VHDL 및 실습

VGA Pattern Generator

과목	VHDL 및 실습
학과	전자공학과
학번	2011144024
이름	유대성 (오전)
제출일	
담당교수	최종성 교수님

1. 주제 및 배경이론

VGA Display 에 대해 알고 이를 통해 원하는 결과를 얻어낼 수 있는 방법에 대해 살펴보자.

DE2 Board VGA DAC

우리가 사용하는 DE2 보드의 VGA 출력 관련 DAC(Digital to Analog Convertor)이다. 제품의 상세 스펙은 아래와 같다.

- CMOS(Complementary metal - oxide - semiconductor) : 집적 회로의 한 종류로 디지털 회로를 구성하는 데 사용한다. TTL 논리 소자에 비해 소비 전력이 적은 논리 회로를 구현할 수 있다. '상보성 금속 산화막 반도체'라는 용어로도 사용된다.
- ADV7123 : 140MHz Triple-10bit High Speed Video DAC : ADV7123 이라는 140MHz 3 * 10bit DA 컨버터를 사용한다.
- 100Hz 의 주파수 조건에서 최대 1600 x 1200 해상도 지원
- 15 핀의 고밀도 D-sub 커넥터를 사용한다.
- Cyclone II FPGA 와 함께 고성능 출력 인코더를 구현할 수 있다.

VGA Display

디스플레이란 화면을 통해 보여주는 기기이다. VGA 는 Video Graphics Array 라는 디스플레이 표준을 의미한다. 디스플레이에 대한 상세한 설명을 아래 적도록 한다.

- VGA 디스플레이는 기본적으로 빛의 3 원색인 RGB 컬러를 사용한다. 컬러는 10bit 기준으로 각 컬러당 2^{10}개의 범위의 컬러를 표현 할 수 있으며 RGB 조합 시 2^{30}개의 컬러의 경우의 수로 만들어 낼 수 있다.
- 기본적으로 사용하는 색상코드는 8bit 로 #000000 인 검정부터 #ffffff 인 흰색까지 표현한다.
- 화면의 한 픽셀 단위로 색상을 구현 할 수 있으며 이 픽셀은 역시 RGB 컬러로 구분된다.
- 화면 또한 클럭에 의해 동작하며 이 클럭은 다시 Horizontal Sync(HS, H cnt), Vertical Sync(VS, V cnt)로 나뉜다. 각 Sync 는 가로, 세로의 방향성을 가지고 있으며 HS 의 카운트는 Width 픽셀만큼 카운트 되고 VS 의 카운트는 Height 픽셀만큼 카운트 된다.
- VGA 의 모든 픽셀이 화면에 보이는 것은 아니다. 640 x 480 해상도에서 전체 픽셀 수는 800 x 525 정도로 꽤나 큰 공간을 차지하고 있다.
- 픽셀은 좌측 상단에서부터 우측 하단 까지 펄스를 통해 순차적으로 출력되고 (프로그레시브 방식) 화면에 보이는 구간의 펄스를 Active 영역이라고 한다. Active 구간 앞 뒤로는 porch 라는 미 출력 구간이 있는데 이는 화면 조정 시에 사용할 수 있는 공간을 말한다.
- HS, VS 의 카운팅 종료 시에 클럭 펄스를 발생시키는데 HS 는 96 픽셀, VS 는 2 라인 동안 펄스를 유지한다.

Altera Quartus II 에서는 프로젝트 생성을 Schematic / HDL 로 나누어 진행할 수 있지만 Schematic 은 다른 플랫폼으로의 변경에 어려움이 있다.

모델 설계

〈 주요 기능 〉

- Active 영역을 가로, 세로로 나누어 HS, VS 펄스를 다루어 본다.
- 가로, 세로를 각각 그리드로 나눠 새로운 단위의 픽셀을 만들고 그 픽셀을 통해 이름을 출력해보자.
- 각 모드를 2 개의 SEL 입력과 MUX 를 통해 제어하고 변경해보자.
-

〈 구현 방식 〉

- RGB 를 각각 다른 HS 구간으로 나누어 화면을 가로로 나누어 보자.
- RGB 를 각각 다른 VS 구간으로 나누어 화면을 가로로 나누어 보자.
- RGB 를 20*15 정도의 새로운 단위로 나누어 이름을 표시해 보자,
- 카운터와 MUX 를 통해 각 화면을 버튼으로 제어해 보자.

2. 소스코드 및 코드 설명(3, 4 단계 함께 기술)

소스코드를 설명과 함께 기술한다.

3. 시뮬레이션 결과 및 설명(3, 4 단계 함께 기술)

코드 시뮬레이션과 그 결과를 기술한다.

HS 를 통한 세로 나누기

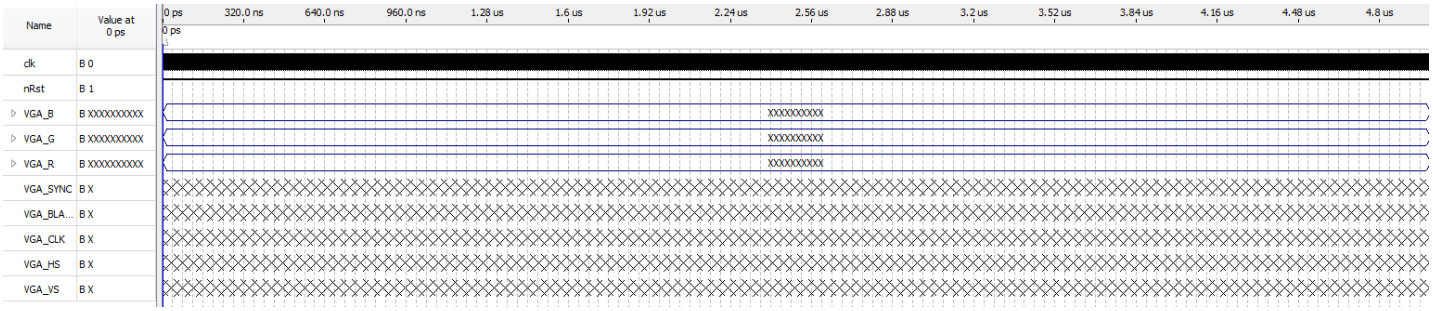
1. VHDL 작성

```vhdl
library ieee;
    use ieee.std_logic_1164.all;
    use ieee.std_logic_arith.all;
    use ieee.std_logic_unsigned.all;

entity VGA_Pattern_Generator is
    port(
        nRst : in std_logic;
        clk : in std_logic;
        VGA_CLK : out std_logic;
        VGA_BLANK : out std_logic;
        VGA_HS : out std_logic;
        VGA_VS : out std_logic;
        VGA_SYNC : out std_logic;
        VGA_R : out std_logic_vector(9 downto 0);
        VGA_G : out std_logic_vector(9 downto 0);
        VGA_B : out std_logic_vector(9 downto 0)
    );
end VGA_Pattern_Generator;

architecture BEH of VGA_Pattern_Generator is

    signal H_cnt : std_logic_vector(9 downto 0);
    signal V_cnt : std_logic_vector(9 downto 0);
    signal pclk : std_logic;

begin

    process(nRst, clk)
    begin
        if(nRst = '0') then
            pclk <= '0';
        elsif rising_edge(clk) then
            pclk <= not pclk;
        end if;
    end process;

    process(nRst, pclk)
    begin
        if(nRst = '0') then
            H_cnt <= (others => '0');
            V_cnt <= (others => '0');
        elsif rising_edge(pclk) then
            if(H_cnt = 799) then
                H_cnt <= (others => '0');
                if(V_cnt = 524) then
                    V_cnt <= (others => '0');
                else
                    V_cnt <= V_cnt + 1;
                end if;
            else
                H_cnt <= H_cnt + 1;
            end if;
        end if;
    end process;

    VGA_CLK <= pclk;
```

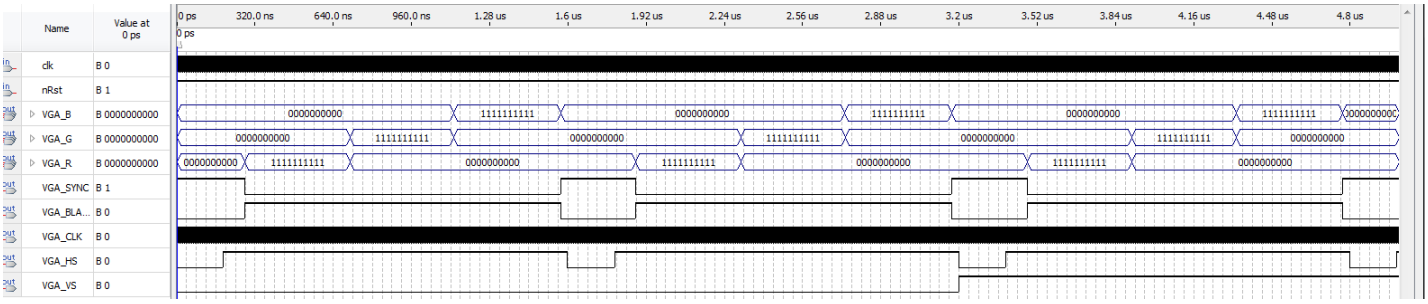
```
58    VGA_HS <= '0' when (H_cnt >= 0) and (H_cnt <= 95) else '1';
59    VGA_VS <= '0' when (V_cnt >= 0) and (V_cnt <= 1) else '1';
60    VGA_BLANK <= '1' when (H_cnt >= 140) and (H_cnt <= 790) else '0';
61    VGA_SYNC <= '0' when (H_cnt >= 140) and (H_cnt <= 790) else '1';
62    VGA_R <= "1111111111" when (H_cnt >= 149) and (H_cnt <= 359) else (others => '0');
63    VGA_G <= "1111111111" when (H_cnt >= 360) and (H_cnt <= 579) else (others => '0');
64    VGA_B <= "1111111111" when (H_cnt >= 580) and (H_cnt <= 789) else (others => '0');
65  end BEH;
```

Colored by Color Scripter

2. VMF 파일 생성 후 입력 펄스 조정



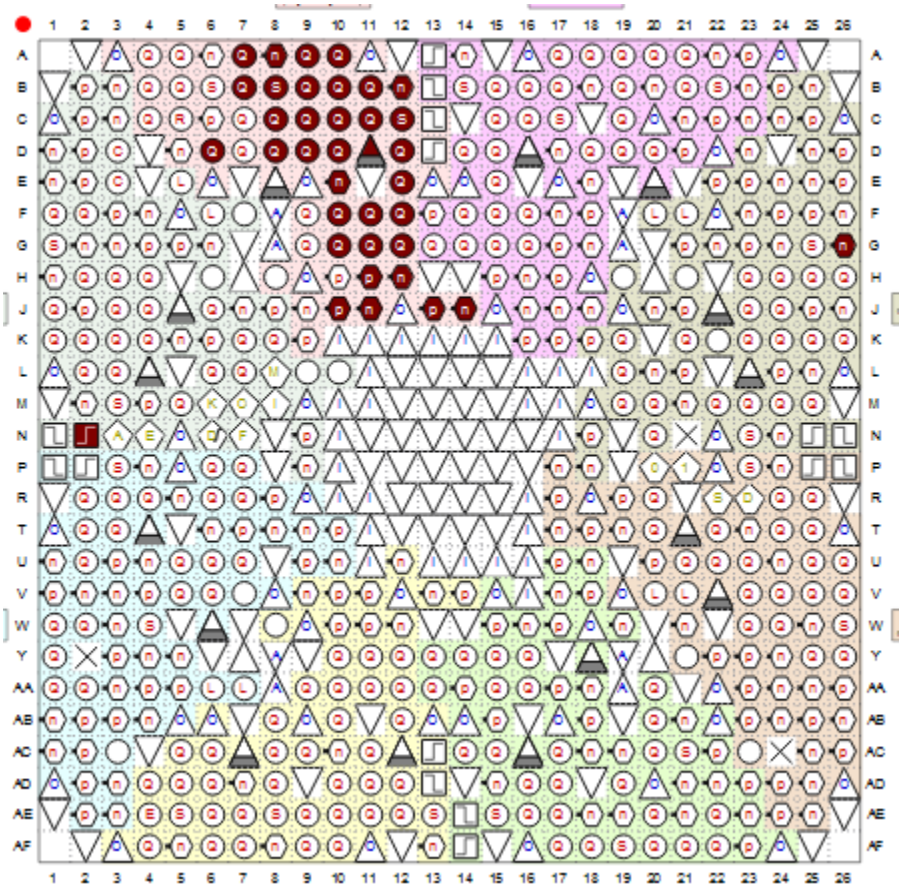
3. Function Simulating 을 통해 출력 펄스 확인



4. Pin Planner 를 통해 입출력 핀 설정

Node Name	Direction	Location
clk	Input	PIN_N2
nRst	Input	PIN_G26
VGA_B[9]	Output	PIN_B12
VGA_B[8]	Output	PIN_C12
VGA_B[7]	Output	PIN_B11
VGA_B[6]	Output	PIN_C11
VGA_B[5]	Output	PIN_J11
VGA_B[4]	Output	PIN_J10
VGA_B[3]	Output	PIN_G12
VGA_B[2]	Output	PIN_F12
VGA_B[1]	Output	PIN_J14
VGA_B[0]	Output	PIN_J13
VGA_BLANK	Output	PIN_D6
VGA_CLK	Output	PIN_B8
VGA_G[9]	Output	PIN_D12
VGA_G[8]	Output	PIN_E12
VGA_G[7]	Output	PIN_D11
VGA_G[6]	Output	PIN_G11
VGA_G[5]	Output	PIN_A10
VGA_G[4]	Output	PIN_B10
VGA_G[3]	Output	PIN_D10
VGA_G[2]	Output	PIN_C10
VGA_G[1]	Output	PIN_A9
VGA_G[0]	Output	PIN_B9
VGA_HS	Output	PIN_A7
VGA_R[9]	Output	PIN_E10
VGA_R[8]	Output	PIN_F11
VGA_R[7]	Output	PIN_H12
VGA_R[6]	Output	PIN_H11
VGA_R[5]	Output	PIN_A8
VGA_R[4]	Output	PIN_C9
VGA_R[3]	Output	PIN_D9
VGA_B[5]	Output	PIN_J11
VGA_B[4]	Output	PIN_J10
VGA_B[3]	Output	PIN_G12
VGA_B[2]	Output	PIN_F12
VGA_B[1]	Output	PIN_J14
VGA_B[0]	Output	PIN_J13
VGA_BLANK	Output	PIN_D6
VGA_CLK	Output	PIN_B8
VGA_G[9]	Output	PIN_D12
VGA_G[8]	Output	PIN_E12
VGA_G[7]	Output	PIN_D11
VGA_G[6]	Output	PIN_G11
VGA_G[5]	Output	PIN_A10
VGA_G[4]	Output	PIN_B10
VGA_G[3]	Output	PIN_D10
VGA_G[2]	Output	PIN_C10
VGA_G[1]	Output	PIN_A9
VGA_G[0]	Output	PIN_B9
VGA_HS	Output	PIN_A7
VGA_R[9]	Output	PIN_E10
VGA_R[8]	Output	PIN_F11
VGA_R[7]	Output	PIN_H12
VGA_R[6]	Output	PIN_H11
VGA_R[5]	Output	PIN_A8
VGA_R[4]	Output	PIN_C9
VGA_R[3]	Output	PIN_D9
VGA_R[2]	Output	PIN_G10
VGA_R[1]	Output	PIN_F10
VGA_R[0]	Output	PIN_C8
VGA_SYNC	Output	PIN_B7
VGA_VS	Output	PIN_D8
<<new node>>		

Pin 설정을 완료한 상태의 화면입니다.



VS 를 통한 세로 나누기

1. VHDL 작성

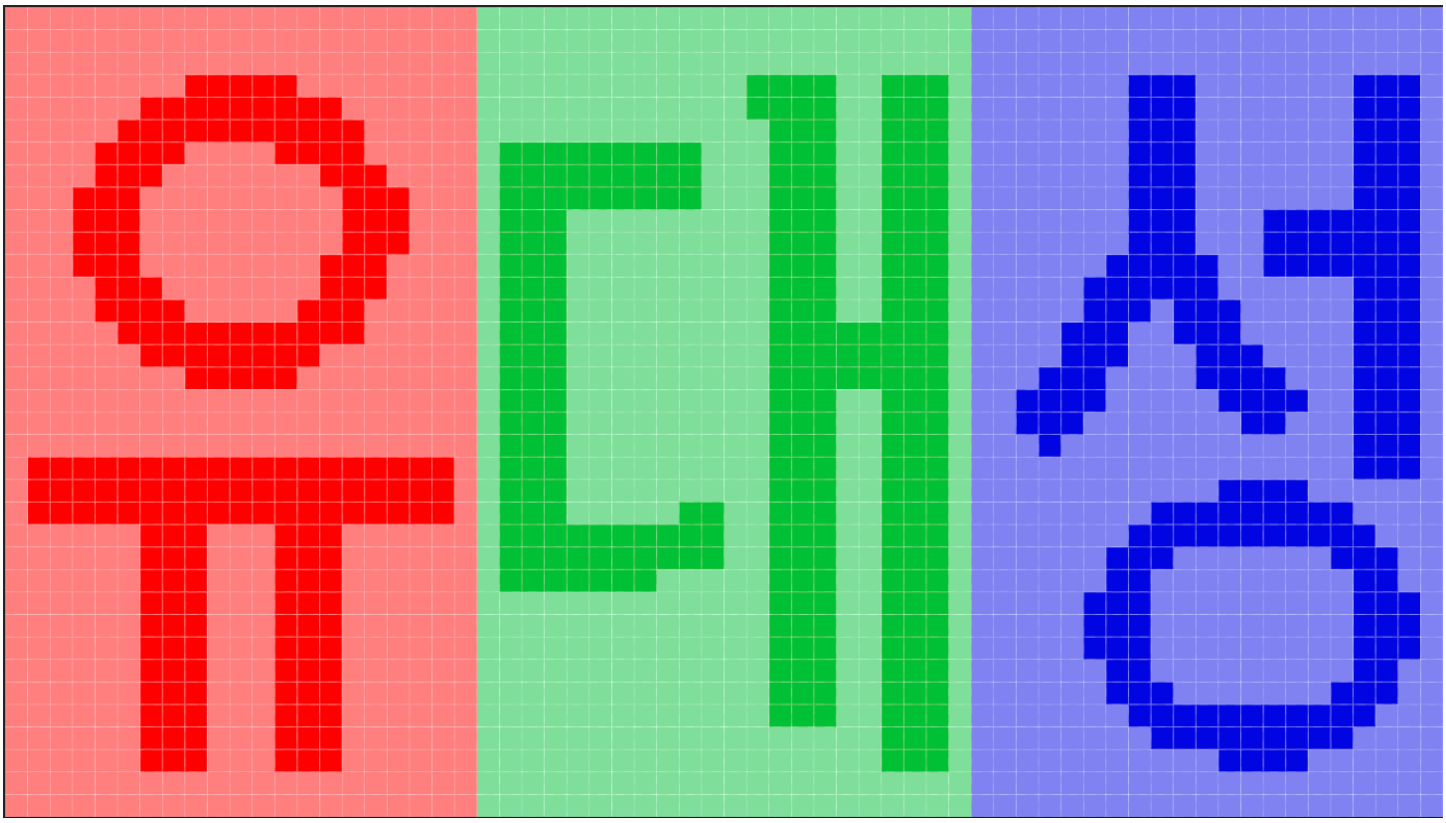
```
    VGA_CLK <= pclk;
    VGA_HS <= '0' when (H_cnt >= 0) and (H_cnt <= 95) else '1';
    VGA_VS <= '0' when (V_cnt >= 0) and (V_cnt <= 1) else '1';
    VGA_BLANK <= '1' when (H_cnt >= 140) and (H_cnt <= 790) else '0';
    VGA_SYNC <= '0' when (H_cnt >= 140) and (H_cnt <= 790) else '1';
    VGA_R <= "1111111111" when (V_cnt >= 31) and (V_cnt <= 153) else
             "1111111111" when (V_cnt >= 396) and (V_cnt <= 516) else (others => '0');
    VGA_G <= "1111111111" when (V_cnt >= 154) and (V_cnt <= 274) else
             "1111111111" when (V_cnt >= 396) and (V_cnt <= 516) else (others => '0');
    VGA_B <= "1111111111" when (V_cnt >= 275) and (V_cnt <= 395) else
             "1111111111" when (V_cnt >= 396) and (V_cnt <= 516) else (others => '0');
end BEH;

                                                    Colored by Color Scripter cs
```

중복되는 코드가 많아 핵심 코드만 작성하였습니다.

2. 이 후 단계는 파형 설정부터, 입출력 핀 설정까지 모두 같으므로 생략하였습니다.

그리드를 나누어 이름 표시하기

1. 픽셀로 이름 표현하기



2. VHDL 작성

```
VGA_CLK <= pclk;
VGA_HS <= '0' when (H_cnt >= 0) and (H_cnt <= 95) else '1';
VGA_VS <= '0' when (V_cnt >= 0) and (V_cnt <= 1) else '1';
VGA_BLANK <= '1' when (H_cnt >= 140) and (H_cnt <= 790) else '0';
VGA_SYNC <= '0' when (H_cnt >= 140) and (H_cnt <= 790) else '1';
VGA_R <=      "1111111111" when (H_cnt >= 229) and (H_cnt <= 279) and (V_cnt >= 51) and (V_cnt <= 60) else
              "1111111111" when (H_cnt >= 209) and (H_cnt <= 299) and (V_cnt >= 61) and (V_cnt <= 70) else
              "1111111111" when (H_cnt >= 199) and (H_cnt <= 309) and (V_cnt >= 71) and (V_cnt <= 80) else
              "1111111111" when (H_cnt >= 189) and (H_cnt <= 229) and (V_cnt >= 81) and (V_cnt <= 90) else
              "1111111111" when (H_cnt >= 269) and (H_cnt <= 309) and (V_cnt >= 81) and (V_cnt <= 90) else
              "1111111111" when (H_cnt >= 189) and (H_cnt <= 219) and (V_cnt >= 91) and (V_cnt <= 100) else
              "1111111111" when (H_cnt >= 289) and (H_cnt <= 319) and (V_cnt >= 91) and (V_cnt <= 100) else
              "1111111111" when (H_cnt >= 179) and (H_cnt <= 209) and (V_cnt >= 101) and (V_cnt <= 110) else
              "1111111111" when (H_cnt >= 299) and (H_cnt <= 329) and (V_cnt >= 101) and (V_cnt <= 110) else
              "1111111111" when (H_cnt >= 179) and (H_cnt <= 209) and (V_cnt >= 111) and (V_cnt <= 120) else
              "1111111111" when (H_cnt >= 299) and (H_cnt <= 329) and (V_cnt >= 111) and (V_cnt <= 120) else
              "1111111111" when (H_cnt >= 179) and (H_cnt <= 209) and (V_cnt >= 121) and (V_cnt <= 130) else
              "1111111111" when (H_cnt >= 299) and (H_cnt <= 329) and (V_cnt >= 121) and (V_cnt <= 130) else
              "1111111111" when (H_cnt >= 179) and (H_cnt <= 209) and (V_cnt >= 131) and (V_cnt <= 140) else
              "1111111111" when (H_cnt >= 289) and (H_cnt <= 319) and (V_cnt >= 131) and (V_cnt <= 140) else
              "1111111111" when (H_cnt >= 189) and (H_cnt <= 219) and (V_cnt >= 141) and (V_cnt <= 150) else
              "1111111111" when (H_cnt >= 289) and (H_cnt <= 319) and (V_cnt >= 141) and (V_cnt <= 150) else
              "1111111111" when (H_cnt >= 189) and (H_cnt <= 229) and (V_cnt >= 151) and (V_cnt <= 160) else
              "1111111111" when (H_cnt >= 279) and (H_cnt <= 309) and (V_cnt >= 151) and (V_cnt <= 160) else
              "1111111111" when (H_cnt >= 199) and (H_cnt <= 309) and (V_cnt >= 161) and (V_cnt <= 170) else
              "1111111111" when (H_cnt >= 209) and (H_cnt <= 289) and (V_cnt >= 171) and (V_cnt <= 180) else
              "1111111111" when (H_cnt >= 229) and (H_cnt <= 279) and (V_cnt >= 181) and (V_cnt <= 190) else

              "1111111111" when (H_cnt >= 159) and (H_cnt <= 349) and (V_cnt >= 221) and (V_cnt <= 230) else
              "1111111111" when (H_cnt >= 159) and (H_cnt <= 349) and (V_cnt >= 231) and (V_cnt <= 240) else
```

```
                    "1111111111" when (H_cnt >= 159) and (H_cnt <= 349) and (V_cnt >= 241) and (V_cnt <= 250) else
                    "1111111111" when (H_cnt >= 209) and (H_cnt <= 239) and (V_cnt >= 251) and (V_cnt <= 260) else
                    "1111111111" when (H_cnt >= 269) and (H_cnt <= 299) and (V_cnt >= 251) and (V_cnt <= 260) else
                    "1111111111" when (H_cnt >= 209) and (H_cnt <= 239) and (V_cnt >= 261) and (V_cnt <= 270) else
                    "1111111111" when (H_cnt >= 269) and (H_cnt <= 299) and (V_cnt >= 261) and (V_cnt <= 270) else
                    "1111111111" when (H_cnt >= 209) and (H_cnt <= 239) and (V_cnt >= 271) and (V_cnt <= 280) else
                    "1111111111" when (H_cnt >= 269) and (H_cnt <= 299) and (V_cnt >= 271) and (V_cnt <= 280) else
                    "1111111111" when (H_cnt >= 209) and (H_cnt <= 239) and (V_cnt >= 281) and (V_cnt <= 290) else
                    "1111111111" when (H_cnt >= 269) and (H_cnt <= 299) and (V_cnt >= 281) and (V_cnt <= 290) else
                    "1111111111" when (H_cnt >= 209) and (H_cnt <= 239) and (V_cnt >= 291) and (V_cnt <= 300) else
                    "1111111111" when (H_cnt >= 269) and (H_cnt <= 299) and (V_cnt >= 291) and (V_cnt <= 300) else
                    "1111111111" when (H_cnt >= 209) and (H_cnt <= 239) and (V_cnt >= 301) and (V_cnt <= 310) else
                    "1111111111" when (H_cnt >= 269) and (H_cnt <= 299) and (V_cnt >= 301) and (V_cnt <= 310) else
                    "1111111111" when (H_cnt >= 209) and (H_cnt <= 239) and (V_cnt >= 311) and (V_cnt <= 320) else
                    "1111111111" when (H_cnt >= 269) and (H_cnt <= 299) and (V_cnt >= 311) and (V_cnt <= 320) else
                    "1111111111" when (H_cnt >= 209) and (H_cnt <= 239) and (V_cnt >= 321) and (V_cnt <= 330) else
                    "1111111111" when (H_cnt >= 269) and (H_cnt <= 299) and (V_cnt >= 321) and (V_cnt <= 330) else
                    "1111111111" when (H_cnt >= 209) and (H_cnt <= 239) and (V_cnt >= 331) and (V_cnt <= 340) else
                    "1111111111" when (H_cnt >= 269) and (H_cnt <= 299) and (V_cnt >= 331) and (V_cnt <= 340) else
                    "1111111111" when (H_cnt >= 209) and (H_cnt <= 239) and (V_cnt >= 341) and (V_cnt <= 350) else
                    "1111111111" when (H_cnt >= 269) and (H_cnt <= 299) and (V_cnt >= 341) and (V_cnt <= 350) else
                    "1111111111" when (H_cnt >= 209) and (H_cnt <= 239) and (V_cnt >= 351) and (V_cnt <= 360) else
                    "1111111111" when (H_cnt >= 269) and (H_cnt <= 299) and (V_cnt >= 351) and (V_cnt <= 360) else
                    (others => '0');

    VGA_G <= "1111111111" when (H_cnt >= 360) and (H_cnt <= 579) else (others => '0');
    VGA_B <= "1111111111" when (H_cnt >= 580) and (H_cnt <= 789) else (others => '0');

end BEH;
```

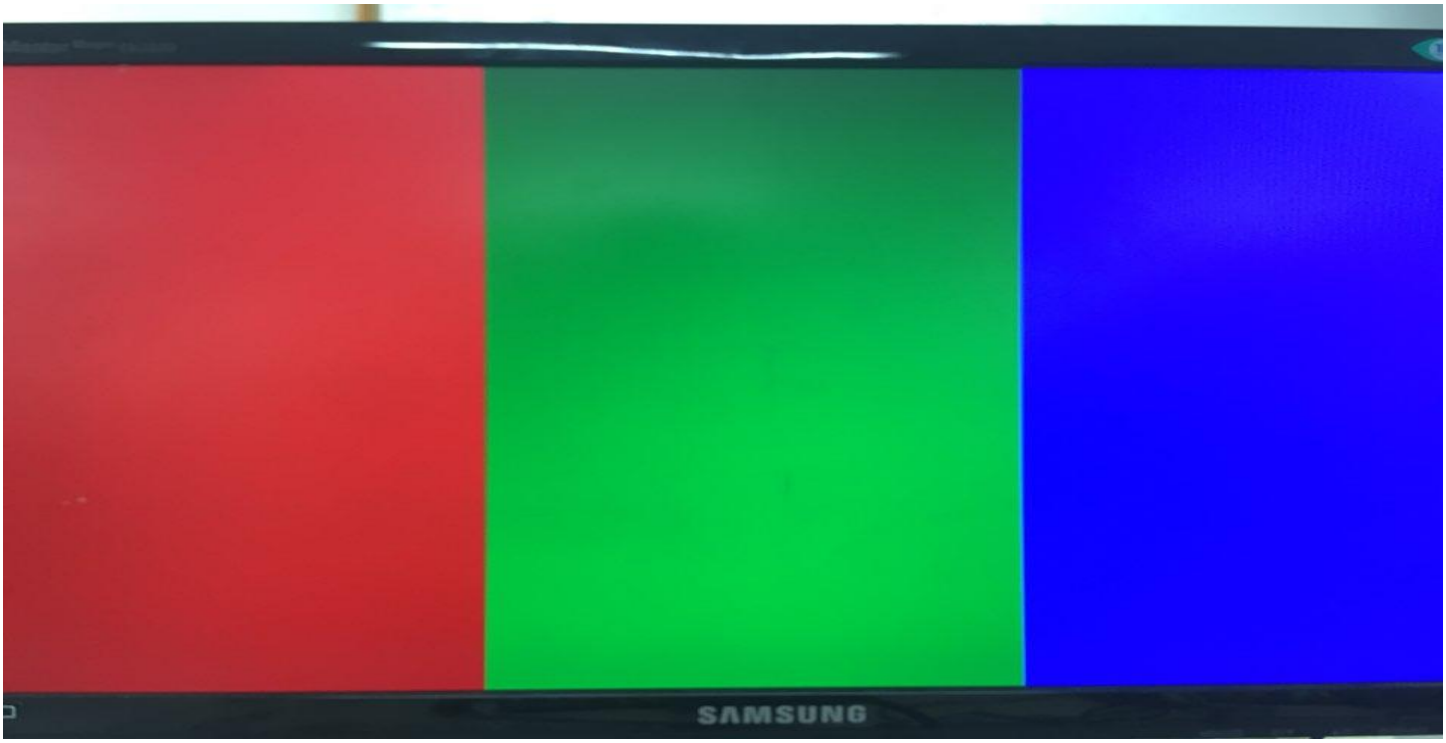

코드가 길어 핵심 코드만 작성하였고, '유'라는 단어를 나타내면서 해당 과제를 충분히 이해할 수 있어 더 이상의 소스코드는 작성하지 않았습니다.
시험이 끝나 여유가 생긴다면 도전해 보도록 하겠습니다.

3. 이 후 단계는 파형 설정부터, 입출력 핀 설정까지 모두 같으므로 생략하였습니다.

5. 실습보드 적용 결과

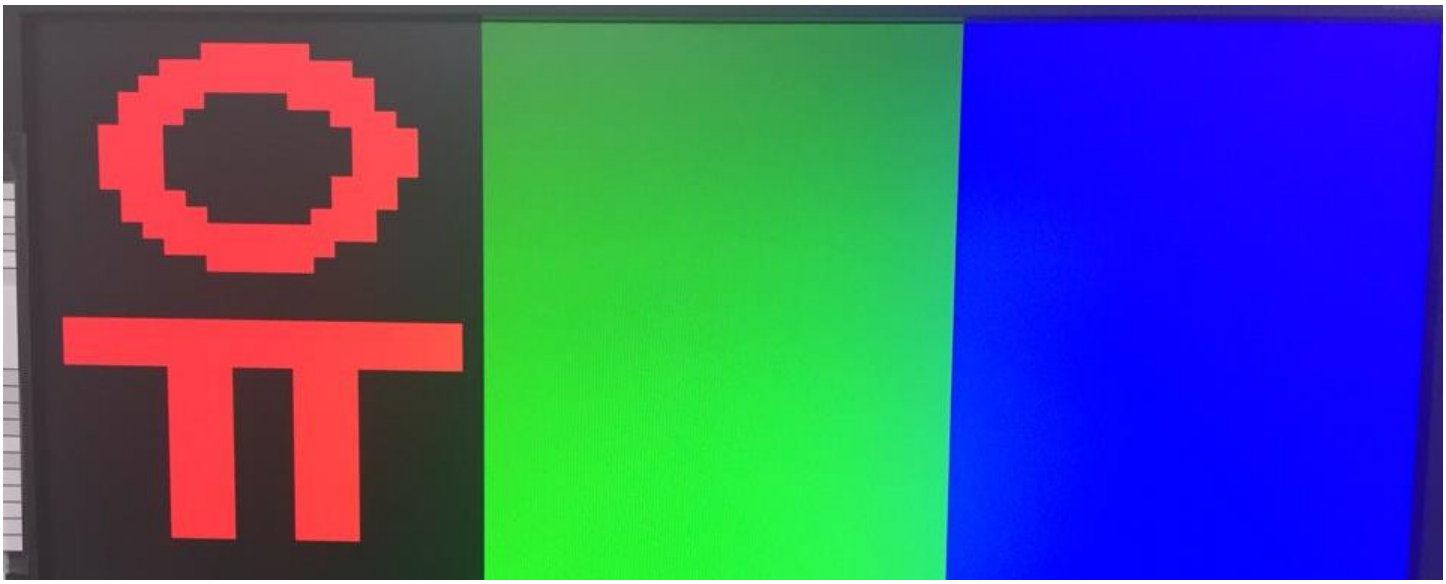
소스를 보드에 다운로드하여 예상한 결과에 맞게 동작하는 지 테스트하시오

HS 를 통한 세로 나누기



VS 를 통한 세로 나누기

그리드를 나누어 이름 표시하기



성을 표현하는 데 성공하였고 해당 과제를 충분히 이해할 수 있어 더 이상 진행하지 않았습니다. 시험이 끝난 이후 시간이 되는 대로 도전해보겠습니다.

6. 실습소감

실습을 통해 경험한 것을 자유롭게 서술하시오.

실습시간 외에는 보드를 사용할 수 없어 항상 급급하게 실습을 진행했는데 이번에 DE2 보드를 빌리게 되어 여유롭게 실습이 가능해졌다. 미리 구할 수 있었다면 이전 디지털 시계에서 사용할 수 있었을 텐데라는 아쉬움이 남지만 시험이 끝나는 대로 다운로드와 코드 수정을 해볼 계획이다. 이번 실습을 통해 보드 내에서 동작하는 것이 아닌 외부 하드웨어도 동작할 수 있다는 것을 보았고 키패드나 다른 외부기기를 추가해 여러가지 동작을 시킬 수 있다면 꽤나 만족스러운 결과물을 얻어 낼 수 있을 것 같다. 이번 실습을 하며 궁금했던 것은 코드 상단에 ieee 라는 표준으로 라이브러리 같은 것을 불러왔는데 혹시 ATMEGA128 정도의 사용자 층이 존재하는 보드라면 위 같이 이름을 쓰는 라이브러리 같은 게 존재하지 않을까? 라는 생각을 했다. 사실 귀찮음에서 비롯된 생각이었지만 이 같은 라이브러리가 있고 또 그걸 잘 이용할 수 있다면 좀 더 향상된 결과물을 만들어 낼 수 있을 것 같다.

7. 문의 사항

실습하다 겪은 어려웠던 점을 기술하시오.

이번 실습 시, 1 초 생성기와 같은 방식으로 top entity 에도 시간 폭을 변경하여 시뮬레이션 해보려고 했으나 function simulation 버튼을 클릭 시 다음 화면에서 멈춰져 결과 파형이 나타나지 않은 문제가 발생하였습니다. 방법을 몰라 대처하지 못했는데 이 점 질문 드립니다. 다음은 멈춤 현상이 발생한 부분의 캡쳐화면입니다.