Editors: Bradley Wang, Jason Lowe-Power

ECS 154B Lab 1, Winter 2018

Due by 8:59 AM on January 22, 2018

Via Canvas

Goals

- Learn how to use Logisim.
- Learn how to use the circuit analyzer tool.
- Learn an alternative to combinational logic.

Logisim

- The project page for Logisim is on Dr. Carl Burch's website.
- You can download Logisim on SourceForge for Windows and OS X. If you are using Linux, your package manager for your distribution may have Logisim.
- You can find a brief tutorial for Logisim on Professor Farrens' class website.
- The **User's Guide** and **Library Reference** in the Help menu in Logisim are also very helpful.

Introduction

When designing a circuit to implement a function, there are two different design routes that you can take: use pure combinational logic, or use a Read Only Memory (ROM). Combinational logic involves combining AND, OR, and NOT gates to implement the Boolean equation that you derive from the truth table. On the other hand, by using a ROM, you simply implement the truth table in hardware.

ROM Implementation

A memory unit can be viewed as simply a truth table in hardware. Here are the steps to creating a ROM implementation of a truth table.

1. Create a ROM, where the number of addressing bits is equal to the number of inputs, and the data bit width is equal to the number of output bits.
2. Using the input bits as the address, fill in the entries of the ROM with the correct outputs for that combination of inputs.
3. To get the output, address the ROM using the concatenation of the input signals.

Example

The following is an example of implementing the XOR function using microcode. Here is the truth table for XOR $\oplus$.

X	Y	X $\oplus$ Y
0	0	0
0	1	1

1	0	1
1	1	0

Applying step 1, we first create a ROM with 2 address bits, because we have 2 inputs. Each entry is 1 bit wide, because there is only output. Next, we fill in the ROM using X and Y as the address bits.

X	Y	Address	Value
0	0	`0x0`	0
0	1	`0x1`	1
1	0	`0x2`	1
1	1	`0x3`	0

The Logisim implementation is included with the given files for the assignment, in the subcircuit **ROM XOR**.

Assignment

1. Implement a combinational circuit using combinational logic.
2. Implement a simple combinational circuit using a ROM.
3. Implement the combinational circuit from part 1 using a ROM

For each circuit, please create a sub-circuit with appropriately named inputs and outputs.

1) Combinatorial Circuit - Combinational Logic

Implement the circuit that has the following truth table using **combinational logic**.

- When creating the combinational circuit using combinational logic, you may only use splitters and these gates: AND, OR, and NOT.
- The logic should be the minimal amount to express the truth table. Do not worry too much about this -- this will be true if the circuit is generated using Logisim's built-in analysis tool, mentioned below.
- **In** are the inputs and **Out** are the outputs.
- All unspecified input and output combinations are don’t cares, as are the **D**s in the table.

In7	In6	In5	In4	In3	In2	In1	In0	Out2	Out1	Out0
0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1	0	1	0
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	1	1	0	1	0
0	0	0	0	0	1	0	0	0	1	1
0	0	0	0	0	1	0	1	1	1	1
0	0	0	0	0	1	1	0	1	0	0
0	0	0	0	0	1	1	1	0	0	0
0	0	0	0	1	0	0	0	1	0	0

0	0	0	0	1	0	1	0	1	1	1
0	0	0	0	1	1	0	0	1	0	1
0	0	0	0	1	1	1	0	0	1	1
0	0	0	1	D	D	D	D	0	1	0
0	1	1	0	D	D	D	D	1	1	1
1	0	1	0	D	D	D	D	1	1	0
1	1	1	0	D	D	D	D	0	1	1

Your sub-circuit should have the following inputs and outputs:

Inputs

- *Input*: the concatenation of the input bits.

Outputs

- *CombinationalOutput*: The concatenation of *Out2-0*, with *Out2* as the top-most (most significant) bit and *Out0* as the bottom-most (least significant) bit.

Circuit Analyzer Tool

Don't be intimidated by the number of inputs when doing the combinational circuit. You can use Logisim's **Analyze Circuit** tool, in the Project drop-down menu, to have Logisim build the circuit for you. To learn how to use it, click on Help → User's Guide. In the User Guide, click on Combinational Analysis and read how to use it. You will find this tool very helpful in this and future labs, if implementing complex combinational logic with gates.

2) Small Combinational Circuit - ROM

First, implement the following simple truth table as follows using a **ROM**. You will find an example of how combinational logic is done in the **ROM XOR** circuit in the provided circuit file. You may only use a ROM and splitters for this part. Your sub-circuit should have the following inputs and outputs:

Inputs

- *SimpleInput*: the concatenation of the input bits.

Outputs

- *ROMSimpleCombinationalOutput*: The concatenation of *Out2-0*, with *Out2* as the top-most (most significant) bit and *Out0* as the bottom-most (least significant) bit.

In3	In2	In1	In0	Out1	Out0
0	0	0	0	1	1
0	0	0	1	1	0
0	0	1	0	0	1
0	0	1	1	1	1

0	1	0	0	1	0
0	1	0	1	0	1
0	1	1	0	1	1
0	1	1	1	1	1
1	0	0	0	1	1
1	0	1	0	1	1
1	1	0	0	1	0
1	1	1	0	1	1
1	1	1	1	D	D

3) Large Combinational Circuit

Then, once you have become familiar with using a ROM for combinational logic, implement the same circuit from Part 1 using a **ROM**. The truth table is identical for both circuits. You may only use a ROM and splitters for this part. Your sub-circuit should have the following inputs and outputs:

Inputs

- *Input*: the concatenation of the input bits.

Outputs

- *ROMCombinationalOutput*: The concatenation of *Out2-0*, with *Out2* as the top-most (most significant) bit and *Out0* as the bottom-most (least significant) bit.

Testing

You will be provided with the following circuits to facilitate testing.

- **Combinational Input**: Generates the inputs for the combinational circuit.
    - **Inputs**:
        - *Clock*: The system clock.
    - **Outputs**:
        - *Input*: The concatenation of the input signals *In7-In0* to the combinational circuit.
- **Simple Input**: Generates the inputs for the simple combinational circuit.
    - **Inputs**:
        - *Clock*: The system clock.
    - **Outputs**:
        - *SimpleInput*: The concatenation of the input signals *In3-In0* to the combinational circuit.

You will also be provided with the following log files to test if your circuits are correct:

- **part1correct.txt**

- The log file containing the correct outputs for the combinational logic circuit using combinational logic.
- The X's in the file indicate don't cares.

- **part2correct.txt**
  - The log file containing the correct outputs for the simple logic circuit using a ROM.
  - The X's in the file indicate don't cares.
- **part3correct.txt**
  - The log file containing the correct outputs for the combinational logic circuit using a ROM.
  - The X's in the file indicate don't cares.

We will be testing your code using Logisim's logging feature. To log the results of your program, do the following:

1. Attach a probe or pin to the wires that you want to log, and give it a name.
2. Click Simulate → Logging.
3. In the Selection tab, select the signals you want to log.
4. Click on the File tab.
5. Select a file to log the signals to.

You will need to create three separate log files, one for each sub-circuit:

Signal Name	Radix	Description
Input	2	The concatenation of In7-0.
CombinatonalOutput	2	The concatenation of Out2-0 from the combinational circuit.

Signal Name	Radix	Description
SimpleInput	2	The concatenation of SimpleIn3-0.
ROMSimpleCombinationalOutput	2	The concatenation of Out2-0 from the ROM combinational circuit.

Signal Name	Radix	Description
Input	2	The concatenation of In7-0.
ROMCombinationalOutput	2	The concatenation of Out2-0 from the ROM combinational circuit.

To see if your circuit is correct, use the Python program, `tester.py`, included with assignment. To use it, type, in your command line, with all files in the same directory:

```
python tester.py correct.txt your.txt
```

where `correct.txt` is the file that contains the correct signals, and `your.txt` is the name of the log file you have your signals in. For example, to test if your combinational circuit is correct, you would type:

```
python tester.py part1correct.txt part1.txt
```

if your log file was named `part1.txt`.

The output should be empty if no inconsistencies were found.

The tester was written for Python 2.7. If you want to use Python 3, you will need to encapsulate each print statement's argument with parentheses. If you are using Windows, you may want to add Python to your system path to make testing easier, if you have not already.

Resetting the Log Files

If your circuit has some errors the first time, in order to retest your file, you should perform the following steps to guarantee a maximally clean, consistent log file:

1. Disable logging file output through the logging window, under the file tab.
2. Delete or move the previously generated log file.
3. Reset your circuit with Ctrl + R, going to Simulate → Reset Simulation, or whatever mechanism by which you return the circuit to its intended starting state.
4. Re-enable logging file output through the logging window.
5. Simulate again.
6. Run `tester.py` again.

If the first line of your log file has a line in which the last number is missing, you may safely delete it. Additionally, if you reset your circuit while you are still logging, you will notice dashes in the log file indicating when the circuit was reset. You may delete everything from those dashes up to the headers to clear your log file.

Grading

- 50% Implementation
    - 12.5% for correct Part 1.
    - 12.5% for correct Part 2.
    - 25% for correct Part 3.
    - Partial credit at the grader's discretion.
- 50% Interactive Grading
- It is possible to receive a lower grade than what you earned, if you do not understand how your implementation works.
- You must attend interactive grading to receive a grade for this project.
- Times for interactive grading will be posted close to when the assignment is due.

Submission

Warning: read the submission instructions carefully. Failure to adhere to the instructions will result in a loss of points.

- Upload to Canvas the zip/tar of your .circ file along with a README file that contains:
    - The names of you and your partner.

    - Any difficulties you had.
    - Anything that doesn't work correctly and why.
    - Anything you feel that the graders should know.
- **Copy and paste the README into the text submission box when you are submitting your assignment**, as well.
- Only one partner should submit the assignment.
- You may submit your assignment as many times as you want.

Hints

- When filling in the values for the ROM in the combinational circuit, it may be worthwhile to write a program to fill in the values for the ROM. If you don't, you may have to fill in a large amount of numbers by hand. It is by no means required, though.
- If you need help, come to office hours for the TAs, or post your questions on Piazza.