Chapter 1

Basic Combinational Logic and the MachXO2
CHAPTER 1. BASIC COMBINATIONAL LOGIC AND THE MACHXO2

1.1 Section Overview

The 7400-series logic chips were the most common digital logic devices around for many years. Most digital logic implementations have moved either towards Programmable Logic Devices (PLDs) or towards Field Programmable Gate Arrays (FPGAs).

The MachXO2 used in the TekBots program is an FPGA made by Lattice Semiconductor. There is more information about this board in Appendix A. This section will introduce you to Lattice’s FPGA development suite, Diamond, by walking you through a simple schematic capture design.

1.2 Objectives

In this section, the following items will be covered:

1. Using the Lattice Diamond software to draft digital logic designs
2. Programming the MachXO2 provided in the ECE 272 kit
3. Verifying combinational logic designs

1.3 Materials

1. Lattice Diamond 2.2 software (Currently installed on the lab computers)
2. MachXO2 breakout board
3. Button Board (8pushbtn.0)
4. USB to mini-USB cable

1.4 Procedure

There are 6 steps to digital logic design:

![Figure 1.1: This six step process is used for designing in ECE 272.](image)

1. **Design**: The context of the design is established in the "Design" step. The context involves defining the desired output, input, and all of the logic required to connect the two. In this step, all of the minimizations and layout is planned for the entry process. While this step is not always the most lengthy, it involves the most thought and effort.

2. **Design Entry**: The actual drafting of the digital logic design occurs in this step. Drafting includes entering the logic gates and blocks to build the design in the software tool.
3. **Design Simulation**: Before committing to hardware, this step tests the design in a controlled computer simulation. If the design does not function as specified in the “Design” step, it is revised.

4. **Synthesize and Map Design**: When the design simulates correctly, the source files are synthesized into a design file that configures the FPGA. Mapping the design to pins assigning the inputs and outputs of the design to IO pins on the MachXO2 FPGA.

5. **Program Hardware**: After the design file is created it is used to configure (program) the MachX02 FPGA. Programming uses the USB to mini-USB cable and the Lattice Diamond software to send a bit stream to configure the FPGA.

6. **Test Hardware**: Verify hardware operation once the FPGA has been programmed. The FPGA should operate exactly as the simulation predicted, but timing problems or incorrect assumptions about the hardware can require the designer to return to the “Design” step.

### 1.5 Design

A good design is the key to any successful project. In electrical engineering, many of our designs stem from a 'Block Diagram' developed near the start of a project and evolved throughout the course of it. Block diagrams can encompass all levels of detail and abstraction in a project, so choosing the scale of the initial diagram can be a challenge. A good starting point is to represent each piece of hardware with a block and define all interfaces with it. Figure 1.2 shows the start of a simple block diagram for this lab.

![Incomplete block diagram for Section 1](image)

At the beginning of a project, the block diagram should include:

- Labelled blocks representing each piece of hardware used in the project
- All interfaces (connections) between blocks, represented with lines
- Which signals are carried on each interface
- How the interfaces connect to each block, for example: pin numbers

Depending on the project, more detail is added to the block diagram, often with very detailed blocks inside of the main hardware blocks.

Figure 1.3 is the design that will be drafted in Diamond.
There are a few design choices that are dictated by the hardware.

1. The buttons on the button board are active low. This means that without a pullup resistor, the button's output will float, producing a high-impedance signal (Z) when they are unpressed and output logic low (0) when they are pressed. With a pullup resistor on the output, the output will be pulled up to a logic high when unpressed and logic low when pressed. Refer to Figure 1.4 to get a partial schematic of the button board. The full schematic can be found in the appendices.

2. The LEDs on the MachXO2 are active low. This means they turn on when a logic low signal is sent to the FPGA pin and they turn off when a logic high signal is sent.

### 1.6 Design Entry

#### 1.6.1 Start Diamond

Click **Start** → **All Programs** → **Lattice Diamond 2.2 (64 bit)** → **Lattice Diamond**
Installation instructions for Diamond can be found on the ECE272 page on TekBots.com

1.6.2 Create a New Project

A project in Lattice Diamond is a collection of sources (Schematic or HDL), testbenches, and simulation outputs. Follow these steps to create the project for this section:

1. Click File → New → Project
2. Click Next to advance
3. In the Project Name field, enter “section1”. Note that the project name automatically creates a subdirectory in the main directory path. However, the main directory must be specified.
4. In the Project Location field, browse to the directory that will store all projects and create a new folder for Section1. Diamond does not automatically create a folder for each project. Click Next.
   
   Using the C:\drive to store projects will make the software operate quicker than using the Z:\drive. Using the Z:\ allows access to project files from any lab machine and is regularly backed up. The files in C:\ are stored locally and can only be accessed from one computer. Keep a back up on Z:\ to avoid frustrating crashes that will cause all project files to be corrupted.

5. In the Add Source window, leave everything blank and click Next.
6. Set the Device Settings to match Figure 1.5 and then click Next.

Figure 1.5: Copy these values for all projects
7. Select *Lattice LSE* as the Synthesis Tool and click *Next*.

8. Review the Project Information and click *Finish*.

9. The project’s *Input Files* folder is empty at this point. To add a source, Click *File → New → File*

10. In the *New File* window, under *Source Files*, select *Schematic Files*.

11. In the *File name* field, enter “section1_schematic”

12. Make sure that the *Add to Implementation* box is checked and then click *New*.

13. The Diamond project screen will now be showing the blank schematic file.

### 1.6.3 Schematic Entry

Follow these steps to create the schematic shown in Figure 1.3:

**Add Symbols:**

Follow these steps to add logic gates to the schematic:

1. Click the *Symbol* button.
2. Select the *lattice.lib* library.
3. Select *and2*.
4. Place one AND gate onto the schematic.
5. Add both of the other gates from Figure 1.3 onto the schematic.

**Connect the Symbols:**

Use Figures 1.7 and 1.8 to understand how to properly draft a schematic. Proper design has inputs on the left stretching vertically and logic gates connected to the correct input lines. Use this general structure to create a schematic that reveals the structure, intent, and flow of the design.
1. Click the Wire button.

2. Wire all of the symbols together. Left click once for each segment, right click to stop placing wire.

3. Add hanging wires off of the inputs and output so that they can be named and designated as inputs or outputs in the next step.

The marker for hanging wires and nodes are the same: a small square. Always double check these to make sure that you have made the connection that you intended.

Define Inputs and Outputs

Labeling the inputs and outputs on the schematic allows Diamond to create a netlist during the synthesis process. A netlist contains all inputs, outputs, and gates for the design.

1. Click the Net Name button and rename the hanging wires A, B, C, and Z. The labels must be placed on the very end of the hanging wires or Diamond will not correctly place IO markers in the next step.

2. Click the IO Port button and specify the type as Input, and add a marker to A, B, and C.

3. Add an Output marker to Z.

4. Save the schematic (Ctrl-S).

1.7 Design Simulation

This design is trivial and the simulation of it will not fit in this lab. Simulation of digital logic designs will be in a future section of ECE 272.
1.8 Assign Pins

1.8.1 Synthesis

During synthesis, the design is minimized and transformed into a netlist describing the hardware. Minimizations reduce the amount of hardware and space required for the design. After synthesizing, Diamond knows how many inputs and outputs are required for the design. Follow this process to synthesize the design.

1. Open the Process tab to the left of the schematic and double click Synthesize Design.

1.8.2 Spreadsheet View

Spreadsheet view is a tool that will connect A, B, C, and Z from the schematic to Inputs/Output pins on the MachXO2. Use Figure 1.10 to organize the information required for Spreadsheet view. The PULLMODE setting determines if the port reads high (1) or low (0) when the port is undriven. PULLMODE UP assigns a pullup resistor to pull the pad voltage up to VCC. This results in an input port reading logic high when it is undriven. The FPGA pins are printed on the silkscreen of the MachXO2 board.
Without a pullup resistor an input pin will float between logic high and logic low when the button is unpressed. Floating inputs cause erratic behavior.

**Run Spreadsheet View**

Follow these steps to run Spreadsheet view and configure the IO pins.

1. Click **Tools → Spreadsheet View**

2. In Spreadsheet View, select **Port Assignments** along the bottom edge. This page contains all the inputs and outputs from your schematic.

3. Assign the IO pins that you chose for each input and output in the **Pin** column.

   Outputs in Spreadsheet View should be assigned to the active low LEDs on the MachXO2 so the output can be quickly verified. Figure 1.12 contains the FPGA pins for the LEDs.

   Inputs in Spreadsheet View should be assigned to empty IO pins on the MachXO2 and later connected to the active low buttons on the button board.
4. Select whether the port will use a pullup or pulldown resistor in the PULLMODE column.

5. Save the pin assignments (Ctrl-S).

LEDs are an extremely useful tool for circuit verification, since they visually show the designer what is happening during any given moment. Carefully picking which signals get assigned to the LEDs can make troubleshooting much easier and save you a lot of time later.

1.8.3 Creating the Programming File

The programmer needs a .jed (JEDEC programming specification) file in order to program the MachXO2. This step uses Diamond to generate a .jed programming file.

1. Open the Process tab to the left of the schematic and double click JEDEC File under Export Files.

1.9 Program Hardware

1.9.1 Program the FPGA

Use this process to program the .jed file onto the FPGA.

Figure 1.12: MachXO2 LED pinout

Figure 1.13: Programmer
1. Plug the MachXO2 into the computer with the provided USB cable.

   Communication between the Breakout Board and a PC via the USB connection cable requires installation of the FTDI chip USB hardware drivers. Loading these drivers enables the computer to recognize and program the Breakout Board. Refer to the Diamond Installation instructions for FTDI driver installation instructions.

2. Click **Tools → Programmer**

3. Click **OK** to identify the correct cable.

   If Diamond cannot identify the target device, manually select "LCMXO2-7000HE" from the **Device** dropdown menu.

4. In the **File Name** column, browse to the .jed file that Diamond created in the project directory.

5. Click **Program**

### 1.9.2 Assemble the Push Button board

The inputs A, B, and C will be provided by the push button board. J1-J8 correspond to each of the buttons in order. For the applications in this lab, if SW_COM is tied to ground, the outputs J1-J8 can be connected to the FPGA and the buttons will operate as active low buttons. To assemble the push button board, follow these steps. All required parts should be in the kit.

1. Solder the buttons onto the board, making sure the buttons are installed in the proper direction.

2. Solder the diodes onto the board, making sure the diodes are installed in the proper direction.

3. Solder male header onto the board in J1-J8, GND, VCC, SW_COM, COM_LVL, and COM_EN.

   Soldering the header into either side of the PCB will function identically, however, it will be easier to use the buttons if they are placed on the opposite side as the buttons.

### 1.10 Test Hardware

Create and fill in a truth table that shows your hardware operates correctly. This truth table has 3 inputs and 1 output. Look at Figure 1.14. Assume that A and B are active low buttons and Z is connected to an active low LED. The filled in row indicates that when A is not pushed, B is pushed, then the output Z turns on the LED. This truth table should match your simulation results.
1.11 Study Questions

1. Turn in a printed copy of the schematic designed in this lab.

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

3. Research discrete logic ICs, such as the 7400 series logic chips. Discuss two advantages and two disadvantages to using an FPGA versus discrete logic ICs.

1.12 Challenge - Extra Credit

Research a ring oscillator. This circuit is a great way to explore the propagation delay of a simple logic gate (the inverter). Credit for the challenge is allocated based on lessons learned from experiments that you design yourself. Here are some beginning ideas:

1. Try to implement a ring oscillator with an integrated circuit of 7400 series logic chips. These ICs will need to be purchased either online, from TekBots in KEC 1110, from the IEEE store in the Dearborn basement, or from a local electronic parts store. How fast does it oscillate? How does the propagation time relate to the VCC powering the integrated circuit?

2. Try to implement the ring oscillator with the FPGA. How fast does it oscillate? Can the FPGA be reconfigured to operate faster or slower?

3. Try to implement the ring oscillator with hand made inverters (either CMOS or RTL). How fast does it oscillate? Does it vary with temperature?