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

entity PID_regulator_ver1 is
  Port ( clk : in STD_LOGIC;
         Sample : in STD_LOGIC;
         reset : in STD_LOGIC;
         Ref : in STD_LOGIC_VECTOR (7 downto 0);
         Act : in STD_LOGIC_VECTOR (7 downto 0);
         Kp : in STD_LOGIC_VECTOR (17 downto 0);
         Ki : in STD_LOGIC_VECTOR (17 downto 0);
         Kd : in STD_LOGIC_VECTOR (17 downto 0);
         Yout : out STD_LOGIC_VECTOR (7 downto 0));
end PID_regulator_ver1;

architecture Behavioral of PID_regulator_ver1 is
  signal Error: integer range -65535 to 65535 := 0;
  signal Prev_Error: integer range -65535 to 65535 := 0;
  signal Integrator: STD_LOGIC_VECTOR (17 downto 0) := (others=>'0');
  -- This alias by default divides the Integrator with 4
  alias Integ: STD_LOGIC_VECTOR (15 downto 0) is Integrator(17 downto 2);
  signal Int_Integ: integer range -65535 to 65535 := 0 ;
  signal Actuator: integer range -65535 to 65535 := 0 ;
  signal P_bidrag: integer range -65535 to 65535 := 0 ;
signal I_bidrag: integer range -65535 to 65535 := 0;
signal D_bidrag: integer range -65535 to 65535 := 0;
signal Saturation: STD_LOGIC;

begin
  Int_Integ <= conv_integer(Integ);

  Integration:
  process (Sample)
    variable Err: integer;
    begin
      if rising_edge(Sample) then
        if Reset='1' then
          Integrator <= (others=>'0');
          Error     <= 0;
        else
          Err := (conv_integer(Ref)-conv_integer(Act));
          -- Deadband = +/-1 - In order to get a more stable regulator
          if Abs(Err) > 1 then
            -- No reset wind-up - no integration when output saturated
            if Saturation = '0' then
              Integrator <= Integrator+Err/2;
            end if;
            -- No negative values allowed - remove if you wants them
            if conv_integer(Integrator)<0 then
              Integrator <= (others=>'0');
            end if;
            Prev_Error <= Error;  -- Remember the previous error
            Error     <= Err;     -- New Error
          end if;
        end if;
      end if;
    end process;
\texttt{P\_bidrag} <= \texttt{conv\_integer(Kp)*Error};
\texttt{I\_bidrag} <= \texttt{conv\_integer(Ki)*Int\_Integ};
\texttt{D\_bidrag} <= \texttt{conv\_integer(Kd)*(Error - Prev\_Error)};

\textbf{Regulator:}
\texttt{process ( P\_bidrag, I\_bidrag, D\_bidrag)}
\begin{itemize}
  \item \texttt{variable Yact: integer;}
  \item \texttt{begin}
    \item \texttt{Yact := P\_bidrag + I\_bidrag + D\_bidrag;}
    \item \texttt{Actuator <= Yact;}
    \item \texttt{Saturation <= '0';}
    \item \texttt{if Yact<0 then}
      \item \texttt{Yact:=0;}
    \item \texttt{elsif Yact>255 then}
      \item \texttt{Yact:=255;}
    \item \texttt{Saturation <= '1';}
    \item \texttt{end if;}
    \item \texttt{Yout <= conv\_std\_logic\_vector ( Yact, 8);}
  \item \texttt{end process;}
\end{itemize}
\texttt{end Behavioral;}