User Manual

Circuit Sandbox v0.3

Table of Contents

User Interface

1. Selector
2. Panner
3. Interactor
4. Eraser
5. Pencils
6. New
7. Open
8. Save
9. Start/Pause
10. Reset
11. Step
12. Set speed
13. Undo
14. Redo
15. Info bar

Play Area

The play area displays your circuit. You can edit the circuit by interacting with the play area. There is no hard limit (apart from your computer hardware) to the size of the circuit.

The colour of each circuit element on the play area depends on a number of factors, including the type of element and whether the element is electrified (see Simulation Rules).

Navigation

The circuit can be zoomed (mouse wheel) or panned (using the Panner tool). Double-clicking with the panner will center the camera on the clicked element.

Edit History

You can undo (Ctrl-Z) or redo (Ctrl-Y) changes made to the circuit at any time.

Toolbox

The Toolbox contains *tools* for editing the circuit in the play area. Tools can be *bound* to any mouse button by clicking on the tool name with the desired mouse button. Clicking or dragging that mouse button in the play area invokes the bound tool.

Circuit Sandbox supports mice with up to five buttons, and treats touchscreen input as a sixth button. A colored box enclosing the name of the tool indicates the mouse button it is bound to:

- Red - Left mouse button
- Blue - Right mouse button
- Green - Middle mouse button
- Cyan - Fourth mouse button (X1)
- Magenta - Fifth mouse button (X2)
- Yellow - Touchscreen input

Selector

The Selector is used to select parts of the circuit for batch operations like moving or copying. For a complete description of selection, see Selection Mode.

Panner

Dragging the Panner pans the circuit. For larger circuits, it may be useful to bind this tool to a secondary mouse button.

Interactor

Certain elements are able receive input. This tool is used to interact with such elements.

Pencils

Elements are the basic building blocks of the circuit. Pencils are used for drawing elements on the circuit. To learn how to use pencils, see Drawing.

Eraser

The eraser is used for removing elements from the circuit. The eraser also behaves like a pencil.

Button Bar

The button bar contains various buttons to control the simulator and save and load circuits:

- **New** (Ctrl-N): Start a new instance of Circuit Sandbox
- **Open** (Ctrl-O): Open an existing circuit file in a new instance of Circuit Sandbox
- **Save** (Ctrl-S): Save the current circuit in the play area to your computer (Holding Shift while clicking this button opens the "Save As" dialog)
- **Toggle simulator state** (Space): Start or pause the simulator
- **Step simulator** (Right arrow): Run the simulator for a single step
- **Reset simulator** (R): Reset the simulator to its initial state
- **Set simulator speed** (Ctrl-Space): Set the FPS of the simulator

Info Bar

The info bar displays useful information about what the cursor is pointing at. When hovering over an element in the play area, it displays its name and other relevant state.

The Simulator

The simulator is initially paused when Circuit Sandbox is launched, and may be started using the Toggle Simulator State button. The circuit may be modified even while the simulator is running and changes made update the simulator immediately.

Circuit Sandbox simulates the circuit in discrete steps, according to predefined rules (see Simulation Rules). The rate at which the circuit is simulated may be adjusted from the default of 5 FPS (frames per second).

When paused, a single step can be simulated using the Step Simulator button. This may be useful for debugging the circuit.

Simulation Rules

There are four kinds of *materials*, and at every step, each element acts as a certain material.

The materials are characterised by the way they conduct electricity across their four sides, and whether they produce electricity. These are the four kinds of materials:

- Insulator: Does not conduct electricity at all
- Insulated Wire: Conducts electricity between opposite sides of the material; does not conduct electricity between adjacent sides
- Conductive Wire: Conducts electricity across all sides
- Source: Produces electricity on all sides

A element or material is *electrified* if electricity can be conducted from a Source to it. Most electrified materials are displayed in a brighter colour in the play area.

The elements act as materials according to the following rules:

- Insulated Wire, Conductive Wire, and Source always act as their respective materials.
- Relays, logic gates, and communicators act as materials depending on the the number of adjacent Signal elements and whether those Signal elements were electrified at the end of the previous step.
- The Signal element acts as a Conductive Wire to Insulated Wire, Conductive Wire, and Signal elements, but acts as an Insulator to all other elements.

Signal is a special type of element that is used to indicate the inputs of relays, logic gates, and communicators.

Relays

Relays act as Conductive Wire when *on*, and Insulator when *off*.

- Positive Relay: *on* if at least one adjacent Signal was electrified in the previous step, and *off* otherwise.
- Negative Relay: *on* if at least one adjacent Signal was not electrified in the previous step, and *off* otherwise

Logic Gates

Logic gates act as Source when *on*, and Conductive Wire when *off*.

- AND Gate: *on* if all adjacent Signals were electrified in the previous step, and *off* otherwise.
- OR Gate: *on* if at least one adjacent Signal was electrified in the previous step, and *off* otherwise
- NAND Gate: *off* if all adjacent Signals were electrified in the previous step, and *on* otherwise.
- NOR Gate: *off* if at least one adjacent Signal was electrified in the previous step, and *on* otherwise.

Communicators

Communicators are the means for the circuit to communicate with the outside world. Each communicator is an endpoint capable of simultaneous bidirectional communication. Communicators transmit and receive data at every step of the simulation, and it is not possible to pause transmitting or receiving of information while the simulation is running. Certain types of communicators have built-in protocols that have a way of specifying a "no-op", effectively pausing communication.

Transmitting and Receiving Information

The design of the transmission and reception behaviour of communicators are intended such that any connected group of communicators (of the same element type) will act as if they are a single communicator.

Receiving Information

Communicators act as Source when *on*, and act as Conductive Wire when *off*. This is similar to the behaviour of logic gates. Whether a communicator is *on* or *off* depends on external inputs and the type of communicator.

Transmitting Information

Communicators transmit an *on* signal if they are adjacent to at least one Signal element that was electrified at the end of the previous step, or connected via elements of the same type to such a communicator. Otherwise, they transmit an *off* signal.

Types of Communicators

There are three types of communicators in Circuit Sandbox: screen, file input, and file output. They differ by where they receive information from and transmit information to.

Screen Communicator

Screen Communicators are used to receive input from the mouse and display output to the screen. The receive an on signal when the mouse is being held down over the element using the Interactor tool, and receive an off signal otherwise. The display brightness of the element is determined by the signal being transmitted.

File Input Communicator

File Input Communicators are used to read data from files. The input file is bound using the Interactor tool. Clicking on the communicator with the Interactor tool will display a file dialog to select the input file. For security reasons, it is not possible for the circuit to dynamically specify a file to open.

Please read Appendix A for the file input protocol.

File Output Communicator

File Output Communicators are used to write data to files. Similarly, the output file is bound using the Interactor tool, and it is not possible for the circuit to dynamically specify a file to write to.

Please read Appendix A for the file output protocol.

Best practices

Although the behavior of relays, logic gates, and communicators are well-defined with any number of adjacent Signals, we recommend the following for clarity:

- Exactly one Signal element should be adjacent to each relay
- At least one Signal element should be adjacent to each logic gate
- Not more than one Signal element should be adjacent to each communicator

Saving and Loading Circuits

Circuits designed with Circuit Sandbox may be saved to disk just like any other files, and they use the `.ccsb` file extension. If Circuit Sandbox is set as the default program to open `.ccsb` files, double-clicking on a save file will open it in Circuit Sandbox.

Platform Support

Circuit Sandbox has been tested to work on Windows and Linux. All display resolutions are supported including HiDPI displays.

Editing Features

The editor in Circuit Sandbox is designed to be efficient to use when mastered.

Drawing

The pencil tools and eraser use the same drawing mechanics.

Points

Clicking with a pencil draws a single element at that point. When clicking on a logic gate, relay or communicator with the corresponding pencil, a Signal element will be drawn instead. This shortcut makes it convenient to draw Signals which are only useful in conjunction with these element types.

Lines

Dragging with a pencil draws elements along a straight line. The line is starts from where the mouse button was held down and ends at the point of release, snapping to the grid as a horizontal or vertical line.

Polylines

A polyline is a connected sequence of lines. Holding Shift allows polylines to be drawn. While Shift is held down, clicking draws a line connected to the end of the polyline. Pressing Backspace deletes the last line. Double-clicking or clicking after releasing Shift completes the polyline.

Drawing a polyline with the Insulated Wire pencil will place Conductive Wire elements between adjacent line segments.

Selection Mode

The selector is used for making selections. Dragging the selector draws the selection rectangle in the play area. When the mouse is released, elements that lie in the selection rectangle are added to the selection. Selected elements are initially marked blue, and are marked red once an operation is performed on the selection. While the selection is blue, it may still be modified. While using the selector, holding Shift adds new elements to the selection and Alt removes elements.

Intelligent Selection

Double-clicking on an element selects that element and logically connected ones. Triple clicking on an elements selects that element and all connected ones. Intelligent selection works as expected with Shift and Alt.

Clipboard management

A clipboard is a temporary data buffer used for copy/paste operations. Circuit Sandbox supports 10 clipboards, labeled 0 to 9. Clipboard 0 is also known as the default clipboard and is used by the Ctrl-C/X/V shortcuts. Clipboards 1-9 are known as the extended clipboards.

Clipboards are shared between all instances of Circuit Sandbox, and will be cleared when the last instance of Circuit Sandbox is terminated.

You can copy to or paste from any clipboard using Ctrl-Shift-C/V, which will launch the clipboard menu. You can select a target clipboard by clicking on the respective button or pressing the corresponding number key.

To make repeated clipboard access convenient, the default clipboard caches the contents of the last used clipboard. When copying to an extended clipboard, the copied content will also be duplicated in the default clipboard. Similarly, pasting from an extended clipboard also causes the pasted content to be cached unless the chosen clipboard is empty.

Eyedropper

The eyedropper is a special tool that is active when holding E. When active, the default behavior of all mouse buttons are disabled and clicking on an element in the play area binds that mouse button to that element.

Beginner mode

Extended usage information are available in-game for some editing tools. Turning on beginner mode will make these additional information visible when using those tools. Press B to toggle beginner mode. Beginner mode is turned off by default.

Keyboard Shortcuts

Here is a summary of all keyboard shortcuts in Circuit Sandbox:

Default shortcuts	
Ctrl-A	Select all
Ctrl-N	Open blank circuit
Ctrl-O	Open

Ctrl-S	Save
Ctrl-Shift-S	Save As
Ctrl-V	Paste from the default clipboard
Ctrl-Shift-V	Paste from a particular clipboard (launches the clipboard menu)
Ctrl-Z	Undo
Ctrl-Y	Redo
Space	Start or pause the simulator
Ctrl-Space	Set FPS
R	Reset the simulator
E (while held down)	Activate eyedropper
B	Enable or disable beginner mode
Esc	Exit a context (e.g. FPS dialog)
Selection mode shortcuts	
Ctrl-C	Copy to the default clipboard
Ctrl-Shift-C	Copy to a particular clipboard (launches the clipboard menu)
Ctrl-X	Cut
H, V	Flip horizontally/vertically
D or Delete	Delete
[,]	Rotate 90° counterclockwise/clockwise
Arrow key	Move selection by one step
Ctrl-[Arrow key]	Move selection by four steps
Clipboard menu shortcuts	
0	Select the default clipboard
1, 2, 3, 4, 5, 6, 7, 8, 9	Select an extended clipboard
Left, Right	Go to previous/next page

Appendix A — File I/O

We usually do not want files to be read from or written to at every single simulation step. The design of file input and output takes into consideration this fact. Instead, the unused direction of the bidirectional communication channel is used to control the flow of data.

Both the transmitting and receiving directions of file communicators operate on discrete chunks of data. The off signal is transmitted indefinitely when there is no data to send, and every chunk of data starts with the on signal to indicate the start of the chunk. Chunks may be transmitted in succession without any space between consecutive chunks.

Note: In the chunk formats below, the command bits are sent from left to right.

File Input Communicator

Files are read byte-by-byte. Briefly, the following steps are used:

1. Your circuit transmits a request for the next byte.
2. When the read operation is completed, the byte is received by the communicator. The number of simulation steps that elapses before the byte is received is unspecified.

Transmission Format

Each transmitted chunk is 3 bits long (including the start-of-chunk signal, denoted by "S")

Transmitted Chunk		Description
S	Command	
1	00	Request for next byte of file.
1	01	Check if more bytes are available from the file (i.e. not yet EOF).
1	10	(reserved for future use)
1	11	(reserved for future use)

Reception Format

Each received chunk is a reply to a transmitted chunk, and they are guaranteed to be received in the order that the requests were transmitted. However, the number of simulation steps before the reply is received is unspecified.

The received chunk is prefixed with the command that was used to request this chunk. The length of the payload depends on the command.

Received Chunk			Description of Payload
S	Command	Payload	
1	00	8 bits	Next byte of file, least significant bit first.
1	01	1 bit	On if at least one byte is available, off otherwise.
1	10		(reserved for future use)
1	11		(reserved for future use)

Loading Files while Simulation is Running

If a byte request command is transmitted when there are no more bytes in the file, a reply will not be received until a new file is loaded (by the user).

When a EOF check returns off (i.e. EOF reached), it means that there are no more bytes available from the input file that is currently associated with this communicator.

However, it is possible to open another file for input without resetting the circuit, meaning that subsequent byte request commands may still receive a valid reply. Circuits that expect multiple files to be loaded while running should be aware of this possibility, or avoid checking for EOF at all.

File Output Communicator

Files are written byte-by-byte. Briefly, the following steps are used:

1. Your circuit transmits a request to write a byte. This byte will be placed into a buffer and written to the disk as soon as possible.
2. After the byte is written to disk, a chunk will be received indicating that the request is complete. The number of simulation steps that elapses before this acknowledgement is received is unspecified.

Transmission Format

Each transmitted chunk is 11 bits long (including the start-of-chunk signal, denoted by “S”), depending on the command.

Transmitted Chunk	Description

S	Command	Payload	
1	00	8 bits	Request to write the payload to file. Least significant bit of payload goes first.
1	01		(reserved for future use)
1	10		(reserved for future use)
1	11		(reserved for future use)

Reception Format

Each received chunk is a reply to a transmitted chunk, and they are guaranteed to be received in the order that the requests were transmitted. However, the number of simulation steps before the reply is received is unspecified.

Received Chunk			Description
S	Command	Payload	
1	00	none	Acknowledgement that file write completed.
1	01		(reserved for future use)
1	10		(reserved for future use)
1	11		(reserved for future use)