

Univerza v Ljubljani

Fakulteta za elektrotehniko

Aljaž Blatnik

**Vrednotenje faznega šuma v ulomkovi  
fazno sklenjeni zanki**

Magistrsko delo

Mentor: prof. dr. Matjaž Vidmar

Ljubljana, 2017



*Ko hodiš, pojdi zmeraj do konca.*

*Spomladi do rožne cvetice,  
poleti do zrele pšenice,  
jeseni do polne police,  
pozimi do snežne kraljice,  
v knjigi do zadnje vrstice,  
v življenju do prave resnice,  
v sebi do rdečice čez eno in drugo lice.*

*A če ne prideš ne prvič ne drugič  
do krova in pravega kova  
poskusi:  
vnovič  
in zopet  
in znova.*

*Tone Pavček*



# Vsebina

<b>1 Problem</b>	<b>15</b>
<b>2 Fazno sklenjena zanka</b>	<b>17</b>
2.1 Delovanje PLL zanke .....	17
2.1.1 Celoštevilski način .....	19
2.1.2 Ulomkovni način .....	20
2.2 Implementacija PLL .....	23
2.2.1 Fazni detektor .....	23
2.2.2 Delilnik frekvence .....	24
2.2.3 Zančno sito .....	26
2.3 Zaključek .....	31
<b>3 Fazni šum</b>	<b>33</b>
3.1 Šum oscilatorja .....	33
3.2 Definicija normiranega faznega šuma .....	35
3.3 Leesonova enačba .....	36
3.3 Fazni šum PLL .....	37
3.3.1 Fazni šum ulomkovnega PLL .....	37
3.4 Meritev faznega šuma .....	39
<b>4 Merjenec</b>	<b>41</b>
4.1 Izbira gradnikov za gradnjo PLL zanke .....	41
4.2 MAX2871 .....	42
4.3 Načrtovanje prototipa .....	45
4.3.1 Zančno sito .....	45
4.3.2 Visokofrekvenčni izhod .....	45

4.3.3 Širokopasovni balun.....	46
4.3.4 Rezultati meritev TCM1-83X+ .....	47
4.3.5 Referenčni vhod .....	48
4.3.6 Napajanje .....	49
4.4 Gradnja prototipa.....	50
4.4.1 Prvi prototip .....	50
4.4.2 Drugi prototip.....	51
<b>5 Podporni elementi merilnega sistema</b>	<b>55</b>
5.1 Komunikacija s PLL čipom.....	55
5.2 Samodejna meritev faznega šuma .....	57
5.3 Postavitev merilnega sistema .....	58
<b>6 Rezultati meritev</b>	<b>61</b>
6.1 Lastni fazni šum spektralnega analizatorja.....	61
6.2 Fazni šum na območju 3 - 6 GHz .....	62
6.3 Vpliv delilnikov na fazni šum .....	63
6.4 Vpliv baluna na fazni šum .....	64
6.5 Vpliv toka črpalke naboja na fazni šum .....	64
6.6 Vpliv linearnosti črpalke naboja na fazni šum .....	65
6.7 Primerjava različnih režimov delovanja faznega detektorja na fazni šum.....	66
6.8 Vpliv deljenje frekvence pred N števcem na fazni šum .....	66
6.9 Fazni šum na celotnem območju od 3 do 6 GHz.....	67
6.10 Komentar meritev .....	70
<b>7 Gradnja lastne PLL zanke s FPGA</b>	<b>71</b>
7.1 Altera EP2C5T144C8 Cyclone II FPGA .....	73
7.1.1 Programska koda.....	74
7.2 Črpalka naboja.....	76
7.3 Gradnja celotnega sistema zanke s FPGA.....	77
<b>8 Zaključek</b>	<b>79</b>

---

<b>Literatura</b>	<b>81</b>
<b>A Shema vezja s PLL zanko</b>	<b>84</b>
<b>B Izvorna koda samodejne meritve faznega šuma</b>	<b>85</b>
<b>C Izvorna koda FPGA</b>	<b>91</b>





## Seznam uporabljenih simbolov

<b>OZNAKA</b>	<b>OPIS</b>
<i>A</i>	Amper, enota toka
<i>ADC</i>	Analogno digitalni pretvornik
<i>AM</i>	Amplitudna modulacija
<i>B</i>	Pasovna širina
<i>DAC</i>	Digitalno analogni pretvornik
<i>dB</i>	Decibel
<i>FET</i>	Poljski tranzistor
<i>FM</i>	Frekvenčna modulacija
<i>FPGA</i>	Programabilni čip (ang. Field-programmable gate array)
<i>FR4</i>	Tip laminata tiskanine
<i>Hz</i>	Herc, enota frekvence
<i>PLL</i>	Fazno sklenjena zanka
<i>PM</i>	Fazna modulacija
<i>SMD</i>	Ohišje za površinsko pritrditev
<i>TCXO</i>	Temperaturno kompenziran kristalni oscilator
<i>V</i>	Volt, enota napetosti
<i>VCO</i>	Napetostno krmiljen oscilator
<i>XOR</i>	Izključujoči ALI
$\pi$	Pi



## Povzetek

Delo se ukvarja z vrednotenjem faznega šuma ulomkove fazno sklenjene zanke. Z opisom teoretičnega ozadja delovanja PLL zanke tako v celoštevilskem kot ulomkovem načinu preide v opis posameznih gradnikov skupaj s tehnološkimi omejitvami in primeri izvedbe. Fazni šum je opisan iz primera vrste šuma, tako matematično kot vizualno, v primeru vkljenjene zanke.

Praktični del predstavlja gradnjo frekvenčnega izvora s pomočjo integriranega čipa MAX2871, dodana so tudi navodila, kako prototip sestavimo v domači delavnici s pomočjo široko dostopnih orodji. Predstavljeni so podporni sistemi za komunikacijo s PLL čipom preko USB povezave, ter metoda za samodejno merjenje faznega šuma s pomočjo programskega jezika Python, ter spektralnega analizatorja.

Meritve prikazujejo vpliv različnih načinov delovanja na spreminjanje faznega šuma. Izmerjen je vpliv toka črpalke naboja, stopnja linearnosti, ter različni režimi za izločanje neželenih špičk. Predstavljena je tudi samodejna meritev faznega šuma v celotnem frekvenčnem področju.

V zaključnem delu je predstavljena izvedba fazno sklenjene zanke s FPGA, ki je bila sicer uspešna, a s stališča faznega šuma zaradi presluha v notranjosti čipa slaba.

**Ključne besede:** fazni šum, fazno sklenjena zanka, ulomkovni način delovanja, programabilni čip, MAX2871, Python, ARM Cortex-M0



## **Abstract**

The work deals with the evaluation of the phase noise in the fractional phase locked loop. The theoretical background of the PLL loop in both integer and fractional mode is presented. Description of each component along with the technical limitations and examples is added. Phase noise is described both mathematically and visually in the case of the locked loop system.

The practical part presents the construction of the frequency source by means of the integrated chip MAX2871. Instructions on how to assemble a prototype in the home workshop by using widely available tools are also added. Support systems to communicate with PLL chip via a USB connection, and a method for automatically measuring the phase noise by using the programming language Python and the spectrum analyzer are described at the end of the chapter.

Phase noise measurements of MAX2871 were performed at different modes in action, changes in the charge pump current, degree of linearity and modes for reducing unwanted spurs. Automatic method of complete spectrum measurement is also presented.

In the final part, implementation of phase locked loop with FPGA is described, which although has been successful, was from the standpoint of phase noise bad due to crosstalk inside the chip.

**Key words:** phase noise, phase locked loop, fractional mode, field programmable gate array, MAX2871, Python, ARM Cortex-M0



## 1 Problem

Šumijo gozdovi domači, a kako šumi oscilator v fazno sklenjeni zanki? Tega iz znanstvenih člankov ne bomo uspeli razbrati. Če vseeno uspemo nekako poiskati vire ki se med seboj vsaj ne izključujejo, naletimo na kup višje matematike ki na koncu skoraj vedno vodi v veličastno simulacijo. Ta pa je za praktičnega inženirja popolnoma neuporabna.

S šumom seveda mislim fazni šum signala na izhodu, ki je posledica večih prispevkov v celotnem sistemu fazno sklenjene zanke (ang. Phase Lock Loop ali kratko PLL). Nekateri prispevki so jasno določeni in izračunljivi, ter potrjeni s preprostimi meritvami. Druge nastane težava. Kakšen je na primer prispevek k faznemu šumu če črpalka nabojev za zančni filter ni simetrična, kako točno njen tok vpliva na fazni šum, ali lahko frekvenco oscilatorja delimo preden jo vodimo v N delilnik, brez poslabšanja razmer? Še huje je, ko se povprašamo po tem kaj se skriva v komercialno dostopnih PLL čipih, ki so zadnja leta dosegli zavirljivo nizke cene. Kaj točno je razlika med načinom za nizek fazni šum, ter načinom za nizek nivo špičk, ki jih povzroči ulomkovni način PLL? Kako se to odraža v spektru? Kakšna matematika se skriva za ulomkovim načinom delovanja, kako hitra je, kakšnega reda, kako to vpliva na fazni šum? Vprašanj je hitro več kot odgovorov. Je gradnja frekvenčnega izvora z oscilatorjem v fazno sklenjeni zanki res postala le še uporaba priporočene izbire in postavitve komponent neposredno iz podatkovnega lista, brez dvoma o sposobnost inženirjev proizvajalca, ki od sebe dajo le okrnjene in deloma olepšane podatke?

Na ta vprašanja želi odgovoriti pričujoče delo, ter trditve podkrepiti z meritvami v realnem svetu. Še več! Vse uporabljene komponente so enostavno dobavljive, cenovno dostopne, tiskanino pa lahko izdelamo kar v domači delavnici. Morda pa komu ugotovljeno še koristi.

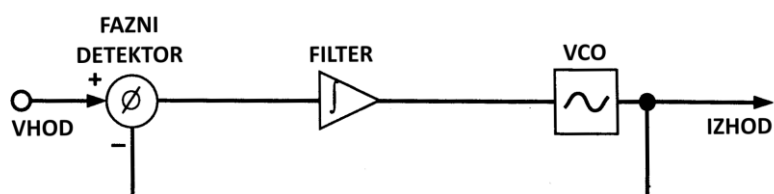




## 2 Fazno sklenjena zanka

Koncept fazno sklenjene zanke (ang. PLL) je že precej star [1] in se intenzivno uporablja predvsem od časov začetka satelitskih komunikacij dalje [2]. Ena izmed prednosti PLL arhitekture je zmožnost doseganja nizkega faznega šuma v širokem frekvenčnem območju in hkrati sposobnost držati frekvenco stabilno, z uporabo ustrezne reference, tudi temperaturno in časovno. Poleg tega je zaradi enostavnosti izvedbe PLL sistema na monolitnem čipu odlična osnova za uporabo v telekomunikacijskih sistemih.

### 2.1 Delovanje PLL zanke



Slika 2.1: Osnovna PLL arhitektura

Osnovno strukturo PLL zanke prikazuje slika 2.1. Med običajnim delovanjem se zanka vklene na vhodni signal, njen izhod pa je konstantna vrednost primerjalnika faze. Zanko sestavlja fazni detektor (fazni primerjalnik), frekvenčno sito (z ojačevalnikom) ter napetostno krmiljen oscilator (VCO). Fazni detektor skupaj z znančnim sitom na izhodu daje enosmerno napetost, ki je proporcionalna fazni razliki izhodne ter vhodne frekvence. Linearizirano ojačenje faznega detektorja označimo s  $K_d$ , vrednost pa običajno zapišemo z volt/radian.

VCO na izhodu proizvaja frekvenco, ki je odvisna od enosmerne napetosti na njegovem vhodu in je načrtovan tako, da je njegovo frekvenčno območje veliko vsaj

toliko, kot to zahteva končni sistem. Faktor spremembe izhodne frekvence v primerjavi z vhodno je določen kot ojačenje VCO, označimo pa ga z  $K_0$ .

Zančno sito je navadno nizko prepustno sito čigar ojačenje enosmerne komponente označimo z  $K_f$ , njegova prenosna funkcija pa je podana z enačbo (2.1):

$$F(s) = \frac{K_f(1+s\tau_z)}{(1+s\tau_p)} \quad (2.1)$$

kjer  $s$  označuje kompleksno spremenljivko Laplacove transformacije,  $\tau_z$  ničlo danega sita,  $\tau_p$  pa njegov pol. Navadno je izvedeno kot preprosto pasivno sito z uporabo uporov ter kondenzatorjev, njegovo ojačenje je takrat enako nič, v kolikor pa potrebujemo ojačenje signala faznega detektorja zaradi zahtev, ki nam jih narekuje napetostno krmiljen oscilator, uporabimo aktivno sito z uporabo operacijskega ojačevalnika, ter pasivnimi komponentami za doseganje želene prenosne funkcije.

Skupno fazno prenosno funkcijo celotnega sistema tako lahko zapišemo kot (ne pozabimo da je frekvenca odvod faze po času):

$$H_\theta(s) = \frac{\theta_{izhod}(s)}{\theta_{vhod}(s)} = \frac{K_d K_0 F(s)/s}{1 + K_d K_0 F(s)/s} \quad (2.2)$$

ki jo lahko prepišemo kot [3]:

$$H_\theta(s) = \frac{1 + \frac{s}{\omega_n} (2\delta - \frac{\omega_n}{\Omega_k})}{1 + \frac{s}{\omega_n} 2\delta + (\frac{s}{\omega_n})^2} \quad (2.3)$$

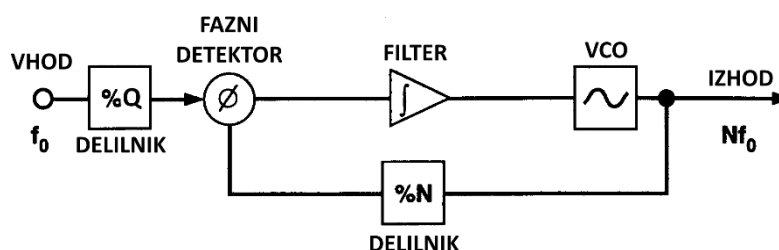
kjer je  $\Omega_k = K_d K_0 K_f$ ,  $\omega_n = \frac{\Omega_k}{\tau_p}$  in  $\delta = \frac{1}{2} (\frac{\omega_n}{\Omega_k} + \omega_n \tau_z)$ .

Vrednost  $\omega_n$  predstavlja pasovno širino zanke, ali naravno frekvenco PLL sistema. Faktor dušenja  $\delta$  določa trajanje prehodnega pojava (nihanje do pravilne vrednosti). Oba faktorja sta pomembna za določanje frekvenčnega in časovnega odziva celotnega sistema.

Kadar je PLL zanka načrtovana, kot je to blokovno predstavljeno na sliki 2.1, izhodna frekvenca preprosto sledi vhodni. Čeprav se sprva zdi, da je tako vezje popolnoma neuporabno, saj na izhodu dobimo signal enake frekvence, ga lahko s pridom izkoriščamo za regeneracijo ure, regeneracijo nosilca ali pa frekvenčno pasovno sito [4], [5].

### 2.1.1 Celoštevilski način

PLL lahko uporabimo tudi kot tehniko za doseganje boljnjih lastnosti oscilatorja, ko fazno sklenemo nestabilen in šumen mikrovalovni oscilator, na nizko šumen stabilen signal, ki pa tiktaka z nižjo frekvenco. Tipičen primer takšnega sistema je uporaba kristalnega oscilatorja kot reference za krmiljenje VCO oscilatorja, ki deluje na znatno višjih frekvencah. V tem primeru v povratni zanki pred primerjalnikom faze potrebujemo delilnik frekvence. Blokovno shemo takšnega sistema prikazuje slika 2.2.



Slika 2.2: PLL zanka z delilnikom frekvence v povratni vezavi

Frekvenca napetostno krmiljenega oscilatorja je zmanjšana za faktor  $N$ . Prenosna funkcija se v tem primeru ne spremeni, z izjemo faktorja ojačenja VCO  $K_0$ , ki je sedaj  $K_0/N$ . Delilnik ima lahko konstantno vrednost, ali pa je nastavljiv, s čimer poljubno nastavljamo frekvenco izhoda.

Slabost takšne zanke je, da lahko izhodno frekvenco spreminjamo samo v diskretnih korakih za večkratnik vhodne frekvence. Kaj kmalu se zgodi, da si želimo manjši frekvenčni korak z uporabo obstoječih elementov, torej brez spreminjanja referenčnega kristala, ali menjave VCO. Kot najpreprostejši ukrep je dodajanje dodatnega delilnika  $Q$  na vrodu referenčnega signala v fazni detektor. S tem učinkovito zmanjšamo vhodno referenčno frekvenco in še bolj razdelimo nabor različnih frekvenc, katere lahko proizvedemo na izhodu VCO. Frekvenca na izhodu bo tako:

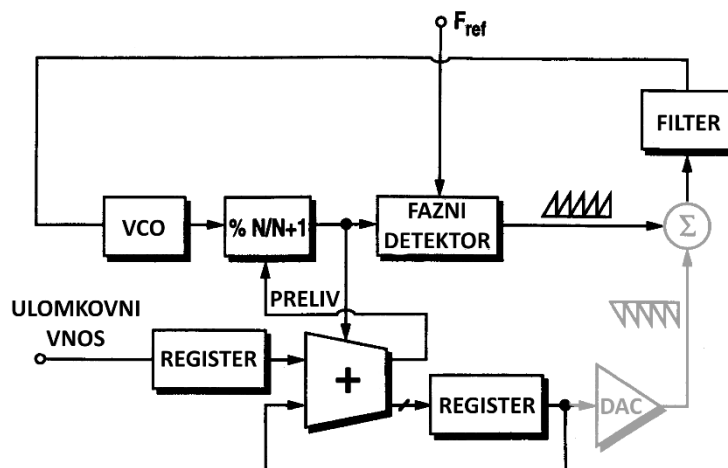
$$f_{izhod} = \frac{Q}{N} f_{vhod} \quad (2.4)$$

Vendar, če uporabljamo visoko vhodno frekvenco, lahko fazni detektor deluje s primerjalno frekvenco v območju od nekaj 10 do preko 100 MHz, kar daje boljše rezultate v smislu faznega šuma, saj višja primerjalna frekvenca pomeni večji vpliv referenčne frekvence, ki je v primeru uporabe kristalnega oscilatorja vnaša manjši fazni šum, kot z uporabo nižjih frekvenc, kjer nam težave povzroča šum nestabilnega VCO. Ravno tako nižja primerjalna frekvenca pomeni daljši čas vklepanja zanke na referenčni signal. Več o tem je zajeto v poglavju 3.3 Fazni šum PLL.

### 2.1.2 Ulomkovni način

Kadar želimo manjši frekvenčni korak, vendar ne želimo zajadrti v omejitve nizkih primerjalnih frekvenc, uporabimo ulomkovni režim delovanja PLL zanke. Ideja tega je, da frekvenco VCO poleg celoštevilskega deljenja delimo še z ulomkom poljubnega števca in imenovalca, ter s tem dosežemo poljubno majhen frekvenčni korak [6], [7].

Najenostavnejši način za doseganje ulomkovnega deljenja je, da frekvenco delimo s faktorjem  $N+1$  vsakih  $M \cdot K$  ciklov (kjer je želeni ulomek  $1/M$ ), ter z  $N$  za preostale  $(1/M) \cdot K$  cikle. Efektivna vrednost deljenja je tako  $N + 1/M$ . Na primer, želimo izhodno frekvenco 835 MHz. Na voljo imamo referenčno frekvenco 100 MHz, vrednost delilnika  $N$  naj bi skakala med 8 in 9, vrednost  $1/M$  je 0.35, kar pomeni da je  $M$  enak 2.86. Vrednost  $N$  je tako 9 vsakih 286 ciklov (če je  $K = 1000$ ), ter 8 preostalih 714 ciklov. Zgodovinsko je bil tak način teoretično implementiran z uporabo fazno zadrževalnega registra, kot to prikazuje slika 2.3.



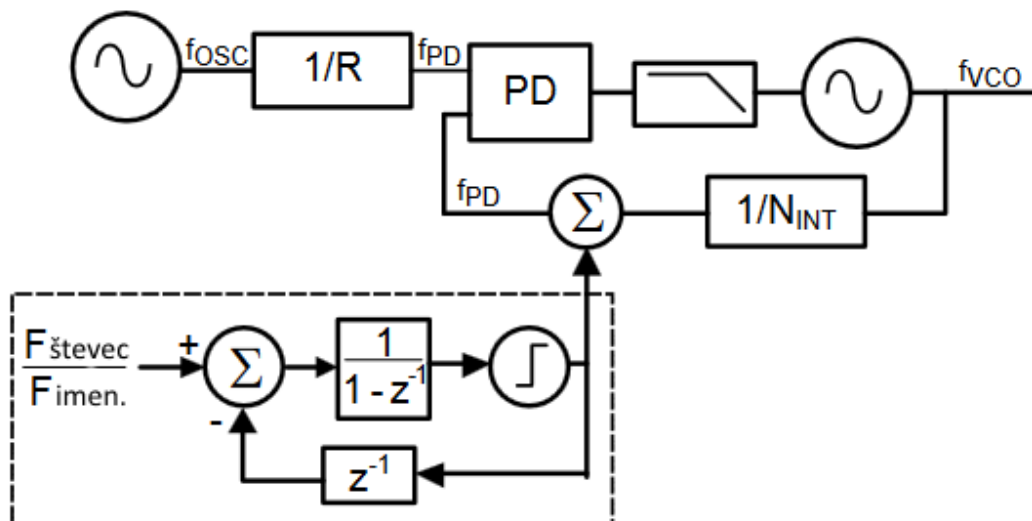
Slika 2.3: Blokovni načrt ulomkovnega PLL

Če uporabimo primer v zgornjem odstavku bi to pomenilo, da je vrednost  $1/M$  (0.35) dodana vrednosti registra, ki povzroči prelitje ko doseže 1. Ob vsakem takem dogodku, bo delilnik  $N$  delil z 9 namesto z 8.

Register drži vrednost napake faze, ki se nabira z delovanjem PLL zanke. To lahko s pridom uporabimo za izboljševanje lastnosti zanke, saj fazni detektor na svojem izhodu proizvede žago, katere vrednost se povečuje z vsako periodo. Te spremembe napetosti modulirajo vhod VCO in povzročajo neželeno modulacijo frekvence na izhodu. Register v zanki drži ravno negativno vrednost žage, s

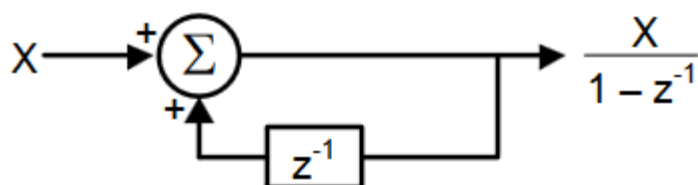
seštevanjem obeh vrednosti, pa lahko vpliv napake na modulacijo vhoda VCO močno omilimo (sivo področje). Seveda moramo pri tem ustrezno razširiti območje delovanja digitalno-analognega pretvornika (DAC) da ustreza zahtevam filtra in uspešno izloča napako, ki prihaja iz faznega detektorja. Težava takšnega načina je hitrost, predvsem DAC pretvornika, zato je rešitev zamrla.

Drugačno predstavitev tradicionalnega načina delovanja prikazuje slika 2.4.



Slika 2.4: Blokovna shema tradicionalnega ulomkovega N modulatorja

Ker se nahajamo v digitalnem časovno-diskretnem sistemu, opis delovanja največkrat predstavlja  $z$  transformacija. Pri tem je predstavlja  $z^{-1}$  časovno zakasnitev enega urinega takta,  $1/(1-z^{-1})$  pa seštevalni register, ki ga je moč predstaviti tudi z blokovno shemo na sliki 2.5.



Slika 2.5: Blokovna shema  $1/(1-z^{-1})$  transformacije

Delovanje je dodajanje prejšnje vrednosti seštevanja izhodu bloka, z drugimi besedami neprestano dodajanje vrednosti spominskemu registru.

N števec (modulatorja s slike 2.4) deluje popolnoma neodvisno v celoštevilskem načinu. Vrednost ulomka mu je dodana z zunanjim blokom (sigma-delta) kot 0 ali 1 pri seštevanju, v natančnem razmerju ulomka. Predhodna vrednost kvantizatorja je

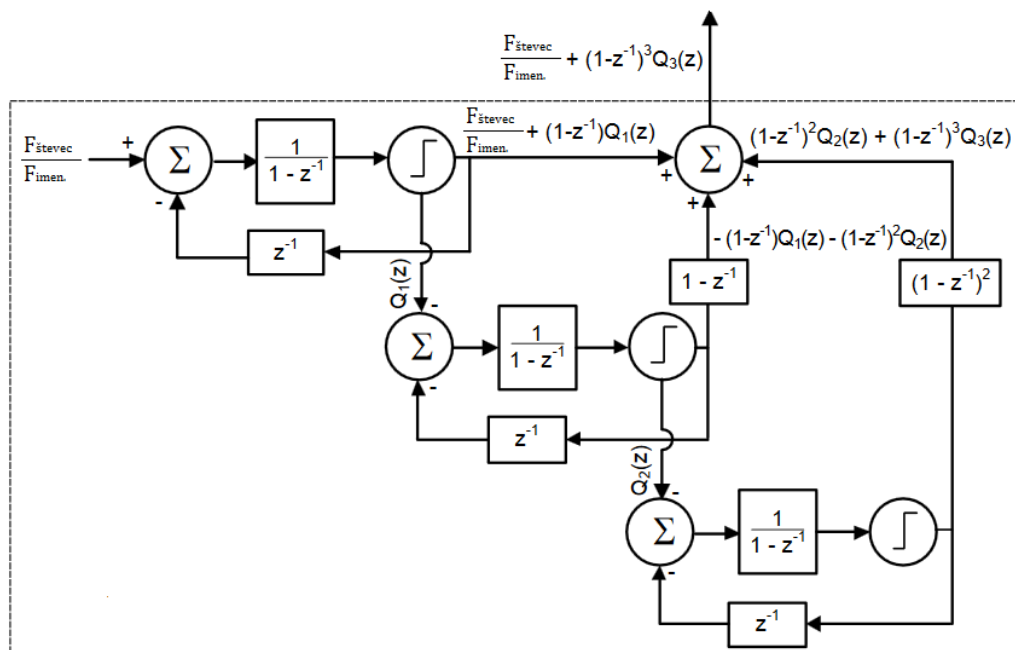
odšteta vhodnemu ulomku, vrednost napake (razlike), pa je shranjena v zadrževalno-seštevalni register (slika 2.5). Kadar je napaka v zadrževalnem registru manj kot ena, je na izhodu kvantizatorja vrednost 0, ko napaka preseže vrednost ena, pa se na izhodu pojavi 1. V naslednjem ciklu je enica odšteta od vrednosti ulomka, cikel pa se zopet ponovi.

Žal ulomkovni način delovanja v spektru izhodnega signala poleg zelene frekvence proizvede še tako imenovane špičke, signal nekoliko nižje in višje frekvence, kar sicer pogosto ne moti delovanja naše končne naprave, a je lahko nadležen pojav v specifičnih primerih, na primer gradnji lokalnega oscilatorja spektralnega analizatorja. Čeprav so špičke matematično natančno določljive ( $\Delta f = \text{frekvenca faznega detektorja}$ ), jih povsem ne moremo odpraviti.

Sigma-delta ( $\Sigma - \Delta$ ) modulator naslavlja ravno ta problem, vrednosti  $N$  in  $N+1$  spreminja na tak način, da se kvantizacijski šum pojavi na višjih frekvencah, kjer ga je lažje izločiti z zančnim sitom. Večina današnjih sodobnih ulomkovnih PLL čipov na tržišču uporablja ravno ta način delovanja [8]. Metoda je vzeta neposredno iz sigma-delta ADC in DAC pretvornikov.

Pri tem uporablja dva načina delovanja. Prvi način omogoča več nivojsko spreminjanje vrednosti  $N$  števca. Če se pri enostopenjskem modulatorju vrednost spreminja le med  $N$  in  $N+1$ , se na primer pri trostopenjskem vrednost števca  $N$  spreminja med osmimi različnimi števili. Strnjeno na kratko, modulatorji višjega reda delujejo bolje kot modulatorji nižjega reda, a je to odvisno od načina uporabe PLL zanke. Drugi način vključuje podrhtavanje sekvence. Držanje razmerja  $N$  in  $N+1$  konstantnega z določenim ulomkom, ni najboljša, saj večja fazni šum. Boljše je, če lahko vrednosti  $N$  in  $N+1$  naključno premešamo tako, da se pojavljata navidezno naključno, a v pravilnem razmerju zelenega ulomka. To rešuje podrhtavanje, ki pa še vedno ne izloča neželenih osnovnih špičk.

Čeprav je tudi struktura prikazana na sliki 2.4 pravzaprav enostopenjski sigma-delta modulator, se je v industriji prijelo pravilo, da se s tem imenom označuje modulatorje vsaj drugega reda, ter brez analogne kompenzacije. Obstaja več načinov kako izdelati tak modulator višjega reda, a pogost način je implementacija strukture MASH (ang. Multi-stage noise shaping). Pri tej arhitekturi je izhod vsake stopnje vhod naslednje, vrednosti napak vseh stopenj pa so seštete skupaj. Blokovno shemo trostopenjskega sigma-delta MASH modulatorja prikazuje slika 2.6.



Slika 2.6: Blokovna shema sigma-delta MASH modulatorja tretjega reda [8]

## 2.2 Implementacija PLL

Z današnje tehnologijo je mogoče skoraj vse sestavne dele PLL implementirati na enem samem monolitnem čipu. Navadno proizvajalci načrtovalcu pustijo odprte roke le pri izbiri filtra v povratni vezavi, pogosto je v sam čip vgrajen tudi napetostno krmiljen oscilator (ali več njih). Takšen način močno poenostavi tiskanino, kamor čip pritrdimo, saj večino visokofrekvenčnih povezav skriva v samo notranjost čipa, kjer so bistveno manj dovzetne za vplive iz okolice. Vendar nam ravno ta poenostavitev zaveže roke, za morebitne optimizacije sestavnih delov PLL zanke. Po drugi strani je tu še cenovni faktor, celotna PLL zanka skupaj z VCO je precej cenejša v izvedbi z enim čipom, kot ločeno (na primer PLL čip in VCO čip). A koncepti posameznih sestavnih delov vseeno ostajajo enaki.

### 2.2.1 Fazni detektor

Idealni fazni detektor bo na svojem izhodu proizvedel napetost proporcionalni razliki faz vhodnih dveh signalov, skupaj s faktorjem ojačenja. Torej:

$$V_{izhod} = K_d(\theta_1 - \theta_2) \quad (2.5)$$

Napetost izhoda je neodvisna od amplitude vhodnih signalov, zato so ti signali navadno digitalni. Idealno delovanje faznega detektorja je v praksi težko izvesti, zato se uporablja kopica različnih prijemov, da delovanje čim bolj približamo teoretičnim zahtevam. Navadno je primerjanje izvedeno z izključujočo ALI operacijo (XOR), ki

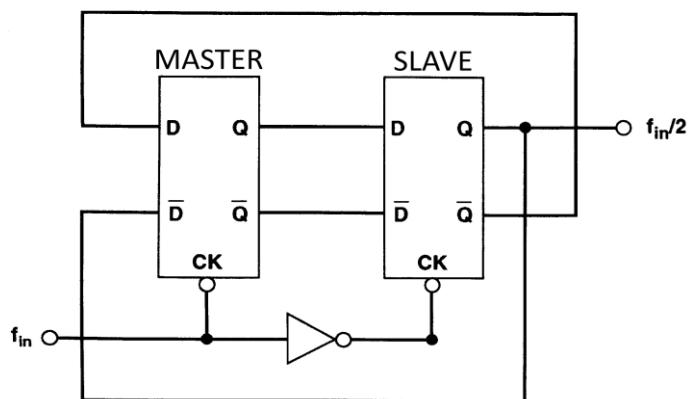
na svojem izhodu ravno tako proizvaja digitalno vrednost, katere vrednost 1/0 je enako razmerju razlike faze vhodnih signalov. Ta signal se nato pelje do integratorja, ki vrednost pretvori v analogno napetost, ali pa do črpalke naboja, ki napaja kondenzator pasivnega zančnega sita.

Slaba lastnost gradnje detektorja z XOR vrati je delovanje v območju majhnih faznih razlik blizu 0 (ali  $2\pi$ ) radianov. V tem primeru je izhodna napetost simetrična glede na izvor in je enaka ne glede na to ali je razlika v fazi na primer 0.1 ali -0.1 radianov. V PLL čipih, ki takšno strukturo uporabljajo, problem rešujejo z vnosom nizko šumne enosmerne napetosti pred vhodom v integrator, kar povzroči, da navidezno delujemo v območju faznega zasuka  $\pi/2$ . Druga možna rešitev je uporaba tako imenovanega robno proženega (ang. Edge-triggered) frekvenčno/faznega detektorja. Fazni detektor je neaktiven, ko sta signala med seboj zamaknjena za  $\pi$  radianov izhodni signal pa ima razmerje 1/0 enako 50%. Vsak drugačen fazni zasuk bo detektor spravil v aktivno stanje, kjer se sprememba faze na vhodu odraža kot sprememba razmerja 1/0 na izhodu v eno ali drugo smer, odvisno od razlike vhodnih signalov. V aktivnem stanju se detektor nahaja toliko časa, dokler razlika v fazi med obema signaloma zopet ne ustreza  $\pi$  radianov. Kadar sta fazi obeh signalov daleč stran od  $180^\circ$  funkcijo faznega detektorja prevzame pomožno vezje, ki izhod sili v stanje logične 1 ali 0 toliko časa, dokler se fazi obeh signalov ne nahajata dovolj skupaj, blizu  $180^\circ$ . Primer takšnega faznega detektorja predstavlja čip AD9901 [9]. Z uporabo takšnega faznega detektorja rešujemo dve muhi na en mah. Primerjava frekvence namreč ni možna dokler sta frekvenci med seboj močno različni, šele ko sta si dovolj blizu, lahko pravzaprav začnemo s primerjavo faze.

### 2.2.2 Delilnik frekvence

Delilnik frekvence je ključen sestavni del fazno sklenjene zanke, saj visoko frekvenco pretvori v nizko, ki jo lahko ustrezno obdelamo v faznem detektorju, ter približamo referenčnemu signalu. Večina delilnikov je izvedena z uporabo digitalnih flip-flop vezji, navadno z uporabo dveh kaskadno vezanih celic, ki delujeta v načinu gospodar-suženj in na svojem izhodu proizvajata polovično vhodno frekvenco. Takšna struktura je prikazana na sliki 2.7.

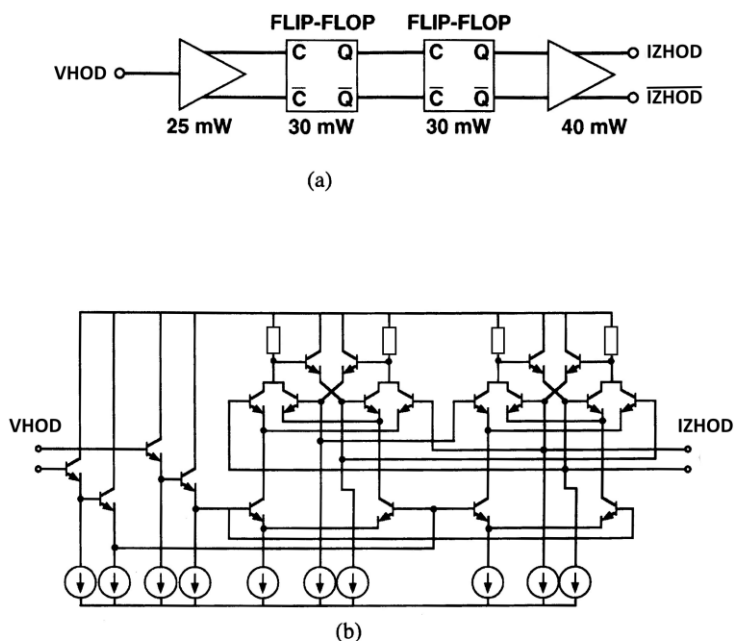




Slika 2.7: Delilnik frekvence s flip-flop celicami

Zgodovinsko jih lahko razdelimo na statične in dinamične. Statični delilniki frekvence delujejo v širokem območju vhodnih frekvenc in uporabljajo mehanizem zapaha (ang. Latch), da shranijo izhodno vrednost pred vhodom v naslednjo stopnjo ali na izhodu vezja. Dinamični delilniki navadno za shranjevanje izhodne vrednosti uporabljajo naboj na visoko impedančnih točkah vezja, kar sicer pomeni da lahko delujejo na znatno višjih frekvencah, a na ožjem frekvenčnem pasu, saj je čas hranjenja naboja znotraj vezja omejen. Zaradi robustnosti delovanja, so statični delilniki zavzeli prevlado na tržišču in se uveljavili kot zanesljiva tehnologija.

Strukturo monolitnega delilnika frekvence s 4, prikazuje slika 2.8.

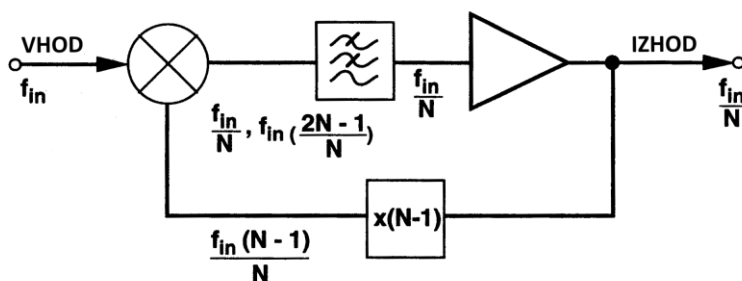


Slika 2.8: Visokofrekvenčni delilnik frekvence z modulo 4

Uporablja D tip flip-flop vezji, ter doseže območje delovanja okoli 5GHz, pri tem pa porablja manj kot 30mW na posamezno flip-flop celico.

Najvišjo frekvenco delovanja običajno določajo uporabljeni tranzistorji z mejno frekvenco  $f_T$ , kar navadno pomeni, da je najvišji takt deljenja omejen s  $f_T/2$  ali  $f_T/3$ . Struktura delilnikov je lahko tudi veliko bolj komplicirana, da ima uporabnik preko digitalnih linij možnost izbire modulo deljenja, tako za celoštevilске kot ulomkovne vrednosti [10].

Tretji tip delilnika je uporaba analognih tehnik za doseganje deljenja. Tako imenovani regenerativni frekvenčni delilniki uporabljajo mešanje, izločanje neželenih frekvenc in ojačenje za doseg želenega deljenja [11]. Posplošeno obliko takšne zanke prikazuje slika 2.9.



Slika 2.9: Regenerativni frekvenčni delilnik

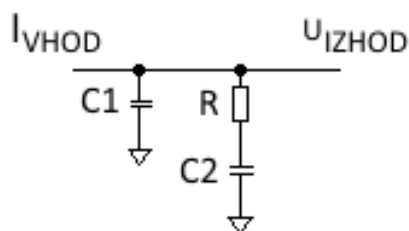
V primeru deljenja z 2, je izhod preprosto peljan nazaj na mešalnik, brez vmesnega množenja. Nizkoprepustno sito je uporabljeno da zaduši  $f_{in} + f_{out}$  komponento mešalnika, ter čezenj spusti le razliko frekvenc  $f_{in} - f_{out}$ , skupaj z ojačevalnikom pa tvori uporabno vezje delilnika. S takšnim načinom lahko izdelamo delilnike, ki delujejo na znatno višjih frekvencah od izvedbe s flip-flop celicami [11].

### 2.2.3 Zančno sito

Zadnji sestavni del PLL zanke je zančno sito. Čeprav je sito mišljeno kot eno izmed najlažjih sestavnih delov zanke, je ravno ta del skoraj vedno prepuščen načrtovalcu končne naprave, saj sito ni vključeno v monolitno izvedbo čipa. Pri tem ločimo dve izvedbi sita. Aktivno in pasivno. Aktivno sito navadno vsebuje operacijski ojačevalnik, ki skupaj s pasivnimi elementi tvori sito želenih lastnosti. Navadno takšno sito uporabimo takrat, ko naš napetostno krmiljen oscilator zahteva znatno višje napetostne nivoje na svojem vhodu, kot jih zmora fazni detektor vključno s svojimi izhodnimi stopnjami. Omejujoč faktor takšnega sita je operacijski ojačevalnik, ki mora biti zadosti hiter, hkrati pa nizko šumni, s čim nižjim tokom mirovanja, ter napetostjo

resnično nič, ko to želimo. Če se teh pravil ne držimo, kvarimo razmere delovanja (vnašamo večji fazni šum v sistem). Izbira ustreznih komponent je tako lahko včasih precej naporno delo.

Drugačna je struktura pasivnega sita. Tu ne uporabljamo aktivnih gradnikov, s tem pa vnašamo dosti manj faznega šuma v sistem, saj je omejujoč faktor aktivnih sit ravno počasni operacijski ojačevalnik. Najpogostejšo obliko pasivnega sita prikazuje slika 2.10.



Slika 2.10: Zančno sito drugega reda

Poleg kondenzatorja  $C_1$  vsebuje vezje še zaporedno vezavo upora  $R$  ter kondenzatorja  $C_2$  saj moramo v prevajalno funkcijo obvezno dodati ničlo in pol, da zagotovimo stabilnost zanke. V nasprotnem primeru se namreč lahko zgodi, da fazni zasuk povratne vezave znaša natanko  $180^\circ$  kar zagotovo pomeni nestabilnost.

Kako torej določimo vrednost elementov povratne zanke? Najprej si izberemo pasovno širino zanke  $B_{zanke}$ , ki tedaj opisuje hitrost zanke. Pravilo proizvajalcev pogosto pravi, da moramo pasovno širino zanke držati vsaj desetkrat nižjo od frekvence faznega detektorja, torej frekvenco reference (če je pred vhodom v fazni detektor ne delimo) -  $B_{zanke} \leq f_{PD}/10$  [12]. Majhna pasovna širina zanke pomeni daljši čas do vklepa in nižje neželene špičke, ki so posledica ulomkovnega načina delovanja. Širša pasovna širina pomeni krajši čas do vklepa, a višje neželene špičke. Načrtovalci PLL čipov so ta problem rešili tako, da se pasovna širina sita prilagaja glede na stanje vklepa na referenco. Ko smo v postopku vklepanja uporabljamo večjo pasovno širino, ko dosežemo želeno frekvenco, pa jo z vklopom dodatnega upora v vezje znižamo [12].

Optimalna izbira je točka, kjer sta fazna šuma VCO ter PLL sistema enaka, vendar moramo zato natančno ovrednotiti oba fazna šuma, hkrati pa tu ne upoštevamo zniževanja nadležnih špičk. S tako izbiro bo pri majhnih frekvenčnih odmikih od nosilca  $\Delta f < B_{zanke}$  prevladoval fazni šum referenčnega oscilatorja na večjih odmikih  $\Delta f > B_{zanke}$  pa fazni šum VCO. Izbiro sedaj zapišemo kot:

$$\omega_m \approx 2\pi B_{zanke} \quad (2.6)$$

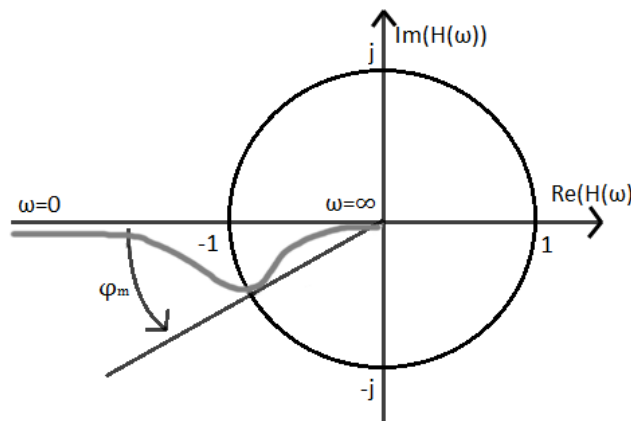
$$C_1 = \frac{K_\varphi K_{vco}}{\omega_m^2 N \sqrt{m}} \quad (2.7)$$

$$C_2 = (m - 1)C_1 \quad (2.8)$$

$$R = \frac{\sqrt{m}}{\omega_m C_2} \quad (2.9)$$

kjer so  $K_\varphi$  ojačenje faznega detektorja,  $K_{vco}$  ojačenje napetostno krmiljenega oscilatorja, oba faktorja določena z razpoložljivo tehnologijo, ter  $N$  izbrani modulo deljenja.  $m$  je izbrano razmerje frekvenc pola in ničle, ki določa pod kakšnimi pogoji bo zanka stabilna.

Preprosto in zanesljivo merilo za stabilnost povratne vezave je fazna varnost (ang. phase margin)  $\varphi_m$  oziroma kôt, ko krivulja prevajalne funkcije seka enotni krog v Nyquist-ovem diagramu, kot to prikazuje slika 2.11.



Slika 2.11: Nyquistov diagram za stabilnost povratne vezave

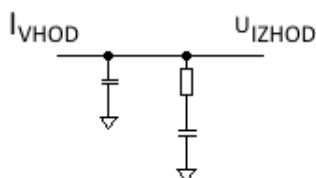
Najvišjo stabilnost zanke dosežemo takrat, ko se nevarni točki -1 karseda daleč izognemo, torej ko prevajalna funkcija seka enotni krog v točki 1. To bi pomenilo izbiro  $m = \infty$  in posledično  $C_1 = 0$ , kar bi dalo odlično stabilnost zanke, vendar pri vrednostih  $N \gg 1$  popolnoma neuporabne rezultate, saj bi izhod črpalke nabojev neposredno moduliral vhod v VCO, to pa bi rezultiralo v širok frekvenčni spekter, ter pokvarjen fazni šum. Smiselna je izbira  $C_1 > 0$  ter posledično  $m < \infty$ . Navadno  $m$  izbiramo med območjem 3 in 20. Spodnja meja daje hitrejšo zanko, zgornja pa

zagotavlja večjo stabilnosti tudi ko se  $N$ ,  $K_\varphi$  ter  $K_{vco}$  spreminjajo. Preračun za nekaj izbranih faktorjev  $m$  je prikazan v tabeli 2.1. Seveda nam delno zavezane roke držijo tudi elementi, ki so na tržišču lahko dobavljivi.

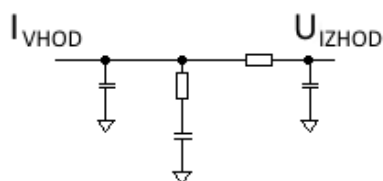
$m$	$C_2/C_1$	$\varphi_m [^\circ]$
1	0	0
1.1	0.1	2.7
1.5	0.5	11.5
2	1	19.5
3	2	30.0
10	9	54.9
20	19	64.8
100	99	78.6
200	199	81.9

Tabela 2.1: Preračun razmerja kondenzatorjev in fazne varnosti glede na izbrani  $m$

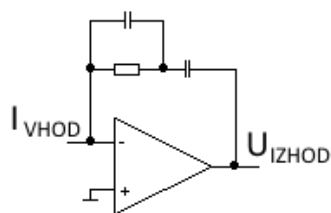
Poleg predstavljene oblike sita, lahko izberemo tudi sita višjega reda, tako pasivne kot aktivne. Nekaj jih prikazujejo slike 2.12, 2.13, 2.14, 2.15.



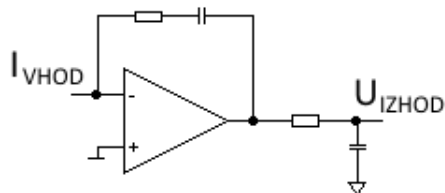
Slika 2.12: Pasivno sito drugega reda



Slika 2.13: Pasivno sito tretjega reda

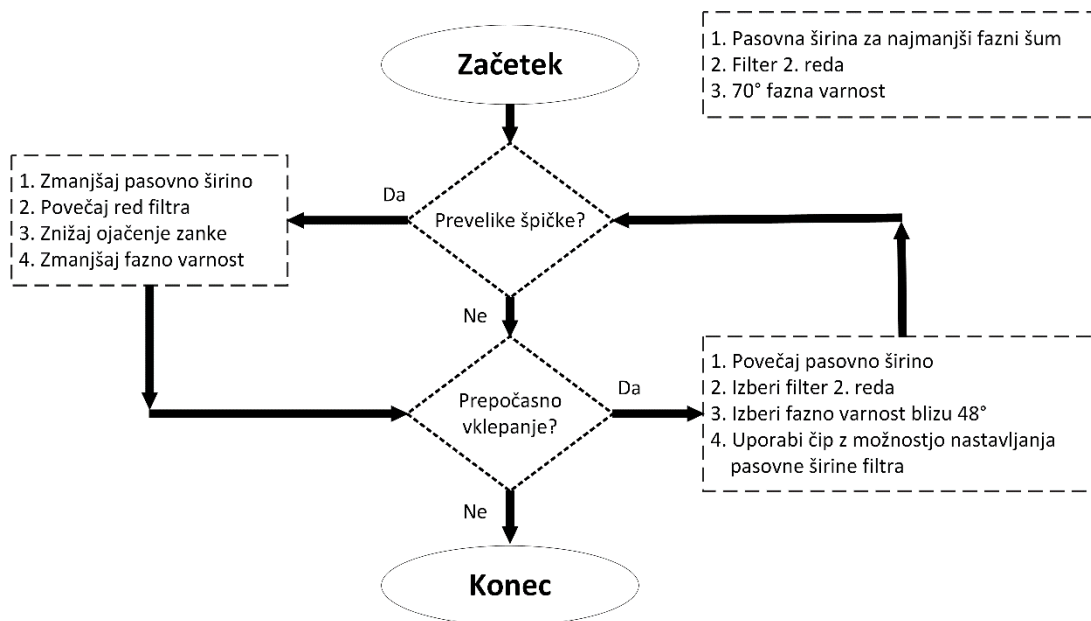


Slika 2.14: Aktivno sito drugega reda



Slika 2.15: Aktivno sito drugega reda

Toda za kakšno sito se odločiti? Pomaga nam lahko diagram za odločitev, ki je prikazan na sliki [12].



Slika 2.16: Diagram odločitve za izbiro zračnega sita

## 2.3 Zaključek

Sodobni PLL čipi, ki jih lahko kupimo na tržišču v svoji notranjosti vsebujejo celo paleto različnih možnosti. Od množenja referenčnega signala, do različnih načinov delovanja, možnosti delovanja z različnimi filtri, nastavljanja linearnosti črpalke naboja. Žal pogosto nimamo vpogleda v notranjost čipa, še posebej ne v logiko za delo z ulomki. Tako natančno ne vemo, kako čip premeša vrednosti med dvema celoštevilskima vrednostnima, da dobimo želen ulomek, ravno tako je težko reči, kaj naredi način za zniževanje nadležnih špičk. Ena izmed možnosti za določanje vpliva je torej meritev.





## 3 Fazni šum

Poglavitni problem inženirjev radijskih zvez je izločanje želenega signala iz šuma. Ko sestavimo najnaprednejši radarski ali telekomunikacijski sistem, kjer so naši napori posvečeni pridobivanju karseda veliko informacij iz signala, slej kot prej naletimo na omejitve, ki jih predstavlja fazni šum.

V splošnem, kadar govorimo o faznem šumu, govorimo o frekvenčni stabilnosti signala. Na frekvenčno stabilnost lahko gledamo iz različnih zornih kotov. Pogosto nas zanima dolgoročna stabilnost oscilatorja opazovana v urah, mesecih ali letih. Mnogi oscilatorji, ki so dolgoročno stabilni (na primer Rubidijeva časovna normala), imajo lahko zelo velik fazni šum. Kadar govorimo o faznem šumu nas zanimajo kratkotrajne spremembe, manjše od sekunde [13]. Lahko rečemo tudi, da je fazni šum frekvenčna domena hitrih, kratkih in naključnih sprememb zaradi nestabilnosti v časovni domeni [14].

### 3.1 Šum oscilatorja

Idealni oscilator bi proizvajal čisti sinusni signal. V frekvenčni domeni bi to predstavili kot par Dirakovih delta funkcij (pozitivna in konjugirana negativna komponenta) na mestu frekvence oscilatorja, torej vsa moč signala se nahaja na eni sami frekvenci. Realni oscilatorji vsebujejo fazno modulirane komponente šuma. [14]. Če predstavimo brezšumni signal z enačbo:

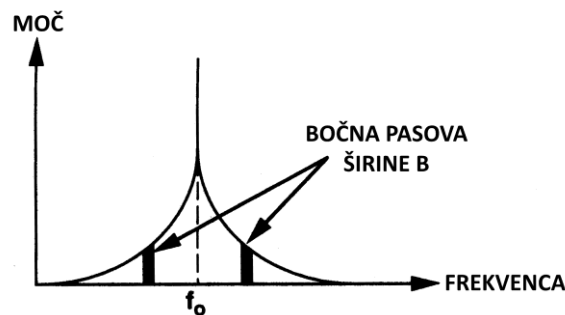
$$v(t) = A * \cos(2\pi f_0 t) \quad (3.1)$$

je fazni šum dodan takšnemu signalu kot stohastični proces predstavljen z  $\varphi$  v enačbi:

$$v(t) = A * \cos(2\pi f_0 t + \varphi(t)) \quad (3.2)$$

Vsak izvor šuma znotraj aktivnih naprav oscilatorja, bo moduliral izhodni signal na tak način, da v frekvenčnem spektru poleg nosilnega signala obstajajo še šumni

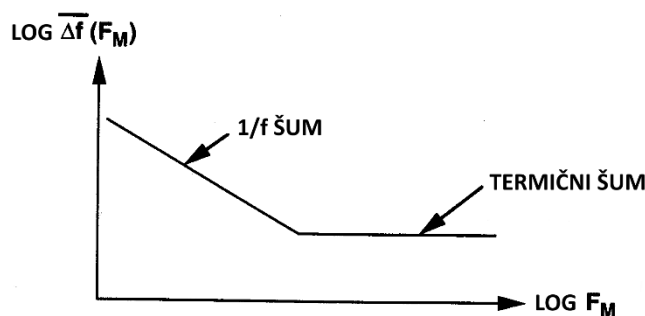
bočni pasovi. Šum lahko razdelimo v tri razrede: AM šum, ki je posledica amplitudne modulacije šumnih izvorov, FM šum, kot posledica frekvenčne modulacije izvorov šuma in PM šum, zaradi fazne modulacije šumnih izvorov. FM in PM šum proizvajata enak rezultat v spektru, zato jih navadno v obravnavi med seboj ne ločujemo [15].



Slika 3.1: Spekter šuma

Fazni šum izvira iz šuma gradnikov našega oscilatorja. Dva primarna tipa šuma v tranzistorjih sta nizkofrekvenčni  $1/f$  šum zaradi zastajanja naboja v polprevodniških plasteh ter termični (beli) šum. Ta dva izvora šuma modulirata osnovno frekvenco signala in rezultirata v AM in FM šumnih pasovih okoli osrednje frekvence.

Spekter FM šuma prikazuje slika 3.2 [15].



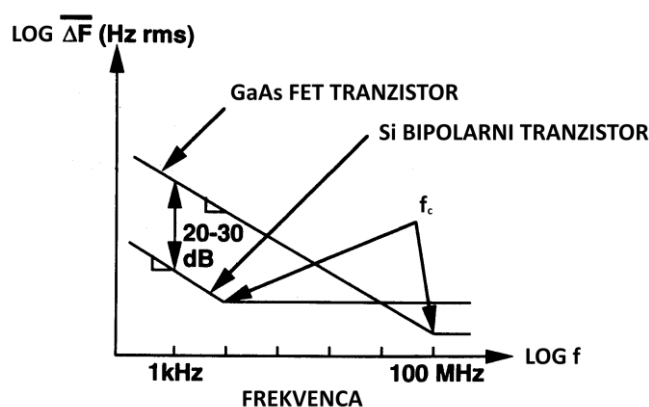
Slika 3.2: FM šum

Moč toplotnega ali termičnega šuma zapišemo z enačbo:

$$P_n = \Delta f k_B T \quad (3.3)$$

kjer je  $T$  navadno 293K,  $k_B$  Boltzmannova konstanta  $k_B \approx 1.38064852 \cdot 10^{-23} \text{ J/K}$ ,  $\Delta f$  pa pasovna širina.

Šum  $1/f$  običajno nima jasne fizikalne razlage. Odvisen je od gradnika, oziroma tehnologije izdelave. Iz tega sledi, da izbira aktivne naprave v oscilatorju ni samoumevna. Slika 3.3 prikazuje primerjavo spektra  $1/f$  šuma Si bipolarnega tranzistorja in GaAs FET tranzistorja. Ker je  $1/f$  šum bipolarnega tranzistorja znatno manjši kot GaAs FET-a, ima prednost tam, kjer želimo čim manj šuma v bližnji okolici nosilca. Kadar nam je bolj pomemben šum daleč stran od nosilca bomo tako izbrali GaAs FET, saj je njegov termični šum nižji v primerjavi z bipolarnim tranzistorjem.



Slika 3.3: Spekter šuma Si bipolarnega tranzistorja in GaAs FET tranzistorja

### 3.2 Definicija normiranega faznega šuma

Zgodovinsko sta obstajali dve nasprotujoči si, a pogosto uporabljani definiciji faznega šuma. Obe definiciji data enake rezultate pri velikih frekvenčnih odmikih od osnovne frekvence, medtem ko se pri manjših odmikih razlikujeta.

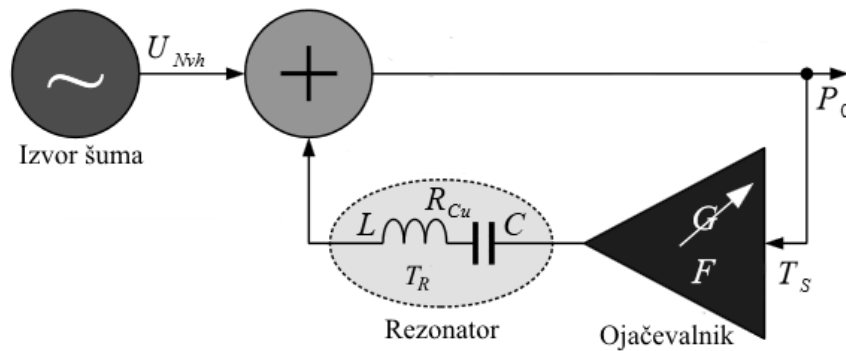
IEEE definicija določa fazni šum kot:

$$\mathcal{L}(\Delta f) = \frac{P_i/2}{BP_0} \quad (3.4)$$

kjer je  $P_0$  moč nosilca,  $B$  pasovna širina, ter  $P_i$  moč na izbranem frekvenčnem odmiku  $\Delta f$ . Fazni šum  $\mathcal{L}(\Delta f)$  je običajno izražen v enotah dBc/Hz kot  $10 \log_{10}(\mathcal{L}(\Delta f)_{LIN})$  in predstavlja moč šuma normirano na nosilec, v 1Hz pasovne širine s centrom na želenem odmiku od nosilca. Izbrani signal ima tako lahko fazni šum  $-60$ dBc/Hz pri odmiku 10kHz, ter fazni šum  $-85$ dBc/Hz pri odmiku 100kHz od nosilca. Fazni šum je lahko izmerjen in izražen z upoštevanjem enega ali obeh bokov, vendar IEEE veleva upoštevanje enega boka [14].

### 3.3 Leesonova enačba

Literatura potek faznega šuma z odmikom od nosilca pogosto opisuje z empirično pridobljenimi podatki. Čeprav tak način na prvi pogled izgleda smiselno, je možno z nekaj matematične spretnosti vse te pojave zajeti v enačbi. Predpostavimo preprost oscilator kot ga prikazuje slika 3.4.

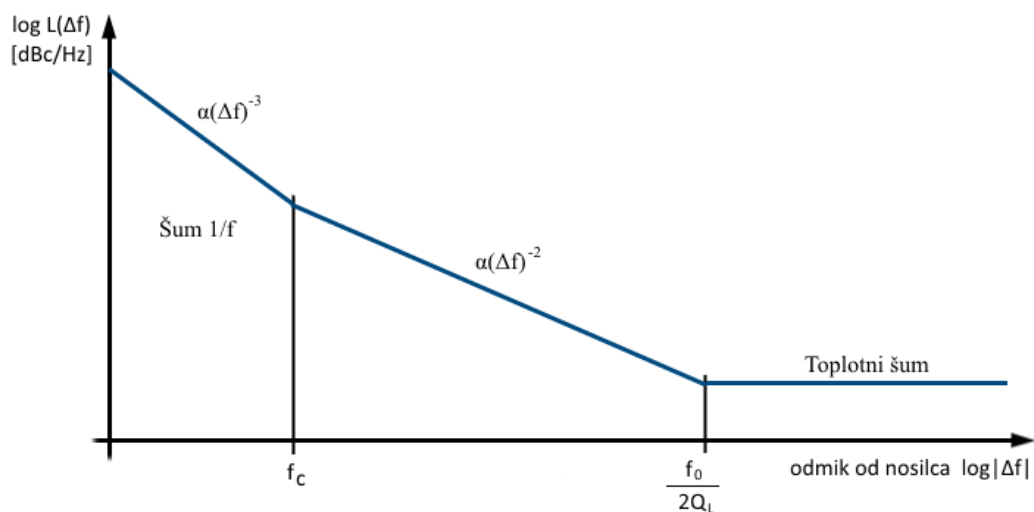


Slika 3.4: RLC oscilator [16]

V zanko se aditivno dodaja šum iz nadomestnega izvora šuma. Šum  $T_R$  je približno enak temperaturi okolice  $T_0 = 290\text{K}$ .  $P_0$  je moč nosilca,  $G$  ojačenje,  $F$  pa šumno število ojačevalnika. Porazdelitev šuma na vhodu opišemo kot:

$$\frac{dP_{Nvh}}{df} = N_0 = k_B(T_R + T_S) \approx k_B T_0 F \quad (3.5)$$

Potek normirane gostote faznega šuma prikazuje slika 3.5.



Slika 3.5: Normirana spektralna gostota faznega šuma

Potek opisuje Leesonova enačba:

$$L(\Delta f) = \frac{1}{P_0} \cdot \frac{dP_\varphi}{df} = \frac{1}{2} \cdot \left[ 1 + \left( \frac{f_0}{2Q_L \Delta f} \right)^2 \right] \cdot \frac{k_B T_0 F}{P_0} \cdot \left( 1 + \frac{f_c}{|\Delta f|} \right) [Hz^{-1}] \quad (3.6)$$

Deljenje z 2 nam da samo fazni šum, nasičenje pa odstrani amplitudni šum. Pri tem je  $Q_L$  kvaliteta obremenjenega resonatorja. Seveda je mogoče enačbo prevesti tudi v logaritemsko skalo:

$$L(\Delta f)_{dBc/Hz} = 10 \log_{10} L(\Delta f) \cdot 1Hz [dBc/Hz] \quad (3.7)$$

Kvaliteta obremenjenega resonatorja je tako ključnega pomena za fazni šum. Če se kvaliteta varikap LC oscilatorja nahaja med 10 in 30, lahko doseže kvaliteta YIG oscilatorja vrednosti preko 300, to pa pomeni boljši fazni šum [16].

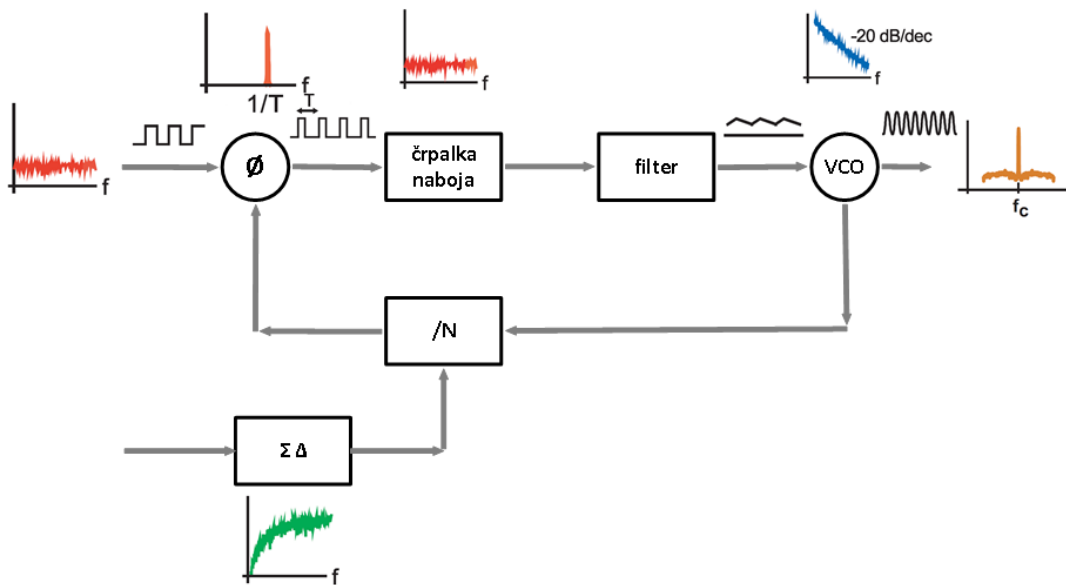
### 3.3 Fazni šum PLL

V strukturi PLL zanke se nahaja precejšnje število faktorjev, ki vplivajo na šum in obliko spektra signala na izhodu, večina od teh je šumnih izvorov, ki na nek način modulirajo vhod v VCO. Osnovni izvor faznega šuma je vsebovan v samem oscilatorju, kot je to opisano v poglavju 3.1 Šum oscilatorja. Poleg šuma oscilatorja se v PLL strukturi nahajajo šumi povezani s pred delilnikom, šum zančnega sira, faznega detektorja, N delilnika.

V realnih sistemih se pojavi še vpliv šumnih napajalnih linij, presluhov med povezavami, resonančnih frekvenc ki so posledica slabe izbire pasivnih komponent, ohišje čipa, bondirne žice znotraj čipa. Vse te vplive lahko s pametnim načrtovanjem znatno zmanjšamo, nekatere pa še vedno ostajajo v rokah načrtovalcev PLL čipov in na njih nimamo vpliva.

#### 3.3.1 Fazni šum ulomkovnega PLL

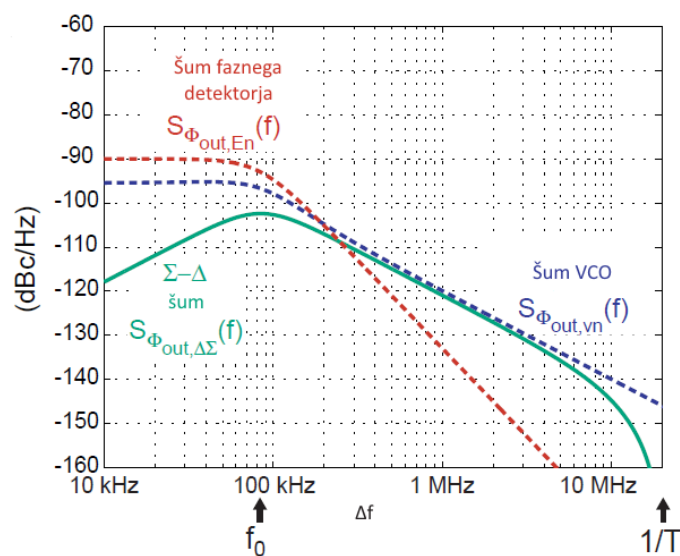
Predstavitev prispevka k faznemu šumu za ulomkovni PLL je lažja grafično. Slika 3.6 prikazuje prispevek in obliko faznega šuma posameznega elementa v verigi, ter obliko spektra signala na izhodu, ki je posledica vseh prispevkov [17].



Slika 3.6: Prispevki k faznemu šumu sestavnih delov ulomkovne PLL zanke

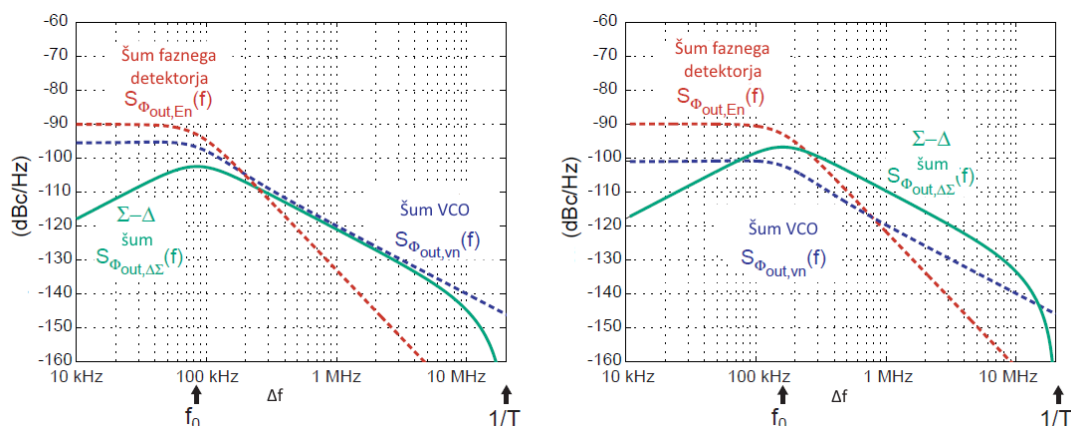
Šum reference na izhodu je dominantnejši pri nižjih frekvenčnih odmikih od nosilca, šum VCO prevladuje pri večjih. Sigma-delta šum ravno tako prevladuje na nižjih frekvenčnih odmikih, njegove višje komponente duši struktura PLL. Zato je Sigma-delta način boljša izbira za doseganje nižjega faznega šuma, kadar potrebujemo ulomkovni način delovanja PLL zanke [17].

Slika 3.7 [17] prikazuje iste prispevke na istem grafu. Fazni šum je izražen v enoti dBc/Hz. Širina zančnega sita je v tem primeru izbrana in simulirana za  $f_0 = 84\text{kHz}$ .



Slika 3.7: Fazni šum ključnih komponent ( $B = 84\text{ kHz}$ )

Poglejmo kaj se spremeni, ko povečamo pasovno širino sita na  $f_0 = 160\text{kHz}$ , kar prikazuje slika 3.8 [17].



Slika 3.8: Primerjava vpliva na fazni šum pri spremembi pasovne širine sita iz 84 kHz na 160 kHz

V drugem primeru na izhodu vidimo večji vpliv šuma faznega detektorja, večji vpliv šuma delta-sigma modulatorja, ter manjši vpliv šuma VCO. Izbira zanke torej ni pomembna le iz vidika stabilnosti zanke, temveč tudi iz vidika faznega šuma celotnega sistema.

### 3.4 Meritev faznega šuma

Fazni šum je mogoče izmeriti s pomočjo spektralnega analizatorja, če je fazni šum testne naprave manjši od faznega šuma lokalnega oscilatorja spektralnega analizatorja. Pri tem moramo seveda paziti, da ne merimo odziva filtra, namesto faznega šuma. Večina spektralnih analizatorjev višjega cenovnega razreda, danes na mestu lokalnega oscilatorja uporablja YIG oscilator ((Yttrium-Iron Garnet), ki ima na visokih frekvencah znatno višjo kvaliteto  $Q_L$  od na primer LC nihajnega kroga ( $100\times$  višja). Obratno ima pri nizkih frekvencah LC nihajni krog znatno nižji fazni šum od merilnega inštrumenta, zato je meritev faznega šuma pri nizkih frekvencah povsem nesmiselna. Tako tudi faznega šuma kristalnih oscilatorjev, ki dosegajo kvaliteto  $Q_L > 3000$ , s spektralnim analizatorjem ne moremo izmeriti.

Pri meritvi moramo paziti, da je izbrana ločljivost spektralnega analizatorja vsaj desetkrat manjša od frekvenčnega odmika  $\Delta f$ . Zaradi lažje meritve moramo šum povprečiti. To napravimo z vključitvijo video sita, njegovo pasovno širino pa nastavimo na veliko manj od ločljivosti spektralnega analizatorja ( $B_{VIDEO} \ll B$ ). Pri tem moramo odčitnemu povprečju prišteti še faktor povprečenja, ki ja za Gauss-ov šum enak 2.51dB.

Sodobni spektralni analizatorji lahko šum povprečijo digitalno. Video sito lahko popolnoma izklopimo, ali pa ga izberemo tako, da je hitrost preleta še vedno zadovoljiva. Število točk povprečenja poljubno nastavljam, da dosežemo dovolj čisto črto in da meritev traja razumen čas. Prednost takega povprečenja je v tem, da lahko spektralni analizator med vsakim preletom sledi maksimumu signala, ter ga pred meritvijo centrira. Tako manjša nihanja v frekvenci signala ne motijo meritve. Ne glede na uporabljeno metodo, pridemo do istega rezultata.

Smiselne meritve faznega šuma se nahajajo v območju med frekvenco preloma  $f_c$ , ki jo določa FM šum aktivnih gradnikov (poglavje 3.1 Šum oscilatorja) do frekvence  $f_0/2Q_L$ . V tem področju velja poenostavljena Leeson-ova enačba ki pravi, da je fazni šum sorazmeren kvadratu odmika frekvence od nosilca. Na manjših odmikih zveza ne velja več, saj fazni šum ni več zadosti majhen v primerjavi z močjo nosilca, dlje pa prevladuje šum spektralnega analizatorja.

Za meritve faznega šuma lahko uporabimo tudi posebne naprave, ki z svojo strukturo, notranjim ali zunanjim virom omogočajo meritve lastnega in zunanjega šuma. Takšni sistemi navadno omogočajo tudi natančnejše meritve faznega šuma pri manjših frekvenčnih odmikih od nosilca [14].



## 4 Merjenec

Teorija zna poiskati vsak izvor faznega šuma znotraj PLL zanke. Žal se je skoraj nemogoče prebiti do uporabnih podatkov, ki bi svoje trditve podkrepile z realnimi meritvami. Kako torej na fazni šum vpliva ločevanje mas pod čipom, tok črpalke naboja, njena linearizacija, različni načini za doseganje boljšega faznega šuma, preklapljanje med mnogimi oscilatorji v notranjosti? Podatkovni listi proizvajalcev so tu modro tiho. Po navadi v njih najdemo le osnovne meritve v najboljših slučajih, pa še v to lahko pogosto dvomimo. V nadaljevanju je predstavljena gradnja visoko frekvenčnega izvora z uporabo ulomkovne PLL zanke na podlagi čipa MAX2871 ter izdelava tiskanine.

### 4.1 Izbira gradnikov za gradnjo PLL zanke

Za začetek si postavimo omejitve:

1. Tiskanino lahko izdelamo v domači delavnici
2. Komponente so lahko dobavljive, njihova vrednost pa ne presega nekaj 10€

Največjo omejitev zagotovo postavlja 1. pogoj. Tako odpadejo vsi čipi, ki so na voljo le v BGA pakiranju. Privoščiti si ne moremo niti vij, ter večslojnih tiskanin. Tudi pri izbiri laminata smo precej omejeni, saj je najlažje dobavljiv material za mojstre v domači delavnici FR4. Če PLL čip še nekako spravimo na 0.8 mm FR4 ploščico, bomo za oscilator, ki tiktaka na frekvencah nad 10 GHz že potrebovali drugačno dielektrično konstanto, saj bi bila debelina povezav za doseganje 50 ohmske impedance enostavno prevelika. Frekvenčni razpon sintetizatorja omejuje 2. točka. Če si zaželimo širokopasovni napetostno krmiljen oscilator (nekaj GHz) bo tehnologija prišla v keramičnem ohišju na Galijeve Arzenidu, za kar pa bo treba kar pošteno seči v žep. Torej uporaba ločenih PLL in VCO čipov nekako ni najbolj stroškovno učinkovita, pri visokih frekvencah bomo imeli tudi težave s tiskanino. Še večja težava je, če želimo

PLL zanko implementirati sami v FPGA, saj na njegov vhod ne moremo direktno peljati 6 GHz velikega signala, pred tem ga moramo ustrezno deliti. Delilniki za visoke frekvence tudi niso poceni, poleg tega jih v verigi potrebujemo več, da signal sploh spravimo na frekvenco primeroma za delo s FPGA.

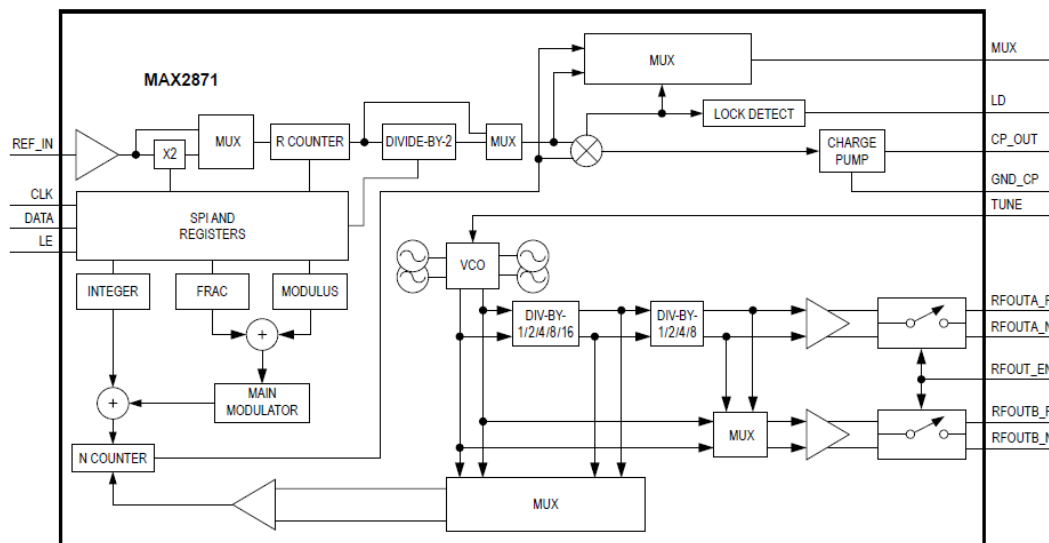
Druga pot je izbira čipov, ki v enem ohišju nosijo tako PLL vezje kot napetostno krmiljen oscilator. Pravzaprav je v notranjosti oscilatorjev več, tudi preko 60. S tem so načrtovalci vso stvar izdelali na siliciju, kjer je tehnologija izdelave dovršena do potankosti, zaradi masovne proizvodnje pa so nizki tudi stroški. Tako odpade potreba po dragem keramičnem ohišju, ki jo zahteva Galijev Arzenid, vendar pa samo z enim oscilatorjem navadno ne moremo pokriti celotnega frekvenčnega območja. Zato jih imamo v notranjosti na voljo več, njihova območja pa se med seboj prekrivajo, s čimer zagotovimo nemoteno delovanje v precej širokem območju.

Večina takšnih čipov za doseganje frekvenc nad 4 GHz na izhodu uporablja množilnik frekvence, ki pa na žalost viša fazni šum za  $20 \log(N)$ , v našem primeru množenja z dva torej  $20 \log(2) = 6.02 \text{ dBc/Hz}$  [18]. Če si želimo oscilator z najnižjim možnim, to seveda ni zeleno.

V času začetka nastajanja tega dela, se je na tržišču pojavil nov čip podjetja Maxim Integrated MAX2871 [19], ki omogoča frekvenčni razpon od 3 GHz do 6 GHz brez uporabe delilnikov ali množilnikov izhodne frekvence, seveda skupaj s PLL v enem čipu. Cena takrat novega čipa ni presegala 15€, le nanj je bilo potrebno čakati dobra dva meseca. Ravno zato je bil izbran kot popoln kandidat za ovrednotenje faznega šuma takšnih čipov.

## 4.2 MAX2871

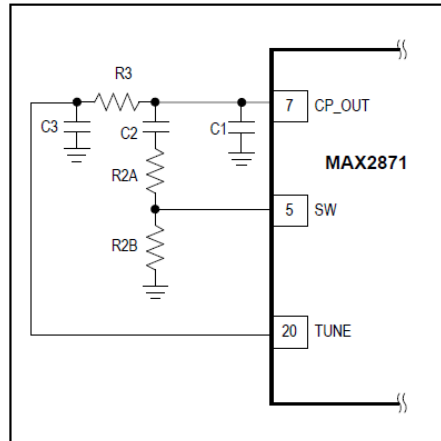
Čip MAX2871 (ter njegov manj zmogljivejši brat MAX2870 – slabši fazni šum) je edini širokopasovni PLL čip z vgrajenimi napetostno krmiljenimi oscilatorji, možnostjo delovanja v celoštevilskem in ulomkovnem načinu, ki ga proizvaja podjetje Maxim Integrated [19]. Kadar ga opremimo z zunanjim referenčnim oscilatorjem in znančnim sitom, je sposoben generiranja frekvenc v območju med 23.5 MHz do 6 GHz, hkrati pa (tako trdi proizvajalec) ohranja nizek fazni šum, ter nadležne špičke. Notranjo strukturo čipa prikazuje slika 4.1.



Slika 4.1: Blokovna shema MAX2871

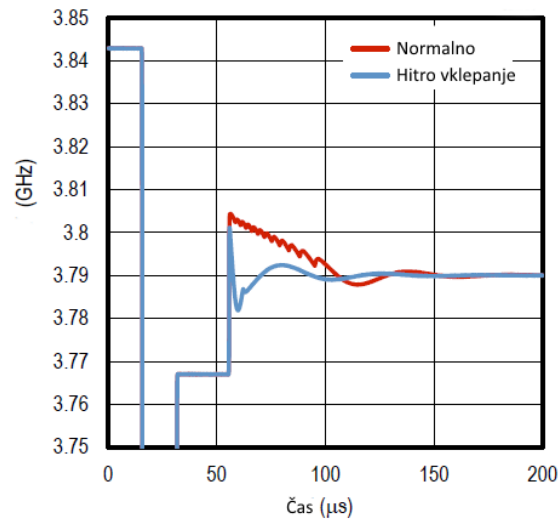
Frekvenčno območje dosega z vgrajenimi 64 oscilatorji, ki delujejo v območju med 3 in 6 GHz, od katerih vsak posamezen pokrije območje okoli 100 MHz. MAX je s svojo interno logiko sposoben izbrati pravi oscilator za želeno frekvenco, svojo odločitev pa lahko nadgradi tudi z merjenjem temperature jedra in posledično boljšo izbiro oscilatorja za dano temperaturo. Dva diferencialna izhoda omogočata moč do +4 dBm, izhodno frekvenco pa lahko delimo z nastavljenim modulom med 1 in 128. Nad obema izhodoma imamo popoln nadzor in ju po potrebi tudi ugasnemo.

Primerjalnik faze lahko tiktaka z najvišjo frekvenco okoli 140 MHz, vhod referenčnega signala pa dovoljuje frekvence do 210 MHz. Digitalno lahko spreminjamo tudi fazo izhodnega signala, ter tako po želji sinhroniziramo več čipov v sistemu. Proizvajalec priporoča uporabo pasivnega zančnega sita tretjega reda, ki ga načrtujemo tako, da mu z vklopom ali izklopom dodatnega upora, nastavljamo serijsko upornost. S tem si privoščimo hitrejše vklepanje zanke, brez da bi kvarili lastnosti skrbno načrtovanega sita. Strukturo takšnega sita prikazuje slika 4.2.



Slika 4.2: Struktura zančnega sita za hitrejši vklep

Pri tem moramo upornost R2 razdeliti na dva dela in sicer  $R2A = \frac{1}{4} R2$ , ter  $R2B = \frac{3}{4} R2$ . Ko se fazna zanka skuša vkleniti na referenčni signal, notranja logika nastavi izhodni tok črpalke naboja na najvišjo možno vrednost, pri tem pa preklopi vhod SW iz stanja visoke impedance na maso. Tako učinkovito izključi upor R2B iz veje. Po določenem času, ki ga določi in izbere načrtovalec glede na časovno konstanto sita, vhod SW ponovno preide v stanje visoke impedance, tok črpalke naboja pa se zniža na najmanjšo vrednost. Razliko v času do vklepa prikazuje slika .



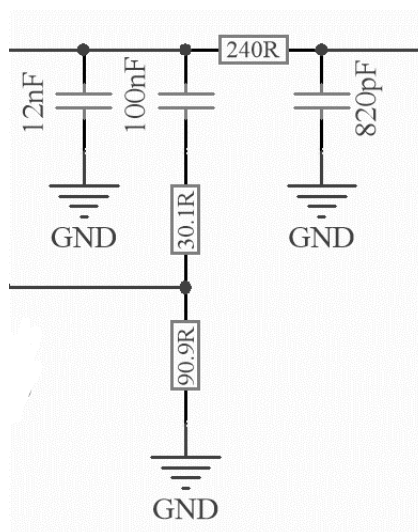
Slika 4.3: Primerjava časa do vklepa zanke s hitrim načinom in brez

S čipom se pogovarjamo preko standardnega SPI vmesnika, dodatno lahko z logičnim signalom neodvisno od interne logike vklopimo ali izklopimo izhodni signal. Na voljo imamo še izhodni signal LD ki sporoča uspešen vklep zanke, ter vhod ali izhod MUX, katerega funkcijo mu programsko nastavimo.

## 4.3 Načrtovanje prototipa

### 4.3.1 Zančno sito

Če želimo preveriti navedbe proizvajalca o doseženem faznem šumu je najbolje, da da se držimo komponent, ki jih je za preizkus uporabil proizvajalec. Vrednosti elementov zančnega sita najdemo v podatkovnem listu, pomagamo pa si lahko tudi s shemo demonstracijske tiskanine, ki jo za bajno vsoto ponuja proizvajalec [20]. Izsek iz shematike celotnega vezja, ki prikazuje zančno sito je zajet na sliki 4.4.



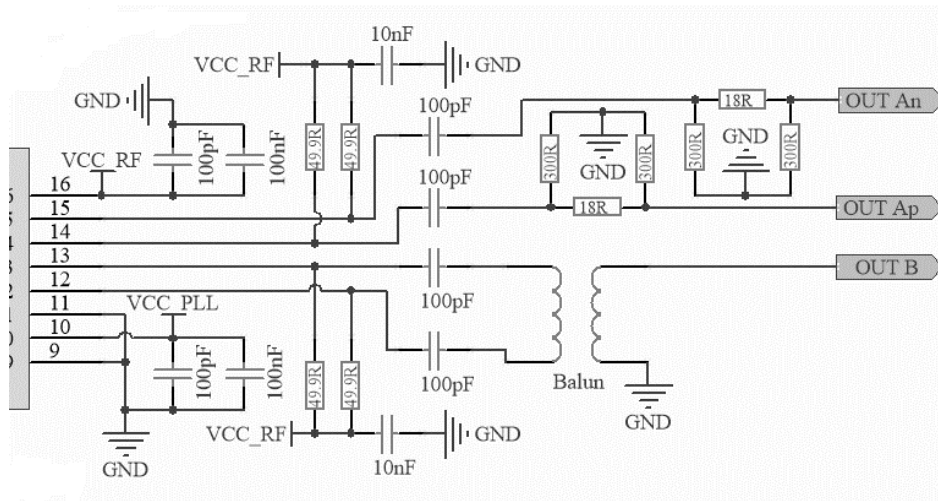
Slika 4.4: Shema uporabljenega zančnega sita

Sito omogoča uporabo hitrega načina za vklop, je načrtovano za frekvenco faznega detektorja 50 MHz, ter ima pasovno širino okoli 65 kHz, odvisno od točnosti uporabljenih komponent. Vsi upori so točnosti 1%, kondenzatorji pa iz keramike tipa X7R z izjemo 820pF, ki uporablja keramiko tipa NP0.

### 4.3.2 Visokofrekvenčni izhod

Ker visokofrekvenčni izhod v notranjosti vsebuje diferencialno strukturo z odprtim kolektorjem, potrebuje zunanji 50 Ohm upor ali tuljavo na napajanje, da lahko deluje. Tuljavica je sicer primernejša izbira, saj preprečuje vdor visokofrekvenčnega signala na napajalno linijo, vendar je z njeno uporabo težko zagotoviti optimalno delovanje v celotnem frekvenčnem območju. Zato sem izbral 50 Ohm pull-up upor in za krmiljenje izhodov uporabil ločen nizko šumni napetostni regulator, čigar napetost je še dodatno filtrirana s feritom v obliki SMD čipa. Prvi prototip je na enem izmed izhodov vseboval tuljavice, kar pa se je z meritvami izkazalo za nepotrebno komplikacijo, zato jo je v drugi različici prototipa nadomestil upor.

Proizvajalec priporoča tudi, da neuporabljene izhode zaključimo na prilagojeno breme, ali pa uporabimo vsaj -3 dB slabilec na vseh izhodnih linijah. Slednje sem tudi sam vključil v prototip, izhodni del pa je prikazan na sliki 4.5.



Slika 4.5: Visoko frekvenčni izhodni del vezja

### 4.3.3 Širokopasovni balun

Diferencialni izhod B je zaključen na širokopasovni balun TCM1-83X+ podjetja Mini Circuits, ki je bil brezplačno pridobljen kot vzorec za izdelavo magistrskega dela. Balun uporabimo z željo, da bi izločili vpliv motečih digitalnih signalov v notranjosti, ki bi lahko povzročali neželene produkte v frekvenčnem spektru izhodnega signala. Zaradi uporabe diferencialnega izhoda bi se ta šum med sabo ravno odštel, ostal pa bi samo čist želeni signal. TCM1-83X+ je prilagojen na 50 Ohm impedanco, njegova zgornja frekvenca pa dosega 8 GHz. Pred vgradnjo v sistem sem z meritvami na vektorskem analizatorju vezji, preveril navedbe proizvajalca o vstavitvenem in povratnem slabljenju. Merilni vzorec je prikazan na sliki 4.6.

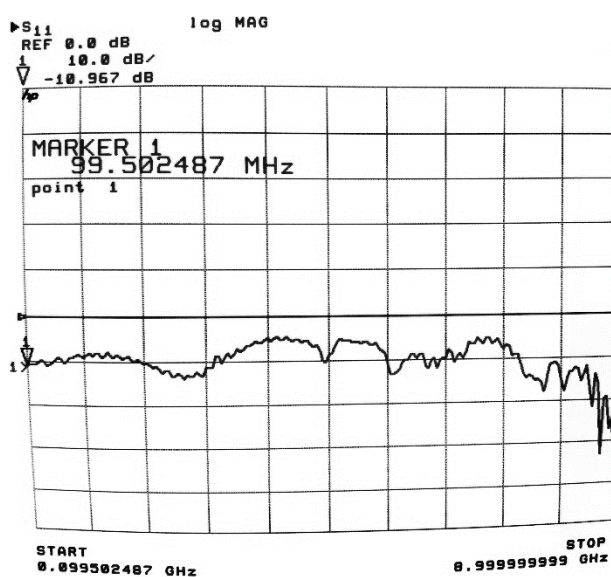


Slika 4.6: Vezje za merjenje karakteristika širokopasovnega baluna

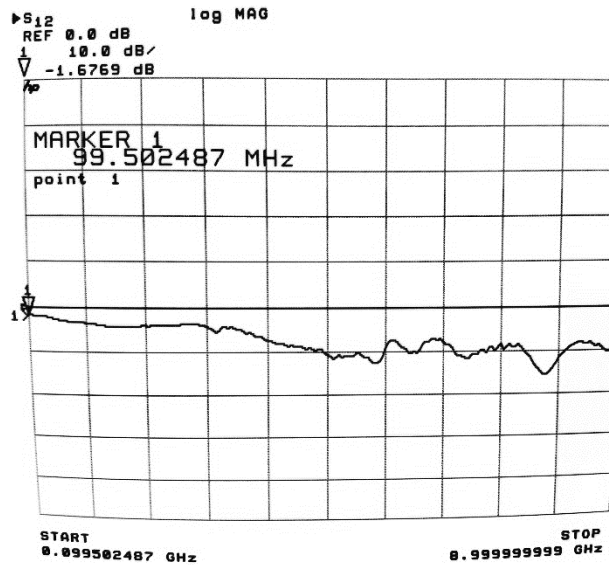
Uporabimo dva baluna iste serije, ki ju vežemo tako, da prvi signal najprej pretvori v diferencialno obliko, drugi pa zopet nazaj. Seveda pri rezultatih upoštevamo uporabo dveh elementov.

#### 4.3.4 Rezultati meritev TCM1-83X+

Za meritev je bil uporabljen vektorski analizator vezji HP 8510V z enoto za merjenje S parametrov HP 8515A, ter frekvenčnim izvorom HP 83621A. Pred meritvijo je bila izvedena polna kalibracija, tudi z upoštevanjem referenčne linije (povezava spodaj). Meritev parametra  $|S_{11}|$  je prikazana na sliki 4.7, meritev parametra  $|S_{12}|$  pa na sliki 4.8. Vse meritve so izvedene v območju od 100 MHz to 9 GHz.



Slika 4.7: Meritev TCM1-83X+ |S11|



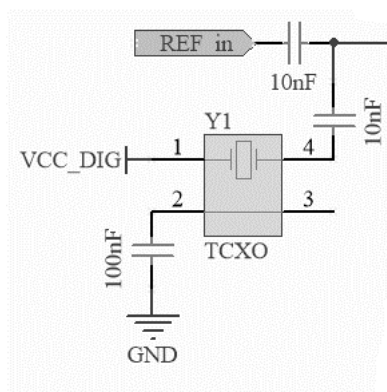
Slika 4.8: Meritev TCM1-83X+ |S12|

Meritve vstavitvenega slabljenja se še nekako ujemajo s podatki iz podatkovnega lista (ko upoštevamo da uporabljamo dva elementa), vendar pa je neprilagojenost precej višja od tistega v podatkovnem listu (za okoli 8 dB). Morda je smiselno dodati še, da se postavitvi elementov in povezav, ki jih priporoča proizvajalec s tiskanino domače izdelave le približamo, vij pod čipom namreč ne moremo izdelati. Vpliv baluna na fazni šum je predstavljen v poglavju Rezultati meritev.

#### 4.3.5 Referenčni vhod

Kot izvor referenčne frekvence uporabljam temperaturno kompenziran kristalni oscilator (TCXO) FOX924B [21]. Zaradi lažje dobavljivosti sem uporabljal oscilator s taktom 20 MHz. Po informacijah z industrijskega projekta, naj bi točno ta tip oscilatorja povzročal neželene diskretne skoke frekvence, ko svoje delo začne opravljati logika za temperaturno kompenzacijo frekvence. Z mojimi poskusi tega nisem mogel potrditi. Frekvenca se je kompenzirala zvezno, tudi pri hitrem segrevanju in ohlajanju, ker sem potrdil tudi na spremembi izhodne frekvence PLL, ki se s kompenzacijo lepo vrne nazaj na želeno vrednost. Ta del vezja prikazuje slika 4.9. V vezju je predviden priključek za zunanjo referenco, napajanje TCXO pa lahko izklopimo preko mostička na sami tiskanini.

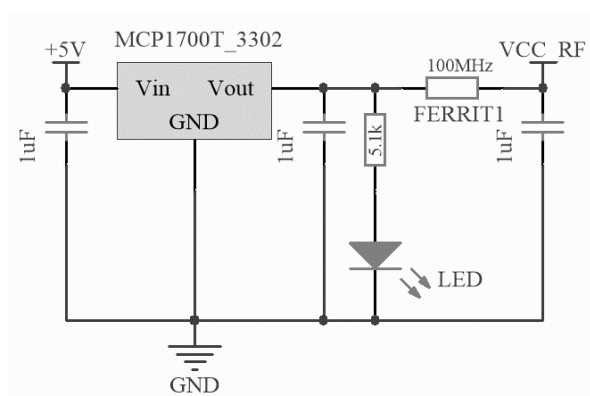




Slika 4.9: Vežje za referenčno frekvenco

### 4.3.6 Napajanje

Po navodilih proizvajalca je napajalne linije različnih delov čipa med seboj pametno ločiti. Seveda je za doseganje čim manjšega šuma primerna uporaba linearnih napetostnih regulatorjev. Sam sem izbral čip v monolitnem pakiranju MPC1700T-3302 podjetja Microchip [22]. Ločil sem napajanje analognega RF izhodnega dela, digitalne PLL interne logike, ter digitalnega dela, ki skrbi za komunikacijo preko SPI in signalizacijo uspešnega vklepa zanke (ang. Lock detect). Napetost RF izhodnega dela in PLL interne logike sem še dodatno filtriral z uporabo ferita v obliki SMD čipa. Z natančno optimizacijo dušenja motenj, se tu nisem ukvarjal. Prvi rezultati so namreč pokazali, da vezje deluje zanesljivo dobro tudi s takšno postavitvijo. Na sliki 4.10, je prikazana shema za enega izmed treh napajalnikov.



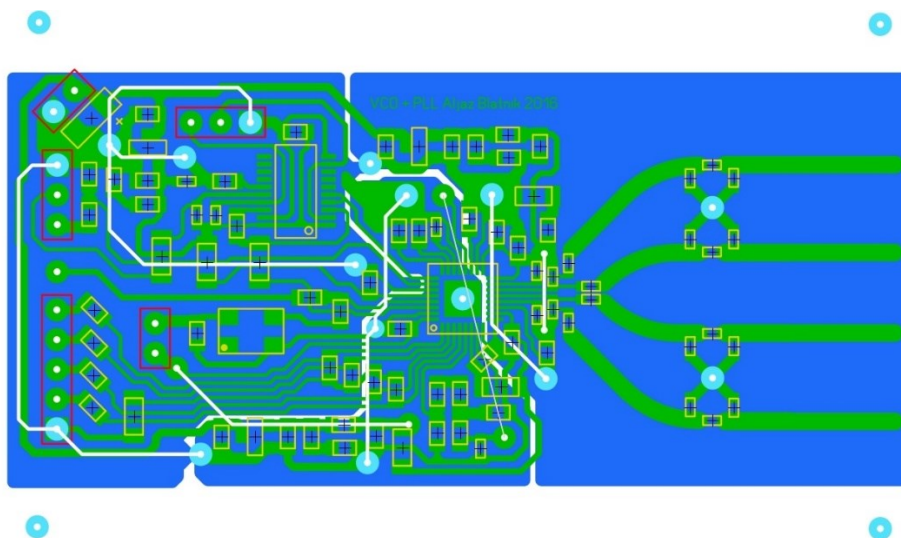
Slika 4.10: Vežje 3.3V napajalnika

Celotna shema vezja z PLL zanko se nahaja v Prilogi A.

## 4.4 Gradnja prototipa

### 4.4.1 Prvi prototip

Prvi prototip je bil zgrajen na dvoslojni tiskanini FR4 debelino 0.8mm. Pod QFN čip MAX2871 je bila zvrtna luknja dimenzije približno 2.5mm, baker ki je pokrival termalno površino je bil odstranjen z ostrim nožem. Spodnja stran tiskanine je bila jedkana tako, da je maso razdelila na tri otoke, s skupno vezavo v eni točki. Tak je tudi nasvet proizvajalca. Združen negativ tiskanine prikazuje slika 4.11.

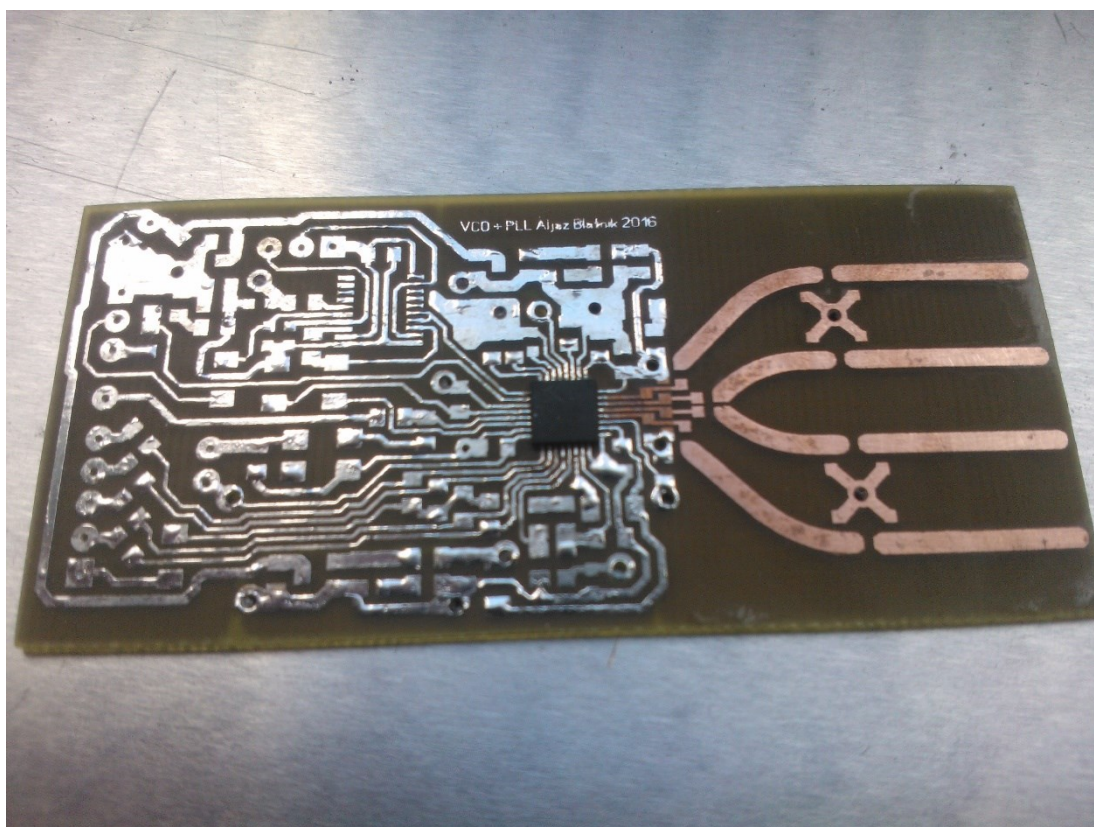


Slika 4.11: Negativ prvega prototipa

Vezju je ob risanju povezav bil dodan tudi čip 74LV07T, ki sem ga našel med staro šaro in ga dodal v vezje kot ločilno stopnjo med MAX-om in zunanjim svetom. Bal sem se namreč, da bi moja malomarnost povzročila kratek stik na izhodnih sponkah, kar bi uničilo več ur mojega dela. Modre pike izven tiskanine so luknje za poravnavo obeh negativov (zgornja in spodnja stran), kar nam močno poenostavi delo, ko lovimo položaja na obeh straneh skupaj.

Najprej izjedkamo zgornjo stran, spodnjo pa zaščitimo z lepilnim trakom. Nato izvrtamo vodilne luknje za določanje položaja spodnjega dela. Tu si pomagamo s sponkami za papir, saj lepo pašejo v luknje dimenzije 0.8mm. Sledi podoben postopek, le za spodnjo stran. Zvrtamo preostale luknje. Tiste, ki niso masa na spodnji strani rahlo povrtamo z nekoliko večjim svedrom, da preprečimo morebitni kratek stik z maso spodaj. Sledi nameščanje čipa MAX2871. Na vse QFN povezave damo nekoliko več cina, da se naredi lep hribček. Na tiskanino previdno položimo čip in ga grobo poravnamo s povezavami. Z vročim zrakom najprej počasi predgrejemo ploščico ter čip, pri tem pa dodajamo stearin, ki ga uporabljamo kot fluks. Nato se s šobo vročega

zraka približamo čipu na razdaljo med 1-2cm, ter počakamo da se cin začne taliti. Površinska napetost bo čip rahlo povlekla na prav položaj, a le, če je tiskanina izdelana brezhibno. S pinceto rahlo pritisnemo na čip, da izrinemo odvečen cin pod nogicami, nato šobo počasi odmikamo in pustimo, da se tiskanina pohladi. Na spodnji strani moramo zaliti še maso QFN čipa. S previdnim segrevanjem na spodnjo stran naneseemo dovolj cina, da ga lahko zalijemo z bakrom v okolici. Tu nujno potrebujemo večjo luknjo, saj površinska napetost cina pri luknjah pod 2 mm preprečuje zlivanje z bakrom v okolici. Končni rezultat je prikazan na sliki 4.12.



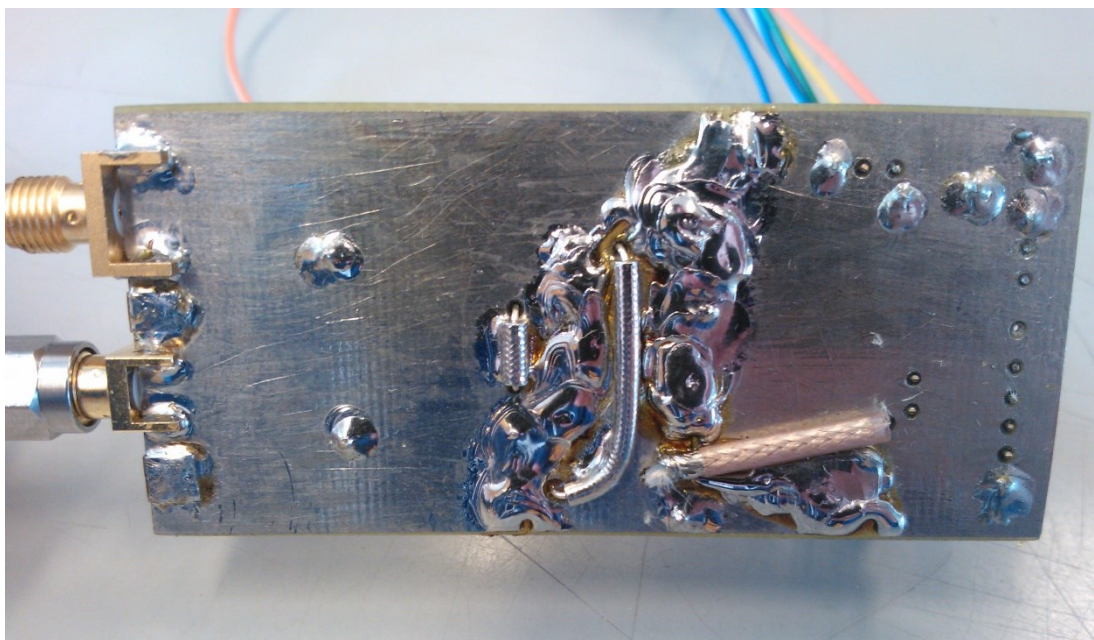
Slika 4.12: Spajkanje QFN čipa MAX2871 na tiskanino

Nato nadaljujemo z nameščanjem preostalih komponent. Na koncu s kosom koaksialnega kabla povežemo še izhod zančnega sita z vhodom v VCO, ki se nahajata na nasprotnih koncih tiskanine. Pri tem plašč kabla zacimimo na maso tiskanine.

#### 4.4.2 Drugi prototip

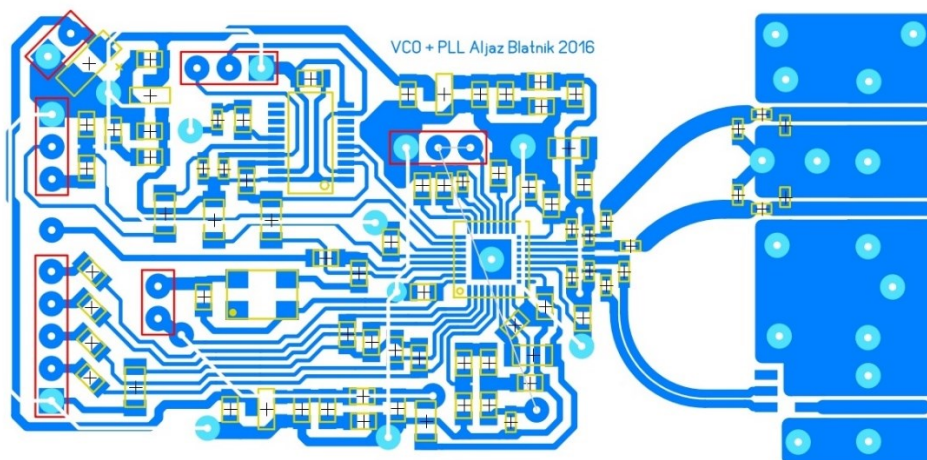
Ob prvih meritvah izdelanega vezja se je izkazalo, da ločevanje mas na spodnji strani pomeni vdor referenčnega signala v izhodni spekter. S preprostim zalivanjem vseh mas z obilico cina, je težava popolnoma izginila (slika 4.13). Enotna masa pomeni lažjo izdelavo tiskanine in celo boljše rezultate. Ne bom trdil, da industrijska

različica večslojnih tiskanin ne da boljših rezultatov z ločenimi masami, a v primeru izdelave tiskanine v domači delavnici jih zagotovo ne.



Slika 4.13: Zalivanje skupne mase

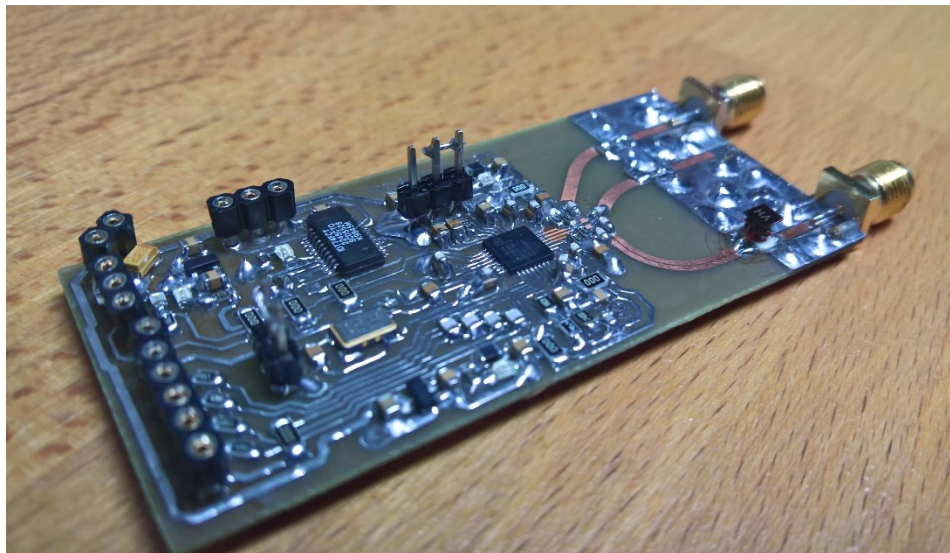
Izdelana je bila nova tiskanina, kjer je masa samo ena, dodal pa sem tudi možnost za nameščanje balun-a, ter nekoliko spremenil izhodne povezave, saj je bila pri prvem prototipu storjena napaka in izhod LD (lock detect) ni bil uporaben. Signal za  $V_{TUNE}$  (vhod v napetostno krmiljen oscilator) lahko sedaj izberemo s kratkostičnikom na tiskanini tako, da uporabimo izhod zančnega sita, maso, ali pa napetost pripeljemo od zunaj. Spremenjen negativ tiskanine prikazuje slika 4.14.



Slika 4.14: Negativ drugega prototipa



Izdelamo jo podobno kot tiskanino prvega prototipa, le spodnjo stran tokrat preprosto zaščitimo in je ne jedkamo. Končni izgled novega prototipa prikazuje slika 4.15.



Slika 4.15: Končni izgled drugega prototipa

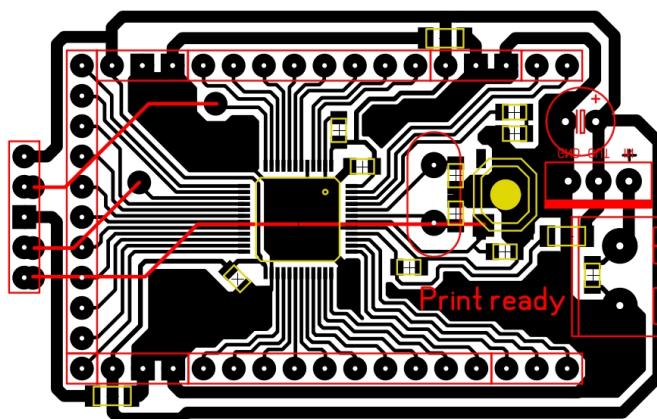


## 5 Podporni elementi merilnega sistema

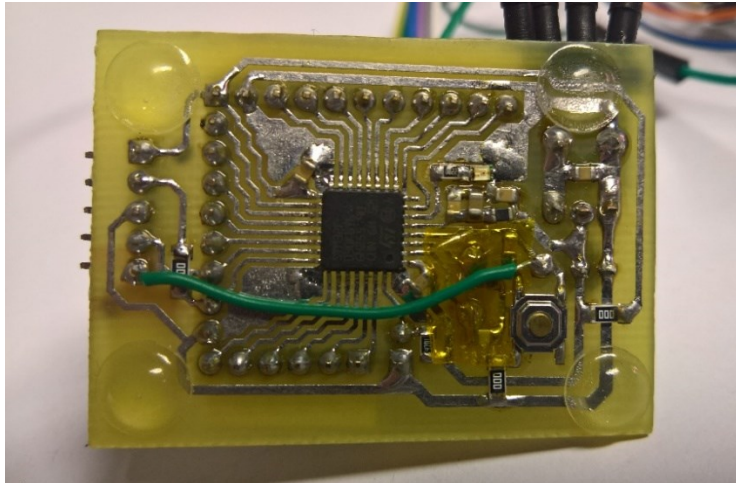
Za komunikacijo s PLL čipom potrebujemo vsaj mikrokrmilnik, da preko serijske SPI nastavlja pet registrov v notranjosti. Iz primerkov frekvenčnih izvorov domače izdelave, ki so jih izdelali drugi, sem hitro opustil idejo o nastavljanju registrov preko štirih ali več tipk. To je zame preveč preveč mukotrpno delo. Zato sem na izbranem ARM-u spisal program za preprosto ukazno vrstico.

### 5.1 Komunikacija s PLL čipom

V predalu je ležal neuporabljen ARM Cortex-M0 STM32F030K6. Ker sem v preteklosti že delal z njim, hkrati pa je več kot dovolj zmogljiv za dano nalogo, z odločitvijo o uporabi nisem okleval. Pripravil sem preprosto tiskanino, ki vse razen napajalnih pinov spelje na zunanje priključke, ima nameščen ustrezen kristal, ter napajalno vezje. Negativ tiskanine je prikazan na sliki 5.1, končni izdelek pa na sliki 5.2.



Slika 5.1: Negativ ARM tiskanine

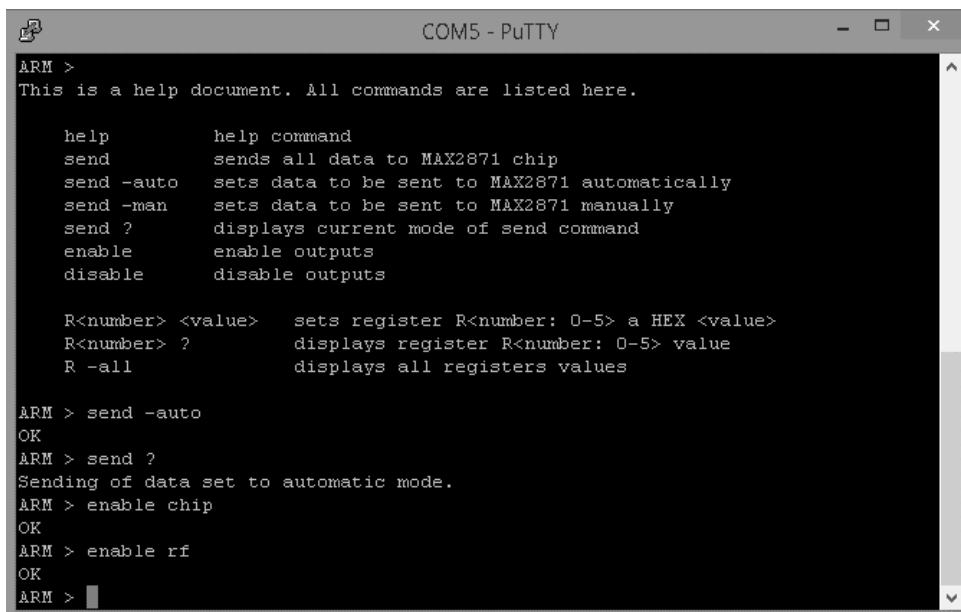


Slika 5.2: Končna slika ARM tiskanine

Pet priključkov uporabimo za upravljanje s PLL čipom, od tega so štirje za SPI komunikacijo, ločena povezava pa je namenjena vklapljanju in izklapljanju RF izhodov. Na dveh povezavah omogočimo UART serijsko povezavo, ki jo peljemo do USB pretvornika UART->USB, tega pa priključimo na računalnik, kjer tako dobimo navidezni serijski port.

Programska oprema na mikrokrmilniku omogoča komunikacijo in upravljanje preko terminalskega programa, kot je na primer brezplačni Putty. Omogoča pregledovanje in urejanje registrov, vrednosti pa lahko zapisujemo v binarni obliki. Vse to mikrokrmilnik prejema kot konstanten tok ASCII znakov, ki jih ob prejetju znaka za novo vrstico obdela, ter vrne odgovor. Hkrati nam ob pritisku tipke odgovori z odmevom (ang. remote echo), da nam teh nastavitev ni potrebno vklapljati v terminalskem programu. Izgled terminalskega okna in nekaj primerov ukazov prikazuje slika 5.3.





```
COM5 - PuTTY
ARM >
This is a help document. All commands are listed here.

help          help command
send          sends all data to MAX2871 chip
send -auto    sets data to be sent to MAX2871 automatically
send -man     sets data to be sent to MAX2871 manually
send ?       displays current mode of send command
enable        enable outputs
disable       disable outputs

R<number> <value>  sets register R<number: 0-5> a HEX <value>
R<number> ?       displays register R<number: 0-5> value
R -all        displays all registers values

ARM > send -auto
OK
ARM > send ?
Sending of data set to automatic mode.
ARM > enable chip
OK
ARM > enable rf
OK
ARM >
```

Slika 5.3: Terminalsko okno za komunikacijo s čipom ARM in upravljanje PLL čipa

Seveda lahko ukaze pošiljamo tudi preko drugega programa, na primer programskega jezika Python, ter tako avtomatiziramo meritev sistema v različnih načinih delovanja.

## 5.2 Samodejna meritev faznega šuma

Med izvajanjem prvih meritev sem ugotovil, da je ročna meritev faznega šuma zamudno delo, predvsem kako izvesti veliko število meritev za določanje prispevkov k faznemu šumu. Porodila se mi je zamisel o samodejni meritvi s pomočjo programskega jezika Python 3.5 ter USB->GPIB pretvornika.

Uporabljeni spektralni analizator Agilent E4445A sicer je povezljiv v omrežje Ethernet, a imam s tem slabe izkušnje. Zanesljivejša in varnejša se mi je zdela uporaba GPIB vmesnika, na kupu neuporabljenih kablov pa sem našel GPIB na USB pretvornik podjetja National Instruments [23]. Za Python seveda že obstaja izdelana knjižnica PyVISA [24], ki omogoča komunikacijo z inštrumenti preko različnih vodil, med drugim tudi preko GPIB. Za to potrebujemo nameščeno medgalaktično knjižnico National Instruments, katere instalacija traja dobro uro.

Skripta izvaja ukaze GPIB, ki jih najdemo v navodilih za uporabo spektralnega analizatorja [25]. Instrument najprej pravilno nastavimo. Ker ga uporabljajo različni ljudje prevzamemo, da je nastavljen popolnoma napačno, zato nastavimo referenčne nivoje, slabljenje, skalo, enote, linije za zapis itd. Ob pisanju kode, sem ugotovil da spektralni analizator omogoča precej funkcij, ki so skrite globoko v menijih. Tako na

primer lahko sprožimo izravnavo odziva filtrov, kalibracijo slabilcev, ali pa napravi povemo, da bomo merili fazni šum, zato naj pravilno optimizira svoje filtre. Začetni izsek kode prikazuje slika 5.4.

```

import visa
import time
import csv
rm = visa.ResourceManager()
inst = rm.open_resource('GPIBO::18::INSTR') # izbira naprave

# odpremo .csv datoteko
f = open("rezultat.csv", 'wt')
writer = csv.writer(f)
#writer.writerow(('kHz', 'dBc/Hz'))

# inicializacija naprave (pocistimo za 'jaz znam' budalami)
inst.write('INIT:CONT ON') # stalni sweep
inst.write('POW:ATT 20') # atenuacija 20 dB
inst.write('DISP:WIND:TRAC:Y:RLEV 10 dbm') # nastavimo ref. nivo
inst.write('POW:ATT:AUTO OFF')
inst.write('DISP:WIND:TRAC:Y:SPAC LOG') # logaritmicna skala
inst.write('DISP:WIND:TRAC:Y:PDIV 10 DB') # 10dB/div
inst.write('UNIT:POW DBM') # amplitudo v dBm
inst.write('POW:GAIN OFF') # brez ojačevalnika
inst.write('DISP:WIND:TRAC:Y:RLEV:OFFS 0.0') # offset 0.0dB
inst.write('CORR:CSET:ALL OFF') # izklopi vse morebitne korekcije
inst.write('POW:ATT:STEP 10') # korak atenuacije 10 dB
inst.write('SWE:TYPE SWE') # sweep type je nastavljen na sweep

```

Slika 5.4: Izsek kode v Pythonu

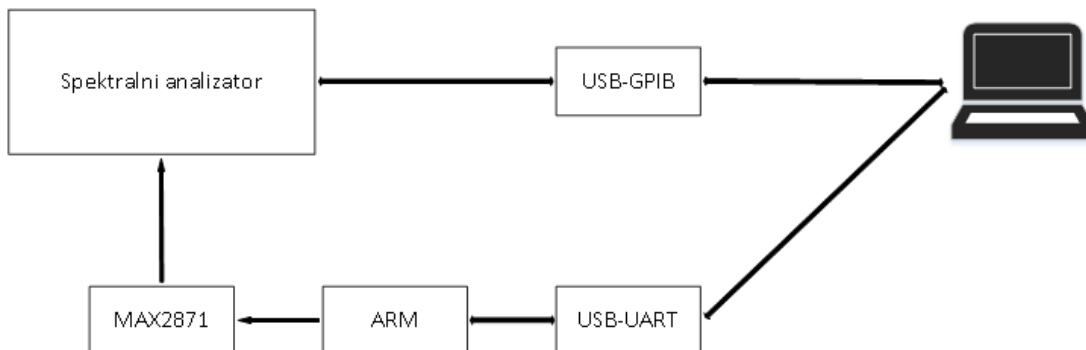
Program najprej opravi vso začetno kalibracijo, nato poišče največjo vrednost, jo postavi na sredino zaslona ter prične z meritvijo. Fazni šum meri v štirih korakih za različni frekvenčni odmik, kar pomeni različno pasovno širino sita, frekvenčni prelet, ter čas meritve. Vrednosti sproti pravilno preračunava, ter jih shranjuje v izhodno .csv datoteko. Ena meritev faznega šuma traja približno minuto. Datoteko lahko potem uporabimo za izris grafov.

Poleg programa za meritev šuma pri eni frekvenci, se le z majhnimi spremembami v programski kodi omogoči še meritev lastnega faznega šuma, ter samodejno meritev faznega šuma v celotnem frekvenčnem območju delovanja PLL zanke. Rezultati meritev se sproti shranjujejo v .csv datoteko.

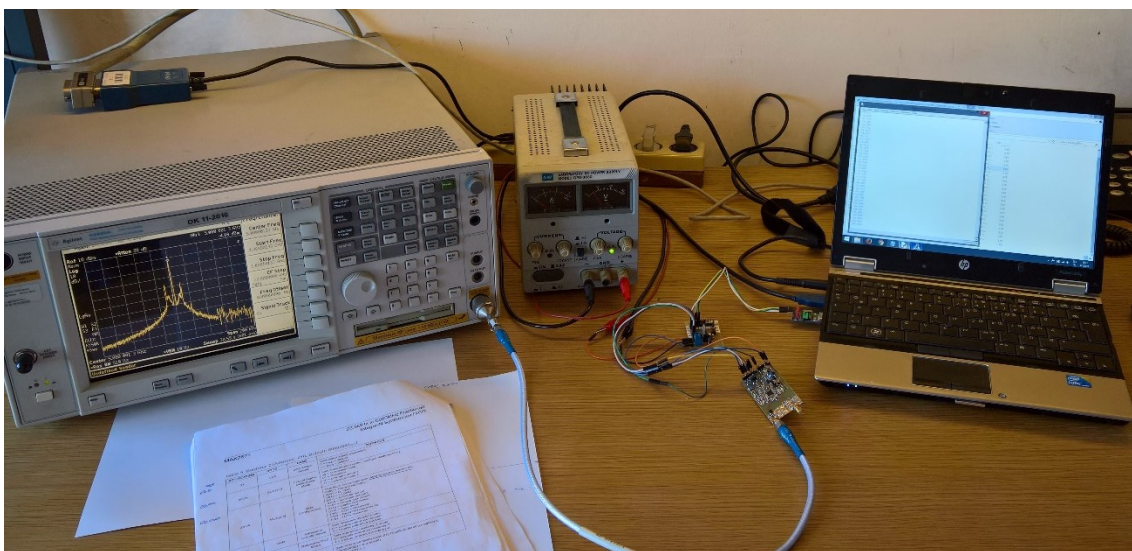
### 5.3 Postavitev merilnega sistema

Celoten merilni sistem tako vsebuje merjenec (PLL zanko s čipom MAX2871), ARM mikrokontroler za komunikacijo z osebnim računalnikom, stabiliziran

napajalnik, spektralni analizator, USB->GPIB pretvornik, USB->UART pretvornik, osebni računalnik, ter kable za ustrezne povezave. Postavitev instrumentov prikazuje slika 5.5, fotografijo izvedbe meritev pa slika 5.6.



Slika 5.5: Postavitev instrumentov



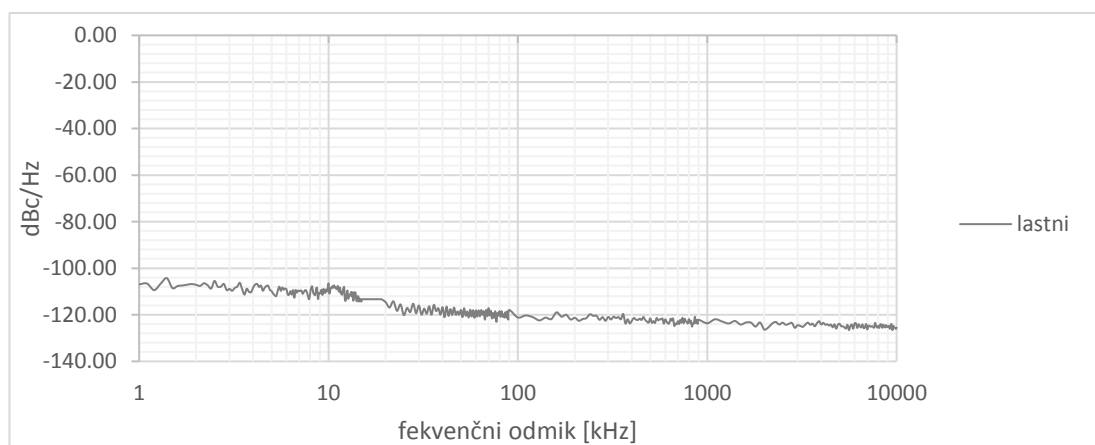
Slika 5.6: Fotografija izvedbe meritev



## 6 Rezultati meritev

Vse meritve so bile izvedene pri sobni temperaturi 23°C, z uporabo spektralnega analizatorja Agilent E44445A. Meritve so v poglavjih komentirane sproti.

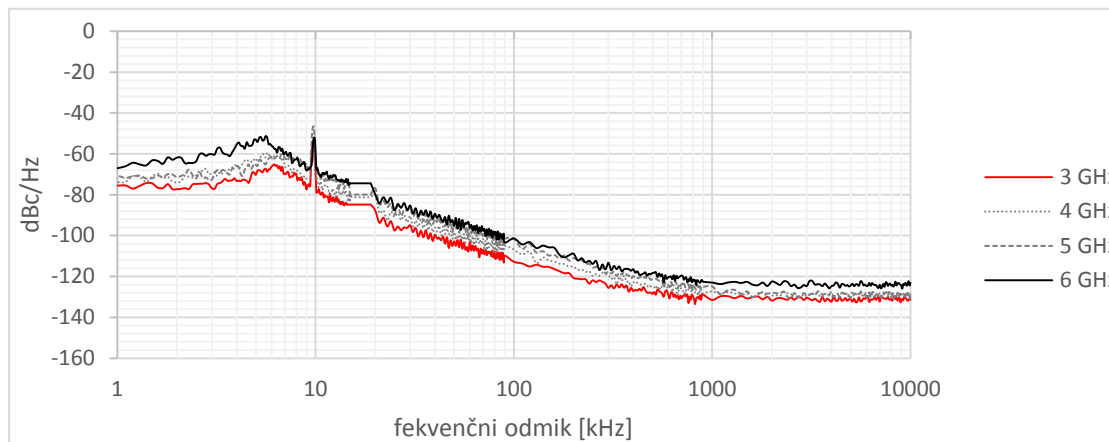
### 6.1 Lastni fazni šum spektralnega analizatorja



Slika 6.1: Lastni fazni šum

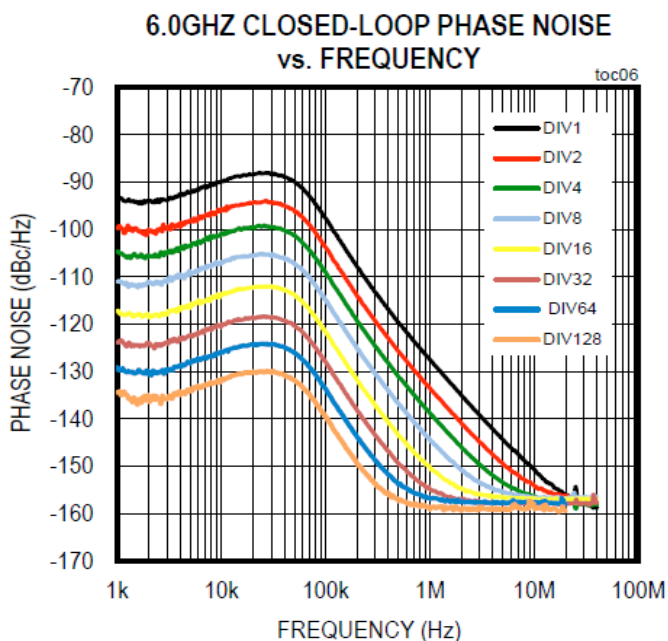
Z vidika faznega šuma spektralni analizator E4445A zagotovo ni najboljša naprava, ki jo lahko dobimo na tržišču, a je za meritev faznega šuma MAX2871, ki ga pričakujemo okoli -90dBc/Hz na odmiku 10kHz, zadostna.

## 6.2 Fazni šum na območju 3 - 6 GHz



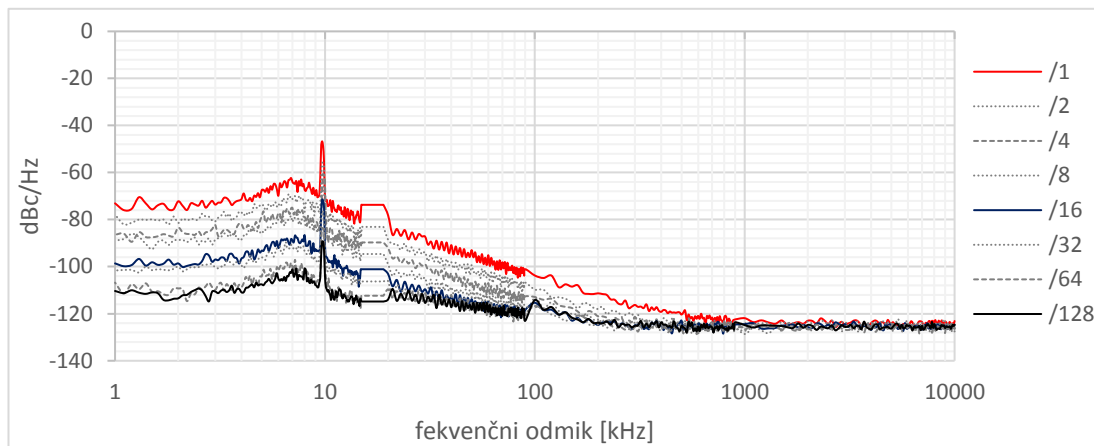
Slika 6.2: Fazni šum na območju 3 - 6 GHz

Fazni šum z višanjem frekvence pričakovano raste in doseže najvišjo vrednost okoli  $-56$  dBc/Hz pri frekvenčnem odmiku blizu 5 kHz. Tu se opazi odstopanje od navedb proizvajalca. Slika 6.3 prikazuje meritve faznega šuma s strani proizvajalca, ki jih navaja v podatkovnem listu. Pri tem je potrebno poudariti, da sam za frekvenco primerjalnika faze uporabljam 20 MHz, med tem ko je proizvajalec za takt uporabljal 50 MHz, ki jih je pripeljal do čipa preko zunanjega vira. Ker temperaturno stabilni oscilatorji delujejo pri svojih osnovnih frekvencah, jih za 50 MHz ne moremo dobiti. Zato se koleno preloma faznega šuma v meritvah ne ujema z navedbami proizvajalca.

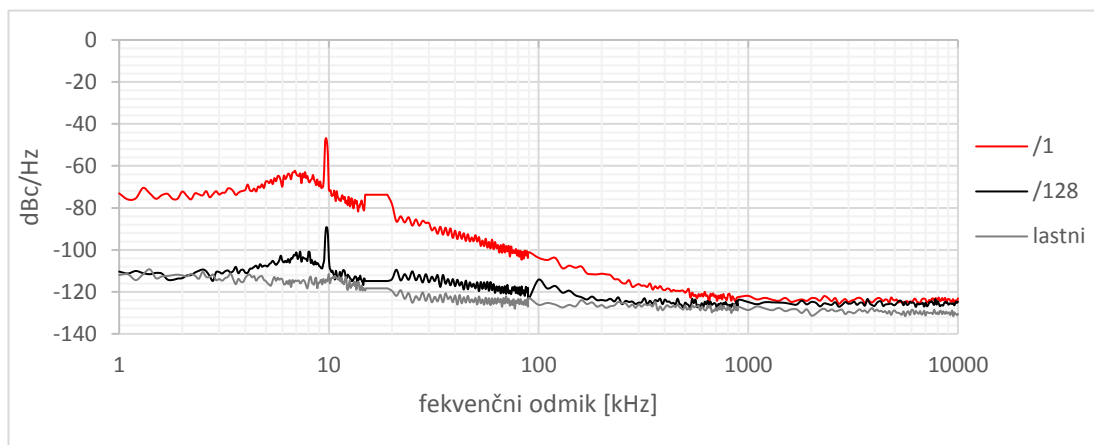


Slika 6.3: Meritev faznega šuma pri 6 GHz s strani proizvajalca

### 6.3 Vpliv delilnikov na fazni šum



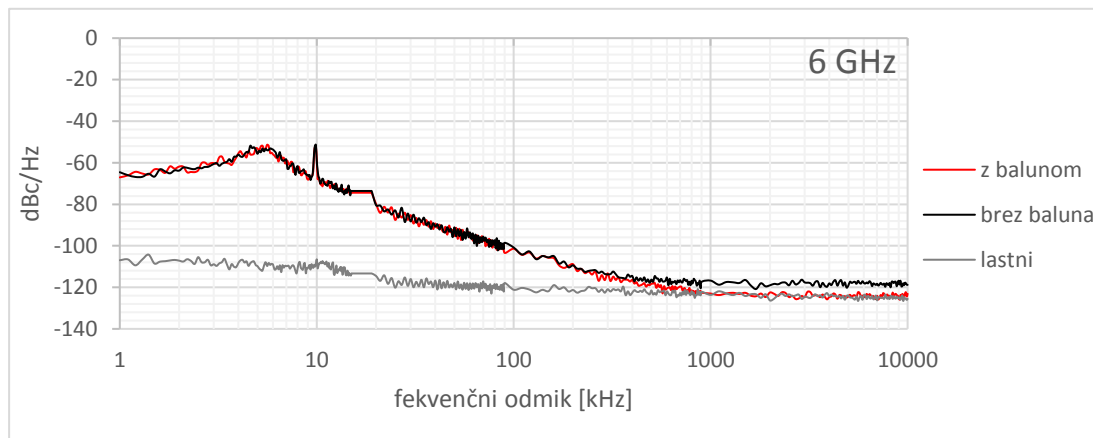
Slika 6.4: Vpliv delilnikov na fazni šum



Slika 6.5: Primerjava dveh skrajnih vrednosti pri vplivu delilnikov na fazni šum

Vpliv je seveda pričakovan, ko se premikamo k nižjim frekvencam, se fazni šum manjša, kar je vidno tudi iz meritev proizvajalca (slika 6.3). Raven odsek pri meritvah okoli frekvenčnega odmika 10.8 kHz je napaka programa za merjenje, ki je bila odkrita šele kasneje.

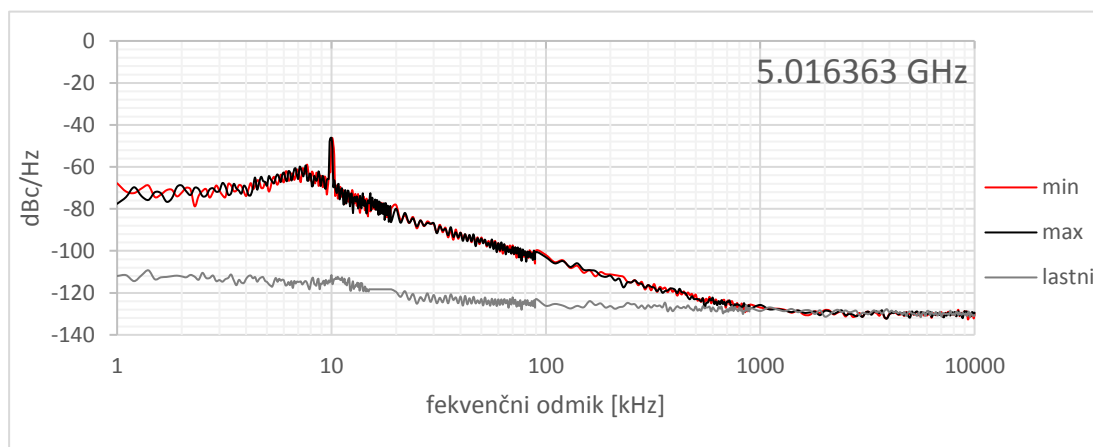
## 6.4 Vpliv baluna na fazni šum



Slika 6.6: Vpliv baluna na fazni šum

Iz slike 6.6 vidimo, da je vpliv baluna prisoten šele pri velikih frekvenčnih odmikih (nad 1 MHz), pri manjših odmikih pa se meritvi ne razlikujeta med sabo.

## 6.5 Vpliv toka črpalke naboja na fazni šum

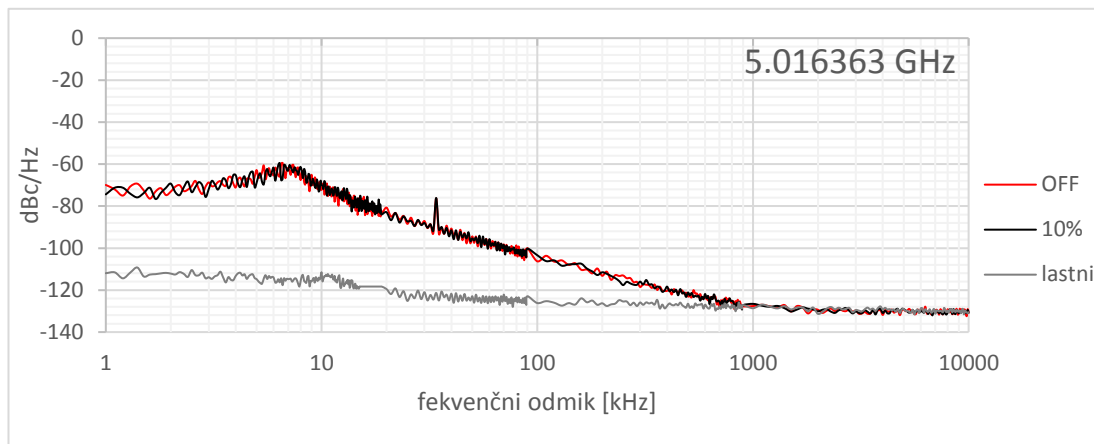


Slika 6.7: Vpliv toka črpalke naboja na fazni šum

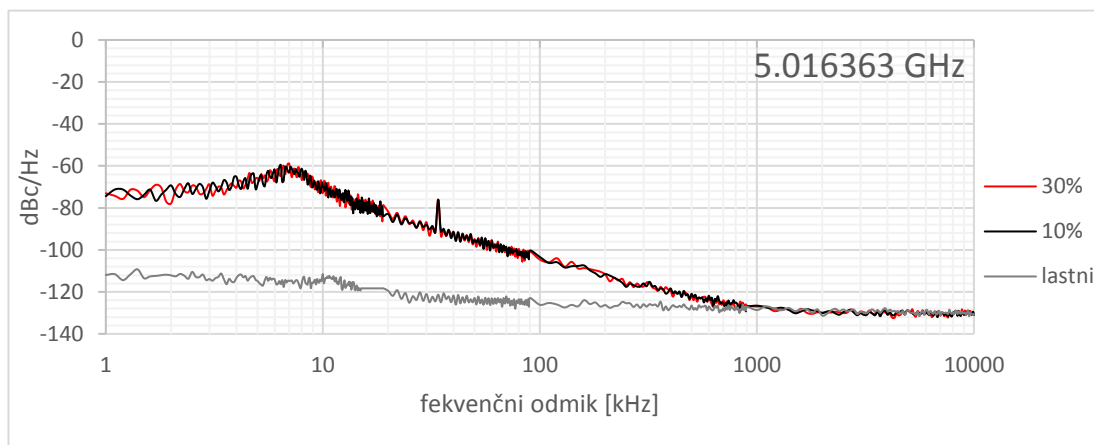
Minimalni tok v mojem primeru znaša 0.32 mA, maksimalni tok pa 5 mA. Izmerjene razlike v faznem šumu ni.



## 6.6 Vpliv linearnosti črpalke naboja na fazni šum



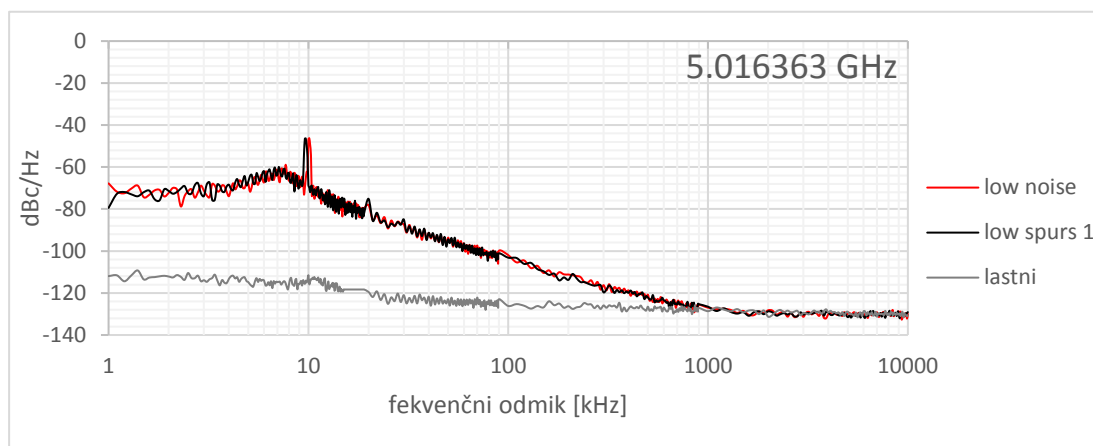
Slika 6.8: Vpliv 10% linearnosti črpalke naboja na fazni šum



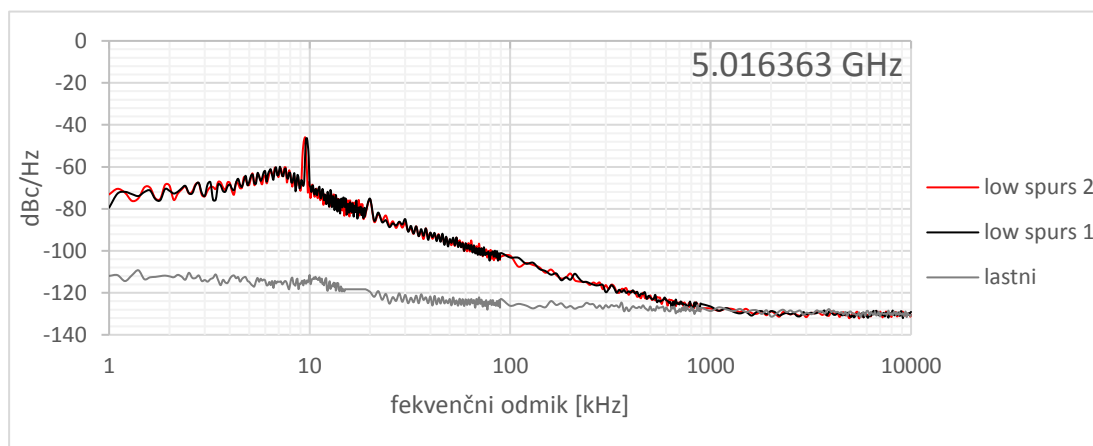
Slika 6.9: Vpliv 30% linearnosti črpalke naboja na fazni šum

Izbira procenta linearnosti naj bi vplivala na zmanjševanje nadležnih špičk pri ulomkovem načinu, izbira procenta pa je prepuščena uporabniku. Kako, v navodilih proizvajalca ni omenjeno. Na fazni šum nastavljanje linearnosti nima vpliva.

## 6.7 Primerjava različnih režimov delovanja faznega detektorja na fazni šum



Slika 6.10: Primerjava načina za nizek fazni šum in nizek nivo špičk 1

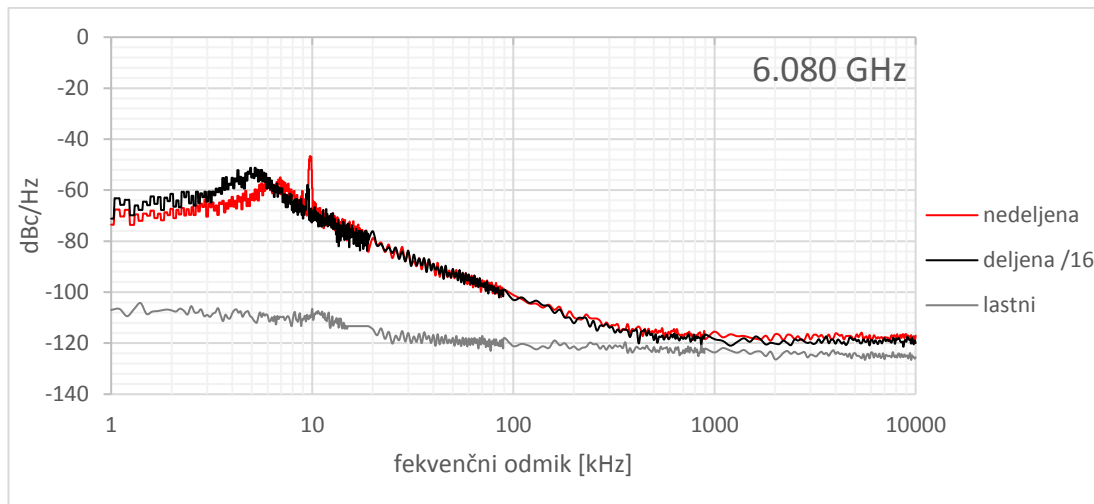


Slika 6.11: Primerjava načina za nizek nivo špičk 1 in 2

MAX2871 ponuja tri različne načine delovanja in sicer način za doseganje najnižjega faznega šuma, ter dva načina za nizke špičke ulomkovnega režima delovanja. Na podlagi meritev je razvidno, da na fazni šum nimajo vpliva.

## 6.8 Vpliv deljenje frekvence pred N števcem na fazni šum

Izbrani PLL čip omogoča, da frekvenco delimo za izbran modulo med 1 in 16, preden jo posredujemo v števec N. To celoštevilsko deljenje je drugačno od tistega, ki se nahaja v N števcu (izvedeno kot  $N/N+1$ ).

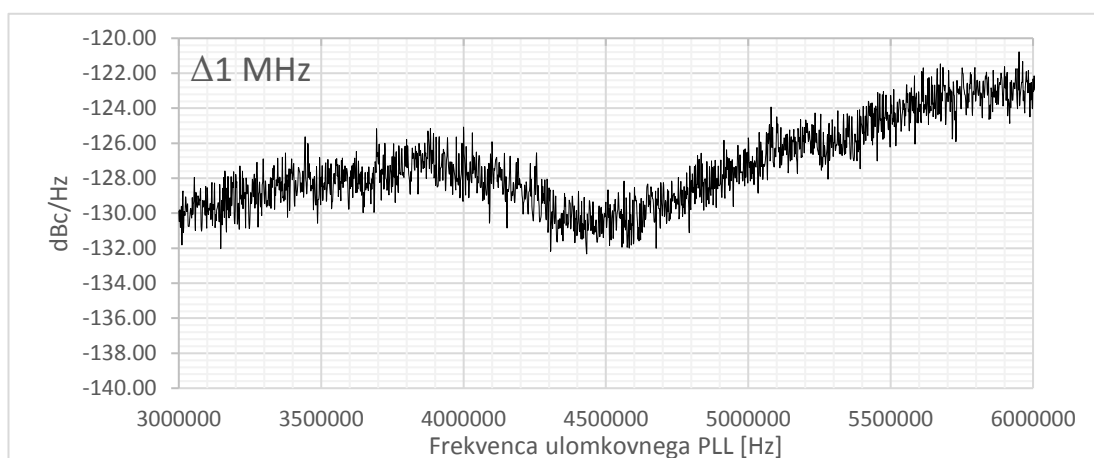


Slika 6.12: Vpliv deljenja frekvence pred N števcem na fazni šum

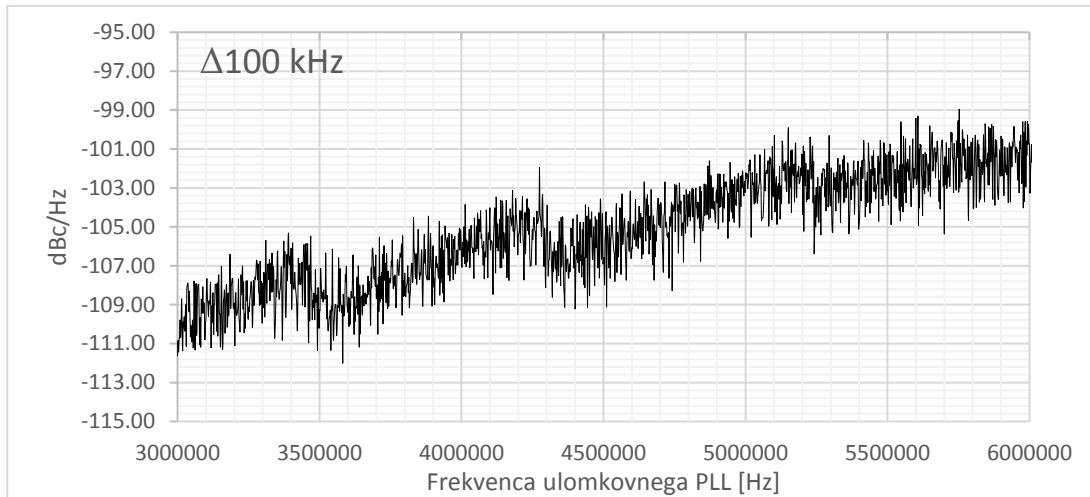
Iz meritev na sliki 6.12 vidimo, da deljenje slabo vpliva na fazni šum pri frekvenčnih odmikih pod 8 kHz, nad tem območjem pa sta fazna šuma primerljiva. Seveda vrednost števca N nastavimo tako, da na izhodu vedno dobimo frekvenco 6.080 GHz.

## 6.9 Fazni šum na celotnem območju od 3 do 6 GHz

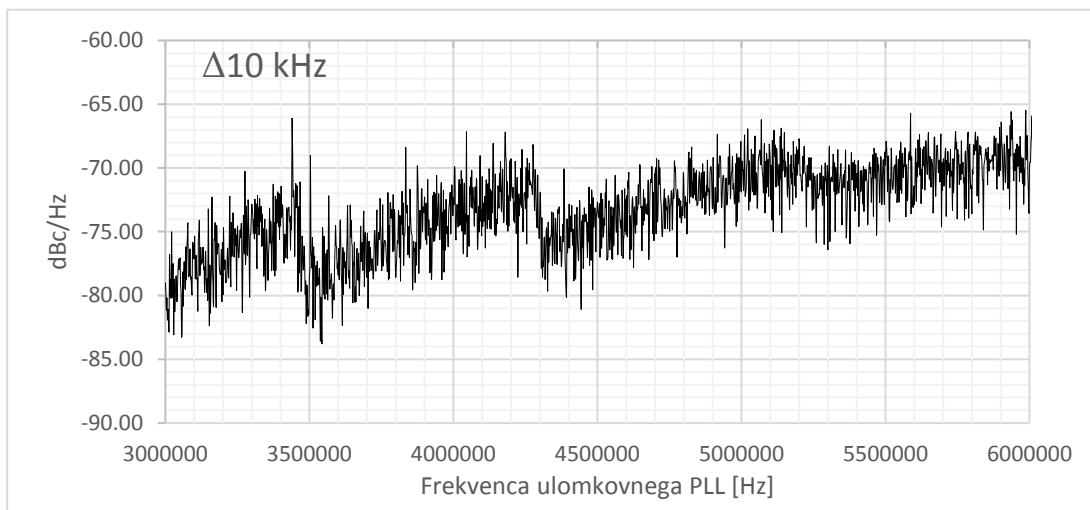
Za konec je zanimiva še meritev faznega šuma, ki ga zmerimo skozi celotno območje delovanja PLL čipa, brez deljenja reference, kar v našem primeru znaša od 3 do 6 GHz. Za meritev sem izbral ulomek z imenovalcem 11 (praštevilo) dovolj nizkim, da je bila celotna samodejna meritev opravljena v 24 urah. Slike od 6.13 do 6.15 prikazujejo spreminjanje faznega šuma skozi izbrano frekvenčno področje pri določenem frekvenčnem odmiku od nosilca.



Slika 6.13: Fazni šum na odmiku 1 MHz



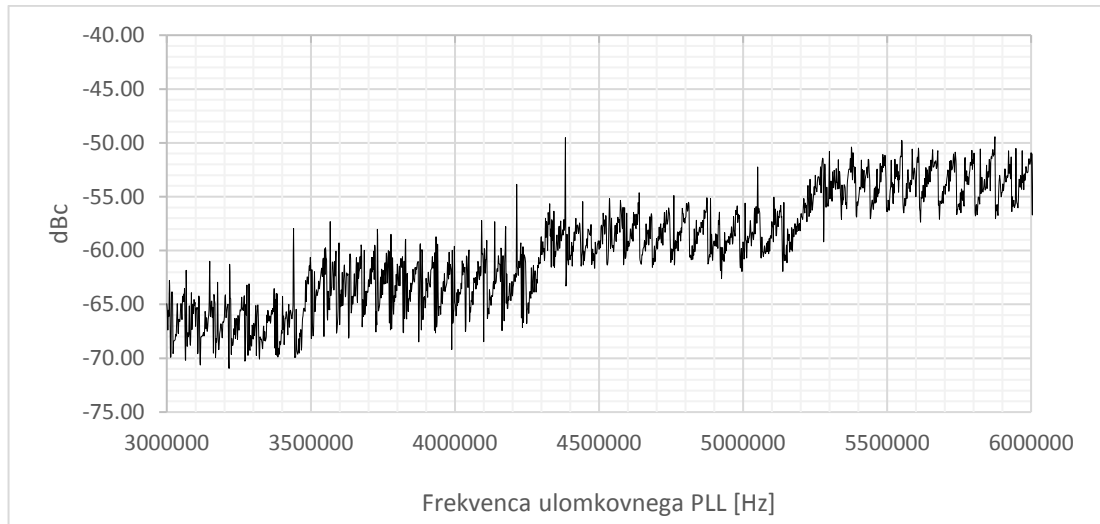
Slika 6.14: Fazni šum na odmiku 100 kHz



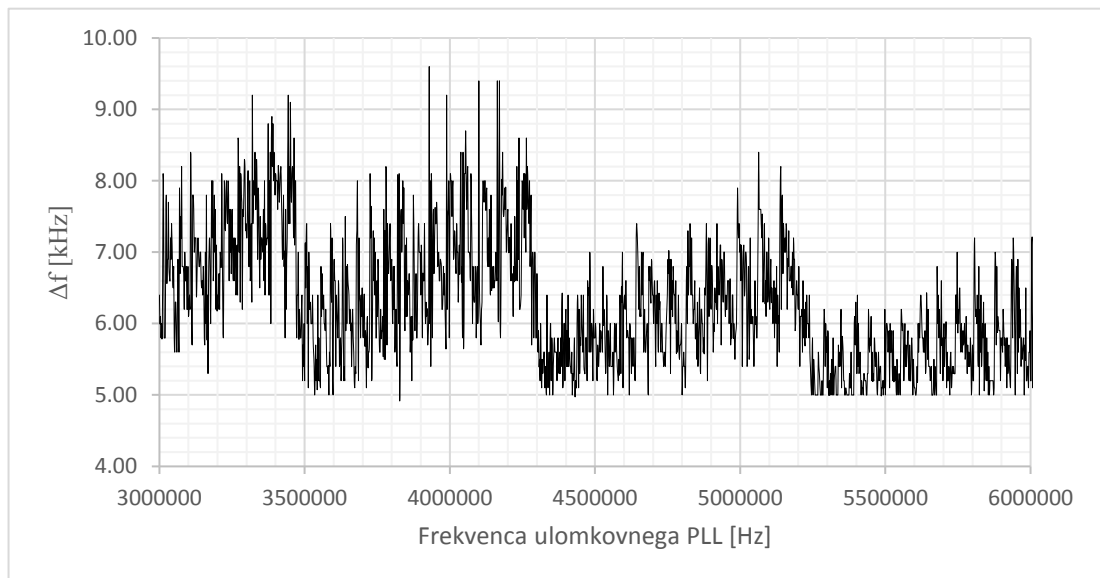
Slika 6.15: Fazni šum na odmiku 10 kHz

Če je pri frekvenčnem odmiku 1 MHz še čutiti vpliv merilnega instrumenta (nahajamo se ravno na meji termičnega šuma), je pri odmikih 100 kHz in 10 kHz moč videti štiri področja, kjer se fazni šum naglo zniža ter nato zopet narašča.

Zanimiva je tudi meritev najvišje vrednosti faznega šuma, to je navadno tam kjer se nahaja nadležna špička, ki je produkt ulomkovnega režima delovanja. Vrednost je prikazana na sliki 6.16, frekvenčni odmik od nosilca, kjer se to zgodi pa na sliki 6.17.



Slika 6.16: Najvišja vrednost faznega šuma v frekvenčnem območju od 3 do 6 GHz



Slika 6.17: Odmik frekvence, pri kateri dosežemo največji fazni šum

Seveda je špička frekvenčno izračunljiva, saj je odvisna od modula  $N$  delilnika ter vrednosti ulomka, nekoliko manj je določljiva moč, kar preverimo z meritvijo.

## 6.10 Komentar meritev

Najbolj zagotovo izstopa preprosta meritev faznega šuma, saj se razlikuje od navedb proizvajalca. Drži, štiri slojne tiskanine nisem izdelal, ravno tako si nisem mogel privoščiti nešteti vij, ki jih proizvede računalniško vodeni vrtalnik. Tudi pri napetostnem regulatorju bi lahko izbral dražji ekvivalent, ki bi vnašal manjši šum, tiskanino primerno oklopil, povečal primerjalno frekvenco.

Podlaga za vse meritve je bilo pravzaprav ugotoviti, ali so takšni ceneni čipi primerni za gradnjo spektralnega analizatorja, odgovor na to pa se je pokazal že veliko prej, kot so to pokazale meritve faznega šuma. MAX2871 in njemu podobni čipi namreč ne vsebujejo mehanizma za generiranje žage, oziroma načina za avtomatski prelet celotnega spektra. Vrednosti mu moramo v njegovo notranjost konstantno vpisovati. Pri tem se na žalost v celotnem spektru vidi digitalni takt, oziroma ukaz, ko mikrokontroler PLL čipu sporoči, da posodobi svoje registre. Takrat se zgodi preklon, signal za hip izgine, v spektru vidimo špičke reference, nato se signal zopet pojavi ter vklene na zeleno frekvenco. Tak pojav ni želen, težko se ga je tudi znebiti. Zato njegova uporaba kot prvi lokalni oscilator v spektralnem analizatorju ne pride v poštev. VCO in PLL čip moramo med seboj ločiti, zaželeno pa je tudi, da uporabimo samo en VCO in ne več njih.

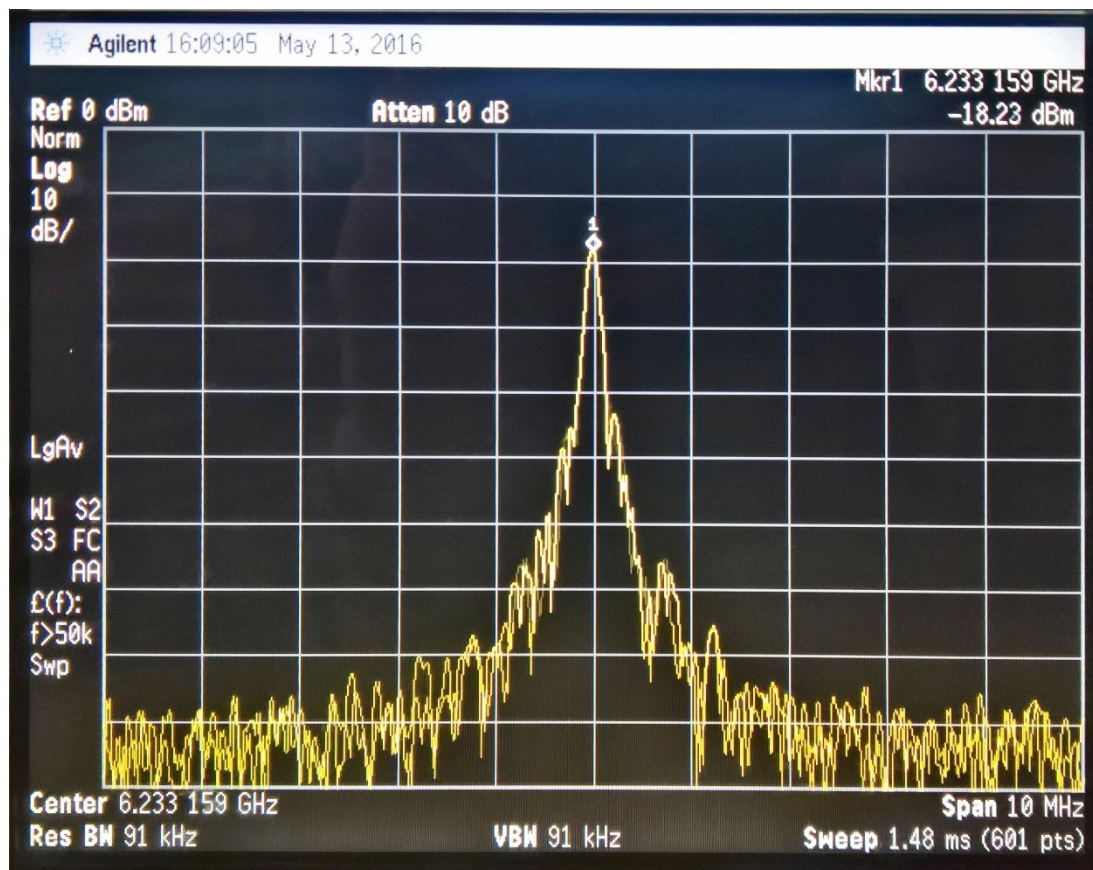
## 7 Gradnja lastne PLL zanke s FPGA

Med pregledovanjem načinov za krmiljenje PLL zanke, sem naletel na spletno stran Angleža, ki je s pomočjo široko dostopnega FPGA čipa sestavil lastno PLL zanko s, po njegovih besedah, odličnim faznim šumom [26]. Žal zanka deluje pri frekvencah pod 100 MHz, kjer je fazni šum manj kritičen, ravno tako je za meritev uporabil spektralni analizator, ki že sam ne dosega dobrega faznega šuma. Za višje frekvence je v času nastajanja tega dela obstajala težava pri pridobivanju delilnika frekvence, saj je bilo potrebno naročilo oddati v Združene države Amerike, plačati visoko ceno čipa, poleg tega pa še poštnino in carino. Za igranje in financiranje iz lastnega žepa se mi je cena zdela previsoka.

Težavo reši MAX2871, saj lahko v svoji notranjosti deli frekvenco in še vsebuje napetostno nastavljiv oscilator. Tako močno omilimo stroške prototipa. Zamisel je bila, da z implementacijo PLL zanke v FPGA čipu preverim še vpliv matematike za podrtavanje modula števca  $N$ , v odvisnosti od stopnje matematičnih funkcij (koliko stopenj je vezanih v verigo za izračun).

Ker je za FPGA še težje izdelati dobro tiskanino, ker drobne povezave res ne moremo med seboj križati s kratkostičniki, in ker težko dobimo čip z manj kot 144 nogicami, sem na spletu poiskal najcenejšo možno razvojno ploščico, ki je bila zadosti za moje potrebe. Izdelal sem zunanjo črpalko naboja, predelal obstoječo tiskanino MAX2871 tako, da za krmiljenje VCO uporabljam zunanjo napetost, enega izmed visokofrekvenčnih izhodov predelal v diferencialnega ter ga peljal na diferencialni vhod FPGA, namestil dodatne blokirne kondenzatorje na vse napajalne linije FPGA, se naučil Verilog za njegovo programiranje, izklopil pamet MAX-a, tako da ne prevzema nadzora nad VCO ter povezal nedeljen izhod na spektralni analizator.

Spekter izhodnega signala prikazuje slika 7.1.



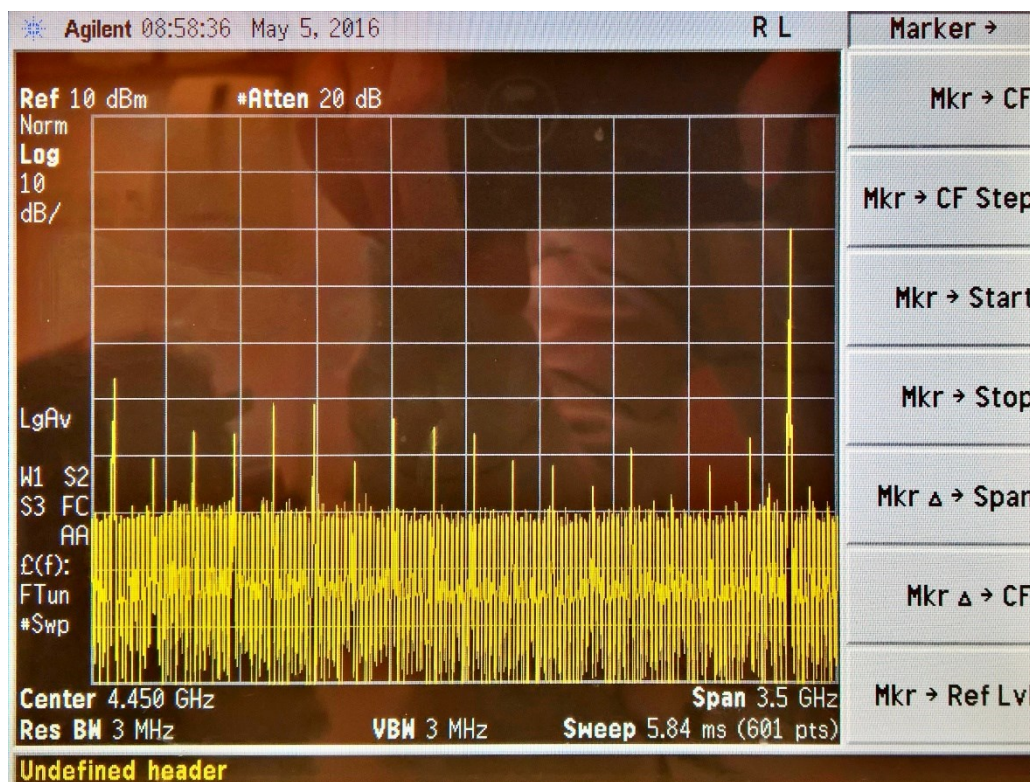
Slika 7.1: Spekter izhodnega signala z implementacija lastne PLL zanke

Zanka je pravilno vklenjena na referenčni signal, vendar so na signalu vidne neželene špičke. Če sem popolnoma iskren, FPGA ni namenjen za uporabo na takšen način. Načrtovalci so v glavah imeli idejo o zamenjavi procesorja, ne o zamenjavi PLL čipa. Da končna izvedba ne bo najboljša, sem vedel še preden sem se lotil gradnje, a sem vseeno poskusil. Veliko nesnago je povzročila prvotno napačno načrtovana črpalka nabojev, ki je vsebovala konstanten tokovni vir z uporabo počasnih operacijskih ojačevalnikov. Gradnja z uporabo preprostega upora je veliko enostavnejša, proizvede pa tudi precej boljši rezultat.

Drug problem, ki se pojavi, je presluh deljene frekvence, ki nastane že v čipu MAX2871. To povzroči malo morje špičk v spektru, kar dela frekvenčni generator že sam po sebi neuporaben. Zaslonski zajem spektra prikazuje slika 7.2.

Seveda je tu meritev faznega šuma, ki bi preveril vpliv matematike višjega reda nesmiselna. Drugi izvori šuma imajo namreč prevelik vpliv nanj.

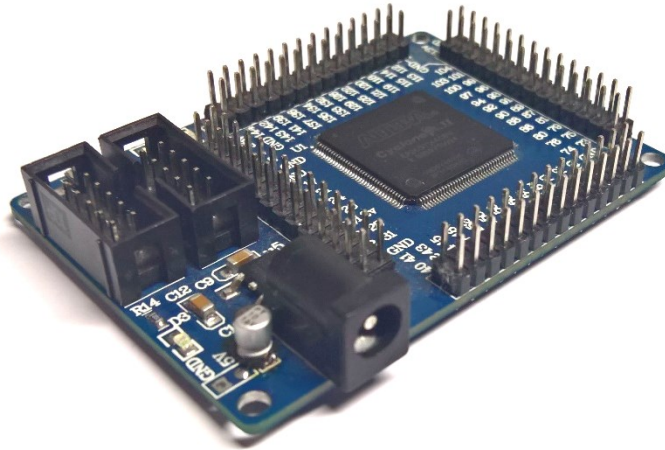




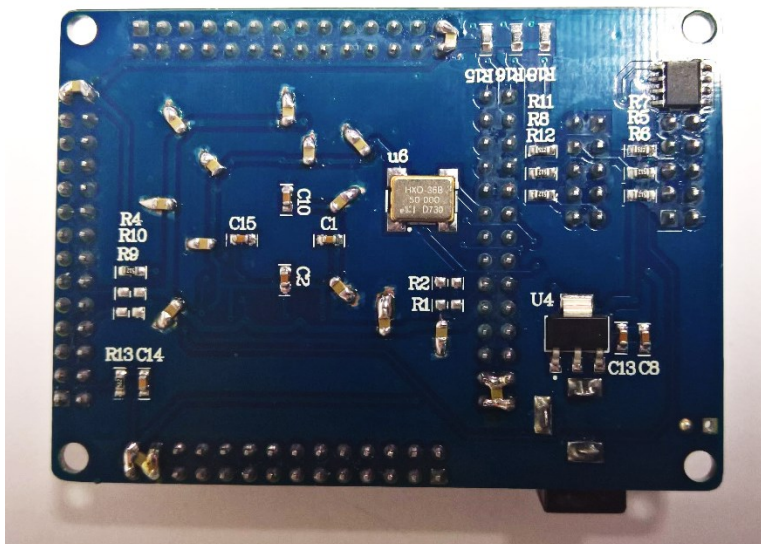
Slika 7.2: Spekter špičk v uporabnem območju

## 7.1 Altera EP2C5T144C8 Cyclone II FPGA

Razvojne ploščice za FPGA čip EP2C5T144C8 so na spletu lahko dostopne, skupaj s poštnino nekaj čez 10 €, kar je celo ceneje, kot če bi čip kupili samostojno [27]. Cyclone II je že precej stara družina FPGA čipov podjetja Intel, ki z novimi razvojnimi okolji niti ni več podprta. Zadnja različica ki ga tako podpira, je Quartus II 13.0 SP1, ki pa je še vedno popolno uporabna. Tiskanino, ki jo navadno dobimo s Kitajske prikazuje slika 7.3. Ker me je skrbelo varčevanje proizvajalca s kondenzatorji, sem na spodnji strani napajalnim linijam dodal še nekaj kondenzatorjev 100 nF, kar je prikazano na sliki 7.4. Za napajanje ploščica zahteva +5V vhodne napetosti, za ostale zahtevane napetosti FPGA pa so linearni regulatorji že nameščeni na tiskanini. Dva priključka za JTAG povezavo omogočata nalaganje konfiguracije neposredno v FPGA ali pa v zato namenski serijski čip.



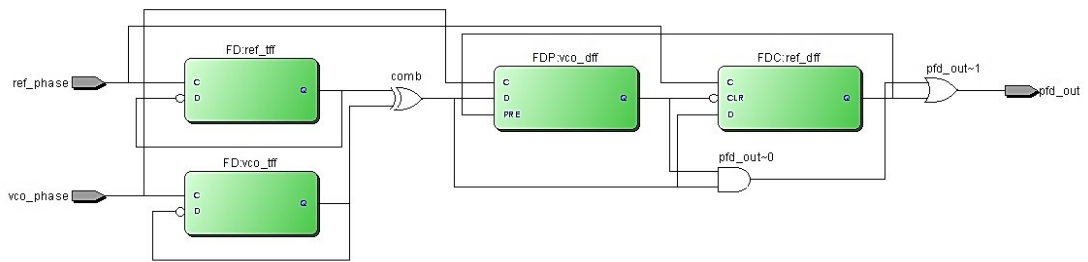
Slika 7.3: EP2C5T144C8



Slika 7.4: Dodajanje blokirnih kondenzatorjev na napajalne linije

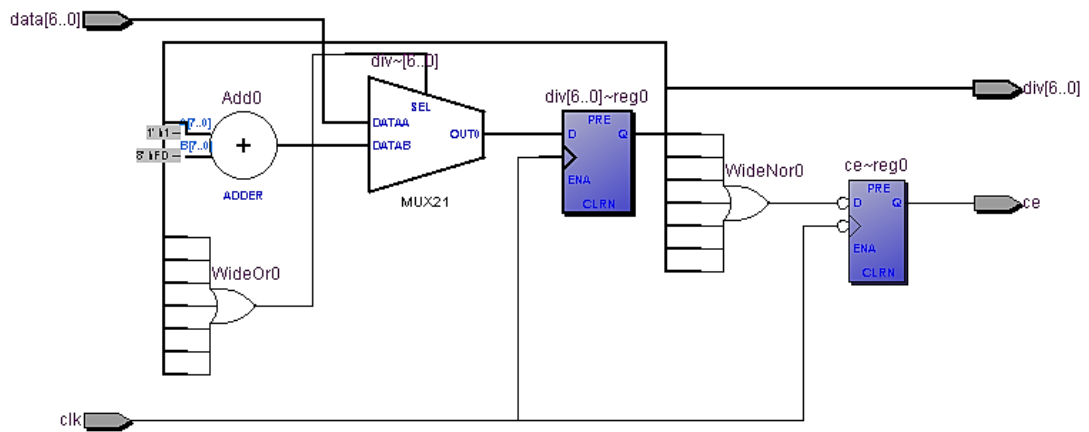
### 7.1.1 Programska koda

V FPGA sem najprej implementiral strukturo faznega detektorja na podlagi čipa AD9901 [9]. Pri tem sem spisal module, ki omogočajo delo z logičnimi vrati, da je struktura identična tisti v podatkovnem listu čipa AD9901. Strukturo faznega detektorja implementiranega v FPGA na prikazuje slika 7.5.

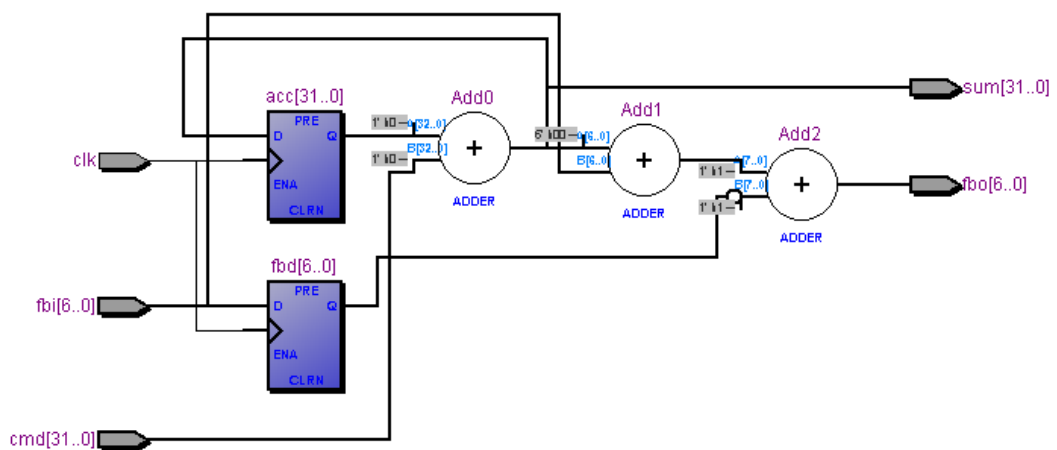


Slika 7.5: Implementacija faznega detektorja v FPGA

Za deljenje spišemo svoj modul ki sprejme uro in vrednost delitelja, vrne pa deljeno uro, ter trenutno vrednost internega števca. Struktura kot jo vidi FPGA je prikazana na sliki 7.6.

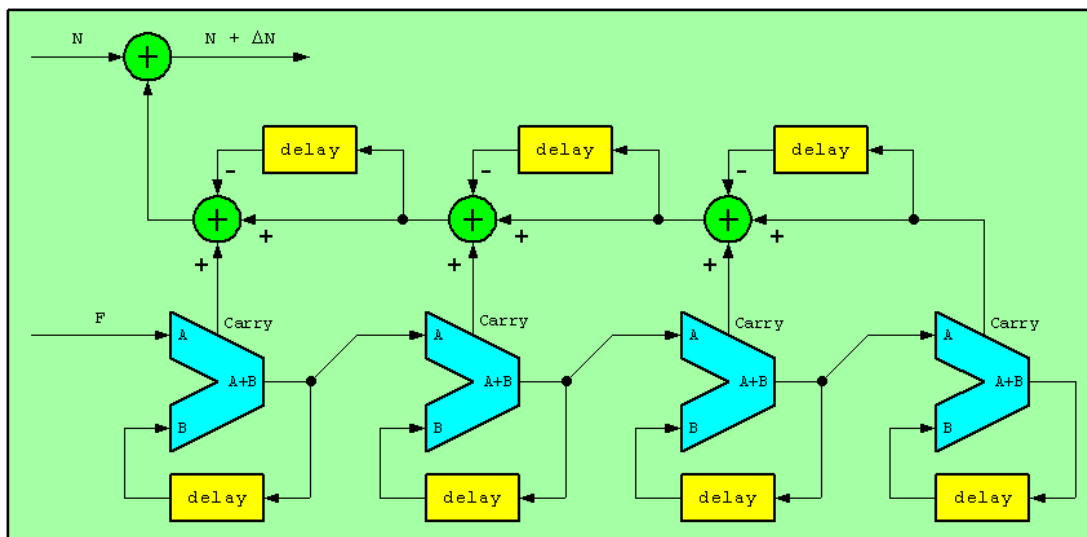


Slika 7.6: Struktura modula za deljenje



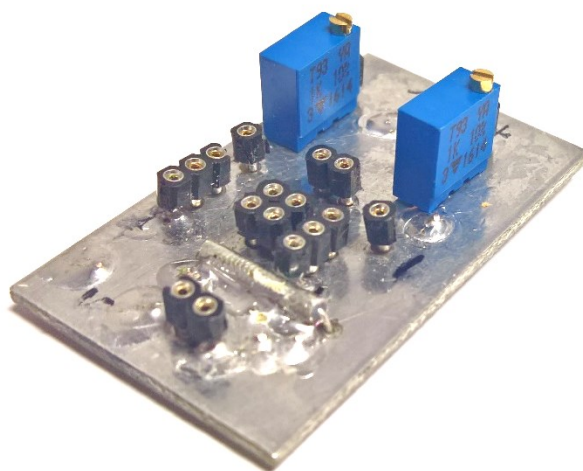
Slika 7.7: Implementacija MASH stopnje v FPGA

Zadnji pomemben sestavi del ulomkovne PLL zanke je MASH stopnja, ki na podlagi vrednosti ulomka spreminja vrednost števec  $N$ . To dela v 4. stopnjah vezanih kaskadno. Ena stopnja implementirana v FPGA je prikazana na sliki 7.7, blokovni diagram delovanja MASH pa na sliki 7.8 [26].



Slika 7.8: Blokovni diagram MASH modulatorja

## 7.2 Črpalka naboja



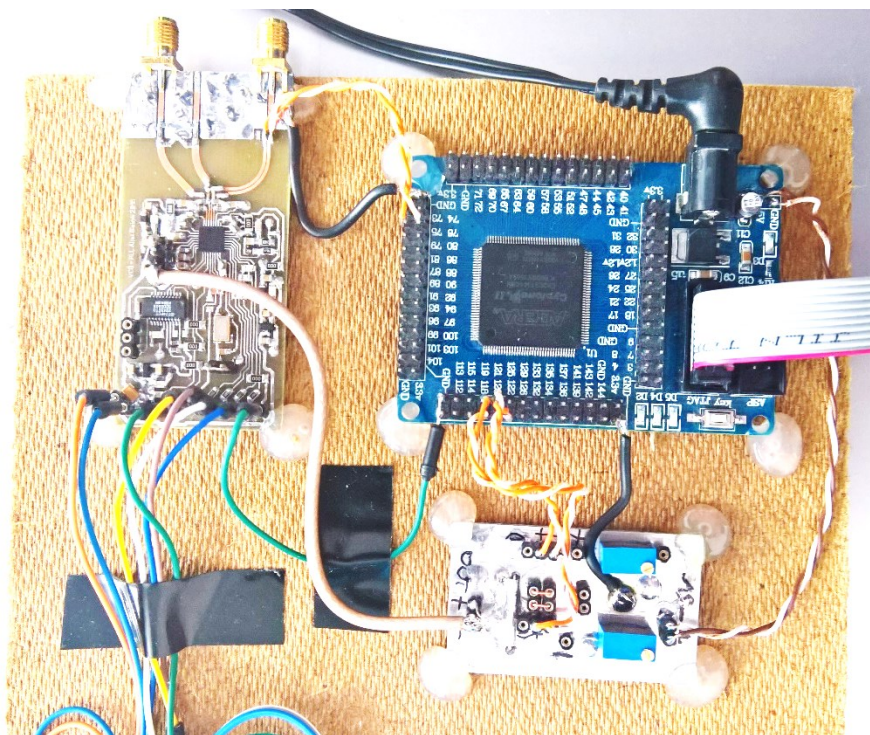
Slika 7.9: Izgled črpalke naboja

Črpalka nabojev je izvedena preprosto z uporabo dveh izhodov FPGA, ki jih lahko programsko preklopimo na maso, napajanje ali v stanje visoke impedance. Do zračnega sita sta povezavi pripeljani preko upora 680 Ohm. Tak pristop se je s



poskušanjem izkazal kot najučinkovitejši z vidika šuma in moduliranja vhoda napetostno krmiljenega oscilatorja. Zanjeno sito je identično tistemu pri MAX2871. Končen izgled prikazuje slika 7.9.

### 7.3 Gradnja celotnega sistema zanke s FPGA



Slika 7.10: Končen izgled PLL zanke s FPGA

Iz lastnih izkušenj mislim, da je najboljša možna prototipna montaža tiskanin na leseno ploščo ter pritrditev z vročim lepilom. Zamenjava tiskanine in popravljanje je tako sila enostavno. Končno postavitve prikazuje slika 7.10. Seveda tudi povezave niso idealne, najbolje bi bilo, da bi tekle ločeno od visokofrekvenčnih virov, vsaka tiskanina pa bi morala biti vgrajena v svoje ohišje, najbolje iz medenine, vendar je bila za začetni prototip izbrana preprosta rešitev.

Če želimo tiskanino izdelati v domači delavnici, smo postavljeni pred omejitve. FPGA čipa s številnimi nopicami ne moremo enostavno povezati na enoslojni tiskanini, brez da bi bili primorani sekati povezovalne linije, ki nosijo signal visokih frekvenc. Če bi tiskanino poslali v izdelavo v tovarno, bi si seveda lahko privoščili precej več, verjetno tudi izboljšali fazni šum, ter se znebili nekaterih presluhov, vendar bi to delo potem dobilo drugačne začetne pogoje. Verilog koda FPGA je priložena v dodatku.



## 8 Zaključek

Torej. Je čip MAX2871 uporaben? Da, za nekatere naloge bo več kot dovolj, tudi zaradi privlačne cene. Za druge ne. Za prvi lokalni oscilator spektralnega analizatorja? Ne, razen če se zadovoljimo s slabim faznim šumom in presluhom digitalnega takta.. Špičke lahko še nekako nadziramo z izbiro ulomka, ali pa jih odstranimo z računalniško obdelavo preden signal prikažemo uporabniku, toda fazni šum je enostavno prevelik. Morda bi ga nekoliko še uspeli znižati z uporabo višje frekvence faznega primerjalnika, ožjim sitom, še boljšim filtriranjem napetosti za VCO, vendar ga z uporabo različnih nastavitev znotraj čipa, ki jih veselo oglašuje proizvajalec, ne bomo mogli znižati.

Se lotimo gradnje PLL s FPGA čipom? Ne, FPGA raje porabimo za težko matematiko, ali morda digitalno obdelavo signala, PLL naloge pa prepustimo za to namenjenim čipom. Sicer dobro PLL zanko lahko sestavimo iz posameznih ločenih komponent, kot je to praksa v spektralnih analizatorjih višjega cenovnega razreda, a bomo za to potrebovali tudi industrijsko izdelavo tiskanin. Vendar se z implementacijo PLL zanke v FPGA veliko naučimo, čeprav proizvedemo popolnoma neuporaben rezultat. Z novim znanjem lahko potem bolj pametno načrtujemo vezje z obstoječimi PLL čipi, oziroma lahko ocenimo, do kakšne mere jim smemo zaupati.

In nenazadnje, dobro je da znamo sami izdelati hitro ter zanesljivo metodo merjenja faznega šuma in s tem prihranimo pri času, ki ga lahko porabimo za dodatno optimizacijo.





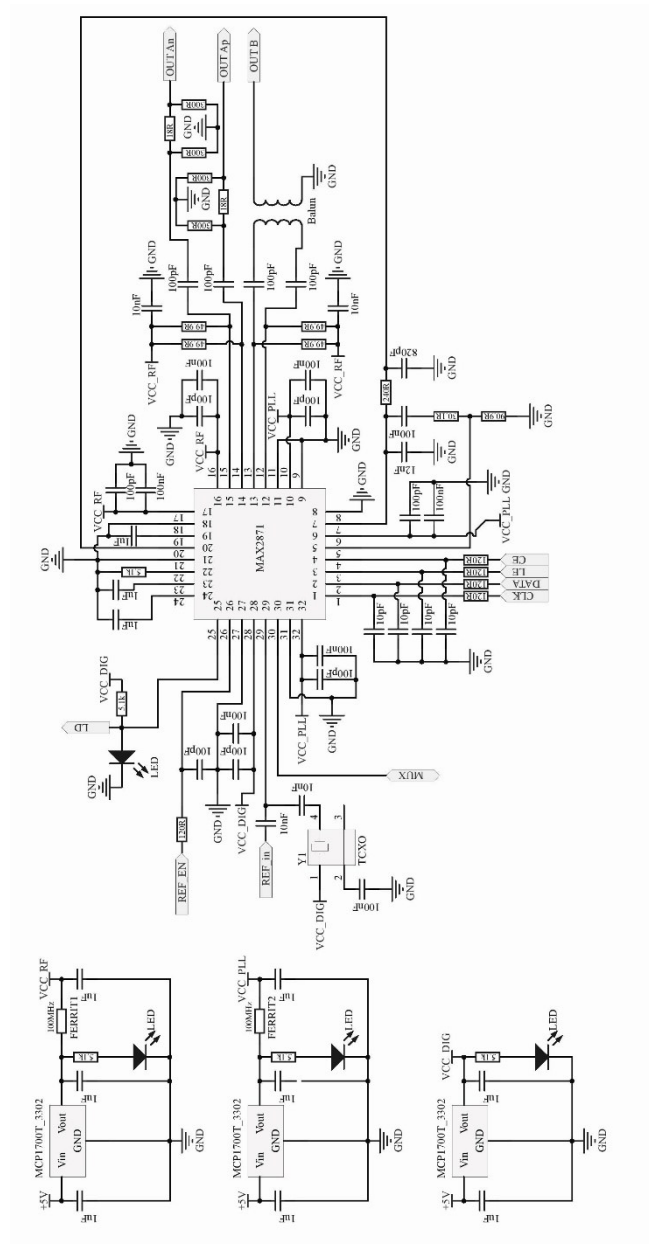
## Literatura

- [1] H. Bellescize, „La Reception Synchrone,“ *Onde Electr.*, Izv. 11, pp. 230-240, 1932.
- [2] W. Lindsey in M. Simon, *Phaselocked Loops and Their Application*, New York: IEEE, 1978.
- [3] A. Buchwald, K. Martin, A. Oki in K. Kobayashi, „A 6-GHz Integrated Phase-Locked Loop Using AlGaAs/GaAs Heterojunction Bipolar Transistors,“ *IEEE Journal of Solid-state Circuits*, Izv. 27, pp. 1752-1762, 1992.
- [4] B. Sklar, *Digital Communications: Fundamentals and Applications*, New York: Prentice-Hall, 1988.
- [5] J. G. Proakis, *Digital Communications*, New York: McGraw-Hill, 1983.
- [6] U. L. Rohde, *Digital PLL Frequency Synthesizers - Theory and Design*, New York: NY: John Wiley and Sons, 1983.
- [7] W. F. Egan, *Frequency Synthesis by Phase Lock*, Wiley Interscience, 1981.
- [8] Texas Instruments, „AN-1879 Fractional N Frequency Synthesis,“ 2013. [Elektronski]. Dostopno: <http://www.ti.com/lit/an/snaa062a/snaa062a.pdf>. [Poskus dostopa 21 1 2017].
- [9] Analog Devices, „AD9901,“ Analog Devices, Norwood, 1999.
- [10] Motorola, *Motorola MECL Data Book*, Motorola, 1993.
- [11] M. K. i. drugi, „A Si Bipolar 28 GHz Dynamic Frequency Divider,“ *IEEE Journal Solid-State Circuits*, Izv. 27, št. 12, pp. 1799-1804, 1992.

- [12] Texas Instruments, „PLL Fundamentals,“ 2011. [Elektronski]. Dostopno: <http://www.ti.com/lit/ml/snap003/snap003.pdf>. [Poskus dostopa 21 1 2017].
- [13] Agilent Technologies, "Phase noise measurement methods and techniques," Agilent Technologies, 2012. [Elektronski]. Dostopno: [http://www.keysight.com/upload/cmc\\_upload/All/PhaseNoise\\_webcast\\_19Jul12.pdf](http://www.keysight.com/upload/cmc_upload/All/PhaseNoise_webcast_19Jul12.pdf). [Accessed 8 1 2017].
- [14] Wikipedia, „Phase noise,“ 29 12 2016. [Elektronski]. Dostopno: [https://en.wikipedia.org/wiki/Phase\\_noise](https://en.wikipedia.org/wiki/Phase_noise). [Poskus dostopa 6 1 2017].
- [15] L. E. Larson, RF and Microwave Circuit Design for Wireless Communications, Boston: Artech House Publishers, 1996.
- [16] M. Vidmar, „Šum v radijskih komunikacijah,“ 2 2016. [Elektronski]. Dostopno: <http://antena.fe.uni-lj.si/literatura/Razno/SRK2016/SRK2016.pdf>. [Poskus dostopa 8 3 2017].
- [17] S. Meninger in M. Perrott, „Sigma\_delta Fractional-N Frequency Synthesis,“ Massachusetts Institute of Technology, Massachusetts, 2004.
- [18] Analog Devices, „HMC834,“ [Elektronski]. Dostopno: <http://www.analog.com/media/en/technical-documentation/data-sheets/hmc834.pdf>. [Poskus dostopa 21 1 2017].
- [19] Maxim Integrated, „MAX2871,“ 5 2016. [Elektronski]. Dostopno: <https://datasheets.maximintegrated.com/en/ds/MAX2871.pdf>. [Poskus dostopa 21 1 2017].
- [20] Maxim Integrated, „MAX2870/MAX2871 Evaluation Kits,“ 11 2015. [Elektronski]. Dostopno: <https://datasheets.maximintegrated.com/en/ds/MAX2870EVKIT.pdf>. [Poskus dostopa 22 1 2017].
- [21] FOX, „3.3V Ultra Miniature SMD HCMOS TCXO/VCTCXO,“ 2007. [Elektronski]. Dostopno: <http://www.foxonline.com/pdfs/fox924.pdf>. [Poskus dostopa 22 1 2017].
- [22] Microchip, „MCP1700,“ 2013. [Elektronski]. Dostopno: <http://ww1.microchip.com/downloads/en/DeviceDoc/20001826C.pdf>. [Poskus dostopa 22 1 2017].
- [23] National Instruments, „NI GPIB-USB-HS,“ National Instruments, [Elektronski]. Dostopno:

- <http://sine.ni.com/nips/cds/view/p/lang/sl/nid/201586>. [Poskus dostopa 22 1 2017].
- [24] Python, „PyVISA: Control your instruments with Python,“ Python, [Elektronski]. Dostopno: <https://pyvisa.readthedocs.io/en/stable/>. [Poskus dostopa 22 1 2017].
- [25] Agilent technologies, „User’s and Programmer’s Reference VOL 1,“ 5 2006. [Elektronski]. Dostopno: <http://cp.literature.agilent.com/litweb/pdf/E4440-90285.pdf>. [Poskus dostopa 22 1 2017].
- [26] A. Holme, „38-76 MHz Fractional-N Synthesizer,“ 2009. [Elektronski]. Dostopno: <http://www.aholme.co.uk/Frac3/Main.htm>. [Poskus dostopa 23 1 2017].
- [27] L. Heller, „Getting started with the EP2C5 Cyclone II Mini Board,“ [Elektronski]. Dostopno: <http://www.leonheller.com/FPGA/FPGA.html>. [Poskus dostopa 23 1 2017].
- [28] J. R. Vig, E. S. Ferre-Pikal, J. C. Camparo, L. S. Cutler, L. Maleki, W. J. Riley, S. R. Stein, C. Thomas, F. L. Walls in J. D. White, IEEE Standard Definitions of Physical Quantities for Fundamental Frequency and Time Metrology – Random Instabilities, IEEE, 1999.
- [29] A. A. Sweet, „A General analysis of noise in Gunn oscillators,“ v *IEEE*, 1972.
- [30] Z. H. A. B. A. Hodisan, „CAE Software Predicts PLL Phase Noise,“ *Microwaves and RF*, pp. 95-100, 1994.
- [31] V.F.Kroupa, „Noise Properties of PLL Systems,“ *IEEE Trans. on Communications*, pp. 2244-2251, 1982.

## A Shema vezja s PLL zanko



Slika A.1: Shema vezja s PLL zanko

## B Izvorna koda samodejne meritve faznega šuma

```
import visa
import time
import csv
rm = visa.ResourceManager()
inst = rm.open_resource('GPIB0::18::INSTR') # izbira naprave

# odpremo .csv datoteko
f = open("rezultat.csv", 'wt')
writer = csv.writer(f)
#writer.writerow(('kHz','dBc/Hz'))

# inicializacija naprave (pocistimo za 'jaz znam' budalami)
inst.write('INIT:CONT ON') # stalni sweep
inst.write('POW:ATT 20') # atenuacija 20 dB
inst.write('DISP:WIND:TRAC:Y:RLEV 10 dbm') # nastavimo ref. nivo
inst.write('POW:ATT:AUTO OFF')
inst.write('DISP:WIND:TRAC:Y:SPAC LOG') # logaritmicna skala
inst.write('DISP:WIND:TRAC:Y:PDIV 10 DB') # 10dB/div
inst.write('UNIT:POW DBM') # amplitudo v dBm
inst.write('POW:GAIN OFF') # brez ojačevalnika
inst.write('DISP:WIND:TRAC:Y:RLEV:OFFS 0.0') # offset 0.0dB
inst.write('CORR:CSET:ALL OFF') # izklopi vse morebitne korekcije
inst.write('POW:ATT:STEP 10') # korak atenuacije 10 dB
inst.write('SWE:TYPE SWE') # sweep type je nastavljen na sweep
inst.write('ADC:DITH AUTO') # samodejno dolocanje ADC podprtavanja
inst.write('ADC:RANG AUTO') # samodejno dolocanje območja delovanja
inst.write('BWID:AUTO ON') # RBW najprej na auto
inst.write('BWID:VID:AUTO ON') # VBW najprej na auto
inst.write('AVER OFF') # izkljuci povprečenje
inst.write('AVER:TYPE:LOG') # pripravimo povprečenje na log-pow
inst.write('DET NORM') # nastavimo detektor na NORMAL
inst.write('DET:AUTO ON') # samodejna nastavitvev
inst.write('FREQ:OFFS 0') # brez freq. odmika
inst.write("TRAC:MODE WRIT") # clear write linija

inst.write('FREQ:SPAN:FULL') # full span
```

```

# zacetek meritve
# funkcije za meritev faznega suma
inst.write('FREQ:SYNT:AUTO ON') # vse auto
inst.write('FREQ:STAR 2700 MHz') # zacetna frekvenca - območje iskanja
inst.write('FREQ:STOP 6200 MHz') # končna frekvenca
inst.write('DISP:WIND:TRAC:Y:RLEV 10 dbm') # nastavimo ref. nivo
input("Pritisni enter za zacetek meritve.")
#time.sleep(10)
print("Pocakamo na umerjanje instrumenta")
# meritve do 10MHz
# poiscemo in zozamo signal
inst.write('CALC:MARK ON') # marker 1 prizig
inst.write('CALC:MARK:MODE POS') # marker 1 to normal mode
inst.write('CALC:MARK:MAX') # peak search marker 1
inst.write('CALC:MARK:TRCK ON') # vkljucimo sledenje signalu
inst.write('CALC:MARK:CENT;*WAI') # marker 1 -> CF
inst.write('FREQ:SPAN 500 MHz') # zmanjsaj span
time.sleep(1)
inst.write('CALC:MARK:MAX') # peak search marker 1
inst.write('CALC:MARK:CENT;*WAI') # marker 1 -> CF
inst.write('FREQ:SPAN 100 MHz') # zmanjsaj span
time.sleep(1)
inst.write('CALC:MARK:MAX') # peak search marker 1
inst.write('CALC:MARK:CENT;*WAI') # marker 1 -> CF
inst.write('FREQ:SPAN 22 MHz') # zmanjsaj span
time.sleep(1)
inst.write('CALC:MARK:MAX') # peak search marker 1
inst.write('CALC:MARK:CENT;*WAI') # marker 1 -> CF
#inst.write('CALC:MARK:TRCK OFF') # izkljucimo sledenje signalu
#time.sleep(1)

# nastavimo filtre
inst.write('BWID:AUTO OFF')
inst.write('BWID:VID:AUTO OFF')
inst.write('BWID 300 kHz') # RBW nastavljanje
inst.write('BWID:VID 3 kHz') # VBW nastavljanje
inst.write('INIT:CONT OFF') # single shot
inst.write('INIT:IMM;*WAI') # sprozi prelet
inst.write('CALC:MARK:MAX') # peak search marker 1
inst.write('CALC:MARK:MODE DELT') # marker delta mode
time.sleep(1)

delta = 10000 # 10000 kHz - 10MHz
while (delta >= 1000): # merimo do 1 MHz
    inst.write("CALC:MARK:X " + str(delta) + " kHz")
    inst.write('CALC:MARK:Y?') # dobimo delta Y vrednost
    dbc = inst.read()
    dbc = float(dbc)

```

```
print(dbc)

# izracun faznega suma
dbc = dbc-54.7712+2.51

writer.writerow((delta,dbc))
#print(str(delta) + "kHz: " + str(dbc) + "dBc/Hz")
delta = delta - 100 # resolucija = 100kHz

# ozje meritve pod 1MHz
inst.write('INIT:CONT ON') # stalni sweep
inst.write('BWID:AUTO ON') # RBW najprej na auto
inst.write('BWID:VID:AUTO ON') # VBW najprej na auto
inst.write('CALC:MARK:MODE POS') # marker 1 to normal mode
inst.write('CALC:MARK:MAX') # peak search marker 1
inst.write('CALC:MARK:TRCK ON') # vkljucimo sledenje signalu
inst.write('CALC:MARK:CENT;*WAI') # marker 1 -> CF
inst.write('FREQ:SPAN 2 MHz') # zmanjsaj span
time.sleep(1)
inst.write('CALC:MARK:MAX') # peak search marker 1
inst.write('CALC:MARK:CENT;*WAI') # marker 1 -> CF
inst.write('CALC:MARK:TRCK OFF') # izkljucimo sledenje signalu

# nastavimo filtre
inst.write('BWID:AUTO OFF')
inst.write('BWID:VID:AUTO OFF')
inst.write('BWID 30 kHz') # RBW nastavljanje
inst.write('BWID:VID 500 Hz') # VBW nastavljanje
inst.write('INIT:CONT OFF') # single shot
inst.write('INIT:IMM;*WAI') # sprozi prelet
inst.write('CALC:MARK:MAX') # peak search marker 1
inst.write('CALC:MARK:MODE DELT') # marker delta mode
time.sleep(2)

while (delta >= 100): # merimo do 100 kHz
    inst.write("CALC:MARK:X " + str(delta) + " kHz")
    inst.write('CALC:MARK:Y?') # dobimo delta Y vrednost
    dbc = inst.read()
    dbc = float(dbc)
    print(dbc)

# izracun faznega suma
dbc = dbc-44.7712+2.51

writer.writerow((delta,dbc))
#print(str(delta) + "kHz: " + str(dbc) + "dBc/Hz")
delta = delta - 10 # resolucija = 10kHz
```

```

# ozje meritve pod 100kHz
inst.write('INIT:CONT ON') # stalni sweep
inst.write('BWID:AUTO ON') # RBW najprej na auto
inst.write('BWID:VID:AUTO ON') # VBW najprej na auto
inst.write('CALC:MARK:MODE POS') # marker 1 to normal mode
inst.write('CALC:MARK:MAX') # peak search marker 1
inst.write('CALC:MARK:TRCK ON') # vkljucimo sledenje signalu
inst.write('CALC:MARK:CENT;*WAI') # marker 1 -> CF
inst.write('FREQ:SPAN 200 kHz') # zmanjsaj span
time.sleep(2)
inst.write('CALC:MARK:MAX') # peak search marker 1
inst.write('CALC:MARK:CENT;*WAI') # marker 1 -> CF
inst.write('CALC:MARK:TRCK OFF') # izkljucimo sledenje signalu

# nastavimo filtre
inst.write('BWID:AUTO OFF')
inst.write('BWID:VID:AUTO OFF')
inst.write('BWID 500 Hz') # RBW nastavljanje
inst.write('BWID:VID 1 kHz') # VBW nastavljanje
inst.write('INIT:CONT OFF') # single shot
inst.write('INIT:IMM;*WAI') # sprozi prelet
inst.write('CALC:MARK:MAX') # peak search marker 1
inst.write('CALC:MARK:MODE DELT') # marker delta mode
time.sleep(1)

while (delta >= 20): # merimo do 20 kHz
    inst.write("CALC:MARK:X " + str(delta) + " kHz")
    inst.write('CALC:MARK:Y?') # dobimo delta Y vrednost
    dbc = inst.read()
    dbc = float(dbc)
    print(dbc)

# izracun faznega suma
dbc = dbc-26.9897+2.51

writer.writerow((delta,dbc))
#print(str(delta) + "kHz: " + str(dbc) + "dBc/Hz")
delta = delta - 1 # resolucija = 1kHz

# ozje meritve pod 10kHz
inst.write('INIT:CONT ON') # stalni sweep
inst.write('BWID:AUTO ON') # RBW najprej na auto
inst.write('BWID:VID:AUTO ON') # VBW najprej na auto
inst.write('CALC:MARK:MODE POS') # marker 1 to normal mode
inst.write('CALC:MARK:MAX') # peak search marker 1
inst.write('CALC:MARK:TRCK ON') # vkljucimo sledenje signalu

```



```
inst.write('CALC:MARK:CENT;*WAI') # marker 1 -> CF
inst.write('FREQ:SPAN 40 kHz') # zmanjsaj span
time.sleep(1)
inst.write('CALC:MARK:MAX') # peak search marker 1
inst.write('CALC:MARK:CENT;*WAI') # marker 1 -> CF
inst.write('CALC:MARK:TRCK OFF') # izključimo sledenje signalu

# nastavimo filtre
inst.write('BWID:AUTO OFF')
inst.write('BWID:VID:AUTO OFF')
inst.write('BWID 200 Hz') # RBW nastavljanje
inst.write('BWID:VID 1 kHz') # VBW nastavljanje
inst.write('INIT:CONT OFF') # single shot
inst.write('INIT:IMM;*WAI') # sprozi prelet
inst.write('CALC:MARK:MAX') # peak search marker 1
inst.write('CALC:MARK:MODE DELT') # marker delta mode
time.sleep(1)

delta = 19000 #Hz
while (delta >= 1000): # merimo do 1 kHz
    inst.write("CALC:MARK:X " + str(delta) + " Hz")
    inst.write('CALC:MARK:Y?') # dobimo delta Y vrednost
    dbc = inst.read()
    dbc = float(dbc)
    print(dbc)

# izracun faznega suma
dbc = dbc-23.0103+2.51

writer.writerow((delta/1000,dbc))
#print(str(delta) + "kHz: " + str(dbc) + "dBc/Hz")
delta = delta - 100 # resolucija = 100Hz

f.close()
rm.close()
```



## C Izvorna koda FPGA

```
module FPGA_PLL(clock,clock_OUT,LED_out,VCO_clock,CP_up,CP_down);
input clock;
input VCO_clock;
output LED_out;
output clock_OUT;
output CP_up, CP_down;

reg [24:0] clock_divider;
reg [ 6:0] N;
reg [31:0] F;

wire LED_out = clock_divider[24];
wire clock_OUT = clock_divider[2];

initial begin
    clock_divider = 24'b0;
    N = 7'd2;
    F = 32'd5000;
end

wire vco_clk, vco_gate, vco_phase;
wire ref_clk, ref_gate, ref_phase;
wire pfd_out, msh_cycle;
wire [6:0] dN, div;
reg msh_gate;

assign vco_clk = VCO_clock;
assign ref_clk = clock;

DIVIDER ref_div(.clk(clock), .ce(ref_gate), .data(7'd99));
DIVIDER vco_div(.clk(VCO_clock), .ce(vco_gate), .data(N + dN), .div(div));

BUFPGCE vco_bufgce(.I(vco_clk), .CE(vco_gate), .O(vco_phase));
BUFPGCE ref_bufgce(.I(ref_clk), .CE(ref_gate), .O(ref_phase));
BUFPGCE msh_bufgce(.I(vco_clk), .CE(msh_gate), .O(msh_cycle));

always @ (negedge vco_clk)
```

```

msh_gate <= (div==7'd4);

MASH msh(.clk(msh_cycle), .F(F), .dN(dN));

AD9901 pfd(.vco_phase(vco_phase), .ref_phase(ref_phase), .pfd_out(pfd_out));
assign CP_up = pfd_out?1'b1:1'b0;
assign CP_down = pfd_out?1'b0:1'b1;

always @ (posedge VCO_clock) begin
    clock_divider <= clock_divider + 1'b1;
end

endmodule

module DIVIDER(
    input clk,
    input [6:0] data,
    output reg ce,
    output reg [6:0] div);

    always @ (posedge clk)
        div <= div? div-1'b1 : data;

    always @ (negedge clk)
        ce <= ~|div;
endmodule

module AD9901(
    input vco_phase,
    input ref_phase,
    output pfd_out);

    wire ref_tff_q, ref_dff_q;
    wire vco_tff_q, vco_dff_q;

    FD ref_tff(.Q(ref_tff_q), .D(~ref_tff_q), .C(ref_phase));
    FD vco_tff(.Q(vco_tff_q), .D(~vco_tff_q), .C(vco_phase));

    wire xout = ref_tff_q ^ vco_tff_q;

    FDC ref_dff(.Q(ref_dff_q), .D(xout), .C(ref_phase), .CLR(~vco_dff_q));
    FDP vco_dff(.Q(vco_dff_q), .D(xout), .C(vco_phase), .PRE(ref_dff_q));

    assign pfd_out = (xout & vco_dff_q) | ref_dff_q;
endmodule

module FD ( // D Flip-Flop
D, // Data Input

```

```
C, // Clock Input
Q // Q output
);
input D, C;
output Q;
reg Q;

initial begin
    Q = 1'b0;
end

always @ ( posedge C ) begin
    Q <= D;
end
endmodule

module FDC( // D Flip-Flop with Asynchronous Clear
D, // Data Input
C, // Clock Input
CLR, // Reset input
Q // Q output
);
input D, C, CLR;
output Q;
reg Q;

initial begin
    Q = 1'b0;
end

always @ ( posedge C or negedge CLR)
if (~CLR) begin
    Q <= 1'b0;
end else begin
    Q <= D;
end
end
endmodule

module FDP( // D Flip-Flop with Asynchronous Preset
D, // Data Input
C, // Clock Input
PRE, // PRESET input
Q // Q output
);
input D, C, PRE;
output Q;
reg Q;
```

```
initial begin
    Q = 1'b0;
end

always @ ( posedge C or posedge PRE)
if (PRE) begin
    Q <= 1'b1;
end else begin
    Q <= D;
end
endmodule

module BUFGCE( // Global Clock Buffer with Clock Enable
I, // I Input
O, // O Output
CE // CE Input
);
input I, CE;
output O;

assign O = CE?I:1'b0;
endmodule

module MASH(
input clk,
input [31:0] F,
output [6:0] dN);

wire [127:0] cmd, sum;
wire [ 27:0] fbi, fbo;

STAGE stage [3:0] (.clk(clk), .cmd(cmd), .sum(sum), .fbi(fbi), .fbo(fbo));

assign cmd = {sum[95:0],F};
assign {fbi,dN} = {7'b0,fbo};

endmodule

module STAGE(
input clk,
input [31:0] cmd,
output [31:0] sum,
input [6:0] fbi,
output [6:0] fbo);

wire carry;
```

```
reg [31:0] acc;
reg [6:0] fbd;

initial {acc,fbd} <= 0;

assign {carry,sum} = acc + cmd;
assign fbo = carry + fbi - fbd;

always @ (posedge clk) begin
    acc <= sum;
    fbd <= fbi;
end

endmodule
```