

UniRC 7 Series & UniRC 10 Pro Series

SDK Guide

Version: v1.0.5

Chapter 1. Protocol Description

1.Protocol Format Description

Field	Index	Byte Size	Description
STX	0	2	0x5566 – Start-of-frame marker
CTRL	2	1	0: need_ack – whether this packet requires acknowledgment. 1: ack_pack – whether this packet is an ACK packet. 2-7: Reserved.
Data_len	3	2	Length of the data field in bytes (little-endian)
SEQ	5	2	Frame sequence number, range 0–65535 (little-endian)
CMD_ID	7	1	Command ID
DATA	8	Data_len	Data payload
CRC16		2	CRC16 checksum over the entire packet (little-endian)

Update Log

Version	Date	Changes	Remarks
V1.0.0			
V1.0.1	2024-10-31	1. Modified commands 0x16 and 0x17: baud rate of data link 2 air unit. 2. Modified command 0x43: added uplink/downlink data volume for data link 2. 3. Added 0x44: retrieve video transmission link information. 4. Added channel mapping type definition table.	
V1.0.2	2025-02-14	Modified signal strength and quality definitions in 0x44.	
V1.0.3	2025-09-06	1. Added command for retrieving remote controller multiple links status. 2. Added recommended signal quality grading for video transmission.	
V1.0.4	2026-03-11	1. Added 0x4F for system status reporting. 2. Enhanced 0x4E to retrieve remote controller multiple links status. 3. Modified 0x44 signal strength – redefined to 0–100.	1. 0x44 currently only applies to the UniRC 7 series. For UniRC 10 Pro series, refer to the command details. 2. 0x4F is currently only supported on UniRC 7 series.
V1.0.5	2026-07-08	1. Merged SDK for UniRC 10 Pro and UniRC 7 series.	

Chapter 2. Protocol Commands

0x40: Get Remote Controller Hardware ID

CMD_ID:0x40 - Hardware ID			
Send Format			
Index	Data Type	Data Name	Description
ACK Format			
	UInt8_t	hardware_id[12]	Hardware ID string (10-digit ID)

eg:

Send (HEX): 55 66 01 00 00 00 00 40 81 9c

Reply (HEX): 55 66 02 0C 00 09 00 40 36 38 30 31 31 33 30 31 31 31 00 00 7b 8b

0x16: Get System Settings

CMD_ID:0x16 - Get System Settings			
Send Format			
Index	Data Type	Data Name	Description
ACK Format			
	UInt8_t	match	Binding status – 0: start binding; 1–2: binding in progress; 3: binding complete.
	UInt8_t	Com1_baud_type	Air unit UART1 baud rate: 1 = BAUD_9600 3 = BAUD_57600 5 = BAUD_115200
	UInt8_t	Joy_type	Stick mode – 0 to 3 correspond to: Mode 1 (Japan), Mode 2 (USA), Mode 3 (China), Custom
	UInt8_t	Rc_bat	Remote controller battery voltage ×10 (V)

	UInt8_t	Com2_baud_type	Air unit UART2 baud rate: 1 = BAUD_9600 3 = BAUD_57600 5 = BAUD_115200
--	---------	----------------	---

0x17: System Settings

CMD_ID:0x17 - System Settings			
Send Format			
Index	Data Type	Data Name	Description
	UInt8_t	match	Binding command value: 1 = start binding, 0 = stop binding. Set to 1 when enabled, 0 when disabled.
	UInt8_t	Com1_Baud_type	Air unit UART1 baud rate: 1 = BAUD_9600 3 = BAUD_57600 5 = BAUD_115200
	UInt8_t	Joy_type	Stick mode – 0 to 3 correspond to: Mode 1 (Japan), Mode 2 (USA), Mode 3 (China), Custom
	UInt8_t	reserved	
	UInt8_t	Com2_Baud_type	Air unit UART2 baud rate: 1 = BAUD_9600 3 = BAUD_57600 5 = BAUD_115200
ACK Format			
	int8_t	sta	Status: 1 = success; negative values indicate errors.

0x42: RC Channel Data

CMD_ID:0x42 - RC Channel Data			
Send Format			
Index	Data Type	Data Name	Description
	UInt8_t	freq	Output frequency: 0 = disabled 1 = 2Hz 2 = 4Hz 3 = 5Hz 4 = 10Hz 5 = 20Hz 6 = 50Hz 7 = 100Hz
ACK Format			
1	int16_t	CH1	Each channel occupies two bytes (default range: 1050–1950).
2	int16_t	CH2	
3	int16_t	CH3	
.....	int16_t		
16	int16_t	CH16	

eg:

Send (HEX): 55 66 01 01 00 00 00 42 02 B5 C0 (4HZ) – must be sent three times consecutively

Send (HEX): 55 66 01 01 00 00 00 42 00 F7 E0 (Disabled) – must be sent three times consecutively

Reply (HEX) (2HZ): 55 66 00 20 00 99 00 42 DC 05 DC 00 DC 05 DC 05 DC 05 DC
05 DC 05 DC 05 DC 05 DC 05 1A 04 DC 05 DC 05 1A 04 1A 04 FF 88

0x43: Obtain RC Link Information

CMD_ID:0x43 - Obtain RC Link Information			
Send Format			
Index	Data Type	Data Name	Description
ACK Format			
	uint16_t	freq	Frequency
	uint8_t	pack_loss_rate	Packet loss rate
	uint16_t	real_pack	Valid packets
	uint16_t	real_pack_rate	Valid packets rate
	uint32_t	data_up	Data link uplink data rate (bytes/s)
	uint32_t	data_down	Data link downlink data rate (bytes/s)
	uint32_t	data_up_2	Data link 2 uplink data rate (bytes/s)
	Uint32_t	Data_down_2	Data link 2 downlink data rate (bytes/s)

eg:

Send (HEX): 55 66 01 00 00 00 00 43 e2 ac

Reply (HEX):

0x44: Obtain Video Transmission Link Information

CMD_ID:0x44 - Obtain Video Transmission Link Information			
Send Format			
Index	Data Type	Data Name	Description
ACK Format			
	uint16_t	video_up	Video uplink bitrate: video_up / 10 Kbps
	uint16_t	video_down	Video downlink bitrate: video_down / 10 Mbps
	uint8_t	channel	Video transmission wireless channel (1–16)
	int16_t	signal_strength	0–100% (5-level grading: 0, 20, 40, 60, 80, 100)
	int8_t	signal_quality	<p>Range: -15 to 30 (no unit). Higher values indicate better signal quality.</p> <p>Grading:</p> <ul style="list-style-type: none">• Strong: > 10• Medium: 5–10• Weak: < 5 <p>(Deprecated - not recommended. Use the field below instead.)</p>
	UInt8_t	signal_quality	<p>Range: 0–100 (no unit). Higher values indicate better signal quality.</p> <p>(New for UniRC 7 series; for UniRC 10 series, the field above is still used.)</p>

eg:

Send (HEX): **55 66 01 00 00 00 00 44 05 dc**

Reply (HEX):

0x47: Obtain Firmware Version

CMD_ID:0x47 - Obtain Firmware Version			
Send Format			
Index	Data Type	Data Name	Description
ACK Format			
	uint32_t	rc_version	Remote controller firmware version
	uint32_t	rf_version	Receiver firmware version
	uint32_t	ground_version	Ground unit video transmission firmware version
	uint32_t	sky_version	Air unit video transmission firmware version

Note: The retrieved version number is a 4-byte hexadecimal value, with the first byte as the low byte and the last byte as the high byte. Ignore the first byte; the remaining three bytes represent the version number. For example, 0x00 0x03 0x05 0x68 corresponds to version 5.3.0. The same logic applies to other version numbers.

eg:

Send (HEX): 55 66 01 00 00 00 00 47 66 ec

Reply (HEX): 55 66 02 10 00 02 00 47 00 03 05 68 07 02 05 69 02 02 00 56 02 02 00 56 6d 21

0x48: Get All Channel Mappings

CMD_ID:0x48 - Get All Channel Mappings			
Send Format			
Index	Data Type	Data Name	Description
ACK Format			
1	UInt8_t	Ch1_type	Mapped physical channel type: 0 - Joystick, wheel, etc. 1 - Button, switch, etc.
1	uint8_t	Ch1_entity_id	Physical channel ID
2	UInt8_t	Ch2_type	Mapped physical channel type: 0 - Joystick, wheel, etc. 1 - Button, switch, etc.
2	uint8_t	Ch2_entity_id	Physical channel ID
3	UInt8_t	Ch3_type	Mapped physical channel type: 0 - Joystick, wheel, etc. 1 - Button, switch, etc.
3	uint8_t	Ch3_entity_id	Physical channel ID
4	UInt8_t	Ch4_type	Mapped physical channel type: 0 - Joystick, wheel, etc. 1 - Button, switch, etc.
4	uint8_t	Ch4_entity_id	Physical channel ID
.....	uint8_t		

eg:

Send (HEX): 55 66 01 00 00 00 00 48 89 1d

Reply (HEX): 55 66 02 20 00 16 00 48 00 00 00 01 00 02 00 03 05 00 05 01 05 02 01 00 01 01 01 02 01 03 00 04 00 05 02 01 02 00 03 00 C1 28

0x49: Get Channel Mapping

CMD_ID:0x49 - Get Channel Mapping			
Send Format			
Index	Data Type	Data Name	Description
	UInt8_t	rc_ch	RC Channel (1-16)
ACK Format			
	UInt8_t	rc_ch	RC Channel (1-16)
	uint8	Type	Mapped physical channel type: 0 - Joystick, wheel, etc. 1 - Button, switch, etc.
	uint8_t	entity_id	Physical channel ID

eg:

Send (HEX): 55 66 01 01 00 00 00 49 02 4F 1C

Reply (HEX): 55 66 02 03 00 17 00 49 02 00 01 33 9F

UniRC 7 Series Channel Mapping Type Definitions

Category	Type	entity_id	Physical switch definitions
Joystick	0	0	J1
	0	1	J2
	0	2	J3
	0	3	J4
	0	8	J5
	0	9	J6
Wheel	0	4	LD1
	0	5	RD1
3-Position Switch	5	0	SA
	5	1	SB
Button	1	0	S1
	1	1	S2
	1	2	S3
	1	3	S4
	1	4	L1
	1	5	L2
	1	6	R1
	1	7	R2
	1	8	R3
	1	9	M1
	1	10	M2
	1	11	M3
	1	12	M4
	1	13	M5
	1	14	M6
Virtual Channel	2	0	NULL
	2	1	RSSI
Not Mapped to Any Physical Channel	3	0	NULL

UniRC 10 Pro Series Channel Mapping Type Definitions

Category	Type	entity_id	Physical switch definitions
Joystick	0	0	J1
	0	1	J2
	0	2	J3
	0	3	J4
	0	4	J5
	0	5	J6
	0	6	J7
	0	7	J8
Wheel	0	10	LD
	0	11	RD
Knob	0	8	LK
	0	9	RK
3-Position Switch	5	0	SA
	5	1	SB
	5	2	SC
	5	3	SD
	5	4	SE
	5	5	SF
Button	1	0	S1
	1	1	S2
	1	2	S3
	1	3	S4
	1	4	L1
	1	5	L2
	1	6	L3
	1	7	R1
	1	8	R2
	1	9	R3
	1	10	M1
	1	11	LS
	1	12	RS
Virtual Channel	2	0	NULL
	2	1	RSSI
Not Mapped to Any Physical Channel	3	0	NULL

0x4A: Set Channel Mapping

CMD_ID:0x4A - Set Channel Mapping			
Send Format			
Index	Data Type	Data Name	Description
	UInt8_t	rc_ch	RC channel (1–16)
	uint8	Type	Mapped physical channel type: 0 - Joystick, wheel, etc. 1 - Button, etc. 5 - 3-Position switch
	uint8_t	entity_id	Physical channel ID
ACK Format			
	UInt8_t	rc_ch	RC channel (1–16)
	int8_t	sta	1 = ok Negative values indicate error codes.

eg:

Send (HEX): 55 66 01 03 00 00 00 4A 02 00 00 4F EB

Reply (HEX): 55 66 02 02 00 18 00 4A 02 01 4C C3

0x4B: Get All Channel Reversals

CMD_ID:0x4B - Get All Channel Reversals			
Send Format			
Index	Data Type	Data Name	Description
ACK Format			
1	int8_t	ch1_reverse	RC channel 1 reverse setting: 1 = normal, -1 = reversed
2	int8_t	Ch2_reverse	RC channel 2 reverse setting: 1 = normal, -1 = reversed
3	int8_t	Ch3_reverse	RC channel 3 reverse setting: 1 = normal, -1 = reversed
4	int8_t	Ch4_reverse	RC channel 4 reverse setting: 1 = normal, -1 = reversed
5	int8_t	Ch5_reverse	RC channel 5 reverse setting: 1 = normal, -1 = reversed
.....	int8_t		

eg:

Send (HEX): 55 66 01 00 00 00 00 4B EA 2D

Reply (HEX): 55 66 02 10 00 19 00 4B 01 FF 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 D1 F9

0x4C: Get Channel Reverse Settings

CMD_ID:0x4C - Get Channel Reverse Settings			
Send Format			
Index	Data Type	Data Name	Description
	UInt8_t	rc_ch	RC channel (1–16)
ACK Format			
	UInt8_t	rc_ch	RC channel (1–16)
	int8_t	reverse	Reverse setting: 1 = normal, -1 = reversed

eg:

Send (HEX): 55 66 01 01 00 00 00 4C 02 BA E3

Reply (HEX): 55 66 02 02 00 1C 00 4C 02 FF 3B F6

0x4D: Set Channel Reverse

CMD_ID:0x4D - Set Channel Reverse			
Send Format			
Index	Data Type	Data Name	Description
	UInt8_t	rc_ch	RC channel (1–16)
	int8_t	reverse	Reverse setting: 1 = normal, -1 = reversed

eg:

Send (HEX): 55 66 01 02 00 00 00 4D 02 FF 0F 86

Reply (HEX): 55 66 02 02 00 1D 00 4D 02 01 8B 65

0x4E: Get Multiple Link Status

CMD_ID:0x4E - Get Multiple Link Status			
Send Format			
Index	Data Type	Data Name	Description
ACK Format			
0	UInt8_t	rc_multi_ctl_mode	Local multiple link mode: 0 = Dual-Control Master 1 = Dual-Control Slave 2 = Relay Master 3 = Relay Slave 4 = Relay 5 = Single-Control
1	UInt8_t	main_vice_link_status	Link status of the other controller: 0 = Disconnected 1 = Connected 2 = Local unit lost control – unable to retrieve status
2	UInt8_t	rc_relay_status	Relay mode channel control status: 0 = Master has control 1 = Slave has control 2 = Local unit lost control – unable to retrieve status
3	UInt8_t	dual_ctl_status	Dual-Control enable switch: 0 = Disabled – master controls all channels 1 = Enabled – slave may control assigned channels

eg:

Send (HEX): 55 66 01 00 00 00 00 4e 4f 7d

Reply (HEX): 55 66 02 04 00 02 00 4E 03 01 00 00 40 AB

0x4F: Get System Status (Active Push on Warning) - Currently Supported Only on UniRC 7 Series

CMD_ID:0x4F - Get System Status			
Send Format			
Index	Data Type	Data Name	Description
	UInt8_t	freq	Output frequency: 0 = disabled, 1 = enabled
ACK Format			
1	uint8_t	G_led_status	Remote controller system LED status: 0 = No status 1 = (Reserved) 2 = (Reserved) 3 = MCU firmware mismatch 4 = Link initialization failed 5 = Ground unit joystick calibration required 6 = (Reserved) 7 = (Reserved) 8 = Ground unit power supply voltage abnormal 9 = Ground unit Bluetooth not detected 10 = Ground unit temperature – Level 1 alert 11 = Ground unit temperature – Level 2 alert 12 = Ground unit temperature – Level 3 alert 13 = Video transmission firmware mismatch 14 = (Reserved) 15 = (Reserved) 16 = (Reserved) 17 = (Reserved) 18 = (Reserved)
2	UInt8_t	s_led_status	Air unit system LED status: 0 = No status 1 = Voltage alert (input voltage below 12V) 2 = Air unit temperature – Level 1 alert 3 = Air unit temperature – Level 2 alert 4 = Air unit temperature – Level 3 alert

eg:

Enable active push (HEX): 55 66 01 01 00 00 00 4f 01 8a 86

Enable active push reply (HEX): 55 66 02 01 00 2B 00 4F 01 59 77

Disable active push (HEX): 55 66 01 01 00 00 00 4f 00 ab 96

Disable active push reply (HEX): 55 66 02 01 00 AF 00 4F 01 90 60

Active system status push (HEX): 55 66 02 02 00 E7 00 4F 03 00 03 51

Chapter 3. Communication Interface

UniRC 7	UniRC 10
1. UART / Serial: /dev/ttyHS3 Baud rate: 115200 2. Bluetooth 3. Type-C (USB virtual serial port (external interface)) 4. UDP (Server IP: 192.168.144.20, Port: 19856)	1. UART / Serial 1: /dev/ttyHS3 Baud rate: 115200 2. UART / Serial 2: /dev/ttyHS0 Baud rate: 115200 3. Bluetooth 4. Type-C (USB virtual serial port (external interface)) 5. UDP (Server IP: 192.168.144.20, Port: 19856)

Note: The client port number must avoid using 19856 to prevent conflicts with the server port.

Note:

When using the serial interface, the ground station APP identifies and adapts to different remote controllers based on the Android system model name (ro.product.model):

UniRC 7 Standard Edition: Standard_94

UniRC 7 Pro Edition: Pro_94

UniRC 10 Pro: Standard-10inch_A2

Two data link interfaces and one SDK interface are available. The following combinations are selectable:

(For UniRC 7, you can switch the data link and SDK interfaces using the UniGCS app.)

Combination	Data Link 1 Interface	Data Link 2 Interface	SDK Interface
1	Serial / Bluetooth	Bluetooth / Serial	Serial / Bluetooth / Type-C
2	Serial / Type-C	Type-C / Serial	Serial / Bluetooth / Type-C
3	UDP / Bluetooth	Bluetooth / UDP	UDP / Bluetooth / Type-C
4	UDP / Type-C	Type-C / UDP	UDP / Bluetooth / Type-C
5	Bluetooth / Type-C	Type-C / Bluetooth	UDP / Serial / Bluetooth / Type-C

(For UniRC 10 Pro, you can switch the data link and SDK interfaces using the UniGCS app.)

Combination	Data Link 1 Interface	Data Link 2 Interface	SDK Interface
1	Bluetooth	Type- C	Bluetooth / Type- C / UDP / Serial 1
2	Bluetooth	UDP	Bluetooth / Type- C / UDP
3	Bluetooth	Serial 1	Bluetooth / Type- C / Serial 1
4	Type- C	Bluetooth	Bluetooth / Type- C / UDP / Serial 1
5	Type- C	UDP	Bluetooth / Type- C / UDP / Serial 2
6	Type- C	Serial 1	Bluetooth / Type- C / Serial 1 / Serial 2
7	Type- C	Serial 2	Type- C / UDP / Serial 1 / Serial 2
8	UDP	Bluetooth	Bluetooth / Type- C / UDP
9	UDP	Type- C	Bluetooth / Type- C / UDP / Serial 2
10	UDP	Serial 2	Type- C / UDP / Serial 2
11	Serial 1	Bluetooth	Bluetooth / Type- C / Serial 1
12	Serial 1	Type- C	Bluetooth / Type- C / Serial 1 / Serial 2
13	Serial 1	Serial 2	Type- C / Serial 1 / Serial 2
14	Serial 2	Type- C	Type- C / UDP / Serial 1 / Serial 2
15	Serial 2	UDP	Type- C / UDP / Serial 2
16	Serial 2	Serial 1	Type- C / Serial 1 / Serial 2

Chapter 4. CRC16 Checksum Code

```
const uint16_t crc16_tab[256];
```

```
/******
```

CRC16 Coding & Decoding $G(X) = X^{16} + X^{12} + X^5 + 1$

```
*****/
```

```
uint16_t CRC16_cal(uint8_t *ptr, uint32_t len, uint16_t crc_init)
```

```
{
```

```
    uint16_t crc, oldcrc16;
```

```
    uint8_t temp;
```

```

crc = crc_init;
while (len--!=0)
{
    temp=(crc>>8)&0xff;
    oldcrc16=crc16_tab[*ptr^temp];
    crc=(crc<<8)^oldcrc16;
    ptr++;

}
//crc=~crc;    ///??
return(crc);
}

```

```

uint8_t crc_check_16bites(uint8_t* pbuf, uint32_t len,uint32_t* p_result)
{
    uint16_t crc_result = 0;
    crc_result= CRC16_cal(pbuf,len, 0);
    *p_result = crc_result;

    return 2;
}

```

```

const uint16_t crc16_tab[256]= {0x00,0x1021,0x2042,0x3063,0x4084,0x50a5,0x60c6,0x70e7,
                                0x8108,0x9129,0xa14a,0xb16b,0xc18c,0xd1ad,0xe1ce,0xf1ef,
                                0x1231,0x2210,0x3273,0x2252,0x52b5,0x4294,0x72f7,0x62d6,
                                0x9339,0x8318,0xb37b,0xa35a,0xd3bd,0xc39c,0xf3ff,0xe3de,
                                0x2462,0x3443,0x420,0x1401,0x64e6,0x74c7,0x44a4,0x5485,
                                0xa56a,0xb54b,0x8528,0x9509,0xe5ee,0xf5cf,0xc5ac,0xd58d,
                                0x3653,0x2672,0x1611,0x630,0x76d7,0x66f6,0x5695,0x46b4,
                                0xb75b,0xa77a,0x9719,0x8738,0xf7df,0xe7fe,0xd79d,0xc7bc,

```

0x48c4,0x58e5,0x6886,0x78a7,0x840,0x1861,0x2802,0x3823,
0xc9cc,0xd9ed,0xe98e,0xf9af,0x8948,0x9969,0xa90a,0xb92b,
0x5af5,0x4ad4,0x7ab7,0x6a96,0x1a71,0xa50,0x3a33,0x2a12,
0xdbfd,0xcbdc,0xfbbf,0xeb9e,0x9b79,0x8b58,0xbb3b,0xab1a,
0x6ca6,0x7c87,0x4ce4,0x5cc5,0x2c22,0x3c03,0xc60,0x1c41,
0xedae,0xfd8f,0xcdec,0xddcd,0xad2a,0xbd0b,0x8d68,0x9d49,
0x7e97,0x6eb6,0x5ed5,0x4ef4,0x3e13,0x2e32,0x1e51,0xe70,
0xff9f,0xefbe,0xdfdd,0xcffc,0xbf1b,0xaf3a,0x9f59,0x8f78,
0x9188,0x81a9,0xb1ca,0xa1eb,0xd10c,0xc12d,0xf14e,0xe16f,
0x1080,0xa1,0x30c2,0x20e3,0x5004,0x4025,0x7046,0x6067,
0x83b9,0x9398,0xa3fb,0xb3da,0xc33d,0xd31c,0xe37f,0xf35e,
0x2b1,0x1290,0x22f3,0x32d2,0x4235,0x5214,0x6277,0x7256,
0xb5ea,0xa5cb,0x95a8,0x8589,0xf56e,0xe54f,0xd52c,0xc50d,
0x34e2,0x24c3,0x14a0,0x481,0x7466,0x6447,0x5424,0x4405,
0xa7db,0xb7fa,0x8799,0x97b8,0xe75f,0xf77e,0xc71d,0xd73c,
0x26d3,0x36f2,0x691,0x16b0,0x6657,0x7676,0x4615,0x5634,
0xd94c,0xc96d,0xf90e,0xe92f,0x99c8,0x89e9,0xb98a,0xa9ab,
0x5844,0x4865,0x7806,0x6827,0x18c0,0x8e1,0x3882,0x28a3,
0xcb7d,0xdb5c,0xeb3f,0xfb1e,0x8bf9,0x9bd8,0xabbb,0xbb9a,
0x4a75,0x5a54,0x6a37,0x7a16,0xaf1,0x1ad0,0x2ab3,0x3a92,
0xfd2e,0xed0f,0xdd6c,0xcd4d,0xbdaa,0xad8b,0x9de8,0x8dc9,
0x7c26,0x6c07,0x5c64,0x4c45,0x3ca2,0x2c83,0x1ce0,0xcc1,
0xef1f,0xff3e,0xcf5d,0xdf7c,0xaf9b,0xbfba,0x8fd9,0x9ff8,
0x6e17,0x7e36,0x4e55,0x5e74,0x2e93,0x3eb2,0xed1,0x1ef0
};