

# ECE 201 - Lab 6

## Multiplexers and Serial Communication

### PURPOSE

To familiarize students with the internal realization of multiplexers, and to show an application of multiplexers and demultiplexers for use in serial communications.

### EQUIPMENT

ECE 201 Lab Kit & *Digi-Trainer*

Simulation Software

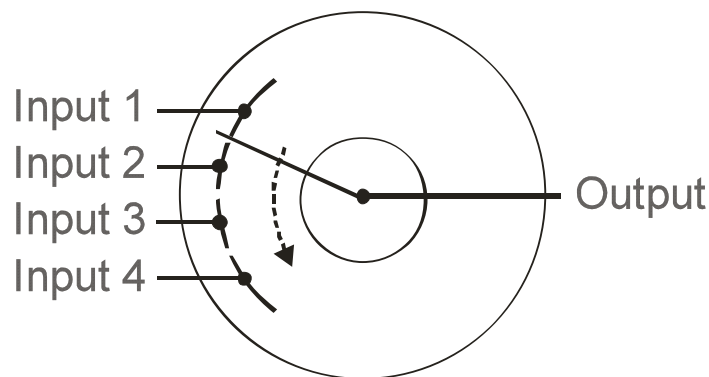
### REQUIREMENTS

- Circuit diagram with pin numbers labeled.
- Verbal description of the function of the final circuit.
- Simulation of functional seven-segment display circuit.

### PROCEDURE

#### Section 1 – The Realization of a 4-bit Multiplexer

A 4-to-1 multiplexer functions like a four-position switch such as the one shown below. The switch contact can be moved to any one of the four positions. The switch can not be moved anywhere else, i.e. it must be in contact with one of the four inputs at any time. The output signal of the rotary switch equals the signal on the input to which the switch rotor has been positioned.



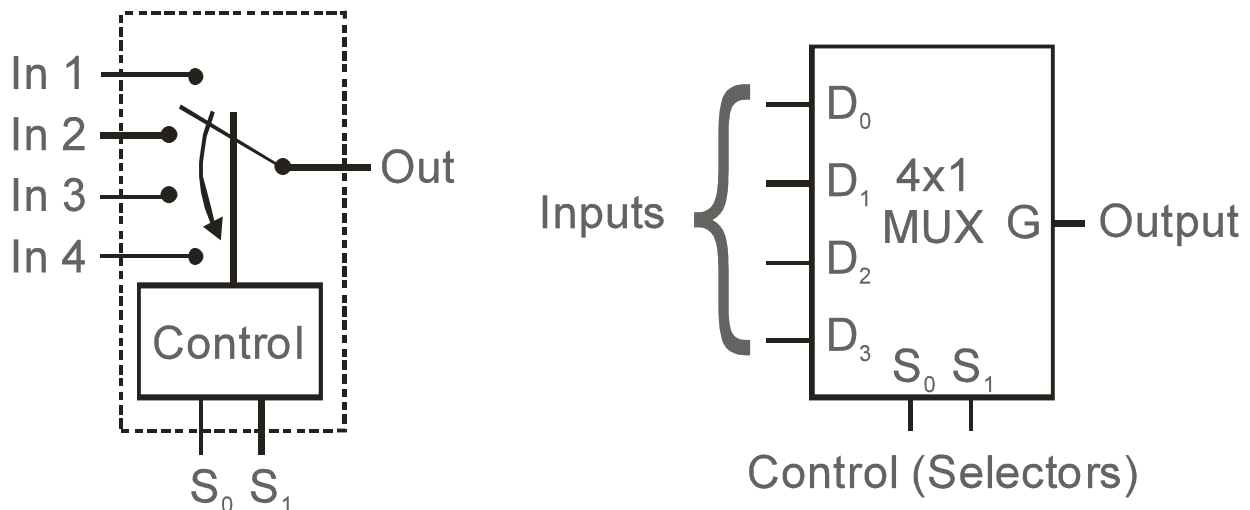
**Figure 1. Four-Position Rotary Switch.**

By now, you may be wondering what this has to do with multiplexers, which are little electronic gizmos having no actual switch contacts, no knob to turn or lever to position, or anything of the sort. It

is helpful to think of a multiplexer as a rotary switch whose position is controlled by a binary number input to the device.

How many control bits would we need to select one of sixteen positions?

The control inputs will henceforth be called *select* lines since they are used to select one (and only one) input to be routed to the output. We can redraw our rotary switch as follows:



**Figure 2. 4-to-1 Multiplexer.**

The selectors  $S_1$  and  $S_0$  determine which “position” the “switch” is in. Note that the *order* of  $S_1$  and  $S_0$  is important;  $S_1$  is the most significant bit whereas  $S_0$  is the least significant bit. If we want to be slightly more technical, we can formulate the Boolean equation for the output function of a four input multiplexer as:

$$\mathbf{f} = \mathbf{s}'_1\mathbf{s}'_0\mathbf{I}_0 + \mathbf{s}'_1\mathbf{s}_0\mathbf{I}_1 + \mathbf{s}_1\mathbf{s}'_0\mathbf{I}_2 + \mathbf{s}_1\mathbf{s}_0\mathbf{I}_3$$

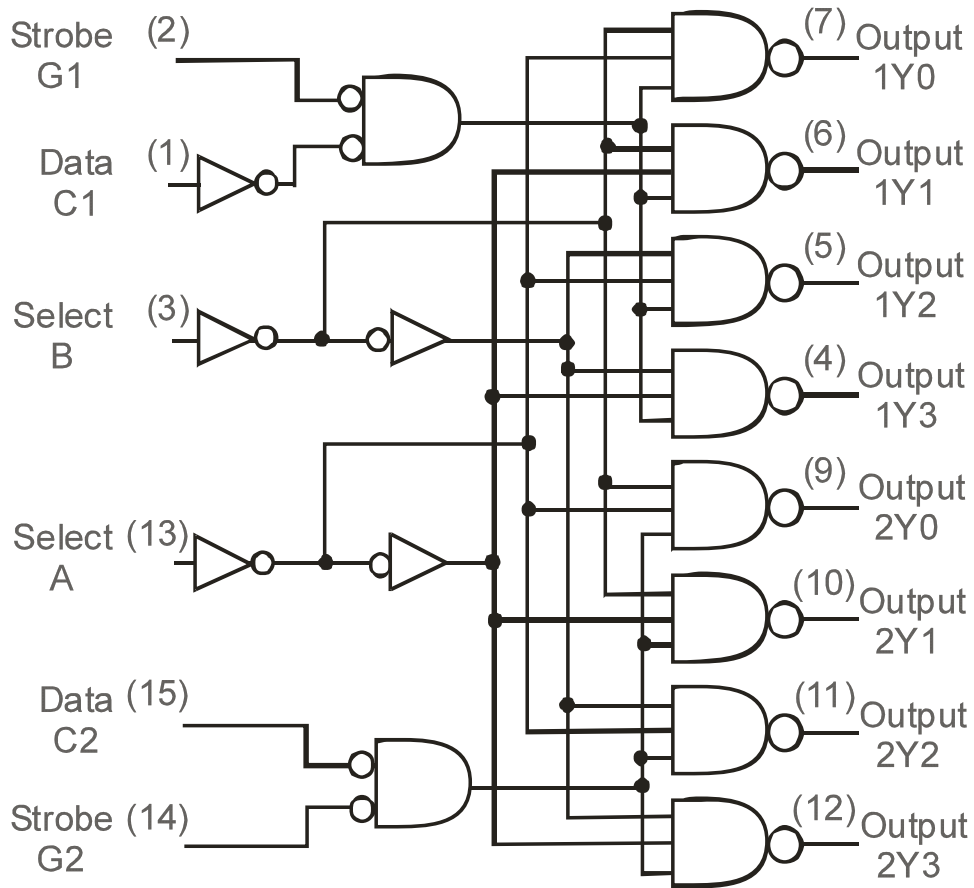
Note that when a binary zero (00) is on the select lines, the output  $\mathbf{f}$  is equal to  $\mathbf{I}_0$ , when a binary three (11) is on the select lines,  $\mathbf{f}$  equals  $\mathbf{I}_3$ , etc.

## Section 2 – Application - Serial Communication

Some notes before we get started: in this section, in addition to using multiplexers and demultiplexers, we will use a 74193 counter chip. You probably have not seen counters yet in your 201 lecture. Don't worry, you don't have to know how counters work in order to complete this lab. You just have to know what they do: they count (in binary). Connect the counter as shown at the end of this lab procedure,

and it will count repeatedly through the numbers 000 to 111 on the “Q” outputs, incrementing once on each clock cycle (toggle). (We'll discuss counter design after introducing sequential logic circuits.)

You'll also need to make use of your 74151 8-to-1 multiplexer chip, and your 74155 chip.

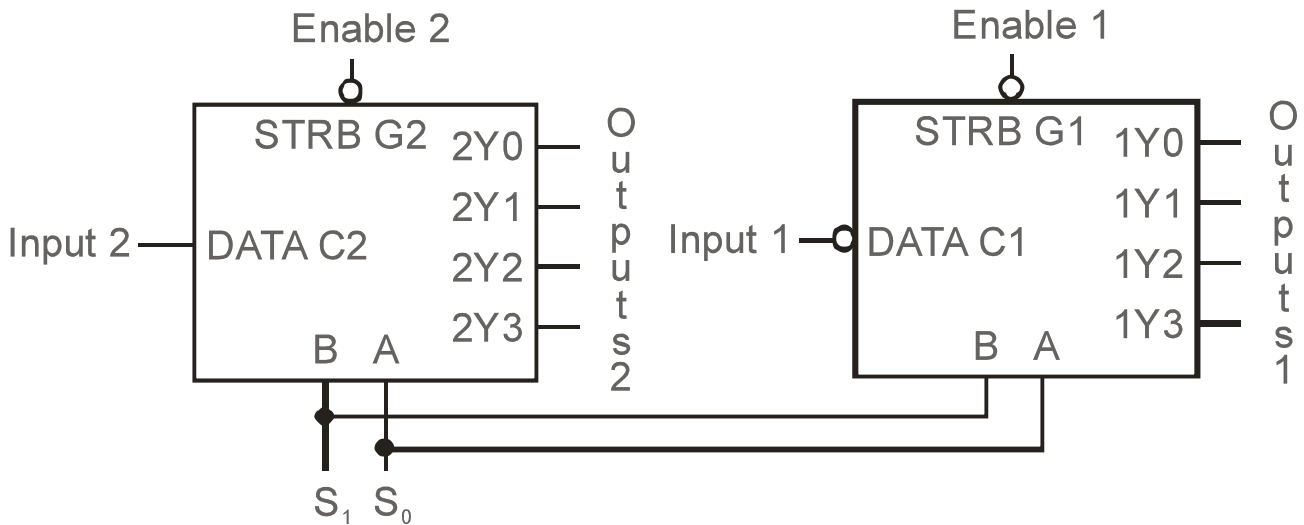


**Figure 3. 74155 Logic Diagram.**

Time multiplexing is often used with LED displays on calculators to reduce the amount of current the battery must supply to light the LED's. Rather than lighting all of the segments at once, one segment (or groups of segments for multiple digits) will be lit for a short time, perhaps 1 ms. Then the next segment (or group) will be lit for an equal time, and so forth until all the segments have been lit. Then the cycle repeats, making each segment of display flash on and off so quickly that the display appears continuous to the human eye, only slightly dimmer. The battery is then required to supply a much smaller average current than when all the segments are displayed continuously. Since LED's use a lot of current, around 10mA (all digits = 8), this can greatly prolong battery life.

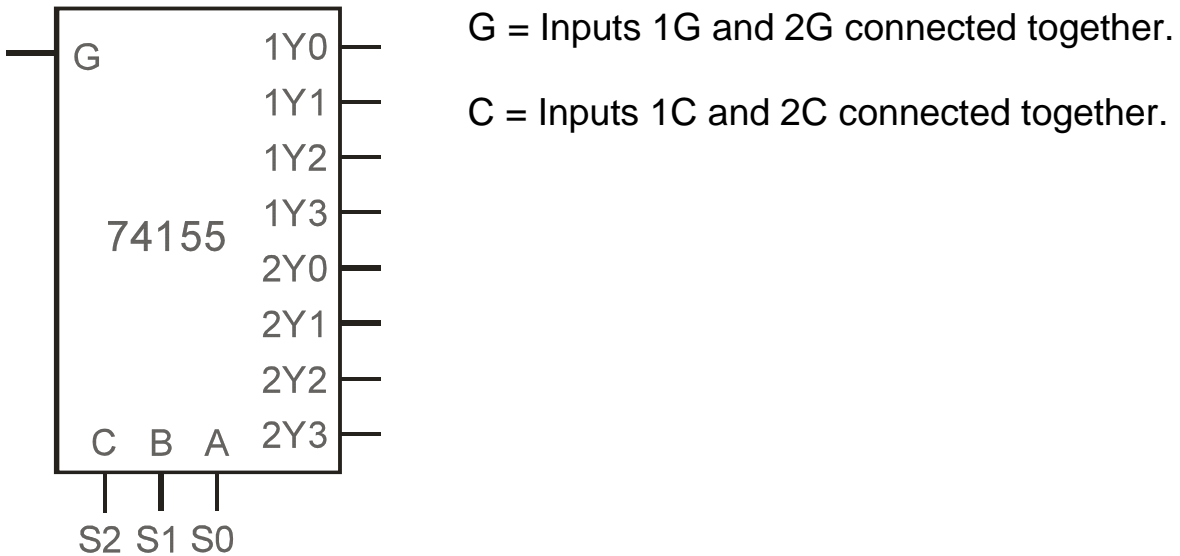
Time Multiplexing is also often used in communication systems where independent data streams must be sent over a single line or channel. The phone company does this on its lines.

We are going to time-multiplex the seven segments of a 7-segment display. The 74193 counter will be used again, where the three least significant bits driving both the Multiplexer and the Demultiplexer. The Demultiplexer is essentially a backwards multiplexer - one input and  $2^n$  outputs which are selected by  $n$  select lines. We will construct a 1-to-8 demultiplexer from the 74155 Dual 1-to-4 demultiplexer. The 74155 has two 1-to-4 demultiplexers, one of which has an inverting input (just to make life more difficult). A functional diagram of the 74155 is shown below.



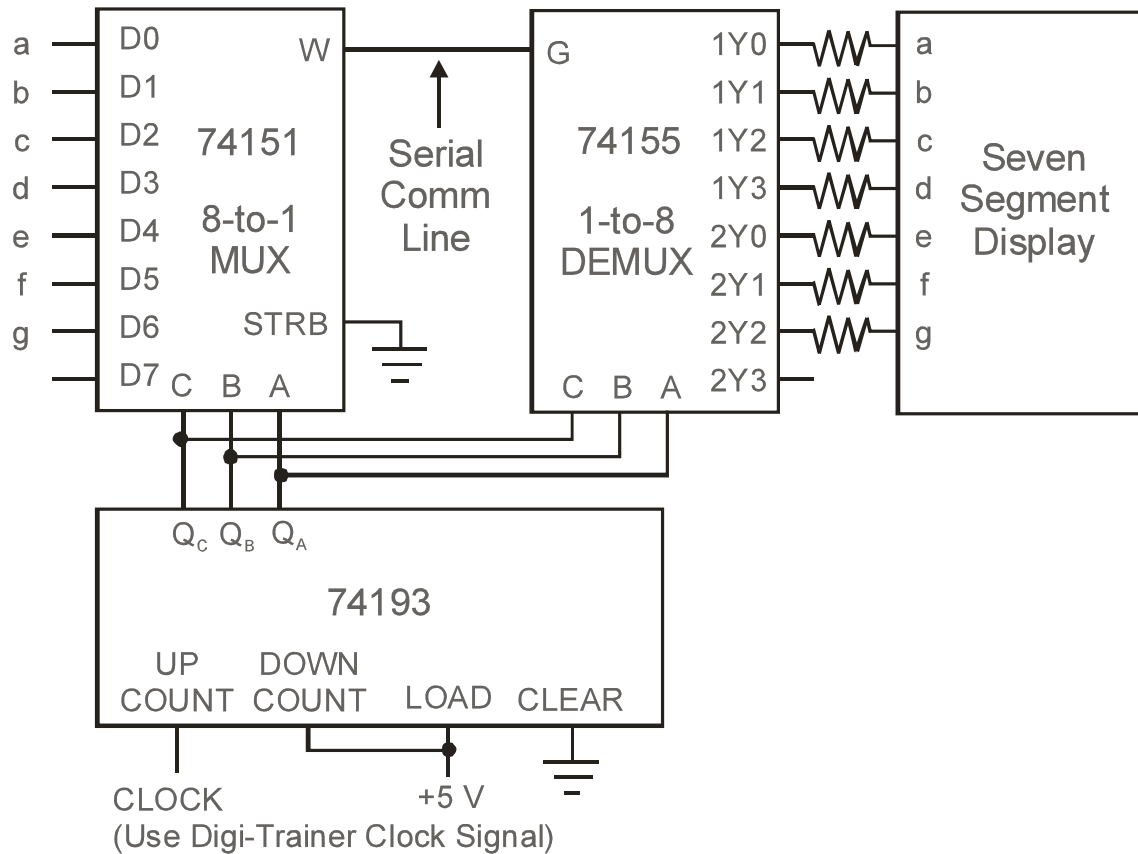
**Figure 4. 74155 Functional Diagram.**

To use as a 1-to-8 demultiplexer, connect as shown below:



**Figure 5. 74155 Used as 1-to-8 Demultiplexer.**

Now connect the following circuit:



**Figure 6. Parallel to Serial Communication.**

**! WARNING!** Do not wire any pin of the seven-segment display to ground. You can destroy it. Also be sure that the bare leads of the resistors do not touch before turning on power.

To operate: wire the inputs to the 74151 as appropriate to light the proper segments for the number “5”. Set the clock on 1 Hz and check that the proper segments light. Once all segments light (or remain dark) as they should—it will take 8 seconds for the number to be displayed—speed the clock up one setting at a time and observe the effect at each speed. At 1 kHz, the display will appear constant though a bit dimmer.

**?** Questions to turn in with the lab report.

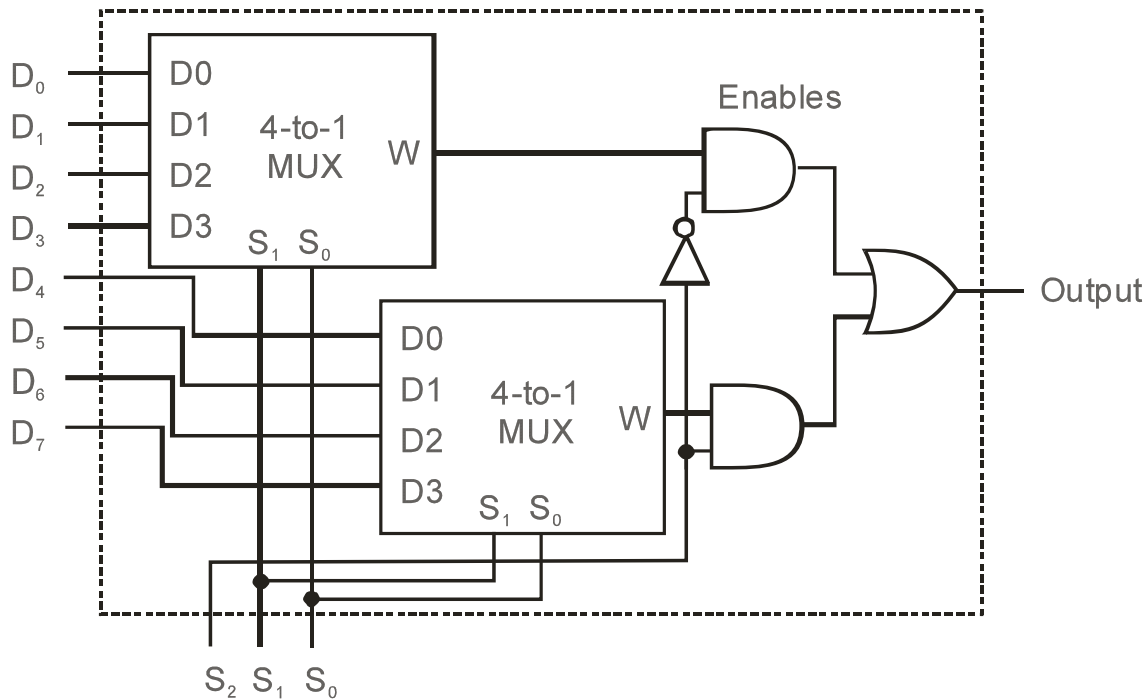
1. Explain why the circuit connection on page 2 acts like a 1:8 demultiplexer
2. If this circuit were being used to transmit data over a single wire, which connection on the final circuit (page 3) corresponds to the data wire? (Ignore timing problems with the counter.)
3. How do you think the phone company uses multiplexing to put many conversations over a single line?
4. What other types of multiplexing can you think of?

## Some notes on the simulation of this lab

Your simulation package does not have 74151 and 74193 chip models. Therefore, you will have to make them from smaller parts that do exist.

The 74151 MUX can be made in one of two ways. One approach would be to simply draw the AND-OR diagram of an 8-to-1 MUX. It's pretty simple, you just need eight 4-input AND's (each of which AND's the appropriate input line with the correct Minterm of the three select lines) and an 8-input OR function. Hint: Once you place a gate in Digital Works, you can right click on it to increase the number of inputs to three or four.

Another approach would be to take advantage of the 4-to-1 MUX you made for Section 1 of this lab. Two 4-to-1 MUXes can easily be connected to form an 8-to-1 MUX as shown below:



**Figure 7. 8-to-1MUX from two 4-to-1 MUXes.**

Digital Works does supply a 4-bit counter macro which could be used for the 74193, but none of the pins are labeled making it hard to use. Since we haven't covered counters yet, a counter macro is supplied to you on the lab web page.

You will also find a macro for an 8-to-1 DEMUX on the lab page. It is not exactly like the 74155 (which can also be used as two 4-to-1 DEMUXes), but it does have the low-active outputs for this lab.