

# **Analiza in načrtovanje sekvenčnih bioloških vezij ter ostale teme s področja biološkega procesiranja**

Uredila izr. prof. dr. Miha Moškon in prof. dr. Miha Mraz

*Januar, 2023*



# Predgovor

Navodila za urejanje:

- Vaše datoteke se nahajajo v direktorijih `Skupina*`, kjer `*` predstavlja številko vaše skupine - glavna datoteka je `textttmain.text`.
- Če mape za vašo skupino še, jo ustvarite.
- Slike shranjujte v svoj direktorij.
- Vse labelle začnite z znaki `g* :`, kjer `*` predstavlja številko vaše skupine.
- Pri referenciranju virov uporabite datoteko `references.bib`, ki se nahaja v korenskem direktoriju projekta.
- Pred dodajanjem novih virov v datoteko `references.bib` dobro preverite, če je vir mogoče že vsebovan v datoteki - v tem primeru se sklicujte na obstoječ vnos.

Ljubljana,  
januar, 2026

*prof. dr. Miha Moškon*  
*prof. dr. Miha Mraz*



## **Zahvala**

*prof. dr. Miha Moškon in prof. dr. Miha Mraz, Ljubljana, januar 2026*



# Kazalo

<b>1</b>	<b>Implementacija biološkega 4-bitnega PIPO registra s preverjanjem paritete</b>	<b>9</b>
	Filip Gros, Marko Horvat, Andraž Klobučar	
1.1	Uvod	9
1.2	Biološki 4-bitni PIPO register	10
1.2.1	D flip-flop vezje	10
1.2.2	Biološki D flip-flop kot ODE model	10
1.2.3	4-bitni PIPO register	11
1.2.4	Pisni načini: WRITE_ALL, WRITE_i in HOLD	12
1.2.5	Implementacijski vidik (ODE model registra)	12
1.3	Izračun paritete	13
1.3.1	Mehanski vidik (implementacija v kodi)	13
1.3.2	Biološki vidik izračuna paritete	13
1.4	Scenariji testiranja napak	14
1.4.1	Scenarij 1: Sprememba vrednosti bita 3	15
1.4.2	Scenarij 2: Sprememba vrednosti bita 1 ob pravilni pariteti	16
1.5	Končni rezultat	17
1.6	Možne izboljšave	18
1.7	Zaključek	18
1.8	Doprinosi avtorjev	19
	<b>Literatura</b>	<b>21</b>





## Poglavje 1

# Implementacija biološkega 4-bitnega PIPO registra s preverjanjem paritete

Filip Gros, Marko Horvat, Andraž Klobučar

**Povzetek** V seminarski nalogi predstavimo biološko implementacijo 4-bitnega PIPO registra, zasnovanega kot biološki ekvivalent klasičnega digitalnega registra. Pomnilne celice modeliramo z navadnimi diferencialnimi enačbami, ki opisujejo produkcijo in razgradnjo regulatornih komponent ter s tem časovno dinamiko bitov. Register omogoča sočasni zapis in branje štirih bitov, poleg tega pa smo sistem nadgradili s sodo pariteto, ki zagotavlja osnovno zaznavanje napak. Delovanje smo preverili s simulacijo različnih pisalnih načinov ter s tremi scenariji umetno ustvarjenih napak na podatkovnih in paritetnem bitu. Rezultati kažejo, da ODE model registra zajame ključne lastnosti takšnega sistema in da preverjanje paritete lahko v določenih primerih pravilno zazna napake v zapisu. Izpostavili pa smo tudi nekaj omejitev v naši zasnovi sistema in predlagali možne izboljšave.

## 1.1 Uvod

Sintetična biologija predstavlja zanimiv pristop za nekonvencionalno procesiranje, kjer informacije kodiramo in obdelujemo ne z digitalnimi vezji ampak s pomočjo biokemijskih procesov. V takšnih sistemih je informacijo mogoče shraniti kot stabilno regulacijsko stanje ali celo kot trajnejši zapis v genetskem materialu, kar omogoča implementacijo pomnilnih enot v biološkem sistemu.

V tej seminarski nalogi predstavljamo implementacijo 4-bitnega PIPO (angl. Parallel In Parallel Out) registra, ki omogoča sočasni zapis in sočasno branje štirih bitov. Register smo zasnovali kot biološki sistem, ki je ekvivalenten klasičnemu

---

Filip Gros

Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, e-mail: fg1343@student.uni-lj.si

Marko Horvat

Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, e-mail: mh3269@student.uni-lj.si

Andraž Klobučar

Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, e-mail: ak1790@student.uni-lj.si

digitalnemu registru. Pomnilniške celice, ki sestavljajo PIPO register, smo implementirali s sistemom navadnih diferencialnih enačb, ki modelirajo produkcijo in razgradnjo regulatornih komponent, ter posledično časovni potek stanj, ki predstavljajo shranjene bite.

Ker pa so biokemijski procesi podvrženi šumu, variabilnosti in motnjam, kar povzroči neželene prehode stanj, smo implementacijo registra nadgradili z biološko zasnovanim osnovnim preverjanjem paritete. Tako smo uvedli preprosto redundanco, ki omogoča zaznavanje napak v zapisu informacij.

Našo implementacijo smo preizkusili s simulacijo, kjer smo izvedli sočasni zapis, selektivni zapis posameznih bitov, na koncu pa še pripravili nekaj primerov simuliranih napak na bitih, da smo preverili delovanje implementiranega mehanizma zaznavanja napak.

## 1.2 Biološki 4-bitni PIPO register

### 1.2.1 D flip-flop vezje

Temeljni gradnik uporabljen v registru je D flip-flop vezje (D celica). D flip-flop ima en podatkovni vhod  $D$ , uro  $CLK$  in izhod  $Q$ , ki shrani vhodno vrednost ob aktivni fronti ure. Idealno delovanje je opisano z relacijo

$$Q(t^+) = D(t) \quad \text{ob pozitivni fronti } CLK,$$

medtem ko v ostalem času celica ohranja shranjeno stanje [1]. V uporabljenem modelu je logična enica predstavljena z visoko analogno vrednostjo (100), logična ničla pa z nizko vrednostjo (0).

V biološki izvedbi je D flip-flop realiziran z gensko regulatornim omrežjem z bistabilno dinamiko, ki je sinhronizirano z oscilatorjem (uro). Koncentracije proteinskih zvrsti prevzemajo vlogo digitalnih spremenljivk, ura pa določa trenutke vzorčenja in zapisa v pomnilne celice.

### 1.2.2 Biološki D flip-flop kot ODE model

Biološko D flip-flop vezje je opisano z navadnimi diferencialnimi enačbami (ODE), ki modelirajo spremembe koncentracij regulatornih proteinov:

$$\frac{dY}{dt} = F(Y, t, \theta),$$

kjer vektor  $Y$  predstavlja koncentracije ključnih zvrsti,  $t$  je čas,  $\theta$  pa parametri sistema (npr. produkcijske in degradacijske konstante, disociacijske konstante ter Hil-

lovi koeficienti). Takšna formulacija je standardna za modeliranje gensko reguliranih sistemov [2].

Vsaka D celica vključuje štiri notranja stanja:

$$[a, \text{not\_}a, q, \text{not\_}q],$$

kjer  $q$  in  $\text{not\_}q$  predstavljata koncentraciji, ki kodirata logično stanje na izhodu  $Q$  in komplementarnem izhodu  $\bar{Q}$ , medtem ko  $a$  in  $\text{not\_}a$  predstavljata vmesni regulatorni par. Vmesni par interpretiramo kot biološki ekvivalent master-latch stopnje, ki pod vplivom vhoda  $d$  zavzame stanje, nato pa se ob aktivni uri to stanje prenese na izhodni par  $q$  in  $\text{not\_}a$ . Vhodna vrednost  $D$  je podana v analogni obliki (0 ali 100), skupni signal  $CLK$  pa sinhronizira zajemanje podatkov.

V nadaljevanju so podane diferencialne enačbe ene D celice, ki smo jih implementirali v funkciji `ff_ode_model`, kjer parametri  $\alpha_1, \alpha_2, \alpha_3, \alpha_4$  predstavljajo maksimalne produkcijske hitrosti,  $\delta_1$  in  $\delta_2$  degradacijske konstante,  $d$  je analogni podatkovni vhod,  $\text{clk}$  pa vrednost ure.

Enačba 1.1 opisuje dinamiko vmesnega stanja  $a$ , enačba 1.2 opisuje komplementarno dinamiko vmesnega stanja  $\text{not\_}a$ , enačba 1.3 opisuje dinamiko izhodnega stanja  $q$  in enačba 1.4 opisuje dinamiko komplementarnega izhoda  $\text{not\_}q$ .

V vseh štirih enačbah je zadnji člen linearen degradacijski prispevek, ki zagotavlja omejenost koncentracij. Nelinearna regulacija produkcije je opisana s Hillovimi členi, kjer parameter  $K_d$  določa prag občutljivosti (približno polovično aktivacijo), Hillov koeficient  $n$  pa strmost prehoda med nizkim in visokim odzivom.

$$\frac{da}{dt} = \alpha_2 \frac{\left(\frac{d}{K_d}\right)^n}{1 + \left(\frac{d}{K_d}\right)^n + \left(\frac{\text{clk}}{K_d}\right)^n + \left(\frac{d}{K_d}\right)^n \left(\frac{\text{clk}}{K_d}\right)^n} + \alpha_2 \frac{1}{1 + \left(\frac{\bar{a}}{K_d}\right)^n} - \delta_1 a \quad (1.1)$$

$$\frac{d\bar{a}}{dt} = \alpha_1 \frac{1}{1 + \left(\frac{d}{K_d}\right)^n + \left(\frac{\text{clk}}{K_d}\right)^n + \left(\frac{d}{K_d}\right)^n \left(\frac{\text{clk}}{K_d}\right)^n} + \alpha_2 \frac{1}{1 + \left(\frac{a}{K_d}\right)^n} - \delta_1 \bar{a} \quad (1.2)$$

$$\frac{dq}{dt} = \alpha_3 \frac{\left(\frac{a}{K_d}\right)^n \left(\frac{\text{clk}}{K_d}\right)^n}{1 + \left(\frac{a}{K_d}\right)^n + \left(\frac{\text{clk}}{K_d}\right)^n + \left(\frac{a}{K_d}\right)^n \left(\frac{\text{clk}}{K_d}\right)^n} + \alpha_4 \frac{1}{1 + \left(\frac{\bar{q}}{K_d}\right)^n} - \delta_2 q \quad (1.3)$$

$$\frac{d\bar{q}}{dt} = \alpha_3 \frac{\left(\frac{\bar{a}}{K_d}\right)^n \left(\frac{\text{clk}}{K_d}\right)^n}{1 + \left(\frac{\bar{a}}{K_d}\right)^n + \left(\frac{\text{clk}}{K_d}\right)^n + \left(\frac{\bar{a}}{K_d}\right)^n \left(\frac{\text{clk}}{K_d}\right)^n} + \alpha_4 \frac{1}{1 + \left(\frac{q}{K_d}\right)^n} - \delta_2 \bar{q} \quad (1.4)$$

### 1.2.3 4-bitni PIPO register

4-bitni PIPO (Parallel In Parallel Out) register je sestavljen iz štirih D flip-flop celic, ki delijo skupni signal ure. Na digitalni ravni register prejme paralelni vhodni vektor

$D = [D_0, D_1, D_2, D_3]$  in ob aktivnem robu ure shrani vrednosti v pomnilne celice, medtem ko v odsotnosti pisanja ohranja stanje [1].

V biološki izvedbi ima vsak bit par koncentracij  $(q_i, not\_q_i)$  za  $i \in \{0, 1, 2, 3\}$ , ki predstavljata logično stanje. Model vključuje tudi peto D celico, v kateri je shranjen paritetni bit štirih podatkovnih bitov. Paritetni flip-flop deluje po isti dinamiki kot ostale celice in je podrobneje opisan v poglavju o pariteti. Podoben pristop zasnove bioloških registrskih vezij je podan v [3].

### 1.2.4 Pisni načini: **WRITE\_ALL**, **WRITE\_i in HOLD**

**WRITE\_ALL** — v tem načinu se v istem časovnem oknu zapišejo vsi štirje podatkovni biti. Za vse bitne celice velja  $D_i^{\text{eff}} = D_i^{\text{in}}$ , kar predstavlja neposreden prenos paralelnega vhoda.

**WRITE\_i** — v tem načinu se zapis izvede samo za izbrani bit  $i$ , ostali biti pa ohranijo trenutno stanje. Za izbrani bit velja  $D_i^{\text{eff}} = D_i^{\text{in}}$ , za ostale pa  $D_j^{\text{eff}} = Q_j$  zaradi povratne povezave  $Q \rightarrow D$ .

**HOLD** — kadar ni aktivnega pisalnega signala, je register v načinu **HOLD**. V tem primeru za vse bite velja  $D_i^{\text{eff}} = Q_i$ , kar omogoča ohranjanje stanja tudi ob prisotnosti ure.

### 1.2.5 Implementacijski vidik (ODE model registra)

Realizacija registra temelji na enotnem ODE modelu, ki zajema dinamiko štirih podatkovnih D flip-flop celic ter paritetne celice. Celoten vektor stanj  $Y$  vsebuje 20 dinamičnih spremenljivk, in sicer sestavljenih iz

- štirih podatkovnih D celic:  $[a_i, not\_a_i, q_i, not\_q_i]$ ,
- paritetne D celice:  $[a_p, not\_a_p, q_p, not\_q_p]$ .

V vsakem časovnem koraku se izvede:

1. določitev pisalnih signalov (**WRITE\_ALL**, **WRITE\_i**, **HOLD**);
2. določitev efektivnih vhodnih vrednosti  $D_i^{\text{eff}}$ ;
3. izračun digitaliziranih bitov in paritetnega bita;
4. posodabljanje stanj D celic z uporabo `ff_ode_model`;
5. združitev rezultatov v vektor odvodov  $dY$ .

Tako pridobljen model ustreza digitalnemu 4-bitnemu PIPO registru z dodatnim paritetnim bitom, pri čemer so vsi signali realizirani kot zvezne proteinske koncentracije, dinamika pa je opisana z navadnimi diferencialnimi enačbami.

### 1.3 Izračun paritete

#### 1.3.1 Mehanski vidik (implementacija v kodi)

V projektu je pariteta implementirana kot sodo-paritetni mehanizem (even parity), pri katerem je paritetni bit določen kot ekskluzivni ALI (XOR) vseh štirih podatkovnih bitov. Matematično je paritetni bit definiran kot

$$\text{Paritetni\_bit} = \text{bit}_0 \oplus \text{bit}_1 \oplus \text{bit}_2 \oplus \text{bit}_3,$$

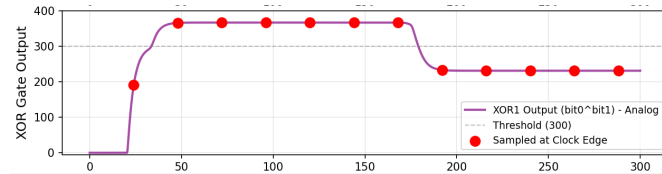
kjer operator  $\oplus$  predstavlja XOR operacijo. Takšna definicija zagotavlja, da je skupno število enic v podatkovnih bitih in paritetnem bitu vedno sodo.

Ker model temelji na zveznih dinamičnih enačbah, so izhodne vrednosti flip-flopov podane v obliki analognih proteinskih koncentracij. Vsak flip-flop ima dva izhoda,  $Q$  in  $\bar{Q}$ , ki predstavljata aktivno in neaktivno stanje. Digitalna vrednost posameznega bita se določi z relacijo

$$\text{bit}_i = \begin{cases} 1, & Q_i > \bar{Q}_i, \\ 0, & \text{sicer.} \end{cases}$$

Na osnovi teh digitaliziranih vrednosti se paritetni bit izračuna s kaskadno XOR logiko. Izračunana pariteta se nato pretvori v analogni ciljni nivo paritetnega flip-flopa, pri čemer vrednost 100 predstavlja logično enico, vrednost 0 pa logično ničlo.

Paritetni bit se shrani v ločenem, petem flip-flopu in se posodobi ob vsakem zapisu katerega koli podatkovnega bita. Ta proces je krmiljen s signalom `parity_write_enable`, ki je aktiven bodisi ob globalnem zapisu bodisi ob zapisu posameznega bita. Shranjena pariteta tako predstavlja referenčno stanje sistema, izračunano na osnovi vrednosti podatkovnih bitov v trenutku zapisa.



Slika 1.1 Operacija XOR nad bitoma 0 in 1

#### 1.3.2 Biološki vidik izračuna paritete

Kot je bilo podrobno opisano v poglavju 1.2.2, so posamezni biološki D flip-flopi modelirani z navadnimi diferencialnimi enačbami, ki opisujejo dinamiko koncen-

tracij regulatornih proteinov in omogočajo stabilno shranjevanje logičnih stanj. Na tej osnovi je v obravnavanem sistemu realiziran tudi paritetni mehanizem.

V biološkem kontekstu pariteta ne predstavlja diskretne logične operacije, temveč emergentno lastnost medsebojno povezanih proteinskih kaskad. Logično stanje posameznega bita je določeno z razmerjem koncentracij izhodnih proteinov  $Q$  in  $\bar{Q}$ , pri čemer prevlada ene izmed zvrsti določa digitalno interpretacijo stanja. Paritetni bit je generiran kot rezultat kombinacije teh signalov, pri čemer je logična XOR funkcija realizirana s pomočjo aktivacijskih in represijskih interakcij med regulatornimi proteini.

Te interakcije so opisane s Hillovimi funkcijami, ki modelirajo kooperativno vezavo regulatorjev na promotorske regije DNA. Hillov koeficient  $n$  določa ostrino prehoda med nizkim in visokim koncentracijskim režimom ter s tem stopnjo približevanja idealizirani digitalni logiki. V našem primeru ima vrednost 6.0. Višje vrednosti koeficienta  $n$  vodijo v bolj izrazite preklopne lastnosti, kar je ključno za zanesljivo implementacijo XOR logike v biološkem sistemu.

Paritetni flip-flop je strukturiran enako kot podatkovni flip-flopi in uporablja identičen ODE model. Njegov vhodni signal predstavlja analogno vrednost, ki ustreza XOR kombinaciji vseh štirih podatkovnih bitov. Za razliko od klasičnih digitalnih vezij se pariteta v takšnem sistemu ne vzpostavi trenutno, temveč se koncentracije proteinov postopoma prilagajajo vhodnim signalom. Zaradi tega ima časovna dinamika sistema ključno vlogo, saj mora biti zagotovljen zadosten čas stabilizacije, preden se stanje vzorči in interpretira kot veljavna logična vrednost.

## 1.4 Scenariji testiranja napak

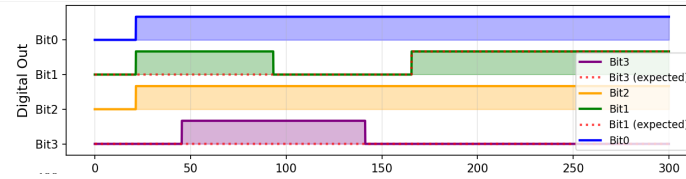
V nadaljevanju so predstavljeni trije deterministični scenariji napak, s katerimi se preverja sposobnost sistema za zaznavanje nepravilnosti v delovanju. Napake so namerno vnesene v natančno določenih časovnih intervalih in ne temeljijo na naključnem šumu, kar omogoča ponovljivo in jasno analizo odziva sistema.

Sistem je zasnovan tako, da simulira dve enaki celici: **celico**, ki predstavlja dejanski sistem, in **referenčno celico**, ki v našem primeru računa pričakovane vrednosti sistema. Obe celici imata enako arhitekturo, ki vključuje:

- 4-bitni PIPO register za hranjenje bitov,
- kaskado XOR vrat za izračun paritete,
- shranjen paritetni bit.

Vse komponente celice in referenčne celice so arhitekturno in funkcionalno enake. Obe celici prejemata enake WRITE signale ob istih časovnih trenutkih, kar zagotavlja neposredno primerljivost njunega delovanja.

Na grafu *Digital Out* so izhodi celice prikazani s polnimi barvnimi črtami in predstavljajo dejanske (merjene) vrednosti bitov. Referenčna celica predstavlja pričakovano delovanje sistema. Pri referenčni celici sta vrednosti bita 0 in bita 2 enaki kot pri celici, medtem ko sta bita 1 in 3 namerno spremenjena. Njuni



**Slika 1.2** Graf vrednosti bitov celice in referenčne celice (rdeča črtkana črta)

pričakovani vrednosti sta prikazani z rdečimi pikčastimi črtami (*expected value*). Prav razlika med dejanskimi signali celice in pričakovanimi signali referenčne celice omogoča simulacijo napak na posameznih bitih.

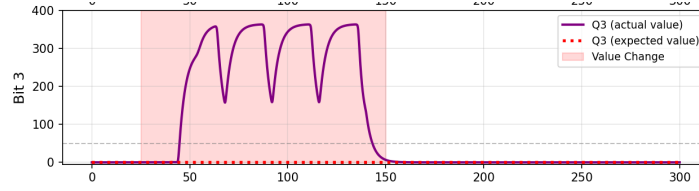
Celica in referenčna celica neodvisno računata pariteto na podlagi svojih trenutnih vrednosti bitov. Na grafu *Parity Value* so z rdeče obarvanimi območji označena časovna obdobja neujemanja paritete, kar pomeni, da se izračunana pariteta celice ne ujema s pariteto referenčne celice. Iz teh območij je mogoče ugotoviti, da je prišlo do napačnega izračuna paritete, hkrati pa tudi časovno določiti pojav napake oziroma okvare, ki je posledica sprememb na bitih 1 in 3. V nadaljevanju so posamezni scenariji napak obravnavani ločeno in podrobneje analizirani.

#### 1.4.1 Scenarij 1: Sprememba vrednosti bita 3

V prvem scenariju je v časovnem intervalu  $t = 25\text{--}150$  tretji bit ( $q_3$ ) dejanske celice prisilno nastavljen na visoko vrednost, čeprav bi glede na pričakovane vhodne podatke moral ostati v nizkem stanju. Na ta način se simulira deterministična napaka v podatkovnem bitu, medtem ko referenčna (expected) celica v istem časovnem intervalu ohranja obstoječo vrednost bita  $q_3$ .

Razlika med dejansko in pričakovano vrednostjo bita  $q_3$  povzroči neujemanje med izračunano in shranjeno pariteto. To neujemanje se na grafu *Parity Value* odrazi kot aktivacija indikatorja napake, kar potrjuje pravilno delovanje mehanizma za zaznavanje nepravilnosti.

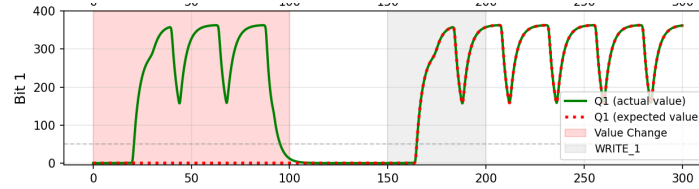
Po koncu intervala napake se dejanska vrednost bita  $q_3$  ponovno ujema s pričakovano vrednostjo, pariteta se stabilizira, sistem pa preide v fazo okrevanja, v kateri indikator napake ni več aktiven.



Slika 1.3 Sprememba vrednosti bita  $q_3$  med  $t = 25\text{--}150$

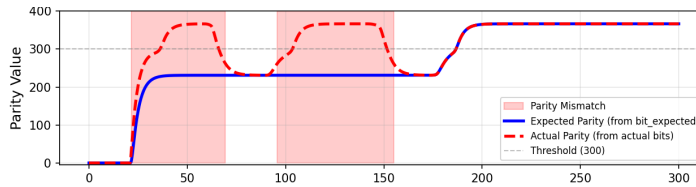
### 1.4.2 Scenarij 2: Sprememba vrednosti bita 1 ob pravilni pariteti

Tretji scenarij obravnava kompleksnejši primer, v katerem se napake pojavijo na več podatkovnih bitih v različnih časovnih intervalih. V časovnem intervalu  $t = 25\text{--}150$  je bit 3 ( $q_3$ ) že okvarjen, medtem ko je v intervalu  $t = 0\text{--}100$  prisilno spremenjena tudi vrednost prvega bita ( $q_1$ ).



Slika 1.4 Sprememba vrednosti bita  $q_1$  med  $t = 0\text{--}100$

Zaradi sočasne prisotnosti dveh napak v podintervalu med  $t = 45\text{--}90$  se pariteta izračuna navidezno pravilno. Napaki na bitih  $q_1$  in  $q_3$  medsebojno izničita v smislu sode paritete, zato se izračunana pariteta ujema s shranjeno referenčno vrednostjo, kljub temu da je dejansko stanje sistema napačno. Ker pa je paritetna napaka prisotna tako pred kot po tem navidezno pravilnem intervalu, je mogoče na podlagi časovne analize sklepati, da gre za območje lažno pravilnega delovanja.



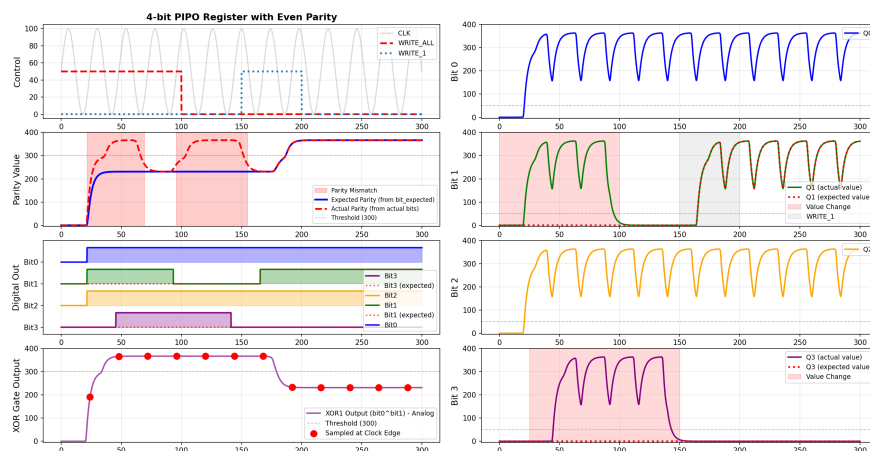
Slika 1.5 Prikaz navidezno pravilnega izračuna paritete med  $t = 45\text{--}90$

Takšen scenarij jasno pokaže omejitve enostavne paritetne zaščite v dinamičnih sistemih, kjer se napake lahko časovno prekrivajo. Z natančnejšo analizo časovnih vzorcev napak, zlasti z upoštevanjem njihove pojavnosti pred in po določenem in-



tervalu, bi bilo mogoče sistem nadgraditi in zaznavati tudi primere lažno pravilno izračunane paritete.

## 1.5 Končni rezultat



Slika 1.6 Končni rezultat izvedene programske kode

Na sliki je prikazan celoten rezultat simulacije 4-bitnega PIPO registra s sodo pariteto, modeliranega z ODE pristopom, ki aproksimira delovanje biološkega sistema. Simulacija vključuje **celico**, ki predstavlja dejanski sistem, ter **referenčno celico**, ki računa pričakovane vrednosti sistema. Prikazani so krmilni signali (ura in signali za zapis), analogna dinamika posameznih bitov, njihova digitalna interpretacija ter izračunana in shranjena pariteta obeh celic.

Slika ponazarja časovno dinamiko sistema, vključno z obdobji stabilnega delovanja, zapisi novih vrednosti ter namerno vnesenimi napakami na posameznih bitih celice. Posebej so izpostavljeni časovni intervali, v katerih pride do odstopanj med vrednostmi bitov celice in pričakovanimi vrednostmi referenčne celice. Ta odstopanja neposredno vplivajo na izračun paritete in sprožijo zaznavo paritetne napake, kar je na grafu *Parity Value* označeno z obarvanimi območji neujemanja.

Prikaz paritete razkriva tako pravilno zaznane napake kot tudi primere navidezno pravilne paritete, do katerih pride zaradi časovnega prekrivanja več napak na različnih bitih.

Rezultati potrjujejo, da ODE model uspešno zajame ključne lastnosti bioloških sistemov, kot so zakasnitev, postopna stabilizacija in občutljivost na časovno usklajenost signalov. Hkrati simulacija jasno pokaže omejitve enostavne paritetne zaščite ter nakazuje možnosti za nadaljnje nadgradnje, kot so časovna analiza napak, razširitev zaznavnih mehanizmov in modularna nadgradnja registra.

## 1.6 Možne izboljšave

Na osnovi opisanih scenarijev je mogoče sistem nadgraditi z algoritmom za sledenje napakam, ki temelji na analizi časovnih intervalov, v katerih je zaznana paritetna napaka. Z opazovanjem začetka in konca intervala napake je mogoče oceniti njeno trajanje, pogostost ter verjetno lokacijo napake znotraj registra. Takšen pristop omogoča razlikovanje med kratkotrajnimi motnjami in vztrajnimi napakami, ki so posledica sistemskih okvar.

Če je napaka prisotna tako na začetku kot na koncu opazovanega intervala, jo lahko obravnavamo kot persistentno, kar bistveno zmanjša negotovost ocene. Daljše in ponavljajoče se napake povečujejo zanesljivost sklepa o okvari določenega bita, medtem ko kratkotrajni pojavi, ki ne trajajo dovolj dolgo za stabilizacijo ODE sistema, povečujejo negotovost in jih je smiselno obravnavati kot prehodne motnje.

Dodatno bi projekt pridobil na uporabni vrednosti in biološki smiselnosti z razširitvijo registra na večje število podatkovnih bitov. V takem primeru neposredna kaskada XOR, katere dolžina je enaka dolžini registra, postane biološko težko izvedljiva zaradi kompleksnosti regulacijskih povezav in povečanih časov stabilizacije. Namesto tega bi bila primernejša hierarhična zasnova paritetnega mehanizma, pri kateri bi se pariteta najprej izračunala za manjše skupine bitov, na primer za zaporedne bloke po štiri bite.

Pariteta celotnega registra bi se nato določila z nadaljnjo kombinacijo paritet posameznih blokov. Takšen pristop omogoča ponovno uporabo iste XOR kaskade za več delov sistema, kar bistveno poenostavi biološko implementacijo in zmanjša število potrebnih regulacijskih interakcij.

Bi pa v pravem biološkem sistemu večje število bitov povečalo nezaželene medsebojne vplive pomnilniških celic. Pojavi, kot so crosstalk, kjer zaradi lastnosti regulatorjev, signali enega bita vplivajo na drugega, lahko povzročijo nezaželene preklope vrednosti in tako povzročijo napake v zapisu. Takšnih vplivov v naši implementaciji nismo upoštevali, vendar bi bilo smiselno za bolj realističen prikaz model implementirati z uporabo stohastičnih pristopov, ki eksplicitno upoštevajo naključnost biokemijskih reakcij in omogočajo, da se napake in odstopanja pojavijo kot posledica šuma, medsebojnih vplivov in drugih dejavnikov, ne zgolj kot ročno vnesene motnje.

## 1.7 Zaključek

V seminarski nalogi smo predstavili našo biološko implementacijo 4-bitnega PIPO registra s paritetnim zaznavanjem napak z uporabo navadnih diferencialnih enačb. Na simulaciji smo prikazali dinamiko takšnega sistema in s tremi determinističnimi scenariji napak demonstrirali, kako naša arhitektura zaznava nepravilnosti v podatkovnih in paritetnih bitih. Tako smo pokazali, kako je mogoče s sintetično biologijo sestaviti gradnik pomnjenja, ki je analogen digitalni zasnovi. Predstavili smo tudi nekaj možnih izboljšav, ki bi jih upoštevali v nadaljnjih raziskavah.

## **1.8 Doprinosi avtorjev**

Filip Gros: Implementacija in opis paritetnega zaznavanja napak, priprava in opis simulacije zaznavanja napak, vizualizacija

Marko Horvat: Implementacija in opis PIPO registra, priprava simulacije delovanja PIPO registra

Andraž Klobučar: Raziskovanje simulacije napak, urejanje kode, pisanje in urejanje dokumenta



## Literatura

1. J. F. Wakerly, *Digital Design: Principles and Practices*. Pearson Prentice Hall, 4 ed., 2005.
2. U. Alon, *An Introduction to Systems Biology: Design Principles of Biological Circuits*. Chapman Hall/CRC, 2006.
3. M. Moškon, Ž. Pušnik, L. Stanovnik, N. Zimic, and M. Mraz, “A computational design of a programmable biological processor,” *Biosystems*, vol. 221, p. 104778, 2022.