Software Documentation

BSXlite
API Documentation

Bosch Sensortec

BOSCH
Invented for life

API Documentation – API description for the BSXlite Library

Document revision	1.2
Document release date	24.02.2015
Document number	BST-BSX03X-SD001-01
Technical reference code(s)	
Notes	Data 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's appearance.

INDEX OF CONTENTS

1. About the BSXlite API Documentation

This manual describes data types and interfaces of BSXlite Fusion Library. This document can be used for detailed information.

1.1 Who should read this?

This information is intended for users who want to perform integration of BSXlite fusion library on various supported platforms and want to get detailed information on BSXlite interfaces and data types. The basic prerequisites for the BSXlite integration are described in the 'Integration Guideline for BSXlite Library' and it is recommended to read this document in beforehand.

2. File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

BsxLibraryDatatypes.h (This file provides the data types used in BSXlite API)

BsxLiteFusionLibrary.h (This file defines the interface of BSXlite Fusion Library)

3. Data Structrures

3.1 BsxLibraryDatatypes.h File Reference

Data Structures

- struct ts_version
- struct ts_dataxyz
- struct ts_dataxyzf32
- struct ts_dataeulerf32
- struct ts_dataquatf32
- struct ts_calibparam
- struct ts_calibprofile
- struct ts_workingModes
- struct ts_HWsensorSwitchList
- struct sensordata_t
- struct libraryinput_t
- struct initParam_t

Data Structure Documentation

initParam_t Struct Reference

Data Fields

- BSX_U8 * **accelspec** acceleration sensor configuration
- BSX_U8 * **magspec** magnetic sensor configuration
- BSX_U8 * **gyrospec** gyroscope sensor configuration
- BSX_U8 * **usecase** use case configuration
- BSX_U8 **accelspec_status** acceleration initialization return status
- BSX_U8 **magspec_status** magnetometer initialization return status
- BSX_U8 **gyrospec_status** gyroscope initialization return status
- BSX_U8 **usecase_status** use case initialization return status
-

libraryinput_t Struct Reference

Contains raw 9DoF sensor data and timestamps as input for BSXlite Fusion Library

Data Fields

- **sensordata_t acc** 3-axis accelerometer data with measurement time stamp
- **sensordata_t mag** 3-axis magnetometer data with measurement time stamp
- **sensordata_t gyro** 3-axis gyroscope data with measurement time stamp

sensordata_t Struct Reference

Contains 3-axis sensor data and measurement time stamp

Data Fields

• ts_dataxyzs32 data	3-axis sensor data
• BSX_U32 **time_stamp**	measurement time stamp

ts_calibparam Struct Reference

Contains sensor calibration parameters for magnetometer or gyroscope. Sensor calibration parameters are estimated by BSXlite Fusion Library.

Data Fields

• ts_dataxyz offset	3-axis offset
• BSX_S16 **radius**	radius in case of magnetometer calibration parameters

ts_calibprofile Struct Reference

Contains estimated sensor calibration parameters and accuracy of these parameters

Data Fields

• **ts_calibparam calibParam**	estimated calibration parameters
• BSX_U8 **accuracy**	accuracy

ts_dataeulerf32 Struct Reference

Orientation in Euler angles.

Data Fields

• BSX_F32 **h**	heading, rotation around the Z axis.
• BSX_F32 **p**	pitch, rotation around the X axis
• BSX_F32 **r**	roll, rotation around the Y axis.
• BSX_F32 **y**	unused

ts_dataquatf32 Struct Reference

Quaternion data.

Data Fields

• BSX_F32 **w**	q[0] data of vector q
• BSX_F32 **x**	q[1] data of vector q

- BSX_F32 **y** q[2] data of vector q
- BSX_F32 **z** q[3] data of vector q

ts_dataxyz Struct Reference

3-axis data as S16

Data Fields

- BSX_S16 **x** x-axis data
- BSX_S16 **y** y-axis data
- BSX_S16 **z** z-axis data

ts_dataxyzf32 Struct Reference

3-axis data as F32

Data Fields

- BSX_F32 **x** x-axis data
- BSX_F32 **y** y-axis data
- BSX_F32 **z** z-axis data

ts_HWsensorSwitchList Struct Reference

Contains flags for hardware sensors, if selected working mode requires data from respective sensor.

Data Fields

- BSX_U8 **acc**
- BSX_U8 **mag**
- BSX_U8 **gyro**

ts_Version Struct Reference

Contains BSXlite version number

Data Fields

- BSX_S16 **major** major version number
- BSX_S16 **minor** minor version number
- BSX_S16 **majorbugFix** bux fix version number
- BSX_S16 **minorbugFix** internal version number

ts_workingModes Struct Reference

Contains working modes to be set. BSXlite supports only one 9DoF working mode.

Data Fields

- BSX_U32 **opMode**   working mode

4. BSXlite API

4.1 BsxLiteFusionLibrary.h File Reference

4.1.1 GENERAL API INTERFACE

- BSX_S8 bsx_get_version (ts_version *version)
- BSX_S8 bsx_init (initParam_t*inputparams)
- BSX_S8 bsx_reset (void)
- BSX_S8 bsx_dostep (libraryinput_t *libraryinput_p)
- BSX_S8 bsx_dopreprocess (libraryinput_t *libraryinput_p)
- BSX_S8 bsx_docalibration (void)
- BSX_S8 bsx_dousecase (void)
- BSX_S8 bsx_set_workingmode (ts_workingModes *workingModes)
- BSX_S8 bsx_get_hwdependency (ts_workingModes workingModes, ts_HWsensorSwitchList *HWsensorSwitchList)
- BSX_S8 bsx_get_calibrationcalltick (BSX_U8 *calibtick)
- BSX_S8 bsx_get_usecasecalltick (BSX_U8 *usecasetick)
- BSX_S8 bsx_set_magcalibprofile (ts_calibprofile *calibprofile)
- BSX_S8 bsx_get_magcalibprofile (ts_calibprofile *calibprofile)
- BSX_S8 bsx_set_gyrocalibprofile (ts_calibprofile *calibprofile)
- BSX_S8 bsx_get_gyrocalibprofile (ts_calibprofile *calibprofile)
- BSX_S8 bsx_set_acccalib_accuracythreshold (BSX_U8)
- BSX_S8 bsx_set_acccalibprofile (ts_calibprofile *)
- BSX_S8 bsx_get_acccalibprofile (ts_calibprofile *)
- BSX_S8 bsx_reset_acccalib ()

4.1.2 DATA INTERFACES

- BSX_S8 bsx_get_accrawdata (ts_dataxyzf32 *rawAccData)
- BSX_S8 bsx_get_acccordata (ts_dataxyzf32 *)
- BSX_S8 bsx_get_accoffset (ts_dataxyzf32 *)
- BSX_S8 bsx_get_acccalibaccuracy (BSX_U8 *)
- BSX_S8 bsx_get_accoffsets_mg (ts_dataxyzf32)
- BSX_S8 bsx_get_magrawdata (ts_dataxyzf32 *rawmagdata)
- BSX_S8 bsx_get_magcordata (ts_dataxyzf32 *corMagData)
- BSX_S8 bsx_get_magoffsets (ts_dataxyzf32 *offset)
- BSX_S8 bsx_get_magcalibaccuracy (BSX_U8 *calibaccuracy)
- BSX_S8 bsx_get_gyrorawdata_rps (ts_dataxyzf32 *rawgyrodata)
- BSX_S8 bsx_get_gyrocordata_rps (ts_dataxyzf32 *corgyrodata)
- BSX_S8 bsx_get_gyrocalibaccuracy (BSX_U8 *gyrocalibaccuracy)
- BSX_S8 bsx_get_orientdata_quat (ts_dataquatf32 *quatData)
- BSX_S8 bsx_get_georotationvector_quat (ts_dataquatf32 *georotationquat)
- BSX_S8 bsx_get_orientdata_euler_rad (ts_dataeulerf32 *eulerDataRad)
- BSX_S8 bsx_get_headingaccuracy_rad (BSX_F32 *headingaccrad)
- BSX_S8 bsx_get_geoheadingaccuracy_rad (BSX_F32 *headingaccrad)
- BSX_S8 bsx_get_orient_datastatus (BSX_U8 *datastatus)

4.1.3 Function Documentation

4.1.3.1 BSX_S8 bsx_docalibration (void)

Performs calibration of sensors (magnetometer and gyroscope). Before calling bsx_docalibration sensor data has to be provided to library by bsx_dopreprocess.

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.2 BSX_S8 bsx_dopreprocess (libraryinput_t * libraryinput_p)

Provides raw sensor data to library and performs preprocessing of input data.

Parameters:

libraryinput_p	-> pointer to sensor data structure which includes sensor data S32 type and time stamp in microseconds of U64 type

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.3 BSX_S8 bsx_dostep (libraryinput_t * *libraryinput_p*)

The main process API gets the raw accelerometer, magnetometer and gyroscope data in LSB format and the current system time in microseconds as input and does the required process based on the configuration.

Parameters:

libraryinput_p	-> pointer to sensor data structure which includes sensor data S32 type and time stamp in microseconds of U64 type

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.4 BSX_S8 bsx_dousecase (void)

Calculates fusion data and has to be called after bsx_docalibration.

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.5 BSX_S8 bsx_get_accrawdata (ts_dataxyzf32 * *rawAccData*)

Get the raw accelerometer data (x,y and z direction) in m/s^2.

Parameters:

**rawAccData,:*	pointer to accelerometer raw data structure

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.6 BSX_S8 bsx_get_calibrationcalltick (BSX_U8 * *calibtick*)

This API gives the calibration tick – It is an external task calling tick which will be set at the end of bsx_dopreprocess, and is used as a trigger input for calibration.

Parameters:

calibtick	-> pointer to get calib tick

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.7 BSX_S8 bsx_get_geoheadingaccuracy_rad (BSX_F32 * *headingaccrad*)

Get estimated heading accuracy of geo magnetic rotation vector in radians

Parameters:

**headingaccrad*	-> Pointer to read the error in heading in radians

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.8 BSX_S8 bsx_get_georotationvector_quat (ts_dataquatf32 * *georotationquat*)

Get the geomagnetic rotation vector as quaternion. Geomagnetic rotation vector is based on accelerometer and magnetometer data only and does not use gyroscope data.

Parameters:

**georotationquat*	-> Orientation quaternion data (w,x,y,z)

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.9 BSX_S8 bsx_get_gyrocalibaccuracy (BSX_U8 * *gyrocalibaccuracy*)

Get the gyroscope calibration accuracy status. This indicates the status of gyroscope calibration. If status reaches 3, it is considered as high accuracy status.

Parameters:

**gyrocalibaccuracy*	-> current gyroscope calibration accuracy status

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.10 BSX_S8 bsx_get_gyrocalibprofile (ts_calibprofile * *calibprofile*)

Get the gyroscope calibration profile (includes offsets and accuracy status).

Usage: If the initial offsets need to be retained to reduce the time taken for initial calibration, use this APIs to store the well estimated offsets. Pre Condition Check: Accuracy status should be stable and high for a certain time.

Parameters:

**calibprofile->*	pointer to gyroscope calibration profile structure

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.11 BSX_S8 bsx_get_gyrocordata_rps (ts_dataxyzf32 * *corgyrodata*)

Get the raw 3-axis gyroscope data in radians/sec.

Parameters:

corgyrodata	-> gyro data in radians/sec

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.12 BSX_S8 bsx_get_gyrorawdata_rps (ts_dataxyzf32 * *rawgyrodata*)

Get the raw 3-axis gryoscope data in radians/sec.

Parameters:

rawgyrodata	-> gyro data in radians/sec

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.13 BSX_S8 bsx_get_headingaccuracy_rad (BSX_F32 * *headingaccrad*)

Get the heading data accuracy in radians. Heading status refers to accuracy level of orientation data (from library) from true heading calculation. This comparison is done with magnetic data coupling.

Parameters:

**headingaccrad*	-> Pointer to read the error in heading in radians

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.14 BSX_S8 bsx_get_hwdependency (ts_workingModes *workingModes*, ts_HWsensorSwitchList * *HWsensorSwitchList*)

Get the sensor switch list for the current working mode. HWsensorSwitchList is a structure of three elements BSX_U8 acc,BSX_U8 mag,BSX_U8 gyro. This function gets the status of these three elements for the given working mode.

Parameters:

**workingModes*	-> Pointer to working mode constants
**HWsensorSwitchList*	-> Pointer to hardware switch list

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.15 BSX_S8 bsx_get_magcalibaccuracy (BSX_U8 * *calibaccuracy*)

Get the magnetometer calibration accuracy status. This indicates the status of gyroscope calibration. If status reaches 3, it is considered as high accuracy status.

Parameters:

calibaccuracy	-> current magnetometer calibration accuracy status

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.16 BSX_S8 bsx_get_magcalibprofile (ts_calibprofile * *calibprofile*)

Get the magnetometer calibration profile (includes offsets and accuracy status).

Usage: If the initial offsets need to be retained to reduce the time taken for initial calibration, use this APIs to store the well estimated offsets.

Pre Condition Check: Accuracy status should be stable and high for a certain time .

Parameters:

**calibprofile->*	pointer to magnetometer calibration profile structure

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.17 BSX_S8 bsx_get_magcordata (ts_dataxyzf32 * *corMagData*)

Get the 3-axis corrected magnetometer data in MicroTesla Corrected = raw – offset.

Parameters:

**corMagData*	-> mag data in MicroTesla

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.18 BSX_S8 bsx_get_magoffsets (ts_dataxyzf32 * *offset*)

Get the estimated parameter of the magnetometer.

Parameters:

offset	-> estimated offsets in MicroTesla

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.19 BSX_S8 bsx_get_magrawdata (ts_dataxyzf32 * *rawmagdata*)

Get the raw 3-axis magnetometer data in MicroTesla.

Parameters:

**rawmagdata*	-> mag data in MicroTesla

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.20 BSX_S8 bsx_get_orient_datastatus (BSX_U8 * *datastatus*)

Get the orientation data accuracy status as 0,1,2,3: Which 3 indicates heading is close to true magnetic heading and similarly 0 indicates unreliable.

Parameters:

datastatus	-> pointer to orientation accuracy status

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.21 BSX_S8 bsx_get_orientdata_euler_rad (ts_dataeulerf32 * eulerDataRad)

Get the orientation Euler data (heading, pitch and roll) in radians.

Parameters:

eulerDataRad	-> Euler data (h,p,r)

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.22 BSX_S8 bsx_get_orientdata_quat (ts_dataquatf32 * *quatData*)

Get the orientation quaternion data.

Parameters:

**quatData*	-> quaternion data (w,x,y,z)

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.26 BSX_S8 bsx_get_usecasecalltick (BSX_U8 * *usecasetick*)

This API gives the Usecase tick; It is an external task calling tick which will be set at the end of bsx_dousecase, and is used as a trigger input for usecase processing.

Usage: Call this api before dousecase, when usecase tick is enabled, dousecase can be called

Parameters:

usecasetick	-> pointer to get usecase tick

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.27 BSX_S8 bsx_get_version (ts_version * *version*)

Get version of the BSXlite Fusion Library. Version is a structure of four element which consists of major, minor, minorbugfix and majorbugfix.

Parameters:

**version->*	Pointer to version structure

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.28 BSX_S8 bsx_init (initParam_t* *inputparams*)

This API initializes the main library process. If the input pointer is NULL then it will initialize with default values defined in the library.

Usage: Call this API to initialize BSX module whenever new use case configuration or sensor spec is needed

Parameters:

inputparams	-> pointer to acc,mag,gyro specs and usecase config Inputparam->accelspec : Pointer to accelspec char array that holds settings for particular accelerometer sensor Inputparam->magspec : Pointer to magspec char array that holds settings for particular magnetometer sensor Inputparam->gyrospec : Pointer to gyrospec char array that holds settings for particular gyroscope sensor Inputparam->usecaseconfig : Pointer to usecase char array that holds settings for particular usecase Inputparam->accelspec_status holds the status if the spec is error free and not modified from original 0 -> error in accelerometer spec char array 1 -> No error in accelerometer spec array and spec corresponds to bma250 2 -> No error in accelerometer spec array and spec corresponds to bma255 3 -> No error in accelerometer spec array and spec corresponds to bma280 Inputparam->magspec_status holds the status if the spec is error free and not modified from original 0 -> error in magnetometer spec char array 1 -> No error in magnetometer spec array and spec corresponds to bmm050 2 -> No error in magnetometer spec array and spec corresponds to bmm150 Inputparam->gyrospec_status holds the status if the spec is error free and not modified from original 0 -> error in gyroscope spec char array 1 -> No error in gyroscope spec array and spec corresponds to bmg160 Inputparam->usecase_status holds the status if the spec is error free and not

	modified from original 0 -> error in usecase char array 1 -> No error in usecase char array

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.29 BSX_S8 bsx_reset (void)

Resets dynamic state of the library.

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.30 BSX_S8 bsx_set_gyrocalibprofile (ts_calibprofile * *calibprofile*)

Set the gyroscope calibration profile(includes offsets and accuracy status).

Used to load the stored offsets whenever there is a restart/reset based on requirement.

Note: Profile parameters offsets and accuracy status can be obtained from bsx_get_magcalibprofile api.

Parameters:

calibprofile->	gyroscope calibration profile structure variable

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.31 BSX_S8 bsx_set_magcalibprofile (ts_calibprofile * *calibprofile*)

Set the magnetometer calibration profile (calibration offset and status).

Usage: Calibration Profiling is used to save and load profiles. This API is used to load the stored offsets whenever there is a restart/reset based on the scenario and requirement. Note: Profile parameters offsets and accuracy status can be obtained from bsx_get_magcalibprofile api.

Parameters:

**calibprofile->*	pointer to magnetometer calibration profile

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.32 BSX_S8 bsx_set_workingmode (ts_workingModes * *workingModes*)

Sets the working operation mode. Working mode indicates the support of library for the selected sensor combinations.

Note: BSXlite Fusion Library supports only BSX_WORKINGMODE_NDOF_GEORV_FMC_OFF working mode. Input is a structure of one element of BSX_U32 type and contains encoded operation mode.

Parameters:

workingModes	->Pointer to working mode

Returns:

Return code

Return values:

0	-> Success
1	-> Error

4.1.3.33 BSX_S8 bsx_set_acccalib_accuracythreshold (BSX_U8)

Set the accel calib accuracy threshold for the accelerometer.

Parameters:

threshold	-> calib accuracy threshold of accelerometer

Returns:

zero for success, non-zero failed

Return values:

0	-> Success

1	-> Error

4.1.3.34 BSX_S8 bsx_set_acccalibprofile (ts_calibprofile *)

Set the accelerometer calibration profile(calib offset and status)

Parameters:

*	calibprofile: Pointer to Accelerometer calibration profile

Returns:

zero for success, non-zero failed

Return values:

0	-> Success
1	-> Error

4.1.3.35 BSX_S8 bsx_get_acccalibprofile (ts_calibprofile *)

Get the acceleromter calibration profile.

Parameters:

*	calibprofile: Pointer to Accelerometer calibration profile

Returns:

zero for success, non-zero failed

Return values:

0	-> Success
1	-> Error

4.1.3.36 BSX_S8 bsx_reset_acccalib ()

reset the accelerometer calibration module

Parameters:

none	

Returns:

zero for success, non-zero failed

Return values:

0	-> Success
1	-> Error

4.1.3.37 BSX_S8 bsx_get_acccordata (ts_dataxyzf32 *)

Get the corrected accelerometer data(x,y and z direction) in m/s^2. Corrected data = raw data – offset.

Parameters:

accData	-> accel data in ms^2

Returns:

zero for success, non-zero failed

Return values:

0	-> Success
1	-> Error

4.1.3.38 BSX_S8 bsx_get_acccalibaccuracy (BSX_U8 *)

Get the accelerometer calibration accuracy status.

Parameters:

**calibaccuracy*	-> current accelerometer calibration accuracy status

Returns:

zero for success, non-zero failed

Return values:

0	-> Success
1	-> Error

4.1.3.39 BSX_S8 bsx_get_accoffset (ts_dataxyzf32*)

Get the acc offsets.

Parameters:

**accoffsets*	-> current accelerometer offsets

Returns:

zero for success, non-zero failed

Return values:

0	-> Success
1	-> Error

4.1.3.40 BSX_S8 bsx_get_accoffsets_mg (ts_dataxyzf32*)

Get the acc offsets in mg.

Parameters:

**accoffsets_mg*	-> current accelerometer offsets in mg

Returns:

zero for success, non-zero failed

Return values:

0	-> Success
1	-> Error

5. Legal disclaimer

5.1 Engineering samples

Engineering Samples are marked with an asterisk (*) or (e) or (E). Samples may vary from the valid technical specifications of the product series contained in this data sheet. They are therefore not intended or fit for resale to third parties or for use in end products. Their sole purpose is internal client testing. The testing of an engineering sample may in no way replace the testing of a product series. Bosch Sensortec assumes no liability for the use of engineering samples. The Purchaser shall indemnify Bosch Sensortec from all claims arising from the use of engineering samples.

5.2 Product use

Bosch Sensortec products are developed for the consumer goods industry. They may only be used within the parameters of this product data sheet. They are not fit for use in life-sustaining or security sensitive systems. Security sensitive systems are those for which a malfunction is expected to lead to bodily harm or significant property damage. In addition, they are not fit for use in products which interact with motor vehicle systems.

The resale and/or use of products are at the purchaser's own risk and his own responsibility. The examination of fitness for the intended use is the sole responsibility of the Purchaser.

The purchaser shall indemnify Bosch Sensortec from all third party claims arising from any product use not covered by the parameters of this product data sheet or not approved by Bosch Sensortec and reimburse Bosch Sensortec for all costs in connection with such claims.

The purchaser must monitor the market for the purchased products, particularly with regard to product safety, and inform Bosch Sensortec without delay of all security relevant incidents.

5.3 Application examples and hints

With respect to any examples or hints given herein, any typical values stated herein and/or any information regarding the application of the device, Bosch Sensortec hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights or copyrights of any third party. The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics. They are provided for illustrative purposes only and no evaluation regarding infringement of intellectual property rights or copyrights or regarding functionality, performance or error has been made.

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

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

Modifications reserved | Printed in Germany
Specifications subject to change without notice
Document number: BST-BSX03X-SD001-01