How to use the STSW-LED1202GUI PC interface software for LED1202 based evaluation boards

Introduction

This GUI software lets you control and configure evaluation boards based on the LED1202 LED driver from your PC.

The GUI provides read and write access to the device registers and configuration files, and lets you edit patterns to build your own LED sequences. An RGB Mode with white point selection lets you manage RGB LEDs by grouping 3 channels per LED (CH0/1/2 for RGB0, CH3/4/5 for RGB1, and so on).

Furthermore, you can build scripts in an environment similar to BASIC to generate LED effects and store them on the STM32L073 microcontroller. You can then launch the same scripts on the board in Standalone Mode.

1 How to run the GUI

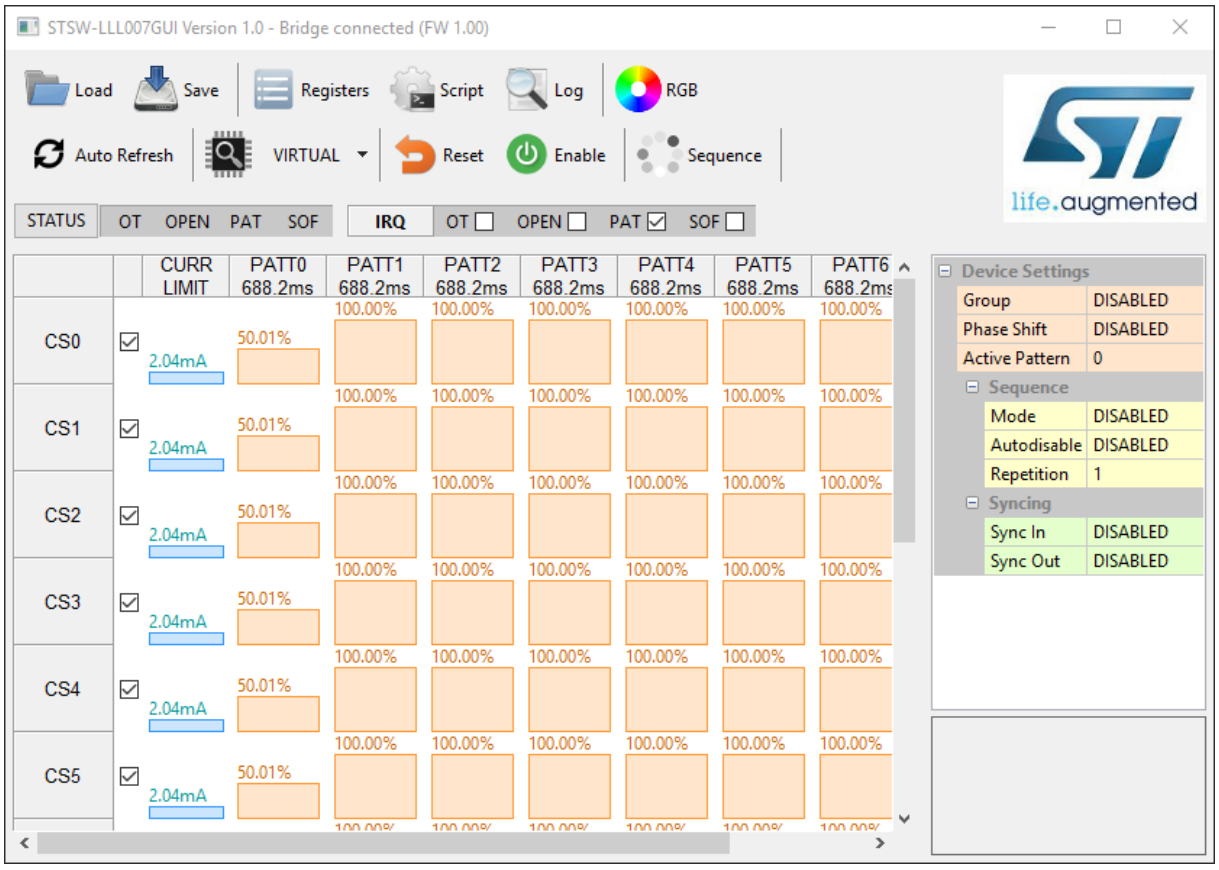
This GUI is designed to communicate with evaluation boards on a Windows PC via USB.

Step 1. Ensure a Virtual Com Port driver for Windows is installed on your PC
ST provides this driver for Windows, but it should already present from Windows 10.

Step 2. Connect the USB cable from the PC to the evaluation board.

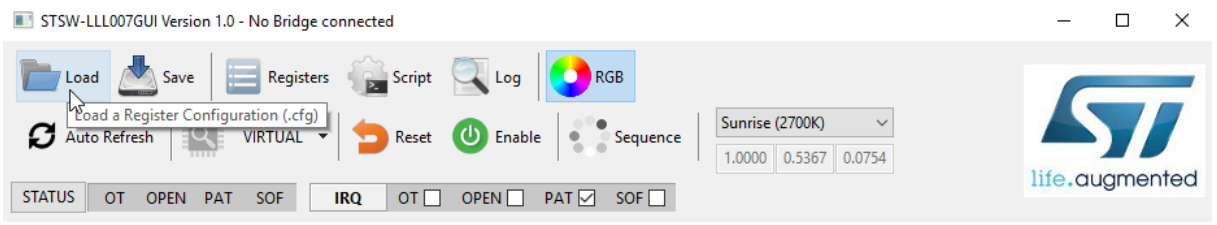
Step 3. Run the GUI by launching the file `STSW-LED1202GUI.exe`.
The title shows the GUI version number and the firmware version of the connected board. If no board is recognised, the title will show the message “No Bridge connected”. The main window shows the reported values are the ones stored inside the GUI itself for a Virtual device. Virtual device doesn’t enable a real bus communication, so it allows to play with GUI even if no any board is connected.

Figure 1. STSW-LED1202GUI main window



Step 4. Move your mouse over the icons to reveal pop-up tooltips describing their function.

Figure 2. Tooltip example



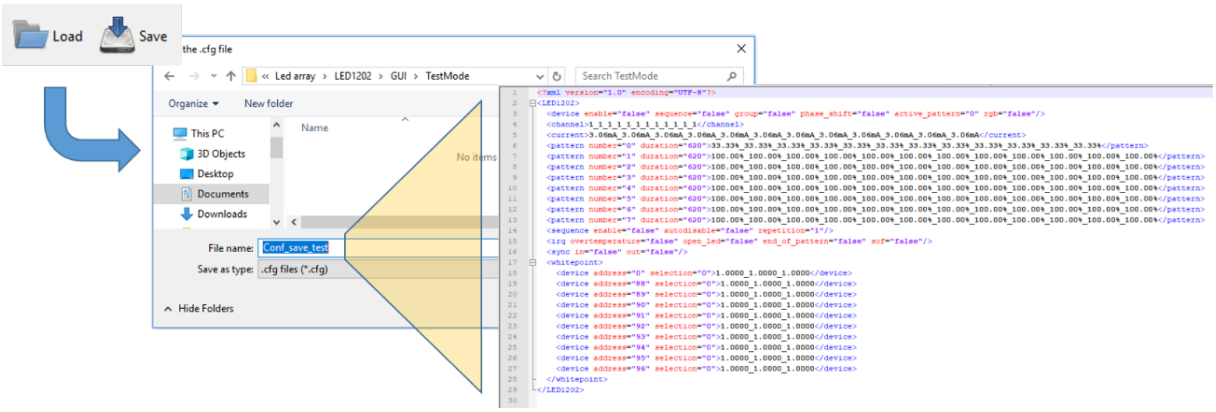
2 Overview of the GUI

The GUI consists of several different configuration and management sections, as well as a title bar showing the connection status of any boards and relevant software and firmware versions.

2.1 Configuration management

LED1202 device configurations can be saved to a .cfg file and subsequently loaded and applied to another device. The configuration file is saved without reference to a specific I2C address, so it can be applied to any device regardless of the I2C address set.

Figure 3. ; File configuration manager and example of configuration file

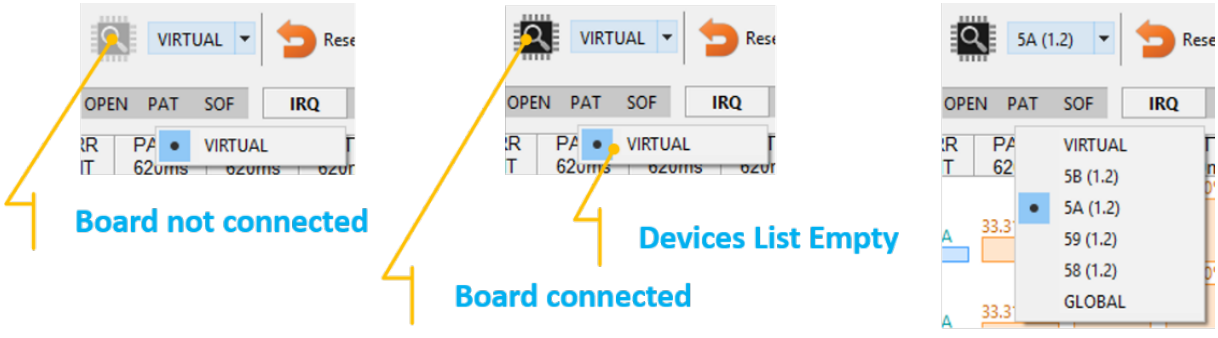


2.2 Device search and selection

An icon with a search symbol in the GUI indicates when an evaluation board is available, and is grayed out when no boards are recognized. When a board is recognized, you can click on the icon to search for available devices.

The VIRTUAL device is always available; it is a simulated device in Virtual Memory and does not involve any external communication that can be logged. It allows you to experiment with the GUI and create configurations that can be saved and loaded into a physical LED1202 device when it is connected.

Figure 4. Device status button and drop-down list



Once the search has concluded, all recognized devices are listed in the drop-down menu. Each device is identified with its I2C address @ 7 bit in hexadecimal and the value of Device ID register in parentheses for silicon revision verification.

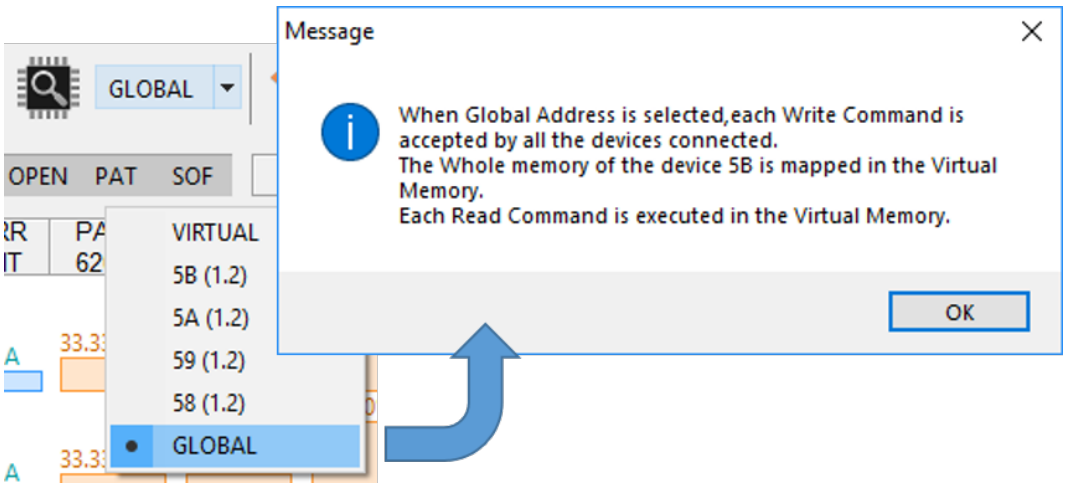
If at least one physical device with a valid I2C address is detected, the drop-down menu also includes the GLOBAL item. Selecting GLOBAL the communication uses the special I2C address of 5Ch @ 7 bit, which is recognized by all the devices regardless of pin A0 and A1 settings. This global address can be used to execute a simultaneous command such as Sequence start to multiple devices.

Note: The GLOBAL channel does not allow read commands to avoid data conflicts caused by devices with different configurations.

When a device is selected, the GUI reads the content of the registers and updates the information displayed in the main window. The first time GLOBAL is selected, a pop-up message appears with information regarding the GUI behaviour in this circumstance.

Note: Selecting GLOBAL overwrites any data using the VIRTUAL configuration with the content of a real device in the Virtual Memory.

Figure 5. Pop up message when GLOBAL is selected for the first time

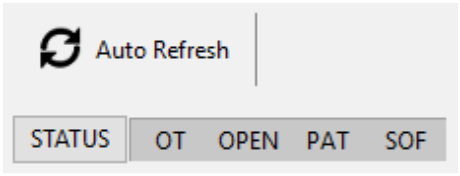


2.3 Status read

Selecting [**Auto Refresh**] causes the GUI to continuously read the device status information in a continuous loop. Alternatively, you can read the device registers manually by pressing the [**STATUS**] button. When a fault is confirmed, the indication name becomes red. If a LED fails, the corresponding channel name (CS0 to CS11) also turns red.

Figure 6. Status buttons

OT = Overtemperature protection
OPEN = open LED fault
PAT = end of pattern sequence
SOF = Start of Frame.

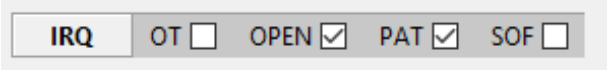


2.4 Interrupt masking to IRQ pin

The IRQ menu allows you to define which failure or status acts on the IRQn pin. A flagged square means the function is active on IRQn, which is equivalent to a ‘0’ (interrupt not masked) field value in the Fault and status mask register.

Figure 7. Indication of which function will generate a level change on IRQn pin

OT = Overtemperature protection
OPEN = open LED fault
PAT = end of pattern sequence
SOF = Start of Frame.



2.5 Device initialization

For any selected device:

- Reset: restore all the register content to the POR value of the selected device
- Enable: enable/disable the selected device

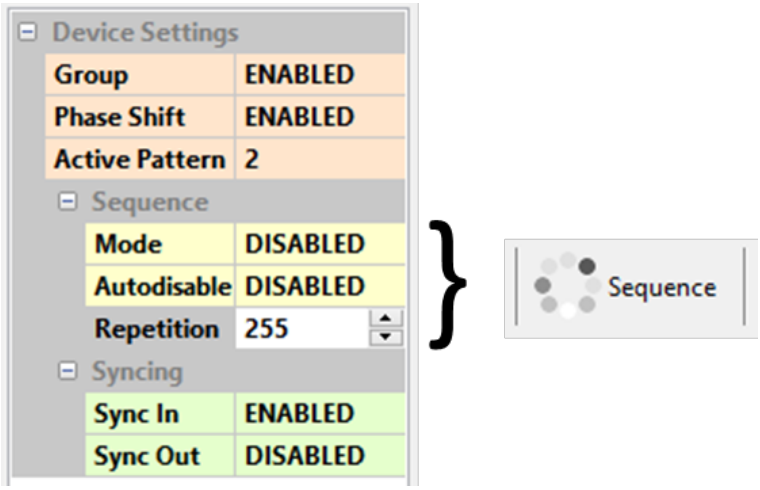
If the selected device is GLOBAL, all the devices on the board will execute the action simultaneously, which is useful for switching all the LEDs on at once.

Figure 8. Reset and Enable buttons

2.6 Device settings and sequence Start/Stop

This section includes Device Settings, Sequence and Synching options.

Figure 9. Settings at device level



Device Settings area:

- Group represents the GCTRL bit and applies the PWM setting for CS0 to all channels; it is disabled in RGB mode.
- Phase Shift represents the SHFT bit and defines whether the channels switch at the same time or with a defined delay.
- Active Pattern ranges from 0 to 7 and acts on PATSEL[2:0] bits; it selects which pattern must be used to set the channel configurations.

Sequence area:

- Mode represents the PATS bit and must be set to ENABLED to allow a sequence to run. Note that all the outputs go OFF as soon as it is set.

- Autodisable represents the AUTODIS bit; if it is ENABLED, the device is disabled at the end of the sequence execution. Note that you must click the [**Enable**] button twice the button to activate the device again:
  1. the first time to reset it
  2. the second time to send the command to reactivate it.
- Repetition ranges from 1 to 255 and acts on the register Pattern Sequence Repetition to set how many times a sequence is repeated:
  – from 1 to 254 represents the same number of repetitions
  – 255 represents an infinite loop
- The Sequence button acts on the PATSR bit to start a sequence if Mode is ENABLED. If Repetition is set to loop infinitely, the sequence can be halted by acting on the Sequence button or setting Mode to DISABLED. Note that when a sequence is set to loop infinitely, the Autodisable setting is not applicable by device, as it never reaches the end.

Synching area:

- Sync In sets if the device to use the internal clock (DISABLED) or accept the clock from pin A1 (ENABLED).
- Sync Out defines whether the device puts the used clock (ENABLED) on pin A0 or not (DISABLED).

In a daisy chain configuration, it is recommended to set the synching bits starting from the last slave of the chain and proceed up to the master device. This procedure avoids any spurious address recognition on A0/A1 pins occurring during device boot.

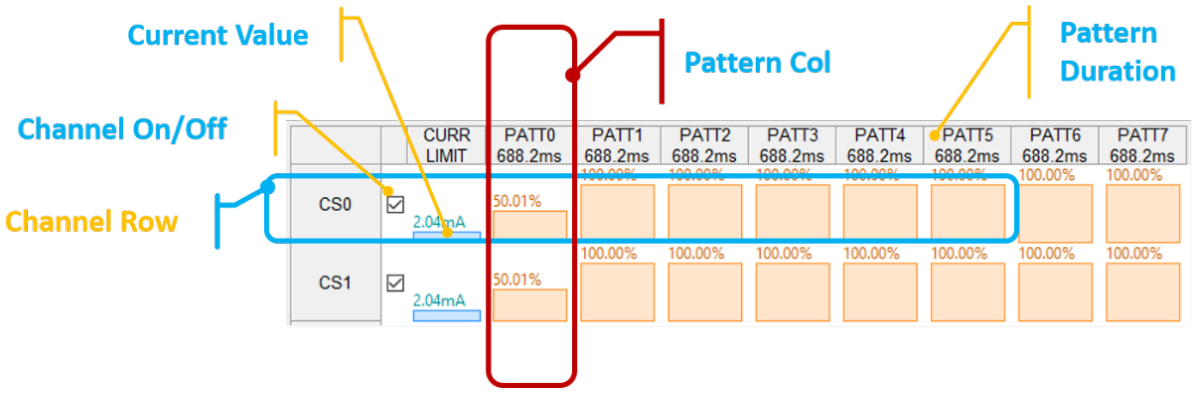
Note: When the master is disabled in a daisy chain configuration, all the devices freeze to their current status as the clock disappears and the internal state machine can no longer progress.

Once the common clock domain is configured appropriately, it is advisable to select the GLOBAL device and then click the [**Enable**] button to ensure that all the devices start synchronously and proceed with the same time base.

2.7 Channel and patterns settings

In the central section of the GUI; it is possible to configure each channel and to define each pattern.

Figure 10. Channel ON/OFF, Current setting, Pattern duration and PWM by channel



The column along the left allows you to enable and disable individual channels through the corresponding check boxes.

The CURR LIMIT column allows you to set the current from 0 to 20 mA per channel by dragging the bar with your mouse or writing the value directly in the box. The indicated value may differ slightly from the actual input because of internal numerical conversion processes.

The PATTx (x ranges from 0 to 7) columns refer to the configuration of each pattern. The pattern duration in ms can be changed by double clicking on the value and entering a new value from 0 to 5660 ms. Similar to the current value, the accepted value may differ slightly after internal rounding operations are performed. Entering ‘0’ will return SKIP, and the paatern will not be executed during the sequence execution.

Note: The column width changes according to the pattern duration from the minimum for SKIP to the maximum for 5660 ms.

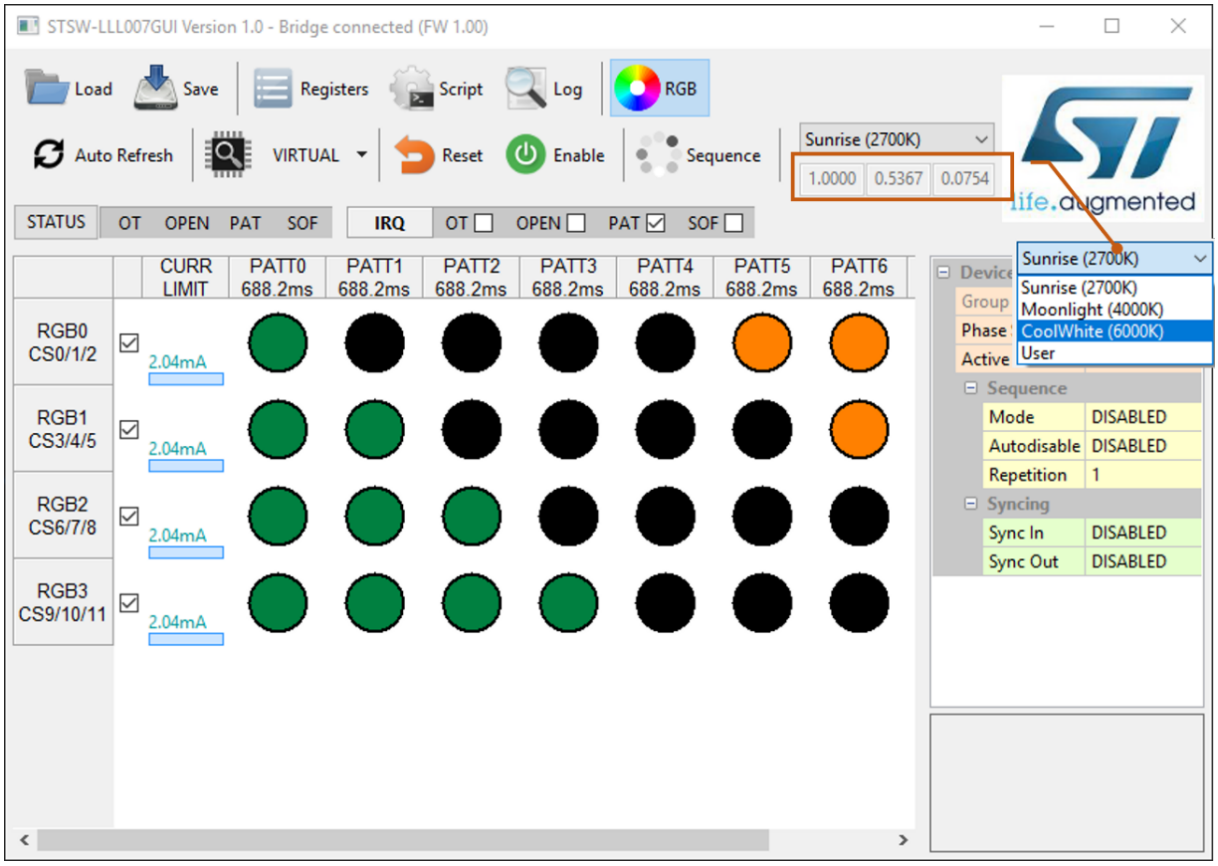
The configuration of each channel in a pattern is set according to the duty cycle in terms of percentage by clicking and dragging the edge of the bar or entering the value in the selected area. This value is also subject to slight variation due to rounding.

If Group is ENABLED, you can adjust Channel 0 of each Pattern and the same configuration is applied to all the 12 channels.

2.8 RGB mode selection

When the connected board uses RGB LEDs, it may be useful to define the desired color instead of setting the individual R, G and B components individually. The GUI is designed to control 4 RGB LEDs per device, with the sequence for each LED as R-G-B starting from CH0-CH1-CH2 and proceeding with the same structure for the subsequent LEDs.

Figure 11. Main window in RGB mode



Selecting the [**RGB**] icon changes the main window by combining channels into groups of three, so the checkbox applies to the corresponding patterns in the same row and selecting the color of the circle defines the PWM value of all the three concurrent channels.

As the PWM values applied to each channel to achieve the desired color are based on a conversion formula computed from a standard RGB LED, the Group setting must be DISABLED in RGB mode in order to allow differentiating the duty cycle by channels.

By clicking on a colored circle, a color selector dialog is shown. Once the new color is selected, the RGB components are applied to the corresponding PWM channels.

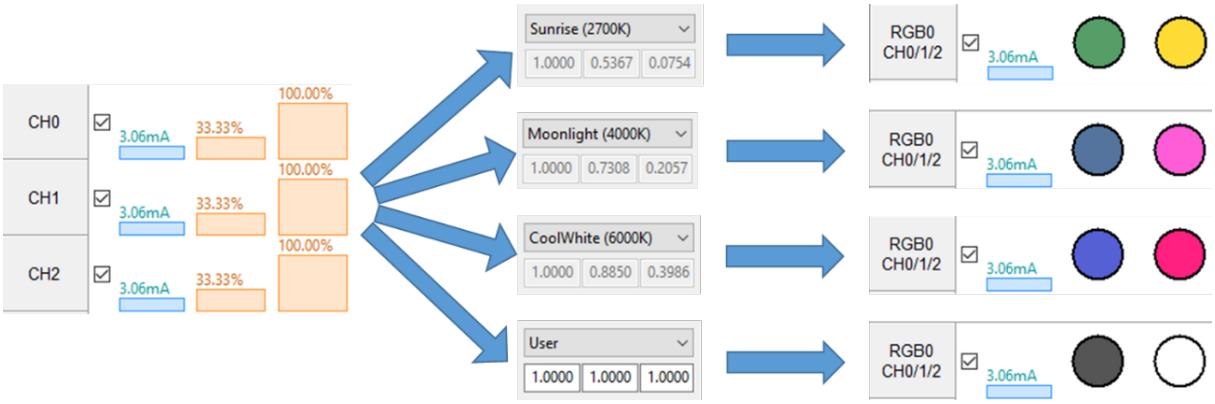
You can choose the color temperature for color to PWM conversion from the dropdown menu, and the applicable conversion coefficients are displayed. On top of the Sunrise (2700K), Moonlight (4000K) and CoolWhite (6000K) presets, you can enter your own coefficients as required.

While separate color temperatures can be assigned to different devices, it is not possible apply a different color temperature to different circles or patterns driven by the same device.

Whenever the GUI switches between monochrome and RGB via the [**RGB**] icon, the appropriate conversion coefficients are applied with the reference values carried over from the previous color space.

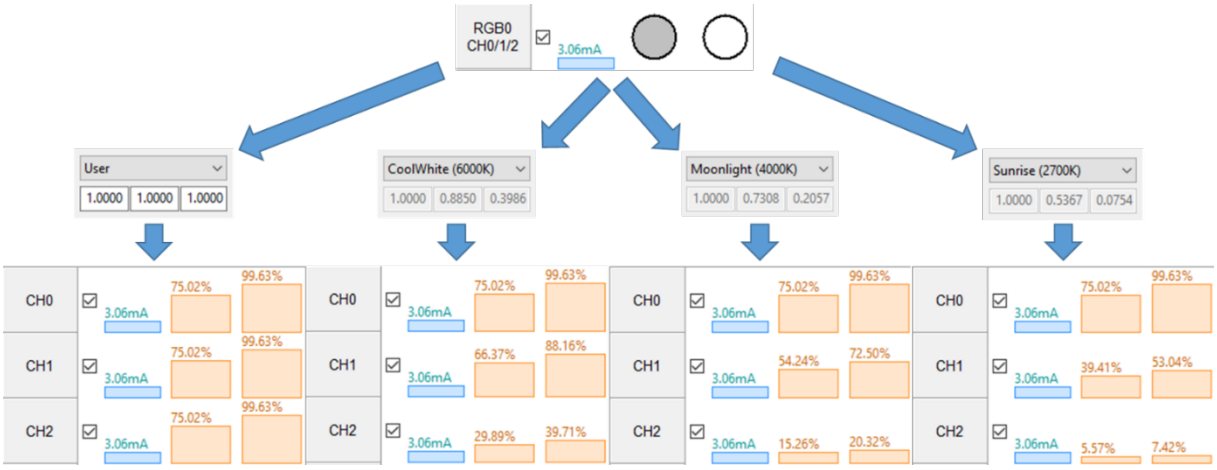
For example, starting from the default configuration and moving from monochrome to RGB, the 33.33% for Pattern0 will become a green color with Sunrise (2700K) temperature. If we modify this color to light gray and return to monochrome, the settings will change for channel 0 (75.02%), channel 1 (39,41%) and channel 2 (5.57%).

Figure 12. From monochrome to RGB



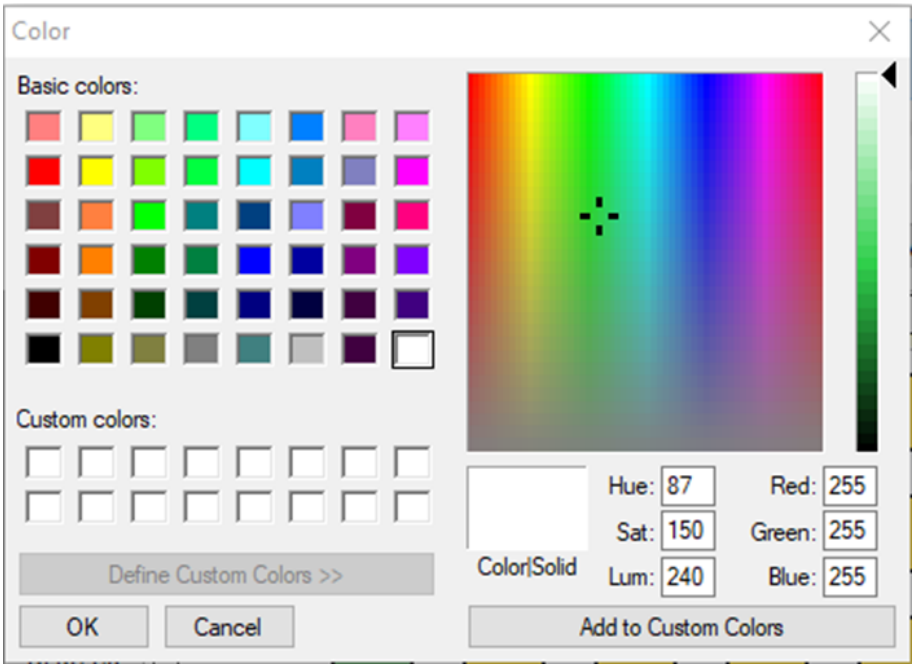
break

Figure 13. From RGB to monochrome



The actual LED colors are generally more accurate than the GUI, especially dark colors that tend to be represented in black on the screen.

Figure 14. **Color selection pop-up in Define Custom Color mode**

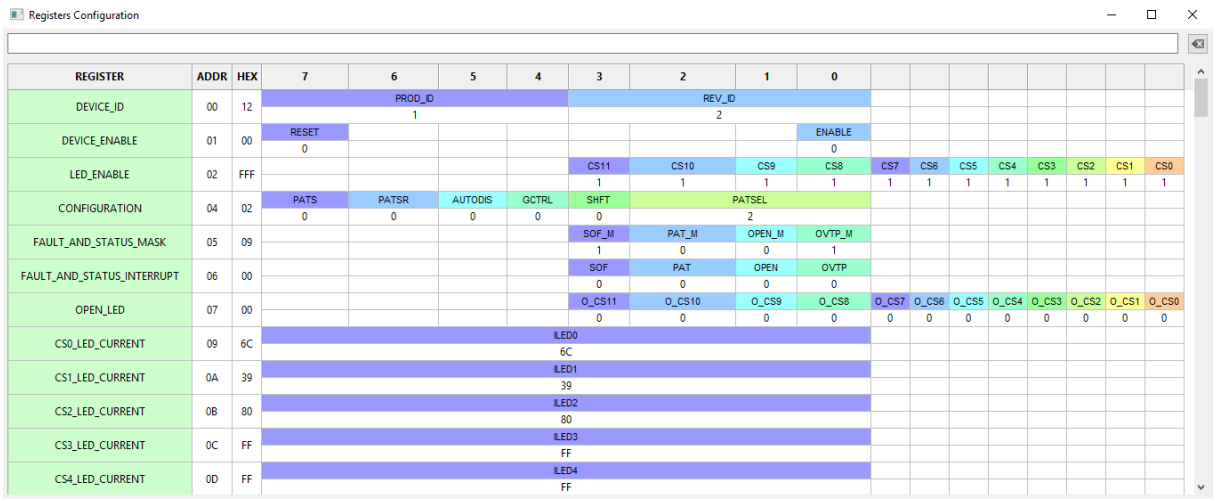


2.9 Special functions

2.9.1 Registers

Clicking on the [**Registers**] icon opens a new window with the full register list for the selected device. Changing the value in the HEX column ,or in the 'bit' columns, allows the GUI to correctly address the Low and the High register.

Figure 15. **Registers window**



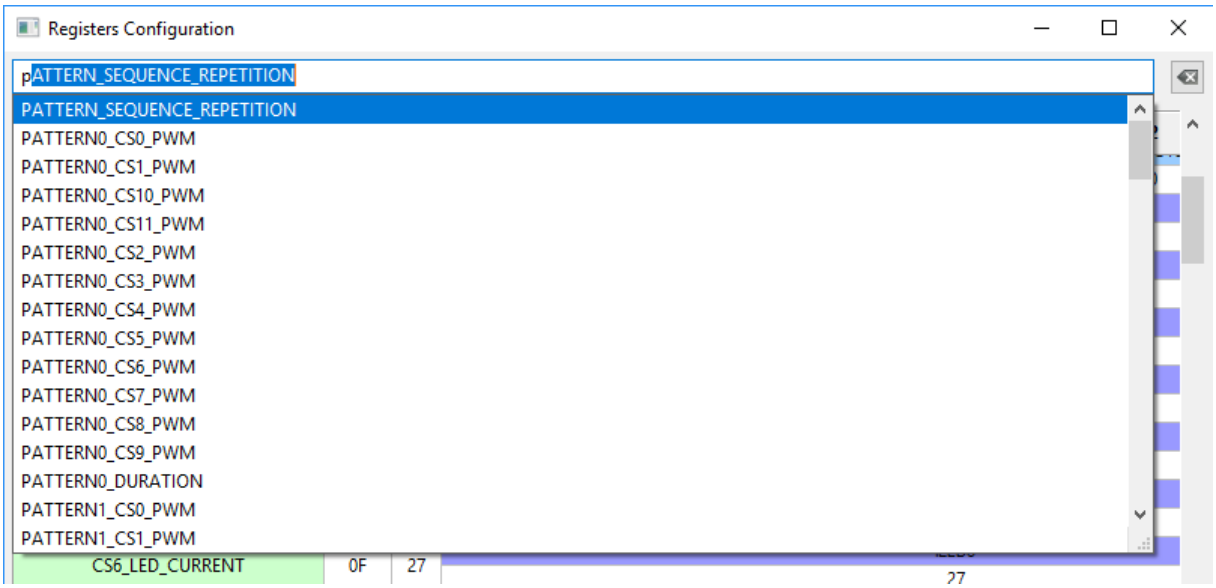
The value provided in the HEX column is reflected in the main window, so when you modify a bar in the main window, the respective register is modified accordingly, and vice versa. Changing the selected device in the main window will close the current register window, which must be reopened to load the relevant settings for the new selected device.

The register value can be modified by clicking on the cell in the HEX column and entering the hexadecimal value or selecting the cell just below the description and entering the hexadecimal value there. The allowed values for any single bit are 0 and 1.

Any modifications will result in updates to the physical register in the device, the read back for confirmation and the update of all the GUI windows.

As this window is manages a long list of 221 registers, a search option is included to help users find the right register quickly.

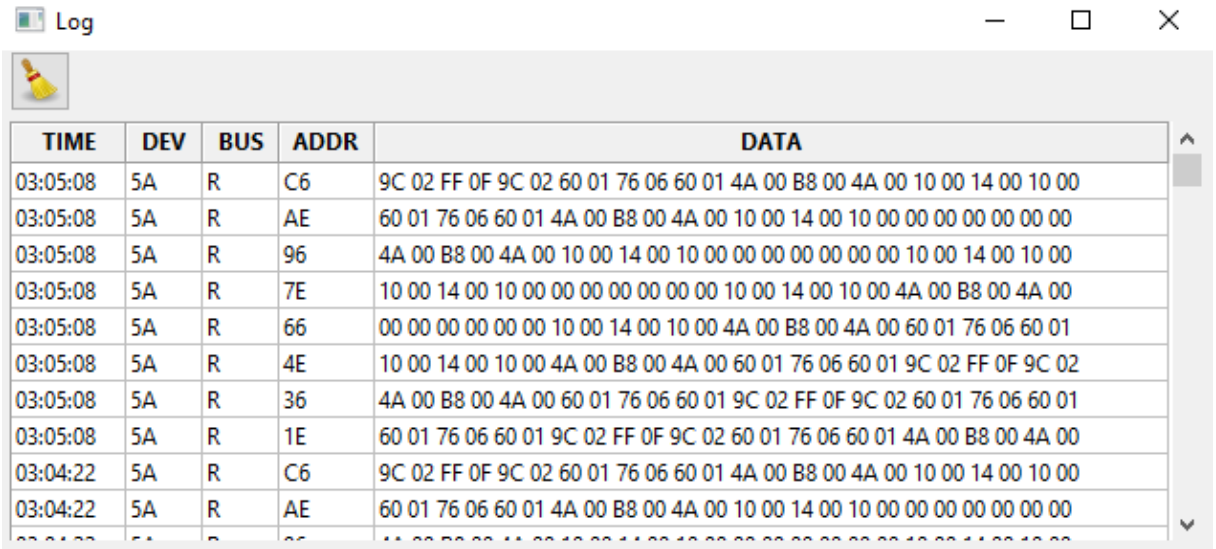
Figure 16. Register search



2.9.2 Log

The [**Log**] button opens a window showing the I2C packets exchanged between the GUI and the real devices, including the sequences sent using global address.

Figure 17. Log window



TIME	DEV	BUS	ADDR	DATA
03:05:08	5A	R	C6	9C 02 FF 0F 9C 02 60 01 76 06 60 01 4A 00 B8 00 4A 00 10 00 14 00 10 00
03:05:08	5A	R	AE	60 01 76 06 60 01 4A 00 B8 00 4A 00 10 00 14 00 10 00 00 00 00 00 00 00
03:05:08	5A	R	96	4A 00 B8 00 4A 00 10 00 14 00 10 00 00 00 00 00 00 00 10 00 14 00 10 00
03:05:08	5A	R	7E	10 00 14 00 10 00 00 00 00 00 00 00 10 00 14 00 10 00 4A 00 B8 00 4A 00
03:05:08	5A	R	66	00 00 00 00 00 00 10 00 14 00 10 00 4A 00 B8 00 4A 00 60 01 76 06 60 01
03:05:08	5A	R	4E	10 00 14 00 10 00 4A 00 B8 00 4A 00 60 01 76 06 60 01 9C 02 FF 0F 9C 02
03:05:08	5A	R	36	4A 00 B8 00 4A 00 60 01 76 06 60 01 9C 02 FF 0F 9C 02 60 01 76 06 60 01
03:05:08	5A	R	1E	60 01 76 06 60 01 9C 02 FF 0F 9C 02 60 01 76 06 60 01 4A 00 B8 00 4A 00
03:04:22	5A	R	C6	9C 02 FF 0F 9C 02 60 01 76 06 60 01 4A 00 B8 00 4A 00 10 00 14 00 10 00
03:04:22	5A	R	AE	60 01 76 06 60 01 4A 00 B8 00 4A 00 10 00 14 00 10 00 00 00 00 00 00 00

The log data is arranged in the following columns:

- TIME: The time after the GUI startup when the packet was transmitted, sorted from newest to oldest.
- DEV: provides information about the device addressed; 5C is the global address.

- BUS: indicates the communication direction: W (write to device) and R (read from device). For the global address only W is reported, as the read is performed on virtual memory.
- ADDR: reports the starting address of the registers in question.
- DATA: shows all the data transferred in the packet, the leftmost byte starts from the address indicated in the ADDR column.

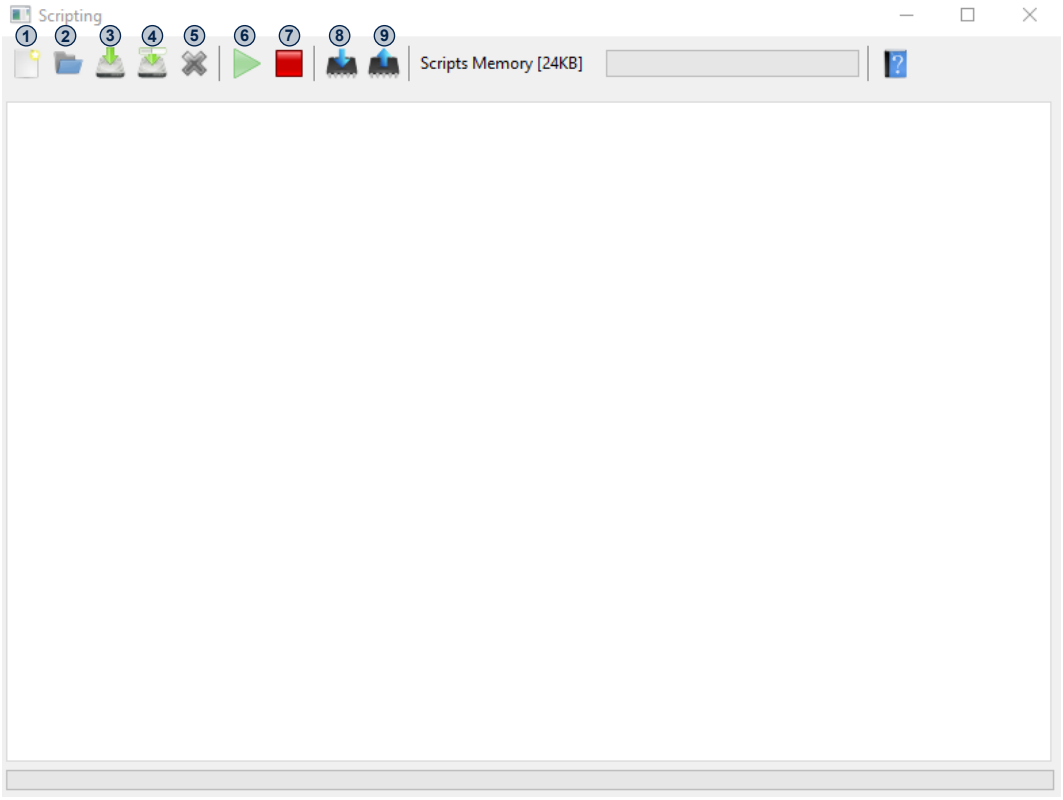
The log tracks all the commands exchanged from during the GUI session. You can clear the log window at any time by clicking on the brush icon.

2.9.3 Script

The [**Script**] button opens a new window for managing scripts.

Figure 18. Script window

1. initiates a new script in an "untitled' tab
2. browse for an existing script file ('.bas' is the default format)
3. quick save of the current script using the name on the active tab
4. opens the save as dialog box
5. closes the active tab; a prompt appears if there are unsaved changes
6. executes the script in the active tab
7. stops a running script
8. saves all the open scripts in MCU memory
9. loads all the scripts from the MCU into the script editor window



2.9.3.1 *Memory indicator*

The script window also has a memory bar that indicates the available and remaining space in the MCU script storage region.

Figure 19. Memory bar indicator

Loading scripts and adding new instructions to existing scripts are reflected in the memory indicator. If the memory limit is exceeded, the script can still be executed in the GUI environment, but you will not be able to download the scripts into MCU memory.

A progress bar in the bottom of the script window indicates the status of a download or upload operation.

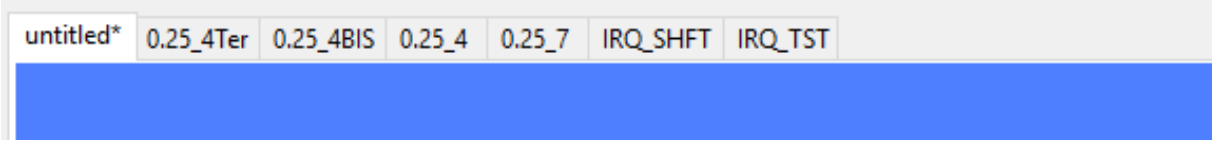
2.9.3.2 *Tabs*

Each new or loaded script is managed in individual tabs that become active when you click on them.

The name in each tab indicates the script file name, to which the default .bas extension is added during a quick save operation.

Figure 20. Multiple scripts managed by tabs

An asterisk (*) after the name indicates that the script has unsaved changes.



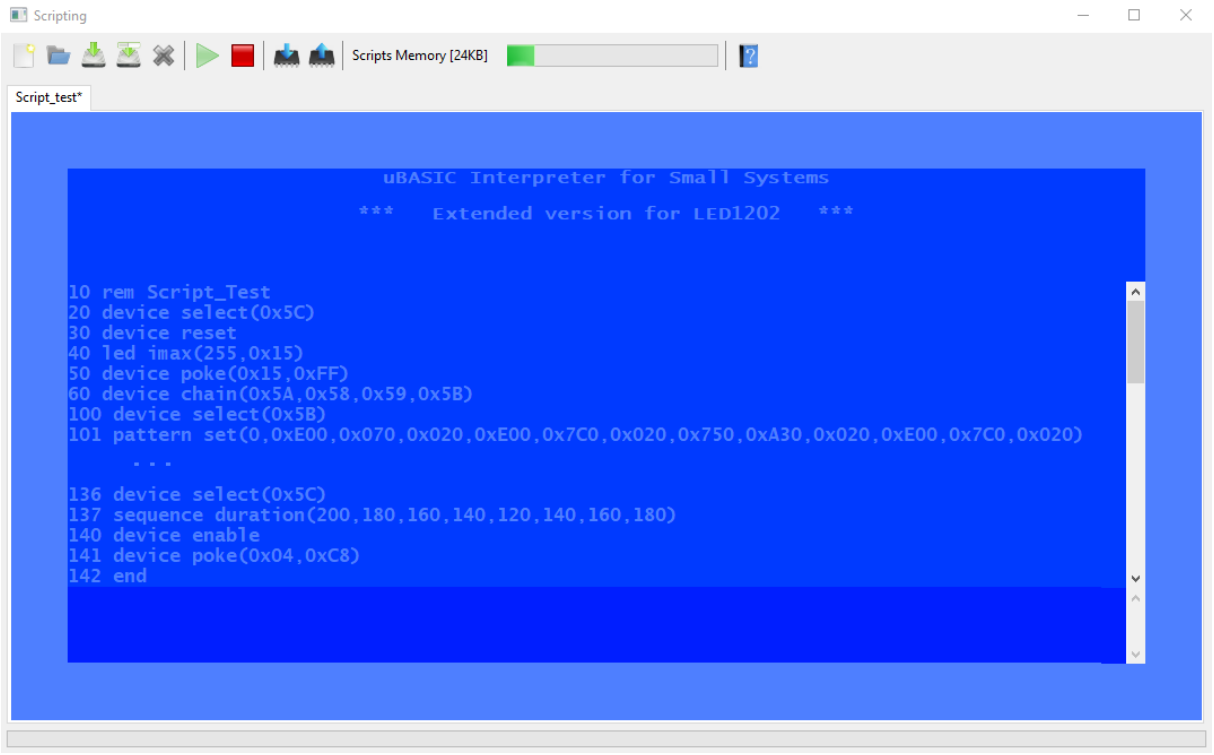
2.9.3.3 *Script editor area*

The script editor is a simple text window where each line of code must be entered entirely.

The script is based on the uBASIC interpreter, which has been extended in order to support the most typical commands for the LED1202 device. Some commands are simple register write or read operations such as pattern set or device status, others are script information such as device select, while others involve more complex routines such as device chain.

The data can be provided in decimal or in hexadecimal format.

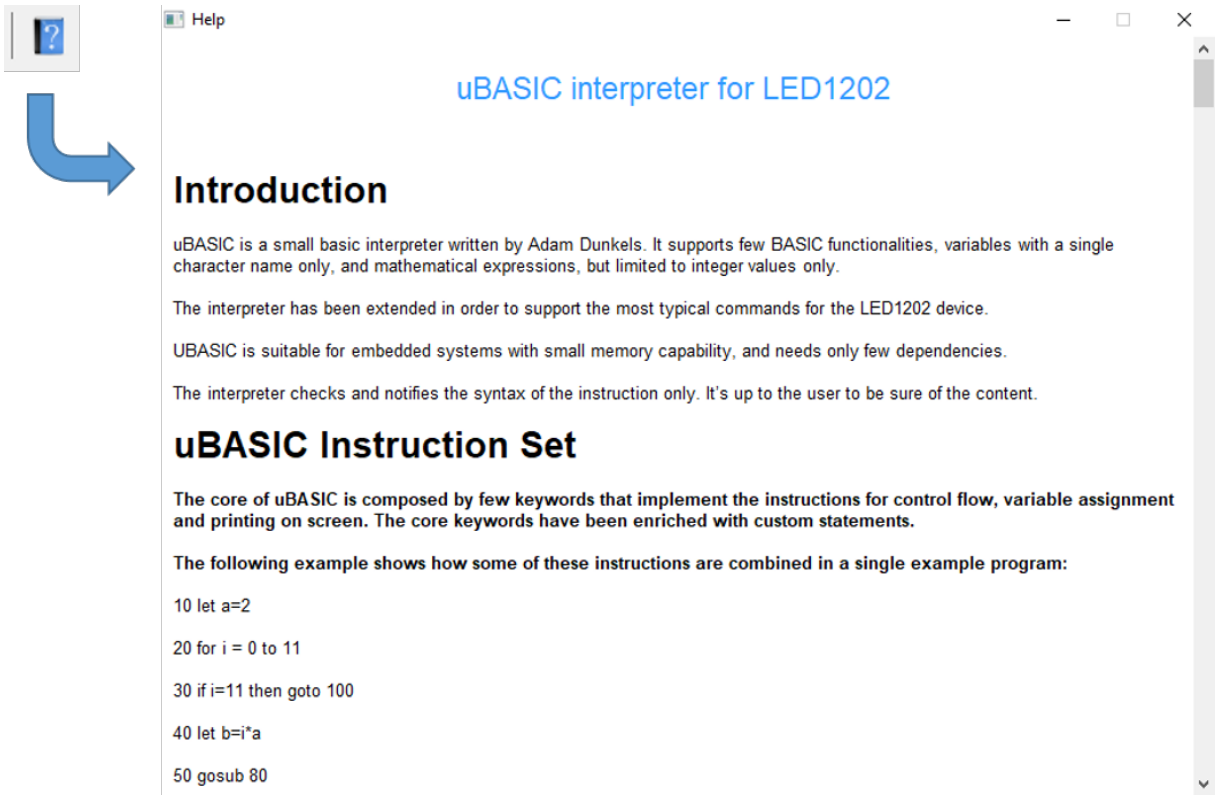
Figure 21. Script editor area



2.9.3.4 *Help*

The help button opens a file with descriptions and examples for the complete LED1202 instruction set.

Figure 22. **Help button for script command description**



Revision history

Table 1. **Document revision history**

Date	Version	Changes
17-Dec-2020	1	Initial release.

Contents

List of figures

List of tables