Basic Combination Logic and the DE10-Lite

Christopher Hunt

04/14/2023

Objectives

The objective of this lab is to provide students with a comprehensive learning experience in digital logic design using programmable logic devices. Throughout the lab, participants will acquire skills in using Quartus Prime for drafting digital logic designs, gain experience in programming the Max10 FPGA on the DE10-Lite development board, understand the process of verifying combinational logic designs to ensure accurate implementation, and learn how to simulate digital logic designs to test their functionality and performance.

Equipment

- Quartus Prime Lite Edition V. 20.1.1
- DE10-Lite kit with MAX10 10M50DAF484C7G FPGA
- USB to USB-B cable

Design

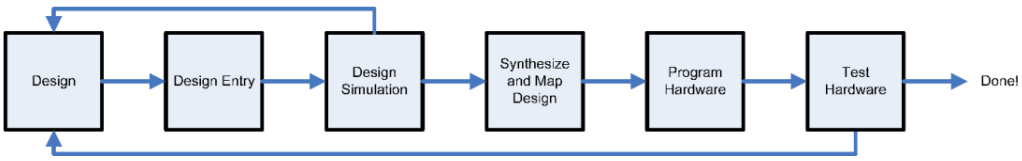

Figure 1: This six-step process is used for good digital logic design.

The process of designing a digital logic system can be summarized into a 6 step design process (fig. 1). The design of this project will utilize three input switches and a single output LED on the DE10-Lite (SW0, SW1, SW2, and LEDR0) (fig. 2).

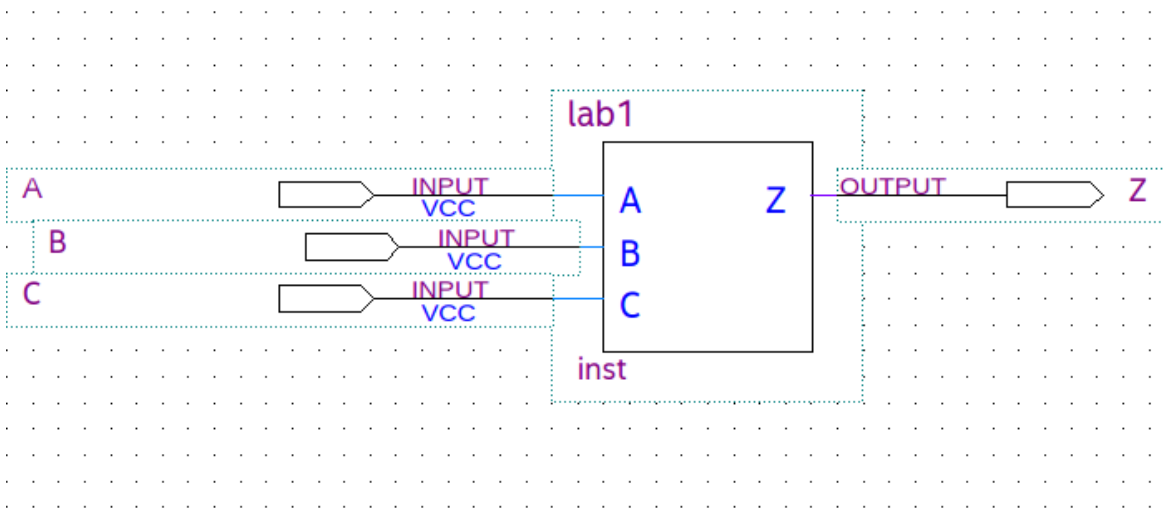

Figure 2: Lab 1 Block Diagram

Pin placement labels were found using the DE10-Lite manual. SW2 is mapped to PIN_D12, SW1 is mapped to PIN_C11, SW0 is mapped to PIN_C10, and LEDR0 is mapped to PIN_A8 (Table 1). To illustrate the basic functionality of Quartus Prime these three inputs will be used in conjunction with three logic gates, an XOR gate, an OR gate, and an AND gate (fig. 4).

Table 1: FPGA PIN

Input	**PIN**
A	PIN_D12
B	PIN_C11
C	PIN_C10
Output	**PIN**
Z	PIN_A8

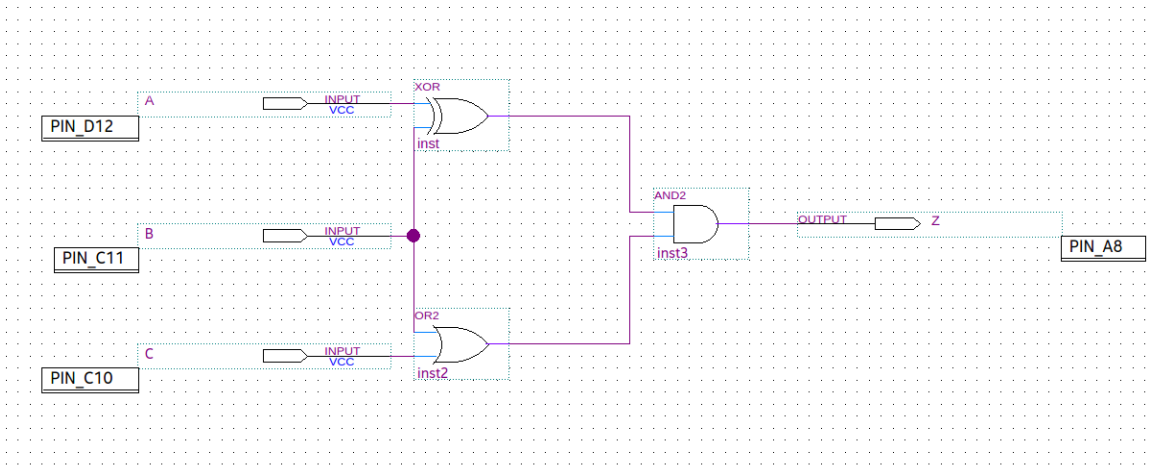

Figure 3: Quartus Prime Schematic

Simulation

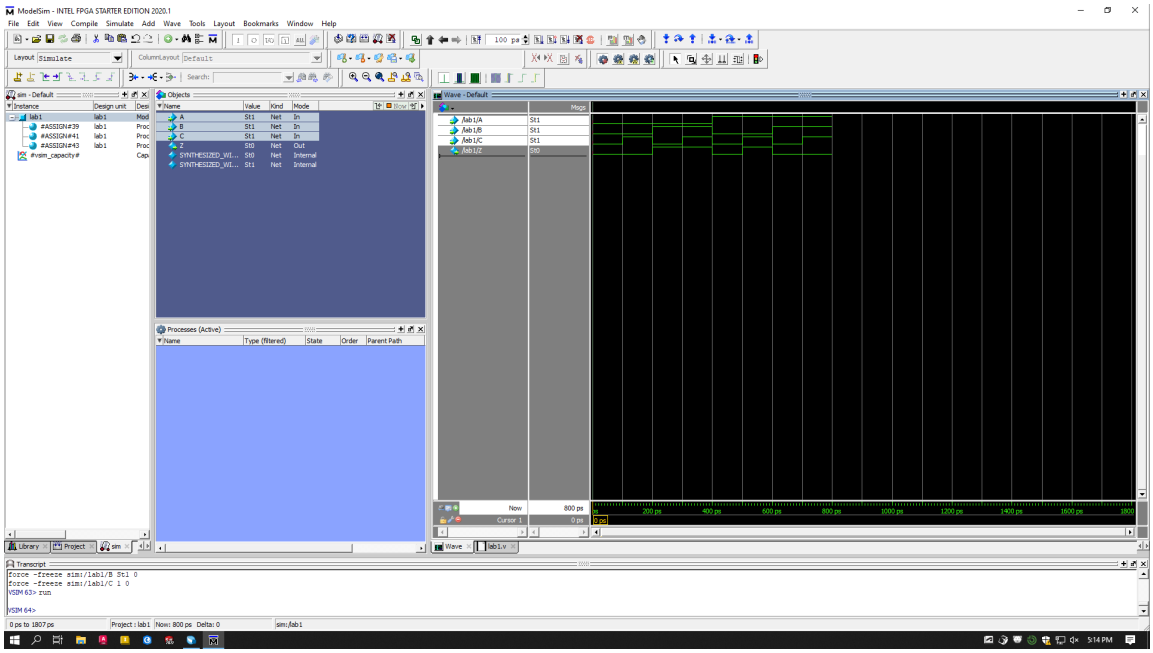

Figure 4: ModelSim Simulation

The simulation of the FPGA design was performed using ModelSim. The design has three input pins and a single output pin. The simulation was run and produced the expected results. The simulation results are shown in Table 2.

Table 2: Logic Gate Truth Table

A	B	C	Z expected	Z actual
0	0	0	0	0
0	0	1	0	0
0	1	0	1	1
0	1	1	1	1
1	0	0	0	0
1	0	1	1	1
1	1	0	0	0
1	1	1	0	0

Implementation

The HDL code for the FPGA design was successfully loaded onto the DE10-Lite FPGA board. The three input pins were correctly routed to the appropriate switches on the board, and the output was observed to match the expected results from the appropriate LED (Table 2). Figure 5 shows a picture of the DE10-Lite FPGA board running the program.

Figure 5: Configured DE10-Lite

Observations

Adhering to proper design steps is crucial for a smooth and efficient logic design process. It is essential for designers to have a clear understanding of the specific device model they are working with and to refer to the model's manual for accurate pin locations and other vital information. In addition, ensuring the correct installation of the software is a critical aspect of the design process, as it lays the foundation for seamless development and implementation of digital logic designs. By paying attention to these fundamental aspects, designers can significantly enhance the quality and accuracy of their work, ultimately leading to successful digital logic projects.

Conclusion

In this lab, we successfully designed and implemented a digital logic system (fig. 3) using Quartus Prime Lite software and a DE10-Lite MAX10 10M50DAF484C7G FPGA. The lab focused on drafting digital logic designs, programming the FPGA, verifying combinational logic designs, and simulating digital logic designs. By following a structured approach and paying attention to essential details, we gained hands-on experience in foundational aspects of digital logic design. This practical knowledge, combined with a deeper understanding of the advantages of programmable logic devices, will serve as a strong foundation for future projects.

Study Questions

1. Describe any problems encountered in this lab and your solutions to those problems.

The main stumbling block we had was getting the proper pin layout assigned to the board. It was crucial that we selected the appropriate hardware device we are working with in order to have the pins we want to line up with what we are designing. At home I am still encountering issues with getting the software installed and working properly. I currently am having issues getting the device drivers to work.

2. Give an example of where discrete logic ICs (such as the 7400 series logic chips mentioned in section 1.1) are used in industry and why.

Discrete logic ICs, such as the 7400 series logic chips, still find applications in certain areas of the industry due to their simplicity, cost-effectiveness, and low power consumption. One example of their use is in designing basic control systems for small-scale automation tasks or simple monitoring systems in manufacturing facilities. The design requirements in these scenarios may not be complex enough to necessitate the use of programmable logic devices. Instead, using basic logic building blocks like these discrete logic chips get the job done in a more cost-effective and efficient manner.

3. Give an example of when you should use an FPGA instead of a PLA and explain why.

An FPGA should be used instead of a PLA when the design requirements needs a high level of flexibility, complexity, or a large number of logic elements. One example of when an FPGA would be a better choice than PLA is in the implementation of digital signal processing systems. A DSP system may involve a large number of arithmetic and logical operations, an FPGA can provide the appropriate framework to implement such designs.

4. Give an example of when you should use a PLA instead of an FPGA and explain why.

A PLA should be used instead of an FPGA when the design requirements are relatively simple, fixed, and not expected to change frequently. One example of such a scenario is implementing basic combinational logic functions or control systems with a limited number of inputs and outputs in applications like alarm systems, control panels, or simple controllers for small appliances. In these applications the designs simplicity and limited logic requirements make using a PLA more cost effective option.

5. Summarize the main differences between FPGAs and CPLDs, other than the difference described in section 1.1.

FPGAs and CPLDs differ in several key aspects: FPGAs have a more flexible architecture and higher logic density, making them suitable for complex designs, while CPLDs are ideal for simpler designs with lower power consumption and faster startup times. FPGAs offer fine-grained programmability, while CPLDs provide a more coarse-grained approach.

Appendix

```
// Copyright (C) 2020  Intel Corporation. All rights reserved.
// Your use of Intel Corporation's design tools, logic functions
// and other software and tools, and any partner logic
// functions, and any output files from any of the foregoing
// (including device programming or simulation files), and any
// associated documentation or information are expressly subject
// to the terms and conditions of the Intel Program License
// Subscription Agreement, the Intel Quartus Prime License Agreement,
// the Intel FPGA IP License Agreement, or other applicable license
// agreement, including, without limitation, that your use is for
// the sole purpose of programming logic devices manufactured by
// Intel and sold by Intel or its authorized distributors.  Please
// refer to the applicable agreement for further details, at
// https://fpgasoftware.intel.com/eula.

// PROGRAM              "Quartus Prime"
// VERSION              "Version 20.1.1 Build 720 11/11/2020 SJ Lite Edition
    "
// CREATED              "Fri Apr 14 17:01:30 2023"

module lab1(
        A,
        B,
        C,
        Z
);


input wire      A;
input wire      B;
input wire      C;
output wire     Z;

wire    SYNTHESIZED_WIRE_0;
wire    SYNTHESIZED_WIRE_1;




assign  SYNTHESIZED_WIRE_0 = A ^ B;

assign  SYNTHESIZED_WIRE_1 = C | B;

assign  Z = SYNTHESIZED_WIRE_0 & SYNTHESIZED_WIRE_1;


endmodule
```