JACOBS
UNIVERSITY

Jacobs University Bremen

CA-ECE-803: Digital design

Title: 7 Segment LED Driver

Lab Report: 3

Experiment conducted by:
Betelhem Nebebe
Maria Namachanja
Neeha Hammad

Instructor: Prof. Dr. Fangning Hu
Execution date: March 16, 2021

1. Introduction

The objective of this lab was to design a circuit in VHDL to display some hexadecimal values on the board's 7-segment decoder.

2. Requirements

The circuit will be implemented on a Spartan-3 Starter Kit board which includes a built-in four-digit seven-segment display, as well as some switches and buttons that will be used for the user interface. The task is to display the value of the leftmost 4 slide switches on the leftmost segment in hexadecimal. On the next segment, we are required to display the value of the rightmost 4 slide switches. In addition, we are required to display the arithmetic sum of the value of the two 4-bit slide switch sets astwo hex digits on the last next 2 segments.

3. Execution

One of the tasks was to calculate the display refresh rate. Since the anode control is operated through the counter and we know that for the time it takes the timer to fill all 15bits each digit turns on exactly once. We can think of this as the entire display refreshing once every 2^{15} counts. Since counting happens with 50Mhz frequency we need to find how many times the counter will count up to $2^{15} - 1$ within one second. The resulting calculation is given by:

refresh rate = 50Mhz / 2^{15}
= 1525 refreshes/second

We designed the remaining circuit blocks which add up to the entire circuit diagram shown on Figure 5. The circuit blocks we designed are able to handle addition of two 4-bit numbers, decide which character needs to be displayed and produce the bit pattern that would light the desired segments. In order to add the two 4-bit numbers we padded them with zeros on the left so that the inputs and output of the addition are always 8-bit long. The output of this operation is then fed as input to the character-selection mux along with the switch values. As a control signal for this mux we again use the position signal coming from the clock divider circuit. The output of this mux is the character that needs to be displayed on the currently enabled display digit. It is fed as a control signal to another mux that ultimately decides the bit pattern that needs to be sent to the 7-segment decoder. The mux in question is the rightmost mux on Figure 1 and its inputs have been omitted for the sake of diagram readability. A corresponding character - bit pattern map is provided below.

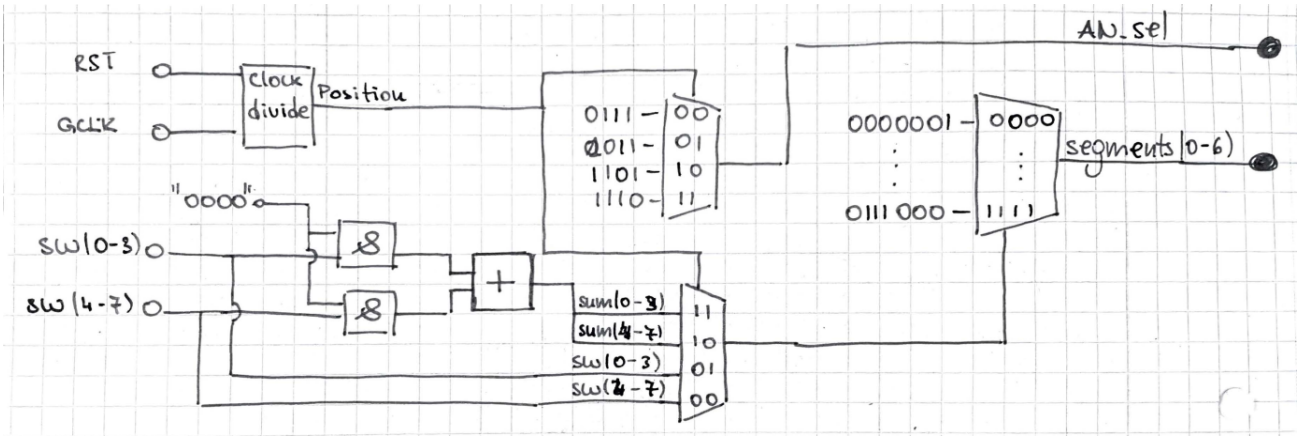


Figure 1

Codes

Initially, when we tried to run the given code, we were getting some warnings related to switches and unused segments.

Updated VHD File:

```
library ieee;
use ieee.STD_LOGIC_1164.all;
use ieee.numeric_std.all;

entity seg_ctrl is
  port(
    gclk     : in  STD_LOGIC;                    -- Global clock input, 50MHz
    rst      : in  STD_LOGIC;                   -- Global reset input
    switches : in STD_LOGIC_VECTOR(7 downto 0);  -- Input switches
    segments : out STD_LOGIC_VECTOR(6 downto 0); -- Segment control
    an_sel   : out STD_LOGIC_VECTOR(3 downto 0)  -- Anode control
    );
end seg_ctrl;

architecture rtl of seg_ctrl is

  -- A 15 bit counter
  signal count : unsigned(14 downto 0);
  signal position : unsigned(1 downto 0);
  -- Storing variables
  signal sumlr : unsigned (7 downto 0);
  signal digit1 : unsigned (3 downto 0);

 begin
  -- Clock divider
  process (gclk, rst)
  begin
    if gclk='1' and gclk'event then
      if rst='1' then
        count <= (others=>'0');       asynchronous reset
      else
        count <= count+1;
      end if;
         end if;
  end process;

  -- Anode decode circuitry
  position <= count(14 downto 13);

  with position select
```

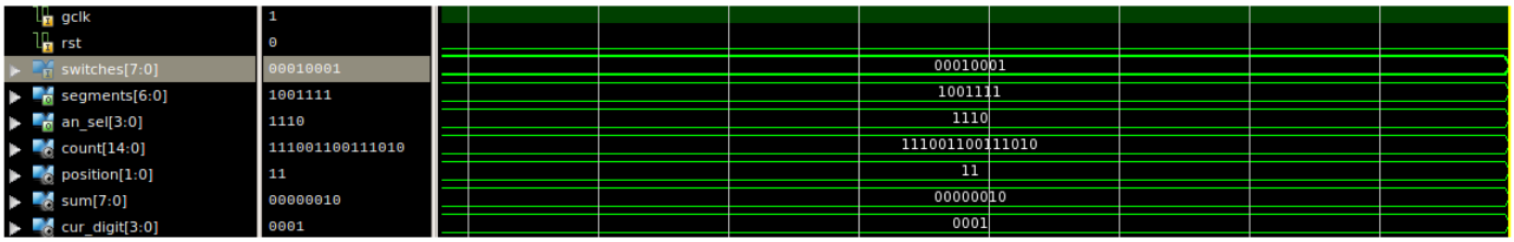
```
 an_sel <= --anodes are low asserted
  "0111" when "00",
  "1011" when "01",
  "1101" when "10",
  "1110" when others;

       with position select
       digit1 <= -- this part was fixed by looking at the output on the FPGA board
and changing order
       sumlr(7 downto 4) when "00",
       sumlr(3 downto 0) when "01",
       unsigned(switches(7 downto 4)) when "10",
       unsigned(switches(3 downto 0)) when others;

       with digit1 select --select segments to be lit up with each value (hard coded)

 segments <= "0000001" when "0000",
                          "1001111" when "0001",
                          "0010010" when "0010",
                          "0000110" when "0011",
                          "1001100" when "0100",
                          "0100100" when "0101",
                          "0100000" when "0110",
                          "0001111" when "0111",
                          "0000000" when "1000",
                          "0000100" when "1001",
                          "0001000" when "1010",
                          "1100000" when "1011",
                          "0110001" when "1100",
                          "1000010" when "1101",
                          "0110000" when "1110",
                          "0111000" when others;


 process(switches(3 downto 0), switches(7 downto 4)) -- sum process
 begin
 sumlr <= ("0000" & unsigned(switches(3 downto 0)))+("0000" &
unsigned(switches(7 downto 4)));

 end process;

end rtl;
```

UCF File

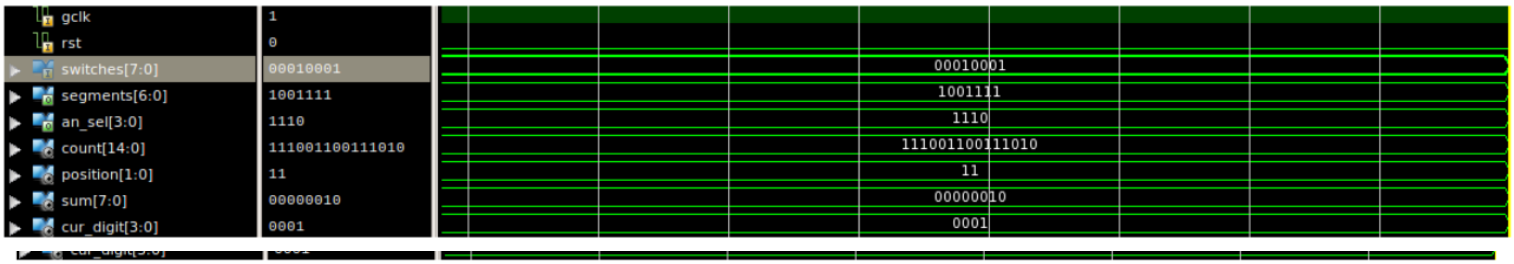
In this project we used 1 push button for the reset, 8 switches for number selection and of course the ports associated with the segments and anode selection.

```
#Connect ports to FPGA pins as specified in the Spartan3 starter board manual
NET "switches<0>" LOC="F12" ;
NET "switches<1>" LOC="G12" ;
NET "switches<2>" LOC="H14" ;
NET "switches<3>" LOC="H13" ;

NET "switches<4>" LOC="J14" ;
NET "switches<5>" LOC="J13" ;
NET "switches<6>" LOC="K14" ;
NET "switches<7>" LOC="K13" ;

NET "gclk" LOC="T9";
NET "rst" LOC="M13";



NET "an_sel<0>" LOC = "D14";
NET "an_sel<1>" LOC = "G14";
NET "an_sel<2>" LOC = "F14";
NET "an_sel<3>" LOC = "E13";



NET "segments<0>" LOC = "N16";
NET "segments<1>" LOC = "F13";
NET "segments<2>" LOC = "R16";
NET "segments<3>" LOC = "P15";
NET "segments<4>" LOC = "N15";
NET "segments<5>" LOC = "G13";
NET "segments<6>" LOC = "E14";
```

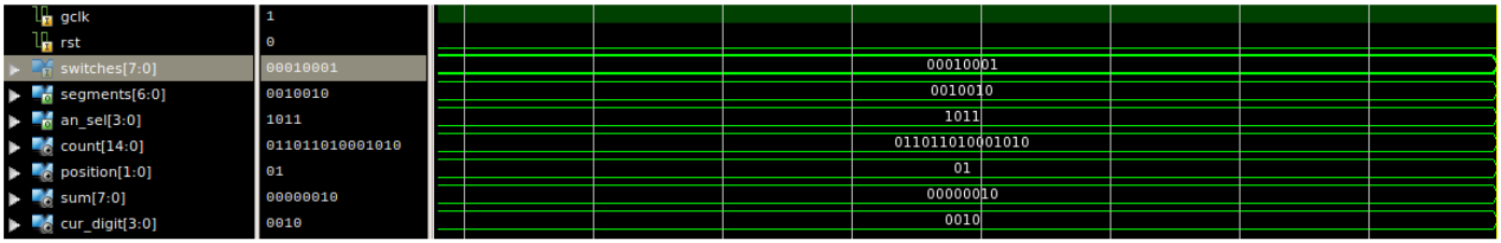
We noticed that the count and position values change regularly. The gclk values also shift between 1 and 0. When the first digit is active we get the following results:



When the second digit is active we get the following results:



When the third digit is active we get the following results:



4. Problems Faced

UCF files are fairly easy but tedious to generate by hand since one needs to name every port individually. We also had a problem while connecting a wire to the Spartan board since we were initially trying to plug it in the wrong direction.

5. Conclusion

The main purpose of the lab was to show a simple implementation of a simple VHDL logic into our FPGA board. With the help of the tasks done in other labs and with the help of the lab webpage, we managed to achieve the goal.

References

http://fpga-fhu.user.jacobs-university.de/?page_id=36