

RAZVOJ DIGITALNIH SISTEMOV

Izpit

23. 01. 2025

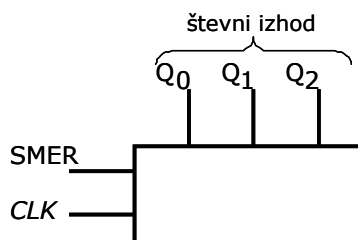
1. Izrazite podano logično funkcijo samo s Shefferjevimi operatorji. Morebitne negacije realizirajte s Shefferjevim operatorjem.

$$f(a,b,c) = \overline{((a+b) \downarrow \bar{c})} \equiv (a \downarrow c)$$

2. Realizirajte funkcijo f v obliki PDNO z redundancami s čim manj izbiralniki 4/1.

$$f(x_1, x_2, x_3, x_4) = V(1, 2, 9, 13, 15) \text{ in } V_x(0, 5, 11, 12)$$

3. Prikažite sintezo sinhronnega dvosmernega 3-bitnega števca z uporabo T flip-flopov: Zapišite tabelo prehajanja stanj in določite enačbe flip-flopov. Števec ima vhod SMER, ki določa smer štetja: Če je SMER='0', števec šteje naraščajoče, sicer padajoče. Imena signalov so razvidna iz spodnje slike.



4. Načrtajte diagram stanj Mooreovega avtomata končnih stanj, ki krmili delovanje garažnih vrat: Garažna vrata imajo vhod VRATA ter vhod ZAŠČITA, ki postane '1' vedno, ko preko motorja steče dovolj velik tok. Z meritvijo toka na motorju obenem izdelamo funkcijo detekcije obeh končnih položajev, kot tudi zaščito proti oviram na poti vrat. Vezje ima 2-bitni izhod za enosmerni motor:

Koda operacije		Funkcija izhoda
OP ₁	OP ₀	
0	0	motor stoji
0	1	motor pomika vrata navzgor
1	0	motor pomika vrata navzdol

Če pritisnemo gumb VRATA, se vrata začno pomikati navzgor. Če na poti naletijo na oviro ali pridejo do zgornje končne lege, se motor ustavi. Če pritisnemo gumb VRATA ponovno, se začnejo gibati v smeri navzdol. Podobno je v obratni smeri: Če na poti naletijo na oviro ali pridejo do spodnje končne lege, se motor ustavi. Če pritisnemo gumb VRATA ponovno, se začnejo pomikati v smeri navzgor.

Rešitev 1. naloge

Realizacija s samimi Shefferjevimi (NAND, oziroma \uparrow) operatorji najprej zahteva pretvorbo funkcije v disjunktivno obliko (če se da, normalno).

$$f(a, b, c) = \overline{\left((a + b) \downarrow \bar{c} \right)} \equiv (a \downarrow c)$$

V ta namen moramo najprej operatorje (EQU, Pierce) v podani funkciji izpisati z disjunktijami in konjunkcijami. Začnemo na najvišjem nivoju, pri negaciji "čez vse" in EQU funkciji - oboje skupaj predstavlja XOR na najvišjem nivoju, ki jo izpišemo po definiciji XOR $\bar{x} \cdot y + x \cdot \bar{y} = x \oplus y$:

$$\begin{aligned} f(a, b, c) &= \left((a + b) \downarrow \bar{c} \right) \oplus (a \downarrow c) = \\ &= \left((a + b) \downarrow \bar{c} \right) \cdot \overline{(a \downarrow c)} + \overline{\left((a + b) \downarrow \bar{c} \right)} \cdot (a \downarrow c) \end{aligned}$$

Izpišemo še oba Pierceva (NOR) operatorja, pri čemer izničimo nastali dvojni negacijo drugega in tretjega člena ($x + y = x \downarrow y$):

$$f(a, b, c) = \overline{\left((a + b) + \bar{c} \right)} \cdot (a + c) + \left((a + b) + \bar{c} \right) \cdot \overline{(a + c)}$$

Uporabimo DeMorganov teorem nad negiranimi členi in nekaj ostalih osnovnih lastnosti Boole-ove logike ($x \cdot x = x$ in $x \cdot \bar{x} = 0$):

$$\begin{aligned} f(a, b, c) &= \overline{(a + b)} \cdot c \cdot (a + c) + \left((a + b) + \bar{c} \right) \cdot \bar{a} \cdot \bar{c} = \\ &= \bar{a} \cdot \bar{b} \cdot c \cdot (a + c) + (a + b + \bar{c}) \cdot \bar{a} \cdot \bar{c} = \\ &= \bar{a} \cdot \bar{b} \cdot c \cdot \bar{a} + \bar{a} \cdot \bar{b} \cdot c \cdot c + \bar{a} \cdot \bar{c} + a \cdot \bar{c} + b \cdot \bar{c} = \\ &= \bar{a} \cdot \bar{b} \cdot c + \bar{a} \cdot \bar{b} \cdot \bar{c} + \bar{a} \cdot \bar{c} + a \cdot \bar{c} + b \cdot \bar{c} = \bar{a} \cdot \bar{b} + \bar{a} \cdot \bar{c} + b \cdot \bar{c} \end{aligned}$$

Dobili smo disjunktivno normalno obliko (DNO):

$$f(a, b, c) = \bar{a} \cdot \bar{b} \cdot c + \bar{a} \cdot \bar{c} + b \cdot \bar{c}$$

Dobljeno DNO pretvorimo v Shefferjevo (NAND) obliko tako, da dvakrat negiramo posamezne člene:

$$\begin{aligned} f(a, b, c) &= \bar{a} \cdot \bar{b} + \bar{a} \cdot \bar{c} + \bar{a} \cdot \bar{b} + \bar{a} \cdot \bar{c} = \overline{\overline{\bar{a} \cdot \bar{b}}} + \overline{\overline{\bar{a} \cdot \bar{c}}} \\ &= \overline{\overline{\bar{a} \cdot \bar{b}}} + \overline{\overline{\bar{a} \cdot \bar{c}}} = (\bar{a} \uparrow \bar{b}) \uparrow (\bar{a} \uparrow \bar{c}) \end{aligned}$$

Preostale negacije spremenljivk izrazimo z uporabo lastnosti $\bar{x} = x \uparrow x$:

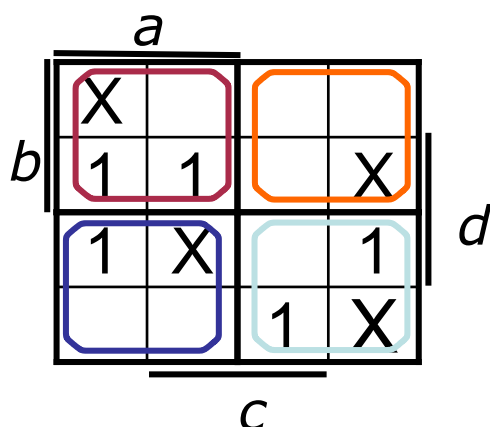
$$\begin{aligned} f(a, b, c) &= (\bar{a} \uparrow \bar{b}) \uparrow (\bar{a} \uparrow \bar{c}) \\ &= ((a \uparrow a) \uparrow (b \uparrow b)) \uparrow ((a \uparrow a) \uparrow (c \uparrow c)) \end{aligned}$$

Rešitev 2. naloge:

Funkcija f je podana v obliki PDNO z redundancami.

$$f = V(1,2,9,13,15) \text{ in } V_x(0,5,11,12)$$

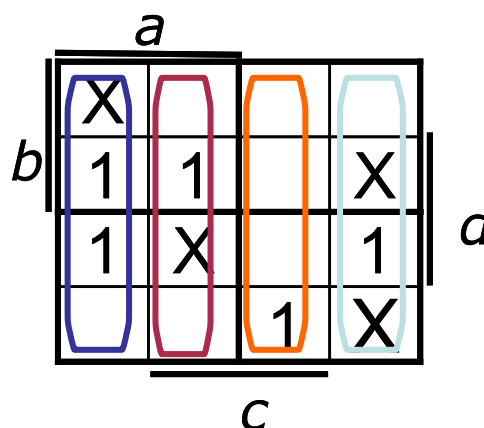
Dobljeno funkcijo vrišemo v Veitchev diagram. Ker iščemo najcenejšo realizacijo z izbiralnikom 4/1, bomo naredili razvoj po vseh kombinacijah naslovnih spremenljivk v Veitchevem diagramu. Če izberemo kot naslovni spremenljivki a b, potem dobimo:



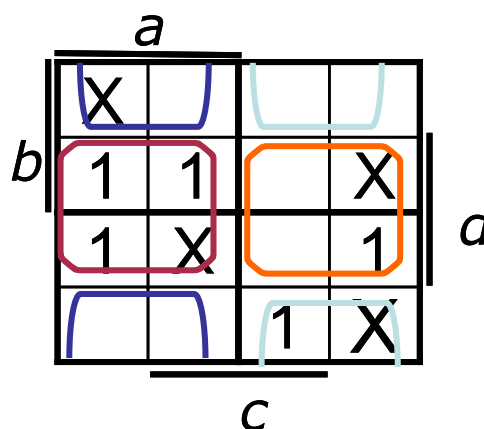
V zgornjem Veitchevem diagramu so označena vsa štiri polja štirih mintermov, če izberemo vhodni spremenljivki a in b. Zgornji levi kvadrat (rdeč) pomeni, da bo to polje izbrano ko bosta $ab="11"$, oranžni kvadrat ko bo $ab="01"$, temno modri ko bo $ab="10"$ in svetlo modri ko bo $ab="00"$. Vsakega od teh kvadratov poskušamo opisati s čimbolj enostavno funkcijo: Vrednost zgornjega levega kvadrata opišemo s spremenljivko d, če postavimo redundanco na '0'. Vrednost spodnjega desnega kvadrata je bolj komplicirana, saj moramo vsako '1' opisati posebej: Za zgornjo '1' v tem kvadratu velja $c \cdot d$, za spodnjo '1' pa $c \cdot d'$. Funkcija bo torej $c \cdot d + c \cdot d'$, kar je enačba funkcije XOR. Najbolj enostavna realizacija je zgornji desni kvadrat, ki je kar '0', če postavimo redundanco na '0'. Zato, da bi pregledali še ostale možnosti, moramo narisati še preostalih pet kombinacij dveh naslovnih vhodov.

Čas pisanja je 60 minut. Vsaka naloga je vredna 10 točk. Na list z rešitvami se podpišite in napišite še vpisno številko. Rezultati bodo objavljeni na domači strani predmeta.

Če izberemo kot naslovni spremenljivki a in c, dobimo levi Veitchev diagram, če a in d, pa desnega. Podobno kot v prejšnjem primeru poiščemo realizacije ustreznih kvadratov in iščemo najcenejšo realizacijo: Izogibamo se veliko različnim funkcijam in iščemo drugače kvadratov, ki vsebujejo same '1' ali same '0'. Pri razvoju po a in c imamo pri $ac="01"$ najneugodnejšo funkcijo, saj vsebuje eno samo '1'.

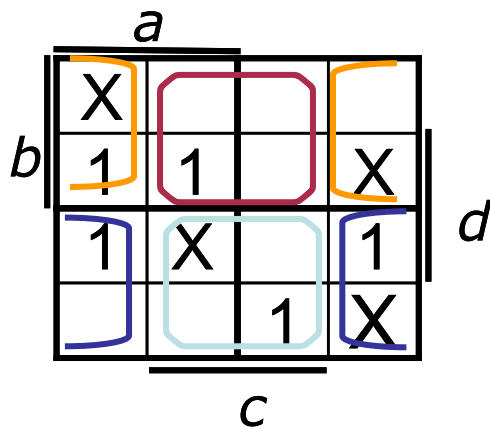


Pri razvoju po a in d nikjer ne nastopa ena sama '1' ali tri '1' ali diagonalna (XOR) dveh '1'.

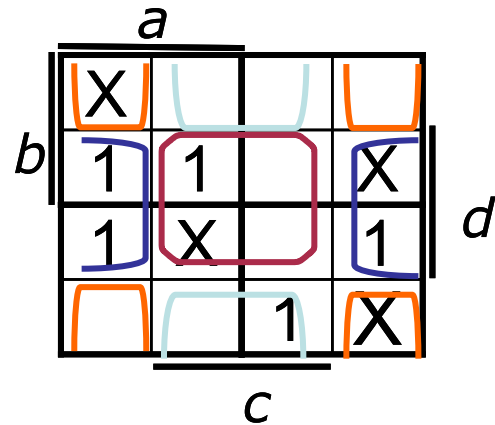
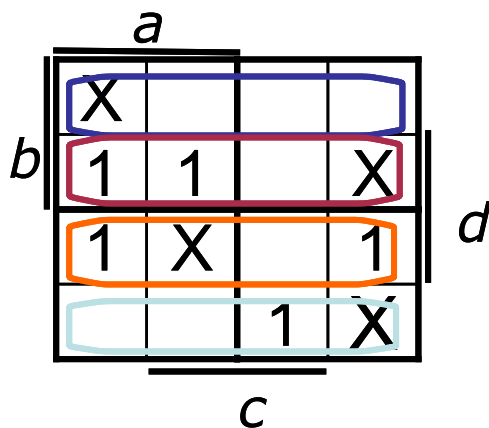


Nato izberemo naslovni spremenljivki b in c, (levi Veitchev diagram) in b in d (desni diagram). Pri razvoju po b in c

imamo pri $bc="11"$ najneugodnejšo funkcijo (rdeč), saj vsebuje eno samo '1'.

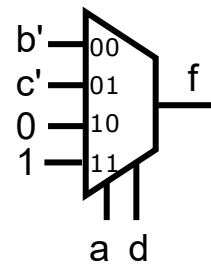


Pri razvoju po b in d dobimo eno (od dveh) možno realizacijo s funkcijskimi ostanki: $F_{00}=a'$; $F_{01}=c'$; $F_{10}=0$ in $F_{11}=a$.



Zadnja kombinacija naslovnih vhodov je cd . Pri razvoju po c in d imamo pri $cd="10"$ najneugodnejšo funkcijo (svetlo modro), saj vsebuje eno samo '1'. Najbolj ugodna kombinacija za realizacijo je torej razvoj po spremenljivkah a in d .

Končna realizacija funkcije je lahko:



Druga možna rešitev je kombinacija naslovnih spremenljivk b in d .

Rešitev 3. naloge:

Postopek sinteze zahteva, da zapišemo tabelo prehajanja stanj števca:

SMER	Q ₂	Q ₁	Q ₀	Q ₂	Q ₁	Q ₀	T ₂	T ₁	T ₀
0	0	0	0	0	0	1	0	0	1
0	0	0	1	0	1	0	0	1	1
0	0	1	0	0	1	1	0	0	1
0	0	1	1	1	0	0	1	1	1
0	1	0	0	1	0	1	0	0	1
0	1	0	1	1	1	0	0	1	1
0	1	1	0	1	1	1	0	0	1
0	1	1	1	0	0	0	1	1	1
1	0	0	0	1	1	1	1	1	1
1	0	0	1	0	0	0	0	0	1
1	0	1	0	0	0	1	0	1	1
1	0	1	1	0	1	0	0	0	1
1	1	0	0	0	1	1	1	1	1
1	1	0	1	1	0	0	0	0	1
1	1	1	0	1	0	1	0	1	1
1	1	1	1	1	1	0	0	0	1

Normalna analiza bi zahtevala, da narišemo Veitch-eve diagrame za štiri spremenljivke za vsak vhod T–FF, vendar ker so T–FF po svoji naravi primerni za realizacijo števec, so praviloma njihove vhodne enačbe zelo enostavne. Iz tabele prehajanja stanj števca določimo enačbe T–FF:
Iz stolpca T₀ se vidi, da je T₀=1'. Iz stolpca T₁ se vidi, da se ponavlja

vzorec 01, če je SMER='0' in 10, če je SMER='1'.

SMER	T ₁
0	Q ₀
1	Q ₀ '

kar lahko kratko zapišemo kot:

$$T_1 = \text{SMER} \cdot \overline{Q_0} + \overline{\text{SMER}} \cdot Q_0 = \text{SMER} \oplus Q_0$$

Za T₂ se da enostavno ugotoviti realizacijo iz Veitch-eve diagrama:

SMER				Q ₀
Q ₂	1	0	0	
	0	0	1	
	0	0	1	
	1	0	0	
Q ₁				

$$T_2 = \text{SMER} \cdot \overline{Q_1} \cdot \overline{Q_0} + \overline{\text{SMER}} \cdot Q_1 \cdot Q_0$$

V enačbi za T₂ poiščemo podobnosti z enačbo za T₁: Enačba za T₁ vsebuje konjunkciji SMER·Q₀' in SMER'·Q₀, ki sta vsebovani tudi v enačbi za T₂, kar nam dodatno poenostavi realizacijo števca. Obenem nam taka realizacija nakazuje osnovno strukturo, ki jo lahko s ponavljanjem razširimo v večbitni dvosmerni sinhroni števec.

Primer podobnega vezja 4-bitnega dvojiškega dvosmernega števca, ki ima še vzporedno nalaganje je 74191¹. Če boste primerjali našo realizacijo in realizacijo v podatkovnem listu, boste opazili, da je v dejanski realizaciji 74191 precej več večvhodnih AND vrat: Delno je razlog za to v dodani logiki za vzporedno nalaganje, delno pa tudi zato, da zagotovimo enakomerno zakasnitev med posameznimi stopnjami števca.

¹ <http://www.alldatasheet.com/view.jsp?Searchword=74191>

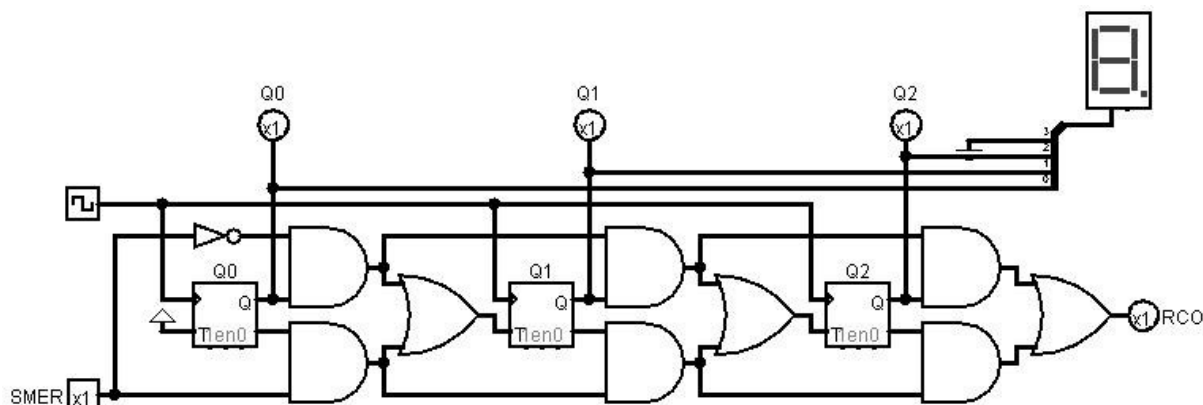
Ko enkrat narišemo vezje dvosmernega števca, zelo spominja na združitev sinhronnega števca za štetje navzgor in sinhronnega števca za štetje navzdol: Če bi števec vseboval samo zgornja AND vrata (vezanih neposredno na T vhod – brez OR) bi bil to števec navzgor, če pa samo spodnja AND bi bil števec navzdol. Signal SMER določa katera AND vrata so omogočena:

- zgornja AND vrata, ko je SMER='0' – štejemo naraščajoče,
- spodnja AND vrata, ko je SMER='1' – štejemo padajoče.

Pri tovrstnih števcih želimo realizirati tudi signal za proženje naslednjih stopenj števca (RCO – oz. ripple carry out, včasih tudi TC – terminal count). RCO je signal, ki postane '1' ob prehodu iz najvišjega stanja števca (v našem primeru je to "111") v stanje "000" pri štetju navzgor in ob prehodu "000" v najvišje stanje števca pri štetju navzdol:

SMER	RCO
0	$Q_0 \cdot Q_1 \cdot Q_2$
1	$Q_0' \cdot Q_1' \cdot Q_2'$

Tak signal uporabljamo pri realizaciji večbitnih števecv tako, da izdelane 3 bitne števecve vezemo kaskadno – torej da signal RCO vezemo na EN signal naslednjega vezja. Za realizacijo takega signala bi narisali enako kombinacijo AND in OR vrat še na izhodu Q_2 , kot kaže spodnja slika:



Opis delovanja in vezje števca je v predlogah vaj na domači strani predmeta v imeniku Logisim\counter\ counter_up_down_3_bit_using_T_FF.circ

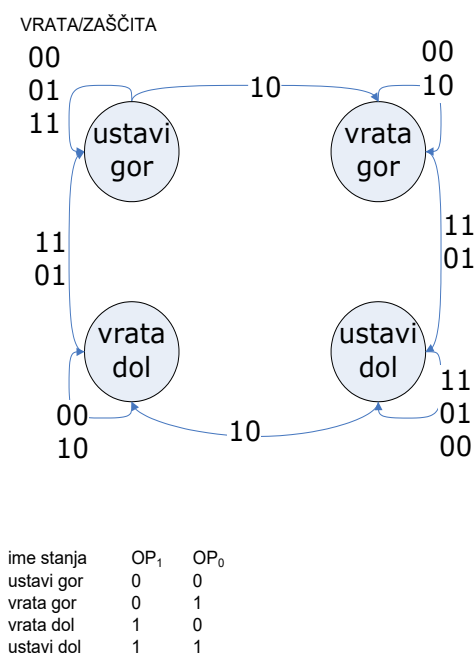
Večina števecv je realizirana v 4-bitni zasnovi, tako da glede na vrednost RCO signala ločimo dve skupini števecv:

- desetiški (BCD) števeci, katerih RCO se postavi na '1' takrat, ko števec preide iz stanja "1001" v "0000" in
- dvojiški (binarni), katerih RCO se postavi na '1' takrat, ko števec preide iz stanja "1111" v "0000". Več o delovanju RCO najdete v opisu delovanja števecv 74161².

² <http://www.alldatasheet.com/view.jsp?Searchword=74161>

Rešitev 4. naloge:

Narišemo Moore-ov diagram stanj:



Iz opisa naloge je razvidno, da stanje "ustavi" ni samo eno, ker si moramo zapomniti v katero smer so se gibala vrata, da bi lahko šli v nasprotni smeri. Glede na to imamo stanji "ustavi gor", ki določa, da se bodo vrata ob naslednjem pritisku na gumb gibala gor in stanje "ustavi dol", ki določa, da se bodo vrata ob naslednjem pritisku na gumb gibala dol. Če stanja ločimo tako, potem v stanju "ustavi gor" ostajamo toliko časa, dokler ne pritisnemo VRATA in jasno na motorju ni napake, se pravi kombinacija "10". Vrata se nato pomikajo gor (preidemo v stanje "vrata gor"). V tem stanju lahko tipko spustimo in vrata se pomikajo navzgor. To se dogaja toliko časa, dokler ne naletimo na pogoj ZAŠČITA='1' (se pravi kombinaciji

"11" in "01". Ko postane pogoj ZAŠČITA='1' se postavimo v stanje "ustavi dol" in v tem stanju ostajamo dokler vztraja pogoj ZAŠČITA='1' oz. dokler ne pritisnemo tipke VRATA='1' (kombinacija "10"). Takrat na podoben način preidemo v stanje "vrata dol", kjer ostanemo dokler ne naletimo na oviro (tla prostora recimo), ko preidemo v stanje "ustavi gor".