BSX sensor fusion for smart sensor systems

Application Note

BSX sensor fusion for smart sensor systems

Document revision	1.0
Document release date	November 2023
Document number	BST-BHI260_BHI360-AN002-01
Sales Part Number	BHI360 : 0 273 017 037 BHI380 : 0 273 017 039 BHI260AB : 0 273 141 367 BHI260AP : 0 273 141 368 BHA260AB : 0 273 141 392
Notes	Data and descriptions in this document are subject to change without notice. Product photos and pictures are for illustration purposes only and may differ from the real product appearance.

Table of Contents

List of figures

List of tables

1 Introduction

This guide provides an overview of sensor fusion technology, specifically focusing on the BSX fusion software integrated into Bosch Sensortec smart sensors. It is designed to help you get started with using BSX fusion software, whether you are working with BHI260AB/BHI260AP (referred to as BHy2xx) or BHI360/BHI380 (referred to as BHI3xx) for your project. The guide is divided into several sections, each addressing different aspects of the software.

The first section of this document introduces the Bosch Sensortec fusion technology, including its functionality, operational principles, and how it is integrated into the smart sensor hardware.

In section 2, we present an overview of different sensor input signals and explain their potential for enhanced accuracy, usefulness, and reliability. In section 3, we summarize and explain the different sensor fusion outputs, referred to as virtual sensors.

A guidance of sensor calibration procedures is given in the next section 4 to ensure optimal performance when using sensor fusion.

After that, the integration concept of sensor fusion software inside BHI360 is shown. We'll also introduce you to the fundamental usage of "bhy2cli," an integral component of the COINES (COmmunication with INertial and Environmental Sensors) software package. COINES is a software package that provides a low-level interface to Bosch Sensortec's application board 3.0 and can be downloaded directly from Bosch Sensortec Website. With the help of bhy2cli, you gain the ability to easily interact with the virtual sensors and achieve different applications.

Lastly, we will explain the Key Performance Indicators (KPIs) used to assess the accuracy and overall functionality of the BSX Sensor Fusion Modules.

By the end of this guide, you should have gained a solid understanding of the Bosch Sensortec sensor fusion software and its effective application in your smart sensor projects.

2 Sensor fusion software (BSX)

2.1 Overview

Sensor fusion is software designed to integrate data from multiple sensors to provide a more accurate and reliable perspective of the environment or enhancing the tracking of the user's movements. Commonly utilized in a variety of applications such as mobile devices, wearables, robotics, and autonomous vehicles, it serves to augment the functionality in these domains.

The goal of sensor fusion software is to improve the accuracy, reliability, and efficiency of the sensor output by combining data from multiple sensors. By integrating data from different sensor sources, the software can mitigate the limitations of individual sensors, such as noise, bias, and drift, thereby offering a more reliable information.

Sensor fusion software typically includes a range of algorithms for data processing: calibration, filtering, and fusion. Calibration algorithms are used to correct for errors in the sensor data, such as bias and drift, and ensure that the data has the required quality. Filtering algorithms aim to remove unwanted noise and other artifacts from the sensor data like disturbances, further smoothing the data to diminish jitter and improve stability. Fusion algorithms are utilized to synthesize data from multiple sensors into a more cohesive and insightful format, such as determining absolute orientation in space, thereby offering a more comprehensive view of the surrounding or of our own movements.

The Bosch Sensortec software, BSX, offers a robust sensor fusion solution. This software includes algorithms for sensor data processing, calibration, and fusion, as well as APIs and tools for developers to integrate it into their applications. The available fusion algorithms can effectively combine sensor signals, including acceleration, angular rate, and magnetic field strength. These algorithms process the sensor data and produce more useful information such as orientation, linear acceleration, tilt-compensated heading, and gravity. A detailed description of all the available outputs, also known as virtual sensors, is provided later in this document.

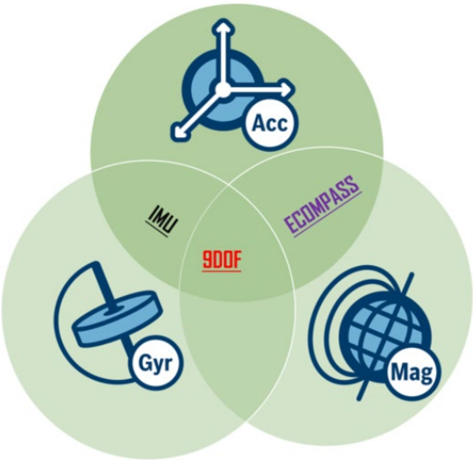


Figure 1: Different sensor fusion algorithms

Figure 1 illustrates three different kinds of sensor fusion algorithms, categorized based on the input sensor data utilized for generating the output readings. These are:

- IMU orientation estimation
- 9DoF orientation estimation
- Tilt compensated heading estimation (eCompass)

IMU Orientation estimation

This algorithm estimates the orientation of a device in three-dimensional space by integrating data from an accelerometer, a gyroscope. It employs sensor fusion techniques to minimize errors and provide accurate data on the relative orientation of the device. Relative orientation pertains to specifying the device's orientation in three-dimensional space. In this specification, both pitch and roll are defined in absolute terms with respect to the ENU coordinate system. However, the yaw component is solely defined relative to the device's initial orientation and does not incorporate a reference to true north, as it does not rely on Earth's magnetic field data. This data can be used in applications such as gaming, virtual reality, and navigation.

9DoF Orientation estimation

Building upon the IMU algorithm, this approach integrates data from a magnetometer alongside the accelerometer and gyroscope data to provide a comprehensive orientation reading. In this case, the algorithm provides absolute orientation data. "Absolute orientation specifies the device's orientation in space, encompassing all three components: yaw, pitch, and roll, in relation to the ENU coordinate system with absolute precision. This precision is achievable due to the availability of both Earth's magnetic field data and gravity data.

Tilt compensated heading estimation (eCompass)

This algorithm leverages data from an accelerometer and a magnetometer to compensate for the tilt of a device, enabling it to provide accurate readings of magnetic fields. The algorithm is used in applications such as electronic compasses and navigation systems where accurate representation of magnetic fields is crucial.

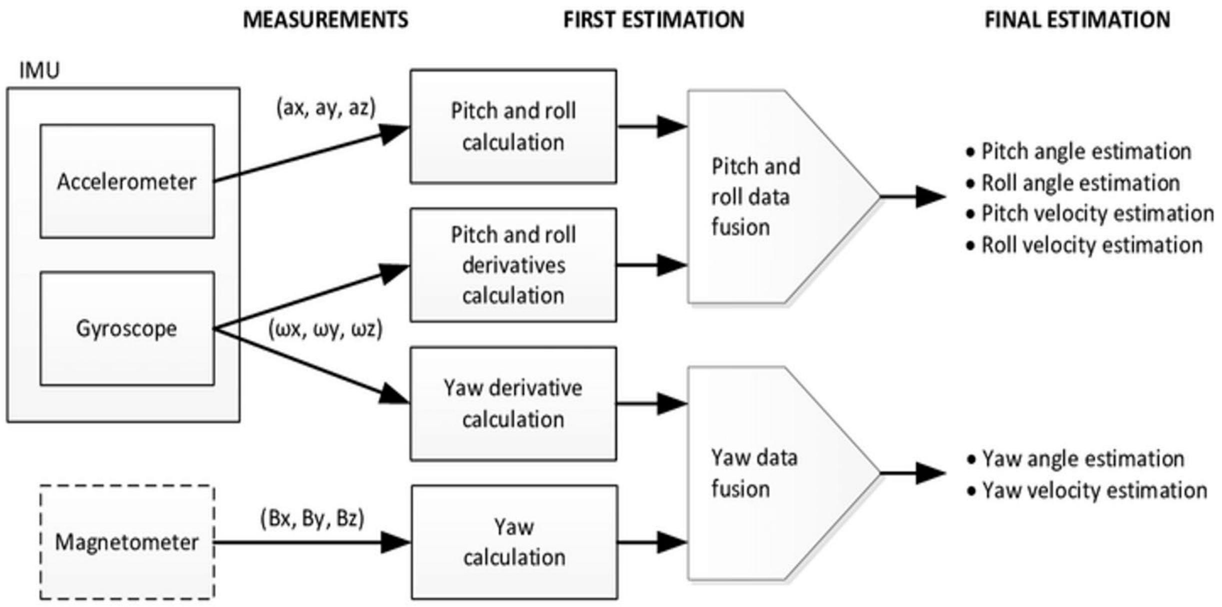


Figure 2: Functional block diagram of BSX orientation estimation

Figure 2 illustrates a simplified representation of the process where distinct input sensor data are handled by various functional blocks within the BSX sensor fusion software. It shows how this process helps in calculating the different output orientation parameters such as Pitch, Roll, and Yaw.

These are just a few examples of the sensor fusion algorithms that BSX offers. The software suite encompasses additional algorithms for tasks such as sensor calibration routines specifically for the accelerometer, gyroscope, and magnetometer, and features like magnetic distortion detection. This functionality ensures the provision of reliable orientation data even in environments characterized by unstable magnetic fields.

2.2 Reference coordinate system and sensor alignment

The reference coordinate system used by BSX to represent the orientation in 3D space is the ENU coordinate system. ENU stands for "East, North, Up" and refers to a coordinate system that has its origin at a specific location on the Earth's surface, with the X-axis pointing towards the East, the Y-axis pointing towards the North, and the Z-axis pointing upwards towards the sky. BSX rotation vectors describe a rotation from the reference coordinate system into the device co-ordinate system.

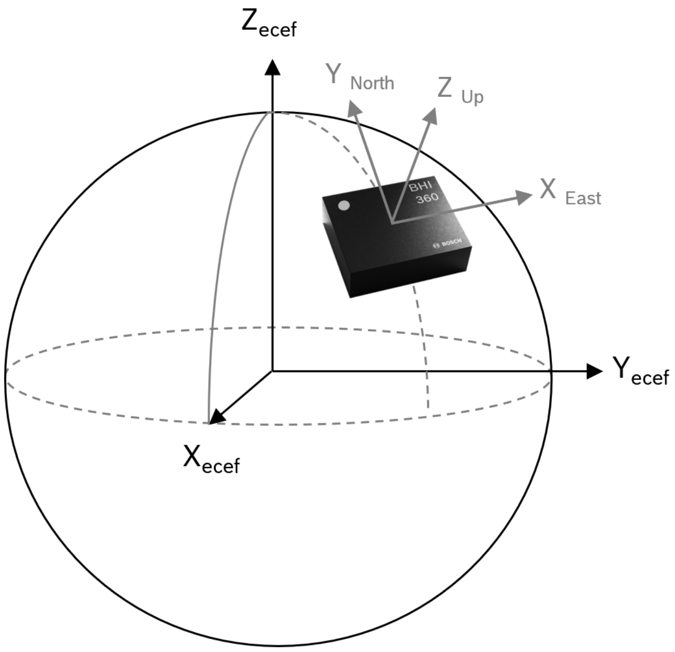


Figure 3: ENU coordinate frame convention used by BSX

The rotations could either be represented using a single quaternion or Euler sequences. For more details, please refer to sections 3.14, 3.15 and 3.17 in this document.

Axis remapping can be performed through the available sensor API. This is quite useful if the sensor coordinate system does not match with the coordinate system of the target device.

2.3 Software data input

BSX is capable of efficiently processing the sensor signals generated by the inertial sensors inside the smart sensor system. Specifically, the inertial measurement unit (IMU) data is transmitted to the sensor fusion software as acceleration and angular rate along the three axes (x, y, z).

Furthermore, BSX can incorporate magnetic field strength data from an external 3-axis magnetometer when connected to the smart sensor. The use of a magnetometer is required for some virtual sensors such as absolute orientation (rotation vector) and tilt compensated heading (eCompass).

Depending on the selected virtual sensor, BSX requires different number and types of physical sensors as inputs. For instance, the virtual sensor “Raw Accelerometer” requires one physical sensor: the accelerometer, while the virtual sensor “Game Rotation Vector” would require two sensors: accelerometer and gyroscope.

The smart sensor allows users to control the input data through a wide set of configuration options. These options range from selecting filtering modes to setting operation range and configuring the output data rate. Below, a list of the supported smart sensors and magnetometers can be found (for detailed supported physical sensor ODR, please check in datasheet):

Table 1: List of supported sensors

Sensor	**Available data rates**
Integrated gyroscope in BHI260AP	25 – 800 Hz
Integrated accelerometer in BHI260AP	12.5 – 800 Hz
Integrated gyroscope in BHI360 or BHI380	25 – 800 Hz
Integrated accelerometer in BHI360 or BHI380	12.5 – 800 Hz
BMM150 External Hall magnetometer	1.5625 – 50 Hz
BMM350 External TMR magnetometer	1.5625 – 400 Hz

2.4 Sensor data output

The smart sensor provides a wide range of sensor outputs. In this application note, only the outputs generated by the BSX sensor Fusion Software will be summarized and explained, which form the content of the next chapter.

3 Virtual sensors

When reading data from the smart sensor, the BSX system provides various outputs in the form of virtual sensors. Some of these virtual sensors also contain accuracy information, which is provided either with the signal together in the payload or as meta event. The following table lists all BSX virtual sensors that are currently available:

Table 2: Outputs for BSX software

FIFO Event	ID (Non Wake-up)	ID (Wake-up)	Payload Format	Accuracy Meta event	Types of signal	Scale Factor Format	Bytes in FIFO	Units of measure after scaling	Requires magnetometer sensor
Rotation Vector	34	35	Quaternion+	N	Defined by format "Quaternion+"	Signed 16-bit fixed point integer, least significant byte first, scaled by 2^{-14}	11	-	BMM150 or BMM350
Game Rotation Vector	37	38	Quaternion+	N	Defined by format "Quaternion+"	Signed 16-bit fixed point integer, least significant byte first, scaled by 2^{-14}	11	-	-
Geomagnetic Rotation Vector	40	41	Quaternion+	N	Defined by format “Quaternion+”	Signed 16-bit fixed point integer, least significant byte first, scaled by 2^{-14}	11	-	BMM150 or BMM350
Orientation	43	44	Euler	Y	Defined by format "Euler"	Signed 16-bit fixed point integer, least significant byte first, scaled by (360 / 2^{15})	7	degrees	BMM150 or BMM350
Accelerometer Passthrough	1	-	3D Vector	N	Dynamic [1)]	Scale = g / (Range) Range = 2048LSB, 4096LSB, 8192LSB, 16384LSB	7	g	-
Gyroscope Passthrough	10	-	3D Vector	N	Dynamic [1)]	Scale = (2 * PI / 360) / (32768.0 / Range) Range = 250dps, 500dps, 1000dps, 2000dps (default)	7	rad/s	-
Magnetometer Passthrough	19	-	3D Vector	N	Unmodified raw data of sensor	Scale = Range / 32768.0 Range = 2000 uT	7	uT	BMM150 or BMM350
Accelerometer Corrected	4	6	3D Vector	Y	Dynamic [1)]	Scale = g / (Range) Range = 2048LSB, 4096LSB, 8192LSB, 16384LSB	7	g	-
Magnetometer Corrected	22	24	3D Vector	Y	corrected sensor data	Scale = Range / 32768.0 Range = 2000 uT	7	uT	BMM150 or BMM350
Gyroscope Corrected	13	15	3D Vector	Y	Dynamic [1)]	Scale = (2 * PI / 360) / (32768.0 / Range) Range = 250dps, 500dps, 1000dps, 2000dps (default)	7	rad/s	-
Gravity	28	29	3D Vector	N	Dynamic [1)]	Scale = g / (Range) Range = 2048LSB, 4096LSB, 8192LSB, 16384LSB	7	g	-
Linear Acceleration	31	32	3D Vector	N	Dynamic [1)]	Scale = g / (Range) Range = 2048LSB, 4096LSB, 8192LSB, 16384LSB	7	g	-

Raw Accelerometer	3	7	3D Vector	N	Dynamic [1]	Scale = g / (Range) Range = 2048LSB, 4096LSB, 8192LSB, 16384LSB	7	g	-
Raw Magnetometer	21	25	3D Vector	N	prefiltered data of sensor	Scale = Range / 32768.0 Range = 2000 uT	7	uT	BMM150 or BMM350
Raw Gyroscope	12	16	3D Vector	N	Dynamic [1]	Scale = (2 * PI / 360) / (32768.0 / Range) Range = 250dps, 500dps, 1000dps, 2000dps (default)	7	rad/s	-
Accelerometer Offset	5	91	3D Vector	Y	Dynamic [1]	Scale = g / (Range) Range = 2048LSB, 4096LSB, 8192LSB, 16384LSB	7	g	-
Magnetometer Offset	23	93	3D Vector	Y	Offset value	Scale = Range / 32768.0 Range = 2000 uT	7	uT	BMM150 or BMM350
Gyroscope Offset	14	92	3D Vector	Y	Dynamic [1]	Scale = (2 * PI / 360) / (32768.0 / Range) Range = 250dps, 500dps, 1000dps, 2000dps	7	rad/s	-

[1] Dynamic: scaled to current dynamic range of sensor

Every virtual sensor data consists of a 1-Byte Sensor ID and a multi-byte payload. The size of the virtual sensor is defined by the column “Bytes in FIFO” in the table shown above. This value includes the Sensor ID byte. The format of the payload is described in the following section.

3.1 Data formats

Format “Quaternion+”

For the three rotation vectors (Rotation Vector, Game Rotation Vector, Geomagnetic Rotation Vector), the “Quaternion+” format is used, where the first Byte (Byte 0) refers to the Sensor ID, the next 8 Bytes (Byte 1 to 8) refer to the orientation in quaternion representation (known as “Rotation vector”).

Table 3: Virtual Sensor Event Format “Quaternion+”

Byte Number	Contents	Format
0	Sensor ID (Rotation Vector, etc.)	8-bit unsigned
1 .. 2	X component of quaternion	Signed 16-bit fixed point integer, least significant byte first, scaled by 2^{-14}
3 .. 4	Y component of quaternion	Signed 16-bit fixed point integer, least significant byte first, scaled by 2^{-14}
5 .. 6	Z component of quaternion	Signed 16-bit fixed point integer, least significant byte first, scaled by 2^{-14}

7 .. 8	W component of quaternion	Signed 16-bit fixed point integer, least significant byte first, scaled by 2^{-14}
9 .. 10	Estimated accuracy in radians	Unsigned 16-bit fixed point integer, least significant byte first, scaled by 2^{-14}

The meaning of the field “Estimated accuracy in radians” for the Quaternion+ data format refers specifically to the accuracy of the heading or yaw component and varies depending on the specific virtual sensor as described below:

- For the Game Rotation Vector, the “Estimated accuracy in radians” is not applicable and is therefore reported as 0.
- For the Rotation Vector, the “Estimated accuracy in radians”, the heading accuracy is represented by a value ranging from 5°(pi/180°) = 0.0873 to pi radians. This value indicates the heading error between the estimated rotation vector and the true orientation of the device, where the value 0.0873 indicates the highest heading accuracy, where pi indicates the poorest heading accuracy.

Format “Euler”

The Orientation Sensor outputs the device orientation as Euler angles: heading, pitch, and roll.

Table 4: Virtual Sensor Event Format “Euler”

Byte Number	Contents	Format
0	Sensor ID (Orientation)	8-bit unsigned.
1 .. 2	Heading	Signed 16-bit fixed point integer, least significant byte first, scaled by (360° / 2^{15})
3 .. 4	Pitch	Signed 16-bit fixed point integer, least significant byte first, scaled by (360° / 2^{15})
5 .. 6	Roll	Signed 16-bit fixed point integer, least significant byte first, scaled by (360° / 2^{15})

Format "3D Vector"

For the many 3 axis sensors (e.g., Accelerometer, Magnetometer, Gyroscope, Gravity, the following layout is used. The scale factor of the values depends on the type of the sensor and optionally on the range setting. Please check table 2 for details.

Table 5: Virtual Sensor Event Format "3D Vector"

Byte Number	Contents	Format
0	Sensor ID	8-bit unsigned.
1 .. 2	X	Signed 16-bit fixed point integer, least significant byte first
3 .. 4	Y	Signed 16-bit fixed point integer, least significant byte first
5 .. 6	Z	Signed 16-bit fixed point integer, least significant byte first

For more information about virtual sensor data formats please refer to BHI360 datasheet chapter of FIFO Data Types and Format.

3.2 ACCELEROMETER PASS THROUGH

This acceleration data represents unprocessed measurements of the acceleration experienced by the target device, recorded by the 3-Axis accelerometer. It provides information about the magnitude and direction of the acceleration experienced by the target device due to the action of an external force, without any manipulation or interpretation of the data.

The data format type is "3D Data". For additional information, see the chapter 3.1 Data formats and the Table 2: Outputs for BSX software.

3.3 RAW ACCELEROMETER

In the default firmware version, this virtual sensor is identical to the pass through acceleration virtual sensor described above. The raw acceleration is available as a separate virtual sensor because, in specific customized versions of the firmware, it can be modified to activate a low-pass filter to denoise the acceleration signal. This filtering is not activated by default.

3.4 ACCELEROMETER CORRECTED

The corrected acceleration virtual sensor provides an offset calibrated acceleration signal, which is defined as the signal wherein the inherent bias or offset (zero g offset) present in the output signal of the accelerometer, when at rest and perpendicular to the gravitational force, has been negated through an internal calibration process. This procedure ensures that the calibrated output signal provides accurate measurements of the acceleration experienced by the object in focus.

The calibration process can vary in accuracy, characterized by values ranging between 0 and 3, indicating the level of precision achieved. Here is what each rating signifies:

- A rating of 0 indicates that no calibration has been performed.
- A rating of 1 indicates that the calibration process has been initiated but with low accuracy.
- A rating of 2 indicates medium accuracy for the estimated offsets.
- A rating of 3 indicates that the calibration process is complete, achieving the highest accuracy for the estimated offsets.

Refer to the Section 4.1. for guidance on how to calibrate the device.

The data format type is “3D Data”. For additional information, see the chapter 3.1 Data formats and the Table 2: Outputs for BSX software.

3.5 ACCELEROMETER OFFSET

This virtual sensor provides the estimated acceleration offsets obtained during accelerometer calibration routine. Calibration procedures for the device are detailed in Section 4.1. The value of the acceleration offset is used in order to obtain the virtual sensor described in 3.4 corrected acceleration.

The data format type is “3D Data”. For additional information, see the chapter 3.1 Data formats and the Table 2: Outputs for BSX software.

3.6 PASS THROUGH GYROSCOPE

This data represents unprocessed measurements of the angular rate experienced by the target device, as recorded by the gyroscope. It provides information about the magnitude and direction of the angular velocity experimented due to rotations affecting the target device, without any manipulation or interpretation of the data.

3.7 RAW GYROSCOPE

In the default firmware version, this virtual sensor is identical to the pass through angular rate virtual sensor described above. The raw angular rate is available as a separate virtual sensor because, in specific customized versions of the firmware, it can be modified to activate a low-pass filter to denoise the angular rate signal. This filtering is not activated by default.

The data format type is “3D Data”. For additional information, see the chapter 3.1 Data formats and the Table 2: Outputs for BSX software.

3.8 GYROSCOPE CORRECTED

The corrected angular rate virtual sensor provides an offset calibrated gyroscope signal, which is defined as the signal wherein the inherent bias or offset (zero-rate offset) present in the output signal of the gyroscope, when at rest, has been negated through an internal calibration process. This procedure ensures that the calibrated output signal provides accurate measurements of the angular rate experienced by the object in focus.

The calibration process can vary in accuracy, characterized by values ranging between 0 and 3, indicating the level of precision achieved. Here is what each rating signifies:

- A rating of 0 indicates that no calibration has been performed.

- A rating of 1 indicates that the calibration process has been initiated but with low accuracy.
- A rating of 2 indicates medium accuracy for the estimated offsets.
- A rating of 3 indicates that the calibration process is complete, achieving the highest accuracy for the estimated offsets.

Refer to the Section 4.2 for guidance on how to calibrate the device.

The data format type is “3D Data”. For additional information, see the chapter 3.1 Data formats and the Table 2: Outputs for BSX software.

3.9 GYROSCOPE OFFSET

This virtual sensor provides the estimated gyroscope offsets obtained during gyroscope calibration routine. Calibration procedures for the device are detailed in Section 4.2. The value of the gyroscope offset is used to obtain the virtual sensor described in 3.8 corrected angular rate.

The data format type is “3D Data”. For additional information, see the chapter 3.1 Data formats and the Table 2: Outputs for BSX software.

3.10 MAGNETOMETER PASS THROUGH

This virtual sensor gives the input magnetic field signal, which is passed through BSX without any processing.

The data format type is “3D Data”. For additional information, see the chapter 3.1 Data formats and the Table 2: Outputs for BSX software.

3.11 RAW MAGNETOMETER

The raw magnetic field is available as a separate virtual sensor because, in specific customized versions of the firmware, it can be modified to activate a low-pass filter to denoise the magnetic field signal. This filtering is activated by default for the magnetometer.

The data format type is “3D Data”. For additional information, see the chapter 3.1 Data formats and the Table 2: Outputs for BSX software.

3.12 MAGNETOMETER CORRECTED

The magnetic field signal presented here is Zero-B offset compensated, which means that any initial offset from the sensor is removed during the calibration process. The resulting output is known as the bias compensated output.

The calibration process can vary in accuracy, characterized by values ranging between 0 and 3, indicating the level of precision achieved. Here is what each rating signifies:

- A rating of 0 indicates that no calibration has been performed.
- A rating of 1 indicates that the calibration process has been initiated but with low accuracy.
- A rating of 2 indicates medium accuracy for the estimated offsets.

- A rating of 3 indicates that the calibration process is complete, achieving the highest accuracy for the estimated offsets.

Refer to the Section 4.3 for guidance on how to calibrate the device.

The data format type is “3D Data”. For additional information, see the chapter 3.1 Data formats and the Table 2: Outputs for BSX software.

3.13 MAGNETOMETER OFFSET

This virtual sensor provides the estimated magnetometer offsets obtained during magnetometer calibration routine. Calibration procedures for the device are detailed in Section 4.3. The value of the magnetometer offset is used to obtain the virtual sensor described in 3.12 corrected magnetic field.

The data format type is “3D Data”. For additional information, see the chapter 3.1 Data formats and the Table 2: Outputs for BSX software.

3.14 ROTATION VECTOR

The rotation vector sensor reports the orientation of the target device relative to the East-North-Up (ENU) coordinate frame. Generally, this is achieved through the integration of readings from the accelerometer, gyroscope, and magnetometer sensors.

The rotation process involves rotating the device by an angle theta around an axis (rotation axis). This rotation transitions involves going from the reference (ENU aligned) orientation coordinate system to the current device coordinate system. The rotation is given as a unit quaternion, a method that avoids the gimbal lock issue which is observed in Euler representation. Along with the quaternion value, the estimated heading accuracy is provided, given in radians value ranging from 0.0873 to pi (5 to 180 degrees, 0.0873 = 5°(pi/180°)). It indicates the heading accuracy from the estimated rotation vector. 0.0873 indicates the highest heading accuracy, where pi indicates the poorest heading estimation. The accuracy value is calculated based on the accuracy of the sensors and the drift in gyroscope depending on the usage of magnetometer.

The data format type is “Quaternion+”. For additional information, see the chapter 3.1 Data formats and the Table 2: Outputs for BSX software.

3.15 GAME ROTATION VECTOR

Similar as rotation vector, game rotation vector is also a quaternion which represents the transformation from the reference (East-North-Up aligned) coordinate system to the current device coordinate system, utilizing only accelerometer and gyroscope signals. In other terms, only IMU data is processed. Since it does not rely on a magnetic field, it cannot determine the true horizontal direction (absolute heading). The orientation of the device is computed only with respect to the initial position. Furthermore, as this virtual sensor does not utilize magnetometer data, the game rotation output maybe subject of heading drift over time.

Since it doesn’t provide the true heading information, the heading absolute accuracy along with the game rotation vector would always be zero and doesn’t have any specific meaning. The data format type is “Quaternion+”. For additional information, see the chapter 3.1 Data formats and the Table 2: Outputs for BSX software.

3.16 GEOMAGNETIC ROTATION VECTOR

Similar to the rotation vector, the geomagnetic vector is a specialized algorithm that calculates the orientation of a device with respect to the Earth's magnetic field. This algorithm combines only the readings from the magnetometer and accelerometer sensors to provide a more precise estimate of orientation than each sensor could provide on its own. The heading value is determined with the help of tilt compensation based on accelerometer data. Along with the quaternion value, the estimated heading accuracy is provided, given in radians value ranging from 0.0524 to pi (3 to 180 degrees, 0.0524 = 3°(pi/180°)). It indicates the heading accuracy from the estimated rotation vector. 0.0524 indicates the highest heading accuracy, where pi indicates the poorest heading estimation. The accuracy value is calculated based on the magnetic field.

The data format type is “Quaternion+”. For additional information, see the chapter 3.1 Data formats and the Table 2: Outputs for BSX software.

3.17 ORIENTATION

This virtual sensor gives the orientation of the target device using Euler angles. To represent the orientation of the device, it can be done either using Euler angles or a quaternion. The quaternion representation is used in the rotation vector, game rotation vector, and geomagnetic vector outputs, explained before. Euler angles are more human-readable than quaternion representation. The Reference coordinate system employed is the ENU system detailed in Chapter 2.2 *Reference coordinate system and sensor alignment*. When utilizing the 9DoF BSX fusion library with an available magnetometer, the Orientation virtual sensor provides absolute orientation, akin to the rotation vector.

According to Euler's rotation theorem, any special rotation can be described using three angles. The Euler angles from BSX consist of 3 values: pitch, roll, and yaw or heading.

The below table offers additional details about Euler angles.

Table 6: Euler angles

Euler angle	Heading (Yaw)	Pitch	Roll
Rotation about	Z axis	X axis	Y axis
Range in radians	$[0\ 2\pi]$	$\left[-\frac{\pi}{2}\ \ \frac{\pi}{2}\right]$	$[-\pi\ \ \pi]$
Positive rotation*	0 → North $\frac{\pi}{2}$ → East π → South $\frac{3\pi}{2}$ → West	tilting the top edge of the device toward the ground creates a positive pitch angle	tilting the left edge of the device toward the ground creates a positive roll angle

*Assuming the initial device position as the depicted in the figure below

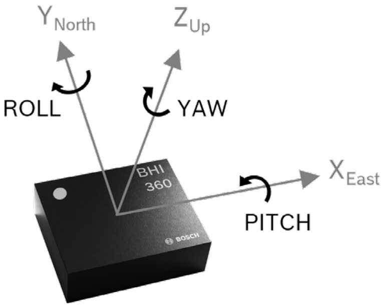


Figure 4: Representation of Euler angles convention for BHI360

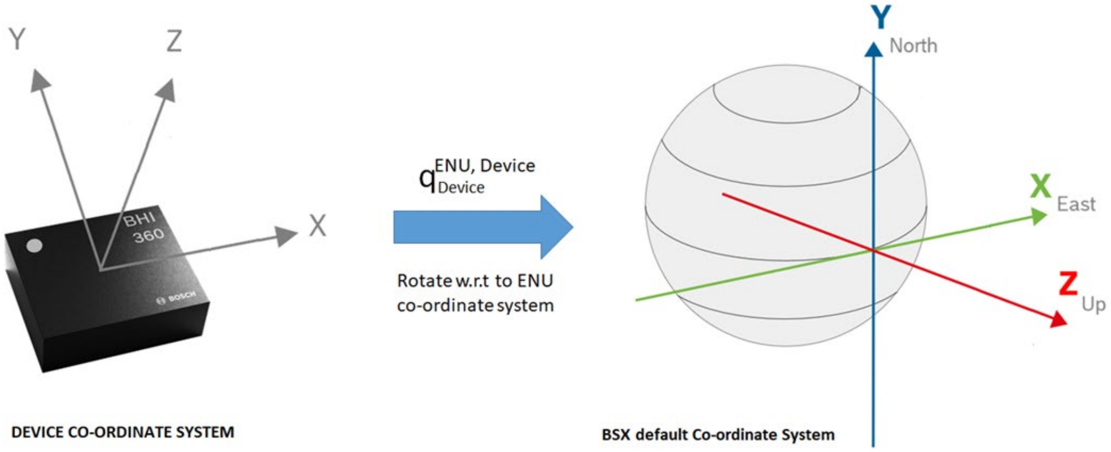

Y
Z
BHI 360
X
DEVICE CO-ORDINATE SYSTEM
q ENU, Device Device
Rotate w.r.t to ENU co-ordinate system
Y North
X East
Z Up
BSX default Co-ordinate System

Figure 5: Example of smart sensor initial position

Specifying the heading, pitch and roll without specifying their sequence is incomplete. To transition from the reference coordinate system to the device coordinate system (the definition of the rotation vector), adhere to the following sequence: Z (-heading) → X (-pitch) → Y (-roll). The angles are negated (indicated by the '-' signs) to conform to the standard convention for positive rotations using the right-hand rule, since the positive rotations used for Euler angles in the BHI360 coordinate system uses the left-hand rule.

A description showing the change in orientation output angles when rotations from 0 to 360 degrees are applied in different axes is provided below.

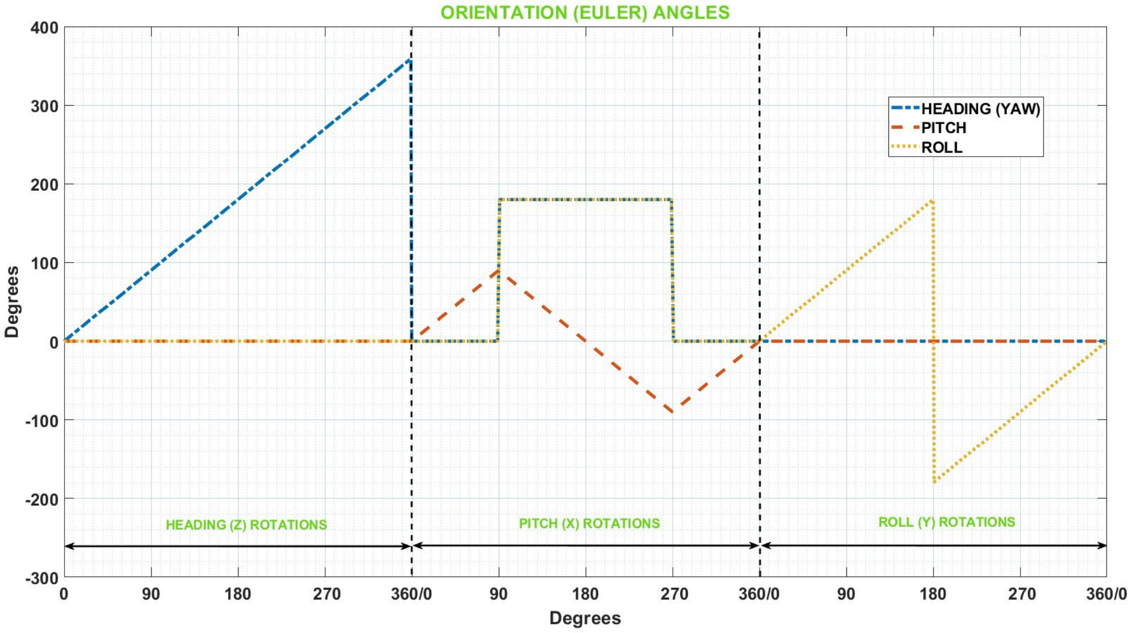


Figure 6: BSX default convention: rotations performed vs orientation output

To convert the rotation vector quaternions to Euler angles, the following conversion formulas can be applied

$$\begin{bmatrix} heading \\ pitch \\ roll \end{bmatrix} = \begin{bmatrix} arctan\left(\dfrac{2q_xq_y - 2q_wq_z}{q_y^2 - q_z^2 + q_w^2 - q_x^2}\right) \\ \arcsin\left(-2q_yq_z - 2q_wq_x\right) \\ arctan\left(\dfrac{2q_xq_z - 2q_wq_y}{q_z^2 - q_y^2 - q_x^2 + q_w^2}\right) \end{bmatrix}$$

The data format type is “Euler”. For additional information, see the chapter 3.1 Data formats and the Table 2: Outputs for BSX software.

3.18 LINEAR ACCELERATION

The linear acceleration output is an algorithm included in the BSX sensor fusion software that exclusively measures the inertial acceleration component derived from the acceleration experienced by the device.

The inertial acceleration is the component of the acceleration caused by changes in the velocity of an object, which result from alterations in speed or direction of movement. This measurement is independent of the gravitational acceleration component, which is measured by the accelerometer and present in other virtual sensors.

In general terms, the linear acceleration algorithm functions by integrating the readings from the device's accelerometer and gyroscope sensors. With the help of estimated orientation, BSX library can subtract the gravitational acceleration component from the total experienced acceleration measured by the accelerometer sensor. This process ensures that only acceleration caused by changes in velocity is registered. The linear acceleration is denoted in the unit of m/s^2.

The data format type is “3D Vector”. For additional information, please refer to Chapter 3.1 Data formats and Table 2: Outputs for BSX software.

3.19 GRAVITY

The gravity vector is an algorithm included in the BSX sensor fusion software that calculates the gravitational acceleration component derived from the acceleration experienced by the device.

The gravitational acceleration is the component of the acceleration caused by the gravitational force generated by a large mass, such as Earth. This measurement is independent of the inertial acceleration component measured by the accelerometer and present in other virtual sensors.

The gravity vector is calculated using the orientation information by fusing the accelerometer and gyroscope signals. The output of the gravity vector is a three-dimensional vector that points in the direction of the gravitational force. This vector can be used to determine the orientation of the device in three-dimensional space and can be combined with other sensor fusion algorithms to provide a more complete picture of the device's motion and orientation.

If the target device is placed horizontally relative to the ground, the gravity vector will display a value near 9.8m/s^2 on the z-axis and approximately 0 m/s^2 in other two axes. If the device is inclined at an angle, this acceleration due to gravity is distributed in all axes according to the device orientation.

The data format type is "3D Vector". For further details, refer to Section 3.1 Data formats and Table 2: Outputs for BSX software.

4 Calibration software

This chapter gives details about how to perform sensor calibration making use of the BSX software calibration algorithms. Sensor data from physical sensors may show deviations from the ideal response due to intrinsic properties and exposure to different environmental conditions. Offset or bias is one of the most important error parameters causing deviation of the sensor signals with respect to the ideal theoretical measurement value. To correct the offset or bias present in the raw data, calibration logic is present in BSX software for each sensor and the corrected data is effectively used for sensor fusion.

Different calibration movements must be performed for each physical sensor to facilitate the calibration logic in the library. The accelerometer sensor requires 6-axis movements for calibration. Whereas gyroscope calibration is based on the device being kept stable and, magnetometer calibration is based on the figure-eight movements. Detailed descriptions are provided in the subsections below.

The calibration process can vary in accuracy, characterized by values ranging between 0 and 3, indicating the level of precision achieved. Here is what each rating signifies:

- A rating of 0 indicates that no calibration has been performed.
- A rating of 1 indicates that the calibration process has been initiated but with low accuracy.
- A rating of 2 indicates medium accuracy for the estimated offsets.
- A rating of 3 indicates that the calibration process is complete, achieving the highest accuracy for the estimated offsets.

4.1 Accelerometer calibration

A detailed description of how to perform 6-axis accelerometer calibration movements is provided below:

1. Keep the device flat on a surface, so that the sensor is oriented in the direction +Z (acc_x= 0 g, acc_y = 0 g, acc_z = 1 g) and keep it stable for 2 seconds
2. Rotate -90 degrees on the Pitch axis, so that the sensor is oriented in the direction +Y (acc_x = 0 g, acc_y = 1 g, acc_z = 0 g) and keep it stable for 2 seconds.
3. Rotate 180 degrees on the Pitch axis, so that the sensor is oriented in the direction -Y (acc_x = 0 g, acc_y = -1 g, acc_z = 0 g) and keep it stable for 2 seconds.
4. Return to the initial position, then rotate 90 degrees on the Roll axis, so that the sensor is oriented in the direction +X (acc_x = 1 g, acc_y = 0 g, acc_z = 0 g) and keep it stable for 2 seconds.
5. Rotate 180 degrees on the Roll axis, so that the sensor is oriented in the direction -X (acc_x = -1 g, acc_y = 0 g, acc_z = 0 g) and keep it stable for 2 seconds.
6. Keep the device upside down, so that the sensor is oriented in the direction -Z (acc_x = 0 g, acc_y = 0 g, acc_z = -1 g) and keep it stable for 2 seconds.
7. The acceleration signal is now calibrated, and the acceleration calibration status should be 3.

It is performed as 6-axis acceleration calibration movements to eliminate the offsets in all six axes.

4.2 Gyroscope calibration

To calibrate the gyroscope signal, keep the device stable for about 10 to 12 seconds. The gyroscope calibration is complete when the calibration accuracy becomes 3. As the offset is observed when the device is at rest, in order to calibrate it, the device needs to be kept in a stable position for a period of time.

The Gyroscope calibration module has a limitation that slow constant rotations below 3°/s are considered noise and will not be tracked.

4.3 Magnetometer Calibration

To perform an accurate calibration of the magnetometer, hold the device in your hand and execute a figure-eight motion in three dimensions. Calibration is confirmed when the calibration accuracy value reaches 3. When the device is rotated in multiple orientations, the magnetic data should ideally form a sphere centered at (0, 0, 0). Enough data points must be collected to ensure the magnetic sphere fully forms. For best results, a three-dimensional figure-eight motion is the most effective way to gather the necessary magnetic data.

5 Interacting with BSX Fusion Software using BHy2CLI Tool

The BSX Fusion software can run in all Bosch Sensortec smart sensor systems, including BHA260AB, BHI260AB, BHI260AP, BHI360 and BHI380. These smart sensors are designed to provide a compact and integrated solution for sensor fusion applications, with built-in processing and fusion capabilities.

In the following chapters we will introduce various tools to interact with the BSX fusion software, using BHI360 as a case study.

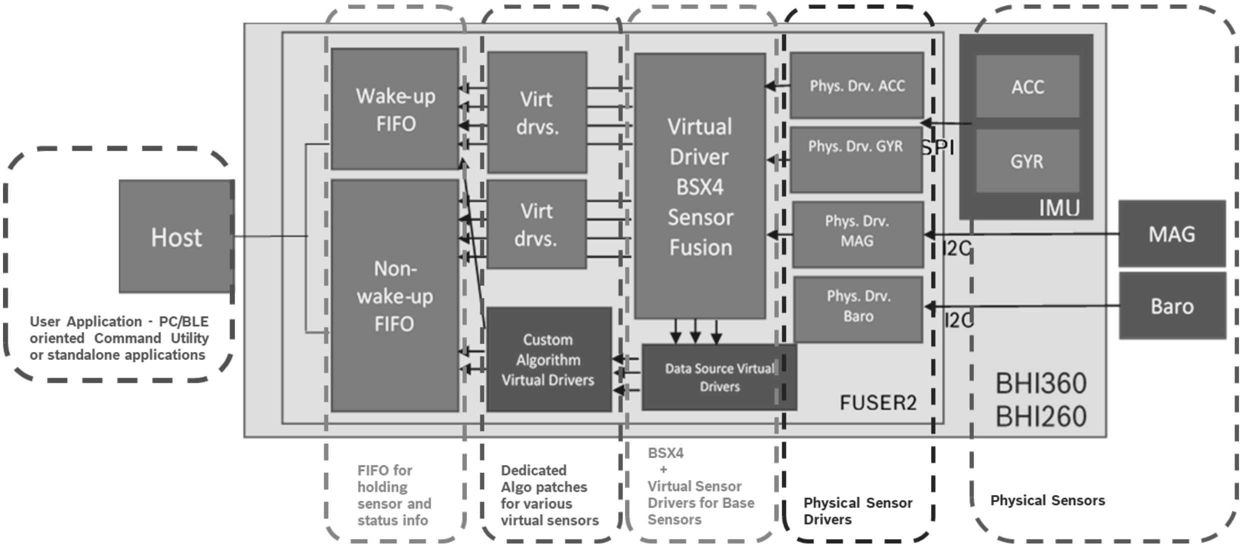

Host
User Application - PC/BLE oriented Command Utility or standalone applications
Wake-up FIFO
Non-wake-up FIFO
Virt drvs.
Virt drvs.
Custom Algorithm Virtual Drivers
Virtual Driver BSX4 Sensor Fusion
Data Source Virtual Drivers
Phys. Drv. ACC
Phys. Drv. GYR
Phys. Drv. MAG
Phys. Drv. Baro
FUSER2
SPI
I2C
I2C
ACC
GYR
IMU
MAG
Baro
BHI360
BHI260
FIFO for holding sensor and status info
Dedicated Algo patches for various virtual sensors
BSX4 + Virtual Sensor Drivers for Base Sensors
Physical Sensor Drivers
Physical Sensors

Figure 7: BSX Integration in BHy2XX/BHI3XX Event-Driven Software Framework

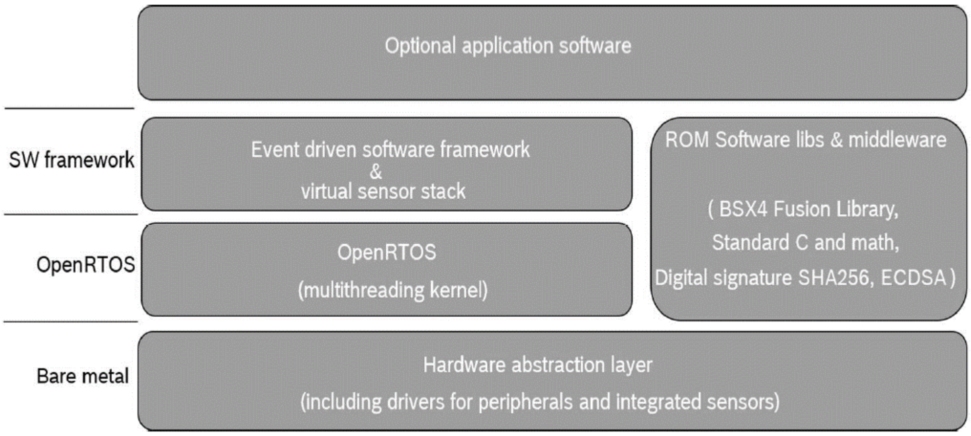

Optional application software
SW framework
Event driven software framework & virtual sensor stack
ROM Software libs & middleware
(BSX4 Fusion Library, Standard C and math, Digital signature SHA256, ECDSA)
OpenRTOS
OpenRTOS (multithreading kernel)
Bare metal
Hardware abstraction layer (including drivers for peripherals and integrated sensors)

Figure 8: Structure of the BHy2XX/BHI3XX Event-Driven Software Framework

BHy2CLI tool

The command-line interface tool BHy2CLI is an application based on the COINES ("COmmunication with INertial and Environmental Sensors") software package. COINES is a software tool that offers a low-level interface to Bosch Sensortec's Application Board 3.0 and allows developers to access and evaluate sensors. BHy2CLI facilitates communication with BHy2xx/BHI3xx series devices using predefined commands, which are converted into host interface commands. BHy2CLI is released in the COINES software package as an example source code of the BHy2xx/BHI3xx and provides a series of commands for users to manage registers, upload firmware to the smart sensor, operate parameter I/Os, and enable/disable virtual sensors, among other functions.

The detailed requirements for installing and using the BHy2CLI tool together with the Application Board 3.0 are described in the "Evaluation Setup Guide" for Smart sensors (See BST-BHI260-BHI360-AN001).

The COINES software package can be found in the Bosch Sensortec website: *COINES | Bosch Sensortec (bosch-sensortec.com)*

Once COINES has been installed, the Application Board 3.0 (with the relevant BHy2xx/BHI3xx shuttle board) has been properly connected to the PC, and the Application Board Firmware has been updated with the latest firmware, we can start using the BHyCLI tool.

In COINES, the bhy2cli tool is located at the path **C:\COINES\v2.8.8\examples\bhy2\examples\bhy2cli_ble**.

1. Use *mingw32-make* to compile and generate the bhy2cli tool.

   > mingw32-make

```
C:\COINES\v2.8.8\examples\bhy2\examples\bhy2cli_ble>mingw32-make
Platform: Windows
cc:  "C:\TDM-GCC-64\bin\gcc.exe".
"Cleaning pc..."
Deleted file - C:\COINES\v2.8.8\coines-api\libcoines-pc.a
Deleted file - C:\COINES\v2.8.8\coines-api\build\PC\ble_com.c.d
Deleted file - C:\COINES\v2.8.8\coines-api\build\PC\ble_com.c.o
Deleted file - C:\COINES\v2.8.8\coines-api\build\PC\cflags.save
Deleted file - C:\COINES\v2.8.8\coines-api\build\PC\coines_bridge.c.d
Deleted file - C:\COINES\v2.8.8\coines-api\build\PC\coines_bridge.c.o
Deleted file - C:\COINES\v2.8.8\coines-api\build\PC\serial_com.c.d
Deleted file - C:\COINES\v2.8.8\coines-api\build\PC\serial_com.c.o
Deleted file - C:\COINES\v2.8.8\coines-api\build\PC\simpleble_lib_loader.c.d
Deleted file - C:\COINES\v2.8.8\coines-api\build\PC\simpleble_lib_loader.c.o
[ MAKE ] coines-api
[ CC ] pc/coines_bridge.c
[ CC ] pc/serial_com/serial_com.c
[ CC ] pc/ble_com/simpleble_lib_loader.c
[ CC ] pc/ble_com/ble_com.c
[ AR ] libcoines-pc.a
[ LD ] bhy2cli
```

2. To get the command help info, use –h flag.

> bhy2cli.exe –h

```
C:\COINES\v2.8.8\examples\bhy2\examples\bhy2cli_ble>bhy2cli.exe -h
Host Interface : SPI
Copyright (c) 2023 Bosch Sensortec GmbH
Version 0.4.9 Build date: Nov  3 2023
Usage:
bhy2cli [<options>]
Options:
  -h OR help    = Print this usage message
  version
        = Prints the version
  -v OR verb <verbose level>
        = Set the verbose level. 0 Error, 1 Warning, 2 Infos
  -b OR ramb <firmware path>
        = Reset, upload specified firmware to RAM and boot from RAM
          [equivalent to using "reset ram <firmware> boot r" successively]
   -d OR flb <firmware path>
        = Reset, upload specified firmware to Flash and boot from Flash
          [equivalent to using "reset fl <firmware path> boot f" successively]
  -n OR reset    = Reset sensor hub
…
```

3. Trigger a soft reset for the smart sensor system.

   >bhy2cli.exe –n

```
C:\COINES\v2.8.8\examples\bhy2\examples\bhy2cli_ble>bhy2cli.exe -n
Host Interface : SPI
Copyright (c) 2023 Bosch Sensortec GmbH
Version 0.4.9 Build date: Nov  3 2023
Reset successful
```

4. Reset, and then load the BSX firmware to RAM and boot.

   >bhy2cli.exe –b <firmware>

```
C:\COINES\v2.8.8\examples\bhy2\examples\bhy2cli_ble>bhy2cli.exe -b
C:\COINES\v2.8.8\examples\bhy2\firmware\bhi360\BHI360.fw
Host Interface : SPI
Copyright (c) 2023 Bosch Sensortec GmbH
Version 0.4.9 Build date: Nov  3 2023
Reset successful
Uploading 127228 bytes of firmware to RAM
Uploading firmware to RAM successful
Waiting for firmware verification to complete
Boot Status : 0x38: No flash installed. Host interface ready. Firmware verification
done.
[D][META EVENT WAKE UP]; T: 0.512093750; Firmware initialized. Firmware version 5991
[D][META EVENT]; T: 0.512093750; Firmware initialized. Firmware version 5991
Booting from RAM successful
```

Where <firmware> is the path to the BHI firmware image file, for example:
C:\COINES\v2.8.8\examples\bhy2\firmware\bhi360\BHI360.fw

5. Check for the presence of BSX.

   >bhy2cli.exe –r 0x24

   Read the register 0x24 to check the presence of BSX. BSX is present if the value of bits 5 to 7 equals 2. If this value equals 0, BSX is not present.

```
C:\COINES\v2.8.8\examples\bhy2\examples\bhy2cli_ble>bhy2cli.exe -r 0x24
Host Interface : SPI
Copyright (c) 2023 Bosch Sensortec GmbH
Version 0.4.9 Build date: Nov  3 2023
Register address: Data
----------------------
0x24            : 4a
Read complete
```

6. Read BSX Version.

>bhy2cli.exe –s 0x027E

```
C:\COINES\v2.8.8\examples\bhy2\examples\bhy2cli_ble>bhy2cli.exe -s 0x027E
Host Interface : SPI
Copyright (c) 2023 Bosch Sensortec GmbH
Version 0.4.9 Build date: Nov  3 2023
Byte hex      dec | Data
-------------------------------------------
0x000000        0 |04 00 54 03
Reading parameter 0x027E successful
```

The 4-byte BSX Version information can be read with the following parameter in Table 7:

Table 7: BSX version

Field Name	**Byte Offset**	**Description**	**Access**
Parameter ID	0x00 - 0x01	BSX Version: 0x027E	-
Length	0x02 - 0x03	Number of bytes to follow = 4	-
Major Version	0x04	BSX Major Version	Read only
Minor version	0x05	BSX Minor version	Read only
Major bug fix version	0x06	BSX Major bug fix version	Read only
Minor bug fix version	0x07	BSX Minor bug fix version	Read only

7. Read BSX Algorithm Parameters

   >bhy2cli.exe –s <Parameter ID>

The user can read the Parameter ID of BSX in Table 8 to know the calibration state. The BSX library currently supports accelerometer, gyroscope, and magnetometer as physical sensors, so the available parameters within this range are 0x0201, 0x0203, and 0x0205.

```
C:\COINES\v2.8.8\examples\bhy2\examples\bhy2cli_ble>bhy2cli.exe -s 0x0201
Host Interface : SPI
Copyright (c) 2023 Bosch Sensortec GmbH
Version 0.4.9 Build date: Nov  3 2023
Byte hex      dec | Data
-------------------------------------------
0x000000        0 |00 40 48 00 03 54 00 04
0x000008        8 |00 30 00 40 04 00 6f 00
0x000010       16 |00 00 3e 00 00 00 0b 00
0x000018       24 |10 00 00 01 00 00 00 00
0x000020       32 |00 00 00 00 00 00 0b 00
0x000028       40 |08 00 00 04 00 00 06 00
0x000030       48 |10 00 00 01 00 00 00 00
0x000038       56 |00 00 00 00 00 00 06 00
0x000040       64 |08 00 00 02
Reading parameter 0x0201 successful
```

Table 8: Algorithm parameters

Parameter ID	Usage	Content Format	Access
0x0201	Calibration state stream for accelerometer	BSX State Exchange Structure	Read/Write
0x0203	Calibration state stream for gyroscope		Read/Write
0x0205	Calibration state stream for magnetometer		Read/Write
0x027D	SIC Matrix		Read/Write

The calibration state stream contains intermediate calibration states for each sensor data. Save these states when the accuracy of the corresponding sensor reaches level 3. Load them at system boot for quicker start-up.

8. Show the list of virtual sensors from the BSX software.

>bhy2cli.exe –i

```
C:\COINES\v2.8.8\examples\bhy2\examples\bhy2cli_ble>bhy2cli.exe -i
Host Interface : SPI
Copyright (c) 2023 Bosch Sensortec GmbH
Version 0.4.9 Build date: Nov  3 2023
Product ID     : 89
Kernel version : 5991
User version   : 5991
ROM version    : 5166
Power state    : active
Host interface : SPI
Feature status : 0x4a
Boot Status : 0x38: No flash installed. Host interface ready. Firmware verification
done.
Virtual sensor list.
Sensor ID |                          Sensor Name |  ID | Ver |  Min rate |  Max rate |
----------+--------------------------------------+-----+-----+-----------+-----------|
        1 |            Accelerometer passthrough | 205 |   1 |    1.5625 |  400.0000 |
        3 |           Accelerometer uncalibrated | 203 |   1 |    1.5625 |  400.0000 |
        4 |              Accelerometer corrected | 241 |   1 |    1.5625 |  400.0000 |
        5 |                 Accelerometer offset | 209 |   1 |    1.0000 |    1.0000 |
        6 |      Accelerometer corrected wake up | 192 |   1 |    1.5625 |  400.0000 |
        7 |   Accelerometer uncalibrated wake up | 204 |   1 |    1.5625 |  400.0000 |
       10 |                Gyroscope passthrough | 207 |   1 |    1.5625 |  400.0000 |
       12 |               Gyroscope uncalibrated | 244 |   1 |    1.5625 |  400.0000 |
       13 |                  Gyroscope corrected | 243 |   1 |    1.5625 |  400.0000 |
       14 |                     Gyroscope offset | 208 |   1 |    1.0000 |    1.0000 |
       15 |                    Gyroscope wake up | 194 |   1 |    1.5625 |  400.0000 |
       16 |       Gyroscope uncalibrated wake up | 195 |   1 |    1.5625 |  400.0000 |
       28 |                       Gravity vector | 247 |   1 |    1.5625 |  400.0000 |
       29 |               Gravity vector wake up | 198 |   1 |    1.5625 |  400.0000 |
       31 |                  Linear acceleration | 246 |   1 |    1.5625 |  400.0000 |
       32 |          Linear acceleration wake up | 197 |   1 |    1.5625 |  400.0000 |
       37 |                 Game rotation vector | 252 |   1 |    1.5625 |  400.0000 |
       38 |         Game rotation vector wake up | 200 |   1 |    1.5625 |  400.0000 |
       43 |                          Orientation | 254 |   1 |    1.5625 |  400.0000 |
       44 |                  Orientation wake up | 202 |   1 |    1.5625 |  400.0000 |
      136 |               Low Power Step counter | 249 |   1 |    1.0000 |    1.0000 |
      137 |              Low Power Step detector | 248 |   1 |    1.0000 |    1.0000 |
      142 |                 Low Power Any motion | 176 |   1 |    1.0000 |    1.0000 |
      153 |                   Multi Tap Detector | 182 |   1 |    1.0000 |    1.0000 |
      154 |   Activity recognition for Wearables | 237 |   1 |    1.0000 |    1.0000 |
      156 |      Low Power Wrist Gesture wake up | 228 |   1 |    1.0000 |    1.0000 |
      158 |         Low Power Wrist Wear wake up | 178 |   1 |    1.0000 |    1.0000 |
      159 |          Low Power No Motion wake up | 181 |   1 |    1.0000 |    1.0000 |
```

9. Enable the virtual sensor of accelerometer.

>bhy2cli.exe –c <number of virtual sensor>: <ODR in Hz>

e.g., bhy2cli -c 4:1

Other virtual sensors can also be enabled similarly.

```
C:\COINES\v2.8.8\examples\bhy2\examples\bhy2cli_ble>bhy2cli -c 4:1
Host Interface : SPI
Copyright (c) 2023 Bosch Sensortec GmbH
Version 0.4.9 Build date: Nov  3 2023


[D][META EVENT]; T: 2080.714953125; Flush complete for sensor id 4
[D][META EVENT]; T: 2080.715562500; Power mode changed for sensor id 4
[D][META EVENT]; T: 2080.715562500; Sample rate changed for sensor id 4
[D]SID: 4; T: 2081.191640625; x: 0.020752, y: -0.107910, z: 1.016113; acc: 0
[D][META EVENT]; T: 2081.191640625; Accuracy for sensor id 4 changed to 0
[D]SID: 4; T: 2081.829093750; x: 0.018555, y: -0.110352, z: 1.013428; acc: 0
[D]SID: 4; T: 2082.466500000; x: 0.019043, y: -0.108398, z: 0.995361; acc: 0
[D][META EVENT]; T: 2082.466500000; Accuracy for sensor id 4 changed to 1
[D]SID: 4; T: 2083.103984375; x: 0.018311, y: -0.108643, z: 0.995361; acc: 1
[D]SID: 4; T: 2083.741546875; x: 0.018311, y: -0.108398, z: 0.995361; acc: 1
[D]SID: 4; T: 2084.379109375; x: 0.018799, y: -0.108154, z: 0.995605; acc: 1
```

10. Enable multiple virtual sensors simultaneously.

>bhy2cli.exe –c <number of virtual sensor>: <ODR in Hz> -c <…>:<…>
e.g., bhy2cli -c 4:25 -c 13:25

```
C:\COINES\v2.8.8\examples\bhy2\examples\bhy2cli_ble>bhy2cli -c 4:25 -c 13:25
Host Interface : SPI
Copyright (c) 2023 Bosch Sensortec GmbH
Version 0.4.9 Build date: Nov  3 2023

[D][META EVENT]; T: 2084.774015625; Power mode changed for sensor id 4
[D][META EVENT]; T: 2084.774015625; Sample rate changed for sensor id 4
[D][META EVENT]; T: 2425.020203125; Power mode changed for sensor id 4
[D][META EVENT]; T: 2425.020203125; Sample rate changed for sensor id 4
[D][META EVENT]; T: 2425.028828125; Power mode changed for sensor id 13
[D][META EVENT]; T: 2425.028828125; Sample rate changed for sensor id 13
[D][META EVENT]; T: 2425.028609375; Flush complete for sensor id 4
[D][META EVENT]; T: 2425.028609375; Flush complete for sensor id 13
[D]SID: 4; T: 2425.276234375; x: 0.017822, y: -0.110107, z: 0.997070; acc: 0
[D][META EVENT]; T: 2425.276234375; Accuracy for sensor id 4 changed to 1
[D]SID: 13; T: 2425.276234375; x: 1.159668, y: 0.000000, z: -0.305176; acc: 0
[D][META EVENT]; T: 2425.276234375; Accuracy for sensor id 13 changed to 0
[D]SID: 4; T: 2425.316015625; x: 0.017822, y: -0.108643, z: 0.994873; acc: 1
[D]SID: 13; T: 2425.316015625; x: 1.159668, y: 0.000000, z: -0.244141; acc: 0
[D]SID: 4; T: 2425.355796875; x: 0.018066, y: -0.109375, z: 0.994629; acc: 1
```

In this example, both the virtual sensor 4 (accelerometer) and the virtual sensor 13 (gyroscope) are enabled at the same time.

6 Using an application example

This section describes how to use an example based on COINES and use the BHI360 on the APP3.0 platform. The same example can be used for BHI260AP just by including different header files of firmware. You can find and include the BHI2 or BHI3 header file under “bhy2\firmware”.

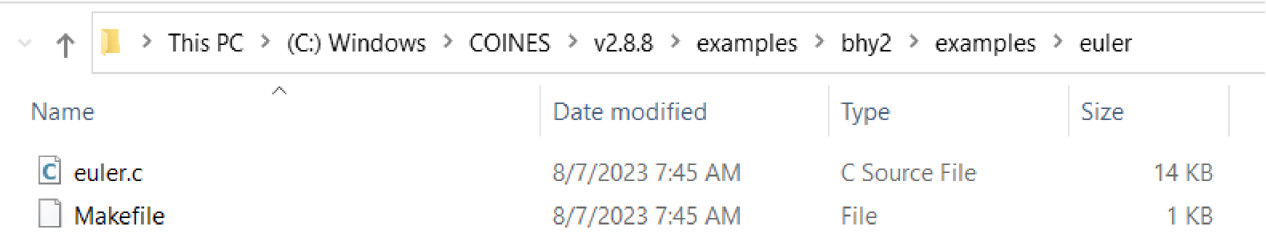
6.1 Reference platform setup

Below are instructions for setting up a reference platform:

1. Use the APP3.0 Board and the BHI360 shuttle.
2. Connect the APP3.0 board to a computer via a USB connection.
3. Download COINES that comes with the BHy2 Sensor APIs and install it properly.

For information on how to set up the complete environment, refer to the software requirements and hardware requirements in the Evaluation Setup Guide Document.

4. After installing COINES successfully, the example folder “COINES\version\examples\bhy2\examples” is created. An example of Euler stream output based on BHY2xx/BHI3xx is included in this folder. In this example, Euler stream can be retrieved by the host system.



This PC > (C:) Windows > COINES > v2.8.8 > examples > bhy2 > examples > euler

Name	Date modified	Type	Size
euler.c	8/7/2023 7:45 AM	C Source File	14 KB
Makefile	8/7/2023 7:45 AM	File	1 KB

The provided code is a C program for interfacing with the BHY2xx/BHI3xx sensor using the BHY2 API. It shows how to initialize the sensor, upload firmware, and read Euler angle data (heading, pitch, roll) from the sensor's FIFO. The code also manages various meta events.

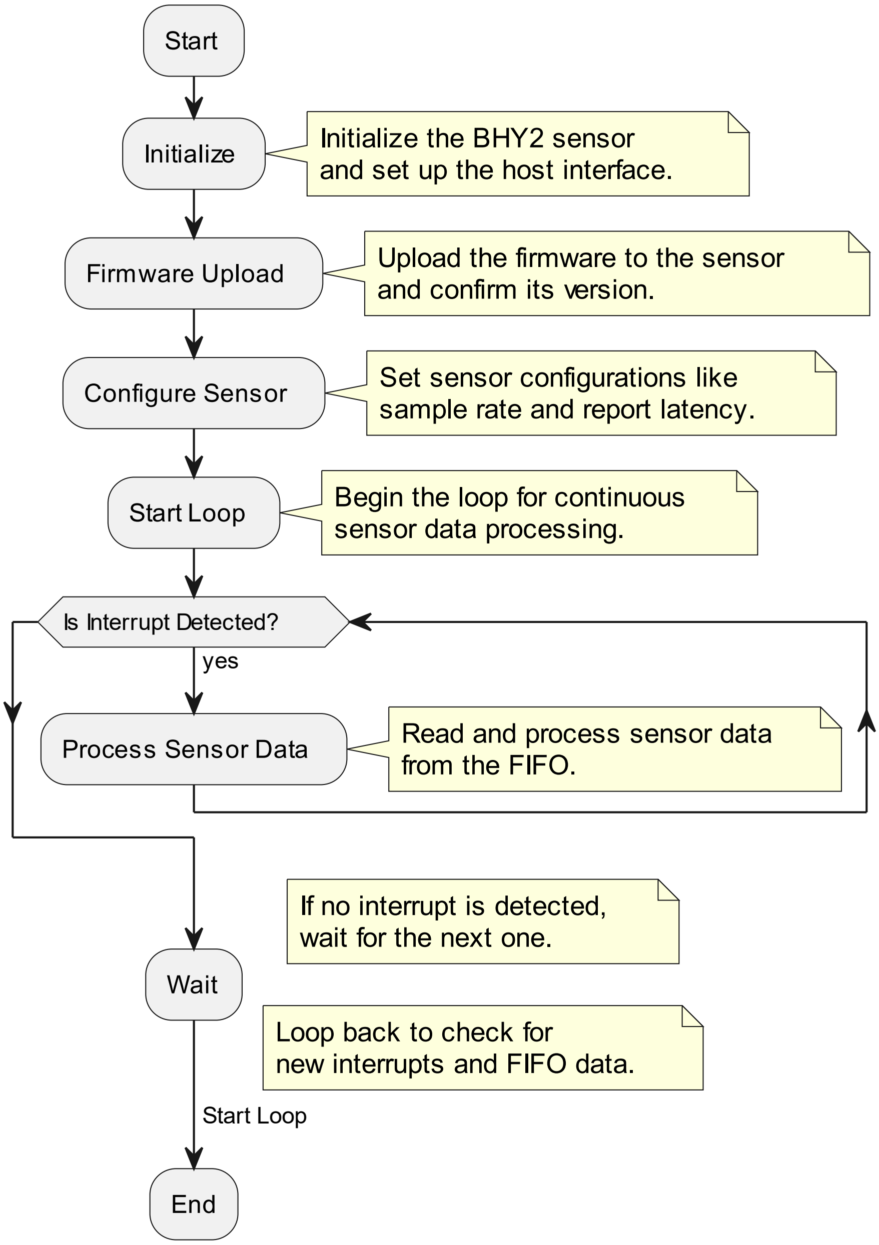
6.2 Explanation of the euler.c example:

The example consists of the following Key parts:

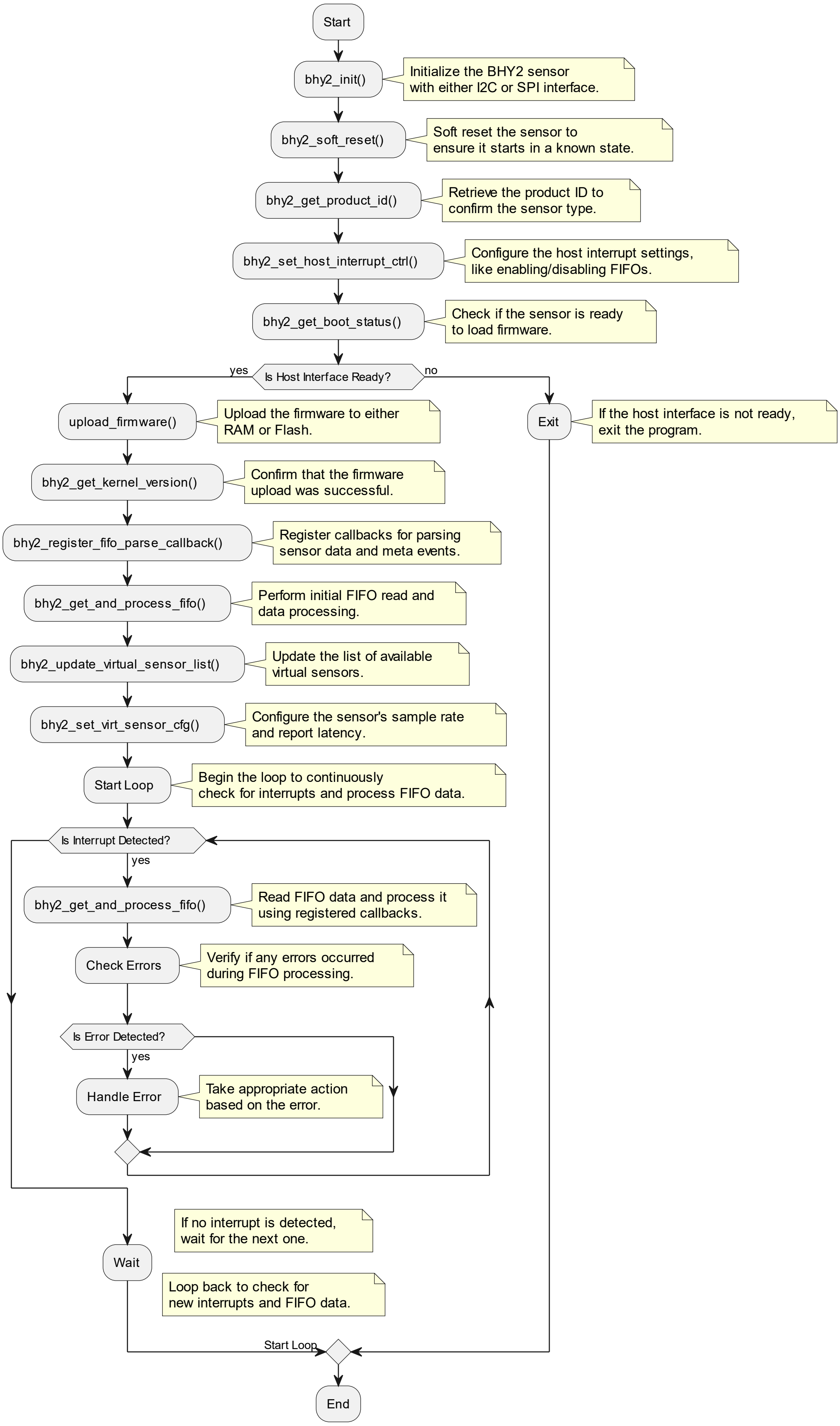
1. **Initialize**: Initializes the BHY2 device and checks for a valid product ID.
2. **Firmware Upload**: Depending on the flag UPLOAD_FIRMWARE_TO_FLASH, it uploads firmware to either RAM or Flash.

3. **Configure Sensor**: Registers callbacks for parsing Euler data and meta events, and sets the sensor's sample rate.
4. **Process Sensor Data**: Continuously reads and processes data from the sensor's FIFO.

The flow diagram below shows the simplified structure of the program.


Start
Initialize
Initialize the BHY2 sensor and set up the host interface.
Firmware Upload
Upload the firmware to the sensor and confirm its version.
Configure Sensor
Set sensor configurations like sample rate and report latency.
Start Loop
Begin the loop for continuous sensor data processing.
Is Interrupt Detected?
yes
Process Sensor Data
Read and process sensor data from the FIFO.
If no interrupt is detected, wait for the next one.
Wait
Loop back to check for new interrupts and FIFO data.
Start Loop
End

A more detailed flow diagram showing the program structure, the functions actually used, and the main task of each function can be seen in the Figure below:


Start
bhy2_init()
Initialize the BHY2 sensor with either I2C or SPI interface.
bhy2_soft_reset()
Soft reset the sensor to ensure it starts in a known state.
bhy2_get_product_id()
Retrieve the product ID to confirm the sensor type.
bhy2_set_host_interrupt_ctrl()
Configure the host interrupt settings, like enabling/disabling FIFOs.
bhy2_get_boot_status()
Check if the sensor is ready to load firmware.
yes
Is Host Interface Ready?
no
upload_firmware()
Upload the firmware to either RAM or Flash.
Exit
If the host interface is not ready, exit the program.
bhy2_get_kernel_version()
Confirm that the firmware upload was successful.
bhy2_register_fifo_parse_callback()
Register callbacks for parsing sensor data and meta events.
bhy2_get_and_process_fifo()
Perform initial FIFO read and data processing.
bhy2_update_virtual_sensor_list()
Update the list of available virtual sensors.
bhy2_set_virt_sensor_cfg()
Configure the sensor's sample rate and report latency.
Start Loop
Begin the loop to continuously check for interrupts and process FIFO data.
Is Interrupt Detected?
yes
bhy2_get_and_process_fifo()
Read FIFO data and process it using registered callbacks.
Check Errors
Verify if any errors occurred during FIFO processing.
Is Error Detected?
yes
Handle Error
Take appropriate action based on the error.
If no interrupt is detected, wait for the next one.
Wait
Loop back to check for new interrupts and FIFO data.
Start Loop
End

Most important Functions used:

- Call `bhy2_init()` to initialize the interface and reference code.
- Call `bhy2_soft_reset()` performs a software reset on the BHY device .
- Call `bhy2_set_host_interrupt_ctrl()` to configure host interrupt control and host interface control settings of the BHY device
- Call `bhy2_get_boot_status()` to check if the sensor is ready to load firmware.
- Call `upload_firmware()` to upload RAM firmware image or Flash firmware image to the BHY device. For the BHI360, only RAM boot is supported.
- Call `bhy2_get_kernel_version()` to read out the kernel version to confirm that booting up was successfull.
- Call `bhy2_register_fifo_parse_callback()`, to register the callback of parsing the meta event and sensor data from the sensor's FIFO in order to receive the required sensor data.
- Call `bhy2_set_virt_sensor_cfg()` to enable the specific virtual sensor and to set the ODR and latency to enable virtual sensors (ODR > 0, sensor enabled; ODR = 0, sensor disabled).
- Call `parse_euler()` parses Euler angle data and prints it
- Call `parse_meta_event()` to handle meta events like sensor errors, FIFO overflow, etc.
- Call `print_api_error()` prints API error messages

6.3 Running the Euler data stream output example

It is advisable to check that the header files (bhy2.h, bhy2_parse.h, common.h) and firmware files (BHI260AP_aux_BMM150-flash.fw.h, BHI260AP_aux_BMM150.fw.h for BHI260AP or BHI360_Aux_BMM150.fw.h for BHI360) in your project.

The Orientation example is located under the folder: COINES\<version>\examples\bhy2\examples\euler

If the euler.exe executable file is not available, start the build process by running the make command below (from the ...\euler folder):

```
> mingw32-make
```

Ensure that the sensor is properly connected to either the I2C or SPI interface as configured.

The example is run with the command below:

```
> .\euler.exe
```

The example code includes the following executions:

1. Enable the Euler data stream output sensor.
2. Enable host interrupt control.
3. Load the firmware into RAM.
4. Initialize the firmware.

Console output:

```
C:\COINES\v2.8.8\examples\bhy2\examples\euler>euler.exe
Host Interface : SPI
BHI260/BHA260/BHI360 found. Product ID read 89
Host interrupt control
    Wake up FIFO enabled.
    Non wake up FIFO enabled.
    Status FIFO disabled.
    Debugging disabled.
    Fault enabled.
    Interrupt is active high.
    Interrupt is level triggered.
    Interrupt pin drive is push-pull.
Loading firmware into RAM.
Booting from RAM.
Boot successful. Kernel version 5991.
[META EVENT WAKE UP] Firmware initialized. Firmware version 5991
[META EVENT] Firmware initialized. Firmware version 5991
Enable Orientation wake up at 100.00Hz.
[META EVENT WAKE UP] Power mode changed for sensor id 44
[META EVENT WAKE UP] Sample rate changed for sensor id 44
SID: 44; T: 0.935500000; h: 1.439209, p: 17.753906, r: 9.206543; acc: 0
[META EVENT WAKE UP] Accuracy for sensor id 44 changed to 0
SID: 44; T: 0.945437500; h: 1.428223, p: 17.742920, r: 9.195557; acc: 0
SID: 44; T: 0.955390625; h: 1.439209, p: 17.731934, r: 9.195557; acc: 0
SID: 44; T: 0.965328125; h: 1.439209, p: 17.720947, r: 9.206543; acc: 0
SID: 44; T: 0.975281250; h: 1.439209, p: 17.709961, r: 9.206543; acc: 0
SID: 44; T: 0.985234375; h: 1.439209, p: 17.698975, r: 9.206543; acc: 0
SID: 44; T: 0.995171875; h: 1.439209, p: 17.677002, r: 9.217529; acc: 0
SID: 44; T: 1.005125000; h: 1.439209, p: 17.666016, r: 9.217529; acc: 0
SID: 44; T: 1.015078125; h: 1.439209, p: 17.655029, r: 9.217529; acc: 0
SID: 44; T: 1.025015625; h: 1.439209, p: 17.644043, r: 9.206543; acc: 0
SID: 44; T: 1.034968750; h: 1.439209, p: 17.644043, r: 9.195557; acc: 0
SID: 44; T: 1.044906250; h: 1.439209, p: 17.633057, r: 9.195557; acc: 0
SID: 44; T: 1.054859375; h: 1.439209, p: 17.622070, r: 9.195557; acc: 0
```

7 Performance Indicators

Key Performance Indicators (KPIs) are defined for the following BSX Sensor Fusion Modules:

- BSX IMU Fusion
- BSX NDOF Fusion
- BSX E-Compass Fusion

These indicators are:

- Orientation Accuracy: Accuracy for Heading, Pitch and Roll rotations.
    A. Static Accuracy: Accuracy for Pitch and Roll rotations for slow movements.
    B. Dynamic Accuracy: Accuracy for Pitch and Roll rotations during fast movements.

- Orientation Stabilization Time: Time required for the fusion to correct an error in the orientation.
- Static Heading Noise (Heading resolution): Noise in the orientation angles when the device is kept stable.
- Calibration time: Time required for the sensor to be calibrated.

Table 9: Performance Indicators*

				Rotation Vector (NDOF: A+M+G)		Game Rotation Vector (IMU: A+G)		Geomagnetic Rotation Vector (Compass: A+M)	
Parameter		**Unit**	**Conditions**	**typical**	**max**	**typical**	**max**	**typical**	**max**
Static accuracy	Heading	degrees	Ambient temperature: 25 °C Horizontal geomagnetic field component: 30μT Heading accuracy measured as average over various device orientations	2	5	n.a.	n.a.	7	10
	Static Heading drift	degrees/ min		n.a.	n.a.	0.01	0.03	n.a.	n.a.
	Pitch	degrees		2	3	2	3	2	3
	Roll	degrees		2	3	2	3	2	3
Dynamic Accuracy	Heading	degrees		7	10	n.a.	n.a.	15	20
	Dynamic heading drift	degrees/ min		n.a.	n.a.	0.6	3	n.a.	n.a.
	Pitch	degrees		7	10	7	10	7	10
	Roll	degrees		7	10	7	10	7	10
Orientation Stabilization time		s		0.2		0.2		2	
Mag calibration time		s		1.6		n.a.	n.a.	5.32	

**Please note that all the KPIs are measured at 100Hz using normal mode after triggering CRT feature and performing the recommended sensor calibration*

8 Document history and modification

Rev. No	Chapter	Description of modification/changes	Date
0.0		Initial public release with BHI360 support	November 2023

Bosch Sensortec GmbH
Gerhard-Kindler-Straße 9
72770 Reutlingen / Germany

contact@bosch-sensortec.com
www.bosch-sensortec.com

Modifications reserved
Preliminary - specifications subject to change without notice
Document number: BST-BHI260_BHI360-AN002-01