Bluetooth Lock Communication Protocol

Kerong Industry (HK) Co., Ltd.

R&D department

17th June, 2019 Monday

Content

Version 1
Abstract 2
1. Detailed description of Bluetooth lock protocol 3
1.1 Hardware protocol 3
1.2 Software protocol 3
1.2.1 Protocol data packet 3
2. Command protocol table 4
3. Response protocol 5
3.1 Response value 5
4. Introduction of Bluetooth command (mobile APP==> lock) 5
4.1 Pairing password verification 6
a) Protocol 6
b) Command function 6
c) DATA format 6
d) Example 6
e) DATA analysis 6
4.2 Get random code. 6
a) Protocol 6
b) Command function 6
c) DATA format 6
d) Example 6
e) DATA analysis 6
4.3 Authority authentication. 7
a) Protocol 7
b) Command function 7
c) DATA format 7
d) Example 8
e) DATA analysis 8
f) DATA simple encryption and decryption function 8
4.4 Status checking 9
a) Protocol 9
b) Command function 9
c) DATA format 9
d) Example 9
e) DATA analysis 9
4.5 Unlock & lock 9
a) Protocol 9
b) Command function 9
c) DATA format 9
d) Example 9
e) DATA analysis 10
4.6 Time setting 10
a) Protocol 10
b) Command function 10

c) DATA format 10
d) Example 10
e) DATA analysis 10
4.7 Time reading 10
a) Protocol 10
b) Command function 10
c) DATA format 10
d) Example 11
e) DATA analysis 11
4.8 Locking mode setting & checking 11
a) Protocol 11
b) Command function 11
c) DATA format 11
d) Example 11
e) DATA analysis 11
4.9 System initialization 11
a) Protocol 11
b) Command function 11
c) DATA format ..... - 11 -
d) Example 12
e) DATA analysis 12
4.10 Lock ID setting & checking 12
a) Protocol 12
b) Command function 12
c) DATA format ..... - 12 -
d) Example 12
e) DATA analysis ..... - 12 -
4.11 Get Bluetooth module MAC address 12
a) Protocol 12
b) Command function 12
c) DATA format ..... - 12 -
d) Example 12
e) DATA analysis ..... - 13 -
4.12 User code application 13
a) Protocol 13
b) Command function 13
c) DATA format ..... - 13 -
d) Example 13
e) DATA analysis ..... - 14 -
4.13 Manager code checking & changing ..... - 14 -
a) Protocol 14
b) Command function 14
c) DATA format ..... - 14 -
d) Example 14
e) DATA analysis ..... - 15 -
4.14 User code checking & changing 15

a) Protocol 15
b) Command function 16
c) DATA format ..... - 16 -
d) Example 16
e) DATA analysis 17
4.15 Delete user code 17
a) Protocol 17
b) Command function 17
c) DATA format 17
d) Example 18
e) DATA analysis 18
4.16 Load user information 18
a) Protocol 18
b) Command function 18
c) DATA format 18
d) Example 19
e) DATA analysis 19
4.17 Unlocking mode setting & checking 19
a) Protocol 19
b) Command function 19
c) DATA format ..... - 20 -
d) Example 20
e) DATA analysis ..... - 20 -
4.18 Get current version No. 20
a) Protocol 20
b) Command function 20
c) DATA format ..... - 20 -
d) Example 20
e) DATA analysis ..... - 21 -
4.19 System exit 21
a) Protocol 21
b) Command function 21
c) DATA format ..... - 21 -
d) Example 21
4.20 Get user amount 21
a) Protocol 21
b) Command function 21
c) DATA format ..... - 21 -
d) Example 21
e) DATA analysis ..... - 21 -
4.21 Get log information 22
a) Protocol 22
b) Command function 22
c) DATA format ..... - 22 -
d) Example 22
e) DATA analysis ..... - 22 -

4.22 Load log 22
a) Protocol 22
b) Command function 22
c) DATA format ..... - 22 -
d) Example 24
e) DATA analysis 24
4.23 Clear log 24
a) Protocol 25
b) Command function ..... - 25 -
c) DATA format ..... - 25 -
d) Example 25
e) DATA analysis 25
4.24 Alarm setting & checking ..... - 25 -
a) Protocol 25
b) Command function 25
c) DATA format 25
d) Example 25
e) DATA analysis 26
4.25 Buzzer setting & checking 26
a) Protocol 26
b) Command function 26
c) DATA format 26
d) Example 26
e) DATA analysis 26
5. Detailed introduction of Bluetooth command (Bluetooth lock==> mobile APP) 27
5.1 Loading status 27
a) Protocol 27
b) Command function 27
c) DATA format 27
d) Example 27
e) DATA analysis 27

Document version

Time	Version	Author	Details
5th March, 2019, Tuesday	V3.0	Wang Yonglin	Establishment of the document
17th June, 2019, Monday	V3.1	Wang Yonglin	1. Add “pairing password verification” command 2. Add “alarm setting & checking” command 3. Add “buzzer setting & checking” command

Abstract

The manual is an introduction to the Bluetooth lock communication protocol.
Section I: Detailed introduction of Bluetooth lock software communication protocol.
section II: Instructions for Bluetooth lock hardware interface.

1. Detailed description of Bluetooth lock protocol

1.1 Hardware protocol

Support serial communication

Item	Description
Baud rate	19200
Data bit	8 bits
Check bit	None
Stop bit	1 bit

1.2 Software protocol

Protocol data packet:

Maximum length of protocol packet is 128;

Minimum length of protocol packet is 6;

Packet format is Big-endian;

Protocol packet format: the format of command protocol and response protocol is the same. General format is below:

No.	Item	Description
1	STX（1byte）	Data head / frame header, fixed value: 0xF5
2	CMD（1byte）	Command, please refer to the protocol command table
3	ASK（1byte）	Response value, please refer to the response table
4	DATALEN（1byte）	Data length
5	ETX（1byte）	Data footer / frame footer, fixed value: 0x5F
6	SUM（1byte）	Check byte and low byte of whole command packet, e.g., if sum of all data is 0x125D, then sum=0x5D
... 6+DATALEN	DATA（DATALEN bytes）	DATA, if the value of DATALEN is zero, the data is null. The format is different for different commands. Please refer to the Introduction of Commands.

2. Command protocol table

No.	Command	Protocol	Description
Bluetooth communication command（Mobile APP==> Bluetooth lock）			
Pairing password verification			
1	**Pairing password verification**	0x0F	If the verification succeeds,then can communicate with Bluetooth lock.
Encryption authority authentication			
1	Get random code	0x20	Get random code, for data encryption of authority authentication command.
2	Authority authentication	0x21	Verify authority of current user.
If authority authentication is not successful, then cannot operate follow commands.			
1	Status checking	0x60	Lock status, lock hook status, power voltage, lock ID
2	Unlock & lock	0x61	Unlocking & locking command
3	Time setting	0x62	Set lock time
4	Time reading	0x63	Get lock time
5	Locking mode setting & checking	0x64	Set & check locking mode (factory is automatic, manual)
6	System initialization	0x65	Clear system data, return to factory setting(If successful, Bluetooth connection will be disconnected automatically)
7	Set & check lock ID	0x66	Set & check lock ID (If successful, Bluetooth connection will be disconnected automatically)
8	Get Bluetooth module MAC address	0x67	Get Bluetooth module MAC address
9	User code application	0x68	Apply periodical user code or one time password
10	Manager code checking & changing	0x69	Check & change manager code
11	User code checking & changing	0x6A	Check & change use code
12	Delete user information	0x6B	Delete user code information
13	Upload user information	0x6C	Upload all user information in the current system
14	Unlocking mode setting & checking	0x6D	Set & check unlocking mode (factory is automatic, manual)
15	Get current version No.	0x6E	Software version, hardware version
16	System exit	0x6F	Disconnect with Bluetooth, enter hibernation
17	Check user code amount	0x70	Check user code amount in current system (Include partially invalid password)
18	Get log information	0x71	Get amount of current logs and the latest log index
19	Load log	0x72	Load all logs in current system
20	Clear log	0x73	Clear all logs in current system
21	Alarm setting & checking	0x74	Set & check alarm status

22	Buzzer setting & checking	0x75	Set & check buzzer
Bluetooth communication protocol (Bluetooth lock==> mobile APP			
1	Uploading status	0x80	Upload lock status automatically after locked & unlocked.
Remark			

3. Response protocol

3.1 Response value

Value	Description
0x10	Correct operation
0x11	Wrong operation
0x12	Timeout
0x13	Unknown command
0x16	Fail checksum
0x17	Unauthenticated authority
0x18	Unauthorized or failed account
0x19	Wrong password
0x1A	Invalid OTP
0x1B	Manager code and user code cannot be the same
0x1C	Invalid application of password time
0x1D	Fail user information saving
0x1E	It is not the time to start user code
0x1F	Upper limit of user application quantity
0x22	No user information in lock system
0x23	New password format error
0x24	Information in loading
0x25	No log in system
0x26	Pairing password is not verified
0x27	Failed pairing password verification
0x00	ASK default value when send command

4. Introduction of Bluetooth commands (mobile phone APP>> Bluetooth lock)

This section explains details about commands sent from mobile phones APP to Bluetooth locks, each command has its own unique data format. Therefore, a detailed description about DATA will be given in this section.

4.1 Pairing password verification

a) Protocol

Command: 0x0F

b) Command function

Before communicate with Bluetooth lock, need to verify the pairing password, or Bluetooth lock will disconnect automatically 30s later.

c) DATA format

```
char  LinkPwd[4];          //Bluetooth pairs connecting password（number ASCII value）
```

d) Example

Mobile APP sends data: F5 0F 00 04 5F 3B 39 31 35 35
Mobile APP receive data: F5 0F 10 00 5F 73

e) DATA analysis

1. 39 31 35 35 means the input pairing password, it is “9155”.
2. Normally, lock ID is different, the pairing password will also be different.
3. Pairing password of every lock is a fixed ACSII value with four digits.
4. If pairing password verification is failed, then cannot operate other functions of Bluetooth lock.
5. Every time connect with Bluetooth lock, need to verify the pairing password with 30s, or Bluetooth lock will disconnect automatically.
6. First time mobile APP verifies pairing password successfully, then can cache that pairing password. Next time connecting with Bluetooth lock, mobile APP can search the pairing password, and send verification command.

4.2 Get random code

a) Protocol

Command: 0x20

b) Command function

Get a random code for data encryption and decryption of authority authentication. The random code will update every time wake up system.

c) DATA format

```
u8 RandData;   // random code
```

d) Example

Mobile APP sends data: F5 20 00 00 5F 74
Mobile APP receives data: F5 20 10 01 5F 46 C1

e)DATA analysis

1. C1 means random code

4.3 Authority authentication

a) Protocol

Command: 0x21

b) Command function

Authenticate whether the connected account and password is valid, transfer the DATA in encrypted way.

c) DATA format

```
typedef union TagPhoneNumber
{
    u8    bytes[6];
        struct    //take the telephone number 15814015470 for example (if the number is odd,  then the forefront should be filled with zero)
    {
      u8 OneByte;          // 0x01
      u8 TwoByte;          // 0x58
      u8 THREEByte;        // 0x14
      u8 FourByte;         // 0x01
      u8 FiveByte;         // 0x54
      u8 SixByte;          // 0x70
    }Datas;
}TPhoneNumberDatas, *PPhoneNumberDatas;
typedef union TagAuthenticationDatas
{
    u8    bytes[13];
    struct
    {
      u8 AuthenticationDatasType;     //0x01  Administrator authentication  0x02 User authentication  0x04 One time password authentication for emergency unlock
      TPhoneNumberDatas  PhoneNumberDatas;  //the user's phone number must in the hexadecimal form.
      u8 PassWord[6];                 //Password, fixed six digits, expressed by character string, for example, the character string "111111" indicates that the password is six individual 1
    }Datas;
    struct
    {
      u8 AuthenticationDatasType;     //0x01 Administrator authentication  0x02 User authentication  0x04 One time password authentication for emergency unlock
      u8 KeeLoq[12]; //OTP, fixed with 10 digits, expressed as a character string, for example, number entered in the APP interface is 0123456789. The OTP is a character string "0123456789"
    } KeeLoqDatas;
}TAuthenticationDatas, *PAuthenticationDatas;
```

d) Example

Administrator login:

Mobile APP sends data: F5 21 00 0D 5F 1C C0 C0 99 D5 C0 95 B1 F1 F1 F1 F1 F1 F1

Mobile APP receives data: F5 21 10 00 5F 85

User login:

Mobile APP sends data: F5 21 00 0D 5F 31 C3 C0 99 D5 C0 95 B0 F9 F4 F2 F1 F0 F9

Mobile APP receives data: F5 21 10 00 5F 85

OTP login:

Mobile APP sends data: F5 21 00 0D 5F 47 C5 F7 F7 F3 F1 F0 F7 F1 F1 F1 F2 C1 C1

Mobile APP receives data: F5 21 10 00 5F 85

e)DATA analysis

1. C0 C0 99 D5 C0 95 B1 F1 F1 F1 F1 F1 F1 stands for the encrypted administrator account information, the decrypt data is “0101 58 14 01 54 70 30 30 30 30 30 30” , therefore, the administrator account is “15814015470” , and the management password is "000000";
2. C3 C0 99 D5 C0 95 B0 F9 F4 F2 F1 F0 F9 stands for encrypted user account information, the decrypt data is “02 01 58 14 01 54 7138 35 33 30 31 38”, therefore, the user account is “15814015471”, and the user password is “853018";
3. C5 F7 F7 F3 F1 F0 F7 F1 F1 F1 F2 C1 C1 stands for encrypted authentication information of the OTP, the decrypt data is “0436 36 32 30 31 36 30 30 30 33 00 00”, so the OTP(one time password) is “6620160003” .

Notice:

1. About the design of mobile APP, lock time should be verified firstly after administrator being authorized to login the lock system;
2. Lock system will be initialized and returns to the factory set after login by the OTP.

f) Simple data encryption and decryption function

```
u8* Encrypt(u8* PlainData,u8 Key,u8 len)    //data encryption
{
   u8 i;
   for(i=0;i<len;i++)
   {
      PlainData[i] =(PlainData[i]^Key);
   }

   return PlainData;
}
u8* Dncrypt(u8* CipherData,u8 Key,u8 len)    //data decryption
{
    u8 i;
    for(i=0;i<len;i++)
   {
      CipherData[i] =(CipherData[i]^Key);
   }
   return CipherData;
}
```

4.4 Status checking

a) Protocol

Command: 0x60

b) Command function

Get lock status(unlocked & locked), lock tongue status, power voltage, lock ID, unlocking mode, locking mode.

c) DATA format

```
 struct
{
    u8 OpenCloseLockState;   //00 Locked     01 Unlocked
    u8 HookState;            //00 lock hook is not locked   01 lock hook is locked
    u16 VotageValue;         //Voltage of battery, unit: mV
    u32          LockId;     //lock ID     0x00000001~0xFFFFFFFF
    u8 OpenLockMode;         //Unlocking mode   00   Unlock manually     01 Unlock automatically
    u8 CloseLockMode;        //Locking mode   00   Lock automatically     01 Lock manually
}LockStatesDatas;
```

d) Example

Mobile APP sends data: F5 60 00 00 5F B4

Mobile APP receives data: F5 60 10 0A 5F BB 00 00 16 D5 00 00 00 02 00 00

e) DATA analysis

1. 00 indicates that the lock is in the locked status (i.e., lock tongue stretches out);
2. 00 indicates that the lock hook is not locked (i.e., lock hook is not into lock tongue);
3. 16 D5 indicates the voltage value of the lock, 0x16D5=5845, therefore, the voltage of the lock is 5845mV;
4. 00 00 00 02 represents the lock ID number, 0x00000002=2, therefore, the lock's ID number is 2;
5. 00 indicates the unlocking mode, currently is unlock automatically;
6. 00 indicates the locking mode, currently is lock automatically

4.5 Lock & unlock

a) Protocol

Command: 0x61

b) Command function

Unlock & lock.

c) DATA format

```
u8   OpenCloseLock;   //0x34: means unlock & lock   other value is wrong, no operation
```

d) Example

Mobile APP sends data: F5 61 00 01 5F EA 34

Mobile APP receives data: F5 61 10 01 5F FA 34

e) DATA analysis

1. 34 is the parameters for unlocking, other values are meaningless
2. If the current mode is locking automatically (factory mode), then it will be automatically locked after unlocking for 5 seconds; if the current mode is locking automatically, then the locking status will execute the unlocking action, the unlocking status will execute the locking action;
3. The Bluetooth lock will upload the lock's status automatically after locking or unlocking (see 0x80 command)

4.6 Time setting

a) Protocol

Command: 0x62

b) Command function

Set lock time（BCD code）

c) DATA format

```
//////////////////////////Time data structure////////////////////////////////////
typedef struct tagDateTime
{       u8 wYear;
        u8 wMonth;
        u8 wDay;
        u8 wHour;
        u8 wMinute;
        u8 wSec;
}TDateTime, * PDateTime;
```

d) Example

Mobile APP sends data: F5 62 00 06 5F 2B 18 08 16 11 28 00

Mobile APP receives data: F5 62 10 00 5F C6

e) DATA analysis

18 08 16 11 28 00 means the lock time was set as 11:28:00, 16th Aug, 2018

4.7 Time reading

a) Protocol

Command: 0x63

b) Command function

Read the current lock time（BCD code）

c) DATA format

```
//////////////////////////Time data structure////////////////////////////////////
typedef struct tagDateTime
{       u8 wYear;
        u8 wMonth;
        u8 wDay;
```

```
    u8 wHour;
    u8 wMinute;
    u8 wSec;
}TDateTime, * PDateTime;
```

d) Example

Mobile APP sends data: F5 63 00 00 5F B7
Mobile APP receives data: F5 63 10 06 5F 3C 18 08 16 11 28 00

e) DATA analysis

18 08 16 11 28 00 means the lock time is 11:28:00, 16th Aug, 2018

4.8 Locking mode setting & checking

a) Protocol

Command: 0x64

b) Command function

Set and check the locking mode, when DATALEN=0x00, it represents for checking; when DATALEN=0x01 , it represents for setting

c) DATA format

```
u8 CloseLockModeParameter;    //Locking mode parameters: 0 means lock automatically    1 means lock manually
```

d) Example

Checking:
Command: F5 64 00 00 5F B8
Response: F5 64 10 01 5F CA 01
Setting:
Command: F5 64 00 01 5F BA 01
Response: F5 64 10 00 5F C8

e) DATA analysis

01 means locking mode is lock manually.

4.9 System initialization

a) Protocol

Command: 0x65

b) Command function

Clear user information and log, reset manager code, system returns to factory setting.

c) DATA format

None

d) Example

Command: F5 65 00 00 5F B9
Response: F5 65 10 00 5F C9

e) DATA analysis

None

4.10 Lock ID setting & checking

a) Protocol

Command: 0x66

b) Command function

Set & check lock ID, when DATALEN=0x0 represents for checking; when DATALEN=0x04 represents for setting.

c) DATA format

```
u32        LockId;        //Lock ID No.
```

d) Example

Checking:
Command: F5 66 00 00 5F BA
Response: F5 66 10 04 5F CF 00 00 00 01
Setting:
Command: F5 66 00 04 5F BF 00 00 00 01
Response: F5 66 10 00 5F CA

e) DATA analysis

00 00 00 01 means lock ID No., 0x00000001=1, so lock ID No. Is 1.
Notice: This command setting function is only for internal use by the manufacturer.

4.11 Get Bluetooth module MAC address

a) Protocol

Command: 0x67

b) Command function

Get Bluetooth module MAC address.

c) DATA format

```
char    MAC[6];    // Bluetooth module MAC address
```

d) Example

Checking:
Command: F5 67 00 00 5F BB
Response: F5 67 10 06 5F 6B E3 FA F4 E7 F5 ED

e) DATA analysis

E3 FA F4 E7 F5 ED stands for the Mac Address of the Bluetooth module (6 bytes hexadecimal data).

4.12 User code application

a) Protocol

Command: 0x68

b) Command function

This command is for administrator to apply user code.

c) DATA format

```
typedef union Tagtimes // take 21:30, 6th March, 2019 for example
{
        u8    bytes[5];
        struct
        {
          u8 YEAR;      //0x19
          u8 MON;       //0x03
          u8 DAY;       //0x06
          u8 HOUR;      //0x21
          u8 MIN;       //0x30
        }Datas;
}Ttime_Datas, *Ptime_Datas;
typedef union TagPhoneNumber
{
        u8    bytes[6];
        struct    //take the telephone number 15814015470 for example (if the number is odd,    then the forefront should be filled with zero)
        {
          u8 OneByte;            // 0x01
          u8 TwoByte;            // 0x58
          u8 THREEByte;          // 0x14
          u8 FourByte;           // 0x01
          u8 FiveByte;           // 0x54
          u8 SixByte;            // 0x70
        }Datas;
}TPhoneNumberDatas, *PPhoneNumberDatas;
typedef union TagApplyUserPwdDatas
{
        u8    bytes[17];
        struct
        {
          u8 ApplyUserPwdDatasType;                    //0x02    periodical user code application          0x03 one time password application              TPhoneNumberDatas    PhoneNumberDatas;    // user's mobile phone number, must in hexadecimal form
```

```
Ttime_Datas StartDateTime;                    /password effective time
        Ttime_Datas EndDateTime;                    //password expiration time
    }Datas;
}TApplyUserPwdDatas, *PApplyUserPwdDatas;
TApplyUserPwdDatas    ApplyUserPwdDatas;
u8 PassWord[6];            //password, must with 6 digits, which should be presented in the form of character string. For example, the character string“111111”stands for the password is “111111”
```

d) Example

Checking:

Command: F5 68 00 11 5F 99 02 01 58 14 01 54 71 19 03 13 11 08 20 03 13 11 08

Response: F5 68 10 06 5F 07 35 33 35 33 31 34

e) DATA analysis

1. 02 01 58 14 01 54 71 19 03 13 11 08 20 03 13 11 08 means apply for user password,
02 means apply for periodical user password
01 58 14 01 54 71 means user account number is: 15814015470;
19 03 13 11 08 20 03 13 11 08 means the valid time for periodical user password is from 11:08, 13th Mar, 2019 to 11:08, 13th Mar, 2020
2. 35 33 35 33 31 34 means the user password is “535314 ".

Notice:
1. If the applied user account is already existed in the lock system, then it will replace the original account information.
2. The new applied account will preferentially replace the storage location of the deleted or expired account.

4.13 Manager code checking & changing

a) Protocol

Command: 0x69

b) Command function

This command is for checking & changing manager code.

c) DATA format

```
typedef union TagPhoneNumber
{
    u8    bytes[6];
    struct    //take the telephone number 15814015470 for example (if the number is odd,    then the forefront should be filled with zero)
    {
      u8 OneByte;          // 0x01
      u8 TwoByte;          // 0x58
      u8 THREEByte;        // 0x14
      u8 FourByte;         // 0x01
      u8 FiveByte;         // 0x54
      u8 SixByte;          // 0x70
```

```
        }Datas;
}TPhoneNumberDatas, *PPhoneNumberDatas;
typedef union TagAdminPwdDatasInfos
{
        u8    bytes[12];
        struct
        {
          TPhoneNumberDatas    PhoneNumberDatas;    //user's mobile phone number must in the hexadecimal form
          u8 PassWord[6];                           //new password must with 6 digits which should be presented in the form of character string, for example, the character string"111111"stands for the password with six individual 1
        }Datas;
}TAdminPwdDatasInfos, *PAdminPwdDatasInfos;
typedef union TagModifyAdminPwdDatas
{
        u8    bytes[12];
        struct
        {
         u8 OldPassWord[6];                         //original password with 6 digits which should be presented in the form of character string, for example, the character string"111111"stands for the password with six individual 1
  u8 NewPassWord[6];                                //new password with 6 digits which should be presented in the form of character string, for example, the character string"111111"stands for the password with six individual 1
        }Datas;
}TModifyAdminPwdDatas, *PModifyAdminPwdDatas;
//////////////////////////////////////////Check or change manager code//////////////////////////////////////////////
            TAdminPwdDatasInfos AdminPwdDatasInfos;
            TModifyAdminPwdDatas ModifyAdminPwdDatas;
```

d) Example

Checking:

Command: F5 69 00 00 5F BD

Response: F5 69 10 0C 5F 2B 01 58 14 01 54 70 30 30 30 30 30 30

Changing:

Command: F5 69 00 0C 5F 2D 30 30 30 30 30 30 36 36 36 36 36 36

Response: F5 69 10 18 5F 4F 01 58 14 01 54 70 36 36 36 36 36 36

e) DATA analysis

1. 01 58 14 01 54 70 30 30 30 30 30 30 represents the management account information need to be checked,
01 58 14 01 54 70 means the management account number is 15814015470,
30 30 30 30 30 30 means the management password is "000000"

2. 30 30 30 30 30 30 36 36 36 36 36 36 represents modifying management password,
30 30 30 30 30 30 stands for the original management password "000000 ",
36 36 36 36 36 36 stands for the new management password is "666666 "

3. 01 58 14 01 54 70 36 36 36 36 36 36 represents the management account information need to be modified,
01 58 14 01 54 70 indicates that the management account number is15814015470 ,
36 36 36 36 36 36 indicates that the management password is "666666"

4.14 User code checking & changing

a) Protocol

Command: 0x6A

b) Command function

Check & change user code.

c) DATA format

```
typedef union TagPhoneNumber
{
        u8    bytes[6];
        struct    //take the telephone number 15814015470 for example (if the number is odd,    then the forefront should be filled with zero)          {
           u8 OneByte;             // 0x01
           u8 TwoByte;             // 0x58
           u8 THREEByte;          // 0x14
           u8 FourByte;           // 0x01
           u8 FiveByte;           // 0x54
           u8 SixByte;            // 0x70
        }Datas;
}TPhoneNumberDatas, *PPhoneNumberDatas;
typedef union TagModifyUserPwdDatas
{
        u8    bytes[18];
        struct
        {
           TPhoneNumberDatas    PhoneNumberDatas;    //User mobile phone number must in hexadecimal form
           u8 OldPassWord[6];             //Original password with 6 digits which should be presented in the form of character string, for example, the
character string“111111”stands for the password with six individual 1
  u8 NewPassWord[6];            //New password with 6 digits which should be presented in the form of character string, for example, the character
string“111111”stands for the password with six individual 1
    }Datas;
}TModifyUserPwdDatas, *PModifyUserPwdDatas;
#define    USER_PWD_DATA_SIZE          24
typedef struct tagUserPwdDataCheckSum
{
        u8              bytes[USER_PWD_DATA_SIZE-1];
        u8              Sum;                          // Sum
}TUserPwdDataCheckSum;
// User information structure
typedef union tagUserPwdInfo
{
           u8 buff[USER_PWD_DATA_SIZE];
        struct
        {
```

```
    u8 UserPwdDatasType;   //0x02 Periodical user code    0x03 One time password    0x82 、invalid periodical user code   0x83 invalid one time password
    TPhoneNumberDatas   PhoneNumberDatas;   //User mobile phone number must in hexadecimal form
    u8 PassWord[6];                         //The password with 6 digits which should be presented in the form of character string, for example, the character string“111111”stands for the password with six individual 1
    Ttime_Datas StartDateTime;              //password effective time
    Ttime_Datas EndDateTime;                //password expiration time
    }Datas;
    TUserPwdDataCheckSum CheckSum;          // check data
}TUserPwdInfo, *PUserPwdInfo;
//////////////////////////////////////Check & change user code//////////////////////////////////////////
TPhoneNumberDatas   PhoneNumberDatas;   //User mobile phone number must in hexadecimal form
TModifyUserPwdDatas     ModifyUserPwdDatas;//Change user code
TUserPwdInfo UserPwdInfos;
```

d) Example

Check:

Command: F5 6A 00 06 5F F7 01 58 14 01 54 71

Response: F5 6A 10 18 5F E0 02 01 58 14 01 54 71 35 31 37 33 31 30 19 03 13 11 08 20 03 13 11 08 FD

Change:

Command: F5 6A 00 12 5F 78 01 58 14 01 54 71 35 31 37 33 31 30 36 36 36 36 36 36

Response: F5 6A 10 18 5F 06 02 01 58 14 01 54 71 36 36 36 36 36 36 19 03 13 11 08 20 03 13 11 08 10

e) DATA analysis

1. 101 58 14 01 54 71 represents the user account need to be checked;
2. 02 01 58 14 01 54 71 35 31 37 33 31 30 19 03 13 11 08 20 03 13 11 08 FD represents the user code information
02 indicates that the user is a periodical user, and user account number is 015814015471, user password is "517310 ", valid user time is from 11:08, 13th Mar, 2019 to 11:08, 13th Mar, 2020,
FD is the checksum of user data;
3. 01 58 14 01 54 71 35 31 37 33 31 30 36 36 36 36 36 36, the account number need to be modified is 015814015471, the original password is “517310”, and the new password is "666666"
4. 02 01 58 14 01 54 71 36 36 36 36 36 36 19 03 13 11 08 20 03 13 11 08 10 represents user code information after modified

02 indicates that the user is a periodical user, and user account number is 015814015471, user code is " 666666", valid user time is from 11:08, 13th Mar, 2019 to 11:08, 13th Mar, 2020

10 is the checksum of user data

4.15 Delete user code information

a) Protocol

Command: 0x6B

b) Command function

This command is for deleting specific user code information or all user code information.

c) DATA format

```
typedef union TagPhoneNumber
```

```
{
    u8   bytes[6];
    struct   //take the telephone number 15814015470 for example (if the number is odd,   then the forefront should be filled with zero)
    {
      u8 OneByte;          // 0x01
      u8 TwoByte;          // 0x58
      u8 THREEByte;        // 0x14
      u8 FourByte;         // 0x01
      u8 FiveByte;         // 0x54
      u8 SixByte;          // 0x70
    }Datas;
}TPhoneNumberDatas, *PPhoneNumberDatas;
//////////////////////////////////////Delete user code information//////////////////////////////////////////////
TPhoneNumberDatas   PhoneNumberDatas;   //User's mobile phone number must in hexadecimal form
```

d) Example

Delete specific user code information:

Command: F5 6B 00 06 5F F8 01 58 14 01 54 71

Response: F5 6B 10 00 5F CF

Delete all user code information:

Command: F5 6B 00 00 5F BF

Response: F5 6B 10 00 5F CF

e) DATA analysis

01 58 14 01 54 71 means the specific account needs to be deleted is 15814015470.

4.16 Upload user information

a) Protocol

Command: 0x6C

b) Command function

This command is for uploading all user information in the system.

c) DATA format

```
#define   USER_PWD_DATA_SIZE          24
typedef struct tagUserPwdDataCheckSum
{
    u8          bytes[USER_PWD_DATA_SIZE-1];
    u8          Sum;                        // Sum
}TUserPwdDataCheckSum;
// User information structure
typedef union tagUserPwdInfo
{
      u8 buff[USER_PWD_DATA_SIZE];
```

```
    struct
    {
      u8 UserPwdDatasType; //0x02 periodical user code  0x03 one time password  0x82 、invalid periodical user code  0x83 invalid one time password
       TPhoneNumberDatas  PhoneNumberDatas;  //User's mobile phone number must in hexadecimal form
       u8 PassWord[6];                       //The password with 6 digits which should be presented in the form of character string, for example, the character string"111111"stands for the password with six individual 1
       Ttime_Datas StartDateTime;            //password effective time
      Ttime_Datas EndDateTime;               //password expiration time
    }Datas;
    TUserPwdDataCheckSum CheckSum;      // check data
}TUserPwdInfo, *PUserPwdInfo;
//////////////////////////////////////////User code information/////////////////////////////////////////////////
  TUserPwdInfo UserPwdInfos;
```

d) Example

Command: F5 6C 00 00 5F C0

Response:

F5 6C 24 78 5F A8 02 01 58 14 01 54 72 31 38 35 35 31 39 19 03 13 11 08 20 03 13 11 08 0A
02 01 58 14 01 54 71 30 31 34 31 33 39 19 03 13 11 08 20 03 13 11 08 FE
02 01 58 14 01 54 73 33 32 34 33 34 39 19 03 13 11 08 20 03 13 11 08 07
02 01 58 14 01 54 74 35 30 32 37 38 37 19 03 13 11 08 20 03 13 11 08 0C
02 01 58 14 01 54 75 32 36 39 36 30 34 19 03 13 11 08 20 03 13 11 08 0B
F5 6C 10 18 5F A8 02 01 58 14 01 54 76 33 38 36 38 38 34 19 03 13 11 08 20 03 13 11 08 16

e) DATA analysis

1. 24 indicates that the data is in transit, followed by the response data packet
2. 02 01 58 14 01 54 72 31 38 35 35 31 39 19 03 13 11 08 20 03 13 11 08 0A
   02 01 58 14 01 54 71 30 31 34 31 33 39 19 03 13 11 08 20 03 13 11 08 FE
   02 01 58 14 01 54 73 33 32 34 33 34 39 19 03 13 11 08 20 03 13 11 08 07
   02 01 58 14 01 54 74 35 30 32 37 38 37 19 03 13 11 08 20 03 13 11 08 0C
   02 01 58 14 01 54 75 32 36 39 36 30 34 19 03 13 11 08 20 03 13 11 08 0B
   indicates five accounts information data are transmitted
   (Notice: one packet can only transmit 5 items at most)
3. 10 indicates that the data is loaded successfully
4. 02 01 58 14 01 54 76 33 38 36 38 38 34 19 03 13 11 08 20 03 13 11 08 16is the last user information in the lock system

Notice: Deleted user information in the lock system will not be loaded and transferred.

4.17 Unlocking mode setting & checking

a) Protocol

Command: 0x6D

b) Command function

Set & check lock unlocking mode, when DATALEN=0x00, represents for checking; when DATALEN=0x01, re[resents for setting.

c) DATA format

u8 OpenLockModeParameter; //unlocking mode parameter：0 indicates manual unlocking mode1 1 indicates automatic unlocking mode

d) Example

Checking:
Command: F5 6D 00 00 5F C1
Response: F5 6D 10 01 5F D2 00
Setting:
Command: F5 6D 00 01 5F C3 01
Response: F5 6D 10 00 5F D1

e) DATA analysis

1. 00 indicates that the unlocking mode of current lock is manual unlocking mode
2. 01 indicates that the unlocking mode of the current lock is automatic unlocking mode

Notice: In the automatic mode, the broadcasting will be stopped after the lock system hibernation, and it must be awakened by an external key. Then user mobile phone APP will be authorized to login automatically after it is connected with Bluetooth. If Bluetooth lock is in the login successful status, every touch of the key, the lock will lock or unlock for once.

4.18 Get current version No.

a) Protocol

Command: 0x6E

b) Command function

Get current software version No. and hardware version No.

c) DATA format

```
//////////////////Version//////////////////////
        struct
        {
            u8 softversion;
            u8 hardversion;
        }version;
```

d) Example

Mobile APP sends data: F5 6E 00 00 5F C2
Mobile APP receives data: F5 6E 10 02 5F DA 03 03

e) DATA analysis

1. 03 means current software version No. is 0.3
2. 03 means current hardware version No. is 0.3

4.19 System exit

a) Protocol

Command: 0x6F

b) Command function

Lock exits the connection mode and enter dormant status.

c) DATA format

None

d) Example

Mobile APP sends data: F5 6F 00 00 5F C3
Mobile APP receives data: F5 6F 10 00 5F D3

4.20 Get user amount

a) Protocol

Command: 0x70

b) Command function

Lock exits the connection mode and enter dormant status

c) DATA format

```
u8          PwdCount;          // User code amount
```

d) Example

Mobile APP sends data: F5 70 00 00 5F C4
Mobile APP receives data: F5 70 10 01 5F D5 00

e) DATA analysis

00 indicates that there is no user in the current lock system.

4.21 Get log information

a) Protocol

Command: 0x71

b) Command function

Get log amount and log pointer in the current system.

c) DATA format

```
//////////////////Log information structure//////////////////////
  typedef struct tagLockLogInfo
{
        u8 bytes[4];
      struct
      {
         u16   LogCount;          // Log amount
         u16   LogPointer;        // Log pointer
      }Datas;

}TLockLogInfo;;
```

d) Example

Mobile APP sends data: F5 71 00 00 5F C5
Mobile APP receives data: F5 71 10 04 5F DA 00 01 00 00

e) DATA analysis

1. 00 01 means log amount in current system is 1.
2. 00 00 means the latest log pointer points at 0.

4.22 Upload log

a) Protocol

Command: 0x72

b) Command function

This command is for loading all logs in the lock system.

c) DATA format

```
typedef union Tagtimes //take 21:30, 6th   Mar, 2019 for example
{
      u8   bytes[5];
```

```
        struct
        {
          u8 YEAR;     //0x19
          u8 MON;      //0x03
          u8 DAY;      //0x06
          u8 HOUR;     //0x21
          u8 MIN;      //0x30
        }Datas;
}Ttime_Datas, *Ptime_Datas;
typedef union TagPhoneNumber
{
        u8   bytes[6];
        struct   //take the telephone number 15814015470 for example (if the number is odd,   then the forefront should be filled with zero)
        {
          u8 OneByte;          // 0x01
          u8 TwoByte;          // 0x58
          u8 THREEByte;        // 0x14
          u8 FourByte;         // 0x01
          u8 FiveByte;         // 0x54
          u8 SixByte;          // 0x70
        }Datas;
}TPhoneNumberDatas, *PPhoneNumberDatas;
#define   LOG_DATA_SIZE          13
typedef struct tagLogDataCheckSum
{
        u8          bytes[LOG_DATA_SIZE-1];
        u8          Sum;                              //Sum
}TLogDataCheckSum;
// User information structure
typedef union tagLockLog
{
          u8 buff[LOG_DATA_SIZE];
        struct
        {
                u8 LogType;     //0x01 management code unlock and lock   0x02   periodical user code unlock and lock   0x03 one time password unlock
and lock
   //0x11 management code initialization     0x14 one time password initialization       0x15   button for initialization
                                //0x41 input management code through keyboard to unlock and lock     0x42 input periodical user code through keyboard
to unlock and lock   0x43   input one time password through keyboard to unlock and lock
                TPhoneNumberDatas   PhoneNumberDatas;   //user's mobile phone number must in the hexadecimal form
               Ttime_Datas DateTime;                    //operation time
        }Datas;
        // check data
        TLogDataCheckSum CheckSum;
}TLockLog, *PLockLog;
//////////////////////////////////////Log//////////////////////////////////////////////
```

TLockLog LockLog;

d) Example

Command: F5 72 00 00 5F C6

Response:

F5 72 24 75 5F 87 15 00 00 00 00 00 00 19 03 26 09 15 75
01 01 58 14 01 54 70 19 03 26 09 15 93
01 01 58 14 01 54 70 19 03 26 09 16 94
01 01 58 14 01 54 70 19 03 26 09 16 94
01 01 58 14 01 54 70 19 03 26 09 16 94
01 01 58 14 01 54 70 19 03 26 09 16 94
01 01 58 14 01 54 70 19 03 26 09 16 94
01 01 58 14 01 54 70 19 03 26 09 16 94
01 01 58 14 01 54 70 19 03 26 09 16 94
F5 72 10 5B 5F 53 01 01 58 14 01 54 70 19 03 26 09 16 94
01 01 58 14 01 54 70 19 03 26 09 16 94
01 01 58 14 01 54 70 19 03 26 09 17 95
01 01 58 14 01 54 70 19 03 26 09 17 95
01 01 58 14 01 54 70 19 03 26 09 17 95
01 01 58 14 01 54 70 19 03 26 09 17 95
01 01 58 14 01 54 70 19 03 26 09 17 95

e) DATA analysis

1. 24 indicates that the data is in transit, followed by the response packet
2. 15 00 00 00 00 00 00 19 03 26 09 15 75
   01 01 58 14 01 54 70 19 03 26 09 15 93
   01 01 58 14 01 54 70 19 03 26 09 16 94
   01 01 58 14 01 54 70 19 03 26 09 16 94
   01 01 58 14 01 54 70 19 03 26 09 16 94
   01 01 58 14 01 54 70 19 03 26 09 16 94
   01 01 58 14 01 54 70 19 03 26 09 16 94
   01 01 58 14 01 54 70 19 03 26 09 16 94
   01 01 58 14 01 54 70 19 03 26 09 16 94

Indicates nine logs are transmitted (Note: one packet can only be transferred nine logs at most)

3.10 indicates that the data loading and transferring are finished successfully

4. 01 01 58 14 01 54 70 19 03 26 09 16 94
   01 01 58 14 01 54 70 19 03 26 09 16 94
   01 01 58 14 01 54 70 19 03 26 09 16 94
   01 01 58 14 01 54 70 19 03 26 09 17 95
   01 01 58 14 01 54 70 19 03 26 09 17 95
   01 01 58 14 01 54 70 19 03 26 09 17 95
   01 01 58 14 01 54 70 19 03 26 09 17 95
   01 01 58 14 01 54 70 19 03 26 09 17 95

Indicates that the last seven logs are transmitted

4.23 Clear log

a) Protocol

Command: 0x73

b) Command function

Clear all logs in the current system

c) DATA format

```
//////////////////Log information Structure//////////////////////
  typedef struct tagLockLogInfo
{
      u8 bytes[4];
    struct
    {
        u16   LogCount;          //Log amount
        u16   LogPointer;        //Log pointer
    }Datas;

}TLockLogInfo;;
```

d) Example

Mobile APP sends data: F5 73 00 00 5F C7

Mobile APP receives data: F5 73 10 04 5F DB 00 00 00 00

e) DATA analysis

1. 00 00 indicates that the log amount in the current system is zero

2. 00 00 indicates that the latest log pointer points to zero

4.24 Alarm setting & checking

a) Protocol

Command: 0x74

b) Command function

Set and check alarm status, when DATALEN=0x00, represents for checking; when DATALEN=0x01, represents for setting.

c) DATA format

```
u8 AlarmParameter; //Alarm status parameters: 0 means enable alarm   1 means disable alarm
```

d) Example

Checking:

Command: F5 74 00 00 5F C8

Response: F5 74 10 01 5F DA 01

Setting:

Command: F5 74 00 01 5F CA 01

Response: F5 74 10 00 5F D8

e) DATA analysis

01 means the current alarm status is “disable”.

4.25 Buzzer setting & checking

a) Protocol

Command: 0x75

b) Command function

Set and check buzzer status, when DATALEN=0x00, represents for checking; when DATALEN=0x01,represents for setting.

c) DATA format

u8 BeepParameter; //Buzzer status parameters: 0 means enable buzzer 1 means disable buzzer

d) Example

Checking:

Command: F5 75 00 00 5F C9

Response: F5 75 10 01 5F DB 01

Setting:

Command: F5 75 00 01 5F CB 01

Response: F5 75 10 00 5F D9

e) DATA analysis

01 means the current buzzer status is “ disable”.

5. Detailed introduction of Bluetooth command (Bluetooth lock==> mobile APP)

This section is mainly about the details of commands which are sent from Bluetooth lock to mobile phone APP. Each command has its own unique data format.

5.1 Upload status

a) Protocol

Command: 0x80

b) Command function

Automatically upload lock or unlock status, lock tongue status, voltage, lock ID number, unlocking and locking mode.

c) DATA format

```
struct
{
    u8 OpenCloseLockState;   //00 Locked     01 unlocked
    u8 HookState;            //00 Lock hook is not locked   01 Lock hook is locked
    u16 VotageValue;         //Lock battery voltage, unit: mV
    u32          LockId;     //Lock ID No.   0x00000001~0xFFFFFFFF
    u8 OpenLockMode;         //Unlocking mode   00   Unlock manually     01 Unlock automatically
    u8 CloseLockMode;        //Locking mode   00   Lock automatically     01 Lock manually
}LockStatesDatas;
```

d) Example

Bluetooth lock sends data: F5 80 00 0A 5F DE 00 00 15 E8 00 00 00 02 01 00

Bluetooth lock receives data: F5 80 10 00 5F E4

e) DATA analysis

1. 00 indicates that the lock is in locked status (i.e., lock tongue is stretches out);
2. 00 indicates that the lock hook is unlocked (i.e., lock hook is not into lock tongue);
3. 15 E8 indicates the voltage lock, 0x15E8=5608, so the voltage is 5608mV;
4. 00 00 00 02 indicates the lock ID No., 0x00000002=2, so the lock ID No. is 2;
5. 01 indicates the unlocking mode, it is automatic unlocking mode currently;
6. 00 indicates the locking mode, it is the automatic locking mode currently

Notice: After authorized successfully, the system will upload the lock status actively after every locking & unlocking. If the system does not receive a response, it will upload the lock status for three times repeatedly.

KERONG

Technology Change Life

KERONG INDUSTRY (HK) CO., LIMITED

ADD: 2nd Floor, Tonghe Industry Park, Fuping North Rd Pingdi Street, Longgang District,

Shenzhen, Guangdong, China

Tel: 0755-28396063 Fax: +86-755-83117533

Website:www.kerong.hk

http://szkerong.en.alibaba.com/

Email: kr30@kerong.hk