ELEKTROTEHNIŠKI VESTNIK 83(4): 194–200, 2016
IZVIRNI ZNANSTVENI ČLANEK

# Realizacija dvojiškega pomnjenja v preprostih bioloških sistemih

**Miha Moškon, Nikolaj Zimic, Miha Mraz**

*Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, Večna pot 113, 1000 Ljubljana, Slovenija*
*E-pošta: miha.moskon@fri.uni-lj.si*

**Povzetek.** V pričujočem članku predstavimo stanje razvoja na področju bioloških pomnilnih struktur, ki so eden od temeljev realizacije biološkega računalnika. Po analogiji s pomnjenjem v elektronskih računalniških sistemih lahko biološke izvedbe pomnjenja razdelimo na dve skupini, in sicer na trajno pomnjenje podatkov neposredno v zapisu DNA in začasno pomnjenje podatkov na podlagi vzdrževanja in preklapljanja logičnih nivojev koncentracij kemijskih zvrsti, katerih ekspresijo definira DNA. V članku predstavimo koncept realizacije izvedbe trajno pomnjenega celičnega programa, katerega ukaze naslavljamo z začasno pomnjenimi vrednostmi biološkega Johnsonovega števca. Za potrebe implementacije števca predlagamo biološko shemo D pomnilne celice v vezavi *Master-Slave*.

**Ključne besede:** pomnjenje, sintezna biologija, modeliranje in simulacije, Johnsonov števec, *Master-Slave* D pomnilna celica

**Implementation of a binary memory in simple biological circuits**

In the paper we give a structured overview of the *state-of-the-art* in the synthetic biological memory structures. A reliable implementation of these structures and their integration with other combinatorial logic components into functional circuits presents an essential step in the implementation of a biological computer. Drawing an analogy from the digital electronic systems used in modern computers, we divide the biological memory structures in two groups. The *Non-volatile structures* that store the information directly into sequences composing the DNA strands and the *Volatile structures* that store the information in a form of concentrations of different chemical species, e.g., proteins, the expression of which is again defined by the corresponding DNA sequences, i.e. by their genes. We propose an implementation of a cellular program stored as a DNA sequence and describe its execution scheme. We introduce an implementation of a biological version of the Johnson's counter. The implementation of the counter is performed with a biological version of the Master-Slave D flip-flop, dynamics of which depends on the regulatory interactions among the selected proteins and their genes. We show how to use the counter to address and execute the cellular program.

**Keywords**: memory, synthetic biology, modelling and simulations, Johsnon counter, Master-Slave D Flip-Flop

## 1 UVOD

V zadnjem desetletju so se raziskave na področju sintezne biologije usmerile na graditev preprostih bioloških sistemov, ki opravljajo posamezne primitivne funkcije, na katerih temeljijo današnji računalniški sistemi. Tako so bili že izdelani *biološki preklopni logični primitivi* (npr. vseh 16 dvovhodnih dvovrednostnih logičnih vrat [1]), že kar natančni in zanesljivi *biološki oscilatorji* [2], v zadnjem času pa nastajajo tudi prve *biološke pomnilne strukture*. Te naj bi opravljale funkciji *kratkotrajnega* (angl. *short term storage*) in *dolgotrajnega pomnjenja* (angl. *long term storage*). Prva funkcija naj bi omogočala začasno pomnjenje vrednosti operacijskih spremenljivk dinamičnega biološkega sistema (npr. pomnjenje režima načina delovanja biološkega zdravila), druga pa trajno pomnjenje izbranih podatkov. Z vidika načina realizacije pomnjenja biološke pomnilne strukture delimo v dve skupini, in sicer na

- strukture, ki omogočajo *trajno pomnjenje* dvovrednostnih podatkov neposredno v DNA zapisu in
- strukture, ki omogočajo *začasno pomnjenje* dvovrednostnih podatkov v obliki vzdrževanja nizkih ali visokih koncentracij opazovanih kemijskih zvrsti (npr. proteinov), ki so rezultat izražanja (ekspresije) posameznih genov DNA-zapisa.

Ob predpostavki, da so biološki preklopni logični primitivi, biološki oscilatorji in biološke pomnilne strukture *modularni*, lahko na podlagi njihovega obstoja začnemo snovati preproste biološke računalnike, saj imamo s tem zagotovljene tako logične, kot tudi pomnilne funkcije, vključno z urinim taktom.

V pričujočem članku predstavimo dosedanji razvoj bioloških pomnilnih struktur in na podlagi tega naredimo korak naprej. Z uporabo obeh vrst pomnjenja predstavimo koncept izvajanja trajno pomnjenega celičnega programa, katerega ukaze naslavljamo (izbiramo) z začasno pomnjeno vrednostjo biološkega Johnsonovega števca.

*Prejet 15. april, 2016*
*Odobren 28. junij, 2016*

## 2 POMNJENJE V RAČUNALNIŠKIH SISTEMIH

Računalniški sistem je kakršenkoli samodejno delujoči sistem, ki je zmožen opravljati funkcije *procesiranja*, *pomnjenja* in *prenašanja* podatkov. Zmožnost procesiranja, decizije ali odločanja nad podatki si predstavljamo kot zmožnost odzivanja ali formiranja odziva na vhodne podatke, pri čemer se posamezni odziv izdela samodejno na podlagi *vgrajene logike* ali *programa*. Zmožnost prenašanja pojmujemo v ožjem pomenu kot mehanizem prenosa *podatkov* in *programov* med pomnilnikom, procesnim segmentom in vhodno/izhodnimi enotami (torej v opazovanem računalniškem sistemu), v širšem pomenu pa kot mehanizem prenosa podatkov in programov med računalniškimi sistemi.

Funkcijo pomnjenja lahko razdelimo na funkciji *trajnega* in časovno omejenega *začasnega* pomnjenja. Zgodovinsko gledano je bila v računalništvu funkcija trajnega pomnjenja od nekdaj namenjena pomnjenju programov in le izbranih podatkov, funkcija začasnega pomnjenja pa pomnjenju vrednosti operacijskih spremenljivk programov.

Večina podatkov, s katerimi danes operirajo računalniški sistemi, je dvovrednostne logične narave. Ne glede na vrsto nosilca podatkov (npr. nivo električne napetosti, namagnetenost medija ali tok fotonov po optičnem omrežju) tako velja, da vrednost nosilca podatkov preslikamo v logične nivoje, ki jih enačimo z naborom logičnih vrednosti $\{0, 1\}$. Tako za vse tri kategorije (procesiranje, pomnjenje in prenašanje) velja, da z logičnega vidika uporabljajo entitete logičnih vrednosti iz množice $\{0, 1\}$ ali *bite*.

## 3 TRAJNO POMNJENJE PODATKOV NEPOSREDNO V ZAPISU DNA

Trajno pomnjenje podatkov *neposredno* v zapisu DNA temelji na *kodiranju* ciljnih dvovrednostnih podatkov (bitov) v zapis DNA, ki temelji na kodnem naboru simbolov $\{\mathbf{A}, \mathbf{C}, \mathbf{G}, \mathbf{T}\}$. Simboli so oznake nukleotidov Adenina, Citozina, Gvanina in Timina, ki so osnovni sestavni deli DNA-zapisa. Ob hipotetični predpostavki, da imamo *algoritem kodiranja* posameznih kodov kodnega nabora $\{0, 1\}$ v kode kodnega nabora $\{\mathbf{A}, \mathbf{C}, \mathbf{G}, \mathbf{T}\}$, je končni korak do želenega zapisa DNA in s tem trajnega pomnjenja le še postopek umetne sinteze zapisa DNA, ki je danes že tržno dosegljiv.

V strokovni literaturi najdemo kar nekaj primerov izvedbe pomnjenja podatkov neposredno v zapisu DNA. Večina avtorjev izvedb poudarja naslednje prednosti tovrstnega pomnjenja:

- izjemno velika gostota pomnjenja (največja mogoča gostota pomnjenja v zapisu DNA je $455$ EB (angl. *exabytes*)/gram ($1$ EB=$10^{18}$ B=$1000$ PB=$10^6$ TB) [3]),
- izjemno dolga življenjska doba oziroma stabilnost zapisa v ambientalnem okolju (angl. *long lifespan in low-maintenance environment*) [4],
- izjemno dolga predvidena življenjska doba formata zapisa, saj zapis DNA postaja *de-facto* standard biološkega zapisa vseh živih organizmov [5],
- izjemno velika energetska učinkovitost zapisa (angl. *energy efficiency*).

| $b_{2*i}$ | $b_{2*i+1}$ | $k_i$ |
|---|---|---|
| 0 | 0 | $A$ |
| 0 | 1 | $T$ |
| 1 | 0 | $G$ |
| 1 | 1 | $C$ |

Tabela 1: Preslikovalna tabela parov bitov v nukleotidni zapis ($i = 0, ..., n-1$)

Avtorji realizacij po drugi plati priznavajo problematičnost zrelosti tehnologije pisanja in branja zapisa DNA. Oba postopka sta kljub rastoči zanesljivosti namreč še vedno dolgotrajna, cenovno nekonkurenčna in s tem primerna samo za hrambo podatkov za daljša časovna obdobja, kot smo jih vajeni danes (npr. za več desetletij ali celo stoletij (angl. *century-scale archives*)). Glede na povedano je tovrstni način hrambe primeren predvsem za dolgotrajno pomnjenje velikih količin (angl. *large-scale*) arhivskih podatkov (angl. *archival storage*), pri čemer je ekonomsko upravičeno le zelo redko (angl. *infrequently accessed*) in sočasno parcialno dostopanje do vsebinsko zaključenih enot podatkov, ki so na pomnilnem mediju DNA shranjene v fizičnem sosedstvu, kar pri postopkih branja in pisanja daje prednost aplikacijam z zaporednim dostopom do podatkov (angl. *sequential access applications*) [3].

### *3.1 Algoritem kodiranja pri neposrednem pomnjenju v zapisu DNA*

Ob predpostavkah, da je zapis DNA lahko poljubno dolg in da je lahko sestavljen iz poljubnega zaporedja nukleotidov (kodov iz kodnega nabora $K = \{\mathbf{A}, \mathbf{C}, \mathbf{G}, \mathbf{T}\}$), smo soočeni s problemom določitve načina kodiranja logičnih vrednosti $0$ in $1$.

Predpostavimo, da je želena hranjena podatkovna vsebina v nam znanem digitalnem svetu zakodirana v zaporedje sode dolžine $2 * n$ bitov $(b_0, b_1, ..., b_{2n-1})$, želimo pa jo prestaviti v zaporedje kodnega sveta nukleotidov dolžine $l$ $(k_0, k_1, ..., k_{l-1})$. Glede na to, da je ciljni kodni nabor $K$ po zalogi vrednosti dvakrat večji od kodnega nabora $\{0, 1\}$, se nam sama od sebe ponuja logika preslikave, predstavljena v preslikovalni tabeli 1, kjer po dva sosednja bita $(b_{2*i}, b_{2*i+1})$ iz bitnega zaporedja preslikujemo v en nukleotid $k_i$ $(i = 0, ..., n-1)$ po pravilu iz tabele. Tako dobimo kompaktnejši zapis z vidika manjšega števila uporabljenih pomnilniških entitet pomnilnega medija.

### *3.2 Churcheva realizacija pomnjenja v zapisu DNA*

V delu [3] Church s soavtorji opiše primer hrambe podatkov neposredno v zapisu DNA, v katerega zakodirajo HTML verzijo dela [6] v obsegu $53.426$ besed in $11$ JPEG slik. Skupna količina podatkov obsega zaporedje $5,27$ Mb. Avtorji se pri realizaciji odločijo za manj ekonomičen pristop kodiranja, kot je bil predstavljen v predhodnem razdelku, in sicer za kodiranje logične vrednosti 0 izbirajo med nukleotidoma **A** in **C**, za kodiranje logične vrednosti 1 pa med nukleotidoma **T** in **G**. S tem se skušajo izogniti štirim ali več sosednim ponovitvam posameznega nukleotida in uravnovešajo frekvenco podzaporedij **CG** na ciljnem kodnem zaporedju. Omejitvi izhajata iz značilnosti postopkov sinteze DNA (pisanja ciljnega zapisa), sekvenciranja DNA (branja ciljnega zapisa) in same stabilnosti molekule DNA. Tako realizacija izgubi na gostoti zapisa, ker je dolžina podatkovnega zapisa v bitni obliki enaka dolžini nukleotidnega zapisa.

Z arhitekturnega vidika realizacije pomnjenja avtorji podatkovni zapis razdelijo na $96$ bitov dolga podzaporedja, vsakemu podzaporedju pa dodajo $19$ bitni naslov, ki označuje lego podatkovnega podzaporedja v celotnem zaporedju zapisa. Sočasno zaradi narave pisanja in branja podzaporedju dodajo še $44$ kontrolnih bitov. Tako pridejo do segmentiranega načina hranjenja podatkov v segmentih po $159$ bitov (ciljnih nukleotidov) in končne dolžine zapisa

$$(96 + 19 + 44) * 54.898 = 8.728.782, \qquad (1)$$

kjer je zapis segmentiran v $54.898$ nukleotidnih podzaporedij (angl. *chunks*), vsota dolžin vseh segmentov zapisa pa $8.728.782$ nukleotidov. Podatkovni (kodirani) del je sestavljen iz $(96 * 54.898)$ $5.270.208$ nukleotidov. V primerjavi z načinom kodiranja, opisanim v razdelku 3.1, so avtorji tako drastično zmanjšali gostoto zapisa.

### *3.3 Goldmanova realizacija pomnjenja v zapisu DNA*

V delu [4] Goldman s soavtorji opiše primer hrambe podatkov neposredno v zapisu DNA, v katerega zakodirajo $739$ KB podatkovne vsebine vseh $154$ Shakespearovih sonetov v obliki ASCII, temu pa dodajo še en članek PDF, eno datoteko JPEG, izsek iz govora Martina Luthra Kinga iz leta $1963$ v formatu MP3 in izsek o kodiranju v formatu Huffmanovega koda. Pri svojem modelu načina zapisa jih vodijo naslednje smernice:

- podobno kot Church se avtorji zaradi lažjega obvladovanja ciljnega zapisa DNA (njegovega sintetiziranja – pisanja in sekvenciranja – branja) odločijo za segmentiran zapis ciljne vsebine v večje število krajših podzaporedij DNA,
- v postopku kodiranja v nukleotidni zapis se odpovedo možnosti sosednosti enakih nukleotidov v ciljnem zapisu zaradi zanesljivosti branja (sekvenciranja) in zaradi tega, ker naravna DNA takšne ponovitve običajno vsebuje; s tem je iz ciljnega nukleotidnega zapisa po kodiranju razvidno, da gre za umetno sintetizirano DNA,
- podatkovne podnize kodirajo večkratno v prekrivnem načinu, s čimer dosežejo štirikratno redundanco zapisa (vsak podatek je shranjen v štirih verzijah) in s tem posredno večjo zanesljivost branja (sekvenciranja) in daljšo življenjsko dobo zapisa.

| | 0 | 1 | 2 |
|---|---|---|---|
| A | C | G | T |
| C | G | T | A |
| G | T | A | C |
| T | A | C | G |

Tabela 2: Kodirna shema tritov v nabor nukleotidov [4]

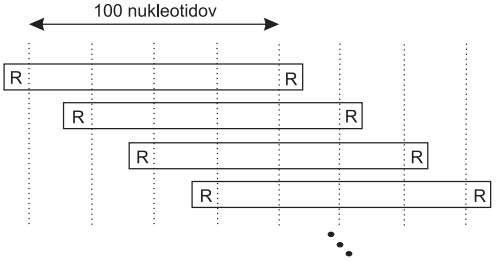
Z arhitekturnega vidika realizacije pomnjenja avtorji ciljni nukleotidni zapis oblikujejo po naslednjih korakih in pravilih:

- posamezni bajt izvirnega podatkovnega zapisa prekodirajo v zaporedje petih trivrednostnih entitet – tritov (angl. *trits*) z zalogo vrednosti $K_1 = \{0, 1, 2\}$; te si kljub manjši zalogi vrednosti novega kodnega nabora lahko privoščijo zaradi narave izvirnega podatkovnega zapisa, ki temelji na podmnožici razširjenega kodnega nabora ASCII,
- posamezni trit prekodirajo v nukleotid po kodirni shemi, predstavljeni v tabeli 2 (levi stolpec tabele označuje predhodno zapisani nulkeotid $n_i$, desni del tabele pa izbiro nukleotida $n_{i+1}$ glede na vrednost trita), ki zagotavlja, da v ciljnem zapisu DNA ne bo enakih sosednih nukloetidov,
- izhodiščni podatkovni nabor z dolžino $739$ KB ($757.051$ B) tako prevedejo na zaporedje $3.785.255$ $(5 * 757.051)$ nukleotidov; te razdelijo na podzaporedja po $100$ nukleotidov, vsako od njih pa opremijo z dodatnimi $17$ kontrolnimi (CRC) in adresnimi nukleotidi; tovrstnih podzaporedij je $37.853$,
- v zadnji fazi vsak podniz z dolžino $117$ nukleotidov sintetizirajo v prekrivnem načinu (sosednja podniza se prekrivata v $75$ nukleotidih) v štirikratni redundanci (glej sliko 1); s tem pridejo do $151.348$ nukleotidnih podzaporedij dolžine $117$ nukleotidov, tako da je vsota vseh podazporedij $17.707.716$ nukleotidov; sosednji prekrivni podnizi so kodirani na podlagi Cricks-Watsonovega komplementa.

Realizacija ni najbolj kompaktna z vidika gostote zapisa, po drugi plati pa ima vgrajeno redundanco in se ciljni zapis zaradi omejitev evidentno razlikuje od DNA, nastale po naravni evoluciji v živečem gostitelju.

### *3.4 Narava pomnjenja v zapisu DNA*

V predhodnih razdelkih smo opisali dva primera realizacije pomnjenja podatkov neposredno v zapisu DNA. Pri tem moramo poudariti, da sta oba zgleda primera *in-vitro* pomnjenja v kontroliranem laboratorijskem okolju, kjer medija DNA ne gosti živeči gostitelj (celica). Povedano drugače, medij hrambe ni osnova za življenje

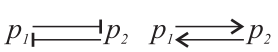



Slika 1: Prekrivna redundanca sintetizirane DNA [4]. Sosednji prekrivni podnizi so kodirani na podlagi Cricks-Watsonovega komplementa ($A = \overline{T}, T = \overline{A}, C = \overline{G}, G = \overline{C}$). Segment R so redundančni podatki neprekrivne narave.

in ekspresijo kemijskih zvrsti nekega živečega organizma. Avtorji se *in-vitro* koncepta poslužujejo zaradi *nedefinirane ekspresije* sintetizirane DNA v gostitelju in možnosti njenih *mutacij* skozi evolucijo gostitelja.

## 4 ZAČASNO POMNJENJE PODATKOV NA PODLAGI VZDRŽEVANJA KONCENTRACIJ KEMIJSKIH ZVRSTI

Začasno pomnjenje dvovrednostnih podatkov temelji na vzdrževanju visokih ali nizkih nivojev koncentracij kemijskih zvrsti (npr. proteinov), ki jih izražajo (ekspresirajo) posamezni geni zapisa DNA. Pogoj za tovrstno pomnjenje je bistabilno delovanje, ki ga dosežemo s pozitivno povratno zanko. To realiziramo bodisi z dvojno represijo ali dvojnim aktiviranjem (glej sliko 2) [7]. Pri




Slika 2: Shemi izvedbe začasnega pomnjenja preko dvojne represije (levo) oziroma dvojne aktivacije (desno), pri čemer $\rightarrow$ označuje aktivacijo (spodbujanje), $\dashv$ pa represijo (inhibicijo ali zaviranje) izražanja reguliranih proteinov $p_1$ in $p_2$.

takšnih izvedbah podatki niso shranjeni neposredno v DNA, je pa v DNA shranjeno *navodilo* ali program za *izražanje* gena, zato govorimo o *posrednem pomnjenju* na podlagi DNA. Ker je naše izhodišče pomnjenje dvovrednostnih podatkov iz kodnega nabora $\{0, 1\}$, potrebujemo za vsak tovrstni podatek ustrezen nosilec – kemijsko zvrst (v primeru s slike 2 sta to proteina $p_1$ in $p_2$). Vsaki od njih moramo določiti tudi mejne koncentracije, da bi znali ustrezno interpretirati logične vrednosti. Nekaj primerov modelov tovrstnega načina pomnjenja najdemo v delih [8], [9], primer realizacije kratkotrajnega pomnjenja, tj. izvedbo pomnilne celice in njeno razširitev v Johnsonov števec, pa predstavimo v naslednjem razdelku.

## 5 MODEL NASLOVLJIVEGA BIOLOŠKEGA CELIČNEGA PROGRAMA

V razdelku 3 smo našteli prednosti trajnega pomnjenja podatkov neposredno v *izoliranem* zapisu DNA. Tovrstna hramba bi imela še večjo uporabno vrednost, če bi bilo mogoče podatke shranjevati v zapis DNA *žive* celice, do njih dostopati (angl. *Read-Only Memory*, ROM) ali jih tekom časa celo spreminjati (angl. *Programmable-Read-Only Memory*, PROM). Zapisovanje manjše količine podatkov in njihova uporaba pri procesiranju sta bila v preteklosti že predstavljena [10]–[13]. V nadaljevanju predstavimo nadgradnjo obstoječih pristopov z modelom trajno hranjenega biološkega celičnega programa, ki ga naslavljamo prek začasno hranjene vrednosti biološkega števca.

*Števec* je eden osnovnih funkcionalnih gradnikov avtomatiziranih sistemov. Namenjen je štetju izbranih dogodkov. Njegovo stanje obravnavamo kot vrednost spremenljivke, ki jo moramo skozi čas pomniti in jo občasno spremeniti (povečati ali zmanjšati). V zadnjem času je že bilo predstavljenih nekaj bioloških izvedb različnih števcev, ki temeljijo na začasnem pomnjenju in ki v kontekstu sintezne biologije obetajo številne nove aplikacije (npr. štetje celičnih delitev) [14]–[16]. Omenjeni števci so s stališča izvedbe neoptimalni, saj je za predstavitev vrednosti enega števca potrebnih $n$ kemijskih zvrsti (npr. proteinov), pri čemer se stanje ali vrednost števca manifestira s prisotnostjo (visoko koncentracijo) ene od kemijskih zvrsti in hkratno odsotnostjo (nizko koncentracijo) preostalih $n - 1$ kemijskih zvrsti (angl. *one-hot encoding*). Tako lahko z $n$ kemijskimi zvrstmi zakodiramo zgolj $n$ različnih stanj posameznega števca.

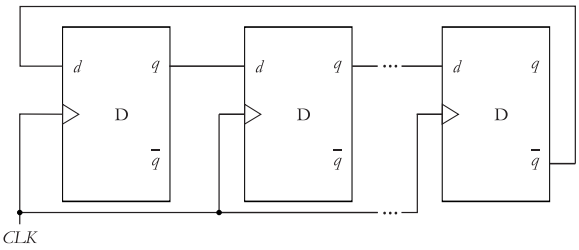
### 5.1 Zasnova izboljšanega biološkega števca

Predpostavimo, da imamo v zapisu DNA žive celice množico genov, pri čemer vsak od njih definira ekspresijo ali izražanje nekega izhodnega proteina. Množico genov imamo za trajno hranjen *celični program*, *izhodne proteine* pa za različne *odzive* celičnega programa. Celični program *naslavljamo* z biološkim programskim števcem, katerega predlog izvedbe in model podamo v nadaljevanju. Pod terminom naslavljanja imamo v mislih izbiro gena, ki na tekočem koraku izvaja ekspresijo ali izražanje izhodnega proteina.

Programski števec v nadaljevanju realiziramo kot Johnsonov števec. Za njegovo izvedbo razen pomnilnih celic ne potrebujemo nikakršne dodatne logike (glej sliko 3). S takšno realizacijo z $n$ *notranjimi proteini* zakodiramo $2n$ različnih stanj števca.

### 5.2 Realizacija biološke pomnilne celice D

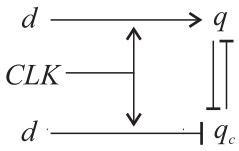
Za izvedbo prej omenjenega Johnsonovega števca potrebujemo realizacijo celice D v biološkem sistemu (npr. v bakterijski celici). To realiziramo z uporabo dveh medsebojno represirajočih se (komplementarnih)




Slika 3: Izvedba Johnsonovega števca z uporabo pomnilnih celic D, ki so občutljive na pozitivno fronto urinega signala ($CLK$). Zaradi preglednosti je signal $RESET$ iz sheme izpuščen.

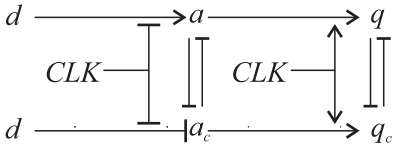
proteinov $q$ in $q_c$, s pomočjo katerih dosežemo bistabilno delovanje. Predpostavimo, da za vsakega od obeh proteinov velja, da ga ekspresirata po dva gena. Prvi gen je aktiven takrat, ko je odsoten komplementarni protein ($q_c$ za $q$ in nasprotno). Drugi gen je aktiven samo ob prisotnosti urinega signala. Če je hkrati aktiven še vhod v celico ($d$), bo aktivno izražanje proteina $q$, v nasprotnem primeru pa proteina $q_c$. Biološka shema realizacije celice D je predstavljena na sliki 4, njeno delovanje pa opišemo z izrazoma

$$q = \overline{q}_c \vee d \cdot CLK, \qquad (2)$$
$$q_c = \overline{q} \vee \overline{d} \cdot CLK.$$




Slika 4: Shema izvedbe pomnilne celice D z uporabo genske regulacije - aktiviranja in represije

Problem predlagane sheme je v možnosti porajanja neželenih večkratnih preklopov stanja celice D v eni urini periodi. Izvedba namreč predpostavlja, da je preklop števca prožen s prisotnostjo urinega signala, ki indicira določen dogodek v sistemu. Ker je preklop predlagane celice D prožen na podlagi nivoja signala, mora biti dolžina visokega nivoja urinega signala (pulza) za pravilno delovanje vezja v točno določenem intervalu. Pulz mora biti dovolj dolg, da celica D izvede ves preklop, in hkrati dovolj kratek, da celica ne izvede več preklopov hkrati [14]. Bolj robustno delovanje lahko dosežemo z vezavo *Master-Slave* pomnilnih celic D, s čimer pridemo do vezja, ki je proženo na fronto in ne zgolj na nivo ure [17] (glej sliko 5). V tej izvedbi uporabimo dve pomnilni celici s slike 4, tj. vhodno (angl. *Master*) in izhodno (angl. *Slave*). Vhodna celica je prožena z zunanjim vhodom $d$ in nizkim nivojem urinega signala $CLK$ ter za pomnjenje uporablja komplementarna proteina $a$ in $a_c$. Izhodna celica je prožena s stanjem vhodne celice (določeno z $a$ in $a_c$) in visokim nivojem urinega signala $CLK$ ter za pomnjenje uporablja komplementarna proteina $q$ in $q_c$. Tako pridemo do pomnilne celice, ki je prožena na pozitivno fronto urinega signala.




Slika 5: Shema izvedbe *Master-Slave* pomnilne celice D z uporabo genske regulacije

### 5.3 Matematični model biološke pomnilne celice D

Pravilnost delovanja predlagane realizacije lahko preverimo s simulacijo na podlagi njenega matematičnega modela, ki temelji na sistemu navadnih diferencialnih enačb (angl. *Ordinary Diferential Equations*, ODE). Model izvedbe *Master-Slave* pomnilne celice D lahko podamo z izrazi

$$\begin{aligned}
\frac{da}{dt} = &\quad \alpha_1 \cdot \Theta(d - K_{d_1}) \cdot \Theta(K_{d_2} - CLK) \\
&+\alpha_2 \cdot \Theta(K_{d_3} - a_c) - \delta \cdot a, \qquad (3)\\
\frac{da_c}{dt} = &\quad \alpha_1 \cdot \Theta(K_{d_1} - d) \cdot \Theta(K_{d_2} - CLK) \\
&+\alpha_2 \cdot \Theta(K_{d_3} - a) - \delta \cdot a_c,\\
\frac{dq}{dt} = &\quad \alpha_1 \cdot \Theta(a - K_{d_1}) \cdot \Theta(CLK - K_{d_2}) \\
&+\alpha_2 \cdot \Theta(K_{d_3} - q_c) - \delta \cdot q,\\
\frac{dq_c}{dt} = &\quad \alpha_1 \cdot \Theta(a_c - K_{d_1}) \cdot \Theta(CLK - K_{d_2}) \\
&+\alpha_2 \cdot \Theta(K_{d_3} - q) - \delta \cdot q_c,
\end{aligned}$$

kjer je $\Theta$ funkcija enotine stopnice, $CLK$, $a$, $a_c$, $q$ in $q_c$ določajo koncentracije opazovanih proteinov, $\alpha_1$ in $\alpha_2$ hitrosti genskega izražanja, $K_{d_1}$, $K_{d_2}$ in $K_{d_3}$ disociacijske konstante, ki opisujejo mejo aktiviranja oziroma represije gena, $\delta$ pa hitrosti degradacije proteinov. Predpostavljamo, da imajo opisani parametri pri vseh štirih proteinih enake vrednosti. Podrobnejša razlaga vzpostavljenega modela presega obseg tega dela. Izhodišča za vzpostavitev bralec najde v [7], [9].

### 5.4 Izvedba biološkega Johnsonovega števca

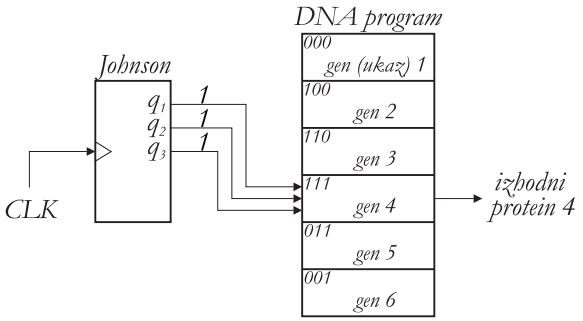
Biološko realizacijo Johnsonovega števca lahko izvedemo z umestitvijo biološke pomnilne celice D s slike 5 v shemo, ki je predstavljena na sliki 3. Vsi segmenti pomnjenja so v končnem vezju izvedeni z enakim številom logičnih nivojev, s čimer se izognemo problemom različnih zakasnitev.

V nadaljevanju matematični model biološke pomnilne celice D umestimo v izvedbo tribitnega Johnsonovega števca. Predpostavimo, da za vrednosti kinetičnih parametrov povzamemo vrednosti, ki so v rangu povprečnih

vrednosti parametrov, značilnih za bakterijske celice (npr. $\alpha_1 = 5$, $\alpha_2 = 2,5$, $K_{d_1} = 10$, $K_{d_2} = 5$, $K_{d_3} = 0,5$, $\delta = 0,1$). Simulacijski rezultati delovanja sheme so pridobljeni na podlagi prej navedenega modela in vrednosti parametrov in so prikazani na sliki 6. Iz nje je razvidna pravilna sekvenca štetja $(000, 100, 110, 111, 011, 001)$.

### *5.5 Opis delovanja celičnega programa*

Izvedbo Johnsonovega števca uporabimo za naslavljanje (izbiro) aktivnega gena, ki je del (ukaz) celičnega programa. Sekvenco izvedbe ukazov (genskih ekspresij) določa zaporedje štetja, prehod na izvedbo naslednjega ukaza pa je prožen s pozitivno fronto urinega signala. Predlagano shemo naslavljanja prikazuje slika 7.




Slika 7: Shema naslavljanja celičnega programa z uporabo predlagane izvedbe Johnsonovega števca. Primer na sliki prikazuje naslavljanje ukaza na pomnilniški lokaciji 111.

## 6 SKLEP

V prispevku smo na začetku podali kratek pregled razvoja področja pomnjenja dvovrednostnih logičnih vrednosti v preprostih bioloških sistemih. V nadaljevanju smo predstavili model realizacije trajno pomnjenega celičnega programa, katerega posamezne ukaze (ekspresije genov) naslavljamo z začasno pomnjenim biološkim programskim števcem. Funkcijo trajnega pomnjenja opravlja zapis DNA, funkcijo začasnega pomnjenja pa vzdrževanje in preklapljanje koncentracij kemijskih zvrsti v gostitelju. Na podlagi idejne zasnove števca, ki ga realiziramo kot Johnosonov števec, predstavimo njegovo delovanje v tribitni različici, ki podpira zalogo šestih različnih stanj števca.

## ZAHVALA

Pričujoča raziskava je bila izvedena v okviru znanstveno raziskovalnega programa *Vseprisotno računalništvo* (P2-0359) in projekta *Načrtovana celična vezja* (J1-6740), ki ju v letih 2013–2017 financira Agencija za raziskave in razvoj Republike Slovenije.

## LITERATURA

[1] R. Gaber, T. Lebar, A. Majerle, B. Šter, A. Dobnikar, M. Benčina, and R. Jerala, "Designable DNA-binding domains enable construction of logic circuits in mammalian cells", *Nature Chemical Biology*, vol. 10, pp. 203–208, 2014.

[2] J. Stricker, S. Cookson, M. R. Bennett, W. H. Mather, L. S. Tsimring, and J. Hasty, "A fast, robust and tunable synthetic gene oscillator", *Nature*, vol. 456, pp. 516–520, 2008.

[3] G. M. Church, Y. Gao, and S. Kosuri, "Next-Generation Digital Information Storage in DNA", *Science*, vol. 337, p. 1628, 2012.

[4] N. Goldman, P. Bertone, S. Chen, C. Dessimoz, E. M. LeProust, B. Sipos, and E. Birney, "Towards practical, high-capacity, low-maintenance information storage in synthesized DNA", *Nature*, vol. 494, pp. 77–80, 2013.

[5] C. Bancroft, T. Bowler, B. Bloom, and C. Clelland, "Long-term storage of information in DNA", *Science*, vol. 293, no. 5536, pp. 1763–5, 2001.

[6] G. M. Church and E. Regis, *Regenesis: How Synthetic Biology Will Reinvent Nature and Ourselves*. Perseus Books Group, USA, 2012.

[7] U. Alon, *An Introduction to Systems Biology*. Chapman & Hall/CRC, 2007.

[8] M. Moškon and M. Mraz, "Systematic approach to computational design of gene regulatory networks with information processing capabilities", *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 11, no. 2, pp. 431–440, 2014.

[9] M. Moškon, "Modeli in metrike dinamike preklopa v enostavnih bioloških sistemih za potrebe računalniških struktur prihodnosti", Ph.D. dissertation, Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2012.

[10] P. Siuti, J. Yazbek, and T. K. Lu, "Synthetic circuits integrating logic and memory in living cells", *Nature Biotechnology*, vol. 31, no. 5, pp. 448–453, 2013.

[11] M. C. Inniss and P. A. Silver, "Building Synthetic Memory", *Current Biology*, vol. 23, pp. R812–R816, 2013.

[12] A. C. Maranhao and A. D. Ellington, "Endowing cells with logic and memory", *Nature Biotechnology*, vol. 31, no. 5, pp. 413–415, 2013.

[13] O. Purcell and T. K. Lu, "Synthetic analog and digital circuits for cellular computation and memory", *Current Opinion in Biotechnology*, vol. 29, pp. 146–155, 2014.

[14] A. E. Friedland, T. K. Lu, X. Wang, D. Shi, G. Church, and J. J. Collins, "Synthetic Gene Networks that Count", *Science*, vol. 324, no. 5931, pp. 1199–1202, 2009.

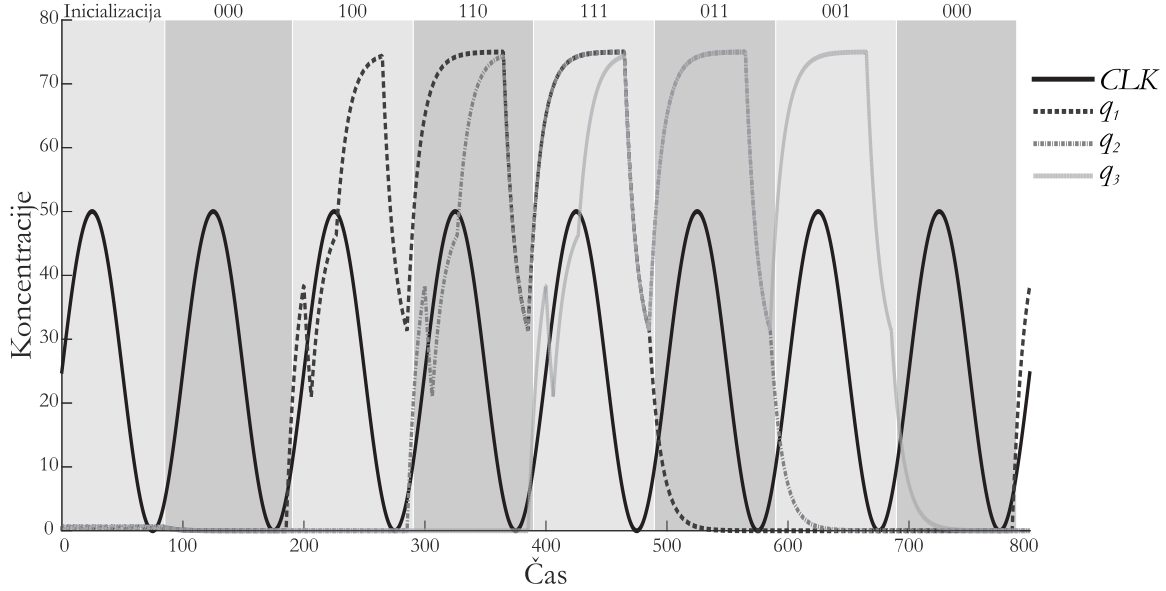
[15] M. Lehmann and K. Sneppen, "Genetic Regulatory Networks that count to 3", *Journal of Theoretical Biology*, vol. 329, pp. 15–19, 2013.

[16] C. Myhrvold, W. M. H. Jonathan W. Kotula and, N. J. Conway, and P. A. Silver, "A distributed cell division counter reveals growth dynamics in the gut microbiota", *Nature Communications*, vol. 6, pp. 1–10, 2015.

[17] M. Moškon, M. Ciglič, N. Zimic, and M. Mraz, "Toward in vivo digital synchronous sequential circuits", *WSEAS Transactions on Circuits and Systems*, vol. 8, no. 3, pp. 301–310, 2009.

**Miha Moškon** je docent na Fakulteti za računalništvo in informatiko v Ljubljani. Raziskovalno se ukvarja z računalniškimi metodami v sistemski in sintezni biologiji, natančneje z metodami za modeliranje, simulacijo, analizo in načrtovanje bioloških sistemov.

**Nikolaj Zimic** je redni profesor na Fakulteti za računalništvo in informatiko v Ljubljani. Raziskovalno se ukvarja s področjem alternativnih procesnih platform in metod ter računalniškimi komunikacijami.




Slika 6: Simulacijski rezultati na modelu trobitnega števca pri urinem signalu ($CLK$) s periodo 100. Meje med logično 0 in 1 se nahajajo pri koncentracijah $K_{d_1}$. Čas in koncentracije opazovanih proteinov so podani brez enot. Vrednosti kinetičnih parametrov so sledeče: $\alpha_1 = 5$, $\alpha_2 = 2,5$, $K_{d_1} = 10$, $K_{d_2} = 5$, $K_{d_3} = 0,5$, $\delta = 0,1$.

**Miha Mraz** je redni profesor na Fakulteti za računalništvo in informatiko v Ljubljani. Raziskovalno se ukvarja s področjem alternativnih procesnih platform in metod procesiranja ter sistemske zanesljivosti.