IoT Sensing SDK

Getting started with IoT Sensing SDK (ISSDK) v1.5 middleware

Rev. 1.2 — 6 March 2017 **User guide**

1 Prerequisites

This document assumes the following knowledge by the user prior to attempting to use the ISSDK v1.5 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 MCUXpresso SDK Builder (previously known as Kinetis Expert)

2 Overview

The IoT Sensing SDK (ISSDK) v1.5 is an optional middleware component included in SDK packages downloaded from MCUXpresso SDK Builder (previously known as Kinetis Expert (KEx)) 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 v1.5 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. 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 v1.5 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.

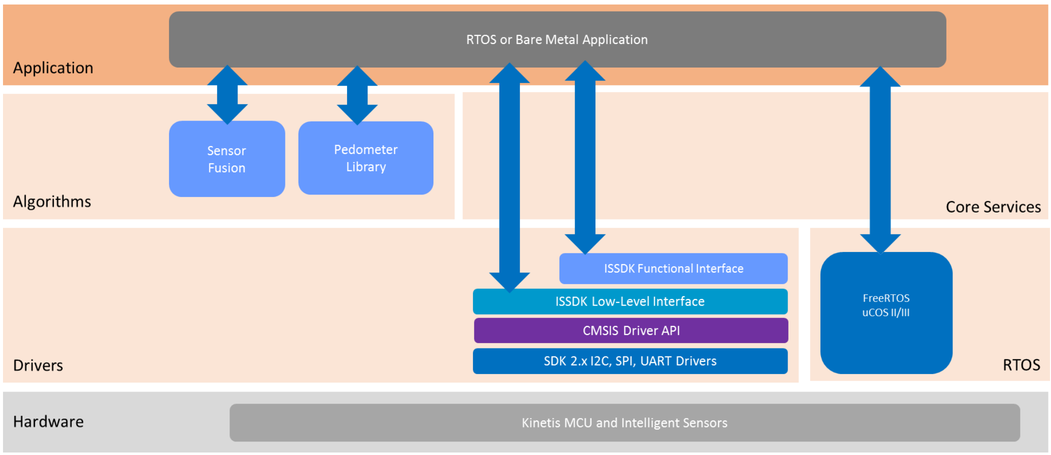

Application
RTOS or Bare Metal Application
Sensor Fusion
Pedometer Library
Algorithms
Core Services
ISSDK Functional Interface
ISSDK Low-Level Interface
CMSIS Driver API
SDK 2.x I2C, SPI, UART Drivers
FreeRTOS uCOS II/III
Drivers
RTOS
Hardware
Kinetis MCU and Intelligent Sensors

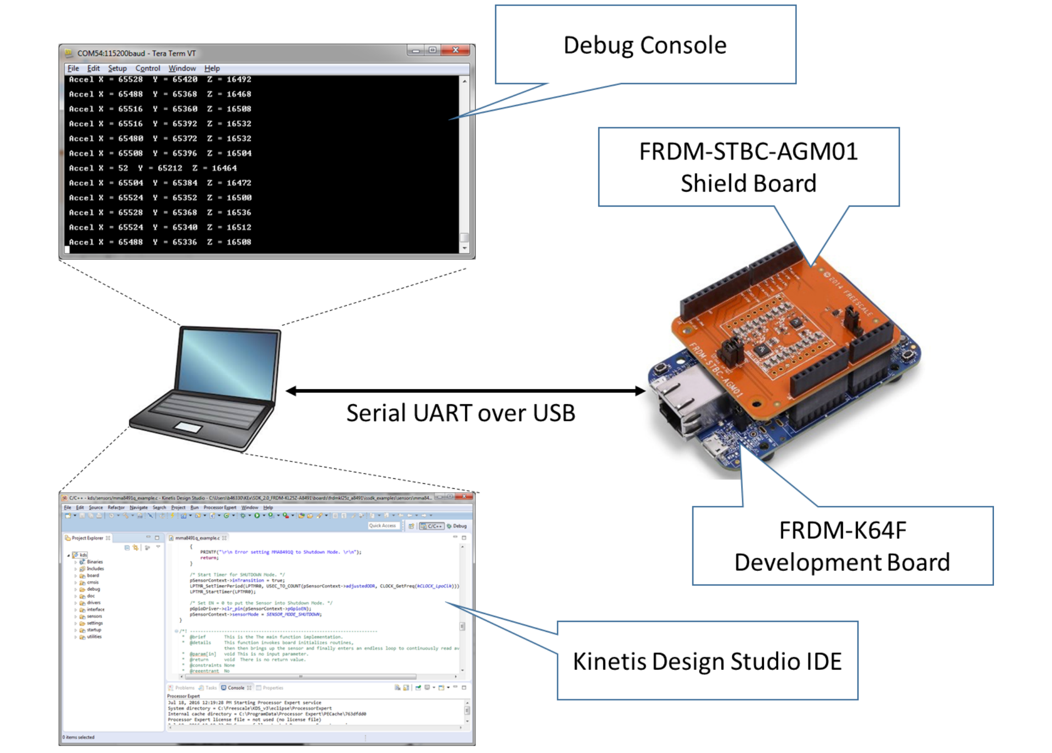
Figure 1. ISSDK v1.5 Architecture

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-K64F-AGM01. This kit 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 solutions for hardware and software that enable customers to evaluate and prototype with sensors quickly and easily. ISSDK v1.5 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 v1.5 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 v1.5 projects.



COM54:115200baud - Tera Term VT
File Edit Setup Control Window Help
Debug Console
FRDM-STBC-AGM01
Shield Board
Serial UART over USB
FRDM-K64F
Development Board
Kinetis Design Studio IDE

```
Accel X = 65528  Y = 65420  Z = 16492
Accel X = 65488  Y = 65368  Z = 16468
Accel X = 65516  Y = 65360  Z = 16508
Accel X = 65516  Y = 65392  Z = 16532
Accel X = 65480  Y = 65372  Z = 16532
Accel X = 65508  Y = 65396  Z = 16504
Accel X = 52  Y = 65212  Z = 16464
Accel X = 65504  Y = 65384  Z = 16472
Accel X = 65524  Y = 65352  Z = 16500
Accel X = 65528  Y = 65368  Z = 16536
Accel X = 65524  Y = 65340  Z = 16512
Accel X = 65488  Y = 65336  Z = 16508
```

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

4 Project deployment

ISSDK v1.5 is fully integrated into the MCUXpresso Web and SDK Builder delivery system. MCUXpresso 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 (MCUXpresso IDE (RedEye), Somnium DRT, KDS, IAR, Keil, GCC), and the target Host OS (Windows, Mac, or Linux).

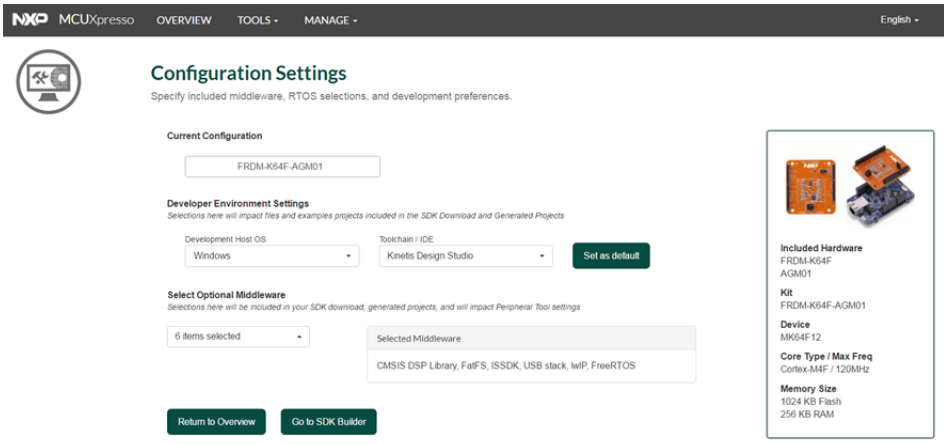
4.1 MCUXpresso Web & SDK Builder

MCUXpresso Web & SDK Builder is a cloud-based system used to build Kinetis SDK 2.0 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-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 are also deployed alongside the ISSDK files.

The following figure shows the MCUXpresso environment for deploying ISSDK (see https://mcuxpresso.nxp.com/en/configuration-settings). 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.



Configuration Settings

Specify included middleware, RTOS selections, and development preferences.

Current Configuration

FRDM-K64F-AGM01

Developer Environment Settings

Selections here will impact files and examples projects included in the SDK Download and Generated Projects

Development Host OS: Windows

Toolchain / IDE: Kinetis Design Studio

Select Optional Middleware

Selections here will be included in your SDK download, generated projects, and will impact Peripheral Tool settings

6 items selected

Selected Middleware

CMSIS DSP Library, FatFS, ISSDK, USB stack, lwIP, FreeRTOS

Included Hardware
FRDM-K64F
AGM01

Kit
FRDM-K64F-AGM01

Device
MK64F12

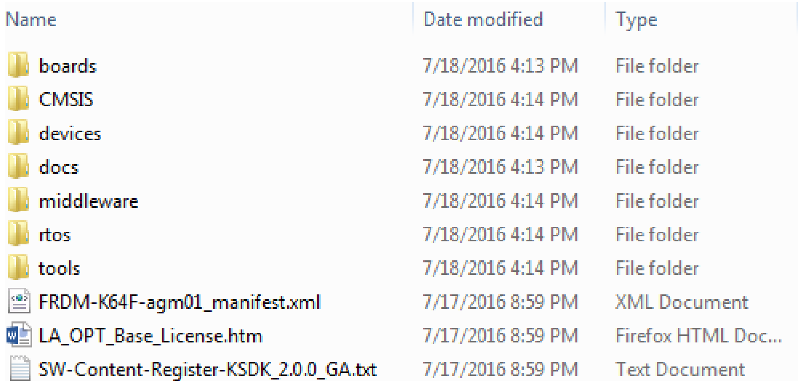
Core Type / Max Freq
Cortex-M4F / 120MHz

Memory Size
1024 KB Flash
256 KB RAM

Figure 3. MCUXpresso SDK Builder web page

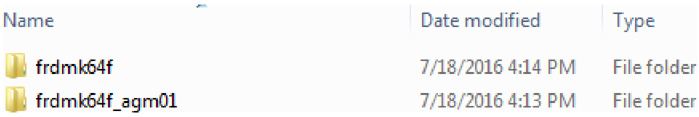
4.2 Deployment directory structure

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



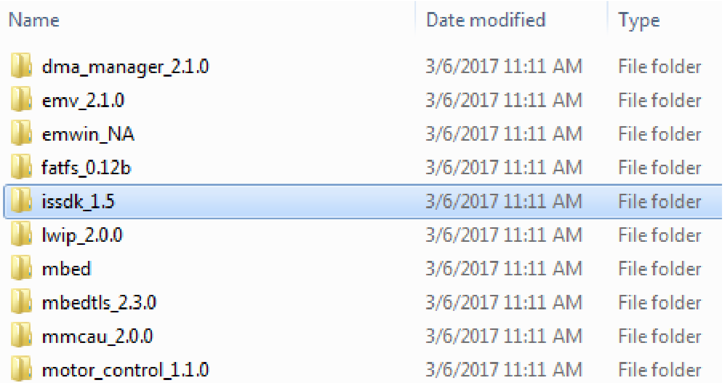
Name	Date modified	Type
boards	7/18/2016 4:13 PM	File folder
CMSIS	7/18/2016 4:14 PM	File folder
devices	7/18/2016 4:14 PM	File folder
docs	7/18/2016 4:13 PM	File folder
middleware	7/18/2016 4:14 PM	File folder
rtos	7/18/2016 4:14 PM	File folder
tools	7/18/2016 4:14 PM	File folder
FRDM-K64F-agm01_manifest.xml	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 v1.5 projects appear 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

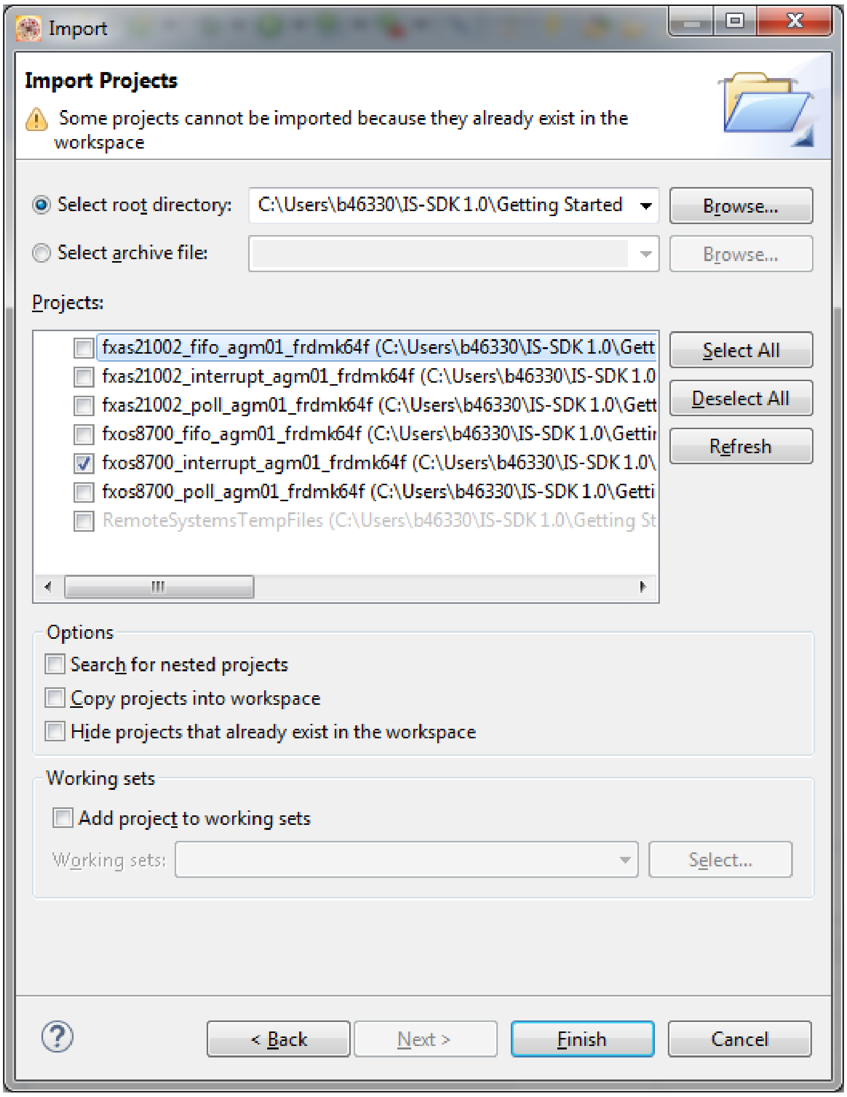
In addition, a new middleware library is created that contains the ISSDK drivers, algorithms and other support files as follows:



Name	Date modified	Type
dma_manager_2.1.0	3/6/2017 11:11 AM	File folder
emv_2.1.0	3/6/2017 11:11 AM	File folder
emwin_NA	3/6/2017 11:11 AM	File folder
fatfs_0.12b	3/6/2017 11:11 AM	File folder
issdk_1.5	3/6/2017 11:11 AM	File folder
lwip_2.0.0	3/6/2017 11:11 AM	File folder
mbed	3/6/2017 11:11 AM	File folder
mbedtls_2.3.0	3/6/2017 11:11 AM	File folder
mmcau_2.0.0	3/6/2017 11:11 AM	File folder
motor_control_1.1.0	3/6/2017 11:11 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). Click **Browse**, then locate and select the folder containing the example projects, and then click **OK**. In the projects area, deselect the projects until your screen looks as follows:



Import

Import Projects

⚠ Some projects cannot be imported because they already exist in the workspace

● Select root directory: C:\Users\b46330\IS-SDK 1.0\Getting Started

○ Select archive file:

Projects:

- □ fxas21002_fifo_agm01_frdmk64f (C:\Users\b46330\IS-SDK 1.0\Gett
- □ fxas21002_interrupt_agm01_frdmk64f (C:\Users\b46330\IS-SDK 1.0
- □ fxas21002_poll_agm01_frdmk64f (C:\Users\b46330\IS-SDK 1.0\Gett
- □ fxos8700_fifo_agm01_frdmk64f (C:\Users\b46330\IS-SDK 1.0\Gettir
- ☑ fxos8700_interrupt_agm01_frdmk64f (C:\Users\b46330\IS-SDK 1.0\
- □ fxos8700_poll_agm01_frdmk64f (C:\Users\b46330\IS-SDK 1.0\Getti
- □ RemoteSystemsTempFiles (C:\Users\b46330\IS-SDK 1.0\Getting St

Options

- □ Search for nested projects
- □ Copy projects into workspace
- □ Hide projects that already exist in the workspace

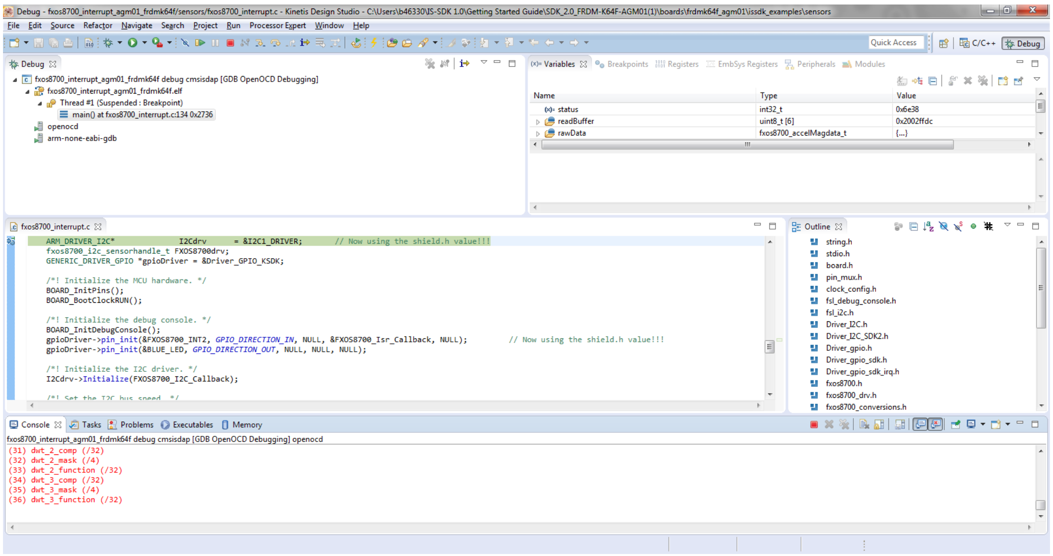
Working sets

- □ Add project to working sets

Working sets:

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 switches to the Debug display and looks similar to the following:



Debug - fxos8700_interrupt_agm01_frdmk64f/sensors/fxos8700_interrupt.c - Kinetis Design Studio - C:\Users\b46330\IS-SDK 1.0\Getting Started Guide\SDK_2.0_FRDM-K64F-AGM01(1)\boards\frdmk64f_agm01\issdk_examples\sensors

```
fxos8700_interrupt_agm01_frdmk64f debug cmsisdap [GDB OpenOCD Debugging]
  fxos8700_interrupt_agm01_frdmk64f.elf
    Thread #1 (Suspended : Breakpoint)
      main() at fxos8700_interrupt.c:134 0x2736
  openocd
  arm-none-eabi-gdb
```

Name	Type	Value
status	int32_t	0x6e38
readBuffer	uint8_t [6]	0x2002ffdc
rawData	fxos8700_accelMagdata_t	{...}

fxos8700_interrupt.c

```
    ARM_DRIVER_I2C*              I2Cdrv       = &I2C1_DRIVER;        // Now using the shield.h value!!!
    fxos8700_i2c_sensorhandle_t FXOS8700drv;
    GENERIC_DRIVER_GPIO *gpioDriver = &Driver_GPIO_KSDK;

    /*! Initialize the MCU hardware. */
    BOARD_InitPins();
    BOARD_BootClockRUN();

    /*! Initialize the debug console. */
    BOARD_InitDebugConsole();
    gpioDriver->pin_init(&FXOS8700_INT2, GPIO_DIRECTION_IN, NULL, &FXOS8700_Isr_Callback, NULL);        // Now using the shield.h value!!!
    gpioDriver->pin_init(&BLUE_LED, GPIO_DIRECTION_OUT, NULL, NULL, NULL);

    /*! Initialize the I2C driver. */
    I2Cdrv->Initialize(FXOS8700_I2C_Callback);

    /*! Set the I2C bus speed. */
```

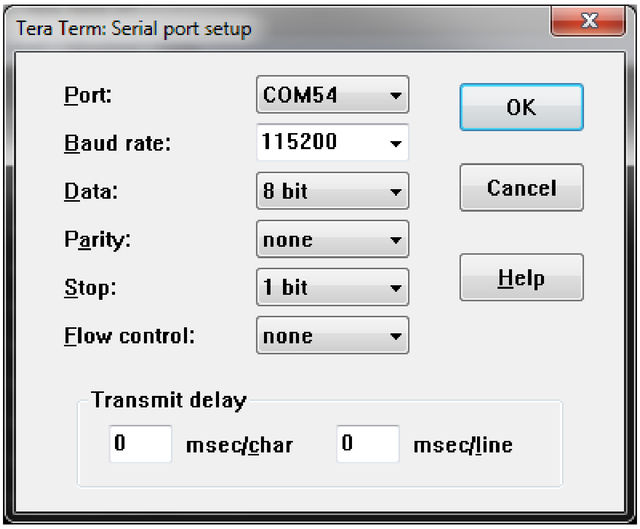
Outline:
string.h
stdio.h
board.h
pin_mux.h
clock_config.h
fsl_debug_console.h
fsl_i2c.h
Driver_I2C.h
Driver_I2C_SDK2.h
Driver_gpio.h
Driver_gpio_sdk.h
Driver_gpio_sdk_irq.h
fxos8700.h
fxos8700_drv.h
fxos8700_conversions.h

```
fxos8700_interrupt_agm01_frdmk64f debug cmsisdap [GDB OpenOCD Debugging] openocd
(31) dwt_2_comp (/32)
(32) dwt_2_mask (/4)
(33) dwt_2_function (/32)
(34) dwt_3_comp (/32)
(35) dwt_3_mask (/4)
(36) dwt_3_function (/32)
```

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 begins to flash on the FRDM-K64F board.

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



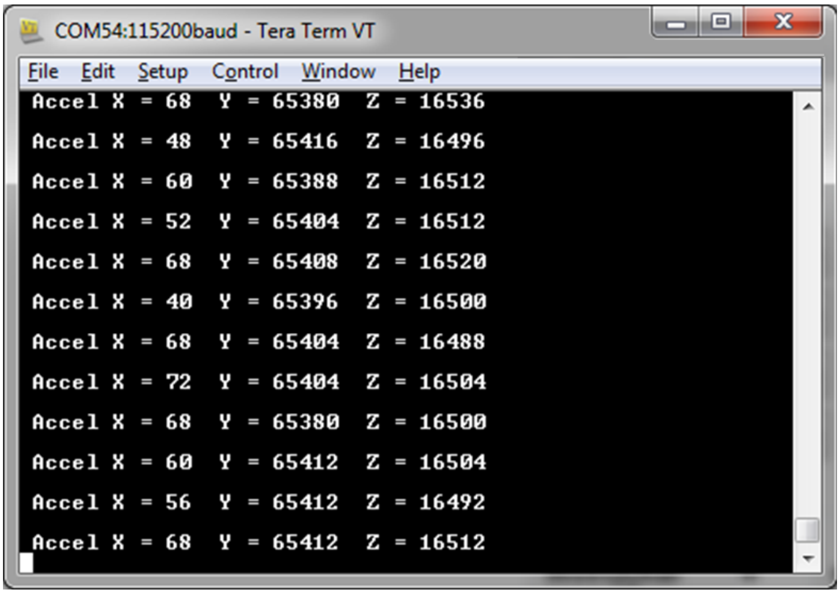
Tera Term: Serial port setup

Port:	COM54
Baud rate:	115200
Data:	8 bit
Parity:	none
Stop:	1 bit
Flow control:	none

Transmit delay

0 msec/char 0 msec/line

The debug console output appears as follows:


COM54:115200baud - Tera Term VT
File Edit Setup Control Window Help

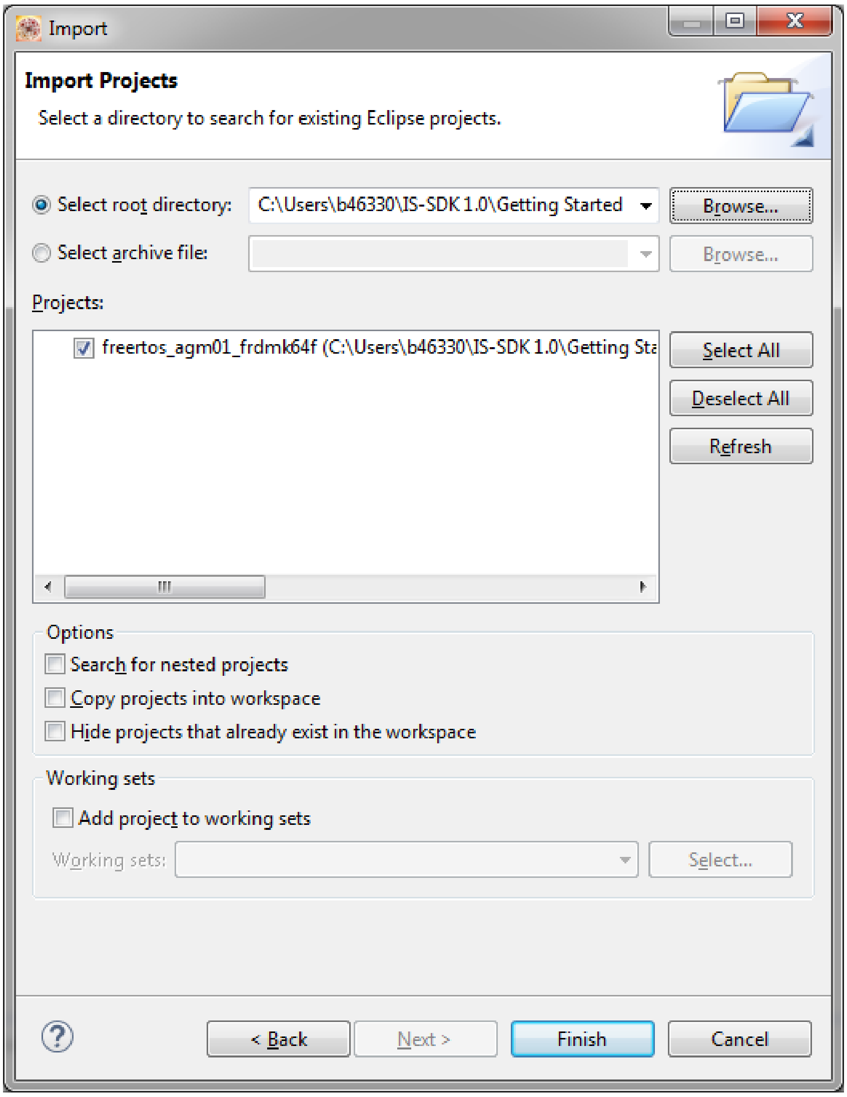
```
Accel X = 68  Y = 65380  Z = 16536
Accel X = 48  Y = 65416  Z = 16496
Accel X = 60  Y = 65388  Z = 16512
Accel X = 52  Y = 65404  Z = 16512
Accel X = 68  Y = 65408  Z = 16520
Accel X = 40  Y = 65396  Z = 16500
Accel X = 68  Y = 65404  Z = 16488
Accel X = 72  Y = 65404  Z = 16504
Accel X = 68  Y = 65380  Z = 16500
Accel X = 60  Y = 65412  Z = 16504
Accel X = 56  Y = 65412  Z = 16492
Accel X = 68  Y = 65412  Z = 16512
```

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 then click **OK**. Your screen looks as follows:



Import Projects

Select a directory to search for existing Eclipse projects.

● Select root directory: C:\Users\b46330\IS-SDK 1.0\Getting Started

○ Select archive file:

Projects:

☑ freertos_agm01_frdmk64f (C:\Users\b46330\IS-SDK 1.0\Getting Sta

Options

□ Search for nested projects

□ Copy projects into workspace

□ Hide projects that already exist in the workspace

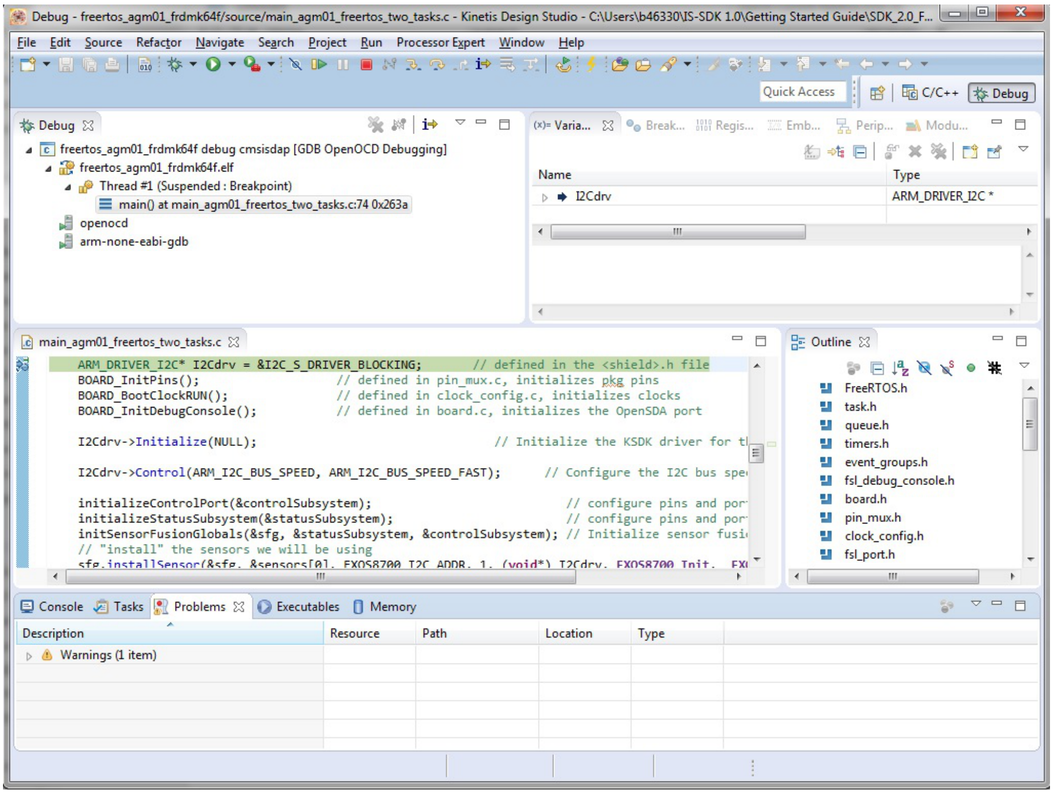
Working sets

□ Add project to working sets

Working sets:

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 switches to the Debug display and looks similar to the following:


Debug - freertos_agm01_frdmk64f/source/main_agm01_freertos_two_tasks.c - Kinetis Design Studio - C:\Users\b46330\IS-SDK 1.0\Getting Started Guide\SDK_2.0_F...
File Edit Source Refactor Navigate Search Project Run Processor Expert Window Help
Quick Access
C/C++
Debug
Debug
Varia...
Break...
Regis...
Emb...
Perip...
Modu...
Outline
Console
Tasks
Problems
Executables
Memory

freertos_agm01_frdmk64f debug cmsisdap [GDB OpenOCD Debugging]
- freertos_agm01_frdmk64f.elf
  - Thread #1 (Suspended : Breakpoint)
    - main() at main_agm01_freertos_two_tasks.c:74 0x263a
- openocd
- arm-none-eabi-gdb

Name	Type
I2Cdrv	ARM_DRIVER_I2C *

main_agm01_freertos_two_tasks.c

```
    ARM_DRIVER_I2C* I2Cdrv = &I2C_S_DRIVER_BLOCKING;       // defined in the <shield>.h file
    BOARD_InitPins();                             // defined in pin_mux.c, initializes pkg pins
    BOARD_BootClockRUN();                         // defined in clock_config.c, initializes clocks
    BOARD_InitDebugConsole();                     // defined in board.c, initializes the OpenSDA port

    I2Cdrv->Initialize(NULL);                                           // Initialize the KSDK driver for tl

    I2Cdrv->Control(ARM_I2C_BUS_SPEED, ARM_I2C_BUS_SPEED_FAST);      // Configure the I2C bus spe

    initializeControlPort(&controlSubsystem);                           // configure pins and por
    initializeStatusSubsystem(&statusSubsystem);                        // configure pins and por
    initSensorFusionGlobals(&sfg, &statusSubsystem, &controlSubsystem); // Initialize sensor fusi
    // "install" the sensors we will be using
    sfg.installSensor(&sfg, &sensors[0], FXOS8700_I2C_ADDR, 1, (void*) I2Cdrv, FXOS8700_Init,  FX
```

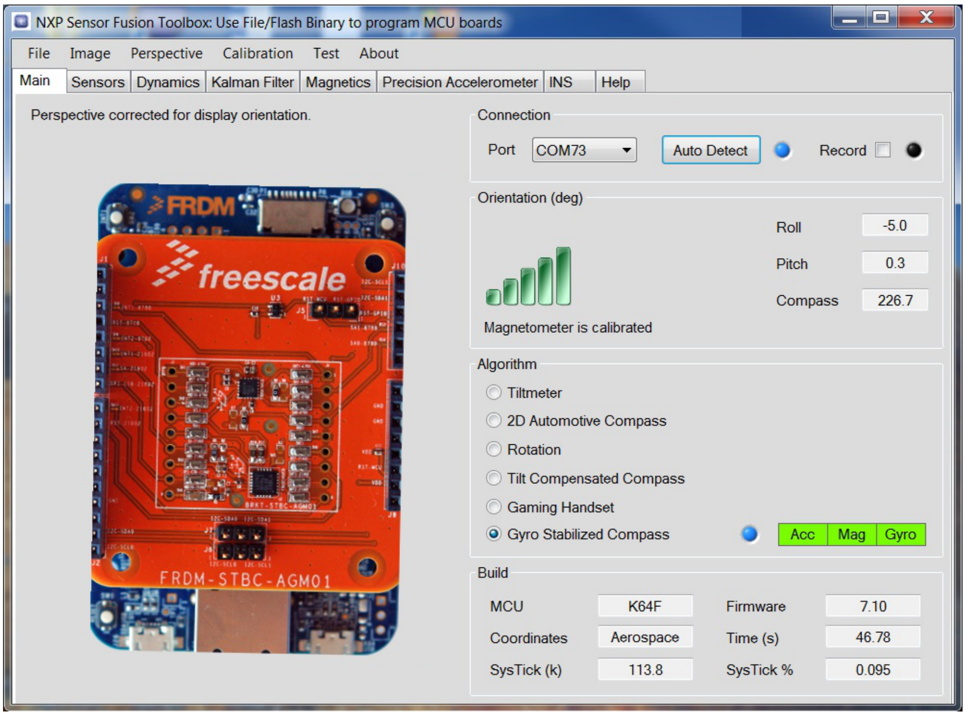
Outline:
- FreeRTOS.h
- task.h
- queue.h
- timers.h
- event_groups.h
- fsl_debug_console.h
- board.h
- pin_mux.h
- clock_config.h
- fsl_port.h

Description	Resource	Path	Location	Type
Warnings (1 item)				

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 begins 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:



NXP Sensor Fusion Toolbox: Use File/Flash Binary to program MCU boards

Perspective corrected for display orientation.

Connection

Port: COM73

Record □

Orientation (deg)

Roll	-5.0
Pitch	0.3
Compass	226.7

Magnetometer is calibrated

Algorithm

- ○ Tiltmeter
- ○ 2D Automotive Compass
- ○ Rotation
- ○ Tilt Compensated Compass
- ○ Gaming Handset
- ● Gyro Stabilized Compass

Build

MCU	K64F	Firmware	7.10
Coordinates	Aerospace	Time (s)	46.78
SysTick (k)	113.8	SysTick %	0.095

This completes the exercise to run the ISSDK 1.5 Sensor Fusion v7.10 project.

7 Revision history

Table 1. Revision history

Revision number	Date	Description
1.2	20170306	Updates for ISSDK v1.5
1.1	20161123	Updates for ISSDK v1.1
1.0	20160803	Initial public release

8 Legal information

8.1 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.

8.2 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.

8.3 Trademarks

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

NXP — is a trademark of NXP B.V.

the NXP logo — is a trademark of NXP B.V.

Freescale — is a trademark of NXP B.V.

the Freescale logo — is a trademark of NXP B.V.

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
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 6 March 2017