1. COMMUNICATION INTERFACE SPECIFICATION

RRUITMUX includes Master (RRUITMUX-M) and Slave (RRUITMUX-S). One typical RRUITMUX implement comprises of one Master and several Slaves.

Master communicates with host (MCU，MPU，Controller) using serial communication interface RS232 or network interface TCPIP and complete corresponding operation according to the host command. The communication parameter is 115200bps, 1 start bit, 8 data bits, 1 stop bit and no parity bit. In communication, the least significant bit of one byte will be firstly transmitted and the least significant byte of command data sequence will be firstly transmitted.

2. PROTOCOL DESCRIPTION

A communication procedure is to send command and data by the host to the Master first, and then the result status and data is returned by the Master to the host.

The following table shows the process of the host sending command:

Host	Direction	Master	Comment
Command Data Block	→		The interval between two consecutive bytes in the command data block should be less than 15ms. During command data block sending, synchronization will lost if the host receives any data from the master and the host should stop command sending and restart the communication after 15ms.

The command data sent to the master by the host should conform to the format of the protocol. The block including operation command symbol, operation control symbol, command operand data and checksum will be sent to the master, then the host will wait for the result of command execution.

After the master receives the command of the host, the command will be executed in less than around 1.5s. During this time, the master will ignore other commands sent by the host.

The result of the command execution is returned as followed:

Master	Direction	Host	Comment
Response data block	→		The interval between two consecutive bytes in the response data block should be less than 15ms.

The response data block includes command operation symbol, result status and response data. After the feedback, a whole communication process finishes.

3. DATA BLOCK FORMAT

A. COMMAND DATA BLOCK

Head	Len	Parity1	Parity2	Cmd	Data[]

Head: 1 byte block start indicator valued as 0xEB.

Len: 1 byte command data block length. The length of command data block excludes the Head and Parity checksum bytes.

Parity1 and Parity2: 2 bytes parity checksum data.

Cmd: 1 byte command operation symbol data.

Data[]: Command operand data. When Len equals 2, no data[] will be presented.

B. RESPONSE DATA BLOCK

Head	Len	Parity1	Parity2	ReCmd	Data[]

Head: 1 byte block start indicator valued as 0xEB.

Len: 1 byte resonse data block length. The length of response data block excludes the Head and Parity checksum bytes.

Parity1 and Parity2: 2 bytes parity checksum data.

ReCmd: 1 byte command operation symbol data.

Data[]: Command execution result data. When Len equals 2, no data[] will be presented.

When command data block do not conform to defined format, the master will not give out any response.

Parity checksum method:

1. Parity1 + Len =0x7F;
2. Parity2 = {Cmd (ReCmd) xor Data[]} + 0x81

4. DETAILED DESCRIPTION OF OPERATION COMMAND

4.1 Get Version

The host uses this command to get the firmware version of the master.

Head	**Len**	**Parity1**	**Parity2**	**Cmd**	**Data[]**
0xEB	0x02	0x7D	0x9C	0x1B	—
Head	**Len**	**Parity1**	**Parity2**	**ReCmd**	**Data[]**
0xEB	0x03	0x7C	variable	0x1B	Version(1bytes)

4.2 Query Slave

The host uses this command to find out all the slaves connected or cascaded to the master.

Head	Len	Parity1	Parity2	Cmd	Data[]
0xEB	0x02	0x7D	0x19	0x98	—
Head	**Len**	**Parity1**	**Parity2**	**ReCmd**	**Data[]**
0xEB	0x22	0x5D	variable	0x98	Cascade_Info(32bytes)

Cascade_Info：32 bytes indicating the status of slaves connected or cascaded to the master. Each byte of this 32 bytes is defined as:

Bit0=0，no slave presented at Cascade-Level-y on Port-x of the master. (x=0~7, y=1~4)

Bit0=1，slave presented at Cascade-Level-y on Port-x of the master. (x=0~7, y=1~4）

Bit1~Bit7 are reserved as 0.

The master has 8 ports and each ports can be connected with a slave or an antenna. The slave connected to one master port can cascade another 3 slaves. That mean one master port can cascade up to 4 level slaves.

The 32 bytes give out a map of the slaves installed and are defined as:

Byte1~4: cascade level 1~4 of Port-0;
Byte5~8: cascade level 1~4 of Port-1;
Byte9~12: cascade level 1~4 of Port-2;
Byte13~16: cascade level 1~4 of Port-3;
Byte17~20: cascade level 1~4 of Port-4;
Byte21~24: cascade level 1~4 of Port-5;
Byte25~28: cascade level 1~4 of Port-6;
Byte29~32: cascade level 1~4 of Port-7;

For example:

If 2 slaves are cascaded to master port 0, the Cascade_Info should be:

EB 22 5D 19 98 01 01 00 … 00 (30 consecutive 0s)

4.3 Query Antenna

The host uses this command to find out all the antennae connected to master or slaves.

Head	Len	Parity1	Parity2	Cmd	Data[]
0xEB	0x02	0x7D	0x0C	0x8B	—
Head	**Len**	**Parity1**	**Parity2**	**ReCmd**	**Data[]**
0xEB	0x22	0x5D	Variable	0x8B	Ant_Info(32bytes)

Ant_Info：32 bytes indicating the status of antennae connected to the slaves. Each byte of this 32 bytes is defined as: bit0~bit7 represent the antenna port0 to antenna port7 of one slave with value 1 for antenna connected and 0 for no antenna connected.

Since the master has 8 ports, each port can support up to 4 cascaded slave and each slave has 8 antenna port, the whole system can hold up to 256 antennae. 32 bytes Ant_Info are defined as:

Byte1~4: cascade level 4~1 on master port-7;
Byte5~8: cascade level 4~1 on master port-6;
Byte9~12: cascade level 4~1 on master port-5;
Byte13~16: cascade level 4~1 on master port-4;
Byte17~20: cascade level 4~1 on master port-3;
Byte21~24: cascade level 4~1 on master port-2;
Byte25~28: cascade level 4~1 on master port-1;
Byte29~32: cascade level 4~1 on master port-0;

The master port also supports an antenna direct connection. In this situation, it equals the antenna connected to port0 of a cascade-level-1 slave on this master port.

For example, if 2 slaves are cascaded on master's port 0, one antenna is connected to antenna port0 of cascade-level-1 slave and another one antenna is connected to antenna port3 of cascade-level-2 slave, the response data should be EB 22 5D 03 8B 00 … 00 (30 consecutive 0s) 08 01.

4.4 Background scan

The host uses this command to start a background scanning on a designated master port. The scanned information about the slaves and antennae connection can be acquired using Query Slave and Query Antenna command.

This command will take maximum 1.5s.

Head	**Len**	**Parity1**	**Parity2**	**Cmd**	**Data[]**
0xEB	0x03	0x7C	Variable	0x1D	Select_Port(1byte)
Head	**Len**	**Parity1**	**Parity2**	**ReCmd**	**Data[]**
0xEB	0x03	0x7C	0x9E	0x1D	0x00

Select_Port：1 byte to designate the master port with 0 for port0, 1 for port1 and 7 for port7.

4.5 Switch antenna

The host uses this command to select one antenna as active antenna.

The host should use antenna information from Query antenna to switch to a scanned existing antenna. If the host try to switch a not-scanned or non-existing antenna, the command will fail.

Head	**Len**	**Parity1**	**Parity2**	**Cmd**	**Data[]**
0xEB	0x03	0x7C	Variable	0x0A	Select_Ant(1byte)
Head	**Len**	**Parity1**	**Parity2**	**ReCmd**	**Data[]**
0xEB	0x03	0x7C	0x8B	0x0A	Status(1byte)

Select_Ant：1 byte antenna address which is defined as follows:

bit7~bit5: for the master port number the antenna belongs to with 000b~111b as port0~port7.

bit4~bit3: for the cascade level of the slave the antenna belongs to with 00b~11b as

cascade-level-1~cascade-level-4.

bit2~bit0: for the slave port number the antenna belongs to with 000b~111b as port0~port7.

For example, the address of the antenna connected to port 0 of cascade-level-1 slave on master port7 is 0xE0 (11100000b) and the address of the antenna connected to port 3 of cascade-level-2 slave on master port0 is 0x0B (00001011b).

Status：1 byte command execution result with 0x00 for success and 0x08 for failure.

Remark: when command data block conforms to the format but the requested Cmd is not supported, the master will feedback a response as:

Head	Len	Parity1	Parity2	ReCmd	Data[]
0xEB	0x03	0x7C	0xE9	0x6C	0x04