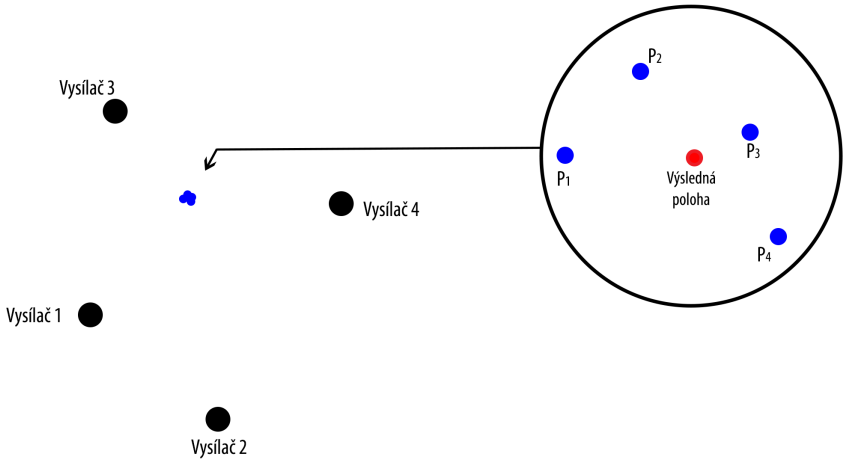
Pro výpočet polohy, podle rovnic definovaných v kapitole 2.2 je třeba, aby se dali vysílače z lokalizačního systému upravit do formy, která je zobrazena na obrázku 2.4. Pro úpravu polohy vysílačů do rozložení, pro které jsou definovány rovnice je třeba provést posun a rotaci daného lokalizačního systému. Operace pro posun a rotace budeme provádět se skupinami 4 vysílačů, které si vybereme z lokalizačního systému.

Jeden z vysílačů musí být umístěn na souřadnicích [0,0,0]. Následně musíme provést rotaci, abychom dosáhli pozice druhého vysílače na souřadnice [a,0,0], kde $a < 0$. Další vysílač musí mít polohu [b,c,0], která bude splňovat podmínky $b > 0$ a $c \neq 0$. Poslední ze čtveřice bodů by následně měl mít polohu [d,e,f], kde $f \neq 0$.

2.2.3 Zvýšení přesnosti

Možností jak zvýšit přesnost určení polohy objektu existuje několik. Jednou z možností je zvýšení přesnosti zjišťování časových rozdílů mezi signály z jednotlivých vysílačů.

Druhou z možností zvýšení přesnosti je princip multilaterace. Tato možnost zvyšuje přesnost na základě získaných časových rozdílů a nijak se nepokouší zvýšit určování přesnosti času. Pokud budeme pracovat v prostoru 2D stačí nám pro určení polohy 3 vysílačů, z jejichž signálu určíme časový rozdíl mezi jednotlivými vysílači. Se zvýšením počtu orientačních bodů můžeme spočítat více poloh zařízení. V rámci přesnosti přijímače se mohou vypočítané polohy lišit. Vypočítáním průměru těchto hodnot získáme jednu výslednou polohu, která bude přesnější. Příklad multilaterace je zobrazen na obrázku 2.5.



Obrázek 2.5: Příklad zvýšení přesnosti pomoci multilaterace ve 2D prostoru.

Získání výsledné polohy pomocí průměru získaných potencionálních poloh (P_x), jak ukazuje obrázek 2.5, můžeme použít, pokud každá vypočítaná potencionální poloha dosahuje, stejné přesnosti. V případě, že by některé potencionální polohy dosahovaly vyšší přesnosti, použijeme přesnost jako váhu pro vypočítání výsledné polohy.

2.3 Způsob komunikace a bezdrátové komunikační zařízení

Jelikož chce objekt znát vlastní polohu, budeme vytvářet aktivní lokalizační systémy. Tyto lokalizační systémy spočívají v tom, že součástí systému jsou radiové vysílače. Lokalizační systém tak vytváří vlastní radiový signál, který můžeme použít i pro komunikaci a ne pouze pro lokalizaci. Pro tento způsob lze tedy využít zařízení, které používá pro komunikaci definovaný standard. Další možností je vytvořit vlastní způsob bezdrátové komunikace. Způsoby komunikací a různá zařízení si popíšeme v následujících částech.

2.3.1 Bezdrátová komunikace

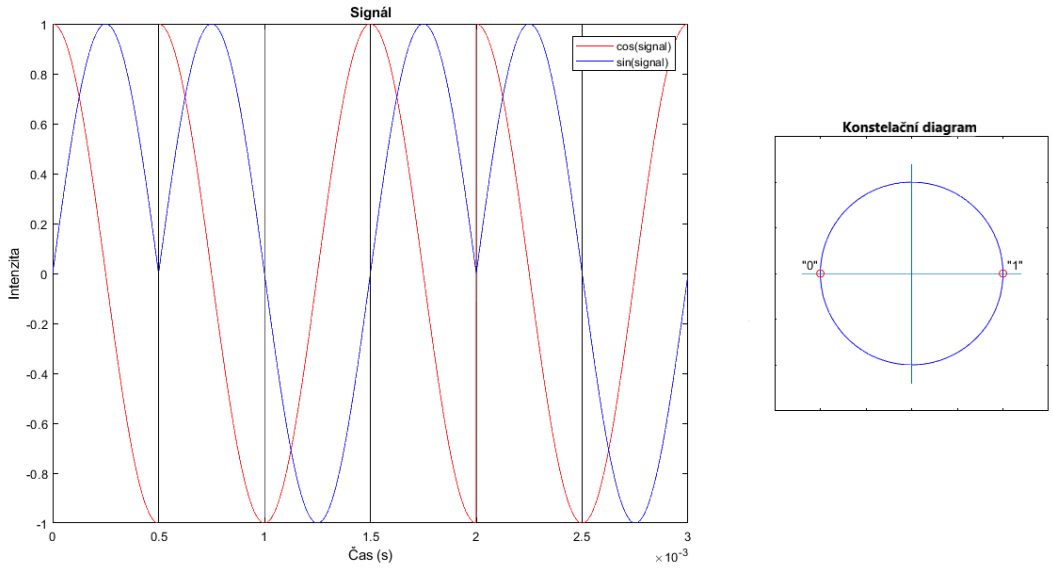
V případě lokalizace jsou přenášeny pseudonáhodné sekvence PRN (Pseudo Random Noice), které se dokola opakují a tvoří lokalizační proud dat. Pseudonáhodné sekvence jsou tvořeny posloupnosti znaku 1 a 0. Jedná se o digitální data, proto pro přenos budeme používat

digitální modulace. Modulace je postup, kdy ovlivňujeme signál na nosné frekvenci vlastním nízkofrekvenčním signálem, který nese informace [48].

Informace u digitální modulace jsou tvořeny posloupností bitů. Modulace signálu umožňuje přenést vyšší počet bitů najednou [34]. A však se zvyšujícím se počtem přenášených bitů najednou se zvyšuje náchylnost přenosu na rušení. V případě přenosu těchto dat je následně definována doba, po kterou je jeden znak vysílán. Po tuto dobu nedochází ke změně signálu, aby všechna přijímací zařízení byla schopna přečíst vysílaná data.

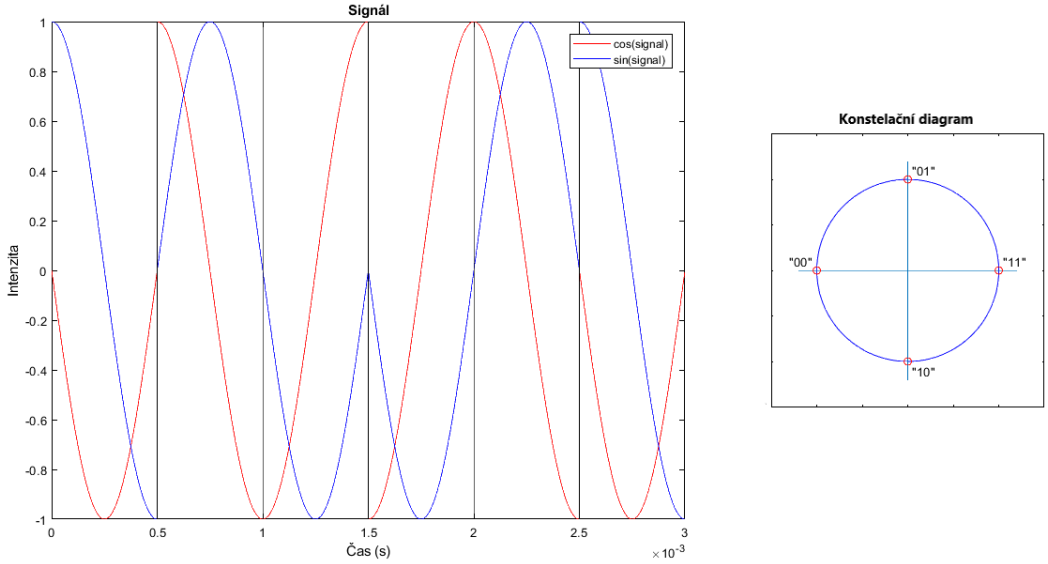
Fázové modulace

Fázové modulace PSK (Phase-shift keying), rozlišují jednotlivé bity zprávy pomocí posunu fáze nosné frekvence. Velikost posunu fáze závisí na počtu přenášených bitů najednou. Jako příklad je na obrázku 2.6 zobrazena modulace typu BPSK [34], u které dochází k přenosu pouze jednoho bitu. Jelikož je přenášen pouze jeden bit zprávy je proveden posuv fáze mezi bity o 180°. Tento způsob modulace používá i systém GPS [43] pro přenos pseudonáhodných sekvencí.



Obrázek 2.6: Zobrazení signálu s digitální modulací BPSK.

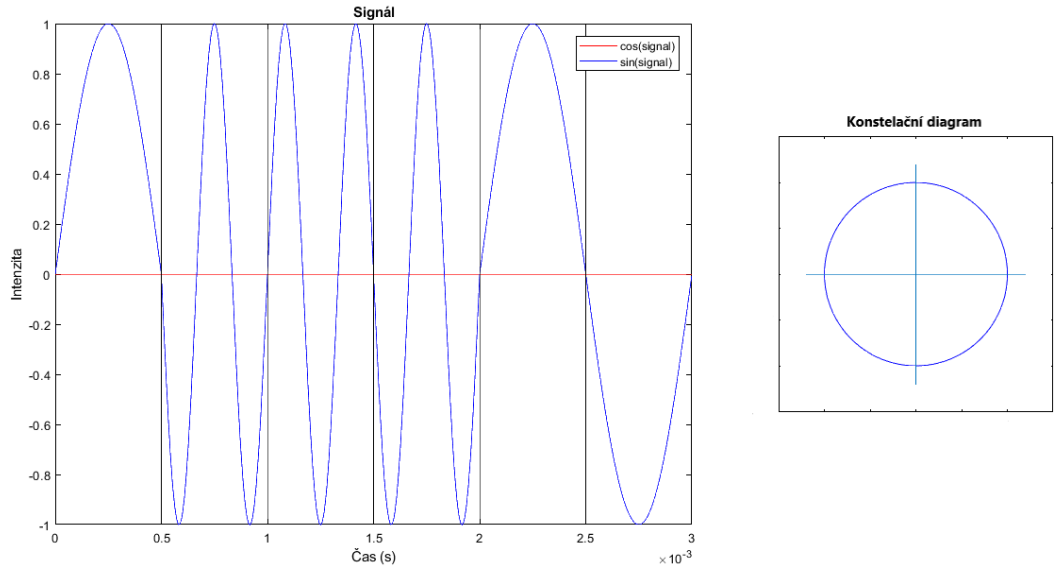
Další z možností fázové modulace je modulace typ QPSK 2.7. Tato modulace přenáší dva bity při jednom posunu fáze, proto posuv fáze mezi jednotlivými přenášenými možnostmi tvoří 90°. Existují i další modulace například 8PSK, 16PSK,...[34] U těchto modulací dochází k přenosu vyššího počtu bitů na jednou.



Obrázek 2.7: Zobrazení signálu s digitální modulací QPSK.

Frekvenční modulace

FSK (Frekvency-shift keying) je způsob modulace, při které dochází ke změně fáze nosné vlny podle hodnoty bitu, který přenášíme. Jedním takovým příkladem je modulace typu GFSK 2.8, kterou pro přenos dat používá zařízení Bluetooth [39]. V jednotkové kružnici nedochází k posunu fáze, proto nejsou zobrazené žádné body pro jednotlivé bity.

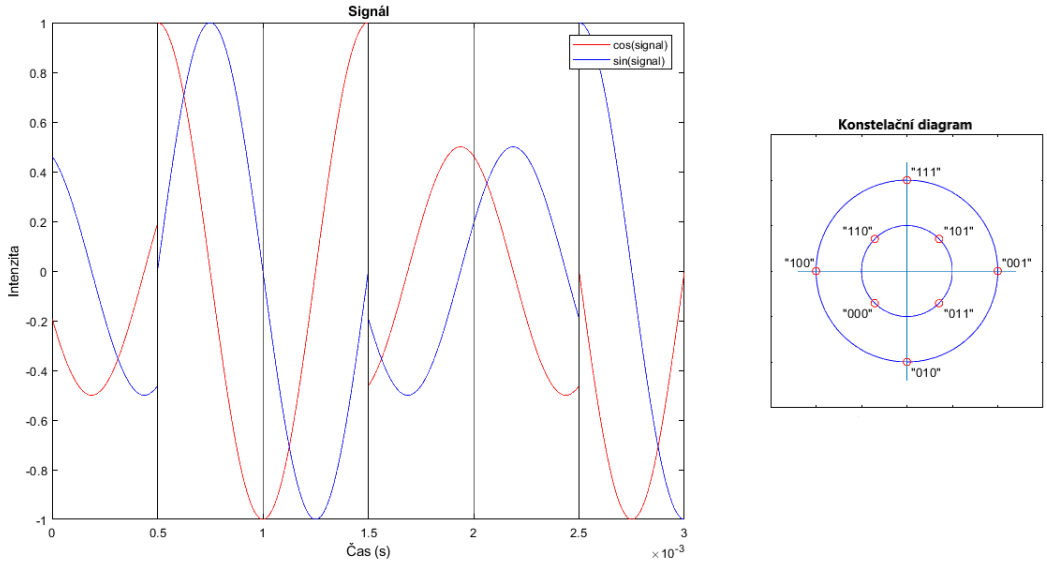


Obrázek 2.8: Zobrazení signálu s digitální modulací GFSK.

Kvadraturní amplitudové modulace

QAM (Quadrature amplitude modulation) je modulace založená na změně fáze signálu, ale zároveň mění i velikost amplitudy signálu. Mezi tyto modulace patří například 8-QAM

2.9, nebo také 16-QAM, 32-QAM, 64-QAM,... [34] Tyto modulace se používají pro vysoký datový přenos. Například WiFi standard 802.11 používá modulaci 256-QAM [1].



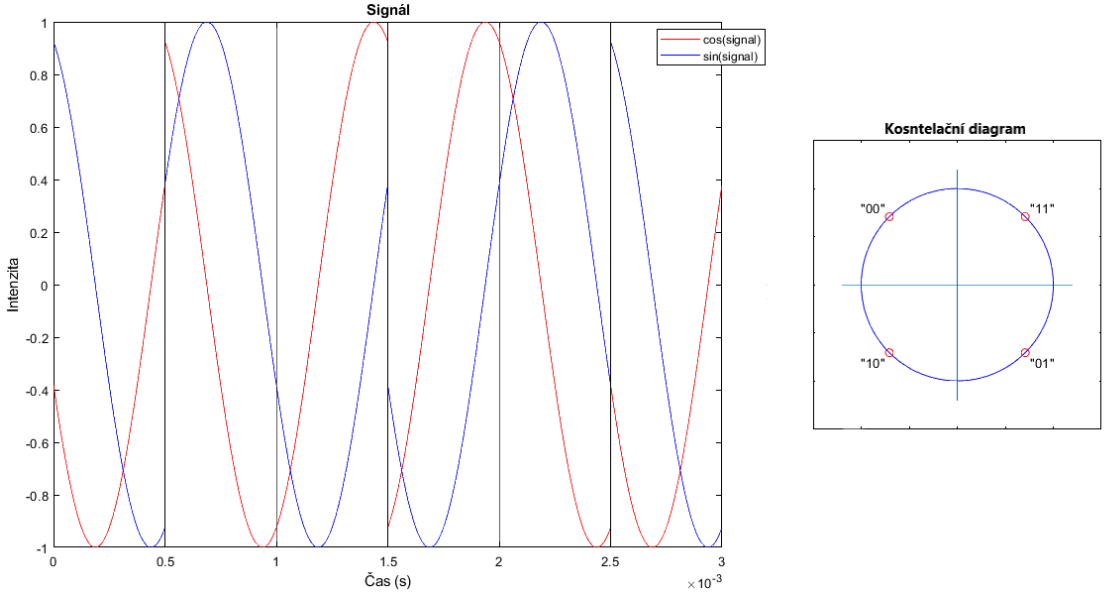
Obrázek 2.9: Zobrazení signálu s digitální modulací 8-QAM.

2.3.2 Lokalizační zařízení

Jako lokalizační zařízení lze použít "ATZB-X-233-XPRO-ND", které používá pro komunikaci standard 802.15.4. Tyto transceivery pro lokalizaci nepoužívají pseudonáhodné sekvence, ale celé zprávy zahrnující další informace o jednotlivých transceiverech. Loklizační data u tohoto zařízení tvoří celé zprávy a pro zpoždění signálu používá čítač označovaný jako "Time-of-Flight", pomoci kterého můžeme počítat dobu cestování signálu. Čítač tvoří oscilátor na frekvenci 16MHz to znamená, že do čítače je přičtena hodnota každých 62,5 ns. Při součinu tohoto času a rychlosti světla nám vychází, že pomoci tohoto čítače můžeme dosáhnout přesnosti 18 metrů. Přesnou přesnost lze nejlépe zjistit pomoci testování. Přesnost lze také zvýšit pomoci multilaterace.

Způsob komunikace

Lokalizační zařízení používá pro bezdrátovou komunikaci transceiver. Pro komunikaci používá standard IEEE 802.15.4, který byl vytvořen za pomoci ZigBee Alliance [33]. Protokol umožňuje vytvářet vlastní bezdrátovou síť WPAN (Wireless Personal Area Network) s nízkou přenosovou rychlostí, která má nízkou spotřebu energie. Standard definuje více způsobů komunikace na různých frekvencích. Vybraný transceiver pracuje na frekvenci 2,4GHz. Pro tuto frekvenci používá standard modulaci signálu typu O-QPSK 2.10 (Offset Qudrature Phase Shift Keying) [36].



Obrázek 2.10: Zobrazení signálu s digitální modulaci O-QPSK.

2.3.3 Softwarové rádio

SDR (Software Defined Radio) [46] je radiová technologie, kde veškerou práci s radiovým signálem provádí software. Pro práci s radiovým signálem je však potřeba mít radiový přijímač respektive vysílač. Konstrukce takovéhoto přijímače se skládá ze dvou částí. Radiového přijímače, který posílá přijímaná data přímo na A/D převodník a o následné zpracování signálu se stará už pouze software [7]. Softwarové rádio tak umožňuje provádět změny ve zpracování signálu a testovat různé způsoby komunikace bez upravování hardwaru. Lze tedy použít jeden radiový přijímač pro různé typy komunikace, jelikož však veškeré zpracování signálu provádí software může být složitější dosáhnout zpracování v reálném čase.

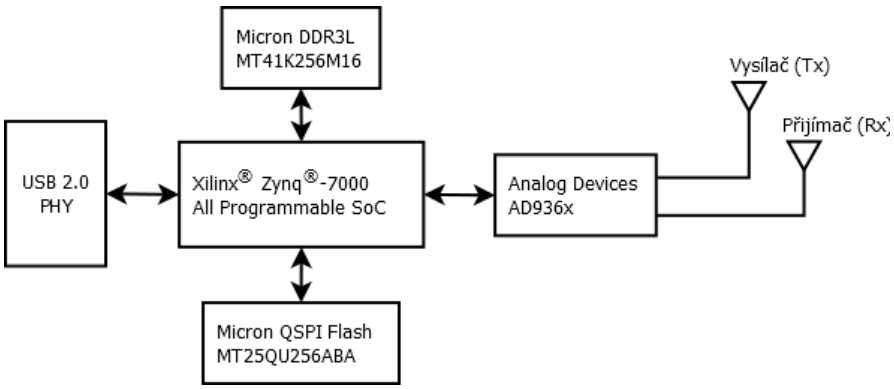
Radiových přijímačů, které posílají digitalizovaný přijímaný signál je velké množství. Většina přijímačů používají pro komunikaci s počítačem rozhraní USB. Důležité parametry, ve kterých se jednotlivé přijímače liší, je rozsah frekvencí, na kterých jsou přijímače schopny přijímat. Druhým rozdílem jsou vlastnosti A/D převodníku. Tedy vzorkovací frekvence a velikost výstupních digitální dat. Dalším důležitou vlastností je, zda se jedná o přijímač nebo vysílač. Pokud chceme vytvořit vlastní lokalizační systém budeme potřebovat jak vysílače, které budou tvořit orientační body, tak zároveň přijímač.

Jedno z takové zařízení jsem použil v bakalářské práci [40]. Konkrétně se jednalo o zařízení Realtek RTL2832U [12]. USB zařízení je radiový přijímač, který umožňuje přijímat signál na frekvencích od 25MHz až po 1750MHz, se vzorkovacím kmitočtem až 2,8 MHz. Zařízení také obsahuje 8-bitový A/D převodník, který vzorkuje přijímaný signál a následně jej posílá v digitální podobě do počítače právě přes rozhraní USB. Další příkladem může být zařízení Airspy mini [10]. Jelikož veškeré následující zpracování signálu provádí software je možno pomoci jednoho zařízení zpracovávat různé signály. Příkladem můžou být aplikace pro zpracování FM rádia [11], GPS signálu [6] a mnohé další.

Použité zařízení v této práci bude ADALM-PLUTO [9], které umožňuje jak přijímat, ale i vysílat radiový signál a proto je vhodné pro tento projekt. Tímto zařízením se zaobírá následující část.

ADALM-PLUTO

Zařízení ADALM-PLUTO je radiový přijímač a vysílač (transceiver) od firmy Analog Devices. Zařízení se skládá z několika částí, které jsou zobrazeny v blokovém schématu 2.11.



Obrázek 2.11: Blokové schéma zařízení ADALM-PLUTO [14].

První částí je radiový transceiver Analog Device AD936x [13]. Transceiver umožňuje jak vysílat tak i přijímat radiový signál najednou a je tedy plně duplexní (full duplex). Proto je připojen k částem označené jako vysílač (Tx) a přijímač (Rx). Tyto části jsou výstupní konektory SMA pro připojení antén, které z důvodu plného duplexu jsou pro vysílač a přijímač zvlášť. Transceiver dále obsahuje 12-bitové A/D převodníky pro přijímaní signálu a D/A převodníky pro vysílání signálu, kde vzorkovací frekvence těchto převodníků je až 20MHz.

Transceiver se v zařízení ADALM-PLUTO vyskytuje ve třech verzích (`AD9361` [15], `AD9363` [16], `AD9364` [17]). Rozdíly mezi jednotlivými typy jsou v kmitočtovém rozsahu 325MHz - 3,8GHz pro `AD9363` a 70MHz - 6,0GHz pro zbylé dva typy. Mezi jednotlivými typy je však rozdíl i v počtu přijímačů a vysílačů, kde typ `AD9365` obsahuje jeden vysílač a jeden přijímač oproti zbylým dvou typům, které mají dva vysílače a přijímače avšak v zařízení ADALM-PLUTO je vždy připojen pouze jeden vysílač a jeden přijímač.

Hlavní část tvoří FPGA modul od firmy Xilinx, kde zařízení ADALM-PLUTO používá konkrétní typ `XC7Z010-1CLG225C4334`. Součásti FPGA modulu je i dvou jádrový procesor typu ARM. Samotné FPGA má pouze 256KB pamětí RAM, která je však v zařízení ADALM-PLUTO rozšířena (`Micron DDR3L`) stejně jako flash paměť (`Micron QSPI Flash`). Na procesoru ARM běží operační systém Linux, který je upraven právě pro zařízení ADALM-PLUTO.

Jelikož na zařízení ADALM-PLUTO je spuštěný operační systém Linux, může toto zařízení vysílat a přijímat signál bez nutnosti připojení k počítači. Na výstupní port USB se tak může připojit další periferní zařízení jako například WiFi, LAN, paměťové zařízení, atd., nebo pomocí tohoto rozhraní připojit zařízení k počítači. Zařízení má dále druhý USB port, který slouží k napájení v případě, že zařízení není připojeno k počítači.

2.4 Dílčí závěr

V první části této kapitoly jsem se zaměřil na lokalizační systémy, kde jsem se podívali na způsob, jakým mohou určovat polohu objektu. Lokalizační systémy jsem tak rozdělil na aktivní a pasivní, kde aktivní systémy jsou takové, které vysílají radiový signál. Tyto loka-

lizační systémy využívají právě vysílaný signál pro lokalizaci. Pasivní lokalizační systémy jsou však složeny pouze z přijímačů a sledují odražený radiový signál od objektů. Radiový signál, který se odráží je tvořen jinými systémy například, televizním nebo radovým vysíláním, a mnoho dalších.

Následně jsem se podíval na jednu z mnoha možných metod výpočtů polohy a to metodu TDOA. U této metody jsem se zaměřil na rovnice výpočtu polohy. Také jsem se zaměřil na rozložení, jaké musí mít jednotlivé orientační body lokalizačního systému, aby bylo vůbec možné spočítat polohu. Poslední v této části je popis způsobů, kterými lze dosáhnout zvýšení přesnosti lokalizačního systému.

V poslední části této kapitoly jsem popsal způsob, jakým mohou zařízení v lokalizačním systému komunikovat. Vybral a popsal jsme několik typů digitální modulace, které se používají pro přenos dat. Následně jsem se podívali na některá radiová zařízení. Jako příklad jsem tak zvolil jedno zařízení, které je tvořeno přímo pro lokalizaci. Na tomto zařízení jsem se podíval co takové zařízení pro tvorbu lokalizačního systému obsahuje. Potom jsem zvolil technologii softwarově definovaného rádia a konkrétně zařízení ADALM-PLUTO. Technologie softwarově definovaného rádia umožňuje provádět různé implementace zpracování signálu bez zásahu do hardwaru.

Kapitola 3

Návrh lokalizačního systému a algoritmů pro zpracování dat

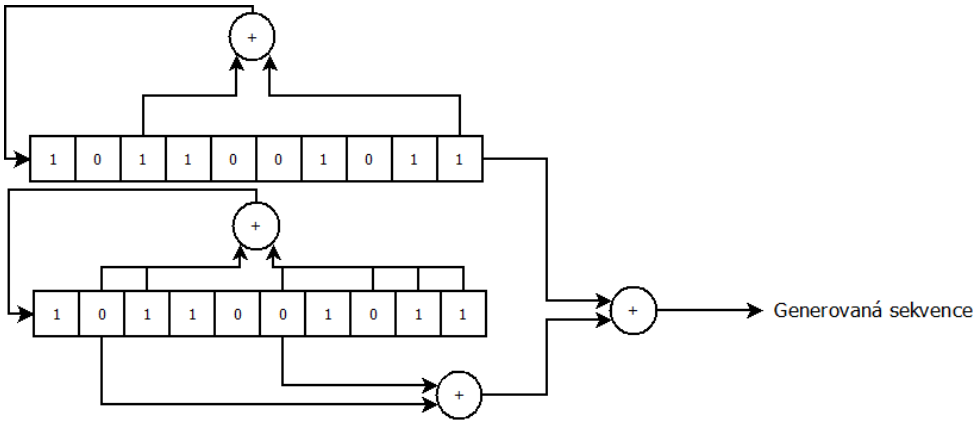
3.1 Návrh algoritmu pro zpracování lokalizačních dat

V předchozí kapitole jsme si popsali některé z možností lokalizace, které se používají. Dále jsme si popsali možnosti komunikace mezi zařízeními za použití radiového signálu. V této kapitole si popíšeme způsob vysílání a data, jaká budou vysílat jednotlivé vysílače. Způsob, kterým budeme lokalizační data vysílat a typ lokalizačních dat použijeme stejný jako se používá u systému GPS [8].

3.1.1 Návrh způsobu komunikace

Systém GPS používá pro lokalizaci zlaté sekvence (Gold code), které moduluje pomoci fázové modulace typu BPSK. Zlatá sekvence je pseudonáhodná sekvence tvořena hodnotami 0 a 1. Pro vytvoření zlaté sekvence se používají dva posuvné registry [4]. Oba registry mají stejnou délku a velikost registru určuje celkovou délku sekvence.

Pro generování signálu budeme používat 10 bitové posuvné registry. Výsledná délka sekvence bude 1023 znaků. Návrh generátoru pseudonáhodné sekvence je zobrazen na obrázku 3.1. Pomoci změny pozic, ze kterých získáváme znaky ze spodního registru pro výslednou sekvenci [3], měníme výslednou sekvenci. Každý z vysílačů bude vysílat jinou, předem definovanou sekvenci.



Obrázek 3.1: Generátor zlaté sekvence.

Jednotlivé vysílače budeme od sebe rozlišovat na základě této sekvence. Sekvence se bude opakovat ve smyčce stále dokola a použijeme metodu digitálního multiplexování typu CDMA. Všechny vysílače tím pádem budou vysílat sekvenci na stejné frekvenci.

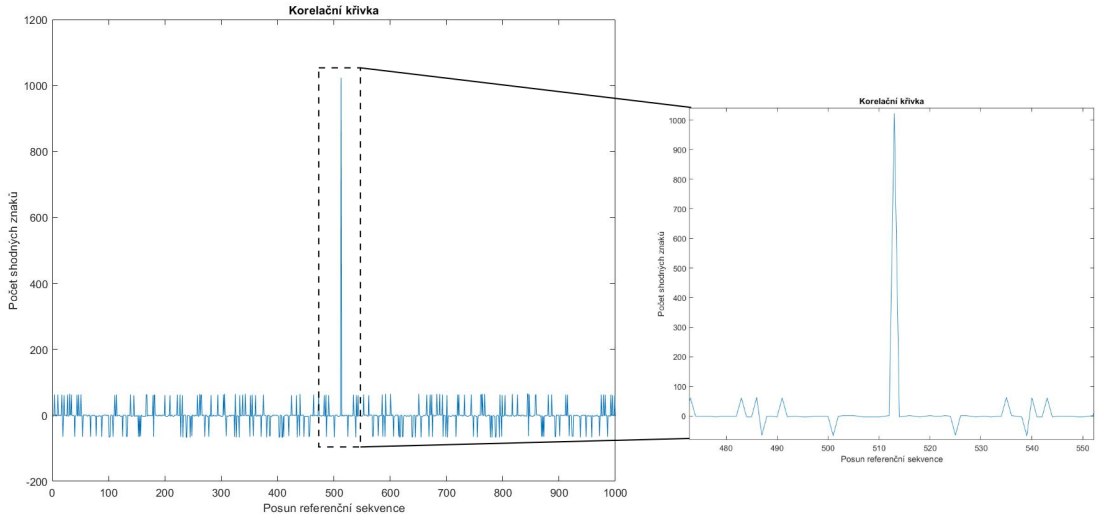
Použijeme digitální modulaci typu BPSK, která sice umožňuje přenos pouze jednoho znaku, nicméně tato metoda je více odolná a při vzájemném rušení nebude docházet k vysoké ztrátě dat. Fázová modulace provádí změny fáze signálu podle posílaných dat, jak je popsáno v kapitole 2.3.

3.1.2 Návrh zpracování signálu

Zpracovávat signál můžeme více způsoby. Jednou z možností je demodulovat signál, kde výsledkem je právě posílaná zlatá sekvence. Následně je prováděna korelace [19], jejíž výsledkem je zjištění časových rozdílů přijetí signálu z jenotlivých vysílačů. Možnost jsem testoval v bakalářské práci [40]. Touto metodou jsem byl schopný určit časové zpoždění na přesnost vzorku přijímaného signálu. Přesnost byla závislá na vzorkovací frekvenci radiového přijímače.

Korelace

Korelace je porovnání přijímaného signálu a referenční zlaté sekvence na straně přijímače. Referenční sekvence je stejná na přijímači jako na vysílači a nesmí být na přijímači nijak posunuta. V případě, že byla posunuta referenční sekvence, pak se musí posunout i výsledek korelace. Korelace vytváří korelační křivku, která je zobrazena na obrázku 3.2. Výsledkem korelace je vrchol této křivky.

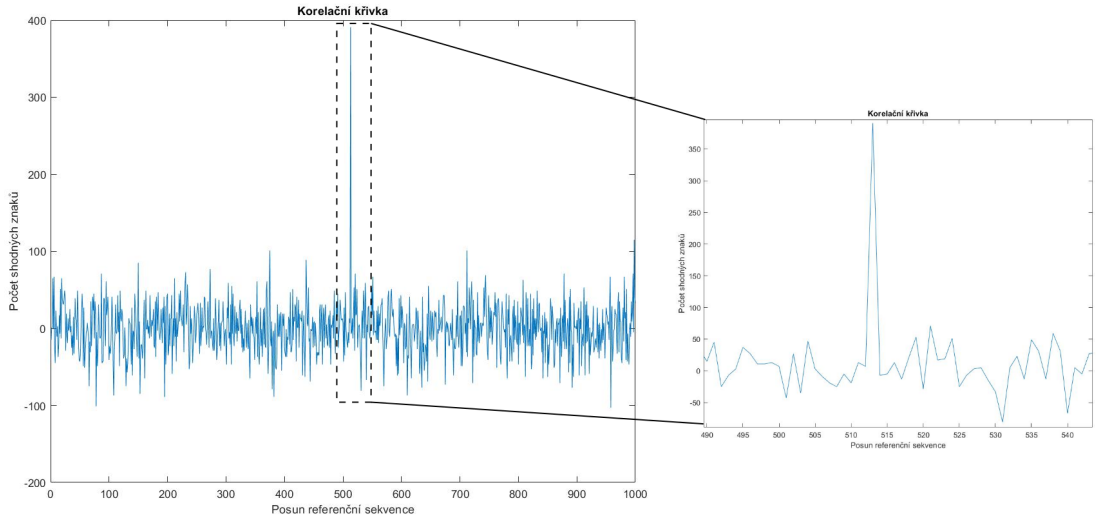


Obrázek 3.2: Korelační křivka pro zlatou sekvenci bez chyb v komunikaci.

Vrchol této křivky určuje čas, kdy byla referenční sekvence přijata. Pro více vysílačů používáme více referenčních sekvencí a výsledkem je více korelačních křivek. Za předpokladu, že jsou vysílače synchronizovány, tedy vysílají počátek sekvence ve stejný čas, pomoci rozdílu ve vrcholech jednotlivých korelačních křivek získáme rozdíl času, a tedy i vzdálenosti

jednotlivých vysílačů od přijímače. Pro výpočet polohy pak mohu následně použít metodu TDOA 2.1.2.

V průběhu přenosu dat mezi vysílači a přijímačem dochází k rušení, které má vliv na určení správného přenášeného znaku. Další rušení vzniká proto, že všechny vysílače vysílají na jedné frekvenci. Provedl jsem tedy opět korelaci, ale uměle jsem vytvořil náhodně v polovině znaků chyby. Výsledek korelace signálu s chybami a referenční sekvence je na obrázku 3.3. Z výsledku korelace je jasně viditelné, že i přes chybu v polovině znaků jsme schopni určit správnou pozici přijetí sekvence.

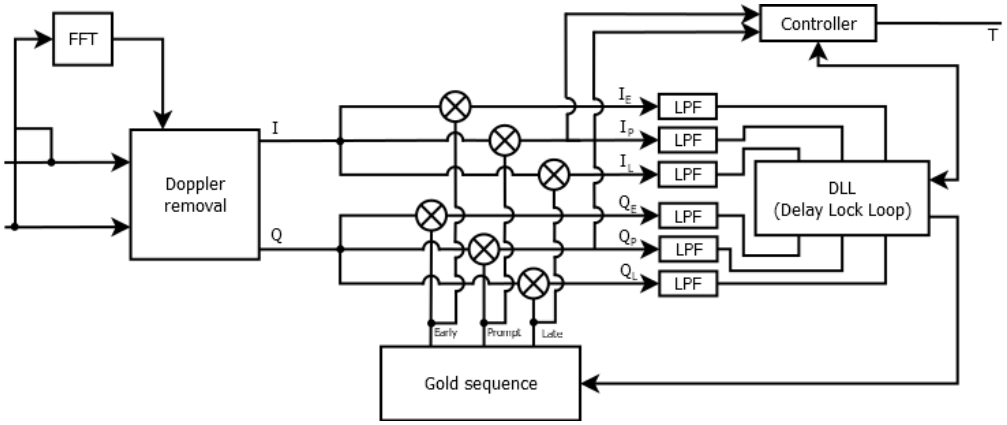


Obrázek 3.3: Korelační křivka pro zlatou sekvenci s chybami.

Poslední z chyb, které ještě může ovlivnit korelační křivku, je špatné určení znaku. Tedy pokud znak -1 je označen jako 1 a obráceně. V takovémto případě je korelační křivka ovlivněna pouze tak, že vrchol, který určuje kdy sekvence přišla nebude kladný, ale záporný.

Multikorelátor

Druhou možností zpracování signálu je pomocí multikorelátoru [21]. Tuto metodu používají pro lokalizaci GNSS systémy. Multikorelátor je část, která provádí synchronizaci přijímaného signálu a referenční sekvence pomoci posouvání referenční sekvence. Blokové schéma multikorelátoru je na obrázku 3.4. Jeden multikorelátor pracuje pouze s jednou sekvencí, která je definovaná při inicializaci multikorelátoru. Pro každý vysílač, tak musí mít přijímač jeden mutlikorelátor. Vstupem multikorelátoru je přijímaný signál a výstup tvoří zpoždění (T), které bylo zjištěno v případě, že multikorelator dosahl synchronizace vstupniho signálu a referenční sekvence.



Obrázek 3.4: Blokové schéma multikorelátoru [21] [18] [23].

Přijímaný signál se v první části multikorelátoru rozdělí. Všechny vzorky I a Q jdou tedy do části označené jako `"Doppler removal"` a druhého prvku označeného jako `"FFT"`. Prvek `"FFT"` provádí rychlou Fourierovu transformaci nad upraveným přijímaným signálem podle rovnice 3.1 [22]. Vzorky I a Q musí mít hodnotu v intervalu $< -1; 1 >$. Rovnice odstraňuje ze signálu modulaci, tedy změnu fáze. Výsledkem je signál sinusového tvaru, který má dvojnásobnou frekvenci, než je frekvence přijímaného signálu. Takto upravený signál lze ještě poslat skrze filtr dolní propusti, aby se odstranilo rušení vzniklé během přenosu. Pomoci rychlé Fourierovi transformace se zjistí frekvence signálu, která se vydělí dvěma a následně tuto hodnotu předá prvku `"Doppler removal"`.

$$2\Phi = asin(Q * I) \tag{3.1}$$

Druhým směrem, kterým je neupravený přijímaný signál vysílán je do prvku `"Doppler removal"`. Tento prvek provádí odstranění Dopplerova jevu ze signálu a výsledkem jsou demodulovaná data, konkrétně tedy fáze signálu. Na základě získané frekvence z `"FFT"` si prvek vytváří vlastní nemodulovaný signál, tedy sinusoidu. Ke generování tohoto referenčního signálu slouží prvke NCO (Numeric control oscillator) [32], který tento signál vytváří. Dále prvek musí určit aktuální úhel signálu a dosáhnout tak stejné fáze referenčního signálu s přijímaným. K tomuto účelů můžeme použít rovnici 3.2, jejíž výsledkem je aktuální úhel signálu. Tato rovnice kopíruje přijímaný signál a úhel, který rovnice určuje se mění zároveň se změnou fáze signálu. Druhou z možností je rovnice 3.3. Tato rovnice není závislá na změně fáze a však výsledná hodnota je pouze ve čtvrtém a prvním kvadrantu jednotkové kružnice. Úhel generovaný touto rovnicí mění svojí fázi jednou za půl amplitudy.

$$\Phi = atan2(Q, I) \tag{3.2}$$

$$\Phi = atan(Q/I) \tag{3.3}$$

Na základě získané frekvence přijímaného signálu a jeho úhlu lze odstranit Dopplerův jev z přijímaného signálu. Frekvence přijímaného signálu se vlivem Dopplerova jevu může v průběhu měnit, proto je třeba provádět kontrolu frekvence signálu a synchronizaci úhlu signálu opakovaně. Odstranění Dopplerova jevu probíhá pomocí vynásobení referenčního signálu s přijímaným signálem a výsledkem je fáze signálu.

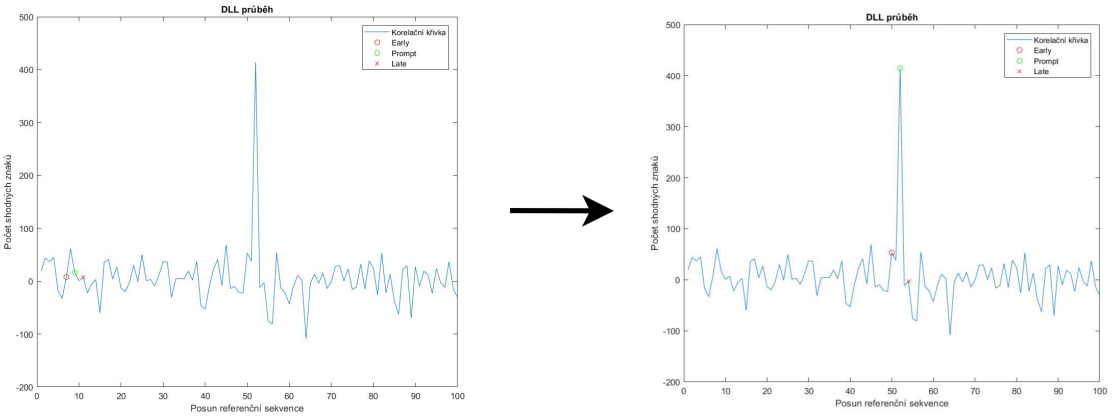
Následně upravený signál je porovnáván s referenční sekvencí. Porovnávání probíhá pomocí vynásobení jednoho vzorku přijímaného signálu s jedním vzorkem referenční sekvence. Tato sekvence je uložena v bloku s názvem `"Gold sequence"`, která je tvořena hodnotami 1 a -1 a je uložena v kruhovém registru. Ze sekvence jsou vybrány tři vzorky, které jsou od sebe v sekvenci stejně vzdálené. Takto získané vzorky jsou označeny jako "*early*", "*late*"a "*prompt*". Jedna hodnota výsledku násobení určuje, zda se fáze upraveného přijímaného signálu shodovala s referenční sekvencí. Po provedení porovnání vzorků sekvence se vzorkem vstupního signálu jsou ze sekvence vybrány následující vzorky, které jsou porovnány s dalším vzorkem signálu. Výsledek porovnání je přiveden na prvky `"LPF"` a porovnání se vzorky "*prompt*"jsou také přivedené na prvek `"Controller"`

LPF (Low Pass Filter) jsou filtry dolní propusti. Další část označená jako `"DLL"` provádí kontrolu vstupu po určitém počtu vzorků, ne pro každý nový prvek. Filtry dolní propusti mohou pracovat v tomto případě jako suma určitého počtu prvků, která bude vydělená právě počtem uložených prvků vstupních dat. Filtry v tomto případě vytváření průměr určitého počtu prvků. Pokud by se prvky chovaly přímo jako filtr dolní propusti a získávaly výsledek každý vzorek, musely by si pamatovat tento určitý počet prvků. Při každém novém prvku by musely provést přepsání nejstarší hodnoty a opět vytvořit průměr. Výpočet sumy pro získání průměru lze zjednodušit a to tak, že si filtry budou pamatovat ještě celkovou sumu. Od této sumy následně odečtou nejstarší prvek a přičtou hodnotu nového prvku. Tím získáme novou sumu bez nutnosti sčítat všechna čísla. V paměti LPF pak nový prvek přepíše ten nejstarší.

Hlavní část multikorelátoru tvoří prvek DLL (Delay Locked Loop) [20]. Tato část již pracuje se signálem, který je porovnán s referenční sekvencí, tedy s výsledkem korelace. DLL provádí porovnání průměru několika posledních vzorků "*early*", "*late*"a "*prompt*"se vzorky I a Q vstupního signálu. Na základě těchto hodnot si vypočítá podle jedné z rovnic 3.4 hodnotu diskriminátoru. Hodnota diskriminátoru se určuje:

$$\begin{aligned} discriminator &= \frac{[(I_E - I_L) * I_P] + [(Q_E - Q_L) * Q_P]}{2} \\ discriminator &= \frac{[(I_E^2 + Q_E^2) - [(I_L^2 + Q_L^2) * Q_P]}{2} \end{aligned} \tag{3.4}$$

Na základě hodnoty diskriminátoru je upravena pozice výběru vzorků "*early*", "*late*"a "*prompt*"z `"Gold sequence"`. Pokud hodnota diskriminátoru je větší jak 0, pak provede rotaci sekvence doprava v `"Gold sequence"`. V opačném případě provede rotaci směrem doleva. Cílem DLL je najít vrchol v korelační křivce a udržet se na tomto vrcholu, tak jak je zobrazeno na obrázku 3.5. O posunu je také informován prvek `"Controller"`.



Obrázek 3.5: Průběh posouvání sekvence pro získáni vrcholu korelační křivky.

Poslední částí multikorelátoru je `"Controller"`. Tato část má na vstupu hodnoty I a Q, které byly porovnány s referenční sekvencí. Prvek si tyto dvě hodnoty sečte a pokud výsledek součtu je větší než jedna což znamená, že referenční sekvence se shoduje s přijímaným signálem, prvek inkrementuje vlastní paměť o jedničku. V opačném případě jedničku odečte. Hodnotu této paměti následně prvek použije pro vytvoření průměru. Na základě výsledného průměru získaných hodnot, určí zda DLL dosáhlo vrcholu korelační křivky. Tato kontrola je prováděna po několika iteracích, které provádí prvek `"DLL"`, od kterého `"Controller"` dostává hodnotu provedeného posunu referenční sekvence.

Pokud průměrná hodnota je menší, než nějaká stanovená mez například 0,6 tedy 60% znamená to, že nebylo dosaženo synchronicity signálu a referenční sekvence. V tomto případě pošle `"Controller"` prvku `"DLL"`, aby provedl posun v referenční sekvenci o jeden znak. Tento postup zabraňuje aby se `"DLL"` nezavěsil na lokálním maximu nebo minimu a hledal globální maximum nebo minimum. Pokud však průměrná hodnota je větší, než nějaká stanovená mez například 0,6 tedy 60%, je na výstup vložena hodnota posunu, která byla proveden na referenční sekvenci prvkem `"DLL"` a `"Controller"`. Následně se tato hodnota vynásobí se vzorkovací frekvencí a výsledkem je zpoždění vrcholu korelační křivky.

Pří demodulaci signálu, který má modulaci BPSK může dojít ke změně fáze demodulovaného signálu. Hledaný vrchol korelační křivky by tvořil globální minimum. Proto `"Controller"` musí provádět kontrolu zavěšení na větší než je stanovené procento synchronicity a nebo menší než je záporná hodnota stanovené synchronicity. Případně o této chybě, tedy že jsou obrácené fáze, muže informovat prvek `"Doppler removal"`, který na základě tohoto může provést otočení fáze vnitřního NCO.

3.2 Návrh vlastního lokalizačního systému

Jedním z hlavních bodů zadání je vytvořit vlastní lokalizační systém pro lokalizaci dronu. Nyní si vytvoříme návrh vlastního lokalizačního systému, ve kterém se budou pohybovat drony. Bude se jednat o vlastní lokalizační systém, který bude mít vlastní souřadnicový systém.

3.2.1 Modulace signálu a metoda lokalizace

Jelikož pro tvorbu vlastního lokalizačního systému použijeme zařízení ADALM-PLUTO 2.3.2, můžeme si zvolit jakou modulaci použijeme. Můžeme také zvolit jaká lokalizační data budeme posílat. V podkapitole 3.1 jsme popsali zlaté sekvence (Gold code), které použijeme jako lokalizační data. Všechna zařízení, která budou vysílat lokalizační data, budou vysílat na jedné frekvenci. Bude tak použita metoda CDMA, kde sekvence slouží pro rozlišení jednotlivých zařízení. Jelikož všechna zařízení budou vysílat na jedné frekvenci, je třeba zvolit metodu modulace signálu takovou, která je odolná na vzájemné rušení. Nejodolnější modulací signálu, kterou jsme si popsali 2.3 je modulace typu BPSK.

Co se týká metody výpočtu polohy, tak zde použijeme metodu TDOA popsanou v podkapitole 2.1.2. Časové zpoždění nám je schopen určit multikolerátor 3.1, který porovnává sekvence z jednotlivých vysílačů s referenční sekvencí. Podle posunu referenčních sekvencí jsme schopni určit časové rozdíly mezi jednotlivými vysílači. Podmínkou, aby tato metoda fungovala je, že všechny vysílače, které tvoří orientační body musí vysílat počátek sekvence ve stejný čas. Na několik možností jak dosáhnout synchronizace času se podíváme v další části.

3.2.2 Základna lokalizačního systému

Každý lokalizační systém se musí skládat z několika vysílačů, které znají vlastní pozici a vysílají vlastní sekvenci. Podle podkapitoly 2.1 potřebujeme pro určení polohy objektu ve 3D prostoru minimálně 4 vysílače, s přesnou vlastní polohou. Vysílačů však může být více. Více vysílačů nám umožní přesněji určit polohu objektu, ale také v případě výpadku jednoho vysílače nedojde k zničení systému, a tím neztratíme možnost určení polohy. Všechny tyto vysílače tvoří takzvané srdce lokalizačního systému. My tuto součást nazveme jako základna lokalizačního systému.

Lokalizační základna se bude skládat z vysílačů, které budou ve smyčce posílat vlastní zlatou sekvenci. Jedna z podmínek kladených na základnu lokalizačního systému je způsob umístění jednotlivých vysílačů. Pro správné a přesné získání je zapotřebí správně umístit vysílače. Jelikož budeme měřit pozici objektu ve 3D prostoru, je vhodné aby všechny vysílače základny nebyly v jedné rovině. Ať už se jedná o roviny X, Y nebo Z. V případě, kdy všechny vysílače budou umístěny ve stejné výšce, nemůže objekt určit přesně svojí výšku, protože neví jestli je výše nebo níže než vysílače tvořící základnu.

Jak již bylo zmíněno základna je složena z několika vysílačů, které znají přesně vlastní polohu. Každý vysílač vysílá jedinečnou pseudonáhodnou na jejíž základě od sebe rozeznáme. V předchozí části jsme také uvedli, že vysílače musí mít synchronizovaný čas, aby bylo možno vysílat počátek sekvence ve stejný čas.

Bez synchronizace signálu

Jednou z možností je neprovádět synchronizaci mezi vysílači. Avšak je potřeba znát časové rozdíly jednotlivých vysílačů. Každý z vysílačů bude vysílat vlastní sekvenci pořád dokola bez synchronizace. Vybere jeden z vysílačů, který prohlásíme za hlavní. Ostatní vysílače budou provádět korelaci své sekvence se sekvencí hlavního vysílače. Časový rozdíl, který jednotlivé vysílače naměří, budou poslány spolu s pozicí vysílače objektu. Objekt provede korelaci všech vysílačů a zjisti časové rozdíly, ke kterým přičte právě zjištěné odchylky.

Nevýhodou této implementace je, že musíme zajistit další komunikační kanál. Na tento budou posílány jak pozice vysílačů, tak časové rozdíly jednotlivých vysílačů. Výhodou této

implementace je, že můžeme volně měnit pozice vysílačů, protože vysílač vysílá svou aktuální polohu.

Se synchronizací signálu

Účelem této implementace je využívat pouze jeden kanál, jelikož zařízení ADALM-PLUTO má pouze jeden vysílač a jeden přijímač. Vysílače tvořící základnu budou mít jednu neměnnou pozici. Všechny vysílače základny znají navzájem pozice všech vysílačů a jejich sekvence. Ze všech vysílačů se zvolí jeden, který bude označen za referenční. Jelikož každý z vysílačů zná svou polohu a polohu referenčního vysílače, může zjistit jaký časový rozdíl má být mezi jeho sekvencí a sekvencí referenčního vysílače. Každý z vysílačů se snaží udržet tento časový rozdíl a tím dochází k synchronizaci vysílačů.

Výhodou tohoto způsobu implementace je, že používáme pouze jeden kanál. Nevýhodu však tvoří fakt, že není nijak posílána poloha jednotlivých vysílačů. V případě změny jednoho vysílače musíme o tom nějakým způsobem dát vědět objektu. V případě, že bychom změnili referenční vysílač musí se změnit pozice tohoto vysílače u všech vysílačů, které jej berou jako referenční.

Základna určuje polohu objektu

V předchozích dvou možnostech jsme si navrhli způsob, jak mohou objekty díky lokalizačnímu systému získat svou polohu. Navrhneme nyní opačnou možnost a to takovou, že lokalizační systém určí polohu objektu. K tomuto účelu se musí objekt stát vysílačem a bude vysílat vlastní pseudonáhodnou sekvenci. Následně bude vysílat pouze jeden vysílač ze základny, který bude označen za referenční. Slouží k tomu, aby se opět všechny zařízení tvořící základnu mohli synchronizovat. Všechny pozemní zařízení mají neměnnou pozici a znají navzájem svou polohu.

Všechny vysílače pak jednomu bodu posílají časový rozdíl mezi sekvencí referenčního vysílače a sekvencí objektu. Na základě známých časových rozdílů a pozic vysílačů lze vypočítat polohu objektu. Tento návrh je možné zkombinovat s předchozím návrhem, kde všechny vysílače vyslali svou sekvenci. Tato možnost následně umožní, že objekt může vypočítat svou polohu a lokalizační systém můžeme vypočítat polohu objektu. A však u tohoto návrhu musíme opět přenášet časové rozdíly a případně i polohu mezi jednotlivými transceivery, které tvoří základu.

3.3 Dílčí závěr

V této kapitole jsem určil, jak bude probíhat komunikace v rámci lokalizačního systému. V první části jsem popsal jak generovat zlatou sekvenci (Gold code), kterou budeme používat pro lokalizaci. Dále jsem si také uvedl jak generovat rozdílné sekvence, abych od sebe rozdělil jednotlivé vysílače. Každý z vysílačů bude mít vlastní sekvenci. V navazující části jsem se podíval na korelaci, tedy porovnávání sekvence s přijímaným signálem. Výsledkem korelace bude zjištění velikosti posunu sekvence na přijímači, což pro více sekvencí z více vysílačů dá dostatečný počet informací pro zjištění polohy. Korelaci je možno provádět pomoci porovnání celé referenčních dat s každým přijatým vzorkem demodulovaného signálu. Druhou možnosti je pak použít multikoreláator, který provádí průběžné srovnávání signálu a referenční sekvence.

V druhé části této kapitoly jsem navrhl několik možností, jak by mohla vypadat základna lokalizačního systému. V těchto návrzích jsem hlavně uvedl, co musí splňovat vysílače a přijímače v lokalizačním systému. V části je popsáno jakým způsobem a jaká zařízení v lokalizačním systému musí provádět synchronizaci, případně jaká dodatečná data o sobě musí dát vědět. Jsou navrženy obě možnosti lokalizace a to že objekt si určí pomoci základny svou polohu, tak i že základna určí polohu objektu.

Kapitola 4

Implementace lokalizačního systému

V úvodu práce bylo stanoveno, že pro radiovou komunikaci použiji softwarové rádio a radiový přijímač zařízení ADALM-PLUTO. V první části této kapitoly se tak zaměřím na možnost komunikace programu se zařízením. V dalších části se pak podívám na implementaci generování vysílaných dat, a způsob jejich ukládání. Následně provedu implementaci vysílání, přijímání a zpracování signálu pro lokalizaci. V poslední části se pak zaměřím na přeložení aplikace a spuštění na zařízení ADALM-PLUTO, které má v sobě umístěný procesor, na kterém běží operační systém linux.

4.1 Komunikace a nastavení zařízení ADALM-PLUTO

Pro komunikaci a ovládání zařízení se požívá knihovna libiio, která slouží jako komunikační rozhraní se zařízením, které používá pro komunikaci IIO (Industrial Input Output) [24][25]. Součásti knihovny je také rozhraní z příkazového řádku, které umožňuje prohledávat adresy pomoci příkazu `"iio_info -s"`. Tímto příkazem získáme seznam všech připojených zařízení podporující IIO a jejich adresy. Následně pak pomoci parametru `"-u"` a adresy zařízení si lze nechat vypsat všechny části zařízení a jejich možnosti nastavení.

Zařízení ADALM-PLUTO lze připojit k počítači pomoci USB rozhraní, ale také může být připojeno do sítě pomoci WiFi nebo LAN. V první řadě se tak musí vytvořit spojení označované jako kontext mezi programem a zařízením. Příklad vytvoření spojení a následné nastavení vysílače je v příloze A.1. Prvním příkazem je vytvoření kontextu, kde parametrem je řetězec s adresou. Dalším krokem je získání zařízení z daného spojení. Jedno fyzické zařízení, ke kterému si vytvoříme spojení může obsahovat více prvků označovaných jako zařízení.

Jedním takovýmto zařízením je `"ad9361-phy"`. Jedná se o čip, který se stará o radiovou komunikaci. V dalším kroku implementace získá program ukazatel na toto zařízení, tak aby ho mohl nastavit. Zbylá část ukázky kódu potom ukazuje jak nastavit jednotlivé kanály a jejich parametry.

4.2 Implementace generátoru zlaté sekvence

Implementaci generátoru zlaté sekvence jsem provedl podle nákresu z obrázku 3.1. Generátor tedy tvoří dvě pole o velikosti jednoho prvku 8 bitů. Do těchto polí se ukládají hodnoty

1 nebo 0, podle aktuálního stavu sekvence. Následně se provádí posun a generování nových znaků. Nové znaky jsou jak výstupní znaky, tedy konkrétní zlatá sekvence, tak i znaky, které se vkládají na začátek polí tvořících generátor.

Při získání jednoho znaku zlaté sekvence se provádí několik operací XOR, které hodnoty vygenerují. Operace XOR je v tomto případě implementována jako součet všech hodnot nad nimiž má být provedena operace. Nad výslednou hodnotou je následně provedena operace modulo hodnotou dvě, pomoci které získáme zbytek po dělení právě hodnotou dva. Následně po vypočítání těchto hodnot se provede samotný posun polí v generátoru. Ten se provádí způsobem, kdy začínám od začátku pole a jako první si do pomocné proměnné ukládám stávající hodnotu prvku pole. Následně se zapíše nová hodnota. Po zapsání nové hodnoty se dočasně uložená hodnota použije jako nová hodnota pro následující prvek pole. Tato operace probíhá ve smyčce, dokud není posunuto celé pole.

Výstupy z generátoru tvoří zlatou sekvenci, které je ukládána do připraveného pole. Jelikož používáme 10 znakové pole pro generování zlaté sekvence, výsledná sekvence je dlouhá 1024 znaků. Výsledné znaky jsou následně převedeny z hodnot 1 a 0 na hodnoty 1 a -1. Převedené hodnoty můžeme následně násobit signálem pokud chceme vytvářet změny fáze a nemusíme provádět porovnání jako kontrolu pro změnu fáze signálu.

4.3 Implementace vysílače

Vysílač potřebuje znát několik parametrů, aby mohl vysílat lokalizační data. Zaprvé potřebuje znát zlatou sekvenci, která tvoří právě lokalizační data. Generování sekvence jsem popsal v předchozí části a definoval jsem, že bude tvořena hodnotami 1 a -1. Dalším z parametrů je velikost jednoho baudu, která bude udaná v sekundách. Baud je hodnota, která definuje délku jednoho přenášeného znaku a všechny přijímače přijímající vysílaný signál musí mít tuto hodnotu stejnou jako vysílač. Z důvodu možnosti změn ve vzorkovací frekvenci, je potřeba znát také tuto hodnotu. Vysílače nebo přijímače pak mohou pracovat na jiné vzorkovací frekvenci a však velikost jednoho baudu zůstane stejná, proto je definována v sekundách. Následný počet vzorků na jeden znak vypočítáme podle rovnice 4.1, kde f_s je vzorkovací frekvence a t_b je délka jednoho baudu právě v sekundách.

$$baud\_size = t_b * f_s \tag{4.1}$$

Na základě získání počtu vzorků pro jeden znak, je upraveno pole se zlatou sekvencí. V poli je tak nyní uložena sekvence po vzorcích. Způsob implementace tak sice zvyšuje využití paměti z důvodu uložení sekvence po vzorcích nikoli po znacích a však snižuje výpočetní složitost, kdy nemusíme pro každý vzorek signálu počítat pozici znaku v poli. Poslední proměnnou musí mít všechny vysílače stejnou a tou proměnnou je modulační frekvence. Jedná se o frekvenci signálu, který bude vysílač vysílat. Vytvoření signálu o dané modulační frekvenci provádí NCO, jehož implementaci provedeme s look-up tabulkou. Do vytvořené tabulky se uloží jedna amplituda signálu. Velikost této amplitudy spočítáme podle rovnice 4.2, kde neznáma f_m určuje právě vzorkovací frekvenci.

$$amplitude\_size = f_s / f_m \tag{4.2}$$

4.3.1 Vysílání pomoci zařízení ADALM-PLUTO

Zařízení ADALM-PLUTO bere na vstupu znaménková 16 bitová čísla a však operace `sin()` a `cos()`, generují hodnoty v intervalu $< -1; 1 >$. Všechny vygenerované hodnoty jsou tedy následně vynásobeny hodnotou $2^{15} - 1$ a následně uloženy do look-up tabulky. Na základě takto vypočítaných hodnot a vytvořených polí s přenášenými daty a jednou amplitudou referenční sekvence, můžeme začít provádět modulaci zlaté sekvence a její následné vysílání. Pro vytvoření vysílaného signálu potřebujeme stanovit vzorky I a Q, kde I bude tvořit kosinusová křivka a Q sinusová křivka. Tyto křivky získáme právě z definované tabulky prvku NCO tak, že kosinusová křivka bude posunuta o čtvrtinu amplitudy.

Po vytvoření všech hodnot pro vysílání alokujeme paměť na zařízení ADALM-PLUTO, do které se budou ukládat vysílaná data. V části jsem uvedl jak vytvořit kontext, tedy spojení se zařízením ADALM-PLUTO a dále jak nastavit jeho parametry, jako jsou nosná frekvence, vzorkovací frekvence a další. V příloze A.2 je příklad základní implementace pro vytvoření a přistupování do paměti pro vysílání.

V první části je získán ukazatel na zařízení, které má na starosti paměť pro vysílání. Ve vytvořeném kontextu s radiovým přijímačem existuje více zařízení. Jedním z těchto zařízení je označené jako `"cf-ad9361-dds-core-lpc"`. Pro získání paměti pro přenos dat se musí vytvořit spojení s touto části zařízení, která pracuje s DA převodníkem pro vysílání, ale také umožňuje spravovat DMA (Direct Memory Access), tedy přímí přístup do paměti. Dale tato část zařízení umožňuje ovládat jádro HDL pro vysílání včetně DDS (Data Distribution Service), tedy kontrolu nad distribucí dat [26].

V následujícím kroku implementace program získá ukazatele na I a Q kanál, aby mohl pomoci těchto kanálů vysílat. Knihovna pro ovládání zařízení je tvořena pro více typů zařízení, které mohou mít více vysílacích kanálů. Proto se musí jednotlivé kanály vybrat a následně aktivovat pomoci operace `iio_channel_enable()`. V případě vysílíní dat se musí nastavit třetí parametr funkce `iio_device_find_channel()` a tím definujeme, že se bude jednat o vysílání na tomto kanále. Následně lze požádat o ukazatel k přístupu do paměti, do které se budou nahrávat data. Jako parametry této operace je velikost paměti a zda se bude jednat o kruhovou paměť či nikoli [25].

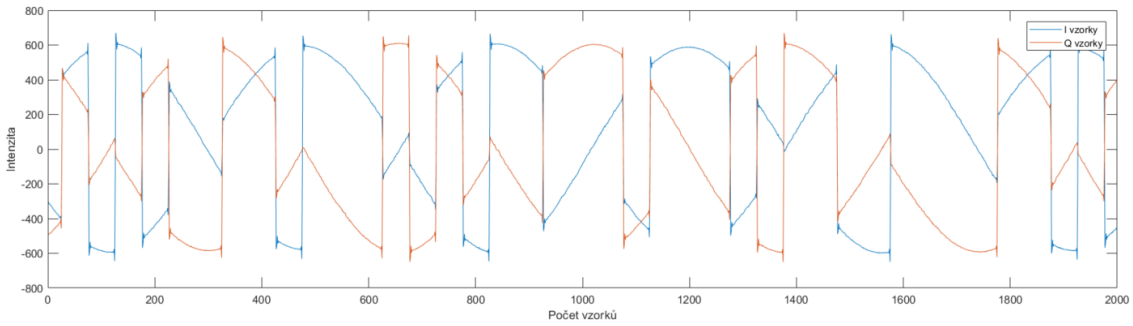
Následně se alokuje paměť pomoci operace `"iio_device_create_buffer()"`. Takto alokovaná paměť se ukládá do struktury `"iio_buffer"`. Součásti této struktury je ukazatel na paměť alokovanou v počítači, ke kterému je zařízení připojené. Dalším prvkem struktury je adresa paměti v zařízení, která byla přidělena právě pro vysílání dat. Tento způsob umožňuje přistupovat a upravovat data v počítači a následně poslat vysílaná data jako jeden blok.

Jelikož knihovna IIO není určena pouze pro jedné zařízení, je nutné dále získat ukazatel na počáteční pozici paměti. Velikost jednoho kroku, tedy vzdálenost mezi jednotlivými daty v paměti a hodnotu ukazatele na konec přidělené paměti. Na základě takto získaných informací můžeme zapisovat data do paměti na správné pozice pomoci smyčky. Vnitřní smyčka prochází jednotlivé prvky paměti a umožňuje do této paměti zapisovat. Po naplnění paměti, která se vytvořila pro komunikaci se zařízením, se smyčka ukončí. Data jsou následně poslána na zařízení pomoci příkazu `iio_buffer_push()`, kde parametr příkazu je právě ukazatel na naplněnou paměť.

4.3.2 Implementace modulace dat

Následně je potřeba provádět naplnění vytvořené paměti, které se provádí pomoci vynásobení hodnoty z pole zlaté sekvence s sinovou a kosinovou hodnotou získané z look-up

tabulky implementovaného prvku NCO. Následně se posuneme na další pozici v paměti, kterou jsme získali od zařízení a pole dat NCO a zlaté sekvence. Pokud dorazíme na konec polí skočí ukazatel zpět na jejich začátek a pokračuje s generování signálu do paměti. Vytvořený vysílaný signál je zobrazen na obrázku 4.1.



Obrázek 4.1: Ukázka vysílaného signálu.

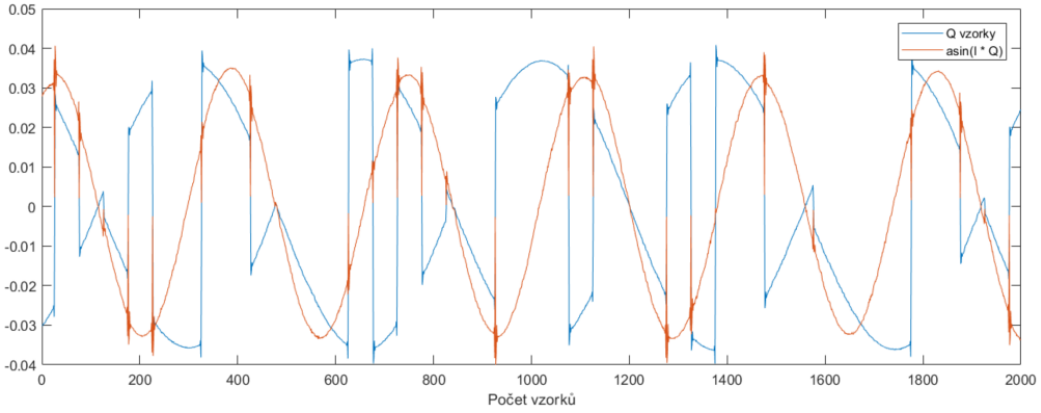
4.4 Implementace přijímače

V první části přijímače musím zajistit získávání radiového signálu. Program musí vytvořit kontext, tady navázat spojení s přijímačem a přijímač nastavit. Postup nastavení jsem popsal v části 4.3.1. Po nastavení parametrů přijímače jako je vzorkovací frekvence, nosná frekvence případně další parametry musí program získat ukazatel na paměť, ve které budou uložena přijímaná data. Postup pro nastavení a přiřazení této paměti pro vysílání jsem popsal v části 4.3.

V případě přijímání dat se musí pouze zvolit jiná část zařízení, oproti vysílání dat. `"cf-ad9361-lpc"` je část, která umožňuje přístup k nastavení přijímání dat, jako alokace paměti, volba kanálů pro příjímání. Při výběru kanálu se musí nastavit třetí parametr na hodnotu 0 tak, aby bylo definováno, že přes dané kanály budeme přijímat data. Následná alokace paměti a přístup do paměti je stejný jako u vysílání dat. Poslední rozdíl je potom v přenosu dat mezi přijímačem a počítačem, kdy použiji funkci `"iio_buffer_refill()"`. Parametrem této funkce je paměť získaná pomoci operace `"iio_device_create_buffer()"`.

4.4.1 Získání modulační frekvence

Data jsou přenášena pomoci modulace signálu BPSK. Pro modulaci je tedy použita určitá modulační frekvence, která je stanovena. Tato frekvence se může lišit z důvodu Dopplerova jevu případně dalších nežádoucích elementů. Pro demodulaci signálu je třeba získat pokud možno přesnou frekvenci. Ze získaných dat pomoci rovnice 3.1, lze získat sinusovou křivku o dvojnásobné frekvenci. Příklad použití rovnice na signálu je zobrazen na obrázku 4.2. Vzorky I a Q je třeba převést tak, aby výsledek jejich násobení byl v intervalu $< -1; 1 >$. Jelikož data jsou uložena ve formátu 16 bitového celého čísla, je třeba všechny vzorky vydělit hodnotou $2^{15} - 1$ a získat, tak data v daném intervalu.



Obrázek 4.2: Graf přijímaného signálu a použití rovnice 3.1.

Signál, který je zobrazen na obrázku 4.2, byl vytvářen jedním zařízením ADALM-PLUTO, které bylo přímo připojené k druhému, které tato data přijímalo. Proto tato vzorová data nemají velké rušení a je vidět, že při změně fáze dochází k různých výkyvům signálu. Jedná se o nežádoucí jev, který můžeme odstranit pomoci filtru dolní propusti. Na takto získaná data použiji následně rychlou Fourierovu transformaci pro získání frekvence.

Rychlá Fourierova transformace

Pro implementaci rychlé Fourierovy transformace (FFT) jsem použil Cooley-Tukey algoritmus [42]. Algoritmus používá rekurzy. Pro zobrazení FFT algoritmu se nejčastěji používá motýlkový diagram [44]. Vstupními daty je pole se signálem, nad kterým provádíme FFT. Data tvoří komplexní čísla. V případě vstupního signálu, který jsem získal ze signálu, tak tento signál bude uložen pouze jako reálná hodnota v poli komplexních čísel. Pro práci s komplexními čísly jsem použil standardní knihovnu jazyka C "complex.h".

Funkce ze začátku rozdělí vstupní data na dvě poloviny a vytvoří tak dvě nové pole s daty. První polovinu dat tvoří prvky z liché pozice v původním poli. Druhé pole následně tvoří hodnoty na sudých pozicích. Následně nad takto vytvořenýma dvěma poli provede rekurzivní volání funkce. Rekurze se opakuje do chvíle dokud ve vstupním poli nejsou pouze dva prvky. Následně je pak prováděna nad vstupními daty diskrétní Fourierova transformace. Takto vypočítané hodnoty, jsou vraceny zpět a používají se dále k výpočtu ve funkci, které vytvořila danou rekurzi.

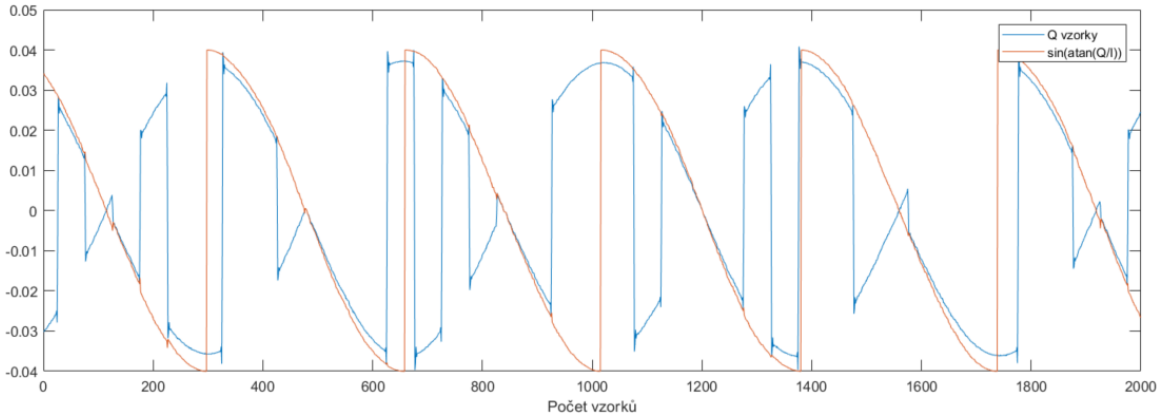
Výslednou frekvenci z provedené rychlé Fourierovy transformace získám pomoci pozice maxima ve výsledném poli. Následně podle vzorce 4.3 provedu převod z pozice v poli na frekvenci signálu. Kde MAX_POS je pozice maximální hodnoty ve výsledném poli po provedení FFT. Dále pak $SAMPL\_FREQ$ je vzorkovací frekvence signálu a FFT_SIZE je velikost vytvořené pole pro FFT Nakonec získanou frekvenci vydělím dvěma a získám tak modulační frekvenci. Dělení dvěma musím použít z důvodu použití vzorce 3.1, který vrací signál s dvou násobnou frekvencí. Frekvence je následně upravena v programu, tak aby se mohla demodulovat data. Demodulace dat je popsána v další části.

$$FREQ = MAX\_POS * (SAMPL\_FREQ/FFT\_SIZE) \tag{4.3}$$

4.4.2 Odstranění Dopplerova jevu

Pro dosažení synchronizace referenční zlaté sekvence s přijímaným signálem je potřeba provést odstranění Dopplerova jevu, se kterým zároveň odstraním z přijímaného signálu modulační frekvenci. Odstranění Dopplerova jevu jsem implementoval jako podělení vstupního signálu s referenční hodnotou, kterou vytváří NCO. Numerický krystalický oscilátor je implementován opět s look-up tabulkou, aby nemusela být pro každý vzorek vypočítávána nová hodnota.

Pro správnou demodulaci je zapotřebí, aby byla frekvence NCO stejná jako frekvence přijímaného signálu. Popis dosažení této vlastnosti je popsán v předchozí části. Dále je také důležité, aby hodnota vydávaná NCO, tedy aktuální úhel signálu, byla stejná jako přijímaný signál v daném okamžiku. Pro získání aktuálního úhlu signálu použiji rovnici 3.3, která nemění fázi podle modulace a tedy fáze signálu. Na obrázku 4.3 je zobrazeno, jakým způsobem kopíruje výsledek rovnice přijímaný signál.



Obrázek 4.3: Graf přijímaného signálu a použití rovnice 3.3 převedené na sinové hodnoty.

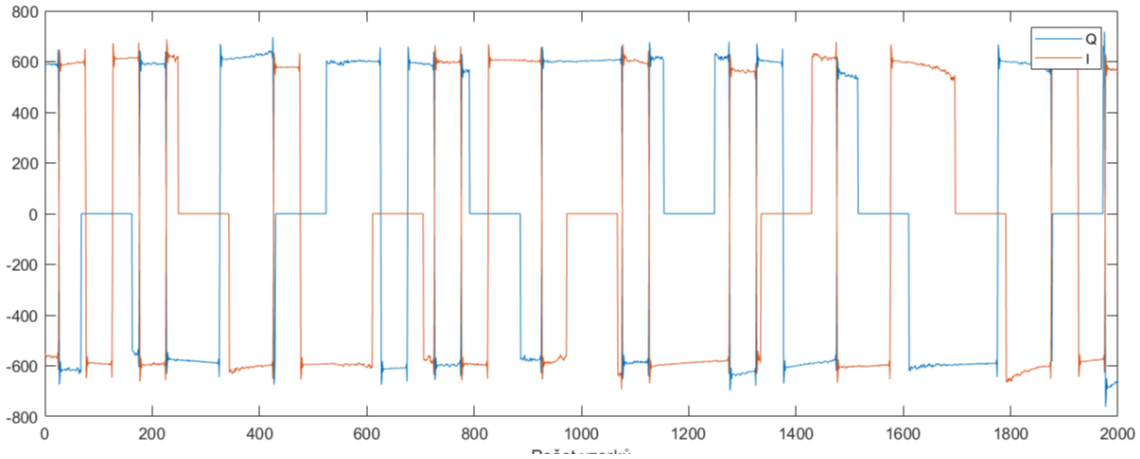
Výsledkem rovnice 3.3 je aktuální hodnota úhlu signálu bez závislosti na fázi signálu. Výsledný generovaný úhel je v intervalu $< -\pi/2; \pi/2 >$, takto získaný úhel tak mění fázi každou půl amplitudy signálu. Takto získaný úhel může být ovlivněn rušením, proto je na získaná data použit filtr dolní propusti. Pomoci rovnice 4.4, se vypočítá pozice hodnoty v look-up tabulce NCO. Tabulka NCO je v intervalu úhlu $< 0; 2\pi >$ a tvoří tak jednu amplitudu. Na základě nově získaného úhlu NEW_ANGLE určíme, pomoci vydělení maximální hodnotou v intervalu NCO a následným vynásobením velikosti tabulky NCO NCO_SIZE pozici úhlu v tabulce. Jelikož používáme filtr dolní propusti je třeba posunout nově získanou hodnotu o polovinu velikosti tohoto filtru NCO_SIZE. Poslední nepopsanou hodnotou NCO_SHIFT je definována ze získané frekvence z předchozí části a určuje posun v NCO tabulce mezi jednotlivými vzorky.

$$NCO\_POS = (\frac{NEW\_ANGLE}{2\pi} * NCO\_SIZE) + (NCO\_SHIFT * \frac{LPF\_SIZE}{2}) \tag{4.4}$$

Nově získaný úhel je třeba ještě porovnat se stávajícím úhlem. Kdy nově vypočítaný úhel posouvám o hodnotu π a měníme tak jeho fázi. Při porovnávání potom hledáme nejnižší odchylku mezi novým a stávajícím úhlem a tím zajistíme to, aby nedocházelo ke změně

fáze referenčního signálu. Tato operace srovnávání úhlu přijímaného signálu a NCO se musí opakovat a to z toho důvodu, že frekvence nastavená u NCO není úplně přesná jako frekvence přijímaného signálu.

Posledním krokem je samotné podělení vstupního signálu hodnotou z NCO. Při implementaci jsem si následně uvědomil, že ve chvíli kdy signál NCO prochází nulou, tedy ve chvíli kdy je v útlumu, dochází k dělení velice malými hodnotami. V extrémním případě pak dělení samotnou nulou. Ve chvíli kdy vzorky I signálu dosahují nuly, tedy jsou v útlumu, druhý vzorek signálu Q dosahuje maximální hodnoty. NCO vydává hodnoty v intervalu $< -1; 1 >$, proto jsem zvolil postup, kdy v případě hodnoty NCO, která se dělí vzorkem I a nebo vzorkem Q se je daná hodnota v intervalu $< -0,4; 0,4 >$ k dělení nedochází a výsledek se automaticky nastaví na hodnotu nula. Odstranění těchto vzorků způsobí díry v datech pro vzorky I a pro vzorky Q. Avšak, jak již jsem napsal, ve chvíli kdy I dochází k útlumu, tak Q dosahuje svého maxima. Proto při spojení těchto dvou hodnot nedochází k celkové ztrátě dat. Ukázka takto upraveného signálu, tedy po odstranění nosné frekvence, je zobrazen na obrázku 4.4.



Obrázek 4.4: Graf přijímaného signálu po odstranění modulační frekvence.

Na výsledném grafu průběhu signálu po odstranění modulační frekvence je vidět, kdy dochází ke změnám znaku. A je vidět, že když signál I dosahuje kladných hodnot pak signál Q dosahuje hodnot záporných. S touto vlastností musíme počítat po čas porovnávání signálu s referenční sekvencí, jejíž postup jsem popsal v následující části.

4.4.3 Synchronizace referenční sekvence a signálu

Poslední částí po zpracování signálu je synchronizace referenční sekvence s přijímaným signálem. Refereční sekvence je uložena jako hodnoty 1 a -1. V návrhu jsem si tuto část na obrázku 3.4 rozdělili na dva bloky. Prvním tímto blokem je DLL [20], který určuje posun referenční sekvence. Na vstupu tohoto prvku je vynásobena referenční sekvence se vstupním signálem. U Q vzorků tyto hodnoty jsou ještě vynásobeny hodnotou mínus jedna, aby došlo ke změně fáze přijímaného signálu pro vzorky Q, tak jak jsem popsal na konci předchozí části. Pozice vzorků z referenční sekvence jsou od sebe vzdálené stejně.

Na vstupu DLL jsou tedy vynásobené vzorky signálu s referenční sekvencí. Prvky označené jako LPF, které jsou umístěny před blokem DLL. DLL nezpracovává hodnotu pro každý vzorek, proto LPF jsou implementovány jako součet všech vstupních hodnot. Tento

součet se následně vydělí počtem vzorků, které byly sečteny a výsledná hodnota je jejich výstupem. Průměr se tak počítá jednou za několik vzorků a to ve chvíli, kdy DLL potřebuje tato data pro výpočet.

V implementaci jsem zvolil hodnotu dvaceti znaků, tedy pokud velikost jednoho baud bude deset vzorků, pak DLL provede výpočet každých 200 vzorků. Při každé provedení DLL vypočítá zda se má referenční sekvence posunout o jedno doleva nebo doprava, na základě hodnoty diskriminátoru, kterou definuje vzorec 3.4. O jednotlivých posunech je také informován blok nazvaný jako `"Controler"`.

`"Controler"` se spouští každý vzorek, kdy na vstupu má hodnotu I a Q porovnanou se středním vzorkem z referenční sekvence. Provede součet výsledné hodnoty I a invertované hodnoty Q, tak aby seděli fáze signálu pro každý znak. Následně zjistí, zda součet těchto hodnot je větší jak nula. Pokud tomu tak je, znamená to, že signál a referenční sekvence vysílají stejný znak a dojde k inkrementaci, tedy přičtení hodnoty jedna, k vnitřní proměnné. Pokud by hodnota byla menší než nula pak referenční sekvence má opačnou fázi než přijímaný signál a dojde k dekrementování vnitřní proměnné. Tento blok také přijímá od DLL zda byla a o kolik posunuta referenční sekvence, a tyto posuny si ukládá.

Následně prvek po několika iterací DLL spustí svůj hlavní výpočet. V případě této implementace je to nastaveno na 200 iterací části DLL. Pří provedení jedné iterace části `"Controler"` se vypočítá průměr hodnoty, kterou si prvek inkrementoval a nebo dekrementoval podle shody signálu s referenční sekvencí. Pokud tento průměr tvoří 0,8 znamená to, že bylo dosaženo synchronizace signálu a referenční sekvence 80%. Po vypočítání této hodnoty se proměnná nastaví zpět na nulu a výpočet začíná znovu. Hodnota však může dosahovat také opačné, tedy záporné hodnoty. Pak synchronizace je stejná, jenom byla obrácená fáze NCO. Po dosažení hodnoty synchronizace 80% prvek začne vydávat výsledek posunu sekvence, což znamená o kolik byla posunuta referenční sekvence a tedy i přijímaný signál. V případě, že nebyla dosažena synchronizace dojde k jednorázovému posunu referenční sekvence o jeden znak sekvence, tedy o velikost baudu. Tím se zajistí, že se multikorelátor nezastaví na lokálním maximu, ale dosáhne globálního maxima.

Prvek vydá hodnotu s každým výsledkem DLL. Dále tak můžu vytvářet průměr z několik získaných hodnot. Sice tímto postupem snížím obnovovací frekvenci lokalizace, nicméně dostaneme vyšší přesnosti posunu sekvence.

4.5 Ovládání aplikace

Aplikace, která provádí vysílání nebo příjímání signálu, případně obojí lze nastavit pomoci vstupního souboru ve formátu XML. Pro přečtení souboru jsem v implementaci použil knihovnu libxml, která je potřebná i pro samotné přeložení a nainstalování knihovny libiio. Veškeré zpracování dat provádí knihovna readXLM.h, kterou jsem implementoval.

V tomto souboru je definován celý lokalizační systém. Kořen stromové struktury tvoří název "locSystem", které obsahuje následující elementy. "device", jedná se o element jehož hodnotou je adresa radiového přijímače. Dalšími elementy pak jsou "frequency"a "samplingFreq", tyto elementy určují nosnou a vzorkovací frekvenci. Hodnoty lze vkládat společně s jednotami až po GHz bez mezer.

Následující element má označení "transmitters", a tento element obsahuje definici všech vysílačů v daném lokalizačním systému. Dále také obsahuje prvky jako "carrier"pro stanovení modulační frekvence a "baud"pro stanovení délky jednoho přenášeného znaku sekvence. Následně jsou ve struktuře definované jednotlivé vysílače "transmitter". Tyto vysílače mají atribut s názvem "type", který může být nastaven na hodnoty "reference", "transmit"nebo

"none". Jednotlivé parametry atributu určují jestli zařízení je daným vysílačem, tedy že má tento signál vysílat a také jestli má nějaký referenční vysílač, se který se má synchronizovat. V případě "none"pak znamená, že vysílač tento signál nemusí zpracovávat.

Každý vysílač následně obsahuje další elementy, které ho charakterizují. Jako je jeho poloha v rámci lokalizačního systému nebo jakou sekvenci daný lokalizační systém vysílá. Co se týká definice sekvence je definována v elementu "goldCode", který obsahuje dvě celá čísla oddělena mezerou v intervalu od 1 do 10. Tyto hodnoty určují z jakých pozic v posuvných registrech generátoru sekvence budou brány hodnoty pro generování.

4.6 Překlad na zařízení ADALM-PLUTO

Zařízení ADALM-PLUTO obsahuje čip Xilinx XC7Z010, kde struktura a vlastnosti zařízení jsou popsány v části 2.3.3. Čip obsahuje jak FPGA tak i ARM procesor, na kterém běží operační systém založený na Linuxové platformě. K zařízení se tedy lze připojit pomoci ssh a přenášet data pomoci scp. Zařízen má však malou paměť pro ukládání velkého množství programů a operační systém se načítá na paměť RAM. Tedy při uložení dat například do kořenového adresáře linuxu, na zařízení budou všechna tato data po restartu zařízení vymazána.

Aplikaci lze přeložit pomoci programu Xilinx SDK, který má překladač "arm-linux-gnueabihf-gcc". V případě použití starší verzi systému na zařízení nebo novějšího překladače je třeba také upravit název interpretu, který bude aplikaci spouštět. Tuto změnu interpretru lze změnit pomoci parametru napsaného v příkladu 4.1, společně s dalšími parametry pro překlad.

```
INTERPRET=-Wl,-dynamic-linker,//lib/ld-linux.so.3
LIB= -liio -lxml2 -lm
```

Příklad 4.1: Změna interpreta a propojení dynamických knihoven.

Dalším potřebným krokem pro správně přeložení je zapotřebí mít dynamické knihovny. Potřebné knihovny jsou libxml2, libiio, libusb a další knihovny, které požadují tyto definované. Je třeba tyto knihovny mít přeložené pro procesory ARM. Pro jednotlivé knihovny lze nalézt a stáhnout zdrojové kódy, které se následně musí přeložit. Pro překlad těchto knihoven tak je zapotřebí petalinux. Druhou možností je přeložené knihovny stáhnout ze zařízení ADALM-PLUTO, kde jsou umístěny ve složce `/usr/lib`.

Pro správné propojení aplikace se zařízením se používá stejného postupu, jako když aplikace běží na počítači. Avšak je třeba změnit adresu zařízení při vytváření kontextu a to na "local:". Další věcí, kterou je třeba zkontrolovat pro správný běh programu je verze systému, který na procesoru běží. Verze systémů musí být aktuální, tedy minimálně v0.30 neboť předchozí verze mají problém s funkcemi sinus a kosinus. V tomto případě tyto funkce hodnoty nepočítají a vrací pro sinus hodnotu 1 a kosinus hodnotu 0.

4.7 Spouštění aplikace na zařízení ADALM-PLUTO

Zařízení ADALM-PLUTO, jak již jsem v předchozích částech zmínil procesor typu ARM, na které běží operační systém linux. Na zařízení lze spustit aplikaci bez nutnosti připojení k počítači. Zařízení obsahuje dva porty Micro-USB. Jeden z portů slouží pouze jako napájecí v případě, že program bude běžet na zařízení a není tak potřeba počítač pro výpočty. Druhý

port slouží pro propojení počítače a zařízení. Přes oba porty se dá zařízení napájet, proto pokud zařízení je připojeno k počítači nepotřebuje zvlášť napájení.

Druhý z portů USB, který slouží pro komunikaci, je typu OTG (On-The-Go). Jedná se o definici, která umožňuje zařízení komunikovat jako hostitelské zařízení. Přes toto USB lze připojit další zařízení jako flash disk, WiFi adaptér, LAN a mnoho dalších. Zařízení ADALM-PLUTO se chová jako hostitelské zařízení a v případě připojení do sítě přes WiFi nebo LAN se toto zařízení lze ovládat na dálku. K zařízení se lze připojit i v případě kdy je připojeno přímo k počítači pomoci příkazu ssh. Zařízení obsahuje pouze jednoho uživatele a tím je "root", a jako heslo "analog".

Na zařízení lze nakopírovat aplikaci, kterou následně skrz příkazový řádek spustíme. Druhou možností spuštění aplikace je aplikaci nahrát na USB flash disk, který následně připojíme k zařízení. Po připojení USB disku zařízení spouští skript `"/lib/mdev/automounter.sh"` [27]. Tento skript prochází připojený USB disk a spouští všechny spustitelné soubory s názvem runme[0-9].sh případně runme[0-9]. USB disk musí být naformátován na ty FAT32, případně ext4.

Zařízení dále obsahuje také jednu LED diodu, pro indikaci stavu zařízení. Typ blikání diody lze upravit pomoci skriptu na několik předdefinovaných typu, jako jsou například vytížení procesoru, stálé svícení, zhasnutá a další možnosti, tak jak je uvedeno v prvním řádku příkladu 4.2. Této vlastnosti lze využít pro kontrolu, zda program a skript uložený na USB disku je spuštěný, případnou indikaci chyby. Při implementaci skriptu, který upravuje funkci diody, následně provede skript zkopírování programu a konfiguračního souboru do zařízení a následně spustí jej.

```
echo cpu > /sys/class/leds/led0\:green/trigger

cp /media/sda1/transmiter /root/
cp /media/sda1/config.xml /root/

chmod +x /root/transmiter

/root/transmiter /root/config.xml
```

Příklad 4.2: Příklad skriptu pro kopii programu a jeho následné spuštění.

4.8 Dílčí závěr

V této kapitole práce jsem se v první části zaměřil na způsob nastavení zařízení ADALM-PLUTO pro vysílání a přijímání radiového signálu pomoci knihovny IIO. Při nastavení zařízení se vytváří kontext, tedy spojení mezi aplikací a zařízením. Zařízení obsahuje více částí, které ovládají různé možnosti zařízení, jako například nastavení nosné a vzorkovací frekvence, dále části přes které lze ovládat DMA a velikost paměti pro přijímání a vysílání dat. Dále jsem definoval, že signál je ukládán na zařízení v 16 bitovým znaménkovým celým číslem.

V druhé části kapitoly jsem se soustředil na samotou implementaci programu pro vysílání a příjímání signálu. V implementaci jsem tak popsal, jaký způsobem se provádí modulace signálu BPSK dále jakým způsobem jsou uložená data tedy sekvence, kterou zařízení vysílá. Dále při implementaci přijímání a zpracování signálu jsem popsal implementaci a

funkci multikorelátoru. Součásti multikorelátoru je část, která provádí odstranění modulační frekvence ze signálu. U této části je velice důležité si dát pozor na správné získávání informací. V případě změny fáze by došlo k otočení jednotlivých stavů informací a tak ztrátě synchronizace.

V poslední z části kapitoly jsem se zaměřil na možnosti spuštění programu na zařízení ADALM-PLUTO, které může pracovat samostatně bez nutnosti připojení k počítači. Pro spuštění aplikace na zařízení je potřebné správně aplikaci přeložit. Dále pak firmware zařízení se stále vyvíjí, tedy až při testování byla vydána nová verze firmwaru, která opravovala chybu s výpočty hodnot sinus a kosinus.

Kapitola 5

Přesnost lokalizačního systému

Jedním z nejdůležitějších aspektů lokalizačního sytému je jeho přesnost. Přesností je myšlena odchylka pozice, která je udána lokalizačním systémem od skutečné pozice. Dalším důležitým aspektem je také jeho pokrytí. Pokrytím je myšleno, na jaké ploše jsme schopni určit svou polohu pomoci lokalizačního systému. V této kapitole se zaměříme na přesnost, která je závislá na vzorkovací frekvenci přijímače. Co se týče pokrytí prostoru lokalizačním systémem, tak tuto plochu můžeme zvětšovat použitím vyššího počtu vysílačů. Dále pokrytí lze ovlivnit požitím různých druhů antén a různým příkonem.

5.1 Stanovení přesnosti lokalizačního systému

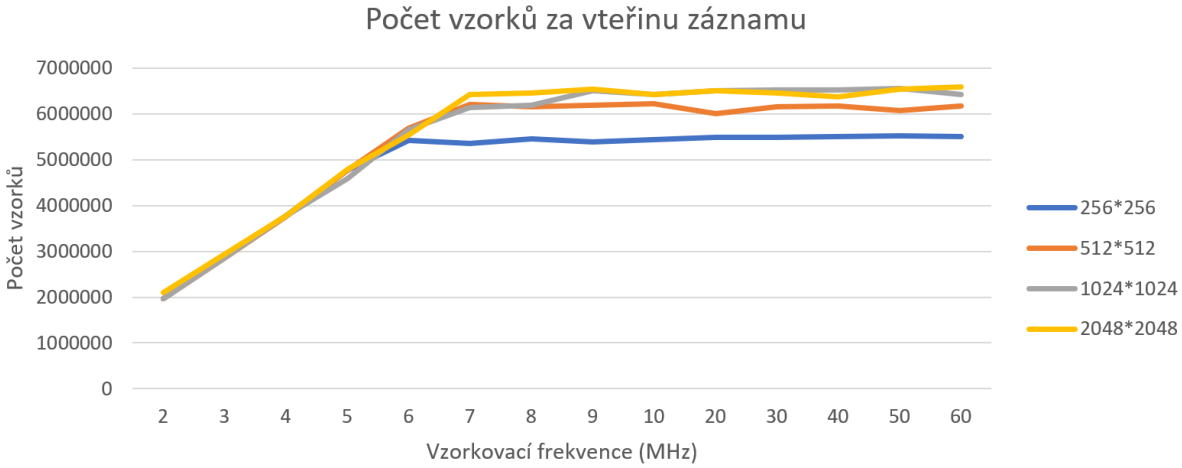
Jak jsem již v úvodu kapitoly zmínil, přesnost lokalizačního systému lze určit za pomoci vzorkovací frekvence. Pro určení přesnosti jsem zvolil tuto metodu z důvodu dosažení nízké vzorkovací frekvence u vysílačů a přijímačů. Přijímaný případně vysílaný signál lze zpracovávat dvěma způsoby. Prvním ze způsobů je použít zařízení ADALM-PLUTO jako samostatnou výpočetní jednotku. Zpracování signálu bude provádět samotné zařízení, které bude potřeba pouze napájet. Druhou z možností je pro zpracování signálu použít jiné výpočetní zařízení, například počítač.

5.1.1 Použití počítače pro zpracování signálu

V případě použití počítače pro zpracování signálu se musí data přenášet mezi zařízením a počítačem přes port Micro-USB 2.0. Přes toto rozhraní lze teoreticky přenášet rychlostí až 480Mb/s (60 MB/s). Zařízení ADALM-PLUTO posílá vzorkovaný signál jako I a Q vzorky, kde jeden vzorek má velikost 16 bitů. Pro jeden vzorek signálu je zapotřebí přenést 32 bitů. Na základě těchto informací jsme schopni určit teoretickou maximální vzorkovací frekvenci, které můžeme dosáhnout, abychom neztráceli data z důvodu, že je nestíháme přenášet mezi zařízením a počítačem. Maximální vzorkovací frekvenci jsem tak vypočítal na 15MHz. Jedná se však o teoretickou přenosovou rychlost, protože v přenosu není zahrnuta žádná další komunikace mezi zařízením a počítačem, případně jiných zařízeních připojených na sběrnici USB. V případě, kdy jsou na zařízení všechny bloky paměti určené pro přenos dat mezi zařízením a počítačem plné, zařízení začne přepisovat již uložená data, která počítači ještě neposlal. Z tohoto důvodu potom dochází k mezerám v datech, které vedou až k nedosažitelné synchronizaci signálu s referenční sekvencí.

Pro stanovení skutečné maximální vzorkovací frekvenci, kterou jsme schopni přenášet data přes USB bez jejich ztráty jsem provedl měření. Měření bylo prováděno pomoci jedno-

duchého programu, který nastaví přijímač na danou vzorkovací frekvenci. Následně program stahuje data z přijímače a však neprovádí s daty žádné zpracování. Program pouze měří vzorků signálu jsme při dané vzorkovací frekvenci schopni přijmout. Výsledek měření jen zobrazen na obrázku 5.1.



Obrázek 5.1: Počet nahraných dat při zvyšující se frekvenci pro rozdílné velikosti paměti.

V grafu je popsáno, kolik dat bylo přeneseno za jednu vteřinu při zvyšující se vzorkovací frekvenci. Jednotlivé křivky grafu pak určují velikost definovaného bloku paměti. Z výsledků testu je zřejmé, že při použití menší velikosti bloku paměti je nižší maximální vzorkovací frekvence, kterou jsme schopni přenést data bez ztráty. To je ovlivněno vyšším procentem řídících signálů oproti datům při komunikaci s přijímačem. Dále pak z výsledku je viditelné, že maximální vzorkovací frekvence, kterou jsme schopni přenášet data bez jejich ztráty je kolem 7 MHz.

Na základě takto stanovené maximální vzorkovací frekvence, které je možné dosáhnout lze určit teoretickou přesnost systému. Radiový signál se šíří v prostředí rychlostí světla. Pokud vydělíme rychlost světla, maximální dosažitelnou vzorkovací frekvencí dostaneme výsledek teoretické přesnosti lokalizačního systému. V tomto případě je to přibližně 42,8 metrů. Zároveň tato hodnota určuje minimální vzdálenost mezi jednotlivými vysílači, tak aby rozdíl sekvence vysílané jednotlivými vysílači byl alespoň jeden vzorek.

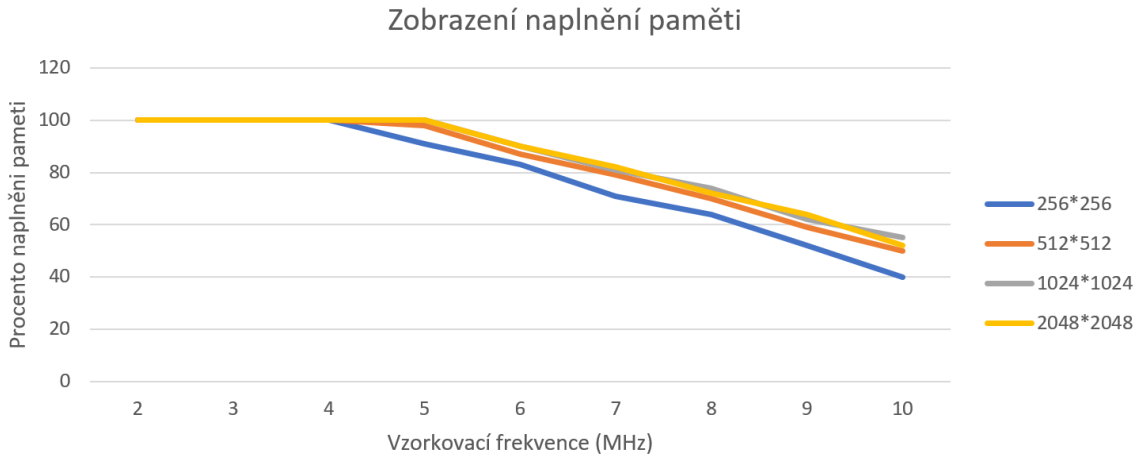
V této části jsem tedy stanovil teoretickou přesnost systému při použití počítače jako výpočetního zařízení, které zpracovává přijímaný signál. Jako nejužší místo jsem zjistil, že se jedná o komunikaci mezi zařízením a počítačem, pro které je použito rozhraní USB 2.0. V další části se pokusíme dosáhnout větší přesnosti a to pomoci odstranění problémové části. Použijeme tedy pro výpočet samotné zařízení ADALM-PLUTO.

5.1.2 Použití zařízení ADALM-PLUTO pro zpracování signálu.

Jak již jsem zmínil v předchozích částech této práce zařízení ADALM-PLUTO obsahuje ARM procesor, na kterém běží operační sytém linux. Je proto možné vytvořenou aplikaci přeložit a spouštět čistě na zařízení. Pro běh aplikace použijeme dvou jádrový ARM procesor, který má pracovní frekvenci 766MHz,

Se zvyšující se vzorkovací frekvencí se zvyšuje i výpočetní složitost, kdy musí program naplnit paměť, ze které následně probíhá vysílání dat. Pro testování jsem použil jednoduchý program, který vysílá sinusoidu bez změn fáze. Následně jsem pak použil pro testování

druhé zařízení, které bylo připojené k počítači a zaznamenávalo všechna data. Postupem měření pak bylo zvyšovat vzorkovací frekvenci na vysílači a pomoci přijímače zjistit hraniční hodnotu. Při zvyšující se vzorkovací frekvenci procesor nestíhal naplnit paměť, ze které se data vysílala a tak vznikaly v signálu mezery. Graf výsledků je zobrazen na obrázku 5.2.



Obrázek 5.2: Procento zaplnění paměti při zvyšující se vzorkovací frekvenci.

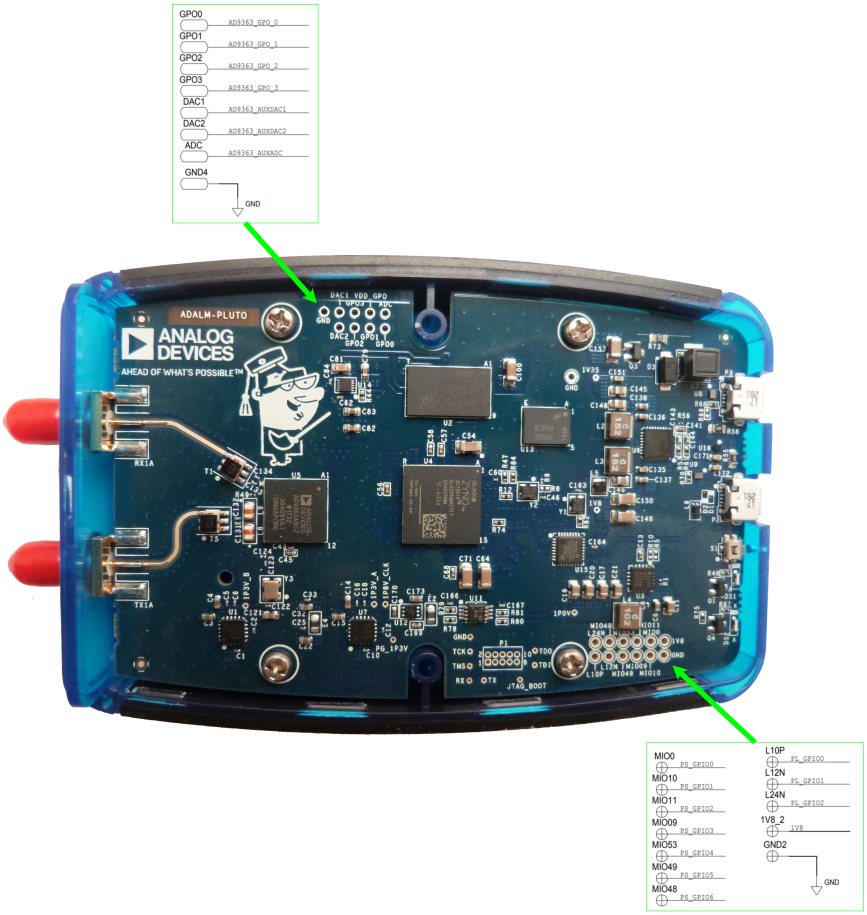
Provedl jsem několik testů, ze kterých jsem vytvořil průměrnou hodnotu reprezentující procento zaplnění paměti. Tedy měřil jsem velikost dat a velikost mezer, které jsem následně přenesl do grafu. Toto měření jsem provedl pro různé velikosti paměťových bloků na straně vysílače. Na základě výsledků je vidět, že procesor nestíhá plnit data již při 6MHz vzorkovací frekvence. Jako testovací data jsem vkládal sinusoidu bez žádných změn fáze. V případě, že bychom na zařízení chtěli zpracovávat ještě modulaci, případně další operace v případě přijímání signálu, maximální možnou vzorkovací frekvenci bez ztráty dat by to snižovalo.

5.2 Zvýšení přesnosti lokalizačního systému

V předchozí části této kapitoly jsem definoval přesnost vytvoření implementace lokalizačního systému. Přesnost implementace jsem stanovil teoreticky z důvodu nízké maximální dosažené vzorkovací frekvenci, tedy přesnosti implementace se pohybuje v desítkách metrů. Implementaci jsme otestoval dvěma způsoby. Jedním ze způsobů bylo využít jiné zařízení pro zpracování signálu a druhou z možností bylo provádět výpočty přímo na zařízení. V této části se tak zaměřím na to jak zvýšit přesnost lokalizačního systému.

5.2.1 Zvýšení datového přenosu

Při implementaci, kdy signál zpracovávalo jiné výpočetní zařízení než samotné zařízení ADALM-PLUTO, jsem pomoci testů stanovil, že k omezení maximální dosažitelné vzorkovací frekvence dochází vlivem přenosu mezi zařízením a počítačem. Omezujícím faktorem je tak přenos dat přes USB 2.0, kterým je zařízení připojené k počítači. Pro zvýšení maximální dosažitelné vzorkovací frekvence je třeba zajistit vyšší přenosovou rychlost mezi zařízením a počítačem, který bude následně zpracovávat lokalizační data. Pro zobrazení možností datového přenosu se musíme podívat na desku zařízení 5.3 a na schéma zařízení [28].



Obrázek 5.3: Fotografie zařízení se zvýrazněnými výstupními piny.

Na desce zařízení na obrázku 5.3 jsou zvýrazněny dvě sady pinů. Vrchní sada pinů, které mají označení `GPO0-3` a `GPO1-2` jsou piny přímo připojené na čip `AD9363`. Jedná se o čip, který provádí přijímání a vysílání signálu. Jednou z možností je připojit výpočetní zařízení přímo k radiovému transceiveru. Pro možnosti přenosu dat přes tyto piny se musí zjistit ze specifikace čipu transceiveru [29].

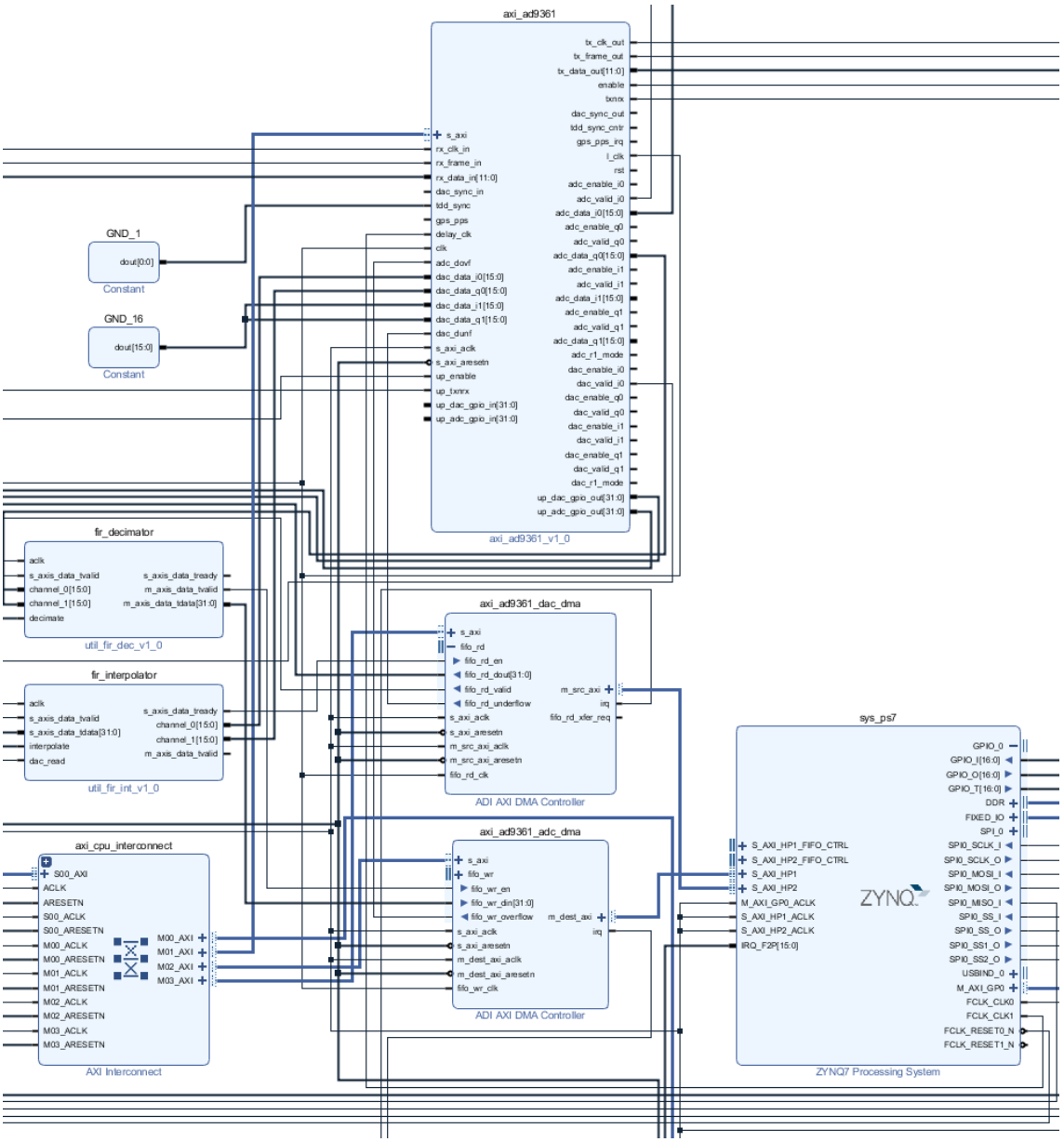
Druhou z možností je další zvýrazněná sada pinů na desce. Jedná se o piny, které jsou připojeny přímo k čipu Xilinx XC7Z010. Tento čip obsahuje FPGA a dvou jádrový ARM procesor. Následně můžeme zařízení ovládat pomocí aplikace a knihovny IIO a upravit logiku FPGA, aby zajistila komunikaci přes tyto piny a odesílala všechna data z přijímače, případně přijímala všechna data na vysílač. Pro komunikaci můžeme pomoci těchto pinů použít sériovou komunikaci jako například SPI, I2C, CAN a další.

5.2.2 Akcelerace zpracování dat

Při druhé možnosti testování jsem použil vestavěný ARM procesor. Tento procesor pouze vytvářel data, která následně vysílal. Testování však ukázalo, že samotný procesor nezvládá vytvářet data dostatečně rychle a dochází tak k mezerám ve vysílaném signálu. Pro zvýšení vzorkovací frekvence potřebujeme zvýšit rychlost generování dat, která následně budou vysílána. Jednou z možností pro akceleraci výpočtu přímo na zařízení je využít druhé jádro procesoru. Tímto způsobem bude potřeba zaručit synchronizaci mezi jádry a v případě

vysílání a přijímaní dat zároveň bychom museli přepínat mezi jednotlivými aplikacemi. Tento způsob nepřináší vysoké zrychlení.

Druhou z možností je provést akceleraci pomoci FPGA. Pro zobrazení možnosti implementace programu pomoci FPGA je popsat jednotlivé bloky implementace, která je na čipu nahrána. Zobrazení jednotlivých implementovaných částí a jejich propojení je zobrazeno na obrázku 5.4.



Obrázek 5.4: Zobrazení bloků FPGA pomoci Xilinx Vivado.

První část označená jako `"sys_ps7"` je ARM procesor umístěný na čipu. Na pravé straně tohoto bloku jsou zobrazeny vstup výstupní piny zajišťující komunikaci pomoci sériové sběrnice SPI. Přes tuto sběrnici komunikuje procesor s přidanou paměti flash a dále také s čipem zajišťující komunikaci s počítačem přes rozhraní USB. Z druhé čísti bloku pak jsou

připojeny prvky s názvem `"axi_ad9361_dac_dma"` a `"axi_ad9361_adc_dma"`. Propojení mezi procesorem a jednotlivými bloky komunikují pomoci protokolu AXI [30].

Tyto bloky zajišťují přímý přístup do paměti pro nahrávání a čtení dat, která následně jdou na vysílač. První z těchto dvou bloků umožňuje přímý přístup do paměti pro vysílání dat a provádí tak přímé čtení z paměti. V případě druhého bloku je to potom naopak, kdy se jedná o přímý přístup do paměti pro zapisování a je zde připojena strana pro přijímač. Posledním blokem je samotný přijímač, který je označen jako `"axi_ad9361"`. Tento blok se stará o spojení mezi přijímacím čipem a FPGA. V případě vytvoření implementace zpracování a vysílání signálu na FPGA lze tuto implementaci napojit právě mezi blok spravující přijímač a bloky starající se o přímý přístup do paměti.

5.2.3 Zvýšení maximální vzorkovací frekvence

Zařízení ADALM-PLUTO dosahuje maximální vzorkovací frekvence 60 MHz. V této práci se mi zpracování signálu na této frekvenci nepodařilo dosáhnout. V případně dosažení vzorkovací frekvence se teoretická přesnost sníží na přibližně 5 metru. Čip `AD9361`, který v zařízení slouží jako přijímač a vysílač, má maximální vzorkovací frekvenci vyšší. Součásti blokového diagramu 5.4 jsou prvky označené jako `"fir_decimator"` a `"fir_interpolator"`.

První ze zmíněných bloků slouží pro vysílání dat a druhý pro jejich přijímání. Podle blokového schématu je vytvořen ze dvou datových linek na jedné straně a pouze jednou na straně druhé. Jejich úkolem je spojení nebo rozdělení dat. Další úkolem, které tyto prvky mají je i umělé snížení vzorkovací frekvence v případě přijímání dat. Jedním z důvodů těchto prvků v návrhu je právě snížení vzorkovací frekvence, tak aby se data mohla přenášet přes USB rozhraní. Při odstranění těchto prvků lze dosáhnout vzorkovací frekvence až 245,76MHz [31]. Při dosažení této vzorkovací frekvence se teoretická přesnost systému zvýší na 1,25 metrů. Další zvýšení přesnosti můžeme provést pomoci multilaterace, která je popsána v části 2.2.2.

5.2.4 Dílčí závěr

V této kapitole jsem se zaměřil na testování implementace a stanovení přesnosti lokalizačního systému. Testováním jsem zjistil, že dosahuji přesnosti lokalizace v desítkách metrů. Jedním z důvodů byla omezená přenosová rychlost při použití USB 2.0 rozhraní, kdy zařízení pracuje pouze jako přijímač a zpracování signálu provádí počítač. Při přijímání dat se začala přepisovat paměť v zařízení a vznikali tak mezery v přijímaném signálu. Pro dosažení vyšší vzorkovací frekvence jsem vytvořil aplikaci, která je spuštěná na zařízení tak, aby nemusela být prováděna komunikace přes USB rozhraní. V tomto případě nestačí výpočetní rychlost ARM procesoru, který je na zařízení.

V druhé polovině kapitoly jsem se proto zaměřil na možnosti jakými odstranit tyto nedostatky a dosáhnout vyšší vzorkovací frekvence. Jednou z možností je pro přenos dat mezi počítače a zařízením použít jiné rozhraní než USB. K tomuto účelu zařízení obsahuje několik vstup výstupních pinů, které lze použít pro přenos dat. Data ze zařízení mohu získávat buď bez použití procesoru zařízení a to tím způsobem, že budu data brát přímo z přijímače. Nebo v druhém případě pak můžeme vytvořit na procesoru zařízení implementaci komunikačního protokolu, pomoci které budu data posílat ze zařízení do počítače vyšší přenosovou rychlostí.

Druhou možností je zrychlení zpracování dat na samotném zařízení. Pro akceleraci můžeme implementaci paralelizovat, tak aby byl procesor plně vytížený. Další možností akcelerace je provedení implementace na FPGA. Součástí FPGA návrhu jsou také bloky, které

uměle snižují vzorkovací frekvenci přijímače. Pro dosažení větší přesnosti mohou být tyto části implementace o odstraněny.

Kapitola 6

Závěr

V práci jsem provedl analýzu již existujících lokalizačních systémů jak aktivních, tak i pasivních. Jelikož potřebuji aby objekt znal vlastní polohu, zvolil jsem pro lokalizaci aktivní lokalizační systém, kde je potřeba vytvořit vlastní lokalizační základu tvořenou vysílači. Proto jsem se podíval na různá zařízení, která by pro tuto lokalizaci šla použít. Nakonec jsem zvolil implementaci pomoci SDR tedy softwarově definovaného rádia a zařízení ADALM-PLUTO. Softwarově definované rádio umožňuje vysílat, ale i zpracovávat různé modulace signálu bez nutnosti úpravy hardwarové části. Tato vlastnost umožní otestovat více možností komunikace, ale i případně různé možnosti lokalizace.

Podíval jsem se na výpočet pro určení polohy, ze kterého je definován i minimální počet vysílačů a také jak zvýšit přesnost výpočtu polohy. Zvoleným způsobem výpočtu polohy zařízení je následně definováno, že všechny vysílače v lokalizačním systému musí být synchronizovány. Pro přesné určení polohy musí tyto vysílače vysílat počátek sekvence, která tvoří lokalizační data, zároveň.

V další části jsem se soustředil na možnosti přenosu dat mezi transceivery, abych mohl provádět lokalizaci. Pro přenos jsem zvolil modulaci typu BPSK a lokalizační data budou tvořit zlaté sekvence (Gold code), tak jak je tomu u systému GPS. Podle GPS systému jsem také popsal možnost zpracování signálu, tedy použití multikorelátoru. Multikorelátor zpracovává signál a na základě dat ze signálu posouvá vlastni referenční sekvence. Na základě posunu referenční sekvence z více vysílačů mohu stanovit časový rozdíl mezi nimi. Následně jsem popsal jak vytvořit vlastní lokalizační systém. Co jednotlivé orientační body systému, které tvoří vysílače musí splňovat, abychom mohli použít lokalizační metodu TDOA.

V následující kapitole jsem se zaměřil na samotnou implementaci aplikace pro vysílání a následně zpracování signálu. Na začátku implementace jsem popsal jakým způsobem lze nastavit radiový přijímač a jak zajistit komunikaci mezi přijímačem a programem. Následně jsem se zaměřil na implementaci generování zlaté sekvence, která tvoří lokalizační data. Dále jsem stanovil, jak takto data budou uložena pro další implementaci. V následující části jsem provedl implementaci vysílání signálu, tedy provedení modulace BPSK pro zlatou sekvenci. Druhým z programů, které jsem implementoval, je zpracování přijímaného signálu. Součástí aplikace je tak demodulace signálu a provádění synchronizace referenčních dat s přijímaným signálem. Na závěr kapitoly jsem se pak podíval na možnost přeložení a běh aplikace na zařízení ADALM-PLUTO.

Na závěr práce jsem se zaměřil na stanovení přesnosti vytvořené implementace lokalizačního systému. Provedl jsem testování, abych stanovil maximální vzorkovací frekvenci. Na základě výsledků jsem nakonec stanovil přesnost teoreticky, a to z důvodu dosahování nízké přesnosti vlivem nízké vzorkovací frekvence. Zaměřil jsem se na důvod proč nejde

dosáhnout vyšší vzorkovací frekvence. Pří implementaci, kdy zařízení pracuje pouze jako přijímač a zpracování signálu provádí pouze počítač, jsem zjistil, že při nastavení vyšších vzorkovacích frekvencí je problém nízké propustnosti USB rozhraní. V druhém případě, tedy kdy program běží na zařízení jsem zjistil, že při zvyšování vzorkovací frekvence nestíhá procesor generovat vysílaná data a vznikají tak mezery v signálu. Na závěr kapitoly jsem se zaměřil na možnosti jakými tyto problémy odstranit a dosáhnout vyšší přesnosti.

Literatura

[1] 802.11 fundamentals: Modulation. mekari.com [online], navštíveno 30.04.2018.
URL https://documentation.meraki.com/MR/WiFi_Basics_and_Best_Practices/802.11_fundamentals%3A_Modulation

[2] Pasivní sledovací systém Věra. Ministerstvo obrany [online], 2001, navštíveno 18.02.2018.
URL http://www.acr.army.cz/technika-a-vyzbroj/protivzdusna-obrana/pasivni-sledovaci-system-vera-3504/

[3] *A Software-Defined GPS and Galileo Receiver*. Birkhäuser Boston, 2007, ISBN 978-0-8176-7390-4.

[4] GPS Signal Plan. Navipedia [online], 2014, navštíveno 08.04.2018.
URL http://www.navipedia.net/index.php/GPS_Signal_Plan

[5] REAL TIME PASSIVE RADAR RUNNING NATIVE ON THE ADALM-PLUTO ARM CPU. rtl-sdr.com [online], 2018, navštíveno 08.04.2018.
URL https://www.rtl-sdr.com/real-time-passive-radar-running-native-on-the-adalm-pluto-arm-cpu/

[6] RTL-SDR TUTORIAL: GPS DECODING AND PLOTTING. [online], 28. 2. 2017, Navštíveno 25. 12. 2018.
URL https://www.rtl-sdr.com/rtl-sdr-tutorial-gps-decoding-plotting/

[7] SDR Radio. [online], Navštíveno 12. 3. 2018.
URL http://www.sdr.ipip.cz

[8] GPS Signal Plan. Navipedia [online], Navštíveno 18.05.2018.
URL http://www.navipedia.net/index.php/GPS_Signal_Plan

[9] ADALM-PLUTO Software-Defined Radio Active Learning Module. Analog devices [online], Navštíveno 2. 5. 2018.
URL http://www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/adalm-pluto.html#eb-overview

[10] Airspy Mini – The Ultimate Performance in a Dongle Form Factor. [online], Navštíveno 25. 12. 2018.
URL https://airspy.com/airspy-mini/

[11] SDR# FM Radio | Getting Started with RTL-SDR and SDR-Sharp and CubicSDR | Adafruit Learning System by lady ada. [online], Navštíveno 25. 12. 2018.
URL https://www.learn.adafruit.com/getting-started-with-rtl-sdr-and-sdr-sharp/sdr-number-fm-radio

[12] SDR Radio - Software Defined Radio. [online], Navštíveno 25. 12. 2018.
URL http://www.sdr.ipip.cz/rtl-sdr

[13] ADALM-PLUTO for SDR Engineers. Analog Devices Wiki [online], Navštíveno 26. 12. 2018.
URL https://wiki.analog.com/university/tools/pluto/engineers

[14] ADALM-PLUTO SDR Active Learning Module. Analog Devices [online], Navštíveno 26. 12. 2018.
URL https://www.analog.com/media/en/news-marketing-collateral/product-highlight/ADALM-PLUTO-Product-Highlight.pdf

[15] RF Agile Transceiver AD9361 Data Sheet. Analog Devices [online], Navštíveno 26. 12. 2018.
URL https://www.analog.com/media/en/technical-documentation/data-sheets/AD9361.pdf

[16] RF Agile Transceiver AD9363 Data Sheet. Analog Devices [online], Navštíveno 26. 12. 2018.
URL https://www.analog.com/media/en/technical-documentation/data-sheets/AD9363.pdf

[17] RF Agile Transceiver AD9364 Data Sheet. Analog Devices [online], Navštíveno 26. 12. 2018.
URL https://www.analog.com/media/en/technical-documentation/data-sheets/AD9364.pdf

[18] Baseband Processing. Navipedia [online], Navštíveno 27.12.2018.
URL https://gssc.esa.int/navipedia/index.php/Baseband_Processing

[19] Correlators. Navipedia [online], Navštíveno 27.12.2018.
URL https://gssc.esa.int/navipedia/index.php/Correlators

[20] Delay Lock Loop (DLL). Navipedia [online], Navštíveno 27.12.2018.
URL https://gssc.esa.int/navipedia/index.php/Delay_Lock_Loop_(DLL)

[21] Multicorrelator. Navipedia [online], Navštíveno 27.12.2018.
URL https://gssc.esa.int/navipedia/index.php/Multicorrelator

[22] Phase Lock Loop (PLL). Navipedia [online], Navštíveno 28.12.2018.
URL https://gssc.esa.int/navipedia/index.php/Phase_Lock_Loop_(PLL)

[23] Tracking Loops. Navipedia [online], Navštíveno 28.12.2018.
URL https://gssc.esa.int/navipedia/index.php/Tracking_Loops

[24] libiio: Main Page. Analog Devices [online], Navštíveno 30.4.2019.
URL https://analogdevicesinc.github.io/libiio/index.html

[25] Linux Industrial I/O Subsystem. Analog Devices Wiki [online], Navštíveno 30.4.2019.
URL https://wiki.analog.com/software/linux/docs/iio/iio

[26] Who can tell me the differences or relationship of ad9361-phy and cf-ad9361-lpc and cf-ad9361-dds-core-lpc? EngineerZone [online], Navštíveno 30.4.2019.
URL https://ez.analog.com/linux-device-drivers/linux-software-drivers/f/q-a/85546/who-can-tell-me-the-differences-or-relationship-of-ad9361-phy-and-cf-ad9361-lpc-and-cf-ad9361-dds-core-lpc

[27] USB OTG. Analog Devices Wiki [online], Navštíveno 3.5.2019.
URL https://wiki.analog.com/university/tools/pluto/devs/usb_otg

[28] Plutosdr Schematic Rev B. Analog Devices Wiki [online], Navštíveno 5.5.2019.
URL https://wiki.analog.com/_media/university/tools/pluto/hacking/plutosdr_schematic_revb.pdf

[29] AD9363 (Rev. D). Analog Devices [online], Navštíveno 6.5.2019.
URL https://www.analog.com/media/en/technical-documentation/data-sheets/AD9363.pdf

[30] AXI Reference Guide. Xilinx [online], Navštíveno 6.5.2019.
URL https://www.xilinx.com/support/documentation/ip_documentation/ug761_axi_reference_guide.pdf

[31] Integrate FIR filters into the FMCOMMS2 HDL design. Analog Devices Wiki [online], Navštíveno 6.5.2019.
URL https://wiki.analog.com/resources/fpga/docs/hdl/fmcomms2_fir_filt

[32] Oscillators. Budapest University of Technology and Economics (BME), Faculty of Electrical Engineering and Informatics (VIK) [online], Srpen 2007, Navštíveno 28.12.2018.
URL https://www.mit.bme.hu/system/files/oktatas/targyak/8501/NCOcut.pdf

[33] ADAMS Jon T.: An Introduction to IEEE STD 802.15.4. [online], navštíveno 29.04.2018.
URL https://web.sonoma.edu/users/f/farahman/sonoma/courses/cet543/resources/802_intro_01655947.pdf

[34] ŽALUD Václav: *Moderní radioelektronika*. BEN - technická literatura, 2000, ISBN 80-86056-47-3.

[35] BERNKOPF Jan: *Lokalizace akustických zrojů metodou TDOA*. Bakalářská práce, Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008, vedoucí práce doc. Ing. Jiří Šebesta, Ph.D.

[36] HOUSTON Ben, JOHNSON Dan: The IEEE 802.15.4 Standard. University of Colorado [online], navštíveno 30.04.2018.
URL http://ecee.colorado.edu/~liue/teaching/comm_standards/2011F_802.15.4/index.htm

[37] HUBÁČEK Petr, HOŠKO Eduard, VESELÝ Jiří: *Základy teorie pasivních systémů I*, kapitola Časoměrně – hyperbolická metoda určování polohy zdroje signálu. Univerzita obrany Brno, 2007, ISBN 978 – 80 – 7231 – 476 - 8, s. 54 – 61.

[38] Navipedia: Main Page. Navipedia [online], 2017, navštíveno 18.02.2018.
URL http://www.navipedia.net/index.php/Main_Page

[39] POOLE Ian: Bluetooth radio interface, modulation, channels. radio-electronics.com [online], navštíveno 30.04.2018.
URL http://www.radio-electronics.com/info/wireless/bluetooth/radio-interface-modulation.php

[40] RYDLO, Štěpán: *Lokalizace objektů v reálném čase*. Bakalářská práce, Vysoké učení technické, Fakulta inforamčních technologií, 2017, vedoucí práce doc. Dr. Ing. Dušan Kolář.
URL http://www.fit.vutbr.cz/study/DP/BP.php?id=20014&file=t

[41] RYDLO, Štěpán: *Detekce dronu v prostoru*. Semestrální projekt, Vysoké učení technické, Fakulta inforamčních technologií, 2019, vedoucí práce prof. Ing., Dipl.-Ing. Martin Drahanský, Ph.D.

[42] SHIRBATE Ruchira, PANSE Tejaswini, RELEKAR Chetan: Design of parallel FFT architecture using Cooley Tukey algorithm. In *2015 International Conference on Communications and Signal Processing (ICCSP)*, IEEE, ISBN 9781479980819.
URL http://www.onlinejet.net/temp/JEngTechnol1283-3830788_103827.pdf

[43] SICKLE Jan Van, DUTTON John A.: Spread Spectrum and Code Modulation of L1 GPS Carrier. e-Education Institute, College of Earth and Mineral Sciences, The Pennsylvania State University [online], navštíveno 30.04.2018.
URL https://www.e-education.psu.edu/geog862/node/1753

[44] SMÉKAL Zdeneněk, SYSEL Petr: *Signálové procesory*. Sdělovací technika, 2006, ISBN 978-80-8664-508-7.

[45] Soonho Kwon Daeoh Kim, Jihye Lee, Sangmi Moon, Myeonghun Chu, Sara Bae, Cheolwoo You, Huaping Liu, Jeong-Ho Kim, Dae Jin Kim, Hosung Park, Jin Young Kim, Cheol-Sung Kim, Intae Hwang: Performance Analysis of 3D Localization for a Launch Vehicle Using TOA, AOA, and TDOA. In *Wireless Personal Communications*, ročník 103, Springer US, 2018, s. 1443–1464.

[46] STEWART Robert, BARLEE Kenneth, ATKINSON Dale, CROCKETT Louise: *Software Defined Radio using MATLAB & Simulink and the RTL-SDR*. Strathclyde Academic Media, 2015, ISBN 9780992978716.

[47] SVOBODA Petr: Možnosti pasivní detekce a ledování vzdušných cílů. *Vojenské rozhledy*, 2001, ISSN 1210-3292.

[48] VŮJTEK Milan: Teorie signálu a informace II. Portál moderní fyziky [online], navštíveno 30.04.2018.
URL
https://fyzika.upol.cz/cs/system/files/download/vujtek/texty/tsii2.pdf

[49] VOJTEK David: Historie, současnost a budoucnost lokalizace a navigace. Institut geoinformatiky, Vysoká škola báňská - Technická univerzita Ostrava, 2017, navštíveno 12.12.2018.
URL http://geoinformatika-1.vsb.cz/vojtek/content/gnps/files/_prez/01/01_prezentace.pdf

[50] Wikipedia contributor: Navigation. Wikipedia [online], 2018, navštíveno 18.02.2018.
URL https://en.wikipedia.org/wiki/Navigation

Příloha A

Ukázky implementace

```
1  struct iio_context *ctx = NULL;
2  struct iio_channel *chn = NULL;
3  struct iio_device *phy;
4
5  //Vytvoreni spojeni mezi programem a zarizenim
6  ctx = iio_create_context_from_uri("usb:1.25.5");
7
8  //Kontrola zda spojeni bylo vytvoreno
9  if(ctx == NULL){
10   fprintf(stderr, "No device on address: %s", param->device);
11   return false;
12 }
13
14 //Ziskani zarizeni, ktere v danem kontextu existuje
15 phy = iio_context_find_device(ctx, "ad9361-phy");
16
17 //Nalezeni kanalu, ktery budeme nastavovat
18 chn = iio_device_find_channel(phy, "voltage0", false);
19
20 //Nastaveni portu pro vysilani, vzorkovaci frekvence a sirky pasma
21 iio_channel_attr_write(chn, "rf_port_select", "A_BALANCED");
22 iio_channel_attr_write_longlong(chn, "sampling_frequency", GHz(2.4));
23 iio_channel_attr_write_longlong(chn, "rf_bandwidth", MHz(2));
24
25 //Nastaveni frekvence, ktere je usisteno na jinem kanalu
26 chn = iio_device_find_channel(phy, "altvoltage1", true);
27 iio_channel_attr_write_longlong(chn, "frequency", param->freq);
28 iio_channel_attr_read_longlong(chn, "frequency", &(param->freq));
```

Příklad A.1: Nastavení zařízení pro vysílání a přijímání.

```
struct iio_buffer *buffer
struct iio_device *dev;
struct iio_channel *tx0_i, *tx0_q;

//Ziskani ukazatele zarizeni v kontextu
dev = iio_context_find_device(ctx, "cf-ad9361-dds-core-lpc");

//Nalezeni jednotlivych kanalu v zarizeni
tx0_i = iio_device_find_channel(dev, "voltage0", 1);
if(!tx0_i)
  tx0_i = iio_device_find_channel(dev, "altvoltage0", 1);

tx0_q = iio_device_find_channel(dev, "voltage1", 1);
if(!tx0_q)
  tx0_q = iio_device_find_channel(dev, "voltage1", 1);

//Aktivovat kanaly
iio_channel_enable(tx0_i);
iio_channel_enable(tx0_q);

while(!endTransmit){
  void *p_dat, *p_end;
  ptrdiff_t p_inc;

  //Ziskani ukazatele na TX buffer IQ pro jeho zapsani
  p_inc = iio_buffer_step(txbuf);
  p_end = iio_buffer_end(txbuf);

  //Cyklus pro zapsani dat do bufferu
  for(p_dat = iio_buffer_first(txbuf, tx0_i); p_dat < p_end; p_dat += p_inc){
    //I vzorek signalu
    ((int16_t *)p_dat)[0];
    //Q vzorek signalu
    ((int16_t *)p_dat)[1];
  }

  //Vlozeni dat na zarizeni
  iio_buffer_push(txbuf);
}
```

Příklad A.2: Inicializace paměti pro vysílání dat.