# EEL 4712C - Digital Design: Lab Report 4

Cole Rottenberg
11062528

March 3rd, 2024

```vhdl
-- This is a test VHDL code block
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;

entity test is
    Port ( a : in  STD_LOGIC;
           b : in  STD_LOGIC;
           c : out  STD_LOGIC);
end test;
```

Listing 1: Test VHDL Code

## Lab Report

### Problem Statement

The lab is broken into 4 individual parts and two groups of two. The first part deals with static timing analysis and the second part deals with implementing a basic VGA driver to display a box on a monitor. The third part deals with implementing a VGA driver to display a moving box on a monitor. The fourth part deals applying the same static timing analysis from the first part to the moving box from the third part.

The 2nd and 3rd part of the lab are the critical design parts of the lab that help us explore the capabilities of the VGA driver and how to implement it. The first and fourth part are more about understanding the timing of the VGA driver and how to properly implement it. The part two is split into three different entities(primarily), the VGA, the VGA sync generator, and the top level component. The inputs to the top level component are relatively static and aren't changing. The outputs are the VGA signals that are sent to the monitor. The signals sent to the monitor are the h_sync, v_sync, and the RGB signals. The system is designed to display a box on the monitor. The sync generator is responsible for generating the sync signals, however the VGA entity is responsible for applying the logic to these signals to display the box on the monitor.

### Design

The VGA sync generator is consisted of a single clocked process that iterates through a double condition statement. The first conditionally block check if the `h_count` is equal the `H_MAX` constant defined in our package. If they are equal to eachother, we reset the counter and go onto the next conditional block, which checks if the `v_count` is equal to the `V_MAX` constant defined in our package. If they are equal to eachother, we reset the counter and exit the process. However, if we the first conditional is false, we increment the `h_count` counter. If the second conditional is false, we increment the `v_count` counter. On a conceptual level this builds a sweeping motion across the rows and then columns until the end. The second part of the generator exist outside a process as three conditions for the `h_sync`, `v_sync`, and `video_on` signals.

These signals are then passed up to the VGA entity which uses the h_count and v_count signals to determine the position of the box on the screen. The draw clocked process uses these counts to determine if the current pixel is within the define constants of: CENTER_X_START, CENTER_X_END, CENTER_Y_START, and CENTER_Y_END. If the pixel is within these bounds, the red, green, and blue signals are set to "0111", "0011", and "1011" respectively. Outside of the process, existing within the architecture, the h_sync, v_sync, and video_on signals are being outputted to the top level component.

The design of the 3rd part builds of the previous part as it uses an identical VGA sync generator. The VGA entity is modified and new values are used to move the box across the screen as well as change the direction of the box. We also need to make use of a clock divider to slow down the clock signal to 1Hz. This slow_clk signal is then used to drive our obj_move process. This process is responsible for moving the box across the screen. The obj_move process uses a new set of signals and constants to determine the position of the box on the screen. It also uses logic to control directional changes when the box reaches the vertical or horizontal bounds of the screen.

The top level entity controls the final output of the VGA signals to the monitor. The VGA entity passes the following outputs to the top level entity: h_sync, v_sync, red, green, blue, and video_on. The top level entity then passes these signals to the VGA port which is connected to the monitor.

### Implementation

### Testing

### Conclusions

## Appendix