

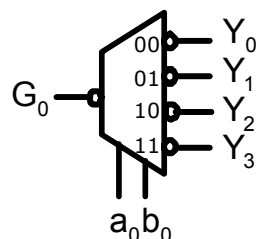
RAZVOJ DIGITALNIH SISTEMOV

Izpit

13. 02. 2017

1. Realizirajte funkcijo $f^3 = V(1, 2, 4, 7)$ z redundantnimi mintermi pri $V_x(0, 3, 6)$ z enim TTL dekodrom 74139. Dekoder 74139 ima vhod za omogočenje elementa (G) in izhode Y_0, Y_1, Y_2, Y_3 v negativni logiki. Njegovo delovanje povzema spodnja tabela:

G_0	a_0	b_0	Y_0	Y_1	Y_2	Y_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

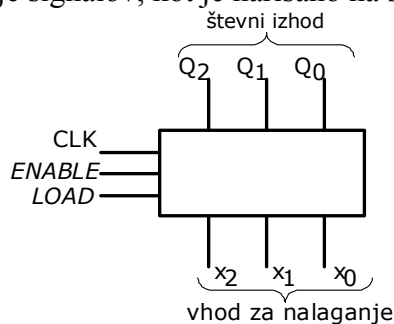


2. Realizirajte podano funkcijo f z redundancami z eno 4-bitno aritmetično-logično enoto (ALU). Negacije vhodnih spremenljivk izvedite z ALU.

$$f(x_1, x_2, x_3, x_4) = V(0, 5, 6, 9, 10, 12) \text{ in } V_x(3, 15)$$

3. Prikažite sintezo 3-bitnega sinhronega števca navzgor z omogočanjem štetja ($ENABLE$) in vzporednim nalaganjem ($LOAD$) z D flip-flopi, izbiralniki 2/1 in logičnimi vrati. Logika vseh krmilnih signalov je pozitivna. S tovrstnim števcem realizirajte števec, ki šteje 2, 3, 4, 5, 2, 3, 4, 5 ...

Uporabite poimenovanje signalov, kot je narisano na spodnji sliki.



4. Pretvorite podani avtomat končnih stanj, ki je podan v Mealy-evi obliki, v Moore-ovo obliko.

Trenutno stanje	Naslednje stanje	
	$x1$	$x2$
S1	S2/y1	S3/y0
S2	S5/y1	S3/y1
S3	S1/y0	S5/y0
S4	S2/y1	S4/y1
S5	S3/y0	S2/y1

Čas pisanja je 60 minut. Vsaka naloga je vredna 10 točk.

Na list z rešitvami se podpišite in napišite še vpisno številko ter kateri predmet pišete (VSŠ, UNI).

Rezultati bodo objavljeni na: <https://estudent.fri.uni-lj.si>

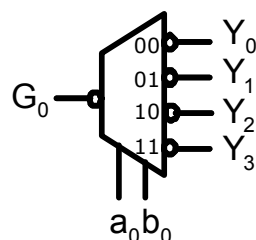
DEVELOPMENT OF DIGITAL SYSTEMS

Written examination

13. 02. 2017

- Implement a given function $f^3 = V(1, 2, 4, 7)$ with redundant minterms $V_x(0, 3, 6)$ using a single TTL decoder 74139. Decoder 74139 features an enable input (G) and outputs Y_0, Y_1, Y_2, Y_3 in negative logic. Its operation and symbol are summarized below:

G_0	a_0	b_0	Y_0	Y_1	Y_2	Y_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

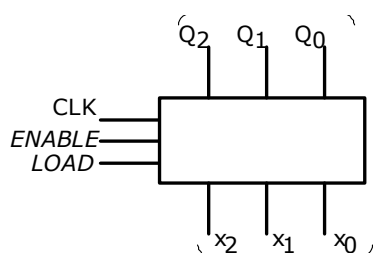


- Implement a given function f using a single 4-bit arithmetic logic unit (ALU). Any resulting negations of variables must be implemented within this ALU.

$$f(x_1, x_2, x_3, x_4) = V(0, 5, 6, 9, 10, 12) \text{ in } V_x(3, 15)$$

- Implement a synchronous 3 bit *up* binary counter using D type flip-flops, multiplexers 2/1 and logic gates: Design the state transition table, determine the flip-flop input equations and draw the resulting circuit using signal names as specified below. The counter has a count enable input (*ENABLE*), a parallel load signal (*LOAD*), which loads a 3-bit input x_2, x_1, x_0 into 3-bit output Q_2, Q_1, Q_0 . Name the signals according to figure below.

Using the this element, implement a counter, which will count in sequence 2, 3, 4, 5, 2, 3, 4, 5 ...



- Convert a given Mealy type finite state machine into its functionally equivalent Moore form.

Current state	Next state	
	$x1$	$x2$
S1	S2/y1	S3/y0
S2	S5/y1	S3/y1
S3	S1/y0	S5/y0
S4	S2/y1	S4/y1
S5	S3/y0	S2/y1

Čas pisanja je 60 minut. Vsaka naloga je vredna 10 točk.

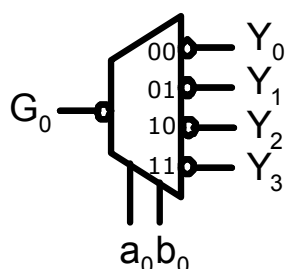
Na list z rešitvami se podpišite in napišite še vpisno številko ter kateri predmet pišete (VSŠ, UNI).

Rezultati bodo objavljeni na: <https://estudent.fri.uni-lj.si>

Rešitev 1. naloge:

Delovanje dekoderja 74139¹ povzema spodnja tabela:

G_0	a_0	b_0	Y_0	Y_1	Y_2	Y_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0



Funkcijo f narišemo v Veitch–ev diagram, da si jo lažje predstavljamo:

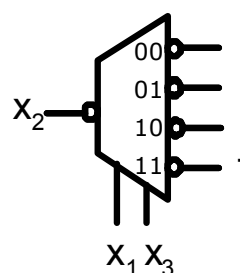
	x_1			
x_2	X	1	X	1
	1	0	1	X
	x_3			

Čim imamo na voljo dekodirnik z ENABLE vhodom v negativni logiki preverimo ali obstaja spremenljivka v osnovni ali negirani obliki, pri kateri so vsa polja enaka '1' vključno z redundancami X.

V zgornjem Veitch–evem diagramu je to spremenljivka x_2 : Namreč, če je $x_2=1$, potem lahko vse redundance izberemo tako, da bo $f=1$ za vse vrednosti $x_2=1$. Ko določimo spremenljivko za omogočenje elementa (G), opazujemo samo preostali del Veitch–evega diagrama. Spodnje 4 vrednosti diagrama narišemo v novem Veitch–evem diagramu 2 spremenljivk.

	x_1	
x_3	0	1
	1	X

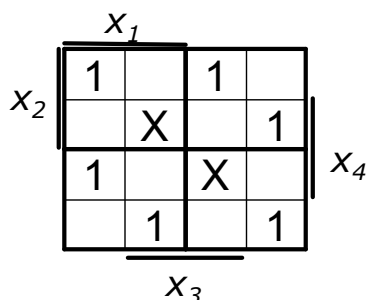
Dekoder ima aktivno nizke izhode, zato iz nastalega diagrama realiziramo *negacijo* funkcije f zato v Veitch–evem diagramu združujemo ničle, kar nastopa samo v primeru ko sta $x_1=1$ in $x_3=1$.



¹<http://www.alldatasheet.com/view.jsp?Searchword=74HC139>

Rešitev 2. naloge:

Funkcijo najprej izrišemo v Veitch–ev diagram:



Funkcija vsebuje same diagonalne člene, zato realizacija v obliki KNO oz. DNO ne nudi minimalne oblike. Če se izkaže, da je funkcija linearna, jo lahko realiziramo s pomočjo XOR funkcij. Linearnost funkcije ugotavljamo tako, da prepogibamo kvadrate diagrama: Začnemo v desnem spodnjem kotu (kjer je minterm 0) in prepognemo kvadrat navzgor, da se spremeni samo ena spremenljivka naenkrat (x_4 postane 0 v prvi iteraciji).

S pomočjo Veitch–evega diagrama izračunamo koeficiente.

Iz enačb sledi: $k_0=1$ in $k_0 \oplus k_3=0$, kar pomeni $1 \oplus k_3=0 \rightarrow k_3=1$.

In če napišemo še enačbo za $k_0 \oplus k_2=0$, kar pomeni $1 \oplus k_2=0$ sledi da je $k_2=1$.

Iz enačbe $k_0 \oplus k_2 \oplus k_4=1$, kar pomeni $1 \oplus 1 \oplus k_4=1 \rightarrow k_4=1$.

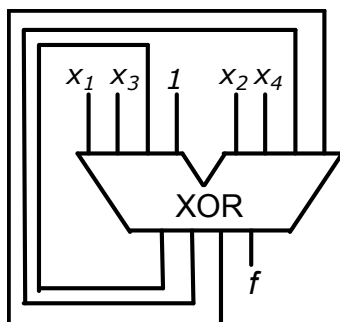
Analiziramo naprej in dobimo $k_0 \oplus k_1 \oplus k_2=1$, kar pomeni $1 \oplus k_1 \oplus 1=0 \rightarrow k_1=1$.

Vstavimo dobljene koeficiente v enačbo za splošno izražavo in dobimo:

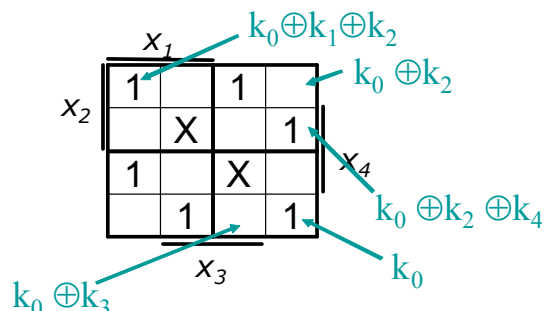
$$f(x_1, x_2, x_3, x_4) = 1 \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_4$$

Aritmetično–logično enota lahko poleg aritmetičnih naenkrat realizira štiri dvovhodne logične operacije *istega tipa* (OR, AND, NOT, NOR, NAND, XOR, XNOR), zato nas zanima realizacija zgornje funkcije z dvovhodnimi operatorji enega tipa. Pri realizaciji uporabimo lastnost združevanja, ki velja za XOR funkcijo.

$$f(x_1, x_2, x_3, x_4) = 1 \oplus ((x_1 \oplus x_2) \oplus (x_3 \oplus x_4))$$



Opazujemo, ali se prepogne na novi kvadrat čisto enako ali pa popolnoma negirano. Če postavimo obe redundanci na '1', lahko s prepogibanjem ugotovimo, da je funkcija linearna.



Podana funkcija je funkcija 4 spremenljivk, zato lahko njeno splošno izražavo kot linearno funkcijo pišemo kot:

$$f(x_1, x_2, x_3, x_4) = k_0 \oplus k_1 x_1 \oplus k_2 x_2 \oplus k_3 x_3 \oplus k_4 x_4$$

Rešitev 3. naloge:

trenutno stanje			naslednje stanje			D-FF		
Q ₂	Q ₁	Q ₀	Q ₂	Q ₁	Q ₀	D ₂	D ₁	D ₀
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	0
0	1	0	0	1	1	0	1	1
0	1	1	1	0	0	1	0	0
1	0	0	1	0	1	1	0	1
1	0	1	1	1	0	1	1	0
1	1	0	1	1	1	1	1	1
1	1	1	0	0	0	0	0	0

Podobno za D₂ narišemo Veitchev diagram

D_2 :

	Q ₂			
Q ₁	1	0	1	0
	1	1	0	0
	Q ₀			

Za D₂ sledi:

$$D_2 = Q_2 Q_1' + Q_2 Q_0' + Q_2' Q_1 Q_0$$

Iz tabele prehajanja stanj števca določimo enačbe D-FF:

Za D₀ se iz tabele vidi $D_0 = Q_0'$

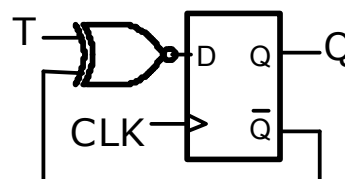
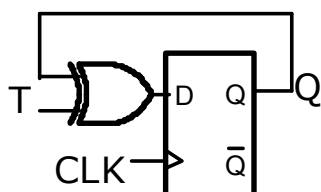
Za D₁ narišemo Veitchev diagram:

D_1 :

	Q ₂			
Q ₁	1	0	0	1
	0	1	1	0
	Q ₀			

$$D_1 = Q_0 \oplus Q_1$$

Če enačbo $D_0 = Q_0'$ zapišemo kot $D_0 = 1 \oplus Q_0$ in jo narišemo v vezju, smo pravzaprav realizirali T-FF s pomočjo D-FF, kot kaže levi del spodnje slike:



Slika 1: Realizacija T-FF s pomočjo D-FF in XOR vrat (levo) in XNOR vrat (desno)

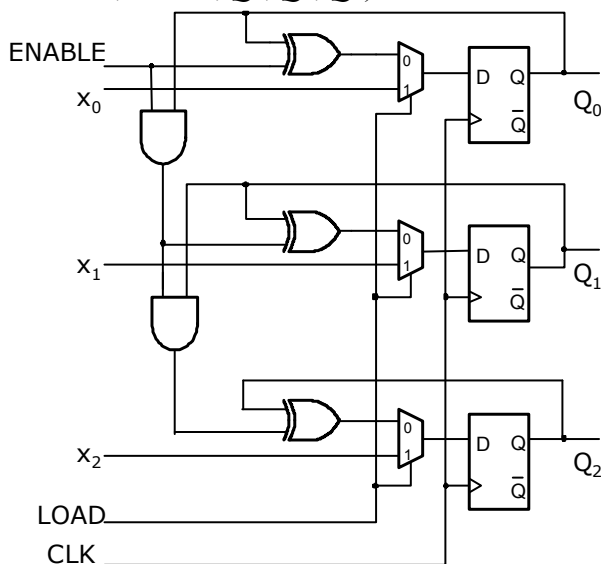
Naloga pravi, da moramo izdelati števec, ki ima vhod za omogočanje štetja (*ENABLE*): Če prvemu "T-FF" (D-FF z XOR vrati) postavimo vhod $T_0 = '0'$ namesto $T_0 = '1'$, vsi FF ne bodo šteli, ampak bodo ohranjali stanje. Torej, če na vhod T_0 postavimo zunanji signal *ENABLE*, števec ne bo štel, ampak ohranjal stanje, če bo *ENABLE* = '0'. V verigi sinhronnega števca so namreč vsi FF vezani tako, da so odvisni od prvega FF. Če stanje ohranja prvi, ga bodo tudi vsi ostali.

Čas pisanja je 60 minut. Vsaka naloga je vredna 10 točk.

Na list z rešitvami se podpišite in napišite še vpisno številko ter kateri predmet pišete (VSŠ, UNI).

Rezultati bodo objavljeni na: <https://estudent.fri.uni-lj.si>

Za realizacijo signala za vzporedno nalaganje pa izkoristimo osnovno lastnost D–FF (pomnjenje). To storimo tako, da na vhod vsakega D–FF postavimo 2/1 izbiralnik, s katerim določimo, ali se bo dana informacija vpisala s števnege vhoda ali preko zunanjih priključkov. Do iste realizacije bi prišli, če bi v osnovni analizi upoštevali ta dva krmilna signala – analiza je veliko bolj zapletena, saj vsebuje Veitcheve diagrame za 5 spremenljivk ($ENABLE$, $LOAD$, Q_2 , Q_1 , Q_0).



Slika 2: Sinhroni števec z vzporednim nalaganjem (LOAD) in omogočanjem štetja (ENABLE) (3-bitna izvedba).

Če želimo z nastalim števcem šteti naraščajoče 2, 3, 4, 5 ..., moramo ko števec pride do stanja 5 ($Q_2Q_1Q_0=101_2$) števec postaviti nazaj na stanje v stanje 2 ($Q_2Q_1Q_0=x_2x_1x_0=010_2$), torej signal $LOAD$ dekodiramo s pomočjo 2–vhodnih AND vrat. Pomembno pri tem je, da se pri dekodiranju zavedamo, da se $Q_2 = '1'$ in $Q_0 = '1'$ v števeni sekvenci pojavlja samo enkrat – če bi se večkrat, bi morali dekodirati tudi Q_1 . Pri tovrstnih števcih ponavadi uporabljamo še en signal $RESET$, s katerim postavimo števec v začetno stanje, kar dosežemo tako, da na vhod D–FF za brisanje asinhrono priključimo signal $RESET$.

Nastali strukturi števca bi lahko na isti način dodali še četrti bit. Tako bi dobili podobno strukturo kot je 4–bitni sinhroni števec z vzporednim nalaganjem 74163². Pri spodnji realizaciji tega vezja so uporabljeni D–FF, proženi na negativni rob signala ure (CLK). Štetje omogočimo s signaloma ENP , ENT (ang. enable parallel, enable transfer). Štetje je izvedeno tako, da se D–FF spremenijo v T–FF, kar dosežemo s pomočjo XNOR vrat, ki imajo en vhod vezan na števeni signal ($ENT=ENP='1'$), drug vhod pa na izhod Q' FF. AND vrata pred XNOR zagotavljajo prenehanje štetja, če velja $ENT \cdot ENP \neq '1'$. Na vhodu D–FF je vezan enostaven izbiralnik iz dveh AND vrat, vezanih na OR vrata. Ta izbiralnik določa, ali bo števec štel, ali bo vzporedno naložil vrednost. Zgornja AND vrata izbiralnika so krmiljena s signalom $LOAD \cdot CLR'$, spodnja pa z $LOAD' \cdot CLR'$. Čim velja, da je $CLR' = '1'$ in $LOAD' = '0'$, se v D–FF asinhrono naloži vsebina na vseh $Q_D Q_C Q_B Q_A = DATA_D DATA_C DATA_B DATA_A$, medtem ko dokler velja $LOAD' = '1'$ in $CLR' = '1'$, bo števec štel navzgor. Če je $CLR' = '0'$, je na obeh vseh AND vrat izbiralnika '0' in stanje vseh D–FF se asinhrono postavi na $Q_D Q_C Q_B Q_A = "0000"$. Števeni izhodi so $Q_D Q_C Q_B Q_A$.

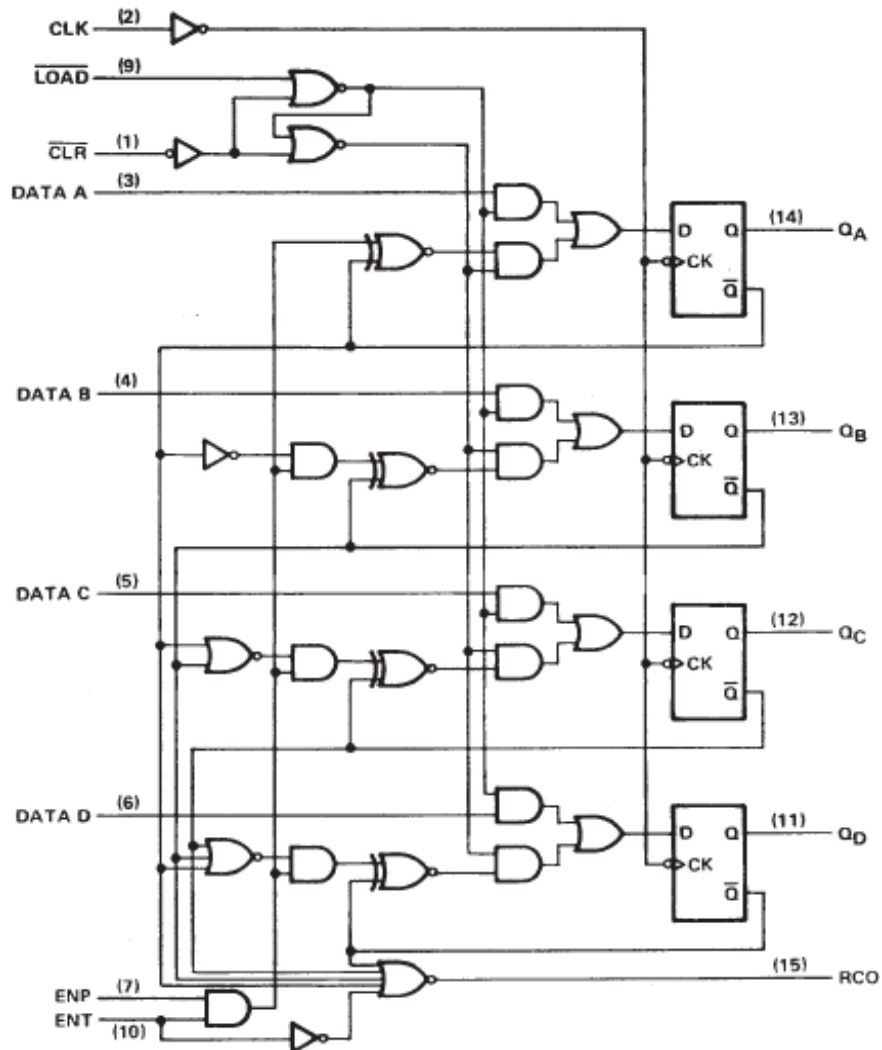
² <http://www.alldatasheet.com/view.jsp?Searchword=74163>

Čas pisanja je 60 minut. Vsaka naloga je vredna 10 točk.

Na list z rešitvami se podpišite in napišite še vpisno številko ter kateri predmet pišete (VSŠ, UNI).

Rezultati bodo objavljeni na: <https://estudent.fri.uni-lj.si>

Izhod *RCO* (*ripple carry out*) se postavi na '1', ko je števec v stanju "1111" ob tem, da je $ENT=1$. Signal *RCO* je izveden z NOR vrati na vhode katerih so priključeni negirani izhodi D-FF, kar je po De Morgan-ovem teoremu ekvivalent AND vratom.



Slika 3: Struktura 4-bitnega MSI sinhronnega števca z vzporednim nalaganjem (74163).

Čas pisanja je 60 minut. Vsaka naloga je vredna 10 točk.

Na list z rešitvami se podpišite in napišite še vpisno številko ter kateri predmet pišete (VSŠ, UNI).

Rezultati bodo objavljeni na: <https://estudent.fri.uni-lj.si>

Rešitev 4. naloge:

V zapisu vidimo, da je upoštevanih vseh pet stanj podanega Mealyjevega avtomata končnih stanj, zato zapišemo tabelo prehajanja za Moore-ov avtomat z novimi oznakami stanj in njihovimi izhodi. Prehode stanj prepišemo iz tabele Mealyjevega avtomata z novimi oznakami, kjer se vrstice pri stanjih z več različnimi izhodi ponovijo. Izhodi Mooreovega avtomata ustrezajo izhodom, ki smo jih upoštevali pri določanju stanj Mealyjevega avtomata.

Trenutno stanje	Naslednje stanje	
	x_1	x_2
S_1	S_2/y_1	S_3/y_0
S_2	S_5/y_1	S_3/y_1
S_3	S_1/y_0	S_5/y_0
S_4	S_2/y_1	S_4/y_1
S_5	S_3/y_0	S_2/y_1

Na vseh prehodih v naslednje stanje pregledamo če ob tem obstaja samo en izhod, oz. ali jih je morda več. Mealy-eva oblika avtomata namreč lahko spreminja izhode glede na trenutno stanje (S_x) in vhod (x_1 oz. x_2), zato so izhodi vezani na prehode med stanji, medtem ko je pri Moore-ovi obliki izhod vezan samo na stanje.

Zgornji avtomat ima pet stanj in dva izhoda, zato je v splošnem možnih 10 različnih stanj v pretvorjenega Moore-ovega avtomata.

Če pregledamo prehode, ki vodijo v naslednje stanje S_1 , opazimo, da bo izhod v edinem prehodu enak y_0 . V *funkcijsko* ekvivalentnem Moore-ovem avtomatu ima zato stanje S_1 izhod y_0 - zaradi enostavnosti mu damo ime S_{10} . Podobno velja za stanje S_2 , ki ga preimenujemo v S_{21} in stanje S_4 , ki ga preimenujemo v S_{41} .

Stanje S_3 pa ima na prehodih dva možna izhoda: y_0 in y_1 , zato vpeljemo dve novi Moore-ovi stanji S_{30} , z izhodom y_0 in S_{31} , z izhodom y_1 . Obnašanje avtomata v obeh Moore-ovih stanjih mora ostati enako originalu, zato polji naslednjih stanj v vrstici S_3 prepišemo, le izhod Moore-ovega stanja se ustrezno spremeni (y_0 v primeru S_{30} in y_1 v primeru S_{31}). Podobno storimo za stanje S_5 in dobimo pretvorjeno, *funkcijsko* ekvivalentno obliko.

Trenutno stanje	Naslednje stanje		Izhod
	x_1	x_2	
S_{10}	S_{21}	S_{30}	y_0
S_{21}	S_{51}	S_{31}	y_1
S_{30}	S_{10}	S_{50}	y_0
S_{31}	S_{10}	S_{50}	y_1
S_{41}	S_{21}	S_{41}	y_1
S_{50}	S_{30}	S_{21}	y_0
S_{51}	S_{30}	S_{21}	y_1

Dobljeni avtomat zgolj *funkcijsko* pooseblja Mealy-evo izvedbo - popolnoma ga ne more nikdar nadomestiti, saj Mealy-vi avtomati na izhodu izkazujejo asinhrono obnašanje (vhod - in posledično tudi izhod - se namreč lahko spremeni neodvisno od ure). Tega Moore-ov avtomat ne izkazuje, saj v naslednje stanje preide le ob aktivnem robu signala ure, zato se Moore-ov *funkcijski* ekvivalent vedno obnaša počasneje - za en cikel signala ure. Slednji je namreč potreben, da pretvorjeni avtomat preide v novo Moore-ovo stanje (npr. S_{31}) in spremeni izhod (npr. na y_1).

Čas pisanja je 60 minut. Vsaka naloga je vredna 10 točk.

Na list z rešitvami se podpišite in napišite še vpisno številko ter kateri predmet pišete (VSŠ, UNI).

Rezultati bodo objavljeni na: <https://estudent.fri.uni-lj.si>