Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Mark Rolih

Analiza možnosti realizacije logičnih reverzibilnih vrat v trostanjskem kvantnem celičnem avtomatu

DIPLOMSKA NALOGA
NA UNIVERZITETNEM ŠTUDIJU

doc. dr. Iztok Lebar Bajec
MENTOR

prof. dr. Miha Mraz
SOMENTOR

Ljubljana, 2013

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01932/2013
Datum: 19.06.2013

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **MARK ROLIH**

Naslov: **ANALIZA MOŽNOSTI REALIZACIJE LOGIČNIH REVERZIBILNIH VRAT V TROSTANJSKEM KVANTNEM CELIČNEM AVTOMATU**

ANALYSIS OF POSSIBLE LOGICAL REVERSIBLE GATE REALIZATION IN TERNARY QUANTUM-DOT CELLULAR AUTOMATA

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Kandidat naj v svojem delu razišče možnosti realizacije trovrednostnih reverzibilnih logičnih vrat v strukturi kvantnega celičnega avtomata. Pri svoji raziskavi naj se usmeri na logično realizacijo Toffolijevih vrat. Rezultate postavitve naj ustrezno analizira s pomočjo razvojnega in modelirnega okolja qdCAD.

Mentor:

doc. dr. Iztok Lebar Bajec

Somentor:

prof. dr. Miha Mraz

Dekan:

prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko
2
UNIVERZA V LJUBLJANI

IZJAVA O AVTORSTVU DIPLOMSKEGA DELA

Spodaj podpisani izjavljam, da sem avtor dela, da slednje ne vsebuje materiala, ki bi ga kdorkoli predhodno že objavil ali oddal v obravnavo za pridobitev naziva na univerzi ali drugem visokošolskem zavodu, razen v primerih kjer so navedeni viri.
S svojim podpisom zagotavljam, da:

- sem delo izdelal samostojno pod mentorstvom doc. dr. Iztoka Lebarja Bajca in somentorstvom prof. dr. Mihe Mraza,
- so elektronska oblika dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko in
- soglašam z javno objavo elektronske oblike dela v zbirki "Dela FRI".

— Mark Rolih, Ljubljana, junij 2013.

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Mark Rolih

Analiza možnosti realizacije logičnih reverzibilnih vrat v trostanjskem kvantnem celičnem avtomatu

POVZETEK

V diplomski nalogi je predstavljena možnost realizacije logičnih vrat, ki ustrezajo pogoju reverzibilnosti in so realizirana v trostanjskem QCA-ju (angl. *quantum-dot cellular automata*) ter pot razvoja logičnih vrat s programskim orodjem.

V prvem delu naloge se osredotočimo na pojem reverzibilnosti in pomen le te v fizičnem in logičnem svetu. Predstavljena so nekatera osnovna logična vrata, ki zadostijo pogojem reverzibilnosti, pozornost pa pritegnejo tudi Toffolijeva in Fredkinova logična vrata. Glede na prednosti posameznih logičnih vrat se odločimo katera bomo realizirali.

V drugem delu je predstavljena trojiška številska osnova in njene prednosti pred dvojiško. Predstavljen je tudi QCA, njegovo delovanje, osnovne strukture in prehod na trostanjski QCA. Z osnovnimi strukturami lahko nato realiziramo izbrana reverzibilna logična vrata iz prvega dela.

V tretjem delu predstavimo realizacijo Toffolijevih vrat v trostanjskem QCA-ju. Temu sledi analiza rezultatov realizacije.

Kljucne besede: kvantni celični avtomat, reverzibilnost, trojiška logika, Toffolijeva vrata

University of Ljubljana
Faculty of Computer and Information Science

Mark Rolih

Analysis of possible logical reversible gate realization in ternary quantum-dot cellular automata

ABSTRACT

In this thesis is presented possibility of logical gate realization that satisfy reversibility condition and logical gate are made in ternary QCA (quantum-dot cellular automata). There is also covered development plan of logical gate with programming tool.

The first part of thesis emphasizes reversibility concept and also its meaning in physical and logical world. Then we look at basic reversible gates and we closely examine Toffoli and Fredkin gates. Based on properties of logical gates we select candidate for possible realization.

The second part describes advantages of ternary numeral system versus binary one. Next we examine how does QCA works and which structures can be implemented in it. There is also shown QCA to ternary QCA transformation. Base ternary QCA's structures are then good for realization of previously selected candidate.

In the third part we talk about Toffoli gate realization in ternary QCA. At the end there are analysis results.

Key words: quantum-dot cellular automata, reversibility, ternary logic, Toffoli gate

ZAHVALA

Zahvaljujem se mentorju doc. dr. Iztoku Lebarju Bajcu in somentorju prof. dr. Mihi Mrazu za vodstvo, nasvete, dosegljivost, potrpežljivost in usmerjanje pri izdelavi diplomske naloge. Prav tako se zahvaljujem družini za vso podporo v času študija, Gabi in Gašperju za pomoč pri odpravljanju pravopisnih napak diplomske naloge.

— Mark Rolih, Ljubljana, junij 2013.

KAZALO

1 Uvod

Kvantna tehnologija postaja vse bolj pomembna in poznana v današnjem času kot alternativna platforma s katero bi lahko realizirali hitrejše računalnike. QCA (kvantni celični avtomat, angl. *Quantum-dot Cellular Automata*) je posebno zanimiv kot alternativna procesna platforma, saj je delo na njem podobno igri v kateri skupaj zlagamo kocke, te pa tvorijo različne raznobarvne vzorce. V smislu QCA-ja so kocke - kvantne celice, vzorci - logična vrata, barve pa urine faze. Da je podobnost s kockami še večja, poskrbi zmožnost grajenja vezij v treh dimenzijah. Na ta način dobimo vezja s katerimi realiziramo računsko logiko.

Odločili smo se za implementacijo *reverzibilne strukture*, kar pomeni, da bi lahko realizirali računalnik, ki je zmogljivejši, kot so današnji. Zaradi reverzibilne lastnosti, bi računalnik lahko deloval naprej in nazaj, ter pri tem porabil zelo malo energije, omogočal nadaljnji razvoj cenejših in vedno zmogljivejših računalnikov.

Pri vsem tem "igranju" se poučimo o splošnem delovanju logičnih funkcij in dobimo vpogled v večje strukture, ki omogočajo delovanje računalnika. Igra postane programiranje. Predstavimo tudi razlike med dvojiško, trojiško in desetiško številsko predstavitvijo

ter njihovo ustreznost v dani situaciji.

Ker se osredotočamo na delo s *trostanjskim* QCA-jem, imamo veliko možnosti, da odkrijemo nove strukture, saj so raziskave v glavnem potekale na dvostanjskem QCA-ju. V tem konceptu je bila obravnavana tudi reverzibilnost. Trostanjski sistemi in reverzibilnost so bili v preteklosti že v uporabi tudi pri delujočih računalnikih, tako da uporabljamo že znane pojme. Naše delo je torej raziskovanje.

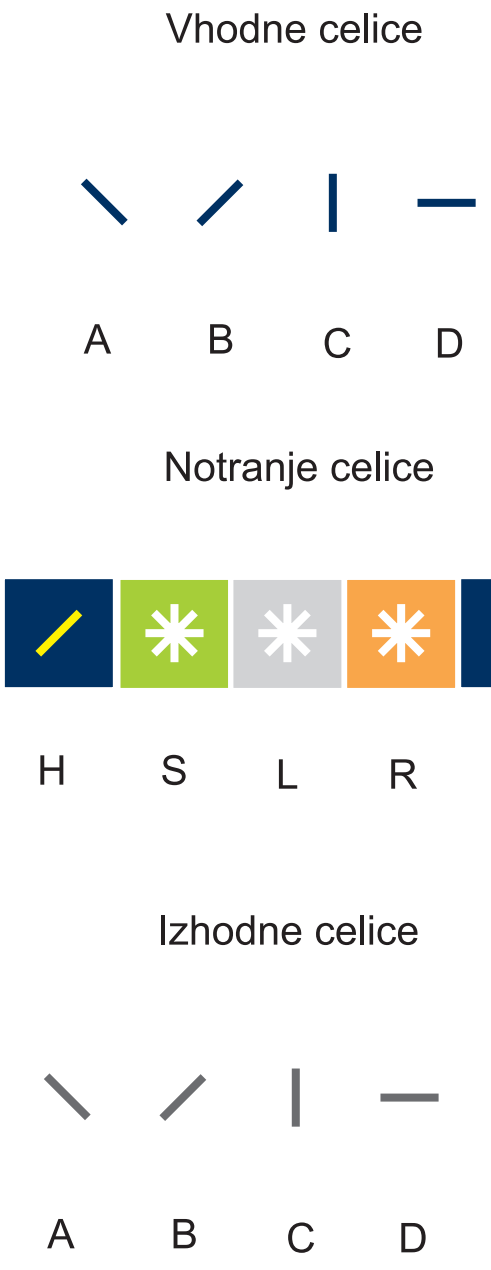
Zakaj govorimo o reverzibilnosti? Zato ker se razvoj današnjih računalnikov približuje pomembni meji, ki je predstavljena z minimalno energijo, potrebno da lahko logične strukture še naprej opravljajo svoje delo in ta znaša $3 * 10^{-21} J$ [1]. Če se bodo osnovne komponente računalnika še naprej manjšale, bomo mejo kmalu dosegli in razvoj zmogljivosti računalnikov se bo upočasnil.

Drugo poglavje se posveča reverzibilnosti. Razloženo je katere vrste reverzibilnosti ločimo in zakaj avtomobil, ki pri vožnji porabi gorivo, ne more pri vzvratni vožni dobiti goriva nazaj. Raziskujemo tudi funkcije, ki so najbolj uporabljane in reverzibilne. S tem si ustvarimo znanje, s katerim med več logičnimi vrati za realizacijo izberemo tista, ki imajo najbolj uporabne lastnosti.

V tretjem poglavju je predstavljena trojiška osnova, klasični QCA in njegovo delovanje. Nato naredimo prehod na razširjenega (trostanjskega) QCA-ja, ki izkoristi prednosti trojiške osnove.

Četrto poglavje govori o poteku dela, o izboru logičnih vrat, njihovi realizaciji in poteku razvoja od funkcije do reverzibilnih trostanjskih logičnih vrat. Torej uporabimo, kar smo se v prejšnjih dveh poglavjih naučili in naredimo delujočo strukturo. Struktura je zgrajena v posebnem programskem orodju, ki omogoča natančno analizo delovanja. Na koncu še pregledamo dobljene rezultate.

Za predstavitev delovanja QCA-ja se uporablja notacija predstavljena na sliki 1.1.



Slika 1.1 Vhodne celice v QCA-ju so vedno prozorne, stanja A, B, C in D obarvana temno modro in vrednost celic se ne spreminja med samim delovanjem QCA-ja. Notranje celice so obarvane s štirimi različnimi barvami, ki so na sliki označene z zaporedjem H, S, L, R, ki je tudi smer prenosa stanj prek celic. Lahko so v vseh stanjih kot vhodne celice in med delovanjem QCA-ja spreminjajo svoja stanja. Aktivne so, ko so modro obarvane in je stanje v njih rumene barve, če pa je narisan bel krogec ali zvezdica, so neaktivne (njihovo stanje ni čisto določeno). Izhodne celice so prozorne, stanja označuje temno siva barva. Lahko se pojavijo v vseh stanjih kot notranje celice in se med delovanjem spreminjajo [5].

2 Reverzibilnost v dvojiških in trojiških sistemih

2.1 Fizična reverzibilnost

Reverzibilnost se deli na fizično in logično reverzibilnost. Fizična reverzibilnost je omejena s fizikalnimi zakonitostmi realnega sveta, logična reverzibilnost pa je omejena s pravili matematičnega sveta, ki veljajo za preslikovalne funkcije. Spoznajmo najprej prednosti fizične reverzibilnosti.

Večina vsakdanjih naprav, ki so v uporabi, deluje enosmerno (nepovratno, angl. *irreversible*). To pomeni, da pretvarjajo določeno količino (snov, material, obliko) iz enega stanja v drugo. Takšne naprave pa ne zmorejo narediti obratnega, pretvoriti količine nazaj v prvotno stanje, v katerem je bila količina pred uporabo naprave. Primeri so: avtomobil, ki porabi določeno količino goriva na neki poti; kuhinjska peč, ki speče meso; zamrzovalnik, ki porabi električno energijo za delovanje ... Takšne naprave ne morejo spremeniti porabljene količine nazaj v prvotno stanje. Če bi bilo to mogoče, bi lahko povrnili porabljeno gorivo, shranili surovo meso, čeprav je bilo predhodno spečeno in povrnili elektriko, ki jo je zamrzovalnik potreboval, da je zamrznil hrano [1].

Tudi naprave, ki trdijo, da so reverzibilne, tega ne počnejo v smislu prave fizične

reverzibilnosti. Na primer vrtalnik ima oznake za nastavitev delovanja naprej in nazaj (angl. *forward, reverse*). S pomočjo funkcije nazaj ne moremo priti do prvotnega stanja (ne moremo odvrtati luknje, ko smo jo enkrat zvrtali). Takšne naprave so torej po delovanju enosmerne [1]. Med enosmerne naprave spadajo tudi računalniki. Ko se program izvaja, ga je nemogoče ustaviti in ga pripraviti do izvršitve predhodno sprejetih ukazov v obratnem vrstnem redu, da iz izračunanega odgovora pridemo nazaj do prvotno zastavljenega vprašanja (angl. *uncompute*) [1]. Da bi bilo to mogoče, bi morali spremeniti še delovanje nekaterih ukazov (npr. nasprotna operacija seštevanja – odštevanja, množenja – deljenja, itd.) in napisati programe, ki bi brez izgube informacije delovali naprej in nazaj. Za trenutno generacijo računalnikov takšna reverzibilna logika ni mogoča [1].

Procesorji pri delovanju za izračun odgovorov na zastavljena vprašanja oddajajo energijo v obliki toplote. Toplota omejuje zmogljivost računalnikov, saj je ni mogoče enostavno odvajati. Pri razvoju vedno hitrejših vezij so posamezne komponente vedno tesneje skupaj, saj se tudi tehnologija izdelave čipov manjša. Gordon Moore je predvidel, da se zmogljivost integriranega vezja zvišuje z eksponentom po stroškovni enoti za faktor 2, vsakih 18 mesecev. Če želimo ohraniti takšen razvoj računalnikov, moramo izgubljeno energijo bolje izkoristiti, saj postaja z večanjem zmogljivosti tudi odvajanje toplote vse težje [6].

V primeru delovanja avtomobila, vrtalnika, zamrzovalnika, se reverzibilnost ne obnese zaradi fizikalne omejitve. Ne moremo poloviti vseh atomov onesnaženosti, ki pridejo iz avtomobila med porabo goriva in narediti avtomobilov s popolnim izkoristkom energije, saj vedno prihaja do izgub. Nasprotno v primeru delovanja računalnikov takšne omejitve ni. V preteklosti so bili narejeni prototipi reverzibilnih računalnikov, vendar niso pustili velikega vtisa na javno mnenje in so tako ostali pozabljeni. To se sedaj spreminja, saj reverzibilno računanje omogoča manjšo porabo električne energije in enako hiter nadaljnji razvoj zmogljivosti računalnikov. Vsak računalnik, izdelan na kvantni tehnologiji, bo omogočal reverzibilno delovanje [1].

Reverzibilno računanje pomeni računati z uporabo fizičnega mehanizma, ki je termodinamično kot tudi logično reverzibilen. Torej je to sistem, ki reciklira lastno energijo in oddaja zelo malo odvečne energije (*adiabatni sistem*) [6]. Njegovi programi pa omogočajo tudi izračun vprašanja iz odgovora, ki je bil rezultat vprašanja. Logična reverzibilnost v računanju omogoča, da se informacije o računskih stanjih ne izgubljajo. Na ta način lahko restavriramo predhodno stanje z računanjem nazaj (od-računanjem rezultatov,

angl. *uncompute answer*). Ampak takšno računanje je lahko vzpostavljeno le, ko je najprej zagotovljena fizična reverzibilnost (nobena enota energije se ne zavrže). Popolna fizična reverzibilnost je praktično nedosegljiva [6].

Večina energije se izgubi, ker niti izolatorji niti prevodniki niso idealni. V manjši tehnologiji, kot so vezja narejena, bolj očiten je ta problem. Za procesorje drži, da s hitrejšim delovanjem dobimo večjo porabo in večjo izgubo energije [1]. V preteklosti (1985-2005) se je učinkovitost mikroprocesorjev povečala za faktor 1000. To pomeni tudi veliko povečanje porabe energije. Ustvarjalci vezij bi zmogli tudi tisočkrat večjo pridobitev v zmogljivosti delovanja računalnika, vendar jih omejuje gostota porabe energije (angl. *power density*). Vezje, ki porabi na kilovate energije, praktično ni uporabno [1]. Sodobni pristopi, kot so: zmanjševanje izgube energije z boljšimi prevodniki, pomanjševanje vezij, zmanjševanje vztrajnosti v elektronskih vezjih (informacija je bolj učinkovito predstavljena), omogočajo napredek še kakšni dve desetletji. Potem bo potreben prehod na alternativne sisteme, ki bodo reverzibilni [1].

Izboljševanje učinkovitosti bi lahko zmanjšalo porabo energije, vendar pridemo do neke meje pod katero ne moremo zaradi omejitev trenutne tehnologije in zaradi fizikalnih omejitev. Potrebovali bi napravo, ki bi ob zagonu porabila nekaj energije in nato delovala dokler je ne zmoti nek zunanji vpliv [1].

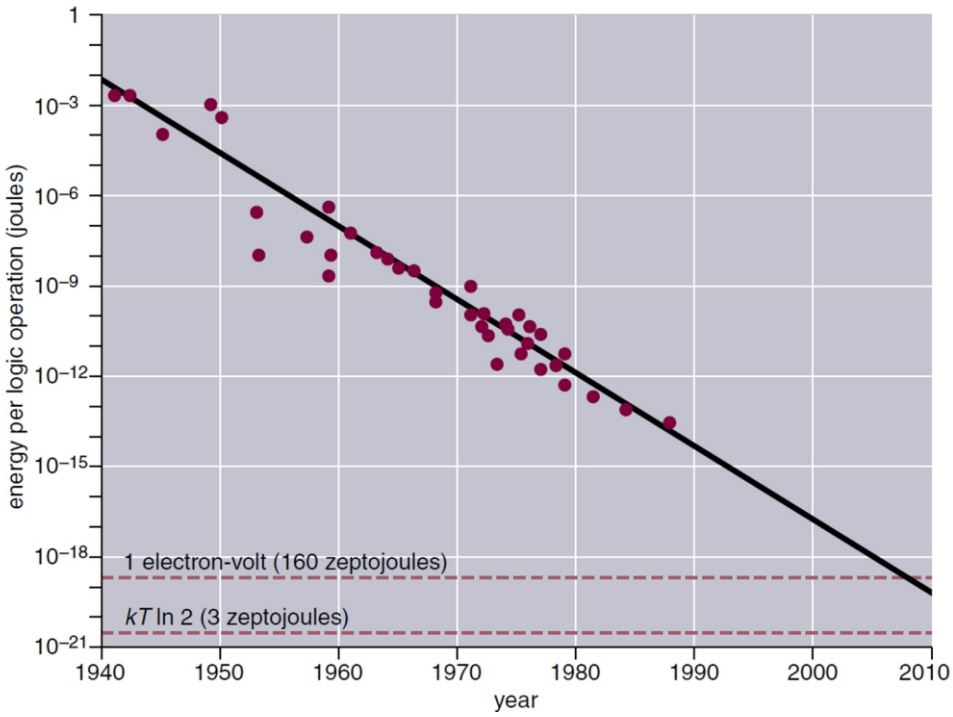
John von Neumann je leta 1949 podal minimalno mejo izgube energije na bit informacije ob izvedbi reverzibilne logične operacije:

$$E \geqslant k * T * ln(2). \tag{2.1}$$

V izrazu (2.1), T predstavlja temperaturo okolja, k je Boltzmannova konstanta, $ln(2)$ pa je naravni logaritem, ki se pojavlja zaradi enega bita informacije. En bit informacije je količina, ki je potrebna za razlikovanje med dvema enakovrednima stanjema. Količine imajo sledeče vrednosti: $T = 300K$, $k = 1,38 * 10^{-23} J/K$ in naravni logaritem $ln(2)$ je približno $0,69315$. Iz tega po izrazu (2.1) sledi, da je minimalna meja izgube energije $E = 3 * 10^{-21} J$. To je enako povprečni energiji ene molekule zraka pri sobni temperaturi [1].

Von Neumannov izraz temelji na *termodinamičnem argumentu*. Če iščemo odgovor na vprašanje z možnim odgovorom da ali ne, imamo za en bit informacije več kot smo jo imeli prej, ko vprašanje še ni bilo postavljeno. Torej računski proces zmanjša entropijo (nevednost) za en bit. Drugi zakon termodinamike pravi, da se skupna entropija ne more

zmanjšati, zato mora biti vzrok zmanjšanja entropije v računalniku posledica povečanja entropije zunaj računalnika. Torej je računalnik povzročitelj vsaj enega bita nereda v svoji okolici, kar se odda v obliki toplote, ki je enaka $k * T * ln(2)$. Von Neumann je predpostavljal, da ima vsaka elementarna sprememba informacije isti učinek kot odgovor na da ali ne vprašanje, torej vsak korak v računski operaciji porabi vsaj $3 * 10^{-21} J$ energije [1]. Na sliki 2.1 je prikazana poraba energije pri računanju.



Slika 2.1 Poraba energije računanja – merjena v J na logično operacijo – mora enakomerno padati, če hoče računalniška zmogljivost še naprej eksponentno naraščati. Kot prikazuje graf, se bomo kmalu približali prelomnici. En elektron-volt je energija posameznega elektrona na potencialu enega volta. Termodinamična meja so 3 zepto-jouli ($3 * 10^{-21} J$) – prehod pod to mejo zahteva, da računalniki postanejo reverzibilni [1].

V šestdesetih letih prejšnjega stoletja je Rolf Landauer odkril, da se porabi $E = 3 * 10^{-21} J$ energije pri računskih operacijah, kot je npr. brisanje vsebine spominske celice (uničevanje informacije). To je nenavadno, saj so domnevali, da se energija porabi pri operacijah, ki tvorijo informacijo, kot so shranjevanje, prenašanje in manipulacija informacije. Torej se pri brisanju vsebine spominske celice prenese v okolje vsaj toplotna energija $E = 3 * 10^{-21} J$ [1].

Čeprav brisanje potrebuje energijo, je Landauer odkril, da to ne velja za vse računalniške

operacije. Primer tega so *inverzna vrata*, ki spremenijo stanje enega bita iz 1 v 0 in iz 0 v 1. V tem primeru ni nobene izgube informacije, stanje pred in po operaciji je znano. Današnja implementacija inverznih vrat potrebuje enako količino energije kot brisanje celice, vendar po zakonih termodinamike ta izguba ni potrebna. Torej so inverzna vrata reverzibilna [1].

V prihodnosti bodo reverzibilna vezja edina fizikalno mogoča pot, da se še izboljšuje zmogljivost računalniških sistemov [6].

V sledečem podpoglavju izvemo kaj je logična reverzibilnost in ali obstaja še več reverzibilnih vrat, kot so inverzna vrata.

2.2 Logična reverzibilnost

S seznanitvijo s fizično reverzibilnostjo smo ugotovili, da je logična reverzibilnost pogoj za fizično. Preslikovalna funkcija je logično reverzibilna, če ustreza naslednjim pogojem:

- preslikava je informacijsko brezizgubna,
- preslikava slika enako število vhodov na enako število izhodov (bijektivna preslikava),
- vsaka kombinacija vhodnih spremenljivk mora vrniti enolično izhodno kombinacijo (poljubni dve izhodni kombinaciji sta različni),
- dvojna uporaba funkcije vrne prvotno vhodno vrednost, $f(f(x)) = x$,
- funkcija in njen inverz trajata različno dolgo [1].

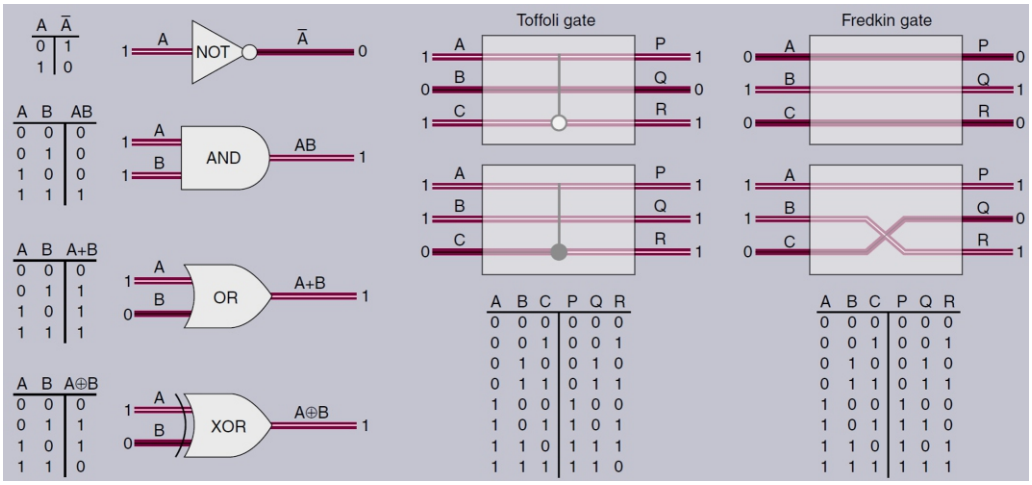
Izmed širše uporabljanih binarnih funkcij sta logično reverzibilni funkciji identiteta (I) in inverz (NOT). Število vhodov pri obeh funkcijah je enako številu izhodov (1 vhod, 1 izhod). Velja tudi $I(I(x)) = x$ in $NOT(NOT(x)) = x$ [1].

Ker identiteta in inverz nista funkciji, ki bi predstavljali funkcijsko poln nabor (primer funkcijsko polnega nabora je par funkcij AND in NOT), z njima ni mogoče realizirati poljubnih logičnih vrat, tako da uporabljamo le ti dve funkciji. Zato sta Edward Fredkin in Tommaso Toffoli prišla do rešitve, ki je podrobneje opredeljena v naslednjem podpoglavju.

2.3 Toffolijeva in Fredkinova logična vrata

Leta 1973 je Charles H. Bennett prišel do ugotovitve, da je mogoče narediti vsako računanje na enosmernih računalnikih reverzibilno [1]. To je demonstriral s pomočjo Turingovega stroja (model računanja, ki oponaša človekovo vedenje pri reševanju problemov in lahko izračuna vse, kar se izračunati da). Turingov stroj je imel tri trakove – enega za računanje, drugega za pisanje zgodovine in tretjega za zapis končnega rezultata. Nato se je postavil v reverzibilno stanje in s pomočjo zgodovine izvedel operacije eno za drugo, kar ga je privedlo do prvotnega stanja [1].

Ker Turingovi stroji niso prijazni za realizacijo (so zelo počasni), sta Edward Fredkin in Tommaso Toffoli leta 1980 predstavila logična vrata, ki omogočajo reverzibilnost [1]. Pri takih vratih velja, da je število vhodov (3) enako številu izhodov (3). Vsaka trojica vhodov mora vrniti enolično trojico izhodov, saj v nasprotnem primeru ne vemo, kateri izhod je posledica katerega vhoda. Par vrat se imenuje Toffolijeva vrata in Fredkinova vrata. Pri Fredkinovih vratih en signal nadzira, če drugi dve podatkovni liniji spremenita stanje ali ne. V Toffolijevih vratih dva signala nadzirata tretjega, če sta prva dva bita enaka 1, je tretji bit invertiran [1]. Ponazoritev Toffolijevih in Fredkinovih vrat je prikazana na sliki 2.2.



Slika 2.2 Slika prikazuje Toffolijeva in Fredkinova vrata ter njuni pravilnostni tabeli. Pri Toffolijevih vratih se le ob prisotnosti vrednosti 1 na prvih dveh vhodih A in B invertira vrednost tretjega izhoda R. Fredkinova vrata samo ob prisotnosti vrednosti 1 na vhodu A, zamenjajo vrednosti drugih dveh vhodov; B=R in C=Q. Na sliki so še reverzibilna inverzna vrata (NOT) in enosmerna (nereverzibilna) AND-, OR- in XOR-logična vrata. Vsa logična vrata imajo tudi pripadajoče pravilnostne tabele, ki prikazujejo preslikavo vhodnih vrednosti v izhodne [1].

Podobno kot inverzna vrata, imata tudi Fredkinova in Toffolijeva vrata funkcijsko lastnost dvojne uporabe, ki vrne prvotno vhodno vrednost. Torej v logična vrata podamo vhodne vrednosti, iz logičnih vrat dobimo izhodne vrednosti. Te izhodne vrednosti podamo spet kot vhod v ista logična vrata, rezultat je vrednost prvotnega vhoda. Velja tudi, da so oboja vrata računsko univerzalna. To pomeni, da lahko računalnik, ki sestoji iz več Fredkinovih ali Toffolijevih vrat, simulira Turingov stroj ali katerikoli drugi ekvivalentni računski model. Sta torej primerna kandidata za resnični reverzibilni računalnik [1].

Poznamo tudi implementacije drugih reverzibilnih vezij. Primer so XOR-logična vrata, ki so v osnovni implementaciji enosmerna. Če spremenimo izhod tako, da ponovimo en vhod, dobimo reverzibilna vrata. Vrata imajo sledečo zgradbo: vhod je par (A,B), izhod pa par (A,A$\oplus$B). Takšna logična vrata so reverzibilna, ker omogočajo sklepanje na vhodno stanje vezja iz izhodnega stanja. Tudi število vhodov (2) je enako številu izhodov (2) [1].

Še nekaj multi funkcionalnih reverzibilnih vrat je prikazanih na sliki 2.3, ki so dvovhodna, trovhodna in kvadrovhodna reverzibilna logična vrata.

Vsa do sedaj omenjena reverzibilna logična vrata so dvovrednostna. Zanima nas tudi kako je z reverzibilnostjo v trovrednostnih logičnih vratih. Zakaj? Zato ker ima trojiška logika določene pomembne lastnosti, ki so podrobneje opisane v prvem podpoglavju tretjega poglavja.

Ker so dvovrednostna vrata podmnožica trovrednostnih, veljajo vse lastnosti dvovrednostnih reverzibilnih vrat tudi za trovrednostna. Razlika je le v številu logičnih vrednosti, ki jih vratom podamo na vhod. Število logičnih vrednosti neposredno vpliva na število permutacij vhodnih vrednosti (v posamezna logična vrata). Iz vira [3] izvemo, da za trovrednostna vrata z dvema vhodoma in dvema izhodoma obstaja 362880 reverzibilnih vrat. Naš načrt dela je izbrati primerna (odvisno od lastnosti) dvovrednostna reverzibilna vrata in ugotoviti ali se reverzibilnost ohranja tudi v trovrednostni logiki, oziroma ali je mogoče ustrezno dopolniti funkcionalnost logičnih vrat do reverzibilnosti.

Gate	Input Vector	Output Vector	Output			
			$P=$	$Q=$	$R=$	$S=$
2*2 Feynman Gate [25]	$I(A,B)$	$O(P,Q)$	A	$A \oplus B$	X	X
3*3 Double Feynman Gate [26].	$I(A,B,C)$	$O(P,Q,R)$	A	$A \oplus B$	$A \oplus C$	X
3*3 Toffoli Gate [23]	$I(A,B,C)$	$O(P,Q,R)$	A	B	$AB \oplus C$	X
3*3 Fredkin Gate [24]	$I(A,B,C)$	$O(P,Q,R)$	A	$A'B \oplus AC$	$A'C \oplus AB$	X
3*3 New Gate [27]	$I(A,B,C)$	$O(P,Q,R)$	A	$AB \oplus C$	$A'C' \oplus B'$	X
3*3 Peres Gate [28]	$I(A,B,C)$	$O(P,Q,R)$	A	$A \oplus B$	R=AB⊕C $AB \oplus C$	X
3*3 NFT Gate [26]	$I(A,B,C)$	$O(P,Q,R)$	$A \oplus B$	$B'C \oplus AC'$	$BC \oplus AC'$	X
4 * 4 BVF Gate[29]	$I(A,B,C,D)$	$O(P,Q,R,S)$	A	$A \oplus B$	C	$C \oplus D$
4*4 HNG Gate [30]	$I(A,B,C,D)$	$O(P,Q,R,S)$	A	B	$A \oplus B \oplus C$	$(A \oplus B)C \oplus AB \oplus D$
4*4 HNFG Gate[31]	$I(A,B,C,D)$	$O(P,Q,R,S)$	A	$A \oplus C$	B	$B \oplus D$
4*4 SCL Gate[32]	$I(A,B,C,D)$	$O(P,Q,R,S)$	A	B	C	$A(B+C) \oplus D$
4*4 TSG Gate[30]	$I(A,B,C,D)$	$O(P,Q,R,S)$	A	$A'C' \oplus B'$	$(A'C' \oplus B') \oplus D$	$(A'C' \oplus B)D \oplus (AB \oplus C)$
4*4 MTSG Gate[33]	$I(A,B,C,D)$	$O(P,Q,R,S)$	A	$A \oplus B$	$A \oplus B \oplus C$	$(A \oplus B)C \oplus AB \oplus D$
Multifunction Reversible (BVMF) Gate[34]	$I(A,B,C,D)$	$O(P,Q,R,S)$	$(A+B) \oplus C$	$A \oplus B$	$AB' \oplus D$	$AB \oplus D$

Slika 2.3 Tabela prikazuje relacije med vhodom in izhodom za različna dvovrednostna reverzibilna logična vrata. Tu lahko opazimo, da so Toffolijeva vrata enostavnejša za realizacijo kot Fredkinova, saj uporabljajo ena XOR-logična vrata, medtem ko Fredkinova vrata potrebujejo dvojna [7].

3 Trovrednostni QCA

3.1 Trojiška osnova

Prejšnje poglavje se je osredotočalo na reverzibilnost, v tem poglavju pa je predstavljena procesna platforma, s katero lahko realiziramo vezja, ki so reverzibilna. Ljudje pri računanju uporabljamo desetiško logiko, v računalništvu pa se najpogosteje uporablja dvojiška logika. To je predvsem zaradi odpornosti vezij proti motnjam (šum), ki povzročajo nepravilno delovanje, kar bi bil pri vezjih v desetiški logiki večji problem (majhna tolerantnost do motenj). A izkaže se, da bi imelo vezje narejeno v trojiški logiki pomembne prednosti pred dvojiškimi vezji, desetiškimi in vsemi drugimi zato se osredotočimo na pomen trojiške logike.

Trojiška osnova je uporabna, ko je dvojiška osnova premajhna, desetiška pa prevelika. Števila, ki jih predstavljajo različni številski sistemi, so ista, razlika med njimi je v predstavitvi posameznih števil. Splošni izraz, ki predstavlja število v katerikoli poziciji je:

$$...d_3r^3 + d_2r^2 + d_1r^1 + d_0r^0... \tag{3.1}$$

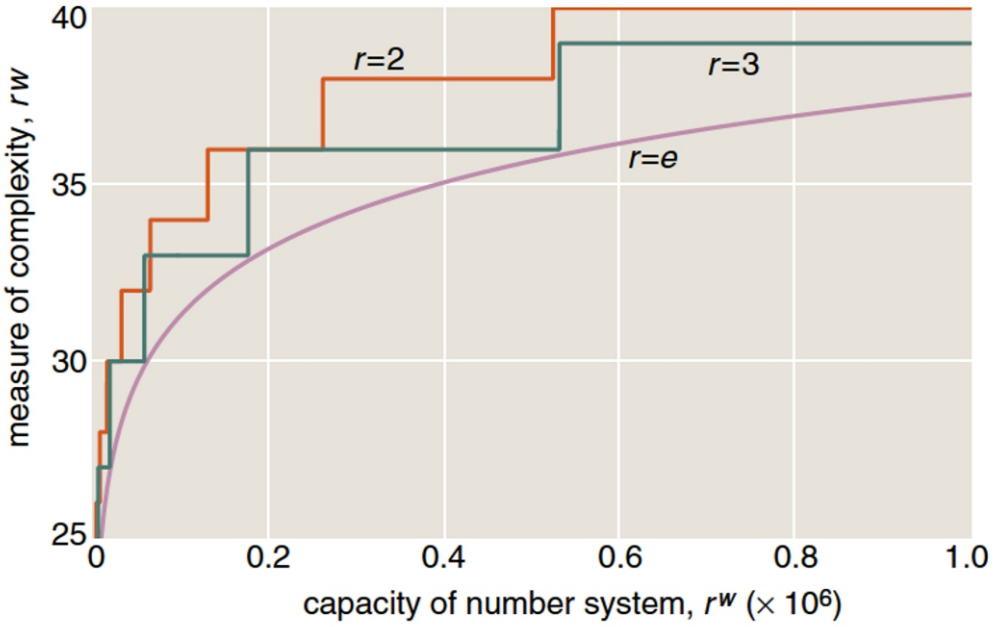
V izrazu (3.1) predstavlja r številsko osnovo, koeficienti d_i pa so pozicije števil. Primer

je predstavitev števila 19 v desetiški osnovi, predstavitev števila 201 v trojiški osnovi in 10011 v dvojiški osnovi [2]. Primer trojiške predstavitve števila 201 po izrazu (3.1) bi bil sledeč:

$$2 * 3^2 + 0 * 3^1 + 1 * 3^0. \tag{3.2}$$

Iz izraza (3.2) sledi, da vse številske osnove predstavljajo vsa števila, vendar je od situacije odvisno, v kolikšni meri so posamezne številske predstavitve dobre. Desetiška osnova je dobra za ljudi, ker lahko štejejo na roke, dvojiška osnova pa je zaradi enostavnosti in zanesljivosti uporabna za računalnike. Trojiška osnova je najboljša zaradi največje učinkovitosti (najbolj ekonomičen način predstavitve števil). Ekonomičnost je vrednost, odvisna od števila pozicij, ki jih posamezno število, predstavljeno v posamezni številski osnovi, zavzame. Najmanj ekonomičen je eniški zapis, saj vsako število predstavlja vsaj toliko pozicij, kolikokrat moramo številu ena prišteti ena, da pridemo do vrednosti izbranega števila. Večja kot je številska osnova, manj je pozicij, ki jih mora predstavljeno število zavzemati. Pomembna je čim manjša številska osnova, ker v primeru velike številske osnove potrebujemo veliko število simbolov (enako velikosti številske osnove), ki opisujejo izbrano število. Največja ekonomičnost je torej najboljše razmerje med številom pozicij, ki jih zavzema izbrano število, in številom simbolov, ki predstavljajo to število. Če za razmerje števila pozicij in števila simbolov dobimo vrednost 2.718 (e, osnova za logaritem), je to najbolj ekonomična vrednost. Vendar potrebujemo celoštevilsko osnovo in številu e je najbližje število 3 [2]. Na sliki 3.1 je prikazana ekonomičnost.

Pomembnost trojiške osnove so opazili že leta 1950 (zmanjšanje stroškov izdelave strojnih komponent računalnika). Izdelano je bilo okrog 50 računalnikov na trojiški osnovi, ampak so pri izdelavi naredili napako in trojiškega sistema niso uporabili za zmanjšanje števila komponent, kar je izničilo njegovo prednost pred dvojiškim sistemom. Prednost trojiškega sistema je tudi večja ekonomičnost trojiške logike. Primer je primerjava dveh števil (x in y): ali je x manjši od y, glede na odgovor je mogoče še vprašanje, ali je x enak y. V trojiški aritmetiki dobimo en odgovor s tremi možnostmi: manj, enako, večje, kar je že končni odgovor. Izdelan je bil tudi znani računalnik na trojiški osnovi z imenom TERNAC, ampak trojiškim računalnikom ni uspelo priti v ospredje, predvsem zaradi dobro uveljavljene dvojiške osnove [2].



Slika 3.1 Graf prikazuje, da je najbolj ekonomična osnova vedno 3, ker je najbližje vrednosti *e*-ja. Na sliki w predstavlja število pozicij, r pa število simbolov. Razmerje med kapacitivnostjo in kompleksnostjo poimenujemo ekonomičnost. Na vodoravni osi je predstavljena kapacitivnost sistema, na navpični osi pa kompleksnost sistema. Najboljše razmerje med kapacitivnostjo in kompleksnostjo sistema predstavlja osnova, ki je enaka *e*. Zaradi celoštevilskih osnov vzamemo najbližjo, ki je 3 [2]. Trojiška logika je najbolj enostavna logika iz množice večvrednostnih logik, trojiški številski sistem pa je najučinkovitejši v predstavljanju števil [5].

Trojiška predstavitev je sestavljena iz množice števil $\{0, 1, 2\}$, a se zaradi simetrije navadno uporablja množica $\{-1, 0, 1\}$. Torej se število 19 zapiše kot število 1 -1 0 1 [2]. Primer (3.3) predstavlja zapis števila 19 v trojiški logiki po izrazu (3.1) z uporabo množice števil $\{-1, 0, 1\}$:

$$1 * 3^3 - 1 * 3^2 + 0 * 3^1 + 1 * 3^0. \tag{3.3}$$

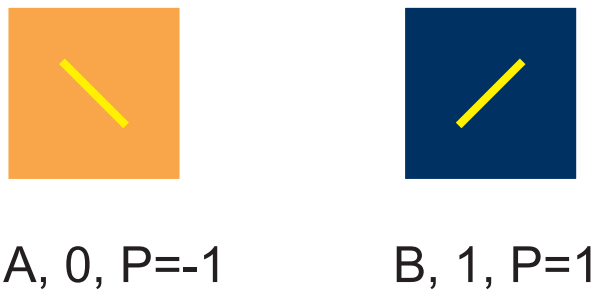
3.2 Celica QCA

Zaradi učinkovitosti trojiške osnove je najboljša izbira za procesno platformo QCA (kvantni celični avtomat, angl. *Quantum-dot Cellular Automata*). QCA je alternativna struktura za procesiranje, ki dosega današnje računalnike po merilih prenosljivosti, zanesljivosti, varčnosti, povečevanja procesne hitrosti in nadzora procesiranja [4]. Temelje je

postavil C. S. Lent v začetku prejšnjega stoletja. Na QCA je mogoče implementirati tako dvojiško kot tudi trojiško logiko, sama implementacija struktur vezij pa je izvedljiva s kvantnimi celicami in njihovo medsebojno interakcijo ter izkoriščanjem kvantne lastnosti *tuneliranja*. Implementacija vezij v QCA-ju bazira na Coulombovi interakciji med elektroni [7].

Kvantna celica je osnovna struktura v QCA-ju in je kvadratne oblike ter omogoča sestavljanje bolj kompleksnih ravninskih struktur. Navadno ima kvantna celica štiri kvantne pike, ki so razporejene vsaka v svoj vogal kvadrata. Kvantna pika ima lastnost zadrževanja elektrona. Elektron se torej nahaja znotraj celice v eni od štirih kvantnih pik. Z lastnostjo *tunelskega učinka* elektron prehaja med kvantnimi pikami. V eni kvantni celici sta navadno dva elektrona, med katerima delujejo odbojne sile (Coulombova sila). Zaradi prisotnosti odbojnih sil elektron preide v kvantno piko, ki maksimizira njegovo stabilno stanje. Torej elektron prehaja v stanje, kjer so odbojne sile okolja in sosednjega elektrona minimalne. Odbojne sile okolja so sile, ki delujejo na določen elektron iz sosednjih celic. Če postavimo v celico dva elektrona, se elektrona glede na rezultanto sil uravnovesita v eno izmed dveh leg.

Za zagotavljanje sredstev za procesiranje mora vsaka procesna platforma uporabljati neko preslikavo iz fizičnih količin v logične vrednosti. Klasični CMOS uporablja napetostne mere, dvovrednostni QCA (Lentov model) pa uporablja polarizacijo [5]. Tako dobimo sistem z dvema stanjema. Kot je razvidno iz slike 3.2, ima eno diagonalno stanje polarizacijo P=-1 in drugo diagonalno stanje polarizacijo P=1. Prvo diagonalno stanje označuje logično vrednost 0, drugo pa logično vrednost 1 [4].



Slika 3.2 Slika prikazuje klasično dvovrednostno QCA-celico po Lentu. Prva celica prikazuje QCA-celico v stanju A, kar ustrezna logični vrednosti 0 in ima polarizacijo enako -1. Druga celica prikazuje QCA-celico v stanju B, logična vrednost 1 in polarizacija je enaka 1 [4].

3.3 Zakonitosti izračunavanja stabilnega stanja celice

Medsebojno učinkovanje sosednjih kvantnih celic določa zakone, po katerih QCA deluje. Če je celica izolirana od zunanjih sil, sta legi P=-1 in P=1 enako verjetni in je celica nevtralna. Kvantne pike imajo pozitivni električni naboj $\frac{n*e}{m}$, kjer je m število kvantnih pik v celici, n število elektronov in e naboj elektrona (pri Lentovi kvantni celici je $m = 4$ in $n = 2$). Kvantne pike z elektronom imajo naboj $\frac{n*e}{m} - e$. Vsota nabojev kvantnih pik znotraj ene celice je enaka 0 in stanji $P = -1$ in $P = 1$ sta energijsko enakovredni [4].

Množica celic je določena s celotno elektrostatično potencialno energijo, ki se izračuna kot energija sistema točkastih nabojev po izrazu (3.4).

$$E = \sum_{i \neq j} \frac{\rho_i * \rho_j}{4 * \pi * \varepsilon_0 * \varepsilon_r * r_{ij}} \tag{3.4}$$

V izrazu (3.4) je ρ_i naboj kvantne točke i, r_{ij} razdalja med kvantnima točkama i in j, ε_0 permitivnost vakuuma in ε_r relativna permitivnost medija (izvedba GaAs ali AlGaAs tehnologije) [4]. Izračun vrednosti E (3.4) za vse mogoče kombinacije leg elektronov določi lego, ki privede do energijsko minimalnega stanja. Potreben je pregled $\binom{m}{n}^k$ možnih kombinacij, kjer je k moč množice celic, kjer ima vsaka m kvantnih točk in n elektronov [4].

3.4 Dvojiško procesiranje v QCA-ju

Procesiranje v dvovrednostnem QCA je podobno kot v trovrednostnem QCA-ju. Ker je trovrednostni QCA razvit na podlagi dvovrednostnega QCA-ja delujeta oba po enakih principih. Razlike so v številu stanj, v katerih je lahko določena kvantna celica in nekatere strukture zato delujejo drugače oziroma so bolj občutljive na motnje.

Z ustreznim časovno omejenim zaklepanjem posameznih celic je onemogočeno prehajanje elektronov in dosežena stabilnost v strukturi. Časovni signal je realiziran v obliki električnega polja, ki se razprostira pod kvantnimi celicami in s svojo prisotnostjo omogoča *tunelski učinek* ali ga z neprisotnostjo onemogoča. S takšnim delovanjem neposredno določa zmožnosti procesnih gradnikov. Časovnost je določena s štirifaznim urinim sistemom: Switch, Hold, Release in Relax. V urini fazi Hold je omogočeno prehajanje signala [4].

QCA ima tri vrste celic, s katerimi realiziramo procesne strukture:

- vhodne celice, kjer z vplivom zunanjih sil dosežemo želeno razporeditev elektronov,

- notranje celice, v katerih se izvede logična obdelava vhodnih dvojiških podatkov,
- izhodne celice, katerih stanje je rezultat procesiranja.

Princip, ki definira delovanje QCA-ja, je robno vodeno računanje. Predpostavlja se, da so vhodne celice (s podatki za procesiranje) postavljene ob meji strukture in so njihova stanja nadzorovana z zunanjimi elektrostatičnimi polji. Prav tako morajo biti izhodne celice (ki vsebujejo izračunane odgovore) na robu strukture, da se njihov rezultat lahko uporabi kot vhod za naslednje stukture, ki bodo ta rezultat uporabljale kot svoj vhod. Stanja se interpretirajo in berejo kot logične vrednosti, ki predstavljajo logične funkcije, implementirane s QCA-jem. Vse druge celice so notranje celice, ki so edine celice, ki opravljajo transformacijo podatkov [5].

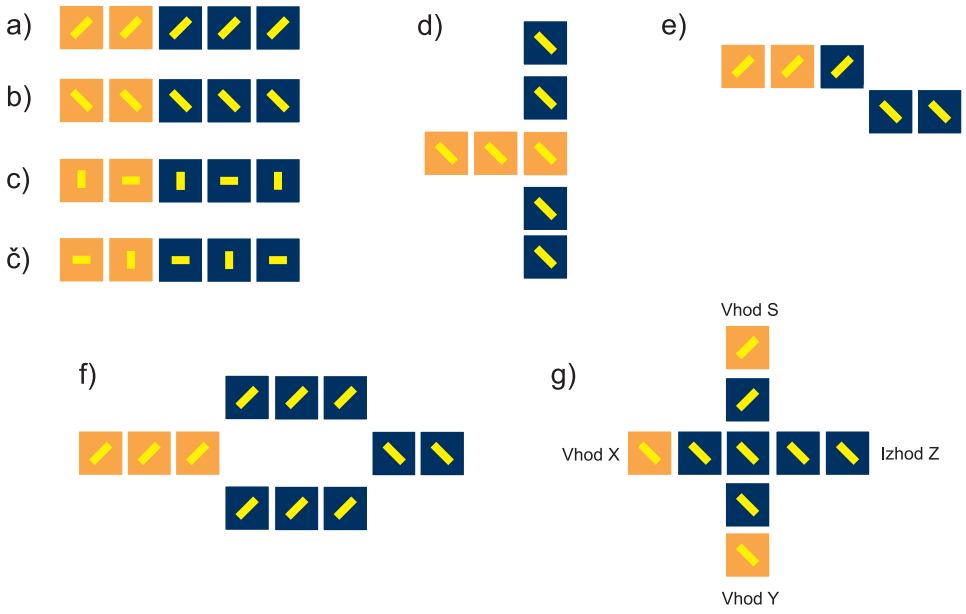
V devetdesetih letih prejšnjega stoletja so razvili strukture (logična vrata), ki določajo poln nabor logičnih funkcij (AND, OR, NOT). AND in OR sta realizirana v obliki dvovrednostnih večinskih vrat (angl. *majority gate*), ki jih označimo z M. NOT predstavlja realizacijo inverznih logičnih vrat. V QCA-strukturah lahko zapišemo {M,NOT} za funkcijsko poln nabor [4].

Na sliki 3.3 so prikazani osnovni gradniki, s katerimi v dvovrednostnem QCA-ju strukturiramo kompleksnejša vezja. Ena od osnovnih struktur v QCA so večinska vrata (M), katerih izraz za izhod je sledeč:

$$\mathrm{M(A, B, C)} = \mathrm{AB} + \mathrm{AC} + \mathrm{BC}. \tag{3.5}$$

Večinska vrata je mogoče realizirati s petimi QCA-celicami. Posebnost je nastavljivo delovanje, kjer eden od treh vhodov služi za spremenitev delovanja celice v AND- ali OR-logično operacijo. Poleg večinskih vrat poznamo še inverter, ki spremeni logično vrednost iz 0 na 1 in obratno. Za prenos signala se uporablja binarna linija, ki ima tudi rotirajočo izvedenko, pri kateri signal glede na sosednjo celico varira: če začnemo z 0 1 0 1 0 in obratno za prenos vrednosti 1 (za prenos informacije pri rotirajoči liniji potrebujemo liho število celic) [7].

Osnovne značilnosti poljubnega računalniškega sistema so procesiranje, prenašanje podatkov in pomnjenje. QCA-strukture so sposobne tako procesiranja kot prenašanja podatkov. Če na celico ne deluje nobena znanja sila, celica ohrani svojo vrednost (pomnjenje). V tem pogledu imajo QCA-strukture vse lastnosti, ki jih potrebujemo za implementacijo poljubnega računalniškega sistema [4].

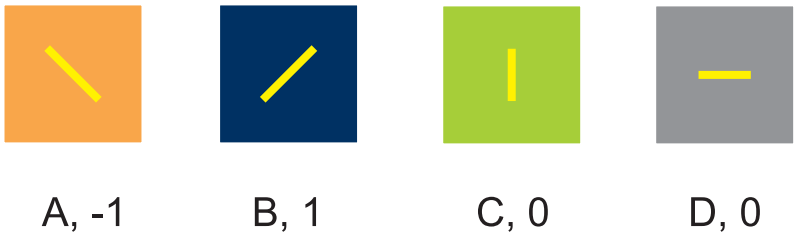


Slika 3.3 Slika prikazuje QCA-celice, ki sestavljajo osnovne gradbene elemente. Primer a prikazuje prenos vrednosti B (logična 1) po liniji, primer b prenos vrednosti A (logična 0) po liniji, primera c in č prenašata vrednosti po rotirani liniji (vrednosti sta še vedno A in B, le predstavitev je rotirana). Opazimo, da se vrednosti prenašajo pravilno le, če je linija dolga liho število celic. Primer d predstavlja razmnoževanje signala, torej se vrednost A prenese na dve različni liniji. Primera e in f ponazarjata šibki in močni inverter (razlika v odpornosti na šum), vhodna vrednost se invertira (B v A). Primer g predstavlja večinska vrata. Delovanje teh je odvisno od vrednosti vhodov S, X, Y in rezultat pride na izhod Z. Glede na vrednost S-a lahko implementiramo logična vrata AND ali OR. X in Y sta spremenljivki [4].

3.5 Procesiranje v strukturah trovrednostnega QCA-ja

Slabost dvovrednostnih QCA-celic je predvsem v dveh stanjih. Razširjena struktura kvantne celice ima zato osem kvantnih pik, vse ostalo ostane enako. Kvantne pike so razporejene v krožnem vzorcu z radijem $D/sin(\pi/8)$, tako da je razdalja med sosednjima kvantnima pikama enaka $2D$. Vsaka kvantna pika ima naboj $\rho_+ = e_0/4$, kjer je e_0 elementaren naboj [5]. To nam omogoča, da imamo namesto dveh možnih stanj štiri. Stanja so A, B, C in D. Ta stanja ustrezajo pozicijama dveh elektronov, ki sta obe diagonali (kot pri dvovrednostnem QCA-ju), horizontala in vertikala. Kakršnekoli druge pozicije so nestabilne in jim ne pripisujemo stanj. Zakonitosti izračunavanja stabilnega stanja so enake kot pri dvovrednostnem QCA-ju. Na sliki 3.4 je prikazana realizacija celic [4].

Pozor, čeprav sta stanji C in D videti kot stanji A in B pri dvovrednostnem QCA-ju, to nista. Pri dvovrednostnem QCA-ju sta v primeru rotirajoče linije ti dve poziciji zasukani, a še vedno le dve. V primeru trovrednostnega QCA-ja pa sta stanji C in D prisotni tudi v navadnih strukturah. Rotirajoča linija pri trovrednostnem QCA-ju ne obstaja, ker je že del implementacije stanj C in D. Kvantna celica v dvovrednostnem QCA-ju sprejme dve stanji, pri trovrednostnem QCA-ju pa štiri stanja.



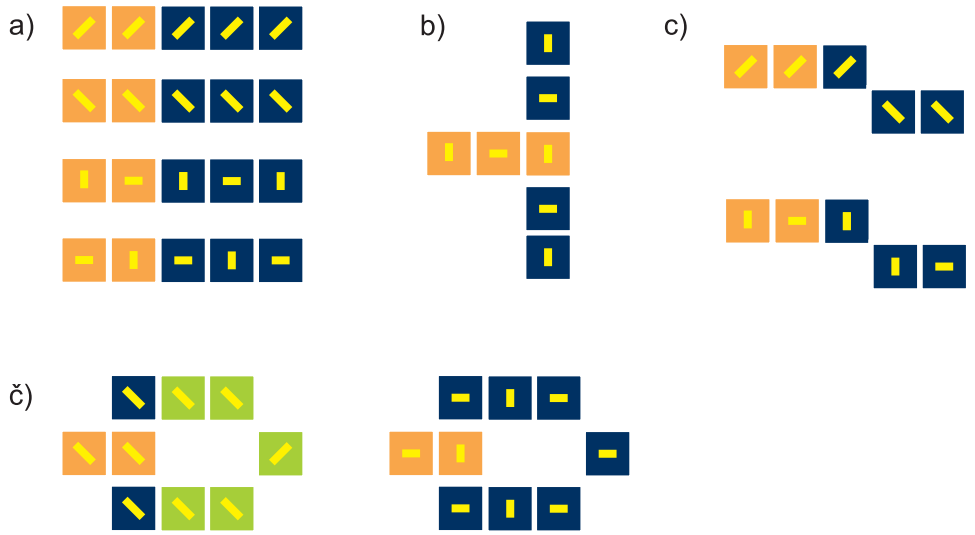
Slika 3.4 Slika prikazuje celice razširjenega QCA-ja, ki določajo stanja A, B, C, D. Poleg stanj je podana logična vrednost, ki jo stanje določa. Slika prikazuje tudi vsa stabilna stanja, ki jih definirata dva elektrona znotraj posamezne QCA-celice. Vsa druga stanja so nestabilna [5].

Primerjava delovanja osnovnih dvojiških logičnih funkcij AND, OR in linije, se izkaže za drugačno, vendar odstopanja niso velika. Po liniji se signala A in B prenašata pravilno, težava nastane pri prenosu stanj C ali D, ki se prenašata spreminjajoče. Prenos stanja C je zaporedje C D C D C D C. Prenos po linij je pravilen, če je linija dolga liho število celic. V nasprotnem primeru je dobljena vrednost D. Podobno velja pri prenosu stanja D, le da se v primeru napačnega delovanja prenese stanje C namesto stanja D.

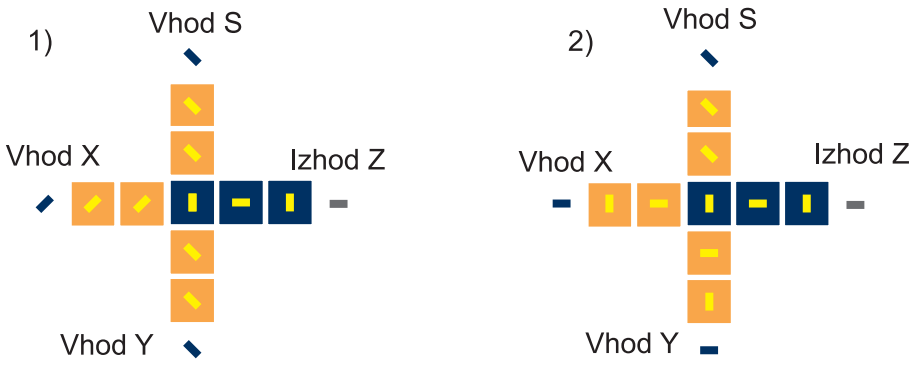
Zaradi testnih procesirnih zmožnosti se je potrebno odpovedati stanju D, kar nas privede do sistema treh stanj $\{\mathrm{A}, \mathrm{B}, \mathrm{C}\} = \{-1, 1, 0\}$, v katerem se lahko nahajajo vhodne ali izhodne celice. Uporablja se Lukasiewiczev trojiški logični sistem za trojiške operacije negacije (inverza), konjunkcije in disjunkcije (AND in OR logični funkciji). Negacija deluje pravilno, večinska funkcija, namenjena za realizacijo logičnih funkcij AND in OR, pa dobi pri dveh kombinacijah vhodnih spremenljivk napačen izhod. Zato se nadgradi večinsko funkcijo, da ustreza funkcionalnosti (uporaba treh večinskih vrat namesto enih) in tako dobi pravilne izhode [4].

3.6 Elementarne trojiške strukture v trovrednostnem QCA-ju

Trovrednostni QCA nam omogoča grajenje kompleksnejših struktur. Na slikah 3.5 in 3.6 so predstavljene primitivne funkcije, ki ponazarjajo linijo, rob, razmnoževanje signala (*fan-out*), inverter, simetrični inverter, AND- in OR- logična vrata (večinska vrata). Po funkcionalnosti so sorodne istim strukturam v dvovrednostnem QCA-ju, razlika je le v trajanju, ki je sinhronizirano z urinim signalom [5].



Slika 3.5 Slika prikazuje delovanje trovrednostnega QCA-ja v različnih primerih. Primer a predstavlja prenos signala po liniji (podobno kot pri klasični QCA-celici), kjer se v prvih dveh primerih signal ohranja (prenos vrednosti B in A), v drugih dveh pa alternira (prenos vrednosti C, D), tako da se vrednosti C in D izmenjujeta glede na sosednjo celico. Pravilna vrednost se prenese v primeru C in D vrednosti le, če je število celic v liniji liho. Primer b predstavlja razmnoževanje signala, primer c sestavljata dva primera inverterja (B v A in C v D), primer č pa je podoben c-ju, le da gre za močni inverter [5].



Slika 3.6 Slika prikazuje večinska vrata za trovrednostni QCA v dveh primerih. Prvi prikazuje napačno delovanje vrat, saj vhod sestavlja nabor vrednosti A, B, A, za izhod Z pa dobimo vrednost D (pravilno bi bilo A, saj so to večinska vrata). Takemu primeru se izognemo z uporabo strukture treh večinskih vrat, ki se izognejo napačnim vhodnim vrednostim. Drugi primer predstavlja pravilno delovanje, saj se vhod A, D, D preslika v izhod D [5].

4 Realizacija Toffolijevih vrat

4.1 Metodologija dela

Delo je potekalo v programskem orodju qdCAD, simulatorju delovanja trovrednostnega kvantnega celičnega avtomata, ki je bil razvit na Fakulteti za računalništvo in informatiko Univerze v Ljubljani. V posebno datoteko (s končnico *.qdStruct*) vpišemo osnovne parametre, ki jih orodje potrebuje za definicijo prostora in postavitev kvantnih celic v njem. To počnemo s spreminjanjem osnovnih parametrov kvantnih celic, kot so:

- delovanje v dvovrednostnem ali trovrednostnem svetu,
- pozicija,
- urina faza posamezne celice,
- tip celice (vhodna, notranja, izhodna),
- začetna vrednost celice.

Poleg tega v datoteki definiramo tudi število vseh celic in dodelimo vrednosti vhodnim celicam. To datoteko nato simulator sprejme in nad simulacijo uporabi vizualizator, da

lahko vidimo strukture, ki smo jih definirali. Orodje omogoča zapis rezultatov v izhodno datoteko in shranitev trenutne vizualizacije v obliki slike.

Iz izhodne datoteke je mogoče dobiti informacije o posameznih preslikavah vhoda na izhod (s tem preverimo, če smo dobili pravilno oziroma pričakovano vrednost), polarizacijo izhoda in toleranco napake izračuna (*epsilon*), iz katere sklepamo na stabilnost izračuna posamezne izhodne vrednosti strukture.

Med samo simulacijo je mogoče vizualno preverjati, kako vpeljana struktura deluje, tako da:

- zaustavimo izvajanje simulacije, se prepričamo o njenem delovanju in potem nadaljujemo z izvajanjem od točke zaustavitve naprej,
- prevzamemo nadzor nad potekom simulacije (naprej ali nazaj) in na ta način podrobneje preverimo prenos vrednosti med sosednjimi kvantnimi celicami,
- izkoristimo tridimenzionalni pogled (strukture lahko gradimo v več ravninah) in približujemo ali oddaljujemo posamezne konstrukte z *zoom* funkcijo,
- pregledamo polarizacijo posamezne celice z grafičnim vmesnikom.

Vizualizator prikazuje kvantne celice, katerih barva pove fazo urine periode posamezne celice, logična vrednost celice pa je predstavljena s pomočjo postavitve elektronov v kvantnih pikah (stabilne pozicije so leva diagonala, desna diagonala, vodoravnica, navpičnica). Vhodne in izhodne celice so vidno drugačne od notranjih, tako da so prozorne, notranje pa pobarvane.

Vse strukture, ki smo jih izdelali, so bile realizirane in testirane s pomočjo simulatorja. Izdelava reverzibilnih trovrednostnih logičnih vrat je potekala v več korakih. Najprej smo preverili obnašanje osnovnih gradnikov kompleksnejših struktur. Ti gradniki so linija, razmnoževalnik signala, inverz, močni inverz in večinska vrata. Večinska vrata omogočajo realizacijo logične funkcije AND ali OR (s konstantno vrednostjo na enem izmed treh vhodov v večinska vrata).

Pred samo realizacijo trovrednostnih reverzibilnih vrat smo imeli sledeče ideje:

- logična vrata v treh dimenzijah,
- drugačna razporeditev celic XOR-vrat, ker je mogoče okrog ene celice v ravnini razporediti osem sosedov,

- spreminjanje delovanja logičnih vrat z izkoriščanjem vpliva skupin sosednjih celic na izbrana vrata,
- redefinicija večinskih vrat z odvzemom celic, kar nam omogoča večjo hitrost,
- izbira Toffolijevih ali Fredkinovih logičnih vrat, ker v dvovrednostni logiki predstavljajo funkcijsko poln nabor in so reverzibilna.

Za realizacijo Toffolijevih vrat smo se odločili zaradi dobrega poznavanja njihovega delovanja, ker jih je mogoče hitro realizirati (majhno število komponent) in uporabljajo le eno kompleksno strukturo (XOR-logična vrata). Vsa ostala znana dvovrednostna trovhodna vrata (Feynmanova vrata, Fredkinova vrata, Peresova vrata, NFT-vrata, New-vrata) so strukturno težja za realizacijo, saj sestoje iz bolj kompleksnih pod-struktur. Večja kompleksnost prinese tudi daljši čas izvajanja simulacije in manjšo odpornost na motnje signala.

Realizacijo smo omejili na ravnino zaradi večje preglednosti posameznih komponent in začeli z realizacijo XOR-logičnih vrat, saj ta še niso bila realizirana v trovrednostnem QCA-ju, so pa del zgradbe Toffolijevih vrat.

4.2 XOR-vrata

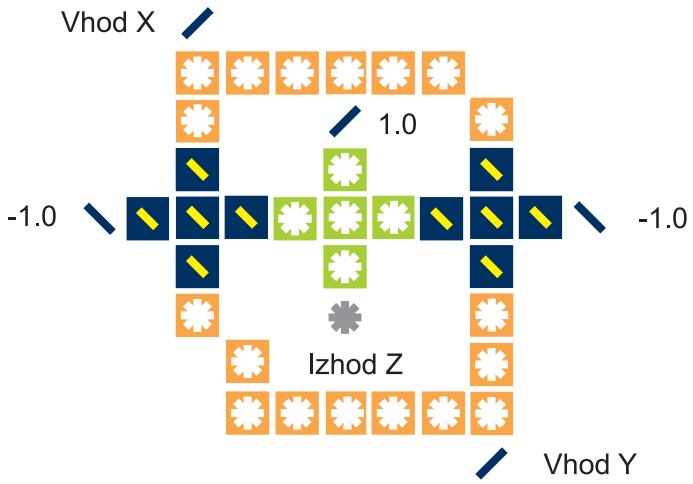
Toffolijeva vrata opisuje izraz:

$$\begin{aligned} [a, b, c] &= [a, b, ((a \wedge b) \oplus c)] \\ &= [a, b, (\neg(a \wedge b) \wedge c) \vee ((a \wedge b) \wedge \neg(c))] \end{aligned} \tag{4.1}$$

Iz izraza (4.1) je razvidno, da Toffolijeva vrata sprejmejo tri vhodne vrednosti in vrnejo tri izhodne vrednosti. Poleg tega so Toffolijeva vrata sestavljena iz enih AND- in enih XOR-logičnih vrat. XOR-logična vrata so sestavljena iz dveh inverznih vrat, dveh AND- in enih OR-logičnih vrat. Zaradi nepoznavanja delovanja Toffolijevih vrat v trovrednostnem svetu, smo se najprej odločili za realizacijo dvovrednostnih vrat. Po pregledu rezultatov bi videli, ali je smiseln prehod v trovrednostno logiko, ali je potrebno poiskati drugačno pot.

V viru [7] smo našli realizacijo dvovrednostnih XOR-logičnih vrat, ki smo jo preslikali v qdCAD-simulator in je prikazana na sliki 4.1. To smo storili, ker je takšna realizacija blizu idealne realizacije, tako po številu kvantnih celic, ki jo sestavljajo, kot tudi po razporeditvi vhodov in izhodov v ravnini. Navadno ostane izhod nedosegljiv za nadaljnje

strukture, ki uporabljajo ta izhod kot vhod. Ta omejitev nastane zaradi geometrijske oblike vezja in se lahko pojavi tudi pri vhodu v strukturo, če to primerno preoblikujemo (takrat je izhod dosegljiv za druge strukture).

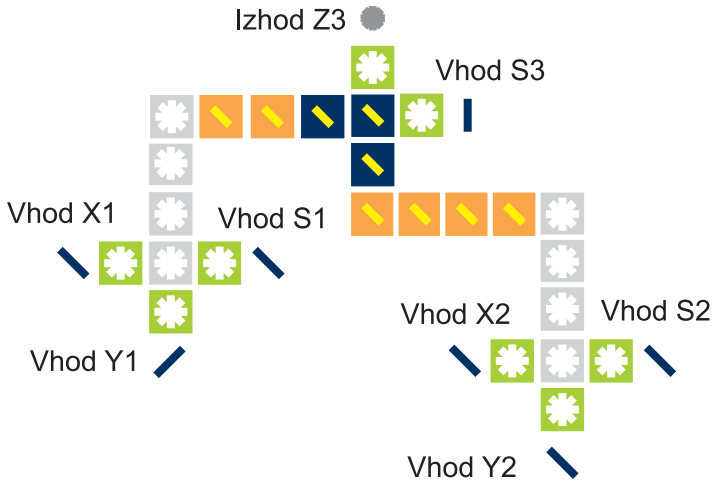


Slika 4.1 Slika prikazuje dvojiška XOR-logična vrata. X, Y sta vhoda, Z pa izhod. Polarizacija 1.0 označuje večinska vrata v vlogi OR-logičnih vrat, -1.0 pa označuje večinska vrata v vlogi AND-logičnih vrat [7].

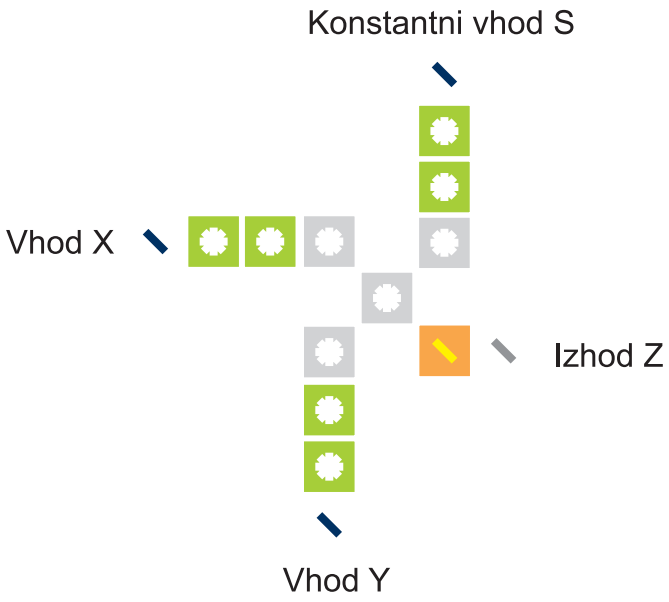
Med realizacijo se je pojavil problem. Dvovrednostna večinska vrata delujejo korektno, dokler smo omejeni na dvovrednostne vhode, ko pa preidemo na trovrednostne, se pojavi pri nekaterih vhodnih vrednostih nepravilno delovanje [4]. To se zgodi pri AND-logičnih vratih in vhodni kombinaciji (A,B,A) in pri OR-logičnih vratih pri vhodni kombinaciji (B,A,B). V obeh primerih dobimo na izhodu vrednost D, kar ni pravilno.

Rešitev je uporaba dodatnih dveh večinskih vrat, ki se povezujejo na tretja večinska vrata, vsaka na svoj vhod, kot prikazuje slika 4.2. Ker sedaj potrebujemo namesto vsakih AND- ali OR-logičnih vrat kombinacijo treh večinskih vrat, se kompleksnost XOR-logičnih vrat že pri navadnih AND- ali OR-logičnih operacijah poveča (število celic je trikrat večje). S pomočjo pregleda različnih virov smo prišli do rešitve v obliki diagonalnih trovrednostnih večinskih vrat [5]. Diagonalna večinska vrata ne zmorejo odpraviti vhodov, ki vodijo v nepravilno delovanje, vendar imajo določene prednosti pred navadnimi večinskimi vrati zaradi geometrijske oblike. Prikazana so na sliki 4.3.

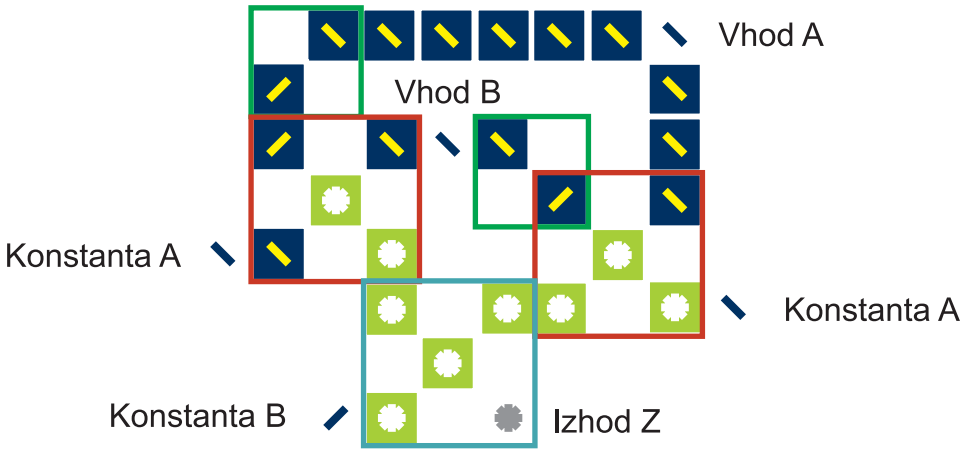
XOR-logična vrata smo tako realizirali s pomočjo diagonalnih večinskih logičnih vrat. Takšna struktura je pripomogla k bolj stabilnemu, enostavnejšemu in hitrejšemu logičnemu vezju, ki ga prikazuje slika 4.4.



Slika 4.2 Slika prikazuje kombinacijo prvih in drugih večinskih vrat na tretja, s pomočjo katere dosežemo selekcijo vhodnih vrednosti v tretja večinska vrata, kar nam omogoča pravilno delovanje večinskih vrat v trovrednostnem QCA-ju [4].



Slika 4.3 Slika prikazuje diagonalna večinska vrata z vhodi S, X, Y in izhodom Z. Vrednost izhoda je podobna kot pri navadnih večinskih vratih. Prednost je bolj stabilno delovanje in hitrejši izračun izhoda. Obstajata dve izvedbi, ki se ločita po številu urinih faz, potrebnih za pridobitev rezultata. Prikazana je počasnejša izvedba, ki je pri bolj kompleksnih vezjih bolj v uporabi zaradi večje odpornosti na motnje [5].

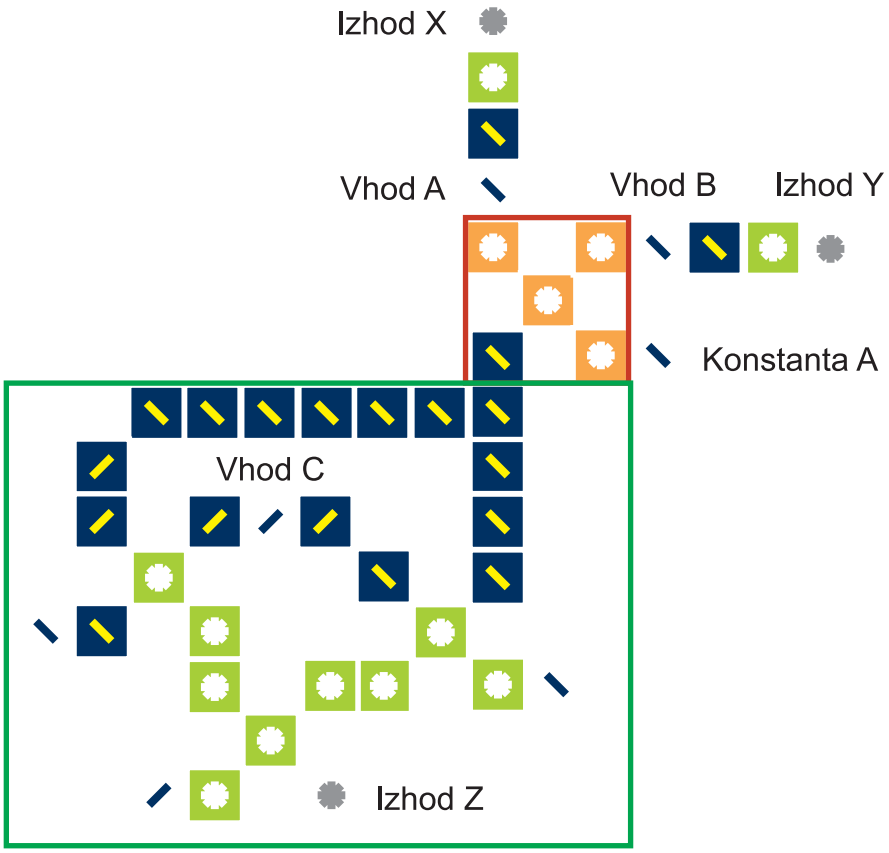


Slika 4.4 Slika prikazuje dvovrednostna XOR-logična vrata, realizirana s pomočjo diagonalnih večinskih vrat. Vrednosti A in B predstavljata vhoda v vezje, Z izhod iz vezja, konstantne vrednosti določajo funkcijo večinskih vrat (delovanje kot AND- ali OR-logična vrata). Vrata delujejo pravilno. Inverzi so označeni z zelenimi območji, AND-vrata z rdečimi in OR-vrata vrata z modrim. Struktura je realizirana s 30 celicami in se izvaja 0.75 urine periode [7].

4.3 Dvovrednostna Toffolijeva vrata

Dvovrednostna Toffolijeva vrata so po izrazu (4.1) sestavljena iz XOR- in AND-logičnih vrat. Za njihovo izgradnjo smo uporabili XOR-logična vrata, ki jih prikazuje slika 4.4. Najprej smo pripeljali dva vhoda v vezje na AND-logična vrata, nato se je rezultat prenesel na XOR-logična vrata. Ker imajo XOR-logična vrata dva vhoda, pripeljemo na en vhod rezultat iz AND-vrat, na drugi vhod pa preostali (tretji) vhod. Izhod sestavlja trojica vrednosti, ki jo sestavljajo prvi in drugi vhod v AND-vrata, ter izhod iz XOR-vrat.

Celotno vezje ima tri vhode in tri izhode. Zaradi motenj, ki nastajajo pri prenosu signala prek strukture, je bilo potrebno XOR-vrata razširiti z dodatnimi celicami, da smo dobili stabilen signal. Pravilne preklope med posameznimi elementarnimi strukturami (večinska vrata, negator) pa zagotavlja korektna razporeditev faz urine periode. Dvovrednostno vezje deluje pravilno in je prikazano na sliki 4.5.



Slika 4.5 Slika prikazuje realizacijo dvovrednostnih Toffolijevih vrat z diagonalno XOR-logično strukturo, ki je na sliki označena z zelenim območjem in AND-logičnimi vrati, ki so v rdečem območju. Toffolijeva vrata imajo tri vhode (A,B,C) in tri izhode (X,Y,Z) in delujejo pravilno. Celotna struktura je zgrajena iz 44 celic in se izvaja eno urino periodo.

4.4 Trovrednostna Toffolijeva vrata

Na dvovrednostna Toffolijeva vrata smo kot vhod podali štiri-vrednostne spremenljivke, saj je implementacija trovrednostenga QCA-ja definirana po izrazu:

$$(\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}) = (-1, 1, 0, 0), \tag{4.2}$$

kar pomeni, da je logična vrednost 0 predstavljena z dvema stanjema C in D.

Analiza rezultatov je pokazala, da ti niso pravilni. Vzrok so bile motnje, ki so nastale zaradi prehoda na trovrednosto logiko. Stanji C in D nista pravilno prehajali med sosednjimi celicami, saj je trovrednosten QCA v nekaterih primerih bolj občutljiv na vpliv celic v bližji okolici. Rešitev je bila preoblikovanje vezja, tako da dodamo večje

število celic in s tem bolj razmaknemo posamezna logična vrata med seboj in vrednosti se lahko brez motenj prenašajo po linijah. Dodatno stabilnost je zagotovilo tudi vstavljanje večjega števila urinih period, ki omogoča pravilno delovanje posameznih vrat.

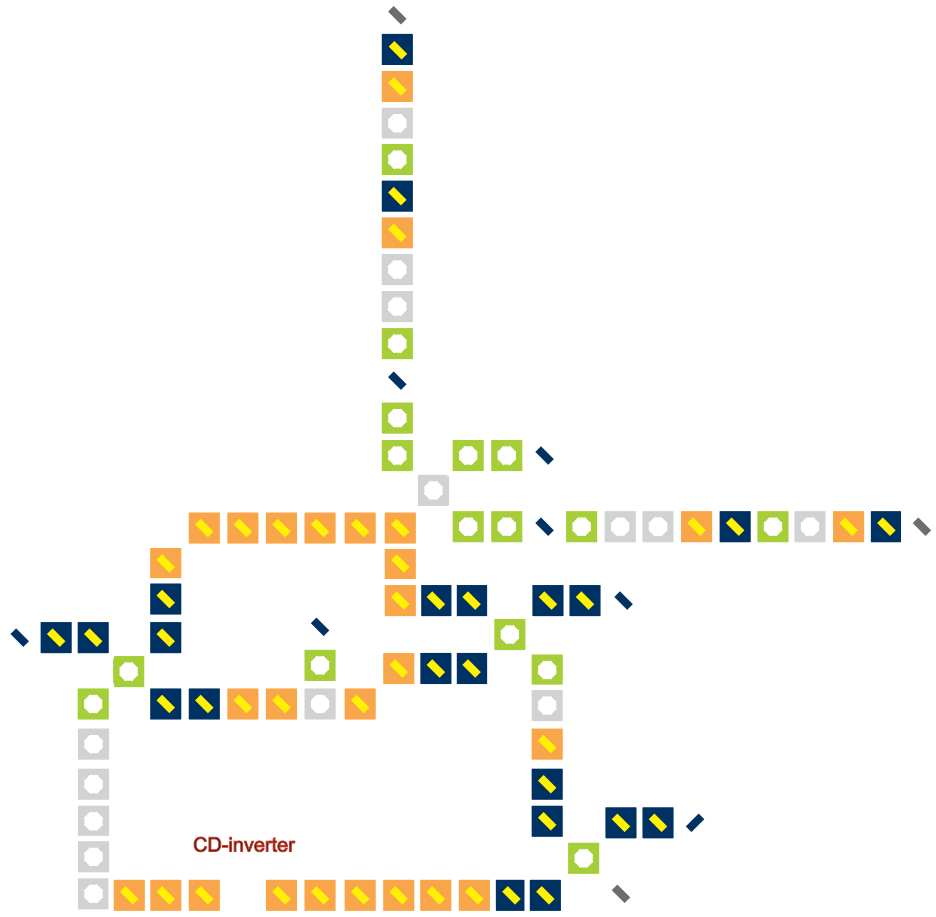
Med popravljanjem vezja opazimo, da prenos signala C in D ni pravilen. OR-logična vrata sprejmejo dva vhoda, ki delujejo pravilno le v primeru vhodnega stanja A ali B. Prenos stanj C in D je v trovrednostnem QCA-ju izmeničen glede na posamezno sosednjo celico. Če se prenaša stanje C je primer prenosa C D C D C, primer prenosa stanja D pa D C D C D. V obeh primerih opazimo, da dobimo pravilno vrednost na liniji samo v primeru, ko je število celic v liniji liho. Posledično lahko OR-logična vrata prejmejo na prvi vhod pravilno vrednost, na drugega pa nepravilno. Velja tudi obratno. To se zgodi v vseh možnih postavitvah OR-logičnih vrat, saj je problem geometrijske narave. OR-logična vrata so vedno enake oblike, tako da lahko problem rešimo s prestrukturiranjem večinskih vrat (kar nam ni uspelo) oziroma z uporabo nove strukture, ki bi odpravila ta problem.

Uspelo nam je realizirati strukturo CD-inverter. Ta struktura transformira stanje C v D ali D v C, pri stanju A ali B pa se vrednost stanja ohranja. Strukturo prikazuje slika 4.6.

Slika 4.6 Slika prikazuje CD-inverter, kjer je vhod označen z A in izhod z Z. V primeru vhoda stanja C vrne struktura stanje D in v primeru vhoda stanja D vrne stanje C, v primeru vhoda stanja A ali stanja B se vrednost vhoda ohranja. Na sliki je primer transformacije stanja D v stanje C. Preskok signala čez prazno celico pomeni, da je takšna struktura manj odporna na motnje, ki jih povzroča vpliv sosednjih celic. Da se ohranja pravilno delovanje, je potrebno to strukturo bolj odmakniti od inverznih in večinskih logičnih vrat, zelo je tudi pomembno pravilno določanje faz urine periode.

Na sliki 4.7 so prikazana trovrednostna Toffolijeva vrata z uporabo CD-inverterja. Iz slike sledi, da signal z vhodnimi stanji (A, B, C, D) vstopa v AND-logična vrata (dva vhoda), potem pa se po izračunu izhoda ta prenese na XOR-logična vrata. Tu se mu pridruži še tretji vhod. XOR-logična vrata delujejo s pomočjo CD-inverterja, ki je postavljen pred OR-logična vrata. Končni rezultat so trije izhodi, med katerimi sta prva dva enaka prvima dvema vhodoma, tretji pa sledi transformaciji v vezju.

Vezje s slike 4.7 deluje pravilno samo za dvovrednostni svet. Trovrednostni svet



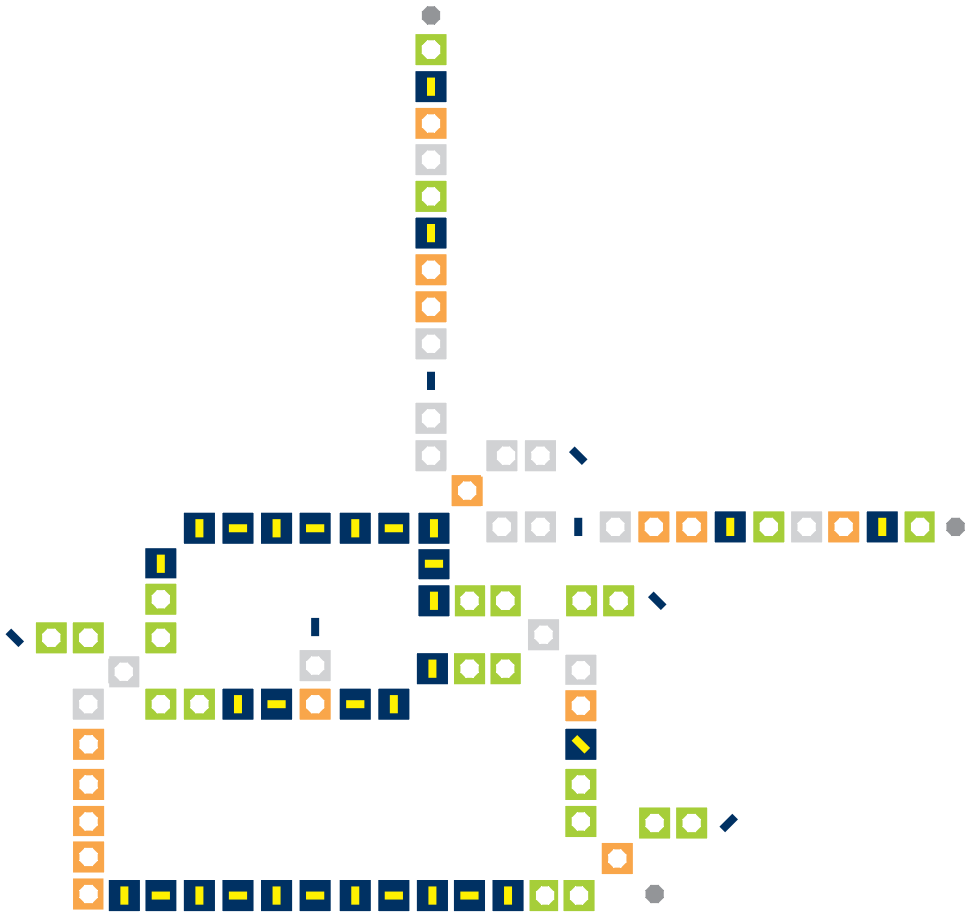
Slika 4.7 Slika prikazuje realizacijo trovrednostnih Toffolijevih vrat, ki so izpeljana iz dvovrednostnih. Rdeča območja predstavljajo AND-logična vrata, modra OR-logična vrata, CD-inverter pa je označen z zelenim območjem. Strukturo sestavlja 89 celic in se izvede v 2.5 urine periode.

prinese enolične izhode (noben izhod se ne ponovi in vsi so permutacije vhodnih trojic). Ampak to ni zadosten pogoj za logično reverzibilnost. V nekaj primerih se stanji (A,B) preslikata v (C,D) ali (D,C). To v logičnem pomenu predstavlja slikanje logičnih vrednosti $(-1, 1)$ v $(0, 0)$, kar pomeni, da takšno vezje ni reverzibilno, saj se informacija o prvotnem stanju izgublja. Problem poskušamo rešiti z odpravljanjem vhodnih vrednosti, ki se ne preslikujejo pravilno.

4.5 Iskanje rešitve

Na sliki 4.7 je predstavljena realizacija trovrednostnih Toffolijevih vrat, ki logično sledijo konstrukciji dvovrednostne različice, vendar ne izpolnjujejo pogoja reverzibilnosti. Da bi pripravili vrata do pravilnega delovanja, smo se osredotočili le na vhodna stanja (A,B,C). Tak pristop odpravi potrebo po strukturi CD-inverter, saj se stanje D ne pojavi na izhodih večinskih vrat. Če pravilno postavimo strukture, tako da so vse med seboj oddaljene liho število celic, se stanje C ne transformira v stanje D.

Popravljena struktura je prikazana na sliki 4.8. Kljub omejitvi na tri možna stanja Toffolijeva vrata ne zadostijo pogojem logične reverzibilnosti.



Slika 4.8 Slika predstavlja končno verzijo realizacije trovrednostnih Toffolijevih vrat, pri čemer CD-inverter ni več potreben. Število celic je 93, izvajanje strukture pa traja 2.5 urine periode.

Intuicija nam pravi, naj začnemo rešitev iskati na drugačen način. V programskem jeziku Python napišemo program, ki simulira delovanje trovrednostne logike (inverz, AND- in OR-logična vrata, XOR-logična vrata). S pomočjo programa preverimo delovanje naše strukture. Dobimo podoben rezultat, se pravi Toffolijeva vrata, ki so bila reverzibilna v dvovrednostnem svetu, a v trovrednostnem niso. Preizkus opravimo še za ostale kandidate, ki so reverzibilni v dvovrednostnem svetu (Fredkinova vrata, Peresova vrata, NFT-vrata, New-vrata, Feynmannova vrata) [7]. Rezultati so enaki. Nobena vrata, ki smo jih preizkusili ne zadoščajo pogojem reverzibilnosti v trovrednostnem svetu.

Svojo pozornost osredotočimo na samo delovanje (definicijo) inverza in AND- in OR-logičnih vrat v trovrednostnem svetu. Definicija razkrije, da preizkušena vrata ohranjajo vrednost logične 0. Se pravi inverz$(0) = 0$, AND$(0, 0) = 0$, OR$(0, 0) = 0$ in XOR$(0, 0) = 0$. To pomeni, da s temi strukturami ne moremo preslikati logične 0 v logično 1 ali -1.

Funkcijo, ki bi zadostila takšnim pogojem, si lahko zamislimo v obliki permutacije vhodnih trojic (a, b, c) v izhodne trojice (x, y, z). Takšna funkcija bi lahko bila rotacijska funkcija, ki opravlja preslikavi stanj (A,B,C) v (B,C,A) ali (A,B,C) v (C,A,B). Vendar tudi takšna preslikava ne omogoča reverzibilnosti. Če uporabimo funkcijo dvakrat, pomeni da izhod iz prve funkcije prejme druga funkcija na vhod in vrne svoj izhod, bi moral biti končni izhod prvotna vhodna vrednost, a se to ne zgodi. Takšno delovanje bi dosegli s trikratno uporabo funkcije oziroma s funkcijo, ki bi preslikovala v eno smer enkrat, za preslikavo nazaj (reverzibilno) pa dvakrat. Torej tudi to ni rešitev.

Če bi se rotacijska funkcija izkazala za reverzibilno, nam ne bi omogočala realizacije, saj logičnih vrat s takšnim delovanjem v trovrednostnem QCA-ju še ne poznamo. Izhod uporabe rotacijske funkcije na trojicah vhodov prikazuje spodnja tabela 4.1

4.6 Analiza rezultatov

Iz delovanja naše realizacije reverzibilnih trovrednostnih Toffoljevih vrat pridemo do sklepa, da ta realizacija ne omogoča reverzibilnosti kot prikazuje tabela 4.1. Tudi rotacijska funkcija, ki preslikuje stanje C v A ali B ji ne zadosti. Da to potrdimo, stopimo korak nazaj in preglejmo izhode za realizaciji Toffolijevih vrat s stanji A, B, C, D (4.7) in za Toffolijeva vrata s stanji A, B, C (4.8). Rezultati so predstavljeni v obliki dveh tabel 4.2 in 4.3.

a b c = x y z	a b c = x y z	a b c = x y z
A A A = A A C	B A A = B A C	C A A = C A C
A A B = A A A	B A B = B A A	C A B = C A A
A A C = A A B	B A C = B A B	C A C = C A B
A B A = A B C	B B A = B B C	C B A = C B C
A B B = A B A	B B B = B B A	C B B = C B A
A B C = A B B	B B C = B B B	C B C = C B B
A C A = A C C	B C A = B C C	C C A = C C C
A C B = A C A	B C B = B C A	C C B = C C A
A C C = A C B	B C C = B C B	C C C = C C B

Tabela 4.1 Tabela prikazuje preslikavo trojice vhodov (a,b,c) v trojico izhodov (x,y,z) za rotacijsko funkcijo. Vse možne kombinacije vhodov se slikajo v izhode, tako da se vsak izhod pojavi le enkrat.

Program, ki simulira trojiško logiko, se zgleduje po viru[1] in njegova izvorna koda se nahaja v prilogi A. Program tudi pri pregledu nekaterih ostalih zanimivih kandidatov za reverzibilna vrata (Fredkinova vrata, Peresova vrata, NFT-vrata, New-vrata, Feynmannova vrata) vrne rezultate, ki po preslikavi dvojiške logike na trojiško ne ohranjajo reverzibilnosti in so prikazani v prilogi A. Toffolijeva vrata smo izbrali zaradi reverzibilnosti in univerzalnosti, vendar se je izkazalo, da je največji problem trovrednostna osnova, zaradi katere se izgubi reverzibilnost. Da bi prišli do možne rešitve, bi bilo potrebno nadaljnje delo v smislu iskanja kandidatov za reverzibilna trovrednostna logična vrata, kot je pregled najrazličnejših virov (v zvezi z reverzibilnostjo in večstanjskimi logičnimi vrati) in preizkušanje kombinacij različnih logičnih vrat. Sama realizacija v trovrednostnem QCA-ju se ni izkazala za zelo problematično. Potrebna je bila pazljivost pri prehajanju signala v odvisnosti od motenj iz ostalih delov strukture in nadzor prehajanja stanj z urinim signalom.

[1] http://c2.com/cgi/wiki?ThreeValuedLogic

a b c = x y z	a b c = x y z	a b c = x y z	a b c = x y z
A A A = A A A	B A A = B A A	C A A = C A A	D A A = D A A
A A B = A A B	B A B = B A B	C A B = C A B	D A B = D A B
A A C = A A C	B A C = B A C	C A C = C A C	D A C = D A C
A A D = A A D	B A D = B A D	C A D = C A D	D A D = D A D
A B A = A B A	B B A = B B B	C B A = C B C	D B A = D B D
A B B = A B B	B B B = B B A	C B B = C B D	D B B = D B C
A B C = A B C	B B C = B B D	C B C = C B A	D B C = D B B
A B D = A B D	B B D = B B C	C B D = C B B	D B D = D B A
A C A = A C A	B C A = B C C	C C A = C C C	D C A = D C A
A C B = A C B	B C B = B C D	C C B = C C D	D C B = D C B
A C C = A C C	B C C = B C A	C C C = C C A	D C C = D C C
A C D = A C D	B C D = B C B	C C D = C C B	D C D = D C D
A D A = A D A	B D A = B D D	C D A = C D A	D D A = D D D
A D B = A D B	B D B = B D C	C D B = C D B	D D B = D D C
A D C = A D C	B D C = B D B	C D C = C D C	D D C = D D B
A D D = A D D	B D D = B D A	C D D = C D D	D D D = D D A

Tabela 4.2 Tabela prikazuje vhodna in izhodna stanja Toffolijevih vrat s štirimi stanji. Pri tem opazimo, da se vse trojice vhodov preslikajo v izhode. Če so vhodi enolično določeni, so prav tako tudi izhodi. Problem se pojavi pri preslikavah, kjer se tretji vhod c preslikuje v izhod z, pri tem pa ne ostane logično enoličen. Lahko si predstavljamo, da stanja A nadomestimo z logično -1, B z 1, C z 0 in D z 0. V tem primeru določene preslikave niso več enolične, npr. zgled (B,C,C)=(B,C,A) in (B,C,D)=(B,C,B), se prepiše v $(1, 0, 0) = (1, 0, -1)$ in $(1, 0, 0) = (1, 0, 1)$. To dokazuje, da se tretji vhod z vrednostjo 0 preslikuje v -1 in 1. Takšna preslikava ni enolična, saj ne velja bijekcija preslikave, torej tudi realizacija ne morejo biti reverzibilna. Rešitev je omejitev stanj na A, B, C, ker potem ni več dveh stanj, kjer bi oba imela logično vrednost 0.

a b c = x y z	a b c = x y z	a b c = x y z
A A A = A A A	B A A = B A A	C A A = C A A
A A B = A A B	B A B = B A B	C A B = C A B
A A C = A A C	B A C = B A C	C A C = C A C
A B A = A B A	B B A = B B B	C B A = C B C
A B B = A B B	B B B = B B A	C B B = C B C
A B C = A B C	B B C = B B C	C B C = C B C
A C A = A C A	B C A = B C C	C C A = C C C
A C B = A C B	B C B = B C C	C C B = C C C
A C C = A C C	B C C = B C C	C C C = C C C

Tabela 4.3 Tabela vsebuje rezultate trovrednostnih Toffolijevih vrat. Brez stanja D, se izognemo primerom, ki so nam v tabeli 4.2 povzročali napačne preslikave. Preslikavi kaže dobro do tretjega stolpca v tabeli. Tu se v spodnjih šestih primerih vrednosti vhoda c, pri preslikavi v izhod z, preslikajo v stanje C. Izgubi se enoličnost preslikav. Rezultati so identični tistim, ki jih dobimo z uporabo programa, ki simulira trojiško logiko.

5 Zaključek

V prihodnosti bodo reverzibilna vezja edina fizikalno mogoča pot, da se še izboljšuje zmogljivost računalniških sistemov [6], zato je dobro razmišljati v tej smeri. Iz tega sledi, da se bo vedno več ljudi ukvarjalo z reverzibilnimi sistemi, in če nekomu uspe dobra realizacija reverzibilnega računalnika, bo to prineslo tehnološki napredek.

Naša realizacija reverzibilnih trostanjskih logičnih vrat ni uspela, vendar smo iz neuspeha dobili pomembne informacije. V trostanjskem QCA-ju je mogoče realizirati kompleksne strukture, ki z malo truda delujejo pravilno. Glede na vir [3] vemo, da obstaja 362880 trovrednostnih vrat, s tremi vhodi in tremi izhodi, ki so tudi reverzibilna. Torej moramo s pomočjo drugih virov oziroma raziskovanja priti do logično reverzibilnih funkcij, ki jih bomo potem lahko realizirali v trostanjskem QCA-ju. Zaenkrat ni vzroka za dvom, da se takšnih struktur ne da narediti v trostanjskem QCA-ju. Rezultati naše realizacije se ujemajo z rezultati programa, ki simulira delovanje trovrednostne logike.

Izvedeli smo, da Toffolijeva vrata, ki so prestrukturirana iz dvovrednostne logike v trovrednostno, ne ohranijo reverzibilnosti. Podobno velja tudi za ostala logična vrata (Feynmanova vrata, Fredkinova vrata, Peresova vrata, NFT-vrata, New-vrata), saj smo

njihove rezultate preverili s programom.

Ideje za nadaljnje raziskovanje bi lahko bile:

- implementacija reverzibilnih logičnih vrat v treh dimenzijah, kjer bi zaradi neraziskanih struktur lahko našli reverzibilna vrata,
- pogledati na problem iz drugega zornega kota, npr. matematičnega, in tam poiskati reverzibilne trostanjske funkcije,
- s pomočjo programa preverjati različne trostanjske logične funkcije in najti takšno, ki ustreza reverzibilnost.

Sledila bi realizacija v trostanjskem QCA-ju in imeli bi procesno platformo, ki bi izkoriščala tako reverzibilnost, kot tudi trostanjsko logiko, kar bi bil dober zagon za QCA-strukture. Sama trostanjska logika omogoča krajši zapis podatkov, ki so sedaj predstavljeni z dvostanjsko logiko. To omogoča tudi hitrejše delovanje računalnika, saj so podatki in ukazi, ki jih računalnik uporablja pri izračunih krajši, in se lahko zaradi tega hitreje prenašajo. Reverzibilnost omogoča prihranek pri porabi energije, kar je dobro za večje, kot tudi manjše računalniške sisteme. Pri prenosnih računalnikih bi potem potrebovali več energije le pri vključitvi, nato bi potrebovali sorazmerno malo energije za računanje in njihova neodvisnost od vira napajanja bi bila daljša. Zaradi manjših energijskih izgub se v okolje prenese manj toplote. To zmanjša stroške hlajenja naprav in omogoči potreben napredek v računski zmogljivosti.

Literatura

[1] B. Hayes, Reverse engineering, American Scientist 94 (2) (2006) 107–111.

[2] B. Hayes, Third base, American Scientist 89 (6) (2001) 490–494.

[3] P. Kerntopf, M. A. Perkowski, M. H. A. Khan, On universality of ternary reversible logic gates, 2013 IEEE 43rd International Symposium on Multiple-Valued Logic 0 (2004) 68–73.

[4] I. Lebar Bajec, N. Zimic, M. Mraz, The ternary quantum-dot cell and ternary logic, Nanotechnology 17 (8) (2006) 1937–1942.

[5] P. Pečar, A. Ramšak, N. Zimic, M. Mraz, I. Lebar Bajec, Adiabatic pipelining: a key to ternary computing with quantum dots, Nanotechnology 19 (49) (2008) 1–12.

[6] S. Sahni, A. Bakshi, Reversible computing: Looking ahead of the curve, http://www.it.iitb.ac.in/ saurabh/documents/revpaper.pdf (Nov. 2004).

[7] N. A. Shah, F. A. Khanday, J. Iqbal, Quantum-dot cellular automata (qca) design of multi-function reversible logic gate, Communications in Information Science and Management Engineering 2 (4) (2012) 8–18.

A Priloga A

A.1 Realizacija Toffolijevih vrat

Programska koda za simulator trojiške logike:

```
# -*- coding: utf-8 -*-
"""
Created on Fri Apr 12 19:31:27 2013

@author: Mark Rolih

This program makes simulation of Taffoli and other gates
in ternary logic, where A=-1, B=1, C=0, D=0 and
A,B,C,D are logic states.

Select gate:
    0 - Toffoli gate
    1 - Fredkin gate
    2 - Double Feynman gate
    3 - Peres gate
    4 - New gate
```

```
    5 - NTF gate
    6 - Rotational gate
"""
"""logic formula for Toffoli gates
X, Y, Z are possible entrances to the
circuit - each have 4 possible values - list states
"""
"""
(X AND Y) XOR Z =
 = (NEG(X AND Y) AND Z) OR ((X AND Y) AND NEG(Z))
"""
"""
Definition of standard and, or, not and xor
operators in tristate logic.
"""
def tand(a,b):
    if (a=='T' and b=='T'):
        return 'T'
    elif ((a=='U' and b=='T') or (a=='T' and b=='U') or
    (a=='U' and b=='U')):
        return 'U'
    else:
        return 'F'

def tor(a,b):
    if (a=='F' and b=='F'):
        return 'F'
    elif ((a=='U' and b=='U') or (a=='F' and b=='U') or
    (a=='U' and b=='F')):
        return 'U'
    else:
        return 'T'

def txor(a,b):
    if ((a=='T' and b=='F') or (a=='F' and b=='T')):
        return 'T'
    elif ((a=='T' and b=='T') or (a=='F' and b=='F')):
        return 'F'
    else:
        return 'U'
```

```
def tnot(a):
    if (a=='T'):
        return 'F'
    elif (a=='F'):
        return 'T'
    else:
        return 'U'
"""
Definition of rotational gate.
"""
def trot(a):
    if (a=='T'):
        return 'F'
    elif (a=='F'):
        return 'U'
    else:
        return 'T'
"""
Simple function that transforms variables
('T','F','U') to ('B','A','C')
only that variables are the same as in tQCA.
"""
def rep(a):
    if (a=='T'):
        return 'B'
    elif (a=='F'):
        return 'A'
    else:
        return 'C'
"""
Generation of all possible sequences to see
the outcome.
"""
selection=int(raw_input())


states=['F','T','U']
for X in states:
    for Y in states:
        for Z in states:
            #result = (X & Y) ^ Z
```

```
#result=(~(X & Y) & Z) | ((X & Y) & ~(Z))

"""Toffoli logic function"""
if (selection==0):
    #zero example Toffoli gate
    result=txor(tand(X,Y),Z)
    print (rep(X)+' '+rep(Y)+' '+rep(Z)+' '+
    '='+' '+rep(X)+' '+rep(Y)+' '+rep(result))

elif (selection==1):
    #first example Fredkin gate
    res1=txor(tand(tnot(X),Y),tand(X,Z))
    res2=txor(tand(tnot(X),Z),tand(X,Y))
    print (rep(X)+' '+rep(Y)+' '+rep(Z)+' '+
    '='+' '+rep(X)+' '+rep(res1)+' '+rep(res2))

elif (selection==2):
    #second example Double Feynman gate
    res1=txor(X,Y)
    res2=txor(X,Z)
    print (rep(X)+' '+rep(Y)+' '+rep(Z)+' '+
    '='+' '+rep(X)+' '+rep(res1)+' '+rep(res2))

elif (selection==3):
    #third example Peres gate
    res1=txor(X,Y)
    res2=txor(tand(X,Y),Z)
    print (rep(X)+' '+rep(Y)+' '+rep(Z)+' '+
    '='+' '+rep(X)+' '+rep(res1)+' '+rep(res2))

elif (selection==4):
    #fourth example New gate
    res1= txor(tand(X,Y),Z)
    res2= txor(tand(tnot(X),tnot(Z)),tnot(Y))
    print (rep(X)+' '+rep(Y)+' '+rep(Z)+' '+
    '='+' '+rep(X)+' '+rep(res1)+' '+rep(res2))

elif (selection==5):
    #fifth example NTF gate
    res1=txor(X,Y)
    res2=txor(tand(tnot(Y),Z),tand(X,tnot(Z)))
```

```
        res3=txor(tand(Y,Z),tand(X,tnot(Z)))
        print (rep(X)+' '+rep(Y)+' '+rep(Z)+' '+
        '='+' '+rep(res1)+' '+rep(res2)+' '+rep(res3))

    elif (selection==6):
        #sixth example Rotational gate
        result=trot(Z)
        print (rep(X)+' '+rep(Y)+' '+rep(Z)+' '+
        '='+' '+rep(X)+' '+rep(Y)+' '+rep(result))
```

A.2 Analiza rezultatov

V spodnjih tabelah so opredeljeni rezultati za logična vrata, ki so bila poleg Toffolijevih in Rotirnih vrat še testirana.

a b c = x y z	a b c = x y z	a b c = x y z
A A A = A A A	B A A = B A A	C A A = C A A
A A B = A A B	B A B = B B A	C A B = C C C
A A C = A A C	B A C = B C A	C A C = C C C
A B A = A B A	B B A = B A B	C B A = C C C
A B B = A B B	B B B = B B B	C B B = C C C
A B C = A B C	B B C = B C B	C B C = C C C
A C A = A C A	B C A = B A C	C C A = C C C
A C B = A C B	B C B = B B C	C C B = C C C
A C C = A C C	B C C = B C C	C C C = C C C

Tabela A.1 Tabela prikazuje izhod za Fredkinova vrata.

a b c = x y z	a b c = x y z	a b c = x y z
A A A = A A A	B A A = B B B	C A A = C C C
A A B = A A B	B A B = B B A	C A B = C C C
A A C = A A C	B A C = B B C	C A C = C C C
A B A = A B A	B B A = B A B	C B A = C C C
A B B = A B B	B B B = B A A	C B B = C C C
A B C = A B C	B B C = B A C	C B C = C C C
A C A = A C A	B C A = B C B	C C A = C C C
A C B = A C B	B C B = B C A	C C B = C C C
A C C = A C C	B C C = B C C	C C C = C C C

Tabela A.2 Tabela prikazuje izhod za Double Feynmanova vrata.

a b c = x y z	a b c = x y z	a b c = x y z
A A A = A A A	B A A = B B A	C A A = C C A
A A B = A A B	B A B = B B B	C A B = C C B
A A C = A A C	B A C = B B C	C A C = C C C
A B A = A B A	B B A = B A B	C B A = C C C
A B B = A B B	B B B = B A A	C B B = C C C
A B C = A B C	B B C = B A C	C B C = C C C
A C A = A C A	B C A = B C C	C C A = C C C
A C B = A C B	B C B = B C C	C C B = C C C
A C C = A C C	B C C = B C C	C C C = C C C

Tabela A.3 Tabela prikazuje izhod za Peresova vrata.

a b c = x y z	a b c = x y z	a b c = x y z
A A A = A A A	B A A = B A B	C A A = C A C
A A B = A B B	B A B = B B B	C A B = C B B
A A C = A C C	B A C = B C B	C A C = C C C
A B A = A A B	B B A = B B A	C B A = C C C
A B B = A B A	B B B = B A A	C B B = C C A
A B C = A C C	B B C = B C A	C B C = C C C
A C A = A A C	B C A = B C C	C C A = C C C
A C B = A B C	B C B = B C C	C C B = C C C
A C C = A C C	B C C = B C C	C C C = C C C

Tabela A.4 Tabela prikazuje izhod za New-vrata.

a b c = x y z	a b c = x y z	a b c = x y z
A A A = A A A	B A A = B B B	C A A = C C C
A A B = A B A	B A B = B B A	C A B = C B A
A A C = A C A	B A C = B C C	C A C = C C C
A B A = B A A	B B A = A B B	C B A = C C C
A B B = B A B	B B B = A A B	C B B = C A B
A B C = B A C	B B C = A C C	C B C = C C C
A C A = C A A	B C A = C B B	C C A = C C C
A C B = C C C	B C B = C C C	C C B = C C C
A C C = C C C	B C C = C C C	C C C = C C C

Tabela A.5 Tabela prikazuje izhod za NTF-vrata.