Computer Systems, Spring 2019

Week 2 Lab

Numbers and Logic Gates

This lab exercise is unassessed: there is nothing to hand in. It covers some basic concepts that will be used throughout the course, including the assessed quizzes, program and examination. There are some problems to work out with paper and pencil, and some problems using a circuit simulator called LogicWorks to try on the computer.

Problems to work out on paper

1. Convert the 8-bit word 0100 1101 to decimal, assuming binary representation.
2. Convert 89 to an 8-bit binary number.
3. Consider the 8-bit word 1001 0110.
   (a) Convert it to decimal, assuming binary representation.
   (b) Convert it to decimal, assuming two's complement representation.
   (c) Explain, in your own words, why a word (such as 1001 0110) has no inherent meaning, but it has multiple meanings corresponding to different number systems (e.g. binary, two's complement, and others that we haven't discussed).
4. Calculate $23+59$ by converting both numbers to 8-bit binary numbers and doing binary addition. Check the result by converting back to decimal.
5. State whether each of the following conversions is possible, and give the reason it is or isn't possible. (But you don't have to do the actual conversion!)
   (a) Convert 12 to a 4-bit binary number.
   (b) Convert 12 to a 4-bit two's complement number.
   (c) Convert 73 to a 5-bit binary number.
   (d) Convert -5 to an 8-bit binary number.
   (e) Convert -128 to an 8-bit two's complement number.
   (f) Convert 128 to an 8-bit two's complement number.
6. (Optional) This experiment is a small introduction to research—see if you can make a surprising discovery! Recall that (1) we have seen in the lectures how to add binary numbers; (2) we have seen that binary and two's complement are different number systems; and (3) we have *not* discussed in lectures how to add two's complement, or how to subtract anything at all. Now, try the following. (In some cases you may observe a carry of 1 from the leftmost position—if that happens simply ignore the carry and just consider the 8-bit sum result you get.)

(a) Convert 95 and -30 to 8-bit two's complement numbers.

(b) Add these 8-bit words using the method for binary addition, *even though they are NOT binary numbers.*

(c) Convert the result back to decimal. What do you observe?

(d) Try the same thing with some different values, such as $24-59$, $-9+28$, and other examples you make up.

Do your experiments suggest a hypothesis? Why is the hypothesis surprising? Can you think of any practical applications?

7. Write down the truth tables for the following logic gates: inv, and2, or2, xor2. You need to know these!

Problems using the computer

LogicWorks is a "schematic capture" software tool for drawing a digital circuit and simulating it. The aim of this part of the lab is to learn the basics of LogicWorks, and to understand the behaviour of a few simple circuits. Read the document *Notes on using LogicWorks*, which is available on Moodle.

1. Launch LogicWorks. Load the circuit called **switch-inv-probe.cct** (it's on Moodle). This is the simplest circuit: just an inverter whose input is connected to a binary switch and whose output is connected to a binary probe. Clich the switch several times and observe the output.

2. Now you will create a circuit, which is similar to the previous example. Put an and2 logic gate on the canvas. Put two *Binary Switch* components on the canvas, and connect them to the inputs of the logic gate. Put a *Binary Probe* component on the canvas, and connect it to the output of the logic gate. Make the diagram reasonably neat and readable. Develop a truth table by simulating the circuit on all possible inputs. To do this, take the first line of the truth table (with inputs 00) and set the binary switches to those values; observe the result on the binary probe and enter it into the output column. Repeat this process for each line in the truth table. Verify that your simulation gives the correct truth table for the component.

3. Test the or2 and xor2 logic gates the same way.

4. In LogicWorks, load the half adder circuit **HalfAdd.cct** (on Moodle). Generate a truth table for the circuit (just as you did for the and2 gate). Check that the half adder is working correctly.

5. Load the full adder circuit **FullAdd.cct** (on Moodle) and work out its truth table using simulation. Check that the circuit is working correctly.

6. Enter the multiplexer circuit mux1 (see the lectures) into LogicWorks, and save it as Mux1.cct. This circuit isn't on Moodle; you need to draw it on the canvas. Simulate it and write out its truth table.