Poglavje 5

Trovrednostna logika

Klasična *dvovrednostna*, *dvostanjska*, *dvojiška* ali *binarna logika* temelji na dveh različnih možnih vrednostih *logične spremenljivke* ali dveh različnih možnih *stanjih nosilca* (npr. električnega napetostnega nivoja) logične spremenljivke. Zalogo vrednosti dvovrednostne logične spremenljivke običajno zapišemo z množico $A = \{0, 1\}$. V domeno dvovrednostne logike sodijo tudi *dvovrednostne logične funkcije* ali *dvovrednostne logične operacije*. Dobro so nam poznane enostavne dvovrednostne logične funkcije kot so AND, OR, NEG, NOR, NAND, implikacija, ekvivalenca itd.

V domeni dvovhodne dvovrednostne logične funkcije, v katero vstopata logični spremenljivki x_1 in x_2 za katerima se skrivata možni vstopajoči logični vrednosti 0 ali 1 ($x_1, x_2 \in \{0, 1\}$), pridemo do pravilnostne tabele štirih različnih vhodnih vektorjev, kot je prikazano v tabeli 5.1. Pri tem nam različne možne vrednosti izhodov (y_1, ... $y_4 \in \{0, 1\}$) omogočajo postavitev šestnajstih različnih dvovhodnih dvovrednostnih logičnih funkcij ($2^4 = 16$).

V domeni dvovrednostne logike nas zanimajo minimalni nabori enostavnih logičnih funkcij, s katerimi lahko realiziramo poljubno kompleksnejšo večvhodno logično funkcijo. Minimalno število gradnikov je pomembno z vidika realizacije, saj imamo v tem primeru opravka z manjšim številom različnih logičnih vrat, ki realizirajo opazovano kompleksno logično funkcijo. Posamezen nabor logičnih funkcij iz predhodno definirane minimalne množice imenujemo za *poln funkcijski*

x_1	x_2	y
0	0	y_1
0	1	y_2
1	0	y_3
1	1	y_4

Tabela 5.1: Pravilnostna tabela poljubne dvovhodne dvovrednostne logične funkcije.

x_1	x_2	y
0	0	y_1
0	$\frac{1}{2}$	y_2
0	1	y_3
$\frac{1}{2}$	0	y_4
$\frac{1}{2}$	$\frac{1}{2}$	y_5
$\frac{1}{2}$	1	y_6
1	0	y_7
1	$\frac{1}{2}$	y_8
1	1	y_9

Tabela 5.2: Pravilnostna tabela poljubne dvovhodne trovrednostne logične funkcije z enim izhodom.

sistem. Tipični tovrstni nabori v dvovrednostni logiki so {NAND}, {NOR}, {AND, OR, NEG} itd.

Poleg klasične dvovrednostne logike, na kateri danes temeljijo *računalniško pomnjenje*, *računalniško procesiranje* in *računalniški prenos podatkov*, obstaja tudi množica drugačnih logik, ki so se začele razvijati že v začetku prejšnjega stoletja. Tako poznamo večje število različnih *trovrednostnih*, *trostanjskih* ali *ternarnih logik* (angl. *ternary logic*) in *večvrednostnih logik* (angl. *multiple valued logic*). V pričujočem poglavju si bomo ogledali nekaj primerov trovrednostnih logik in skušali razjasniti smisel njihovega obstoja.

V primerjavi z dvovrednostno logiko v *trovrednostni logiki* (angl. *ternary logic*) zalogo vrednosti logične spremenljivke in nosilca stanj logične spremenljivke razširimo na tri vrednosti ali na tri razpoložljiva stanja. Predpostavimo, da je zaloga teh vrednosti podana z množico $A = \left\{0, \frac{1}{2}, 1\right\}$. Po zgledu tabele 5.1 za dvovhodno dvovrednostno logično funkcijo, lahko napravimo tabelo tudi za dvovhodno trovrednostno funkcijo (glej tabelo 5.2). Pri tem nam različne možne vrednosti izhodov (y_1, ... $y_9 \in \left\{0, \frac{1}{2}, 1\right\}$) omogočajo postavitev 19.683 različnih trovrednostnih dvovhodnih logičnih funkcij ($3^9 = 19.683$). Ugotovimo lahko, da se nam tako pri istem številu vhodov in izhodov močno poveča število različnih logičnih funkcij v primerjavi z dvovrednostnimi dvovhodnimi logičnimi funkcijami. Pravilnostne tabele dvohvodnih trovrednostnih logičnih funkcij z enim izhodom zaradi preglednosti mnogokrat navajamo v matrični obliki, kot je to predstavljeno v tabeli 5.3.

V splošnem ob n vhodnih spremenljivkah lahko relaiziramo $(3^3)^n$ različnih trovrednostnih funkcij.

5.1 Značilnosti trovrednostnega zapisa podatkov

Trovrednostni zapis podatkov ima z vidika pomnjenja podatkov pomembne prednosti, opisane v delu [8]. Predpostavimo, da smo soočeni s problemom pomnjenja naravnih števil. Zapis poljubnega naravnega števila m ($m \in \mathbb{N}$)

x_1/x_2	0	$\frac{1}{2}$	1
0	$y_{0,0}$	$y_{0,\frac{1}{2}}$	$y_{0,1}$
$\frac{1}{2}$	$y_{\frac{1}{2},0}$	$y_{\frac{1}{2},\frac{1}{2}}$	$y_{\frac{1}{2},1}$
1	$y_{1,0}$	$y_{1,\frac{1}{2}}$	$y_{1,1}$

Tabela 5.3: Alternativen prikaz pravilnostne tabele poljubne dvovhodne trovrednostne logične funkcije z enim izhodom.

lahko formalno zapišemo s polinomskim izrazom

$$m = \sum_{i=0}^{\infty} d_i r^i, \tag{5.1}$$

pri čemer r predstavlja *bazo zapisa* ($r \in \mathbb{N}$), d_i pa posamezen *znak* ali *digit* zapisa ($\forall i : d_i \in \{0, ..., r-1\}$). S spremembo spodnje meje vsote iz 0 na $-\infty$ v izrazu (5.1), bi lahko problem zapisa naravnega števila m razširili na problem zapisa poljubnega pozitivnega realnega števila.

Najpogostejši zapis naravnih števil v realnem svetu je desetiški ali decimalni [?], pri čemer znaki ali digiti prihajajo iz množice $D_{10} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, baza pa je 10 ($r = 10$). Slednje po viru [8] izhaja iz načina štetja, ki pri človeku temelji na številu prstov na obeh rokah. V računalništvu vse od šestdesetih let prejšnjega stoletja dominira dvojiški ali dvovrednostni zapis, pri čemer znaki imenovani *biti* (angl. *binary digit*) prihajajo iz množice $D_2 = \{0, 1\}$. Število 19 tako v desetiškem sistemu ($r = 10$) zapišemo z izrazom

$$19_{10} = 1 * 10^1 + 9 * 10^0, \tag{5.2}$$

v dvojiškem sistemu ($r = 2$) pa z izrazom

$$19_{10} = 10011_2 = 1 * 2^4 + 0 * 2^3 + 0 * 2^2 + 1 * 2^1 + 1 * 2^0. \tag{5.3}$$

Zapis števila 19 v trovrednostnem - trojiškem sistemu ($r = 3$) se ob upoštevanju množice znakov ali digitov

$$D_3 = \{0, 1, 2\}, \tag{5.4}$$

ki jih imenujemo za *trite*, zapiše z izrazom

$$19_{10} = 201_3 = 2 * 3^2 + 0 * 3^1 + 1 * 3^0. \tag{5.5}$$

Zapis poljubnega naravnega števila na osnovi digitov lahko ovrednotimo na osnovi različnih kriterijev. Dve najpogostejši cenilki sta

- *število različnih uporabljenih digitov*, kar sovpada z velikostjo baze r in
- *dolžina zapisa* w opazovanega števila m.

Velja, da večje število različnih digitov vodi do krajših dolžin zapisov in obratno manjše število različnih digitov do daljših dolžin zapisov. Iz povedanega sklepamo, da sta si cenilki kontradiktorni.

Glede na izbrani cenilki se nam ponujajo različne rešitve načina zapisa. Po kriteriju iz prve alineje so najugodnejši zapisi z minimalno možno bazo digitov ($r = 1$). V tem primeru imamo v bazi samo en znak in dolžina zapisa v digitih sovpada z velikostjo zapisanega števila. Omenjeni pristop v praksi lahko rezultira v zelo dolge zapise. Omenimo, da opisani *unarni zapis* ne sovpada z formalno definicijo zapisa števila iz izraza (5.1).

Po kriteriju iz druge alineje so najugodnejši zapisi z zelo velikimi bazami. Tako bi lahko pri bazi $r = 1.000.000$ poljubno število iz intervala od 0 do 999.999 zapisali z enim samim digitom, pri čemer smo v tem primeru soočeni z eksplozijo števila digitov in vprašljivo je, kako izredno veliko število različnih hipotetičnih stanj, ki jih potrebujemo za interpretacijo digitov, implementirati na nosilnem signalu prenosa, procesiranja ali mediju pomnjenja logičnih spremenljivk.

Osnovna hipoteza avtorja članka [8] temelji na predpostavki, da je za zapis naravnih števil baza $r = 2$ premajhna, baza $r = 10$ pa prevelika. Avtor za idealen zapis smatra tisti zapis, kjer je razmerje med dolžino zapisa (w) in velikostjo potencialne zaloge digitov (r) *ustrezno*. Slednjo ustreznost po avtorju dosežemo, ko je dosežen minimum funkcije iz izraza

$$r * w, \tag{5.6}$$

ob zadrževanju konstantne vrednosti funkcije iz izraza

$$r^w. \tag{5.7}$$

Problem je analitično najlažje rešljiv, če predpostavimo, da sta števili r in w realni ($r, w \in \mathbb{R}$). Na tem mestu definirajmo pojem *ekonomičnosti baze* (angl. *radix economy*), povzet po viru [45] z izrazom

$$E(r, w) = r * \lfloor \log_r(w) + 1 \rfloor. \tag{5.8}$$

V primeru, da sta r in w naravni števili ($r, w \in \mathbb{N},$), izraz $E(r, w)$ predstavlja bazo r pomnoženo s številom potrebnih digitov za zapis števila w. Ekonomičnost baze nam tako oceni ceno hrambe ali procesiranja števila w v bazi r, pri čemer je cena posameznega digita proporcialna bazi r.

Predpostavimo, da funkcijo $E(r, w)$ delno poenostavimo v obliko

$$E(r, w) \approx r * \log_r(w) = r * \frac{\ln(w)}{\ln(r)} = \ln(w) * \frac{r}{\ln(r)}. \tag{5.9}$$

$\ln(w)$ iz izraza lahko smatramo za konstanto, kvocient spremenljivke z njeno naravno logaritemsko funkcijo pa doseže minimum pri vrednosti $r = e$. Tako število e proglasimo za bazo z največjo ekonomičnostjo. Ker število e ni ustrezno za bazo, funkcija pa je pred minimumom padajoča ($1 < r < e$), po njem pa rastoča ($r > e$), lahko kandidata za optimalno bazo iščemo v sosednjih celih

x	$\overline{x}$
0	2
1	1
2	0

Tabela 5.4: Pravilnostna tabela Łukasiewiczeve trovrednostne negacije (NEG).

številih števila e. Kandidata sta števili 2 in 3, pri čemer sta po vrsti oceni njune ekonomičnosti

$$\frac{2}{\ln(2)} \approx 2,89, \frac{3}{\ln(3)} \approx 2,73. \tag{5.10}$$

Glede na rezultat izračuna lahko smatramo, da je celoštevilčna baza $r = 3$ najugodnejša celoštevilčna baza glede na ocenjevalni kriterij njene ekonomičnosti.

5.2 Vrste trovrednostnih logik

V preteklosti je prišlo do porajanja več različnih ternarnih ali trovrednostnih logičnih sistemov. Med seboj se razlikujejo po kodnih zapisih in po definicijah osnovnih logičnih operacij. V domeni kodnih zapisov so najpogostejši kodni zapisi $\{0, \frac{1}{2}, 1\}$, $\{0, 1, 2\}$ in $\{-1, 0, 1\}$. Pri tem prvi in zadnji element vsake od naštetih množic po vrsti enačimo z logičnima vrednostima `false` in `true` ali logičnima vrednostima 0 in 1, kot smo jih bili vajeni v dvovrednostnem svetu. Osrednji element vsake od neštetih množic predstavlja vrednost spremenljivke ali stanje nosilca, ki je nedoločeno (angl. *unknown, indeterminate*) ali vmesno.

V nadaljevanju si bomo ogledali *trovrednostno logiko po J. Łukasiewiczu* in *uravnoteženi trovrednostni način zapisa*.

5.2.1 Trovrednostna logika po J. Łukasiewiczu

Trovrednostna logika se razvije po zaslugi Poljaka J. Łukasiewicza v dvajsetih letih prejšnjega stoletja. Osnovna ideja, ki jo avtor uvede v svojo logiko, je uvedba tretje logične vrednosti, ki odraža stanje *nedoločenosti* vrednosti logične spremenljivke (angl. *indeterminate*), poleg nam že znanih logičnih vrednosti *true* in *false* [46]. Łukasiewicz poveže logične vrednosti s kodnim zapisom $D_3 = \{0, 1, 2\}$ na osnovi izraza

$$false \equiv 0, indeterminate \equiv 1, true \equiv 2. \tag{5.11}$$

Na osnovi interpretacije logičnih vrednosti s kodnim zapisom vzpostavi pravilnostne tabele za osnovne logične operatorje, kot so NEG, AND, OR in implikacija, ki so predstavljene v tabelah 5.4, 5.5, 5.6 in 5.7. Najzanimivejša je zadnja tabela, v kateri se vsi izhodi oblikujejo po znani preslikavi $\overline{x}_1 \vee x_2$, razen pri vhodni kombinaciji ($x_1 = 1, x_2 = 1$), kjer izhod ni 1, temveč 2. Operacija implikacije je ena od operacij, po kateri se različne trovrednostne logike med seboj razlikujejo.

x_1	x_2	y
0	0	0
0	1	0
0	2	0
1	0	0
1	1	1
1	2	1
2	0	0
2	1	1
2	2	2

Tabela 5.5: Pravilnostna tabela Łukasiewiczeve trovrednostne konjunkcije (AND).

x_1	x_2	y
0	0	0
0	1	1
0	2	2
1	0	1
1	1	1
1	2	2
2	0	2
2	1	2
2	2	2

Tabela 5.6: Pravilnostna tabela Łukasiewiczeve trovrednostne disjunkcije (OR).

x_1	x_2	y
0	0	2
0	1	2
0	2	2
1	0	1
1	1	2
1	2	2
2	0	0
2	1	1
2	2	2

Tabela 5.7: Pravilnostna tabela Łukasiewiczeve trovrednostne implikacije ($\Rightarrow$).

x	$T(x)$
0	1
1	1
2	1

Tabela 5.8: Pravilnostna tabela Łukasiewiczeve trovrednostne T funkcije.

Izkaže se, da trovrednostne funkcije definirane v tabelah 5.4, 5.5, 5.6 in 5.7 ne predstavljajo polnega funkcijskega nabora [46]. J. Slupecki l. 1936 v ta namen uvede še funkcijo T, s čimer dobimo funkcionalno polno algebro (poln funkcijski nabor), definirano s četvorčkom

$$\langle E = \{0, 1, 2\}, \Rightarrow, \mathrm{NEG}(), T() \rangle, \tag{5.12}$$

pri čemer funkcija $\Rightarrow$ predstavlja implikacijo definirano s pravilnostno tabelo 5.7, funkcija NEG negacijo predstavljeno s pravilnostno tabelo 5.4, funkcija T pa je definirana s pravilnostno tabelo 5.8.

5.2.2 Uravnoteženi trovrednostni način zapisa

Uravnoteženi trovrednostni način zapisa (angl. *balanced ternary sistem*) temelji na izboru kodne množice treh znakov, ki omogoča še ekonomičnejši zapis podatkov. Primer takšnega kodnega nabora je množica $A = \{-1, 0, 1\}$. Z vpeljavo negativno predznačenega digita pride do poenostavitve zapisa predznaka zapisanega števila in do posplošitve nekaterih računskih operacij (npr. odštevanje se realizira kot seštevanje, pri čemer se le zamenjajo predznačeni biti). Posebnega znaka za predznačitev podatka tako ne potrebujemo več. Poglejmo si nekaj primerov zapisov števil na osnovi uporabe uravnoteženega trovrednostnega zapisa, povzetih po [47], pri čemer v trovrednostnem delu zapisa zaradi preglednosti namesto znaka -1 uporabimo znak T:

$$10_3 = 1 * 3^1 + 0 * 3^0 = 3_{10}, \tag{5.13}$$

$$10T_3 = 1 * 3^2 + 0 * 3^1 - 1 * 3^0 = 8_{10}, \tag{5.14}$$

$$T00_3 = -1 * 3^2 + 0 * 3^1 + 0 * 3^0 = -9_{10}. \tag{5.15}$$

Ob predpostavki, da nam indeks potence i iz izraza (5.1) lahko steče tudi v smeri negativnih števil, pri čemer digite negativnih potenc polinoma navajamo desno od decimalne pike, ki ločuje levi del pozitivnih potenc polinoma, bi tako realno število $-\frac{2}{3}$ zapisali z izrazom

$$T.1_3 = -1 * 3^0 + 1 * 3^{-1} = -1 + \frac{1}{3} = -\frac{2}{3}_{10}. \tag{5.16}$$

Poljuben zapis z bazo r lahko pretvorimo v uravnoteženi trovrednostni zapis na osnovi izraza

$$(a_n a_{n-1} ... a_1 a_0 . c_1 c_2 c_3 ...)_r = \sum_{k=0}^{n} a_k r^k + \sum_{k=1}^{\infty} c_k r^{-k}, \tag{5.17}$$

Oznaka	Kodni nabor	Poimenovanje nabora
A	$\{0, 1, 2\}$	*Unbalanced ternary*
B	$\{0, \frac{1}{2}, 1\}$	*Fractional unbalanced ternary*
C	$\{-1, 0, 1\}$	*Balanced ternary*
D	$\{F, ?, T\}$	*Unknown state logic*
E	$\{T, F, T\}$	*Ternary coded binary*

Tabela 5.9: Pravilnostna tabela poljubne dvovhodne dvovrednostne logične funkcije.

kjer so v levem delu izraza originalni digiti med a_0 in c_1 ločeni s piko, r baza zapisa v trovrednostnem zapisu, a_k digiti levo od pike v trovrednostnem zapisu, c_k pa digiti desno od pike v trovrednostnem zapisu. Preizkusimo podani izraz na delni prevedbi negativnega realnega števila $-25,4_{10}$ v uravnoteženi trovrednostni zapis.

$$-25,4_{10} = -(1T * 101^1 + 1TT * 101^0 + 11 * 101^{-1}), \tag{5.18}$$

$$= -(1T * 101 + 1TT + \frac{11}{101}), \tag{5.19}$$

V izrazu (5.18) uporabimo polinomski nastavek, v izrazu (5.19) pa desno stran poenostavimo glede na vrednosti potenc. Če bi hoteli izraz poenostaviti do končnega zapisa, bi morali osvojiti tudi metodi množenja in deljenja, kar pa presega obseg pričujočega dela. Bralcu predlagamo, da si omenjene metode poišče v ustrezni literaturi. Enostaven opis omenjenih metod najdemo tudi na Wikipedii [47].

5.2.3 Preostali trovrednostni logični sistemi

V strokovni literaturi najdemo množico različnih trovrednostnih logik. V delu [46] so tako opisane Postova logika, Bochvarjevea logika, Kleenova logika, Allenova in Givoneova logika, Pradhanova logika itd.

5.3 Kodni nabori

V predhodnih razdelkih smo v domeni trovrednostnih sistemov uporabljali različne kodne nabore digitov. V tabeli 5.9 so navedeni najpogostejši ternarni kodni nabori, povzeti po viru [48]. Poimenovanja kodnih naborov so v tabeli namenoma neprevedena, zaradi redke rabe v slovenskem jeziku. Kodna nabora A in B sta najpogosteje v rabi, kodni nabor C je zanimiv zaradi možnosti opustitve predznačevanja numeričnih podatkov (zaradi nepreglednosti zapisa se namesto znaka -1 uporabljajo v literaturi tudi znaki T, i ali $\overline{1}$), kodni nabor E pa služi kot izhodišče za preslikovanje iz ternarnega v binarni vrednostni prostor.

x	y
T	y_1
0	y_2
1	y_3

Tabela 5.10: Splošna pravilnostna tabela enovhodne trovrednostne funkcije.

x	y
T	T
0	T
1	T

x	y
T	0
0	0
1	0

x	y
T	1
0	1
1	1

Tabela 5.11: Pravilnostne tabele za enovhodne trovrednostne funkcije konstante.

5.4 Enovhodne trovrednostne funkcije

V tabeli 5.10 je podana splošna oblika enovhodne trovrednostne funkcije na osnovi *uravnoteženega kodnega nabora* (angl. *balanced ternary*) z notacijo T. Iz tabele je razvidno, da je možno realizirati $3^3 = 27$ različnih enovhodnih trovrednostnih logičnih funkcij. Razdelimo jih v tri skupine. Le te so sledeče [48]:

- *funkcije konstante*: omenjene funkcije na izhodu formirajo eno od vrednosti trovrednostnega kodnega nabora; obstajajo tri takšne različne funkcije, navedene v tabeli 5.11; vektorji vrednosti izhodnih funkcij so po vrsti y =TTT, y =000 in y =111;
- *„one to one“ funkcije*: za slednje funkcije je značilno, da se vsaka možna vhodna vrednost preslika v sebi lastno izhodno vrednost; glede na povedano so tovrstne funkcije *logično reverzibilne*; tovrstnih različnih funkcij je šest in so sledeče:
  - y =T01 - buffer,
  - y =T10 - swap 0/1,
  - y =0T1 - swap T/0,
  - y =01T - rotate up (shift up),
  - y =1T0 - rotate down (shift down),
  - y =10T - swap T1 (invert).
- *„many to one“ funkcije*: tovrstnih različnih funkcij je 18, njihovi pomeni pa presegajo obseg pričujočega dela; v splošnem gre za funkcije, kjer v izhodnem vektorju nastopata dva kodna znaka vhodnega nabora, pri čemer se eden od njiju na izhodni strani ponovi;

x_1/x_2	0	1	2
0	0	0	0
1	0	1	0
2	0	0	2

Tabela 5.12: Pravilnostna tabela dvovhodne trovrednostne logične funkcije enakosti [49].

x_1/x_2	0	1	2
0	0	0	0
1	0	1	1
2	0	1	2

Tabela 5.13: Pravilnostna tabela dvovhodne trovrednostne logične funkcije *MIN* [49].

Več o enovhodnih trovrednostnih funkcijah in možnostih njihove uporabe pri gradnji logičnih struktur si bralec lahko prebere v delu [48].

5.5 Dvovhodne trovrednostne funkcije

Že na začetku pričujočega poglavja smo izračunali, da v domeni dvohvodnih trovrednostnih logičnih funkcij obstaja kar 19.863 različnih funkcij. V virih [49], [50] najdemo dober pregled zanimivejših dvovhodnih trovrednostnih funkcij predstavljenih z uporabo kodnega nabora $A = \{0, 1, 2\}$. Avtor izpostavi pomene sledečih funkcij:

- *funkcija enakosti*: pri paru vhodnih vrednosti, ki sta enaki, se izhodna vrednost ohranja, pri ostalih parih vhodnih vrednostih pa je izhodna vrednost 0; pravilnostna tabela takšne funkcije je predstavljena v tabeli 5.12;

- *MIN in MAX funkcija*: funkcija na izhodu vrne minimalno ali maskimalno vrednost para vhodnih vrednosti; funkciji sta predstavljeni v tabeli 5.13 in 5.14;

- *TNOR funkcija*: ternarna NOR funkcija predstavlja poln nabor funkcij[1]; pravilnostna tabela takšne funkcije je predstavljena v tabeli 5.15;

- *TXOR funkcija*: pravilnostna tabela takšne funkcije je predstavljena v tabeli 5.16;

[1] S TNOR funkcijami lahko realiziramo poljubno trovrednostno logično funkcijo ne glede na njeno število vhodov in izhodov.

x_1/x_2	0	1	2
0	0	1	2
1	1	1	2
2	2	2	2

Tabela 5.14: Pravilnostna tabela dvovhodne trovrednostne logične funkcije *MAX* [49].

x_1/x_2	0	1	2
0	2	1	0
1	1	1	0
2	2	1	0

Tabela 5.15: Pravilnostna tabela dvovhodne trovrednostne logične funkcije *TNOR* [50].

x_1/x_2	0	1	2
0	0	1	2
1	1	1	1
2	2	1	0

Tabela 5.16: Pravilnostna tabela dvovhodne trovrednostne logične funkcije *TXOR* [50].

5.6 Trovrednostno procesiranje

Trovrednostno procesiranje (angl. *ternary computing*) se je v preteklosti uporabljalo tudi v realnih računalniških sistemih. V prejšnjem stoletju je bilo v petdesetih letih v nekdanji Sovjetski zvezi zgrajenih kar nekaj trovrednostnih računalnikov (angl. *ternary computer* ali *trinary computer*), ki so izvajali operacije v trovrednostnem zapisu in na osnovi trojiške logike. Najbolja znana družina tovrstnih sistemov je bila serija računalnikov Setun (1.generacija l.1958, 2.generacija l.1970), razvitih na Moskovski državni univerzi. Temeljili so na uravnoteženi množici digitov zapisovanja $\{-1, 0, 1\}$. Osnovna enota zapisa sorodna današnjemu bitu je bil *trit*, šest tritov pa je predstavljalo *tryte*, ki je soroden današnjemu byte-u. Obstajajo splošne ocene, da je zapis v tryte-u ekvivalenten podatkovni vsebini zapisani v 9,5 bitih. Prvi emulator trovrednostnega računalnika v ZDA je bil implementiranem l.1973 na Borroughs B1700 platformi.

5.7 Povzetek poglavja

Področje trojiškega zapisa in ustrezne logike še danes buri duhove znanstvene srenje s področja računalništva. Eden od pionirjev računalništva Donald Knuth trdi [51], da se bo ternarno procesiranje zaradi svojih dobrih plati nedvomno vrnilo v računalniške strukture. Kot eden od možnih nosilcev za trovrednostni kodni nabor se v zadnjem času omenjajo optični mediji, pri katerih bi se odsotnost svetlobe interpretirala kot vrednost 0, dve ortogonalno polarizirani svetlobi pa bi predstavljali nosilca logičnih vrednosti 1 in -1.

Z uporabo trovrednostne logike pridemo do evidentnega skrajšanja zapisa podatkov, s čimer dobimo krajše enote obdelave, kar omogoča hitrejšo obdelavo podatkov, manj dostopov do pomnilnika, manjšo kompleksnost priključkov itd.

Literatura

[1] "The scale of things." `http://science.energy.gov/bes/community-resources/scale-of-things-chart/`, September 2016.

[2] M. Hak, *The MEMS Handbook*. CRC Press, 2002.

[3] M. Mack, "The multiple lives of Moore's law," *IEEE Spectrum*, vol. 4, pp. 29–35, 2015.

[4] D. Kodek, *Arhitektura in organizacija računalniških sistemov*. Bi-Tim, Slovenija, 2008.

[5] *From editors of Scientific American: Understanding nanotechnology*. Warner Books, ZDA, 2002.

[6] "2001: A Space Odyssey," 1968.

[7] A. Adamatzky, B. Costello, and T. Asai, *Reaction diffusion computers*. Elsevier, 2005.

[8] B. Hayes, "Third base," *American Scientist*, vol. 89, no. 6, 2001.

[9] W. Aspray, *John Von Neumann and The Origins Of Modern Computing*. The MIT Press, England, 1990.

[10] "There's Plenty of Room at the Bottom." `https://en.wikipedia.org/wiki/There%27s_Plenty_of_Room_at_the_Bottom/`, September 2016.

[11] E. Regis, *Nano – the emerging science of nanotechnology*. BackBay Books, 1995.

[12] C. Lent, P. Tougaw, W. Porod, and G. Bernstein, "Quantum cellular automata," *Nanotechnology*, vol. 4, 1993.

[13] C. Lent and P. Tougaw, "Lines of interacting quantum-dot cells: a binary wire," *Journal of Applied Physics*, vol. 74, 1993.

[14] I. L. Bajec and M. Mraz, "Večstanjsko procesiranje v strukturah kvantnih celičnih avtomatov," *Elektrotehniški vestnik*, vol. 73, no. 2-3, 2006.

[15] P. Pečar, "Uporaba adiabatnega pristopa pri realizaciji trojiškega procesiranja na osnovi kvantnih celičnih avtomatov," Master's thesis, Faculty of computer and Information science, University of Ljubljana, 2007.

[16] G. Snider, A. Orlov, I. Amlani, and G. Bernstein, "Quantum-dot cellular automata: line and majority logic gate," *Japanese Journal of Applied Physics*, vol. 38, 1999.

[17] K. Walus, T. Dysart, G. Jullien, and R. Budiman, "Qcadesigner: a rapid design and simulation tool for quantum dots cellular automata," *IEEE Transactions on Nanotechnology*, vol. 3, 2004.

[18] T. Cole and J. Lusth, "Quantum-dot cellular automata," *Progress in Quantum Electronics*, vol. 25, 2001.

[19] Z. Kohavi, *Switching and finite automata theory.* McGraw-Hill Inc., USA, 1978.

[20] M. Niemier and P. Kogge, *Nano, Quantum and Molecular Computing - Implications to High Level Design and Validation*, ch. Origins and Motivations for Design Rules in QCA. Kluwer Academic Publishers, Boston, 2004.

[21] "Reverzibilnost procesiranja." `http://strangepaths.com/reversible-computation/2008/01/20/en/`, Oktober 2016.

[22] B. Hayes, "Reverse engineering," *American Scientist*, vol. 94, 2006.

[23] P. R. Yelekar and S. S. C. Hanson, "Introduction to reversible logic gates and its application," *International journal of computer applications*, 2011.

[24] S. M. R. Taha, *Reversible logic synthesis methodologies with application to quantum computing.* Springer, Switzerland, 2016.

[25] P. J. Denning and T. G. Lewis, "Computers that can run backwards," *American Scientist*, vol. 105, no. 5, 2017.

[26] "Reversible." `http://www.eng.fsu.edu/\symbol{126}mpf/pubs.htm`, Maj 2015.

[27] "Reversible." `http://www.cise.ufl.edu/research/revcomp/`, Maj 2015.

[28] K. Perumalla, *Introduction to Reversible Computing.* Chapman & Hall/CRC Press, 2014.

[29] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information.* Cambridge University Press, UK, 2009.

[30] C. Calude and G. Păun, *Computing with cells and atoms, An introduction to quantum, DNA and membrane computing.* Taylor and Francis, London, 2001.

[31] "Single particle interference." http://player.slideplayer.com/26/8574538/#, December 2017.

[32] "Quantum computing." http://en.wikipedia.org/wiki/Quantum_computing, Maj 2015.

[33] M. Nagy and S. G. Akl, "Quantum computation and quantum information," tech. rep., 2005. Technical report, 2005-496.

[34] "Elementary quantum notation." http://www-users.cs.york.ac.uk/schmuel/comp/node6.html, Maj 2015.

[35] J. Virant, *Načrtovanje nanoračunalniških struktur.* Didakta, Slovenija, 2007.

[36] M. Hirvensalo, *Quantum Computing.* Springer Verlag, 2001.

[37] H. Stöcker, *Matematični priročnik z osnovami računalništva.* Tehniška založba, Slovenija, 2006.

[38] O. Bronštejn, K. Semendjajev, G. Musiol, and H. Mühlig, *Matematični priročnik.* Tehniška založba, Slovenija, 1997.

[39] A. Pittenger, *An Introduction to Quantum Computing Algorithms.* Birkhäuser, ZDA, 2000.

[40] D. Deutsch and R. Jozsa, "Rapid solutions of problems by quantum computation," *Proceedings of the Royal Society of London*, vol. 439, no. 1907, pp. 553–558, 1992.

[41] "Quantum programming languages." http://www.dcs.gla.ac.uk/~simon/quantum/, November 2015.

[42] "GUI environment for Quantum Computer Simulator." http://qcad.osdn.jp/, November 2015.

[43] "Quantiki homepage." http://www.quantiki.org/wiki/list-qc-simulators, November 2015.

[44] E. Dahl, "Programming with D-Wave: Map coloring problem," tech. rep., D-Wave Systems, 2013.

[45] "Radix economy." https://en.wikipedia.org/wiki/Radix_economy, October 2017.

[46] D. Miller and M. Thornton, *Multiple Valued Logic, Concepts and Representations.* Morgan and Claypool Publishers, USA, 2008.

[47] "Balanced ternary - Wikipedia." https://en.wikipedia.org/wiki/Balanced_ternary/, September 2016.

[48] J. Connelly, C. Patel, and A. Chavez, "Ternary Computing Testbed 3-Trit Computer Architecture," tech. rep., California Polytechnic State University of San Luis Obispo advised by Professor Phillip Nico, 2008.

[49] E. Katiyar, "A Treatise on the Fundamentals of Ternary Arithmetic and Logic," *International Journal of Computer Science Engineering*, vol. 6, no. 8, 2017.

[50] G. Trishala and K. Ragini, "Design and Implementation of Ternary Logic Circuits for VLSI Applications," *International Journal of Innovative Technology and Exploring Engineering*, vol. 9, 2020.

[51] "Ternary computer - Wikipedia." `https://en.wikipedia.org/wiki/Ternary_computer/`, September 2016.