AsReader®

AsReaderGUN

SDK Reference Guide

Applies to ASR-R250G and ASR-L251G

Asterisk, Inc.

Revision History

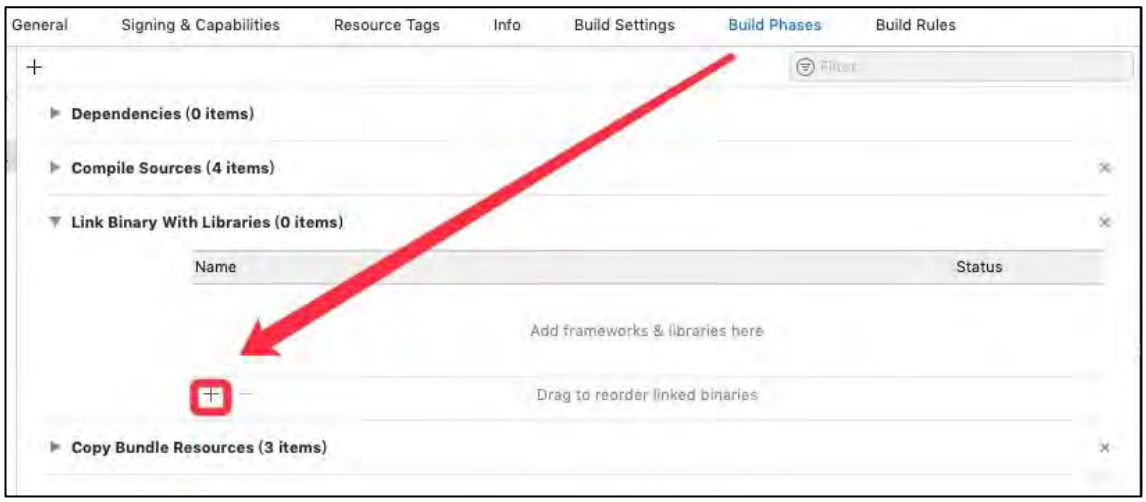
Version	Modified Contents	Date
1.0	Initial version	2020/9/18
1.1	Add an appendix	2020/10/20
1.2	Delete “stopDecode” Add “setBarcodeMode (With Custom prefix suffix)”	2021/04/20
1.3	Add description of “continuousMode”	2022/04/07

Contents

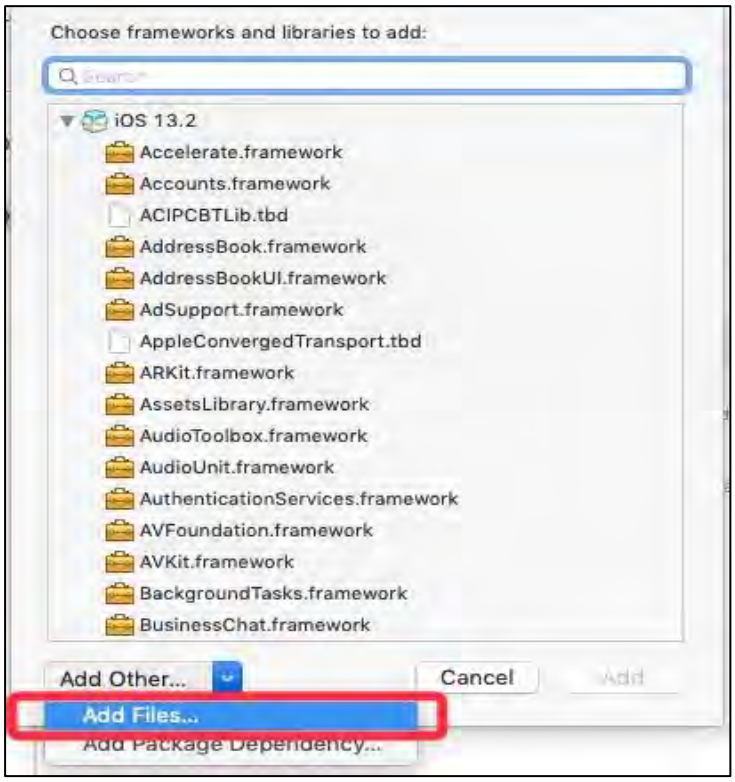
1. Preparation for SDK Usage

1.1. Add SDK

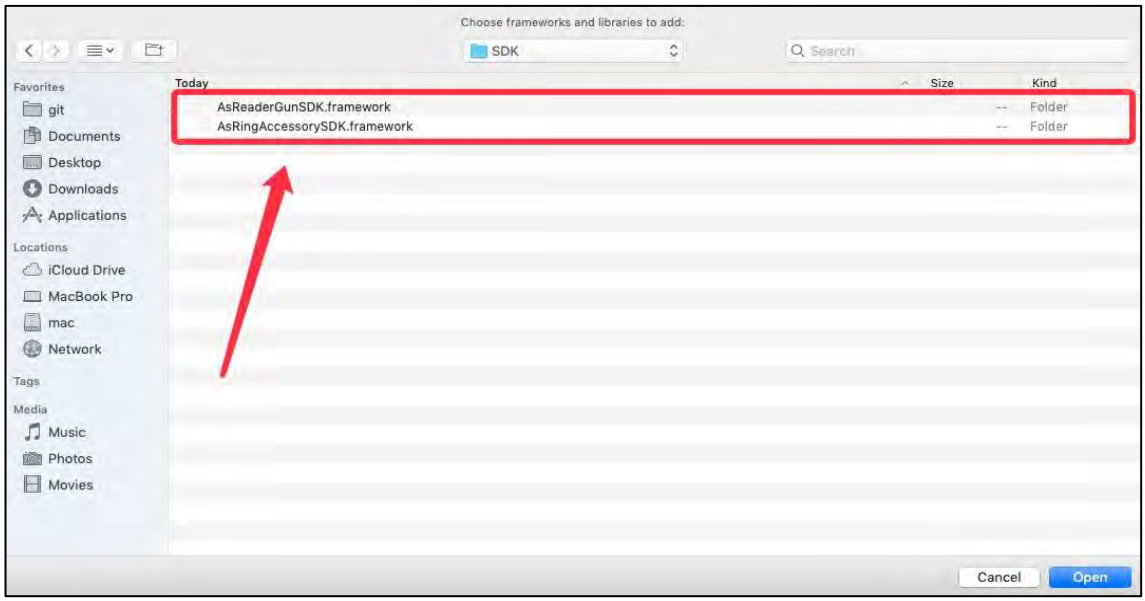
1. TARGET -> Build phases -> Link Binary with Libraries



2. Select “Add Other…”, “Add Files…”



3. Add AsReaderGUNSDK.framework, AsRingAccessorySDK.framework



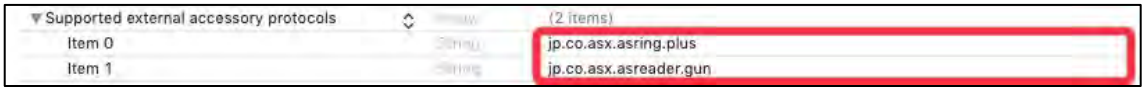
4. Complete as shown

5. Add Protocols

In the info.plist file, add the protocols corresponding to the following devices to the "Supported external accessory protocols".

jp.co.asx.asring.plus

jp.co.asx.asreader.gun



1.2. Import the Header File

The ObjectiveC project needs to import the header file in the class where you want to use the SDK. Please refer to the following code:

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

2. AsReaderGUN Class

2.1. Properties

2.1.1. @property(strong, nonatomic) NSString *deviceModel;

Description: Sets or gets the protocol for the connection of the AsReaderGUN accessories. (For example: com.asreader.gun)

2.1.2. @property (assign,nonatomic,readonly) BOOL isConnect;

Description: Gets the connection status of the AsReaderGUN. YES: connected NO: disconnected

2.2. Function

2.2.1. initWithDeviceModel

Function	- (instancetype)initWithDeviceModel:(NSString *)deviceModel;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
deviceModel	IN	NSString	The connection protocol for the AsReaderGUN accessory. (For example: com.asreader.gun)
-	OUT	AsReaderGUN	AsReaderGUN object

■**Function Description:** This function is used to initialize the AsReaderGUN object. ■**Sample code:** AsReaderGUN * AsReaderGUN = [[AsReaderGUN alloc] initWithDeviceModel:@"com.asreader.gun"];

2.2.2. deviceModel

Function	- (NSString *)deviceModel;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	NSString	The connection protocol for the AsReaderGUN accessory.
■**Function Description:** Gets the connection protocol for the AsReaderGUN accessory. ■**Sample code:** NSString * deviceModel = [AsReaderGUN deviceModel];			

2.2.3. address

Function	- (NSString *)address;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	NSString	The address AsRing+ assigned to AsReaderGUN.
■**Function Description:** Get the address AsRing+ assigned to AsReaderGUN. ■**Sample code:** NSString * address = [AsReaderGUN address];			

2.2.4. getAsReaderGUNVersion

Function	- (NSString *)getAsReaderGUNVersion;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	NSString	SDK version

■**Function Description:**

Get the version of the SDK

■**Sample code:**

```
NSString * sdkVersion= [AsReaderGUN getAsReaderGUNVersion];
```

3. AsReader Class

3.1. Properties

For the storage destination and the default value of each item, please refer to Appendix 11.1.

3.1.1. @property (nonatomic, strong) AsReaderGUN *mAsReaderGUN;

Description:
Gets or sets the interface of AsReaderGUN.

3.1.2. @property (nonatomic, assign) BuzzerState buzzer;

Description:
Gets or sets the buzzer state of AsReader.
Enumerated type BuzzerState (see 5.1.3)

3.1.3. @property (nonatomic, assign) VibratorState vibrator;

Description:
Gets or sets the vibrator state of AsReader.
Enumerated type VibratorState (see 5.1.4)

3.1.4. @property (nonatomic, assign) int operationTime;

Description:
Gets or sets the operation time of the RFID module of AsReaderGUN. (unit: ms)

3.1.5. @property (nonatomic, assign) int inventoryTime;

Description:
Gets or sets the inventory time of the RFID module of AsReaderGUN. (unit: ms)

3.1.6. @property (nonatomic, assign) int idleTime;

Description:
Gets or sets the idle time of the RFID module of AsReaderGUN. (unit: ms)

3.1.7. @property (nonatomic, assign) int sleepTime;

Description:
Gets or sets the automatic sleep time of AsReaderGUN. (unit: s)
Note: ASR-R250G is timed when the connection is disconnected, and ASR-L251G is timed when the connection is successful and no operation has been performed.

3.1.8. @property (nonatomic, assign) int autoOffTime;

Description:
Gets or sets the automatic shut-off time after AsReaderGUN disconnects. (unit: s)

3.1.9. @property (nonatomic, strong) NSString *accessPassword;

Description:
Gets or sets the Access password for the RFID tag.

3.1.10. @property (nonatomic, assign) SessionType inventorySession;

Description:
Gets or sets the Session value of the RFID module of AsReaderGUN.
Enumerated type SessionType (see 5.1.5)

3.1.11. @property (nonatomic, assign) SessionFlag sessionFlag;

Description:
Gets or sets the Session Flag of AsReaderGUN's RFID module.
Enumerated type SessionFlag (see 5.1.6)

3.1.12. @property (nonatomic, strong) NSString *serialNumber;

Description:
Gets or sets the serial number of the AsReaderGUN device.

3.1.13. @property (nonatomic, assign) BOOL continuousMode;

Description:
Gets or sets whether the RFID module of AsReaderGUN reads continuously. To use the Continuous mode, ContinuousMode=YES must be set in app. For firmware reasons, the continuous scan mode is off by default each time the power is turned on or the AsReader and App are successfully connected. YES: Continuous scan is valid NO: Continuous scan is invalid

3.1.14. @property (nonatomic, assign) int powerGain;

Description:
Gets or sets the Power value for RFID tag inventory. Struct CMinMaxValue (see 5.2.1)

3.1.15. @property (nonatomic, assign) BOOL isUseKeyAction;

Description:
Gets or sets whether the AsReaderGUN uses the hardware key. YES: uses the hardware key. NO: doesn’t use the hardware key.

3.1.16. @property (nonatomic, assign) SelectFlag useSelectionMask;

Description:
Gets or sets the Select Flag of AsReaderGUN in the RFID module. Enumerated type SelectFlag (see 5.1.7)

3.1.17. @property (nonatomic, assign) BOOL rssiMode;

Description:
Gets or sets whether to get RSSI data when inventorying RFID tags.
YES: get the RSSI data
NO: does not get RSSI data

3.1.18. @property (nonatomic, assign) BOOL epcMaskMatchMode;

Description:
Gets or sets whether to use the EPC Mask function
YES: use the EPC mask function
NO: does not use the EPC mask function

3.1.19. @property (nonatomic, assign) AlgorithmType algorithm;

Description:
Gets or sets the Q value to be fixed or dynamic when inventorying RFID tags.
Enumerated type AlgorithmType (see 3.3.1)

3.1.20. @property (nonatomic, assign) int minQ;

Description:
Gets or sets the min Q value when inventorying RFID tags.
Range: 0~15

3.1.21. @property (nonatomic, assign) int maxQ;

Description:
Gets or sets the max Q value when inventorying RFID tags.
Range: 0~15

3.1.22. @property (nonatomic, assign) int qValue;

Description:
Gets or sets the current Q value.
Range: 0~15

3.1.23. @property (nonatomic, assign) int linkProfileValue;

Description: Gets or sets the current Link Profile value for the RFID module of AsReaderGUN. Range: 0~3

3.1.24. @property (nonatomic, assign) int defaultLinkProfileValue;

Description: Gets or sets the default Link Profile value for the RFID module of AsReaderGUN. Range: 0~3 Note: Only support ASR-R250G.

3.1.25. @property (nonatomic, assign) int maskTypeValue;

Description: Gets or sets the Mask type for the RFID module of AsReaderGUN. 0: NoMask, 1: SelectionMask, 2: EPCMask

3.2. Function

3.2.1. initWithAsReaderGUN

Function	-(id)initWithAsReaderGUN:(AsReaderGUN*)device delegate:(id<AsReaderDelegate>)delegate;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
device	IN	AsReaderGUN	AsReaderGUN object.
delegate	IN	AsReaderDelegate	Set a delegate to handle events such as AsReaderGUN's buttons being pressed or released. This delegate can receive information or notifications from the AsReaderGUN device.

-	OUT	AsReader	AsReaderobject.

■**Function Description:**
The function used to initialize the AsReader object
Note: This function is called once receiving "AsReaderGUNConnected" notification.
■**Sample code:**
(Note: AsReaderGUN is the object of the AsReaderGUN class)
- (void)AsReaderGUNConnected:(NSNotification *)notification {
dispatch_async(dispatch_get_main_queue(), ^{
AsReader * asReader = [[AsReader alloc]
initWithAsReaderGUN:AsReaderGUN delegate:self];
});
}			

3.2.2. disconnect

Function	- (void)disconnect;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	-	-	-
■**Function Description:** Disconnect from the AsReaderGUN. ■**Sample code:** (Note: asReader is the object of the AsReader class.) [asReader disconnect];			

3.2.3. getAction

Function	- (CommandType)getAction;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	CommandType	Command type Enumerated type CommandType (see 6.1.1)
■**Function Description:** Gets the status of the current action of the AsReader instance. ■**Sample code:** (Note: asReader is the object of AsReader class.) CommandType type = [asReader getAction];			

3.2.4. setDelegate

Function	- (void)setDelegate:(id<AsReaderDelegate>)delegate;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
delegate	IN	AsReaderDelegate	AsReaderDelegate
■**Function Description:** Set delegate for AsReader. ■**Sample code:** (Note: asReader is the object of AsReader class.) [asReader setDelegate:self];			

3.2.5. setScanMode

Function	- (void)setScanMode:(ScanMode)scanMode;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
scanMode	IN	ScanMode	Scan mode. Enumerated type ScanMode (see 6.1.2)
■**Function Description:** Set the scan mode of AsReaderGUN. ■**Sample code:** (Note: asReader is the object of AsReader class.) [asReader setScanMode: RFIDScanMode];			

3.2.6. getScanMode

Function	- (ScanMode)getScanMode;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	ScanMode	Scan mode. Enumerated type ScanMode (see 6.1.2)
■**Function Description:** Gets the current scan mode of AsReaderGUN. ■**Sample code:** (Note: asReader is the object of AsReader class.) ScanMode mode = [asReader getScanMode];			

3.2.7. inventory

Function	- (ResultCode)inventory;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	ResultCode	The result that will be returned when the function is called. Enumerated type ResultCode (see 5.1.1)

■**Function Description:**

Start to inventory tags.

It is called back in the delegate functions changedActionState (see 4.4) and readTag (see 4.3).

■**Sample code:** (Note: asReader is the object of AsReader class.)

```
ResultCode resultCode = [asreader inventory];
if (resultCode == ResultNoError) {
    //Called function succcessfully.
}else{
    //Called function failed.
}
```

3.2.8. readMemory

Function	-(ResultCode)readMemory:(MemoryBank)bank offset:(int)offset length:(int)length;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
bank	IN	MemoryBank	The memory banks of a tag. Enumerated type MemoryBank (see 5.1.2)
offset	IN	int	The offset of target banks. (Unit: word)
length	IN	int	Length of data to be read. (Unit: word)
-	OUT	ResultCode	The result that will be returned when the function is called. Enumerated type ResultCode (see 5.1.1)

■**Function Description:**
This function is used to inventory the data of the specific memory bank of the RFID tag.
It is called back in the delegate function accessResult (see 4.7).
■**Sample code:** (Note: asReader is the object of AsReader class.)

```
ResultCode resultCode = [asreader readMemory:Bank_EPC offset:16 length:4];
if (resultCode == ResultNoError) {
    //Called function successfully.
}else{
    //Called function failed.
}
```

3.2.9. writeMemory

Function	-(ResultCode)writeMemory:(MemoryBank)bank offset:(int)offset value:(NSString *)value;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
bank	IN	MemoryBank	The memory banks of a tag. Enumerated type MemoryBank (see 5.1.2)
offset	IN	int	The offset of target banks. (Unit: word)
value	IN	NSString	Data to write. (the data is a hexadecimal value string)
-	OUT	ResultCode	The result that will be returned when the function is called. Enumerated type ResultCode (see 5.1.1)

■**Function Description:**
This function is used to write data to the specific memory bank of the RFID tag.
It is called back in the delegate function accessResult (see 4.7).
■**Sample code:** (Note: asReader is the object of AsReader class.)

```
ResultCode resultCode = [asreader writeMemory:Bank_EPC offset:16
value:@"1234"];
if (resultCode == ResultNoError) {
    //Called function successfully.
}else{
    //Called function failed.
}
```

3.2.10. lock

Function	- (ResultCode)lock:(LockParam *)param;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
param	IN	LockParam	LockParamobject.
-	OUT	ResultCode	The result that will be returned when the function is called. Enumerated type ResultCode (see 5.1.1)

■**Function Description:**

This function is used to lock a specific memory bank of RFID tags.

It is called back in the delegate function accessResult (see 4.7).

■**Sample code:** (Note: asReader is the object of AsReader class; param is the object of LockParam class.)

```
ResultCode resultCode = [asreader lock:param];
if (resultCode == ResultNoError) {
    //Called function successfully.
}else{
    //Called function failed.
}
```

3.2.11. unlock

Function	- (ResultCode) unlock:(LockParam *)param;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
param	IN	LockParam	LockParamobject.
-	OUT	ResultCode	The result that will be returned when the function is called. Enumerated type ResultCode (see 5.1.1)

■**Function Description:**

Function is to unlock the tag memory bank which was locked. After unlocking, the default password can be used to rewrite the RFID tag data.

It is called back in the delegate function accessResult (see 4.7).

■**Sample code:** (Note: asReader is the object of AsReader class; param is the object of LockParam class.)

```
ResultCode resultCode = [asreader unlock:param];
```

```
if (resultCode == ResultNoError) {
    //Called function successfully.
}else{
    //Called function failed.
}
```

3.2.12. permaLock

Function	- (ResultCode)permaLock:(LockParam *)param;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
param	IN	LockParam	LockParamobject.
-	OUT	ResultCode	The result that will be returned when the function is called. Enumerated type ResultCode (see 5.1.1)

■**Function Description:**
This function is used to permanently lock the data of the specific memory bank of the tag. Permanently locked tag memory bank data can not be changed and unlocked. It is called back in the delegate function accessResult (see 4.7).
■**Sample code:** (Note: asReader is the object of AsReader class; param is the object of LockParam class.)

```
ResultCode resultCode = [asreader permaLock:param];
if (resultCode == ResultNoError) {
    //Called function successfully.
}else{
    //Called function failed.
}
```

3.2.13. kill

Function	- (ResultCode)kill:(NSString *)killPassword;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
killPassword	IN	NSString	The password for killing the tags.
-	OUT	ResultCode	The result that will be returned when the function is called.

			Enumerated type ResultCode (see 5.1.1)

■**Function Description:**

This function is used to kill RFID tags. Killed tags can no longer be used.

It is called back in the delegate function accessResult (see 4.7).

■**Sample code:** (Note: asReader is the object of AsReader class.)

```
ResultCode resultCode = [asreader kill: @"00000000"];
if (resultCode == ResultNoError) {
    //Called function successfully.
}else{
    //Called function failed.
}
```

3.2.14. stop

Function	- (ResultCode)stop;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	ResultCode	The result that will be returned when the function is called. Enumerated type ResultCode (see 5.1.1)

■**Function Description:**

Stop taking inventory of tags.

It is called back in the delegate function changedActionState (see 4.4).

■**Sample code:** (Note: asReader is the object of AsReader class.)

```
ResultCode resultCode = [asreader stop];
if (resultCode == ResultNoError) {
    //Called function successfully.
}else{
    //Called function failed.
}
```

3.2.15. stopSync

Function	- (ResultCode)stopSync;		
Parameters	**IN/OUT**	**Types**	**Descriptions**

-	OUT	ResultCode	The result that will be returned when the function is called. Enumerated type ResultCode (see 5.1.1)

■**Function Description:**

Stops scanning in the synchronized thread.

It is called back in the delegate function changedActionState (see 4.4).

■**Sample code:** (Note: asReader is the object of AsReader class.)

```
ResultCode resultCode = [asreader stopSync];
if (resultCode == ResultNoError) {
    //Called function successfully.
}else{
    //Called function failed.
}
```

3.2.16. defaultParameter

Function	- (ResultCode)defaultParameter;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	ResultCode	The result that will be returned when the function is called. Enumerated type ResultCode (see 5.1.1)

■**Function Description:**

Restores all parameters to their default values.

■**Sample code:** (Note: asReader is the object of AsReader class.)

```
ResultCode resultCode = [asreader defaultParameter];
if (resultCode == ResultNoError) {
    //Called function successfully.
}else{
    //Called function failed.
}
```

3.2.17. saveParameter

Function	- (ResultCode)saveParameter;		
Parameters	**IN/OUT**	**Types**	**Descriptions**

-	OUT	ResultCode	The result that will be returned when the function is called. Enumerated type ResultCode (see 5.1.1)

■**Function Description:**
Save all parameter values to the device memory.
■**Sample code:** (Note: asReader is the object of AsReader class.)
ResultCode resultCode = [asreader saveParameter];
if (resultCode == ResultNoError) {
//Called function successfully.
}else{
//Called function failed.
}			

3.2.18. regionName

Function	- (NSString *)regionName;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	NSString	region name of RF module.
■Function Description: Gets the current region information for AsReaderGUN. ■Sample code: (Note: asReader is the object of AsReader class.) NSSting * regionName = [asreader regionName];			

3.2.19. serialNumber

Function	- (NSString *)serialNumber;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	NSString	Serial number
■Function Description: Gets the current serial number for AsReaderGUN. ■**Sample code:** (Note: asReader is the object of AsReader class.) NSSting * serialNumber = [asreader serialNumber];			

3.2.20. rFModuleVersion

Function	- (NSString *)rFModuleVersion;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	NSString	The version number of the RF module.
■Function Description: Get the RF module version of AsReaderGUN. ■**Sample code:** (Note: asReader is the object of AsReader class.) NSSting * rFModuleVersion = [asreader rFModuleVersion];			

3.2.21. firmwareVersion

Function	- (NSString *)firmwareVersion;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	NSString	FW version
■Function Description: Get the FW version of AsReaderGUN. ■**Sample code:** (Note: asReader is the object of AsReader class.) NSSting * firmwareVersion = [asreader firmwareVersion];			

3.2.22. powerGainRange

Function	- (CMinMaxValue)powerGainRange;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	CMinMaxValue	Range that Power can be set. Struct CMinMaxValue (see 5.2.1)

■Function Description:
Gets the values of the maximum and minimum power that the AsReaderGUN can be set.
■**Sample code:** (Note: asReader is the object of AsReader class.)
int min = asReader.powerGainRange.min; // The minimum value of power can be set
int max = asReader.powerGainRange.max; // The maximum power can be set			

3.2.23. batteryStatus

Function	- (int)batteryStatus;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	int	the amount of battery left. Range: 0, 1, 2, 3, 4
■Function Description: Get the remaining battery power of AsReaderGUN. **■Sample code:** int battery = [asReader batteryStatus];			

3.2.24. clearEpcMask

Function	- (ResultCode)clearEpcMask;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	ResultCode	The result that will be returned when the function is called Enumerated type ResultCode (see 5.1.1)

■Function Description:
Remove EPC mask data from AsReaderGUN memory.
■Sample code: (Note: asReader is the object of AsReader class.)
ResultCode resultCode = [asreader clearEpcMask];
if (resultCode == ResultNoError) {
  //Called function successfully.
}else{
  //Called function failed.
}			

3.2.25. saveEpcMask

Function	- (ResultCode)saveEpcMask;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	ResultCode	The result that will be returned when the function is called

			Enumerated type ResultCode (see 5.1.1)

■**Function Description:**

Save the EPC mask data into AsReaderGUN memory.

■**Sample code:** (Note: asReader is the object of AsReader class.)

```
ResultCode resultCode = [asreader saveEpcMask];
if (resultCode == ResultNoError) {
    //Called function successfully.
}else{
    //Called function failed.
}
```

3.2.26. epcMaskCount

Function	- (int)epcMaskCount;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	int	Number of EPC masks.

■**Function Description:**

Gets the number of EPC masks stored in memory of AsReaderGUN.

■**Sample code:** (Note: asReader is the object of AsReader class.)

```
int epcMaskCount = [asReader epcMaskCount];
```

3.2.27. addEpcMask

Function	-(ResultCode)addEpcMask:(int)offset length:(int)length mask:(NSString *)mask;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
offset	IN	int	The offset of the mask to be add. (Unit: bit)
length	IN	int	The length of the mask to be add. (Unit: bit)
mask	IN	NSString	The mask data. (a NSString in HEX form)
-	OUT	ResultCode	The result that will be returned when the function is called.

			Enumerated type ResultCode (see 5.1.1)

■**Function Description:**

Adds EPC masks.

■**Sample code:** (Note: asReader is the object of AsReader class.)

```
ResultCode resultCode = [asreader addEpcMask:32 length:16 mask:@"1234"];
if (resultCode == ResultNoError) {
    //Called function successfully.
}else{
    //Called function failed.
}
```

3.2.28. addEpcMask

Function	- (ResultCode)addEpcMask:(AsSelectMaskEPCParam *)mask;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
mask	IN	AsSelectMaskEPCParam	AsSelectMaskEPCParamobject.
-	OUT	ResultCode	The result that will be returned when the function is called Enumerated type ResultCode (see 5.1.1)

■**Function Description:**

Adds EPC masks in the form of AsSelectMaskEPCParamobject.

■**Sample code:** (Note: asReader is the object of AsReader class. mask is the object of class AsSelectMaskEPCParam.)

```
ResultCode resultCode = [asreader addEpcMask:mask];
if (resultCode == ResultNoError) {
    //Called function successfully.
}else{
    //Called function failed.
}
```

3.2.29. getEpcMask

Function	- (AsSelectMaskEPCParam *)getEpcMask:(int)index;

Parameters	IN/OUT	Types	Descriptions
index	IN	int	Indexs of the EPC masks
-	OUT	AsSelectMaskEPCParam	AsSelectMaskEPCParamobject
■**Function Description:** Gets the EPC mask for the specified index. ■**Sample code:** (Note: asReader is the object of AsReader class.) AsSelectMaskEPCParam * mask= [asreader getEpcMask:0];			

3.2.30. getLBTMask

Function	- (NSArray *)getLBTMask;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	NSArray	Mask array
■**Function Description:** Gets a mask array of frequency table. ■**Sample code:** (Note: asReader is the object of AsReader class.) NSArray * lbtArray = [asreader getLBTMask];			

3.2.31. getLBT

Function	- (NSArray *)getLBT;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	NSArray	LBT information
■**Function Description:** Gets the LBT information. ■**Sample code:** (Note: asReader is the object of AsReader class.) NSArray * lbtArray = [asreader getLBT];			

3.2.32. setLBT

Function	- (void)setLBT:(NSArray *)table;

Parameters	IN/OUT	Types	Descriptions
table	IN	NSArray	LBT information (Whosearray element is LbtItem object, see 5.7)
■**Function Description:** Sets LBT. ■**Sample code:** (Note: asReader is the object of AsReader class. Array is the object of the class NSArray (whose array element is LbtItem object).) [asreader setLBT:array];			

3.2.33. getLBTFrequency

Function	- (NSString *)getLBTFrequency:(int)slot;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
slot	IN	int	The frequency position of the LBT frequency table
-	OUT	NSString	Band information
■**Function Description:** Gets the frequency band information. ■**Sample code:** (Note: asReader is the object of AsReader class.) NSString * frequency =[asreader getLBTFrequency:0];			

3.2.34. startBuzzerWithBuzzerTime

Function	- (ResultCode)startBuzzerWithBuzzerTime:(int)buzzerTime;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
buzzerTime	IN	int	Buzzer time (unit: ms)
-	OUT	ResultCode	The result that will be returned when the function is called Enumerated type ResultCode (see 5.1.1)
■**Function Description:** Makes the buzzer to beep for a certain time. ■**Sample code:**(Note: asReader is the object of the AsReader class.) ResultCode resultCode = [asreader startBuzzerWithBuzzerTime:100];			

```
if (resultCode == ResultNoError) {
//Called function successfully.
}else{
    //Called function failed.
}
```

3.2.35. startVibratorWithVibratorTime

Function	- (ResultCode)startVibratorWithVibratorTime:(int)vibratorTime;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
vibratorTime	IN	int	Vibrating time (unit: ms)
-	OUT	ResultCode	The result that will be returned when the function is called. Enumerated type ResultCode (see 5.1.1)

■**Function Description:**

Makes the vibrator to vibrate for a certain time.

■**Sample code:**(Note: asReader is the object of the AsReader class.)

```
ResultCode resultCode = [asreader startVibratorWithVibratorTime:100];
if (resultCode == ResultNoError) {
//Called function successfully.
}else{
    //Called function failed.
}
```

3.2.36. startDecode

Function	- (ResultCode)startDecode;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	ResultCode	The result that will be returned when the function is called Enumerated type ResultCode (see 5.1.1)

■**Function Description:**
Starts to scan barcodes.
It is called back in the delegate functiosns changedActionState (see 4.4) and the detectBarcode (see 4.5, 4.6).
Note:
To stop scanning barcode, please apply **3.2.15. stopSync**
■**Sample code:**(Note: asReader is the object of the AsReader class.)

```
ResultCode resultCode = [asreader startDecode];
if (resultCode == ResultNoError) {
//Called function successfully.
}else{
    //Called function failed.
}
```

3.2.37. setBarcodeParam

Function	- (ResultCode)setBarcodeParam:(AsParamValue *)paramData;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
paramData	IN	AsParamValue	AsParamValueobject
-	OUT	ResultCode	The result that will be returned when the function is called Enumerated type ResultCode (see 5.1.1)

■**Function Description:**
Sets barcode configuration information.
■**Sample code:** (Note: asReader is the object of class AsReader. asParamValue is the object of class AsParamValue.)

```
ResultCode resultCode = [asreader setBarcodeParam: asParamValue];
if (resultCode == ResultNoError) {
    //Called function successfully.
}else{
    //Called function failed.
}
```

3.2.38. getBarcodeParam

Function	- (AsParamValue *)getBarcodeParam:(NSNumber *)paramData;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
paramData	IN	NSNumber	Barcode parameters to be obtained See 8.2.1 ParamName.
-	OUT	AsParamValue	AsParamValueobject.

■**Function Description:**
Gets barcode configuration information.
■**Sample code:**(Note: asReader is the object of the AsReader class.)

```
AsParamValue * asParamValue= [asreader getBarcodeParam: [NSNumber numberWithInt:UPCA]];
```

3.2.39. usedSelectionMask

Function	- (BOOL)usedSelectionMask:(int)index;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
index	IN	int	The index of mask
-	OUT	BOOL	Indicates whether the mask corresponding to the current index is being used. YES: Being used. NO: Not being used.

■**Function Description:**
Returns whether the mask pointed to by a specified index in AsReaderGUN is being used.
■**Sample code:**(Note: asReader is the object of the AsReader class.)

```
BOOL isUsed = [asreader usedSelectionMask:0];
if (isUsed) {
//The mask corresponding to the current index is being used.
}else{
// The mask corresponding to the current index is not being used.
}
```

3.2.40. getSelectionMask

Function	- (AsSelectMaskParam *)getSelectionMask:(int)index;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
index	IN	int	The index of mask
-	OUT	AsSelectMaskParam	AsSelectMaskParam object
■Function Description: Gets the Selection mask for a specified index. **■Sample code:**(Note: asReader is the object of the AsReader class.) AsSelectMaskParam * asSelectMaskParam = [asreader getSelectionMask:0];			

3.2.41. setSelectionMask

Function	- (void)setSelectionMask:(int)index withParam:(AsSelectMaskParam *)param;		
Parameters	IN/OUT	Types	Descriptions
index	IN	int	The index of mask
param	IN	AsSelectMaskParam	AsSelectMaskParam object
■Function Description: Sets the Selection mask where the specified index points. **■Sample code:** (Note: asReader is the object of the AsReader class. asSelectMaskParam is the object of the class AsSelectMaskParam.) [asreader setSelectionMask:0 withParam: asSelectMaskParam];			

3.2.42. removeSelectionMask

Function	- (void)removeSelectionMask:(int)index;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
index	IN	int	The index of mask
■Function Description: Removes the Selection mask at the location to which the specified index points to. **■Sample code:**(Note: asReader is the object of the AsReader class.) [asreader removeSelectionMask:0];			

3.2.43. clearSelectionMask

Function	- (void)clearSelectionMask;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	-	-	-
■**Function Description:** Remove all Selection masks. ■**Sample code:**(Note: asReader is the object of the AsReader class.) [asreader clearSelectionMask];			

3.2.44. getAlgorithm

Function	- (AlgorithmType)getAlgorithm;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	AlgorithmType	Type of Q Enumerated type AlgorithmType (see 3.3.1)

■**Function Description:**
Gets the anti-collision mode.
■**Sample code:**(Note: asReader is the object of the AsReader class.)
AlgorithmType type= [asreader getAlgorithm];
if (type = FixedQ) {
// Currently in Fixed Q mode
}else if (type = DynamicQ) {
// Currently in Dynamic Q mode
}			

3.2.45. setBarcodeMode

Function	-(ResultCode)setBarcodeMode:(BOOL)enabled isKeyAction:(BOOL)isKeyOn;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
enabled	IN	BOOL	Enable/disable barcode mode
isKeyOn	IN	BOOL	Not use

-	OUT	ResultCode	The execution result of the function

■**Function Description:**

Set the current device mode to Barcode mode.

Note:

In case you wish to customize prefixes/suffixes, please apply 3.2.46. setBarcodeMode (With Custom prefix suffix) instead of this function.

In the case that Barcode mode is set to be valid, any RFID-related commands cannot be executed.

If Barcode mode is set to be invalid, the device can only execute RFID-related commands.

■**Sample code:** (Note: asraeder is an instance of AsReader class)

```
ResultCode resultCode = [asraeder setBarcodeMode:YES isKeyAction:YES];
if (resultCode == ResultNoError) {
    //The function was called successfully.
}else{
    //The function is failed called.
}
```

3.2.46. setBarcodeMode (With Custom prefix suffix)

Function	**-(ResultCode)setBarcodeMode:(BOOL)enabled isKeyAction:(BOOL)isKeyOn isCustomPreSuffixOn:(BOOL)isCustomPreSuffixOn;**		
Parameters	**IN/OUT**	**Types**	**Descriptions**
enabled	IN	BOOL	Enable/disable barcode mode
isKeyOn	IN	BOOL	Not use
isCustomPreSuffixOn	in	BOOL	Enable/disable prefix/suffix
-	OUT	ResultCode	The execution result of the function

■**Function Description:**

Set the current device mode to Barcode mode and set prefix , suffix.

Note:

In case you wish to customize prefixes/suffixes, please apply this function instead of 3.2.45.

In the case that Barcode mode is set to be valid, any RFID-related commands cannot be executed.

If Barcode mode is set to be invalid, the device can only execute RFID-related commands.

■**Sample code:** (Note: asraeder is an instance of AsReader class)

```
ResultCode resultCode = [asraeder setBarcodeMode:YES isKeyAction:YES
isCustomPreSuffixOn:YES];
if (resultCode == ResultNoError) {
    //The function was called successfully.
}else{
    //The function is failed called.
}
```

3.2.47. isBarcodeModule

Function	- (BOOL)isBarcodeModule;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	BOOL	Gets whether Barcode module is supported or not. YES: support NO: not supported

■**Function Description:**

Gets whether the current AsReaderGUN supports the Barcode module.

■**Sample code:**(Note: asReader is the object of the AsReader class.)

```
BOOL isSupportBarcodeModule = [asreader isBarcodeModule];
if (isSupportBarcodeModule) {
// Supports the Barcode module.
}else{
// Does not support the Barcode module.
}
```

3.2.48. isRFIDModule

Function	- (BOOL)isRFIDModule;		
Parameters	**IN/OUT**	**Types**	**Descriptions**

-	OUT	BOOL	Whether RFID module is supported or not. YES: support NO: not supported

■**Function Description:**

Gets whether the current AsReaderGUN supports the RFID module.

■**Sample code:**(Note: asReader is the object of the AsReader class.)

```
BOOL isSupportRFIDModule = [asreader isRFIDModule];
if (isSupportRFIDModule) {
// Supports the RFID module.
}else{
// Does not support the RFID module.
}
```

3.3. Enum

3.3.1. AlgorithmType

Definitions	Descriptions
FixedQ = 0	Fixed Q
DynamicQ = 1	Dynamic Q

4. AsReaderDelegate Class

4.1. readerInitialized

Function	- (void)readerInitialized:(AsReader *)reader;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
reader	IN	AsReader	Object of AsReader

■**Function Description:**

Once connected to the AsReaderGun successfully, this function is called to obtain the initialized AsReader object.

■**Sample code:**

```
- (void)readerInitialized:(AsReader *)reader {

}
```

4.2. updateDeviceState

Function	- (void)updateDeviceState:(ResultCode)error;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
error	IN	ResultCode	The updated result of AsReader. Enumerated type ResultCode (see 5.1.1)

■**Function Description:**

This function is called back if an error occurs while executing some functions/properties (※). It returns the result of AsReader's execution of the command. (For example: Error occurs when obtaining whether EAN13 barcode type is supported.)

(※): For the functions/properties, please refer to the following sections:

3.1.2 ~ 3.1.25, 3.2.1, 3.2.16, 3.2.18 ~ 3.2.21, 3.2.23 ~ 3.2.33, 3.2.37 ~ 3.2.46

■**Sample code:**

```
- (void)updateDeviceState:(ResultCode)error {
      NSString * errorMessage = [AsResultCode msg:error];
}
```

4.3. readTag

Function	- (void)readTag:(NSString *)tag rssi:(float)rssi phase:(float)phase frequency:(float)frequency;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
tag	IN	NSString	PCEPC number of the tag be inventoried. (Hex NSString)
rssi	IN	float	RSSI value of the tag
phase	IN	float	Phase value of the tag
frequency	IN	float	Frequency value of the tag
■Function Description: Once inventory(see3.2.7) is called or the Trigger key is pressed, this delegate function will be called back.			

4.4. changedActionState

Function	- (void)changedActionState:(CommandType)action resultCode:(NSInteger)resultCode;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
action	IN	CommandType	Action type Enumerated type CommandType (see 6.1.1)
resultCode	IN	NSInteger	The result that will be returned when the function is called. Enumerated type ResultCode (see 5.1.1)

■**Function Description:**
Once a function like inventory (see 3.2.7) or stop (see 3.2.14) is called, this function is called back to receive the execution result.
■**Sample code:**

```
- (void)changedActionState:(CommandType)action
resultCode:(NSInteger)resultCode {
// action: Type of command
// resultCode: The result of command execution
}
```

4.5. detectBarcode

Function	- (void)detectBarcode:(BarcodeType)barcodeType codeId:(NSString *)codeId barcode:(NSString *)barcode;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
barcodeType	IN	BarcodeType	Barcode type. Enumerated type BarcodeType (see 7.2.1)
codeId	IN	NSString	The code ID of the barcode.
barcode	IN	NSString	barcode data

■**Function Description:**
Once the function startDecode (see 3.2.26) is called, this function is called back.
■**Sample code:**

```
- (void)detectBarcode:(BarcodeType)barcodeType codeId:(NSString *)codeId
barcode:(NSString *)barcode{

// barcodeType: Barcode type
// codeId: The code ID of the barcode
// barcode: Barcode data
}
```

4.6. detectBarcode

Function	- (void)detectBarcode:(BarcodeType)barcodeType codeId:(NSString *)codeId barcodeData:(NSData *)barcodeData;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
barcodeType	IN	BarcodeType	Barcode type. Enumerated type BarcodeType (see 7.2.1)
codeId	IN	NSString	The code ID of the barcode.
barcodeData	IN	NSData	barcode data
■Function Description: Once the function startDecode (see 3.2.26) is called, this function is called back.			

4.7. accessResult

Function	- (void)accessResult:(ResultCode)error actionState:(CommandType)action epc:(NSString *)epc data:(NSString *)data rssi:(float)rssi phase:(float)phase frequency:(float)frequency;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
error	IN	ResultCode	The executionresult of the function Enumerated type ResultCode (see 5.1.1)
action	IN	CommandType	AsReaderGUN's operation command. Enumerated type CommandType (see 6.1.1)
epc	IN	NSString	Tag's EPC value (Hex)
data	IN	NSString	The inventoried tag data (Hex)
rssi	IN	float	RSSI value of the tag.
phase	IN	float	Phase value of the tag.
frequency	IN	float	Frequency value of the tag.

■**Function Description:** This function is called when writeMemory, readMemory, lock, unlock, permaLock, and kill are executed. * writeMemory, readMemory, lock, unlock, permaLock, and kill, please refer to sections (3.2.9) (3.2.8) (3.2.10) (3.2.11) (3.2.12) (3.2.13).

4.8. onAsReaderLeftModeKeyEvent

Function	- (BOOL)onAsReaderLeftModeKeyEvent:(BOOL)status;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
status	IN	BOOL	The state of the Mode key to the left of AsReaderGUN YES: The trigger button is pressed NO: The trigger button is released
-	OUT	BOOL	YES: switch mode between Barcode and RFID. NO: keep the current mode.

■**Function Description:**
Once the left Mode key of AsReaderGUN is pressed or released, the function is called back.
■**Sample code:**
- (BOOL)onAsReaderLeftModeKeyEvent:(BOOL)status{
return YES;
}			

4.9. onAsReaderRightModeKeyEvent

Function	- (BOOL)onAsReaderRightModeKeyEvent:(BOOL)status;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
status	IN	BOOL	The state of the Mode key to the right of AsReaderGUN

			YES: The trigger button is pressed NO: The trigger button is released
-	OUT	BOOL	YES: switch mode between Barcode and RFID. NO: keep the current mode.

■**Function Description:**
Once the right Mode key of AsReaderGUN is pressed or released, the function is called back.
■**Sample code:**
- (BOOL) onAsReaderRightModeKeyEvent:(BOOL)status{
return YES;
}			

4.10. onAsReaderTriggerKeyEvent

Function	- (BOOL)onAsReaderTriggerKeyEvent:(BOOL)status;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
status	IN	BOOL	The Trigger button state of AsReaderGUN. YES: The trigger button is pressed NO: The trigger button is released
-	OUT	BOOL	YES: perform the SDK default operation, press The trigger button to start the scan and release to stop the scan NO: does not perform SDK defaults

■**Function Description:**
Once the Trigger button of AsReaderGUN is pressed or released, the function is called back.
■**Sample code:**
- (BOOL) onAsReaderTriggerKeyEvent:(BOOL)status{
return YES;
}			

5. AsRfidValues Class

5.1. Enum

5.1.1. ResultCode

Definitions	Descriptions
ResultNoError = 0x0000	Succeed in result.
ResultOtherError = 0x0001	An error has occurred due to unknown reason.
ResultUndefined = 0x0002	An Undefined Error
ResultMemoryOverrun = 0x0003	Accessing to memory out of range.
ResultMemoryLocked = 0x0004	Memory is locked.
ResultInsufficientPower = 0x000B	Battery power is low.
ResultNonSpecificError = 0x000F	Not a specific error.
ResultInOperation = 0xE000	In operation.
ResultOutOfRange = 0xE001	Out of range.
ResultNotConnected = 0xE100	Not connected to Device.
ResultInvalidParameter = 0xE200	Invalid parameter transmitted.
ResultInvalidResponse = 0xE300	Returned invalid parameter.
ResultNotSupportFirmware = 0xEE00	Unsupported firmware.
ResultTimeout = 0xEFFF	Exceeded allowed accessing time.
ResultHandleMismatch = 0xF001	Handle mismatch.
ResultCRCError = 0xF002	CRC error on tag response.
ResultNoTagReply = 0xF003	No Tag Reply.
ResultInvalidPassword = 0xF004	Invalid password.
ResultZeroKillPassword = 0xF005	Zero Kill password.
ResultTagLost = 0xF006	Tag lost.
ResultCommandFormatError = 0xF007	Command format error.
ResultReadCountInvalid = 0xF008	Read count invalid.
ResultOutOfRetries = 0xF009	Out of retries.
ResultParamError = 0xFFFB	Parameter error.

ResultBusy = 0xFFFC	Busy.
ResultInvalidCommand = 0xFFFD	Invalid Command.
ResultLowBattery = 0xFFFE	Low Battery.
ResultOperationFailed = 0xFFFF	Operation failed.
ResultOverHeated = 0xFFF6	Over Heated.
ResultModuleOverHeated = 0x0307	Module Over Heated.

5.1.2. MemoryBank

Definitions	Descriptions
Bank_Reserved	Refers to Reseved memory Bank.
Bank_EPC	Refers to EPC memory Bank.
Bank_TID	Refers to TID memory Bank.
Bank_User	Refers to User memory Bank.

5.1.3. BuzzerState

Definitions	Descriptions
Buzzer_Off	Turn off the buzzer
Buzzer_Low	Buzzer low
Buzzer_High	Buzzer high

5.1.4. VibratorState

Definitions	Descriptions
Vibrator_Off	Turn off the vibrator
Vibrator_On	Turn on vibrator

5.1.5. SessionType

Definitions	Descriptions
Session_S0	inventoried S0
Session_S1	inventoried S1
Session_S2	inventoried S2

Session_S3	inventoried S3

5.1.6. SessionFlag

Definitions	Descriptions
SessionFlag_A	A only
SessionFlag_B	B only
SessionFlag_AB	A or B

5.1.7. SelectFlag

Definitions	Descriptions
SelectFlag_NotUsed	Not use flag
SelectFlag_SL	SL is valid.
SelectFlag_NOT_SL	SL is invalid.
SelectFlag_All	Both are valid.

5.1.8. MaskTargetType

Definitions	Descriptions
MaskTarget_S0	inventoried S0
MaskTarget_S1	inventoried S1
MaskTarget_S2	inventoried S2
MaskTarget_S3	inventoried S3
MaskTarget_SL	Selection Flags

5.1.9. MaskActionType

Definitions	Descriptions
MaskAction_AB	Tag Matching: assert SL or inventoried → A Tag Not-Matching: retract SL or inventoried → B
MaskAction_AN	Tag Matching: assert SL or inventoried → A Tag Not-Matching: do nothing
MaskAction_NB	Tag Matching: do nothing Tag Not-Matching: retract SL or inventoried → B
MaskAction_MN	Tag Matching: negate SL or (A → B, B → A)

	Tag Not-Matching: do nothing
MaskAction_BA	Tag Matching: retract SL or inventoried → B Tag Not-Matching: assert SL or inventoried → A
MaskAction_BN	Tag Matching: retract SL or inventoried → B Tag Not-Matching: do nothing
MaskAction_NA	Tag Matching: do nothing Tag Not-Matching: assert SL or inventoried → A
MaskAction_NM	Tag Matching: do nothing Tag Not-Matching: negate SL or (A → B, B → A)

5.1.10. MaskType

Definitions	**Descriptions**
MaskType_NO_MASK	No MASK.
MaskType_Selection	Selection MASK.
MaskType_EPC	EPC MASK.

5.2. Struct

5.2.1. CMinMaxValue

The settable range of the RFID output power value:

Definitions	Types	Descriptions
min	int	Minimum settable power
max	int	Maximum settable power

5.3. LockParam

5.3.1. Properties

5.3.1.1. @property (nonatomic) BOOL killPassword;

Description:
Gets or sets whether to control the Kill password area.
YES: controls the Kill password area
NO: do not control the Kill password area

5.3.1.2. @property (nonatomic) BOOL accessPassword;

Description:
Gets or sets whether to control the Access password area.
YES: controls the Access password area
NO: do not control the Access password area

5.3.1.3. @property (nonatomic) BOOL epc;

Description:
Gets or sets whether to control the EPC bank.
YES: controls the EPC bank
NO: do not control the EPC bank

5.3.1.4. @property (nonatomic) BOOL tid;

Description:
Gets or sets whether to control the TID bank.
YES: controls the TID bank
NO: do not control the TID bank

5.3.1.5. @property (nonatomic) BOOL user;

Description:

Gets or sets whether to control the User bank. YES: controls the User bank NO: do not control the User bank

5.4. AsResultCode

5.4.1. Function

5.4.1.1. msg

Function	+(NSString *)msg:(ResultCode)code;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
code	IN	ResultCode	The result of the function being called Enumerated type ResultCode (see 5.1.1)
-	OUT	NSString	Descriptions of the ResultCode
■Function Description: Convert the ResultCode to its corresponding text description. **■Sample code:** NSString * errorMessage = [AsResultCode msg:ResultNoError];			

5.5. AsSelectMaskParam

5.5.1. Properties

5.5.1.1. @property (nonatomic) MaskTargetType target;

Description: Gets or sets the MaskTargetType for the current Mask. Enumerated type MaskTargetType (see 5.1.8)

5.5.1.2. @property (nonatomic) MaskActionType action;

Description:
Gets or sets the MaskActionType for the current Mask. Enumerated type MaskActionType (see 5.1.9)

5.5.1.3. @property (nonatomic) MemoryBank bank;

Description:
Gets or sets the MemoryBank for the current Mask. Enumerated type MemoryBank (see 5.1.2)

5.5.1.4. @property (nonatomic) int offset;

Description:
Gets or sets the offset for the current Mask.

5.5.1.5. @property (strong, nonatomic) NSString *mask;

Description:
Gets or sets the current Mask. (Hex)

5.5.1.6. @property (nonatomic) int length;

Description:
Gets or sets the length of the current Mask.

5.5.1.7. @property (nonatomic) BOOL used;

Description:
Gets or sets whether the current Mask is used. YES: Use Mask NO: Not use Mask

5.5.2. Function

5.5.2.1. index

Function	- (int)index;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
-	OUT	int	Mask order
■Function Description: Gets the current Mask order of the AsSelectMaskParam object. **■Sample code:** int index = [param index];			

5.5.2.2. initWithIndex

Function	- (id)initWithIndex:(int)index;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
index	IN	int	Mask order
-	OUT	AsSelectMaskParam	AsSelectMaskParam object
■Function Description: Creates an AsSelectMaskParam object due to the index value and Selection Mask value. **■Sample code:** AsSelectMaskParam *param = [[AsSelectMaskParam alloc]initWithIndex:0];			

5.5.2.3. initWithParameterIndex

Function	-(id)initWithParameterIndex:(int)index target:(MaskTargetType)maskTarget action:(MaskActionType)maskAction bank:(MemoryBank)maskBank offset:(int)maskOffset mask:(NSString *)maskData used:(BOOL)usedMask;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
index	IN	int	Mask order (0~7)

maskTarget	IN	MaskTargetType	Target value for Selection Masks Enumerated type MaskTargetType (see 5.1.8)
maskAction	IN	MaskActionType	Action value for Selection Masks Enumerated type MaskActionType (see 5.1.9)
maskBank	IN	MemoryBank	MemoryBank value for Selection Masks Enumerated type MemoryBank (see 5.1.2)
maskOffset	IN	int	Offset
maskData	IN	NSString	Mask data (Hex)
usedMask	IN	BOOL	Whether to use the current Mask
■Function Description: It creates AsSelectMaskParam object which makes up Selection Mask information with given parameter values. **■Sample code:** AsSelectMaskParam *param = [[AsSelectMaskParam alloc] initWithParameterIndex:0 target:MaskTarget_SL action:MaskAction_AB bank:Bank_EPC offset:32 mask:@"1234"used:YES];			

5.5.2.4. initWithParameterLength

Function	-(id)initWithParameterLength:(int)index target:(MaskTargetType)maskTarget action:(MaskActionType)maskAction bank:(MemoryBank)maskBank offset:(int)maskOffset mask:(NSString *)maskData length:(int)maskLength used:(BOOL)usedMask;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
index	IN	int	Mask order (0~7)
maskTarget	IN	MaskTargetType	Target value for the Selection Mask Enumerated type MaskTargetType (see 5.1.8)
maskAction	IN	MaskActionType	Action value for Selection Masks

			Enumerated type MaskActionType (see 5.1.9)
maskBank	IN	MemoryBank	MemoryBank value for Selection Masks Enumerated type MemoryBank (see 5.1.2)
maskOffset	IN	int	Offset
maskData	IN	NSString	Mask data (Hex)
maskLength	IN	int	Mask length (Unit: bit)
usedMask	IN	BOOL	Whether to use the current Mask

■**Function Description:**
it creates AsSelectMaskParam objects that make up Selection Mask information with given parameter values.
■**Sample code:**

```
AsSelectMaskParam *param = [[AsSelectMaskParam alloc]
initWithParameterIndex:0 target:MaskTarget_SL action:MaskAction_AB
bank:Bank_EPC offset:32 mask:@"1234" length:16 used:YES];
```

5.6. AsSelectMaskEPCParam

5.6.1. Properties

5.6.1.1. @property (nonatomic) int offset;

Description:
Gets or sets the offset of the Selection mask data.

5.6.1.2. @property (nonatomic) int length;

Description:
Gets or sets the length of the target tag data.

5.6.1.3. @property (strong, nonatomic) NSString *mask;

Description: Gets or sets the Selection Mask data (Hex).

5.7. LbtItem

5.7.1. Properties

5.7.1.1. @property (nonatomic) int mSlot;

Description: Gets or sets the frequency position of the LBT frequency table.

5.7.1.2. @property (nonatomic) BOOL mIsUsed;

Description: Gets or sets whether to use the specific frequency of the LBT frequency table. YES: used NO: not used

5.7.1.3. @property (strong, nonatomic) NSString *frequency;

Description: Gets or sets the frequency value of the LBT frequency table.

5.7.2. Function

5.7.2.1. init

Function	-(id)init;

Parameters	IN/OUT	Types	Descriptions
-	OUT	LbtItem	LbtItem object
■**Function Description:** Initialize the LbtItem object. ■**Sample code:** LbtItem *lbtl = [[LbtItem alloc] init];			

5.7.2.2. initWithSlot

Function	-(id)initWithSlot:(int)slot isUsed:(BOOL)isUsed;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
slot	IN	int	The frequency position of the LBT frequency table
isUsed	IN	BOOL	Whether to use the specified frequency of the LBT frequency table. YES: used NO: not used
-	OUT	Id	LbtItem object
■**Function Description:** Creates and Initializes LbtItem. ■**Sample code:** LbtItem *lbtl = [LbtItem alloc] initWithSlot:1 isUsed:YES];			

6. AsPacket Class

6.1. Enum

6.1.1. CommandType

Definitions	Descriptions
CommandInventory = 0x66	Inventory in progress.
CommandReadMemory = 0x72	Read Memory in progress.
CommandWriteMemory = 0x77	Write Memory in progress
CommandKill = 0x6B	Kill Tag in progress.
CommandLock = 0x6C	Lock in progress.
CommandUnlock = 0x6D	Unlock in progress.
CommandPermaLock = 0x70	Perma Lock in progress.
CommandBlockWrite = 0x57	Block Write in progress.
CommandBlockErase = 0x45	Block Erase in progress.
CommandStop = 0x73	Operation Stopped.
CommandDefaultParam = 0x61	Default Param in progress.
CommandSaveParam = 0x53	Save Param in progress.
CommandDecodeStart = 0x64	Scan Barcode in progress.
CommandBuzzerStart = 0x75	Start Buzzer in progress.
CommandVibratorStart = 0x76	Start Vibrator in progress.

6.1.2. ScanMode

Definitions	Descriptions
RFIDScanMode	RFID mode
BarcodeScanMode	Barcode mode

7. AsBarcodeType Class

7.1. Function

7.1.1. getBarcodeString

Function	+(NSString *)getBarcodeString:(BarcodeType)barcodeType;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
barcodeType	IN	BarcodeType	Enumerated type BarcodeType (see 7.2.1)
-	OUT	NSString	Text of the barcode type
■Function Description: The BarcodeType can be obtained as a string from the specified BarcodeType enumeration (see 7.2.1). **■Sample code:** NSString * barcodeName = [AsBarcodeType getBarcodeString: BarcodeTypeEAN13];			

7.2. Enum

7.2.1. BarcodeType

The following are the enumeration definitions of the read barcode types.

Enum Names	**Descriptions**
BarcodeTypeNoRead	NR (It will be returned when no barcode is scanned within a certain period of time.)
BarcodeTypeAustralianPost	Australian Post
BarcodeTypeAztecCode	Aztec Code
BarcodeTypeBooklandEAN	Bookland EAN
BarcodeTypeBritishPost	British Post

BarcodeTypeCanadianPost	Canadian Post
BarcodeTypeChinaPost	China Post
BarcodeTypeCodabar	Codabar
BarcodeTypeCodablockF	Codablock F
BarcodeTypeCode11	Code 11
BarcodeTypeCode128	Code 128
BarcodeTypeCode16K	Code 16K
BarcodeTypeCode32	Code 32
BarcodeTypeCode39	Code 39
BarcodeTypeCode49	Code 49
BarcodeTypeCode93	Code 93
BarcodeTypeComposite	EAN-UCC Composite Code
BarcodeTypeD2of5	Discreate 2 of 5
BarcodeTypeDataMatrix	Data Matrix
BarcodeTypeEAN128	UCC/EAN-128
BarcodeTypeEAN13	EAN-13
BarcodeTypeEAN13CouponCode	EAN-13 with Extended Coupon Code
BarcodeTypeEAN8	EAN-8
BarcodeTypeI2of5	Interleaved 2 of 5
BarcodeTypeIATA	IATA 2 of 5
BarcodeTypeISBT128	ISBT 128
BarcodeTypeISBT128Concat	ISBT-128 Concat.
BarcodeTypeJapanesePost	Japanese Post
BarcodeTypeKixPost	Kix (Netherlands) Post
BarcodeTypeKoreaPost	Korea Post
BarcodeTypeMacroMicroPDF	Macro Micro PDF
BarcodeTypeMaxiCode	MaxiCode
BarcodeTypeMicroPDF	Micro PDF 417
BarcodeTypeMSI	MSI
BarcodeTypeParameterFNC3	Parameter (FNC3)
BarcodeTypePDF417	PDF-417
BarcodeTypePlanetCode	Planet Code
BarcodeTypePlesseyCode	Plessey Code

BarcodeTypePostnet	Postnet
BarcodeTypeQRCode	QR Code
BarcodeTypeR2of5	Straight 2 of 5
BarcodeTypeRSS	RSS
BarcodeTypeScanletWebcode	Scanlet Webcode
BarcodeTypeTelepen	Telepen
BarcodeTypeTLC39	TCIF Linked Code 39
BarcodeTypeTriopticCode	Trioptic Code 39
BarcodeTypeUPCA	UPC-A
BarcodeTypeUPCE	UPC-E
BarcodeTypeVeriCode	VeriCode
BarcodeTypeX2of5	Matrix 2 of 5
BarcodeTypeRSSLimited	DataBar
BarcodeTypeChineseSensible	Chinese-Sensible Code

8. AsParamName Class

8.1. Function

8.1.1. getName

Function	+(NSString *)getName:(ParamName)paramName;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
paramName	IN	ParamName	Enumerated type ParamName (see 8.2.1)
-	OUT	NSString	Text of the barcode type.
■**Function Description:** Gets the barcode type as a string from parameters set with independent variables. ■**Sample code:** NSString * barcodeName = [AsParamName getName:EAN13];			

8.2. Enum

8.2.1. ParamName

Enumeration of barcode types can be set:

Enum Names	**Descriptions**
Codabar	Codabar
Code39	Code 39
Code32Pharmaceutical	Code 32 Pharmaceutical
I2of5	Interleaved 2 of 5
NEC2of5	NEC 2 of 5
Code93	Code 93
R2of5	Straight 2 of 5 Industrial

A2of5	Straight 2 of 5 IATA
X2of5	Matrix 2 of 5
Code11	Code 11
Code128	Code 128
GS1128	GS1-128
Telepen	Telepen
UPCA	UPC-A
UPCACouponCode	UPC-A/EAN-13 with Extended Coupon Code
CouponGS1DataBarOutput	Coupon GS1 DataBar Output
UPCE0	UPC-E0
UPCE1	UPC-E1
EAN13	EAN/JAP-13
EAN8	EAN/JAP-8
MSI	MSI
RSS14	RSS-14
RSSLimit	RSS Limited
RSSExp	RSS Expanded
TriopticCode	Trioptic Code
CodablockA	Codablock A
CodablockF	Codablock F
PDF417	PDF 417
MacroPDF417	MacroPDF417
MicroPDF	MicroPDF 417
ComCode	EAN/UCC Composite Code
GS1Emulation	GS1 Emulation
TLC39	TCIF Linked Code 39
ChinaPost	China Post
KoreaPost	Korea Post
QRCode	QR Code
Matrix	Data Matrix
MaxiCode	MaxiCode
AztecCode	Aztec Code
HanXinCode	Chinese Sensible (Han Xin) Code

PostalCodes	2D Postal Codes
Code39Pharmaceutical	Code 39 Pharmaceutical

9. AsParamValue Class

9.1. Properties

9.1.1. @property (assign, readwrite) ParamName paramName;

Description:
Gets or sets the type of barcode that AsReaderGUN can scan.
Enumerated type ParamName (see 8.2.1)

9.1.2. @property (assign, readwrite) unsigned int value;

Description:
Gets or sets whether the specified barcode type can be scanned by AsReaderGUN.
1: Can be scanned
0: Can not be scanned

9.2. Function

9.2.1. setEnabled

Function	- (void)setEnabled:(BOOL)value;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
value	IN	BOOL	Sets whether to allow scanning of a specified barcode type. YES: scanning is allowed NO: scanning is not allowed
■Function Description: Sets whether AsReaderGUN can scan the specified barcode type set by ParamName. **■Sample code:** AsParamValue * paramValue = [asReader getBarcodeParam: EAN13]; [paramValue setEnabled:NO];			

10. AsMaskActionType Class

10.1. Function

10.1.1. toString

Function	+(NSString*)toString:(MaskActionType)actionType targetType:(MaskTargetType)targetType;		
Parameters	**IN/OUT**	**Types**	**Descriptions**
actionType	IN	MaskActionType	Mask action type Enumerated type MaskActionType (see 5.1.9)
targetType	IN	MaskTargetType	Mask target type Enumerated type MaskTargetType (see 5.1.8)
-	OUT	NSString	Description of Mask action type and tag type
■Function Description: According the MaskActionType and MaskTargetType, gets the corresponding explanation through the string. **■Sample code:** NSString * actionMessage = [AsMaskActionType toString:MaskAction_AB targetType:MaskTarget_S0];			

11. Appendix

11.1. Prarameter Informations

※Only for ASR-L251G.

Properties	Default Value	Store in L251G F/W	Save in SDK instance	Restore to the Default Value When Power On
buzzer	ON (HIGH)	√		
vibrator	ON	√		
operationTime	0 (ms):Not use time		√	
inventoryTime	400(ms)			√
idleTime	300(ms)			√
sleepTime	65535(s)	√		
autoOffTime	65535(s)	√		
accessPassword	"0"			√
inventorySession	0:S0			√
sessionFlag	2:A or B			√
serialNumber		√		
continuousMode	YES			√
powerGain	300:30.0 dBm			√
isUseKeyAction				
useSelectionMask	No		√	
rssiMode	No		√	
epcMaskMatchMode	No		√	
algorithm	1:Dynmaic			√
minQ	0			√
maxQ	15			√
qValue	4			√
linkProfileValue	1			√
defaultLinkProfileValue				
maskTypeValue	0:NoMASK		√	