CPSC 121: Models of Computation
Lab #4: Time-Division Multiplexing

Objectives

In this lab, you will be introduced to the method of time-division multiplexing which uses a type of circuit called a demultiplexer. You will also learn about the Arithmetic Logic Unit (ALU) and see how multiplexers are used in it.

1 Pre-lab

Imagine you are calling a friend in Asia. While you are talking, there is one signal connecting you and your friend. However, it would be too expensive to directly wire every telephone in North America to every telephone in Asia. Laying one wired connection across the Pacific Ocean is expensive enough.

Have a look at the animation at http://en.wikipedia.org/wiki/File:Telephony_multiplexer_system.gif. Here we see a solution for wiring telephones in North America to telephones in Asia: have one wire between the continents, and connect telephone conversations as needed. (This technique is known as *time-division multiplexing*.)

TODO (pre-lab): In the animation, five conversations are being connected. Draw what it would look like if six conversations were being connected.

The connection scheme in the animation uses a circuit called a *demultiplexer* to undo the effect of the multiplexer. The multiplexer selects the caller's signal and passes it to the demultiplexer, which then selects the appropriate receiver to send the signal to, thus connecting the two friends.

A 1-to-2 demultiplexer has two inputs: `s`, analogous to `s` (select) in our multiplexer, and `y`, an input wire.
It produces two outputs: `x0` and `x1`, the demultiplexed signals. It works like this:

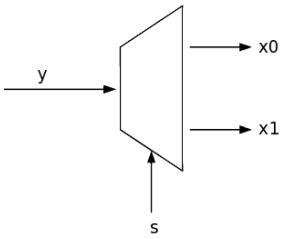
- When `s` = 0, `x0` outputs the value of `y`, and `x1` outputs LOW.
- When `s` = 1, `x1` outputs the value of `y`, and `x0` outputs LOW.

TODO (pre-lab): Fill in a truth table for the demultiplexer.

s	y	x0	x1
0	0		
0	1		
1	0		
1	1		

TODO (pre-lab): Provide a circuit diagram for a 1-to-2 demultiplexer.
Your diagram should use specific gates, it should NOT be the symbol shown below.

The symbol for a demultiplexer looks like the opposite of a multiplexer:



In the previous lab, you saw that you could make a 1-of-4 multiplexer out of three 1-of-2 multiplexers. **TODO (pre-lab): Provide a diagram of how you could make a 1-to-4 demultiplexer using 1-to-2 demultiplexers.** Your 1-to-4 demultiplexer will have 4 outputs: x0, x1, x2, and x3. The table below shows which output returns the value of y for different values of select1 and select0. Your diagram **must** agree with the table.

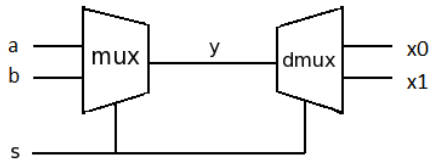
select1	select0	output that returns y
0	0	x0
0	1	x1
1	0	x2
1	1	x3

2 Demultiplexers

TODO: Using the circuit diagram from the third TODO of your prelab, implement a 1-to-2 demultiplexer on the Magic Box. Leave enough room on your breadboard to wire up a multiplexer for the next part of this lab. Verify that it works as anticipated and then show it to your TA.

Next, you will be implementing a multiplexer-demultiplexer scheme on the Magic Box.
The instructions below detail this process:

1. Turn off the power to the Magic Box while wiring.
2. Wire up a **1-of-2 multiplexer chip (LS157)** next to your demultiplexer. Connect power, ground, and the enable input. (Remember from Lab 3 that enable should be *low*.) Also connect one pair of inputs, a and b, to two switches.
3. Connect the two select inputs by wiring the s input of the demultiplexer chip to the s input of the multiplexer. Now, the multiplexer and demultiplexer share the same select signal.
4. Next, connect the out output of the multiplexer to the y input of the demultiplexer. The multiplexer and demultiplexer should now be connected as shown in this diagram:



5. Verify that your mux-demux scheme works as expected by trying out different combinations of a, b, and s.

Now connect the select inputs of the multiplexer and demultiplexer to a clock output by removing the s input of the demultiplexer from the switch it is connected to and connecting it to the Magic Box's **clock output**. Remember that only the last three contact points connect to the clock signal as the first one connects to the probe.

TODO: Set both a and b to see what happens when your clock output is set to 1 Hz. Then do this again with a clock output of 2 Hz, and again at 1 kHz. What do you notice? (Note: see the Magic Box User Manual page 6 for changing the clock frequency.)

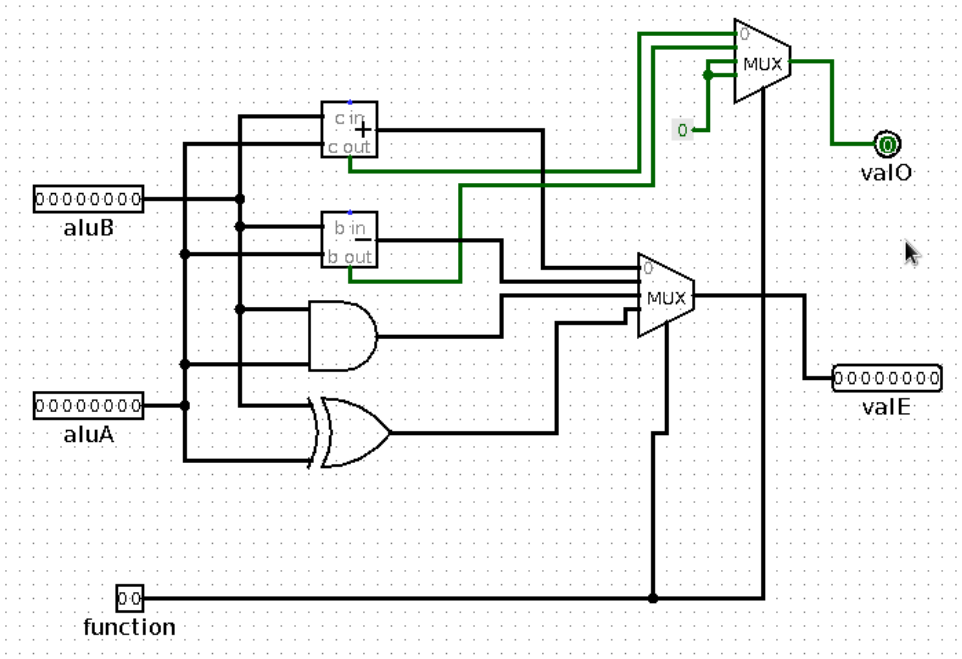
3 Multiplexing

TODO: Using Logisim, implement a circuit with three inputs: a, b, and s. This circuit should work as follows: when `s` is off, the output is `a` AND `b`. When `s` is on, the output is `a` XOR `b`. As such, the `s` input will be selecting which function to apply to `a` and `b` – do we AND or XOR? (*Note*: Multiplexers can be found in the `Plexers` folder in Logisim. You do not have to construct one from AND, OR and NOT gates).

4 An Arithmetic Logic Unit

Download the file `conciseALU.circ` from the course website, and load it in Logisim. It will look like the image below. This circuit simulates a component of a fully-functional computer processor, like the one in your computer. Over the course of the term, you will gain the tools needed to explore and understand any part of this circuit. Today, we will only look at one component of the processor: the Arithmetic Logic Unit (ALU). The ALU is the processor's "calculator" and is responsible for its arithmetic and logical calculations.

Notice there are three inputs: `aluA`, `aluB`, and `function`. There are two outputs: `valE`, and `valO`.



TODO: By analogy to the last circuit from Section 3, describe how the `aluA`, `aluB`, and `function` inputs determine the value of `valE` – what does this value represent?

TODO: By analogy to the last circuit from Section 3, describe how the `aluA`, `aluB`, and `function` inputs determine the value of `valO` – what does this value represent?

5 Further Analysis

The Concise Y86 ALU only has the functions to add, subtract, AND, and XOR. **TODO (further analysis): How could the Y86 ALU be used in a larger circuit to multiply numbers or divide numbers? Are there any values that it cannot evaluate? Why or why not?**

6 End of Lab Survey

TODO: To help us improve these labs both this term and for future offerings, complete the survey at http://www.tinyurl.com/cs121labs.

7 Magic Box cleanup

TODO: Before leaving the lab, show your Magic Box to your TA.

A Marking scheme

All labs are out of ten marks, with two marks for pre-labs, and eight marks for in-lab work. In more detail:

- Two marks - Pre-lab questions
- Five marks - In-lab questions. In this lab, it is one mark for the demltiplexer implementation, one mark for the mux-demux scheme, one mark for the multiplexing circuit, and two marks for the concise ALU questions.
- One mark - Further analysis.
- One mark - End of lab survey.
- One mark - Magic Box cleanup.

TAs may at their discretion award one bonus mark, such as for completing the challenge problem.

B Challenge Problem

Addition is a basic operation in many ALUs. Using only basic logic gates (ie. AND, OR, XOR, NAND), design a circuit that takes two two-bit binary numbers and produces their sum. The circuit must be able to output the sum `11 + 11` correctly.