DroneCAN Protocol

Version Modification Record

Version	Date	Author	Modification Content
V1.0.0	2024/07/24		Initial Version
V1.0.1	2025/10/30		Adopt standard #1030 for throttle control; add file transfer #223
V1.0.2	2026/02/25		In message upload #20130, the original transmission of electrical frequency data has been modified to transmit **actual rotational speed (RPM)**.

Table of Contents

1. Overview

This protocol adopts the CAN2.0B frame format (29-bit identifier) and uses extended frame data frames (traditional CAN, supporting a maximum of 8-byte transmission) for data interaction. It is designed to manage CAN transmission, implement broadcast/point-to-point data transmission, and complete data upload, service request, register read/write and designated control functions.

2. Terminology and Document Specifications

Table 1 Document Terminology

CAN	ControllerAreaNetwork, a communication protocol for control local area network
CAN-ID	Control section of CAN message
Dronecan	A highly reliable CAN application protocol for aerospace and robotics (see official website) DroneCAN
Node-ID/ Node	Device number on the bus

Note: All transmission data in this protocol is packaged and sent in **little-endian format**.

3. Protocol Description

3.1. Baud Rate and Sampling Point

Table 2 Baud Rate and Reference Sampling Point

Baud Rate	Bit Time	Reference Sampling Point	Recommended Max ESC Quantity
1000K(default)	1.00us	75.0%	≤32
800K	1.25us	80.0%	≤16
500K	2.00us	87.5%	电调数≤8
250K	4.00us	87.5%	电调数≤4

Note: The bus load rate is recommended to not exceed 70%. The maximum ESC quantities for different baud rates in the above table are for reference only; the actual number depends on the user-configured throttle refresh rate and operating parameter upload rate of the user.

3.2. Protocol CAN-ID Segmentation

This protocol uses the CAN2.0B frame format (29-bit identifier). The XC-UAVCAN-V1.0.0 protocol segments the CAN-ID as follows:

Message frame

Field name	Priority					Message type ID													Service not message	Source node ID						
CAN ID bits	28	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
Allowed values																						0	1...127						
CAN ID bytes	3					2								1								0							

Service frame

Field name	Priority					Service type ID								Request not response	Destination node ID							Service not message	Source node ID						
CAN ID bits	28	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
Allowed values															1...127							1	1...127						
CAN ID bytes	3					2								1								0							

Table 3 CAN-ID Segmentation Field Description

Field Definition	Bit Width	Description	
Priority	5	Defined by CAN2.0B; the smaller the CAN-ID, the higher the priority in bus competitive transmission. Value range: 0~31. • Highest priority: 0 • Lowest priority: 31 • Broadcast: same Priority for multi-frame transmission • Service: same Priority for multi-frame transmission (consistent for requester and responder)	0x00=Exceptional 0x02 =Immediate 0x04 =Fast 0x08 =High 0x10 =Nominal 0x14 =Low 0x18 =Slow 0x1f =Optional
Message type ID	16	Message type ID range:0~65535	[0,20000): Standard message types [20000,21000): Supplier-specific message types [21000,65535]: Reserved for future use
Service type ID	8	Service type ID range:0~255	[0,100): Standard service types [100,200): Reserved for future use [200,255]: Supplier-specific service types
Service not message	1	Service frame identifier	0: Non-service frame 1: Service frame
Request not response	1	Identifies whether the data frame is a request or response frame	0: Response to service data request 1: Service data request
Destination node ID	7	Node-ID of the CAN frame receiver (0~127)	
Source Node ID	7	Source Node ID refers to the node’s own ID. The valid range is 1－127 inclusive, among which 126 and 127 are reserved IDs.	

3.3. Protocol Data Transmission Specifications

The CAN2.0B protocol specifies a maximum of 8 bytes for a single transmission. This protocol divides the CAN data segment of a frame into a Transfer payload segment and a Tail byte segment, with the following valid payload structure:

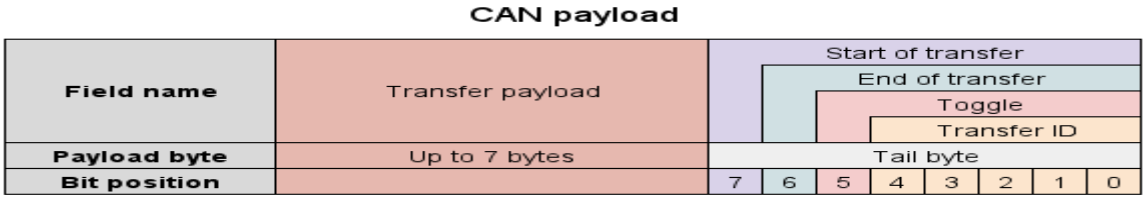


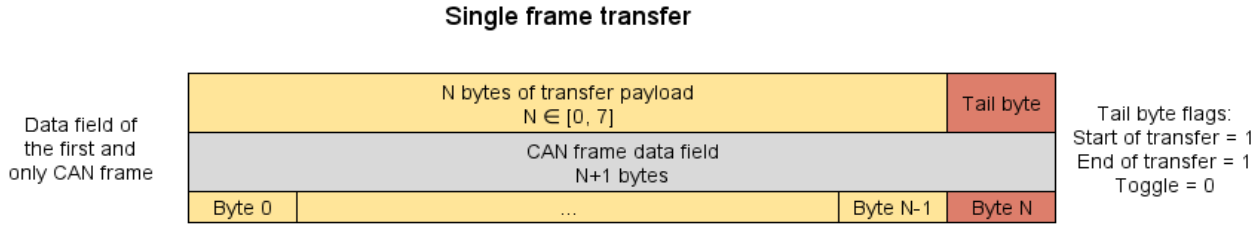
Table 4 Tail Byte Segmentation Description

Tail byte Field	Single-Frame Transmission	Multi-Frame Transmission
Start of transfer	Always 1	1 for the first frame, 0 for others
End of transfer	Always 1	0 for others, 1 for the last frame
Toggle	Always 0	0 for the first frame, toggles for each subsequent frame until the last frame
Transfer ID	Increment by 1 for each complete data packet transmission under the same Type ID, cycling 0~31	

Note: A single data packet refers to the entire multi-frame transmission process from start to end! Each Type ID has its own independent Transfer ID.

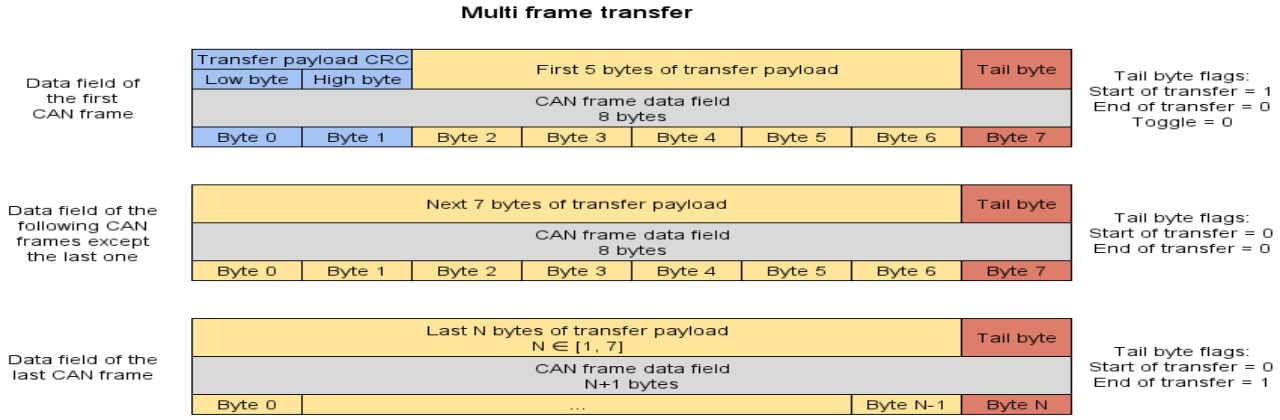
3.3.1 Single-frame Transmission

Single-frame transmission is typically used for data segments with no more than 7 bytes. The data field format is summarized as follows:



3.3.2 Multi-frame Transmission

Multi-frame transmission is used for data segments exceeding 7 bytes. The data field format is summarized as follows:



Two new concepts are introduced for multi-frame transmission:

Concept	Description
Transfer CRC	Calculated based on the transfer payload, prefixed with the data type signature, and arranged in little-endian byte order.

Toggle bit	0 for the first frame, toggles for each subsequent frame until the last frame.

CRC Calculation Steps:

1. Add 64Bit signature: SignatureCrc = crcAddSignature(0xffff, SignatureValue);
2. Add Payload segment: CRC = crcAdd(SignatureCrc, NBytesOfPayload, N);

4. Protocol Application

4.1. ESC Function Description

- ✓ Sends a heartbeat packet (including node status and other messages) approximately every 1 second after power-on.
- ✓ Supports parameter saving, emergency stop, factory reset, upgrade control, node shutdown and node restart.
- ✓ Enables register parameter read and write operations.
- ✓ Realizes data message upload.

4.2. Supplier ID Usage Plan

Broadcast frame ID: Uses the supplier ID range 20000~20999 defined in UAVCAN.

Service frame ID: Uses the supplier ID range 200~255 defined in UAVCAN.

Table Frame Type List

Frame Type	Frame Category	Frame ID (Hex/Dec)	Signature Code	Priority	Data Frame Description
Broadcast	Command Control	0x4e8e(20110)	0x4fdf12a8708a2030	Fast	Global ESC adjustment commands
	ID R/W	0x4e8f(20111)	0xc0e59ae2bdd136dd	Nominal	Read/write ID address and throttle ID address setting
	MSG1	0x4ea2(20130)	0xb95f98a2bb955035	LOW	ESC uploads current electrical frequency, bus current and status data
	MSG2	0x4ea3(20131)	0x2c8e7c3aaca9e11e	LOW	ESC uploads current bus voltage, throttle and temperature data
	Test	0x4ea4(20132)	0x17653f2149bd5d96	LOW	ESC uploads current test data
	NodeStatus(Standard)	0x0155(341)	0x8e17239cc6469287	Nominal	Node status message: power-on time, health status, etc.
	Raw Command (Standard)	0x0406(1030)	0x217f5c87d7ec951d	High	Throttle data, transmitted to 1~8-axis ESCs
Service	Register R/W	0xd2(221)	0xbec765b0e53b2ea1	Fast	Register read and write operations
	Baud R/W	0xd3(222)	0x278afc50d7f6f50d	Fast	Baud rate configuration
	Node Get(Standard)	0x01(1)	0x71e7208378f86c8e	Fast	Node information acquisition
	Node Control(Standard)	0x0A(10)	0xf0320c121476f054	Exceptional	Node command control
	Get File	0xd4(223)	0x7c0982c183e01c94	Fast	File data acquisition

4.3. Data Type Description

4.3.1. Command Control.20110

Frame Type	SUB-ID	Command	NodeId	Reserved, Filled with 0
Broadcast	20110	Payload[0]	1	2

*Description: Priority = Fast

Command:	Node Id:	
◇ 0:Disable all message upload ◇ 1:Disable all message upload (excluding heartbeat packets) ◇ 10:Manually trigger a heartbeat packet (disable all message upload is invalid) ◇ 100: Enable automatic ESC message upload ◇ 0xFE: Restart all ESCs	>127: All ESCs respond Others: Corresponding ESC responds	

4.3.2. Device ID Address Setting.20111

Frame Type	SUB-ID	Command	NodeId	ThrotId
Broadcast	20111	Payload[0]	1	2

*Description: Priority =Nominal

Command:	NodeId: Set new ID address, value range [0x01~0x7C] ThrotId: [0x01~0x08]
◇ 0:Set new node address and throttle address (only valid when ESC is not working; ESC restarts after successful setting and takes effect)。 ◇ 1:Cancel address setting (uses address encoder value if equipped; NodeId parameter is invalid) ◇ 2:Acquire node address and throttle address ◇ 3:Upload node address and throttle address	

4.3.3. Throttle Transmission.[1030]

Frame Type	SUB-ID	Throttle data
Broadcast	1030	Payload[0,N]

*Description: Priority = High

1. The number of throttle channels sent depends on user requirements; multi-frame transmission is required if more than 4 channels are sent.

2. Each throttle channel occupies 14 bits (int14) with the highest bit as the sign bit. Currently, throttle values must be ≥ 0 (values < 0 are discarded). A continuous out-of-range throttle value will trigger a throttle loss fault. Since a maximum of 7 bytes (56 bits) can be transmitted at a time, the method to parse 56-bit data into 4×14-bit throttle data is as follows:

To-be-sent throttle (HEX): [0x123, 0x234, 0x345, 0x456]

Memory format (HEX): [0x23. 0x01. 0x34. 0x02. 0x45. 0x03. 0x54. 0x04]

Memory format (HEX): [0010 0011_0000 0001 . 0011 0100_ 0000 0010 . 0100 0101_0000 0011 . 0101 0110_0000 0100]

Remove the highest 2 bits of each 16-bit throttle data.

Binary throttle (BIN) [0010 0011 00 0001 . 0011 0100 00 0010 , 0100 0101 00 0011 , 0101 0110 00 0100]

Split the throttle data by 8 bits to form 7-byte CAN format data.

Binary throttle (BIN) [0010 0011 , 0000 0100 , 1101 0000 . 0010 0100 , 0101 0000 , 1101 0101 , 1000 0100]

CAN Data (HEX) [0x23 , 0x04 , 0xD0 , 0x24 , 0x50 , 0xD5 , 0x84]

Refer to the appendix for the C program related to splitting. 1 void x_MakeThrot(uint16_t *throt,uint8_t *throtOut)

3. The minimum CAN throttle refresh cycle is 2ms (supports 500Hz adjustment frequency); throttle value range: 0~8191.

4. Refer to Appendix 1 for the C program of throttle data splitting: void x_MakeThrot(uint16_t *throt,uint8_t *throtOut).

4.3.4. Message Upload 1.20130

Frame Type	SUB-ID	Actual Rotational Speed	Bus current	Running status
Broadcast	20130	Payload[0,1,3]	[3,4]	[5,6]

*Description: Priority = Low

1.**Actual Rotational Speed**: Data type/ **uint24_t**; Range /**0 ~ 16777215**; Unit/RPM

2.母Bus Current: Data type/ **int16_t**; Range/ **-32768 ~ +32767**; Unit/0.1A,

3. Running Status: the current system operation

Transmission Setting: Default upload cycle 50ms; automatic upload enabled after power-on; can be disabled via the global command control (20050).

Running Status Bit Definition

Bit	Function	Description	Bit	Function	Description
BIT0	Overvoltage	0: Normal,1: Overvoltage	BIT8	Motor Stall	0:Normal,1:Stall
BIT1	Undervoltage	0: Normal,1: Undervoltage	BIT9	MOS Open Circuit	0: Normal,1: Open Circuit
BIT2	Overcurrent	0: Normal,1: Overcurrent	BIT10	MOS Short Circuit	0: Normal,1: Short Circuit
BIT3	Throttle Signal Source	0:PWM,1:CAN	BIT11	Motor Overtemperature	0: Normal,1: Overtemperature
BIT4	Throttle Loss	0: Normal,1: Loss	BIT12	Current Op-Amp Abnormality	0: Normal,1: Abnormality
BIT5	Throttle Not 0	0: Zeroed, 1: Not Zeroed	BIT13	NodeID Setting Source	0:Software Setting, 1:Encoder
BIT6	MOS Overtemperature	0: Normal,1: Overtemperature	BIT14	Motor Phase Line Status	0: Normal,1: Short Circuit
BIT7	Capacitor Overtemperature	0: Normal,1: Overtemperature	BIT15	Motor Running Status	0: Stopped, 1: Running

4.3.5. Message Upload 2.20131

Frame Type	SUB-ID	Output Throttle	Bus Voltage	MOS Temp	Cap Temp	Motor Temp
Broadcast	20131	Payload[0,1]	[2,3]	4	5	6

*Description: Priority = Low

1. Output Throttle: Current throttle value received by the ESC.

2. Bus Voltage: Data type/int16_t; Range/ -32768 ~ +32767; Unit/ 0.1V.

3. Temperature Calculation: Actual Temperature = Transmitted Data - 40 (Unit: ° C).

Example: Transmitted data = 23 → Actual temperature = 23-40 = -17° C.

Transmission Setting: Default upload cycle 100ms; automatic upload enabled after power-on; can be disabled via the global command control (20051).

4.3.6. Test Message Upload.20132

Frame Type	SUB-ID	Received Data Frame Count
Broadcast	20132	Payload[0,3]

*Description: Priority = Low

Transmission Setting: Default upload cycle 200ms; automatic upload disabled after power-on; can be enabled via the global command control (20051).

4.3.7. Status Message Upload.341

Frame Type	SUB-ID	Power-On Time	Health Status (uint2)	Current Mode (uint3)	Other Mode (uint3)	User-Defined
Broadcast	341	Payload[0,3]	4	4	4	[5,6]

*Description: Priority = Nominal

1. Power-On Time: Elapsed time since power-on; Unit second /sec
2. Status Definition:

Health Status & Mode Definition

Health Status		Current Mode		Other Mode (Reserved)
:0, Normal Mode		:0, Operation Mode		
:1, System Parameter Fault		:1, Initialization Mode		
:2, Major Component Fault		:2, Sensor Calibration Mode		
:3, Severe System Fault		:3, Firmware Update Mode		
		:7, Node Unavailable		

4.4. Service Data Type Description

4.4.1. Register Read/Write.221

Request : Type A, Read (2byte)

Frame Type	SUB-ID	Operation Command	Register Index
Service → Request	221	Payload[0]	1

Request : Type B, Write (4byte)

Frame Type	SUB-ID	Operation Command	Register Index	Register Value
Service → Request	221	Payload[0]	1	[2,3]

*Description: Priority = Nominal

1. Operation Command Definition:

Command Value = 0	Read single register value
Command Value = 1	Read all register parameters
Command Value = 2	Write single register value

2. Register index, Register Operation Index Address,0~255

Response: Type 1: Other Errors (2byte)

Frame Type	SUB-ID	Operation Status	Index
Service ← Response	221	Payload[0]	1

Response: Type 2: Return Register Value (4byte)

Frame Type	SUB-ID	Operation Status	Index	Register value
Service ← Response	221	Payload[0]	1	[2,3]

Response: Type 3: Return All Register Parameters (23byte)

Frame Type	SUB-ID	Operation Status	Index	Register Value	Name	Default Value	Lower Limit	Upper Limit	Attribute
Service ← Response	221	Payload[0]	1	[2,3]	[4,15]	[16,17]	[18,19]	[20,21]	22

3.Operation Status Definition:

Status Value	Description	Corresponding Response Type
Value =0x10	Operation Successful	Type 2 / Type 3
Value =0x11	Register Index Out of Range	Type 1 (Index = max supported index)
Value =0x12	Operation Attribute Error (e.g., writing to a read-only register)	Type 2
Value =0x13	Written Value Out of Range (e.g., exceeding register upper/lower limit)	Type 2
Value =0x14	Register Unreadable and Unwritable	Type 1

✧ Register Value: The stored value of the register

✧ Name: The corresponding 12-byte register name, using only standard characters

✧ Default Value: The value restored after factory reset

✧ Lower Limit: The minimum settable value of the register

✧ Upper Limit: The maximum settable value of the register

✧ Attribute: Including the following attributes

Bit0	Bit1	Bit2	Bit3~7
0: Readable, 1: Unreadable	0:Unwritable, 1:Writable	0:Lost after reset, 1:Retained after reset	Reserved

4.4.2. Baud Rate Setting.222

Request : Type A, Read (2byte)

Frame Type	SUB-ID	Baud Rate (uint8)
Service → Request	222	Payload[0]

*Description: Priority = Nominal

Payload[0]	4	5	6	7
Baud Rate	250k	500K	800k	1000K(default)

Request : Type B, Write (4byte)

Frame Type	SUB-ID	Operation Status
Service → Request	222	Payload[0]

1. Operation Status

SUCCESS = 0	Operation Successful
FAILURE = 1	Operation Failed (cannot start or take effect)

4.4.3. File Transfer.223

Request : Type C, Read Register Configuration File (2byte)(Command values 4 & 5 valid)

Frame Type	SUB-ID	File Type	File Offset (4b)
Service → Request	221	Payload[0]	[2,5]

*Description:Priority = Nominal

1. File Type Definition:

Value = 0x1	Read register configuration file (Chinese version)
Value = 0x2	Read register configuration file (English version)

Response: Type 4, Return Register Configuration File (N byte)

Frame Type	SUB-ID	Operation Status (1b)	File Offset (4b)	Total File Length (4b)	File Content N(0~256)
Service ← Response	221	Payload[0]	[1,4]	[5,8]	[9,9+N]

2. Operation Status Definition:

Status Value	Description
Value = 0x1	Read Chinese version of register configuration file
Value = 0x2	Read English version of register configuration file
Value = 0xF1	Data Out of Range
Value = 0xF2	Access Denied (not allowed in current status)
Value = 0xFF	Unknown Error

Key Note: A single multi-frame packet supports a maximum of 256 bytes of transmission. Files larger than 256 bytes must be split into multiple multi-frame packets for transmission.

4.4.4. Node Information Acquisition.1

Request :(0byte)

Frame Type	SER-ID
Service → Request	1

Response:(40+N+M byte)

Frame Type	SER-ID	StatusNode		
Service ← Response	1	Payload[0,6]		
Software Version		Optional Field Flag	Version Control Hash	Software Signature
[7,8]		9	[10,13]	[14,21]
Hardware Version		Unique Device Code	Authenticity Certificate (<=255)	
[22,23]		[24,39]	[40,40+N]	
Name (<=80)				
[40+N,40+N+M]				

*Description: Priority >=Fast

StatusNode	Refer to 4.3.7 Status Message Upload - 341.
Software/Hardware Version	Numeric value (100 = V1.0.0; other versions follow this rule).
Optional Field Flag	Indicates the setting status of optional fields (vcs_commit and image_crc).
Version Control Hash	Stores the commit hash/revision number of the version control system (e.g., Git); reserved, filled with 0.
Software Signature	Program hash value; reserved, filled with 0.
Unique Device Code	First 12 bits = chip UID; last 4 bits reserved, filled with 0.
Authenticity Certificate	Hardware authenticity certificate (max 255 bytes); reserved, filled with 0.
Name:	Node name; not used for now, filled with 0.

4.4.5. Node Command Control.10

Request:(3 . . . 258byte)

Frame Type	SER-ID	Command(1b)	Argument(6b)
Service → Request	10	Payload[0]	[1,6]

*Description: Priority >=Exceptional

OPCODE_SAVE = 0	1. Instruct the node to save data (mandatory for permanent saving of power-down retention register parameters). 2. Must be executed when the node is not working.
OPCODE_ERASE = 1	Restore the node to factory settings (takes effect after power-off).
RESTART = 0xF3	Restart the node (executable in any status, no return message). The requester can verify the restart by querying the power-on time in the node status.

Response: (1byte)

Frame Type	SER-ID	Argument	Status
Service ← Response	10	Payload[0,5]	[6]

1. Argument Definition:

SUCCESS = 0	Operation Successful
FAILURE = 1	Operation Failed (cannot start or take effect)
BAD COMMAND = 3	Command not supported
BAD STATE = 5	Operation not allowed in current node status
INTERNAL ERROR = 6	Unexpected error during operation

2. Status Definition:

SUCCESS = 0	Operation Successful (not returned for restart command)
FAILURE = 1	Operation Failed

5. Appendix 1 - Reference Program

```
/*************************************************************************************************
** CRC Calculation (CRC-16-CCITT-FALSE)
** Official Reference:  http://reveng.sourceforge.net/crc-catalogue/16.htm#crc.cat.crc-16-ccitt-false
** Initial Value: 0xFFFF
** Polynomial: 0x1021
** Reflected: No
** Output XOR: 0
** Check Value: 0x29B1
*************************************************************************************************/
uint16_t crcAddByte(uint16_t crc_val, uint8_t byte)
{
    crc_val ^= (uint16_t) ((uint16_t) (byte) << 8);
    for (int j = 0; j < 8; j++){
        if (crc_val & 0x8000U){
            crc_val = (uint16_t) ((uint16_t) (crc_val << 1) ^ 0x1021U);
        }
        else{
            crc_val = (uint16_t) (crc_val << 1);
        }
    }
    return crc_val;
}
uint16_t crcAdd(uint16_t crc_val, const uint8_t* bytes, size_t len)
{
    while (len--){
        crc_val = crcAddByte(crc_val, *bytes++);
    }
    return crc_val;
}
uint16_t crcAddSignature(uint16_t crc_val, uint64_t data_type_signature)
{
    for (int shift_val = 0; shift_val < 64; shift_val += 8) {
        crc_val = crcAddByte(crc_val, (uint8_t) (data_type_signature >> shift_val));
    }
    return crc_val;
}
```

```
/*************************************************************************************************
** Throttle Data Packaging Function
** Note: The length of the throt pointer must be > 4; the length of the throtOut pointer must be > 8.
*************************************************************************************************/
void x_MakeThrot(uint16_t *throt,uint8_t *throtOut)
{
    uint64_t temp = 0;
    uint8_t i = 0;
    //删除高两位
    throt[0] &= 0x3fffu;
    throt[1] &= 0x3fffu;
    throt[2] &= 0x3fffu;
    throt[3] &= 0x3fffu;
    //4 个 14 位油门合成
    temp = (throt[0]<<42) | (throt[1]<<24)|(throt[2]<<12)|throt[3];
    For(i = 0; i < 7; i++)
    {
      throtOut[7 - i] = (uint8_t)(temp & 0xFF);
      temp>>8;
    }
}
```

6. Appendix 2 - Register List

No (Hex)	Value	Register Name	Default Value	Lower Limit	Upper Limit	R/W/E	Description
00H	_	PRVersion	100	0	0xffff	R	Protocol version (100 = V1.0.0)
01H	_	HWVersion	100	0	0xffff	R	Hardware version (100 = V1.0.0)
02H	_	SoftVersion	100	0	0xffff	R	Software version (100 = V1.0.0)
03H	_	UploadCtrBit	0xffff	0	0xffff	R/W	Bit: Upload control bit (0 = disabled for all):
	_						BIT0: Enable Status Message Upload (341)
	_						BIT1: Enable Message Upload 1 (20130)
	_						BIT2: Enable Message Upload 2 (20131)
04H	_	CanBaudRate	7	4	7	R/W/E	CAN baud rate setting: 4:250k 5:500K 6:800k 7:1000K(default)
0cH	_	ExCommand	0	0	0xffff	R	Node control (see 4.4.5 Node Command Control - 10)
0dH	_	HealthStatus	0	0	3	R	Node health status (see 4.3.7 Status Message Upload - 341)
0eH	_	ModeStatus	1	0	3	R	
0fH	_	VendorStatus	0	0	255	R	Vendor-defined status
10H	_	ThrotSet	0	0	PWM_CNT	R/W	Throttle signal (can be updated via broadcast or register read/write)
11H	_	ThrotCrtOut	0	0	PWM_CNT	R	Current actual output throttle value
12H	_	ThrotMode	0	0	1	R/W	Throttle signal source: 0=CAN throttle , 1=PWM throttle
13H	_	EleFrequency	0	0	0xffff	R	Motor rotor electrical frequency (Unit: 0.1Hz)
18H	_	VoltageBus	0	0	0xffff	R	Bus voltage (Unit: 0.1V)
20H	_	CurrentBus	0	0	32767	R	Bus current (Unit: 0.1A)
28H		MotorPoles	21	1	100	R/W/E	Motor pole pairs, related to the upload data of throttle value
30H	_	MosTem	75	0	255	R	MOS temperature (Actual temp = Transmitted data - 40; Unit: ° C)
31H	_	CapTem	75	0	255	R	Capacitor temperature (Actual temp = Transmitted data - 40; Unit: ° C)
32H	_	MotorTem	75	0	255	R	Motor temperature (Actual temp = Transmitted data - 40; Unit: ° C)
33H	_	RunStatus	0	0	0xffff	R	Running status code (see 4.3.4 Message Upload 1 - 20130 for bit definition)

Note: R/W/E = Readable/Writable/Power-down Retention. The above list is for reference only; the actual register configuration shall prevail.