Basecam GPS IMU V.1.0

GNSS-aided AHRS/IMU module

Features

- Integrated GNSS-antenna for a compact size
- Full set of navigation sensors: 3-axis gyroscope, 3-axis accelerometer, 3-axis magnetometer, barometer (altimeter), GNSS receiver (concurrent Galileo, GLONASS, GPS / QZSS)
- Powerful MCU running UKF for a sensor fusion at high rates
- Thermal stabilization helps to keep high precision with the change of environmental temperatures
- Interfaces: UART, USB, CAN*, I2C*
- Onboard SD card for a high-rate data logging with the actual date-time information and flexible set of data to log
- Backup battery for a GNSS "warm start" mode and real-time clock
- Open serial API protocol allows using this sensor in various applications
- Compatible with all versions of the SimpleBGC32 controllers**
- Functionality will be constantly improved by the firmware updates
- Cost-effective solution compared to other GNSS-enabled solutions on the market

** CAN and I2C interfaces will be implemented in future releases of the firmware*
*** For full support of all functions of the Basecam GPS IMU, it's required to update the main controller's firmware to the version 2.68b7 and above.*

Overview

The **Basecam GPS IMU** is an GNSS-aided compact AHRS/IMU module developed by Basecamelectronics company to work as an external IMU sensor* with all versions of SimpleBGC32 controllers. It has much better precision compared to the internal IMU sensor, that allows improving precision of stabilization in demanding applications, where the regular IMU sensor does not work reliably.

GPS IMU connects to any free UART port of the main controller and provides precise attitude and heading information, that can be used as a reference to correct the internal IMU sensor, solving its common problems: gyroscope drift, an affection of linear accelerations, drift caused by the changes in environmental temperature.

**Note that currently GPS IMU module does not replace the regular IMU sensor, though this functionality may be implemented in future.*

Document revision: 1.0.63 (13. Aug. 2020)
Hardware version: 1.1
Firmware version: 1.0

Specifications

Mechanical	
Size of the enclosure	71×57×20 mm
Weight	45 g
Mounting holes	3 mm × 4

Electrical	Min	Typ	Max
Power supply	4.5 V	5 V	5.5 V
Drawn current (at 5V)		150 mA	400 mA (warming up)
Thermal stabilization		250 mWt	1500 mWt
Backup battery (CR1216) - current drawn in backup mode - lifetime (estimated)		3V 16 μA 75 days	

Communications	
Interfaces	1× UART 1× USB 1× CAN* 1× I2C* 1× microSD
Max. baud rates	UART: 2 MHz, I2C: 800 kHz, CAN: 2 MHz
Protocols	See "Basecam GPS_IMU Serial API 1.0.pdf" for a protocol specification
Data output	· Raw sensors data (accelerometer, magnetometer, barometer, gyroscope, GNSS) · Attitude (quaternion, DCM, Euler) · Angular rate · Navigation position and velocity
Max. data output rate	200 Hz (faster rates available by request)

Absolute maximum ratings	
Working temperature	-40..+85 °C
Acceleration	1000G for 1 ms
ESD rating	HBM: Class 2, 2000 V CDM: Class 3, 250 V

Performance		
Startup time	1 seconds (communication ready) 3 seconds (first valid data ready)	
GNSS best lock time**	Cold Hot	26 s 1 s
GNSS receiver sensitivity	Tracking & Navigation Reacquisition Cold start Hot start	−165 dBm −158 dBm −146 dBm −155 dBm
GNSS systems	Galileo, GLONASS, GPS, QZSS concurrently	
GNSS receiver update rate	10 Hz	
Internal update rates	Gyroscope data Accelerometer Magnetometer Attitude and heading Position & velocity	2000 Hz 2000 Hz 50 Hz 200 Hz 200 Hz
Angular velocity limits	±2000 deg./sec.	
Acceleration limits	16G	
Magnetic field strength limits	16 Gauss	
Factory calibrations	Gyroscope bias, accelerometer bias and scale, magnetometer bias and scale (at the single working temperature point)	
Thermal stabilization	Setpoint (adjustable) Time to warm up (typ.): - from 0°C to 50°C - from 20°C to 50°C	50 °C 40 sec. 25 sec.

** CAN and I2C interfaces will be implemented in future releases of the firmware*

*** **WARNING:** this version of GPS IMU has small integrated GNSS antenna, so good conditions for reception of GNSS signal is strictly required! If used indoor or in dense urban terrain, the startup time may be much longer the best specified values, or even no fix.*

Installation

When choosing a position for the GPS IMU module, take into account the following conditions that are mandatory for a normal work of all sensors:

Magnetometer: as a magnetic field of the earth is very weak, even relatively small external magnetic fields can distort the measurements. Try to keep magnetometer as far as possible from a soft iron (ferromagnetic metal parts, screws) and a hard iron (permanent magnets). If it's hard to avoid those factors, it's recommended to calibrate magnetometer after installing in a new position. Such calibration is able to compensate affection of the hard iron and soft iron. It's extremely important to keep a magnetometer far away from variable magnetic fields that can't be calibrated: electric motors, power supply cables, ferromagnetic parts that can change their position.

GNSS receiver: keep a clear view of the sky from the top surface of the enclosure for the best GNSS-signal reception; do not place metallic material that can obscure signal, too close to it; keep electronics that emits EMI noise or strong signals that can interfere GPS signal, as far as possible.

Gyroscope and accelerometer: try to minimize the level of vibrations, as they seriously impact these sensors.

Barometer: keep free access of outside air but prevent strong airflows that may create variations of pressure inside the box where sensor is located.

Installing on a gimbal

SimpleBGC32 system supports several mounting positions for the GPS IMU: on the stabilized platform, on the gimbal's arms, or on the outer frame. All possible orientations are supported, with the condition that axes of the device are aligned to axes of the IMU sensor in a normal position.

Choose a position considering requirements for a normal work of all sensors. Take into account that the rotating joints between the external IMU and the stabilized platform may add additional errors caused by the combination of errors in angles measured by the encoders, and flexibility of joints, arms or a suspension of a gimbal.

Connecting to a gimbal and setting up

Please refer to "SimpleBGC32 User Manual 2.6x", section 18 "Using an external IMU sensor to improve the precision of stabilization".

SimpleBGC32 GUI version 2.68b7+ provides a dedicated tool to show diagnostics information for the connected GPS IMU module. Also, from this tool it's possible to make sensor's calibrations and to upgrade firmware.

Sensor calibrations

The basic version of GPS IMU module comes with the simple factory calibrations. You can re-calibrate selected sensors at any time if for some reasons factory calibration becomes obsolete.

IMPORTANT: wait at least 1 minute after powering ON the device, to let it warm up to working temperature! Otherwise, calibrations will be not precise.

Calibrating magnetometer (biases and scales)

It is advised to re-calibrate magnetometer after the installation in a new mounting position or after serious changes in the magnetic environment. Calibration removes the interference of the hard and soft iron located near the device.

1. Press the service button 3 times quickly to start calibration.
2. Rotate device by all directions to collect data in multiple different orientations. This step is divided into 2 phases:
    a) At the first 7-8 seconds system collects a time-separated data points. During this time, it's important to make several long rotations by at least two axes. Green LED flashes with a constant rate.
    b) Then the system collects the remaining number of data points separated by the angle from each other. The short pulse of green LED signals an acceptance of each new point. Rotate device to different angles until the required number of points are collected.
3. When enough points are collected, the system computes calibrations values and emits a series of short pulses by the green LED, which means that the calibration was finished successfully.

Calibrating gyroscope (biases only)

1. Make a single short press on the service button
2. Fix the device firmly, preventing any motion, rotation or vibration. Green LED flashes 2 times per second while the device collects data. It takes several seconds.
3. When finished, system emits a series of short pulses by the green LED.

Calibrating accelerometer (biases and scales)

This procedure needs to place the device in 6 positions in a sequence, making each axis to point exactly up and exactly down. The order of the sequence of positions does not matter.

1. Place the device in the first position in order (for example, Z-axis pointing down) and fix it. A tolerance of 2-3 degrees is allowed.
2. Press the service button 2 times quickly to start calibration in this position.
3. During the calibration, the green LED flashes 2 times per second. When finished, it makes 2 short flashes and device returns to normal operation.
4. Move the device to the next position in order and repeat steps 1-3.

When all 6 positions are passed, system computes calibration values and emits a series of short pulses by the green LED.

LED status indication

The multicolor LED signals the basic modes of operation by color:

- 🟢 Green – GNSS receiver is used in a solution. This is the main working mode when there is a good reception of the GNSS signal.
- 🟡 Yellow – GNSS receiver is not used; device provides attitude and heading information without the compensation of accelerations; position and velocity information is undefined.
- 🔴 Red – hardware fault, device is not operational

When a connection to the external host is established and device sends data, LED flashes 2 times per second.

Service button

- **1 click** – calibrate gyroscope
- **2 clicks** – calibrate accelerometer (single position)
- **3 clicks** – calibrate magnetometer
- **long press** – if connected by USB, enter mass-storage device mode. Device appears as a USB disc allowing to work with the content of the installed SD card. Functions that require access to the SD card (like calibrations, data logging) will be unavailable in this mode.

General configuration parameters

General configuration parameters are located in the "CONFIG.INI" which you can find in the root directory of the SD card.

Name	**Default value**	**Range**	**Units**	**Description**
IMU_HEAT_TEMP	50	-40 – +80	°C	Thermostat temperature for IMU
IMU_HEAT_PWR	500	0 – 1500	mW	Maximum heater power for thermostat that will be consumed from power supply
UART1_BR	115200	9600 – 921600	Bd	Baud rate for UART1
UART1_AHRS_RATE*	0	0 – 200	Hz	Frequency of sending data for AHRS_RATE. Disabled when set to 0.
UART1_HELPER_RATE*	0	0 – 200	Hz	Frequency of sending data for HELPER_RATE. Disabled when set to 0.
UART2_BR	115200	9600 – 921600	Bd	Baud rate for UART2
UTC_TIME_ZONE	0	0 – 24	Hour	UTC time zone
DYNAMIC_MODEL	4	0 – 8		Dynamic platform model:

				0: portable 2: stationary 3: pedestrian 4: automotive 5: sea 6: airborne with <1g acceleration 7: airborne with <2g acceleration 8: airborne with <4g acceleration

Remarks:

* If parameter is enabled, module sends CMD_AHRS_HELPER or CMD_HELPER_DATA messages to the SimpleBGC32 gimbal controller. Use it for compatibility with the old versions of firmware prior to 2.68x that does not support GPS_IMU module natively: GPS_IMU should be mounted on the gimbal's frame and initiate sending data for correction of the gimbal's internal IMU sensor.

Data logging

GPS IMU is able to save realtime data to log files in the on-board SD card in a format SCSV (semicolon-separated values). Up to 2 independent channels can be configured (for example, to log some data at high rates, other data at low rates).

Log configuration parameters are located in the "CONF_LOG.INI" which you can find in the root directory of the SD card, and have the following format:

```
LOG<ch>_<name>=<value>,
```

where

<ch> – channel (1 or 2)
<name> – name of the parameter or name of the data set
<value>– value of the parameter

To select which data to log, set "1" or "2" to the corresponding data sets, for example:

LOG1_GYR_XYZ=0 - disabled
LOG1_GNSS_POS_LLA=1 - enabled, log instant values
LOG1_ACCEL_XYZ=2 - enabled, average values of the variable between log events

The full list of available datasets can be found in the document "Basecam GPS IMU Serial API" .

Additional parameters:

- LOG<ch>_INTERVAL_MS – interval between the data samples, from 5 to 60000 ms.
- LOG<ch>_SYNC_PERIOD_MS – how often new portions of data will be synchronized with the file allocation table (FAT)
  NOTE: do not set SYNC_PERIOD_MS too low, because SD card has a limited lifetime resource for the "write" cycles.
- LOG<ch>_FILES_TO_ROTATE - number of files to keep in rotation scheme, 1..98. When the new log file is created, the oldest file is deleted.

User-defined data logging

Host controller can send user-defined external data that will be logged together with the internal IMU data. To send data, use CMD_USER_DATA_LOG message. To define which data is enabled for logging in configuration, use CMD_GET_USER_CONF_LOG message. The following parameters defines how to log external data:

- LOG_USER_CH<pipe_idx>_NAME=<name_of_pipe>
  - <pipe_idx> starts from 0
  - <name_of_pipe> - name to be logged in the header of log file. If pipe contains more than one value, index will be appended
- LOG<ch>_USER_CH<pipe_idx>_CONF=<data_type>;<data_size>;<is_enabled> - data format for each pipe
  - <ch> - main loggin channel, 1 or 2
  - <data_type>: 1 - float, 2 - int32, 3 - int16
  - <data_size>: number of values in a pipe, 1..15
  - <is_enabled>: 1 if pipe should be logged, 0 to skip it

Example:

```
LOG_USER_CH0_NAME=USER_TIMESTAMP
LOG1_USER_CH0_CONF=3;1;1
LOG_USER_CH1_NAME=IMU_ANGLE
LOG1_USER_CH1_CONF=1;3;1
LOG_USER_CH2_NAME=BATTERY_VOLTAGE
LOG_USER1_CH2_CONF=3;1;1
```

NOTE: when implementing user data logging on a host side, remember that the payload size in the CMD_USER_DATA_LOG message it limited by 255 bytes according to Serial API's specification. You can send exceeding data in a separate message, properly specifying which pipes are included by the ACTIVE_PIPE_MASK parameter.

SBGC32 gimbal controller, used as a host controller, allows logging of its realtime data. More information is provided in Appendix A: Logging data from SBGC32 gimbal controller.

Log file rotation

Log file has format LOG<ch>_<NN>.CSV, where

<ch> - channel
<NN> - sequence number in a rotation queue

Each time device is restarted, NN is selected from the unused numbers (01..99). If the number of files exceeds the maximum allowed (10 by default), file that goes next to the selected NN, is deleted.

Firmware update instructions

WARNING: always load a proper firmware that matches the hardware version of the device! The latest firmware can be downloaded from the www.basecamelectronics.com/Basecam_GPS_IMU/

Updating firmware from SD card

This option works even if the device is unresponsive (i,e, previous attempt of updating firmware was failed and device is not accessible by any interface).

The sequence of actions to update the firmware from SD card:

1. Power off the device.
2. Copy the firmware file named "FWUPDATE.BIN" to the root directory of the SD card and install card into the device.
3. Turn on the device power or connect by USB cable to PC
4. Wait for the firmware update process to finish.

LED indication during the firmware update:

- ⊙ Flashing green: firmware updated is in progress
- ● Solid green: firmware is successfully updated
- ● Solid red: critical error, firmware can not be updated

Updating firmware from the SimpleBGC32 GUI

This option works if the GPS IMU module is connected to the main gimbal controller by the UART interface and is fully functional.

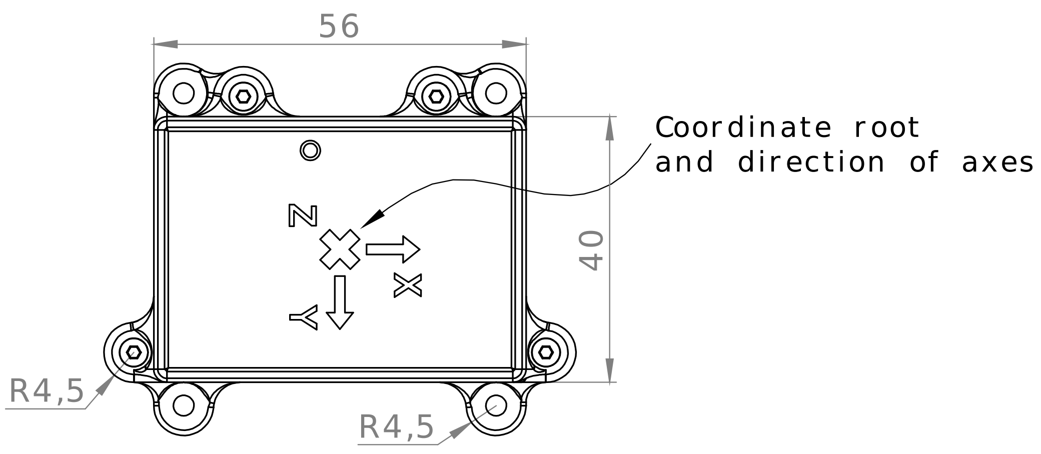
1. Connect the main SimpleBGC32 controller to PC (over USB, bluetooth or any other ways).
2. Run SimpleBGC32 GUI and connect it to the main controller.
3. Open "External IMU" tab and configure Basecam GPS IMU as described in the SimpleBGC32 User Manual (skip this step if it is already configured and works properly)
4. Press the "Show status" button to display information from the device in a new window
5. Press "Firmware update.." button, select *.hex file and press "Open". Firmware update process starts.

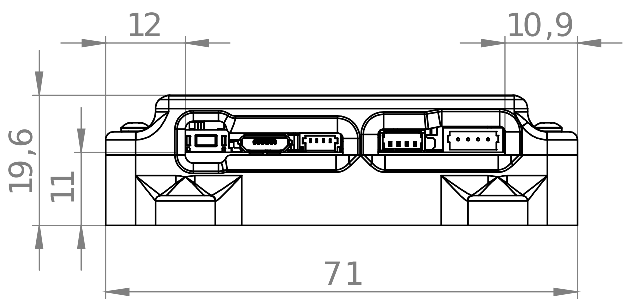
Important note: do not interrupt the updating process - it will make device inaccessible by the UART anymore! You still can use other methods to update the firmware.

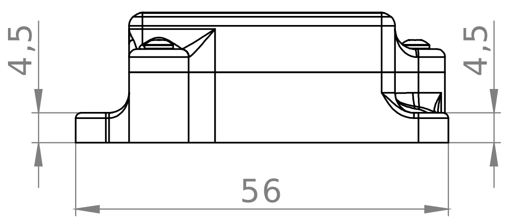
Updating firmware in USB DFU mode

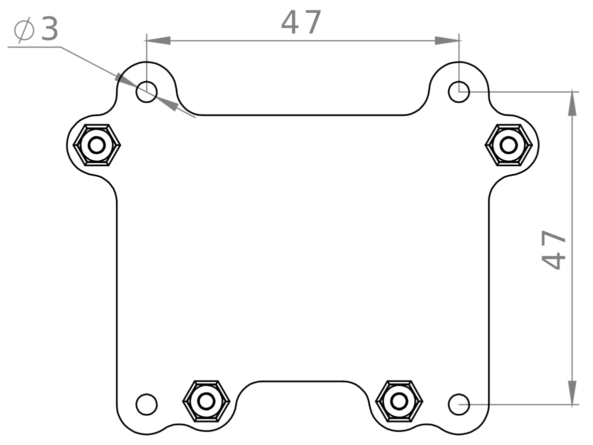
This method works for experienced users only and requires additional steps to enter device into a bootloader mode. Please contact our support for more information.

Dimensions and pinout

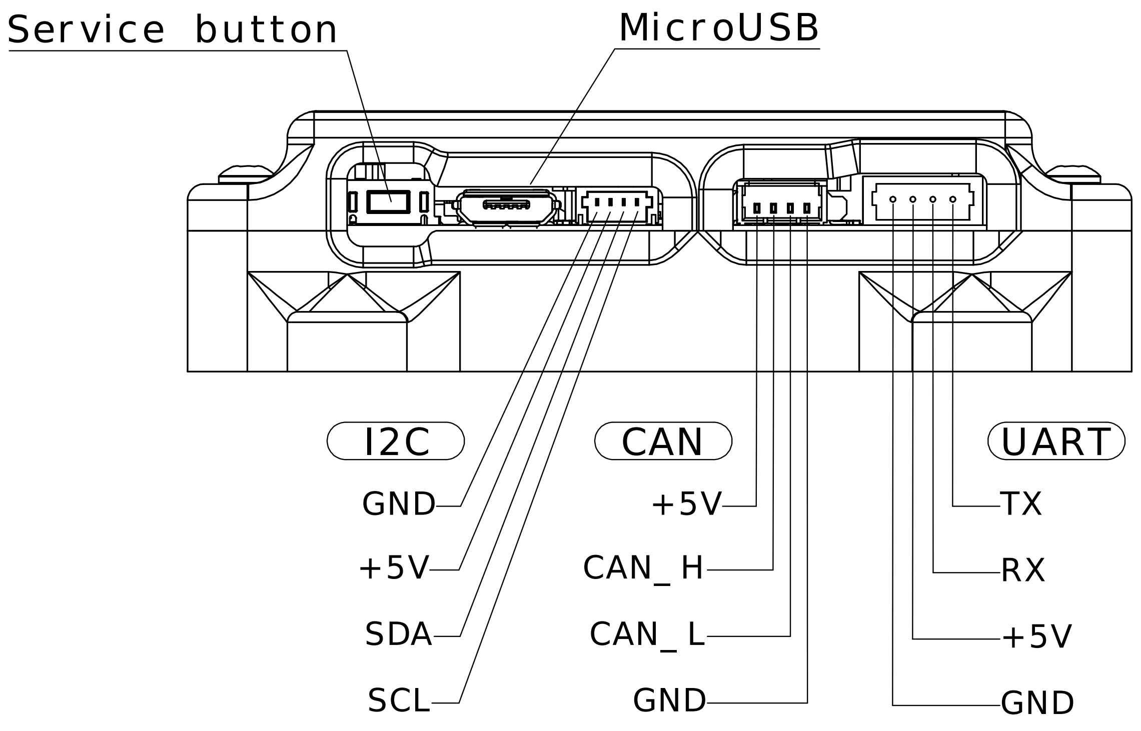








DIMENTIONS IN MILLIMETERS


Service button
MicroUSB
I2C
GND
+5V
SDA
SCL
CAN
+5V
CAN_ H
CAN_ L
GND
UART
TX
RX
+5V
GND

Appendix A: Logging data from SBGC32 gimbal controller

When using together with the SBGC32 gimbal controllers*, it is possible to log realtime data received from it. To enable logging, configure which data to request as described in the section "User-defined data logging". The following table specifies all data available for logging:

data_name	Type	pipe_idx	data_type	data_size	Description
IMU_ANGLE_RAD[3]	float	0	1	3	IMU Euler angles, ROLL-PITCH-YAW *Units: radians*
GYRO_DATA[3]	int16	1	3	3	Data from gyroscope sensor with the calibrations applied, X-Y-Z *Units: 0,06103701895 degree/sec.*
IMU_REF_ERROR[2]	float	2	1	2	Module of error vector between the actual unit vector and the reference unit vector, for attitude and heading corrections in sensor fusion algorithm.
JOINT_ANGLE[3]	int16	3	3	3	Relative angle of joints between two arms of gimbal structure, in order ROLL-PITCH-YAW. Value 0 corresponds to "home" position. *Units: 0,02197265625 degree*
Z_VECTOR[3] H_VECTOR[3]	float	4	1	6	IMU attitude/heading in a form of gravity and heading vectors (allows to reconstruct a rotation matrix, DCM).
IMU_ANGLE[3]	int16	5	3	3	IMU Euler angles, ROLL-PITCH-YAW *Units: 0,02197265625 degree.*
TARGET_ANGLE[3]	int16	6	3	3	Setpoint angles (Euler ROLL-PITCH-YAW) *Units: 0,02197265625 degree.*
TARGET_RATE[3]	int16	7	3	3	Commanded rate (over Euler axes ROLL-PITCH-YAW) *Units: 0,06103701895 degree/sec*
ACC_DATA[3]	int16	8	3	3	Data from the accelerometer sensor with the calibrations applied. *Units: 1/512 G*
RC_DATA[6]	int16	9	3	6	RC signal value assigned to the ROLL, PITCH, YAW, CMD, FC_ROLL, FC_PITCH channels *Units: normal range is -16384..16384, -32768 is for 'undefined' signal*
RC_CHANNELS[15]	int16	10	3	15	Raw RC signal value from multi-channel RC input (sum-ppm, s-bus, spektrum, serial api). *Units: normal range is -16384..16384, -32768 is for 'undefined' signal*
MOTOR_OUTPUT	int16	11	3	3	Instant motor output, proportional to torque, where ±32767 corresponds to 100% (ROLL-PITCH-YAW order)
TEMP_SENS	int16	12	3	7	Temperature of IMU, Frame IMU, main MCU, on-board temp. sensor, ext. motor temp sensors (ROLL, PITCH, YAW). *Units: °C*
SYSTEM_ERRORS	int16	13	3	6	Array of system errors: 1: Error flags 2: Emergency stop error sub-code 3: I2C errors counter 4: CAN bus errors counter 5: CAN bus error flags: (bit0: warn irq, bit1: passive irq, bit2: off irq) 6: COM port errors counter

TMP_VARS_F	float	14	1	10	Temporarily variables used in scripts
TMP_VARS_16	int16	15	3	10	Temporarily variables used in scripts, rounded to 16bit integer

** This function is supported in the "extended" versions of controller only, frw. ver. 2.69b7+*

Configuration of SBGC32 data in CONG_LOG.INI:

```
LOG_USER_CH0_NAME=IMU_ANGLE_RAD
LOG_USER_CH1_NAME=GYRO_DATA
LOG_USER_CH2_NAME=IMU_REF_ERROR
LOG_USER_CH3_NAME=JOINT_ANGLE
LOG_USER_CH4_NAME=Z_H_VECT
LOG_USER_CH5_NAME=IMU_ANGLE
LOG_USER_CH6_NAME=TARGET_ANGLE
LOG_USER_CH7_NAME=TARGET_RATE
LOG_USER_CH8_NAME=ACC_DATA
LOG_USER_CH9_NAME=RC_DATA
LOG_USER_CH10_NAME=RC_CHANNELS
LOG_USER_CH11_NAME=MOTOR_OUTPUT
LOG_USER_CH12_NAME=TEMP_SENS
LOG_USER_CH13_NAME=SYSTEM_ERRORS
LOG_USER_CH14_NAME=TMP_VAR_F
LOG_USER_CH15_NAME=TMP_VAR
...
# All pipes are disabled, change trailing 0 to 1 to enable only required pipes
LOG1_USER_CH0_CONF=1;3;0
LOG1_USER_CH1_CONF=3;3;0
LOG1_USER_CH2_CONF=1;2;0
LOG1_USER_CH3_CONF=3;3;0
LOG1_USER_CH4_CONF=1;6;0
LOG1_USER_CH5_CONF=3;3;0
LOG1_USER_CH6_CONF=3;3;0
LOG1_USER_CH7_CONF=3;3;0
LOG1_USER_CH8_CONF=3;3;0
LOG1_USER_CH9_CONF=3;6;0
LOG1_USER_CH10_CONF=3;15;0
LOG1_USER_CH11_CONF=3;3;0
LOG1_USER_CH12_CONF=3;7;0
LOG1_USER_CH13_CONF=3;6;0
LOG1_USER_CH14_CONF=1;10;0
LOG1_USER_CH15_CONF=3;10;0
```