Counter 6bit mit Testbench

Counter Code

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;

entity counter_6bit is
port (
  res_n            : in  std_logic;
  clk_i            : in  std_logic;
  enable_i         : in  std_logic;
  up_down_i        : in  std_logic;
  count_o          : out unsigned(5 downto 0)
  );
end entity;

architecture rtl of counter_6bit is

signal count : unsigned(5 downto 0);
begin
  count_p: process(res_n, clk_i)
  begin
    if (res_n = '0') then
     count <= "110000";
    elsif (clk_i'event and clk_i = '1') then
      if (enable_i = '1') then

      if (count = 64) then
          count <= (others => '0');
        else
          if (up_down_i = '1') then
            count <= count + 1;
          else
            count <= count - 1;
          end if;
        end if;
      end if;
    end if;
  end process;
  count_o <= count;
end rtl;

configuration counter_6bit_conf of counter_6bit is
  for rtl
  end for;
end counter_6bit_conf;
```

Testbench

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

ENTITY counter_6bit_tb IS
END counter_6bit_tb;

ARCHITECTURE behavior OF counter_6bit_tb IS
  COMPONENT counter_6bit
  PORT(
    res_n            : in  std_logic;
    clk_i            : in  std_logic;
    enable_i         : in  std_logic;
    up_down_i        : in  std_logic;
    count_o          : out unsigned(5 downto 0)
    );
  END COMPONENT;

   signal res_n           :  std_logic:='1';
   signal clk             :  std_logic;
   signal enable          :  std_logic:='0';
   signal up_down         :  std_logic:='0';
   signal count_o         :  unsigned(5 downto 0);
   signal clk_signal      :  std_logic;
   constant clk_period    :  time :=100 us;
   signal count           : unsigned(5 downto 0);

BEGIN
  uut: counter_6bit
  PORT MAP(
    res_n => res_n,
    clk_i => clk,
    enable_i => enable,
    up_down_i => up_down,
    count_o => count_o
  );

  generate_clock: process
  begin
     clk_signal <= '1';
     wait for clk_period * 0.5;
     clk_signal <= '0';
     wait for clk_period * 0.5;
  end process ;
  clk <= clk_signal;

  tb : PROCESS
  BEGIN
    wait for 500 us;
    res_n <= '0';
```

```
    wait for 10 us;
    enable <= '1';
    res_n <= '1';

    wait for 1 ms;
    up_down <= '1';

    wait for 1 ms;
    enable <= '0';

    wait for 100 us;
    assert false report "Simulation ended" severity FAILURE;
   END PROCESS;
END behavior;
```