Thermal Module

Protocol Guide

Table of Contents

1.Function description

Function	Function description
Upload	1. Supports upgrading through serial port and tool.
Image	1. RAW and YUV data splicing output by row, and only SP422 format is supported.
	2. Image parameter adjustment, including brightness, sharpness, contrast, auto gain, detail enhancement, pseudo colors, 2D NR, 3D NR, scene mode and WDR.
	3. Supports shutter control, including setting shutter mode (auto/manual) and shutter time interval.
Device	1. You can get the device firmware version and device SN.
	2. Supports heartbeat detection.
	3. Supports restart through software.
	4. Supports UNC (Nonuniformity Correction).
	5. Supports SFFC.
	6. Supports correction of bad pixels and columns.
	7. Get chamber temperature and detector temperature.
	8. Get and get temperature measurement information, and secondary calibration of temperature measurement .

2.Abbreviation

Abbreviation	Function
SPI	Serial Peripheral Interface, a high-speed, full duplex, and synchronous communication bus.
RAW	Original image file.
YUV	A coding method for color.
SP422	A storage format for YUV422 sampling, where adjacent Y shares its adjacent Cb (U) and Cr (V).
NUC	Nonuniformity Correction.
SFFC	Senior Flat Field Correction
CCITT	International Telegraph and Telephone Consultative Committee. CCITT-16 is used in this document. The algorithm of checksun: Based on all incoming and outgoing messages, calculates two cyclic redundancy check (CRC) through CCITT-16 which is initialized to be 0. The format: (Polynomial = x ^ 16 + x ^ 12 + X ^ 5 + 1). CRC1 only uses the first 6 bytes of the data packet for calculation. CRC2 uses all previous bytes in the data packet (Bytes 0 to N) for calculation.
CRC	Cyclic Redundancy Check.
LSB	Least Significant Bit.
MSB	Most Significant Bit.
2D de-noise	2D noise reduction.
3D de-noise	3D noise reduction.
SW version	Software Version.
FW version	Firmware version.
Camera serial number	Serial number of thermal module.
Sensor serial number	Serial number of the sensor.

3.Communication Protocol

3.1 Overview

The serial communication protocol is responsible for the communication between main and sub devices, and the thermal module communication protocol consists of the basic protocol, the extended protocol, and the temperature monitoring protocol, completing the data transmission.

3.2 Serial configuration

Default configuration of the serial port (Note: For each byte of information, transmit the lowest bit first)

Configuration range	Value
Baud rate	115200
Data bit	8
Stop bit	1
Odd-even examination	None

3.3 Instruction Format of Basic Protocol

The specific content format of the basic protocol and the protocol package of the thermal module is as follows:

BYTE #	Upper Byte	Comments
1	Process Code	0x6E
2	Status	The return value of packet status, legal: 0x00, illegal: non-zero
3	Reserved	Reserved (0x00)
4	Function	Command code (0x00 corresponds to the expansion protocol, 0x50 corresponds to the temperature monitoring protocol)

BYTE #	Upper Byte	Comments
5	Byte Count (MSB)	The high byte of Argument bytes in data packet
6	Byte Count (LSB)	The low byte of Argument bytes in data packet
7	CRC1 (MSB)	Adopts the CCITT check method to check the 6 bytes before CRC1, the high byte of the checksum.
8	CRC1 (LSB)	Adopts the CCITT check method to check the 6 bytes before CRC1, the low byte of the checksum.
N	Argument	The content data of each data packet. Different command codes have different data length.
N+1	CRC2 (MSB)	Adopts the CCITT check method to check all bytes before CRC2, the high byte of the checksum.
N+2	CRC2 (LSB)	Adopts the CCITT check method to check all bytes before CRC2, the low byte of the checksum.

3.4 Format of Expansion Protocol Instruction

The protocol package content format of the thermal module expansion protocol is as follows:

BYTE #	Upper Byte	Comments
1	Process Code	ox6E
2	Status	The return value of packet status, legal: oxoo, illegal: non-zero
3	Reserved	Reserved (oxoo)
4	Function	oxoo
5	Byte Count (MSB)	The high byte of Argument bytes in data packet
6	Byte Count (LSB)	The low byte of Argument bytes in data packet
7	CRC1 (MSB)	Adopts the CCITT check method to check the 6 bytes before CRC1, the high byte of the checksum.

BYTE #	Upper Byte	Comments
8	CRC1 (LSB)	Adopts the CCITT check method to check the 6 bytes before CRC1, the low byte of the checksum.
BYTE #	Argument	Payload of module communication expansion protocol
9	DHSCP Function	Command code of module communication expansion protocol
N	DHSCP Function	The payload of module communication expansion protocol. Different command codes have different data length.
N+1	CRC2 (MSB)	Adopts the CCITT check method to check all bytes before CRC2, the high byte of the checksum.
N+2	CRC2 (LSB)	Adopts the CCITT check method to check all bytes before CRC2, the low byte of the checksum.

3.5 Protocol Package Status

The second byte of the three protocol is the return value of packet status, legal: oxoo, illegal: non-zero. According to the return states, the return types are as following:

Status Byte Value(hex)#	Upper Byte	Comments
0x00	CAM_OK	True
0x02	CAM_NOT_READY	The module is not ready.
0x03	CAM_RANGE_ERROR	Parameter range error
0x04	CAM_CHECKSUM_ERROR	Parity error
0x05	CAM_UNDEFINED_PROCESS_ERROR	Unknown processing code
0x06	CAM_UNDEFINED_FLIR_FUNCTION_ERROR	Undefined command code of module communication protocol
0x07	CAM_TIMEOUT_ERROR	Timeout
0x09	CAM_BYTE_COUNT_ERROR	Byte count error

Status Byte Value(hex)#	Upper Byte	Comments
0x10	CAM_UNDEFINED_FUNCTION_ERROR	Undefined command code of module communication expansion protocol
0x11	CAM_CFG_ERROR	Configuration operation error
0x12	CAM_FLASH_ERROR	Flash operation error
0x13	CAM_UART_ERROR	Serial port operation error
0x14	CAM_ALGORITHM_ERROR	Algorithm operation error
0x15	CAM_MEMORY_ERROR	Memory operation error

4. Picture Parameters Adjustment

4.1 Brightness

Descriptions

Get/Set the image brightness.

Definition

The protocol adopts basic protocol.

Command	Read command code	Byte count	Carried data	Description
Brightness	0x55	Command of getting: 0 (Response: 2)	None	Get the current brightness
		Set command: 2 (Response: 2)	Byte 0: 0x00 Byte 1: brightness level range: 0 to +255	Set the brightness

Sample

1. Get the current brightness. The current brightness is 255 according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x55	0x00 0x00	0x6a 0x85	—	0x00 0x00
Reply	0x6E	0x00	0x00	0x55	0x00 0x02	0x4a 0xc7	0x00 0xff	0x1e 0xf0

2. Set the brightness to 128

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x55	0x00 0x02	0x4a 0xc7	0x00 0x80	0x91 0x88

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Reply	0x6E	0x00	0x00	0x55	0x00 0x02	0x4a 0xc7	0x00 0x80	0x91 0x88

4.2 Sharpness

Descriptions

Get/Set the image brightness.

Definition

The protocol adopts basic protocol.

Command	Read command code		Carried data	Description
Sharpness	0xe3	Command of getting: 0 (Response: 2)	None	Get the current sharpness
		Command of setting: 2 (Response: 2)	Byte 0: 0x00 1: sharpness level range:-20 to +100	Set the sharpness

Sample

1. Get the current sharpness. The current sharpness is 100 according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0xe3	0x00 0x00	0x26 0xda	—	0x00 0x00
Reply	0x6E	0x00	0x00	0xe3	0x00 0x02	0x06 0x98	0x00 0x64	0x2c 0x22

2. Set the sharpness to 40

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0xe3	0x00 0x02	0x06 0x98	0x00 0x28	0xa5 0x6a

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Reply	0x6E	0x00	0x00	0xe3	0x00 0x02	0x06 0x98	0x00 0x28	0xa5 0x6a

4.3 Contrast

Descriptions

Get/Set the image contrast

Definition

The protocol adopts the expansion protocol

Command	Command code (DHSCP)	Byte count	Carried data	Description
Contrast	0x2d	Command of getting: 0 (Response: 2)	None	Get the current contrast
		Command of setting: 2 (Response: 2)	Byte 0: 0x00 1: contrast level range: 0 to +100	Set the contrast

Sample

1. Get the current contrast. The current contrast is 100 according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x01	0xcf 0x9a	0x2d	—	0xf5 0xcf
Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x2d	0x00 0x64	0xe8 0xb5

2. Set the contrast to 50

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x2d	0x00 0x32	0xd2 0x86
Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x2d	0x00 0x32	0xd2 0x86

Note

None

4.4 Histogram Equalization

Descriptions

Get/Set the histogram equalization

Definition

The protocol adopts the extension protocol

Command	Command code (DHSCP)	Byte count	Carried data	Description
Histogram Equalization	0x32	Command of getting: 0 (Response: 2)	None	Get the current histogram equalization
		Command of setting: 2 (Response: 2)	Byte 0 -1: Histogram Equalization range: 0-32	Set the histogram equalization

Sample

1. Get the current histogram equalization The current histogram equalization is 16 according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Comman	0x6E	0x00	0x00	0x00	0x00	0xcf	0x32	—	0x16

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
d					0x01	0x9a			0x11
Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x32	0x00 0x10	0xb9 0xf4

2. Set the histogram equalization to 32

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x32	0x00 0x20	0x8f 0xa7
Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x32	0x00 0x20	0x8f 0xa7

Note

None

4.5 Automatic Gain

Description

Get/Set the auto gain value of the image

Definition

The protocol adopts basic protocol.

Command	Read command code	Byte count	Carried data	Description
Automatic Gain	0x3e	Command of getting: 0 (Response: 2)	None	Get the of current auto gain value
		Command of setting: 2 (Response: 2)	Byte 0-1: Auto gain value Range: 0-255	Set the auto gain value

Sample

1. Get the current auto gain value. The current auto gain value is 255 according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x3e	0x00 0x00	0x01 0x1f	—	0x00 0x00
Reply	0x6E	0x00	0x00	0x3e	0x00 0x02	0x21 0x5d	0x00 0xff	0x1e 0xfo

2. Set the auto gain value to 1

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x3e	0x00 0x02	0x21 0x5d	0x00 0x01	0x10 0x21
Reply	0x6E	0x00	0x00	0x3e	0x00 0x02	0x21 0x5d	0x00 0x01	0x10 0x21

Note

None

4.6 Detail Enhancement

Description

Get/Set the detail enhancement.

Definition

The protocol adopts the expansion protocol.

Command	Command code (DHSCP)			Description
Detail Enhancement	0x2c	Command of getting: 0 (Response: 2)	None	Get the value of detail enhancement
		Command of setting: 2 (Response: 2)	Byte 0-1: detail enhancement Range: 0-128	Set the detail enhancement

Sample

1. Get the current detail enhancement value. The current detail enhancement value is 128 according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x01	0xcf 0x9a	0x2c	—	0xe5 0xee
Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x2c	0x00 0x80	0x62 0x2f

2. Set the detail enhancement to 64.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x2c	0x00 0x40	0xbb 0x63
Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x2c	0x00 0x40	0xbb 0x63

Note

None

4.7 Pseudo Colors

Description

Get/Set the pseudo colors

Definition

The protocol adopts basic protocol.

Command	Read command code	Byte count	Carried data	Description
Pseudo Colors	0x10	Command of getting: 0 (Response: 2)	None	Get the current pseudo color
		Command of setting: 2 (Response: 2)	0x0000-white hot 0x0001-black hot 0x0002-fusion 0x0003-rainbow	Set the pseudo color

Command	Read command code			Description
			0x0004-golden autumn 0x0005-midday 0x0006-iron red 0x0007-amber 0x0008-jade 0x0009-sunset 0x000a-icefire 0x000b-painting 0x000c-pomegranate 0x000d-emerald 0x000f-spring 0x0010-summer 0x0011-autumn 0x0012-winter	

Sample

1. Get the current pseudo color mode. The current pseudo color mode is iron red according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x10	0x00 0x00	0x9c 0xd8	—	0x00 0x00
Reply	0x6E	0x00	0x00	0x10	0x00 0x02	0xbc 0x9a	0x00 0x06	0x60 0xc6

2. Set the pseudo color mode to iron red

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x10	0x00 0x02	0xbc 0x9a	0x00 0x06	0x60 0xc6
Reply	0x6E	0x00	0x00	0x10	0x00 0x02	0xbc 0x9a	0x00 0x06	0x60 0xc6

Note

None

4.8 2D NR

Description

Get/Set the value of 2D NR

Definition

The protocol adopts the expansion protocol.

Command	Command code (DHSCP)	Byte count	Carried data	Description
2D NR 2D: Fault recovery	0x03	Command of getting: 0 (Response: 2)	None	getting the current 2D noise reduction Value
		Command of setting: 2 (Response: 2)	Byte 0-1: the value of 2D NR Range: 0-100	Set the value of 2D NR

Sample

1. Get the current value of 2D NR. The current value of 2D NR is 100 according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x01	0xcf 0x9a	0x03	—	0x30 0x63
Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x03	0x00 0x64	0x75 0x72

2. Set the 2D NR value is 128.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x03	0x00 0x32	0x4f 0x41
Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x03	0x00 0x32	0x4f 0x41

Note

None

4.9 3D NR

Descriptions:

Get/Set the value of 3D NR

Definition:

The protocol adopts the expansion protocol.

Command	Command code (DHSCP)	Byte count	Carried data	Description
3D NR 3D: Fault recovery	0x04	Command of getting: 0 (Response: 2)	None	getting the current 3D noise reduction Value
		Command of setting: 2 (Response: 2)	Byte 0-1: the value of 3D NR Range: 0-100	Set the values of 3D NR

Sample

1. Get the current value of 3D NR. The current value of 3D NR is 100 according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x01	0xcf 0x9a	0x04	—	0x40 0x84

Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x04	0x00 0x64	0xfo 0xe2

2. Set the value of 3D NR to 128

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x04	0x00 0x32	0xca 0xd1
Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x04	0x00 0x32	0xca 0xd1

Note

None

4.10 Wide Dynamic Range

Description

Get/Set the status of Wide Dynamic Range (WDR)

Definition

The protocol adopts the expansion protocol.

Command	Command code (DHSCP)	Byte count	Carried data	Description
Wide dynamic range (WDR)	0x48	Command of getting: 2 (Response: 3)	0x0000-ON/OFF status	Get the current WDR mode status
		Command of setting: 3 (Response: 3)	Enable the current WDR mode Byte 0-1: 0x0000 Byte 2: 0x00, WDR disabled 0x01: WDR enabled	Set the WDR mode

Sample

1. Get the current WDR mode status. The current status is disabled according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x48	0x00 0x0d	0x65 0xa1
Reply	0x6E	0x00	0x00	0x00	0x00 0x04	0x9f 0x3f	0x48	0x00 0x0d 0x00	0x9d 0x03

2. Enable the WDR mode

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x04	0x9f 0x3f	0x48	0x00 0x0d 0x01	0x8d 0x22
Reply	0x6E	0x00	0x00	0x00	0x00 0x04	0x9f 0x3f	0x48	0x00 0x0d 0x01	0x8d 0x22

Note

None

4.11 Picture Freeze

Description

Get/Set the status of image freeze.

Definition

The protocol adopts basic protocol.

Command	Read command code	Byte count	Carried data	Description
Picture Freeze	0x0f	Command of getting: 0 (Response:	None	Get the current status of image freeze

Command	Read command code	Byte count	Carried data	Description
		2)		
		Command of setting: 2 (Response: 2)	Byte 0-1: image freeze enabled	Set image freeze

Sample

1. Get the current status of image freeze. The current status is disabled according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x0f	0x00 0x00	0xf3 0x8a	—	0x00 0x00
Reply	0x6E	0x00	0x00	0x0f	0x00 0x02	0xd3 0xc8	0x00 0x00	0x00 0x00

2. Enable image freeze

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x0f	0x00 0x02	0xd3 0xc8	0x00 0x01	0x10 0x21
Reply	0x6E	0x00	0x00	0x0f	0x00 0x02	0xd3 0xc8	0x00 0x01	0x10 0x21

Note

None

4.12 FFC Mode

Description

Get/Set the FFC mode

Definition

The protocol adopts basic protocol.

Command	Read command code	Byte count	Carried data	Description
Correction Mode	0x0b	Command of getting: 0 (Response: 2)	None	Get Current FFC mode
		Command of setting: 2 (Response: 2)	0x0000-Manual mode 0x0001-Auto mode	Set the FFC mode

Sample

1. Get the current FFC mode. The current mode is Manual according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x0b	0x00 0x00	0x2f 0x4a	—	0x00 0x00
Reply	0x6E	0x00	0x00	0x0b	0x00 0x02	0x0f 0x08	0x00 0x00	0x00 0x00

2. Set the FFC mode to Auto

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x0b	0x00 0x02	0x0f 0x08	0x00 0x01	0x10 0x21
Reply	0x6E	0x00	0x00	0x0b	0x00 0x02	0x0f 0x08	0x00 0x01	0x10 0x21

Note

None

4.13 FFC Cycle

Description

Get/Set the FFC cycle

Definition

The protocol adopts basic protocol.

Command	Read command code		Carried data	Description
FFC cycle	0x0d	Command of getting: 0 (Response: 4)	None	Get the FFC cycle
		Command of setting: 4 (Response: 4)	Byte 0-1: with high gain FFC cycle ×25 unit: second (10–1200) Byte 2-3: with low gain FFC cycle ×25 unit: second (10–1200)	Set the FFC cycle

Sample

1. Get the current FFC cycle. The current cycle is 9000 / 25 = 360 s according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x0d	0x00 0x00	0x9d 0xea	—	0x00 0x00
Reply	0x6E	0x00	0x00	0x0d	0x00 0x04	0xdd 0x6e	0x23 0x28 0x23 0x28	0x75 0x2a

2. Set the current cycle to 9000 / 25 = 360 s

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x0d	0x00 0x04	0xdd 0x6e	0x23 0x28 0x23 0x28	0x75 0x2a
Reply	0x6E	0x00	0x00	0x0d	0x00 0x04	0xdd 0x6e	0x23 0x28 0x23 0x28	0x75 0x2a

Note

1. The FFC cycle is only valid in auto mode.
2. At present, there is no distinction between high gain mode and low gain mode. When issuing protocols, set the same values for both modes.

4.14 Restore Default

Description

Restore Default

Definition:

The protocol adopts basic protocol.

Command	Read command code	Byte count	Carried data	Description
Restore Default	0x03	Command of getting: 0 (Response: 0)	None	Restore Default

Sample

Restore Default

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x03	0x00 0x00	0x86 0xeb	—	0x00 0x00
Reply	0x6E	0x00	0x00	0x03	0x00 0x00	0x86 0xeb	—	0x00 0x00

Note

None

4.15 Scene Mode

Description

Get/Set the scene mode

Definition

The protocol adopts the expansion protocol.

Command	Command code (DHSCP)	Byte count	Carried data	Description
Scene Mode	0x2a	Command of getting: 0 (Response: 2)	None	Get the current Scene mode
		Command of setting: 2 (Response: 2)	0x0000: low dynamic 0x0001: high dynamic 0x0002: self-adaptive dynamic	Set the scene mode

Sample

1. Get the current scene mode. The current mode is high dynamic according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x01	0xcf 0x9a	0x2a	—	0x85 0x28
Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x2a	0x00 0x01	0x51 0x26

2. Set the scene mode to self-adaptive dynamic

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x2a	0x00 0x02	0x61 0x45
Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x2a	0x00 0x02	0x61 0x45

Note

None

4.16 Shutter Frequency

Description

Get/Set the shutter frequency

Definition

The protocol adopts the expansion protocol.

Command	Command code (DHSCP)	Byte count	Carried data	Description
Scene Mode	0x29	Command of getting: 0 (Response: 4)	None	Get the shutter frequency
		Command of setting: 4 (Response: 4)	Byteo-4: shutter frequency	Set the shutter frequency

Sample

1. Get the shutter frequency. The shutter frequency is 777 according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x01	0xcf 0x9a	0x29	—	0xb5 0x4b
Reply	0x6E	0x00	0x00	0x00	0x00 0x05	0x8f 0x1e	0x29	0x00 0x00 0x03 0x09	0x64 0xb2

2. Set the shutter frequency to 100

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x05	0x8f 0x1e	0x29	0x00 0x00 0x00 0x64	0x8c 0xea

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Reply	0x6E	0x00	0x00	0x00	0x00 0x05	0x8f 0x1e	0x29	0x00 0x00 0x00 0x64	0x8c 0xea

Note

None

4.17 Digital Zoom

Description

Digital zoom

Definition

The protocol adopts the expansion protocol.

Command	Command code (DHSCP)	Byte count	Carried data	Description
Digital zoom	0x56	Command of getting: 0 (Response: 1)	None	Get the current multiple
		Command of setting: 1 (Response: 1)	0x01 (Normal) 0x02: 2. 0x04: 4. 0x08: 8.	Set magnification

Sample

Get the current magnification. Get the current magnification according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x01	0xcf 0x9a	0x56	—	0x3a 0x33

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Reply	0x6E	0x00	0x00	0x00	0x00 0x02	0xff 0xf9	0x56	0x02	0x84 0x5b

2. Set the magnification to 2

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x02	0xff 0xf9	0x56	0x02	0x84 0x5b
Reply	0x6E	0x00	0x00	0x00	0x00 0x02	0xff 0xf9	0x56	0x02	0x84 0x5b

Note

None

4.18 Image Flip

Description

Image flip

Definition

The protocol adopts basic protocol.

Command	Command code (DHSCP)	Byte count	Carried data	Description
Image flip	0x11	Command of getting: 0 (Response: 2)	None	Get the current direction
		Command of setting: 2 (Response: 2)	0x00 0x00: normal direction 0x00 0x01: Vertical flip 0x00 0x02: Horizontal flip 0x00 0x03: Mirror	Set the flip direction

Sample

1. Get the current image flip direction. Get the current image flip direction according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x11	0x00 0x00	0xab 0xe8	—	0x00 0x00
Reply	0x6E	0x00	0x00	0x11	0x00 0x02	0x8b 0xaa	0x00 0x01	0x10 0x21

2. Set image flip to vertical flip

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x11	0x00 0x02	0x8b 0xaa	0x00 0x01	0x10 0x21
Reply	0x6E	0x00	0x00	0x11	0x00 0x02	0x8b 0xaa	0x00 0x01	0x10 0x21

Note

None

4.19 Save Configuration

Description

Save the current configuration.

Definition

The protocol adopts basic protocol.

Command	Command code (DHSCP)	Byte count	Carried data	Description
Save the current configuration.	0x01	Command of getting: 0 (Response: 0)	None	Save the current configuration.

Sample

1. Save the current configuration. Save the current configuration according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x01	0x00 0x00	0xe8 0x8b	—	0x00 0x00
Reply	0x6E	0x00	0x00	0x01	0x00 0x00	0xe8 0x8b	—	0x00 0x00

Note

After starting the device, set auto save after configuration to avoid frequent saving.

5.NUC

5.1 Heartbeat Detection

Description

Do heartbeat detection

Definition

The protocol adopts basic protocol.

Command	Read command code	Byte count	Carried data	Description
Heartbeat Detection	0x00	Command: 0 (Response: 0)	None	Do heartbeat detection

Sample

1. Do heartbeat detection

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x00	0xdf 0xbb	—	0x00 0x00
Reply	0x6E	0x00	0x00	0x00	0x00 0x00	0xdf 0xbb	—	0x00 0x00

Note

None

5.2 UNC (Nonuniformity Correction)

Description

Do nonuniformity correction (NUC)

Definition

The protocol adopts the expansion protocol.

Command	Command code (DHSCP)	Byte count	Carried data	Description
UNC (Nonuniformity Correction)	0x01	Command of getting: 0 (Response: 2)	None	Get the current status of NUC Response data: 0x0001: Enable 0x0000: disabled
		Command of setting: 2 (Response: 2)	0x0000: disable NUC 0x0001: Enable NUC 0x0002: Blackbody 2 operation 0x0003: Blackbody 1 operation 0x0004: Save NUC data 0x0005: Restore the NUC parameters to factory default 0x0005: Refresh the NUC parameters in memory	Set NUC status, blackbody operation, save data, restore factory data, refresh data in memory.

Sample

1. Get the current status of NUC. The current status is disabled according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x01	0xcf 0x9a	0x01	—	0x10 0x21
Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x01	0x00 0x00	0x37 0x30

2. Enable UNC

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x01	0x00 0x01	0x27 0x11
Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x01	0x00 0x01	0x27 0x11

Note

The normal collection process of NUC is as follows: collect the data of blackbody 1 (0x0003) towards the high-temperature blackbody → collect the data of blackbody 2 (0x0002) towards the high-temperature blackbody → save the data after the effect is normal (0x0004)

Before saving data, the original NUC data can be restored trough command 0x0007

5.3 SFFC

Descriptions:

Do senior flat field correction (SFFC)

Definition:

The protocol adopts the expansion protocol.

Command	Command code (DHSCP)	Byte count	Carried data	Description
5. SFFC	0x21	Command of getting: 0	None	Get the status of SFFC

Command	Command code (DHSCP)	Byte count	Carried data	Description
		(Response: 2)		Response data: 0x0001: Enable 0x0000: disabled
		Command of setting: 2 (Response: 2)	0x0000: Disabled SFFC 0x0001: Enabled SFFC 0x0002: Collect the data module of SFFC 0x0003: Save the SFFC data	Set the SFFC status, collect data, and save data.

Sample

1. Get the current status of SFFC. The current status is disabled according to the protocol example.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x01	0xcf 0x9a	0x21	—	0x34 0x43
Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x21	0x00 0x00	0xb1 0xf6

2. Set SFFC

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x21	0x00 0x01	0xa1 0xd7
Reply	0x6E	0x00	0x00	0x00	0x00 0x03	0xef 0xd8	0x21	0x00 0x01	0xa1 0xd7

Note

None

5.4 Bad Pixel and Bad Column

Description

Do correction of bad pixels and columns.

Definition

The protocol adopts the expansion protocol.

Command	Command code (DHSCP)	Byte count	Carried data	Description
Bad Pixel and Bad Column	0x02	Command of getting: 4 (Response: 8)	Get the bad column Byte 0-1: 0xfb01 Byte 2-3: the bad column number to be get (1–8)	Get the bad column Response data: Byte 0-1: 0xfb01 Byte 2-3: the total number of configured columns Byte 4-5: get the bad column number (1–8) Byte 6-7: get the coordinate x of the bad column
		Command of getting: 2 (Response: 4)	Get the On/Off status of manual mode Byte 0-1: 0xfc01	Get the On/Off status of manual mode Response data: Byte 0-1: 0xfc01 Byte 2-3: enabled 0x0000 disabled 0x0001 enabled
		Command of getting: 4 (Response: 10)	Get a bad pixel Byte 0-1: 0xfc02 Byte 2-3: the bad pixel number to be get (1–8)	Get a bad pixel Byte 0-1: 0xfc02 Byte 2-3: the total number of configured pixels Byte 4-5: get the bad column number (1-1024) Byte 6-7: get the coordinate x of the bad

Command	Command code (DHSCP)	Byte count	Carried data	Description
				pixel Byte 8-9: get the coordinate y of the bad pixel
		Command of setting: 8 (Response: 8)	Set a bad column Byte 0-1: oxfb02 Byte 2-3: the total number of the configured bad columns Byte 4-5: the number of the configured bad column Byte 6-7: the coordinate x of the configured bad column	Set a bad column
		Command of setting: 4 (Response: 4)	Set the On/Off status of manual mode Byte 0-1: oxfc01 Byte 2-3: the On/Off status to be configured 0x0000 disabled 0x0001 enabled	Set the On/Off status of manual mode
		Command of setting: 10 (Response: 10)	Set a bad pixel Byte 0-1: oxfc02 Byte 2-3: the total number of configured bad pixels Byte 4-5: the number of configured bad pixels	Set up a bad pixel

Command	Command code (DHSCP)	Byte count	Carried data	Description
			Byte 6-7: the coordinate x of the configured bad pixel Byte 8-9: the coordinate y of the configured bad pixel	
		Command of setting: 2 (Response: 2)	Save to FLASH Byte 0-1: oxfc03	Save the bad pixel and bad column to FLASH

Sample

1. Get the bad pixel numbered 1. According to the protocol example, the total number of configured bad pixel is1, the coordinate x of the bad pixel is 50, and the coordinate y of the bad pixel is 100.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x05	0x8f 0x1e	0x02	0xfc 0x02 0x00 0x01	0xea 0xbd
Reply	0x6E	0x00	0x00	0x00	0x00 0x0b	0x6e 0xd0	0x02	0xfc 0x02 0x00 0x01 0x00 0x01 0x00 0x32 0x00 0x64	0x2f 0xc8

2. Set a bad pixel (50, 50)

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x0b	0x6e 0xd0	0x02	0xfc 0x02 0x00 0x01 0x00 0x01 0x00 0x32 0x00 0x32	0x15 0xfb
Reply	0x6E	0x00	0x00	0x00	0x00 0x0b	0x6e 0xd0	0x02	0xfc 0x02 0x00 0x01 0x00 0x01 0x00 0x32 0x00 0x32	0x15 0xfb

3. Delete a bad pixel (50, 50)

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	DHSCP Function	DHSCP Argument	CRC2
Command	0x6E	0x00	0x00	0x00	0x00 0x0b	0x6e 0xd0	0x02	0xfc 0x02 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00	0xba 0x1f
Reply	0x6E	0x00	0x00	0x00	0x00 0x0b	0x6e 0xd0	0x02	0xfc 0x02 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00	0xba 0x1f

Note

1. The maximum number of bad columns can be set 8, and the maximum number of bad

pixels can be set 1024

2. When getting bad pixels (bad columns), you can get the information of bad pixel (bad column) 1 first. Based on the total number returned, the protocol can be called multiple times to get the information of each bad pixel (bad column).
3. When setting multiple bad pixels (bad columns), the protocol needs to be called multiple times to set the coordinates according to their numbers. For example, if you want to set two bad pixels (10,10) and (20,20), you need to issue two protocols, and enter 2 for total number. The coordinates of bad pixel 1 is (10,10), and the coordinates of bad pixel 2 is (20,20).
4. Deleting bad pixels is to set the pixel number and coordinates to zero, and the total number of bad pixels will change accordingly. The same applies to bad columns.

5.5 Detector Width and Height Query

Description

There is only one sensor (640 × 512) available at present.

5.6 FFC

Description

Flat field correction

Definition

The protocol adopts basic protocol.

Command	Read command code	Byte count	Carried data	Description
Flat field correction	0x0c	Set command: 0 (Response: 0)	—	Flat field correction

Sample

Do flat field correction once.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x0c	0x00 0x00	0xaa 0xda	—	0x00 0x00

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Reply	0x6E	0x00	0x00	0x0C	0x00 0x00	0xaa 0xda	—	0x00 0x00

Note

None

5.7 Get FPA temperature

Description

Get FPA temperature

Definition

The protocol adopts basic protocol.

Command	Read command code	Byte count	Carried data	Description
Get FPA temperature	0x20	Command of getting: 2 (Response: 2)	Byte 0-1: 0x0000	Get FPA temperature Response data: Byte 0-1: FPA temperature x 10 Unit (° C)

Sample

Get FPA temperature once, 56/10=25.6° C

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x20	0x00 0x02	0x79 0x3f	0x00 0x00	0x00 0x00
Reply	0x6E	0x00	0x00	0x20	0x00 0x02	0x79 0x3f	0x01 0x00	0x33 0x31

Note

None

5.8 Device Model

Description

Get the device model

Definition

The protocol adopts basic protocol.

Command	Read command code	Byte count	Carried data	Description
Information of Software version	0x66	Command of getting: 0 (Response: 32)	None	Get the current device model

Sample

1. Get the software version of the current module

According to the protocol example, the model of the current software is NYX6403000SFPM1.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x66	0x00 0x00	0xf6 0x70	—	0x00 0x00
Reply	0x6E	0x00	0x00	0x66	0x00 0x20	0xd2 0x12	0x4e 0x59 0x58 0x36 0x34 0x30 0x33 0x30 0x30 0x30 0x53 0x46 0x50 0x4d 0x31 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00	0x0d 0x6b

Note

1. Argument: Byte0-31: Device Model

2. Data parsing: The protocol gets hexadecimal numbers, which is the converted values of characters through ASCII code. Based on the ASCII code table, parse the software version information according to the corresponding characters.

5.9 Serial Number

Description

Get device SN.

Definition

The protocol adopts basic protocol.

Command	Read command code	Byte count	Carried data	Description
Serial number	0x04	Command of getting: 0 (Response: 64)	None	Get the device SN info.

Sample

1. Get the device SN.

According to the protocol example, the model of the current software is abcdef.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x04	0x00 0x00	0x03 0x7b	—	0x00 0x00
Reply	0x6E	0x00	0x00	0x04	0x00 0x40	0x4b 0xbf	0x61 0x62 0x63 0x64 0x65 0x66 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00	0x; 0x

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
							0x00 0x00 0x00 0x00 0x00, 0x000x00 0x00, 0x000x00 0x00, 0x000x00 ... ... 0x00, 0x000x00 0x00, 0x000x00	

Note

1. Argument: Byte0-31: Device serial number.

2. Data parsing: The protocol gets hexadecimal numbers, which is the converted values of characters through ASCII code. Based on the ASCII code table, parse the software version information according to the corresponding characters.

3. Byte32-63: the numbers of other device, and the specific verification code cannot be provided. You need to get the first 32 bytes for the device serial number.

5.10 Software version

Description

Get device version information.

Definition

The protocol adopts basic protocol.

Command	Read command code	Byte count	Carried data	Description
Information of Software version	0x05	Command of getting: 0 (Response: 64)	None	Get the software version of the current module

Sample

1. Get the software version of the current module

According to the protocol example, the current software version is DG617M00MSR01F20220110.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x05	0x00 0x00	0x34 0x4b	—	0x00 0x00
Reply	0x6E	0x00	0x00	0x05	0x00 0x40	0x7c 0x8f	0x44 0x47 0x36 0x31 0x37 0x4d 0x30 0x30 0x4d 0x53 0x52 0x30 0x31 0x46 0x32 0x30 0x32 0x32 0x30 0x31 0x31 0x30 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x30 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00	0xbc 0xb9

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
							0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00	

Note

1. Argument: Byte0-31: SW version (the software version of the current module, highlight in yellow in the table) Byte32-63: FW version (firmware version, it is 0 currently)

2. Data parsing: The protocol gets hexadecimal numbers, which is the converted values of characters through ASCII code. Based on the ASCII code table, parse the software version information according to the corresponding characters.

5.11 Deployment Process

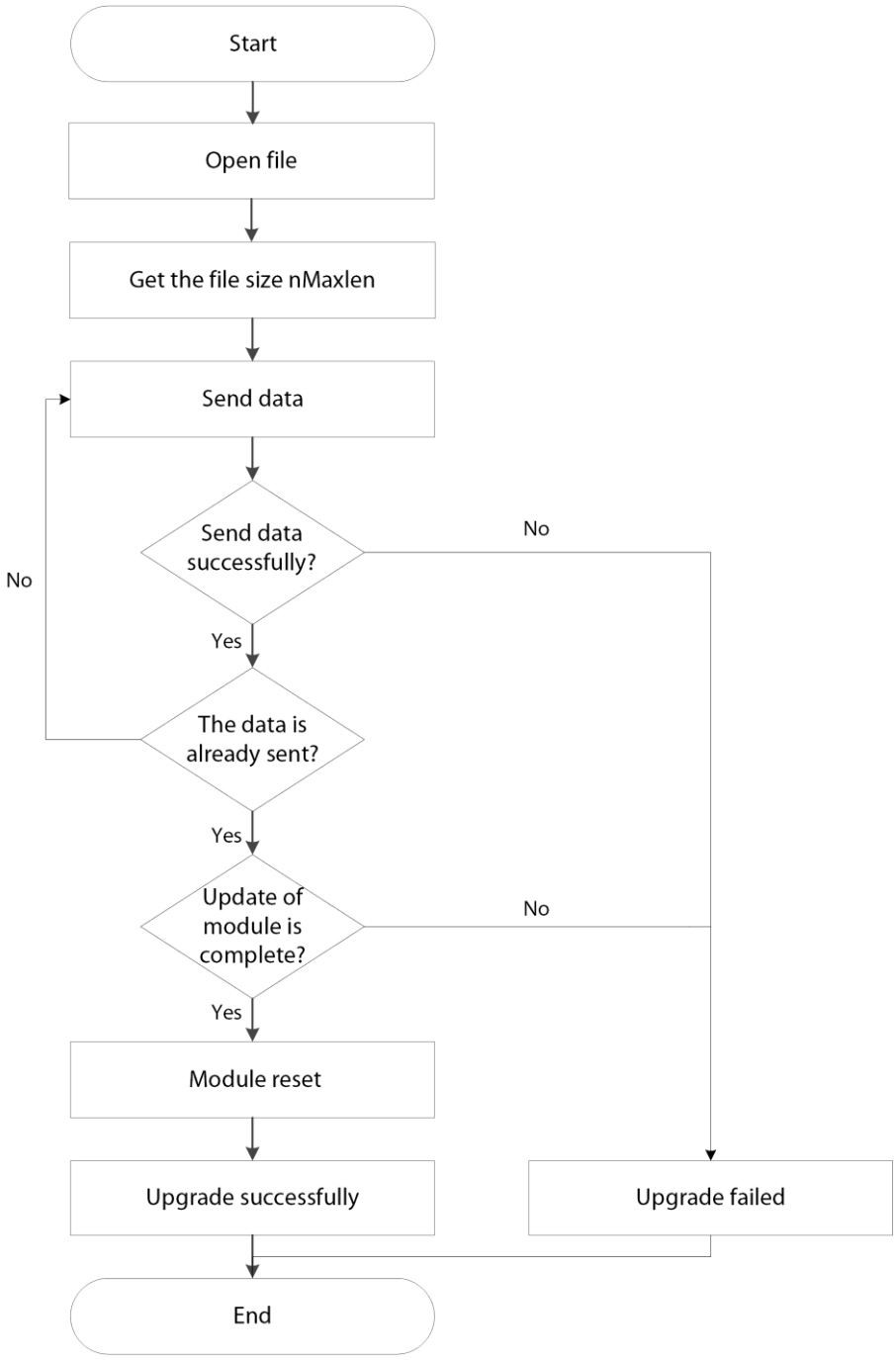


Figure 1 Upgrade process

5.11.1 Process Description

5.11.2 Open file

Parameter Name	Function Prototype	Parameter descriptions	Header file
fopen	FILE* fopen(const char* filename, const char* mode)	Input parameter Filename is the path and name of the file to be opened. Mode represents the mode of opening files, and common ways to open files include r\w\a\rb\wb\ab. Return value Pointer of type FILE, which represents the stream of opened files	stdio.h

5.12 Software Version

Description

Get device version information.

Definition

The protocol adopts basic protocol.

Command	Read command code	Byte count	Carried data	Description
Information of Software version	0x05	Command of getting: 0 (Response: 64)	None	Get the software version of the current module

Sample

1. Get the software version of the current module

According to the protocol example, the current software version is DG617M00MSR01F20220110.

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
Command	0x6E	0x00	0x00	0x05	0x00 0x00	0x34 0x4b	—	0x00 0x00
Reply	0x6E	0x00	0x00	0x05	0x00 0x40	0x7c 0x8f	0x44 0x47 0x36 0x31 0x37 0x4d 0x30 0x30 0x4d 0x53 0x52 0x30 0x31 0x46 0x32 0x30 0x32 0x32 0x30 0x31 0x31 0x30 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x30 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00	0xbc 0xb9

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	Argument	CRC2
							0x00 0x00 0x00 0x00	

Note

1. Argument: Byte0-31: SW version (the software version of the current module, highlight in yellow in the table) Byte32-63: FW version (firmware version, it is 0 currently)

2. Data parsing: The protocol gets hexadecimal numbers, which is the converted values of characters through ASCII code. Based on the ASCII code table, parse the software version information according to the corresponding characters.

5.12.1 Get the size of the data in the file

Parameter Name	Function Prototype	Parameter descriptions	Header file
ftell	long ftell(FILE* stream)	Input parameter Stream is a pointer to a stream (FILE type) of an opened file. Return value The offset of the current file position pointer relative to the starting position of the file (the unit is byte).	stdio.h

5.12.2 Sending Data

Get data through the opened file, 4000 bytes data each time (if the last packet of data is less than 4000 bytes, send the data according to the actual bytes), and then combine it into a protocol and send it to the device through the serial port. Determine whether the device has successfully received the data through the return value.

The fsteek and fread functions are required for reading data from a file. fsteek is used to locate the file pointer to the last read location, while fread is used to read data.

Parameter Name	Function Prototype	Parameter descriptions	Header file
fseek	int fseek(FILE* stream, long offset,	Input parameter	stdio.h

Parameter Name	Function Prototype	Parameter descriptions	Header file
	int origin)	Stream is a pointer to a stream (FILE type) of an opened file. Offset is the amount of offset to be moved. Origin specifies the reference point for moving the position. Return value o indicates success, non-zero values indicate failure.	
fread	size_t fread(void* ptr, size_t size, size_t count, FILE* stream)	Input parameter Ptr is a pointer to a buffer used for storing read data Size is the size of each data item, usually binary open to 1. Count is the number of data items to be read. Stream is a pointer to the stream (FILE type) of an opened file. Return value Indicates the actual amount of the read data. If it is less than count, it indicates an error	

b. After reading the data, it is necessary to assemble a protocol and send it to the device in a fixed format. Protocol filed

Sending data:

Byte	Description	Comments
1	Process field	ox6E
2	State	oxoo
3	Reserved bit	oxoo
4	Protocol byte	oxoo
5	Byte count of the	The high byte of (byte count of data packet +9)

Byte	Description	Comments
	sent data	
6	Byte count of the sent data	The low byte of (byte count of data packet +9)
7	Checksun of CRC	Adopts the CCITT check method to check the front 6 byte of the data header, the high byte of the checksum.
8	Checksun of CRC	Adopts the CCITT check method to check the front 6 byte of the data header, the low byte of the checksum.
9	Sub protocol byte	0x22
10-13	Total size of file	Total size of file, arranging in big endian byte order
14-17	File offset	File offset, initially set to 0, increases by 4000 after each data transmission, arranging in big endian byte order
18-N	Data	data
N+1	CRC2 (MSB)	Adopts the CCITT check method to check all bytes before CRC2, the high byte of the checksum.
N+2	CRC2 (LSB)	Adopts the CCITT check method to check all bytes before CRC2, the low byte of the checksum.

Receiving data:

Byte	Description	Comments
1	Process field	0x6E
2	State	0x00
3	Reserved bit	0x00
4	Protocol byte	0x00
5	Receiving byte count of the data	0x00
6	Receiving byte count of the data	0x1
7	Checksun of CRC	0xCF
8	Checksun of CRC	0x9A

9	Sub protocol byte	0x22
10	CRC2 (MSB)	0x04
11	CRC2 (LSB)	0x20

The CRC calculation method:

```
static unsigned short gCrcTable [256] = {
    0x0000, 0x1021, 0x2042, 0x3063, 0x4084, 0x50a5,
    0x60c6, 0x70e7, 0x8108, 0x9129, 0xa14a, 0xb16b,
    0xc18c, 0xd1ad, 0xe1ce, 0xf1ef, 0x1231, 0x0210,
    0x3273, 0x2252, 0x52b5, 0x4294, 0x72f7, 0x62d6,
    0x9339, 0x8318, 0xb37b, 0xa35a, 0xd3bd, 0xc39c,
    0xf3ff, 0xe3de, 0x2462, 0x3443, 0x0420, 0x1401,
    0x64e6, 0x74c7, 0x44a4, 0x5485, 0xa56a, 0xb54b,
    0x8528, 0x9509, 0xe5ee, 0xf5cf, 0xc5ac, 0xd58d,
    0x3653, 0x2672, 0x1611, 0x0630, 0x76d7, 0x66f6,
    0x5695, 0x46b4, 0xb75b, 0xa77a, 0x9719, 0x8738,
    0xf7df, 0xe7fe, 0xd79d, 0xc7bc, 0x48c4, 0x58e5,
    0x6886, 0x78a7, 0x0840, 0x1861, 0x2802, 0x3823,
    0xc9cc, 0xd9ed, 0xe98e, 0xf9af, 0x8948, 0x9969,
    0xa90a, 0xb92b, 0x5af5, 0x4ad4, 0x7ab7, 0x6a96,
    0x1a71, 0x0a50, 0x3a33, 0x2a12, 0xdbfd, 0xcbdc,
    0xfbbf, 0xeb9e, 0x9b79, 0x8b58, 0xbb3b, 0xab1a,
    0x6ca6, 0x7c87, 0x4ce4, 0x5cc5, 0x2c22, 0x3c03,
    0x0c60, 0x1c41, 0xedae, 0xfd8f, 0xcdec, 0xddcd,
    0xad2a, 0xbd0b, 0x8d68, 0x9d49, 0x7e97, 0x6eb6,
    0x5ed5, 0x4ef4, 0x3e13, 0x2e32, 0x1e51, 0x0e70,
    0xff9f, 0xefbe, 0xdfdd, 0xcffc, 0xbf1b, 0xaf3a,
    0x9f59, 0x8f78, 0x9188, 0x81a9, 0xb1ca, 0xa1eb,
    0xd10c, 0xc12d, 0xf14e, 0xe16f, 0x1080, 0x00a1,
    0x30c2, 0x20e3, 0x5004, 0x4025, 0x7046, 0x6067,
    0x83b9, 0x9398, 0xa3fb, 0xb3da, 0xc33d, 0xd31c,
    0xe37f, 0xf35e, 0x02b1, 0x1290, 0x22f3, 0x32d2,
    0x4235, 0x5214, 0x6277, 0x7256, 0xb5ea, 0xa5cb,
    0x95a8, 0x8589, 0xf56e, 0xe54f, 0xd52c, 0xc50d,
    0x34e2, 0x24c3, 0x14a0, 0x0481, 0x7466, 0x6447,
```

```
        0x5424, 0x4405, 0xa7db, 0xb7fa, 0x8799, 0x97b8,
        0xe75f, 0xf77e, 0xc71d, 0xd73c, 0x26d3, 0x36f2,
        0x0691, 0x16b0, 0x6657, 0x7676, 0x4615, 0x5634,
        0xd94c, 0xc96d, 0xf90e, 0xe92f, 0x99c8, 0x89e9,
        0xb98a, 0xa9ab, 0x5844, 0x4865, 0x7806, 0x6827,
        0x18c0, 0x08e1, 0x3882, 0x28a3, 0xcb7d, 0xdb5c,
        0xeb3f, 0xfb1e, 0x8bf9, 0x9bd8, 0xabbb, 0xbb9a,
        0x4a75, 0x5a54, 0x6a37, 0x7a16, 0x0af1, 0x1ad0,
        0x2ab3, 0x3a92, 0xfd2e, 0xed0f, 0xdd6c, 0xcd4d,
        0xbdaa, 0xad8b, 0x9de8, 0x8dc9, 0x7c26, 0x6c07,
        0x5c64, 0x4c45, 0x3ca2, 0x2c83, 0x1ce0, 0x0cc1,
        0xef1f, 0xff3e, 0xcf5d, 0xdf7c, 0xaf9b, 0xbfba,
        0x8fd9, 0x9ff8, 0x6e17, 0x7e36, 0x4e55, 0x5e74,
        0x2e93, 0x3eb2, 0x0ed1, 0x1ef0
};
/**
* @brief        Calculate CRC checksum
* @param        data        Data
* @param        length    Data length
* @return return description
*             CRC checksum
//
unsigned short crcCalc(char* data, unsigned short length)
{
        unsigned short count;
        unsigned int crc = 0;
        unsigned int final = 0;
        unsigned int temp;
        for (count = 0; count < length; ++count)
        {
                temp = (data[count] ^ (crc >> 8)) & 0xff;
                crc = gCrcTable[temp] ^ (crc << 8);
        }
        return (unsigned short)(crc ^ final);
```

}

5.12.3 Get the Upgrade Progress

Get the upgrade progress after sending the data. When the upgrade progress reaches 100, it indicates that the upgrade completes. If the time exceeds 100 seconds and the progress has not reached 100, it indicates that the upgrade fails.

Sending data:

Byte	Description	Comments
1	Process field	0x6E
2	State	0x00
3	Reserved bit	0x00
4	Protocol byte	0x00
5	Byte count of the data	0x00
6	Byte count of the data	0x01
7	Checksun of CRC	0xCF
8	Checksun of CRC	0x9A
9	Sub protocol byte	0x2B
10	CRC2 (MSB)	0x95
11	CRC2 (LSB)	0x09

Receiving data:

Byte	Description	Comments
1	Process field	0x6E
2	State	0x00
3	Reserved bit	0x00
4	Protocol byte	0x00
5	Byte count of the data	0x00

Byte	Description	Comments
6	Byte count of the data	0x03
7	Checksun of CRC	0xEF
8	Checksun of CRC	0xD8
9	Sub protocol byte	0x2B
10	Upgrade progress	0x00
11	Upgrade progress	0x??, the maximum is 0x64 (decimal is 100)
12	CRC2 (MSB)	0x??, when the upgrade progress is 100, the value is 0x5A
13	CRC2 (LSB)	0x??, when the upgrade progress is 100, the value is 0x5

5.12.4 Reset the Module

After the upgrade completes, reset the module with the following protocol:

Sending data:

Byte	Description	Comments
1	Process field	0x6E
2	State	0x00
3	Reserved bit	0x00
4	Protocol byte	0x02
5	Byte count of the data	0x00
6	Byte count of the data	0x01
7	Checksun of CRC	0xA1
8	Checksun of CRC	0xFA
9	Sub protocol byte	0x00
10	CRC2 (MSB)	0x00
11	CRC2 (LSB)	0x00

Receiving data:

Byte	Description	Comments
1	Process field	0x6E
2	State	0x00
3	Reserved bit	0x00
4	Protocol byte	0x02
5	Byte count of the data	0x00
6	Byte count of the data	0x00
7	Checksun of CRC	0xB1
8	Checksun of CRC	0xDB
9	CRC2 (MSB)	0x00
10	CRC2 (LSB)	0x00

6. Temperature Measurement

6.1 Get/Set the basic temperature parameters

Description

Get/Set the basic temperature parameters (relative humidity, atmospheric temperature, radiation coefficient, relative distance, reflection temperature)

Definition

The protocol adopts the general protocol

Command	Command code (scp)	Byte count of payload	Payload content	Description
Get basic temperature parameters	0x87	Get: 1 Response: 30	Byte 0: 0x04	Get basic temperature parameters Response data: Byte 0-1: 0x04 0x00

Command	Command code (scp)	Byte count of payload	Payload content	Description
				Byte 2- Byte 21: DH_RadCommonTemparam Byte 22-29: 0x00
Set basic temperature parameters	0x88	Set: 21 Response: 2	Byte 0: 0x04 Byte 1-20: DH_RadCommonTemparam	Set basic temperature parameters Response data: Byte 0-1: 0x04 0x00

//The structure of temperature parameters

```
typedef struct DH_RadCommonTemparam
{
    unsigned int relativeHumidity;          //Relative humidity percentage
represents 0%~100%
    float atmosphericTemperature;           //Atmospheric temperature, accurate
to °,Range -40.00—100.00
    float objectEmissivity;                 //Target radiation coefficient 0.5—1.0001
    float objectDistance;                   //Distance, with an accuracy of 0.1
meters ,Range 0-100
    float reflectedTemperature;             //Reflection temperature ,Range
-40.00—550.00
}DH_RadCommonTemparam;
```

Sample

Get basic temperature parameters

Relative humidity 0

Atmospheric temperature 25.00

Radiation coefficient 0.97

Relative distance 2.00

Reflection temperature 25.00

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x50	0x00 0x02	0xa1 0x37	0x87	0x04	0xc2 0x8b
Reply	0x6E	0x00	0x00	0x50	0x00 0x1f	0x62 0xab	0x87	0x04 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0xc8 0x41 0xec 0x51 0x78 0x3f 0x00 0x00 0x00 0x40 0x00 0x00 0xc8 0x41 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00	0xc0 0xde

Set basic temperature parameters

Relative humidity 0

Atmospheric temperature 25.00

Radiation coefficient 0.98

Relative distance 2.00

Reflection temperature 25.00

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x50	0x00 0x16	0xf3 0x82	0x88	0x04 0x00 0x00 0x00	0x83 0x56

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
								0x00 0x00 0x00 0xc8 0x41 0x48 0xe1 0x7a 0x3f 0x00 0x00 0x00 0x40 0x00 0x00 0xc8 0x41	
Reply	0x6E	0x00	0x00	0x50	0x00 0x03	0xb1 0x16	0x88	0x04 0x00	0x5e 0x3f

6.2 Get the Maximum, Minimum, and Average Temperature of the Rules

Command	Command code (scp)	Byte count of payload	Payload content	Description
Get temperature and coordinates	0x5b	Get: 9 Response: 391	Byte 0: 0x04 Byte 1-4: the total number of the rules Byte 5-8: Radiation coefficient	Get the maximum temperature, minimum temperature, center point temperature, and coordinates Response data: Byte 0: 0x04 (SETCONFIG) Byte 1: 0x00 (00: success FF: failure) Byte 2-5: the total number of the rules ruleIdNum Byte 6-390: TherRadRuleTempInfo【12】

/ / / Get all the structure of the temperature rules information 32 byte

typedef struct TherRadRuleTempInfo

{

unsigned int ruleId; //Configuration item number

float temperatureAve; //Average temperature

float temperatureMax; //Maximum temperature

float temperatureMin; //Minimum temperature

float bodyTempMax; //The maximum temperature in the body

unsigned int res[3];

}TherRadRuleTempInfo;

Sample

Get the maximum temperature, minimum temperature, and average temperature of the 2 rules radiation coefficient 0.97

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x50	0x00 0x0a	0x20 0x3f	0x5b	0x04 0x02 0x00 0x00 0x00 0xec 0x51 0x78 0x3f	0xea 0x2b
Reply	0x6E	0x00	0x00	0x50	0x01 0x87	0x53 0x2b	0x5b	0x04 0x00 0x02 0x00 0x00 0x00 ... ... (The structure data of TherRadRuleTempInfo)	0xd4 0x26

6.3 Get the temperature and coordinates of the Center Point

Description

Get center point temperature and coordinates

Definition

Command	Command code (scp)	Byte count of payload	Payload content	Description
Get temperature and coordinates	0x24	Get: 14 Response: 67	Byte 0: 0x04 Byte 1-4: 0x00 0x00 0x00 0x00 Byte 5-8: Coordinate x 0x00 0x10 0x00 0x00 Byte 9-12: Coordinate y 0x00 0x10 0x00 0x00	Get center point temperature and coordinates Response data: Byte 0: 0x04 (SETCONFIG) Byte 1: 0x00 (00: success FF: failure) Byte 2-65: DH_PAL_RadiometrySpotAreaInfo

```
typedef struct DH_PAL_RadiometrySpotAreaInfo
{
    DH_PAL_RADIOMETRY_SPOT_AREA_SIZE spotArea;///< spot area 4 bytes
    Point      location; ///< Spot coordinates 8 bytes
    float      baseTemp; ///< Reference temperature, output 4 bytes
    float      spotTemp; ///< Spot temperature, output 4 bytes
    NI_U16      spotGrey; ///< Spot grayscale, output 2 bytes
    NI_U16    placeHolder;///< Placeholder 2 bytes
    unsigned int       res[10];
}DH_PAL_RadiometrySpotAreaInfo;
```

Sample

Get the temperature at the center point coordinates of 320 and 256

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x50	0x00 0x0e	0x60 0xbb	0x24	0x04 0x00 0x00 0x00 0x00 0x00 0x10 0x00 0x00 0x00 0x10 0x00 0x00	0x11 0xdf

Reply	0x6E	0x00	0x00	0x50	0x00 0x43	0xf9 0xd2	0x24	0x04 0x00 ... ... (The structure data) DH_PAL_RadiometrySpotAreaInfo	0x1f 0xea

6.4 Get/Set the ID Coordinates of Temperature Rules

Command	Command code (scp)	Byte count of payload	Payload content	Description
Get ID Coordinates of temperature rules	0x09	Get: 2 Response: 3367	Byte 0: 0x03	Get the ID coordinates of temperature rules Response data: Byte 0: 0x03 (SETCONFIG) Byte 1: 0x00 (00: success FF: failure) Byte 2-3365: thermoTemp_AllToolRuleCfg
Temperature Testing Rules ID Coordinates	0x09	Get: 3366 Response: 3	Byte 0: 0x02 Byte 1-3364: thermoTemp_AllToolRuleCfg	Set the ID coordinates of temperature rules Response data: Byte 0: 0x02 (SETCONFIG) Byte 1: 0x00 (00: success FF: failure)

Get/Set the ID coordinates of temperature rules

{

unsigned int ruleNum;

ThermoTemp_RuleCfg rules[TEMP_RULE_MAX]; //TEMP_RULE_MAX == 12

}ThermoTemp_AllRuleCfg;

thermoTemp_AllToolRuleCfg 3364个字节 3364 bytes

```
typedef enum DH_PAL_RADIOMETRY_METERTYPE{
    typedef enum DH_PAL_RADIOMETRY_METERTYPE{       // Spot temperature measurement
    typedef enum DH_PAL_RADIOMETRY_METERTYPE{      / / Line Temperature Measurement
    typedef enum DH_PAL_RADIOMETRY_METERTYPE{      / / Area Temperature Measurement
}DH_PAL_RADIOMETRY_METERTYPE;


typedef enum DH_PAL_RADIOMETRY_AREASUBTYPE{
    DH_PAL_RADIOMETERY_AREAPOLYGON,                  //Polygon
    DH_PAL_RADIOMETERY_AREARECT,                     //Rectangular
    DH_PAL_RADIOMETERY_AREAELLIPSE                   //Elliptical
}DH_PAL_RADIOMETRY_AREASUBTYPE;
```

The structure of Temperature measurement rule

```
typedef struct ThermoTemp_RuleCfg
{
    bool enable; // Rule enabled
    unsigned char ruleId; /// Rule ID
    unsigned char type; /// Type       Above DH_PAL_RADIOMETRY_METERTYPE enumerate
    unsigned char subType; /// Sub type. It is only valid when the type is region.
                                    /// 如上DH_PAL_RADIOMETRY_AREASUBTYPE enumerate
    unsigned char pointNum; /// Number of coordinate points
    char ruleName[63+1]; /// Name
    unsigned int rpoint[12][2]; /// Coordinates
    int    blackBodyTemp;     // Calibrate the blackbody temperature, accuracy 0.1, and increase the actual
value by 10 times
     int    BlackErrorRange; // Blackbody error range temperature, accuracy 0.1, and increase the actual value
by 100 times
    unsigned int res[14];
    Point osdLoc; ///    Point 8 bytes, fill in 0
    bool localEnable; // Local parameters enabled
    float localDist; /// Local target distance,range    0.0—100.0
```

float localReflect; /// Local reflection temperature range -40.00—550.00

float localObjEmiss; /// Local radiation coefficient ,range 0.5—1.0001

unsigned char almCon; /// Alarm conditions

unsigned char almResType; /// Comparison types of alarm conditions, such as average temperature, maximum temperature, minimum temperature, temperature slope, and temperature difference

unsigned char almEnable; /// whether the alarm condition enabled

float almDiff; /// difference

float almHit; /// Alarm threshold

unsigned int almDuration; ///Alarm duration

bool almFlag;// alarm start flag

unsigned long almStartTime;//alarm start time

}ThermoTemp_RuleCfg;

Note: Unused variables such as OSD area and alarm are filled with 0x00

EG:ThermoTemp_RuleCfg

01 01 02 01 04 52 65 63 74 31 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 26 16 00 00 00 13 00 00 73 17 00 00 00 13 00 00 26 16 00 00 a0

14 00 00 73 17 00 00 a0 14 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 40

00 00 C8 41 EC 51 78 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00

Sample

Get 2 rules

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2

Command	0x6E	0x00	0x00	0x50	0x00 0x02	0xa1 0x37	0x09	0x03	0x8a 0xfb
Reply	0x6E	0x00	0x00	0x50	0x0d 0x27	0xa3 0xac	0x09	0x03 0x00 ... ... The structure data thermoTemp_AllToolRuleCfg	0xb3 0x03

Set 1 rules

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x50	0x0d 0x26	0xb3 0x8d	0x09	0x02 + The structure data thermoTemp_AllToolRuleCfg	0x55 0x2a
Reply	0x6E	0x00	0x00	0x50	0x00 0x03	0xb1 0x16	0x09	0x02 0x00	0xf8 0xf3

6.5 Switching between High and Low Temperature Modes

Description

Get/Set Gain (high temperature and low temperature)

Definition

The protocol adopts the general protocol

Command	Command code (scp)	Byte count of payload	Payload content	Description
Get temperature and coordinates	0x0a	Get: 0 Response: 2	None	Get the current temperature range Response data: Byte 0-1: temperature range (0x0001: high temperature，0x0002: low temperature)
		Set: 2 Response: 2	Byte 0-1: temperature range (0x0001: high	Set temperature range

			temperature，0x0002: low temperature)	

Sample

Get the current temperature mode. According to the protocol example, the mode is low temperature.

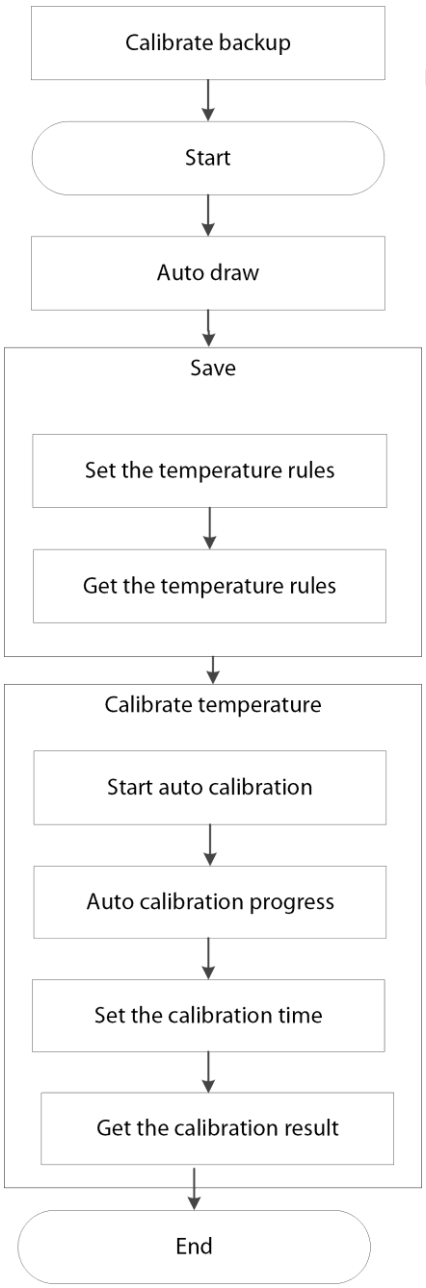
Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x0a	0x00 0x00	0x18 0x7a	—	0x00 0x00
Reply	0x6E	0x00	0x00	0x0a	0x00 0x02	0x38 0x38	0x00 0x02	0x20 0x42

Set the current range to high temperature

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x0a	0x00 0x02	0x38 0x38	0x00 0x01	0x10 0x21
Reply	0x6E	0x00	0x00	0x0a	0x00 0x02	0x38 0x38	0x00 0x01	0x10 0x21

6.6 Temperature Measurement Calibration

6.6.1. Calibration Process



6.6.2. Reference Temperature Compensation of Secondary Calibration

Description

Get the reference temperature compensation of secondary calibration

Preparation

Power on the device and have it heated for at least 1 hour.

Prepare blackbodies. Prepare four blackbodies to cover the following calibration temperatures.

The calibration plan requires two blackbodies in the image before starting the calibration process. The recommended calibration temperature of low temperature range: 40 °C and 110 °C; the recommended calibration temperature of high temperature range: 300 °C and 550 °C;

Adjust the calibration distance according to the focal length

Definition

The protocol adopts the temperature measurement protocol

	Read command code (SCP)	Byte count	Carried data	Description
Reference temperature compensation of secondary calibration	0x0c	Command of getting: 1 (Response: 82)	Get the reference temperature compensation of secondary calibration Byte 0-1: 0x04	Get the reference temperature compensation of secondary calibration Response data: Byte 0-1: 0x0400 Byte 2-13: <Array reference compensation compensation value * 100, array index 0-3 represents low temperature range 1-3, high temperature range Byte 14-33: Compensation grayscale table coefficient * 1000, array index 0-3 represents low temperature range 1-3, high temperature range Byte 34-37: Number of segments in high temperature range and normal temperature range Byte 38-67: The compensation value for the grayscale difference table, high

	Read command code (SCP)	Byte count	Carried data	Description
				temperature range and normal temperature range, the maximum value is 255, up to 30 numbers Byte 68-69: Reserved bit Byte 70-81: Reserved bit

typedef struct ThermNucAutoCaliOftParam

{

int oftTemp[3]; //<Reference temperature compensation value * 100, array index 0-2 represents low temperature range 1-3

int diffCoef[5]; //< Compensation grayscale table coefficient * 1000, array index 0-2 represents low temperature range 1-3, 3-4 represents a, b values of high temperature range

int highNormalNum; //Number of segments in high temperature range and normal temperature range

char highData[30]; //< The compensation value for the grayscale difference table, high temperature range and normal temperature range, the maximum value is 255, up to 30 numbers

char res1[2];

char res1[2];

}ThermNucAutoCaliOftParam;

Due to the lack of ranging in the normal temperature secondary calibration (plan 2), only softTemp [0] (reference temperature compensation: used for low-temperature 40° blackbody correction), diffCoef [0] (low-temperature grayscale compensation coefficient a: used for low-temperature 110° blackbody correction), diffCoef [3] (high-temperature grayscale compensation coefficient a: used for high-temperature 300° blackbody correction), and diffCoef [4] (high-temperature grayscale compensation coefficient b: used for high-temperature 550° blackbody correction) are used.

Sample

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x50	0x00 0x02	0xa1 0x37	0x0c	0x04	0x05 0xe9
Reply	0x6E	0x00	0x00	0x50	0x00 0x53	0xeb 0xe3	0x0c	04 00 89 0a 00 00 ...	0xe2 0xce

6.6.3. Automatic Draw

Description

Start automatic draw

Definition

The protocol adopts the temperature measurement protocol

Command	Read command code (SCP)		Carried data	Description
Start automatic draw	0x80	Command of getting: 1 (Response: 2)	Start automatic draw Byte 0-1: 0x04	Start automatic draw

Sample

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x50	0x00 0x03	0xb1 0x16	0x80	0x04	0xf7 0x9e
Reply	0x6E	0x00	0x00	0x50	0x00 0x53	0xeb 0xe3	0x80	0x0400	0xf7 0x9e

6.6.4. Start Auto Calibration

Description

Start auto calibration

Definition

The protocol adopts the temperature measurement protocol

Command	Read command code (SCP)	Byte count	Carried data	Description
Reference temperature compensation of secondary calibration	0x81	Command: 3 (Response: 2)	Start calibration Byte 0: 0x04 Byte 1: start marking Byte 2: environment mode	Start calibration

Sample

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x50	0x00 0x04	0c1 0xf1	0x81	0x04 0100	0x44 0x7d
Reply	0x6E	0x00	0x00	0x50	0x00 0x03	0xb1 0x16	0x81	0x0400	0xc0 0xae

6.6.5 Auto Calibration Progress

Description

Get auto calibration progress

Definition

The protocol adopts the temperature measurement protocol

Command	Read command code (SCP)	Byte count	Carried data	Description
Set auto calibration progress	0x82	Command of getting: 1 (Response: 6)	Start calibration Byte 0: 0x04	Start calibration Response: Byte 0-1: 0x0400 Byte 2-5: Calibration progress

Sample

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x50	0x00 0x02	0xa1 0x37	0x82	0x04	0x3d 0x7e
Reply	0x6E	0x00	0x00	0x50	0x00 0x04	0xc1 0xf1	0x82	0x0400 0x00	0xec 0x90

6.6.6 Get Auto Calibration Results

Description

Get auto calibration results

Definition

The protocol adopts the temperature measurement protocol

Command	Read command code (SCP)	Byte count	Carried data	Description
Get auto calibration results	0x83	Command of getting: 1 (Response: 1082)	Get the results of secondary calibration (normal temperature environment) Byte 0: 0x04	Get the results of secondary calibration (normal temperature environment) Response data: Byte 0-1: 0x0400 Byte 2-1081: SecThermNucBlackCaliResultArray Byte 2-181: Calibration results of normal-temperature blackbody at low

Command	Read command code (SCP)	Byte count	Carried data	Description
				temperature range secThermNucBlackCaliResult[0] Byte 541-720: Calibration results of normal-temperature blackbody at high temperature range
		Command: 2 (Response: 2)	Byte 0: 0x03 Byte 1: 0x00: environment mode	Secondary calibration (full environment) calibration recovery

32-bit operating system 1080 bytes calibration results of multiple-mode blackbody (secondary calibration plan 2)

```
typedef struct SecThermNucBlackCaliResultArray
{
    secThermNucBlackCaliResult[6];//The subscript 012 represents normal temperature segment, low temperature segment, and high temperature segment of low temperature range. 345 represents normal temperature segment, low temperature segment, and high temperature segment of high temperature range.
}SecThermNucBlackCaliResultArray;


}SecThermNucBlackCaliResultArray;
{
    char caliResultTime[32];    // Time of calibration completion
    int   caliResultNum;
    ThermNucBlacktCaliTempCfg caliTempCfg[2];
    signed char   allCaliEnable;
    char   res1[3];
    int    res[3];
}secThermNucBlackCaliResult;

typedef struct ThermNucBlacktCaliTempCfg
{
    char   caliRuleName[32];          // The name of the calibration rule
    int    preBlackBodyTemp;          // The blackbody temperature before calibration
    int    resultBlackBodyTemp;    // Calibration result: blackbody temperature
    int    caliErrorRangeTemp;     // Calibration error
    signed char   bCaliEnable;             // Whether the calibration is successful
    char   res1[3];
    int    res[4];
}ThermNucBlacktCaliTempCfg;
```

Sample

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x50	0x00 0x02	0xa1 0x37	0x83	0x04	0x0e 0x4f
Reply	0x6E	0x00	0x00	0x50	0x13 0x3b	0x50 0x6d	0x83	0x32 30 32 34 2d 30 39 2d 32 34 20 31...	0xfe 0x8e

6.6.7. Set the Time of Secondary Calibration

Description

Set the time of secondary calibration

Definition

The protocol adopts the temperature measurement protocol

Command	Read command code (SCP)		Carried data	Description
Set the time of secondary calibration	0x84	Set command: 36 (Response: 2)	Time of secondary calibration Byte 0-1: 0x0400 Byte 2: Gains mode Byte 3: environment mode Byte 4-35: Time	Set the time of secondary calibration

Sample

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x50	0x00 0x24	0xe5 0x93	0x84	0x04 02 00 32 30 32 34 2d ...	0xaf 0x44

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Reply	0x6E	0x00	0x00	0x50	0x00 0x03	0xb1 0x16	0x84	0x0400	0x2b 0x5e

6.6.8. Set Calibration Plan

Description

Set calibration plan

Definition

The protocol adopts the temperature measurement protocol

Command	Read command code (SCP)		Carried data	Description
Set calibration plan	0x93	Set command: 33 (Response: 2)	Calibration plan Byte 0: 0x04 Byte 1-32: Calibration plan	Set calibration plan

typedef struct CaliScheme

{

unsigned char CaliSchemes; //Calibration plan (1: full environment calibration 2: secondary calibration plan 2)

unsigned char res[31];

}CaliScheme;

Sample

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x50	0x00 0x22	0x85 0x55	0x93	0x04 01 00 00 00 …	0x12 0xc9
Reply	0x6E	0x00	0x00	0x50	0x00	0xb1	0x93	0x0400	0xed

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
					0x03	0x16			0xad

6.6.9. Secondary Calibration (Normal Temperature) Restore to Default

Description

Secondary calibration (normal temperature) restore to default

Definition

The protocol adopts the temperature measurement protocol

Command	Read command code (SCP)		Carried data	Description
Secondary calibration (normal temperature) restore to default	0x98	Command: 1 (Response: 2)	Byte 0: 0x04	Secondary calibration (normal temperature) restore to default

Sample

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x50	0x00 0x02	0xa1 0x37	0x98	0x04	0xd1 0xc6
Reply	0x6E	0x00	0x00	0x50	0x00 0x03	0xb1 0x16	0x98	0x0400	0x1d 0x5c

6.6.10. Calibration Backup

Description

Calibration backup

Definition

The protocol adopts the temperature measurement protocol

Command	Read command code (SCP)		Carried data	Description
Calibration backup	0x99	Set command: 1 (Response: 2)	Byte 0-1: 0x04	Calibration backup

Sample

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x50	0x00 0x02	0xa1 0x37	0x99	0x04	0xe2 0xf7
Reply	0x6E	0x00	0x00	0x50	0x00 0x03	0xb1 0x16	0x99	0x0400	0x2a 0x6c

6.6.11. Restore backup

Description

Restore backup

Definition

The protocol adopts the temperature measurement protocol

Command	Read command code (SCP)		Carried data	Description
Restore backup	0x9a	Set command: 1 (Response: 2)	Byte 0: 0x04	Restore backup

Sample

Protocol	Process Code	Status	Reserved	Function	Byte Count	CRC1	SCP Function	SCP Argument	CRC2
Command	0x6E	0x00	0x00	0x50	0x00 0x02	0xa1 0x37	0x9a	0x04	0xb7 0xa4
Reply	0x6E	0x00	0x00	0x50	0x00 0x07	0xf1 0x92	0x9a	0x04 00 01 00 00 00	0x1f 0xd9