circuitArduino Documentation

This is a circuit simulator derived from the java circuit simulator developed by Paul Falstad. (https://www.falstad.com/circuit/)

It essentially includes all the features of that simulator. Later, Paul Falstad and collaborators ported the java simulator to javascript and much more features have been added since, which are not included in the present simulator.

As this simulator closely follows the P. Falstad version, the instructions of use he provides mostly apply here. These can be found in https://www.falstad.com/circuit/directions.html

1. UI features

circuitArduino user interface differs from the original simulator in the following:

- When a component is dragged connections to other components are kept. In order to break connections, double click the component before dragging.
- To exit the component insertion mode you can simply double click anywhere on the background, in addition to the standard method of pressing 'esc'.

2. Arduino simulation

2.1 Selecting the sketch

circuitArduino supports simulating Arduino sketches. To do so, you must first create a sketch in your favorite IDE and compile it to an .hex file. In the official Arduino IDE this is accomplished by selecting 'Sketch'->'Export compiled binary'. This will create a .ino.hex file in the sketch folder. To simulate this sketch, in circuitArduino do:

> go to 'Arduino' menu -> 'Select sketch' and browse and select your .hex file

Notice that you must compile the code to an atmega328P based board such as the arduino Uno.

2.2 Setting up ports

The arduino catalogue in circuitArduino consists of Digital ports and Analog ports. You insert these as you do with any other component in the circuit model, i.e., open the context menu (right-click), select the 'Arduino' category and choose the desired item. These components are displayed as other 1-terminal components of the simulator, such as 1-terminal voltage sources and logic inputs/outputs.

While an arduino Digital port can act as a voltage source or a voltage reading depending on how it is configured in the sketch, an arduino Analog port is always a reading port. You can choose the ports address by editing the component properties.

2.3 Known limitations

1) The simulator replicates the atmega328P-based boards, therefore is not compatible, for instance, with Arduino mega.

2) The simulator uses the sample time of the fastest timer as the step size. Arduino timers do not increment at each clock tick but at a fraction of the clock frequency, set by the pre-scaler parameter. For instance, if the pre-scaler is 64 in a 16MHz processor, the timer frequency is 0.25MHz and its sampling time is 4μs. To make the simulation faster, the timer having the shortest sampling time is found and the simulator uses that value as the simulation step size. A consequence of this is that non timed instructions in your code will be executed less frequently, since the simulated Arduino will effectively process at a slower rate (at 0.25MHz in the example above).

2.4 Examples

The table below lists the basic categories of Arduino demos and identify the ones included in the software. To access them go to Circuits->Arduino and select the category and the demo desired.

Category	Demo	Included	Comment
Basic	Analog Read Serial	✓	Read a potentiometer, print its state out to the Arduino Serial Monitor.
	Blink	✓	Turn an LED on and off.
	Digital Read Serial	✓	Read a switch, print the state out to the Arduino Serial Monitor.
	Fade	✓	Demonstrates the use of analog output to fade an LED.
	Read Analog Voltage	✓	Reads an analog input and prints the voltage to the Serial Monitor.
Digital	Blink Without Delay	✓	Blink an LED without using the delay() function.
	Button	✓	Use a pushbutton to control an LED.
	Debounce	✓	Read a pushbutton, filtering noise.
	Digital Input Pullup	✓	Demonstrates the use of INPUT_PULLUP with pinMode().
	State Change Detection	✓	Count the number of button pushes.
	Tone Keyboard	-	A three-key musical keyboard using force sensors and a piezo speaker.
	Tone Melody	-	Play a melody with a Piezo speaker.
	Tone Multiple	-	Play tones on multiple speakers sequentially using the tone() command.
	Tone Pitch Follower	-	Play a pitch on a piezo speaker depending on an analog input.
Analog	Analog In Out Serial	✓	Read an analog input pin, map the result, and then use that data to dim or brighten an LED.
	Analog Input:	✓	Use a potentiometer to control the blinking of an LED.
	Analog Write Mega	✓	Fade 12 LEDs on and off, one by one, using an Arduino Mega board.
	Calibration	✓	Define a maximum and minimum for expected analog sensor values.
	Fading	✓	Use an analog output (PWM pin) to fade an LED.
	Smoothing		Smooth multiple readings of an analog input.