UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNITVO IN INFORMATIKO

Sekvenčni primerjalnik dveh n-bitnih števil

Realizacija s kvantnimi celularnimi avtomati

Seminarska naloga
pri predmetu
Optične in nanotehnologije

Blaž Lampreht, Boštjan Žankar,
Luka Stepančič, Igor Vizec

Ljubljana, November 2007

1 Uvod

Za seminarsko nalogo je bilo potrebno realizirati sekvenčni primerjalnik dveh dvo/štiri bitnih števil. Naša realizacija predstavlja primerjalnik dveh n bitnih števil. Za realizacijo je bilo na voljo orodje QCADesigner.

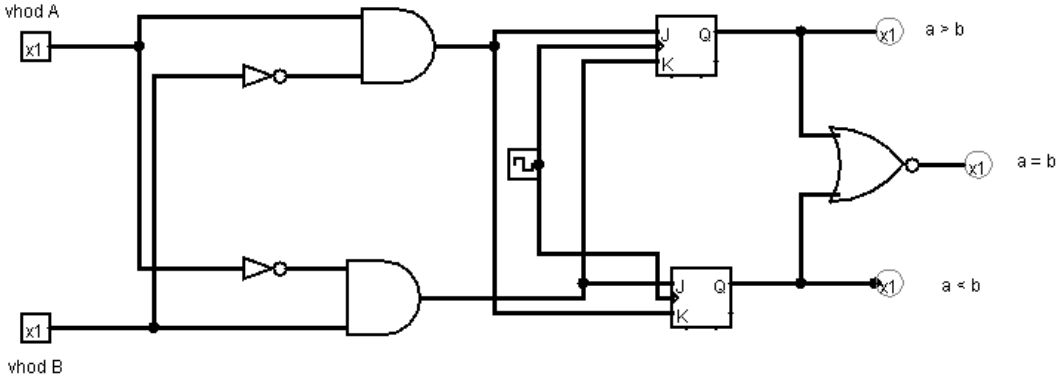
1.1 Opis

Sekvenčni primerjalnik je vezje ki primerja dva vhodna niza bit po bit. Primerjava poteka nad istoležečima bitoma obeh števil, in sicer od najlažjega do najtežjega bita. Na izhodu pa dobimo tri signale: $a < b$, $a > b$, $a = b$. Izhodni signali so medeseboj izključujoči, saj je naenkrat lahko aktiven le en izhod.

2 Realizacija z JK-celicami

2.1 Opis

Na vhoda A in B pripeljemo števili, kateri želimo primerjati. Števili se primerjata bit po bit in kadar je eden od vhodov večji (torej 1, ko je drugi vhod 0), se na 1 postavi izhod pripadajoče JK celice. Če pogledamo na prejšnjo shemo, ugotovimo da, ko je vhod A postavljen na 1 in vhod B na 0, bosta vhoda zgornje JK celice postavljena J na 1 in K na 0, kar da izhod 1. To pomeni, da je število A večje od števila B. Kadar sta bita enaka, bodo na vseh vhodih v JK celice ničle, kar pomeni, da se stanji obeh JK celic ohranjata. Če pa sta števili popolnoma enaki, se stanji celic ohranjata že od začetka primerjanja, kar pomeni, da je potrebno obe JK celici na začetku resetirati.



Slika 1: Logično vezje implementacije

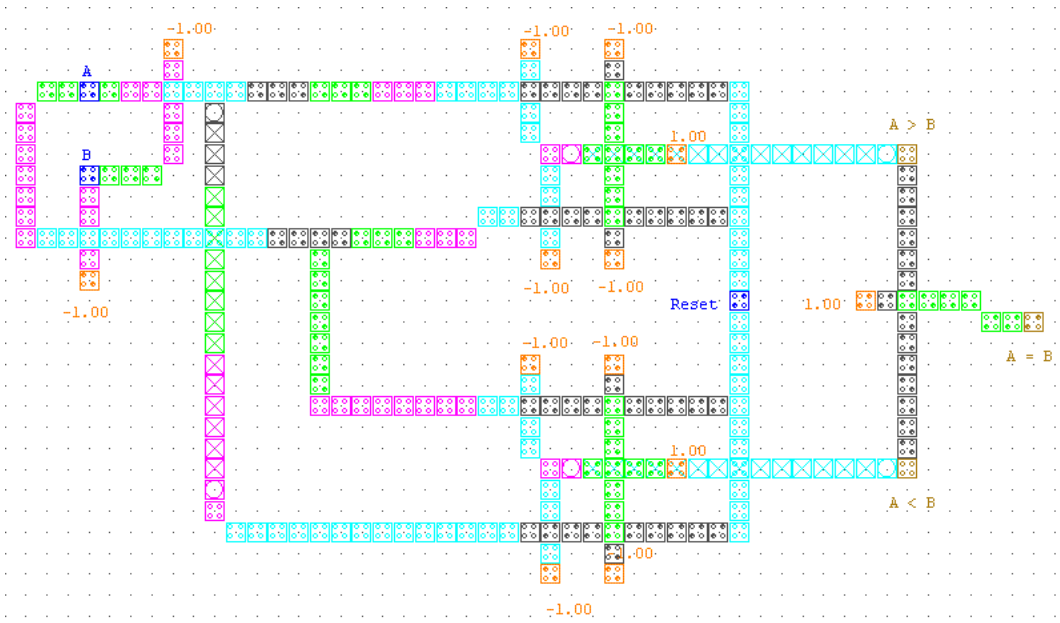
$$D^1q = j\bar{q} \vee \bar{k}q$$
(a)

J	K	D^1q
0	0	q
0	1	0
1	0	1
1	1	$\bar{q}$

(b)

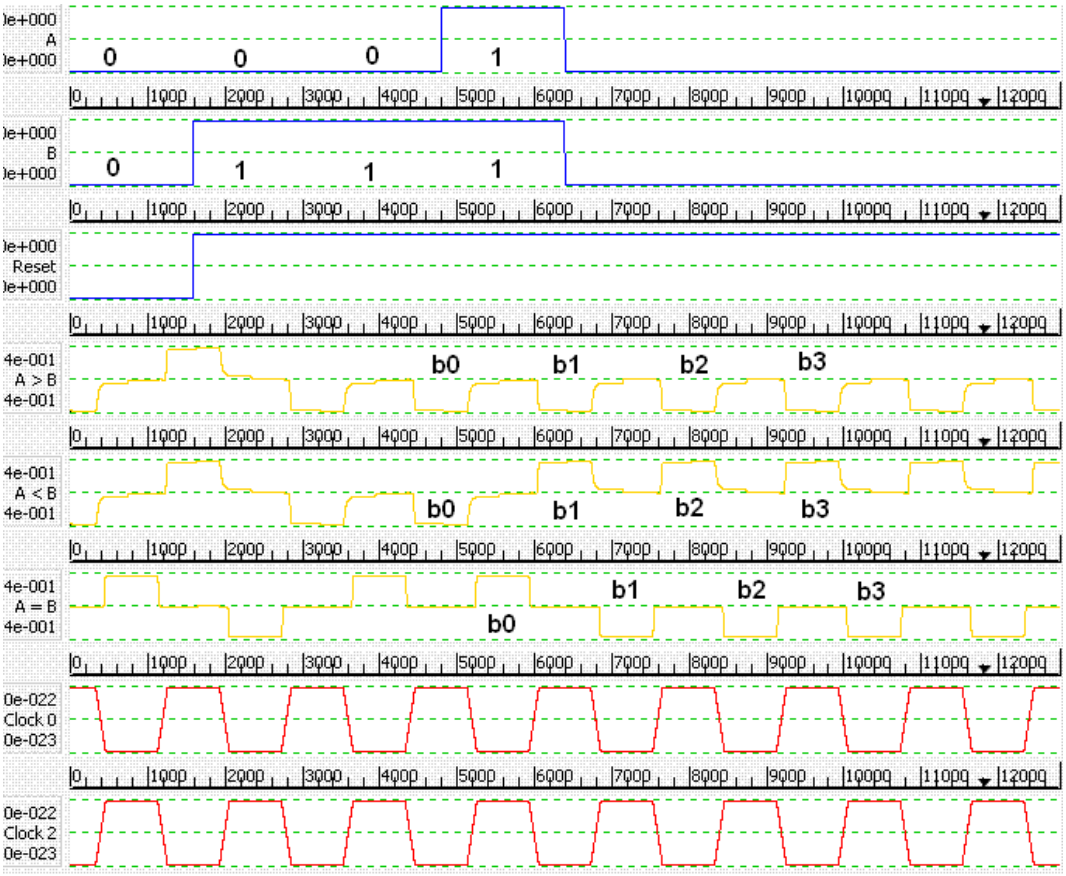
Slika 2: Enačba in pravilnostna tabela JK celice

2.2 Implementacija v QCADesigner-ju



Slika 3: Vezje JK implementacije v QCA

Za izhoda $a > b$ in $a < b$ gledamo Clock 2 od 3. urine periode naprej, za izhod $a = b$, pa Clock 0 od 4. periode dalje.



Slika 4: Primer JK: $A > B$

3 Realizacija z RS-celicami

Vezje skevenčnega primerjalnika, lahko realiziramo tudi na osnovi RS pomnilne celice. Enačbe vezja so zelo podobne realizaciji z JK pomnilno celico, saj je JK celica nadgradnja RS celice. Torej je edina omejitev pogoj $rs = 0$. Torej potrebujemo enačbe za izhode $a < b$, $a = b$ ter $a > b$.

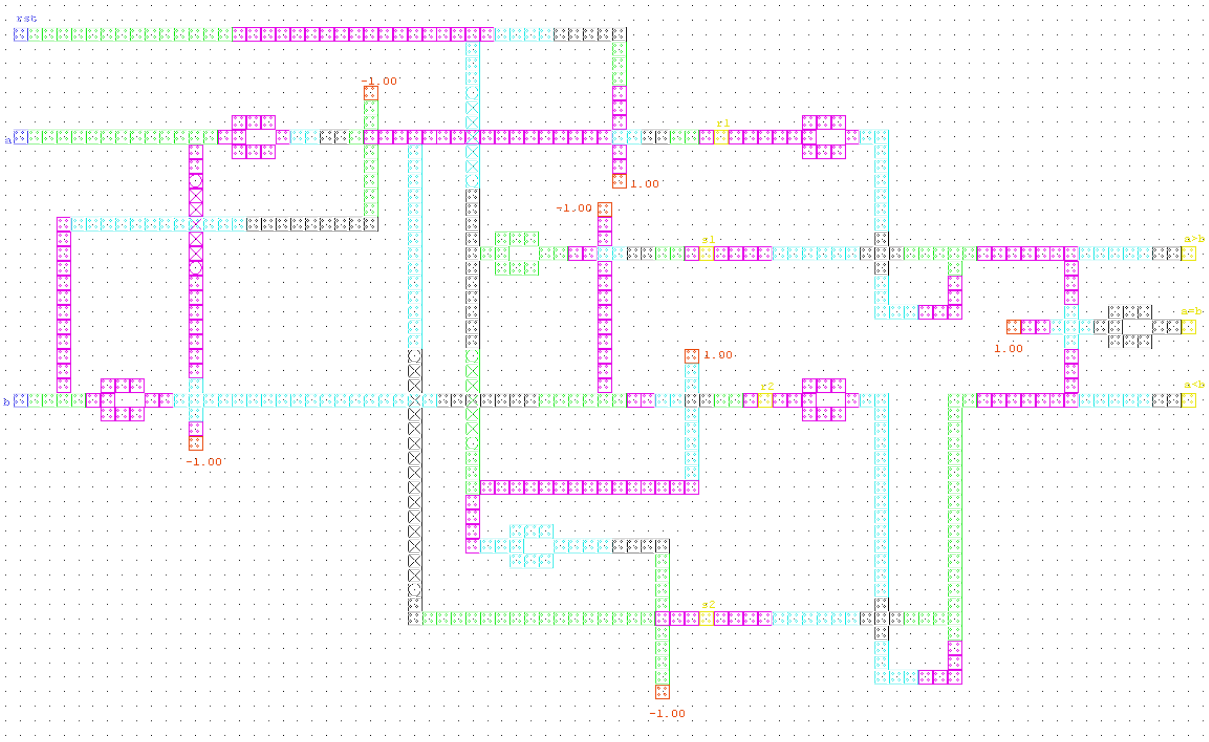
$$a < b: \ D^1p = p\left(\overline{(\bar{a}b \vee \ rst)}\right) \vee \left(\left(a\bar{b}\ \overline{rst}\right)\right)$$

$$a > b: \ D^1q = q\left(\overline{\left(a\bar{b} \vee rst\right)}\right) \vee \left(\left(\bar{a}b\overline{rst}\right)\right)$$

$$a = b: \ e = \overline{q \vee p} = \bar{q}\bar{p}$$

Iz zgornjih enačb vidimo da v vhoda r,s celic vstopata vrednosti $\bar{a}b$ in $a\bar{b}$. Ker pa se moramo držati pogoja $rs = 0$, sledi $\bar{a}ba\bar{b} = 0$ (postulat o inverznih

elementih).

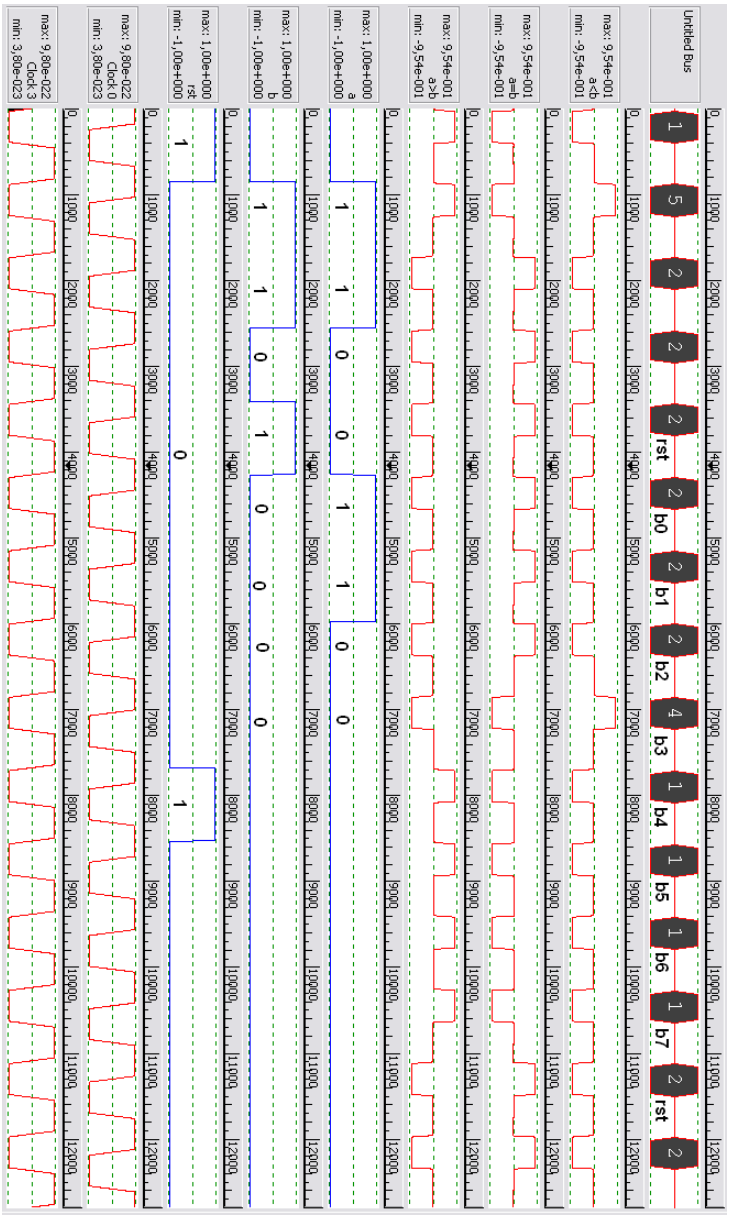


Slika 5: Vezje RS implementacije

Slika 5 prikazuje realizacijo sekvenčnega primerjalnika z RS celicama v QCA. Samo vezje lahko razbijemo na dva dela. Prvi del je potreben za izračune vhodov v celici, na podlagi vhodov v vezje a, b in *reset* signala. Drugi del, pa je sekevnčni, ki izpisuje na izhod stanje primerjave do časa n, ter računa stanje $n + 1$. Na vhoda a in b vodimo števili, ki ju primerjamo. Vzamemo bit po bit, od najlažjega (LSB), do najtežjega (MSB) bita. Da pa seveda primerjalnik primerja prave bite, ga je potrebno predhodno resetirati oz postaviti na vhod reset logično enico. Reset postavi oba R vhoda celic na "1", S vhoda pa na nič. S tem zagotovimo logično stanje nič za q pomnilne celice. Po resetu pa takoj na vhode dajemo bite primerjanja. Vsak nov vhodni vzorec a, b in *reset* signalov se pojavi na izhodu zakasnjen za 4 cikle QCA ure.
Graf prikazuje primerjanje dveh števil, in sicer

$$a = 51_{(10)} = 00110011_{(2)}$$

$$b = 11_{(10)} = 00001011_{(2)}$$



Slika 6: Primer za RS, $a = 51,\ b = 11$

Za lažje branje so izhodi oz. rezultati primerjanja zapiasni v vodilu. Če se na vodilu pojavi vrednost 2 to pomeni ali reset, ali pa da smo dosedaj vedno primerjali bita z isto vrenostjo. Vrednost 1 na vodilu pomeni, da je število a večje od b, ravno obratno pa nam pove vrednost 4 na vodilu.

4 Realizacije v minimalni obliki

4.1 Implementacija z majoritetnimi vrati s 3 vhodi

Če enačbe za serijski primerjalnik ustrezno priredimo za nabor logičnih funkcij $\{maj,\ not\}$ in sicer v minimalni obliki, se logično vezje razmeroma poenostavi. Izhajamo iz enačb RS-celice:

$$a > b:\ D^1p = \bar{r}_p p \vee s_p = a\bar{b} \vee ap \vee \bar{b}p$$
$$a < b:\ D^1q = \bar{r}_q q \vee s_q = \bar{a}b \vee \bar{a}q \vee bq$$

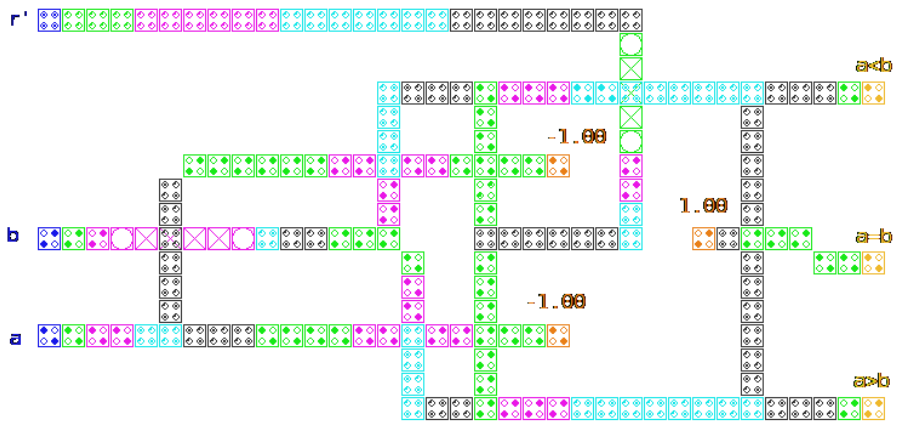
Te enačbe lahko zapišemo v obliki:

$$D^1p = maj(a, \bar{b}, p)$$
$$D^1q = maj(\bar{a}, b, q)$$

Ker je vsebina celic na začetku nedefinirana, potrebujemo še dodaten reset signal r, ki celice ponastavi na logično "0" preden gresta p in q ponovno v majoritetna vrata. Ta bo služil tudi za nakazovanje konca števil (oz. niza bitov števil) in omogočil novo primerjavo.

$$D^1p = \bar{r}\ maj(a, \bar{b}, p)$$
$$D^1q = \bar{r}\ maj(\bar{a}, b, q)$$
$$e = \bar{p}\bar{q}$$

Na podoben način lahko lahko rezulat dobimo z minimizacijo z Veitch-evim diagramom. In sicer iščemo vzorec, ki predstavlja majoritetna vrata.
Da iz dobljenih enačb sestavimo vezje, potrebujemo predhodno resetirati vrednosti p in q. To lahko dosežemo na več načinov, med katerimi je najlažje tako, da signale p in q množimo z resetom (log. funkcija 'AND') preden jih pripeljemo nazaj v majortitetna vrata. Taka realizacija ima zakasnitev 4 u.p., glavni vzrok temu so vhodi zunaj vezja, sicer bi se lahko doseglo 1.u.p.



Slika 7: Minimizirana oblika - 3-vhodna maj. vrata

4.2 Implementacija z majoritetnimi vrati s 5 vhodi

Sedaj se pojavi vprešanje ali bi lahko še dodatno minimizirali serijski primerjalnik, če bi imeli na voljo majoritetna vrata s 5 ali več vhodi?
Predpostavimo še, da morata biti vhoda a in b ob vsakem resetu enaka "0" - s tem zagotovimo, da so ob resetu (signal $r \equiv 0$) na majoritetnih vratih vsaj tri logične ničle. Ta pogoj lahko krajše zapišemo:

$$r \vee \bar{a}\bar{b}$$

Izkaže se, da se ob upoštevanju zgornjega pogoja enačba lahko zreducira v obliko, kjer imamo na voljo samo ena 5-vhodna majoritetna vrata:

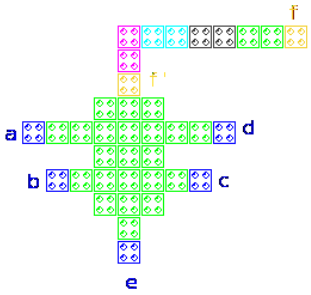
$$\begin{aligned} D^1p &= \bar{r}\ maj(a, \bar{b}, p) \\ &= \bar{r}a\bar{b} \vee \bar{r}ap \vee \bar{r}\bar{b}p \\ &= \bar{r}a\bar{b} \vee \bar{r}ap \vee \bar{r}\bar{b}p \vee a\bar{b}p \\ &= maj(\bar{r}, 0, a, \bar{b}, p) \end{aligned}$$

in podobno:

$$D^1q = maj(\bar{r}, 0, \bar{a}, b, q)$$

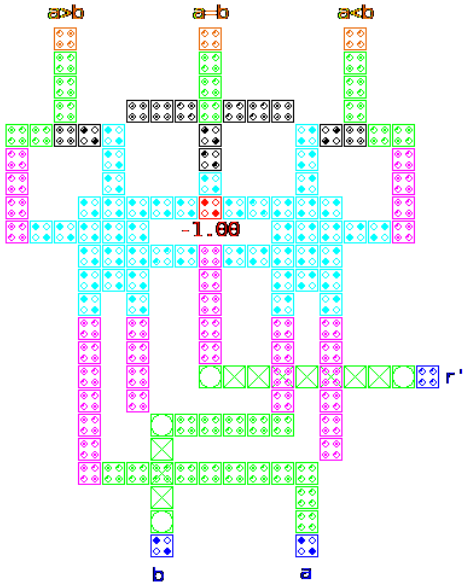
$$e = \bar{p}\bar{q}$$

Na srečo se je izkazalo, da lahko realiziramo maj. vrata z več vhodi, ki so razmeroma enakomerno obteženi, primer kaže slika 8. Vrata delujejo na



Slika 8: Majoritetna vrata s 5 vhodi

podoben princip kot trovhodna majoritetna vrata - številčnejša vrednosti na vhodih “zmaga”.
Z uporabo podobnih vrat nam je uspelo implementirati različico, ki vsega skupaj vsebuje samo tri majoritetna vrata (2x5vhodna 1x3vhodna), končno vezje se tako še dodatno poenostavi.



Slika 9: Minimizirana oblika - 5-vhodna maj. vrata

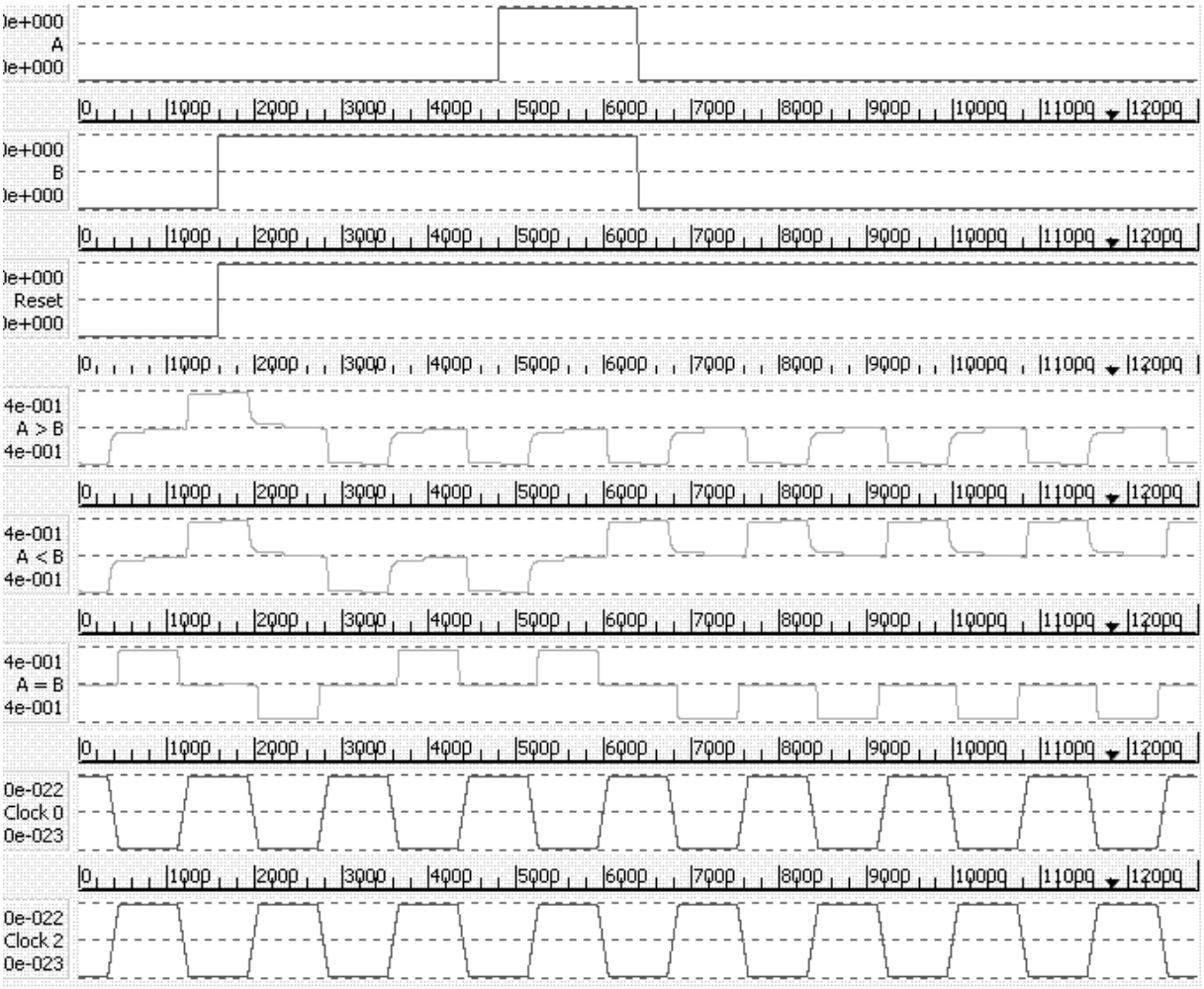
Pri tej različici nastopijo težave z stabilnostjo signalov, ki izhajajo iz maj. vrat - ali drugače rečeno, signali se stabilizirajo kasneje kot pri vratih s tremi vhodi. Te težave vendarle lahko razmeroma enostavno odpravimo z adiabatnim preklopom. Ostane samo potreben pogoj, da sta vhoda a in b ob resetu vedno enaka ”0”. Ampak to je razmeroma enostavno izvedljivo.

Primeri grafov so v prilogah. Pomembna prednost pri tej realizaciji je nizka zakasnitev, ta je namreč samo 1 u.p.

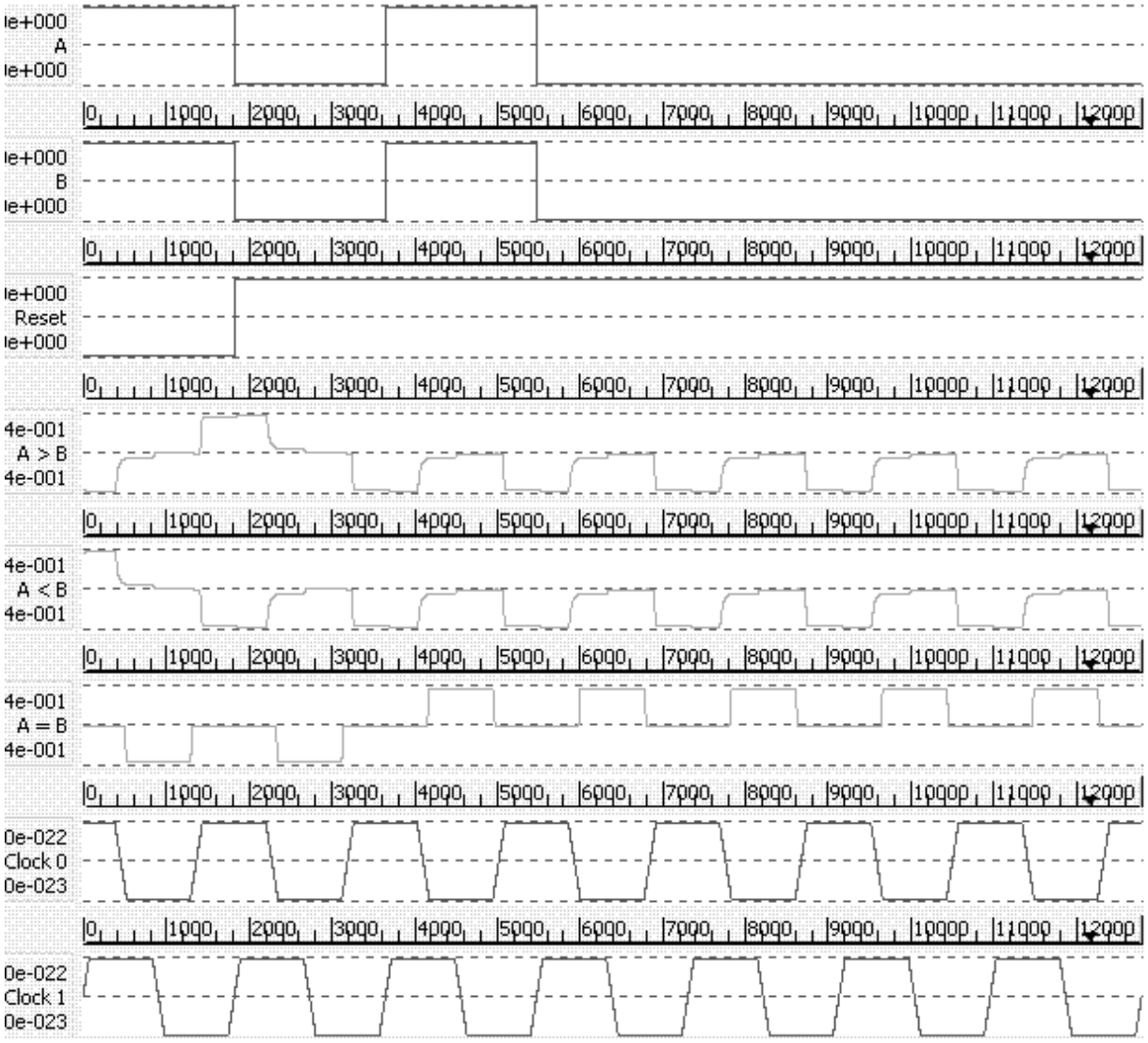
5 Zaključek

Realizacija primerjalnika ni potekala brez težav, saj je bilo potrebno paziti, da uporabljenih celic ni bilo preveč, ker se je drugače simulator začel obnašati nenavadno. Potrebno je bilo tudi paziti, da konstante (-1 in 1) niso bile preblizu ostalim celicam, saj so drugače preveč vplivale nanje. V ta namen smo pri uporabi majoritetnih vrat konstante zamaknili za eno do dve celici. Pri vsakem križanju linij je bilo potrebno uporabiti več plasti ter smiselno uporabiti dele (clock-e) urine periode, tako da križani liniji nista vplivali ena na drugo. Pri negaciji linije je bilo potrebno uporabiti drug del urine periode za negiran del linije, da je negacija vedno delovala pravilno. Prav tako je bilo potrebno uporabiti drug del urine periode pri liniji ki se je prelomila, ter pri izhodih iz majoritetnih vrat. Navkljub težavam, pa je bilo delo zanimivo in poučno, saj nam je malce bolj razjasnilo kvantne avtomate in delo z njimi.

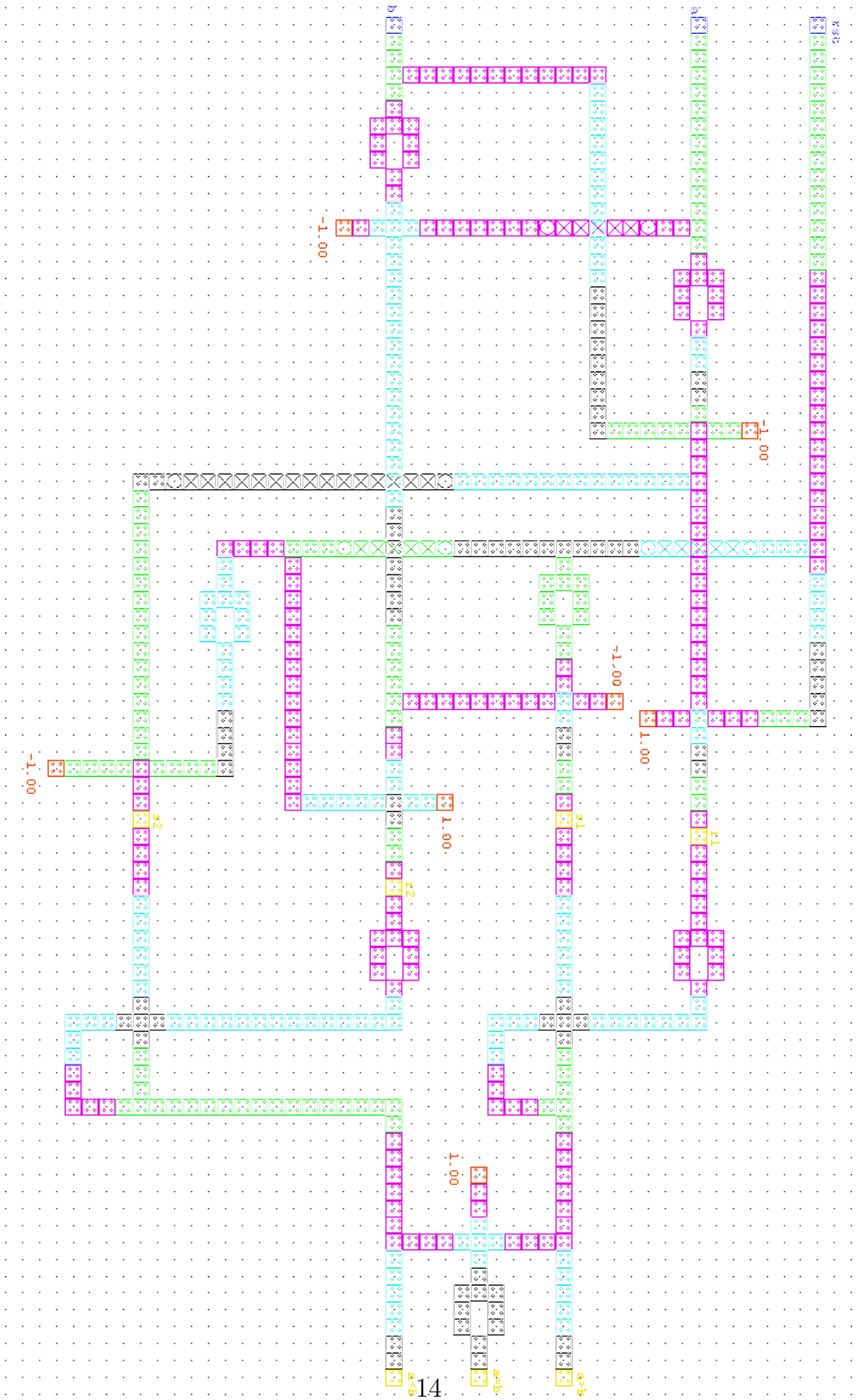
Priloge



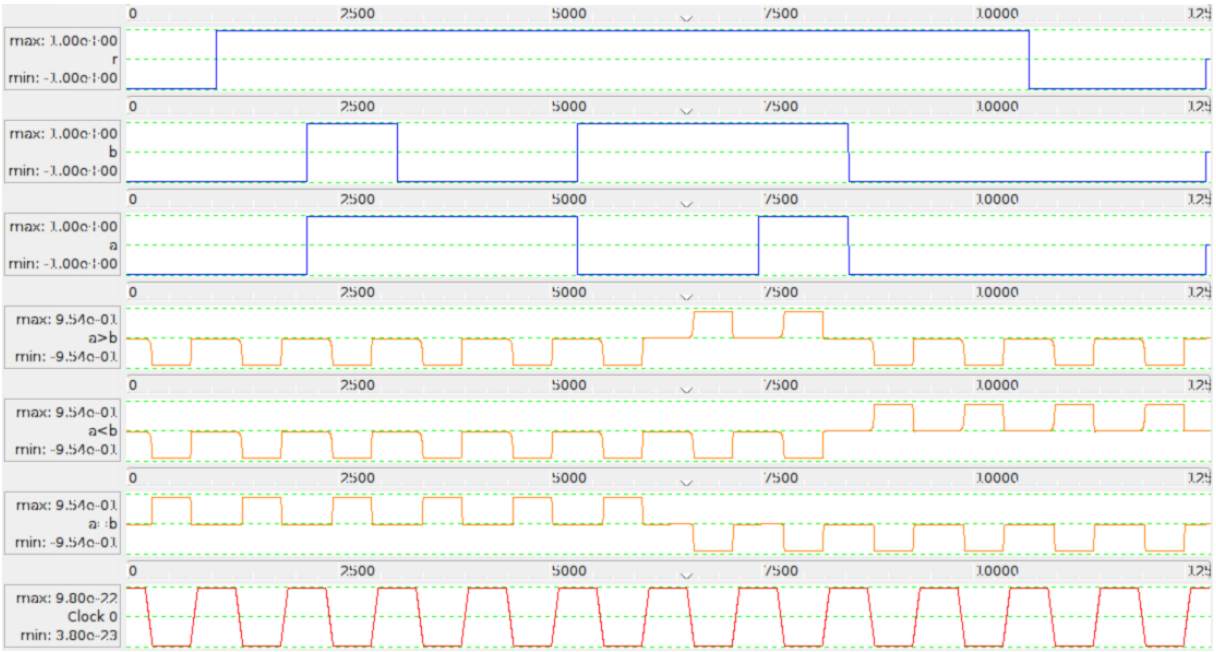
Slika 10: Primer JK: ($A < B$)



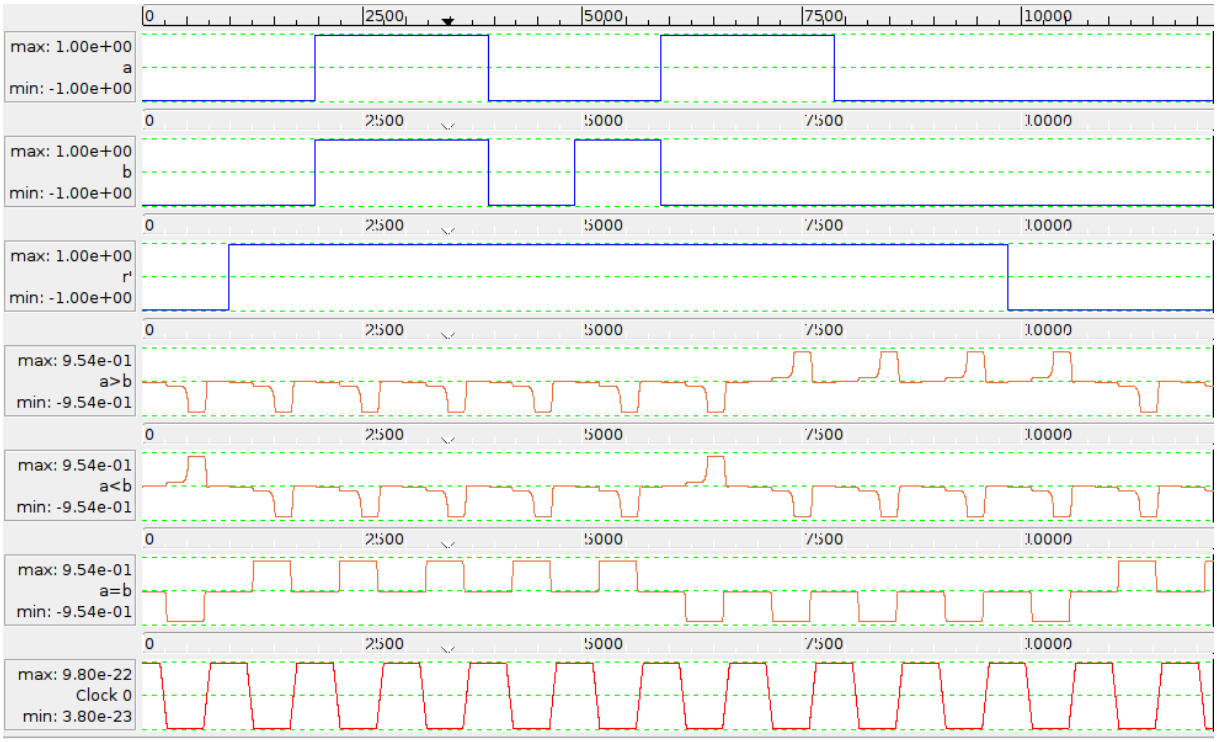
Slika 11: Primer JK: $(A = B)$



Slika 12: RS - vezje



Slika 13: Min. oblike, 3-vh. majoritetna vrata



Slika 14: Min. oblike, 5-vh. majoritetna vrata