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

Revision history

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

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 www.nxp.com/iotsensingsdk.

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.

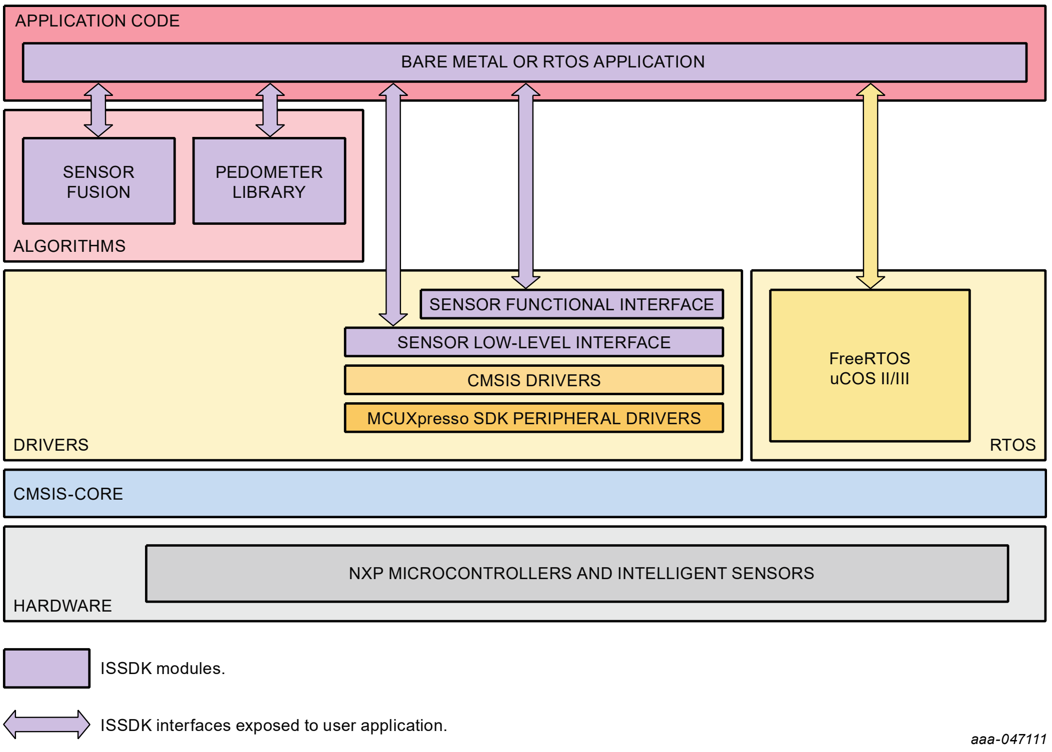


Figure 1. ISSDK architecture

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.

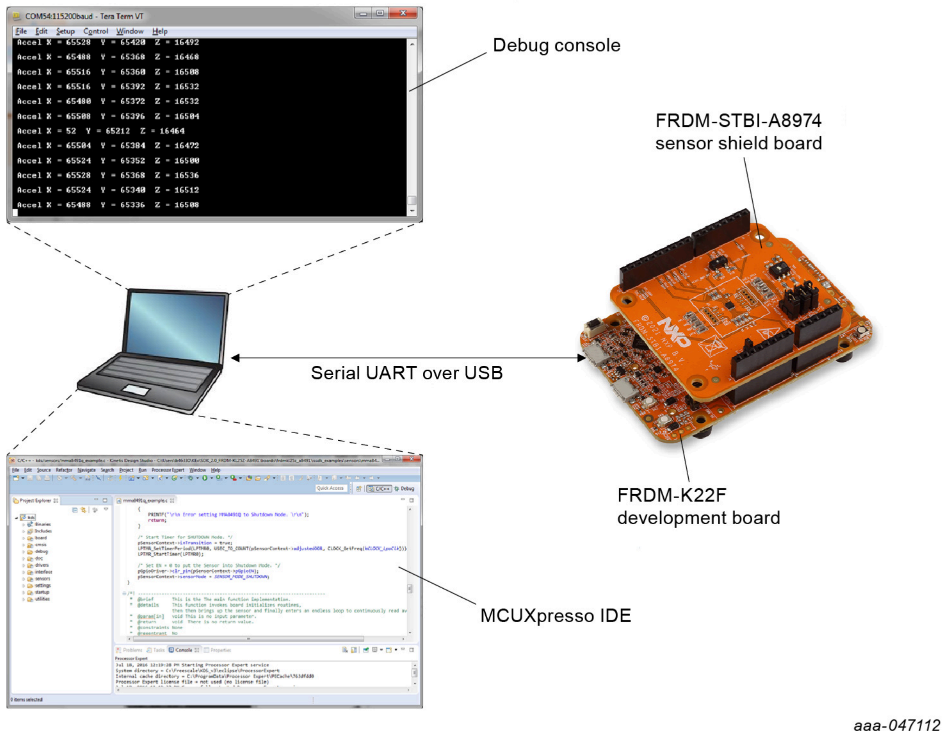


Figure 2. Freedom Sensor Toolbox development environment

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.

4 Project deployment

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.

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.

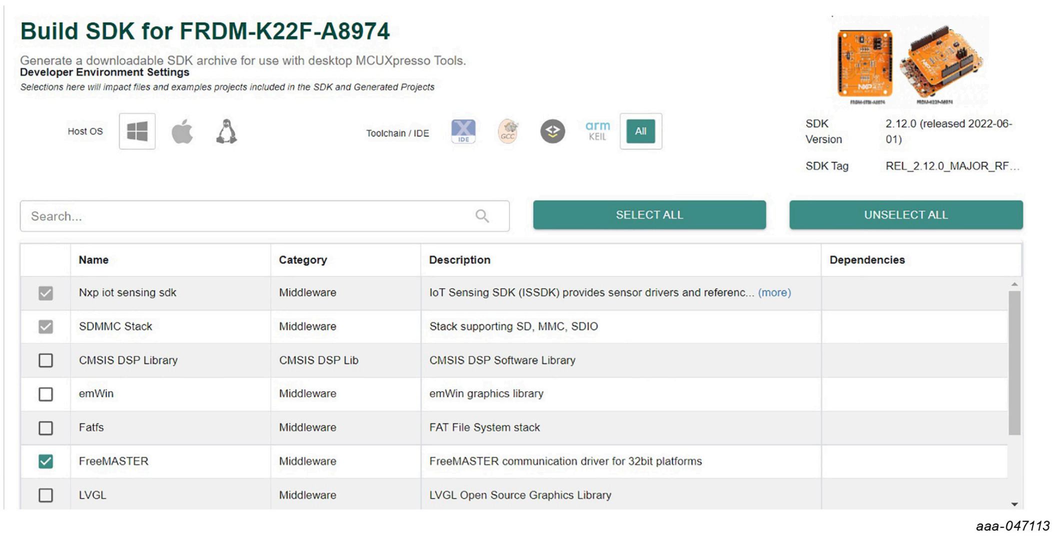


Figure 3. MCUXpresso SDK Builder webpage

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.

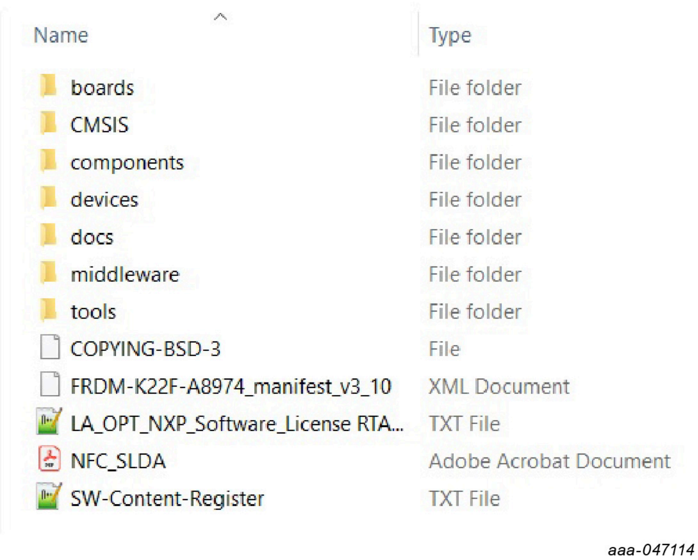


Figure 4. FRDM-K22F-A8974 kit SDK package directory structure

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.

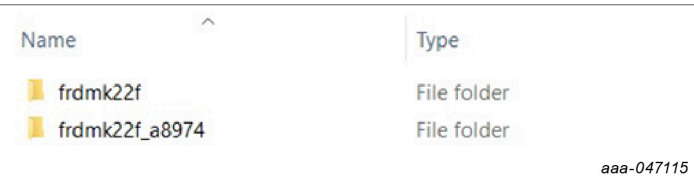


Figure 5. Directory structure showing frdmk22f base project and frdmk22f_a8974 (ISSDK) project folders

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

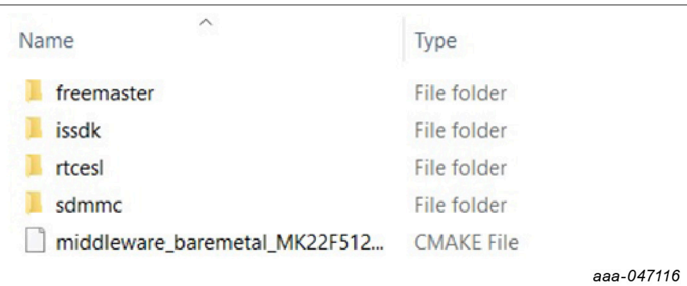


Figure 6. Middleware components available as part of SDK package

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.

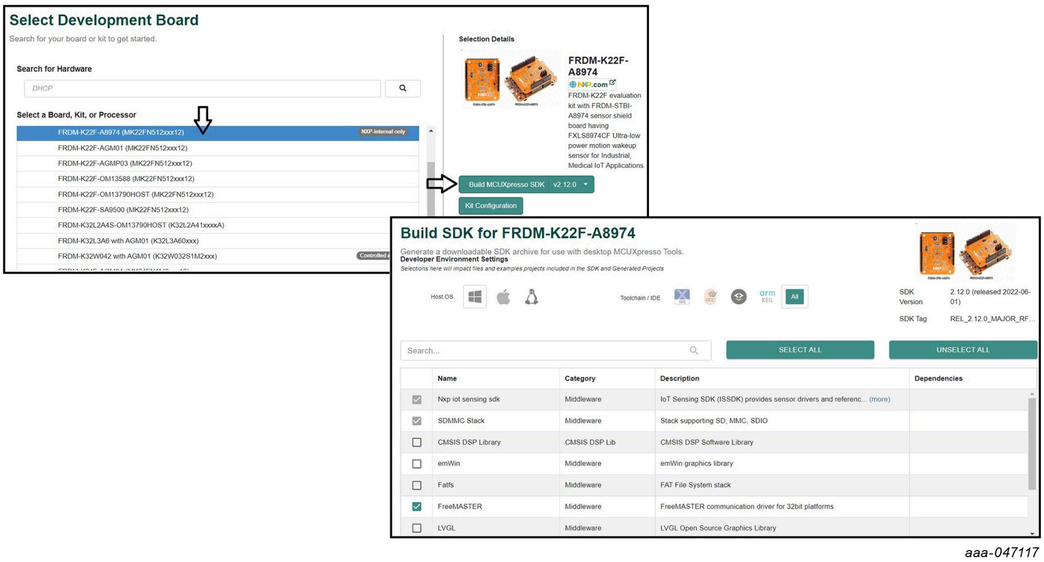
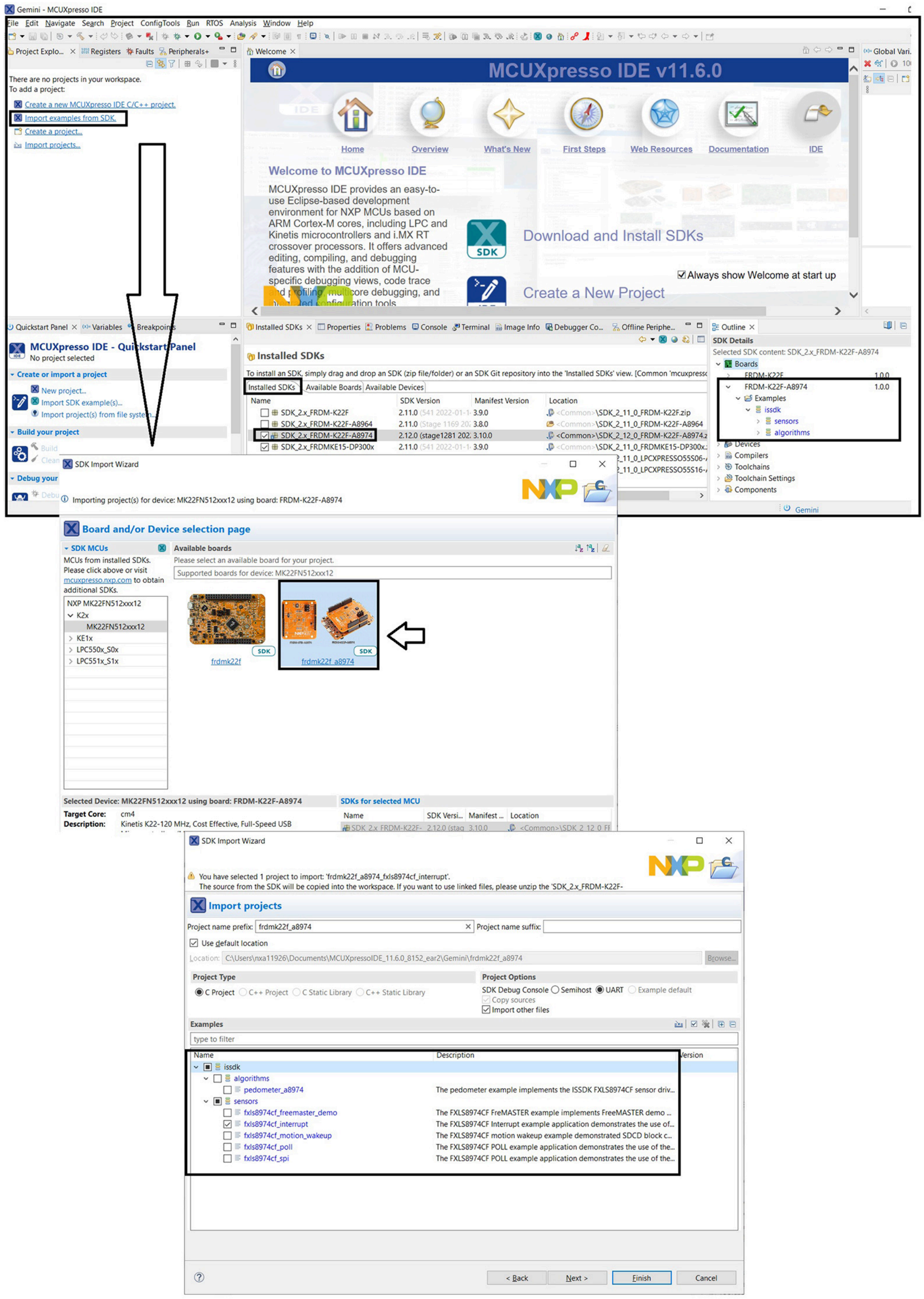


Figure 7. FRDM-K22F-A8974 kit configuration and SDK package download

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.



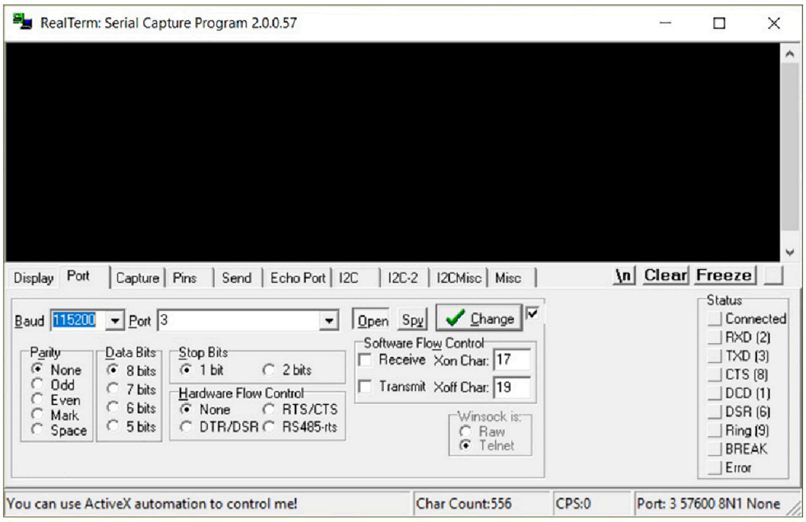
aaa-047118

Figure 8. Creating a standalone project in the workspace

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.

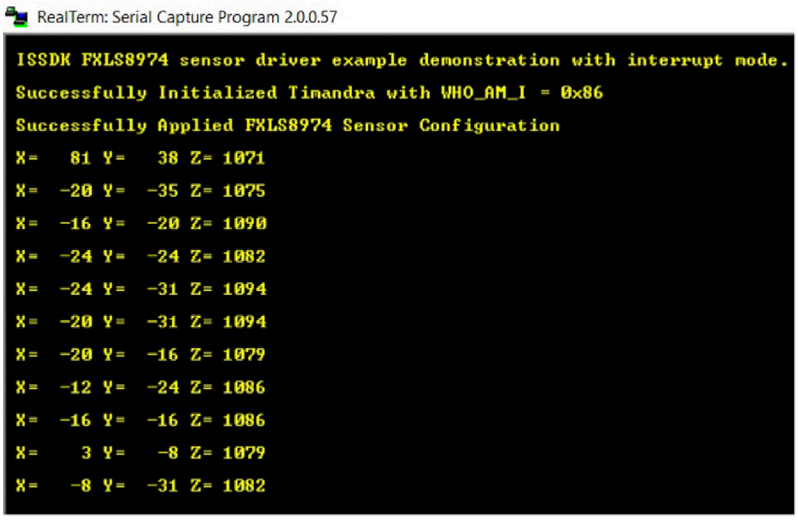
Next, start a terminal emulation program with the serial port set as shown in Figure 9.



aaa-047119

Figure 9. Terminal emulation program with serial port set

Figure 10 displays the debug console window and output.



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

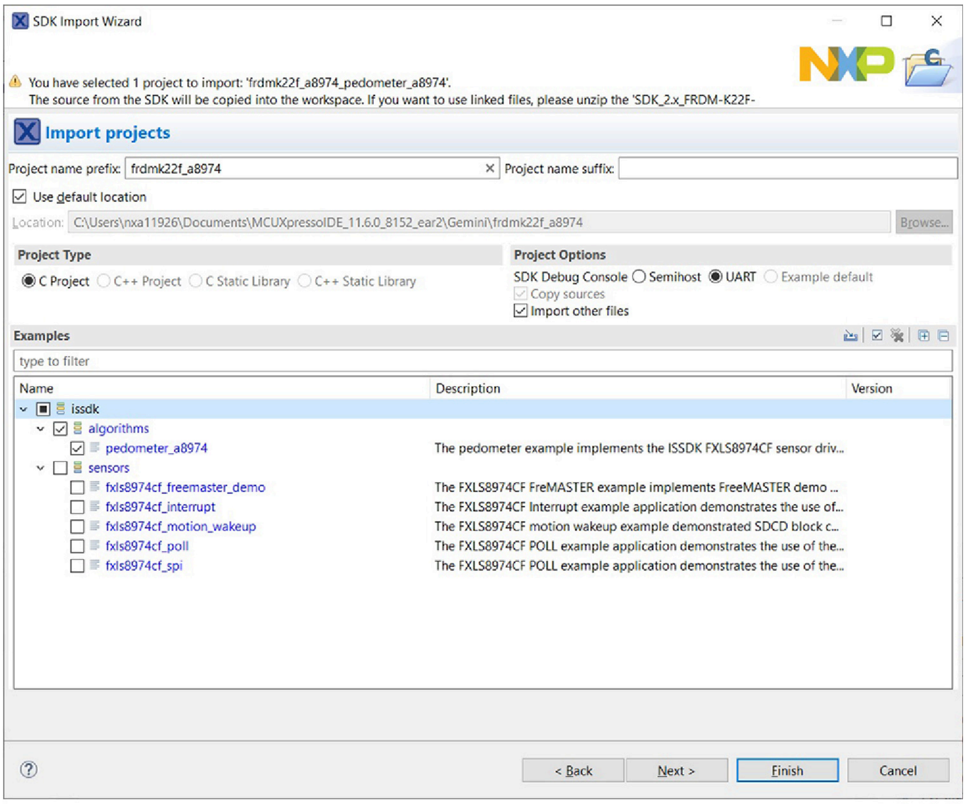


Figure 11. SDK import wizard for pedometer project

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

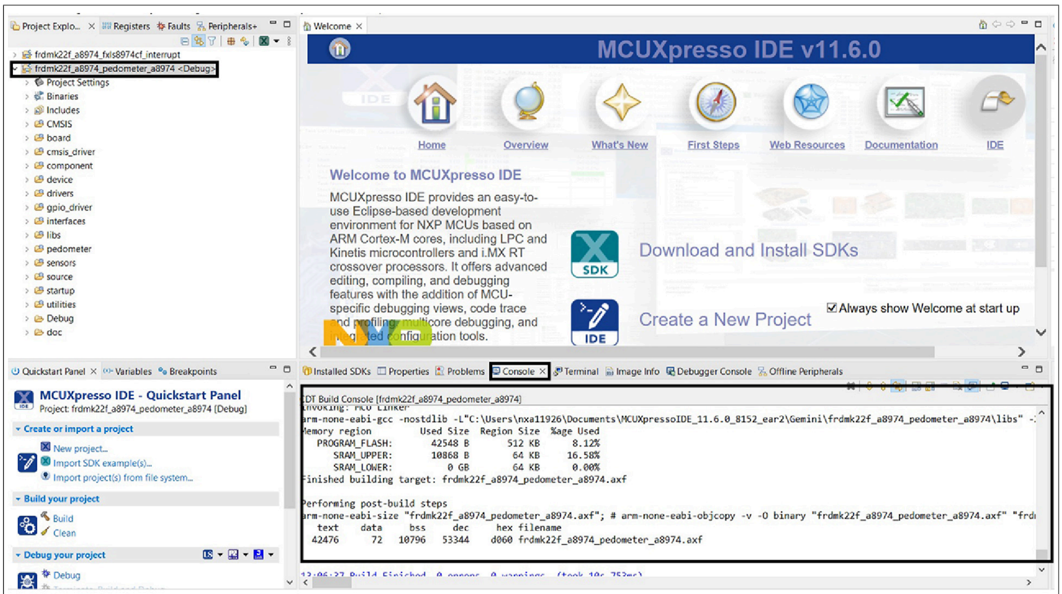


Figure 12. Pedometer algorithm example project in MCUXpresso IDE

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.*

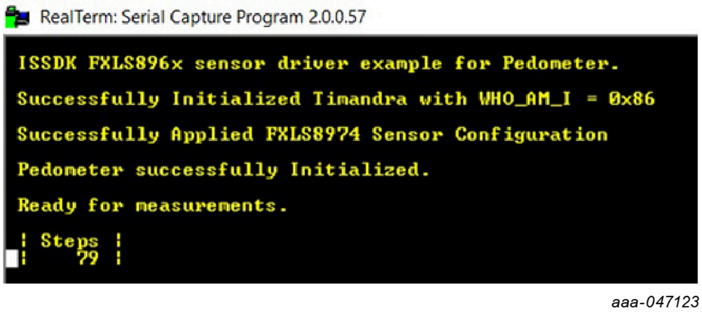


Figure 13. Serial capture program displaying step counts

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.

Figures

Contents

Date of release: 7 July 2022
Document identifier: ISSDKGETSTARTEDUG