AN14507

Using LVGL Simulator with FreeMASTER

Rev. 1.0 — 6 January 2025

Application note

Document information

Information	Content
Keywords	AN14507, MCXA153, LVGL, GUI Guider, FreeMASTER
Abstract	This application note describes how to use GUI Guider generate a LVGL simulator integrated with FreeMASTER.

1 Introduction

This documentation describes how to prepare and set up a runtime debugging panel demo software with GUI Guider and FreeMASTER. A simple LED blink demo based on FRDM MCXA153 is created to use with FreeMASTER and GUI Guider. This demo has many parameters, such as the blink mode. The brightness can be controlled by FreeMASTER.

1.1 Requirements

The hardware requirements are as follows:

- FRDM-MCXA153
- Windows PC
- Type-C USB cable

The software requirements are as follows:

- MCUXpressoIDE v11.10.0
- Windows OS
- FreeMASTER 3.2
- GUI Guider 1.8.0

1.2 System overview

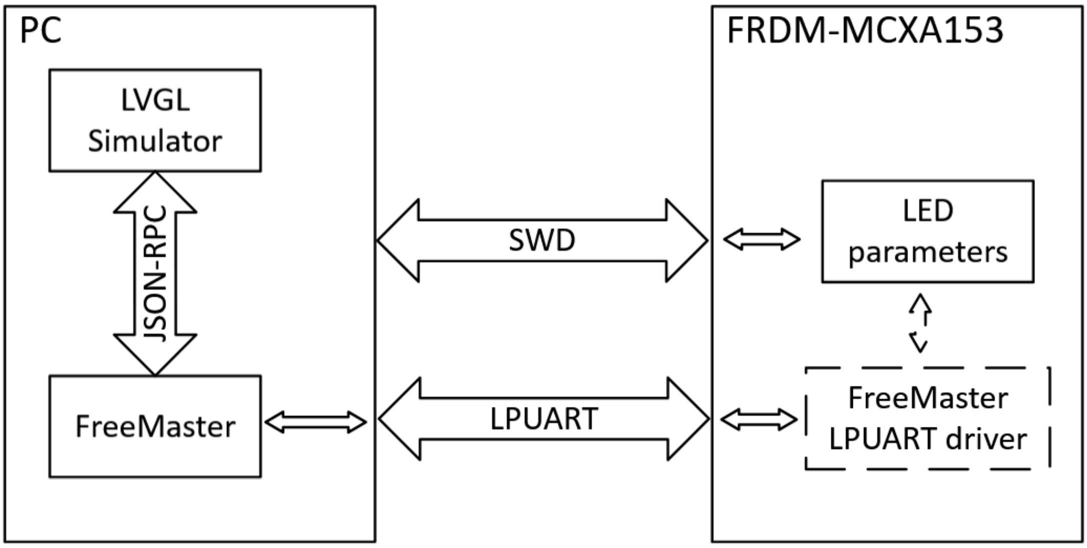


Figure 1. System overview

In this demo, the LVGL simulator and FreeMASTER run on Windows PC, and FreeMASTER communicates with the FRDM-MCXA153 board over SWD or LPUART. The LVGL simulator and FreeMASTER can modify the LED parameters on the FRDM-MCXA153 board to control the LED states. The whole system is shown in Figure 1.

In this demo, LED have 3 working modes (OFF, LOGIC, and PWM). In the OFF mode, the LED does not light up. In the LOGIC mode, the LED blinks with a given interval. In the PWM mode, the LED lights up with a given brightness. Figure 2 shows the LEDs connection of FRDM-MCXA153. To set 3 LEDs into the PWM mode, FLEXPWM and CTIMER are used to generate the PWM signals.

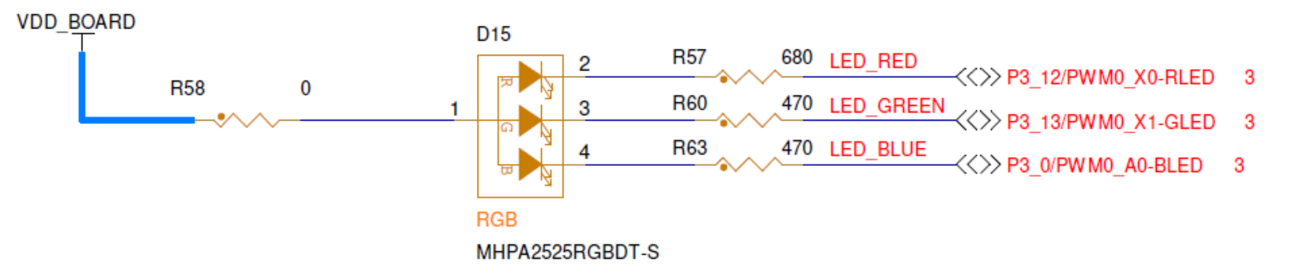


Figure 2. Onboard LEDs connection

The firmware sets up 3 FreeRTOS tasks to control LED states and monitor changes of LED parameters. If the LED mode is changed, the task reinitializes the LED to the selected mode and works following the parameters, such as the delay (in microseconds) or the PWM duty cycle.

2 Software setup

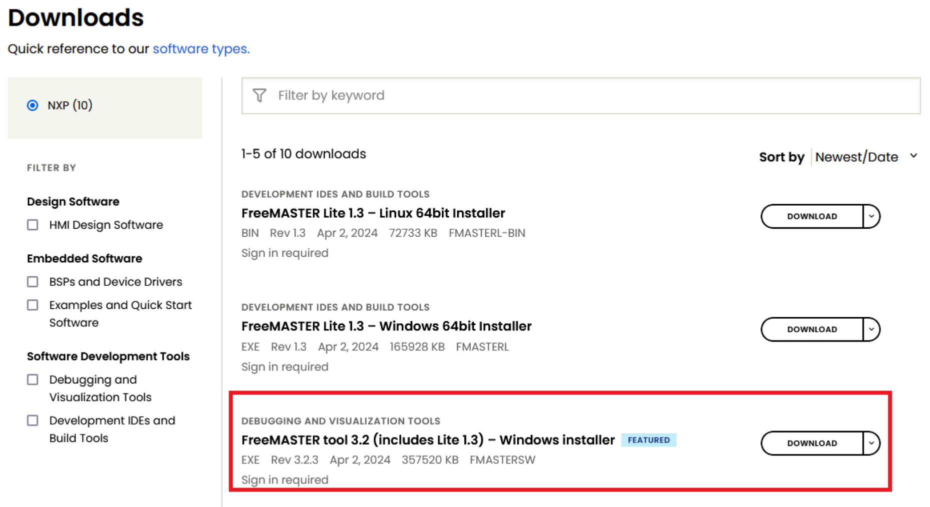
This section describes how to set up the software.

2.1 Installing FreeMASTER

FreeMASTER is a debug tool used primarily for real-time debugging, data visualization, and parameter tuning of an embedded system. FreeMASTER provides a graphical user interface for Windows users. This allows developers to monitor and control the variable of an embedded system.

Install FreeMASTER as follows:

1. Visit https://www.nxp.com/design/design-center/software/development-software/freemaster-run-time-debugging-tool:FREEMASTER and download FreeMASTER.



Downloads

Quick reference to our software types.

1-5 of 10 downloads

DEVELOPMENT IDES AND BUILD TOOLS
FreeMASTER Lite 1.3 – Linux 64bit Installer
BIN Rev 1.3 Apr 2, 2024 72733 KB FMASTERL-BIN
Sign in required

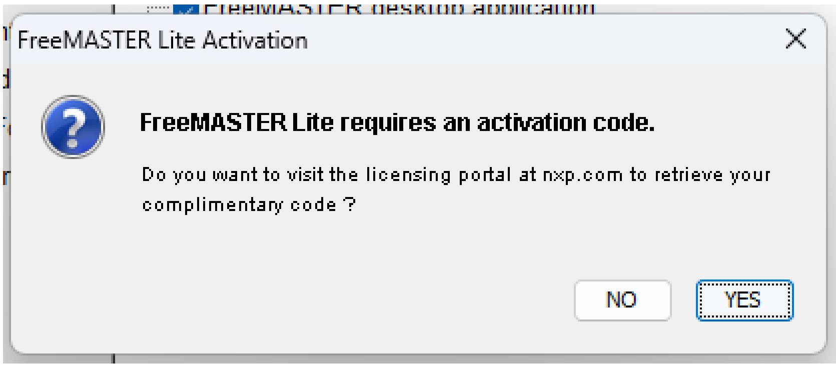
DEVELOPMENT IDES AND BUILD TOOLS
FreeMASTER Lite 1.3 – Windows 64bit Installer
EXE Rev 1.3 Apr 2, 2024 165928 KB FMASTERL
Sign in required

DEBUGGING AND VISUALIZATION TOOLS
FreeMASTER tool 3.2 (includes Lite 1.3) – Windows installer FEATURED
EXE Rev 3.2.3 Apr 2, 2024 357520 KB FMASTERSW
Sign in required

● NXP (10)
FILTER BY
Design Software
□ HMI Design Software
Embedded Software
□ BSPs and Device Drivers
□ Examples and Quick Start Software
Software Development Tools
□ Debugging and Visualization Tools
□ Development IDEs and Build Tools
Filter by keyword
Sort by Newest/Date
DOWNLOAD
DOWNLOAD
DOWNLOAD

Figure 3. Downloading FreeMASTER

2. Install the software. FreeMASTER Lite needs a license. The installation program automatically redirects to a license register page. Read and accept the Software Terms and Conditions to get a license. Enter the license when the installation program requires it.

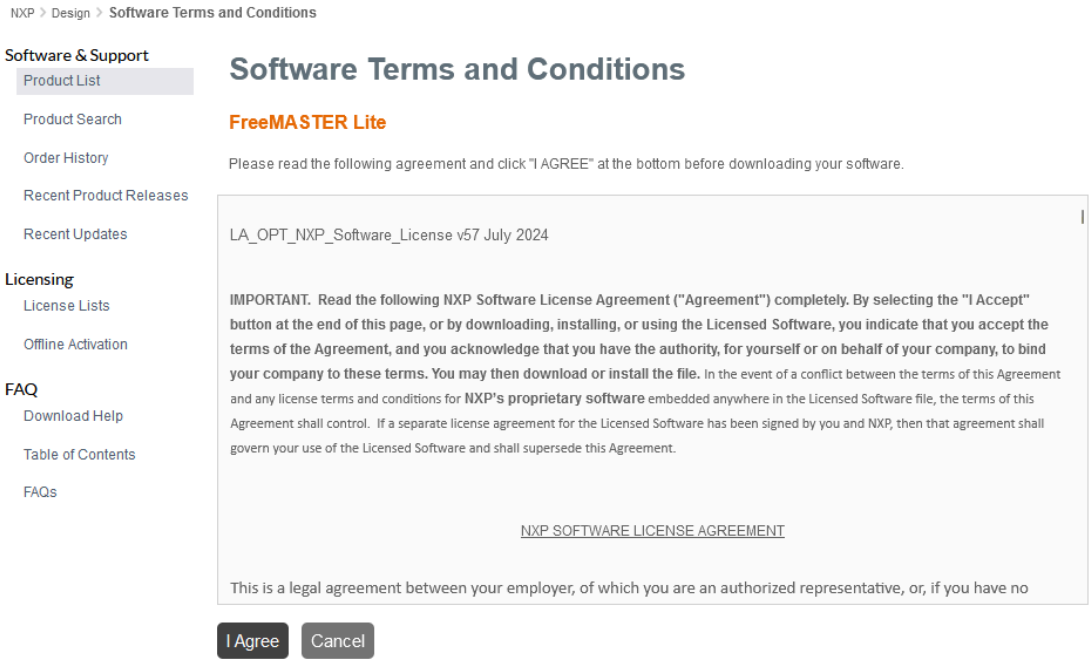


FreeMASTER Lite Activation

FreeMASTER Lite requires an activation code.

Do you want to visit the licensing portal at nxp.com to retrieve your complimentary code ?

Figure 4. Activation code notification



Software Terms and Conditions

FreeMASTER Lite

Please read the following agreement and click "I AGREE" at the bottom before downloading your software.

LA_OPT_NXP_Software_License v57 July 2024

IMPORTANT. Read the following NXP Software License Agreement ("Agreement") completely. By selecting the "I Accept" button at the end of this page, or by downloading, installing, or using the Licensed Software, you indicate that you accept the terms of the Agreement, and you acknowledge that you have the authority, for yourself or on behalf of your company, to bind your company to these terms. You may then download or install the file. In the event of a conflict between the terms of this Agreement and any license terms and conditions for NXP's proprietary software embedded anywhere in the Licensed Software file, the terms of this Agreement shall control. If a separate license agreement for the Licensed Software has been signed by you and NXP, then that agreement shall govern your use of the Licensed Software and shall supersede this Agreement.

NXP SOFTWARE LICENSE AGREEMENT

This is a legal agreement between your employer, of which you are an authorized representative, or, if you have no

Figure 5. Requesting FreeMASTER Lite license

2.2 Installing GUI Guider

GUI Guider is a graphical user interface development tool. Users can rapidly design a high-quality GUI project with GUI Guider. To install GUI Guider, visit https://www.nxp.com/design/design-center/software/development-software/gui-guider:GUI-GUIDER to download and install GUI Guider.

3 Setting up FreeMASTER on the board

FreeMASTER supports many communication interfaces, such as UART, Ethernet, and Debugger.

In this demo, users can use either an onboard debugger or LPUART. FreeMASTER is always available through the onboard debugger. Users must turn on the "OPTION_USE_FREEMASTER_SERIAL" option in the "source/main.c" file to use LPUART with FreeMASTER.

3.1 UART

FreeMASTER is configured with the MCUXpresso Config Tools in this demo. Users can check the configuration in the MCUXpresso Config Tools.

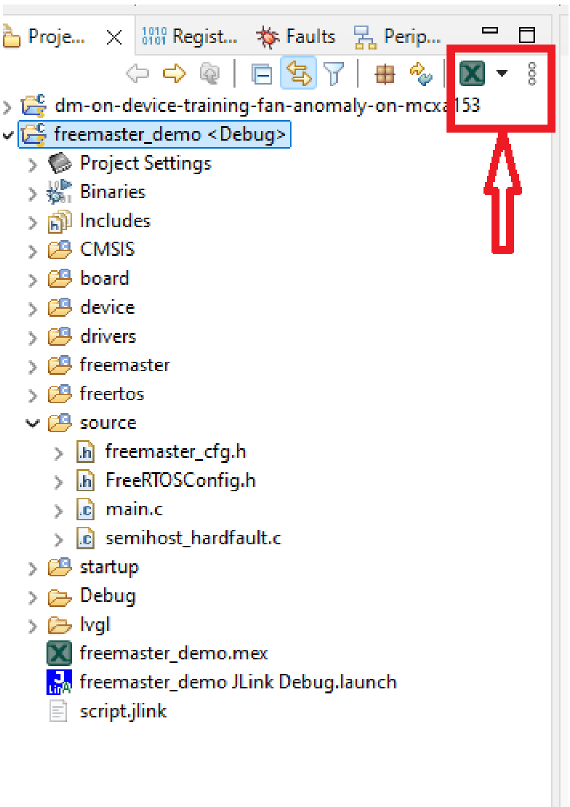
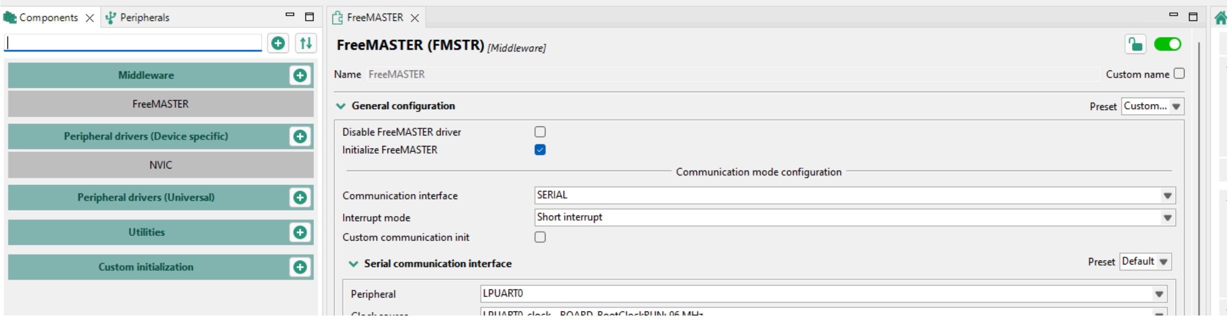


Figure 6. MCUXpresso Config Tools



FreeMASTER (FMSTR) *[Middleware]*

Name FreeMASTER   Custom name □

General configuration   Preset Custom...

Disable FreeMASTER driver □
Initialize FreeMASTER ☑

Communication mode configuration

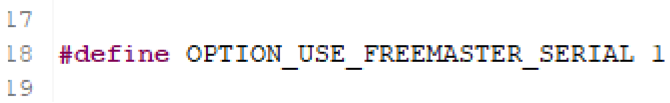
Communication interface SERIAL
Interrupt mode Short interrupt
Custom communication init □

Serial communication interface   Preset Default

Peripheral LPUART0

Figure 7. FreeMASTER in MCUXpresso Config Tools

To enable the FreeMASTER LPUART driver, turn on the "OPTION_USE_FREEMASTER" option in the "source/main.c" file. This is a software implementation and it is not generated by the MCUXpresso Config Tools. Change this definition to 1 to enable the FreeMASTER LPUART driver and change it to 0 to disable the FreeMASTER LPUART driver.



```

#define OPTION_USE_FREEMASTER_SERIAL 1

```

Figure 8. FreeMASTER LPUART option

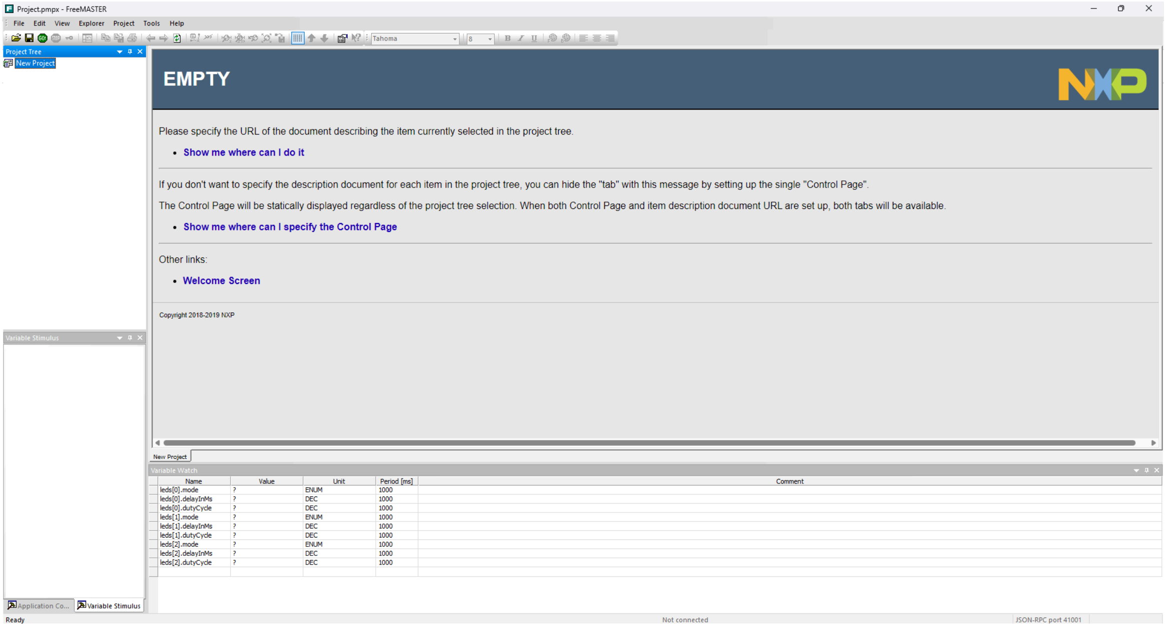
Users can also change the FreeMASTER configuration in the MCUXpresso Config Tools to enable high-level features, such as password protection and application commands.

3.2 Debugger

Users can also use the onboard debugger with FreeMASTER on FRDM-MCXA153. FreeMASTER over the debugger is always available when the debugger is connected and no user action is required. However, FreeMASTER only supports basic features, such as reading/writing memory in this way.

4 FreeMASTER project

The FreeMASTER demo project is stored in the "Debug/Project.pmpx" file. Double-click this file to open the FreeMASTER project.



EMPTY

Please specify the URL of the document describing the item currently selected in the project tree.

- Show me where can I do it

If you don't want to specify the description document for each item in the project tree, you can hide the "tab" with this message by setting up the single "Control Page".

The Control Page will be statically displayed regardless of the project tree selection. When both Control Page and item description document URL are set up, both tabs will be available.

- Show me where can I specify the Control Page

Other links:

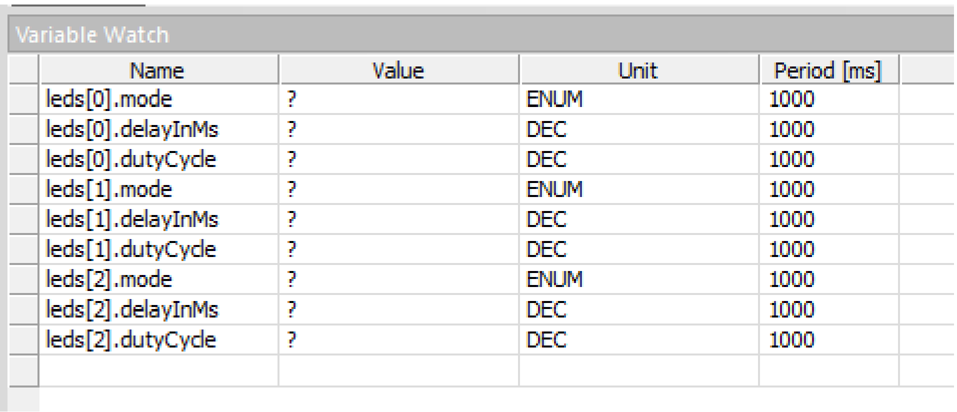
- Welcome Screen

Copyright 2018-2019 NXP

Name	Value	Unit	Period [ms]	Comment
leds[0].mode	?	ENUM	1000	
leds[0].delayInMs	?	DEC	1000	
leds[0].dutyCycle	?	DEC	1000	
leds[1].mode	?	ENUM	1000	
leds[1].delayInMs	?	DEC	1000	
leds[1].dutyCycle	?	DEC	1000	
leds[2].mode	?	ENUM	1000	
leds[2].delayInMs	?	DEC	1000	
leds[2].dutyCycle	?	DEC	1000	

Figure 9. FreeMASTER demo project

After opening this project, the "Variable Watch" window appears. This window contains the parameter variables used in the demo project. The "Value" column is "?" because FreeMASTER has not established the communication. The "Unit" column shows the variable type and the "Period" column shows the variable refresh period.

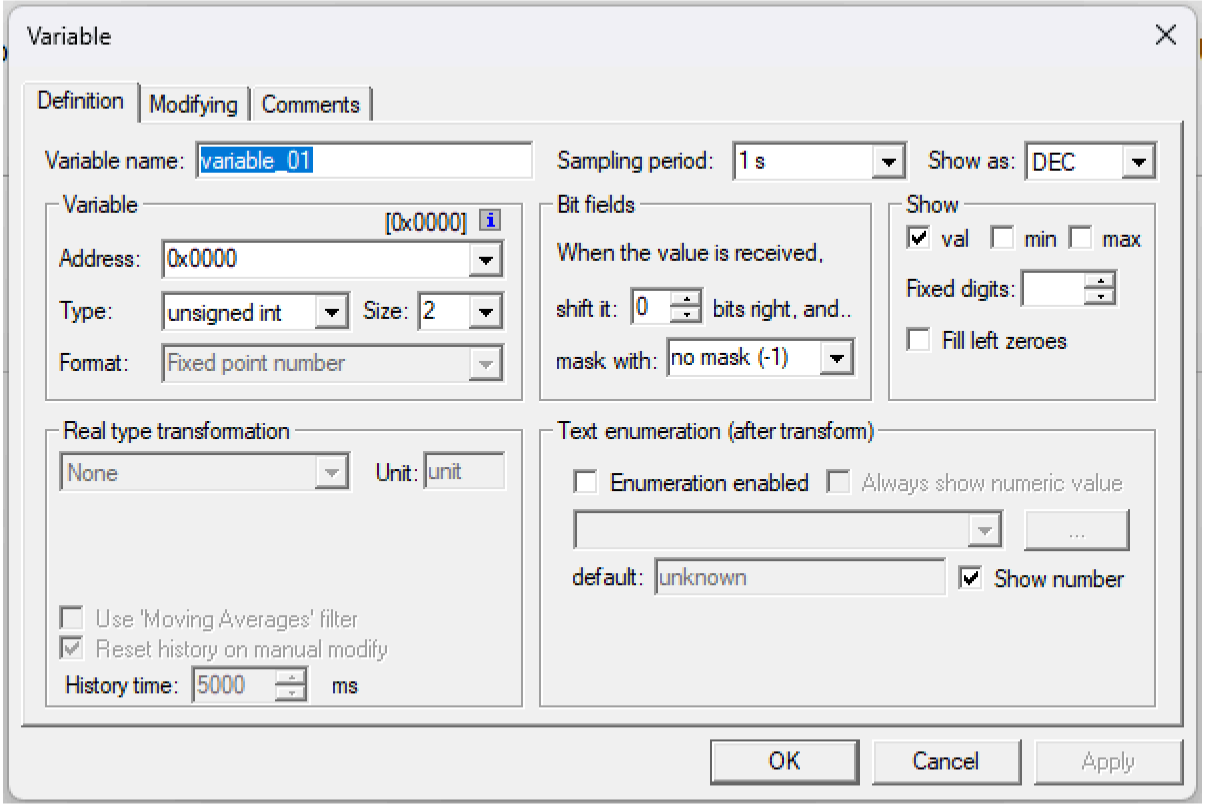


Variable Watch

Name	Value	Unit	Period [ms]
leds[0].mode	?	ENUM	1000
leds[0].delayInMs	?	DEC	1000
leds[0].dutyCycle	?	DEC	1000
leds[1].mode	?	ENUM	1000
leds[1].delayInMs	?	DEC	1000
leds[1].dutyCycle	?	DEC	1000
leds[2].mode	?	ENUM	1000
leds[2].delayInMs	?	DEC	1000
leds[2].dutyCycle	?	DEC	1000

Figure 10. Variable Watch window

Users can watch a new variable by double clicking an empty row. If a map file is loaded properly, users can directly access the wanted variable in the "Address" field. A custom variable can also be added by setting a correct address and size.



Variable

Definition | Modifying | Comments

Variable name: variable_01

Sampling period: 1 s

Show as: DEC

Variable [0x0000]

Address: 0x0000

Type: unsigned int

Size: 2

Format: Fixed point number

Bit fields

When the value is received,

shift it: 0 bits right, and..

mask with: no mask (-1)

Show

☑ val □ min □ max

Fixed digits:

□ Fill left zeroes

Real type transformation

None

Unit: unit

□ Use 'Moving Averages' filter

☑ Reset history on manual modify

History time: 5000 ms

Text enumeration (after transform)

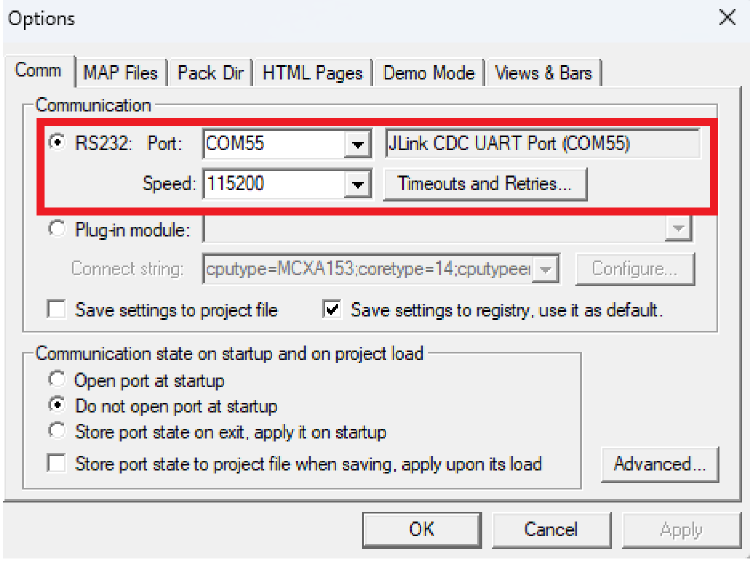
□ Enumeration enabled □ Always show numeric value

default: unknown

☑ Show number

Figure 11. Watch new variable

In this demo project, FreeMASTER uses the CMSIS-DAP onboard debugger to communicate with the FRDM-MCXA153 board if users want to use LPUART to connect to the board. Open the "Project -> Options" menu, select "RS232" and enter a correct port and speed. Click the green "GO" button or use the "Ctrl + G" shortcut to start the communication.



Options
Comm | MAP Files | Pack Dir | HTML Pages | Demo Mode | Views & Bars
OK
Cancel
Apply

Communication

● RS232: Port: COM55 JLink CDC UART Port (COM55)

Speed: 115200 Timeouts and Retries...

○ Plug-in module:

Connect string: cputype=MCXA153;coretype=14;cputypeei Configure...

□ Save settings to project file ☑ Save settings to registry, use it as default.

Communication state on startup and on project load

○ Open port at startup

● Do not open port at startup

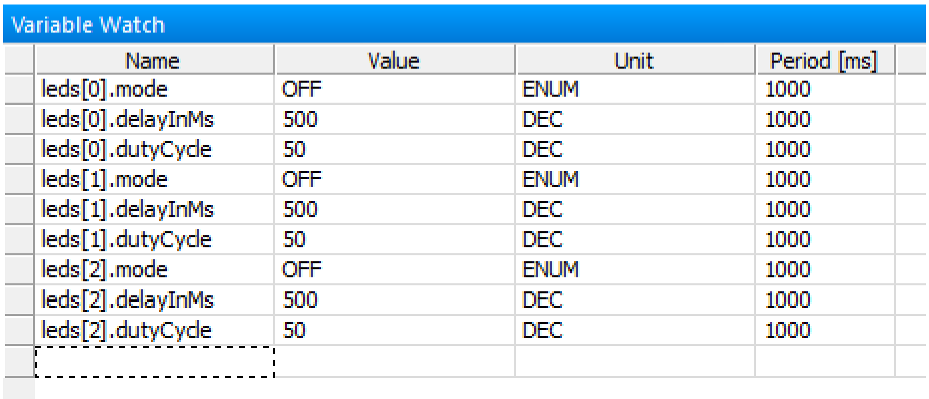
○ Store port state on exit, apply it on startup

□ Store port state to project file when saving, apply upon its load

Advanced...

Figure 12. Change communication

After the communication is stablished, the "Variable Watch" window automatically refreshes the variables.

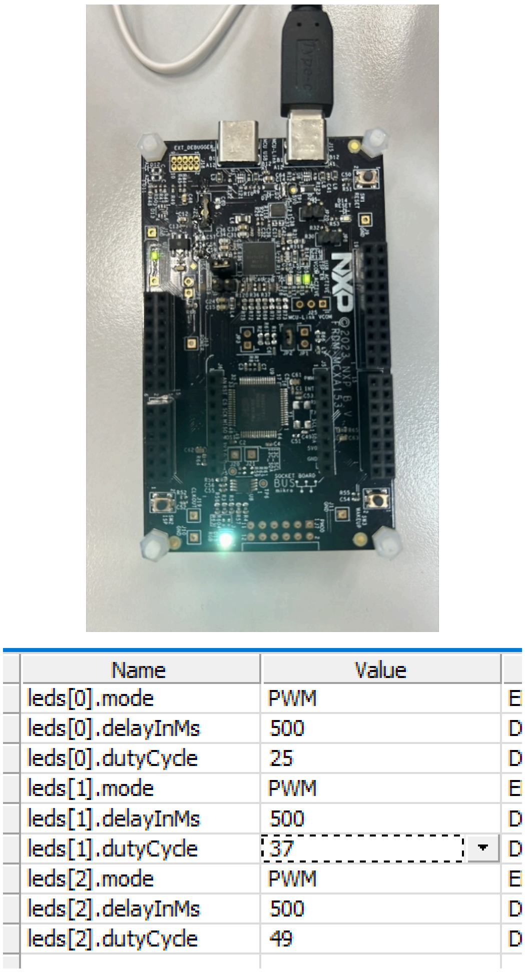


Variable Watch

Name	Value	Unit	Period [ms]
leds[0].mode	OFF	ENUM	1000
leds[0].delayInMs	500	DEC	1000
leds[0].dutyCycle	50	DEC	1000
leds[1].mode	OFF	ENUM	1000
leds[1].delayInMs	500	DEC	1000
leds[1].dutyCycle	50	DEC	1000
leds[2].mode	OFF	ENUM	1000
leds[2].delayInMs	500	DEC	1000
leds[2].dutyCycle	50	DEC	1000

Figure 13. Refreshed variables

Now you can edit these variables and the states of the onboard LEDs will change accordingly. For example, if you change the "leds[0].mode" field to "PWM", the blue LED lights up with 50 % brightness. To change the brightness, edit the "leds[0].dutyCycle" field value from 0 to 100. If the "leds[0].mode" field has the "LOGIC" value, the LED will blink with the 500-ms interval. To change the delay time, edit the "leds[0].delayInMs" field from 0 to 1000 with 10-ms steps. The other LEDs can be operated in the same way.



Name	Value	
leds[0].mode	PWM	E
leds[0].delayInMs	500	D
leds[0].dutyCycle	25	D
leds[1].mode	PWM	E
leds[1].delayInMs	500	D
leds[1].dutyCycle	37	D
leds[2].mode	PWM	E
leds[2].delayInMs	500	D
leds[2].dutyCycle	49	D

Figure 14. LEDs state

5 GUI Guider project

The GUI Guider demo project is located at "lvgl/lvgl.guiguider". To open this project, double-click this file or select this file in the "Import a local project" field.

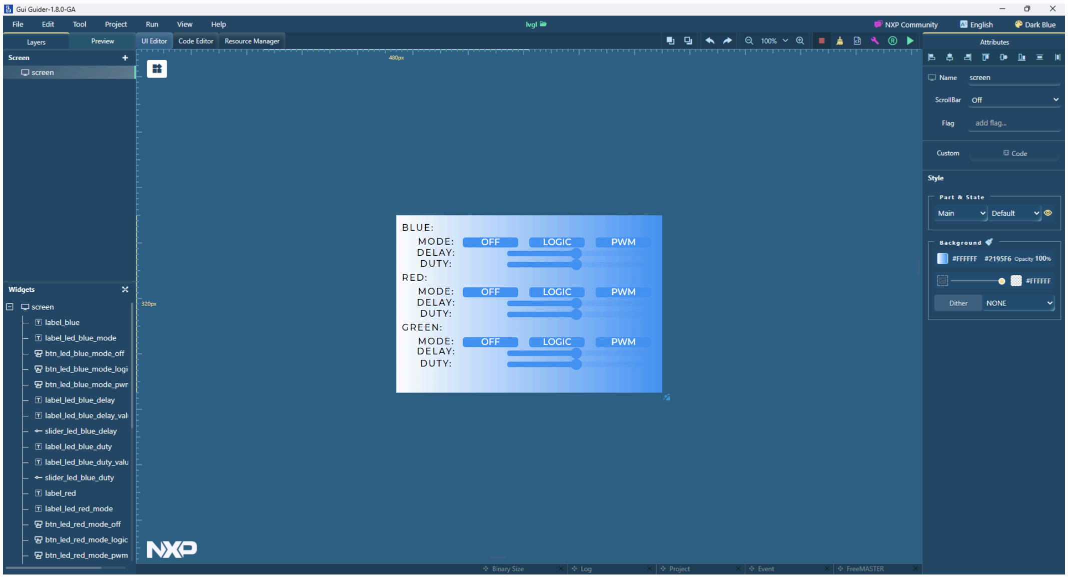


Figure 15. GUI Guider project

In this demo project, the buttons and sliders are working with FreeMASTER. The buttons can write the selected LED work mode, and the slider can change the delay interval or duty cycles. Check the 'Event" window to add or modify events. For example, the below figures show the button and slider events. The LED blue OFF mode button writes the "leds[0].mode" variable to enumerate the "OFF" value. The slider is complex. It adds custom code to the "released" event. The code gets the current slider value, multiplies it by 10 as delay intervals, and writes it to "leds[0].delayInMs". This can simply change the properties of the slider. We want a 10-ms step, so we write a custom code. You can write more complex code to fit your applications.

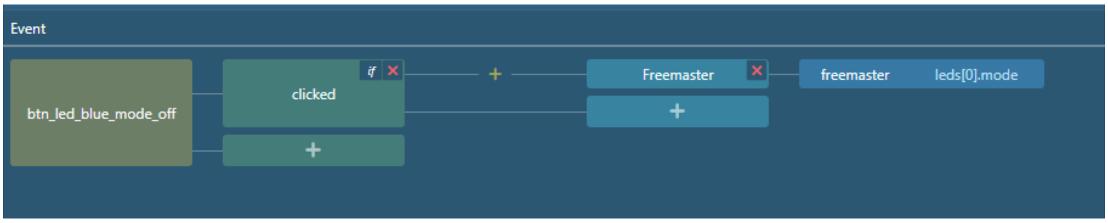
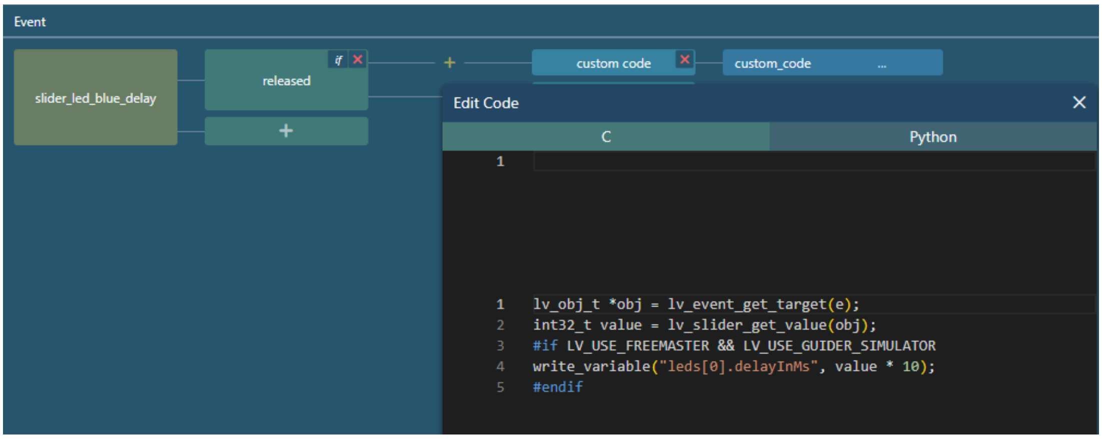


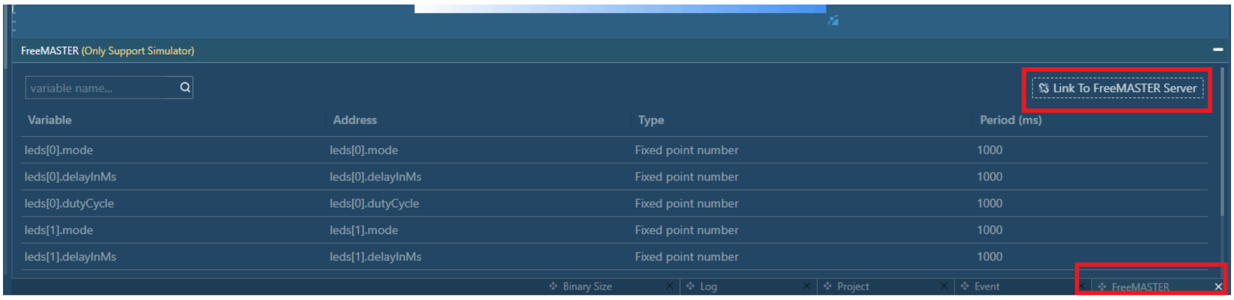
Figure 16. LED blue OFF mode button event



```
lv_obj_t *obj = lv_event_get_target(e);
int32_t value = lv_slider_get_value(obj);
#if LV_USE_FREEMASTER && LV_USE_GUIDER_SIMULATOR
write_variable("leds[0].delayInMs", value * 10);
#endif
```

Figure 17. LED blue delay interval slider event

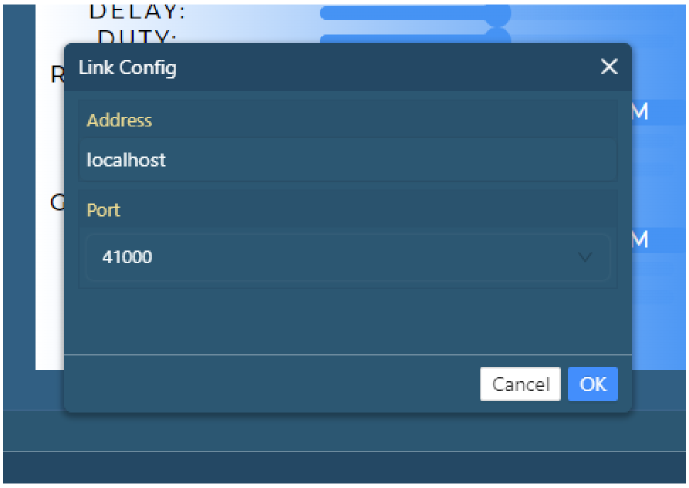
To connect FreeMASTER, open FreeMASTER and load the project as mentioned. Open the "FreeMASTER" window in the lower right corner in the GUI Guider and click the "Link To FreeMASTER Server" button. Modify the connection parameters if you changed it. Otherwise, keep the default value.



FreeMASTER (Only Support Simulator)

Variable	Address	Type	Period (ms)
leds[0].mode	leds[0].mode	Fixed point number	1000
leds[0].delayInMs	leds[0].delayInMs	Fixed point number	1000
leds[0].dutyCycle	leds[0].dutyCycle	Fixed point number	1000
leds[1].mode	leds[1].mode	Fixed point number	1000
leds[1].delayInMs	leds[1].delayInMs	Fixed point number	1000

Figure 18. FreeMASTER window



Link Config

Address: localhost

Port: 41000

Figure 19. Connecting to FreeMASTER server

After connecting to the FreeMASTER server, run the simulator with FreeMASTER. Otherwise, the simulator cannot read or write with FreeMASTER. To run the simulator, click the "Generate Code & Build & Run" button or press the "Ctrl + Q" shortcut. Run the C simulator rather than the MicroPython simulator. This project uses custom code, and it is only implemented in C.

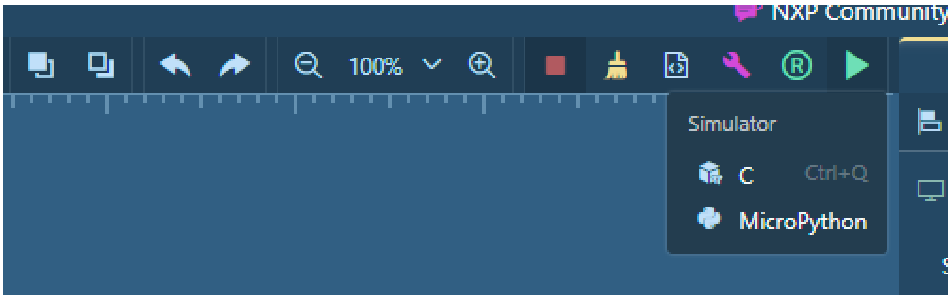
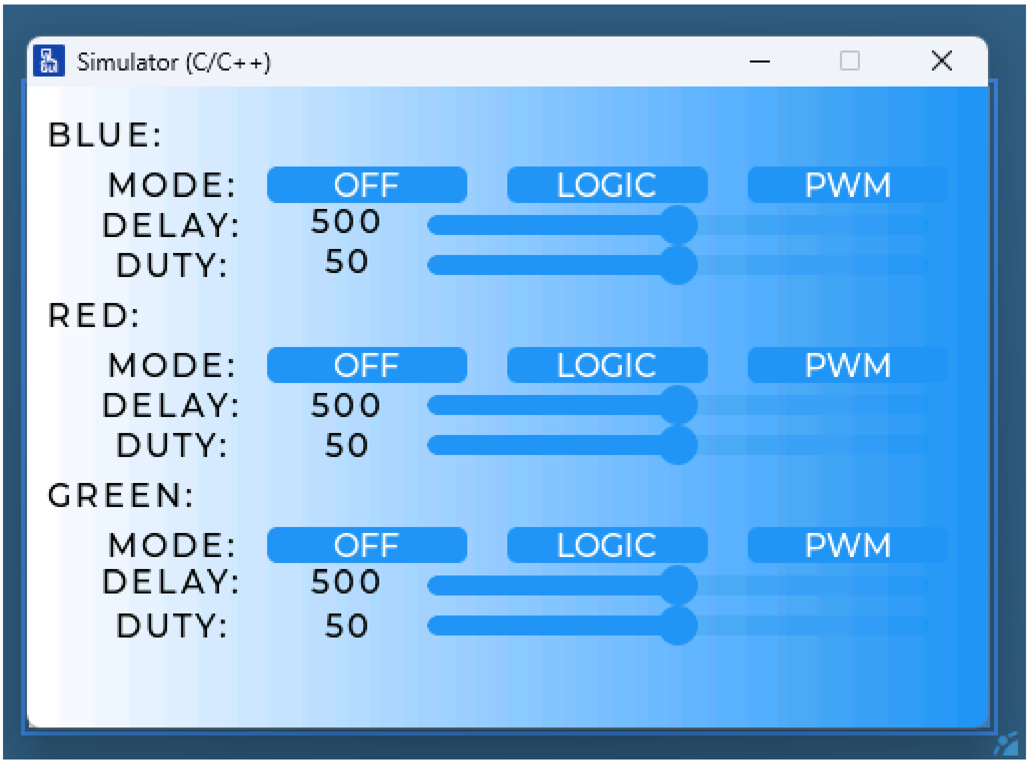


Figure 20. Running the simulator



BLUE:

MODE:

DELAY: 500

DUTY: 50

RED:

MODE:

DELAY: 500

DUTY: 50

GREEN:

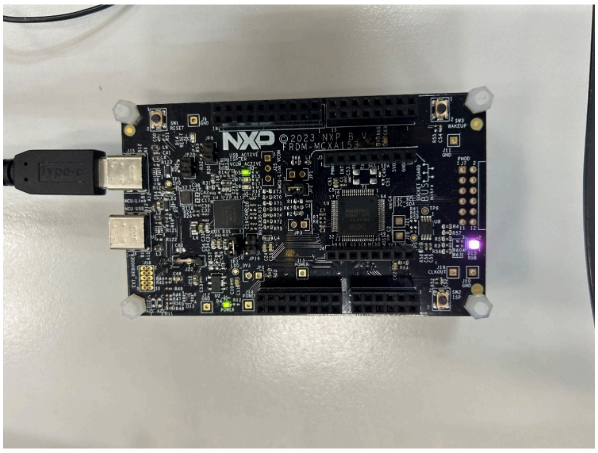
MODE:

DELAY: 500

DUTY: 50

Figure 21. Simulator options

Click the button or drag the slider to change the selected LED work states.



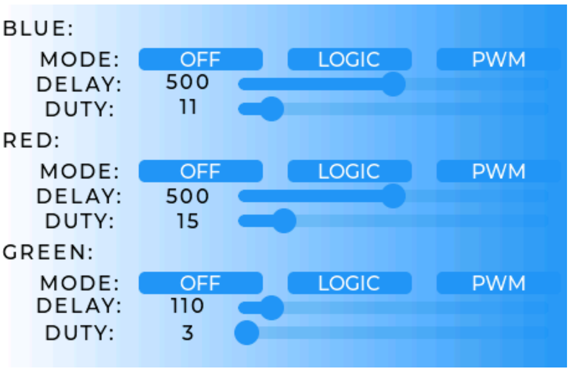


Figure 22. Demo running

6 Note about the source code in the document

Example code shown in this document has the following copyright and BSD-3-Clause license:

7 Revision history

Table 1. Revision history

Document ID	Release date	Description
AN14507 v.1.0	06 January 2024	• Initial version

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

HTML publications — An HTML version, if available, of this document is provided as a courtesy. Definitive information is contained in the applicable document in PDF format. If there is a discrepancy between the HTML document and the PDF document, the PDF document has priority.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

Amazon Web Services, AWS, the Powered by AWS logo, and FreeRTOS — are trademarks of Amazon.com, Inc. or its affiliates.

Microsoft, Azure, and ThreadX — are trademarks of the Microsoft group of companies.

Contents

Please be aware that important notices concerning this document and the product(s) described herein, have been included in section 'Legal information'.

For more information, please visit: https://www.nxp.com

Document feedback

Date of release: 6 January 2025
Document identifier: AN14507