Worksheet 4 (Lab)

In the next few labs, we will be using a software package called LogicWorks to simulate digital circuits. This worksheet contains three exercises. The first two exercises are for you to become familiar with LogicWorks. The third exercise is more substantial. The exercises are unassessed, but you should hand in your design work for **Exercise 3** at the end of the lab. Your tutor's comments will help you to understand circuit design, which is an important part of the course.

Exercise 1: Getting Started with LogicWorks

In AMS, set up the CS1Q exercise called "Week19". In your CS1Q folder there should now be a folder called "Week19". Within this folder is a file called "Exercise1". Start LogicWorks ("Start" menu then "Programs" menu), select "Open" from the "File" menu, and select "Exercise1". Note that simply double-clicking on a LogicWorks file in your workspace does not work properly.

The LogicWorks window is divided into four main areas: at the top, the *menus* and *toolbar*; at the bottom, the *timing area*, which we will not be using; on the right, the *component library*, and in the center, the *design area*.

The following steps will take you through the construction of a simple circuit which allows an AND gate to be tested.

1. In the component library, scroll down until you see a component called "AND-2". Select it by double-clicking. Click somewhere in the design area to place the component. You will see that the AND gate is still attached to the pointer, so that another one can be placed. You only need one for the moment, so click on the "Pointer" button in the toolbar to return to a normal pointer.

2. Select a "Binary Switch" from the component library, and click in the design area to place it somewhere to the left of the AND gate. Click again to place a second switch below the first one.

3. Click on the "Draw signal" button in the toolbar, which has a light-weight + symbol on it. The pointer will change to a + symbol. Click on the output of one of the switches (the horizontal line on the right), then on one of the inputs of the AND gate. You should see a red line, representing a wire, joining the switch to the input. If this does not work, try clicking again, closer to the output or input.

4. You can correct any mistakes by changing the pointer to an eraser (click on the lightning bolt symbol to the right of the pointer button, in the toolbar) and clicking on any part of the circuit that you want to delete.

5. Connect the output of the second switch to the second input of the AND gate.

6. Select a "Binary Probe" from the component library and place it to the right of the AND gate. Connect the output of the AND gate to the input of the binary probe.

7. Click on the "Run simulator" button, at the right of the second row of the toolbar. Notice that the binary probe changes from "Z" to "0". By clicking on the switches (make sure that you have a normal pointer) you can change the values of the inputs of the AND gate, and see the corresponding output on the binary probe.

8. As with other applications, there is a "Save" option in the "File" menu, which you should use to save your work.

Exercise 2: The Majority Voting Circuit

1. Close "Exercise1" and open "Exercise2".

2. Build the majority voting circuit from the lecture notes (Lecture 5 Slide 7). Connect binary switches to the inputs and connect a binary probe to the output.

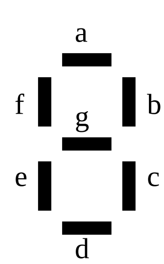
3. Run the simulator and test the circuit, by setting the input switches and observing the output. Construct a truth table by systematically testing all 8 combinations of input values.

4. Check that this truth table is the same as the one in the lecture notes.

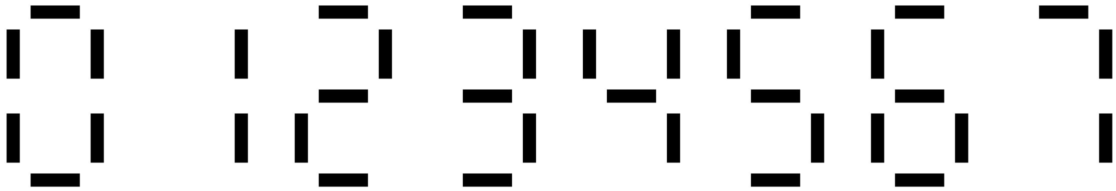
Exercise 3: Driving a 7 Segment Display

Close "Exercise2" and open "Exercise3".

In LogicWorks, select the component "7-Seg Disp - Black " from the component library and place it in the design area. This is the single digit display from last week's tutorial. It represents a single digit display of the type used in, for example, digital watches. The 7 segments (bars) can each be either lit or unlit. For each segment there is a binary input (labelled $a \ldots g$) which should be set to 1 in order to light that segment. There is also an input for the decimal point (labelled "*dot*"), which we won't use; connect it to a "GND" symbol so that it is fixed at 0. The segments are labelled as follows.



The aim of this exercise is to build a circuit which converts a 3 bit binary input, representing a number from 0 to 7, into the correct values for the inputs $a \ldots g$ of the display. The usual way of representing the digits is as follows.



1. Last week you should have worked out a truth table showing $a \ldots g$ as functions of x, y, z, where xyz is the binary representation of the digit to be displayed. If you did not do this last week, do it now.

2. For each of $a \ldots g$, use a Karnaugh map to work out a minimised formula.

3. Draw a circuit which calculates $a \ldots g$ from xyz. There might be opportunities to reuse subformulae; it might also be possible to achieve further simplifications by factorising parts of the formulae. Draw the circuit on paper before you start building it in LogicWorks. This will help you to see how to lay it out neatly and help you not to make mistakes.

4. Build the circuit, connect binary switches to the inputs, and test it. It is best to organise the circuit neatly, so you can keep track of which components are doing what. It is also a good idea to build and test the parts for $a \ldots g$ one at a time.

Hand in your design work from this exercise (Questions 1, 2, 3) at the end of the lab.

Supplementary

Repeat this exercise with a 4 bit input, so that the appropriate hexadecimal digit is displayed.

In a real electronic device, the 7 segment display driver might be better implemented by using the input value to address a small area of ROM (read only memory) containing a 7 bit word for each digit; this word would be fed into the display.
