IoT Sensing SDK Getting started with IoT Sensing SDK (ISSDK) middleware

Rev. 1.5 — 7 July 2022

User guide

Document information

Information	Content
Keywords	IoT Sensing SDK, ISSDK, MCUXpresso, middleware
Abstract	Getting started with IoT Sensing SDK (ISSDK) v1.8 middleware



Getting started with IoT Sensing SDK (ISSDK) middleware

Revision number	Date	Description
1.5	20220707	Updates for ISSDK v.1.8
1.4	20180403	Updates for ISSDK v1.7
1.3	20170525	Updates for ISSDK v1.6
1.2	20170306	Updates for ISSDK v1.5
1.1	20161123	Updates for ISSDK v1.1
1.0	20160803	Initial public release

Getting started with IoT Sensing SDK (ISSDK) middleware

1 Prerequisites

This document assumes completion of the following prerequisites prior to attempting to use the ISSDK v1.8 middleware:

- One of the recommended IDEs is installed on the development PC (see the release notes)
- A FRDM-K22F-A8974 sensor kit is connected to the development PC
- User understanding of the debug environment set up for the Freedom family of development boards using OpenSDA or third-party debugger with their IDE of choice
- User familiarity with the MCUXpresso SDK and MCUXpresso SDK Builder

2 Overview

The IoT Sensing Software Development Kit (ISSDK) is the embedded software framework that enables the NXP digital and analog sensors platforms for IoT applications. ISSDK provides a unified set of sensor support models that target the NXP portfolio of sensors across a broad range of Arm Cortex core-based Microcontrollers. ISSDK is offered as a middleware component in MCUXpresso SDK for supported microcontrollers. ISSDK relies on the SDK 2.x drivers and project release infrastructure to create a unified user experience. ISSDK v1.8 combines a set of robust sensor drivers and algorithms along with example applications to allow a user to get started using NXP sensors quickly.

For more information on ISSDK, go to <u>www.nxp.com/iotsensingsdk</u>.

2.1 ISSDK architecture

Figure 1 shows the high-level *layer cake* architecture of the ISSDK v1.8 middleware. ISSDK is designed to provide separable layers of functionality that a customer can choose to use or ignore based on their specific needs. In addition, the ISSDK architecture is portable due to the use of open APIs (ARM Ltd. CMSIS Driver APIs). ISSDK is designed to allow users to start with the smallest production footprint (memory and CPU load) as is practical for their particular application. Typically, the smallest production footprint is achieved by selecting the Bare Metal option; however, some applications may prefer using one of the RTOSs supplied with MCUXpresso SDK 2.x.

Getting started with IoT Sensing SDK (ISSDK) middleware



In the following sections, this guide focuses on how ISSDK can be deployed via MCUXpresso for a specific Freedom Sensor Toolbox sensor demonstration kit called the FRDM-K22F-A8974. This kit combines the Kinetis FRDM-K22F development board with an FRDM-STBI-A8974 sensor shield to provide a standalone, low-cost sensor development platform.

3 NXP Freedom Sensor Toolbox Sensor Development Ecosystem

NXP provides a sensor development ecosystem called the Freedom Sensor Toolbox. This ecosystem is designed to provide solutions for hardware and software that enable customers to evaluate and prototype with sensors quickly and easily. ISSDK v1.8 is deployed on top of the Freedom Sensor Toolbox hardware platforms and is expected to become the embedded software support platform for the ecosystem.

The following figure shows how the Freedom Sensor Toolbox development hardware can be used to explore the ISSDK v1.8 software. In this example, the MCUXpresso IDE is used to compile, load, and launch an existing project into the FRDM-K22F-A8974 kit. The customer may then launch a terminal emulator to examine the debug console output provided for many ISSDK v1.8 projects.

Getting started with IoT Sensing SDK (ISSDK) middleware



Figure 2. Freedom Sensor Toolbox development environment

More information about the Freedom Sensor Toolbox development ecosystem can be found at http://nxp.com/sensortoolbox. The remainder of this document focuses on the steps involved in using the FRDM-K22F-A8974 development kit with the ISSDK enablement software.

Project deployment 4

ISSDK v1.8 is fully integrated into the MCUXpresso SDK Builder delivery system. MCUXpresso includes both cloud and locally based tools to collect and build projects from the MCUXpresso SDK repositories, MCUXpresso SDK 2.x is built using a hierarchy of deployed Git repositories. Specific project codebases are built through the online tool. A given codebase is specified by its target (device, board, or kit desired), the version of MCUXpresso SDK 2.x, the supported IDEs (MCUXpresso IDE, IAR, Keil, GCC), and the target Host OS (Windows, Mac, or Linux).

4.1 MCUXpresso SDK Builder

MCUXpresso SDK Builder is a cloud-based system used to build MCUXpresso SDK 2.x packages. ISSDK is an optional component that can be deployed by MCUXpresso in two ways:

- If the customer selects a FRDM sensor kit, such as the FRDM-K22F-A8974, then the ISSDK sensor drivers and example applications appropriate for that kit are deployed into the package.
- If the customer selects a supported device or FRDM board and checks the box for optional ISSDK support, then all the sensor drivers and example applications are deployed into the package.

Getting started with IoT Sensing SDK (ISSDK) middleware

Note, in both cases, the MCUXpresso SDK 2.x drivers and example applications are also deployed alongside the ISSDK files.

Figure 3 shows the MCUXpresso environment for deploying ISSDK (see https://mcuxpresso.nxp.com/en/configuration-settings). In this example, the customer has selected the FRDM-K22F-A8974 kit, the MCUXpresso IDE, and Windows host operating system. Notice that ISSDK middleware component has been selected by default because the target is a board/shield kit. When the customer selects the Build SDK Package, the request is sent to the build servers. Requests for packages are served in order and when the package is ready, a notification is returned to the customer. The customer may download the package (a zip file) and deploy it into their local system.

elope	te a downloadable SDK archive er Environment Settings here will impact files and examples project			Reversaria
	Host OS	Toolch	ain / IDE 🧱 🎯 🐨 KEIL	SDK 2.12.0 (released 2022-06- 01) SDK Tag REL_2.12.0_MAJOR_RF.
earcl	h		Q	UNSELECT ALL
	Name	Category	Description	Dependencies
	Nxp iot sensing sdk	Middleware	IoT Sensing SDK (ISSDK) provides sensor drivers and referenc (more)	
	SDMMC Stack	Middleware	Stack supporting SD, MMC, SDIO	
	CMSIS DSP Library	CMSIS DSP Lib	CMSIS DSP Software Library	
	emWin	Middleware	emWin graphics library	
	Fatfs	Middleware	FAT File System stack	
~	FreeMASTER	Middleware	FreeMASTER communication driver for 32bit platforms	
	LVGL	Middleware	LVGL Open Source Graphics Library	
				aaa-0

4.2 Deployment directory structure

Once the MCUXpresso package has been downloaded, the user can extract the package on their local machine. Figure 4 displays the MCUXpresso directory structure.

Getting started with IoT Sensing SDK (ISSDK) middleware

Name	Туре
📜 boards	File folder
CMSIS	File folder
components	File folder
🦲 devices	File folder
📙 docs	File folder
📜 middleware	File folder
📜 tools	File folder
COPYING-BSD-3	File
FRDM-K22F-A8974_manifest_v3_10	XML Document
LA_OPT_NXP_Software_License RTA	TXT File
NFC_SLDA	Adobe Acrobat Document
W-Content-Register	TXT File
	aaa-04711

The CMSIS, devices, docs, RTOS, and tools directories are unchanged from standard MCUXpresso SDK 2.x deployments. ISSDK v1.8 projects appear as new targets in the boards directory. Figure 5 illustrates the directory where frdmk22f_a8974 (ISSDK) reference example projects are available, as well as the base projects for the frdmk64f.



In addition, a new middleware component is created that contains the ISSDK drivers, algorithms, and other support files as shown in <u>Figure 6</u>.

Name	Туре
📕 freemaster	File folder
issdk	File folder
📜 rtcesl	File folder
J sdmmc	File folder
middleware_baremetal_MK22F512	CMAKE File
	aaa-047116
Figure 6. Middleware components available as par	t of SDK package

5 Build and run a sensor driver example

Choose FRDM-K22F-A8974 kit configuration and download SDK package from MCUXpresso SDK Builder. Figure 7 and Figure 8 illustrate how to get the FRDM-K22F-A8974 SDK package from MCUXpresso configuration.

Getting started with IoT Sensing SDK (ISSDK) middleware



Figure 7. FRDM-K22F-A8974 kit configuration and SDK package download

Install downloaded SDK package into MCUXpresso IDE (drag and drop SDK package into "Installed SDKs" view). Start SDK import wizard, import any existing ISSDK example by choosing frdmk64_agm01 board for your project, build the imported project and load the image to FRDM-K22F-A8974 kit. These actions create a standalone project in your workspace. See Figure 8.

NXP Semiconductors

IoT Sensing SDK

Getting started with IoT Sensing SDK (ISSDK) middleware

e 🔊 A	iphends+ ° 0 № Wetcome x B ≪ I 0 10 MCUXpresso IDE v11.6.0 X ≪ I 0 10 X ≪ I 0 10
There are no projects in your workspace. To add a project:	
Create a new MCUXpresso IDE C/C++ project.	
Create a project iba Import projects	Home Overview What's New First Steps Web Resources Documentation IDE
	Welcome to MCUXpresso IDE
	MCUXpresso IDE provides an easy-to- use Eclipse-based development environment for NXP MCUs based on
	ARM Cortex-Mores, including LPC and Kinetis microcontrollers and IMX RT Download and Install SDKs
	editing, compiling, and debugging
	features with the addition of MCU- specific debugging views, code trace of drafting must core debugging, and Not that description for the section of the sec
Γ	
O Quickstart Panel × ∞ Variables * Breakpoints MCUXpresso IDE - Quickstart/Pa No project selected	
No project selected Create or import a project	To install an SDK, simply drag and drop an SDK (zip file/folder) or an SDK Git repository into the 'Installed SDKs' view. [Common 'mcwpress, SERDM-K22E 100
New project Import SDK example(s)	Installed SDIG [®] Available Boards Available Devices V FROM-K22F-A8974 1.0.0 Name SDK Version Manifest Version Location V Examples
 Import project(s) from file system. Build your project 	
Ruild	Common SQL 2, FLOOR MACE 2 Model Zero observation Zero observatio Zero observation Zero obse
Debug your	110_UPCXPRESSOSS16v
Debu The project(s) for device: N	MK22FN512ox12 using board: FRDM-K22F-A8974
SDK MCUs 8 An	selection page valiable boards Pk (%)
MCUs from installed SDKs. Plu Please click above or visit	supported boards for device. MK22PM512xxx12
additional SDKs.	
 K2x MK22FN512xxx12 	
> KE1x > LPC550x_S0x > LPC551x_S1x	Interviced as a second
7 0001201	INTERAL BUZZ
Selected Device: MK22FN512xxx1	12 using board: FRDM-K22F-A8974 SDKs for selected MCU
Target Core: cm4 Description: Kinetis K22-120 M	Hrz, Cost Effective, Full-Speed USB Amme SDK Versi. Manifest. Location #BSDK 2x FRDM-422F - 212.0 (stag 3:100
	S SDK Import Wizard — 🗆 🗙
	▲ You have selected 1 project to import: Yomki221,8974 (hid9714c) interrupt: The source from the SDV will be copied into the workspace. If you want to use limited files, please uptio the SDX 2x FRDM-K22F-
	Vio Name setected 1 project to import: "form222_48974_fold89744_interrupt: The source from the SDK will be copied into the workspace. If you want to use linked files, please unity the SDK_2x_FRDM-K22F- Vio Tiport projects Project name prefix: Interrupt:
	Import Project 1 project to import: findm222_48974 (sitestructure): Import Project Project Project reprojects Project reprojects Project reprojects Incomer Inform Royal Ro
	Import project 1 project to import "india22,48974, fuid89744, linesrgift." Import projects Project frame prefix "India22,48974 Import project Colores Import project frame Import project fr
	Import project 1 project to import "india22,44974, fuid8744, fuid
	Import project Project Type
	View selected 1 project to import: finde224,2474, fuid9744, interrupt: The source from BSOK will be copied into the workspace. If you want to use linked files, please using the SDK, 2x, FRDM-K22F. View of the SDK will be copied into the workspace. If you want to use linked files, please using the SDK, 2x, FRDM-K22F. View of the SDK will be copied into the workspace. If you want to use linked files, please using the SDK, 2x, FRDM-K22F. View of the SDK will be copied into the workspace. If you want to use linked files, please using the SDK, 2x, FRDM-K22F. View of the SDK will be copied into the workspace. If you want to use linked files, please using the SDK, 2x, FRDM-K22F. View of the SDK will be copied into the workspace. If you want to use linked files, please using the SDK will be used to the SDK
	Import Source for the SDK with Boogned into the workspace. If you want to use inked files, please unity the SDK 2x FRDM-K22r. Import projects Project for projects Project for default location Constant Character (Character (Ch
	Import Source for the SDK with Boogned into the workspace. If you want to use inked files, please unity the SDK 2x FRDM-K22r. Import projects Project for projects Project for default location Constant Character (Character (Ch
	Import Source for the SDK with Boogned into the workspace. If you want to use inked files, please unity the SDK 2x FRDM-K22r. Import projects Project for projects Project for default location Constant Character (Character (Ch
	Import Source for the SDK with Boogned into the workspace. If you want to use inked files, please unity the SDK 2x FRDM-K22r. Import projects Project for projects Project for default location Constant Character (Character (Ch

Notice that the code is ready to start in the file *fxls8974cf_interrupt.c*, which is the main application for this example.

Start the program execution. The blue LED begins to flash on the FRDM-K22F board.

Next, start a terminal emulation program with the serial port set as shown in Figure 9.

Getting started with IoT Sensing SDK (ISSDK) middleware

RealTerm: Serial Capture Program 2.0.0.57				×
n he he he he he he		1 0	F	
Baud 115200	2C-2 I2CMisc Misc Spy	<u>\n</u> <u>Clear</u>	Status	mected (2) (3) (3) (3) (1) (6) (9) EAK

Figure 9. Terminal emulation program with serial port set

Figure 10 displays the debug console window and output.

늘 RealTerm: Serial Capture Program 2.0.0.57 ISSDK FXLS8974 sensor driver example demonstration with interrupt mode Successfully Initialized Timandra with WHO_AM_I = 0x86 Successfully Applied FXLS8974 Sensor Configuration 81 Y= 38 Z= 1071 -20 Y= -35 Z= 1075 -16 Y= -20 Z= 1090 -24 Y= -24 Z= 1082 -24 Y= -31 Z= 1094 -20 Y= -31 Z= 1094 -20 Y= -16 Z= 1079 -24 Z= 1086 Υ = -16 Z= 1086 -8 Z= 1079 3 Υ = -8 Y= -31 Z= 1082 aaa-047120

Figure 10. Debug console output

In this example, each data ready interrupt from the FXLS8974CF triggers the application to read the raw X-, Y-, Z-axis accelerometer values.

6 Build and run pedometer algorithm example

ISSDK also provides reference pedometer algorithm example using NXP accelerometers. The example demonstrates configuration of all registers required to use the sensor as the acceleration source for the pedometer. Pedometer algorithm measures step counts based on fxls8974cf accelerometer data.

In order to build and run the pedometer algorithm example, click the SDK import wizard and import ISSDK pedometer algorithm example by choosing the frdmk22f_a8974 board

Getting started with IoT Sensing SDK (ISSDK) middleware

for your project. The SDK import wizard and the imported ISSDK pedometer algorithm create a standalone project in your workspace. See <u>Figure 11</u>.

Minport projects	
Project name prefix: frdmk22f_a8974	× Project name suffix:
Use default location	
Location: C:\Users\nxa11926\Documents\MCUXpressolDE_11.6.0_8152_e	ear2\Gemini\frdmk22f_a8974 Browse
Project Type	Project Options
C Project C++ Project C Static Library C++ Static Library	SDK Debug Console ○ Semihost UART ○ Example default Copy sources Import other files
Examples	🔟 🗹 🙀 🗷 E
type to filter	
Name v 🔳 🗟 issdk	Description Version
	The pedometer example implements the ISSDK FXLS8974CF sensor driv The FXLS8974CF FreMASTER example implements FreeMASTER demo The FXLS8974CF Interrupt example application demonstrates the use of The FXLS8974CF motion wakeup example demonstrated SDCD block c The fXLS8974CF POLL example application demonstrates the use of the
	The FXLS8974CF POLL example application demonstrates the use of the
□ ≡ fxls8974cf_poll	The FXLS8974CF POLL example application demonstrates the use of the

Figure 11. SDK import wizard for pedometer project

Build the pedometer algorithm project in the MCUXpresso IDE and load the image to FRDM-K22F-A8974 kit. See Figure 12.

E 18 7 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8	MCUXpresso IDE v11.6.0
CMSI22_abs//Decometer_abs///_SOBDags Composed Settings Solutions Solutions CMSIS	🔤 🏠 🥥 🔶 🕥 🖾 🔿
board cmsis_driver component c@ device	Home Overview What's New First Steps Web-Resources Documentation IDE Welcome to MCUXpresso IDE
divers d	MCUXpresso IDE provides an easy-to- use Eclipse-based development environment for NAP MCUs based on ARM Cortex-M cores, including LPC and Kinetis microcontrollers and iMX RT crossover processors. It offers advanced editing, compiling, and debugging features with the addition of MCU- specific debugging views, code trace not are future without addition of MCU- specific debugging, and work etc. Infour ation tools.
	Of Installed SDKs Properties Problems Console Aption Aption Console Q Debugger Console Q Offline Peripherals
MCUXpresso IDE - Quickstart Panel Project: frdmk22f_a8974_pedometer_a8974 [Debug]	DT Build Console (frdmk22f_a8974_pedometer_a8974) nvoxing: nco Linker
reate or import a project	<pre>hrm-none-eabi-gcc -nostdlib -L"C:\Users\nxa11926\Documents\MCUXpressoIDE_11.6.0_8152_ear2\Gemini\frdmk22f_a8974_pedometer_a8974\lib: demory region Used Size Region Size %age Used</pre>
New project	PROGRAM_FLASH: 42548 B 512 KB 8.12% SRAM_UPPER: 10868 B 64 KB 16.58%
Import SDK example(s) Import project(s) from file system	SRAM_LOWER: 0 GB 64 KB 0.00%
ild your project	Finished building target: frdmk22f_a8974_pedometer_a8974.axf
	Performing post-build steps urm-none-eabi-size "frdmk22f_a8974_pedometer_a8974.axf"; # arm-none-eabi-objcopy -v -0 binary "frdmk22f_a8974_pedometer_a8974.axf" !
S Build Clean	text data bis dec hex filename 42476 72 10796 53344 d060 frdmk22f a8974 pedometer a8974.axf
ebug your project 💽 👻 🔛 🔹	
Debug	12:06:27 Duild Einichad & annance & wannings (teak 10: 750mr)
the result of the second second	* <
	aaa-0

© 2022 NXP B.V. All rights reserved.

Getting started with IoT Sensing SDK (ISSDK) middleware

Start the program execution. The green LED begins to flash on the FRDM-K22F-A8974 board.

Next, start a terminal emulation program with the serial port set as shown in Figure 9.

When the pedometer algorithm example runs successfully, the step counts are printed to the terminal as shown in Figure 13.

Note: To get the step counts to change, walk with the board.



Figure 13. Serial capture program displaying step counts

Getting started with IoT Sensing SDK (ISSDK) middleware

7 Legal information

7.1 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.

7.2 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.

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.

7.3 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

Kinetis — is a trademark of NXP B.V.

ISSDKGETSTARTEDUG

Getting started with IoT Sensing SDK (ISSDK) middleware

Figures

Fig. 1.	ISSDK architecture	.4
Fig. 2.	Freedom Sensor Toolbox development	
	environment	.5
Fig. 3.	MCUXpresso SDK Builder webpage	6
Fig. 4.	FRDM-K22F-A8974 kit SDK package	
	directory structure	7
Fig. 5.	Directory structure showing frdmk22f base	
	project and frdmk22f_a8974 (ISSDK)	
	project folders	.7
Fig. 6.	Middleware components available as part	
-	of SDK package	7
		1

Fig. 7.	FRDM-K22F-A8974 kit configuration and SDK package download	8
Fig. 8.	Creating a standalone project in the workspace	
Fig. 9.	Terminal emulation program with serial port set	
Fig. 10.	Debug console output	
Fig. 11.	SDK import wizard for pedometer project	
Fig. 12.	Pedometer algorithm example project in MCUXpresso IDE	11
Fig. 13.	Serial capture program displaying step counts	12

Getting started with IoT Sensing SDK (ISSDK) middleware

Contents

1	Prerequisites	3
2	Overview	
2.1	ISSDK architecture	3
3	NXP Freedom Sensor Toolbox Sensor	
	Development Ecosystem	4
4	Project deployment	
4.1	MCUXpresso SDK Builder	5
4.2	Deployment directory structure	
5	Build and run a sensor driver example	7
6	Build and run pedometer algorithm	
	example	10
7	Legal information	13

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

© 2022 NXP B.V.

All rights reserved.

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

Date of release: 7 July 2022 Document identifier: ISSDKGETSTARTEDUG