

# RAZVOJ DIGITALNIH SISTEMOV

Izpit 18. 02. 2011

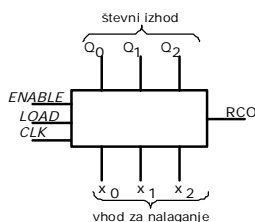
1. Določite minimalno normalno obliko (MNO) za logično funkcijo  $f$  z redundantnimi vhodnimi kombinacijami.

$$f^4 = \sum (0, 1, 6, 7, 14, 15) \text{ in } \sum_x (3, 5, 9, 11, 13)$$

2. Pretvorite podano število  $x$ , zapisano v šestnajstiškem zapisu IEEE754 oblike enojne natančnosti (32 bit single precision) v realno število:

$$x = 3FF00000_{16}$$

3. Prikažite sintezo 3-bitnega sinhronnega števca navzgor z omogočanjem štetja (ENABLE) in vzporednim nalaganjem (LOAD) z D flip-flopi, izbiralniki 2/1 in logičnimi vrati. Podatek se vzporedno nalaga z vhodov ( $x_2, x_1, x_0$ ). Števec naj ima poleg števnega izhoda ( $Q_2, Q_1, Q_0$ ) tudi izhodni prenos za krmiljenje naslednjih stopenj (RCO – ripple carry out). Logika vseh krmilnih signalov je pozitivna. Uporabite poimenovanje signalov, kot je narisano na spodnji sliki.



4. Načrtajte diagram stanj Moore-ovega 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 <sub>1</sub> | OP <sub>0</sub> |                            |
| 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.

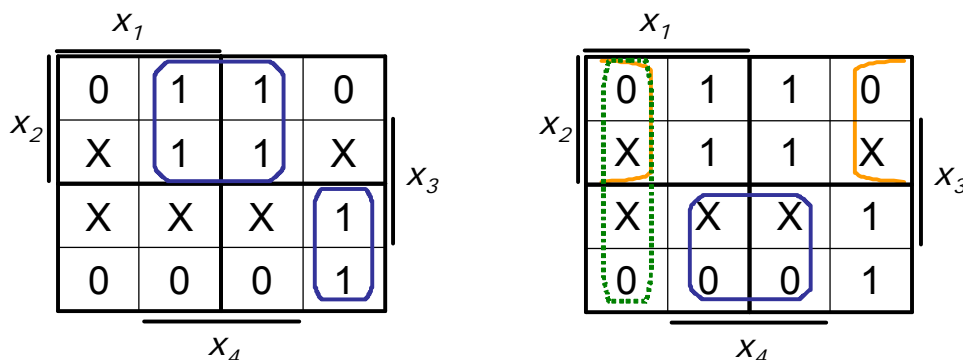
Na list z rešitvami se podpišite in napišite še vpisno številko ter kateri predmet pišete (VSŠ, UNI).

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

Rešitev 1. naloge:

Funkcijo v PDNO z redundancami moramo izpisati v Veitch–ev diagram, od koder bomo lahko izvajali postopek minimizacije.

$$f^4 = \sum(0,1,6,7,14,15) \text{ in } \sum_x(3,5,9,11,13)$$



Prikazana sta dva Veitch–eva diagrama. Levega uporabljamo za tvorbo MDNO, saj prikazuje funkcijo  $f$ , desnega za tvorbo MKNO, ker združujemo minterme negirane funkcije.

MDNO:

$$f_{MDNO} = x_2 \cdot x_4 + \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_4}$$

Za MKNO pa združujemo ničle (izražamo negirano funkcijo) in izrazimo minimalno obliko negirane funkcije. Nato uporabimo De Morgan–ov teorem, da pridemo do konjunktivne izražave.

$$\begin{aligned} \overline{f} &= x_1 \cdot \overline{x_4} + \overline{x_2} \cdot x_4 + x_2 \cdot \overline{x_4} \\ f &= \overline{x_1 \cdot \overline{x_4} + \overline{x_2} \cdot x_4 + x_2 \cdot \overline{x_4}} \\ f &= \overline{x_1 + x_4} \cdot \overline{x_2 + x_4} \cdot \overline{x_2 + x_4} \\ f_{MKNO} &= (\overline{x_1 + x_4}) \cdot (\overline{x_2 + x_4}) \cdot (\overline{x_2 + x_4}) \end{aligned}$$

Za minimalno normalno obliko moramo določiti, katera izvedba funkcije je cenejša (manjši COST vezja).

| OBLIKA | VRAT | VHODOV | COST |
|--------|------|--------|------|
| MDNO   | 3    | 7      | 10   |
| MKNO   | 4    | 9      | 13   |

Cenejša izvedba je MDNO, saj je strošek vezja manjši (COST), zato je MNO=MDNO.

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 2. naloge: Število  $x$  je podano v IEEE754 zapisu enojne natančnosti.

$$x = 3FF00000_{16}$$

Število izpišemo v dvojiško obliko, tako da vsako šestnajstiško števko pretvorimo v 4 bitno dvojiško število:

|      |      |      |      |      |      |      |      |
|------|------|------|------|------|------|------|------|
| 3    | F    | F    | 0    | 0    | 0    | 0    | 0    |
| 0011 | 1111 | 1111 | 0000 | 0000 | 0000 | 0000 | 0000 |

Število razstavimo skladno z obliko IEEE754 (predznak, odmaknjen eksponent, mantisa)

**0 01111111 111000000000000000000000**

Iz bita predznaka sledi, da bo število pozitivno.

Izračunamo še vrednost eksponenta, tako da upoštevamo odmik: Odmaknjen eksponent znaša

**01111111**<sub>2</sub> = **3F**<sub>16</sub> = **127**<sub>10</sub>. Če izračunamo vrednost dejanskega eksponenta odštejemo odmik (127<sub>10</sub>) tako da je dejanski eksponent enak 0.

Mantisi dodamo vodilno enico in decimalno piko v dvojiškem sistemu premaknemo za vrednost eksponenta (v našem primeru ne premikamo pike, ker je eksponent nič).

**1.111000000000000000000000**

Dobljeno vrednost pretvorimo v desetiški sistem.

$$x_2 = 1.111000000000000000000000$$

$$x_{10} = 1 \cdot 2^0 + 1 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3} + 0 \cdot 2^{-4} + \dots$$

$$x_{10} = 1 + 0.5 + 0.25 + 0.125 = 1.875$$

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 3. naloge:

Postopek sinteze zahteva, da zapišemo tabelo prehajanja stanj števca:

| trenutno stanje |                |                | naslednje stanje |                |                | D–FF           |                |                |
|-----------------|----------------|----------------|------------------|----------------|----------------|----------------|----------------|----------------|
| Q <sub>2</sub>  | Q <sub>1</sub> | Q <sub>0</sub> | Q <sub>2</sub>   | Q <sub>1</sub> | Q <sub>0</sub> | D <sub>2</sub> | D <sub>1</sub> | D <sub>0</sub> |
| 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              |

Iz tabele prehajanja stanj števca določimo enačbe D–FF:

Za D<sub>0</sub> se iz tabele vidi D<sub>0</sub> = Q<sub>0</sub>'

Za D<sub>1</sub> narišemo Veitchev diagram

D<sub>1</sub>:

|                |   |                |   |
|----------------|---|----------------|---|
|                |   | Q <sub>2</sub> |   |
| Q <sub>1</sub> | 1 | 0              | 0 |
|                | 0 | 1              | 1 |
|                |   | Q <sub>0</sub> |   |

$$D_1 = Q_0 \oplus Q_1$$

Podobno za D<sub>2</sub> narišemo Veitchev diagram

D<sub>2</sub>:

|                |   |                |   |
|----------------|---|----------------|---|
|                |   | Q <sub>2</sub> |   |
| Q <sub>1</sub> | 1 | 0              | 1 |
|                | 1 | 1              | 0 |
|                |   | Q <sub>0</sub> |   |

Za D<sub>2</sub> sledi:

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

iz česar lahko izpostavimo:

$$D_2 = Q_2 \cdot (Q_1' + Q_0') + Q_2' \cdot Q_1 \cdot Q_0$$

Uporabimo De Morganovo enakost:

$$D_2 = Q_2 \cdot (Q_1 Q_0)' + Q_2' \cdot Q_1 \cdot Q_0$$

iz česar sledi:

$$D_2 = Q_2 \cdot (Q_1 \cdot Q_0)' + Q_2' \cdot Q_1 \cdot Q_0$$

Upoštevamo definicijo XOR operacije (a⊕b=a'·b+a·b')

$$D_2 = Q_2 \oplus Q_1 \cdot Q_0$$

Signal RCO postane '1' takrat, ko števec prešteje do svoje največje vrednosti – v našem primeru postane '1', ko gre stanje števca iz "111" na "000".

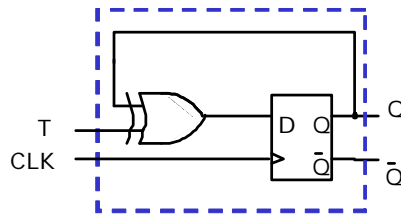
$$RCO = Q_2 \cdot Q_1 \cdot Q_0$$

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>

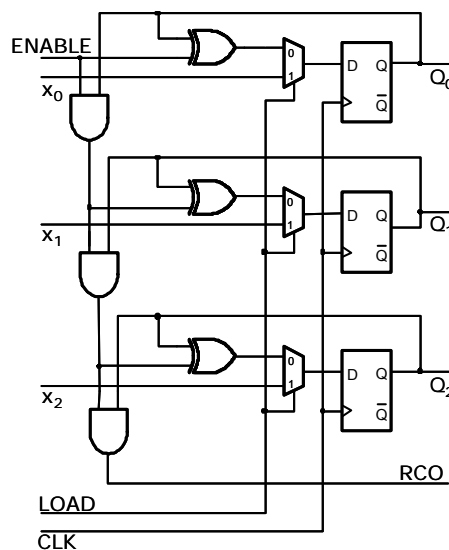
Naloga pravi, da moramo izdelati števec, ki ima vhod za omogočanje štetja (ENABLE). Če vezje analiziramo, vidimo, da smo pravzaprav realizirali T–FF s pomočjo D–FF in XOR vrat, kot kaže spodnja slika:



Slika 1: Realizacija T-FF s pomočjo D-FF.

Če prvemu "T–FF" (D–FF z XOR vrati) postavimo vhod  $T_0 = '0'$  namesto  $T_0 = '1'$ , vsi T–FF ne bodo štel, 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 sinhronega števca so namreč vsi T–FF vezani tako, da so odvisni od prvega T–FF: Če stanje ohranja prvi, ga bodo tudi vsi ostali.

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 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, 2, 3, 4, 5 ..., moramo števec, ko le–ta pride do stanja 5 ( $Q_2Q_1Q_0 = 101_2$ ) postaviti nazaj na stanje v stanje 2 ( $Q_2Q_1Q_0 = x_2x_1x_0 = 010_2$ ), torej na LOAD vhod pripeljemo s pomočjo dodatnih dvovhodnih AND vrat. Pomembno pri tem je, da se pri dekodiranju zavedamo, da se ( $Q_2 = 1$  in  $Q_0 = 1$  v števnici sekvenci pojavlja samo enkrat – če bi se večkrat bi morali dekodirati tudi  $Q_1$ ).

Pri tovrstnih števcih ponavadi uporabljamo še zunanji signal RESET, s katerim postavimo števec v začetno stanje, kar dosežemo tako, da na vhod izbiralnikov vodimo LOAD OR RESET.

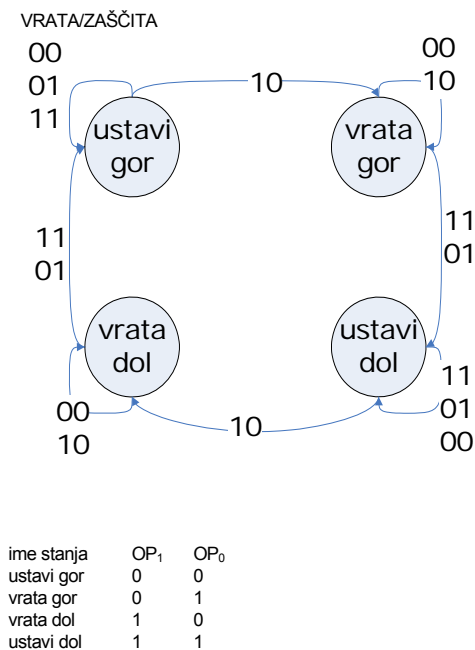
Čas pisanja je 60 minut. Vsaka naloga je vredna 10 točk.

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:

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 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 Mealy-ev tip. Dejanska realizacija ne vsebuje avtomata, temveč en T-FF in relejno logiko.