State-Maschinen

Code

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

entity state_machine is
  port (
    clk_i         : in  std_logic;
    res_n         : in  std_logic;
    enable_i      : in  std_logic;
    output_o      : out std_logic_vector(3 downto 0)
    );
end entity;

architecture rtl of state_machine is
  type state_def is (init_state, state1, state2, state3, state4);

  attribute enum_coding: string;
  attribute enum_coding of state_def: type is "0000 0001 0011 0010 0110"; -- Gray code

  signal state      : state_def;
  signal next_state : state_def;
  signal hi         : std_logic;
  signal low        : std_logic;

begin
hi  <= '1';
low <= '0';

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

state_func : process(state)
begin
  case state is
       when init_state =>
          next_state <= state1;
          output_o <= "0000";
```

```
        when state1 =>
          next_state <= state2;
          output_o <= "0001";

        when state2 =>
          next_state <= state3;
          output_o <= "0010";

        when state3 =>
          next_state <= state4;
          output_o <= "0011";

        when state4 =>
          next_state <= init_state;
          output_o <= "0100";

        when others =>
          next_state <= init_state;
          output_o <= "0000";
    end case;
end process;
end rtl;

configuration state_machine_conf of state_machine is
  for rtl
  end for;
end state_machine_conf;
```

Testbench

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

entity state_machine_tb is
end entity state_machine_tb;

architecture beh of state_machine_tb is
  component state_machine
  port(
    res_n             : in  std_logic;
    clk_i             : in  std_logic;
    enable_i          : in  std_logic;
    output_o          : out std_logic_vector(3 downto 0)
   );
  end component;

  component clock_gen
  port (
    reset_length_i : in  integer;        --:= 3;
    clk_period_i   : in  time;           --:= 40 ns;
    clk            : out std_logic;
    reset_n        : out std_logic
    );
  end component;

  signal res_n        : std_logic;
  signal clk          : std_logic;
  signal enable       : std_logic := '0';
  signal reset_length : integer := 3;
  signal clk_period   : time    := 10 us;
  signal output       : std_logic_vector(3 downto 0);

begin
clk_gen_i: clock_gen
    port map (
      reset_length_i  => reset_length,
      clk_period_i    => clk_period,
      clk             => clk,
      reset_n         => res_n
    );

dut: state_machine
  port map (
    res_n => res_n,
    clk_i => clk,
    enable_i => enable,
    output_o => output
  );

control_tb: process
begin
  wait for 10 us;
  enable <= '0';

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

  wait for 120 us;
  enable <= '0';

  wait for 20 us;
  wait;
end process;
end beh;

configuration state_machine_tb_conf of state_machine_tb is
  for beh
    for dut: state_machine
      use configuration work.state_machine_conf;
    end for;

    for clk_gen_i: clock_gen
      use configuration work.clock_gen_rtl_conf;
    end for;
  end for;
end state_machine_tb_conf;
```