# Week 3 Lab B: Multiplexers

## Objectives

Develop understanding and experience of:

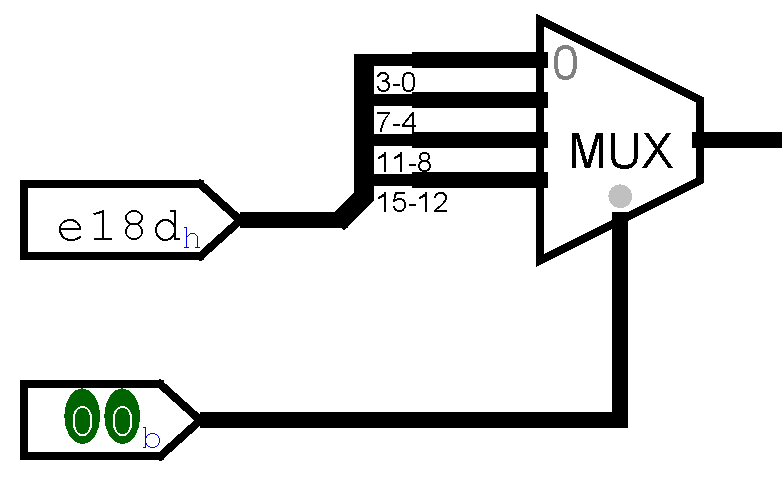
1. Using a multiplexer to select one from a set of signals.

## Investigate Multiplexers

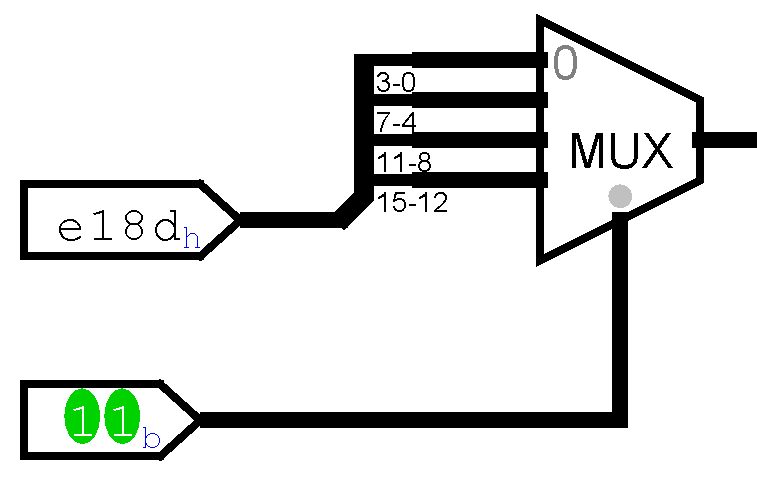
A multiplexer will be a very useful component as we start to connect components together to form a programmable computer. The select signal indicates which of the values should be passed on.

1. General information about multiplexers
2. How many bits will the select signal need if we have two choices? 1
3. How many bits will the select signal need if we have four choices? 2
4. How many bits will the select signal need if we have eight choices? 3
5. Predict the output from a multiplexer.

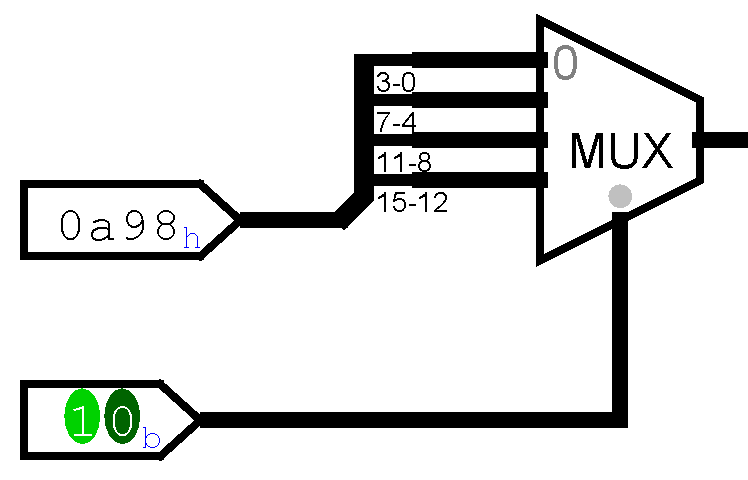
The following multiplexer passes on a different part of the input data according to the select signal. Note that the data input is 16 bits which is shown as 4 hex digits. The select input is 2 bits. Remember that bits are numbered from the least significant bit, so bits 3-0 are the right-most hex digit. What would be the output in each of these cases? The first one is done for you.



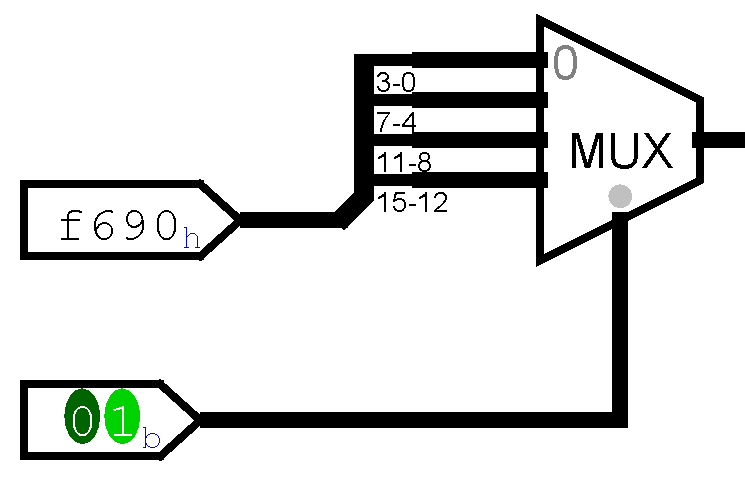
* 1. Output **d**



* 1. Output e



* 1. Output a

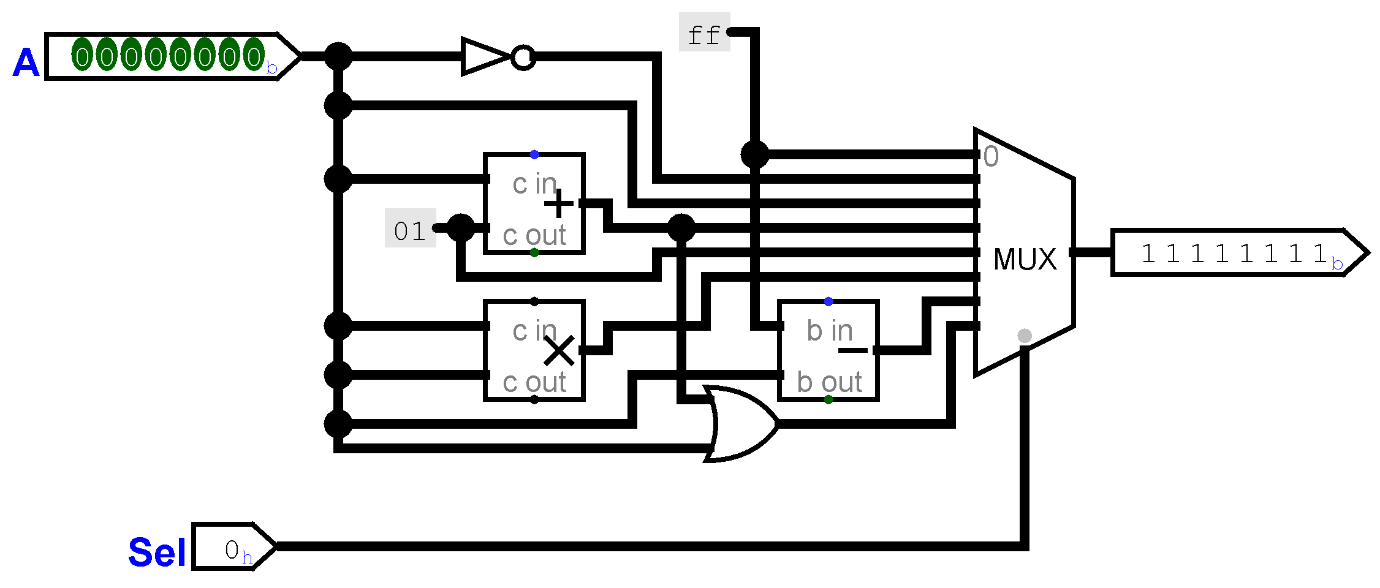


* 1. Output 9

1. Mystery Multiplexer.

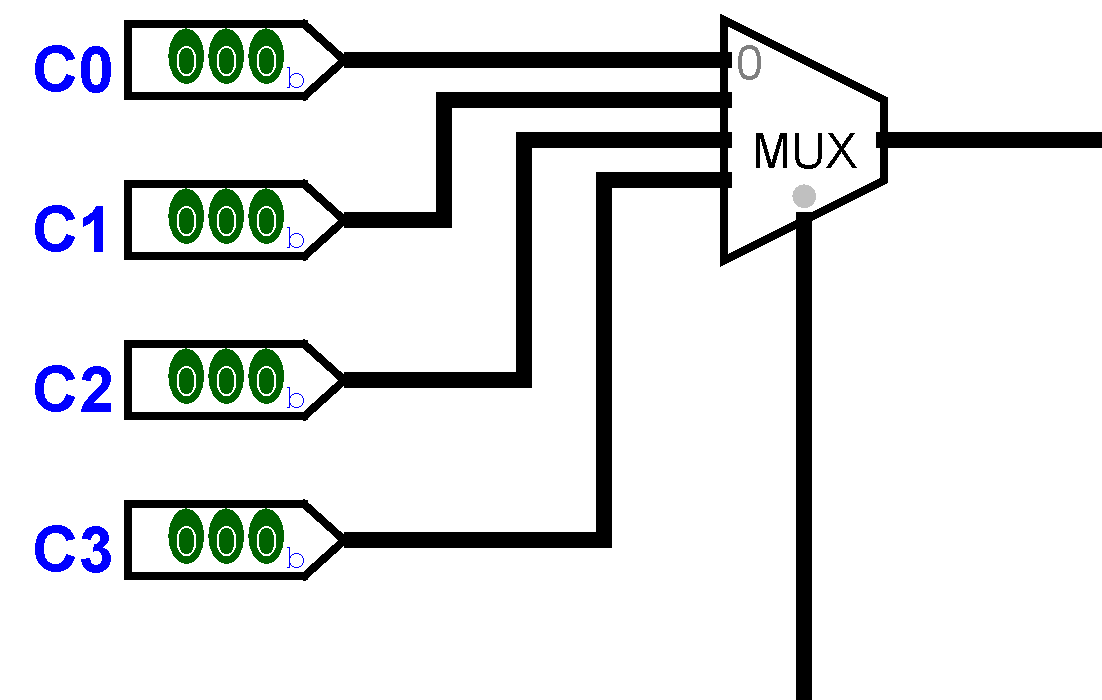
You are not expected to build the circuit shown below. Trace the lines back from the input choices to the MUX to fill in the table with a short description of what each choice does. The first two have been done for you

Remember that a dot indicates a signal that is being copied along multiple paths. When lines just cross over each other they do so without affecting each other. Note that the select input has the pin showing in hexadecimal.



|  |  |
| --- | --- |
| Select | Output |
| 0 | Hex ff which is 11111111 in binary (whatever value A has) |
| 1 | The result of a bitwise NOT on A |
| 2 | Same as A |
| 3 | Add the constant 01 to a |
| 4 | Constant 01 |
| 5 | A is multiplied by a |
| 6 | FF minus A |
| 7 | A + 1 or A |
|  | Select input is displayed |

1. In Logisim Evolution, create a circuit that has four 3-bit input pins as the input choices (labelled C0 to C3 below in the partial image below). Use a separate input pin for the select input (not shown below). All inputs need to all feed into a multiplexer (from the Plexers tools in Logisim Evolution).

Set the select bits and data bits properties for the multiplexer. The select bits controls how many choices the multiplexer can handle. The data bits setting is the bit width of each choice.

Connect the output of the MUX to a splitter and split the signal into its three separate bits. Connect the separate bits to the three inputs of an RGB LED from the Input/Output tools in Logisim Evolution. Make sure that you keep the order RGB, so bit 0 (the least significant digit) needs to be the Blue value. You may need to change the direction the RGB LED is facing to make it easier to arrange the wires.

Set the inputs to 4 different binary values. Test your circuit by changing the value of the select signal and checking that the colour shown on the LED is as expected. Paste some images of your testing.

## Extensions - Build Multiplexers

To achieve the aim of showing that all components for a computer can be constructed from basic logic gates, you should have a go at building some multiplexers from scratch. Make sure that you test and save your circuits and add suitable images to this worksheet.

|  |  |  |  |
| --- | --- | --- | --- |
| D0 | D1 | S | X |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

1. Create a multiplexer from basic gates that selects (output X) one of two 1-bit data signals (D0 and D1) according to the value on a 1-bit select signal (S). The truth table is shown above. The output X matches D0 when S is 0 and matches D1 when S is 1.

Another way of explaining it is that the multiplexer will output 1 in the situation when  
 (D0 is 1 AND S is 0) OR (D1 is 1 AND S is 1), otherwise the multiplexer will output zero. You should be able to do this using AND, NAND, NOT, OR, NOR or XOR gates and it may help to start with the parts in brackets.

1. In the previous task, you created a multiplexer which had two 1-bit data inputs. Copy that circuit into a new one (in the same project) and use as a starting point to create a multiplexer that has two 4-bit data inputs and a 1-bit select input, this will be a 4-bit 2-to-1 MUX. You may need to use the technique of copying a 1-bit signal to a multiple bit signal from last week.
2. Make sure that your 4-bit 2-to-1 MUX from the previous task has labels for the inputs and output so that you can use it as a subcircuit. You should have used sub-circuits in last week’s lab.

Create a 4-bit, 4-to-1 MUX by using your 4-bit 2-to-1 MUX as subcircuits (you will need three of that previous MUX). Remember that a 4-to-1 MUX will need a 2-bit select signal. Make sure that you have labelled the inputs and outputs on your 4-to-1 MUX so that you can use it as a sub-circuit in the exercise below.

1. Create a 4-bit 8-to-1 MUX two different ways:
   1. Using two 4-to-1 MUXes and a 2-to1 MUX
   2. Using four 2-to-1 MUXes and a 4-to-1 MUX.