DJI R SDK

Protocol and User Interface

V2.2 2020.10

Edited by_____

Approved by_____

Edited on_____

Document No. _____

SZ DJI TECHNOLOGY CO., LTD.

Release Notes

Version	Date	Section	Reason for Change	Description of Change
1.0.0.0	July 17, 2019			Draft document
2.0.0.0	October 8, 2019	3	1. Deleted sample code 2. Added cyclic redundancy check (CRC) parameters description	1. First release 2. Added CRC model parameters description
2.1.0.1	May 11, 2020	2.3 3.3 3.4	Added commands and CRC pattern sample	1. Added module version protocol 2. Added sample of command group pack 3. Added CRC sample code
2.1.0.2	June 17, 2020	2.3, 3.1	Added external device control command and hardware support description	1. Added joystick command 2. Added CAN support
2.2.0.3	June 22, 2020	2.3	Added commands	1. Added function to obtain handheld gimbal user parameters 2. Added function to set handheld gimbal user parameters 3. Added function to set gimbal operating mode 4. Added Recenter and Selfie 5. Added third-party camera motion command
2.2.0.4	July 16, 2020	2.3	Added commands	1. Added Follow Mode settings 2. Added Auto Tune settings and information push function 3. Added ActiveTrack settings 4. Added function to obtain camera status

Contents

1. DJI R SDK Protocol Introduction

The DJI R SDK protocol is a simple, easy, stable, and reliable communication protocol. A third party can control the handheld gimbal device movement and obtain its partial information via the DJI R SDK protocol. With the support of the DJI R SDK protocol, the handheld gimbal device has greater extensibility and can be applied in more scenarios.

2. DJI R SDK Protocol Description

2.1 Data Format

The data packet format of the DJI R SDK protocol is shown below:

SOF	Ver/Length	CmdType	ENC	RES	SEQ	CRC-16	DATA	CRC-32
1-byte	2-byte	1-byte	1-byte	3-byte	2-byte	2-byte	n-byte	4-byte

- Figure 1 Data Packet Format -

2.2 Field Description

Domain	Offset	Size	Descriptions
SOF	0	1	The frame header is set as 0xAA
Ver/Length	1	2	[15:10] - Version number (0 by default) [9:0] - The length of the entire frame Note: LSB first
CmdType	3	1	[4:0] - Reply type 0 - No reply is required after data is sent 1 - Can reply or not after data is sent 2-31 - Reply is required after data is sent [5] - Frame type 0 - Command frame 1 - Reply frame [7:6] - Reserve (0 by default)
ENC	4	1	[4:0] - The length of supplementary bytes when encrypting (16-byte alignment is required when encrypting) [7:5] - Encryption type 0 - Unencrypted 1 - AES256 encryption
RES	5	3	Reserved byte segment
SEQ	8	2	Serial number
CRC-16	10	2	Frame header check
DATA	12	n	Data segment (description is shown below)
CRC-32	n+12	4	Frame check (the entire frame)

- Figure 2 Data Packet Field Description -

There are two kinds of data segment content according to the frame type:

1. When the frame type is command frame, the content of the data segment is shown as below:

Domain	Offset	Domain	Descriptions
CmdSet	0	1	Command set
CmdID	1	1	Command code
CmdData	2	n-2	Data content

- Figure 3 Command Frame Data Segment Content -

2. When the frame type is reply frame, the content of the data segment is shown as below:

Domain	Offset	Size	Descriptions
DATA	0	n	Data content

- Figure 4 Reply Frame Data Segment Content -

2.3 Detailed Descriptions

2.3.1 Commands Set and Command ID

The command sets and command codes used by the handheld gimbal are shown below:

CmdSet	CmdID	Descriptions
0x0E	0x00	Control handheld gimbal position 2.3.4.1 Handheld Gimbal Position Control
	0x01	Control handheld gimbal speed 2.3.4.2 Handheld Gimbal Speed Control
	0x02	Obtain the angle information of handheld gimbal, including joint angle and attitude angle 2.3.4.3 Obtain Handheld Gimbal Information
	0x03	Set handheld gimbal limit angle 2.3.4.4 Handheld Gimbal Limit Angle Settings
	0x04	Obtain handheld gimbal limit angle 2.3.4.5 Obtain Handheld Gimbal Limit Angle
	0x05	Set handheld gimbal motor stiffness 2.3.4.6 Handheld Gimbal Motor Stiffness Settings
	0x06	Obtain handheld gimbal motor stiffness 2.3.4.7 Obtain Handheld Gimbal Motor Stiffness
	0x07	Set information push of handheld gimbal parameters 2.3.4.8 Handheld Gimbal Parameter Information Push Settings
	0x08	Push handheld gimbal parameters 2.3.4.9 Push Handheld Gimbal Parameter

0x0E	0x09	Obtain module version number 2.3.4.10 Obtain Module Version Number
	0x0A	Push joystick control command 2.3.4.11 External Device Control Command Push
	0x0B	Obtain handheld gimbal user parameters 2.3.4.12 Obtain Handheld Gimbal User Parameter
	0x0C	Set handheld gimbal user parameters 2.3.4.13 Handheld Gimbal User Parameters Settings
	0x0D	Set handheld gimbal operating mode 2.3.4.14 Handheld Gimbal Operating Mode Settings
	0x0E	Set handheld gimbal Recenter, Selfie, and Follow modes 2.3.4.15 Handheld Gimbal Recenter, Selfie, and Follow Modes Settings
0x0D	0x00	Third-party camera motion command 2.3.5.1 Third-Party Camera Motion Command
	0x01	Third-party camera status obtain command 2.3.5.2 Third-Party Camera Status Obtain Command

- Figure 5 Command Set and Command -

2.3.2 Return Code

Return codes currently supported by the handheld gimbal are shown below:

Error Code Value	Implication
0x00	Command execution succeeds
0x01	Command parse error
0x02	Command execution fails
0xFF	Undefined error

- Figure 6 Return Code Implication -

2.3.3 Device ID

The device ID is a 4-byte figure used to differentiate devices that connect to the DJI R SDK system. The ID must be submitted to DJI for approval and can only be used once approved. The device IDs currently in use are listed below:

Device ID	Descriptions
0x00000000	Reserved
0x00000001	DJI R SDK
0x00000002	Remote controller

- Figure 7 Device ID -

2.3.4 Gimbal Command Set Data Segment Details

2.3.4.1 Handheld Gimbal Position Control

CmdSet = 0x0E CmdID = 0x00 (the data segment details are shown below):

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	2	yaw_angle	int16_t	yaw angle, unit: 0.1° (range: -1800 to +1800)
	2	2	roll_angle	int16_t	roll angle, unit: 0.1° (range: -1800 to +1800)
	4	2	pitch_angle	int16_t	pitch angle, unit: 0.1° (range: -1800 to +1800)
	6	1	ctrl_byte	uint8_t	[7:4] - Reserved (must be 0) [3] - Whether the pitch axis is valid/invalid 0: Valid 1: Invalid [2] - Whether the roll axis is valid/invalid 0: Valid 1: Invalid [1] - Whether the yaw axis is valid/invalid 0: Valid 1: Invalid [0] - Control mode 0: Incremental control 1: Absolute control
	7	1	time_for_action	uint8_t	Command execution speed, unit: 0.1s This field is used to set the motion speed when the gimbal is executing this command. For example, when this field is 20, the gimbal will rotate to the position desired within 2s at a constant speed.
Reply frame	0	1	return code	uint8_t	Refer to error return code 2.3.2 Return Error

- Figure 8 Position Control Command -

2.3.4.2 Handheld Gimbal Speed Control

CmdSet = 0x0E CmdID = 0x01 (the data segment details are shown below):

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	2	yaw_speed	int16_t	Unit: 0.1°/s (range: 0°/s to 360°/s)
	2	2	roll_speed	int16_t	Unit: 0.1°/s (range: 0°/s to 360°/s)
	4	2	pitch_speed	int16_t	Unit: 0.1°/s (range: 0°/s to 360°/s)
	6	1	ctrl_byte	uint8_t	[7] - Control Bit 0: Release speed control 1: Take over speed control [6:4] - Reserved, (must be 0) [3] - Camera focal length 0: The moving speed will take the impact of camera focal length into consideration 1: The moving speed will not take the impact of camera focal length into consideration [2:0] - Reserved (must be 0)
Reply frame	0	1	return code	uint8_t	Refer to error return code 2.3.2 Return Code

- Figure 9 Speed Control Command -

Note: This command can only control for 0.5s each time it is issued due to safety reasons. If users require continuous speed, they can send this command periodically. If users want to stop the rotation of three axes immediately, they can set the fields of yaw_speed, pitch_speed, and roll_speed as 0.

2.3.4.3 Handheld Gimbal Information Obtaining

CmdSet = 0x0E CmdID = 0x02 (the data segment details are shown below):

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1	ctrl_byte	uint8_t	0x00: No operation 0x01: Obtain the attitude angle of handheld gimbal 0x02: Obtain the joint angle of handheld gimbal
Reply frame	0	1	return code	uint8_t	Refer to error return code 2.3.2 Return Code
	1	1	data_type	uint8_t	0x00: Data is not ready 0x01: The current angle is attitude angle 0x02: The current angle is joint angle

Reply frame	2	2	yaw	int16_t	yaw axis angle (unit: 0.1°)
	4	2	roll	int16_t	roll axis angle (unit: 0.1°)
	6	2	pitch	int16_t	pitch axis angle (unit: 0.1°)

- Figure 10 Obtain Gimbal Information Command -

2.3.4.4 Handheld Gimbal Limit Angle Settings

CmdSet = 0x0E CmdID = 0x03 (the data segment details are shown below):

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1	ctrl_byte	uint8_t	0x00: No operation 0x01: Set handheld gimbal limit angle
	1	1	pitch_max	uint8_t	Max. tilt axis angle (range: 0 to 179)
	2	1	pitch_min	uint8_t	Min. tilt axis angle (range: 0 to 179)
	3	1	yaw_max	uint8_t	Max. pan axis angle (range: 0 to 179)
	4	1	yaw_min	uint8_t	Min. pan axis angle (range: 0 to 179)
	5	1	roll_max	uint8_t	Max. roll axis angle (range: 0 to 179)
	6	1	roll_min	uint8_t	Min. roll axis angle (range: 0 to 179)
Reply frame	0	1	return code	uint8_t	Refer to error return code 2.3.2 Return Code

- Figure 11 Set Gimbal Limit Angle Command -

2.3.4.5 Obtain Handheld Gimbal Limit Angle

CmdSet = 0x0E CmdID = 0x04 (the data segment details are shown below):

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1	ctrl_byte	uint8_t	0x00: No operation 0x01: Obtain handheld gimbal limit angle
Reply frame	0	1	return code	uint8_t	Refer to error return code 2.3.2 Return Code
	1	1	pitch_max	uint8_t	Max. tilt axis angle (range: 0 to 179)
	2	1	pitch_min	uint8_t	Min. tilt axis angle (range: 0 to 179)
	3	1	yaw_max	uint8_t	Max. pan axis angle (range: 0 to 179)
	4	1	yaw_min	uint8_t	Min. pan axis angle (range: 0 to 179)
	5	1	roll_max	uint8_t	Max. roll axis angle (range: 0 to 179)
	6	1	roll_min	uint8_t	Min. roll axis angle (range: 0 to 179)

- Figure 12 Obtain Gimbal Limit Angle Command -

2.3.4.6 Handheld Gimbal Motor Stiffness Settings

CmdSet = 0x0E CmdID = 0x05 (the data segment details are shown below):

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1	ctrl_byte	uint8_t	0x00: No operation 0x01: Set handheld gimbal motor stiffness
	1	1	pitch_stiffness	uint8_t	VALUE : 0 ~ 100
	2	1	roll_stiffness	uint8_t	VALUE : 0 ~ 100
	3	1	yaw_stiffness	uint8_t	VALUE : 0 ~ 100
Reply frame	0	1	return code	uint8_t	Refer to errtor return code 2.3.2 Return Code

- Figure 13 Set Motor Stiffness Command -

2.3.4.7 Obtain Handheld Gimbal Motor Stiffness

CmdSet = 0x0E CmdID = 0x06 (the data segment details are shown below):

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1	ctrl_byte	uint8_t	0x00: No operation 0x01: Obtain handheld gimbal motor stiffness
Reply frame	0	1	return code	uint8_t	Refer to error return code 2.3.2 Return Code
	1	1	pitch_ stiffness	uint8_t	VALUE : 0 ~ 100
	2	1	yaw_ stiffness	uint8_t	VALUE : 0 ~ 100
	3	1	roll_ stiffness	uint8_t	VALUE : 0 ~ 100

- Figure 14 Obtain Motor Stiffness Command -

2.3.4.8 Handheld Gimbal Parameter Push Settings

CmdSet = 0x0E CmdID = 0x07 (the data segment details are shown below)

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1	ctrl_byte	uint8_t	0x00: No operation 0x01: Enable handheld gimbal parameter push 0x02: Disable handheld gimbal parameter push
Reply frame	0	1	return code	uint8_t	Refer to error return code 2.3.2 Return Code

- Figure 15 Gimbal Push Setting Command -

2.3.4.9 Handheld Gimbal Parameter Push

CmdSet = 0x0E CmdID = 0x08 (the data segment details are shown below):

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1	ctrl_byte	uint8_t	[0]: Angle information valid symbol 0: Angle information currently pushed is invalid (attitude angle, joint angle) 1: Angle information currently pushed is valid (attitude angle, joint angle) [1]: Valid symbol of angle limit information 0: Angle limit information currently pushed is invalid 1: Angle limit information currently pushed is valid [2]: Valid symbol of motor stiffness information 0: Motor stiffness information currently pushed is invalid 1: Motor stiffness information currently pushed is valid
	1	2	yaw_angle	int16_t	Unit: 0.1°
	3	2	roll_ angle	int16_t	Unit: 0.1°
	5	2	pitch_ angle	int16_t	Unit: 0.1°
	7	2	yaw_joint_agnle	int16_t	Unit: 0.1°
	9	2	roll_ joint_agnle	int16_t	Unit: 0.1°
	11	2	pitch_ joint_agnle	int16_t	Unit: 0.1°
	13	1	pitch_max	uint8_t	Max. tilt axis angle (range: 0 to 179)
	14	1	pitch_min	uint8_t	Min. tilt axis angle (range: 0 to 179)
	15	1	yaw_max	uint8_t	Max. pan axis angle (range: 0 to 179)
	16	1	yaw_min	uint8_t	Min. pan axis angle (range: 0 to 179)
	17	1	roll_max	uint8_t	Max. roll axis angle (range: 0 to 179)
	18	1	roll_min	uint8_t	Min. roll axis angle (range: 0 to 179)
	19	1	pitch_stiffness	uint8_t	VALUE : 0 ~ 100
	20	1	yaw_ stiffness	uint8_t	VALUE : 0 ~ 100
	21	1	roll_ stiffness	uint8_t	VALUE : 0 ~ 100

- Figure 16 Gimbal Parameter Push Command -

2.3.4.10 Obtain Module Version Number

CmdSet = 0x0E CmdID = 0x09 (the data segment details are shown below):

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	4	Device ID	uint32_t	Refer to 2.3.3 Device ID Number for specific device IDs.
Reply frame	0	1	Return code	uint8_t	Refer to 2.3.2 Return Code for return codes.
	1	4	Device ID	uint32_t	Refer to 2.3.3 Device ID Number for specific device IDs.
	5	4	Version Number	uint32_t	0xAABBCCDD means that the version is: AA.BB.CC.DD

- Figure 17 Obtain the Definition of Module Version Number -

2.3.4.11 External Device Control Command Push

CmdSet = 0x0E CmdID = 0x0A (this command is used by external devices to control the gimbal. For example, the joystick or dial can use this command to control the gimbal to rotate.)

The controllers currently supported are shown below:

Controller Type	Descriptions
0x00	Unknown controller
0x01	Joystick controller
0x02	Dial controller

- Figure 18 External Controller Type -

When the gimbal uses the joystick to control, the Y and X directions of the joystick map to the pitch and yaw axes by default.

Gimbal Angular Speed	Joystick Speed
pitch_speed	Y_speed
roll_speed	0
yaw_speed	X_speed

- Figure 19 Joystick Controller Default Mapping Relationship -

Users can use this command to change the mapping relationship when necessary. For example, the joystick can be mapped to pitch and roll axes.

Gimbal Angular Speed	Joystick Speed
pitch_speed	Y_speed
roll_speed	X_speed
yaw_speed	0

- Figure 20 Joystick Controller Changing Mapping Relationship -

The data segment details sent by the joystick controller are shown below:

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1	device_type	uint8_t	0x01: Joystick controller
	1	2	pitch_speed	int16_t	VALUE : -15000 ~ 15000
	3	2	roll_speed	int16_t	VALUE : -15000 ~ 15000
	5	2	yaw_speed	int16_t	VALUE : -15000 ~ 15000
Reply frame					This command has no reply frame

- Figure 21 Joystick Controller Data Segment -

Notes: VALUE in the previous table means

$$value = \frac{adc\_value - middle\_value}{adc\_range} * 15000$$

adc_value: ADC sample value of the current joystick
middle_value: joystick median
adc_range: sampling precision of ADC

Users can use an external dial to control parameters such as the focus and exposure of the gimbal or the camera via the gimbal settings.

The data segment details sent by the dial controller are shown below:

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1	device_type	uint8_t	0x02: Dial controller
	1	2	dial_speed	int16_t	VALUE : -2048 ~ 2048
Reply frame					This command has no reply frame

- Figure 22 Dial Controller Data Segment -

2.3.4.12 Obtaining Handheld Gimbal User Parameters

CmdSet = 0x0E CmdID = 0x0B) (the gimbal user parameters can be obtained in TLV format, separately, or in combination.) TLV means ID+LENGTH+VALUE; ID refers to the command type; LENGTH refers to the VALUE length; and VALUE refers to the control status. The data type of VALUE depends on the ID. The data segment details are shown below:

Name	Type	Length	Type	Value
Parameter table number selection	0x00	1	uint8_t	0x00: Parameter table 0 0x01: Parameter table 1 0x02: Parameter table 2

Special functions under Follow mode	0x22	1	uint8_t	[6-7] uint8_t: 2 (Reserved bit) [3-5] uint8_t: 3 roll 360 mode settings 0 = normal 3-axis mode 1 = 2-axis mode 2 = ROLL 360 mode 3 = 3D_ROLL360 mode [2] uint8_t: 1 Reserved bit [1] uint8_t: 1 Reserved bit [0] uint8_t: 1 Reserved bit
Motor special function	0x23	1	uint8_t	VALUE: [0] whether to power off the motor [1-7] Reserved

- Figure 23 User Parameters Data Segment -

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1-N	read_ids	uint8_t[1]	Read id
Reply frame	0	1	return code	uint8_t	Refer to error return code 2.3.2 Return Code
	1	1~N-1	tlv_buffer	uint8_t[1]	Refer to the previous table for the TLV format

- Figure 24 Obtain Handheld Gimbal User Parameters -

2.3.4.13 Handheld Gimbal Parameter Information Push Settings

CmdSet = 0x0E CmdID = 0x0C (the gimbal user parameters can be obtained in TLV format, separately, or in combination. TLV means ID+LENGTH+VALUE; ID refers to the command type; LENGTH refers to the VALUE length; VALUE refers to the control status. The data type of VALUE depends on the ID. For data segment details, refer to the user parameter data segment of 2.3.4.12 Obtaining Handheld Gimbal User Parameters

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1	tlv_id	int8_t	TLV id (refer to the previous table for definition of TLV.)
	1	2	tlv_length	uint8_t	TLV data length (the data length is decided by the corresponding ID data length shown in the previous table.)

Command frame	2	3-4	tlv_data	uint8_t[2]	TLV data segment (the data length is decided by the corresponding ID data length shown in the previous table.)
Reply frame	0	1	return code	uint8_t	Refer to error return code 2.3.2 Return Code
	1	1~N-1	tlv_buffer	uint8_t[1]	For TLV format, refer to user parameter data segment of 2.3.4.12 Obtaining Handheld Gimbal User Parameters

- Figure 25 Set Handheld Gimbal User Parameters -

2.3.4.14 Handheld Gimbal Operating Mode Settings

CmdSet = 0x0E CmdID = 0x0D (the data segment details are shown below):

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1	Operating Mode	uint8_t	0xFE: Mode remains unchanged
	1	1	Landscape and portrait mode	uint8_t	0x00: Do not switch landscape and portrait mode 0x01: Switch to landscape mode, with a 0° rotation around the X axis. 0x02: Switch to landscape mode, with a 180° rotation around the X axis. 0x03: Switch to portrait mode, with a 90° rotation around the X axis. 0x04: Switch to portrait mode, with a -90° rotation around the X axis. 0x05: Switch between landscape and portrait mode (the gimbal will automatically adapt to the most appropriate angle) 0xFF: Restore to default mode (the gimbal will automatically adapt to the most appropriate angle)
Reply frame	0	1	return code	uint8_t	Refer to error return code 2.3.2 Return Code

- Figure 26 Handheld Gimbal Operating Mode Settings -

2.3.4.15 Handheld Gimbal Recenter, Selfie, and Follow Modes Settings

CmdSet = 0x0E CmdID = 0x0E (the data segment details are shown below):

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1	Operating Mode	uint8_t	VALUE : 0xFE
	1	1	Recenter and Selfie command	uint8_t	0x01: execute Recenter once 0x02: execute Selfie once
Reply frame	0	1	return code	uint8_t	Refer to error return code 2.3.2 Return Code

- Figure 27 Set Handheld Gimbal Recenter and Selfie -

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1	Operating Mode	uint8_t	VALUE: 0x00: Gimbal Lock mode 0x02: Gimbal Yaw Follow mode 0x03: Sport mode
	1	1	Recenter and Selfie command	uint8_t	0x00: unchanged Notes: When choosing the above modes, this field must be set as 0
Reply frame	0	1	return code	uint8_t	Refer to error return code 2.3.2 Return Code

- Figure 28 Set Handheld Gimbal Follow Mode -

2.3.4.16 Gimbal Auto Calibration Settings

CmdSet = 0x0E CmdID = 0x0F (realize relevant functions of gimbal auto calibration such as gimbal stiffness auto calibration.) This command uses TLV format, which is ID+LENGTH+VALUE. ID refers to the command type; LENGTH refers to the VALUE length; VALUE refers to the control status. The data type of VALUE depends on the ID. This command issues multiple TLV combinations each time, realizing combined commands control. Data segment details are shown below:

Name	Type	Length	Type	Value
Control parameters self-tuning	0x00	1	uint8_t	[0]: Symbol of enabling 0: Stop self-tuning 1: Start self-tuning [7:1]: Self-tuning type 0: default mode 1: single attitude mode Notes: Self-tuning type selects 1

- Figure 29 Set Gimbal Auto Calibration -

2.3.4.17 Gimbal Auto Calibration Status Push

CmdSet = 0x0E CmdID =0x10, (realize the progress and status push of gimbal control parameter auto calibration.) This command uses TLV format, which is ID+LENGTH+VALUE. ID refers to the command type; LENGTH refers to the VALUE length; VALUE refers to the control status. The data type of VALUE depends on the ID. The data segment details are shown below:

Name	Type	Length	Type	Value
Control parameters self-tuning	0x00	6	uint8_t	VALUE: Byte0: Auto calibration status and result 0 = No auto calibration 0x01: auto calibration is running 0x02: auto calibration completed 0x03: auto calibration error Byte1: Auto calibration progress (Range: 0 to 100) Byte2-5: Auto calibration error status preserved

- Figure 30 Gimbal Auto Calibration Status Push -

2.3.4.18 Gimbal ActiveTrack Settings

CmdSet = 0x0E CmdID = 0x11 (the data segment details are shown below):

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1	Enable ActiveTrack	uint8_t	VALUE: 0x03: switch the start or stop status of tracking

- Figure 31 Gimbal Auto Calibration Status Push -

2.3.5 Camera Command Set Data Segment Details

2.3.5.1 Third-Party Camera Motion Command

CmdSet = 0x0D CmdID =0x00 (the data segment details are shown below):

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	2	Camera control command	Uint16_t	0x0001: shutter 0x0002: stop shuttering 0x0003: start recording 0x0004: stop recording 0x0005: center focus 0x000B: end center focus
Reply frame	0	1	return code	uint8_t	Refer to error return code

- Figure 32 Third-Party Camera Motion Command -

2.3.5.2 Third-Party Camera Status Obtain Command

CmdSet = 0x0D CmdID = 0x01 (the data segment details are shown below):

Frame Type	Data				
	Offset	Size	Name	Type	Descriptions
Command frame	0	1	Camera status obtain	uint8_t	0x01: query recording status
Reply frame	0	1	return code	uint8_t	Refer to error return code
	1	1	Camera status	uint8_t	VALUE: 0x00: not recording 0x02: recording

- Figure 33 Third-Party Camera Status Obtain Command -

3. Notices

3.1 Hardware Support

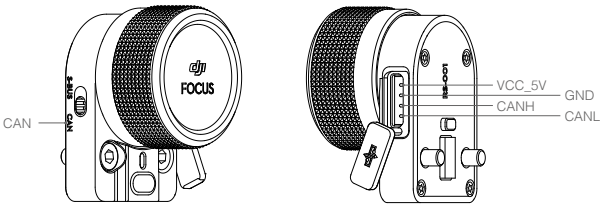
The communication interface for DJI RS 2 is CAN and its parameters are shown below:

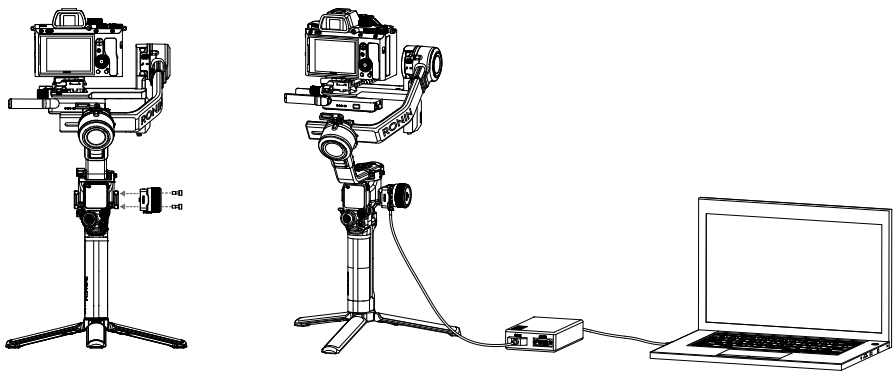
Baud rate	Frame type	CAN Tx	CAN Rx
1M	Standard frame	0x223	0x222

- Figure 34 CAN Communication Parameters -

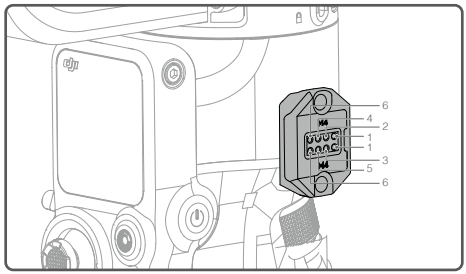
3.1.1 Device Connection Diagram

Below shows how DJI RS 2 connects to a PC via the CAN converter:





3.1.2 Ronin Series Accessories (RSA)/NATO Ports



Pin	Signal	Description	Notes
1	VCC	Power output	Supply voltage range is 8 V ± 0.4 V, rated output current is 0.8 A, and the peak value is 1.2 A
2	CANL	CANL	/
3	SBUS_RX	SBUS input	/
4	CANH	CANH	/
5	AD_COM	Accessory detect port	DJI RS 2 has a built-in pull-up resistor and it is recommended to use an accessory with a 10-100k pull-down resistor. The NATO port will not output power unless an accessory is mounted
6	GND	GND	/

3.2 Software Support

The CRC16 and CRC32 parameters used in the data packet is shown below:

Name	Width	Poly	Init	RefIn	RefOut	XorOut
CRC16	16	0x8005	0xc55c	True	True	0x0000
CRC32	32	0x04c11db7	0xc55c0000	True	True	0x00000000

3.3 Command Sample

Below is a simple example of gimbal position control command to introduce how to use CRC16 and CRC32 group pack test.

The gimbal will move to a certain position once the following command is sent:

AA 1A 00 03 00 00 00 00 22 11 ***A2 42*** 0E 00 20 00 30 00 40 00 01 14 ***7B 40 97 BE***

- Figure 35 CRC Parameters -

3.4 CRC Code Sample

The CRC16 used in this protocol can refer to custom_crc16.c, custom_crc16.h.
The CRC32 used in this protocol can refer to custom_crc32.c, custom_crc32.h.

Notes: An executable file that is compiled with this code can use a -v parameter to produce the corresponding CRC pattern and the CRC16 and CRC32 values generated by the command shown in Section 3.3.

```
PS E:\work> .\custom_crc16.exe -v
width            = 16
poly             = 0x8005
reflect_in       = true
xor_in           = 0xc55c
reflect_out      = true
xor_out          = 0x0000
crc_mask         = 0xffff
msb_mask         = 0x8000
0x42a2
PS E:\work> .\custom_crc32.exe -v
width            = 32
poly             = 0x04c11db7
reflect_in       = true
xor_in           = 0xc55c0000
reflect_out      = true
xor_out          = 0x00000000
crc_mask         = 0xffffffff
msb_mask         = 0x80000000
0xbe97407b
PS E:\work>
```

- Figure 37 CRC Code Sample -