KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
PROGRAMŲ INŽINERIJOS KATEDRA

Gediminas Dagys

MODEMO PROJEKTAVIMAS IR TYRIMAS

Magistro darbas

Vadovas
prof. dr. E. Kazanavičius

KAUNAS, 2005

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
PROGRAMŲ INŽINERIJOS KATEDRA

TVIRTINU
Katedros vedėjas
doc. E. Bareiša
2005-05-20

MODEMO PROJEKTAVIMAS IR TYRIMAS

Informatikos inžinerijos mokslo magistro baigiamasis darbas

Kalbos konsultantė
Lietuvių kalbos katedros lektorė vedėjas
dr. J. Mikelionienė
2005-05-16

Recenzentas
doc. dr. S. Maciulevičius
2005-05-20

Vadovas
Kompiuterių katedros
prof. dr. E. Kazanavičius
2005-05-19

Atliko
IFM 9/5 gr. stud.
G. Dagys
2005-05-16

KAUNAS, 2005

Summary

In this paper a modification of the DCSK (Direct chaos shift keying) is considered. This modification, instead of using the reference signal part as in DCSK, transmits the information bit without it. The receiver then uses a unique key, known only to it and the sender, to decode the bit.

The effectiveness of the system and two ways of its realization are tried, with this paper describing the experiments that are held.

The effectiveness is defined by systems capability to transmit data.

The theoretical experiments include testing five different keys and testing noise resistibility.

The practical experiments include production of two models of the system, their comparison in functional and synthesis levels.

The experiments are successful if the system model can be implemented in both ways that are tried, it is implemented in both ways, and the synthesis results are comparable.

Turinys

Lentelių sąrašas

Paveikslėlių sąrašas

1 ĮVADAS

Kiekvieną kartą atsirandant naujam duomenų perdavimo būdui, taip pat atsirasdavo keletas jo realizavimo variantų. Realizuojant skirtingas perdavimo sistemas skirtingomis technologijomis, susidarė daugybė perdavimo sistemų, technologijų, aparatūros bei algoritmų. Kitaip tariant, komunikacijų sistemos pradėjo tobulėti greičiau nei augo poreikiai. Todėl dabar pagrindinis dėmesys sutelktas į jau sukurtų protokolų, aparatūros efektyvesnį panaudojimą, tų pačių resursų efektyvesnį, taupesnį energijos požiūriu praplėtimą. Tam jau yra įdiegti ir diegiami nauji standartai. Taip pat labai svarbu perduodamos informacijos saugumas, tam naudojama daug kriptografijos, kodavimo būdų.

Egzistuojant daugybei sistemų kūrimo būdų, iškyla jų vertinimo problema, nes tą pačią sistema galima realizuoti ne tik parenkant skirtingas technologijas, bet taip pat pritaikant skirtingus algoritmus ar netgi virtualias priemones kaip, tarkime, neuroniniai tinklai.

Šiame darbe aprašomas chaosiniu signalu pagrįstos skaitmeninės sistemos kūrimo kokybės ir efektyvumo patikrinimas. Eksperimentas atliekamas sukuriant sistemos modelį panaudojant SystemC aparatūros aprašymo kalbą ir neuroprocesorių. Tokiu būdu yra teoriškai patikrinamas vieno iš chaosinių duomenų perdavimo būdų veikimas. Taip pat ištirta šiuo būdu veikiančio modemo kūrimo dviem skirtingais metodais galimybė. Tokio modemo paskirtis būtų perduoti duomenis energetiniuose tinkluose.

Pirminis projekto etapas – matematinis modelis, kurio metu aprašomos testuojamos sistemos funkcinės dalys. Jis sudarytas iš siųstuvo ir imtuvo. Duomenų perdavimui užtikrinti naudojama DCSK technologijos modifikacija, paremta chaosiniu signalu. Siunčiamo signalo gavimui užtikrinti naudojamas raktas – bitų seka, kuria koduojamas siunčiamas signalas vėliau koreliuojamas imtuve ir gaunami išsiųsti duomenys.

Sistemos veikimo principui patikrinti sukuriamas matematinis modelis tiek siųstuvui, tiek imtuvui atskirai. Teorinei perdavimo kokybei patikrinti sukuriamas matematinis modelis simuliuoti kanalui. Atliekami eksperimentai, skaičiuojamas perdavimo klaidų lygis.

Sistema aprašoma SystemC kalba. Imtuvo koreliacija taip pat realizuojama ir eksperimentiniu neuroprocesoriumi, tam kad būtų patikrintas skirtingų metodų efektyvumas. Aprašius sistemos veikimą, atliekamas modeliavimas. Jei abiejų modelių testavimo rezultatai sutampa ir tenkina DCSK modemo funkcionalumą, SystemC modelis sintezuojamas ir sintezės rezultatai lyginami su atitinkamais neuroprocesoriaus parametrais. Imtuvą realizuojant neuroniniame tinkle, naudojamas eksperimentinis neuroprocesorius pritaikytas tai pačiai technologijai.

Šio darbo tikslas yra ištirti galimo DCSK modemo funkcionalumą. Palyginti jo realizacijos efektyvumą aprašant modelį tiesiogiai ir kaip neuroninį tinklą.

Tam, kad būtų pasiektas tikslas, abu modeliai turi būti įgyvendinti iki sintezės lygmens, arba gauti atitinkami modelį apibūdinantys duomenys. Tam kad būtų galima iš sintezės rezultatų spręsti apie realizacijos efektyvumą, turi atitikti abiem būdais realizuoto modemo modelio modeliavimo rezultatai.

Matematinis modelis aprašomas MatLab matematiniu paketu. Modelis testuojamas su parinktais duomenimis.

SystemC modelis prieš sintezę testuojamas, modeliavimo metu naudojant matematinio modelio testo metu gautus duomenis. Patvirtinus modelio funkcionalumą, jis sintezuojamas parinktais įrankiais į parinktą technologiją. Sintezės įrankiai pasirenkami pagal tai, kaip jie prieinami ir kaip tinka pasirinktai technologijai. SystemC tekstas turi būti sintezuojamas.

Pagrindinis sistemos modulis - imtuvas realizuojamas neuroniniame tinkle, panaudojant neuroprocesorių, pritaikytą tai pačiai technologijai. Rezultatai turi atitikti SystemC modeliavimo rezultatus, nes tai parodo modelių funkcinį lygiavertiškumą.

Realizacijų efektyvumo palyginimui naudojami SystemC sintezės analizės duomenys, bei tas pačias charakteristikas nusakantys pasirinkto neuroprocesoriaus parametrai.

2 ANALIZĖ

2.1 Chaosinė duomenų perdavimo sistema

Skaitmeninėse duomenų perdavimo sistemose dvejetainė informacija yra perduodama iš vienos vietos į kitą priskiriant bitų sekas simboliams, o simbolius įvairioms analoginių funkcijų charakteristikoms; papratai tokiose sistemose naudojamos periodinės funkcijos gali būti išreikštos kaip sinusoidžių suma, tuo tarpu chaosinėse duomenų perdavimo sistemose naudojamos funkcijos yra neperiodinės ir chaotiškos [3].

Daugelis modernių komunikacijų sistemų, tokių kaip mobilieji telefonai arba radijas kenčia nuo efekto atsirandančio priimant tą patį signalą, imtuvą pasiekiantį skirtingais keliais. Tokiais atvejais chaosinės sistemos gali pateikti geresnius rezultatus, nes chaosinio signalo segmentų tarpusavio koreliacija yra daug mažesnė negu periodiško signalo [3].

Chaosinio signalo spektrinio tankio galia yra nekintanti, o energija yra paskirstyta per visą dažnių juostą. Be to, chaosiniams signalams generuoti užtenka paprastų analoginių grandinių [3]. Buvo sukurta daug technologijų, pvz., PSK, CSK, TR ir kt. Jos pasižymi kokybišku duomenų atkūrimu perduodant informaciją, iki 15dB BER, efektyvumu, sąlyginai nedidele projektavimo bei realizavimo kaina. Kaip minėta, technologijų yra daug, visos jos turi savus trūkumus ir viena kitą papildo. Pvz. DCSK, naudojančią kartojamą nešantį signalą, papildo CDSK sistema naudodama sumatorių.

Keletas bendrų trūkumų būtų sinchronizavimas, nedidelis greitis dėl skaičiavimų apimties, kadangi reikia realiu laiku skaičiuoti koreliaciją, bei ne pats geriausias atsparumas triukšmams lyginant su kitomis šiuolaikinėmis sistemomis, kai nėra svarbus daugelio perdavimo kelių aspektas.

Viena iš naujų, sparčiai besivystančių chaosinės duomenų perdavimo sistemos taikymo sričių yra informacijos perdavimas energetiniais tinklais. Čia pagrindinė problema iškyla dėl daugelio tuo pačiu kanalu besinaudojančių siųstuvų bei imtuvų interferencijos, todėl chaosinio signalo koreliacinės savybės tampa labai naudingos.

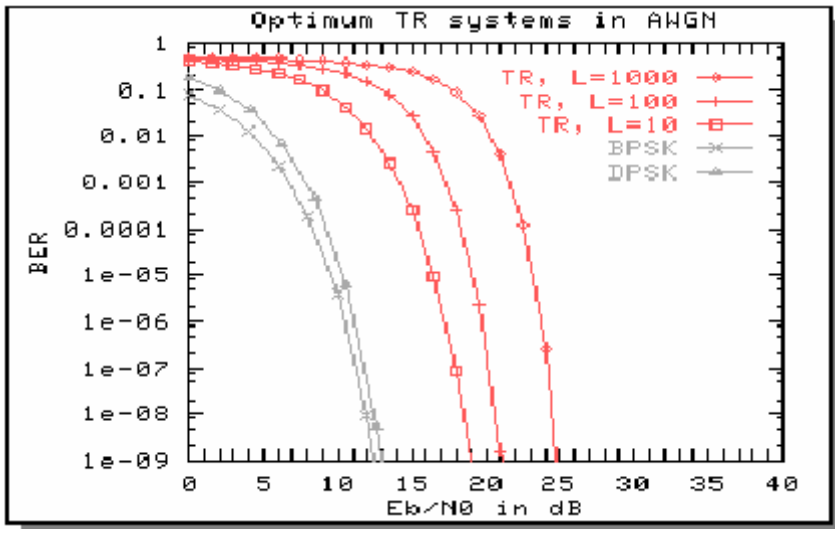
2.2 Technologijų apžvalga

2.2.1 PCSK

Viena iš pirmųjų technologijų, kurių pagrindu buvo pradėtos kurti chaosinės duomenų perdavimo sistemos, buvo PSK – *phase shift keying*. Ši technologija naudoja sinusinio signalo fazes. Pats paprasčiausias metodas naudoja dvi fazes: 0º ir 180º. Sinuso signalas diskretizuojamas. Kiekvieno bito būseną apibūdina ankstesnis bitas. Jei sinusoidės fazė nekeičiama, tai vadinasi reikšmė ir lieka tokia, kokia buvo. Jei fazė pasikeičia 180º, vadinasi bito būsena keičiama iš aukštos į žemą arba atvirkščiai. Didinant fazių skaičių gaunama sudėtingesnė šios technologijos realizacija. Šio tipo technologijos nėra efektyvios, bet jos ypač atsparios triukšmams. Kaip teigiama [1], BER (*bit error rate*, klaidingai perduodamų bitų skaičius) arba signalo ir triukšmo santykis PSK technologijai yra 13,3 dB.

2.2.2 TR

TR (*transmitters systems*) perdavimo technologijoje nešančiajam dažniui naudojamas radijo dažnio diapazono triukšmas. Radijo dažnio triukšmo signalas naudojamas kaip atraminis. Ši sistema nėra efektyvi, reikalauja galingo nešančiojo signalo, tačiau ši sistema yra atspari triukšmams. Lyginant su BPSK, TR sistema BER parametru ją lenkia.



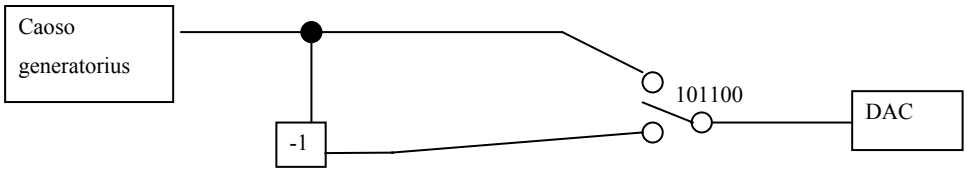
1 pav. TR atsparumo triukšmams charakteristika [1]

Nors ir pasižyminti geromis atsparumo triukšmams charakteristikomis (1 pav.), ši technologija informacijos perdavimui jėgos linijomis netinka, nes joje nėra sprendžiama daugelio vienu metu veikiančių kanalų problema.

2.2.3 DCSK

Viena iš naujesnių technologijų chaosiniu signalu pagrįstose sistemose – DCSK technologijų grupė (*Differential chaos shift keying*). Kaip teigiama [3], šios technologijos pagalba išgaunamas vienas geriausių BER. Pagrindinis šios sistemos privalumas – priešpriešinė moduliacija, tai suteikia maksimalų atstumą tarp signalo elementų. Tam tikrose taikymų srityse, kaip tarkime informacijos perdavimas jėgos linijomis, lyginant su įprastinėmis sinusoidinėmis sistemomis, ši sistema žymiai atsparesnė triukšmams. Daug šios sistemos patobulinimų šalinant trūkumus sukūrė naujas technologijos pakraipas kaip FM-DCSK, CDMA, SD-DCSK, QCSK.

Sistemos veikimas paremtas nešančiojo chaosinio signalo segmentų koreliacija. Kiekvienas siunčiamas informacinis bitas atvaizduojamas dvejomis chaosinėmis funkcijomis [2]. Pirma funkcija vadinama atraminiu arba nešančiuoju signalu, kita priklauso nuo pernešamo skaitmeninės informacijos turinio. Siunčiant informacinį bitą, pirmąją jo siuntimo periodo pusę į išėjimą tiesiogiai perduodamas chaoso generatoriaus išėjimas (nešantysis signalas) (2 pav.), antroji periodo pusė priklauso nuo siunčiamo bito reikšmės.



2 pav. DCSK veikimo proncipas

Persiunčiant „1“ nešantysis signalas pakartojamas dar kartą, siunčiant „0“ antroji bito periodo dalis gaunama invertuojant nešantįjį signalą.

Atkoduojant, gautas signalas koreliuojamas su savimi, tik pavėlintu per puse bito trukmės periodo. Teigiama koreliaciją reiškia „1“, neigiama koreliacija reiškia „0“.

Modifikuojant DCSK technologiją galima gauti kitus duomenų perdavimo būdus, atitinkančius specifinius poreikius, tokius kaip atsparumas triukšmams arba daugelio kanalų vienoje dažnių juostoje galimybė.

2.2.4 FM-DCSK

Siekiant dar labiau pagerinti DCSK sistemos savybes, tokias kaip atsparumas triukšmams, duomenų perdavimo sparta, buvo sukurta FM–DCSK technologija, kurios pagrindas - dažninė chaosinio signalo moduliaciją. Chaosinį signalą siųstuve generuoja analoginė fiksuotos fazės grandinė. FM-DCSK imtuve koreliuojama tiesiogiai, kaip ir DCSK.

Šios technologijos atsparumo triukšmams savybės yra panašios i DCSK. Tačiau FM-DCSK leidžia pasiekti didesnį duomenų srautą. Taip yra dėl to, kad čia visada yra pastovi perduodamo bito energija, o DCSK, siunčiant chaosinį signalą tiesiogiai (nemoduliuotą), bito energija priklauso nuo paties chaosinio signalo savybių.

Šiuo atveju moduliuojamas signalas palyginus su nešančiuoju kinta lėtai, todėl yra išgaunamas pastovus signalo galingumas, priklausantis tik nuo nešančiojo signalo dažnio ir galingumo [3].

Šiame darbe pasirinktos technologijos duomenų perdavimo sparta nėra svarbi, o dėl reikalingo papildomo dažninio moduliatoriaus, siųstuvo projektavimas tampa sudėtingesnis, todėl šiuo atveju šią technologiją pasirinkti kaip tyrimo objektą nėra prasminga.

2.2.5 QCSK

Norint padidinti perduodamų duomenų srauto greitį, buvo sukurta QCSK (*Quadrature chaos shift keying*) technologija [4]. Čia pagrindinė idėja yra dviejų ortogonalių chaosinių signalų generavimas. Kiekvienas signalas tuo pačiu metu panaudojamas išsiųsti skirtingam bitui. Kadangi abu signalai yra ortogonalūs, jie vienas kito neslopina.

QCSK galima užkoduoti du bitus vienu simboliu. Ortogonalūs chaosiniai signalai sukuriami panaudojant Furjė bei Hilberto transformacijas. Signalas siunčiamas kaip ir DCSK atveju, tik dalinamas į dvi dalis kas pusę periodo. Informacinis signalas generuojamas Hilberto transformacijos pagalba. Kaip ir DCSK atveju, demoduliuojant informaciją, informacija atkuriama informacinei daliai koreliuojantis su atramine, tik čia vykdomos dvi koreliacijos vienu metu, vienoje iš jų naudojant signalą perleistą per Hilberto filtrą [4].

QCSK technologija yra pranašesnė už DCSK perduodamų duomenų sparta, tačiau didesnis atsparumas triukšmams pasireiškia tik esant ilgesniam koreliacijos periodui (vėlinimui) [4]. Šios technologijos realizaciją yra dar sudėtingesnė negu FM-DCSK, todėl ją pasirinkti kaip tyrimo objekta taip pat neprasminga.

2.3 Chaosinio signalo generavimas

Sistemoms tampant sudėtingesnėms ir mažesnėms, šie procesai neaplenkė ir chaosiniu signalu pagrįstųjų. Pagrindinė problema pritaikyti chaosinę sistemą KMOP technologijai yra chaosinio signalo generatorius. Šiai problemai spręsti sukurti būdai yra pvz. Bernulio perstūmimas arba LFSR (*Linear feedback shift register*).

Vienas iš LFSR arba pseudochaosinio signalo panaudojimo pavyzdžių yra SD-DCSK technologija [5]. Čia įprastinis chaosinio signalo generatorius pakeičiamas pseudochaosiniu, kurio išėjimo geresnėm atsitiktinumo charakteristikom pasiekti, generuojamas signalas yra priklausomas nuo įeinančių bitų sekos.

Analoginių chaosinio signalo generavimo schemų panaudojimas vienlustėse sistemose yra labai nepraktiškas, nes jos dažniausiai turi būti realizuotos išorėje, tuo tarpu pseudochaosinio signalo generatorius realizuojamas skaitmeninėje logikoje standartiniais komponentais.

2.4 Rakto panaudojimas

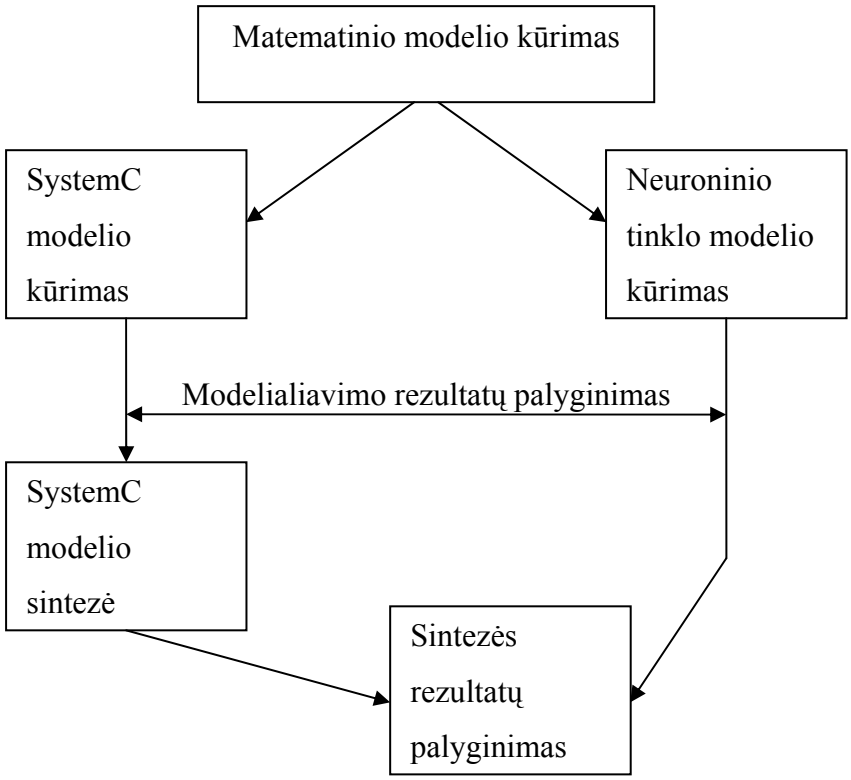
Vienas iš pagrindinių chaosinės duomenų perdavimo sistemos privalumų yra galimybė vienu metu naudoti daug perdavimo kanalų plačiame spektre, tam jo neskaidant į dažnių juostas. Tai gali būti pasiekta panaudojant DCSK sistemos modifikaciją, nenaudojančią atraminės dalies. Vietoje atraminės dalies, kiekviena siųstuvo imtuvo pora turi identišką, tačiau skirtingą nei visų kitų porų raktą. Kadangi teorinis skirtingų rakto kombinacijų skaičius yra begalinis, kanalų skaičiaus riba tampa priklausoma tik nuo triukšmingumo ir tinkamo skirtingų raktų parinkimo.

Raktų generavimui gali būti panaudotas tiek pseudochaosinio signalo generatorius tiek periodinės, eksponentinės ar kitokios funkcijos.

Taigi šiame darbe tiriamas modemas, kurio atraminė signalo atkarpa užuot perduodama kartu su signalu prieš kiekvieną siunčiamą bitą, yra išsaugoma identiška tiek siųstuvo, tiek imtuvo atmintyje ir naudojama ta pati kiekvienam bitui. Tai veikia kaip apsaugos priemonė, nes imtuvas atkoduos tik identišką raktą turinčio siųstuvo signalą. Taip pat tokiu būdu dvigubai padidėja duomenų perdavimo sparta, nes nebereikalinga atraminė dalis. Daugelis tokių siųstuvo imtuvo sistemų turinčių skirtingus raktus gali naudotis ta pačia dažnių juosta viena su kita nesąveikaudamos.

2.5 Darbo struktūra

Darbas vykdomas trimis etapais. Pradedant teoriniu modeliu ir baigiant sinteze.



3 pav. Darbo struktūra

Matematiniame modelyje aprašomas DCSK modemo funkcionalumas. Patikrinamas jo veikimas, duomenų siuntimo priėmimo galimybė, atsparumas triukšmams, rakto formos įtaka. Šie parametrai nusako ar prototipas, kurio realizacijos efektyvumas bus tikrinamas, gali būti traktuojamas kaip modemas.

Modelio kūrimo fazė išsiskiria į du etapus: SystemC modelio kūrimą, ir neuroninio tinklo modelio kūrimą. Neuroniniame tinkle realizuojamas tik imtuvo modelis, nes to pakanka tinklo panaudojimo tinkamumui patvirtinti.

SystemC modelio sintezė vykdoma pasitelkiant *Synopsys* sintezatorių. Sintezės analizei reikalingi duomenys gaunami pasitelkiant *Synopsys design_analyzer* įrankį.

Neuroninio tinklo, realizuoto kaip vienlustis neuroprocesorius, atitinkami parametrai gaunami iš neuroprocesoriaus kūrėjo.

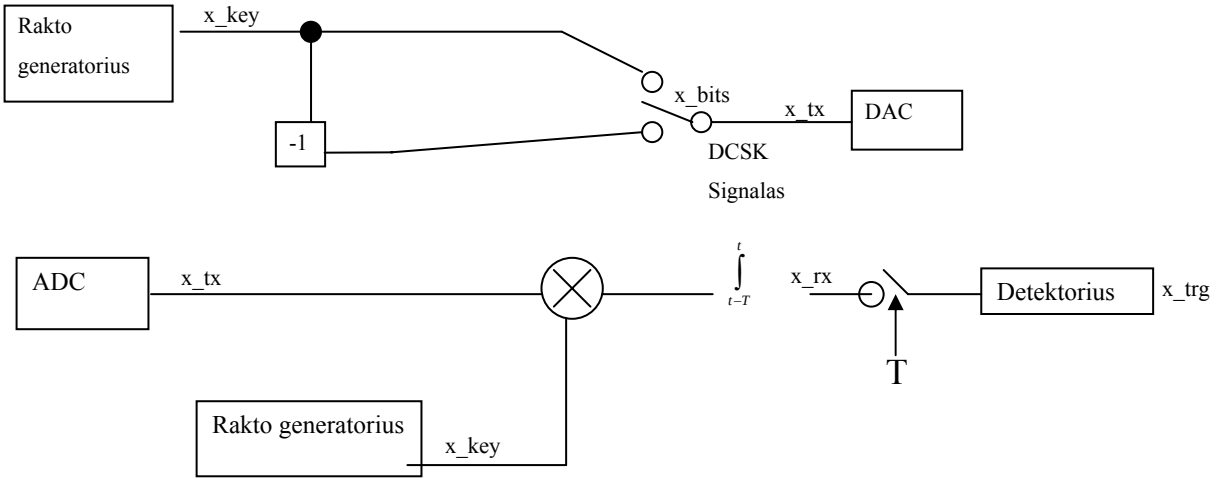
Palyginami susintezuoto, prie pasirinktos bibliotekos pririšto SystemC modelio sintezės parametrai su prie tos pačios bibliotekos pririšto neuronų procesoriaus sintezės parametrais.

3 TEORINĖ DALIS

3.1 Matematinis modelis

Modeliavimui panaudojamas Matlab 6 matematinių skaičiavimų paketas. Modelis skirtas patikrinti teorinį tokios sistemos funkcionalumą prieš pradedant SystemC ir neuroprocesoriaus modelių kūrimą.

Sistemos modelis realizuojamas dviem etapais – atskirai siųstuvas bei imtuvas (4 pav.).



4 pav. Sistemos funkcinė diagrama

Čia:

T = s_klen	- rakto ilgis žodžiais
x_key[1..T]	- raktas
x_tx[1..n]	- siunčiamas signalas
x_rx[1..n]	- koreliacijos rezultatas
x_trg[1..n]	- detektoriaus rezultatas
x_bits[1..n/T]	- perduodamų bitų reikšmės

Siųstuve kiekvienam bitui užkoduoti panaudojama tapati nešančiojo signalo atkarpa – raktas. Šis raktas atsižvelgiant į perduodamo bito logines reikšmes siųstuve yra invertuojamas arba ne.

Imtuve gaunamas signalas nuolatos koreliuojamas su turimu raktu. Priklausomai nuo to koks bitas yra perduodamas, koreliacijos rezultate gaunamas teigiamas arba neigiamas pikas.

Pasitelkiant elementarų komparatorių ar sudėtingesnį skaitmeninį palyginimo algoritmą iš koreliacinės kreivės gaunamas signalas labai panašus į pradinę siunčiamų bitų seką.

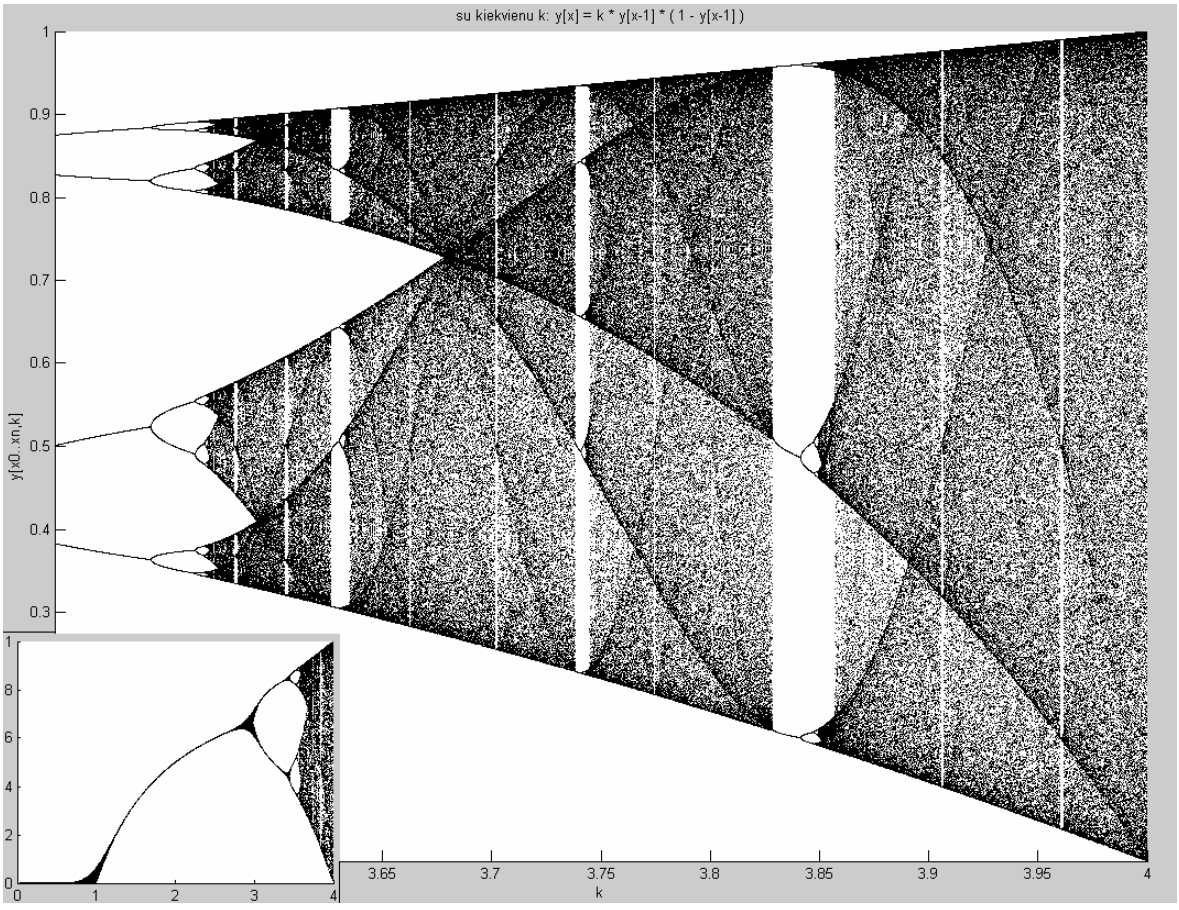
3.2 Siųstuvas

Siųstuvas sudarytas iš rakto generatoriaus, inverterio bei jungiklio. Bito “1“ atveju generuojama rakto funkcija. Jei siunčiamas „0“ rakto funkcija invertuojama, bet raktas naudojamas tas pats.

Kaip raktas sistemai naudojama pseudochaosinė funkciją:

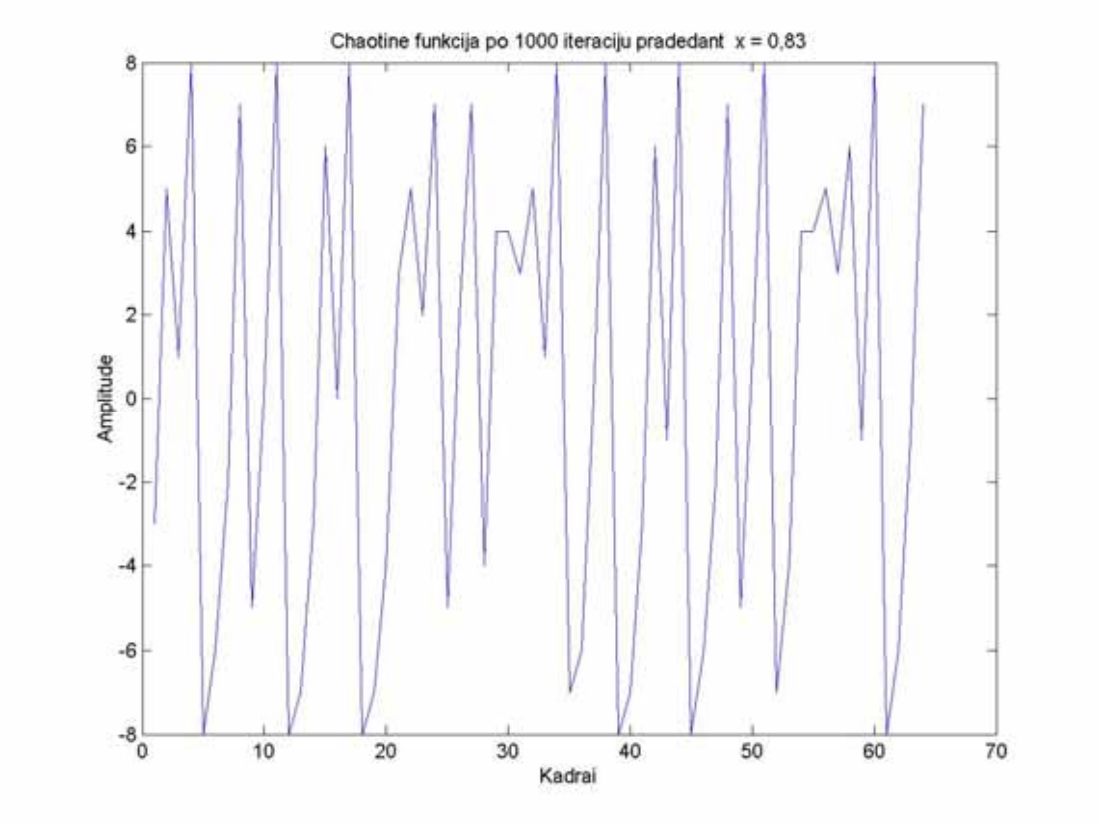
$$y[x] = k \cdot y[x-1] \cdot (1 - y[x-1]) \qquad (1)$$

Chaosinės funkcijos charakteristikos atsiskleidžia koeficientui k artėjant į 4. Ir ties 4 pasiekia optimumą. Kai k > 4 funkcija tampa nestabili. Atliekant iteracijas su koeficientu 4, nuo ~100-tosios iteracijos funkcijos reikšmės pasiskirsto tarp 0 ir 1 maždaug tolygiai, pseudoatsitiktine tvarka. Šios funkcijos charakteristikas galima apibūdinti jos bifurkacine kreive (5 pav.).



5 pav. Pseudochaosinės funkcijos bifurkacinė kreivė

Pseudochaosinė funkcija su koeficientu k=4:



6 pav. Pseudochaosinė funkcija

Čia kadrai atitiktų 5 pav. pavaizduotas y reikšmes tiesk k=4.

Be chaosinės funkcijos raktui generuoti taip pat panaudotos keletas kitų funkcijų:

Naudojami parametrai:

$T = s\_klen$ - rakto ilgis žodžiais

$i = 0,1,2..T$

$t(i) = i \cdot (\pi / T)$

Rakto funkcijos:

$x\_key[i] = \sin(t \cdot i \cdot (i/4))$ (2)

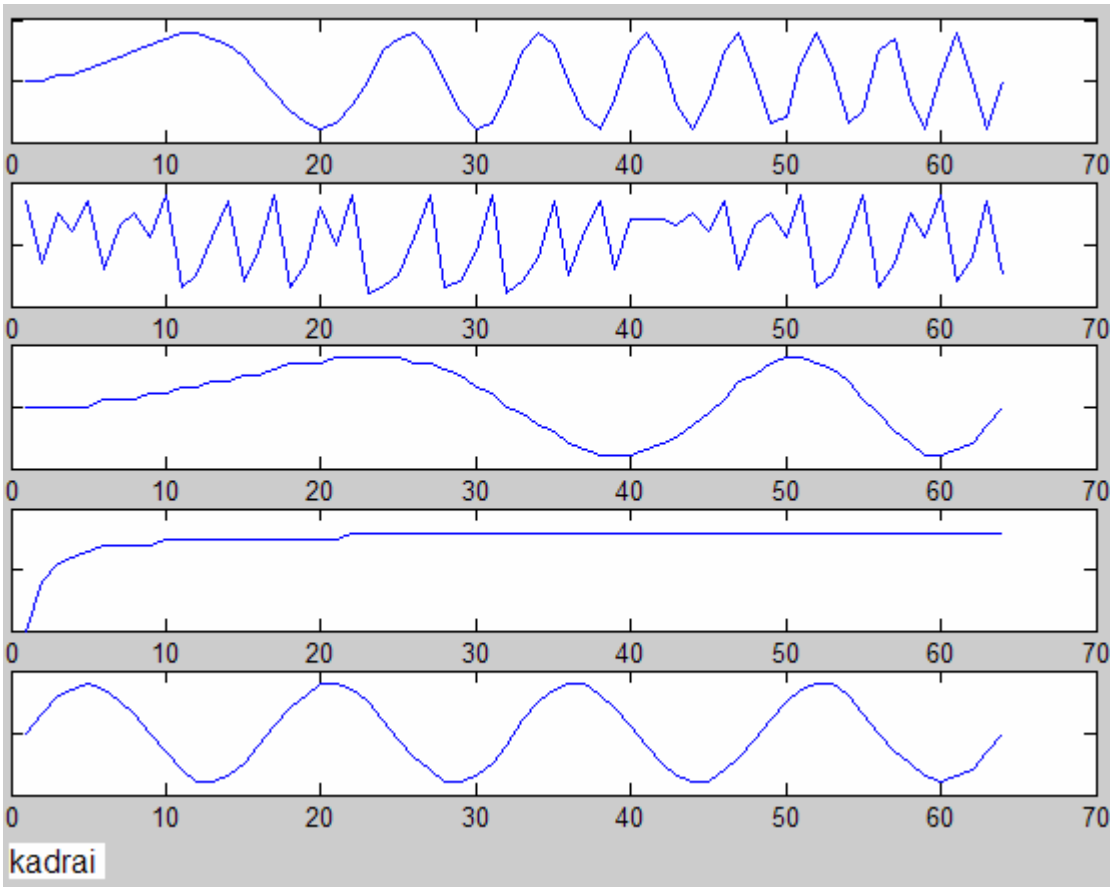
$x\_key[i] = 4 \cdot x\_key[i-1] \cdot (1 - x\_key[i-1])$ (3) (žr.(1))

$x\_key[i] = \sin(t(i) \cdot (i/16))$ (4)

$x\_key[i] = \sin(t(i) \cdot (1/i))$ (5)

$x\_key[i] = \sin(8t(i))$ (6)

Pagal funkcijas (2),(3),(4),(5),(6) gautos raktų sekos (7 pav.):



7 pav. Rakto funkcijų grafikas

Inverteris matematiniame modelyje realizuotas kaip daugyba iš -1. Siunčiant "0" siunčiama invertuota rakto funkcija:

$i = 0,1,2..n$

$$x\_tx[i] = \begin{cases} x\_key[i \bmod T] & , x\_bits[i/T] = 1 \\ -x\_key[i \bmod T] & , x\_bits[i/T] = 0 \end{cases} \qquad (7)$$

Kaip matyti 4 pav., siųstuvas praktiškai susideda iš dviejų elementų – inverterio ir vieno bito multiplekserio. MatCad priemonėmis jis modeliuojamas panaudojant vektorius duomenų saugojimui.

Siųstuvo signalas kvantuojamas keturiais bitais. Tai reiškia kad jo išėjimo reikšmių (sveikų skaičių) kitimo diapazonas bus nuo -8 iki 7.

3.3 Kanalas

Taip pat įvertinami kanalo (9 pav.) triukšmai bei slopinimas. Matematiniame modelyje baltam Gauso triukšmui generuoti naudojama standartinė *random* funkcija.

Kanalo įėjimo išėjimo funkciją gali išreikšti taip:

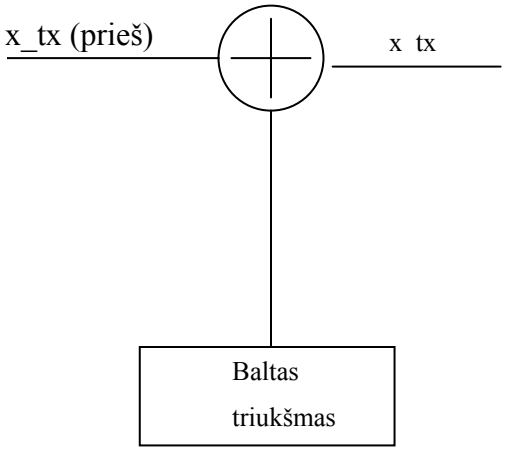
$$x\_tx[i] = x\_tx[i] + ((rand - 0.5) \times am \times nl) \qquad (8)$$

Čia:

rand – triukšmo generavimui naudojama funkcija,

am – siunčiamo signalo amplitudė,

nl – triukšmo lygis (santykis su **am**)



8 pav. Kanalo funkcinė diagrama

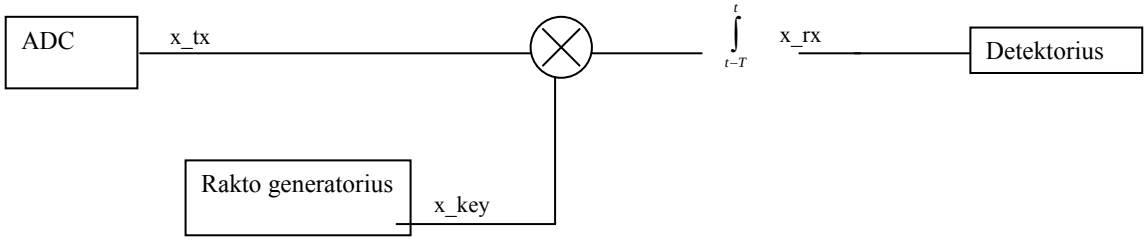
Triukšmo lygis **nl** nurodo santykį, kurį sudaro triukšmo amplitudė su signalo amplitude.

Kanalas realizuotas atskira funkcija, todėl yra lengvai parametrizuojamas ir tinka tolesniems atsparumo triukšmams tyrimams.

3.4 Imtuvas

Imtuvas sudarytas iš rakto generatoriaus, kuris koreliuojamas su gautu signalu duomenų atkūrimui, sumatoriaus ir detektoriaus, kuris pagal lygį nustato gaunamos informacijos turinį t.y. “1” arba “0”.

Imtuvo blokinė diagrama (10 pav.):



9 pav. Imtuvo funkcinė diagrama

x_trg

Rakto generatorius naudojamas tas pats kaip ir siųstuve, šiuo atveju tai atmintyje įrašyta kadrų seka. Gaunamas signalas koreliuojamas su raktu, t.y. kiekvienas rakto kadras dauginamas paeiliui su gauto signalo (kartu su triukšmais) kadrais, o gautas rezultatas nuolatos sumuojamas. Koreliacija vykdoma pradedant kiekvienu gauto signalo kadru.

Kaip parodyta formulėje (9), ties kiekvienu įeinančiu kadru, vykdoma koreliacija, atitinkanti vieno perduodamo bito ilgį.

$$x\_rx[i] = \sum_{k=1}^{T} x\_key[k] \cdot x\_tx[i+k], \text{ kai } i = 1,2,3..n \qquad (9)$$

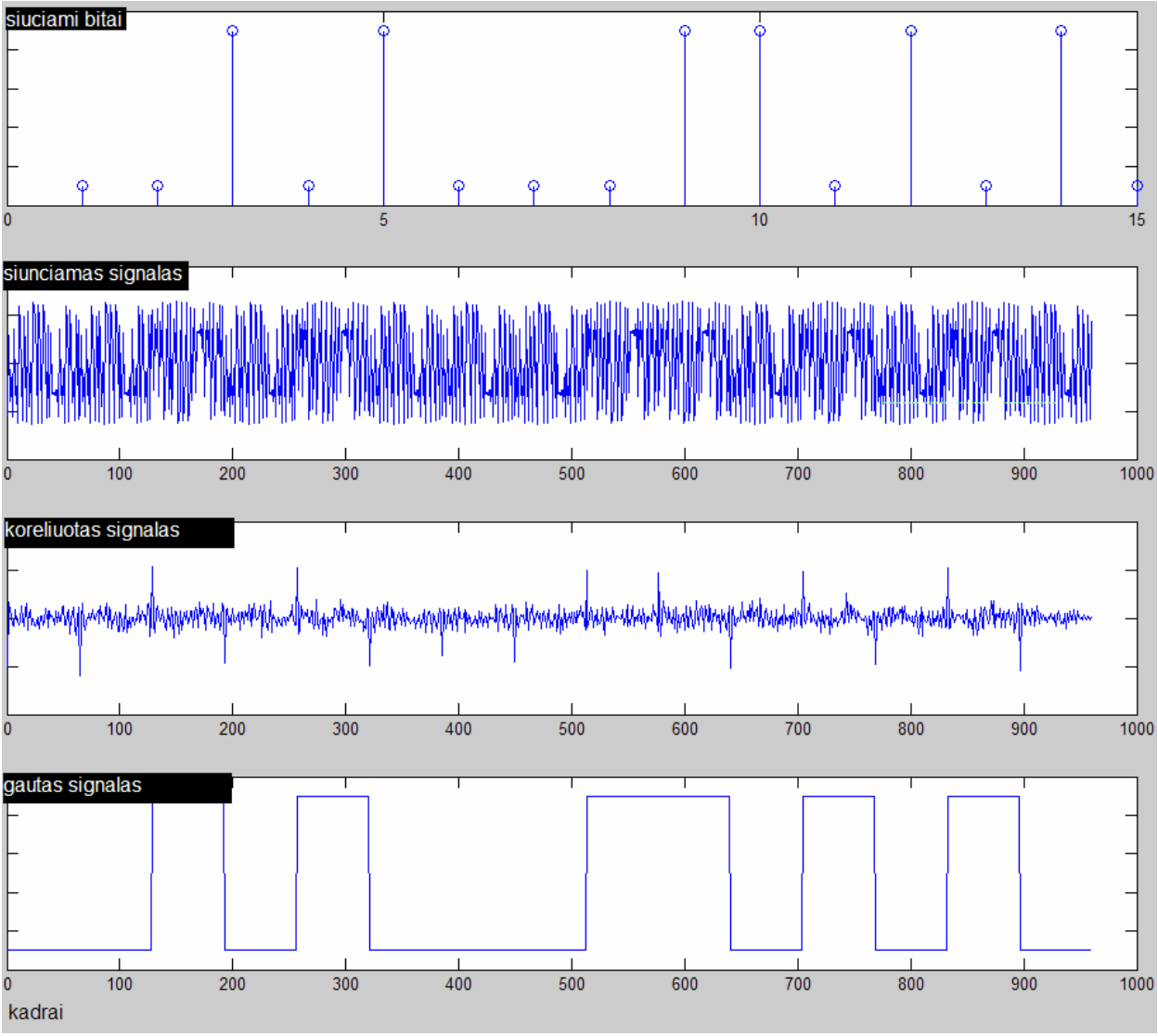
Čia:

T – koreliacijos kadrų skaičius, vieno perduodamo bito ilgis kadrais.

“1” ir “0” atpažinimui naudojamas lygmens metodas. Tikslui pasiekti galima panaudoti paprasčiausia komparatorių, tačiau siekiant išgauti didesnį atsparumą triukšmams, galima naudoti ir sudėtingesnį adaptyvų algoritmą. Paprastu atveju, jei pastaroji reikšmė didesnė už nustatytą lygi, tai priskiriamas “1”, jei mažesnė, tai priskiriamas “0”. Signalo transformacijos matomos (11 pav.). Čia ketvirtasis signalas gaunamas iš trečiojo pagal (10) formulę (11 pav.).

$$x\_trg[i] = \begin{cases} 1 & , x\_rx[i] > lev \\ 0 & , x\_rx[i] < -lev \end{cases}, \text{ kai } i = 1,2,3..n \qquad (10)$$

Čia lev – lygmuo, modelyje išreiškiamas procentais nuo amplitudės.



10 pav. Rakto generatoriaus funkcija bei gautas, koreliuotas ir detektuotas signalai

Čia „gautas signalas“ atitinka *x_trg*. Toliau rezultatai bus lyginami su koreliuotu signalu.

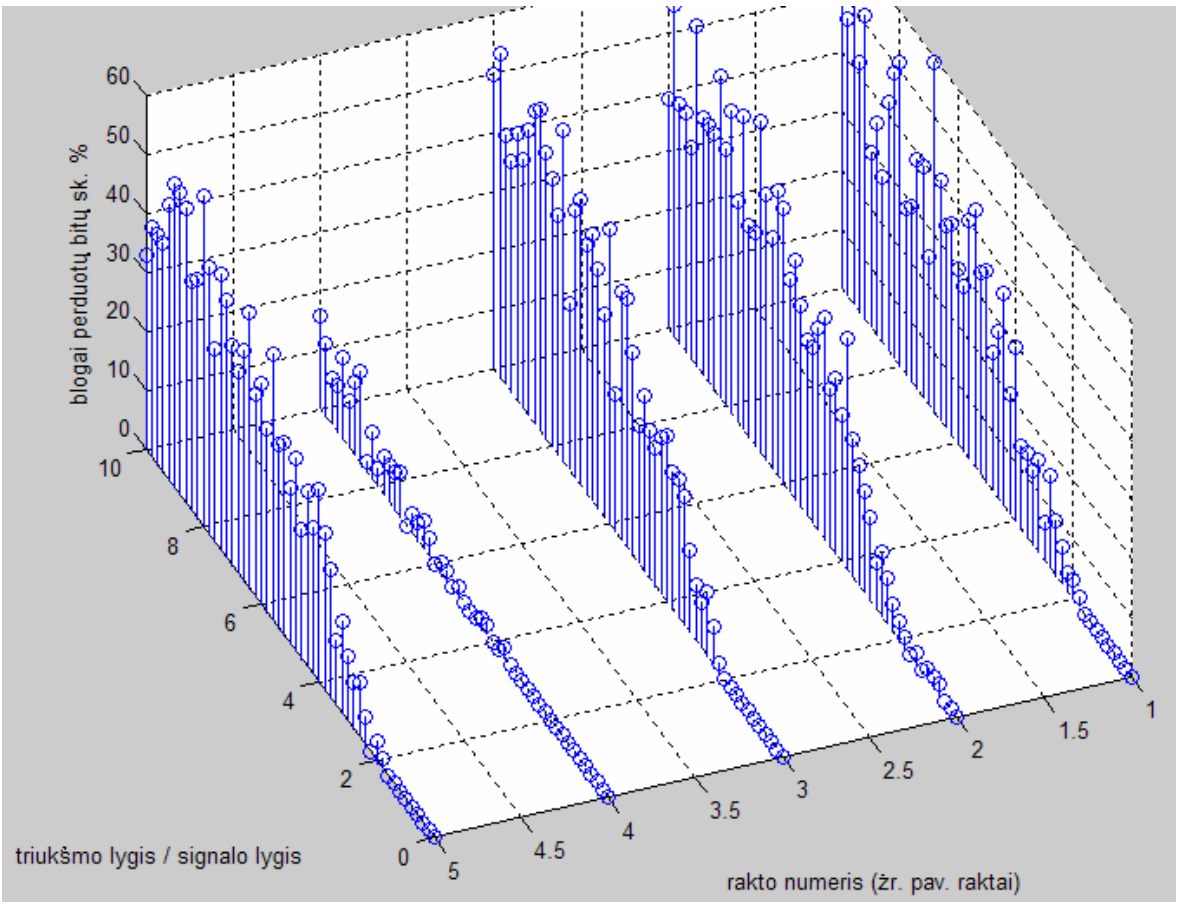
Imtuvo matematinis modeliavimas rodo, kad tokia duomenų perdavimo technologija, bent idealiu atveju yra veiksminga.

3.5 Sistemos atsparumo triukšmams charakteristikos

BER (*Bit Error Rate*) yra dažniausiai duomenų perdavimo sistemose naudojamas parametras, skirtas sistemos atsparumui triukšmams nusakyti. Šis parametras parodo tikimybe, kad bus teisingai perduotas siunčiamas bitas.

Šiuo atveju yra panaudotas paprastesnis metodas, kai yra parodomas santykinis klaidingai gautų bitų skaičius. Šio skaičiaus nulinė reikšmė atitinka vienetinę BER reikšmę, tai reiškia bitas bus tikrai perduotas teisingai. Tuo tarpu ~50% skaičius reiškia kad duomenų perdavimas jau nebeefektyvus, nes tokią pat charakteristiką parodytu ir dviejų atsitiktinių bitų sekų palyginimas.

Grafikas (12 pav.) parodo sistemos atsparumo triukšmams charakteristikas naudojant kiekvieną iš penkių skyriuje 3.2 (Siųstuvas) aprašytą raktą:



11 pav. Klaidų lygis naudojant skirtingus raktus

Čia kvantavimas 12 bitų (imtuvo) bei 4 bitų (siųstuvo), diskretizavimas - 64 kadrai vienam bitui, triukšmo lygis iki 1:10 (10dB).

Iš grafiko matyti, kad su keturiais iš penkių raktų, efektyvus duomenų perdavimas įmanomas triukšmo lygiui siekiant ne daugiau 1:2 (3dB). Tuo tarpu geriausias charakteristikas rodo ketvirtasis (eksponentinis) raktas.

3.6 Išvada

Matematinis modelis aprašytas pasitelkiant MatLab metamatinių skaičiavimų paketą. Sudarant modelį buvo sukurti šie failai:

Transmitter.m – šiame faile aprašytas siųstuvo veikimas.

Reciever.m – šiame faile aprašytas imtuvo veikimas.

Commchan.m – šiame faile realizuotas kanalas.

Getkey.m – šiame faile aprašytas rakto generatorius.

Chkey.m – šiame faile aprašyta chaosinės funkcijos rakto generatorius.

DTriger.m – šiame faile aprašytas detektorius.

GetBer.m – šiame faile aprašytas klaidingų bitų skaičiaus skaičiavimo algoritmas.

Testch.m – šiame faile aprašytas chaosinės funkcijos bifurkacinės kreivės gavimas.

TestChKey.m – šiame faile aprašytas vieno rakto testavimas.

Testbench.m -šiame failo aprašyti visi modemo matematinio modelio modeliavimo etapai.

Matematinio modelio testavimo rezultatai rodo, kad sistema su visais testuotais raktais gali patikimai perduoti duomenis, kai triukšmo lygis signalo lygi viršija ne daugiau kaip du kartus (3dB). Ši charakteristika galėtų būti pagerina pritaikant sudėtingesnį bito detektavimo algoritmą, ir/arba pritaikius kitus raktus.

Šia technologija pagrįsto modemo modelis gali būti panaudotas tolimesniems tyrimams kaip teorinis modifikuoto DCSK modemo modelis.

4 EKSPERIMENTINĖ DALIS

4.1 Aprašymas

Šio eksperimento tikslas yra ištirti dviejų DCSK modifikacijos modemo realizacijų efektyvumą. Palyginti tiesioginį vienlustės sistemos projektavimo būda ir vienluščio neuroprocesoriaus panaudojimą, juos kokybiškai įvertinant.

Atliekant eksperimentą, tiesiogiai projektuojant, sistemos modelis kuriamas pasitelkiant *SystemC* technologiją. Kompiliavimui ir modeliavimui naudojamas *gcc c++* kompiliatorius ir *SystemC 2.0.1* biblioteka. Banginių rezultatų vaizdavimui naudojama *GTKwave_win32* programa.

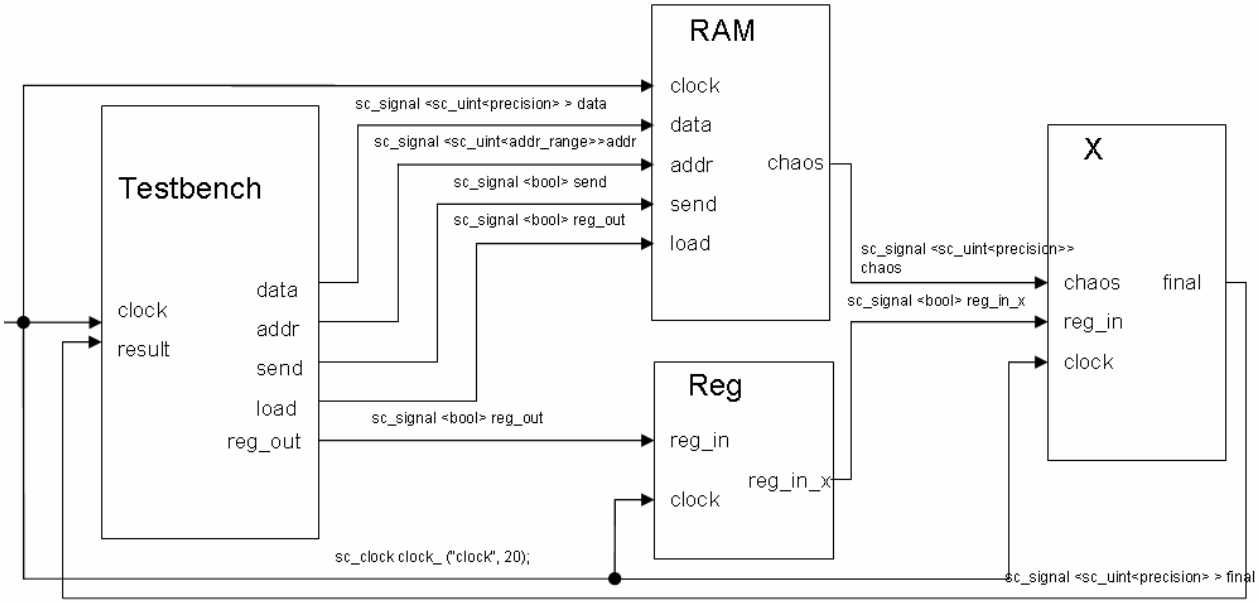
SystemC modelio sintezei naudojamas *Synopsys* sintezatorius. Sintezės rezultatų peržiūrai bei analizei atlikti naudojama *Synopsys design_analyzer* aplinka. Sintezuojant naudojama standartinė *synopsys Class* biblioteka.

Neuroninio tinklo modelio kūrimui panaudojamas MatLab paketas. Taip pat naudojamas *MS Office* paketo komponentas *Excel* reikalingas neuroprocesoriaus atminties formavimui iš parinktos teorinės neuroninio tinklo struktūros.

4.2 Siųstuvo modelis SystemC

4.2.1 Modelis

Siųstuvo schemos paprastumas matomas jau matematiniame modelyje. Tam, kad būtų realizuotas SystemC, jam reikalingi inverteris (*X*), registras (*Reg*) ir atmintis (*RAM*). Atmintis šiuo atveju yra skirta saugoti rakto kadrų sekai. Registras siunčiant duomenis saugo siunčiamo bito reikšmę. Inverteris priklausomai nuo *Reg* saugomos reikšmės, invertuoja iš *RAM* ateinančią kadro reikšmę arba ne.



12 pav. Siųstuvo blokinė diagrama

Siųstuvo darbas turi dvi fazes: rakto užkrovimą į atmintį ir normalų darbo režimą.

Testbench – koordinuoja siųstuvo darbą. Rakto užkrovimo fazėje iš failo skaito raktą, siunčia jį ir adresą į atmintį, nustato signalus *send* ir *load* atitinkamai į **false** ir **true**. Duomenų siuntimo fazėje skaito siunčiamus duomenis iš failo ir siunčia juos tarpiniam moduliui REG. Taip pat renka duomenis ateinančius iš siųstuvo ir rašo juos į tekstinį failą. Sinchronizuoja į modulius RAM ir Reg paduodamus duomenis.

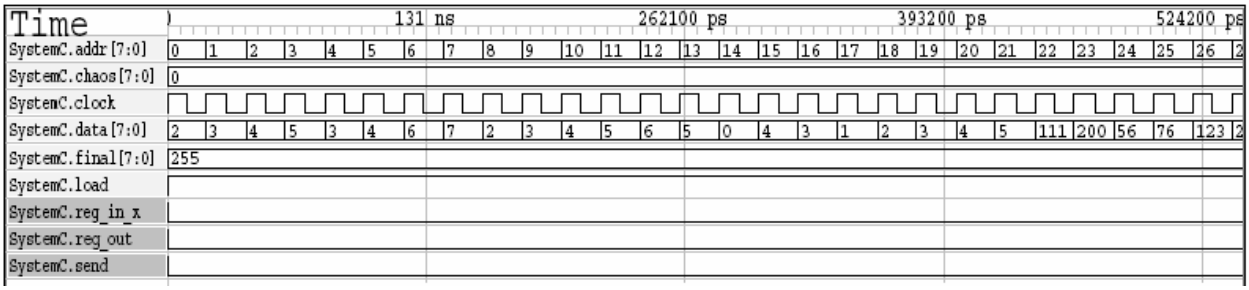
RAM – Turi du darbo režimus. Skaitymo ir siuntimo. Kai *send* = **false** ir *load* = **true**, modulis įrašinėja į savo atmintį į portą data gautus duomenis adresu, kurį gauna į portą *addr*. Kai *send* = **true** ir *load* = **false**, atmintis persijungia į siuntimo režimą. Kiekviename cikle siunčia į portą *chaos* duomenis, saugomus celėje adresu iš *addr* prievado.

Multiplyer – priklausomai nuo ateinančio signalo iš registro invertuoja arba ne iš RAM atėjusį raktą ir siunčia jį testinei aplinkai Testbench.

REG – tarpinis atminties elementas duomenų perdavimui į daugintuvą. Jis taip pat atlieka vėlinimo funkciją tam, kad būtų kompensuotas laikas, kurį užtrunka rakto skaitymas iš RAM.

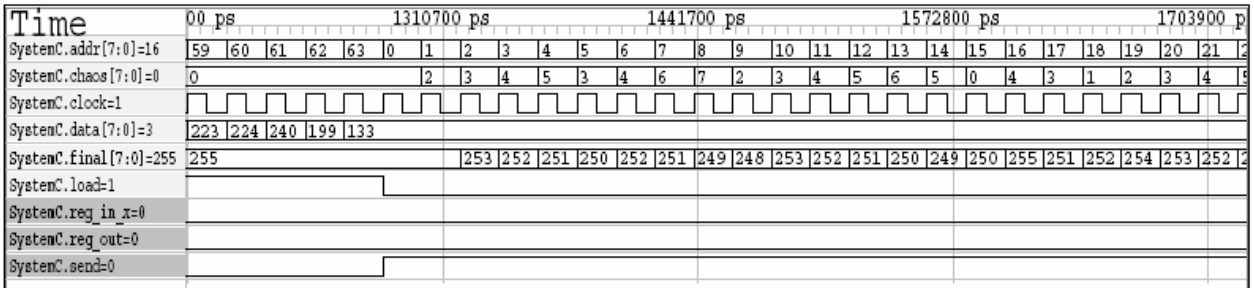
4.2.2 Modeliavimo rezultatai

Testuojant siųstuvo SystemC modelį gauti rezultatai rodo kad jis atitinka jam numatytą funkcionalumą. Realizuotos numatytos rakto užkrovimo bei signalo siuntimo galimybės



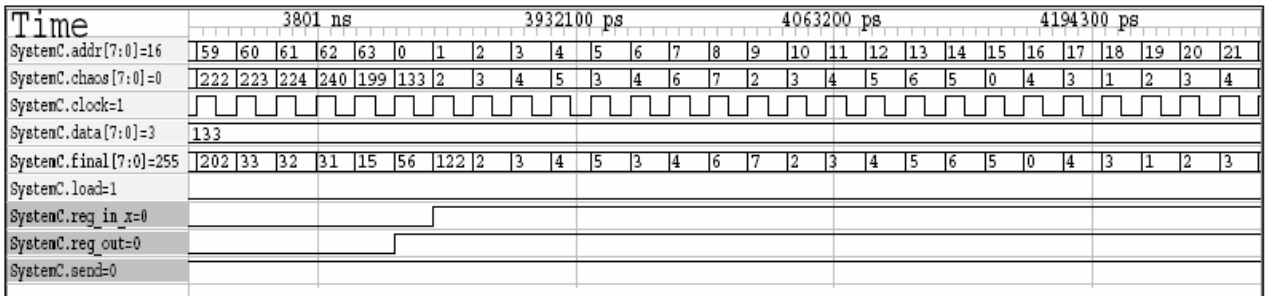
13 pav. Siųstuvo modeliavimo rezultatai

Čia (13 pav.), modeliavimo pradžioje vyksta rakto užkrovimas iš testinės aplinkos į atmintį *RAM*.



14 pav. Siųstuvo modeliavimo rezultatai

Čia (14 pav.) baigiamas rakto užkrovimas ir pradedamas užkrauto rakto siuntimas į kodavimo įrenginį. Signalas **load** perjungiamas į loginį nulį, o signalas **send** į vienetą. Bito užkodavimas trunka du sinchronizacijos signalo periodus.



15 pav. Siųstuvo modeliavimo rezultatai

Čia (15 pav.) pavaizduotas eilinis duomenų siuntimas. Į siųstuvo įėjimą **reg_out** siunčiamas koduotinas bitas. Po dviejų taktų išėjime **final** pasirodo pirmasis užkoduoto bito kadras. Po 64 taktų užkoduotas bitas baigiamas siųsti ir pradedamas siųsti kitas.

4.2.3 Sintezė

Siųstuvo SystemC modelis sintezuojamas pasitelkiant *Synopsys* sintezatorių. Sintezės analizei gauti panaudojamas *Sysnopsys* įrankis *design_analyzer.*

Susintezavus ir atlikus analizę gauti rezultatai pavaizduoti 4 lentelėje.

1 lentelė. Siųstuvo komponentų sintezės ataskaita

	RAM	Reg	X
Laikas [fs]	0.82	0.82	0.82
Galia [uW]	957.2969	918.5561	20.4930
Plotas	13456.126953	14.091249	189.750000

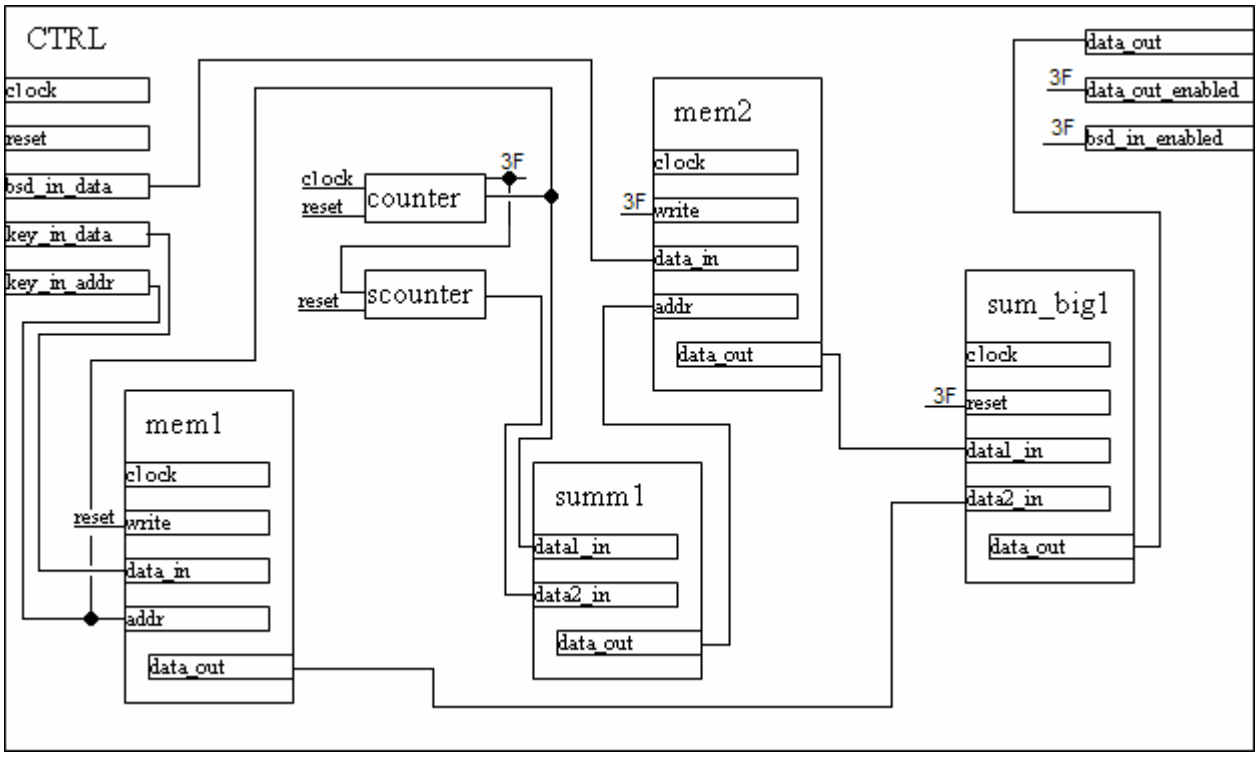
Čia laikas, kurio reikia signalui nueiti tolimiausią galimą atstumą schemoje.

Kadangi *RAM* atminties apimtis, vėlinimai ir kt. labai priklauso nuo panaudotos technologijos, taip pat todėl, kad papildomai imtuvo realizacijai panaudoto neuroprocesoriaus atminties parametrai nėra įtraukti į rezultatus, šiuo atveju siųstuvo medelio, *RAM* charakteristikos tyrimui naudojamos nebus. Tai galutiniam rezultatui įtakos neturės, nes abiem atvejais reikalingos atminties kiekis yra labai artimas – apie 200 baitų (jei naudojama rakto ir koreliacijos trukmė kadrais - 64), nes bendras reikalingos atminties kiekis yra siųstuvo rakto buferis plius imtuvo rakto buferis plius imtuvo įėjimo buferis (64+64+64).

4.3 Imtuvo modelis SystemC

4.3.1 Modelis

Imtuvo modelis realizuojamas pasitelkiant tuos pačius įrankius kurie buvo naudojamo kuriant siųstuvo modelį. Čia raktui saugoti taip pat naudojama atmintis, tačiau dar vienas tokio paties dydžio atminties blokas reikalingas kaip įėjimo buferis (16 pav.).



16 pav. Imtuvo blokinė diagrama

Čia atmintis *mem1* yra skirta saugoti rakto kadrų sekai. Raktas užkraunamas darbo pradžioje, rakto užkrovimo metu **reset** laikomas aukštas, taigi ir *mem1* **write.** Tiksiklis bendras visai schemai ir keičia adreso skaitliuka *counter*. Kai *counter* reikšmė pasiekia rakto ilgio reikšmę, generuojamas signalas **3F.**

Imtuvo veikimą galima apibūdinti kai pasikartojančių ciklų seką. Kiekvieno ciklo metu atmintyje *mem1* įrašytas raktas koreliuojamas su įėjimo buferio atmintyje *mem2* esančiais duomenimis. Skaitliukas *counter* kiekvieną ciklą pradeda nuo nulio, o jo reikšmė naudojama kaip adresas *mem1*. Atminties *mem2* adresas skaičiuojamas sumuojant skaitliuko *counter* ir adreso postūnio skaitliuko *scounter* reikšmes. Šis sumavimas atliekamas sumatoriuje *summ1*.

Kiekvieno koreliacijos ciklo pabaigoje generuojamas signalas 3F, jis siunčiamas į išėjimus nurodančius kad imtuvas jau apskaičiavo naują koreliacijos reikšmę, bei kad yra pasirengęs priimti naują įeinančio signalo reikšmę įėjime. Pagrindinis sumatorius *sum_big1*, atliekantis reikšmių daugybą ir sumavimą nustatomas į nulį. Vienetu padidinama adreso postūmio skaitliuko reikšmė.

4.3.2 Modeliavimo rezultatai

Imtuvo modeliavimui naudojami rezultatai gauti siųstuvo modeliavimo metu, ir atitinkantys matematinio modelio modeliavimo rezultatus.

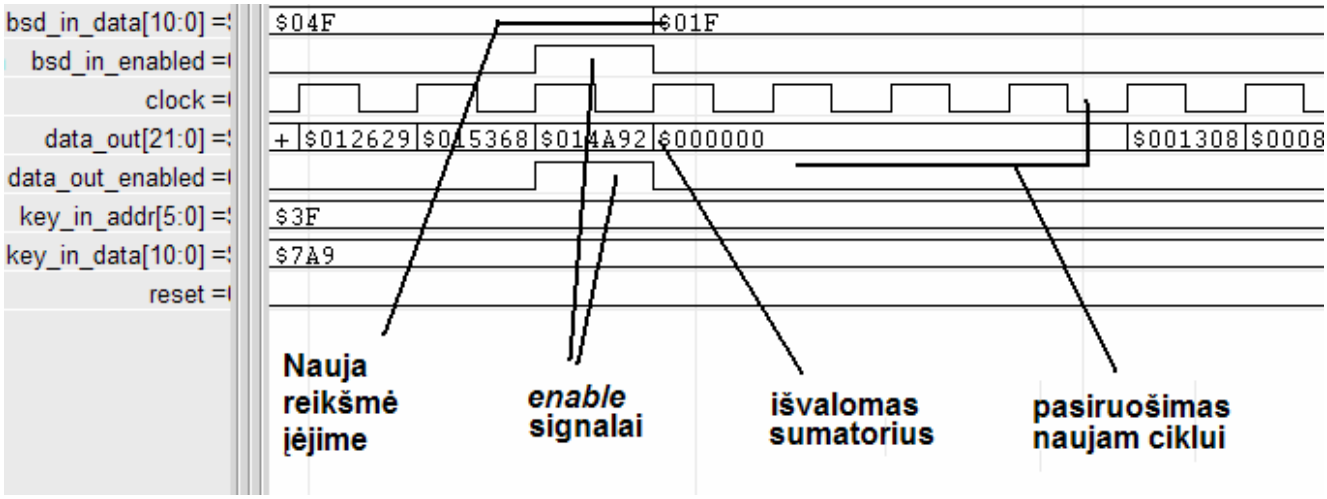
Modeliavimo metu gautiem rezultatam peržiūrėti naudojamos programos GTKWave bei MatLab.

Sistemos signalų laikinė diagrama:

17 pav. Imtuvo modeliavimas – signalai

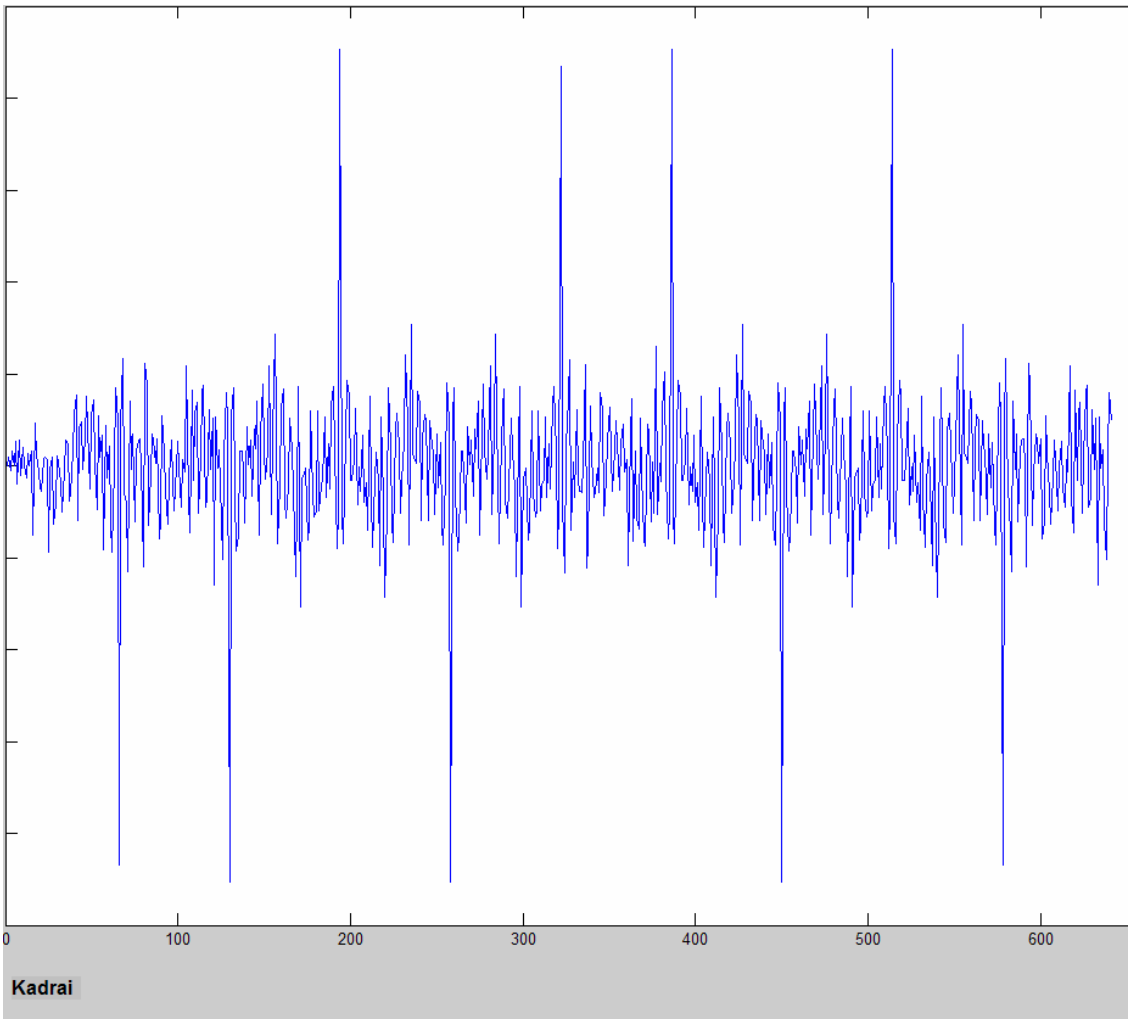
Pradzioje 64 (rakto ilgis) taktus laikomas aukštas **reset** signalas. Tuo metu sistema neveikia, o iš išorės vykdomas rakto užkrovimas. Dar po vieno rakto ilgio pasirodo pirmoji apskaičiuota koreliacijos reikšmė. Nustatomas **data_out_enabled** signalas. Taip pat nustatomas **bsd_in_enabled**, reiškiantis kad įėjime turėtų būti paduota sekanti įeinančio signalo reikšmė.

Sistemos signalų laikinė diagrama, ciklo pasikeitimas:



18 pav. Imtuvo modeliavimas – signalai

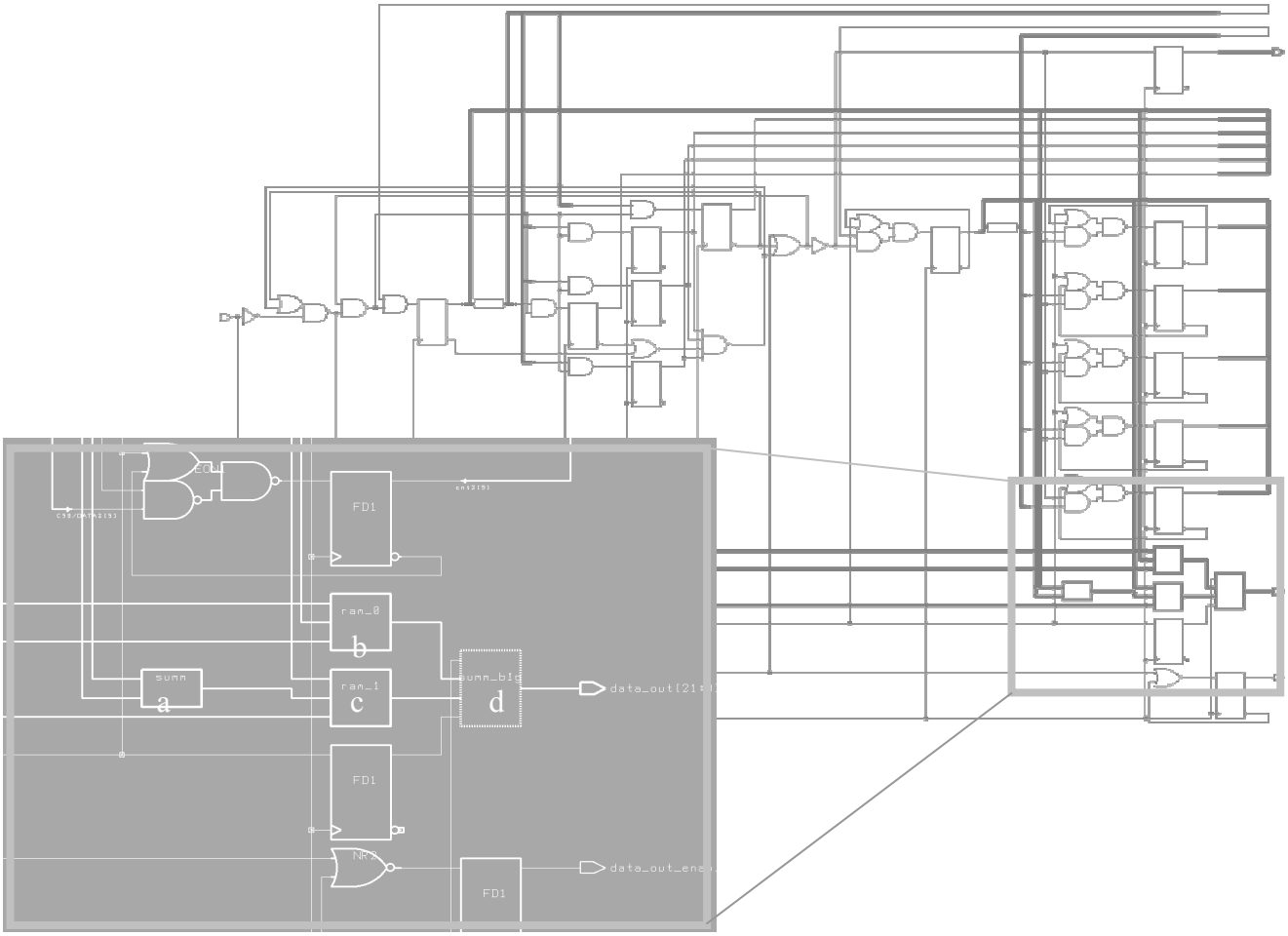
Paskutinių reikšmių prieš išvalant sumatorių seka sudaro imtuvo išėjimo reikšmių seką. Ši seka tai įeinančio signalo ir rakto koreliacinė kreivė:



19 pav. Imtuvo modeliavimo rezultatai

4.3.3 Sintezės rezultatai

Imtuvo modelis sintezuojamas pasitelkus *Synopsys* sintezatorių bei *design_analyzer* aplinką.



20 pav. Imtuvo kontrolinio bloko struktūrinė schema

Imtuvo kontrolinio bloko schemoje raidėmis a, b, c ir d atitinkamai pažymėti vidiniai komponentai – sumatorius, atminties blokai bei koreliacijos sumatorius.

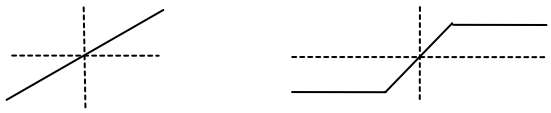
2 lentelė. Imtuvo SystemC modelio sintezės reaultatai

Ilgiausio tako laikas [fs]	1.41
Plotas (neskaitant atminties)	2192
Atminties plotas (mem1 ir mem2)	10276
Bendras plotas	12468
Galia (be atminties) [uW]	525.0

4.4 Imtuvo neuroniniame tinkle modelis

4.4.1 Neuroninio tinklo perdavimo funkcijos

Sukurtas neuroninis procesorius turi dvi perdavimo funkcijas tiesinę, bei tiesinę su apribojimu.



21 pav. Neuroninio tinklo perdavimo funkcijos

Kairėje tiesinė, dešinėje tiesinė su apribojimais

Neuroninio tinklo perdavimo funkcijų formulės neuroniniame procesoriuje:

$$F(X)=\frac{\sum_{i=1}^{N}\frac{X_i*W_i}{K}}{M_1} \tag{11}$$

Neurono su tiesine perdavimo funkcija formulė

$$F(X)=\begin{cases} M_1 & ,kai \quad \sum_{i=1}^{N}\frac{X_i*W_i}{K}>=M_1 \\ M_2 & ,kai \quad \sum_{i=1}^{N}\frac{X_i*W_i}{K}<M_2 \\ \sum_{i=1}^{N}\frac{X_i*\overline{X_i}}{K} & ,kai \quad \sum_{i=1}^{N}\frac{X_i*W_i}{K}<M_1 \quad ir \quad \sum_{i=1}^{N}\frac{X_i*W_i}{K}>M_2 \end{cases} \tag{12}$$

Neurono funkcijos formulė kai perdavimo funkcija tiesė su apribojimais

Funkcijose panaudoti kintamieji:

W_i -Neurono svoris

M_1-Didžiausia funkcijos reikšmė

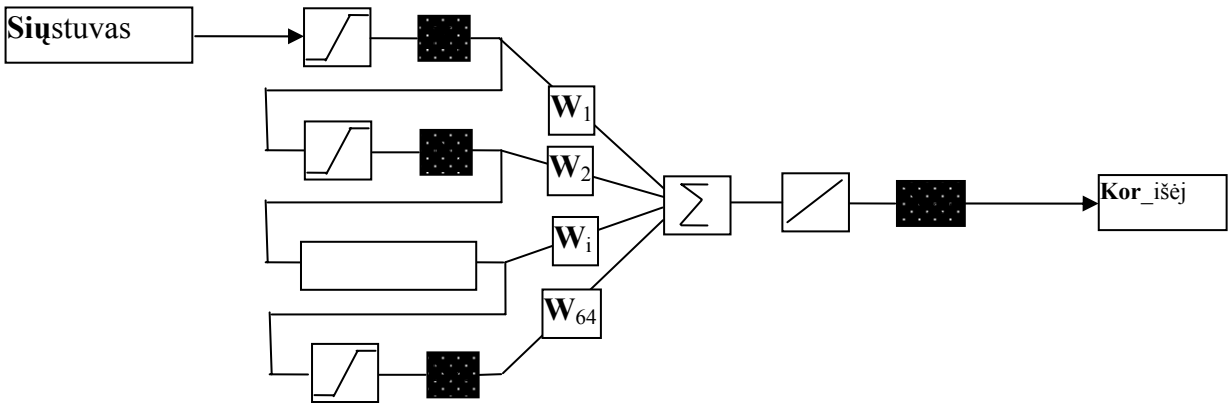
M_1-Mažiausia funkcijos reikšmė

K_1-Koeficientas norint sumažinti sumos ploti

N-Neurono įėjimų skaičius

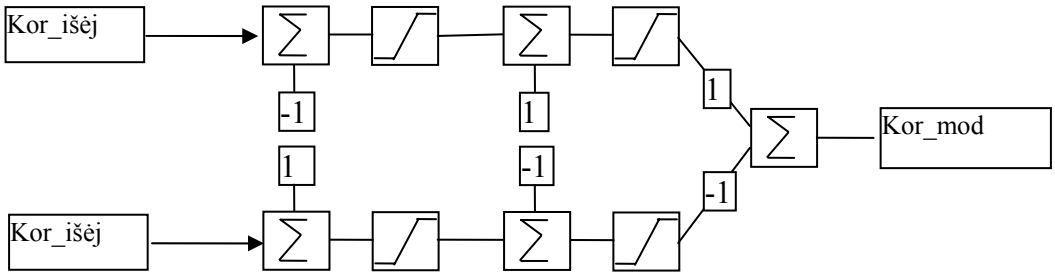
4.4.2 Neuroninio tinklo struktūra

Koreliacija neuroniniame tinkle skaičiuojama užvėlinus įėjimo reikšmes bei kiekvieną jų padauginus iš rakto reikšmių, kai rakto reikšmės yra vietoj N_{kor} neurono svorių.



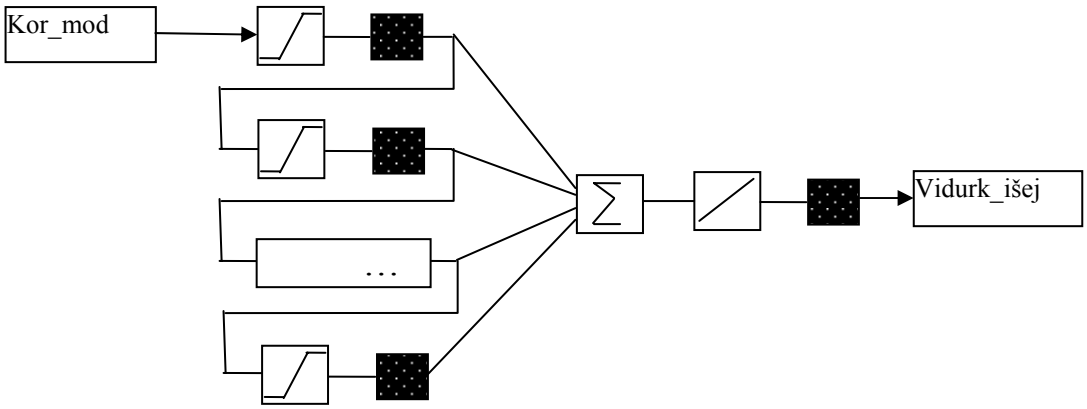
22 pav. Koreliaciją skaičiuojantis neuroninis tinklas

Tikslesniam bito išskaičiavimui iš koreliacijos reikalingas papildomas neuroninis tinklas apskaičiuojantis koreliacijos modulio paskutinių reikšmių vidurkį.



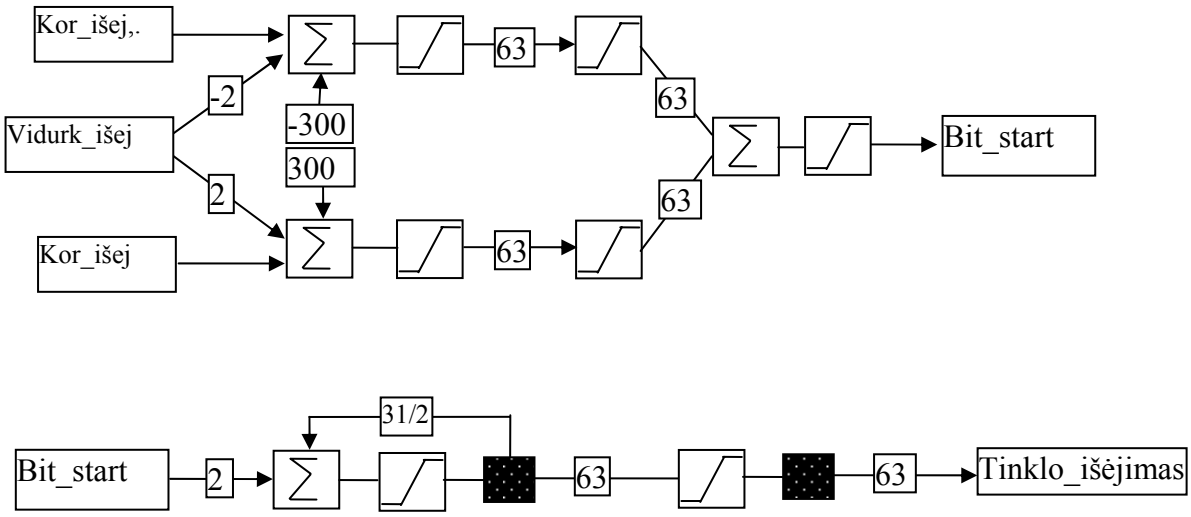
23 pav. Neuroninis tinklas skaičiuojantis modulį

Vidurkis skaičiuojamas užvėlinus koreliacijos modulio reikšmes. Konkrečiame uždavinyje naudojamas 64 reikšmių vėlinimas



24 pav. Vidurkio skaičiavimas

Bito išskaičiavimui užtenka 2 sluoksnių neuroninio tinklo su 3 neuronais tačiau neuroniniui procesoriui ribojant maksimalų svorio dydį tenka tinklo dydį padidinti iki 3 sluoksnių bei 5 neuronų dydžio.

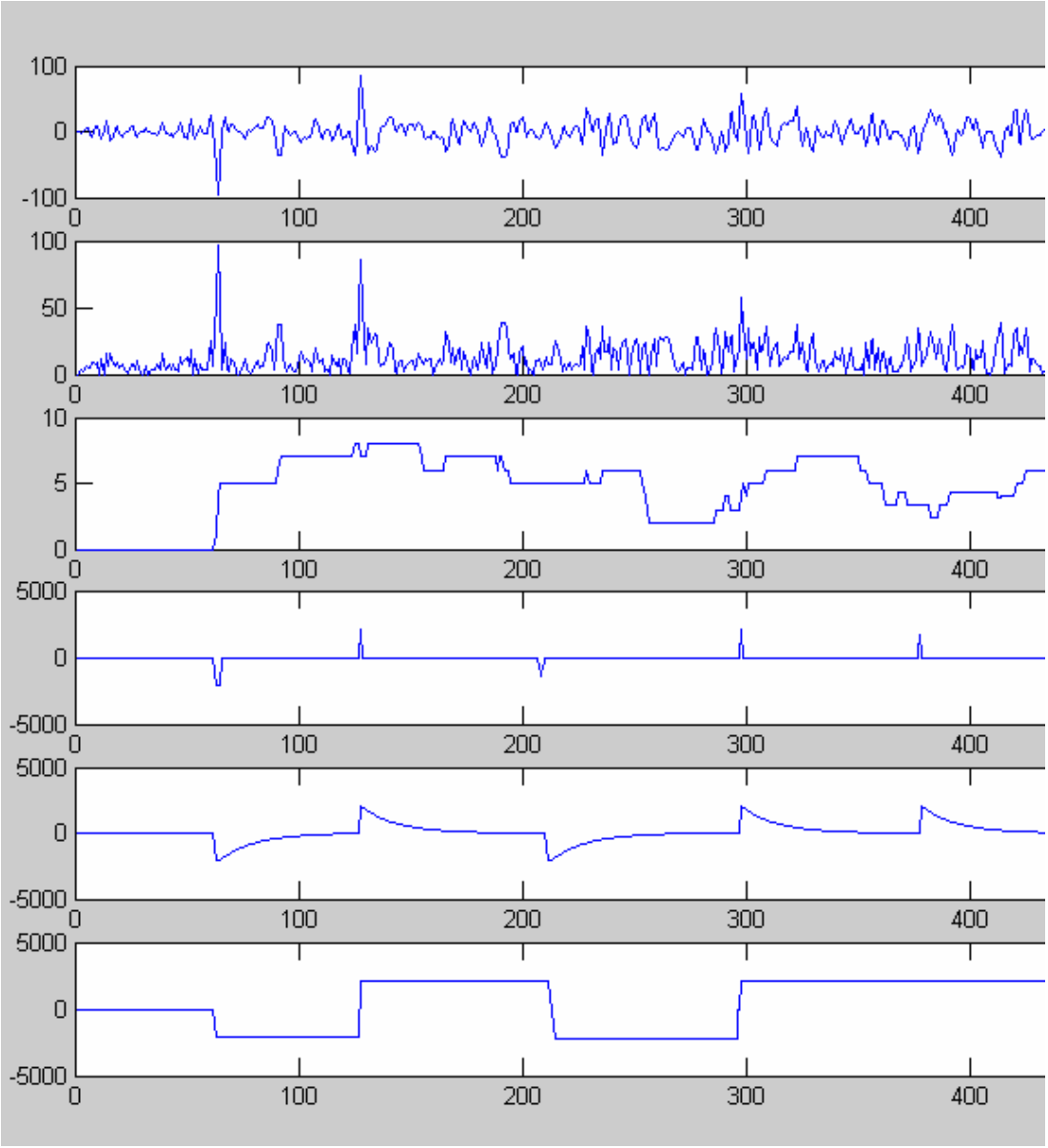


25 pav. Bito išskaičiavimui skirtas neuroninis tinklas

Viršuje fiksuojama bito pradžia; apačioje formuojamas stačiakampis bito signalas.

Neuroninio tinklo veikimo principas demonstruojamas sekančiame paveiksle.

Skaičiuojant koreliacijos vidurkį pasiekta, kad bito lygio frontas dinamiškai keistųsi priklausomai nuo koreliacijos triukšmingumo. Tokių būdu gaunamas pakankamai tiksliai išskiriamas bitas.



26 pav. Neuroninio tinklo dalinių taškų išėjimo kreivės, apdorojant triukšmingą signalą

1-triukšminga koreliacija; 2-koreliacijos modulis; 3-vidurkis; 4-užfiksuoti bitai; 5,6-formuojama stačiakampę bito forma.

4.4.3 Neuronų procesoriaus sintezės rezultatai

Neuronų procesoriaus sintezės rezultatai yra gauti iš procesoriaus kūrėjo (žr. pirmą priedą).

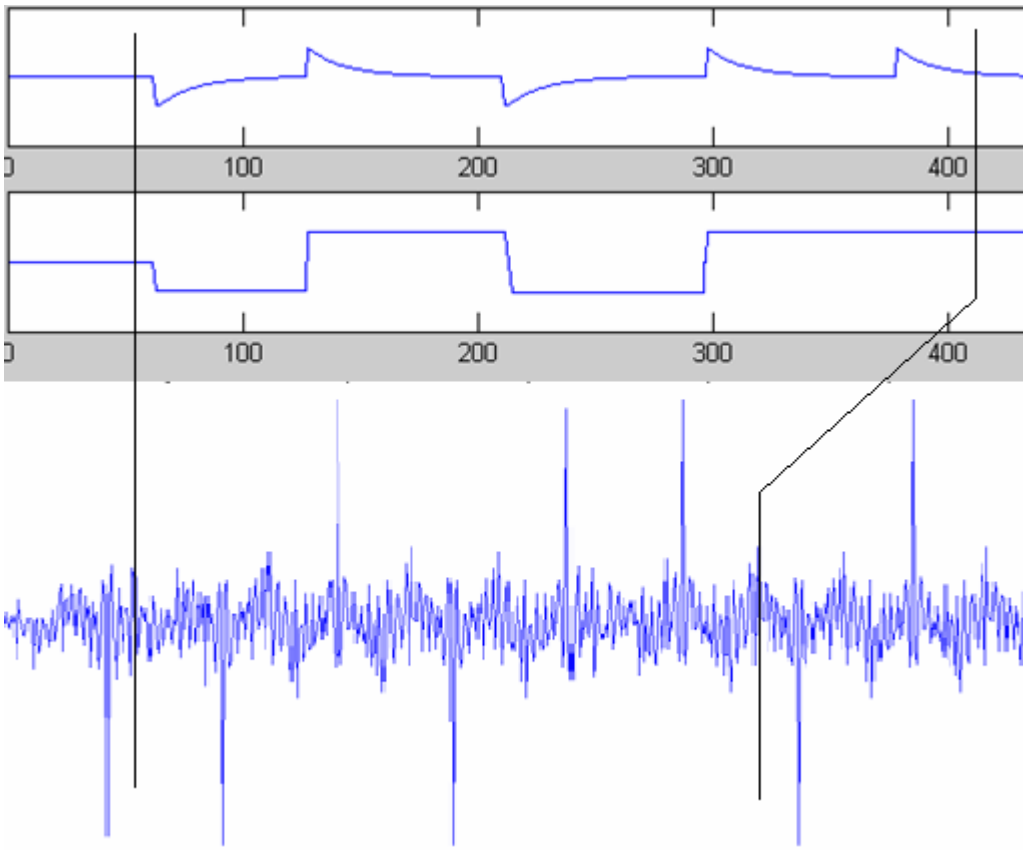
3 lentelė. Neuronų procesoriaus sintezės rezultatai

Ilgiausio tako laikas [fs]	9.20
Bendras plotas	6182
Galia [uW]	206.7745

4.5 Eksperimento rezultatai

4.5.1 Modeliavimo rezultatų palyginimas

Modeliavimo rezultatų sutapimas yra būtinas tam, kad eksperimento rezultatai būtų prasmingi. Rezultatų sutapimas šiuo atveju reiškia kad abu imtuvo modeliai turi atkoduoti vienodą bitų seką iš tų pačių pradinių duomenų.



27 pav. Imtuvo modelių rezultatų palyginimas

26 paveiksle matyti, kad gauta tapati loginė bitų seka.

Išvada: modelių funkcionalumas sutampa, todėl jie palyginimui tinkami.

4.5.2 Sintezės rezultatų palyginimas

4 lentelė. Eksperimento sintezės rezultatai

	SystemC	Neuroprocesorius
Ilgiausio tako laikas [fs]	1.41	9.20
Bendras plotas	2395.841249	6182
Galia [uW]	1464.0491	206.7745

Neuronų procesoriaus modelyje nebuvo realizuotas imtuvas, tačiau tai įtakos rezultatam neturi, nes jo išnaudojami resursai čia pateiktų parametrų atžvilgiu yra vienodi.

Atsižvelgiant į gautus rezultatus negalima vienareikšmiškai nustatyti vieno kurio nors modelio pranašumo, tačiau išryškėja jų privalumai bei trūkumai.

Greitaveika. Abiem atvejai vienai reikšmei užkoduoti reikalingas beveik taspats tiksiklio taktų skaičius, kai tuo tarpu iš ketvirtos lentelės matyti, kad ilgiausias signalo kelias neuroprocesoriaus modelio atveju trunka net apie šešis kartus ilgiau, o tai gali turėti įtakos schemos sinchronizacijai, maksimaliam galimam dažniui, taigi ir greitaveikai.

Plotas ir galia. Rezultatai rodo, kad SystemC atveju plotas yra apie du su puse karto mažesnis. Abiejų modelių duomenys čia pateikti neįskaičiuojant atminties. Tuo tarpu galia apie septynis kartus mažesnė neuroprocesoriaus atveju.

5 IŠVADOS

Tam kad būtų patikrintas DCSK technologijos modifikacijos veiksmingumas, bei ištirti du jos įgyvendinimo būdai, buvo sukurtas sistemos matematinis modelis, bei suprojektuoti ir realizuoti du jos modeliai. Atsižvelgiant į eksperimento rezultatus galima pateikti šias išvadas:

- Modifikuota DCSK technologija, kai bitas siunčiamas be atraminės dalies, o vietoje jos imtuvas ir siųstuvas naudoją tik vienai porai žinomą raktą, yra funkcionali.

- Naudojant net paprastus, periodiniu signalu pagrįstus raktus, sistema užtikrina duomenų perdavimą esant 3dB triukšmui.

- Technologija gali būti įgyvendinta tiesiogiai kaip vienlustė sistema, pvz. panaudojant *SystemC* aparatūros aprašymo kalbą.

- Technologija gali būti įgyvendinta neuroniniu tinklu, pvz. panaudojant neuroprocesorių.

- Tirta *SystemC* realizacija yra daigiau nei du kartus efektyvesnė sunaudojamų aparatūrinių resursų požiūriu negu tirta realizacija panaudojant neuroprocesorių.

- Tirtų *SystemC* bei neuroprocesoriaus realizacijų greitaveikos skirtumas yra minimalus.

- Projektavimo laiko atžvilgiu neuroprocesorius yra efektyvesnis, nes jo atveju nuo matematinio modeliavimo prie realios sistemos pereinama iškart, be tarpinių programavimo ir modeliavimo etapų.

6 LITERATŪRA

[1] Aalborg univercity. Chaos in communication.
Prieiga per internetą: < http://kom.aau.dk/~egka/lect1/slides/Theory3.pdf >

[2] Aalborg univercity. Description of DCSK system.
Prieiga per internetą: < http://kom.aau.dk/~egka/lect1/index.htm#Description >

[3] IEICE transactions, fundamentals. 1998 rugsėjis.
FM-DCSK: A robust modulation scheme for chaotic communications.
Prieiga per internetą: < http://kom.aau.dk/~egka/lect1/material/e81-a_9_1798.pdf >

[4] IEEE transactions on circuits and systems - I: Fundamental theory and applications. 2001.
Prieiga per internetą: < http://kom.aau.dk/~egka/lect1/material/ieee_2001_b.pdf >

[5] IEEE transactions on circuits and systems - I: Fundamental theory and applications. 2001.
Application of symbolic dynamics to differential chaos shift keying.
Prieiga per internetą: < http://kom.aau.dk/~egka/lect1/material/tcas1_2002.pdf >

[6] IEEE transactions on circuits and systems.
CMOS implementation of a chaos-based communication system.
Prieiga per internetą: < http://kom.aau.dk/~egka/lect1/material/paper2.pdf >

[7] Bob Paddock. Noise / Chaos / Random Numbers and Linear Feedback Shift Registers. Circuit cellar online.
Prieiga per internetą: < http://www.csonline.net/bpaddock/cco/noise/c89r4.htm >

7 TERMINŲ IR SANTRUMPŲ ŽODYNAS

DCSK – direct chaos shift keying

CSK – chaos shift keying

PSK – phase shift keying

FM-DCSK – frequency modulated DCSK

SystemC – viena iš aparatūros aprašymo kalbų

Raktas – modemo naudojama kadrų seka

Neuroprocesorius – procesorius skirtas neuroniniam tinklui apdoroti

BER – bit error rate

8 PRIEDAS 1. Matematinio modelio išeities tekstai

Transmitter.m – siųstuvo veikimas

```
function x_tx = Transmitter( x_key, x_bits )
% x_key  - buffer holding the key sequence
% x_bits - array holding logical bits
x_keyi = - x_key;
s_klen = length( x_key ); % length of the key buffer in samples
s_blen = length( x_bits ); % length of the bit buffer
x_tx = zeros( s_klen * s_blen, 1 ); %initialize signal buffer
for i = 0 : s_blen-1
  if x_bits(i+1) == 1
    x_tx( (i*s_klen)+1:(i*s_klen)+s_klen ) = x_key;
  else
    x_tx( (i*s_klen)+1:(i*s_klen)+s_klen ) = x_keyi;
  end;
end;
```

Reciever.m – imtuvo veikimas

```
function x_rx = Receiver( x_txn, x_key, prec )
% x_tx  - received signal
% x_key - key buffer
% prec  - numer of bits per sample
% length of the signal in samples
s_slen = length( x_txn );
%calculate amplitude
ampl_n = pow2( prec );
ampl_o = max(x_txn)-min(x_txn);
%scale
x_txn = x_txn * ( ampl_n / ampl_o );
%quantize
x_txn = round( x_txn );
% length of the key buffer in samples
s_klen = length( x_key );
%initialize output buffer
x_rx = zeros( s_slen, 1 );
% calculate amplitude
amp = pow2( prec );
% correlation with the key
for i = 0 : s_slen-1
  c = 0;
  for j = 1 : s_klen
    if i + j <= s_slen
      c = c + ( ( x_txn(i+j) * x_key(j) ) / amp );
    end;
  end;
  x_rx(i+1) = c;
end;
```

Commchan.m – kanalas

```
function x_txn = CommChan( x_tx, nl )


s_slen = length( x_tx );

x_txn = zeros( s_slen, 1 );

am = max(x_tx) - min(x_tx);

for i = 1 : s_slen
  x_txn(i) = x_tx(i) + ( (rand - 0.5) * am * nl );
end;
```

Getkey.m – rakto generatorius

```
function K = GetKey( s_len, prec, kn )
% s_len  - length of the key in samples
% b_prec - nuber of bits per sample

K = zeros( s_len, 1 );

t = 0:pi/(s_len-1):pi;

if kn == 1
  for i = 1 : s_len
    K(i) = sin(t(i)*(i/4));
  end;
else if kn == 2
  %for i = 1 : s_len
  %  K(i) = sin(t(i)*(i/2))*sin(t(i));
  %end;
  K = ChKey( 64, 4, 1000, 0.5 );
else if kn == 3
  for i = 1 : s_len
    K(i) = sin( t(i) * (i/16) );
  end;
else if kn == 4
  for i = 1 : s_len
    K(i) = sin(t(i)*(1/i));
  end;
else if kn == 5
  for i = 1 : s_len
    K(i) = sin( t(i)*8 );
  end;
end;end;end;end;end;

%calculate amplitude
ampl_n = pow2( prec );
ampl_o = max(K)-min(K);

%scale
K = K * ( ampl_n / ampl_o );

%quantize
K = round( K );
```

Chkey.m – chaosinės funkcijos rakto generatorius

```
function x_key = ChKey( s_len, prec, start, init )
% s_len  - length of the key in samples
% b_prec - nuber of bits per sample
% start  - number of iterations to start from
% init   - initial seed for the key

x_key = zeros( s_len, 1 );

x = init;
for i = 1 : 1000;
  x = 3.99 * x * ( 1 - x );
end;
for i = 1 : s_len;
  x_key(i) = 3.99 * x * ( 1 - x );
  x = x_key(i);
end;

%shift to the center
x_key = x_key - 0.5;

%calculate amplitude
ampl = pow2( prec );

%scale
x_key = x_key * ampl;

%quantize
x_key = round( x_key );
```

DTriger.m – detektorius

```
function x_trg = DTriger( x_rx, lev )
% x_rx - received signal
% lev  - cut level

s_slen = length( x_rx ); % signal buffer length in samples

x_trg = zeros( s_slen, 1 );

lvalue = 0;
for i = 1 : s_slen
  if x_rx(i) > lev
    lvalue = 1;
  end;
  if x_rx(i) < -lev
    lvalue = 0;
  end;
  x_trg(i) = lvalue;
end;
```

GetBer.m – klaidingų bitų skaičiaus skaičiavimo algoritmas

function err = GetBER(x_trg, x_bits, s_T, s_dT)

```
s_slen = length( x_trg );
s_blen = length( x_bits );

err = 0;

for i = 1 : s_blen
  j = (s_T * (i - 1)) + s_dT;
  if (j<=s_slen)&&(i<=s_blen)
    if x_trg(j) ~= x_bits(i)
      err = err + 1;
    end;
  end;
end;

err = err / s_blen * 100;
```

Testch.m – chaosinės funkcijos bifurkacinės kreivės gavimas

```
close all;
clear all;

p = 3.5:0.0001:4;
f = zeros( length( p ), 100 );

for i = 1 : length( p )
  x = 0.83;
  for j = 1 : 1000;
    x = p(i) * x * ( 1 - x );
  end;
  for j = 1 : 100;
    f(i,j) = p(i) * x * ( 1 - x );
    x      = f(i,j);
  end;
end;

fp = zeros(100,1);

hold on;
figure( 1 );
for i = 1 : 100
  subplot( 1, 1, 1 );
  %stem( p.', f(1:length(p),i), 'fill' );
  %scatter( p.', f(1:length(p),i), 1, '.' );
  plot( p(1:length(p)).', f(1:length(p),i), 'k.', 'MarkerSize', 1 );
end;
xlabel( 'k' );
ylabel( 'y[x0..xn,k]' )
title( 'su kiekvienu k: y[x] = k * y[x-1] * ( 1 - y[x-1] )' );

%subplot( 1, 1, 1 ), plot( x, zeros(1:length(x),1) );
%subplot( 1, 1, 1 ), plot( x, ones(1:length(x),1) );
```

TestChKey.m – vieno rakto testavimas

```
close all;
clear all;
x_key  = ChKey( 64, 4, 1000, 0.5 );
figure(1);
subplot(1,1,1), plot( x_key );
```

Testbench.m – visi modemo matematinio modelio modeliavimo etapai

```
%close all;
clear all;

s_klen = 64; % length of the key in samples
prectr = 4;  % number of bits per sample in the transmitter
precrc = 12;  % number of bits per sample in the receiver
bitcnt = 15; % number of bits to simulate
lev    = 50; % teshhold level for triger in recog circuit
%nlev   = 0; % noise level

%generate bit sequence
x_bits = round( rand( bitcnt, 1 ) );

nlev   = 0 : 0.2 : 0.4;
x_err  = zeros( 5, length(nlev) );

figure( 1 );
for i = 1 : 5
  x_key  = GetKey( s_klen, prectr, i  );
  subplot( 5, 1, i ), plot( x_key );
  for j = 1 : length(nlev)
    x_tx  = Transmitter( x_key, x_bits );
    x_txn = CommChan( x_tx, nlev(j) );
    x_rx  = Receiver( x_txn, x_key, precrc );
    x_trg = DTriger( x_rx, lev );
    x_err(i,j) = GetBER( x_trg, x_bits, s_klen, 2);
    [i,j]
  end;
end;

figure( 2 );
subplot( 1, 1, 1 ), stem3( nlev, 1:5, x_err );
ylabel( 'rakto numeris (zr. pav. raktai)' );
zlabel( 'blogai perduotu bitu sk. %' );
xlabel( 'triuksmo lygis / signalo lygis' );
camorbit(-75,20);

%figure( 2 );
%subplot( 1, 1, 1 );
%hold on;
%for i = 1 : 5
   % plot( x_err(i,:), 'color', [(1/i) (1-1/i) (.5/i)] );
    %end;
%ylabel( 'bad bits transmitten %' );
%xlabel( 'noise level / signal level' );

%figure( 3 );
%subplot( 1, 1, 1 ), plot( x_key );
plotcnt = 4;
subplot( plotcnt, 1, 1 ), stem( x_bits );
subplot( plotcnt, 1, 2 ), plot( x_tx );
subplot( plotcnt, 1, 3 ), plot( x_rx );
subplot( plotcnt, 1, 4 ), plot( x_trg );
```

9 KITI PRIEDAI

Priedai pateikiami kompaktiniame diske:

- SystemC išeities tekstai modeliavimui bei sintezei
- Šio darbo skaitmeninė kopija