Practical No.04: Mod 25

CODE

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity mod25 is
   Port (
      rst    : in  STD_LOGIC;
      pr     : in  STD_LOGIC;
      clk    : in  STD_LOGIC;
      dir    : in  STD_LOGIC;
      Q      : out STD_LOGIC_VECTOR (4 downto 0);
      clk_div : out STD_LOGIC             -- Added output
   );
end mod25;

architecture mod25_arch of mod25 is
   signal Qtemp   : STD_LOGIC_VECTOR (4 downto 0) := "00000";
   signal counter : STD_LOGIC_VECTOR (30 downto 0) := (others => '0');
   signal clk_div_int : STD_LOGIC;
begin
```

```
-- Clock divider
process(clk)
begin
   if rising_edge(clk) then
      counter <= counter + 1;
   end if;
end process;

clk_div_int <= counter(25);  -- Internal divided clock
clk_div <= clk_div_int;      -- Output it to testbench

-- MOD-25 counter
process(rst, pr, clk_div_int, dir)
begin
   if rst = '1' then
      Qtemp <= (others => '0');
   elsif pr = '1' then
      Qtemp <= (others => '1');
   elsif falling_edge(clk_div_int) then
      if dir = '1' then
         if Qtemp < "11000" then    -- 24 decimal
            Qtemp <= Qtemp + 1;
         else
            Qtemp <= (others => '0');
```

```
            end if;
         else
            if Qtemp > "00111" then    -- 7 decimal
               Qtemp <= Qtemp - 1;
            else
               Qtemp <= (others => '1');
            end if;
         end if;
      end if;
   end process;

   Q <= Qtemp;
end mod25_arch;
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity mod25 is
   Port (
      rst    : in  STD_LOGIC;
      pr     : in  STD_LOGIC;
      clk    : in  STD_LOGIC;
      dir    : in  STD_LOGIC;
      Q      : out STD_LOGIC_VECTOR (4 downto 0);
```

```
        clk_div : out STD_LOGIC              -- Added output
    );
end mod25;

architecture mod25_arch of mod25 is
    signal Qtemp   : STD_LOGIC_VECTOR (4 downto 0) := "00000";
    signal counter : STD_LOGIC_VECTOR (30 downto 0) := (others => '0');
    signal clk_div_int : STD_LOGIC;
begin

    -- Clock divider
    process(clk)
    begin
        if rising_edge(clk) then
            counter <= counter + 1;
        end if;
    end process;

    clk_div_int <= counter(25);  -- Internal divided clock
    clk_div <= clk_div_int;      -- Output it to testbench

    -- MOD-25 counter
    process(rst, pr, clk_div_int, dir)
    begin
```

```
        if rst = '1' then
            Qtemp <= (others => '0');
        elsif pr = '1' then
            Qtemp <= (others => '1');
        elsif falling_edge(clk_div_int) then
            if dir = '1' then
                if Qtemp < "11000" then    -- 24 decimal
                    Qtemp <= Qtemp + 1;
                else
                    Qtemp <= (others => '0');
                end if;
            else
                if Qtemp > "00111" then    -- 7 decimal
                    Qtemp <= Qtemp - 1;
                else
                    Qtemp <= (others => '1');
                end if;
            end if;
        end if;
    end process;

    Q <= Qtemp;
end mod25_arch;
```

TB-Code

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

entity mod25_tb is
end mod25_tb;

architecture behavior of mod25_tb is

   -- Component Declaration for the Unit Under Test (UUT)
   component mod25
      port(
         rst     : in  std_logic;
         pr      : in  std_logic;
         clk     : in  std_logic;
         dir     : in  std_logic;
         Q       : out std_logic_vector(4 downto 0);
         clk_div : out std_logic        -- Must match entity!
      );
   end component;

   -- Inputs
   signal rst : std_logic := '0';
```

```
    signal pr  : std_logic := '0';
    signal clk : std_logic := '0';
    signal dir : std_logic := '0';

    -- Outputs
    signal Q      : std_logic_vector(4 downto 0);
    signal clk_div : std_logic;

    -- Clock period
    constant clk_period : time := 10 ns;

begin
    -- Instantiate UUT
    uut: mod25 port map (
        rst     => rst,
        pr      => pr,
        clk     => clk,
        dir     => dir,
        Q       => Q,
        clk_div => clk_div
    );

    ----------------------------------------------------------------
    -- Clock generation
```

```
-----------------------------------------------------------------
clk_process: process
begin
   clk <= '0';
   wait for clk_period / 2;
   clk <= '1';
   wait for clk_period / 2;
end process;

-----------------------------------------------------------------
-- Stimulus
-----------------------------------------------------------------
stim_proc_dir: process
begin
   dir <= not dir;
   wait for 320 ns;
end process;

stim_proc_rst: process
begin
   wait for 680 ns;
   rst <= '1';
   wait for 40 ns;
   rst <= '0';
```

```
    wait;
  end process;

  stim_proc_pr: process
  begin
    wait for 750 ns;
    pr <= '1';
    wait for 40 ns;
    pr <= '0';
    wait;
  end process;

end behavior;
```

.xdc

```
set_property IOSTANDARD LVCMOS33 [get_ports {Q[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Q[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Q[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Q[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Q[0]}]
set_property PACKAGE_PIN W5 [get_ports clk]
set_property PACKAGE_PIN U16 [get_ports clk_div]
set_property PACKAGE_PIN T1 [get_ports dir]
set_property PACKAGE_PIN U1 [get_ports pr]
```

```
set_property PACKAGE_PIN W2 [get_ports rst]
set_property PACKAGE_PIN P1 [get_ports {Q[4]}]
set_property PACKAGE_PIN N3 [get_ports {Q[3]}]
set_property PACKAGE_PIN P3 [get_ports {Q[2]}]
set_property PACKAGE_PIN U3 [get_ports {Q[1]}]
set_property PACKAGE_PIN W3 [get_ports {Q[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports clk]
set_property IOSTANDARD LVCMOS33 [get_ports clk_div]
set_property IOSTANDARD LVCMOS33 [get_ports dir]
set_property IOSTANDARD LVCMOS33 [get_ports pr]
set_property IOSTANDARD LVCMOS33 [get_ports rst]
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets pr]
```

Port Names	**FPGA Pins**
clk	W5
clk div	U16
dir(Direction)	T1
PR(preset)	U1
RST	W2
Q4	P1
Q3	N3
Q2	P3
Q1	V3
Q0	W3