CH9329 chip serial port

letter of agreement

V1.0

Document Change Log

version	scope of change	change content	Edited by
V1.0	document creation	Create a document, first	TECH2

The CH9329 chip has 3 serial communication modes:

Serial port communication mode 0: protocol transmission mode (default);

Serial communication mode 1: ASCII mode;

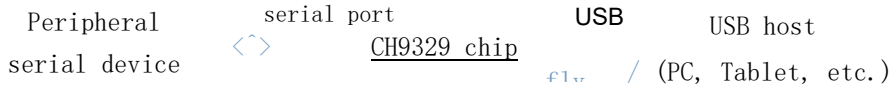
Serial communication mode 2: transparent transmission mode.

The CH9329 chip works in the serial communication mode 0 (protocol transmission mode) by default. This agreement is mainly used to specify the serial communication protocol for the CH9329 chip to work in this mode.

In any mode, the chip detects that the SET pin is low and automatically switches to the "protocol transmission mode", and the client serial device can configure parameters. Therefore, when parameter configuration is required, the SET pin can be set to low level first, and then configure.

1. communication structure

The communication structure diagram between peripheral serial devices (PC, MCU or other serial devices) and CH9329 chip is as follows:



2. way of communication

The communication between the peripheral serial device (PC, MCU or other serial device) and the CH9329 chip is master-slave, the peripheral serial device is the master, and the CH9329 chip is the slave. The commands are all initiated by the peripheral serial device, and the CH9329 chip responds passively. If the peripheral serial device cannot receive the response from the CH9329 chip within 500mS or the response information is wrong, it will be considered that the communication has failed.

2.1、Frame Format Description

The unit of communication is frame, that is, it is sent in the form of data packet. Each frame of data has frame header byte, address code, command code, follow-up data length, follow-up data and accumulated sum. If the CH9329 chip receives an error frame, it returns an error response frame or discards it directly.

Below, the communication frame initiated by the peripheral serial device is called "command packet", and the communication frame returned by CH9329 chip is called "response packet". For the "command packet", after the peripheral serial device sends it, it needs to wait for the CH9329 chip to return the "response packet", and determine whether the command is successfully executed according to the "response packet". If an error status is returned or the "response packet" cannot be received, retry or error handling is required according to the situation.

Note: All the data described below are in hexadecimal format.

The command packet and response packet data format is as follows:

frame	address	command	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
2 bytes	1 byte	1 byte	1 byte	N bytes (0-64)	1 byte

Frame header: 2 bytes, fixed at 0x57, 0xAB;

Address code: occupies 1 byte, the default is 0x00, and can receive command packets with any address code. If the chip address is set to 0x01---0xFE, it can only receive command packets with

the corresponding address code or address code 0xFF. 0xFF is a broadcast packet, and the chip does not need to respond;

Command code: 1 byte, the effective range of the command code of the frame initiated by the peripheral serial device is: 0x01---0x3F, the command code when the CH9329 chip sends a normal response packet is: the original command code| 0x80; the command when the CH9329 chip sends an abnormal response packet

The command code is: original command code| 0xC0;

Follow-up data length: 1 byte, mainly used to record the length of the actual follow-up data of the packet, including only follow-up data

Part, excluding frame header byte, address code, command code and accumulated sum byte;

Subsequent data: occupies N bytes, and the valid range of N is 0---64.

Cumulative sum: occupies 1 byte, the calculation method is: SUM = HEAD+ADDR+CMD+LEN+DATA.

2.2、Command code description

Table 1 - command code table

command name	naming	command description
CMD_GET_INFO	0x01	**Get chip version and other information** Use this command to obtain information such as the version
CMD_SEND_KB_GENERAL_DATA	0x02	**Send normal data from USB keyboard** Send ordinary keyboard data packets to the chip through this command, simulating the action of
CMD_SEND_KB_MEDIA_DATA	0x03	**Send USB keyboard multimedia data** Use this command to send multimedia keyboard data packets to the chip to simulate the press
CMD_SEND_MS_ABS_DATA	0x04	**Send USB absolute mouse data** Send absolute mouse data packets to the chip through this command
CMD_SEND_MS_REL_DATA	0x05	**Send USB relative mouse data** Send relative mouse data packets to the chip through this command
CMD_SEND_MY_HID_DATA	0x06	**Send USB custom HID device data** Use this command to send custom HID class device data packets to
CMD_READ_MY_HID_DATA	0x87	**Read USB custom HID device data** Use this command to read custom HID class device data packets from the chip Note: After the PC downloads a custom HID data packet to the
CMD_GET_PARA_CFG	0x08	**Get parameter configuration** Get the current parameter configuration information from
CMD_SET_PARA_CFG	0x09	**Set parameter configuration** Set the current parameter configuration information to the
CMD_GET_USB_STRING	0x0A	**Get string descriptor configuration** Get the currently used USB string
CMD_SET_USB_STRING	0x0B	**Set string descriptor**

		Use this command to set the currently used USB string
CMD_SET_DEFAULT_CFG	0x0C	**Restore factory default configuration** Use this command to restore the parameter configuration and
CMD_RESET	0x0F	**reset chip** Control the chip through this command to perform software reset

2.2.1、CMD_GET_INFO

Use this command to obtain information such as the version number, USB enumeration status, and keyboard case indicator status from the chip.

Peripheral serial device Ling chip:

frame	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x01	0x00	no data	0x03

This command takes no parameters.

Chip Ling peripheral serial device:

frame	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x81	0x08	8 bytes of data	0x??

The returned 8 bytes of follow-up **data** are as follows:

(1), 1 byte chip version number: such as 0x30 means V1.0, such as 0x31 means V1.1;

(2), 1 byte USB enumeration status:

0x00 means that the USB port is not connected to the computer or not recognized;

0x01 indicates that the USB port has been connected to the computer and recognized successfully;

(3), 1 byte current keyboard size indicator light status information;

Bit 0: The status of the NUM LOCK indicator light on the keyboard, 0: off; 1: on;

Bit 1: Keyboard CAPS LOCK indicator status, 0: off; 1: on;

Bit 2: keyboard SCROLL LOCK indicator status, 0: off; 1: on;

Bit 7---3: invalid;

(4), 5 bytes reserved;

2.2.2、CMD_SEND_KB_GENERAL_DATA

Send ordinary keyboard data packets to the chip through this command, simulating the action of pressing or releasing ordinary keys. Support full keyboard and combination key operation, and support 8+6 non-conflicting keys, 8 of which are 8 control keys (left Ctrl, right Ctrl, left Shift, right Shift, left Windows, right Windows, left Alt and right Alt) , 6 is an ordinary key other than the 6 control keys.

Peripheral serial device Ling chip:

frame	address code	command code	Subsequent data	follow-up data	cumulative

HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x02	8	8 bytes of data	0x??

This command carries 8 bytes of follow-up data, and the follow-up data is the key value of a common button on a USB keyboard. as followed:

(1), the first byte: a control key of 1 byte, each bit represents a key, as follows:

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
Right Windows key	Right Alt key	Right Shift key	Right Ctrl key	Left Windows key	Left Alt key	Left Shift key	Left Ctrl key

(2), the second byte: 1 byte 0x00, this byte must be 0x00;

(3), the 3rd-8th byte: 6-byte common button value, which can indicate that 6 buttons are pressed at most, if no button is pressed

Fill in 0x00 below;

For specific common keyboard keys and corresponding key codes, see Appendix 1 - "CH9329 Key Code Table".

Chip Ling peripheral serial device:

frame	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x82	1	1 byte data	0x??

The returned 1-byte follow-up data is: current command execution status.

The following examples illustrate:

Example 1: To simulate pressing the "A" key first, and then releasing the "A" key, you need to send 2 command packets as follows:

(1), Simulate pressing the "A" key: 0x57、0xAB、0x00、0x02、0x08、0x00、0x00、0x04、0x00、
0x00、0x00、0x00、0x00、0x10。

(2), simulate the release of the "A" key: 0x57、0xAB、0x00、0x02、0x08、0x00、0x00、0x00、0x00、
0x00、0x00、0x00、0x00、0x0C。

Example 2: To simulate pressing the "Left Shift" + "A" keys at the same time, and then release them, you need to send 2 command packets:

(1) Simulate pressing the "Left Shift" + "A" keys at the same time: 0x57, 0xAB, 0x00, 0x02, 0x08, 0x02, 0x00, 0x04, 0x00, 0x00, 0x00, 0x00 , 0x00, 0x12.

(2) Simulate release of all keys: 0x57, 0xAB, 0x00, 0x02, 0x08, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 , 0x0C.

2.2.3、CMD_SEND_KB_MEDIA_DATA

Send multimedia keyboard data packets to the chip through this command to simulate the press or release action of multimedia keys.

Peripheral serial device Ling chip:

frame	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x03	2	2 bytes of data	0x??

This command carries 2 bytes of follow-up data, and the follow-up data is the key value of the USB keyboard multimedia key. For specific common keyboard keys and corresponding key codes, see Appendix 1 - "CH9329 Key Code Table".

Chip Ling peripheral serial device:

frame	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x83	1	1 byte data	0x??

The returned 1-byte follow-up data is: current command execution status.

The following examples illustrate:

Example 1: To simulate pressing the "Mute" key of the multimedia first, and then releasing the "Mute" key of the multimedia, two command packets need to be sent:

(1), press the "Mute" key of the multimedia: 0x57、0xAB、0x00、0x03、0x04、0x02、0x04、0x00、
0x00、0x0F。

(2), Simulate the "Mute" key for releasing multimedia: 0x57, 0xAB, 0x00, 0x03, 0x04, 0x02, 0x00, 0x00,
0x00、0x0B。

2.2.4、 CMD_SEND_MS_ABS_DATA

Use this command to send absolute mouse data packets to the chip to simulate absolute mouse related actions (including pressing and releasing the left, middle and right buttons) put, scroll wheel up and down, move up and down, left and right).

Peripheral serial device Ling chip:

frame header	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x04	7	7 bytes of data	0x??

This command carries 7 bytes of follow-up data, and the 7 bytes of follow-up data is the data packet of the USB absolute mouse, in order: (1), the first byte: must be 0x02;
(2), the second byte: 1 byte of the mouse button value, the lowest 3 digits each represent a button, as follows:

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
0	0	0	0	0	Middle key	right click	left button

BIT2---BIT0: 1 means the key is pressed, and 0 means the key is released or not pressed.
(2), the 3rd-4th byte: 2-byte X-axis coordinate value, the low byte is in front, and the high byte is in the back; (3), the 5th-6th byte: 2-byte Y Axis coordinate value, the low byte is in front, and the high byte is in the back; (4), the 7th byte: 1 byte rolling gear number,

If it is 0, it means that there is no action for scrolling;

0x01---0x7F, means scroll up, unit: number of teeth;

0x81---0xFF, means scroll down, unit: number of teeth;

Chip Ling peripheral serial device:

frame header	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x84	1	1 byte data	0x??

The returned 1-byte follow-up data is: current command execution status.

Note: The default simulated absolute mouse resolution of the chip is 4096 * 4096. When the peripheral serial device downloads the absolute value of XY, it needs to calculate according to its own screen resolution, and then download the calculated value.

For example, the current screen resolution is: X_MAX(1280) * Y_MAX(768), you need to move to point (100, 120), and you need to perform the following calculation:

X_Cur = (4096 * 100) / X_MAX;

Y_Cur = (4096 * 120) / Y_MAX;

The following examples illustrate:

For example 1: To simulate pressing the "left" button of the mouse first, and then release the "left" button of the mouse, you need to send 2 command packets as follows:

(1), press the "left" button of the mouse: 0x57、0xAB、0x00、0x04、0x07、0x02、0x01、0x00、0x00、
0x00、0x00、0x00、0x10。

(2), release the "left" button of the mouse: 0x57、0xAB、0x00、0x04、0x07、0x02、0x00、0x00、0x00、
0x00、0x00、0x00、0x0F。

Example 2: Suppose the screen resolution is: 1280*768, and the control mouse moves to the position (100, 100) first, and then moves to the position (968, 500), you need to send 2 command packets as follows:

(1), move to (100, 100) position:

Calculate position X1 = (100 * 4096) / 1280 = 320 = 0x140

Calculate position Y1 = (100 * 4096) / 768 = 533 = 0x215

The sending command packets are: 0x57, 0xAB, 0x00, 0x04, 0x07, 0x02, 0x00, 0x40, 0x01, 0x15, 0x02, 0x00 , 0x67.

(2), move to (968,500) position:

Calculate position X1 = (968 * 4096) / 1280 = 3097 = 0xC19

Calculate position Y1 = (500 * 4096) / 768 = 2667 = 0xA6B

The sending command packets are: 0x57, 0xAB, 0x00, 0x04, 0x07, 0x02, 0x00, 0x19, 0x0C, 0x6B, 0x0A, 0x00 , 0xA9.

2.2.5、 CMD_SEND_MS_REL_DATA

Use this command to send relative mouse data packets to the chip to simulate relative mouse actions (including pressing and releasing the left, middle and right buttons, scrolling the wheel up and down, and moving up and down and left and right).

Peripheral serial device Ling chip:

frame header	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x05	5	5 bytes of data	0x??

This command carries 5 bytes of follow-up data, and the follow-up data is the data packet of the USB relative to the mouse, in order:

(1), the first byte: must be 0x01;

(2), the second byte: 1 byte of the mouse button value, the lowest 3 digits each represent a button, as follows:

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
0	0	0	0	0	Middle key	right click	left button

BIT2---BIT0: 1 means the key is pressed, and 0 means the key is released or not pressed.

(3), the third byte: 1 byte X direction (ordinate, up and down direction) moving distance;

A. Not moving: Byte 3 = 0x00, it means not moving in the X-axis direction;

B. Move right: 0x01 <= Byte 3 <= 0x7F; moving pixels = byte 3;

C. Move left: 0x80 <= Byte 3 <= 0xFF; moving pixels = 0x00 - byte 3;

(4), the fourth byte: 1 byte Y direction (ordinate, up and down direction) movement distance;

A. Not moving: Byte 4 = 0x00, it means not moving in the direction of the Y axis;

B. Move right: 0x01 <= Byte 4 <= 0x7F; moving pixels = byte 4;

C. Move left: 0x80 <= Byte 4 <= 0xFF; moving pixels = 0x00 - byte 4;

(5), the 5th byte: 1 byte scroll wheel rolling teeth number,

0x01---0x7F, indicating that the screen scrolls up, unit: number of teeth;

0x81---0xFF, indicating that the screen scrolls down, unit: number of teeth;

The distance calculation method for scrolling down:

For example, the byte is 0x81, the actual moving distance=0x100-0x81=127 pixels; for example, the byte is 0xFF, the actual moving distance=0x100-0xFF=1 pixel.

Chip Ling peripheral serial device:

frame header	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x85	1	1 byte data	0x??

The returned 1-byte follow-up data is: current command execution status.

The following examples illustrate:

For example 1: To simulate pressing the "left" button of the mouse first, and then release the "left" button of the mouse, you need to send 2 command packets as follows:

(1), press the "left" button of the mouse: 0x57、0xAB、0x00、0x05、0x05、0x01、0x01、0x00、0x00、
0x00、0x0E。

(2), release the "left" button of the mouse: 0x57、0xAB、0x00、0x05、0x05、0x01、0x00、0x00、0x00、
0x00、0x0D。

Example 2: Control the mouse to move 3 pixels to the left, and then move down 5 pixels, then you need to send 2 command packets:

(1) First move 3 pixels to the left: 0x57, 0xAB, 0x00, 0x05 , 0x05, 0x01, 0x00, 0xFD, 0x00, 0x00 , 0x0A.

(2), move down 5 pixels: 0x57, 0xAB, 0x00, 0x05, 0x05, 0x01, 0x00, 0x00, 0x05, 0x00、0x12。

2.2.6、CMD_SEND_MY_HID_DATA

Use this command to send a custom HID class device data packet to the chip.

Peripheral serial device Ling chip:

frame	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x06	N	N bytes of data	0x??

This command carries N bytes of follow-up data. The follow-up data is the HID data packet that is expected to be uploaded via USB. The effective range of N is: 0-64;

Chip Ling peripheral serial device:

frame	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x86	1	1 byte data	0x??

The returned 1-byte follow-up data is: current command execution status.

2.2.7、CMD_READ_MY_HID_DATA

Use this command to read custom HID class device data packets from the chip. After the PC downloads a custom HID data packet to the chip, it is automatically packaged and sent to the

peripheral serial device by the chip serial port.

Chip Ling peripheral serial device:

frame	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x87	N	N bytes of data	0x??

This command carries N bytes of follow-up data. The follow-up data is the HID data packet downloaded from the USB. The valid range of N is:

0-64;

Note: This command is actively sent by the chip to the peripheral serial device, and the peripheral serial device does not need to respond.

2.2.8、CMD_GET_PARA_CFG

Use this command to obtain the current parameter configuration information from the chip. For specific parameters, see the description of the returned data below.

Peripheral serial device Ling chip:

frame	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x08	0	none	0x??

This command does not take any parameter data.

Chip Ling peripheral serial device:

frame	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x88	50	50 bytes of data	0x??

The 50 bytes of follow-up data returned are:

(1), 1-byte chip working mode: the effective value is 0x00-0x03, 0x80---0x83, and the default is 0x80;

0x00: Working mode 0 set by the software, standard USB keyboard (common + multimedia) + USB mouse (absolute mouse + relative mouse);

0x01: Working mode 1 set by software, standard USB keyboard (common);

0x02: Working mode 2 set by the software, standard USB mouse (absolute mouse + relative mouse);

0x03: Working mode 3 set by software, standard USB custom HID device;

0x80: Working mode 0 set by hardware pin, standard USB keyboard (common + multimedia) + USB mouse (absolute mouse standard + relative mouse); currently MODE1 pin is high level, MODE0 pin is high level;

0x81: Working mode 1 set by hardware pins, standard USB keyboard (common); current MODE1 pin is high level, MODE0 pin is low level;

0x82: Working mode 2 set by hardware pins, standard USB mouse (absolute mouse + relative mouse); currently MODE1 pin is low level, MODE0 pin is high level;

0x83: Working mode 3 set by hardware pin, standard USB custom HID device; current MODE1 pin is low level, MODE0 pin is low level;

(2), 1 byte chip serial port communication mode, the effective value is 0x00-0x02, 0x80---0x82, the default is 0x80;

0x00: Serial port communication mode 0 set by software, protocol transmission mode;

0x01: Serial port communication mode 1 set by software, ASCII mode;

0x02: Serial port communication mode 2 set by software, transparent transmission mode;

0x80: Serial port communication mode 0 set by hardware pins, protocol transmission mode; current CFG1 pin is high level, CFG0 pin is high level;

0x81: Serial port communication mode 1 set by hardware pins, ASCII mode; current CFG1 pin is high level, CFG0 pin is low level;

0x82: Serial port communication mode 2 set by hardware pins, transparent transmission mode; the current CFG1 pin is low level, and the CFG0 pin is high level;

(3), 1 byte chip serial port communication address, the valid range is 0x00--0xFF, the default is 0x00;

(4), 4-byte chip serial port communication baud rate, high byte first, the default is 0x00002580, that is, the baud rate is 9600bps;

(5), 2 bytes reserved;

(6), 2-byte chip serial port communication packet interval, the effective range is 0x0000--0xFFFF, the default is 3, and the unit is mS, that is, if the chip does not receive the next byte for more than 3mS, it means that the packet is over;

(7) The VID and PID of the 4-byte chip USB, the default chip VID is 0x1A86, and the PID is 0xE129. In different working modes, the PID is different;

(8), 2-byte chip USB keyboard upload time interval (only valid in ASCII mode), the effective range is 0x0000--0xFFFF, the default is 0, the unit is mS, that is, the chip uploads the next packet immediately after uploading the first packet of data packet data;

(9), 2-byte chip USB keyboard release delay time (only valid in ASCII mode), the valid range is 0x0000--0xFFFF, the default is 1, and the unit is mS, that is, 1 mS after the chip uploads the button and presses the data packet Upload key release data packet;

(10), 1-byte chip USB keyboard automatic carriage return flag (only valid in ASCII mode), the valid range is 0x00--0x01, 0x00 means no automatic carriage return, 0x01 means automatic carriage return after the end of the package;

(11), 8-byte chip USB keyboard carriage return (only valid in ASCII mode), 4 bytes in one group, 2 groups in total, that is, 2 different carriage return characters can be set, and the default ASCII value is 0x0D Carriage return;

(12), 8 bytes of chip USB keyboard filter start and end character strings, the first 4 bytes are filter start characters, and the last 4 bytes are filter end characters;

(13), 1 byte chip USB string enable flag,

Bit 7: 0 means disable; 1 means enable custom string descriptor;

Bits 6-3: Reserved;

Bit 2: 0 means disable; 1 means enable custom vendor string descriptor;

Bit 1: 0 means disable; 1 means enable custom product string descriptor;

Bit 0: 0 means disable; 1 means enable custom serial number string descriptor;

(14), 1 byte chip USB keyboard fast upload flag (only valid in ASCII mode), the effective range is 0x00--0x01, 0x00 means that the USB keyboard upload speed is normal, 0x01 means enable the USB keyboard fast upload mode, enable fast After uploading mode, after uploading 1 character, the release button packet will not be sent, and the next character will be uploaded, and the release button packet will be sent only after all characters are uploaded.

(15), 12 bytes reserved;

2.2.9、CMD_SET_PARA_CFG

Use this command to set the current parameter configuration information to the chip, and the specific parameter format is described in the previous command.

Peripheral serial device Ling chip:

frame header	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x09	50	50 bytes of data	0x??

This command carries 50 bytes of follow-up data. For the specific data format, see the return of the " CMD_GET_PARA_CFG " command. Notice:

(1) When the chip working mode is set, the effective range is: 0x00-0x03;

(2) When the chip serial port communication mode is set, the effective range is: 0x00-0x02;

(3) After all parameters are set, it will be enabled at the next power-on.

Chip Ling peripheral serial device:

frame header	address code	command code	Subsequent data length	follow-up data	cumulative sum
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x89	1	1 byte data	0x??

The returned 1-byte follow-up data is: current command execution status.

2.2.10、 CMD_GET_USB_STRING

Get the currently used USB string descriptor configuration from the chip through this command.

Peripheral serial device Ling chip:

frame	address code	command code	Subsequent	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x0A	1	1 byte data	0x??

This command takes a parameter of 1 byte, in order:

(1), 1 byte string type, 0x00 represents the manufacturer string descriptor; 0x01 represents the product string descriptor;

0x02 indicates the serial number string descriptor;

Chip Ling peripheral serial device:

frame	address code	command code	Subsequent	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x8A	2+N	2+N bytes of data	0x??

The returned 2+N bytes of follow-up data are as follows:

(1), 1 byte string type;

(2), 1 byte string length, valid range is 0---23;

(3), the current character string descriptor of N bytes, the effective range of N is: 1-23;

2.2.11、 CMD_SET_USB_STRING

Set the currently used USB string descriptor configuration to the chip through this

command.

Peripheral serial device Ling chip:

frame	address code	command code	Subsequent	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x0B	2+N	2+N bytes of data	0x??

This command takes 2+N bytes of parameters, in order:

(1), 1 byte string type, 0x00 represents the manufacturer string descriptor; 0x01 represents the product string descriptor;

0x02 indicates the serial number string descriptor;

(2), 1 byte string length, valid range is 0---23;

(3), N byte string descriptors, the effective range of N is: 1-23;

Chip Ling peripheral serial device:

frame	address code	command code	Subsequent	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x8B	1	1 byte data	0x??

The returned 1-byte follow-up data is: current command execution status.

2.2.12、CMD_SET_DEFAULT_CFG

Use this command to restore the parameter configuration and character string configuration information of the chip to the factory default settings.

Peripheral serial device Ling chip:

frame	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x0C	0	none	0x??

This command takes no parameters.

Chip Ling peripheral serial device:

frame	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x8C	1	1 byte data	0x??

The returned 1-byte follow-up data is: current command execution status.

2.2.13、CMD_RESET

Control the chip to perform software reset control through this command.

Peripheral serial device Ling chip:

frame	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x0F	0	none	0x??

This command takes no parameters.

Chip Ling peripheral serial device:

frame	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0x8F	1	1 byte data	0x??

The returned 1-byte follow-up data is: current command execution status.

2.3. Error response packet

If the command packet received by the chip has problems such as command code error, verification error, or execution failure, it needs to respond with an error response packet. The error response packet contains 1 byte follow-up data, which is the command

frame	address code	command code	Subsequent data	follow-up data	cumulative
HEAD	ADDR	CMD	LEN	DATA	SUM
0x57、0xAB	0x00	0xC?	1	1 byte data	0x??

The returned 1-byte follow-up data is: current command execution status.

Returned command code = original command code| 0xC0;

Table 2 - The command execution status is as follows

state name	status code	status description
DEF_CMD_SUCCESS	0x00	command executed successfully
DEF_CMD_ERR_TIMEOUT	0xE1	The serial port receives a byte
DEF_CMD_ERR_HEAD	0xE2	The serial port receives the
DEF_CMD_ERR_CMD	0xE3	The command code received by the
DEF_CMD_ERR_SUM	0xE4	Accumulated and checked values do
DEF_CMD_ERR_PARA	0xE5	Parameter error
DEF_CMD_ERR_OPERATE	0xE6	Frame OK, Execution Failed

Appendix 1 - "CH9329 key code table"

1. Common keys and corresponding key code table:

serial number	symbol		HID Page	HID Code	serial number	symbol		HID Page	HID Code
1	~	`	07	35	54	>	.	07	37
2	!	1	07	1E	55	?	/	07	38
3	@	2	07	1F	56	Keycode56 (*BJ)		07	87
4	#	3	07	20	57	Shift (R)		07	E5
5	$	4	07	21	58	Ctrl (L)		07	E0
6	%	5	07	22	60	Alt (L)		07	E2
7	^	6	07	23	61	Ctrl (L)		07	2C
8	&	7	07	24	62	Alt (R)		07	E6
9	*	8	07	25	64	Ctrl (R)		07	E4
10	(	9	07	26	75	Insert		07	49
11	)	0	07	27	76	Delete		07	4C
12		-	07	2D	79	Left Arrow		07	50
13	+	=	07	2E	80	Home		07	4A
14	Keycode14 (*J)		07	89	81	End		07	4D
15	Back Space		07	2A	83	↑		07	52
16	Tab		07	2B	84	↓		07	51
17	Q		07	14	85	PgUp		07	4B
18	W		07	1A	86	PgDn		07	4E
19	E		07	08	89	→		07	4F
20	R		07	15	90	Num Lock		07	53
21	T		07	17	91	7	Home	07	5F
22	Y		07	1C	92	4	←	07	5C
23	U		07	18	93	1	End	07	59
24			07	0C	95	/		07	54
25	O		07	12	96	8	↑	07	60
26	P		07	13	97	5		07	5D
27	{	[	07	2F	98	2	↓	07	5A
28	}	]	07	30	99	0	Ins	07	62
29	Keycode29 (*4)		07	31	100	*		07	55
30	Caps Lock		07	39	101	9	PgUp	07	61
31	A		07	04	102	6	→	07	5E
32	S		07	16	103	3	PgDn	07	5B
33	D		07	07	104	.	Del	07	63
34	F		07	09	105	-		07	56
35	G		07	0A	106	+		07	57
36	H		07	0B	107	Keycode107 (*B)		07	85
37	J		07	0D	108	Enter_R		07	58
38	K		07	0E	110	ESC		07	29

39	L	07	0F	112	F1	07	3A
40	: ;	07	33	113	F2	07	3B
41	“ ‘	07	34	114	F3	07	3C
42	Keycode42 (*5BJ)	07	32	115	F4	07	3D
43	Enter_L	07	28	116	F5	07	3E
44	Shift (L)	07	E1	117	F6	07	3F
45	Keycode45 (*5B)	07	64	118	F7	07	40
46	Z	07	1D	119	F8	07	41
47	X	07	1B	120	F9	07	42
48	C	07	06	121	F10	07	43
49	V	07	19	122	F11	07	44
50	B	07	05	123	F12	07	45
51	N	07	11	124	Print Screen	07	46
52	M	07	10	125	Scroll Lock	07	47
53	<	07	36	126	Pause	07	48
* 4 _ 104 Keyboard Only * 5 _ 105 Keyboard				*B _ 107 Keyboard Only *J _ 109 Keyboard Only			

serial number	symbol	HID Page	HID Code
131 (*J)	Japanese J131	07	8B
132 (*J)	Japanese J132	07	8A
133 (*J)	Japanese J133	07	88
150	KoreaKC-L,Key_Hangul	07	90
151	Korea KC-R, Key_Hanja	07	91
ACPI	Power	01	81
ACPI	Sleep	01	82
ACPI	Wake-up	01	83
Windows Key	L_WIN	07	E3
Windows Key	R_WIN	07	E7
Windows Key	APP	07	65

2. Multimedia keys and corresponding key code table:

For the ACPI key, there are 2 bytes in total, the first byte is REPORT ID, which is fixed at 0x01, and the second byte is the ACPI key code.

byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	00000001b							
2	00000b					Wake-up	Sleep	Power
1: key pressed 0: key release								

For other multimedia keys, it occupies 4 bytes, the first byte is REPORT ID, which is fixed at 0x02, and the second byte

Up to the fourth byte is the multimedia key value.

byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	00000010b							
2	Eject	CD Stop	Prev. Track	Next Track	Play/Pause	Mute	Volume-	Volume+
3	Refresh	WWW Stop	WWW Forward	WWW Back	WWW Home	WWW Favorites	WWW Search	E-Mail
4	Rewind	Record	Minimize	My Computer	Screen Save	Calculator	Explorer	Media
1: key pressed 0: key release								