Cyclone IV EP4CE115F29 device

Ludovic Provost 300208450

Dexter Day 300192263

Group #16

Laboratory 1: Introduction to VHDL and ASM design

CEG3155: Digital Systems II

Prof. Rami Abielmona

Due October 4th 2023

Objectives:

The main objective of this laboratory is to complete a design in ASM and to implement it using VHDL.

- Design, realize and test a circuit using RTL;
- Design a solution to a given problem using the ASM methodology;
- Demonstrate a complete understanding for circuit realization using ASM and VHDL.

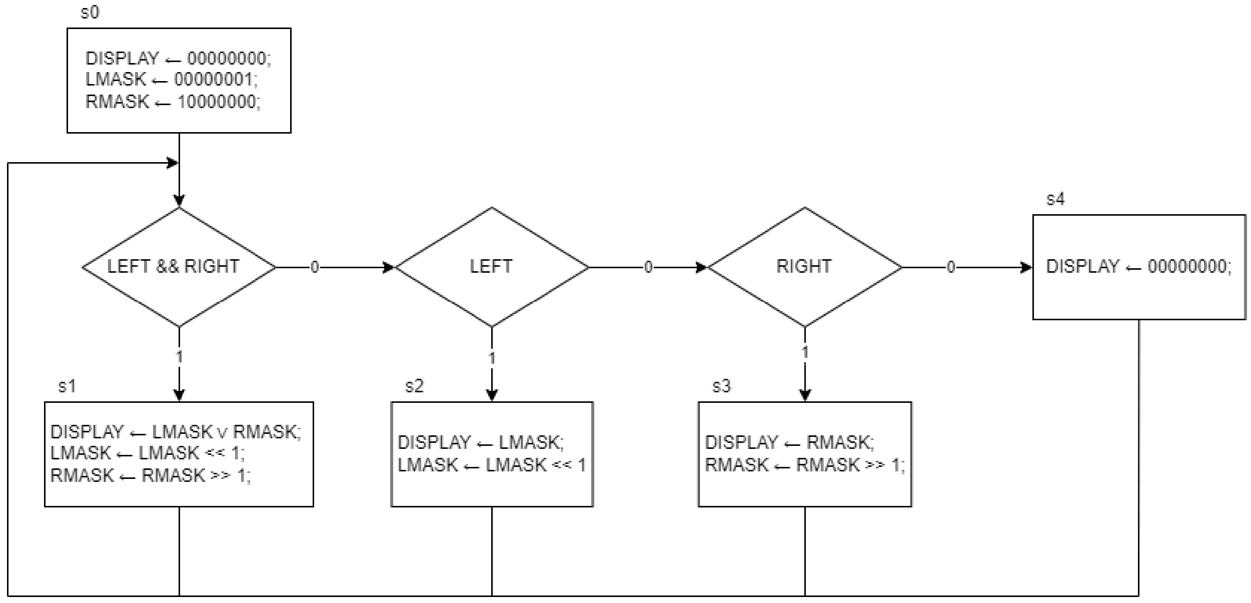
Problem and Solution Discussion:

The light display controller:

The design objective of this lab was to design and demonstrate a light display controller with three buttons, a left button, right button, and a reset button. When the left button is pressed the light would move from right to left across the display, when the right was pressed the light would move from left to right. When both are depressed the movements would happen simultaneously. When nothing is pressed the output would display nothing.

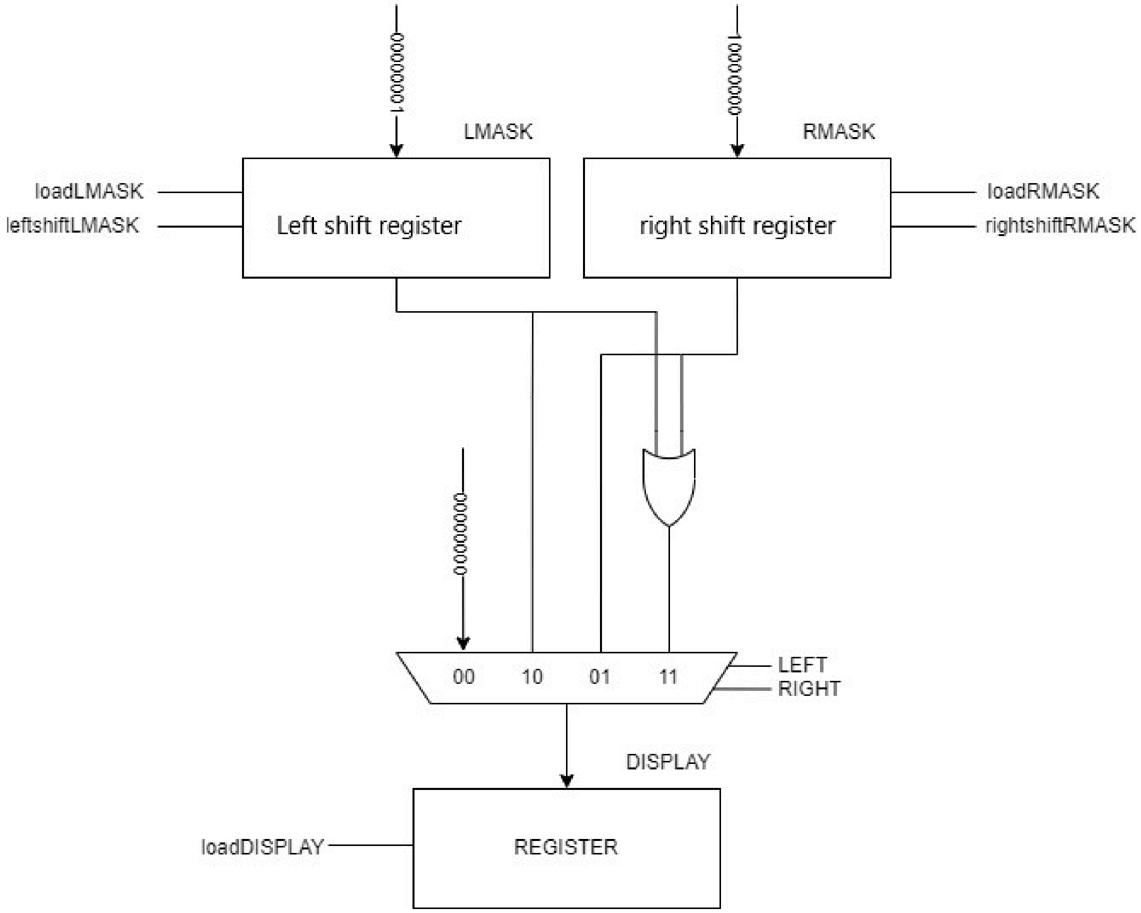
In order to properly design this circuit first the pseudocode had to be analyzed and ASM diagrams created. The pseudocode was given in the lab manual and based on that an ASM chart, datapath, detailed ASM chart and control path could be derived. Below is the first step of the ASM chart, in this chart the pseudocode from the lab manual is transformed into state boxes, conditional boxes and output state boxes. The functionality and steps the circuit takes can be followed through this simple ASM chart.

ASM CHART



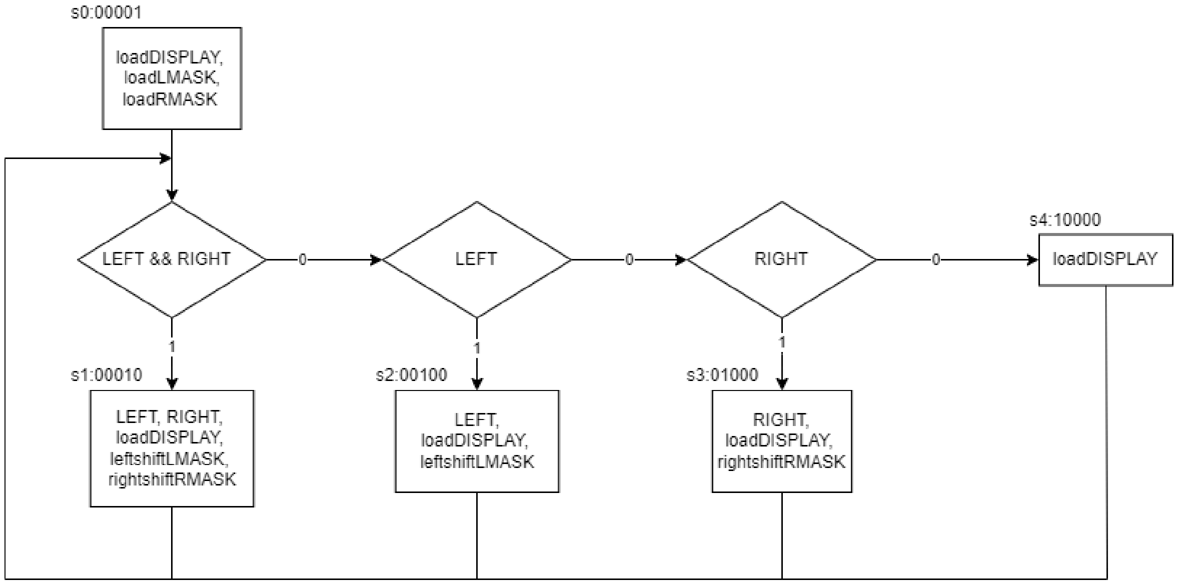
Once the ASM chart was designed and the components required to complete the operations were understood a datapath could be visualized and graphed. Below is the completed datapath and components that would be later coded in structural VHDL.

ASM DATAPATH



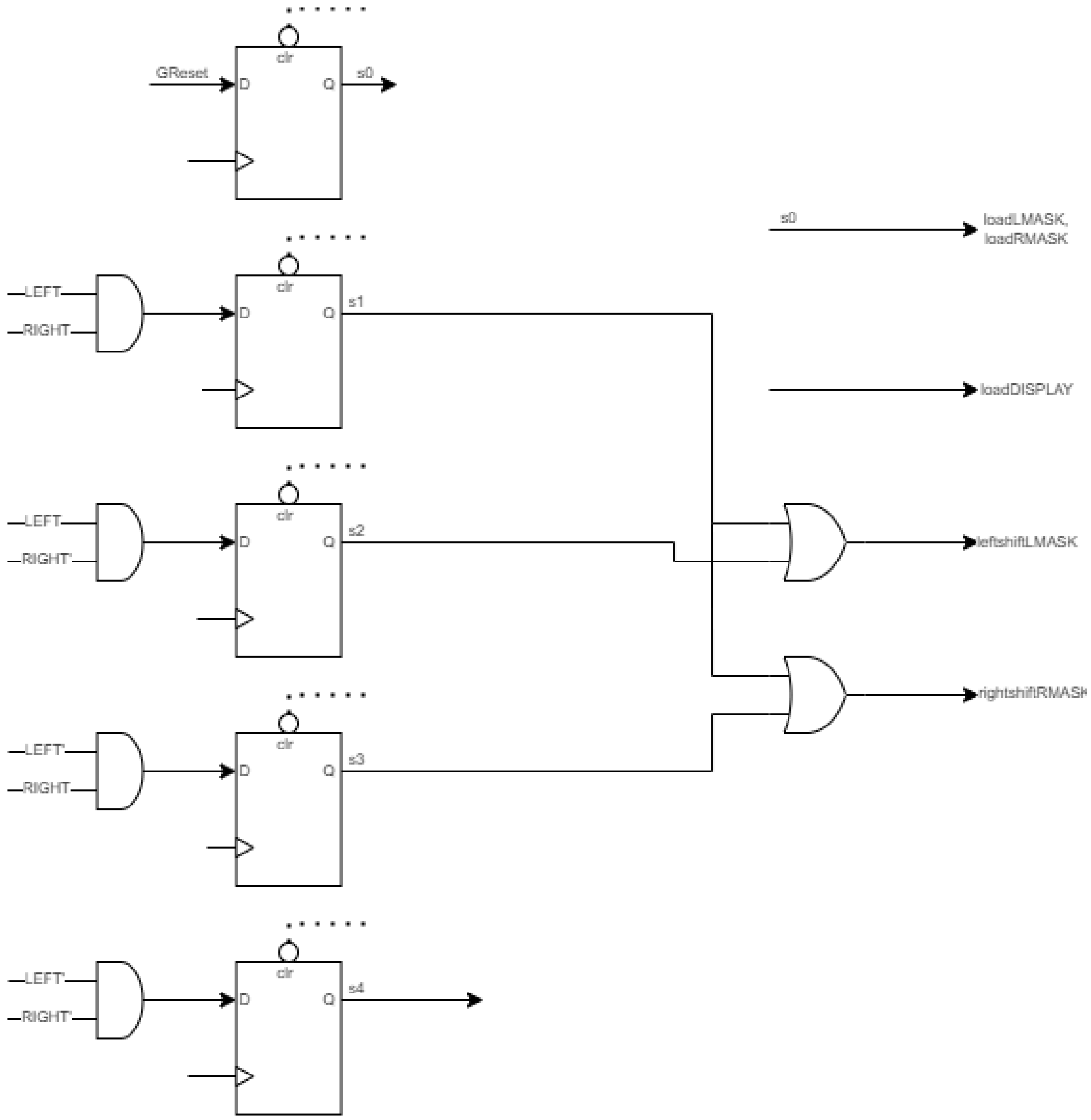
Using this datapath diagram the exact operations and required inputs could be determined in order to create a detailed ASM chart. Below is the detailed ASM chart where the planned states and required inputs have been setup for implementation in VHDL.

DETAILED ASM CHART


s0:00001
loadDISPLAY, loadLMASK, loadRMASK
LEFT && RIGHT
0
LEFT
0
RIGHT
0
s4:10000
loadDISPLAY
1
s1:00010
LEFT, RIGHT, loadDISPLAY, leftshiftLMASK, rightshiftRMASK
1
s2:00100
LEFT, loadDISPLAY, leftshiftLMASK
1
s3:01000
RIGHT, loadDISPLAY, rightshiftRMASK

Finally the control path can be created turning state boxes into flip flops that can only be activated once at a time starting at S0. The control path will control the inputs that are then connected to the datapath creating the final top level entity.

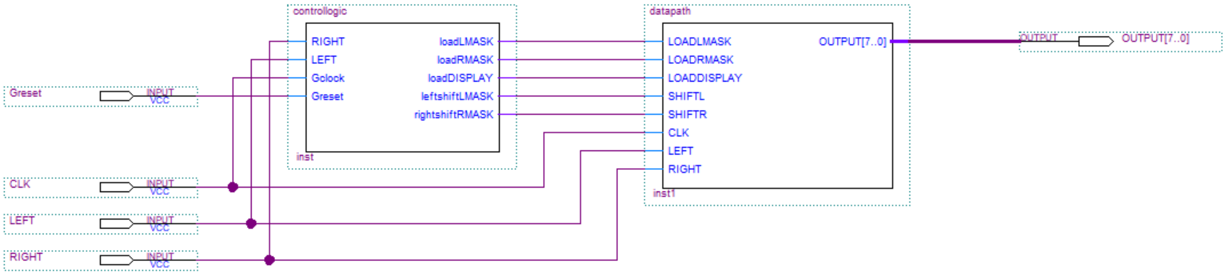
ASM CONTROL LOGIC



Once the control path had been planned the coding of the actual blocks and components inside of the circuit could begin. The larger blocks include:

- Left shift register
- Right shift register
- Output register
- 16-8 bit or gate
- 32-8 bit mux
- Datapath
- Control logic

The last two being the top level entities that connect to finish the circuit and can be connected to input and output pins. The top level entity completed in Quartus can be shown below with the connected datapath and control path. Each containing the components shown in their respective ASM charts.



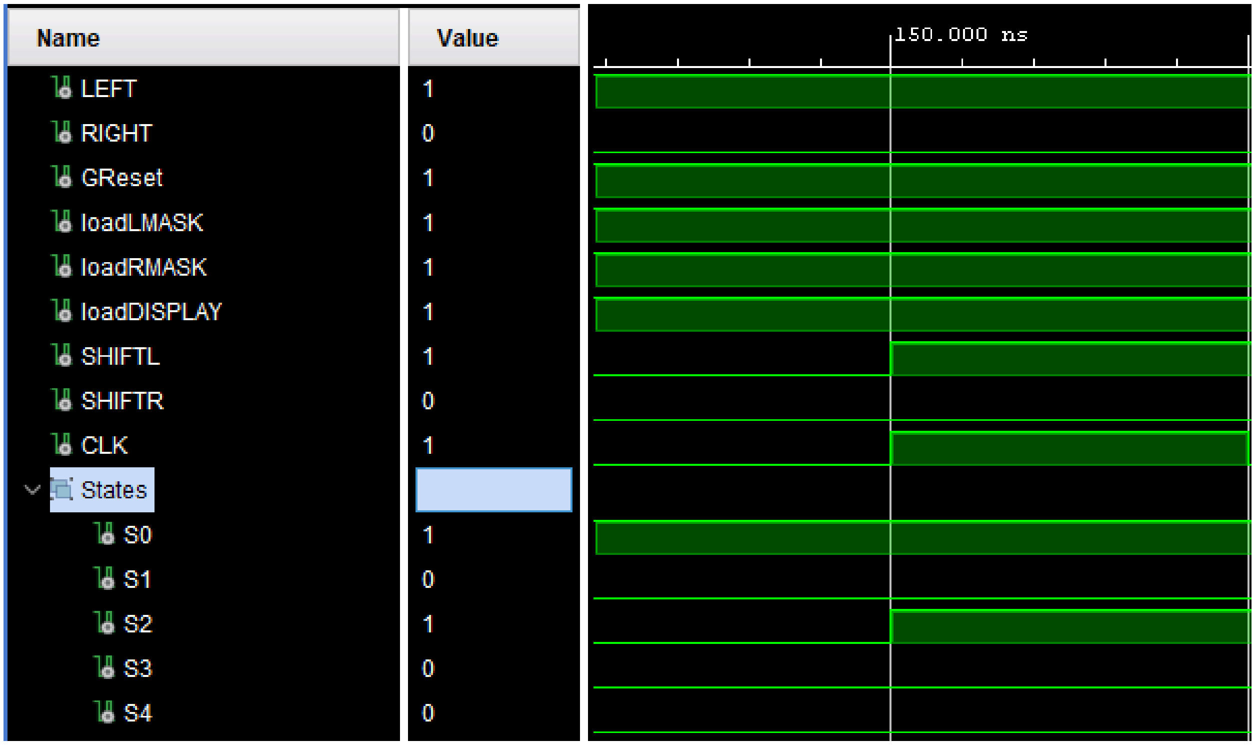
These inputs right, left and Greset could be connected to buttons. The clock connected to a clock pin, and the outputs connected to eight led lights in a row in order to properly display the lights traveling from one end to the other.

Once pins were connected the circuit was demonstrated functioning correctly in line with the predicted simulations shown in the verification section of the report.

Verification:

Functional Simulations of major components:

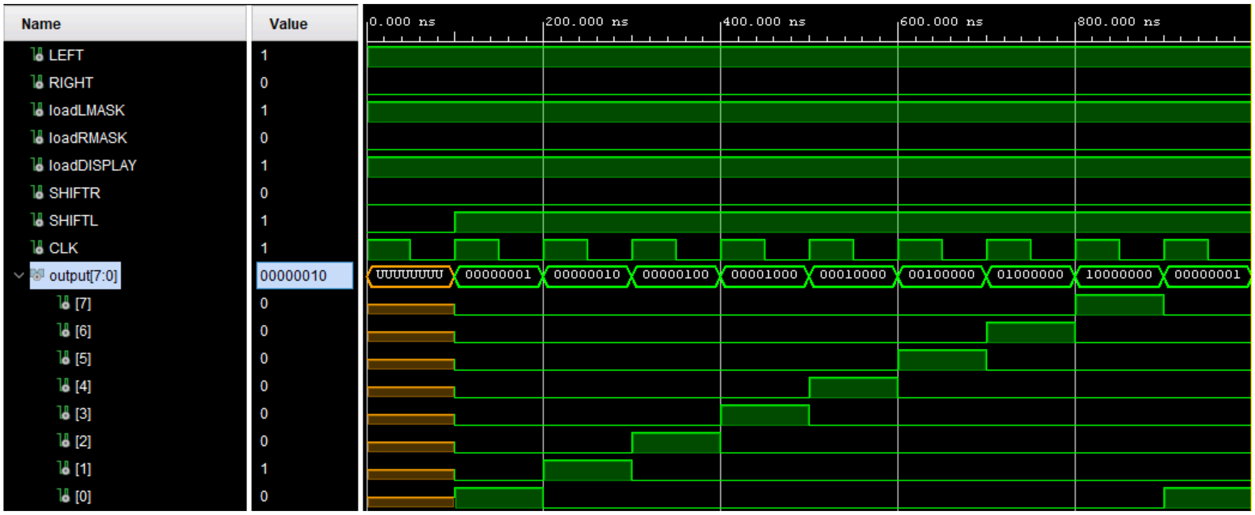
Simulation of Control Logic:



Name	Value
LEFT	1
RIGHT	0
GReset	1
loadLMASK	1
loadRMASK	1
loadDISPLAY	1
SHIFTL	1
SHIFTR	0
CLK	1
States	
S0	1
S1	0
S2	1
S3	0
S4	0

This waveform shows the outputs of the control logic when only LEFT is selected. The outputs correspond to the expected values: both load signals turn on when state ‘s0’ is reached and the correct shift signal turns on when state ‘s2’ is reached.

Simulation of Data Path:

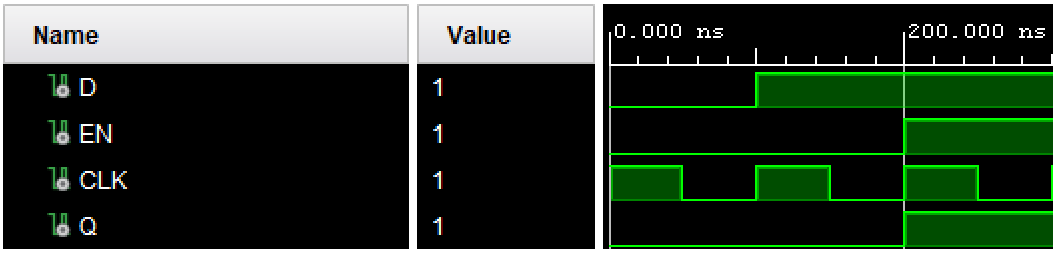


Name	Value
LEFT	1
RIGHT	0
loadLMASK	1
loadRMASK	0
loadDISPLAY	1
SHIFTR	0
SHIFTL	1
CLK	1
output[7:0]	00000010
[7]	0
[6]	0
[5]	0
[4]	0
[3]	0
[2]	0
[1]	1
[0]	0

The signals are as we expect from the datapath component. The waveform above simulates the datapath component's outputs if only LEFT is turned on.

Testbenches Utilized in the Verification Process:

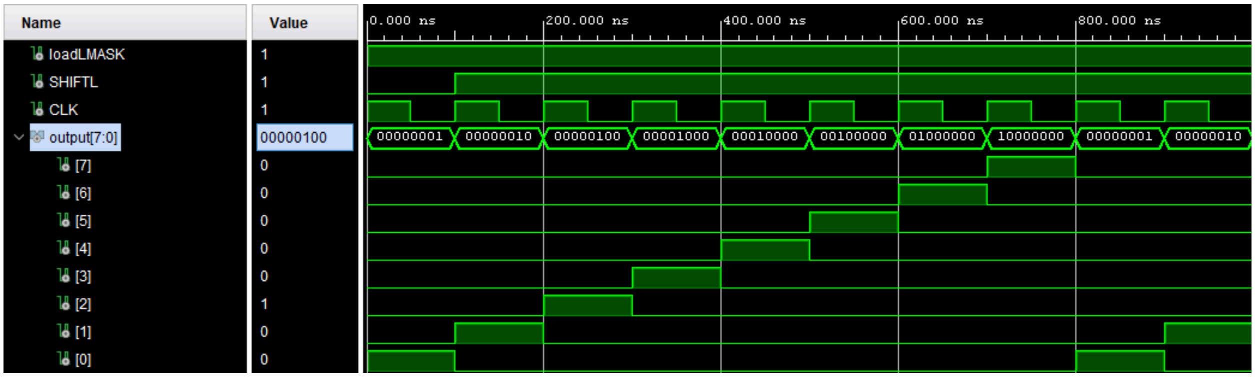
Simulation of D Flip-Flop with Enable:



Name	Value
D	1
EN	1
CLK	1
Q	1

This testbench was used to debug undefined values. The issue was fixed after the VHDL code for the D flip-flops with enable was changed to the behavioral code given on Brightspace (DFF_2.vhd).

Simulation of Left Shift Register:

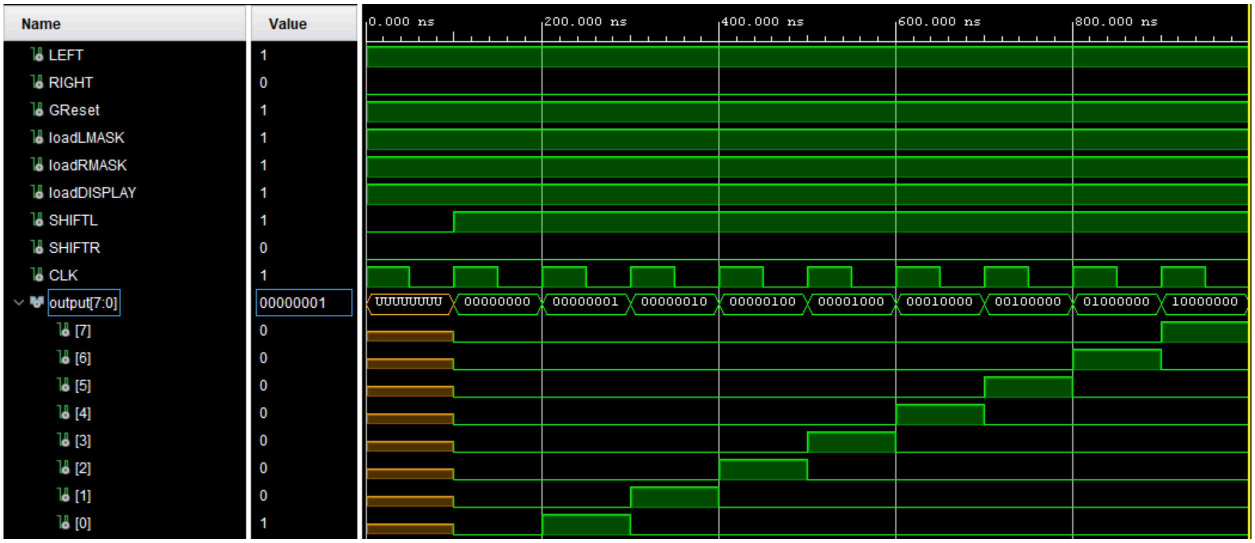


Name	Value
loadLMASK	1
SHIFTL	1
CLK	1
output[7:0]	00000100
[7]	0
[6]	0
[5]	0
[4]	0
[3]	0
[2]	1
[1]	0
[0]	0

A testbench was used to simulate the left shift register and was useful when needing to isolate an issue pertaining to the 2-to-1 multiplexer which is in charge of selecting between a parallel-loaded input or the shifted register value. The selector was found to be undefined ('U') when it needed to be '0'. This was fixed by using D flip-flops with asynchronous reset.

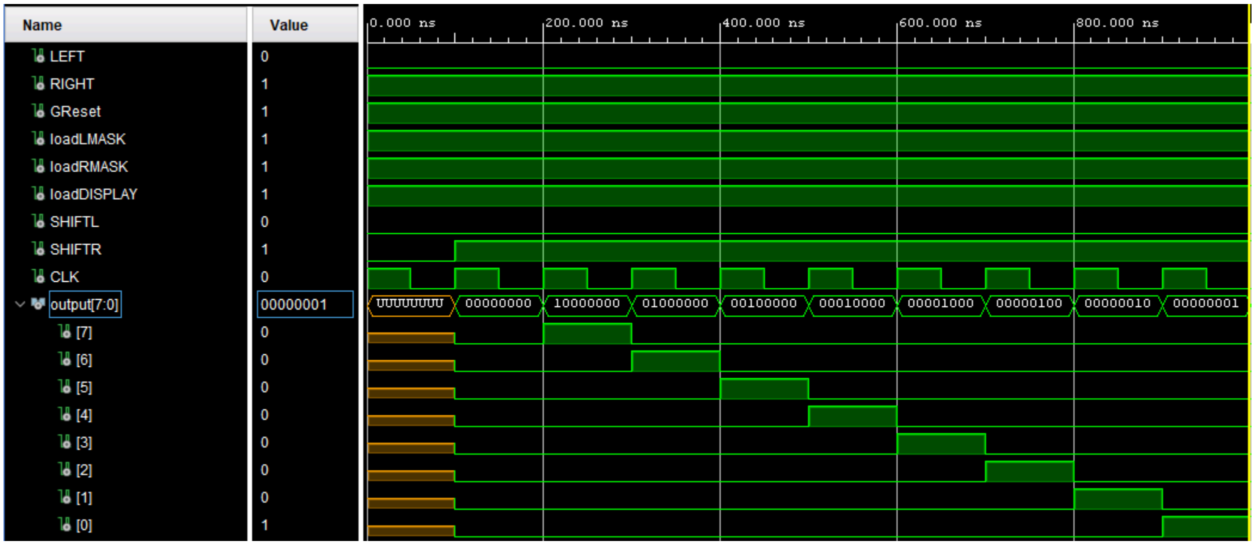
Top-Level Simulation:

With only LEFT:



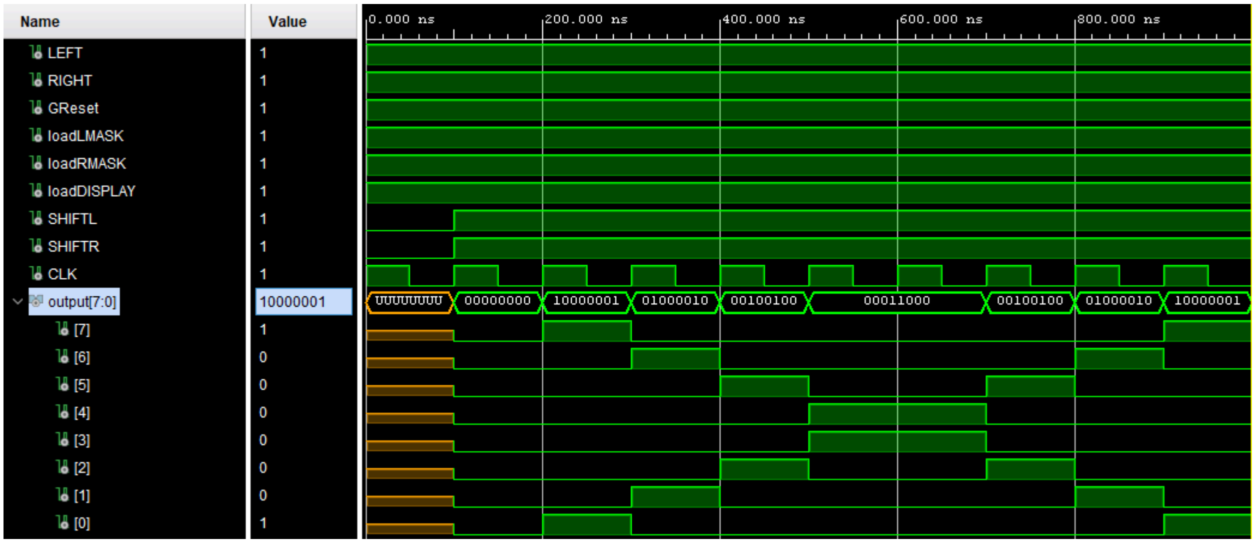
Name	Value
LEFT	1
RIGHT	0
GReset	1
loadLMASK	1
loadRMASK	1
loadDISPLAY	1
SHIFTL	1
SHIFTR	0
CLK	1
output[7:0]	00000001
[7]	0
[6]	0
[5]	0
[4]	0
[3]	0
[2]	0
[1]	0
[0]	1

With only RIGHT:



Name	Value
LEFT	0
RIGHT	1
GReset	1
loadLMASK	1
loadRMASK	1
loadDISPLAY	1
SHIFTL	0
SHIFTR	1
CLK	0
output[7:0]	00000001
[7]	0
[6]	0
[5]	0
[4]	0
[3]	0
[2]	0
[1]	0
[0]	1

With both LEFT and RIGHT:



Lab Discussion:

As shown in the verification process, there were several issues with the circuit that we ran into during development that needed testing and re-working in order to achieve the finished final product.

Firstly, there were some struggles when deciding how to implement the left and right shifting registers properly. Although we could have used two universal shift registers, we decided that designing a simpler shift register would be less trouble to design and to troubleshoot at later stages. Thus, the ‘LMASK’ and ‘RMASK’ registers are only able to parallel load and shift in one direction. We did not see a need for an overcomplicated register for the display, so it is a simple 8 bit register.

Problems related to the use of home designed DFF were run into, which then had to be changed to the professor given behavioral D flip flop designs. These problems plagued both register outputs and outputs from the control logic file. Once changed the design was able to be successfully simulated. Again problems were run into when running on the DE2 altera board as the clock was far too fast for humans to visualize the changes in the bits. In order to change this the clock value was first slowed then eventually swapped to a switch which allowed each action to be observed functioning.

After these setbacks and challenges to the design were overcome, the light display controller was able to be successfully implemented, simulated, and demoed on the board. With the completion of this lab a complete understanding of ASM methodology and structural VHDL coding has been developed for future use in laboratories and other projects.

Conclusion and Appendices:

The expected output from the theoretical part of the lab matches the data we obtained in the simulations and on the ALTERA board. Therefore, we can conclude that we met all objectives for this lab: we designed a solution to the problem using the ASM methodology and we have demonstrated an understanding for circuit realization. We hope to have fewer issues with the VHDL implementation for the next laboratories. All VHDL source files are commented and available here along with the simulation waveforms: Drive folder of VHDL files for lab 1