Digital Systems - 4

Shift Registers. Counters

Pere Palà Schönwälder

iTIC http://itic.cat

February 2025

Shift Register

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

entity shift_reg is
port( serial_in, clk, shift_ena  : in std_logic;
      serial_out                 : out std_logic);
end shift_reg;

architecture behav of shift_reg is
  signal q: std_logic_vector(3 downto 0);
begin
  -- clock process
  process(clk)
  begin
    if rising_edge(clk) then
       if shift_ena = '1' then
          q(3 downto 1) <= q(2 downto 0);
            q(0) <= serial_in;
       end if;
     end if;
  end process;
  -- concurrent assignment
  serial_out <= q(3);
end behav;
```

Shift Register Test Bench 1/3

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

entity shift_reg_tb is
end;

architecture behav of shift_reg_tb is

  signal serial_in  : std_logic;
  signal clk        : std_logic;
  signal shift_ena  : std_logic;
  signal serial_out : std_logic;
begin
  dut : entity work.shift_reg     --Direct instantiation here
  port map(
    serial_in  => serial_in,
    clk        => clk,
    shift_ena  => shift_ena,
    serial_out => serial_out);

  -- architecture continues...
```

Shift Register Test Bench 2/3

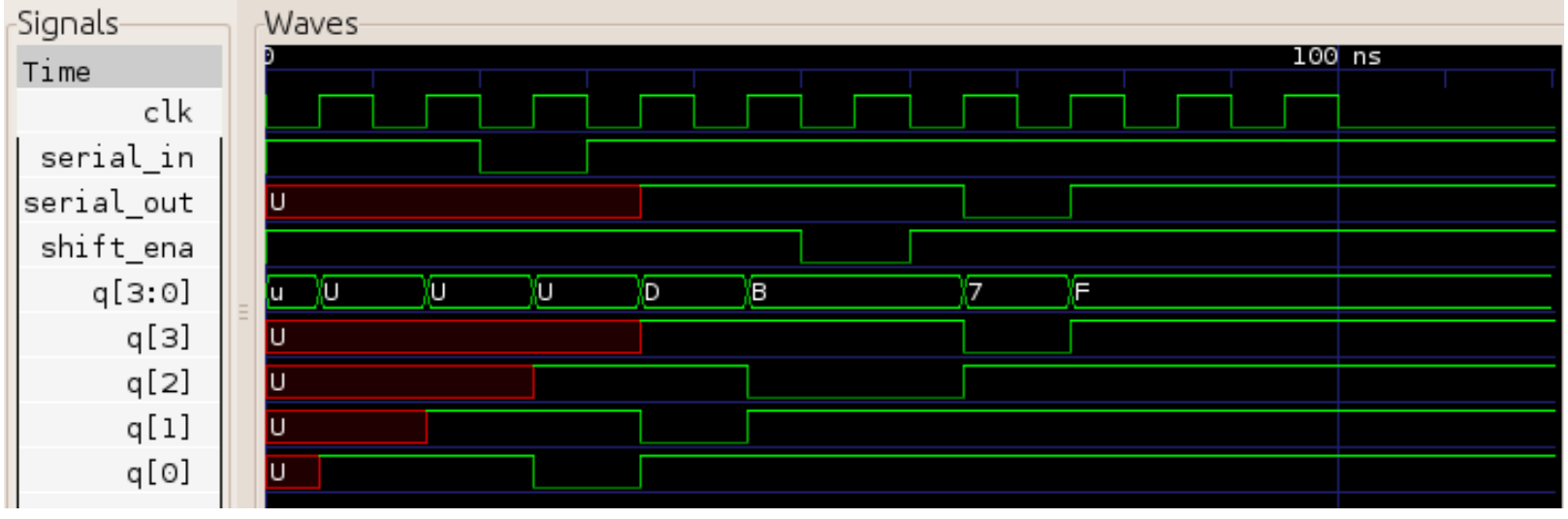
```
  clk_process : process
  begin  --the clock process
    clk <= '0';
    wait for 5 ns;
    for i in 1 to 20 loop
      clk <= not clk;
      wait for 5 ns;
    end loop;
    wait;
  end process clk_process;

  shift_ena <= '1',                    --one way to set signal
               '0' after 50 ns, '1' after 60 ns;

  stim_process : process
  begin  --another way
    serial_in <= '1';
    wait for 20 ns;
    serial_in <= '0';
    wait for 10 ns;
    serial_in <= '1';
    wait for 90 ns;
    wait;                              --wait forever
  end process stim_process;
end behav;
```

Shift Register Test Bench 3/3

```
$ ghdl -a sr.vhd
$ ghdl -a sr_tb.vhd
$ ghdl -e sr_tb
$ ghdl -r sr_tb --vcd=sr.vcd
$ gtkwave sr.vcd
```



Counter

```
library ieee; use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
entity counter is
port( clk, reset : in  std_logic;
      tc         : out std_logic);
end counter;

architecture behav of counter is
  signal q: unsigned(3 downto 0);
begin
  process(clk) -- clock process
  begin
    if rising_edge(clk) then
      if reset = '1' then
        q <= "0000";
      elsif q = 9 then
        q <= "0000";
      else
        q <= q + 1;
      end if;
    end if;
  end process;
  tc <= '1' when q = "0000" else '0';
end behav;
```

Counter Test Bench 1/2

```
...
begin
  dut : entity counter.shift_reg ...

  clk_process: process
  begin              --the clock process
    clk <= '0';
    wait for 5 ns;
    for i in 1 to 40 loop
       clk <= not clk;
       wait for 5 ns;
    end loop;
    wait;
  end process clk_process;

  reset <='1', '0' after 10 ns;
end behav ;
```

Counter Test Bench 2/2

```
$ ghdl -a counter.vhd
$ ghdl -a counter_tb.vhd
$ ghdl -e counter_tb
$ ghdl -r counter_tb --vcd=counter.vcd
$ gtkwave counter.vcd
```

