```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.all;
USE IEEE.STD_LOGIC_ARITH.all;
USE IEEE.STD_LOGIC_UNSIGNED.all;
USE work.dice_states.all;


ENTITY backgrnd IS
        PORT(
        clock                           : in STD_LOGIC;
        countX, countY                  : in STD_LOGIC_VECTOR(9 downto 0);
        vga_red, vga_green, vga_blue: out BOOLEAN
        );
END backgrnd;

ARCHITECTURE back of backgrnd IS

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

SIGNAL   equal_x, equal_y               : BOOLEAN;
SIGNAL   raster_x, raster_y, raster_xy  : BOOLEAN;
SIGNAL   equalcolor_x,equalcolor_y      : BOOLEAN;
SIGNAL   stop, bonus, penalty           : BOOLEAN;

SIGNAL   lattice_vga_red,lattice_vga_green,lattice_vga_blue : BOOLEAN;
SIGNAL   block_vga_red,  block_vga_green,  block_vga_blue   : BOOLEAN;
SIGNAL   label_vga_red,  label_vga_green,  label_vga_blue   : BOOLEAN;
-------------------------------------------------------------------------------



BEGIN

-------------------------------------------------------------------------------
panel_lattice : process(clock)

BEGIN

IF clock'EVENT and clock = '1' THEN


       raster_xy <=((countX >= panel_left_x) and (countX  <= panel_right_x))
                   and ((countY  >= panel_top_y) and (countY <= panel_bottom_y))
                   and (((countX and raster_width)  = all_zeros )
                   or ((countY and raster_length) = all_zeros ) );

        lattice_vga_red   <= raster_xy;
        lattice_vga_green <= raster_xy;
        lattice_vga_blue  <= raster_xy;



END IF;
END process panel_lattice;
-------------------------------------------------------------------------------

panel_block : process(clock)

BEGIN

IF clock'EVENT and clock = '1' THEN
        equalcolor_x   <= (countX > panel_left_x) and (countX  < panel_right_x);
        equalcolor_y   <= (countY > panel_top_y ) and (countY  < panel_bottom_y);
```

```
--   the stop block is x = 224, y=352 ---
--   it is in second column, seventh row
        stop <= ((countX(9 downto 5) = second_column_x1(5 downto 1))
                and (countY(9 downto 5) = seventh_row_y1(5 downto 1)))
                or ((countX(9 downto 5) = seventh_column_x1(5 downto 1))
                and (countY(9 downto 5) = third_row_y1(5 downto 1)));

--   the bonus block is x = 256, y= 192 --
--   it is in third column, second row
        bonus<= ((countX(9 downto 5) = third_column_x1(5 downto 1))
                and (countY(9 downto 5) = second_row_y1(5 downto 1)))
                or ((countX(9 downto 5) = eighth_column_x1(5 downto 1))
                and (countY(9 downto 5) = sixth_row_y1(5 downto 1)));
--   the penalty block is x = 352, y = 384--
--   it is in sixth column, eighth row
        penalty <= ((countX(9 downto 5) = sixth_column_x1(5 downto 1))
                   and (countY(9 downto 5) = eighth_row_y1(5 downto 1)))
                   or ((countX(9 downto 5) = fifth_column_x1(5 downto 1))
                   and (countY(9 downto 5) = fourth_row_y1(5 downto 1)));

--  stop block color is red, bonus color is green, penalty color is white
--  color, the normal background color is blue.
--

        block_vga_red   <= stop or penalty;
        block_vga_green <= bonus or penalty;
        block_vga_blue  <= (equalcolor_x and equalcolor_y) xor (bonus or stop);

END IF;
END process panel_block;

label_block : process(clock)

Begin
If clock'event and clock='1' then
   label_vga_red   <= (countX >="0111110000" and countX <="1000001000") and
                      ((countY >="0010100000" and countY <="0010111000") or
                       (countY >="0011100000" and countY <="0011111000"));


   label_vga_green <= (countX >="0111110000" and countX <="1000001000") and
                      ((countY >="0011000000" and countY <="0011011000") or
                      (countY >="0011100000" and countY <="0011111000"));

   label_vga_blue  <= (countX >="0111110000" and countX <="1000001000") and
                      (countY >="0011100000" and countY <="0011111000");
end If;
end process label_block;

    vga_red   <= (lattice_vga_red   OR block_vga_red   OR label_vga_red);
    vga_green <= (lattice_vga_green OR block_vga_green OR label_vga_green);
    vga_blue  <= (lattice_vga_blue  OR block_vga_blue  OR label_vga_blue);
END back;
```