IoT Sensing SDK

Getting started with IoT Sensing SDK (ISSDK) v1.0 middleware

Rev. 1.0 — 4 August 2016 **User guide**

Document information

Info	Content
Abstract	Getting started with IoT Sensing SDK (ISSDK) v1.0 middleware

Revision history

Rev	Date	Description
1.0	20160803	Initial public release

Contact information

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

1. Prerequisites

This document assumes the following knowledge by the user prior to attempting to use the ISSDK v1.0 middleware:

- One of the recommended IDEs has been installed on the development PC (see the release notes)
- A FRDM-K64F-AGM01 sensor kit is connected to the development PC
- User understands the debug environment setup for the Freedom family of development boards using OpenSDA or third-party debugger with their IDE of choice
- Exposure to the Kinetis SDK and Kinetis Expert website

2. Overview

The IoT Sensing SDK (ISSDK) v1.0 middleware is an optional Kinetis Expert (KEx) package designed to provide support for Kinetis MCU projects using NXP sensors. ISSDK relies on the SDK 2.0 drivers and project release infrastructure to create a unified user experience. ISSDK v1.0 combines a set of robust sensor drivers and algorithms along with example applications to allow a customer to get started with using NXP sensors quickly.

2.1 ISSDK v1.0 Architecture

Fig 1 shows the high-level *layer cake* architecture of the ISSDK v1.0 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.'s CMSIS Driver APIs). ISSDK is designed to allow users to start with as small a production footprint (memory and CPU load) as is practical for their particular application. This is typically done by selecting the Bare Metal option; however, some applications may prefer using one of the RTOSs supplied with SDK 2.0.

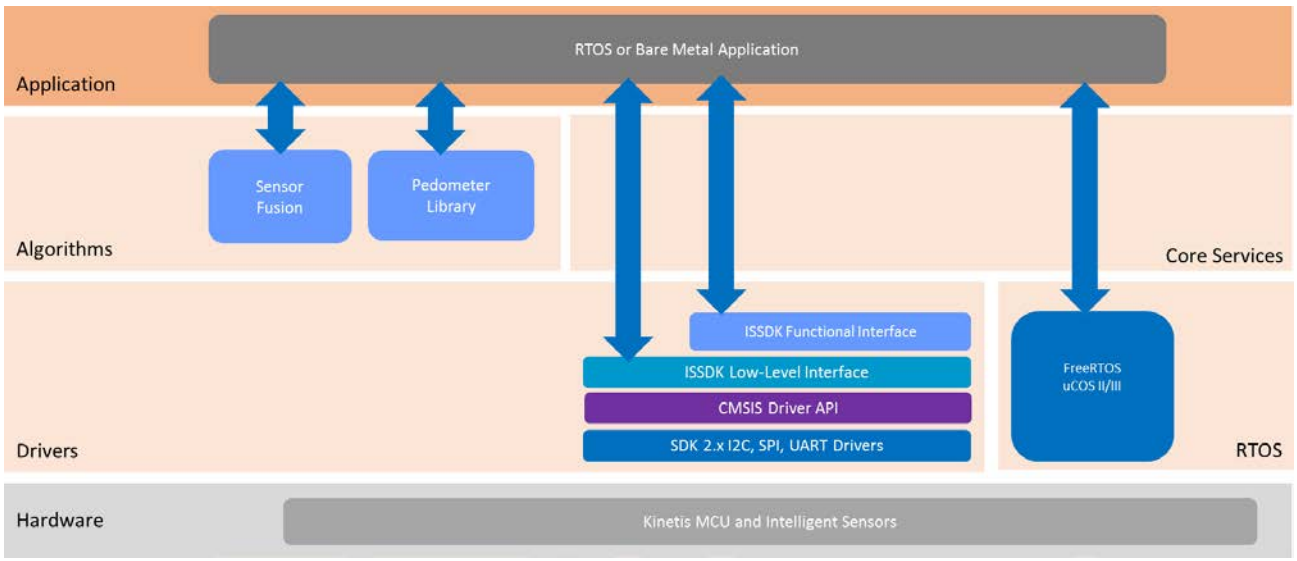


Fig 1. ISSDK v1.0 Architecture

In the following sections, this guide will focus on how ISSDK can be deployed via KEx for a specific Freedom Sensor Toolbox target called the FRDM-K64F-AGM01. This target combines the Kinetis FRDM-K64F development board with a FRDM-STBC-AGM01 sensor shield to provide a stand-alone, low cost sensor development platform.

3. NXP Freedom Sensor Toolbox: Sensor Development Ecosystem

NXP Sensors provide a sensor development ecosystem called the Freedom Sensor Toolbox. This ecosystem is designed to provide for hardware and software solutions that enable customers to evaluate and prototype with sensors quickly and easily. ISSDK v1.0 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 figure below shows how the Freedom Sensor Toolbox development hardware can be used to explore the ISSDK v1.0 software. In this example, the Kinetis Design Studio IDE is used to compile, load and launch an existing project into the FRDM-K64F-AGM01 kit. The customer may then launch a terminal emulator to examine the debug console output provided for many ISSDK v1.0 projects.

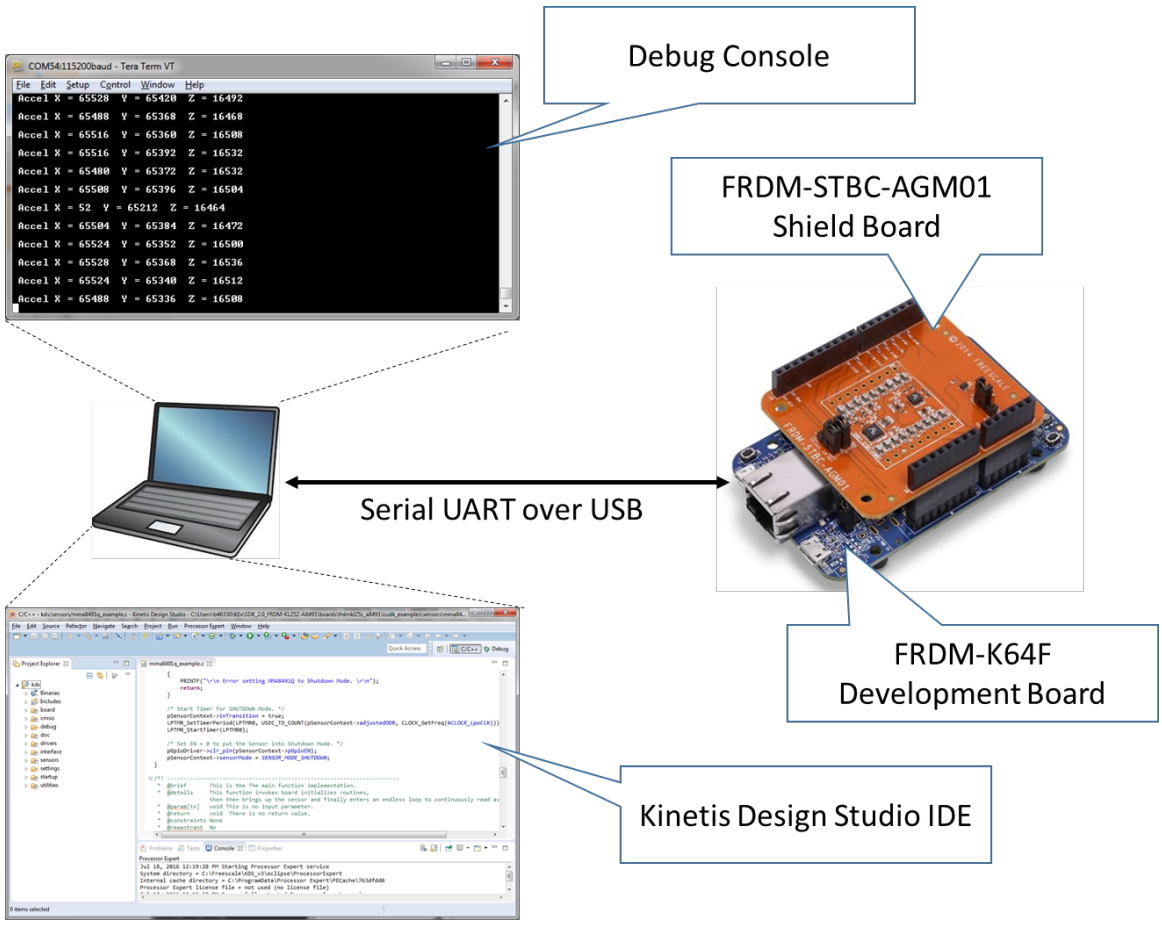


Fig 2. Freedom Sensor Toolbox development environment

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 to use the FRDM-K64F-AGM01 development kit with the ISSDK enablement software for it.

4. Project Deployment

ISSDK 1.0 is fully integrated into the Kinetis Expert (KEx) delivery system. KEx includes both cloud and locally based tools to collect and build projects from the Kinetis SDK repositories. KSDK 2.0 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 KSDK (2.0 only), the supported IDEs, (KDS, IAR, Atollic, Keil, GCC), and the target Host OS (Windows, Mac, or Linux).

4.1 Kinetis Expert

Kinetis Expert (KEx) is a cloud-based system used to build Kinetis SDK 2.0 packages. ISSDK is an optional component that can be deployed by KEx in two ways:

- If the customer selects a FRDM sensor kit, such as the FRDM-K64F-AGM01, 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.

It should be noted that in both cases the SDK 2.0 drivers and example applications will also be deployed alongside the ISSDK files.

The figure below shows the KEx environment for deploying ISSDK (see http://kex.nxp.com). In this example, the customer has selected the FRDM-K64F-AGM01 kit, the Kinetis Design Studio 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. At this point, the customer may download the package (a zip file) and deploy it into their local system.

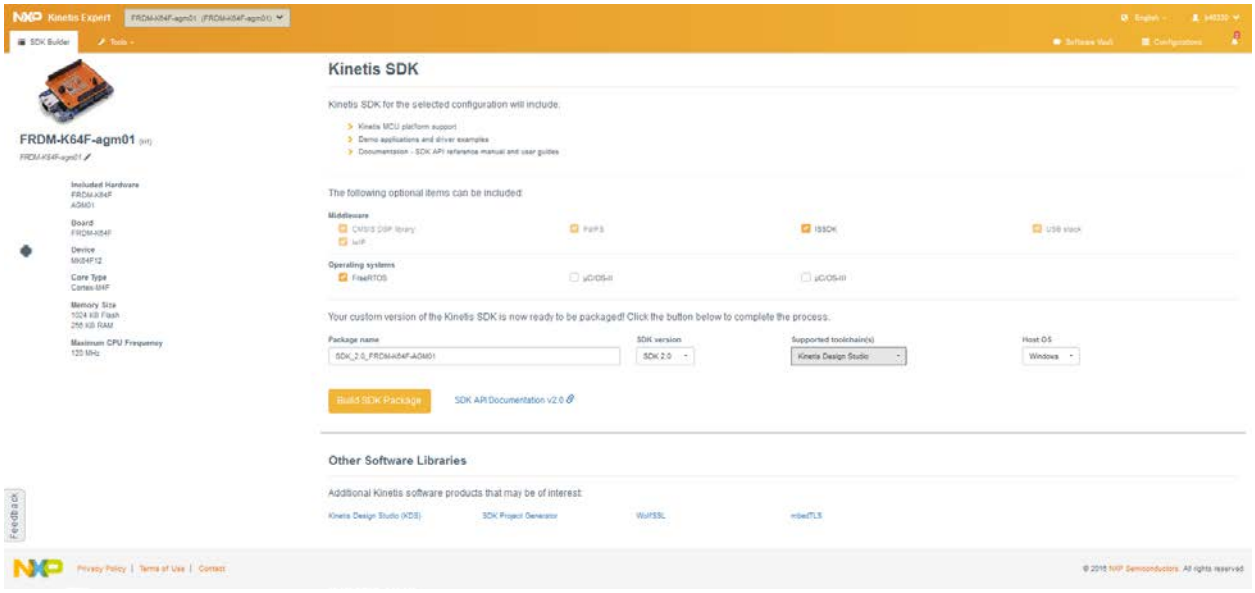
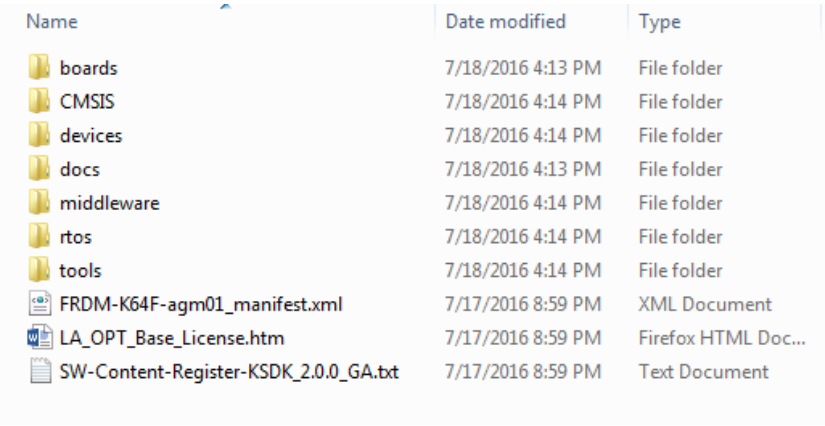


Fig 3. Kinetis Expert web page

4.2 Deployment Directory Structure

Once the KEx package has been downloaded, the user can unzip the package on their local machine. The directory structure appears as follows:

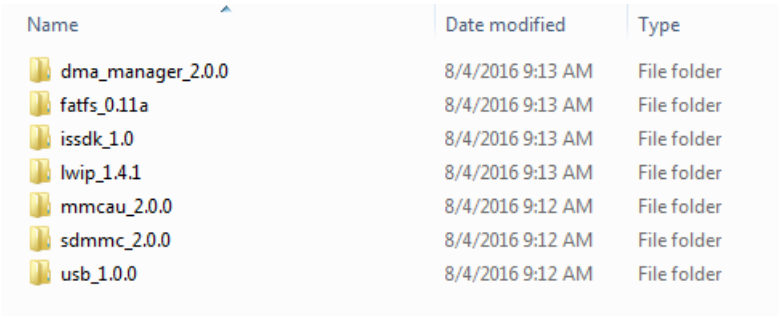


Name	Date modified	Type
boards	7/18/2016 4:13 PM	File folder
CMSIS	7/18/2016 4:14 PM	File folder
devices	7/18/2016 4:14 PM	File folder
docs	7/18/2016 4:13 PM	File folder
middleware	7/18/2016 4:14 PM	File folder
rtos	7/18/2016 4:14 PM	File folder
tools	7/18/2016 4:14 PM	File folder
FRDM-K64F-agm01_manifest.xml	7/17/2016 8:59 PM	XML Document
LA_OPT_Base_License.htm	7/17/2016 8:59 PM	Firefox HTML Doc...
SW-Content-Register-KSDK_2.0.0_GA.txt	7/17/2016 8:59 PM	Text Document

The CMSIS, devices, docs, rtos, and tools directories are unchanged from standard KSDK 2.0 deployments. ISSDK v1.0 appears as new targets in the /boards directory. In this example the frdmk64f_agm01 (ISSDK) projects are created (as well as the base projects for the frdmk64f) as follows:

Name	Date modified	Type
frdmk64f	7/18/2016 4:14 PM	File folder
frdmk64f_agm01	7/18/2016 4:13 PM	File folder

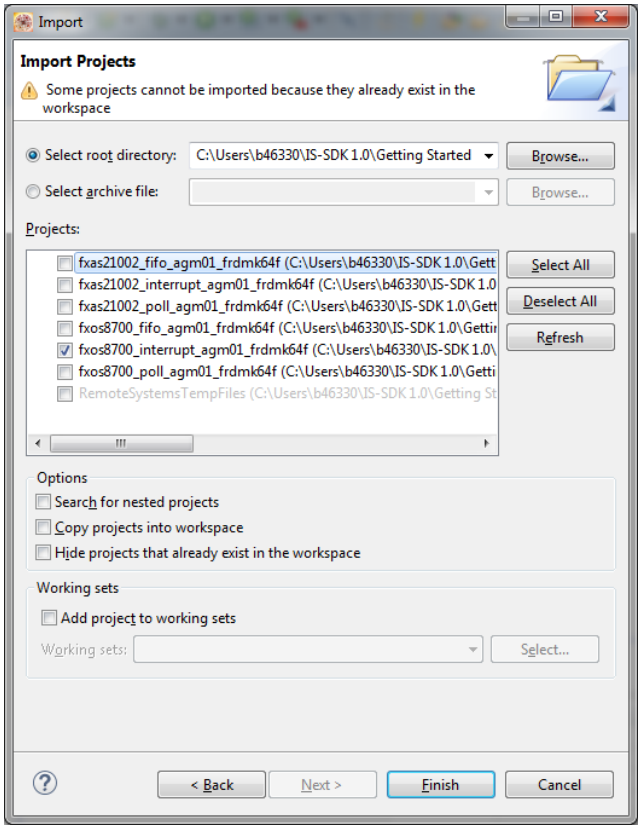
In addition, a new middleware library is created that contains the ISSDK drivers, algorithms and other support files as follows:



Name	Date modified	Type
dma_manager_2.0.0	8/4/2016 9:13 AM	File folder
fatfs_0.11a	8/4/2016 9:13 AM	File folder
issdk_1.0	8/4/2016 9:13 AM	File folder
lwip_1.4.1	8/4/2016 9:13 AM	File folder
mmcau_2.0.0	8/4/2016 9:12 AM	File folder
sdmmc_2.0.0	8/4/2016 9:12 AM	File folder
usb_1.0.0	8/4/2016 9:12 AM	File folder

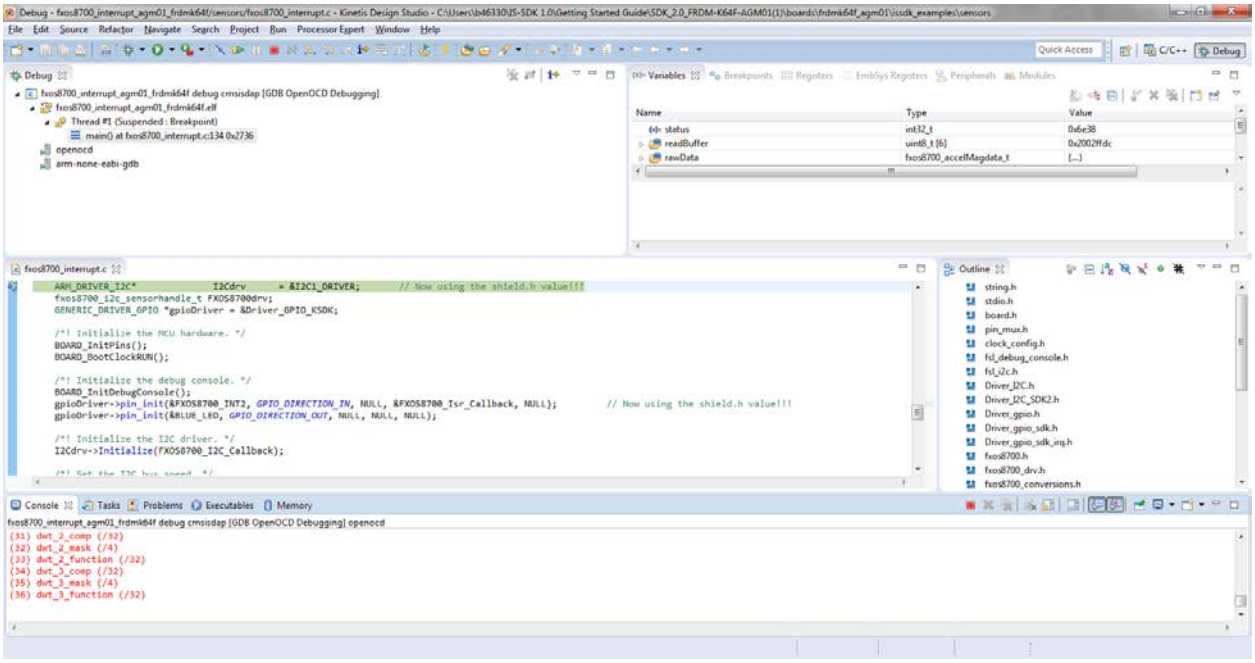
5. Build and Run a Sensor Driver Example

As a first example, start KDS 3.2 and import one of the Sensor Driver Example projects. Start KDS 3.2. Import an existing project into KDS (File->Import->General->Existing Project) then select Browse and OK. Deselect the projects until your screen looks as follows:



Click Finish and you should see the project appear in the left pane of the KDS. Do a Clean and Build the project. It should build with no errors.

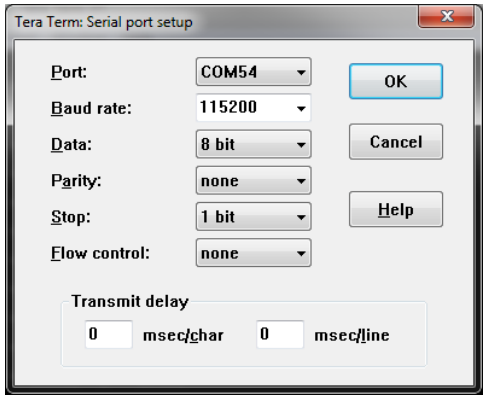
Create a Debug Configuration (depending on the OpenSDA driver installed on your Freedom board). After starting the program in the debugger, your screen will switch to the Debug display and look similar to the following:



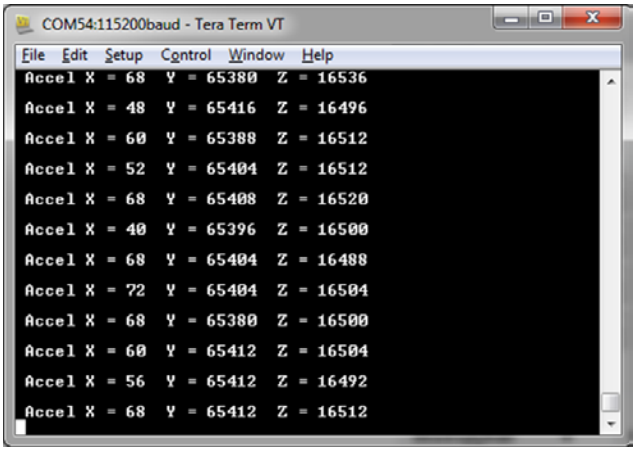
Notice that the code is ready to start in the file fxos8700_interrupt.c, which is the main application for this example.

Start the program execution. The blue LED will begin to flash on the FRDM-K64F board.

Next, start a terminal emulation program with the serial port set as follows:



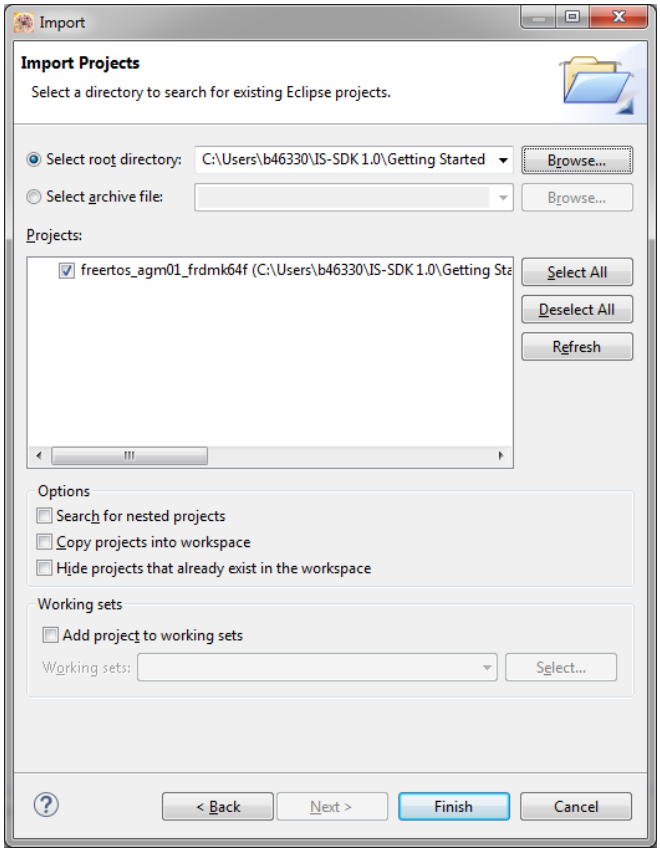
The debug console output should appear as follows:



In this example, each *data ready* interrupt from the FXOS8700 triggers the application to read the raw X-, Y-, Z-axis accelerometer values. These raw values are then converted to 16-bit unsigned integers and output in real time to the debug console.

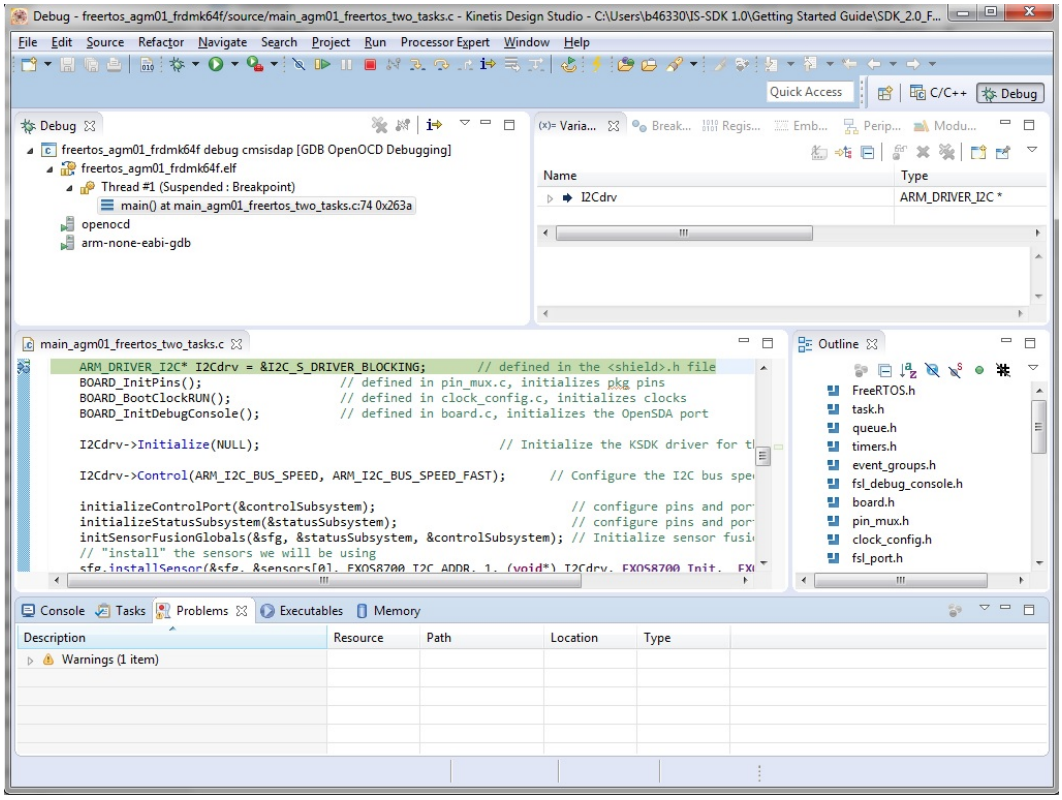
6. Build and Run Sensor Fusion

This example uses the same codebase downloaded in the previous section for the FRDM-K64F-AGM01 board. Start KDS 3.2 and import a Sensor Fusion Application project. Start KDS 3.2 and import an existing project into KDS (File->Import->General ->Existing Project) from **<install_dir>\boards\frdmk64f_agm01\issdk_examples\algorithms\sensorfusion\freertos_agm01\kds**, then select Browse and OK. Your screen looks as follows:



Click Finish and you should see the project appear in the left hand pane of the KDS. Do a Clean and Build the project. It should build with no errors.

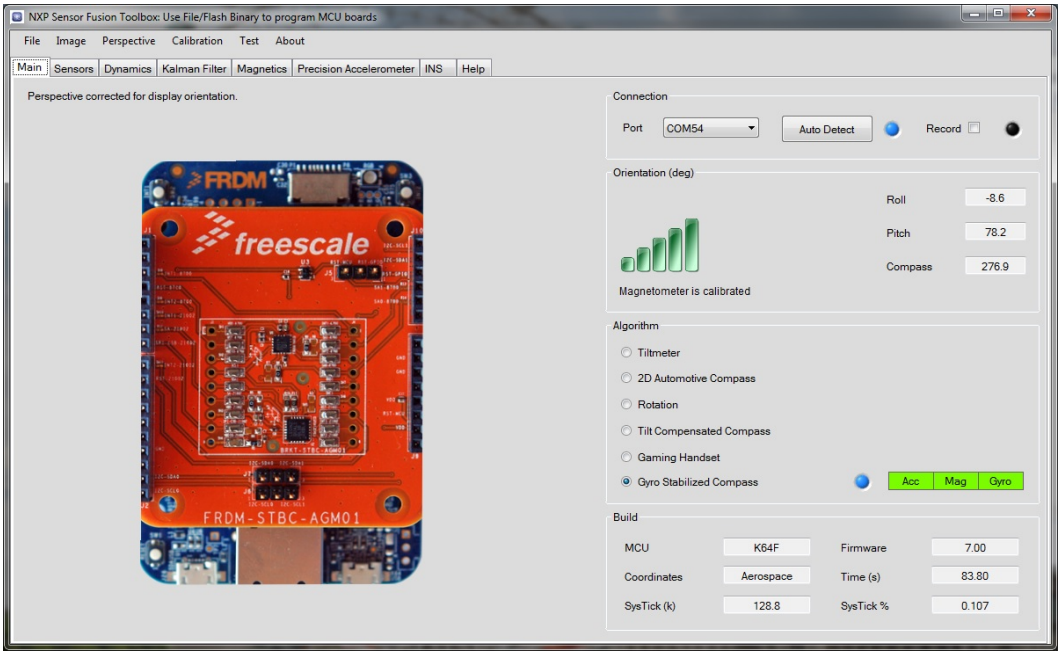
Create a Debug Configuration (depending on the OpenSDA driver installed on your Freedom board). After starting the program in the debugger, your screen will switch to the Debug display and look similar to the following:



Notice that the code is ready to start in the file **main_agm01_freertos_two_tasks.c**, which is the main application for this example.

Start the program execution. The green LED will begin to flash on the FRDM-K64F board.

You can now install the Sensor Fusion GUI application in order to visualize the operation of the Sensor Fusion application. Go to http://nxp.com/sensorfusion, download the latest Sensor Fusion for Windows GUI and install it. Launch the Sensor Fusion GUI. Select the Port from the pull-down menu. The GUI main screen should look similar to this:



This completes the exercise to run the ISSDK 1.0 Sensor Fusion v7.00 project.

7. Legal information

7.1 Definitions

Draft — The document is a draft version only. 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 herein and shall have no liability for the consequences of use of such information.

7.2 Disclaimers

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp/com/SalesTermsandConditions.

7.3 Trademarks

NXP, the NXP logo, Freescale and the Freescale logo are trademarks of NXP B.V. All other product or service names are the property of their respective owners.

8. List of figures

9. Contents

Date of release: 4 August 2016
Document identifier: ISSDKGETSTARTEDUG