UNIVERZA V MARIBORU

Fakulteta za elektrotehniko,
računalništvo in informatiko

Gregor Donaj

GENERATOR NAKLJUČNIH ŠTEVIL REALIZIRAN S KAOTIČNIM OSCILATORJEM

Diplomsko delo

Maribor, september 2010

Diplomsko delo univerzitetnega študijskega programa elektrotehnika

GENERATOR NAKLJUČNIH ŠTEVIL REALIZIRAN S KAOTIČNIM OSCILATORJEM

Študent:	Gregor Donaj
Študijski program:	UN ŠP Elektrotehnika
Smer:	Elektronika
Mentor:	izr. prof. dr. Tomaž Dogša

Maribor, september 2010

FERI
Fakulteta za elektrotehniko, računalništvo in informatiko
Smetanova ulica 17
2000 Maribor

Številka: E-2602
Datum in kraj: 25. 05. 2010, Maribor

Na osnovi 330. člena Statuta Univerze v Mariboru (Ur. l. RS, št. 1/2010)

SKLEP O DIPLOMSKEM DELU

1. **Gregorju Donaju,** študentu univerzitetnega študijskega programa ELEKTROTEHNIKA, smer Elektronika, se dovoljuje izdelati diplomsko delo pri predmetu Nelinearna elektronika.

2. **MENTOR: izred. prof. dr. Tomaž Dogša**

3. **Naslov diplomskega dela:**
GENERATOR NAKLJUČNIH ŠTEVIL REALIZIRAN S KAOTIČNIM OSCILATORJEM

4. **Naslov diplomskega dela v angleškem jeziku:**
RANDOM NUMBER GENERATOR BASED ON CHAOTIC OSCILLATOR

5. Diplomsko delo je potrebno izdelati skladno z "Navodili za izdelavo diplomskega dela" in ga oddati v treh izvodih (en vezan izvod in dva nevezana izvoda) ter en izvod elektronske verzije do 25. 05. 2011 v referatu za študentske zadeve.

Pravni pouk: Zoper ta sklep je možna pritožba na senat članice v roku 3 delovnih dni.

Dekan:

Obvestiti:
- kandidata,
- mentorja,
- odložiti v arhiv.

ZAHVALA

Zahvalujem se svojemu mentorju izr. prof. dr. Tomažu Dogši za pomoč pri izdelavi diplomske naloge in gospodu Boštjanu Založniku za pomoč s programsko opremo.

Zahvalujem se tudi staršem, ki so mi omogočili študij.

GENERATOR NAKLJUČNIH ŠTEVIL REALIZIRAN S KAOTIČNIM OSCILATORJEM

Ključne besede: generatorji naključnih števil, teorija kaosa, kaotična vezja, dinamična vezja, statistični testi, testiranje naključnosti

UDK: 621.3.049.7(043.2)

Povzetek

V diplomski nalogi je predstavljen generator naključnih števil, ki temelji na analognem kaotičnem vezju, in statistično testiranje njegove primernosti. Predstavljeni so osnovni pojmi iz teorije kaosa in generatorjev naključnih števil. Podrobneje so opisani tudi statistični testi, ki se uporabljajo za preverjanje ustreznosti generatorja naključnih števil za kriptografske namene. Izbrano je kaotično vezje in postopek pridobivanja naključnih števil iz njegovega obnašanja. Delovanje vezja je preverjeno s simulacijami, meritvami in statističnimi testi.

RANDOM NUMBER GENERATOR BASED ON CHAOTIC OSCILLATOR

Key words: random number generators, chaos theory, chaotic circuits, dynamic circuits, statistical tests, testing randomness

UDK: 621.3.049.7(043.2)

Abstract

This diploma work describes a random number generator based on an analogue chaotic circuit. Basic concepts from chaos theory and random number generators are presented. A description of statistical tests is also given. These tests are used for testing random number generator used for cryptographic purposes. A chaotic circuit and an algorithm for generating random numbers are chosen. The operation of the circuit is tested with simulations, measurements and statistical tests.

Kazalo

1 Uvod

Generatorji naključnih števil se uporabljajo na različnih področjih kot so razne računalniške simulacije in igrice, statistične analize ter igre na srečo. Eno izmed teh področij je tudi kriptografija, kjer so generatorji naključnih števil potrebni za tvorjenje šifer.

Prenos sporočil po informacijskih kanalih je večkrat napaden s strani tretjih oseb, ki poskušajo sporočilo prebrati ali pa ga spremeniti. Da bi povečali varnost prenosa sporočil, jih šifriramo s pomočjo kriptografskih sistemov. Ti potrebujejo ključ, ki ga dobimo z generatorjem naključnih števil ali generatorjem psevdonaključnih števil. Takšni generatorji nam dajejo binarno zaporedje ničel in enic.

Obstaja več vrst takšnih generatorjev. Generatorje psevdonaključnih števil najdemo že v prevajalnikih večine programskih jezikov, vendar zaporedje števil, ki ga dajejo, ni dovolj naključno za kriptografske namene.

Čeprav poznamo tudi boljše psevdonaključne generatorje, so njihovi izhodi natančno določeni s semeni. Zato ob njih uporabljamo generatorje naključnih števil, ki generirajo semena. Generatorji naključnih števil uporabljajo različne fizikalne pojave za generiranje števil [9].

V šestdesetih letih dvajsetega stoletja se je začela razvijati teorija kaosa. Ta se ukvarja z obnašanjem sistemov, ki so sicer deterministični, njihovo obnašanje pa vseeno izgleda naključno. Uporaba teorije kaosa v kriptografiji se že kaže v razvoju analognih in digitalnih kriptografskih sistemov. V teh sistemih se kaotična vezja uporabljajo za kaotično maskiranje, modulacijo ali preklapljanje.

V diplomski nalogi je predstavljena ideja uporabe kaotičnega elektronskega oscilatorja za tvorjenje naključnih števil. Kaotični oscilatorji so zelo občutljivi na zunanje vplive, kot so temperatura in napajalna napetost. Zato lahko že majhne spremembe v teh vplivih, ki so lahko posledice naključnih pojavov, povzročijo drugačno obnašanje oscilatorjev. Iz obnašanja oscilatorja bomo izluščili neko število. Predvidevamo, da če bodo zunanji vplivi res naključni, bodo dobljena števila tudi naključna.

Delovanje generatorja bomo preverili s statističnimi testi, ki so namenjeni za testiranje generatorjev naključnih števil za kriptografske namene.

2 Teorija kaosa

Teorija kaosa je matematično področje, ki se ukvarja s preučevanjem obnašanja nekaterih sistemov, ki imajo določene značilnosti. Pravimo, da je njihovo obnašanje kaotično ali pa ga imenujemo kar kaos. Ta nastopa le v deterministčnih, nelinearnih, dinamičnih sistemih. Začetki teorije kaosa segajo do šestdesetih let dvajsetega stoletja [18].

Dinamični sistemi so fizikalni sistemi, ki se spreminjajo skozi čas. Glede na čas, v katerem se spreminjajo, jih lahko ločimo na dve skupini. Prva so diskretni sistemi, ki jih opazujemo le v posameznih trenutkih v času. Njihovo obnašanje lahko opisujemo z diferenčnimi enačbami. Druga skupina pa so dinamični sistemi, ki jih lahko opazujemo v zveznem času. Njihovo obnašanje pa opisujemo z diferencialnimi enačbami. Nelinearni sistemi so tisti, v katerih lahko odnose med spremenljivkami opišemo le z nelinearnimi enačbami [18].

Da je nek sistem deterministčen pomeni, da je njegovo obnašanje vedno natančno določeno z začetnimi pogoji. Primeri deterministčnih sistemov so tisti, ki jih opisujemo z Newtonovo mehaniko. Če v primeru poševnega meta poznamo začetne pogoje (hitrost telesa in kot pod katerim ga vržemo), lahko za vsak trenutek v času napovemo lego telesa. Sistemi, ki niso deterministični, nastopajo na primer v kvantni mehaniki, kjer lahko obnašanje sistemov napovedujemo le z verjetnostnimi funkcijami.

Dinamične sisteme lahko ločimo tudi na konzervativne, ki s časom ne izgubljajo energije, in na disipativne, ki izgubljajo energijo [18].

Obnašanje sistemov lahko prikazujemo na dva načina. Lahko narišemo graf odvisnosti spremenljivk stanja sistema od časa ali pa narišemo potek trajektorije v prostoru stanj ali faznem prostoru. Če narišemo graf po času lahko vidimo kako se posamezna spremenljivka spreminja skozi čas ter kako hitro se spreminja [18].

Prostor stanj je abstraktni matematični prostor, katerega koordinatne osi predstavljajo spremenljivke stanja sistema. Če povežemo točke ki predstavljajo stanje nekega sistema ob različnih časih, dobimo krivuljo v prostoru stanj, ki jo imenujemo trajektorija. Z njo lahko prikažemo, kako se spreminjajo spremenljivke stanja. Iz nje pa ne vidimo, kako hitro se spreminjajo. Območje v prostoru stanj, v katerem se giblje trajektorija, imenujemo atraktor [18].

Enostaven primer dinamičnega sistema, ki ni kaotičen, je harmonično nihalo. Opišemo ga z diferencialno enačbo

$$\ddot{x} + \omega^2 x = 0, \tag{2.1}$$

ki ima le eno spremenljivko stanja – x. Harmonično nihalo pa lahko opišemo tudi s sistemom enačb

$$\begin{aligned} \dot{x} &= y, \\ \dot{y} &= -\omega^2 x, \end{aligned} \tag{2.2}$$

ki ga dobimo tako, da v enačbo 2.1 vpeljemo novo spremenljivko $y = \dot{x}$. Podobno lahko tudi druge enačbe višjih redov preoblikujemo v sisteme enačb prvega reda, ki jih lahko enostavno numerično rešujemo. Trajektorija harmoničnega nihanja v prostoru stanj se giblje po elipsi s središčem v izhodišču.

V šestdesetih letih dvajsetega stoletja se je Edward Lorenz ukvarjal z modeliranjem vremena. Med računalniškimi simulacijami je opazil, da se njegov model ob podobnih začetnih pogojih začenja obnašati zelo različno. Lorenz je začel podrobneje preučevati ta pojav. Njegovo odkritje velja za začetek teorije kaosa [6].

Pogledali smo si že primer sistema, ki ni kaotičen. Primeri kaotičnih sistemov so Lorenzev sistem:

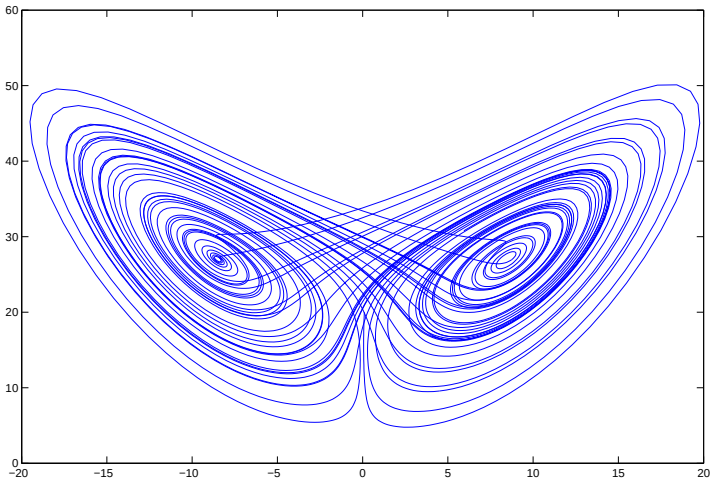
$$\begin{aligned} \dot{x} &= -v_1 x + v_1 y, \\ \dot{y} &= -xz + v_2 x - y, \\ \dot{z} &= xy - az, \end{aligned} \tag{2.3}$$

ali pa Rösslerjev sistem:

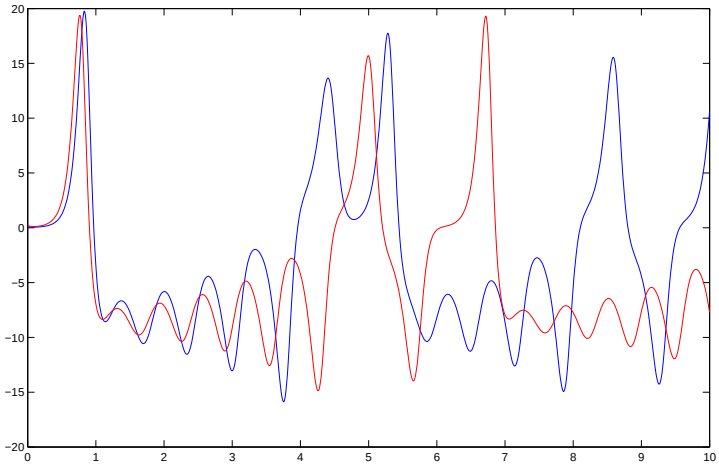
$$\begin{aligned} \dot{x} &= -(y + z), \\ \dot{y} &= x + ay, \\ \dot{z} &= b + z(x - c). \end{aligned} \tag{2.4}$$

V teh enačbah se nahajajo nekatere konstante oziroma parametri, katerim še nismo pripisali vrednosti. Na splošno se ti sistemi ne obnašajo kaotično, razen ob določenih vrednostih teh parametrov. Če v Lorenzev sistem vstavimo vrednosti parametrov $v_1 = 10$, $v_2 = 28$ in $a = 8/3$, potem dobimo kaotični sistem [18]. Slika 2.1 prikazuje njegov atraktor v ravnini x–z, slika 2.2 pa prikazuje potek spremenljivke x v Lorenzevem sistemu za dva podobna začetna pogoja. Lepo je vidna podobnost v potekih

spremenljivke na začetku in razlika med potekih proti koncu.



Slika 2.1: Lorentzov atraktor



Slika 2.2: Potek spremenljivke stanja x v Lorentzovem sistemu ob različnih začetnih pogojih

Kaotično obnašanje najdemo tudi v laserjih, dinamiki tekočin, nekaterih kemičnih reakcijah, gibanju nebesnih teles, spreminjanju populacije v nekem ekološkem sistemu in elektronskih vezjih [15].

2.1 Osnovne značilnosti kaotičnih sistemov

Vsi kaotični sistemi imajo nekatere skupne značilnosti, kot so:

1. kaotični sistemi so deterministični in nelinearni;
2. obnašanje kaotičnih sistemov izgleda neurejeno;
3. kaotični sistemi imajo povratno vezavo;
4. kaotični sistemi so lahko preprosti. V diskretnem času so to lahko sistemi z eno spremenljivko, v zveznem času pa sistemi z vsaj tremi spremenljivkami;
5. kaotično obnašanje je avtonomen proces. To pomeni, da se vzdržuje sam, brez kakšne zunanje regulacije;
6. kaotično obnašanje ni posledica nenatančnih meritev ali računskih napak;
7. spremenljivke stanja se gibljejo v omejenem območju prostora stanj, imenovanem atraktor;
8. obnašanje sistema je hiperobčutljivo na začetne pogoje;
9. dolgoročno napovedovanje obnašanje ni možno;
10. kratkoročno napovedovanje pa je možno;
11. Fourierev spekter obnašanja je širok s posameznimi vrhi;
12. trajektorija ima lahko fraktalne lastnosti;
13. med spreminjanjem nekega parametra lahko nekaotični sistem preteče več tipičnih korakov in preide v kaotičnega [18].

Hiperobčutljivost

Poglejmo si podrobneje hiperobčutljivost na začetne pogoje, ki jo je odkril Lorenz med proučevanjem vremena. Hiperobčutljivost pomeni, da lahko že zelo majhne razlike v enem ali več začetnih pogojih nekega sistema, ki se zdijo nepomembne, s časom povzročijo velike razlike v obnašanju sistema. Takšne razlike so lahko posledica nenatančnih meritev, šuma ali pa zaokroževanja podatkov. Razlike med obnašanjem so na začetku majhne in šele čez nekaj časa postanejo večje in celo primerljive z velikostjo

atraktorja. Če poznamo nek sistem, lahko z ustreznim modelom napovedujemo njegovo obnašanje. Čeprav bo napoved na začetku točna, bo zaradi napak v podatkih in v računski metodi, ki jo uporabljamo, prišlo kasneje do odstopanja.

Če si ogledamo potek trajektorij, ki izhajajo iz podobnih začetnih pogojev, v nekaotičnih in kaotičnih sistemih, opazimo, da se trajektoriji v nekaotičnih sistemih začneta približevati ali pa ostajata na enaki razdalji. V kaotičnih sistemih pa se zgodi ravno nasprotno – trajektoriji se začneta oddaljevati. Ker sta trajektoriji na začetku blizu, bosta to še nekaj časa ostali in se počasi oddaljevali. Na večji razdalji se bo tudi razdalja hitreje večala. V nekaterih primerih lahko večanje razdalj med trajektorijama primerjamo z eksponentno funkcijo. Če poznamo začetne pogoje takšnega sistema, lahko torej za omejen čas napovedujemo njegovo obnašanje.

Primer, ki ga srečamo vsak dan, za tak sistem je vreme, ki ga lahko napovedujemo kratkoročno, ne moremo pa ga napovedovati dolgoročno. V meteorologiji se občutljivost na začetne pogoje imenuje tudi metuljev pojav.

Hiperobčutljivost je ena najpomembnejših lastnosti kaotičnih sistemov in bo tudi pomembna v nadaljevanju diplomske naloge. Njen vpliv je prikazan tudi na sliki 2.2.

Kot smo že omenili, opisujemo dinamične sisteme z diferencialnimi enačbami. Te pa rešujemo z računalnikom z numeričnimi metodami. Hiperobčutljivosti bi se lahko poskusili izogniti s tem, da bi uporabili računske postopke z večjo natančnostjo. S tem bi sicer zmanjšali napake, ki nastanejo zaradi računske metode, vendar pa nebi preprečili večjih odstopanj. Dejstvo, da ne moremo dolgoročno napovedovati obnašanje sistemov, torej ni posledica omejene natančnosti računalnikov [18].

Kaotični atraktor

Omenjali smo že atraktor. To je množica točk ali območje v prostoru stanj, v katerem se dinamični sistem v nekem trenutku lahko nahaja. Trajektorije se gibljejo skozi določene dele atraktorja pogosteje kot pa skozi druge. Ta območja niso odvisna od začetnih pogojev.

Nekaotični atraktorji imajo običajno obliko točk, krivulj ali pa gladkih ploskev. Majhne spremembe v začetnih pogojih nimajo velikega vpliva na obliko atraktorja. V primeru nihala bi sprememba začetnih pogojev povzročila le spremembo velikosti elipse ali pa spremembo razmerja med širino in višino elipse, po kateri se giblje trajektorija.

Kaotični atraktorji nimajo neke pravilne geometrije. Pogosto jih imenujemo tudi čudni atraktorji. Nimamo enotne definicije za kaotični atraktor. Definiramo ga lahko

kot kompleksna ravnina v prostoru stanj, h kateri se giblje trajektorija asimptotično in kjer se obnaša kaotično. Druga možna definicija je, da je kaotični atraktor takšen atraktor, ki kaže hiperobčutljivost na začetne pogoje. Oblika kaotičnih atraktorjev je nepravilna s kompleksno notranjo strukturo. Primer atraktorja je na sliki 2.3.

Lyapunov eksponent

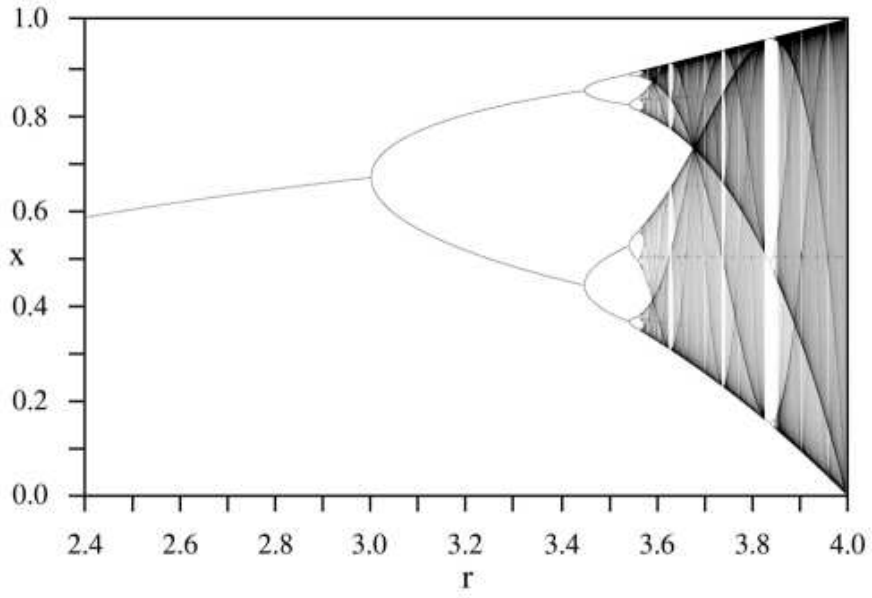
Kako hitro se dve trajektoriji v povprečju na atraktorju približujeta oziroma oddaljujeta, lahko podamo z Lyapunovim eksponentom. Da ga lahko definiramo, se mora razdalja med trajektorijama spreminjati eksponentno. Negativen Lypunov eksponent pomeni, da se trajektoriji približujeta, pozitiven pa pomeni, da se oddaljujeta. Običajno najdemo pozitiven eksponent le v kaotičnih sistemih, negativen eksponent pa še ne pomeni, da sistem ni kaotičen. Lyapunov eksponent se izračuna kot povprečje za območje celotnega atraktorja. Če je ta negativen, to še ne pomeni, da ni posameznih manjših območij, kjer je eksponent pozitiven. Zato negativen eksponent še ne pomeni nujno, da sistem ni kaotičen [18].

Bifurkacije

Za kaotične sisteme velja, da se obnašajo kaotično le ob nekih določenih vrednostih parametrov sistema. Če izberemo en parameter in ga spreminjajo, spreminjamo s tem tudi obnašanje sistema. Ta parameter imenujemo kontrolni parameter.

Bifurkacije so nenadne spremembe v obnašanju nekega sistema, ki se zgodijo ob nekih vrednostih kontrolnega parametra. Bifukacijske točke so vrednosti parametra, kjer te spremembe nastopijo. Kontrolni parameter zato imenujemo tudi bifurkacijski parameter.

Običajno so spremembe v bifurkacijskih točkah podvojitve neke količine, s katero opisujemo obnašanje sistema. V primeru elektronskih vezij lahko govorimo o podvojitvah periode. Tako imamo na začetku eno periodo, potem dve, potem štiri itn. V določenih intervalih bifurkacijskega parametra pa dobimo kaotično obnašanje. Ti intervali so običajno manjši od intervalov, kjer se sistem ne obnaša kaotično. Bifurkacije lahko prikažemo tudi grafično v bifurkacijskem diagramu, ki prikazuje obnašanje sistema glede na bifurkacijski parameter. Na sliki 2.3 je prikazan bifurkacijski diagram za logistično enačbo, ki opisuje spreminjaje velikosti populacije neke vrste v diskretnih časovnih intervalih [18].



Slika 2.3: Bifurkacijski diagram za logistično enačbo [15]

2.2 Kaotična elektronska vezja

Vsa elektronska vezja, ki vsebujejo kondenzator ali tuljavo, so zvezni dinamični sistemi. Za spremenljivke stanja v teh sistemih običajno izberemo trenutne vrednosti tokov skozi tuljave ali flukse v tuljavah in trenutne napetosti ali elektrine na kondenzatorjih.

Le nekatera dinamična električna vezja se lahko obnašajo kaotično. Pogoj, da se neko vezje lahko obnašajo kaotično, je, da je to vezje neavtonomno nelinearno vezje vsaj drugega reda ali pa avtonomno nelinearno vezje vsaj tretjega reda. To pomeni, da so vezju prisotni vsaj dva oz. vsaj trije reaktivni elementi – kondenzatorji ali tuljave [5].

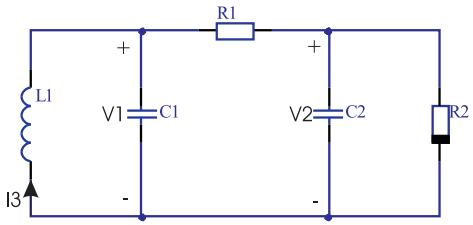
V elektronskih vezjih se lahko kondenzatorji in tuljave pojavijo kot parazitni elementi. Te moramo v nekaterih primerih tudi upoštevati, kadar govorimo o številu reaktivnih elementov. Tako lahko kaotično obnašanje kažejo tudi vezja, ki sicer izgledajo enostavnejša, kot pa je pogoj za kaotično obnašanje.

Kaotična vezja srečamo bolj redko v elektroniki. Znan je primer Chujevega oscilatorja, ki je prikazan na sliki 2.4. Na sliki je *R2* nelinearni upor, ki ga realiziramo s pomočjo operacijskih ojačevalnikov, bifurkacijski parameter pa je upor *R1*. Celotna realizacija vezja je na sliki 2.5. Karakteristika upora *R2* je na sliki 2.6. Enačbe, ki

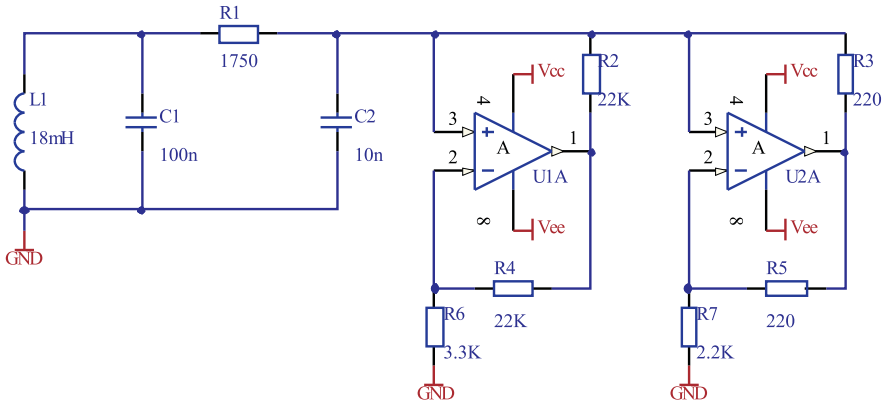
opisujejo obnašanje Chujevega oscilatorja so:

$$\begin{aligned} \dot{I}_3 &= -\frac{1}{L}U_1, \\ \dot{U}_1 &= \frac{1}{C_1}I_3 - \frac{G}{C_1}(U_1 - U_2), \\ \dot{U}_2 &= \frac{G}{C_2}(U_1 - U_2) - \frac{1}{C_2}\cdot f(U_2), \end{aligned} \tag{2.5}$$

kjer je f karakteristika upora $R2$, G pa prevodnost upora $R1$ [5]. Zraven Chujevega oscilatorja poznamo še druga vezja, ki se lahko obnašajo kaotično, kot so Duffing-Holmesov oscilator [13] ali pa vezje, ki temelji na Lorenzevem sistemu [2].



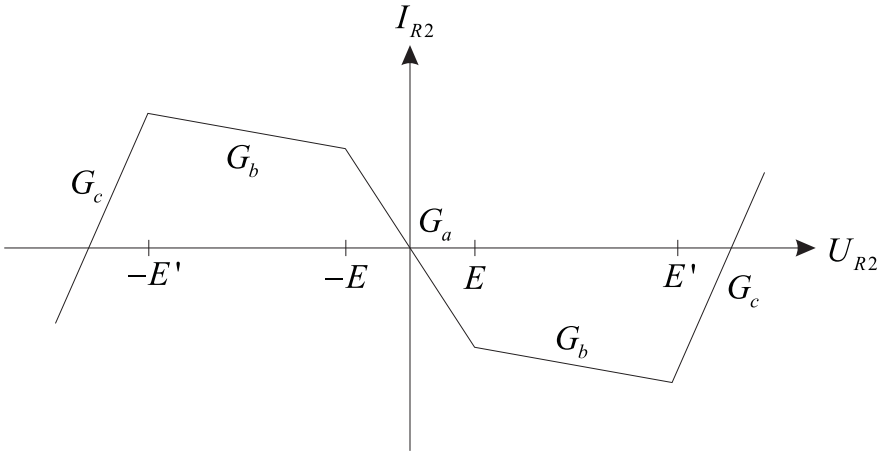
Slika 2.4: Kaotični Chujev oscilator



Slika 2.5: Realizacija Chujevega oscilatorja

Na splošno imamo v kaotičnih vezjih več elementov, ki jih lahko uporabljamo kot bifurkacijski parameter. Zraven različnih uporov so to lahko tudi vrednosti kondenzatorjev ali pa tuljav.

V primeru Chujevega oscilatorja smo analizirali vezje in izpeljali enačbe, ki ga opisujejo. Lahko pa storimo tudi obratno. Če imamo nek sistem podan samo z enačbami, lahko s sintezo dobimo vezje, ki jim ustreza. Če hočemo diferencialne enačbe pretvoriti v vezje bomo rabili različna elemente, kot so konveksni in konkavni upori, analogni množilniki ter operacijski ojačevalniki, ki nam lahko služijo kot ojačevalniki, integratorji, diferenciatorji in seštevalniki.



Slika 2.6: Karakteristika upora $R2$ v Chujevem oscilatorju

Poglejmo si primer sinteze vezja. Sestaviti hočemo spodbujan harmonični oscilator, ki ga opisuje diferencialna enačba

$$\ddot{x} - A\dot{x} + Bx = 0. \tag{2.6}$$

Števili A in B izberimo tako, da bo imela enačba $B\lambda^2 - A\lambda + 1 = 0$ dve kompleksni rešitvi, katerih realna komponenta je rahlo večja od 0. Tako dosežemo počasno večanje amplitude. Če izberemo $A = 0.02$ in $B = 1$ ter dodamo začetni pogoj $x(0) = 0$, bo rešitev diferencialne enačbe funkcija

$$x(t) = C \cdot e^{0,01 \cdot t} \cdot \sin 0,99995 \cdot t, \tag{2.7}$$

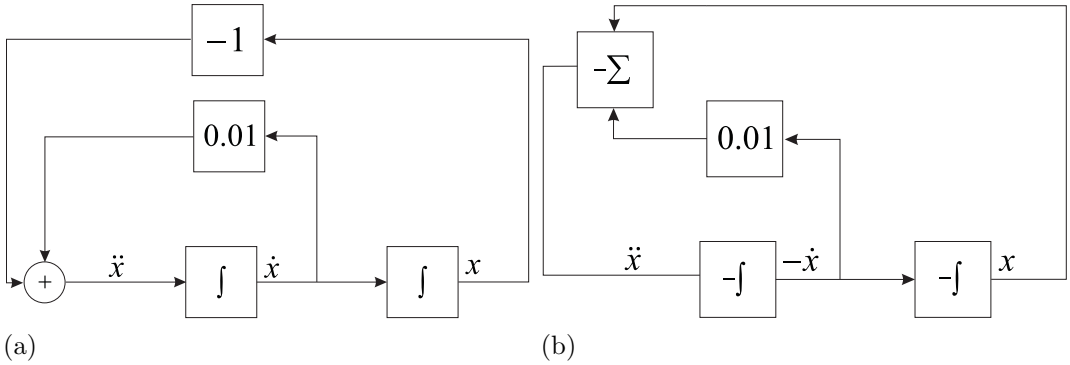
kjer je C poljubna konstanta.

Sedaj začnemo s sintezo vezja. Enačbo najprej preuredimo tako, da dobimo na levi strani le najvišji odvod spremenljivke x:

$$\ddot{x} = 0,02\dot{x} - x. \tag{2.8}$$

Enačbo preoblikujemo v blok shemo, ki je prikazana na sliki 2.7a. Z operacijskim ojačevalcem lahko sestavimo preprost integrator, ki pa na izhodu spremeni predznak. Prav tako lahko sestavimo preprost seštevalnik, ki tudi spremeni predznak. Blok shemo lahko preuredimo v shemo, ki je prikazana na sliki 2.7b. Tukaj smo upoštevali, da lahko te gradnike sestavimo z manj elementi.

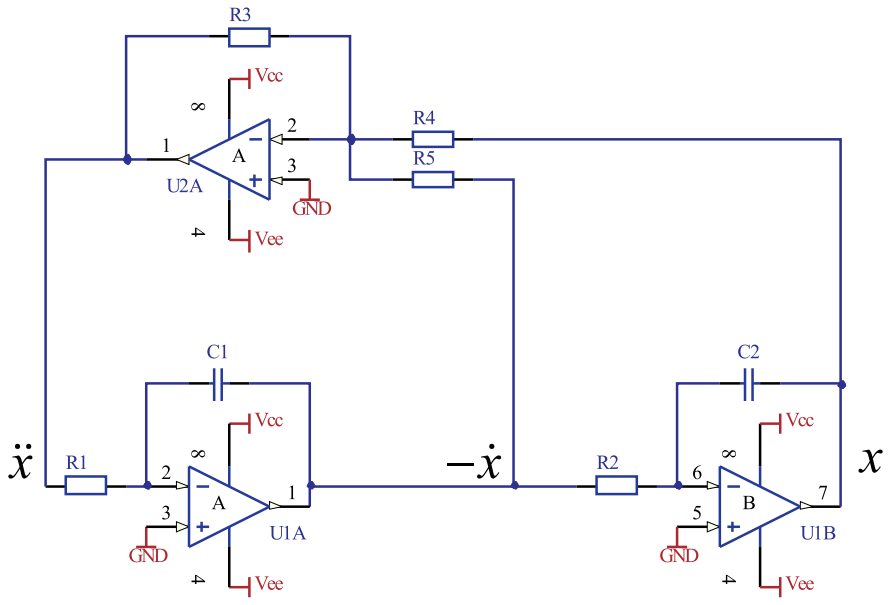
Iz blok sheme lahko sestavimo vezje, ki je prikazano na sliki 2.8. Vrednosti uporov $R3$, $R4$ in $R5$ izberemo tako, da se bosta napetosti seštevali s faktorjema A oz. B, vrednosti ostalih elementov pa tako, da bo $R1 \cdot C1 = 1ms$ in $R2 \cdot C2 = 1ms$.



Slika 2.7: Blok shema za enačbo 2.8: (a) neposredno izpeljana blok shema, (b) preurejena blok shema za implementacijo.

Vezje je harmonični oscilator, katerega amplituda se bo počasi večala, dokler se ne bo približala napajalni napetosti operacijskih ojačevalcev. Od tedaj naprej bo amplituda stabilna, v nihanju pa bo prisotno manjše popačenje.

Ko smo iskali rešitev enačbe, nismo povedali obeh začetnih pogojev. Če bi pogledali vezje ob vklopu, bi dobili začetna pogoja $x = 0$ in $\dot{x} = 0$. S tema pogojema pa bi se rešitev enačbe glasila $x(t) = 0$. Vezje bi torej mirovalo v nestabilnem ravnotežnem stanju. V praksi ne moremo zagotoviti takšnega mirovanja sistema, Zato bo vezje delovalo.



Slika 2.8: Električna shema za realizacijo enačbe 2.8

3 Generatorji naključnih števil

Generator naključnih števil je fizikalni ali pa računalniški sistem, ki generira zaporedje števil, ki bo izgledalo naključno. Delimo jih na "resnične" generatorje naključnih števil in na generatorje psevdonaključnih ali kvazinaključnih števil.

"Resnični" generatorji naključnih števil uporabljano nek fizikalni, nedeterministični vir entropije in postopek procesiranja za tvorjenje naključnih števil. Viri entropije so lahko šum v električnih vezjih, pojavi v polprevodnikih, časi med uporabnikovimi pritiski tipk na tipkovnici ali kombinacije teh pojavov. Če hočemo fizikalne generatorje naključnih števil neposredno uporabljati kot vir naključnih števil, morajo ti prestati vrsto statističnih testov, ki nam povedo ali so števila res porazdeljena enakomerno in naključno.

Obstajajo generatorji naključnih števil, ki uporabljajo kot vir entropije termični šum, tranzistorski šum, radioaktivne razpade ali kvantne pojave [17].

Pri fizikalnih generatorjih naključnih števil imamo večkrat težavo, da ti nimajo enakomerne porazdelitve tvorjenih števil (ang. bias). To pomeni, da ničle in enice niso enakomerno porazdeljene. To neenakomernost lahko zmanjšamo s tem, da uporabimo operacijo ekskluzivne disjunkcije (ang. exclusive or), ki jo na kratko označujemo XOR ali s simbolom $\otimes$.

Imejmo dva vira naključnih števil, ki nam dajeta zaporedje ničel in enic. Tema viroma priredimo naključni spremenljivki X in Y. Naključna spremenljivka naj zavzame številsko vrednost 0, če nam vir da bit 0 in številsko vrednost 1, če nam vir da bit 1. Za vsako od teh naključnih spremenljivk lahko izračunamo matematično upanje

$$\begin{aligned} E(X) &= 1 \cdot p + 0 \cdot (1-p) = p, \\ E(Y) &= 1 \cdot q + 0 \cdot (1-q) = q, \end{aligned} \tag{3.1}$$

kjer je p verjetnost, da nam da prvi vir bit 1, q pa verjetnost, da nam da drugi vir bit 1. Vpeljimo novo naključno spremenljivko $X \otimes Y$, ki naj zavzame številsko vrednost vrednost 1, ko bo natanko ena od spremenljivk X in Y imela vrednost 1 in vrednost 0 sicer. Tako bo naša nova spremenljivka ustrezala operaciji XOR med bitom iz prvega vira in bitom iz drugega vira.

Če imamo dva neodvisna naključna vira in s tem neodvisni naključni spremenljivki X in Y z matematičnima upanja $E(X) = \mu$ in $E(Y) = \sigma$, potem je

$$E(X \otimes Y) = \mu + \sigma - 2\mu\sigma = \frac{1}{2} - 2(\mu - \frac{1}{2})(\sigma - \frac{1}{2}). \tag{3.2}$$

Operacija XOR bo v primeru neodvisnih virov vedno zmanjšala neenakomernost. Če bosta matematični upanji spremenljivk blizu 1/2, potem bo matematično upanje za rezultat operacije XOR zelo blizu 1/2. Poglejmo primer, ko bo $\mu = \sigma = 0,6$. Potem bo $E(X \otimes Y) = 0,48$.

Učinek zmanjševanja neenakomernosti lahko povečamo, če operacijo XOR uporabimo večkrat. Kadar imamo n neodvisnih virov z enakim matematičnim upanjem μ, bo veljalo

$$E(X_1 \otimes X_2 \otimes \ldots \otimes X_n) = \frac{1}{2} + (-2)^{n-1}(\mu - \frac{1}{2})^n. \tag{3.3}$$

Poglejmo primer, ko bo $\mu = 0,6$ in $n = 6$. Potem bo $E(X_1 \otimes \ldots \otimes X_6) = 0,499968$.

Imejmo sedaj zaporedje neodvisnih naključnih bitov $\epsilon_1, \epsilon_2 \ldots \epsilon_n$, ki jim podobno kot prej priredimo naključne spremenljivke $X_1, X_2 \ldots X_n$. Naj bo matematično upanje spremenljivk $\mu \neq 1/2$. Neenakomernost lahko zmanjšamo tako, da tvorimo novo zaporedje $X_1 \otimes X_2, X_3 \otimes X_4, ...$ (slika 3.1a). Slaba stran te metode je, da dobimo kot rezultat zaporedje, ki je dvakrat krajše od prvotnega. Lahko pa tvorimo zaporedje primerljive dolžine na dva načina. Pri prvem tvorimo iz zaporedja $X_1, \ldots X_n$ zaporedje $Y_1, \ldots Y_n$ s predpisom: $Y_1 = X_1$ in $Y_i = X_i \otimes Y_i - 1$ za vsak $i \geq 2$ (slika 3.1b). V primeru velikega n zelo zmanjšamo neenakomernost, vendar dobimo med zaporednima bitoma korelacijo

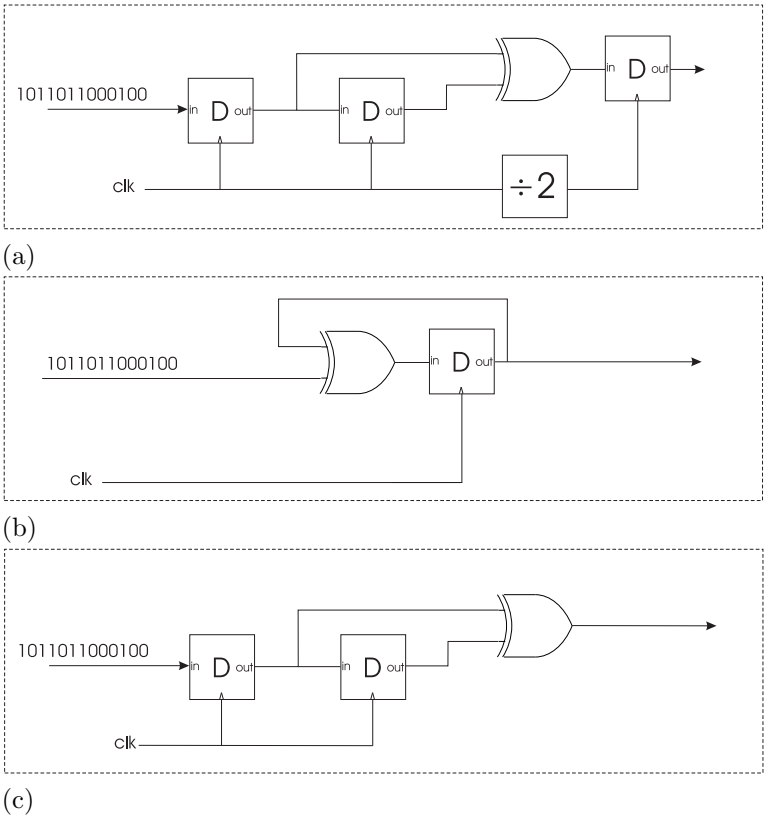
$$corr(Y_{i-1}, Y_i) = -2(\mu - \frac{1}{2}). \tag{3.4}$$

Tej korelaciji se lahko spet izognemo tako da uporabimo vsak drugi bit. Drugi način je, da tvorimo novo zaporedje $Y_2, \ldots Y_n$ s predpisom $Y_i = X_{i-1} \otimes X_i$ (slika 3.1c). Tudi pri tej metodi zmanjšamo neenakomernost, vendar dobimo korelacijo med biti v končnem zaporedju

$$corr(Y_{i-1}, Y_i) = \frac{2(\mu - \frac{1}{2})^2}{1 - 2\mu + 2\mu^2} \approx 4(\mu - \frac{1}{2})^2. \tag{3.5}$$

Korelacija je v tem primeru sicer manjša, vendar ne predstavlja vse odvisnosti med biti končnega zaporedja [3]. Sheme iz logičnih gradnikov za te tri metode so prikazane

na sliki 3.1.



Slika 3.1: Uporaba XOR za zmanjšanje neenakomernosti: (a) XOR na vsaka dva bita, (b) XOR iz trenutnega vhoda in prejšnjea izhoda, (c) XOR iz zadnjih dveh bitov.

Generatorji naključnih števil so relativno počasni. Kadar potrebujemo večje število naključnih podatkov, jih uporabljamo v kombinacijo z generatorji psevdonaključnih števil. Ti sprejmejo eno ali več števil kot vhodne podatke in dajo večjo količino psevdonaključnih števil. Podatke, ki jih sprejmejo imenujemo semena (ang. seeds). Vsi izhodi generatorja so natančno določeni s semeni, zato te generatorje imenuje psevdonaključni. Takšni generatorji sicer dajejo zaporedje števil, ki izgleda naključno, vendar imajo neko periodo, po kateri se števila začnejo ponavljati. Ker so vsa števila generatorja psevdonaključnih števil natančno določena s semenom in so algoritmi teh generatorjev znani, moramo v kriptografiji poskrbeti, da ostanejo semena tajna.

Primer psevdonaključnega generatorja je linearni kongruenčni generator (ang. linear congruential generator), ki uporablja rekurzivno formulo

$$X_{n+1} = (aX_n + b) \mod m \tag{3.6}$$

in lahko generira m različnih števil. Obstajajo različne posplošitve te metode, kot so

kvadratna kongruenčna metoda (ang. quadritoc congruential method). Drugi algoritmi so metoda sredine kvadrata (ang. middle square method), množenje s prenosom (ang. Multiply-with-carry), Couveyoujeva metoda, aditivni generator števil in drugi [7].

3.1 Testiranje ustreznosti generiranja naključnih števil

Generator deluje pravilno, če so števila, ki jih podaja porazdeljena enakomerno in so neodvisna. Za bitno zaporedje rečemo, da je porazdeljeno enakomerno, če se bita 0 in 1 pojavila z enako verjetnostjo. Neodvisnost pa pomeni, da je verjetnost, da se na nekem mestu pojavi 0 ali 1, neodvisna od prejšnjih bitov. V tem primeru bomo rekli, da je zaporedje bitov primerno. Za uporabo v kriptografiji mora biti zaporedje bitov nenapovedljivo.

Ali je neko zaporedje števil dobljenih iz generatorja naključnih števil zares primerno ne moremo z gotovostjo potrditi ali zavrniti. Lahko pa si pomagamo z različnimi testi statističnih hipotez.

Ustreznost generatorja naključnih števil preverjamo s statističnimi testi. V splošnem postavimo pri statističnih testih dve hipotezi. Prva je ničelna hipoteza. V našem primeru se bo ničelna hipoteza vedno glasila: števila so primerna. Druga pa je alternativna hipoteza, ki pove ravno nasprotno. Pri nas: števila niso primerna.

Pri vsakem statističnem testu izračunamo neko statistiko, ki je relevantna za našo hipotezo. Pod predpostavko, da velja ničelna hipoteza, bo imela ta statistika neko referenčno porazdelitev. Za to porazdelitev določimo kritično območje. To so tiste vrednosti statistike, v katerega teoretično pade nek delež najbolj ekstremnih vrednosti statistike. To so tiste vrednosti spremenljivke, ki jih testna statistika zavzame, ko zaporedje izgleda najmanj primerno. Ko imamo vzorec, na njem izračunamo testno statistiko. Kadar je vrednost testne statistike v kritičnem območju, zavrnemo ničelno hipotezo, sicer pa je ne zavrnemo. To še ne pomeni, da jo sprejmemo, ampak da se na podlagi tega testa še nismo odločili.

Pri testiranju statističnih hipotez lahko napravimo dve napaki. V primeru pravilne ničelne hipoteze lahko napačno sprejmemo alternativno hipotezo oz. da napačno zavrnemo ničelno hipotezo. Tedaj napravimo napako I. vrste. Če pa v primeru pravilne alternativne hipoteze napačno sprejmemo ničelno hipotezo, napravimo napako II. vrste. Verjetnost napake I. vrste imenujemo tudi stopnja signifikance statistike α. Običajno se v kripografiji uporabljajo vrednost od $\alpha = 0,001$ do $\alpha = 0,01$ [9]. V našem primeru bomo uporabljali $\alpha = 0,01$.

Za testiranje naključnih števil obstaja več različnih testov. Pet osnovnih testov je: frekvenčni test, serijski test, poker test, iteracijski test in avtokorelacijski test [8].

Ameriški Nacionalni inštitut za standarde in tehnologijo je izdal zbirko statističnih testov za preverjanje generatorjev naključnih števil. V njej je 15 različnih testov, s katerimi preverjamo naključnost bitnega zaporedja. Spodaj so našteti testi, ki so vključeni v to zbirko.

1. **Frekvenčni test** ali **monobitni test** (ang. Frequency test). Ta test je namenjen testiranju relativne frekvence ničel in enic v celotnem zaporedju. Ob predpostavki naključnega zaporedja bo relativna frekvenca enic blizu 1/2. Test preveri ali je odstopanje relativne frekvence znotraj dopustnih mej.

2. **Frekvenčni test znotraj bloka** (ang. Frequency test within a block). Pri tem testu celotno zaporedje razdelimo na več manjših blokov, v katerih testiramo relativne frekvence enic.

3. **Iteracijski test** (ang. Runs test). Pri tem testu preštejemo celotno število iteracij v podanem zaporedju. Iteracija je neprekinjeno podzaporedje enakih bitov. S testom preverimo ali je število iteracij v podanem zaporedju znotraj mej, ki jih pričakujemo pri naključnem zaporedju. S tem preverimo ali se dva zaporedna bita razlikujeta preveč ali premalokrat.

4. **Test za najdaljšo iteracijo enic v bloku** (ang. Test for the longest run of ones in a block). Pri tem testu zaporedje spet razdelimo na bloke in preverjamo kolikokrat se v blokih pojavljajo dolga zaporedja enic.

5. **Test ranga binarne matrike** (ang. Binary matrix rank test). Zaporedje spet razdelimo na bloke in bite iz posameznih blokov zapišemo v kvadratne matrike. Določimo koliko matrik ima binarni rang enak številu vrstic v matriki, koliko matrik ima binarni rang za ena manjši in koliko je vseh ostalih matrik. Števila matrik primerjamo s pričakovanimi rezultati za naključno zaporedje. S tem testom preverjamo morebitne linearne odvisnosti med podzaporedji fiksne dolžine.

6. **Test z diskretno Fourierevo transformacijo** ali **spektralni test** (ang. Discrete Fourier transform test). S tem testom preverjamo prisotnost periodičnih komponent v testiranem zaporedju. Preverjamo absolutne vrednosti v diskretni Fourierevi transformiranki zaporedja. Ob predpostavki naključnega zaporedja bodo te vrednosti pod neko kritično vrednostjo.

7. **Test ujemanja s predlogami brez prekrivanja** (ang. Non-overlapping Template Matching Test). Namen tega testa je ugotoviti ali se v zaporedju prevečkrat nahajajo določena krajša zaporedja fiksnih dolžin. Pri tem testu uporabljamo okno, dolžine predloge, ki ga premikamo po en bit skozi zaporedje. Kadar najdemo neko podzaporedje bitov, ki je enako predlogi se pomaknemo naprej za dolžino celotenega okna ozirmoa predloge.

8. **Test ujemanja s predlogami s prekrivanjem** (ang. Overlapping Template Matching Test). Ta test je podoben prejšnjemu, le da se v vsakem primeru pomikamo z oknom za en bit naprej.

9. **Maurerjev "univerzalni statistični" test** (ang. Maurer's "universal statistical" test). Preverjamo število bitov med dvema enakima vzorcema v zaporedju. S tem ugotovimo ali je možno zaporedje opazno brezizgubno stisniti, kar bi pomenilo, da zaporedje ni primerno.

10. **Test linearne kompleksnosti** (ang. Linear complexity test). S tem testom preverjamo dolžino pomičnega registra z linearno povratno vezavo. Ugotovimo ali je zaporedje dovolj kompleksno, da ga lahko smatramo kot naključno.

11. **Serijski test** (ang. Serial test). S tem testom preverimo frekvenco vseh možnih prekrivajočih se bitnih vzorcev določene dolžine. S tem preverimo ali se vsi vzorci pojavijo s pričakovano verjetnostjo.

12. **Test približne entropije** (ang. Approximate entropy test). Podobno kot v prejšnjem testu preverjamo frekvence določenih vzorcev. Tukaj pa primerjamo frekvenco dveh prekrivajočih se blokov zaporednih dolžin s pričakovano frekvenco.

13. **Test kumulativnih vsot** (ang. Cumulative sums test). S tem testom preverimo kako se z naključnim sprehodom, določenim z delno vsoto bitnega zaporedja, oddaljujemo od srednje vrednosti. Pri tem pretvorimo bite 0 in 1 v števila -1 in 1.

14. **Test naključnega oddaljevanja** (ang. Random excursion test). Preverjamo število ciklov, ki se pojavijo z določeno pogostostjo v naključnem sprehodu.

15. **Varianta testa naključnega oddaljevanja** (ang. Random excursion test). Pri tem testu pa preverjamo kolikokrat je bilo katero število obiskano v naključnem sprehodu.

Navedene teste bomo uporabljali v nadaljevanju za preverjanje delovanja našega generatorja naključnih števil. Podrobnejši opis testov najdemo v [9].

Vsak od teh testov nam kot končni rezultat vrne P-vrednost. To je verjetnost, da je idealen generator naključnih števil tvoril zaporedje, na katerem bi dobili enako ali slabšo vrednost testne statistike. Če je P-vrednost enaka 1 potem izgleda zaporedje popolnoma primerno, če pa je P-vrednost enaka 0, potem izgleda popolnoma neprimerno. Za vse teste si izberemo stopnjo signifikance, običajno $\alpha = 0,01$. Kadar bo $P < \alpha$ bomo ničelni hipotezo zavrnili. V nasprotnem primeru bomo rekli, da je zaporedje uspešno prestalo test. Če bomo v vseh testih dobili vrednosti $P > \alpha$, bomo rekli, da je zaporedje primerno.

Za primer si podrobneje poglejmo delovanje frekvenčnega testa. Da zaporedje prestane ta test je osnovni pogoj, da lahko izvajamo naslednje teste.

Teste bomo izvajali na dva načina. V prvem – enostavnem načinu imamo eno testno zaporedje, ki ga testiramo. Iz vhodnega zaporedja tvorimo novo zaporedje s številskimi vrednostmi -1 in 1. Nato izračunamo vsoto novega zaporedja in jo normiramo s korenom dolžine zaporedja.

Naj bo ϵ testno zaporedje dolžine n. Iz njega tvorimo zaporedje $X = 2\epsilon - 1$, izračunamo njegovo vsoto $S_n = X_1 + X_2 + \ldots + x_n$ in testno statistiko $s = \frac{S_n}{\sqrt{n}}$. Ob predpostavki naključnega zaporedja in velikega števila n, se bo po DeMoivre-Laplaceovem izreku porazdelitev testne statistike prilegala standardizirani normalni porazdelitvi.

Za pravilno delovanje testa rabimo dolgo zaporedje. Za enostavnejšo ponazoritev delovanja testa, bomo prikazali njegovo delovanje na zaporedju, ki je dolgo le 10 bitov. Naj bo $\epsilon = 1011010101$ naše testno zaporedje. Potem je $n = 10$, $S_n = 2$ in $s = \frac{|S_n|}{\sqrt{n}} = 0,632455532$. P-vrednost izračunamo s pomočjo komplementarne funkcije napake (ang. complementary error function):

$$P = erfc\left(\frac{s}{\sqrt{2}}\right) = \frac{2}{\sqrt{\pi}} \int_{s/\sqrt{2}}^{\infty} e^{-u^2} du = 0,527089. \tag{3.7}$$

Ker je P-vrednost večja od $\alpha = 0,01$, rečemo da je zaporedje prestalo test na stopji signifikance $0,01$.

Za boljšo zanesljivost pa teste izvajamo na drug – sestavljen način. Pri tem vsak test izvedemo tako, da podatke iz generatorja naključnih števil razdelimo na več blokov in testiramo vsak blok podatkov za sebe. Testiramo jih tako kot v enostavnem način. Ker smo izbrali neko stopnjo signifikance, pričakujemo, da primerno zaporedje v nekem številu blokov ne bo prestalo testa. Če smo izbrali $\alpha = 0,01$, potem pričakujemo, da pri posameznem testu približno 1% blokov ne bo prestalo testa. Zrave tega deleža bomo

prevarjali tudi ali so P-vrednosti, ki smo jih dobili na posameznih blokih porazdeljene enakomerno.

V primeru da smo sprejeli ničelno hipotezo, še vedno ne moremo z gotovostjo trditi, da so števila res primerna. Trdimo le, da s statističnimi testi na našem vzorcu nismo odkrili znakov, ki bi kazali na nasprotno. Če pa smo sprejeli alternativno hipotezo, pa tudi ne moremo z gotovostjo trdi, da števila niso primerna. Če smo odkrili kak vzorec v številih, je to res lahko slabost generatorja, lahko pa je tudi slučajni pojav v našem vzorcu. Ravno vplivu teh slučajnih pojavov se izognemo s tem, da izvedemo teste na več blokih podatkov [9].

3.2 Kaotični oscilatorji primerni za generator naključnih števil

Naš namen je načrtovati generator naključnih števil, ki temelji na kaotičnem oscilatorju. Hočemo sestavi čim enostavnejše vezje, katerega bo mogoče skalirati na obliko primerno za integracijo. To pomeni, da mora biti sestavljeno iz elementov, ki jih je mogoče izdelati tudi v integrirani tehnologiji, in da mora imeti čim manjšo porabo.

Dinamična elektronska vezja vsebujejo kondenzatorje in tuljave. Ker je tuljavo težavno vključiti v integrirano vezje iščemo kaotično vezje, ki ne vsebuje tuljav. Hočemo tudi vezje, ki vsebuje čim manj operacijskih ojačevalnikov in podobnih elementov.

Drugi del vezja bo predstavljal A/D pretvorbo, s katero bomo generirali števila. Pri tem se pojavita dve ideji. Opazujemo lahko napetost v neki točki vezja. S pomočjo komparatorja jo primerjamo z neko fiksno napetostjo. Ena možnost je, da v trenutku, ko se spremeni izhod komparatorja, preberemo napetost v neki drugi točki vezja s pomočjo analogno-digitalnega pretvornika. Druga možnost je, da s pomočjo števca merimo čas med dvema ali več spremembami na izhodu komparatorja.

Za neko vezje lahko narišemo atraktor v ravnini dveh napetosti v vezju. Iščemo vezje, ki bo imelo "širok" atraktor. Za uporabo analogno-digitalnega pretvornika si želimo, da se bo napetost gibala v čim širšem območju. Podobno velja za uporabo števca.

Chujev oscilator za nas ni primeren, ker vsebuje tuljavo. Da bo vezje enostavno, iščemo oscilator, ki bo imelo tudi čim manj analognih množilnikov. Iščemo sistem s čim manj "kvadratnimi členi". To so členi ki jih zapišemo kot kvadrat neke spremenljivke stanja ali pa produkt dveh spremenljivk stanja. Lorenzev sistem vsebuje dva kvadratna

člena, za implementacijo katerih bi rabili množilnik. Rösslerjev pa še vedno enega. J. C. Sprott je leta 1994 odkril še 19 sistemov z dvema ali pa enim kvadratnim členom. S. J. Linz in J. C. Sprott pa sta 1999 prestavil vrsto kaotičnih sistemov brez kvadratnih členov.

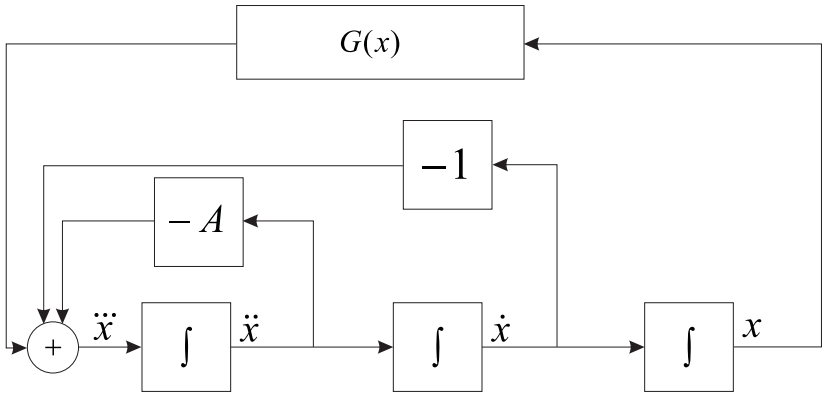
Izmed predstavljenih sistemov lahko družino štirih izmed njih opišemo z diferencialno enačbo tretjega reda

$$\dddot{x} = -A\ddot{x} - \dot{x} + G(x), \tag{3.8}$$

kjer je A konstanta enaka ali rahlo večja od $0,6$, $G(x)$ pa ena izmed naslednjih funkcij:

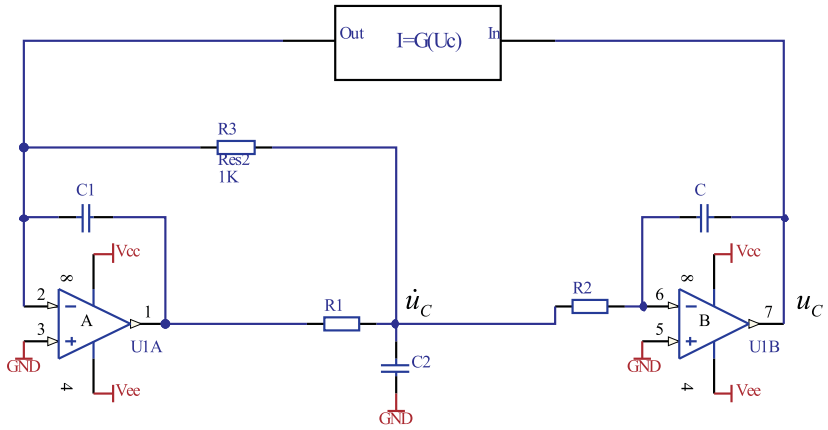
1. $G(x) = |x| - 2$,
2. $G(x) = -6 \cdot \max\{x, 0\} + 0,5$,
3. $G(x) = 1,2x - 4,5 \cdot \mathrm{sign}(x)$,
4. $G(x) = -1,2x + 2 \cdot \mathrm{sign}(x)$.

Ostale sisteme lahko najdemo v [12]. Prednost teh enačb je, da jo lahko napravimo sintezo vezja samo z uporabo uporov, kondenzatorjev, uporov, diod in operacijskih ojačevalnikov. Na sliki 3.2 je blok shema vezja, ki ustreza enačbi 3.8.

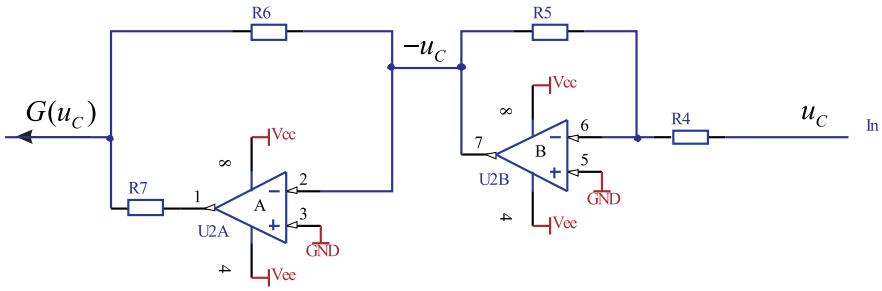


Slika 3.2: Blok shema vezja za enačbo 3.8

Poenostavljena realizacija osnovnega vezja, ki ustreza enačbi 3.8 je na sliki 3.3. Za poenostavitev vezja smo srednji integrator realizirali z RC-členom. Ker uporabljamo preproste integratorje, katerih izhod je integral nasprotne vrednosti vhoda, se v vezju pojavijo napetosti, ki predstavljajo $-\dot{x}$ in $-\ddot{x}$, namesto $\dot{x}$ in $\ddot{x}$. Vrednost $\dddot{x}$ je v vezju realizirana s tokom, ki teče v kondenzator $C1$.



Slika 3.3: Poenostavljeno osnovno vezje za realizacijo enačbe 3.8

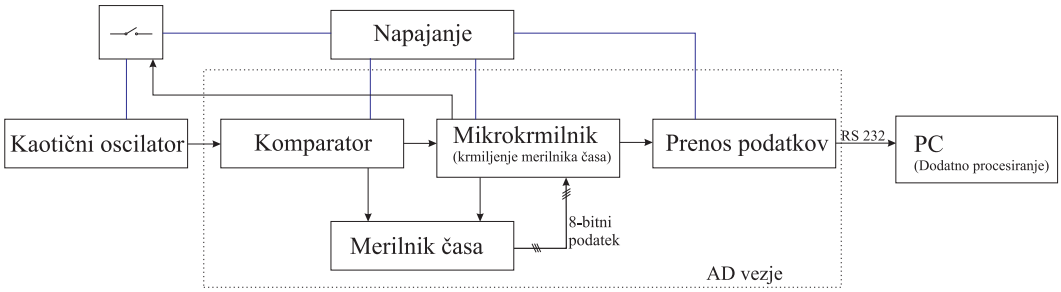


Slika 3.4: Realizacija funkcije $G(x) = Bx + C \cdot \mathrm{sign}(x)$

$G(x)$ predstavlja nelinearni upor. Na sliki 3.4 je predstavljena njegova realizacija za enačbo $G(x) = Bx + C \cdot \mathrm{sign}(x)$. Desni del vezja je invertirajoči ojačevalnik, ki nam da napetost $-x$. Levi del s komparatorjem pa predstavlja funkcijo predznaka [11]. Realizacije ostalih funkcij lahko najdemo v [11].

4 Načrtovanje in analiza kaotičnega generatorja naključnih števil

Za preizkušanje delovanja vezja, smo sestavili prototip na eksperimentalni oziroma rastrski ploščici. Vezje je sestavljeno iz: kaotičnega oscilatorja, komparatorja, merilnika časa, mikrokrmilnika in vezja za prenos števil na osebni računalnik. Blok shema vezja je prikazana na sliki 4.1.



Slika 4.1: Blok shema generatorja naključnih števil

4.1 Kaotični oscilator

Pri izdelavi kaotičnega oscilatorja smo se odločili za dve različici, ki jih je predlagal Sprott [12]. Prvi oscilator opisuje enačba

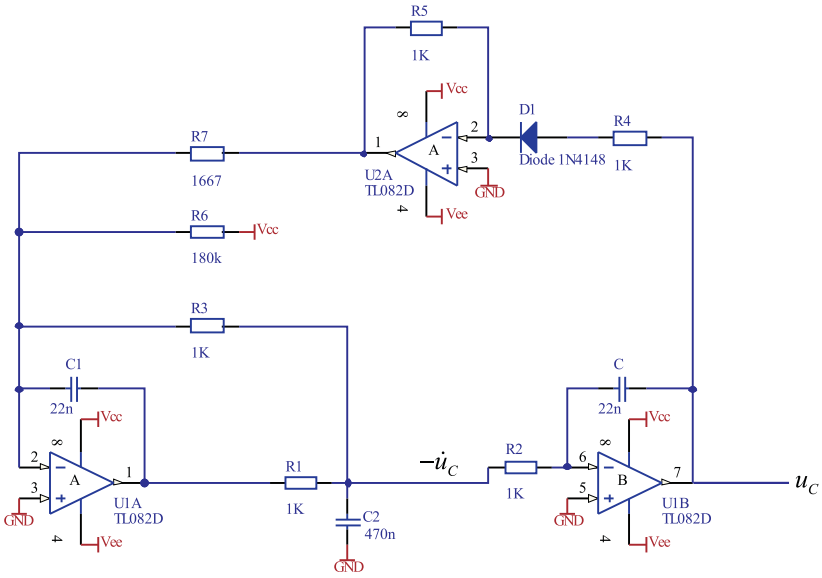
$$\dddot{x} = -A \cdot \ddot{x} - \dot{x} - B \cdot \max\{x, 0\} + C, \tag{4.1}$$

kjer so konstante $A = 0,6$, $B = 6$ in $C = 0,5$. Njegova poenostavljena shema je prikazana na sliki 4.2. Drugi oscilator pa opisuje enačba

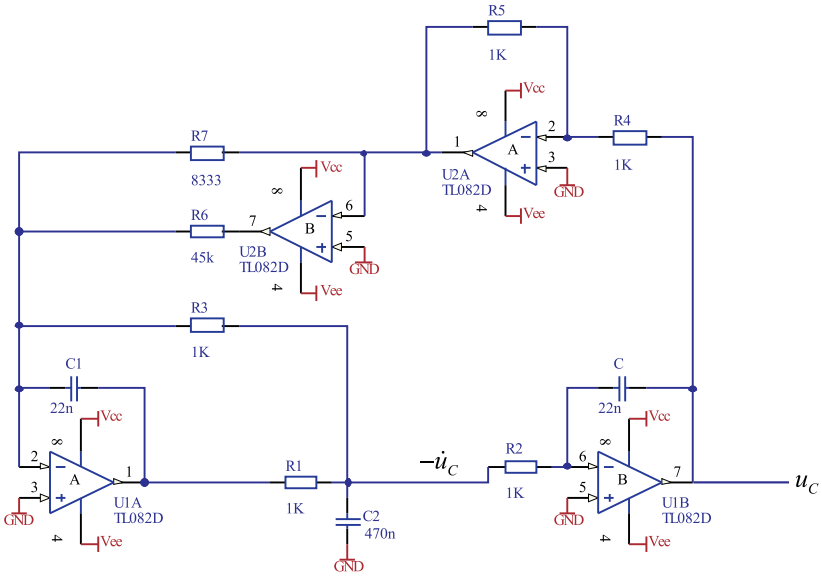
$$\dddot{x} = -A \cdot \ddot{x} - \dot{x} - B \cdot x + C \cdot \text{sign}(x), \tag{4.2}$$

kjer so konstante $A = 0,6$, $B = 1,2$ in $C = 2$. Njegova poenostavljena shema pa je prikazana na sliki 4.3. V obeh primerih je bil najprej odkrit kaotični sistem v obliki enačbe in potem realiziran v obliki elektronskega vezja.

Za operacijski ojačevalniki smo v obeh primerih izbrali običajni splošnonamenski ojačevalniki tipa $TL082$. Konstanti B in C v enačbah sta neposredno povezani z



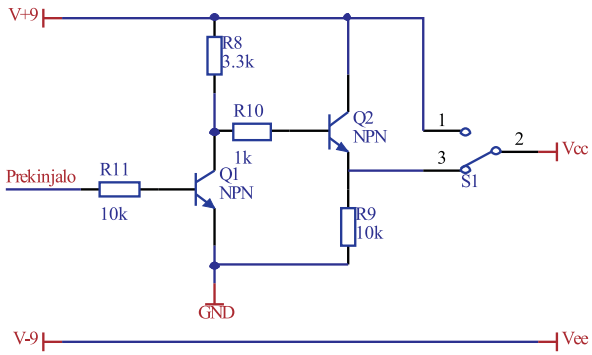
Slika 4.2: Poenostavljena realizacija enačbe 4.1



Slika 4.3: Poenostavljena realizacija enačbe 4.2

uporoma $R6$ in $R7$, zato sta vrednosti teh dveh uporov tako neobičajni. V praktični izvedbi je vezje sestavljeno z nastavljivima uporoma. Vezje je napajano s simetrično napetostjo $\pm 9V$.

Napajanje kaotičnega oscilatorja med delovanjem prekinjamo, da lahko preizkusimo situacijo, ko generator naključnih števil deluje le ob vklopu napajanja in se pozneje izklopi, dokler ga ne poženemo ponovno. Napajanje smo prekinjali le na pozitivni veji, saj s tem že dosežemo prenehanje delovanja oscilatorja. Vezje s katerim prekinjamo napajanje je na sliki 4.4. Stikalo $S1$ nam dopušča, da lahko vezje deluje tudi z neprekinjenim napajanjem. Z *V+9* in *V-9* so označene točke v vezju, ki so



Slika 4.4: Vezje za prekinjanje napajanja kaotičnega oscilatorja

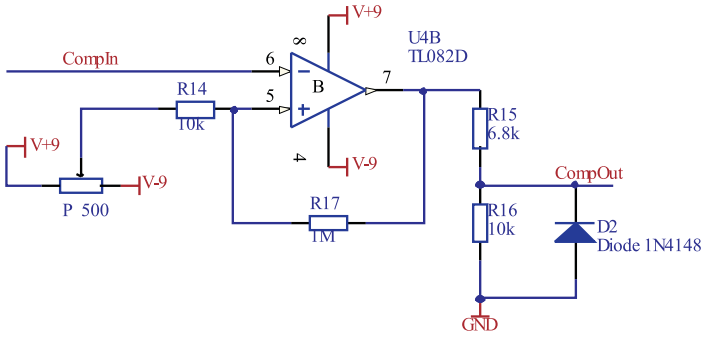
neposredno priključene na napajalnik, z *Vcc* in *Vee* pa napajanje za kaotični oscilator. Ker negativnega napajanja ne prekinjamo, sta *V-9* in *Vee* povezana.

4.2 AD vezje

Naš oscilator je analogno vezje, kot rezultat generatorja pa potrebujemo bitno zaporedje. Zato rabimo vezje, s katerih bomo obnašanje vezja pretvorili v neko digitalno obliko.

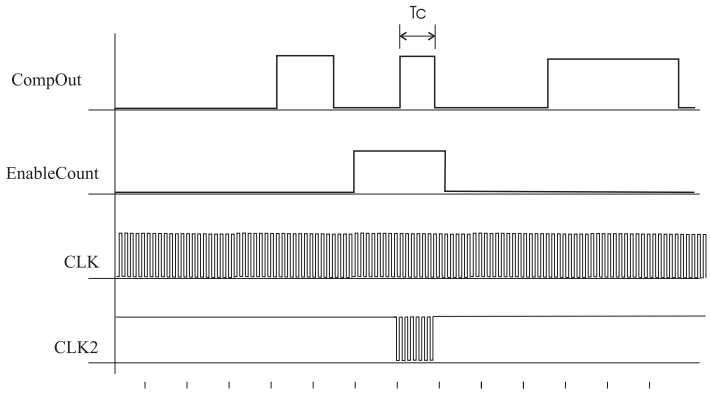
Komparator

Med delovanjem vezja opazujemo neko napetost v njem. V našem primeru smo se odločili, da opazujemo napetost U_C. Opazujemo jo s komparatorjem s histerezo, prikazanim na sliki 4.5. Opazovano napetost vodimo na negativni vhod komparatorja. Delilec napetost in dioda na izhodu poskrbita za ustrezne napetosti na izhodu, da ga lahko pripeljemo na mikrokrmilnik. Komparator s histerezo smo izbrali, ker smo med poskušanjem ugotovili, da prihaja do motenj v delovanju običajnega komparatorja. S potenciometrom smo nastavili na pozitivni vhod napetost blizu ničle.



Slika 4.5: Komparator s histerezo

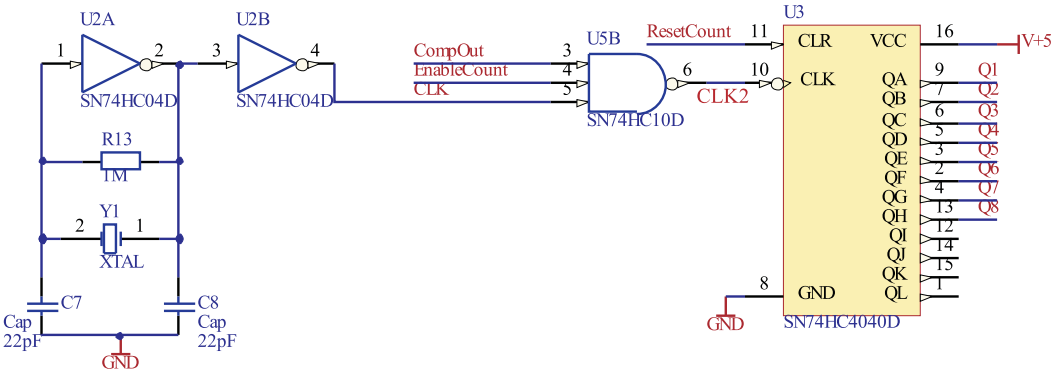
Izhod iz komparatorja nam torej pove ali je opazovana napetost v oscilatorju trenutno nad ali pod neko vrednostjo. Naš namen je bil s števcem izmeriti čas, ko je napetost nad to vrednostjo. Da smo lahko merili dolžino impulzov na izhodu komparatorja, smo morali poskrbeti, da bo signal za omogočitev števca prej na visokem nivoju kot pa izhod iz komparatorja in pozneje na nižjem nivoju. Potek signalov je prikazan na sliki 4.6. Na primeru slike bomo merili dolžino srednjega impulza v signalu *CompOut*. Iz števila impulzov v signalu *CLK2* tvorimo naključno število.



Slika 4.6: Tipični potek signalov *EnableCount*, *CompOut*, *CLK* in *CLK2*

Merilnik časa

Merjenje dolžine impulza nam omogoča vezje, ki je prikazan na sliki 4.7. V vezju merilnika časa imamo oscilator s kvarčnim kristalom. Med testiranjem vezja smo preizkusili delovanje z dvema različnima kristaloma: z *8Mhz* in z *2,4576MHz*. Delovanje števca krmilijo negirana IN vrata. Števec bo štel impulze iz oscilatorja kadar bosta tako izhod iz komparatorja kot tudi signal za omogočitev štetja iz mikrokrmilnika na visokem nivoju.

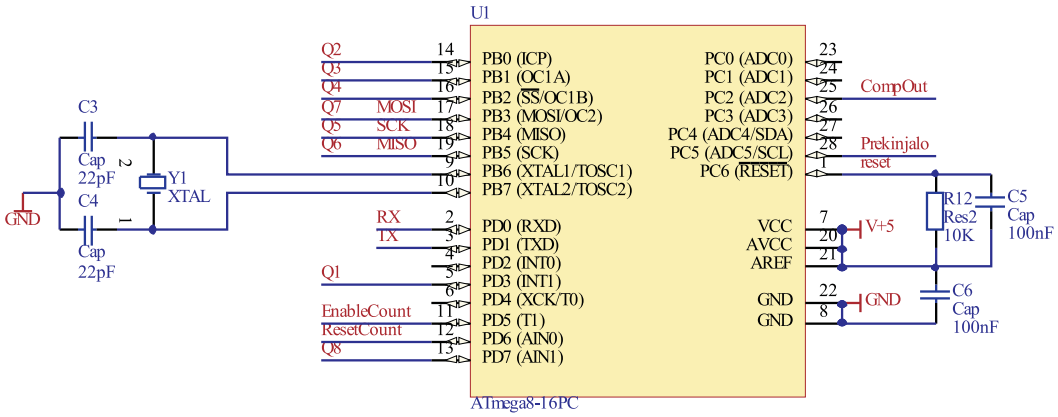


Slika 4.7: Vezje za merjenje časa

S signalom *EnableCount* omogočimo, da bo števec deloval, s signalom *ResetCount* pa resetiramo števec. Opazujemo le spodnjih 8 bitov števca - izhodi *Q1* do *Q8*.

Mikrokrmilnik

Celotno vezje krmili mikrokrmilnik tipa ATMEL AtMega 8. Priključitev mikrokrmilnika je prikazana na sliki 4.8.



Slika 4.8: Mikrokrmilnik

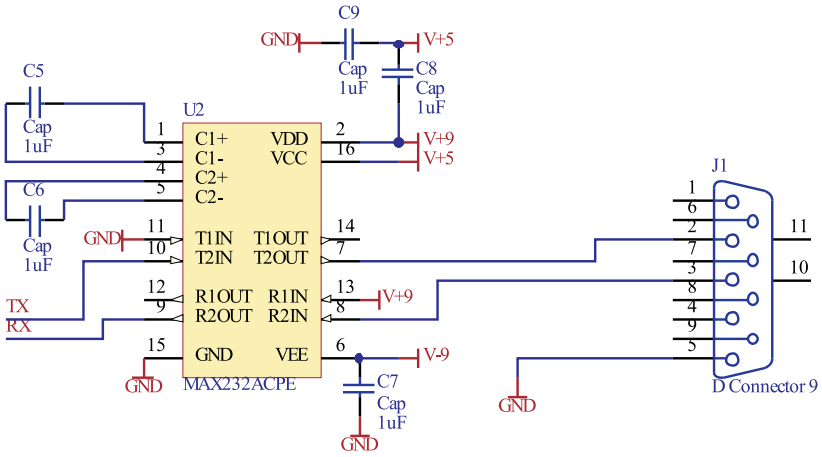
Signali *MOSI*, *MISO*, *SCK* in *reset* se uporabljajo pri programiranju mikrkrmilnika s pomočjo vmesnika *ISP*. Signali *Q1* do *Q8* in *CompOut* so vhodi v mikrokrmilnik, signali *Prekinjalo*, *EnableCount* in *ResetCount* pa izhodi. Ker so nekateri priključki uporabljeni za programiranje in za branje podatkov iz števca, moramo med programiranjem števec odstraniti iz vezja. Signala *RX* in *TX* se uporabljata za asinhrono komunikacijo.

Vezje za prenos števil

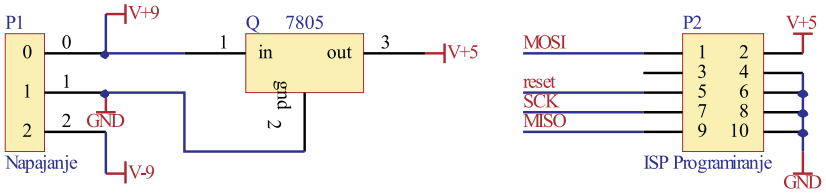
Števila pošiljamo na osebni računalnik po vodilu *RS232*. Mikrokrmilnik ima integriran vmesnik za pošiljanje podatkov, vendar zahteva standard *RS232* drugačne napetostne nivoje, kot pa mikrokrmilnik. Te pretvorimo z vezjem na sliki 4.9, ki temelji na integriranem vezju *MAX232*.

Ostali deli vezja

Zraven že naštetih komponent so v vezju še stabilizator za napajanje integriranih vezij in priključek za programiranje, ki sta prikazana na sliki 4.10.



Slika 4.9: Vezje za pretvorbo napetostnih nivojev



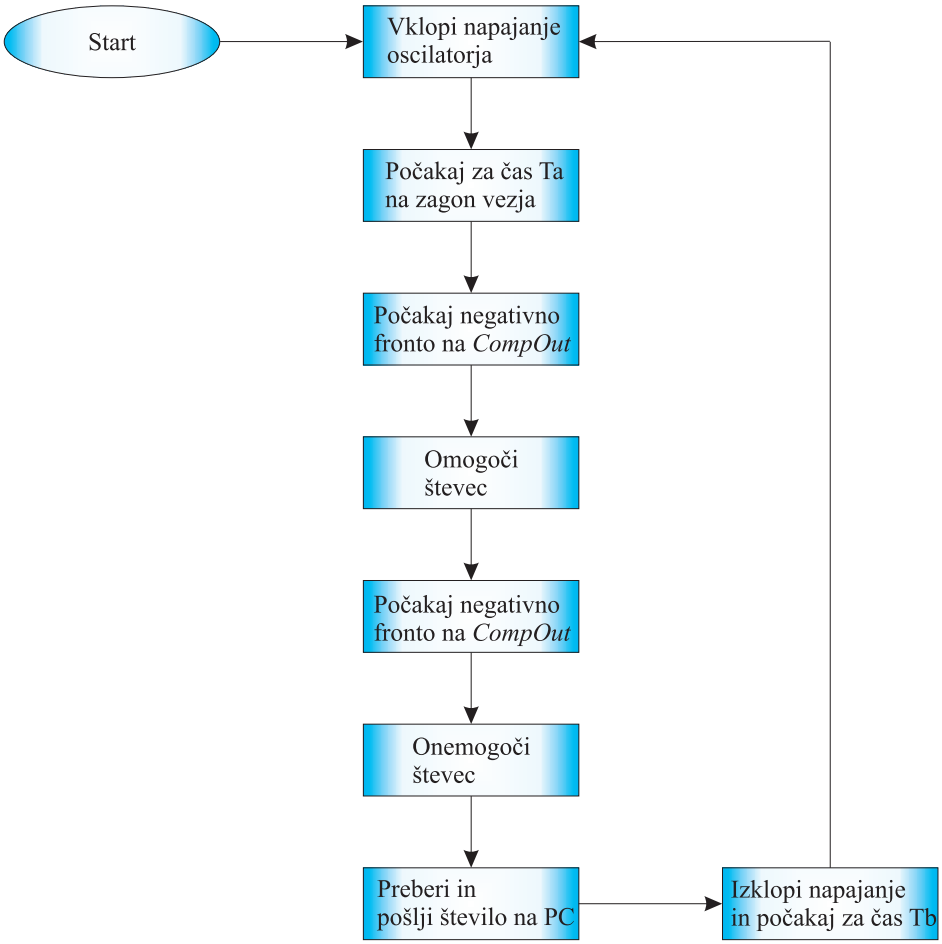
Slika 4.10: Stabilizator in priključek za programator

4.3 Program mikrokrmilnika

AD vezje je zasnovano tako, da je naloga mikrokrmilnika čim bolj preprosta. Ta mora poskrbeti za to, da omogoči napajanje oscilatorja, resetira števec, ga ob pravem trenutku omogoči in onemogoči ter prebere število s števca.

Ustrezno je potrebno določiti čas, ko vklopimo in izklopimo signal *EnableCount*. Obe spremembi se morata zgoditi, ko bo signal CompOut na nizkem nivojo. Najenostavneje je, da spremenimo signal *EnableCount* takoj po negatovni fronti na signalu *CompOut*. Diagram poteka delovanja programa je prikazan na sliki 4.11.

Ob zagonu programa mikrokrmilnik vklopi napajanje kaotičnega oscilatorja. Nato počaka za čas $T_a = 10ms$, da ta začne oscilirati. Medtem tudi resetiramo števec. Ko zazna naslednjo negativno fronto na izhodu komparatorja, omogoči števec in ob naslednji negativni fronti ga spet onemogoči. Tako poskrbimo, da bo dolžina impulza pravilno izmerjena. Prebrano število s števca pošljemo na osebni računalnik. Medtem spet izklopimo napajanje kaotičnega oscilatorja za čas T_b in počamo, da se vsi tokovi padejo na nič, preden začnemo naslednjo mejenje.

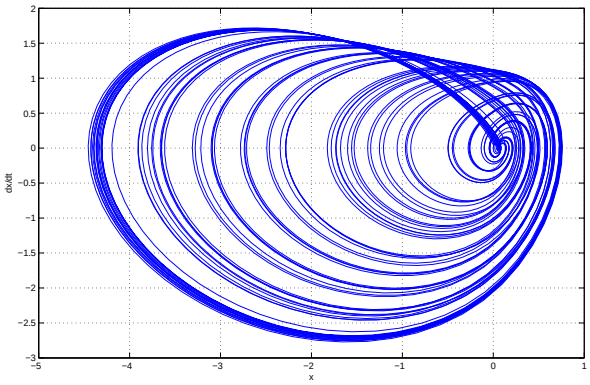


Slika 4.11: Diagram poteka programa mikrokrmilnika

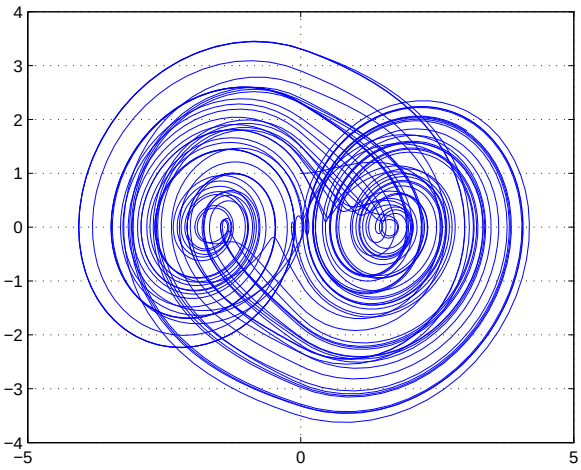
4.4 Analiza s simulacijo

Vezje smo preiskusili v dveh različicah, ki jih opisujeta enačbi 4.1 in 4.2. Obnašanje teh dveh enačb smo najprej preverili s programov MATLAB, s katerim narišemo atraktorja v ravnini x–$\dot{x}$ za ti dve enačbi. Prikazana sta na slikah 4.12 in 4.13. Obnašanje vezja smo potem preverili s simulacijo z računalniškim programom SPICE. Na slikah 4.14 in 4.15 sta prikazana atraktorja za oscilatorja na slikah 4.2 in 4.3, dobljena z računalniško simulacijo vezja. Ker imamo v vezju integratorje, ki spremenijo predznak, moramo ordinatno os zrcaliti. Vidna je podobnost med atraktorji iz simulacije sistema podanega z enačbo in simulacijo vezja.

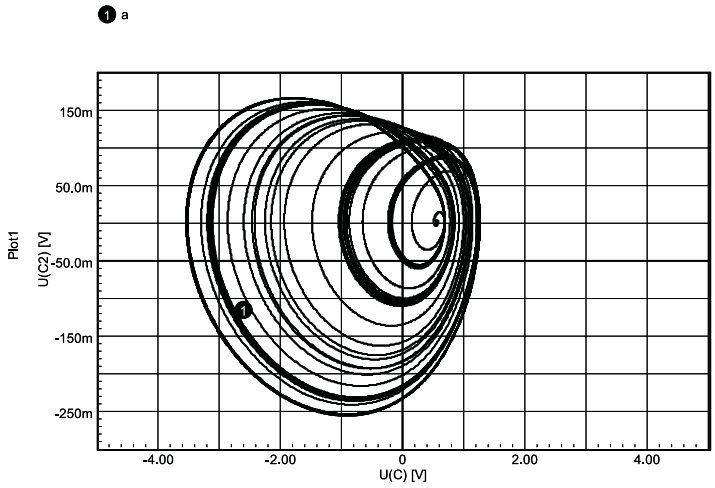
Na sliki 4.16 je prikazan potek napetosti na kondenzatorju C v oscilatorju na sliki 4.3. Vklop napajanja je ob času 0. Na sliki je lepo vidno, da rabi oscilator za zagon le nekaj milisekund. Hiperobčutljivost smo s pomočjo simulacije prikazali na sliki 4.17. Prikazan je potek napetosti na kondenzatorju C za različne napajalne napetosti v območju od $8,8V$ do $9,2V$ v korakih po $0,1V$. Vidna je podobnost potekov na začetku



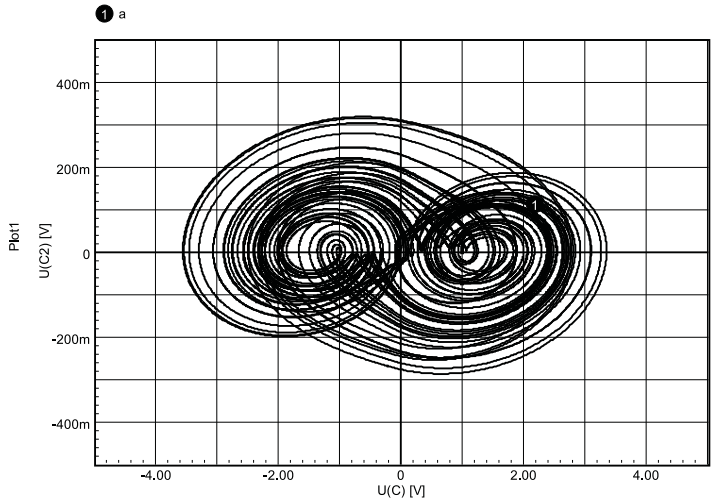
Slika 4.12: Atraktor enačbe 4.1 – MATLAB simulacija



Slika 4.13: Atraktor enačbe 4.2 – MATLAB simulacija

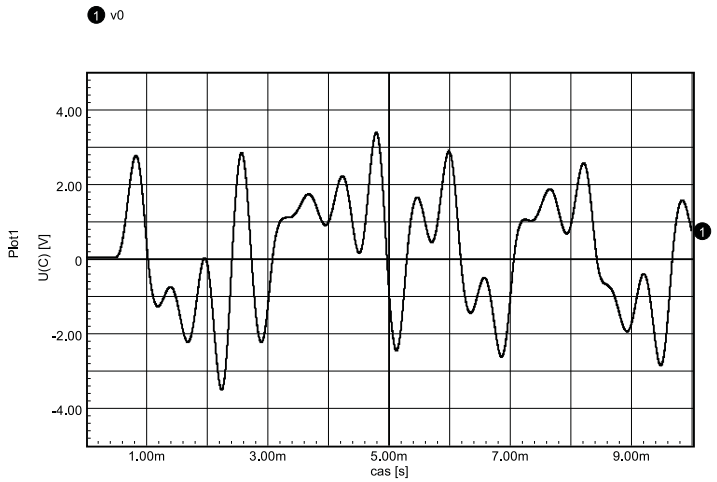


Slika 4.14: Atraktor vezja na sliki 4.2 – SPICE simulacija



Slika 4.15: Atraktor vezja na sliki 4.3 – SPICE simulacija

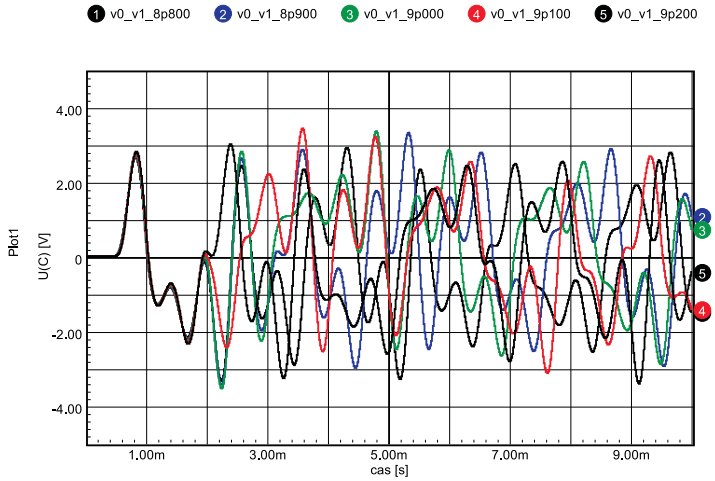
in razlika na koncu simulacije. Vidimo, da so poteki napetosti po $10ms$ že popolnoma različni. Podobne rezultate bi dobili, če bi spreminjali nek drugi parameter v vezju, kot je na primer temperatura.



Slika 4.16: Potek napetosti U_C v vezju na sliki 4.3

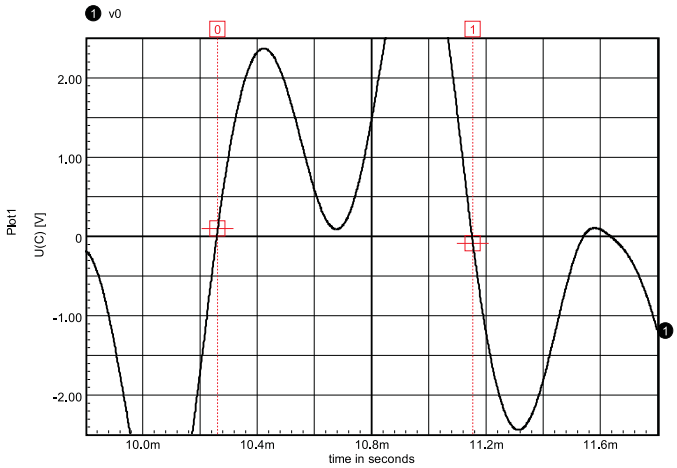
S simulacijo lahko še ponazorimo delovanje števca. Uporabimo skripto za merjenje dolžine impulza, ki smo ga opisali na začetku razdelka 4.2. Skripta je zapisana v prilogah.

Najprej smo postavili kurzorje na pozicijo $10ms$. Takšno vrednost smo izbrali, ker smo iz simulacij in poskusov na prototipu videli, da se po tem času poteki napetosti v posameznih poskusih že močno razlikujejo. V vezju smo uporabili komparator s histerezo. Ker hočemo samo ponazoriti delovanje komparatorja, izberemo le približne napetostne nivoje za preklop komparatorja $+0.1V$ in $-0.1V$. Nato oba kurzorja



Slika 4.17: Poteki napetosti U_C ob različnih napajalnih napetostih

postavimo na desno do vrednosti $-0.1V$ in še enkrat na desno do vrednost $+0.1V$. Tako sta oba kurzorja na poziciji, kjer se bo pojavila pozitivna fronta na izhodu komparatorja v prototipu. Drugi kurzor pomaknemo še enkrat naprej do vrednosti $-0.1V$, kjer se pojavi naslednja negativna fronta na izhodu komparatorja. S tem smo kurzorja postavili na začetek in konec impulza na izhodu komparatorja. Končna pozicija kurzorjev je prikazana na sliki 4.18.



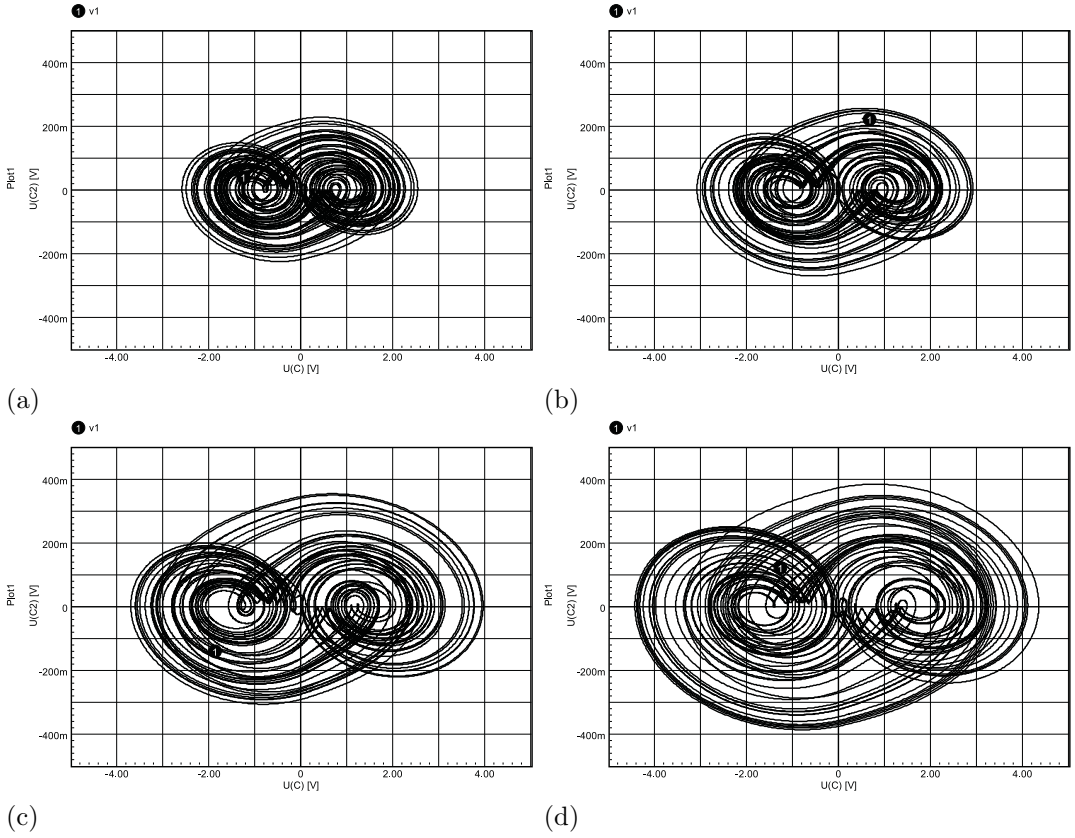
Slika 4.18: Merjenje časa T_c oziroma tvorjenje števila.

V primeru poteka na sliki je dolžina impulza $T_c = 891,552ms$. Če bi za merjenje tega časa uporabili števec s frekvenco $8MHz$ in z njega preberali spodnjih 8 bitov, bi dobili dvojiško vrednost 11011100.

4.5 Analiza toleranc

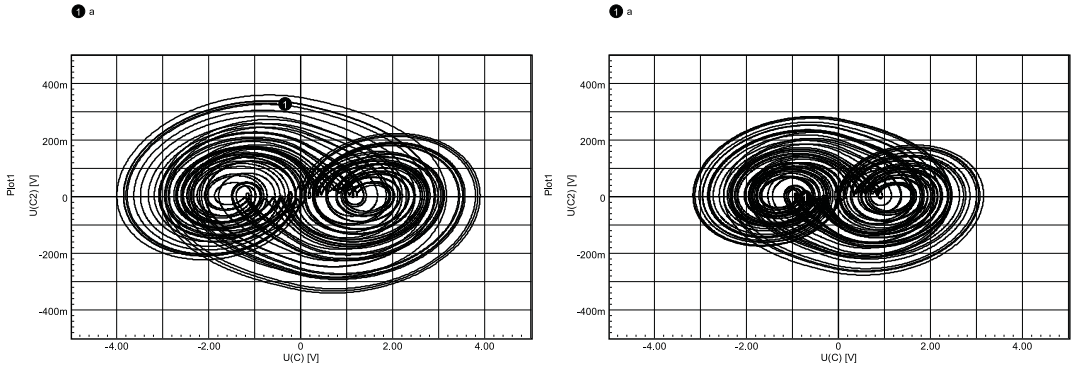
Z računalniškimi simulacijami smo preverili tolerance oscilatorja. Pravilno delovanje oscilatorja smo preverili s pogledom na atraktor. V prikazu rezultatov se omejimo le na drugi oscilator, ki je prikazan na sliki 4.3.

Pogledali smo delovanje oscilatorja glede na napajanje. Nominalna napajalna napetost je $\pm 9V$. Atraktor delovanja oscilatorja pri tej napetosti je na sliki 4.15. Delovanje preverimo pri toleranci napajanja $\pm 20\%$. Slika 4.19 prikazuje atraktor za drugi oscilator v ravnini U_C–U_{C2} za različne napajalne napetosti od $\pm 7V$ do $\pm 11V$. Na sliki vidimo, da v vseh primerih oscilator deluje v kaotičnem režimu. Deluje tudi pri nižjih napetostih, vendar tedaj ni zagotovljeno pravilno napajanje mikrokrmilnika in ostalih integriranih vezij.



Slika 4.19: Atraktor drugega oscilatorja pri različnih napajalnih napetostih:
(a) napajanje $\pm 7V$, (b) napajanje $\pm 9V$, (c) napajanje $\pm 10V$, (d) napajanje $\pm 11V$.

Preverili smo tudi delovanje oscilatorja pri različnih vrednostih uporov *R6* in *R7*. Vrednosti upora *R6* spreminjamo od $40k\Omega$ do $50k\Omega$ v korakih po $1k\Omega$. V vsakem primeru smo ugotovili, da oscilator pravilno deluje. Atraktorji, ki smo jih pri tem dobili, so na sliki 4.20. Zaradi boljše preglednosti sta na sliki prikazana le dva atraktorja



(a) (b)

Slika 4.20: Atraktor drugega oscilatorja pri različnih vrednostih upora $R6$:
(a) $R6 = 40k\Omega$, (b) $R6 = 50k\Omega$.

za primera, ko je upornost $40k\Omega$ in $50k\Omega$. Podobne rezultate smo ugotovili, ko smo spreminjali upor R7 od $7k\Omega$ do $9k\Omega$ v korakih po 200Ω. Oscilator je bil zmeraj v kaotičnem režimu.

Izvedli smo tudi Monte-Carlo analizo toleranc. Tolerance uporov nastavimo na $\pm 5\%$, tolerance kondenzatorjev na $\pm 10\%$, toleranco napajanja pa na $\pm 20\%$. Simulirali smo 100 primerov. V vsakem primeru smo ugotovili, da je oscilator pravilno deloval.

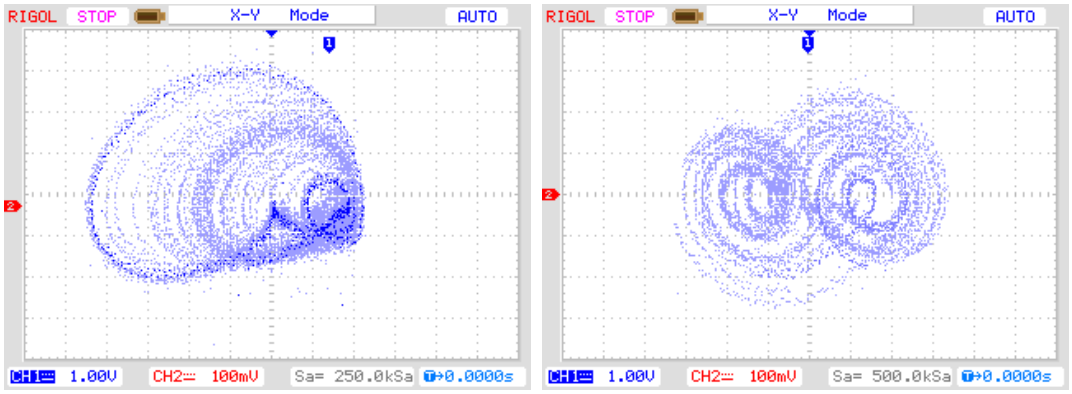
4.6 Meritve prototipa

Delovanje sestavljenega prototipa generatorja smo preverili še z meritvami. Najpomembnejše meritve se nanašanjo na pravilno delovanje oscilatorja. Preverjali smo ali ta res deluje v kaotičnem režimu. Na voljo imamo dva oscilatorja, prvi je na sliki 4.2, drugi pa na sliki 4.3.

O pravilnem delovanju kaotičnega oscilatorja se najlažje prepričamo z meritvijo z osciloskopom. V X–Y delovanja opazujemo napetosti U_C in U_{C2}. Tako vidimo atraktor v ravnini teh dveh spremenljivk. Slika na osciloskopu pri tem ne sme mirovati, ampak se mora zmeraj malo spreminjati. Na sliki 4.21a vidimo osciloskopski posnetek delovanja prvega oscilatorja, na sliki 4.21b pa posnetek delovanja drugega oscilatorja.

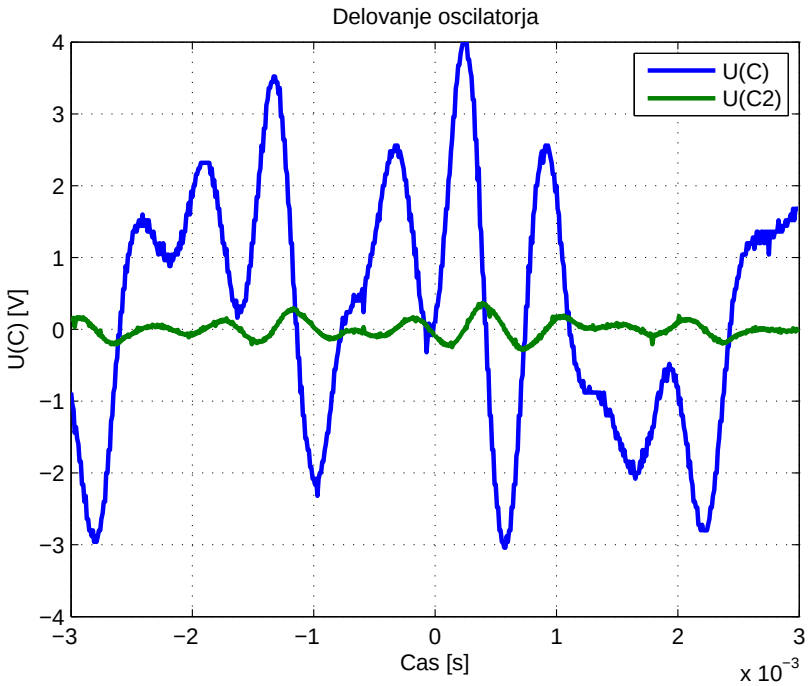
Dodatno k atraktorju še lahko opazujemo poteke napetosti v vezju v različnih situacijah. Na sliki 4.22 je prikazan potek napetosti U_C in U_{C2} po času v drugem oscilatorju med neprekinjenim delovanjem. Na sliki so vidna neperiodična nihanja v oscilatorju, ki so tipična za kaotični način delovanja.

Na sliki 4.23 je prikazan potek napetosti U_C ob vklopu napajanja oscilatorja. Vklop se zgodi ob času $0ms$. Vidimo, da oscilator takoj začne nihati. Na sliki 4.24



(a) (b)

Slika 4.21: X–Y posnetka delovanja obeh oscilatorjev

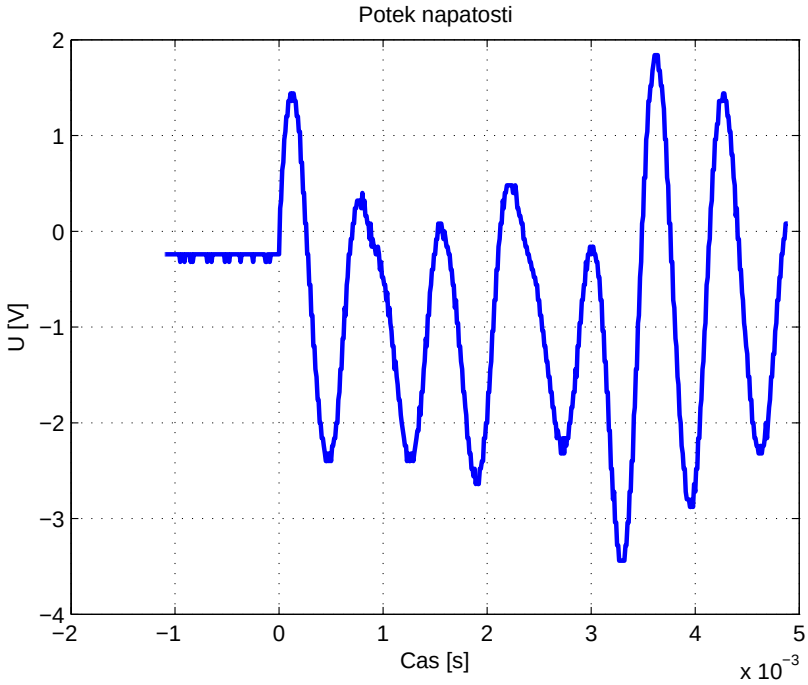


Slika 4.22: Potek napetosti U_C in U_{C2} med delovanjem drugega oscilatorja

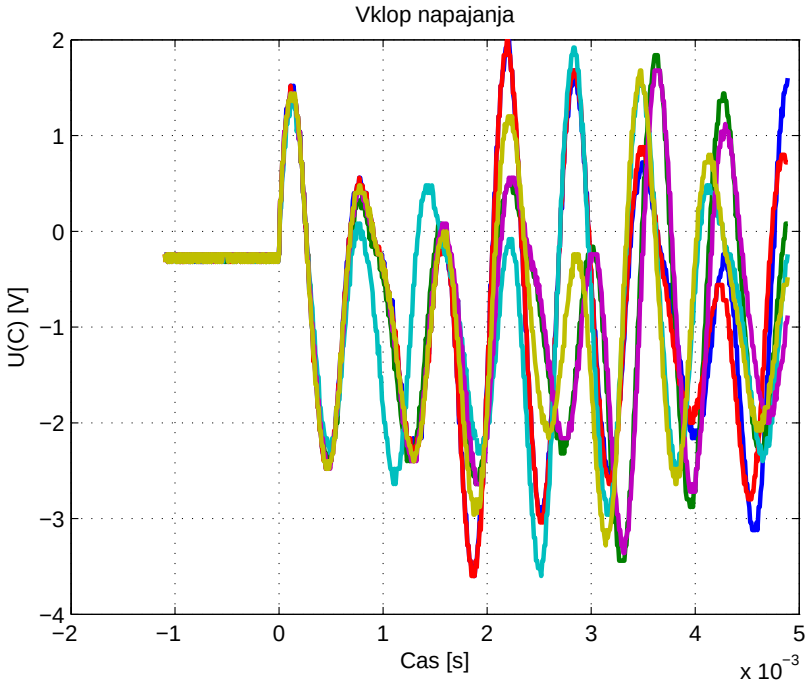
imamo podoben prikaz, vendar je na tej sliki 6 različnih potekov napetosti U_C ob vklopu napajanja. Poteki so posneti brez spreminjanja vezja in v kratkih časovnih razmikih, zato so različni poteki posledica naključnih vplivov v vezju. Tudi na tej sliki je vklop napajanja ob času $0ms$. Vidne so podobnosti v potekih takoj po vklopu napajanja in razlike, ki se počasi večajo. Vidimo, da se poteki že po $5ms$ močno razlikujejo. V razdelku 4.3 smo zapisali, da po vklopu napajanja počakamo $10ms$ in nato začnemo z meritvijo.

Tukaj sedaj vidimo, da bo $10ms$ zadosti dolg interval, po katerem lahko pričakujemo razlike v potekih napetosti. S tem je meritev že močno odvisna od začetnih

pogojev, kar pa je bistveno za delovanje našega generatorja.



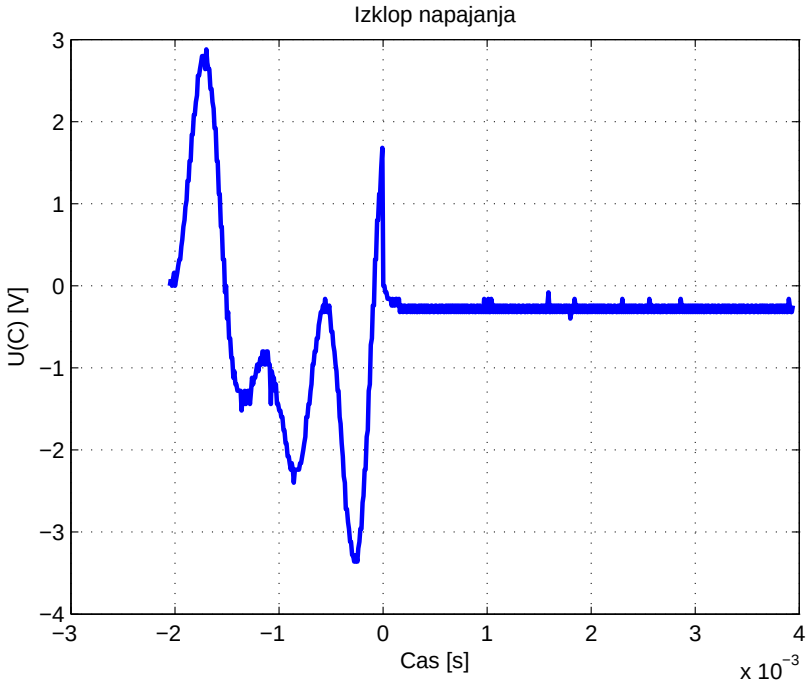
Slika 4.23: Potek napetosti U_C ob vklopu napajanja na drugem oscilatorju



Slika 4.24: Poteki napetosti U_C v intervalu šestih vklopov

Na sliki 4.25 pa je prikazan potek napetosti U_C ob izklopu napajanja. Tukaj se napajanja izklopi ob času $0ms$. Viden je nenaden padec napetosti ob izklopu, kar kaže

na to, da se oscilator hitro razelektri in je pripravljen na ponovni zagon.



Slika 4.25: Potek napetosti U_C ob izklopu napajanja na drugem oscilatorju

Tabela 4.1: Meritve porabe toka vezja s prvim oscilatorjem

	Napajalna veja	
Način napajanja	Pozitivna	Negativna
Brez	$44mA$	$20mA$
Prekinjeno	$60mA$	$31mA$
Neprekinjeno	$78mA$	$51mA$

Izmerili smo tudi porabo toka za različne primere delovanja. Tok smo izmerili glede na izbiro oscilatorja in glede na njegovo napajanje. Oscilator lahko napajamo neprekinjeno ali pa njegovo napajanje prekinjamo, tako kot ga prekinjamo med samim delovanjem. Tok še izmerimo brez napajanja oscilatorja.Rezultati meritev so prikazani v tabelah 4.1 in 4.2. V tabeli 4.1 so izmerjeni toki za vezje s prvim oscilatorjem. Ker imamo simetrično napajanje, so podani podatki za pozitivno in za negativno vejo napajanja.

4.7 Zajemanje in obdelava števil

Preizkusili smo več različic generatorja naključnih števil. Razlikujemo jih po uporabljenem vezju in delovanju mikrokrmilnika. Izbrana imamo dva kaotična oscilatorja. Prvi

Tabela 4.2: Meritve porabe toka vezja z drugim oscilatorjem

	Napajalna veja	
Način napajanja	Pozitivna	Negativna
Brez	$44mA$	$20mA$
Prekinjeno	$55mA$	$31mA$
Neprekinjeno	$63mA$	$35mA$

je na sliki 4.2, drugi pa na sliki 4.3. Potem lahko v vezju na sliki 4.7 uporabimo različne kvarčne kristale. Testirali smo s kristaloma s frekvencami $8MHz$ in $2,4576MHz$. Različne pa smo nastavili tudi premore, kjer se izklopi napajanje kaotičnega oscilatorja. Skupaj smo sestavili in preiskusili 6 različnih vezij, ki so spodaj opisana.

- Vezje 1: drugi kaotični oscilator, $8MHz$ števec, $15ms$ prekinitve napajanja.
- Vezje 2: drugi kaotični oscilator, $2,4576MHz$ števec, $15ms$ prekinitve napajanja.
- Vezje 3: prvi kaotični oscilator, $2,4576MHz$ števec, $15ms$ prekinitve napajanja.
- Vezje 4: prvi kaotični oscilator, $8MHz$ števec, $15ms$ prekinitve napajanja.
- Vezje 5: drugi kaotični oscilator, $2,4576MHz$ števec, $100ms$ prekinitve napajanja.
- Vezje 6: drugi kaotični oscilator, $8MHz$ števec, $100ms$ prekinitve napajanja.

Podatki se od mikrokrmilnika pošiljajo na osebni računalnik po vodilu *RS232*. Poslani podatki so 8-bitna števila, ki jih preberemo s števca. Na računalniku se sprejemajo in se v obliki desetiški števil, ločenih s presledki, zapisujejo v tekstovne datoteke po 16K (16384) števil. Ker naj generator naključnih števil daje binarno zaporedje, smo morali ta števila preoblikovati. Tukaj se spet pojavi več možnosti, kako iz 8-bitnih števil dobljenih od mikrokrmilnika dobimo binarno zaporedje. S programom smo iz datotek na osebnem računalniko oblikovali novo tekstovno datoteko v kateri so zapisane le ničle in enice (brez presledkov). Takšna oblika datoteke je bila potrebna, da smo lahko kasneje pognali program za statistično preverjanje. Skupaj smo preiskusili 8 različnih variant pretvorbe, ki so spodaj opisane.

- Varianta 1: dobljeno število iz mikrokrmilnika zapišemo v binarni obliki z najbolj uteženim bitom spredaj. Lahko bi zapisali število tudi z najmanj uteženim bitom spredaj, vendar s tem nebi dobili zares drugačne variante, saj bi le spremenili

vrstni red bitov ne pa dejstva ali so dovolj naključni. Tako dobimo 8 bitov za vsako število iz mikrokrmilnika.

- Varianta 2: dobljeno število pretvorimo v binarno obliko, vendar v datoteko zapišemo le 4 najmanj utežene bite. S tem, da smo odrezali sprednje bite, poskušamo dobiti bolj enakomerno porazdelitev bitov. Tako dobimo 4 bite za vsako število iz mikrokrmilnika.

- Varianta 3: podobno kot pri drugi varianti zapišemo le zadnja dva bita. Tako dobimo 2 bita za vsako število iz mikrokrmilnika.

- Varianta 4: podobno kot pri prejšnjih variatan zapišemo le zadnji bit. Tako dobimo 1 bit za vsako število iz mikrokrmilnika.

- Varianta 5: zapišemo le predzadnji bit v datoteko. Ker smo med testiranjem opazili, da je pri naketerih različicah vezja prišlo do večje neenakomernosti na zadnjem bitu, kot na predzadnjem bitu, smo se odločili tudi za to varianto. Tako dobimo 1 bit za vsako število iz mikrokrmilnika.

- Varianta 6: za osembitni podatek iz mikrokrminika izračunamo rezultat operacije *XOR* in ga zapišemo v novo datoteko. Ker smo pri analizi števil iz mikrokrmilnika opazili, da so ti dokaj enakomerno porazdeljeni, smo predvidevali neodvisnost med biti v binarnem zapisu števil. Ker lahko z operacijo *XOR* zmanjšamo neenakomernost števil, predvidevamo, da smo z operacijo *XOR* (ki ima enak učinek, kot izračun sode paritete na teh številih) zmanjšali neenakomernost. Rezultat bi lahko tudi negirali (s tem bi dobili isti rezultat kot pri izračunu lihe paritete), vendar bi s tem le zamenjali ničle z enicami in obratno. S tem pa nebi spremenili naključnosti števil. Tako dobimo 1 bit za vsako število iz mikrokrmilnika.

- Varianta 7: uporabimo operacijo *XOR* na predzadnjem bitu, za vsak podatek od drugega naprej. Pri tem za prvi podatek enostavno zapišemo predzadnji bit. Za vsak naslednji podatek pa zapišemo rezultat operacije *XOR* med predzadnjim bitom trenutnega števila in predzadnjim bitom prejšnjega števila. Pri tej metodi pa lahko prihaja do medsebojne odvisnosti med biti v končni datoteki, ker se števila uporabijo dvakrat. Tako dobimo 1 bit za vsako število iz mikrokrmilnika.

- Varianta 8: podobno kot pri prejšnji varianti uporabimo operacijo *XOR* na predzadnjem bitu. Tokrat pa jo uporabimo le za vsak drug podatek iz mikrokrmilnika. S tem se izognemo odvisnosti med dobljenimo biti v končni datoteki. Tako dobimo 1 bit na vsaki dve števili iz mikrokrmilnika.

Skupaj imamo torej 6 različnih vezij in 8 variant algoritmov za generiranja zaporedja bitov. S tem imamo 48 različic generatorja naključnih števil, ki smo jih testirali.

Generator naključnih števil je na tem mestu sestavljen iz vezja in računalniškega programa na osebnem računalniku, ki oblikuje končni izhod generatorja. Razlog za uporabo računalnika je v tem, da smo števila generirali s kaotičnim vezjem le šestkrat (za vse različice vezja). Pretvorba z računalnikom v končne različice je bistveno hitrejša. Tako smo v kratkem času dobili večje število različnih zaporedij naključnih števil, ki jih lahko testiramo.

Po opravljenem testiranju smo pogledali, katere različice generatorja prestanejo statistične teste. Izmed njih izberemo končno različico generatorja. Vsi algoritmi so toliko preprosti, da jih je možno implementirati v mikrokrmilnik ali pa jih celo trdo zvezati v vezju.

4.8 Testiranje primernosti

Dobljene naključne bite iz vseh različic generatorja smo testirali s programom, ki izvede 15 različnih statističnih testov opisanih v razdelku 3.1. Izvorna koda za program je dostopna na spletu [10]. Vsi testi so izvedeni na stopnji signifikance $\alpha = 0.01$. Dodatno k temu še nekateri testi potrebujejo dodatne parametre, ki jih nastavimo na priporočene vrednosti oziroma jih pustimo na privzetih vrednostih. Priporočene vrednosti lahko najdemo v literaturi [9].

- Pri frekvenčnem testu znotraj bloka nastavimo dolžino bloka 128.
- Pri testu ujemanja s predlogami brez prekrivanja nastavimo dolžino bloka 9.
- Pri testu ujemanja s predlogami s prekrivanjem nastavimo dolžino bloka 9.
- Pri testu približne entropije nastavimo dolžino bloka 10.
- Pri serijskem testu nastavimo dolžino bloka 16.
- Pri testu linearne kompleksnosti nastavimo dolžino bloka 500.

Za testiranje različic smo zbrali 2^{18} (262144) naključnih števil na različico, ki jih preoblikujemo v končne oblike. Prvi izmed testov je frekvenčni test. Ker je ta osnoven za nadaljnje teste, preverimo najprej rezultate tega testa, ki so zapisani v tabeli 4.3. V tabeli so rezultati za vseh 48 različic generatorja glede na uporabljeno vezje in algoritem

Tabela 4.3: Rezultati frekvenčnega testa na različicah generatorja

	varianta algoritma							
vezje	1	2	3	4	5	6	7	8
1					✓	✓	✓	✓
2			✓			✓	✓	✓
3						✓		
4					✓	✓	✓	✓
5							✓	✓
6						✓	✓	✓

Tabela 4.4: Rezultati statističnih testov na različicah generatorja (število ne prestanih testov)

	variant algoritma							
vezje	1	2	3	4	5	6	7	8
1					1	0	1	2
2			5			0	1	0
3						1		
4					1	0	0	0
5							1	1
6						1	1	1

za tvorjenje končnega zaporedja. S kljukico so označene tiste različice, ki so prestale frekvenčni test.

Za vse različice, ki so prestale začetni frekvenčni test izvedemo še ostale teste. Pri preizkušanju različic smo imeli manjšo število naključnih števil za vsako različico, zato nekateri testi niso delovali. V tabeli 4.4 je za te različice prikazano število testov, ki jih različica generatorja ni prestala. Tukaj moramo upoštevati, da se test naključnega oddaljevanja in varianta testa naključnega oddaljevanja nista izvedla pri nobeni različici zaradi premajhne količine števil. Prav tako se Maurerjev "univerzalni statistični" test zaradi majhne količine števil ni izvedel v nobeni različici razen pri vezju 2, varianti algoritma 3.

V tabeli vidimo, da so se naključna števila iz vezja 2, variante algoritma 3 najslabše odrezali. Večina ostalih različic pa je prestala vse teste ali pa je padla le pri enem testu. V vseh primerih, kjer je različica padla le pri enem testu, je to bil test ujemanja s predlogami brez prekrivanja, ki se izvaja s 148 različnimi vzorci. Ker smatramo, da zaporedje pade na testu, ko pade za vsaj enega od teh vzorcev, ni presenetljivo, da nekaj različic pade na tem testu. Različica z vezjem 1, varianto algoritma 8 je padla

tudi na testu z diskretno Fourierevo transformacijo.

Sedaj smo se odločili, katero različico bomo izbrali za naš končen generator. Ker generator deluje zelo počasi in rabimo večjo količino naključnih števil, izberemo le eno različico. Vezji 5 in 6 delujeta zaradi večjega premora počasneje od ostalih. Vezji 3 in 4 pa vsebujeta prvi kaotični oscilator, ki ima večjo porabo in se kaže bolj občutljiv na spremembe elementov. Tako smo izbirali še med vezjema 1 in 2.

Odločiti se moramo tudi za varianto algoritma. Prvih 5 variant se je slabše odneslo pri statističnih testih. Varianta 7 ima teoretično korelacijo med zaporednimi biti. Ker varianta 8 rabi za vsaki izhodni bit dve vhodni števili, je dvakrat počasnejša. Končno smo se odločili za različico sestavljeno iz vezja 1 in variantne algoritma 6.

Izbran generator testiramo tako, da z njim tvorimo večjo količino naključnih števil. Skupaj smo tvorili $10 \cdot 2^{20}$ (10485760) naključnih bitov, na katerih izvajamo statistične teste. Pri prvem – enostavnem načinu testiranja testiramo vsa števila kot en blok podatkov. Ker je ta blok podatkov zadosti velik, se lahko izvedejo vsi statistični testi. Da bo zaporedje prestalo statistične teste, moramo vse P-vrednosti biti večje od 0.01. P-vrednosti so prikazane v tabeli 4.5. Serijski test in test kumulativnih vsot dajeta po dve P-vrednost, ki sta obe zapisani v tabeli. Test ujemanja s predlogami brez prekrivanja, test naključnega oddaljevanja in varianta testa naključnega oddaljevanja pa vrnejo več P-vrednosti. V tabeli so zapisane le najmanjše. Natančni rezultati so v prilogah. Iz tabele vidimo, da je zaporedje prestalo vse statistične teste.

V drugem – sestavljenem načinu testiranja razdelimo vsa števila na 1000 blokov, ki jih testiramo vsakega za sebe. Ker imamo veliko količino blokov in stopnjo signifikance $\alpha = 0.01$, pričakujemo, da bodo nekatera zaporedja padla posamezne teste. Določimo lahko spodnjo mejo, koliko blokov mora prestati posamezni test. Pri 1000 blokih mora vsak test prestati 981 blokov. Izjemi sta test naključnega oddaljevanja in varianta testa naključnega oddaljevanja, ki pa ju nismo izvedli zaradi premajhne količine števil v posameznem bloku. Prav tako se niso izvedli test približne entropije, Maurerjev "univerzalni statistični test" test, test ranga binarne matrike, test ujemanja s predlogami brez prekrivanja in test linearne kompleksnosti. Ti testi se niso izvedli, ker niso izpolnjeni pogoji za njih. Da bi lahko izpolnili vse pogoje za teste, bi morali tvoriti 10^9 naključnih bitov, kar pa bi pri trenutni različici generatorja trajalo skoraj eno leto. Rezultati za ostale teste so prikazani v tabeli 4.6. Iz tabele vidimo, da generator ni prestal testa z diskretno Fourierevo transformacijo. Zaporedje tega testa ni

Tabela 4.5: Testiranje izbrane različice – enostavni način

Test	P-vrednost	Uspešnost
Frekvenčni	0,5356	✓
Frekvenčni znoraj bloka	0,2405	✓
Iteracijski	0,7461	✓
Najdaljša iteracija enic	0,1420	✓
Binarni rang matrike	0,3012	✓
Fouriereva transformacija	0,7147	✓
Predloge brez prekrivanja	$\geq$ 0,0200	✓
Predloge s prekrivanjem	0,9995	✓
Maurejev	0,5686	✓
Linearna kompleksnost	0,4833	✓
Serijski	0,9703, 0,6769	✓
Približna entropija	0,0968	✓
Komulativne vsote	0,4989, 0,8854	✓
Nakjučno oddaljevanje	$\geq$ 0,0946	✓
Varianta oddaljevanja	$\geq$ 0,4444	✓

prestalo zaradi neenakomerne porazdelitve P-vrednosti.

Tabela 4.6: Testiranje izbrane različice – sestavljeni način

Test	P-vrednost	Uspešnost porazdelitve	Delež	Uspešnost deleža
Frekvenčni	0,4788	✓	0,991	✓
Frekvenčni znotraj bloka	0,2430	✓	0,990	✓
Iteracijski	0,8377	✓	0,993	✓
Najdaljša iteracija enic	0,5261	✓	0,984	✓
Fouriereva transformacija	0,0000		0,988	✓
Predloge s prekrivanjem	0,0437	✓	0,991	✓
Serijski	0,1202, 0,3635	✓	0,987, 0,988	✓
Komulativne vsote	0,0915, 0,2133	✓	0,994, 0,993	✓

Na sliki 4.26 so prikazani rezultati testa Fouriereve transformacije. V prvih desetih stolpcih so prikazane relativne porazdelitve P-vrednosti dobljenih pri testiranju posameznih blokov. Vse možne vrednosti za P-vrednosti so interval od 0 do 1, ki smo ga razdelili na 10 enakih podintervalov. V enajsti vrstici je P-vrednost za χ^2 test za enakomerno porazdelitev teh relativnih frekvenc. Vidimo, da je končna P vrednost zelo majhna, kar kaže na to, da vrednosti niso enakomerno porazdeljene. V naslednjem

stolpcu je delež blokov, ki so prestali osnovni test s Fourierevo transformacijo. Ta je znotraj dovoljenih mej. Natančnejši rezultati testov so v prilogah.

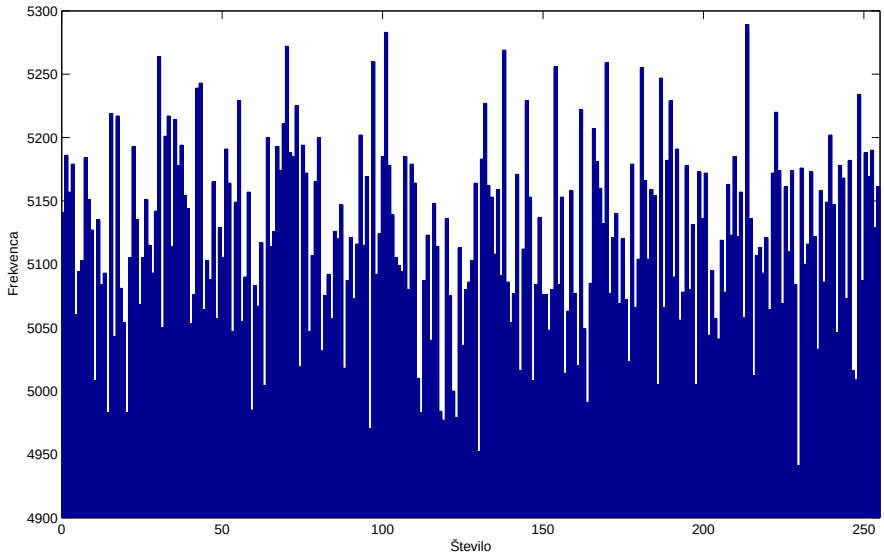
C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	P-VALUE	PROPORTION	STATISTICAL TEST
125	110	84	76	100	97	133	71	91	113	0.000022	0.9880	FFT

Slika 4.26: Rezultati testa Fouriereve transformacije na izbrani različici

5 Tvorjenje večbitnih števil

Naš generator daje zaporedje eno-bitnih števil. Več-bitna števila tvorimo tako, da vzamemo več zaporednih bitov iz generatorja ter iz njih sestavimo novo število. Pri tem ni pomembno ali je prvi bit najmanj ali najbolj utežen, saj s tem ne vplivamo na naključnost ali neodvisnost dobljenih več-bitnih števil.

Iz naših $10 \cdot 2^{20}$ 1-bitnih števil lahko tvorimo $1,25 \cdot 2^{20}$ 8-bitnih ali pa $640 \cdot 2^{10}$ 16-bitnih števil. Tvorimo jih tako, da je prvi bit najbolj utežen. Slika 5.1 prikazuje porazdelitev dobljenih 8-bitnih števil.

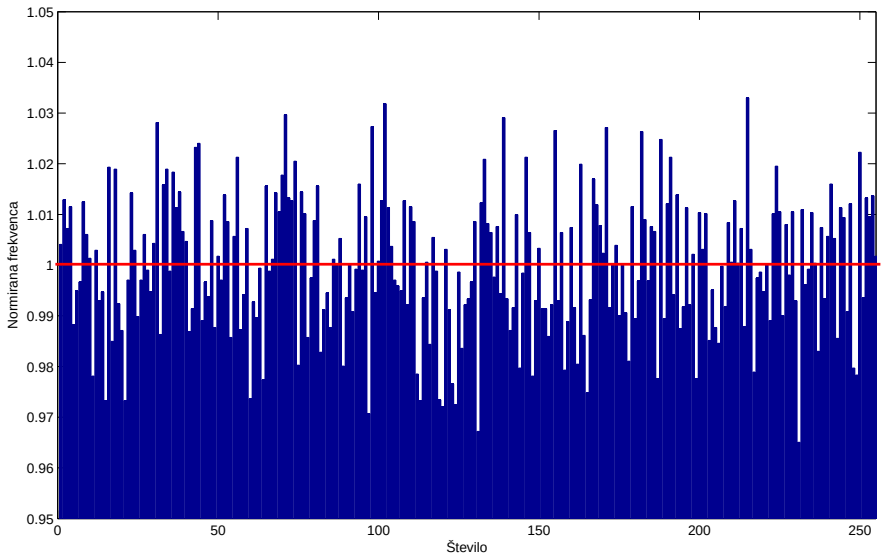


Slika 5.1: Porazdelitev 8-bitnih števil

Skupaj imamo $1,25 \cdot 2^{20}$ števil, ki zavzemajo vrednosti od 0 do 255, kar pomeni, da se mora vsako število pojaviti v povprečju $1,25 \cdot 2^{20}/256 = 5120$ krat. Slika 5.2 prikazuje porazdelitev teh števil, normirano s 5120. Na tej sliki vidimo, da frekvence odstopajo od povprečne vrednosti največ za približno 4%.

Preverimo lahko ali porazdelitev števil ustreza enakomerni porazdelitvi. To storimo s Pearsonovim χ^2 testom. Pri tem testu izračunamo testno statistiko

$$X^2 = \sum_{i=0}^{n} \frac{(O_i - E_i)^2}{E_i}. \tag{5.1}$$



Slika 5.2: Normirana porazdelitev 8-bitnih števil

Tukaj je n število različnih vrednosti ali pa intervalov vrednosti, ki jih lahko zavzamejo števila, E_i pričakovana frekvenca števil ob predpostavki enakomerne porazdelitve, O_i pa opazovana frekvenca števil. V našem primeru je $n = 256$ in $E_i = 5120$. Iz podatkov izračunamo

$$X^2 = \sum_{i=0}^{255} \frac{(O_i - 5120)^2}{5120} = 242,1125. \tag{5.2}$$

Ker seštevamo kvadrate razlik med opazovano frekvenco in pričakovano frekvenco, bo manjša vrednost testne statisteke pomenila, da so števila porazdeljena bolj enakomerno. Izračunamo tudi mejno vrednost za testno statistiko iz χ^2 porazdelitve. Na stopnji signifikance $\alpha = 0,01$ je $X_0^2 = 293,2478$. Ker je $X^2 < X_0^2$, sklepamo, da so števila porazdeljena enakomerno.

6 Sklep

Predstavili smo metodo za sintezo dinamičnih elektronskih vezij, ki so podana z diferencialno enačbo, in prikazali nekaj enostavnih kaotičnih sistemov, ki je je možno enostavno in brez uporabe tuljav implementirati v elektronska vezja.

Generator naključnih števil smo sestavili iz kaotičnega oscilatorja, vezja za analogno-digitalno pretvorbo in dodatnega procesiranja na osebnem računalniku. Procesiranje na računalniku je toliko enostavno, da ga lahko enostavno implementiramo v logično vezje. Tako lahko sestavimo celotni generator naključnih števil v obliki vezja.

Skupaj smo testirali 48 različic generatorja. Izvedeni statistični testi na končni različici potrjujejo primernosti števil, ki jih tvorimo. Izjema je le test z diskretno Fourierevo transformacijo. Ta nam zavrne hipotezo o primernosti. vendar smo testirali na relativno majhnem vzorcu, kar lahko zmanjša zanesljivost testov.

Za bolj podrobno analizo našega generatorja naključnih števil, bi morali tvoriti več števil in ponovno izvajati teste. Z večjo količino števil bi lahko izvedli tudi preostale teste, ki jih sedaj nismo mogli, vendar bi zajemanje zadostne količine naključnih števil trajalo mnogo dlje. Bolj podrobno bi veljalo tudi preveriti zakaj je test s diskretno Fourierevo transformacijo zavrnil hipotezo o primernosti.

Dobljena števila smo preverjali s paketom testov, ki jih je izdal ameriški Nacionalni inštitut za standarde in tehnologijo. Ti testi so namenjeni splošnemu preverjanju generatorjev naključnih števil, ki bi naj služili v kriptografske namene. Za preverjanje uporabnosti našega generatorja na konkretnem področju moramo preveriti, če naš generator ustreza dolčenim standardom, kateri so predpisani na tem področju.

Prednost našega generatorja naključnih števil je ta, da njegovo obnašanje ni določeno z nekim algoritmom. Zato ga ni možno natančno napovedati, kot pri generatorjih psevdonaključnih števil. Prav tako je prednost, da je vezje sestavljeno iz preprostih gradnikov, ki so integrabilni.

Slabost je, da uporablja analogno vezje. V množični proizvodnji ne moremo zagotoviti, da bodo vsi generatorji enaki. Zato je potrebna bolj podrobna analiza toleranc kaotičnega oscilatorja. Velika slabost je tudi njegova počasnost. V trenutni različici

generatorja tvorimo le približno 35 bitov na sekundo.

Z nadaljnjim delom bi lahko tvorili več števil in tako izboljšali zanesljivost testov. Poskusimo lahko sestaviti tudi nove različice generatorja z drugimi oscilatorji in algoritmi za tvorjenje končnih števil. Uporabimo lahko tudi druge pakete statističnih testov. Primer takega paketa je DIEHARD, ki ga je razvil G. Marsiglia [16]. Ta paket vsebuje nakaj testov, ki jih mi nismo uporabljali.

Izboljšati bi predvsem veljalo hitrost generatorja. To lahko storimo s tem, da ne prekinjamo napajanja in s tem simuliramo situacijo, da generator deluje v aplikaciji, kjer napajanje ni težavno. Izberemo lahko drugi kaotični oscilator ali pa že uporabljenega skaliramo na večjo frekvenco. Tedaj je potrebno pospešiti tudi ostale komponente v generatorju, da bo generator pravilno deloval.

S podrobno matematično analizo delovanja sistema lahko poiščemo tista območja v prostoru stanj, kjer se trajektorije najhitreje oddaljujejo. Vezje potem poskušamo sestaviti tako, da z krmilnimi impulzi povzročimo v vezju začetne pogoje, ki so v tem področju. S tem skrajšamo čas, ki ga potrebujemo za zagon oscilatorja.

Seveda lahko izboljšamo hitrost generatorja s tem, da uporabimo več oscilatorjev ali pa ga uporabimo v kombinaciji z generatorjem psevdonaključnih števil.

7 Literatura

[1] Comscire, http://www.comscire.com/Products/R2000KU/, prebrano 2.8.2010.

[2] K. M. Cuomo, A. V. Oppenheim, S. H. Strogatz, Synchronization of Lorentz-Based Chaotic Circuits with Applications to Communications, *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing*, Vol. 40 (1993), No. 10, str. 626-633.

[3] R. B. Davies, Exclusive OR (XOR) and hardware random number generators, http://www.robertnz.net/pdf/xor2.pdf, prebrano 19.7.2010.

[4] R. B. Davies, Hardware Random Number Generators, http://www.robertnz.net/hwrng.htm, prebrano dne 2.8.2010.

[5] T. Dogša, *Nelinearna elektronika, zbrano gradivo*, FERI, Maribor, 2008.

[6] J. Gleick, *Kaos: rojstvo nove vede*, 1. izd., Državna založba Slovenije, Ljubljana, 1991.

[7] D. Knuth, *The art of computer programming, Volume 2: seminumerical algorithms*, 3. ed., Addison-Wesley, 1998.

[8] A. Manezes, P. van Oorschot, S. Vanstone, Pseudorandom Bits and Sequences, v: *Handbook of applied cryptography*, CRC Press, 1997, str. 169-190.

[9] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, S. Vo, *A Statistical Test Suite for Random and Speudorandom Number Generators for Cryptographic Application*, National Institute of Standards and Technology, 2008.

[10] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, S. Vo, A Statistical Test Suite for the Validation of Random Number Generators and Pseudo Random Number Generators for Cryptographic Applications, http://csrc.nist.gov/groups/ST/toolkit/rng/documentation_software.html, prebrano 31.7.2010.

[11] J. C. Sprott, A new class of chaotic circuit, *Physics Letters A*, Vol. 266 (2000), Issue 1, str. 19-23.

[12] J. C. Sprott, Simple chaotic systems and circuits, *American Journal of Physics*, Vol. 68 (2000), No. 8, str 758-763.

[13] E. Tamaševičiūtė, A. Tamaševičius, G. Mykolaitis, S. Bumelienė E. Lindberg, Analogue Elictrical Circuit for Simulation of the Duffin-Holmes equation, *Nonlinear Analysis: Modellng and Control*, Vol. 13 (2008), No. 2, str. 241-252.

[14] Bernd Ulmann, A True Random Number Generator, http://www.vaxman.de/projects/rng/rng.html, prebrano dne 2.8.2010.

[15] Wikipedia: Chaos theory, http://en.wikipedia.org/wiki/Chaos_theory, prebrano dne 30.7.2010.

[16] Wikipedia: Diehard tests, http://en.wikipedia.org/wiki/Diehard_tests, prebrano dne 26. avgust 2010.

[17] Wikipedia: Random number generator, http://en.wikipedia.org/wiki/Random_number_generator, prebrano dne 2.8.2010.

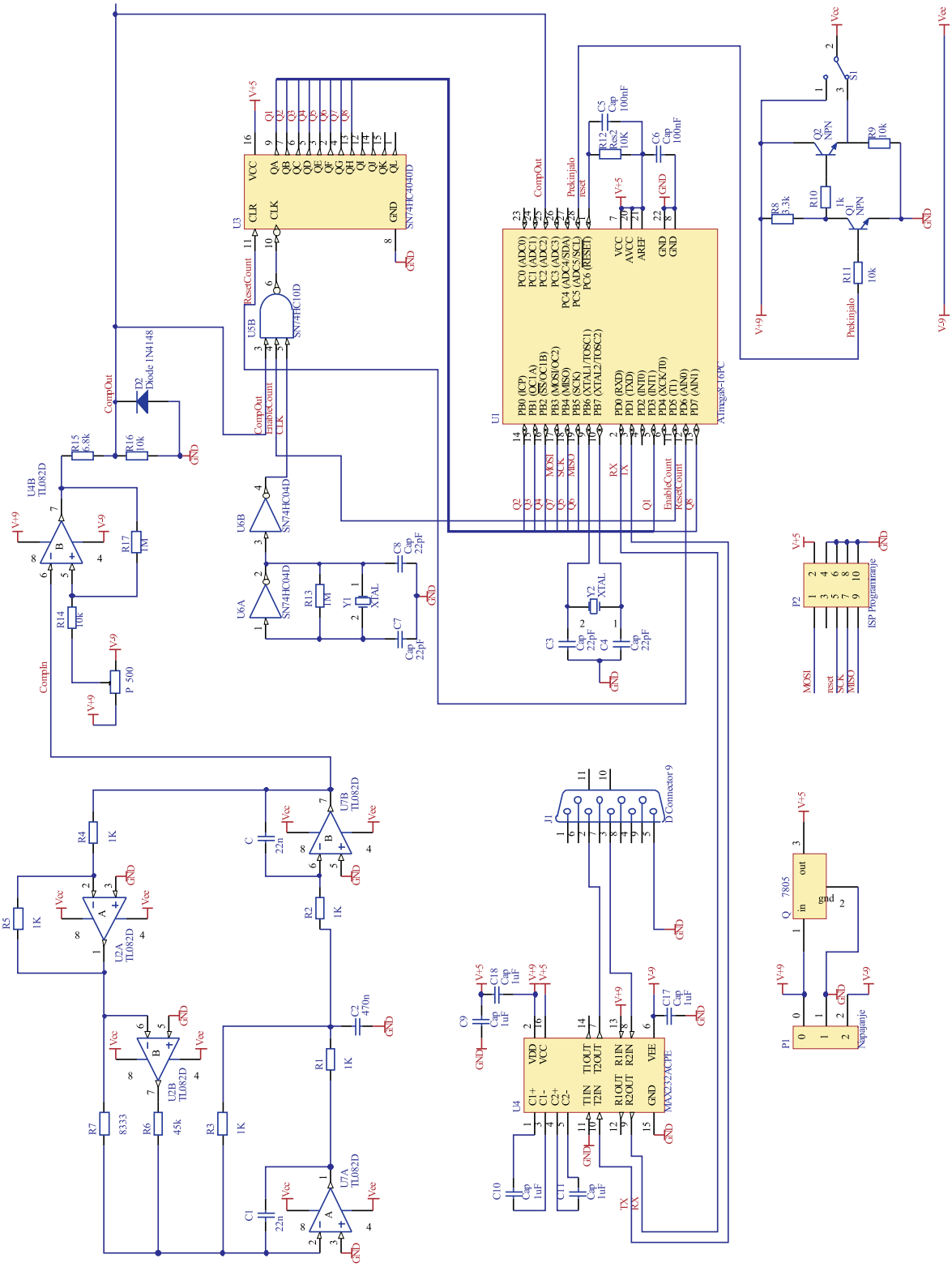
[18] G. P. Williams, *Chaos Theoty Tamed*, Joseph Henry Press, Washington D.C., 1997.

8 Priloge

8.1 Seznam slik

8.2 Seznam tabel

8.3 Shema vezja



Slika 8.1: Končno vezje z drugim kaotičnim oscilatorjem

8.4 Skripta za merjenje časa impulza

Zapisana je skripta, s katero med simulacijo merimo dolžina časa, ko je opazovana napetost nad določenim nivojem. Na shemah je ta napetost označena z U_C, v računalniški simulaciji in s tem tudi v tej skripti pa z $V(7)$.

```
* kurzorje postavimo na pozicijo 10ms
* gledali bomo napetost V(7)
homeCursors
setCursor 0 10m
setCursor 1 10m
* s kurzorjem 0 gremo do prehoda od -0.1 V do 0.1 V
moveCursorRight 0 V(7) -0.1
moveCursorRight 0 V(7) 0.1
* enako z kurzorje 1
moveCursorRight 1 V(7) -0.1
moveCursorRight 1 V(7) 0.1
* z kurzorjem 1 nadaljujemo spet do -0.1V
moveCursorRight 1 V(7) -0.1

* izracunaj cas med kurzorjema
cas = getCursorX(1)-getCursorX(0)
print cas
```

Slika 8.2: Skripta za merjenje časa impulza

8.5 Rezultati statističnih analiz

Prikazani so rezultati statističnih analiz za teste izvedene na prvi način na izbrani različici. Prikazani so tudi strnjeni rezulatati testov izvedenih na drugi način. Za podrobnejšo razlago testov in rezultatov glej [9].

Frekvenčni test

```
      FREQUENCY TEST
-----------------------------------------
COMPUTATIONAL INFORMATION:
-----------------------------------------
(a) The nth partial sum = -2006
(b) S_n/n               = -0.000191
-----------------------------------------
SUCCESS p_value = 0.535597
```

Frekvenčni test znotraj bloka

```
BLOCK FREQUENCY TEST
-----------------------------------------
COMPUTATIONAL INFORMATION:
-----------------------------------------
(a) Chi^2           = 82204.843750
(b) # of substrings = 81920
(c) block length    = 128
(d) Note: 0 bits were discarded.
-----------------------------------------
SUCCESS p_value = 0.240546
```

Iteracijski test

```
RUNS TEST
-----------------------------------------
COMPUTATIONAL INFORMATION:
-----------------------------------------
(a) Pi                        = 0.499904
(b) V_n_obs (Total # of runs) = 5243404
(c) V_n_obs - 2 n pi (1-pi)
    ----------------------    = 0.228931
      2 sqrt(2n) pi (1-pi)
-----------------------------------------
SUCCESS p_value = 0.746121
```

Test za najdaljšo iteracijo enic v bloku

```
  LONGEST RUNS OF ONES TEST
--------------------------------------------
COMPUTATIONAL INFORMATION:
--------------------------------------------
(a) N (# of substrings)  = 1048
(b) M (Substring Length) = 10000
(c) Chi^2                = 9.610980
--------------------------------------------
      F R E Q U E N C Y
--------------------------------------------
<=10  11  12  13  14  15 >=16 P-value  Assignment
  88 233 243 192 117  79   96 SUCCESS p_value = 0.142020
```

Test ranga binarne matrike

```
RANK TEST
--------------------------------------------
COMPUTATIONAL INFORMATION:
--------------------------------------------
(a) Probability P_32 = 0.288788
(b)             P_31 = 0.577576
(c)             P_30 = 0.133636
(d) Frequency   F_32 = 2928
(e)             F_31 = 5891
(f)             F_30 = 1421
(g) # of matrices    = 10240
(h) Chi^2            = 2.400123
(i) NOTE: 0 BITS WERE DISCARDED.
--------------------------------------------
SUCCESS p_value = 0.301176
```

Test z diskretno Fourierevo transformacijo

```
FFT TEST
--------------------------------------------
COMPUTATIONAL INFORMATION:
--------------------------------------------
(a) Percentile = 95.002460
(b) N_l        = 4980865.000000
(c) N_o        = 4980736.000000
(d) d          = 0.365572
--------------------------------------------
SUCCESS p_value = 0.714684
```

Test ujemanja s predlogami brez prekrivanja

```
  NONPERIODIC TEMPLATES TEST
--------------------------------------------------------------------------------
  COMPUTATIONAL INFORMATION
--------------------------------------------------------------------------------
LAMBDA = 2559.984375 M = 1310720 N = 8 m = 9 n = 10485760
--------------------------------------------------------------------------------
F R E Q U E N C Y
Template   W_1  W_2  W_3  W_4  W_5  W_6  W_7  W_8    Chi^2   P_value Assignment Index
--------------------------------------------------------------------------------
000000001 2544 2577 2590 2568 2568 2667 2521 2508  6.969281 0.539951 SUCCESS   0
000000011 2519 2598 2555 2689 2553 2580 2554 2530  8.557198 0.381014 SUCCESS   1
000000101 2602 2625 2491 2608 2533 2630 2531 2566  7.904369 0.442867 SUCCESS   2
000000111 2587 2586 2598 2614 2549 2579 2556 2535  2.784597 0.947139 SUCCESS   3
000001001 2497 2615 2595 2493 2581 2499 2557 2578  6.949887 0.542049 SUCCESS   4
000001011 2581 2503 2471 2533 2567 2619 2583 2586  6.898524 0.547620 SUCCESS   5
000001101 2554 2617 2522 2648 2559 2576 2601 2574  5.903978 0.657987 SUCCESS   6
000001111 2569 2561 2571 2595 2570 2588 2585 2514  2.042539 0.979680 SUCCESS   7
000010001 2570 2483 2548 2597 2579 2575 2661 2541  7.552450 0.478361 SUCCESS   8
000010011 2507 2599 2612 2510 2560 2572 2552 2615  5.158978 0.740457 SUCCESS   9
000010101 2586 2578 2569 2598 2569 2598 2567 2510  2.667463 0.953464 SUCCESS  10
000010111 2568 2549 2544 2620 2690 2629 2550 2595 10.923347 0.206080 SUCCESS  11
000011001 2602 2629 2559 2500 2514 2491 2567 2597  7.442564 0.489720 SUCCESS  12
000011011 2531 2586 2572 2597 2580 2582 2560 2535  1.834736 0.985666 SUCCESS  13
000011101 2513 2508 2603 2556 2521 2534 2592 2564  4.045342 0.853010 SUCCESS  14
000011111 2526 2533 2542 2509 2585 2549 2561 2501  3.649521 0.887281 SUCCESS  15
000100011 2592 2447 2543 2576 2529 2568 2607 2619  8.506276 0.385639 SUCCESS  16
000100101 2562 2549 2560 2503 2598 2594 2547 2613  3.617551 0.889878 SUCCESS  17
000100111 2544 2542 2597 2554 2540 2565 2614 2564  2.158890 0.975749 SUCCESS  18
000101001 2498 2601 2470 2582 2496 2540 2454 2579 12.199546 0.142520 SUCCESS  19
000101011 2564 2545 2632 2492 2570 2583 2524 2523  5.390518 0.715137 SUCCESS  20
000101101 2583 2619 2589 2545 2570 2612 2562 2617  4.500923 0.809341 SUCCESS  21
000101111 2501 2573 2575 2631 2644 2513 2575 2565  7.448094 0.489146 SUCCESS  22
000110011 2595 2567 2628 2508 2541 2553 2524 2520  4.810758 0.777598 SUCCESS  23
000110101 2620 2532 2532 2702 2488 2643 2601 2506 16.972324 0.030399 SUCCESS  24
000110111 2519 2605 2581 2560 2596 2556 2505 2526  3.894546 0.866510 SUCCESS  25
000111001 2582 2531 2495 2571 2476 2542 2553 2539  5.468700 0.706505 SUCCESS  26
000111011 2523 2526 2638 2578 2442 2594 2572 2525 10.254319 0.247628 SUCCESS  27
000111101 2630 2571 2568 2545 2562 2547 2496 2567  3.890178 0.866894 SUCCESS  28
000111111 2567 2552 2543 2551 2562 2549 2533 2523  1.092071 0.997597 SUCCESS  29
001000011 2587 2525 2569 2594 2480 2643 2542 2522  7.372703 0.497006 SUCCESS  30
001000101 2531 2576 2608 2569 2627 2578 2575 2494  5.203410 0.735631 SUCCESS  31
001000111 2578 2514 2506 2565 2557 2613 2647 2537  6.585114 0.581983 SUCCESS  32
001001011 2631 2549 2535 2566 2561 2626 2593 2586  4.828372 0.775753 SUCCESS  33
001001101 2479 2596 2419 2569 2536 2643 2541 2511 15.369774 0.052342 SUCCESS  34
001001111 2608 2565 2603 2508 2566 2594 2606 2602  4.832071 0.775365 SUCCESS  35
001010011 2530 2642 2518 2513 2496 2562 2519 2619  8.426857 0.392922 SUCCESS  36
001010101 2593 2589 2619 2605 2574 2585 2530 2560  3.702008 0.882959 SUCCESS  37
001010111 2565 2588 2599 2543 2543 2592 2593 2570  2.070518 0.978773 SUCCESS  38
001011011 2667 2652 2532 2503 2565 2549 2563 2624 11.394938 0.180310 SUCCESS  39
001011101 2580 2463 2590 2528 2589 2557 2507 2626  7.978486 0.435575 SUCCESS  40
001011111 2581 2541 2569 2569 2571 2623 2529 2618  3.791024 0.875469 SUCCESS  41
001100101 2597 2659 2635 2555 2531 2601 2482 2641 12.926832 0.114392 SUCCESS  42
001100111 2576 2581 2646 2484 2577 2517 2499 2555  7.980443 0.435383 SUCCESS  43
001101011 2591 2523 2501 2623 2512 2581 2615 2593  6.723574 0.566729 SUCCESS  44
```

001101101 2600 2564 2547 2644 2570 2519 2598 2637 7.273220 0.507462 SUCCESS 45
001101111 2534 2642 2511 2656 2541 2621 2518 2529 10.434849 0.235821 SUCCESS 46
001110101 2487 2680 2591 2570 2655 2523 2666 2553 17.162286 0.028463 SUCCESS 47
001110111 2522 2474 2584 2572 2432 2603 2522 2586 12.083726 0.147507 SUCCESS 48
001111011 2536 2437 2533 2506 2551 2582 2592 2501 9.863612 0.274729 SUCCESS 49
001111101 2464 2567 2604 2515 2569 2566 2522 2516 6.754786 0.563306 SUCCESS 50
001111111 2640 2600 2516 2550 2619 2555 2539 2537 5.864395 0.662419 SUCCESS 51
010000011 2550 2648 2588 2495 2490 2438 2587 2552 13.505405 0.095603 SUCCESS 52
010000111 2544 2503 2513 2625 2528 2628 2577 2521 7.028574 0.533553 SUCCESS 53
010001011 2543 2610 2626 2550 2614 2511 2599 2585 5.944622 0.653435 SUCCESS 54
010001111 2519 2568 2475 2573 2538 2633 2576 2605 6.962917 0.540639 SUCCESS 55
010010011 2527 2585 2452 2523 2634 2570 2556 2515 9.034470 0.339396 SUCCESS 56
010010111 2579 2493 2550 2482 2550 2610 2564 2643 8.298473 0.404871 SUCCESS 57
010011011 2592 2693 2478 2625 2481 2570 2543 2597 15.217816 0.055046 SUCCESS 58
010011111 2619 2539 2623 2531 2648 2572 2590 2545 7.172059 0.518188 SUCCESS 59
010100011 2491 2578 2496 2573 2535 2522 2627 2592 6.840380 0.553949 SUCCESS 60
010100111 2474 2546 2493 2572 2573 2591 2541 2636 7.874875 0.445787 SUCCESS 61
010101011 2547 2530 2587 2552 2578 2546 2570 2526 1.469344 0.993205 SUCCESS 62
010101111 2491 2633 2557 2612 2574 2521 2563 2564 5.877210 0.660984 SUCCESS 63
010110011 2489 2549 2644 2491 2538 2538 2567 2543 7.386352 0.495578 SUCCESS 64
010110111 2671 2485 2585 2511 2626 2475 2560 2531 13.492071 0.096003 SUCCESS 65
010111011 2604 2495 2632 2491 2596 2498 2554 2545 8.688877 0.369218 SUCCESS 66
010111111 2594 2571 2511 2600 2538 2560 2560 2595 2.823650 0.944932 SUCCESS 67
011000111 2512 2611 2540 2544 2565 2499 2595 2476 7.104521 0.525399 SUCCESS 68
011001111 2567 2517 2684 2523 2564 2578 2551 2571 7.752463 0.458016 SUCCESS 69
011010111 2516 2542 2541 2590 2524 2527 2600 2599 3.646756 0.887506 SUCCESS 70
011011111 2639 2571 2457 2594 2495 2579 2572 2540 9.396352 0.309971 SUCCESS 71
011101111 2538 2502 2592 2510 2540 2550 2522 2541 3.907564 0.865366 SUCCESS 72
011111111 2565 2663 2511 2580 2614 2620 2618 2568 9.449332 0.305819 SUCCESS 73
100000000 2544 2577 2590 2568 2568 2667 2521 2508 6.969281 0.539951 SUCCESS 74
100010000 2531 2548 2573 2565 2561 2568 2498 2469 5.399508 0.714146 SUCCESS 75
100100000 2606 2574 2474 2564 2512 2562 2549 2599 5.524344 0.700341 SUCCESS 76
100101000 2502 2622 2477 2562 2504 2471 2591 2591 10.939395 0.205157 SUCCESS 77
100110000 2595 2579 2551 2611 2586 2598 2551 2583 2.829673 0.944587 SUCCESS 78
100111000 2479 2595 2494 2550 2592 2581 2543 2506 6.831364 0.554932 SUCCESS 79
101000000 2581 2556 2572 2625 2595 2660 2636 2566 8.837463 0.356193 SUCCESS 80
101000100 2479 2585 2646 2644 2553 2574 2513 2578 9.866213 0.274542 SUCCESS 81
101001000 2572 2544 2570 2577 2575 2526 2502 2571 2.284306 0.971032 SUCCESS 82
101001100 2493 2625 2620 2520 2640 2537 2521 2537 9.249862 0.321658 SUCCESS 83
101010000 2579 2633 2537 2563 2547 2596 2650 2519 7.062021 0.529956 SUCCESS 84
101010100 2616 2619 2538 2554 2533 2650 2598 2544 7.139950 0.521611 SUCCESS 85
101011000 2572 2559 2589 2578 2476 2589 2602 2490 6.412210 0.601161 SUCCESS 86
101011100 2582 2583 2623 2489 2654 2550 2538 2557 7.860607 0.447204 SUCCESS 87
101100000 2566 2523 2566 2548 2480 2602 2517 2522 5.267526 0.728637 SUCCESS 88
101100100 2667 2510 2549 2627 2505 2615 2535 2605 11.015417 0.200827 SUCCESS 89
101101000 2593 2499 2582 2648 2529 2550 2552 2610 6.733574 0.565632 SUCCESS 90
101101100 2593 2488 2637 2537 2574 2557 2514 2537 6.294811 0.614246 SUCCESS 91
101110000 2553 2584 2563 2444 2549 2553 2585 2572 6.071352 0.639240 SUCCESS 92
101110100 2515 2458 2514 2596 2594 2619 2511 2656 12.967463 0.112974 SUCCESS 93
101111000 2613 2566 2574 2553 2586 2574 2529 2523 2.542703 0.959710 SUCCESS 94
101111100 2521 2618 2548 2567 2575 2576 2577 2594 2.831112 0.944505 SUCCESS 95
110000000 2507 2643 2548 2580 2515 2549 2537 2550 5.258751 0.729596 SUCCESS 96
110000010 2612 2531 2526 2514 2509 2594 2535 2520 5.169559 0.739309 SUCCESS 97
110000100 2615 2569 2562 2501 2583 2616 2624 2573 5.869130 0.661889 SUCCESS 98
110001000 2536 2516 2603 2514 2530 2627 2530 2543 5.273726 0.727959 SUCCESS 99
110001010 2510 2558 2627 2591 2517 2580 2565 2612 5.226061 0.733164 SUCCESS 100

```
110010000 2604 2588 2593 2644 2554 2657 2695 2640 18.161806 0.020046 SUCCESS 101
110010010 2560 2582 2484 2605 2574 2623 2543 2491  7.070481 0.529048 SUCCESS 102
110010100 2513 2652 2492 2566 2508 2475 2510 2642 13.932198 0.083550 SUCCESS 103
110011000 2580 2410 2590 2605 2505 2534 2579 2535 12.326289 0.137229 SUCCESS 104
110011010 2538 2579 2596 2496 2669 2463 2571 2611 12.222362 0.141555 SUCCESS 105
110100000 2509 2547 2545 2604 2552 2603 2641 2575  5.508284 0.702122 SUCCESS 106
110100010 2512 2544 2606 2638 2616 2573 2509 2611  7.786276 0.454621 SUCCESS 107
110100100 2556 2573 2521 2546 2675 2605 2537 2509  8.195279 0.414631 SUCCESS 108
110101000 2531 2680 2568 2601 2578 2507 2556 2570  8.177160 0.416359 SUCCESS 109
110101010 2570 2560 2506 2501 2537 2611 2638 2510  7.357387 0.498609 SUCCESS 110
110101100 2596 2572 2645 2583 2571 2596 2573 2568  4.384243 0.820899 SUCCESS 111
110110000 2516 2562 2558 2584 2594 2571 2483 2553  3.948751 0.861717 SUCCESS 112
110110010 2593 2620 2678 2539 2527 2517 2515 2483 12.099294 0.146828 SUCCESS 113
110110100 2666 2584 2526 2633 2500 2585 2606 2528 10.370354 0.239988 SUCCESS 114
110111000 2588 2611 2596 2514 2537 2536 2624 2516  5.630430 0.688551 SUCCESS 115
110111010 2528 2478 2556 2568 2551 2571 2512 2525  4.667892 0.792411 SUCCESS 116
110111100 2662 2607 2484 2526 2525 2587 2527 2514  9.980809 0.266375 SUCCESS 117
111000000 2612 2623 2613 2499 2550 2512 2519 2566  6.999774 0.536657 SUCCESS 118
111000010 2562 2593 2549 2528 2579 2639 2640 2542  6.290405 0.614738 SUCCESS 119
111000100 2538 2578 2584 2604 2616 2649 2568 2499  7.340152 0.500416 SUCCESS 120
111000110 2507 2681 2622 2526 2493 2523 2512 2533 12.661983 0.124021 SUCCESS 121
111001000 2619 2575 2605 2606 2565 2648 2562 2491  8.237160 0.410653 SUCCESS 122
111001010 2530 2525 2450 2545 2600 2541 2523 2649 10.382766 0.239182 SUCCESS 123
111001100 2602 2494 2523 2585 2531 2589 2569 2507  5.124647 0.744174 SUCCESS 124
111010000 2514 2485 2536 2579 2534 2594 2641 2613  8.032526 0.430299 SUCCESS 125
111010010 2601 2455 2568 2515 2534 2536 2617 2631  9.832842 0.276955 SUCCESS 126
111010100 2546 2584 2508 2573 2562 2571 2567 2593  1.983334 0.981519 SUCCESS 127
111010110 2574 2554 2675 2595 2643 2481 2539 2525 11.911642 0.155190 SUCCESS 128
111011000 2495 2492 2606 2564 2576 2466 2511 2555  9.087791 0.334944 SUCCESS 129
111011010 2629 2636 2556 2540 2485 2593 2507 2512  9.203763 0.325400 SUCCESS 130
111011100 2500 2519 2547 2568 2520 2556 2579 2553  3.044723 0.931520 SUCCESS 131
111100000 2672 2616 2572 2542 2612 2524 2539 2587  8.615657 0.375748 SUCCESS 132
111100010 2478 2548 2545 2647 2496 2562 2567 2517  8.345960 0.400426 SUCCESS 133
111100100 2578 2583 2497 2603 2552 2650 2594 2664 10.834155 0.211274 SUCCESS 134
111100110 2545 2563 2523 2616 2573 2555 2574 2522  2.655645 0.954078 SUCCESS 135
111101000 2551 2510 2596 2605 2581 2666 2648 2582 10.430342 0.236111 SUCCESS 136
111101010 2554 2542 2521 2566 2614 2541 2534 2625  4.079003 0.849926 SUCCESS 137
111101100 2591 2483 2638 2617 2560 2461 2497 2593 12.557880 0.127994 SUCCESS 138
111101110 2507 2466 2496 2548 2588 2635 2577 2492 10.990569 0.202234 SUCCESS 139
111110000 2653 2595 2593 2564 2563 2589 2523 2638  7.793713 0.453876 SUCCESS 140
111110010 2528 2635 2548 2538 2552 2614 2527 2663  8.872286 0.353184 SUCCESS 141
111110100 2561 2425 2614 2589 2618 2607 2637 2561 13.531364 0.094828 SUCCESS 142
111110110 2602 2560 2603 2524 2518 2474 2491 2613  8.741832 0.364541 SUCCESS 143
111111000 2536 2628 2604 2594 2455 2542 2506 2562  9.114887 0.332697 SUCCESS 144
111111010 2576 2408 2536 2612 2643 2629 2612 2506 17.741945 0.023247 SUCCESS 145
111111100 2530 2665 2536 2567 2537 2633 2558 2559  7.440923 0.489891 SUCCESS 146
111111110 2565 2663 2511 2580 2614 2620 2618 2568  9.449332 0.305819 SUCCESS 147
```

Test ujemanja s predlogami s prekrivanjem

```
    OVERLAPPING TEMPLATE OF ALL ONES TEST
---------------------------------------------
COMPUTATIONAL INFORMATION:
---------------------------------------------
(a) n (sequence_length)      = 10485760
(b) m (block length of 1s)   = 9
(c) M (length of substring)  = 1032
(d) N (number of substrings) = 10160
(e) lambda [(M-m+1)/2^m]     = 2.000000
(f) eta                      = 1.000000
---------------------------------------------
   F R E Q U E N C Y
  0   1   2   3   4 >=5   Chi^2   P-value  Assignment
---------------------------------------------
3747 1865 1402 1019 705 1422  0.154139 0.999530 SUCCESS
```

Maurerjev "univerzalni" statistični test

```
UNIVERSAL STATISTICAL TEST
-----------------------------------------
COMPUTATIONAL INFORMATION:
-----------------------------------------
(a) L         = 10
(b) Q         = 10240
(c) K         = 1038336
(d) sum       = 9523286.685954
(e) sigma     = 0.001128
(f) variance  = 3.356000
(g) exp_value = 9.172324
(h) phi       = 9.171681
(i) WARNING:  0 bits were discarded.
-----------------------------------------
SUCCESS p_value = 0.568600
```

Test linearne kompleksnosti

```
-----------------------------------------------------
L I N E A R  C O M P L E X I T Y
-----------------------------------------------------
M (substring length)     = 500
N (number of substrings) = 20971
-----------------------------------------------------
        F R E Q U E N C Y
-----------------------------------------------------
  C0   C1   C2   C3   C4   C5   C6    CHI2    P-value
-----------------------------------------------------
Note: 260 bits were discarded!
 230  628 2641 10392 5333 1331  416  5.484500 0.483332
```

Serijski test

```
       SERIAL TEST
-------------------------------------------
 COMPUTATIONAL INFORMATION:
-------------------------------------------
(a) Block length    (m) = 16
(b) Sequence length (n) = 10485760
(c) Psi_m               = 64616.100000
(d) Psi_m-1             = 32328.931250
(e) Psi_m-2             = 16342.128125
(f) Del_1               = 32287.168750
(g) Del_2               = 16300.365625
-------------------------------------------
SUCCESS p_value1 = 0.970279
SUCCESS p_value2 = 0.676927
```

normalsize

Test približne entropije

```
APPROXIMATE ENTROPY TEST
-----------------------------------------
COMPUTATIONAL INFORMATION:
-----------------------------------------
(a) m (block length)    = 10
(b) n (sequence length) = 10485760
(c) Chi^2               = 1083.268074
(d) Phi(m)        = -6.931424
(e) Phi(m+1)        = -7.624519
(f) ApEn                = 0.693096
(g) Log(2)              = 0.693147
-----------------------------------------
SUCCESS p_value = 0.096823
```

Test kumulativnih vsot

```
      CUMULATIVE SUMS (FORWARD) TEST
-----------------------------------------
COMPUTATIONAL INFORMATION:
-----------------------------------------
(a) The maximum partial sum = 3725
-----------------------------------------
SUCCESS p_value = 0.498891

      CUMULATIVE SUMS (REVERSE) TEST
-----------------------------------------
COMPUTATIONAL INFORMATION:
-----------------------------------------
(a) The maximum partial sum = 2318
-----------------------------------------
SUCCESS p_value = 0.885368
```

Test naključnega oddaljevanja

```
  RANDOM EXCURSIONS TEST
--------------------------------------------
COMPUTATIONAL INFORMATION:
--------------------------------------------
(a) Number Of Cycles (J) = 1508
(b) Sequence Length (n)  = 10485760
(c) Rejection Constraint = 500.000000
--------------------------------------------
SUCCESS x = -4 chi^2 =  9.387346 p_value = 0.094576
SUCCESS x = -3 chi^2 =  5.067548 p_value = 0.407692
SUCCESS x = -2 chi^2 =  9.361365 p_value = 0.095490
SUCCESS x = -1 chi^2 =  2.132626 p_value = 0.830506
SUCCESS x =  1 chi^2 =  1.867374 p_value = 0.867175
SUCCESS x =  2 chi^2 =  3.493074 p_value = 0.624436
SUCCESS x =  3 chi^2 =  5.138625 p_value = 0.399198
SUCCESS x =  4 chi^2 =  1.676395 p_value = 0.891860
```

Varianta testa naključnega oddaljevanja

```
RANDOM EXCURSIONS VARIANT TEST
--------------------------------------------
COMPUTATIONAL INFORMATION:
--------------------------------------------
(a) Number Of Cycles (J) = 1508
(b) Sequence Length (n)  = 10485760
--------------------------------------------
SUCCESS (x = -9) Total visits = 1392; p-value = 0.608446
SUCCESS (x = -8) Total visits = 1373; p-value = 0.525620
SUCCESS (x = -7) Total visits = 1413; p-value = 0.631388
SUCCESS (x = -6) Total visits = 1447; p-value = 0.737699
SUCCESS (x = -5) Total visits = 1476; p-value = 0.845997
SUCCESS (x = -4) Total visits = 1504; p-value = 0.978038
SUCCESS (x = -3) Total visits = 1460; p-value = 0.695888
SUCCESS (x = -2) Total visits = 1454; p-value = 0.570240
SUCCESS (x = -1) Total visits = 1515; p-value = 0.898574
SUCCESS (x =  1) Total visits = 1489; p-value = 0.729366
SUCCESS (x =  2) Total visits = 1495; p-value = 0.891293
SUCCESS (x =  3) Total visits = 1486; p-value = 0.857818
SUCCESS (x =  4) Total visits = 1412; p-value = 0.508803
SUCCESS (x =  5) Total visits = 1382; p-value = 0.444406
SUCCESS (x =  6) Total visits = 1455; p-value = 0.771066
SUCCESS (x =  7) Total visits = 1531; p-value = 0.907529
SUCCESS (x =  8) Total visits = 1530; p-value = 0.917619
SUCCESS (x =  9) Total visits = 1403; p-value = 0.642853
```

Strnjeni rezultati drugega načina testiranja

Zvezdica označuje test, katerega zaporedje ni prestalo. Stolpci *C1* do *C10* prikazujejo število P-vrednosti, ki se nahajajo v posameznih podintervalih. Stolpec *P-VALUE* označuje P-vrednost za χ^2-test enakomerne porazdelitve. Stolpec *PROPOTION* delež P-vrednosti nad $\alpha = 0.01$.

```
-------------------------------------------------------------------------------
RESULTS FOR THE UNIFORMITY OF P-VALUES AND THE PROPORTION OF PASSING SEQUENCES
-------------------------------------------------------------------------------
   generator is <koncna>
-------------------------------------------------------------------------------
 C1  C2  C3  C4  C5  C6  C7  C8  C9 C10  P-VALUE  PROPORTION  STATISTICAL TEST
-------------------------------------------------------------------------------
 98  93  88  98 113 100  90 104 119  97  0.478839   0.9910    Frequency
 91 120  92 102  94  95  99 121  93  93  0.242986   0.9900    BlockFrequency
105  80  89 100  97  93  93 120 102 121  0.091487   0.9940    CumulativeSums
 94  97  99  96  98  83 118 116  88 111  0.213309   0.9930    CumulativeSums
 86 105 106 103  95 103  96 110 104  92  0.837781   0.9930    Runs
111  90 100 103 110  85  87 104 108 102  0.526105   0.9840    LongestRun
125 110  84  76 100  97 133  71  91 113  0.000022 * 0.9880    FFT
102  76 105  92  98 107 138  99  85  98  0.004365   0.9910    OverlappingTemplate
 93 117  96 101 101  80  97 111 118  86  0.120207   0.9870    Serial
106 117  98 116  86  99  93 102  90  93  0.363593   0.9880    Serial
```

8.6 Naslov študenta

Gregor Donaj
Moškanjci 50
2272 Gorišnica

8.7 Življenjepis študenta

Rojen:

- 14. januar 1986, Ptuj

Šolanje:

- 1993 – 2001 Osnovna šola Gorišnica
- 2001 – 2005 II. gimnazija Maribor
- 2005 – 2010 Fakulteta za elektrotehniko, računalništvo in informatiko, Maribor
- 2006 – Fakulteta za naravoslovje in matematiko, Maribor

8.8 Vsebina zgoščenke

Vsebina zgočenke je razdeljena na spodaj naštete mape. V vsakem imeniku se nahaja datoteka `Preberi me.pdf` s podrobnejšim opisom vsebine mapa.

- **Diplomsko delo** - besedilo diplomskega dela v formatu .pdf, izvorni dokumenti za LaTeX in vse slike v .eps formatu.
- **Literatura** - prosto dostopni viri iz literature.
- **Shema vezja** - sheme vezja v obliki Altium projekta.
- **Simulacije vezja** - datoteke za simulacije vezja v programu SPICE.
- **Simulacije enačbe** - skripte za simulacije enačbe in risanje grafov iz diplomske za program MATLAB.
- **Statistical Test Suite** - izvorna koda za program s statističnimi testi in preveden program za okolje Windows.
- **Podatki** - Vsa zajeta števila iz vezja in pretvorjena števila.
- **Rezultati** - Rezultati statističnih analiz.
- **Programi** - Izvorna koda za program mikrokrmilnika (CodeVision AVR), sprejemnik podatkov na osebnem računalniku (Visual Basic 6) in pretvornik števil v različne variante (Visual C++ 6).
- **Meritve** - osciloskopski posnetki merjenja prototipa.

UNIVERZA V MARIBORU
Fakulteta za elektrotehniko, računalništvo in informatiko
(ime fakultete)

IZJAVA O ISTOVETNOSTI TISKANE IN ELEKTRONSKE VERZIJE DIPLOMSKEGA DELA IN OBJAVI OSEBNIH PODATKOV AVTORJA

Ime in priimek avtorja (avtorice): Gregor Donaj
Vpisna številka: 93594188
Študijski program: FERI-E UNI ELEKTRONIKA
Naslov diplomskega dela: GENERATOR NAKLJUČNIH ŠTEVIL REALIZIRAN S KAOTIČNIM OSCILATORJEM

Mentor: Tomaž Dogša
Somentor:

Podpisani-a Gregor Donaj izjavljam, da sem za potrebe arhiviranja oddal-a elektronsko verzijo diplomskega dela v Digitalno knjižnico Univerze v Mariboru. Diplomsko delo sem izdelal-a sam-a ob pomoči mentorja. V skladu s 1. odstavkom 21. člena Zakona o avtorskih in sorodnih pravicah (Ur. l. RS, št. 16/2007) dovoljujem, da se zgoraj navedeno diplomsko delo objavi na portalu Digitalne knjižnice Univerze v Mariboru.

Tiskana verzija diplomskega dela je istovetna elektronski verziji, ki sem jo oddal-a za objavo v Digitalno knjižnico Univerze v Mariboru. Podpisani-a izjavljam, da dovoljujem objavo osebnih podatkov, vezanih na zaključek študija (ime, priimek, leto in kraj rojstva, datum zagovora, naslov zaključnega dela) na spletnih straneh in v publikacijah UM.

Kraj in datum: Maribor, 29.08.2010

Podpis avtorja (avtorice):

FERI
Fakulteta za elektrotehniko, računalništvo in informatiko
Smetanova ulica 17
2000 Maribor

IZJAVA O USTREZNOSTI DIPLOMSKEGA DELA

Podpisani mentor Tomaž Dogša izjavljam, da je
(ime in priimek mentorja)
študent Gregor Donaj izdelal diplomsko
(ime in priimek študenta-tke)

delo z naslovom: Generator naključnih števil realiziran s kaotičnim oscilatorjem
(naslov diplomskega dela)

v skladu z odobreno temo diplomskega dela, Navodili o pripravi diplomskega dela in mojimi navodili.

Datum in kraj:

27.8.2010, Maribor

Podpis mentorja: