IRIS Protocol

*Intra-rocket Interface System (IRIS)*

# Preface:

This document outlines the protocol implemented for communications between systems on sounding rockets.

# Protocol Structure:

This protocol consists of two layers. The Data Format Layer and the Communication Layer which encompasses the Physical Layer and is not entirely defined by this protocol. This protocol outlines the payload formats and stipulates that the Communication Layer must implement packet synchronisation and error checking with a cyclic redundancy check.

## Data Format Layer:

This layer consists of the format of the payload. The payload format is defined by data structures presented in the following pages.

## Communication Layer:

This layer includes the Physical Layer and is implementation dependent. For example, an IRIS implementation running on a CAN2.0 bus will embed its 28-bit device identifier into the 29 available bits in the CAN Extended ID. This allows for efficient filtering of packets. Likewise, CAN contains inbuilt error checking with a 15-bit CRC therefore, explicit error checking in the format is not required.

A different implementation may operate over RS232 which does not have synchronisation or error checking. It is therefore required that Serial Line Interface Protocol and a 32-bit CRC are added to the start and end of each packet respectively to remain IRIS compliant. Likewise, the device identifier field must also be placed at the start of each packet.

## Packet structure for IRIS compliant Communication Layer:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Device identifier** | **Packet index** | **Packet identifier** | **Payload data** | **CRC** | **End of frame byte** |
| **Data type** | 28-bit | uint8\_t | uint16\_t | uint8\_t | Up to uint32\_t | uint8\_t |
| **Length (bytes)** | 28-bit | 1 | 2 | Variable – up to 7 bytes | Up to 4 | 1 |
| **Description** | Field that signifies the sender and receiver ID | Signifies index of the incoming packet in multi-frame payloads | Signify the packet type | Fields containing payload data | Cyclic redundancy check. 32-bit if using custom Communication Layer | 0xC0 END byte |

**Fields implemented in CAN protocol.**

**Payload fields.**

## Device identifier encoding:

The device ID field specifies the unique identifier of the sending and receiving devices. This field contains two 14-bit identifiers packet into a 28-bit field. The first 14 fields within the ID specify the device ID of the sending device starting with the least significant bit (LSB) first. The second 14-bits of the field specify the device ID of the receiving device (LSB first). This is the ID that a receiving device can read the know that the packet is intended for them. Likewise, the device can see the sender device ID to know which device it must send its response to.

**28-bit Identifier Structure**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| **Receiver ID** | | | | | | | | | | | | | | **Sender ID** | | | | | | | | | | | | | |

## Multi-frame transmission:

In order to be compatible with CAN2.0, single frames are limited to a size of 8 bytes. This means that for payloads that span more than 5 bytes, they will need to be sent over multiple frames. To prevent the receiver from losing count or missing frames without knowing, a packet index is included as the first byte in the payload byte sequence. As an example, see the following payload fields for a GPS1\_State\_Res packet:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | GPS1 good | GPS 1 latitude | GPS 1 longitude | GPS1 altitude | GPS1 satellites tracked |
| Value | 0 – error, 1 - good | *Decimal degrees* | *Decimal degrees* | *Units of ‘m’* | - |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t | uint8\_t |
| Length (bytes) | 1 | 4 | 4 | 4 | 1 |

This payload contains 14 bytes which is greater than the 5-byte limit per packet. This payload will thus need to be split into three packets:

**Packet 1 payload:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Name | Packet identifier | Packet index | Byte 0 | Byte 1 | Byte 2 | Byte 3 | Byte 4 |
| Value | 0x0009 | 0 | *-* | - | - | - | - |
| Data type | uint16\_t | uint8\_t | uint8\_t | uint8\_t | uint8\_t | uint8\_t | uint8\_t |
| Length (bytes) | 2 | 1 | 1 | 1 | 1 | 1 | 1 |

**Packet 2 payload:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Name | Packet identifier | Packet index | Byte 5 | Byte 6 | Byte 7 | Byte 8 | Byte 9 |
| Value | 0x0009 | 1 | *-* | - | - | - | - |
| Data type | uint16\_t | uint8\_t | uint8\_t | uint8\_t | uint8\_t | uint8\_t | uint8\_t |
| Length (bytes) | 2 | 1 | 1 | 1 | 1 | 1 | 1 |

**Packet 3 payload:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Name | Packet identifier | Packet index | Byte 10 | Byte 11 | Byte 12 | Byte 13 |
| Value | 0x0009 | 2 | *-* | - | - | - |
| Data type | uint16\_t | uint8\_t | uint8\_t | uint8\_t | uint8\_t | uint8\_t |
| Length (bytes) | 2 | 1 | 1 | 1 | 1 | 1 |

To handle an incoming payload spread over multiple packets, the receiver must know based on the identifier that the packet requires multiple transmissions. The receiver must wait to receive all the incoming packets, order them correctly with the packet index and deserialise the data into the relevant data structures.

It is therefore possible to transmit up to 256 sequential packets summing to a maximum payload size of 1280 bytes (excluding packet index).

## Serial Line Interface Protocol (SLIP):

This layer is used to synchronize incoming packets on an unsynchronised serial interface such as RS232. The end of a frame is denoted by the END byte 0xC0 and any occurrence of the END or ESC (0xDB) byte within the packet is escaped by preceding it with the ESC character.

# Contents

[Preface: 1](#_Toc162024782)

[Protocol Structure: 1](#_Toc162024783)

[Data Format Layer: 1](#_Toc162024784)

[Communication Layer: 1](#_Toc162024785)

[Packet structure for IRIS compliant Communication Layer: 1](#_Toc162024786)

[Multi-frame transmission: 2](#_Toc162024787)

[Serial Line Interface Protocol (SLIP): 3](#_Toc162024788)

[Contents 4](#_Toc162024789)

[Payload Fields: 9](#_Toc162024790)

[BAT\_VOL\_REQ 9](#_Toc162024791)

[BAT\_VOL\_RES 9](#_Toc162024792)

[CONTINUITY\_REQ 9](#_Toc162024793)

[CONTINUITY \_RES 9](#_Toc162024794)

[FIRE\_DROGUE\_REQ 10](#_Toc162024795)

[FIRE\_DROGUE\_RES 10](#_Toc162024796)

[FIRE\_MAIN\_REQ 10](#_Toc162024797)

[FIRE\_MAIN\_RES 10](#_Toc162024798)

[GPS1\_STATE\_REQ 11](#_Toc162024799)

[GPS1\_STATE\_RES 11](#_Toc162024800)

[GPS2\_STATE\_REQ 11](#_Toc162024801)

[GPS2\_STATE\_RES 11](#_Toc162024802)

[ACCEL1\_STATE\_REQ 12](#_Toc162024803)

[ACCEL1\_STATE\_RES 12](#_Toc162024804)

[ACCEL2\_STATE\_REQ 12](#_Toc162024805)

[ACCEL2\_STATE\_RES 12](#_Toc162024806)

[GYRO1\_STATE\_REQ 13](#_Toc162024807)

[GYRO1\_STATE\_RES 13](#_Toc162024808)

[GYRO2\_STATE\_REQ 13](#_Toc162024809)

[GYRO2\_STATE\_RES 13](#_Toc162024810)

[MAG1\_STATE\_REQ 14](#_Toc162024811)

[MAG1\_STATE\_RES 14](#_Toc162024812)

[MAG2\_STATE\_REQ 14](#_Toc162024813)

[MAG2\_STATE\_RES 15](#_Toc162024814)

[BARO1\_STATE\_REQ 15](#_Toc162024815)

[BARO1\_STATE\_RES 15](#_Toc162024816)

[BARO2\_STATE\_REQ 16](#_Toc162024817)

[BARO2\_STATE\_RES 16](#_Toc162024818)

[FLASH\_MEMORY\_STATE\_REQ 16](#_Toc162024819)

[FLASH\_MEMORY\_STATE\_RES 16](#_Toc162024820)

[FLASH\_MEMORY\_CONFIG\_SET 17](#_Toc162024821)

[GPS\_TRACKING\_CONFIG\_REQ 17](#_Toc162024822)

[GPS\_TRACKING\_CONFIG\_RES 17](#_Toc162024823)

[GPS\_TRACKING\_CONFIG\_SET 18](#_Toc162024824)

[GPS\_TRACKING\_PACKET 18](#_Toc162024825)

[STREAM\_PACKET\_TYPE\_0 19](#_Toc162024826)

[STREAM\_PACKET\_TYPE\_1 20](#_Toc162024827)

[STREAM\_PACKET\_TYPE\_2 20](#_Toc162024828)

[STREAM\_PACKET\_TYPE\_3 21](#_Toc162024829)

[STREAM\_PACKET\_TYPE\_4 21](#_Toc162024830)

[STREAM\_PACKET\_TYPE\_5 21](#_Toc162024831)

[STREAM\_PACKET\_TYPE\_6 21](#_Toc162024832)

[STREAM\_PACKET\_TYPE\_7 21](#_Toc162024833)

[STREAM\_PACKET\_CONFIG\_REQ 21](#_Toc162024834)

[STREAM\_PACKET\_CONFIG\_RES 22](#_Toc162024835)

[HEART\_BEAT\_CONFIG\_PACKET\_SET 22](#_Toc162024836)

[HEART\_BEAT\_PACKET 22](#_Toc162024837)

[ARM\_DROGUE\_REQ 23](#_Toc162024838)

[ARM\_MAIN\_REQ 23](#_Toc162024839)

[ARM\_MAIN\_RES 24](#_Toc162024840)

[ARM\_DROGUE\_RES 24](#_Toc162024841)

[SYSTEM\_STATE\_PACKET\_REQ 24](#_Toc162024842)

[SYSTEM\_STATE\_PACKET\_TYPE\_0\_RES 25](#_Toc162024843)

[SYSTEM\_STATE\_PACKET\_TYPE\_1\_RES 28](#_Toc162024844)

[SYSTEM\_STATE\_PACKET\_TYPE\_2\_RES 28](#_Toc162024845)

[SYSTEM\_STATE\_PACKET\_TYPE\_3\_RES 28](#_Toc162024846)

[SYSTEM\_STATE\_PACKET\_TYPE\_4\_RES 28](#_Toc162024847)

[SYSTEM\_STATE\_PACKET\_TYPE\_5\_RES 28](#_Toc162024848)

[SYSTEM\_STATE\_PACKET\_TYPE\_6\_RES 28](#_Toc162024849)

[SYSTEM\_STATE\_PACKET\_TYPE\_7\_RES 28](#_Toc162024850)

[SERVO1\_ACTUATE\_REQ 29](#_Toc162024851)

[SERVO1\_ACTUATE\_RES 29](#_Toc162024852)

[SERVO1\_STATE\_REQ 29](#_Toc162024853)

[SERVO1\_STATE\_RES 29](#_Toc162024854)

[SERVO1\_SET\_STATE 30](#_Toc162024855)

[SERVO1\_POSITION\_REQ 30](#_Toc162024856)

[SERVO1\_POSITION\_RES 30](#_Toc162024857)

[SERVO2\_ACTUATE\_REQ 31](#_Toc162024858)

[SERVO2\_ACTUATE\_RES 31](#_Toc162024859)

[SERVO2\_STATE\_REQ 31](#_Toc162024860)

[SERVO2\_STATE\_RES 31](#_Toc162024861)

[SERVO2\_SET\_STATE 32](#_Toc162024862)

[SERVO2\_POSITION\_REQ 32](#_Toc162024863)

[SERVO2\_POSITION\_RES 32](#_Toc162024864)

[SERVO3\_ACTUATE\_REQ 33](#_Toc162024865)

[SERVO3\_ACTUATE\_RES 33](#_Toc162024866)

[SERVO3\_STATE\_REQ 33](#_Toc162024867)

[SERVO3\_STATE\_RES 33](#_Toc162024868)

[SERVO3\_SET\_STATE 34](#_Toc162024869)

[SERVO3\_POSITION\_REQ 34](#_Toc162024870)

[SERVO3\_POSITION\_RES 34](#_Toc162024871)

[SERVO4\_ACTUATE\_REQ 35](#_Toc162024872)

[SERVO4\_ACTUATE\_RES 35](#_Toc162024873)

[SERVO4\_STATE\_REQ 35](#_Toc162024874)

[SERVO4\_STATE\_RES 36](#_Toc162024875)

[SERVO4\_SET\_STATE 36](#_Toc162024876)

[SERVO4\_POSITION\_REQ 36](#_Toc162024877)

[SERVO4\_POSITION\_RES 36](#_Toc162024878)

[SERVO5\_ACTUATE\_REQ 37](#_Toc162024879)

[SERVO5\_ACTUATE\_RES 37](#_Toc162024880)

[SERVO5\_STATE\_REQ 37](#_Toc162024881)

[SERVO5\_STATE\_RES 38](#_Toc162024882)

[SERVO5\_SET\_STATE 38](#_Toc162024883)

[SERVO5\_POSITION\_REQ 38](#_Toc162024884)

[SERVO5\_POSITION\_RES 38](#_Toc162024885)

[SERVO6\_ACTUATE\_REQ 39](#_Toc162024886)

[SERVO6\_ACTUATE\_RES 39](#_Toc162024887)

[SERVO6\_STATE\_REQ 39](#_Toc162024888)

[SERVO6\_STATE\_RES 40](#_Toc162024889)

[SERVO6\_SET\_STATE 40](#_Toc162024890)

[SERVO6\_POSITION\_REQ 40](#_Toc162024891)

[SERVO6\_POSITION\_RES 40](#_Toc162024892)

[DC\_MOTOR1\_PWM\_SET 41](#_Toc162024893)

[DC\_MOTOR1\_VELOCITY\_SET 41](#_Toc162024894)

[DC\_MOTOR1\_POSITION\_SET 41](#_Toc162024895)

[DC\_MOTOR1\_STATE\_SET 42](#_Toc162024896)

[DC\_MOTOR1\_STATE\_REQ 42](#_Toc162024897)

[DC\_MOTOR1\_STATE\_RES 42](#_Toc162024898)

[DC\_MOTOR2\_PWM\_SET 43](#_Toc162024899)

[DC\_MOTOR2\_VELOCITY\_SET 43](#_Toc162024900)

[DC\_MOTOR2\_POSITION\_SET 43](#_Toc162024901)

[DC\_MOTOR2\_STATE\_SET 44](#_Toc162024902)

[DC\_MOTOR2\_STATE\_REQ 44](#_Toc162024903)

[DC\_MOTOR2\_STATE\_RES 44](#_Toc162024904)

[DC\_MOTOR3\_PWM\_SET 44](#_Toc162024905)

[DC\_MOTOR3\_VELOCITY\_SET 45](#_Toc162024906)

[DC\_MOTOR3\_POSITION\_SET 45](#_Toc162024907)

[DC\_MOTOR3\_STATE\_SET 45](#_Toc162024908)

[DC\_MOTOR3\_STATE\_REQ 46](#_Toc162024909)

[DC\_MOTOR3\_STATE\_RES 46](#_Toc162024910)

[DC\_MOTOR4\_PWM\_SET 46](#_Toc162024911)

[DC\_MOTOR4\_VELOCITY\_SET 46](#_Toc162024912)

[DC\_MOTOR4\_POSITION\_SET 47](#_Toc162024913)

[DC\_MOTOR4\_STATE\_SET 47](#_Toc162024914)

[DC\_MOTOR4\_STATE\_REQ 47](#_Toc162024915)

[DC\_MOTOR4\_STATE\_RES 48](#_Toc162024916)

[PRESSURE\_SENSOR1\_READ\_REQ 48](#_Toc162024917)

[PRESSURE\_SENSOR1\_READ\_RES 48](#_Toc162024918)

[PRESSURE\_SENSOR2\_READ\_REQ 48](#_Toc162024919)

[PRESSURE\_SENSOR2\_READ\_RES 48](#_Toc162024920)

[PRESSURE\_SENSOR3\_READ\_REQ 49](#_Toc162024921)

[PRESSURE\_SENSOR3\_READ\_RES 49](#_Toc162024922)

[PRESSURE\_SENSOR4\_READ\_REQ 49](#_Toc162024923)

[PRESSURE\_SENSOR4\_READ\_RES 49](#_Toc162024924)

[TEMPERATURE\_SENSOR1\_READ\_REQ 50](#_Toc162024925)

[TEMPERATURE\_SENSOR1\_READ\_RES 50](#_Toc162024926)

[TEMPERATURE\_SENSOR2\_READ\_REQ 50](#_Toc162024927)

[TEMPERATURE\_SENSOR2\_READ\_RES 50](#_Toc162024928)

[TEMPERATURE\_SENSOR3\_READ\_REQ 50](#_Toc162024929)

[TEMPERATURE\_SENSOR3\_READ\_RES 51](#_Toc162024930)

[TEMPERATURE\_SENSOR4\_READ\_REQ 51](#_Toc162024931)

[TEMPERATURE\_SENSOR4\_READ\_RES 51](#_Toc162024932)

# Payload Fields:

## BAT\_VOL\_REQ

Request battery voltage.

Identifier: 0x0000

Payload:

*No payload fields.*

## BAT\_VOL\_RES

Battery voltage response.

Identifier: 0x0001

Payload:

|  |  |
| --- | --- |
| Name | Battery voltage |
| Value | - |
| Data type | float32\_t |
| Length (bytes) | 4 |

## CONTINUITY\_REQ

Request continuity.

Identifier: 0x0002

Payload:

*No payload fields.*

## CONTINUITY \_RES

Continuity response.

Identifier: 0x0003

Payload:

|  |  |  |
| --- | --- | --- |
| Name | Drogue e-match state | Main e-match state |
| Value | *ematchState* | *ematchState* |
| Data type | uint8\_t | uint8\_t |
| Length (bytes) | 1 | 1 |

*ematchState*

|  |  |
| --- | --- |
| Value | Result |
| 0 | *OPEN\_CIRCUIT* |
| 1 | *SHORT\_CIRCUIT* |
| 2 | *GOOD* |
| 3 | *EMATCH\_ERROR* |

## FIRE\_DROGUE\_REQ

Fire drogue channel request.

Identifier: 0x0004

Payload:

*No payload fields.*

## FIRE\_DROGUE\_RES

Fire drogue channel response.

Identifier: 0x0005

Payload:

|  |  |
| --- | --- |
| Name | Fire drogue result |
| Value | 0 – success, 1 - error |
| Data type | uint8\_t |
| Length (bytes) | 1 |

## FIRE\_MAIN\_REQ

Fire main channel request.

Identifier: 0x0006

Payload:

*No payload fields.*

## FIRE\_MAIN\_RES

Fire main channel response.

Identifier: 0x0007

Payload:

|  |  |
| --- | --- |
| Name | Fire main result |
| Value | 0 – success, 1 - error |
| Data type | uint8\_t |
| Length (bytes) | 1 |

## GPS1\_STATE\_REQ

Request GPS 1 state.

Identifier: 0x0008

Payload:

*No payload fields.*

## GPS1\_STATE\_RES

GPS 1 state response.

Identifier: 0x0009

Payload:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | GPS1 good | GPS 1 latitude | GPS 1 longitude | GPS1 altitude | GPS1 satellites tracked |
| Value | 0 – error, 1 - good | *Decimal degrees* | *Decimal degrees* | *Units of ‘m’* | - |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t | uint8\_t |
| Length (bytes) | 1 | 4 | 4 | 4 | 1 |

## GPS2\_STATE\_REQ

Request GPS 2 state.

Identifier: 0x000A

Payload:

*No payload fields.*

## GPS2\_STATE\_RES

GPS 2 state response.

Identifier: 0x000B

Payload:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | GPS2 good | GPS2 latitude | GPS2 longitude | GPS2 altitude | GPS2 satellites tracked |
| Value | 0 – error, 1 - good | *Decimal degrees* | *Decimal degrees* | *Units of ‘m’* | - |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t | uint8\_t |
| Length (bytes) | 1 | 4 | 4 | 4 | 1 |

## ACCEL1\_STATE\_REQ

Request accelerometer 1 state.

Identifier: 0x000C

Payload:

*No payload fields.*

## ACCEL1\_STATE\_RES

Accelerometer 1 state response.

Identifier: 0x000D

Payload:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | acc1 good | acc1X | acc1Y | acc1Z |
| Value | 0 – error, 1 - good | *Units of ‘g’* | *Units of ‘g’* | U*nits of ‘g’* |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 | 4 |

## ACCEL2\_STATE\_REQ

Request accelerometer 2 state.

Identifier: 0x000E

Payload:

*No payload fields.*

## ACCEL2\_STATE\_RES

Accelerometer 2 state response.

Identifier: 0x000F

Payload:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | acc2 good | acc2X | acc2Y | acc2Z |
| Value | 0 – error, 1 - good | *Units of ‘g’* | *Units of ‘g’* | U*nits of ‘g’* |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 | 4 |

## GYRO1\_STATE\_REQ

Request gyroscope 1 state.

Identifier: 0x0010

Payload:

*No payload fields.*

## GYRO1\_STATE\_RES

Gyroscope 1 state response.

Identifier: 0x0011

Payload:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | gyro1 good | gyro1X | gyro1Y | gyro1Z |
| Value | 0 – error, 1 - good | *Units of ‘deg/s’* | *Units of ‘deg/s’* | *Units of ‘deg/s’* |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 | 4 |

## GYRO2\_STATE\_REQ

Request gyroscope 2 state.

Identifier: 0x0012

Payload:

*No payload fields.*

## GYRO2\_STATE\_RES

Gyroscope 2 state response.

Identifier: 0x0013

Payload:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | gyro2 good | gyro2X | gyro2Y | gyro2Z |
| Value | 0 – error, 1 - good | *Units of ‘deg/s’* | *Units of ‘deg/s’* | *Units of ‘deg/s’* |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 | 4 |

## MAG1\_STATE\_REQ

Request magnetometer 1 state.

Identifier: 0x0014

Payload:

*No payload fields.*

## MAG1\_STATE\_RES

Magnetometer 1 state response.

Identifier: 0x0015

Payload:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | mag1 good | mag1X | mag1Y | mag1Z |
| Value | 0 – error, 1 - good | *Units of ‘uT’* | *Units of ‘uT’* | *Units of ‘uT’* |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 | 4 |

## MAG2\_STATE\_REQ

Request magnetometer 2 state.

Identifier: 0x0016

Payload:

*No payload fields.*

## MAG2\_STATE\_RES

Magnetometer 1 state response.

Identifier: 0x0017

Payload:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | mag2 good | mag2X | mag2Y | mag2Z |
| Value | 0 – error, 1 - good | *Units of ‘uT’* | *Units of ‘uT’* | *Units of ‘uT’* |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 | 4 |

## BARO1\_STATE\_REQ

Request barometer 1 state.

Identifier: 0x0018

Payload:

*No payload fields.*

## BARO1\_STATE\_RES

Barometer 1 state response.

Identifier: 0x0019

Payload:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | baro1 good | Baro1 pressure | Baro1 temperature | Baro1 altitude |
| Value | 0 – error, 1 - good | *Units of ‘Pa’* | *Units of ‘deg C’* | *Units of ‘m’* |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 | 4 |

## BARO2\_STATE\_REQ

Request barometer 2 state.

Identifier: 0x001A

Payload:

*No payload fields.*

## BARO2\_STATE\_RES

Barometer 2 state response.

Identifier: 0x001B

Payload:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | baro2 good | Baro2 pressure | Baro2 temperature | Baro2 altitude |
| Value | 0 – error, 1 - good | *Units of ‘Pa’* | *Units of ‘deg C’* | *Units of ‘m’* |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 | 4 |

## FLASH\_MEMORY\_STATE\_REQ

Request flash memory state.

Identifier: 0x001C

Payload:

*No payload fields.*

## FLASH\_MEMORY\_STATE\_RES

Flash memory state response.

Identifier: 0x001D

Payload:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | flash good | Flash write speed | Available flash memory |
| Value | 0 – error, 1 - good | *Units of ‘Hz’* | Units of Kbytes |
| Data type | uint8\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 |

## FLASH\_MEMORY\_CONFIG\_SET

Flash memory state response.

Identifier: 0x001E

Payload:

|  |  |  |
| --- | --- | --- |
| Name | flash logging enabled | Flash write speed |
| Value | 0 – diabled, 1 - enabled | *Units of ‘Hz’* |
| Data type | uint8\_t | float32\_t |
| Length (bytes) | 1 | 4 |

## GPS\_TRACKING\_CONFIG\_REQ

Request GPS tracking configuration.

Identifier: 0x001F

Payload:

*No payload fields.*

## GPS\_TRACKING\_CONFIG\_RES

GPS tracking configuration response.

Identifier: 0x0020

Payload:

|  |  |  |
| --- | --- | --- |
| Name | GPS tracking enabled | GPS tracking chirp frequency |
| Value | 0 – disabled, 1 - enabled | *Units of ‘Hz’* |
| Data type | uint8\_t | float32\_t |
| Length (bytes) | 1 | 4 |

## GPS\_TRACKING\_CONFIG\_SET

Set GPS tracking configuration.

Identifier: 0x0021

Payload:

|  |  |  |
| --- | --- | --- |
| Name | GPS tracking enabled | GPS tracking chirp frequency |
| Value | *0 – disabled, 1- enabled* | *Units of ‘Hz’* |
| Data type | uint8\_t | float32\_t |
| Length (bytes) | 1 | 4 |

## GPS\_TRACKING\_PACKET

GPS tracking data packet.

Identifier: 0x0022

Payload:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | GPS1 latitude | GPS1 longitude | GPS1 altitude | GPS1 satellites tracked |
| Value | *Decimal degrees* | *Decimal degrees* | *Units of ‘m’* | - |
| Data type | float32\_t | float32\_t | float32\_t | uint8\_t |
| Length (bytes) | 4 | 4 | 4 | 1 |

STREAM\_PACKET\_CONFIG\_SET

Configuration of data streaming

Identifier: 0x0023

Payload:

|  |  |  |
| --- | --- | --- |
| Name | Packet type enabled | Packet stream frequency |
| Value | *Value from 0-7* | *Units of Hz* |
| Data type | uint8\_t | float32\_t |
| Length (bytes) | 1 | 4 |

## STREAM\_PACKET\_TYPE\_0

Data streaming packet.

Identifier: 0x0024

Payload:

*<Start of payload>*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Timestamp | GPS1 latitude | GPS1 longitude | GPS1 altitude | GPS1 satellites tracked |
| Value | *Milliseconds since boot* | *Decimal degrees* | *Decimal degrees* | *Units of ‘m’* | - |
| Data type | uint32\_t | float32\_t | float32\_t | float32\_t | uint8\_t |
| Length (bytes) | 4 | 4 | 4 | 4 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Acc1X | Acc1Y | Acc1Z | VelX | VelY | VelZ |
| *Units of ‘g’* | *Units of ‘g’* | *Units of ‘g’* | *Units of m/s* | *Units of m/s* | *Units of m/s* |
| float32\_t | float32\_t | float32\_t | float32\_t | float32\_t | float32\_t |
| 4 | 4 | 4 | 4 | 4 | 4 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Baro1 altitude | gyro1X | gyro1Y | gyro1Z | Quaternion 1 | Quaternion 2 |
| *Units of m* | *Units of rad/s* | *Units of rad/s* | *Units of rad/s* | *q1* | *q2* |
| float32\_t | float32\_t | float32\_t | float32\_t | float32\_t | float32\_t |
| 4 | 4 | 4 | 4 | 4 | 4 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Quaternion 3 | Quaternion 4 | Battery voltage | Flight State | Ambient temperature | Available flash memory |
| *q3* | *q4* | *Units of V* | *Enum defined below* | Units of deg C | Units of Kbytes |
| float32\_t | float32\_t | float32\_t | uint8\_t | float32\_t | float32\_t |
| 4 | 4 | 4 | 1 | 4 | 4 |

*<End of payload>*

***Flight State:***

|  |  |
| --- | --- |
| State description | Value |
| Idle on pad | 0 |
| Launched | 1 |
| Burnout | 2 |
| Apogee | 3 |
| Main chute deploy | 4 |
| Landed | 5 |

## STREAM\_PACKET\_TYPE\_1

Data streaming packet.

Identifier: 0x0025

Payload:

## STREAM\_PACKET\_TYPE\_2

Data streaming packet.

Identifier: 0x0026

Payload:

## STREAM\_PACKET\_TYPE\_3

Data streaming packet.

Identifier: 0x0027

Payload:

## STREAM\_PACKET\_TYPE\_4

Data streaming packet.

Identifier: 0x0028

Payload:

## STREAM\_PACKET\_TYPE\_5

Data streaming packet.

Identifier: 0x0029

Payload:

## STREAM\_PACKET\_TYPE\_6

Data streaming packet.

Identifier: 0x002A

Payload:

## STREAM\_PACKET\_TYPE\_7

Data streaming packet.

Identifier: 0x002B

Payload:

## STREAM\_PACKET\_CONFIG\_REQ

Request configuration of data streaming

Identifier: 0x002C

Payload:

*No payload*

## STREAM\_PACKET\_CONFIG\_RES

Request configuration of data streaming

Identifier: 0x002D

Payload:

*<Start of payload>*

|  |  |  |
| --- | --- | --- |
| Name | Packet type enabled | Packet stream frequency |
| Value | *Value from 0-7*  *Else – No stream packets enabled* | *Units of Hz* |
| Data type | uint8\_t | float32\_t |
| Length (bytes) | 1 | 4 |

## HEART\_BEAT\_CONFIG\_PACKET\_SET

Heart beat configuration packet.

Identifier: 0x002E

Payload:

|  |  |  |
| --- | --- | --- |
| Name | Heart beat enabled | Heart beat chirp frequency |
| Value | *0 – disabled, 1- enabled* | *Units of ‘Hz’* |
| Data type | uint8\_t | float32\_t |
| Length (bytes) | 1 | 4 |

## HEART\_BEAT\_PACKET

Heart beat packet.

Identifier: 0x002F

Payload:

|  |  |
| --- | --- |
| Name |  |
| Value |  |
| Data type |  |
| Length (bytes) |  |

## ARM\_DROGUE\_REQ

Request to arm system to fire main e-match.

Any value provided other than 0 or 1 is interpreted as a request for the current drogue arming state.

Identifier: 0x0030

Payload:

|  |  |
| --- | --- |
| Name | Drogue arm state set |
| Value | *0 – Disarmed,*  *1 – Armed,*  *Else – Arm state request* |
| Data type | uint8\_t |
| Length (bytes) | 1 |

## ARM\_MAIN\_REQ

Request to arm system to fire drogue e-match.

Any value provided other than 0 or 1 is interpreted as a request for the current main arming state.

Identifier: 0x0031

Payload:

|  |  |
| --- | --- |
| Name | Main arm state set |
| Value | *0 – Disarmed,*  *1 – Armed,*  *Else – Arm state request* |
| Data type | uint8\_t |
| Length (bytes) | 1 |

## ARM\_MAIN\_RES

Response to request to arm main e-match.

Identifier: 0x0032

Payload:

|  |  |
| --- | --- |
| Name | Arm main state |
| Value | *1 – armed, 0 - disarmed* |
| Data type | uint8\_t |
| Length (bytes) | 1 |

## ARM\_DROGUE\_RES

Response to request to arm drogue e-match.

Identifier: 0x0033

Payload:

|  |  |
| --- | --- |
| Name | Arm drogue state |
| Value | *1 – armed, 0 - disarmed* |
| Data type | uint8\_t |
| Length (bytes) | 1 |

## SYSTEM\_STATE\_PACKET\_REQ

Requests system state packet.

Identifier: 0x0034

Payload:

|  |  |
| --- | --- |
| Name | State packet type |
| Value | *Values 0 – 7* |
| Data type | uint8\_t |
| Length (bytes) | 1 |

## SYSTEM\_STATE\_PACKET\_TYPE\_0\_RES

Requests system state packet.

Identifier: 0x0035

*Payload:*

*<Start of payload>*

|  |  |  |
| --- | --- | --- |
| Name | Timestamp | Battery voltage |
| Value | *Milliseconds since boot* |  |
| Data type | uint32\_t | float32\_t |
| Length (bytes) | 4 | 4 |

|  |  |  |  |
| --- | --- | --- | --- |
| Drogue e-match state | Main e-match state | Arm drogue state | Arm main state |
| *Flight State* | *Flight State* | *1 – armed, 0 - disarmed* | *1 – armed, 0 - disarmed* |
| uint8\_t | uint8\_t | uint8\_t | uint8\_t |
| 1 | 1 | 1 | 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| GPS1 good | GPS1 latitude | GPS1 longitude | GPS1 altitude | GPS1 satellites tracked |
| 0 – error, 1 - good | *Decimal degrees* | *Decimal degrees* | *Units of ‘m’* | - |
| uint8\_t | float32\_t | float32\_t | float32\_t | uint8\_t |
| 1 | 4 | 4 | 4 | 1 |

|  |  |  |  |
| --- | --- | --- | --- |
| acc1 good | acc1X | acc1Y | acc1Z |
| 0 – error, 1 - good | *Units of ‘g’* | *Units of ‘g’* | U*nits of ‘g’* |
| uint8\_t | float32\_t | float32\_t | float32\_t |
| 1 | 4 | 4 | 4 |

|  |  |  |  |
| --- | --- | --- | --- |
| acc2 good | acc2X | acc2Y | acc2Z |
| 0 – error, 1 - good | *Units of ‘g’* | *Units of ‘g’* | U*nits of ‘g’* |
| uint8\_t | float32\_t | float32\_t | float32\_t |
| 1 | 4 | 4 | 4 |

|  |  |  |  |
| --- | --- | --- | --- |
| gyro1 good | gyro1X | gyro1Y | gyro1Z |
| 0 – error, 1 - good | *Units of ‘deg/s’* | *Units of ‘deg/s’* | *Units of ‘deg/s’* |
| uint8\_t | float32\_t | float32\_t | float32\_t |
| 1 | 4 | 4 | 4 |

|  |  |  |  |
| --- | --- | --- | --- |
| gyro2 good | gyro2X | gyro2Y | gyro2Z |
| 0 – error, 1 - good | *Units of ‘deg/s’* | *Units of ‘deg/s’* | *Units of ‘deg/s’* |
| uint8\_t | float32\_t | float32\_t | float32\_t |
| 1 | 4 | 4 | 4 |

|  |  |  |  |
| --- | --- | --- | --- |
| mag1 good | mag1X | mag1Y | mag1Z |
| 0 – error, 1 - good | *Units of ‘uT’* | *Units of ‘uT’* | *Units of ‘uT’* |
| uint8\_t | float32\_t | float32\_t | float32\_t |
| 1 | 4 | 4 | 4 |

|  |  |  |  |
| --- | --- | --- | --- |
| baro1 good | Baro1 pressure | Baro1 temperature | Baro1 altitude |
| 0 – error, 1 - good | *Units of ‘Pa’* | *Units of ‘deg C’* | *Units of ‘m’* |
| uint8\_t | float32\_t | float32\_t | float32\_t |
| 1 | 4 | 4 | 4 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| flash good | Flash write speed | Available flash memory | GPS tracking enabled | GPS tracking chirp frequency |
| 0 – error, 1 - good | *Units of ‘Hz’* | Units of Kbytes | 0 – disabled, 1 - enabled | *Units of ‘Hz’* |
| uint8\_t | float32\_t | float32\_t | uint8\_t | float32\_t |
| 1 | 4 | 4 | 1 | 4 |

|  |  |  |  |
| --- | --- | --- | --- |
| Stream packet type enabled | Packet stream frequency | Heart beat enabled | Heart beat chirp frequency |
| *Value from 0-7* | *Units of Hz* | *0 – disabled, 1- enabled* | *Units of ‘Hz’* |
| uint8\_t | float32\_t | uint8\_t | float32\_t |

## SYSTEM\_STATE\_PACKET\_TYPE\_1\_RES

Requests system state packet.

*Identifier: 0x0036*

## SYSTEM\_STATE\_PACKET\_TYPE\_2\_RES

Requests system state packet.

*Identifier: 0x0037*

## SYSTEM\_STATE\_PACKET\_TYPE\_3\_RES

Requests system state packet.

*Identifier: 0x0038*

## SYSTEM\_STATE\_PACKET\_TYPE\_4\_RES

Requests system state packet.

*Identifier: 0x0039*

## SYSTEM\_STATE\_PACKET\_TYPE\_5\_RES

Requests system state packet.

*Identifier: 0x003A*

## SYSTEM\_STATE\_PACKET\_TYPE\_6\_RES

Requests system state packet.

*Identifier: 0x003B*

## SYSTEM\_STATE\_PACKET\_TYPE\_7\_RES

Requests system state packet.

*Identifier: 0x003C*

## SERVO1\_ACTUATE\_REQ

Requests change of servo 1 position.

*Identifier: 0x003D*

|  |  |
| --- | --- |
| Name | Servo position |
| Value | *0 – min range, 65535 – max range* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## SERVO1\_ACTUATE\_RES

Response to servo 1 change position request.

*Identifier: 0x003E*

|  |  |
| --- | --- |
| Name | State packet type |
| Value | *0 – success, 1 - error* |
| Data type | uint8\_t |
| Length (bytes) | 1 |

## SERVO1\_STATE\_REQ

Request servo 1 state.

*Identifier: 0x003F*

## SERVO1\_STATE\_RES

Request servo 1 state.

*Identifier: 0x0040*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Servo enabled | Minimum angle limit | Maximum angle limit | Current draw |
| Value | *0 – Servo disabled*  *1 – servo enabled* | *Degrees* | *Degrees* | *Current in milliamps* |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 | 4 |

## SERVO1\_SET\_STATE

Get servo 1 range.

*Identifier: 0x0041*

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Servo enabled | Minimum angle limit | Maximum angle limit |
| Value | *0 – Servo disabled*  *1 – servo enabled* | *Degrees* | *Degrees* |
| Data type | uint8\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 |

## SERVO1\_POSITION\_REQ

Request servo 1 angle.

*Identifier: 0x0042*

## SERVO1\_POSITION\_RES

Request servo 1 position.

*Identifier: 0x0043*

|  |  |
| --- | --- |
| Name | Servo position |
| Value | *0 – min range, 65535 – max range* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## SERVO2\_ACTUATE\_REQ

Requests change of servo 2 position.

*Identifier: 0x0044*

|  |  |
| --- | --- |
| Name | Servo position |
| Value | *0 – min range, 65535 – max range* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## SERVO2\_ACTUATE\_RES

Response to servo 2 change position request.

*Identifier: 0x0045*

|  |  |
| --- | --- |
| Name | State packet type |
| Value | *0 – success, 1 - error* |
| Data type | uint8\_t |
| Length (bytes) | 1 |

## SERVO2\_STATE\_REQ

Request servo 2 state.

*Identifier: 0x0046*

## SERVO2\_STATE\_RES

Request servo 2 state.

*Identifier: 0x0047*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Servo enabled | Minimum angle limit | Maximum angle limit | Current draw |
| Value | *0 – Servo disabled*  *1 – servo enabled* | *Degrees* | *Degrees* | *Current in milliamps* |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 | 4 |

## SERVO2\_SET\_STATE

Get servo 2 range.

*Identifier: 0x0048*

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Servo enabled | Minimum angle limit | Maximum angle limit |
| Value | *0 – Servo disabled*  *1 – servo enabled* | *Degrees* | *Degrees* |
| Data type | uint8\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 |

## SERVO2\_POSITION\_REQ

Request servo 2 angle.

*Identifier: 0x0049*

## SERVO2\_POSITION\_RES

Request servo 2 position.

*Identifier: 0x004A*

|  |  |
| --- | --- |
| Name | Servo position |
| Value | *0 – min range, 65535 – max range* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## SERVO3\_ACTUATE\_REQ

Requests change of servo 3 position.

*Identifier: 0x004B*

|  |  |
| --- | --- |
| Name | Servo position |
| Value | *0 – min range, 65535 – max range* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## SERVO3\_ACTUATE\_RES

Response to servo 3 change position request.

*Identifier: 0x004C*

|  |  |
| --- | --- |
| Name | State packet type |
| Value | *0 – success, 1 - error* |
| Data type | uint8\_t |
| Length (bytes) | 1 |

## SERVO3\_STATE\_REQ

Request servo 3 state.

*Identifier: 0x004D*

## SERVO3\_STATE\_RES

Request servo 3 state.

*Identifier: 0x004E*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Servo enabled | Minimum angle limit | Maximum angle limit | Current draw |
| Value | *0 – Servo disabled*  *1 – servo enabled* | *Degrees* | *Degrees* | *Current in milliamps* |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 | 4 |

## SERVO3\_SET\_STATE

Get servo 3 range.

*Identifier: 0x004F*

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Servo enabled | Minimum angle limit | Maximum angle limit |
| Value | *0 – Servo disabled*  *1 – servo enabled* | *Degrees* | *Degrees* |
| Data type | uint8\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 |

## SERVO3\_POSITION\_REQ

Request servo 3 angle.

*Identifier: 0x0050*

## SERVO3\_POSITION\_RES

Request servo 3 position.

*Identifier: 0x0051*

|  |  |
| --- | --- |
| Name | Servo position |
| Value | *0 – min range, 65535 – max range* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## SERVO4\_ACTUATE\_REQ

Requests change of servo 4 position.

*Identifier: 0x0052*

|  |  |
| --- | --- |
| Name | Servo position |
| Value | *0 – min range, 65535 – max range* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## SERVO4\_ACTUATE\_RES

Response to servo 4 change position request.

*Identifier: 0x0053*

|  |  |
| --- | --- |
| Name | State packet type |
| Value | *0 – success, 1 - error* |
| Data type | uint8\_t |
| Length (bytes) | 1 |

## SERVO4\_STATE\_REQ

Request servo 4 state.

*Identifier: 0x0054*

## SERVO4\_STATE\_RES

Request servo 4 state.

*Identifier: 0x0055*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Servo enabled | Minimum angle limit | Maximum angle limit | Current draw |
| Value | *0 – Servo disabled*  *1 – servo enabled* | *Degrees* | *Degrees* | *Current in milliamps* |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 | 4 |

## SERVO4\_SET\_STATE

Get servo 4 range.

*Identifier: 0x0056*

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Servo enabled | Minimum angle limit | Maximum angle limit |
| Value | *0 – Servo disabled*  *1 – servo enabled* | *Degrees* | *Degrees* |
| Data type | uint8\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 |

## SERVO4\_POSITION\_REQ

Request servo 4 angle.

*Identifier: 0x0057*

## SERVO4\_POSITION\_RES

Request servo 4 position.

*Identifier: 0x0058*

|  |  |
| --- | --- |
| Name | Servo position |
| Value | *0 – min range, 65535 – max range* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## SERVO5\_ACTUATE\_REQ

Requests change of servo 5 position.

*Identifier: 0x0059*

|  |  |
| --- | --- |
| Name | Servo position |
| Value | *0 – min range, 65535 – max range* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## SERVO5\_ACTUATE\_RES

Response to servo 5 change position request.

*Identifier: 0x005A*

|  |  |
| --- | --- |
| Name | State packet type |
| Value | *0 – success, 1 - error* |
| Data type | uint8\_t |
| Length (bytes) | 1 |

## SERVO5\_STATE\_REQ

Request servo 5 state.

*Identifier: 0x005B*

## SERVO5\_STATE\_RES

Request servo 5 state.

*Identifier: 0x005C*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Servo enabled | Minimum angle limit | Maximum angle limit | Current draw |
| Value | *0 – Servo disabled*  *1 – servo enabled* | *Degrees* | *Degrees* | *Current in milliamps* |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 | 4 |

## SERVO5\_SET\_STATE

Get servo 5 range.

*Identifier: 0x005D*

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Servo enabled | Minimum angle limit | Maximum angle limit |
| Value | *0 – Servo disabled*  *1 – servo enabled* | *Degrees* | *Degrees* |
| Data type | uint8\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 |

## SERVO5\_POSITION\_REQ

Request servo 5 angle.

*Identifier: 0x005E*

## SERVO5\_POSITION\_RES

Request servo 5 position.

*Identifier: 0x005F*

|  |  |
| --- | --- |
| Name | Servo position |
| Value | *0 – min range, 65535 – max range* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## SERVO6\_ACTUATE\_REQ

Requests change of servo 6 position.

*Identifier: 0x0060*

|  |  |
| --- | --- |
| Name | Servo position |
| Value | *0 – min range, 65535 – max range* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## SERVO6\_ACTUATE\_RES

Response to servo 6 change position request.

*Identifier: 0x0061*

|  |  |
| --- | --- |
| Name | State packet type |
| Value | *0 – success, 1 - error* |
| Data type | uint8\_t |
| Length (bytes) | 1 |

## SERVO6\_STATE\_REQ

Request servo 6 state.

*Identifier: 0x0062*

## SERVO6\_STATE\_RES

Request servo 6 state.

*Identifier: 0x0063*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Servo enabled | Minimum angle limit | Maximum angle limit | Current draw |
| Value | *0 – Servo disabled*  *1 – servo enabled* | *Degrees* | *Degrees* | *Current in milliamps* |
| Data type | uint8\_t | float32\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 | 4 |

## SERVO6\_SET\_STATE

Get servo 6 range.

*Identifier: 0x0064*

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Servo enabled | Minimum angle limit | Maximum angle limit |
| Value | *0 – Servo disabled*  *1 – servo enabled* | *Degrees* | *Degrees* |
| Data type | uint8\_t | float32\_t | float32\_t |
| Length (bytes) | 1 | 4 | 4 |

## SERVO6\_POSITION\_REQ

Request servo 6 angle.

*Identifier: 0x0065*

## SERVO6\_POSITION\_RES

Request servo 6 position.

*Identifier: 0x0066*

|  |  |
| --- | --- |
| Name | Servo position |
| Value | *0 – min range, 65535 – max range* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## DC\_MOTOR1\_PWM\_SET

Set DC motor 1 PWM duty cycle.

*Identifier: 0x0067*

|  |  |
| --- | --- |
| Name | Motor PWM |
| Value | *0 – 0% duty cycle,*  *65535 – 100% duty cycle* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## DC\_MOTOR1\_VELOCITY\_SET

Set DC motor 1 PWM duty cycle.

*Identifier: 0x0068*

|  |  |
| --- | --- |
| Name | Motor velocity |
| Value | *Encoder ticks per second* |
| Data type | uint32\_t |
| Length (bytes) | 4 |

## DC\_MOTOR1\_POSITION\_SET

Set DC motor 1 position.

*Identifier: 0x0069*

|  |  |
| --- | --- |
| Name | Motor position |
| Value | *Encoder ticks* |
| Data type | uint32\_t |
| Length (bytes) | 4 |

## DC\_MOTOR1\_STATE\_SET

Set DC motor 1 state.

*Identifier: 0x006A*

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Motor enable | Home motor | Current limit |
| Value | *0 – Disabled, 1 - enabled* | *0 – No change*  *1 – Home motor* | *Current in 0.1 milliamps* |
| Data type | uint8\_t | uint8\_t | uint32\_t |
| Length (bytes) | 1 | 1 | 4 |

## DC\_MOTOR1\_STATE\_REQ

Requests DC motor 1 state.

*Identifier: 0x006B*

## DC\_MOTOR1\_STATE\_RES

DC motor 1 state response.

*Identifier: 0x006C*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Motor enable | Motor homed | Motor position | Motor velocity | Motor current |
| Value | *0 – Disabled,*  *1 - enabled* | *0 – not homed,*  *1 - homed* | *Encoder ticks* | *Encoder ticks per second* | *Current in 0.1 milliamps* |
| Data type | uint8\_t | uint8\_t | uint32\_t | uint32\_t | uint32\_t |
| Length (bytes) | 1 | 1 | 4 | 4 | 4 |

## DC\_MOTOR2\_PWM\_SET

Set DC motor 2 PWM duty cycle.

*Identifier: 0x006D*

|  |  |
| --- | --- |
| Name | Motor PWM |
| Value | *0 – 0% duty cycle,*  *65535 – 100% duty cycle* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## DC\_MOTOR2\_VELOCITY\_SET

Set DC motor 2 PWM duty cycle.

*Identifier: 0x006E*

|  |  |
| --- | --- |
| Name | Motor velocity |
| Value | *Encoder ticks per second* |
| Data type | uint32\_t |
| Length (bytes) | 4 |

## DC\_MOTOR2\_POSITION\_SET

Set DC motor 2 position.

*Identifier: 0x006F*

|  |  |
| --- | --- |
| Name | Motor position |
| Value | *Encoder ticks* |
| Data type | uint32\_t |
| Length (bytes) | 4 |

## DC\_MOTOR2\_STATE\_SET

Set DC motor 2 state.

*Identifier: 0x0070*

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Motor enable | Home motor | Current limit |
| Value | *0 – Disabled, 1 - enabled* | *0 – No change*  *1 – Home motor* | *Current in 0.1 milliamps* |
| Data type | uint8\_t | uint8\_t | uint32\_t |
| Length (bytes) | 1 | 1 | 4 |

## DC\_MOTOR2\_STATE\_REQ

Requests DC motor 2 state.

*Identifier: 0x0071*

## DC\_MOTOR2\_STATE\_RES

DC motor 2 state response.

*Identifier: 0x0072*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Motor enable | Motor homed | Motor position | Motor velocity | Motor current |
| Value | *0 – Disabled,*  *1 - enabled* | *0 – not homed,*  *1 - homed* | *Encoder ticks* | *Encoder ticks per second* | *Current in 0.1 milliamps* |
| Data type | uint8\_t | uint8\_t | uint32\_t | uint32\_t | uint32\_t |
| Length (bytes) | 1 | 1 | 4 | 4 | 4 |

## DC\_MOTOR3\_PWM\_SET

Set DC motor 3 PWM duty cycle.

*Identifier: 0x0073*

|  |  |
| --- | --- |
| Name | Motor PWM |
| Value | *0 – 0% duty cycle,*  *65535 – 100% duty cycle* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## DC\_MOTOR3\_VELOCITY\_SET

Set DC motor 3 PWM duty cycle.

*Identifier: 0x0074*

|  |  |
| --- | --- |
| Name | Motor velocity |
| Value | *Encoder ticks per second* |
| Data type | uint32\_t |
| Length (bytes) | 4 |

## DC\_MOTOR3\_POSITION\_SET

Set DC motor 3 position.

*Identifier: 0x0075*

|  |  |
| --- | --- |
| Name | Motor position |
| Value | *Encoder ticks* |
| Data type | uint32\_t |
| Length (bytes) | 4 |

## DC\_MOTOR3\_STATE\_SET

Set DC motor 3 state.

*Identifier: 0x0076*

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Motor enable | Home motor | Current limit |
| Value | *0 – Disabled, 1 - enabled* | *0 – No change*  *1 – Home motor* | *Current in 0.1 milliamps* |
| Data type | uint8\_t | uint8\_t | uint32\_t |
| Length (bytes) | 1 | 1 | 4 |

## DC\_MOTOR3\_STATE\_REQ

Requests DC motor 3 state.

*Identifier: 0x0077*

## DC\_MOTOR3\_STATE\_RES

DC motor 3 state response.

*Identifier: 0x0078*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Motor enable | Motor homed | Motor position | Motor velocity | Motor current |
| Value | *0 – Disabled,*  *1 - enabled* | *0 – not homed,*  *1 - homed* | *Encoder ticks* | *Encoder ticks per second* | *Current in 0.1 milliamps* |
| Data type | uint8\_t | uint8\_t | uint32\_t | uint32\_t | uint32\_t |
| Length (bytes) | 1 | 1 | 4 | 4 | 4 |

## DC\_MOTOR4\_PWM\_SET

Set DC motor 4 PWM duty cycle.

*Identifier: 0x0079*

|  |  |
| --- | --- |
| Name | Motor PWM |
| Value | *0 – 0% duty cycle,*  *65535 – 100% duty cycle* |
| Data type | uint16\_t |
| Length (bytes) | 2 |

## DC\_MOTOR4\_VELOCITY\_SET

Set DC motor 4 PWM duty cycle.

*Identifier: 0x007A*

|  |  |
| --- | --- |
| Name | Motor velocity |
| Value | *Encoder ticks per second* |
| Data type | uint32\_t |
| Length (bytes) | 4 |

## DC\_MOTOR4\_POSITION\_SET

Set DC motor 4 position.

*Identifier: 0x007B*

|  |  |
| --- | --- |
| Name | Motor position |
| Value | *Encoder ticks* |
| Data type | uint32\_t |
| Length (bytes) | 4 |

## DC\_MOTOR4\_STATE\_SET

Set DC motor 4 state.

*Identifier: 0x007C*

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Motor enable | Home motor | Current limit |
| Value | *0 – Disabled, 1 - enabled* | *0 – No change*  *1 – Home motor* | *Current in 0.1 milliamps* |
| Data type | uint8\_t | uint8\_t | uint32\_t |
| Length (bytes) | 1 | 1 | 4 |

## DC\_MOTOR4\_STATE\_REQ

Requests DC motor 4 state.

*Identifier: 0x007D*

## DC\_MOTOR4\_STATE\_RES

DC motor 4 state response.

*Identifier: 0x007E*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Motor enable | Motor homed | Motor position | Motor velocity | Motor current |
| Value | *0 – Disabled,*  *1 - enabled* | *0 – not homed,*  *1 - homed* | *Encoder ticks* | *Encoder ticks per second* | *Current in 0.1 milliamps* |
| Data type | uint8\_t | uint8\_t | uint32\_t | uint32\_t | uint32\_t |
| Length (bytes) | 1 | 1 | 4 | 4 | 4 |

## PRESSURE\_SENSOR1\_READ\_REQ

Read pressure sensor 1 request.

*Identifier: 0x007F*

## PRESSURE\_SENSOR1\_READ\_RES

Read pressure sensor 1 request.

*Identifier: 0x0080*

|  |  |  |
| --- | --- | --- |
| Name | Raw ADC value | Pressure reading |
| Value |  | *Pressure in millibar* |
| Data type | uint16\_t | uint32\_t |
| Length (bytes) | 2 | 4 |

## PRESSURE\_SENSOR2\_READ\_REQ

Read pressure sensor 2 request.

*Identifier: 0x0081*

## PRESSURE\_SENSOR2\_READ\_RES

Read pressure sensor 2 request.

*Identifier: 0x0082*

|  |  |  |
| --- | --- | --- |
| Name | Raw ADC value | Pressure reading |
| Value |  | *Pressure in millibar* |
| Data type | uint16\_t | uint32\_t |
| Length (bytes) | 2 | 4 |

## PRESSURE\_SENSOR3\_READ\_REQ

Read pressure sensor 3 request.

*Identifier: 0x0083*

## PRESSURE\_SENSOR3\_READ\_RES

Read pressure sensor 3 request.

*Identifier: 0x0084*

|  |  |  |
| --- | --- | --- |
| Name | Raw ADC value | Pressure reading |
| Value |  | *Pressure in millibar* |
| Data type | uint16\_t | uint32\_t |
| Length (bytes) | 2 | 4 |

## PRESSURE\_SENSOR4\_READ\_REQ

Read pressure sensor 4 request.

*Identifier: 0x0085*

## PRESSURE\_SENSOR4\_READ\_RES

Read pressure sensor 4 request.

*Identifier: 0x0086*

|  |  |  |
| --- | --- | --- |
| Name | Raw ADC value | Pressure reading |
| Value |  | *Pressure in millibar* |
| Data type | uint16\_t | uint32\_t |
| Length (bytes) | 2 | 4 |

## TEMPERATURE\_SENSOR1\_READ\_REQ

Read pressure sensor 1 request.

*Identifier: 0x0087*

## TEMPERATURE\_SENSOR1\_READ\_RES

Read pressure sensor 1 request.

*Identifier: 0x0088*

|  |  |  |
| --- | --- | --- |
| Name | Raw ADC value | Temperature reading |
| Value |  | *Temperature in 0.01 degrees Kelvin* |
| Data type | uint16\_t | uint32\_t |
| Length (bytes) | 2 | 4 |

## TEMPERATURE\_SENSOR2\_READ\_REQ

Read pressure sensor 2 request.

*Identifier: 0x0089*

## TEMPERATURE\_SENSOR2\_READ\_RES

Read pressure sensor 2 request.

*Identifier: 0x008A*

|  |  |  |
| --- | --- | --- |
| Name | Raw ADC value | Temperature reading |
| Value |  | *Temperature in 0.01 degrees Kelvin* |
| Data type | uint16\_t | uint32\_t |
| Length (bytes) | 2 | 4 |

## TEMPERATURE\_SENSOR3\_READ\_REQ

Read pressure sensor 3 request.

*Identifier: 0x008B*

## TEMPERATURE\_SENSOR3\_READ\_RES

Read pressure sensor 3 request.

*Identifier: 0x008C*

|  |  |  |
| --- | --- | --- |
| Name | Raw ADC value | Temperature reading |
| Value |  | *Temperature in 0.01 degrees Kelvin* |
| Data type | uint16\_t | uint32\_t |
| Length (bytes) | 2 | 4 |

## TEMPERATURE\_SENSOR4\_READ\_REQ

Read pressure sensor 4 request.

*Identifier: 0x008D*

## TEMPERATURE\_SENSOR4\_READ\_RES

Read pressure sensor 4 request.

*Identifier: 0x008E*

|  |  |  |
| --- | --- | --- |
| Name | Raw ADC value | Temperature reading |
| Value |  | *Temperature in 0.01 degrees Kelvin* |
| Data type | uint16\_t | uint32\_t |
| Length (bytes) | 2 | 4 |