

POGLAVLJE 5.

STANDARDNE SEKVENCIJALNE MREŽE

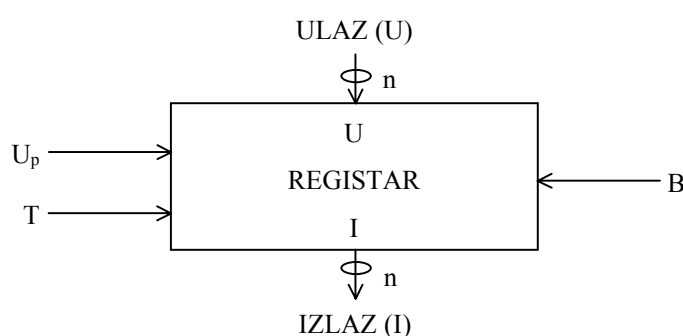
Prikazana je sinteza registara i brojača kao standardnih sekvencijalnih mreža. Ilustruje se minimizacija Bulovih jednačina koje opisuju funkciju prelaza i funkciju izlaza brojača. Takođe, kaskadno vezivanje brojača je opisano u nekoliko zadataka kao primer povezivanja modula standardnih sekvencijalnih mreža u okviru složenog digitalnog sistema.

REGISTRI:

Registri predstavljaju skup elementarnih automata i kombinacionih mreža koje omogućuju pamćenje reči i u opštem slučaju omogućuju izvršavanje sledećih elementarnih operacija:

- postavljanje registara na nulu;
- prijem reči iz drugog registra, kombinacione mreže ili brojača;
- prenos reči u drugi registar, kombinacionu mrežu ili brojač;
- pretvaranje direktnog koda u komplementarni kod i obrnuto;
- pomeraj reči u levo i desno za dati broj razreda;
- pretvaranje tipa serijsko/paralelno i paralelno/serijsko.

Logički simbol za registar prikazuje Slika 5.1.



Slika 5.1: Logički simbol registra

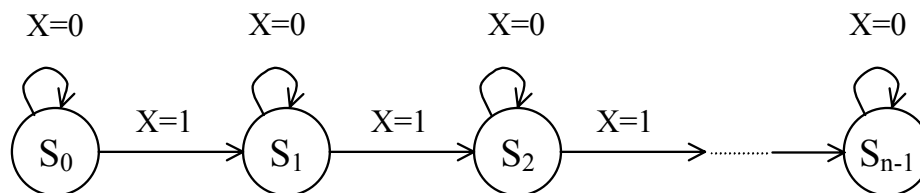
Uzima se da su ulaz i izlaz registra vektori dužine n bita, U i I respektivno. Izlaz registra odgovara njegovom stanju, a radi sinhronizacije koristi se signal takta koji se dovodi na ulaz T . Pored toga, radi upravljanja radom registra postoje i dva upravljačka ulaza. Prvi, U_p , se koristi za kontrolu upisa u registar, a drugi B za brisanje sadržaja registra. Funkcija prelaza (izlaza) registra može se opisati kao

$$I(t+1) = \begin{cases} U(t) & \text{za } U_p = 1 \\ I(t) & \text{u protivnom} \end{cases}$$

Upravljački ulaz B može biti sinhron ili asinhron i služi za unos specijalnog stanja $(0,0,\dots,0,0)$ u registar. Najčešće služi za postavljanje sadržaja na nulu na početku rada (inicijalizacija).

BROJAČI:

Brojači su sekvencijalne mreže sa jednim binarnim ulazom X (brojački impuls), čiji dijagram stanja predstavlja repetitivni ciklus. Broj različitih stanja u ciklusu se naziva modul ili osnova brojača (brojač modula N je brojač sa N stanja). Slika 5.2 prikazuje dijagram stanja brojača po modulu N .



Slika 5.2: Dijagram stanja brojača po modulu N

Ako se stanja označe celim brojevima $0, 1, \dots, N-1$, funkcija prelaza brojača može se analitički izraziti u obliku $S(t+1) = (S(t) + x) \bmod N$.

Kao memorijski elementi u brojačima se koriste flip-flopovi. Brojač od n flip-flopova je n -bitni binarni brojač ili binarni brojač modula $N = 2^n$, ako ima 2^n stanja koja se menjaju u sekvenci binarnih brojeva. Za realizaciju brojača sa N stanja potrebno je $m \geq \log_2 N$ memorijskih elemenata.

Kod većine brojačkih modula izlaz odgovara stanju, uz uobičajeni dodatni izlaz (npr. TSE) koji ima vrednost 1 kada se brojač nađe u stanju $N-1$. Taj dodatni izlaz naziva se zadnjom cifrom brojača i koristi se za kaskadnu vezu brojača.

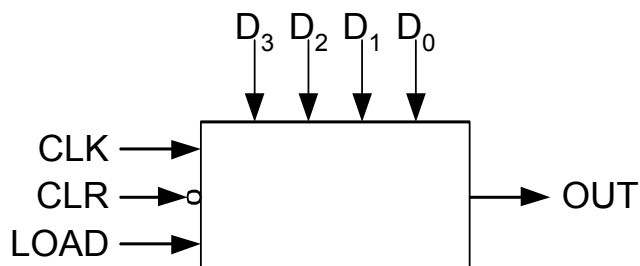
Brojači se kao komponente digitalnog sistema mogu realizovati kao:

- brojači nagore (koduju rastući niz binarnih brojeva),
- brojači nadole (koduju opadajući niz binarnih brojeva),
- obostrani brojači (zavisno od izabranog režima rada, mogu da koduju rastući ili opadajući niz binarnih brojeva).

5.1 ZADATAK:

Izvršiti sintezu četvorobitnog registra sa paralelnim ulazom i serijskim izlazom. Serijski izlaz realizovati sa bita najniže važnosti (LSB bit koji je označen indeksom 0). Sintezu izvršiti pomoću standardnih logičkih kola i D flip-flopova, kao i pomoću VHDL jezika za opis fizičke arhitekture.

Sa stanovišta spoljnih priključaka, ovakav registar prikazuje Slika 5.3.



Slika 5.3: Registar sa paralelnim ulazom i serijskim izlazom

REŠENJE:

Na osnovu slike prikazane u zadatku vidi se da registar pored 4 signala za paralelni upis vrednosti u registar (D_3 do D_0) ima i ulaz za signal takta (CLK), ulaz za postavljanje registra u početno stanje (CLR) koji je aktivan na niskom nivou kao i ulazni signal dozvole upisa u registar (LOAD) koji je aktivan na visokom nivou. Serijski izlaz registra je označen signalom (OUT).

Za fizičku realizaciju traženog registra potrebna su 4 flip-flopa, pri čemu će se svakom od njih pridružiti odgovarajuća kombinatorna mreža koje je kontrolisana signalom LOAD (signal dozvole paralelnog upisa). Neka se kombinatorna mreža ispred i -tog flip-flopa označava prenosnom funkcijom F_i . Funkciju F_i koja se dovodi na ulaz i -tog flip flopa, prikazuje Tabela 5.1.

LOAD	F_i
0	Q_{i+1}
1	D_i

Tabela 5.1: Tabela funkcije F_i

Dakle, funkcija F_i ima oblik $F_i = \overline{\text{LOAD}} \cdot Q_{i+1} + \text{LOAD} \cdot D_i$, pri čemu za $i=3$, Q_{i+1} ima vrednost 0, odnosno da se prilikom pomeranja sa leve strane upisuje nula.

Primeru radi, prikazaće se stanja flip-flopova u registru kada je nakon paralelnog upisa sadržaja $D_3D_2D_1D_0=1101$, signal LOAD postavljen na nulu (nije aktivan), a zatim su generisana 4 takt impulsa CLK (pri tom se smatra da je signal CLR sve vreme neaktivan, odn. $\text{CLR}=1$).

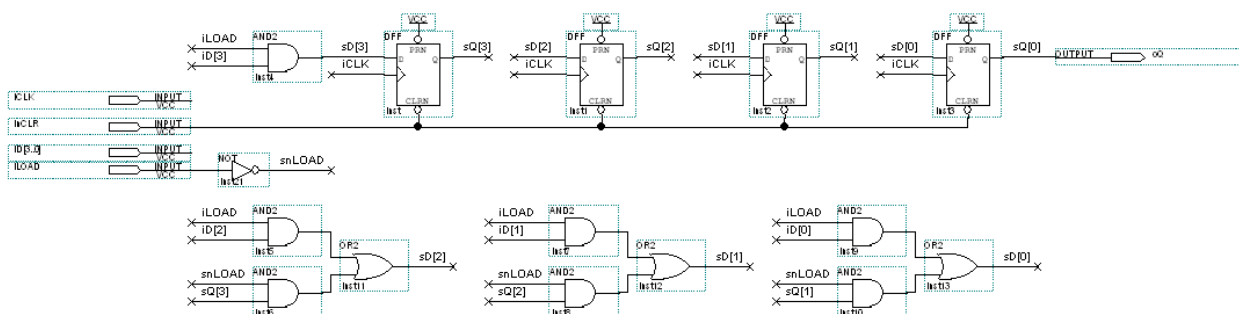
Jedna od najčešćih primena ovakvih registara jeste konverzija sadržaja registra u serijski niz digitalnih podataka radi serijskog prenosa podataka.

LOAD	Sadašnje stanje				Informacioni ulazi				CLK	Sledeće stanje				OUT Q ₀
	Q ₃	Q ₂	Q ₁	Q ₀	D ₃	D ₂	D ₁	D ₀		Q ₃	Q ₂	Q ₁	Q ₀	
1	×	×	×	×	1	1	0	1	⌋	1	1	0	1	1
0	1	1	0	1	×	×	×	×	⌋	0	1	1	0	0
0	0	1	1	0	×	×	×	×	⌋	0	0	1	1	1
0	0	0	1	1	×	×	×	×	⌋	0	0	0	1	1
0	0	0	0	1	×	×	×	×	⌋	0	0	0	0	0

Tabela 5.2: Tabela stanja pomeračkog registra

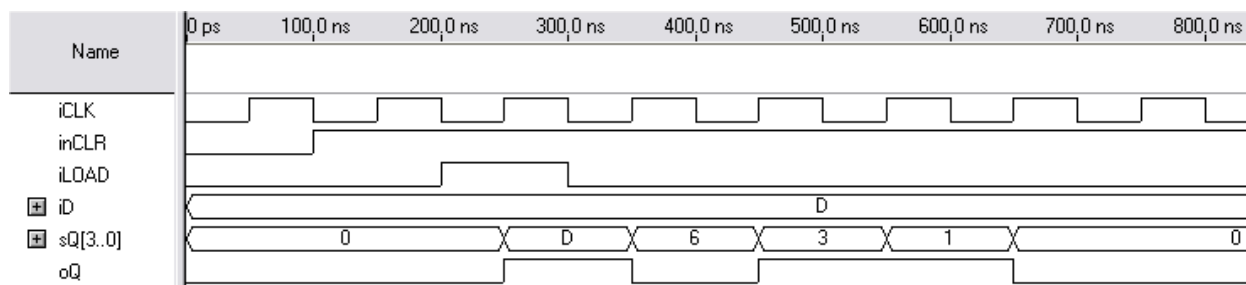
Ukoliko se pomerački registar sa mogućnošću paralelnog upisa i serijskim izlazom nalazi u digitalnom uređaju koji šalje podatke, a sa prijemne strane je registar sa serijskim ulazom i paralelnim izlazom, moguće je informaciju od n bita preneti na fizički udaljeno mesto korišćenjem samo dve električne veze - informacione linije (prenos signala OUT) i linije takta (prenos signala CLK).

Svi signali na logičkoj šemi, Slika 5.4, označeni su u skladu sa prethodnim tekstom.



Slika 5.4: Logička šema registra sa paralelnim ulazom i serijskim izlazom

Slika 5.5 prikazuje simulaciju pomeračkog registra sa paralelnim ulazom čija je električna šema data na prethodnoj slici. Prikazani vremenski dijagram odgovara primeru koji prikazuje Tabela 5.2.



Slika 5.5: Simulacija rada registra

Prilikom sinteze ovog registra u VHDL-u u okviru entiteta REGISTRAR definišaće se ulazni i izlazni signali registra, i to:

- iCLK - ulazni signal takta (registar je aktivan na uzlaznu ivicu takta);
- inCLR- ulazni signal koji u registar upisuje sve nule, tj. inicijalizacija registra (aktivan je u stanju logičke nule);
- iLOAD- ulazni signal koji u registar upisuje sadržaj ulaza podataka D (aktivan je takođe u stanju logičke jedinice);
- iD - četvorobitni ulazni signal podataka koji se upisuju u registar kada je signal load u aktivnom stanju i
- oQ - izlazni jednobitni signal.

U okviru arhitekture ARH_REGISTAR uvešće se signal sREG koji predstavlja sadržaj četvorobitnog registra.

Potrebno je da se u listi osetljivosti procesa koji opisuje ponašanje željenog pomeračkog registra navedu signali iCLK i inCLR, budući da je funkcija inicijalizacije registra na logičkoj šemi realizovana kao asinhrona. Signale iLOAD i iD ne treba navoditi, pošto je funkcija upisa sadržaja sinhrona.

VHDL kod registra opisanog u ovom zadatku ima sledeći oblik:

```

LIBRARY ieee;
USE ieee.std_logic_1164.all;

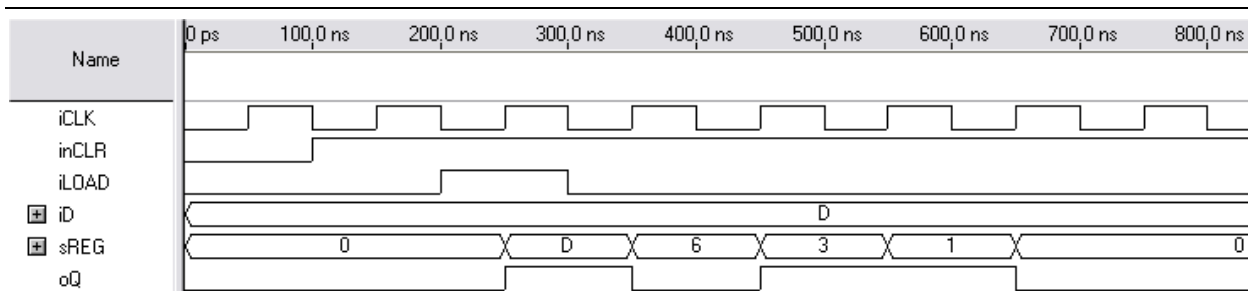
ENTITY REGISTAR IS PORT(
    iCLK, inCLR,
    iLOAD: IN STD_LOGIC;
    iD: IN STD_LOGIC_VECTOR(3 DOWNTO 0);
    oQ: OUT STD_LOGIC);
END REGISTAR;

ARCHITECTURE ARH_REGISTAR OF REGISTAR IS
    -- stanje cetvorobitnog registra
    SIGNAL sREG: STD_LOGIC_VECTOR(3 DOWNTO 0);
BEGIN
    PROCESS (iCLK, inCLR) BEGIN
        IF (inCLR = '0') THEN
            sREG <= "0000"; -- postavljanje pocetnog stanja, asinhrono
        ELSIF (iCLK'EVENT AND iCLK = '1') THEN
            IF (iLOAD = '1') THEN
                sREG <= iD; -- paralelni upis
            ELSE
                -- pomeranje sadrzaja registra prema LSB bitu
                sREG <= ('0' & sREG(3) & sREG(2) & sREG(1));
            END IF;
        END IF;
    END PROCESS;

    -- izlazni signal, LSB bit izlazi prvi
    oQ <= sREG(0);

END ARH_REGISTAR;
```

Simulaciju prethodno navedenog VHDL koda prikazuje Slika 5.6. Takođe, i u ovom slučaju prikazani vremenski dijagram odgovara primeru koji prikazuje Tabela 5.2.



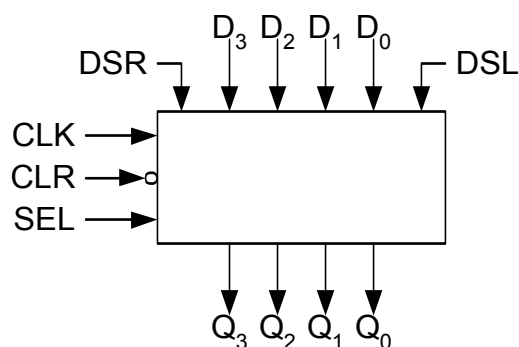
Slika 5.6: Simulacija rada registra

5.2 ZADATAK:

Izvršiti sintezu pomeračkog registra koji realizuju sledeće operacije:

S ₁	S ₀	Operacija
0	0	paralelni unos
0	1	pomeranje udesno →
1	0	pomeranje ulevo ←
1	1	ništa ne radi

Tabela 5.3: Funkcije pomeračkog registra



Slika 5.7: Blok šema registra

Blok šemu registra prikazuje Slika 5.7. Operaciju postavljanja inicijalne vrednosti registra realizovati sinhrono sa signalom takta.

Prilikom pomeranja ulevo, na LSB poziciju registra treba upisati vrednost koja se nalazi na ulaznom priključku DSL. Slično, kod pomeranja u desno na MSB poziciju registra treba upisati vrednost koja se nalazi na ulaznom priključku DSR.

REŠENJE:

Na osnovu operacija koje treba realizovati može se formirati tabela koja prikazuje promene stanja registra, za svaku operaciju posebno, Tabela 5.4.

S ₁	S ₀	Q ₃	Q ₂	Q ₁	Q ₀	Vrsta pomeranja
0	0	D ₃	D ₂	D ₁	D ₀	paralelni unos
1	0	DSR	Q ₃	Q ₂	Q ₁	pomeranje udesno
0	1	Q ₂	Q ₁	Q ₀	DSL	pomeranje ulevo
1	1	Q ₃	Q ₂	Q ₁	Q ₀	ništa ne radi

Tabela 5.4: Tabela operacija pomeranja

VHDL sinteza traženog pomerača sledi direktno iz tabele operacija pomeranja. Traženi pomerač se može opisati pomoću jednog procesa koji opisuje

sekvencijalnu mrežu koja na osnovu vrednosti adresnih ulaza (CASE iskaz) realizuje potrebno pomeranje. Prilikom sinteze ovog registra u VHDL-u u okviru entiteta POMERACKI_REG definišu se sledeći ulazni i izlazni signali registra:

- iCLK - ulazni signal takta (registar je aktivan na uzlaznu ivicu takta);
- inCLR- ulazni signal koji u registar upisuje sve nule, tj. inicijalizacija registra (aktivan je u stanju logičke nule);
- iDSL - vrednost koja se upisuje u LSB bit registra prilikom pomeranja ulevo
- iDSR - vrednost koja se upisuje u MSB bit registra prilikom pomeranja ulevo
- iSEL - adresni ulazi za odabiranje vrste operacije (vektor od dva bita);
- iD - četvorobitni ulazni signal podataka koji se upisuju u registar kada je adresirana operacija upisa u registar i
- oQ - trenutno stanje registra (četvorobitni vektor).

```

LIBRARY ieee;
USE ieee.std_logic_1164.all;

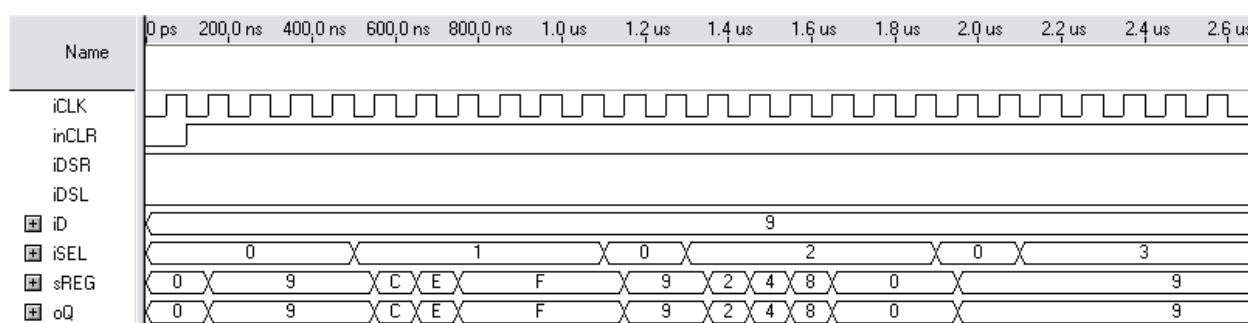
ENTITY POMERACKI_REG IS PORT(
    iCLK, inCLR: IN STD_LOGIC;
    iDSL, iDSR: IN STD_LOGIC;
    iSEL: IN STD_LOGIC_VECTOR(1 DOWNTO 0);
    iD: IN STD_LOGIC_VECTOR(3 DOWNTO 0);
    oQ: OUT STD_LOGIC_VECTOR(3 DOWNTO 0));
END POMERACKI_REG;

ARCHITECTURE ARH_POMERACKI_REG OF POMERACKI_REG IS
    -- stanje registra
    SIGNAL sREG: STD_LOGIC_VECTOR(3 DOWNTO 0);
BEGIN
    PROCESS (iCLK) BEGIN
        IF (iCLK'EVENT AND iCLK = '1') THEN
            -- sinhrono postavljanje pocetne vrednosti
            IF (inCLR = '0') THEN
                sREG <= "0000";
            ELSE
                CASE iSEL IS
                    -- upis u registar
                    WHEN "00" => sREG <= iD;
                    -- pomeranje u desno
                    WHEN "01" => sREG <= (iDSR & sREG(3) & sREG(2) & sREG(1));
                    -- pomeranje u levo
                    WHEN "10" => sREG <= (sREG(2) & sREG(1) & sREG(0) & iDSL);
                    -- zadržavanje stanja
                    WHEN OTHERS => sREG <= sREG;
                END CASE;
            END IF;
        END IF;
    END PROCESS;

    -- preslikavanje stanja registra na izlazni vektor
    oQ <= sREG;
END ARH_POMERACKI_REG;

```

Slika 5.8 prikazuje simulaciju rada realizovanog registra. Na početku vremenskog dijagrama registar se postavi u početno stanje ($inCLR=0$). Nakon toga se pomoću adresnog ulaza $iSEL=0$ u registar upiše stanje na ulaznom vektoru $iD=9$. U vremenskom trenutku 500ns adresni ulaz menja vrednost na $iSEL=1$. Time počinje pomeranje sadržaja registra u desno, sa upisom vrednosti $iDSR=1$ na MSB bit registra. Na kraju pomeranja sadržaj registra je $sREG=F$. U vremenskom trenutku 1,2 μ s se ponovo upisuje vrednost 9 u registar, $iSEL=0$, da bi se nakon 200ns započela operacija pomeranja sadržaja registra u levo, $iSEL=2$. Tokom ove operacije u LSB bit registra se upisuje vrednost $iDSL=0$, da bi na kraju sadržaj registra bio $sREG=0$. Na kraju vremenskog dijagrama se ilustruje zadržavanje upisane vrednosti u registru postavljanjem adresnog ulaza na vrednost $iSEL=3$.



Slika 5.8: Simulacija rada realizovanog registra

5.3 ZADATAK:

Projektovati brojač po modulu 5 sa zaštitom od ulaska brojača u nedefinisana stanja. Brojač projektovati za realizaciju sa JK flip-flopovima (Tabela 5.5) i sa standardnim logičkim kolima. Omogućiti postavljanje brojača u početno stanje ulaznim signalom koji je aktivan na niskom nivou.

Isprojektovati ekvivalentan brojač u VHDL jeziku za opis fizičke arhitekture.

Q(t)	Q(t+1)	J	K
0	0	0	×
0	1	1	×
1	0	×	1
1	1	×	0

Tabela 5.5: Tabela prelaza JK flip-flopova

REŠENJE:

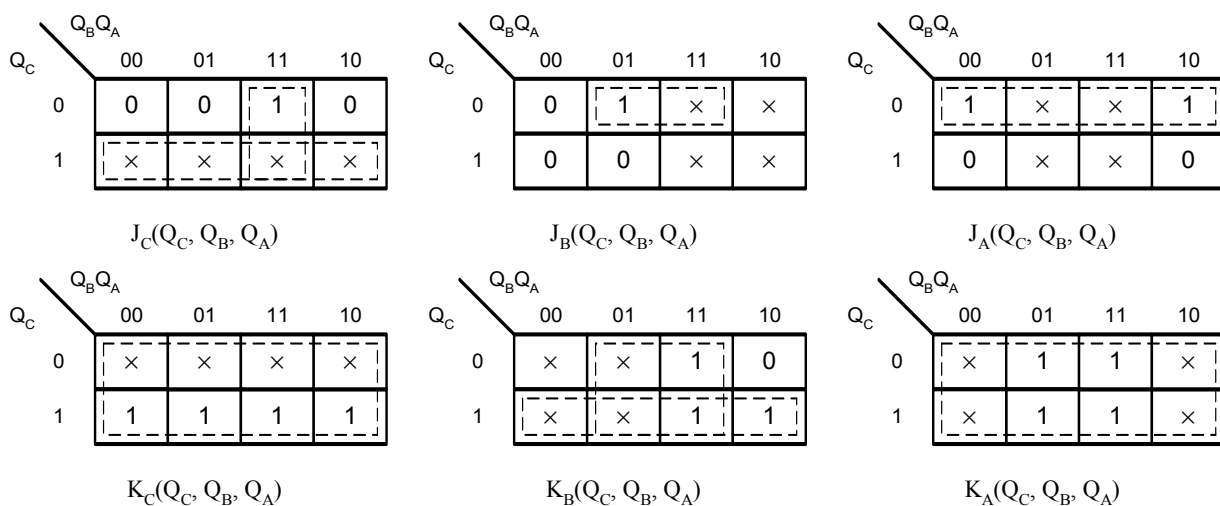
Da bi se realizovao traženi brojač, neophodno je odrediti broj memorijskih elemenata koji su neophodni za realizaciju zadatog brojača, a to je $n=3$ ($2^3=8 \geq 5$). Kako je najveći broj koji brojač realizovan sa 3 memorijska elementa može da dostigne jednak $2^3-1=7$, sledi da će postojati tri nedefinisana stanja: 5, 6 i 7. Zaštita od ulaska u ova stanja će se realizovati na taj način da ukoliko se brojač nađe u stanjima 5, 6 ili 7, automatski postaviti u početno stanje 0.

Kako su na raspolaganju JK flip-flopovi, uzimajući u obzir njihovu tabelu prelaza (Tabela 5.5), moguća stanja brojača i opisanu zaštitu od ulaska u nedefinisana stanja, može se formirati sledeća kombinaciona tablica brojača (Tabela 5.6).

Trenutno stanje				Naredno stanje			Ulazi flip-flopora					
S_i	Q_C	Q_B	Q_A	Q_C	Q_B	Q_A	J_C	K_C	J_B	K_B	J_A	K_A
0	0	0	0	0	0	1	0	×	0	×	1	×
1	0	0	1	0	1	0	0	×	1	×	×	1
2	0	1	0	0	1	1	0	×	×	0	1	×
3	0	1	1	1	0	0	1	×	×	1	×	1
4	1	0	0	0	0	0	×	1	0	×	0	×
5	1	0	1	0	0	0	×	1	0	×	×	1
6	1	1	0	0	0	0	×	1	×	1	0	×
7	1	1	1	0	0	0	×	1	×	1	×	1

Tabela 5.6: Kombinaciona tabela brojača

Na osnovu dobijene tabele, moguće je, primenjujući metode minimizacije Karnoovim mapama, odrediti ulazne promenljive korišćenih memorijskih elemenata (JK flip-flopora), Slika 5.9.



Slika 5.9: Karnoove mape

Minimizacijom se dobijaju sledeće funkcije:

$$J_C = Q_B \cdot Q_A$$

$$K_C = 1$$

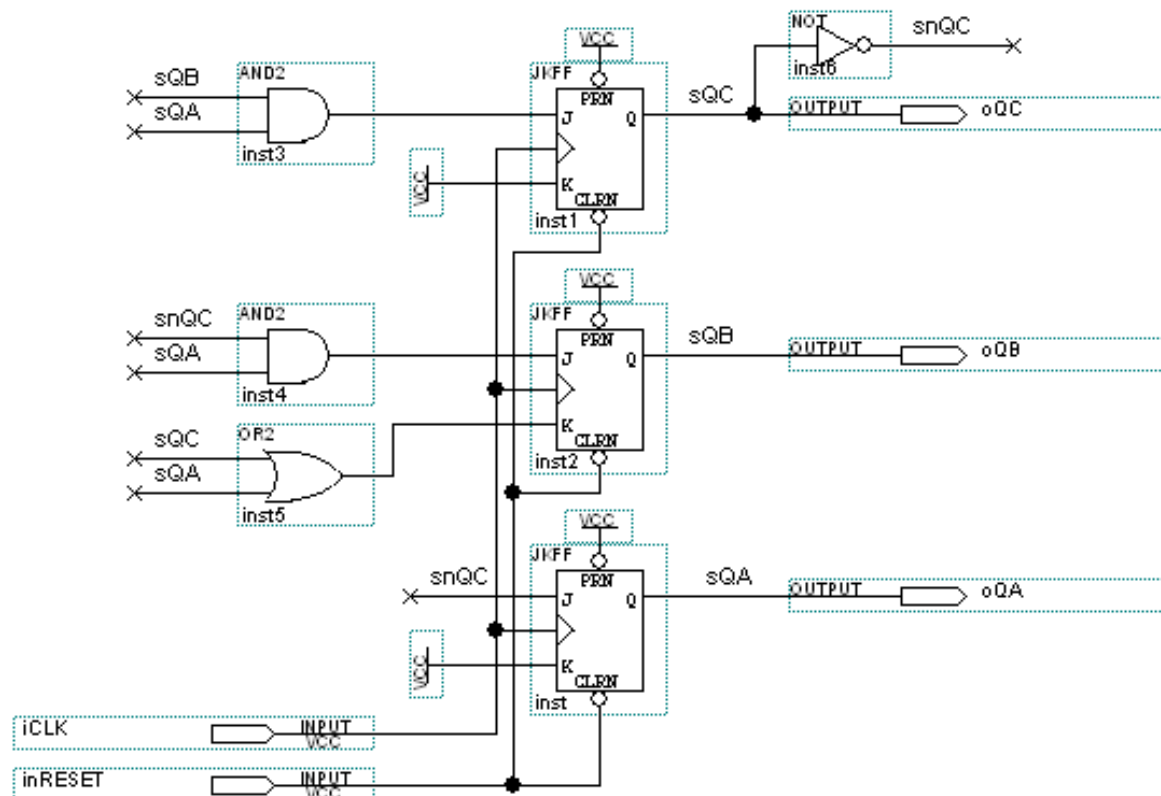
$$J_B = \overline{Q_C} \cdot Q_A$$

$$K_B = Q_A + Q_C$$

$$J_A = \overline{Q_C}$$

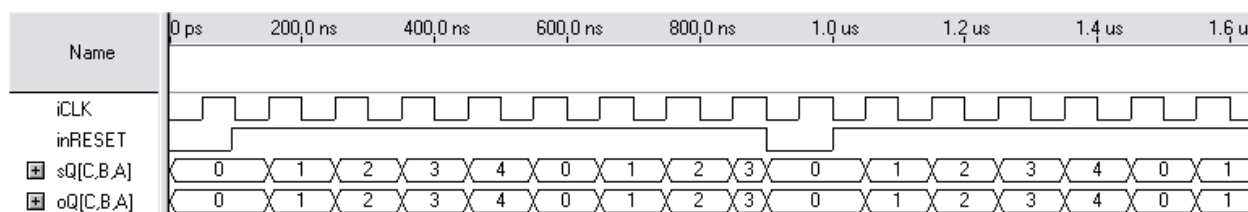
$$K_A = 1$$

Ovim je realizovan postupak sinteze brojača modula 5. Na osnovu dobijenih jednačina se pristupa formiranju logičke šeme brojača, Slika 5.10.



Slika 5.10: Logička šema brojača modula 5 sa zaštitom od ulazaka u nedefinisano stanje

Slika 5.11 prikazuje simulaciju rada realizovanog brojača. Na vremenskom dijagramu u vremenskom trenutku 900ns se vidi da je brojač vraćen u početno stanje aktiviranjem ulaznog signala inRESET. Ovo se desilo odmah po postavljanju signala inRESET na nizak logički nivo, bez obzira na signal takta iCLK. Ovakva realizacija inicijalizacije stanja brojača se zove asinhrona.



Slika 5.11: Simulacija rada realizovanog brojača

Ekvivalentan brojač sa stanovišta ulazno/izlaznih signala i prelazaka stanja se može na sledeći način realizovati pomoću VHDL jezika za opis fizičke arhitekture.

```

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;

ENTITY BROJAC_M5 IS PORT(
    iCLK,
    inRESET: IN STD_LOGIC;
    oCNT: OUT INTEGER RANGE 0 TO 4 );
END BROJAC_M5;

ARCHITECTURE ARH_BROJAC_M5 OF BROJAC_M5 IS
    CONSTANT cHIGH: INTEGER := 4; -- kraj ciklusa brojanja (modulo-1)
    SIGNAL sCNT: INTEGER RANGE 0 TO 4; -- stanje brojaca
BEGIN

    PROCESS(iCLK) BEGIN
        IF(iCLK'EVENT AND iCLK = '1') THEN
            IF (inRESET = '0') THEN -- sinhroni reset
                sCNT <= 0;
            ELSE
                IF (sCNT >= cHIGH) THEN
                    sCNT <= 0; -- kraj ciklusa brojanja->resetuj brojac
                ELSE
                    sCNT <= sCNT + 1; -- brojanje
                END IF;
            END IF;
        END IF;
    END PROCESS;

    oCNT <= sCNT; -- preslikavanje stanja brojaca na izlazni vektor

END ARH_BROJAC_M5;

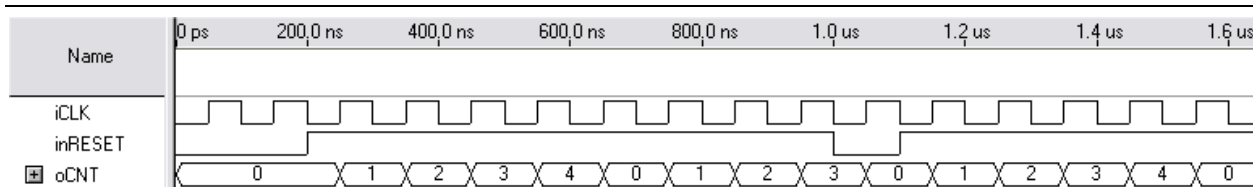
```

Izlazni signal koji prikazuje stanje brojača, oCNT, je tipa INTEGER. Njegov opseg je ograničen na 5 vrednosti (0 do 4), radi formiranja brojača modula 5. Na osnovu ovih informacija VHDL prevodilac će za realizaciju brojača rezervirati tri memorijska elementa ($5 \leq 2^3 = 8$).

U zaglavlju arhitekture je definisana konstanta cHIGH koja sadrži vrednost poslednjeg stanja brojača, tj. cHIGH=4. Ova konstanta se kasnije u kodu koristi za realizaciju brojača zadatog modula, tako što realizovani brojač broji od 0 do cHIGH-1 što u ovom slučaju čini ukupno 5 stanja (brojač modula 5). Takođe, u zaglavlju arhitekture je definisan signal sCNT koji će poslužiti za realizaciju brojača.

Brojač je realizovan sa jednim procesom, sa sinhronim postavljanjem u početno stanje. U slučaju da signal za inicijalizaciju nije aktivan (inRESET≠0) vrši se brojanje u skladu sa dijagramom prelaza stanja brojača. Iskazom IF (sCNT=cHIGH) se vrši provera da li je brojač stigao do kraja ciklusa (do vrednosti 4). Ako jeste, stanje brojača se vraća nazad u početno stanje, a u suprotnom se njegova vrednost uvećava za 1.

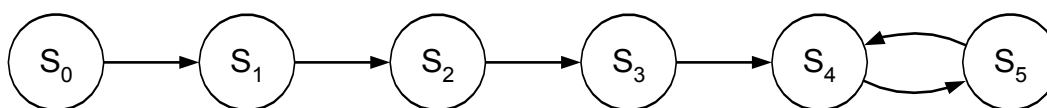
Vremenski dijagram simulacije rada realizovanog brojača prikazuje Slika 5.12.



Slika 5.12: Simulacija rada realizovanog brojača

5.4 ZADATAK:

Projektovati brojač sa 6 stanja (S_i ; $i=0,1, 2, 3, 4, 5$), pomoću D flip-flopa i NI kola. Brojač treba da broji prema pravilu koje prikazuje Slika 5.13.



Slika 5.13: Dijagram stanja traženog brojača

REŠENJE:

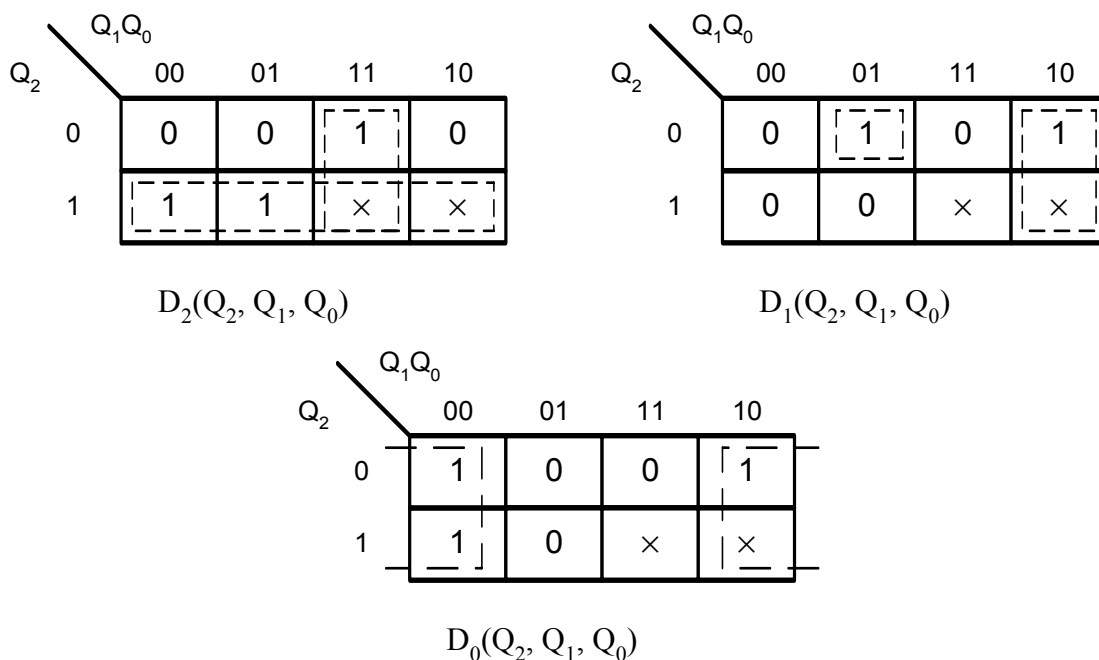
Da bi se realizovao traženi brojač, neophodno je odrediti broj memorijskih elemenata koji su neophodni za realizaciju zadatog brojača. Za predstavljanje ukupno šest stanja brojača potrebna su 3 memorijska elementa (flip-flopa) jer je $2^3=8 \geq 6$.

Na osnovu dobijenog dijagrama prelaza stanja brojača formira se tabela prelaza stanja brojača, Tabela 5.7.

Trenutno stanje brojača (izlazi flip-flopa)			Naredno stanje brojača (ulazi u flip-flobove)		
Q_2	Q_1	Q_0	D_2	D_1	D_0
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	0	0

Tabela 5.7: Tabela prelaza stanja brojača

Sada je moguće pomoću Karnoovih mapa izvršiti minimizaciju ovog automata. Karnoove karte prikazuje Slika 5.14.



Slika 5.14: Karnoove mape

Minimizacijom pomoću Karnoovih mapa su dobijene sledeće funkcije:

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

$$D_1 = Q_1 \cdot \overline{Q_0} + \overline{Q_2} \overline{Q_1} \cdot Q_0$$

$$D_0 = \overline{Q_0}$$

Ove jednačine je potrebno pomoću De Morganog zakona prevesti u formu koja odgovara realizaciji pomoću NI kola:

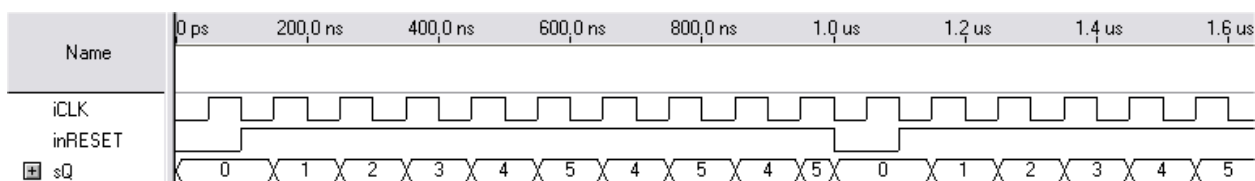
$$D_2 = Q_2 + Q_1 \cdot Q_0 = \overline{\overline{Q_2} \cdot \overline{Q_1} \cdot \overline{Q_0}}$$

$$D_1 = Q_1 \cdot \overline{Q_0} + \overline{Q_2} \overline{Q_1} \cdot Q_0 = \overline{\overline{Q_1} \cdot Q_0 + \overline{\overline{Q_2} \cdot \overline{Q_1}} \cdot \overline{Q_0}}$$

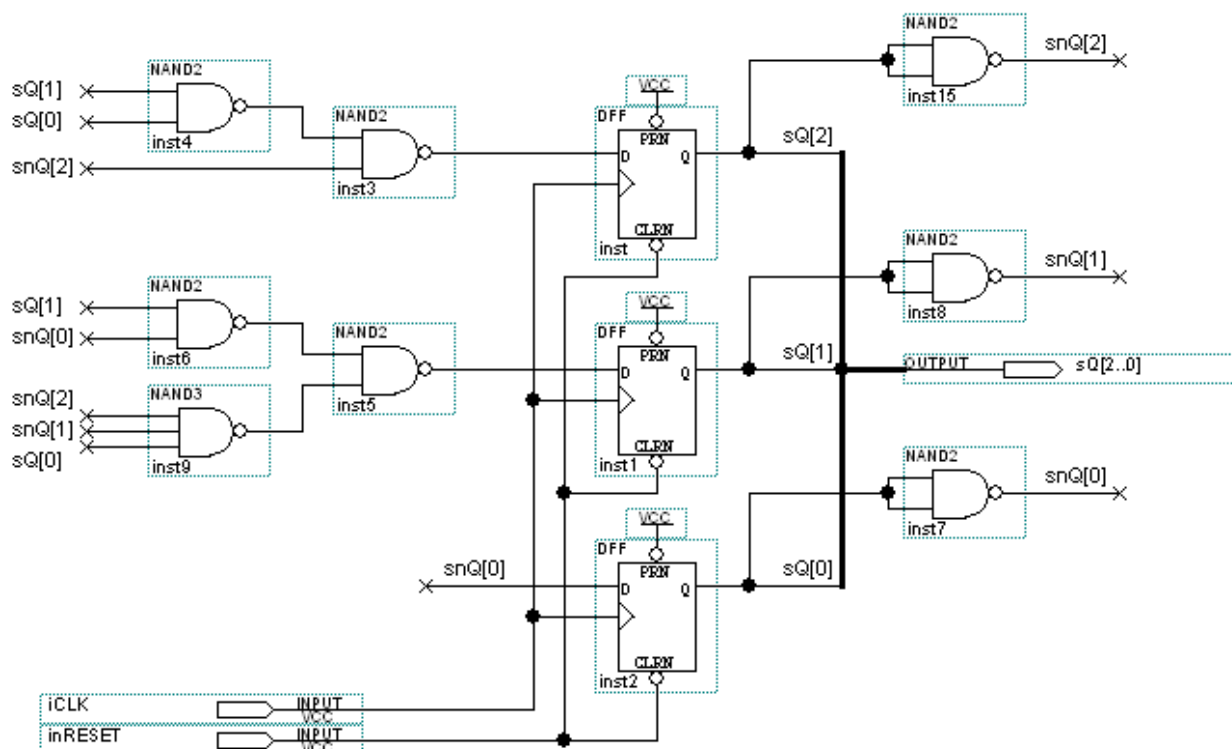
$$D_0 = \overline{Q_0}$$

Na osnovu dobijenih jednačina pristupa se sintezi traženog brojača tako što se formira odgovarajuća šema logičkih elemenata, Slika 5.16.

Slika 5.15 prikazuje simulaciju rada realizovanog brojača. Na vremenskom dijagramu se vidi asinhrona priroda postavljanja brojača u početno stanje.



Slika 5.15: Simulacija rada realizovanog brojača



Slika 5.16: Logička šema realizovanog brojača

Brojač čiji dijagram prelaza stanja prikazuje Slika 5.13 moguće je realizovati i u VHDL-u. Entitet brojača sadrži dva ulazna signala: takt signal (iCLK) i signal za postavljanje početne vrednosti brojača (inRESET), i izlazni signal koji prikazuje stanje brojača (oQ).

Jedan od načina realizacije brojača u VHDL jeziku za opis fizičke arhitekture je prikazan u nastavku:

```

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;

ENTITY BROJAC_M6 IS
    PORT (
        iCLK:    IN  STD_LOGIC;
        inRESET: IN  STD_LOGIC;
        oQ:      OUT INTEGER RANGE 0 TO 5);
END BROJAC_M6;

ARCHITECTURE ARH_BROJAC_M6 OF BROJAC_M6 IS BEGIN
    PROCESS (iCLK)
        CONSTANT cHIGH: INTEGER := 5; -- kraj ciklusa brojanja (modulo-1)
        VARIABLE vCNT:  INTEGER RANGE 0 TO 5; -- stanje brojaca
    BEGIN
        IF (iCLK'EVENT AND iCLK='1') THEN
            IF (inRESET = '0') THEN -- sinhroni reset
                vCNT := 0;
            ELSE
                IF (vCNT = cHIGH) THEN
                    vCNT := 4; -- kraj ciklusa brojanja->resetuj brojac
                END IF;
            END IF;
        END IF;
    END PROCESS;

```



```

ELSE
    vCNT := vCNT + 1;  -- broj anje
END IF;
END IF;
END IF;

oQ <= vCNT; -- preslikavanje stanja brojača na izlazni vektor
END PROCESS;
END ARH_BROJAC_M6;

```

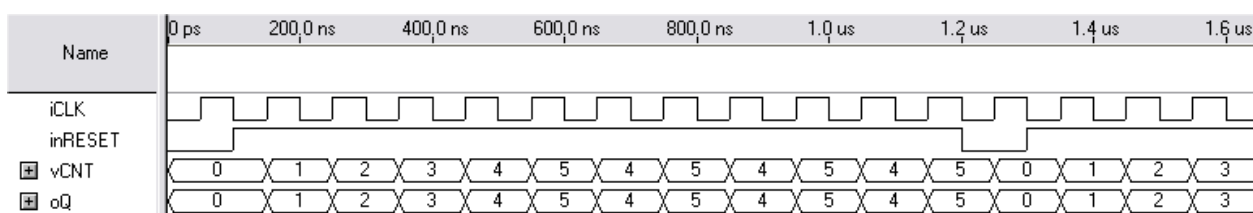
Izlazni signal koji prikazuje stanje brojača je tipa INTEGER. Njegov opseg je ograničen na 6 vrednosti (0 do 5). Na osnovu ovih informacija, VHDL prevodilac će za realizaciju brojača rezervirati tri memorijska elementa.

Brojač je realizovan sa jednim procesom. U zaglavlju procesa je definisana konstanta cHIGH koja sadrži vrednost poslednjeg stanja brojača, tj. cHIGH=5. Takođe, u zaglavlju procesa je definisana promenljiva vCNT. Ova promenljiva, kao i prethodno definisana konstanta, ima domen samo u procesu u kome je realizovana. To znači da se ona ne može koristiti nigde izvan procesa. Time se postiže određeni stepen lokalizacije važenja određenih identifikatora, te se isti takvi identifikatori mogu koristiti i na drugim mestima koji nisu u dodiru sa ovim procesom.

Važno je napomenuti da sinteza promenljivih u VHDL-u nije dobro definisana. Zbog toga se one koriste samo za potebe izračunavanja i to u slučajevima gde se sa sigurnošću može predvideti rezultat, što je ovde slučaj.

Brojač je realizovan sa sinhronim postavljanjem u početno stanje. U slučaju da signal za inicijalizaciju nije aktivan (inRESET≠0) vrši se brojanje u skladu sa dijagramom prelaza stanja brojača. Iskazom IF (vCNT=cHIGH) se vrši provera da li je brojač stigao do kraja ciklusa (do vrednosti 5). Ako jeste, brojač menja stanja na 4 (vCNT:=4), a u suprotnom se uvećava njegova vrednost za 1. Na kraju procesa se vrednost promenljive vCNT preslikava na izlazni vektor oQ.

Vremenski dijagram rada realizovanog brojača prikazuje Slika 5.17. Na slici je interesantan vremenski interval od 1,2μs do 1,3μs. U vremenskom trenutku 1,2μs aktiviran je signal inRESET za postavljanje početne vrednosti brojača. međutim inicijalizacija istog je suspendovana do prve rastuće ivice takt signala iCLK gde je aktivan signal inRESET (1,25μs). Tek tada se stanje brojača menja u inicijalnu vrednost. Takav način postavljanja početne vrednosti brojača se naziva sinhron.



Slika 5.17: Vremenski dijagram realizovanog brojača

5.5 ZADATAK:

U VHDL jeziku za opis fizičke arhitekture izvršiti sintezu brojača po modulu 8 sa mogućnošću paralelnog upisa i brojanja gore-dole (paralelni brojač), Tabela 5.8 prikazuje tabelu prelaza stanja brojača.

CLK	CLR	LOAD	U/D	FUNKCIJA BROJAČA
$\overline{\downarrow}$	0	\times	\times	$Q \leftarrow 0$
$\overline{\downarrow}$	1	0	\times	$Q_{n+1} \leftarrow \text{DATA}$
$\overline{\downarrow}$	1	1	0	$Q_{n+1} \leftarrow Q_n - 1$
$\overline{\downarrow}$	1	1	1	$Q_{n+1} \leftarrow Q_n + 1$

Tabela 5.8: Tablica prelaska stanja brojača

REŠENJE:

Signali koje prikazuje Tabela 5.8 su poredani po opadajućem prioritetu, odn. najveći prioritet ima signal takta CLK. Treba primetiti da je aktivna opadajuća ivica takt signala. Sledeći po prioritetu je signal CLR čija je namena postavljanje brojača u stanje 0 (reset). Pošto je signal takta većeg prioriteta od signala reseta, sledi da treba realizovati sinhroni reset. Ako je aktivan signal reseta, tada bez obzira na vrednost ostalih signala (oznaka \times = "bilo koja vrednost") brojač mora biti resetovan. Signal sledećeg prioriteta je signal dozvole upisa nove vrednosti u brojač, LOAD. Kao i signal reseta i on je aktivan na niskom nivou. Najnižeg prioriteta je signal koji određuje smer brojanja, signal U/D. Ako su prethodna dva signala nekativna, tj. na visokom nivou, tada se vrši brojanje na dole ako je U/D jednak nuli ili brojanje na gore u suprotnom slučaju (U/D=1).

Ovaj redosled prioriteta se u VHDL-u realizuje ugnježđenim uslovnim iskazima. Prvi iskaz proverava da li je aktivna opadajuća ivica takt signala. Sledeći proverava stanje reset linije i tako redom. Odgovarajući VHDL kod sledi u nastavku.

```

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;

ENTITY BROJAC_M8 IS
  PORT (
    iCLK, inCLR,
    inLOAD, iUP_DOWN: IN STD_LOGIC;
    iD: IN      UNSIGNED(2 DOWNT0 0);
    oQ: BUFFER UNSIGNED(2 DOWNT0 0) );
END BROJAC_M8;

ARCHITECTURE ARH_BROJAC_M8 OF BROJAC_M8 IS
  BEGIN

```

```

PROCESS (iCLK) BEGIN
  IF (iCLK'EVENT AND iCLK = '0') THEN -- opadajuca ivica takt sinla
    IF (inCLR = '0') THEN -- sinhroni reset
      oQ <= "000";
    ELSE
      IF (inLOAD = '0') THEN
        oQ <= iD; -- dozvoljen paralelni upis
      ELSE
        IF (iUP_DOWN = '0') THEN
          oQ <= oQ - 1; -- brojanje na dolje
        ELSE
          oQ <= oQ + 1; -- brojanje na gore
        END IF;
      END IF;
    END IF;
  END IF;
END IF;
END PROCESS;

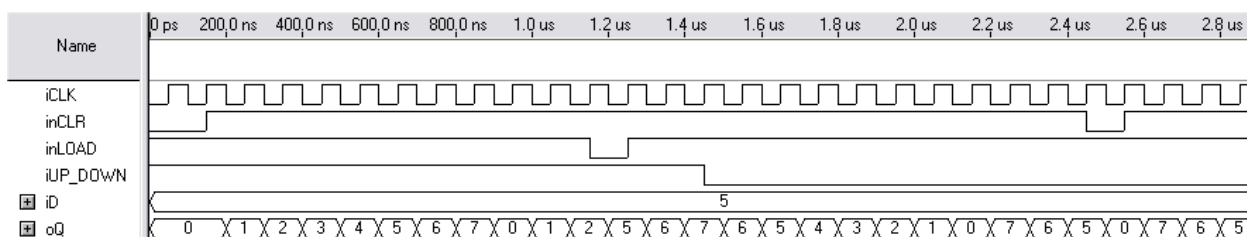
END ARH_BROJAC_M8;

```

U prethodnom VHDL opisu brojača korićen je tip UNSIGNED. Za razliku od tipa INTEGER gde se zadaje opseg vrednosti brojača, kod tipa UNSIGNED se direktno zadaje broj memorijskih elemenata za realizaciju brojača. Druga razlika između ova dva tipa je ta što ako je realizovani brojač tipa INTEGER nije moguće pristupiti određenom bitu brojača. Ovo je moguće kod realizacije brojača tipa UNSIGNED (slične situacije su korišćene kod realizacije ALJ).

Druga razlika između ovog i prethodnih VHDL opisa brojača je u tome što u ovom slučaju nije korišćen ni namenski signal za opis brojača kao ni promenljiva. Izlazni signal brojača, oQ, je u ovom slučaju realizovan u BUFFER režimu. Time je omogućeno korišćenje literala oQ i sa desne strane znaka dodele vrednosti signalu (npr. $oQ \leq oQ - 1$), što ne bi bilo moguće da je izlazni signal realizovan u OUT režimu kao u prethodnim slučajevima.

Slika 5.18 prikazuje vremenski dijagram rada realizovanog brojača.



Slika 5.18: Simulacija rada realizovanog brojača

Nakon početnog postavljanja inicijalne vrednosti brojača ($inCLR=0$), brojač je iskonfigurisan da broji na gore. To je postignuto postavljanjem signala $inCLR$, $inLOAD$ i iUP_DOWN na visoki logički nivo. U vremenskom trenutku $1,2\mu s$ brojač učitava vrednost koja se nalazi na ulaznom vektoru $iD=5$, pošto je ulazni signal dozovole upisa u brojač aktivan, $inLOAD=0$, i naišla je opadajuća ivica takt signala, $iCLK$. Nakon 300ns od prethodnog trenutka brojač počinje da broji na

dole pošto je neposredno pre ove aktivne ivice takt signala, ulazni signal uUP_DOWN promenio vrednost na nulu. Postavljanje početne vrednosti brojača je ponovo prikazano u vremenskom trenutku $2,5\mu s$.

5.6 ZADATAK:

- a). Izvršiti sintezu četvorobitnog brojača nagore modula 16 sa mogućnošću paralelnog upisa i kontrole brojanja pomoću D flip-flopa (prenosnu funkciju prikazuje Tabela 5.9) i standardnih logičkih kola.

E	D	$Q(t+1)$
0	\times	$Q(t)$
1	0	0
1	1	1

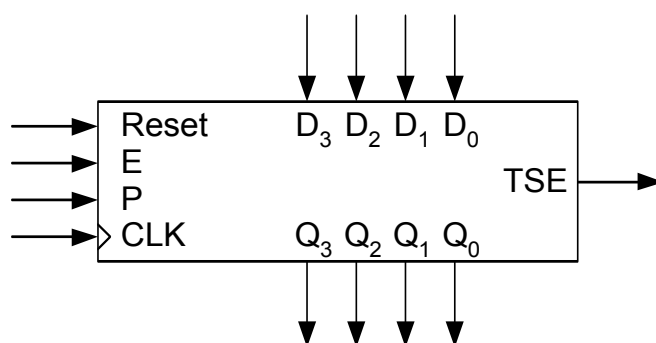
Tabela 5.9: Prenosna funkcija D-flip-flopa

Obezbediti izlazni signal TSE koji ukazuje da je brojač na kraju ciklusa brojanja F_{Hex} . Ovaj izlazni signal formirati pomoću kombinacione mreže, bez baferovanja signala na izlazu. Tabela 5.10 prikazuje prenosnu funkciju brojača. Takođe, izvršiti sintezu ekvivalentnog brojača u VHDL jeziku za opis fizičke arhitekture.

Reset	CLK	E	P	$Q(t+1)$	TSE
0	\times	\times	\times	0	0
1	\downarrow	0	\times	$Q(t)$	0
1	\uparrow	1	1	$D(t)$	1 za $Q(t)=F_{HEX}$
1	\uparrow	1	0	$Q(t)+1$	1 za $Q(t)=F_{HEX}$

Tabela 5.10: Prenosna funkcija brojača

- b). Skicirati blok šemu brojača modula 138 povezivanjem potrebnog broja realizovanih brojača modula 16. Na osnovu blok šeme formirati VHDL opis brojača modula 138.



Slika 5.19: Komponenta koju treba projektovati

REŠENJE:

a)

Analizom postavke zadatka vidi se da je signal za postavljanje brojača u početno stanje, Reset, aktivan na niskom nivou i da je nezavisan od signala takta CLK, tj. asinhron je. Signal dozvole brojanja, označen sa E, je takođe asinhron i aktivan je na visokom logičkom nivou. Paralelni upis omogućava da se u brojač, signalom P koji je u ovom slučaju sinhron i aktivan na nivou logičke jedinice, dozvoli upis inicijalnog sadržaja, preko ulaznih signala D[3:0], od kojeg će brojač početi da broji.

Signal dozvole brojanja E i izlazni signal TSE koji ukazuje na kraj ciklusa brojanja (brojač je u stanju $F_{\text{Hex}}=1111_{\text{Bin}}$) su signali čiji značaj dolazi do izražaja pri kaskadnom vezivanju brojača radi realizacije brojača većeg modula (kao što će to biti slučaj u delu zadatka pod b).

Da bi se izvršila sinteza traženog brojača polazi se od formiranja tabele stanja brojača (Tabela 5.11).


















E	P	CLK	Sadašnje stanje (t)				Sledeće stanje (t+1)				TSE
			q ₃	q ₂	q ₁	q ₀	d ₃	d ₂	d ₁	d ₀	
0	×	×	Q ₃	Q ₂	Q ₁	Q ₀	Q ₃	Q ₂	Q ₁	Q ₀	0
1	1		×	×	×	×	D ₃	D ₂	D ₁	D ₀	1 za Q(t)=0 _{HEX}
1	0		0	0	0	0	0	0	0	1	0
1	0		0	0	0	1	0	0	1	0	0
1	0		0	0	1	0	0	0	1	1	0
1	0		0	0	1	1	0	1	0	0	0
1	0		0	1	0	0	0	1	0	1	0
1	0		0	1	0	1	0	1	1	0	0
1	0		0	1	1	0	0	1	1	1	0
1	0		0	1	1	1	1	0	0	0	0
1	0		1	0	0	0	1	0	0	1	0
1	0		1	0	0	1	1	0	1	0	0
1	0		1	0	1	0	1	0	1	1	0
1	0		1	0	1	1	1	1	0	0	0
1	0		1	1	0	0	1	1	0	1	0
1	0		1	1	0	1	1	1	1	0	0
1	0		1	1	1	0	1	1	1	1	0
1	0		1	1	1	1	0	0	0	0	1

Tabela 5.11: Tabela stanja brojača

Izlazni signal TSE ukazuje kraj na ciklusa brojanja, tj. kada je naredno stanje brojača inicijalno (0_{HEX}). To je ekvivalentno tome da je trenutno stanje brojača

F_{HEX} i da je aktivan ulazni signal dozvole brojanja E. Prema tome, na osnovu gornje tabele se dobija da je $TSE = Q_3 \cdot Q_2 \cdot Q_1 \cdot Q_0 \cdot E$.

Pomoću asinhronih ulaza u flip-flopove za postavljanje flip-flopa u stanje 0 realizovaće se postavljanje početnog stanja brojača ulaznim signalom Reset.

Dozvola brojanja će se realizovati koristeći flip-flopove sa ulaznim signalom dozvole upisa, tako što će se ulazni signal E direktno dovesti na ulaze dozvole upisa u flip-flopove.

Za potrebe određivanja ulaznih funkcija svakog od 4 potrebna D flip-flopa, sledeći korak je formiranje Karnoovih karti (Slika 5.20) za minimizaciju funkcija ulaza.

Radi jednostavnijeg postupka određivanja ulaznih funkcija prvo će se odrediti funkcije koje zavise samo od prethodnog stanja brojača, tj. one odgovaraju ulazima D flip-flopa brojača koji nema mogućnost paralelnog upisa. Kasnije će se uzeti u obzir i njihova zavisnost od signala dozvole paralelnog upisa (P).

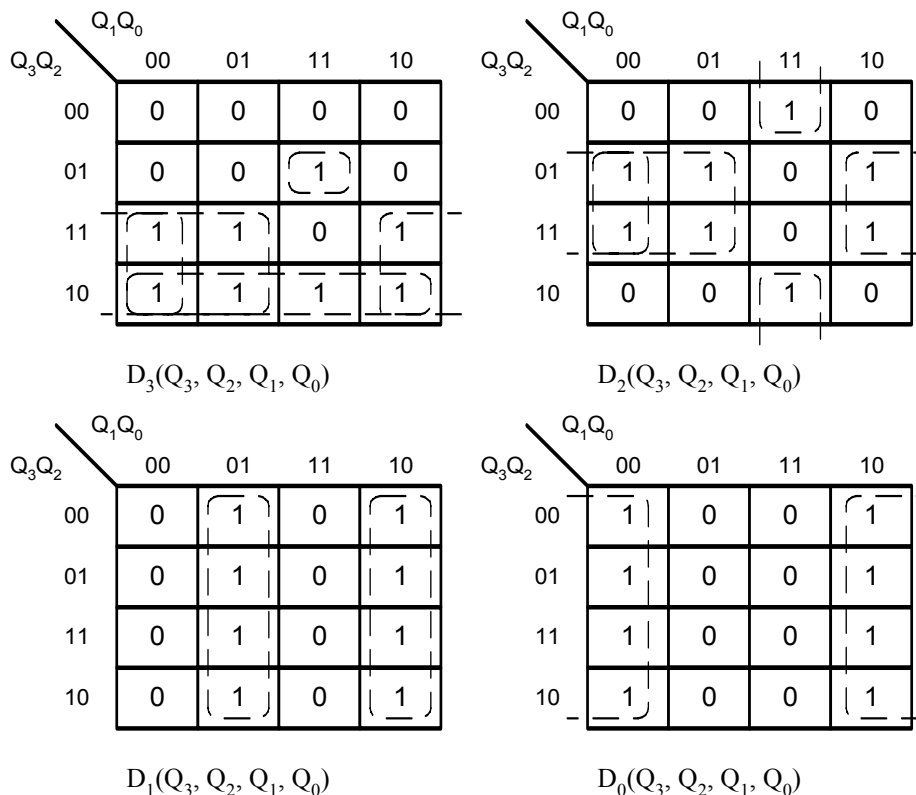
Na osnovu formiranih Karnoovih karti dobijaju se sledeće funkcije:

$$D_0' = \bar{Q}_0,$$

$$D_1' = \bar{Q}_1 \cdot Q_0 + Q_1 \cdot \bar{Q}_0,$$

$$D_2' = Q_2 \cdot \bar{Q}_1 + Q_2 \cdot \bar{Q}_0 + \bar{Q}_2 \cdot Q_1 \cdot Q_0,$$

$$D_3' = Q_3 \cdot \bar{Q}_2 + Q_3 \cdot \bar{Q}_1 + Q_3 \cdot \bar{Q}_0 + \bar{Q}_3 \cdot Q_2 \cdot Q_1 \cdot Q_0.$$



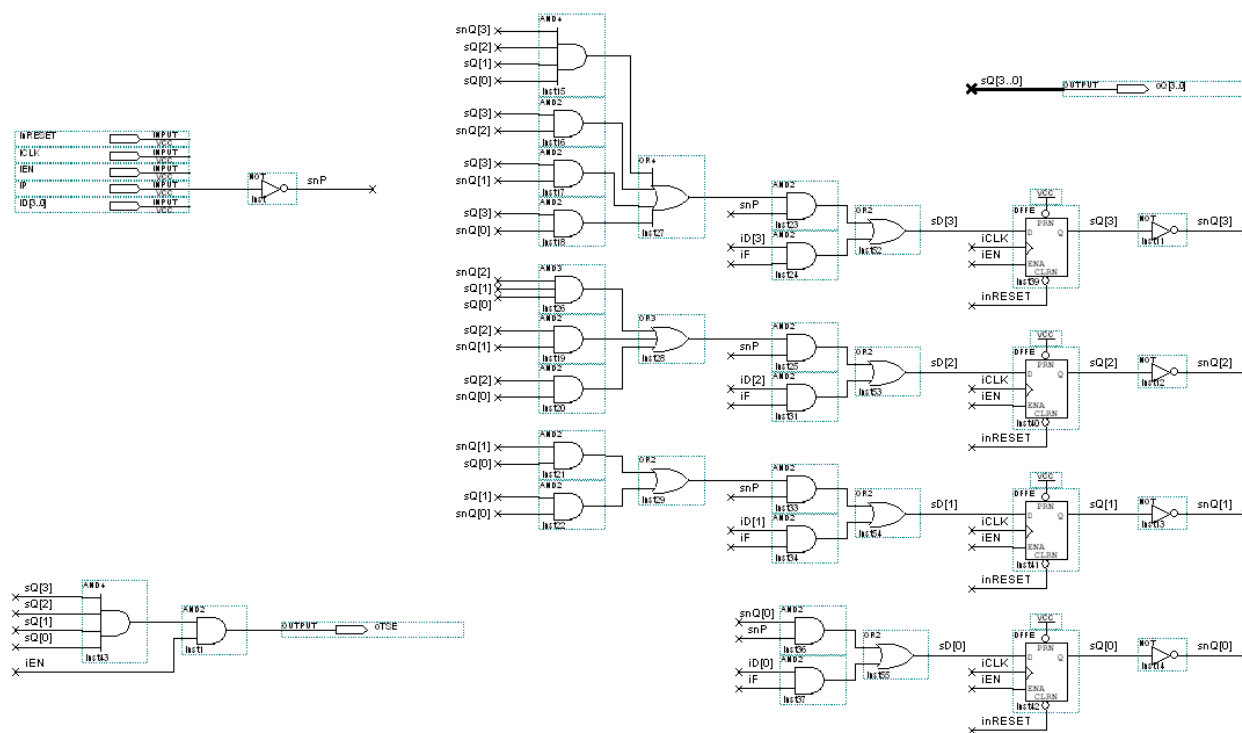
Slika 5.20: Karnoove karte funkcija D_3' , D_2' , D_1' i D_0'

Funkcije su označene sa ' zbog toga što to nisu kompletni izrazi pošto je izostavljen paralelni upis vrednosti sa ulaznih linija D[3:0] u brojač.

Da bi se ostvario brojač sa paralelnim upisom, na D ulaz svakog od flip flopova treba dovesti i logički nivo promenljive za paralelni upis (D_3, D_2, D_1 i D_0). To se može uraditi preko multipleksera 2×1 , pri čemu se u oba slučaja kao adresni signal koristi ulazni signal P koji je aktivan na nivou logičke jedinice. Multiplekser 2×1 se jednostavno realizuje pomoću dva dvoulazna I kola i jednog dvoulaznog ILI kola. Izlaz iz multipleksera je zapravo ulaz u odgovarajući flip-flop, pa sledi da je:

$$D_i = D_i' \cdot \bar{P} + P \cdot D_i.$$

Na osnovu dobijenih funkcija pristupa se formiranju logičke šeme realizovanog brojača, Slika 5.21.



Slika 5.21: Logička šema traženog brojača modula 16

Sledi VHDL opis ekvivalentnog brojača.

```

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;

ENTITY BROJAC_M16 IS PORT (
    iCLK      : IN  STD_LOGIC;
    inRESET,
    iEN, iP   : IN  STD_LOGIC;
    iDATA     : IN  STD_LOGIC_VECTOR(3 DOWNTO 0);
    oQ        : OUT STD_LOGIC_VECTOR(3 DOWNTO 0);
    oTSE      : OUT STD_LOGIC );
END BROJAC_M16 ;

```

```

ARCHITECTURE ARH_BROJAC_M16 OF BROJAC_M16 IS
    -- stanje brojača
    SIGNAL sCNT: UNSIGNED(3 DOWNT0 0);
BEGIN

    -- brojac sa asinhronim resetom
    PROCESS(inRESET, iCLK) BEGIN
        IF (inRESET = '0') THEN -- asinhroni reset
            sCNT <= "0000";
        ELSIF (iCLK'EVENT AND iCLK = '1') THEN
            IF (iEN = '1') THEN -- dozvoljeno brojanje?
                IF (iP = '1') THEN -- dozvoljen paralelni upis?
                    sCNT <= UNSIGNED(iDATA); -- paralelni upis
                ELSE
                    sCNT <= sCNT + 1; -- brojanje
                END IF;
            END IF;
        END IF;
    END PROCESS;

    -- formiranje izlaznog signala koji
    -- označava kraj ciklusa brojanja
    PROCESS (iEN, sCNT) BEGIN
        IF (iEN='1' AND sCNT = 15) THEN
            oTSE <= '1'; -- kraj ciklusa brojanja
        ELSE
            oTSE <= '0';
        END IF;
    END PROCESS;

    -- formiranje izlaznog vektora
    oQ <= STD_LOGIC_VECTOR(sCNT);

END ARH_BROJAC_M16;

```

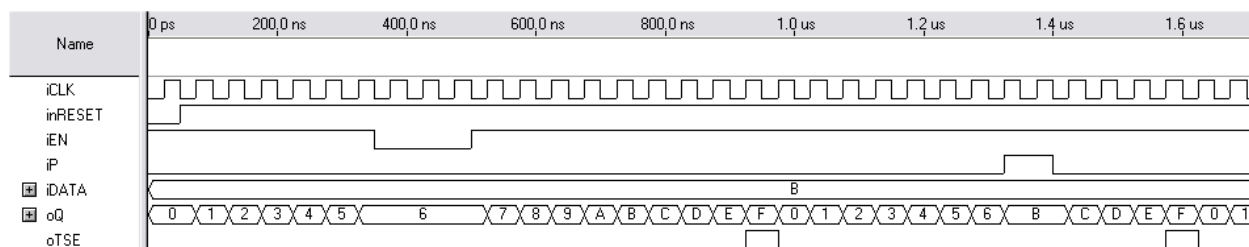
Svi ulazni i izlazni signali realizovanog brojača su tipa `std_logic`. Pošto u VHDL-u nije moguće realizovati brojač ovim tipom, u ovom slučaju je za realizaciju brojača upotrebljen tip `UNSIGNED`. Zbog toga je potrebna konverzija tipova prilikom upisa vrednosti u brojač (`sCNT<=UNSIGNED(iDATA)`) i prosleđivanja stanja brojača na izlazni vektor (`oQ <= STD_LOGIC_VECTOR(CNT)`).

Formiranje izlaznog signala `oTSE` je realizovano u zasebnom procesu. Ovakva realizacija je neophodna pošto je u zadatku specificirano da ovaj signal treba formirati kao kombinacionu mrežu bez baferovanja signala na izlazu. Pomoću ovog nezavisnog procesa formira se tražena kombinaciona mreža koja je nezavisna od korišćenih flip-floпова za realizaciju brojača (prvi proces).

Vremenski dijagram rada brojača prikazuje Slika 5.22.

Na početku vremenskog dijagrama je brojač postavljen u inicijalno stanje pomoću ulaznog signal `inRESET` koji je na niskom logičkom nivou. Nakon što se `inRESET` postavi u stanje logičke jedinice, brojač broji na svaku rastuću ivicu takt signala sve dok je aktivan signal dozvole brojanja `iEN`. Sa njegovi postavljanjem na nizak logički nivo brojač zadržava svoje stanje sve dok se ponovo ne vrati na visok nivo (vremenski interval 350ns – 500ns). Na kraju cilusa brojanja brojač aktivira izlazni signal `oTSE` ($t=900\text{ns}$ i $t=1,55\mu\text{s}$). U vremenskom trenutku

$t=1,35\mu s$ se aktivira ulazni signal dozvole upisa sadržaja u brojač iP . Pošto je aktivan i signal dozvole brojanja iEN , na prvoj sledećoj rastućoj ivici takt signala se upisuje novi sadržaj u brojač. Isti sadržaj se upisuje u brojač na svakoj sledećoj rastućoj ivici takta sve dok su aktivni signali iEN i iP (u ovom slučaju to su dve periode takta).



Slika 5.22: Simulacija rada realizovanog brojača

b)

Da bi se realizovao brojač modula 138 potrebna su 2 brojača realizovana u zadatku pod a). Maksimalni broj stanja koja može da realizuje brojač dobijen kaskadnom vezom takva 2 brojača je $16 \cdot 16 = 256$ stanja. Potrebno je, dakle, na neki način obezbediti da brojač nakon 138-og stanja pređe u početno, pri čemu početno stanje može biti bilo koje od mogućih 256 stanja.

To se može učiniti na više načina. Jedan od njih jeste da brojač kreće iz stanja $(255-137)=118=0111\ 0110_{\text{Bin}}=76_{\text{Hex}}$, broji do $255=1111\ 1111_{\text{Bin}}=FF_{\text{Hex}}$ kada signal TSE prima vrednost 1, a zatim kada se brojač nađe u stanju 255 ulaz dozvole paralelnog upisa P se postavlja na nivo logičke jedinice i u brojač se upisuje sadržaj 0111 0110. Brojač realizovan na ovaj način će brojati od 118 do 255, što čini ukupno 138 stanja, i time je realizovan brojač modula 138.

Primitimo da signal Reset koji je brojač modula 16 postavljao u nulto stanje (stanje 0000) i bio aktivan na nivou logičke nule. Pošto ovo stanje nije obuhvaćeno sa skupom stanja brojača modula 138, ulazi za postavljanje inicijalnog stanja korišćenih brojača modula 16 treba da budu stalno na visokom logičkom nivou.

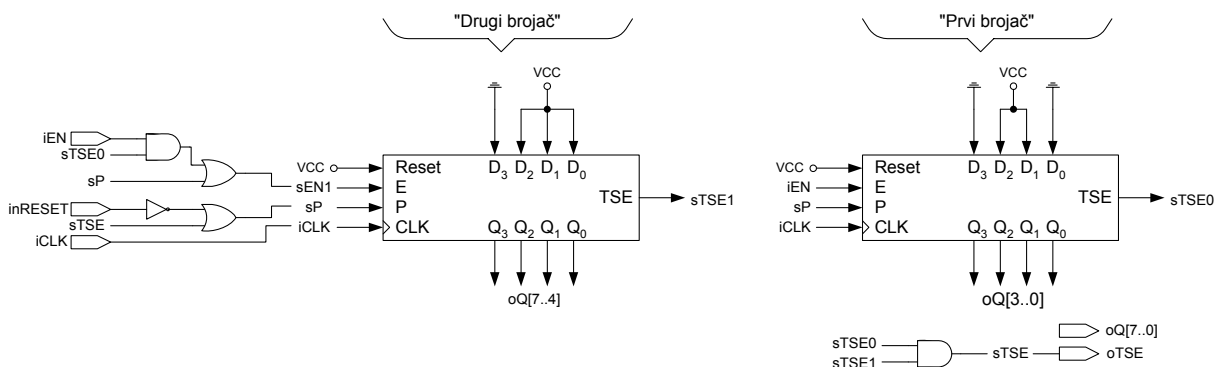
Ovakvom realizacijom je moguće pomoću spoljnog ulaza brojač postaviti u stanje 118, tako što se u taj spoljni signal, npr. RESET, dovede na ulazni signal paralelnog upisa u oba brojača. Kompatibilnosti radi, sa brojačem modula 16 realizovanim u zadatku pod a), neka i ovaj signal bude aktivan na niskom logičkom nivou. Na ovaj način stanje 118 postaje inicijalno stanje realizovanog brojača modula 138.

Prema tome, na ulaz P pojedinačnih brojača se dovodi: $P = \overline{\text{RESET}} + \text{TSE}$, gde je signal TSE koji označava kraj ciklusa brojanja (brojač je u 138-mom stanju, odnosno u stanju 255) aktivan kada su aktivni signali TSE oba pojedinačna brojača, odnosno: $\text{TSE} = \text{TSE}_1 \cdot \text{TSE}_0$.

Moguće je uvesti i ulazni signal dozvole brojanja, EN , za realizovani brojač modula 138. On se direktno dovodi na ulaz dozvole brojanja brojača koji broji

donja 4 bita brojača modula 138 (Q_3 do Q_0). Drugi brojač (Q_7 do Q_4) svoje stanje uvećava za jedan samo u slučaju kada prvi brojač dođe do kraja ciklusa brojanja, tj. aktivan je signal TSE_0 . Naravno, pri tome mora biti aktivan i signal dozvole brojanja. Pored toga signal dozvole brojanja za drugi brojač mora biti aktivan i kada se vrši upisivanje određenog sadržaja, Tabela 5.10. Prema tome na ulaz dozvole brojanja drugog brojača se dovodi $EN_1 = EN \cdot TSE_0 + P$.

Slika 5.23 prikazuje blok šemu brojača modula 138 realizovanu u skladu sa prethodnom analizom.



Slika 5.23: Blok šema realizacije brojača modula 138 pomoću brojača modula 16

Odgovarajući VHDL kod koji realizuje prethodno opisani brojač modula 138 ima sledeći oblik:

```

LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY BROJAC_M138 IS PORT(
    iCLK, iEN, inRESET: IN STD_LOGIC;
    oQ: OUT STD_LOGIC_VECTOR(7 DOWNTO 0);
    oTSE: BUFFER STD_LOGIC );
END BROJAC_M138;

ARCHITECTURE ARH_BROJAC_M138 OF BROJAC_M138 IS
    -- opis entiteta komponente brojac modula 16
    -- od cega ce se formirati brojac modula 138
    COMPONENT BROJAC_M16 PORT (
        iCLK      : IN  STD_LOGIC;
        inRESET,
        iEN, iP   : IN  STD_LOGIC;
        iDATA     : IN  STD_LOGIC_VECTOR(3 DOWNTO 0);
        oQ        : OUT STD_LOGIC_VECTOR(3 DOWNTO 0);
        oTSE      : OUT STD_LOGIC );
    END COMPONENT;

    -- konstantne vrednosti koje se koriste prilikom
    -- povezivanja brojac modula 16. To je potrebno
    -- zbog toga sto u VHDLu nije dozvoljeno direktno
    -- koristiti konstantu u listi povezivanja portova
    CONSTANT cONE: STD_LOGIC := '1';
    CONSTANT cHIGH_DATA: STD_LOGIC_VECTOR := "0111";
    CONSTANT cLOW_DATA: STD_LOGIC_VECTOR := "0110";

```

```

-- identifikacija kraja ciklusa brojanja
-- pojedinačni brojaci
SIGNAL sTSE0, sTSE1: STD_LOGIC;
SIGNAL sP: STD_LOGIC; -- upis početne vrednosti oba brojaca
SIGNAL sEN1: STD_LOGIC; -- dozvola brojanja "vi seg" brojaca
BEGIN

-- upisati početne vrednosti u brojac ako je dosao zahtev za
-- inicijalizaciju brojaca ili kada je
-- brojac dosao do kraja ciklusa brojanja
sP <= NOT(inRESET) OR oTSE;

-- "vi si" brojac broji kada je aktivan signal dozvole brojanja i
-- kada je "ni zi" brojac dosao do kraja ciklusa brojanja
-- signal takodje treba da je aktivan
-- kada se upisuju podaci u brojac
sEN1 <= (iEN AND sTSE0) OR sP;

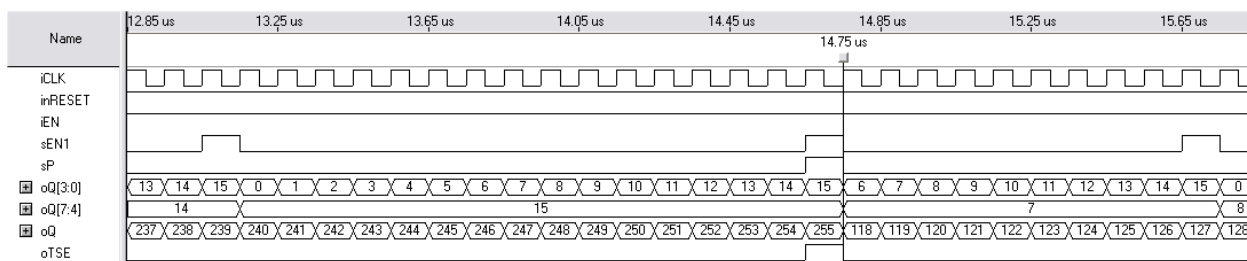
-- "ni zi" brojac
eBROJAC0: BROJAC_M16 PORT MAP(iCLK=>iCLK, inRESET=>cONE,
                                iEN=>iEN, iP=>sP,
                                iDATA=>cLOW_DATA,
                                oQ=>oQ(3 DOWNTO 0), oTSE=>sTSE0);

-- "vi si" brojac
eBROJAC1: BROJAC_M16 PORT MAP(iCLK=>iCLK, inRESET=>cONE,
                                iEN=>sEN1, iP=>sP,
                                iDATA=>cHIGH_DATA,
                                oQ=>oQ(7 DOWNTO 4), oTSE=>sTSE1);

-- formiranje izlaznog signala koji
-- označava kraj ciklusa brojanja
oTSE <= sTSE0 AND sTSE1;
END ARH_BROJAC_M138;

```

Slika 5.24 prikazuje jedan deo vremenskog dijagrama simulacije rada realizovanog brojača modula 138. Na slici je prikazan način rada oba brojaca. Prvi brojac ($oQ[3:0]$) broji sa svakom rastućom ivicom takta, dok drugi brojac ($oQ[7:4]$) broji samo kada prvi brojac dođe do kraja ciklusa brojanja ($sEN1=1$). Stanje realizovanog brojača modula 138 prikazuje signal oQ . U vremenskom trenutku $t=14,75\mu s$ se završava ciklus brojanja i brojac učitava početnu vrednost ($sEN1=1$ i $sP=1$). Takođe, u toku poslednjeg stanja brojaca aktivan je signal $oTSE$ koji ukazuje na kraj ciklusa brojanja.



Slika 5.24: Deo vremenskog dijagrama simulacije rada realizovanog brojača modula 138

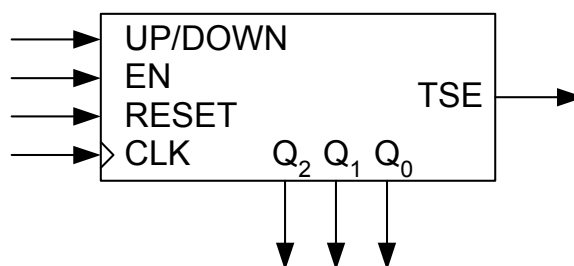
5.7 ZADATAK:

- a). Izvršiti sintezu brojača modula 8 sa mogućnošću brojanja na gore i na dole, kao i sa upravljanjem brojanja i generisanjem signala TSE za detekciju kraja jednog ciklusa brojanja. Prenosnu funkciju brojača kog treba sintetizovati prikazuje Tabela 5.12.

CLK	RESET	E	UP/DOWN	Q(t+1)	TSE
	0	×	×	0	0
	1	0	×	Q(t)	0
	1	1	1	Q(t)+1	1 za Q(t)=15
	1	1	0	Q(t)-1	1 za Q(t)=0

Tabela 5.12: Prenosna funkcija brojača

Sintezu izvršiti pomoću VHDL jezika za opis fizičke arhitekture. Izgled komponente koju treba projektovati prikazuje Slika 5.25.



Slika 5.25: Izgled komponente koju treba projektovati

- b). Povezati potreban broj ovakvih brojača sa odgovarajućom logikom da bismo dobili dvosmerni brojač modula 167.

REŠENJE:

a)

Na osnovu prenosne funkcije brojača date u tekstu zadatka, vidi se da ovaj brojač treba da ima sinhroni reset, tj. da za RESET=0 na izlazu brojača je stanje 0. U slučaju da je kontrolni signal E=0, brojač neće vršiti brojanje, već će ostati u tekućem stanju. U zavisnosti od signala UP/DOWN, vršiće se ili brojanje na gore (UP/DOWN=1) ili brojanje na dole (UP/DOWN=0). Na izlazu brojača postoji i signal TSE koji pokazuje da je brojač na kraju ciklusa brojanja.

Sinteza brojača modula 8 predstavlja sintezu trobitnog brojača za čiju realizaciju su neophodna tri memorijska elementa (tri D flip-flopa). Ilustracije radi, prelaze stanja brojača prikazuje Tabela 5.13.

Na osnovu tabele prelaza stanja brojača je moguće odrediti prenosne funkcije koje definišu ulazne signale u flip-flopove kada je aktivan signal dozvole brojanja EN. Pri tome prenosne funkcije su zavisne od trenutnog stanja brojača (Q_2 , Q_1 , Q_0) i ulaznog signala UP/DOWN.

Iz tabele se može odrediti i prenosna funkcija za izlazni signal TSE, koji je aktivan kad brojač stigne na kraj ciklusa brojanja. Kad brojač broji na gore, kraj ciklusa je kad su sva tri flip-flopa u stanju 1. U slučaju brojanja na dole, kraj ciklusa brojanja je kada su svi flip-flopovi u stanju 0. Stoga se signal TSE može definisti sledećom jednačinom:

$$TSE = UP/DOWN \cdot Q_2 \cdot Q_1 \cdot Q_0 + \overline{UP/DOWN} \cdot \overline{Q_2} \cdot \overline{Q_1} \cdot \overline{Q_0} ,$$

UP/DOWN	Trenutno stanje			Naredno stanje			TSE
	Q ₂	Q ₁	Q ₀	D ₂	D ₁	D ₀	
0	0	0	0	1	1	1	1
0	0	0	1	0	0	0	0
0	0	1	0	0	0	1	0
0	0	1	1	0	1	0	0
0	1	0	0	0	1	1	0
0	1	0	1	1	0	0	0
0	1	1	0	1	0	1	0
0	1	1	1	1	1	0	0
1	0	0	0	0	0	1	0
1	0	0	1	0	1	0	0
1	0	1	0	0	1	1	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	1	0
1	1	0	1	1	1	0	0
1	1	1	0	1	1	1	0
1	1	1	1	0	0	0	1

Tabela 5.13: Tabela prelaza brojača

Sledi VHDL opis traženog brojača.

```

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;

ENTITY BROJAC_M8 IS PORT(
    iCLK:      IN   STD_LOGIC;
    inRESET:   IN   STD_LOGIC;
    iEN:       IN   STD_LOGIC;
    iUP_DOWN:  IN   STD_LOGIC;
    oQ:        OUT  UNSIGNED(2 DOWNTO 0);
    oTSE:      OUT  STD_LOGIC);
END BROJAC_M8;

ARCHITECTURE ARH_BROJAC_M8 OF BROJAC_M8 IS
    -- stanje brojača
    SIGNAL scnt: UNSIGNED(2 DOWNTO 0);
BEGIN

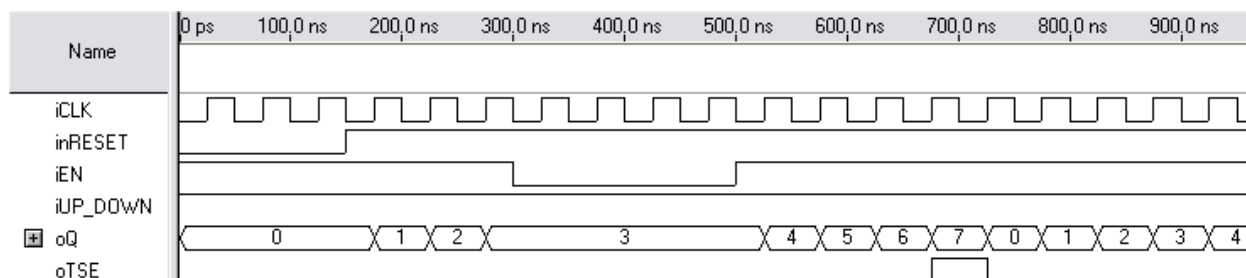
```

```
-- brojac modula 8
PROCESS(iCLK) BEGIN
  IF (iCLK'EVENT AND iCLK='1') THEN
    IF (inRESET = '0') THEN -- sinhroni reset
      sCNT <= "000";
    ELSE
      IF (iEN = '1') THEN -- brojanje dozvoljeno?
        IF (iUP_DOWN = '1') THEN
          sCNT <= sCNT + 1; -- brojanje na gore
        ELSE
          sCNT <= sCNT - 1; -- brojanje na dolje
        END IF;
      END IF;
    END IF;
  END IF;
END PROCESS;

-- generisanje signala indikacije kraja ciklusa brojanja
PROCESS (iEN, iUP_DOWN, sCNT) BEGIN
  IF ((iEN='1' AND iUP_DOWN='1' AND sCNT=7) OR -- na gore
      (iEN='1' AND iUP_DOWN='0' AND sCNT=0)) THEN -- na dolje
    oTSE<='1';
  ELSE
    oTSE<='0';
  END IF;
END PROCESS;

-- preslikavanje stanja brojača na izlazni vektor
oQ <= sCNT;
END ARH_BROJAC_M8;
```

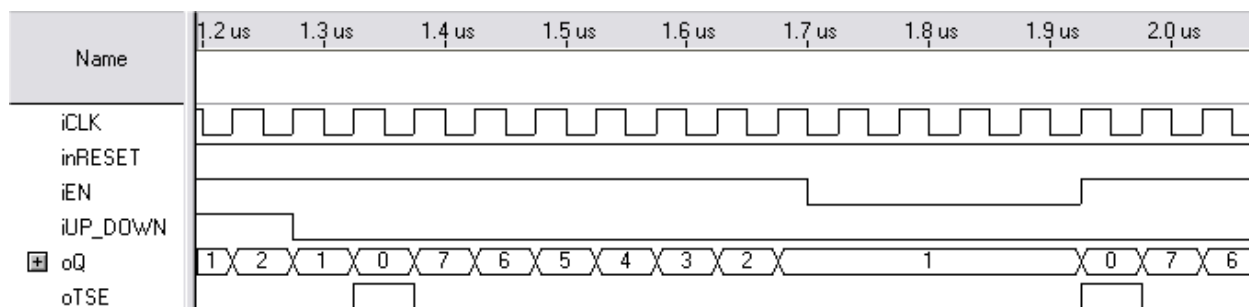
Vremenski dijagram rada brojača tokom brojanja na gore prikazuje Slika 5.26. Na početku vremenskog dijagrama brojač je postavljen u početno stanje signala inRESET koji je u tom slučaju na niskom nivou. Prelaskom ovog signala na visok nivo brojač počinje da broji na gore pošto su signali iEN i iUP_DOWN na visokom nivou. U slučaju da je signal dozvole brojanja iEN pao na nizak naponski nivo, kao u vremenskom intervalu od 300ns do 500ns, brojač zadržava svoje trenutno stanje. Na kraju ciklusa brojanja, kada dostigne vrednost 7_{HEX}, brojač generiše izlazni signal oTSE.



Slika 5.26: Simulacija rada registra prilikom brojanja na gore

U nastavku simulacije rada realizvanog brojača proverava se rad brojača prilikom brojanja da dole, Slika 5.27. U vremenskom trenutku 1,255μs signal

iUP_DOWN menja vrednost na nulu, čime započinje ciklus brojanja na dole. Vidi se da brojač broji dok je signal dozvole brojanja na visokom nivou. Inače zadržava trenutnu vrednost. Na kraju ciklusa brojanja, kad dostigne vrednost 0_{HEX}, na dole brojač generiše izlazni signal oTSE.



Slika 5.27: Simulacija rada registra prilikom brojanja na dole

b)

U drugom delu zadatka neophodno je povezati potreban broj sintetizovanih brojača modula 8 sa ciljem dobijanja brojača modula 167. Kako je $8^2=64 < 167$, odnosno, $8^3=512 > 167$, može se zaključiti da je za realizaciju brojača modula 167 potrebno povezati 3 brojača modula 8.

U slučaju da prilikom brojanja, sve tri cifre “najnižeg” brojača postanu 1, njegov izlazni signal TSE₀ postaje 1 i on se dovodi na ulaz EN drugog brojača modula 8, zajedno sa signalom dozvole brojanja brojača EN, koji u tom trenutku počinje da broji. Da bi treći “najviši” brojač počeo da broji potrebno je da su i TSE₀=1 i TSE₁=1. Tako se na ulaz EN trećeg brojača dovodi signal $EN_2 = EN \cdot TSE_1 \cdot TSE_0$.

Izlaz iz ovako dobijenog brojača biće devetobitni signal $I_8 I_7 I_6 I_5 I_4 I_3 I_2 I_1 I_0$ koji može da broji od 0 do 511. Međutim, kako je potreban brojač modula 167, mora se detektovati poslednju cifru u ciklusu brojanja i da se tada brojač postavi u nultu stanje. Neka početna vrednost brojača bude 0_{HEX}. Kada brojač modula 167, broji na gore, moguća stanja su 0,1,...,166, pa je poslednje stanje u kome brojač može da se nađe 166. Već u sledećem ciklusu brojač treba da je u inicijalnom stanju. Kada brojač broji na dole moguća stanja su 0,511,510,...,346, pa je brojač potrebno resetovati na izlasku iz stanja 346. To se binarno može predstaviti kao: $166_{10} = 010100110_2$, odakle sledi da je signal za resetovanje brojača tokom brojanja na gore $RST_1 = UPDOWN \cdot I_7 \cdot I_5 \cdot I_2 \cdot I_1$. U slučaju brojanja na dole brojač se resetuje kada dođe u stanje $346_{10} = 101011010_2$ pa se dobija sledeća jednačina: $RST_2 = UPDOWN \cdot I_8 \cdot I_6 \cdot I_4 \cdot I_3 \cdot I_1$. Dakle, odgovarajući signal za postavljanje brojača u inicijalno stanje će biti $RST = RST_1 + RST_2 + RESET$, gde je signal RESET ulazni reset signal doveden spolja.

Odgovarajući VHDL kod je prikazan u nastavku.

```

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;

ENTITY BROJAC_M167 IS PORT(
    iCLK, inRESET, iEN, iUP_DOWN: IN STD_LOGIC;
    oQ: BUFFER UNSIGNED(8 DOWNTO 0);
    oTSE: OUT STD_LOGIC
);
END BROJAC_M167;

ARCHITECTURE ARH_BROJAC_M167 OF BROJAC_M167 IS

    -- opis entiteta komponente brojaca modula 8
    -- od cega ce se formirati brojac modula 167
    COMPONENT BROJAC_M8 PORT (
        iCLK: IN STD_LOGIC;
        inRESET: IN STD_LOGIC;
        iEN: IN STD_LOGIC;
        iUP_DOWN: IN STD_LOGIC;
        oQ: OUT UNSIGNED(2 DOWNTO 0);
        oTSE: OUT STD_LOGIC);
    END COMPONENT;

    -- signali koji se koriste prilikom povezivanja brojaca
    SIGNAL stSE0, stSE1, stSE2: STD_LOGIC;
    SIGNAL sen0, sen1, sen2: STD_LOGIC;
    SIGNAL snRESET, sRESET_UP, sRESET_DOWN: STD_LOGIC;

BEGIN

    -- dozvol a broj anja pojedinačni h broj aca
    sen0 <= iEN;
    sen1 <= iEN AND stSE0;
    sen2 <= iEN AND stSE0 AND stSE1;

    -- formiranje signala za postavljanje brojaca u početno stanje
    -- tokom brojanja na gore
    sRESET_UP <= '1' WHEN ((iUP_DOWN='1') AND (oQ = 167)) ELSE '0';
    -- tokom brojanja na dole
    sRESET_DOWN <= '1' WHEN ((iUP_DOWN='0') AND (oQ = 346)) ELSE '0';
    -- signal za postavljanje brojaca u početno stanje
    -- korišćeno je trouglažno I kol o zbog negativne logike signala
    snRESET <= inRESET AND NOT(sRESET_UP) AND NOT(sRESET_DOWN);

    -- povezivanje brojaca modula 8
    eBROJAC0: BROJAC_M8 PORT MAP(iCLK=>iCLK, inRESET=>snRESET,
                                iEN=>sen0, iUP_DOWN=>iUP_DOWN,
                                oQ=>oQ(2 DOWNTO 0), oTSE=>stSE0);
    eBROJAC1: BROJAC_M8 PORT MAP(iCLK=>iCLK, inRESET=>snRESET,
                                iEN=>sen1, iUP_DOWN=>iUP_DOWN,
                                oQ=>oQ(5 DOWNTO 3), oTSE=>stSE1);
    eBROJAC2: BROJAC_M8 PORT MAP(iCLK=>iCLK, inRESET=>snRESET,
                                iEN=>sen2, iUP_DOWN=>iUP_DOWN,
                                oQ=>oQ(8 DOWNTO 6), oTSE=>stSE2);

    -- formiranje izlaznog signala koji ukazuje na kraj ciklusa brojanja
    oTSE <= sRESET_UP OR sRESET_DOWN;

END ARH_BROJAC_M167;

```


5.8 ZADATAK:

Izvršiti sintezu rednog brojača modula 5 pomoću D flip-flopova i VHDL jezika za opis fizičke arhitekture. Brojač ne poseduje paralelni ulaz podataka i njegovo početno stanje je 00001 binarno.

Tabela 5.14 prikazuje tabelu prelaza D flip-flopa.

Q(t)	Q(t+1)	D
0	0	0
0	1	1
1	0	0
1	1	1

Tabela 5.14: Tabela prelaza D flip-flopa

REŠENJE:

Redni brojači su sekvencijalne mreže čija je odlika da sadrže onoliko memorijskih elemenata koliki im je moduo. Realizuju se kao paralelni brojači, što znači da se stanja memorijskih elmenata menjaju sinhrono sa nailaskom takta. Memorijski elementi spojeni su tako da je izlaz prethodnog vezan za ulaz narednog, s tim da je izlaz poslednjeg memorijskog elmenta u nizu vezan na ulaz prvog. Ovako povezivanje omogućuje da se početno uneti sadržaj (pomoću set i reset ulaza) kruži u brojaču, pomerajući se za po jedan bit sa svakim impulsom takta. Zbog toga se ovi brojači često nazivaju kružni brojači. Redni brojači imaju mogućnost direktnog očitavanja, a ukoliko im se granice otvore dobija se pomerački registar.

Na osnovu tablice prelaza D flip-flopa i funkcionisanja rednog brojača vrši se njegova sinteza što ilustruje Tabela 5.15.

Posmatranjem tabele lako se uočava da je: $D_A = Q_E$; $D_B = Q_A$; $D_C = Q_B$; $D_D = Q_C$; $D_E = Q_D$. Na osnovu dobijenih jednačina može se izvršiti sinteza rednog brojača pomoću D flip-flopova, Slika 5.28.

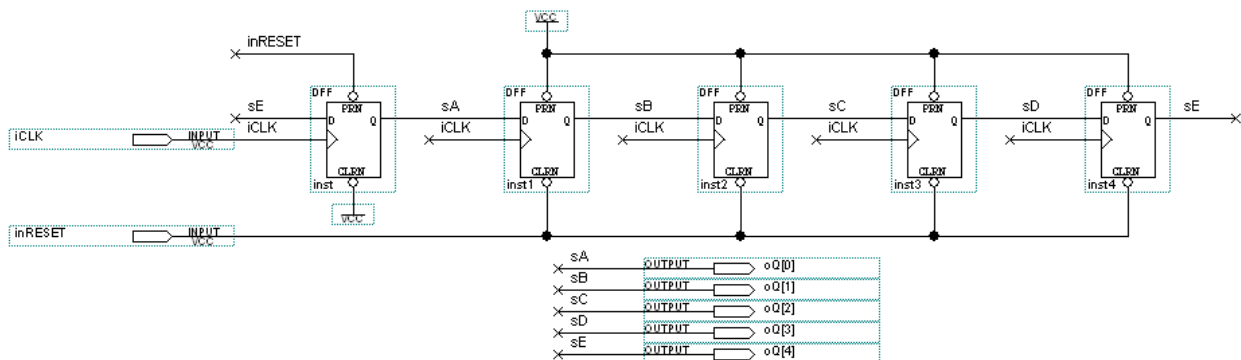
Na slici je flip-flop najmanje važnosti prikazan sa krajnje leve strane (izlazni signal sA). Ostali flip-flopovi su poredani redom do flip-flopa najveće važnosti koji je sa krajnje desne strane (izlazni signal sE).

i	Q _E	Q _D	Q _C	Q _B	Q _A	D _E	D _D	D _C	D _B	D _A
1	0	0	0	0	1	0	0	0	1	0
2	0	0	0	1	0	0	0	1	0	0
3	0	0	1	0	0	0	1	0	0	0
4	0	1	0	0	0	1	0	0	0	0
5	1	0	0	0	0	0	0	0	0	1

Tabela 5.15: Tabela prelaza stanja rednog brojača modula 5

Postavljanje početne vrednosti brojača na prethodnoj šemi je realizovano korišćenjem asinhronih ulaza flip-flopova za postavljanje vrednosti flip-flopa na 0 ili 1. Kod realizacije je signal za inicijalizaciju (inRESET) kod krajnjeg levog flip-flopa doveden na ulaz za postavljanje flip-flopa na 1 (PRN ulaz), a kod svih

ostalnih flip-flopora na ulaz za postavljanje flip-flopa na 0 (CLRn ulaz). Suprotni ulazi flip-flopora su postavljeni na visok logički nivo radi prevencije njihovog aktiviranja i dovođenja brojača u nedozvoljeno stanje.



Slika 5.28: Logička šema rednog brojača modula 5

VHDL kod rednog brojača modula 5 sa sinhronim resetom je prikazan u nastavku.

```

LIBRARY ieee;
USE ieee.std_logic_1164.all;

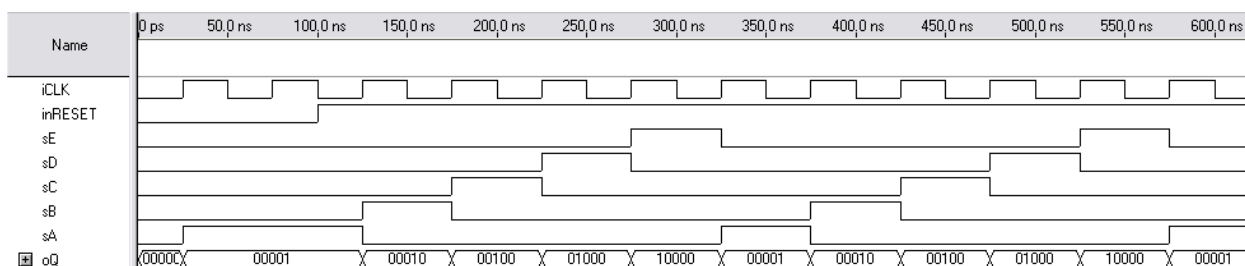
ENTITY BROJAC_M5_REDNI IS
    PORT (
        iCLK:      IN  STD_LOGIC;
        inRESET:   IN  STD_LOGIC;
        oQ:        OUT STD_LOGIC_VECTOR(4 DOWNTO 0) );
END BROJAC_M5_REDNI;

ARCHITECTURE ARH_BROJAC_M5_REDNI OF BROJAC_M5_REDNI IS
    -- signali koji predstavljaju izlaze iz flip-flopora
    SIGNAL sA,sB,sC,sD,sE: STD_LOGIC;
BEGIN
    -- opis rednog brojac
    PROCESS (iCLK) BEGIN
        IF (iCLK'EVENT AND iCLK = '1') THEN
            IF(inRESET = '0')THEN -- sinhroni reset
                sA <= '1';
                sB <= '0';
                sC <= '0';
                sD <= '0';
                sE <= '0';
            ELSE
                sA <= sE;
                sB <= sA;
                sC <= sB;
                sD <= sC;
                sE <= sD;
            END IF;
        END IF;
    END PROCESS;

    -- formiranje izlaznog vektora
    oQ <= (sE & sD & sC & sB & sA);
END ARH_BROJAC_M5_REDNI;

```

Slika 5.29 prikazuje vremenski dijagram realizovanog rednog brojača. Na početku vremenskog dijagrama je postavljanjem ulaznog signala `inRESET` na nizak naponski nivo brojač postavljen u početno stanje. Nakon toga, postavljanjem signala `inRESET` na visoko nivo, upisani sadržaj brojača kruži za po jedno mesto prema bitu veće važnosti sa svakom rastućom ivicom takt signala `iCLK`.



Slika 5.29: Vremenski dijagram rednog brojača

5.9 ZADATAK:

Na osnovu rednog brojača iz prethodnog zadatka izvršiti sintezu Džonsonovog brojača modula 10 pomoću D flip-flova i VHDL jezika za opis fizičke arhitekture. Početno stanje brojača je 0.

Tabela 5.16 prikazuje tabelu prelaza D flip-flopa.

Q(t)	Q(t+1)	D
0	0	0
0	1	1
1	0	0
1	1	1

Tabela 5.16: Tabela prelaza D flip-flopa

REŠENJE:

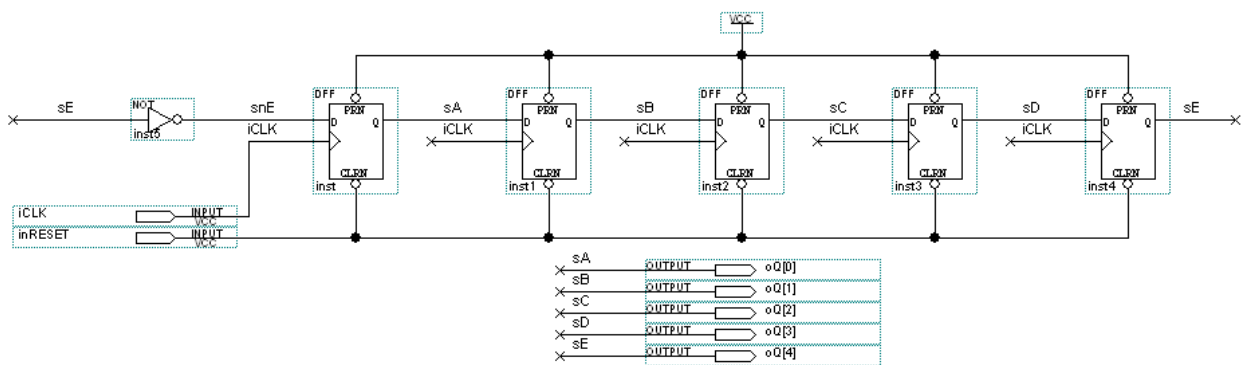
Džonsonovi brojači su specijalni slučaj rednog brojača. Memorijski elementi su takođe spojeni tako da je izlaz prethodnog vezan za ulaz narednog, sa tom razlikom da je invertovani izlaz poslednjeg memorijskog elementa u nizu vezan na ulaz prvog. Takvom se realizacijom postižu, za primer 5 memorijskih elemenata, prelasci stanja brojača koje ilustruje Tabela 5.17.

Vidi se da je kod ove vrste brojača sa n memorijskih elemenata moguće realizovati brojač modula $2 \cdot n$.

Logičku šemu Džonsonovog brojača modula 10 prikazuje Slika 5.30.

i	Q _A	Q _B	Q _C	Q _D	Q _E	D _A	D _B	D _C	D _D	D _E
0	0	0	0	0	0	1	0	0	0	0
1	1	0	0	0	0	1	1	0	0	0
2	1	1	0	0	0	1	1	1	0	0
3	1	1	1	0	0	1	1	1	1	0
4	1	1	1	1	0	1	1	1	1	1
5	1	1	1	1	1	0	1	1	1	1
6	0	1	1	1	1	0	0	1	1	1
7	0	0	1	1	1	0	0	0	1	1
8	0	0	0	1	1	0	0	0	0	1
9	0	0	0	0	1	0	0	0	0	0

Tabela 5.17: Prelasci stanja Džonsonovog brojača modula 10



Slika 5.30: Logička šema Džonsonovog brojača modula 10

Ekvivalentan VHDL kod (izuzev signala za postavljanje početnog stanja, inRESET, koji u slučaju VHDL realizacije sinhron) je prikazan u nastavku.

```

LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY BROJAC_M10_JONSON IS
  PORT (
    iCLK:      IN  STD_LOGIC;
    inRESET:   IN  STD_LOGIC;
    oQ:        OUT STD_LOGIC_VECTOR(4 DOWNTO 0) );
END BROJAC_M10_JONSON;

ARCHITECTURE ARH BROJAC M10 JONSON OF BROJAC M10 JONSON IS
  -- signali koji predstavljaju izlaze iz flip-flopa
  SIGNAL sA,sB,sC,sD,sE: STD_LOGIC;
BEGIN
  -- opis Jonsonovog brojaca
  PROCESS (iCLK) BEGIN
    IF (iCLK'EVENT AND iCLK = '1') THEN
      IF(inRESET = '0') THEN -- sinhroni reset
        sA <= '0';
        sB <= '0';

```

```

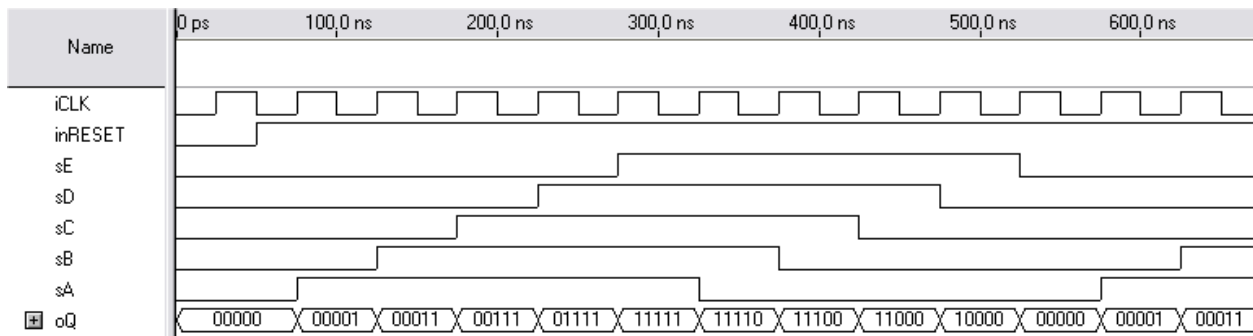
    sC <= '0';
    sD <= '0';
    sE <= '0';
ELSE
    sA <= NOT(sE);
    sB <= sA;
    sC <= sB;
    sD <= sC;
    sE <= sD;
END IF;
END IF;
END PROCESS;

oQ <= (sE & sD & sC & sB & sA); -- formiranje izlaznog vektora

END ARH_BROJAC_M10_JONSON;

```

Vremenski dijagram simulacije rada realizovanog brojača prikazuje Slika 5.31. Vidi se da prelasci stanja realizovanog brojača odgovaraju prelascima stanja koje prikazuje Tabela 5.17.



Slika 5.31: Vremenski dijagram Džonsonovog brojača

