

Arduino Simulator usage instructions

In this manual we will be using the following simulator URL: `http://arduino.cognixtec.com/arduino/`. Replace in the following steps with the one you are using.

1 Install the board

The first step is to install the board in the Arduino IDE. We should go to the simulator site and copy the link “Arduino IDE package”.



Figure 1: Arduino IDE package link

Then in the Arduino IDE menu File > Preferences we add the copied URL:

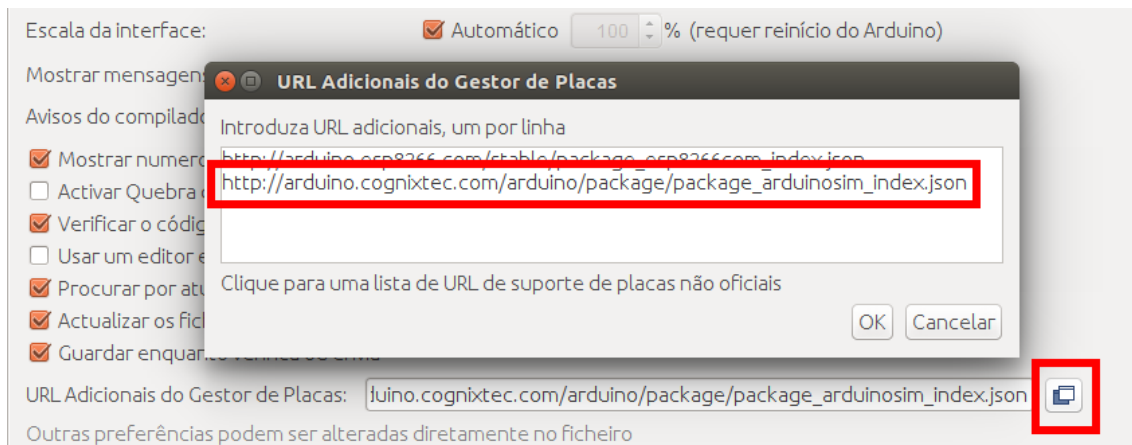


Figure 2: Arduino IDE Board Manager URL

The final step to install the board is access the menu Tools > Board: ... > Board Manager and find the **arduinosisim** board.
There we press the Install button.

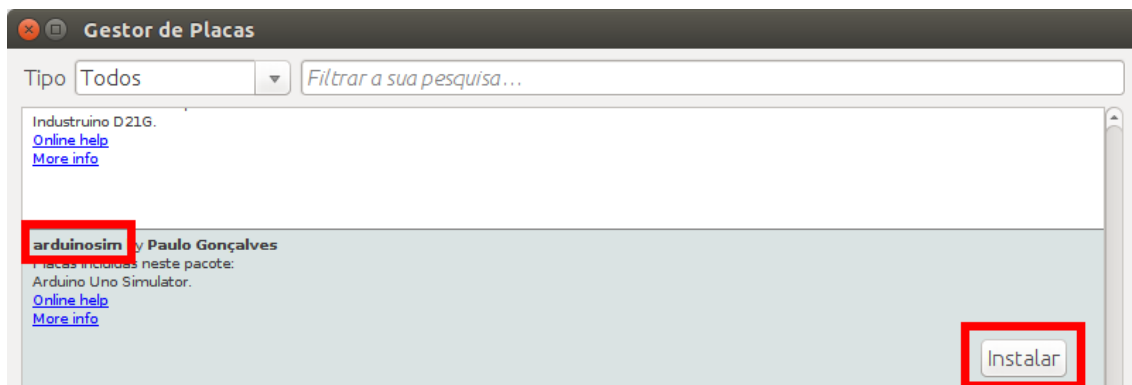


Figure 3: Arduino IDE Board Manager installation

From this moment we can select the “Arduino Uno Simulator” board.

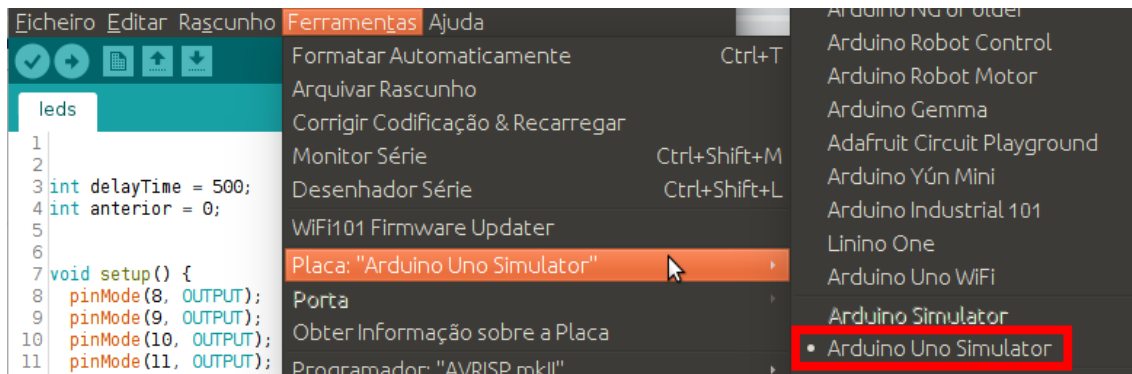


Figure 4: Arduino IDE Board Selection

2 Register

Now that the board is installed in the Arduino IDE we need to register an account in the site.

We go to the “Register” link:

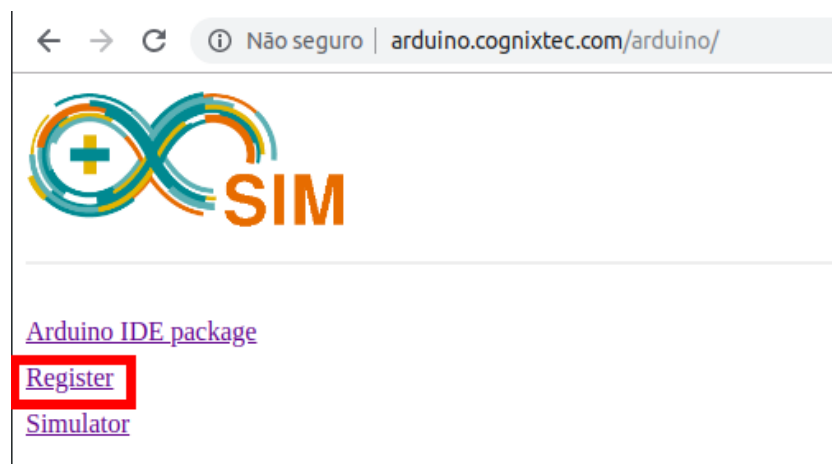


Figure 5: Register link

There we choose a username (an email), a password and we put our name.

The username must be an email that we have access because it will be sent an email to confirm the account.



A screenshot of a web browser showing the registration page for the Arduino Simulator. The browser's address bar displays "Não seguro | arduino.cognixtec.com/arduino/register". The page features the Arduino Simulator logo at the top, which consists of a stylized infinity symbol with a plus sign inside a circle, followed by the word "SIM" in orange. Below the logo, there are four input fields labeled "Username", "Password", "Repeat password", and "Name". At the bottom of the form is a "Register" button.

Figure 6: Register

We will receive an email like the one bellow and must press the “Activate account” button.

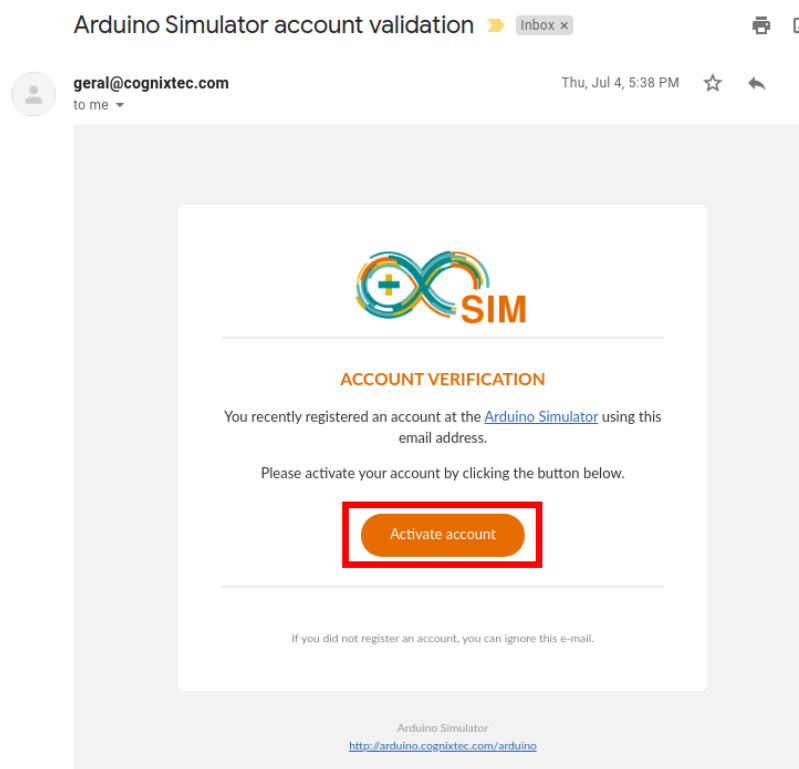


Figure 7: Account verification email

After that we can login with the email and chosen password:



Figure 8: Login

3 Simulator usage

The interface presented to us has a toolbar at the top, a component palette on the left, and a drawing area in the center. Components must be dragged from the pallet to the drawing area to be added to the circuit. Connections are made by clicking on a connection point (circles in blue, red or black) and dragging to another one. Components or connections can be selected with the mouse and deleted with the Delete key. We can only add one Arduino to the drawing area (circuit).

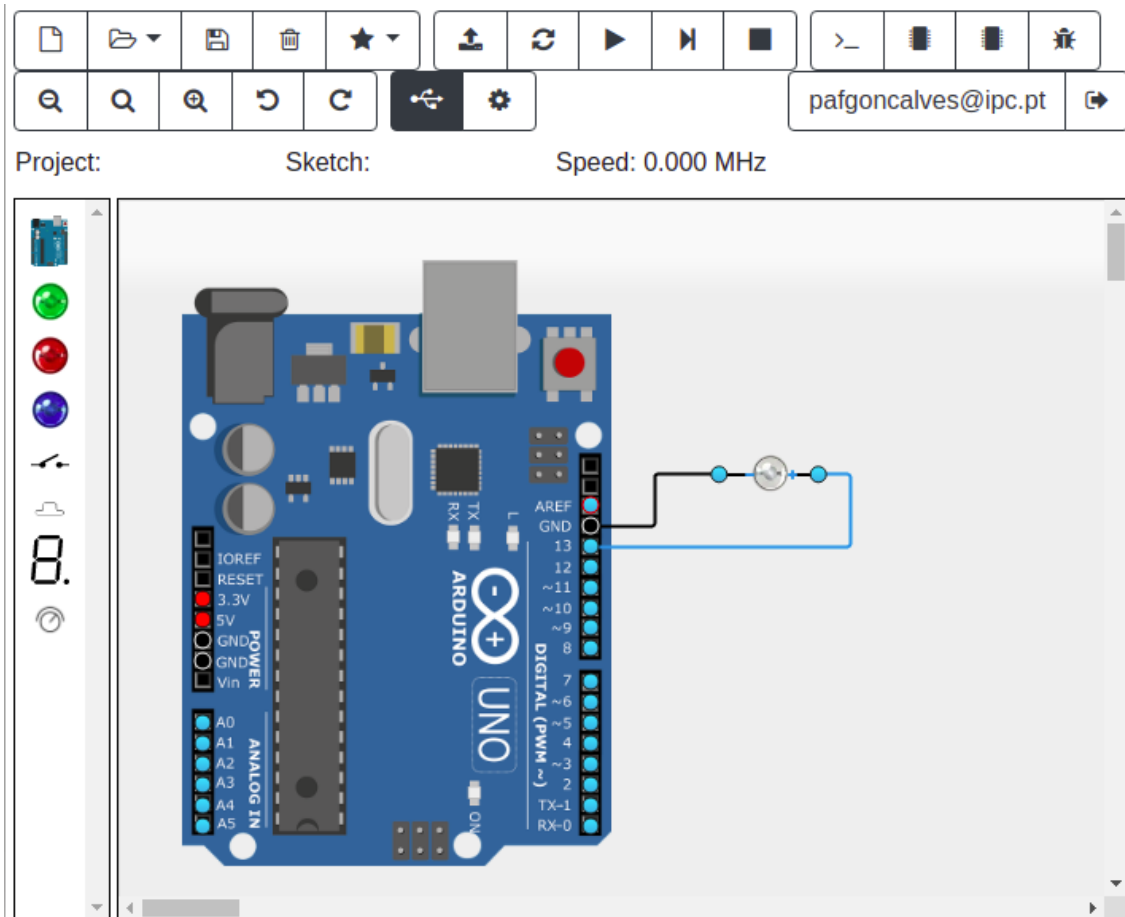


Figure 9: Simulator interface

3.1 Components

3.1.1 LEDs

Although in a real circuit there is a need to connect an LED with a resistor in series, in the simulator the LEDs are considered with an internal resistor so there are no resistors on the pallet.

The LED's have the regular polarity:

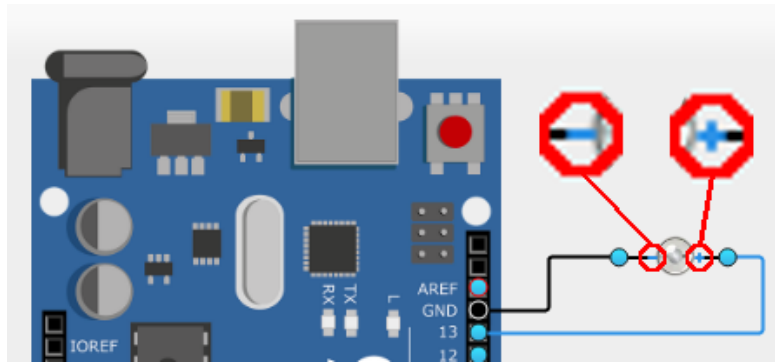


Figure 10: Simulator LEDs connection

3.1.2 Switch

The switch component can have two states, on or off. To change state click on the component.

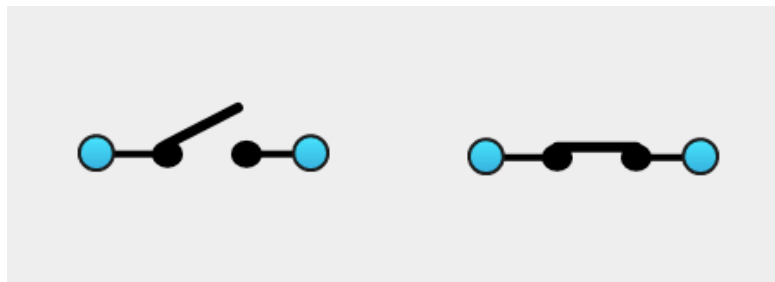


Figure 11: On-off switch

3.1.3 Push button

The push button component can also have two states, on or off. While the component is pressed with the mouse it is in the on state. For the off state simply release the mouse button.

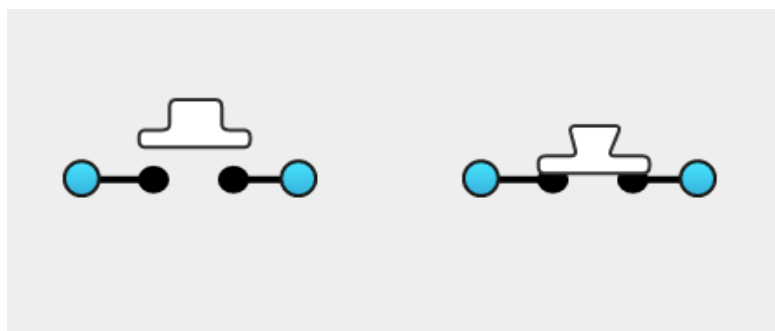


Figure 12: Push button

3.1.4 7 segment display

The 7 segment display is of common cathode. The pins are identified in the image.

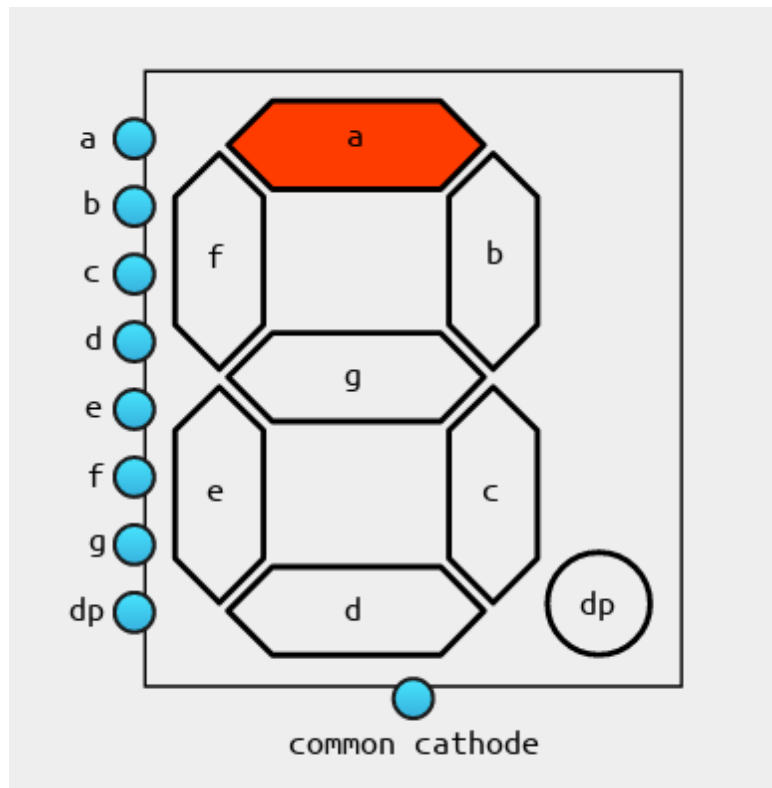


Figure 13: 7 segment display

3.1.5 Potentiometer

The potentiometer component can be used to connect to the analog inputs of the Arduino. The output is the middle connector and the green slider can be moved with the mouse.

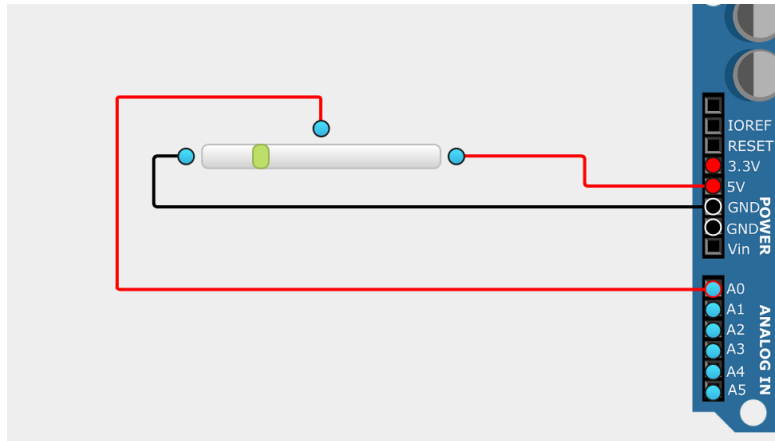


Figure 14: Potentiometer

3.2 Programming

The programming (sending of the code from the Arduino IDE to the Simulator) is done in the same way as any other Arduino. Also, the USB symbol in the Simulator interface must be active for the programming to work:

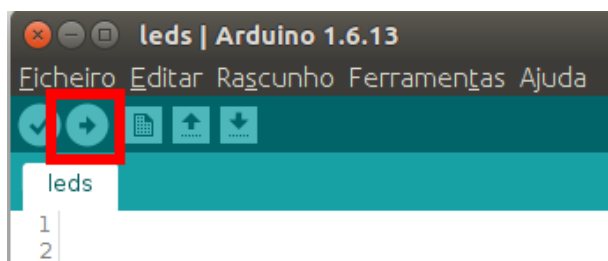



Figure 15: Arduino IDE programming button



Figure 16: Simulator programming active

It can happen that Operating System firewall gives a warning about a connection to the Internet. It's normal and must be allowed.

If there is a problem contacting the Simulator, the programmer will show a message like the one below. The easiest solution is to cancel and try again, but the parameters to put in the box can be found in the config area  of the Simulator interface. They are Programmer URL and Simulation ID.

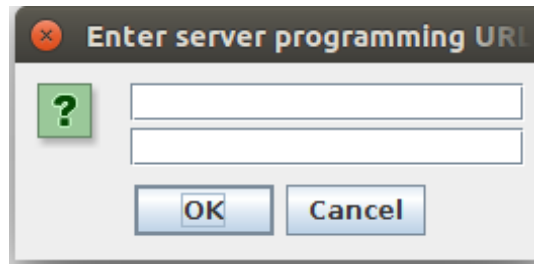


Figure 17: Programming tool parameters

Now we can use the toolbar buttons to pause, run or step-by-step the microcontroller. We can also inspect the FLASH and SRAM and add or remove breakpoints in the FLASH addresses or source code lines (in Simulator debug window). It can also be used to manage projects (create new, save, open and delete).



Figure 18: Toolbar

3.3 Project management

In the first part of the toolbar, it's possible to create a new project, open an existing one, save the current work or delete the project. When saving, if it is the first time, it will be asked a name for the project.

There is also a menu with some predefined circuits (the one with a star), that can be loaded into the drawing area.

The last button in Figure 19 allows the user to directly upload a HEX or ELF file with code for the Arduino.

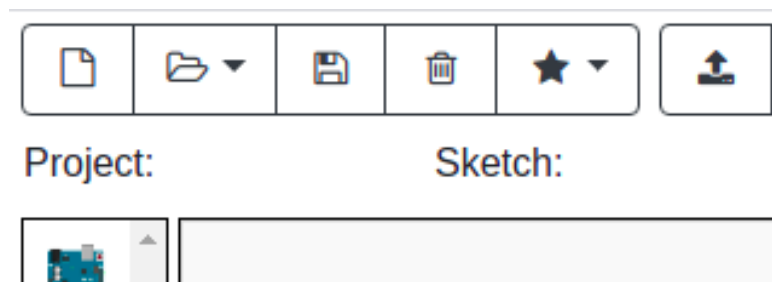


Figure 19: Project management buttons

3.4 FLASH inspection window

In the FLASH inspection window is possible to see the disassemble of the binary code loaded to the Arduino.

If using the step-by-step mode the line signalled with the green dashed line is the execution point. It's possible to click in the address (at pink) to insert or remove a breakpoint. If a breakpoint is installed the address is highlighted in orange.

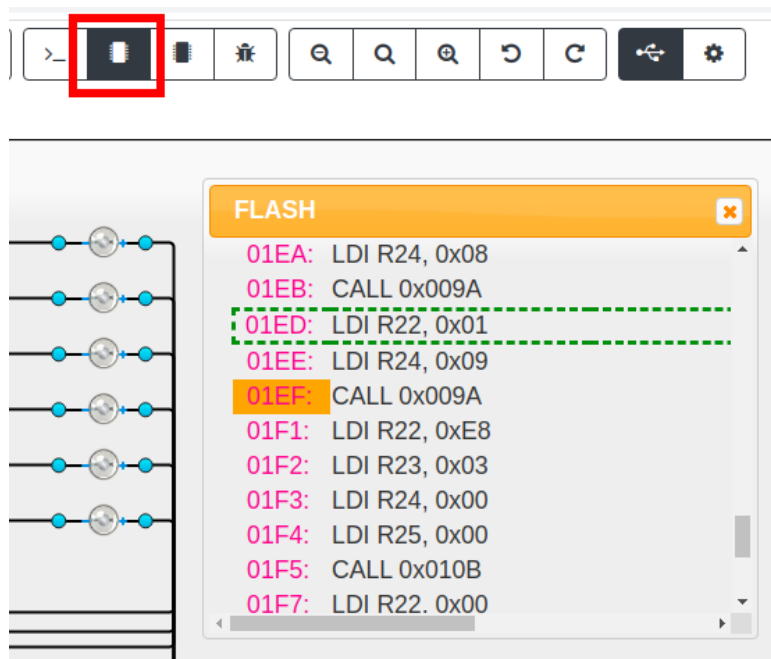


Figure 20: FLASH inspection window

3.5 SRAM inspection window

In the SRAM inspection window is possible to analyse the contents of the memory of the microcontroller. In addition to regular memory, all processor registers are mapped in memory.

When a register is written it is highlighted in orange for brief moments for the user to have an idea of the memory zones that are being manipulated.

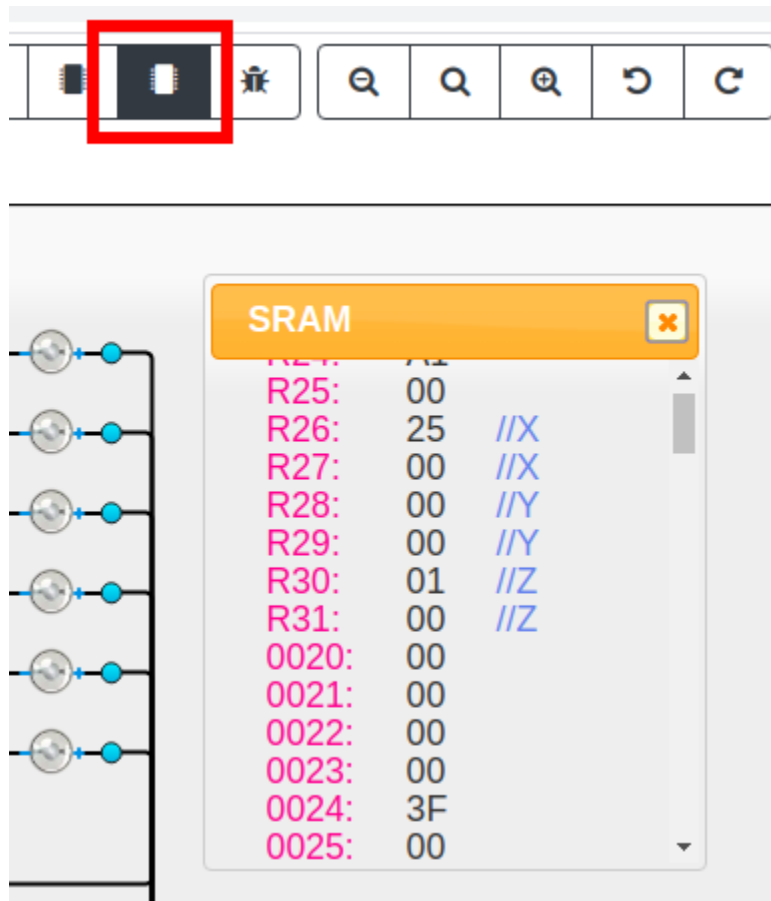


Figure 21: SRAM inspection window

3.6 Debug window

In the debug window it is possible to see the source code of the Arduino sketch if an ELF file was loaded to the simulator (the default for Arduino IDE programming).

The lines marked in yellow are the ones that correspond to some instruction in the binary program loaded in the microcontroller and can be clicked to add a breakpoint. When a breakpoint is installed the line is highlighted in orange.

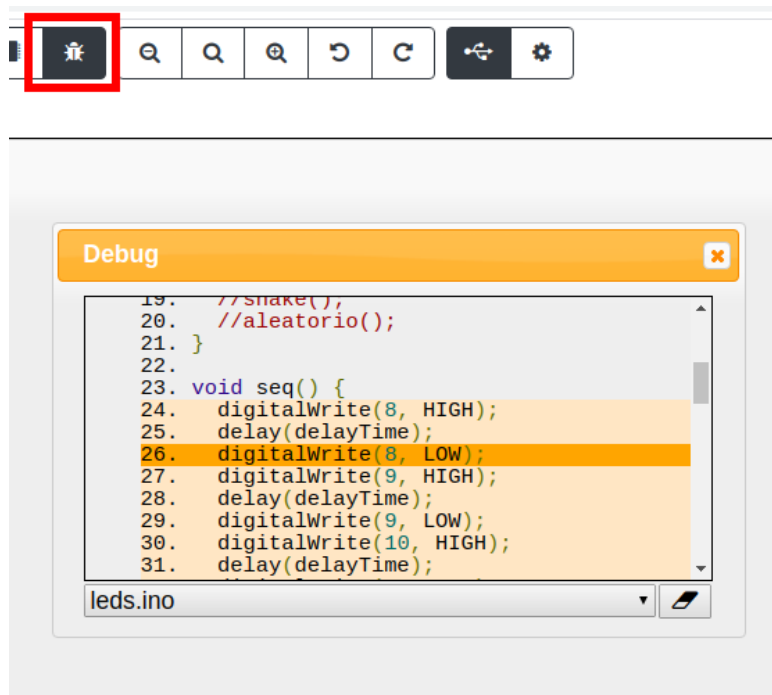


Figure 22: Debug window

3.7 Serial port monitor

The serial port 0 of the microcontroller is implemented but the input/output is redirected from the TX/RX pins to the serial port monitor. It is not possible to use these pins in the circuit for serial input/output.

It has similar functions to the Serial Monitor of the Arduino IDE. Every write to the serial port will show up in the Serial monitor of the simulator and is possible to write in the text box to send data that will be read by the microcontroller.

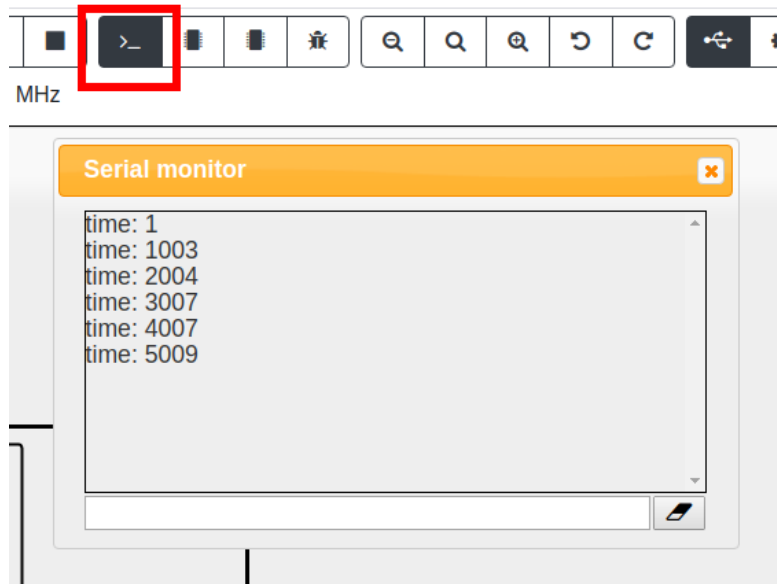


Figure 23: Serial port monitor window

4 Implemented features

The Arduino Simulator implements the following peripherals:

- **External Interrupt**
The External Interrupts work as expected and all registers configurations are followed.
- **Pin Change Interrupt**
The Pin Change Interrupts work as expected and all registers configurations are followed.
- **I/O Ports**
All Port B , Port C and Port D inputs/outputs that are available on the Arduino Uno are implemented.
- **Timer 0**
Timer 0 is implemented in a way to simulate what is expected from the Arduino API that is one interrupt every 1 ms. This allows the delay API function to work as expected and make delays work with time clock regardless of the simulator speed. Every configuration of the timer registers is ignored.
- **USART 0**
The USART 0 is emulated and not connected to the TX/RX pins on the board. The input/output is redirected to the Serial Monitor of the simulator.

- **Analog to Digital Converter**

The Analog to Digital Converter on the Port A (ADC0 to ADC5) is implemented. The AREF pin is also implemented.

- **Implemented Interrupts:**

- INT 2: External Interrupt Request 0
- INT 3: External Interrupt Request 1
- INT 4: Pin Change Interrupt Request 0
- INT 5: Pin Change Interrupt Request 1
- INT 6: Pin Change Interrupt Request 2
- INT 17: Timer/Counter0 Overflow
- INT 19: USART Rx Complete
- INT 20: USART Data Register Empty
- INT 22: ADC Conversion Complete

- **Instructions**

All AVR instructions except DES, SLEEP and SPM are implemented.