Pomnenje s kvantnimi celičnimi avtomati

Jasmina Pegan, Blaž Rojc

November 24, 2019

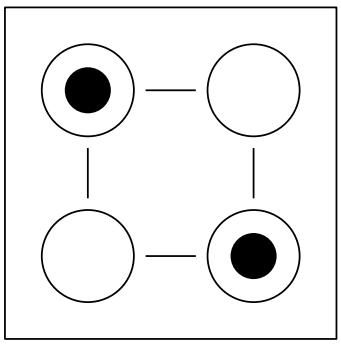
1 Uvod

Kvantne celice, osnovni gradniki kvantnih celičnih avtomatov, same po sebi ne omogočajo pomnenja [2]. Podobno kot pri tranzistorjih mora načrtovalec digitalnega vezja sestaviti strukturo iz kvantnih celic, ki pomnenje omogoči. V tem delu bomo predstavili principe pomnenja v kvantnih celičnih avtomatih in nekatere strukture iz kvantnih celic, ki simulirajo delovanje tradicionalnih pomnilnih celic.

2 Pregled področja

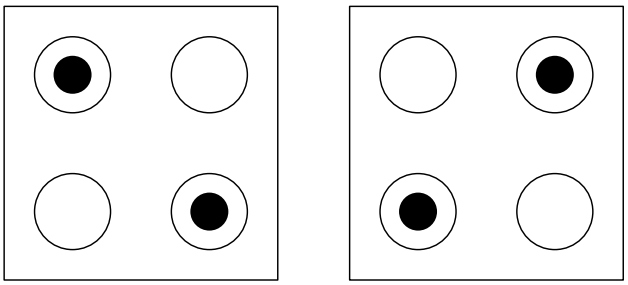
2.1 Kvantni celični avtomati

Kvantni celični avtomati predstavljajo izvedbo celičnih avtomatov, v katerih so osnovni gradniki kvantne celice. Kvantna celica je konstrukt kvadratne oblike, ki vsebuje štiri okrogle kvantne pike in štiri tunele. Vsak tunel povezuje dve sosednji kvantni piki. Shema takšne celice je prikazana na sliki 1.



Slika 1: Shema kvantne celice. Poleg štirih kvantnih pik in tunelov sta prikazana še dva elektrona, ki se nahajata v piki levo zgoraj in v piki desno spodaj.

V kvantni celici sta ujeta dva elektrona. Vsak od njiju se lahko nahaja v eni izmed štirih kvantnih pik, med katerimi se lahko pomika prek tunelov, ki jih povezujejo. Elektrona se zaradi odbojnih sil med njima v odsotnosti zunanjih vplivov postavita v eno izmed dveh stabilnih stanj, prikazanih na sliki 2.

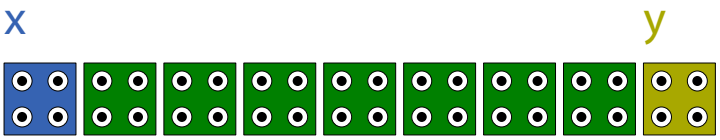


Slika 2: Stabilni stanji kvantne celice. Na levi je prikazano stanje $P = -1$, na desni pa stanje $P = 1$.

Stanje $P = -1$ interpretiramo kot logično vrednost 0, stanje $P = 1$ pa kot logično vrednost 1 [3]. Tuneli med pikami so opremljeni s pregradami. Elektrona lahko prosto prehajata med pikami, dokler pregrade niso vzpostavljene. Vzpostavitev pregrad prehajanje elektronov onemogoči, kar nam omogoča, da stanje celice odčitamo. Take celice označujemo kot izhodne. Celico lahko tudi prisilimo, da zasede določeno stanje. Take celice označujemo kot vhodne.

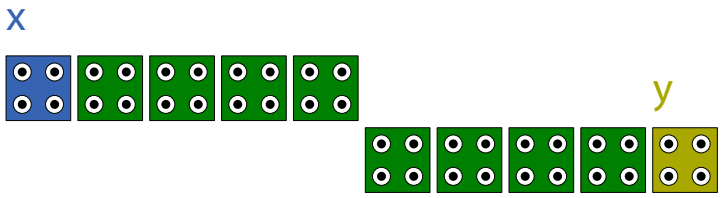
Kvantni celični avtomat je konstrukt, ki vsebuje eno ali več kvantnih celic, ki zaradi medsebojne bližine interaktirajo druga z drugo. Sile med elektroni težijo k vzpostavitvi stanja z najmanjšo skupno energijo. V odsotnosti zunanjih sil se kvantna celica postavi v eno izmed dveh stabilnih stanj z verjetnostjo $\frac{1}{2}$, ampak če so v njeni bližini druge kvantne celice, se ta verjetnost spremeni. S pravilno postavitvijo celic lahko dosežemo, da se avtomat obnaša kot tradicionalna Boolova preklopna funkcija.

Gradnjo kompleksnejših struktur pričnemo z osnovnimi gradniki. Osnovni gradniki predstavljajo poln funkcijski nabor, s katerim lahko implementiramo poljubno preklopno funkcijo. Najenostavnejši gradnik je vodilo, prikazano na sliki 3.



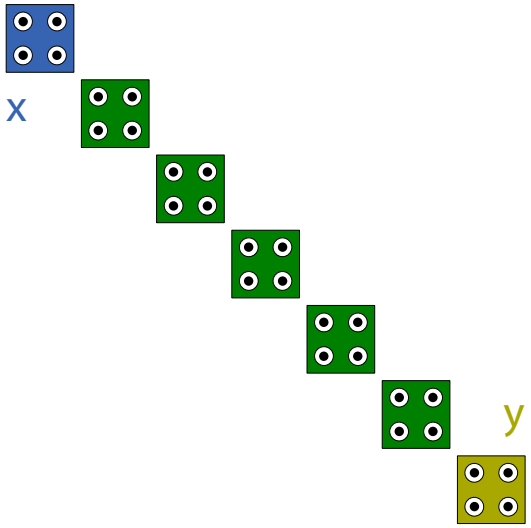
Slika 3: Vodilo. Signal potuje od vhoda x do izhoda y.

Vodila preslikajo vhodno vrednost na izhod. Uporabljamo jih za povezovanje gradnikov v avtomatu. Drugi gradnik je negator, prikazan na sliki 4.



Slika 4: Negator. Signal potuje od vhoda x do izhoda y in je pri tem negiran.

Negatorje lahko uporabimo tudi kot diagonalno vodilo, kjer je dolžina takega vodila sodo število. Primer take uporabe je prikazan na sliki 5.

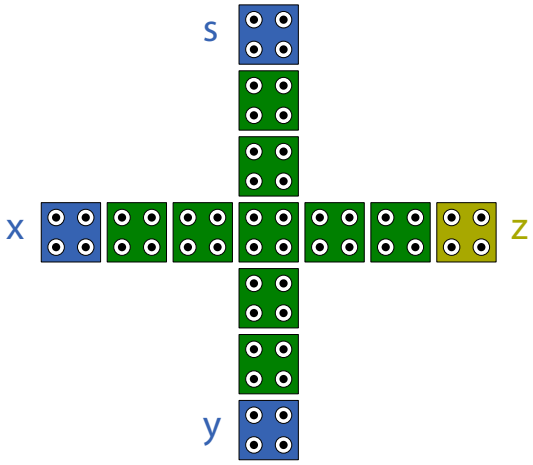


Slika 5: Vodilo iz negatorjev. Signal potuje od vhoda x do izhoda y. Pri tem je negiran $2k$-krat, zato se logična vrednost ne spremeni.

Zadnji osnovni gradnik so večinska oz. majoritetna vrata, prikazana na sliki 6.

Izhod majoritetnih vrat zavzame večinsko vrednost na treh vhodih. Z vezavo konstante na enega izmed vhodov majoritetnih vrat dobimo gradnik, ki predstavlja konjunkcijo ali disjunkcijo, kot je prikazano na sliki 7.

Tok podatkov v avtomatu usmerjamo s pomočjo urinih con. Vsaki celici dodelimo eno izmed štirih urinih con, ki se ciklično izmenjujejo. Tako omogočimo logično separacijo delov avtomata in način interakcije med njimi.



Slika 6: Majoritetna vrata. Izhod z zavzame večinsko vrednost na vhodih s, x in y.

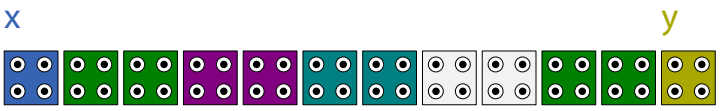
s	x	y	z	
0	0	0	0	$AND(x,y)$
0	0	1	0	
0	1	0	0	
0	1	1	1	
1	0	0	0	$OR(x,y)$
1	0	1	1	
1	1	0	1	
1	1	1	1	

Slika 7: Tabela vrednosti izhoda majoritetnih vrat pri različnih vhodnih vrednostih. Majoritetna vrata se glede na x in y pri $s = 0$ obnašajo kot konjunkcija, pri $s = 1$ pa kot disjunkcija.

2.2 Problematika pomnenja

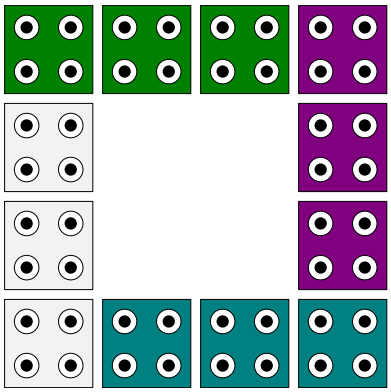
Ko je ura odsotna, je stanje kvantne celice v odsotnosti zunanjih sil nedeterministično. Tako kvantnih celic samih ne moremo uporabiti za hrambo podatkov. Uvedba ure omogoča, da kvantne celice spravimo v zaklenjeno stanje, v katerem prehajanje elektronov med pikami ni mogoče, kar pomeni, da je stanje zaklenjene kvantne celice deterministično. Za pomnenje potrebujemo še način za nadzor stanja, ki naj ga zaklenjena celica zavzame. To lahko dosežemo z veriženjem celic, ki ne pripadajo isti urini coni. Takšna veriga je prikazana na sliki 8.

Za samostojno pomnenje to verigo sklenemo v obroč, prikazan na sliki 9. V obroču se stanje med potekom urinega cikla ohranja. Za smiselno uporabo želimo to strukturo prirediti tako, da lahko hranjeno stanje poljubno nastavljamo. Kako to storimo, je stvar implementacije. V nadaljevanju bomo



Slika 8: Veriga kvantnih celic s sekvenčnimi urinimi conami. Različne barve celic, ki niso vhodne ali izhodne, označujejo različne urine cone.

predstavili nekaj obstoječih implementacij pomnilnih struktur.



Slika 9: Obroč kvantnih celic s sekvenčnimi urinimi conami. Tak obroč sam po sebi ni uporaben, saj ne vsebuje vhodnih in izhodnih celic.

3 Pomnilne celice

Za gradnjo sekvenčnih vezij poleg logičnih vrat potrebujemo tudi pomnilne celice. Osnovne pomnilne celice, ki omogočajo hranjenje enega bita informacije, so RS (Reset Set), JK (Jump Kill), T (Trigger) in D (Delay) pomnilna celica. Delovanje pomnilnih celic je podano v tabelah na sliki 10.

r	s	q	D^1q
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	×
1	1	1	×

RS celica

j	k	q	D^1q
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

JK celica

t	q	D^1q
0	0	0
0	1	1
1	0	1
1	1	0

T celica

d	q	D^1q
0	0	0
0	1	0
1	0	1
1	1	1

D celica

Slika 10: Pravilnostne tabele pomnilniških celic. × predstavlja nedovoljeno stanje.

3.1 Obstoječe implementacije

V članku [4] iz leta 2003 so predstavljene implementacije štirih osnovnih pomnilnih celic. Vse celice porabijo en urin cikel za procesiranje vhodov. Pri postavitvi celic vir uporablja pravilo čimbolj preprostih oblik urinih con in čimmanj le-teh. Po drugi strani pa so nekatere celice zasukane za kot 45° ali zamaknjene za polovico svoje velikosti, kar lahko negativno vpliva na izvedljivost implementacije. Vir uporablja križanje linij, kar je možno doseči z uporabo alternirajočega vodila (rotiranih celic).

Vse implementacije pomnilnih celic v viru [1] iz leta 2018 temeljijo na izpeljavah izraza, ki opisuje RS celico. Te implementacije porabijo manj celic ter ponudijo tako izhodno stanje Q kot njegovo negacijo $\bar{Q}$, so pa zato urine cone bolj kompleksne in uporabljeni so manj stabilni negatorji kot v prvem viru. Vse celice se izvedejo v 1.5 urinega cikla, le D pomnilna celica ima latenco 1.25 urinega cikla.

4 Implementacija

V tem poglavju bomo primerjali predlagane implementacije pomnilnih celic. V viru [1] so enačbe posameznih pomnilnih celic izpeljane iz enačb RS celice:

$$Qb = \overline{\overline{B \cdot CLK} \cdot Q} = B \cdot CLK + \overline{Q}, \tag{1}$$

$$Q = \overline{\overline{A \cdot CLK} \cdot Qb} = A \cdot CLK + \overline{Qb} = A \cdot CLK + \overline{B \cdot CLK} \cdot Q, \tag{2}$$

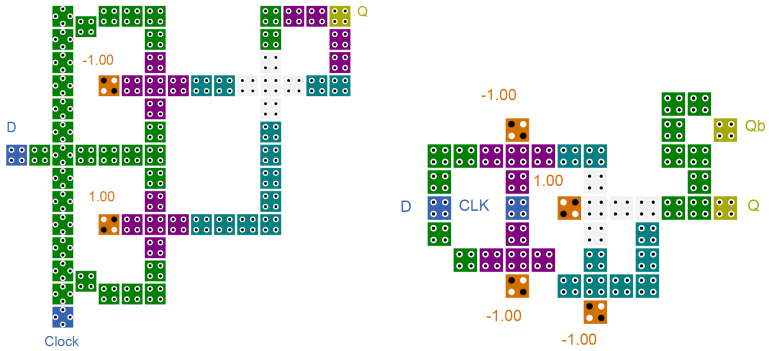
pri čemer je Q izhod celice, Qb njegova negacija, A in B sta vhoda ter CLK je vhodni urin signal. V tabeli na sliki 11 je primerjava števila osnovnih gradnikov v implementacijah pomnilnih celic v [1] in [4].

Pomnilna celica	Vir	Število majoritetnih vrat	Število negatorjev	Število celic
D	[4]	3	2	68
	[1]	4	2	43
JK	[4]	5	2	90
	[1]	6	2	78
RS	[4]	4	1	76
	[1]	4	2	38
T	[4]	5	4	92
	[1]	6	2	81

Slika 11: Primerjava števila gradnikov za sestavo pomnilnih celic.

4.1 D pomnilna celica

Na sliki 12 vidimo implementaciji D pomnilne celice po obeh virih. Vir [4] uporabi križanje linij urinega signala in vhoda D z uporabo alternirajočega vodila zasukanih celic. Vir [1] v enačbah 1 in 2 zamenja vhoda A in B z D in $\overline{D}$ ter tako dobi enačbo D pomnilne celice.

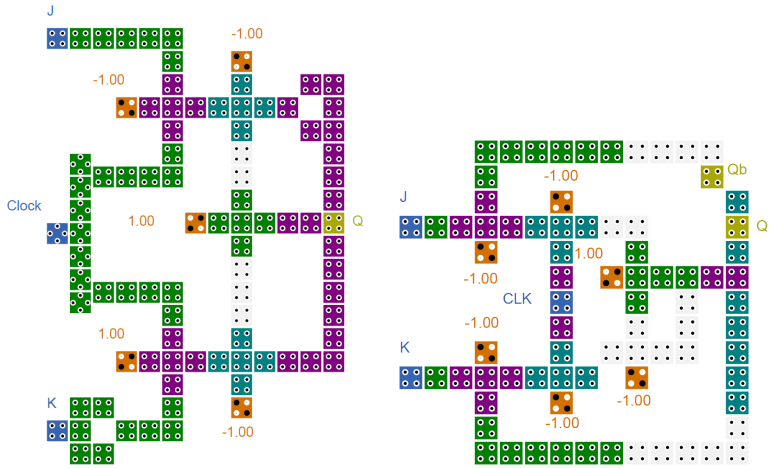


(a) D pomnilna celica po viru [4] (b) D pomnilna celica po viru [1]

Slika 12: Implementaciji D pomnilne celice

4.2 JK pomnilna celica

Na sliki 13 vidimo implementaciji JK celice po obeh virih. Vir [4] uporabi alternirajoče vodilo za širjenje urinega signala za bolj stabilen prenos signala naprej, vidimo tudi dva okrepljena negatorja. Da izpelje enačbo JK pomnilne celice, vir [1] strukturi za RS celico doda dve majoritetni vrati.

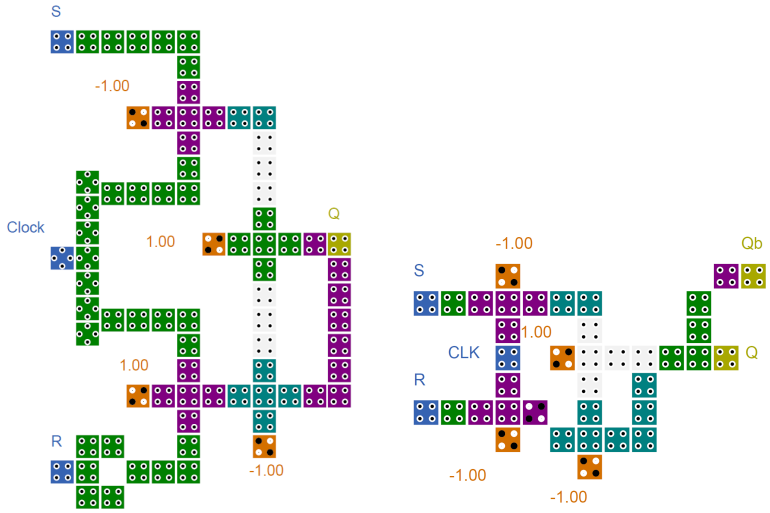


(a) JK pomnilna celica po viru [4] (b) JK pomnilna celica po viru [1]

Slika 13: Implementaciji JK pomnilne celice

4.3 RS pomnilna celica

Na sliki 14 vidimo implementaciji RS celice po obeh virih. Vir [4] kot pri JK celici uporabi alternirajoče vodilo za širjenje urinega signala in en okrepljen negator. Vir [1] v enačbah 1 in 2 zamenja vhoda A in B s S in R ter tako dobi enačbo RS pomnilne celice.

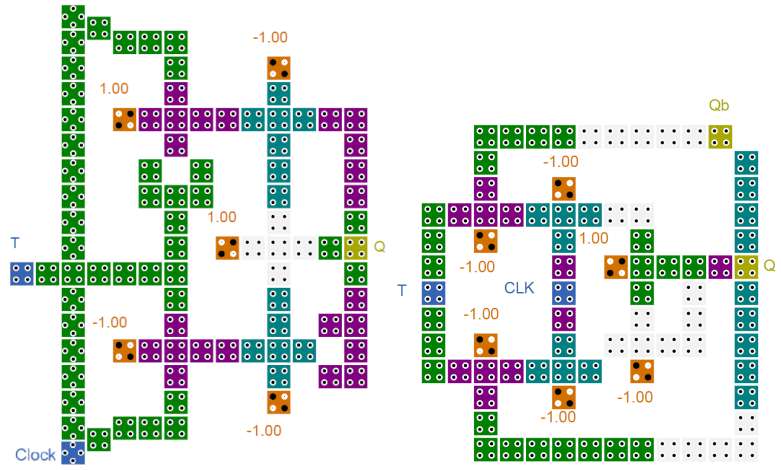


(a) RS pomnilna celica po viru [4] (b) RS pomnilna celica po viru [1]

Slika 14: Implementaciji RS pomnilne celice

4.4 T pomnilna celica

Na sliki 15 vidimo implementaciji T celice po obeh virih. Vir [4] spet uporabi alternirajoče vodilo za širjenje urinega signala in dva okrepljena negatorja. Tudi prenosa z alternirajočega vodila prek polovično zamaknjenih celic učinkujeta kot negatorja. Vir [1] pridobi enačbo za T pomnilno celico iz JK pomnilne celice s povezavo vhodov J in K.



(a) T pomnilna celica po viru [4] (b) T pomnilna celica po viru [1]

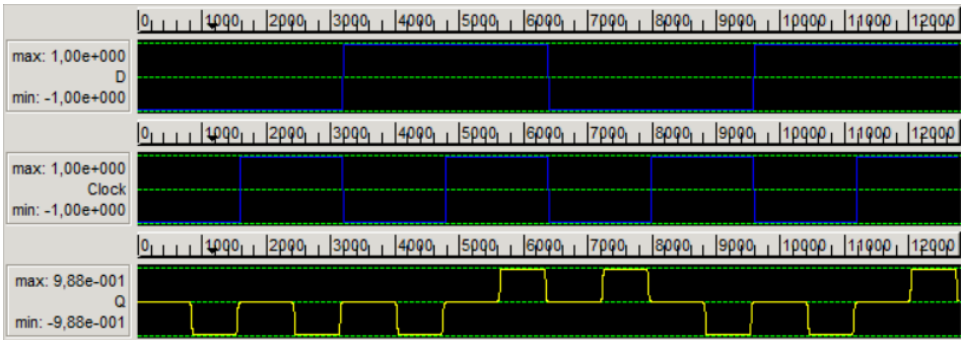
Slika 15: Implementaciji T pomnilne celice

4.5 Analiza delovanja

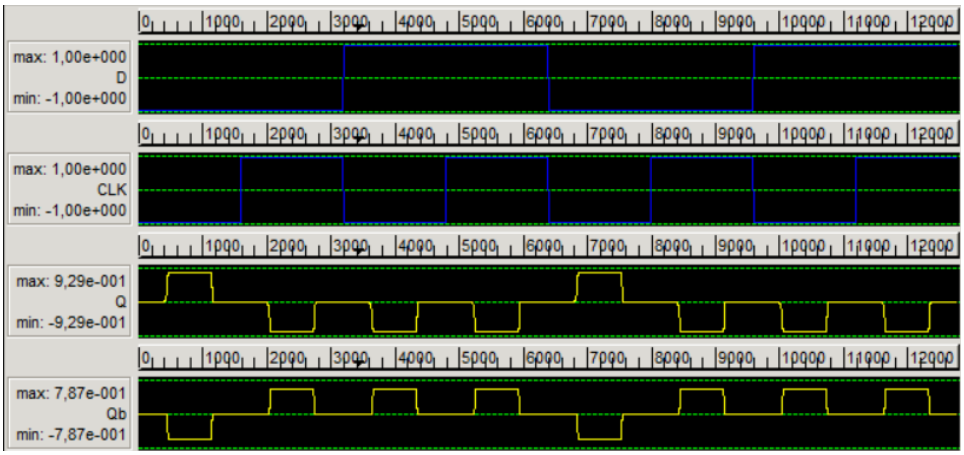
TODO: implementacijam spremeniti opis pravilnosti simulacije
V sledečem poglavju bomo analizirali rezultate simulacij pomnilnih celic in jih primerjali s pričakovanimi pravilnostnimi tabelami.

4.5.1 D pomnilna celica

Na sliki 16 vidimo obnašanje Q celice v odvisnosti od vhoda D in urinega signala *Clock*. Implementacija po viru [4] potrebuje za obdelavo signala 1 urin cikel, implementacija po viru [1] pa 5 urinih faz, kar je 1.25 urinega cikla.
Ob primerjavi rezultatov simulacij D celic opazimo, da se implementacija vira [4] obnaša pravilno, druga implementacija pa v primeru $d = 1, q = 0$ vrne $D^1q = 0$, kar ni skladno s pričakovanji.



(a) Simulacija delovanja D pomnilne celice po viru [4]

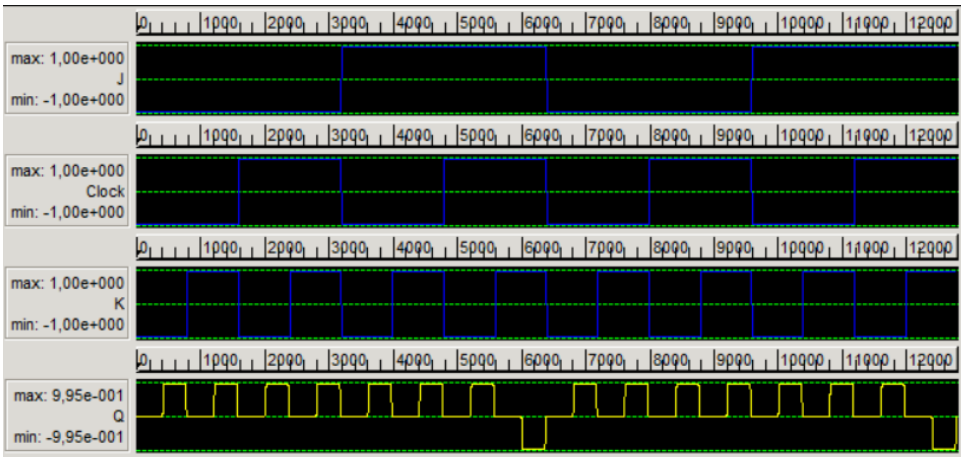


(b) Simulacija delovanja D pomnilne celice po viru [1]

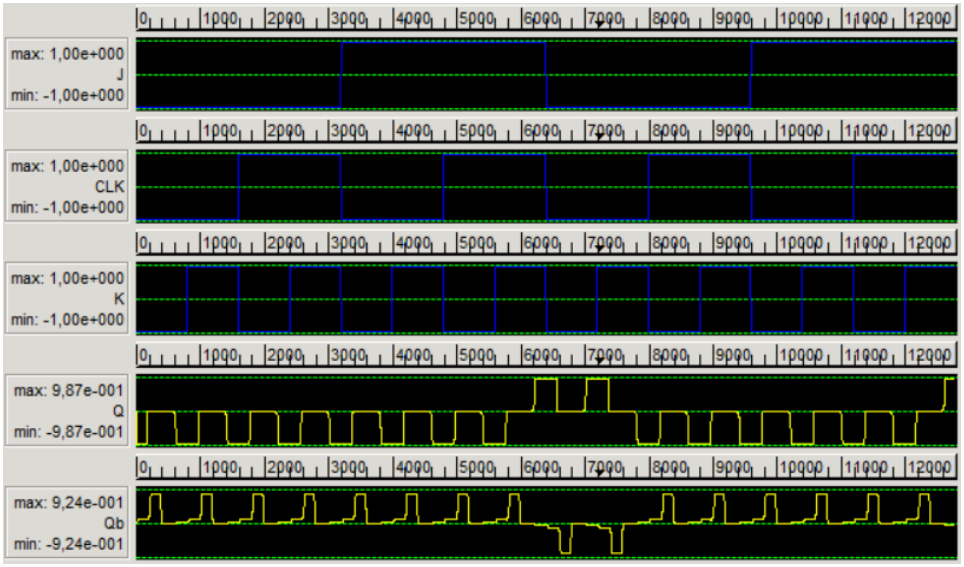
Slika 16: Simulaciji D pomnilne celice

4.5.2 JK pomnilna celica

Na sliki 17 vidimo obnašanje Q celice v odvisnosti od vhodov J in K ter urinega signala *Clock*. Implementacija po viru [4] potrebuje za obdelavo signala 1 urin cikel, implementacija po viru [1] pa 6 urinih faz, kar je 1.5 urinega cikla.
Če primerjamo rezultate simulacij D celic opazimo, da implementacija vira [4] ob vhodu $j = 1, k = 0, d = 1$ včasih vrne 1 namesto 0, vhod $j = 1, k = 1, d = 1$ pa vrne 1. Drugi vir pri $j = 0, k = 1, D = 1$ in pri $j = 1, k = 1, d = 0$ vrne 0.



(a) Simulacija delovanja JK pomnilne celice po viru [4]

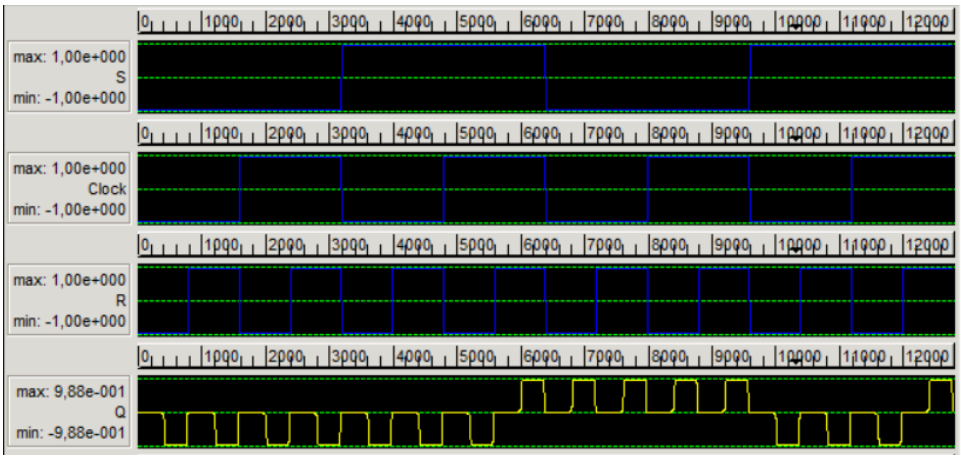


(b) Simulacija delovanja JK pomnilne celice po viru [1]

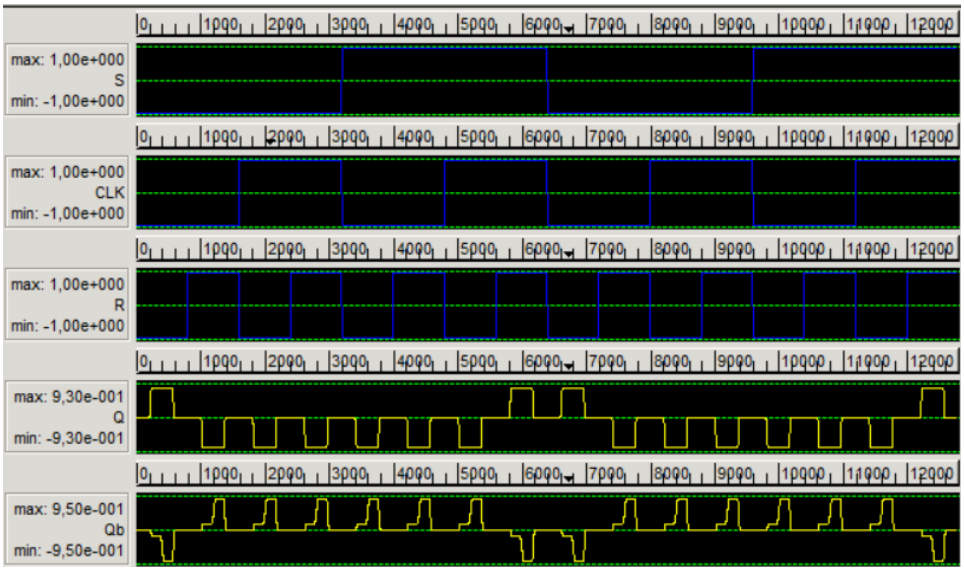
Slika 17: Simulaciji JK pomnilne celice

4.5.3 RS pomnilna celica

Na sliki 18 vidimo obnašanje Q celice v odvisnosti od vhodov R in S ter urinega signala *Clock*. Implementacija po viru [4] potrebuje za obdelavo signala 1 urin cikel, implementacija po viru [1] pa 6 urinih faz, kar je 1.5 urinega cikla.
Simulacija RS celice po viru [4] $r = 1, s = 0, q = 0$ in $r = 0, s = 0, q = 1$ vrne napačen rezultat, implementacija po [1] pa ima napačen rezultat pri $r = 1, s = 0, q = 0$ in $r = 0, s = 0, q = 0$.



(a) Simulacija delovanja RS pomnilne celice po viru [4]

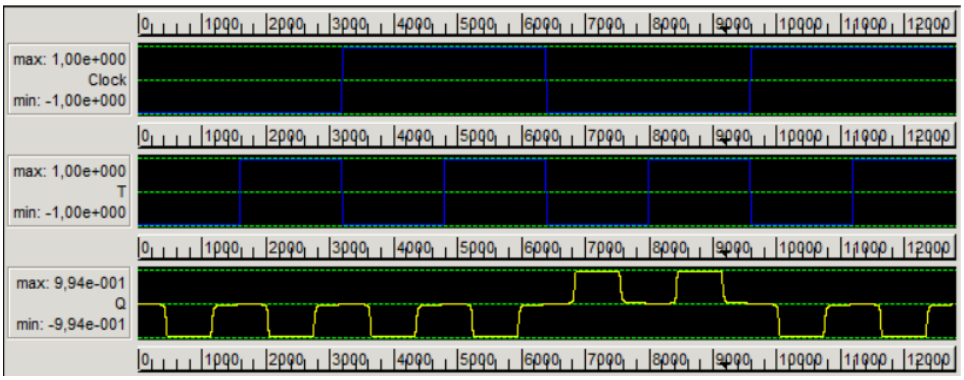


(b) Simulacija delovanja RS pomnilne celice po viru [1]

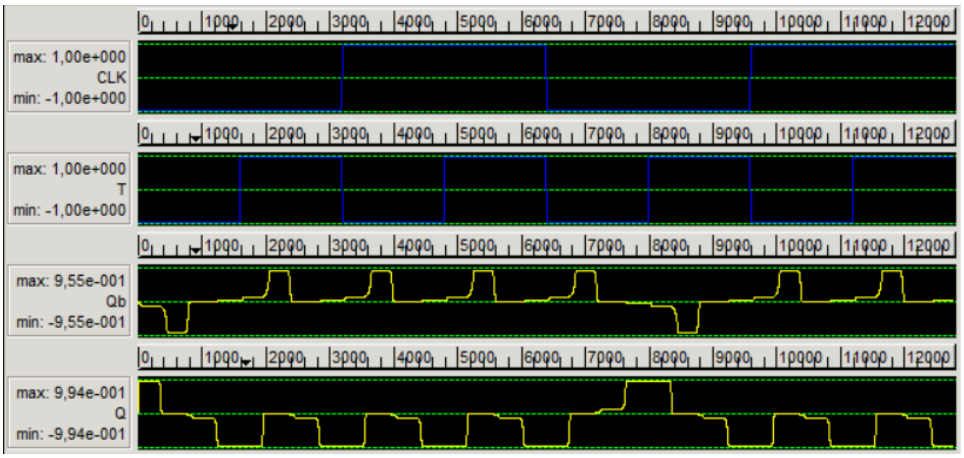
Slika 18: Simulaciji RS pomnilne celice

4.5.4 T pomnilna celica

Na sliki 19 vidimo obnašanje Q celice v odvisnosti od vhoda T in urinega signala *Clock*. Implementacija po viru [4] potrebuje za obdelavo signala 1 urin cikel, implementacija po viru [1] pa 6 urinih faz, kar je 1.5 urinega cikla. Implementacija T pomnilne celice po viru [4] deluje pravilno. Druga implementacija pri $t = 1, q = 0$ vrne 0 namesto 1.



(a) Simulacija delovanja T pomnilne celice po viru [4]



(b) Simulacija delovanja T pomnilne celice po viru [1]

Slika 19: Simulaciji T pomnilne celice

5 Zaključek

TODO: kratek povzetek glavnih ugotovitev + smernice za nadaljnje delo

References

[1] Chakrabarty, R., D. K. Mahato, A. Banerjee, S. Choudhuri, M. Dey,

and N. K. Mandal: *A novel design of flip-flop circuits using quantum dot cellular automata (qca).* 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC), pages 408–414, jan 2018.

[2] Janež, Miha: *Metode razmeščanja in povezovanja logičnih primitivov kvantnih celičnih avtomatov: doktorska disertacija.* PhD thesis, Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, 2012.

[3] Lent, C S, P D Tougaw, W Porod, and G H Bernstein: *Quantum cellular automata.* Nanotechnology, 4(1):49–57, jan 1993.

[4] Vetteth, Anoop, Konrad Walus, Vassil S. Dimitrov, and Graham A. Jullien: *Quantum-dot cellular automata of flip-flops.* ATIPS Laboratory, 2500:1–5, 2002.