ECS-154B, Winter 2013, Lab 1

Due by 11:55 p.m. on Jan 21, 2014

Via Smartsite

Goals

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

Logisim

- You can download logisim at http://ozark.hendrix.edu/burch/logisim/index.html
- You can find a brief tutorial for logisim at http://american.cs.ucdavis.edu/academic/ecs154a/postscript/logisim-tutorial.pdf
- I also found the user guide and library reference under the help menu in logisim to be very helpful.

Introduction

When designing a circuit to implement some 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, where as 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.

XOR		
X	Y	X XOR Y
0	0	0
0	1	1
1	0	1
1	1	0

Above is the truth table for XOR.

Applying step 1 we first create a ROM with 2 address bits, because we have 2 inputs, whose entries are 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	0	0
0	1	1	1
1	0	2	1
1	1	3	0

The logisim implementation is included with the given files for the assignment if you want to look at it. Its name is ROMXOR.

ROM and Sequential Circuits

You can also use a ROM to implement Morre sequential circuits. Now instead of implementing the truth table, the ROM implements the next state transition table and the outputs in hardware. Here is how to do it

1. Create a ROM where the number of address bits is equal to the number of state bits + the number of input bits. The data bit width is the number of state bits + the number of output bits.
2. Fill in the ROM using the state transition table. Each entry in the ROM contains the next state bits along with the output at that the current state.
3. To address the ROM concatenate the next state bits from the ROM with the input signals. (A portion of the output of the ROM feeds back into it self)

Example

Suppose we want to create a sequential circuit that while it is in state 0, it outputs 01, when in state 1 it outputs 11, and changes between states whenever the input, X, is 1. The machine starts in state 0. The transition table for this machine would be as follows.

Current State	X	Next State	Output
0	0	0	01
0	1	1	01
1	0	1	11
1	1	0	11

Applying step 1 we would first create a ROM that has 2 address bits, 1 for the next state and 1 for the input X, that has data entries that are 3 bits wide, 1 for the next state and 2 for the output. Applying step 2 and using the top bit in each entry in the ROM to store the next state and the bottom 2 for the output

Current State	X	Address	Next State	Output	Value In ROM
0	0	0	0	01	1
0	1	1	1	01	5
1	0	2	1	11	7
1	1	0	0	11	3

Because of the way that we decided to implement the machine the top most output of the ROM feeds back into the top most bit of its input. A logisim implementation of this circuit is included with the assignment. Its name is ROMSeqEx. It would probably be very helpful to look at.

Sequential Circuits using ROM in logisism

In order to implement sequential circuits with using ROM, the ROM must be synchronous. Logisim does not come with a built in synchronous ROM but you can make your own by connecting a register to the output of the built in ROM module.

Assignment

For this assignment you will implement a combinational circuit using both combinational logic and a ROM, and a sequential circuit using only a ROM. For each circuit please create a subcircuit with appropriately named inputs and outputs.

Combinational Circuit

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

- When creating the combinational circuit using combinational logic you can only use AND, OR, NOT, XOR gates and splitters.
  - In addition the logic must be the minimal amount to express the truth table
- When creating the combinational circuit using a ROM, you can only use the ROM module.
- In are the inputs and Out are the outputs.
- All unspecified input/output combinations are don't cares.

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

Your subcircuit should have the following inputs and outputs.

Inputs:

- In7-0

Outputs

- Out: The concatenation of Out2-0, with Out2 as the top most bit and Out0 as the bottom most bit.

The Circuit Analyzer Tool

Don't be intimidated by the number of inputs when doing the combinational circuit. You can use logisim's circuit analyzer tool under 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 Combinational Analysis and read how to use it. You will likely find this tool very helpful in this and future labs.

Sequential Circuit

- For this section you can only use the ROM, Register, Splitter, and Clock modules in logisim.

Implement a Moore Model sequential circuit that takes as input a stream of bits. If at least 2 bits out of every 4 are 1 the circuit should output 1 after seeing the 4th bit and 0 at all other times. The circuit Example with bits being input to the machine from right to left

Input: 0000 0001 0011 1111

Output: 0 0000 0001 0001 0000

Remember since we are implementing a Moore model the output will be delayed by one clock tick.

Your subcircuit should have the following inputs and outputs.

Inputs:

- Clock: The system clock
- X: The current value in the input bit stream

Outputs

- Out: The output from your sequential circuit.

Testing

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

- CombinationalInput: Generates the inputs for the combinational circuit
  - Inputs:
    - Clock: The system clock
  - Outputs: From top to bottom In7-In0, the input signals to the combinational circuit
- SequentialInput: Generates the inputs for the sequential circuit
  - Inputs:
    - Clock: The system clock
  - Outputs:
    - X: The input bit stream for your sequential circuit
    - NotDone: 1 when the test is still ongoing. Connect this to the sel input on your ROM.

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

- CombOut.correct.log
  - The log file containing the correct outputs for the combinational logic circuit using combinational logic.
  - X's in the file indicate don't cares
- ROMCombOut.correct.log
  - The log file containing the correct outputs for the combinational logic using a ROM.
  - X's in the file indicate don't cares
- ROMSeqOut.correct.log
  - The log file containing the correct outputs for the sequential circuit using a ROM.
  - 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 subcircuit.

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

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

ROMSeqOut.log		
Signal Name	Radix	Description
X	2	The input bit stream X.
ROMSeqOut	2	The output from the sequential circuit.

To see if your circuit is correct use the python program, test.py, included with assignment. To use it type

```
python tester.py correct_log_file    your_log_file_name
```

where correct_log_file is the file that contains the correct signals and your_log_file_name 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 CombOut.correct.log CombOut.log
```

if your log file was named CombOut.log

Important!:

If your circuit has some errors the first time, in order to retest you file you must do the following steps in this order

1. Reset your circuit by pressing cntrl + r
2. Delete the contents of your log file except for the headers (the names of the signals)
3. Simulate again
4. Run test.py again

This is the first time that we are using the log function to grade. If you find some errors in the testing program please post them to Picassa.

Grading

- 95% Implementation
    - 1/3 for each circuit
- 5% Interactive Grading
    - If there are still errors in your circuit you will have 5 minutes to fix them in interactive grading.
- It is possible to receive a lower grade than this 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 after the assignment is due.

Submission

- Submit your .circ file on Smartsite.
- Also submit 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 I should know
- You can submit your assignment as many times as you want
- You have 7 slip days that you can use.

Hints

- When filling in the values for ROM in the combinational circuit it would probably be a good idea to write a program to fill in the values for the ROM. If you don't you would have to fill in 256 values by hand.

- If you need help come to office hours or post your questions on Piazza.