การออกแบบวงจรควบคุมจอ VGA ด้วย FPGA

(Design and Implementation of VGA Controller on FPGA)

จอภาพหรือ Monitor

จอภาพ เป็นอุปกรณ์แสดงผลการทำงานของคอมพิวเตอร์หรืออุปกรณ์อื่นๆทั้งในรูปแบบของตัวอักษรและรูปภาพ การเชื่อมต่อระหว่างจอภาพกับคอมพิวเตอร์หรืออุปกรณ์อื่นๆ มีหลายแบบ เช่น VGA และ HDMI

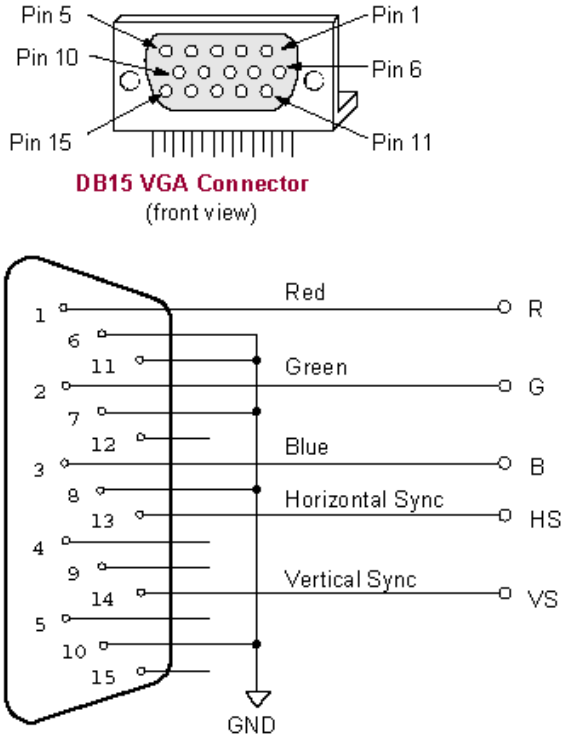
VGA คืออะไร

VGA ย่อมาจาก Video Graphic Adapter คืออุปกรณ์สำหรับควบคุมการแสดงผลของเครื่องคอมพิวเตอร์หรืออุปกรณ์อื่นๆ ออกทางจอภาพ ที่เราเรียกกันจนคุ้นหูว่า การ์ดจอ ซึ่งการ์ดจอจะเป็นตัวเชื่อมต่อระหว่างเมนบอร์ดกับจอภาพ จะช่วยให้หน้าจอสามารถแสดงผลได้อย่างเต็มประสิทธิภาพ

การทำงานขอจอภาพระบบ VGA

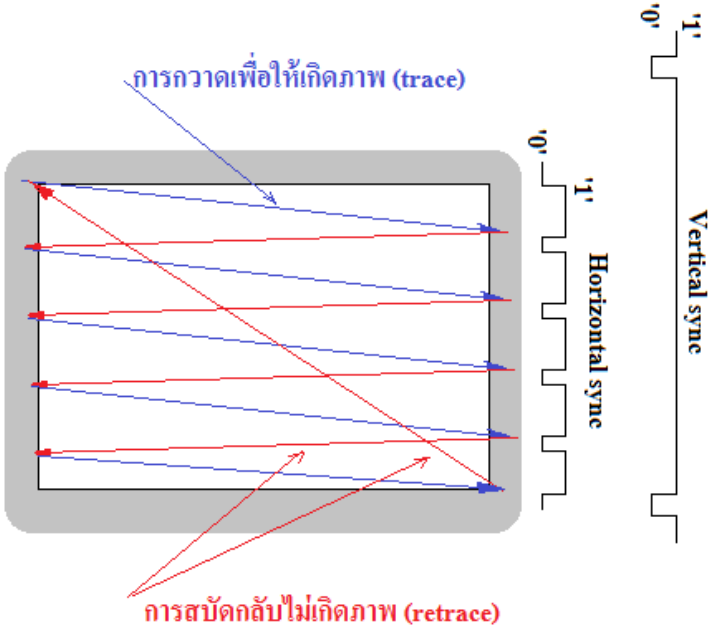
การควบคุมจอภาพระบบ VGA ใช้สัญญาณ 5 เส้น

- R ควบคุมค่าความสว่างของจุดสีแดง มีค่าได้ตั้งแต่ 0V ถึง 0.7 V คิดเป็นเปอร์เซ็นต์ความสว่าง 0 % - 100%
- G ควบคุมค่าความสว่างของจุดสีเขียว มีค่าได้ตั้งแต่ 0V ถึง 0.7 V คิดเป็นเปอร์เซ็นต์ความสว่าง 0 % - 100%
- B ควบคุมค่าความสว่างของจุดสีน้ำเงิน มีค่าได้ตั้งแต่ 0V ถึง 0.7 V คิดเป็นเปอร์เซ็นต์ความสว่าง 0 % - 100%
- Horizontal Synchronization (Hsync) ควบคุมการแสดงจุดภาพในแนวราบจากขอบซ้ายสุด ไปขอบขวาสุด ของจอภาพ แล้วกลับ ไปเริ่มต้นที่ขอบซ้ายสุดมาใหม่
- Vertical Synchronization (Vsync) ควบคุมการแสดงจุดภาพในแนวดิ่งจากด้านบนสุด ไปล่างสุดของจอภาพ แล้วกลับไปเริ่มต้นที่ด้านบนใหม่



รูปที่ 1 DB15 VGA Connector

การทำงานของสัญญาณ Vertical Sync และ Horizontal Sync



รูปที่ 2 ลักษณะการวาดจุดสีเพื่อให้เกิดเป็นภาพเมื่อเทียบกับสัญญาณ Hsync กับ Vsync

สัญญาณซิงค์ (Synchronous Signal)

จากการทำงานข้างต้น สัญญาณควบคุมจุดสี RGB ต้องมีช่วงควบคุมให้เกิดการแสดง (ลอจิก 1) กับควบคุมไม่ให้เกิดการแสดง(ลอจิก 0) เพื่อให้เกิดความเข้าใจของช่วงเวลาต่างๆขอยกตัวอย่างการควบคุมให้เกิดการแสดงภาพขนาด 640 x 480 จุด (pixel) และ fram rate 60 Hz

แนวระดับ (Horizontal)

horizontal display 640 จุด (ช่วงแสดงจุดภาพ)

hsync front porch 16 จุด (ช่วงไม่แสดงจุดภาพ)

hsync back porch 48 จุด (ช่วงไม่แสดงจุดภาพ)

hsync retrace 96 จุด (ช่วงไม่แสดงจุดภาพ)

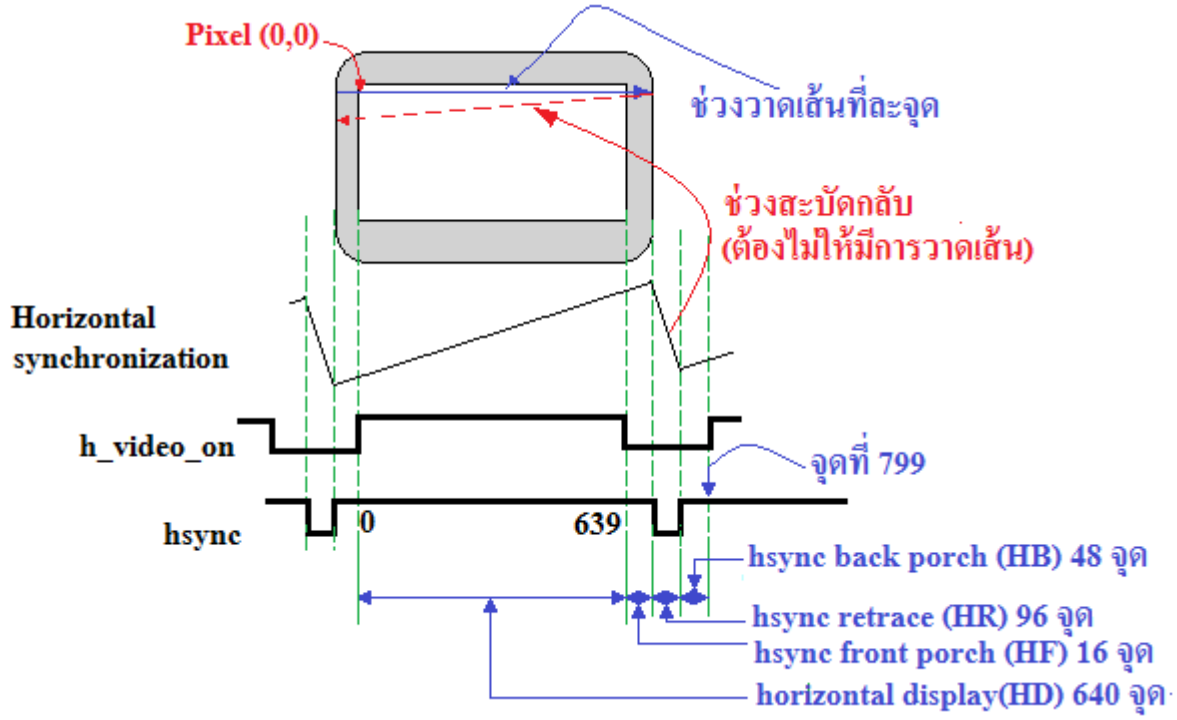
แนวตั้ง (Vertical)

vertical display 480 เส้น (ช่วงแสดงจุดภาพ)

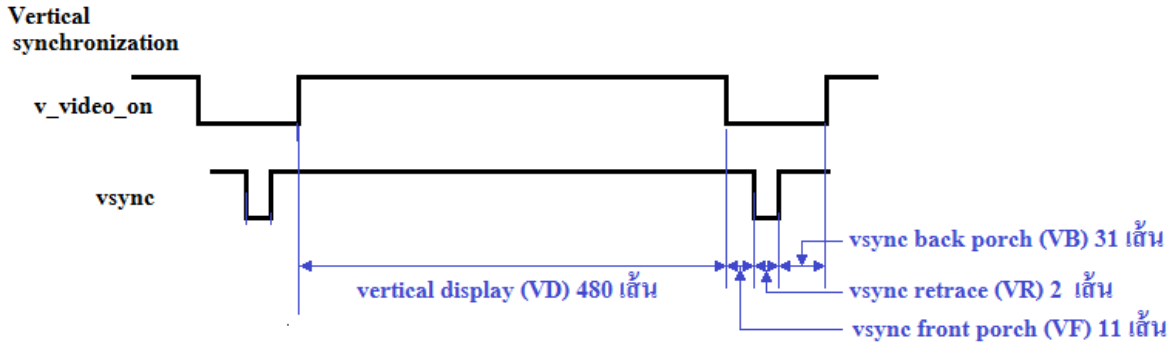
vsync front porch 11 เส้น (ช่วงไม่แสดงจุดภาพ)

vsync back porch 31 เส้น (ช่วงไม่แสดงจุดภาพ)

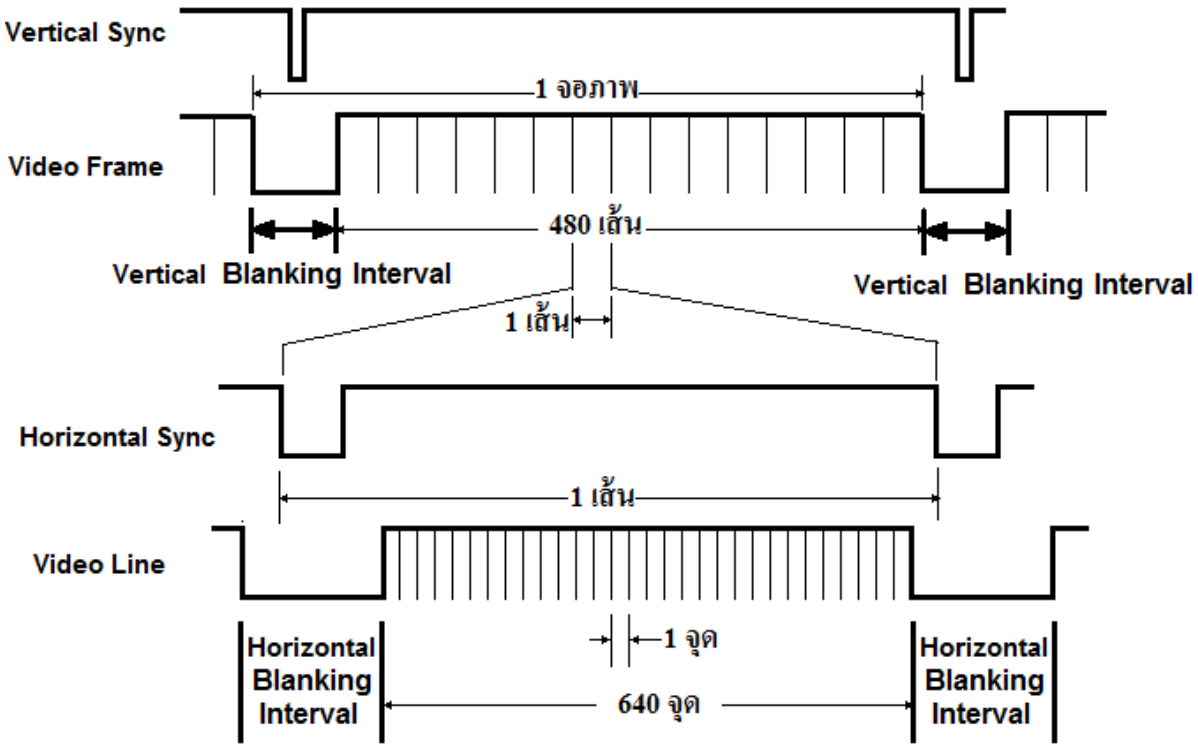
vsync retrace 2 เส้น (ช่วงไม่แสดงจุดภาพ)



รูปที่ 3 จำนวนจุดที่กวาดแนวระดับ



รูปที่ 4 จำนวนเส้นที่กวาดแนวตั้ง



รูปที่ 4 จำนวนจุดและจำนวนเส้นที่กวาดแนวระดับและแนวตั้ง

กรณีถ้าเป็นการแสดงภาพรูปแบบอื่นๆ จะมีค่าตามตารางนี้

Format		Horizontal (in Pixels)				Vertical (in Lines)				total Horizontal (in Pixels)	total Vertical (in Lines)	Pixel Clock (MHz)
	frame	Active Video	Front Porch	Sync Pulse	Back Porch	Active Video	Front Porch	Sync Pulse	Back Porch			
640x480,60Hz	60	640	16	96	48	480	11	2	31	800	524	25.152
640x480,50Hz	50	640	16	96	48	480	11	2	31	800	524	20.96
640x480,100Hz	100	640	16	96	48	480	11	2	31	800	524	41.92

หรือ

VGA Timings

The following table lists timing values for several popular resolutions.

Format	Pixel Clock (MHz)	Horizontal (in Pixels)				Vertical (in Lines)			
		Active Video	Front Porch	Sync Pulse	Back Porch	Active Video	Front Porch	Sync Pulse	Back Porch
640x480, 60Hz	25.175	640	16	96	48	480	11	2	31
640x480, 72Hz	31.500	640	24	40	128	480	9	3	28
640x480, 75Hz	31.500	640	16	96	48	480	11	2	32
640x480, 85Hz	36.000	640	32	48	112	480	1	3	25
800x600, 56Hz	38.100	800	32	128	128	600	1	4	14
800x600, 60Hz	40.000	800	40	128	88	600	1	4	23
800x600, 72Hz	50.000	800	56	120	64	600	37	6	23
800x600, 75Hz	49.500	800	16	80	160	600	1	2	21
800x600, 85Hz	56.250	800	32	64	152	600	1	3	27
1024x768, 60Hz	65.000	1024	24	136	160	768	3	6	29
1024x768, 70Hz	75.000	1024	24	136	144	768	3	6	29
1024x768, 75Hz	78.750	1024	16	96	176	768	1	3	28
1024x768, 85Hz	94.500	1024	48	96	208	768	1	3	36

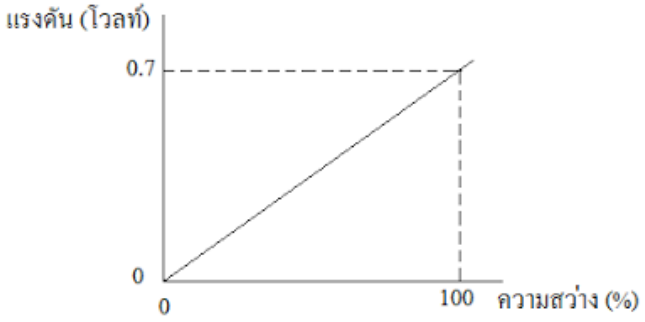
การสร้างจุดสี

จุดสีเกิดจากการผสมสีของ สีแดง (R: Red) เขียว (G: Green) และสีน้ำเงิน (B: Blue) ถ้าใช้ 8 บิต แทนแต่ละสี จะได้ ระดับความสว่างของแต่ละสีเป็น 256 ระดับ เมื่อเอามารวมกันทั้ง 3 สี จุดสีที่สามารถจะสร้างได้เป็น 256 x 256 x 256 = 16,777,216 สี

สำหรับการทดลองนี้ แต่ละสีใช้เพียง 1 บิต จึงมีค่าความสว่างของแต่ละสีเพียง 2 ระดับคือ มืดกับสว่างสุด เมื่อนำมาผสมกันจึงได้เท่ากับ 2 x 2 x 2 = 8 สี

Red	Green	Blue	สี
0	0	0	ดำ
0	0	1	น้ำเงิน
0	1	0	เขียว
0	1	1	ฟ้า
1	0	0	แดง
1	0	1	ม่วง
1	1	0	เหลือง
1	1	1	ขาว

ความสว่างของสีแต่ละสีเมื่อเทียบกับแรงดันที่ป้อนเข้าที่ขา R G และ B เมื่อใช้สัญญาณแต่ละสีเพียง 1 บิต



รูปที่ 5 ระดับความสว่างของแต่ละสีเมื่อเทียบกับแรงดันที่ป้อนให้ที่ขา RGB

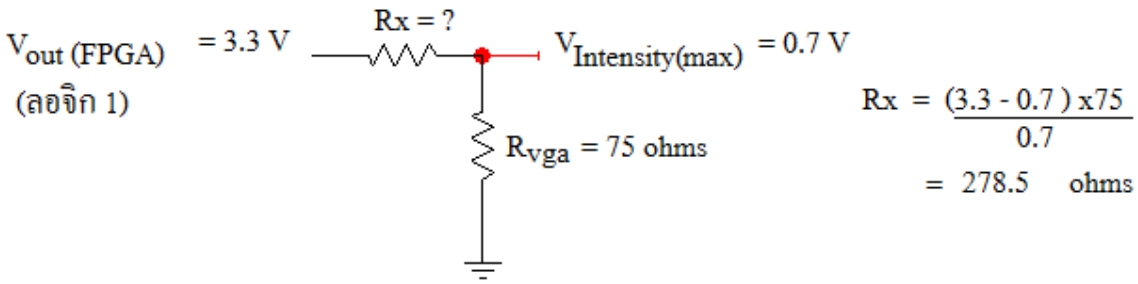
วงจร

ความต้านทานภายในของ ขา R G B ของ VGA แต่ละขา = 75 ohms

แรงดันที่ FPGA จ่ายเมื่อเป็นลอจิก '1' = 3.3 V

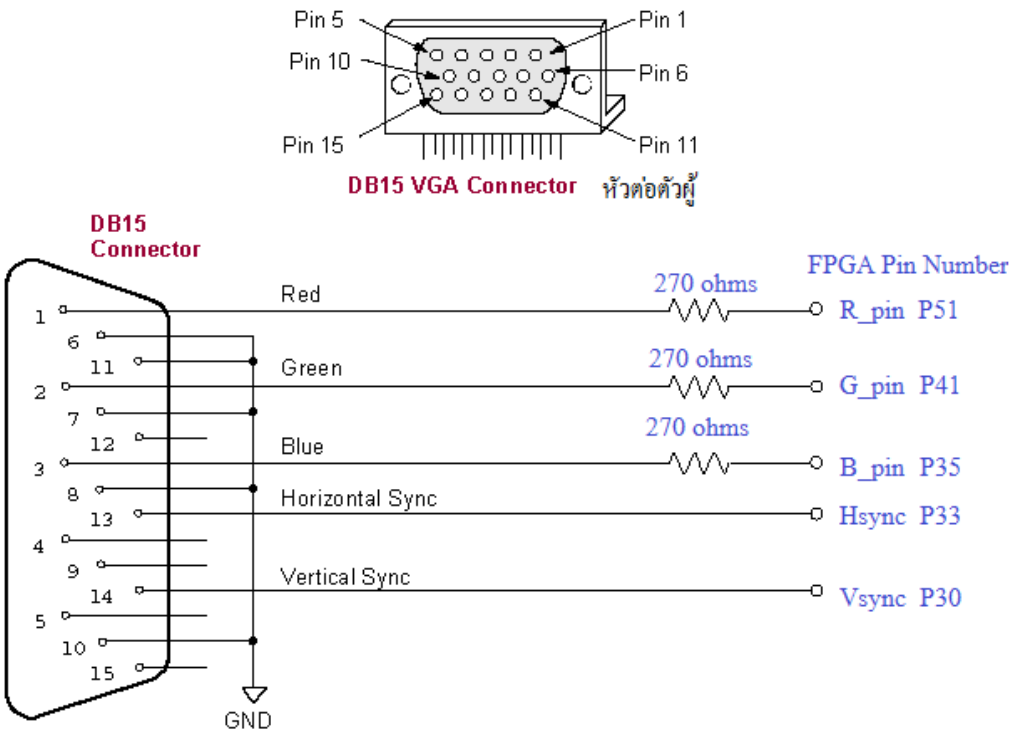
แรงดันที่ต้องการเพื่อให้ความสว่าง 100 % = 0.7 V

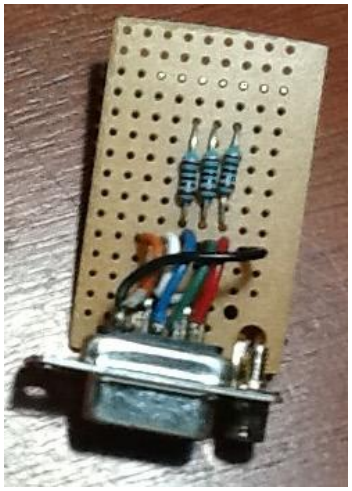
ดังนั้นต้องใช้ความต้านทานต่ออนุกรมระหว่า FPGA กับขา RGB แต่ละขาเท่ากับ

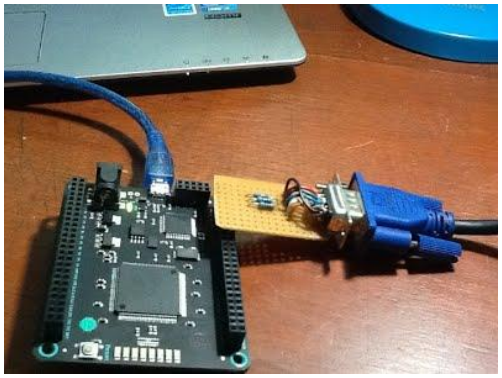


รูปที่ 6 การคำนวณหาค่าความต้านทาน

ในที่นี้ใช้ 250 โอหม์ หรือ 270 โอหม์

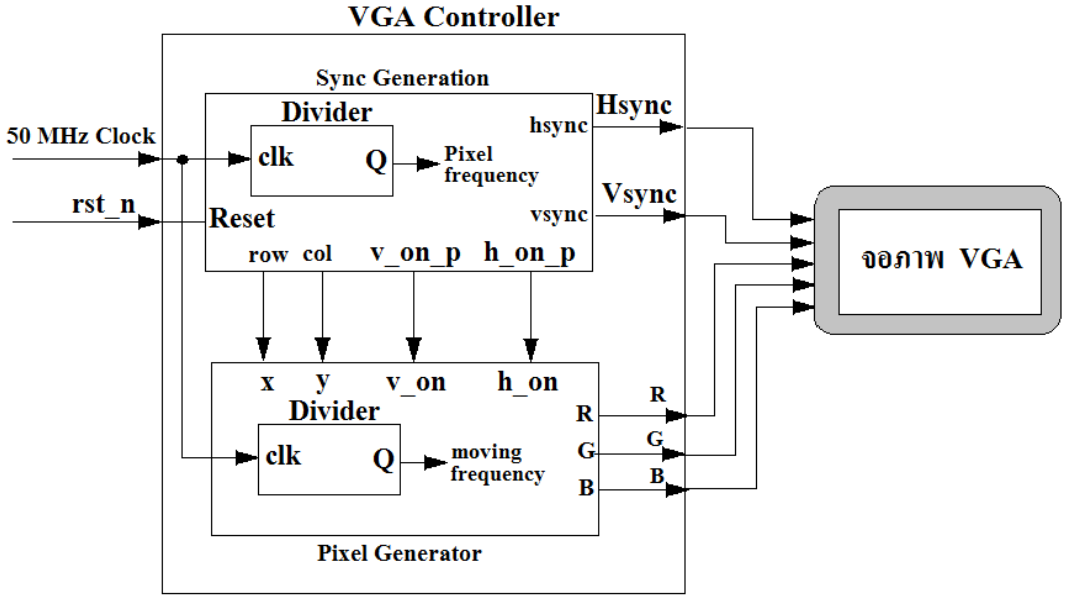






รูปที่ 6 การต่อความต้านทานเข้ากับหัวต่อ DB15

โมดูล VGA Controller



รูปที่ 7 ไดอะแกรมของ วงจรควบคุม VGD

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity vga_sync is
   port (clk, rst_n: in std_logic;
         hsync : out std_logic;
         h_on_p : out std_logic;
         vsync : out std_logic;
         v_on_p : out std_logic;
         row : out natural;
         col : out natural
         );

end vga_sync;

architecture arch of vga_sync is
         constant HF: integer:= 16; -- hsync front porch
         constant HR: integer:= 96; -- hsync retrace
         constant HB: integer:= 48; -- hsync back porch
         constant HD: integer:= 640; -- horizontal display

         constant VF: integer:= 11; -- vsync front porch
         constant VR: integer:= 2; -- vsync retrace
         constant VB: integer:= 31; -- vsync back porch
         constant VD: integer:= 480; -- vertical display

   signal v_freq, h_freq: std_logic;
         signal v_on, h_on: std_logic;
         signal reset: std_logic;

         signal x : natural := 0;
         signal y : natural := 0;
```

```
        begin
        -- Frequency dividre--
        DIVIDER1 : entity work.DIVIDER
   generic map(fin => 50000000,
                fout => 21000000)
   port map  (CLK=>clk,
               Q => h_freq );

        reset <= not rst_n;
        row <= y;
        col <= x;
        h_on_p <= h_on;
        v_on_p <= v_on;

-- Process begin –
---------Process การสร้างสัญญาณ Hor sync และ สัญญาณ ON/OFF Pixel ในแนวราบ----------

   h_gen: process(h_freq,reset)
        variable h_count : natural := 0;
        begin
                if(reset = '1') then h_count := 0; end if;
                if h_freq'event and h_freq = '1' then

                        if h_count <= HF then
                                h_on <= '0';
                                hsync <= '1';
                        elsif (h_count > HF) and (h_count <= HF+HR) then
                                h_on <= '0';
                                hsync <= '0';
                        elsif (h_count > HF+HR) and (h_count <= HF+HR+HB) then
                                h_on <= '0';
                                hsync <= '1';
                        elsif (h_count > HF+HR+HB) and (h_count <= HF+HR+HB+HD) then
                                h_on <= '1';
                                hsync <= '1';
                        elsif (h_count > HF+HR+HB+HD) then
                                h_count := 0;
                                h_on <= '0';
                                hsync <= '1';
                        end if;
                        h_count := h_count + 1;
                end if;
        end process;

---------Process การสร้างสัญญาณ Ver sync และ สัญญาณ ON/OFF Pixel ในแนวดิ่ง ----------
 v_gen: process(h_on,h_freq,reset)
        variable v_count : natural := 0;
        begin
          if(reset = '1') then v_count := 0; end if;
                if h_on'event and h_on = '0' then
                        if v_count <= VF then
                                v_on <= '0';
                                vsync <= '1';
                        elsif (v_count > VF) and (v_count <= VF+VR) then
                                v_on <= '0';
                                vsync <= '0';
                        elsif (v_count > VF+VR) and (v_count <= VF+VR+VB) then
```

```
                        v_on <= '0';
                        vsync <= '1';
                elsif (v_count > VF+VR+VB) and (v_count <= VF+VR+VB+VD) then
                        v_on <= '1';
                        vsync <= '1';
                elsif (v_count > VF+VR+VB+VD) then
                        v_count := 0;
                        v_on <= '0';
                        vsync <= '1';
                end if;
                v_count := v_count + 1;
        end if;
    end process;

----------Process การสร้าง ตำแหน่ง row และ column ของจุดสี----------
  xy_gen: process(v_on,reset)

    begin
    if h_freq'event and h_freq = '1' then
        if((h_on='0') and (v_on='0')) then
            x<=0;
            y<=0;
        elsif((h_on='1')and (v_on='1')) then
            x <= x+1;
            if(x>=639) then
              x <= 0;
                y <= y+1;
              if(y>=479) then
                        y<=0;
                end if;
           end if;
        end if;
    end if;
    end process;
end arch;
```

ตัวอย่าง แบบที่ 1 สร้างสี่เหลี่ยมสีแดงขนาด 50x50 pixel ที่ตำแหน่ง row = 50 col = 50 พื้นหลังสีขาว

```
-- Engineer:     Narong Buabthong
--
-- Create Date:    12/12/2015
-- Design Name: :VGA Controller
-- Module Name: : Pixel Generator  and top module  type 1
--
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity vga_controller is
  port (clk, rst_n: in std_logic;
        hsync : out std_logic;
        vsync : out std_logic;
        r_pin : out std_logic;
        g_pin : out std_logic;
        b_pin : out std_logic
        );
```

```
end vga_controller;

architecture arch of vga_controller is
        signal x : natural := 0;
        signal y : natural := 0;
        signal h_on : std_logic;
        signal v_on : std_logic;
        signal s_r : std_logic;
        signal s_g : std_logic;
        signal s_b : std_logic;
begin

---------เชื่อมต่อกับ โมดูล Sync Generator -------------
vga_sync : entity work.vga_sync
  port map (clk => clk,
            rst_n => rst_n,
            hsync => hsync,
            h_on_p => h_on,
            vsync => vsync,
            v_on_p => v_on,
            row  => y,
            col  => x );

---------Process การสร้างสัญญาณ pixel R G และ B ในแต่ละตำแหน่ง (row, col) -------------
-- Gen pixel --
r_pin <= s_r and h_on and v_on;
g_pin <= s_g and h_on and v_on;
b_pin <= s_g and h_on and v_on;
gen_pix: process(clk,x,y)
begin
        if clk'event and clk = '1' then
                if ((x>=50) and (x<100) and (y>=50) and (y<100)) then
                        s_r <= '1';
                        s_g <= '0';
                        s_b <= '0';
                else
                        s_r <= '1';
                        s_g <= '1';
                        s_b <= '1';
                end if;
        end if;
end process;
end arch;
```

VGA_Controller.ucf

```
NET "clk" TNM_NET = clk;
TIMESPEC TS_clk = PERIOD "clk" 50 MHz HIGH 50%;

NET "clk"   LOC = P56  | IOSTANDARD = LVTTL;
NET "rst_n" LOC = P38 | PULLUP;
NET "hsync" LOC = P33 | IOSTANDARD = LVTTL;
NET "vsync" LOC = P30 | IOSTANDARD = LVTTL;
NET "r_pin" LOC = P51  | IOSTANDARD = LVTTL;
NET "g_pin" LOC = P41 | IOSTANDARD = LVTTL;
NET "b_pin" LOC = P35 | IOSTANDARD = LVTTL;
```