# 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]

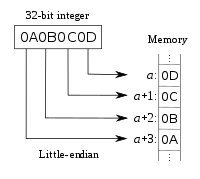
## Package structure

The messages (payloads) are sent using the [Lightweight Serial Package Communication](https://github.com/kdhansen/LSPC) interface which uses COBS (consistent overhead byte stuffing) for packing.

## Data order

Data which consists of multiple bytes are sent in **little-endian format**.

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 type** | **Payload** | |
| PC🡪STM32 | 0x01 |  | |

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Set parameter** | | | | | | |  |
| Set a configurable parameter (or array of parameters) | | | | | | |
| Direction | **Message type** | **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 type** | **Payload** | |
| PC🡪STM32 | 0x04 |  | |

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

|  |  |  |  |
| --- | --- | --- | --- |
| **System settings** | | |  |
| Set miscellaneous system settings | | |
| Direction | **Message type** | **Payload** | |
| PC🡪STM32 | 0x10 | *NOT DEFINED YET* | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Estimator settings** | | | | | |  |
| Set estimator settings | | | | | |
| Direction | **Message type** | **Payload** | | | | |
| PC🡪STM32 | 0x11 | uint16  estimate\_msg\_prescaler | | |  | |
| **Controller settings** | | | | | |  |
| Set miscellaneous controller settings | | | | | |
| Direction | **Message type** | **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 type** | **Payload** | |
| PC🡪STM32 | 0x20 | float  yaw | |

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

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Attitude reference setpoint** | | | | | |  |
| Quaternion setpoint for attitude controller in attitude reference control mode | | | | | |
| Direction | **Message type** | **Payload** | | | | |
| PC🡪STM32 | 0x30 | float  q.w | float  q.x | float  q.y | float  q.z | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Angular velocity reference setpoint [body frame]** | | | | |  |
| Angular velocity setpoint for attitude controller in angular velocity reference control mode  Angular velocity is defined in body frame | | | | |
| Direction | **Message type** | **Payload** | | | |
| PC🡪STM32 | 0x31 | float  omega.x | float  omega.y | float  omega.z | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Angular velocity reference setpoint [inertial frame]** | | | | |  |
| Angular velocity setpoint for attitude controller in angular velocity reference control mode  Angular velocity is defined in inertial frame | | | | |
| Direction | **Message type** | **Payload** | | | |
| PC🡪STM32 | 0x32 | float  omega.x | float  omega.y | float  omega.z | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Velocity reference setpoint [inertial frame]** | | | | |  |
| Velocity setpoint for velocity controller when the system is in velocity control mode  Translational velocity is defined in inertial frame | | | | |
| Direction | **Message type** | **Payload** | | | |
| PC🡪STM32 | 0x33 | float  vel.x | float  vel.y | float  vel.yaw | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Velocity reference setpoint [heading frame]** | | | | |  |
| Velocity setpoint for velocity controller when the system is in velocity control mode  Translational velocity is defined in heading frame, such that x-velocity points in the direction of the robots x-axis projected down onto the flat groundplane | | | | |
| Direction | **Message type** | **Payload** | | | |
| PC🡪STM32 | 0x34 | float  vel.x | float  vel.y | float  vel.yaw | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MPC path reference** | | | | |  |
| Polynomial path reference for the MPC – in this case using a 6th order polynomial  The path polynomial is defined in inertial (global) frame | | | | |
| Direction | **Message type** | **Payload** | | | |
| PC🡪STM32 | 0x35 | float desired\_velocity | float  desired\_heading | float  path\_coeff[7] | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Calibrate IMU** | | |  |
| Enter IMU calibration mode  Note that the IMU can only be calibrated when the controller is in Off mode | | |
| Direction | **Message type** | **Payload** | |
| PC🡪STM32 | 0xE0 | uint32  magic\_key  0x12345678 | |

|  |  |  |  |
| --- | --- | --- | --- |
| **~~Request CPU load and task status~~** | | |  |
| ~~Request formatted CPU load and task status string~~ | | |
| ~~Direction~~ | **~~Message type~~** | **~~Payload~~** | |
| ~~PC🡪STM32~~ | ~~0xE1~~ | ~~uint32~~  ~~magic\_key~~  ~~0x12345678~~ | |

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

|  |  |  |  |
| --- | --- | --- | --- |
| **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 type** | **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 type** | **Payload** | |
| PC🡪STM32 | 0xF1 | uint32  magic\_key  0x12345678 | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Debug messages** | | |  |
| Used for debug text messages up to 250 characters | | |
| Direction | **Message type** | **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 type** | **Payload** | |
| STM32🡪PC | 0x01 |  | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Get parameter response** | | | | | | |  |
| Response message for reading a configurable parameter | | | | | | |
| Direction | **Message type** | **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 type** | **Payload** | | | |
| STM32🡪PC | 0x03 | uint8  type | uint8  param | bool  acknowledged | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Store parameters acknowledge** | | |  |
| Acknowledge of writing the parameters into the EEPROM | | |
| Direction | **Message type** | **Payload** | |
| STM32🡪PC | 0x04 | bool  acknowledged | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Dump parameters** | | | |  |
| Raw byte-dump of parameters | | | |
| Direction | **Message type** | **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 | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Attitude Controller info** | | | | | | | | Sent periodic @ controller rate |
| Attitude controller info | | | | | | | |
| Direction | **Message** | **Payload** | | | | | | |
| STM32🡪PC | 0x13 | float  time |  |  |  |  |  | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Velocity Controller info** | | | | | | | | Sent periodic @ controller rate |
| Velocity controller info | | | | | | | |
| Direction | **Message** | **Payload** | | | | | | |
| STM32🡪PC | 0x14 | float  time |  |  |  |  |  | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **MPC info** | | | | | | | | 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 type** | **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 type** | **Payload** | |
| STM32🡪PC | 0xE1 | uint8 msg[1 - 250] | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Math dump messages** | | |  |
| Used for math debugging – will be parsed by PC and dumped into tabulated .txt file in "~/kugle\_dump/" | | |
| Direction | **Message type** | **Payload** | |
| STM32🡪PC | 0xFF | float variables[1 - 62] | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Debug messages** | | |  |
| Used for debug text messages up to 250 characters | | |
| Direction | **Message type** | **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-L125*](https://github.com/mindThomas/Kugle-Embedded/blob/master/KugleFirmware/Libraries/Devices/LSPC/MessageTypes.h#L28-L125)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **uint8 type** | **uint8**  **param** | **uint8**  **arraySize** | **Data**  **type** | **Description** |
| N/A  General |  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| 0x02  Behavioral | 0x01 | 1 | bool | IndependentHeading |
| 0x02 | 1 | bool | YawVelocityBraking |
| 0x03 | 1 | bool | StepTestEnabled |
| 0x04 | 1 | bool | VelocityControllerEnabled |
| 0x05 | 1 | bool | JoystickVelocityControl |
| 0x04  Estimator |  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| 0x03  Controller |  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| N/A  MPC |  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| 0x05  Model |  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| 0x01  Debug | 0x01 | 1 | bool | EnableLogOutput |
| 0x02 | 1 | bool | EnableRawSensorOutput |
|  |  |  |  |
| 0x06  Test |  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |  |