Grundlagen der Technischen Informatik

Übung zur Vorlesung

Ü5: Kombinatorische und sequentielle Automatenmodelle in VHDL

Aufgabe 1:

Erstellen Sie ein VHDL-Prozeß-Modell mit 2 Prozessen, das die Funktionen

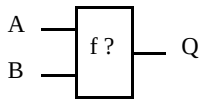
$$a = b \mid c \,\&\, d \qquad \text{und}$$
$$e = a \,\&\, \bar{c}$$

berechnet.

Aufgabe 2:

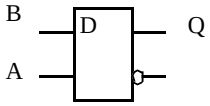
Erklären Sie die folgenden Abbildungen von VHDL-Fragmenten nach Schaltungen:

```
if A = '1' then
   Q <= B;
else
   Q <= '0';
end if;
```



Kombinatorische Schaltung

```
if A = '1' then
   Q <= B;
end if;
```



Sequentielle Schaltung (Latch)

Aufgabe 3:

Eine Vorderflanke vom Signal c beschreibt man in VHDL als (c'event and c = '1').

Beschreiben Sie ein Vorderflankengesteuerten D-Flip-Flop als mit einem VHDL-Prozeß!

Aufgabe 4:

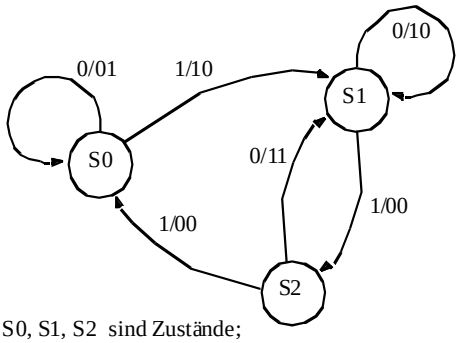
Ergänzen Sie die Beschreibung des Flip-Flops aus der Aufgabe 3.

Teil a) um ein asynchrones Reset,

Teil b) um ein synchrones Reset.

Aufgabe 5:

Teil a) Erstellen Sie ein VHDL-Prozeß-Modell in der Huffmann-Normalform-Kodierung, das den im Diagramm dargestellten endlichen Automaten beschreibt.



S0, S1, S2 sind Zustände;
0/01 ist für X = 0 und Y1 = 0, Y2 = 1.

Graphische Darstellung

Teil b) Was muss ich am VHDL-Code ändern, um einen asynchronen Automaten zu erhalten?

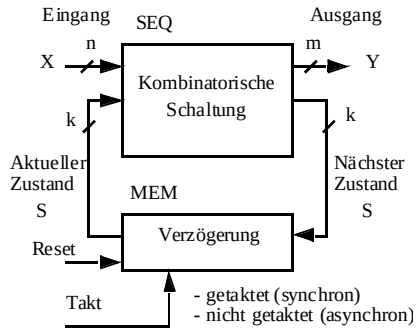
Teil c) Was passiert bei dem synchronen Automaten, wenn X <= '1' einmalig gesetzt wird?

Teil d) Was berechnet der Automat bei einem Takt von 5ns? Füllen Sie die Tabelle aus !

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
Y_1	u												
Y_2	u												
NS	u												
CS	S0												

Verschiedene Darstellungen von Automaten:

Modell A = (S,X,Y,δ,λ)
S = {S0, S1, S2}
X = {0,1} Y = $\{0,1\}^2$
δ : S x X -> S
λ : S x X-> Y



Implementierung Huffman-Normalform

- process realization of machine

```
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


    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;
```