

RM16xx instruction manual



www.rockmong.com

上海岩獾科技有限公司
V1.4

Contents

RM16xx instruction manual.....	1
一、 Product Features.....	3
二、 Product selection.....	3
三、 Main parameters.....	3
四、 Installation method.....	4
五、 Wiring method	5
六、 Drive	6
七、 Introduction to Debugging Tool Usage.....	6
1. Overview of Tools.....	6
2. Digital I/O	7
2.1 Introduction to Pin Modes.....	7
2.2 Status Introduction	7
3. Modify the power on status.....	8
4. Modify SN.....	8
八、 Introduction to Basic Programming - Single IO Operation	9
1. Obtain the device	9
2. Read input status	9
3. Control output status	9
4. Read the output status	9
九、 Introduction to Basic Programming 2- Simultaneous Bit Operation for Multiple IOs ...	10
1. Read input status	10
2. Control the output status	10
3. Read the output status	10
十、 Other programming introductions.....	12
1. Multiple input ports read simultaneously.....	12
2. Multiple output ports are written simultaneously.....	12
3. Multiple output IOs simultaneously read status	13

一、 Product Features

- Power supply: DC 7-30V;
- Input/output: Multi channel optocoupler isolated digital input; Multi channel transistor output;
- Communication interface: USB;
- Communication speed: One read and write only takes 0.5ms (limited by computer environment and other factors);
- Communication protocol: No need to worry about the underlying implementation, just call library functions, multi-threaded safety, easy portability, simple to use, and efficient;
- Cross platform: Supports Windows Linux、Android、Mac OS;
- Provide various language routines: C/C++, C #, Python, Java, LabView, etc;
- Support modifying the output status when powered on;

二、 Product selection

model	input/output
RM1604	4 入 4 出
RM1608	8 入 8 出
RM1616	16 入 16 出
For more channels, please contact us	

三、 Main parameters

Parameter	describe
Rated voltage of power supply	DC 7-30V
Operation indicator	The green LED is always on to indicate normal operation, and off to indicate a disconnected USB connection
Input	When the power supply is 24V: logic high 18-24V, logic low 0-18V When the power supply is 12V: logic high 6-12V, logic low 0-6V When the power supply is 9V: logic high 6-9V, logic low 0-6V
Input instructions	Red LED indicator
Transistor output	Maximum 30V 3A, NPN type
Output indication	Red LED indicator
communication interface	USB
communication rate	Maximum 2KHz
temperature range	Industrial grade, -40℃~85℃

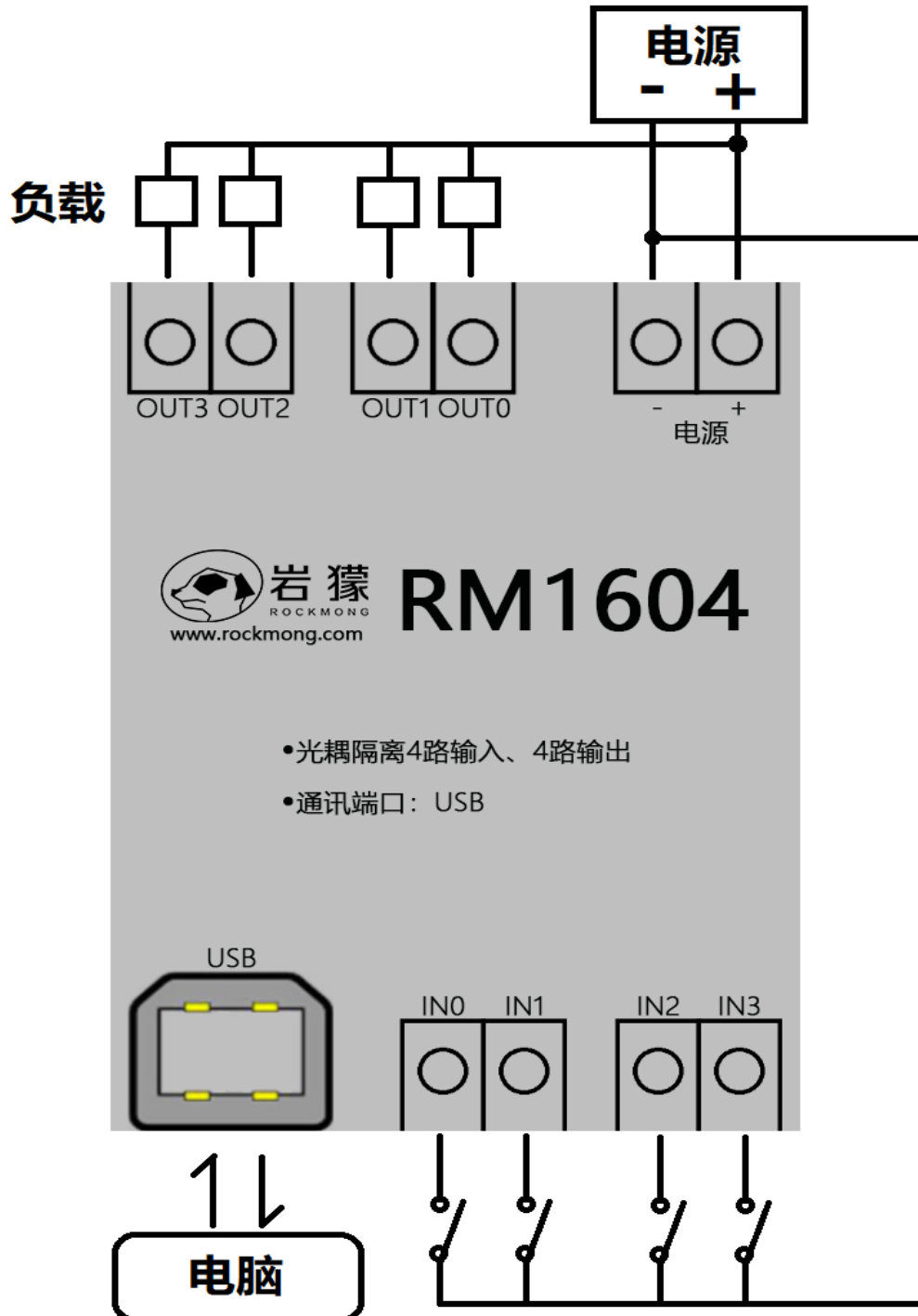
四、 **Installation method**

Industrial control box type, guide rail, threaded copper column type

五、Wiring method

Taking RM1604 with 4 inputs and 4 outputs as an example. Other RM16 models are similar.

1. The input terminal can be connected to switches, NPN output sensors, PLCs, etc., using a common cathode connection method.
2. The load at the output end can be a relay PLC、 Three color lights, etc., use the common anode connection method.



六、 Drive

Windows、Linux、Android、Mac OS require no installation of drivers. (Win7 or lower versions require installation)

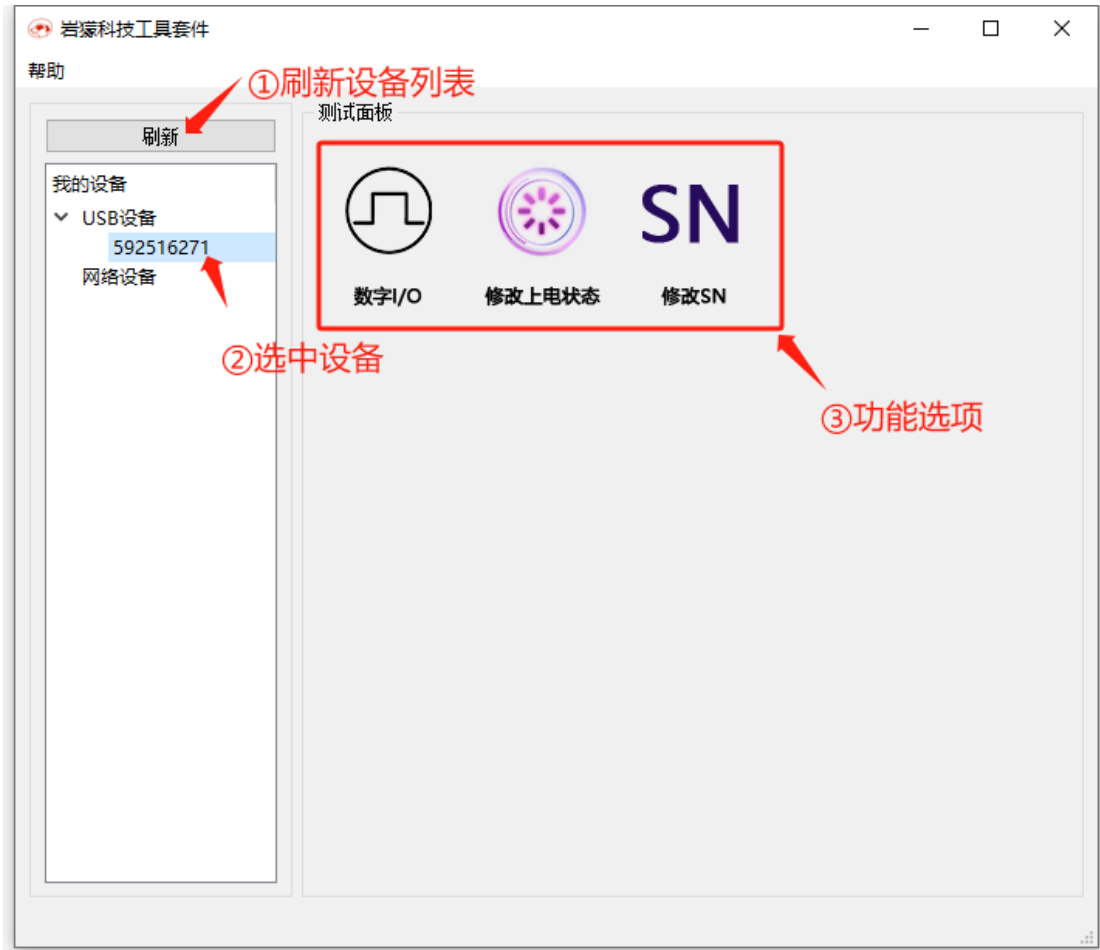
七、 Introduction to Debugging Tool Usage

Debugging tool icon:



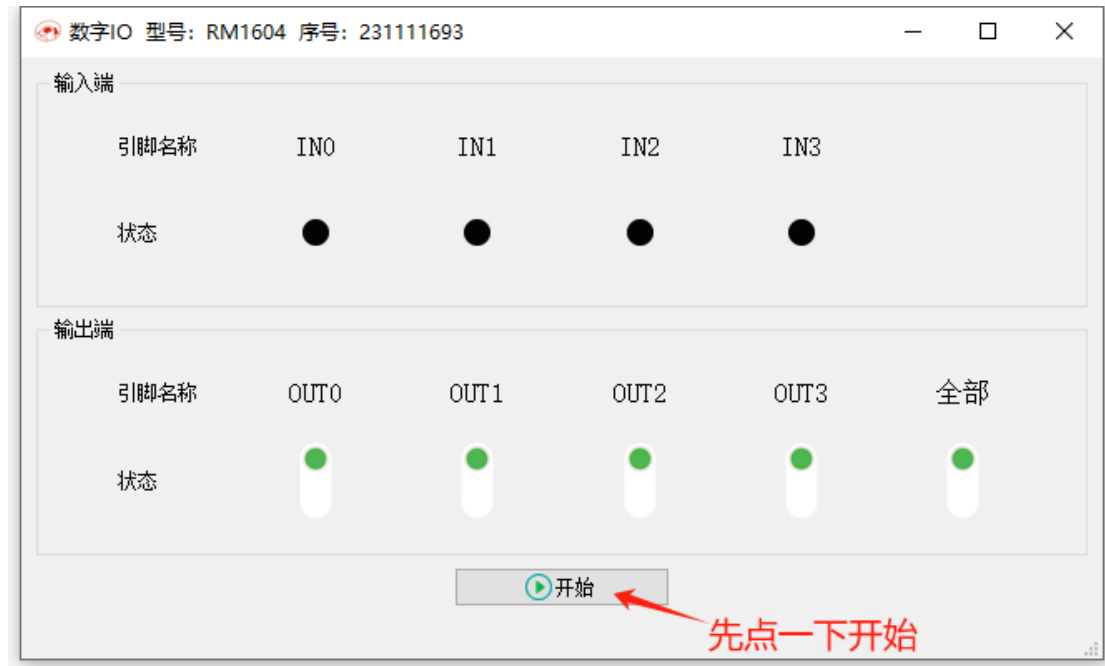
1. Overview of Tools

Double click to open:



2. Digital I/O

Used for real-time reading and writing of IO. After clicking on "Digital I/O" to open it:



2.1 Introduction to Pin Modes

1. Input mode: used to read the level status of IO. For example, used to identify switch status, read sensor output, etc.
2. Output mode: Used to control the output IO level status. For example, controlling light bulbs, driving relays, tri color lights, and so on.

2.2 Status Introduction

1. Input terminal: IO level status indicator light. Black represents low level, green represents high level.
2. Output terminal: IO level control switch button. Black represents low level, green represents high level. Clicking it will flip the voltage level.

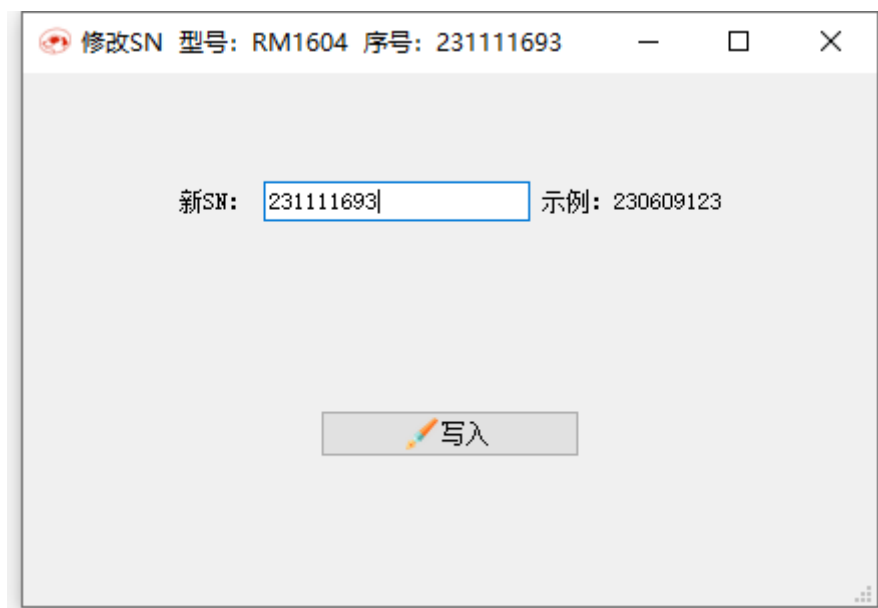
3. Modify the power on status

Used to modify the output status of IO when powered on. The status options refer to the description in the "Digital I/O" chapter. After modification, click the write button. After the successful write dialog box pops up, re plug and unplug the USB cable to take effect.



4. Modify SN

First, fill in the new SN with a length range of 1-9 digits. After writing, click the write button. After the prompt is successful, return to the main program page and click the refresh button to refresh the device list.



八、 Introduction to Basic Programming - Single IO Operation

We need to use two libraries, librockmong and libusb-1.0

Here we only introduce library functions in C language, similar to other languages. Please refer to the Demo_iO routines in each language for details.

1. Obtain the device

```
1. // Scan USB devices to obtain a list of device serial numbers.
2. // If the return value is greater than 0, it represents the number of devices
   obtained. If it is equal to 0, it means that the device has not been inserted.
   If it is less than 0, it means an error has occurred.
3. int UsbDevice_Scan(int* SerialNumbers);
```

2. Read input status

```
1. // Read pin status
2. //SerialNumber: Equipment serial number
3. //Pin: Pin number. 0 IN0. 1, IN1...
4. //PinState: Return pin status. 0, low level. 1. High level
5. // Function return: 0, normal< 0, Exception
6. int IO_ReadPin(int SerialNumber, int Pin, int *PinState);
```

3. Control output status

```
1. //Control pin output status
2. //SerialNumber: Equipment serial number
3. //Pin: Pin number. 0, OUT0. 1, OUT1...
4. //PinState: Pin status. 0, the transistor is conducting. 1. The transistor is
   disconnected
5. // Function return: 0, normal< 0, Exception
6. int IO_WritePin(int SerialNumber, int Pin, int PinState);
7.
```

4. Read the output status

```
1. // Read the status of output pins
2. //SerialNumber: Equipment serial number
```

```
3. //Pin: Pin number. 0, P0. 1, P1...
4. //PinState: Return pin status. 0, low level. 1. High level
5. // Function return: 0, normal< 0, Exception
6. int IO_ReadOutputPin(int SerialNumber, int Pin, int *PinState);
```

九、 Introduction to Basic Programming 2-Simultaneous Bit Operation for Multiple IOs

Here we only introduce library functions in C language, similar to other languages. For details, please refer to the Demo_IO-Bit routine in each language.

1. Read input status

```
1. // Simultaneously read the status of all input pins
2. //SerialNumber: Equipment serial number
3. //PinState: Return pin status. Each bit represents an IO. If bit0 is IN0, Bit1 is IN1, and so on
4. // The corresponding bit is 0, low level. 1. High level
5. // Function return: 0, normal< 0, Exception
6. int IO_ReadPin_Bit(int SerialNumber, int* PinState);
```

2. Control the output status

```
1. // Simultaneously control the output status of all pins
2. //SerialNumber: Equipment serial number
3. //PinState: Write pin status. Each bit represents an IO. If bit0 is OUT0, Bit1 is OUT1, and so on
4. // The corresponding bit is 0, and the relay is disconnected (the transistor is conducting). 1. Relay engagement (transistor disconnection)
5. // Function return: 0, normal< 0, Exception
6. int IO_WritePin_Bit(int SerialNumber, int PinState);
```

3. Read the output status

```
1. // Simultaneously read the status of all output pins
2. //SerialNumber: Equipment serial number
3. //PinState: Return pin status. Each bit represents an IO. If bit0 is OUT0, Bit1 is OUT1, and so on
```

```
4. // The corresponding bit is 0, and the relay is disconnected (the transistor
   is conducting). 1. Relay engagement (transistor disconnection)
5. // Function return: 0, normal< 0, Exception
6. int IO_ReadOutputPin_Bit(int SerialNumber, int* PinState);
```

十、 Other programming introductions

Here we only introduce library functions in C language, similar to other languages. For details, please refer to the Demo_iO-Multi routine in each language.

1. Multiple input ports read simultaneously

```
1. typedef struct
2. {
3.     uint8_t Pin;    // Pin number
4. }IO_Read_Struct_Tx_t;
5.
6. typedef struct
7. {
8.     uint8_t Ret;     // Return: 0, normal< 0, Exception
9.     uint8_t PinState; // Pin status
10. }IO_Read_Struct_Rx_t;
11.
12. // Simultaneously reading the status of multiple input ports
13. //SerialNumber: Equipment serial number
14. //TxStruct: Send data structure pointer
15. //RxStruct: Receive data structure pointers
16. //Number: The number of structures
17. // Function return: 0, all normal< 0, there is an exception
18. int IO_ReadMultiPin(int SerialNumber, IO_Read_Struct_Tx_t* TxStruct, IO_Read_Struct_Rx_t* RxStruct, int Number);
```

2. Multiple output ports are written simultaneously

```
1. typedef struct
2. {
3.     uint8_t Pin;    // Pin number
4.     uint8_t PinState; // Pin status
5. }IO_Write_Struct_Tx_t;
6.
7. typedef struct
8. {
9.     uint8_t Ret;     // Return: 0, normal< 0, Exception
10. }IO_Write_Struct_Rx_t;
11.
12. // Simultaneously writing multiple input port states
13. //SerialNumber: Equipment serial number
```

```
14. //TxStruct: Send data structure pointer
15. //RxStruct: Receive data structure pointers
16. //Number: Quantity of structural units
17. // Function return: 0, all normal< 0, there is an exception
18. int IO_WriteMultiPin(int SerialNumber, IO_Write_Struct_Tx_t* TxStruct, IO_Write_Struct_Rx_t* RxStruct, int Number);
```

3. Multiple output IOs simultaneously read status

```
1. struct IO_ReadOutput_TxStruct
2. {
3.     uint8_t Pin;
4. };
5. typedef struct IO_ReadOutput_TxStruct IO_ReadOutput_TxStruct_t;
6.
7. struct IO_ReadOutput_RxStruct
8. {
9.     uint8_t Ret;
10.    uint8_t PinState;
11. };
12. typedef struct IO_ReadOutput_RxStruct IO_ReadOutput_RxStruct_t;
13.
int IO_ReadMultiPin(int SerialNumber, IO_ReadStruct_Tx_t* TxStruct, IO_ReadStruct_Rx_t*
```