```
----------------------------------------------------------------------------------
-- Engineer: Taggart Bonham and Will Chisholm
--
-- Create Date: 05/23/2017 09:05:13 PM
-- Design Name:
-- Module Name: CurPixel- Behavioral
-- Project Name: Etch-a-Sketch final project
-- Target Devices: Digilent Basys3 Board
-- Tool Versions: Vivado 2016.1
-- Description: Logic for the current pixel on the screen. Reacts to control
--signals based on rotary encoder input.

----------------------------------------------------------------------------------



library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity CurPixel is
    Port ( clk: in STD_LOGIC;
           UP : in STD_LOGIC; --tells which direction to move the current location of the pixel
           DOWN : in STD_LOGIC;
           LEFT : in STD_LOGIC;
           RIGHT : in STD_LOGIC;
           curLoc : out STD_LOGIC_VECTOR(16 downto 0)); --new current location of pixel
end CurPixel;

architecture Behavioral of CurPixel is

--x,y locations
-- since 640 x 480, in order to make lines bigger we divided both values by 2 for storage
signal cur_x_loc: unsigned(8 downto 0):= "100000000"; --initializes to a spot far away from the
edges so it's easy to see where the pixel starts
signal cur_y_loc : unsigned(7 downto 0):= "10000000";

constant MAX_X : integer := 320; --screen x from 0 to 640/2 = 320
constant MAX_Y: integer := 240; --y from 0 to 480/2 = 240


begin

xReg: process(clk) --updates x location
begin

if rising_edge(clk) then
    if (RIGHT = '1') then
        if (cur_x_loc + 1 < MAX_X) then
            cur_x_loc <= cur_x_loc + 1;
        end if;

    elsif (LEFT = '1') then
        if (cur_x_loc - 1 > 0) then
            cur_x_loc <= cur_x_loc - 1;
        end if;
    end if;
end if;

end process;

yReg: process(clk) --updates y location
begin

if rising_edge(clk) then
    if (DOWN = '1') then
        if (cur_y_loc + 1 < MAX_Y) then
            cur_y_loc <= cur_y_loc + 1;
        end if;

    elsif (UP = '1') then
        if (cur_y_loc - 1 > 1) then
            cur_y_loc <= cur_y_loc - 1;
        end if;
    end if;
end if;

end process;

--combines the x and y into one std_logic_vector to output
output: process(clk, cur_x_loc, cur_y_loc)
begin
    if (rising_edge(clk)) then
        curLoc <= STD_LOGIC_VECTOR(cur_x_loc) & STD_LOGIC_VECTOR(cur_y_loc);
    end if;

end process;

end Behavioral;
```