Codesniffer mit Testbench

Codesniffer Code

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

entity codesniffer is
  port (
    input         : in  std_logic_vector(1 downto 0);
    trigger       : out  std_logic;
    clk_i         : in  std_logic;
    res_n         : in  std_logic;
    enable_i      : in  std_logic
    );
end entity;

architecture rtl of codesniffer is
  type state_def is (z0, z1, z2, z3);
  attribute enum_decoding: string;
  attribute enum_decoding of state_def: type is "01 11 10 10";
  signal state, next_state : state_def;

  begin

  state_reg : process (res_n, clk_i)
  begin
    if (res_n = '0') then
        state <= z0;
    elsif (clk_i'event and clk_i = '1') then
      if enable_i = '1' then
        state <= next_state;
      end if;
    end if;
  end process;

  -- implementation of the state_machine function
  state_func : process(state)
  begin
    case state is
        when z0 =>
          if (input="01") then
            next_state <= z1;
          else
            next_state <= z0;
          end if;
          trigger <= '0';
```

```
        when z1 =>
          if (input="11") then
            next_state <= z2;
          elsif (input ="01") then
            next_state <= z1;
          else
            next_state <= z0;
          end if;
          trigger <= '0';

        when z2 =>
          if (input="10") then
            next_state <= z3;
          elsif (input ="01") then
            next_state <= z1;
          else
            next_state <= z0;
          end if;
          trigger <= '0';

        when z3 =>
          if (input = "01") then
            next_state <= z1;
          else
            next_state <= z0;
          end if;
          trigger <= '1';

        when others =>
          next_state <= z0;
      end case;
  end process; -- state function
end rtl; -- architecture

configuration codesniffer_conf of codesniffer is
  for rtl
  end for;
end codesniffer_conf;
```

Testbench

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

ENTITY codesinffer_tb IS
END codesinffer_tb;

ARCHITECTURE behavior OF codesinffer_tb IS
  COMPONENT codesinffer
  PORT(
   input        : in  std_logic_vector(1 downto 0);
```

```
    trigger      : out std_logic;
     res_n       : in  std_logic;
     clk_i       : in  std_logic;
     enable_i    : in  std_logic
     );
  END COMPONENT;

  SIGNAL res_n          :std_logic := '1';
  SIGNAL clk            :std_logic;
  SIGNAL enable         :std_logic;
  SIGNAL clk_signal     :std_logic;
  SIGNAL input          :std_logic_vector(1 downto 0);
  SIGNAL clk_period     :time := 2 ms;
  SIGNAL trigger        :std_logic;

BEGIN
  uut: codesinffer
    PORT MAP(
    res_n => res_n,
    clk_i => clk,
    enable_i => enable,
    input => input,
   trigger => trigger
  );

  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;

  control_sig : PROCESS
  begin
    wait for 500 us;
    res_n <= '1';

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

    wait for 2.5 ms;
    res_n<='1';

    wait for 2 ms;
    enable <= '1';

    wait for 150 ms;
    assert false report "End of simulation" severity FAILURE;
  END PROCESS;
```

```
END behavior;
```