Introduction to the Use of Computers

Binary logic, integers and arithmetic

Friday 12th October 2012

This lab session is about the binary notation system and its use in binary logic, and in the representation and manipulation of integers in computers.

1. This part of the lab allows you to interactively design and investigate logical circuits.

   (a) using a web browser, navigate to `http://www.doc.gold.ac.uk/~mas01cr/teaching/is50004a/Logic-Sim/`. You should find a mostly blank page, with some drag-and-droppable graphical components on the left hand side. These components represent logic gates, and representations for inputs and outputs. Drag and drop an input, the NOT gate, and an output onto the white area, and connect them up with wires (by click-and-dragging). Verify that the output changes appropriately when you change the input value (by clicking on it).

   (b) using just the NAND gate and wires, implement the AND logic gate; verify that it works using two inputs and an output indicator.

   (c) using just the NAND gate and wires, implement the OR logic gate; verify that it works using two inputs and an output indicator.

   (d) using just the NAND gate and wires, implement the XOR logic gate; verify that it works using two inputs and an output indicator.

   (e) using all the available logic gates, implement a one-bit half-adder; verify that it works using two inputs and two output indicators.

   (f) experiment with the other items on the page. Can you describe what they do?

2. This part of the lab is a series of questions to reinforce the material covered in lectures.

   (a) For each of the following pairs of quantities, state which is bigger (assuming the strict meanings of SI prefixes):

      i. 1MB, 1MiB
      1MiB = 1048576B > 1000000B = 1MB

      ii. 1000KiB, 1024kB
      1000KiB = 1024000B = 1024kB

      iii. 1Mb, 1024KiB
      1Mb = 125000B < 1048576B = 1024KiB

      iv. 8kb, 1KiB
      8kb = 1000B < 1024B = 1KiB

      v. 10000GB, 1TB
      10000GB $= 10^{13}B > 10^{12}B =$ *1TB*

   (b) Express each of the following numbers in binary representation:

      i. $(11)_{10}$
      $(1011)_2$

      ii. $(101)_{10}$
      $(1100101)_2$

iii. $(\mathrm{af})_{16}$
(10101111)$_2$
iv. `0x6e`
(01101110)$_2$
v. $(255)_{10}$
(11111111)$_2$

(c) Express each of the following numbers in decimal representation:

i. $(11)_2$
(3)$_{10}$
ii. $(101)_2$
(5)$_{10}$
iii. $(\mathrm{af})_{16}$
(175)$_{10}$
iv. `0x6e`
(110)$_{10}$
v. $(1100011)_2$
(99)$_{10}$

(d) Express each of the following numbers in hexadecimal representation:

i. $(11)_{10}$
(b)$_{16}$
ii. $(101)_{10}$
(65)$_{16}$
iii. $(255)_{10}$
(ff)$_{16}$

(e) Explain, with the use of a diagram if appropriate

i. how logic gates can be used to construct a one-bit half-adder;
ii. how two half-adders and some logic gates can be used to construct a one-bit full-adder;
iii. how full-adders can be chained to make multiple-bit adders.