Sveučilište u Zagrebu

Prirodoslovno-matematički fakultet

Mateja Batelić

Impulsno neuronsko računanje

Zagreb, 2019.

Ovaj rad izrađen je u Istraživačkoj jedinici Fotonika i kvantna optika Znanstvenog centra izvrsnosti za napredne materijale i senzore, koji djeluje na Institutu Ruđer Bošković u Zagrebu pod vodstvom dr. sc. Maria Stipčevića, u suradnji sa prof. dr. sc. Hrvojem Buljanom s Fizičkog odsjeka Prirodoslovno-matematičkog fakulteta Sveučilišta u Zagrebu, te je predan na natječaj za dodjelu Rektorove nagrade u akademskoj godini 2018./2019.

Sadržaj

1 Uvod

Razvoj računala odigrao je ključnu ulogu u razvoju tehnologije. Računala svoj postanak bilježe još od razdoblja 5000. godina prije Krista kada su stari Egipćani koristili kamenčiće kako bi zbrajali i oduzimali. Daljnji razvoj uslijedio je kroz izume logaritamskih ili pomičnih računala u 17. stoljeću, zatim Schickardovog prvog mehaničkog kalkulatora 1623. godine nakon kojeg je uslijedio izum Babbageovog diferencijalnog stroja u 19. stoljeću. Vrlo brzo potom slijedi razvoj elektromehaničkih strojeva i računala kakve poznajemo danas. Utjecaj računala u današnjem svijetu je neizmjeran, a to potvrđuje masivna uporaba računala u širokoj proizvodnji i potrošnji, kao i korištenje strojeva u medicinske, znanstvene, obrazovne i mnoge druge svrhe. Računala su omogućila i paralelan procvat svih grana znanosti (medicine, fizike, kemije, biologije, matematike, itd.), od kojih posebno treba istaknuti razvoj kvantne fizike. Paralelan razvoj kvantne fizike i računala doveo nas je do nivoa na kojem je današnja tehnologija, a sve je započelo Einsteinovim objašnjenjem fotoelektričnog efekta, što je jedno od najznačajnijih otkrića u povijesti kvantne mehanike. Važnost fotoelektričnog efekta leži u kvantiziranju svjetlosti odnosno u tome da se svjetlost sastoji od točno određenih količina energije ili kvanata svjetlosti koji su kasnije nazvani fotoni [1]. Objašnjenje fotoelektričnog efekta pokrenulo je dodatna istraživanja svjetlosti pa je tako stvorena i grana fizike fotonika koja se bavi proučavanjem mogućnosti razvoja uređaja utemeljenih na svojstvima međudjelovanja svjetlosti i tvari. Jedan od široko korištenih uređaja danas su detektori fotona koji generiraju signal u obliku električnog impulsa prilikom detektiranja jednog fotona iz dolaznog snopa [2].

Daljnjim razvojem kvantne fizike, pojavila se potreba za računanjem sve težih matematičkih zadataka. Feynman je prvi uočio da je klasičnim računima sve teže računati kompleksnije kvantne sustave pa je predložio da se kvantnim računalima računaju teški matematički problemi.

Danas postoje tri računalne paradigme: klasično računanje, kvantno računanje i impulsno računanje. Klasično računanje svodi se na manipuliranje znamenaka pomoću deterministčkih algoritama. Takav način računanja predstavlja Turingov model u von Neumannovoj arhitekturi i najčešće se izvodi u digitalnoj tehnici gdje je najmanja količina informacije jedan bit. Kod takvog načina računanja nema slučajnosti. Kvantno računalo pak koristi kvantne bitove - qubite. Posebnost qubita je u tome što omogućava superpoziciju dva stanja, što znači da istovremeno može biti u logičkom stanju 0 i logičkom stanju 1, s određenom vjerojatnošću, dok klasični bitovi mogu poprimiti samo jedno od navedena dva stanja. Kvantno računalo koristi načela superpozicije, kvantnog sprezanja i odgovarajuće manipulacije qubitima kako bi postiglo značajno veću brzinu izvođenja izvjesnog i doduše skromnog, ali značajnog skupa matematičkih algoritama. S druge strane, impulsno računanje koristi nedeterminističke slučajne impulse u vremenu koje je moguće dobiti detektiranjem fotona

pomoću uređaja za detekciju fotona. Jedan od takvih uređaja je SPAD čip[1]. Upravo takav način dobivanja slučajnih signala korišten je u ovom radu kako bi se proučavalo impulsno računanje.

Impulsno računanje odnosno RPC[2] ne slijedi ni Turingovu teoriju ni von Neumannovu arhitekturu, kao ni qubite, već više sliči radu mozga. Zbog toga se impulsno računanje još naziva i biološki nadahnuto računanje[3], a nerijetko i neuromorfno [3]. Ovakvo računanje zasniva se na logičkim sklopovima i Boolovoj algebri, ali uz dodatak u odnosu na klasična računala da se koriste slučajni signali, dok je kod klasičnih računala signal deterministički određen i spremljen u memoriji računala. Može se reći da se impulsno računanje oslanja na kvantni efekt, konkretno kvantnu neodređenost (slučajnost), ali koristi samo klasičnu informaciju bez kvantnih korelacija. Za razliku od impulsnih neuronskih mreža, koje imaju velikih problema kod izvršavanja matematičkih operacija, RPC upravo koristi mogućnost izvršavanja matematičkih operacija, kako elementarnih tako i složenijih, u jednom „neuron". RPC računalo posuđuje digitalne impulse od digitalnog računala, slučajnost od kvantne mehanike te masivnu paralelizaciju i neuronalnu strukturu od neuronskih mreža i kvantnog računala. No za razliku od ostalih paradigmi, RPC računalo pored informacije koristi i vrijeme kao dodatnu dimenziju u računanju: trenutno stanje računala ne nosi nikakvu infomaciju već je potrebno statistički pratiti niz stanja kroz neko vrijeme: to ga čini izuzetno robustnim na moguće pogreške u hardveru.

Kod impulsnog računanja (RPC) ulazni podaci su nizovi slučajnih impulsa u kojima se impulsi javljaju slučajno u vremenu s praktično beskonačnom vremenskom rezolucijom, to jest nije ih moguće efikasno spremiti u memoriju. Osim toga jedan te isti broj (na primjer: „3.14") može biti prikazan s beskonačno mnogo različitih nizova slučajnih impulsa. To znači da se kod ovog tipa računanja jedna te ista kalkulacija koja polazi od istih ulaznih podataka nikada ne odvija na isti način, što ima svoje prednosti u npr. borbi protiv prisluškivanja algoritama i rezultata računanja. Rezultat obrade ulaznih podataka je izlazni niz slučajnih impulsa kojeg je također nemoguće efikasno spremiti za kasniju upotrebu. Osim toga taj izlazni niz traje samo toliko dok su na ulazu računala prisutni ulazni nizovi.

Impulsno računanje ima nekoliko jedinstvenih karakteristika, koje su sve redom ključne za funkcioniranje živih bića: RPC operacije koriste mnogo manje resursa (sklopova i energije) nego slične digitalne operacije, omogućuju brzo računanje i sa sporim hardverom te su otporne na pogreške u hardveru (npr. bolesti i ranjavanja). S druge strane, digitalnim računalima moguće je postići daleko veću preciznost i brzinu nego što je moguće dobiti s predvidivom RPC tehnologijom. Današnja digitalna računala rade s brojevima od 64 bita, što odgovara 18 decimalnih znamenki točnosti, dok specijalni programi mogu raditi aritmetiku s proizvoljnim brojem znamenki. Za kriptografiju koja se koristi u svakodnevnom radu na Internetu, koristi se aritmetika s 300 decimalnih znamenki (1 024 bita), kako u

[1] Single Photon Avalanch Diode (eng.)
[2] Random Pulse Computing (eng.)
[3] Biologically Inspired Computing (eng.)

računalima tako i u mobitelima. Kod RPC računala (koje se još teorijski razvija i za koje postoji malo praktičnih realizacija) relativna preciznost eksperimentalno dobivenog rezultata prema stvarnoj (teorijskoj) vrijednosti je u najboljem slučaju 1 naprama 1 000, to jest 3 decimalna mjesta. Dva su ograničenja: hardversko i statističko. RPC radi donekle u analognom režimu, iako koristi digitalne signale i logičke krugove. Analogni dio je u vremenu. Iznos i stabilnost (ponovljivost) kašnjenja signala kroz logičko sklopovlje predstavlja krajnje ograničenje u ostvarivoj preciznosti matematičkih operacija. Statističko ograničenje je i samo dvojako. Prvo imamo odstupanje niza slučajnih impulsa od savršene eksponencijalne distribucije te mogućnosti autokorelacija i međusobnih (cross) korelacija među signalima. Drugo, i fundamentalno problematičnije, jest to što određivanje prosječne frekvencije izlaznog signala traje određeno vrijeme i to se vrijeme ne može skratiti zbog fundamentalnih zakona statistike. Na primjer, da bismo odredili prosječnu frekvenciju niza slučajnih impulsa s točnošću od 1% potrebno je izmjeriti vremensko trajanje pojave barem 10 000 impulsa, a da bi se točnost povećala 10 puta, potrebno je izmjeriti barem 100 puta više impulsa. Naime, iz same pretpostavke eksponencijalne distribucije, proizlazi da preciznost određivanja frekvencije N impulsa iznosi $1/\sqrt{N}$. Zbog toga, od RPC paradigme ne treba očekivati preciznost koja bi zamijenila digitalna računala, pa ni brzinu u izračunu nekog zbroja ili složene matematičke funkcije. Prednosti RPC mogu doći do izražaja u slučajevima gdje je potrebna istovremena obrada ogromnog broja ulaznih parametara (na primjer video slike), a što vodi na moguće bolje učenje i umjetnu inteligenciju [4]. Također RPC bi mogao imati prednosti u slučajevima gdje je potreban rad s vrlo malom energijom, npr. kod autonomnih nenadziranih[4] senzora ili u Svemiru zbog svoje tolerancije na oštećenja hardvera i šum [5]. Pored toga, RPC ima mnogo mogućih ili već dokazanih primjena u dekoderima i enkoderima za potrebe telekomunikacija [6].

Jedna od bitnih razlika ove računalne paradigme s obzirom na druge dvije jest istovremeno računanje sa svim ulaznim parametrima, bez sekvencijalnosti. Dok kod klasičnog i kvantnog računala, od trenutka započinjanja obrade podataka do trenutka njezinog završetka, na izlazu nema nikakvog korisnog podatka. Na primjer, tri broja se množe tako da se prvo pomnože dva, a onda taj rezultat pomnoži s trećim. Ukoliko je račun složeniji, utoliko dulje vremena nema nikakvog korisnog izlaznog podatka. Međutim, kod impulsnog računala, izlazna vrijednost se formira odmah prilikom ulaska ulaznih podataka, tako da čak i ako se oni mijenjaju u vremenu, izlazna vrijednost optimalnom brzinom prati promjene ulaznih parametara. Na taj način se postiže kratko vrijeme odziva čak i sa sporim hardverom. Ovo je izrazito važna karakteristika za preživljavanje živih bića kako bi mogla čim prije, pa makar i s nevelikom točnošću, započeti reakciju na opasnost.

Nakon što je 1956. godine von Neumann dao ideju za istraživanje RPC-a, 50-tih i 60-tih godina prošlog stoljeća pojavilo se nekoliko patenata i članaka na tu temu [7, 8, 9]. Nakon toga, nastupio je

[4] Unsupervised (eng.)

duži period neaktualnosti te teme, sve do 2014. godine kada se ponovo pobudila ideja za razvijanjem računala koja rade na principu sličnom radu mozga živih bića. Paralelno s time razvijala se i umjetna inteligencija pa su u ovom području vrlo nedavno napravljena istraživanja impulsnog računanja korijena i kvadrata brojeva, te polinoma. [10].

U ovom radu proučavana je preciznost matematičkih operacija pojedinih sklopova temeljenih na Boolovoj algebri i binarnim operacijama u svrhu unaprjeđenja osnovnih računskih operacija: zbrajanja, oduzimanja, množenja, dijeljenja i uspoređivanja brojeva. Cilj poboljšanja je postići bolju točnost i veću brzinu sa što manjim korištenjem hardverskih resursa. Dodatno, proučavana je i autokorelacija izlaznog signala koja bi također trebala biti što manja, jer se čini da ona može loše utjecati na točnost impulsnog računanja. U radu su predstavljeni rezultati simulacija koje vjerno imitiraju sklopove za osnovne računske operacije kako bi se ispitala preciznost i učinkovitost poboljšanih sklopova prije njihove realizacije te uopće mogućnost poboljšanja istih.

Realizacija sklopova ostvarena je pomoću FPGA[5] čipa u kojem su uprogramirani sklopovi za spomenute matematičke operacije. Na kraju, prezentirani su rezultati tako realiziranih sklopova te uspoređeni sa teorijskim očekivanjima.

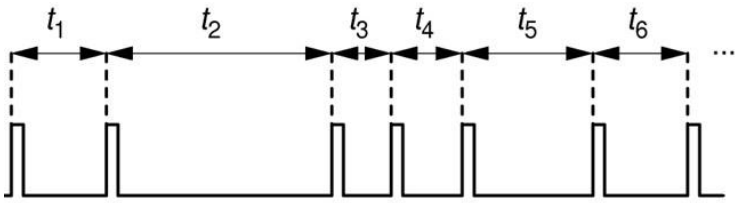
[5] Field-Programmable Gate Array (eng.)

1 Teorijska pozadina

2.1 Detektori fotona kao izvor slučajnosti

Najpoznatiji izvor slučajnih događaja je izvor radioaktivnih raspada. Takva kvantno bazirana slučajnost je idealna za korištenje u računalima sa slučajnim impulsima, no takvi izvori u pravilu ne omogućavaju više od nekoliko desetaka tisuća slučajnih događaja u sekundi zbog sporosti detektora, točnije Geiger-Millerovih brojača, i moguće opasnosti po zdravlje radioaktivnijih izvora te kompliciranе zakonske regulative za nabavu, prodaju, uvoz i izvoz radioaktivnog materijala. Zbog toga, u ovom radu korišten je izvor slučajnih događaja u kojem navedeni sustav zamjenjujemo izvorom konstantnog svjetla i detektorom fotona na koje ono pada, odnosno umjesto radioaktivnih raspada, koristimo detekcije fotona. Pri tome se dobivaju slučajni impulsi po istom principu, ali sa znatno većom frekvencijom i manjom potrošnjom energije.

Detektor fotona je uređaj koji prilikom detekcije fotona proizvodi električni impuls, obično kvadratnog oblika te konstantne amplitude i trajanja. Primjer niza detekcija na vremenskoj osi prikazan je na slici (1). Ukoliko je intenzitet iluminacije detektora konstantan, proces generiranja detekcija je stacionaran te vremena između susjednih impulsa slijede eksponencijalnu distribuciju [11], upravo kao da se radi o detekciji gama zraka pomoću Geiger-Millerovog brojača.



Slika 1: Niz slučajnih električnih impulsa (RPT) u vremenu, pri čemu je svaki impuls konstante duljine trajanja. Vremena između susjednih impulsa t_i $(i = 1, 2, ...)$ slijede eksponencijalnu distribuciju [11].

Detektore fotona koristimo kao izvore slučajnosti u sklopovima za računanje sa slučajnim impulsima proučavanim u ovom radu. Fotoni se detektiraju s nekom vjerojatnošću, a u tu svrhu koristili smo detektore sa SPAD[6] diodama. Kad govorimo o slučajnim događajima, zapravo mislimo na električne signale u obliku logičkih impulsa koji izlaze iz detektora fotona. Uzlazni brid svakog impulsa na izlazu detektora predstavlja jedan slučajni događaj [11].

[6] Single Photon Avalanch Diode (eng.)

2.2 Eksponencijalna distribucija

Eksponencijalna distribucija je posljedica slučajnosti i međusobne neovisnosti fotokonverzija uzrokovanih konstantnom jačinom rasvjete koja pada na SPAD diode. Osnovne pretpostavke koje opisuju izvor slučajnih događaja su sljedeće:

I. Vjerojatnost da će se slučajni događaj dogoditi u sljedećih Δt vremena jednaka je $\lambda\Delta t$ u granici $\Delta t \rightarrow 0$, neovisno o trenutku promatranja.

II. Veličina λ konstantna je u vremenu tj. vrijedi $\lambda(t) \equiv \lambda$.

III. Slučajni događaji su međusobno nezavisni.

Zanima nas funkcija gustoće vjerojatnosti duljine intervala između susjednih događaja $F(t)$ koja proizlazi iz ovih uvjeta, budući da istu možemo eksperimentalno mjeriti i na taj način provjeriti ispravnost pretpostavki za pojedini sustav. Pretpostavimo da se jedan slučajni događaj dogodio u trenutku $t = 0$. Pitamo se kolika je vjerojatnost da će se sljedeći slučajni događaj dogoditi u kratkom periodu nakon isteka proizvoljnog vremena t. Zbog međusobne neovisnosti događaja (uvjet III), ta je vjerojatnost jednaka umnošku vjerojatnosti da se *nije* dogodio nijedan događaj do trenutka t i vjerojatnosti da se dogodio jedan događaj upravo kratko *nakon* trenutka *t*. Da bismo to izračunali, podijelimo t u kratke intervale $\Delta t = t/N$ gdje je N proizvoljan veliki prirodni broj. Prema pretpostavkama I i II, vjerojatnost da se nije dogodio nijedan događaj u bilo kojem intervalu Δt iznosi $1 - \lambda\Delta t$. Zbog pretpostavke III, ukupna vjerojatnost da se nije dogodio događaj do trenutka t jednaka je *N*-toj potenciji izraza $1 - \lambda\Delta t$. Vjerojatnost, pak, da se događaj dogodio u kratkom vremenskom intervalu nakon trenutka *t*, jednaka je $\lambda\Delta t$. Dakle ukupna vjerojatnost da će interval između dva susjedna događaja biti jednak *t* dana je s

$$(1 - \lambda\Delta t)^N \lambda\Delta t. \qquad (1)$$

Uzimajući u obzir da je $\Delta t = t/N$, tražena funkcija gustoće vjerojatnosti dana je kao:

$$F(t)\Delta t = \left(1 - \lambda\frac{t}{N}\right)^N \lambda\,\Delta t, \qquad (2)$$

u granici kada $N \rightarrow \infty$, odnosno:

$$F(t) = \lambda \lim_{N\to\infty} \left(1 - \lambda\frac{t}{N}\right)^N. \qquad (3)$$

Koristeći Eulerovu formulu, prethodni izraz možemo zapisati kao:

$$F(t) = \lambda e^{-\lambda t}. \qquad (4)$$

Time smo kao rezultat pretpostavki dobili eksponencijalnu funkciju gustoće vjerojatnosti koja opisuje slučajno distribuirane događaje u vremenu. Fizički proces koji generira ovakve događaje naziva se Poissonov proces.

Nadalje, možemo pokazati kolika je **vjerojatnost koincidencije** događaja iz dva nezavisna izvora slučajnih događaja (npr. detektora fotona) unutar vremenskog intervala trajanja ΔT_c. Neka je vjerojatnost koincidencije dva događaja $p_c = p_1 \cdot p_2$, pri čemu je p_1 vjerojatnost da se događaj iz prvog detektora nađe u intervalu ΔT_c, a p_2 vjerojatnost da se događaj iz drugog detektora nađe u intervalu ΔT_c. Može se pisati:

$$p_1 = f_1 \int_0^{\Delta T_c} e^{-t f_1} dt = \left(1 - e^{-\Delta T_c f_1}\right), \quad (5)$$

$$p_2 = f_2 \int_0^{\Delta T_c} e^{-t f_2} dt = \left(1 - e^{-\Delta T_c f_2}\right). \quad (6)$$

Odnosno,

$$p_i = 1 - \left(1 - \Delta T_c f_i + \frac{(\Delta T_c f_i)^2}{2!} \mp \cdots\right) = \Delta T_c f_i \mp O(\Delta T_c f_i)^2, \quad (7)$$

pri čemu smo pretpostavili da je promatrani interval kratak u usporedbi s prosječnim intervalima između događaja unutar nizova: $\Delta T_c \, f_i \ll 1$. Stoga, približno vrijedi $p_c = \Delta T_c f_1 \cdot \Delta T_c f_2$.

Iz ovoga slijedi vrlo zanimljiva ideja: ukoliko ΔT_c držimo konstantnim, vjerojatnost koincidencije proporcionalna je umnošku dva proizvoljna realna broja od f_1 i f_2. S obzirom na to da je električki sklop za koincidenciju vrlo jednostavan (AND logička vrata) proizlazi da se množenje dva broja može vrlo jednostavno napraviti ukoliko se frekvencije impulsa shvate kao ulazne i izlazne varijable. Međutim, iz gore navedenog uvjeta slijedi da je frekvencija koincidencije $f_c = p_c/\Delta T_c = f_1 \cdot f_2 \cdot \Delta T_c$ pojedinačno manja (dapače, mnogo manja) i od f_1 i od f_2. Takav niskofrekventni signal zahtijeva dugo vrijeme brojanja impulsa da bi se dobila dobra točnost. To je problem kojeg ćemo riješiti tzv. diskretizacijom vremena pojašnjenom u potpoglavlju *Diskretizacija vremena* [12].

2.3 Mrtvo vrijeme i afterpulsevi

U stvarnosti, na izlazu detektora fotona, ne dobiva se identična eksponencijalna distribucija, kao što je izvedeno u potpoglavlju *Eksponencijalna funkcija i distribucija eksponencijalne funkcije,* zbog dva efekta - mrtvog vremena[7] i preostalih impulsa[8], odnosno tzv. 'afterpulseva'. Nakon što detektor detektira jedan foton, on tada ulazi u proces lavinskog pojačanja[9] naboja koji traje konačno vrijeme. Pritom

[7] Deadtime (eng.)
[8] Afterpulses (eng.)
[9] Avalanch (eng.)

postoji mogućnost da se pojavi novi foton, ali njegova detekcija ne će biti zabilježena, već će se njegov signal nadodati prethodno detektiranom. Takav efekt nazivamo efektom mrtvog vremena detektora, a mrtvo vrijeme možemo shvatiti kao vrijeme u kojem nije moguće detektirati sljedeći foton, odnosno kao vremenski period oporavka detektora. Srećom, ovaj efekt mrtvog vremena utječe na eksponencijalnu distribuciju samo utoliko što je eksponencijalna distribucija odsječena u području od $t = 0$ do mrtvog vremena, odnosno taj dio eksponencijalne distribucije fali. U ovom eksperimentu, mrtvo vrijeme korištenih detektora iznosi 25 ns i znatno je kraće od vremena između diskretiziranih impulsa koje iznosi 1 000 ns pa ovaj efekt možemo zanemariti [2].

Također, prilikom procesa lavinskog pojačanja naboja, postoji pojačana vjerojatnost pojave nositelja naboja, u ovom slučaju elektrona, nakon kratkog vremenskog intervala, koji otprilike iznosi nekoliko desetaka nanosekudni, nakon detekcije stvarnih fotona. Takvi elektroni nisu uzrokovani stvarnim fotonima, već prilikom pojačanja u detektoru dio naboja ostane zarobljen u nekoj nečistoći i pojavi se u obliku signala fotona nakon duljeg vremenskoj intervala prilikom emitiranja iz materijala. Pojava takvih lažnih detekcija naziva se „preostali impulsi“, odnosno tzv. afterpulsevi. Zbog toga, afterpulsevi predstavljaju korelacije između detekcija fotona, što označava odstupanje od njihove slučajnosti, što dovodi do korištenja detektora sa zanemarivo malo vjerojatnosti pojavljivanja afterpulseva, poput fotomultiplikatora. Upravo takvi detektori korišteni su prilikom ispitivanja sklopova u ovom radu jer oni u sebi sadržavaju razrijeđeni plin koji kad se ionizira dolaskom elektrona, omogućuje sporije gibanje čestica tog plina te time smanjuje pojavu afterpulseva. Općenito, ovaj efekt može doprinijeti i do 3% afterpulseva, no u ovom eksperimentu korišteni detektori imaju vjerojatnost afterpulseva oko 0.01% te se stoga i ovaj efekt može zanemariti [2].

2.4 Diskretizacija vremena

Iako se slučajni impulsi mogu pojaviti u bilo kojem trenutku, iz praktičnih razloga mogućnosti ostvarenja sklopova u FPGA čipu, potrebno je vrijeme „diskretizirati“ odnosno slučajne impulse smjestiti u diskretne, periodične vremenske odsječke u kojima impuls postoji ili ne postoji. U našem eksperimentalnom uređaju, izvor slučajnosti predstavlja detektor fotona, no impulsi se „filtriraju“ kroz poseban sklop koji generira impulsni takt[10] perioda T = 1 000 ns. Impuls se generira ukoliko se pojavi barem jedna detekcija unutar tekućeg vremenskog intervala duljine T. U protivnom, impuls se ne generira u tom intervalu. Time slučajnost nije umanjena: diskretni niz slučajnih impulsa može se shvatiti kao niz slučajnih bitova koji poprimaju vrijednost 1 kada je impuls prisutan odnosno 0 kada ga nema i kod kojih je vjerojatnost pojavljivanja impulsa konstantna i jednaka Dt*f, gdje je f prosječna frekvencija slučajnih događaja, a Dt duljina jednog vremenskog intervala. Prednost ove reprezentacije niza

[10] Clock (eng.)

slučajnih impulsa je u tome što sad umjesto frekvencijama baratamo s vjerojatnostima koje su bezdimenzionalne veličine. Korištenje takvih, vremenski diskteriziranih nizova, ima tehnološku prednost bolje prilagođenosti izvedbi praktičnih sklopova u modernim FPGA čipovima.

2.5 Autokorelacijski koeficijent

Autokorelacijski koeficijent [14] s pomakom k, definiran kao:

$$a_k = \frac{\sum_{i=1}^{N-k}(x_i-\bar{x})(x_{i+k}-\bar{x})}{\sum_{i=1}^{N-k}(x_i-\bar{x})^2}, \quad (8)$$

gdje je x_i i-ti bit u nizu od N bitova $\{x_1 \dots x_N\}$, mjeri stupanj korelacije između originalnog niza slučajnih impulsa i njegove kopije pomaknute za k vremenskih intervala. Budući da je autokorelacija u našim sklopovima mala te da opada geometrijskom progresijom (kao eksponencijalna funkcija pomaka k), u ovom radu proučavamo samo autokorelaciju s pomakom 1, što prethodnu formulu svodi na:

$$a_1 = \frac{\sum_{i=1}^{N-1}(x_i-\bar{x})(x_{i+1}-\bar{x})}{\sum_{i=1}^{N-1}(x_i-\bar{x})^2}. \quad (9)$$

2.6 Logička vrata

Logička vrata su jednostavni sklopovi koji predstavljaju logičke operacije koje se temelje na Boolovoj algebri, koju je u 19. stoljeću razvio George Bool. Ona imaju jedan ili više ulaza te samo jedan izlaz. U osnovne logičke sklopove spadaju OR, AND, NOT, NOR, NAND, XOR i XNOR. U daljnjem tekstu opisana su logička vrata OR, AND, NOT i XOR jer su ona ključna prilikom izrade sklopova za operacije zbrajanja, oduzimanja, množenja i dijeljenja. Svi opisani sklopovi prikazani su i u izvedbenoj shemi. Iako je danas gotovo isključivo u uporabi CMOS[11] logika, koja koristi CMOS tranzistore, ovdje prikazujemo stariju DL[12] logiku, koja koristi diode i otpornike, radi jednostavnosti. Postoje pozitivna i negativna DL logika, a navedeni sklopovi prikazani su u pozitivnoj DL logici. Prilikom izvedbe pretpostavlja se idealna I-R karakteristika diode, pri čemu je $R \simeq 5\ k\Omega$, a unutrašnji otpor izvora $R_s \simeq 100\ \Omega$.

[11] Complementary Metal Oxide Semiconductor (eng.)
[12] Digital Logic (eng.)

2.6.1 OR (ILI) vrata

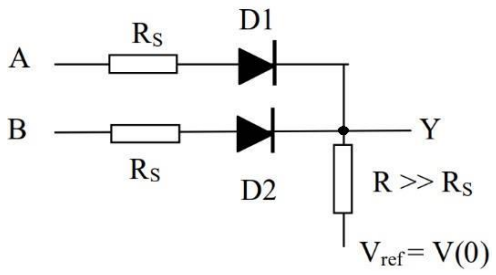
Logička OR (ILI) vrata [15] predstavljaju elektronički sklop koji oponaša operaciju disjunkcije, odnosno stanje na izlazu je 1 ako je stanje bilo kojeg ulaza jednako 1. Grafički prikaz vrata i pripadna tablica istinitosti prikazani su na slici (2).

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

(a) Grafički prikaz logičkih OR vrata. (b) Tablica istinitosti logičkih OR vrata.

Slika 2: Grafički prikaz logičkih OR (ILI) vrata sa pripadnom tablicom istinitosti. A i B predstavljaju stanja na ulazu, a Y predstavlja stanje na izlazu.

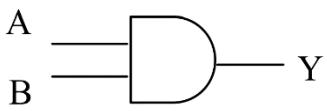
Izvedbena shema OR vrata u pozitivnoj DL logici prikazana je na slici (3). Kod DL logike, ako su oba ulaza A i B u stanju V(0), onda je i izlazni napon V(0), a u slučaju kad je jedan od ulaza A ili B ili oba u stanju V(1), tada je izlazni napon jednak V(1) ili nešto manji. Ulazni naponi se mogu razlikovati po iznosu, ali na izlazu će se pojaviti napon po iznosu jednak najvišem ulaznom naponu. Na taj način se blokiraju ostale diode.



Slika 3: Izvedba logičkih OR (ILI) vrata u pozitivnoj DL logici. R_S i R predstavljaju otpornike, a D1 i D2 diode, dok je V_{ref} izlazni napon. A i B predstavljaju ulazne napone, a Y napon na izlazu.

2.6.2 AND (I) vrata

Logička AND (I) vrata [15] predstavljaju elektronički sklop koji oponaša operaciju konjukcije, odnosno stanje na izlazu je 1 samo ako su stanja svih ulaza jednaka 1. Grafički prikaz vrata, kao i tablica istinitosti prikazani su na slici (4).



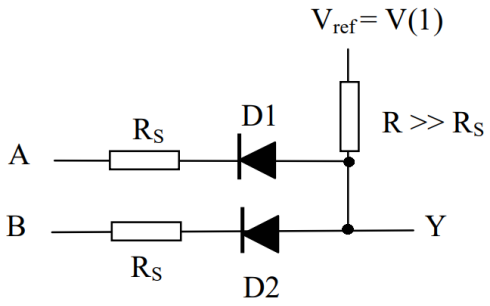
(a) Grafički prikaz logičkih AND vrata.

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

(b) Tablica istinitosti logičkih AND vrata.

Slika 4: Grafički prikaz logičkih AND (I) vrata sa pripadnom tablicom istinitosti. A i B predstavljaju stanja na ulazu, a Y predstavlja stanje na izlazu.

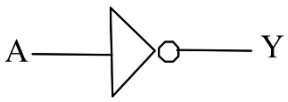
Izvedbena shema AND vrata u pozitivnoj DL logici prikazana je na slici (5). U slučaju kad se napon na nekom od ulaza A ili B ili na oba jednak V(0), tada je i izlazni napon jednak V(0). Jedino ako su oba ulazna napona jednaka V(1), a razlikuju se jedino u šumu, tada će se na izlazu pojaviti napon koji je iznosom jednak najnižem ulaznom naponu. Na taj način će se blokirati ostale diode.



Slika 5: Izvedba logičkih AND (I) vrata u pozitivnoj DL logici. R_S i R predstavljaju otpornike, a D1 i D2 diode, dok je V_{ref} izlazni napon. A i B predstavljaju ulazne napone, a Y napon na izlazu.

2.6.3 NOT (NE) vrata

Logička NOT vrata [15] predstavljaju elektronički sklop koji oponaša operaciju negacije, odnosno stanje na izlazu je nula ako je stanje na ulazu jedan i obrnuto i jedina su logička vrata sa samo jednim ulazom i izlazom pa ih još nazivamo i inverterom. Grafički prikaz vrata, kao i pripadna tablica istinitosti prikazani su na slici (6).

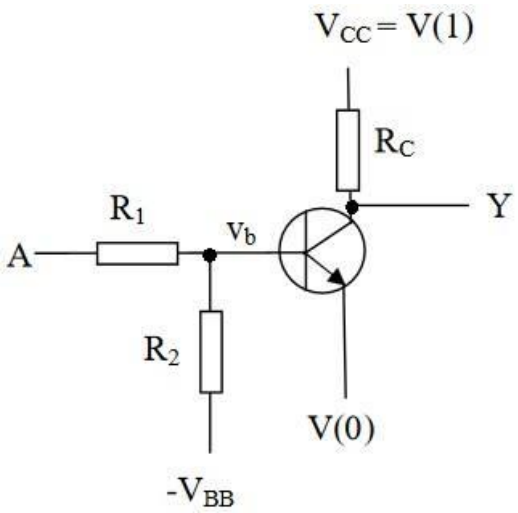


A	Y
0	1
1	0

(a) Grafički prikaz logičkih NOT vrata. (b) Tablica istinitosti logičkih NOT vrata.

Slika 6: Grafički prikaz logičkih NOT (NE) vrata sa pripadnom tablicom istinitosti. A predstavlja stanje na ulazu, a Y predstavlja stanje na izlazu.

Izvedbena shema NOT vrata u pozitivnoj logici ostvarena je pomoću BJT tranzistora, koji je aktivni element, pa se takva logika naziva RTL[13] logika i prikazana je na slici (7).

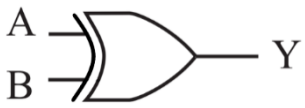


Slika 7: Izvedba logičkih NOT (NE) vrata u RTL logici. R_1, R_2 i R_c predstavljaju otpornike, a V_{CC}, $-V_{BB}$, V_b i $V(0)$ predstavljaju napone. A predstavlja ulazni napon, a Y napon na izlazu.

[13] Resistor-Transistor (eng.)

2.6.4 Exclusive-or (XOR) vrata

Logička XOR vrata [15] predstavljaju elektronički sklop koji je nastao kombiniranjem prethodno opisanih logičkih vrata te radi na principu da je izlazno stanje jednako 1 jedino kad je jedan od ulaznih stanja jednak 1. U slučaju kad su oba ulazna stanja jednaka 0 ili 1, tada je izlazno stanje jednako 0. Grafički prikaz vrata, kao i tablica istinitosti prikazani su na slici (8).



A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

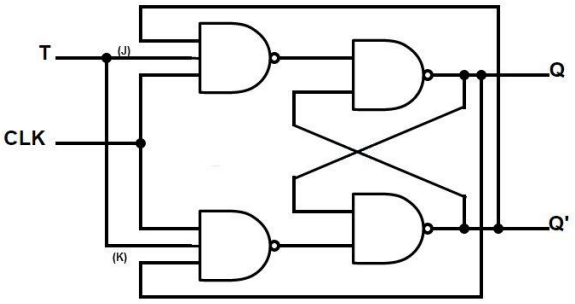
(a) Grafički prikaz logičkih XOR vrata. (b) Tablica istinitosti logičkih XOR vrata.

Slika 8: Grafički prikaz logičkih XOR vrata sa pripadnom tablicom istinitosti. A i B predstavljaju stanja na ulazu, a Y predstavlja stanje na izlazu.

2.7 Flip-flop (FF) i slučajni flip-flop (RFF[14])

Prethodno navedena osnovna logička vrata mogu se kombinirati u logičke sklopove. Ovisno o načinu njihovog kombiniranja, logičke sklopove dijelimo na kombinatorne i sekvencijalne sklopove. Razlika između kombinatornih i sekvencijalnih sklopova je u tome što vrijednost na izlazu kombinatornih sklopova ovisi o trenutnim ulaznim vrijednostima, dok sekvencijalni sklopovi memoriraju prethodne vrijednosti na ulazima pomoću povratne veze. Sekvencijalni sklopovi sastoje se od flip-flopova, koji imaju dva različita stanja. Postoji više vrsta flip-flopova (SR flip-flop, JK flip-flop, T flip-flop, D flip-flop, itd.), a ovdje ćemo opisati TFF[15] (T flip-flop) jer je na temelju njega razvijen TRFF[16]. TFF radi na principu toga da kad dolazni signal na taktnom ulazu (ovdje označen sa CLK) mijenja stanje iz niskog[17] u visoko[18], stanje na izlazu Q ostaje nepromijenjeno ukoliko je na ulazu T nisko stanje, odnosno izlazno stanje se mijenja ukoliko je na ulazu T visoko stanje. Shema TFF-a i pripadna tablica istinitosti prikazani su na slici (9).

[14] Random flip-flop (eng.)
[15] Toggle flip-flop / T-type flip-flop (eng.)
[16] Random T-type flip flop (eng.)
[17] LOW (eng.)
[18] HIGH (eng.)

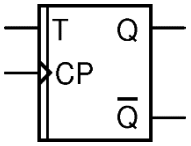


Q	T	Q(t+1)
0	0	0
0	1	1
1	0	1
1	1	0

(a) Shema TFF-a. (b) Tablica istinitosti TFF-a.

Slika 9: Shema TFF-a sa pripadnom tablicom istinitosti. T predstavlja stanje na ulazu, CLK je taktni ulaz, a Q i Q' predstavljaju stanja na izlazu.

TRFF je slučajna verzija TFF-a. Simbol za TRFF prikazan je na slici (10). Taj simbol koristit ćemo i u shemama pojedinih sklopova za RPC računalo u slijedećim poglavljima.



Slika 10: Shema TRFF-a. T predstavlja stanje na ulazu, CP je impulsni ulaz, a Q i predstavljaju stanja na izlazu.

Rad TRFF-a zasniva se na istom principu kao i rad TFF-a, ali uz razliku da impulsni ulaz (označen CP) djeluje sa vjerojatnošću 50%. Zbog toga, ukoliko se ulaz T drži na visokom stanju, pri svakom nailasku impulsa na impulsni ulaz (CP) izlaz Q mijenja stanje na slučajan način, odnosno generira slučajne bitove. Slučajni flip-flop omogućuje generiranje slučajnih impulsa, donošenje slučajnih odluka (npr. propuštanje ili nepropuštanje impulsa) te slučajnih brojeva. Generički slučajni flip-flop po prvi puta predložen je i razvijen u CEMS[19]-FKO[20]. U ovom radu realiziran je putem detektora fotona na čipu.

[19] Center of Excellence for Advanced Materials and Sensing (eng.)
[20] Fotonika i kvantna optika (hrv.)

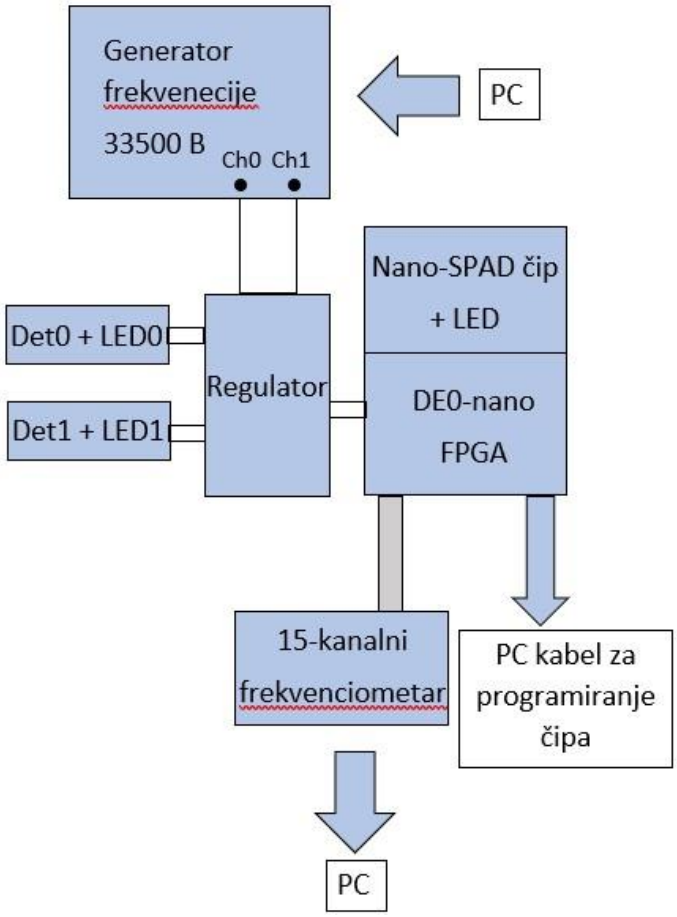
3 Eksperimentalni postav

Eksperimentalni postav prikazan je na slici (11), a pripadna shema na slici (12). Samo srce postava čini univerzalna evaluacijska ploča DE0-nano (Terasic) koja sadrži rekonfigurabilni FPGA čip velikog kapaciteta za logičke sklopove, familije Cyclone IV od Intela. Sklopovi za aritmetiku sa slučajnim nizovima impulsa (RPT[21]) realizirani su u tom čipu.

Dvije nezavisne ulazne varijable (p0 i p1) realizirane su putem dva nezavisna detektora fotona (označenih s D0 i D1) kojima se frekvencije detekcije mogu proizvoljno namjestiti u rasponu od oko 10 kHz do 3 MHz, što odgovara vrijednostima varijable od 0.01 do 0.99. Svaki detektor obasjan je sa po jednom LED[22] diodom (LED0 i LED1) čiju jačinu emisije svjetlosti, odnosno intenzitet regulira poseban regulator na način da frekvencija detekcije fotona na izlazu iz detektora odgovara frekvenciji periodičnog signala kojeg daje dvokanalni generator valnih oblika (Keysight). Budući da se dvije frekvencije koje proizvodi generator frekvencije mogu nezavisno mijenjati pomoću računala, moguće je proizvoljno namjestiti vrijednosti ulaznih varijabli p0 i p1. Time je omogućena automatizacija serija mjerenja u svrhu povećanja brzine, veće točnosti i bolje reproducibilnosti rezultata. Tako dobiveni nizovi slučajnih impulsa odlaze u FPGA čip gdje se obrađuju sklopovima koje smo testirali, a rezultirajuće frekvencije, odnosno varijable, mjere se 15-kanalnim mjeračem frekvencije signala, skraćeno frekvenciometrom, i šalju u računalo. Za neke aritmetičke sklopove bili su nam potrebni slučajni flip-flopovi. Oni su realizirani putem 23 niza slučajnih impulsa koje generira nanoSPAD čip obasjan crvenom LED diodom konstantnog intenziteta svjetlosti. Sam nanoSPAD čip i DE0-nano pločica povezani sa tiskanom pločicom, koja sadrži FPGA čip. Poseban program na računalu postavlja ulazne parametre, mjeri izlazne te sve bilježi u datoteku radi kasnije obrade mjerenja i grafičkog prikaza. Nakon pojave netočnih rezultata uslijed pojave neočekivanih efekata u elektroničkim sklopovima, ključnu ulogu odigrao je osciloskop (Rigol), koji je bio korišten u svrhu debugiranja.

Slika 11: Eksperimentalni postav sa svim korištenim uređajima osim računala. Detaljnije u tekstu.

[21] Random Pulse Train (eng.)
[22] Light Emitting Diode (eng.)

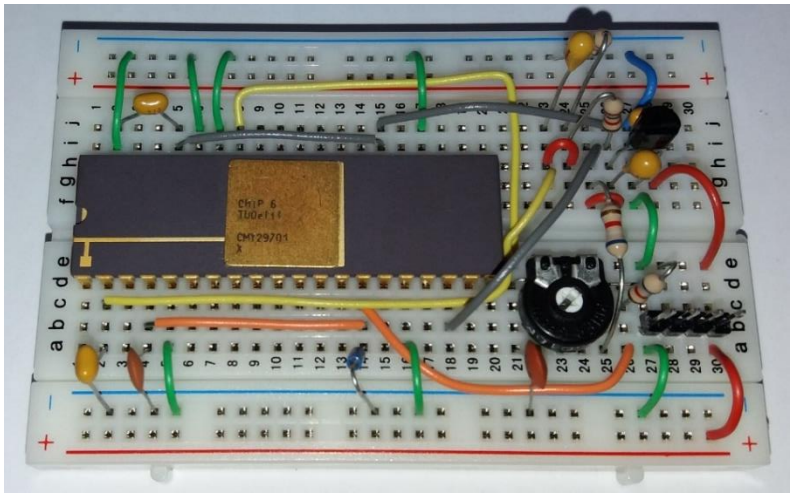


Slika 12: Shema eksperimentalnog postava sa svim korištenim uređajima.

3.1 Nano-SPAD čip

Nano-SPAD čip sastoji se od 23 SPAD fotodiode i prikazan je na slici (13), a odlikuje se sljedećim karakteristikama:

- Efikasnost: 35% na valnoj duljini od 650 nm
- Mrtvo vrijeme: 10 ns
- Nema uočenih afterpulseva (manje od 1 u nekoliko tisuća)
- Maksimalna brzina brojanja: cca 70 MHz
- Niska frekvencija šuma: cca 100 Hz



Slika 13: NanoSPAD čip.

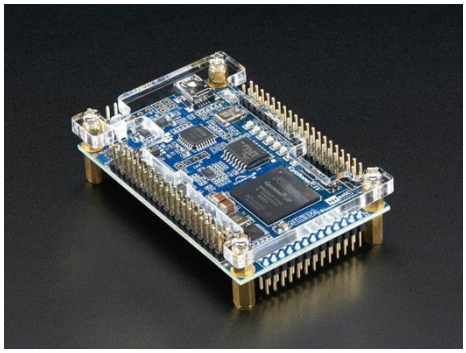
Njegov rad zasniva se na pojačanju naboja koji je nastao konverzijom fotona putem mehanizma lavine[23]. Tu lavinsku navalu struje zaustavljamo pasivnim gašenjem. Svaka od 23 diode inverzno je okrenuta i napaja se preko jednog otpornika dovoljno velikog otpora kako bi u trenutku prolaska struje kroz diodu, napon na njoj pao ispod napona proboja. Tada lavina prestaje, a napon na diodi se ponovo diže do radnog napona te je SPAD dioda spremna za detekciju sljedećeg fotona. Količina naboja koja je uspjela proći kroz diodu od trenutka kad je dioda počela provoditi struju pa sve dok se nije ugasila, prolazi u pojačalo gdje se formira u logički impuls standardne širine i visine [19].

3.2 DE0-nano programabilna pločica s FPGA čipom

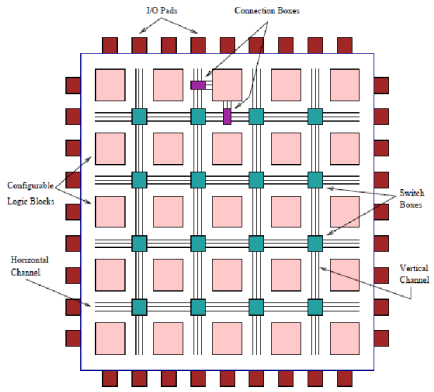
Kako ne bismo morali sastavljati elektroničke sklopove putem izrade tiskanih pločica i lemljenjem, odlučili smo se za moderniji i jednostavniji način – programiranjem univerzalnog logičkog čipa koji se naziva FPGA[24], kako bi se implementirale proučavane operacije zbrajanja, oduzimanja, množenja i dijeljenja. Pritom je korištena komercijalna edukacijska pločica DE0-nano koju proizvodi tajvanska tvrtka Terasic, kako je spomenuto i u uvodnom dijelu poglavlja *Teorijska pozadina*. Na slici (14) prikazan je FPGA čip sa pripadnom unutarnjom arhitekturom. Iako detalji implementacije ovise o

[23] Avalanche (eng.)
[24] Field-programmable gate array (eng.)

proizvođaču, osnovna struktura je u suštini ista. FPGA sklopovi građeni su od logičkih blokova međusobno povezanih poveznicama i prespojnom mrežom [8].



(a) Slika FPGA čipa.



(b) Unutarnja arhitektura FPGA čipa.

Slika 14: FPGA čip sa pripadnom shemom. Detaljnije opisano u tekstu.

Konfigurabilni logički blokovi prikazani su roza bojom na slici (14.b) i namijenjeni su za implementaciju logičkih funkcija. Svaki logički blok sastoji se od niza memorijskih elemenata – flip-flopova, multipleksera i drugih logičkih vrata. Ulazni/izlazni blokovi dostupni su na periferiji FPGA čipa i označeni su crvenom bojom, a omogućuju veze s vanjskim signalima. Konačno, sivom bojom označene su međupoveznice koje omogućavaju interakciju između logičkih blokova.

Programiranje FPGA sklopovlja vrši se pomoću specijaliziranih jezika za opis sklopovlja – HDL[25]. Najpoznatiji HDL jezici su Verilog i VHDL[26]. Prilikom programiranja čipa korišten je jezik Verilog [16, 17].

U pločici DE0-nano, ulazni električni signali pretvaraju se u niz impulsa širine 500 ns unutar vremenskih odsječaka širokih Dt = 1 000 ns. Na taj način dobili smo jedan drugačiji niz slučajnih impulsa u kojem se impulsi mogu javljati samo u diskretnim trenucima (s vremenskom granulacijom od 1 000 ns).

[25] Hardware Description Language (eng.)
[26] Very High Speed Integrated Circuit HDL (eng.)

3.3 Elektronička oprema

U elektroničku opremu spadaju dva detektora fotona, dvokanalni generator valnih oblika (Keysight, model 33500A), osciloskop (Rigol, model DS2202A) te 15-kanalni frekvenciometar, matična ploča i regulator. Detektori fotona, matična ploča, regulator i generator valnih oblika, kao i softver za njegovo povezivanje sa računalom projektirani su i izrađeni u CEMS-FKO, dok su ostali instrumenti nabavljeni na tržištu.

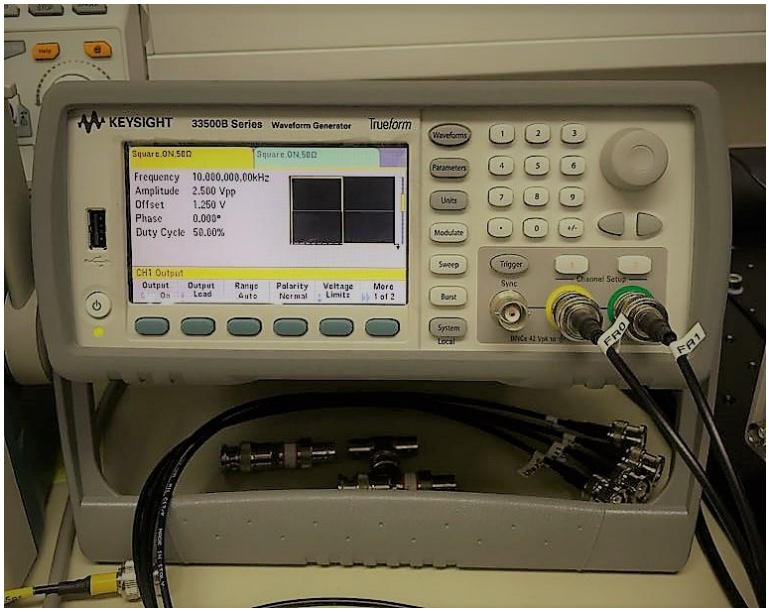
3.3.1 Detektori fotona

Detektori fotona su uređaji koji prilikom detekcije fotona na svom izlazu proizvode električni signal, na način koji je opisan u potpoglavlju *Detektori fotona kao izvor slučajnosti* u dijelu *Teorijska pozadina*, a kao senzor korištena je SPAD dioda SUR500 tvrtke Laser Components. Kao što je objašnjeno u potpoglavlju *Mrtvo vrijeme i afterpulsevi*, ovi detektori nemaju afterpulsing što znači da proizvode vrlo „čistu“ eksponencijalnu distribuciju, odnosno gotovo potpuno slučajne nizove električnih impulsa. Detektori su prikazani na slici (15).

Slika 15: Detektori fotona D0 i D1.

3.3.2 Generator valnih oblika

Generator valnih oblika je uređaj koji proizvodi periodičke signale podesive frekvencije, pri čemu signali imaju neki od standardnih harmoničkih oblika: kvadratni, sinusni, impulsni, trokutasti ili programabilan. Slika (16) prikazuje generator valnih oblika Keysight, model 33500B sa trokutastim harmoničkim oblikom.



Slika 16: Generator valnih duljina sa trokutastim harmoničkim oblikom prikazanim na zaslonu, model Keysight 33500B.

Osim frekvencije, u određenim granicama proizvoljno se mogu namjestiti amplituda, naponski pomak (offset) te mnogi drugi parametri tako generiranih električnih signala. Generator valnih oblika, kao i osciloskop, koristili smo za provjeru rada sklopova (debugiranje), ali ne i za samo RPC računanje, budući da su tako dobiveni signali periodični, to jest maksimalno uređeni, dok nama za RPC računanje trebaju slučajni odnosno maksimalno neuređeni signali.

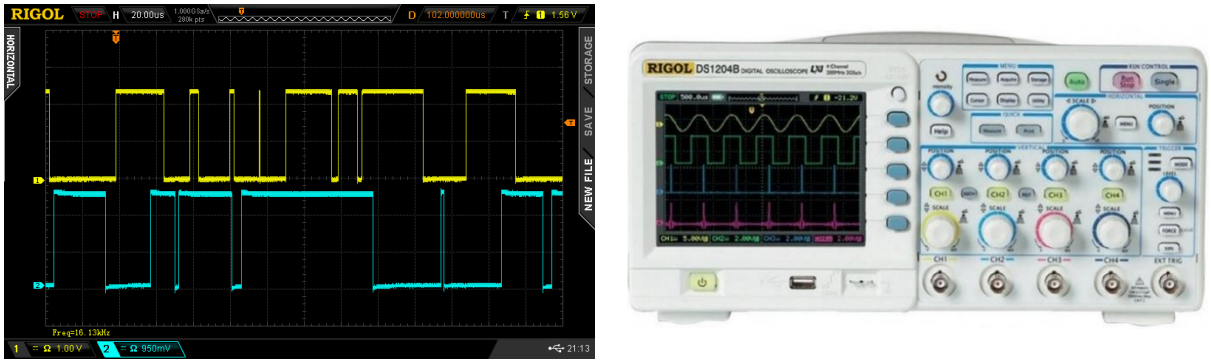
Princip rada zasniva se na generiranju valnog oblika putem revolucionarne, patentirane varijacije metode izravne digitalne sinteze signala (DDS[27]), kojeg je proizvođač nazvao TrueForm. Uobičajena DDS metoda svodi se na to da se digitalno zapisani nivoi napona pretvaraju u analogni napon velikom brzinom i tako oblikuju proizvoljni valni oblik izlaznog signala. Mane uobičajene metode su veliko drhtanje signala[28], što je za faktor 1000 poboljšano u TrueForm metodi.

3.3.3 Osciloskop

Osciloskop je elektronički uređaj za vizualizaciju vremenskih promjena napona, to jest za vizualizaciju valnih oblika te omogućuje promatranje međusobne zavisnosti dvaju ili više signala na zaslonu. Zbog toga je jedina svrha osciloskopa u ovom eksperimentu bilo otklanjanje pogrešaka

[27] Digital Data Signal (eng.)
[28] Jitter (eng.)

prilikom mjerenja koja su davala neočekivane rezultate, a koji su smatrani neispravnima u odnosu na računalno simulirana mjerenja. Osciloskop radi na principu brzog uzorkovanja električnih signala tj. brzog mijenjanja napona u ekvidistantnim vremenskim točkama, pretvaranja napona u digitalni oblik putem analogno-digitalne pretvorbe te crtanja tako dobivene krivulje na LCD zaslonu. Jedan primjer spomenute digitalne krivulje prikazan je na slici (17.a), dok je sami uređaj prikazan na slici (17.b). Koristili smo uređaj marke Rigol, model DS2202A.



(a) Nizovi slučajnih impulsa. (b) Uređaj osciloskopa.

Slika 17: Osciloskop marke Rigol, model DS2202A sa primjerom signala prikazanog na (a) dijelu ove slike.

3.3.4 Ostala elektronička oprema

Frekvenciometar se sastoji od dva ulaza i petnaest izlaza u svrhu istovremenog dobivanja rezultata za različite sklopove te za različite verzije istih sklopova. Time su sama mjerenja ubrzana i na temelju korištenja istih nizova fotona izračunate su proučavane matematičke operacije radi bolje preciznosti i veće točnosti rezultata. U svrhu mjerenja frekvencije korištena je **tiskana pločica** sa FPGA čipom u kojeg su uprogramirani brojači koji služe za mjerenje frekvencije. Podaci koje mjere ti brojači prebacuju se u PC računalo preko USB3.0 sučelja koje je realizirano putem dodatnog komunikacijskog čipa na toj pločici, a koji je i sam programabilan.

Regulator frekvencije poseban je analogno-digitalni elektronički sklop čiji je analogni dio izveden na zasebnoj tiskanoj pločici također razvijenoj u CEMS-LKFO, dok je digitalni dio isprogramiran u firmware-u pločice pločice DE0-nano.

3.4 Programi

Python je programski jezik opće namjene kojeg je napisao Guido van Rossum 1990. godine. Zbog svoje jednostavnosti i lakoće čitanja i pisanja, u ovom eksperimentu korišten je u svrhu simuliranja sklopova za računske operacije kako bi se mogle ispitati različite verzije sklopova za pojedine operacije. Primjer simulacije jednog od programa prikazan je u *Dodatku A*.

ENT[29] je program koji na zadanom nizu slučajnih brojeva ispituje entropiju, korelacije, hi-kvadrat test, srednju vrijednost, itd. Za potrebe ovog eksperimenta korišten je u svrhu računanja korelacijskog koeficijenta a_1 [20].

Quartus Prime je softver tvrtke Intel za programiranje FPGA čipa u kojem su realizirani svi proučavani sklopovi. Korištena je verzija 17.1.

4 Sklopovi za matematičke operacije i njihove simulacije

U ovom poglavlju opisani su proučavani sklopovi i rezultati pripadnih simulacija. Simulacije su napravljene prije samih mjerenja u programu Python koje predstavljaju teorijska očekivanja kako bi se dobiveni rezultati mogli usporediti sa mjerenim vrijednostima. Cilj je bio dobiti što točnije rezultate matematičkih operacija zbrajanja, oduzimanja, množenja i dijeljenja kako bi se pomoću njih mogle izvoditi kompleksnije matematičke operacije poput računanja polinoma ili korijena brojeva. Pri tome, ideja za način rada sklopova inspirirana je živčanim sustavima živih bića gdje su ulazne i izlazne varijable impulsi. Da bi naši sklopovi radili, ključno je da su ulazni impulsi slučajni, u smislu koji je objašnjen ranije. Jedna od mnogih motivacija za ovakav pristup računanju je obrađivanje ulaznih podataka u neprekidnom nizu[30], što omogućuje maksimalnu brzinu uz hardver zadane brzine. Maksimalna brzina obrade postiže se time što se izlazni impulsi formiraju gotovo istovremeno s nailaskom ulaznih impulsa, bez prethodne akumulacije i obrade, kao što to radi digitalno računalo. Osim toga, zbog stohastičke prirode signala (greška opada s recipročnim kvadratnim korijenom vremena brojenja izlaznih impulsa), glavni dio informacije dolazi prvi, a popravci mnogo kasnije pa je i to razlog efikasnosti obrade podataka RPC načinom. U ovom radu krenuli smo od sklopova poznatih u znanstvenoj i patentnoj literaturi i unaprijedili ih našim idejama.

Unaprijeđeni sklopovi trebaju imati što manje logičkih vrata kako bi njihova izrada bila što ekonomičnija u smislu utroška prostora, energije te u pogledu cijene. Sklopovi za zbrajanje i množenje imaju minimalan broj logičkih vrata, što znači da ih je nemoguće više od toga pojednostaviti pa su ti sklopovi ispitivani i eksperimentalno i pomoću simulacija kako bi se provjerila koreliranost izlaznog

[29] Pseudorandom Number Sequence Test (eng.)
[30] Streamline computing (eng.)

signala. Sklopovi za oduzimanje i dijeljenje unaprijeđeni su u odnosu na one iz članaka te su eksperimentalno ispitani samo poboljšani sklopovi dok su i jedni i drugi simulirani. Performanse svih sklopova dodatno su ispitivane preko autokorelacijskih koeficijenata, koristeći program za statističku analizu BENT[31], modificirane verzije programa ENT opisanog u potpoglavlju *Programi* u dijelu *Eksperimentalni postav*. Autokorelacijski koeficijenti su od iznimne važnosti jer nam daju uvid u koreliranost izlaznog signala dobivenog proučavanim sklopovima, a cilj je da izlazni signal bude što manje koreliran odnosno da korelacija teži u nulu. Naime, pri složenim računima u RPC računalu, izlazni signali iz nekih sklopova služit će i kao ulazni signali u druge sklopove, radi daljnje obrade podataka. Međutim, sklopovi za RPC-u funkcioniraju pod pretpostavkom da ulazni signali nemaju autokorelacije i da su međusobno nekorelirani. Ukoliko to nije slučaj, odnosno ukoliko ulazni impulsi nisu posve slučajni, dolazi do pogrešaka u računanju. Mi u ovom radu nismo proučavali utjecaj autokorelacije na točnost, već to ostavljamo za nastavak istraživanja. Također, autokorelacije su i u literaturi za sada dosta slabo istražena problematika.

Prije početka izlaganja, bitno je ovdje dati nekoliko općih napomena.

Prvo, svi programi za kompjutorske simulacije i obradu podataka napisani su počevši od „prazne stranice“. Ne koristimo komercijalne ili tuđe programe budući da se radi o novom području istraživanja te stoga takvih programa još nema.

Drugo, pri simulaciji rezultata binarnih operacija za jedan par ulaznih parametara (p_0, p_1) simulurali smo najmanje $4 \cdot 10^7$ impulsa (bitova) radi postizanja statističke greške manje od mjerenih veličina. To je učinilo dolje prezentirane simulacije vrlo zahtjevnima u smislu količine CPU vremena.

Treće, nerijetko u našim sklopovima javlja se niz impulsa vjerojatnosti 1, dakle periodički signal kod kojeg su svi mogući impulsi ostvareni. U našim eksperimentima to je naprosto simetrični kvadratni signal frekvencije 1 MHz, a u simulacijama i shemama ga dogovorno označavamo oznakom "CLK".

Konačno, pri konstrukciji novih odnosno poboljšanih sklopova ovdje ispitivanih, bitan element predstavlja slučajni T flip-flop (TRFF), opisan u potpoglavlju *Slučajni flip-flop* poglavlja *Teorijska pozadina*, koji je izumljen u CEMS-FKO.

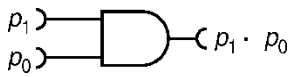
4.1 Opis i simulacija sklopa za množenje

4.1.1 Množenje pomoću logičkih AND vrata

Najjednostavniji RPC sklop je, možda ponešto iznenađujuće, sklop za množenje. On se sastoji od samo jednih logičkih AND vrata, kao što je prikazano na slici (18). Ovaj sklop radi na principu da se

[31] Better ENT (eng.)

na izlazu generira jedan impuls samo u slučaju kada se na ulazu pojave istovremeni impulsi, to jest u istom vremenskom intervalu trajanja Dt. U svim ostalim slučajevima, izlazna vrijednost je 0, odnosno izlaznog impulsa nema. Ako su vjerojatnosti impulsa dvaju ulaznih signala p_0 i p_1, vjerojatnost impulsa na izlazu je $p_0 \cdot p_1$ budući da se radi o vjerojatnosti istovremenog događanja dva nezavisna događaja, točnije impulsa. Uočimo da bi se množenje tri broja moglo ostvariti s jednim logičkim AND vratima koja imaju tri ulaza, odnosno množenje n varijabli bi se moglo ostvariti pomoću jednih logičkih AND vrata s n ulaza. Ovakva realizacija množenja nije samo krajnje ekonomična u količini potrebnog hardvera nego nema sukcesivnosti niti produljenja vremena za izvršavanje te operacije s brojem ulaznih varijabli, što je u oštroj suprotnosti s načinom rada digitalnih računala gdje množenje zahtjeva daleko veću količinu logičkih vrata i gdje bi se n brojeva moralo sukcesivno množiti $n-1$ puta.



Slika 18: Shema sklopa za množenje pomoću logičkih AND vrata. p_0 i p_1 predstavljaju vjerojatnosti jedinice.

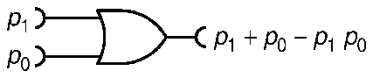
Prema shemi na slici (18), napravljena je simulacija sklopa za množenje, kako bi se prije samih mjerenja dobilo osjećaj o očekivanim vrijednostima i točnosti računanja s upravo opisanim sklopom. Simulacija radi na principu da se zadaju vjerojatnosti impulsa ulaznih nizova, p_0 i p_1, a program putem pseudoslučajnog generatora slučajnih brojeva generira nizove jedinica i nula koji predstavljaju impuls odnosno odsustvo impulsa respektivno, u pojedinom vremenskom odsječku Δt, na način da i tim nizovima vjerojatnosti pojave impulsa odgovaraju zadanim vjerojatnostima p_0 i p_1. Koristeći AND logičku operaciju među ulaznim nizovima bitova, program simulira izlazni niz impulsa. Radi lakše analize izlaznog niza, on se na disk računala ispisuje u obliku jedinstvenog binarnog niza (binarna datoteka) u kojem jedinica predstavlja prisutnost, a nula odsutnost impulsa. Daljnja obrada podataka putem programa BENT radi se nad tom izlaznom binarnom datotekom. S obzirom da je ovaj sklop već ranije ispitivan eksperimentalno, u ovom radu nisu prikazani rezultati simulacija već samo mjerenja prikazana u poglavlju *Rezultati*.

4.2 Opisi i simulacije sklopova za zbrajanje

4.2.1 Približno zbrajanje pomoću logičkih OR vrata

Sklop za približno zbrajanje prikazanog na slici (19) može se realizirati pomoću logičkih OR vrata, koja rade na principu da na izlazu propuštaju bilo koji impuls koji se pojavi na barem jednom od

ulaza. Problem predstavlja ako se na oba ulaza istovremeno pojave impulsi jer tada logička OR vrata na svom izlazu daju samo jedan impuls, odnosno jedan impuls se izgubi. Što su veće vjerojatnosti istovremene pojave impulsa na ulazima, to je navedeni efekt veći, kao i pripadna pogreška zbrajanja. Iz ove analize jednostavno je zaključiti da je vjerojatnost jedinice na izlazu jednaka $p_0 + p_1 - p_0 \cdot p_1$. Dakle, račun ovakvog zbrajanja je ispravan samo kada su je umnožak vjerojatnosti ulaznih impulsa vrlo mali naspram njihovog zbroja. No, to nužno ne mora biti slučaj budući da je jedino ograničenje na ulazne vjerojatnosti p_0 i p_1 to da su one u rasponu [0, 1]. Druga korist ovog sklopa može biti ako se u računu baš traži ova funkcija, naime ona se računa vrlo precizno.



Slika 19: Shema sklopa za približno zbrajanje pomoću logičkih OR vrata. p_0 i p_1 predstavljaju vjerojatnosti jedinice.

Prema shemi na slici (19), napravljena je simulacija sklopa za množenje, kako bi se prije samog eksperimenta još jednom uvjerili u ispravnosti rada sklopa i postojanja mogućeg mjesta za poboljšanje. Simulacija radi na sličnom principu kao i simulacija sklopa za množenje, samo što se ovdje koristi logička operacija OR među ulaznim nizovima bitova. Opet, radi lakše analize izlaznog niza, on se na disk računala ispisuje u obliku jedinstvenog binarnog niza (binarna datoteka) u kojem jedinica predstavlja prisutnost, a nula odsutnost impulsa. Daljnja obrada podataka vrši se nad tom izlaznom binarnom datotekom, pomoću programa BENT, kao i kod sklopa za množenje. Analogno kao i kod sklopa za množenje, u ovom radu predstavljeni su samo rezultati mjerenja u poglavlju *Rezultati*.

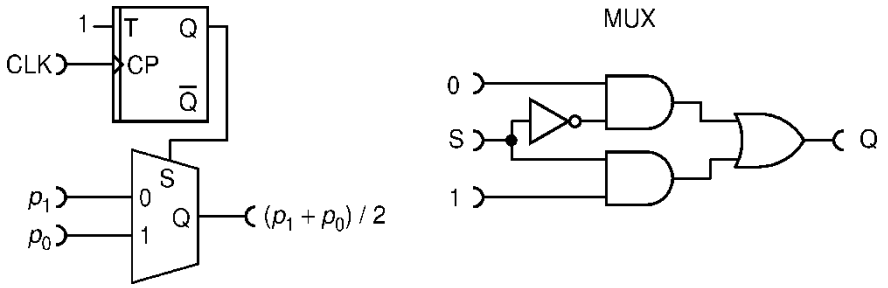
Naivno, moglo bi se pomisliti da ovaj sklop ulazi u zasićenje kada su ulazne vjerojatnosti „dovoljno“ velike. Naime, njihov zbroj može biti u rasponu od 0 do 2, a izlazna vjerojatnost jedinice ne može premašiti 1. Zbog toga se ovaj sklop u literaturi ponekad naziva *saturirajuće zbrajalo* [18]. Međutim, može se pokazati da rezultat matematičke operacije koju obavlja ovaj sklop ne može premašiti 1. Krenimo od jednakosti:

$$p_0 + p_1 - p_0 p_1 = 1 - (1 - p_0)(1 - p_1). \qquad (10)$$

Iz navedene jednakosti može se uočiti da su oba izraza u zagradama u granicama intervala [0, 1], što znači da je i njihov produkt u istim granicama, čime dobivamo da je oduzimanjem tog produkta od jedinice, taj produkt i dalje u granicama intervala [0, 1], što je i trebalo dokazati. To znači da ovaj sklop ne ulazi u zasićenje i da uvijek precizno računa funkciju $p_0 + p_1 - p_0 p_1$ za bilo koje moguće vrijednosti ulaznih parametara.

4.2.2 Zbrajanje s MUX[32] sklopom

Sklop za egzaktno zbrajanje može se realizirati pomoću multiplexera odnosno MUX sklopa s 2 ulaza. Zbrajalo s MUX sklopom, kao i sam MUX sklop, prikazani su na slici (20). MUX radi na principu propuštanja jednog ili drugog ulaza, ovisno o tome da li je kontrolni ulaz S jednak 1 (visoko stanje) ili 0 (nisko stanje). Ako sad zamislimo da ulaz S provodi pola vremena u stanju 0, a pola u stanju 1 na slučajan način, onda će vjerojatnost impulsa na izlazu iz MUX-a biti jednaka $(p_0 + p_1)/2$. Problem pojave dvostrukih pulseva na ulazu sada je izbjegnut jer MUX uvijek provodi na svoj izlaz samo jedan od ulaza, ali je zbroj sada podijeljen s faktorom 2. To međutim nije mana nego prednost budući da se egzaktno zbrajanje ne bi uopće moglo postići za slučajeve kada je zbroj ulaznih vjerojatnosti veći od 1. Na ovaj način izlazna vjerojatnost je točno ograničena na raspon [0, 1]. Preostaje pitanje kako osigurati da je ulaz S u slučajnom stanju i u koincidenciji s ulaznim impulsima. U literaturi ovaj problem nema zadovoljavajuće rješenje (ili se rješava pseudoslučajno), dok smo ga mi riješili egzaktno putem slučajnog flip-flopa, kako je i prikazano na slici (20), kojeg „pogoni" signal vjerojatnosti 1, što znači da je prisutan svaki impuls. Taj signal je na slici (20) označen s CLK. Vrijednost „1" na ulazu TRFF-a znači da je taj ulaz konstantno na visokom logičkom nivou tako da TRFF generira slučajan bit u sinkronizaciji sa svakim ulaznim impulsom.



(a) Shema sklopa za točno zbrajanje. (b) Shema MUX sklopa.

Slika 20: Shema sklopa za točno zbrajanje pomoću MUX sklopa i TRFF-a.

Prema shemi na slici (20), i za ovaj sklop napravljena je simulacija sklopa za egzaktno zbrajanje. Simulacija radi na sličnom principu kao i simulacija sklopa za množenje, samo što se ovdje koristi logička operacija MUX među ulaznim nizovima bitova. Kao i prije, radi lakše analize izlaznog niza, on se na disk računala ispisuje u obliku jedinstvenog binarnog niza (binarna datoteka) u kojem jedinica predstavlja prisutnost, a nula odsutnost impulsa. Daljnja obrada podataka radi se nad tom izlaznom binarnim datotekom pomoću programa BENT. Kao i kod prethodna dva sklopa, u ovom radu prikazani su samo rezultati mjerenja u poglavlju *Rezultati*.

[32] Multiplexer (eng.)

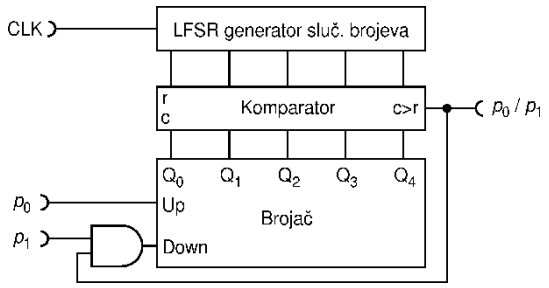
Nužan uvjet da bi izlazni niz bio Poissonov proces potrebno je da ulazni nizovi također budu slučajni Poissonovi procesi u sve tri do sada navedene operacije (AND, OR i MUX), koji kao takvi nemaju autokorelacije. Stoga je za precizan rad tih sklopova bitno da su ulazni nizovi bez autokorelacija, a to je općenito slučaj za sve RPC sklopove.

4.3 Opisi i simulacije sklopova za dijeljenje

Prema dosadašnjem stanju u literaturi, poznate metode oduzimanja zasnivaju se na dijeljenju, stoga sklopove za dijeljenje opisujemo prve.

U radu [10] opisano je dijelilo te je prikazano na slici (21), a radi na principu negativne povratne sprege: izlazni signal se množi djeliteljem, dok sklop s brojačem i komparatorom magnitude „namješta" izlazni rezultat y tako da je $y \cdot p_1 = p_0$, iz čega onda proizlazi da je $y = p_0/p_1$. Svaki impuls ulaznog niza p_1 povećeva stanje brojača za 1, dok umnoženi izlazni impulsi i impulsi niza p_0 smanjuju stanje brojača ujedno povećavajući vjerojatnost izlaznih impulsa. To vodi na uspostavljenje ravnoteže koja se postiže u slučaju kad je $y \cdot p_1 = p_0$. Kritično stanje nastupa kad je $p_1 \approx p_0$ jer tada brojač može lako premašiti maksimalni kapacitet ili odbrojiti ispod nule. Da se to ne bi dogodilo, iako nije nacrtano, pretpostavlja se da je brojač maksimiziran na svoj najveći broj i minimiziran na nulu. Brojač većeg kapaciteta, odnosno s više bitova, stoga daje veću točnost u tom području ulaznih varijabli te je stvar procjene za pojedinu primjenu koliko se bita želi uzeti.

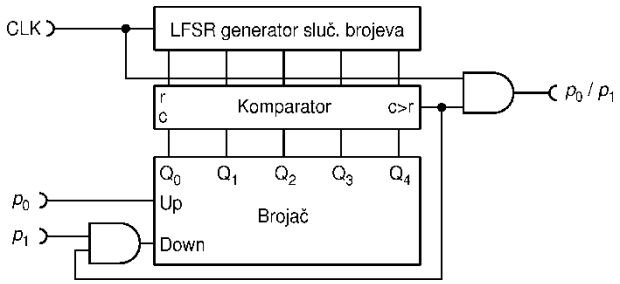
Prije svega, uočili smo da je sklop opisan u spomenutom znanstvenom radu pogrešan, to jest vrijedi sljedeće: 1) opisani sklop uopće ne proizvodi slučajne impulse na izlazu već kvadratne impulse promjenjive duljine i 2) frekvencija tih impulsa je drastično manja od potrebne frekvencije. Nije jasno otkud ta pogreška no činjenica je da u tom radu sklop nije ni simuliran ni eksperimentalno izveden pa se greška mogla potkrasti, iako bi se moglo raditi i o namjernom pojednostavljivanju detalja sklopa. Zbog toga smo unaprijedili opisani sklop da radi ispravno preko dvije verzije te je na kraju prezentiran i sklop bez komparatora i TRFF-ova.



Slika 21: Shema sklopa za dijeljenje prema radu [10] za kojeg je utvrđeno da ne radi ispravno.

4.3.1 Sklop za dijeljenje 1

S obzirom na prethodno uočene greške sklopa za dijeljenje prikazanog na slici (21), zaključeno je da bi ispravan sklop trebao imati još jedna logička AND vrata spojena na CLK (signal vjerojatnosti 1). Za potrebe ovog rada ispravan sklop nazvan je S31 te je prikazan na slici (22).

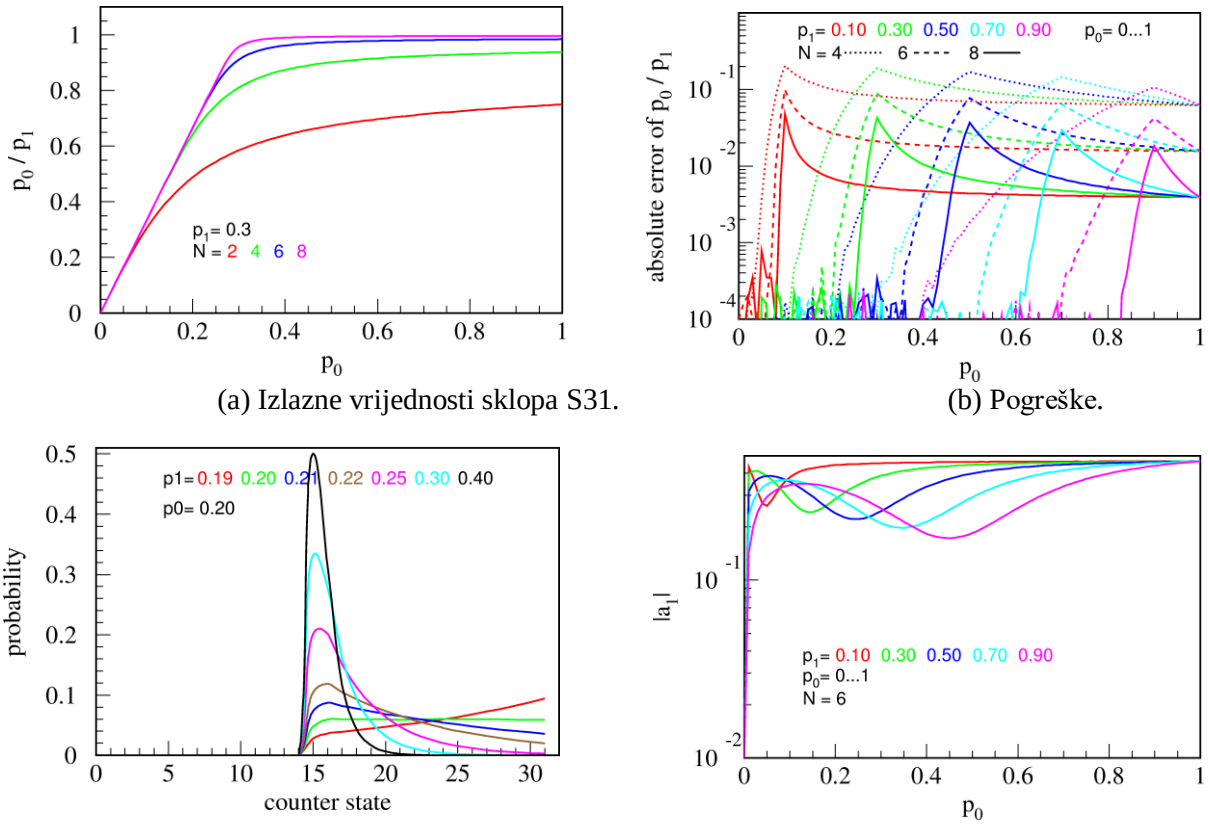


Slika 22: Shema ispravnog sklopa za dijeljenje S31 kod kojeg su dodana logička AND vrata spojena na taktni impuls CLK.

Na sličan način kao što je opisano ranije, načinjen je program za kompjutorsku simulaciju sklopa S31, samo što je ovaj složeniji te je omogućavao specificiranje broja bitova brojača, a sadržava i simulaciju distribucije stanja brojača kako bi se vidjelo što se događa s brojačem za pojedine vrijednosti ulaznih varijabli, budući da preciznost dijeljenja ovisi o kapacitetu brojača, odnosno o najvećem broju kojeg brojač može prikazati. Rezultati simulacija prikazani su na slici (23).

Slika (23.a) prikazuje izlazne vrijednosti djelitelja kao funkcije p_0 u rasponu [0, 0.85], za fiksnu vrijednost $p_1 = 0.3$. Simulacije su načinjene za brojače s brojem bitova N = 2, 4, 6 i 8 te su prikazane krivuljama različitih boja.

Radi lakšeg i točnijeg praćenja odstupanja rezultata dijeljenja od stvarne vrijednosti rezultata p_0/p_1, nacrtan je i graf pogreške (23.b) koji prikazuje pogrešku kao funkciju p_1 za diskretne vrijednosti $p_1 = 0.1, 0.3, 0.5, 0.7$ i 0.9 (krivulje raznih boja), p_0 u rasponu [0, 1] i broja bitova brojača N = 4, 6 i 8 (različit tip crtkanja krivulja). Nadalje, slika (23.c) prikazuje distribuciju stanja brojača kapaciteta N = 5 bita promatranu kroz dulje vrijeme za 7 parova vrijednosti p_1 i p_0. Moguća stanja brojača su u rasponu [0, 31]. Konačno, na slici (23.d) prikazana je autokorelacija izlaznog niza za široki raspon ulaznih parametara i N = 6 kao neku tipičnu veličinu brojača.



(a) Izlazne vrijednosti sklopa S31.

(b) Pogreške.

(c) Distribucija stanja brojača.

(d) Autokorelacija izlaznog niza.

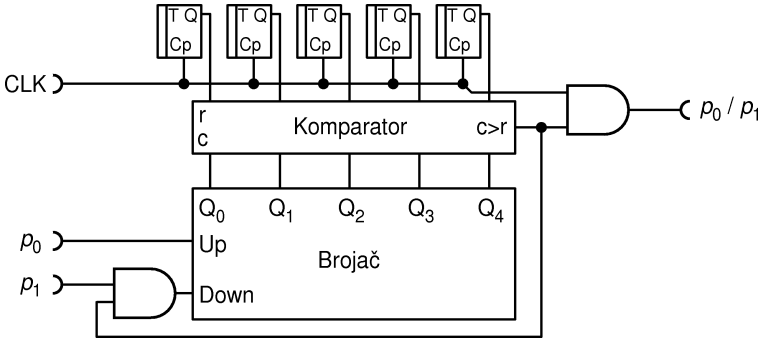
Slika 23: Simulacija sklopa za dijeljenje S31.

Iz gore prikazanih slika simuliranog sklopa S31 vidimo da je preciznost rezultata dijeljenja nedovoljno dobra, naročito blizu kritičnog uvjeta $p_1 \approx p_0$, ali i da se poboljšava s kapacitetom brojača. Međutim pogreška od oko 0.7% se zadržava i daleko od kritičnog uvjeta čak i kad je N = 8 bitova. Nadalje, uspoređujući distribucije stanja brojača za različite vrijednosti broja N, uočeno je da za p_1 dovoljno veći od p_0, brojač se „drži“ sredine maksimalnog broja kojeg može postići, uz određene fluktuacije koje su tu namjerno izazvane slučajnošću LFSR[33]-a kako bi se smanjila autokorelacija izlaznog niza. Međutim, kada je $p_0 \geq p_1$ brojač ulazi u zasićenje, a izlazni niz u vjerojatnost 1. Također, stanja brojača pokazuju da su korelacije najmanje kod uvjeta $p_0 \approx p_1/2$, iako su i tada veće od broja cca 0.2, dok su u ostalim slučajevima gotovo maksimalne. Niz s toliko visokom autokorelacijom vrlo vjerojatno nije upotrebljiv u daljnjim računima RPT-a.

[33] Linear Feedback Shift Register (eng.)

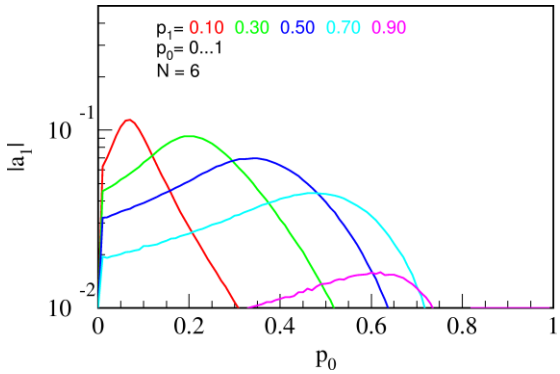
4.3.2 Sklop za dijeljenje 2

Vrlo jaka autokorelacija izlaznog niza u sklopu za dijeljenje S31 nastaje zbog toga što LFSR proizvodi niz ulančanih pseudoslučajnih bitova koji izazivaju korelacije u izlaznom nizu. Najjednostavniji način da se to ukloni jest da se LFSR zamijeni generatorom stvarno slučajnih bitova i to na način da je svaki bit nezavisna uniformna varijabla. To se može vrlo jednostavno postići upotrebom TRFF-ova, kako je prikazano na slici (24). Takav sklop za dijeljenje nazivamo S30.



Slika 24: Shema sklopa za dijeljenje S30 sa slučajnim flip-flopovima (TRFF-ovima).

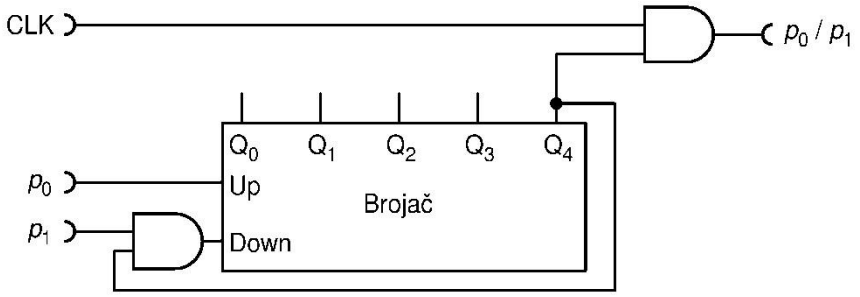
Sklop S30 simuliran je jednako kao i sklop S31. Rezultati za oba sklopa, za vrijednost funkcije, pogrešku i distribuciju stanja brojača identični su do na statističke pogreške te stoga oni nisu prikazani u nastavku ovog potpotpoglavlja. Ono što je različito jest autokorelacija izlaznog niza: kao što se moglo i očekivati, S30 pokazuje znatno manju autokorelaciju, što je vidljivo na slici (25). Autokorelacija se ne smanjuje značajno za brojač s brojem bitova većim od N = 6, iako se točnost i dalje povećava s porastom broja N, jednako kao i kod sklopa S31. Autokorelacija izlaznog niza najmanja je za $p_0 \approx (2/3) \cdot p_1$ te je uglavnom u rasponu brojeva od 0.01 do 0.1, što bi moglo biti zadovoljavajuće za preciznost u daljnjoj obradi signala u RPC računalu.



Slika 25: Simulacija autokorelacijskog koeficijenta a_1 izlaznog niza sklopa za dijeljenje S30.

4.3.3 Sklop za dijeljenje 3

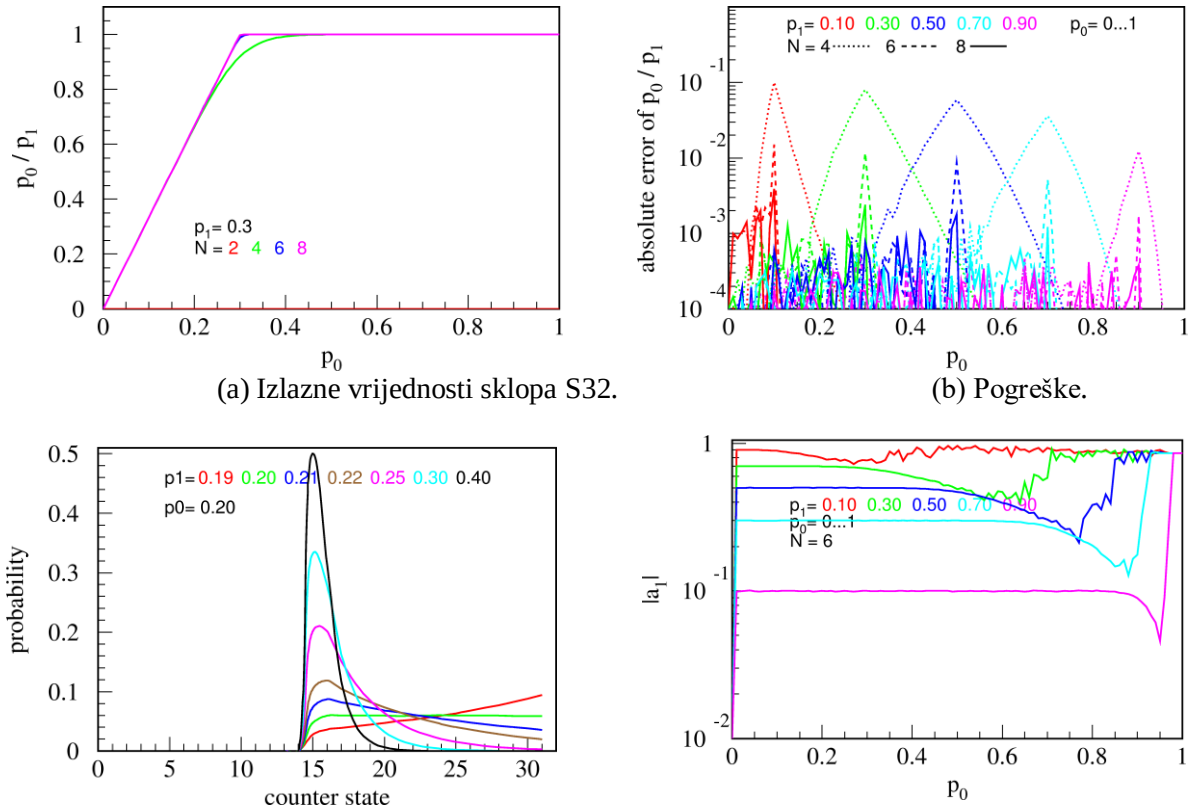
U prethodna dva dijela, komparator magnitude ima najveću složenost. Sklop za usporedbu dva 4-bitna broja ima oko 56 ekvivalentnih logičkih vrata s 2 ulaza, što je usporedbom s jednostavnošću sklopova za množenje i zbrajanje, daleko financijski nepovoljnije. Nadalje, sklopovi za slučajnost također su skupi u smislu hardverskih resursa, a sve to zajedno nije rezultiralo velikom točnošću dijeljenja. Čini se kao da su mala složenost i dobra autokorelacija međusobno suprotstavljeni zahtjevi. Kako bismo ispitali mogućnost povećanja točnosti dijeljenja i pojednostavljenja sklopa za dijeljenje, konstruirali smo novi originalni sklop S32 u kojem nema ni komparatora ni izvora slučajnosti. Taj sklop prikazan je na slici (26).



Slika 26: Shema sklopa za dijeljenje S32.

Simulacije ovog sklopa isprogramirane su na prethodno opisan način te su prikazane jednakim redoslijedom, kao i za sklop S31, na slici (27). Slika (27. a) prikazuje izlazne vrijednosti djelitelja kao funkcije p_0 u rasponu [0, 0.85], za fiksnu vrijednost $p_1 = 0.3$ i veličine brojača N = 2, 4, 6 i 8 bita, dok slika (27.b) prikazuje odstupanja od točnog rezultata. Distribucija stanja brojača prikazana na slici (27.c) te autokorelacijski koeficijent a_1 na slici (27.d).

Promatrajući slike (27.a) i (27.b) vidimo da je ovaj sklop S32 daleko točniji od prethodna dva (S31 i S30), čak i za brojač od samo N = 6 bitova. Primjećujemo i da se pogreška naglo smanjuje izvan kritičnog područja $p_0 \approx p_1$, što je također značajno poboljšanje. Distribucija stanja brojača prikazana na slici (27.c) ne pokazuje veće neobičnosti: i dalje brojač ulazi u zasićenje, ali to ne smeta budući da je tada izlazni signal vrlo blizu vjerojatnosti 1.



(a) Izlazne vrijednosti sklopa S32.

(b) Pogreške.

(c) Distribucija stanja brojača.

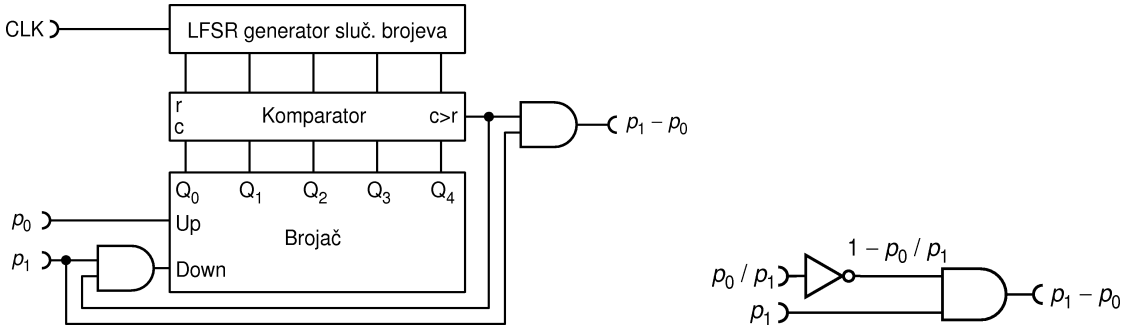
(d) Autokorelacija izlaznog niza.

Slika 27: Simulacija sklopa za dijeljenje S32.

Time su postignuti svi glavni ciljevi: znatno veća točnost i bitno jednostavniji sklop s manje upotrijebljenih resursa. Međutim, plaćena je cijena u visokoj autokorelaciji izlaznog signala, koja vrlo vjerojatno nije dovoljno niska za daljnje korištenje izlaznog niza u RPC-u. No, ovdje smo naučili kako postići preciznost, kao i što utječe na smanjenje autokorelacije pa napore budućih istraživanja treba usmjeriti ka kombiniranju dobrih svojstava raznih sklopova za dijeljenje, uz bolje razumijevanje samih autokorelacija.

4.4 Opisi i simulacije sklopova za oduzimanje

Kao što je spomenuto gore, prema dosadašnjem stanju u literaturi, poznate metode oduzimanja zasnivaju se na dijeljenju. Jedan takav sklop opisan je u radu [10] te je prikazan na slici (28. a), gdje su x_1 i x_2 vjerojatnosti impulsa, uz pretpostavku da je $x_1 \geq x_2$. Koristi se dijelilo koje obavlja operaciju x_2/x_1 te jednakost: $(1 - x_2/x_1) \cdot x_1 = x_1 - x_2$, na način da se dodaju logička NOT i AND vrata, kao što je prikazano na slici (28.b). Efektivno, radi se o sklopu za dijeljenje S31 kojem je dodan sklop za pretvaranje dijeljenja u oduzimanje. Taj sklop nazvali smo S9.

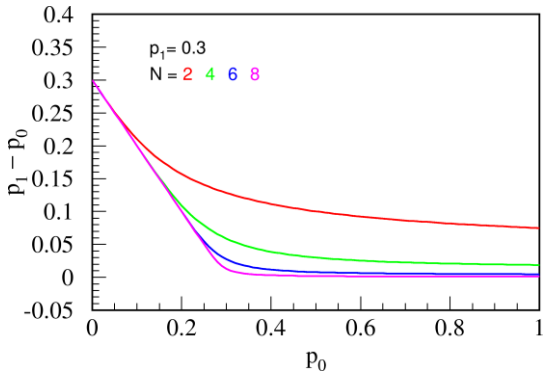


(a) Sklop za oduzimanje S9. (b) Sklop za pretvaranje dijeljenja u oduzimanje.

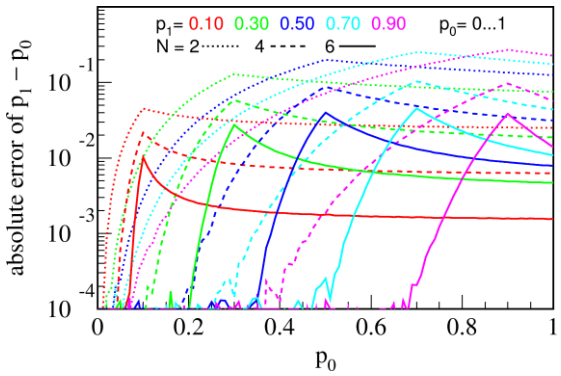
Slika 28: Shema sklopa za oduzimanje S9 s pripadnim sklopom za pretvaranje dijeljenja u oduzimanje.

Ovakav sklop naslijedio je loše strane prethodno opisanog sklopa za dijeljenje, budući da ga koristi kao svoj dio. Glavna mana ovog uređaja jest, dakle, korištenje LFSR generatora pseudo slučajnih bitova budući da on uzrokuje jake korelacije među izlaznim signalima.

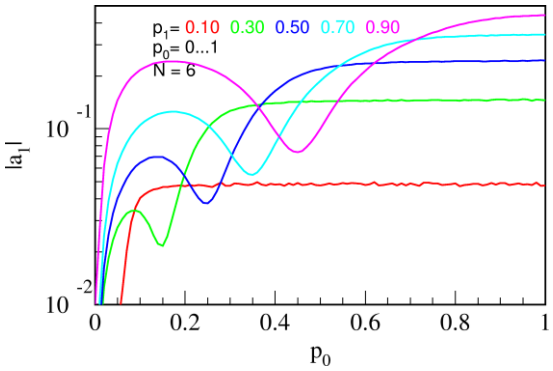
Analiza točnosti ovog sklopa putem simulacija dala je rezultate izlaznih vrijednosti sklopa prikazanih na slici (29.a) te pripadne pogreške i autokorelacijske koeficijente a_1 na slikama (29.b) i (29.c). Vidimo da su točnosti, za različite vrijednosti parametra p_1 i broja bitova N vrlo slične onima za sklop S31 koji se nalazi u srcu ovog oduzimala. Autokorelacijski koeficijent izlaznog niza čini se prevelikim za praktičnu uporabu u daljnjim računima te postiže minimume pri istim uvjetima kao i kod sklopa S31. Vrlo vjerojatno ovakav sklop nije primjenjiv za daljnje račune zbog slabe točnosti, sporog opadanja pogreške s p_0 kada je $p_0 \geq p_1$ te velike autokorelacije.



(a) Izlazne vrijednosti sklopa S9.



(b) Pogreške.

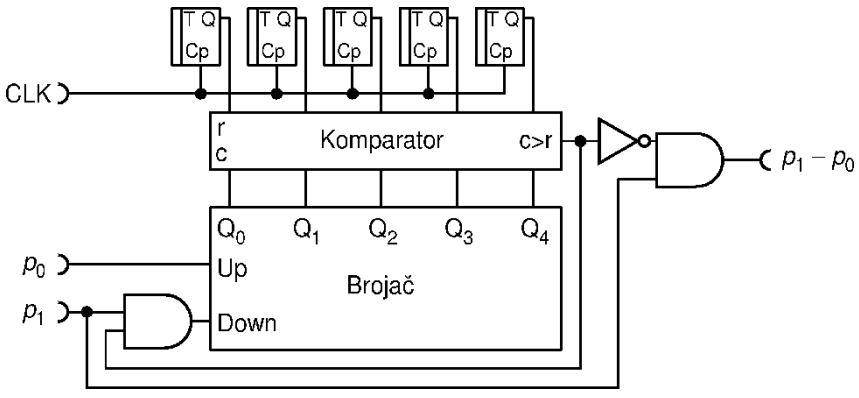


(c) Autokorelacija izlaznog niza.

Slika 29: Simulacija sklopa za oduzimanje S9.

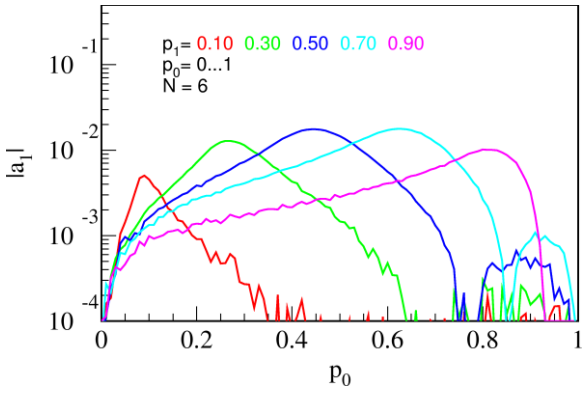
4.4.1 Sklop za oduzimanje 1

Kako bismo popravili bar neke loše karakteristike prethodnog sklopa, odlučili smo ispitati modifikaciju u kojoj je LFSR zamijenjen s odgovarajućim brojem TRFF sklopova koji generiraju nekorelirane slučajne bitove. Rezultirajući sklop za oduzimanje, kojeg smo nazvali S8, prikazan je na slici (30).



Slika 30: Shema sklopa za oduzimanje S8.

Rezultati simulacija vrijednosti funkcije i pogreške za sklop S8 praktično su nepromijenjeni u odnosu na sklop S9, pa ih ovdje ne prikazujemo. Slično kao i u sklopovima za dijeljenje gdje je LFSR zamijenjen s TRFF, jedina primjetna promjena je u nižoj autokorelaciji izlaznog niza sklopa S9, koja je prikazana na slici (31).



Slika 31: Autokorelacija izlaznog niza sklopa za oduzimanje S8.

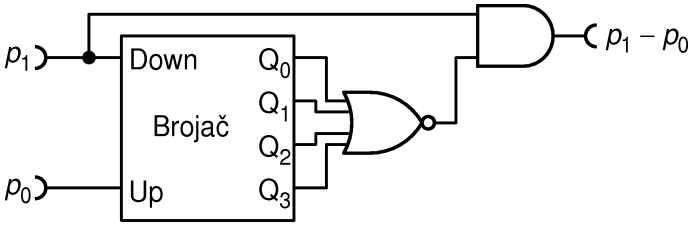
Primjećujemo da je autokorelacija uglavnom ispod 1%, što znači da je drastično poboljšana u odnosu na sklop S9, što je pak vrlo vjerojatno dovoljno za daljnju uporabu u RPC računalu. Problem međutim ostaje slaba točnost.

Do sada prikazani sklopovi za oduzimanje putem dijeljenja sadrže tri operacije (dijeljenje, množenje pomoću logičkih AND vrata i oduzimanje od jedinice pomoću logičkih NOT vrata) te na taj način demonstriraju jednostavno RPC računalo.

4.4.3 Sklop za oduzimanje 2

Sada je već jasno da bi korištenje preciznog dijelila S32 kao osnove za oduzimalo rezultiralo visokom točnošću oduzimala, ali neprihvatljivo visokom autokorelacijom izlaznog niza. Stoga nismo istraživali u tom smjeru, već smo potražili drugo rješenje.

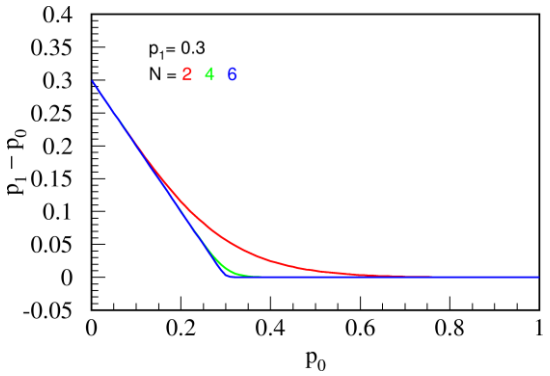
Sklop S1, prikazan na slici (32), nema u stvari nikakve veze s dijeljenjem te u tom smislu predstavlja bitnu inovaciju na području oduzimanja, kao i novost u području sklopovlja za RPC.



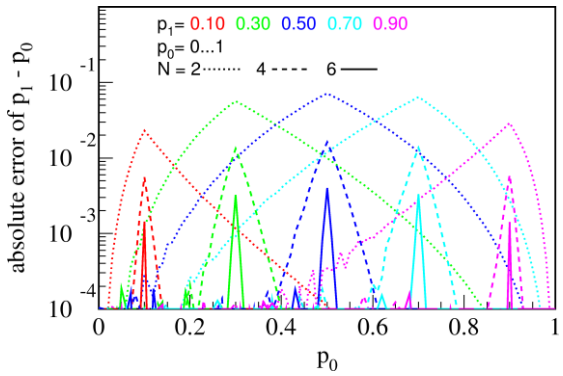
Slika 32: Shema sklopa za oduzimanje S1.

Sklop radi na slijedećem principu. Polazimo od pretpostavke da je $p_1 \geq p_0$. To znači da u ulaznom nizu karakteriziranom vjerojatnošću p_1 imamo veći broj impulsa u jedinici vremena nego u nizu karakteriziranom vjerojatnošću p_0. U tom slučaju, da bismo izračunali $p_1 - p_0$, trebamo iz prvog niza izbaciti onoliko impulsa koliko ih sadržava drugi niz. Uloga brojača je u tome da broji impulse pristigle iz niza p_0 te, ako i dok god je taj broj veći od broja impulsa pristiglih iz niza p_1, ne propušta impulse iz p_1 na izlaz. Na shemi se lako vidi da dok god je stanje brojača veće od nule, impulsi iz p_1 ne mogu proći kroz logička AND vrata. Prema tom načinu rada sklop bi morao raditi egzaktno, ali problem nastaje kada je $p_0 \approx p_1$, pri čemu uslijed statističkih fluktuacija, brojač može nabrojiti do vrlo velike vrijednosti, tako da je dobar kapacitet brojača neophodan za preciznost željene matematičke opracije oduzimanja.

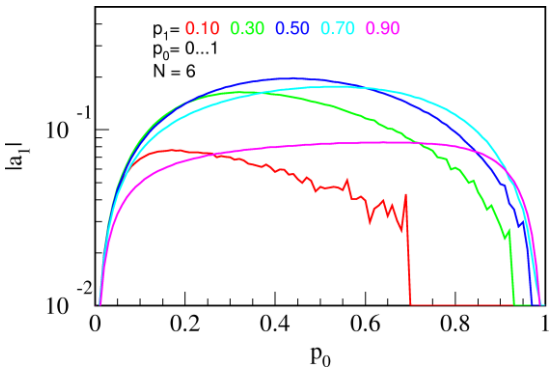
Na slikama (33.a) i (33.b) prikazane su izlazne vrijednosti prethodno opisanog sklopa S9, kao i pripadna odstupanja od teorijskih vrijednosti, dok je na slici (33.c) prikazana autokorelacija izlaznog niza.



(a) Izlazne vrijednosti sklopa S1.



(b) Pogreške.



(c) Autokorelacija izlaznog niza.

Slika 33: Simulacija sklopa za oduzimanje S1.

Promatrajući slike (33.a) i (33.b) vidimo da su pogreške doista male već za N ≥ 4. No, budući da se impulsi iz niza p_1 oduzimaju sukcesivno, to može prouzrokovati velike razmake između susjednih

impulsa i narušiti eksponencijalnu distribuciju, odnosno proizvesti autokorelaciju, kao što je vidljivo na slici (33c). Međutim, autokorelacija je ipak poprilično niska, uglavnom u rasponu od 0.08 do 0.15.

Ovaj sklop je obećavajući zbog njegove jednostavnosti te relativno dobrih polaznih rezultata nasuprot drugih sklopova. Moguće je da bi se raznim modifikacijama mogle dobiti verzije s povećanom točnošću ili s manjim korelacijama (ili oboje) optimalne za pojedine namjene.

5 Rezultati mjerenja

Analiza idejnih sklopova putem simulacija, koje smo predstavili u prethodnom poglavlju, ponudila je rješenja za četiri osnovne matematičke operacije: množenje, zbrajanje, dijeljenje i oduzimanje. Čak štoviše, budući da su izlazi svih sklopova u stvari vjerojatnosti – realni brojevi u intervalu [0, 1] – sklop za oduzimanje može se iskoristiti i kao sklop za usporedbu dva broja, odnosno odgovor na pitanje koji je broj veći ili manji. Naime, ukoliko je $p_0 \geq p_1$, onda je $p_1 - p_0 = 0$, odnosno te dvije tvrdnje su ekvivalentne. Na taj način dobili smo pet funkcija nužnih i dostatnih za gradnju univerzalnog RPC računala (bez memorije).

Od svih simuliranih sklopova, za praktičnu izvedbu odabrali smo najuspjelije ili najinteresantnije sklopove te testirali niz praktičnih sklopova koji pokrivaju sve četiri elementarne matematičke, odnosno binarne operacije i operaciju usporedbe (veće – manje) dva broja:

1. Množenje putem AND vrata
2. Zbrajanje putem OR vrata
3. Zbrajanje putem MUX sklopa
4. Dijeljenje putem sklopova S30 i S32
5. Oduzimanje (i uspoređivanje) putem sklopova S1 i S8.

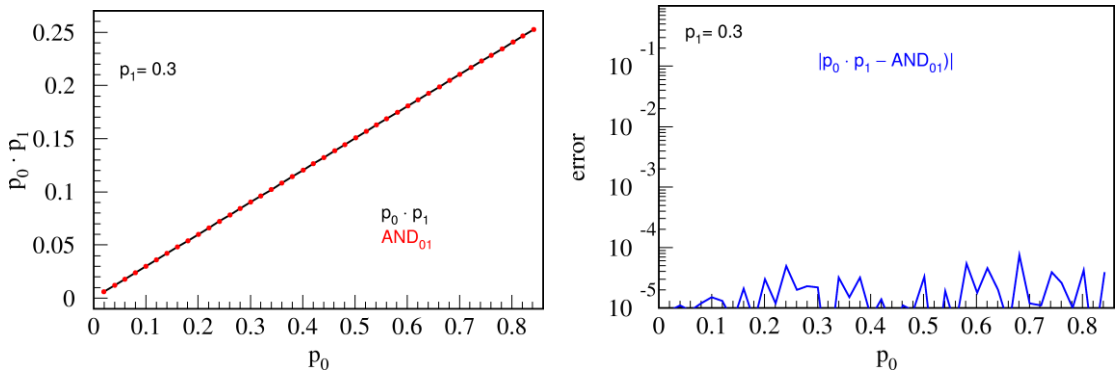
Osim toga, za sklop za dijeljenje S30, koji je ekvivalentan sklopu u članku [10], mjerili smo izlaznu frekvenciju impulsa s izlaza iz komparatora (kako je tamo nacrtano) kako bismo potvrdili da taj izlaz ne daje ni približno točan rezultat dijeljenja, to jest da je shema u članku pogrešna.

Praktična realizacija ovih sklopova u FPGA čipu zahtijevala je vrlo pažljivu prilagodbu idejnih shema, prikazanih u prethodnom poglavlju *Sklopovi za matematičke operacije i njihove simulacije*, na način da se putem nekoliko precizno sinkroniziranih taktnih signala (*clock*) osigura sekvencijalno i pravovremeno pristizanje svih signala na ulaze logičkih sklopova. To je bilo potrebno kako bi se izbjeglo djelomično propuštanje i „sjeckanje" impulsa odnosno pojavu tzv. „glitch"-eva. Prilikom osmišljavanja programa za FPGA čip, provjera signala na pojedinim kontrolnim točkama bila je od vitalnog značenja.

Svaka mjerna točka, odnosno mjerenje vjerojatnosti jedinice na izlazu sklopa za jedan par ulaznih varijabli (p_0, p_1), mjeri se 60 sekundi, neovisno o frekvenciji pulseva koja se mjeri. Za tipičnu mjerenu frekvenciju od 100 kHz ($p = 0.1$), to iznosi $6 \cdot 10^6$ točaka, odnosno relativnu statističku pogrešku od $\sqrt{6 \cdot 10^6} / 6 \cdot 10^6 = 4 \cdot 10^{-4}$. Toliko mala pogreška ne vidi se na našim grafovima stoga ju nismo niti crtali.

5.1 Množenje putem AND vrata

Množenje je operacija koja se može ostvariti jednostavno i s visokom preciznošću. Na slici (34.a) prikazana je simulirana krivulja, koja je nerazlučivo bliska teorijskoj, zajedno sa mjerenim točkama. Da bi se bolje procijenila točnost ovih mjerenja na slici (34.b), prikazana je razlika između mjerenih točaka i očekivanog rezultata, koja je u granicama statističke pogreške.



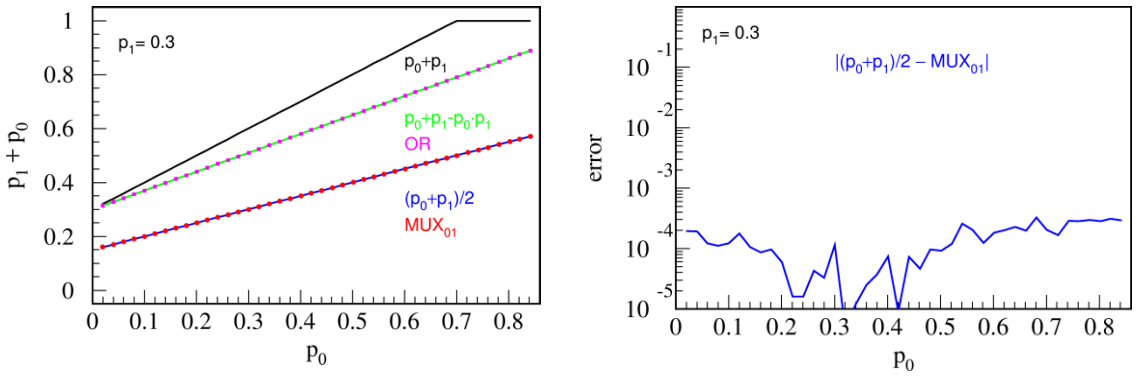
(a) Mjerenja i teorijska krivulja za množenje. (b) Apsolutna pogreška mjerenih točaka.

Slika 34: Rezultati mjerenja sklopa za množenje putem logičkih AND vrata, zajedno s pripadnom apsolutnom pogreškom mjerenih točaka. Na (a) dijelu slike crvenom bojom označena su mjerenja, dok je crnom bojom označena teorijska krivulja. Na (b) dijelu slike je plavom bojom označena apsolutna pogreška mjerenih točaka.

5.2 Zbrajanje putem OR vrata i MUX sklopa

Kada bi postojao sklop za zbrajanje ulaznih vjerojatnosti (takav sklop do sada nije otkriven), on bi morao ući u zasićenje. Točnije, kada zbroj ulaznih varijabli prelazi 1, izlazna vrijednost mogla bi biti najviše 1, budući da je to najveća moguća vjerojatnost pojave impulsa. Primjer zasićenja prikazan je na slici (35.a) crnom krivuljom.

Međutim, kao što je pojašnjeno u prethodnom poglavlju, OR vrata u stvari rade operaciju $p_0 + p_1 - p_0 \cdot p_1$ čiji rezultat ne može premašiti 1. Budući da je to vrlo neobična operacija i da se ne može izravno iskoristiti za zbrajanje, nas ovdje više zanima (polu)zbrajalo s MUX sklopom koje izvršava operaciju $(p_0 + p_1)/2$. Zbog toga pogreška sklopa za zbrajanje putem logičkog OR sklopa ovdje nije prikazana. Mjerene točke zajedno sa pripadnim simuliranim krivuljama sklopova za zbrajanje prikazane su na slici (35.a). Oba sklopa daju rezultate nerazlučivo bliske teorijskim vrijednostima. Da bi se bolje procijenila točnost mjerenja sklopa za zbrajanje putem MUX-a, na slici (35.b) prikazana je razlika između mjerenih točaka i očekivanog rezultata, koja je u granicama statističke pogreške.



(a) Mjerenja i teorijska krivulja za zbrajanje. b) Apsolutna pogreška mjerenih točaka za MUX sklop.

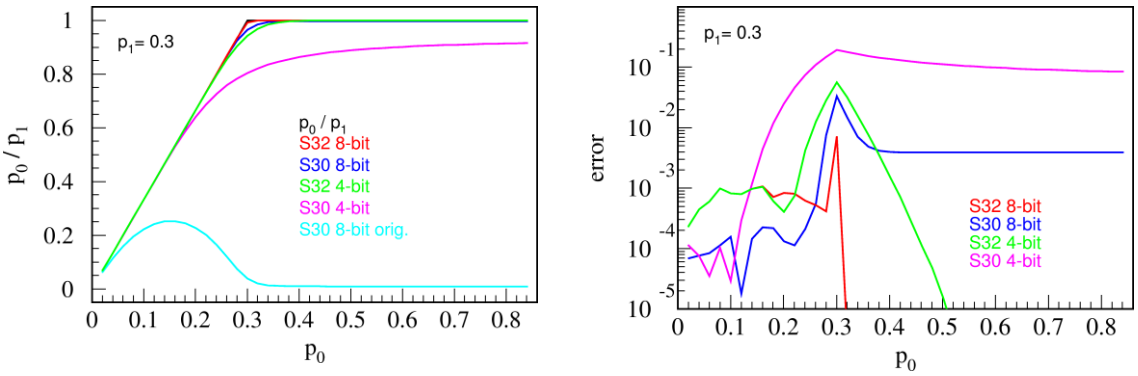
Slika 35: Rezultati mjerenja sklopova za zbrajanje putem logičkih OR vrata i MUX sklopa, zajedno s pripadnom apsolutnom pogreškom mjerenih točaka za MUX sklop. Na (a) dijelu slike crnom bojom prikazana je teorijska krivulja za zbrajanje, zelenom bojom simulirana krivulja sklopa za zbrajanje putem logičkih OR vrata, roza kvadratićima mjerene točke realiziranog sklopa za zbrajanje putem logičkih OR vrata te plavom bojom simulirana krivulja sklopa za zbrajanje pomoću MUX-a, kao i pripadna mjerenja označena crvenim kvadratićima.

5.3 Dijeljenje

Praktično smo izveli, programiranjem u FPGA čipu, četiri sklopa za oduzimanje: S30 s brojačem od N = 4 bita i od N = 8 bitova, te S32 također s brojačem od N = 4 bita i od N = 8 bitova. Ideja je bila vidjeti funkcioniranje sklopova visoke točnosti (N = 8 bitova) i manje točnosti (N = 4 bita). Na slici (36.a) prikazane su krivulje izračuna za sva 4 sklopa. Na istoj slici također je prikazan i rezultat izlaza iz samog komparatora označen kao „S30 8 bit orig.“, kako je originalno opisano u članku [10], dok je na slici (36.b) prikazano odstupanje pojedinih sklopova od točnog rezultata dijeljenja, kako bi se detaljnije utvrdila točnost mjerenih rezultata.

Dobiveni rezultat sklopa za dijeljenje iz članka [10] (S30 8-bit orig.) nema veze s očekivanim rezultatom dijeljenja, odnosno takav sklop je netočan. Sve ostale verzije sklopa za dijeljenje su u skladu sa simulacijama, izuzev sklopa S30 s 4-bitnim brojačem koji je nešto manje precizan.

Apsolutne pogreške su u skladu sa simulacijama osim što kod nižih vrijednosti varijable p_0 su nešto veće pogreške sklopa S32 od sklopa S30, koje zapravo predstavljaju statističku fluktuaciju spomenutu u uvodu ovog poglavlja, koja je veća od one u simulacijama jer je simuliran znatno veći broj impulsa nego što se moglo dobiti mjerenjima. No, simulacija i mjerenja se poklapaju u granicama statističke pogreške.



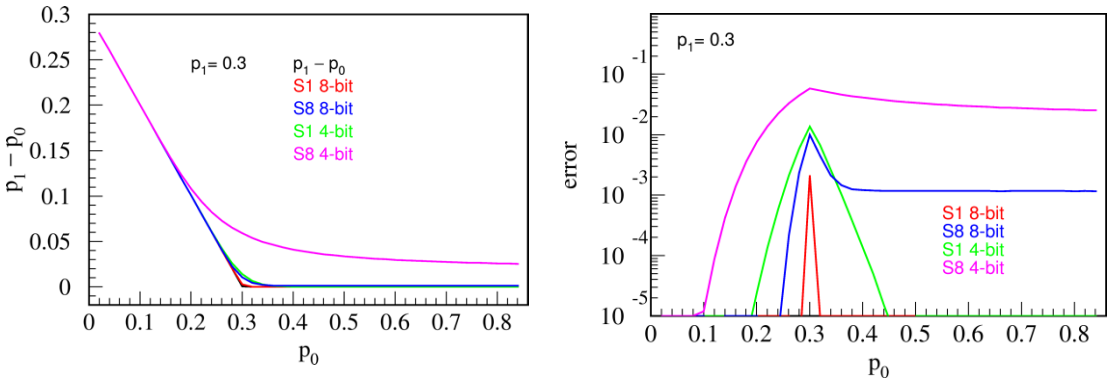
(a) Mjerene krivulje i teorijska krivulja za dijeljenje. (b) Apsolutna razlika.

Slika 36: Rezultati mjerenja sklopova za dijeljenje S30 i S32 za brojača kapaciteta N = 4 bita i N = 8 bitova, zajedno sa pripadnim apsolutnim razlikama između točnih i mjerenih rezultata dijeljenja, kao i rezultati mjerenja sklopa za dijeljenje iz članka [10]. Na (a) dijelu slike crnom bojom predstavljena je teorijska krivulja operacije dijeljenja, dok su crvenom, tamno plavom, zelenom i roza bojom označene krivulje sklopova za dijeljenje. Sklop iz članka [10] označen je svjetlo plavom bojom. Na (b) dijelu slike istim bojama kao i na (a) dijelu slike označene su krivulje pogrešaka za sklopove S30 i S32.

5.4 Oduzimanje i usporedba brojeva

Eksperimentalno smo realizirali i testirali ukupno četiri verzije sklopova za oduzimanje: S1 s brojačem od N = 4 bita i od N = 8 bitova, te S8 također s brojačem od N = 4 bita i od N = 8 bitova. Ideja je bila vidjeti funkcioniranje sklopova visoke točnosti (8 bita) i manje točnosti (4 bita). Na slici (37.a) prikazane su, raznim bojama, krivulje vrijednosti izlazne vjerojatnosti impulsa za sva 4 sklopa. Za bolju procjenu točnosti ovih sklopova, na slici (37.b) prikazana su odstupanja mjerenih vjerojatnosti impulsa na izlazu sklopova od teorijskih vrijednosti.

Kao i kod dijeljenja, jedino veće odstupanje događa se kod sklopa S8 s 4-bitnim brojačem, što je lako shvatiti budući da taj sklop koristi sklop za dijeljenje S30. Također, odstupanja od teorijskih vrijednosti 8-bitnih verzija su najviše kad su ulazne varijable približno jednake, ispod 1%, dok su u ostalim slučajevima vrlo male.



(a) Mjerene krivulje i teorijska krivulja za oduzimanje. (b) Apsolutna razlika.

Slika 37: Rezultati mjerenja sklopova za oduzimanje S1 i S8 za brojače kapaciteta N = 4 bita i N = 8 bitova, zajedno sa pripadnim apsolutnim razlikama između točnih i mjerenih rezultata oduzimanja. Na (a) dijelu slike crnom bojom predstavljena je teorijska krivulja operacije oduzimanja, dok su crvenom, tamno plavom, zelenom i roza bojom označene krivulje sklopova za dijeljenje. Na (b) dijelu slike istim bojama kao i na (a) dijelu slike označene su krivulje pogrešaka za sklopove S1 i S8.

6 Zaključak

U ovome radu proučavali smo veći broj elektroničkih sklopova čija je namjena primjena u računalu koje radi na principu vremenski slučajnih logičkih električnih impulsa (RPC). Ideja za takvo računalo inspirirana je time što se u mozgu i živčanom sustavu živih bića također propagiraju električki impulsi veoma nalik logičkim impulsima.

Prikazano istraživanje obuhvaća detaljniju karakterizaciju pojedinih sklopova koji izvršavaju elementarne matematičke ili logičke operacije. Neke smo sklopove ispitivali detaljnije nego li je to do sada opisano u literaturi, a kvantitativno ispitivanje korelacija je, prema našim saznanjima, ovdje napravljeno po prvi puta. Konkretno, za množenje smo koristili AND sklop, za zbrajanje OR i MUX sklopove, a za dijeljenje, oduzimanje i uspoređivanje brojeva koristili smo složene sklopove s brojačima, komparatorima i izvorima slučajnosti. Posebno, sklop za oduzimanje S8 u sebi sadrži kombinaciju nekoliko sklopova: sklop za dijeljenje, sklop za množenje i sklop za oduzimanje od jedinice (NOT) te u tom smislu već demonstrira jednostavno RPC računalo. Nakon ispitivanja rada sklopova putem simulacija napravljenih u programu *Python,* izabrali smo jedan skup sklopova koji je pokazao najbolje karakteristike, a koji je dostatan za gradnju univerzalnog računala te potom realizirali u rekonfigurabilnom FPGA čipu (Intel Cyclone IV) i eksperimentalno ispitali. Za izvršenje eksperimenta bilo je potrebno izraditi nekoliko novih uređaja (hardver) za generiranje i prikupljanje podataka (višekanalni frekvenciometar, generator slučajnih impulsa), uz korištenje standardnih laboratorijskih uređaja poput generatora valnih oblika i osciloskopa.

Najvažniji rezultati ovog rada su eksperimentalno ostvarenje sklopova u RPC-u i ispitivanja koji pokazuju da je, u danas raspoloživoj FPGA tehnici FPGA čipova, moguće izraditi precizne sklopove za RPC računalo uz znatno manji utrošak resursa u odnosu na ostvarenje istih operacija na istom čipu u uobičajenom digitalnom načinu rada. Ispitani sklopovi odlično prate teorijska očekivanja, kao što je i vidljivo na slikama (34), (35), (36) i (37) iz prethodnog poglavlja *Rezultati mjerenja*. Postignuta preciznost računa od $10^{-6} - 10^{-3}$ nije konkurencija digitalnim računalima koja mogu imati proizvoljnu točnost, ali je sasvim dostatna za rješavanje vrsta problema, poput obrade video zapisa, za koje se očekuje da bi ih RPC mogao riješiti znatno brže i točnije od digitalnih računala. Također, sklopovi simulirani putem računalnih programa (softvera) koje smo napisali posebno za ovo istraživanje koristit će nam i u nastavku istraživanja.

Iako je ovdje demonstriran set operacija dostatan za univerzalno računalo, ostaje otvorenim pitanje da li se veći broj ovih sklopova može kombinirati, a da se pri tome sačuvaju dobra preciznost i ostala svojstva impulsnog računanja, s obzirom na to da neki od danas poznatih sklopova proizvode snažno autokorelirane nizove impulsa. To, kao i pitanje pojednostavljenja, poboljšanja i osmišljavanja novih sklopova, kao i njihova daljnja primjena, teme su budućih znanstvenih istraživanja u ovom mladom području.

7 Zahvale

Ovim putem zahvaljujem se mentoru dr. sc. Mariu Stipčeviću na pomoći, savjetima i strpljenju te nadasve na pojašnjavanju nejasnoća i svim komentarima i prijedlozima tijekom cijele realizacije ovog rada. Zahvaljujem se i prof. dr. sc. Hrvoju Buljanu na njegovom doprinosu i savjetima. Također, želim se zahvaliti i svim zaposlenicima centra na ugodnoj radnoj atmosferi, kao i svim članovima obitelji i prijateljima na potpori i ohrabrivanju prilikom pisanja samoga rada.

Literatura

[1] D. Subašić, „Einstein i kvant svjetlosti“, Diplomski rad, Prirodoslovno matematički fakultet Sveučilišta u Zagrebu (2015).

[2] M. Stipčević, “Active quenching circuit for single-photon detection with Geiger mode avalanche photodiodes”, Appl. Opt. 48, 1705-1714 (2009).

[3] E. M. Petriu, “Applications of Random-Pulse Machine Concept to Neural Network Design”, IEEE Transactions on Instrumentation and Mesurement, vol. 45, no. 2, April 1996.

[4] A. Alaghi, C. Li, J. Hayes, “Stochastic circuits for real-time image-processing applications”, Design Automation Conference (DAC) 2013 50th ACM / EDAC / IEEE, pp. 1-6, May 2013.

[5] A. Alaghi and J. P. Hayes, “Computing with Randomness”, IEEE Spectrum, vol.55, no.3, pp. 46-51, Mar. 2018.

[6] A. Alaghi, W. Qian, J.P. Hayes, “The Promise and Challenge of Stochastic Computing”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 37, no. 8, pp. 1515-1531, Aug 2018.

[7] J. von Neumann, “Probabilistic logics and synthesis of reliable organisms from unreliable components”, in C. Shannon and J. McCarthy (editors), Automata Studies, pages 43--98, Princeton University Press. 1956.

[8] S. T. Ribeiro, “Random-pulse machines”, IEEE Trans. Electron. Comput., vol. EC-16, pp. 261-276, June 1967.

[9] B. R. Gaines, “Stochastic Computing Systems”, Boston, MA: Springer US, pp. 37-172. 1969.

[10] Y Liu, KK Parhi. "Computing polynomials using unipolar stochastic logic", ACM Journal on Emerging Technologies in Computing Systems (JETC) 13 (3), 42, 2017.

[11] M. Stipčević, “Quantum random flip-flop and its applications in random frequency synthesis and true random number generation”, Rev. Sci. Instrum. 87, 035113 (2016).

[12] I. Pavlić, „Statistička teorija i primjena“, 3. izdanje, Tehnička Knjiga, Zagreb, 1985.

[13] Stipčević M., Skenderović H., Gracin D., “Characterization of a novel avalanche photodiode for single photon detection in VIS-NIR range”, Opt. Express 18,17448-17459 (2010).

[14] M. Stipčević, J. Bowers, “Spatio-temporal optical random number generator”, Opt. Express 23, 11619-11631 (2015).

[15] S. Paunović, „Digitalna elektronika“, Školska knjiga, 2. izdanje, Zagreb, 1999.

[16] J. Serrano, “Introduction to FPGA design”, CAS - CERN Accelerator School Course on Digital Signal Processing, pp. 231-247, 2007.

[17] L. Karlović, „Ugradnja instrukcije množenja na procesoru P16 s minimalnim skupom instrukcija“, završni rad br. 174, FER, Zagreb, 2008.

[18] V. T. Lee, A. Alaghi, L. Ceze, “Correlation Manipulating Circuits for Stochastic Computing”, Proc. Design Autom. Test Europe Conf. Exhibition (DATE), pp. 1417-1422, Mar. 2018.

[19] Veerappan, C., & Charbon, E., “A Low Dark Count p-i-n Diode Based SPAD in CMOS Technology”, IEEE Transactions on Electron Devices, 63, 65-71 (2015).

[20] URL: http://www.fourmilab.ch/random/

Sažetak

Ime i prezime autora: Mateja Batelić

Naslov rada: Impulsno neuronsko računanje

Proučavani su simulirani i praktični sklopovi za računanje osnovnih matematičkih operacija, u paradigmi impulsnog neuromorfnog računanja, sa svrhom postizanja što veće točnosti. Proučavana je i autokorelacija izlaznih nizova slučajnih impulsa, točnije autokorelacijski koeficijent prvog reda, koji treba biti što manji tj. težiti prema nuli, budući da dobiveni niz možemo shvatiti i kao ulazni niz za sljedeće računske operacije, dok autokorelacija smanjuje preciznost računa. Kao realan izvor slučajnih događaja, koriste se detektori fotona obasjani LED diodama. Pomoću električnih signala dobivenih iz dva detektora fotona moguće je implementirati binarne matematičke operacije. U ovom radu iznijeli smo rezultate već ranije poznatih sklopova za zbrajanje i za množenje te unaprijeđenih sklopova za oduzimanje i dijeljenje. Uočeno je da sklopovi za zbrajanje i množenje daju dovoljno točne rezultate sa minimalnom uporabom logičkih vrata prilikom izrade istih, dok pojedine verzije sklopova za oduzimanje i dijeljenje daju vrlo točne rezultate već prilikom korištenja 4-bitnih brojača. Također smo demonstrirali i mogućnost usporedbe brojeva, što zajedno s četiri osnovne operacije, u principu, omogućuje gradnju univerzalnog računala.

Ključne riječi: niz slučajnih impulsa (RPT), autokorelacijski koeficijent, detektori fotona

Summary

First and last name of the author: Mateja Batelić

Title: Neuronal pulse computing

Simulated and practical circuits for calculating elementary arithmetic operations in the pulsed neuromorphic computing paradigm have been studied for the purpose of achieving accuracy as high as possible. The autocorrelation of the output sequences of random impulses, namely the first order autocorrelation coefficient, has been studied. It should be as small as possible, since an output sequence can be used as the input sequence for subsequent computational operations, while autocorrelation decreases the precision of the calculation. As a realistic source of random events, photon detectors illuminated with LED diodes were used. By the use of electrical signals obtained from two photon detectors, it is possible to implement binary mathematical operations. In this paper, we have outlined the results of the already well-known addition and multiplication circuits, as well as the improved subtraction and division circuits. It was noted that the addition and multiplication circuits give enough accuracy already with the minimum use of logic gates, while some versions of subtracting and division circuits give very accurate results when using 4-bit counters. We also demonstrated the ability to compare two numbers, which together with the four elementary operations, in principle, allows the construction of a universal computer.

Keywords: random pulse train (RPT), autocorrelation coefficient, photon detectors

Dodatak A

Kao primjer simulacije izabran je sklop za oduzimanje S1 jer je pripadna simulacija dovoljno jednostavna i razumljiva za prezentaciju. Na slici (38) prikazan je glavni dio simulacije za računanje izlaznog niza prilikom samo jedne kombinacije parametara p_0 i p_1. Simulacija radi na način da se prvo na slučajan način odaberu ulazni impulsi označeni kao *r1* i *r2* pomoću funkcije *random.random()* iz modula *Radnom,* koja generira decimalan slučajan broj u intervalu [0, 1]. Ako su generirani brojevi manji od zadanih ulaznih vrijednosti parametara p_0 i p_1, označenih sa *p1* i *p2*, znači da je ulazna vrijednost 1. U suprotnom je ulazna vrijednost 0. Za svaku vrijednost *r2* < *p2* se vrijednost brojača povećava za 1, dok se za svaku vrijednost *r1* < *p1* vrijednost brojača smanjuje za 1. Konačno, ovisno o tome da li je brojač jednak nuli ili različit od nje, vrijednost varijable *propusteni* ostaje nepromijenjena ili se povećava za 1. Time se dobiva ukupan broj propuštenih impulsa na izlazu iz sklopa pa se konačan rezultat operacije oduzimanja dobiva dijeljenjem broja izlaznih jedinica sa ukupnim brojem vremenskih

```
for i in range(broj_dogadaja):
        r1 = random.random()
        r2 = random.random()
        #1
        if r2<p2:
                brojac += 1
        #2
        if r1<p1:
                brojac -= 1
        if brojac == 0 and r1<p1:
                propusteni += 1
                if len(byte) == 8:
                        number = 0
                        for l in range(8):
                                number += byte[l](2*l)
                        entry = struct.pack('<B', number)
                        f.write(entry)
                        byte = []
                        byte.append(1)
                else:
                        byte.append(1)
        if brojac != 0 or r1>p1:
                if len(byte) == 8:
                        number = 0
                        for l in range(8):
                                number += byte[l](2*l)
                        entry = struct.pack('<B', number)
                        f.write(entry)
                        byte = []
                        byte.append(0)
                else:
                        byte.append(0)

number = 0
for l in range(len(byte)):
        number += byte[l](2*l)
entry = struct.pack('<B', number)
f.write(entry)
lista_brojac.append(lista_brojac_j)
broj_propustenih_dogadaja.append(propusteni)
```

Slika 38: Glavni dio simulacije sklopa za oduzimanje S1 napravljene u programskom jeziku *Python*

intervala koji je nazvan *broj_dogadaja*. Na taj način dobiven je slučajan niz izlaznih impulsa koji se upisuje u datoteku *f* radi primjene programa BENT za računanje autokorelacijskog koeficijenta a_1. Konačno, grafovi prezentirani u poglavlju *Sklopovi za matematičke operacije i njihove simulacije* dobiveni su na način da se napravi petlja koja će određeni broj puta izračunati željenu matematičku operaciju te automatski rezultate pohraniti u posebnu datoteku iz koje će podaci kasnije biti ispisani na ekran pomoću modula *Matplotlib*.