SLN-VIZNLC-IOT-GSG

SLN-VIZNLC-IOT Getting Started Guide

Rev. 0 — 10 March 2023 User guide

Document information

Information	Content
Keywords	SLN-VIZNLC-IOT-GSG, Smart Lock, IoT
Abstract	This guide walks you through the process of getting up and running with your SLN-VIZNLC-IOT board

1 Plug it in

Welcome to the SLN-VIZNLC-IOT Getting Started Guide.

This guide walks you through the process of getting up and running with your SLN-VIZNLC-IOT board. This guide takes you through the steps of unboxing your kit, running the out-of-box smart lock demo application, as well as downloading, modifying, and debugging the firmware source code for your kit.

Before we begin, make sure to check the box your kit came in for any marks or other damages, and should you find anything, be sure to report it to your local NXP representative.

2 Smart Lock

2.1 Unbox

The box your kit arrives in should contain a few different things, including:

- A packing list paper
- Fully assembled VIZNLC kit
- USB-A > USB-C Cable (x1)
- Jumpers (x2)

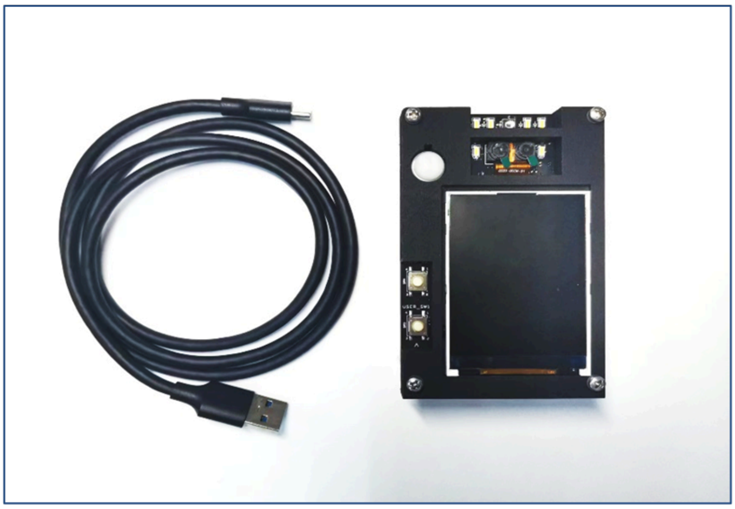


Figure 1. Items inside the box

2.2 Power on

Before we begin, put a jumper on J3 (pin 2 and 3 on the left side) as shown in the yellow highlighted box in Figure 35. Then remove the protective film from the RGB and IR camera as shown in Figure 2. This protective film is used to protect the lens of each camera during transport. However, failure to remove may cause the image capture not to work correctly.

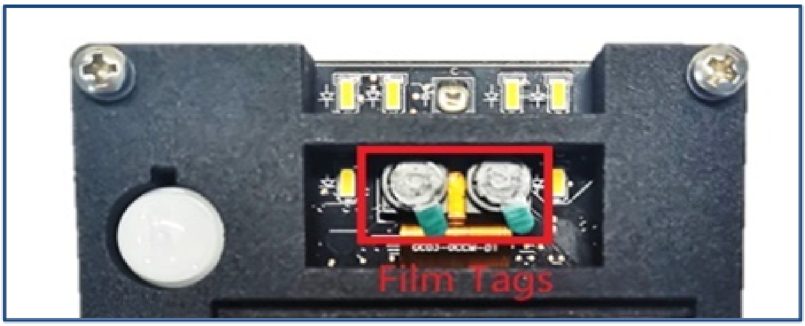


Figure 2. Protective film covering the two cameras: RGB and IR

To get started, follow the steps below:

1. Take the USB-A > USB-C cable provided inside the kit. Plug the USB-A end into USB port on your computer and the USB-C end into the USB port of the kit. The USB connector of the kit supplies power to the board and supports data transfer capabilities for Mass Storage Device (MSD) programming and virtual serial port communication.

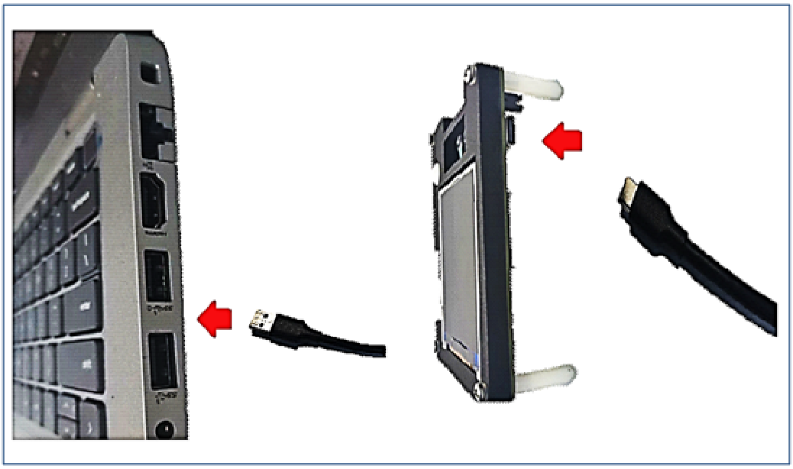


Figure 3. Power on the SLN-VIZNLC-IOT kit

2. Once the application is ready, your computer detects a new USB COM device, and automatically installs the required drivers. A message confirms when the installation is completed.
3. After powering on, the onboard TFT screen streams video directly from the RGB camera alongside a GUI overlay, providing information such as:
   - Locked/Unlocked status whether a face is recognized
   - Current App Type (Smart Lock/Access)
   - ON/OFF status of Wi-Fi and Bluetooth LE
   - Number of registered users

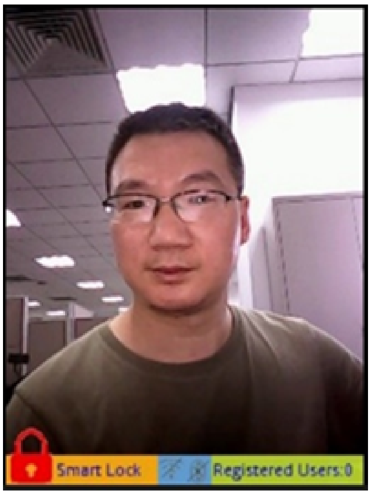


Figure 4. The screen with video preview

2.3 Register and recognize a face

Let us get started with a demonstration of the out-of-box features of this application.

To use the recognition feature of the SLN-VIZNLC-IOT, a face must be registered in the local face database of the kit. To begin registering a new face, press the **Manual Registration button (SW4)** on the kit.

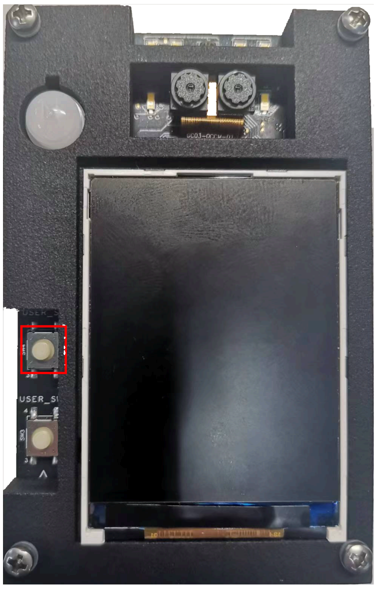


Figure 5. Manual Registration button (SW4)

Once pressed, a message indicating registration is taking place pops up at the top of the screen. The speaker plays an audio message confirming that the registration has started.

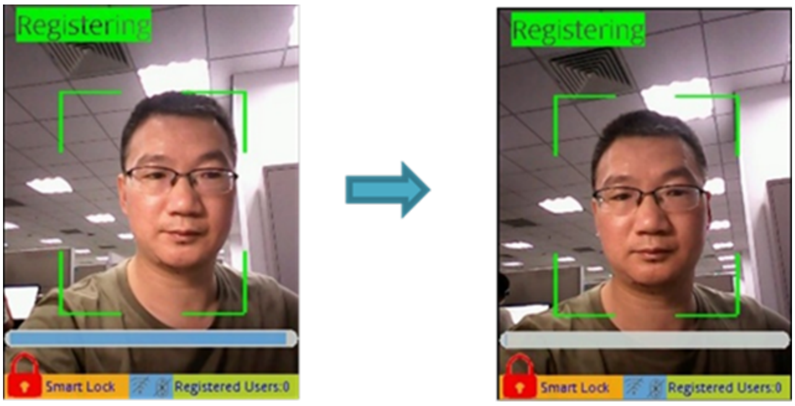


Figure 6. Registering screen

Figure 6 shows that while registration is taking place, the GUI displays:

- A “Registering” message
- Face alignment guidelines
- A countdown timer bar

The guidelines help you align your face correctly during registration, and the countdown timer bar indicates how much longer it takes until the registration process times out and fails.

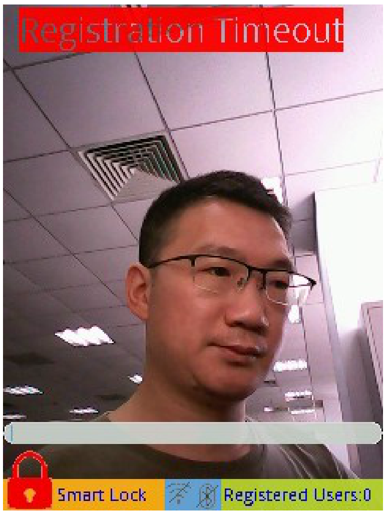
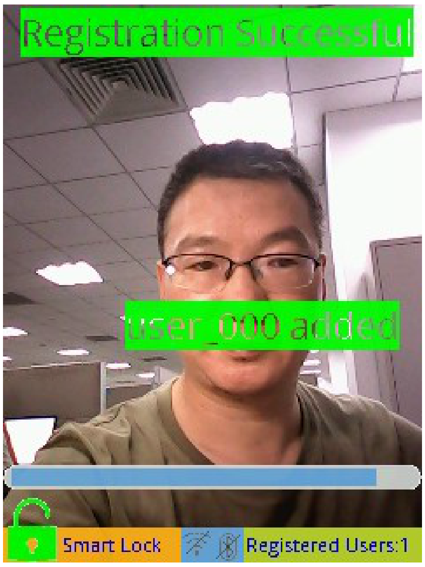


Figure 7. Registration Timeout

As an additional measure to help with registering your face, the kit even plays a warning audio prompt if too much of the side of your face is exposed during the registration process, saying “Look at Camera”. The kit plays this warning audio until your face is properly pointed toward the camera.

Should your face fails to register, simply press the **SW4** button again to retry.

Once your face is successfully registered, the kit displays the message "Registration Successful", and a unique identifier is assigned to your face. The number of registered users is also updated automatically.

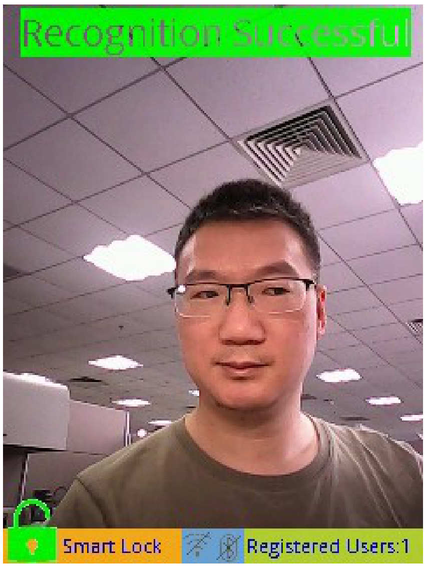


Registration Successful

user_000 added

Figure 8. Registration Successful

Once registered, the kit displays the message “Recognition Successful” and plays a corresponding audio file when a recognized face is detected.



Recognition Successful

Figure 9. Recognition Successful

2.4 Liveness detection and anti-spoofing

The Liveness detection and anti-spoofing features of the SLN-VIZNLC-IOT are switched ON by default. Therefore, enabling the system to distinguish between your actual face and a printout or phone display image of your face.

This feature helps to defend against some of the most frequent face recognition "spoof" attacks.

One such spoof attack is when a malicious actor uses a picture of someone to gain access to their face-protected materials. The malicious actor does this spoof by requiring an actual face of the user to unlock the system rather than simply a picture of their face.

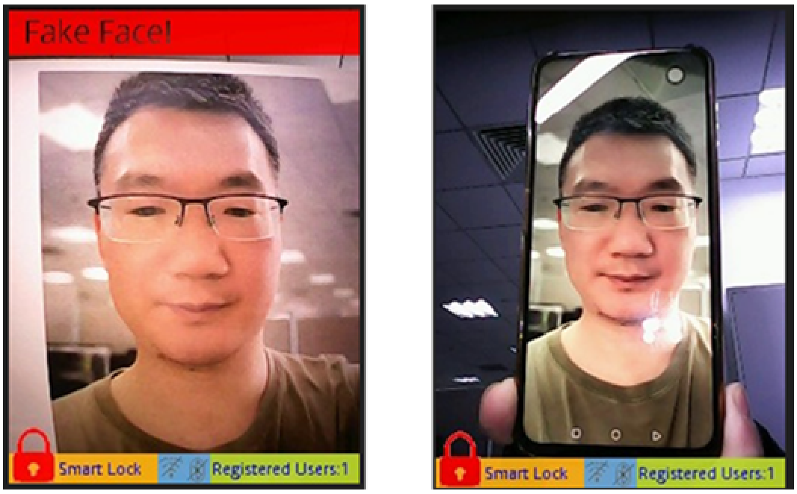


Figure 10. Printed picture and phone display spoof attack

As shown in Figure 10 using a phone display or a printed picture of a face does not trigger the “Recognition Successful” message.

2.5 Connect to serial CLI

The Smart Lock software installed by default on the SLN-VIZNLC-IOT kit provides a convenient serial-based CLI. This CLI is used to retrieve useful runtime data and configure various application settings. Connecting to the serial-based CLI of the kit can be done using a serial terminal emulator program like PuTTY or Tera Term.

Before we begin, make sure that you have a serial terminal emulator like PuTTY or Tera Term installed on your computer.

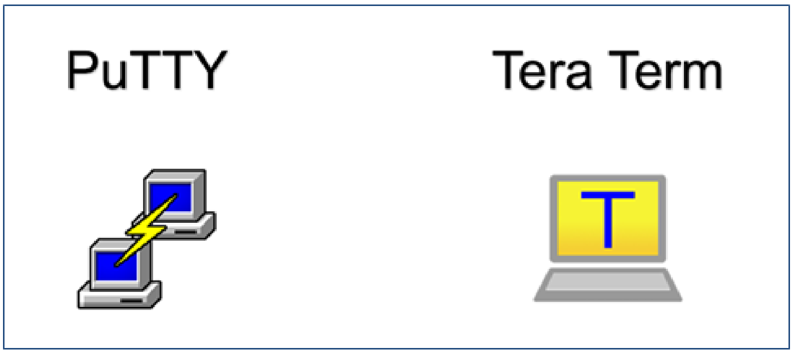
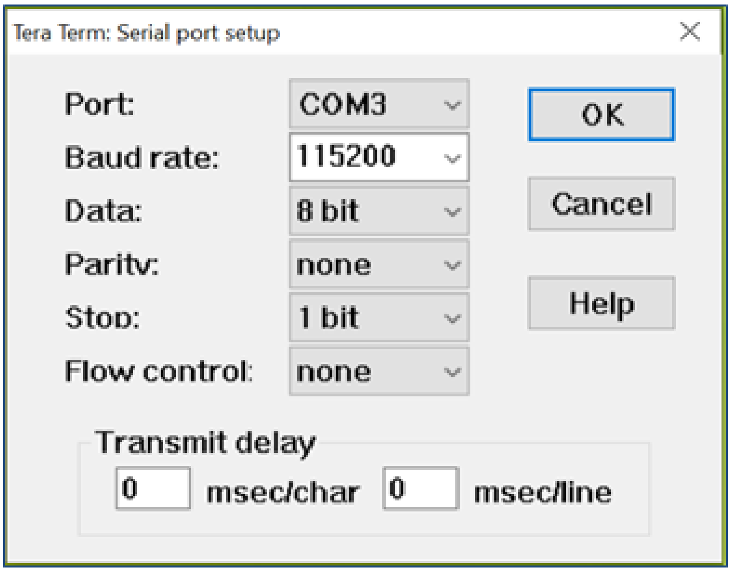


Figure 11. PuTTY and Tera Term

Note: *If you are using a Windows machine, we recommend Tera Term for its ability to discover connected COM devices automatically. Additionally, if there is a disconnect, Tera Term reconnects to a device.*

Establish a serial connection with your device by entering the serial settings as shown in Figure 12. Ensure to replace the COM port setting with the COM port associated with your kit.



Port: COM3

Baud rate: 115200

Data: 8 bit

Parity: none

Stop: 1 bit

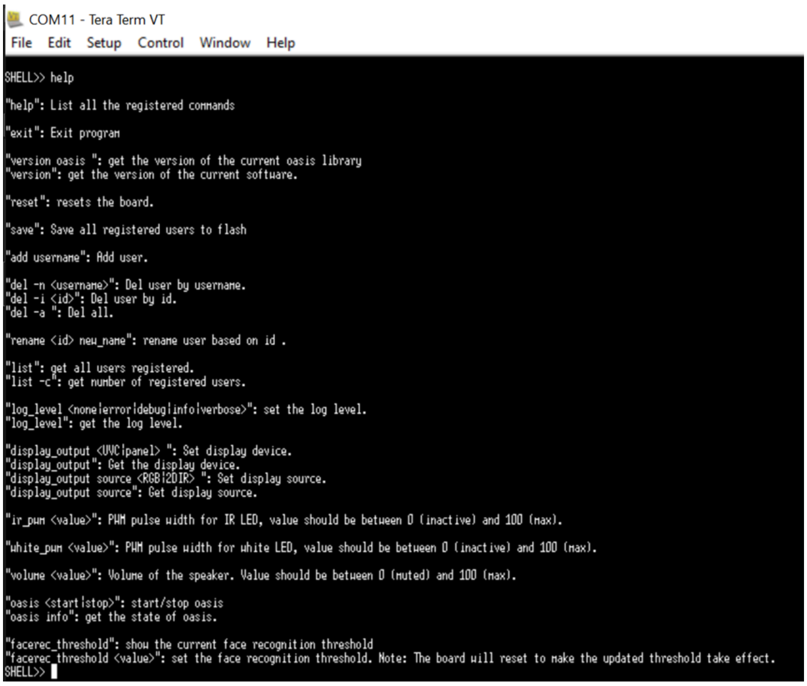
Flow control: none

Transmit delay

0 msec/char 0 msec/line

Figure 12. Serial port setup

Once connected, a blank terminal screen appears that echoes back any characters that you type. Typing the `help` command prints a list of all available commands and a brief description of their functionalities.



```
SHELL>> help

"help": List all the registered commands

"exit": Exit program

"version oasis ": get the version of the current oasis library
"version": get the version of the current software.

"reset": resets the board.

"save": Save all registered users to flash

"add username": Add user.

"del -n <username>": Del user by username.
"del -i <id>": Del user by id.
"del -a ": Del all.

"rename <id> new_name": rename user based on id .

"list": get all users registered.
"list -c": get number of registered users.

"log_level <none|error|debug|info|verbose>": set the log level.
"log_level": get the log level.

"display_output <UVC|panel> ": Set display device.
"display_output": Get the display device.
"display_output source <RGB|2DIR> ": Set display source.
"display_output source": Get display source.

"ir_pwm <value>": PWM pulse width for IR LED, value should be between 0 (inactive) and 100 (max).

"white_pwm <value>": PWM pulse width for white LED, value should be between 0 (inactive) and 100 (max).

"volume <value>": Volume of the speaker. Value should be between 0 (muted) and 100 (max).

"oasis <start|stop>": start/stop oasis
"oasis info": get the state of oasis.

"facerec_threshold": show the current face recognition threshold
"facerec_threshold <value>": set the face recognition threshold. Note: The board will reset to make the updated threshold take effect.
SHELL>>
```

Figure 13. Help menu

3   Build and run

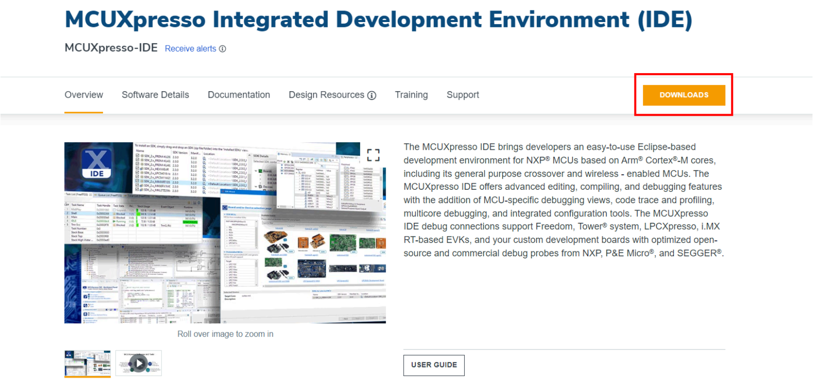
3.1 Getting MCUXpresso IDE

The MCUXpresso IDE brings developers an easy-to-use eclipse-based development environment for NXP MCUs based on Arm Cortex-M cores, including its general-purpose crossover and wireless-enabled MCUs.

The MCUXpresso IDE offers advanced editing, compiling, and debugging features. It also offers MCU-specific debugging views, code trace and profiling, multicore debugging, and integrated configuration tools.

To download MCUXpresso IDE, follow the steps below:

1. Go to MCUXpresso IDE homepage and click the **Downloads** button.



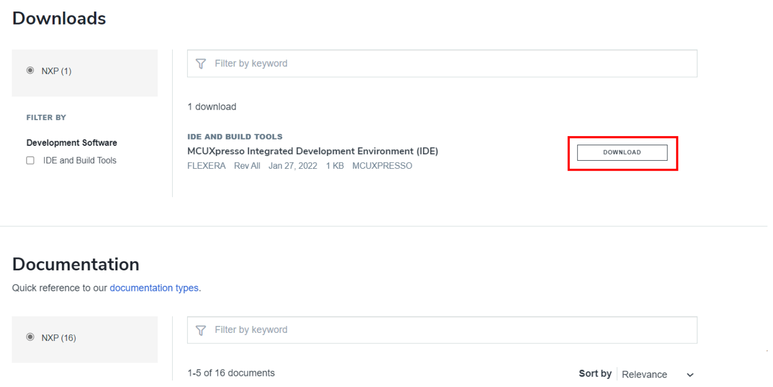
MCUXpresso Integrated Development Environment (IDE)

MCUXpresso-IDE Receive alerts

The MCUXpresso IDE brings developers an easy-to-use Eclipse-based development environment for NXP® MCUs based on Arm® Cortex®-M cores, including its general purpose crossover and wireless - enabled MCUs. The MCUXpresso IDE offers advanced editing, compiling, and debugging features with the addition of MCU-specific debugging views, code trace and profiling, multicore debugging, and integrated configuration tools. The MCUXpresso IDE debug connections support Freedom, Tower® system, LPCXpresso, i.MX RT-based EVKs, and your custom development boards with optimized open-source and commercial debug probes from NXP, P&E Micro®, and SEGGER®.

Figure 14. MCUXpresso IDE homepage

2. The **Downloads** page appears. Next, click the **Download** button.



Downloads

● NXP (1)

FILTER BY

Development Software

□ IDE and Build Tools

1 download

IDE AND BUILD TOOLS

MCUXpresso Integrated Development Environment (IDE)

FLEXERA Rev All Jan 27, 2022 1 KB MCUXPRESSO

Documentation

Quick reference to our documentation types.

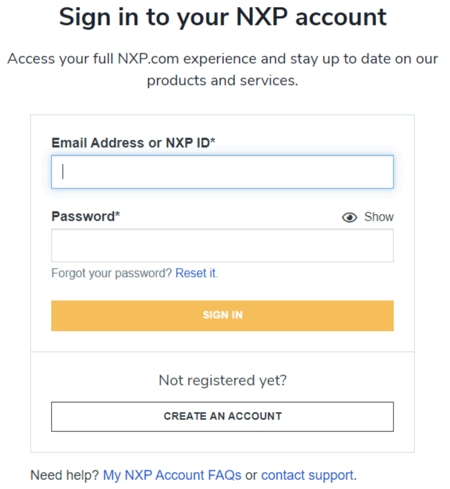
● NXP (16)

1-5 of 16 documents

Sort by Relevance

Figure 15. Downloads page

3. The **Sign In** page appears. Login with your credentials.



Sign in to your NXP account

Access your full NXP.com experience and stay up to date on our products and services.

Email Address or NXP ID*

Password*

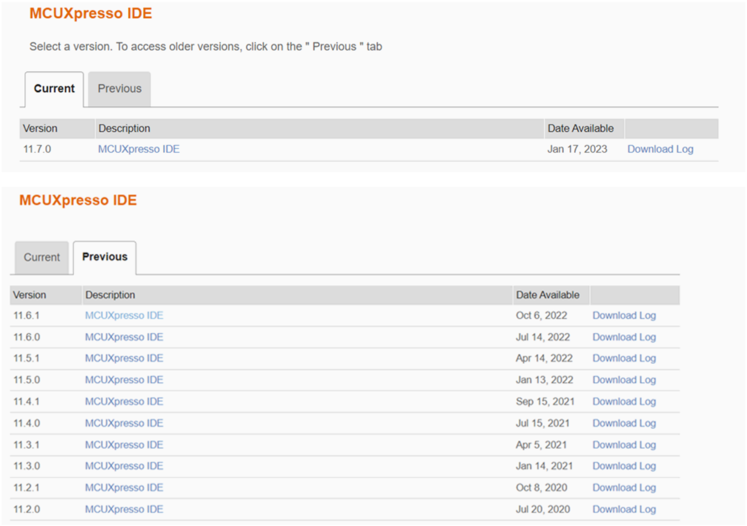
Forgot your password? Reset it.

Not registered yet?

Need help? My NXP Account FAQs or contact support.

Figure 16. Sign In page

4. Once you have signed in, select **Version 11.6.1** or newer from the list.



MCUXpresso IDE

Select a version. To access older versions, click on the " Previous " tab

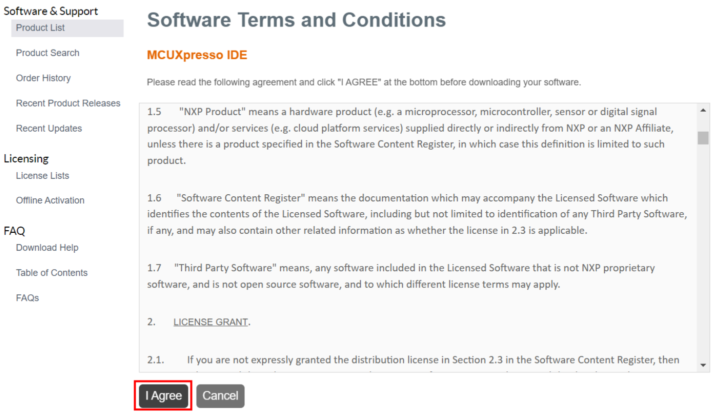
Version	Description	Date Available	
11.7.0	MCUXpresso IDE	Jan 17, 2023	Download Log

MCUXpresso IDE

Version	Description	Date Available	
11.6.1	MCUXpresso IDE	Oct 6, 2022	Download Log
11.6.0	MCUXpresso IDE	Jul 14, 2022	Download Log
11.5.1	MCUXpresso IDE	Apr 14, 2022	Download Log
11.5.0	MCUXpresso IDE	Jan 13, 2022	Download Log
11.4.1	MCUXpresso IDE	Sep 15, 2021	Download Log
11.4.0	MCUXpresso IDE	Jul 15, 2021	Download Log
11.3.1	MCUXpresso IDE	Apr 5, 2021	Download Log
11.3.0	MCUXpresso IDE	Jan 14, 2021	Download Log
11.2.1	MCUXpresso IDE	Oct 8, 2020	Download Log
11.2.0	MCUXpresso IDE	Jul 20, 2020	Download Log

Figure 17. Select Version page

5. The **Software Terms and Conditions** page appears. Read the conditions and click the **I Agree** button.



Software Terms and Conditions

MCUXpresso IDE

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

1.5 "NXP Product" means a hardware product (e.g. a microprocessor, microcontroller, sensor or digital signal processor) and/or services (e.g. cloud platform services) supplied directly or indirectly from NXP or an NXP Affiliate, unless there is a product specified in the Software Content Register, in which case this definition is limited to such product.

1.6 "Software Content Register" means the documentation which may accompany the Licensed Software which identifies the contents of the Licensed Software, including but not limited to identification of any Third Party Software, if any, and may also contain other related information as whether the license in 2.3 is applicable.

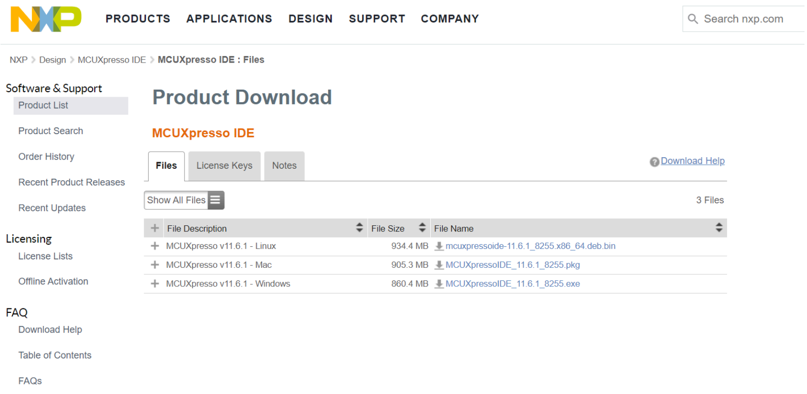
1.7 "Third Party Software" means, any software included in the Licensed Software that is not NXP proprietary software, and is not open source software, and to which different license terms may apply.

2. LICENSE GRANT.

2.1. If you are not expressly granted the distribution license in Section 2.3 in the Software Content Register, then

Figure 18. Software terms and conditions page

6. The **Product Download** page appears from where you can download the MCUXpresso IDE. Download the appropriate version for your system.



Product Download

MCUXpresso IDE

3 Files

File Description	File Size	File Name
MCUXpresso v11.6.1 - Linux	934.4 MB	mcuxpressoide-11.6.1_8255.x86_64.deb.bin
MCUXpresso v11.6.1 - Mac	905.3 MB	MCUXpressoIDE_11.6.1_8255.pkg
MCUXpresso v11.6.1 - Windows	860.4 MB	MCUXpressoIDE_11.6.1_8255.exe

Figure 19. Product Download page

7. Open the downloaded application and follow the instructions found in the installer.

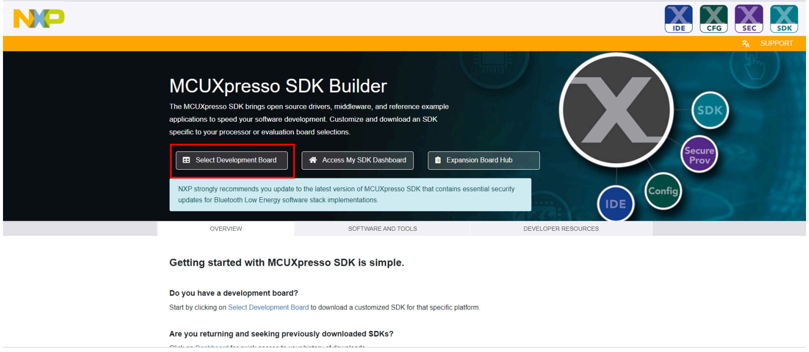
3.2 Installing the SDK

MCUXpresso SDK is a comprehensive software enablement package designed to simplify and accelerate application development with NXP microcontrollers based on Arm Cortex-M cores. The MCUXpresso SDK includes production-grade software with integrated RTOS (optional), stacks and middleware, reference software, and more. It is available in custom downloads based on user selections of MCU, evaluation board, and optional software components.

Before building the SLN-VIZNLC-IOT SDK example projects, the target SDKs (EVK-MIMXRT1060 and K32W061DK6) must be imported into MCUXpresso IDE. However, no need to import LPC845 as it is already preinstalled in the IDE.

To install MCUXpresso SDK, follow the steps below:

1. Launch the MCUXpresso SDK. The **MCUXpresso SDK Builder** welcome screen appears when the application is launched, as shown in Figure 20.



MCUXpresso SDK Builder

The MCUXpresso SDK brings open source drivers, middleware, and reference example applications to speed your software development. Customize and download an SDK specific to your processor or evaluation board selections.

NXP strongly recommends you update to the latest version of MCUXpresso SDK that contains essential security updates for Bluetooth Low Energy software stack implementations.

Getting started with MCUXpresso SDK is simple.

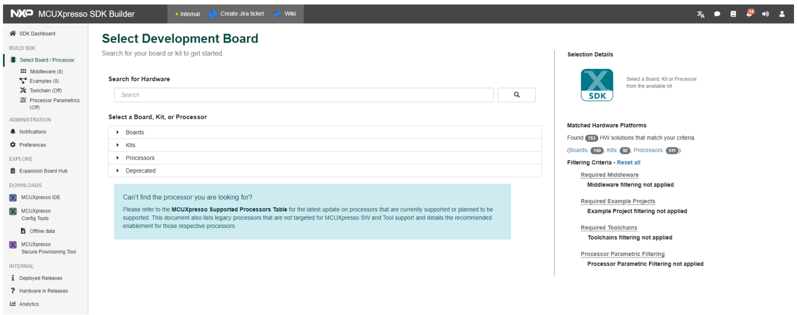
Do you have a development board?

Start by clicking on Select Development Board to download a customized SDK for that specific platform.

Are you returning and seeking previously downloaded SDKs?

Figure 20. SDK Builder welcome screen

2. Click **Select Development Board**. After signing in, **Select Development Board** page appears.



Select Development Board

Search for your board or kit to get started.

Can't find the processor you are looking for?

Please refer to the **MCUXpresso Supported Processors Table** for the latest update on processors that are currently supported or planned to be supported. This document also lists legacy processors that are not targeted for MCUXpresso SW and Tool support and details the recommended enablement for those respective processors.

Selection Details

Select a Board, Kit or Processor from the available kit

Matched Hardware Platforms

Found 753 HW solutions that match your criteria.

(Boards: 150, Kits: 92, Processors: 511)

Filtering Criteria - Reset all

Required Middleware

Middleware filtering not applied

Required Example Projects

Example Project filtering not applied

Required Toolchains

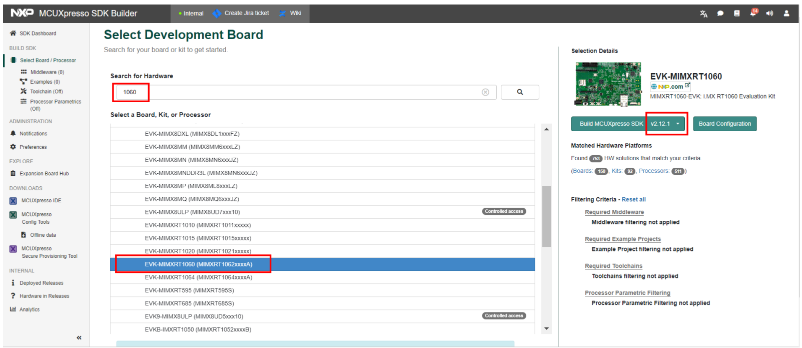
Toolchains filtering not applied

Processor Parametric Filtering

Processor Parametric Filtering not applied

Figure 21. Select Development Board page

3. Search **1060** and select **EVK-MIMXRT1060** board. Then click **Build MCUXpresso SDK V2.12.1**.



Select Development Board

Search for your board or kit to get started.

Search for Hardware: 1060

Select a Board, Kit, or Processor

- EVK-MIMX8DXL (MIMX8DL1xxxFZ)
- EVK-MIMX8MM (MIMX8MM6xxxLZ)
- EVK-MIMX8MN (MIMX8MN6xxxJZ)
- EVK-MIMX8MNDDR3L (MIMX8MN6xxxJZ)
- EVK-MIMX8MP (MIMX8ML8xxxLZ)
- EVK-MIMX8MQ (MIMX8MQ6xxxJZ)
- EVK-MIMX8ULP (MIMX8UD7xxx10) Controlled access
- EVK-MIMXRT1010 (MIMXRT1011xxxxx)
- EVK-MIMXRT1015 (MIMXRT1015xxxxx)
- EVK-MIMXRT1020 (MIMXRT1021xxxxx)
- EVK-MIMXRT1060 (MIMXRT1062xxxxA)
- EVK-MIMXRT1064 (MIMXRT1064xxxxA)
- EVK-MIMXRT595 (MIMXRT595S)
- EVK-MIMXRT685 (MIMXRT685S)
- EVK9-MIMX8ULP (MIMX8UD5xxx10) Controlled access
- EVKB-IMXRT1050 (MIMXRT1052xxxxB)

Selection Details

EVK-MIMXRT1060
NXP.com
MIMXRT1060-EVK: i.MX RT1060 Evaluation Kit

Matched Hardware Platforms

Found 753 HW solutions that match your criteria.
(Boards: 150, Kits: 92, Processors: 511)

Filtering Criteria - Reset all

Required Middleware
Middleware filtering not applied

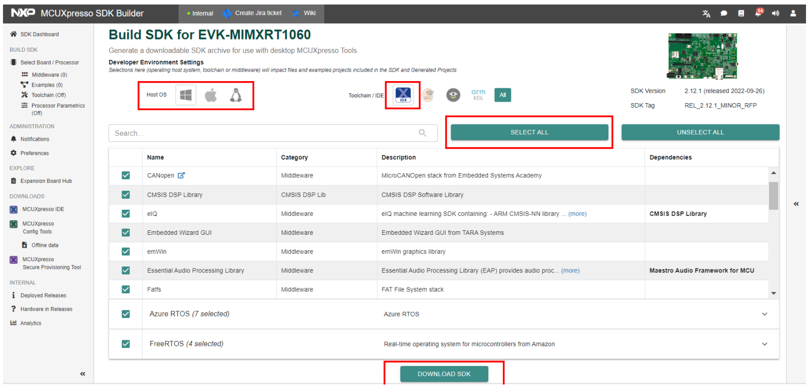
Required Example Projects
Example Project filtering not applied

Required Toolchains
Toolchains filtering not applied

Processor Parametric Filtering
Processor Parametric Filtering not applied

Figure 22. Select EVK-MIMXRT1060 SDK

4. To build SDK, select your **Host OS** and **Toolchain / IDE**, and all other necessary SDK components. Then click **Download SDK** button.



Build SDK for EVK-MIMXRT1060

Generate a downloadable SDK archive for use with desktop MCUXpresso Tools.

Developer Environment Settings

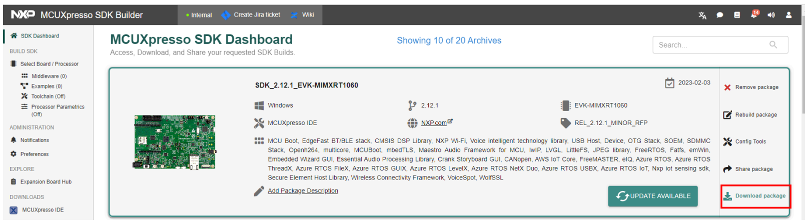
Selections here (operating host system, toolchain or middleware) will impact files and examples projects included in the SDK and Generated Projects

SDK Version: 2.12.1 (released 2022-09-26)
SDK Tag: REL_2.12.1_MINOR_RFP

	Name	Category	Description	Dependencies
☑	CANopen	Middleware	MicroCANOpen stack from Embedded Systems Academy	
☑	CMSIS DSP Library	CMSIS DSP Lib	CMSIS DSP Software Library	
☑	eIQ	Middleware	eIQ machine learning SDK containing: - ARM CMSIS-NN library ... (more)	CMSIS DSP Library
☑	Embedded Wizard GUI	Middleware	Embedded Wizard GUI from TARA Systems	
☑	emWin	Middleware	emWin graphics library	
☑	Essential Audio Processing Library	Middleware	Essential Audio Processing Library (EAP) provides audio proc... (more)	Maestro Audio Framework for MCU
☑	Fatfs	Middleware	FAT File System stack	
☑	Azure RTOS *(7 selected)*		Azure RTOS	
☑	FreeRTOS *(4 selected)*		Real-time operating system for microcontrollers from Amazon	

Figure 23. Build SDK and download

5. Dashboard page shows the built SDK. Click **Download** to download the SDK archive to your PC.



MCUXpresso SDK Dashboard

Access, Download, and Share your requested SDK Builds.

Showing 10 of 20 Archives

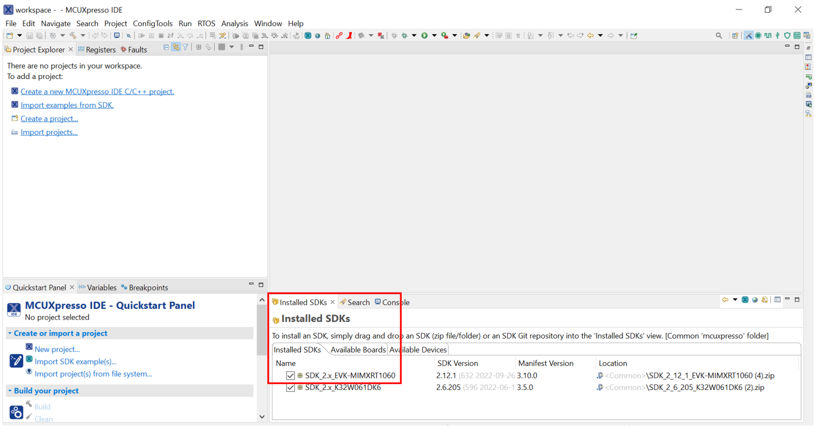
SDK_2.12.1_EVK-MIMXRT1060 2023-02-03

Windows 2.12.1 EVK-MIMXRT1060
MCUXpresso IDE NXP.com REL_2.12.1_MINOR_RFP

MCU Boot, EdgeFast BT/BLE stack, CMSIS DSP Library, NXP Wi-Fi, Voice intelligent technology library, USB Host, Device, OTG Stack, SOEM, SDMMC Stack, Openh264, multicore, MCUBoot, mbedTLS, Maestro Audio Framework for MCU, lwIP, LVGL, LittleFS, JPEG library, FreeRTOS, Fatfs, emWin, Embedded Wizard GUI, Essential Audio Processing Library, Crank Storyboard GUI, CANopen, AWS IoT Core, FreeMASTER, eIQ, Azure RTOS, Azure RTOS ThreadX, Azure RTOS FileX, Azure RTOS GUIX, Azure RTOS LevelX, Azure RTOS NetX Duo, Azure RTOS USBX, Azure RTOS IoT, Nxp iot sensing sdk, Secure Element Host Library, Wireless Connectivity Framework, VoiceSpot, WolfSSL

Figure 24. Downloading SDK archive

6. To install, drag and drop the SDK archive file to the IDE **Installed SDKs** view.



workspace - - MCUXpresso IDE
File Edit Navigate Search Project ConfigTools Run RTOS Analysis Window Help
Project Explorer
Registers
Faults
Quickstart Panel
Variables
Breakpoints
MCUXpresso IDE - Quickstart Panel
No project selected
Create or import a project
New project...
Import SDK example(s)...
Import project(s) from file system...
Build your project
Build
Clean
Installed SDKs
Search
Console

There are no projects in your workspace.
To add a project:

- Create a new MCUXpresso IDE C/C++ project.
- Import examples from SDK.
- Create a project...
- Import projects...

Installed SDKs

To install an SDK, simply drag and drop an SDK (zip file/folder) or an SDK Git repository into the 'Installed SDKs' view. [Common 'mcuxpresso' folder]

Installed SDKs | Available Boards | Available Devices

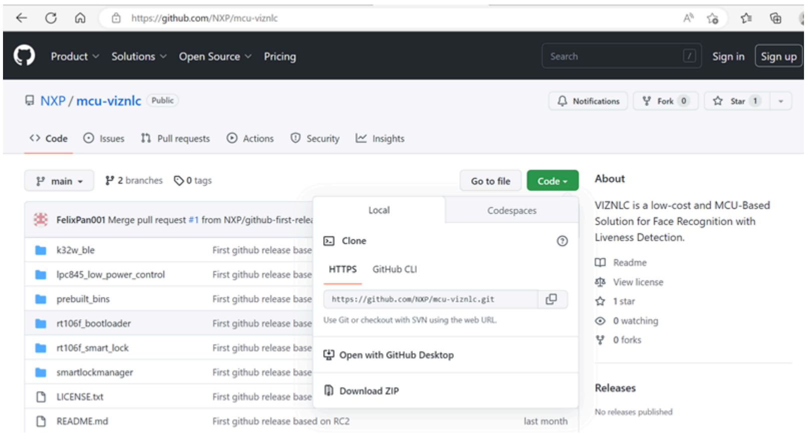
Name	SDK Version	Manifest Version	Location
☑ SDK_2.x_EVK-MIMXRT1060	2.12.1 (632 2022-09-26	3.10.0	<Common>\SDK_2_12_1_EVK-MIMXRT1060 (4).zip
☑ SDK_2.x_K32W061DK6	2.6.205 (596 2022-06-1	3.5.0	<Common>\SDK_2_6_205_K32W061DK6 (2).zip

Figure 25. Install SDK

7. Download and install K32W061DK6 SDK V2.6.205 in the same way.

3.3 Downloading SLN-VIZNLC-IOT projects

The SLN-VIZNLC-IOT out-of-box projects are published under the NXP GitHub page. You can either clone the repository using Git or download a zip folder containing the source code from mcu-viznlc.



FelixPan001 Merge pull request #1 from NXP/github-first-relea

k32w_ble	First github release base
lpc845_low_power_control	First github release base
prebuilt_bins	First github release base
rt106f_bootloader	First github release base
rt106f_smart_lock	First github release base
smartlockmanager	First github release base
LICENSE.txt	First github release base
README.md	First github release based on RC2

last month

https://github.com/NXP/mcu-viznlc.git

Use Git or checkout with SVN using the web URL.

VIZNLC is a low-cost and MCU-Based Solution for Face Recognition with Liveness Detection.

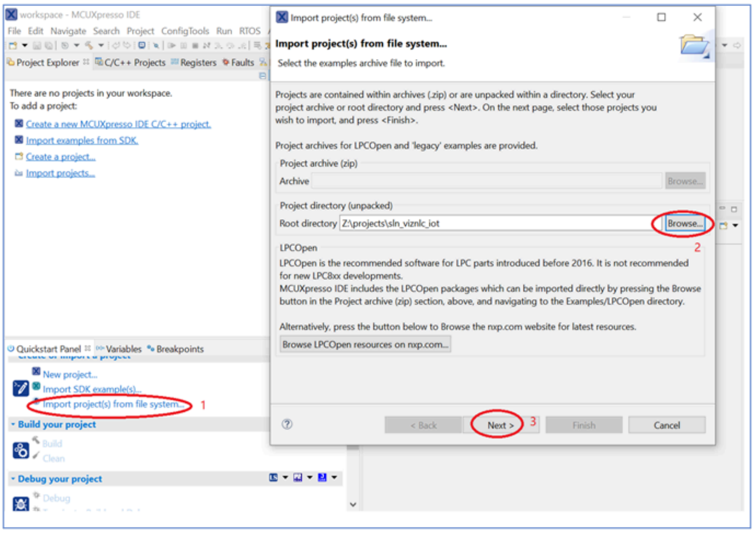
Figure 26. SLN-VIZNLC-IOT GitHub repository

Note: *If downloading a zipped archive, ensure to unzip this folder before proceeding to the next step.*

3.4 Importing SLN-VIZNLC-IOT projects

To import the projects we downloaded into the IDE, follow the steps below:

1. Click **Import project(s) from file system...**, then **Browse** your project path where you unzipped the `sln_viznlc_iot` source code, and click **Next**.



There are no projects in your workspace.
To add a project:

- Create a new MCUXpresso IDE C/C++ project.
- Import examples from SDK.
- Create a project...
- Import projects...

Import project(s) from file system...

Select the examples archive file to import.

Projects are contained within archives (.zip) or are unpacked within a directory. Select your project archive or root directory and press <Next>. On the next page, select those projects you wish to import, and press <Finish>.

Project archives for LPCOpen and 'legacy' examples are provided.

Project archive (zip)

Archive

Project directory (unpacked)

Root directory Z:\projects\sln_viznlc_iot

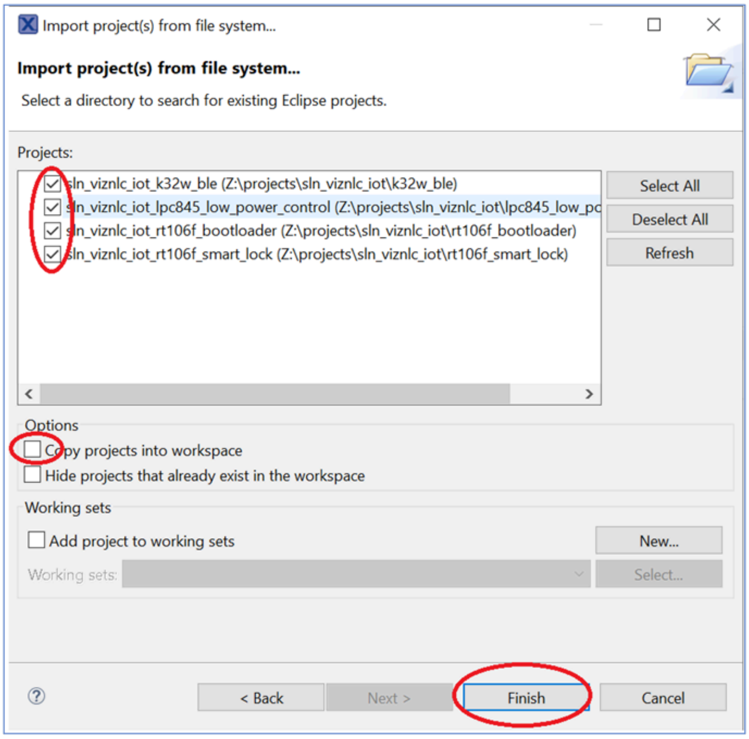
LPCOpen

LPCOpen is the recommended software for LPC parts introduced before 2016. It is not recommended for new LPC8xx developments.
MCUXpresso IDE includes the LPCOpen packages which can be imported directly by pressing the Browse button in the Project archive (zip) section, above, and navigating to the Examples/LPCOpen directory.

Alternatively, press the button below to Browse the nxp.com website for latest resources.

Figure 27. Import project(s) from file system

2. Import the files specified in Figure 28 from the displayed screen.



Import project(s) from file system...

Select a directory to search for existing Eclipse projects.

Projects:

☑ sln_viznlc_iot_k32w_ble (Z:\projects\sln_viznlc_iot\k32w_ble)
☑ sln_viznlc_iot_lpc845_low_power_control (Z:\projects\sln_viznlc_iot\lpc845_low_pc
☑ sln_viznlc_iot_rt106f_bootloader (Z:\projects\sln_viznlc_iot\rt106f_bootloader)
☑ sln_viznlc_iot_rt106f_smart_lock (Z:\projects\sln_viznlc_iot\rt106f_smart_lock)

Options
□ Copy projects into workspace
□ Hide projects that already exist in the workspace

Working sets
□ Add project to working sets
Working sets:

Figure 28. Import project(s) from file system

3. Once successfully imported, you should see projects open in the **Project Explorer** pane on the left side of the IDE.

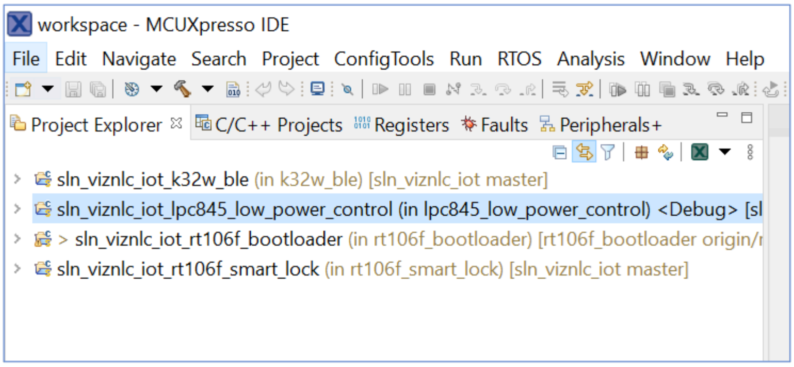

workspace - MCUXpresso IDE
File Edit Navigate Search Project ConfigTools Run RTOS Analysis Window Help
Project Explorer
C/C++ Projects
Registers
Faults
Peripherals+
sln_viznlc_iot_k32w_ble (in k32w_ble) [sln_viznlc_iot master]
sln_viznlc_iot_lpc845_low_power_control (in lpc845_low_power_control) <Debug> [sl
> sln_viznlc_iot_rt106f_bootloader (in rt106f_bootloader) [rt106f_bootloader origin/r
sln_viznlc_iot_rt106f_smart_lock (in rt106f_smart_lock) [sln_viznlc_iot master]

Figure 29. SLN-VIZNLC-IOT projects

3.5 Building the SLN-VIZNLC-IOT projects

The SLN-VIZNLC-IOT SDK allows you to build the smart lock application directly. The application is made up of four subprojects:

- The `lpc845_low_power_control` project manages the power control of the system. LPC845 works as the host MCU. PIR sensor activates the host MCU and powers the RT106F part.
- The `k32w_ble` project implements the feature of Bluetooth LE module (UART over Bluetooth LE).
- The `rt106f_bootloader` and `rt106f_smart_lock projects` are the out-of-box applications that we used earlier to demonstrate the SLN-VIZNLC-IOT face recognition capabilities. The bootloader manages to jump into the smart lock application.

These above applications are flashed onto your SLN-VIZNLC-IOT kit by default.

In the **Project Explorer** pane, select the project file and navigate to the **QuickStart Panel**. To start the compilation and linking of this application, click the **Build** option.

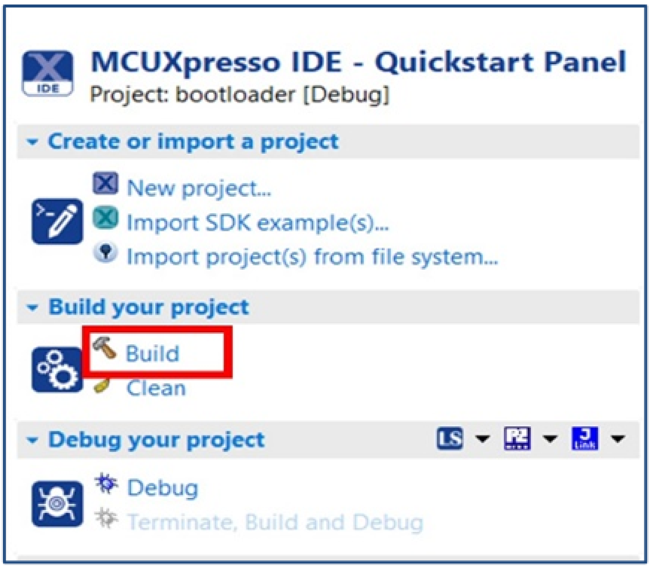
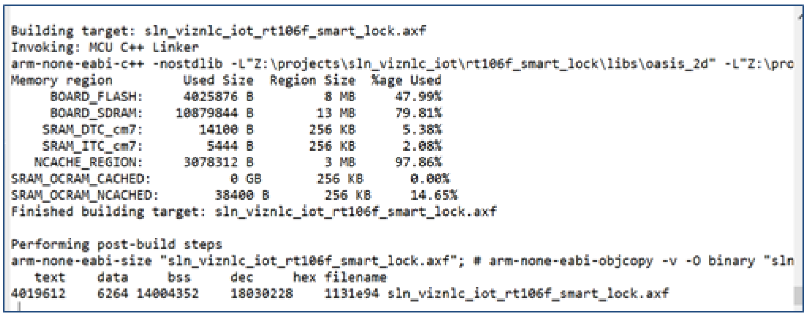

MCUXpresso IDE - Quickstart Panel
Project: bootloader [Debug]
Create or import a project
New project...
Import SDK example(s)...
Import project(s) from file system...
Build your project
Build
Clean
Debug your project
Debug
Terminate, Build and Debug

Figure 30. How to build

Building may take a few minutes to complete, but do not worry, as it is normal for applications of this size. Once finished, a message as shown in Figure 31 appears at the bottom of the IDE. Consider `rt106f_smart_lock` project as an example.



```
Building target: sln_viznlc_iot_rt106f_smart_lock.axf
Invoking: MCU C++ Linker
arm-none-eabi-c++ -nostdlib -L"Z:\projects\sln_viznlc_iot\rt106f_smart_lock\libs\oasis_2d" -L"Z:\pro
Memory region         Used Size  Region Size  %age Used
     BOARD_FLASH:     4025876 B         8 MB     47.99%
     BOARD_SDRAM:    10879844 B        13 MB     79.81%
    SRAM_DTC_cm7:       14100 B       256 KB      5.38%
    SRAM_ITC_cm7:        5444 B       256 KB      2.08%
   NCACHE_REGION:     3078312 B         3 MB     97.86%
SRAM_OCRAM_CACHED:          0 GB       256 KB      0.00%
SRAM_OCRAM_NCACHED:       38400 B       256 KB     14.65%
Finished building target: sln_viznlc_iot_rt106f_smart_lock.axf

Performing post-build steps
arm-none-eabi-size "sln_viznlc_iot_rt106f_smart_lock.axf"; # arm-none-eabi-objcopy -v -O binary "sln
   text    data     bss     dec     hex filename
4019612    6264 14004352    18030228    1131e94 sln_viznlc_iot_rt106f_smart_lock.axf
```

Figure 31. Build message

3.6 Flashing and debugging SLN-VIZNLC-IOT projects

With the SLN-VIZNLC-IOT application project compiled, it is now time to program associated binaries of this project into flash.

Flashing and debugging the SLN-VIZNLC-IOT kit requires a SEGGER J-Link with a 9-pin Cortex-M adapter and V7.60d or newer version of the J-Link Software and Documentation Pack. This new version of J-Link can be found on the SEGGER website.

Note: *There is a problem to program/debug RT106F using the default J-Link V7.70d in MCUXpresso IDE V11.6.1. J-Link software must be updated, and J-link software V7.60d has been verified.*

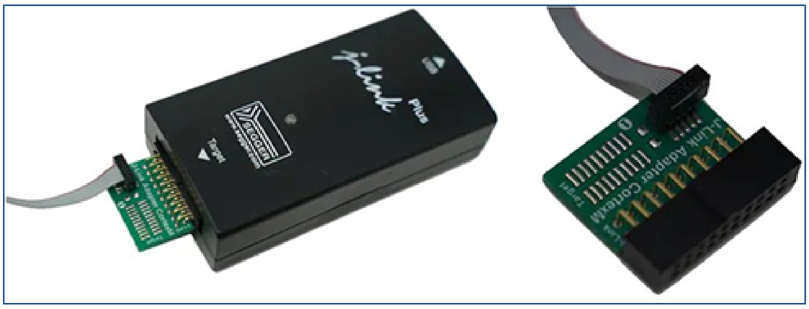
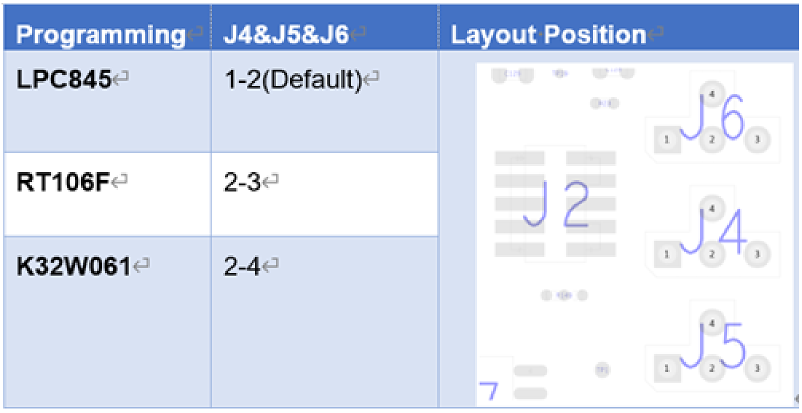


Figure 32. J-Link 9-pin adapter

Note: *The SLN-VIZNLC-IOT kit has one SWD interface connector (J2), which supports programming and debugging the RT106F, LPC845, and K32W061 via a SEGGER J-Link debug probe. To select which MCU to program/debug, the J4, J5, and J6 connectors, each must be set to the position indicated in Figure 33.*



Programming	J4&J5&J6	Layout Position
LPC845	1-2(Default)	
RT106F	2-3	
K32W061	2-4	

J6
4
1
2
3
J2
J4
4
1
2
3
J5
4
1
2
3
7

Figure 33. Switch debug interface

To flash the kit, follow the steps below:

1. Select the programming MCU on J4, J5, and J6 connectors.
2. Attach your J-Link debug probe into the J2 header as shown in Figure 34 and connect your J-Link to your computer via USB.

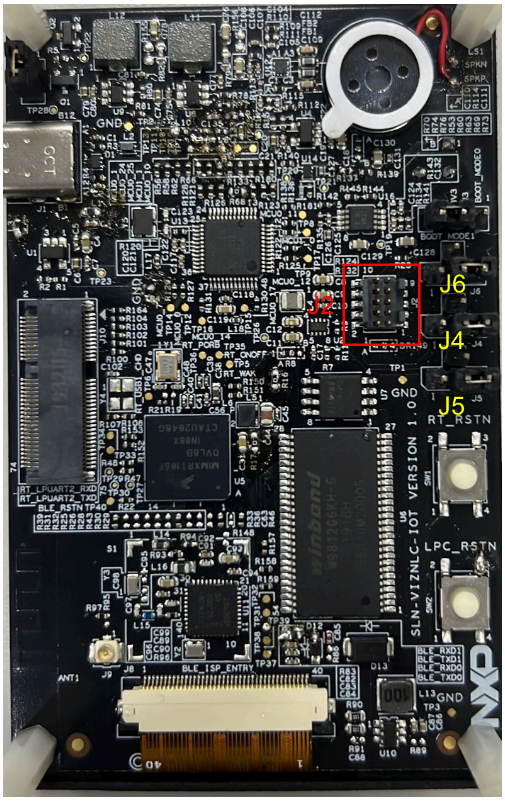


Figure 34. Attach J-Link debugger

3. Next, provide power to the kit by plugging a USB-C cable into the USB-C port of the kit and plug the other end into your laptop/PC.

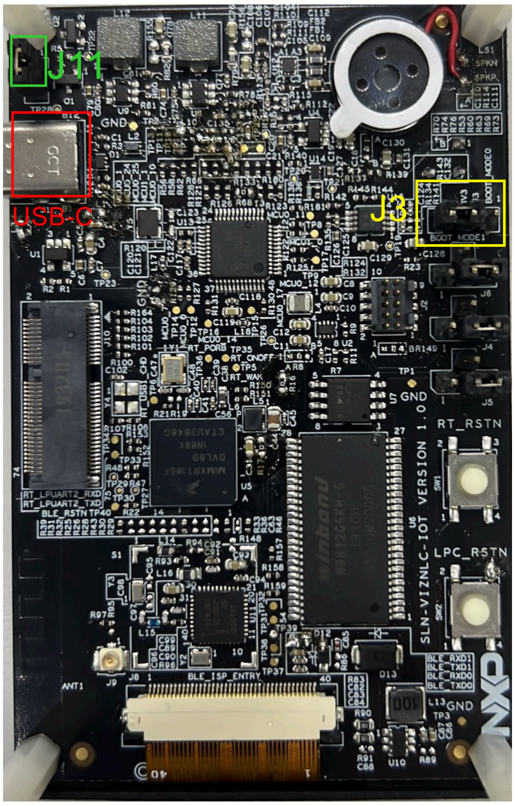


Figure 35. Power on SLN-VIZNLC-IOT kit

Note: *LPC845 controls the power supply of RT106F and K32W061. Therefore, during the debug stage, we can bypass it by connecting J11 jumper as shown in the green highlighted box in Figure 35.*

4. Select the MCU-related project in the **Project Explorer** pane. Consider `lpc845_low_power_control` as an example below.
5. To start the process of loading the binary into flash and begin debugging, choose the **Debug** option in the **QuickStart Panel**.

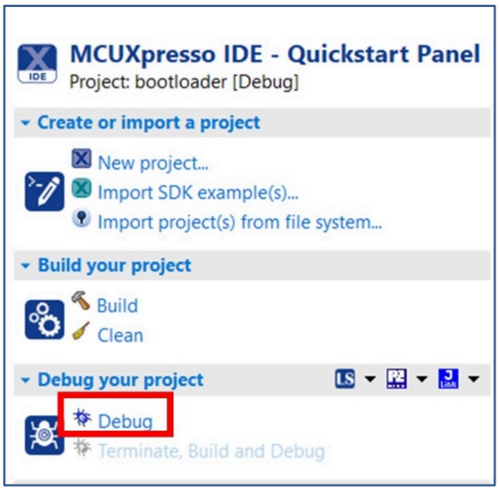
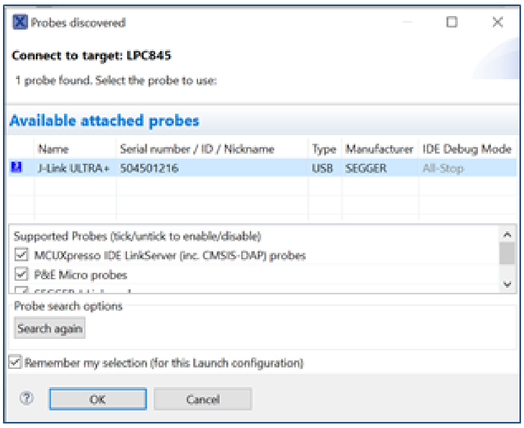

MCUXpresso IDE - Quickstart Panel
Project: bootloader [Debug]
Create or import a project
New project...
Import SDK example(s)...
Import project(s) from file system...
Build your project
Build
Clean
Debug your project
Debug
Terminate, Build and Debug

Figure 36. Debug your project

6. Select the J-Link probe that is connected to your kit and click the **OK** button. Flashing tool is launched.

All information provided in this document is subject to legal disclaimers.



Probes discovered

Connect to target: LPC845

1 probe found. Select the probe to use:

Available attached probes

	Name	Serial number / ID / Nickname	Type	Manufacturer	IDE Debug Mode
	J-Link ULTRA+	504501216	USB	SEGGER	All-Stop

Supported Probes (tick/untick to enable/disable)

☑ MCUXpresso IDE LinkServer (inc. CMSIS-DAP) probes

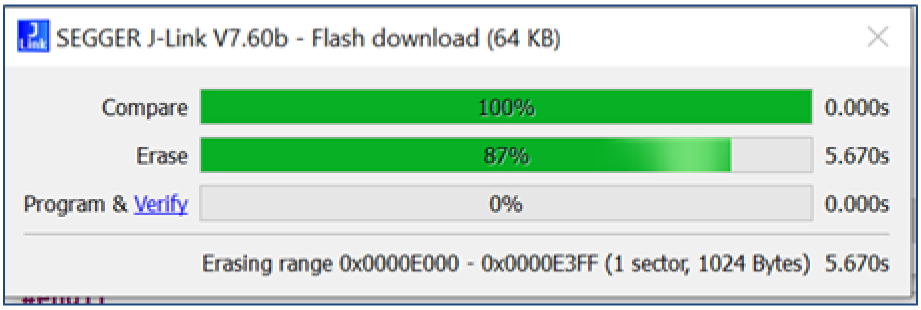
☑ P&E Micro probes

Probe search options

☑ Remember my selection (for this Launch configuration)

Figure 37. Probes discovered

7. Now, proceed to flash the binary associated with the currently selected project.

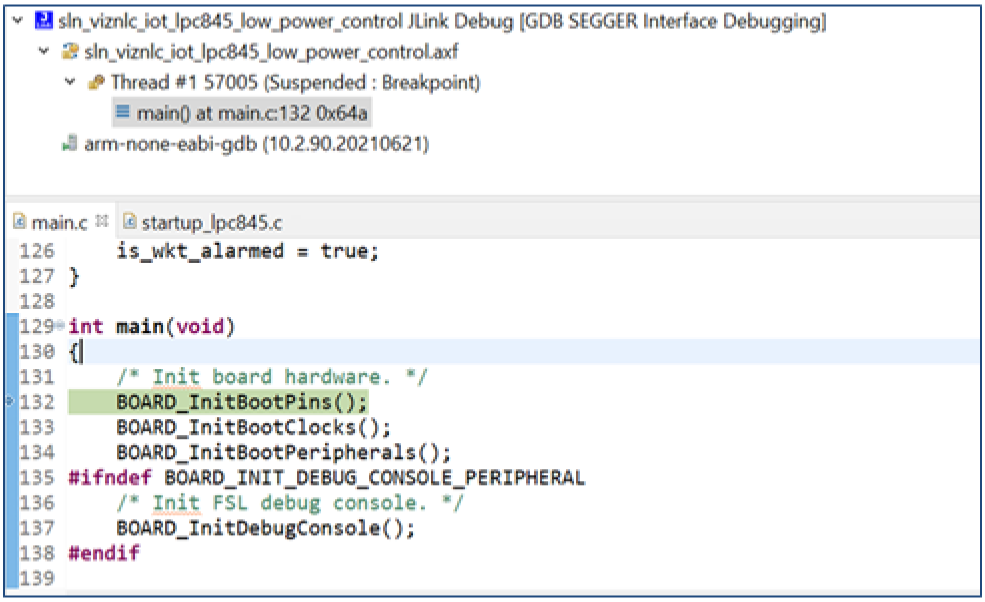


SEGGER J-Link V7.60b - Flash download (64 KB)

```
Compare           100%   0.000s
Erase              87%   5.670s
Program & Verify    0%   0.000s
Erasing range 0x0000E000 - 0x0000E3FF (1 sector, 1024 Bytes)  5.670s
```

Figure 38. Flash downloading

8. After successfully debugging the application, the program breaks at the project main to start the **GDB** debug process.



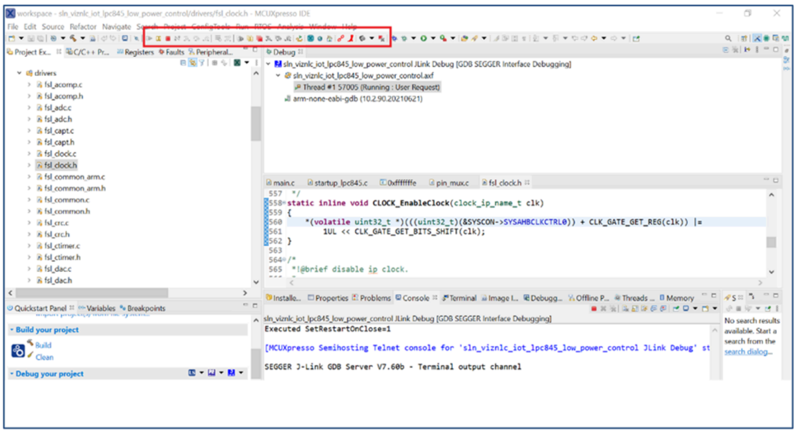
```
sln_viznlc_iot_lpc845_low_power_control JLink Debug [GDB SEGGER Interface Debugging]
  sln_viznlc_iot_lpc845_low_power_control.axf
    Thread #1 57005 (Suspended : Breakpoint)
      main() at main.c:132 0x64a
  arm-none-eabi-gdb (10.2.90.20210621)
```

```c
    is_wkt_alarmed = true;
}

int main(void)
{
    /* Init board hardware. */
    BOARD_InitBootPins();
    BOARD_InitBootClocks();
    BOARD_InitBootPeripherals();
#ifndef BOARD_INIT_DEBUG_CONSOLE_PERIPHERAL
    /* Init FSL debug console. */
    BOARD_InitDebugConsole();
#endif
```

Figure 39. Start to debug project

9. To suspend the debug session, click the **pause/play** button on the top of the screen.


workspace - sln_viznlc_iot_lpc845_low_power_control/drivers/fsl_clock.h - MCUXpresso IDE
File Edit Source Refactor Navigate Search Project ConfigTools Run RTOS Analysis Window Help
Project Ex... C/C++ Pr... Registers Faults Peripheral...
Debug
sln_viznlc_iot_lpc845_low_power_control JLink Debug [GDB SEGGER Interface Debugging]
sln_viznlc_iot_lpc845_low_power_control.axf
Thread #1 57005 (Running : User Request)
arm-none-eabi-gdb (10.2.90.20210621)
drivers
fsl_acomp.c
fsl_acomp.h
fsl_adc.c
fsl_adc.h
fsl_capt.c
fsl_capt.h
fsl_clock.c
fsl_clock.h
fsl_common_arm.c
fsl_common_arm.h
fsl_common.c
fsl_common.h
fsl_crc.c
fsl_crc.h
fsl_ctimer.c
fsl_ctimer.h
fsl_dac.c
fsl_dac.h
main.c
startup_lpc845.c
0xfffffffe
pin_mux.c
fsl_clock.h
Quickstart Panel
Variables
Breakpoints
Build your project
Build
Clean
Debug your project
Installe...
Properties
Problems
Console
Terminal
Image I...
Debugg...
Offline P...
Threads ...
Memory
No search results available. Start a search from the search dialog...

```
 */
static inline void CLOCK_EnableClock(clock_ip_name_t clk)
{
    *(volatile uint32_t *)(((uint32_t)(&SYSCON->SYSAHBCLKCTRL0)) + CLK_GATE_GET_REG(clk)) |=
        1UL << CLK_GATE_GET_BITS_SHIFT(clk);
}

/*
 *!@brief disable ip clock.
```

```
sln_viznlc_iot_lpc845_low_power_control JLink Debug [GDB SEGGER Interface Debugging]
Executed SetRestartOnClose=1

[MCUXpresso Semihosting Telnet console for 'sln_viznlc_iot_lpc845_low_power_control JLink Debug' st

SEGGER J-Link GDB Server V7.60b - Terminal output channel
```

Figure 40. Debugging project

> ***Note:*** *If you want to debug* `rt106f_smart_lock` *project, ensure that you have flashed the bootloader.*

3.7 Additional resources

If you have made it to this section, you have successfully finished the SLN-VIZNLC-IOT getting started experience.

For a comprehensive understanding of all the Out-of-Box Experience (OoBE) features, including the additional demo applications that come flashed with the kit, see *SLN-VIZNLC-IOT User Guide*.

To start building your own applications and learn more about the software architecture, available developer tools, and more, head over to the *SLN-VIZNLC-IOT Software Developer Guide*.

4 Acronyms

Table 1 lists the acronyms used in this document.

Table 1. Acronyms

Acronym	Definition
TFT	Thin Film Transistor
HAL	Hardware Abstraction Layer
OoBE	Out-of-Box Experience
MSD	Mass Storage Device
VIZNLC	Vision Low Cost
FW	Firmware
SW	Software
HW	Hardware
PIR	Passive InfraRed
IR	InfraRed
GUI	Graphical User Interface

5 Revision history

The Table 2 summarizes the changes done to this document since the initial release.

Table 2. Revision history

Revision history	Date	Substantive changes
0	10 March 2023	Initial release

6 Legal information

6.1 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.

6.2 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 http://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 data sheet 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.

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.

6.3 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.

AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, µVision, Versatile — are trademarks and/or registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved.

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: http://www.nxp.com

Date of release: 10 March 2023
Document identifier: SLN-VIZNLC-IOT-GSG