Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

NXP Sensor Fusion

NXP Sensor Fusion for Kinetis MCUs

Rev. 2.0 — 12 August 2016 **User guide**

Document information

Info	Content
Keywords	Sensor fusion, accelerometer, gyroscope, magnetometer, altimeter, pressure
Abstract	Provides full details on the structure and use of the NXP Sensor Fusion Library for Kinetis MCUs.

Revision history

Rev	Date	Description
2.0	20160812	This version of the user manual documents Version 7.00 of the NXP Sensor Fusion Library. The prior published version was Version 5.00 of the NXP Sensor Fusion Library. The kit has undergone major restructuring, and most of this document is new. This document was written by Michael Stanley and Mark Pedley.

Contact information

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

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

1. Introduction

Sensor fusion is a process by which data from several different sensors are *fused* to compute something more than could be determined by any one sensor alone. An example is computing the orientation of a device in three-dimensional space. That orientation is then used to alter the perspective presented by a 3D GUI or game.

The NXP Sensor Fusion Library for Kinetis MCUs (also referred to as *Fusion Library* or *development kit*) provides advanced functions for computation of device orientation, linear acceleration, gyro offset and magnetic interference based on the outputs of NXP inertial and magnetic sensors.

Version 7.00 of the development kit has the following features:

- Full source code for the sensor fusion libraries
- IDE-independent software based upon the NXP Kinetis Software Development Kit (KSDK).
- The Fusion Library no longer requires Processor Expert for component configuration.
- Supports both bare-metal and RTOS-based project development. Library code is now RTOS agnostic.
- Optional standby mode powers down power-hungry sensors when no motion is detected.
- 9-axis Kalman filters require significantly less MIPS to execute
- All options require significantly less memory than those in the Version 5.xx library.
- Full documentation including user manual and fusion data sheet

The fusion library is supplied under a liberal BSD open source license, which allows the user to employ this software with NXP MCUs and sensors, *or those of our competitors*. Support for issues relating to the default distribution running on NXP reference hardware is available via standard NXP support channels. Support for nonstandard platforms and applications is available at https://community.nxp.com/community/sensors/sensorfusion.

This document is part of the documentation for NXP Sensor Fusion Library for Kinetis MCUs software. Its use and distribution are controlled by the license agreement in the Software licensing section.

1.1 Software licensing

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of Freescale Semiconductors, Inc., nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

1.2 Software features

Version 7.00 of the Fusion Library is delivered as part of the Intelligent Sensing SDK (ISSDK), which itself is included within various Kinetis SDKs generated via the Kinetis Expert or KEX tool (http://kex.nxp.com). By *various*, we mean that ISSDK and the fusion library are being released in a phased manner within the KEX ecosystem. Initially, only the FRDM_K22F and FRDM_K64F Freedom boards will be supported. Others will be added as time goes on.

V7.00 was redesigned from the ground up for easy portability. It provides access to source code for all functions. Software features include:

- Fusion and magnetic calibration algorithms
- Programmable multirate sensor sampling
- Programmable sensor fusion rate
- Supported frames of reference include AEROSPACE, Android and Windows 8.
- Includes drivers for NXP motion sensors
- Minimum functions to learn.
    - Write sensor readings into one set of global structures using the addToFifo function.
    - Read fusion results from a different set of global structures
- Ability to compile and link any combination of standard algorithms
    - Accelerometer only (tilt)
    - Magnetometer only eCompass (vehicle)
    - Gyro only orientation (relative rotation)
    - Accelerometer plus magnetometer 6-axis eCompass
    - Accelerometer plus gyroscope orientation (gaming)
    - Accelerometer plus magnetometer and gyroscope (full 9-axis)
- NXP's award-winning magnetic calibration software, which provides geomagnetic field strength, hard and soft iron corrections and quality-of-fit indication
- Optional standby mode powers down power hungry sensors when no motion is detected

- Directly compatible with the NXP Sensor Fusion Toolbox for Android and Windows (Fusion Toolbox). Example programs include predefined Bluetooth/UART interfaces compatible with the Fusion Toolbox.

1.3 Supporting documentation

1.3.1 Included in the kit

Included in the docs directory, mentioned in Section 1.2:

- This user guide
- NXP Sensor Fusion Library for Kinetis Data Sheet
- NXP Application Note AN5016, Rev. 2.0: Trigonometry Approximations
- NXP Application Note AN5017, Rev 2.0: Aerospace, Android and Windows 8 Coordinate Systems
- NXP Application Note AN5018, Rev. 2.0: Basic Kalman Filter Theory
- NXP Application Note AN5019, Rev. 2.0: Magnetic Calibration Algorithms
- NXP Application Note AN5020, Rev. 2.0: Determining Matrix Eigenvalues and Eigenvectors by Jacobi Algorithm
- NXP Application Note AN5021, Rev. 2.0: Calculation of Orientation Matrices from Sensor Data
- NXP Application Note AN5022, Rev. 2.0: Quaternion Algebra and Rotations
- NXP Application Note AN5023, Rev. 2.0: Sensor Fusion Kalman Filters
- NXP Application Note AN5286, Rev. 2.0: Precision Accelerometer Calibrations

1.3.2 Found elsewhere

- www.nxp.com/sensorfusion
- MCU on Eclipse blog at https://mcuoneclipse.com/
- Kinetis Design Studio software at www.nxp.com/kds
- Euler Angles at en.wikipedia.org/wiki/Euler_Angles
- Introduction to Random Signals and Applied Kalman Filtering, 3rd edition, by Robert Grover Brown and Patrick Y.C. Hwang, John Wiley & Sons, 1997
- Quaternions and Rotation Sequences, Jack B. Kuipers, Princeton University Press, 1999
- NXP Freedom development platform home page at nxp.com/freedom
- OpenSDA User's Guide, NXP Semiconductors N.V., Rev 0.93, 2012-09-18
- NXP OpenSDA support page at nxp.com/opensda
- PE micro OpenSDA support page at www.pemicro.com/opensda
- Segger OpenSDA support page at https://www.segger.com/opensda.html

1.4 Requirements

1.4.1 MCU

Minimum requirements for the standard "all algorithms / no RTOS" build are:

- 40MHz ARM Cortex M0+ or higher MCU
- 60K bytes Flash NVM
- 12.5K bytes RAM
- 1 I²C Port
- 1 UART Port

MCUs that include floating point units will consume significantly fewer CPU cycles, although an FPU-less M0+ works fine.

In practice, the sensor fusion code has proven highly portable, and can be ported to almost any 32-bit MCU meeting the requirements above.

NXP markets a line of Arduino™ compatible Freedom Development Boards which make an ideal platform for sensor fusion development projects. Figure 1 below illustrates the FRDM-K64F, which easily meets all of the requirements above.

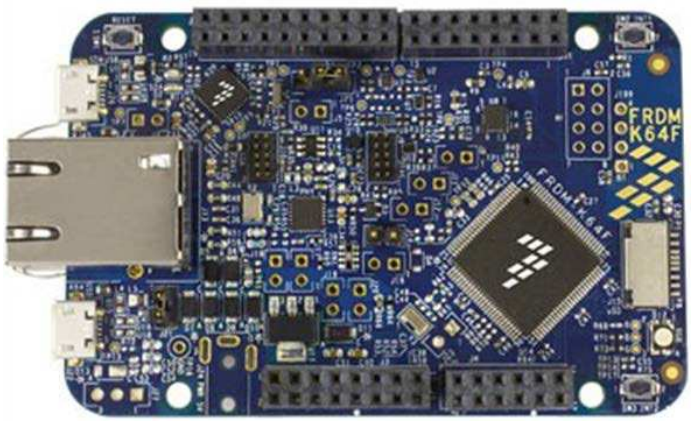


Fig 1. K64F Freedom Development Platform

NXP Freedom boards proven to be compatible on past releases of the toolkit include:

- FRDM_KL25Z
- FRDM_KL26Z
- FRDM_KL46Z
- FRDM_K20D50M
- **FRDM_K22F**
- **FRDM_K64F**
- FRDM_KV31F
- FRDM_KEAZ128

The above is not an all-inclusive list; it simply represents the list of boards that the sensor fusion team at NXP has targeted in the past. The initial release of Version 7.00 sensor fusion offers out-of-the-box support for only the boards shown in bold.

Please consult http://www.nxp.com/freedom, the relevant MCU datasheet, and the NXP Sensor Fusion for Kinetis MCUs Datasheet for additional detail.

1.4.2 Sensors

The development kit supports sensors with both I^2C and SPI interfaces. Table 1 specifies the sensor types required on a per algorithm basis. The descriptions shown in the left column match those used in the Sensor Fusion Toolbox for Windows GUI. The second column includes the enabling bit-field as defined in build.h in your project.

Table 1. Sensor types required as function of algorithm

Algorithm	Related IfDef	Pressure	Accelerometer	Magnetometer	Gyro
Altitude	F_1DOF_P_BASIC	X			
Tiltmeter	F_3DOF_G_BASIC		X		
2D Automotive Compass	F_3DOF_B_BASIC			X	
Rotation	F_3DOF_G_BASIC				X
Tilt Compensated Compass	F_6DOF_GB_BASIC		X	X	
Gaming Handset	F_6DOF_GY_KALMAN		X		X
Gyro Stabilized Compass	F_9DOF_GBY_KALMAN		X	X	X

Many NXP Freedom development boards include one or more sensors. These are summarized in Table 2.

Table 2. Sensors by Freedom Development Platform

Board	Sensors
FRDM-KL02Z	MMA8451 accel
FRDM-KL05Z	MMA8451 accel
FRDM-KE02Z	MMA8451 accel
FRDM-KE06Z	MMA8451 accel
FRDM-KL25Z	MMA8451 accel
FRDM-K20D50M	MMA8451 accel
FRDM-KL26Z	FXOS8700 accel + mag
FRDM-K64F	FXOS8700 accel + mag
FRDM-K22F	FXOS8700 accel + mag
KV31F	FXOS8700 accel + mag
FRDM-KL46Z	MMA8451 accel + MAG3110
FRDM-KEAZ128	No Sensors

In addition, NXP has released a number of Freedom-compatible sensor boards specifically designed to support various algorithms shown in Table 1. These include:

- FRDM-FXS-9-AXIS (deprecated), which contains an FXOS8700CQ 6-axis accelerometer/magnetometer combination sensor and an FXAS21000 gyroscope
- FRDM-FXS-MULTI (deprecated), which contains all features of the FRDM-FXS-9-AXIS plus additional sensors
- FRDM-FXS-MULTI-B (deprecated), which contains all features of the FRDM-FXS-MULTI plus Bluetooth Module and battery

- FRDM-STBC-AGM01, which replaces the FRDM-FXS-9-AXIS. It contains FXOS8700CQ 6-axis accelerometer/magnetometer combination sensor and an FXAS21002 gyroscope
- The FRDM-FXS-MULT2-B replaces the FRDM-FXS-MULTI and FRDM-FXS-MULTI-B. It uses the newer FXAS21002 gyroscope, but it otherwise has the same sensor content.

The first three boards were built from the same PCB design and differ only in the number of parts on the platform. Those first three boards are no longer in production, but the Fusion Library works with any of the above boards. The FRDM-FXS-MULTI-B & FRDM-FXS-MULT2-B boards are the only ones that supports Bluetooth communications to the NXP Sensor Fusion Toolbox for Android.

Key components of the FRDM-FXS-MULTI-B and FRDM-FXS-MULT2-B are identified in Fig 2. The two boards have essentially the same layout. The major difference between the two is the new and improved FXAS21002 on the FRDM-FXS-MULT2-B.

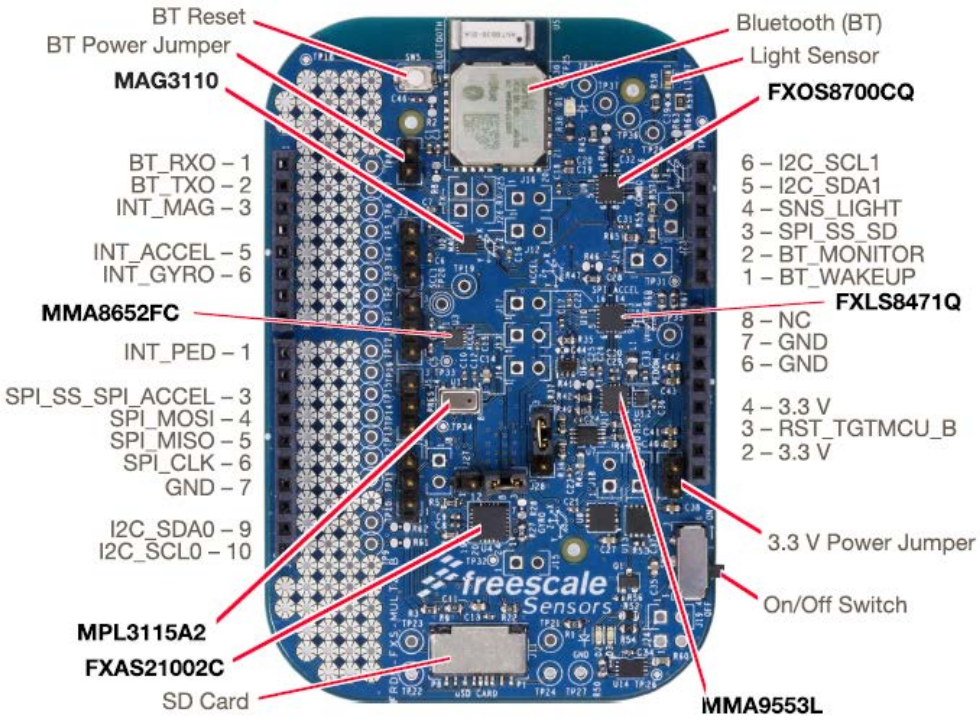


Fig 2. FRDM-Mult2-B with key components

Fig 3 shows the FRDM-FXS-MULTI-B Sensor Development Platform plugging into the FRDM-KL25Z Freedom Development Platform.

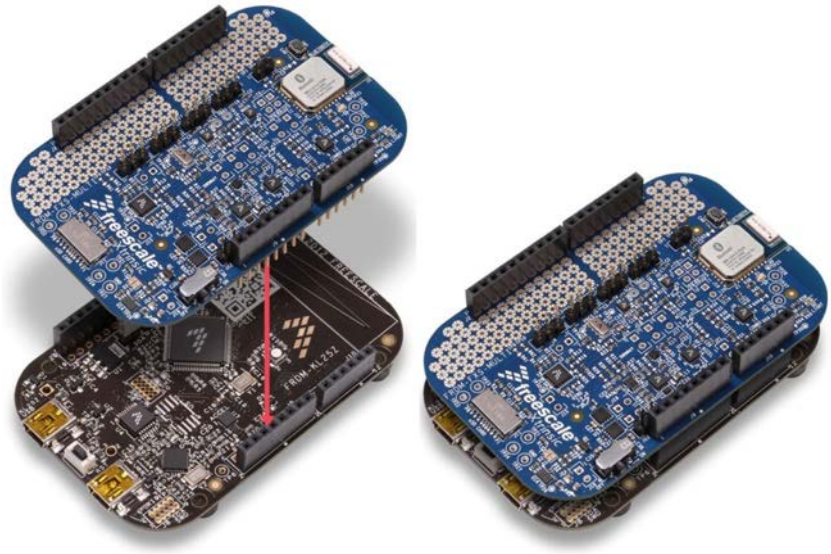


Fig 3. FRDM-FXS-MULTI-B mated to FRDM-KL25Z

Note: *When mating the two boards, check to ensure that the sensor board, when configured with a battery mounted on the back, is free of any obstacles presented by the base, NXP Freedom Development board. In particular, we have noted problems with FRDM-K22F boards. You will need to remove or relocate the battery.*

The FRDM-STBC-AGM01 is shown in Fig 4. It is a much simpler board than those shown above, but it still has all the components necessary to implement full 9-axis sensor fusion. It is compatible with all the base boards listed above. Note J6 and J7 jumpers near the bottom of the board. These let you select between two different sources of I^2C signals.

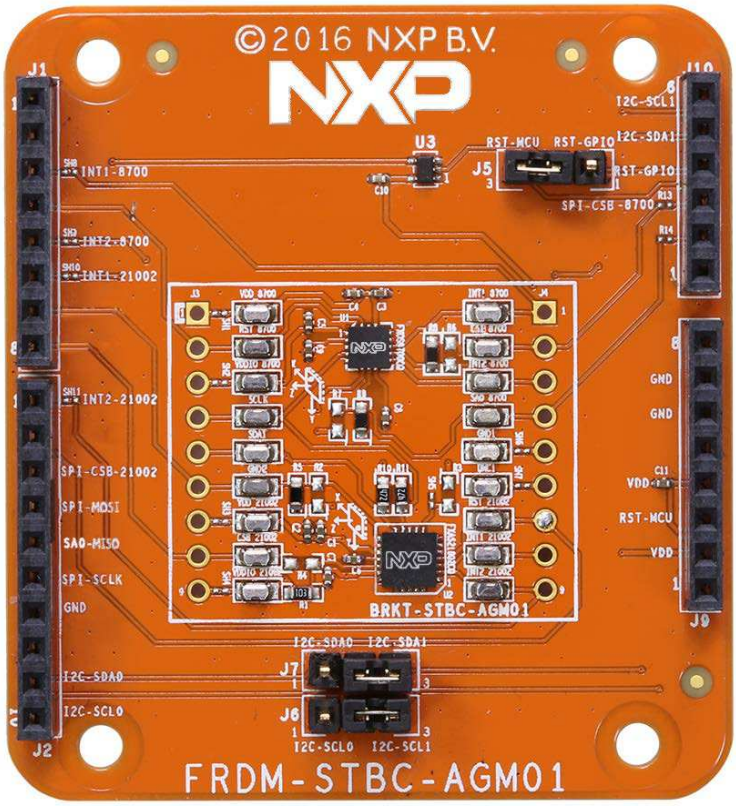


Fig 4. FRDM-STBC-AGM01 Freedom Development Platform

1.4.3 MCU Peripherals

Version 7.00 of the development kit represents a major departure from the design flow supported by Version 5.00 and earlier. Reference designs are no longer based on Processor Expert components, and instead rely on the Kinetis SDK. The software has been refactored to improve portability and support a more intuitive, object-oriented, interface. Accordingly, specific hardware requirements are associated with subsystems that will be explored in more detail in Section 3. A basic summary is:

Table 3. Primary subsystems and associated peripherals

Table description (optional)

Subsystem	Required	Peripherals Required	Comments
Main	Yes	Timer	PIT or similar if bare metal; SysTick commonly used for RTOS
Control	Debug minimum	UART	Example projects utilize two, one for OpenSDA wired connections, one for Bluetooth.
Status	No	Tri-Color LED	
ISSDK	Yes	I^2C and/or SPI	As required by the specific set of sensors used

Each project contains an include file called issdk_hal.h. This file pulls in header files for a Freedom board and sensor shield. When starting with a new board configuration, you should gather the schematics and user manuals for your board(s), then fill out a table similar to Table 4.

Table 4. FRDM-K64F and FRDM-FXS-MULT2-B / peripheral assignments

Subsystem	Function	Suggestion	Comments
Bare-metal main	Timer	PIT0	Encapsulated in driver_pit.c/.h
FreeRTOS main	Timer	SysTick	Possible interaction with Sensor Toolbox metrics measurement
Control	UART	UART3 on PTC17:16	Connects to Bluetooth module on sensor shields
		UART0 on PTB17:16	Connects to OpenSDA CDC[1] (UART/USB) on the Freedom board
Status	Red LED	PTB22	
	Green LED	PTE26	
ISSDK	I^2C	I^2C0 on PTC11:10	All I^2C sensors on FRDM-FXS-MULT2-B (excluding the FXLS8471Q)
	Accel/Mag	FXOS8700	I^2C address 0x1E on the FRDM-FXS-MULT2-B
	Gyro	FXAS21002	I^2C address 0x20 on the FRDM-FXS-MULT2-B
	Altimeter	MPL3115A2	I^2C address 0x60 on the FRDM-FXS-MULT2-B

1.4.4 IDE independence

As mentioned in the prior section, version 7.00 of the development kit represents a major departure from the design flow supported by Version 5.00 and earlier. Reference designs are no longer based on Processor Expert components, and instead rely on the Kinetis SDK. CodeWarrior support has been dropped. IAR support has been added, and any screen shots shown in this user manual will generally be taken from IAR projects.

The fusion library itself is written entirely in C, and should be portable to any IDE.

The Kinetis Design Studio IDE is available for no cost at: http://www.nxp.com/kds.

1.4.5 Enablement tools

The demo sensor fusion builds include a serial command interpreter which can communicate sensor and fusion status back to Windows or Android-based graphical user interfaces. Using those same interfaces, the user can select which fusion algorithm to monitor and experiment with other fusion options. There are two versions of the GUI, which are described in the following two sections.

Both versions of the toolbox are supported by the same serial packet protocol which is encapsulated within the **control subsystem** shown in Fig 26. This subsystem can be omitted in your final product build, but we highly recommend that it be retained throughout the product development/debug phases. *NXP support via the Sensor Fusion Community generally requires that the developer have access to the Sensor Fusion Toolbox for Windows.* Section 9 discusses many ways in which the visualization

1. CDC = Communications Device Class

capabilities of this tool can help you to easily diagnose many implementation common problems.

Both versions of the Sensor Fusion Toolbox are free.

1.4.5.1 Sensor Fusion Toolbox for Android

This is the original GUI first released to the world via Google Play in early 2013. The app is designed to communicate via standard Bluetooth with an NXP Freedom board equipped with a FRDM-FXS-MULT2-B sensor shield.

The application lets you visually explore sensor fusion tradeoffs. It also includes extensive sensor fusion tutorial information in the in-app documentation.

The latest version of this app may be downloaded at https://play.google.com/store/apps/details?id=com.sensors.fusion.

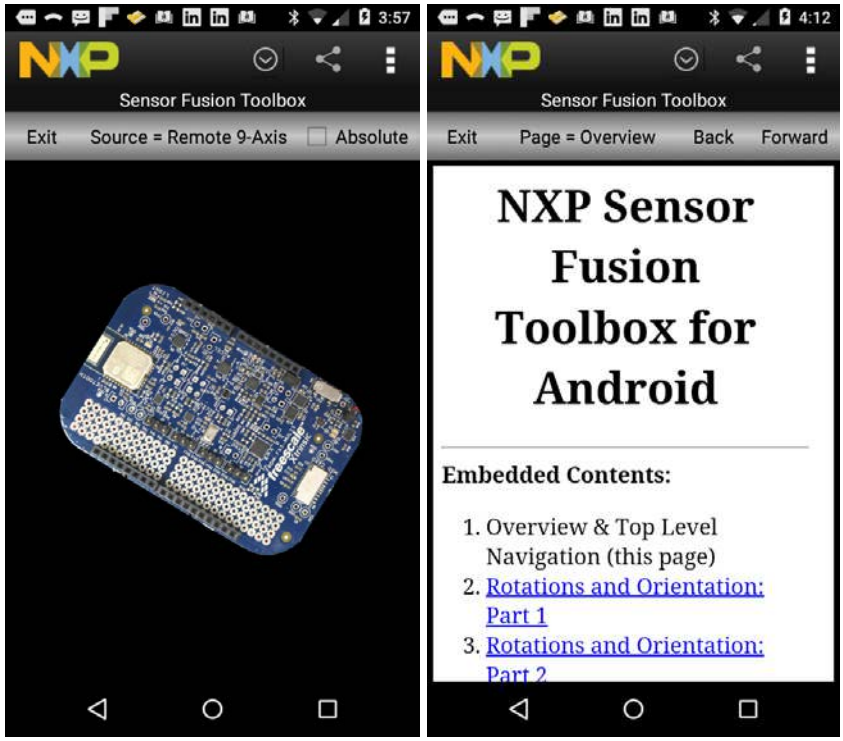


Fig 5. The Sensor Fusion Toolbox for Android

Select the Navigation icon and then **Documentation** to bring up the in-app help, shown on the right side of Fig 5.

1.4.5.2 Sensor Fusion Toolbox for Windows

The Android version of the toolbox described above can provide a lot of insights into the behavior of the various sensor fusion algorithms in the toolkit, but it pales in comparison to the advanced features contained in the Windows version of the tool (Fig 6).

In addition to the Device view shown above, the Sensor Fusion Toolbox includes:

- Sensor outputs versus time graphs
- Fusion outputs versus time graphs
- Magnetic calibration sample constellation and calibration outputs
- Altimeter/temperature sensor tab
- Precision accelerometer calibration functions (new for this release)
- INS tab to play with double integration to approximate position (new for this release)
- The ability to save and restore accelerometer, magnetometer and gyroscope calibration coefficients for quick startup.

If your Windows PC has a Bluetooth interface or USB dongle, you can communicate wirelessly via Bluetooth to your FRDM-FXS-MULT2-B equipped Freedom board. If your shield does not include Bluetooth support, you can connect via USB to your PC. The interface controls are identical in both cases.

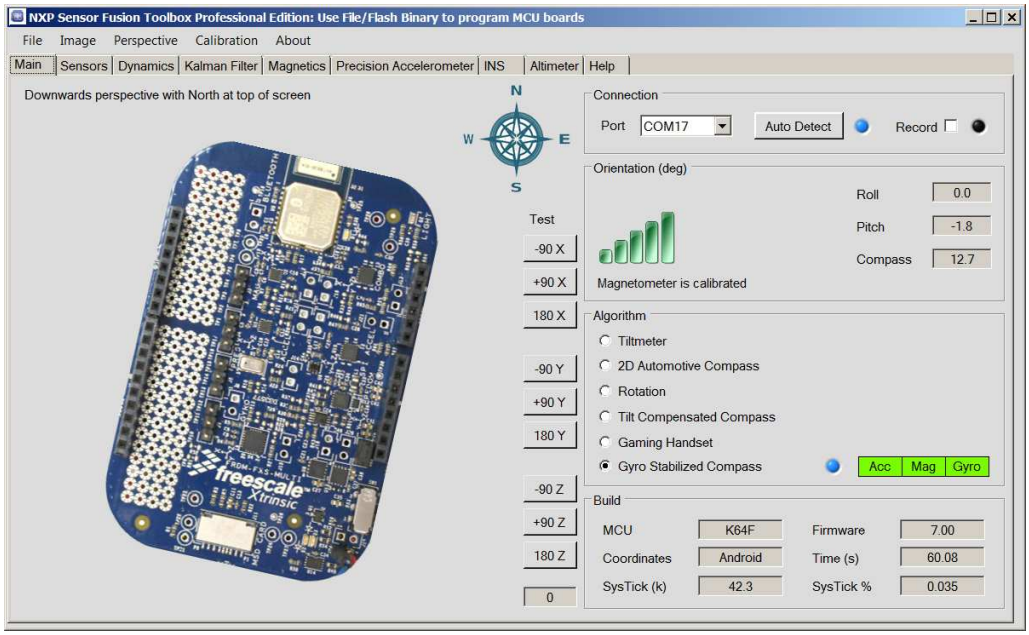


Fig 6. Sensor Fusion Toolbox for Windows

The Sensor Fusion Toolbox for Windows can be downloaded from http://www.nxp.com/sensorfusion.

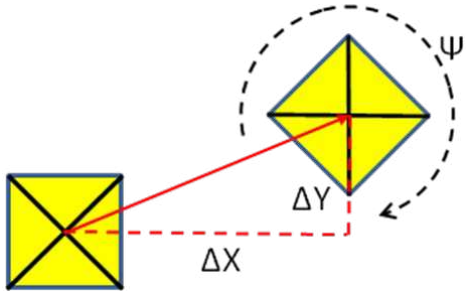
2. Sensor Fusion topics and options

If you are familiar with accelerometers, magnetometers, gyroscopes and frames of reference, you can skip Section 2.1. Likewise, if you are already an experienced user of prior versions of the Freescale/NXP sensor fusion library, you can skip all the way to Section 2.8, which addresses a topic new to Version 7.00.

2.1 Vocabulary

If you consult any text on the dynamics of rigid bodies, you will quickly learn that any movement of any rigid body from point A to point B can be characterized as a translation plus a rotation.

Any movement from point A to point B can be decomposed into a translation plus optional rotation



In 3 dimensions, we need 6 DOF: ΔX, ΔY, ΔZ, ϕ, θ, ψ

Fig 7. Degrees of freedom explained

It takes six numbers to characterize that movement: change in X, Y and Z and rotations about X-, Y- and Z-axes. Notice that we are talking about the minimum set of numbers required to unambiguously specify a given movement. We are NOT talking about the number of sensors required to measure that movement.

So now, let's talk about sensors. A basic 3-axis accelerometer returns values for linear acceleration in each of three orthogonal directions.

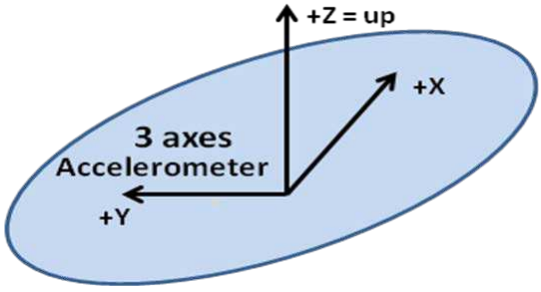


Fig 8. 3-Axis accelerometer

When you look at the figure, you can immediately see where the terms axis/axes come from. They refer to the sensor coordinate system axes.

There is an important thing you should consider about accelerometers at rest. When one of the axes associated with the sensor frame of reference is parallel to the gravity vector, as it is in the figure above, you will get no additional information from the other two acceleration numbers. They will both be zero, and you will be unable to tell if the accelerometer is rotated about the axis parallel to gravity.

The next device in our toolbox is the gyro, which returns rates of rotation about each of the three sensor axes. Notice that we are talking about sensor axes here. **As the sensor rotates, so does its frame of reference for the next measurement.**

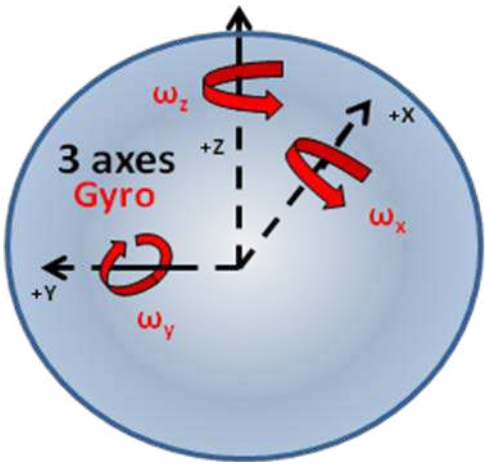


Fig 9. 3-Axis gyroscope returns the rotation rates about each of X-, Y- and Z-axes

A 3-axis magnetometer will return the X, Y and Z components of the ambient magnetic field. This is nominally the earth field for many applications, but readings may include significant offsets and distortions due to hard/soft iron effects. **The magnetometer is also subject to the same issue as an accelerometer – if one of the sensor axes is parallel to the ambient magnetic field vector, then the other two sensor axes will return values of zero.** The good news is that since the earth magnetic field and gravity are never colinear[2], between our accelerometer and magnetometer, we have enough information to figure out the current device orientation, regardless of how we rotate the sensor.

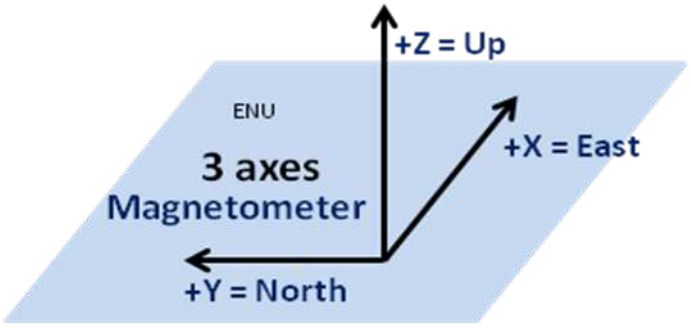


Fig 10. 3-Axis magnetometer will allow you to align yourself with the earth's magnetic field

Our first three sensors each returned a three-dimensional vector. But the pressure sensor returns just a single scalar value. Changes in pressure can be used to infer changes in altitude, which adds another source of information when computing vertical locations.

2. Except at the geomagnetic poles

Fig 11. Pressure[3] can be our 10th axis.

Combine an accelerometer with a gyro and you get a 6-axis Inertial Measurement Unit, or IMU.

Add a magnetometer to an IMU and you have a MARG (Magnetic, Angular Rate and Gravity) sensor. Add a compute engine to a MARG and you get an AHRS (Attitude and Heading Reference System).

A full 10-axis sensor subsystem = accelerometer + gyro + magnetometer + pressure

Use “DOF” when describing motion. Use “axis” or “axes” when describing sensor configurations.

2.2 3-Axis tilt

The 3-axis tilt algorithm is enabled with the **F_3DOF_G_BASIC** parameter in build.h. This algorithm requires only a single 3-axis accelerometer. It is only capable of modeling roll, pitch and tilt from vertical. It has no sense of compass orientation. This implies that the four orientations shown in Fig 12 will return exactly the same tilt value.

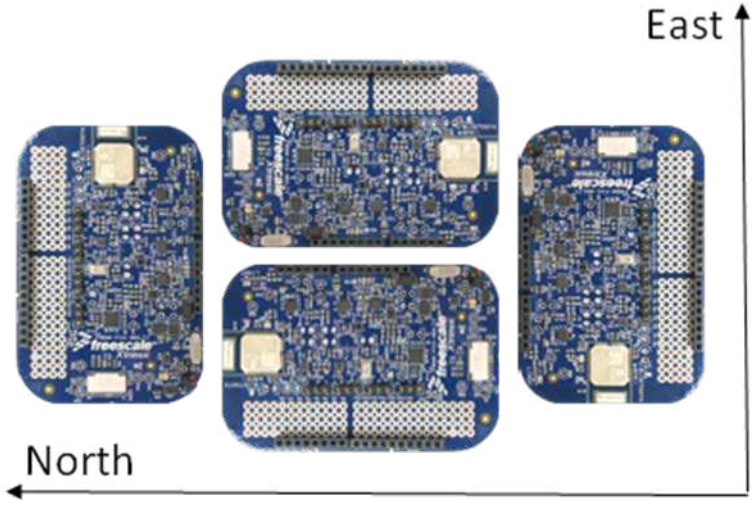


Fig 12. Equivalent tilt cases

This can result in some interesting behaviors when you invert the sensor board in the Sensor Fusion Toolbox. You may observe a “twist” by 180 degrees in the apparent orientation of the board. Both before and after are correct because of the ambiguity in the orientation representation.

3. The NXP MPL3115A2 can provide pressure or an estimate of altitude. It can also provide a temperature measurement.

2.3 2D automotive eCompass

This algorithm utilizes a 3-axis magnetometer to compute compass heading, while forcing the apparent orientation of the board to always be flat with respect to the earth.

For general use, the 6-axis tilt compensated eCompass discussed later is considered superior. But that algorithm is subject to errors due to linear acceleration. In contrast, the 2D automotive eCompass has no accelerometer, and therefore is immune to that issue. But it IS still sensitive to magnetic interference, and tilt will introduce errors in the apparent heading.

Tilt errors result from the fact that over most of the Earth's surface, the magnetic field does not point to magnetic north, it points north and down. So tilting the sensor can rotate some of the up-down component of the field into the X-Y components. This then introduces errors in the heading computation.

This is easy to see using the Sensor Fusion Toolbox. Simply select the 2D algorithm option and tilt the board from horizontal.

2.4 3-Axis rotation

This option, which utilizes only a 3-axis gyroscope, is excellent for modeling rotations in three dimensions. But it has zero sense of up and down or compass heading. Orientation can be computed by integrating rotational rates over time, but if you do not know the initial orientation, you have no baseline for determining absolute orientation at any point in time.

2.5 6-Axis tilt-compensated eCompass

In Section 2.3, we noted that the 2D compass was subject to tilt error. When we utilize both an accelerometer and a magnetometer, we can mathematically align the sensor axes to the global earth frame of reference. This gives a full orientation estimate as well as accurate compass heading.

The limitations of this particular algorithm is that it is sensitive to magnetic interference as well as linear acceleration. Again, this is easy to see using the Sensor Fusion Toolbox. Wave a magnet a few inches from your board, or simply shake the board vigorously, and you will see the effects.

2.6 6-Axis gaming

This algorithm requires a 3-axis accelerometer and a 3-axis gyroscope. The gyro gives us the beautiful rotations we saw in the 3-axis rotation algorithm, and the accelerometer gives us an orientation with respect to gravity. What we do not get is compass heading. And this algorithm is still sensitive (although much less than the eCompass algorithm) to linear acceleration.

2.7 9-Axis

The 9-axis algorithm is generally considered to be the gold standard, as it does a good job of trading off the strengths and weaknesses of accelerometer, gyroscope and magnetometer to compute orientation with respect to the global earth frame. It also includes logic to identify magnetic and acceleration jamming, and ignore the affected sensor. The "Main" tab of the Sensor Fusion Toolbox for Windows includes a feature that identifies when magnetic jamming is detected.

You can still see some effects due to magnetic interference or linear acceleration, but they are largely attenuated. However, **they cannot be fully attenuated when those sources of interference are nearing steady state**. The algorithm takes advantage that we know how each of the 3 vectors in question (gravity, earth field and rotation) should change relative to the others when the board rotates. Introduction of constant offsets into assumed gravity and earth field vectors will throw off those calculations.

2.8 Inertial navigation – truth or fiction?

Accelerometers measure linear acceleration minus gravity. If you know the orientation of the accelerometer, you can subtract out the gravity component, leaving you with only linear acceleration. The linear acceleration vector can then be rotated from sensor to global frame of reference. Integrate that once and you get velocity, twice you get position. We all learned that in college. This is the basis for inertial navigation, AKA dead reckoning. These techniques have been in use since World War II.

BUT… those systems could afford to spend lots of money carefully calibrating sensors to remove offset and gain errors in the sensors. Today's MEMS and solid state sensors are mass manufactured with low cost in mind.

Let us consider a modern commercial MEMS accelerometer, with a specified post board mount accuracy of ±20 m*g* and an output noise level of 99 $\mu g/\sqrt{Hz}$. Conservatively modeling that 20 m*g* solely in the X direction and with random noise distributed across all three axes, a simple Matlab simulation shows:

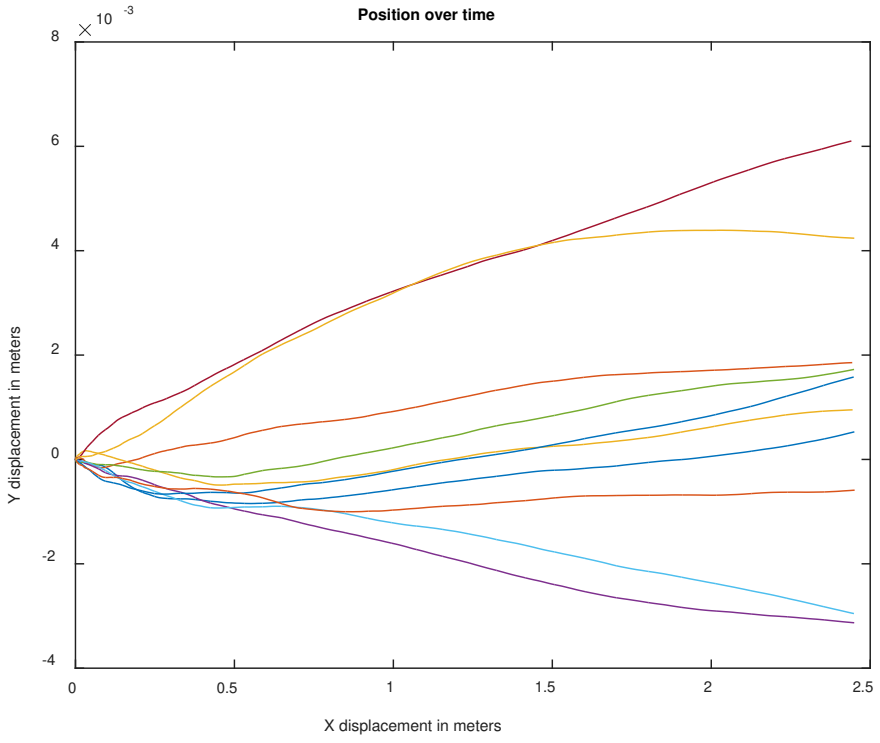


Fig 13. XY displacement over a 5 second interval (10 runs) for an uncalibrated accelerometer

The large displacement in X is due to the 20 m*g* offset in X. That 20 m*g* offset yields a velocity error which increases linearly with time, and a position error that grows quadratically.

The smaller displacement in Y is strictly due to wideband noise. Velocity errors in accelerometer noise will grow as a function of the square root of time. Position errors will grow as a function of time$^{2/3}$.

Doubly integrating even small input errors produces very large errors in time. Not understanding this basic math is one of the most common errors we see engineers new to the field making. If you would like to dig deeper, reference [9] provides a solid basis for understanding these relationships.

But what if we could calibrate out that 20 m*g* post-board mount offset? Version 7.00 of the sensor fusion library includes routines which allow you to do a one-time/one-temperature calibration which can reduce that 20 m*g* offset down to much lower levels. The same simulation used above, but with a value of 350 μg residual nonlinearity, yields the graph below.

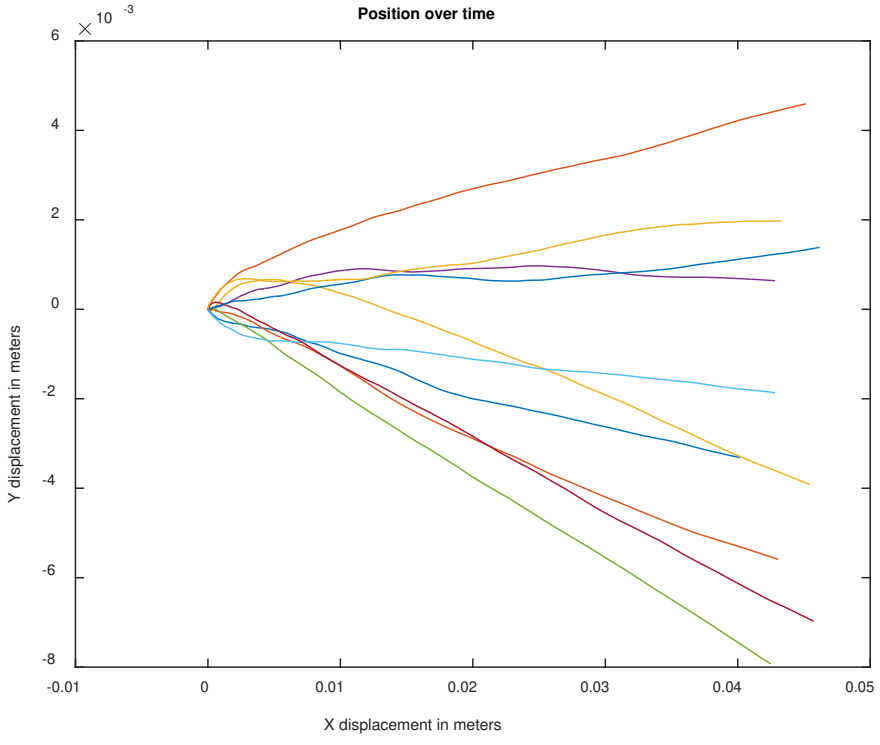


Fig 14. XY displacement over a 5 second interval (10 runs) for a calibrated accelerometer

Notice the dramatic improvement in the X-displacement, from a maximum of 2.5 meters to 5 cm. Bear in mind that these errors will continue to grow over time. But the point is that using inertial data to *aide* other systems (like GPS) is now possible so long as you do a reasonable job of precalibrating out known sensor errors.

The new “INS” tab in the Sensor Fusion Toolbox for Windows lets you experiment with errors in position over time. You will see very quick explosions in error for uncalibrated sensors. But as the discussion above shows, you will get better results after calibrating your sensors.

3. Quick Start Guides

3.1 Getting hardware

Hardware requirements were outlined in Sections 1.4.1, 1.4.2 and 1.4.3. No matter what MCU and sensor set you select for your final application, we recommend that you start with, and keep available, a known working solution based on the boards in those sections. When it comes time to debug your production design, you will find it useful to have that working example of sensor fusion as a reference.

The KSDK includes both bare-metal and FreeRTOS sensor fusion projects for each ISSDK-supported MCU. Start with the sensor shield required by one of those projects. Any V7.00 sensor fusion community pre-releases will include an example utilizing FRDM-K64F and FRDM-FXS-MULT2-B boards

1. Development boards can be purchased at http://www.nxp.com/freedom.
2. Visit http://www.nxp.com/opensda and install the appropriate OpenSDA drivers for your board. If desired, update the bootloader firmware on your board using the procedures outlined.

   Note: *Programming boards via the Sensor Fusion Toolbox require that the OpenSDA implementation on your embedded board supports drag and drop programming.*

3.2 Getting the KSDK

Version 2.xx of the KSDK is custom generated for each specific Kinetis MCU. The KSDK contains only code and examples compatible with that MCU and the RTOS and tool options you select. You can configure and download your version at http://kex.nxp.com.

Within a KEX-created KSDK, you can expect to see a directory tree similar[4] to:

- SDK_2.x_FRDM-K64F
  - boards
    - frdmk64f
      - demo_apps
      - driver_examples
      - …
    - frdmk64f_agm01
      - frdm_k64f.c
      - frdm_k64f.h
      - frdm_stbc_agm01_shield.h
      - issdk_examples
        - algorithms
          - sensorfusion
        - sensors
          - fxas21002
          - fxos8700
    - frdmk64f_mult2b
      - frdm_k64f.c
      - frdm_k64f.h
      - frdm_stbc_mult2b_shield.h

4. This document was written prior to the final KSDK inclusion of ISSDK and the Sensor Fusion V7.00 library. Some changes may have occurred in the interim.

- issdk_examples
  - algorithms
    - sensorfusion
  - sensors
    - fxas21002
    - fxos8700
    - fxls8471q
    - mag3110
    - mma865x
    - mpl3115
- CMSIS
- devices
  - MK64F12
    - drivers
    - iar
- docs
- middleware
  - issdk_1.0
    - algorithms
      - sensorfusion
        - docs
        - sources
    - sensors
      - fxas21002.h
      - fxos8700.h
      - fxls8471q.h
      - mag3110.h
      - mma865x.h
      - mpl3115.h
- rtos
  - freertos_8.2.3

KSDK elements highlighted in red are utilized by the sensor fusion examples.

See Section 1.3.1 for supporting documentation included in the sensorfusion/**docs** directory. The **issdk_examples/algorithms/sensorfusion** directory will include example projects for both bare-metal and FreeRTOS-based applications. Eventually you can expect to see example projects for KDS, Keil, IAR and other IDEs. Prerelease versions of the Fusion Library will only include IAR examples.

3.3 Compiling binaries

The development kit is compatible with any number of different development environments. Additional information on software and tool options for Kinetis MCUs can be found at http://www.nxp.com/products/software-and-tools/run-time-software/kinetis-software-and-tools:KINETIS_SWTOOLS.

The Kinetis Design Studio IDE can be downloaded for free from http://kex.nxp.com.

Sample projects are completely preconfigured. You should only have to:

1. Open the supply project(s) in the IDE of your choice
2. Compile
3. Download into your board

3.4 Microsoft Windows

You do not need to do any program development to start experimenting with sensor fusion algorithms on your board. The Sensor Fusion Toolbox for Windows comes with a variety of prebuilt binaries that can be installed on your development board.

The Sensor Fusion Toolbox for Windows can be downloaded from http://www.nxp.com/sensorfusion. The tool is free and is compatible with all the fusion features discussed in this user guide.

Assuming you already have a Freedom board and sensor shield in hand:

1. Connect Freedom board to PC using a USB cable
2. Start the Sensor Fusion Toolbox

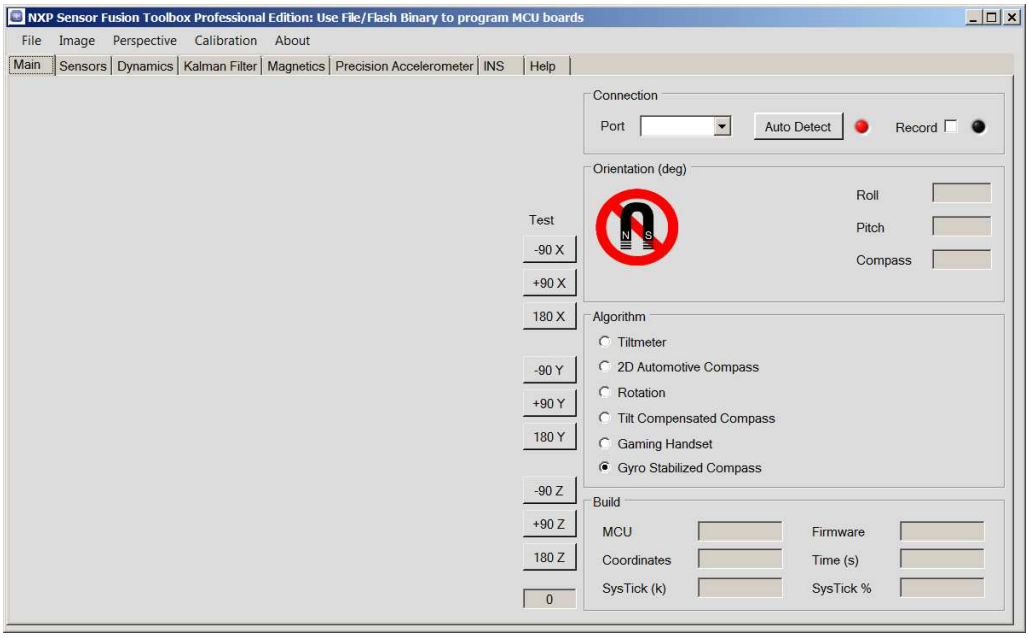


Fig 15. Sensor Fusion Toolbox For Windows at Startup

3. Program your board:
    a. Click File->Flash Kinetis Binary-><freedom board name>-><shield name>-><sensor_combo>.
    b. Check your USB connection as per the resulting dialog box and click OK after reading the full set of directions
    c. Select your board (MBED if using the ARM CMSIS-DAP bootloader) in the “Save As” dialog and click “Save”
    d. Click OK when informed that the board flashed.
    e. unplug and plug the board back in
4. Click the **Auto Detect** button. If your board is recognized, you will see the following dialog.

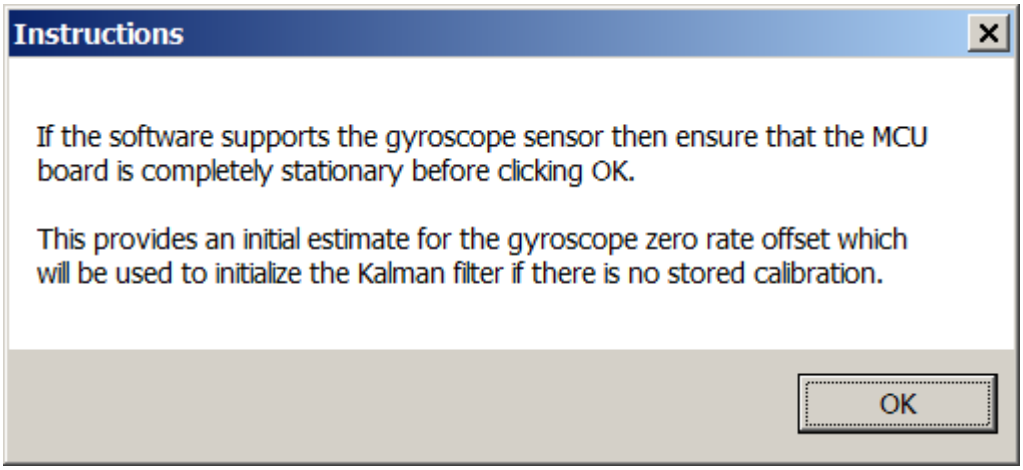


Fig 16. Get ready to calibrate your gyroscope

Put your board flat on a tabletop before hitting OK. This will allow the board discovery process to also measure stationary gyro offsets. These are then used to initialize Kalman filter gyro offset values. Next you should see

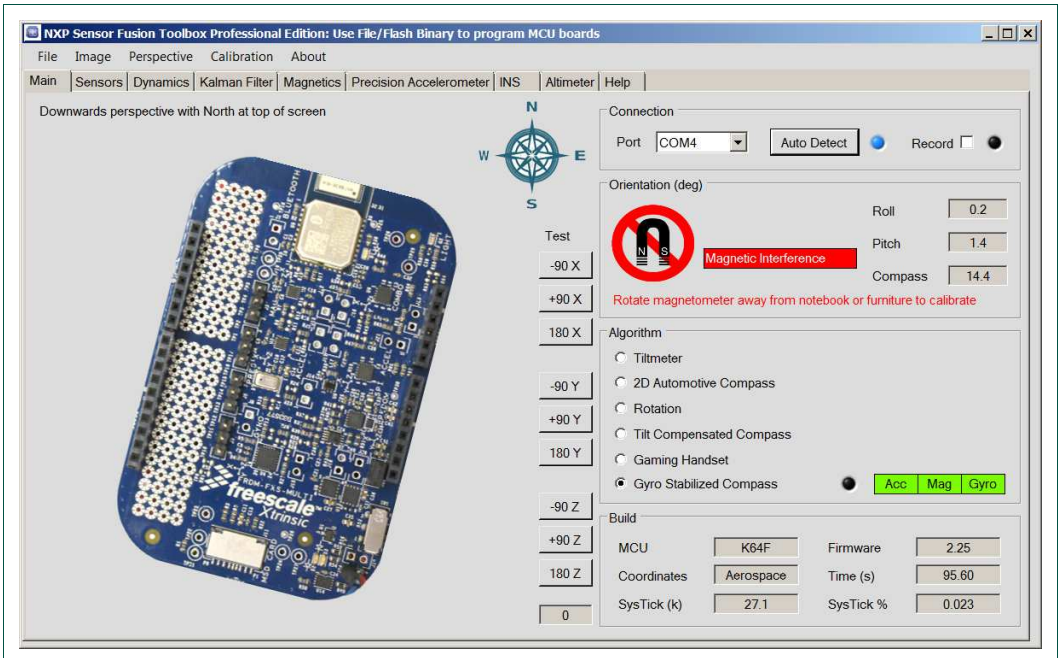


Fig 17. Sensor Fusion Toolbox For Windows at startup

5. Pick up your board and rotate it in space, away from any objects that contain ferrous materials, such as furniture and computers, until the Magnetic Interference notice disappears.

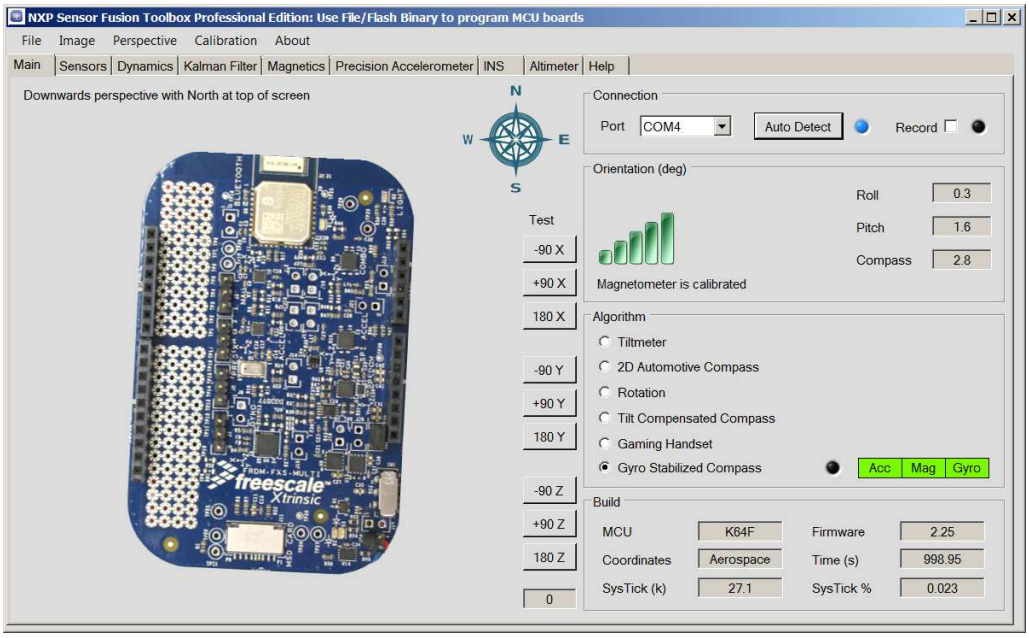


Fig 18. Sensor Fusion Toolbox For Windows after Magnetic Calibration

At this point, you have a working board that is properly communicating to the Sensor Fusion Toolbox. You can now start experimenting with the features listed in Section 1.4.5.2.

3.5 Android

The Sensor Fusion Toolbox for Android communicates via a development board via Bluetooth. Supported shield boards are the FRDM-FXS-MULTI-B and FRDM-FXS-MULT2-B boards. The former is no longer in production, having been replaced by the latter. The two boards look almost identical.

1. Program your development board using the standard sensor fusion demo (see Section 3.3).
2. The Sensor Fusion Toolbox for Android can be directly downloaded from Google's Play Store at https://play.google.com/store/apps/details?id=com.sensors.fusion. Alternately you can search for [NXP sensor fusion] in Google Play.

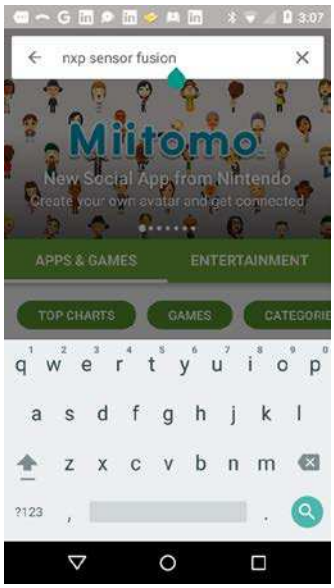


Fig 19. Searching for the App in Google's Play Store

If you search instead of using the direct link above, you may see two versions listed. The top version (with the diamond chip logo) is the legacy Freescale version of the tool. It will not be maintained going forward.

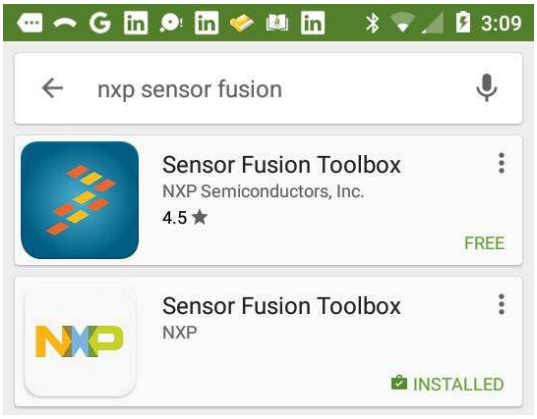


Fig 20. Sensor Fusion Toolbox For Android options

3. You should select the icon with the white and pastel NXP logo. It has all the features of the original, and will be the Android platform going forward.
4. Install the application in the normal manner.
5. Power up your sensor board. Ensure that the power jumper next to the Bluetooth module is installed ("A" in Fig 21).
6. FRDM-FXS-MULTI-B and FRDM-FXS-MULT2-B boards utilize Bluetooth modules from BlueRadios. Note the last six digits on the second line of the radio module ("B" in Fig 21).