CAN

Overview

Standards

- ISO11898:
  - high-speed CAN
  - For faster transmit
- ISO11519
  - low-speed CAN
  - For longer distance transmit

Physical layer	ISO 11898 (High speed)						ISO 11519-2 (Low speed)					
Communication speed	Up to 1 Mbps						Up to 125 kbps					
Maximum bus length	40 m/ 1 Mbps						1 km/ 40 kbps					
Number of connected units	Maximum 30						Maximum 20					
Bus topology[*1]	Recessive			Dominant			Recessive			Dominant		
	Min.	Nom.	Max.	Min.	Nom.	Max.	Min.	Nom.	Max.	Min.	Nom.	Max.
CAN_High (V)	2.00	2.50	3.00	2.75	3.50	4.50	1.60	1.75	1.90	3.85	4.00	5.00
CAN_Low (V)	2.00	2.50	3.00	0.50	1.50	2.25	3.10	3.25	3.40	0.00	1.00	1.15
Potential difference (H–L) (V)	-0.5	0	0.05	1.5	2.0	3.0	-0.3	-1.5	-	0.3	3.00	-
	Twisted pair wire (shielded/unshielded) Loop bus Impedance (Z): 120 Ω (Min. 85 Ω, Max. 130 Ω) Bus resistivity (r): 70 MΩ/m Bus delay time: 5 ns/m Terminating resistance: 120 Ω (Min. 85 Ω, Max. 130 Ω)						Twisted pair wire (shielded/unshielded) Open bus Impedance (Z): 120 Ω (Min. 85 Ω, Max. 130 Ω) Bus resistivity (r): 90 MΩ/m Bus delay time: 5 ns/m Terminating resistance: 2.20 Ω (Min. 2.09 Ω, Max. 2.31 Ω) CAN_L and GND capacitance: 30 pF/m CAN_H and GND capacitance: 30 pF/m CAN_L and GND capacitance: 30 pF/m					

Basic Idea of CAN

> Note
>
> CAN is Half duplex, so only one device can send data at each time and the rest will be the receiving side

Physical Layer:

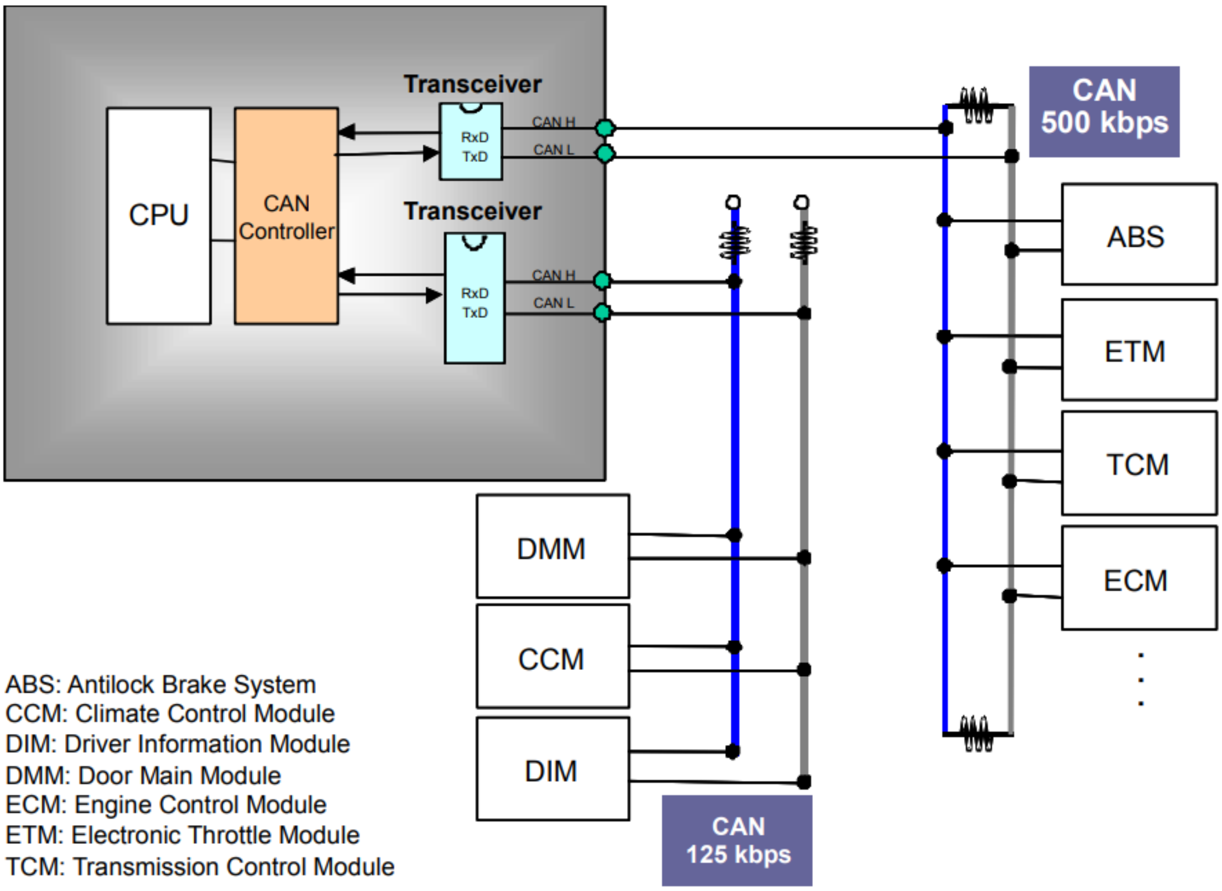
- Two lines: CAN high and CAN low
- In order to have different voltagesw, we need two Resistors on each sides.
- A CAN controller unit
- Two transceivers

CAN Transceivers

- The transceiver translates the logic level messages from the controller into the CAN differential scheme on the CANH and CANL pins of the CAN transceiver.
- With RX and TX, it allows to send and receive for each CAN node

CAN Controller

The controller can be thought of as an MCU, the part of the CAN node that processes all the information to and from the CAN bus.



Voltage Difference (CAN_H - CAN_L)

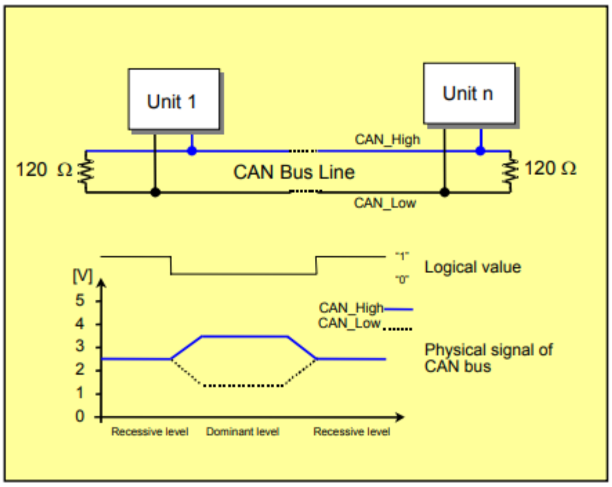
5V differential data
(Under IOS11898 standard)

- Dominant: when voltage difference is 2V
  - CAN_H(3.5V) - CAN_L(1.5V) = 2V
- Recessive: when voltage difference is 0V
  - CAN_H(2.5V) - CAN_L(2.5V) = 0V

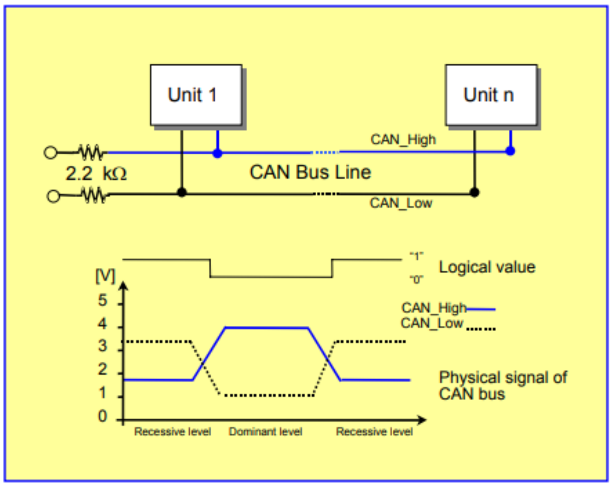
Sending 1 and 0:

- Sending 1 requires to set CAN_H = CAN_L = 2.5V (recessive)

- Sending 0 requires to set CAN_H = 3.5V; CAN_L = 1.5V (dominant)



[For ISO11898 (125 kbps to 1 Mbps)]



[For ISO11519-2 (10 kbps to 125 kbps)]

Protocal Layer

- Asynchronous
  - Baud Rate
  - Bit Timing
- ID: each device has an ID and when a device sends out a frame of data, other device will check if the ID is the ID that it want to receive, if it is receive it othervise reject it.

> Note
>
> Remote frame:
>
> - stardard format 11-bit ID
> - extended format: 29-bit ID

CAN Protocol

A frame is like a packet

- ==Data frame 📍
  - This frame is used by the transmit unit to send a message to the receive unit.
- ==Remote frame 📍
  - This frame is used by the receive unit to request transmission of a message that has the same ID from the transmit unit.
- ==Error frame
  - When an error is detected, this frame is used to notify other units of the detected error.
- ==Overload frame
  - This frame is used by the receive unit to notify that it has not been prepared to receive frames yet.
- ==Interface space
  - This frame is used to separate a data or remote frame from a preceding frame.

Data Frame

seven fields

2. Speed Controller Sending Message Format

The format in which the speed controller sends feedback data to the CAN bus.

Identifier is determined by 0x200 + speed controller ID (e.g., if the speed controller ID is 1, then the identifier of that speed controller is 0x201)

16

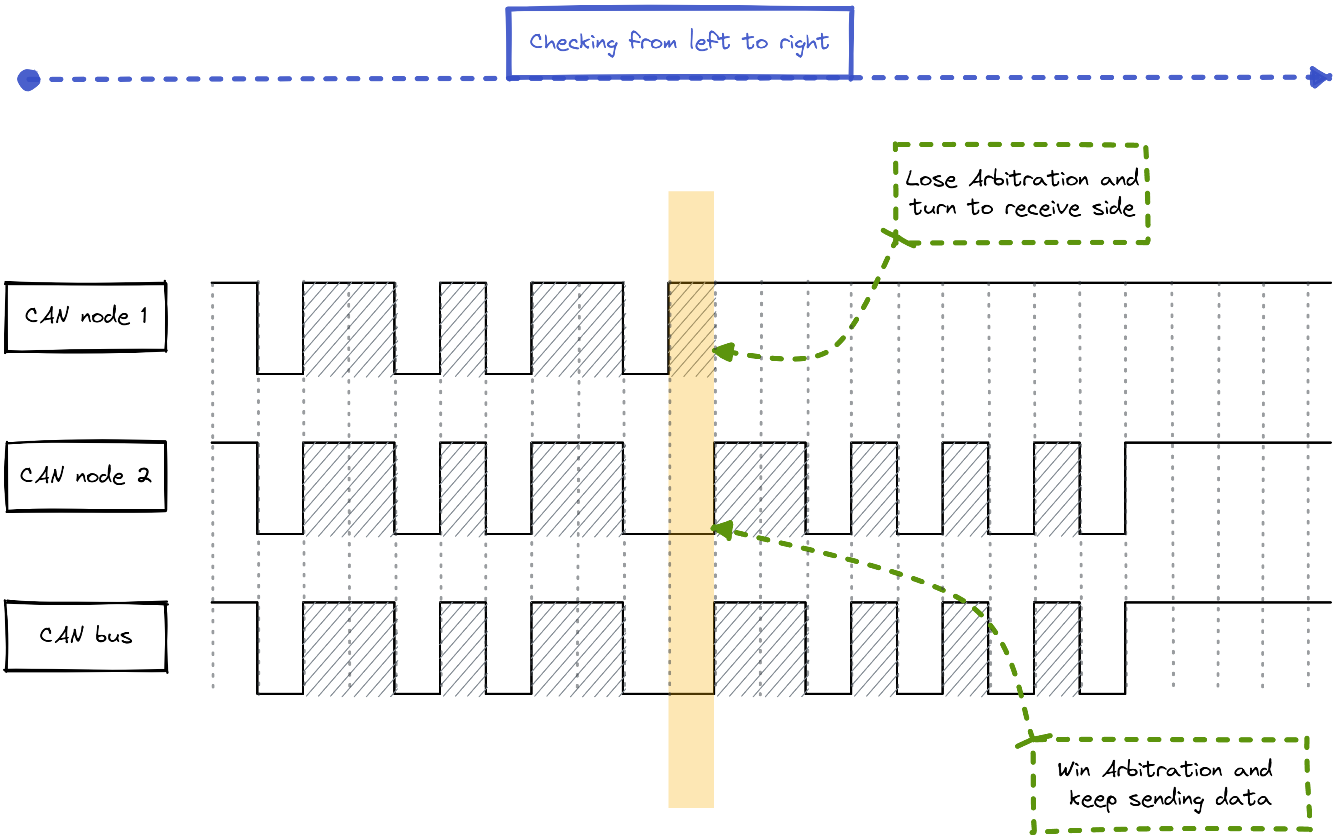
Frame type: Standard Frame format: DATA
DLC: 8 Bytes

Data Fields	Description
DATA[0]	Controls the rotor mechanical angle in higher order byte (8 bits)
DATA[1]	Controls the rotor mechanical angle in lower order byte (8 bits)
DATA[2]	Controls the rotational speed in higher order byte (8 bits)
DATA[3]	Controls the rotational speed in lower order byte (8 bits)
DATA[4]	Actual torque current in higher order byte (8 bits)
DATA[5]	Actual torque current in lower order byte (8 bits)
DATA[6]	Motor temperature
DATA[7]	Null

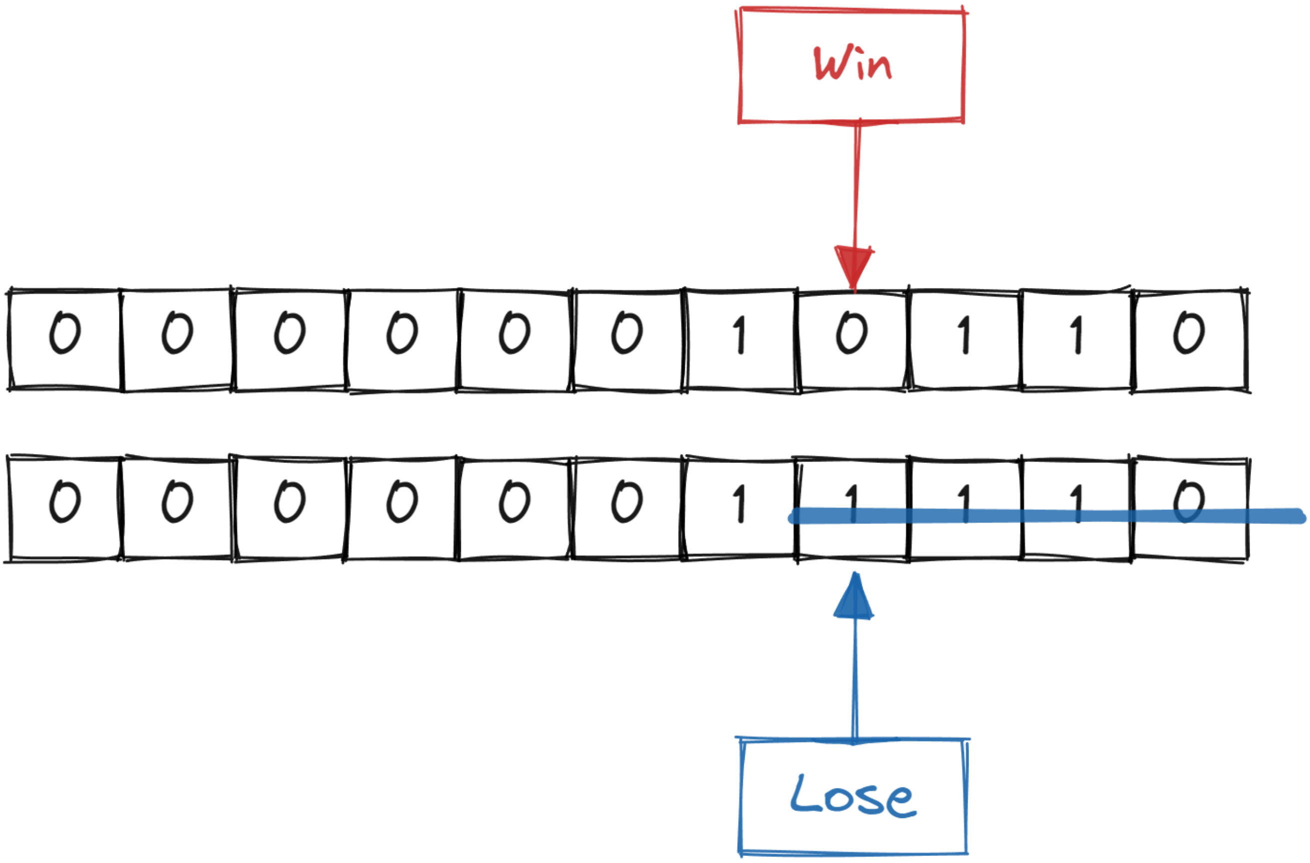
Sending frequency
1KHz by default (can be changed inside RoboMaster Assistant)
Rotor mechanical angle value range 0 ~ 8191 (corresponding mechanical angle range: 0 ~ 360°)
Expressed rotor speed value unit rpm
Expressed motor temperature unit °C

Arbitration Field

Example1:



Example2:



Remote Frame

✎ Note

Remote Frame set RTR to be recessive, and has no data field
Data Frame set RTR to be dominant

(1) Start of frame (SOF)
This field indicates the beginning of a frame.

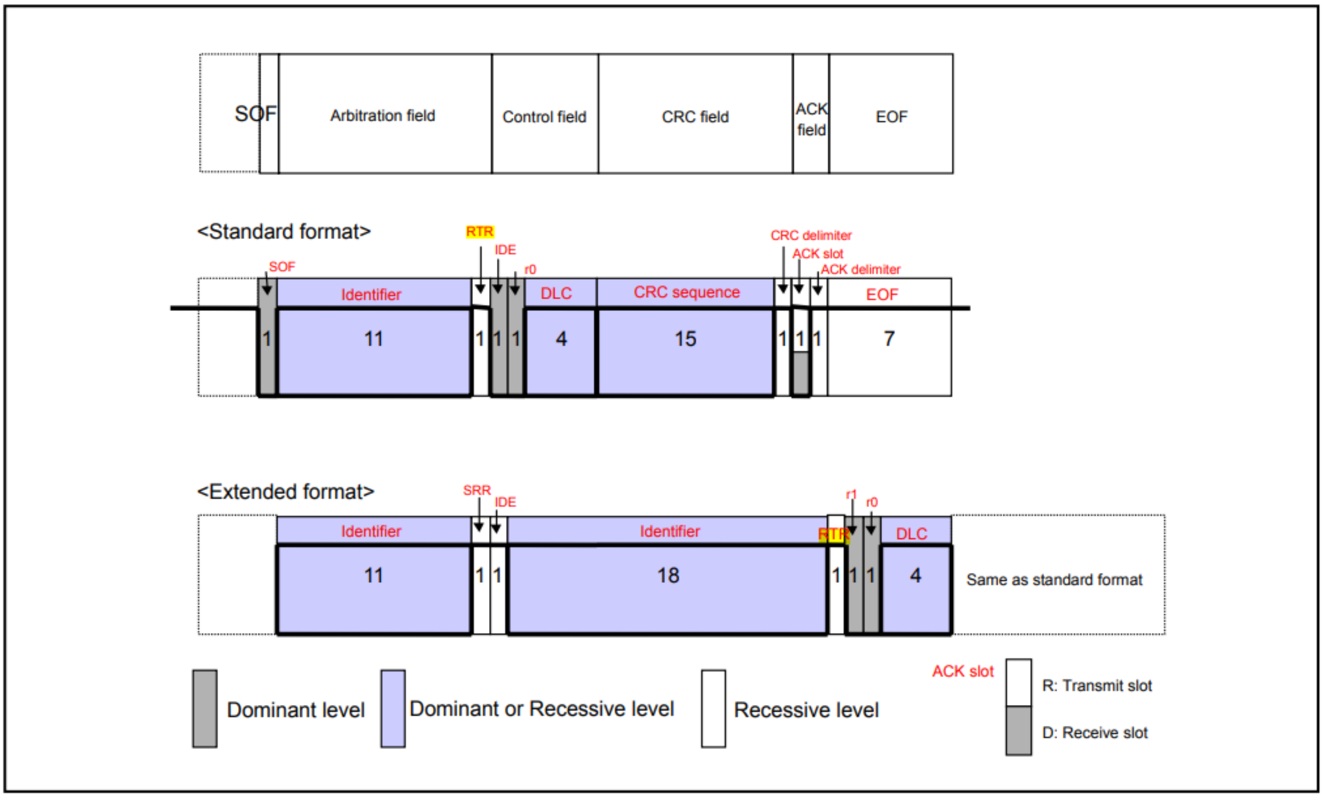
(2) Arbitration field
This field indicates the priority of data. A data frame with the same ID can be requested.

(3) Control field
This field indicates reserved bits and the number of data bytes.

(4) CRC field
This field is used to check the frame for a transmission error.

(5) ACK field
This field indicates a signal for confirmation that the frame has been received normally.

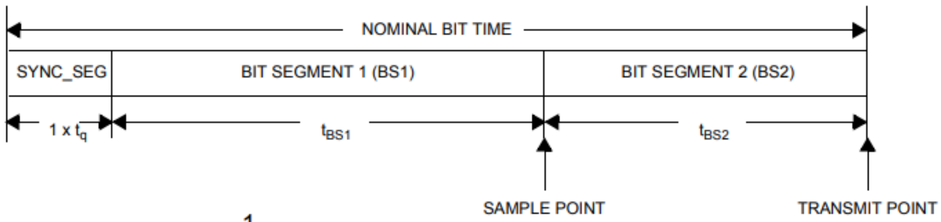
(6) End of frame
This field indicates the end of a remote frame.



CAN information on STM32 chip

- Two CAN (CAN1 and CAN2)
- Baud rate: 1Mbit/s
- 11 identifier & 29 identifier
- Three transmit mailbox
- Two receive FIFOs
- Each CAN has a 256 bytes of SRAM
- 28 shared acceptance filters

- Mask mode
    - 0x1x2
- Identifier list mode
    - 000111



$$\text{BaudRate} = \frac{1}{\text{NominalBitTime}}$$

$$\text{NominalBitTime} = 1 \times t_q + t_{BS1} + t_{BS2}$$

with:

$t_{BS1} = t_q$ x (TS1[3:0] + 1),

$t_{BS2} = t_q$ x (TS2[2:0] + 1),

t_q = (BRP[9:0] + 1) x t_{PCLK}

where t_q refers to the Time quantum

t_{PCLK} = time period of the APB clock,

BRP[9:0], TS1[3:0] and TS2[2:0] are defined in the CAN_BTR register.

Baudrate Calculation

- APB1 peripheral clocks: 42Mhz
- set BRP = 2
- $t_q = \frac{1}{\frac{42Mhz}{3}} = 71.42857142857143ns$
- set $TS1 + 1 = 10$ and set $TS2 + 1 = 3$
- $t_{BS1} = t_q * (TS1 + 1) = 71.1428 * 10 = 714.28ns$
- $t_{BS2} = t_q * (TS2 + 1) = 71.1428 * 3 = 214.28ns$
- $NominalBitTime = 1 * t_q + t_{BS1} + t_{BS2} = 71.429 + 714.28 + 214.28 \approx 1000ns$
- $BaudRate = \frac{1}{1000ns} = 1Mbsp$

Motor ECU data

Receiving

CAN Communication Protocol

1. Speed Controller Receiving Message Format

The two identifiers (0x200 and 0x1FF) control the current output of each of the four speed controllers by ID numbers. The controllable current range is -16384 ~ 0 ~ 16384. The corresponding speed controller output torque current range is -20 ~ 0 ~ 20 A.

Identifier: 0x200 Frame format: DATA
Frame type: Standard DLC: 8 Bytes

Data Fields	Description	Speed Controller ID
DATA[0]	Controls the current value in higher order byte (8 bits)	1
DATA[1]	Controls the current value in lower order byte (8 bits)	
DATA[2]	Controls the current value in higher order byte (8 bits)	2
DATA[3]	Controls the current value in lower order byte (8 bits)	
DATA[4]	Controls the current value in higher order byte (8 bits)	3
DATA[5]	Controls the current value in lower order byte (8 bits)	
DATA[6]	Controls the current value in higher order byte (8 bits)	4
DATA[7]	Controls the current value in lower order byte (8 bits)	

Identifier: 0x1FF Frame format: DATA
Frame type: Standard DLC: 8 Bytes

Identifier: 0x1FF Frame format: DATA
Frame type: Standard DLC: 8 Bytes

15

Data Fields	Description	Speed Controller ID
DATA[0]	Controls the current value in higher order byte (8 bits)	5
DATA[1]	Controls the current value in lower order byte (8 bits)	
DATA[2]	Controls the current value in higher order byte (8 bits)	6
DATA[3]	Controls the current value in lower order byte (8 bits)	
DATA[4]	Controls the current value in higher order byte (8 bits)	7
DATA[5]	Controls the current value in lower order byte (8 bits)	
DATA[6]	Controls the current value in higher order byte (8 bits)	8
DATA[7]	Controls the current value in lower order byte (8 bits)	

Transimit

2. Speed Controller Sending Message Format
The format in which the speed controller sends feedback data to the CAN bus.

Identifier is determined by 0x200 + speed controller ID (e.g., if the speed controller ID is 1, then the identifier of that speed controller is 0x201)

Frame type: Standard   Frame format: DATA
DLC: 8 Bytes

Data Fields	Description
DATA[0]	Controls the rotor mechanical angle in higher order byte (8 bits)
DATA[1]	Controls the rotor mechanical angle in lower order byte (8 bits)
DATA[2]	Controls the rotational speed in higher order byte (8 bits)
DATA[3]	Controls the rotational speed in lower order byte (8 bits)
DATA[4]	Actual torque current in higher order byte (8 bits)
DATA[5]	Actual torque current in lower order byte (8 bits)
DATA[6]	Motor temperature
DATA[7]	Null

Sending frequency
1KHz by default (can be changed inside RoboMaster Assistant)
Rotor mechanical angle value range 0 ~ 8191 (corresponding mechanical angle range: 0 ~ 360°)
Expressed rotor speed value unit rpm
Expressed motor temperature unit °C