**VGA\_driver v1**

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 vga\_driver is

Port ( clk : in STD\_LOGIC;

r : out STD\_LOGIC\_VECTOR (3 downto 0);

g : out STD\_LOGIC\_VECTOR (3 downto 0);

b : out STD\_LOGIC\_VECTOR (3 downto 0);

hs : out STD\_LOGIC;

vs : out STD\_LOGIC);

end vga\_driver;

architecture Behavioral of vga\_driver is

signal row : unsigned (9 downto 0);

signal col : unsigned (9 downto 0);

begin

process(clk)

variable hCnt : integer range 0 to 3500;

variable vCnt : integer range 0 to 1100;

variable hTmp : unsigned (11 downto 0);

variable vTmp : unsigned (11 downto 0);

begin

if rising\_edge(clk) then

hCnt := hCnt + 1;

if hCnt = 3200 then

vCnt := vCnt + 1;

if vCnt = 521 then

vCnt := 0;

end if;

hCnt := 0;

end if;

hTmp := to\_unsigned(hCnt, 12); -- convert integer to vector

vTmp := to\_unsigned(vCnt, 12);

if hCnt > 2624 and hCnt < 3008 then

hs <= '1';

else

hs <= '0';

end if;

if vCnt > 489 and vCnt < 493 then

vs <= '1';

else

vs <= '0';

end if;

row <= vTmp(11 downto 2); -- divide by 4

col <= hTmp(11 downto 2); -- divide by 4

end if;

end process;

process(row, col)

begin

if row < 240 and col < 320 then

r <= "0110";

g <= "1000";

b <= "0000";

elsif row < 480 and col < 640 then

r <= "0000";

g <= "1000";

b <= "0110";

else

r <= "0000";

g <= "0000";

b <= "0000";

end if;

end process;

end Behavioral;

**Mover**

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 mover is

Port ( clk : in STD\_LOGIC;

x\_loc : out STD\_LOGIC\_VECTOR (9 downto 0);

y\_loc : out STD\_LOGIC\_VECTOR (9 downto 0));

end mover;

architecture Behavioral of mover is

begin

process(clk)

variable count0 : unsigned(9 downto 0);

variable count1 : unsigned(9 downto 0);

variable count2 : unsigned(9 downto 0);

variable count3 : unsigned(9 downto 0);

begin

if rising\_edge(clk) then

count0 := count0 + 1;

if count0 = 1000 then

count0 := "0000000000";

count1 := count1 + 1;

if count1 = 1000 then

count1 := "0000000000";

count2 := count2 + 1;

if count2 = 400 then

count2 := "0000000000";

end if;

end if;

end if;

end if;

x\_loc <= std\_logic\_vector(count2);

y\_loc <= std\_logic\_vector(count2);

end process;

end Behavioral;

**Brick V1**

entity bricks is

Port ( bricksL1 : in STD\_LOGIC\_VECTOR(31 downto 0); -- level 1 bricks (1 = unbroken)

bricksL2 : in STD\_LOGIC\_VECTOR(31 downto 0);

bricksL3 : in STD\_LOGIC\_VECTOR(31 downto 0);

--bricksL4 : in STD\_LOGIC\_VECTOR(31 downto 0);

x\_draw : in STD\_LOGIC\_VECTOR (9 downto 0);

y\_draw : in STD\_LOGIC\_VECTOR (9 downto 0);

r : out STD\_LOGIC\_VECTOR (3 downto 0);

g : out STD\_LOGIC\_VECTOR (3 downto 0);

b : out STD\_LOGIC\_VECTOR (3 downto 0));

end bricks;

-- bricksL1 bits:

-- bit #: 0 1 2 ... 13

-- brick: b1 b2 b3 ... b14

-- bit #: 14 15 16 ... 27 28 29 30 31

-- brick row2: b15 b16 b17 ... b24 0 0 0 0

architecture Behavioral of bricks is

signal bricksL1\_color : STD\_LOGIC\_VECTOR (11 downto 0);

signal bricksL2\_color : STD\_LOGIC\_VECTOR (11 downto 0);

signal bricksL3\_color : STD\_LOGIC\_VECTOR (11 downto 0);

--signal bricksL4\_color : STD\_LOGIC\_VECTOR (11 downto 0);

begin

r <= bricksL1\_color(11 downto 8) or bricksL2\_color(11 downto 8) or bricksL3\_color(11 downto 8);

g <= bricksL1\_color(7 downto 4) or bricksL2\_color(7 downto 4) or bricksL3\_color(7 downto 4);

b <= bricksL1\_color(3 downto 0) or bricksL2\_color(3 downto 0) or bricksL3\_color(3 downto 0);

-- LEVEL 1 BRICKS

block1: process(bricksL1, x\_draw, y\_draw)

-- 2 rows; starting pos= (132, 160); each brick is 25x8 with 2 px padding on bottom and right

begin

bricksL1\_color <= x"000";

-- ROW 1

if (unsigned(y\_draw) >= 160 and unsigned(y\_draw) <= 168) then

-- if the first brick is unbroken

if (bricksL1(0) = '1') then

-- if within x bounds -> x\_draw >= 132 + 27\*(bit num) AND x\_draw <= 157+ 27\*(bit num); [132,157]

if (unsigned(x\_draw) >= 132 + 27\*0 and unsigned(x\_draw) <= 157 + 27\*0) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(1) = '1') then

if (unsigned(x\_draw) > 132 + 27\*1 and unsigned(x\_draw) <= 157+27\*1) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(2) = '1') then

if (unsigned(x\_draw) > 132 + 27\*2 and unsigned(x\_draw) <= 157 + 27\*2) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(3) = '1') then

if (unsigned(x\_draw) > 132 + 27\*3 and unsigned(x\_draw) <= 157 + 27\*3) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(4) = '1') then

if (unsigned(x\_draw) > 132 + 27\*4 and unsigned(x\_draw) <= 157 + 27\*4) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(5) = '1') then

if (unsigned(x\_draw) > 132 + 27\*5 and unsigned(x\_draw) <= 157 + 27\*5) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(6) = '1') then

if (unsigned(x\_draw) > 132 + 27\*6 and unsigned(x\_draw) <= 157 + 27\*6) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(7) = '1') then

if (unsigned(x\_draw) > 132 + 27\*7 and unsigned(x\_draw) <= 157 + 27\*7) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(8) = '1') then

if (unsigned(x\_draw) > 132 + 27\*8 and unsigned(x\_draw) <= 157 + 27\*8) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(9) = '1') then

if (unsigned(x\_draw) > 132 + 27\*9 and unsigned(x\_draw) <= 157 + 27\*9) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(10) = '1') then

if (unsigned(x\_draw) > 132 + 27\*10 and unsigned(x\_draw) <= 157 + 27\*10) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(11) = '1') then

if (unsigned(x\_draw) > 132 + 27\*11 and unsigned(x\_draw) <= 157 + 27\*11) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(12) = '1') then

if (unsigned(x\_draw) > 132 + 27\*12 and unsigned(x\_draw) <= 157 + 27\*12) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(13) = '1') then

if (unsigned(x\_draw) > 132 + 27\*13 and unsigned(x\_draw) <= 157 + 27\*13) then

bricksL1\_color <= x"FF3";

end if;

end if;

end if;

-- ROW 2

if (unsigned(y\_draw) >= 150 and unsigned(y\_draw) <= 158) then

-- if the first brick is unbroken

if (bricksL1(14) = '1') then

-- if within x bounds -> x\_draw >= 132 + 27\*(bit num) AND x\_draw <= 157+ 27\*(bit num); [132,157]

if (unsigned(x\_draw) >= 132 + 27\*0 and unsigned(x\_draw) <= 157 + 27\*0) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(15) = '1') then

if (unsigned(x\_draw) > 132 + 27\*1 and unsigned(x\_draw) <= 157+27\*1) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(16) = '1') then

if (unsigned(x\_draw) > 132 + 27\*2 and unsigned(x\_draw) <= 157 + 27\*2) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(17) = '1') then

if (unsigned(x\_draw) > 132 + 27\*3 and unsigned(x\_draw) <= 157 + 27\*3) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(18) = '1') then

if (unsigned(x\_draw) > 132 + 27\*4 and unsigned(x\_draw) <= 157 + 27\*4) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(19) = '1') then

if (unsigned(x\_draw) > 132 + 27\*5 and unsigned(x\_draw) <= 157 + 27\*5) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(20) = '1') then

if (unsigned(x\_draw) > 132 + 27\*6 and unsigned(x\_draw) <= 157 + 27\*6) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(21) = '1') then

if (unsigned(x\_draw) > 132 + 27\*7 and unsigned(x\_draw) <= 157 + 27\*7) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(22) = '1') then

if (unsigned(x\_draw) > 132 + 27\*8 and unsigned(x\_draw) <= 157 + 27\*8) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(23) = '1') then

if (unsigned(x\_draw) > 132 + 27\*9 and unsigned(x\_draw) <= 157 + 27\*9) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(24) = '1') then

if (unsigned(x\_draw) > 132 + 27\*10 and unsigned(x\_draw) <= 157 + 27\*10) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(25) = '1') then

if (unsigned(x\_draw) > 132 + 27\*11 and unsigned(x\_draw) <= 157 + 27\*11) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(26) = '1') then

if (unsigned(x\_draw) > 132 + 27\*12 and unsigned(x\_draw) <= 157 + 27\*12) then

bricksL1\_color <= x"FF3";

end if;

end if;

if (bricksL1(27) = '1') then

if (unsigned(x\_draw) > 132 + 27\*13 and unsigned(x\_draw) <= 157 + 27\*13) then

bricksL1\_color <= x"FF3";

end if;

end if;

end if;

end process;

-- LEVEL 2 BRICKS

--block2: process(bricksL2, x\_draw, y\_draw)

--begin

-- bricksL2\_color <= x"000";

-- if (unsigned(y\_draw) >= 140 and unsigned(y\_draw) <= 160) then

-- if (bricksL1(0) = '1') then

-- if (unsigned(x\_draw) >= 12 and unsigned(x\_draw) <= 37) then

-- bricksL1\_color <= x"FF3";

-- end if;

-- end if;

-- end if;

--end process;

end Behavioral;

**\*Note: if myip disappears, right click on the myip block diagram and select ‘validate design’**

**Paddle**

entity paddle is

Port ( x\_loc : in STD\_LOGIC\_VECTOR (31 downto 0); -- To enable full length paddle, set bit 10 to 1

x\_draw : in STD\_LOGIC\_VECTOR (31 downto 0);

y\_draw : in STD\_LOGIC\_VECTOR (31 downto 0);

r : out STD\_LOGIC\_VECTOR (3 downto 0);

g : out STD\_LOGIC\_VECTOR (3 downto 0);

b : out STD\_LOGIC\_VECTOR (3 downto 0));

end paddle;

architecture Behavioral of paddle is

begin

process(x\_loc, x\_draw, y\_draw)

begin

r <= "0000";

g <= "0000";

b <= "0000";

-- if within vert bounds

if(unsigned(y\_draw) >= 500 and unsigned(y\_draw) < 510) then

-- if full screen paddle is enabled

if (x\_loc(31) = '1') then

if(unsigned(x\_draw) >= 120 and unsigned(x\_draw) < 520) then

r <= "1111";

g <= "1111";

b <= "1111";

end if;

else

if (unsigned(x\_draw) >= 320 and unsigned(x\_draw) < 320 + 27) then

r <= "1111";

g <= "1111";

b <= "1111";

end if;

end if;

end if;

end process;

end Behavioral;

**Scoreboard**

entity scoreboard is

Port ( score : in STD\_LOGIC\_VECTOR (31 downto 0); -- score to draw

x\_draw : in STD\_LOGIC\_VECTOR (9 downto 0); -- electron beam position

y\_draw : in STD\_LOGIC\_VECTOR (9 downto 0);

r : out STD\_LOGIC\_VECTOR (3 downto 0);

g : out STD\_LOGIC\_VECTOR (3 downto 0);

b : out STD\_LOGIC\_VECTOR (3 downto 0));

end scoreboard;

architecture Behavioral of scoreboard is

type graphicsArray is array (positive range<>) of std\_logic;

constant dg0 : graphicsArray(0 to 599) := ( '1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1',

'1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1',

'1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1',

'1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1',

'1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1',

'1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','0','0','0','0','0','0','0','0','1','1','1','1','1','1',

'1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1',

'1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1',

'1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1',

'1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1',

'1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1',

'1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1','1' );

begin

process(score, x\_draw, y\_draw)

variable x\_pos, y\_pos : integer range 9 downto 0; -- position inside sprite

variable pixel : std\_logic;

begin

r <= "0000";

g <= "0000";

b <= "0000";

-- if within 20x30 box

if (unsigned(y\_draw) >= 50 and unsigned(y\_draw) < 80 and unsigned(x\_draw) >= 160 and unsigned(x\_draw) < 180) then

-- change this to 0 for digit 0, etc

if(score = "1") then

x\_pos := to\_integer(unsigned(x\_draw)) - 160; -- find pixel within digitArray

y\_pos := to\_integer(unsigned(y\_draw)) - 50;

pixel := dg0((y\_pos\*20) + x\_pos);

if (pixel = '1') then

r <= "1111";

g <= "1111";

b <= "1111";

end if;

**Ball**

entity ball is

Port ( x\_loc : in STD\_LOGIC\_VECTOR (9 downto 0); -- position on screen

y\_loc : in STD\_LOGIC\_VECTOR (9 downto 0);

x\_draw : in STD\_LOGIC\_VECTOR (9 downto 0); -- electron beam position

y\_draw : in STD\_LOGIC\_VECTOR (9 downto 0);

r : out STD\_LOGIC\_VECTOR (3 downto 0);

g : out STD\_LOGIC\_VECTOR (3 downto 0);

b : out STD\_LOGIC\_VECTOR (3 downto 0));

end ball;

architecture Behavioral of ball is

type spriteArray is array (positive range <>) of std\_logic\_vector(11 downto 0);

-- 5x7=35

signal ball : spriteArray(0 to 34) := (

x"FFF", x"FFF", x"FFF", x"FFF", x"FFF", x"FFF", x"FFF",

x"FFF", x"FFF", x"FFF", x"FFF", x"FFF", x"FFF", x"FFF",

x"FFF", x"FFF", x"FFF", x"FFF", x"FFF", x"FFF", x"FFF",

x"FFF", x"FFF", x"FFF", x"FFF", x"FFF", x"FFF", x"FFF",

x"FFF", x"FFF", x"FFF", x"FFF", x"FFF", x"FFF", x"FFF"

);

begin

process(x\_loc, y\_loc, x\_draw, y\_draw)

variable x\_pos, y\_pos : integer range 9 downto 0; -- position inside sprite

variable pixel : std\_logic\_vector(11 downto 0);

begin

r <= "0000";

g <= "0000";

b <= "0000";

-- if screen position >= start position of ball

if(unsigned(x\_draw) >= unsigned(x\_loc) and unsigned(y\_draw) >= unsigned(y\_loc)) then

-- if screen position < end position of ball (y + 5 and x + 7)

if(x\_draw < std\_logic\_vector(unsigned(x\_loc)+7) and y\_draw < std\_logic\_vector(unsigned(y\_loc)+5)) then

x\_pos := to\_integer(unsigned(x\_draw) - unsigned(x\_loc)); -- find pixel to draw in sprite (x = 0->6)

y\_pos := to\_integer(unsigned(y\_draw) - unsigned(y\_loc)); -- y = 0 -> 4

pixel := ball((y\_pos\*7) + x\_pos);

r <= pixel(11 downto 8);

g <= pixel(7 downto 4);

b <= pixel(3 downto 0);

end if;

end if;

end process;