maxon

UAV-ESC

Communication Guide

UAV-ESC Electronic Speed Controller
Communication Guide
CCMC | Edition 2022-11 | DocID rel_10886

maxon

READ THIS FIRST

THESE INSTRUCTIONS ARE INTENDED FOR QUALIFIED TECHNICAL PERSONNEL.

IMPORTANT NOTICE: PREREQUISITES FOR PERMISSION TO COMMENCE COMMISSIONING
The UAV-ESC is considered as partly completed machinery according to EU Directive 2006/42/EC, Article 2, Clause (g) and **is intended to be incorporated into or assembled with other machinery or other partly completed machinery or equipment**.

WARNING
RISK OF INJURY
OPERATING THE DEVICE WITHOUT THE FULL COMPLIANCE OF THE SURROUNDING SYSTEM WITH THE EU DIRECTIVE 2006/42/EC MAY CAUSE SERIOUS INJURIES!
- Do not operate the device, unless you have made completely sure that the other machinery fully complies with the EU directive's requirements!
- Do not operate the device, unless the other machinery fulfills all relevant health and safety aspects!
- Do not operate the device, unless all respective interfaces have been established and fulfill the requirements stated in this document!

PRIOR COMMENCING WITH ANY ACTIVITIES YOU MUST CAREFULLY READ AND UNDERSTAND THIS MANUAL AND YOU MUST FOLLOW THE INSTRUCTIONS GIVEN THEREIN.

YOU MUST NOT PUT THE DEVICE INTO SERVICE UNLESS YOU HAVE MADE COMPLETELY SURE ABOUT THE FOLLOWING:
- You must make sure that the surrounding system with all involved components (such as motor, propeller, flight controller, other connected electronics or devices) does fully comply with any applicable law as well as local rules and regulations!
- You must make sure that the surrounding system does fulfill all relevant health and safety aspects!
- You must make sure that all respective interfaces have been correctly established and that they fulfill the herein stated requirements!

OBSERVE THE FOLLOWING BEFORE TAKE-OFF AND KEEP IN MIND DURING THE ENTIRE FLIGHT:
- Check on applicable local rules and regulations in respect to flight permissions, no-fly zones, restricted areas, and other flight restrictions.
- Check for airworthiness and full operational condition of your aircraft.
- Check all components for tight fit before every flight. Make sure that all motors, propellers, and other parts are installed correctly. Do not attempt to fly the aircraft with worn or damaged components.
- DO NOT approach or touch the aircraft while the motors or propellers are running or while the aircraft is powered.

USE THIS PRODUCT STRICTLY ACCORDING TO THE INFORMATION GIVEN IN THE PRESENT DOCUMENT.
MAXON DOES NOT ASSUME ANY LIABILITY FOR LOSS, DAMAGE, CLAIMS, OR COSTS OF ANY KIND (INCLUDING, BUT NOT EXCLUDED TO) CONSEQUENTIAL, DIRECT, INDIRECT, OR INCIDENTAL DAMAGES, LOST PROFITS OR LOST SAVINGS, PERSONAL INJURY OR LACK OF CARE, OR CLAIMS OF THIRD PARTIES, EVEN IF MAXON WAS INFORMED OF SUCH LOSS, DAMAGE, CLAIMS, OR COSTS THAT MAY DIRECTLY OR INDIRECTLY ARISE FROM USING THIS PRODUCT IN AN IMPROPER OR UNSUITABLE WAY.

TABLE OF CONTENTS

maxon

1 ABOUT

The present document provides you with information on the UAV-ESC communication interfaces.

Find the latest edition on the present document as well as additional documentation and software for UAV-ESC unmanned aerial vehicle electronic speed controllers on the Internet
➔https://uav.maxongroup.com

1.1 About this Document

1.1.1 Intended Purpose

The purpose of the present document is to familiarize you with the UAV-ESC Electronic Speed Controller. It will highlight the tasks for safe and adequate installation and commissioning. Follow the described instructions…

- to avoid dangerous situations,
- to keep installation and/or commissioning time at a minimum,
- to increase reliability and service life of the described equipment

The present document is part of a documentation set and contains performance data and specifications, information on fulfilled standards, details on connections and pin assignment, and wiring examples. The below overview shows the documentation hierarchy and the interrelationship of its individual parts:

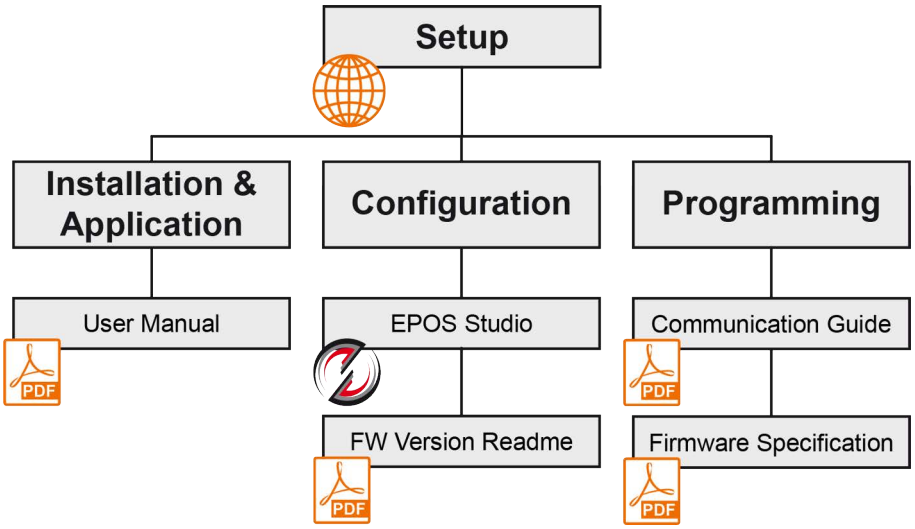

Figure 1-1 Documentation structure

1.1.2 Target Audience

The present document is intended for trained and skilled personnel. It conveys information on how to understand and fulfill the respective work and duties.

The present document is a reference book. It does require particular knowledge and expertise specific to the equipment described.

1.1.3 How to use

Throughout the document, the following notations and codes will be used.

Notation	Meaning
UAV-ESC	stands for all supported sensorless controller types of the UAV-ESC family
«Abcd»	indicating a title or a name (such as of document, product, mode, etc.)
(n)	refers to an item (such as part numbers, list items, etc.)
➔	denotes “see”, “see also”, “take note of” or “go to”

Table 1-1 Notation used

1.1.4 Trademarks & Brand Names

For easier legibility, registered brand names are listed below and will not be further tagged with their respective trademark. It must be understood that the brands (the list below is not necessarily concluding) are protected by copyright and/or other intellectual property rights even if their legal trademarks are omitted in the later course of this document.

Brand name	Trademark owner
CANopen® CiA®	© CiA CAN in Automation e.v, DE-Nuernberg
DroneCAN®	by open source community «DroneCAN Development Team»

Table 1-2 Brand names and trademark owners

1.1.5 Sources for additional Information

For further details and additional information, please refer to below listed sources:

Item	Reference
[1]	CAN in Automation's CAN Specification 2.0 www.can-cia.org
[2]	CiA 301 CANopen application layer and communication profile www.can-cia.org
[3]	USB Implementers Forum: Universal Serial Bus Revision 2.0 Specification: www.usb.org/developers/docs/usb20_docs/
[x]	DroneCAN protocol https://dronecan.github.io/

Table 1-3 Sources for additional information

Find the latest edition of the present document and other information here:
➔https://uav.maxongroup.com/

If you should encounter any problems or if you have any questions, feel free to contact the maxon Support Center:
➔https://support.maxongroup.com/hc/en-us/

1.1.6 Copyright

CCMC | UAV-ESC Communication Guide | Edition 2022-11 | DocID rel_10886

1.2 About the Device

maxon's «UAV-ESC Electronic Speed Controller» are high-performance, sensorless, small-sized, fully digital, smart control units. They feature field oriented control (FOC) for brushless EC (BLDC) motors and drive systems without Hall sensors.

maxon

••page intentionally left blank••

2 USB COMMUNICATION

2.1 UAV-ESC USB Command Reference

2.1.1 Read Functions

2.1.1.1 ReadObject

Read an object value from the Object Dictionary at the given Index and Subindex.

Request Frame		
OpCode	BYTE	0x60
Len	BYTE	2 (number of words)
Parameters	BYTE	Node-ID
	WORD	Index of Object
	BYTE	Subindex of Object

Response Frame		
OpCode	BYTE	0x00
Len	BYTE	4 (number of words)
Parameters	DWORD	➔"Communication Error Code Definition" on page 4-27
	BYTE [4]	Data Bytes Read

2.1.1.2 InitiateSegmentedRead

Start reading an object value from the Object Dictionary at the given Index and Subindex.

Request Frame		
OpCode	BYTE	0x81
Len	BYTE	2 (number of words)
Parameters	BYTE	Node-ID
	WORD	Index of Object
	BYTE	Subindex of Object

Response Frame		
OpCode	BYTE	0x00
Len	BYTE	5…132 (number of words)
Parameters	DWORD	➔"Communication Error Code Definition" on page 4-27
	DWORD	Object Data Length (total number of bytes)
	BYTE	Length (max. 255 bytes)
	BYTE [0…254]	Data Bytes Read

2.1.1.3 SegmentRead

Read a data segment of the object initiated with the command ➔«InitiateSegmentedRead».

Request Frame				
OpCode	BYTE		0x62	
Len	BYTE		1 (number of words)	
Parameters	BYTE	[Bit 0] [Bit 1…7]	ControlByte	Toggle Bit Not used
	BYTE		Dummy Byte without meaning	

Response Frame				
OpCode	BYTE		0x00	
Len	BYTE		3…131 (number of words)	
Parameters	DWORD		➔“Communication Error Code Definition” on page 4-27	
	BYTE		Length (max. 255 bytes)	
	BYTE	[Bit 0] [Bit 1] [Bit 2…7]	ControlByte	Toggle Bit Last Data Segment Not used
	BYTE [0…254]		Data Bytes Read	
	BYTE		Dummy Byte when length odd	

2.1.2 Write Functions

2.1.2.1 WriteObject

Write an object value to the Object Dictionary at the given Index and Subindex.

Request Frame		
OpCode	BYTE	0x68
Len	BYTE	4 (number of words)
Parameters	BYTE	Node-ID
	WORD	Index of Object
	BYTE	Subindex of Object
	BYTE [4]	Data Bytes to write

Response Frame		
OpCode	BYTE	0x00
Len	BYTE	2 (number of words)
Parameters	DWORD	➔“Communication Error Code Definition” on page 4-27

2.1.2.2 InitiateSegmentedWrite

Start writing an object value to the Object Dictionary at the given Index and Subindex. Use the command ➔«SegmentWrite» to write the data.

Note that gateway communication is not supported.

Request Frame		
OpCode	BYTE	0x69
Len	BYTE	4 (number of words)
Parameters	BYTE	Node-ID
	WORD	Index of Object
	BYTE	Subindex of Object
	DWORD	Object Data Length (total number of bytes)

Response Frame		
OpCode	BYTE	0x00
Len	BYTE	2 (number of words)
Parameters	DWORD	➔“Communication Error Code Definition” on page 4-27

2.1.2.3 SegmentWrite

Write a data segment to the object initiated with the command ➔«InitiateSegmentedWrite».

Note that gateway communication is not supported.

Request Frame				
OpCode	BYTE		0x6A	
Len	BYTE		1…129 (number of words)	
Parameters	BYTE		Length (max. 255 bytes)	
	BYTE	[Bit 0] [Bit 1] [Bit 2…7]	ControlByte	Toggle Bit Last Data Segment Not used
	BYTE [0…254]		Data Bytes to write	
	BYTE		Dummy Byte when length odd	

Response Frame				
OpCode	BYTE		0x00	
Len	BYTE		3 (number of words)	
Parameters	DWORD		➔“Communication Error Code Definition” on page 4-27	
	BYTE		Length written (max. 255 bytes)	
	BYTE	[Bit 0] [Bit 1…7]	ControlByte	Toggle Bit Not used

2.2 Data Link Layer

2.2.1 Flow Control

The UAV-ESC always communicates as a slave.

A frame is only sent as an answer to a request. All commands send an answer. The master must always initiate communication by sending a packet structure.

The data flow while transmitting and receiving frames are as follows:

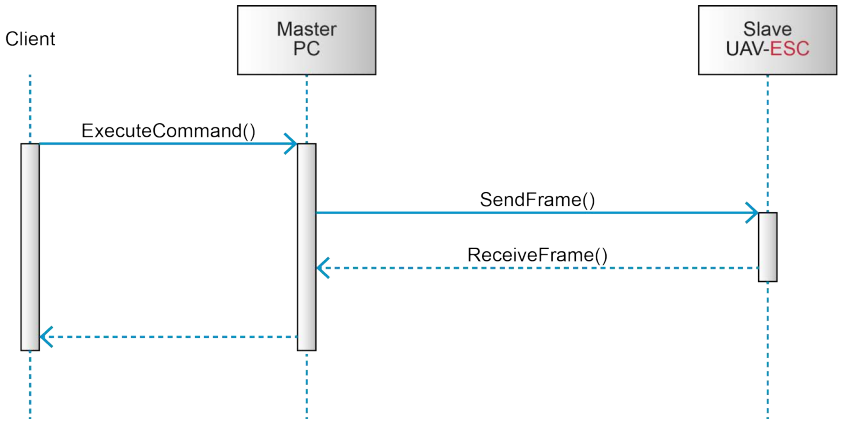

Figure 2-2 USB communication – Commands

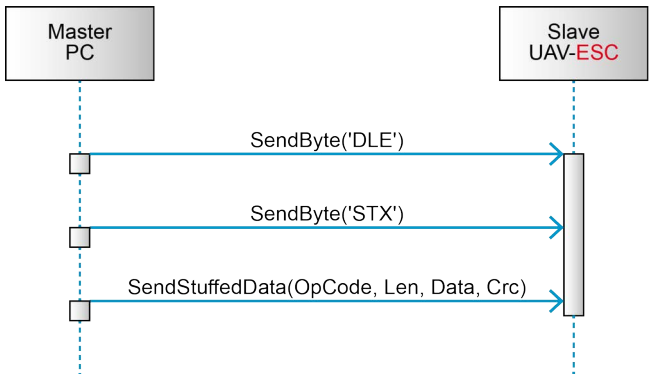

Figure 2-3 USB communication – Sending a data frame to UAV-ESC

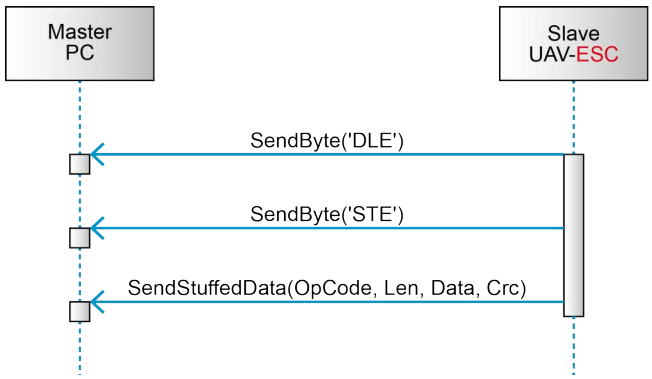

Figure 2-4 USB communication – Receiving a response data frame from UAV-ESC

2.2.2 Frame Structure

The data bytes are sequentially transmitted in frames. A frame composes of…

- synchronization (and byte stuffing),
- header,
- variably long data field, and
- 16-bit long cyclic redundancy check (CRC) for verification of data integrity.

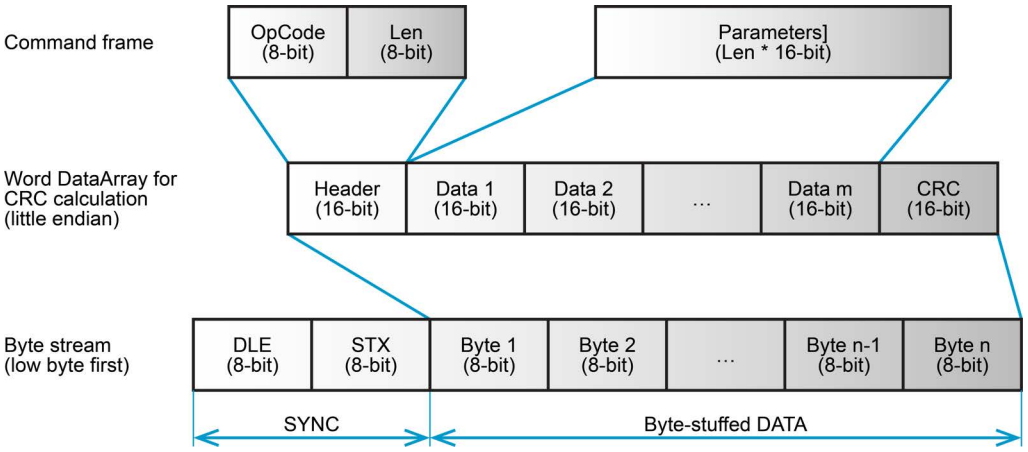

Figure 2-5 USB communication – Frame structure

SYNC The first two bytes are used for frame synchronization.

- **DLE** Starting frame character “DLE” (Data Link Escape) = 0x90
- **STX** Starting frame character “STX” (Start of Text) = 0x02

HEADER The header consists of 2 bytes. The first field determines the type of data frame to be sent or received. The next field contains the length of the data fields.

- **OpCode** Operation command to be sent to the slave. For details on the command set ➔“UAV-ESC USB Command Reference” on page 2-9.
- **Len** Represents the number of words (16-bit value) in the data fields [0…143].

DATA The data fields contain the parameters of the message. The low byte of the word is transmitted first.

- **Data[i]** The parameter word of the command. The low byte is transmitted first.
- **CRC** 16-bit long cyclic redundancy check (CRC) for verification of data integrity.

Note
As a reaction to a bad OpCode or CRC value, the slave sends a frame containing the corresponding error code.

For an example on composition and structure of UAV-ESC messages ➔“Example: Command Instruction” on page 2-17.

2.2.3 Cyclic Redundancy Check (CRC)

CRC is used for verification of data integrity.

2.2.3.1 CRC Calculation

Note

- *The 16-bit CRC checksum uses the algorithm CRC-CCITT.*
- *For calculation, the 16-bit generator polynomial "$x^{16}+x^{12}+x^5+x^0$" is used.*
- *The CRC is calculated before data stuffing and synchronization.*
- *Add a CRC value of "0" (zero) for CRC calculation.*
- *The data frame bytes must be calculated as a word.*

2.2.3.2 CRC Algorithm

ArrayLength: Len + 2	WORD DataArray[ArrayLength]
Generator Polynom G(x):	10001000000100001 (= $x^{16}+x^{12}+x^5+x^0$)
DataArray[0]: DataArray[1]: DataArray[2]: … DataArray[ArrayLength-1]:	HighByte(Len) + LowByte(**OpCode**) Data[0] Data[1] … 0x0000 (CrcValue)

```
WORD CalcFieldCRC(WORD* pDataArray, WORD ArrayLength)
{
   WORD shifter, c;
   WORD carry;
   WORD CRC = 0;

   //Calculate pDataArray Word by Word
   while(ArrayLength--)
   {
      shifter = 0x8000;            //Initialize BitX to Bit15
      c = *pDataArray++;           //Copy next DataWord to c
      do
      {
         carry = CRC & 0x8000;     //Check if Bit15 of CRC is set
         CRC <<= 1;                //CRC = CRC * 2
         if(c & shifter) CRC++;    //CRC = CRC + 1, if BitX is set in c
         if(carry) CRC ^= 0x1021;  //CRC = CRC XOR G(x), if carry is true
         shifter >>= 1;            //Set BitX to next lower Bit, shifter = shifter/2
      } while(shifter);
   }
   return CRC
}
```

Figure 2-6 USB communication – CRC algorithm

2.2.4 Byte Stuffing

The sequence “DLE” and “STX” are reserved for frame start synchronization. If the character “DLE” appears at a position between “OpCode” and “CRC” and is not a starting character, the byte must be doubled (byte stuffing). Otherwise, the protocol begins to synchronize for a new frame. The character “STX” needs not to be doubled.

EXAMPLES:

Sending Data	0x21, **0x90**, 0x45
Stuffed Data	0x21, **0x90**, **0x90**, 0x45

Sending Data	0x21, **0x90, 0x02,** 0x45
Stuffed Data	0x21, **0x90**, **0x90**, **0x02**, 0x45

Sending Data	0x21, **0x90**, **0x90**, 0x45
Stuffed Data	0x21, **0x90**, **0x90**, **0x90**, **0x90**, 0x45

Important:
Byte stuffing is used for all bytes (CRC included) in the frame except the starting characters.

2.2.5 Transmission Byte Order

To send and receive a word (16-bit) via the serial port, the low byte will be transmitted first.

Multiple byte data (word = 2 bytes, long word = 4 bytes) are transmitted starting with the less significant byte (LSB) first.

A word will be transmitted in this order: byte0 (LSB), byte1 (MSB).

A long word will be transmitted in this order: byte0 (LSB), byte1, byte2, byte3 (MSB).

2.2.6 Timeout Handling

The timeout is handled over a complete frame. Hence, the timeout is evaluated over the sent data frame, the command processing procedure and the response data frame. For each frame (frames, data processing), the timer is reset and timeout handling will recommence.

Object	Index	Subindex	Default
USB Frame Timeout	0x2006	0x00	500 [ms]

Table 2-4 USB communication – Timeout handling

Note
To cover special requirements, the timeout may be changed by writing to the Object Dictionary!

2.2.7 Slave State Machine

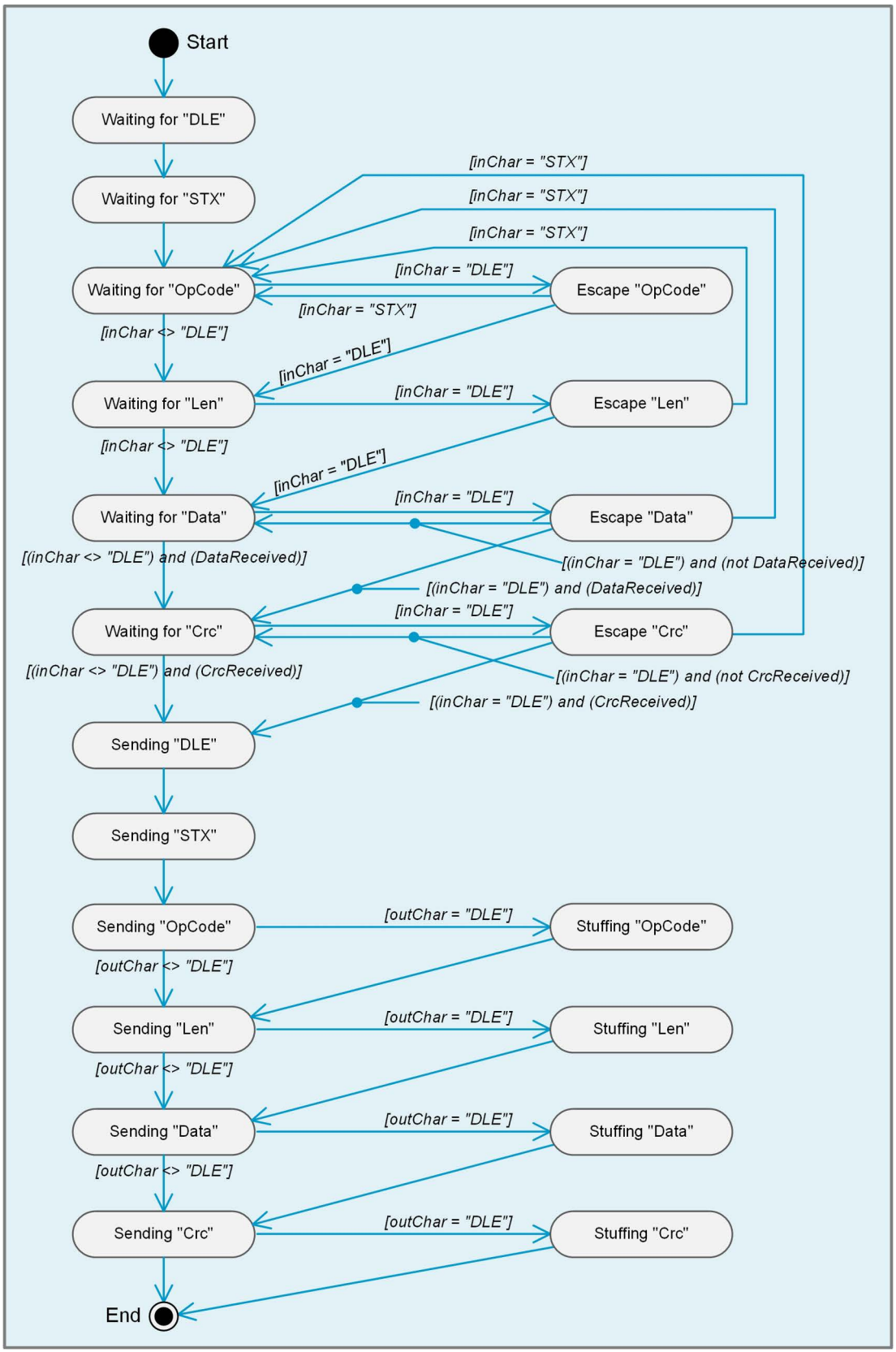

Figure 2-7 USB communication – Slave State Machine

2.2.8 Example: Command Instruction

The following example demonstrated composition and structure of the UAV-ESC messages during transmission and reception via USB.

The command sent to the UAV-ESC is “ReadObject”, it can be used to read an object with up to 4 bytes.

ReadObject “Home Position” (Index = 0x30B0, Subindex = 0x00) from Node-ID 1

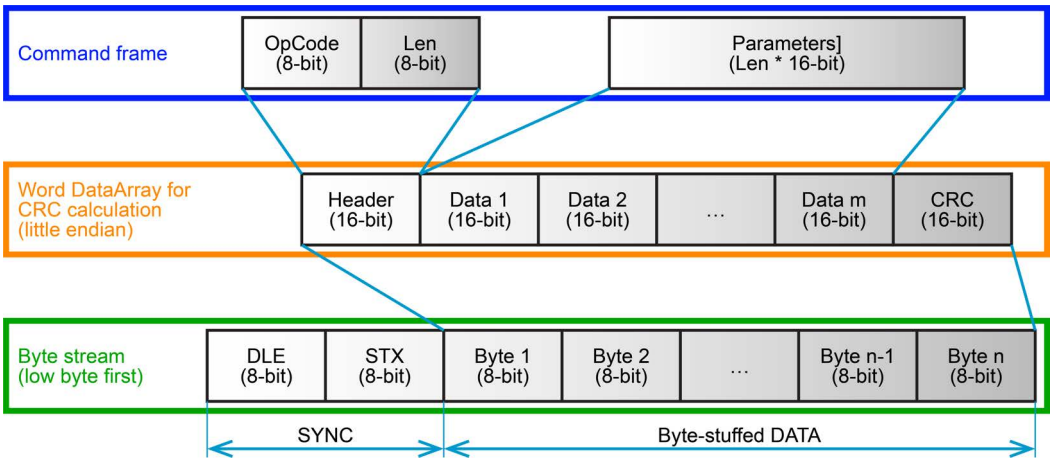

Figure 2-8 USB communication – Command instruction (example)

A) SETUP

1) Setup the desired frame (for details ➔“UAV-ESC USB Command Reference” on page 2-9).

Request frame			
OpCode	BYTE	Read object	0x60
Len	BYTE	Number of words	0x02
Parameters	BYTE	Node-ID	0x01
	WORD	Index of object	0x30B0
	BYTE	Subindex of object	0x00

B) CRC CALCULATION

For details ➔“Cyclic Redundancy Check (CRC)” on page 2-14):

2) Prepare the word DataArray for CRC calculation (little endian):

DataArray	
DataArray[0]	0x0260
DataArray[1]	0xB001
DataArray[2]	0x0030
DataArray[3]	**0x0000**

→use CRC value of “0” (zero)

Important:

- *Make sure that the CRC is calculated correctly. If the CRC is not correct, the command will neither be accepted nor processed.*
- *CRC calculation includes all bytes of the data frame except synchronization bytes and byte stuffing.*
- *The data frame bytes must be calculated as a word.*
- *For calculation, use a CRC value of “0” (zero).*

3) Calculate the CRC (use algorithm as to ➔“CRC Algorithm” on page 2-14):
ArrayLength = Len + 2

CrcValue = CalcFieldCRC(&DataArray, ArrayLength)
DataArray[ArrayLength-1] = CrcValue

4) Add the CRC value to the DataArray:

DataArray	
DataArray[0]	0x0260
DataArray[1]	0xB001
DataArray[2]	0x0030
DataArray[3]	**0x622E**

→the calculated CRC value

C) COMPLETION

5) Pack the DataArray to a byte stream (low byte first).
6) Add sync bytes.
7) Add byte stuffing (➔"Byte Stuffing" on page 2-15).
8) Transmit the stuffed byte stream (➔"Transmission Byte Order" on page 2-15):
SendStuffedData(&DataArray)
Transmission order (low byte first):
0x90,**0x02**,0x60,0x02,0x01,0xB0,0x30,0x00,0x2E,0x62

D) WAIT FOR RECEIVE FRAME

9) The UAV-ESC will answer to the command "ReadObject" with an answer frame and the returned parameters in the data block as follows:
Reception order (low byte first):
0x90,**0x02**,0x00,0x04,0x00,0x00,0x00,0x00,0x01,0x90,0x90,0x00,0x00,0x9A,0x5C

Important:
- *Do not send any data before the receive frame or a timeout is present.*
- *UAV-ESC cannot process data simultaneously.*

E) REMOVE BYTE STUFFING AND THE SYNCHRONIZATION ELEMENTS

10) Byte stream without stuffing and synchronization elements:
0x00,0x04,0x00,0x00,0x00,0x00,0x01,0x90,0x00,0x00,0x9A,0x5C

F) CRC CHECK

For details ➔chapter "2.2.3 Cyclic Redundancy Check (CRC)" on page 2-14):

11) Prepare the word DataArray for CRC calculation (little endian):

DataArray	
DataArray[0]	0x0400
DataArray[1]	0x0000
DataArray[2]	0x0000
DataArray[3]	0x9001
DataArray[4]	0x0000
DataArray[5]	0x5C9A

12) Calculate the CRC (use algorithm as to ➔"CRC Algorithm" on page 2-14). Thereby, valid value for CRC is "0" (zero):

```
ArrayLength= Len + 2
CrcValue = CalcFieldCRC(&DataArray, ArrayLength)
Valid = (0x0000 == CrcValue)
```

G) CHECK

13) Check the UAV-ESC receive frame.

Response frame				
OpCode	BYTE	Read object	0x00	
Len	BYTE	Number of words	0x04	
Parameters	BYTE	Node-ID	0x01	
	DWORD	Communication error	0x00000000	→no error
	DWORD	Data bytes read	0x00009001	→36'865 inc

Important:

- *If the error code is unequal to "0" (zero), the command was not processed!*
- *Check ➔"Communication Error Code Definition" on page 4-27 for error details.*
- *Fix the error before attempting to resend the data frame.*

2.3 Physical Layer

2.3.1 USB

ELECTRICAL STANDARD

The UAV-ESC's USB interface follows the «Universal Serial Bus Specification Revision 2.0». You may wish to download the file from the Internet ➔www.usb.org/developers/docs/usb20_docs/.

3 CAN COMMUNICATION

3.1 General Information

maxon UAV-ESC drives' CAN interface follows the DroneCAN specifications. Only the messages ot the DroneCAN V1 (15 Feb 2022) standard data types are supported which are relevant for a ESC device:

- uavcan.equipment.esc
- uavcan.protocol.GetNodeInfo
- uavcan.protocol.NodeStatus
- uavcan.protocol.RestartNode
- uavcan.protocol.param

3.1.1 Documentation

For further information on DroneCAN as well as respective specifications references are listed in ➔"Sources for additional Information" on page 1-6.

3.2 DroneCAN messages

3.2.1 Process data channel

Index	Message	Field	Data type	Value	CANopen Indices
1030 (Rx)	uavcan.equipment.esc.RawCommand	cmd	int14[<=20]	duty cycle (velocity)	0x30F0-01
1031 (Rx)	uavcan.equipment.esc.RPMCommand	cmd	int14[<=20]	Rpm value (velocity)	0x30F0-05
1034 (Tx)	uavcan.equipment.esc.Status	error count	uint32	Error counter. Resets when the UAV restarts	N/A
		voltage	float16	power supply voltage	0x2200-01
		current	float16	measured average motor current	0x30D1-01
		temperature	int18	Power stage temperature in Kelvin	0x3201-01
		rpm	uint7	actual velocity	0x606C-00
		power_rating_pct	uint5	PWM duty cycle actual value (range 0-95)	0x3203-01
		esc_index	uint5	unique ESC index	0x2020-00
341 (Tx) [a]	uavcan.protocol.NodeStatus	uptime_sec	uint32	time since boot (seconds)	N/A
		health	uint2	node health	0x3244-04
		mode	uint3	current mode	0x3244-05
		sub_mode	uint3	0 [1]	N/A
		vendor_specific_status_code	uint16	Statusword bits (see Table 2-6)	0x6041-00

Table 3-5 Object data channel

[a] Currently marked as unused in the DroneCAN standard.

3.2.2 Service data channel

Index	Message	Field	Data type	Value	CANopen Indices
1 (Rx)	uavcan.protocol.GetNoteInfo	N/A (request)	N/A	N/A	N/A
1 (Tx)	uavcan.protocol.GetNodeInfo	status	NodeStatus		
		software_version.major	uint8	Revision number (high byte of high word)	0x1018-03
		software_version.minor	uint8	Revision number (low byte of high word)	0x1018-03
		software_version.optional_field_flags	uint8	2 (CRC)	N/A
		software_version.vcs_commit	uint32	0 (unused)	N/A

Continued on next page.

Index	Message	Field	Data type	Value	CANopen Indices
		software_version.image_crc	uint64	Application CRC[1] 32-bit CRC is stored in low dword of the field.	N/A
		hardware_version.major	uint8	Product code (high byte of high word)	0x1018-02
		hardware_version.minor	uint8	Product code (low byte of high word)	0x1018-02
		hardware_version.unique_id	uint8 [16]	Bytes 1..8: Internal serial number complete Bytes 9..12: Vendor Id Bytes 13..16: 0 (unused)	0x2100-01 0x1018-01
		hardware_version.certificate_of_authenticity	uint8 [≤255]	No data (length set to 0)	N/A
		name	uint8 [≤80]	com.maxon.uav.uav-esc	N/A
10 (Rx)	uavcan.protocol.param.ExecuteOpcode	opcode	uint8	0 = save 1 = erase	0x1010-1 0x1011-1
		argument	int48	Reserved, set to 0.	N/A
10 (Tx)	uavcan.protocol.param.ExecuteOpcode	argument	int48	Populated with abort code from CIL access.	N/A
		ok	bool	True if operation was performed successfully, false otherwise.	N/A
5 (Rx)	uavcan.protocol.RestartNode	magic_number	uint40	0xACCE551B1E	N/A (reboots the board)
5 (Tx)	uavcan.protocol.RestartNode	ok	bool	False if unsuccessful. If successful, no response to the RestartNode request is transmitted as the board has rebooted.	
11 (Rx)	uavcan.protocol.param.GetSet	Index	uint13	Index of parameter to retrieve. Only used in name field is empty.	N/A
		value	Value (union type of integer, float, bool or string)	The value to set the identified parameter to. If empty, request is treated as a get. If populated, request is treated as a set.	
		name	uint8 [<=92]	Name of the parameter to retrieve.	N/A

Continued on next page.

Index	Message	Field	Data type	Value	CANopen Indices
11 (Tx)	uavcan.protocol.param.GetSet	void5	void5	Padding	N/A
		value	Value (union type of integer, float, bool or string)	Value of requested parameter.	
		void5	void5	Padding	N/A
		default_value	Value (union type of integer, float, bool or string)	Default value if reset by uavcan.protocol.param.ExecuteOpcode	
		void6	void6	Padding	N/A
		max_value	NumericValue (union of integer or float)	Maximum value of the parameter (if numeric, otherwise empty).	
		void6	void6	Padding	N/A
		min_value	NumericValue (union of integer or float)	Minimum value of the parameter (if numeric, otherwise empty).	
		name	uint8[<=92]	Name of the parameter retrieved or set. Empty indicates no parameter found.	N/A

Table 3-6 Service data channel | DroneCAN messages for SDO

3.2.3 Parameters

The following table shows the details of the parameters supported. All parameters are int64 type of value field. Parameters are accessed via uavcan.protocol.param.GetSet.

Parameter	OBD Index	OBD Description
uavcan_node_id	0x2000-00	➔Node-ID
uavcan_bit_rate	0x2001-00	➔CAN bit rate
uavcan_esc_index	0x2020-00	➔ESC index
uavcan_node_stat_id	0x2021-01	Node status configuration ➔Node status data type ID
uavcan_node_stat_rate	0x2021-02	Node status configuration ➔Node status rate
uavcan_esc_stat_id	0x2022-01	ESC status configuration ➔ESC status data type ID
uavcan_esc_stat_rate	0x2022-02	ESC status configuration ➔ESC status rate
uavcan_esc_rawcmd_id	0x2024-01	Raw command configuration ➔Raw command data type ID
uavcan_esc_rawcmd_rate	0x2024-02	Raw command configuration ➔Raw command minimum rate
uavcan_esc_rpmcmd_id	0x2025-01	RPM command configuration ➔RPM command data type ID
uavcan_esc_rpmcmd_rate	0x2025-02	RPM command configuration ➔RPM command minimum rate
ctl_cur_p_gain	0x30A0-01	Velocity control parameter set ➔Current controller P gain [1]
ctl_cur_i_gain	0x30A0-02	Velocity control parameter set ➔Current controller I gain [1]
ctl_vel_p_gain	0x30A2-01	Velocity control parameter set ➔Velocity controller P gain [1]

Continued on next page.

Parameter	OBD Index	OBD Description
ctl_vel_i_gain	0x30A2-02	Velocity control parameter set ➔Velocity controller I gain [1]
ctl_vel_ff_vel_gain	0x30A2-03	Velocity control parameter set ➔Velocity controller FF velocity gain [1]
ctl_vel_ff_acc_gain	0x30A2-04	Velocity control parameter set ➔Velocity controller FF acceleration gain [1]
bemf_cur_g1	0x30A5-01	Back EMF observer parameter set ➔Kalman current gain [1]
bemf_bemf_g1	0x30A5-02	Back EMF observer parameter set ➔Kalman Back EMF gain [1]
bemf_cur_g2	0x30A5-03	Back EMF observer parameter set ➔Kalman current gain [2]
bemf_bemf_g2	0x30A5-04	Back EMF observer parameter set ➔Kalman Back EMF gain [2]
syn_st_ang_acc	0x30AF-01	Synchronous startup parameter set ➔Synchronous angular acceleration
syn_st_acc_cur_ratio	0x30AF-02	Synchronous startup parameter set ➔Synchronous acceleration current ratio
syn_st_max_vel	0x30AF-03	Synchronous startup parameter set ➔Maximum synchronous velocity
syn_st_alig_angl	0x30AF-04	Synchronous startup parameter set ➔Startup alignment angle
syn_st_alig_time	0x30AF-05	Synchronous startup parameter set ➔Startup alignment time
syn_st_alig_cur_ratio	0x30AF-06	Synchronous startup parameter set ➔Startup alignment current ratio
syn_st_trans_upp	0x30AF-07	Synchronous startup parameter set ➔Transition upper threshold
syn_st_trans_low	0x30AF-08	Synchronous startup parameter set ➔Transition lower threshold
syn_st_min_vel	0x30AF-09	Synchronous startup parameter set ➔Synchronous minimum velocity
syn_st_trials	0x30AF-0A	Synchronous startup parameter set ➔Maximum synchronous startup trials
bemf_cur_g1	0x30A5-01	Back EMF observer parameter set ➔Kalman current gain [1]
bemf_bemf_g1	0x30A5-02	Back EMF observer parameter set ➔Kalman Back EMF gain [1]
bemf_cur_g2	0x30A5-03	Back EMF observer parameter set ➔Kalman current gain [2]
bemf_bemf_g2	0x30A5-04	Back EMF observer parameter set ➔Kalman Back EMF gain [2]
syn_st_ang_acc	0x30AF-01	Synchronous startup parameter set ➔Synchronous angular acceleration
syn_st_acc_cur_ratio	0x30AF-02	Synchronous startup parameter set ➔Synchronous acceleration current ratio
syn_st_max_vel	0x30AF-03	Synchronous startup parameter set ➔Maximum synchronous velocity
syn_st_alig_angl	0x30AF-04	Synchronous startup parameter set ➔Startup alignment angle
syn_st_alig_time	0x30AF-05	Synchronous startup parameter set ➔Startup alignment time
syn_st_alig_cur_ratio	0x30AF-06	Synchronous startup parameter set ➔Startup alignment current ratio
syn_st_trans_upp	0x30AF-07	Synchronous startup parameter set ➔Transition upper threshold
syn_st_trans_low	0x30AF-08	Synchronous startup parameter set ➔Transition lower threshold
syn_st_min_vel	0x30AF-09	Synchronous startup parameter set ➔Synchronous minimum velocity
syn_st_trials	0x30AF-0A	Synchronous startup parameter set ➔Maximum synchronous startup trials
mot_cmd_vel_min	0x30F0-02	Set value command ➔Set value velocity minimum

Continued on next page.

Parameter	OBD Index	OBD Description
mot_cmd_vel_max	0x30F0-03	Set value command ➔Set value velocity maximum
mot_cur_nom	0x3001-01	Motor data ➔Nominal current
mot_cur_limit	0x3001-02	Motor data ➔Output current limit
mot_nb_poles	0x3001-03	Motor data ➔Number of pole pairs
mot_wind_time_cons	0x3001-04	Motor data ➔Thermal time constant winding
mot_torq_cons	0x3001-05	Motor data ➔Torque constant
mot_sys_tr_param	0x3002-03	Electrical system parameters ➔Transition parameter (a)
mot_sys_in_param	0x3002-04	Electrical system parameters ➔Input parameter (b)
mot_max_prof_vel	0x607F-00	➔Max profile velocity
mot_max_speed	0x6080-00	➔Max motor speed
mot_prof_acc	0x6083-00	➔Profile acceleration
mot_prof_dec	0x6084-00	➔Profile deceleration
mot_max_acc	0x60C5-00	➔Max acceleration
esc_conf_misc	0x3000-04	➔Axis configuration
esc_ctrl	0x6040-00	➔Controlword
esc_stat	0x6041-00	➔Statusword (PVM-specific Bits)
esc_err_nb	0x1003-00	Error history ➔Number of errors
esc_err_hist1	0x1003-01	Error history ➔Error history 1
esc_err_hist2	0x1003-02	Error history ➔Error history 2
esc_err_hist3	0x1003-03	Error history ➔Error history 3
esc_err_hist4	0x1003-04	Error history ➔Error history 4
esc_err_hist5	0x1003-05	Error history ➔Error history 5

Table 3-7 DroneCAN parameters (GetSet)

4 COMMUNICATION ERROR CODE DEFINITION

An abort object will be sent over the network instead of a response to a SDO request if the request has failed. The same abort code will be sent as part of the response to other transfer requests (such as USB).

4.1 Communication Errors (Abort Codes)

Following abort codes are defined by CANopen Communication Profile CiA 301. Codes greater 0x0F00 0000 are maxon-specific

Abort code	Name	Cause
0x0000 0000	No abort	Communication successful
0x0503 0000	Toggle error	Toggle bit not alternated
0x0504 0000	SDO timeout	SDO protocol timed out
0x0504 0001	Command unknown	Command specifier unknown
0x0504 0004	CRC error	CRC check failed
0x0601 0000	Access error	Unsupported access to an object
0x0601 0001	Write only error	Read command to a write only object
0x0601 0002	Read only error	Write command to a read only object
0x0602 0000	Object does not exist error	Last read or write command had wrong object index or subindex
0x0604 0043	General parameter error	General parameter incompatibility
0x0604 0047	General internal incompatibility error	General internal incompatibility in device
0x0606 0000	Hardware error	Access failed due to hardware error
0x0607 0010	Service parameter error	Data type does not match, length or service parameter do not match
0x0607 0013	Service parameter too short error	Data type does not match, length of service parameter too long
0x0609 0011	Subindex error	Last read or write command had wrong object subindex
0x0609 0030	Value range error	Value range of parameter exceeded
0x0800 0000	General error	General error
0x0800 0020	Transfer or store error	Data cannot be transferred or stored
0x0800 0022	Wrong device state error	Data cannot be transferred or stored to application because of present device state
0x0F00 FFBF	Illegal command error	Command code is illegal (does not exist)

Table 4-8 Communication errors

••page intentionally left blank••

LIST OF FIGURES

LIST OF TABLES

INDEX

maxon

CCMC | UAV-ESC Communication Guide | Edition 2022-11 | DocID rel_10886