Seminar pri predmetu ONT: tQCA 1x2 RAM

Tomi Erlih, Nejc Ilc, Marko Tišler, Damjan Vidonja
Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, Ljubljana, Slovenija

Povzetek. Seminarska naloga obravnava kvantne celične avtomate, ki temeljijo na trojiški logiki. S pomočjo orodja za simulacijo kvantnih struktur *qdCAD* smo načrtovali in izdelali model trojiške pomnilne celice (WD pomnilna celica) ter ga nato nadgradili v delujoče RAM vezje, ki je sposobno hranjenja dveh tritov informacije. Podana je tudi primerjava razvite kvantne trojiške pomnilne celice z dvojiškimi izvedenkami.

Ključne besede. kvantni celični avtomat, trojiška logika, WD pomnilna celica, RAM pomnilnik

Kazalo.

Univerza v Ljubljani
Fakulteta
za računalništvo
in informatiko

1. Uvod

1.1. Cilji naloge

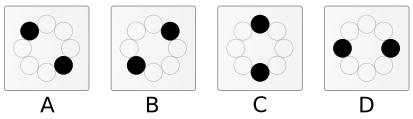
V okviru laboratorijskih vaj pri predmetu Optične in nanotehnologije v 4. letniku univerzitetnega študija na Fakulteti za računalništvo in informatiko v Ljubljani, smer Računalniški sistemi, smo dobili nalogo načrtovati in zgraditi pomnilno celico, ki bo delovala v trojiški logiki, torej bo sposobna hraniti 1 trit [1] informacije. Naloga je obsegala tudi uporabo te pomnilne celice za izvedbo 1x2 RAM vezja. Vse strukture naj bi bile realizirane kot trojiški kvantni celični avtomat - tQCA [2] .

1.2. Motivacija za trojiško logiko

Zakaj sploh trojiška logika? Na kratko zato, ker je osnova 2 premajhna, 10 prevelika, 3 pa je najbližje idealu. Idealen zapis je namreč ustrezno razmerje med dolžino zapisa (w) in velikostjo potencialne zaloge vrednosti številk (odvisna od osnove r). Torej iščemo $min\ (rw)$, pri konstantni vrednosti r^w. Analitični pristop nam ponuja realno rešitev e ; najbližje celo število pa je ravno 3 (več o tem v poglavju 3.4.1). Torej, trojiški zapis velja kot najefektivnejši in najekonomičnejši in to je dovolj velika motivacija za ukvarjanje z idejo o procesiranju v okviru trostanjske logike.

1.3. Osnove tQCA

Cilj je postaviti kvantne celične strukture, ki so zmožne ne samo dvovrednostnih, temveč tudi večvrednostnih logičnih operacij. Zakaj? S tehnološkega vidika je možno na isto površino celice postaviti več kot štiri kvantne pike, kolikor jih je v standardni QCA celici. Strukture s tako nadgrajeno QCA celico imenujemo razširjeni QCA (ang. Extended Quantum-Dot Cellular Automata - EQCA). Za realizacijo trojiške celice smo uporabili celico z osmimi pikami, še vedno pa imamo le dva elektrona. Taka celica ima štiri možna stanja: A, B, C in D (slika 1). A in B predstavljata logični stanji „0“ in „1“, C predstavlja „1/2“ (tretje stanje), D pa smo razglasili za vrednost, ki se nikoli ne pojavi na vhodu, lahko pa je prisotna znotraj notranje strukture. Ostalih stabilnih stanj ni mogoče doseči zaradi medsebojne odbojne sile elektronov.



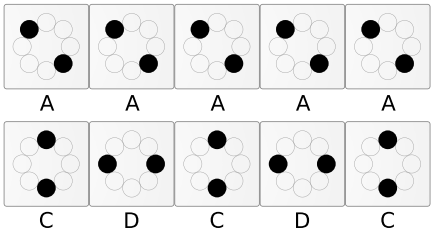
Slika 1. Osnovna stanja trojiške kvantne celice

1.4. Primitivi tQCA

Preden se posvetimo kompleksnejšim strukturam, si oglejmo osnove primitive, ki pomenijo izhodišče načrtovanja.

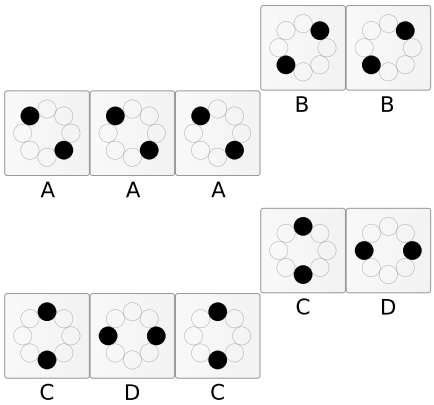
[1] $1\ trit\ =\ log_2 3\ bit.$
[2] ang. ternary Quantum Cellular Automata

- **Linija** – namenjena je prenosu signala. Stanji A in B se prenašata nespremenjena iz celice v celico, C in D pa alternirajoče, kar pomeni, da mora biti za pravilno delovanje linija lihe dolžine (slika 2).



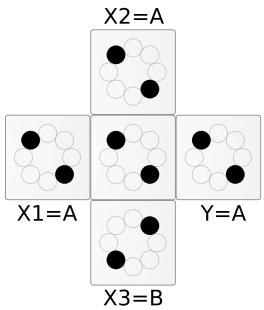
Slika 2. Primer prenosa po liniji

- **Negacija** – pri negaciji se A preslika v B in B v A. C in D se preneseta nespremenjeno (slika 3).



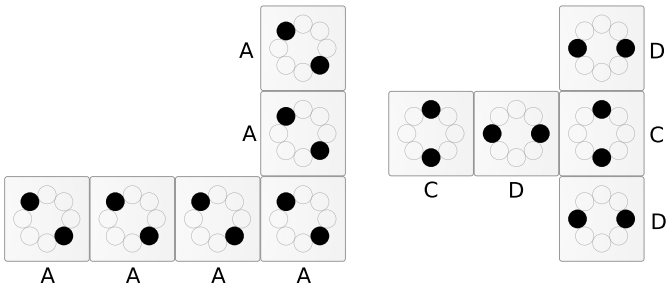
Slika 3. Primer negacije

- **Majoritetna vrata** – z majoritetnimi vrati je mogoče tvoriti AND in OR, oz. v trojiški logiki bolje rečeno MIN in MAX funkcijo (slika 4).



Slika 4. Primer majoritetnih vrat

- **Kotna linija** – S kotno linijo je mogoče spremeniti smer iz vodoravne v navpično in obratno (slika 5).
- **Odcep** – Z odcepom je možno „razmnožiti“ signal na dve liniji (slika 5).



Slika 5. Primer kotne linije (levo) in odcepa (desno)

Prikazani primitivi bi teoretično morali delovati, tudi če so vse celice na isti uri, vendar se to mišljenje v praksi izkaže za precej utopično. Rešitev problema stabilnosti z adiabatnim nadzorovanjem preklopa stanj [1] je opisana v poglavju 3.3.

2. Metode

2.1. Kvantna pomnilna celica

Pomnilna celica je struktura, ki mora, na tak ali drugačen način, hraniti podan podatek poljubno dolgo in hkrati omogočati njegovo branje. V današnji tranzistorski tehnologiji se podatek hrani v obliki količine naboja, ki se nato lahko tolmači kot logična „1“ ali „0“. Kvantne celične strukture take oblike hranjenja podatka ne poznajo, saj v njih obstajajo le trenutna stanja, katera lahko tolmačimo kot določene logične vrednosti. Stanje posamezne celice se nenehno spreminja glede na trenutno stanje vseh okoliških celic. Zaradi take dinamike kvantnih celičnih struktur, nam že krožna struktura, oziroma struktura s povratno povezavo, omogča pomnjenje želenega podatka.

2.1.1. WD pomnilna celica Z znanjem, ki smo ga pridobili tekom semestra, smo zgradili posebno vrsto pomnilne celice, kateri smo dali ime „WD pomnilna celica“. Vhodna signala sta „W/R“ in „Data“, zato po analogiji s poimenovanjem dvojiških pomnilnih celic RS („Reset“ in „Set“) in JK („Jump“ in „Kill“), inicialki signalov tvorita ime. WD pomnilna celica je sposobna operirati z 1 tritom informacije, torej podatkom v trojiškem zapisu.

2.1.2. Logična shema WD pomnilne celice Izdelana WD pomnilna celica omogoča vse želene operacije - pisanje podatkov in njih poljubno dolgo hranjenje ter branje.

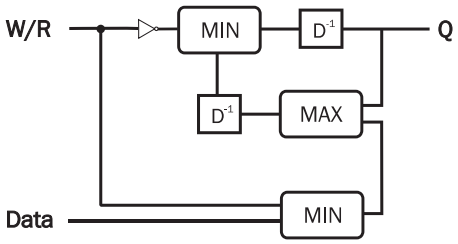
Pisanje deluje po principu: uniči trenutni notranji podatek in ga nato prepiši z želenim drugim podatkom. Hranjenje podatka je zagotovljeno s kroženjem podatka znotraj pomnilne celice. Branje celice deluje tako, da se v vsakem obhodu podatka skozi „krog“, proti izhodu pošlje hranjeni podatek, ki izhod doseže le v primeru, da je

trenutna operacija *branje*. Opisani princip ni primeren zgolj za spodaj opisano trojiško logiko, ampak tudi za poljubno večvrednostno logiko.

Na sliki 6 je prikazana logična shema delovanja WD pomnilne celice. Za prikaz so uporabljeni splošni bloki z napisi in pomenijo logično funkcijo majoritetnih vrat (MIN oziroma MAX). V funkciji negatorja je simbol vzet iz tranzistorske logike, kar morda na prvi pogled ni smiselno, saj tu ni govora o tovrstni tehnologiji. Vendar če upoštevamo dejstvo, da je signal W/R lahko po vrednosti le A ali B, torej prevedeno v dvojiški jezik le „0“ ali „1“, je funkcija trojiškega negatorja ekvivalentna tistemu iz dvojiškega sveta. Torej:

$$\neg\, A \;=\; B$$

$$\neg\, B \;=\; A\;.$$



Slika 6. Logična shema delovanja WD pomnilne celice

2.1.3. Enačba pomnjenja Zgoraj opisano in prikazano delovanje še najlažje opišemo s sledečo enačbo:

$$Q \;=\; max(min(Data,\; W/R),\; min(Q(t-1),\; \neg W/R))$$

Iz enačbe je razvidno delovanje pomnilne celice, ki ga bomo sedaj poskušali na kratko ubesediti. Majoritetna vrata, ki na vhodu sprejmejo signal W/R in $Data$, so konfigurirana kot funkcija MIN, kar pomeni: če je $W/R = A$, se vrši operacija branja in se signal $Data$ „uniči“, postavi na A, da ne moti predhodno shranjene vrednosti. Če pa je signal $W/R = B$, se podatek na $Data$ prenese naprej.

Vloga negatorja signala W/R je trivialna - vrednost A preslika v B in obratno. Negiran signal W/R v primeru operacije pisanja v naslednjih majoritetnih vratih s funkcijo MIN pobriše trenutno shranjeno vrednost in tako pripravi celico na vpis nove čez en urin cikel. Ta vpis se zgodi v majoritetnih vratih s funkcijo MAX.

Delovanje WD pomnilne celice je podano tudi v tabeli 1 (oznaka „X“ pomeni poljubno vrednost).

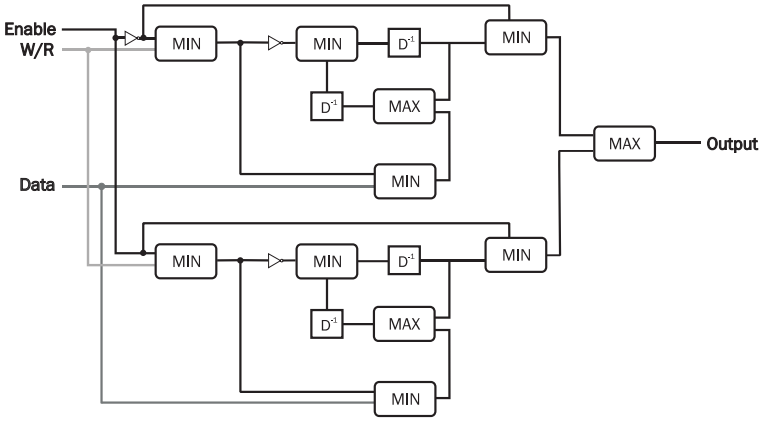
2.2. Kvantni 1x2 RAM

Od WD pomnilne celice je samo še korak do kompleksnejšega RAM pomnilnega vezja, vsebujoč kontrolo vpisa oziroma branja - signal *Enable*.

Operacija	W/R	$Data$	$Q(t)$
Branje	A	X	Q(t-1)
Pisanje A	B	A	A
Pisanje B	B	B	B
Pisanje C	B	C	C

Tabela 1. Delovanje WD pomnilne celice

2.2.1. Logična shema 1x2 RAM vezja Za hranjenje 2 trit-ov informacije potrebujemo dve WD pomnilni celici in ustrezni signal, ki bo določal aktivno celico - tisto, v katero se piše ali se iz nje bere. To funkcijo ima signal *Enable*: če je $Enable = A$, je aktivna prva (zgornja) celica, drugače pa druga (spodnja). Logična shema je podana na sliki 7.



Slika 7. Logična shema delovanja 1x2 RAM vezja

Samo delovanje je identično delovanju samostojne WD celice s to razliko, da s signalom *Enable* nadzorujemo aktivnost celice.

2.2.2. Enačba delovanja Izhod posamezne pomnilne celice zaradi preglednosti označimo z Q_1 in Q_2. Logične enačbe se zato glasijo:

$$Q_1 = max(min(Data, min(\neg Enable,\ W/R)),\ min(Q(t-1), \neg min(\neg Enable,\ W/R)))$$

$$Q_2 = max(min(Data,\ min(Enable,\ W/R)),\ min(Q(t-1),\ \neg min(Enable,\ W/R)))$$

$$Output = max(\ min(\neg Enable,\ Q_1),\ min(Enable,\ Q_2))$$

2.3. qdCAD

Programski paket *qdCAD* sestavljata programa *qdCAD_sim* in *qdCAD_plot*, ki skupaj omogočata vizualno simuliranje delovanja kvantnih celičnih struktur. Aplikacijo so razvili v Laboratoriju za računalniške strukture in sisteme na Fakulteti za računalništvo in informatiko [2].

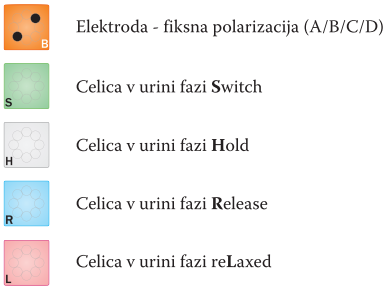
2.3.1. qdCAD_sim Program *qdCAD_sim* je program, ki je izključno namenjen izračunu trenutnega stanja vsake posamezne celice kvantne celične strukture, ki jo želimo simulirati. Program nima grafičnega vmesnika in se zato poganja iz ukazne vrstice. Z njim upravljamo s pomočjo vhodne konfiguracijske datoteke, v kateri definiramo celotno kvantno celično strukturo, natančnost izračunov ter sekvenco vhodnih testnih vektorjev, nad katerimi želimo testirati podano strukturo. Program nam v konzoli, tekom izvajanja simulacije, izpisuje izhodne vrednosti za vsak posamezen vhodni vektor, hkrati pa beleži natačen potek simulacije v izhodno datoteko. Iz nje je podroben potek simulacije težko razbrati, zaradi ogromne količine podatkov. Za boljšo predstavo o poteku simulacije zato uporabimo program *qdCAD_plot*.

2.3.2. qdCAD_plot Program *qdCAD_plot* je namenjen grafični predstavitvi poteka simulacije. V njegovem oknu se izriše definirana celična struktura ter njeno trenutno stanje. Izris poteka simulacije je sočasen s samo simulacijo, saj se na progam *qdCAD_plot* poveže program *qdCAD_sim* in mu tako lahko podaja aktualne izračune simulacije. Program ima vgrajeno funkcionalnost, ki omogoča enostavno povečevanje in zmanjševanje izrisanega, kar lahko izkoristimo za bolj natančno opazovanje določenega dela strukture oziroma boljši pregled nad celotno strukturo.

3. Rezultati in diskusija

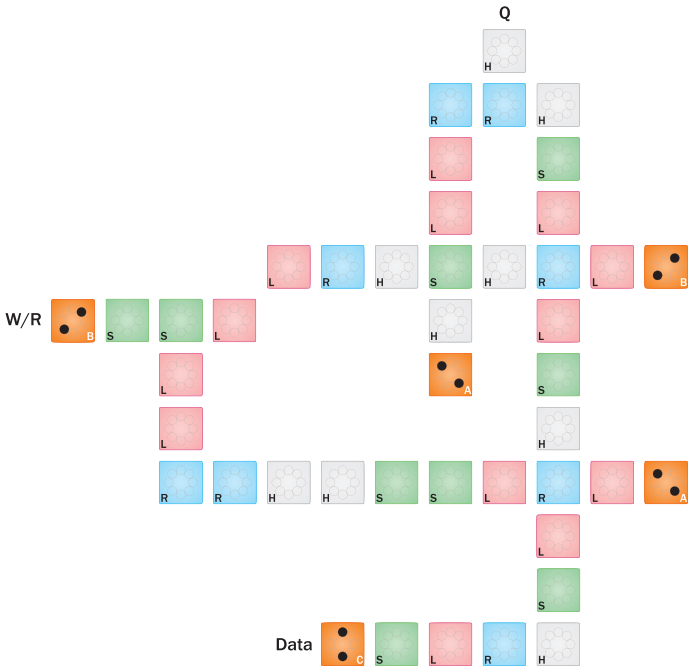
3.1. Modeli tQCA vezij

Za načrtovanje in simulacijo trojiških kvantnih celičnih struktur smo uporabili simulator qdCAD (glej poglavje 2.3). Za prikazovanje modelov zgrajenih struktur smo definirali osnovne gradnike - kvantne celice - ki so prikazani in opisani na sliki 8.



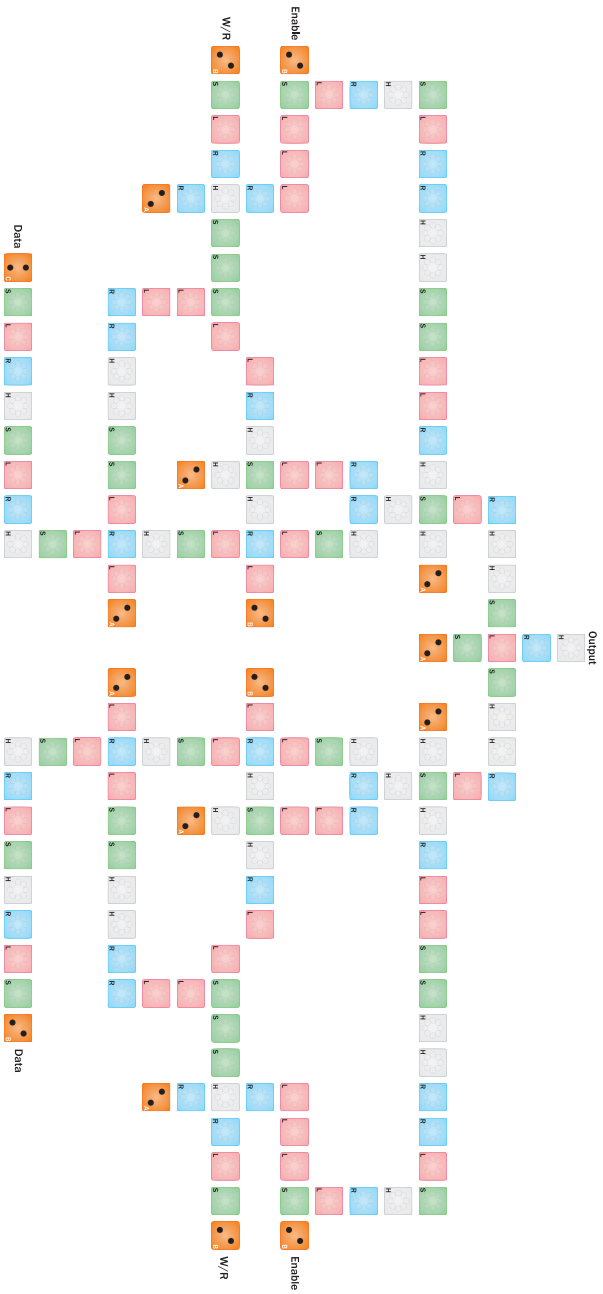
Slika 8. Pomen kvantnih celičnih struktur uporabljenih v modelu trojiške pomnilne celice.

3.1.1. Model WD pomnilne celice Slika 9 prikazuje model WD pomnilne celice, ki služi kot osnova za pomnejnje 1 trit-a informacije (vrednosti A,B,C).



Slika 9. Trojiška WD pomnilna celica.

3.1.2. Model tQCA 1x2 RAM vezja Z nadgradnjo osnovne WD pomnilne celice s signalom *En* (Enable), smo zgradili kompleksnejše vezje, ki opravlja funkcijo trojiškega 1x2 RAM pomnilnika, prikazanega na sliki 10. Velja omeniti, da smo zaradi poenostavitve problema časovne potratnosti simuliranja delovanja te strukture, njeno velikost zmanjšali z uporabo podvajanja vhodnih signalov *Data*, *W/R* in *Enable*.



Slika 10. tQCA 1x2 RAM vezje

3.2. Rezultati simulacij

3.2.1. Simulacije Pravilnost delovanja WD pomnilne celice smo preverili s pomočjo simulacij, ki s pomočjo natančnih matematičnih izračunov podajo rezultat, na podlagi katerega lahko sklepamo o korektnosti delovanja strukture. Simulacije smo izvajali z različnimi vhodnimi vektorji (nabori spremenljivk) in tako preverili pravilno delovanje v predvidenih robnih primerih. Rezultati nekaterih simulacij so navedeni v nadaljevanju. Podani so s tabelo in grafom izvajanja simulacije. V tabeli so opisani posamezni vhodni vektorji, ki se izvedejo ob natanko dolčenem koraku ure. Za lažjo vizualno predstavo delovanja je podan tudi graf simulacije. Vsaka komponenta vhodnega vektorja iz tabele je v grafu predstavljena kot vhodni signal, ki se s časom spreminja tako, kot je navedeno v tabeli. V katerem koraku simulacije se trenutno nahajamo, lahko razberemo s pomočjo grafa adiabatnega preklopa ali „ure“. Ker imajo strukture zakasnitve nekaj urinih period, moramo to upoštevati ob odčitavanju vrednosti izhodnega signala. Za lažjo orientacijo je izhodnemu signalu dodano zamaknjeno številčenje urinih period. Izhodni signal se od vhodnih signalov in ure razlikuje v tem, da ni povezan. To pa zato, ker so stanja celic satabilna samo, ko se nahajajo v $HOLD$ fazi ure (označeno s črko H v grafu).

3.2.2. Simulacija WD pomnilne celice Zaporedje elektrod na vhodu v vezje pomnilne celice je:

$$W/R,\ B,\ A,\ A,\ Data.$$

Vrednosti A in B predstavljajo konstantne vrednosti - elektrode na majoritetnih vratih, ki jim definirajo funkcijo (MIN ali MAX). Rezultati so razvidni iz tabele 2 in grafa na sliki 11. Pisanje vrednosti v celico potrebuje 2 polna urina cikla iz razlogov, omenjenih v poglavju 3.3.

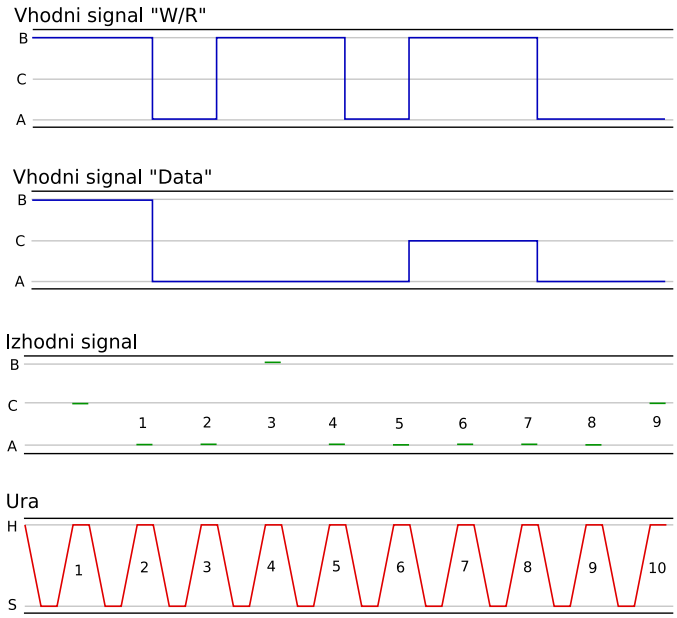
Cikel	Operacija	W/R	$Data$	Q
1	Pisanje B	B	B	C
2	Pisanje B	B	B	A
3	Branje	A	A	A
4	Pisanje A	B	A	B
5	Pisanje A	B	A	A
6	Branje	A	A	A
7	Pisanje C	B	C	A
8	Pisanje C	B	C	A
9	Branje	A	A	A
10	Branje (slepo)	A	A	C

Tabela 2. Simulacija WD pomnilne celice

3.2.3. Simulacija 1x2 RAM vezja Zaporedje elektrod na vhodu v RAM vezje je sledeče:

$$W/R_0,\ En_0,\ A,\ A,\ B,\ A,\ A,\ Data_0,\ W/R_1,\ En_1,\ A,\ A,\ B,\ A,\ A,\ Data_1,\ B.$$

Rezultati dveh simulacij so razvidni iz tabel 3 in 4 in grafov na slikah 12 in 13. 1x2 RAM vezje vsebuje dve celici - označimo ju kot celica 1 in celica 2 in niti ni pomembno, kje, relativno ena na drugo, se v resnici nahajata.



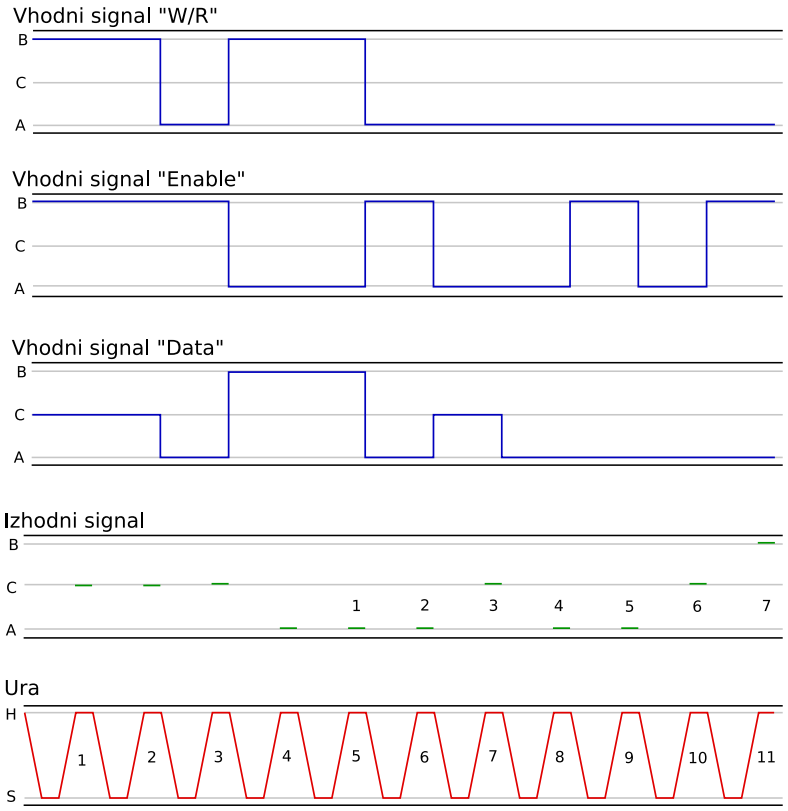
Slika 11. Graf simulacije WD pomnilne celice

Cikel	Operacija	W/R_0	En_0	$Data_0$	W/R_1	En_1	$Data_1$	Q
1	Pisanje C v celico 1	B	B	C	B	A	C	C
2	Pisanje C v celico 1	B	B	C	B	A	C	C
3	Branje iz celice 1	A	B	A	A	A	A	C
4	Pisanje B v celico 2	B	A	B	B	B	B	A
5	Pisanje B v celico 2	B	A	B	B	B	B	A
6	Branje iz celice 1	A	B	A	A	A	A	A
7	Branje iz celice 2	A	A	C	A	B	C	C
8	Slepi cikel	A	A	A	A	B	A	A
9	Slepi cikel	A	B	A	A	A	A	A
10	Slepi cikel	A	A	A	A	B	A	C
11	Slepi cikel	A	B	A	A	A	A	B

Tabela 3. 1. simulacija 1x2 RAM vezja

3.3. Omejitve in izkušnje pri načrtovanju

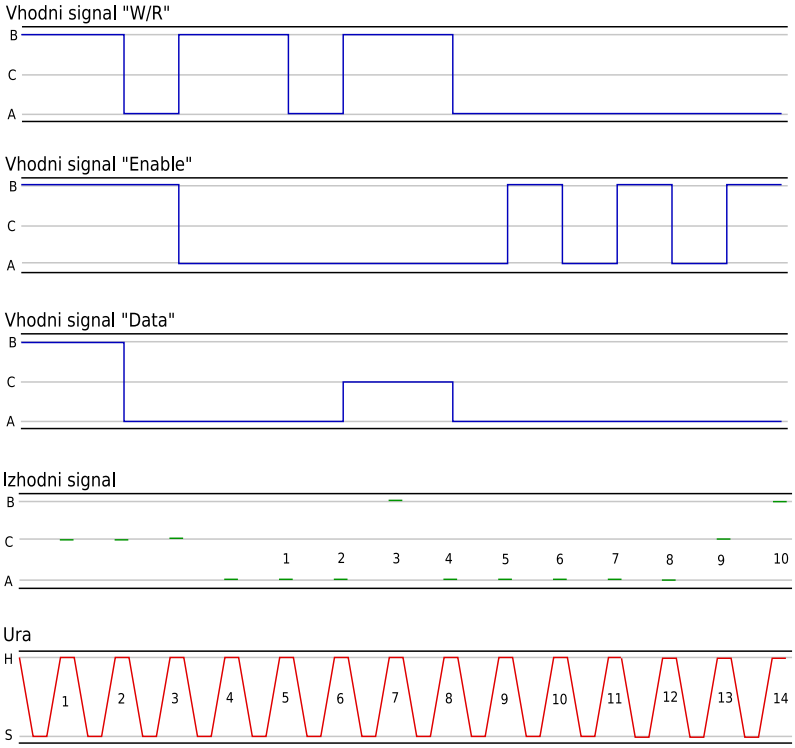
Tekom načrtovanja vezja smo naleteli na kar nekaj omejitev in težav, ki so vodile v nestabilno ali napačno delovanje vezja. Večino teh problemov smo rešili z uporabo dodatnih urinih period, kar pa je doprineslo k še večjim zakasnitvam vezja. Problem se je pojavil že pri dodajanju linij, saj postane linija s petimi ali več celicami vezanimi na isto urino periodo že nestabilna in ne prenese vseh vrednosti pravilno. Zaradi tega v vezju nismo uporabili več kot treh zaporednih celic vezanih na isto periodo ure. Druga omejitev so odcepi (ang. fan-out), kjer mora prva celica novega odcepa iz linije preiti na naslednjo urino periodo v nasprotnem primeru prenos vrednosti ne deluje pravilno. Enako kot za odcepe velja tudi za realizacijo negatorja.



Slika 12. Graf 1. simulacije 1x2 RAM vezja

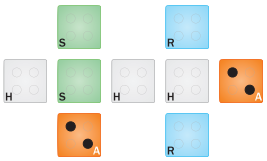
Cikel	Operacija	W/R_0	En_0	$Data_0$	W/R_1	En_1	$Data_1$	Q
1	Pisanje B v celico 1	B	B	B	B	A	B	C
2	Pisanje B v celico 1	B	B	B	B	A	B	C
3	Branje iz celice 1	A	B	A	A	A	A	C
4	Pisanje A v celico 2	B	A	A	B	B	A	A
5	Pisanje A v celico 2	B	A	A	B	B	A	A
6	Branje iz celice 2	A	A	A	A	B	A	A
7	Pisanje C v celico 2	B	A	C	B	B	C	B
8	Pisanje C v celico 2	B	A	C	B	B	C	A
9	Branje iz celice 2	A	A	A	A	B	A	A
10	Branje iz celice 1	A	B	A	A	A	A	A
11	Slepi cikel	A	A	A	A	B	A	A
12	Slepi cikel	A	B	A	A	A	A	A
13	Slepi cikel	A	A	A	A	B	A	C
14	Slepi cikel	A	B	A	A	A	A	B

Tabela 4. 2. simulacija 1x2 RAM vezja

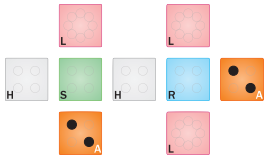


Slika 13. Graf 2. simulacije 1x2 RAM vezja

Na začetku smo pomnilno celico načrtovali kot vezje, ki za operacijo branja in pisanja porabi en sam cikel ure, a se je ta realizacija izkazala za neizvedljivo. Majoritetna vrata, ki so ključni element celotnega vezja, namreč za pravilno delovanje potrebujejo kar tri urine periode. Ker mora biti pomnilna struktura krožna, pomeni, da bo imela tudi najmanj dva odcepa, s čimer pa smo že prekoračili dolžino enega urinega cikla. Na začetku smo ta problem poskušali rešiti tako, da smo število urinih period, ki jih porabijo majoritetna vrata, zmanjšali za eno s tem, da smo centralno in izhodno celico vezali na isto periodo ure (slika 14). Taka struktura sama sicer deluje, a če je izhod vezan na vhod novih majoritetnih vrat, kar je bila v našem primeru edino možna realizacija, struktura ne deluje več pravilno. Zaradi vseh teh omejitev trajata operaciji branja in pisanja v pomnilno celico dva cikla ure, kar je posledica strukture na sliki 15.



Slika 14. Nedelujoča strukutura dveh majoritetnih vrat



Slika 15. Delujoča strukutura dveh majoritetnih vrat

3.4. Primerjava dvojiške in trojiške pomnilne celice

V tem razdelku bomo podali primerjavo med dvojiško in trojiško pomnilno celico v smislu zmogljivosti pomnjenja, porabe prostora in zakasnitve. Skušali bomo tudi odgovoriti na vprašanje, ali je „flip-flop“-u mogoče res odbila zadnja ura.

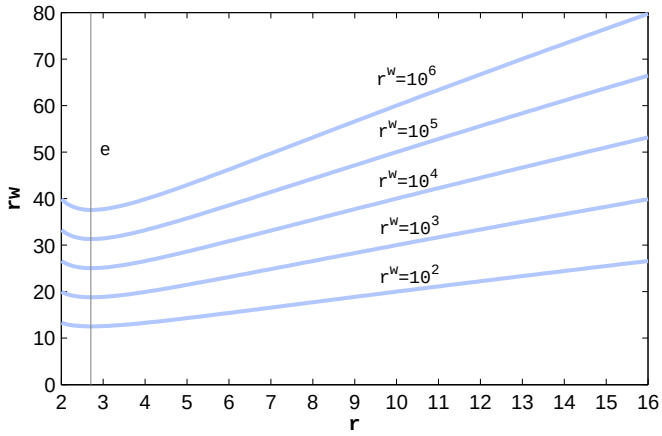
3.4.1. Zakaj trojiški zapis? Marsikdo gleda na uporabo dvojiškega zapisa in dvojiške logike v računalništvu kot nekaj samoumevnega in kar je še huje, kot nekaj najboljšega. Da to ne drži, je znano že dovolj dolgo, da je bil v nekdanji Sovjetski zvezi narejen računalnik Setun [3], delujoč na osnovah trojiške logike. Mnogo razprav in knjig je bilo napisanih na temo koristnosti in učinkovitosti trojiškega zapisa, zato bomo v nadaljevanju zgolj povzeli bistvene ideje.

Mogoče ste se že kdaj vprašali, zakaj ljudje uporabljamo desetiški sistem, računalniki pa dvojiškega. Razlaga je enostavna: ljudje imamo deset prstov, s katerimi si pomagamo pri štetju in nam je tak sistem najnaravnejši. Računalniki (vsaj večina današnjih) pa ima drugačen razlog - zanesljivost. Zaradi šuma je bilo namreč težko narediti elektronske elemente z več nivoji napetosti. Tako sta dve stanji, „0“ in „1“, vse kar danes „vidijo“ računalniki. Pa je tako res najbolje?

Če želimo najti najkompaktnejši in posledično najučinkovitejši zapis vrednosti, si moramo najprej ogledati, kako so števila sploh predstavljena. V splošnem lahko vsako število v poljubnem številskem sistemu zapišemo v obliki:

$$d_{n-1}r^{n-1} + \ldots + d_3r^3 + d_2r^2 + d_1r^1 + d_0r^0 \ ,$$

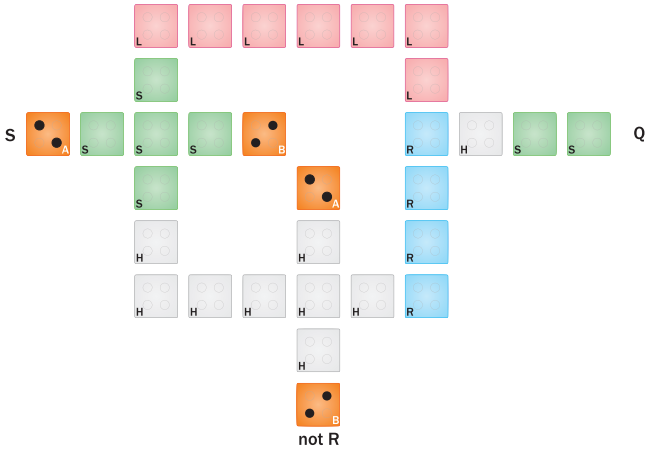
kjer je n število števk, r osnova („base“ ali „radix“), navadno pozitivno celo število, in d_i števke v zapisu števila. Vsak r določa število števk, ki jih potrebujemo za zapis neke vrednosti števila. Kot primer poglejmo zapis števila 6: 110 dvojiško, 20 trojiško. Število števk, ki jih potrebujemo za zapis nekega števila, bomo označili kot širina w. Torej, za dosego maksimalne kompaktnosti zapisa želimo čim manjši produkt rw pri konstantni vrednosti r^w. Analitično je pokazano, da je optimalna osnova število e, najbližje celo število pa 3 [4]. To nazorno prikazuje graf na sliki 16.



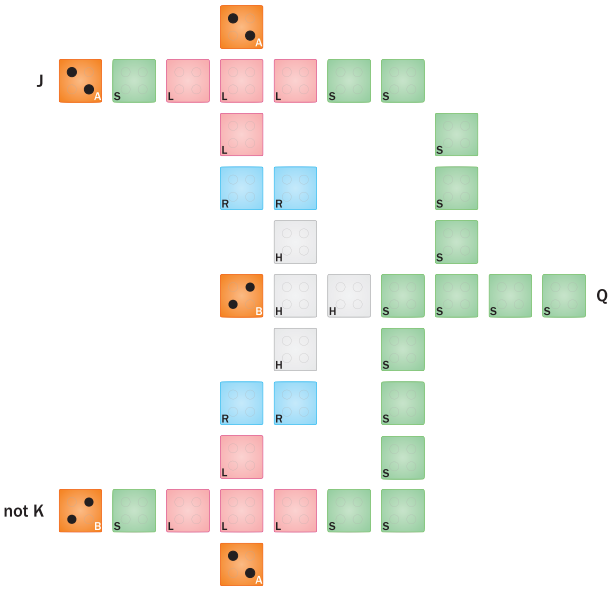
Slika 16. Število e je najbolj optimalna osnova v smislu kompaktnosti zapisa, najbližje celo število pa je 3.

3.4.2. Dvojiška pomnilna celica Z orodjem QCADesigner [5] smo zgradili in simulirali delovanje dveh tipov dvojiške pomnilne celice: RS in JK. Njuna struktura je prikazana na slikah 18 in 19.

Slika 17. Pomen kvantnih celičnih struktur uporabljenih v modelih dvojiških pomnilnih celic.



Slika 18. RS pomnilna celica kot binarni kvantni celični avtomat.



Slika 19. JK pomnilna celica kot binarni kvantni celični avtomat.

3.4.3. Trojiška pomnilna celica V primerjavi z dvojiškima pomnilnima celicama bo nastopala WD pomnilna celica, ki smo jo načrtovali in zgradili v okviru te seminarske naloge (slika 9).

3.4.4. Primerjava karakteristik pomnilnih celic Sedaj, ko imamo implementirane različne tipe pomnilnih celic tako v dvojiški (RS in JK) kot tudi v trojiški obliki (WD celica), poglejmo njihove lastnosti in poskušajmo na podlagi vzporedne primerjave določiti „zmagovalko pomnjenja“. Kriteriji za ocenjevanje so povezani z omejitvami

pri načrtovanju. Zanima nas, kako dobro smo minimizirali število uporabljenih celic, kako veliko površino smo pri tem zasedli, kakšna je zakasnitev vezja in kar je najpomembnejše - kakšen je izkoristek prostora glede na moč informacije, ki jo celica nosi. Rezultati primerjave so zbrani v tabeli 5.

	tip celice		
	RS	JK	WD
število celic	31	37	44
površina	88	110	144
zakasnitev (ciklov)	1	1	2
koncentracija celic	0.3523	0.3364	0.3056
bitov/celico	0.03226	0.02703	0.03602

Tabela 5. Primerjava pomnilnih celic

Število celic pomeni število uporabljenih kvantnih celic za realizacijo vezja. Vidimo, da je tu RS celica boljša od JK in WD. Dejstvo je, da je ta tip celice najenostavnejši tudi v logičnem smislu, saj denimo kombinacija $R = S = 1$ na vhodu ne vrši funkcije inverza hranjene vrednosti kot pri JK celici, temveč vsiljuje logično 1.

Kriterij *površina* pomeni produkt višine in širine vezja oziroma število celic, ki so tako ali drugače porabljene in niso na razpolago za ostale strukture. Zmagovalka v tej katogoriji je zopet RS celica. Razlog za to je razviden iz same konstrukcije celice, saj je zelo kompaktna in zaokrožena.

Naslednja kategorija, *koncentracija celic*, povezuje prvi dve v njunem razmerju, ki pomeni odstotek „koristnih“ celic v primerjavi z vsem zasedenim prostorom. „Koristnost“ pomeni, da je na določenem delu površine kvantna celica in ne prazen prostor. Iz podobnih razlogov kot prej je RS celica tista, ki ima najbolj ugodno razmerje oziroma najvišjo koncentracijo celic na površino.

Pri načrtovanju vezij je treba biti pozoren na razne omejitve, ki zagotavljajo stabilnost strukture. Več o tem je napisanega v poglavju 3.3. Iz tega tudi izhaja večja zakasnitev WD celice (2 periodi), saj je bilo treba za potrebe stabilnosti intenzivneje uporabiti principe adiabatnega preklapljanja oziroma „ure“ [1]. Vezje potrebuje 2 urini periodi za branje hranjene vrednosti, oziroma da je na izhodu vidna ustrezna posledica vhoda. Če upoštevamo dejstvo, da pisanje potrebuje 2 zaporedna ukaza (2 cikla ure), je zakasnitev primerno večja - 4 urine cikle. RS in JK celica sta v tem razdelku zato boljši, saj imata zgolj 1 periodo zakasnitve.

Do sedaj imamo nespornega zmagovalca odprtega prvenstva v kvantnem pomnjenju: RS celica! Je bil torej ves trud okoli vpeljave trojiške logike v kvantne celične avtomate zaman? Zavoljo zadnjega kriterija primerjave vendarle ne. Glavna prednost trojiškega zapisa je ravno v njegovi kompaktnosti, ki tu pride v ospredje. Bistveno pri načrtovanju pomnilnih elementov je spraviti čimveč informacije na enoto površine. Količino informacije v primeru dvojiške logike merimo v bitih, za potrebe trojiške logike pa se je izoblikoval izraz *trit*. Relacija med bitom in tritom je naslednja:

$$1\ trit\ =\ log_2 3\ bit.$$

Če torej vse prevedemo na bite in dobljeno delimo s številom celic v izvedbi, vidimo, da je tu WD celica najboljša! Torej, čeprav je dvojiška RS celica veliko manjša, lahko trojiška WD clica hrani za 11,66 % večje število bitov na porabljeno kvantno celico.

Cena, ki jo za to plačamo je večja zakasnitev, ki pa se pozna samo na začetku delovanja vezja. Takoj zatem je namreč izhod aktualen vsako periodo ure.

Predvidevamo lahko, da se bo stabilnost kvantnih celičnih struktur s časom povečevala in posledično bo možno zgraditi trojiško pomnilno celico z manjšo zakasnitvijo, kar bi lahko pomenilo zaton dosedanjih „flip-flop“-ov in vzpon „flip-flap-flop“-ov, kot je napovedal že gospod Donald Knuth skoraj 40 let nazaj [6].

4. Zaključek

V zgodovini računalništva je pod nesporno „diktaturo“ dvojiške logike pogosto tlela ideja po nečem učinkovitejšem - po trojiškem. Poleg tega so se rodile zamisli o drugačnem načinu gradnje računalniških sistemov. Ena izmed alternativ - kvantno računalništvo - je postala oprijemljiva in dovzetna za uporabo večvrednostne logike. Iz kvantnih celičnih avtomatov z dvema stabilnima stanjema so se razvili taki s tremi, čakajoč na potrditev svoje uporabnosti in naš cilj je bil ravno to - pokazati, da je s pazljivim in vztrajnim načrtovanjem celičnih struktur mogoče zgraditi delujočo pomnilno celico, sposobno vpisa in hranjenja enega trita informacije ter jo uporabiti kot del večje RAM strukture.

Ne samo da je v tem poročilu predstavljena in opisana WD pomnilna celica (in njena nadgradnja v 1x2 RAM vezje) dokaz o pravilnosti delovanja tQCA struktur, temveč tudi podpira predvidevanja, da ima trojiška logika v kontekstu alternativnih platform procesiranja svetlo prihodnost. V primerjavi z dvojiškimi pomnilnimi celicami namreč lahko WD celica hrani za 11,66 % več informacije na uporabljeno celico, kar je gotovo obetaven začetek.

Literatura

[1] P. Pečar, Uporaba adiabatnega pristopa pri realizaciji trojiškega procesiranja na osnovi kvantnih celičnih avtomatov, MSc thesis, Univerza v Ljubljani, Fakulteta za računalništvo in informatiko (2007).

[2] http://lrss.fri.uni-lj.si.

[3] http://www.computer-museum.ru/english/setun.htm.

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

[5] http://www.qcadesigner.ca.

[6] D. E. Knuth, The art of computer programming, Vol. 2, Addison-Wesley, 1969.