Ü5: Kombinatorische und sequentielle Automatenmodelle in VHDL

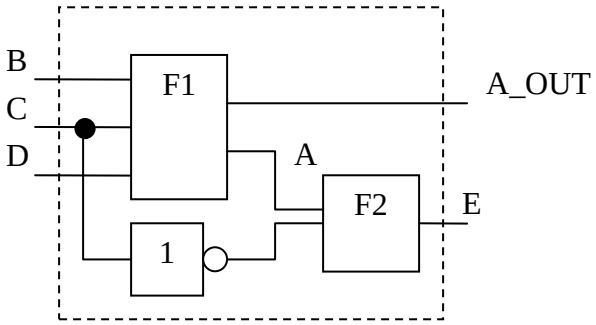
Aufgabe 1:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY aufgabe_eins IS
        PORT(B, C, D : IN std_logic;
             A_OUT, E :  OUT std_logic);
END aufgabe_eins;

ARCHITECTURE aufgabe_eins_arch OF aufgabe_eins IS
signal A : std_logic;
BEGIN
        P1      :process (B, C, D)
                BEGIN
                        A_OUT <= B or C and D
                        A <= B or C and D
                end process P1;

        P2      :process (A,C)
                BEGIN
                        E <= A and not C
                end process P2;
END aufgabe_eins_arch;
```



Aufgabe 2:

a) UND-Gatter
b) 1-Pegel gesteuertes D-Flip-Flop

Ü5: Kombinatorische und sequentielle Automatenmodelle in VHDL

Aufgabe 3:

Vorderflankengesteuertes D-Flip-Flop

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY dff IS
        PORT(D, CLK : IN std_logic;              -- D und Clock Signale
             Q,Q_not:  OUT std_logic);           -- Q und Q_not  Ausgangs-Signale
END dff;

ARCHITECTURE dff_arch OF dff IS
BEGIN
        process (CLK)
        BEGIN
                if (CLK'event and CLK = '1') then
                        Q <= D;
                        Q_not <= not D;
                end if;
        end process;
END dff_arch;
```

Ü5: Kombinatorische und sequentielle Automatenmodelle in VHDL

Aufgabe 4:

Teil a) Vorderflankengesteuertes D-Flip-Flop mit asynchronem Reset:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

 ENTITY dff IS
        PORT(D,CLK, RESET : IN std_logic;
                Q,Q_not:  OUT std_logic);
 END dff;

 ARCHITECTURE dff_arch OF dff IS
 BEGIN
        process(CLK, RESET)
        begin
                if (RESET='1') then
                        Q <= '0';
                        Q_not <= '1';
                elsif (CLK'event and CLK='1') then
                        Q <= D;
                        Q_not <= not D;
                end if;
        end process;
END dff_arch;
```

Teil b) Vorderflankengesteuertes D-Flip-Flop mit synchronem Reset:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

 ENTITY dff IS
        PORT(D,CLK, RESET : IN std_logic;
                Q,Q_not:  OUT std_logic);
 END dff;

 ARCHITECTURE dff_arch OF dff IS
 BEGIN
        process(CLK)
        begin
                if (CLK’event and CLK =’1’) then
                        if (RESET = '1') then
                                Q <= ’0’;
                                Q_not <= '1';
                        else
                                Q <= D;
                                Q_not <= not D;
                        end if;
                end if;
        end process;
END dff_arch;
```

Ü5: Kombinatorische und sequentielle Automatenmodelle in VHDL

Aufgabe 5:

Teil a) VHDL-Prozeß-Modell in Huffman-Normalform für den gegebenen endlichen Automaten:

```
entity FSM is
        port (CLK,RESET, X: in bit;
        Y1,Y2: out bit);
end FSM;

architecture BEH_2PR of FSM is
        type STATE is (S0, S1, S2);
        signal CURRENT_STATE, NEXT_STATE: STATE;

begin
        SEQ: process (CURRENT_STATE, X)
        begin
                case CURRENT_STATE is
                        when S0 =>
                                if (X = ‘0‘) then
                                        NEXT_STATE <= S0;
                                        Y1 <= ‘0‘; Y2 <= ‘1‘;
                                else
                                        NEXT_STAE <= S1;
                                        Y1 <= ‘1‘; Y2 <= ‘0‘;
                                end if;
                        when S1 =>
                                if X = ‘1‘ then
                                        NEXT_STATE <= S2;
                                        Y1 <= ‘0‘; Y2 <= ‘0‘;
                                else
                                        NEXT_STATE <= S1;
                                        Y1 <= ‘1‘; Y2 <= ‘0‘;
                                end if;
                        when S2 =>
                                if X = ‘1‘ then
                                        NEXT_STATE <= S0;
                                        Y1 <= ‘0‘; Y2 <= ‘0‘;
                                else
                                        NEXT_STATE <= S1;
                                        Y1 <= ‘1‘; Y2 <= ‘1‘;
                                end if;
                end case;
        end process;

        -- process to hold synchronous elements
        MEM: process (CLK,RESET)
        begin
                if RESET then
                        CURRENT_STATE <= S0;
                elsif CLK’event and CLK = ’1’ then
                        CURRENT_STATE <= NEXT_STATE;
                end if;
        end process;
end BEH_2PR;
```

Ü5: Kombinatorische und sequentielle Automatenmodelle in VHDL

Teil b) Änderung für einen asynchronen Automaten:

```
...
        -- process to hold synchronous elements
        MEM: process (NEXT_STATE,RESET)
        begin
                if RESET then
                        CURRENT_STATE <= S0;
                        NEXT_STATE <= S0;
                else
                        CURRENT_STATE <= NEXT_STATE;
                end if;
        end process;
...
```

Teil c) Ausgabe, wenn X dauerhaft auf '1' gesetzt wird:

Y1 = 100100100 ...
Y2 = 000000000 ...

Teil d) Berechnung des Automaten bei gegebener X-Folge und einem Takt von 5ns:

t [ns]	0	2	5	8	10	11	12	13	15	16	20	25	30
X	0	1	1	1	1	0	1	0	0	1	1	1	1
Y1	u	1	0	0	0	1	0	1	1	0	0	1	0
Y2	u	0	0	0	0	1	0	1	0	0	0	0	0
NS	u	S1	S2	S2	S0	S1	S0	S1	S1	S2	S0	S1	S2
CS	S0	S0	S1	S1	S2	S2	S2	S2	S1	S1	S2	S0	S1