

**FEITIAN**

# aR530 Developer Guide for iOS



Copyright ©2015 Feitian Technologies Co., Ltd

<http://www.ftsafes.com>

## Revision History:

| Date                         | Revision | Description            |
|------------------------------|----------|------------------------|
| 20 <sup>th</sup> , Jan, 2015 | 1.0      | Release first version  |
| April 16, 2015               | 1.1      | Add get reader UID API |

## Software Developer's Agreement

All Products of Feitian Technologies Co., Ltd. (Feitian) including, but not limited to, evaluation copies, diskettes, CD-ROMs, hardware and documentation, and all future orders, are subject to the terms of this Agreement. If you do not agree with the terms herein, please return the evaluation package to us, postage and insurance prepaid, within seven days of their receipt, and we will reimburse you the cost of the Product, less freight and reasonable handling charges.

1. Allowable Use – You may merge and link the Software with other programs for the sole purpose of protecting those programs in accordance with the usage described in the Developer's Guide. You may make archival copies of the Software.
2. Prohibited Use – The Software or hardware or any other part of the Product may not be copied, reengineered, disassembled, decompiled, revised, enhanced or otherwise modified, except as specifically allowed in item 1. You may not reverse engineer the Software or any part of the product or attempt to discover the Software's source code. You may not use the magnetic or optical media included with the Product for the purposes of transferring or storing data that was not either an original part of the Product, or a Feitian provided enhancement or upgrade to the Product.
3. Warranty – Feitian warrants that the hardware and Software storage media are substantially free from significant defects of workmanship or materials for a time period of twelve (12) months from the date of delivery of the Product to you.
4. Breach of Warranty – In the event of breach of this warranty, Feitian's sole obligation is to replace or repair, at the discretion of Feitian, any Product free of charge. Any replaced Product becomes the property of Feitian.

Warranty claims must be made in writing to Feitian during the warranty period and within fourteen (14) days after the observation of the defect. All warranty claims must be accompanied by evidence of the defect that is deemed satisfactory by Feitian. Any Products that you return to Feitian, or a Feitian authorized distributor, must be sent with freight and insurance prepaid.

EXCEPT AS STATED ABOVE, THERE IS NO OTHER WARRANTY OR REPRESENTATION OF THE PRODUCT, EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

5. Limitation of Feitian's Liability – Feitian's entire liability to you or any other party for any cause whatsoever, whether in contract or in tort, including negligence, shall not exceed the price you paid for the unit of the Product that caused the damages or are the subject of, or indirectly related to the cause of action. In no event shall Feitian be liable for any damages caused by your failure to meet your obligations, nor for any loss of data, profit or savings, or any other consequential and incidental damages, even if Feitian has been advised of the possibility of damages, or for any claim by you based on any third-party claim.

6. Termination – This Agreement shall terminate if you fail to comply with the terms herein. Items 2, 3, 4 and 5 shall survive any termination of this Agreement.

## Contents

|  |          |
|--|----------|
| <b>Chapter 1. Overview.....</b>                          | <b>1</b> |
| <b>Chapter 2. Development Overview .....</b>             | <b>2</b> |
| 2.1 Describe static library .....                        | 2        |
| <b>Chapter 3. Definitions .....</b>                      | <b>5</b> |
| 3.01 Get operation handle of aR530.....                  | 5        |
| 3.02 Setting the current instance of the object.....     | 5        |
| 3.03 Get the reader insert event .....                   | 5        |
| 3.04 Get the reader plug out event .....                 | 5        |
| 3.05 Get the device information.....                     | 6        |
| 3.06 Get the response data from card.....                | 6        |
| 3.07 Ge the lib version .....                            | 7        |
| 3.08 Get the reader device ID.....                       | 7        |
| 3.09 get the firmware version.....                       | 7        |
| 3.10 Connect to contactless card.....                    | 8        |
| 3.11 To disconnect with the current connected card ..... | 8        |
| 3.12 Get the contactless card type .....                 | 8        |
| 3.13 Send command to contactless card .....              | 9        |
| 3.14 To authentication with current block.....           | 9        |
| 3.15 To read block (removed in new SDK) .....            | 10       |
| 3.16 To update block(removed in new SDK) .....           | 10       |
| 3.17 To do initialization with current block .....       | 11       |
| 3.18 Read value.....                                     | 11       |
| 3.19 To write value.....                                 | 12       |
| 3.20 To increment value.....                             | 12       |
| 3.21 To decrement value .....                            | 13       |
| 3.22 get the device UID.....                             | 13       |

# Chapter 1. Overview

This document describes how to develop application based on Feitian aR530, and the document has guide developers to using API to do operate with Feitian aR530.

We through four parts to describe aR530 SDK.

1.1 First part, describe application dynamic library and JAR file, to let developer have a basic concept.

1.2 Second part, we do explain all support APIs, and have to do each API means

1.3 Third part, the details of APIs

1.4 Describe the development process and notice items

## Chapter 2. Development Overview

### 2.1 Describe static library

The aR530 SDK based on static library (.a) and header file (.h), the below chart listed related file:

| File         | Describe                   |
|--------------|----------------------------|
| libFTaR530.a | Support armv7/armv7s/arm64 |
| Header file  | \Include                   |

To develop application, customer need to import above file and also import framework which related with libFTaR530.a

AVFoundation.framework

Foundation.framework

CoreAudio.framework

CoreMedia.framework

ExternalAccessory.framework

AudioToolbox.framework

MediaPlayer.framework

### 2.2 Describe operation APIs

| Number              | API name               | Description   | Status      |
|---------------------|------------------------|---|-------------|
| initialization APIs |                        |   |             |
| 1                   | SharedInstance         | Get the aR530 operating handle                        | Implemented |
| 2                   | setDeviceEventDelegate | To set current reader event, which is instance object | Implemented |
| Reader event APIs   |                        |   |             |

|  |                            |   |             |
|--|----------------------------|---|-------------|
| 3  | FtaR530DidConnected        | Get reader plug in event  | Implemented |
| 4  | FTaR530DidDisConnected     | Get reader plug out event   | Implemented |
| 5  | FTaR530GetInfoDidComplete  | Get reader information event  | Implemented |
| 6  | FTNFCDidComplete           | Get the response data from card   | Implemented |
| Get version                                      |                            |   |             |
| 7  | getDeviceID                | Get reader hardware id  | Implemented |
| 8  | getFirmwareVersion         | Get reader firmware version   | Implemented |
| 9  | getLibVersion              | Get the current lib version   | Implemented |
| Contactless card functions (Type A/B and Felica) |                            |   |             |
| 10   | NFC_Card_Open              | Connect to contactless card   | Implemented |
| 11   | NFC_Card_Close             | To disconnect the current card  | Implemented |
| 12   | NFC_Card_Recognize         | Get the current contactless card type (after NFC_Card_Open can call this API) | Implemented |
| 13   | NFC_Card_Transmit          | Send APDU commands to card  | Implemented |
| Mifare card functions                            |                            |   |             |
| 14   | Mifare_GeneralAuthenticate | To authentication the current card  | Implemented |
| 15   | Mifare_ReadBinary          | Read block data   | Implemented |
| 16   | Mifare_UpdateBinary        | Update block data   | Implemented |
| 17   | Mifare_ClassicBlockInitial | To do initialization with the special block                                   | Implemented |
| 18   | Mifare_ClassicReadValue    | Read value from card  | Implemented |



|    |                          |                        |             |
|----|--------------------------|------------------------|-------------|
| 19 | Mifare_ClassicStoreBlock | Write value into block | Implemented |
| 20 | Mifare_ClassicIncrement  | To increment value     | Implemented |
| 21 | Mifare_ClassicDecrement  | To decrement value     | Implemented |

## Chapter 3. Definitions

### 3.01 Get operation handle of aR530

```
#include "FTaR530.h"  
+(id)sharedInstance;
```

Please refer to aR530 demo code

### 3.02 Setting the current instance of the object

```
#include "FTaR530.h"  
-(void)setDeviceEventDelegate:(id<FTaR530Delegate>)delegate;
```

Please refer to aR530 demo code

### 3.03 Get the reader insert event

```
#include "FTaR530.h"  
-(void)FTaR530DidConnected;
```

Please refer to aR530 demo code

### 3.04 Get the reader plug out event

```
#include "FTaR530.h"  
-(void)FTaR530DidDisConnected;
```

Please refer to aR530 demo code

### 3.05 Get the device information

```
#include "FTaR530.h"
/*@Name:          -(void)FTaR530GetInfoDidComplete:(unsigned char *)retData
retDataLen:(unsigned int)retDataLen      functionNum:(unsigned int)functionNum
errCode:(unsigned int)errCode;
  *@Function:      This function will be callback when Get Firmware Version or Get Device
ID function completed.
  *@Parameter:    OUT:(1).(unsigned char *)retData:   return Data
  *                (2).(unsigned int)retDataLen:      return Data length
  *                (3).(unsigned int)functionNum:      The function number
  *                (4).(unsigned int)errCode:          The error code(0-succes,other
value-error code)
  *
  */
-(void)FTaR530GetInfoDidComplete:(unsigned char *)retData retDataLen:(unsigned
int)retDataLen      functionNum:(unsigned int)functionNum  errCode:(unsigned
int)errCode;
```

Please refer to aR530 demo code

### 3.06 Get the response data from card

```
/*@Name:          -(void)FTNFCDidComplete:(nfc_card_t)cardHandle retData:(unsigned
char *)retData  retDataLen:(unsigned int)retDataLen  functionNum:(unsigned
int)funcNum errCode:(unsigned int)errCode;
  *@Function:      This function will be callback when NFC function completed.
  *@Parameter:    OUT:(1).(nfc_card_t)cardHandle:      the card's handle
  *                (2).(unsigned char *)retData:        return data
  *                (3).(unsigned int)retDataLen:          return data length
  *                (4).(unsigned int)funcNum:             The function number
  *                (5).unsigned int)errCode:             The error code(0-success, other
value-error code)
  *
  */
-(void)FTNFCDidComplete:(nfc_card_t)cardHandle retData:(unsigned char *)retData
retDataLen:(unsigned int)retDataLen      functionNum:(unsigned int)funcNum
errCode:(unsigned int)errCode;
```

Please refer to aR530 demo code

### 3.07 Get the lib version

```
#include "FTaR530.h"
-(NSString *)getLibVersion;
```

Please refer to aR530 demo code

### 3.08 Get the reader device ID

```
#include "FTaR530.h"
/*@Name:      +(void)getDeviceID:(id<FTaR530Delegate>)delegate;
*@Function:   Get device ID
*@Parameter:  IN:(id<FTaR530Delegate>)delegate:
*/
-(void)getDeviceID:(id<FTaR530Delegate>)delegate;
```

Please refer to aR530 demo code

To get the response data, need call below API:

FTNFCDidComplete: retData: retDataLen: functionNum: errCode:  
FT\_FUNCTION\_NUM\_GET\_DEVICEID

### 3.09 get the firmware version

```
#include "FT_aR530.h"
/*@Name:      +(void)getFirmwareVersion:(id<FTaR530Delegate>)delegate;
*@Function:   Get firmware version
*@Parameter:  IN:(id<FTaR530Delegate>)delegate:
*/
-(void)getFirmwareVersion:(id<FTaR530Delegate>)delegate;
```

Please refer to aR530 demo code

To get the response data, need call below API:

FTNFCDidComplete: retData: retDataLen: functionNum: errCode:  
FT\_FUNCTION\_NUM\_GET\_FIRMWAREVERSION

### 3.10 Connect to contactless card

```
#include "FT_aR530.h"
/*@Name:      +(void)NFC_Card_Open:(id<FTaR530Delegate>)delegate
 *@Function:   Open the CardReader and Connect to SmartCard
 *@Parameter:  IN:(id<FTaR530Delegate>)delegate
 */
-(void)NFC_Card_Open:(id<FTaR530Delegate>)delegate;
```

Please refer to aR530 demo code

To get the response data, need call below API:

FTaR530GetInfoDidComplete: retDataLen: functionNum: errCode:  
FT\_FUNCTION\_NUM\_OPEN\_CARD

### 3.11 To disconnect with the current connected card

```
#include "FT_aR530.h"
/*@Name:      +(void)NFC_Card_Close:(nfc_card_t)card
               delegate:(id<FTaR530Delegate>)delegate;
 *@Function:   Close the CardReader and Disconnect with SmartCard
 *@Parameter:  IN:(1).(nfc_card_t)card: SmartCard's handle has been Opened
 *              (2).(id<FTaR530Delegate>)delegate
 */
-(void)NFC_Card_Close:(nfc_card_t)card delegate:(id<FTaR530Delegate>)delegate;
```

Please refer to aR530 demo code

To get the response data, need call below API:

FTaR530GetInfoDidComplete: retDataLen: functionNum: errCode:  
FT\_FUNCTION\_NUM\_CLOSE\_CARD

### 3.12 Get the contactless card type

```
#include "FT_aR530.h"
/*@Name:      +(void)NFC_Card_Recognize:(nfc_card_t)card
               delegate:(id<FTaR530Delegate>)delegate;
 *@Function:   Recognize the smartcard's type
 *@Parameter:  IN: (1).nfc_card_t card: the pointer SmartCard Handle
 *              (2).(id<FTaR530Delegate>)delegate
 */
```

```
-(void)NFC_Card_Recognize:(nfc_card_t)card
delegate:(id<FTaR530Delegate>)delegate;
```

Please refer to aR530 demo code

To get the response data, need call below API:

FTaR530GetInfoDidComplete: retDataLen: functionNum: errCode:  
FT\_FUNCTION\_NUM\_RECOGNIZE

### 3.13 Send command to contactless card

```
/*@Name: +(void)NFC_Card_Transmit:(nfc_card_t)card sendBuf:(unsigned char *)sendBuf
sendLen:(unsigned int)sendLen delegate:(id<FTaR530Delegate>)delegate;
*@Function: Transmit APDU to Smart Card
*@Parameter: IN: (1).nfc_card_t card: The pointer of smart card handle
*              (2).(unsigned char *)sendBuf: Send buffer
*              (3).(unsigned int)sendLen: Send buffer length
*              (4).(id<FTaR530Delegate>)delegate
*/

-(void)NFC_Card_Transmit:(nfc_card_t)card sendBuf:(unsigned char *)sendBuf
sendLen:(unsigned int)sendLen delegate:(id<FTaR530Delegate>)delegate;
```

Please refer to aR530 demo code

To get the response data, need call below API:

FTaR530GetInfoDidComplete: retDataLen: functionNum: errCode:  
FT\_FUNCTION\_NUM\_TRANSMIT

### 3.14 To authentication with current block

```
/*@Name: +(void) Mifare_GeneralAuthenticate:(nfc_card_t)card blockNum:(unsigned
char)blockNum keyType:(unsigned char)keyType key:(unsigned char *)key
delegate:(id<FTaR530Delegate>)delegate;
*@Funciton: Authentication the key of Block
*@Parameter: IN: (1).nfc_card_t card: the pointer of smartcard handle
*              (2).(unsigned char)blockNum: block number
*              (3).(unsigned char)keyType: key tyep(A or B)
*              (4).(unsigned char *)key: Authentication key
*              (5).(id<FTaR530Delegate>)delegate
*/
```

```
-(void) Mifare_GeneralAuthenticate:(nfc_card_t)card blockNum:(unsigned
char)blockNum keyType:(unsigned char)keyType key:(unsigned char *)key
delegate:(id<FTaR530Delegate>)delegate;
```

Please refer to aR530 demo code

To get the response data, need call below API:

```
FTaR530GetInfoDidComplete: retDataLen: functionNum: errCode:
FT_FUNCTION_NUM_AUTHENTICATE
```

### 3.15 To read block (removed in new SDK)

```
/*@Name: +(void)Mifare_ReadBinary:(nfc_card_t)card blockNum:(unsigned
char)blockNum size:(unsigned char)size delegate:(id<FTaR530Delegate>)delegate;
*@Function: Read the binary data of block
*@Parameter: IN: (1).nfc_card_t card: The Pointer of smart card handle
*              (2).(unsigned char)blockNum: The block number
*              (3).size:(unsigned char)size: the data length to read
*              (4).(id<FTaR530Delegate>)delegate
*/
-(void)Mifare_ReadBinary:(nfc_card_t)card blockNum:(unsigned char)blockNum
size:(unsigned char)size delegate:(id<FTaR530Delegate>)delegate;
```

Please refer to aR530 demo code

To get the response data, need call below API:

```
FTaR530GetInfoDidComplete: retDataLen: functionNum: errCode:
FT_FUNCTION_NUM_READ_BINARY
```

### 3.16 To update block(removed in new SDK)

```
/*@Name: +(void)mifare_UpdateBinary:(nfc_card_t)card blockNum:(unsigned
char)blockNum data:(unsigned char *)data size:(unsigned char)size
delegate:(id<FTaR530Delegate>)delegate;
*@Function: Update the binary data of block
*@Parameter: IN: (1).nfc_card_t card: The pointer of smart card handle
*              (2).(unsigned char)blockNum: The block number
*              (3).(unsigned char *)data: data buffer
*              (4).(unsigned char)size: data buffer length
*              (5).(id<FTaR530Delegate>)delegate
*/
```

```

-(void)Mifare_UpdateBinary:(nfc_card_t)card    blockNum:(unsigned char)blockNum
data:(unsigned char *)data    size:(unsigned char)size
delegate:(id<FTaR530Delegate>)delegate;

```

Please refer to aR530 demo code

To get the response data, need call below API:

```

FTaR530GetInfoDidComplete: retDataLen: functionNum: errCode:
FT_FUNCTION_NUM_UPDATE_BINARY

```

### 3.17 To do initialization with current block

```

/*@Name:          +(void)Mifare_ClassicBlockInitial:(nfc_card_t)card
blockNum:(unsigned char)blockNum delegate:(id<FTaR530Delegate>)delegate;
*@Function:       Initialize the value of block
*@Parameter:     IN: (1).nfc_card_t card: The pointer of smart card handle
*                (2).(unsigned char)blockNum: The block number
*                (3).(id<FTaR530Delegate>)delegate
*/
-(void)Mifare_ClassicBlockInitial:(nfc_card_t)card    blockNum:(unsigned
char)blockNum delegate:(id<FTaR530Delegate>)delegate;

```

Please refer to aR530 demo code

To get the response data, need call below API:

```

FTaR530GetInfoDidComplete: retDataLen: functionNum: errCode:
FT_FUNCTION_NUM_INIT_BLOCK

```

### 3.18 Read value

```

/*@Name:          +(void)Mifare_ClassicReadValue:(nfc_card_t)card blockNum:(unsigned
char)blockNum delegate:(id<FTaR530Delegate>)delegate;
*@Function:       Read the value of block
*@Parameter:     IN: (1).nfc_card_t card: The pointer of smart card handle
*                (2).(unsigned char)blockNum: The block number to read
*                (3).(id<FTaR530Delegate>)delegate
*/
-(void)Mifare_ClassicReadValue:(nfc_card_t)card blockNum:(unsigned char)blockNum
delegate:(id<FTaR530Delegate>)delegate;

```

Please refer to aR530 demo code

To get the response data, need call below API:

```

FTaR530GetInfoDidComplete: retDataLen: functionNum: errCode:

```



## FT\_FUNCTION\_NUM\_READ\_VALUE

## 3.19 To write value

```

/*@Name:      +(void)Mifare_ClassicStoreBlock:(nfc_card_t)card
blockNum:(unsigned char)blockNum valueAmount:(unsigned int)valueAmount
delegate:(id<FTaR530Delegate>)delegate;
* @Function:   Store value to the block
* @Parameter:  IN: (1).nfc_card_t card: The pointer of smart card handle
*              (2).(unsigned char)blockNum: The block number
*              (3).(unsigned int)valueAmount: The value to store
*              (4).(id<FTaR530Delegate>)delegate
*/
-(void)Mifare_ClassicStoreBlock:(nfc_card_t)card blockNum:(unsigned char)blockNum
valueAmount:(unsigned int)valueAmount delegate:(id<FTaR530Delegate>)delegate;

```

Please refer to aR530 demo code

To get the response data, need call below API:

FTaR530GetInfoDidComplete: retDataLen: functionNum: errCode:

FT\_FUNCTION\_NUM\_STORE\_BLOCK

## 3.20 To increment value

```

/*@Name:      +(void)Mifare_ClassicIncrement:(nfc_card_t)card blockNum:(unsigned
char)blockNum valueAmount:(unsigned int)valueAmount
delegate:(id<FTaR530Delegate>)delegate;
* @Function:   Increment the value of block
* @Parameter:  IN: (1).nfc_card_t card: The pointer of smart card handle
*              (2).(unsigned char)blockNum: The block number
*              (3).(unsigned int)valueAmount: The value to Increment
*              (4).(id<FTaR530Delegate>)delegate
*/
-(void)Mifare_ClassicIncrement:(nfc_card_t)card blockNum:(unsigned char)blockNum
valueAmount:(unsigned int)valueAmount delegate:(id<FTaR530Delegate>)delegate;

```

Please refer to aR530 demo code

To get the response data, need call below API:

FTaR530GetInfoDidComplete: retDataLen: functionNum: errCode:

FT\_FUNCTION\_NUM\_INCREMENT

### 3.21 To decrement value

```

/*@Name: +(void)Mifare_ClassicDecrement:(nfc_card_t)card    blockNum:(unsigned
char)blockNum valueAmount:(unsigned int)valueAmount
    delegate:(id<FTaR530Delegate>)delegate;
* @Function:    Decrement the value of block
* @Parameter:  IN: (1).nfc_card_t card: The pointer of smart card handle
*               (2).(unsigned char)blockNum: The block number
*               (3).(unsigned int)valueAmount: The value to Decrement
*               (4).(id<FTaR530Delegate>)delegate
*/
-(void)Mifare_ClassicDecrement:(nfc_card_t)card blockNum:(unsigned char)blockNum
valueAmount:(unsigned int)valueAmount delegate:(id<FTaR530Delegate>)delegate;

```

Please refer to aR530 demo code

To get the response data, need call below API:

FTaR530GetInfoDidComplete: retDataLen: functionNum: errCode:  
FT\_FUNCTION\_NUM\_DECREMENT

### 3.22 get the device UID

```

#include "FT_aR530.h"
/*@Name:      +(void)aR530_GetDeviceUID:(id<FTaR530Delegate>)delegate;
* @Function:   Get device UID
* @Parameter: IN:(id<FTaR530Delegate>)delegate:
*/
- (void)getDeviceUID:(id<FTaR530Delegate>)delegate;

```

Please refer to aR530 demo code

To get the response data, need call below API:

FTNFCDidComplete: retData: retDataLen: functionNum: errCode:  
FT\_FUNCTION\_NUM\_GET\_FIRMWAREVERSION