Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Méréstechnika és Információs Rendszerek Tanszék

Borbély Zsombor

KOMPLEX BURKOLÓRA ALAPOZOTT INFORMÁCIÓFELDOLGOZÁS SDR PLATFORMON

KONZULENS

Krébesz Tamás

BUDAPEST, 2018

Tartalomjegyzék

Tartalomjegyzék **2**
Összefoglaló **5**
Abstract **6**
1 Bevezető **7**
2 Az SDR platform matematikai háttere **8**
2.1 A komplex burkoló elmélete 8
2.1.1 Hilbert transzformáció 8
2.1.2 Analitikus jelek 9
2.1.3 Komplex burkolók 11
2.2 Alapsávi ekvivalens előállítása a gyakorlatban 12
2.3 Komplex burkolóra alapozott modulációs eljárások 15
3 Az SDR platform felépítése **18**
3.1 Koncepcionális felépítés 18
3.2 SDR vevő megvalósítások 19
3.2.1 Homodin vevő 20
3.2.2 Heterodin vevő 20
3.2.3 Direkt konverziós vevő 22
3.3 Megvalósítási kihívások 23
3.3.1 Termikus zaj 23
3.3.2 Flicker zaj 23
4 Beágyazott méréstechnikai rendszer **25**
4.1 RTL SDR 25
4.2 Intelligens spektrumanalizátor tervezése Matlab segítségével 28
4.3 A spektrumanalizátor specifikációja 29
4.4 A spektrumanalizátor szoftveres rendszerterve 30
4.4.1 Adatfeldolgozást végző függvény 30
4.4.2 Adatfeldolgozó függvény 33
4.4.3 Vezérlő modul 34
5 Az adatfeldolgozás megvalósítása **36**
5.1 RTL SDT bekalibrálása 36
5.1.1 Frekvenciakorrekciós tényező 36

5.1.2 Erősítés és Mintavételi frekvencia 40
5.1.3 Mintavételi frekvencia 41
5.1.4 Bemeneti adat típusa 41
5.2 A rendszerterv szerinti implementáció 43
5.2.1 DC ofszet oka és kiszűrése 46
5.2.2 Átviteli karakterisztika korrekciója 49
5.2.3 Az adatgyűjtés validációja 50
5.2.4 Adatfeldolgozás megvalósítása 54
5.2.5 Grafikus interfész megvalósítása 59
Összegzés és továbbfejlesztési lehetőségek 63
Irodalomjegyzék 64

HALLGATÓI NYILATKOZAT

Alulírott **Borbély Zsombor**, szigorló hallgató kijelentem, hogy ezt a diplomatervet meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy hitelesített felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Kelt: Budapest, 2018. 12. 15.

...…….
Borbély Zsombor

Összefoglaló

Napjainkban egyre elterjedtebbek a szoftverdefiniált rádiók, melyek fő tulajdonsága, hogy a korábban hardveresen implementált funkciókat szoftveres implementációval helyettesítik. Ennek következtében az SDR-ek egyszerűen újrakonfigurálhatók, és felhasználásuk sokrétűbbé vált, mint hardveres társaiké. A szoftverdefiniált rádiók segítségével könnyen lehet prototípusokat, illetve újrakonfigurálható rádiós rendszereket készíteni.

Diplomatervezési feladatomként egy választott SDR platformot használva jelfeldolgozó alkalmazást készítettem, mely képes egy adott frekvenciasávot megmérni, kihasználtságáról információkat gyűjteni és kiértékelni melyek azok a sávok, amelyekben más rádiós rendszerek zavarása nélkül kezdeményezhető vezeték nélküli adatátvitel.

A munkám során megismerkedtem a komplex burkolók matematikai hátterével, és a különböző SDR architektúrákkal. Ezen ismeretek birtokában egy RTL2832U chip alapú vevőt és MATLAB-ot felhasználva megterveztem és implementáltam egy intelligens spektrumanalizátort. Az alkalmazással méréseket végeztem, és a mérési eredmények segítségével validáltam a spektrumanalizátor működését.

Abstract

Nowadays, Software Defined Radios (SDR) are becoming more and more popular. The main feature of these devices, that some of the physical layer function are replaced by software. Therefore, SDRs can be easily reconfigured and there are several ways to utilize them. With SDR, one can easily create prototypes or reconfigurable radio systems.

The goal of my thesis is to create a signal processing application using a chosen SDR platform. This application should be capable of measuring a given frequency band and collecting information about its utilization. Based on the utilization, the application should evaluate which bands are not allocated.

In this paper, I studied the mathematical background of the complex envelopes and various SDR architectures. Using this knowledge, I designed and implemented a spectrum analyzer using an RTL2832U chip-based receiver and MATLAB. After the implementation, measurements were performed using the application. Based on these measurement results, I validated the operation of the spectrum analyser.

1 Bevezető

A szoftverdefiniált rádióknak hagyományos rádiókkal szembeni különbségük, hogy a rádión vett információ feldolgozása szoftveresen történik. Ennek nagy előnye a flexibilitás és újrakonfigurálhatóság. Mivel a feldolgozást végző szoftveres algoritmust bármikor módosíthatjuk, tetszőleges modulációs/demodulációs eljárás végrehajtható a rendszer újrakonfigurálásával.

Diplomatervem célja egy kiválasztott SDR platformon beágyazott spektrumanalizátor elkészítése volt, amely autonóm módon megkeresi a hasznos információt a vizsgált spektrumon, majd kiértékeli, hogy adott sávszélesség előírás mellett mely sávok tekinthetők szabadnak.

A dolgozat második és harmadik fejezetében bemutatom a komplex burkolón alapuló információfeldolgozás matematikai hátterét, ismertetem a különböző SDR implementációkat az alapján, hogy milyen eljárással nyerik ki a komplex burkolót a rádiófrekvenciás jelekből.

A dolgozat negyedik fejezetében ismertetem a feladat elvégzésére kiválasztott SDR platformot, majd a rendszerterv alfejezetben a spektrumanalizátor felépítését mutatom be.

Az ötödik fejezetben a spektrumanalizátor implementációjával, és a mért adatok validálásával foglalkozom. Az első alfejezetében ismertetem hogyan kalibráltam be a SDR rádiót, míg a második alfejezetben az implementációs problémákat küszöbölöm ki és a mért adatok, illetve az implementált algoritmus validálását ismertetem.

2 Az SDR platform matematikai háttere

2.1 A komplex burkoló elmélete

A teljesen szoftveresen megvalósított rádiós rendszer vevőjének a feladata, hogy az RF sávban található nagy frekvenciás jeleket az antennáról közvetlenül digitalizálja és továbbítsa a beágyazott feldolgozó rendszer felé. A gyakorlatban ez azt jelentené, hogy tetszőleges centerfrekvenciájú és modulációjú jelet kéne mintavételezni, hiszen a feldolgozás szoftveresen történne. Mivel a maximális mintavételi frekvenciára technológia megkötések vannak, ezért szükség van egy olyan eljárás kidolgozására, amely segítségével az RF jeleket szét lehet bontani egy vivőfrekvenciás és egy alapsávi komponensre.

A problémára a komplex burkolók alkalmazása jelenti a megoldást. Komplex burkolót alkalmazva a vett jelhez szükséges mintavételi frekvencia nem az RF jel legnagyobb frekvenciakomponensétől, hanem az RF jel sávszélességétől függ. A komplex burkolós eljárás célja, hogy a RF jelet alapsávra keverjük, majd onnan az adott modulációhoz tartozó demodulációt megvalósító algoritmus segítségével előállítsuk az adó oldal által küldött információ becslőjét.

2.1.1 Hilbert transzformáció

A Hilbert transzformáció egy lineáris transzformáció, mely az egyoldalas Fourier transzformáció valós és képzetes részei közötti kapcsolatot reprezentálja [1]. Egy $g(t)$ jel időbeli Hilbert transzformáltját az (1) egyenlet írja le, jelölése $\hat{g}(t)$.

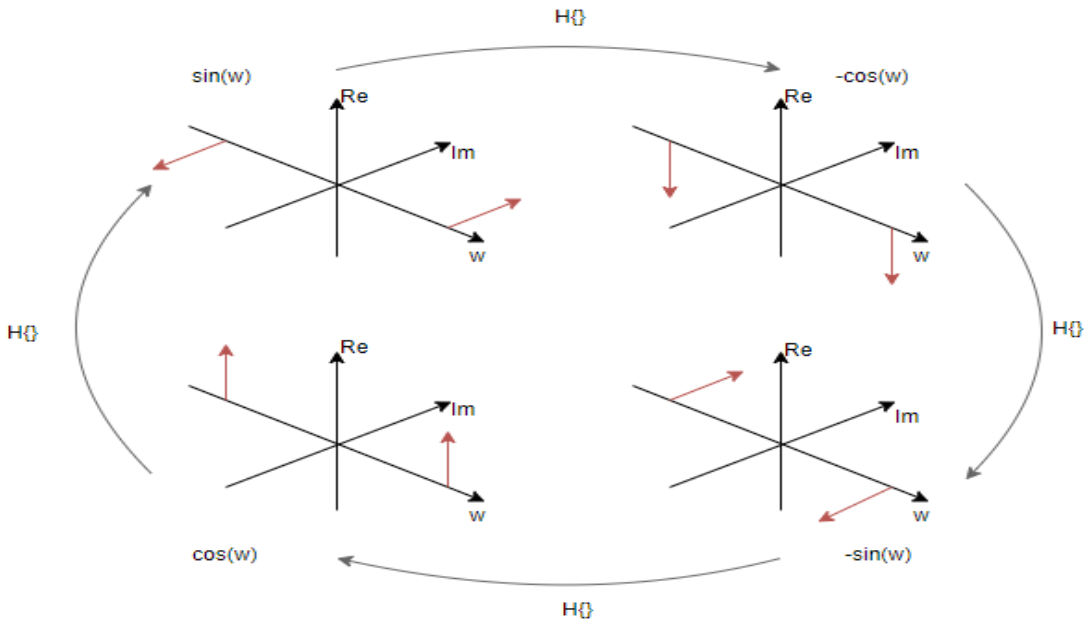
$$H\{g(t)\} = \hat{g}(t) = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{g(\tau)}{t-\tau} d\tau \tag{1}$$

Az (1) egyenlet úgy is felírható, mint $g(t)$ és $\frac{1}{\pi t}$ függvény konvolúciója [2]. Ennek következtében a Hilbert transzformáció frekvenciatartományban felírható úgy is, mint

$g(t)$ és $\frac{1}{\pi t}$ Fourier transzformáltjának a szorzata. Ha vesszük $\frac{1}{\pi t}$ Fourier transzformáltját, akkor a (2) egyenlet szerinti eredményre jutunk:

$$H\left\{\frac{1}{\pi t}\right\} = -j * sgn(f) \tag{2}$$

Ha megvizsgáljuk a szinusz jelre a Hilbert transzformáció hatását, majd azt többször elvégezzük, akkor az 1. ábrán látható eredményre jutunk. A transzformáció a szinuszból először mínusz koszinuszt, majd mínusz szinuszt, koszinuszt és végül újra szinuszt képez. A transzformáció ezen tulajdonsága miatt nevezzük a műveletet kvadratúra szűrőnek is [1]. Az 1. ábrán a szinusz és Hilbert transzformáltjainak komplex spektruma látható w szögsebesség függvényében.



1. ábra - Hilbert transzformáció fázisforgató hatása

2.1.2 Analitikus jelek

Valós jelek esetén a jel analitikus formájának fő tulajdonsága, hogy negatív frekvenciatartományban nem rendelkezik spektrális komponensekkel.

Ahhoz, hogy a negatív tengelyen ne legyen spektruminformáció, a spektrumnak olyan transzformáltját kell hozzáadni önmagához, melynek a pozitív oldali spektruma

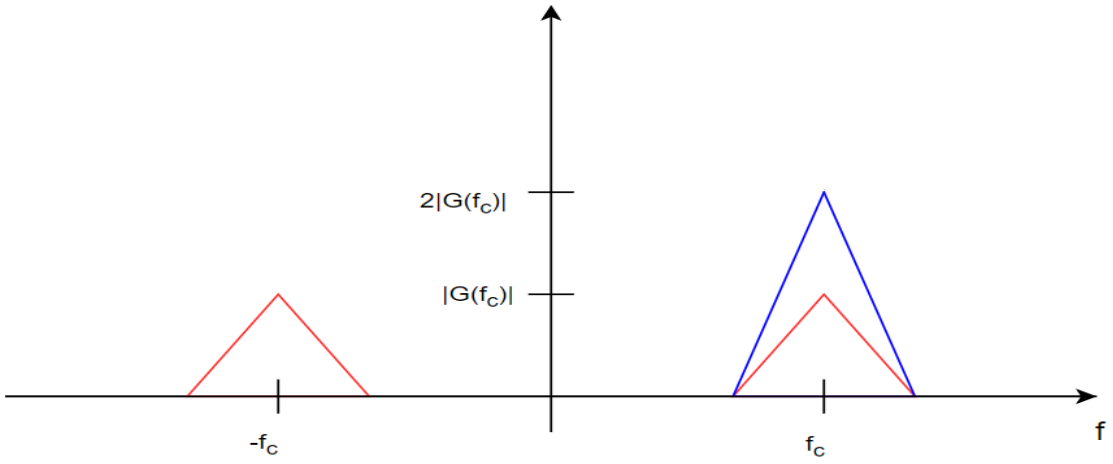
változatlan maradna, a negatív oldali pedig ellentétes előjelűvé változna. Az előbb ismertetett transzformációt a Hilbert transzformált jel j-vel való szorzása képzi, míg a összeget, mely segítésével a jelnek a negatív oldali spektrumát eltüntetjük, a jel analitikus formájának nevezzük. Ezek alapján a $g_+(t)$ analitikus jel, a (3) egyenlet szerint [4]:

$$g_+(t) = g(t) + j\hat{g}(t) \tag{3}$$

Ha a (3) egyenletnek vesszük a Fourier transzformáltját, akkor a (4) egyenlet szerinti eredményre jutunk. Az jelek Fourier transzformáltját az egyenletben nagy betűvel jelöltem.

$$\begin{aligned} G_+(f) = G(f) \; + \; j * \hat{G}(f) = \; & G(f) + j(-j * sgn(f) * G(f)) \\ & = G(f) * (1 + sgn(f)) \end{aligned} \tag{4}$$

A (4) egyenletben $sgn(f)$ az előjel függvény. Mivel $1 + sgn(f)$ negatív f esetén 0 értéket vesz fel, ezért $G_+(f)$ spektruma egyoldalas. Az eredeti és analitikus jelet a *2. ábra* szemlélteti, ahol az eredeti jel spektrumát kékkel, míg az analitikus jelét pirossal jelöltem.



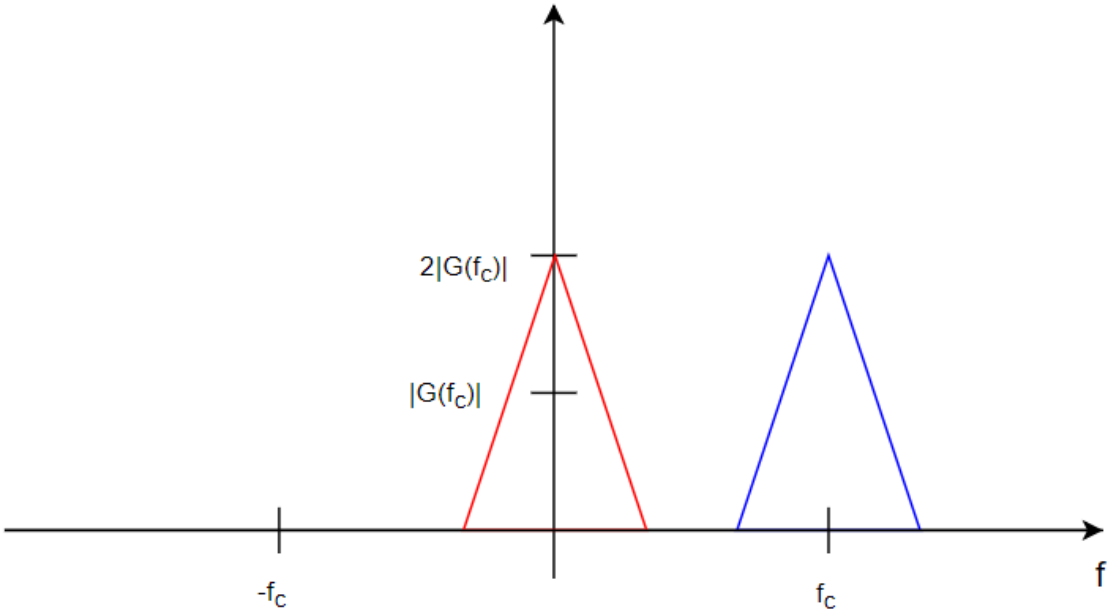
2. ábra - Az eredeti jel (piros spektrum) és a belőle képzett analitikus jel (kék spektrum) a frekvenciatartományban[4]

2.1.3 Komplex burkolók

Ahogyan a (5) egyenleten látható, minden analitikus jel felbontható egy lassan változó alapsávi jelre – vagy más néven alapsávi ekvivalens jelre - és egy gyorsan változó komplex exponenciális tagra. Az alapsávi ekvivalens jelet komplex burkolóként is szoktuk nevezni és csak ez hordozza az információt [2].

$$g_+(t) = g(t) + j\hat{g}(t) = \tilde{g}(t) * e^{j2\pi f_c t} \qquad (5)$$

A (5) egyenletben $\tilde{g}(t)$ az eredeti $g(t)$ jelnek alapsávra transzformált ekvivalense. Az egyenlet vizuális szemléltetése a 3. ábrán látható. Pirossal szemléltettem $\tilde{g}(t)$ alapsávi ekvivalenst, míg a $g_+(t)$ analitikus jelet kékkel jelöltem.



3. ábra – Az analitikus jel (kék spektrum) és annak alapsávra transzformált ekvivalense, (piros spektrum)

A komplex burkoló, ahogy a neve is sugallja egy komplex mennyiség mely felbontható valós és képzetes részre a (6) egyenlet szerint.

$$\tilde{g}(t) = g_I(t) + jg_Q(t) \tag{6}$$

Eszerint a jel rendelkezik egy valós $g_I(t)$ (In-phase) és egy képzetes $g_Q(t)$ (Quadrature) komponenssel. A gyakorlatban a komplex burkolóval végzett jelfeldolgozásnál ezeket a jel komponenseket tudjuk előállítani könnyedén. Ennek az a következmény, hogy a gyakorlatban külön feldolgozási ág kell az I és Q komponenshez [4].

Összefoglalva a komplex burkolók használatával sikerült elérni, hogy g(t) valós függvény felírható legyen alapsávi ekvivalens és komplex exponenciális tag szorzatára, ahol az alapsávi ekvivalens információtartalma megegyezik g(t) valós jelével, viszont elegendő csak 2B frekvenciával mintavételezni, ahol B az alapsávi ekvivalens sávszélessége.

2.2 **Alapsávi ekvivalens előállítása a gyakorlatban**

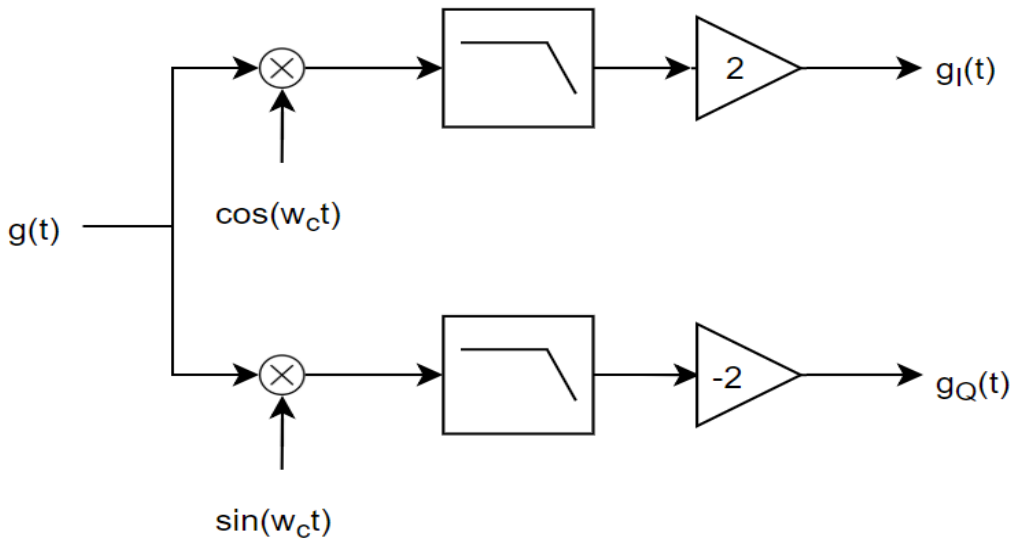
Az alapsávi ekvivalens jel előállítása a gyakorlatban az I és Q komponensek külön előállításával történik, mivel komplex jel realizálása nem lehetséges. Ha vesszük a valós

g(t) jelünket, akkor a következő összefüggés érvényes a jel és a komplex burkoló közti kapcsolatban [3]:

$$g(t) = Re\{\tilde{g}(t)e^{jw_c t}\} = Re\{[g_I(t) + jg_Q(t)]e^{jw_c t}\} = \quad (7)$$

$$g_I(t)\cos(w_c t) - g_Q(t)\sin(w_c t)$$

A (7) egyenlet szerint g(t) valós jel felbontható I és Q komponensre úgy, hogy a komponensekhez 90 fokkal eltolt vivőket rendelünk. Az alapsávi ekvivalens I és Q komponensének előállítása szorzások segítségével könnyen elvégezhető. Egy ilyen keverő áramkört kvadratúra keverőnek hívunk, és blokkvázlata az 4. ábrán látható.



4. ábra - Kvadratúra keverő

Az I és Q feldolgozási ágra szétosztott beérkező jelet először az w_C vivőfrekvencián működő lokális oszcillátor által generált szinuszos jellel szorozzuk meg. A két ágon az oszcillátor frekvenciája azonos, de fázisuk 90 fokban eltér. Amikor a lokális oszcillátor jelét összeszorozzuk az eredeti jellel, akkor nem csak az alapsávban jelenik meg a modulált jel, hanem a vivőfrekvencia kétszeresén is meg fog jelenni. Ezt a nagyfrekvenciás jelet aluláteresztő szűrővel lehet kiszűrni. A lokális oszcillátorral végzett szorzásnál fontos, hogy az oszcillátor és a vivő szinkronban legyenek, hiszen szorzásnál a vett jel arányos az oszcillátor és a vivőjel szögfüggvénye közti fáziskülönbség

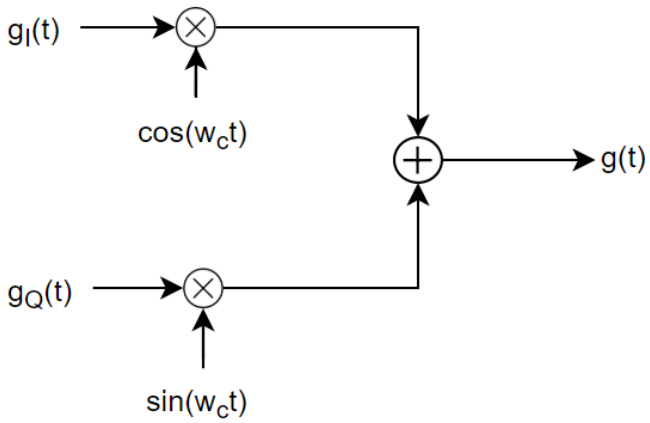
koszinuszával [3]. A (8) és (9) egyenlet a 4. ábrán látható kvadratúra keverés szögfüggvénytől való függését vezeti le. A (8) egyenleten az I ág, míg a (9) egyenleten a Q ágon elvégzett szorzást mutatja, mindkét ágon a mixer fáziseltérését φ-vel jelöltem.

$$(g_I(t)\cos(w_c t) + g_Q(t)\sin(w_c t)) * \cos(w_c t + \varphi) = \quad (8)$$
$$g_I(t)\cos(\varphi) - g_Q(t)\sin(\varphi)$$
$$+ g_I(t)\cos(2w_c t + \varphi) + g_Q(t)\sin(2w_c t + \varphi)$$

$$(g_I(t)\cos(w_c t) + g_Q(t)\sin(w_c t)) * \sin(w_c t + \varphi) = \quad (9)$$
$$g_Q(t)\cos(\varphi) + g_I(t)\sin(\varphi)$$
$$- g_I(t)\cos(2w_c t + \varphi) + g_Q(t)\sin(2w_c t + \varphi)$$

Mindkét ágon az eredmény tartalmaz $2w_c$-s tagokat, melyek szűrők segítségével kiszűrhetők, továbbá mindkét ágon megjelenik $g_I(t)$ és $g_Q(t)$ $\sin(\varphi)$ és $\cos(\varphi)$ függvényében. Ha φ értéke nem nulla – vagy 90° egész számú többszöröse – akkor mindkét ágban meg fog jelenni $g_I(t)$ és $g_Q(t)$ egyszerre. A jelenség az angol nyelvű elnevezése IQ imbalance, a gyakorlatban a jelenség oka a lokáloszcillátor fázis csúszása $g_I(t)$ és $g_Q(t)$-hez képest, megjegyzendő, hogy a két ágon a fáziscsúszás értéke eltérhet implementációs okokból. A jelenség kompenzálására több algoritmus is létezik, diplomatervemben ezekkel az algoritmusokkal nem foglalkoztam.

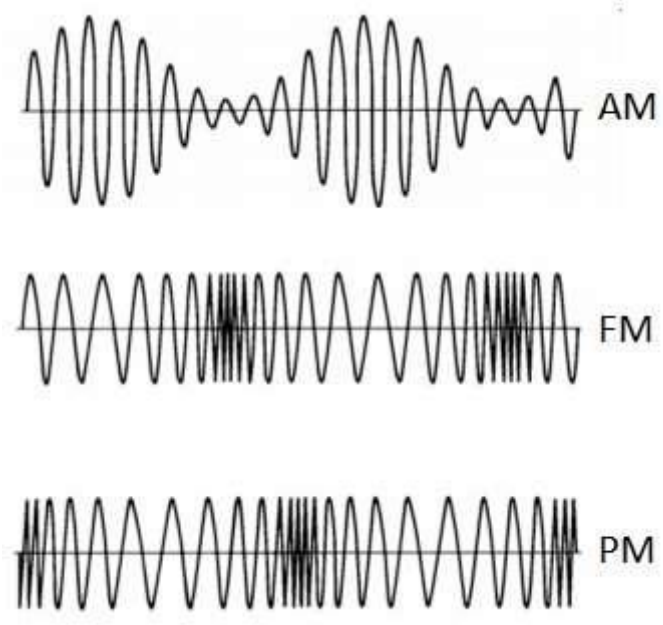
Alapsávi ekvivalensből sávátereszto jel előállítása az 5. ábra szerint történik. A g(t) jel előállításához ugyanúgy szükség van egy lokális oszcillátorra, továbbá két mixerre. Mixelésénél az I és Q ágat ugyanúgy 90 fokkal eltolt vivőjellel kell keverni.



5. ábra - Sávátereszto jel előállítása komplex burkoló segítségével

2.3 **Komplex burkolóra alapozott modulációs eljárások**

Minden modulációs eljárásnak az alapja, hogy egy tipikusan alapsávi modulálandó jelet egy vivő jel segítségével próbáljuk felkeverni magasabb frekvenciára. A moduláció célja, hogy a periodikus szinuszos vivő jelet információval lássuk el. A vivő jel kétféleképpen látható el információhordozó tulajdonsággal; vagy az amplitúdójában hordozhatjuk az információt, vagy a szögében. Ez alapján beszélhetünk amplitúdómodulációról, illetve attól függően, hogy a vivő jelünk fázisának vagy frekvenciájának változtatásával akarunk információt vinni, fázis és frekvenciamodulációról. A különböző modulációs eljárásokat a 6. ábrán szemléltettem.



6. ábra – Moduláció szinuszos modulálandó jel esetén

Az alapsávi komplex burkoló felbontható egy amplitúdóra és egy komplex exponenciális tagra, ami a szöget tartalmazza.

$\tilde{g}(t) = g_I(t) + jg_Q(t) = a(t) * e^{j\theta(t)}$

$a(t)$ az időben változó amplitúdó, $\theta(t)$ pedig az időben változó fázis. $\tilde{g}(t)$-ből $g(t)$ a következőképp írható le[4]:

$$g(t) = Re\{\tilde{g}(t)e^{jw_c t}\} = Re\{a(t)e^{jw_c t + j\theta(t)}\} = \quad (10)$$

$$a(t)cos(w_c t + \theta(t))$$

$$a(t) = |\tilde{g}(t)| = \sqrt{g_I^2(t) + g_Q^2(t)} \tag{11}$$

$$\theta(t) = \arctan\left(\frac{g_Q(t)}{g_I(t)}\right) \tag{12}$$

Amplitúdó moduláció esetén az információt $a(t)$, fázis moduláció esetén $\theta(t)$, míg frekvenciamoduláció esetén $\frac{d}{dt}\theta(t)$ fogja hordozni.

Analóg AM demoduláció esetén az alapsávi jel előállításához elegendő a 4. ábra szerinti kvadratúra szorzó felső (In phase) ágát használni amennyiben feltételezzük, hogy a lokáloszcillátor jele fázishelyes. A QAM jelnél már mindkét ágra szükség van az alapsávi jelek előállításához. Mind AM és QAM jelek demodulálásánál elégséges a 5. ábrán látható kvadratúra keverő használata, mivel amplitúdómoduláció egy lineáris művelet, a keverővel való szorzás eredményeként visszakapjuk az alapsávi jeleket, további műveletre nincs szükség.

Szögmodulációk esetén, a moduláció és demoduláció során az amplitúdómodulációhoz képest további jelfeldolgozási feladatok szükségesek. Szögmodulációk általános alakja a (13) egyenleten látható, az információt hordozó jelet az egyenletben $\mu(t)$-vel jelöltem.

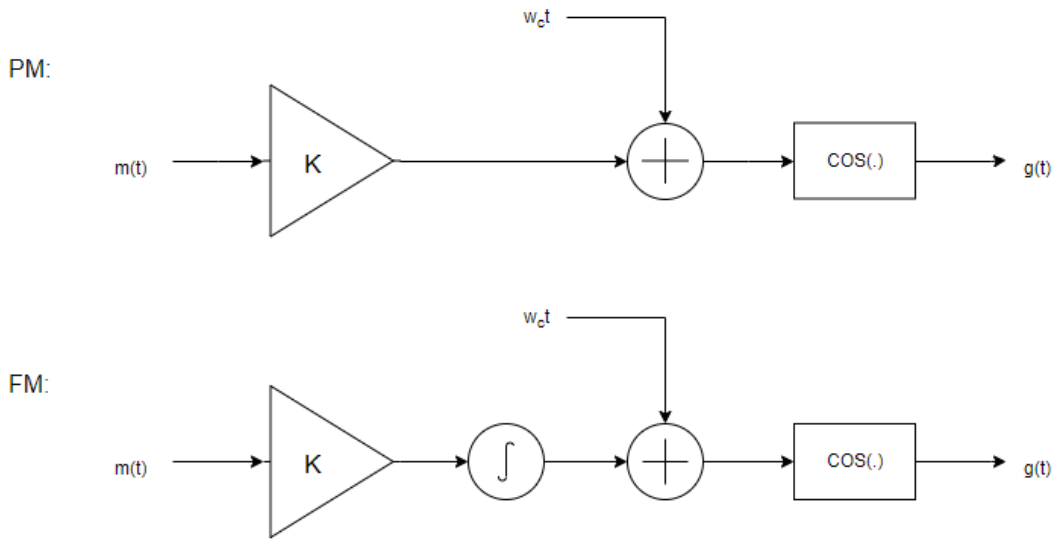
$$g(t) = A * cos(w_c t + \theta(t)) \tag{13}$$

Ahol A konstans, μ-re pedig a következő teljesül szög ás frekvenciamoduláció esetén:

$$\theta_{PM}(t) = k * m(t) \tag{14}$$

$$\frac{d}{dt}\theta_{FM}(t) = k * m(t) \tag{15}$$

Ahol k az erősítési tényező, m(t) pedig a modulálandó jelünk. Az g(t) jel előállítását m(t) jelből a 7. ábra szerinti blokkvázlatok alapján történik.



7. ábra – PM és FM modulátor

Az FM és PM jelek előállítása hasonlóan történik, különbség, hogy FM jel előállításánál az információt a szög változási sebessége hordozza, így egy integrátor tag is szükséges.

Az I és Q komponens, és a g(t) modulált jelek közt a következő összefüggések állapíthatók meg az (5) egyenletből kiindulva.

$$\begin{aligned} g(t) &= A\cos(w_c t + \mu(t)) \\ &= A\cos(\mu(t))\cos(w_c t) - A\sin(\mu(t))\sin(w_c t) \end{aligned} \tag{16}$$

$$g_I(t) = A\cos\big(\mu(t)\big), \quad g_Q(t) = A\sin(\mu(t)) \tag{17}$$

$$\mu(t) = arc\left(g_I(t) + jg_Q(t)\right) = arctan\left(\frac{g_Q(t)}{g_I(t)}\right) \tag{18}$$

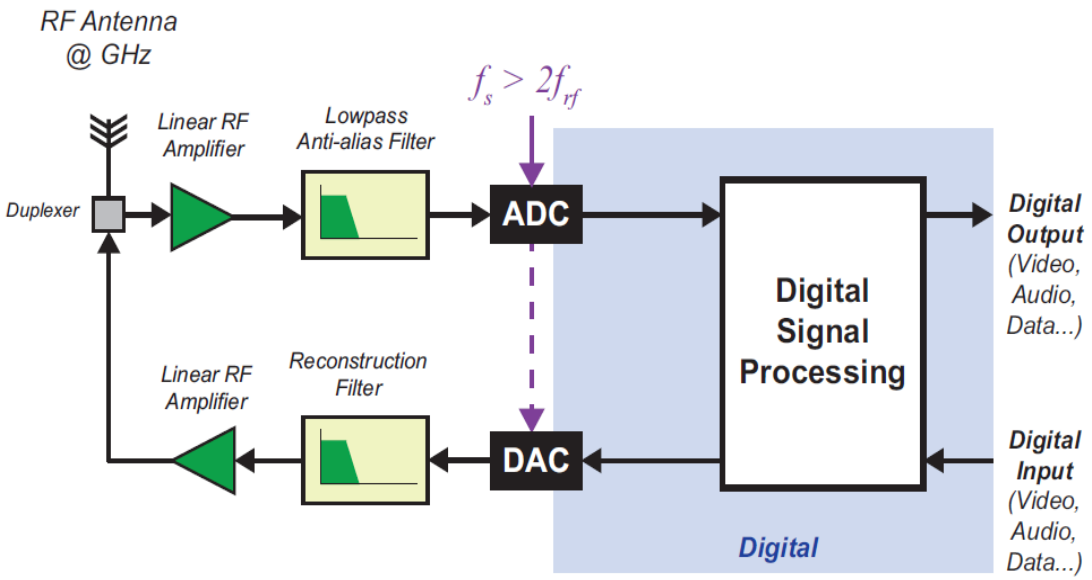
$$m_{FM}(t) = \frac{1}{2\pi K}\frac{d}{dt}\mu(t) \qquad m_{PM}(t) = \frac{1}{2\pi K}\mu(t) \tag{19}$$

3 Az SDR platform felépítése

Az SDR (Software Defined Radio) platform jellemzően egy beágyazott rendszerből vagy PC-ből és egy RF front-endből áll. Az RF front end a RF jelek nagyfrekvenciás jelkondicionálásáért felel, amely lehetővé teszi az AD átalakítást. A jelfeldolgozást tipikusan PC vagy beágyazott rendszer végzi. Az SDR rádió előnye más rádiókkal szemben, nagymértékű flexibilitás, hiszen a demodulációt szoftver végzi. A nagymértékű flexibilitásnak viszont az ára, hogy a demodulációt végző szoftvernek nagy erősforrásigénye lehet. A korábbi évtizedekben nem álltak rendelkezésre olyan dedikált chipek melyek a kvadratúra keverést elvégezzék, így a komplex burkoló előállítása analóg módon történt. A 2010-es években viszont már megjelentek olcsón olyan chipek és AD/DA átalakítók mely segítségével a komplex burkoló előállítása chipen belül történik, ezáltal az SDR rádiók piaci ára jelentősen lecsökkent. A SDR rádiók népszerűsége széleskörű használhatóságukból adódik. Az olcsó SDR rádiók segítségével könnyen ki lehet építeni kognitív rádiós alkalmazásokat, vagy akár spektrumanalizátor alapfunkcióit is meg lehet valósítani.

3.1 Koncepcionális felépítés

Egy általános SDR rádió 3 részre bontható; az analóg RF és ha van IF erősítőket és szűrőket tartalmazó analóg elektronikára, az analóg digitális átalakítást végző f_s mintavételi frekvencián működő AD és DA átalakítók, és a jelfeldolgozást elvégző teljesen digitális DSP részre. Egy ilyen koncepcionális felépítést mutat a 8. ábra.



8. ábra - Általános SDR rádió[5]

Az erősítő célja az antennáról vagy a DA-tól vett jelet úgy erősítse, hogy nagy erősítés mellet kicsi legyen a torzítás és a zajtényező. A vett jelet az AD átalakítás előtt célszerű kondicionálni átlapolódás gátló szűrővel. A szűrő tipikus töresponti frekvenciája az AD átalakító mintavételi frekvenciájának a fele. Célja az erősítő nemlinearitása által okozott esetleges kombinatorikus frekvenciák szűrése, illetve az átlapolódás megakadályozása. Adás esetén szintén alkalmazunk szűrőt, amelynek segítségével a DA kimenet jellemző lépcsős jelét simítjuk.

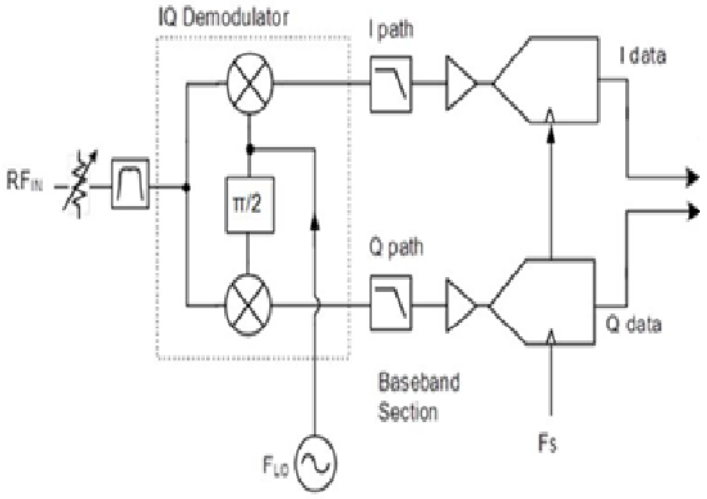
Az AD és DA átalakítók esetén a legfontosabb paraméterek a mintavételi frekvencia és az átalakító be- és kimeneti bitszélessége. Ezen paraméterek növelésével (főként a mintavételi frekvencia növelésével) az átalakítók bonyolultsága és ezzel természetesen ára is arányosan növekszik. Ennek következtében a gyakorlati megvalósításoknál olyan megoldásokra törekedtek, melyekben kisebb mintavételi frekvenciával ekvivalens eredményre jutunk, mint a 8. ábrán mutatott megvalósítás.

3.2 **SDR vevő megvalósítások**

Az idők során többféle SDR vevős megvalósítás jött létre, az egyes implementációkat erősen befolyásolta a technológiai háttér, mint például korra jellemző AD átalakítók fejlettsége, illetve természetesen a gyártási költségek alacsonyan tartása.

3.2.1 Homodin vevő

Homodin vagy más néven Zero-if vevők az RF jelet közvetlenül alapsávra keverik, vagyis az I és Q ágon is a mixer a vett jelet és annak vivőfrekvenciájával azonos rádiófrekvenciájú lokáljelet szorozza össze, ahogy ez a 9. ábrán látható.

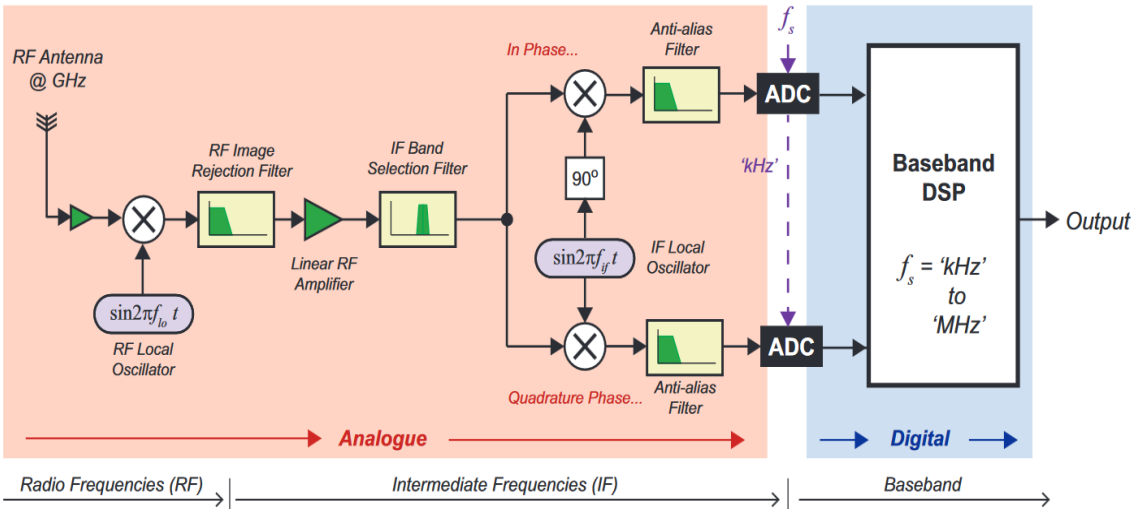


9. ábra – Zero IF vevő[14]

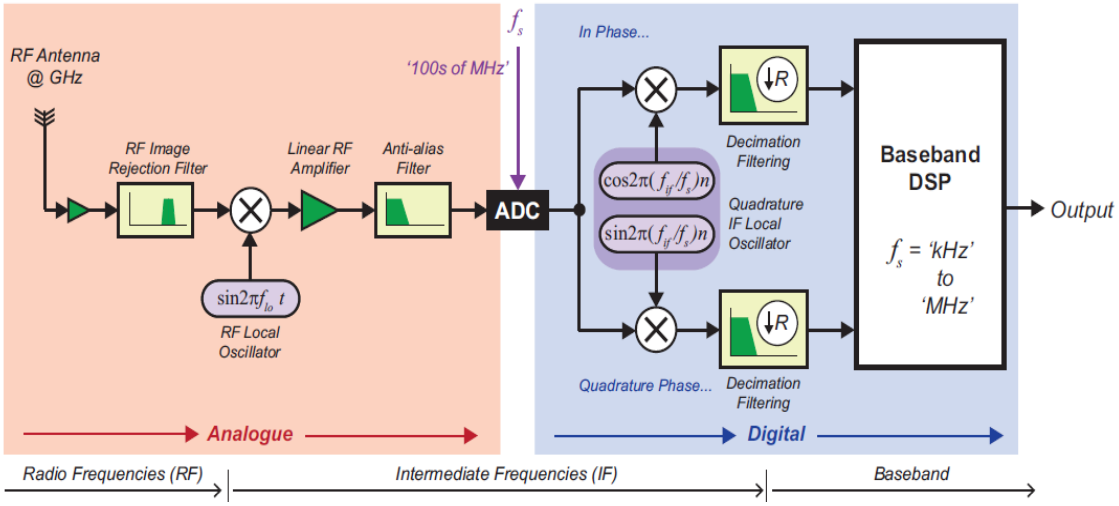
A megvalósítás egyszerű és olcsó, hiszen csak két szorzó kell alkalmazni, az AD átalakítónak pedig elég csak alacsony frekvencián működnie. Hátrányai közé tartozik, hogy az analóg szorzó bemenetei között áthallás, ezáltal lokálszivárgás léphet fel. További nehézség, hogy két azonos RF frekvenciájú 90 fok fáziskülönbségű szinuszos jel előállítása nem könnyű, ha a fáziskülönbség 90 foktól eltér, akkor a demodulált jelben torzítások tapasztalhatók.

3.2.2 Heterodin vevő

A heterodin vevők szemben a homodin vevőkkel szemben nem alapsávra hanem egy köztes IF frekvenciára keverik le a rádiófrekvenciás jelet, majd az I és Q komponens előállítása nem alapsávon, hanem IF frekvencián történik.– Az I és Q komponens előállítása történhet analóg módon, illetve digitálisan is, ahogyan ezt a 10. és a 11. ábra mutatja. Ezen megvalósítások előnye a homodin vevővel szemben, hogy a IF frekvenciás kvadratúra keverők kevesebb torzítást okoznak az I és Q komponensekben.



10. ábra – Heterodin vevő analóg IQ keverővel vevő [5]



11. ábra - Heterodin vevő digitális IQ keverővel [5]

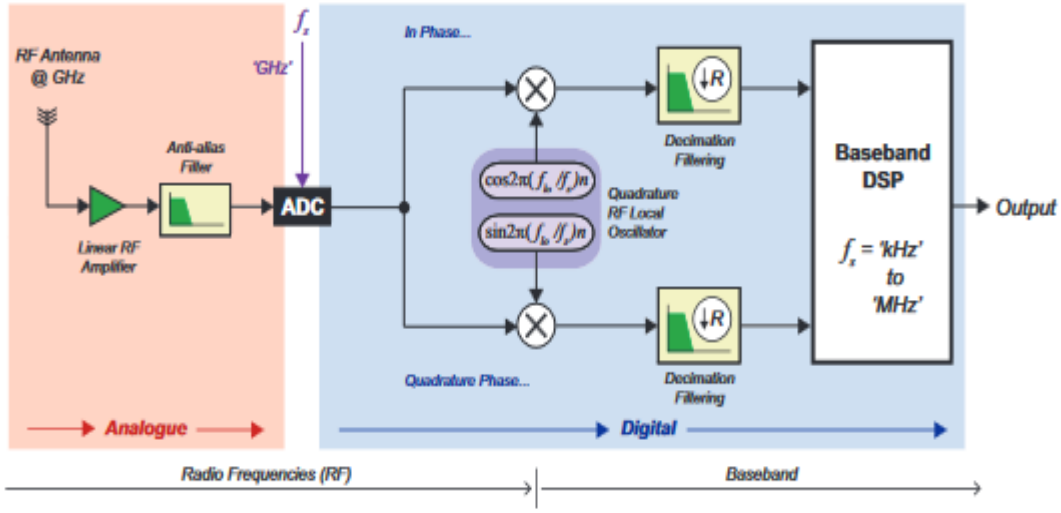
A 10. ábra a heterodin, analóg keverést alkalmazó vevőt mutatja be. A RF jelet először egy lokális RF oszcillátorral és sávszűrővel IF frekvenciára keveri, majd egy IF lokális oszcillátor segítségével előállítja a komplex burkoló jelének I és Q komponenseit. A megoldás előnye, hogy az AD-nek az alapsávra kevert jelet kell mintavételezni, így jellemzően elegendő pár MHz-es mintavételi frekvencia. Megemlítendő, hogy ezen megvalósításon az IF frekvencián történő keverést mindig ugyanarra az IF frekvenciasávra kell elvégezni, mivel hangolható sávszűrőt nehéz készíteni. Az AD átalakítás alapsávon történik, ennek következménye, hogy a jelet flicker zaj éri.

A 11. ábrán látható a heterodin, digitális keveréses megvalósítás, ez a legelterjedtebb SDR vevő architektúra. Az AD átalakítás IF sávban történik, az alapsávi ekvivalens előállítása pedig digitális tartományban. Ennek az elrendezésnek az előnye a

kevesebb alkatrész szám, továbbá a flicker zaj kiküszöbölése. A keverés megvalósítása chipen belül sokáig nem volt lehetséges, mivel az I és Q jelfeldolgozási ágnak szinkronban kell lennie, ilyen chipek sokáig nem voltak elérhetőek a piacon.

3.2.3 Direkt konverziós vevő

A direkt konverziós megvalósítás elterjedése a közelmúltra jellemző, mivel a megvalósításhoz szükséges gigahertzes nagyságrendben működő AD átalakítókhoz szükséges technológia már rendelkezésre áll. A megvalósítás nem keveri IF frekvenciára a rádió jelet, hanem közvetlen mintavételezi az RF jelet, utána digitális tartományban végzi el a kvadratúra keverést, ahogyan ez a 12. ábrán is látható.



12. ábra – Direkt konverziós vevő[5]

. Ezen felépítés előnye a heterodin vevővel szemben, hogy az RF frekvencián működő ADC segítségével az analóg RF áramköri részben sok komponenst meg lehet spórolni, hiszen nincs szükség középfrekvenciára való keverésre, így az ezekhez szükséges szűrőkre és erősítőkre sem. Ezzel a megvalósítással az SDR vevőnek mind az előállítási költségét, mind a fogyasztását csökkenteni lehet a közeljövőben.

3.3 Megvalósítási kihívások

3.3.1 Termikus zaj

Termikus zaj oka a T hőmérsékletre jellemző atomi szintű rezgések. Egy tetszőleges passzív hálózat kapcsai közti zajteljesítmény sűrűségét a Planck törvény segítségével jól le lehet írni, továbbá 30 GHz-es frekvenciatartomány alatt a függvény közelíthető a függvény Taylor sorának első tagjaival [9].

$$S(f) = \frac{hf}{e^{\frac{hf}{kT}}} \sim kT \tag{20}$$

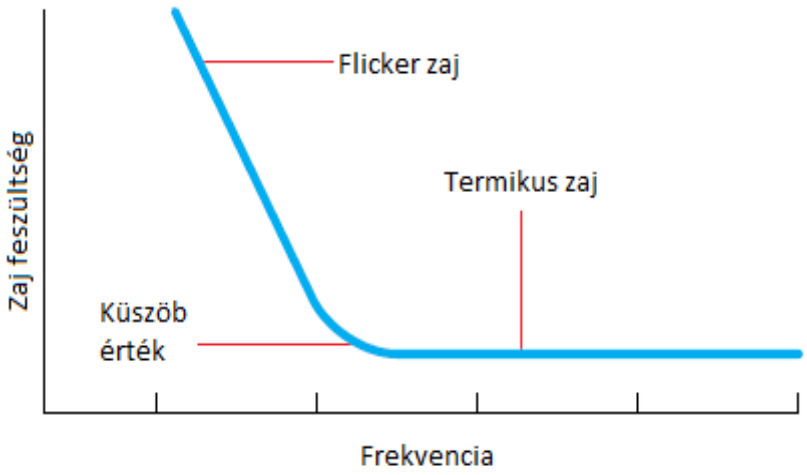
Mivel a gyakorlatban az átviteli sávban a zaj egyenletes eloszlásúnak tekinthető, ezért a zaj teljesítményét a B átviteli sávban a:

$$P_{zaj} \sim GkTB \tag{21}$$

alakban írható le, ahol G rendszerünk erősítési tényezője, mely konstansként közelíthető a B átviteli tartományban. A termikus zaj az adott B frekvenciatartományban közelíthető fehér zajként. A zaj elsősorban az analóg RF áramköri részben okoz gondot, ezért célszerű olyan erősítők használata, melyek a lehető legkisebb mértékben rontják a zajtényezőt és a jel-zaj viszonyt. Ezen erősítőket LNA-nak (Low Noise Amplifier) nevezzük.

3.3.2 Flicker zaj

A flicker zaj, vagy más néven $\frac{1}{f}$ zaj a MOSFET áramkörökre jellemző. A zaj jellemzője $\frac{1}{f}$-es spektruma, vagyis a zaj csak kis frekvenciákon egy küszöbfrekvenciáig szignifikáns, utána hatása elhanyagolható. A termikus és flicker zaj közti kapcsolatot a 13. ábra szemlélteti [8].



13. ábra - Flicker és termikus zaj

A flicker zaj spektruma miatt SDR rádióknál elsősorban analóg-digitális átalakításnál okoz gondot. Kiküszöbölését a heterodin SDR rádióknál ismertetett IF frekvencián történő mintavételezésével próbálták kiküszöbölni.

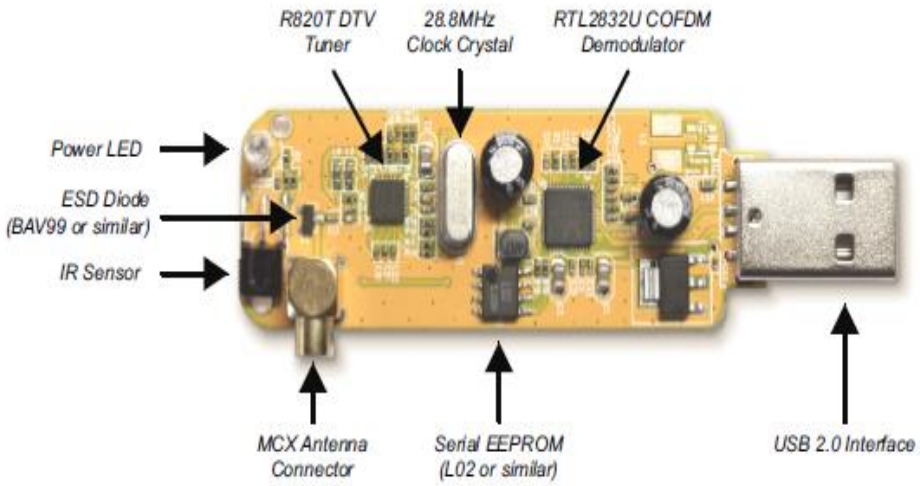
4 Beágyazott méréstechnikai rendszer

4.1 RTL SDR

A Diplomatervezés első félévében egy egyszerű RTL2832U chipsetre épülő USB-vel ellátott dongle-t használtam. Az eszköz a 3.2.2 fejezetben ismertetett heterodin vevős felépítésű, a keverést digitális tartományban végzi el, a komplex burkolóra alapozva. A dongle a technikai paramétereit a 1. táblázat mutatja be, míg a fő komponenseket 14. ábra mutatja.

Tuner chip	R820T
Demodulátor chip	RTL2832U
Működési frekvenciatartomány	24 – 1766MHz
A/D átalakítók bitszélessége	8 bit
Interfész	USB2
Maximális mintavételi frekvencia	3,2MHz
Antenna bemeneti impedancia	50 Ohm

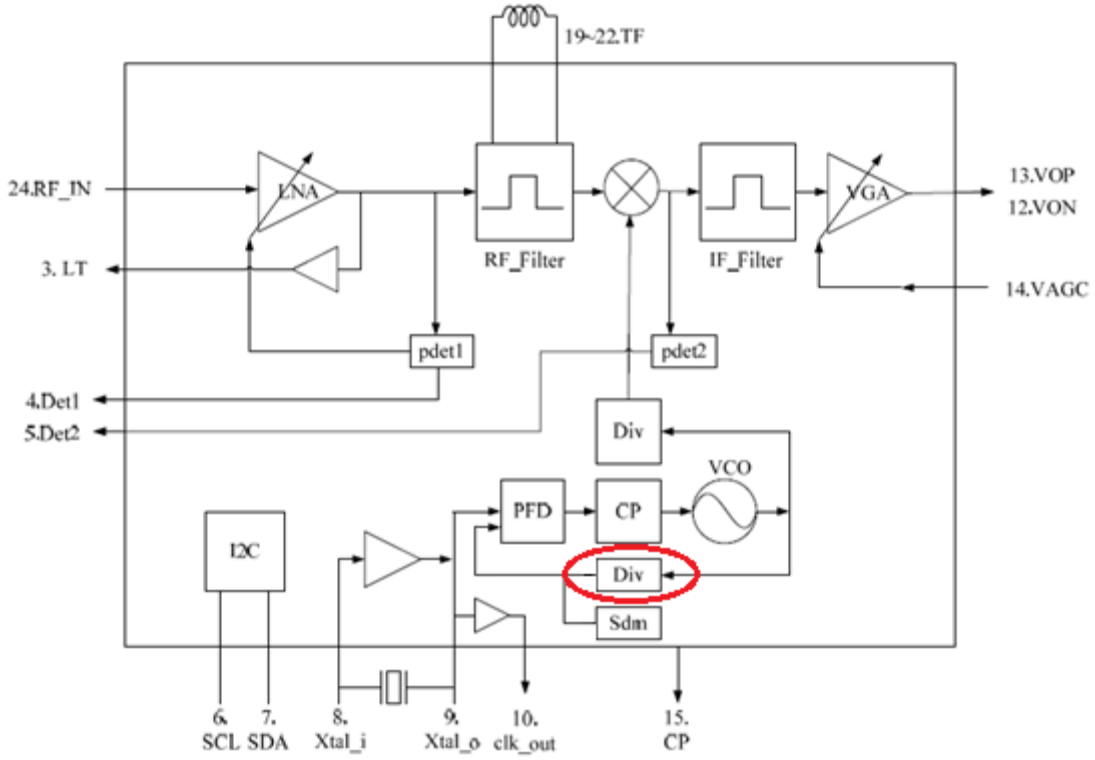
1. táblázat - RTL-SDR



14. ábra - SDR komponensek[5]

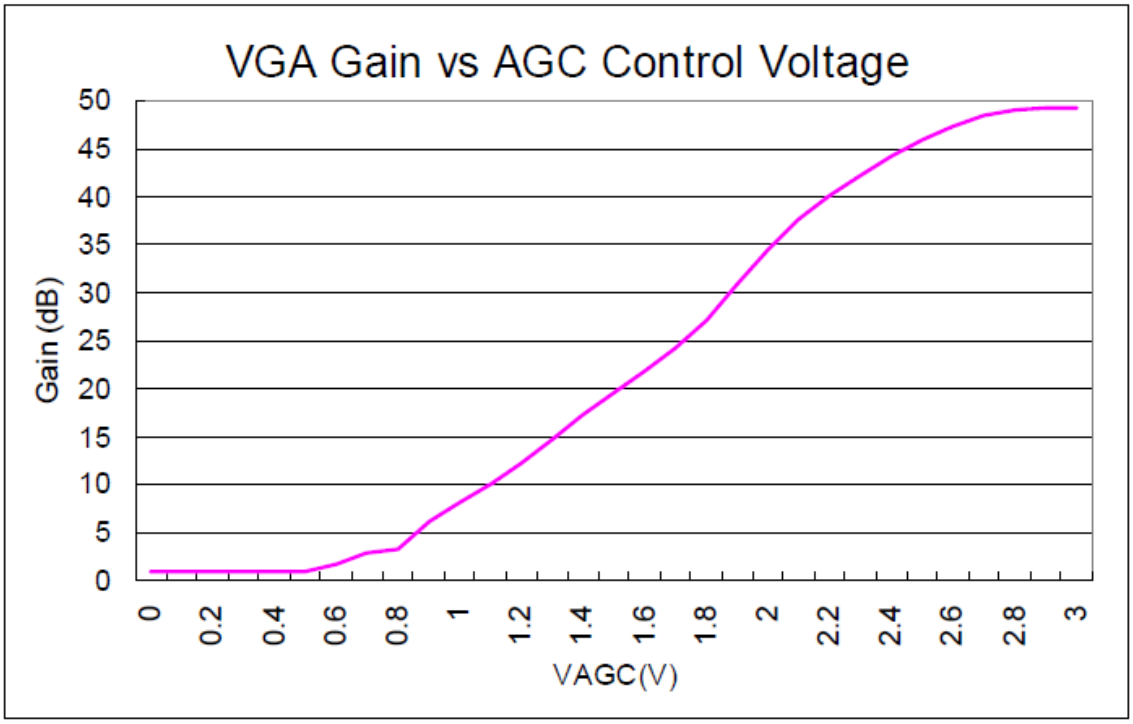
A dongle-ben két chip található: a Rafael Micro R820T és a RTL2832U. Mindkét chip egy közös 28,8MHz-es órajelet használ bemenetként, a működéshez szükséges frekvencia előállításáért a chipen belüli PLL és DDS felel. Az R820T feladata, hogy az

antennáról vett jelet erősítse és középfrekvenciára keverje. A 15. ábrán látható blokkvázlat szerint a bemeneti órajel frekvencia előállítása egy szintézer PLL segítségével történik. Az előállítandó frekvencia értékét a visszacsatoló ágon található osztó határozza meg, melyet a 15. ábrán pirossal jelöltem meg.



15. ábra - R820T blokkvázlat[7]

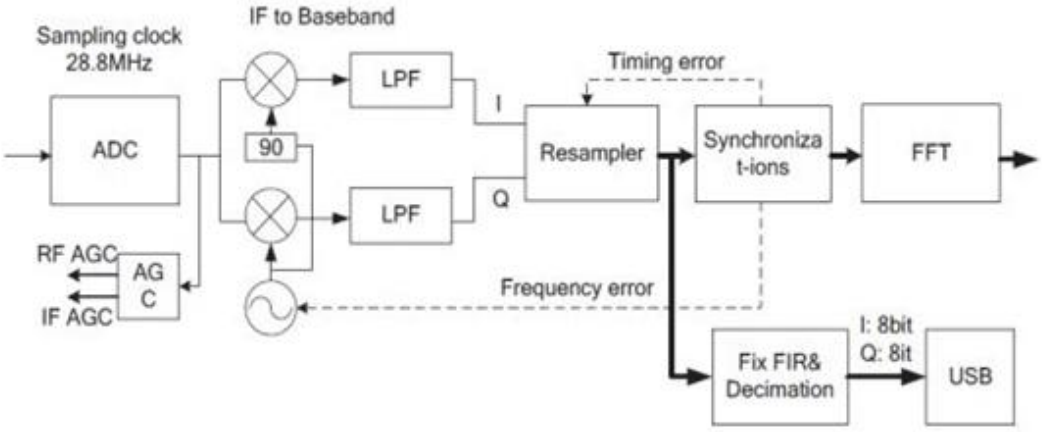
Az R820T-t a beérkező RF jelet egy LNA-val erősíti, a középfrekvenciára történő keverés előtt egy tükörszelektivitást biztosító szűrővel biztosítja, hogy az adott középfrekvencián a kiválasztott jel jelenjen meg. A középfrekvenciás jel végül egy sávszűrés után egy változtatható erősítésű erősítővel (VGA) tovább erősíthető. Mind az LNA és VGA erősítése szabályozható. Az LNA (Low Noise Amplifier) szabályzásáért egy power detector blokk felel, mely az ábrán pdet1 jelöléssel található. Az LNA erősítés szabályzásának a célja az optimális jelszint beállítása. A VGA (Voltage Gain Amplifier) az erősítést szabályzó feszültséget az ábrán láthatóan egy külső forrásból (RTL2832U) kapja vagy az I2C buszon keresztül programozott értéket. A VGA által elérhető erősítés értékek a 16. ábrán láthatók. A maximális erősítés érték 49,6 dB.



16. ábra - VGA erősítési ételek a VGAC függvényében [7]

A chip I2C buszon keresztül programozható. A beállítható paraméterek közé tartozik a belső szorzóra adott frekvencia értéke. A programozással beállíthatjuk a PLL által szintetizálandó frekvenciát, a szűrők sávszélesség, az erősítési értékeit az erősítőknek, illetve a szorzónak, illetve a AGC funkcióit az erősítőknek.

A RTL2832U chip fő feladata a középfrekvenciás jel A/D átalakítása, majd alapsávi ekvivalenssé alakítása. Ezek mellet usb slave interfészt szolgáltat a hoszt számítógépnek, illetve I2C interfészen keresztül továbbítja a tuner-hez a konfigurációs információkat. Mivel a dongle képes DVB-T dekódolásra is ezért feltételezhető, hogy a chipen belül valamilyen mikrokontroller vagy DSP található. A 17. ábrán látható blokkvázlat a chip fontosabb blokkjait kívánja illusztrálni, mivel azonban a gyártó nem tett elérhetővé dokumentációt, a chip pontos képességeit illetően nem rendelkezünk mindenre kiterjedő pontos információkkal [6].



17. ábra – Az RTL2832 chip blokkvázlata

Az A/D átalakítás 28,8MHz-en történik, a mintavételezett adatok felbontása 8 bit. Az alapsávi ekvivalens előállítása kvadratúra keveréssel történik. A digitális chipben az I és Q ág adatmennyiségének a csökkentése decimáló szűrő segítségével történik. A szűrt adatok az USB porton továbbításra kerülnek.

4.2 **Intelligens spektrumanalizátor tervezése Matlab segítségével**

A diplomatervezési feladatomat Matlab segítségével oldottam meg. A választás azért esett a Matlab-ra, mivel könnyű jelfeldolgozó algoritmusokat implementálni segítségével, továbbá a MathWorks support packag-et készített mellyel Matlab szkripttel, mind Simulink modellekkel programozni lehetett az SDR rádiót. A szoftvercsomag a Communications System Toolbox™ bővítményeként volt elérhető, Support Package for RTL-SDR Radio néven.

Az SDR vevő MATLAB-on való konfigurálásához és az adatok kinyeréséhez egy `comm.SDRRTLReceiver` nevű osztályt kellet példányosítani.

Az objektum segítségével az SDR rádió vételi paramétereit lehetett hangolni. A hangolható paramétereket az alábbi táblázat foglalja össze:

Paraméter	Leírás
`Centerfrequency`	Az vevő középfrekvenciája
`EnableTunerAGC`	Automatic Gain Controll

`TunerGain`	Erősítési tényező
`SampleRate`	Mintavételi frekvencia
`SamplesPerFrame`	Framen belüli mintaszám
`OutputDataType`	Kimeneti adattípus
`FrequencyCorrection`	Frekvencia korrekció

Az objektum inicializálása után adatokat a sdrrx() függvény meghívása után lehetett megkapni. A vett adat egy komplex vektort(frame). A tömb adattípusát az objektum inicializálása határozza meg, a tömb hossza a `SamplesPerFrame` paramétertől, míg a vett adat típusa az `OutputDataType` paramétertől függ. A függvény kimeneti argumentuma alaphelyzetben a rádiótól vett adat, de opcionálisan további három kimeneti argumentummal is visszatérhet:

- rxdata : a rádióból vett frame
- len : a frame hossza
- lost : hardveres feldolgozás során elvesztett adatok száma
- late : késleltetése a vett adatoknak

A megvalósítás során csak a vett adatokat tartalmazó rxdata paramétert használtam. Az adatok vétele után a release() függvény segítségével lehet az allokált objektumot törölni.

4.3 **A spektrumanalizátor specifikációja**

A Diplomatervként megvalósítandó spektrumanalizátort a korábban bemutatott RTL SDR dongle segítségével kellet megvalósítani. A cél egy olyan spektrumanalizátor megvalósítása volt, amely megmér egy adott spektrumtartományt, majd az adott spektrumtartományban megkeresi azokat a sávokat, melyek már foglaltak, továbbá megkeresi azokat a frekvenciasávokat, amelyek allokálhatók egy adott kommunikációhoz. Mivel előfordulhat, hogy az adott frekvenciasáv az időben nem folytonosan, hanem csak bizonyos időintervallumokban, (véletlenszerűen vagy periodikusan) foglalt, ezért a méréseket nem elég egyszer elvégzeni, hanem folytonosan kell végezni.

4.4 A spektrumanalizátor szoftveres rendszerterve

A spektrumanalizátor szoftverének fejlesztése a korábbi fejezetekben említett Matlab segítésével történt. A szoftver hierarchikusan felbontható egy GUI által támogatott top modulra és az alatta lévő modulok által meghívott függvényekre.

A top modul két függvényt hív meg, az egyik az RTL SDR-ből begyűjti az adatokat, míg a másik függvény feldolgozza a mérési eredményeket. A top modul a mérés és feldolgozás után, a kinyert adatokat grafikusan megjeleníti, illetve eltárolja lokálisan egy adott fájlban.

4.4.1 Adatfeldolgozást végző függvény

A megvalósítás első lépése egy olyan adatgyűjtő függvény megvalósítása volt, mely végig pásztáz egy adott spektrumot, majd az egyes spektrumrészeket összefűzi és megjeleníti. Fontos megemlíteni, hogy a végigpásztázás során azzal a feltételezéssel éltem, hogy a mérés időtartama kisebb, mint a csatorna koherencia ideje, vagyis mérés közben a csatorna nem változik.

Az adatgyűjtő függvénynek két funkciót kell ellátnia, az RTL SDR inicializálását, illetve az adott spektrum végig pásztázása közben az adatok begyűjtése. Ebben az alfejezetben ezeket az funkciókat ismetetem.

Inicializálás

Inicializálás során a adatfeldolgozó függvény a bemeneti paramétereket felhasználva segédváltozóakt hoz létre, melyekre az adatgyűjtés során szükség lesz. Az egyik ilyen segédváltozó a centerfrekvenciák tömbje mely azokat a frekvenciákat tartalmazza, melyekre a mérés során a SDR rádió hangolva lesz.

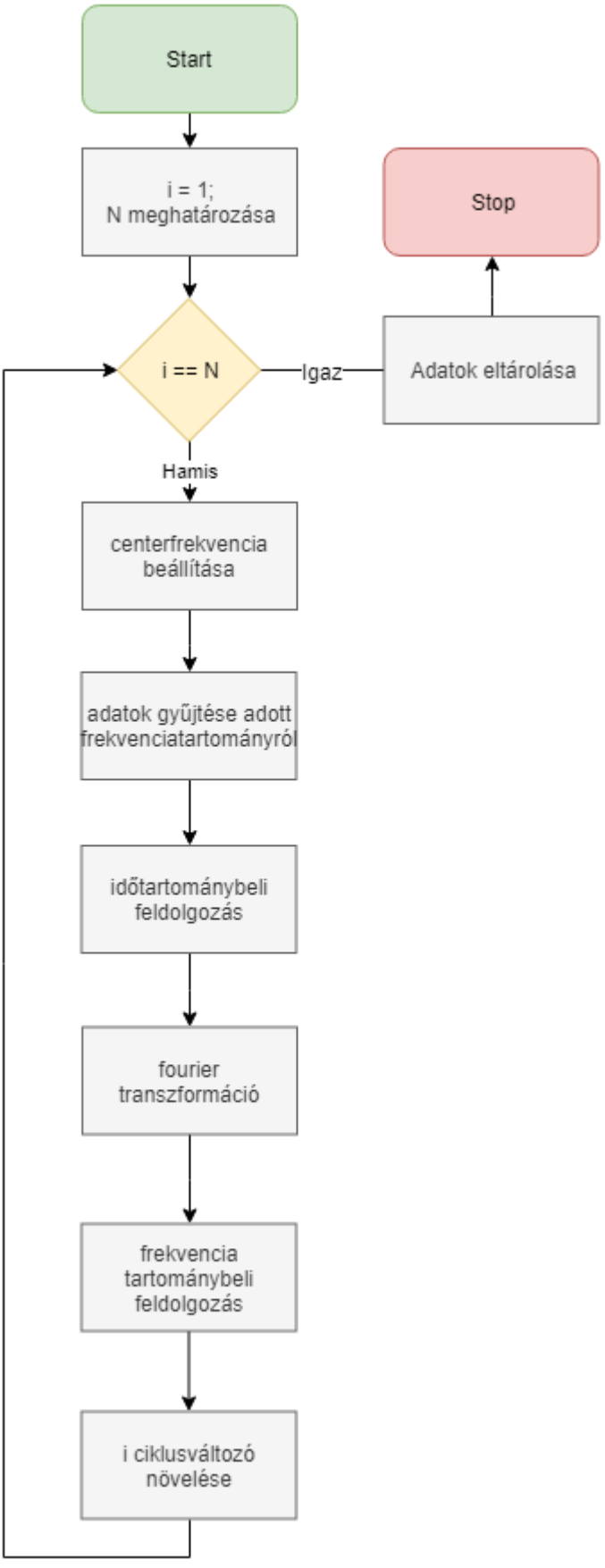
A tömb triviálisan előállítható a két határfrekvencia és a mintavételi frekvencia felhasználásával, hiszen a mintavételi tétel miatt a lépésközt a mintavételi frekvencia meghatározza. Figyelni kellet viszont, hogy a kezdő frekvenciából, a végső frekvenciáig nem lehet egész lépésközzel eljutni, ezért a tervezéskor úgy definiáltam, hogy addig a frekvenciáig kell mérni, ameddig éppen túl halad a specifikált végső frekvencián. Ehhez

a lépésközök számát a (22) egyenlet alapján határoztam meg, vagyis a lépésszám a mérendő frekvenciasáv és a mintavételi frekvencia hányadosa felfele kerekítve.

$$N = \left\lceil \frac{f_{stop} - f_{start}}{f_s} \right\rceil \tag{22}$$

Adatgyűjtést végző ciklus

Az adatfeldolgozó függvény magja egy for ciklus, melynek diagrammja a 18. ábrán látható. A ciklus egységnyi lépésközzel 1-től indul és N-ig számol, amely pont megegyezik a centerfrekvenciák számával. A ciklusváltozót a 18. ábrán i-vel jelöltem. A ciklusmagban a feldolgozás mindig centerfrekvencia beállításával kezdődik, csak ezt követően kezdi gyűjteni az információt az adott spektrumon, majd a begyűjtött adatokat Fourier transzformáció előtt kondicionálja (szűrés, ablakozás), majd Fourier transzformálja FFT segítségével. Mivel a transzformálandó adat komplex (I és Q komponens), ezért a transzformáció során a komplex jel abszolút értékét képezve történik a kiértékelés. A spektruminformáció kinyeréséhez nem szükséges az adott pillanathoz tartozó fázisérték, csak a komplex amplitúdó normája, ezért a fázisértékek nem lesznek eltárolva. A Fourier transzformált adatok végül ki lesznek átlagolva, ezáltal csökkenve a spektrum varianciáját. Az adatgyűjtő blokk a feldolgozott adatokat egy táblázatban eltárolja, amit az adatfeldolgozó blokk a későbbiekben fel tud használni.



18. ábra- Az adatfeldolgozást végző ciklus

4.4.2 Adatfeldolgozó függvény

Az adatok begyűjtése után a feladat, az adatok kvalitatív és kvantitatív kiértékelése. Első lépésként a begyűjtött spektrumrészeket kell összefűzni. A spektrumrészek összefűzése után hozzá kell rendelni a spektrumhoz egy értelmezési tartományt, azaz a frekvenciasávot, ahol a mérést végeztük. A frekvenciasáv felbontása FFT hosszától függ, illetve attól, hogy hányszor kell a centerfrekvenciát a mérés során változtatni. Az összefüggést a (23) egyenlet mutatja be, ahol n a teljes hossz n_{fft} a FFT felbontása.

$$n = \left\lceil \frac{f_{stop} - f_{start}}{f_s} \right\rceil * n_{fft} \tag{23}$$

A feldolgozó algoritmus feladatai közé tartozik a foglalt sávok felismerése, azok sávszélességének megmérése, továbbá a használt spektrum időbeni kihasználtsága. A kihasználtságot egy méréssel nem lehet megállapítani, annak kiszámításához ismétlődő mérések sorozata szükséges. Az adatfeldolgozás egyszerűsített blokkvázlata 19. ábrán látható.

19. ábra - Adatfeldolgozó függvény folyamatábra

A spektrumban a foglalt sávok meghatározására egy algoritmust dolgoztam ki, amely megkeresi a spektrumban a lokálisan maximális jelszinttel rendelkező pontokat, majd az

ezen csúcspontokhoz tartozó frekvenciától történő elhangolás mértékét (pozitív és negatív irányban), ahol a jelszint egy adott, előre meghatározott küszöbértékre csökken. Az így definiált sávot foglaltnak jelöljük. A küszöb értéke megegyezik egy zajszint jellegű mennyiség átlagértékével. A küszöbszint meghatározásával részletesen az 5.2.-es alfejezetben foglalkozom. Az egyes maximumpontok meghatározása addig történik, amíg van a spektrumban a küszöbszint fölött 3 decibellel még maximumérték. A használt sávok információit egy olyan struktúrában lesznek eltárolva, mely minden használt sávnak feljegyezi a maximumpontját, sávszélességét, az alsó, illetve felső határfrekvenciáját.

A felhasznált frekvenciasávok megállapítása után meg kellet állapítani, hogy az adott spektrumon hol vannak azok a szabad sávok, mely allokálhatók egy adott kommunikációhoz. Ehhez bemeneti paraméterként egy olyan sávszélesség paramétert használtam, mely szükséges az adott modulációt használó adáshoz.

A szabad sávok listája könnyen előállítható, mindössze a komplementerét kell venni az allokált spektrumok halmazának, a vizsgálandó frekvenciasávon. Ezt követően a kapott halmazból ki kell szűrni azokat a tartományokat, amelyek sávszélességben nem felelnek meg az adott követelménynek. A számítás végén kapott eredmények egy olyan táblázatban lesznek eltárolva, mely az adatgyűjtés során definiált spektrális lépésközönként tárolja az adott sávhoz rendelhető kihasználtságot. A tároláshoz használt táblázat a 20. ábrán látható, az ábrán ΔF a centerfrekvenciák közti távolság, a kihasználtság pedig egy 0 és1 közötti arányszám.

	1.	2.	3.	.....	N - 1	N
Sáv kezdőfrekvenciája	F_{start}	F_{start}+ 2ΔF	F_{start} + 2ΔF		F_{stop}- 2ΔF	F_{stop}- ΔF
Kihasználtság	U[1]	U[2]	U[3]		U[4]	U[5]

20. ábra - Kihasználtsági adatok tárolása

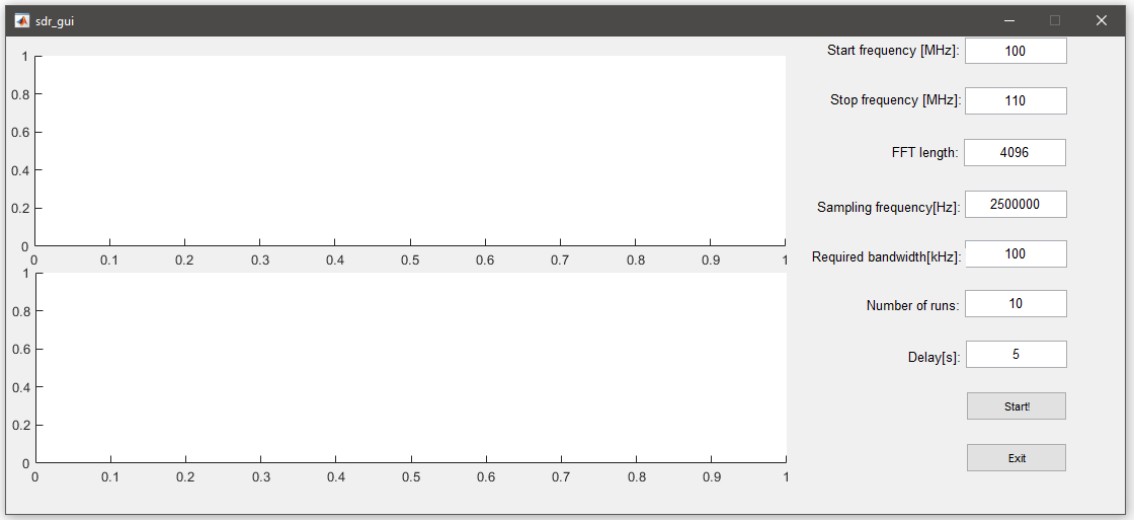
4.4.3 Vezérlő modul

A vezérlő modul feladata a mérés indítása, paraméterek átadása az alblokkoknak, továbbá az aktuálisan mért eredmények eltárolása mellet a kihasználtsági adatok frissítése felhasználva a korábbi mérések eredményét. A paraméterek megadását egy grafikus

környezeten keresztül lehet megadni, a megadható paraméterek az alábbi felsorolásban láthatók:

- Start frekvencia
- Stop frekvencia
- FFT hossz
- Mintavételi frekvencia
- Szükséges szabad sávszélesség
- Futtatások száma
- Futtatások közötti időköz

A grafikus interfész illusztrációja 21. ábrán látható, a grafikus interfész bal oldalán a mérési eredmények megjelenítésére szolgáló grafikus megjelenítők, míg jobb oldalán a megadható paraméterek listája.



21. ábra – Grafikus interfész

A mérések gyakoriságának meghatározása összetett kérdés. A mérés idejének gyakorisága nyilvánvalóan függ az adott sávról rendelkezésünkre álló *a priori* ismerettől, hiszen ahol tudjuk, hogy adás van ott nem szükséges mérést végezni. Viszont olyan sáv mérésekor, ahol nincs nekünk a priori ismeretünk, ott a mérések elvégzése ideje szintén kérdéses, hiszen előfordulhat, hogy egy hétig folyamatosan kell mérni, hogy hasznos jelet mérjünk a zajon kívül az adott csatornán. Az ITU ajánlása szerint [11] a mérések tipikusan 24 órásak szoktak lenni.

5 Az adatfeldolgozás megvalósítása

5.1 RTL SDT bekalibrálása

Az adatfeldolgozás első lépéseként a Matlabban inicializálható `comm.SDRRTLReceiver` osztály objektumát kellet létrehozni. Ahogy a korábbi fejezetben meg lett említve az objektum révén be lehetett hangolni a vételi paramétereket, mint például a centerfrekvencia, erősítés vagy a frekvencia korrekciós tényező. Ebben az alfejezetben a frekvenciakorrekció meghatározásáról, a megfelelő mintavételi frekvencia és erősítési tényező beállítását fogom ismertetni.

5.1.1 Frekvenciakorrekciós tényező

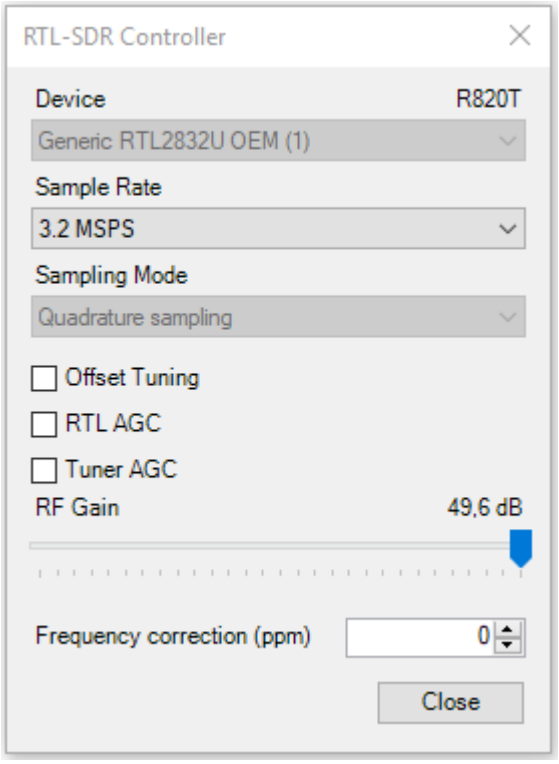
Az SDR rádióban alkalmazott alkatrészek, mint a lokál oszcillátor vagy a keverő, erősítők stb. nem ideálisak, az alkatrészek paraméterként megadott névleges értékeiktől eltérhetnek adott szórással. Lokáloszcillátor esetén a valós és a névleges frekvencia közti eltérés értéke néhány tized ppm-től akár százalékos nagyságrendig terjedhet. Az egyes alkatrészek hibáinak meghatározása körülményes és időigényes feladat. Ezért a frekvencia korrekciót érdemes úgy elvégezni, hogy a rádió által fogható spektrumban keresni kell egy olyan referencia jelet, melynek ismerjük a pontos frekvenciáját, majd a mérendő és mért jel frekvenciájának arányából megismerhetjük az eltérést.

A mérés elvégzéséhez fontos követelmény volt, hogy a SDR rádiót a mérés előtt fél órával be kellet kapcsolni, hogy elérje az üzemi hőmérsékletét az eszköz, és így hőmérsékleti tényezők ne befolyásolják a mérés eredményét.

A mérendő frekvencia kiválasztásánál fontos követelmény volt, hogy a mérendő jel minél nagyobb frekvenciájú legyen. Ennek az az oka, hogy nagyobb frekvenciákon a lokáloszcillátor frekvenciahibája erőteljesebben jelentkezik; például míg 50kHz frekvencia mellett a 10 ppm hiba csak 0.5 Hz elérést okoz addig 1 GHz-en esetén akár 10kHz-es hibát is okozhat.

Ennek függvényében a korrekció meghatározásához megfelelő alkalmazás volt a GSM hálózatokban a BTS-ek (Base TransceiverStation) által lesugárzott FCCH (Frequency Correction Channel) által sugárzott jel [12]. GSM sávok eredetileg 900MHz-en illetve 1800MHz-en találhatóak. Mivel a rádió névlegesen csak 1766 MHz-ig alkalmas vételre, ezért a méréshez a 900 MHz-es úgynevezett E-GSM (Extended GSM) bandet választottam. E-GSM nél a downlink sávok 925 és 960 MHz között találhatóak. Ebben a

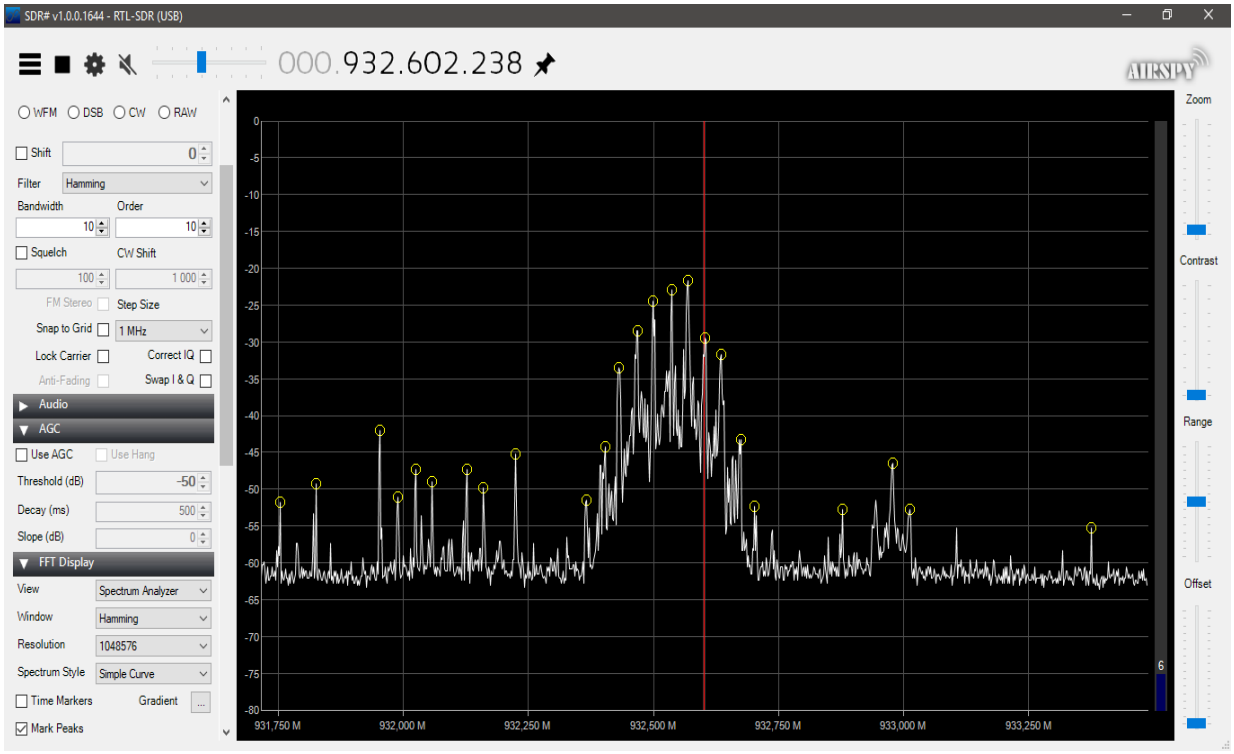
sávban a Downlink frekvenciák 200 kHz-es távolságban helyezkednek, az első downlink csatorna centerfrekvenciája 925.2 MHz, míg az utolsó csatorna 959.8 MHz. A BTS adótorony és mobil állomások között időosztásban, 577 us-os burst-tel kommunikál. 8 burst ad ki egy framet, és 51 framet ad ki egy úgynevezett Control Multiframet. A Control Multiframe első és utána levő minden 10. framejét használja az úgynevezett FCCH. Az FCCH burst-ök azzal a tulajdonságokkal rendelkeznek, hogy csupa 0-ás bitet tartalmaznak. Ennek illetve, hogy a GSM modulációjaként használt GMSK-nak (Gaussian Minimum Shift Keying) köszönhetően, FCCH burst-ök sugárzása estén a downlink csatorna centerfrekvenciája felett 67.7kHz-el egy szinusz fog sugárzódni. Ezt az információt kihasználva végeztem el méréseket, első lépésként megkerestem a legkisebb BTS által sugárzott downlink csatornát, hogy meghatározzam az offset hibájának az irányát, majd a kiválasztott downlink csatornáknál megmértem az FCCH burst-ök frekvenciáját. A méréshez az SDRSharp programot használtam, a mérés során, a 22. ábra szerint, a frekvencia korrekciót 0 ppm-re állítottam, a nyereséget a maximális 49,6dB-re. Sample rate-nek a maximális 3,2 MSPS-t választottam, továbbá az FFT mintaszámát 524288 pontosra állítottam, hogy a picket fence hatást csökkentsem.



22. ábra - SDRSharp konfigurációs menü

A mérés során az SDRSharp által megjelenített RF FFT GRAPHRA illetve az audió kimentre hagyatkoztam. Az audió kimenetre a centerfrekvencia 400Hz-es sugarában

elvégzett amplitúdómoduláció kimenetét vezettem. Az audió kimenet alkalmazása azért volt hasznos, mivel a körülbelül 10 burstönként megjelenő szinusz jel jellegzetes hangzása könnyen felismerhető. A SDR sharp kezelőfelülete a 23. ábrán látható. Az ábrán a korrekció előtt elvégzett mérés látható, a 932,6 MHz-es, 1012-es számú downlink csatornán, a spektrumon látható piros függőleges vonal a Tuning Bart jelöli, és mindig a centerfrekvenciát vesz fel, mely pontos értéke a menüsor baloldalán látható. A Tuning Bar a spektrumom épp a kívánt FCCH adásnál látható. Mivel az FCCH adások rövidek, ezért, hogy megtaláljam az adás centerértékét, az időbeli átlagolást maximumra növeltem.



23. ábra - SDR sharp kezelőfelülete

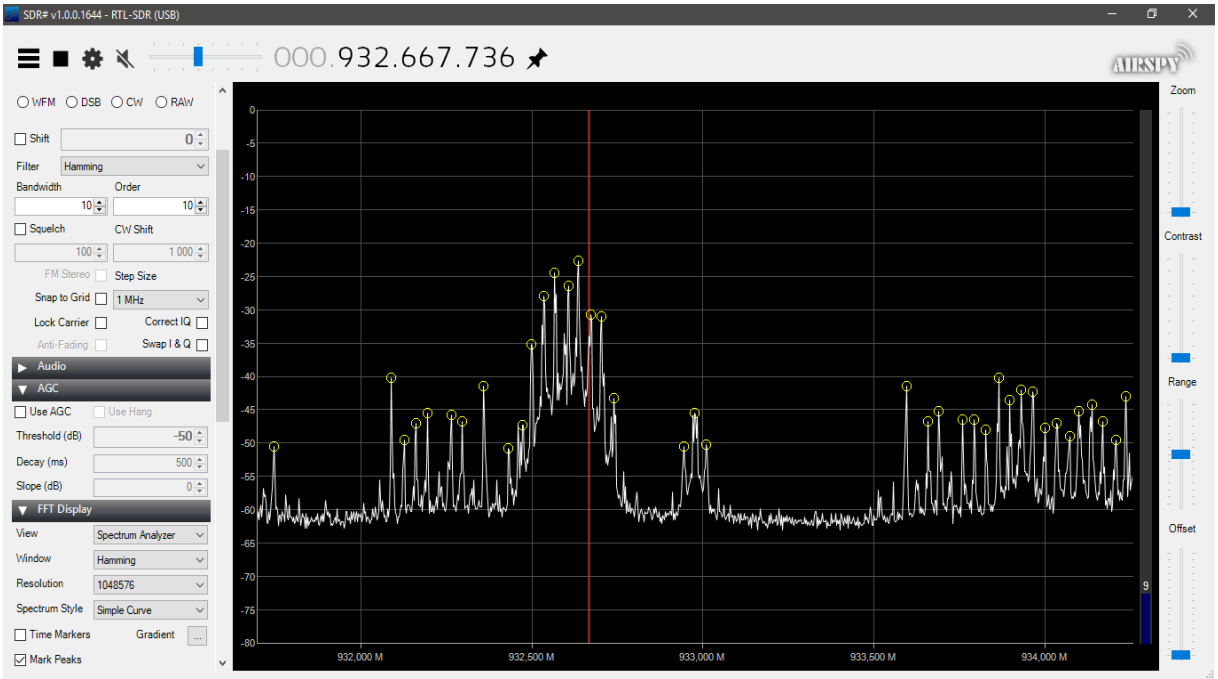
A mérési eredményeket a 2. Táblázatban láthatók, a mérés során az átlagos eltérés -66,5 kHz volt, az eltérésre átlagosan **71,002 pmm** jött ki.

Csatorna centerfrekvencia	Mérendő frekvencia	Mért frekvencia	Eltérés/Korrekciós tényező
932600000 MHz	932667708,3 MHz	932602238 MHz	70,202 ppm
934000000 MHz	934067708,3 MHz	933999615 MHz	72,905 ppm
934400000 MHz	934467708,3 MHz	934400819 MHz	71,585 ppm

934600000 MHz	934667708,3 MHz	934603514 MHz	68,686 ppm
935000000 MHz	935067708,3 MHz	935002500 MHz	69,741 ppm
937200000 MHz	937267708,3 MHz	937201280 MHz	70,879 ppm
939800000 MHz	939867708,3 MHz	939801216 MHz	70,751 ppm
940400000 MHz	940467708,3 MHz	940399860 MHz	72,148 ppm
940800000 MHz	940867708,3 MHz	940799386 MHz	72,622 ppm
941800000 MHz	941867708,3 MHz	941800873 MHz	70,965 ppm
942600000 MHz	942667708,3 MHz	942601216 MHz	70,541 ppm

2. Táblázat - Kalibrációs mérési eredmények

A korrekciós tényező kiszámolása után újra elvégeztem a mérést a 23. ábrán látható 1012-es számú ARFCN (Absolute Radio-frequency channel number) csatornán. A 24. ábrán átható, hogy korrekció után a mért FCCH csatorna körülbelül 28 Hz-el tér el a mérendő értéktől, ez körülbelül 29 ppb eltérés. A maradó hibának a feltételezhető oka a leolvasáskor elvégzett hiba.



24. ábra - SDRSharpos mérés korrekció után

5.1.2 Erősítés és Mintavételi frekvencia

Matlab használatakor a SDR rádió erősítési tényezője a `TunerGain` és `EnableTunerAGC` paraméter segítségével állítható, paraméterként az erősítés érékét kell decibelben megadni.

A rádióban az erősítésért az R820T chip felel, melyet a rádión belüli I2C buszon keresztül lehet programozni. A chipen belül az erősítést a LNA a mixer és RF-AGC (Radio Frequency Automatic Gain Control) komponens végzi. AGC esetén az erősítés állítása a beérkező jel teljesítménye alapján történik.

Habár a RTL2832U chip nem végez erősítést, ennek ellenére ez a chip is rendelkezik egy belső AGC komponenssel, melynek a mintavételezésnél vett jel SNR-je alapján próbálja állítani az erősítés értéket.

Az AGC-k használata el lett vetve, mivel az egymás melletti csatornákat eltérő módon erősíti, így a feldolgozáskor a maximumpontok illetve alapzajszint meghatározása problémákat okozhat.

Az erősítés beállításának pontos metódusa nem volt elérhető, a Matlab szkript inicializáláskor a `void setTunerGain(double gain);` driver függvényt hívta meg. Interneten elérhető driverek, és a matlab dokumentáció alapján tanulmányozása által feltételezhető, hogy a megadott lebegőpontos számot egy listában elérhető erősítési érékek közül a legközelebbihez rendeli, majd az adott erősítési érték szerint állítja az LNA és a szorzó erősítését.

Az elérhető erősítések értékét Matlabon keresztül az `infostruct = info(rx)` parancs segítségével értem el, eredményként az alábbi látható információkat nyertem ki.

```
infostruct =
    The following values show radio settings, not the property
    values of RTL-SDR receiver System object. For more information,
    type 'help comm.SDRRTLReceiver'.

               RadioName: 'Generic RTL2832U OEM'
            RadioAddress: '0'
               TunerName: 'R820T'
            Manufacturer: 'Realtek'
                 Product: 'RTL2838UHIDIR'
              GainValues: [0 0.9000 1.4000 2.7000 3.7000 7.7000 8.7000
12.5000 14.4000 15.7000 16.6000 19.7000 20.7000 22.9000 25.4000 28 29.7000
32.8000 33.8000 36.4000 37.2000 38.6000 40.2000 42.1000 43.4000 43.9000
44.5000 48 49.6000]
      RTLCrystalFrequency: 28800000
    TunerCrystalFrequency: 28800000
             SamplingMode: 'Quadrature'
```

```
          OffsetTuning: 'Disabled'
        CenterFrequency: 88900000
             SampleRate: 250000
    FrequencyCorrection: 0
```

Az elérhető erősítések értéke 0 és 49.6 dB közötti étékek lehetnek, ezek az érékek a R820T adatlapjával összhangban vannak. Mivel az alkalmazás célja, hogy minél több használt sávot találjon, és a torzítás mértéke kevésbé volt fontos, az erősítési tényezőt maximálisra állítottam.

5.1.3 Mintavételi frekvencia

A beállítható paraméterek közül fontos tényező volt még a mintavételi frekvencia. A Matlab által támogatott mintavételi frekvenciák 225-300 kHz és 900 – 3000 kHz között voltak. Fontos megemlíteni, hogy amíg a mintavételi frekvenciának az állításához a RTL2832U chipet kell programozni, ezzel párhuzamosan a R820T tuner chipben az aluláteresztő szűrő sávszélessége is módosul. Interneten elérhető driverekben a maximális sávszélesség értéke 1.7MHz, az ehhez tartozó mintavételi frekvencia pont 3MHz, így feltételezhető, hogy átviteli karakterisztika nem lesz feltétlenül lapos, a mintavételi frekvencia feléhez közeledve csillapítás lesz várható.

A mintavételi frekvencia mellet állítható paraméter volt a frameknek a hossza is, vagyis egy frameben hány I és Q komponenst reprezentáló minta érkezik meg. A paraméter beállításakor megfigyelhető volt, hogy a mintavételi frekvencia növelése esetén a csomagszám elkezdett csökkeni. Ezt az adatgyűjtés során alkalmazott `step()` függvény `loss` visszatérési értékének vizsgálatával volt megállapítható. A loss rate megfigyelése alapján arra jutottam, hogy 280 FPS (Frame per Second) mellet stabilan adatvesztés nélkül történik a mintavételezés. Ez gyakorlatban azt jelenti, hogy körülbelül 2,3Mhz mintavételi frekvencia estén a maximális frame nagyság 8192.

5.1.4 Bemeneti adat típusa

Az comm.SDRRTLReceiver osztálykor meg kell adni, hogy a rádióból érkező adat milyen formátumban legyen. Három lehetőségek közül lehetett választani:

- 'int16'

- 'double'
- 'single'

A tervezés során a ’single’ adattípust választottam. Ezen adattípus mellet a beérkező adat komplex, 1 és -1 közötti értékeket vehet fel. Az adatokat feldolgozás során dB skáláztam, viszont teljesítmény jelzőt nem rendeltem értékekhez, így a későbbiekben a mérési eredményeket viszonyszámként decibelben adom meg.

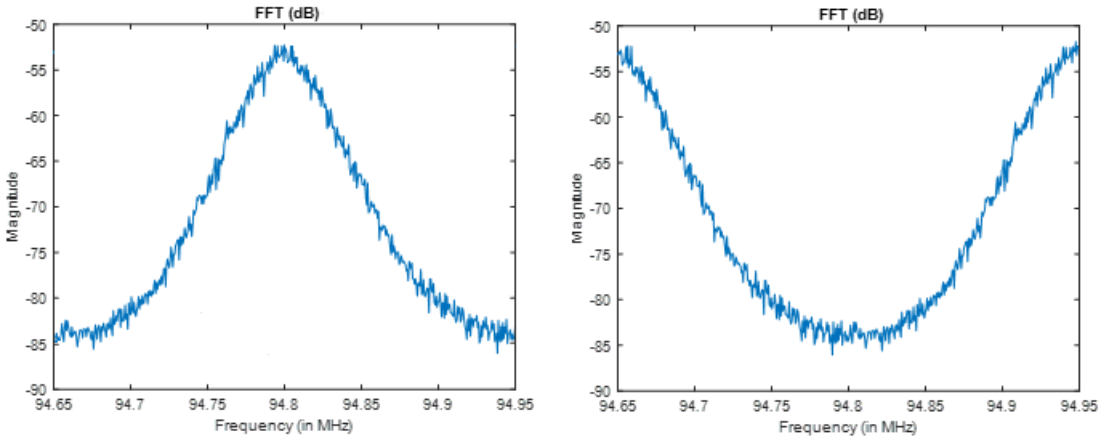
5.2 **A rendszerterv szerinti implementáció**

Az adatfeldolgozó algoritmust a rendszerterv szerinti implementáció megvalósításával kezdtem. A kezdeti paraméterek megadását követően, a center frekvencia értékeket határoztam meg, az értékékeket egy tömbben tároltam. A kezdő frekvencia a startfrekvenciaként megadott paraméter, a többi érték pedig a start frekvencia fölött, a mintavételi frekvencia egész számú többszöröseként találhatóak, egészen a stop frekvenciáig.

```
tuner_freq       = start_freq:(rtlsdr_fs):stop_freq;
```

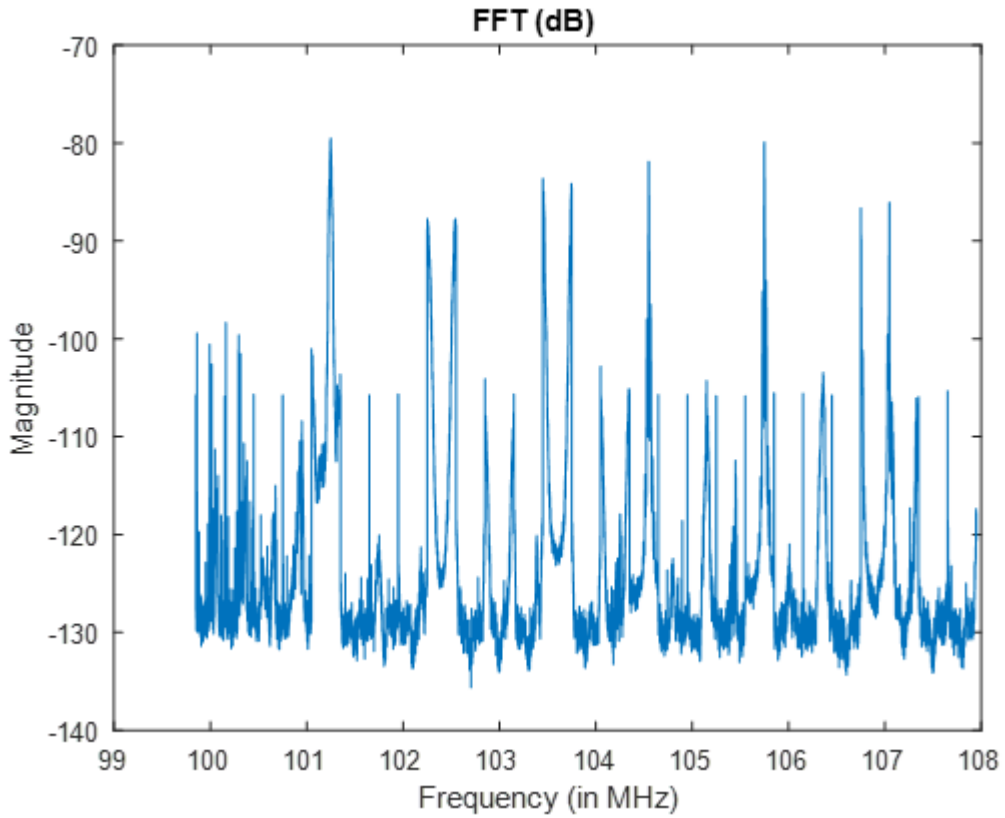
Ezt követően egy ciklusban elkezdtem feldolgozni adott sávonként a spektruminformációt. A ciklus végigment az összes tuner frekvencián, azt beállította aktuális centerfrekvenciaként, majd minden sávból 32 framet gyűjtött össze. Minden framenek vettem a Fourier transzformáltját, majd egy átmeneti bufferben eltároltam a mérési eredményeket. A Fourier transzformációt a Matlab beépített `fftshift` függvényével végeztem.

Az `fftshift` függvény ugyanazzal az algoritmussal végzi el a Fourier transzformációt, mint az `fft` függvény. A különbség, hogy az `fftshift` függvény gondoskodik arról, hogy a [-B,0] tartomány a fft tömb elejére, míg [0,B] tartomány a fft tömb végére kerüljön, ahol B a mintavételi frekvencia fele. A `fft` és `fftshift` függvény közti különbséget a 25. ábrán illusztráltam, a mérés során a centerfrekvenciát 94,8 MHz-re állítottam, a mintavételi frekvenciát pedig 300 kHz-re. A 25. ábra bal oldalán a `fftshift`-el elvégzett DFT eredménye, míg a jobb oldalon a `fft`-vel végzett DFT eredménye látható.



25. ábra – Centerfrekvencia körüli spektrum az FFT tömb korrekciójáva(bal ábra) és korrekciója nélkül(jobb ábra)

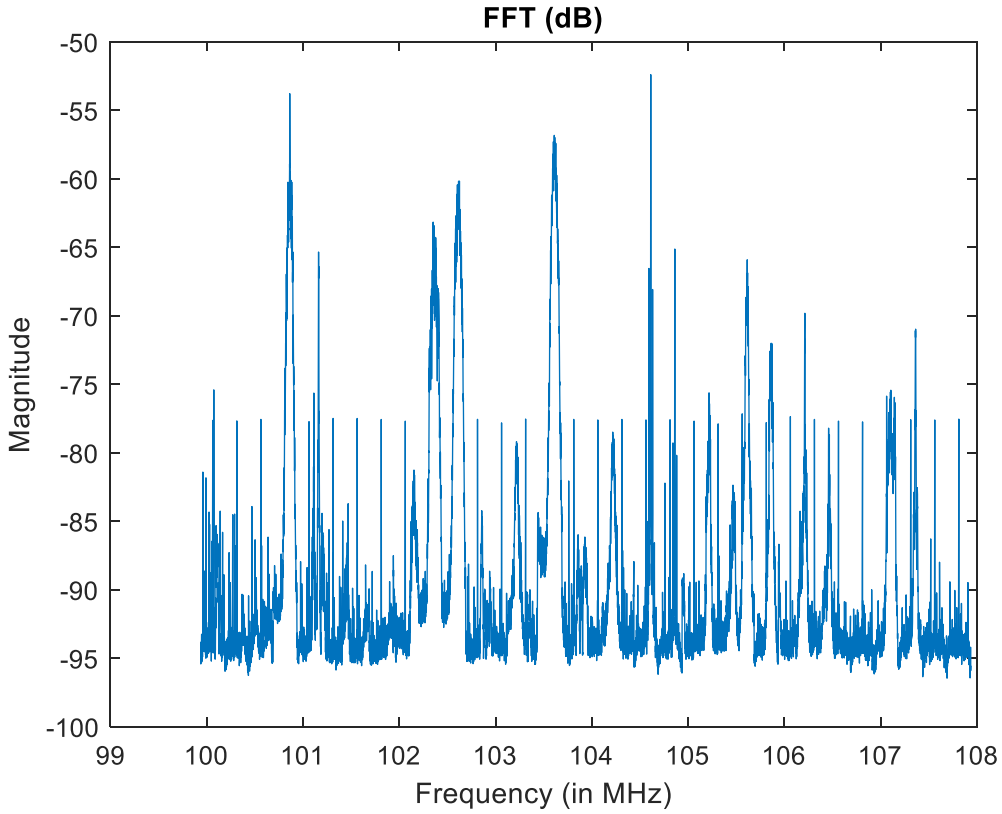
A 32 mérési adat begyűjtése után minden FFT rácspontra a 32 mérés átlagát vettem. Az átlagolás segítségével a mért spektruminformáció varianciáját próbáltam csökkenteni. A ciklus lefutását követően a sávonként eltárolt spektruminformációkat összefűztem egy nagy tömbbe, majd `plot()` segítségével ábrázoltam a spektrumot. A spektrumértékekhez tartozó frekvenciákat `linspace()` függvénnyel generáltam. Teszteléshez 100 és 108 MHz közti frekvenciasávot vállaszottam, mivel azon könnyű validálni a teszteredményeket, a frekvenciasávon található FM adók miatt. Mintavételi frekvenciát 3MHz-re állítottam, Az FFT felbontását 4096 pontosra állítottam. A mért spektrum a 26. ábrán látható. A spektrum megjelenítésekor az amplitúdó érétkeket dB-ben adtam meg. Az ábrán feltűnik, hogy bizonyos spektrumalakzatok ismétlődnek, egymás után kétszer jelennek meg.



26. ábra - rendszerterv szerinti megvalósítás

A probléma oka az volt, hogy a Matlab konfigurációs fájljába be volt állítva az, hogy bufferelve legyenek az adatok. A bufferelés sajnos nem volt beállítható inicializáláskor, a szükséges konfigurációs fájlt debugolással kerestem meg. Inicializáláskor break pontot raktam a inicializációhoz, majd „step into” funkcióval megkerestem azt a kódsort, ahol az bufferelés megtörténik.

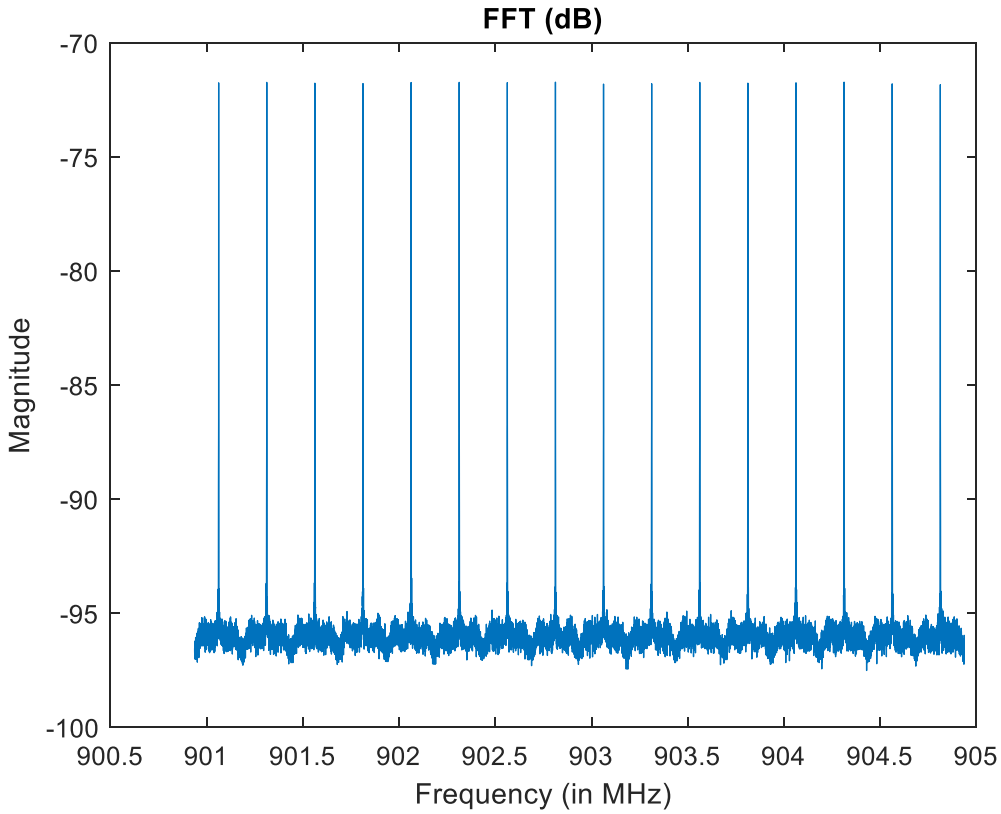
A szükséges konfigurációkat RTLSDRFullDriverConfigT.m fájlban találtam meg. A fájlban van egy olyan függvény, amely egy struktúrával tér vissza előre definiált paraméterekkel. A struktúra egyik tagja a `'NumUSBBuffers'` volt melynek az alapértelmezett értéke 32 volt. A paraméter 0-ra állításával, megismételtem a tesztet, és ahogy a 27, ábrán is látszik a probléma megoldódott.



27. ábra - FM csatornák vizsgálása rendszerterv modell szerint

A 27. ábrán látható mérési eredmény tanulmányozásakor, két probléma is előkerült. Az egyik ilyen probléma, hogy a centerfrekvenciák helyén egy tüskeszerű impulzus kerül elő, amplitúdója közel állandó, mindig hozzáadódik a mért jel spektrumához. A másik probléma az, hogy a centerfrekvenciáktól fs/2 távolságra a mért spektrum erősen csillapodik mindkét irányban. A két feldolgozási jelenség jól megvizsgálható, ha egy olyan sávot vizsgálunk meg, ahol nincsenek hasznos jelek csak konstans zaj. Ehhez a 901-905 MHz-es ISM sávot vizsgáltam meg úgy, hogy az SDR rádió antennáját kevettem. A mérési eredmény a 28. ábrán látható. Az ábrán látható mérési eredményről leolvasható,

hogy a maximumpontok -71,7 és -71,6 dB között ingadoznak, az átlagzajszint -96 dB, míg a minimum -97,5 dB.



28. ábra - Zajos csatorna mérése

A továbbiakban egyesével megvizsgáltam a centerfrekvencia körüli értékeket. A centerfrekvencia körüli kiemelkedés keskenysáv, átlagosan 1,1 kHz sávszélességűnek mértem. A spektrum többi szakaszát vizsgálva megállapítottam, hogy az erős csillapodás 100 kHz frekvencián történik 300 kHz mintavételi frekvencia mellet. A 28. ábrán látható mérések különböző mintavételi frekvenciával, illetve más frekvenciasávon való megismétlése hasonló eredményeket hozott. Mind a centerfrekvenciánál található kiugrás, mind az aluláteresztő hatás megjelent az eltérő paraméterekkel való tesztelés során.

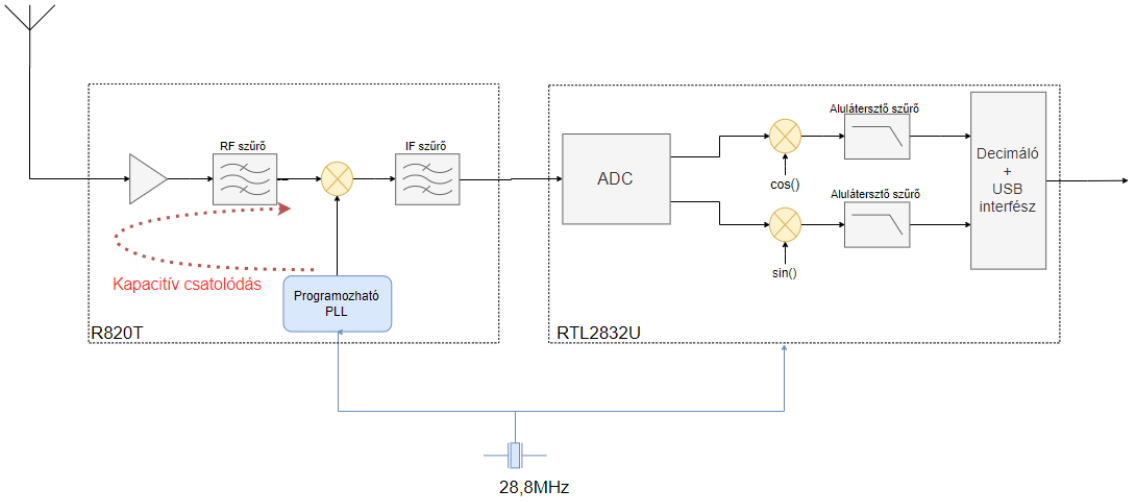
.

5.2.1 DC ofszet oka és kiszűrése

A DC ofszetnek több lehetséges oka lehet. Az egyik gyakori ok szokott lenni a lokálszivárgás, a másik pedig a tápzaj megjelenése az AD bemenetén. Szintén lehetséges

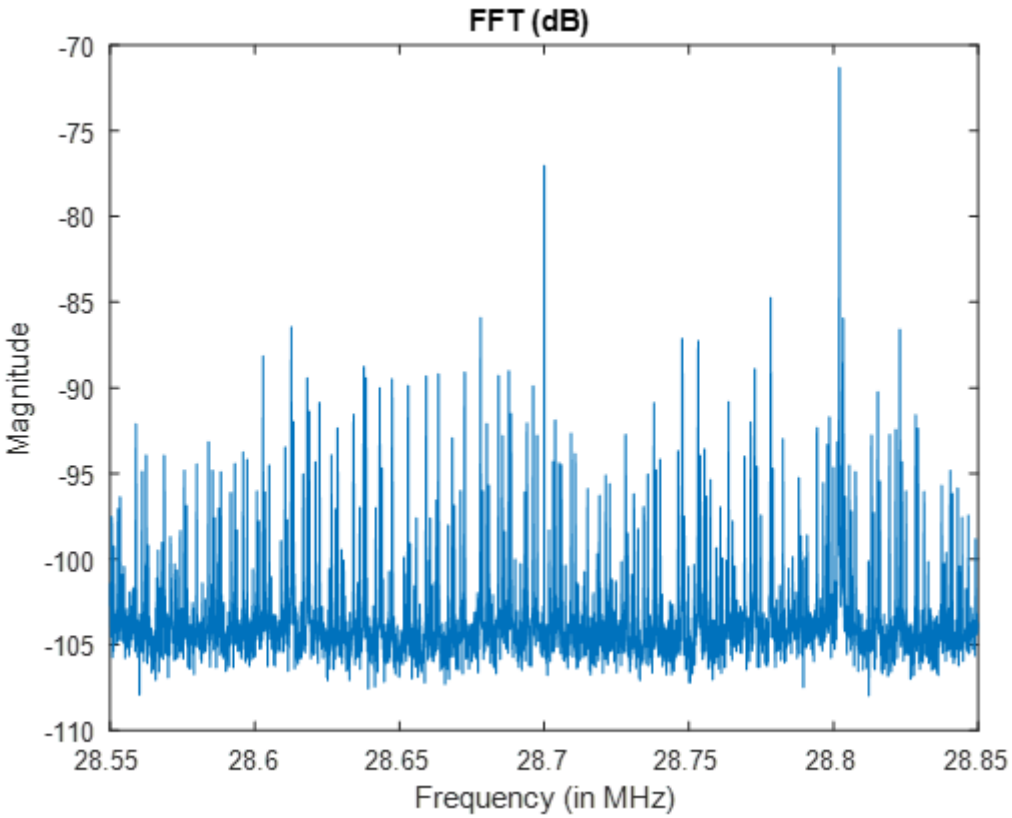
ok, hogy a bemeneti adatunk véges hosszú (SamplesPerFrame), ezért a levágott szinusz komponensek DC ofszetet okoznak. Megemlíthető okként a korábbi fejezetben ismertetett Flicker zaj van jelen.

Lokálszivárgás rádiófrekvenciás eszközökben tipikusan mixerek miatt lép fel, az analóg mixer bemenetei között tipikusan kapacitív csatolódás lép fel, ezáltal mixelés eredményeként a lekeverendő jelbe belekeveredik a mixeléshez használt szinusz, aminek az eredménye egy DC komponens. Diplomatervemhez használt SDR rádióban analóg mixer az R820T chipben található. A lokálszivárgást nagy valószínűséggel a chipen belüli kapacitív csatolás okozza. A lehetséges pontokat, ahol a kapacitív csatolás előfordulhat, az R820T chipben, a 29. ábrán látható.



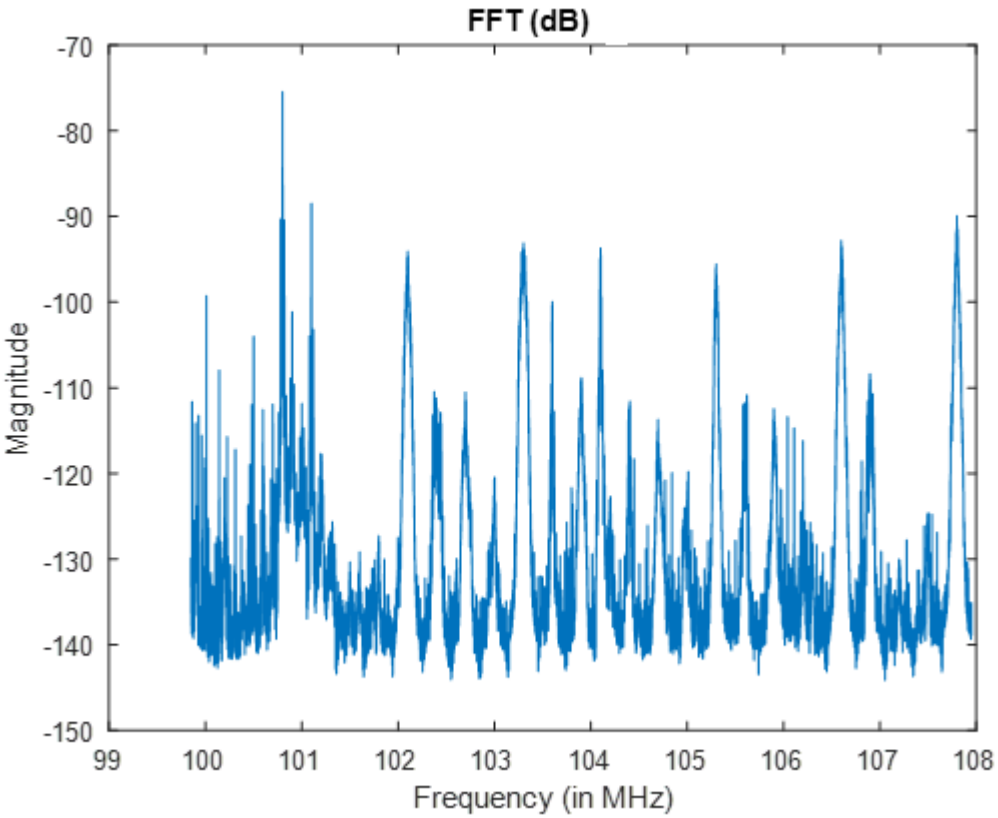
29. ábra - Kapacitív csatolódás

Megjegyzendő azonban, hogy egy hasonló kapacitív hatás kép fel akkor is, ha a spektrumot 28,8MHz környékén vagy annak egész számú többszörösérén vizsgáljuk. Ilyenkor az SDR dongle órajelforrásának a kapacitív csatolása tapasztalható. A jelenség a 30. ábrán látható.



30. ábra - Lokálszivárgás 28,8MHz-en

A DC offszet eliminálására többféle lehetőség van. Egy kézenfekvő megoldás egy aluláteresztő vagy mozgóablakos átlagoló implementálása. A szűrős megvalósítások problémája, hogy a lokálszivárgás kiszűrésével, a centerfrekvenciához közel torzulni fog a spektrum. Diplomatervemben egy egyszerűbb megoldást implementáltam, az adatfeldolgozó ciklusban a begyűjtött időbeli információból kivontam annak az időbeli átlagát. Ennek következménye a centerfrekvenciákon megjelenő negatív kiugrások. Ez problémát jelentet az adatfeldolgozó algoritmus tesztelése során, mely egy küszöbértékig számolja egy allokált sáv sávszélességét. A problémát úgy próbáltam orvosolni, hogy az „elveszett” DC tagot a szomszédos tagokat felhasználó köbös spline metódust használtam. Az interpoláció harmadfokú polinomokkal próbálja a hiányzó pontot meghatározni. Az algoritmus segítségével pontosabb becslést kapok, mint lineáris interpoláció segítségével, cserébe több számítási időt igényel. A lokálszivárgás mentes spektrum a 31. ábrán található.

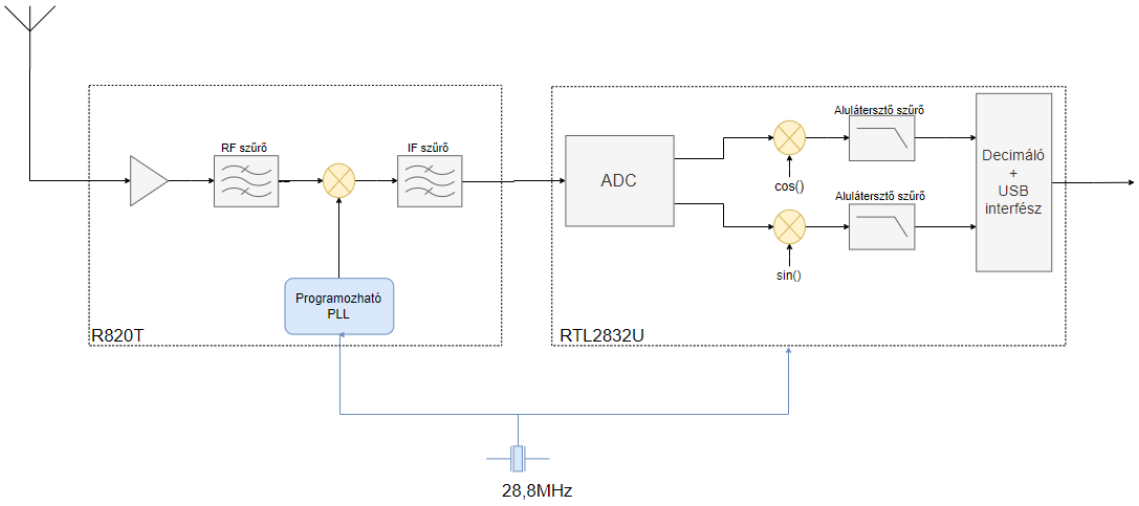


31. ábra - Lokálszivárgásmentes spektrum

5.2.2 Átviteli karakterisztika korrekciója

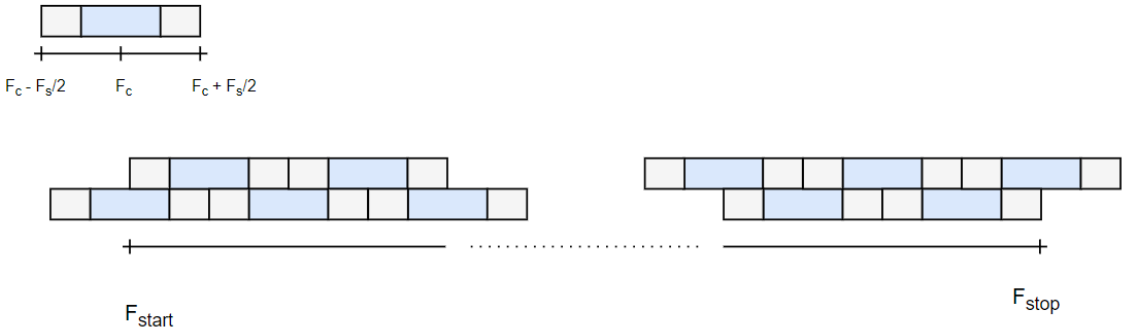
A 28. ábra vizsgálatával megfigyelhető, hogy a rádió által begyűjtött spektruminformáció 3 MHz-es mintavételi frekvencia esetén 1 MHz környékén erősen elkezd csillapodni, ami miatt torzul a mért spektrum. A mérést kisebb mintavételi frekvenciákon való elvégzése hasonló eredményeket produkál, a mintavételi frekvencia feléhez közeledve a spektrum elkezd torzulni az erős csillapítás miatt.

A jelenség okozója az R820T-ben alkalmazott sávszűrő, illetve a RTL2832U aluláteresztő és decimáló szűrője. A tuner sáváteresztő szűrője a mintavételezés miatt megjelenő átlapolódást próbálja megakadályozni, míg az RTL2832U chipben a szűrést a mixelés következtében létrejött magasabb frekvenciájú komponensek kiszűrésére alkalmazzák. A rendszerben látható szűrőket a 32. ábrán szemléltettem.



32. ábra - Az SDR rádió felépítése

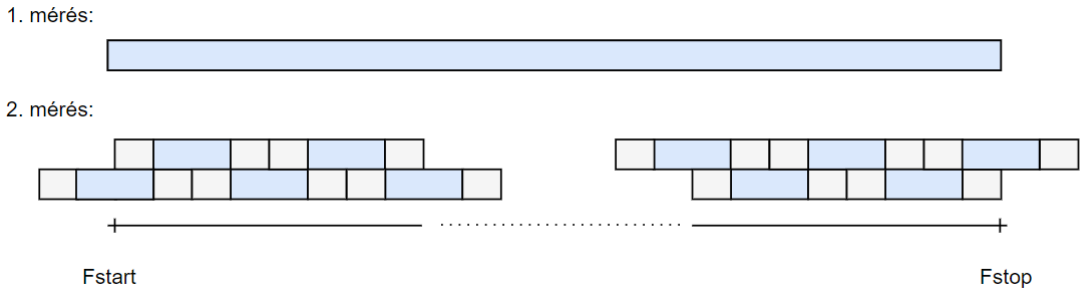
A csillapítást az feldolgozást végző algoritmus átdolgozásával oldottam meg. Az eddigi Fs lépésköz helyett Fs/2 lépésközönként mértem spektrumot, egyenletes lépésközzel. Az egyes mérésekből csak a Fs/4 sávszélességig tartottam meg az információkat – a többi információt eldobtam. A módosítás illusztrációja a 33. ábrán látható. A felhasznált spektrum információkat kékkel, míg az elhanyagoltokat szürkével jelöltem.



33. ábra - Módosított adatgyűjtés

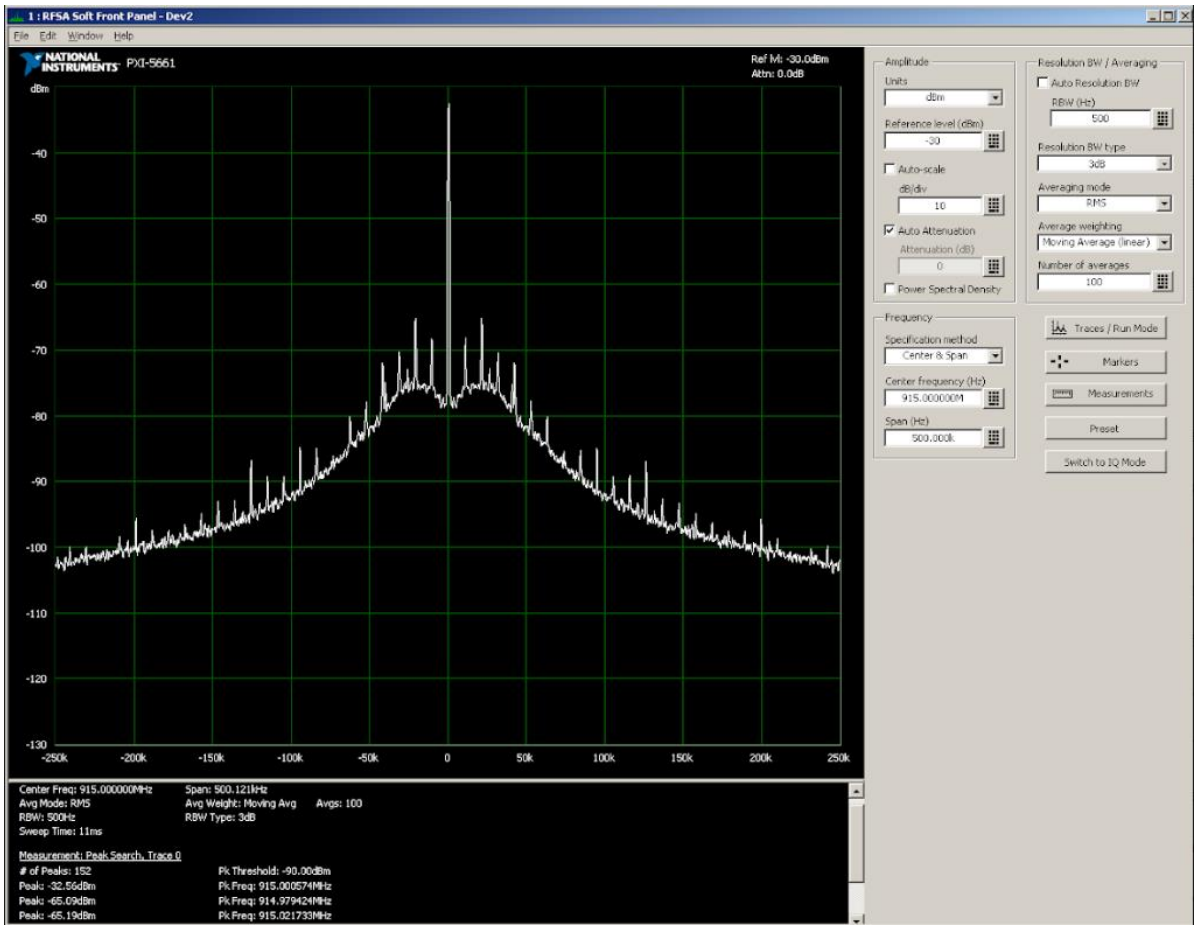
5.2.3 Az adatgyűjtés validációja

Az implementált adatgyűjtő függvény helyességéről egy olyan teszt segítségével győződtem meg, melyben össze-komparáltam, hogy egy centerfrekvencia körül maximális mintavételi frekvenciával begyűjtött spektruminformáció megegyezik-e azzal, mintha az adott centerfrekvencia körül kisebb mintavételi frekvenciákat alkalmazva több spektrumszeletből áll össze az adott spektruminformáció. A validációs tesztet a 34. ábra szemlélteti.



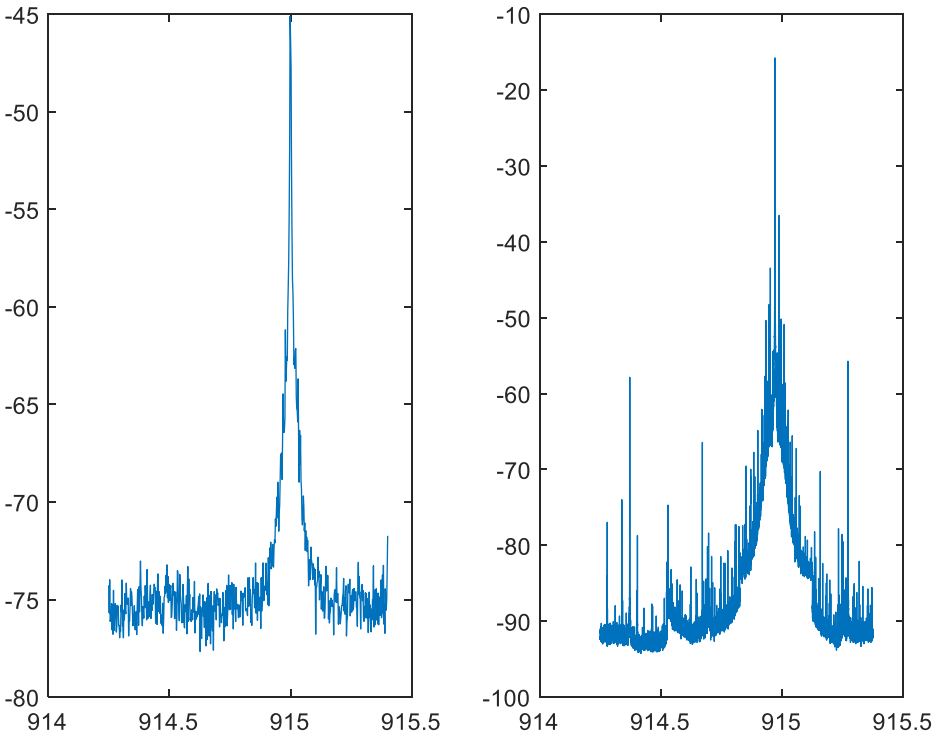
34. ábra - Validációs teszt illusztrációja

A teszt elvégzéséhez szükség volt egy referencia jelre, melynek pontosan ismerjük a centerfrekvenciáját, és jel spektruma a mérések között minimálisan változik. A referencia jel generálásához egy TRF6900A ISM adó-vevő adó oldalát használtam. A mérendő jel 915 MHz-en volt generálva. A generált jel először egy nagypontosságú, NI PXI 5661 RF jelanalizátor segítségével lett megmérve, mérőműszer által mért spektrumot 35. ábra szemlélteti.



35. ábra - Referencia jel 915MHz-en

A validációhoz 2 mérést végeztem az SDR rádió segítségével. Mindkét mérés esetén a start frekvencia 914.25 MHz volt. Az első méréssel 1,2 MHz-es mintavételi frekvenciával megvizsgáltam a spektrumot 915.45 MHz-ig úgy, hogy a center frekvencia 914.85 MHz volt, míg a FFT pontszáma 1024. A második mérésnél 300kHz -es mintavételi frekvenciával és 1024 pontos FFT-vel lemértem ugyanazt a sávot. A választott spektrumot a 36. ábra mutatja, bal oldalon a 1. mérés, míg a jobb oldalon a 2. mérés.

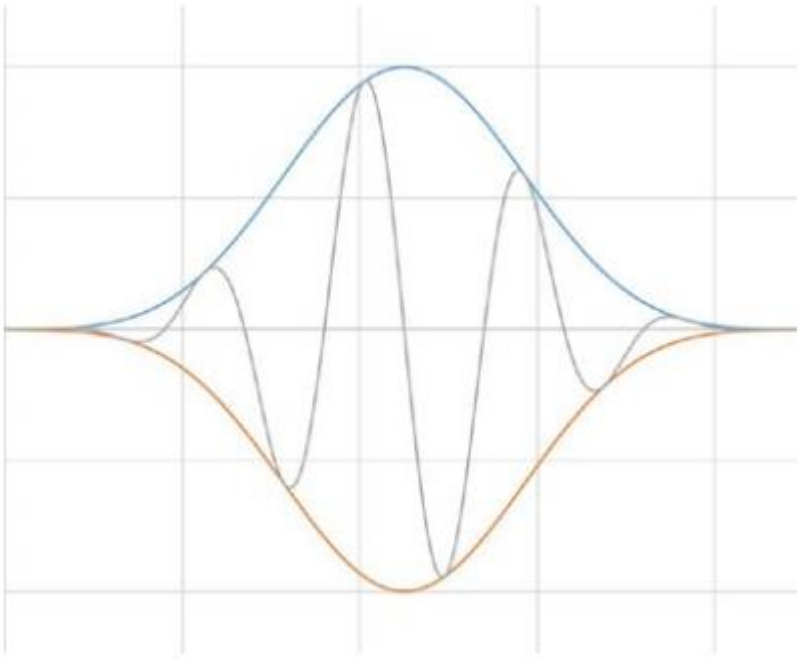


36. ábra - Spektrum 480 MHz-en

A mérések elvégzése után a két spektrumot statisztikai módszerekkel hasonlítottam össze, vettem a két mérési eredményt, és megnéztem a keresztkorrelációjukat.

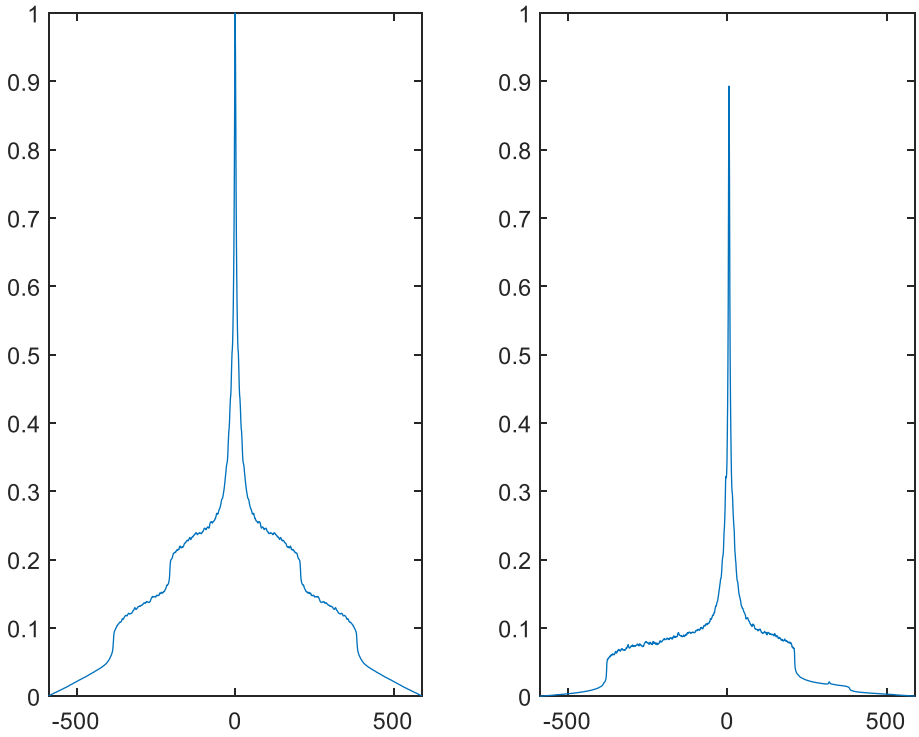
Keresztkorreláció elvégzéséhez célszerű azonos méretű adatsokat összehasonlítani, de a második mérés során az FFT felbontását nem állítottam negyedére, így a felbontása négyszeres lett. Ezt úgy küszöböltem ki, hogy a 2. mérés FFT tömbjére inverz Fourier transzformációt vettem, majd a Fourier transzformációt újra elvégeztem negyed akkora felbontással.

A inverz Fourier transzformáció elvégzése után az adathalmazon Hanning ablakot alkalmaztam azért, hogy az újból elvégzett Fourier transzformáció során ne lépjen fel a spektrum-szivárgás jelensége. A spektrum-szivárgás amiatt lép fel, hogy az időtartománybeli jelsorozatban vannak olyan frekvenciájú tagok, melyek nem jelenek meg egész számú periódussal, vagyis a jelsorozat valamelyik végén szakadással rendelkeznek[17]. Emiatt az FFT elvégzése során nagyfrekvenciás komponensek szivárognak be a spektrumba. Ennek hatását az Hanning ablak alkalmazása úgy csillapítja, hogy az időtartománybeli jelsorozatban megpróbálja a szakadások értékeit csökkenteni a 37. ábra szerinti módszerrel.



37. ábra - Hanning ablak

Két keresztkorrelációt komparáltam össze, az 1. mérés önmagával vett keresztkorrelációját (autokorreláció), illetve a 2. és az 1. mérés keresztkorrelációját. A keresztkorrelációkat normalizálva ábrázoltam, a keresztkorrelációs függvények a 38. ábrán láthatók. A 37. bal oldalán az autokorreláció látható, a függvény szimmetrikus és maximumpontja 1 ha az eltolás értéke 0. A keresztkorrelációs függvényre már nem teljesül a szimmetrikusság, a keresztkorreláció értéke hasonlóan 0 eltolás esetén maximális, normalizált értéke 0.893. Az eredményül kapott keresztkorreláltsági értékből azt a következtést vontam le, hogy az adatfeldolgozó algoritmus jól működik, az eltérések okának a zaj, illetve a 2. mérésnél elvégzet transzformációk torzító hatását tekintettem.



38. ábra - Keresztkorrelációk

5.2.4 Adatfeldolgozás megvalósítása

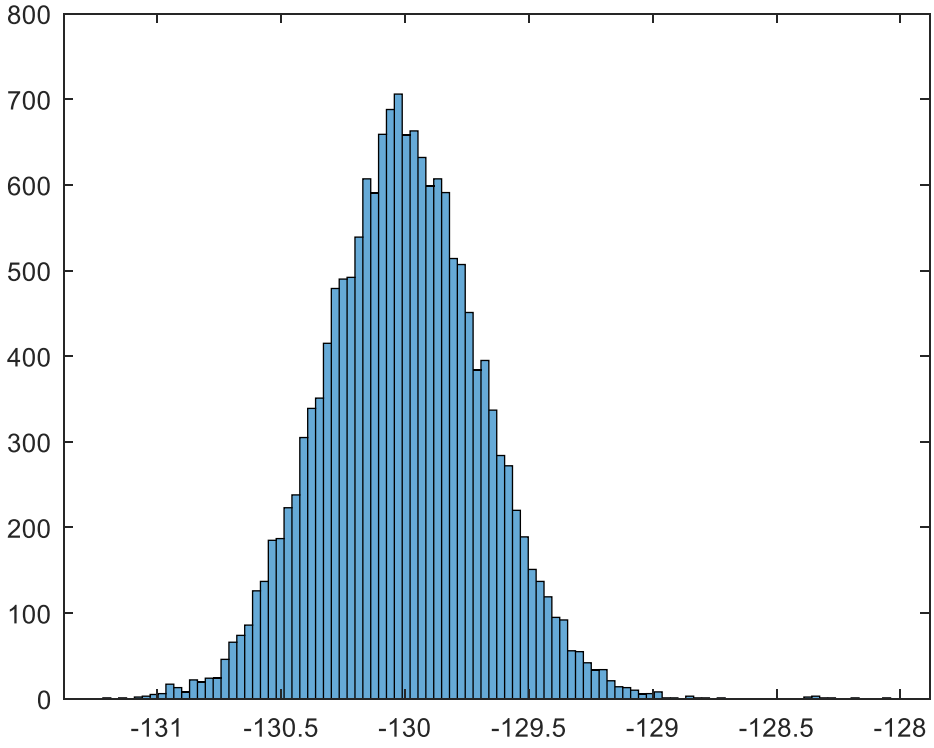
Az adatfeldolgozást végző függvény bemenete az adatgyűjtés során generált Fourier transzformáltat tartalmazó tömb, a hozzá tartozó frekvenciaértékeket tartalmazó tömb, illetve skalár paraméter, mely megadja, hogy mekkora sávszélességű szabad sávot keressen a függvény. Első lépésként a zaj-küszöböt határoztam meg, amely a használt spektrumok sávszélességének meghatározásához szükséges. A zaj-küszöb meghatározáshoz statisztikai elemzést végeztem a mért spektrumon.

A statisztikai elemzéshez a Matlab hisztogram függvényét használtam. A hisztogram függvény a bemeneti adatokat értékük alapján diszjunkt osztályokba sorolja be. Az osztályok két fontos paramétere a gyakoriság, illetve osztályközök értéke. A függvény hívásakor a bemeneti paraméterként meg kellet adni a vizsgálandó halmazt, illetve létrehozandó osztályok számát. Az osztályszámra tipikusan a Sturges-szabályt [13] alkalmazzák, mely szerint:

$$k = 1 + log_2(n) \qquad (24)$$

vagyis a létrehozandó osztályok száma (k) az elemszám (n) logaritmusától függ.

A gyakorlatban ez az érték a vizsgálat során túl alacsonynak bizonyult, ezért az osztályszámot 100-ra állítottam. A spektrum vizsgálata előtt a jel értékeket dBm-re konvertáltam, tudva, hogy az antenna impedancia 50 Ohm volt. Az első hisztogram, melyet a 39. ábrán lehet látni a korábban ismertetett 901-905MHz-es sávot ábrázolja. Az ábrán a vízszintes tengelyen a teljesítmény értékek növekvő sorrendben láthatók, míg a függőleges tengelyen az egyes osztályok elemszáma van ábrázolva. Látható, hogy a hisztogram értékek közelítőleg haranggörbe értéket vesznek fel.

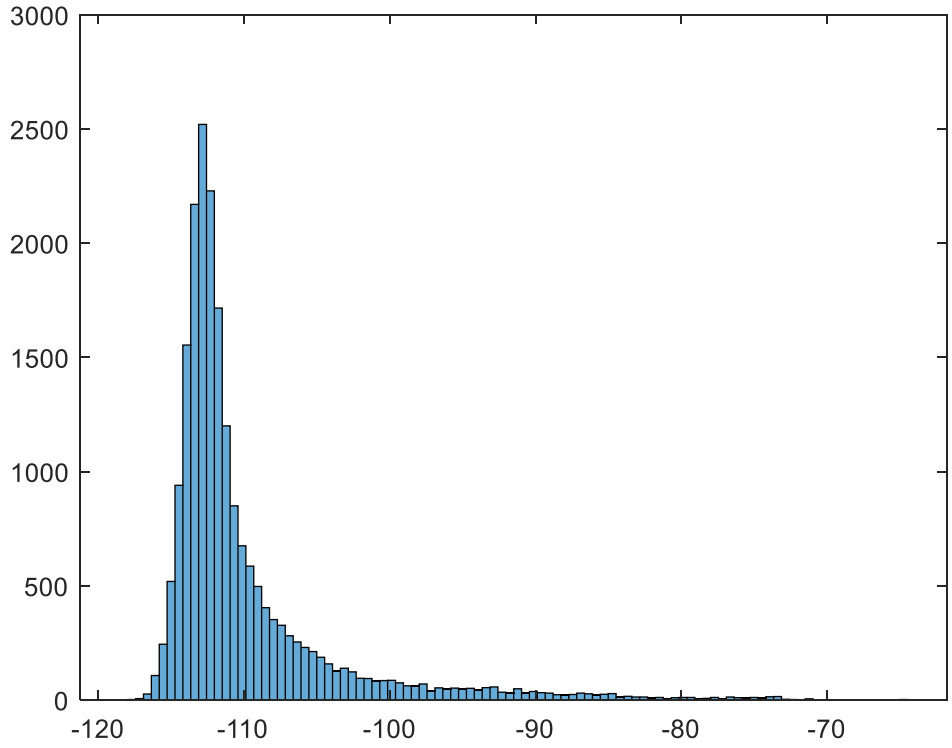


39. ábra - Zaj hisztogramja

Ellenőrzésképp egy tesztet futtatam mely a Lilliefors teszt szerint ellenőrzi, hogy az adott mintahalmaz normál eloszlású-e. A teszt azt a nullhipotézisből indul ki, hogy a mért adat normál eloszlású. A mérést többször is elvégeztem, P = 0.05 szignifikanciaszint mellet a függvény mindig 0 értékkel tért vissza, vagyis a nullhipotézis nem lett megcáfolva. A hisztogram vizsgálattal megállapítottam, hogy zajos spektrum esetén a spektrumon felvett étékek egy normáleloszlású valószínűségi változóval leírhatók. Az implementációban

A hisztogram vizsgálatot elvégzetem olyan frekvencia sávon az FM rádiók által használt 100-108MHz-es spektrumon is, az eredményeket a 40. ábrán lehet látni. A hisztogramról itt már nem mondható el, hogy normál eloszlású, melyet a Lilliefors teszt is igazol. A

hisztogram abban hasonlít a zaj hisztogramhoz, hogy az osztályok elemszáma egy maximumpontig nő, majd a maximumpont után elkezd csökkenni. További hasonlóság, hogy a maximumpontig a hisztogram által felvett görbe hasonlít a normál eloszláshoz. Az elemzést azzal folytattam, hogy külön megvizsgáltam a maximumpont előtti elemek, illetve egész halmaz valószínűségi eloszlását. A teljes halmazra nézve a khi négyzet tesztet végeztem el, a MATLAB beépített khi négyzet tesztelő függvénye, a Chi-square goodness-of-fit test igazolta, hogy az eloszlás khi négyzetes, vagyis k darab független normális eloszlás valószínűségi változóinak a négyzete.



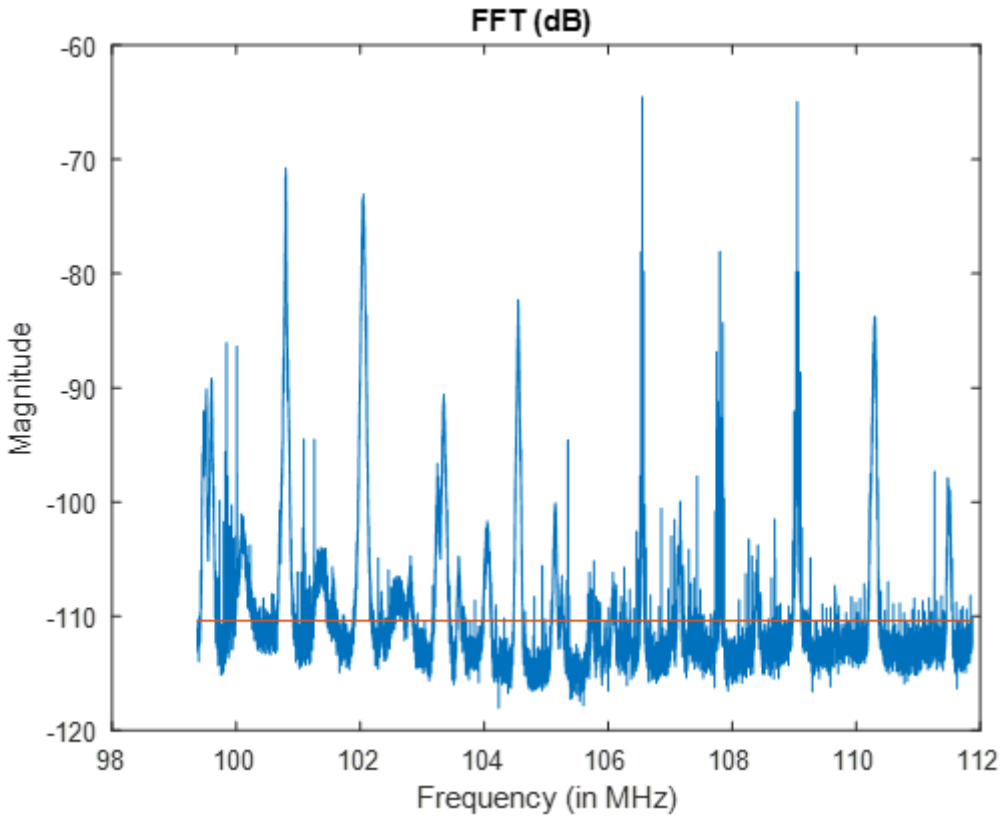
40. ábra - FM hisztogram

Ha megvizsgáljuk a hisztogram maximumpont előtti értékeit, akkor a felvett értékek a maximumpont előtt normális eloszlás szerint fordulnak elő. Erről a viselkedésről egy újabb normálteszt futtatásával győződtem meg. Ehhez vettem a maximumpont és maximumpont előtti osztályok tagjait, majd hozzáadtam az összes elemhez a maximumpont értékét. A művelet után az végig mentem az összes tagon úgy, hogy a páros indexűeket negatív előjellel láttam el. Az ismételt Lilliefors teszt a konvertálás után igazolta, hogy az eloszlás normál eloszlású.

A fentiek ismertében azt a küszöb szintet, melyet felhasználva az egyes használt sávok sávszélességet határoztam meg, a zaj-küszöb és a normál eloszlású zaj szórásának összegeként határoztam meg a (25) egyenlet szerint.

$$k_{küszöb} = k_{zaj-küszöb} + 2\sigma \qquad (25)$$

A gyakorlatban a küszöb pontos értékének meghatározását kísérletezéssel határoztam meg: megnéztem, hogy a szórás hányszorosát érdemes hozzáadni a legpontosabb eredmény meghatározáshoz. Az implementáció során 2 szigma értéket választottam, ilyenkor a zaj-küszöb és a küszöb közötti eltérés 3-4 decibel. A küszöbszint illusztrációja a 41. ábrán található.



41. ábra – Küszöbszint illusztrációja

A küszöb meghatározását követően a küszöbszintből kiemelkedő tartományokat összegyűjtöttem egy struktúra alapú tömbbe. Az allokált adatokat tartalmazó struktúra 4 információt tárolt el az adott spektrumrészletről:

- Centerfrekvencia

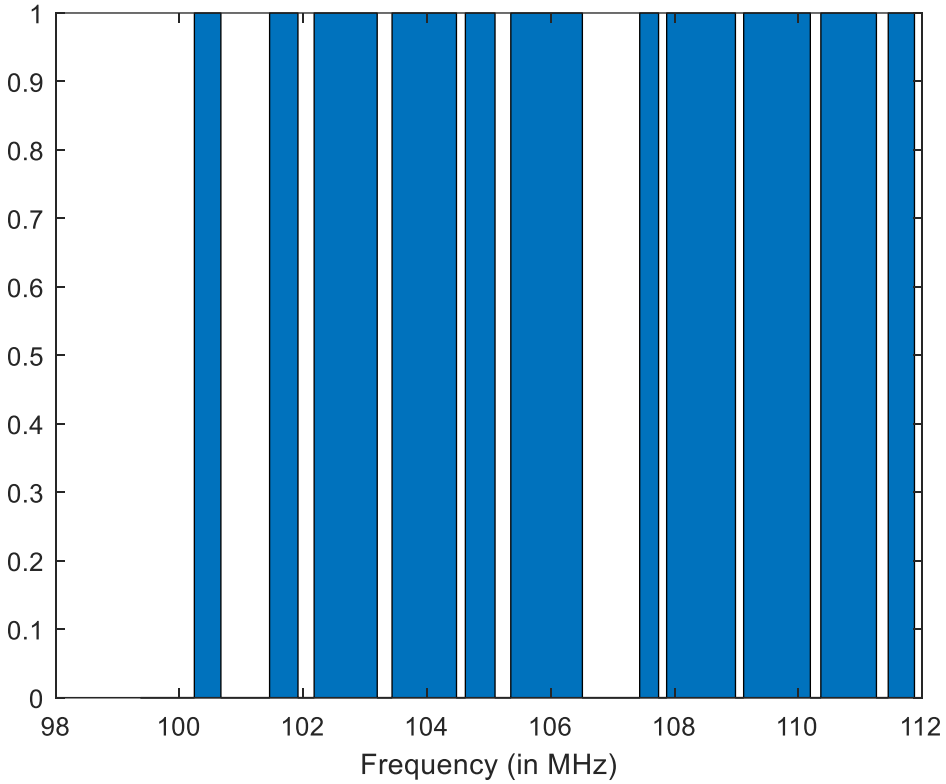
- Alsó határfrekvencia
- Felső határhatárfrekvencia
- Sávszélesség

A használt spektrumok megkeresése mindig a spektrum maximumpontjával kezdődött. A maximumponttól kezdve mind felfele, mind lefele elkezdtem törölni a spektrális információkat egészen addig, míg először elértem a korábban meghatározott küszöböt. Amint elértem mind az alsó, mind a felső határfrekvenciát, a szükséges adatokat eltároltam egy előre definiált tömbben.

Az algoritmus a fentebb ismertette metódus szerint ciklikusan keresi azokat a spektrum részeket, melyek az amplitúdója magasabb a zajnál. A ciklus kilépési feltétele az volt, hogy a maximumpont értéke a küszöb felett legalább 10dB-lell kell lennie.

A szabad sávok meghatározása az allokált sávok tömbjének felhasználásával történt. Az algoritmus ciklikusan végigment az allokált tömbökön, és megvizsgálta, hogy az allokált frekvenciasávok között van-e elegendő hely szabad sávnak. Az első és utolsó allokált sáv között meg kellet azt is vizsgálni, hogy a start és stop frekvencia között van-e elég szabad hely. A szabad sávokat egy skalár tömbben tároltam el, bináris formátumban. A tömb mérete megegyezett az FFT tömb hosszával, hiszen a szabad spektruminformációt az FFT tömb elemeihez rendelem hozzá.

A keresés végén a függvény visszatérési érékként visszaadta a meghatározott küszöbszintet, követelményeknek megfelelő szabad sávok tömbjét. A 41. ábrán látható méréshez tartozó szabad frekvenciasávok halmazát a 42. ábrán szemléltettem. Az ábrán a kék sávok szemléltetik a szabad sávokat. A mérés során a szükséges szabad sávszélességet 300kHz-re állítottam.

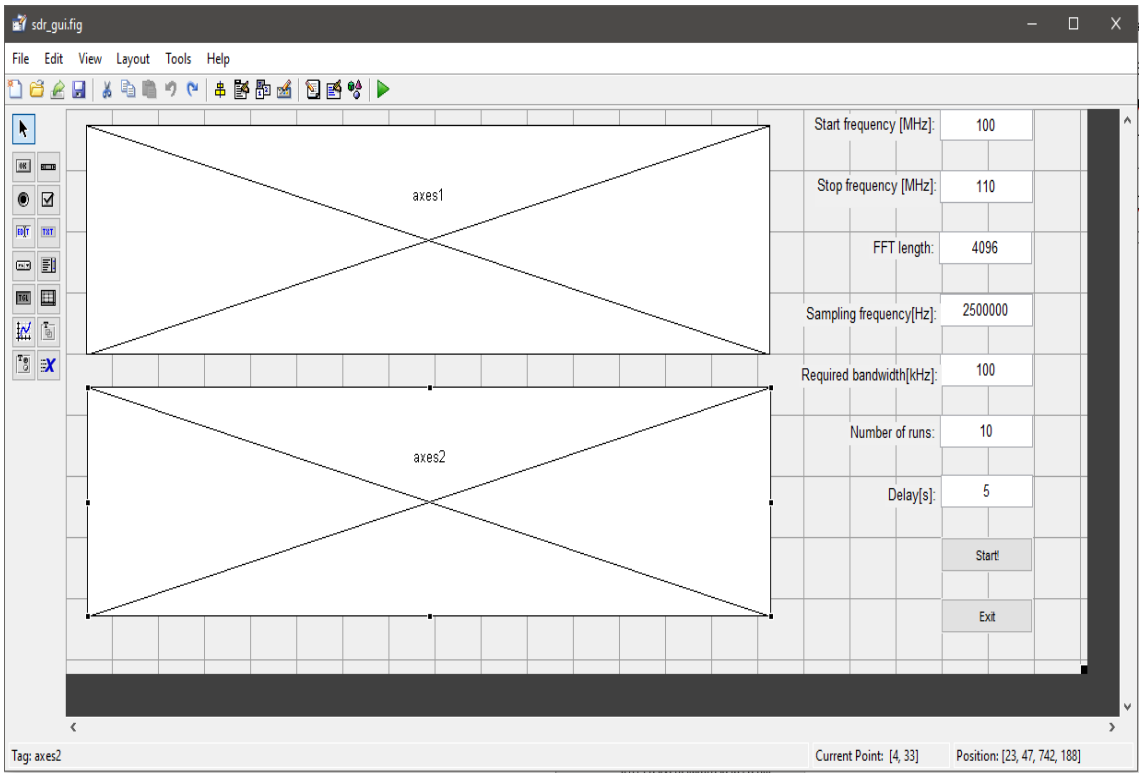


42. ábra - Szabad frekvenciasávok

5.2.5 Grafikus interfész megvalósítása

Az adatfeldolgozó és adatgyűjtő függvények vezérélét, illetve a bemeneti adatok megadását egy grafikus interfésszel ellátott vezérlőblokk végezte. A vezérlés feladata az adatgyűjtéshez és feldolgozáshoz szükséges függvényparaméterek átadása, az ismételt futtatás ütemezése, illetve a mérési adatok kiértékelése volt.

A grafikus interfész létrehozásához a MATLAB-ba beágyazott GUIDE-ot használatam. A GUIDE (Graphical User Interface Developement Enviorement) segítségével eseményvezért grafikus interfészt lehet létrehozni korábban létrehozott szkriptjeinknek. A GUI tervezése két részre bontható. A tervezés során létrejön egy fig kiterjesztésű fájl. Ez a fájl bináris, szerkeszteni nem lehet, csak a GUIDE elindításával MABLAB-on belül. A GUIDE -ben megnyitott .fig fájl segítségével a programunk GUI-jának kinézetét lehet megtervezni: például létre lehet hozni gombokat, menüsorokat, listákat, megjelenítő felületeket, be lehet állítani az ablakméretet a 43. ábrán látható módon.



43. ábra - GUIDE kezelőfelület

A tervezés során létrejön egy .m kiterjesztésű fájl is[15]. Ebben a fájlban a .fig fájlban létrehozott objektumokhoz lehet megtervezni az esemény vezérelt viselkedést, vagyis például mi történik, ha megnyomjuk az egyik gombot, vagy lenyomjuk valamelyik billentyűt.

A spektrumanalizátorhoz a 43. ábrán látható GUI-t terveztem meg. A kezelőfelület bal oldalán az alábbi bemeneti paramétereket lehet megadni:

- Start frekvencia (MHz)
- Stop frekvencia (MHz)
- FFT felbontása
- Mintavételi frekvencia(Hz)
- A keresni kívánt szabad sávszélesség (kHz)
- Mérések száma
- Mérések közti várakozási idő (s)

A bal alsó sarokban két nyomógombot definiáltam, a START gomb segítségével el lehet indítani a mérést, míg a STOP gomb segítségével le lehet állítani a mérést és törölni a memóriában tárolt átmeneti adatokat.

A jobb oldalon két grafikon megjelenítő felület, más néven axis látható. Céljuk a 41. és a 42. ábra tartalmának megjelenítése, minden mérés végén az aktuális mérési eredményekkel.

A GUI függvény futtatásával első lépésként maga a grafikus interfész inicializálódik, kezeli a bemenetei paramétereket, majd elkezd a definiált eseményekre várni. Két eseményt definiáltam, az egyik a START gomb megnyomása, a másik pedig a STOP gomb megnyomása volt.

Az események kezeléséhez létre kellet honi egy-egy callback funkciót. A callback funkcióknak 3 bemenete volt[16]:

- hObject: a gomb kezelője(handle)
- eventdata: esemény információk
- handles: eseményeket és adatokat tartalmazó struktúra

A függvényben először meg kellet hívni a `handles = guidata(hObject);` parancsot, majd a függvény végén a `guidata(hObject, handles);` parancsot. Az első parancs a handlesbe átmásolja hObject aktuális állapotát, változóit. A handles segítségével módosítani lehetet a hObject belső adatait, majd a második paranccsal visszamásoltam a hObjectbe a módosított adatokat. A fenti két paranccsal hatékonyan lehet adatot elmenteni úgy, hogy ne vesszenek el az információk a függvény lefutása után.
A jelfeldolgozó algoritmusomat a fent említett két parancs közé ágyaztam be. A gomb megnyomása után első lépésként a bemeneti paramétereket olvastam ki a paraméterekhez tartozó editbox-okból. Az alábbi kódrészlet a kezdő frekvenciát olvassa ki a megfelelő editbox-ból:

```
stf     = str2double(get(handles.start_freq,'string'))*1e6;
```

A start_freq editbox tartalmára a handles struktúra elméjeként hivatkozok, az editbox-ból csak sztringként tudom kinyerni a szükséges információt, amelyet végül double értékké konvertálok.

Az adatok kinyerése után egy ciklust indítok, amely a futásszámig számol el. A ciklusban először meghívom az adatgyűjtő és feldolgozó paramétereket a szükséges bemeneti

argumentumokkal, majd a függvények visszatérési értékét felhasználva kirajzolom a mérési eredményeket. A ciklus utolsó művelete a kihasználtsági adatok frissítése, és az eredmények kiírása egy fájlba. A kihasználtsági adatok tárolásához egy 2 x N-es mátrixot használok, ahol N-et a FFT felbontása a kezdő és végfrekvencia határoz meg. A felső sorban a frekvenciaadatok az alsó sorban pedig a kihasználtsági adatok találhatóak. A kihasználtsági adatok frissítéséhez mindig az iterációs változó aktuális értékét használtam.

Összegzés és továbbfejlesztési lehetőségek

A munkám során megismerkedtem a komplex burkolók matematikai hátterével, és a különböző SDR architektúrákkal. Ezen ismeretek birtokában egy RTL2832U chip alapú vevőt és MATLAB-ot felhasználva megterveztem és implementáltam egy intelligens spektrumanalizátort. Az alkalmazással méréseket végeztem, és ezen mérési eredmények segítségével validáltam a spektrumanalizátor működését.

Az alkalmazás képes egy adott frekvenciasávot megmérni, és kihasználtságáról információkat gyűjteni. A szoftver küszöbszint meghatározását végző algoritmus tovább optimalizálása megfontolandó lehet. A spektrumanalizátor funkcionalitását növelni lehetne a grafikus interfész funkcionalitásának növelésével, továbbá a mért jelek teljesítményének pontos meghatározásával, viszont ehhez további kalibrációs mérések szükségesek.

További fejlesztési lehetőség lehet, hogy a spektrumanalizátor nem csak a spektrum kihasználtságát határozza meg, hanem spektrumalakból és annak időbeli változásából próbál következtetni az adott jel modulációjára. Másik lehetőség, hogy egy adatbázist felhasználva próbálja felismerni, hogy az adott sávon milyen rádiószolgáltatás található.

Irodalomjegyzék

[1] Dr. Huba Antal, Dr. Lipovszki György: Méréselmélet

http://www.mogi.bme.hu/TAMOP/mereselmelet/

[2] Simon Haykin: Communication Systems - Fourth Edition

[3] Dr. Kolumbán Géza, Krébesz Tamás István, Francis C.M. Lau: Theory and Application of Software Defined Electronics

https://users.itk.ppke.hu/~kolumban/own_papers/software12defined_electronicspdf

[4] Charan Langton: Hilbert Transform, Analytic Signal and the Complex Envelope

http://complextoreal.com/wp-content/uploads/2013/01/tcomplex.pdf

[5] Robert W. Steward, Kenneth W Barlee, Dale S. W. Atkinson Louise H. Crockett: Software Defined Radio using Matlab & Simulink and the RTL-SDR

https://www.researchgate.net/publication/287760034_Software_Defined_Radio_using_MATLAB_Simulink_and_the_RTL-SDR

[6] Realtek RTL2832U, The mystery chip at the heart of RTL-SDR. VERSION 1

[7] Rafael Microelectronics: R820T High Performance Low Power Advanced Digital TV Silicon Tuner Datasheet

https://rtl-sdr.com/wp-content/uploads/2013/04/R820T_datasheet-Non_R-20111130_unlocked.pdf

[8] NI: Sampling Quality - General Analog Concepts

http://www.ni.com/white-paper/5356/en/

[9] Dr. Bognár György: Rádiófrekvenciás integrált áramkörök tervezése és alkalmazása

[10] Richard G. Lyons, Understanding Digital Signal Processing (2nd Edition)

[11] ITU-R SM.2256-1, Spectrum occupancy measurements and evaluation

https://www.itu.int/dms_pub/itu-r/opb/rep/R-REP-SM.2256-1-2016-PDF-E.pdf

[12] Calibration of the SDR frequency using GSM signals,

https://inst.eecs.berkeley.edu/~ee123/sp14/lab/lab2/lab2/Time_Frequency_Part_II_GSM.html

[13] Herbert A. Sturges: The choice of a class interval.

[14] NI: The NI Vector Signal Transceiver Hardware Architecture,

http://www.ni.com/product-documentation/14028/en/

[15] Create a Simple App Using GUIDE, Matlab documentation

https://www.mathworks.com/help/matlab/creating_guis/

[16] Handle Object Behavior, Matlab documentation

https://www.mathworks.com/help/matlab/matlab_oop/handle-objects.html

[17] NI: Understanding FFTs and Windowing

http://download.ni.com/evaluation/pxi/Understanding%20FFTs%20and%20Windowing.pdf