Department of Electronics and Telecommunications
Bachelor degree in Electronic Engineering

Digital Systems Electronics

LAB 1: Multiplexers, Light Emitting Diodes, Switches and Testbench.

Due date: March 26, 2024
Delivery date: March 23, 2024

Group: 19

Contributions:

- Valtorta Alexander James
- Tedesco Angelo
- Urgu Sara

The members of the group listed above declare under their own responsibility that no part of this document has been copied from other documents and that the associated code is original and has been developed expressly for the assigned project.

Contents

1 Introduction

1.1 Aim of this project

The purpose of this laboratory is to connect input-output devices to the FPGA chip and implement circuits that use these devices for controlling the LEDs, implement a 2-to-1 multiplexer and a 5-to-1 multiplexer.

1.2 Content and structure of the document

The report is divided into three sections, each for every exercise of the laboratory assignment. Moreover, each section contains the key parts in order to design and implement each circuit:
- Design Entry, which describes the circuit to be implemented on the FPGA device, in particular it is indicated the overall architecture of the circuit, by reporting the name of the source file or files and the key role of each of them.
- Functional Simulation, which describes the testbench and the approach used to assess the completed design.
- Synthesis, which describes the synthesis process and results.

2 Controlling the LEDs

2.1 Desing entry

A simple module to connect the switches SW to the LEDs LEDR was provided [Es1.vhd]. After declaring the switches and LEDs as input and output arrays respectively, since in the DE1 are provided 10 toggle switches and 10 red lights, it was possible to describe the circuit by a simple assignment statement in the behavioural styled architecture:

```
LEDR <= SW;
```

2.2 Functional simulation

In order to assess the design the testbench instantiates the component under test (or device under test DUT) in the architecture by referring to the port declaration of the original design. Internal signals (SW_t,LEDR_t) used for the simulation are stated as array in the declarative part of the testbench, as well as the component, defined in Es1.vhd . The descriptive part is composed by the mapping of the internal signals to the component and the proper simulation of the commutation of the switches, obtained by assigning first the value corresponding to an 'open' switch (1111111111) that would coincide with lit LEDs, and then, after a short period of time, the opposite value (0000000000) to simulate shut down LEDs.

2.3 Synthesis

In order to test the functionality of the circuit it is necessary to use Quartus Prime. After the creation of a new project wizard, careful to name it as the top-level entity [Es.1] and select the right type of device or circuit implementation [5CSEMA5F31C6], it is required to insert the VHDL code by means of Quartus Prime Text Editor. Then it is just a matter of compilation of the designed circuit in order to analyze the file, synthesize the circuit, and implement it onto the FPGA. Once the compilation is successful it is possible to implement the circuit into the FPGA, after providing the correct pin assignment using the file DE1_SoC.qsf, and observe the results

3 2-to-1 Multiplexer

3.1 Desing entry

To design the four-bit wide 2-to-1 multiplexer it is necessary to implement first a one-bit wide 2-to-1 multiplexer [mux_2_1_1bit.vhd] by means of a dataflow styled architecture so that, once the inputs x,y,s and output m are declared, only the logical description of the multiplexer is needed:

```
m <= (NOT (s) AND x) OR (s AND y);
```

The four-bit wide 2-to-1 multiplexer [mux_2_1_4bit.vhd] is implemented through a structured architecture that declares as a component the one-bit wide 2-to-1 multiplexer design and the internal signals to be assigned to the components, in order to properly map the signals to the four one-bit wide 2-to-1 multiplexer.

Lastly, the assignment of the inputs and output of the four-bit wide 2-to-1 multiplexer to the corresponding switches and LEDs of the DE1, recognised using the DE1_SoC.qsf file, is needed [mux21_4bit_map.vhd]. To do so it is necessary to declare as component the four-bit wide 2-to-1 multiplexer previously designed and in the entity those assigned to the DE1. Then it is only a matter of mapping the signals to the inputs and output of the multiplexer.

3.2 Functional simulation

In order to assess the completed design two testbenches are needed: one for the one-bit wide 2-to-1 multiplexer and one for the four-bit wide 2-to-1 multiplexer.

For the former [tb_mux21_1bit.vhd] after the typical "empty" entity the declarative part of the behavioural architecture consists of the component (defined in mux_2_1_1bit.vhd) and internal signals for the simulation (X_t,Y_t,M_t,S_t). In the descriptive part, the mapping of the internal signals to the inputs and output of the multiplexer (A,B,SEL,K) takes place, as well as the assignment of casual values for the input signals, so that it is possible to simulate the behaviour of the switch by allocating the values corresponding to open (1) and closed (0) switch interspaced with a short period of time.

The structure of the testbench for the four-bit wide 2-to-1 multiplexer [tb_mux21_4bit.vhd] is equivalent to the previous one, just with the corresponding component (defined in mux_2_1_4bit.vhd) and the appropriate internal signals (A,B,K,SET) so that they are properly associated with the inputs and output of the component (x,y,s,m). By defining the MUX 2 to 1, binary version, as a component e were then able to use PORT MAP to map the various signals to the port we wanted. Our implementation didn't use vectors, but it would have also been possible to use them by then assigning each mux to it's corresponding vector value of the various signals, as an example:
X1: MUX2to1 PORT MAP(X(0),Y(0),S(0),pout(0));
X2: MUX2to1 PORT MAP(X(1),Y(1),S(1),pout(1));
X3: MUX2to1 PORT MAP(X(2),Y(2),S(2),pout(2));
X4: MUX2to1 PORT MAP(x(3),Y(3),S(3),pout(3));

3.3 Synthesis

The steps taken for the simulation through ModelSim of this code were similar if not identical to the last exercise: In order to test the functionality of the circuit it is necessary to use Quartus Prime. After the creation of a new project wizard, careful to name it as the top-level entity [Es.1] and select

the right type of device or circuit implementation [5CSEMA5F31C6], it is required to insert the VHDL code by means of Quartus Prime Text Editor. Then it is just a matter of compilation of the designed circuit in order to analyze the file, synthesize the circuit, and implement it onto the FPGA. Once the compilation is successful it is possible to implement the circuit into the FPGA, after providing the correct pin assignment using the file DE1_SoC.qsf, and observe the results

4 5-to-1 Multiplexer

4.1 Desing entry

A three-bit wide 5-to-1 multiplexer can be implemented by using three one-bit wide 5-to-1 multiplexers each of which can be obtained by using four one-bit wide 2-to-1 multiplexers, so it is necessary to use a structural approach.

In order to do so, the one-bit wide 2-to-1 multiplexer [mux_2_1_1bit.vhd], already designed in the previous exercise, is used as a component in the declarative part of the architecture for the one-bit wide 5-to-1 multiplexer [mux_5_1_1bit.vhd], alongside the declaration of internal signals for said component. Then it is only required the suitable mapping of the internal signals and the inputs and outputs of the one-bit wide 5-to-1 multiplexer to the components of each of the four one-bit wide 2-to-1 multiplexer.

The implementation of the three-bit wide 5-to-1 multiplexer [mux_5_1_3bit.vhd] requires the same approach but considering the right elements. In particular, the component declared was that of the one-bit wide 5-to-1 multiplexer, so that the inputs and output of the three-bit wide 5-to-1 multiplexer are associated to the components of the three one-bit wide 5-to-1 multiplexer.

Lastly, the assignment of the inputs and output of the three-bit wide 5-to-1 multiplexer to the corresponding switches and LEDs of the DE1, recognised using the DE1_SoC.qsf file, is needed [mux51_3bit_map.vhd]. In order to do so it is necessary to declare the three-bit wide 5-to-1 multiplexer as a component and the switches and LEDs signals assigned to the DE1 in the entity. Finally, following the provided assignment, it is only a matter of associating the proper constant values, switches and LEDs to the right components of the three-bit wide 5-to-1 multiplexer.

4.2 Functional simulation

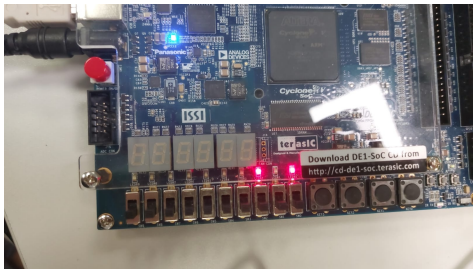
In order to assess the completed design two testbenches are required: one for the one-bit wide 5-to-1 multiplexer and one for the three-bit wide 5-to-1 multiplexer.

For the one for the one-bit wide 5-to-1 multiplexer [tb_mux51_1bit.vhd] after the typical empty entity the declarative part of the behavioural architecture is composed by the component, defined in mux_5_1_1bit.vhd, and internal signals for the simulation (U_t,V_t,W_t,A_t,B_t,J_t,SEL_t). The descriptive part holds the mapping of said internal signals to the component, followed by an assignment of casual values for the inputs signals in order to then simulate the commutation of the switch by assigning all possible values waiting a short amount of time between each assignment.

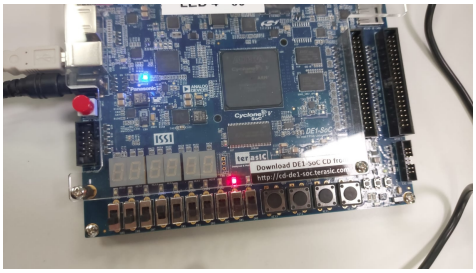
The same approach was followed for the testbench of the three-bit wide 5-to-1 multiplexer [tb_mux51_3bit.vhd], mindful of the use of the appropriate component, defined in mux_5_1_3bit.vhd, in order to map the internal signals declared for the simulation (U3_t,V3_t,W3_t,A3_t,B3_t,J3_t,SEL3_T) to the component. The assignment of casual values and the simulation of the commutation happens in the same manner.

4.3 Synthesis

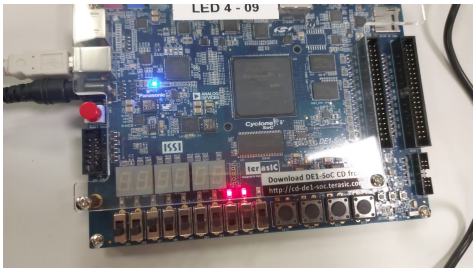
Once the compilation executed in Quartus Prime is successful, and the right pin assigned is performed, the results cam be observed by toggling the switches and observing the LEDs:



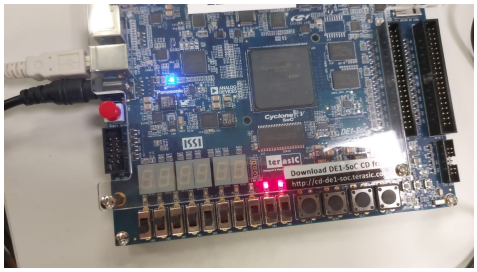
(a) Input U



(b) Input V



(c) Input X



(d) Input Y

Figure 1