

# **Binäre Bitmustererkennung über eine Zustandsmaschine in VHDL**

## **Inhaltsverzeichnis:**

Inhaltsverzeichnis: .....	1
Aufgabenstellung: .....	2
Code: .....	2
Ports:.....	2
Zustände:.....	2
Zustandsdiagramm:.....	3
VHDL-Code: .....	4
Test Bench: .....	7
Simulation:.....	10

## **Aufgabenstellung:**

Es ist eine Mustererkennung über eine Zustandsmaschine in VHDL zu entwickeln.  
Es geht dabei um die Erkennung einer bestimmten Bitfolge aus einem seriellen Datenstrom.

Es ist ein Zustandsdiagramm und eine Simulation zu machen.

Es ist an einem 3 Bit Ausgang jeweils anzugeben wie viele Bits des Musters gerade erkannt wurden und an einem 1 Bit Ausgang anzugeben wenn das Muster vollständig detektiert wurde.

Weiters ist darauf zu achten das es einen POR und einen Reset Eingang gibt.

**Code:** 10110

Zu erkennendes Muster.

### **Ports:**

clk\_in: Takteingang

serial\_in: Serieller Eingang

reset\_in: Reset

por\_in: Power On Reset

yet\_right\_out: 3 Bit Ausgang zur Anzeige der bereits erkannten Bits des Musters.

pattern\_detect\_out: Ausgang zur Anzeige des Erkennens des Musters

### **Zustände:**

Z0:

yet\_right\_out = 000

pattern\_detect\_out = 0

Z1:

yet\_right\_out = 001

pattern\_detect\_out = 0

Z2:

yet\_right\_out = 010

pattern\_detect\_out = 0

Z3:

yet\_right\_out = 011

pattern\_detect\_out = 0

Z4:

yet\_right\_out = 100

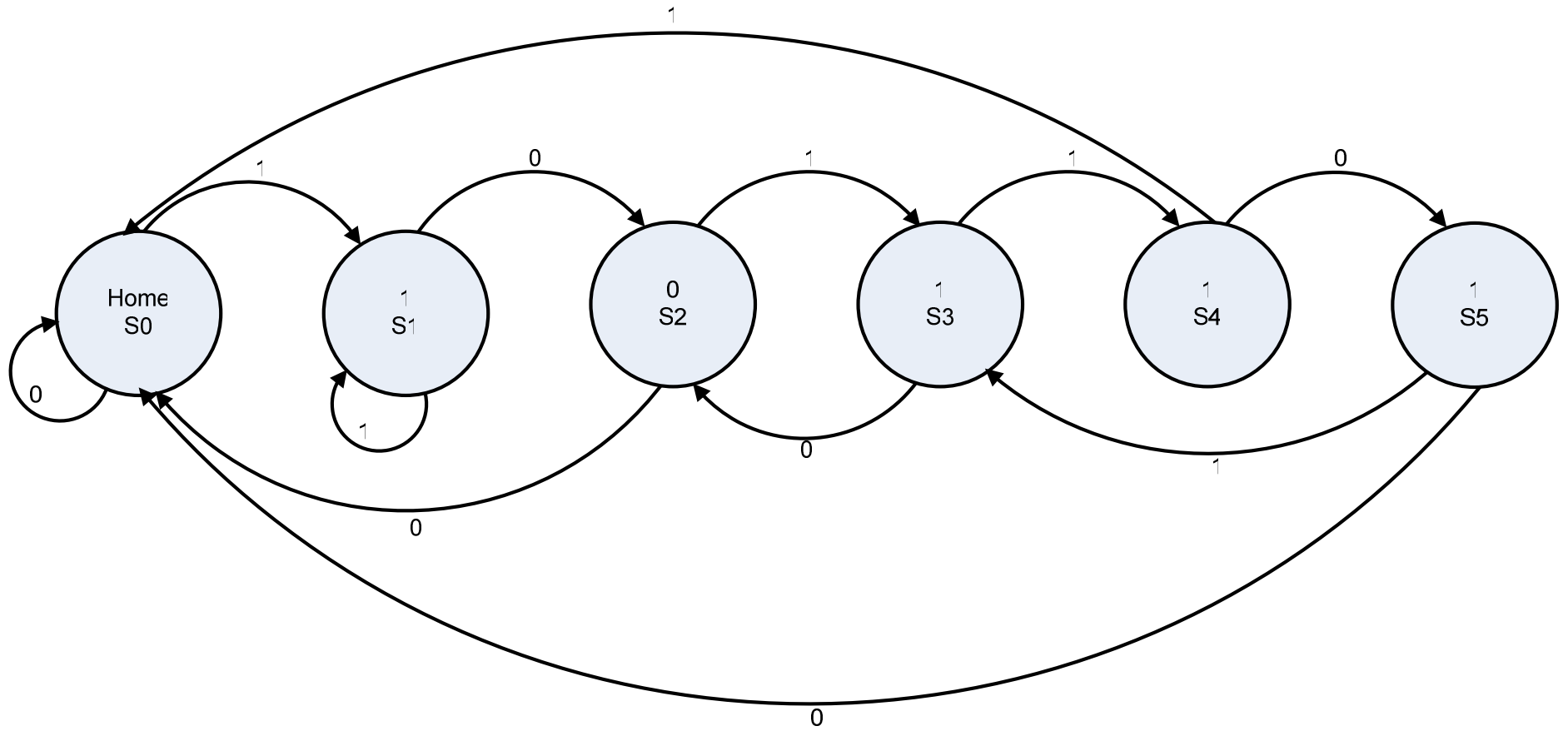
pattern\_detect\_out = 0

Z5:

yet\_right\_out = 101

pattern\_detect\_out = 1

**Zustandsdiagramm:**



## VHDL-Code:

```
-----  
-----  
-- Company: HTBLuVA Bulme Graz-Goesting  
-- Engineer: Bernhard Wintersperger  
--  
-- Create Date:      10:09:25 10/19/2007  
-- Design Name:  
-- Module Name:      statem_source - Behavioral  
-- Project Name:  
-- Target Devices:  
-- Tool versions:  
-- Description:  
--  
-- Dependencies:  
--  
-- Revision:  
-- Revision 0.01 - File Created  
-- Additional Comments:  
--  
-----  
-----  
library IEEE;  
use IEEE.STD_LOGIC_1164.ALL;  
use IEEE.STD_LOGIC_ARITH.ALL;  
use IEEE.STD_LOGIC_UNSIGNED.ALL;  
  
---- Uncomment the following library declaration if  
instantiating  
---- any Xilinx primitives in this code.  
--library UNISIM;  
--use UNISIM.VComponents.all;  
  
entity statem_source is  
    Port ( clk_in : in  STD_LOGIC;  
          serial_in : in  STD_LOGIC;  
          reset_in : in  STD_LOGIC;  
          por_in : in  STD_LOGIC;  
          yet_right_out : out  STD_LOGIC_VECTOR (2 downto 0);  
          pattern_detect_out : out  STD_LOGIC);  
end statem_source;  
  
architecture Behavioral of statem_source is  
  
type zustaende is (S0,S1,S2,S3,S4,S5); --Zustaende  
signal state : zustaende;  
  
begin
```

```

mustererkennung : process(clk_in, serial_in)
begin
if(clk_in'event and clk_in = '0') then --wenn flankenaenderung
                                an clk_in und clk=0
    if(por_in = '0' and reset_in = '0') then --wenn por_in
                                                und reset_in 0 ist dann...
    case state is
        when S0 =>
            yet_right_out <= "000"; --kein Bit erkannt
            pattern_detect_out <= '0';
            if(serial_in = '1') then
                state <= S1; --wenn serial_in 1 ist dann
                                state = S1
            else
                state <= S0; --wenn serial_in 0 ist dann
                                state = S0
            end if;
        when S1 =>
            yet_right_out <= "001"; --ein Bit erkannt
            pattern_detect_out <= '0';
            if(serial_in = '0') then
                state <= S2; --wenn serial_in 0 ist dann
                                state = S2
            else
                state <= S1; --wenn serial_in 1 ist dann
                                state = S1
            end if;
        when S2 =>
            yet_right_out <= "010"; --zwei Bits erkannt
            pattern_detect_out <= '0';
            if(serial_in = '1') then
                state <= S3; --wenn serial_in 1 ist dann
                                state = S3
            else
                state <= S0; --wenn serial_in 0 ist dann
                                state = S0
            end if;
        when S3 =>
            yet_right_out <= "011"; --drei Bits erkannt
            pattern_detect_out <= '0';
            if(serial_in = '1') then
                state <= S4; --wenn serial_in 1 ist dann
                                state = S4
            else
                state <= S2; --wenn serial_in 0 ist dann
                                state = S2
            end if;
        when S4 =>
            yet_right_out <= "100"; --vier Bits erkannt
            pattern_detect_out <= '0';
            if(serial_in = '0') then

```

```

        state <= S5;  --wenn serial_in 0 ist dann
                    state = S5
    else
        state <= S0;  --wenn serial_in 1 ist dann
                    state = S0
    end if;
when S5 =>
    yet_right_out <= "101";  --alle Bits erkannt
    pattern_detect_out <= '1';  --Muster erkannt
    if (serial_in = '0') then
        state <= S0;  --wenn serial_in 0 ist dann
                    state = S0
    else
        state <= S3;  --wenn serial_in 1 ist dann
                    state = S3
    end if;
end case;
else
    state <= S0;  --wenn por_in und reset_in 1 ist
                dann state = S0
end if;
end if;

end process;

end Behavioral;

```

## Test Bench:

```
-----  
-----  
-- Company: HTBLuVA Bulme Graz-Goesting  
-- Engineer: Bernhard Wintersperger  
--  
-- Create Date: 12:08:33 10/19/2007  
-- Design Name: statem_source  
-- Module Name: statem_tb.vhd  
-- Project Name: statemachine  
-- Target Device:  
-- Tool versions:  
-- Description:  
--  
-- VHDL Test Bench Created by ISE for module: statem_source  
--  
-- Dependencies:  
--  
-- Revision:  
-- Revision 0.01 - File Created  
-- Additional Comments:  
--  
-- Notes:  
-- This testbench has been automatically generated using types  
std_logic and  
-- std_logic_vector for the ports of the unit under test.  
Xilinx recommends  
-- that these types always be used for the top-level I/O of a  
design in order  
-- to guarantee that the testbench will bind correctly to the  
post-implementation  
-- simulation model.  
-----  
-----  
LIBRARY ieee;  
USE ieee.std_logic_1164.ALL;  
USE ieee.std_logic_unsigned.all;  
USE ieee.numeric_std.ALL;  
  
ENTITY statem_tb_vhd IS  
END statem_tb_vhd;  
  
ARCHITECTURE behavior OF statem_tb_vhd IS  
  
    -- Component Declaration for the Unit Under Test (UUT)  
    COMPONENT statem_source  
    PORT(  
        clk_in : IN std_logic;  
        serial_in : IN std_logic;  
        reset_in : IN std_logic;
```

```

        por_in : IN std_logic;
        yet_right_out : OUT std_logic_vector(2 downto 0);
        pattern_detect_out : OUT std_logic
    );
END COMPONENT;

--Inputs
SIGNAL clk_in : std_logic := '0';
SIGNAL serial_in : std_logic := '0';
SIGNAL reset_in : std_logic := '0';
SIGNAL por_in : std_logic := '0';

--Outputs
SIGNAL yet_right_out : std_logic_vector(2 downto 0);
SIGNAL pattern_detect_out : std_logic;

BEGIN

    -- Instantiate the Unit Under Test (UUT)
    uut: statem_source PORT MAP(
        clk_in => clk_in,
        serial_in => serial_in,
        reset_in => reset_in,
        por_in => por_in,
        yet_right_out => yet_right_out,
        pattern_detect_out => pattern_detect_out
    );

    clk : PROCESS (clk_in) --prozess um clock zu generieren
    (läuft immer parallel mit)
    BEGIN
        IF clk_in'EVENT AND clk_in = '1' then
            clk_in <= '0' AFTER 10 NS;
        ELSE
            clk_in <= '1' AFTER 10 NS;
        END IF;
    END PROCESS;

    tb : PROCESS
    BEGIN
        -- Wait 100 ns for global reset to finish
        wait for 100 ns;

        por_in <= '1' after 1ns,          --Power on Reset
            '0' after 18ns;

        reset_in <= '0' after 1ns,      --Reset Knoepfchen
            '1' after 570ns,
            '0' after 590ns;
    END PROCESS;

```

```

serial_in <= '0' after 1ns, --S0
    '1' after 18ns,      --S1
    '0' after 38ns,     --S2
    '1' after 58ns,     --S3
    '1' after 78ns,     --S4
    '0' after 98ns,     --S5, Muster fertig
    '0' after 118ns,    --S0
    '1' after 138ns,    --S1
    '0' after 158ns,    --S2
    '1' after 178ns,    --S3
    '1' after 198ns,    --S4
    '0' after 218ns,    --S5, Muster fertig
    '1' after 238ns,    --S3
    '1' after 258ns,    --S4
    '0' after 278ns,    --S5, Muster fertig
    '0' after 298ns,    --S0
    '1' after 318ns,    --S1
    '1' after 338ns,    --S1
    '0' after 358ns,    --S2
    '0' after 378ns,    --S0
    '1' after 398ns,    --S1
    '0' after 418ns,    --S2
    '1' after 438ns,    --S3
    '0' after 458ns,    --S2
    '1' after 478ns,    --S3
    '1' after 498ns,    --S4
    '0' after 518ns,    --S5, Muster fertig
    '1' after 538ns,    --S3
    '0' after 558ns,    --S2
    '1' after 578ns,    --S3
    '0' after 598ns;    --S2

    wait; -- will wait forever
END PROCESS;

END;
```

## Simulation:

