SimpleBGC32 2.6x serial protocol specification

Applicable for 32-bit boards with firmware 2.6x

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;
- rev. 0.13 – 05.06.2016: new command CMD_BEEP_SOUND; new adjustment variables;
- rev. 0.14 – 21.06.2016: CMD_ADJ_VARS_STATE described;
- rev. 0.15 – 09.07.2016: CMD_READ_PARAMS_EXT2 was extended; CMD_AUTO_PID - CFG_FLAGS was extended; CMD_CALIB_INFO was documented; CMD_DATA_STREAM_INTERVAL was corrected;
- rev. 0.16 – 10.08.2016: MavLink parameters are described in the CMD_READ_PARAMS_EXT2; several minor corrections;
- rev. 0.17 – 21.10.2016: new commands CMD_CONTROL_CONFIG, CMD_CALIB_ORIENT_CORR; CMD_READ_PARAMS_EXT2 extended;
- rev. 0.18 – 23.03.2017: CMD_HELPER_DATA extended by the FRAME_HEADING parameter; CMD_CONTROL is extended by the `CONTROL_FLAG_AUTO_TASK;` new command CMD_CALIB_ACC_EXT_REF; document structure is updated;

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

Many parameters are grouped in arrays, that is indicated by the square brackets notation: "ANGLE[3]". Parameters that are split to axes, always go in the order ROLL, PITCH, YAW for the Euler angles, or X, Y, Z for the axes of reference.

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
#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_CALIB_INFO 49
#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_BEEP_SOUND 89
#define CMD_ENCODERS_CALIB_OFFSET_4  26
#define CMD_ENCODERS_CALIB_FLD_OFFSET_4 27
#define CMD_CONTROL_CONFIG 90
#define CMD_CALIB_ORIENT_CORR 91
#define CMD_COGGING_CALIB_INFO 92
#define CMD_CALIB_COGGING 93
#define CMD_CALIB_ACC_EXT_REF 94

#define CMD_MAVLINK_INFO 250
#define CMD_MAVLINK_DEBUG 251
#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 information

Name	Type	Min	Max	Possible values, remarks
BOARD_VER	1u			Multiplied by 10: 3.0 => 30

FIRMWARE_VER	2u			Split into decimal digits X.XX.X, for example 2305 means 2.30b5
				`major_ver = (int)(FIRMWARE_VER/1000);`
				`minor_ver = (int)((FIRMWARE_VER%1000)/10);`
				`beta_ver = FIRMWARE_VER%10;`

RESERVED	1b			

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`
FRW_EXTRA_ID	4u			Used for specific builds only
RESERVED	7b			

CMD_BOARD_INFO_3 – additional board information

Name	Type	Min	Max	Possible values, remarks
DEVICE_ID	9b			Unique Id used to identify each controller in licensing system
MCU_ID	12b			MCU ID, unique
EEPROM_SIZE	4u			Size of available EEPROM in current device. Generally 32K bytes
SCRIPT_SLOT1_SIZE SCRIPT_SLOT2_SIZE SCRIPT_SLOT3_SIZE SCRIPT_SLOT4_SIZE SCRIPT_SLOT5_SIZE	2u			size of user-written scripts stored in each slot, 0 if slot is empty.
RESERVED	34b			

CMD_READ_PARAMS_3 – read/write system configuration part 1

Receive parameters for a single profile.

Name	Type	Min	Max	Possible values, remarks

PROFILE_ID		1u			profile ID to read or write. To access current (active) profile, specify 255. Possible values: 0..4
axis = (1..3)	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.* (starting from ver. 2.60 is deprecated; replaced by the ACC_LIMITER3)
EXT_FC_GAIN		1s	-127	127	
axis = (1..3)	RC_MIN_ANGLE	2s	-720	720	*Units: degrees*
	RC_MAX_ANGLE	2s	-720	720	*Units: degrees*

	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	15	
	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`

SERIAL_SPPED		1u			Baud rate for the main UART1 port (where USB normally connects)
					`115200 = 0`
					`57600 = 1`
					`38400 = 2`
					`19200 = 3`
					`9600 = 4`
					`256000 = 5`

RC_TRIM[3]		1s*3	-127	127	
RC_DEADBAND		1u	0	255	
RC_EXPO_RATE		1u	0	100	

RC_VIRT_MODE		1u			The mode of the RC_ROLL input pin operation:
					`RC_VIRT_MODE_NORMAL = 0`
					`RC_VIRT_MODE_CPPM = 1`
					`RC_VIRT_MODE_SBUS = 2`

				RC_VIRT_MODE_SPEKTRUM = 3 RC_VIRT_MODE_API = 10
RC_MAP_ROLL RC_MAP_PITCH RC_MAP_YAW RC_MAP_CMD RC_MAP_FC_ROLL RC_MAP_FC_PITCH	1u*6			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 **Analog source** 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 1u			Mix the value received from the FC channel, to the value received from the selected RC channels, with the 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[3]	1s*3	-127	127	
AXIS_TOP AXIS_RIGHT FRAME_AXIS_TOP FRAME_AXIS_RIGHT	1s 1s 1s 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_DEADBAND	1u	0	255	*Units: 0.1 of gyro sensor's units.*

GYRO_SENS	1u			deprecated
I2C_SPEED_FAST	1u	0	1	If set, use 800kHz ultra-fast speed mode, otherwise use 400kHz speed

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 MENU_CMD_1 MENU_CMD_2 MENU_CMD_3 MENU_CMD_4 MENU_CMD_5 MENU_CMD_LONG	1u*9			Assign action to various event sources. See CMD_EXECUTE_MENU for available actions

MOTOR_OUTPUT[3]	1u*3			Motor output mapping
				`DISABLED = 0`
				`ROLL = 1`
				`PITCH = 2`
				`YAW = 3`
				`I2C_DRV#1 = 4`
				`I2C_DRV#2 = 5`
				`I2C_DRV#3 = 6`
				`I2C_DRV#4 = 7`

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)

FOLLOW_ROLL_MIX_START	1u	0	90	
FOLLOW_ROLL_MIX_RANGE	1u	0	90	
BOOSTER_POWER[3]	1u*3	0	255	Additional power to correct lost synchronization
FOLLOW_SPEED[3]	1u*3	0	255	
FRAME_ANGLE_FROM_MOTORS	1u	0	1	
RC_MEMORY[3]	2s*3	-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*4			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[3]	1u*3	0	15	
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) FRAME_IMU_FF = (1<<6) OVERHEAT_STOP_MOTORS = (1<<7) CENTER_YAW_AT_STARTUP = (1<<8) SWAP_RC_SERIAL_UART_B = (1<<9) UART_B_SERIAL_API = (1<<10) BLINK_BAT_LEVEL = (1<<11) ADAPTIVE_GYRO_TRUST = (1<<12)
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) MENU_BUTTON_DISABLE_FOLLOW = (1<<9) TIMELAPSE_FRAME_FIXED = (1<<10) RC_KEEP_MIX_RATE = (1<<11)
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 * implemented in special builds of firmware only
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 * used for 2-axis systems only
CUR_IMU	1u			currently selected IMU IMU_TYPE_MAIN=1 IMU_TYPE_FRAME=2
CUR_PROFILE_ID	1u			profile ID which is currently active in the controller, 0...4

CMD_READ_PARAMS_EXT – read/write system configuration part 2

Name		Type	Min	Max	Possible values, remarks
PROFILE_ID		1u			profile ID to read or write. To access current (active) profile, specify 255. Possible values: 0..4
axis = (1..3)	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[3]		2u*3	0	1000	Low-pass filter -3dB cut-off frequency, Hz
FILTERS_EN[3]		1u*3			Set of bits (0 – disable all): EN_NOTCH1 = 1 EN_NOTCH2 = 2 EN_NOTCH3 = 4 EN_LPF = 8
ENCODER_OFFSET[3]		2s*3			*Units: 0,02197265625 degree*
ENCODER_FLD_OFFSET[3]		2s*3			*Units: 0,02197265625 degree*
ENCODER_MANUAL_SET_TIME[3]		1u*3	0	255	*Units: 10ms*
MOTOR_HEATING_FACTOR[3]		1u*3	0	255	
MOTOR_COOLING_FACTOR[3]		1u*3	0	255	
RESERVED		2b			

FOLLOW_INSIDE_DEADBAND	1u	0	255	
MOTOR_MAG_LINK[3]	1u*3	0	255	Deprecated, replaced by the MOTOR_MAG_LINK_FINE
MOTOR_GEARING[3]	2u*3			Real number encoded as 8.8 fixed point (1.0f → 256)
ENCODER_LIMIT_MIN[3] ENCODER_LIMIT_MAX[3]	1s*3 1s*3	-127	127	*Units: 3 degree* Startig from ver. 2.61 is deprecated, replaced by the FRAME_CAM_ANGLE_MIN.
NOTCH1_GAIN[3] NOTCH2_GAIN[3] NOTCH3_GAIN[3]	1s*3 1s*3 1s*3	-100	100	Notch gain, in dB (positive – notch, negative – peak filter)
BEEPER_VOLUME	1u	0	255	
ENCODER_GEAR_RATIO[3]	2u*3			*Units: 0.001*
ENCODER_TYPE[3]	1u*3			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 ENC_TYPE_RLS_ORBIS = 14 TYPE_RLS_ORBIS_PWM = 15 Bit 4: SKIP_DETECTION = 1 Bit 7: ENCODER_IS_GEARED = 1
ENCODER_CFG[3]	1u*3			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[3]	1u*3	0	255	
OUTER_I[3]	1u*3	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	-90	90	*Units: 1 degree*

ACC_LPF_FREQ	2u	0	1000	*Units: 0.01Hz*
D_TERM_LPF_FREQ[3]	1u*3	0	60	*Units: 10Hz*

CMD_READ_PARAMS_EXT2 – read/write system configuration part 3

Name		Type	Min	Max	Possible values, remarks
PROFILE_ID		1u			profile ID to read or write. To access current (active) profile, specify 255. Possible values: 0..4
channel = (1..2)	MAV_SRC	1u			Disabled=0 UART1=1 RC_SERIAL=2 UART2=3 USB VCP=4
	MAV_SYS_ID	1u	0	255	
	MAV_COMP_ID	1u	0	255	

	MAV_CFG_FLAGS	1u			FLAG_BAUD_MASK = ((1<<0) | (1<<1) | (1<<2)) // baud rate idx 0..5
					FLAG_PARITY_EVEN = (1<<3) // even parity
					FLAG_HEARTBEAT = (1<<4) // send heartbeat
					FLAG_DEBUG = (1<<5) // send debug to GUI
					FLAG_RC = (1<<6) // use RC values

	MAV_RESERVED	4b			
MOTOR_MAG_LINK_FINE[3]		2u*3	0	65000	*Units: 0.01*
ACC_LIMITER[3]		1u*3	0	200	*Units: 5 degrees/sec²*
PID_GAIN[3]		1u*3	0	255	pid_gain_float[axis] = 0.1 + PID_GAIN[axis]*0.02
FRAME_IMU_LPF_FREQ		1u	0	200	*Units: Hz*
AUTO_PID_CFG		1u			See 'CFG_FLAGS' in the CMD_AUTO_PID
AUTO_PID_GAIN		1u	0	255	See 'GAIN_VS_STABILITY' in the CMD_AUTO_PID
FRAME_CAM_ANGLE_MIN[3] FRAME_CAM_ANGLE_MAX[3]		2s*3 2s*3			Soft limits for motor's angles *(frw. ver. 2.61+)* *Units: 1 degree*
GENERAL_FLAGS2		2u			*(frw. ver. 2.61+)* SEARCH_LIMIT_ROLL = (1<<0) SEARCH_LIMIT_PITCH = (1<<1) SEARCH_LIMIT_YAW = (1<<2) *(frw. ver. 2.62b7+)* AUTO_CALIBRATE_MOMENTUM = (1<<3) USE_MOMENTUM_FEED_FORWARD = (1<<4) MOTORS_OFF_AT_STARTUP = (1<<5) FC_BELOW_OUTER = (1<<6)
AUTO_SPEED		1u	1	255	*(frw. ver. 2.61+)*

				Speed used in automated tasks. The same range as for the RC_SPEED parameter
AUTO_ACC_LIMITER	1u	1	255	*(frw. ver. 2.61+)* Acceleration limiter used in automated tasks. The same range as for ACC_LIMITER parameter *Units: 5 degrees/sec²*
IMU_ORIENTATION_CORR[3]	2s			*(frw. ver. 2.61+)* The rotation angle of correction of main IMU sensor misalignment over its local X,Y,Z axis. *Units: 0.01 degrees*
TIMELAPSE_TIME	2u			*(frw. ver. 2.60+)* Time for the time-lapse motion sequence *Units: seconds*
EMERGENCY_STOP_RESTART_DELAY	2u			*Units: ms*
TIMELAPSE_ACC_PART	1u	0	200	*Units: 0.2%*
MOMENTUM[3]	2u*3			*(frw.ver. 2.62b7+)*
MOMENTUM_CALIB_STIMULUS[3]	1u*3	1	255	*(frw.ver. 2.62b7+)*
MOMENTUM_ELITPICITY[3]	1u*3	1	255	*(frw.ver. 2.62b7+)* *Units: 0.05*
FOLLOW_RANGE[3]	1u*3	1	180	*(frw.ver. 2.62b7+)* *Units: degrees*

STAB_AXIS[3]	1u*3			*(frw.ver. 2.62b7+)*
				Bits0..1: axis assigned for each motor:
				`0 – default`
				`1 – ROLL`
				`2 - PITCH`
				`3 – YAW`
				
				Bits2..4: enable automatic selection of best matching axis:
				`bit2: ROLL`
				`bit3: PITCH`
				`bit4: YAW`

RESERVED	74b			

CMD_REALTIME_DATA_3 - receive real-time data

Name		Type	Min	Max	Possible values, remarks
axis = (1..3)	ACC_DATA	2s			raw data from accelerometer sensor *Units: 1/512 G*
	GYRO_DATA	2s			raw data from gyroscope sensor *Units: 0,06103701895 degree/sec.*
SERIAL_ERR_CNT		2u	0	65535	

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 SUB_ERR_I2C_ERRORS = 1 SUB_ERR_DRV_OTW = 2 SUB_ERR_DRV_FAULT = 3 SUB_ERR_ENCODER_IMU_ANGLE = 4 SUB_ERR_CALIBRATION_FAILED = 5 SUB_ERR_INTERNAL_SYSTEM_ERROR = 6 SUB_ERR_ENCODER_CALIB_BAD_SCALE = 7 SUB_ERR_OVER_TEMPERATURE = 8 SUB_ERR_BAD_MOTOR_POLES_INVERT = 9 SUB_ERR_NOT_ENOUGH_MEMORY = 10 SUB_ERR_IMU_SENSOR_NOT_RESPONDING = 11 SUB_ERR_MOTOR_OVERHEAT_PROTECTION = 13 SUB_ERR_MOTOR_IS_LOCKED = 14 SUB_ERR_BAD_IMU_HEALTH = 15 SUB_ERR_INFINITE_RESET = 16 SUB_ERR_WRONG_INITIAL_POSITION = 17
IMU_REFERENCE_SRC	1u			*(frw. ver. 2.62b7+)* bits 0..2: Z (gravity) reference source bits 3..5: H (heading) reference source bit6: if set, frame heading is frozen and used for H-reference Possible values for reference: REF_NO = 0 - no reference REF_INTERNAL = 1 - reference is provided by the internal sensor like accelerometer or magnetometer REF_EXTERNAL = 2 - reference is set externally by the serial API or MavLink REF_TRANSLATE = 3 - translate reference from other IMU (frame -> main, main -> frame)
FRAME_IMU_REFERENCE_SRC	1u			*(frw. ver. 2.62b7+)* see IMU_REFERENCE specification
RESERVED	1b			
RC_ROLL RC_PITCH RC_YAW	2s 2s 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 2s	1000	2000	External FC PWM values. May be zero if their inputs are mapped to RC control or command.
IMU_ANGLE[3]	2s*3	-32768	32767	IMU angles in 14-bit resolution per full turn *Units: 0,02197265625 degree*

FRAME_IMU_ANGLE[3]	2s*3	-32768	32767	Angles measured by the second IMU (if present), in 14-bit resolution. *Units: 0,02197265625 degree*
TARGET_ANGLE[3]	2s*3	-32768	32767	Target angles, in 14-bit resolution *Units: 0,02197265625 degree*
CYCLE_TIME	2u			*Units: microseconds*
I2C_ERROR_COUNT	2u			Number of registered errors on I2C bus
ERROR_CODE	1u			deprecated, replaced by the SYSTEM_ERROR variable
BAT_LEVEL	2u			Battery voltage *Units: 0.01 volt*
RT_DATA_FLAGS	1u			bit0 set - motors are turned ON

CUR_IMU	1u			Currently selected IMU that provides angles and raw sensor data
				`IMU_TYPE_MAIN=1`
				`IMU_TYPE_FRAME=2`

CUR_PROFILE	1u	0	4	Currently selected profile
MOTOR_POWER[3]	1u*3	0	255	

CMD_REALTIME_DATA_4 - receive extended version of real-time data

Name	Type	Min	Max	Possible values, remarks
...The beginning of the message includes all data from the CMD_REALTIME_DATA_3				
STATOR_ROTOR_ANGLE[3]	2s*3			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*
RESERVED	1b			
BALANCE_ERROR[3]	2s*3	-512	512	Error in balance (0 – perfect balance, 512 - 100% of the motor power is required to hold a camera)
CURRENT	2u			Actual current consumption. *Units: mA*
MAG_DATA[3]	2s*3	-1000	1000	Raw data from magnetometer *Units: relative, calibrated for current environment to give ±1000 for each axis.*
IMU_TEMPERATURE FRAME_IMU_TEMPERATURE	1s 1s	-127	127	Temperature of IMU sensors. *Units: Celsius*
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
MOTOR_OUT[3]	2s*3	-10000	10000	Motor effective output, proportional to torque. Max. value of ±10000 equals to applying full power. *(encoder firmware ver. 2.61+)*
RESERVED	30b			

CMD_CONFIRM – confirmation of previous command or finished calibration

Name	Type	Min	Max	Possible values, remarks
CMD_ID	1u			Command ID to confirm
DATA	1u or 2u			DATA depends on command to be confirmed

CMD_ERROR – error executing previous command

Data depends on error type.

Name	Type	Min	Max	Possible values, remarks
ERROR_CODE	1u			
ERROR_DATA	4b			

CMD_GET_ANGLES - Information about actual gimbal control state

Name		Type	Min	Max	Possible values, remarks
axis = (1..3)	IMU_ANGLE	2s			IMU angles in 14-bit resolution per full turn `Units: 0,02197265625 degree`
	TARGET_ANGLE	2s			Target angles, in 14-bit resolution `Units: 0,02197265625 degree`
	TARGET_SPEED	2s			Target speed that gimbal should keep, over Euler axes `Units: 0,1220740379 degree/sec`

CMD_GET_ANGLES_EXT - Information about angles in different format

Name		Type	Min	Max	Possible values, remarks
axis = (1..3)	IMU_ANGLE	2s			IMU angles in 14-bit resolution per full turn `Units: 0,02197265625 degree`
	TARGET_ANGLE	2s			Target angles, in 14-bit resolution `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`
	RESERVED	10b			

CMD_READ_PROFILE_NAMES_3 – receive profile names from EEPROM

Name	Type	Min	Max	Possible values, remarks
PROFILE_NAME[5]	48b*5			Each name is encoded in UTF-8 format and padded with '\0' character to 48 byte size

CMD_I2C_READ_REG_BUF – result of reading from I2C device

Name	Type	Min	Max	Possible values, remarks
DATA	1..255b			Data length depends on the DATA_LEN parameter in the request.

CMD_AUTO_PID – progress of PID auto tuning

Name		Type	Min	Max	Possible values, remarks
P[3]		1u*3			
I[3]		1u*3			
D[3]		1u*3			
LPF_FREQ[3]		2u*3			
ITERATION_CNT		2u			
axis = (1..3)	TRACKING_ERROR	float			Current error between the target and actual system response
	RESERVED	6b			
RESERVED		10b			

CMD_DEBUG_VARS_INFO_3 – receive a specification of the debug variables

Name		Type	Min	Max	Possible values, remarks
DEBUG_VARS_NUM		1u	1	255	Number of variables in this messages
var = (1...DEBUG_VARS_NUM)	VAR_NAME	string			1st byte is size, following by the ASCII characters. Note that '\0' character is not required at the end of the string.
	VAR_TYPE	1u			0..3bits - type: 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) 4..7bits - flags:

					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)
	RESERVED	2b			

CMD_DEBUG_VARS_3 – values of debug variables reflecting a state of the system.

The number of variables and their types are not strictly defined and may vary depending on the firmware version. Use CMD_DEBUG_VARS_INFO_3 to obtain a specification of the variables in run-time.

Name	Type	Min	Max	Possible values, remarks
VAR_VALUE[N]	?			size and type of each variable is encoded by the CMD_DEBUG_VARS_INFO_3 structure

CMD_READ_EXTERNAL_DATA – receive user data, stored in the EEPROM

External systems can use this area to store their configurations.

Name	Type	Min	Max	Possible values, remarks
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 for 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. “Analog” type is used to adjust parameter by RC signal. MIN_VAL and MAX_VAL specify a working range, that is mapped to a native range of particular parameter.

Name		Type	Min	Max	Possible values, remarks
slot = (1..10)	TRIGGER_SRC_CH	1u			See the *RC_MAP_ROLL* parameter definition
	TRIGGER_ACTION[5]	1u*5			See the *CMD_EXECUTE_MENU* command for a list of available actions
slot = (1..15)	ANALOG_SRC_CH	1u			See the *RC_MAP_ROLL* parameter definition
	VAR_ID	1u			bits0..6: the ID of variable. Full list of adjustable variables is given in the Appendix B bit7: if set, the value is processed as a "multiplier" for a given variable. *(frw. ver. 2.62b6+)*
	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 a portion of data read from EEPROM at the specified address.

Name	Type	Min	Max	Possible values, remarks
ADDR	4u			Address of a portion of data, 64-byte aligned
DATA	?			All remaining bytes are counted as data. Size is specified in the CMD_EEPROM_READ outgoing command.

CMD_CALIB_INFO – receive information required for the "Calibration helper" dialog window.

Name	Type	Min	Max	Possible values, remarks
PROGRESS	1u	0	100	Progress of operation in percents
IMU_TYPE	1u			1 – main IMU, 2 – frame IMU
ACC_DATA[3]	2s*3			See the ACC_DATA parameter in the CMD_REALTIME_DATA_3
GYRO_ABS_VAL	2u			Amplitude of gyro signal
ACC_CUR_AXIS	1u	0	2	ACC axis to be calibrated
ACC_LIMITS_INFO	1u			Bit set of calibrated limits, where bits 0...5 corresponds to the index in array [+X,-X,+Y,-Y,+Z,-Z]
IMU_TEMP_CELS	1s	-127	127	IMU temperature, Celsius
TEMP_CALIB_GYRO_ENABLED	1u	0	1	Set to 1 if gyro temperature calibration is enabled
TEMP_CALIB_GYRO_T_MIN_CELS TEMP_CALIB_GYRO_T_MAX_CELS	1s 1s	-127	127	Range of temperature calibration *Units: Celsius*
TEMP_CALIB_ACC_ENABLED				Set to 1 if ACC temperature calibration is enabled
TEMP_CALIB_ACC_SLOT_NUM[6]	1u*6	0	3	The number of calibrated temperature slots for accelerometer for each limit, in order [+X,+Y,+Z,-X,-Y,-Z]
TEMP_CALIB_ACC_T_MIN_CELS TEMP_CALIB_ACC_T_MAX_CELS	1s 1s			Range of temperature calibration *Units: Celsius*
H1_ERR_LENGTH	1u	0	255	The length of error vector between estimated and referenced

				heading vectors. *Unit vector=100*
RESERVED	7b			

CMD_READ_FILE – result of reading file from internal filesystem

In case of success:

Name	Type	Min	Max	Possible values, remarks
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 the end of file is reached

In case of errors:

Name	Type	Min	Max	Possible values, remarks
ERR_CODE	1u			see error definitions in the CMD_WRITE_FILE command

CMD_SCRIPT_DEBUG – state of execution of user-written script

Name	Type	Min	Max	Possible values, remarks
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.

Name	Type	Min	Max	Possible values, remarks
Z1_VECT[3]	4f*3	-1.0f	1.0f	Unit vector that points down (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_REALTIME_DATA_CUSTOM – configurable realtime data *(frw. ver. 2.60+)*

Name	Type	Min	Max	Possible values, remarks
TIMESTAMP_MS	2u			Timestamp in milliseconds
The set of varaibles below depends on requested data, see the CMD_REALTIME_DATA_CUSTOM request specifications				
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_ADJ_VARS_STATE – receive the state of adjustable variables

Name	Type	Min	Max	Possible values, remarks
Firmware ver. prior to 2.62b5				
TRIGGER_RC_DATA	2s	-500	500	RC signal for the "trigger" variable slot
TRIGGER_ACTION	1u	0	255	ID of the triggered action. The full set of actions is given in the specification of MENU_CMD_1..5 parameters
ANALOG_RC_DATA	2s	-500	500	RC signal for the "analog" variable slot
ANALOG_VALUE	4s			Current value of the variable after all calculations
RESERVED	6b			
Firmware ver. 2.62b5+				
TRIGGER_RC_DATA	2s	-500	500	RC signal for the "trigger" variable slot
TRIGGER_ACTION	1u	0	255	ID of the triggered action. The full set of actions is given in the specification of MENU_CMD_1..5 parameters
ANALOG_SRC_VALUE	2s	-16384	16384	Signal value requested in the ANALOG_SRC_ID
ANALOG_VAR_VALUE	4f			Value of variable requested in the ANALOG_VAR_ID
LUT_SRC_VALUE	2s	-16384	16384	Signal value requested in the LUT_SRC_ID. Always encoded in a range -16384..16384.
LUT_VAR_VALUE	4f			Current value of variable requested in the LUT_VAR_ID

Outgoing commands

CMD_BOARD_INFO – request board and firmware information

Simple format: no parameters

Extended format:

Name	Type	Min	Max	Possible values, remarks
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 is not checked

CMD_BOARD_INFO_3 – request additional board information

No parameters

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

No parameters

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

No parameters

CMD_CALIB_ACC – calibrate accelerometer
CMD_CALIB_GYRO – calibrate gyroscope
CMD_CALIB_MAG – calibrate magnetometer

Simple format: no parameters. Starts regular calibration of currently active IMU, selected by the CMD_SELECT_IMU_3 command.

Extended format:

Name	Type	Min	Max	Possible values, remarks
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

No parameters

CMD_USE_DEFAULTS – reset to factory defaults

Name	Type	Min	Max	Possible values, remarks
PROFILE_ID	1u	0	4	profile ID to reset. Special values: 253 – erase EEPROM 254 – reset currently selected profile

CMD_CALIB_POLES – calibrate poles and direction

No parameters

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

Name	Type	Min	Max	Possible values, remarks
PROFILE_ID	1u	0	4	profile ID to load. If value >4, currently selected profile is loaded.

CMD_WRITE_PARAMS,
CMD_WRITE_PARAMS_3 - write parameters to board and saves to EEPROM
CMD_WRITE_PARAMS_EXT – write extended parameters
CMD_WRITE_PARAMS_EXT2 – write extended parameters

Data structure is the same as for the corresponding CMD_READ_PARAMS_xx incoming command.

CMD_RESET – reset device

Simple format: no parameters. Resets the device without delay and confirmation

Extended format:

Name	Type	Min	Max	Possible values, remarks
CONFIRM	1u			0 – no confirmation 1 - command CMD_RESET will be sent back for confirmation
DELAY_MS	2u			After confirmation is sent, waits for a given time (in ms) before reset.

CMD_BOOT_MODE_3 – enter bootloader mode to upload firmware

Simple format: no parameters. Enters boot mode without delay and confirmation

Extended format:

Name	Type	Min	Max	Possible values, remarks
CONFIRM	1u			0 – no confirmation 1 - command CMD_RESET will be sent back for confirmation
DELAY_MS	2u			After confirmation is sent, waits for a given time (in ms) before reset. External application can free up resources and properly close the serial connection before controller enters boot mode.

CMD_CALIB_OFFSET – calibrate follow offset

No parameters

CMD_CALIB_BAT - calibrate internal voltage sensor

Name	Type	Min	Max	Possible values, remarks
ACTUAL_VOLTAGE	2u			*Units: 0.01V*

Confirmation is sent.

CMD_CONTROL – control gimbal movement

Name	Type	Min	Max	Possible values, remarks
Legacy format: mode is common for all axes				

CONTROL_MODE	1u			Bits 0..3 for mode, bits 4..7 for flags.
				
				Modes:
				
				`MODE_NO_CONTROL=0`
				If this mode is set for all axes, finish serial control and restore normal RC control. If set for single axis, does not change its current control mode.
				
				`MODE_SPEED=1`
				Camera travels with the given speed in the Euler coordinates until the next CMD_CONTROL command comes. Given angle is ignored.
				
				`MODE_ANGLE=2*`
				Camera travels to the given Euler angle with the fixed speed. Speed is decreased near target to keep control smooth. Low-pass filter may be applied for the same reason.
				
				`MODE_SPEED_ANGLE=3`
				Camera travels with the given speed. Additionally,

				controller keeps the given angle and fix accumulated error by the outer PI-loop. This mode allows the most precise type of control (see fig.1 for example), but it requires pretty fast update rate to keep it smooth, or apply low-pass filtering for speed and angle.
				
				`MODE_RC=4*`
				Angle parameter is used as RC signal and overrides any other signal source, assigned to this axis. Normal working range is -500..500.
				The flag `CONTROL_FLAG_AUTO_TASK` can affect this mode (see below).
				Prior to 2.61 frw. ver., 'SPEED' parameter is ignored.
				
				`MODE_ANGLE_REL_FRAME=5*`
				First, the neutral point of a camera relative to a frame is found in the Euler coordinates for a given axis. Than, the given angle value is add to this point, and camera travels to it. Note that the given angle does not relate to a particular motor, it relates to global Euler angles!
				
				Flags:
				
				`CONTROL_FLAG_AUTO_TASK=(1<<6)`
				(frw. ver. 2.62b7+)
				- If mode is one of the <**`MODE_ANGLE`**, **`MODE_ANGLE_REL_FRAME`**>, the task is processed with the speed and acceleration configured for *automated tasks*. If the SPEED parameter is provided, it's used instead. When all target angles are reached with the 1-degree tolerance, confirmation is sent: `CMD_CONFIRM(CMD_CONTROL, 1)`.
				Use this flag to move gimbal to a certain position as fast as possible, and receive confirmation when the target is reached.
				- If mode is **`MODE_RC`**, this flag forces a control in the "speed" mode, with the dead-band, trimming and inversion settings are NOT applied to the provided RC signal, but the LPF, Expo curve and ACC limiter are still applied. Use this flag to control gimbal from remote applications, where signal is well-defined and you need to have a direction of rotation that does not depend on gimbal's setings.
				
				`CONTROL_FLAG_HIGH_RES_SPEED=(1<<7)`
				(frw.ver 2.60+)
				Speed units changed to 0.001 deg/sec for extremely slow motion (like timelapse shooting)
				
				** In the control modes "MODE_ANGLE", "MODE_RC", and "MODE_ANGLE_REL_FRAME", if the "SPEED" parameter > 0, it is used instead of the default value defined by the RC settings. Relationship: SPEED = settings.RC_SPEED*16*

Extended format (firmware ver. 2.55b5+): mode is set independently for each axes				
CONTROL_MODE[3]	1u*3			see definition above
The remaining part is common for all formats				
axis = (1..3)	SPEED	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	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 for all axes, and all data set to zeros. All parameters that was changed by the CMD_CONTROL_CONFIG, will be restored to their default values.
- Optimal rate of sending this command is 50..100Hz. If the rate of CMD_CONTROL command is lower, use a low-pass filtering to prevent step-wise response. It can be set by the command CMD_CONTROL_CONFIG.
- Confirmation is sent on each CMD_CONTROL command. Additional confirmation is sent when the target angle is reached, if the flag "`CONTROL_FLAG_AUTO_TASK`" is set.
- See the Appendix A for a source code examples

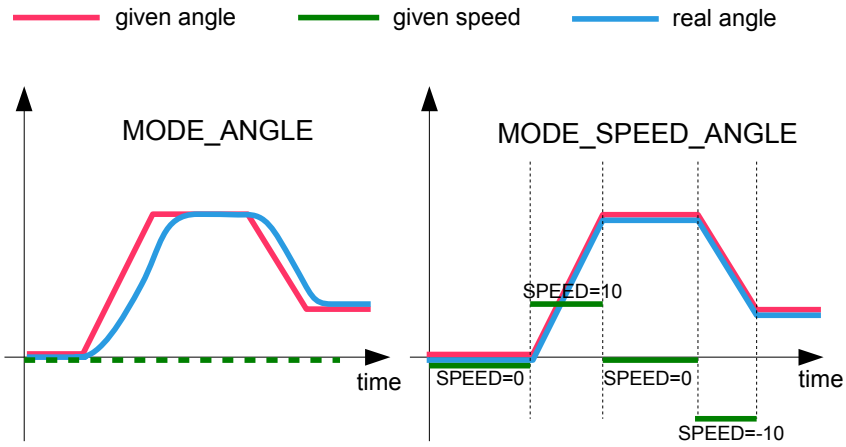

Fig.1: difference in control modes MODE_ANGLE and MODE_SPEED_ANGLE

CMD_CONTROL_CONFIG – configure the handling of CMD_CONTROL command *(frw. ver. 2.61+)*

Name	Type	Min	Max	Possible values, remarks
TIMEOUT_MS	2u	0	65535	0 - disable timeout >0 - if no CMD_CONTROL command will come in a given time on any channel, serial control will be finished. Default value after startup is 0 (no timeout). *Units: ms*
CH1_PRIORITY CH2_PRIORITY CH3_PRIORITY CH4_PRIORITY THIS_CH_PRIORITY	1u*5	0	255	Channels are counted in order: UART1, RC_SERIAL, UART2, USB_VCP (how they are named in the User Manual). THIS_CH means current port, where command is sent. Values: 0 - do not change the priority 1..255 - set the priority of a given channel. In case of concurrent CMD_CONTROL commands, they will be accepted only on a channel that has higher or equal priority than others. *Default value is 0 for all channels after startup.*

axis = (1..3)	ANGLE_LPF	1u	0	15	LPF factor for filtering the 'ANGLE' parameter in the modes "MODE_ANGLE", "MODE_SPEED_ANGLE". Helps to keep smooth control even if update rate is slow. *0 – do not change* *Default value is 0 – no filtering is applied.*
	SPEED_LPF	1u	0	15	LPF factor for filtering the 'SPEED' parameter in the modes "MODE_SPEED", "MODE_SPEED_ANGLE". Helps to keep smooth control even if update rate is slow. *0 – do not change* *Default value is 0 – no filtering is applied.*
	RC_LPF	1u	0	15	LPF factor for filtering RC signal in the mode "MODE_RC". Helps to keep smooth control even if update rate is slow. *0 – do not change.* *Default value is taken from the "RC_LPF" GUI parameter.*
	RESERVED	4b			
RC_EXPO_RATE		1u	0	100	Exponential curve for filtering RC signal in the mode "MODE_RC". *0 – do not change* *Default value is taken from the "RC_EXPO_RATE" GUI parameter.*
RESERVED		12b			

Confirmation is sent on success.

CMD_TRIGGER_PIN - trigger output pin

Name	Type	Min	Max	Possible values, remarks

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`
				`PIN_AUX1 = 16`
				`PIN_AUX2 = 17`
				`PIN_AUX3 = 18`
				`PIN_BUZZER = 32`
				`PIN_SSAT_POWER** = 33`
				
				** PIN_SSAT_POWER triggers 3.3V power line in the Spektrum connector (low state enables power)

STATE	1u			`LOW = 0` - pin can sink up to 40mA `HIGH = 1` - pin can source up to 40mA

Confirmation is sent only if pin is not occupied for other functions and was really triggered.

CMD_MOTORS_ON - switch motors ON

No parameters. Confirmation is sent

CMD_MOTORS_OFF - switch motors OFF

No parameters. Confirmation is sent

CMD_EXECUTE_MENU - execute menu command

Name	Type	Min	Max	Possible values, remarks
CMD_ID	1u			MENU_CMD_NO = 0 MENU_CMD_PROFILE1 = 1 MENU_CMD_PROFILE2 = 2 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 MENU_HOME_POSITION_SHORTEST = 38 MENU_CENTER_YAW_SHORTEST = 39 MENU_ROTATE_YAW_180 = 40 MENU_ROTATE_YAW_180_FRAME_REL = 41 MENU_SWITCH_YAW_180_FRAME_REL = 42 MENU_SWITCH_POS_ROLL_90 = 43 MENU_START_TIMELAPSE = 44 MENU_CALIB_MOMENTUM = 45 MENU_LEVEL_ROLL = 46

CMD_HELPER_DATA – provide helper data for AHRS system

Use this command to increase the precision of attitude estimation under certain conditions like curved or accelerated motion.

Legacy format (prior to frw. ver. 2.60):

Name	Type	Min	Max	Possible values, remarks

FRAME_ACC[3]	2s*3	-	-	Linear acceleration of the frame, [X,Y,Z] components in a coordinate system **COORD_SYS_GROUND_YAW_ROTATED** (see description below). *Units: 1g/512 ≈ 0,019160156 m/s²*
FRAME_ANGLE_ROLL FRAME_ANGLE_PITCH	2s 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.*

Extended format (frw. ver. 2.60+):

Name	Type	Min	Max	Possible values, remarks
FRAME_ACC[3]	2s*3	-	-	Linear acceleration of the frame, [X,Y,Z] components in a given coordinate system. *Units: 1g/512 ≈ 0,019160156 m/s²*
FRAME_ANGLE_ROLL FRAME_ANGLE_PITCH	2s 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.*
FLAGS	1u			bits 0..2: **COORD_SYS_GROUND_YAW_ROTATED = 1** Global system rotated with the camera over Z axis: Y-axis is aligned with the main IMU's Y-axis (points forward), X-axis points right, Z-axis points down (nadir) **COORD_SYS_GROUND = 2*** Global system: Y-axis matches true North, X-axis matches true East, Z-axis matches nadir **COORD_SYS_FRAME = 3**** System that is linked to the gimbal's outer frame: Y-axis matches frame's "forward", X-axis matches frame's "right", Z-axis matches frame's "down". * *A magnetometer sensor should be installed and calibrated to give global reference for the main IMU. If no magnetometer present, Y-axis points arbitrary direction.* ** *One of the following conditions should be satisfied:* *- a 2nd frame-mounted IMU and YAW encoder in the regular firmware* *- 3 encoders and the "encoder" version of firmware* bit7: Use FRAME_HEADING parameter as a heading reference to align the IMU's local coordinate system to the Earth, or to compensate gyro drift by the YAW axis if frame is fixed. If bit is not set, FRAME_HEADING is ignored (frw. ver. 2.62b7+)
FRAME_SPEED[3]	2s*3	-	-	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 this information. *Units: 0,06103701895 degree/sec*
FRAME_HEADING (frw. ver. 2.62b7+)	2s	-16384	16384	Angle of the frame relative to North by the YAW axis. On first occurrence, YAW angle will be updated, taking into account the position of the main IMU relative to a frame. Then it

				will be used only as a reference for a gyro drift correction. If frame is fixed, it's enough to set this value once. But if frame is moving, it should be update with some adequate rate to reflect the rotation. Remarks: *bit7 in the FLAGS parameter should be set to use this value. *Provided angle may be wrapped to +-180 degrees or 0..360 degrees. *Special value of 32767 stops the use of this reference and makes IMU heading unreferenced. *Units: 0,02197265625 degree.*
RESERVED	1b			

It's enough to feed fresh ACC and angles data with the pretty low rate 5-20 Hz, because strong low-pass filter is applied internally. If the FRAME_SPEED data need to be provided, data rate should be much higher, up to 125 Hz.

How to ensure that the ACC correction is applied properly:

1. Temporarily set the "ACC LPF" filter parameter in the GUI to 5-10Hz – it will remove noise but keep fast reaction of "IMU_G_ERR" variable in the "Monitoring" tab of the GUI.
2. Without motion, when you tilt frame, FRAME_ACC[] that is passed in this command, should have all components close to zero. The IMU_G_ERR variable should be near zero, too.
3. Without correction, when you shake gimbal - you will see that the IMU_G_ERR changes significantly. With the correction applied, when you shake gimbal, IMU_G_ERR always stays near zero - it means that the external accelerations are compensated.
4. When you rotate frame relative to Earth in all directions, or rotate camera relative to frame, the 3rd test is still passed correctly – it means that the ACC correction vector is translated from the frame to the main IMU properly.

CMD_GET_ANGLES, CMD_GET_ANGLES_EXT - Request information related to IMU angles and RC control state

No parameters.

CMD_SELECT_IMU_3 – Select which IMU to configure

Name	Type	Min	Max	Possible values, remarks
IMU_TYPE	1u			`IMU_TYPE_MAIN=1` `IMU_TYPE_FRAME=2`

If the selected IMU is not connected, command is ignored.

CMD_READ_PROFILE_NAMES_3 – Request profile names stored in EEPROM

No parameters

CMD_WRITE_PROFILE_NAMES_3 – Writes profile names to EEPROM

Name	Type	Min	Max	Possible values, remarks
PROFILE_NAME[5]	48b*5			Each name is encoded in UTF-8 format and padded with '\0' character to 48 byte size

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.
To save updated parameters permanently, use the CMD_SAVE_PARAMS_3 command.

Name		Type	Min	Max	Possible values, remarks
NUM_PARAMS		1u	1	40	Number of parameters in command
for N = (1..NUM_PARAMS)	PARAM<N>_ID	1u			ID of parameter. See the Appendix B for a list of available variables.
	PARAM<N>_VALUE ...	4b			Value depends on type of parameter. 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 are packed according to IEEE-754.

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.
On success, CMD_SET_ADJ_VARS_VAL is sent in response.

Name		Type	Min	Max	Possible values, remarks
NUM_PARAMS		1u	1	40	Number of parameters in command
for N = (1..NUM_PARAMS)	PARAM<N>_ID	1u			ID of parameter. See the Appendix B for a list of available variables.

CMD_SAVE_PARAMS_3 – Saves current params from volatile memory to EEPROM

No parameters.

Use this command to save parameters updated by the "Adjustable Variables", permanently to EEPROM, to the active profile slot.

CMD_AUTO_PID – Starts automatic PID calibration

Name	Type	Min	Max	Possible values, remarks
PROFILE_ID	1u			switch to this profile before start of the calibration and save result there
CFG_FLAGS	1u			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 - if set, sends tuned parameters to this channel AUTO_PID_CFG_KEEP_CURRENT = 16 - if set, starts from existing settings. If not set, starts from zero AUTO_PID_CFG_TUNE_LPF_FREQ = 32 - if set, tunes LPF filters, too AUTO_PID_CFG_ALL_PROFILES = 64 - if set, updates parameters in all profiles
GAIN_VS_STABILITY	1u	0	255	
RESERVED	16b			

CMD_SERVO_OUT – Output PWM signal on the specified pins

Name	Type	Min	Max	Possible values, remarks
SERVO_TIME[8]*	2s*8	-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. ** Although command takes 8 values, the real number of hardware outputs depends on board version and may be less.*

CMD_I2C_WRITE_REG_BUF – writes data to any device connected to I2C line

Name	Type	Min	Max	Possible values, remarks
DEVICE_ADDR	1u			bit0: I2C port 0 for external port (IMU sensor is connected) 1 for internal port (EEPROM) bit1..7: I2C address

REG_ADDR	1u			register to write
DATA	?			remaining bytes are counted as data

On successful writing, confirmation CMD_CONFIRM is sent in response.

CMD_I2C_READ_REG_BUF – requests reading from any device connected to I2C line

Name	Type	Min	Max	Possible values, remarks
DEVICE_ADDR	1u			bit0: I2C port 0 for external port (IMU sensor is connected) 1 for internal port (EEPROM) bit1..7: I2C address
REG_ADDR	1u			register to write
DATA_LEN	1u			length of data to read

On successful reading, CMD_I2C_READ_REG_BUF command is sent in response.

CMD_DEBUG_VARS_INFO_3 – request information about debug variables

No parameters.

CMD_DEBUG_VARS_3 – request values of debug variables

No parameters.

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

Name	Type	Min	Max	Possible values, remarks
DATA	128b			

Confirmation is sent on success.

CMD_READ_EXTERNAL_DATA – request user data, stored in the EEPROM

No parameters.

CMD_READ_EXTERNAL_DATA is sent in response.

CMD_API_VIRT_CH_CONTROL – update a state of 32 virtual channels.

These channels can be found in the GUI as “API_VIRT_CHXX” and assigned as RC source to control camera or to do other tasks.

Name	Type	Min	Max	Possible values, remarks
API_VIRT_CH[32]	2s*3	-500	500	Value may go slightly outside these limits.

	2			Use a special value “-10000” to mark that channel has “undefined” state (its treated as “signal lost” like with the regular RC inputs)

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

Name	Type	Min	Max	Possible values, remarks
ADDR	4u			Address should be aligned to 64
DATA	?			All remaining bytes counted as data, arbitrary size.

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.

Name	Type	Min	Max	Possible values, remarks
ADDR	4u			address should be aligned to 64
SIZE	2u			size should be aliged to 64 bytes and less than 256

On success, CMD_EEPROM_READ is sent in response.

CMD_CALIB_INFO – request information required for the "Calibration helper" dialog window

Name	Type	Min	Max	Possible values, remarks
IMU_TYPE	1u			1 – main IMU, 2 – frame IMU
RESERVED	11b			

On success, CMD_CALIB_INFO is sent in response.

CMD_READ_FILE – read file from internal filesystem

This command reads a portion of data from a file with the identifier FILE_ID, started at PAGE_OFFSET pages (1page = 64byte) and to the end of file, but not more then MAX_SIZE bytes. Size of a portion should not exceed maximum allowed command data length (256 bytes). The result or error code is sent in the incoming command CMD_READ_FILE.

Name	Type	Min	Max	Possible values, remarks
FILE_ID	2u			
PAGE_OFFSET	2u			offset from the beginning, in pages. 1 page = 64 bytes.
MAX_SIZE	2u			
RESERVED	14b			

CMD_WRITE_FILE – write file to internal filesystem

This command writes a portion of data to a file with the 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.

Name	Type	Min	Max	Possible values, remarks
FILE_ID	2u			
FILE_SIZE	2u			Full size of a file
PAGE_OFFSET	2u			offset from the beginning, in pages. 1 page = 64 bytes.
DATA	?			All remaining bytes are counted as data. Size should be less then FILE_SIZE parameter. If data is empty, file will be 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

Name	Type	Min	Max	Possible values, remarks
MODE	1u			0 – stop 1 – start 2 – start with debug information is sent back in the CMD_SCRIPT_DEBUG
SLOT	1u			

RESERVED	32b			

CMD_AHRS_HELPER – send or request attitude of the IMU sensor.

Use this command to provide a reference or replace the attitude estimated by the internal IMU sensor, by the attitude from a high-grade external IMU. Send this command with the 20-50 Hz rate.

Name	Type	Min	Max	Possible values, remarks
MODE	1u			bit0: 0 – get, 1 – set bit1: 0 – main IMU, 1 – frame IMU bit2: if set, use as reference. Any internal reference (if present) is disabled. 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 down (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 the gyroscope sensor's zero bias manually

Name	Type	Min	Max	Possible values, remarks
IMU_TYPE	1u			0 – main IMU, 1 – frame IMU
GYRO_ZERO_CORR[3]	2s*3			Zero offset for each axis in order X, Y, Z *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* – a commands sent by the controller with the fixed rate without request *(frw. ver. 2.60+)*

For each serial interface, only one unique combination of CMD_ID + CONFIG bytes may be registered. If the data stream is already registered, it will be updated. To unregister it, specify INTERVAL_MS=0. The total number of data streams over all serial interfaces is limited to 10.
Take care of the serial bandwidth: if data flow exceeds bandwidth, particular samples may be skipped. The same is true when the TX buffer is full when sending long commands in parallel, like CMD_READ_PARAMS_3.
The interval is maintained with the +-1ms tolerance for the individual sample, but the averaged sample rate exactly matches to specified.

Name	Type	Min	Max	Possible values, remarks
CMD_ID	1u			Command ID to be sent by this data stream. All supported commands are listed for the "CONFIG" parameter below.
INTERVAL_MS	2u			Interval between messages, in milliseconds. Value 1 means each cycle (0.8ms) Send value 0 to unregister data stream.
CONFIG	8b			Configuration specific to each command: **CMD_REALTIME_DATA_3** **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)
RESERVED	10b			

If the data stream is successfully registered or updated, the CMD_CONFIRM is sent in answer.

CMD_REALTIME_DATA_CUSTOM – request configurable realtime data *(frw. ver. 2.60+)*

Name	Type	Min	Max	Possible values, remarks
FLAGS	4u			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*
RESERVED	6b			

CMD_BEEP_SOUND – play melody by motors or emit standard beep sound

Name	Type	Min	Max	Possible values, remarks

MODE	2u			Pre-defined melodies:
				`BEEPER_MODE_CALIBRATE = (1<<0)`
				`BEEPER_MODE_CONFIRM = (1<<1)`
				`BEEPER_MODE_ERROR = (1<<2)`
				`BEEPER_MODE_CLICK = (1<<4)`
				`BEEPER_MODE_COMPLETE = (1<<5)`
				`BEEPER_MODE_INTRO = (1<<6)`
				Custom melody:
				`BEEPER_MODE_CUSTOM_MELODY = (1<<15)`

NOTE_LENGTH	1u	1	255	The length of each note in the custom melody mode. *Units: 8ms*
DECAY_FACTOR	1u	0	15	Set the envelope "attack-decay" after each pause,that makes sound more natural. The bigger value, the longer decay. 0 - no decay. *Note: envelope takes effect only in the encoder-enabled firmware or when motors are OFF. The same is true for the 'volume' parameter in the GUI.
RESERVED	8b			
NOTE_FREQ_HZ[N]	2u*N	554	21000	Array of 2u elements, size N = 0..30, - melody to play if mode=BEEPER_MODE_CUSTOM_MELODY. Special value 21000 used to make pause and restart envelope. *Units: Hz*

Example1: simple melody with short B5, D6, G6 notes and envelope:
```
00 80 05 03 00 00 00 00 00 00 00 00 DB 03 DB 03 08 52 DB 03 DB 03 08 52 96 04 96 04 08
52 1F 06 1F 06 1F 06 1F 06 1F 06
```

Example2: standard "calibration" sound:
```
01 00 00 03 00 00 00 00 00 00 00 00
```

Example3: single beep 1 second at 3kHz:
```
00 80 7D 00 00 00 00 00 00 00 00 00 B8 0B
```

CMD_ENCODERS_CALIB_OFFSET_4 - calibrate offset of encoders

No parameters.

CMD_ENCODERS_CALIB_FLD_OFFSET_4 - start field offset calibration of encoders

Simple format: no parameters

Extended format (*frw. ver. 2.62b6+*):

Name	Type	Min	Max	Possible values, remarks
CALIB_ANGLE[3]	2s*3	-16384	16384	Angle range to move during calibration. Default is 40º. *Units: 0,02197265625 degree.*

CMD_ADJ_VARS_STATE – request the state of adjustable variable in the given trigger and analog slots.

Firmware ver. prior to 2.62b5:

Name	Type	Min	Max	Possible values, remarks
TRIGGER_SLOT	1u	0	9	
ANALOG_SLOT	1u	0	14	

Firmware ver. 2.62b5+:

Name	Type	Min	Max	Possible values, remarks
TRIGGER_SLOT	1u	0	9	"Trigger" slot number to show its state
ANALOG_SRC_ID	2u			Signal source to show its value
ANALOG_VAR_ID	1u			Variable ID to show its value
LUT_SRC_ID	2u			Signal source to show its value
LUT_VAR_ID	1u			Variable ID to show its value

CMD_CALIB_ORIENT_CORR – start the calibration of sensor misalignment correction *(frw. ver. 2.61+)*

Name	Type	Min	Max	Possible values, remarks
RESERVED	16b			

Confirmation is sent immediately. After calibration is finished, CMD_READ_PARAMS_EXT2 is sent with new values in the IMU_ORIENTATION_CORR[3].

CMD_CALIB_ACC_EXT_REF – refine the accelerometer calibration of the main IMU sensor *(frw. ver. 2.62b7+, encoders)*

Use this command to refine the ACC calibration in the main IMU sensor by providing the reference ACC vector from the external well-calibrated IMU in the frame's coordinates. By using three encoders, gimbal controller is able to convert it to the main IMU's local coordinates, compare to measured ACC vector and use it to refine existing calibration: zero offset for two horizontal axes and scale factor for the vertical axis.

Name	Type	Min	Max	Possible values, remarks
ACC_REF[3]	2s*3			Reference ACC vector [X,Y,Z] in gimbal frame's coordinates (X-

				axis points right, Y-axis points forward, Z-axis points down relative to frame). *Units: 1g/512 ≈ 0,019160156 m/s²*
RESERVED	14b			

Conditions:

- One of the sensor's axis should be aligned to a gravity vector with the 20-degree tolerance
- Existing ACC calibration should be good enough

Possible usage scenario:

1. Rotate gimbal to a leveled position by the CMD_CONTROL and run this command – X,Y-axis offset will be refined
2. Tilt gimbal 90-degree down and run it again – Z-axis offset and Y-axis scale will be refined.
3. Return gimbal back to leveled position and run it again – Z-axis scale will be refined. This is enough to have correct ACC readings inside the working range ROLL=0, PITCH = [0..90].

Calibration takes about 0.5 seconds (controller averages multiple data samples to reduce noise).
Confirmation is sent only if all conditions are satisfied.

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 are written in C++ for Arduino platform only.

Libraries

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

Appendix B: Definition of dynamically configurable parameters

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	15	
FOLLOW_LPF_PITCH	17	1u	0	15	
FOLLOW_LPF_YAW	18	1u	0	15	
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	15	
RC_LPF_PITCH	23	1u	0	15	
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_ANGLE	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	
ACC_LIMITER_YAW	41	2s	0	1275	
PID_GAIN_ROLL	42	1u	0	255	Gain is calculated as 0.1 + PID_GAIN[axis]*0.02
PID_GAIN_PITCH	43	1u	0	255	
PID_GAIN_YAW	44	1u	0	255	
LPF_FREQ_ROLL	45	2u	10	400	Units: Hz
LPF_FREQ_PITCH	46	2u	10	400	
LPF_FREQ_YAW	47	2u	10	400	
TIMELAPSE_TIME	48	2u	1	3600	Units: sec