ADVANCED FEATURES

Component Delays and Families

LogicSim 2.8 specifies the propagation delays for a device by storing 6 real values for each pin :

- minimum, mean and maximum rise time
- minimum, mean and maximum fall time

A selected component is assigned a set of delays with Component Delays… from the Circuit menu which gives a choice of a family or none, the latter allows delays for each pin to be changed on an individual basis – the values set must satisfy 0<=Min<=Mean<=Max.   If more than one component of the same type has been selected, for example by holding down the âŒ˜ key while clicking, the delays are applied to all of them.   Copies made of a component will inherit its family or delays.

A family and its associated delays are defined with Create new family… from the Circuit menu.   Components are made members of a family   with the Component Delays… dialogue.   Components start life as a member of the Default Delays family which, as distributed, has all its delays zero.   LogicSim families must be stored in the Families folder inside the LogicSim folder.

A simulation session runs in one of four possible delay modes: minimum, mean, maximum, random.   In random mode the delays are distributed between the specified minimum and maximum values.   A logic(X->0) transition uses the   logic(1->0) delays, logic(X->1)uses the logic(0->1) delays.

Input and output pins react to delays in slightly different ways.   In the case of an output pin, after an internal change the algorithm 'waits' the specified time before updating the node that is attached to that pin.   For an input pin, after a node changes, the algorithm waits the specified time before updating the internal value (and then proceeding with the internal simulation).   However, if the input node changes its value again before the internal value is updated, then the previous change is discarded, this mechanism acts like a low-pass filter. Devices like monostables can often be simulated by a single gate just by setting appropriate delays.

Extra Text and Pictures

Picture and text objects created with a program like ClarisDraw can be added to a circuit by pasting them in from the clipboard.

Drop-Replace Editing

Deleting a component normally removes its attached wiring. However, if a replacement is dropped on top of a component the connections will be remade automatically. This feature works well in simple cases, a two input NOR gate being replaced by a two input NAND gate for instance, but in more complicated cases check that the wiring is really what you wanted.

Simulation Speed

A LogicSim simulation cannot run as fast as a real circuit so the user can set the "Simulator clock rate" which is measured in hertz (Hz).   This is the number of simulation time units that happen in one (real world) second.   In fact the number specified by the user is a target – the computer may be too slow to achieve it, in which case, the simulation will run as fast as it can (which will be affected by other applications running in the background).

The Signals Window

This window is an oscilloscope display of the signal levels present at various points in the circuit.   Clicking with the mouse pointer inside this window creates a vertical cursor which can be dragged and used to identify the precise time and sequence of events.   The two buttons on the bottom left stretch or compress the time axis with the cursor acting as an anchor point.   If the simulation becomes stable the recording halts, to prevent long periods of inactivity overwriting more interesting signals, but the time axis can still be advanced manually with the arrow button.   This increases time by the number of steps indicated in the adjacent editable text zone.   This is particularly useful if you are trying to create a picture showing what happens when the inputs to a stable circuit are changed.   Whether or not   clicking a switch and or button on the circuit advances the signals record can be set on the Simulation Options panel.

Customisation and Modules

LogicSim creates the pallet of standard components by searching the folder "Default components"   (or a folder with this alias) which must be in the same folder as the LogicSim application.   New components or modules can be put in the "Default components" folder (or one of its sub-folders), and rarely-used ones removed, to customise the pallet.   Components are loaded in folder and file name-order, allowing control over how components are arranged on the display.   Loading large numbers of components may take a few seconds when a new document is opened and can be stopped with âŒ˜-. or by typing the esc key.

From a technical point of view are three types of component files: standard components, modules created from a LogicSim circuit, and code resources compiled from Pascal or C source using the software development kit (SDK). However, although each of these types has its own distinct icon they are all used, and can be loaded, in exactly the same ways.

Any LogicSim circuit can be stored as a "module" and then used as a building block in more complicated circuits.   The example circuit "3-IP AND cct" performs a 3-input AND operation. Create Module… from the Circuit menu brings up a preview showing what the module will look like;   the names of the input switches/buttons and output LEDs are used as pin labels. If the default appearance is acceptable the module can be saved immediately with the

Save… button. The appearance of the module can be edited with a drawing package using the Copy and Paste buttons to transfer the PICT image via the clipboard. The the blue and red circles in the Module dialogue indicate the positions of the input and output terminals respectively and can be repostioned by dragging. The finished module is "3-IP AND" in the Examples folder. It is not easy to get good results with some drawing packages – start with a 1pt pen and a 1dpi grid and experiment.

A module can include other modules in its defining circuit and these nested modules will be automatically found and loaded with the main module. However, take care editing a module "B" that is nested inside another module "A" because if the number of inputs or outputs of module B is changed and then module A is loaded and used in a simulation it is possible for LogicSim to quit unexpectedly with a Type 1 error. Give modules unique names to avoid an unintended module, that just happens to have the right name, being found first and loaded. The Info window shows the progress of automatic loading process which searches down from the LogicSim application directory.

If you have Finder 7.1.3 or later you will find that modules can be loaded by dragging straight from the Finder as well as with the Circuit menu item. Parts can also be dragged from the circuit window onto the Finder or other a suitably drag-friendly application for storage. A component that is dragged onto the Trash is deleted.

If you are areally serious user you will want to write your own modules in C or Pascal, the details of how to do this are in the SDK folder distributed with LogicSim.

The Stimuli Window

Switches and clocks can be programmed to change states automatically at preset times by listing the sequence and times of actions as commands in the Stimuli window. The command syntax is:

    <ComponentName>: <TimeSpec1> <State1> <TimeSpec2> <State2> …

Syntax errors are reported in the Info window. Anything following // on a line is treated as a comment. In LogicSim 2.8 the allowed values for the states are: for the switch output L0,L1,LX,LZ; for the Clock ON,OFF. Unsigned integer TimeSpecs are treated as absolute times in simulation cycles after the Execute button has been pressed. TimeSpecs preceded by a '+' sign are interpreted as the number of simulation cycles to wait after the previous command until the next. For example, given the circuit ou wish to start the clock at Time=1000 and stop it 500 cycles later, and also to set the switch to logic(0) at Time=0, to logic(1) at Time=500, and return to logic(0) at Time=750. The following lines need to be typed into the Stimuli Window:

```
foo: 1000 ON +500 OFF  //Hello World!
bar: 0 L0 500 L1 750 L0
```

or, equivalently

```
foo: 1000 ON 1500 OFF
//Hello Mum!
bar: 0 L0 +500 L1 +250 L0
```

The commands are listed in the Info window as they are executed.