API Protocol

In this protocol the following abbreviation are used:

- PC : onboard Ubuntu PC running ROS
- STM32 : Embedded processing system running FreeRTOS

All estimates are provided in SI units as follows:

- Time: [seconds]
- Position: [meters]
- Velocity: [meters/second]
- Angle: [radians]
- Angular velocity: [radians/second]

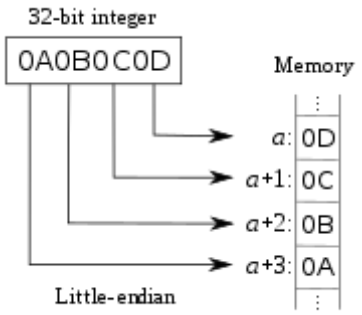
Package structure

The messages (payloads) are sent using the Lightweight Serial Package Communication interface which uses COBS (consistent overhead byte stuffing) for packing.

Data order

Data which consists of multiple bytes are sent in **little-endian format**. This is the same format as is used internally in the STM32 ARM Cortex-M microprocessor and also used by default by the ARM CMSIS DSP library.

In little endian, you store the **least** significant byte in the smallest address. Here's how it would look:



In the API package a 32-bit integer as illustrated above would be sent as:

Byte 1	Byte 2	Byte 3	Byte 4
0D	0C	0B	0A

Data types

Data type	uint8 valueType	Byte length	Description
bool	0x01	1 byte	Boolean: true = 0x01, false = 0x00
uint8	0x03	1 byte	Unsigned integer
uint16	0x04	2 bytes	
uint32	0x05	4 bytes	
int8	N/A	1 byte	Signed integer
int16	N/A	2 bytes	
int32	N/A	4 bytes	
float	0x02	4 bytes	Floating point integer, single precision

PC to embedded board

Test message [reserved]		
Can be used for miscellaneous tests but is generally not used		
Direction	**Message**	**Payload**
PC→STM32	0x01	

Get parameter			
Read a configurable parameter (see Parameters table)			
Direction	**Message**	**Payload**	
PC→STM32	0x02	uint8 type	uint8 param

Set parameter						
Set a configurable parameter (or array of parameters)						
Direction	**Message**	**Payload**				
PC→STM32	0x03	uint8 type	uint8 param	uint8 valueType	uint8 arraySize	uint8 [1-246] raw param bytes

Store parameters		
Write current parameters into EEPROM		
Direction	**Message**	**Payload**
PC→STM32	0x04	

Dump parameters		
Request a raw (byte-)dump of all parameters		
Direction	**Message**	**Payload**
PC→STM32	0x05	

System settings		
Set miscellaneous system settings		
Direction	**Message**	**Payload**
PC→STM32	0x10	*NOT DEFINED YET*

Estimator settings			
Set estimator settings			
Direction	**Message**	**Payload**	
PC→STM32	0x11	uint16 estimate_msg_prescaler	

Controller settings				
Set miscellaneous controller settings				
Direction	**Message**	**Payload**		
PC→STM32	0x12	uint8 mode	uint8 type	

Controller modes	
uint8 mode	**Description**
0x00	Off
0x01	Quaternion reference control (thus "angle" setpoint)
0x02	Angular velocity reference control (angular velocity reference) in body frame) – *quaternion reference will automatically be generated/integrated based on angular velocity reference*
0x03	Velocity control (eg. for joystick control)
0x04	Path following MPC
0xFF	Unknown

Controller type	
uint8 type	**Description**
0x00	Unknown
0x01	LQR controller
0x02	Sliding Mode controller

Yaw correction		
Heading correction input with current heading/yaw angle defined in inertial frame		
Direction	**Message**	**Payload**
PC→STM32	0x20	float yaw

Position correction			
Position correction input with current position defined in inertial (global) frame			
Direction	**Message**	**Payload**	
PC→STM32	0x21	float x	float y

Orientation (quaternion) reference					
Quaternion setpoint for attitude controller in attitude reference control mode					
Direction	**Message**	**Payload**			
PC→STM32	0x30	float q.w	float q.x	float q.y	float q.z

Angular velocity reference					
Angular velocity setpoint for balance controller in angular velocity reference control mode Angular velocity is defined in body frame					
Direction	**Message**	**Payload**			
PC→STM32	0x31	uint8 frame	float omega.x	float omega.y	float omega.z

Reference Frame type	
uint8 frame	**Description**
0x00	Body frame
0x01	Inertial frame
0x02	Heading frame (x-velocity points in the direction of the robots x-axis projected down onto the flat ground plane)

Balance controller reference									
Combination of quaternion and angular velocity setpoint for the balance controller									
Direction	**Message**	**Payload**							
PC→STM32	0x32	uint8 frame	float q.w	float q.x	float q.y	float q.z	float omega.x	float omega.y	float omega.z

Velocity controller reference					
Velocity setpoint for velocity controller when the system is in velocity control mode					
Direction	**Message**	**Payload**			
PC→STM32	0x33	uint8 frame	float vel.x	float vel.y	float vel.z

MPC path reference						
Polynomial path reference for the MPC – in this case using a 9^{th} order polynomial The path polynomial is defined in inertial (global) frame						
Direction	**Message**	**Payload**				
PC→STM32	0x34	float desired_velocity	float desired_heading	float path_length	float coeffs_x[10]	float coeffs_y[10]

Calibrate IMU		
Enter IMU calibration mode Note that the IMU can only be calibrated when the controller is in Off mode		
Direction	**Message**	**Payload**
PC→STM32	0xE0	uint32 magic_key 0x12345678

~~**Request CPU load and task status**~~		
~~Request formatted CPU load and task status string~~		
~~Direction~~	~~**Message**~~	~~**Payload**~~
~~PC→STM32~~	~~0xE1~~	~~uint32 magic_key 0x12345678~~

No longer needed, since CPU load is sent every second automatically after boot

Restart controller		
Restarts Balance controller application/task in embedded firmware		
Direction	**Message**	**Payload**
PC→STM32	0xE2	uint32 magic_key 0x12345678

Enter bootloader		
Used to enter USB bootloader mode to flash/update the embedded firmware Note that bootloader can only be entered when the controller is in Off mode		
Direction	**Message**	**Payload**
PC→STM32	0xF0	uint32 magic_key 0x12345678

Reboot		
Restart the embedded firmware Note that this will perform a hard reset independent of which state the system and controller is in		
Direction	**Message**	**Payload**
PC→STM32	0xF1	uint32 magic_key 0x12345678

Debug messages		
Used for debug text messages up to 250 characters		
Direction	**Message**	**Payload**
PC→STM32	0xFF	uint8 msg[1 - 250]

Embedded board to PC

Test message [reserved]	
Can be used for miscellaneous tests but is generally not used	

Direction	**Message**	**Payload**
STM32→PC	0x01	

Get parameter response	
Response message for reading a configurable parameter	

Direction	**Message**	**Payload**				
STM32→PC	0x02	uint8 type	uint8 param	uint8 valueType	uint8 arraySize	uint8 [1-246] raw param bytes

Set parameter acknowledge	
Acknowledge a change of parameter	

Direction	**Message**	**Payload**		
STM32→PC	0x03	uint8 type	uint8 param	bool acknowledged

Store parameters acknowledge	
Acknowledge of writing the parameters into the EEPROM	

Direction	**Message**	**Payload**
STM32→PC	0x04	bool acknowledged

Dump parameters	
Raw byte-dump of parameters	

Direction	**Message**	**Payload**	
STM32→PC	0x05 (first)	uint16 parameters_size_bytes	uint8 packages_to_follow
STM32→PC	0x05 (following)	uint8 [0-250] raw param bytes	

System info	Sent periodic @ 1 Hz
Miscellaneous system info	

Direction	**Message**	**Payload**		
STM32→PC	0x10	float time	float battery_pct	float current_consumption

State Estimates										Sent periodic @ estimator rate
Latest state estimates – note that velocities are given in inertial frame and position is given in inertial frame based on position where robot was turned on										
Direction	**Message**	**Payload**								
STM32→PC	0x11	float time	float q.w	float q.x	float q.y	float q.z	float dq.w	float dq.x	float dq.y	float dq.z
		float pos.x	float pos.y	float vel.x	float vel.y	float COM.X	float COM.Y	float COM.Z		

Controller info							Sent periodic @ controller rate
General controller info							
Direction	**Message**	**Payload**					
STM32→PC	0x12	float time	uint8 type	uint8 mode	float torque1	float torque2	float torque3	float compute_time
		float delivered_torque1		float delivered_torque2		float delivered_torque3		

Balance Controller info *(NOT IMPLEMENTED YET)*							Sent periodic @ controller rate
Balance controller info (called AttitudeControllerInfo in code)							
Direction	**Message**	**Payload**					
STM32→PC	0x13	float time					

Velocity Controller info *(NOT IMPLEMENTED YET)*							Sent periodic @ controller rate
Velocity controller info							
Direction	**Message**	**Payload**					
STM32→PC	0x14	float time					

Controller debug					Sent periodic @ controller rate
Velocity controller info					
Direction	**Message**	**Payload**			
STM32→PC	0x15	float time	float orient.roll	float orient.pitch	float orient.yaw
		float orient_ref.roll		float orient_ref.pitch	float orient_ref.yaw
		float orient_integral.roll		float orient_integral.pitch	float orient_integral.yaw
		float omega.x		float omega.y	float omega.z
		float omega_ref.x		float omega_ref.y	float omega_ref.z
		float vel.x	float vel.y	float vel_kinematics.x	float vel_kinematics.y
		float vel_ref.x	float vel_ref.y	float torque[3]	float S[3]

MPC info *(NOT IMPLEMENTED YET)*	Sent periodic @ MPC rate
General MPC info	

Direction	**Message**	**Payload**					
STM32→PC	0x20	float time					

Predicted MPC trajectory	Sent periodic @ MPC rate
Trajectory point from the recent MPC trajectory prediction – note that the velocity is given in inertial frame but the position is given in a robocentric inertial frame, hence with the origin at the current robot position Time corresponds to the time of the MPC computation	

Direction	**Message**	**Payload**								
STM32→PC	0x21	float time	uint8 horizon_index	float q.w	float q.x	float q.y	float q.z	float dq.w	float dq.x	float dq.y	float dq.z
		float pos.x	float pos.y	float vel.x	float vel.y						

Raw sensor info – MPU9250 IMU	Sent periodic @ reading rate
Raw sensor values from the IMU and used covariance (in row-major format)	

Direction	**Message**	**Payload**								
STM32→PC	0x30	float time	float acc.x	float acc.y	float acc.z	Float acc.cov[9]	float gyro.x	float gyro.y	float gyro.z	Float gyro.cov[9]
			float mag.x	float mag.y	float mag.z	float mag.cov[9]				

Raw sensor info – MTI200 IMU	Sent periodic @ reading rate
Raw sensor values from the IMU	

Direction	**Message**	**Payload**									
STM32→PC	0x31	float time	float acc.x	float acc.y	float acc.z	float gyro.x	float gyro.y	float gyro.z	float mag.x	float mag.y	float mag.z

Raw sensor info – Encoders	Sent periodic @ reading rate
Raw sensor values from the wheel encoders	

Direction	**Message**	**Payload**			
STM32→PC	0x32	float time	float angle1	float angle2	float angle3

Raw sensor info – Battery								Sent periodic @ reading rate
Raw sensor values from the two batteries								
Direction	**Message**	**Payload**						
STM32→PC	0x33	float time	float vbat1	float vbat2	float current1	float current2	float pct1	float pct2

Calibrate IMU acknowledge		
Acknowledge of initiation of IMU calibration		
Direction	**Message**	**Payload**
STM32→PC	0xE0	bool acknowledged

CPU load and task status response		
Response of the formatted CPU load and task status response text message up to 250 characters		
Direction	**Message**	**Payload**
STM32→PC	0xE1	uint8 msg[1 - 250]

Restart controller acknowledge		
Acknowledge of balance controller restart		
Direction	**Message**	**Payload**
STM32→PC	0xE2	bool acknowledged

Math dump messages		
Used for math logging – will be parsed by PC and dumped into tabulated .txt file in "~/kugle_dump/"		
Direction	**Message**	**Payload**
STM32→PC	0xFA	float variables[1 - 62]

Sensor dump messages		
Used for raw sensor logging – will be parsed by PC and dumped into tabulated .txt file in "~/kugle_dump/"		
Direction	**Message**	**Payload**
STM32→PC	0xFB	float variables[1 - 62]

Covariance dump messages		
Used for covariance logging – will be parsed by PC and dumped into tabulated .txt file in "~/kugle_dump/"		
Direction	**Message**	**Payload**
STM32→PC	0xFC	float variables[1 - 62]

Debug messages		
Used for debug text messages up to 250 characters		
Direction	**Message**	**Payload**
STM32→PC	0xFF	uint8 msg[1 - 250]

Parameters

List/table of configurable parameters

See https://github.com/mindThomas/Kugle-Embedded/blob/master/KugleFirmware/Libraries/Devices/LSPC/MessageTypes.h#L28-L178

See also https://github.com/mindThomas/Kugle-Embedded/blob/master/KugleFirmware/Libraries/Modules/Parameters/Parameters.h

uint8 type	uint8 param	uint8 arraySize	Data type	Parameter	Description
0x01 Debug	0x01	1	bool	EnableDumpMessages	
	0x02	1	bool	EnableRawSensorOutput	
	0x03	1	bool	UseFilteredIMUinRawSensorOutput	
	0x04	1	bool	DisableMotorOutput	
0x02 Behavioral	0x01	1	bool	IndependentHeading	
	0x02	1	bool	YawVelocityBraking	
	0x03	1	bool	StepTestEnabled	
	0x04	1	bool	SineTestEnabled	
	0x05	1	bool	CircleTestEnabled	
	0x06	1	uint8	PowerButtonMode	
0x03 Controller	0x01	1	float	ControllerSampleRate	
	0x02	1	uint8	ControllerType	
	0x03	1	uint8	ControllerMode	
	0x04	1	bool	EnableTorqueLPF	
	0x05	1	float	TorqueLPFtau	
	0x06	1	bool	MotorFailureDetection	
	0x07	1	bool	EnableTorqueSaturation	
	~~0x08~~	~~1~~	~~float~~	~~TorqueMax~~	~~NOT USED~~
	0x09	1	bool	TorqueRampUp	
	0x0A	1	float	TorqueRampUpTime	
	0x0B	1	bool	DisableQdot	
	0x0C	1	bool	DisableQdotInEquivalentControl	
	0x0D	1	bool	DisableOmegaXYInEquivalentControl	
	0x0E	1	bool	AngularVelocityClampsEnabled	
	0x0F	3	float	AngularVelocityClamps	
	0x10	1	uint8	ManifoldType	
	0x11	3	float	K	
	0x12	1	float	Kx	
	0x13	1	float	Ky	
	0x14	1	float	Kz	
	0x15	1	float	Kv_x	
	0x16	1	float	Kv_y	
	0x17	1	float	Kvi_x	
	0x18	1	float	Kvi_y	
	0x19	1	float	gamma	
	0x1A	1	bool	ContinousSwitching	
	0x1B	1	bool	EquivalentControl	
	0x1C	3	float	eta	

	0x1D	3	float	epsilon	
	0x1E	3 x 8	float	LQR_K	Matrix values stored in row-major order
	0x1F	1	float	LQR_MaxYawError	
	0x20	1	float	VelocityControl_AccelerationLimit	
	0x21	1	float	VelocityControl_UseOmegaRef	
	0x22	1	float	VelocityController_MaxTilt	
	0x23	1	float	VelocityController_MaxIntegralCorrection	
	0x24	1	float	VelocityController_VelocityClamp	
	0x25	1	float	VelocityController_IntegralGain	
	0x26	1	float	VelocityController_AngleLPFtau	
	0x27	1	float	VelocityController_OmegaLPFtau	
0x04 Estimator	0x01	1	float	EstimatorSampleRate	
	0x02	1	bool	EnableSensorLPFfilters	
	0x03	1	bool	EnableSoftwareLPFfilters	
	0x04	3	float	SoftwareLPFcoeffs_a	
	0x05	3	float	SoftwareLPFcoeffs_b	
	0x06	1	bool	CreateQdotFromQDifference	
	0x07	1	bool	UseMadgwick	
	0x08	1	bool	EstimateBias	
	0x09	1	bool	SensorDrivenQEKF	
	0x0A	1	bool	UseCoRvelocity	
	0x0B	1	bool	UseVelocityEstimator	
	0x0C	1	bool	EnableVelocityLPF	
	0x0D	1	bool	EnableWheelSlipDetector	
	0x0E	1	bool	UseQdotInVelocityEstimator	
	0x0F	1	bool	EstimateCOM	
	0x10	1	bool	EstimateCOMminVelocity	
	0x11	1	float	MaxCOMDeviation	
	0x12	1	float	MadgwickBeta	
	0x13	1	float	GyroCov_Tuning_Factor	
	0x14	1	float	AccelCov_Tuning_Factor	
	0x15	9	float	cov_gyro_mpu	
	0x16	9	float	cov_acc_mpu	
	0x17	1	float	sigma2_bias	
	0x18	1	float	sigma2_omega	
	0x19	1	float	sigma2_heading	
	0x1A	1	float	GyroscopeTrustFactor	
	0x1B	1	float	eta_encoder	
	0x1C	1	float	eta_accelerometer	
	0x1D	1	float	var_acc_bias	
	0x1E	1	float	var_acceleration	
0x05 Model	0x01	1	float	l	
	0x02	1	float	COM_X	
	0x03	1	float	COM_Y	
	0x04	1	float	COM_Z	
	0x05	1	float	CoR	

	0x06	1	float	g	
	0x07	1	float	rk	
	0x08	1	float	Mk	
	0x09	1	float	Jk	
	0x0A	1	float	rw	
	0x0B	1	float	Mw	
	0x0C	1	float	i_gear	
	0x0D	1	float	Jow	
	0x0E	1	float	Jm	
	0x0F	1	float	Jw	
	0x10	1	float	Mb	
	0x11	1	float	Jbx	
	0x12	1	float	Jby	
	0x13	1	float	Jbz	
	0x14	1	float	Bvk	
	0x15	1	float	Bvm	
	0x16	1	float	Bvb	
	0x17	1	float	EncoderTicksPrRev	
	0x18	1	float	TicksPrRev	
	0x19	1	float	SaturationTorqueOfMaxOutputTorque	
0x06 Test	0x01	1	float	tmp	For test only
	0x02	1	float	tmp2	

Controller type	
uint8 type	**Description**
0x01	LQR controller
0x02	Sliding mode controller

Controller mode	
uint8 mode	**Description**
0x00	Off
0x01	Quaternion control (balance controller reference)
0x02	Velocity control (velocity reference)
0x03	Path following *(NOT IMPLEMENTED YET)*

Power button mode	
uint8 mode	**Description**
0x00	Power off
0x01	Start/stop Quaternion control
0x02	Start/stop Velocity control

Sliding manifold type	
uint8 type	**Description**
0x00	Quaternion derivative manifold with inertial frame quaternion error
0x01	Quaternion derivative manifold with body frame quaternion error
0x02	Angular velocity manifold with inertial frame quaternion error
0x03	Angular velocity manifold with body frame quaternion error *(SUGGESTED)*
0x04	Combined velocity and quaternion derivative manifold *(TEST ONLY)*