SimpleBGC 2.5 serial protocol specification

Applicable for 32-bit boards with firmware 2.5x

Revision history

- rev. 0.1 - 24.03.2015: this is first revision
- rev. 0.2 – 27.03.2015: add missed data
- rev. 0.3 – 30.04.2015: add missed data in CMD_READ_PARAMS_EXT
- rev. 0.4 – 01.07.2015: CMD_CONTROL extended format; add MENU_CMD_LEVEL_ROLL_PITCH; FRAME_ANGLE_XX replaced by ROTOR_ANGLE_XX in the CMD_REALTIME_DATA_4; CMD_AHRS_HELPER updated;
- rev. 0.5 – 30.07.2015: PROFILE_FLAGS1, GENERAL_FLAGS1 set is extended; CMD_EXECUTE_MENU set is extended; FRAME_CAM_ANGLE_XX is deprecated;
- rev. 0.6 – 12.08.2015: new mode in the CMD_CONTROL: MODE_ANGLE_REL_FRAME; new commands CMD_GET_ANGLES_EXT, CMD_SET_ADJ_VARS_VAL;
- rev. 0.7 – 22.10.2015: new config parameters ORDER_OF_AXES, EULER_ORDER; set of PROFILE_FLAGS1, GENERAL_FLAGS1 extended; SKIP_GYRO_CALIB options extended;
- rev. 0.8 – 09.11.2015: CMD_AHRS_HELPER is extended;
- rev. 0.9 – 22.12.2015: new command CMD_GYRO_CORRECTION; list of adjustable variables was extended by the FRAME_HEADING_ANLGE, GYRO_HEADING_CORRECTION; GENERAL_FLAGS1, PROFILE_FLAGS1 set was extended;
- rev. 0.10 – 13.02.2016: CMD_AUTO_PID updated; NOTCH_GAIN range extended;
- rev. 0.11 – 07.03.2016: new command CMD_READ_PARAMS_EXT2; new parameter MOTOR_MAG_LINK_FINE; new command CMD_CALIB_MOTOR_MAG_LINK; ACC_LIMITER split to axes; extended form of CMD_HELPER_DATA;
- rev. 0.12 – 02.04.2016: new commands CMD_DATA_STREAM_INTERVAL, CMD_REALTIME_DATA_CUSTOM;

Overview

Serial API allows external application or device to communicate with the SimpleBGC controller via UART port. Each controller has one or more UART ports that can be used to send and receive Serial API commands. Commands may be used to retrieve actual system state and realtime data, change settings, control gimbal, trigger pin state, execute various actions, get access to internal EEPROM and I2C bus, and so on. Moreover, SimpleBGC GUI software uses the same Serial API to communicate with the board, so all of its functions may be implemented in third-party applications.

Message format

Communications is initiated from the GUI side (host) by sending *outgoing* commands. The controller board may do some action and send response (further named as *incoming* commands). Each command consists of the *header* and the *body*, both with checksum. Commands with the wrong header or body checksum, or with the body size that differs from expected, should be ignored.

Board can work on different serial baud rate, so the GUI should find proper baud rate by sending CMD_BOARD_INFO command on every speed ant wait for response, until valid response is received.

32bit boards with firmware version 2.40, works only with parity=EVEN COM-port setting. Starting from 2.41, both EVEN and NONE parity are supported (NONE is default, and EVEN is detected automatically). So beside baud rates, host should vary parity setting when connecting to boards ver.>3.0

Make a small delay after sending each command to prevent overflow of the input buffer. Delay should be about 10-20 ms, and depends on the size of the request and response. If new serial data comes when the input buffer is full, whole message will be lost. There is also a control of overflow of the output buffer on the board's side: if it have to write an answer to the output buffer, it hangs until buffer will have enough space to accept new data. If requests comes with too big rate, it may negatively affect normal operation of the board and impact stabilization.

Input and output commands have the same format, described below:

Header:

character '>'
command ID - 1u
data_size – 1u, may be zero
header checksum = (command ID + data_size) modulo 256 - 1u

Body:

[array of bytes *data_size* length]
body checksum - 1u

Checksum is calculated as a sum of all bytes modulo 256.

Example: outgoing command to read Profile2:

0x3E (>)	0x52 (R)	0x01	0x53	0x01	0x01
	command id	data size	header checksum	data	body checksum
header				body	

Data type notation

- 1u – 1 byte unsigned

- 1s – 1 byte signed
- 2u – 2 byte unsigned (little-endian order)
- 2s – 2 byte signed (little-endian order)
- 4f – float (IEEE-754 standard)
- 4s – 4 bytes signed (little-endian order)
- string – ASCII character array, first byte is array size
- Nb – byte array size N

Command ID definitions

```
#define CMD_READ_PARAMS  82
#define CMD_WRITE_PARAMS  87
#define CMD_REALTIME_DATA  68
#define CMD_BOARD_INFO  86
#define CMD_CALIB_ACC  65
#define CMD_CALIB_GYRO  103
#define CMD_CALIB_EXT_GAIN  71
#define CMD_USE_DEFAULTS  70
#define CMD_CALIB_POLES  80
#define CMD_RESET  114
#define CMD_HELPER_DATA 72
#define CMD_CALIB_OFFSET  79
#define CMD_CALIB_BAT  66
#define CMD_MOTORS_ON   77
#define CMD_MOTORS_OFF  109
#define CMD_CONTROL   67
#define CMD_TRIGGER_PIN  84
#define CMD_EXECUTE_MENU 69
#define CMD_GET_ANGLES  73
#define CMD_CONFIRM  67

// Board v3.x only
#define CMD_BOARD_INFO_3  20
#define CMD_READ_PARAMS_3 21
#define CMD_WRITE_PARAMS_3 22
#define CMD_REALTIME_DATA_3  23
#define CMD_REALTIME_DATA_4  25
#define CMD_SELECT_IMU_3 24
#define CMD_READ_PROFILE_NAMES 28
#define CMD_WRITE_PROFILE_NAMES 29
#define CMD_QUEUE_PARAMS_INFO_3 30
#define CMD_SET_ADJ_VARS_VAL 31
#define CMD_SAVE_PARAMS_3 32
#define CMD_READ_PARAMS_EXT 33
#define CMD_WRITE_PARAMS_EXT 34
#define CMD_AUTO_PID 35
#define CMD_SERVO_OUT 36
#define CMD_I2C_WRITE_REG_BUF 39
#define CMD_I2C_READ_REG_BUF 40
#define CMD_WRITE_EXTERNAL_DATA 41
#define CMD_READ_EXTERNAL_DATA 42
#define CMD_READ_ADJ_VARS_CFG 43
#define CMD_WRITE_ADJ_VARS_CFG 44
#define CMD_API_VIRT_CH_CONTROL 45
#define CMD_ADJ_VARS_STATE 46
#define CMD_EEPROM_WRITE 47
#define CMD_EEPROM_READ 48
#define CMD_BOOT_MODE_3 51
#define CMD_SYSTEM_STATE 52
```

```
#define CMD_READ_FILE 53
#define CMD_WRITE_FILE 54
#define CMD_FS_CLEAR_ALL 55
#define CMD_AHRS_HELPER 56
#define CMD_RUN_SCRIPT 57
#define CMD_SCRIPT_DEBUG 58
#define CMD_CALIB_MAG 59
#define CMD_GET_ANGLES_EXT 61
#define CMD_READ_PARAMS_EXT2 62
#define CMD_WRITE_PARAMS_EXT2 63
#define CMD_GET_ADJ_VARS_VAL 64
#define CMD_CALIB_MOTOR_MAG_LINK 74
#define CMD_GYRO_CORRECTION 75
#define CMD_DATA_STREAM_INTERVAL 85
#define CMD_REALTIME_DATA_CUSTOM 88
#define CMD_DEBUG_VARS_INFO_3 253
#define CMD_DEBUG_VARS_3 254
#define CMD_ERROR  255
```

Incoming commands

CMD_BOARD_INFO – version and board info information

- BOARD_VER - 1u (split into decimal digits X . X, for example 10 means 1.0)
- FIRMWARE_VER - 2u (split into decimal digits X . XX . X, for example 2305 means 2.30b5)
- DEBUG_MODE - 1u (should hide DEBUG output if DEBUG_MODE = 0)
- BOARD_FEATURES – 2u
- CONNECTION_FLAGS – 1u
- FRW_EXTRA_ID - 4u
- reserved – 7b

CMD_BOARD_INFO_3 – additional board information

- deviceID 9b – device ID
- mcuID 12b - MCU ID
- EEPROM_SIZE – 4u
- SCRIPT_SLOT1_SIZE – 2u – size of user-written scripts stored in each slot, 0 if slot is empty.
  SCRIPT_SLOT2_SIZE – 2u
  SCRIPT_SLOT3_SIZE – 2u
  SCRIPT_SLOT4_SIZE – 2u
  SCRIPT_SLOT5_SIZE - 2u
- reserved - 34b

CMD_READ_PARAMS_3 – Receive parameters

Receive parameters for single profile together with general parameters .

Profile parameters:

- PROFILE_ID – 1u (ID of profile to read, starting from 0)
- for(axis in [ROLL, PITCH, YAW]) {
  - P - 1u
  - I - 1u (multiplied by 100)
  - D - 1u
  - POWER - 1u
  - INVERT – 1u (checked=1, not checked=0)
  - POLES - 1u
- }
- ACC_LIMITER_ALL - 1u
- EXT_FC_GAIN_ROLL - 1s

- EXT_FC_GAIN_PITCH – 1s
-
- for(axis in [ROLL, PITCH, YAW]) {
    - RC_MIN_ANGLE - 2s
    - RC_MAX_ANGLE - 2s
    - RC_MODE - 1u
    - RC_LPF – 1u
    - RC_SPEED – 1u
    - RC_FOLLOW - 1u
- }
- GYRO_TRUST – 1u
- USE_MODEL – 1u
- PWM_FREQ – 1u
- SERIAL_SPEED – 1u
- RC_TRIM_ROLL - 1s
- RC_TRIM_PITCH - 1s
- RC_TRIM_YAW - 1s
- RC_DEADBAND - 1u
- RC_EXPO_RATE - 1u
- RC_VIRT_MODE – 1u
-
- RC_MAP_ROLL – 1u
- RC_MAP_PITCH – 1u
- RC_MAP_YAW – 1u
- RC_MAP_CMD – 1u
- RC_MAP_FC_ROLL – 1u
- RC_MAP_FC_PITCH – 1u
-
- RC_MIX_FC_ROLL - 1u
- RC_MIX_FC_PITCH - 1u
-
- FOLLOW_MODE – 1u
- FOLLOW_DEADBAND – 1u
- FOLLOW_EXPO_RATE – 1u
- FOLLOW_OFFSET_ROLL – 1s

- FOLLOW_OFFSET_PITCH – 1s
- FOLLOW_OFFSET_YAW - 1s
-
- AXIS_TOP – 1s
- AXIS_RIGHT – 1s
- FRAME_AXIS_TOP – 1s
- FRAME_AXIS_RIGHT – 1s
- FRAME_IMU_POS - 1u
- GYRO_LPF – 1u
- GYRO_SENS - 1u
- I2C_INTERNAL_PULLUPS – 1u
- SKIP_GYRO_CALIB – 1u
-
- RC_CMD_LOW – 1u
- RC_CMD_MID – 1u
- RC_CMD_HIGH – 1u
-
- MENU_CMD_1 - 1u
- MENU_CMD_2 - 1u
- MENU_CMD_3 - 1u
- MENU_CMD_4 - 1u
- MENU_CMD_5 - 1u
- MENU_CMD_LONG - 1u
-
- OUTPUT_ROLL - 1u
- OUTPUT_PITCH – 1u
- OUTPUT_YAW – 1u
-
- BAT_THRESHOLD_ALARM – 2s
- BAT_THRESHOLD_MOTORS – 2s
- BAT_COMP_REF – 2s
-
- BEEPER_MODES – 1u
-
- FOLLOW_ROLL_MIX_START - 1u

- FOLLOW_ROLL_MIX_RANGE - 1u
-
- BOOSTER_POWER_ROLL - 1u
- BOOSTER_POWER_PITCH - 1u
- BOOSTER_POWER_YAW - 1u
-
- FOLLOW_SPEED_ROLL - 1u
- FOLLOW_SPEED_PITCH - 1u
- FOLLOW_SPEED_YAW - 1u
-
- FRAME_ANGLE_FROM_MOTORS - 1u
-
- RC_MEMORY_ROLL – 2s
- RC_MEMORY_PITCH – 2s
- RC_MEMORY_YAW – 2s
-
- SERVO1_OUT – 1u
- SERVO2_OUT – 1u
- SERVO3_OUT – 1u
- SERVO4_OUT – 1u
- SERVO_RATE – 1u
-
- ADAPTIVE_PID_ENABLED – 1u
- ADAPTIVE_PID_THRESHOLD – 1u
- ADAPTIVE_PID_RATE – 1u
- ADAPTIVE_PID_RECOVERY_FACTOR – 1u
-
- FOLLOW_LPF_ROLL – 1u
- FOLLOW_LPF_PITCH – 1u
- FOLLOW_LPF_YAW – 1u
-
- GENERAL_FLAGS1 – 2u
- PROFILE_FLAGS1 - 2u
- SPEKTRUM_MODE - 1u
-

- ORDER_OF_AXES – 1b
- EULER_ORDER - 1b
-
- CUR_IMU - 1u (currently selected IMU)
- CUR_PROFILE_ID – 1u (profile ID which is currently active in the controller)

CMD_READ_PARAMS_EXT – read extended set of params for

- PROFILE_ID – 1u (ID of profile to read, starting from 0)
- for(1..3) {
  - NOTCH_FREQ[3] – 1u * 3
  - NOTCH_WIDTH[3] – 1u * 3
- }
- LPF_FREQ[3] – 2u * 3
- FILTERS_EN[3] – 1u * 3
- ENCODER_OFFSET[3] – 2s * 3
- ENCODER_FLD_OFFSET[3] – 2s * 3
- ENCODER_MANUAL_SET_TIME[3] – 1u * 3
- MOTOR_HEATING_FACTOR[3] - 1u * 3
- MOTOR_COOLING_FACTOR[3] – 1u * 3
- RESERVED – 2b
- FOLLOW_INSIDE_DEADBAND - 1u
- MOTOR_MAG_LINK[3] – 1u * 3 (deprecated, replaced by MOTOR_MAG_LINK_FINE)
- MOTOR_GEARING[3] – 2u * 3
- ENCODER_LIMIT_MIN[3] – 1s * 3
- ENCODER_LIMIT_MAX[3] – 1s * 3
- NOTCH1_GAIN[3] – 1s * 3
- NOTCH2_GAIN[3] – 1s * 3
- NOTCH3_GAIN[3] – 1s * 3
-
- BEEPER_VOLUME – 1u
- ENCODER_GEAR_RATIO[3] – 2u * 3
- ENCODER_TYPE[3] – 1u * 3
- ENCODER_CFG[3] – 1u * 3
- OUTER_P[3] – 1u * 3
- OUTER_I[3] – 1u * 3

- MAG_AXIS_TOP – 1s
- MAG_AXIS_RIGHT – 1s
- MAG_TRUST – 1u
- MAG_DECLINATION – 1s
- ACC_LPF_FREQ – 2u
- D_TERM_LPF_FREQ[3] – 1u * 3

CMD_READ_PARAMS_EXT2 – read extended set of parameters

- RESERVED – 16b
- MOTOR_MAG_LINK_FINE[3] – 2u * 3
- ACC_LIMITER3[3] – 1u * 3
- RESERVED - 125b

CMD_REALTIME_DATA_3 - receive real-time data for

- for(axis in [ROLL, PITCH, YAW]) {
  - ACC – 2s
  - GYRO – 2s
- }
- SERIAL_ERROR_CNT – 2u
- SYSTEM_ERROR – 2u
- SYSETEM_SUB_ERROR – 1u
- RESERVED - 3b
- RC_ROLL - 2s
- RC_PITCH - 2s
- RC_YAW - 2s
- RC_CMD – 2s
- EXT_FC_ROLL – 2s
- EXT_FC_PITCH – 2s
- ANGLE_ROLL – 2s
- ANGLE_PITCH – 2s
- ANGLE_YAW – 2s
- FRAME_IMU_ANGLE_ROLL – 2s
- FRMAE_IMU_ANGLE_PITCH – 2s
- FRAME_IMU_ANGLE_YAW – 2s

- RC_ANGLE_ROLL - 2s
- RC_ANGLE_PITCH - 2s
- RC_ANGLE_YAW - 2s
- CYCLE_TIME - 2u
- I2C_ERROR_COUNT - 2u
- ERROR_CODE – 1u (deprecated, use 16bit SYSTEM_ERROR above)
- BAT_LEVEL - 2u
- OTHER_FLAGS - 1u
- CUR_IMU - 1u
- CUR_PROFILE – 1u
- MOTOR_POWER_ROLL – 1u
- MOTOR_POWER_PITCH – 1u
- MOTOR_POWER _YAW- 1u

CMD_REALTIME_DATA_4 - receive extended real-time data

- **..all data from CMD_REALTIME_DATA_3..**
- ROTOR_ANGLE[3] – 2s*3
- RESERVED – 1b
- BALANCE_ERROR[3] – 2s*3
- CURRENT – 2u (units: mA)
- MAG_DATA[3] – 2s*3
- IMU_TEMPERATURE – 1s (units: Celsius)
- FRAME_IMU_TEMPERATURE – 1s (units: Celsius)
- IMU_G_ERR – 1u
- IMU_H_ERR - 1u
- RESERVED - 36b

CMD_CONFIRM – confirmation of previous command

- CMD – 1u
- DATA – depends on CMD

Board sends confirmation on commands: A, G, P, W, etc. DATA is empty unless mentioned in command description.

CMD_ERROR – error on executing previous command

- ERROR_CODE – 1u

- ERROR_DATA – 4b

Data depends on error type.

CMD_GET_ANGLES - Information about actual RC control state

- for(axis in [ROLL, PITCH, YAW]) {
    - IMU_ANGLE - 2s
    - RC_TARGET_ANGLE - 2s
    - RC_SPEED - 2s
- }

CMD_GET_ANGLES_EXT - Information about angles in different format

- for(axis in [ROLL, PITCH, YAW]) {
    - IMU_ANGLE - 2s
    - RC_TARGET_ANGLE - 2s
    - STATOR_ROTOR_ANGLE – 4s
    - RESERVED - 10b
- }

CMD_READ_PROFILE_NAMES_3 – receive profile names from EEPROM
Each name is encoded in UTF-8 format and padded with '\0' character to 48 byte size

- PROFILE1_NAME – 48b
- PROFILE2_NAME – 48b
- PROFILE3_NAME – 48b
- PROFILE4_NAME – 48b
- PROFILE5_NAME – 48b

CMD_GET_PARAMS_3 – receive information about configurable parameters: type, range, etc.
--not yet implemented--

CMD_I2C_READ_REG_BUF – result of reading from I2C device

- DATA – 1..255 byte, depends on the DATA_LEN parameter in the request.

CMD_AUTO_PID – progress of PID auto tuning

- P[3] – 1u * 3
- I[3] – 1u * 3
- D[3] – 1u * 3
- LPF_FREQ[3] – 2u * 3
- ITER_NUM - 2u
- for(1..3) {
    - TRACKING_ERROR – float
    - RESERVED – 6b
- }
- RESERVED – 10b

CMD_DEBUG_VARS_INFO_3 – receive specification of the debug variables

- DEBUG_VARS_NUM – 1u - number of debug vars

for(i=0; i<DEBUG_VARS_NUM; i++) {
- VAR_NAME – string
- VAR_TYPE – 1u (see definitions below)
- RESERVED – 2b

}

CMD_DEBUG_VARS_3 – values of some variables reflecting a state of the system.
A set and an order of variables is not strictly defined, and may vary depending on the firmware version. Use *CMD_DEBUG_VARS_INFO_3* to get a specification of the variables.

for(i=0; i<DEBUG_VARS_NUM; i++) {
- VAR_VALUE – <size and type from CMD_DEBUG_VARS_INFO_3 structure>

}

CMD_READ_EXTERNAL_DATA – receive user data, stored in the EEPROM
- data – 128b

CMD_SET_ADJ_VARS_VAL – receive the values of adjustable variables.
See corresponding outgoing command for format description.

CMD_READ_ADJ_VARS_CFG – receive the configuration of mapping of control inputs to adjustable variables
There are 10 “trigger” slots and 15 “analog” slots. “Trigger” type is used to execute action depending on the RC signal level, where full range is split into 5 levels (see Available actions). “Analog” type is used to adjust parameter by RC signal. MIN_VAL and MAX_VAL specify a working range, that is combined with the native range of particular parameter (see List of available parameters)

for(i=0; i<10; i++) {
- SRC_CH – 1u
- ACTION1 – 1u
- ACTION2 – 1u
- ACTION3 – 1u
- ACTION4 – 1u
- ACTION5 – 1u

}
for(i=0; i<15; i++) {
- SRC_CH – 1u
- PARAM_ID – 1u
- MIN_VAL – 1u
- MAX_VAL – 1u

}
- RESERVED – 8b

CMD_RESET – notification on device reset

Device sent this command when goes to reset. There is a delay 1000ms after this command is sent and reset is actually done. External application can free up resources and properly close the serial connection.

CMD_EEPROM_READ – receive block of data from EEPROM at the specified address.
- ADDR – 4u, 64-byte aligned
- DATA – any size, as specified in the CMD_EEPROM_READ outgoing command.

CMD_READ_FILE – result of reading file from internal filesystem
In case of success:

- FILE_SIZE – 2u – total size of file, bytes
- PAGE_OFFSET – 2u – offset that was requested, in pages. 1 page = 64 bytes
- DATA – size that was requested, or less if end of file is reached

In case of errors:

- ERR_CODE – 1u (see error definitions in the CMD_WRITE_FILE command)

CMD_SCRIPT_DEBUG – state of execution of user-written script

- CMD_COUNT – 2u – current command counter
- ERR_CODE – 1u (see error definitions in the CMD_WRITE_FILE command)

CMD_AHRS_HELPER – current attitude in vector form.

- Z1_VECTOR[3] – 4f * 3
- H1_VECTOR[3] – 4f * 3

CMD_REALTIME_DATA_CUSTOM – configurable realtime data (ver. 2.59+)

- TIMESTAMP_MS – 2u
- DATA – variable length, depends on request. See specification below.

Outgoing command

CMD_BOARD_INFO – request board and firmware information

Simple format: no parameters

Extended format:

- CFG – 2b - configuration for this serial driver:
  - for UARTs – period (in ms) between 20-bytes packets for BLE mode
  - for USB – not used
- RESERVED – size undefined

CMD_BOARD_INFO_3 – request additional board information

CMD_REALTIME_DATA,
CMD_REALTIME_DATA_3 – request real-time data, response is CMD_REALTIME_DATA_3

CMD_REALTIME_DATA_4 – request extended real-time data, response is CMD_REALTIME_DATA_4

CMD_CALIB_ACC – calibrate accelerometer
CMD_CALIB_GYRO – calibrate gyroscope

Simple format: no parameters. Starts regular calibration of currently active IMU (set by `CMD_SELECT_IMU_3 command`)

Extended format (for both commands):

- IMU_IDX – 1u (0 – currently active IMU, 1 – main IMU, 2 – frame IMU)
- ACTION – 1u
  1 – do regular calibration
  2 – reset all calibrations and restart
  3 – do temperature calibration
  4 – enable temp. calib. data, if present and restart
  5 – disable temp. calib. data (but keep in memory) and restart
  6 – copy calibration from the sensor's EEPROM to the main EEPROM ("restore factory calibration" option)
  7 – copy calibration from the main EEPROM to the sensor's EEPROM
- RESERVED - 10b

If all parameters are valid, confirmation is sent immediately on reception and in the end of calibration.

CMD_CALIB_EXT_GAIN – calibrate EXT_FC gains

CMD_USE_DEFAULTS – reset to factory defaults

- PROFILE_ID – 1u – profile to reset, 0..NUM_PROFILE-1
  Special values:
  253 – erase EEPROM

254 – reset current profile

CMD_CALIB_POLES – calibrate poles and direction

CMD_READ_PARAMS,
CMD_READ_PARAMS_3 – request parameters from the board
CMD_READ_PARAMS_EXT – request extended parameters
CMD_READ_PARAMS_EXT2 – request extended parameters

- PROFILE_ID – 1u – profile to load

CMD_WRITE_PARAMS,
CMD_WRITE_PARAMS_3 - write parameters to board and saves to EEPROM
CMD_WRITE_PARAMS_EXT – write extended parameters
Data structure is the same as for corresponding CMD_READ_PARAMS_xx incoming command.

CMD_RESET – reset device

Simple format: reset device without delay and confirmation

Extended format:

- CONFIRM – 1u (0 – no confirmation, 1 - command CMD_RESET will be sent back)
- DELAY_MS – 2u - delay before reset, in ms. External application can free up resources and properly close the serial connection.

CMD_BOOT_MODE_3 – enter bootloader mode to upload firmware

Simple format: enter without delay and confirmation

Extended format:

- CONFIRM – 1u (0 – no confirmation, 1 - command CMD_RESET will be sent back)
- DELAY_MS – 2u - delay before entering bootloader mode, in ms.

CMD_CALIB_OFFSET – calibrate follow offset

CMD_CALIB_BAT - calibrate battery (voltage sensor)
- ACTUAL_VOLTAGE - 2u

CMD_CONTROL – control gimbal movement
- CONTROL_MODE – 1u
- SPEED_ROLL – 2s
- ANGLE_ROLL – 2s
- SPEED_PITCH – 2s
- ANGLE_PITCH – 2s
- SPEED_YAW – 2s
- ANGLE_YAW – 2s

Extended format (firmware ver. 2.55b5): mode is set independently for each axes, that allows to have RC control mixed with serial control, or different control modes for different axes:
- CONTROL_MODE_ROLL – 1u

CONTROL_MODE_PITCH – 1u
CONTROL_MODE_YAW – 1u
- SPEED_ROLL – 2s
- ANGLE_ROLL – 2s
- SPEED_PITCH – 2s
- ANGLE_PITCH – 2s
- SPEED_YAW – 2s
- ANGLE_YAW – 2s

CMD_TRIGGER_PIN - trigger output pin
- PIN_ID - 1u
- STATE - 1u

Confirmation is sent only if pin is not used for input and is really triggered.

CMD_MOTORS_ON - switch motors ON
Confirmation send 'M'

CMD_MOTORS_OFF - switch motors OFF
Confirmation send 'm'

CMD_EXECUTE_MENU - execute menu command
- CMD_ID - 1u

CMD_HELPER_DATA – pass helper data
- FRAME_ACC_X – 2s
- FRAME_ACC_Y – 2s
- FRAME_ACC_Z – 2s
- FRAME_ANGLE_ROLL – 2s
- FRAME_ANGLE_PITCH – 2s

Extended form supported in 2.59+ firmware:
- FRAME_ACC[3] – 2s * 3
- FRAME_ANGLE_ROLL – 2s
- FRAME_ANGLE_PITCH – 2s
- COORD_SYS – 1u
- FRAME_SPEED[3] – 2s * 3
- RESERVED – 3b

CMD_GET_ANGLES, CMD_GET_ANGLES_EXT - Request information about angles and RC control state
See description for incoming command.

CMD_SELECT_IMU_3 – Select which IMU to configure
- IMU_TYPE – 1u

CMD_READ_PROFILE_NAMES_3 – Request profile names stored in EEPROM

CMD_WRITE_PROFILE_NAMES_3 – Writes profile names to EEPROM
Each name is encoded in UTF-8 format and padded with '\0' character to 48 byte size
- PROFILE1_NAME – 48b
- PROFILE2_NAME – 48b

- PROFILE3_NAME – 48b
- PROFILE4_NAME – 48b
- PROFILE5_NAME – 48b

CMD_GET_PARAMS_3 – Request information about configurable parameters: type, range, current value
In response, board may send multiple CMD_GET_PARAMS_3 commands if all data will not fit to single command.
--not yet implemented--

CMD_SET_ADJ_VARS_VAL – Update the value of selected parameter(s).
This command is intended to change parameters on-the-fly during system operation, and does not save parameters to EEPROM. You need to send CMD_SAVE_PARAMS_3 to do this. List of available parameters .

- NUM_VARS - 1u
- PARAM1_ID – 1u
- PARAM1_VALUE – 4s
- PARAM2_ID – 1u
- PARAM2_VALUE – 4s

...repeat for remaining parameters...
On success, confirmation is sent in response.

CMD_GET_ADJ_VARS_VAL – Query the actual value of selected parameter(s).
This command requests actual values of adjustable parameters. List of available parameters .

- NUM_VARS - 1u
- PARAM1_ID – 1u
- PARAM2_ID – 1u

...repeat for remaining parameters...
On success, CMD_SET_ADJ_VARS_VAL is sent in response.

CMD_SAVE_PARAMS_3 – **Saves current params from volatile memory to EEPROM, to the active profile slot.**

CMD_AUTO_PID – **Starts automatic PID calibration**

- PROFILE_ID – 1u - switch to this profile before start of calibration
- CFG_FLAGS – 1u
- GAIN_VS_STABILITY – 1u
- RESERVED - 16b

CMD_SERVO_OUT – **Output PWM signal on the specified pins**
Although it takes 8 values, the real number of hardware outputs depends on board version and may be less.

- SERVO1_TIME – 2s - shared with FC_ROLL
- SERVO2_TIME – 2s - shared with FC_PITCH
- SERVO3_TIME – 2s - shared with RC_PITCH
- SERVO4_TIME – 2s - shared with AUX1
- SERVO5_TIME – 2s - reserved
- SERVO6_TIME – 2s - reserved
- SERVO7_TIME – 2s - reserved
- SERVO8_TIME – 2s - reserved

CMD_I2C_WRITE_REG_BUF – writes data to any device connected to I2C line

- DEVICE_ADDR – 1u

  bit0: I2C port: 0 for main (sensor) port, 1 for second (EEPROM) port
  bit1..7: address

- REG_ADDR – 1u
- DATA – remaining bytes

On successful writing, confirmation CMD_CONFIRM is sent in response.

CMD_I2C_READ_REG_BUF – requests reading from any device connected to I2C line
Meaning of parameters are the same as for CMD_I2C_WRITE_REG_BUF command.
- DEVICE_ADDR – 1u
- REG_ADDR – 1u
- DATA_LEN – 1u

On successful reading, CMD_I2C_READ_REG_BUF command is sent in response.

CMD_DEBUG_VARS_INFO_3 – request information about debug variables

CMD_DEBUG_VARS_3 – request values of debug variables

CMD_WRITE_EXTERNAL_DATA – stores any user data to the dedicated area in the EEPROM
- data – 128b

CMD_READ_EXTERNAL_DATA – request user data, stored in the EEPROM
- data – 128b

CMD_API_VIRT_CH_CONTROL – update a state of 32 virtual channels that named "API_VIRT_CHXX" in the GUI
These channels can be selected as RC source to control camera or to do other tasks.
- VAL_CH1 – 2s
- …
- VAL_CH32 - 2s

CMD_READ_ADJ_VARS_CFG – request configuration of mapping of control inputs to adjustable variables
CMD_READ_ADJ_VARS_CFG incoming command is sent in response.

CMD_WRITE_ADJ_VARS_CFG – writes configuration of mapping of control inputs to adjustable variables
- Data format is the same as in corresponding CMD_READ_ADJ_VARS_CFG incoming command.

On success, confirmation is sent in response.

CMD_EEPROM_WRITE – writes a block of data to EEPROM to specified address
- ADDR – 4u, 64-byte aligned
- DATA – any size, 64-byte aligned

On success, confirmation CMD_CONFIRM is sent with parameters CMD_EEPROM_WRITE, ADDR.

CMD_EEPROM_READ – request a reading of block of data from EEPROM at the specified address and size.
- ADDR – 4u, 64-byte aligned
- SIZE – 2u, 64-byte aligned

On success, CMD_EEPROM_READ is sent. See its description.

CMD_READ_FILE – read file from internal filesystem

- FILE_ID – 2u
- PAGE_OFFSET – 2u
- MAX_SIZE – 2u
- RESERVED – 14b

This command reads a portion of data from the file with identifier FILE_ID, started at PAGE_OFFSET pages (1page = 64byte). MAX_SIZE bytes will be read or less, if file end is reached. Size should not exceed maximum allowed command data length. Read data or error code is sent in the incoming command CMD_READ_FILE.

CMD_WRITE_FILE – write file to internal filesystem

- FILE_ID – 2u
- FILE_SIZE - 2u
- PAGE_OFFSET – 2u
- DATA – 0 or any size

This command writes a portion of data to a file with identifier FILE_ID. If file is not exists, it is created. If FILE_SIZE is not equal to existing file size, file is adjusted to new size. If DATA is empty, file is deleted.
In response CMD_CONFIRM is sent, with parameter ERR_CODE. Possible codes:

```
NO_ERROR = 0
ERR_EEPROM_FAULT = 1
ERR_FILE_NOT_FOUND = 2
ERR_FAT = 3
ERR_NO_FREE_SPACE = 4
ERR_FAT_IS_FULL = 5
ERR_FILE_SIZE = 6
ERR_CRC = 7
ERR_LIMIT_REACHED = 8
```

CMD_FS_CLEAR_ALL – delete all files from internal filesystem
Returns CMD_CONFIRM with parameter ERR_CODE (see definitions in the CMD_WRITE_FILE command)

CMD_RUN_SCRIPT – start or stop user-written script

- MODE – 1u (0 – stop, 1 – start, 2 – start with debug information is sent back in the CMD_SCRIPT_DEBUG)
- SLOT – 1u
- RESERVED – 32b

CMD_CALIB_MAG – run magnetometer calibration
Simple format: not parameters
Extended format: not implemented

CMD_AHRS_HELPER – send or request attitude of the IMU sensor.
Use this command to replace internal IMU calculations by high-grade external IMU, providing new data with 50-100 Hz rate.

- MODE – 1u
- Z1_VECTOR[3] – 4f*3
- H1_VECTOR[3] – 4f*3

CMD_GYRO_CORRECTION – correct gyroscope sensor manually

- IMU_TYPE – 1u
- GYRO_ZERO_CORR[X] - 2s
- GYRO_ZERO_CORR[Y] - 2s
- GYRO_ZERO_CORR[Z] - 2s
- GYRO_ZERO_HEADING_CORR – 2s

CMD_DATA_STREAM_INTERVAL – register or update *data stream* – a sequence of commands sent by controller with the fixed rate without request. (ver. 2.59+)

- CMD_ID – 1u
- INTERVAL_MS – 2u
- CONFIG – 10b
- RESERVED – 10b

For each serial interface, only one unique combination of CMD_ID + CONFIG bytes may be registered. If data stream is already registered, it will be updated. To unregister it, specify INTERVAL_MS=0. Total number of data streams over all serial interfaces is limited (for 2.59 ver. limit is 10)
If data stream is successfully registered or updated, CMD_CONFIRM is sent in answer.
Take care about serial bandwidth: if data flow exceed bandwidth, particular samples may be skipped. The same is true when TX buffer is full, when sending long commands like CMD_READ_PARAMS_3.
Interval is maintained with +-1ms tolerance for individual sample, but averaged sample rate exactly matches to specified.
Meaning of CONFIG bytes is specific for each command and described in the 'Parameters' section.

CMD_REALTIME_DATA_CUSTOM – request configurable realtime data (ver. 2.59+)

- FLAGS – 4u
- RESERVED - 6b

Variables description and range

Name	Type	Min	Max	Possible values, remarks
CMD_BOARD_INFO - Version information				
BOARD_VER	1u			Multiplied by 10: 3.0 => 30
FIRMWARE_VER	2u			major_ver = (int)(FIRMWARE_VER/1000); minor_ver = (int)((FIRMWARE_VER%1000)/10); beta_ver = FIRMWARE_VER%10;
BOARD_FEATURES	2u			Bit set: BOARD_FEATURE_3AXIS = 1 BOARD_FEATURE_BAT_MONITORING = 2 BOARD_FEATURE_ENCODERS = 4 BOARD_FEATURE_BODE_TEST = 8 BOARD_FEATURE_SCRIPTING = 16 BOARD_FEATURE_CURRENT_SENSOR = 32
CONNECTION_FLAG	1u			Bit set: CONNECTION_USB = 1
CMD_READ_PARAMS_3, CMD_WRITE_PARAMS_3				
PROFILE_ID	1u			profile ID to read or write. To read or write current (active) profile, specify 255. Possible values: 0..4
P	1u	0	255	
I	1u	0	255	divided by 100 when displayed in the GUI
D	1u	0	255	
POWER	1u	0	255	
INVERT	1u	0	1	
POLES	1u	0	255	
ACC_LIMITER_ALL	1u	0	255	Units: 5 degrees/sec^2 0 – disabled. (from ver. 2.59 is deprecated; replaced by the ACC_LIMITER3)
EXT_FC_GAIN	1s	-127	127	
RC_MIN_ANGLE	2s	-180	180	
RC_MAX_ANGLE	2s	-180	180	
RC_MODE	1u			0..2 bits – mode: RC_MODE_ANGLE = 0 RC_MODE_SPEED = 1 3rd bit – control is inverted, if set to 1
RC_LPF	1u	0	16	
RC_SPEED	1u	0	255	
RC_FOLLOW	1u	-127	127	ROLL, PITCH: this value specify follow rate for flight controller. YAW: if value != 0, “follow motor” mode is

				enabled.
GYRO_TRUST	1u	0	255	
USE_MODEL	1u	0	1	
PWM_FREQ	1u			PWM_FREQ_LOW = 0 PWM_FREQ_HIGH = 1 PWM_FREQ_ULTRA_HIGH = 2 *(BOARD_VER>=30)*
SERIAL_SPPED	1u			115200 = 0 57600 = 1 38400 = 2 19200 = 3 9600 = 4
RC_TRIM_ROLL RC_TRIM_PITCH RC_TRIM_YAW	1s	-127	127	
RC_DEADBAND	1u	0	255	
RC_EXPO_RATE	1u	0	100	
RC_VIRT_MODE	1u			Mode of RC_ROLL input pin operation: RC_VIRT_MODE_NORMAL = 0 RC_VIRT_MODE_CPPM = 1 RC_VIRT_MODE_SBUS = 2 (BOARD_VER >= 30) RC_VIRT_MODE_SPEKTRUM = 3 (BOARD_VER >= 30) RC_VIRT_MODE_API = 10 (BOARD_VER >= 30)
RC_MAP_ROLL RC_MAP_PITCH RC_MAP_YAW RC_MAP_CMD RC_MAP_FC_ROLL RC_MAP_FC_PITCH	1u			Assigns pin input or virtual channel (in serial modes), and specifies input mode. INPUT_NO = 0 **PWM source** RC_INPUT_ROLL = 1 RC_INPUT_PITCH = 2 EXT_FC_INPUT_ROLL = 3 EXT_FC_INPUT_PITCH = 4 RC_INPUT_YAW = 5 (BOARD_VER >= 30) **Analog source** Input number + 32 (5th bit is set) BOARD_VER < 30: RC_INPUT_ROLL = 33 RC_INPUT_PITCH = 34 EXT_FC_INPUT_ROLL = 35 EXT_FC_INPUT_PITCH = 36 BOARD_VER >= 30: ADC1 = 33 ADC2 = 34 ADC3 = 35 **RC Serial source (CPPM/SBUS/SPEKTRUM):** Virtual channel (1..31) + 64 (6th bit is set)

				API Virtual control source Virtual channel (1..31) + 128 (7th bit is set)
RC_MIX_FC_ROLL RC_MIX_FC_PITCH	1u			Add FC channel to selected RC channels with given rate. bits 0..5: mix rate. For example, 0 - no mix (100% RC) 32 - 50% RC, 50% FC, 63 - 0% RC, 100% FC bits 6,7: target RC channel 0 - no mix 1 - ROLL 2 - PITCH 3 - YAW
FOLLOW_MODE	1u			FOLLOW_MODE_DISABLED=0 FOLLOW_MODE_FC=1 FOLLOW_MODE_PITCH=2
FOLLOW_DEADBAND	1u	0	255	
FOLLOW_EXPO_RATE	1u	0	100	
FOLLOW_OFFSET_ROLL FOLLOW_OFFSET_PITCH FOLLOW_OFFSET_YAW	1s	-127	127	
FOLLOW_ROLL_MIX_START	1u	0	90	
FOLLOW_ROLL_MIX_RANGE	1u	0	90	
AXIS_TOP AXIS_RIGHT FRAME_AXIS_TOP FRAME_AXIS_RIGHT	1s			Main IMU and frame IMU orientation: X = 1 Y = 2 Z = 3 -X = -1 -Y = -2 -Z = -3
FRAME_IMU_POS	1u			Location of the frame IMU: FRAME_IMU_DISABLED = 0 FRAME_IMU_BELOW_YAW = 1 FRAME_IMU_ABOVE_YAW = 2 FRAME_IMU_BELOW_YAW_PID_SOURCE = 3
GYRO_LPF	1u	0	5	0 means no LPF, 5 means LPF at maximum
I2C_INTERNAL_PULLUPS	1u	0	1	
SKIP_GYRO_CALIB	1u			Skip calibration of gyroscope. 0 – do not skip 1 – skip always 2 – try to calibrate but skip if motion is detected
RC_CMD_LOW RC_CMD_MID RC_CMD_HIGH	1u			Available actions: MENU_CMD_NO = 0 MENU_CMD_PROFILE1 = 1 MENU_CMD_PROFILE2 = 2

MENU_CMD_1..5 MENU_CMD_LONG				MENU_CMD_PROFILE3 = 3 MENU_CMD_SWAP_PITCH_ROLL = 4 MENU_CMD_SWAP_YAW_ROLL = 5 MENU_CMD_CALIB_ACC = 6 MENU_CMD_RESET = 7 MENU_CMD_SET_ANGLE = 8 MENU_CMD_CALIB_GYRO = 9 MENU_CMD_MOTOR_TOGGLE = 10 MENU_CMD_MOTOR_ON = 11 MENU_CMD_MOTOR_OFF = 12 MENU_CMD_FRAME_UPSIDE_DOWN = 13 MENU_CMD_PROFILE4 = 14 MENU_CMD_PROFILE5 = 15 MENU_CMD_AUTO_PID = 16 MENU_CMD_LOOK_DOWN = 17 MENU_CMD_HOME_POSITION = 18 MENU_CMD_RC_BIND = 19 MENU_CMD_CALIB_GYRO_TEMP = 20 MENU_CMD_CALIB_ACC_TEMP = 21 MENU_CMD_BUTTON_PRESS = 22 MENU_CMD_RUN_SCRIPT1 23 MENU_CMD_RUN_SCRIPT2 24 MENU_CMD_RUN_SCRIPT3 25 MENU_CMD_RUN_SCRIPT4 26 MENU_CMD_RUN_SCRIPT5 27 MENU_CMD_CALIB_MAG 33 MENU_CMD_LEVEL_ROLL_PITCH 34 MENU_CMD_CENTER_YAW 35 MENU_CMD_UNTWIST_CABLES 36 MENU_CMD_SET_ANGLE_NO_SAVE 37
OUTPUT_ROLL OUTPUT_PITCH OUTPUT_YAW	1u			DISABLED = 0 ROLL = 1 PITCH = 2 YAW = 3
BAT_THRESHOLD_ALARM	2s	-3000	3000	Negative means means alarm is disabled *Units: 0.01V*
BAT_THRESHOLD_MOTORS	2s	-3000	3000	Negative value means function is disabled *Units: 0.01V*
BAT_COMP_REF	2s	-3000	3000	Negative value means compensation is disabled. *Units: 0.01V*
BEEPER_MODES	1u			BEEPER_MODE_CALIBRATE=1 BEEPER_MODE_CONFIRM=2 BEEPER_MODE_ERROR=4 BEEPER_MODE_ALARM=8 BEEP_BY_MOTORS=128 *(if this flag is set, motors emit sound instead of internal buzzer)*
BOOSTER_POWER_ROLL BOOSTER_POWER_PITCH BOOSTER_POWER_YAW	1u	0	255	Additional power to correct broken synchronization
FOLLOW_SPEED_ROLL FOLLOW_SPEED_PITCH FOLLOW_SPEED_YAW	1u	0	255	

CUR_IMU	1u			IMU_TYPE_MAIN=1 IMU_TYPE_FRAME=2
FRAME_ANGLE_FROM_MOTORS	1u	0	1	
RC_MEMORY_ROLL RC_MEMORY_PITCH RC_MEMORY_YAW	2s	-36767	32767	Initial angle that is set at system start-up, in 14bit resolution *Units: 0,02197265625 degree*
SERVO1_OUT SERVO2_OUT SERVO3_OUT SERVO4_OUT	1u			Disabled = 0 1..32 - Virtual channel number as source of data to be output
SERVO_RATE	1u	5	40	PWM frequency, 10 Hz per unit.
ADAPTIVE_PID_ENABLED	1u			Set of bits (0 – disable all): EN_ROLL = 1 EN_PITCH = 2 EN_YAW = 4
ADAPTIVE_PID_THRESHOLD	1u	0	255	
ADAPTIVE_PID_RATE	1u	1	255	
ADAPTIVE_PID_RECOVERY_FACTOR	1u	0	10	
FOLLOW_LPF_ROLL FOLLOW_LPF_PITCH FOLLOW_LPF_YAW	1u	0	16	
CUR_PROFILE	1u	0		Active profile, 0..4
GENERAL_FLAGS1	2u			REMEMBER_LAST_USED_PROFILE = (1<<0) UPSIDE_DOWN_AUTO = (1<<1) SWAP_FRAME_MAIN_IMU = (1<<2) BLINK_PROFILE = (1<<3) EMERGENCY_STOP = (1<<4) MAGNETOMETER_POS_FRAME = (1<<5)
PROFILE_FLAGS1	2u			ADC1_AUTO_DETECTION = (1<<0) ADC2_AUTO_DETECTION = (1<<1) ADC3_AUTO_DETECTION = (1<<2) FOLLOW_USE_FRAME_IMU = (1<<4) BRIEFCASE_AUTO_DETECTION = (1<<5) UPSIDE_DOWN_AUTO_ROTATE = (1<<6) FOLLOW_LOCK_OFFSET_CORRECTION = (1<<7) START_NEUTRAL_POSITION = (1<<8)
SPEKTRUM_MODE	1u			0 Auto-detection (default) 1 DSM2/11ms/10bit 2 DSM2/11ms/11bit 3 DSM2/22ms/10bit 4 DSM2/22ms/11bit 5 DSMX/11ms/10bit 6 DSMX/11ms/11bit 7 DSMX/22ms/10bit

				8 DSMX/22ms/11bit
ORDER_OF_AXES	1u			Order of hardware axes, counting from a camera: PITCH_ROLL_YAW = 0 YAW_ROLL_PITCH = 1 ROLL_YAW_PITCH* = 2 ROLL_PITCH_YAW = 3 * not implemented
EULER_ORDER	1u			Order of Euler angles to represent the current orientation of a camera and the target of stabilization: PITCH_ROLL_YAW = 0 ROLL_PITCH_YAW = 1 LOCAL_ROLL* = 2 ROLL_LOCAL* = 3 YAW_ROLL_PITCH = 4 YAW_PITCH_ROLL = 5 * dedicated for 2-axis systems only
CMD_READ_PARAMS_EXT, CMD_WRITE_PARAMS_EXT - Extended parameters				
NOTCH_FREQ	1u	0	255	Center frequency, x2 Hz (value 10 means 20Hz)
NOTCH_WIDTH	1u	0	255	Width of -3dB gain band, Hz
LPF_FREQ	2u	0	1000	Low-pass filter -3dB cut-off frequency, Hz
FILTERS_EN	1u			Set of bits (0 – disable all): EN_NOTCH1 = 1 EN_NOTCH2 = 2 EN_NOTCH3 = 4 EN_LPF = 8
NOTCH_GAIN	1s	-100	100	Notch gain, in dB (positive – notch, negative – peak filter)
ENCODER_OFFSET	2s			*Units: 0,02197265625 degree*
ENCODER_FLD_OFFSET	2s			*Units: 0,02197265625 degree*
ENCODER_MANUAL_SET_TIME	1u	0	255	*Units: 10ms*
MOTOR_HEATING_FACTOR	1u	0	255	
MOTOR_COOLING_FACTOR	1u	0	255	
FOLLOW_INSIDE_DEADBAND	1u	0	255	
MOTOR_MAG_LINK	1u	0	255	Deprecated, replaced by MOTOR_MAG_LINK_FINE
MOTOR_GEARING	2u			Real number encoded as 8.8 fixed point (1.0f → 256)

ENCODER_LIMIT_MIN	1s	-127	127	*Units: 3 degree*
ENCODER_LIMIT_MAX	1s	-127	127	*Units: 3 degree*
NOTCH1_GAIN NOTCH2_GAIN NOTCH3_GAIN	1u	0	100	
BEEPER_VOLUME	1u	0	255	
ENCODER_GEAR_RATIO	2u			*Units: 0.001*
ENCODER_TYPE	1u			Bits 0..3: ENC_TYPE_AS5048A = 1 ENC_TYPE_AS5048B = 2 ENC_TYPE_AS5048_PWM = 3 ENC_TYPE_AMT203 = 4 ENC_TYPE_MA3_10BIT = 5 ENC_TYPE_MA3_12BIT = 6 ENC_TYPE_ANALOG = 7 ENC_TYPE_I2C_DRV1 = 8 ENC_TYPE_I2C_DRV2 = 9 ENC_TYPE_I2C_DRV3 = 10 ENC_TYPE_I2C_DRV4 = 11 ENC_TYPE_AS5600_PWM = 12 ENC_TYPE_AS5600_I2C = 13 Bit 4: SKIP_DETECTION = 1
ENCODER_CFG	1u			For SPI encoders: SPI_SPEED_1MHz = 0 SPI_SPEED_2MHz = 1 SPI_SPEED_4MHz = 2 SPI_SPEED_500kHz = 3 For I2C_DRV: internal encoder type
OUTER_P	1u	0	255	
OUTER_I	1u	0	255	
MAG_AXIS_TOP MAG_AXIS_RIGHT	1s			X = 1 Y = 2 Z = 3 -X = -1 -Y = -2 -Z = -3
MAG_TRUST	1u	0	255	
MAG_DECLINATION	1s	-127	127	Units: 1 degree
ACC_LPF_FREQ	2u	0	1000	Units: 0.01Hz
D_TERM_LPF_FREQ[3]	1u	0	60	Units: 10Hz
CMD_READ_PARAMS_EXT2, CMD_WRITE_PARAMS_EXT2 - Extended parameters set2				
MOTOR_MAG_LINK_FIN	2u	0	65535	Units: 0.01

E				
ACC_LIMITER[3]	1u	0	200	Units: 5 degrees/sec^2
CMD_REALTIME_DATA_3 - Real-time data				
ACC GYRO	2s			raw data from sensors
DEBUG	2s			debug variables
RC_ROLL RC_PITCH RC_YAW	2s	1000	2000	RC control channels values (PWM or normalized analog)
RC_CMD	2s	1000	2000	RC command channel value (PWM or normalized analog)
EXT_FC_ROLL EXT_FC_PITCH	2s	1000	2000	External FC PWM values. May be zero if their inputs are mapped to RC control or command.
ANGLE_ROLL ANGLE_PITCH ANGLE_YAW	2s	-32768	32767	Camera angles in 14-bit resolution per full turn *Units: 0,02197265625 degree*
RC_ANGLE_ROLL RC_ANGLE_ROLL RC_ANGLE_ROLL	2s	-32768	32767	RC angles, in 14-bit resolution *Units: 0,02197265625 degree*
FRAME_ANGLE_ROLL FRAME_ANGLE_PITCH FRAME_ANGLE_YAW	2s	-32768	32767	Frame angles detected by the second IMU (if present), in 14-bit resolution. *Units: 0,02197265625 degree*
CYCLE_TIME	2u			
I2C_ERROR_COUNT	2u			Number of registered errors on I2C bus
SYSTEM_ERROR	2u			Set of bits (0 – no error): ERR_NO_SENSOR (1<<0) ERR_CALIB_ACC (1<<1) ERR_SET_POWER (1<<2) ERR_CALIB_POLES (1<<3) ERR_PROTECTION (1<<4) ERR_SERIAL (1<<5) *Beside that, extended error contains bits:* ERR_LOW_BAT1 (1<<6) ERR_LOW_BAT2 (1<<7) ERR_GUI_VERSION (1<<8) ERR_MISS_STEPS (1<<9) ERR_SYSTEM (1<<10) ERR_EMERGENCY_STOP (1<<11)
SYSTEM_SUB_ERROR	1u			Specifies the reason of emergency stop
BAT_LEVEL	2u			Battery voltage *Units: 0.01 volt*

OTHER_FLAGS	1u			bit0 set - motors turned ON bit1..7 - reserved
CUR_PROFILE	1u	0		Active profile, 0..4
CUR_IMU	1u			Currently selected IMU `IMU_TYPE_MAIN=1` `IMU_TYPE_FRAME=2` *(BOARD_VER>=30 only)*
CMD_REALTIME_DATA_4				
STATOR_ROTOR_ANGLE	2s			Relative angle for joints between two arms of gimbal structure, measured by encoder (with offset and gearing calibration is applied), by 2nd IMU or by other algorithms. Value 0 corresponds to normal position (each arms forms 90 degrees with the next order arm). *Units: 0,02197265625 degree*
BALANCE_ERROR_ROLL BALANCE_ERROR_PITCH BALANCE_ERROR_YAW	2s	-512	512	Error in balance (0 – perfect balance, 512 - 100% motor power is required to hold camera)
CURRENT	2u			Actual current consumption. *Units: mA*
MAG_DATA_ROLL MAG_DATA_PITCH MAG_DATA_YAW	2s			Raw data from magnetometer
IMU_TEMPERATURE FRAME_IMU_TEMPERATURE	1s	-127	127	Temperature of IMU boards. *Units: Celsius*
~~FRAME_CAM_ANGLE_ROLL~~ ~~FRAME_CAM_ANGLE_PITCH~~ ~~FRAME_CAM_ANGLE_YAW~~	2s			Deprecated starting from version 2.55, see STATOR_ROTOR_ANGLE instead
IMU_G_ERR	1u	0	255	Error between estimated gravity vector and reference vector for currently active IMU *Units: 0.1 degree*
IMU_H_ERR	1u	0	255	Error between estimated heading vector and reference vector for currently active IMU *Units: 0.1 degree*
CMD_CONTROL - Control				
CONTROL_MODE* extended format: CONTROL_MODE_ROLL CONTROL_MODE_PITC	1u			Bits 0..3 for mode, bits 4..7 for flags. `MODE_NO_CONTROL=0` `MODE_SPEED=1` `MODE_ANGLE=2` `MODE_SPEED_ANGLE=3`

H CONTROL_MODE_YAW				MODE_RC=4 MODE_ANGLE_REL_FRAME=5 • MODE_SPEED – camera travels with the given speed in the Euler coordinates until the next CMD_CONTROL command comes. Given angle is ignored. • MODE_ANGLE – camera travels to the given point in the Euler coordinates with the given speed. If speed=0, default speed is used (set in the GUI). • MODE_SPEED_ANGLE – camera travels with the given speed while the actual angle matches the given angle. Additionally, PID controller keeps the given angle. This mode allows the most precise and error-proof control. See fig.1 for example. • MODE_RC - angle parameter overrides RC signal input data. Should be in range -500...500. Speed parameter is ignored. • MODE_ANGLE_REL_FRAME – first, neutral point of a camera relative to a frame is found in the Euler coordinates. Than, given angle value is add to this point, and camera travels to it with the given speed. If speed=0, default speed is used (set in the GUI). For example, if the ANGLE parameter = 0 and camera made 2 full turns by YAW, it will make 2 turns back and returns to neutral point. This mode may be helpful in untwisting cables, for example. CONTROL_FLAG_HIGH_RES_SPEED=(1<<7) • CONTROL_FLAG_HIGH_RES_SPEED – speed units changed to 0.001 deg/sec for extremely slow motion (timelapse shooting) *(frw.ver 2.59+)*
SPEED_ROLL SPEED_PITCH SPEED_YAW	2s	- - -	- - -	Speed of rotation. If acceleration limiter is enabled in the settings, given speed may be limited. *Units: 0,1220740379 degree/sec* *or 0.001 degree/sec, if* *CONTROL_FLAG_HIGH_RES_SPEED is set*
ANGLE_ROLL ANGLE_PITCH ANGLE_YAW	2s	-32768	32767	Target angle. Ignored in the MODE_SPEED mode. If mode=MODE_RC, it specifies RC data in range -500..500 *Units: 0,02197265625 degree.*
Notes: • Serial control overrides RC control. To switch back to RC, send this command with the mode=MODE_NO_CONTROL and all data set to zeros. • Send this command with rate 50Hz or less • See Appendix A for source code example				
CMD_TRIGGER_PIN - Trigger pin				
PIN_ID	1u			Triggers pin only if it is not used for input RC_INPUT_ROLL = 1 RC_INPUT_PITCH = 2 EXT_FC_INPUT_ROLL = 3 EXT_FC_INPUT_PITCH = 4

				RC_INPUT_YAW = 5 (BOARD_VER >= 30) PIN_AUX1* = 16 PIN_AUX2* = 17 PIN_AUX3* = 18 PIN_BUZZER* = 32 PIN_SSAT_POWER** = 33 * On boards v1.x (based on Atmega328p) PIN_AUX1..3 are not present as outputs, and should be soldered to pin2, pin11, pin12 of MCU correspondingly. PIN_BUZZER is mapped to pin32 of MCU. ** PIN_SSAT_POWER triggers 3.3V power line in the Spektrum connector (low state enables power)
STATE	1u			LOW = 0 HIGH = 1 LOW - pin can sink up to 40mA HIGH - pin can source up to 40mA
CMD_GET_ANGLES – information about angles in system				
IMU_ANGLE	2s	-32768	32767	Actual angle measured by IMU. After 2 full turns, angle is cycled *Units: 0,02197265625 degree.*
RC_TARGET_ANGLE	2s	-32768	32767	Target angle that gimbal should keep. Angle is set by RC or control command 'C'. *Units: 0,02197265625 degree.*
RC_SPEED	2s	- - -	- - -	Target speed that gimbal should keep. Speed is set by RC or control command 'C'. Zero speed means control is idle (target is reached) *Units: 0,1220740379 degree/sec*
CMD_GET_ANGLES_EXT – information about angles in system, different format				
IMU_ANGLE	2s	-32768	32767	Actual angle measured by IMU. After 2 full turns, angle is cycled *Units: 0,02197265625 degree.*
RC_TARGET_ANGLE	2s	-32768	32767	Target angle that gimbal should keep. *Units: 0,02197265625 degree.*
STATOR_ROTOR_ANGLE	4s			Relative angle for joints between two arms of gimbal structure, measured by encoder or 2nd IMU. Value 0 corresponds to normal position of a gimbal. This angle does not overflow after multiple turns. *Units: 0,02197265625 degree*
CMD_EXECUTE_MENU - Execute menu command				
CMD_ID	1u			Executes a menu command (acts like the menu button or RC control channel) See the RC_CMD_LOW parameter inside the CMD_READ_PARAMS_3 command for available menu commands.
CMD_SELECT_IMU_3 - Select IMU to configure				

IMU_TYPE	1u			IMU_TYPE_MAIN=1 IMU_TYPE_FRAME=2 If selected IMU is not connected, command is ignored.
CMD_SET_ADJ_VARS_VAL – Set the values of multiple adjustable parameters				
NUM_PARAMS	1u	1	40	Number of parameters in command
PARAM<N>_ID	1u			ID of parameter. Full list is in Appendix B.
PARAM<N>_VALUE ...	4b			Value depends on type of parameter. Types and min, max range should be requested from board by CMD_GET_PARAMS_3 command. Values are packed according to C-language memory model, little-endian order. 1- or 2-byte types converted to 4-byte using C-language type conversions. Floats packed according to IEEE-754.
CMD_GET_ADJ_VARS_VAL – Query the values of multiple adjustable parameters				
NUM_PARAMS	1u	1	40	Number of parameters in command
PARAM<N>_ID	1u			ID of parameter. Full list is in Appendix B.
CMD_AUTO_PID - Start automatic PID calibration				
PROFILE_ID	1u			
CFG_FLAGS	1u			Set of bits: AUTO_PID_STOP = 0 AUTO_PID_CFG_ROLL = 1 AUTO_PID_CFG_PITCH = 2 AUTO_PID_CFG_YAW = 4 AUTO_PID_CFG_SEND_GUI = 8 AUTO_PID_CFG_KEEP_CURRENT = 16 AUTO_PID_CFG_TUNE_LPF_FREQ = 32
GAIN_VS_STABILITY	1u	0	255	
CMD_SERVO_OUT - Output PWM signal on the specified pin				
SERVO1_TIME SERVO2_TIME SERVO3_TIME SERVO4_TIME SERVO5_TIME SERVO6_TIME SERVO7_TIME SERVO8_TIME	2s	-1	20000	value < 0: free up this pin and make it floating value = 0: configure this pin as output and set it to 'Low' state value > 0: PWM pulse time, us. Should be less than PWM period, configured by the “SERVO_RATE” parameter. Regular servo accept values in range about 500..2500 us, 1500 us is neutral position, PWM period is 20000 us or less.
CMD_DEBUG_VARS_INFO_3 – definition of debug variables passed in CMD_DEBUG_VARS_3				
DEBUG_VARS_NUM	1u	1	255	
VAR_NAME	string			1st byte is size, following by ASCII characters
VAR_TYPE	1u			*Type (0..3 bits):* VAR_TYPE_UINT8 = 1 VAR_TYPE_INT8 = 2

				VAR_TYPE_UINT16 = 3 VAR_TYPE_INT16 = 4 VAR_TYPE_UINT32 = 5 VAR_TYPE_INT32 = 6 VAR_TYPE_FLOAT = 7 (IEEE-754) *Flags (4..7 bits):* VAR_FLAG_ROLL = 16 its belong to ROLL axis VAR_FLAG_PITCH = 32 its belong to PITCH axis VAR_FLAG_YAW = 48 its belong to YAW axis VAR_FLAG_ANGLE14 = 64 its an angle (14bit per turn)
ARR_SIZE	2u			
CMD_API_VIRT_CH_CONTROL – update a state of all virtual channels that named "API_VIRT_CHXX" in the GUI				
VAL_CH1 .. VAL_CH32	2s	-500	500	Value may go outside these limits and will be clipped. Use a special value "-10000" to mark that channel has "undefined" state (its treated as "signal lost" like with RC inputs)
CMD_AHRS_HELPER – get or set attitude of main or frame IMU (use to set or correct attitude from external high-grade IMU and to receive attitude in rotation matrix form instead of Euler angles)				
MODE	1u			bit0: 0 – get, 1 – set bit1: 0 – main IMU, 1 – frame IMU bit2: if set, use as reference only bit3: if set, translate from camera to frame (or back) and use as a reference bit4: if set, use Z1 only bit5: if set, use H1 only Below some useful combinations of flags are described in details. *GET modes (provided data and other flags are ignored):* 0 - request the main IMU attitude 2 - request the frame IMU attitude *SET modes:* 1 - use as a camera attitude (replace the attitude estimated by the main IMU) 3 - use as a frame attitude (regardless of 2nd IMU is enabled or not) 5 - use as a reference for the main IMU (to correct gyro drift using GYRO_TRUST factor) 7 - use as a reference for the frame IMU 11 - use as a frame attitude, translate to the camera coordinates and use as a reference for the main IMU. 15 – use as a reference for the frame IMU, translate to the camera coordinates and use as a reference for the main IMU. *Modes 1,5 should be used if an external AHRS source is installed on the camera's platform. Modes 3,7,11,15 should be used if an external AHRS source is installed on the frame (above all motors).*

				Bit3 is taken into account only if all motor angles are known from encoders or may be estimated using other ways. *Bits 4..5 can be combined with the previous values to selectively correct/replace only H1 or Z1 attitude vectors. For example, you can leave Z1 corrected by the internal accelerometer, and correct only H1 (heading) by an external magnetometer.*
Z1_VECT[3]	4f*3	-1.0f	1.0f	Unit vector that points Up (Z-axis in normal position)
H1_VECT[3]	4f*3	-1.0f	1.0f	Unit vector that points towards North (Y-axis in normal position)
CMD_GYRO_CORRECTION – correct gyro sensor manually				
IMU_TYPE	1u			0 – main IMU, 1 – frame IMU
GYRO_ZERO_CORR[X] GYRO_ZERO_CORR[X] GYRO_ZERO_CORR[X]	2s			Zero offset for each axis *Units: 0.001 gyro sensor unit*
GYRO_ZERO_HEADING_CORR	2s			Zero offset for global Z axis to correct a heading only. This correction is distributed to all axes automatically. *Units: 0.001 gyro sensor unit*
CMD_DATA_STREAM_INTERVAL - register or update *data stream (ver. 2.59+)*				
CMD_ID	1u			Command ID to be sent by this data stream. All possible commands are listed below.
INTERVAL_MS	2u			Interval between messages, in milliseconds. Value 1 means each cycle (0.8ms) If set to 0 – unregister data stream
CONFIG	10b			Configuration specific to each command: **CMD_REALTIME_DATA_3** – no parameters **CMD_REALTIME_DATA_4** – no parameters **CMD_REALTIME_DATA_CUSTOM** • flags – 4u, see command specification **CMD_AHRS_HELPER** • imu_type – 1u (0 – main IMU, 1 – frame IMU)
CMD_REALTIME_DATA_CUSTOM – request for configurable realtime data (ver. 2.59+)				
FLAGS	4u			Bit set, each bit specify which data to include in response • bit0: IMU angles • bit1: RC target angles • bit2: RC target speed • bit3: Stator-rotor angle • bit4: IMU sensor gyro data • bit5: RC signal assigned to standard inputs • bit6: IMU attitude as rotation matrix • bit7: All RC channels captured from s-bus, Sum-PPM or spektrum input. • bit8: IMU sensor ACC data

				See specification of response for more details
CMD_REALTIME_DATA_CUSTOM – response for configurable realtime data (ver. 2.59+)				
TIMESTAMP_MS	2u			Timestamp in milliseconds
IMU_ANGLES[3]	2s*3			Main IMU angles (Euler) *Units: 0,02197265625 degree.*
TARGET_ANGLES[3]	2s*3			Target angles that gimbal should keep (Euler) *Units: 0,02197265625 degree.*
TARGET_SPEED[3]	2s*3			Target speed that gimbal should keep, over Euler axes *Units: 0,06103701895 degree/sec*
STATOR_ROTOR_ANGLE[3]	2s*3			Relative angle of joints (motors) *Units: 0,02197265625 degree.*
GYRO_DATA[3]	2s*3			Gyro sensor data after calibrations are applied
RC_DATA[6]	2s*6			RC data in high resolution, assigned to the ROLL, PITCH, YAW, CMD, FC_ROLL, FC_PITCH inputs. *Units: normal range is -16384..16384, -32768 is for 'undefined' signal*
Z1_VECTOR[3] H1_VECTOR[3]	4f*6	-1.0f	1.0f	IMU attitude in a form of rotation matrix (2 rows as gravity and heading vectors, 3rd row can be calculated as cross-product of them).
RC_CHANNELS[18]	2s*18			All RC channels captured from s-bus, spektrum or Sum-PPM inputs. *Mapped to -16384..16384, -32768 is for 'undefined' signal*
ACC_DATA[3]	2s*3			Accelerometer sensor data with calibrations
CMD_HELPER_DATA - Pass helper data from an outer system Used to increase precision of the stabilization				
FRAME_ACC[3]	2s	-	-	Linear acceleration of the frame, [X,Y,Z] components in a given coordinate system (see below). Helps to keep horizon during accelerated motion. *Units: 1g/512 ≈ 0,019160156 m/s²*
FRAME_ANGLE_ROLL FRAME_ANGLE_PITCH	2s	-32768	32767	Inclination of the outer frame in a given coordinate system. Pass zero values to not use this information. *Units: 0,02197265625 degree.*
COORD_SYS	1u			**`COORD_SYS_GROUND_YAW_ROTATED=1`** `(default)` Ground system rotated with the camera over Z axis. Z points Up, X points right, Y points forward.
FRAME_SPEED[3]	2s	-	-	Angular speed of the frame, [X,Y,Z] components in a given coordinate system. Helps to increase a precision of stabilization in systems w/out encoders or 2nd IMU. Pass zero values to not use it. *Units: 0,06103701895 degree/sec*

* The difference between control modes is illustrated on the picture below:

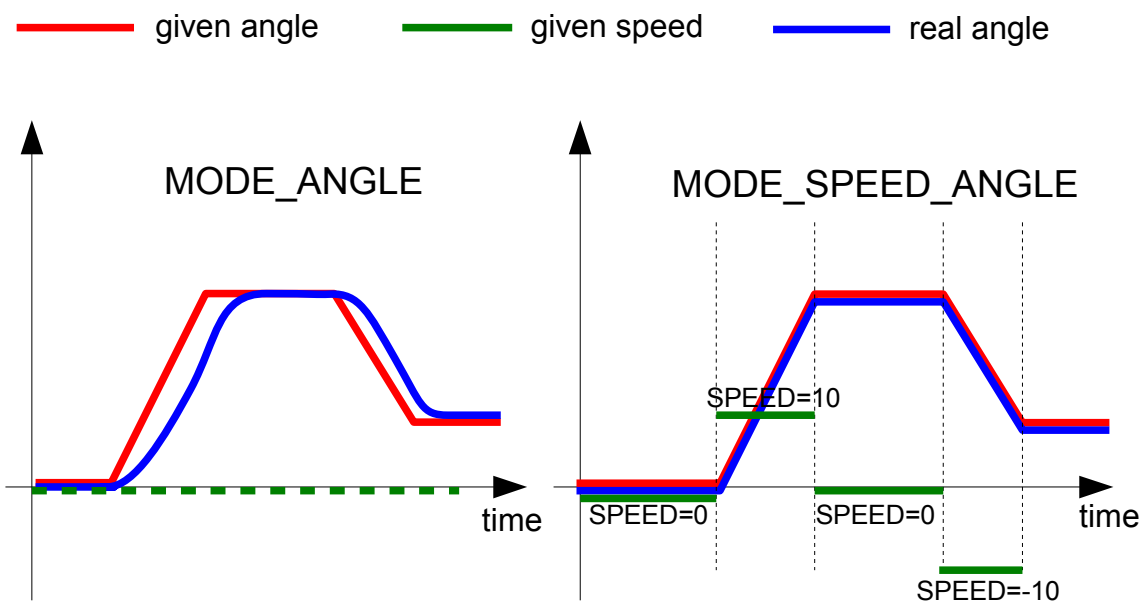

Fig.1 – Control modes

Appendix A: Examples and libraries

Examples can be downloaded from the link:

https://github.com/alexmos/sbgc-api-examples

See README for details.

Currently, examples provided for Arduino platform only.

Libraries

C++ library included as a part of examples folder.

Appendix B: Definition of dynamically configurable parameters

Used in CMD_SET_ADJ_VARS, CMD_GET_PARAMS_3, CMD_READ_ADJ_VARS_CFG, CMD_WRITE_ADJ_VARS_CFG

WARNING: this is not final and complete specification. Use CMD_GET_PARAMS_3 to receive actual list of parameters supported by current firmware.

NAME	ID	TYPE	MIN	MAX	REMARK
P_ROLL	0	1u	0	255	
P_PITCH	1	1u	0	255	
P_YAW	2	1u	0	255	
I_ROLL	3	1u	0	255	
I_PITCH	4	1u	0	255	
I_YAW	5	1u	0	255	
D_ROLL	6	1u	0	255	
D_PITCH	7	1u	0	255	
D_YAW	8	1u	0	255	
POWER_ROLL	9	1u	0	255	
POWER_PITCH	10	1u	0	255	
POWER_YAW	11	1u	0	255	
ACC_LIMITER	12	2s	0	1275	Units: degrees/sec^2
FOLLOW_SPEED_ROLL	13	1u	0	255	
FOLLOW_SPEED_PITCH	14	1u	0	255	
FOLLOW_SPEED_YAW	15	1u	0	255	
FOLLOW_LPF_ROLL	16	1u	0	16	
FOLLOW_LPF_PITCH	17	1u	0	16	
FOLLOW_LPF_YAW	18	1u	0	16	
RC_SPEED_ROLL	19	1u	0	255	
RC_SPEED_PITCH	20	1u	0	255	
RC_SPEED_YAW	21	1u	0	255	
RC_LPF_ROLL	22	1u	0	16	
RC_LPF_PITCH	23	1u	0	16	
RC_LPF_YAW	24	1u	0	16	
RC_TRIM_ROLL	25	1s	-127	127	
RC_TRIM_PITCH	26	1s	-127	127	

RC_TRIM_YAW	27	1s	-127	127	
RC_DEADBAND	28	1u	0	255	
RC_EXPO_RATE	29	1u	0	100	
FOLLOW_MODE	30	1u	0	2	0 – disabled 1 – Follow flight controller 2 – “Follow PITCH,ROLL” mode
RC_FOLLOW_YAW	31	1u	0	1	0 – disabled 1 - “Follow YAW” mode
FOLLOW_DEADBAND	32	1u	0	255	
FOLLOW_EXPO_RATE	33	1u	0	100	
FOLLOW_ROLL_MIX_START	34	1u	0	90	
FOLLOW_ROLL_MIX_RANGE	35	1u	0	90	
GYRO_TRUST	36	1u	0	255	
FRAME_HEADING_ANLGE	37	2s	-1800	1800	Units: 0.1 degrees
GYRO_HEADING_CORRECTION	38	2s	-20000	20000	Units: 0.001 of gyro sensor units
ACC_LIMITER_ROLL	39	2s	0	1275	Units: degrees/sec^2
ACC_LIMITER_PITCH	40	2s	0	1275	Units: degrees/sec^2
ACC_LIMITER_YAW	41	2s	0	1275	Units: degrees/sec^2