QU-950-4-HF-3.0

Datasheet

(Revision V3.4)

Quick Ohm Küpper & Co. GmbH

2025/3/4

Quick Ohm Küpper & Co. GmbH
Cronenfelder Straße 75
423494 Wuppertal

kontakt@quick-ohm.de
+49 202 40 43 3

Contents

1. Introduction

The QU-950-4-HF is a proximity reader/writer that supports contactless cards (ISO 14443 Type A/B, Mifare, FeliCa TM). It supports RS232, RS485, Wiegand interface. Wiegand supports 32/34/64/66 bits, which can be configured by commands. The RS485 interface communication adopts the general Modbus protocol, which is convenient to be embedded in any master control.

QU-950-HF is an IP65 protection (waterproof) reader, suitable for access control system, attendance system, parking management and other systems.

2. Hardware Specifications

- Card Type: ISO14443-4 TYPE A/B, Mifare FeliCa
- Communication interface: RS232/RS485, Wiegand
- LED: 2 LEDs (red, blue)
- Buzzer: Yes
- Power: DC 7-24V
- Maximum power consumption: 200mA (12V)
- Card reading distance: ≥5CM (depending on the type of card)
- Dimension: 106(H)*48(W)*22(D) mm
- Operating temperature: -25 to +85°C
- Storage temperature: -40 to +125°C

3. Connection line sequence

3.1. Version A (12-line sequence)

PIN NAME	COLOR	TYPE	PIN NUMBER
PWR-IN	RED	POWER	1
PWR-GND	BLACK	POWER	2
RS232-RX	GRAY	I/O	3
RS232-TX	PINK	I/O	4
WGD-D0	GREEN	O	5
WGD-D1	WHITE	O	6
PWR-GND	BLUE	POWER	7
LED	BROWN	I	8
BEEP	YELLOW	I	9
TAMPER	VIOLET	O	10
RS485+	ORANGE	I/O	11
RS485-	SKY BLUE	I/O	12

3.2. Version B (4-wire sequence)

PIN NAME	COLOR	TYPE	PIN NUMBER
PWR-IN	RED	POWER	1
PWR-GND	BLACK	POWER	2
RS485+	YELLOW	I/O	3
RS485-	GREEN	I/O	4

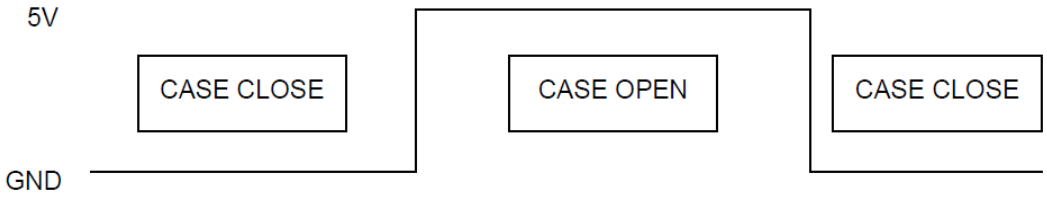
4. Operation and use

4.1 The card reader is powered on, the buzzer beeps, and the blue LED lights up. Indicates that the device is working properly.

4.2 When the card is swiped, the card reader beeps once and turns off the blue LED for 100ms and then turns on again. The card data is stored in the reader within the specified time, and the card data is sent to the controller through weigand (the Wiegand interface needs to be turned on through the RS232 command). The reader reads the card every 30ms. The card reader will not read the card again until the card is out of range of the antenna.

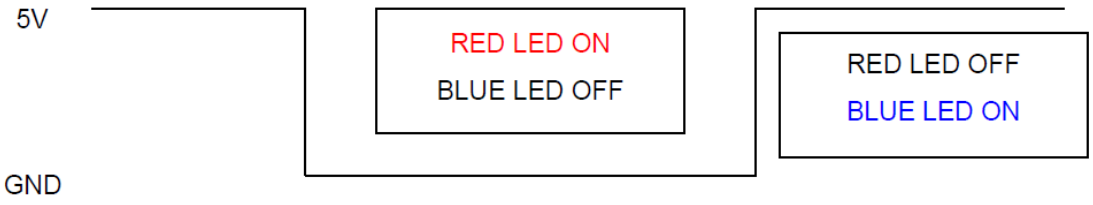
4.3 Alarm function

When it is forced to open, the reader sounds an alarm. Open the case, send the TAMPER signal (high level 5V), and beep. When the chassis is closed, it returns to GND.



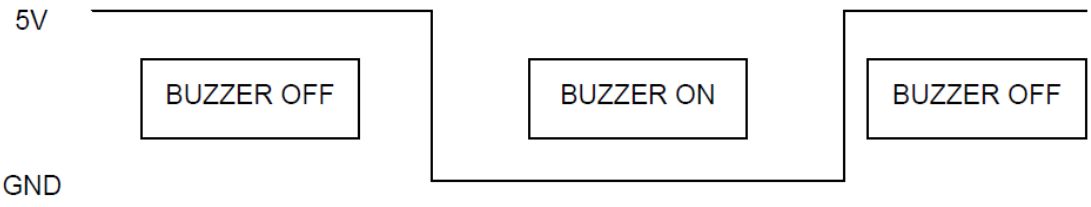
4.4 LED Control

When the LED control line of the reader is grounded, the red LED is turned on (the blue LED is turned off); when it is connected to 5V, the blue LED is turned on (the red LED is turned off).



4.5 Buzzer control

When the buzzer control line of the reader is grounded, the buzzer keeps beeping; when it is connected to 5V, the buzzer is turned off.



5. Function configuration

RS232 interface is used for Wiegand and RS485 communication parameter configuration.
These settings will be saved in the reader, no need to repeat the settings.
The communication parameters are fixed as: 9600bps, 8 data, no parity, 1 stop bit.

5.1. Wiegand Set

Command Format

STX	LENH	LENL	CMD	DATA	LRC
0x02	0x00	0x02	0xE0	DATA[0]	LRC

LRC：XOR value of LENH byte to DATA byte

DATA [0] Description

DATA[0]	State	Description
Bit7	0	Disable Wiegand output
	1	Enable Wiegand output
Bit6~4	RFU	RFU
Bit3	0	4 bytes card information
	1	8 bytes card information
Bit2	0	send check digit
	1	Do not send check digit
Bit1	0	Card information forward output
	1	Card information reverse output
Bit0	RFU	RFU

Receive command format

STX	LENH	LENL	ResponseCode	LRC
0x02	0x00	01	ResponseCode	LRC

Response Code:

Response Code	Description
0x00	Command executed successfully
0x01	Command execution failed
0x10	Command undefined
0x11	Command send and receive timeout

5.2. ModBus parameter settings

Send

STX	LENH	LENL	CMD	DATA	LRC
0x02	0x00	n+1	0xE5	DATA[n]	LRC

LRC：XOR value of LENH byte to DATA byte

Receive

STX	LENH	LENL	ResponseCode	DATA	LRC
0x02	0x00	n+1	ResponseCode	DATA[n]	LRC

DATA [0]: R/W flag

DATA[0]	Description
0x00	write parameter
0x01	read parameter

DATA [1~n]: sub command

DATA[1]	Sub command	DATA[2~n] length	Default Value(hex)
0x00	Slave address	1	01
0x01	RS485 Parameters & Baudrate	8	000001C200080100
0x02	Card Serial number clear time	1	1E (UNIT : 100ms)
0x11	Discrete Inputs Address(case open)	2	0000
0x12	Coils Address(Buzzer，LED control)	2	0000
0x13	Input Register Address(Card Serical Number)	2	0000

RS485 Parameters & Baudrate(Sub command = 0x01) Description

DATA[2~n]	Description	Value(hex)
DATA[2]	Serial type	00 : RS485
DATA[3~6]	Baudrate	0001C200 : 115200bps 0000E100 : 57600bps 00009600 : 38400bps 00004B00 : 19200bps 00002580 : 9600bps
DATA[7]	Bit Length	08 : 8 bit 09 : 9 bit
DATA[8]	Stop bits	01 : 1 bit 02 : 1.5 bit 03 : 2 bit
DATA[9]	Parity	00 : none 01 : odd 02 : even

5.3. Example

1）Enable Wiegand interface

Data [0]: 0x80 – Enable Wiegand interface, output 4 bytes of card data in the forward direction, and send check digit.
Send：02 00 02 E0 80 62
Receive：02 00 01 00 01（00：Command executed successfully）

2）Read device address

Slave address number as 0x01
Data [0]: 0x01 - Read flag
Data [1]: 0x00 – Sub Command (Slave address)

Send：02 00 03 E5 01 00 E7
Receive：02 00 02 00 01 03（00：Command executed successfully，01：Slave address number）

IMPORTANT: There is just STX send, no ETX (from StxEtx Frame)

6. Wiegand

6.1. Data format

The data format is configurable, see the function configuration chapter for details

<34bit> Default

Parity	Data[1-32]	Parity
1bit	32bit	1bit
Bit1	Bit2 ... Bit33	Bit34

First Bit 1 : bit2~bit17 Even parity

Data [1-32] : Card data

Last Bit 34 : bit18~bit33 Odd parity

<66bit>

Parity	Data[1-64]	Parity
1bit	64bit	1bit
Bit1	Bit2 ... Bit65	Bit66

First Bit 1 : bit2~bit33 Even parity

Data [1-64] : Card data

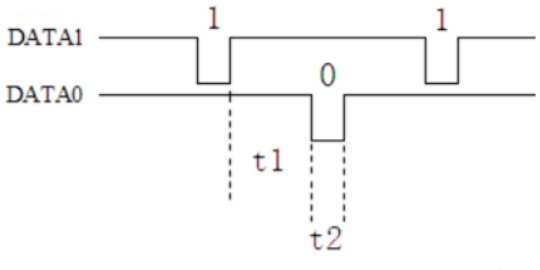
Last Bit 66 : bit34~bit65 Odd parity

6.2. Wiegand Output time and level

Level = 5V

t1 = 2.16 mS

t2 = 138 uS



7. Modbus

If the user wants to connect more than two readers in series, please set the readers to different slave addresses.
(e.g.： device A = 0x01, device B = 0x02)

7.1. Modbus Address Map & Parameters

1） Card Serial/SystemParameter(Input Register)Address Map

	Offset(word)	Hi	Lo	Description
Card Serial Number(hex)	0001	SN(1)	SN(2)	Card Serial Number
	0002	SN(3)	SN(4)	
	……			
	0016	SN(31)	SN(32)	
Length(hex)	0017	RFU	Length	
Card Serial Number(ASCII)	0018	SN(1)H	SN(1)L	Card Serial Number
	0019	SN(2)H	SN(2)L	
	0049	SN(32)H	SN(32)L	
Length(ASCII)	0050	RFU	Length	
System Parameter	0051	SlaveAdd	RS485Speed	System Parameter
	0052	CardData HoldTime H	CardData HoldTime L	
	0053	Alarmfunctions	RFU	
	0054	KeepCardData	AutoBeep	
….	….	….	….	…
Block data	0160	BlockData(1)	BlockData(2)	MIFARE 1K/4K Block Data
	0161	BlockData(3)	BlockData(4)	
	…	…	…	
	0167	BlockData(15)	BlockData(16)	

2） Output Port(Coils) Address Map

	Offset(bit)	Description
Buzzer	1	0:off
		1:on
LED	2	0:off
		1:on
Output_ Buzzer	3	0: Output low
		1: Output high
Output_ LED	4	0: Output low
		1: Output high

3） Input Port(Discrete Inputs) Address Map

	Offset(bit)	Description
Csae sensor	1	0:closed
		1:opened

4） Write Single Register Address Map

	Register	ValueHi	ValueLo	Description
Set Parameter	0000	00	Slaveaddress	set device address
	0001	00	RS485 Speed	Set RS485 rate
	0002	CardData HoldTime H	CardData HoldTime L	Set card data retention time （UNIT: 10ms）
	0003	KeepCardData 0x00: Removals 0x01: Non-removal	AutoBeep 0x00:Disable 0x01:Enable	KeepCardData: After the card data retention time has expired, whether to clear the card data if the card has not changed. AutoBeep: After the card data retention time has expired, whether to continue to beep once if the card has not changed.
	1000	00	Alarmfunctions 00 : turn off the alarm 01 : Turn on the alarm	Set switch alerts

RS485 Speed Description：

RS485 Speed	Speed
0x02	9600
0x03	19200
0x04	38400
0x05	115200
0x06	57600
Other	9600

5） Default value

Parameter name	size	Default value (hex)	Description
Slave address	1 byte	01	No.1
RS485 Baudrate	1 byte	05	115200/8/n /1
Card Serial number clear time	2 byte	012C	3000ms(UINT:10ms)
KeepCardData	1 byte	00	Removals
AutoBeep	1 byte	00	Disable
Discrete Inputs Address(case open)	2 byte	0000	Base Address
Coils Address(Buzzer，LED control)	2 byte	0000	Base Address
Input Register Address(Card Serical Number)	2 byte	0000	Base Address

7.2. Example of Communication Protocol

1） ReadCardData (Read Input Register 0x04)

UID = 0x76, 0x40, 0x9B, 0xF0

RegisterData：

	0	1	2	3	4	5	6	7	8	9
0	0x76	0x40	0x9B	0xF0	0x00	0x00	0x00	0x00	0x00	0x00
1	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
2	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
3	0x00	0x00	0x00	0x04	0x37	0x36	0x34	0x30	0x39	0x42
4	0x46	0x30	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
5	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
6	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
7	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
8	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00
9	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x08

Host Send

DeviceAddress	FunctionCode	StartAddressHi	StartAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x03	0x00	0x00	0x00	0x0a	0xc5,0xcd
0x01	0x04	0x00	0x00	0x00	0x0a	0x70,0x0d

DeviceAddress= 0x01

StartAddress= (StartAddressHi *256) + StartAddressLo = 0x0000

Quantity= (QuantityHi *256) + QuantityLo = 0x000A

Host receives (success)

DeviceAddress	FunctionCode	ByteCount	Data	CRC
0x01	0x03	0x14	0x76,0x40,0x9b,0xf0,0x00,0x00,0x00,0x00,0x00,0x00,0x00, 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00	0xcb, 0x4b
0x01	0x0	0x14	0xfe,0xcb,0x47,0xa5,0x00,0x00,0x00,0x00,0x00,0x00,0x00, 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00	0xfd, 0xad

ByteCount= ((QuantityHi*256) + QuantityLo)*2

DeviceAddress= 0x01

Card UID 4Byte = 0xfe, 0xcb, 0x47, 0xa5

Host receive (failed)

DeviceAddress	FunctionCode	ByteCount	CRC
0x01	0x83/0x84	0x01/0x02/0x03/0x04	0xxx, 0xxx

2） ReadSystemParameter (Read Input Register 0x04)

Host Send

DeviceAddress	FunctionCode	StartAddressHi	StartAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x03	0x00	0x32	0x00	0x03	0xa4,0x04
0x01	0x04	0x00	0x32	0x00	0x03	0x11,0xc4

DeviceAddress= 0x01

StartAddress= (StartAddressHi *256) + StartAddressLo = 0x0032

Quantity= (QuantityHi *256) + QuantityLo = 0x0003

Host receives (success)

DeviceAddress	FunctionCode	ByteCount	Data	CRC
0x01	0x03	0x06	0x01,0x05,0x01,0x2c,0x00,0x00	0x2c, 0x91
0x01	0x04	0x06	0x01,0x05,0x01,0x2c,0x00,0x00	0x6d, 0x77

ByteCount= ((QuantityHi*256) + QuantityLo)*2

DeviceAddress= 0x01

SlaveAdd = 0x01

RS485Speed = 0x05(115200)

CardDataHoldTime = 0x012c (uint: 10ms)

Alarmfunctions = 0x00(DISABLE)

RTU = 0x00

Host receive (failed)

DeviceAddress	FunctionCode	ByteCount	CRC
0x01	0x83/0x84	0x01/0x02/0x03/0x04	0xxx, 0xxx

3） LED Red On(Write Single Coil 0X05)

LED Red On/ LED Blue Off

Host Send

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x01	0xff	0x00	0xdd,0xfa

Host receive

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x01	0xff	0x00	0xdd,0xfa

4） LED Blue On(Write Single Coil 0X05)

LED Red Off/ LED Blue On

Host Send

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x01	0x00	0x00	0x9c,0x0a

Host receive

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x01	0x00	0x00	0x9c,0x0a

5） Buzzer On(Write Single Coil 0X05)

Host Send

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x00	0xff	0x00	0x8c,0x3a

Host receive

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x00	0xff	0x00	0x8c,0x3a

6） Buzzer Off(Write Single Coil 0X05)

Host Send

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x00	0x00	0x00	0xcd,0xca

Host receive

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x00	0x00	0x00	0xcd, 0xca

7） OutPut Buzzer Low Level(Write Single Coil 0x05)

Host Send

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x02	0x00	0x00	0x6c,0x0a

Host receive

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x02	0x00	0x00	0x6c,0x0a

8） OutPut Buzzer High Level(Write Single Coil 0x05)

Host Send

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x02	0xff	0x00	0x2d,0xfa

Host receive

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x02	0xff	0x00	0x2d,0xfa

9） OutPut LED Low Level(Write Single Coil 0x05)

Host Send

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x03	0x00	0x00	0x3d,0xca

Host receive

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x03	0x00	0x00	0x3d,0xca

10） OutPut LED High Level(Write Single Coil 0x05)

Host Send

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x03	0xff	0x00	0x7c,0x3a

Host receive

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x05	0x00	0x03	0xff	0x00	0x7c,0x3a

11） Case Sensor On (Read Discrete Input 0x02)

Host Send

DeviceAddress	FunctionCode	StartAddressHi	StartAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x02	0x00	0x00	0x00	0x01	0xb9, 0xca

Host receive (Case Open)

DeviceAddress	FunctionCode	ByteCount	Data	CRC
0x01	0x02	0x01	0x01	0x60, 0x48

Host receive (Case Close)

DeviceAddress	FunctionCode	ByteCount	Data	CRC
0x01	0x02	0x01	0x00	0xa1, 0x88

12） Set the device address (Write Single Register 0x06)

Host Send

DeviceAddress	FunctionCode	RegisterAddrHi	RegisterAddrLo	OutValueHi	OutValueLo	CRC
0x01	0x06	0x00	0x00	0x00	0x01	0x48, 0x0a

Set the device address to 0x01 (OutValueLo). The address range is 0x01~0xF7.

Note: The address prohibition is set to 0x00, 0x00 is the broadcast address.

The Returns is an error, if OutValueLo is 0.

The host receives (the address is set successfully and takes effect immediately)

DeviceAddress	FunctionCode	RegisterAddrHi	RegisterAddrLo	OutValueHi	OutValueLo	CRC
0x01	0x06	0x00	0x00	0x00	0x01	0x48, 0x0a

Host reception (address setting failed)

DeviceAddress	FunctionCode	ExceptionCode	CRC
0x01	0x86	0x01/0x02/0x03/0x04	0xxx, 0xxx

13） Set the communication rate (Write Single Register 0x06)

Host Send

DeviceAddress	FunctionCode	RegisterAddrHi	RegisterAddrLo	OutValueHi	OutValueLo	CRC
0x01	0x06	0x00	0x01	0x00	0x05	0x18,0x09

The communication rate is: 0x05 (see the RS485 Speed table for details)

The host receives (the setting is successful, it will take effect immediately)

DeviceAddress	FunctionCode	RegisterAddrHi	RegisterAddrLo	OutValueHi	OutValueLo	CRC
0x01	0x06	0x00	0x01	0x00	0x05	0x18,0x09

Host receive (setup failed)

DeviceAddress	FunctionCode	ExceptionCode	CRC
0x01	0x86	0x01/0x02/0x03/0x04	0xxx, 0xxx

14） Set the card data retention time (Write Single Register 0x06)

Host Send

DeviceAddress	FunctionCode	RegisterAddrHi	RegisterAddrLo	OutValueHi	OutValueLo	CRC
0x01	0x06	0x00	0x02	0x01	0x2c	0x28,0x47

Card data retention time = (OutValueHi * 256) + OutValueLo, unit: 10ms

The host receives (the setting is successful, it will take effect immediately)

DeviceAddress	FunctionCode	RegisterAddrHi	RegisterAddrLo	OutValueHi	OutValueLo	CRC
0x01	0x06	0x00	0x02	0x01	0x2c	0x28,0x47

Host receive (setup failed)

DeviceAddress	FunctionCode	ExceptionCode	CRC
0x01	0x86	0x01/0x02/0x03/0x04	0xxx, 0xxx

15） Set automatic working mode (Write Single Register 0x06)

Host Send

DeviceAddress	FunctionCode	RegisterAddrHi	RegisterAddrLo	KeepCardData	AutoBeep	CRC
0x01	0x06	0x00	0x03	0x00	0x00	0x79,0xca

Host receives (setting is successful and takes effect immediately)

DeviceAddress	FunctionCode	RegisterAddrHi	RegisterAddrLo	OutValueHi	OutValueLo	CRC
0x01	0x06	0x00	0x03	0x00	0x00	0x79,0xca

Host receivs (setting failed)

DeviceAddress	FunctionCode	ExceptionCode	CRC
0x01	0x86	0x01/0x02/0x03/0x04	0xxx, 0xxx

16） Turn off the alarm function (Write Single Register 0x06)

Host Send

DeviceAddress	FunctionCode	RegisterAddrHi	RegisterAddrLo	OutValueHi	OutValueLo	CRC
0x01	0x06	0x10	0x00	0x00	0x00	0x8d,0x0a

The host receives (the setting is successful, it will take effect immediately)

DeviceAddress	FunctionCode	RegisterAddrHi	RegisterAddrLo	OutValueHi	OutValueLo	CRC
0x01	0x06	0x10	0x00	0x00	0x00	0x8d,0x0a

Host receive (setup failed)

DeviceAddress	FunctionCode	ExceptionCode	CRC
0x01	0x86	0x01/0x02/0x03/0x04	0xxx, 0xxx

17） Turn on the alarm function (Write Single Register 0x06)

Host Send

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x06	0x10	0x00	0x00	0x01	0x4c,0xca

The host receives (the setting is successful, it will take effect immediately)

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x06	0x10	0x00	0x00	0x01	0x4c,0xca

Host receive (setup failed)

DeviceAddress	FunctionCode	ExceptionCode	CRC
0x01	0x86	0x01/0x02/0x03/0x04	0xxx, 0xxx

18） Read version information (Extended Function Code 0x41)

Host Send

DeviceAddress	FunctionCode	OutPutAddressHi	OutPutAddressLo	QuantityHi	QuantityLo	CRC
0x01	0x41	0x00	0x00	0x00	0x0a	0xbd,0xc2

Host receive

DeviceAddress	FunctionCode	ByteCount	Data	CRC
0x01	0x41	0x14	0x51,0x55,0x39,0x35,0x30,0x34,0x48,0x46,0x32,0x30, 0x32,0x32,0x30,0x37,0x31,0x34,0x31,0x2e,0x30,0x38	0xcd, 0x83

Data [0~7]: “QU9504HF” The name of the firmware

Data [8~15]: “20220714” Firmware development date

Data [16~19]: “1.08” Firmware version

7.3. Mifare 1K/4K commands

1） Read a single data block (Write multiple registers Code 0x10)

The read data is temporarily stored in the input register; the starting register address is 0x00A0, a total of 8 consecutive registers.

When using the 0x03/0x04 function code to read data, the fixed start address is 0x00A0, and the fixed register number is 0x0008.

The read data automatic clearing time refers to CardDataHoldTime.

Host Send

DeviceAdd	Code	StartRegHi	StartRegLo	QuantityHi	QuantityLo	DataCount	Data	CRC
0x01	0x10	0x00	0x64	0x00	0x05	0x0A	Data	0xxx,0xxx

Data:

Data[0]	Data[1]	Data[2]	Data[3]~ Data[8]	Data[9]
0x21	KeyID	BlockNum	Key(6Bytes)	0x00

Key ID:

Bit	Value	Descript
Bit0	0	Key A
	1	Key B
Bit1	0	using the key in the command
	1	using the key downloaded by command， the key in the command can be a random number
Bit6~bit2	0x00~0x1F	if use the downloaded key, this is the index of the key
Bit7	0	Res

BlockNum: S50 card from 0 to 0x3F; S70 card from 0 to 0xFF

The host receives (read successfully)

DeviceAdd	Code	StartRegHi	StartRegLo	QuantityHi	QuantityLo	CRC
0x01	0x10	0x00	0x64	0x00	0x05	0x41,0xd5

The host receives (failed to read)

DeviceAddress	FunctionCode	ExceptionCode	CRC
0x01	0x90	0x01/0x02/0x03/0x04	0xxx, 0xxx

2） Write a single data block (Write multiple registers Code 0x10)

Host Send

DeviceAdd	Code	StartRegHi	StartRegLo	QuantityHi	QuantityLo	DataCount	Data	CRC
0x01	0x10	0x00	0x64	0x00	0x0D	0x1A	Data	0xxx,0xxx

Data:

Data[0]	Data[1]	Data[2]	Data[3]~ Data[8]	Data[9]~ Data[24]	Data[25]
0x22	KeyID	BlockNum	Key(6Bytes)	BlockData(16Bytes)	0x00

Key ID:

Bit	Value	Descript
Bit0	0	Key A
	1	Key B
Bit1	0	using the key in the command
	1	using the key downloaded by command, the key in the command can be a random number.
Bit6~bit2	0x00~0x1F	if use the downloaded key, this is the index of the key
Bit7	0	Res

BlockNum: S50 card from 0 to 0x3F; S70 card from 0 to 0xFF

The host receives (write successfully)

DeviceAdd	Code	StartRegHi	StartRegLo	QuantityHi	QuantityLo	CRC
0x01	0x10	0x00	0x64	0x00	0x0D	0x40,0x13

The host receives (write failure)

DeviceAddress	FunctionCode	ExceptionCode	CRC
0x01	0x90	0x01/0x02/0x03/0x04	0xxx, 0xxx

3） Download the key to the module (Write multiple registers Code 0x10)

Host Send

DeviceAdd	Code	StartRegHi	StartRegLo	QuantityHi	QuantityLo	DataCount	Data	CRC
0x01	0x10	0x00	0x64	0x00	0x04	0x08	Data	0xxx,0xxx

Data:

Data[0]	Data[1]	Data[2]~ Data[7]
0x2D	KeyIndex	Key(6Bytes)

Key Index：0x00~0x1F

The host receives (write successfully)

DeviceAdd	Code	StartRegHi	StartRegLo	QuantityHi	QuantityLo	CRC
0x01	0x10	0x00	0x64	0x00	0x04	0x80,0x15

The host receives (write failure)

DeviceAddress	FunctionCode	ExceptionCode	CRC
0x01	0x90	0x01/0x02/0x03/0x04	0xxx, 0xxx

4） Operation example

// Write block1, use the key in the command
01100064000d1a 22 00 01 ffffffffffff ffffffffffffffffffffffffffffffff 00 1e9c
01100064000d4013

// read block1, use the key in the command
0110006400050a 21 00 01 ffffffffffff 00 bbff
01100064000541d5
// Read block data, data clearing time reference: CardDataHoldTime
010400A00008f1ee
010410 ffffffffffffffffffffffffffffffff 155C

// Write block1, use the key in the command
01100064000d1a 22 00 01 ffffffffffff 000102030405060708090a0b0c0d0e0f 00 f86b
01100064000d4013

// Read block1, use the key in the command
0110006400050a 21 00 01 ffffffffffff 00 bbff
01100064000541d5

//read block data
010400A00008f1ee
010410 000102030405060708090a0b0c0d0e0f 5f3b

// Download key, key number 0x00
01100064000408 2d 00 ffffffffffff 053b
0110006400048015

// Download key, key number 0x1f
01100064000408 2d 1f ffffffffffff ebfa
0110006400048015

//Write block1, use the key numbered 0x1f, the key data in the command can be any value
01100064000d1a 22 7e 01 112233445566 ffffffffffffffffffffffffffffffff 00 0d20
01100064000d4013

//Read block1, use the key numbered 0x00, the key data in the command can be any value
0110006400050a 21 02 01 ffffffffffff 00 a29f
01100064000541d5

//read block data
010400A00008f1ee
010410 ffffffffffffffffffffffffffffffff 155C

//Write block1, use the key numbered 0x1f, the key data in the command can be any value
01100064000d1a 22 7e 01 665544332211 000102030405060708090a0b0c0d0e0f 00 183f
01100064000d4013

// Read block1, use the key numbered 0x00, the key data in the command can be any value
0110006400050a 21 02 01 aabbccddeeff 00 f6a7
01100064000541d5

//read block data
010400A00008f1ee
010410 000102030405060708090a0b0c0d0e0f 5f3b

//error example

// Write block1, use the key in the instruction, the key is wrong
01100064000d1a 22 00 01 112233ffffff ffffffffffffffffffffffffffffffff 00 c8f4
0190044dc3

// Read block1, use the key numbered 0x01, the key is not downloaded
0110006400050a 21 06 01 aabbccddeeff 00 c467
0190044dc3