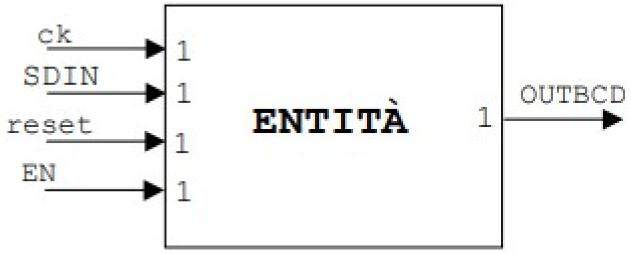
6 September 2006

Create a clock-synchronous VHDL entity with a serial input (**`SDIN`**) and a serial output (**`OUTBCD`**).

The entity must count the number of 1 in the input stream starting from the activation of the enable signal (**`EN`**) for 8 consecutive clock cycles. The count must be coded in BCD and transferred to the output during the next 4 clock cycles.

The entity must continue to count as described for as long as the entity is enabled. The output is in high impedance except when it transfers the output bitstream.

Finally, the entity must have an asynchronous reset signal (**`reset`**).



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

--serial input; serial output.
--The entity should count the occurences of '1' in the input bitstream;
--it should start counting from an enabling signal
--(EN) commutation, for 8 clock periods of time and in the meantime output should
--be kept in high impedance state;
--the number of ones should be coded in 4 bits BCD format and later
--transferred to the output bitstream.
--The entity should continue its operation as EN is activated.
--Asynchronous should be performed.

entity BCD is
   port(RST, CLK, SIN, EN: in std_logic;
        SOUT: out std_logic);
end BCD;

architecture Behavioral of BCD is
   signal cnt: integer range 0 to 15:=0;
   signal reg: std_logic_vector (3 downto 0):="0000";
   --NB: since the package "unsigned" is used, reg can be simply
   --incremented; 4 bits of reg can be manipulated as an unsigned
   --value.
begin

process (EN, RST, CLK)
begin
   --1) RST=1: general reset;
   if (RST = '1') then
      cnt <= 0;
      reg <= "0000";
      SOUT <= 'Z';
   --2) RST=0:
   elsif (rising_edge(CLK)) then
      if (cnt < 8) then
         --3)
         --(8 CLK periods of time)
         --EN=0: doesn't count;
         --EN=1 counts.
         --cnt is incremented anyway.
         --SOUT is kept in high impedance state.
         if (EN = '1' and SIN = '1') then
            reg <= reg + 1;
         end if;
         SOUT <= 'Z';
         cnt <= cnt + 1;
      --4) reg is tranferred to SOUT from LSB to MSB (4 CLK periods of time).
      elsif (cnt > 7 and cnt < 11) then
         SOUT <= reg (0);
         reg <= 'Z' & reg (3 downto 1);
         cnt <= cnt + 1;
      --5) by the time the MSB is tranferred to SOUT, cnt and reg should be
      --reset to 0 and 0000 in order to start a new count.
      elsif (cnt = 11) then
         SOUT <= reg (0);
         reg <= "0000";
         cnt <= 0;
      end if;
   end if;
end process;

end Behavioral;
```