```
----------------------------------------------------------------------------------
-- Engineer: Taggart Bonham and Will Chisholm
-- Create Date: 05/23/2017 09:05:13 PM
-- Module Name: knobcontrollertest_top- Behavioral
-- Project Name: Etch-a-Sketch final project
-- Target Devices: Digilent Basys3 Board
-- Tool Versions: Vivado 2016.1
-- Description: test top for knobs and controllers for oscillioscopes
----------------------------------------------------------------------------------



library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.numeric_std.ALL;
library UNISIM;
use UNISIM.VComponents.all;

entity kct_TOP is
    Port ( mclk : in STD_LOGIC;
           r,g,b : in STD_LOGIC;
           XA, XB, YA, YB : in STD_LOGIC;
           rst : in STD_LOGIC;
           u, d : out STD_LOGIC;
end kct_TOP;

architecture Behavioral of kct_TOP is


--controller for design
--inputs from rotary encoders turned into control signals
--responsible for moving the current location and clearing
Component cwController is
PORT (     clk : in STD_LOGIC;
           a1 : in STD_LOGIC;
           a2 : in STD_LOGIC;
           b1 : in STD_LOGIC;
           b2 : in STD_LOGIC;
           rst : in STD_LOGIC;
           UP : out STD_LOGIC;
           DOWN : out STD_LOGIC;
           LEFT : out STD_LOGIC;
           RIGHT : out STD_LOGIC;
           CLR : out STD_LOGIC);
  end component;


-- Signals for the serial clock divider, which divides the 100 MHz clock down to 1 MHz
constant SCLK_DIVIDER_VALUE: integer := 2;--100 / 2;
signal sclkdiv: unsigned(1 downto 0) := (others => '0');  -- clock divider counter
signal sclk_unbuf: std_logic := '0';    -- unbuffered serial clock
signal sclk: std_logic := '0';          -- internal serial clock


--wiring
--outputs from VGA_SYNC
signal vid_on : STD_LOGIC;
signal x : STD_LOGIC_VECTOR(9 downto 0);
signal y: STD_LOGIC_VECTOR(8 downto 0);

--inputs to CurrentPixel
signal PIXELUP : STD_LOGIC;
signal PIXELDOWN : STD_LOGIC;
signal PIXELLEFT : STD_LOGIC;
signal PIXELRIGHT : STD_LOGIC;



--inputs to 2x1 Mux
signal CLR : std_logic;
signal currentPixel : STD_LOGIC_VECTOR(16 downto 0);


--------------------------------------------------
begin
-- Clock buffer for sclk
-- The BUFG component puts the signal onto the FPGA clocking network

u <= PIXELUP;
d <= PIXELDOWN;

Slow_clock_buffer: BUFG
    port map (I => sclk_unbuf,
              O => sclk );

-- Divide the 100 MHz clock down to 25 MHz, then toggling a flip flop gives the final
-- 1 MHz system clock
Serial_clock_divider: process(mclk)
begin
    if rising_edge(mclk) then
        if sclkdiv = 1 then
            sclkdiv <= (others => '0');
            sclk_unbuf <= NOT(sclk_unbuf);
        else
            sclkdiv <= sclkdiv + 1;
        end if;
    end if;
end process Serial_clock_divider;


--controller converts rotary signals into Left right up down signals
--additionally drives the clearing of the screen
controls: cwController port map(
    clk => sclk,
    a1 => XA,
    a2 => XB,
    b1 => YA,
    b2 => YB,
    rst => '0',
    UP => PIXELUP,
    DOWN => PIXELDOWN,
    LEFT => PIXELLEFT,
    RIGHT => PIXELRIGHT,
    CLR => CLR);

end Behavioral;
```