MRX-C2

Objective-C SDK 开发手册

MRX-C2

Asterisk, Inc.
2023/2/1

修改日志

版本	修订内容	修订日期
1.0	初稿	2020/8/25
1.1	添加代理方法 - (void)writeData:(NSData *)data; 添加条码类型 MRXC2_BarcodeTypeITF14, MRXC2_BarcodeTypeITF6, MRXC2_BarcodeTypeAIM128, MRXC2_BarcodeTypeISSN, MRXC2_BarcodeTypeISBN, MRXC2_BarcodeTypeGS1Databar	2023/2/1

目录

1 SDK 使用 5

1.1. 添加 SDK 5
1.2. 导入头文件 8
1.3. SDK 的使用 8
1.3.1. 获取 MRXC2Manager 对象 8
1.3.2. 设置 MRXC2Manager 对象的 delegate 的委托对象 8
1.3.3. 实现 MRXC2Manager 对象代理的代理方法 8
1.3.4. 搜索 MRXC2 设备列表 8
1.3.5. 设置选取的 MRXC2 对象的 delegate 的委托对象 8
1.3.6. 实现 MRXC2 对象 delegate 的代理方法 9
1.3.7. 连接 MRXC2 对象 10
1.3.8. 开始扫描条码 10
1.3.9. 停止扫描条码 10
1.3.10. 断开连接 MRXC2 对象 10

2 MRXC2 类 11

2.1. 属性 11
2.2. 方法 12
2.2.1. initWithCBPeripheral 12
2.2.2. connect 12
2.2.3. disconnect 12
2.2.4. startScan 12
2.2.5. stopScan 13
2.2.6. setBeepOn 13
2.2.7. setVibrationOn 13
2.2.8. getBeepStatus 13
2.2.9. getVibrationStatus 14
2.2.10. setBarcodeTimeOut 14
2.2.11. getBarcodeTimeOut 14

2.2.12. getBarcodeType 14
2.2.13. getSDKVersion 15
2.3. 代理 15
2.3.1 whenMRXC2Ready 15
2.3.2 whenMRXC2InfomationReceived 15
2.3.3 receivedMRXC2Battery 16
2.3.4 receivedBarcodeData 16
2.3.5 startScanStatus 16
2.3.6 stopScanStatus 17
2.3.7 setBeepStatus 17
2.3.8 setVibrationStatus 18
2.3.9 setBarcodeTimeOutStatus 18
2.3.10 receivedBeepIsOn 18
2.3.11 receivedVibrationIsOn 19
2.3.12 receivedBarcodeTimeOutValue 19
2.3.13 receivedData 20
2.3.14 writeData 20
2.4. 枚举 20
2.4.1. MRXC2_BarcodeType 20
2.4.2. MRXC2_BarcodeTimeOut 23

3 MRXC2Manager 类 24

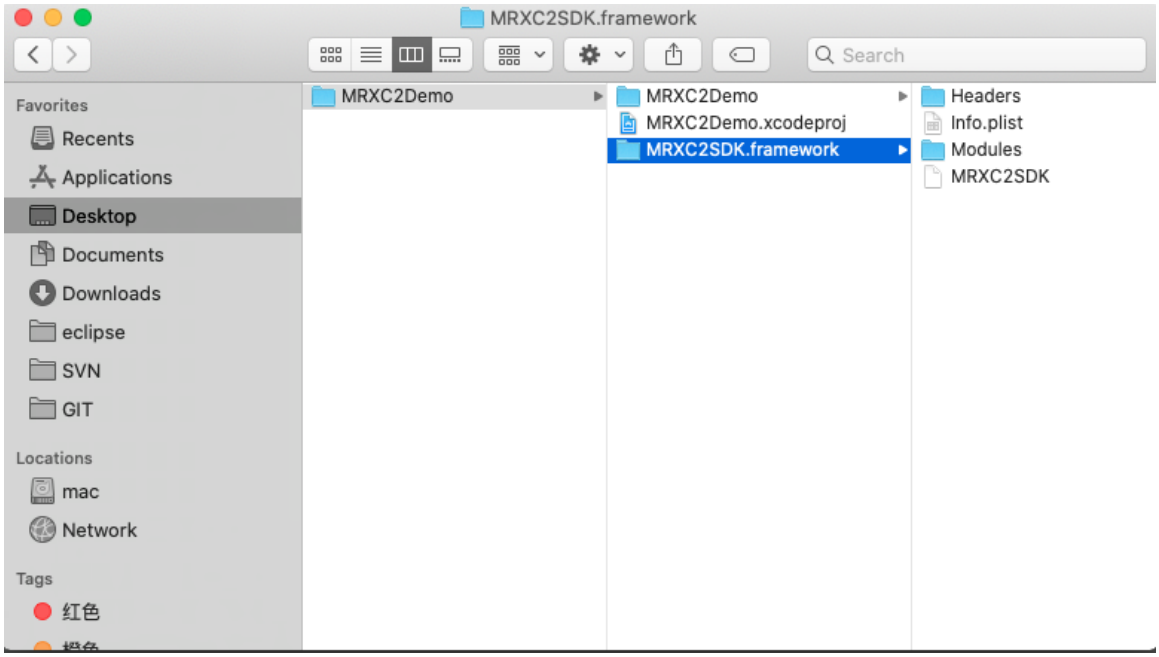
3.1. 属性 24
3.2. 方法 24
3.2.1. sharedMRXC2Manager 24
3.2.2. startSearching 25
3.2.3. stopSearching 25
3.2.4. connectMRXC2 25
3.2.5. disconnectMRXC2 25
3.2.6. sendDataToMRXC2 26
3.2.7. getCurrentConnectedMRXC2 26
3.3. 代理 26
3.3.1. whenReceivedMRXC2List 26
3.4. 枚举 27
3.4.1 MRXC2_DeviceBLEStatus 27

1 SDK 使用

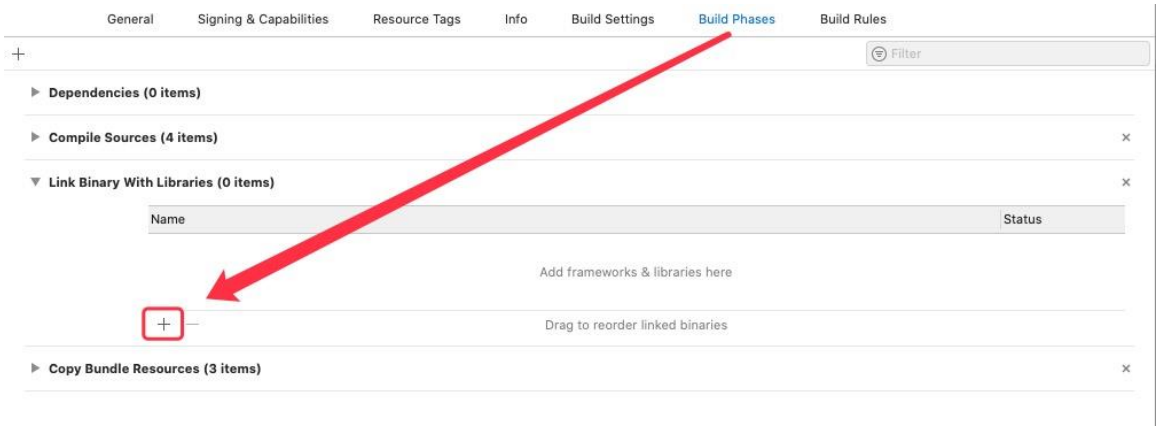
1.1. 添加 SDK

1. 工程导入 MRXC2.framework

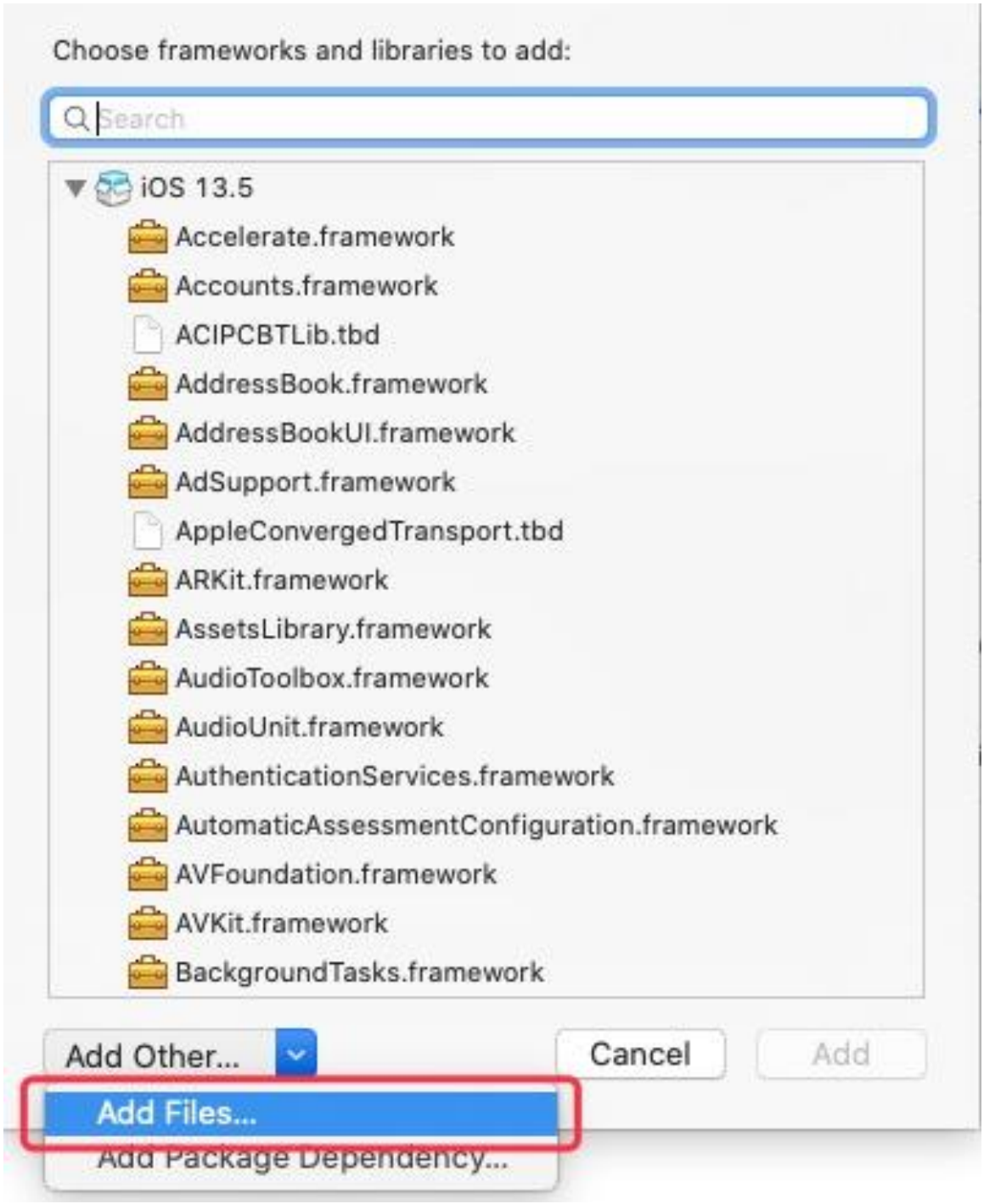
   将 MRXC2.framework 文件拷贝到工程的文件夹中



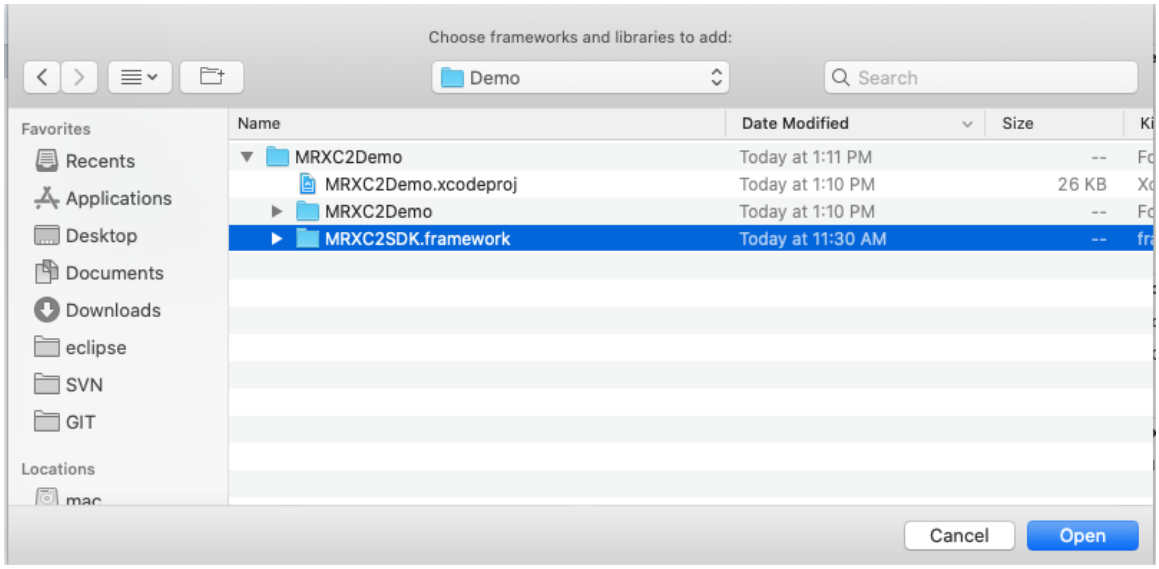
2. TARGET -> Build phases -> Link Binary with Libraries



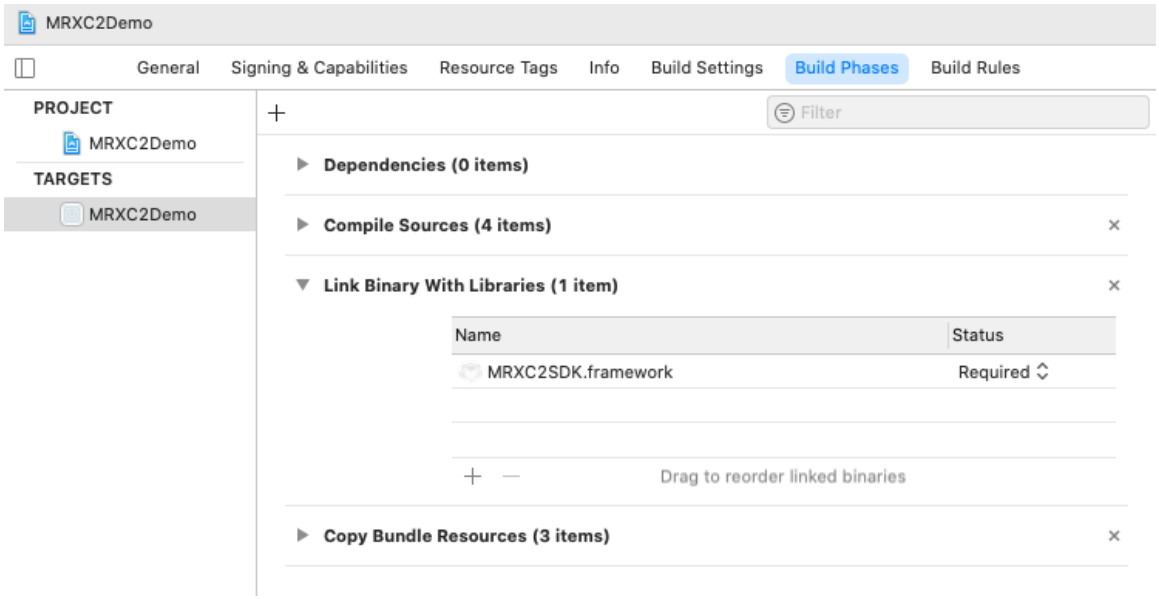
3. 选择 “Add Other…”, “Add Files…



4. 添加 MRXC2SDK.framework



5. 添加完成后如下图:



1.2. 导入头文件

ObjectC 项目需要在使用 SDK 的类中引用头文件，以下作为参考：

```
#import <MRXC2SDK/ MRXC2SDK.h>
```

1.3. SDK 的使用

SDK 主要两部分组成：MRXC2 类和 MRXC2Manager 类。

MRXC2 类对 MRXC2 设备进行的封装，用来获取 MRXC2 设备的信息、连接或断开 MRXC2 设备，并且给 MRXC2 设备发送命令。

MRXC2Manager 类是蓝牙管理对象，用来搜索，获取 MRXC2 对象，收发蓝牙命令。

1.3.1. 获取 MRXC2Manager 对象

```
MRXC2Manager*c2Manager= [MRXC2Manager sharedMRXC2Manager];
```

1.3.2. 设置 MRXC2Manager 对象的 delegate 的委托对象

```
c2Manager.delegate = self;
```

1.3.3. 实现 MRXC2Manager 对象代理的代理方法

```
-(void)whenReceivedMRXC2List:(NSArray *)MRXC2List{
    //搜索 MRXC2 设备列表
    MRXC2List ：搜索到的 MRXC2 设备的列表 (数组元素为 MRXC2 对象)。
}
```

1.3.4. 搜索 MRXC2 设备列表

```
[c2Manager startSearching];
```

搜索到的 MRXC2 设备列表会通过代理方法“whenReceivedMRXC2List:”（请参照 3.3.1）回调。

1.3.5. 设置选取的 MRXC2 对象的 delegate 的委托对象

从搜索到的 MRXC2 设备列表中选取一个设备作为 MRXC2 对象。例如选择列表中的第一个设备。

```
MRXC2 * c2 = [MRXC2List objectAtIndex:0];
c2.delegate = self;
```

1.3.6. 实现 MRXC2 对象 delegate 的代理方法

```
- (void)whenMRXC2Ready:(BOOL)isReady{
    //接收当前 MRXC2 设备连接的状态。
    if(isReady){
        //连接
    }else {
        //断开
    }
}
- (void)receivedBarcodeData:(NSData*)barcodeData barcodeType:
(MRXC2_BarcodeType)barcodeType{
    //接收 MRXC2 读取到的条码数据。
    barcodeData：读取到的条码
    barcodeType：读取到的条码类型
}

- (void)startScanStatus:(BOOL)status{
    If(status){
        //开始扫描成功
    }else{
        //开始扫描失败
    }
}

- (void)stopScanStatus:(BOOL)status{
    If(status){
        //停止扫描成功
    }else{
        //停止扫描失败
    }
}
```

1.3.7. 连接 MRXC2 对象

```
[c2 connect];
```

连接的结果会通过代理方法“whenMRXC2Ready:”（请参照 2.3.1）回调。

1.3.8. 开始扫描条码

在 MRXC2 已经连接的情况下，即：代理方法“whenMRXC2Ready:”（请参照 2.3.1）回调，并且回调参数“isReady”是“YES”的情况可以进行扫描条码。

```
[c2 startScan];
```

当前 MRXC2 设备开始扫描的结果会通过代理方法“startScanStatus:”（请参照 2.3.5）回调。
扫描到的条码的结果会通过代理方法“receivedBarcodeData：barcodeType:” （请参照 2.3.4）回调。

1.3.9. 停止扫描条码

```
[c2 stopScan];
```

当前 MRXC2 设备停止扫描的结果会通过代理方法“stopScanStatus:”（请参照 2.3.6）回调。

1.3.10. 断开连接 MRXC2 对象

```
[c2 disconnect];
```

断开的结果会通过代理方法“whenMRXC2Ready:”（请参照 2.3.1）回调。

2 MRXC2 类

MRXC2 对象是用来连接断开 MRXC2，扫描条码数据，停止扫描条码，对 MRXC2 进行声音震动等相关设定。

2.1. 属性

属性名	属性	类型	描述
name	nonatomic, strong, readonly	NSString	MRXC2 设备名。
peripheral	nonatomic, strong, readonly	CBPeripheral	通过蓝牙与 iOS 连接的外设对象(MRXC2 设备)
isConnected	nonatomic, assign, readonly	BOOL	MRXC2 设备连接状态。
battery	nonatomic, assign, readonly	int	MRXC2 设备电池电量。
manufacturerName	nonatomic, strong, readonly	NSString	MRXC2 设备生产商名称。
hardwareVersion	nonatomic, strong, readonly	NSString	MRXC2 设备硬件版本。
firmwareVersion	nonatomic, strong, readonly	NSString	MRXC2 设备固件版本。
delegate	nonatomic, weak	MRXC2Delegate	MRXC2Delegate (参照 2.3 代理)

2.2. 方法

2.2.1. initWithCBPeripheral

- (instancetype)initWithCBPeripheral:(CBPeripheral *)cBPeripheral;				
	名称	**In/Out**	**类型**	**描述**
参数	cBPeripheral	In	CBPeripheral	iOS 通过蓝牙连接的外围设备对象（MRXC2 设备）。
返回值	-	Out	MRXC2	MRXC2 对象。
方法说明: 创建并初始化 MRXC2 对象。（注：SDK 内部使用） **示例代码:**				

2.2.2. connect

- (void)connect;
方法说明: 连接 MRXC2 设备（通过蓝牙连接）。 此方法执行后，回调“whenMRXC2Ready”（参照 2.3.1 whenMRXC2Ready）。 **示例代码:**（注：c2 为 MRXC2 类的实类对象） [c2 connect];

2.2.3. disconnect

- (void)disconnect;
方法说明: 断开连接 MRXC2 设备。 此方法执行后，回调“whenMRXC2Ready”（参照 2.3.1 whenMRXC2Ready）。 **示例代码:**（注：c2 为 MRXC2 类的实类对象） [c2 disconnect];

2.2.4. startScan

- (void)startScan;
方法说明: 开始扫描条码。 此方法执行后，回调“receivedBarcodeData”（参照 2.3.4）代理。 **示例代码:**（注：c2 为 MRXC2 类的实类对象） [c2 startScan];

2.2.5. stopScan

- (void)stopScan;
方法说明: 停止扫描条码。 **示例代码:**（注：c2 为 MRXC2 类的实类对象） [c2 stopScan];

2.2.6. setBeepOn

- (void)setBeepOn:(BOOL)isOn;				
	名称	**In/Out**	**类型**	**描述**
参数	isOn	In	BOOL	MRXC2 设备蜂鸣状态： YES：开启, NO：关闭。
方法说明: 设置当前 MRXC2 设备蜂鸣状态。 此方法执行后，回调“setBeepStatus”（参照 2.3.7）代理。 **示例代码:**（注：c2 为 MRXC2 类的实类对象） [c2 setBeepOn:YES];				

2.2.7. setVibrationOn

- (void)setVibrationOn:(BOOL)isOn;				
	名称	**In/Out**	**类型**	**描述**
参数	isOn	In	BOOL	MRXC2 设备振动状态： YES：开启, NO：关闭。
方法说明: 设置当前 MRXC2 设备振动状态。 此方法执行后，回调“setVibrationStatus”（参照 2.3.8）代理。 **示例代码:**（注：c2 为 MRXC2 类的实类对象） [c2 setVibrationOn:YES];				

2.2.8. getBeepStatus

- (void)getBeepStatus;
方法说明: 获取当前 MRXC2 设备蜂鸣状态。 此方法执行后，回调“receivedBeepIsOn”（参照 2.3.10）代理。 **示例代码:**（注：c2 为 MRXC2 类的实类对象） [c2 getBeepStatus];

2.2.9. getVibrationStatus

- (void)getVibrationStatus;
方法说明: 获取当前 MRXC2 设备振动状态。 此方法执行后，回调“receivedVibrationIsOn”（参照 2.3.11）代理。 **示例代码:**（注：c2 为 MRXC2 类的实类对象） [c2 getVibrationStatus];

2.2.10. setBarcodeTimeOut

- (void)setBarcodeTimeOut:(MRXC2_BarcodeTimeOut)barcodeTimeOut;				
	名称	**In/Out**	**类型**	**描述**
参数	barcodeTimeOut	In	MRXC2_BarcodeTimeOut	扫码超时时间。 枚举类型（参照 2.4.2）。
方法说明: 设置当前 MRXC2 设备扫码超时时间。 此方法执行后，回调“setBarcodeTimeOutStatus”（参照 2.3.9）代理。 **示例代码:**（注：c2 为 MRXC2 类的实类对象） [c2 setBarcodeTimeOut:BarcodeTimeOut_4S];//设置扫描超时时间为 4s				

2.2.11. getBarcodeTimeOut

- (void)getBarcodeTimeOut;
方法说明: 获取当前 MRXC2 设备扫码超时时间。 此方法执行后，回调“receivedBarcodeTimeOutValue”（参照 2.3.12）代理。 **示例代码:**（注：c2 为 MRXC2 类的实类对象） [c2 getBarcodeTimeOut];

2.2.12. getBarcodeType

+ (NSString *)getBarcodeType:(MRXC2_BarcodeType)barcodetype;				
	名称	**In/Out**	**类型**	**描述**
参数	barcodetype	In	MRXC2_BarcodeType	条码类型枚举（参照 2.4.1）。
方法说明: 返回条码类型枚举的文本说明。 **示例代码:** NSString *string = [MRXC2 getBarcodeType:MRXC2_BarcodeTypeCode39];				

2.2.13. getSDKVersion

+ (NSString *)getSDKVersion;				
	名称	**In/Out**	**类型**	**描述**
返回值	-	Out	NSString	SDK 版本。
方法说明: 获取当前 MRXC2 版本。 **示例代码:** NSString *sdkString = [MRXC2 getSDKVersion]; // sdkString 为 SDK 版本				

2.3. 代理

2.3.1 whenMRXC2Ready

- (void)whenMRXC2Ready:(BOOL)isReady;				
	名称	**In/Out**	**类型**	**描述**
参数	isReady	OUT	BOOL	MRXC2 连接的状态： YES：连接, NO：断开。

方法说明:
接收当前 MRXC2 连接的状态。
调用 connect（参照：2.2.2）或 disconnect（参照：2.2.3）方法后回调该代理。
示例代码:
- (void)whenMRXC2Ready:(BOOL)isReady{
if (isReady) {
// 连接
}else{
//断开
}
}				

2.3.2 whenMRXC2InfomationReceived

- (void)whenMRXC2InfomationReceived;

方法说明:
SDK 连接成功后，获取到设备的软件版本、硬件版本、生产商名称后回调代理。
示例代码:
- (void)whenMRXC2InfomationReceived{
 	
}

2.3.3 receivedMRXC2Battery

函数名	- (void)receivedMRXC2Battery:(int)battery;			
	名称	In/Out	类型	描述
参数	battery	Out	int	MRXC2 设备的电池电量。 (0,1,2,3,4)

方法说明:
接收当前 MRXC2 设备的电池电量。
示例代码:

```
- (void)receivedMRXC2Battery:(int)battery{
    // battery  电池电量
}
```

2.3.4 receivedBarcodeData

- (void)receivedBarcodeData:(NSData *)barcodeData barcodeType:(MRXC2_BarcodeType)barcodeType;				
	名称	In/Out	类型	描述
参数	barcodeData	Out	NSData	MRXC2 读取到的条码数据。
参数	barcodeType	Out	MRXC2_BarcodeType	MRXC2 读取到的条码类型。 枚举类型（参照 2.4.1）。

方法说明:
接收 MRXC2 读取到的条码数据。
调用 startScan（参照：2.2.4）方法后回调该代理。
示例代码:

```
- (void)receivedBarcodeData:(NSData *)barcodeData
barcodeType:(MRXC2_BarcodeType)barcodeType{
    // barcodeData 读取到的条码
    // barcodeType 读取到的条码类型
}
```

2.3.5 startScanStatus

- (void)startScanStatus:(BOOL)status;				
	名称	In/Out	类型	描述
参数	status	Out	BOOL	MRXC2 设备开始扫描的状态： YES：开始扫描成功, NO： 开始扫描失败。

方法说明:
接收当前 MRXC2 设备开始扫描的结果。
调用 startScan（参照：2.2.4）方法后回调该代理。
示例代码:

```
- (void)startScanStatus:(BOOL)status{
    if (status) {
        // 开始扫描成功
    } else {
        // 开始扫描失败
    }
}
```

2.3.6stopScanStatus

- (void)stopScanStatus:(BOOL)status;				
	名称	**In/Out**	**类型**	**描述**
参数	status	Out	BOOL	MRXC2 设备停止扫描的状态： YES：停止扫描成功, NO： 停止扫描失败。

方法说明:
接收当前 MRXC2 设备停止扫描的结果。
调用 stopScan（参照：2.2.5）方法后回调该代理。
示例代码:

```
- (void)stopScanStatus:(BOOL)status{
    if (status) {
        // 停止扫描成功
    } else {
        // 停止扫描失败
    }
}
```

2.3.7setBeepStatus

- (void)setBeepStatus:(BOOL)status;				
	名称	**In/Out**	**类型**	**描述**
参数	status	Out	BOOL	设置 MRXC2 设备蜂鸣的状态： YES：设备蜂鸣成功, NO： 设备蜂鸣失败。

方法说明:
接收设置当前 MRXC2 设备蜂鸣的结果。
调用 setBeepOn（参照：2.2.6）方法后回调该代理。
示例代码:

```
- (void)setBeepStatus:(BOOL)status{
    if (status) {
        // 设备蜂鸣成功
    } else {
        // 设备蜂鸣失败
    }
}
```

2.3.8setVibrationStatus

- (void)setVibrationStatus:(BOOL)status;				
	名称	**In/Out**	**类型**	**描述**
参数	status	Out	BOOL	设置 MRXC2 设备振动的状态： YES：设备振动成功, NO： 设备振动失败。

方法说明:
接收设置当前 MRXC2 设备振动的结果。
调用 setVibrationOn（参照：2.2.7）方法后回调该代理。
示例代码:

```
- (void)setVibrationStatus:(BOOL)status{
    if (status) {
        // 设备振动成功
    } else {
        // 设备振动失败
    }
}
```

2.3.9setBarcodeTimeOutStatus

- (void)setBarcodeTimeOutStatus:(BOOL)status;				
	名称	**In/Out**	**类型**	**描述**
参数	status	Out	BOOL	设置 MRXC2 设备扫码间隔时间的状态： YES：扫码超时成功, NO： 扫码超时失败。

方法说明:
接收设置当前 MRXC2 设备扫码超时时间的结果。
调用 setBarcodeTimeOut（参照：2.2.10）方法后回调该代理。
示例代码:

```
- (void)setBarcodeTimeOutStatus:(BOOL)status{
    if (status) {
        // 扫码超时成功
    } else {
        // 扫码超时失败
    }
}
```

2.3.10 receivedBeepIsOn

- (void)receivedBeepIsOn:(BOOL)isOn;				
	名称	**In/Out**	**类型**	**描述**
参数	status	Out	BOOL	MRXC2 设备蜂鸣的状态：

				YES：设备蜂鸣开启, NO： 设备蜂鸣关闭。

方法说明:
接收当前 MRXC2 设备蜂鸣的状态。
调用 getBeepStatus（参照：2.2.8）方法后回调该代理。
示例代码:

```
- (void)receivedBeepIsOn:(BOOL)isOn {
    if (status) {
        // 设备蜂鸣开启
    } else {
        // 设备蜂鸣关闭
    }
}
```

2.3.11 receivedVibrationIsOn

- (void)receivedVibrationIsOn:(BOOL)isOn;

	名称	In/Out	类型	描述
参数	status	Out	BOOL	MRXC2 设备振动的状态： YES：设备振动开启, NO： 设备振动关闭。

方法说明:
接收当前 MRXC2 设备振动的状态。
调用 getVibrationStatus（参照：2.2.9）方法后回调该代理。
示例代码:

```
- (void)receivedVibrationIsOn:(BOOL)isOn {
    if (status) {
        // 设备振动开启
    } else {
        // 设备振动关闭
    }
}
```

2.3.12 receivedBarcodeTimeOutValue

- (void)receivedBarcodeTimeOutValue:(MRXC2_BarcodeTimeOut)barcodeTimeOut;

	名称	In/Out	类型	描述
参数	status	Out	BOOL	MRXC2 设备扫码超时时间。

方法说明:
接收当前 MRXC2 设备扫码超时时间。
调用 getBarcodeTimeOut（参照：2.2.11）方法后回调该代理。
示例代码:

```
- (void)receivedBarcodeTimeOutValue:(MRXC2_BarcodeTimeOut)barcodeTimeOut {
    // barcodeTimeOut 扫码超时时间
}
```

2.3.13 receivedData

- (void)receivedData:(NSData *)data;				
	名称	In/Out	类型	描述
参数	data	Out	NSData	SDK 接收到的所有数据。

方法说明:
接收 MRXC2 发送给 SDK 的所有数据。
示例代码:

```
- (void)receviedData:(NSData *)data{
    // data    MRXC2 发送给 SDK 的数据
}
```

2.3.14 writeData

- (void)writeData:(NSData *)data;				
	名称	In/Out	类型	描述
参数	data	Out	NSData	SDK 发出的所有数据。

方法说明:
接收 SDK 发送给 MRXC2 的所有数据。
示例代码:

```
- (void)receviedData:(NSData *)data{
    // data    SDK 发送给 MRXC2 的数据
}
```

2.4. 枚举

2.4.1. MRXC2_BarcodeType

定义	描述
MRXC2_BarcodeTypeCode39	Code39
MRXC2_BarcodeTypeCode11	Code11
MRXC2_BarcodeTypeCodabar	Codabar
MRXC2_BarcodeTypeEAN13	EAN-13
MRXC2_BarcodeTypeCode128	Code128

MRXC2_BarcodeTypeEAN13With2Supps	EAN 13 with 2 Supps.
MRXC2_BarcodeTypeDiscrete2Of5	Discrete 2 of 5
MRXC2_BarcodeTypeEAN13With5Supps	EAN-13 with 5 Supps.
MRXC2_BarcodeTypeIATA2Of5	IATA 2 of 5
MRXC2_BarcodeTypeMSI	MSI
MRXC2_BarcodeTypeInterleaved2Of5	Interleaved 2 of 5
MRXC2_BarcodeTypeEAN128	EAN-128
MRXC2_BarcodeTypeCode93	Code93
MRXC2_BarcodeTypeUPCE1	UPC-E1
MRXC2_BarcodeTypeUPCA	UPC-A
MRXC2_BarcodeTypeUPCE1With2Supps	UPC-E1 with 2 Supps.
MRXC2_BarcodeTypeUPCAWith2Supps	UPC-A with 2 Supps.
MRXC2_BarcodeTypeUPCAWith5Supps	UPC-A with 5 Supps.
MRXC2_BarcodeTypeUPCE1With5Supps	UPC-E1 with 5 Supps.
MRXC2_BarcodeTypeTriopticCode39	Trioptic Code39
MRXC2_BarcodeTypeUPCE0	UPC-E
MRXC2_BarcodeTypeBooklandEAN	Bookland EAN
MRXC2_BarcodeTypeUPCE0With2Supps	UPC-E with 2 Supps.
MRXC2_BarcodeTypeCouponCode	Coupon Code
MRXC2_BarcodeTypeUPCE0With5Supps	UPC-E with 5 supps.
MRXC2_BarcodeTypeGS1DataBarLimitedRSSLimited	GS1 DataBar Limited (RSS-Limited)
MRXC2_BarcodeTypeEAN8	EAN-8

MRXC2_BarcodeTypeGS1DataBarRSS14	GS1 DataBar (RSS-14)
MRXC2_BarcodeTypeEAN8With2Supps	EAN-8 with 2 Supps.
MRXC2_BarcodeTypeGS1DataBarExpandedRSSExpanded	GS1 DataBar Expanded (RSS-Expanded)
MRXC2_BarcodeTypeEAN8With5Supps	EAN-8 with 5 Supps.
MRXC2_BarcodeTypeMatrix2Of5	Matrix 2 of 5
MRXC2_BarcodeTypeChinaPostChinese2Of5	China Post (Chinese 2 of 5)
MRXC2_BarcodeTypeCode32	Code32
MRXC2_BarcodeTypeUKPlessey	UK Plessey
MRXC2_BarcodeTypeISBT128	ISBT128
MRXC2_BarcodeTypePDF417	PDF417
MRXC2_BarcodeTypeAztec	Aztec
MRXC2_BarcodeTypeMicroPDF417	MicroPDF417
MRXC2_BarcodeTypeQR	QR
MRXC2_BarcodeTypeDataMatrix	DataMatrix
MRXC2_BarcodeTypeMicroQR	Micro QR
MRXC2_BarcodeTypeHanXinCode	HanXin Code
MRXC2_BarcodeTypeMaxicode	Maxicode
MRXC2_BarcodeTypeITF14	ITF-14
MRXC2_BarcodeTypeITF6	ITF-6
MRXC2_BarcodeTypeAIM128	AIM 128
MRXC2_BarcodeTypeISSN	ISSN
MRXC2_BarcodeTypeISBN	ISBN

MRXC2_BarcodeTypeGS1Databar	GS1-Databar

2.4.2. MRXC2_BarcodeTimeOut

定义	描述
MRXC2_BarcodeTimeOut_4S	扫描超时时间 4s
MRXC2_BarcodeTimeOut_8S	扫描超时时间 8s
MRXC2_BarcodeTimeOut_16S	扫描超时时间 16s
MRXC2_BarcodeTimeOut_24S	扫描超时时间 24s
MRXC2_BarcodeTimeOut_30S	扫描超时时间 30s
MRXC2_BarcodeTimeOut_60S	扫描超时时间 60s
MRXC2_BarcodeTimeOut_90S	扫描超时时间 90s
MRXC2_BarcodeTimeOut_120S	扫描超时时间 120s
MRXC2_BarcodeTimeOut_300S	扫描超时时间 300s

3 MRXC2Manager 类

MRXC2Manager 是蓝牙管理对象，用来搜索，获取 MRXC2 对象，收发蓝牙命令。

3.1. 属性

属性名	属性	类型	描述
deviceBLEStatus	readonly	MRXC2_DeviceBLEStatus	iOS 设备蓝牙状态。 枚举类型（参照 3.4.1）。
isSearching	readonly	BOOL	当前蓝牙的检索状态。 YES：检索中， NO： 未检索。
centeralManager	nonatomic, strong, readonly	CBCentralManager	CBCentralManager 的对象。
delegate	nonatomic, weak	MRXC2ManagerDelegate	MRXC2ManagerDelegate （参照 3.3）代理。

3.2. 方法

3.2.1. sharedMRXC2Manager

+ (MRXC2Manager *)sharedMRXC2Manager;

	名称	In/Out	类型	描述
返回值	-	Out	MRXC2Manager	MRXC2Manager 对象。

方法说明:
创建并初始化 MRXC2Manager 对象（单例模式）。
示例代码:

```
MRXC2Manager *c2Manager = [MRXC2Manager sharedMRXC2Manager];
```

3.2.2. startSearching

- (BOOL)startSearching;				
	名称	**In/Out**	**类型**	**描述**
返回值	-	Out	BOOL	方法执行结果: YES：方法执行成功， NO： 方法执行失败。
方法说明: 开始查找 MRXC2 设备。 此方法执行后，回调“whenReceivedMRXC2List”（参照 3.3.1）代理。 **示例代码:**（注：c2Manager 为 MRXC2Manager 类的实类对象） [c2Manager startSearching];				

3.2.3. stopSearching

- (void)stopSearching;
方法说明: 停止查找 MRXC2 设备。 **示例代码:**（注：c2Manager 为 MRXC2 类的实类对象） [c2Manager stopSearching];

3.2.4. connectMRXC2

- (void)connectMRXC2:(MRXC2 *)MRXC2;				
	名称	**In/Out**	**类型**	**描述**
参数	MRXC2	In	MRXC2	MRXC2 对象。
方法说明: 连接指定的 MRXC2 设备。（注：SDK 内部使用） **示例代码:**				

3.2.5. disconnectMRXC2

- (void)disconnectMRXC2:(MRXC2 *)MRXC2;				
	名称	**In/Out**	**类型**	**描述**
参数	MRXC2	In	MRXC2	MRXC2 对象。
方法说明: 断开连接指定的 MRXC2 设备。（注：SDK 内部使用） **示例代码:**				

3.2.6. sendDataToMRXC2

- (void)sendDataToMRXC2:(MRXC2 *)MRXC2 data:(NSData *)data;				
	名称	**In/Out**	**类型**	**描述**
参数	MRXC2	In	MRXC2	MRXC2 对象。
参数	data	In	NSData	发送的数据。
方法说明: 向指定的 MRXC2 设备发送数据。（注：SDK 内部使用） **示例代码:**				

3.2.7. getCurrentConnectedMRXC2

- (MRXC2 *)getCurrentConnectedMRXC2;				
	名称	**In/Out**	**类型**	**描述**
返回值	-	Out	MRXC2	MRXC2 对象。
方法说明: 获取当前已连接成功的 MRXC2 设备对象。 **示例代码:**（注：c2Manager 为 MRXC2Manager 类的实类对象） MRXC2 *c2 = [c2Manager getCurrentConnectedMRXC2];				

3.3. 代理

3.3.1. whenReceivedMRXC2List

- (void)whenReceivedMRXC2List:(NSArray *)MRXC2List;				
	名称	**In/Out**	**类型**	**描述**
参数	MRXC2List	Out	NSArray	MRXC2 设备的列表 (数组元素为 MRXC2 对象)。

方法说明:
接收一个其元素为 MRXC2 对象的数组（方法会多次回调）。
调用 startSearching（参照 3.2.2）方法后回调该代理。
示例代码:
- (void)whenReceivedMRXC2List:(NSArray *)MRXC2List{
// MRXC2List 元素为 MRXC2 对象的数组。
}				

3.4. 枚举

3.4.1 MRXC2_DeviceBLEStatus

定义	描述
MRXC2_DeviceBLEStatus_PowerOff	iOS 设备蓝牙已关闭。
MRXC2_DeviceBLEStatus_PowerOn	iOS 设备蓝牙已开启。
MRXC2_DeviceBLEStatus_Unsupported	iOS 设备不支持蓝牙。