CSEE 3827: Fundamentals of Computer Systems

Schematic Style Guide[1]

Schematics are not only design entry; they are documentation. In addition to a correctly operating circuit, you want people to understand your circuit relatively easily. Consistent adherence to good style rules helps with the latter. As with any style guide, some rules are strict while others require judgement. The recurring theme in all cases is clarity. If it is too overwhelming to produce a clear and working design all at once, get your design working first, then clean it up.

1 Layout

1.1 Maintain a left-to-right flow.

Keep inputs on the left outputs on the right. This prevents the spaghetti effect and keeps the schematic readable. One notable exception are feedback signals. By their very nature, they feed "back" from downstream to upstream, so they should be shown sending information opposite of the main flow. Use the vertical dimension strategically, for example for control signals, or to create a grid of otherwise convoluted logic.

1.2 Use direct connections, within reason.

Place gates and modules to reduce wire crossings and rats nests. Try to avoid long wires. If a direct connection is not possible or reasonable, use tunnels. Carefully deployed tunnels are preferable to a rats nest of wires.

1.3 Do not leave any inputs unconnected.

Use constant 1 or 0 instead, as unconnected inputs are ambiguous. The standard Logisim modules are exceptions to this rule, as they often have many configuration inputs, and connecting them all would crowd the schematic.

1.4 Avoid excessive hierarchy.

Submodules are indirections, which if unnecessary simply hurt readability. However, in certain cases the submodule helps clarity, for example when instantiating multiple instances of the same sub-circuit or when forgoing submodules would induce overload from too much circuitry crammed together at once.

2 Naming

2.1 Give all submodules descriptive and visible names.

Keep the Circuit Name and Shared Label consistent, so that block labels in the schematic appear in the list of circuits in the Logisim project.

2.2 Give all pins and ports clear, descriptive names.

The reader should be able to tell what a wires function is by its name. For example, use MEM_FLASH_A15 instead of just A15. This way, you know its address bit-15 of the flash memory.

[1] This style guide adapted from prior classes, https://blog.upverter.com/2014/05/28/schematic-style-guide/, and http://bec-systems.com/site/1116/schematics-style.

2.3 Keep names short.

Just because your software lets you enter 32 or 64 character net names, doesn't mean you should. Again, the goal is clarity. No names is no information, but lots of long names are clutter, which then decreases clarity.

2.4 Use standard names for standard pins.

For example, use CLK for clock, EN for enable, SEL for select, etc. See this ANSI/IEEE standard for a full list.

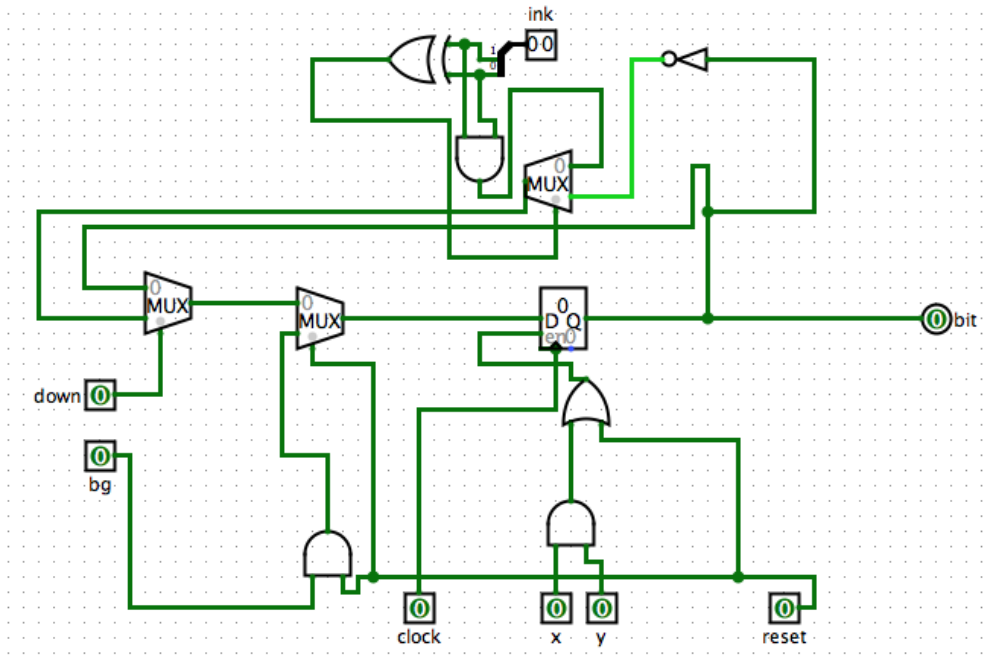
2.5 Use uppercase for all pin and port names.

2.6 Use notes judiciously.

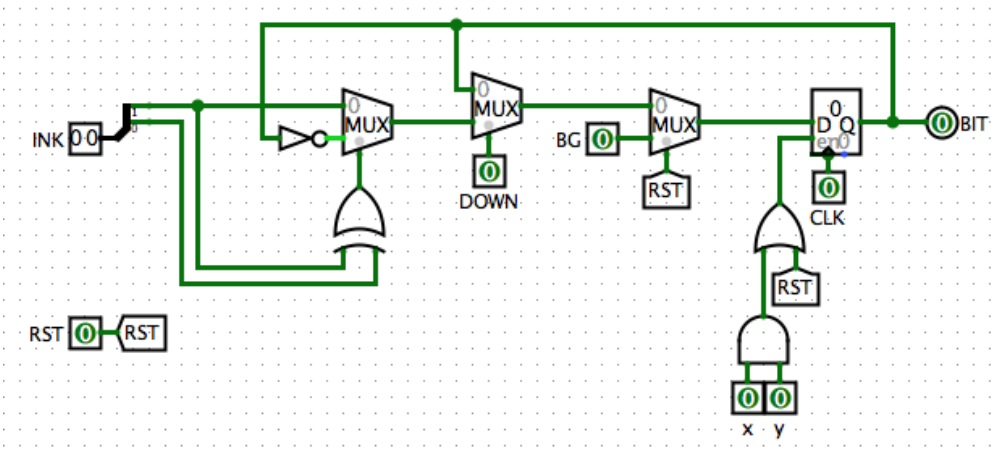
Like code comments, brief text notes can help explain non-obvious features of the design.

3 Examples

Before #1

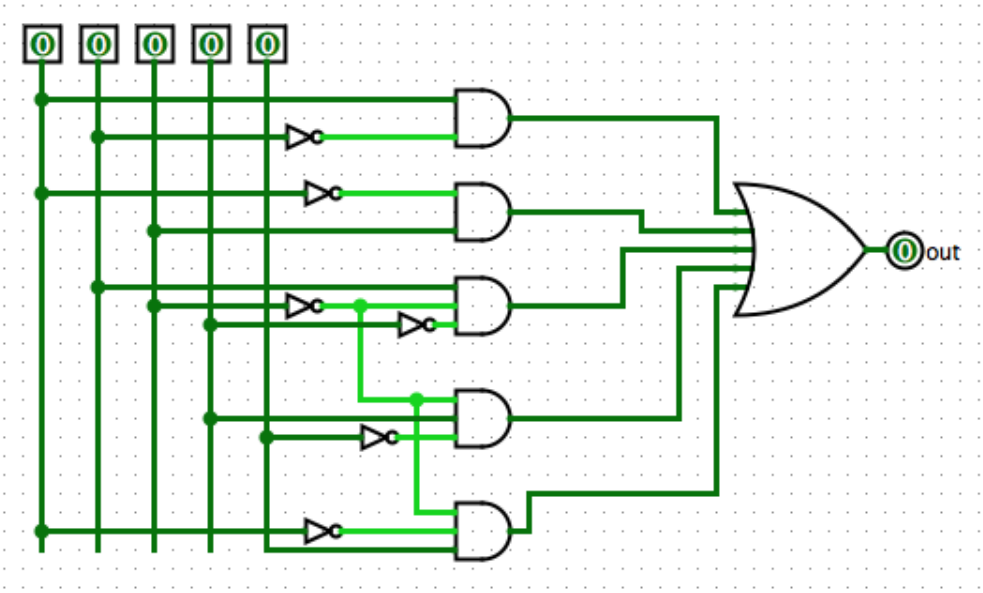


After #1

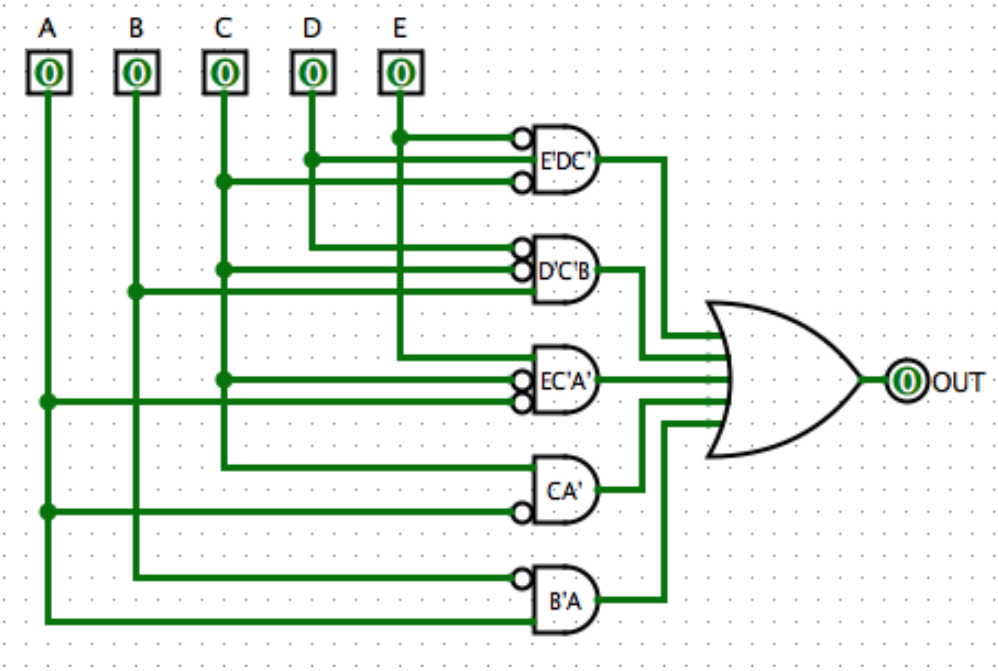


- Located single-destination pins near their destination thereby shortening wires
- All logic flows left to right
- Removed two logically unnecessary gates
- Used vertical dimension to differentiate control signals
- Used tunnel for standard signals that go everywhere (RST and CLK)
- Capitalized pin names

Before #2



After #2



- Labeled inputs
- Used inverted gate inputs to reduce inverter-spaghetti
- Reordered product terms to minimize wire crossings
- Labeled AND gates with product terms as logic somewhat complex
- Centered OR gate
- Left inputs at top to create regular grid structure. Technically a violation of left-to-right flow, but judged that this was cleaner for this logic.