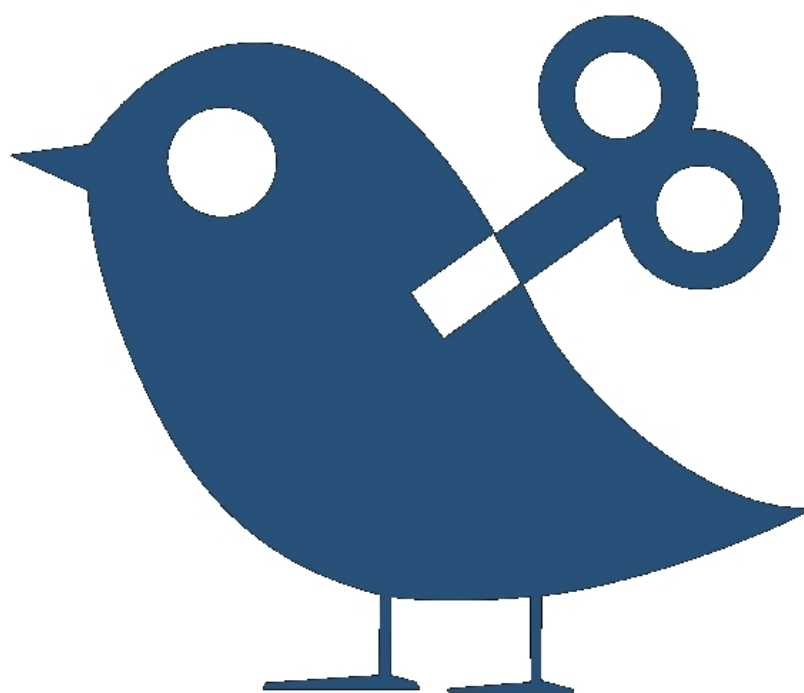


Turret

CANaerospace



Edition 06/17/2023

Contents

1	INTRODUCTION	4
2	COMMUNICATION LAYERS	4
3	DATA TYPES.....	6
4	MESSAGE STRUCTURE	8
4.1	GENERAL MESSAGE FORMAT	8
4.2	NODE SERVICE DATA (NSH/NSL) MESSAGE FORMAT	9
4.3	EMERGENCY EVENT DATA (EED) MESSAGE FORMAT	9
5	NODE SERVICE PROTOCOL.....	9
5.1	IDENTIFICATION SERVICE (IDS)	12
5.2	DATA DOWNLOAD SERVICE (DDS)	13
5.3	DATA UPLOAD SERVICE (DUS)	14
5.4	TRANSMISSION INTERVAL SERVICE (TIS)	15
5.5	FLASH PROGRAMMING SERVICE (FPS)	15
5.6	NODE-ID SETTING SERVICE (NIS).....	16
6	DESCRIPTION OF COMMAND IDENTIFIERS	17
6.1	TURRET COMMANDS.....	17
6.1.1	<i>Stop Actuators</i>	<i>17</i>
6.1.2	<i>Yaw Position Set-Point.....</i>	<i>17</i>
6.1.3	<i>Yaw Velocity Set-Point.....</i>	<i>17</i>
6.1.4	<i>Yaw Relative Position Set-Point.....</i>	<i>17</i>
6.1.5	<i>Yaw Actuator Mode.....</i>	<i>18</i>
6.1.1	<i>Pitch Position Set-Point.....</i>	<i>18</i>
6.1.2	<i>Pitch Velocity Set-Point.....</i>	<i>18</i>
6.1.3	<i>Pitch Relative Position Set-Point.....</i>	<i>18</i>
6.1.4	<i>Pitch Actuator Mode.....</i>	<i>19</i>
6.1.5	<i>Remember Current Position.....</i>	<i>19</i>
6.1.6	<i>Set To Saved Position.....</i>	<i>19</i>
6.2	CAMERA COMMANDS	20
6.2.1	<i>Camera1 Focus Control.....</i>	<i>20</i>
6.2.2	<i>Camera1 Diaphragm Control.....</i>	<i>20</i>
6.2.3	<i>Camera2 Focus Control.....</i>	<i>20</i>
6.2.4	<i>Camera2 Diaphragm Control.....</i>	<i>20</i>
6.2.5	<i>Camera3 Focus Control.....</i>	<i>21</i>
6.2.6	<i>Camera3 Diaphragm Control.....</i>	<i>21</i>
6.2.7	<i>Camera4 Focus Control.....</i>	<i>21</i>
6.2.8	<i>Camera4 Diaphragm Control.....</i>	<i>21</i>
6.3	COVER COMMANDS	21
6.3.1	<i>Cover Control</i>	<i>21</i>
6.4	FAN COMMANDS	22
6.4.1	<i>Fan Control</i>	<i>22</i>
6.5	BOLT COMMANDS	22
6.5.1	<i>Bolt Control.....</i>	<i>22</i>
6.5.2	<i>Bolt Stroke</i>	<i>22</i>

6.6	FUSE COMMANDS	22
6.6.1	Fuse Control.....	22
6.7	SOFTWARE TRIGGER COMMANDS.....	23
6.7.1	Trigger Fuse	23
6.7.2	Trigger Control.....	23
6.7.1	Trigger Heartbeat.....	23
6.8	BATTERY COMMANDS	23
6.8.1	Charging Control.....	23
6.9	HEATER COMMANDS	24
6.9.1	OES Heater Control.....	24
6.9.2	Drives Heater Control	24
7	DESCRIPTION OF DATA IDENTIFIERS.....	24
7.1	TURRET DATA.....	24
7.1.1	Yaw Actual Position	24
7.1.2	Yaw Actual Velocity	24
7.1.3	Yaw Motor Temperature	24
7.1.4	Yaw Current Actuator Mode.....	25
7.1.5	Yaw Current Bus Voltage.....	25
7.1.6	Yaw Actual Current.....	25
7.1.7	Yaw Current Humidity	25
7.1.8	Yaw Power Stage Temperature	25
7.1.9	Yaw Number Of Revolution	25
7.1.10	Yaw Current Status	26
7.1.11	Pitch Actual Position	26
7.1.12	Pitch Actual Velocity.....	26
7.1.13	Pitch Motor Temperature.....	26
7.1.14	Pitch Current Actuator Mode.....	26
7.1.15	Pitch Current Bus Voltage.....	27
7.1.16	Pitch Actual Current.....	27
7.1.17	Pitch Current Humidity	27
7.1.18	Pitch Power Stage Temperature	27
7.1.19	Pitch Number Of Revolution	27
7.1.20	Pitch Current Status	27
7.1.21	Bolt State	28
7.1.22	Fan State.....	28
7.1.23	Case Temperature	28
7.1.24	Charging current.....	28
7.1.25	Charging State	28
7.1.26	Fuse State	29
7.1.27	Global Shot Counter.....	29
7.1.28	OES Heater State	29
7.1.29	Drives Heater State.....	29
7.1.30	Periphery Status.....	30
7.1.31	Global Pitch Position (Inclinometer)	30
7.1.32	Global Roll Position (Inclinometer)	30
7.2	COVER DATA	31
7.2.1	Cover State (always sent by default at 100ms interval)	31
7.3	ROVER GNSS DATA.....	31
7.3.1	Current Heading	31
7.3.2	Current Accuracy	31
7.3.3	Global Yaw Position (heading).....	31
7.4	BASE GNSS DATA.....	31
7.4.1	Latitude.....	31

7.4.2	Longitude.....	32
7.4.3	Altitude Mean Sea Level.....	32
7.4.4	Position Accuracy Estimate.....	32
7.4.5	UTC Time	32
7.5	ZOOM DATA.....	32
7.5.1	Camera0 Zoom Position.....	32

1 Introduction

CANaerospace is an extremely lightweight protocol/data format definition which was designed for the highly reliable communication of microcomputer-based systems in airborne applications via CAN (Controller Area Network). The purpose of this definition is to create a standard for applications requiring an efficient data flow monitoring and easy time-frame synchronization within redundant systems. The definition is kept widely open to allow implementation of user-defined message types and protocols. CANaerospace can be used with CAN 2.0A and 2.0B (11-bit and 29-bit identifiers) and any bus data rate.

2 Communication layers

Enabling both ATM and PTP communication for CAN requires the introduction of independent network layers to isolate the different types of communication. This is realized for CANaerospace by forming CAN identifier groups as shown in Figure 1. The resulting structure creates Logical Communication Channels (LCCs) and assigns a specific communication type (ATM, PTP) to each of the LCCs. User-defined LCCs provide the necessary freedom for designers and allow the implementation of CANaerospace according to the needs of specific applications.


Channel Acronym	Communication Type	Description	CAN Identifier Range	Message Priority
EED	ATM	Emergency Event Data Channel	0 - 127	Highest
NSH	PTP	High Priority Node Service Data Channel	128 - 199	
UDH	ATM/PTP	High Priority User-Defined Data Channel	200 - 299	
NOD	ATM	Normal Operation Data Channel	300 - 1799	
UDL	ATM/PTP	Low Priority User-Defined Data Channel	1800 - 1899	
DSD	ATM/PTP	Debug Service Data Channel	1900 - 1999	
NSL	PTP	Low Priority Node Service Data Channel	2000 - 2031	Lowest

Figure 1. Logical Communication Channels for CANaerospace

As a side effect, the CAN identifier groups in Figure 1 affect the priority of the message transmission in case of bus arbitration. The communication channels are therefore arranged according to their relative importance:

- **Emergency Event Data Channel (EED):** This communication channel is used for messages which require immediate action (i.e. system degradation or reconfiguration) and have to be transmitted with very high priority. Emergency Event Data uses ATM communication exclusively.
- **High/Low Priority Node Service Data Channel (NSH/NSL):** These communication channels are used for client/server interactions using PTP communication. The

corresponding services may be of the connection-oriented as well as the connectionless type. NSH/NSL may also be used to support test and maintenance functions.

- **Normal Operation Data Channel (NOD):** This communication channel is used for the transmission of the data which is generated during normal system operation and described in the CANaerospace identifier assignment list. These messages may be transmitted periodically or aperiodically as well as synchronously or asynchronously. All messages which cannot be assigned to other communication channels shall use this channel.
- **High/Low Priority User-Defined Data Channel (UDH/UDL):** This channel is dedicated to communication which cannot, due to their specific characteristics, be assigned other channels without violating the CANaerospace specification. As long as the defined identifier range is used, the message content and the communication type (ATM, PTP) for these channels may be specified by the system designer. To ensure interoperability it is highly recommended that the use of these channels is minimized.
- **Debug Service Data Channel (DSD):** This channel is dedicated to messages which are used temporarily for development and test purposes only and are not transmitted during normal operation. As long as the defined identifier range is used, the message content and the communication type (ATM, PTP) for these channels may be specified by the system designer.

3 Data types

For data representation, the most commonly used basic data types are defined. Additionally, combined data types (i.e. two, three and four 16-bit and 8-bit data types in one CAN message) and aggregate data types (64-bit double float) are supported. Other data types can be added to the type list as required. The type number in the range of 0-255 is used for data type specification as described in section 4.

Data Type	Range	Bits	Explanation	Type
NODATA	n.a.	0	«No data» type	0 (0x00)
ERROR	n.a.	32	Emergency event data type	1 (0x01)
FLOAT	1-bit sign 23-bit fraction 8-bit exponent	32	Single precision floating-point value according to IEEE-754-1985	2 (0x02)
LONG	-2147483647 to 2147483648	32	2's complement integer	3 (0x03)
ULONG	0 to 4294967295	32	unsigned integer	4 (0x04)
BLONG	n.a.	32	Each bit defines a discrete state. 32 bits are coded into four CAN data bytes	5 (0x05)
SHORT	-32768 to +32767	16	2's complement short integer	6 (0x06)
USHORT	0 to 65535	16	unsigned short integer	7 (0x07)
BSHORT	n.a.	16	Each bit defines a discrete state. 16 bits are coded into two CAN data bytes	8 (0x08)
CHAR	-128 to +127	8	2's complement char integer	9 (0x09)
UCHAR	0 to 255	8	unsigned char integer	10 (0x0A)
BCHAR	n.a.	8	Each bit defines a discrete state. 8 bits are coded into a single CAN data byte	11 (0x0B)
SHORT2	-32768 to +32767	2 x 16	2 x 2's complement short integer	12 (0x0C)
USHORT2	0 to 65535	2 x 16	2 x unsigned short integer	13 (0x0D)
BSHORT2	n.a.	2 x 16	2 x discrete short	14 (0x0E)

Data Type	Range	Bits	Explanation	Type
CHAR4	-128 to +127	4 x 8	4 x 2's complement char integer	15 (0x0F)
UCHAR4	0 to 255	4 x 8	4 x unsigned char integer	16 (0x10)
BCHAR4	n.a.	4 x 8	4 x discrete char	17 (0x11)
CHAR2	-128 to +127	2 x 8	2 x 2's complement char integer	18 (0x12)
UCHAR2	0 to 255	2 x 8	2 x unsigned char integer	19 (0x13)
BCHAR2	n.a.	2 x 8	2 x discrete char	20 (0x14)
MEMID	0 to 4294967295	32	Memory ID for upload/download	21 (0x15)
CHKSUM	0 to 4294967295	32	Checksum for upload/download	22 (0x16)
ACHAR	0 to 255	8	ASCII character	23 (0x17)
ACHAR2	0 to 255	2 x 8	2 x ASCII character	24 (0x17)
ACHAR4	0 to 255	4 x 8	4 x ASCII character	25 (0x17)
CHAR3	-128 to +127	3 x 8	3 x 2's complement char integer	26 (0x17)
UCHAR3	0 to 255	3 x 8	3 x unsigned char integer	27 (0x17)
BCHAR3	n.a.	3 x 8	3 x discrete char	28 (0x17)
ACHAR3	0 to 255	3 x 8	4 x ASCII character	29 (0x17)
DOUBLEH	1-bit sign 52-bit fraction 11-bit exponent	32	Most significant 32 bits of double precision floating-point value according to IEEE-754-1985	30 (0x17)
DOUBLEL	1-bit sign 52-bit fraction 11-bit exponent	32	Least significant 32 bits of double precision floating-point value according to IEEE-754-1985	31 (0x17)
RESVD	n.a.	xx	Reserved for future use	32-99 (0x20-0x63)
UDEF	n.a.	xx	User-defined data types	102-255 (0x66-0xFF)

4 Message structure

The coding of the data into the CAN message bytes is according to the «**Big Endian**» definition as used by Motorola 68K, SPARC, PowerPC, MIPS and other major processor architectures. All CAN messages consist of 4 header bytes for identification and between 1 and 4 data bytes for the actual data.

4.1 General message format

The general message format uses a 4-byte message header for node identification, data type, message code and service code (for normal operation data (NOD), the service code field is user-defined). This allows identification of each message by any receiving unit without the need for additional information. Every message type uses the same layout for the CAN data bytes 0-3, while the number and the data type used for CAN data bytes 4-7 is user-defined:

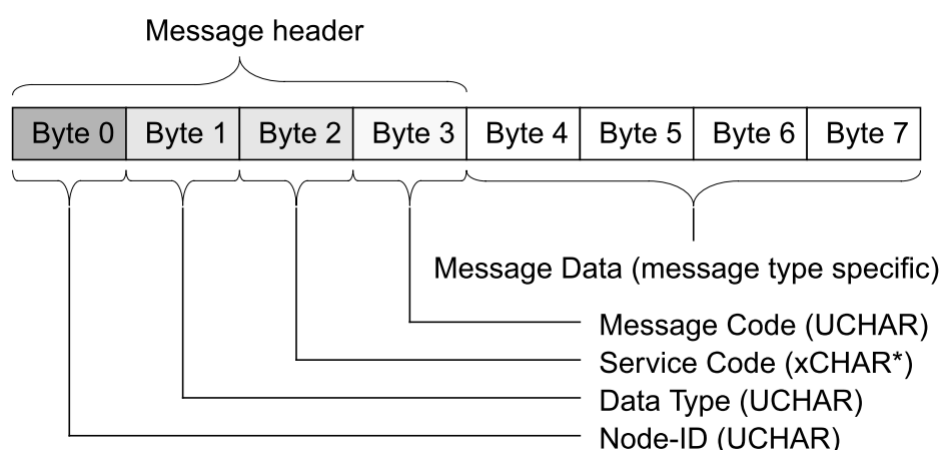


Figure 2. CANaerospace Self-Identifying Message Format

- **Node-ID:** For ATM communication (EED, NOD), the Node Identifier specifies the transmitting node. For PTP communication (NSH, NSL) it specifies the addressed node (client, server). For PTP communication, Node_ID «0» is used to address all stations in the network (multicast).
- **Data Type:** The Data Type specifies how the payload of the message shall be interpreted with respect to its data type (i.e. floating-point data or number of bytes in case of integer data). The corresponding data type code is taken from the CANaerospace data type list which allows also user-defined data type definitions.
- **Service Code:** For Normal Operation Data (NOD) the Service Code delivers information about the integrity of the parameter transmitted with the message. This may be the result of a continuous sensor built-in test, the current validity flag of a navigation signal or other parameter specific information. In case of PTP communication the Service Code specifies the service for the corresponding client/server interaction.
- **Message Code:** For Normal Operation Data (NOD) the Message Code is incremented by one for each message with a particular CAN identifier by the transmitting node. After reaching the value of 255, the Message Code rolls over to zero. This allows receiving stations to determine missing or delayed messages and to react accordingly.

Concerning PTP communication (NSH, NSL) the Message Code is used in conjunction with the Service Code to specify the service for the corresponding client/server interaction in more detail.

4.2 Node Service Data (NSH/NSL) message format

Node Service Data (NSH/NSL) is data associated to the node service protocol as specified in section 5. The message format is similar to NOD. Node service data, however, is transmitted on specific identifiers only:

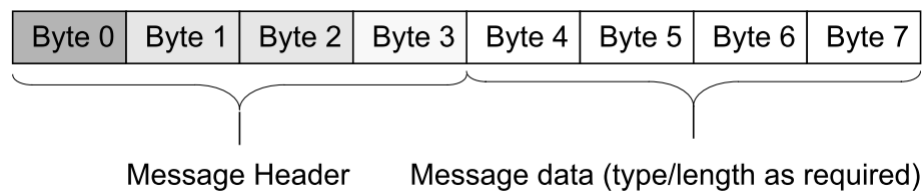


Figure 3. Node Service Data Message Format

4.3 Emergency Event Data (EED) message format

Emergency Event Data (EED) is transmitted asynchronously by the affected unit whenever an error situation occurs. The corresponding data contains information about the location within the unit at which the error occurred, the offending operation and the error code:

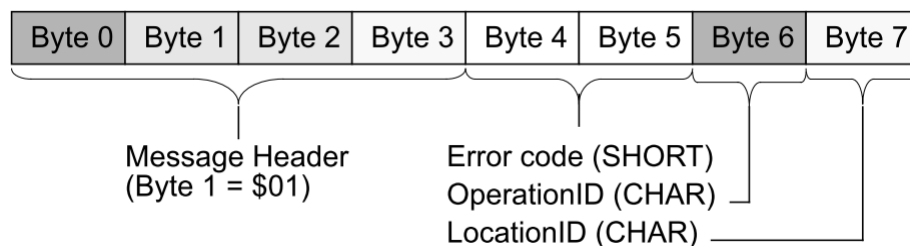


Figure 4. Emergency Event Data Message Format

5 Node service protocol

In parallel to the data transfer during normal operation (Emergency Event Data, Normal Operation Data), the node service protocol provides a connection-oriented communication using a handshake mechanism. This protocol has been implemented to support command/response type connections between two nodes for specific operations, i.e. for data download or client/server actions. Note that node service requests requiring action but no response are possible as well. Requests of this type may be sent to a specific or all nodes (broadcast).

The node service protocol may be run either in high priority or low priority mode, selected by identifier. For the high priority mode, 36 node service communication channels

are available, while the low priority mode offers 16 communication channels. Each communication channel consists of one CAN identifier for the node service request and the immediately following one for the node service response. The identifier assignment for the high priority node service channels is as follows:

Node Service Channel	Node Service Request ID	Node Service Response ID
0	128 (0x80)	129 (0x81)
1	130 (0x82)	131 (0x83)
...
34	196 (0xC4)	197 (0xC5)
35	198 (0xC6)	199 (0xC7)

This is the identifier assignment for the low priority node service channels:

Node Service Channel	Node Service Request ID	Node Service Response ID
100	2000 (0x7D0)	2001 (0x7D1)
101	2002 (0x7D2)	2003 (0x7D3)
...
114	2028(0x7EC)	2029 (0x7ED)
115	2030 (0x7EE)	2031 (0x7EF)

A node service is initiated by a node service request message, transmitted on the corresponding identifier. All nodes attached to the network are obliged to continuously monitor these identifiers and check if received messages contain the own personal node-ID. If a match is detected, the corresponding node has to react by performing the required action and transmitting a node service response message on the corresponding identifier within 100ms (if this was required by the request type). The node service response must again contain the personal node-ID of the addressed node. Any node in the network is allowed to initiate node services. It is recommended, however, that each node in the network initiating node service requests uses a dedicated node service channel to avoid potential hand-shaking conflicts. The channel on which a particular node service is run may be defined by the user. If only one service channel is used, node services must be run on channel 0 by default. Some frequently used types of node services are already specified below, other services may be added as required.

Each CANaerospace unit must support at least the Identification Service (**IDS**) on Node Service Channel 0. This makes sure that a CANaerospace network can be scanned for attached units to determine their status, header type and identifier assignment. Note that within a CANaerospace network, other header types than the standard CANaerospace header and several identifier assignment schemes (including entirely user-defined ones)

are supported. Whenever possible, it is strongly recommended to use the proposed standard header and identifier assignment, however.

Node Service	Service Code	Response Required	Action
IDS	0	Yes	Identification service. Requests a « sign-of-life » response together with configuration information from the addressed node.
NSS	1	No	Node synchronization service, used to trigger a specific node or to perform a network wide time synchronization.
DDS	2	Yes	Data download service. Sends a block of data to another node.
DUS	3	Yes	Data upload service. Receives a block of data from another node.
SCS	4	Yes	Simulation Control Service. Allows to change the behavior of the addressed node by controlling internal simulation software functions.
TIS	5	Yes	Transmission Interval Service. Sets the transmission rate of a specific CAN message transmitted by the addressed node.
FPS	6	Yes	FLASH Programming Service. Triggers a node-internal routine to store configuration data permanently into non-volatile memory.
STS	7	No	State Transmission Service. Causes the addressed node to transmit all its CAN messages once.
FSS	8	Yes	Filter Setting Service. Used to modify the limit frequencies of node-internal highpass, lowpass or bandpass filters.
TCS	9	Yes	Test Control Service. Triggers internal test functions of the addressed node.
BSS	10	No/Yes (result dependent)	CAN Baudrate Setting Service. Sets the CAN baudrate of the addressed node.
NIS	11	Yes	Node-ID Setting Service. Sets the Node-ID of the addressed node.
MIS	12	Yes	Module Information Service. Obtains information about installed modules within the addressed node.

MCS	13	Yes	Module Configuration Service. Configures installed modules within the addressed node.
CSS	14	Yes	CAN-ID setting service. Sets the CAN-ID of a specific message the addressed node transmits.
DSS	15	Yes	CAN-ID Distribution Setting Service. Sets the identifier distribution code transmitted as Byte 2 of the response to an IDS request.
XXS	16-99		Reserved for future use.
	100-255		User-defined services.

5.1 Identification Service (IDS)

The identification service is a client/server type service. It is used to obtain a «**sign-of-life**» indication from the addressed node and check if its identifier distribution and message header format is compliant with the network that it is attached to. The addressed node returns a 4-byte status information which contains information about hardware/software revision of the system, together with the identifier distribution and message header format the node is programmed for. Consequently, the data type of the response message is UCHAR4.

The term «**identifier distribution**» refers to the assignment of CAN identifiers to physical data objects.

Distribution ID (Byte 2 of IDS response)	Identifier distribution
0	CANaerospace standard
1-99	Reserved for future use
100-255	User-defined distribution schemes

Likewise, a CANaerospace network may choose to deviate from the standard CANaerospace header format as described in section 4.1. If this is the case, byte 3 of the message must inform an interrogating node about this fact by returning a non-zero (user-defined) value in byte 3 of the IDS response. Note that in this case the IDS service must still be supported using the CANaerospace standard header format:

Message Data Byte	Data Field Description	Service Request	Service Response
0	Node-ID	<node-ID>	<node-ID>
1	Data Type	NODATA	UCHAR4
2	Service Code	0	0
3	Message Code	<0>	<as in request>

4-7	Message Data	n.a.	Byte 0: Hardware Revision Byte 1: Software Revision Byte 2: Identifier Distribution (0 = CANaerospace standard distribution) Byte 3: Header Type (0 = CANaerospace standard header)
-----	--------------	------	---

5.2 Data Download Service (DDS)

The data download is a connection-oriented service and is used to send a block of data to another node. The size of the data block may be in the range of 1-1020 bytes, specified by the message number field of the header. To initiate the service, the requesting station sends a «**start download request message**» to the addressed node, specifying a memory destination identifier and the number of messages which will be transmitted (the data type may change during the download to allow structured data to be sent). It then waits for the response. If the service response is received within 100ms and the message data is XON, the requesting station may transmit the specified number of data messages:

Message Data Byte	Data Field Description	Service Request	Service Response
0	Node-ID	<node-ID>	<node-ID>
1	Data Type	MEMID	LONG
2	Service Code	2	2
3	Message Code	<0-255>	<as in request>
4-7	Message Data	<memory destination identifier>	-2 = INVALID -1 = ABORT 0 = XOFF 1 = XON

The addressed node now accepts data until the final message number has been reached. To control download speed, the addressed node may send a service response at any time during the download process, specifying the current message number and XOFF or XON. By specifying ABORT or INVALID, the download is cancelled immediately without further action. The transmitting station has to react correspondingly by stopping or resuming data transmission.

After the last message has been received, the addressed node transmits a service response with a checksum calculated from summing up all received data. This allows the requesting node to determine if all data has been received properly:

Message Data Byte	Data Field Description	Service Request	Service Response
0	Node-ID	<node-ID>	<node-ID>
1	Data Type	<any>	CHKSUM
2	Service Code	2	2
3	Message Code	<last number>	<last number>
4-7	Message Data	<download data>	<download data>

5.3 Data Upload Service (DUS)

The data upload is a connection-oriented service and is used to receive a block of data from another node. The size of the data block may be in the range of 1-1020 bytes, specified by the message number field of the header. To initiate the service, the requesting station sends a «**start upload request message**» to the addressed node, specifying the source memory identifier and the number of messages which is expected to be received. It then waits up to 100ms for a service response. After having transmitted the service response, the addressed station waits 10ms and then transmits the requested number of data messages:

Message Data Byte	Data Field Description	Service Request	Service Response
0	Node-ID	<node-ID>	<node-ID>
1	Data Type	MEMID	LONG
2	Service Code	3	3
3	Message Code	<0-255>	<as in request>
4-7	Message Data	<source memory identifier>	-1 = ABORT 0 = OK

The requesting node now accepts data at the maximum transmission speed until the final message number has been reached. After the last data message, the addressed node transmits a service response with a checksum calculated from summing up all transmitted data. This allows the requesting node to determine if all data has been received properly. Additionally, the requesting node continuously checks the proper message number sequence during the process to detect failures:

Message Data Byte	Data Field Description	Service Response
0	Node-ID	<node-ID>
1	Data Type	CHKSUM
2	Service Code	3

3	Message Code	<last number>
4-7	Message Data	<checksum>

5.4 Transmission Interval Service (TIS)

The transmission interval service is a connection-oriented service and is used to set the transmission rate of a specific CAN message transmitted by the addressed node. This is accomplished by specifying the corresponding CAN identifier and the desired transmission rate for this data in milliseconds. The allowed range for the CAN identifier and the transmission rate is user-defined:

Message Data Byte	Data Field Description	Service Request	Service Response
0	Node-ID	<node-ID>	<node-ID>
1	Data Type	SHORT2	NODATA
2	Service Code	5	5
3	Message Code	0	0 = OK -6 = CAN identifier or transmission rate out of range
4-5	Message Data	<CAN identifier>	n.a.
6-7	Message Data	<transmission rate in ms>	n.a.

5.5 FLASH Programming Service (FPS)

The FLASH programming service is a connection-oriented service used to store configuration data (i.e. settings made through BSS, NIS, CSS or similar services) into internal non-volatile memory. It usually triggers a node-internal software programming routine. A user-defined security code (transmitted as the message code) makes sure that this service can only be issued by “authorized” nodes in the network:

Message Data Byte	Data Field Description	Service Request	Service Response
0	Node-ID	<node-ID>	<node-ID>
1	Data Type	NODATA	NODATA
2	Service Code	6	6
3	Message Code	<security code>	0 = OK -3 = invalid security code
4-7	Message Data	n.a.	n.a.

5.6 Node-ID Setting Service (NIS)

The Node-ID setting service is a connection-oriented service used to set the ID of the addressed node. The new ID becomes effective immediately and is used by the affected node a.) to identify itself in the Node-ID field for all normal operation data (NOD) messages and b.) to determine if it is addressed by received node service requests:

Message Data Byte	Data Field Description	Service Request	Service Response
0	Node-ID	<node-ID>	<node-ID>
1	Data Type	NODATA	NODATA
2	Service Code	11	11
3	Message Code	<new node-ID> 1 <= ID <= 199	0 = OK -6 = node-ID out of range
4-7	Message Data	n.a.	n.a.

6 Description of Command Identifiers

6.1 Turret Commands

All Examples of commands for master with Node-ID = 100 (0x64).

6.1.1 Stop Actuators

CAN identifier	Parameter Name	Data Type	Units	Notes
300 (0x12C)	Stop actuators	UCAR		-

Command Byte

Bit Number	Bit Name	Description
0	Stop yaw actuator	1 = Stop
1	Stop pitch actuator	1 = Stop

6.1.2 Yaw Position Set-Point

CAN identifier	Parameter Name	Data Type	Units	Notes
1300 (0x514)	Yaw position set-point	FLOAT	deg	-180° ÷ 180°

Example: move to position 20 degree

Data[0]	Data[1]	Data[2]	Data[3]	Data[4]	Data[5]	Data[6]	Data[7]
0x64	0x02	0x00	0x00	0x41	0xA0	0x00	0x00

6.1.3 Yaw Velocity Set-Point

CAN identifier	Parameter Name	Data Type	Units	Notes
1301 (0x515)	Yaw velocity set-point	FLOAT	deg/s	-

Example: rotate at -10 deg/s

Data[0]	Data[1]	Data[2]	Data[3]	Data[4]	Data[5]	Data[6]	Data[7]
0x64	0x02	0x00	0x00	0xC1	0x20	0x00	0x00

6.1.4 Yaw Relative Position Set-Point

CAN identifier	Parameter Name	Data Type	Units	Notes
1302 (0x516)	Yaw relative position set-point	FLOAT	deg	-

Example: move 90 degrees

Data[0]	Data[1]	Data[2]	Data[3]	Data[4]	Data[5]	Data[6]	Data[7]
0x64	0x02	0x00	0x00	0x42	0xB4	0x00	0x00

6.1.5 Yaw Actuator Mode

CAN identifier	Parameter Name	Data Type	Units	Notes
1307 (0x51B)	Yaw actuator mode	UCHAR	-	-

Command Byte

Parameter Name	Description
Actuator mode	0 – Free run 1 – Without brake 2 – Auto brake

Example: auto brake

Data[0]	Data[1]	Data[2]	Data[3]	Data[4]	Data[5]	Data[6]	Data[7]
0x64	0x0A	0x00	0x00	0x02	0x00	0x00	0x00

6.1.1 Pitch Position Set-Point

CAN identifier	Parameter Name	Data Type	Units	Notes
1310 (0x51E)	Pitch position set-point	FLOAT	deg	-180° ÷ 180°

Example: move to position 20 deg.

Data[0]	Data[1]	Data[2]	Data[3]	Data[4]	Data[5]	Data[6]	Data[7]
0x64	0x02	0x00	0x00	0x41	0xA0	0x00	0x00

6.1.2 Pitch Velocity Set-Point

CAN identifier	Parameter Name	Data Type	Units	Notes
1311 (0x51F)	Pitch velocity set-point	FLOAT	deg/s	-

Example: rotate at -10 deg/s

Data[0]	Data[1]	Data[2]	Data[3]	Data[4]	Data[5]	Data[6]	Data[7]
0x64	0x02	0x00	0x00	0xC1	0x20	0x00	0x00

6.1.3 Pitch Relative Position Set-Point

CAN identifier	Parameter Name	Data Type	Units	Notes
1312 (0x520)	Pitch relative position set-point	FLOAT	deg	

Example: move 90 degrees

Data[0]	Data[1]	Data[2]	Data[3]	Data[4]	Data[5]	Data[6]	Data[7]
0x64	0x02	0x00	0x00	0x42	0xB4	0x00	0x00

6.1.4 Pitch Actuator Mode

CAN identifier	Parameter Name	Data Type	Units	Notes
1317 (0x525)	Pitch actuator mode	UCHAR	-	-

Command Byte

Parameter Name	Description
Actuator mode	0 – Free run 1 – Without brake 2 – Auto brake

Example: auto brake

Data[0]	Data[1]	Data[2]	Data[3]	Data[4]	Data[5]	Data[6]	Data[7]
0x64	0x0A	0x00	0x00	0x02	0x00	0x00	0x00

6.1.5 Remember Current Position

CAN identifier	Parameter Name	Data Type	Units	Notes
1420 (0x58C)	Remember current position	UCHAR	-	-

Command Byte

Parameter Name	Description
Remember current position	1 – Remember current yaw and pitch position

6.1.6 Set To Saved Position

CAN identifier	Parameter Name	Data Type	Units	Notes
1421 (0x58D)	Set to saved position	UCHAR	-	-

Command Byte

Parameter Name	Description
Set to saved position	1 – Set yaw and pitch to saved position

6.2 Camera Commands

6.2.1 Camera1 Focus Control (not use)

CAN identifier	Parameter Name	Data Type	Units	Notes
1340 (0x53C)	Camera1 focus control	CHAR	-	-

Command Byte

Parameter Name	Description
Focus control	-1 – Rotate in one direction 0 – Stop +1 – Rotate in the other direction

Example: rotate

Data[0]	Data[1]	Data[2]	Data[3]	Data[4]	Data[5]	Data[6]	Data[7]
0x64	0x09	0x00	0x00	0x01	0x00	0x00	0x00

6.2.2 Camera1 Diaphragm Control (not use)

CAN identifier	Parameter Name	Data Type	Units	Notes
1341 (0x53D)	Camera1 diaphragm control	CHAR	-	-

Command Byte

Parameter Name	Description
Diaphragm control	-1 – Rotate in one direction 0 – Stop +1 – Rotate in the other direction

6.2.3 Camera2 Focus Control (not use)

CAN identifier	Parameter Name	Data Type	Units	Notes
1350 (0x546)	Camera2 focus control	CHAR	-	-

6.2.4 Camera2 Diaphragm Control (not use)

CAN identifier	Parameter Name	Data Type	Units	Notes
1351 (0x547)	Camera2 diaphragm control	CHAR	-	-

6.2.5 Camera3 Focus Control (not use)

CAN identifier	Parameter Name	Data Type	Units	Notes
1360 (0x550)	Camera3 focus control	CHAR	-	-

6.2.6 Camera3 Diaphragm Control (not use)

CAN identifier	Parameter Name	Data Type	Units	Notes
1361 (0x551)	Camera3 diaphragm control	CHAR	-	-

6.2.7 Camera4 Focus Control (not use)

CAN identifier	Parameter Name	Data Type	Units	Notes
1370 (0x55A)	Camera4 focus control	CHAR	-	-

6.2.8 Camera4 Diaphragm Control (not use)

CAN identifier	Parameter Name	Data Type	Units	Notes
1371 (0x55B)	Camera4 diaphragm control	CHAR	-	-

6.3 Cover Commands

6.3.1 Cover Control

CAN identifier	Parameter Name	Data Type	Units	Notes
1400 (0x578)	Cover control	UCHAR	-	-

Command Byte

Parameter Name	Description
Cover control	0 – Close cover 1 – Open cover

Example: Open cover

Data[0]	Data[1]	Data[2]	Data[3]	Data[4]	Data[5]	Data[6]	Data[7]
0x64	0x0A	0x00	0x00	0x01	0x00	0x00	0x00

6.4 Fan Commands

6.4.1 Fan Control

CAN identifier	Parameter Name	Data Type	Units	Notes
1403 (0x57B)	Fan control	UCHAR	-	-

Command Byte

Parameter Name	Description
Fan control	0 – Fan off 1 – Fan on

6.5 Bolt Commands

6.5.1 Bolt Control (not use)

CAN identifier	Parameter Name	Data Type	Units	Notes
1401 (0x579)	Bolt control	UCHAR	-	-

Command Byte

Name	Description
Bolt control	1 – Cock the bolt

6.5.2 Bolt Stroke (not use)

CAN identifier	Parameter Name	Data Type	Units	Notes
1406 (0x57E)	Bolt stroke	LONG	counts	-

6.6 Fuse Commands

6.6.1 Fuse Control (not use)

CAN identifier	Parameter Name	Data Type	Units	Notes
1416 (0x57F)	Fuse control	UCHAR	-	-

Command Byte

Parameter Name	Description
Fuse control	0 – Unlock 1 – Lock

6.7 Software Trigger Commands

6.7.1 Trigger Fuse (not use)

CAN identifier	Parameter Name	Data Type	Units	Notes
1410 (0x582)	Trigger fuse	UCHAR	-	-

Command Byte

Parameter Name	Description
Trigger fuse	0 – Unlock 1 – Lock (default)

6.7.2 Trigger Control

CAN identifier	Parameter Name	Data Type	Units	Notes
1411 (0x583)	Trigger control	UCHAR	-	-

Command Byte

Parameter Name	Description
Trigger control	0 – No shooting 1 – Shooting

6.7.1 Trigger Heartbeat

CAN identifier	Parameter Name	Data Type	Units	Notes
1412 (0x584)	Trigger heartbeat	UCHAR	counts	Incremental

Note: Sent after the trigger is unlocked at a frequency ≥ 4 Hz.

6.8 Battery Commands

6.8.1 Charging Control (for GShG)

CAN identifier	Parameter Name	Data Type	Units	Notes
1414 (0x586)	Charging control	UCHAR	-	-

Command Byte

Parameter Name	Description
Charging control	0 – Charging off 1 – Charging on

6.9 Heater Commands

6.9.1 OES Heater Control

CAN identifier	Parameter Name	Data Type	Units	Notes
1430 (0x596)	OES heater control	UCHAR	-	-

Command Byte

Parameter Name	Description
OES heater control	0 – Heater off 1 – Heater on

6.9.2 Drives Heater Control (not use)

CAN identifier	Parameter Name	Data Type	Units	Notes
1432 (0x598)	Drives heater control	UCHAR	-	-

Command Byte

Parameter Name	Description
Drives heater control	0 – Heater off 1 – Heater on

7 Description of Data Identifiers

TIS (*Transmission Interval Service*) is used to set the time interval.

7.1 Turret Data

Node-ID = 1

7.1.1 Yaw Actual Position

CAN identifier	Parameter Name	Data Type	Units	Notes
1303 (0x517)	Yaw actual position	FLOAT	deg	-180 ÷ 180

7.1.2 Yaw Actual Velocity

CAN identifier	Parameter Name	Data Type	Units	Notes
1304 (0x518)	Yaw actual velocity	FLOAT	deg/s	-

7.1.3 Yaw Motor Temperature

CAN identifier	Parameter Name	Data Type	Units	Notes
1305 (0x519)	Yaw motor temperature	FLOAT	°C	-

7.1.4 Yaw Current Actuator Mode

CAN identifier	Parameter Name	Data Type	Units	Notes
1308 (0x51C)	Yaw current actuator mode	UCHAR	-	-

Parameter Name	Description
Actuator mode	0 – Free run 1 – Without brake 2 – Auto brake

7.1.5 Yaw Current Bus Voltage

CAN identifier	Parameter Name	Data Type	Units	Notes
1461(0x5B5)	Yaw current bus voltage	FLOAT	V	-

7.1.6 Yaw Actual Current

CAN identifier	Parameter Name	Data Type	Units	Notes
1462(0x5B6)	Yaw actual current	FLOAT	A	-

7.1.7 Yaw Current Humidity

CAN identifier	Parameter Name	Data Type	Units	Notes
1464(0x5B8)	Yaw current humidity	FLOAT	%	-

7.1.8 Yaw Power Stage Temperature

CAN identifier	Parameter Name	Data Type	Units	Notes
1465 (0x5B9)	Yaw power stage temperature	FLOAT	°C	-

7.1.9 Yaw Number Of Revolution

CAN identifier	Parameter Name	Data Type	Units	Notes
1466 (0x5BA)	Yaw number of revolution	LONG	-	-

7.1.10 Yaw Current Status

CAN identifier	Parameter Name	Data Type	Units	Notes
1467 (0x5BB)	Yaw current status	UCAR	-	-

Bit Number	Bit Name	Description
0-1	Target reached	0 – Not reached 1 – Reached 2 – Jog
2-3	Actuator mode	0 – Free run 1 – Without brake 2 – Auto brake
4-6	-	Reserved
7	Actuator state	0 – Disabled 1 – Enabled

7.1.11 Pitch Actual Position

CAN identifier	Parameter Name	Data Type	Units	Notes
1313 (0x521)	Pitch actual position	FLOAT	deg	-180 ÷ 180

7.1.12 Pitch Actual Velocity

CAN identifier	Parameter Name	Data Type	Units	Notes
1314 (0x522)	Pitch actual velocity	FLOAT	deg/s	-

7.1.13 Pitch Motor Temperature

CAN identifier	Parameter Name	Data Type	Units	Notes
1315 (0x523)	Pitch motor temperature	FLOAT	°C	-

7.1.14 Pitch Current Actuator Mode

CAN identifier	Parameter Name	Data Type	Units	Notes
1318 (0x526)	Pitch actuator mode	UCHAR	-	-

Parameter Name	Description
Actuator mode	0 – Free run 1 – Without brake 2 – Auto brake

7.1.15 Pitch Current Bus Voltage

CAN identifier	Parameter Name	Data Type	Units	Notes
1481(0x5C9)	Pitch current bus voltage	FLOAT	V	-

7.1.16 Pitch Actual Current

CAN identifier	Parameter Name	Data Type	Units	Notes
1482(0x5CA)	Pitch actual current	FLOAT	A	-

7.1.17 Pitch Current Humidity

CAN identifier	Parameter Name	Data Type	Units	Notes
1484(0x5CC)	Pitch current humidity	FLOAT	%	-

7.1.18 Pitch Power Stage Temperature

CAN identifier	Parameter Name	Data Type	Units	Notes
1485 (0x5CD)	Pitch Power Stage Temperature	FLOAT	°C	-

7.1.19 Pitch Number of Revolution

CAN identifier	Parameter Name	Data Type	Units	Notes
1486 (0x5CE)	Pitch number of revolution	LONG	-	-

7.1.20 Pitch Current Status

CAN identifier	Parameter Name	Data Type	Units	Notes
1487 (0x5CF)	Pitch current status	UCAR	-	-

Bit Number	Bit Name	Description
0-1	Target reached	0 – Not reached 1 – Reached 2 – Jog
2-3	Actuator mode	0 – Free run 1 – Without brake 2 – Auto brake
4-6	-	Reserved
7	Actuator state	0 – Disabled 1 – Enabled

7.1.21 Bolt State (not use)

CAN identifier	Parameter Name	Data Type	Units	Notes
1402 (0x57A)	Bolt state	UCHAR	-	-

Parameter Name	Description
Bolt state	0 – Bolt extended 1 – Bolt retracted

7.1.22 Fan State

CAN identifier	Parameter Name	Data Type	Units	Notes
1404 (0x57C)	Fan state	UCHAR	-	-

Parameter Name	Description
Fun state	0 – Disabled 1 – Enabled

7.1.23 Case Temperature

CAN identifier	Parameter Name	Data Type	Units	Notes
1405 (0x57D)	Case temperature	FLOAT	°C	-

7.1.24 Charging current

CAN identifier	Parameter Name	Data Type	Units	Notes
1413 (0x585)	Charging current	FLOAT	A	-

7.1.25 Charging State

CAN identifier	Parameter Name	Data Type	Units	Notes
1415 (0x587)	Charging state	UCHAR	-	-

Parameter Name	Description
Charging state	0 – Disabled 1 – Enabled

7.1.26 Fuse State

CAN identifier	Parameter Name	Data Type	Units	Notes
1417 (0x589)	Fuse state	UCHAR	-	-

Parameter Name	Description
Fuse state	0 – Unlocked 1 – Locked

7.1.27 Global Shot Counter

CAN identifier	Parameter Name	Data Type	Units	Notes
1418 (0x589)	Global shot counter	ULONG	counts	-

7.1.28 OES Heater State

CAN identifier	Parameter Name	Data Type	Units	Notes
1431 (0x597)	OES heater state	UCHAR	-	-

Bit Number	Bit Name	Description
0	OES heater state	0 – Disabled 1 – Enabled
1	OES heater current	0 – No current 1 – Current flows

7.1.29 Drives Heater State (not use)

CAN identifier	Parameter Name	Data Type	Units	Notes
1433 (0x599)	Drives heater state	UCHAR	-	-

Parameter Name	Description
Drives heater state	0 – Disabled 1 – Enabled

7.1.30 Periphery Status

CAN identifier	Parameter Name	Data Type	Units	Notes
1435 (0x59B)	Periphery status	ULONG	-	-

Bit Number	Bit Name	Description
0	Fan state	0 – Disabled 1 – Enabled
1	OES heater state	0 – Disabled 1 – Enabled
2	OES heater current	0 – No current 1 – Current flows
3	Drives heater state	0 – Disabled 1 – Enabled
4	Free run button state	0 – Released 1 – Pressed
5	Bolt state	0 – Bolt extended 1 – Bolt retracted
6	Fuse state	0 – Unlocked 1 – Locked
7	Charging state	0 – Disabled 1 – Enabled
8	Case temperature sensor state	0 – Ok 1 – Fault
9-31	-	Reserved

7.1.31 Global Pitch Position (Inclinometer)

CAN identifier	Parameter Name	Data Type	Units	Notes
1801 (0x709)	Global pitch position	FLOAT	deg	-

7.1.32 Global Roll Position (Inclinometer)

CAN identifier	Parameter Name	Data Type	Units	Notes
1802 (0x70A)	Global roll position	FLOAT	deg	-

7.2 Cover Data

Node-ID = 2

7.2.1 Cover State (always sent by default at 100ms interval)

CAN identifier	Parameter Name	Data Type	Units	Notes
1409 (0x581)	Cover state	UCHAR	-	-

Parameter Name	Description
Cover state	0 – Closed 1 – Opened 2 – Middle

7.3 Rover GNSS Data

Node-ID = 5

7.3.1 Current Heading

CAN identifier	Parameter Name	Data Type	Units	Notes
1507 (0x5E3)	Current heading	LONG	deg*1e-5	-

7.3.2 Current Accuracy

CAN identifier	Parameter Name	Data Type	Units	Notes
1509 (0x5E5)	Current accuracy	LONG	deg*1e-5	-

7.3.3 Global Yaw Position (heading)

CAN identifier	Parameter Name	Data Type	Units	Notes
1800 (0x708)	Global yaw position	FLOAT	deg	-

7.4 Base GNSS Data

Node-ID = 6

7.4.1 Latitude

CAN identifier	Parameter Name	Data Type	Units	Notes
1520 (0x5F0)	Latitude	LONG	deg*1e-7	-

7.4.2 Longitude

CAN identifier	Parameter Name	Data Type	Units	Notes
1521 (0x5F1)	Longitude	LONG	deg*1e-7	-

7.4.3 Altitude Mean Sea Level

CAN identifier	Parameter Name	Data Type	Units	Notes
1523 (0x5F3)	Altitude mean sea level	LONG	mm	-

7.4.4 Position Accuracy Estimate

CAN identifier	Parameter Name	Data Type	Units	Notes
1524 (0x5F4)	Position accuracy estimate	ULONG	m*1e-4	-

7.4.5 UTC Time

CAN identifier	Parameter Name	Data Type	Units	Notes
1528 (0x5F8)	UTC time	ULONG	s	-

7.5 Zoom Data

Node-ID = 7

7.5.1 Camera0 Zoom Position

CAN identifier	Parameter Name	Data Type	Units	Notes
1342 (0x53E)	Camera0 zoom position	LONG	count	-