

Aiken Counter:**Aufgabenstellung:**

Ein synchroner Zähler im Aiken-Code mit 3 Eingängen und 4 Ausgängen. Es soll einen enable Eingang, einen enable Ausgang und einen Reset Eingang geben.

Eingänge:

reset_in
clock_in
enable_in

Ausgänge:

aiken_out
enable_out

Zählertabelle:

<i>Dez</i>	<i>Aiken</i>
0	0000
1	0001
2	0010
3	0011
4	0100
5	1011
6	1100
7	1101
8	1110
9	1111
Fehler	0101

VHDL-Code:

```
-----  
-- Company: HTBLuVA Bulme Graz-Goesting  
-- Engineer: Bernhard Wintersperger  
--  
-- Create Date: 10:04:07 10/05/2007  
-- Design Name:  
-- Module Name: AikCount_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 AikCount_source is  
  Port ( reset_in : in STD_LOGIC;  
        clock_in : in STD_LOGIC;  
        enable_in : in STD_LOGIC;  
        aiken_out : out STD_LOGIC_VECTOR (3 downto 0);  
        enable_out : out STD_LOGIC);  
end AikCount_source;  
  
architecture Behavioral of AikCount_source is  
  
    signal code : integer range 0 to 9; --variable um Dez hochzuzählen  
  
begin
```

```

P1: process( clock_in )
begin
  if( clock_in'event and clock_in = '1' ) then --wenn an clock_in flanke auf 1 geht...
    if(reset_in = '1') then --wenn reset_in 1 ist dann zähler rücksetzen
      code <= 0;
    else
      if(code = 9) then --wenn code auf maximum ist dann rücks.
        code <= 0;
      else
        if (enable_in = '1') then --wenn reset_in = 0 und
          code < 9 und enable_in = 1 dann code+1
          code <= code + 1;
        end if;
      end if;
    end if;
  end if;
end process;

enable_out <= '1' when code = 9 else '0'; --wenn code auf maximum, enable out = 1

aiken_out <= --zuweisungstabelle für den ausgang
"0000" WHEN code = 0 ELSE
"0001" WHEN code = 1 ELSE
"0010" WHEN code = 2 ELSE
"0011" WHEN code = 3 ELSE
"0100" WHEN code = 4 ELSE
"1011" WHEN code = 5 ELSE
"1100" WHEN code = 6 ELSE
"1101" WHEN code = 7 ELSE
"1110" WHEN code = 8 ELSE
"1111" WHEN code = 9 ELSE
"0101";

end Behavioral;

```

Test Bench:

```
-----  
-- Company: HTBLuVA Bulme Graz-Goesting  
-- Engineer: Bernhard Wintersperger  
--  
-- Create Date: 11:01:09 10/05/2007  
-- Design Name: AikCount_source  
-- Module Name: AikCount_TB.vhd  
-- Project Name: AikCount  
-- Target Device:  
-- Tool versions:  
-- Description:  
--  
-- VHDL Test Bench Created by ISE for module: AikCount_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 AikCount_TB_vhd IS  
END AikCount_TB_vhd;
```

```
ARCHITECTURE behavior OF AikCount_TB_vhd IS
```

```
    -- Component Declaration for the Unit Under Test (UUT)  
    COMPONENT AikCount_source  
    PORT(  
        reset_in : IN std_logic;  
        clock_in : IN std_logic;  
        enable_in : IN std_logic;  
        aiken_out : OUT std_logic_vector(3 downto 0);  
        enable_out : OUT std_logic  
    );  
    END COMPONENT;
```

```

--Inputs
SIGNAL reset_in : std_logic := '0';
SIGNAL clock_in : std_logic := '0';
SIGNAL enable_in : std_logic := '0';

--Outputs
SIGNAL aiken_out : std_logic_vector(3 downto 0);
SIGNAL enable_out : std_logic;

BEGIN

-- Instantiate the Unit Under Test (UUT)
uut: AikCount_source PORT MAP(
    reset_in => reset_in,
    clock_in => clock_in,
    enable_in => enable_in,
    aiken_out => aiken_out,
    enable_out => enable_out
);

clk : PROCESS (clock_in) --prozess um clock zu generieren (läuft immer parallel mit)
BEGIN
    IF clock_in'EVENT AND clock_in = '1' then
        clock_in <= '0' AFTER 20 NS;
    ELSE
        clock_in <= '1' AFTER 20 NS;
    END IF;
END PROCESS;

tb : PROCESS
BEGIN

    -- Wait 100 ns for global reset to finish
    wait for 100 ns;

enable_in <= '1' after 5 ns,      --setze enable auf 1
        '0' after 630 ns,      --setze enable auf 0
        '1' after 700 ns;      --setze enable auf 0
reset_in <= '1' after 230 ns,    --setze reset auf 1
        '0' after 250ns;       --setze reset auf 0

    wait; -- will wait forever
END PROCESS;

END;
```

Simulation:

