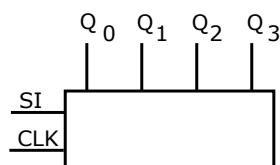


RAZVOJ DIGITALNIH SISTEMOV

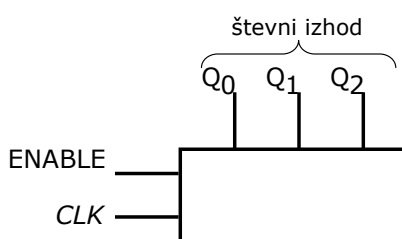
2. kolokvij

11.1. 2023

1. Realizirajte RS-flip flop z uporabo T flip flopa in logičnih vrat.
2. Narišite vezje 4-bitnega pomikalnega registra z JK celicami in logičnimi vrati. Register ima zaporedni vhod SI (ang. serial input) in vzporedni izhod (Q_0 , Q_1 , Q_2 , Q_3).



3. Prikažite sintezo 3-bitnega sinhronega števca navzgor z omogočanjem štetja (ENABLE) s T flip-flopi in logičnimi vrati. Če je $ENABLE='1'$, števec šteje, če je $ENABLE='0'$ števec ohranja stanje.



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.

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. Rezultati bodo objavljeni na domači strani predmeta.

Rešitev 1. naloge

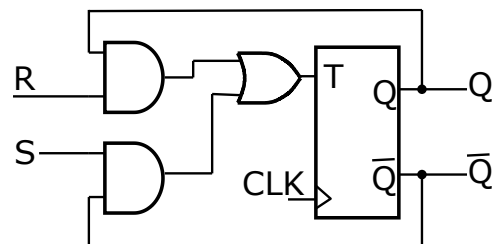
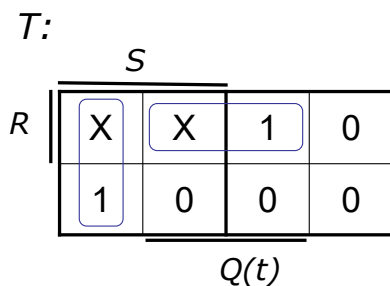
Za vezje RS-FF narišemo pravilnostno tabelo, pri čemer na vhodni strani zberemo vhode R, S in trenutno stanje $Q(t)$, na izhodni pa naslednje stanje $Q(t+1)$. RS-FF opravlja tri funkcije (HOLD, RESET, SET) glede na kombinacijo vhodnih signalov, medtem ko T-FF opravlja samo dve (HOLD, INVERT).

S	R	$Q(t)$	$Q(t+1)$	T	funkcija RS	funkcija T
0	0	0	0	0	HOLD	HOLD
0	0	1	1	0	HOLD	HOLD
0	1	0	0	0	RESET	HOLD
0	1	1	0	1	RESET	INVERT
1	0	0	1	1	SET	INVERT
1	0	1	1	0	SET	HOLD
1	1	0	X	X	/	/
1	1	1	X	X	/	/

Iz tabele narišemo Veitchev diagram za vhod T v odvisnosti od vhodov R, S in trenutnega stanja $Q(t)$.

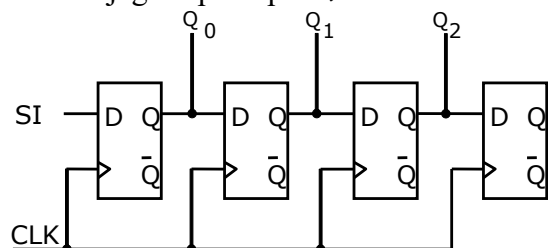
$$T = S \cdot \overline{Q(t)} + R \cdot Q(t)$$

Vezje narišemo.



Rešitev 2. naloge:

Zaporedno–vzporedni (SIPO) pomikalni register, realiziran s pomočjo D–FF, je veriga kaskadno vezanih D–FF, v kateri je izhod prejšnjega flip–flopa Q_{i-1} vezan na vhod naslednjega flip–flopa D_i .



Če želimo pomikalni register sestaviti iz JK–FF in logičnih vrat, je en način realizacije realizirati celico D–FF s pomočjo JK–FF in logičnih vrat.

V ta namen zapišemo tabelo D–FF, pri kateri dodamo izhodni stolpec JK vhodov.

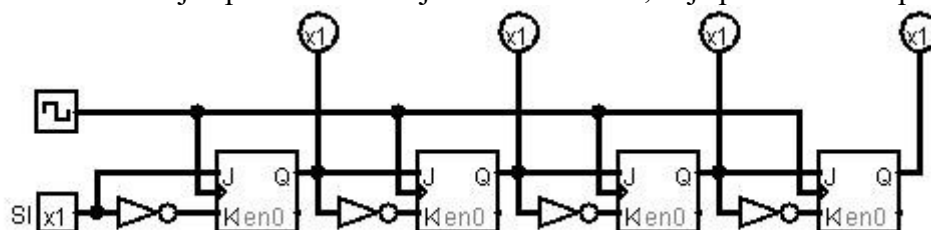
D	$Q(t)$	$Q(t+1)$	J	K	Pomen stanja
0	0	0	0	X	RESET/HOLD
0	1	0	X	1	INVERT/RESET
1	0	1	1	X	INVERT/SET
1	1	1	X	0	SET/HOLD

Iz tabele izrazimo J in K vhoda s pomočjo D vhoda in trenutnega stanja $Q(t)$. Če redundance (X) postavimo primerno, dobimo:

$$J = D \text{ in } K = \overline{D}$$

Če nastali D–FF sestavimo skupaj v 4–bitni pomikalni register dobimo spodnjo realizacijo.

Ena možna rešitev je opisana realizacija D–FF z JK–FF, ki je prikazana na spodnji sliki.

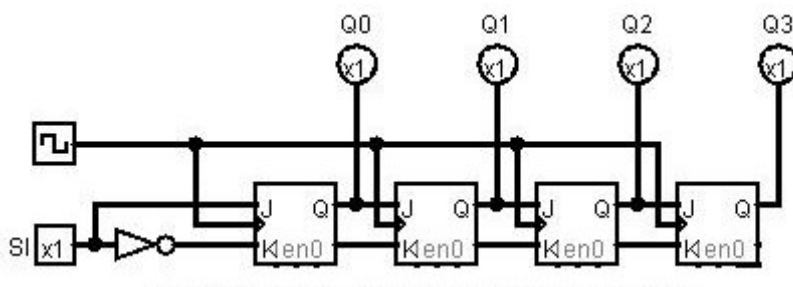


Drugo možnost predstavlja neposredna realizacija s pomočjo tabele pomikanja med opazovanim mestom Q_i in naslednjim mestom Q_{i+1} pomikalnega registra. Ponovno zapišemo vzbujačno tabelo za opazovano mesto in določimo vhode glede na spremembo stanja ($Q_{i+1}(t) \rightarrow Q_{i+1}(t+1)$) na $(i+1)$ mestu.

Trenutna vsebina registra		Vsebinska registra ob pomiku	Vhoda FF na mestu $i+1$		Pomen stanja
$Q_i(t)$	$Q_{i+1}(t)$	$Q_{i+1}(t+1)$	J_{i+1}	K_{i+1}	
0	0	0	0	X	RESET/HOLD
0	1	0	X	1	INVERT/RESET
1	0	1	1	X	INVERT/SET
1	1	1	X	0	SET/HOLD

Če v dobljenih vseh za mesto $(i+1)$ izberemo primerne redundance, dobimo vhoda $J_{i+1} = Q_i(t)$ in $K_{i+1} = Q_i(t)'$. To velja tudi za vhodno mesto, torej bomo na vhodni JK–FF vezali vhod $J_0 = x(t)$ in $K_0 = x(t)'$.

Neposredna realizacija pomikalnega registra z JK–FF je prikazana na spodnji sliki.

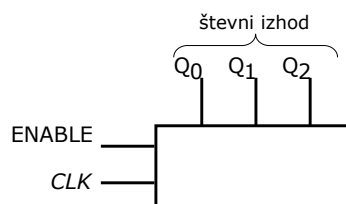


Vezje se nahaja v Logisim predlogah rešenih nalog na domači strani predmeta:
Logisim\shift_reg\shift_reg_4bit_using_jkff.circ

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. Rezultati bodo objavljeni na domači strani predmeta.

Rešitev 3. naloge:

Postopek sinteze zahteva, da zapišemo tabelo prehajanja stanj števca:



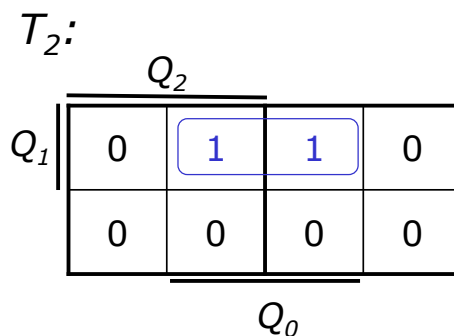
trenutno stanje			naslednje stanje			T-FF		
Q ₂	Q ₁	Q ₀	Q ₂	Q ₁	Q ₀	T ₂	T ₁	T ₀
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1	1
1	1	0	1	1	1	0	0	1
1	1	1	0	0	0	1	1	1

Iz tabele prehajanja stanj števca določimo enačbe T-FF:

Za T_0 se iz tabele vidi $T_0 = 1$

Za T_1 se iz tabele vidi $T_1 = Q_0 \cdot 1 = Q_0 \cdot T_0$

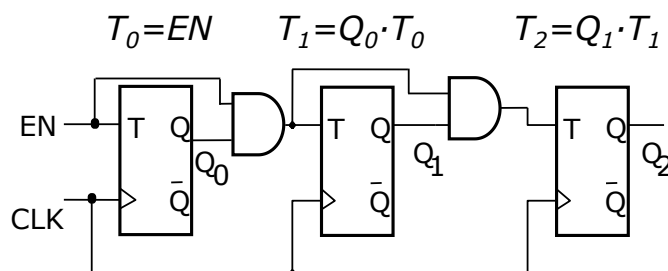
Za T_2 narišemo Veitchev diagram



$$T_2 = Q_0 \cdot Q_1 = T_1 \cdot Q_1$$

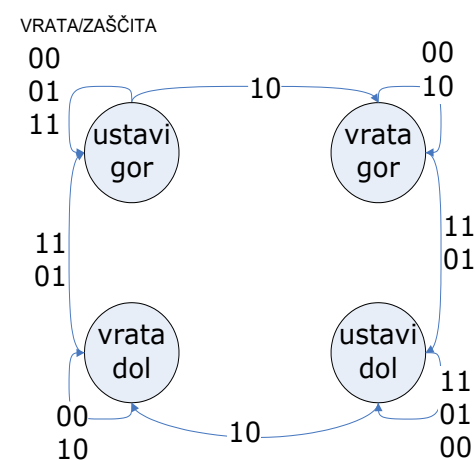
Naloga pravi, da moramo dodati še vhod za omogočanje štetja (ENABLE). Če vsem T-FF postavimo $T_0 = '0'$ namesto $T_0 = '1'$, flip-flopi števca ne bodo šteli, ampak bodo ohranjali stanje (po definiciji T-FF). Če torej na prvi vhod T_0 postavimo zunanji signal ENABLE oz. EN, števec ne bo štel, ampak ohranjal stanje, če bo $EN = '0'$. Na vse naslednje FF priključimo prejšnji vhod ($T_{i+1} = Q_i \cdot T_i$), zato so izhodi vseh AND vrat enaki '0', ko je $EN = '0'$.

Do istega sklepa bi prišli, če bi risali Veitcheve diagrame za 4 spremenljivke (EN, Q_2 , Q_1 , Q_0).



Rešitev 4. naloge:

Narišemo Mooreov diagram stanj:



ime stanja	OP ₁	OP ₀
ustavi gor	0	0
vrata gor	0	1
vrata dol	1	0
ustavi dol	1	1

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 stanja "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".

Takšna realizacija še zdaleč ni optimalna: Bolje bi bilo, če bi avtomat realizirali kot Mealyev tip. Dejanska realizacija ne vsebuje avtomata, temveč en T–FF in relejno logiko.