

Counters

Last updated 9/16/20

Counters

These slides review the design for several types of counters

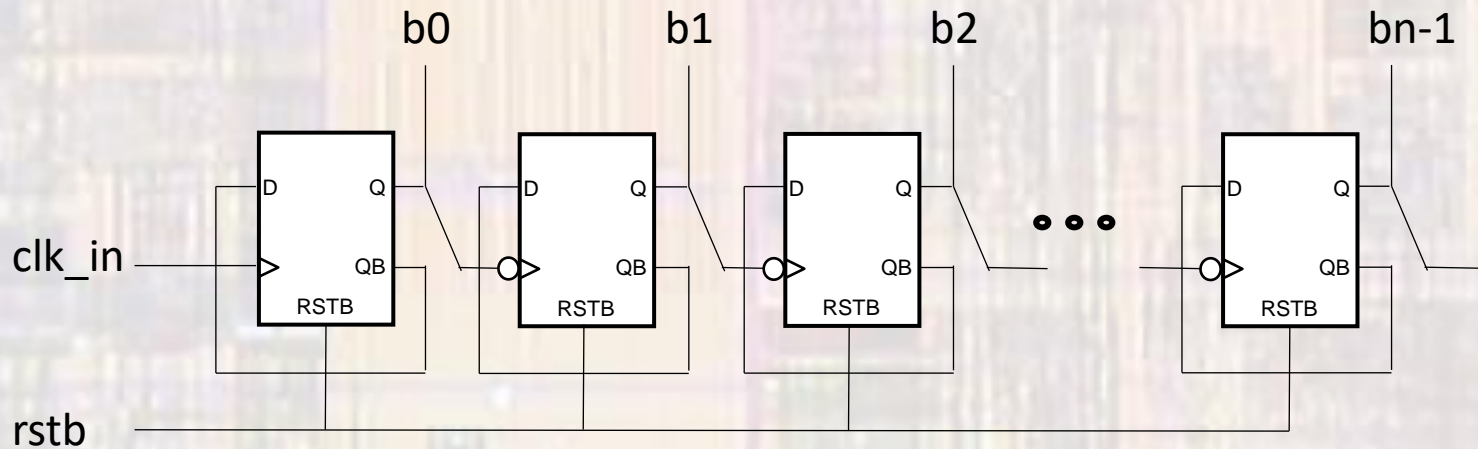
Upon completion: You should be able to design various types of counters
(up/dn/signed/unsigned/mod)

Counters

- Up Counter

- Count in binary

- 0000 → 0001 → 0010 → 0011 → 0100 ... 1111 → 0000 ...



What's wrong with this solution

Counters

- Counter - n bit - unsigned

```
-----  
-- counter_unsigned_n_bit.vhdl  
-- created 2/29/17  
-- tj  
--  
-- rev 0  
-----  
--  
-- n bit unsigned up-counter example  
-----  
--  
-- Inputs: rstb, clk  
-- Outputs: cnt  
-----  
library ieee;  
use ieee.std_logic_1164.all;  
use ieee.numeric_std.all;  
entity counter_unsigned_n_bit is  
  generic(  
    n: natural := 4  
  );  
  port (  
    i_clk : in std_logic;  
    i_rstb : in std_logic;  
    o_cnt : out std_logic_vector(n-1 downto 0)  
  );  
end entity;
```

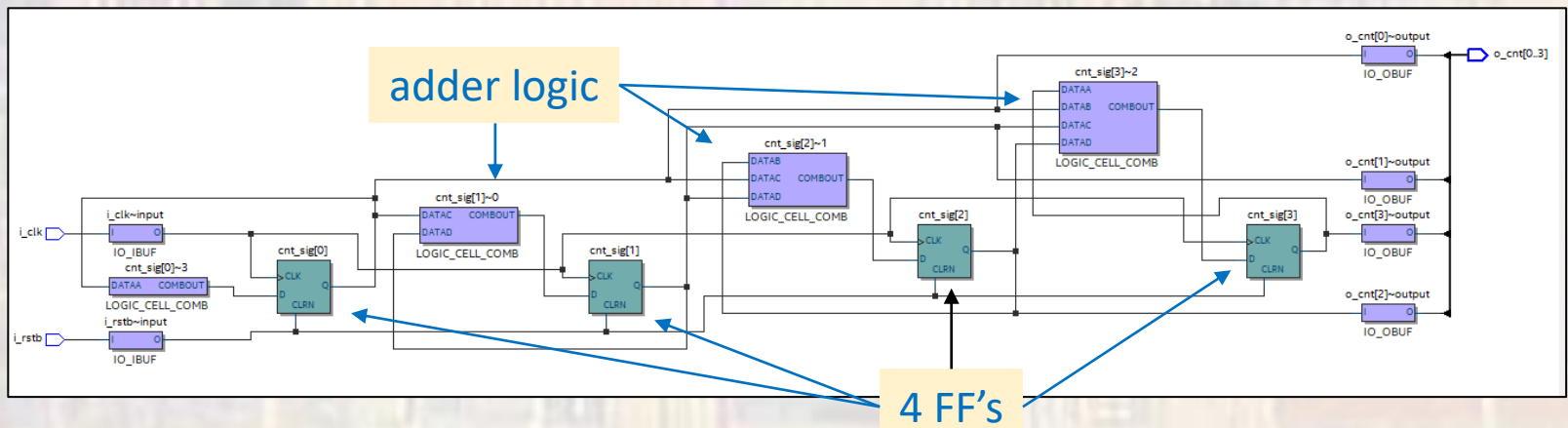
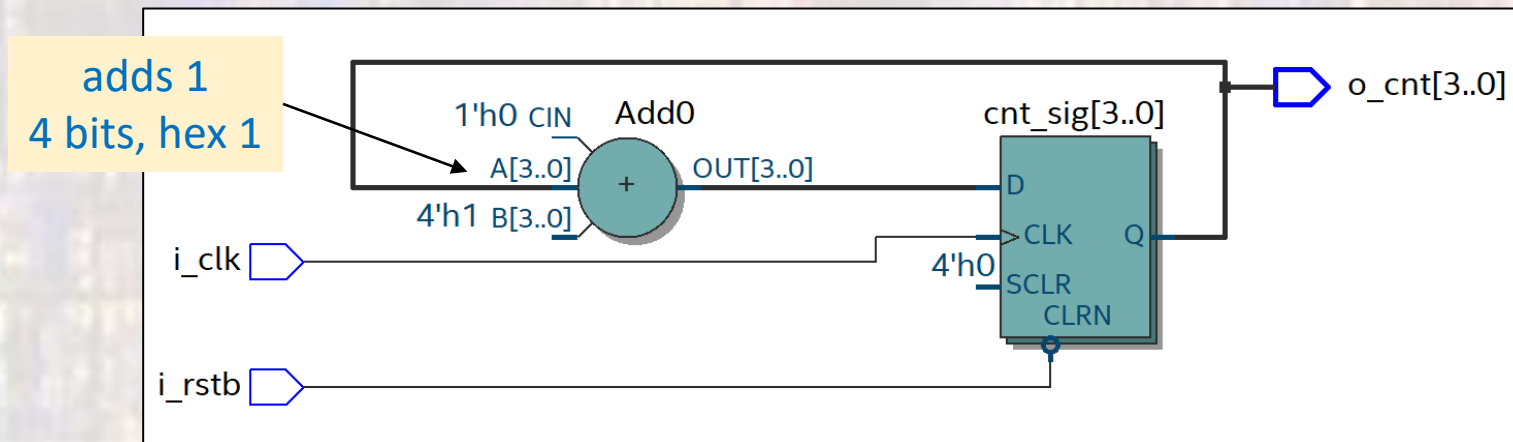
```
architecture behavioral of counter_unsigned_n_bit is  
  --  
  -- internal signals  
  --  
  signal cnt_sig: unsigned(n-1 downto 0);  
begin  
  count: process(i_clk, i_rstb)  
    begin  
      -- reset  
      --  
      if (i_rstb = '0') then  
        cnt_sig <= (others => '0');  
      -- rising clk edge  
      --  
      elsif (rising_edge(i_clk)) then  
        cnt_sig <= cnt_sig + 1;  
      end if;  
    end process;  
  --  
  -- output logic  
  --  
  o_cnt <= std_logic_vector(cnt_sig);  
end behavioral;
```

why?
2 - reasons

cast

Counters

- Counter - n bit – unsigned – (default)



Counters

- Counter - n bit – unsigned (6 bit version)

```
-----  
-- counter_unsigned_n_bit_tb.vhdl  
--  
-- created: 1/26/18  
-- by: johnsontimoj  
-- rev: 0  
--  
-- testbench for n bit counter  
-- of counter_unsigned_n_bit.vhdl  
-- brute force implementation  
-----  
library ieee;  
use ieee.std_logic_1164.all;  
entity counter_unsigned_n_bit_tb is  
  generic(  
    N: natural := 6  
  );  
  -- no ports - testbench  
end entity;
```

```
architecture testbench of counter_unsigned_n_bit_tb is  
  signal CLK: std_logic;  
  signal RSTB: std_logic;  
  
  signal CNT: std_logic_vector((N - 1) downto 0);  
  constant PER: time := 20 ns;  
  
  -----  
  -- Component prototype  
  -----  
  COMPONENT counter_unsigned_n_bit  
  generic(  
    n: natural := 4  
  );  
  PORT  
  (  
    i_rstb : IN STD_LOGIC;  
    i_clk  : IN STD_LOGIC;  
    o_cnt  : OUT STD_LOGIC_VECTOR(n-1 downto 0)  
  );  
  END COMPONENT;  
  
  -----  
begin  
  
  -----  
  -- Device under test (DUT)  
  -----  
  DUT: counter_unsigned_n_bit  
  generic map(  
    N => N  
  )  
  port map(  
    i_rstb => RSTB,  
    i_clk  => CLK,  
    o_cnt  => CNT  
  );  
  
end architecture;
```

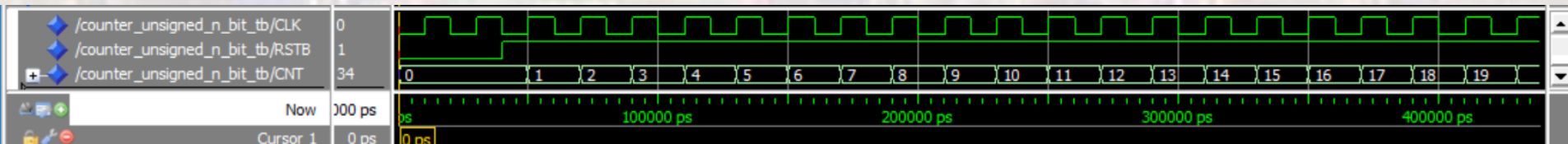
```
-----  
-- Test processes  
-----  
  
-- Clock process  
clock: process -- no sensitivity list allowed  
begin  
  CLK <= '0';  
  wait for PER/2;  
  infinite: loop  
    CLK <= not CLK; wait for PER/2;  
  end loop;  
end process clock;  
  
-- Reset process  
reset: process -- no sensitivity list allowed  
begin  
  RSTB <= '0'; wait for 2*PER;  
  RSTB <= '1'; wait;  
end process reset;  
  
-- Run Process  
-- empty  
  
-----  
-- End test processes  
-----  
end architecture;
```

Counters

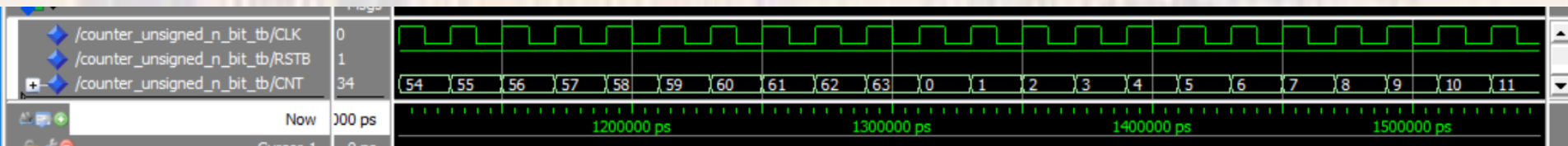
- Counter - n bit – unsigned (6 bit version)

reset

counting



wrap



Counters

Mod counter

Counters

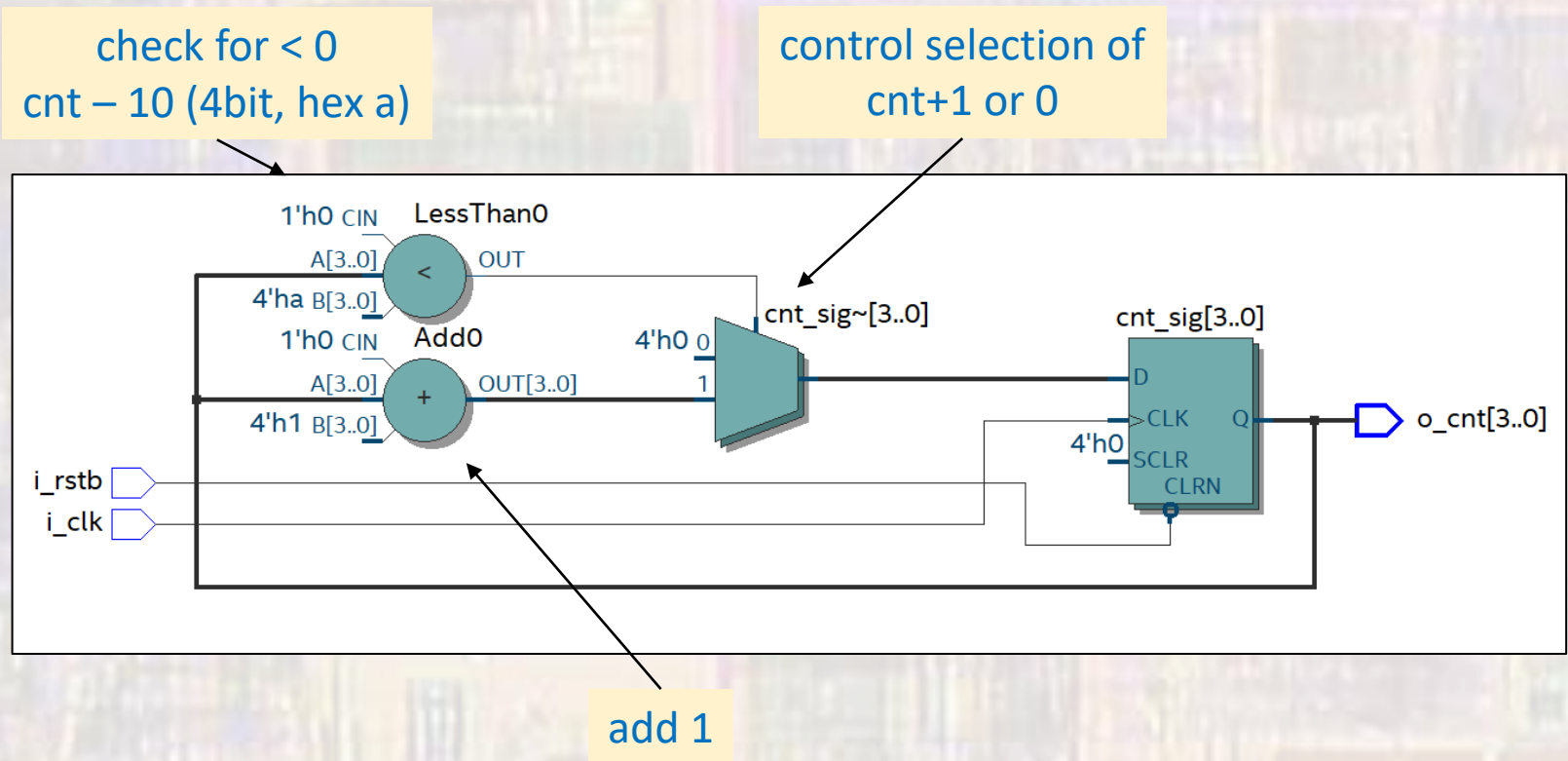
- Mod 11 counter

```
-----  
-- counter_mod_11.vhd1  
--  
-- created 3/17/17  
-- tj  
--  
-- rev 0  
-----  
--  
-- mod 11 counter example  
--  
-----  
-- Inputs: rstb, clk  
-- Outputs: cnt[3:0]  
-----  
library ieee;  
use ieee.std_logic_1164.all;  
use ieee.numeric_std.all;  
entity counter_mod_11 is  
  port (  
    i_clk : in std_logic;  
    i_rstb : in std_logic;  
  
    o_cnt : out std_logic_vector(3 downto 0)  
  );  
end entity;
```

```
architecture behavioral of counter_mod_11 is  
  --  
  -- internal signals  
  signal cnt_sig: unsigned(3 downto 0);  
begin  
  count: process(i_clk, i_rstb)  
  begin  
    -- reset  
    if (i_rstb = '0') then  
      cnt_sig <= (others => '0');  
    -- rising clk edge  
    --  
    elsif (rising_edge(i_clk)) then  
      if (cnt_sig < 10) then  
        cnt_sig <= cnt_sig + 1;  
      else  
        cnt_sig <= (others => '0');  
      end if;  
    end if;  
  end process;  
  
  --  
  -- Output logic  
  --  
  o_cnt <= std_logic_vector(cnt_sig);  
end behavioral;
```

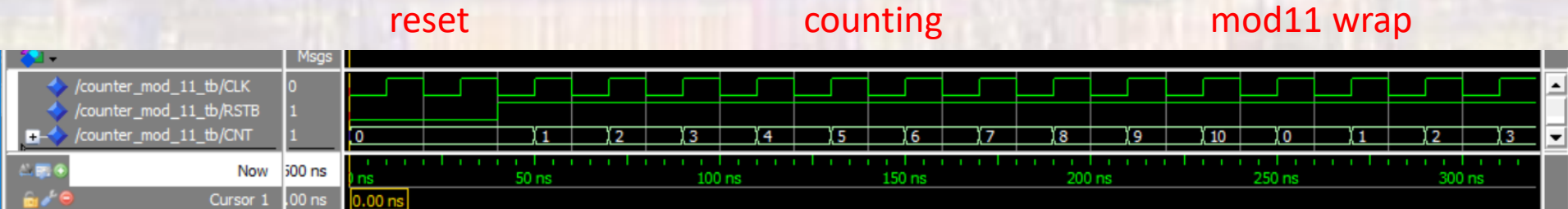
Counters

- Mod 11 counter



Counters

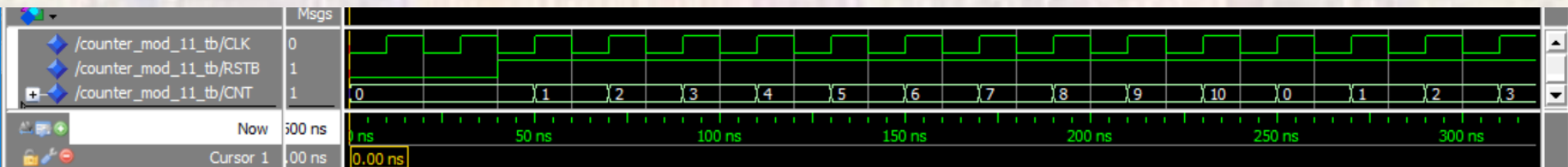
- Mod 11 counter



reset

counting

mod11 wrap



Counters

Up/Down counter

Counters

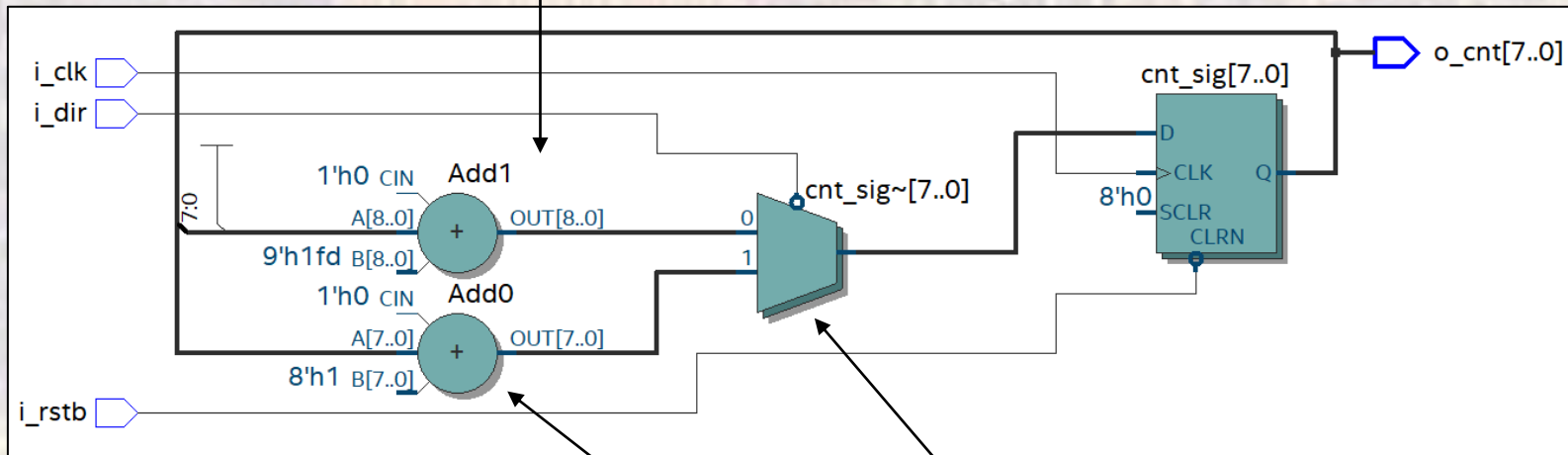
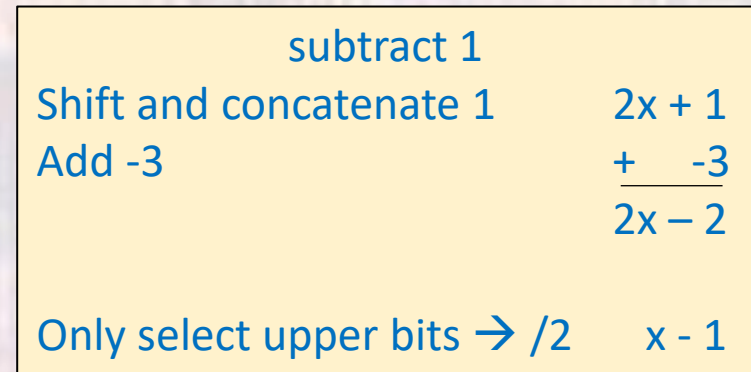
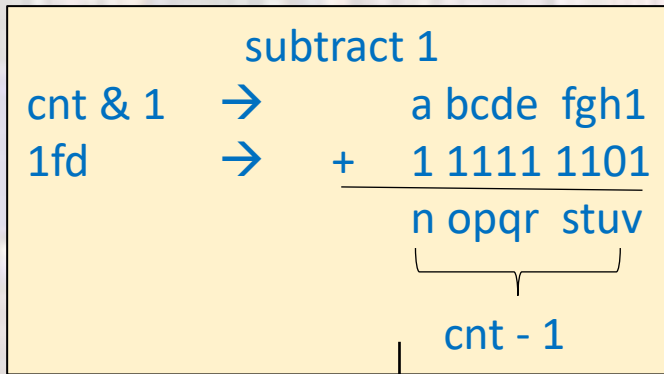
- up/down signed nbit counter

```
-----  
-- counter_updn_signed_nb.vhdl  
--  
-- created 2/29/17  
-- tj  
--  
-- rev 0  
-----  
--  
-- n bit up/down signed counter example  
--  
-----  
--  
-- Inputs: rstb, clk, dir  
-- Outputs: cnt[n-1:0]  
--  
-----  
--  
-- counts up when dir = 0  
-- counts down when dir = 1  
--  
-----  
library ieee;  
use ieee.std_logic_1164.all;  
use ieee.numeric_std.all;  
entity counter_updn_signed_nb is  
  generic(  
    n: natural := 8  
  );  
  port (  
    i_clk: in std_logic;  
    i_rstb: in std_logic;  
    i_dir: in std_logic;  
  
    o_cnt : out std_logic_vector(n-1 downto 0)  
  );  
end entity;
```

```
architecture behavioral of counter_updn_signed_nb is  
  --  
  -- internal signals  
  --  
  signal cnt_sig: signed(n-1 downto 0);  
begin  
  count: process(i_clk, i_rstb)  
  begin  
    --  
    -- reset  
    --  
    if (i_rstb = '0') then  
      cnt_sig <= (others => '0');  
    --  
    -- rising clk edge  
    --  
    elsif (rising_edge(i_clk)) then  
      if(i_dir = '0') then  
        cnt_sig <= cnt_sig + 1;  
      else  
        cnt_sig <= cnt_sig - 1;  
      end if;  
    end if;  
  end process;  
  
  --  
  -- Output logic  
  --  
  o_cnt <= std_logic_vector(cnt_sig);  
end behavioral;
```

Counters

- up/down signed nbit counter (default 8 bit)



add 1

select + or -

Counters

- up/down signed nbit counter (6 bit test)

```
-----  
-- counter_updn_signed_nb_tb.vhdl  
--  
-- created: 3/17/18  
-- by: johnsontim  
-- rev: 0  
--  
-- testbench for up down counter  
-- of counter_updn_signed_nb.vhdl  
--  
-----  
library ieee;  
use ieee.std_logic_1164.all;  
  
entity counter_updn_signed_nb_tb is  
  generic(  
    N: natural := 6  
  );  
  -- no port entry - testbench  
end entity;
```

```
architecture testbench of counter_updn_signed_nb_tb is  
  signal CLK: std_logic;  
  signal RSTB: std_logic;  
  signal DIR: std_logic;  
  
  signal CNT: std_logic_vector((N - 1) downto 0);  
  
  constant PER: time := 20 ns;  
  
  -----  
  -- Component prototype  
  -----  
  COMPONENT counter_updn_signed_nb  
  GENERIC ( n : NATURAL := 8 );  
  PORT  
  (  
    i_rstb : IN STD_LOGIC;  
    i_clk  : IN STD_LOGIC;  
    i_dir  : IN STD_LOGIC;  
    o_cnt  : OUT STD_LOGIC_VECTOR((n - 1) downto 0)  
  );  
END COMPONENT;  
  
-----  
begin  
  
-----  
-- Device under test (DUT)  
-----  
DUT: counter_updn_signed_nb  
  generic map(  
    n => N  
  )  
  port map(  
    i_clk  => CLK,  
    i_rstb => RSTB,  
    i_dir  => DIR,  
    o_cnt  => CNT  
  );
```

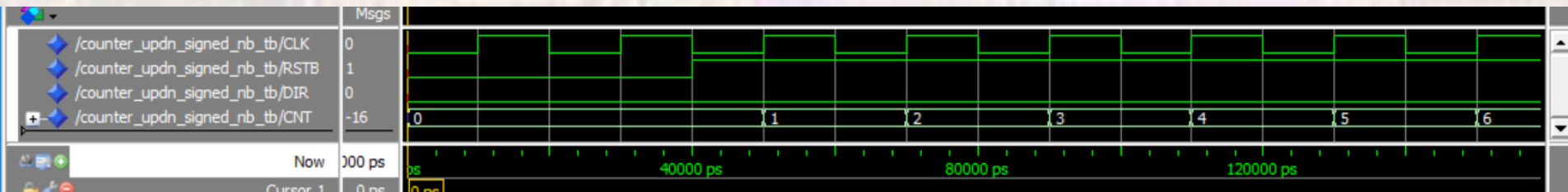
```
-----  
-- Test processes  
-----  
  
-- Clock process  
clock: process -- no sensitivity list allowed  
begin  
  CLK <= '0';  
  wait for PER/2;  
  infinite: loop  
    CLK <= not CLK; wait for PER/2;  
  end loop;  
end process clock;  
  
-- Reset process  
reset: process -- no sensitivity list allowed  
begin  
  RSTB <= '0'; wait for 2*PER;  
  RSTB <= '1'; wait;  
end process reset;  
  
-- Run Process  
run: process -- no sensitivity list allowed  
begin  
  -- initialize inputs  
  DIR <= '0';  
  -- run code  
  wait for 68*PER;  
  DIR <= '1';  
  wait for 68*PER;  
end process run;  
  
-----  
-- End test processes  
-----  
  
end architecture;
```

Counters

- up/down signed nbit counter (6 bit version)

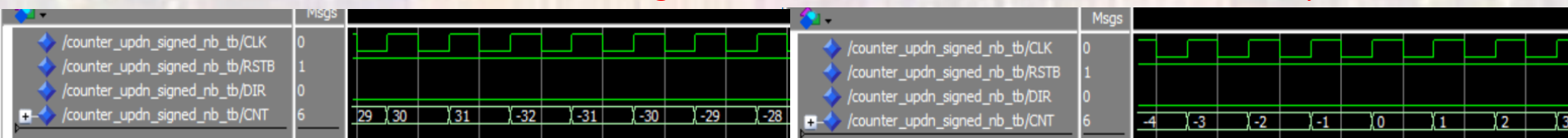
reset

counting up



transition to negative

transition to positive



counting down

