IG Devices CAN Protocol Specifications

Document : IGDCPS.3
Revision : 3 - Dec 4, 2013
Covers Firmware V2.1

SBG Systems
3bis, chemin de la Jonchère
92500 Rueil Malmaison
FRANCE

Email : support@sbg-systems.com
Phone : +33 (0)1 80 88 45 00

Revision history

Rev.	Date	Author	Information
3	4. Dec. 2013	Nicolas MICHEL	Updated heave configuration frame
2	28. May. 2012	Alexis GUINAMARD	Updated protocol with firmware V2 features. Added a chapter for IG-20/30 specific commands Added Heave and delta angle outputs Added commands: Motion profile commands: SBG_SEND_MP_BUFFER, SBG_VALIDATE_MP_BUFFER, SBG_MP_INFO SBG_HEAVE_CONF SBG_VIRTUAL_ODO_CONF SBG_ADVANCED_OPTIONS Updated commands: SBG_ODO_CONFIG SBG_RESTORE_DEFAULT_SETTINGS SBG_LOW_POWER_MODE SBG_SYNC_OUT_CONF SYNC_IN_CONF SBG_HEADING_SOURCE SBG_USER_BUFFER Removed obsolete commands and configurations: Removed SBG_VELOCITY_CONSTRAINTS Removed SBG_FILTER_ATTITUDE_ERRORS Removed Error log commands Removed Internal magnetometer calibration methods
1	Mar 21, 2011	Raphaël Siryani	First version of this document:

Index

1. General information about protocol

1.1. Introduction

The protocol described in this documentation is used to communicate with a CAN IG-Device on a Controller-area network (CAN) bus.

The CAN bus is a message based protocol designed in a first time for automotive applications and used today in almost all industries.

The IG-Device CAN implementation supports both CAN 2.0A and CAN 2.0B standards in a very versatile manner.

This documentation contains all information needed to configure and integrate an IG-Device to a CAN bus.

In the Development Kit, SBG Systems provides a very powerful and easy to use C library that implements all IG-Device CAN features. If you have to develop your own implementation, it's still strongly recommended that you read the sbgCan Library documentation and source codes carefully.

1.2. CAN specifications

The CAN interface support the following configurations:

- Speed: 1 MBit/s, 500 kBit/s, 250 kBit/s, 125 kBit/s, 100 kBit/s, 50 kBit/s, 20 kBit/s, 10 kBit/s.
- Up to 8 bytes data field per frame
- Standard 11 bits and extended 29 bits identifiers

1.3. Frames format

All data frames sent on the CAN bus have a common format, which is described in the CAN 2.0 specifications. You can find below a short summary of a CAN frame organization:

Field	SOF	Arbitration field	Control field	Data field	CRC	ACK	EOF
Size	1 bit	12 or 32 bits	6 bits	0 to 8 bytes	16 bits	2 bits	7 bits
Description	Start of Frame	Identifier (11 or 29 bits) + frame options	Length of the data field (in bytes)	Data	CRC code	CAN Acknowledge	End of Frame

1.4. Frames identifiers and translation

Each CAN message uses a unique identifier encoded on 11 bits for a CAN 2.0A standard message or on 29 bits for a CAN 2.0B extended message. In order to avoid incompatibilities with other materials, each CAN message id can be individually defined or even disabled using the following special id:

```
	SBG_DISABLED_FRAME  0xF0000000
```

Some configuration commands, such as SBG_CAN_ID_OUTPUT_TRIGGERS_CONF (0x35), use a CAN message identifier as input. To identify a CAN message, the device always uses the default CAN message ids as listed in the next chapter. More generally, all CAN messages ids referenced in this documentations are defaults ones.

1.5. Communicating with the device

1.5.1. Configuration frames

The user can change or retrieve the device configuration by sending configuration frames. Generally, to send or get a configuration, the same CAN message id is used. Only the data part of the frame is used to determine if the user wants to set or get a setting.

When a configuration has been properly done, the device should acknowledge by sending back its new configuration. Most of the time, this acknowledge frame is exactly the same as the frame sent to configure the device. If the configuration could not be changed, or if you have provided an invalid frame, the device will return the last valid settings if possible.

The current configuration can then be saved into the non volatile memory by sending the SBG_CAN_ID_SAVE_SETTINGS (0x2F) command.

The IG-Devices CAN implementation is very versatile thus, all frames can be activated or not according to the user need. Moreover, the user may also configure each frame identifiers to fit with his own Ids.

Please find below all available configuration messages with their default CAN id.

CAN message name with default id	Description
SBG_CAN_ID_SAVE_SETTINGS (0x2F)	Save current configuration into non volatile memory.
SBG_CAN_ID_RESTORE_SETTINGS (0x30)	Restore current configuration to factory defaults.
SBG_CAN_ID_LOW_POWER_MODE (0x31)	Set/Get device and GPS power options.
SBG_CAN_ID_DEVICE_INFO (0x32)	Return device information such as id and versions.
SBG_CAN_ID_USER_ID (0x33)	Set/Get a custom user id.
SBG_CAN_ID_USER_BUFFER (0x34)	Write/Read some custom data to/from the device.
SBG_CAN_ID_OUTPUT_TRIGGERS_CONF (0x35)	Set/Get triggers configuration for an output frame.
SBG_CAN_ID_OUTPUT_MAIN_LOOP_DIVIDER (0x36)	Set/Get the main loop frequency divider.
SBG_CAN_ID_PROTOCOL_MODE (0x37)	Set/Get the CAN protocol configuration.
SBG_CAN_ID_FRAME_ID (0x38)	Set/Get the CAN id for a CAN frame.
SBG_CAN_ID_FILTER_FREQUENCIES (0x39)	Set/Get sensors low pass filters frequencies.
SBG_CAN_ID_KALMAN_FILTER (0x3A)	Set/Get main loop period and Kalman filter options.

CAN message name with default id	Description
SBG_CAN_ID_FILTER_HEADING_SOURCE (0x3C)	Set/Get reference heading source for the Kalman filter.
SBG_CAN_ID_MAGNETIC_DECLINATION (0x3D)	Set/Get the local earth magnetic declination.
SBG_CAN_ID_REFERENCE_PRESSURE (0x3E)	Set/Get the reference pressure for altitude computation.
SBG_CAN_ID_GPS_OPTIONS (0x3F)	Set/Get GPS options such as dynamic model.
SBG_CAN_ID_NAV_SOURCES (0x40)	Set/Get velocity and position aiding sources.
SBG_CAN_ID_GPS_LEVER_ARM (0x41)	Set/Get lever arm between the GPS antenna and the device.
SBG_CAN_ID_GRAVITY_MAGNITUDE (0x42)	Set/Get the local gravity magnitude.
SBG_CAN_ID_AUTO_ORIENTATION_OFFSET (0x44)	Perform an auto orientation offset computation.
SBG_CAN_ID_PRE_ORIENTATION_OFFSET (0x45)	Set/Get the pre orientation offset matrix.
SBG_CAN_ID_POST_ORIENTATION_OFFSET (0x46)	Set/Get the post orientation offset matrix.
SBG_CAN_ID_CALIB_MAG (0x47)	Perform/Save a soft/hard iron magnetometers calibration.
SBG_CAN_ID_CALIB_MAG_MANUAL (0x48)	Set/Get the magnetometers calibration matrix and offset.
SBG_CAN_ID_CALIB_GYRO_BIAS (0x49)	Perform/Save a gyroscopes bias estimation.
SBG_CAN_ID_EXT_DEVICE (0x4A)	Set/Get the external device configuration.
SBG_CAN_ID_EXT_DEVICE_CONF (0x4B)	Send/Read configurations of the external module.
SBG_CAN_ID_ODO_CONFIG (0x4C)	Set/Get general odometers configurations.
SBG_CAN_ID_ODO_DIRECTION (0x4D)	Set/Get odometers velocity direction.
SBG_CAN_ID_ODO_LEVER_ARM (0x4E)	Set/Get lever arm between the odometer and the device.
SBG_CAN_ID_LOGIC_IN_CHANNEL (0x4F)	Set/Get logic input settings.
SBG_CAN_ID_LOGIC_OUT_CHANNEL (0x50)	Set/Get logic output settings.
SBG_CAN_ID_SEND_MP_BUFFER (0x52)	Send a part of a Kalman filter motion profile
SBG_CAN_ID_VALIDATE_MP_BUFFER (0x53)	Validate a motion profile
SBG_CAN_ID_MP_INFO (0x54)	Get the current motion profile information
SBG_CAN_ID_HEAVE_CONF (0x55)	Set/get Heave configuration
SBG_CAN_ID_VIRTUAL_ODO_CONF (0x56)	Set/Get virtual odometer configuration
SBG_CAN_ID_ADVANCED_OPTIONS (0x57)	Set/Get advanced options

Table 1: Configuration frames with their corresponding id

Note: *If you don't save the current configuration into non volatile memory using the command SBG_SAVE_SETTINGS, all modifications will be lost after next device reboot*

1.5.2. Output frames

The device sends its data using different per-defined output frames. When possible, each CAN message outputs only a complete set of data such as (x,y,z) calibrated accelerometers.

All output frames can be configured with a trigger mask and can be sent using three modes:

- Continuously at each divided main loop iteration,
- After the activation of a trigger,
- Directly after a user request.

If an output is configured to be sent continuously, it will be regularly sent by the device following a predefined order defined in the 2 below and at a configurable rate.

If an output is configured to be activated by a trigger, it will be automatically sent as soon as a new data has arrived.

Triggered outputs can be very useful for example with GPS data that are updated at a low rate. If the output is configured to be triggered, then the device is able to send the new raw GPS position each time a new data is available.

The user can retrieve any output by sending an empty frame with the desired output identifier. For example, if the CAN frame used to retrieve gyroscopes uses the id 0x06 then you just have to send an empty CAN message with the identifier 0x06.

Please find below all available output messages in the order they will be sent and with their default CAN id.

CAN message name with default id	Description
SBG_CAN_ID_OUTPUT_TIMESTAMP_TRIGGER (0x00)	Device time stamp and triggers status.
SBG_CAN_ID_OUTPUT_DEVICE_STATUS (0x01)	Device status BIT field.
SBG_CAN_ID_OUTPUT_UTC_TIME (0x02)	Absolute UTC time based on GPS data.
SBG_CAN_ID_OUTPUT_QUATERNION (0x03)	3D orientation expressed using quaternions.
SBG_CAN_ID_OUTPUT_EULER (0x04)	3D orientation expressed using Euler angles.
SBG_CAN_ID_OUTPUT_HEADING (0x05)	Heading with its accuracy.
SBG_CAN_ID_OUTPUT_GYROSCOPES (0x06)	Calibrated x,y,z rate of turn values.
SBG_CAN_ID_OUTPUT_ACCELEROMETERS (0x07)	Calibrated x,y,z acceleration values.
SBG_CAN_ID_OUTPUT_MAGNETOMETERS (0x08)	Calibrated and normalized 3D magnetic field.
SBG_CAN_ID_OUTPUT_TEMPERATURES (0x09)	Calibrated internal device temperatures.
SBG_CAN_ID_OUTPUT_GYRO_TEMPERATURES (0x0A)	Calibrated internal gyroscopes temperatures.
SBG_CAN_ID_OUTPUT_POSITION_1 (0x0B)	Inertial enhanced latitude and longitude position.
SBG_CAN_ID_OUTPUT_POSITION_2 (0x0C)	Inertial enhanced altitude and position accuracy.
SBG_CAN_ID_OUTPUT_VELOCITY_1 (0x0D)	Inertial enhanced x and y velocities.
SBG_CAN_ID_OUTPUT_VELOCITY_2 (0x0E)	Inertial enhanced z velocity and velocity accuracy.
SBG_CAN_ID_OUTPUT_GYROSCOPES_RAW (0x0F)	Raw rate of turn values.
SBG_CAN_ID_OUTPUT_ACCELEROMETERS_RAW (0x10)	Raw acceleration values.
SBG_CAN_ID_OUTPUT_MAGNETOMETERS_RAW (0x11)	Raw magnetic field values.
SBG_CAN_ID_OUTPUT_TEMPERATURES_RAW (0x12)	Raw internal device temperatures.

CAN message name with default id	Description
SBG_CAN_ID_OUTPUT_GYRO_TEMPERATURES_RAW (0x13)	Raw internal gyroscopes temperatures.
SBG_CAN_ID_OUTPUT_BAROMETER (0x14)	Barometric pressure and altitude.
SBG_CAN_ID_OUTPUT_MAG_CALIB_DATA (0x15)	Magnetometers data used to calibrate magnetic field.
SBG_CAN_ID_OUTPUT_ODOMETER_VELOCITIES (0x16)	Odometer channel 0 and 1 raw velocities.
SBG_CAN_ID_OUTPUT_GPS_INFO (0x17)	Raw GPS information such as fix quality.
SBG_CAN_ID_OUTPUT_GPS_SVINFO (0x18)	GPS space vehicles detailed information.
SBG_CAN_ID_OUTPUT_GPS_POSITION_1 (0x19)	Raw GPS latitude and longitude in WGS84 format.
SBG_CAN_ID_OUTPUT_GPS_POSITION_2 (0x1A)	Raw GPS altitude and horizontal/vertical accuracies.
SBG_CAN_ID_OUTPUT_GPS_VELOCITY_1 (0x1B)	Raw GPS x and y velocity in NED format.
SBG_CAN_ID_OUTPUT_GPS_VELOCITY_2 (0x1C)	Raw GPS z velocity in NED Format and speed accuracy.
SBG_CAN_ID_OUTPUT_GPS_COURSE (0x1D)	Raw GPS course and course accuracy.
SBG_CAN_ID_OUTPUT_GPS_TRUE_HEADING (0x1E)	Raw GPS true heading and accuracy.
SBG_CAN_ID_OUTPUT_DELTA_ANGLES (0x1F)	1000Hz coning integration output
SBG_CAN_ID_OUTPUT_HEAVE (0x20)	Heave output

Table 2: Available output frames with their default CAN id

1.5.3. Input frames

The device can accept specifics frames in order to get some external aiding information. These aiding data will be fused by the internal Kalman with inertial sensors enabling advance inertial aided solutions.

The following aiding data can be provided to the device:

- External true heading data,
- External velocity in device coordinate frame,
- External NED position in WGS84 format.

When an input frame is received by the device, it does not sent back any answer or acknowledge in order to avoid bus overloading.

Please find below all available aiding input messages with their default CAN id.

CAN message name with default id	Description
SBG_CAN_ID_SEND_FILTER_HEADING (0x62)	Input external heading with its accuracy.
SBG_CAN_ID_SEND_NAV_VELOCITY_LOCAL_1 (0x63)	Input external x and y velocities in the device frame.
SBG_CAN_ID_SEND_NAV_VELOCITY_LOCAL_2 (0x64)	Input external z velocity and velocity accuracy.
SBG_CAN_ID_SEND_NAV_VELOCITY_NED_1 (0x65)	Input external North and East velocities in NED frame.
SBG_CAN_ID_SEND_NAV_VELOCITY_NED_2 (0x66)	Input external Down velocity and velocity accuracy.
SBG_CAN_ID_SEND_NAV_POSITION_1 (0x67)	Input external latitude and longitude position.
SBG_CAN_ID_SEND_NAV_POSITION_2 (0x68)	Input external altitude and position accuracy.

Table 3: Aiding frames with their corresponding id

1.6. Types definition

1.6.1. Scalar types

Type	Description
`bool`	8 bits boolean. 0x00 is FALSE. 0X01 is TRUE
`uint8`	8 bits unsigned int
`int8`	8 bits signed int
`uint16`	16 bits unsigned int
`int16`	16 bits signed int
`uint32`	32 bits unsigned int
`int32`	32 bits signed int
`uint40`	40 bits unsigned int
`int40`	40 bits signed int
`uint64`	64 bits unsigned int
`int64`	64 bits signed int
`frac16`	16 bits fractional number : 1 sign bit; 15 bits for fractional part
`fixed32`	32 bits fixed point number : 1 sign bit; 11 bits for integer part, 20 bits for fractional part
`fixed64`	64 bits fixed point number : 1 sign bit; 31 bits for integer part, 32 bits for fractional part
`float`	32 bits single floating point, standard IEEE 754 format
`double`	64 bits single floating point, standard IEEE 754 format

1.6.2. Vectors and matrices

Vectors are stored in a 1D array.

$$V=\begin{pmatrix}V_0\\V_1\\V_2\end{pmatrix}$$

Vector V is stored in memory this way :

V_0	V_1	V_2

All matrices are stored in a 1D array. They are expressed in vector column format.
For example, with a $M_{3\times 3}$ matrix :

$$M=\begin{pmatrix}U_0 & V_0 & W_0\\U_1 & V_1 & W_1\\U_2 & V_2 & W_2\end{pmatrix}$$

M is stored in memory this way:

U_0	U_1	U_2	V_1	V_2	V_3	W_0	W_1	W_2

2. Configuration frames

2.1. General settings

SBG_CAN_ID_SAVE_SETTINGS (0x2F)

This command is used to save current device settings permanently.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	yes	yes	yes

To save the whole current settings into the device non volatile memory, the user should send this command with an empty payload.

The device should return, with the same frame, the operation result as parameter.

Possible values for the status field are:

- `SBG_CAN_OPERATION_SUCCESS      0x00        Operation could be properly performed`
- `SBG_CAN_OPERATION_FAILURE      0x01        Operation could not be performed correctly`

Field	ID	LEN	DATA
Value	0x2F	0x01	Status (uint8)
Size (bytes)	12 or 32 bits	6 bits	1

Note: *If Status returns SBG_SAVE_RESTORE FAILURE please make sure that the save setting command is sent without any payload.*

SBG_CAN_ID_RESTORE_SETTINGS (0x30)

This command restores all current settings to factory default. Magnetometers and gyroscopes calibration data are not affected by this command.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	yes	yes	yes

To restore default setting, the user should send this command with an empty payload.

The device should first return, the operation result as parameter using the same frame and the current communication settings. After all settings have been reset, the device will use default communication settings (1 Mbit/s, standard ids, default ids, …)

Possible values for the status field are:

• `SBG_CAN_OPERATION_SUCCESS`	`0x00`	`Operation could be properly performed`
• `SBG_CAN_OPERATION_FAILURE`	`0x01`	`Operation could not be performed correctly`

Field	**ID**	**LEN**	**DATA**
Value	0x30	0x01	Status (uint8)
Size (bytes)	12 or 32 bits	6 bits	1

> ***Note:*** *After settings are reverted to defaults, the configuration is directly saved into FLASH memory, and the device performs a complete reboot.*

SBG_CAN_ID_LOW_POWER_MODE (0x31)

This command is used to set or get the GPS power mode. The user can turn on or off the GPS receiver.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	yes	yes	yes

The command below is used to set the new power mode. If the new power mode has been changed, the device should return the exact same frame as the sent one.

Possible values for Imu Power are:

```
SBG_IG_MAX_PERF     (0x00)       Max performance mode. Reduces latency,
                                 allows heave computation
SBG_IG_NORMAL_MODE  (0x02)       Normal mode; Limited power consumption
                                 but we cannot get heave computation on this mode
```

Possible values for GPS Power are:

```
SBG_GPS_MAX_PERF    (0x00)       Max performance mode
SBG_GPS_ECO_MODE_1  (0x01)       Current peaks are limited. startup might be longer
                                 than in MAX_PERF mode. (Only supported with
                                 GPS board hardware V.3 and above)
SBG_GPS_ECO_MODE_2  (0x02)       Current consumption is reduced when GPS has a good fix.
                                 (Only supported with GPS board hardware V.3 and above)
SBG_GPS_OFF_MODE    (0x05)       GPS is turned off
```

Field	ID	LEN	DATA	
Value	0x31	0x02	IMU power. Set to 0x02	GPS power (uint8)
Size (bytes)	12 or 32 bits	6 bits	1	1

To retrieve the GPS power mode, the user should send the frame with an empty payload. The device should return the command above.

> ***Note:*** *After a power mode change, the current configuration is directly saved into FLASH memory, and the device performs a complete reboot.*

SBG_CAN_ID_DEVICE_INFO (0x32)

This command is used to retrieve device information such as the product code, device unique serial number, firmware and hardware revisions.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	yes	yes	yes

To retrieve the device information, the user should send this command with an empty payload.

The device should return thirteen frames: eight for the product code (string of 32 chars), one for the device number (uint32), one for the firmware revision (uint32), one for the main board hardware revision (uint32) and one for the GPS/Top board hardware revision (uint32).

All revisions are sent using the format describe below. The example encodes the version 1.2.5.61

Field	Major	Minor	Revision	Build
Size	uint8	uint8	uint8	uint8
Example	0x01	0x02	0x05	0x3D

Each received frame contains an index used to check which frame is being received. All frames have the exact same size of 5 bytes, a unit8 for the index and a 4 bytes for the payload.

You will find below a detailed description of each 13 frames sent by the device.

The first 8 frames are used to send the 32 bytes product code:

Field	ID	LEN	DATA	
Value	0x32	0x05	Response index (uint8) 0 to 7	Product code (0-7) (string)
Size (bytes)	12 or 32 bits	6 bits	1	4

After that, the device returns its unique serial number:

Field	ID	LEN	DATA	
Value	0x32	0x05	Response index (uint8) 8	Device number (uint32)
Size (bytes)	12 or 32 bits	6 bits	1	4

Command used to return the device firmware revision:

Field	ID	LEN	DATA	
Value	0x32	0x05	Response index (uint8) 9	Firmware revision (uint32)
Size (bytes)	12 or 32 bits	6 bits	1	4

Command used to return the calibration data revision:

Field	ID	LEN	DATA	
Value	0x32	0x05	Response index (uint8) 10	Calibration data revision (uint32)
Size (bytes)	12 or 32 bits	6 bits	1	4

Command used to return the main board hardware revision:

Field	ID	LEN	DATA	
Value	0x32	0x05	Response index (uint8) 11	Main board revision (uint32)
Size (bytes)	12 or 32 bits	6 bits	1	4

Command used to retrieve the GPS/Top board revision:

Field	ID	LEN	DATA	
Value	0x32	0x05	Response index (uint8) 12	Top board revision (uint32)
Size (bytes)	12 or 32 bits	6 bits	1	4

SBG_CAN_ID_USER_ID (0x33)

This command is used to set or get a custom id for the device in order to easily identify it.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	yes	yes	yes

The command below is used to set a custom id. If the new id mode has been changed, the device should return the exact same frame as the sent one.

Field	ID	LEN	DATA
Value	0x33	0x04	User id (uint32)
Size (bytes)	12 or 32 bits	6 bits	4

To retrieve the device custom id, the user should send this frame with an empty payload. The device should return the command above.

SBG_CAN_ID_USER_BUFFER (0x34)

This command allows, as a complement to the device user id, having a read/write access to a dedicated user space memory in the device.

This 64 bytes long buffer can be saved to the flash non volatile memory using SBG_SAVE_SETTINGS command. This user buffer could be represented as an array of 16 uint32 that can be accessed using an index.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	yes	yes	yes

To write data into the buffer, the user should send the command below with the desired index and the value to be written. If the data has been written, the device should return the exact same command as the sent one.

Field	ID	LEN	DATA	
Value	0x34	0x05	Index (uint8)	Data to be written (uint32)
Size (bytes)	12 or 32 bits	6 bits	1	4

To retrieve a value previously written into the user buffer, the user should send this frame with the value index to retrieve.

Field	ID	LEN	DATA
Value	0x34	0x01	Index (uint8)
Size (bytes)	12 or 32 bits	6 bits	1

2.2. Output settings

SBG_CAN_ID_OUTPUT_TRIGGERS_CONF (0x35)

This command is used to configure and retrieve, for each output frame, its trigger mask. The trigger mask defines how an output frame will be sent by the device.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	yes	yes	yes

The following triggers can generate an output. The triggers can be combined with a bitwise OR:

```
SBG_TRIGGER_MAIN_LOOP_DIVIDER    0x0001      Trigger on the main loop frequency divider
SBG_TRIGGER_MAGNETOMETERS        0x0002      Trigger on a new magnetometers data
SBG_TRIGGER_BAROMETER            0x0004      Trigger on a new barometer data
SBG_TRIGGER_GPS_VELOCITY         0x0008      Trigger on a new GPS velocity
SBG_TRIGGER_GPS_POSITION         0x0010      Trigger on a new GPS position
SBG_TRIGGER_GPS_COURSE           0x0020      Trigger on a new GPS heading
SBG_TRIGGER_TIME_PULSE           0x0040      Trigger on an input time pulse
SBG_TRIGGER_EXT_EVENT            0x0080      Trigger on a new external event
SBG_TRIGGER_ODO_VELOCITY_0       0x0100      Trigger on a new odometer 0 velocity
SBG_TRIGGER_ODO_VELOCITY_1       0x0200      Trigger on a new odometer 1 velocity
SBG_TRIGGER_EXT_TRUE_HEADING     0x0400      Trigger on a new true heading
SBG_TRIGGER_VIRTUAL_ODOMETER     0x0800      Trigger when the virtual odometer reached
                                             desired distance
```

The following command is used to configure a trigger mask for a specific output. If the output trigger has been changed, the device should return the exact same command as the sent one. The output frame id field should be one of the id listed in the 2.

Field	ID	LEN	DATA	
Value	0x35	0x04	Output frame id (uint16)	Trigger mask (uint16)
Size (bytes)	12 or 32 bits	6 bits	2	2

The command below is used to retrieve the trigger configuration of a specific output. The user should send a frame that contains only the output CAN message id as listed in the 2 and get the command above.

Field	ID	LEN	DATA
Value	0x35	0x02	Output frame id (uint16)
Size (bytes)	12 or 32 bits	6 bits	2

SBG_CAN_ID_OUTPUT_MAIN_LOOP_DIVIDER (0x36)

Thanks to this command, it's possible to define at which rate an output frames with the `SBG_TRIGGER_MAIN_LOOP_DIVIDER` trigger enabled, will be outputted.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	yes	yes	yes

The Kalman filter frequency, defined within the motion profile (or user defined on IG-20/30), is divided by the main loop divider. For example, if the Kalman filter frequency is 100 Hz and the main loop divider has been set to 4 then all output frames, triggered on the main loop divider, will be outputted at 25 Hz.

The following formula could be used to compute the output rate:

$F_{out}=\frac{F_{kal}}{divider}$ where F_{kal} is the Kalman filter frequency.

The command below is used to set the new desired filter frequency divider. If the new divider has been changed, the device should return the exact same frame as the sent one.

Note: *0 Value is an invalid parameter.*

Field	ID	LEN	DATA
Value	0x36	0x01	Frequency divider (uint8)
Size (bytes)	12 or 32 bits	6 bits	1

To retrieve the current frequency divider, the user should send the frame with an empty payload. The device should return the command above.

2.3. Protocol settings

SBG_CAN_ID_PROTOCOL_MODE (0x37)

This command is used to set or get the device CAN bus configuration such as the bus speed. The CAN bus implementation and especially timing settings respects the CAN in Automation (CiA) DS-102 standard.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	yes	yes	yes

The command below is used to set the CAN bus baud rate according to the CiA DS-102 standard. If the new bit rate has been changed successfully, the device should return the exact same command as the sent one using the current bit rate.

Possible values for bit rate are:

- 1000 kBit/s
- 500 kBit/s
- 250 kBit/s
- 125 kBit/s
- 100 kBit/s
- 50 kBit/s
- 20 kBit/s
- 10 kBit/s

Field	ID	LEN	DATA	
Value	0x37	0x08	Bit rate (uint16)	Reserved 6*(uint8)
Size (bytes)	12 or 32 bits	6 bits	2	6

To retrieve the CAN bus speed, the user should send this frame with an empty payload. The device should return the command above.

> ***Note 1 :*** *When changing the protocol mode, the acknowledge is transmitted using the current speed. The new setting such as the bit rate is effectively applied just after.*

SBG_CAN_ID_FRAME_ID (0x38)

This command allows to set or get which CAN id should be used for a specific CAN frame. It's also possible to define if the CAN message is a standard or an extended one and to disable completely a CAN message.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	yes	yes	yes

The following command is used to set which CAN id will used for a specific CAN message. If the new CAN id has been set successfully, the device should return the exact same command as the sent one.

The default frame id field should be one of the id listed in the 1, 2 or 3. To completely disable a CAN frame, just set the new frame id field to the following value:

```
SBG_DISABLED_FRAME  0xF0000000
```

Possible values for message type are:

- `SBG_STANDARD_ID   0x00`
- `SBG_EXTENDED_ID   0x01`

Field	ID	LEN	DATA		
Value	0x38	0x07	Default frame id (uint16)	Message type (uint8)	New frame id (uint32)
Size (bytes)	12 or 32 bits	6 bits	2	1	4

The command below is used to retrieve the used CAN frame id and type (standard or extended) for a specific CAN message. The user should send a command that contains only the default CAN message id as listed in one of the 1, 2 or 3 and get the command above.

Field	ID	LEN	DATA
Value	0x38	0x02	Default frame id (uint16)
Size (bytes)	12 or 32 bits	6 bits	2

***Note about IDs optimal selection:** The IG-500 CAN controller can filter unwanted incoming CAN messages without any CPU intervention. Especially when multiple devices are operating on the same CAN bus, it can be important to set optimal values for the messages identifiers. Improper ID selection might lead to lower performance or incorrect behavior due to high CPU usage.*

The CAN controller uses two registers to check if a message is accepted or not: A message ID is internally set to define a message base, and a mask, where all bits set to 1 defines a constant bit that is always present in all handled Ids, and a bit left to 0 represent variable bit.

So the best filtering strategy is to define a fixed ID base that is common to all IG-500 messages, and a variable part to handle each message. Other talking devices should set their base message to another value in order to filter out all unwanted messages. The more common bits there is on the enabled frames, the best filtering can the controller do.

*For example, All messages could be set to 0x22FF88** , where all bits are constant, except the "**" which are the variable bits.*

The worse ID selection would be if there was no common bit to each identifier. Then the CAN controller would have to accept all incoming messages and the CPU would perform all filtering labor work.

***Note 2:** Based on all configured and enabled frames, the IG-500 automatically configures the best filtering strategy.*

2.4. *Kalman filter settings*

2.4.1. Motion profile setting

Motion profile setting is allowed in a configuration level, but should not be performed in live, because updating a motion profile requires a device reset.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	no	no	yes	yes	yes

SBG_CAN_ID_SEND_MP_BUFFER (0x52)

This sends a part of the motion profile buffer into the IG-Device:

The index where to place the data in the internal motion profile is specified. The buffer size is defined directly by the frame size. Send the following frame to transmit a part of a motion profile buffer:

Field	ID	LEN	DATA	
Value	0x52	0x02 + X	index (uint16)	Raw motion profile buffer
Size (bytes)	12 or 32 bits	6 bits	2	X

SBG_CAN_ID_VALIDATE_MP_BUFFER (0x53)

This command has no parameter and is used to validate the buffer once it is fully sent.

The sensor should answer with the same frame ID with 1 byte payload corresponding to the validation status:

Possible values for the status field are:

- SBG_CAN_OPERATION_SUCCESS	0x00	Operation could be properly performed
- SBG_CAN_OPERATION_FAILURE	0x01	Operation could not be performed correctly

Field	ID	LEN	DATA
Value	0x53	0x01	status (uint8)
Size (bytes)	12 or 32 bits	6 bits	1

The following actions are performed at reception of this frame:

1. Check that the buffer is consistent and compatible with current hardware / software
2. Send an Acknowledge with corresponding result
3. In case of good configuration
   1. Save all settings to flash memory
   2. Reboot the device

> ***Note:*** *As a reboot is initiated if the buffer Is valid, a long latency should be expected (< 2s) after the ACK is received.*

SBG_CAN_ID_MP_INFO (0x54)

This command has no parameter and is used to retrieve the current motion profile info: Unique identifier, as well as version number.

The device should answer with a SBG_MP_INFO, with a 8 bytes payload as follows:

Field	ID	LEN	DATA	
Value	0x54	0x08	Motion profile ID (uint32)	Motion profile revision (uint32)
Size (bytes)	12 or 32 bits	6 bits	4	4

2.4.2. Heading configurations

SBG_CAN_ID_FILTER_HEADING_SOURCE (0x3C)

This command is used to set the heading source used by the Kalman filter as a reference.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	yes	yes	yes	yes	yes

Possible values for the heading source are:

• `SBG_HEADING_SOURCE_NONE`	`0x00`	Not available for IG-500N
• `SBG_HEADING_SOURCE_MAGNETOMETERS`	`0x01`	Magnetometers used as heading (default)
• `SBG_HEADING_SOURCE_GPS_COURSE`	`0x02`	GPS course used as heading
• `SBG_HEADING_SOURCE_GPS_ACCELERATIONS`	`0x03`	GPS + accelerometer heading source
• `SBG_HEADING_SOURCE_USER`	`0x05`	User inputs heading reference via main protocol
• `SBG_HEADING_SOURCE_REMOTE_MAG`	`0x06`	Remote magnetometers are used as heading
• `SBG_HEADING_SOURCE_REMOTE_TRUE_HEADING`	`0x07`	Remote true heading used (dual antenna)

Send the command below to configure which heading source should be used by the Kalman filter. If the new setting has been applied successfully, the device should return the exact same frame as the sent one.

Field	ID	LEN	DATA
Value	0x3C	0x08	Heading source (uint8)
Size (bytes)	12 or 32 bits	6 bits	1

To retrieve the heading source used by the Kalman filter, the user should send this frame with an empty payload. The device should return the command above.

SBG_CAN_ID_MAGNETIC_DECLINATION (0x3D)

This command is used to define or get the local north magnetic declination. For IG-500N and IG-500E devices, when magnetometers are used as heading reference, this information is very important to ensure good attitude and navigation results.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	yes	yes	yes	yes	yes

The valid magnetic declination range is between $[-\pi, +\pi]$ and is positive toward east and negative toward west.

Field	ID	LEN	DATA
Value	0x3D	0x04	declination (fixed32)
Unit	-	-	rad
Size (bytes)	12 or 32 bits	6 bits	4

To retrieve the magnetic declination used by our Kalman filter, the user should send this frame with an empty payload. The device should return the command above.

2.4.3. Heave configuration

SBG_CAN_ID_HEAVE_CONF (0x55)

This function is used to configure the heave operation.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	no	no	yes	yes	yes

If enabled, the output latency should be slightly increased due to added computations.

Send the command below to configure which heave operation. If the new setting has been applied successfully, the device should return the exact same frame as the sent one.

Heave period should be set between 0.3 and 20.0.

Field	ID	LEN	DATA		
Value	0x55	0x08	Enable Heave (bool)	Heave period (s) (fixed32)	Reserved – padding; Leave to 0
Size (bytes)	12 or 32 bits	6 bits	1	4	3

To retrieve the heading source used by the Kalman filter, the user should send this frame with an empty payload. The device should return the command above.

2.5. Navigation settings

On IG-500N and IG-500E, it is possible to configure the input source for both position and velocity aiding data. The embedded Kalman Filter will use these aiding data to correct for position and velocity drifts that come from accelerometers integration.

The pressure sensor can also be configured.

SBG_CAN_ID_REFERENCE_PRESSURE (0x3E)

This command is used to set/get the reference pressure used for barometric altitude calculations. By default, the reference pressure is set to 1 013.25 hPa.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	no	yes	no	yes	yes

Send the command below to configure the reference pressure. If the reference pressure is set to 0, the device will read the current pressure and use it as the reference one. If the new setting has been applied successfully, the device should return the exact same frame as the sent one.

Field	ID	LEN	DATA
Value	0x3E	0x04	Reference Pressure (uint32)
Units	-	-	Pa
Size (bytes)	12 or 32 bits	6 bits	4

To retrieve the reference pressure used for altitude calculation, the user should send this frame with an empty payload. The device should return the command above.

SBG_CAN_ID_NAV_SOURCES (0x40)

This command is used to configure/retrieve both the position and velocity measurement source to be used as aiding data by the Kalman filter.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	no	no	no	yes	yes

Possible values for position source are:

• `SBG_POS_SRC_GPS`	`0x00`	Position corrected using GPS data
• `SBG_POS_SRC_GPS_AND_BARO`	`0x01`	Position corrected using GPS and barometer (default)
• `SBG_POS_SRC_USER`	`0x02`	Position corrected using user provided data

Possible values for velocity source are:

• `SBG_VEL_SRC_GPS`	`0x00`	Velocity corrected using GPS data (default)
• `SBG_VEL_SRC_USER`	`0x02`	Velocity corrected using user provided data
• `SBG_VEL_SRC_ODO`	`0x03`	Velocity corrected using odometer data (IG-500E only)

Send the command below to configure both position and velocity aiding source. If the new setting have been applied successfully, the device should return the exact same frame as the sent one.

Field	ID	LEN	DATA	
Value	0x40	0x02	Position source (uint8)	Velocity source (uint8)
Size (bytes)	12 or 32 bits	6 bits	1	1

To retrieve the position and velocity aiding sources, the user should send this frame with an empty payload. The device should return the command above.

SBG_CAN_ID_GPS_LEVER_ARM (0x41)

This command is used to specify/retrieve the vector from the device to the GPS antenna. The lever arm is expressed in meters the device coordinate frame. For more information, please read the User Manual documentation.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	no	no	no	yes	yes

Send the command below to configure the lever arm between the device and the GPS antenna. The lever arm is expressed in millimeters, in the device coordinate frame. If the new setting have been applied successfully, the device should return the exact same frame as the sent one.

Field	ID	LEN	DATA		
Value	0x41	0x06	Lever arm X (int16)	Lever arm Y (int16)	Lever arm Z (int16)
Unit	-	-	mm	mm	mm
Size (bytes)	12 or 32 bits	6 bits	2	2	2

To retrieve the GPS lever arm, the user should send this frame with an empty payload. The device should return the command above.

SBG_CAN_ID_GRAVITY_MAGNITUDE (0x42)

This command is used to set/get the local earth gravity magnitude used for navigation computation. An invalid gravity magnitude can lead to significant errors in velocity and position calculation for IG-500N and IG-500E devices.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	no	no	yes	yes	yes

The default gravity magnitude value is set to 9.809 $m\ .\ s^{-2}$.

Send the command below to configure the local earth gravity magnitude. If the new setting has been applied successfully, the device should return the exact same frame as the sent one.

Field	ID	LEN	DATA
Value	0x42	0x04	Gravity magnitude (fixed32)
Unit	-	-	m .s^{-2}
Size (bytes)	12 or 32 bits	6 bits	4

To retrieve the earth local gravity magnitude, the user should send this frame with an empty payload. The device should return the command above.

2.6. Coordinate frame transformation settings

The following commands are used to make sensor transformations, automatically or manually, directly on sensors outputs, or on the orientation output.

Two types of alignment can be done, a pre-alignment, the whole device frame including sensors outputs are aligned with the new coordinate system or a post-alignment, only the calculated orientation will be rotated.

For IG-500N and IG-500E devices, only the pre-alignment options are valid. Indeed, for correct navigation computations, the IG-500N and IG-500E need a consistent information between calculated outputs and sensors measurements.

SBG_CAN_ID_AUTO_ORIENTATION_OFFSET (0x44)

This command automatically align the device to the coordinate frame's axis (Z, X and Y, or X Y and Z).

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	yes	yes	yes

Possible values for offset type are:

• `SBG_OFFSET_PRE_ROT_Z_RESET`	`0x03`	Calculate a pre-rotation on Z axis
• `SBG_OFFSET_PRE_ROT_XY_RESET`	`0x04`	Calculate a pre-rotation on XY axis
• `SBG_OFFSET_PRE_ROT_XYZ_RESET`	`0x05`	Calculate a pre-rotation on XYZ axis
• `SBG_OFFSET_POST_ROT_Z_RESET`	`0x06`	Calculate a post-rotation on Z axis
• `SBG_OFFSET_POST_ROT_XY_RESET`	`0x07`	Calculate a post-rotation on XY axis
• `SBG_OFFSET_POST_ROT_XYZ_RESET`	`0x08`	Calculate a post-rotation on XYZ axis

Send the command below to initiate an orientation offset computation. If the device has successfully computed an offset matrix, it should should return the exact same frame as the sent one.

Field	**ID**	**LEN**	**DATA**
Value	0x44	0x01	Offset type (uint8)
Size (bytes)	12 or 32 bits	6 bits	1

The orientation offset cannot be retrieved using this command. To do it, please use these two commands, SBG_CAN_ID_PRE_ORIENTATION_OFFSET (0x45) and SBG_CAN_ID_POST_ORIENTATION_OFFSET (0x46).

Note: *After a pre-rotation reset is initiated, the current configuration is directly saved into FLASH memory, and the device performs a complete reboot.*

SBG_CAN_ID_PRE_ORIENTATION_OFFSET (0x45)

This command allows to set/get manually the rotation to be applied on sensors input using a pre-rotation matrix.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	yes	yes	yes

The rotation is defined with a normalized quaternion (w, x, y, z) expressed using frac16 numbers as described in the paragraph 1.6.1Scalar types.

Send the command below to set the pre-rotation offset matrix. If the device has successfully defined the new offset matrix, it should return the exact same frame as the sent one.

Field	ID	LEN	DATA			
Value	0x45	0x08	Quat w (frac16)	Quat x (frac16)	Quat y (frac16)	Quat z (frac16)
Size (bytes)	12 or 32 bits	6 bits	2	2	2	2

To retrieve the current pre-rotation matrix, the user should send this frame with an empty payload. The device should return the command above.

> ***Note:*** *After a pre-rotation reset is initiated, the current configuration is directly saved into FLASH memory, and the device performs a complete reboot.*

SBG_CAN_ID_POST_ORIENTATION_OFFSET (0x46)

This command allows to configure manually the rotation to be applied on orientation output using a post-rotation matrix.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	yes	no	no

The rotation is defined with a normalized quaternion (w, x, y, z) expressed using frac16 numbers as described in the paragraph 1.6.1Scalar types.

Send the command below to set the post-rotation offset matrix. If the device has successfully defined the new offset matrix, it should return the exact same frame as the sent one.

Field	ID	LEN	DATA			
Value	0x46	0x08	Quat w (frac16)	Quat x (frac16)	Quat y (frac16)	Quat z (frac16)
Size (bytes)	12 or 32 bits	6 bits	2	2	2	2

To retrieve the current post-rotation matrix, the user should send this frame with an empty payload. The device should return the command above.

2.7. Magnetometers and gyroscopes calibration

SBG_CAN_ID_CALIB_MAG (0x47)

This command allows to load default magnetometers calibration and save current settings permanently.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	yes	yes	yes	yes	yes

Possible values for argument are:

• `SBG_CAN_CALIB_LOAD_DEFAULT`	`0x00`	Load factory default magnetometers calibration
• `SBG_CAN_CALIB_SAVE`	`0x05`	Save the current used calibration into flash memory

Possible values for error are:

• `SBG_CAN_CALIB_WRONG_PARAMETER`	`0x09`	Calibration argument is invalid

Send the command below to execute a magnetic calibration operation. If the device has successfully executed the command, the returned frame should be exactly the same as the sent one. In case of an error, the returned frame will contain an error code detailed in the table below.

Field	ID	LEN	DATA
Value	0x47	0x01	Argument / error (uint8)
Size (bytes)	12 or 32 bits	6 bits	1

SBG_CAN_ID_CALIB_MAG_MANUAL (0x48)

The manual magnetometers calibration procedure allows to manually send both soft and hard iron compensation coefficients directly to the device. It may be useful if an external calibration procedure is used.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	yes	yes	yes	yes	yes

The calibrated magnetic field data is calculated using the following formula:

$$M_{Cal}=\begin{pmatrix} AGm_0 & AGm_3 & AGm_6 \\ AGm_1 & AGm_4 & AGm_7 \\ AGm_2 & AGm_5 & AGm_8 \end{pmatrix} \times \left[\begin{pmatrix} M_X \\ M_Y \\ M_Z \end{pmatrix} - \begin{pmatrix} Bias_X \\ Bias_Y \\ Bias_Z \end{pmatrix} \right]$$

Where AGm is the alignment and gain matrix and Bias is the magnetic field offset vector.

To define both the bias vector and the alignment and gain matrix, the user has to send twelve frames. The first three frames are used to send the bias vector and the nine last ones are used for the alignment and gain matrix.

Each sent frame should contain an index used to identify which frame is being sent and a fixed32 that stores the component currently transferred. For each sent frame, the device should return an acknowledge by sending the exact same frame.

First, send the following frame three times (x, y, z) to set the Bias vector:

Field	ID	LEN	DATA	
Value	0x48	0x05	Index (0 to 2) (uint8)	$\text{Bias}_{\text{Index}}$ (fixed32)
Size (bytes)	12 or 32 bits	6 bits	1	4

Send the following frame nine times for each matrix component:

Field	ID	LEN	DATA	
Value	0x48	0x05	Index (3 to 11) (uint8) 1	$\text{AGm}_{\text{Index-3}}$ component (fixed32)
Size (bytes)	12 or 32 bits	6 bits	1	4

To retrieve the magnetometers calibration parameters, just send a frame with an empty payload. The device should return twelve frames as describe above.

Note: *If you would like to keep the current magnetometer calibration for the next start-up, don't forget to use the command SBG_CAN_ID_CALIB_MAG (0x47) with `SBG_CAN_CALIB_SAVE` argument to save it.*

SBG_CAN_ID_CALIB_GYRO_BIAS (0x49)

This command allows to save the current gyroscopes value as a bias. Three options are available for bias estimation: coarse, medium and fine which will respectively measure gyroscopes bias for 0.25s, 1s, and 3s.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	no	no	yes	yes	yes

Possible values for argument are:

• `SBG_CAN_CALIB_LOAD_DEFAULT`	`0x00`	Load factory default gyroscopes bias
• `SBG_CAN_CALIB_MEASURE_COARSE`	`0x04`	Acquire gyroscopes bias for 0.25 second
• `SBG_CAN_CALIB_SAVE`	`0x05`	Save the current gyroscopes bias into flash memory
• `SBG_CAN_CALIB_MEASURE_MEDIUM`	`0x06`	Acquire gyroscopes bias for 1 second
• `SBG_CAN_CALIB_MEASURE_FINE`	`0x07`	Acquire gyroscopes bias for 3 seconds (best accuracy)

Send the command below to execute a gyroscopes bias estimation operation. If the device has successfully executed the command, the returned frame should be exactly the same as the sent one.

Field	ID	LEN	DATA
Value	0x49	0x01	Argument (uint8)
Size (bytes)	12 or 32 bits	6 bits	1

Note 1: *If you would like to use the current estimated bias for the next start-up, don't forget to send the command with `SBG_CAN_CALIB_SAVE` argument to save it in non volatile flash memory.*

Note 2: *If configured properly, the Kalman filter continuously tracks the gyroscopes bias. To reduce bias estimation at the next start-up, you can save the gyroscopes bias estimated by the Kalman filter at any time into non volatile flash memory by sending the command with `SBG_CAN_CALIB_SAVE` argument.*

2.8. External device settings

SBG_CAN_ID_EXT_DEVICE (0x4A)

This command is used to set/get the external material configuration such as the external port baud rate and the external material type connected to the IG-500E.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	no	no	no	no	yes

Possible values for device type are:

`SBG_EXT_NONE`	`0x00`	No device is connected to the external port
`SBG_EXT_IG_DEVICE`	`0x03`	An other IG-Device is connected to the external port
`SBG_EXT_NMEA`	`0x04`	A NMEA device is connected to the external port

Possible values for UART options are:

`SBG_EXT_PORT_RS232`	`0x00`	Set the external port in RS-232 mode (default)
`SBG_EXT_PORT_RS422`	`0x01`	Set the external port in RS-422 mode
`SBG_EXT_PORT_FAST_SLEW`	`0x00`	Fast slew rate, baudrate is not limited. (default)
`SBG_EXT_PORT_SLOW_SLEW`	`0x02`	Slow slew rate for EMI reduction. Max baudrate: 230 400 bps.

Possible values for baud rate are:

- 921 600 bps
- 460 800 bps
- 230 400 bps
- 57 600 bps
- 38 400 bps
- 19 200 bps
- 9 600 bps (default)

***Note:** A device reboot is necessary to take the UART port options into account.*

Send the command below to define the external device configuration. If the new settings have been applied successfully, the device should return the exact same frame as the sent one.

Field	ID	LEN	DATA		
Value	0x4A	0x07	Device type (uint8)	Baud rate (uint32)	UART options (uint16)
Size (bytes)	12 or 32 bits	6 bits	1	4	2

To retrieve the external device configuration, the user should send this frame with an empty payload. The device should return the command above.

SBG_CAN_ID_EXT_DEVICE_CONF (0x4B)

This command is used to set/get settings to the external device module. In other words, when an external device type is chosen using the command SBG_CAN_ID_EXT_DEVICE (0x4A), some settings, specific to that external device type, become available.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	no	no	no	no	yes

To set/get settings to an external device module, a raw buffer has to be sent and received by the device. The external device module will extract, from the received buffer, a specific command. To return an answer, the external device module will encapsulate the answer in a buffer and send it back using the same mechanism.

We will only describe how to send and receive these buffers. For more details about specifics commands for external device module, please refer to external device user manuals.

Sending a raw buffer of up to 500 bytes long is a four steps operation as describe below:

1) *Start the buffer transmission by sending its size*

To initiate a buffer transmission set the control byte to `SBG_CAN_EXT_DEVICE_CONF_SIZE (0xFE)` with the buffer size to transmit as describe in the command below:

Field	ID	LEN	DATA	
Value	0x4B	0x03	Control byte (0xFE) (uint8)	Payload size in bytes (uint16)
Size (bytes)	12 or 32 bits	6 bits	1	2

2) *Send the buffer itself after splitting it*

To send the buffer itself, it has to be split into blocks of 7 bytes. Only the last block could be smaller than 7 bytes if needed. Send each block in the correct order using the command below and don't forget to specify the correct block index for each sent frame.

Field	ID	LEN	DATA	
Value	0x4B	0x01 + n	Block index (uint8)	Payload data n * (uint8)
Size (bytes)	12 or 32 bits	6 bits	1	n

3) *Conclude the buffer transmission*

After all block have been sent, to validate the buffer transmission, just send the command below with the special control byte SBG_CAN_EXT_DEVICE_CONF_END (0xFF).

Field	ID	LEN	DATA
Value	0x4B	0x01	Control byte (0xFF) (uint8)
Size (bytes)	12 or 32 bits	6 bits	1

4) *Wait for the answer*

If the buffer has been successfully transmitted and received by the external device module, the device should return an answer using the same mechanism as describe above.

Note: *Don't hesitate to have a look at the sbgCan implementation of this command to better understand it.*

Note2: *In case of error during transmission, an empty buffer can be sent back to indicate the problem.*

2.9. Odometers configurations

Three sets of commands are provided to configure the odometer management by the IG-500E.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	no	no	no	no	yes

The different parameters are:

- Odometer channel: may be 0 or 1.
- Odometer sensitive axis:

```
SBG_ODO_X                 0x00         Odometer sensitive axis is X
SBG_ODO_Y                 0x01         Odometer sensitive axis is Y
SBG_ODO_Z                 0x02         Odometer sensitive axis is Z
```

- Odometer direction:

```
SBG_ODO_DIR_POSITIVE      0x00         Odometer velocity is positive
SBG_ODO_DIR_NEGATIVE      0x01         Odometer velocity is negative
```

- Odometer pulses per meter. This parameter corresponds to the odometer gain.
- Odometer Lever Arm with respect to the sensor.
- Gain Options, as defined below:

Bit 7	Bit [6-0]
1 → Enable automatic gain correction by GPS 0 → Disable automatic gain correction by GPS	Gain error in %

Note: *In order to stop or inhibit the odometer operation, user should do the following operations:*

1) Disable the odometer from the Kalman filter velocity source

2) In case of long term odometer turn off, disable the Odometer logic input pin.

Note 2: *GPS automatic gain compensation is only available on a single odometer axis.*

SBG_CAN_ID_ODO_CONFIG (0x4C)

Send the command below to define for an odometer channel main configuration: Sensitive axis, gain, error and automatic GPS correction.

Field	ID	LEN	DATA			
Value	0x4C	0x07	Channel (uint8)	Axis (uint8)	Pulses per meter (fixed32)	GainOpt (uint8)
Size (bytes)	12 or 32 bits	6 bits	1	1	4	1

To retrieve the configuration for a specific odometer, the user should send the frame below. The device should return the command above.

Field	ID	LEN	DATA
Value	0x4C	0x01	Channel (uint8)
Size (bytes)	12 or 32 bits	6 bits	1

SBG_CAN_ID_ODO_DIRECTION (0x4D)

Send the command below to define for an odometer channel (0 or 1) its measurement direction. If the new setting has been applied successfully, the device should return the exact same frame as the sent one.

Field	ID	LEN	DATA	
Value	0x4D	0x02	Channel (uint8)	Measurement direction (uint8)
Size (bytes)	12 or 32 bits	6 bits	1	1

To retrieve the measurement direction for a specific odometer, the user should send the frame below. The device should return the command above.

Field	ID	LEN	DATA
Value	0x4D	0x01	Channel (uint8)
Size (bytes)	12 or 32 bits	6 bits	1

SBG_CAN_ID_ODO_LEVER_ARM (0x4E)

This command is used to set/get the lever arm between the device and a specific odometer. Two odometers can be configured so a channel field is used to define which odometer is being configured.

Send the command below to define the lever arm for an odometer channel (0 or 1). The lever arm is expressed in (signed) millimeters, in the device coordinate frame. If the new setting has been applied successfully, the device should return the exact same frame as the sent one.

Field	ID	LEN	DATA			
Value	0x4E	0x07	Channel (uint8)	Lever arm X (int16)	Lever arm Y (int16)	Lever arm Z (nit16)
Unit	-	-	-	mm	mm	mm
Size (bytes)	12 or 32 bits	6 bits	1	2	2	2

To retrieve the lever arm for a specific odometer, the user should send the frame below. The device should return the command above.

Field	ID	LEN	DATA
Value	0x4E	0x01	Channel (uint8)
Size (bytes)	12 or 32 bits	6 bits	1

2.10. Logic input and output options

SBG_CAN_ID_LOGIC_IN_CHANNEL (0x4F)

This command is used to set/get the configuration for a specific input channel. Two input channels can be configured so a channel field is used to define which input channel is being configured.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	no	yes	no	yes

Possible values for input type are:

• `SBG_IN_DISABLED`	`0x00`	This input channel is not used
• `SBG_IN_EVENT`	`0x01`	This input channel generate an event trigger
• `SBG_IN_TIME_PULSE`	`0x03`	This input channel represents a 4-PPS signal
• `SBG_IN_ODOMETER`	`0x05`	This input channel is used as an odometer input
• `SBG_IN_ODOMETER_DIRECTION`	`0x06`	Odometer direction sensing

Possible values for input sensitivity are:

• `SBG_IN_FALLING_EDGE`	`0x00`	An event is generated on a falling edge
• `SBG_IN_RISING_EDGE`	`0x01`	An event is generated on a rising edge
• `SBG_IN_LEVEL_CHANGE`	`0x02`	An event is generated either on a falling or rising edge

> ***Note:*** *Input sensitivity and delay are not used when the logic input is configured as odometer direction. In this case, only the input pin level is as general purpose input when a new odometer pulse is detected.*

Send the command below to define for an input channel (0 or 1) its type, sensitivity and delay in nanosecond. If the new settings have been applied successfully, the device should return the exact same frame as the sent one.

Field	ID	LEN	DATA			
Value	0x4F	0x07	Channel (uint8)	Input type (uint8)	Input sensitivity (uint8)	Delay ns (uint32)
Size (bytes)	12 or 32 bits	6 bits	1	1	1	4

To retrieve the configuration for a specific input channel, the user should send the frame below. The device should return the command above.

Field	ID	LEN	DATA
Value	0x4F	0x01	Channel (uint8)
Size (bytes)	12 or 32 bits	6 bits	1

SBG_CAN_ID_LOGIC_OUT_CHANNEL (0x50)

This command is used to set/get the configuration for a specific output channel. Only one channel can be configured but for upward compatibly, the command contains a channel field that should always be set to 0.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	yes	yes	yes

Possible values for output type are:

- `SBG_OUT_DISABLED` `0x00` This output channel is not used
- `SBG_OUT_MAIN_LOOP_START` `0x01` An output signal is generated when a main loop starts
- `SBG_OUT_MAIN_LOOP_DIVIDER` `0x02` An output signal is generated when a divided main loop starts
- `SBG_OUT_TIME_PULSE_COPY` `0x03` Copy of the GPS time a time pulse signal
- `SBG_OUT_VIRTUAL_ODO` `0x05` Virtual odometer logic output: Enabled each x meters of travel

Possible values for output polarity are:

- `SBG_IN_FALLING_EDGE` `0x00` An output event will generate a falling edge
- `SBG_IN_RISING_EDGE` `0x01` An output event will generate a rising edge
- `SBG_IN_TOGGLE` `0x02` An output event will toggle the output pin level

Send the command below to define for an output channel (only 0) its type, sensitivity and duration in millisecond. If the new settings have been applied successfully, the device should return the exact same frame as the sent one.

Field	ID	LEN	DATA			
Value	0x50	0x04	Channel (uint8)	Output type (uint8)	Output polarity (uint8)	Delay ms (uint8)
Size (bytes)	12 or 32 bits	6 bits	1	1	1	1

To retrieve the configuration for a specific output channel, the user should send the frame below. The device should return the command above.

Field	ID	LEN	DATA
Value	0x50	0x01	Channel (uint8)
Size (bytes)	12 or 32 bits	6 bits	1

SBG_CAN_ID_VIRTUAL_ODO_CONF (0x56)

This function is used to configure the distance between two pulses when the sync out signal is used as a virtual odometer.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	no	no	no	yes	yes

Send the command below to define for an output channel (only 0) its type, sensitivity and duration in millisecond. If the new settings have been applied successfully, the device should return the exact same frame as the sent one.

Field	CMD	LEN	DATA	
Value	0x56	8	Distance in m (fixed32)	Reserved; Leave to 0 (uint32)
Size (bytes)	1	2	4	4

To retrieve the configuration, the user should send the same frame with an empty payload. The device should return the command above.

2.11. Advanced options

SBG_CAN_ID_ADVANCED_OPTIONS (0x57)

The IG-500 now includes a set of advanced features that can be turned on or off with a single command.

Commands availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
no	no	no	yes	yes	yes

All features are contained in a 32 bit bit-mask described as below:

```
SBG_SETTING_ENABLE_CONING         0x00000001   Use coning integration in the Kalman
                                               filter instead of gyroscopes values
SBG_SETTING_ALTITUDE_ABOVE_MSL    0x00000004   - 0 -> Altitude is above Ellipsoid
                                               - 1 -> Altitude is above Mean Sea Level
SBG_SETTING_DECLINATION_AUTO      0x00000008   Enable/Disable automatic magnetic
                                               declination computation
SBG_SETTING_GRAVITY_AUTO          0x00000010   Enable/Disable automatic local gravity
                                               computation
SBG_SETTING_OUTPUT_UNBIASED_GYRO  0x00000020   Enable/Disable Kalman unbiased gyroscope
                                               and delta angle outputs
SBG_SETTING_OUTPUT_UNBIASED_ACCEL 0x00000040   Enable/Disable Kalman unbiased
                                               accelerometer output
SBG_SETTING_MAG_HORIZONTAL_ONLY   0x00000080   Restrict magnetometers use to horizontal
                                               position (used in certain 2D calibrations)
SBG_SETTING_STATIC_INIT           0x00000100   Initialize the Kalman filter with he no
                                               motion assumption for 10 seconds for
                                               faster startup
SBG_SETTING_STATIC_INIT_UNTIL_MOTION_DETECTED 0x00000200 Force the Kalman filter to use
                                               the static assumption, until a motion is
                                               detected. This allows to:
                                               - Estimate gyro bias without heading
                                               - Get a faster startup
```

This function gives access to several miscellaneous settings by the use of a bit-mask.

Send the command below to define miscellaneous settings. If the new settings have been applied successfully, the device should return the exact same frame as the sent one.

Field	ID	LEN	DATA	
Value	0x57	0x08	Options1 (uint32)	Reserved Leave to 0
Size (bytes)	12 or 32 bits	6 bits	4	4

To retrieve the configuration the user should send the same frame without payload. The device should return the command above.

2.12. IG-20 and IG-30 Specific commands

SBG_CAN_ID_FILTER_FREQUENCIES (0x39)

This command is used to configure the three accelerometers, gyroscopes and magnetometers low pass filters cut-off frequencies.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	no	no	no

The command below is used to set the new low pass filters cut-off frequency to use for each sensor. If the three new filter frequencies have been changed successfully, the device should return the exact same frame as the sent one.

Field	ID	LEN	DATA		
Value	0x39	0x06	Cut-off gyro. (int16)	Cut-off accel. (int16)	Cut-off mag. (int16)
Unit	-	-	10^{-1} Hz	10^{-1} Hz	10^{-1} Hz
Size (bytes)	12 or 32 bits	6 bits	2	2	2

To retrieve the gyroscopes, accelerometers and magnetometers low pass filters cut-off frequencies, the user should send the frame with an empty payload. The device should return the command above.

Note 1: *If a cut-off frequency is set to 0, it will just be ignored by the device without returning any error. Using this mechanism, it's possible to easily change each cut-off frequency individually.*

Note2: *To disable a low pass filter, just set the filter frequency greater or equal to the Kalman filter frequency.*

SBG_CAN_ID_KALMAN_FILTER (0x3A)

This command is used to configure the internal main loop frequency and to define some filter options.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	no	no	no

The Kalman period setting is used to define the filter frequency. A period of 1 equals to 0.0001 s so if you set the Kalman period to 100, the actual period will be 0.01 s and the Kalman filter will run at 100 Hz.

The following formula could be used to compute the Kalman filter frequency:

$$F_{kal}=\frac{1}{Period_{kal}\cdot 10^{-4}}$$ where F_{kal} is the Kalman filter frequency in Hz.

Available Kalman filter options that can be combined using a bitwise OR:

```
SBG_FILTER_OPTION_ENABLE_ATTITUDE            0x10   Compute attitude and navigation
```

Send the command below to set the Kalman filter period and filter options. If the new settings have been applied successfully, the device should return the exact same frame as the sent one.

Field	ID	LEN	DATA	
Value	0x3A	0x04	Kalman filter period (uint16)	Kalman filter options (uint16)
Size (bytes)	12 or 32 bits	6 bits	2	2

To retrieves the Kalman filter period and options, the user should send the frame with an empty payload. The device should return the command above.

SBG_CAN_ID_GPS_OPTIONS (0x3F)

This command is used to configure/retrieve some GPS options such as the dynamic model used for position computation or if the GPS altitude should be referenced to mean sea level or ellipsoid.

Command availability:

IG-20	IG30A	IG-30G	IG-500A	IG-500N	IG-500E
yes	yes	yes	no	no	no

Possible values for dynamic model are:

`SBG_GPS_MODEL_STATIONARY`	`0x01`	The GPS assume to be steady
`SBG_GPS_MODEL_PEDESTRIAN`	`0x02`	Slow movements, low bandwidth
`SBG_GPS_MODEL_AUTOMOTIVE`	`0x03`	Medium bandwidth, altitude almost fixed
`SBG_GPS_MODEL_SEA`	`0x04`	Medium bandwidth, altitude is fixed
`SBG_GPS_MODEL_AIRBONE_1G`	`0x05`	Medium bandwidth, altitude can change
`SBG_GPS_MODEL_AIRBONE_2G`	`0x06`	High bandwidth, altitude can change (default)
`SBG_GPS_MODEL_AIRBONE_4G`	`0x07`	Very high bandwidth, no low pass filter on GPS

Possible GPS options that can be combined using at bitwise OR:

`SBG_GPS_DISABLE_SBAS`	`0x00`	Helper that disable SBAS and ranging
`SBG_GPS_ENABLE_SBAS_DIFF_CORRECTIONS`	`0x01`	Enable SBAS corrections (default)
`SBG_GPS_ENABLE_SBAS_RANGING`	`0x02`	Enable SBAS ranging (default)
`SBG_GPS_ALTITUDE_ABOVE_MSL`	`0x04`	Referenced to MSL instead of ellipsoid

Send the command below to configure the GPS dynamic model and some GPS options. If the new setting have been applied successfully, the device should return the exact same frame as the sent one.

Field	ID	LEN	DATA	
Value	0x3F	0x02	Dynamic model (uint8)	GPS options (uint8)
Size (bytes)	12 or 32 bits	6 bits	1	1

To retrieve the GPS dynamic model and options the user should send this frame with an empty payload. The device should return the command above.

3. Output frames

3.1. Time and status related outputs

SBG_CAN_ID_OUTPUT_TIMESTAMP_TRIGGER (0x00)

Contains the time in ms since the last device reset and a trigger mask that indicates, for the current loop, which triggers have been activated.

The time since reset information is measured at each beginning of a sampling and calculation loop.

The trigger mask is a bitwise OR of the following masks:

SBG_TRIGGER_MAIN_LOOP_DIVIDER	0x0001	Trigger on the main loop frequency divider
SBG_TRIGGER_MAGNETOMETERS	0x0002	Trigger on a new magnetometers data
SBG_TRIGGER_BAROMETER	0x0004	Trigger on a new barometer data
SBG_TRIGGER_GPS_VELOCITY	0x0008	Trigger on a new GPS velocity
SBG_TRIGGER_GPS_POSITION	0x0010	Trigger on a new GPS position
SBG_TRIGGER_GPS_COURSE	0x0020	Trigger on a new GPS heading
SBG_TRIGGER_TIME_PULSE	0x0040	Trigger on an input time pulse
SBG_TRIGGER_EXT_EVENT	0x0080	Trigger on a new external event
SBG_TRIGGER_ODO_VELOCITY_0	0x0100	Trigger on a new odometer 0 velocity
SBG_TRIGGER_ODO_VELOCITY_1	0x0200	Trigger on a new odometer 1 velocity
SBG_TRIGGER_EXT_TRUE_HEADING	0x0400	Trigger on a new true heading
SBG_TRIGGER_VIRTUAL_ODOMETER	0x0800	Trigger when the virtual odometer reached desired distance

Field	ID	LEN	DATA	
Value	0x00	0x06	Time since reset (uint32)	Trigger mask (uint16)
Unit	-	-	ms	**-**
Size (bytes)	1	6 bits	4	2

SBG_CAN_ID_OUTPUT_DEVICE_STATUS (0x01)

This output contains the device status used to monitor device initialization, health. Some status are only updated during device initialization, others are updated in real time.

The following bits are stored in the 32 bits bit-mask:

Bit	Name	Description
LSB	SBG_CALIB_INIT_STATUS_MASK	Set to 1 if the calibration structure is well initialized
1	SBG_SETTINGS_INIT_STATUS_MASK	Set to 1 if the settings structure is well initialized
2	SBG_ACCEL_X_SELF_TEST_STATUS_MASK	Set to 1 if the X accelerometer has passed self test
3	SBG_ACCEL_Y_SELF_TEST_STATUS_MASK	Set to 1 if the Y accelerometer has passed self test
4	SBG_ACCEL_Z_SELF_TEST_STATUS_MASK	Set to 1 if the Z accelerometer has passed self test
5	SBG_ACCEL_RANGE_STATUS_MASK	Set to 1 if accelerometers do not exceed operating range.
6	SBG_GYRO_X_SELF_TEST_STATUS_MASK	Set to 1 if the X gyroscope has passed self test
7	SBG_GYRO_Y_SELF_TEST_STATUS_MASK	Set to 1 if the Y gyroscope has passed self test
8	SBG_GYRO_Z_SELF_TEST_STATUS_MASK	Set to 1 if the Z gyroscope has passed self test
9	SBG_GYRO_RANGE_STATUS_MASK	Set to 1 if gyroscopes do not exceed operating range.
10	SBG_MAG_CALIBRATION_STATUS_MASK	Set to 1 if the magnetic field calibration looks OK
11	SBG_ALTI_INIT_STATUS_BIT_MASK	Set to 1 if altimeter could initialize
12	SBG_GPS_STATUS_BIT_MASK	Set to 1 if GPS receiver or the external device has been initialized successfully and sends valid data
13	SBG_G_MEASUREMENT_VALID_MASK	Set to 1 if gravity is observable sufficiently for proper Kalman filter operation
14	SBG_HEADING_MEASUREMENT_VALID_MASK	Set to 1 if heading is observable sufficiently for proper Kalman filter operation
15	SBG_VEL_MEASUREMENT_VALID_MASK	Set to 1 if velocity is observable sufficiently for proper Kalman filter operation
16	SBG_POS_MEASUREMENT_VALID_MASK	Set to 1 if position is observable sufficiently for proper Kalman filter operation
17	SBG_UTC_VALID_MASK	Bit mask for GPS UTC Validation: Leap Seconds already known
18	SBG_UTC_ROUGH_ACCURACY_MASK	Bit mask for UTC time validation with a 0.25s accuracy
19	SBG_UTC_SYNCHRONIZED_MASK	Bit mask for UTC time synchronization with < 1µs accuracy
20	SBG_PROTOCOL_OUTPUT_STATUS	Set to 1 if outputted data are not saturating the protocol
[21 – 31]	-	Reserved – Set to 0

Field	ID	LEN	DATA
Value	0x01	0x04	Device status (uint32)
Size (bytes)	12 or 32 bits	6 bits	4

SBG_CAN_ID_OUTPUT_UTC_TIME (0x02)

This structure contains the following absolute UTC time referenced information:

- Year after 2000, range [0 to 255]
- Month of year, range [1 to 12]
- Day of month, range [1 to 31]
- Microsecond time of day encoded on 5 bytes

Field	**ID**	**LEN**	**DATA**			
Value	0x02	0x08	Year (uint8)	Month (uint8)	Day (uint8)	Time of day (uint40)
Unit	-	-	-	-	-	µs
Size (bytes)	12 or 32 bits	6 bits	1	1	1	5

3.2. Orientation outputs

SBG_CAN_ID_OUTPUT_QUATERNION (0x03)

Device 3d orientation represented with a normalized quaternion (w, x, y, z) expressed using frac16 numbers as described in the paragraph 1.6.1Scalar types.

Field	ID	LEN	DATA			
Value	0x03	0x08	Q0 (frac16)	Q1 (frac16)	Q2 (frac16)	Q3 (frac16)
Size (bytes)	12 or 32 bits	6 bits	2	2	2	2

SBG_CAN_ID_OUTPUT_EULER (0x04)

Device 3d orientation represented using Euler angles in radians. Multiply each component by 10^{-4} to get the value in radians.

Field	ID	LEN	DATA		
Value	0x04	0x06	Roll (int16)	Pitch (int16)	Yaw (int16)
Unit	-	-	10^{-4} rad	10^{-4} rad	10^{-4} rad
Size (bytes)	12 or 32 bits	6 bits	2	2	2

SBG_CAN_ID_OUTPUT_HEADING (0x05)

Kalman estimated heading and heading accuracy. The heading value is the same as the Yaw value returned by the frame SBG_CAN_ID_OUTPUT_EULER (0x04). This frame can be used as an aiding heading data for an other device that accepts the aiding frame SBG_CAN_ID_SEND_FILTER_HEADING (0x62). Multiply each component by 10^{-5} to get the values in degrees.

Field	ID	LEN	DATA	
Value	0x05	0x08	Heading (int32)	Heading accuracy (uint32)
Unit	-	-	10^{-5} degrees	10^{-5} degrees
Size (bytes)	12 or 32 bits	6 bits	4	4

3.3. Calibrated Sensors output

SBG_CAN_ID_OUTPUT_GYROSCOPES (0x06)

Fully calibrated and Kalman unbiased gyroscopes values in radians per second. Multiply each component by 10^{-3} to get the value in radians per second.

Field	ID	LEN	DATA		
Value	0x06	0x06	Rate of turn X (int16)	Rate of turn Y (int16)	Rate of turn Z (int16)
Unit	-	-	10^{-3} rad.s^{-1}	10^{-3} rad.s^{-1}	10^{-3} rad.s^{-1}
Size (bytes)	12 or 32 bits	6 bits	2	2	2

SBG_CAN_ID_OUTPUT_DELTA_ANGLES (0x1F)

Coning integration output from the 1kHz gyroscopes integration. These values should be used as the gyroscopes values and should be multiplied by 10^{-3} to get the value in radians per second.

Field	ID	LEN	DATA		
Value	0x1F	0x06	Coning integration X (int16)	Coning integration Y (int16)	Coning integration Z (int16)
Unit	-	-	10^{-3} rad.s^{-1}	10^{-3} rad.s^{-1}	10^{-3} rad.s^{-1}
Size (bytes)	12 or 32 bits	6 bits	2	2	2

SBG_CAN_ID_OUTPUT_ACCELEROMETERS (0x07)

Fully calibrated and Kalman unbiased accelerometers values in meters per second per second. Multiply each component by 10^{-2} to get the value in meters per second per second.

Field	ID	LEN	DATA		
Value	0x07	0x06	Acceleration X (int16)	Acceleration Y (int16)	Acceleration Z (int16)
Unit	-	-	10^{-2} m.s^{-2}	10^{-2} m.s^{-2}	10^{-2} m.s^{-2}
Size (bytes)	12 or 32 bits	6 bits	2	2	2

SBG_CAN_ID_OUTPUT_MAGNETOMETERS (0x08)

Fully calibrated and normalized magnetometers values in arbitrary units. Multiply each component by 10^{-3} to get the value in the A.U. If well calibrated, the norm of the magnetic vector should be equal to 1.

Field	ID	LEN	DATA		
Value	0x08	0x06	Magnetometer X (int16)	Magnetometer Y (int16)	Magnetometer Z (int16)
Unit	-	-	10^{-3} A.U.	10^{-3} A.U.	10^{-3} A.U.
Size (bytes)	12 or 32 bits	6 bits	2	2	2

SBG_CAN_ID_OUTPUT_TEMPERATURES (0x09)

Device temperature measured near the accelerometers/magnetometers and internal ADC temperature in °C. Multiply each component by 10^{-2} to get the value in °C.

Field	ID	LEN	DATA	
Value	0x09	0x04	Accels/Mags temperature (int16)	ADC temperature (int16)
Unit	-	-	10^{-2} °C	10^{-2} °C
Size (bytes)	12 or 32 bits	6 bits	2	2

SBG_CAN_ID_OUTPUT_GYRO_TEMPERATURES (0x0A)

Internal gyroscopes X, Y, Z temperatures in °C used for gyroscopes temperature compensation. Multiply each component by 10^{-2} to get the temperature in °C.

Field	ID	LEN	DATA		
Value	0x0A	0x06	Gyro X temperature (int16)	Gyro Y temperature (int16)	Gyro Z temperature (int16)
Unit	-	-	10^{-2} °C	10^{-2} °C	10^{-2} °C
Size (bytes)	12 or 32 bits	6 bits	2	2	2

3.4. Kalman Navigation outputs

SBG_CAN_ID_OUTPUT_POSITION_1 (0x0B)

Latitude and longitude enhanced using inertial data and expressed in the WGS84 format. Multiply each component by 10^{-7} to get the value in degrees.

Field	ID	LEN	DATA	
Value	0x0B	0x08	Latitude (int32)	Longitude (int32)
Unit	-	-	10^{-7} degrees	10^{-7} degrees
Size (bytes)	12 or 32 bits	6 bits	4	4

SBG_CAN_ID_OUTPUT_POSITION_2 (0x0C)

Altitude in millimeters expressed either in height above ellipsoid or mean sea level according to the GPS configuration. Horizontal and vertical accuracies estimated by the Kalman filter are expressed in centimeters.

Field	ID	LEN	DATA		
Value	0x0C	0x08	Altitude (int32)	Horizontal accuracy (uint16)	Vertical accuracy (uint16)
Unit	-	-	mm	cm	cm
Size (bytes)	12 or 32 bits	6 bits	4	2	2

SBG_CAN_ID_OUTPUT_VELOCITY_1 (0x0D)

X and Y velocities enhanced using inertial data and expressed in the device coordinate frame in centimeters per second.

Field	ID	LEN	DATA	
Value	0x0D	0x08	Velocity X (int32)	Velocity Y (int32)
Unit	-	-	cm.s^{-1}	cm.s^{-1}
Size (bytes)	12 or 32 bits	6 bits	4	4

SBG_CAN_ID_OUTPUT_VELOCITY_2 (0x0E)

Z velocity enhanced using inertial data and expressed in the device coordinate frame in centimeters per second. The velocity accuracy is estimated by the Kalman filter and is expressed in centimeters per second.

Field	ID	LEN	DATA	
Value	0x0E	0x06	Velocity Z (int32)	Velocity accuracy (uint16)
Unit	-	-	cm.s^{-1}	cm.s^{-1}
Size (bytes)	12 or 32 bits	6 bits	4	2

SBG_CAN_ID_OUTPUT_HEAVE (0x20)

Relative altitude output with respect to mean sea level. Heave computation must be enabled for this output to work correctly.

Field	ID	LEN	DATA
Value	0x20	0x04	Relative Altitude
Unit	-	-	mm
Size (bytes)	12 or 32 bits	6 bits	4

3.5. Raw sensors output

SBG_CAN_ID_OUTPUT_GYROSCOPES_RAW (0x0F)

Raw gyroscopes 16 bits values as read by the Analog to Digital Converted (ADC).

Field	ID	LEN	DATA		
Value	0x0F	0x06	Raw gyroscope X (uint16)	Raw gyroscope Y (uint16)	Raw gyroscope Z (uint16)
Size (bytes)	12 or 32 bits	6 bits	2	2	2

SBG_CAN_ID_OUTPUT_ACCELEROMETERS_RAW (0x10)

Raw accelerometers 16 bits values as read by the Analog to Digital Converted (ADC).

Field	ID	LEN	DATA		
Value	0x10	0x06	Raw accelerometer X (uint16)	Raw accelerometer Y (uint16)	Raw accelerometer Z (uint16)
Size (bytes)	12 or 32 bits	6 bits	2	2	2

SBG_CAN_ID_OUTPUT_MAGNETOMETERS_RAW (0x11)

Raw magnetometers 12 bits values as read by the Analog to Digital Converted (ADC).

Field	ID	LEN	DATA		
Value	0x11	0x06	Raw magnetometer X (uint16)	Raw magnetometer Y (uint16)	Raw magnetometer Z (uint16)
Size (bytes)	12 or 32 bits	6 bits	2	2	2

SBG_CAN_ID_OUTPUT_TEMPERATURES_RAW (0x12)

Raw accelerometers/magnetometers and ADC temperatures 12 bits values as read by the Analog to Digital Converted (ADC).

Field	ID	LEN	DATA	
Value	0x12	0x04	Accels/Mags temperature (uint16)	ADC temperature (uint16)
Size (bytes)	12 or 32 bits	6 bits	2	2

SBG_CAN_ID_OUTPUT_GYRO_TEMPERATURES_RAW (0x13)

Raw internal gyroscopes temperature 12 bits values as read by the Analog to Digital Converted (ADC).

Field	ID	LEN	DATA		
Value	0x13	0x06	Gyro temperature X (uint16)	Gyro temperature Y (uint16)	Gyro temperature Z (uint16)
Size (bytes)	12 or 32 bits	6 bits	2	2	2

3.6. Other sensors Outputs

SBG_CAN_ID_OUTPUT_BAROMETER (0x14)

Barometric pressure measured by the pressure sensor and altitude calculated using pressure information and a reference pressure configured by user. The default reference pressure is set to 101 325 Pascals.

Field	ID	LEN	DATA	
Value	0x14	0x08	Pressure (uint32)	Altitude (int32)
Unit	-	-	Pascals	cm
Size (bytes)	12 or 32 bits	6 bits	4	4

SBG_CAN_ID_OUTPUT_MAG_CALIB_DATA (0x15)

This output contains the data used to calibrate magnetometers hard and soft iron effects. The sbgCenter application and the iron calibration library use this 6 bytes buffer to compute magnetometers calibration.

Field	ID	LEN	DATA
Value	0x15	0x06	Calibration data buffer 6 * (uint8)
Size (bytes)	12 or 32 bits	6 bits	6

SBG_CAN_ID_OUTPUT_ODOMETER_VELOCITIES (0x16)

Odometers raw velocities only computed using detected pulses and odometer pulses per meter setting. The two odometers channels are expressed in millimeters per second.

Field	ID	LEN	DATA	
Value	0x16	0x08	Odometer 0 velocity (int32)	Odometer 1 velocity (int32)
Unit	-	-	mm.s^{-1}	mm.s^{-1}
Size (bytes)	12 or 32 bits	6 bits	4	4

3.7. Raw GPS outputs

SBG_CAN_ID_OUTPUT_GPS_INFO (0x17)

This frame contains some basic information about the GPS Status :

- GPS time of the week in milliseconds,
- GPS fix information and UTC status,
- Number of satellites used by GPS for navigation solution.

The GPS flags is composed of the following flags:

Bit	(MSB) 7	6	5	4	3	2	1	0 (LSB)
Field	Ext Fix info		GPS_TRUE_ HEAD_VALID	GPS_VALID_UTC	GPS_VALID_WKN	GPS_VALID_TOW	GPS Fix info	
Desc.	**00**=GPS_STD_FIX **01**=GPS_DGPS_FIX **10**=GPS_FRTK_FIX **11**=GPS_RTK_FIX		**1** = Valid true heading data	**1** = Valid UTC data (leap seconds known)	**1** = Week number known	**1** = Time of week known.	**00**=SBG_NO_FIX **01**=SBG_TIME_ONLY **10**=SBG_2D_FIX **11**=SBG_3D_FIX	

Now here is the frame structure:

Field	ID	LEN	DATA		
Value	0x17	0x06	GPS time of week (uint32)	GPS status flags (uint8)	Used satellites count (uint8)
Unit	-	-	ms	-	-
Size (bytes)	12 or 32 bits	6 bits	4	1	1

SBG_CAN_ID_OUTPUT_GPS_SVINFO (0x18)

This output send advanced information about satellites tracked by the GPS receiver. When a new space vehicle information should be sent, for each channel used by the GPS receiver, the frame below will be send. For example an U-Blox 6 receiver have fifty channels so it can theoretically send the following frame up to fifty times. In real conditions, the fifty channels are never all in use by the GPS receiver.

For each sent space vehicle, the following information are provided:

- A channel field that represents which GPS receiver channel is being sent,
- The satellite id returned by the space vehicle tracked by this channel,
- The signal strength indication in dbHz also named carrier to noise ratio,
- The space vehicle azimuth where 1 LSB = 32/45 degree
- The space vehicle elevation where 1 LSB = 32/45 degree

The quality flags contains the following data:

Bit	Name	Description
0	`SBG_SV_USED_FOR_NAV`	SV is used for navigation
1	`SBG_SV_DIFF_AVAILABLE`	Differential correction is available for this SV
2	`SBG_SV_ORBIT_AVAILABLE`	Orbit information is available for this SV
3	`SBG_SV_ORBIT_EPHEMERIS`	Orbit information is Ephemeris
4	`SBG_SV_UNHEALTHY`	SV is unhealthy / shall not be used

[5 - 7]	`SBG_SV_QUAL_IDLE (0x00)`
	`SBG_SV_QUAL_SEARCHING_1 (0x01)`
	`SBG_SV_QUAL_SERACHING_2 (0x02)`
	`SBG_SV_QUAL_DETECTED_UNUSABLE (0x03)`
	`SBG_SV_QUAL_CODE_LOCK_ON (0x04)`
	`SBG_SV_QUAL_CODE_AND_CARRIER_LOCKED_1 (0x05)`
	`SBG_SV_QUAL_CODE_AND_CARRIER_LOCKED_2 (0x06)`
	`SBG_SV_QUAL_RECEIVING_DATA (0x07)`
		This channel is idle
		Channel is searching
		Signal acquired
		Signal detected but unusable
		Code Lock on Signal
		Code and Carrier locked
		Code and Carrier locked
		Code and Carrier locked, receiving 50bps data

Field	ID	LEN	DATA					
Value	0x18	0x06	Channel (uint8)	Satellite id (uint8)	Quality flags (uint8)	Signal Strength (uint8)	Azimuth (int8)	Elevation (int8)
Units	-	-	-	-	-	dbHz	32/45 degs	32/45 degs
Size (bytes)	12 or 32 bits	6 bits	1	1	1	1	1	1

Note: *Please refer to the User Manual documentation for more details about advanced GPS output.*

SBG_CAN_ID_OUTPUT_GPS_POSITION_1 (0x19)

Raw GPS latitude and longitude expressed in the WGS84 format. Multiply each component by 10^{-7} to get the value in degrees.

Field	ID	LEN	DATA	
Value	0x19	0x08	Latitude (int32)	Longitude (int32)
Unit	-	-	10^{-7} degrees	10^{-7} degrees
Size (bytes)	12 or 32 bits	6 bits	4	4

SBG_CAN_ID_OUTPUT_GPS_POSITION_2 (0x1A)

Raw GPS altitude in millimeters expressed either in height above ellipsoid or mean sea level according to the GPS configuration. Raw GPS horizontal and vertical accuracies expressed in centimeters.

Field	ID	LEN	DATA		
Value	0x1A	0x08	Altitude (int32)	Horizontal accuracy (uint16)	Vertical accuracy (uint16)
Unit	-	-	mm	cm	cm
Size (bytes)	12 or 32 bits	6 bits	4	2	2

SBG_CAN_ID_OUTPUT_GPS_VELOCITY_1 (0x1B)

Raw GPS horizontal velocities expressed in centimeters per second in a North East Down (NED) coordinate system.

Field	ID	LEN	DATA	
Value	0x1B	0x08	Velocity North (int32)	Velocity East (int32)
Unit	-	-	$cm.s^{-1}$	$cm.s^{-1}$
Size (bytes)	12 or 32 bits	6 bits	4	4

SBG_CAN_ID_OUTPUT_GPS_VELOCITY_2 (0x1C)

Raw GPS vertical velocity expressed in centimeters per second in a North East Down (NED) coordinate system. Raw GPS velocity accuracy estimate expressed in centimeters per second.

Field	ID	LEN	DATA	
Value	0x1C	0x06	Velocity Down (int32)	Velocity accuracy (uint16)
Unit	-	-	$cm.s^{-1}$	$cm.s^{-1}$
Size (bytes)	12 or 32 bits	6 bits	4	2

SBG_CAN_ID_OUTPUT_GPS_COURSE (0x1D)

Raw GPS course of motion and course accuracy estimate expressed in degrees. Multiply each component by 10^{-5} to get the value in degrees.

Field	ID	LEN	DATA	
Value	0x1D	0x08	Course (int32)	Course accuracy (uint32)
Unit	-	-	10^{-5} degrees	10^{-5} degrees
Size (bytes)	12 or 32 bits	6 bits	4	4

SBG_CAN_ID_OUTPUT_GPS_TRUE_HEADING (0x1E)

Raw GPS true heading if available and true heading accuracy estimate expressed in degrees. Multiply each component by 10^{-5} to get the value in degrees.

Field	ID	LEN	DATA	
Value	0x1E	0x08	True heading (int32)	True heading accuracy (uint32)
Unit	-	-	10^{-5} degrees	10^{-5} degrees
Size (bytes)	12 or 32 bits	6 bits	4	4

4. Input frames

4.1. Attitude related aiding source

When the device has been configured with the command SBG_CAN_ID_FILTER_HEADING_SOURCE (0x3C) to use a user source for heading reference, this frame is used to input a new heading information.

As soon as this command is received by the device, the Kalman filter will use this new heading information to compensate for the heading drift. Each heading information is associated with an accuracy field needed by the Kalman filter to determine the amount of corrections to apply.

If your measurement is very accurate (accuracy field near 0.0), the filter will rely mostly on the provided information otherwise, it will correct only a small error on the estimated heading.

SBG_CAN_ID_SEND_FILTER_HEADING (0x62)

This command allows to input a heading reference provided by the user. This heading reference will be used by the Kalman filter to compensate for heading drifts.

IG-500A	IG-500N	IG-500E
yes	yes	yes

The valid range for heading is between $[-\pi, +\pi]$ and for heading accuracy is between $[0; +2\pi]$.

Send the command below to input a heading reference to the Kalman filter. To convert a heading or a heading accuracy expressed in degrees into the command data format, just multiply it by 10^5.

Field	ID	LEN	DATA	
Value	0x62	0x08	Heading (int32)	Heading accuracy (uint32)
Unit	-	-	10^{-5} degrees	10^{-5} degrees
Size (bytes)	12 or 32 bits	6 bits	4	4

The device does not answer to this command.

4.2. Navigation related aiding source

When the device has been configured with the command SBG_CAN_ID_NAV_SOURCES (0x40) to use a user source for velocity/position aiding, the following commands are used to send a new velocity/position information.

As soon as those commands are received by the device, the Kalman filter will use this new velocity/position information to compensate for the drift in velocity/position. Each information is associated with an accuracy field used by the Kalman filter to determine the amount of corrections to apply.

If your measurement is very accurate (accuracy field near 0.0), the filter will rely mostly on the provided information otherwise, it will correct only a small error on the estimated velocity/position.

SBG_CAN_ID_SEND_NAV_VELOCITY_LOCAL_1 (0x63)

This command allows to input a velocity reference provided by the user. This velocity reference will be used by the Kalman filter to compensate for velocity drifts.

IG-500A	IG-500N	IG-500E
no	yes	yes

Send the command below to input a velocity reference to the Kalman filter. The velocity is expressed in centimeters per second in the device coordinate system.

Field	ID	LEN	DATA	
Value	0x63	0x08	Velocity X (int32)	Velocity Y (int32)
Unit	-	-	cm.s^{-1}	cm.s^{-1}
Size (bytes)	12 or 32 bits	6 bits	4	4

The device does not answer to this command.

Note: *The new velocity reference will only be used by the Kalman filter when the two commands SBG_CAN_ID_SEND_NAV_VELOCITY_LOCAL_1 (0x63) and SBG_CAN_ID_SEND_NAV_VELOCITY_LOCAL_2 (0x64) are received in this specific order.*

SBG_CAN_ID_SEND_NAV_VELOCITY_LOCAL_2 (0x64)

This command allows to input a velocity reference provided by the user. This velocity reference will be used by the Kalman filter to compensate for velocity drifts.

IG-500A	IG-500N	IG-500E
no	yes	yes

Send the command below to input a velocity reference and the velocity accuracy to the Kalman filter. The velocity is expressed in centimeters per second in the device coordinate system.

Field	ID	LEN	DATA	
Value	0x64	0x06	Velocity Z (int32)	Velocity accuracy (uint16)
Unit	-	-	cm.s^{-1}	cm.s^{-1}
Size (bytes)	12 or 32 bits	6 bits	4	2

The device does not answer to this command.

Note: *The new velocity reference will only be used by the Kalman filter when the two commands SBG_CAN_ID_SEND_NAV_VELOCITY_LOCAL_1 (0x63) and SBG_CAN_ID_SEND_NAV_VELOCITY_LOCAL_2 (0x64) are received in this specific order.*

SBG_CAN_ID_SEND_NAV_VELOCITY_NED_1 (0x65)

This command allows to input a velocity reference provided by the user. This velocity reference will be used by the Kalman filter to compensate for velocity drifts.

IG-500A	IG-500N	IG-500E
no	no	no

Send the command below to input a velocity reference to the Kalman filter. The velocity is expressed in centimeters per second in the North East Down (NED) coordinate system.

Field	ID	LEN	DATA	
Value	0x63	0x08	Velocity North (int32)	Velocity East (int32)
Unit	-	-	cm.s^{-1}	cm.s^{-1}
Size (bytes)	12 or 32 bits	6 bits	4	4

The device does not answer to this command.

Note: *The new velocity reference will only be used by the Kalman filter when the two commands SBG_CAN_ID_SEND_NAV_VELOCITY_NED_1 (0x65) and SBG_CAN_ID_SEND_NAV_VELOCITY_NED_2 (0x66) are received in this specific order.*

SBG_CAN_ID_SEND_NAV_VELOCITY_NED_2 (0x66)

This command allows to input a velocity reference provided by the user. This velocity reference will be used by the Kalman filter to compensate for velocity drifts.

IG-500A	IG-500N	IG-500E
no	yes	yes

Send the command below to input a velocity reference and the velocity accuracy to the Kalman filter. The velocity is expressed in centimeters per second in the North East Down (NED) coordinate system.

Field	ID	LEN	DATA	
Value	0x66	0x06	Velocity Down (int32)	Velocity accuracy (uint16)
Unit	-	-	$cm.s^{-1}$	$cm.s^{-1}$
Size (bytes)	12 or 32 bits	6 bits	4	2

The device does not answer to this command.

> ***Note:*** *The new velocity reference will only be used by the Kalman filter when the two commands SBG_CAN_ID_SEND_NAV_VELOCITY_NED_1 (0x65) and SBG_CAN_ID_SEND_NAV_VELOCITY_NED_2 (0x66) are received in this specific order.*

SBG_CAN_ID_SEND_NAV_POSITION_1 (0x67)

This command allows to input a position reference provided by the user. This position reference will be used by the Kalman filter to compensate for position drifts.

IG-500A	IG-500N	IG-500E
no	yes	yes

Send the command below to input a latitude and longitude position reference to the Kalman filter. The position is expressed in the WGS84 format. To convert a latitude or longitude expressed in degrees into the command data format, just multiply it by 10^7.

Field	ID	LEN	DATA	
Value	0x67	0x08	Latitude (int32)	Longitude (int32)
Unit	-	-	10^{-7} degrees	10^{-7} degrees
Size (bytes)	12 or 32 bits	6 bits	4	4

The device does not answer to this command.

> ***Note:*** *The new position reference will only be used by the Kalman filter when the two commands SBG_CAN_ID_SEND_NAV_POSITION_1 (0x67) and SBG_CAN_ID_SEND_NAV_POSITION_2 (0x68) are received in this specific order.*

SBG_CAN_ID_SEND_NAV_POSITION_2 (0x68)

IG-500A	IG-500N	IG-500E
no	yes	yes

This command allows to input a position reference provided by the user. This position reference will be used by the Kalman filter to compensate for position drifts.

Send the command below to input an altitude reference, an horizontal accuracy and a vertical accuracy to the Kalman filter. The altitude is expressed in millimeters in WGS84 format. The horizontal and vertical accuracies are expressed in centimeters.

Field	ID	LEN	DATA		
Value	0x68	0x08	Altitude (int32)	Horizontal accuracy (uint16)	Vertical accuracy (uint16)
Unit	-	-	mm	cm	cm
Size (bytes)	12 or 32 bits	6 bits	4	2	2

The device does not answer to this command.

> ***Note:*** *The new position reference will only be used by the Kalman filter when the two commands SBG_CAN_ID_SEND_NAV_POSITION_1 (0x67) and SBG_CAN_ID_SEND_NAV_POSITION_2 (0x68) are received in this specific order.*

5. Support

If you have any question, comment or problem with the use of this low level protocol, we would be glad to help you, so please feel free to contact us. Please do not forget to mention your Device ID of your IG devices (written on your device label).

You can contact us by:

- Email : support@sbg-systems.com
- Phone : +33 (0)1 80 88 45 00