Advanced CS 316 Logisim Toys

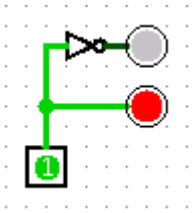
For the more adventerous students, we have put together a small collection of additoinal Logisim components that can let your MIPS processor do some nifty things, while showing off your hardware design and assembly programming skills. The following have been added to the cs316.jar library. Just download a fresh copy, or use the updated jar file all ready installed in the csuglab.

One great way to use these is to implement “memory mapped IO”, which is a common feature on actual computer hardware. Here, you can simply define addresses in a particular range (say 0x00000000 to 0x0000FFFF) to go to various devices, and addresses above this range to go to the actual RAM memory. So you might define writes to 0x000001000 to go to a special “output display” register, which can then display a number using LEDs. Or you can make it so that writing a data value to 0x00013355 actually writes a pixel to an LCD display at coordinates X=0x33, Y=0x55. Similarly, reads from a special address might read from a keyboard, or special input register, rather than regular memory.

Here are a bunch of custom devices that work great for memory mapped input and output:

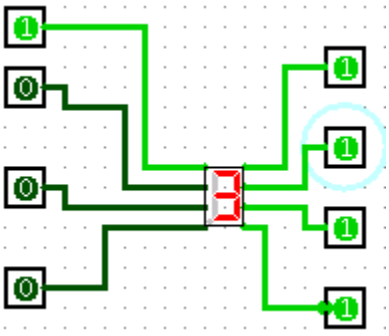
Simple LED

This is a simple colored light with one input, your choice of colors. Stateless.



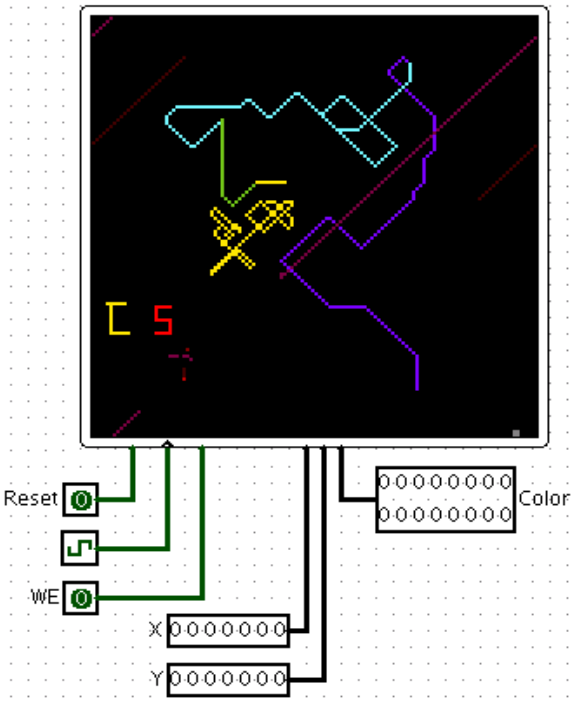
Seven Segment Display

This is a simple 7-segment display. The eight inputs control the seven segments and the decimal point. Your choice of colors. Stateless.



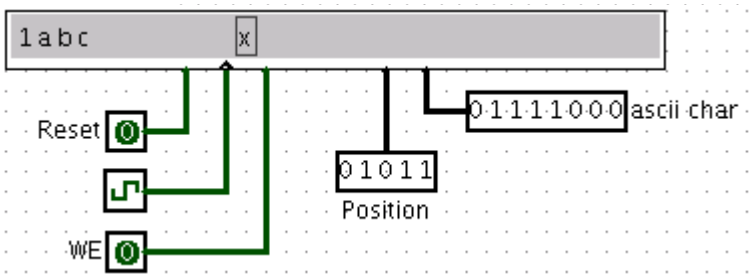
128x128 LCD Video Display

An LCD video output, with 16 bpp color depth. If WE is high, then on the rising edge of the clock it writes a pixel at (X, Y) of the specified color. The 16 bit color value is standard 5-5-5 RGB (1 unused bit, 5 red bits, 5 green bits, then 5 blue bits). Has built-in video memory to keep track of pixels already written. For kicks, you can combine this with the Joystick below, and a little circuitry (registers for the current X,Y position, adders for changing the cursor position, a ROM acceleration table, etc.). Voila, you have an instant digital 16bpp etch-a-sketch inside Logisim, and can make pictures like the one shown here.



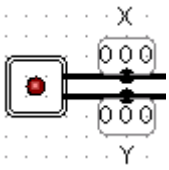
32 character LCD display

An LCD ascii display. If WE is high, then on the rising edge of the clock it writes an ascii character to the specified position (which you can see outlined by the cursor).



Joystick

A classic ATARI style joystick, with 3 bits of resolution in both axes. The two outputs give the joystick position as a 3 bit integer in the range -2 to +2 for both X and Y axes. To use the joystick, use the "poke tool" (the hand icon) to drag the red ball around.



Keyboard Queue

A simplified keyboard buffer. Use the "poke tool" to enter ascii data and newlines (and edit too, with arrows and backspace). The single output gives the ascii value of the leftmost character. If RE is high, then on the rising edge of the clock the leftmost character is popped off and the input queue is shifted left one character.