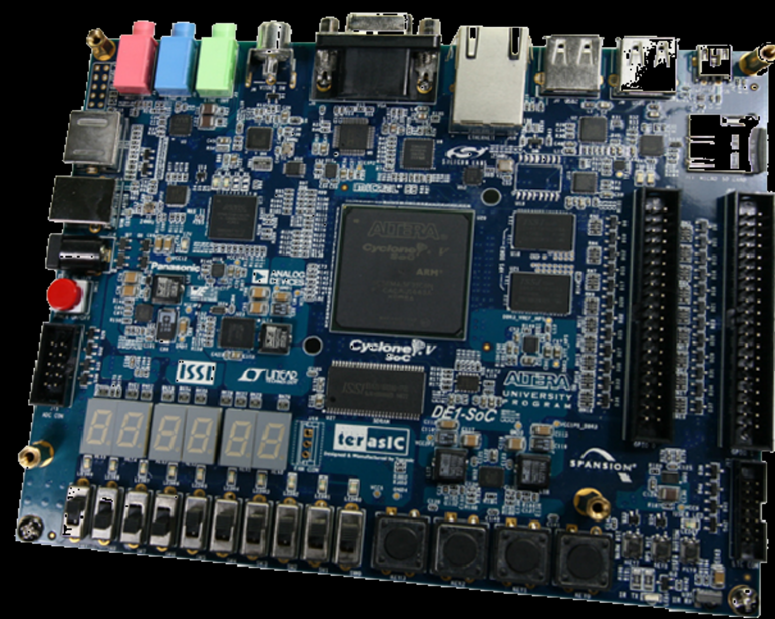
Lab 2 Preparation

Lab 2

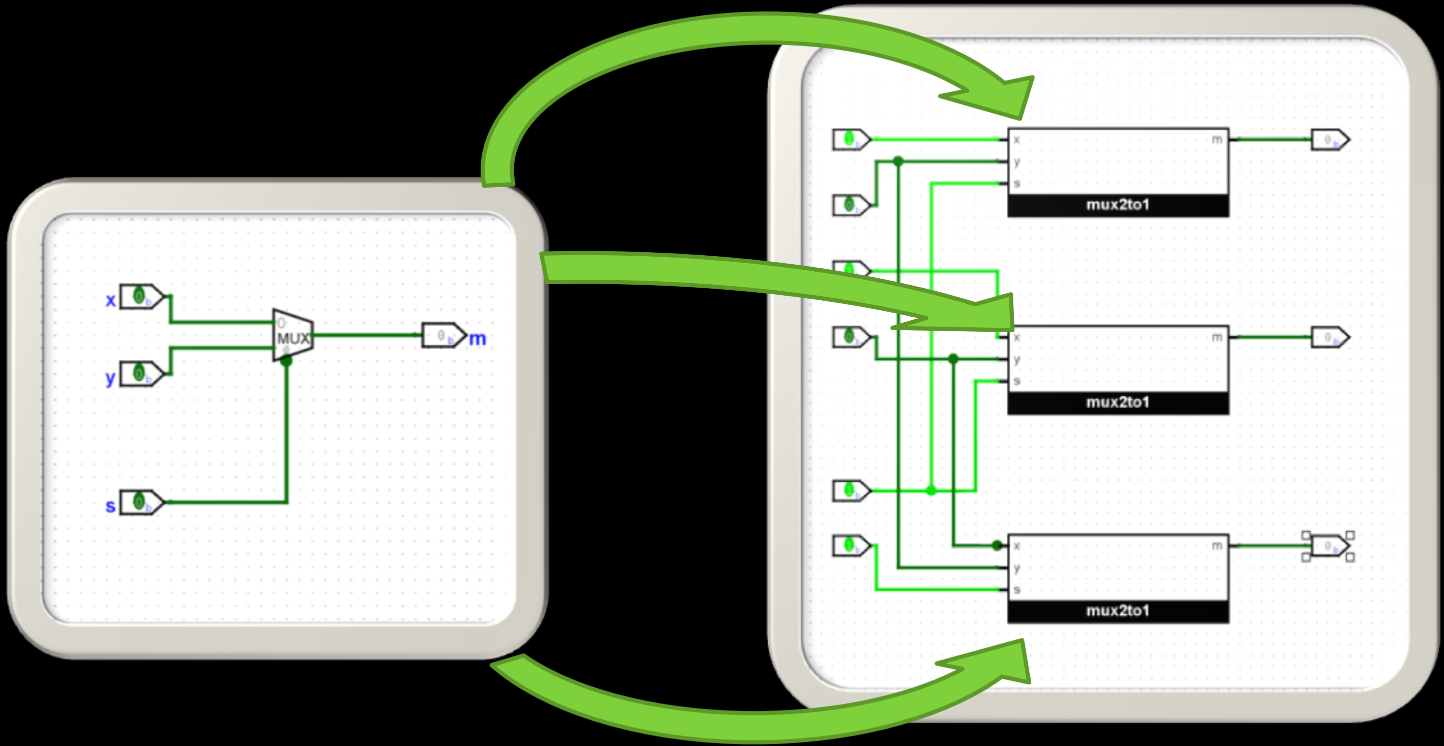
- Lab 2 topics:
  - Multiplexers (cont'd)
  - Design hierarchy
  - Decoders
    - 7-segment displays



- Intro to useful components in Logisim.

Tasks for Lab 2

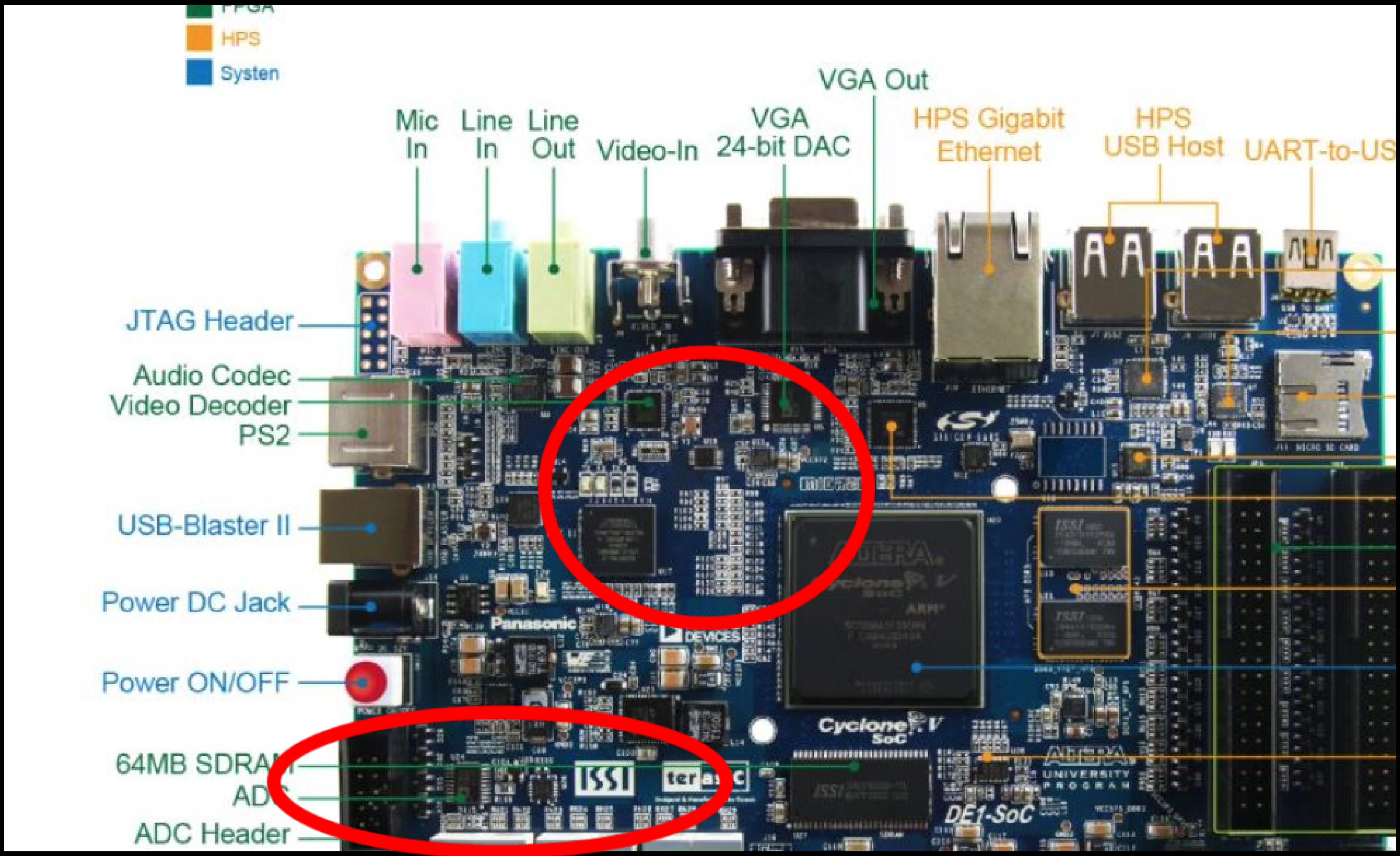
- Part I: Creating modules
  - Once created, a module can be used as a component.



Tasks for Lab 2

- Part I: Creating modules
  - Create a simple module that makes a wrapper for the mux circuit we provide.
  - Set inputs to buttons (labeled `SW0`, `SW1`, `SW2`) and output to LED (labeled `LEDR`)
    - Labels correspond to DE1-SOC inputs and outputs

Meet the DE1-SoC board!



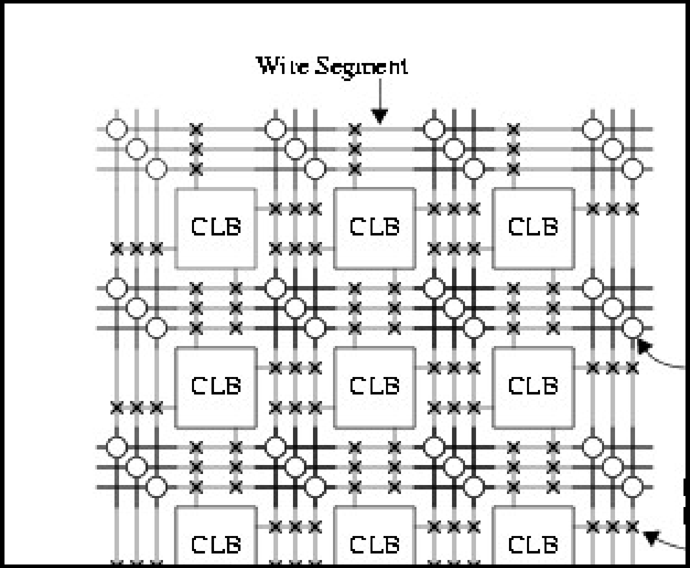
Meet the DE1-SoC board!

- What is the DE1-SOC?
  - It's a System On a Chip (SoC) with:
    - Altera's Cyclone® V 5CSEMA5F31 FPGA, and
    - a Dual-core ARM Cortex-A9 hard processor (HPS)
    - 64 MB SDRAM on FPGA device

    - Six 7-segment displays
    - 10 toggle switches
    - 10 LEDs
    - 9 green LEDs
    - Four pushbutton switches

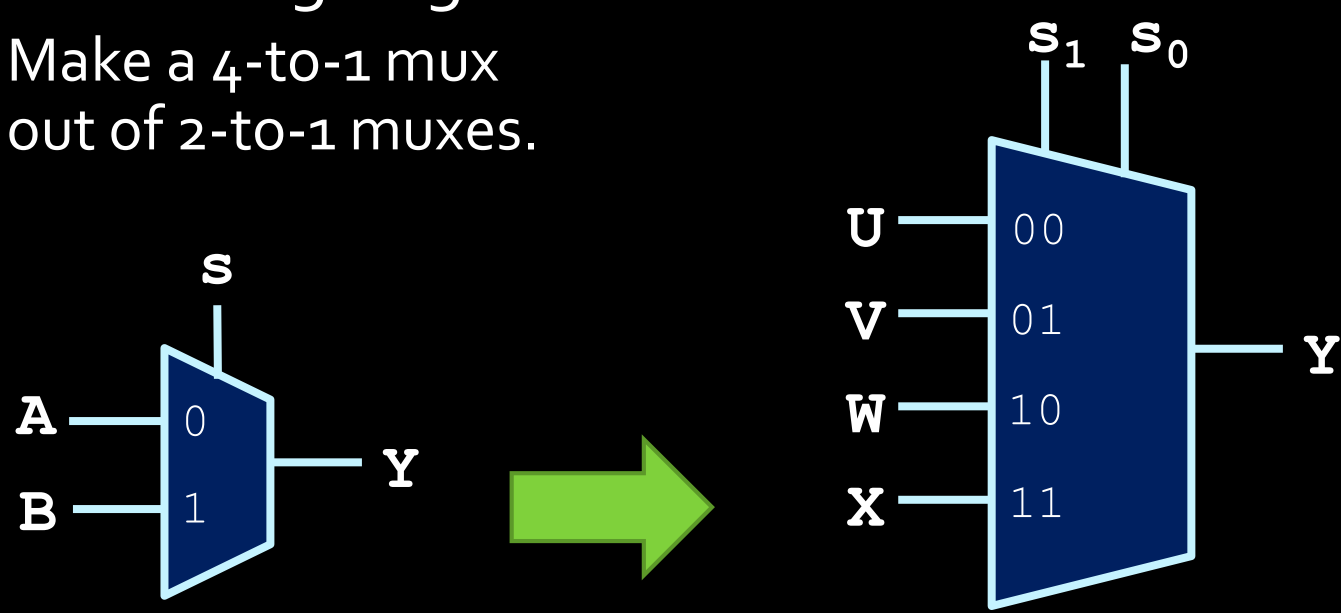
What does that mean?

- Key term: FPGA.
  - Stands for Field Programmable Gate Array.
  - A regular network of logic that can be programmed and reprogrammed to implement any circuit.
  - These type of circuits aren't generally built by hand; they're programmed using languages like Verilog or VHDL.



Tasks for Lab 2

- Part II: Designing with modules.
  - Make a 4-to-1 mux out of 2-to-1 muxes.

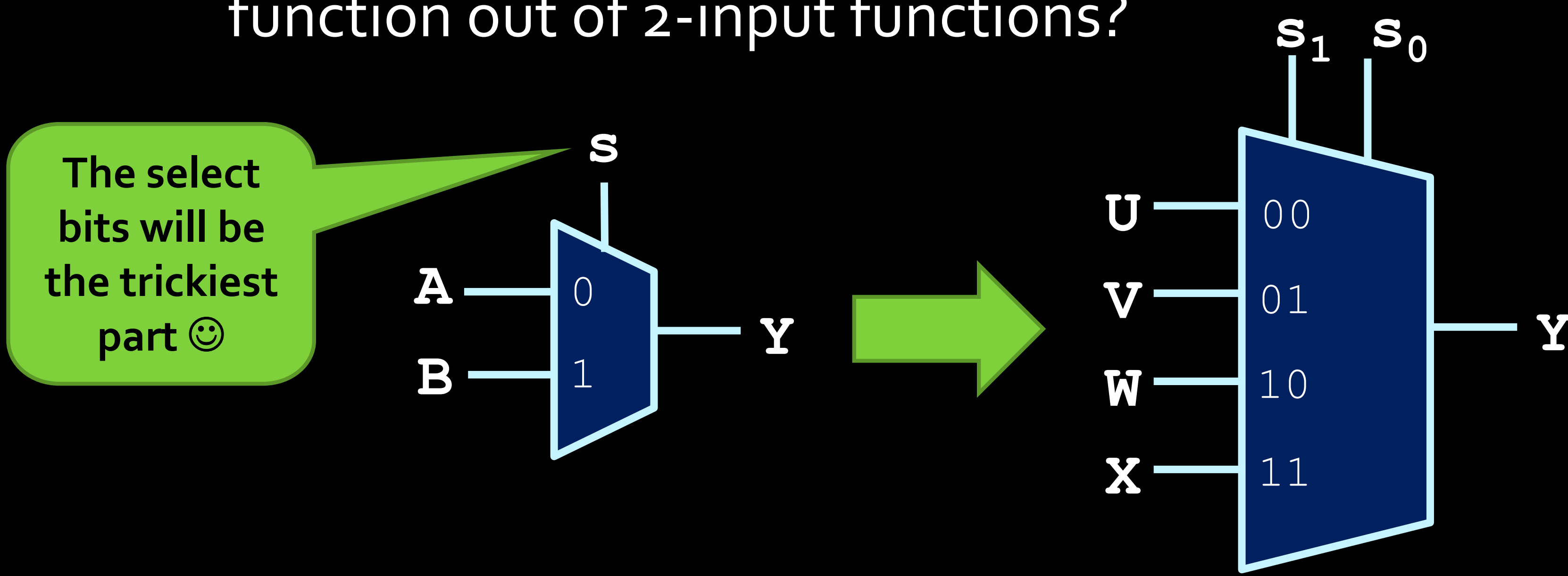


s_0	Y
0	A
1	B

s_1	s_0	Y
0	0	U
0	1	V
1	0	W
1	1	X

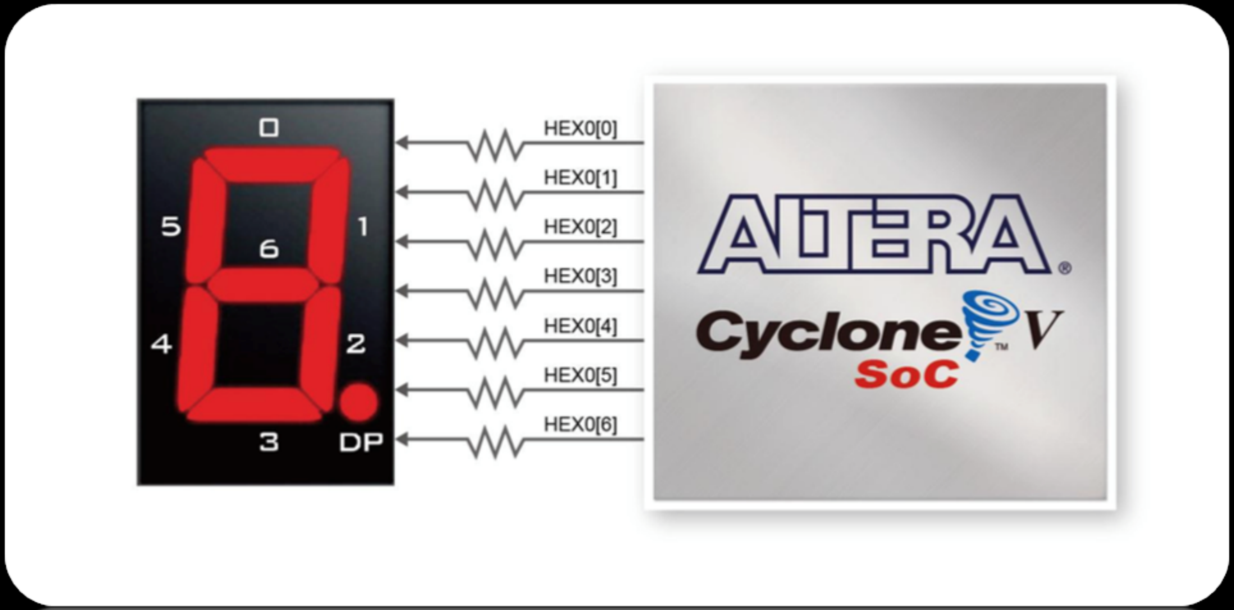
Tasks for Lab 2

- Part II: Designing with modules.
  - If each 2-to-1 mux can handle 2 inputs, how to build something that handles 4?
  - How would you make a 4-input function out of 2-input functions?



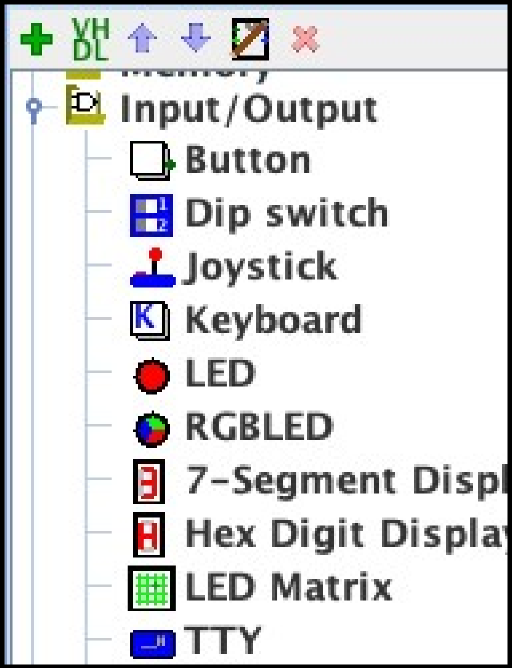
Tasks for Lab 2

- Part III: The 7-segment decoder.
  - This is one of the components in the Logisim toolkit.
  - Also one of the components on the DE1-SOC board!



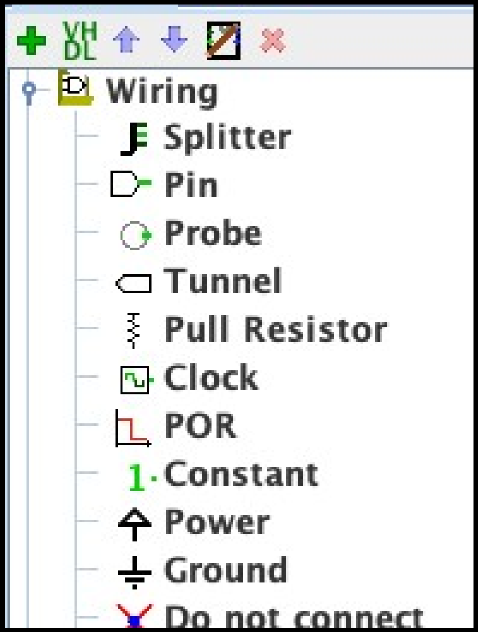
Exploring Logisim Components

- This components and others are listed under Input/Output, for future reference:
  - Button: Can be mapped to the switches and buttons on the DE1 board. Only outputs a 1 when held down with Poke.
  - 7-Segment Display: Can be mapped to the 7-segment display on the DE1 board.
  - LED: Can be mapped to the outputs on the DE1 board.
- Note: always start with the default input/output type from the tool bar and only switch to the above if necessary



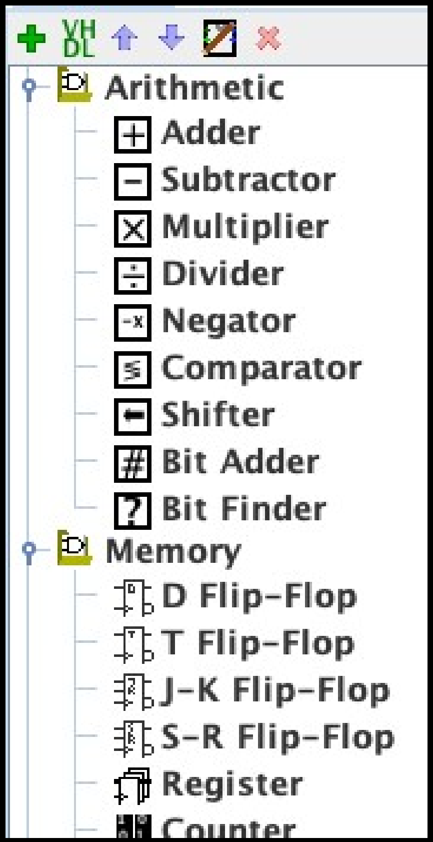
Useful Components in Logisim

- Wiring:
  - Splitter: Splits buses into individual wires or smaller buses. Works both ways.
  - Clock
  - Constant: Outputs a constant value (can be multiple bits on a bus).
  - Bit Extender: Pads or sign extends bits on a bus.
- You can even make a transistor circuit with the components at the bottom ☺

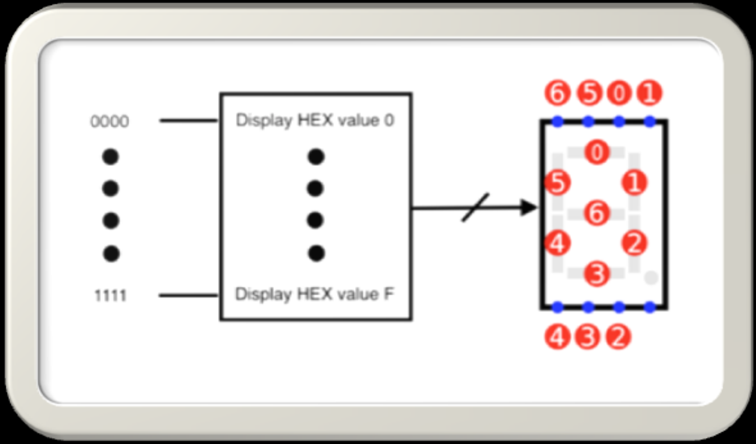


Useful Components in Logisim

- Arithmetic and memory:
  - Some of the arithmetic components will be useful in later labs. Details about each one can be found in http://www.cburch.com/logisim/docs/2.3.0/libs/arith/index.html
    - This doc is for an earlier version, some components may look different now.
  - We will be exploring the memory components later in the course.



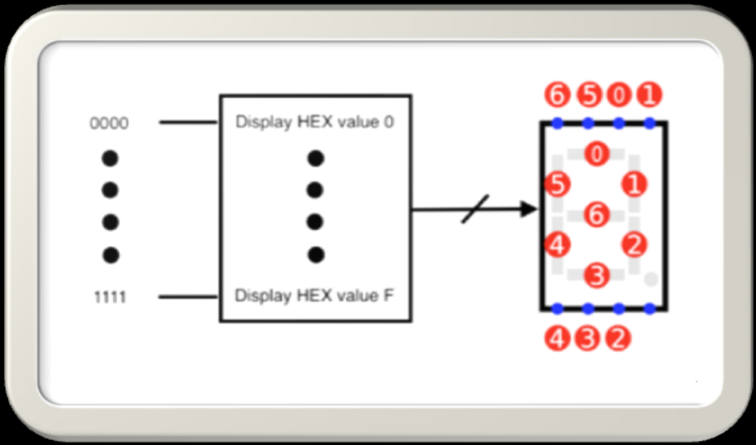
Tasks for Lab 2

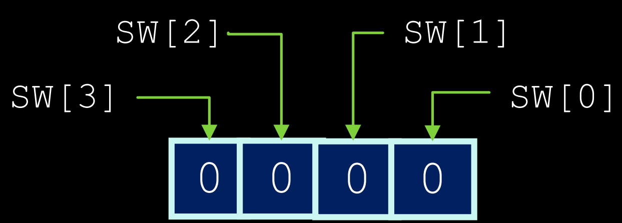


- The diagram on the right illustrates how to use the inputs to the 7-segment decoder (or the HEX decoder) to activate the segments.
- Each segment is active-high, meaning that if you set an input to 1, the corresponding segment will turn on.
  - Aside: DE1-SOC board is the opposite (i.e. active-low)

Tasks for Lab 2

- Ultimate goal:
  1. Take a 4-bit input coming from the switches on the and interpret those as binary number:





  2. Create seven circuits, one for each segment on the right to activate each segment based on the 4-bit input values.
     - For example: If input is `0000`, display "`0`" on the segments. If input is `1111`, display "`F`" on the segments.

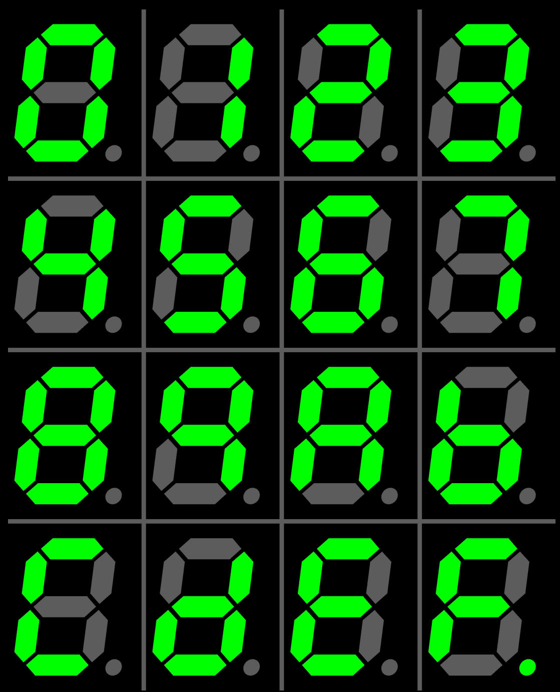
Activating 7-seg displays

- The diagram on the right illustrates the 16 digits we want to show on the 7-segment display.

- How do we make this happen?
  - Consider segment 0 (the top segment in each digit).
  - Need to set it high in the following input cases:

Input		Display
0000	--	"0"
0010	--	"2"
0011	--	"3"
0101	--	"5"
0110	--	"6"
0111	--	"7"
1000	--	"8"
1001	--	"9"
1010	--	"A"
1100	--	"C"
1110	--	"E"
1111	--	"F"

- How do we express this?



Activating 7-seg displays

- Answer: Karnaugh Maps!

	$\overline{SW1}*\overline{SW0}$	$\overline{SW1}*SW0$	$SW1*SW0$	$SW1*\overline{SW0}$
$\overline{SW3}*\overline{SW2}$				
$\overline{SW3}*SW2$				
$SW3*SW2$				
$SW3*\overline{SW2}$				

- If we can fill in these table values, we can figure out the circuit's behaviour.

Segment 0 truth table

SW3	SW2	SW1	SW0	HEX[0]
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

Segment 0 Karnaugh Map

- Now to fill in the table below….

	$\overline{SW1}*\overline{SW0}$	$\overline{SW1}*SW0$	$SW1*SW0$	$SW1*\overline{SW0}$
$\overline{SW3}*\overline{SW2}$	1	0	1	1
$\overline{SW3}*SW2$	0	1	1	1
$SW3*SW2$	1	0	1	1
$SW3*\overline{SW2}$	1	1	0	1

- What are the groupings that you see here?
  - Yes, overlapping is allowed ☺

Segment 0 Karnaugh Map

- What are the equations for these groups?

	$\overline{SW1}*\overline{SW0}$	$\overline{SW1}*SW0$	$SW1*SW0$	$SW1*\overline{SW0}$
$\overline{SW3}*\overline{SW2}$	1	0	1	1
$\overline{SW3}*SW2$	0	1	1	1
$SW3*SW2$	1	0	1	1
$SW3*\overline{SW2}$	1	1	0	1

SW2*SW1*SW0

SW2*SW1

SW3*SW1*SW0

SW3*SW1

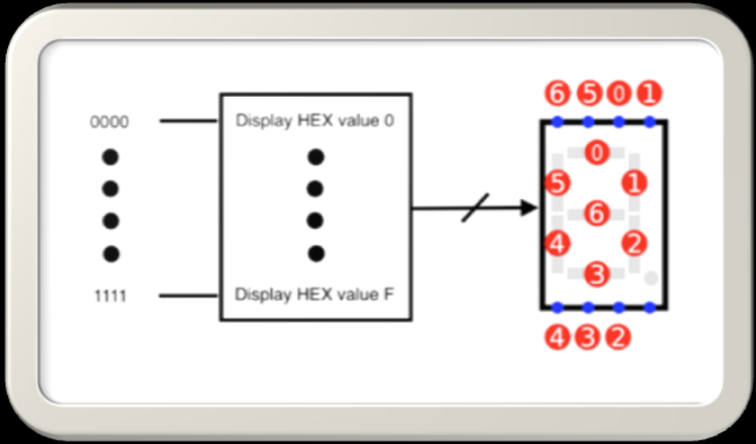
SW2*SW1*SW0

SW3*SW0

Can you figure out which terms are inverted to make these groups work?

Tasks for Lab 2

- Repeat this process seven times to implement the behaviour for each of the seven segments in the HEX display.
  - Try to get the minimal circuit for each!



- Once you're done, your seven circuits go into the decoder module in the middle of the diagram above.
  - Make sure to test each segment as you go!