```
--------------------------------------------------------------------------------
--
--  Purpose:
--
--    Carry Lookahead Adder
--
--  Discussion:
--
--
--  Licensing:
--
--    This code is distributed under the GNU LGPL license.
--
--  Modified:
--
--    2014.04.09
--
--  Author:
--
--    Young W. Lim
--
--  Parameters:
--
--    Input: an(31:0), bn(31:0), ci
--
--    Output: {co, cn(31:0)} = an(31:0) + bn(31:0) + ci
--------------------------------------------------------------------------------

library STD;
use STD.textio.all;

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;




architecture ksa of adder is

  function logstep(x : natural) return natural is
    variable n : natural := 1;
    variable i : natural := 0;
  begin
    while n < x loop
      n := n * 2 ;
      i := i + 1;
    end loop;
    return (i);
```

```
  end function logstep;


  constant ND : natural := WD/BD; -- (8 = 32/4)
  constant SH : natural := logstep(WD);

  -- ga(j) : WD-bit G
  -- pa(j) : WD-bit P
  -- ca(j) : WD-bit Carry
  signal ga : std_logic_vector (WD-1 downto 0) := (others=>'0');
  signal pa : std_logic_vector (WD-1 downto 0) := (others=>'0');
  signal ca : std_logic_vector (WD-1 downto 0) := (others=>'0');

  -- type array2d is array (SH-1 downto 0) of std_logic_vector (WD-1 downto 0);
  type array2d is array (SH downto 0, WD-1 downto 0) of std_logic;

  -- type array1d is array (ND-1 downto 0) of  std_logic;
  -- signal ci1d : array1d := (others=> '0');


begin

  process (an, bn, ci)
    -- ga2d(i, j) : i-th level WD-bit G
    -- pa2d(i, j) : i-th level WD-bit P
    -- co2d(i, j) : i-th level WD-bit Carry
    variable ga2d : array2d := ((others=> (others=> '0'))); -- hold ga(31:0) data
    variable pa2d : array2d := ((others=> (others=> '0'))); -- hold pa(31:0) data
    variable co2d : array2d := ((others=> (others=> '0'))); -- hold co(31:0) data

    variable cn_val : std_logic_vector (WD-1 downto 0) := (others=>'0');

    variable offset : integer := 1;
    variable l : line;

  begin
    write(l, String'("........................................."));
    writeline(output, l);

    for j in WD-1 downto 0 loop
       if (an(j) = '1') then
         write(l, String'("1"));
       else
         write(l, String'("0"));
       end if;
       if (j rem 4 = 0) then
         write(l, String'("_"));
       end if;
    end loop;
    writeline(output, l);
```

```
    for j in WD-1 downto 0 loop
       if (bn(j) = '1') then
         write(l, String'("1"));
       else
         write(l, String'("0"));
       end if;
       if (j rem 4 = 0) then
         write(l, String'("_"));
       end if;
    end loop;
    writeline(output, l);



    -- i : level index; offset = 2^i

   for j in WD-1 downto 0 loop
     pa2d(0, j) := an(j) xor bn(j);
     ga2d(0, j) := an(j) and bn(j);
   end loop;

    offset := 1;
    for i in 1 to SH loop
--      write(l, String'("i="));
--      write(l, i);
--      write(l, String'(" "));
--      writeline(output, l);
      for j in WD-1 downto 0 loop
        if (j >= offset) then
          pa2d(i, j) := (pa2d(i-1, j) and pa2d(i-1, j-offset));
          ga2d(i, j) := (pa2d(i-1, j) and ga2d(i-1, j-offset)) or ga2d(i-1, j);
        else
          pa2d(i, j) := pa2d(i-1, j);
          ga2d(i, j) := ga2d(i-1, j);
        end if;

        pa(j) <= pa2d(i,j);
        ga(j) <= ga2d(i,j);


       if (pa2d(i,j) = '1') then
         write(l, String'("1"));
       else
         write(l, String'("0"));
       end if;
       if (j rem 4 = 0) then
         write(l, String'("_"));
       end if;
```

```
      end loop;

      write(l, offset);

      offset := offset *2;

      writeline(output, l);

    end loop;

    for j in WD-1 downto 0 loop
      if (j = 0) then
        cn_val(j) := pa2d(0,j) xor ci;
      else
        cn_val(j) := pa2d(0,j) xor ga2d(SH, j-1);
      end if;
      ga(j) <= ga2d(0, j);
      pa(j) <= pa2d(SH, j);
      ca(j) <= ga2d(SH, j);
    end loop;


    co <= ga2d(SH, WD-1);
    cn <= cn_val;

  end process;



end ksa;
```