



# iOS Software Development Kit

How to Use StarIO for [DK-AirCash](#)

This SDK contains an Xcode Objective-C project for iOS devices.

## Tools Needed:

- Xcode 7.0 or later
- StarIO iOS SDK

In case of LAN I/F, DK-AirCash firmware Version 2.0 or later is required for this SDK.  
In case of Bluetooth I/F, DK-AirCash firmware Version 1.0 or later is required for this SDK.

To use StarIO.framework version 3.14.0 or later, you need to add the following frameworks.

- External Accessory framework
- Core Bluetooth framework
- To upgrade StarIO.framework when you are using version 3.13.1 or earlier, you need to add Core Bluetooth framework into your project.

Please refer to [here](#) for more information.

**StarIO SDK Compatibility OS :        iOS 7.0 or later**

**StarIO SDK Compatibility Chart**

Device	CPU
<b>iPad 2</b>	Armv7
<b>iPad (3rd Generation)</b>	Armv7
<b>iPad (4th Generation)</b>	Armv7s
<b>iPad Air</b>	Arm64
<b>iPad Air 2</b>	Arm64
<b>iPad mini</b>	Armv7
<b>iPad mini 2</b>	Arm64
<b>iPad mini 3</b>	Arm64
<b>iPad mini 4</b>	Arm64
<b>iPad Pro</b>	Arm64
<b>iPhone 4s</b>	Armv7
<b>iPhone 5</b>	Armv7s
<b>iPhone 5s</b>	Arm64
<b>iPhone 5c</b>	Armv7s
<b>iPhone 6</b>	Arm64
<b>iPhone 6 Plus</b>	Arm64
<b>iPhone 6s</b>	Arm64
<b>iPhone 6s Plus</b>	Arm64
<b>iPod touch (5th Generation)</b>	Armv7
<b>iPod touch (6th Generation)</b>	Arm64

*Note:* iPad, iPhone, iPod, and iPod touch are trademarks of Apple Inc., registered in the U.S. and other countries. iPad Air and iPad mini are trademarks of Apple Inc. iOS is a trademark or registered trademark of Cisco in the U.S. and other countries and is used under license.




## Table of Contents

- ❖ [About this Manual](#)
- ❖ [Star Printer Compatibility Chart](#)
- ❖ [Connecting a Star Printer to an iOS Device](#)
- ❖ [Getting Started](#)
- ❖ [Using the SDK with Star Micronics POS Printers](#)
- ❖ [Overview of how this iOS SDK is designed](#)
- ❖ [StarIO Framework](#)
- ❖ [The StarIO Methods Overview](#)
  - [SMPort Class](#)
  - [SMBluetoothManager Class](#)
- ❖ [Functionality](#)
  - [Help](#)
  - [Get Cash Drawer Firmware Information](#)
  - [Get Printer Status](#)
  - [Get Cash Drawer Status](#)
  - [Sample Receipt + Open Cash Drawer](#)
  - [Open Cash Drawer](#)
  - [Bluetooth Pairing + Connect](#)
  - [Bluetooth Disconnect](#)
- ❖ [Tips for software application development when using StarIO](#)
- ❖ [Additional Resources](#)
- ❖ [SDK Version History](#)

## About this Manual

This manual is designed to help you understand StarIO and how to build an iOS application to interact with Star DK-AirCash. It is important to understand the basics of the Objective-C language. Although this SDK is for iOS, there are SDKs available for many different operating systems and programming languages at our website in the Developers section. Check the Developers section of our site for the newest SDKs, technical documentation, FAQs, and many more additional resources.

### Key Legend:

<i>Warning</i>		Explains potential issues
<i>Avoid Doing This</i>		Explains things not to do
<i>Note</i>		Provides important information and tips

### CAUTION:

- The information in this manual is subject to change without notice.
- STAR MICRONICS CO., LTD. has taken every measure to provide accurate information, but assumes no liability for errors or omissions.
- STAR MICRONICS CO., LTD. is not liable for any damages resulting from the use of information contained in this manual.
- Reproduction in whole or in part is prohibited.

## Connecting a Star Device to an iOS Device

### Wired LAN / Wireless LAN Interface

Star DK-AirCash ship with DHCP enabled by default. If your network supports DHCP, be sure to make the necessary configurations so that your Star DK-AirCash will automatically get an IP Address.

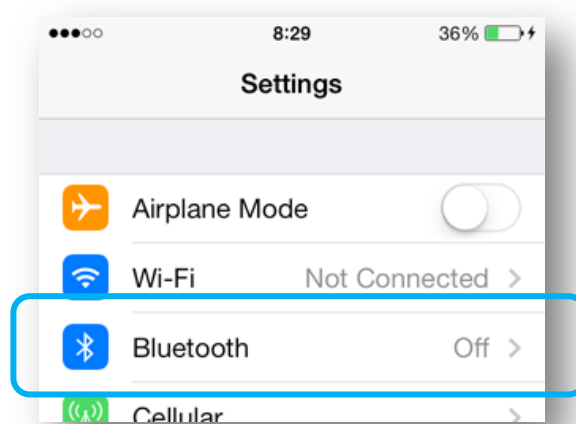
Use Star POS devices with the #9100 Multi Session disabled.

Please refer to "guidelines-ethernet\_win\_en.pdf" for how to confirm and change the #9100 Multi Session setting and how to set the Static IP Address.

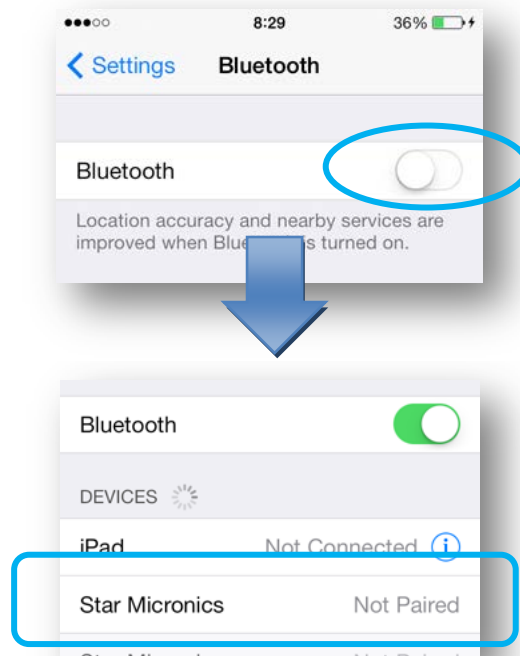
### Bluetooth Interface

#### ◆Pairing

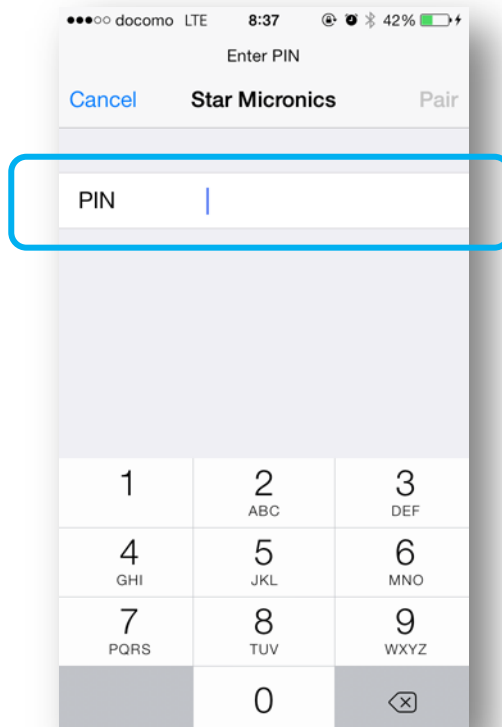
1. Make sure that the interface selection of Star POS device you want to configure the settings is set to Bluetooth, and turn the device on.  
If SSP is used by the Star POS device, press the PAIR button for 5 seconds or more to start pairing.
2. Tap Settings > Bluetooth.



3. Tap Bluetooth to turn it on. Your iOS device searches and displays the Bluetooth devices in range. Tap the Star portable printer and DK-AirCash you want to pair with.



4. If a PIN code is used for Star Bluetooth device pairing, enter the PIN and tap Pair.



5. When the pairing is complete, you'll see this message.



#### ◆How to change the Bluetooth Device Name

The Star Bluetooth Utility can be downloaded from Apple App Store to change the iOS Port Name.

To confirm iOS Port Name, select [Settings]-[General]-[About] after Bluetooth pairing is established. The iOS Port Name will be shown under the Bluetooth address.

## Getting Started

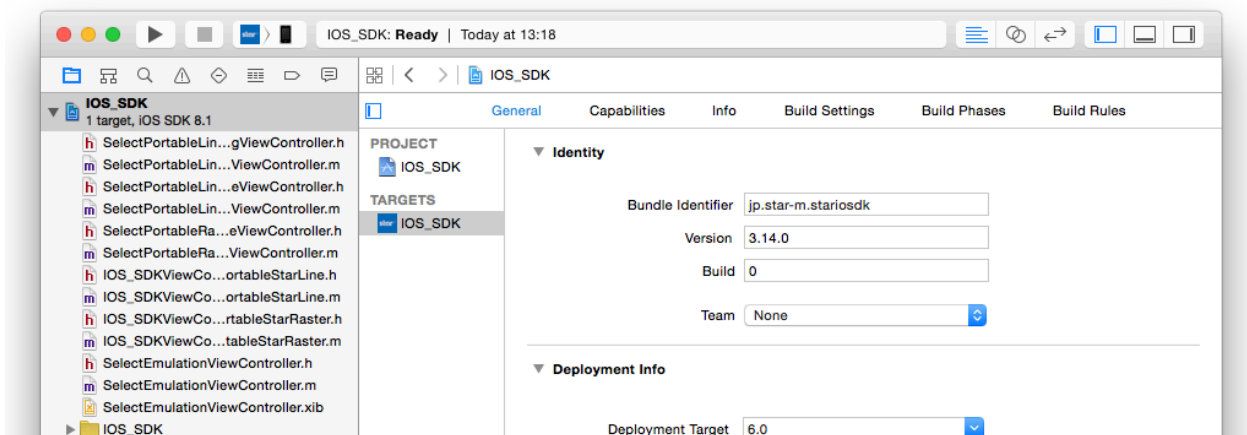
To build an iOS project, Xcode are needed. These tools are available in one package from the [Apple Developer Site](#) or Apple App Store. It is important to note that in order to produce applications that will actually run on an iOS device, you must be part of the Apple Developer Program, which requires a yearly subscription. While it is possible to obtain these tools from Apple App Store as stated above, your application will only be able to run in the iOS Simulator and will not install on an actual device.

It is assumed Xcode have already been installed on your Mac at this point. Should you need assistance or additional information, visit the [Resources](#) section of the Apple Developer Site.

How to open the Star iOS SDK project in Xcode:

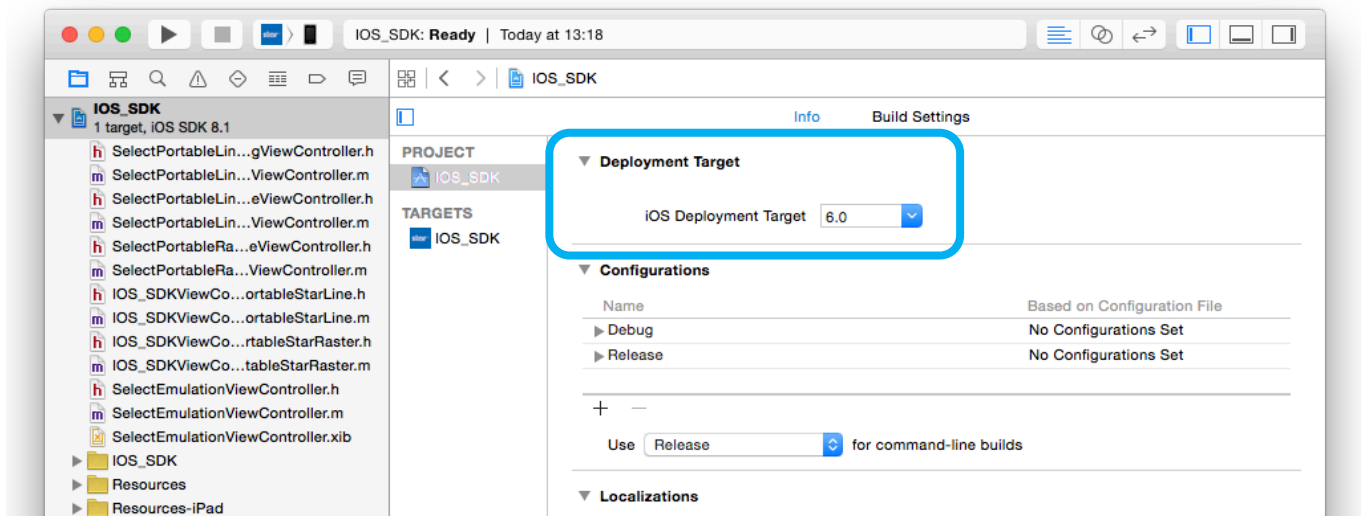


1. Unzip the Star iOS SDK folder and open it.



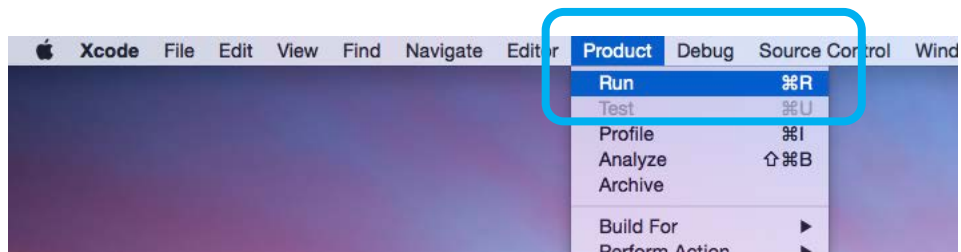
2. Open IOS\_SDK.xcodeproj.





- .3. Set the iOS Deployment Target to 6.0 or later.

Running the project:



1. Use the shortcut ⌘R or click Product in the top menu bar and then Run.

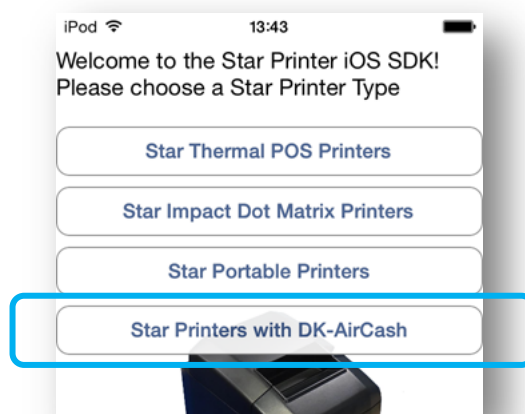
## Using the SDK with Star Micronics Devices

### Port Name and Interface Relation:

StarIO uses specific port names to identify what port will be used. These are very important to understand because not following the naming convention correctly will fail to communicate with the printer.

Interface	Port Name
Wired LAN/Wireless LAN I/F (TCP/IP)	TCP: "IP Address"
Bluetooth	BT: "iOS Port Name" * _

### Using a DK-AirCash:

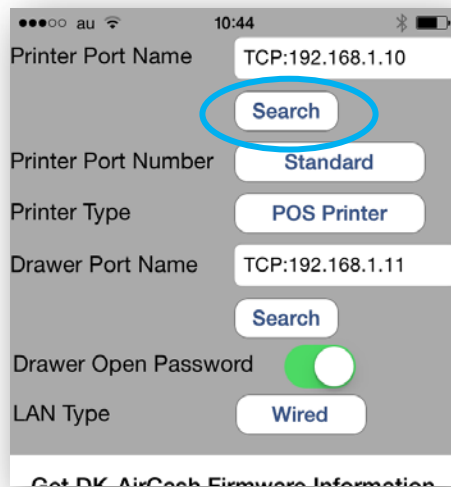


1. Tap "Star Printer with DK-AirCash".

## Wired LAN / Wireless LAN and Bluetooth device setting

### ·To search for the printer and configure connection

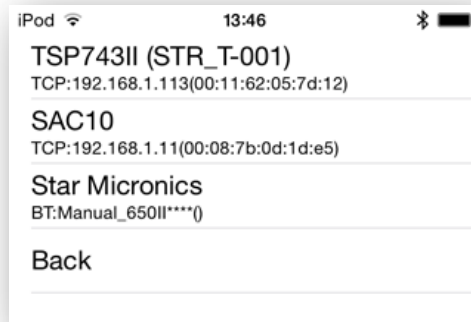
1. Tap "Search" to find all connected Star Printers.



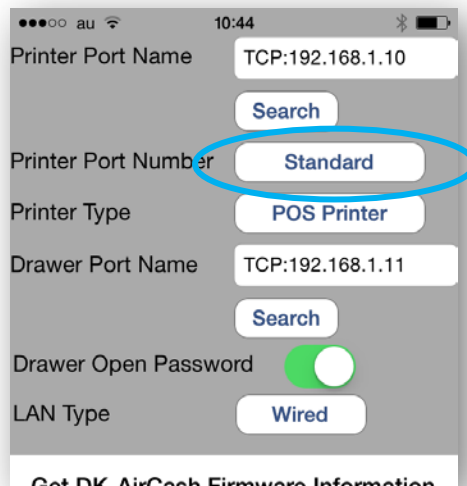
2. Tap the interface type of the printer you want to connect to.  
When tapping "LAN", the model names of Wired LAN/Wireless LAN printers you can connect to are displayed.  
When tapping "Bluetooth", the port names of paired printers you can connect to are displayed.



3. Tap the name of printer you want to connect to.

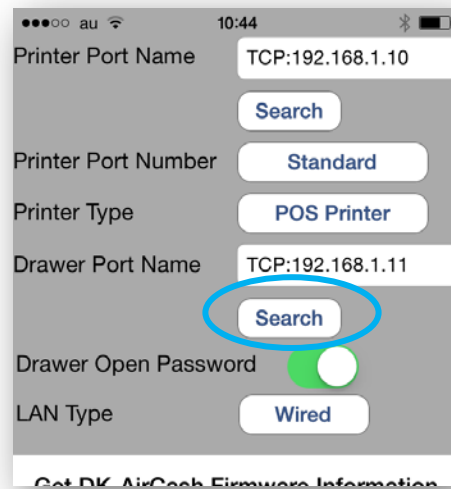


4. In Wired LAN / Wireless LAN, configure the TCP Port by clicking the “Standard” dropdown if necessary. This sample application allows you to choose any port from 9100 to 9109. Configuring the port might be necessary when using a router such as Apple’s AirPort Express.

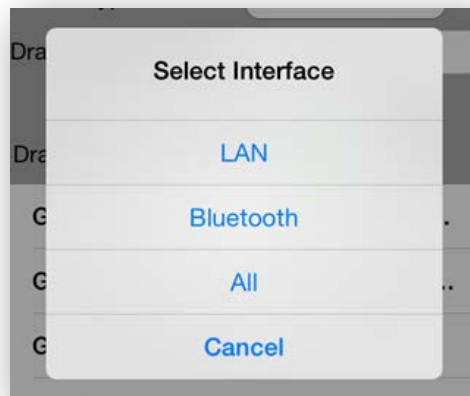


·To search for the DK-AirCash and configure connection

1. Tap "Search" .



2. Tap the interface type of the DK-AirCash you want to connect to.  
When tapping "LAN", the model names of DK-AirCash in Wired LAN/Wireless LAN you can connect to are displayed.  
When tapping "Bluetooth", the port names of paired DK-AirCash in Bluetooth you can connect to are displayed.



3. Tap the name of device you want to connect to.



· **To configure connection manually**

1. Type the IP Address or the Bluetooth Device Name manually in the "PortName" field as shown below.

**For Wired LAN / Wireless LAN Interface**

Type: TCP:<IP Address> \*Without brackets

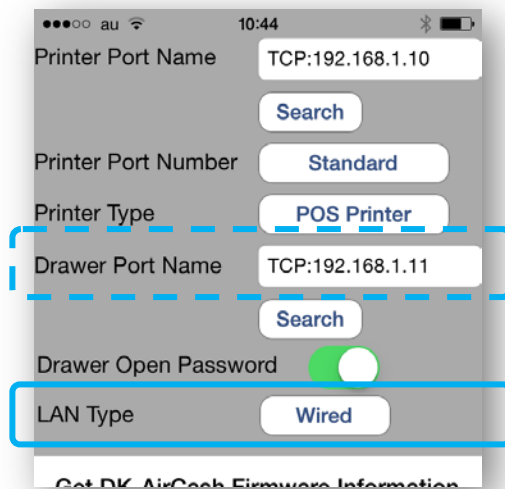
Printer Port Name TCP:192.168.1.10  
Search  
Printer Port Number Standard  
Printer Type POS Printer  
Drawer Port Name TCP:192.168.1.11  
Search  
Drawer Open Password ☒  
LAN Type Wired  
Get DK AirCash Firmware Information

**For Bluetooth Interface**

Type: BT:<Port Name of Bluetooth Device> \*Without brackets

Printer Port Name BT:Star Micronics  
Search  
Printer Port Number Standard  
Printer Type POS Printer  
Drawer Port Name BT:DK-AirCash  
Search  
Drawer Open Password ☒  
LAN Type Wired  
Get DK AirCash Firmware Information

· **Wired LAN/Wireless LAN – How to select LAN type.**



When the DK-AirCash which is specified by Drawer Port Name is Wired or Wireless LAN device, select the LAN type for your connection.

In case of wired LAN connection, select "Wired". " " (a empty string) is specified for the portSetting parameter of getPort method.

In case of wireless LAN connection, select "Wireless". ";wl" is specified for the portSetting parameter of getPort method.

## Overview of How This iOS SDK is Designed

This overview will touch briefly on key components of the SDK.

All functionality is located in the IOS\_SDK project and IOS\_SDK target.

Run the program from the IOS\_SDKViewControllerDKAirCash.m file; this source code is the starting point for a DK-AirCash.

See how specific functions work by clicking on the other source files. For example, “code128.m” corresponds to the 1D barcode Code128 in the GUI.

It is important to note that not every function is available for a DK-AirCash. The first page of each SDK manual shows which functions are supported.



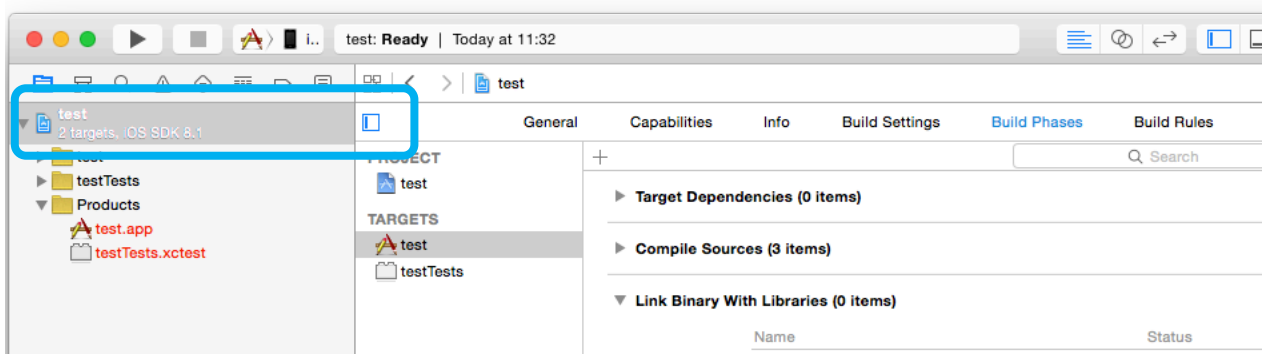


## The StarIO Framework

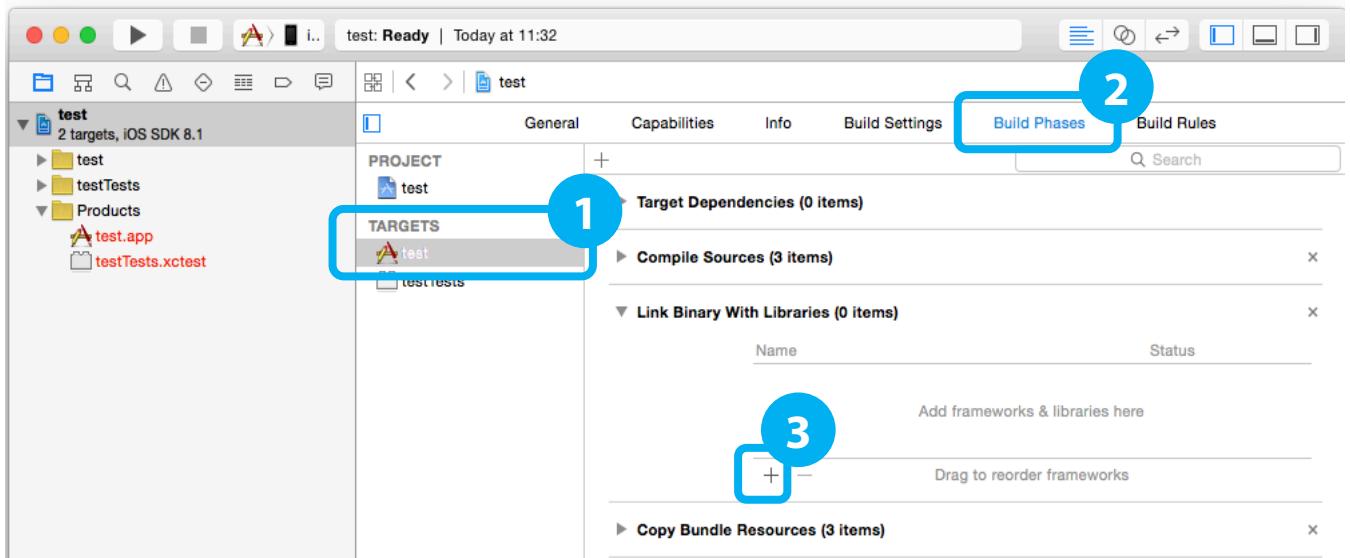
The StarIO framework is already included when the Star iOS SDK is loaded in Xcode; there's no need to include it again when testing our SDK. However, when you are building your own application, it is necessary to add the StarIO framework into it to utilize the StarIO methods.

### ◆When building a new application

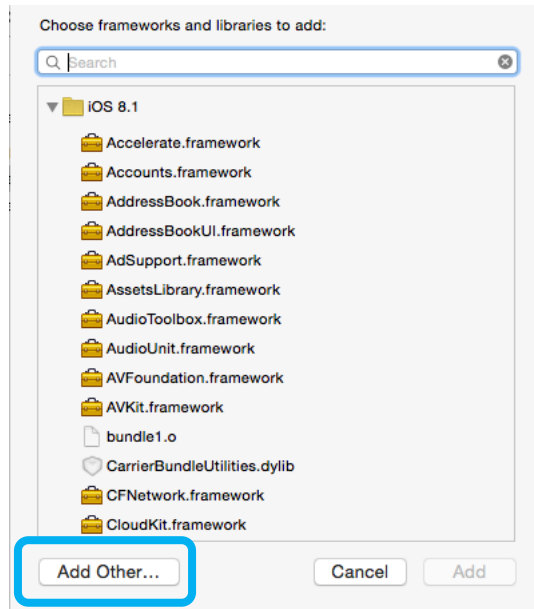
#### 1. Add StarIO.framework into your project



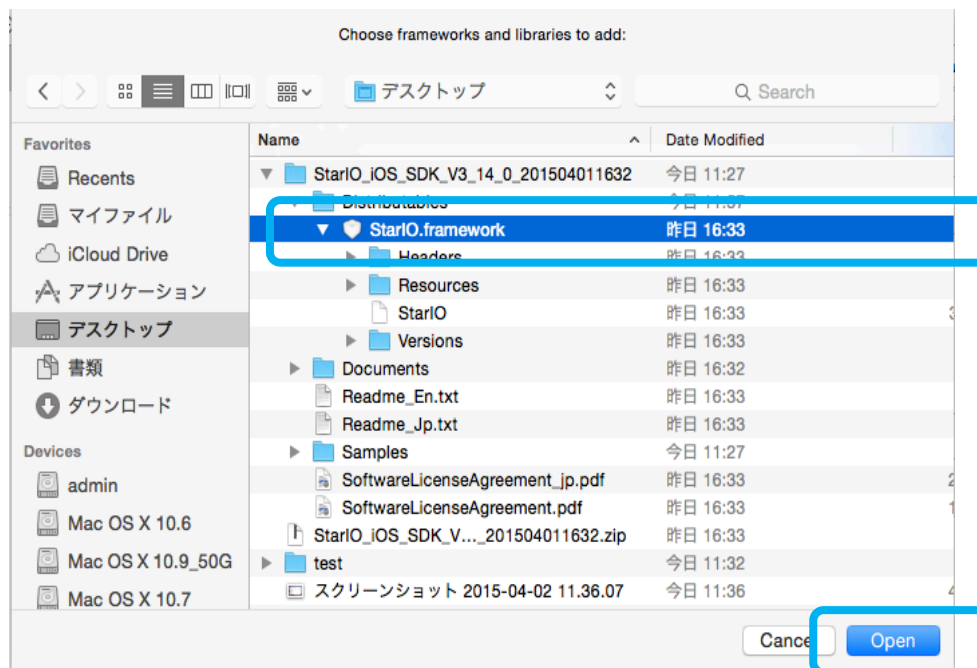
1. Click the created project.



2. Open a target, click the Build Phases tab, click the + of Link Binary With Libraries.

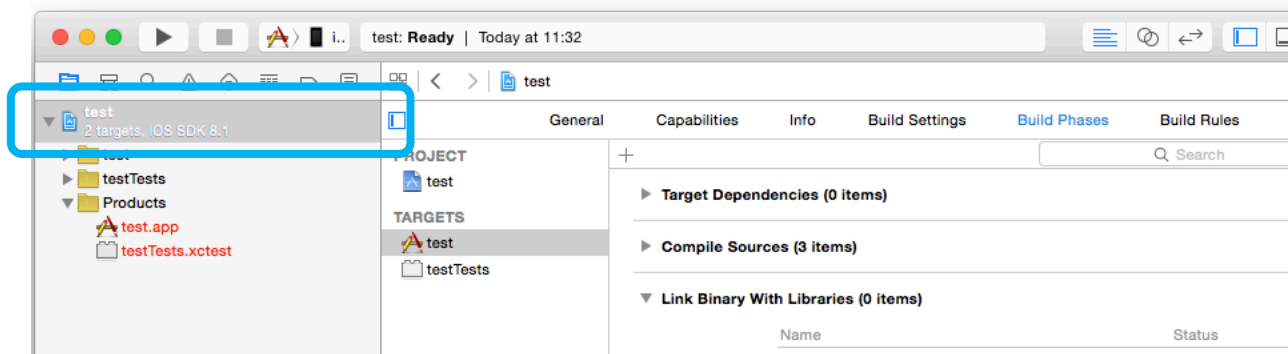


3. Click the **Add Other...** button.

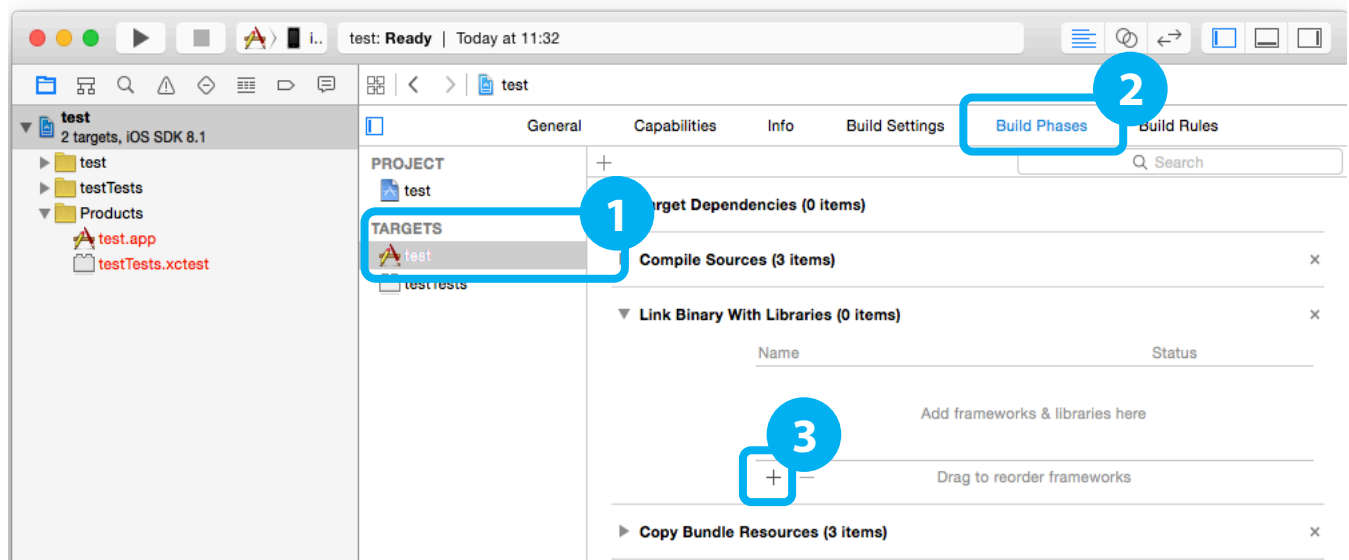


4. Browse to the location of where the Star iOS SDK was unzipped and select StarIO.framework. Then click Open.
5. The framework is added to your project and all StarIO methods are now available to you.

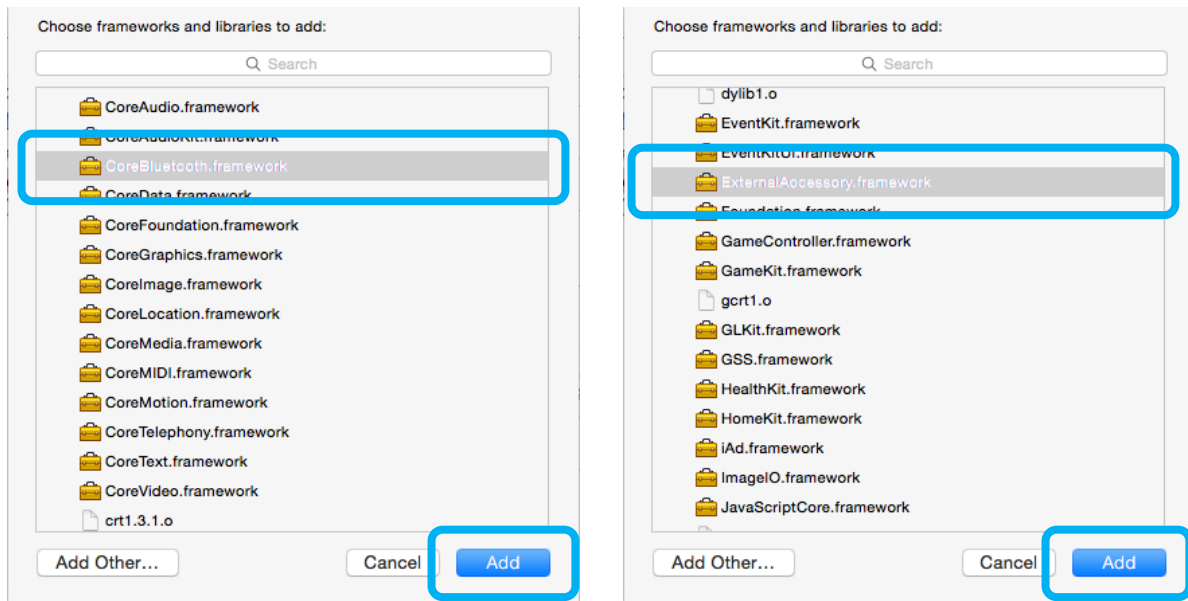
## 2. Add other framework into your project.



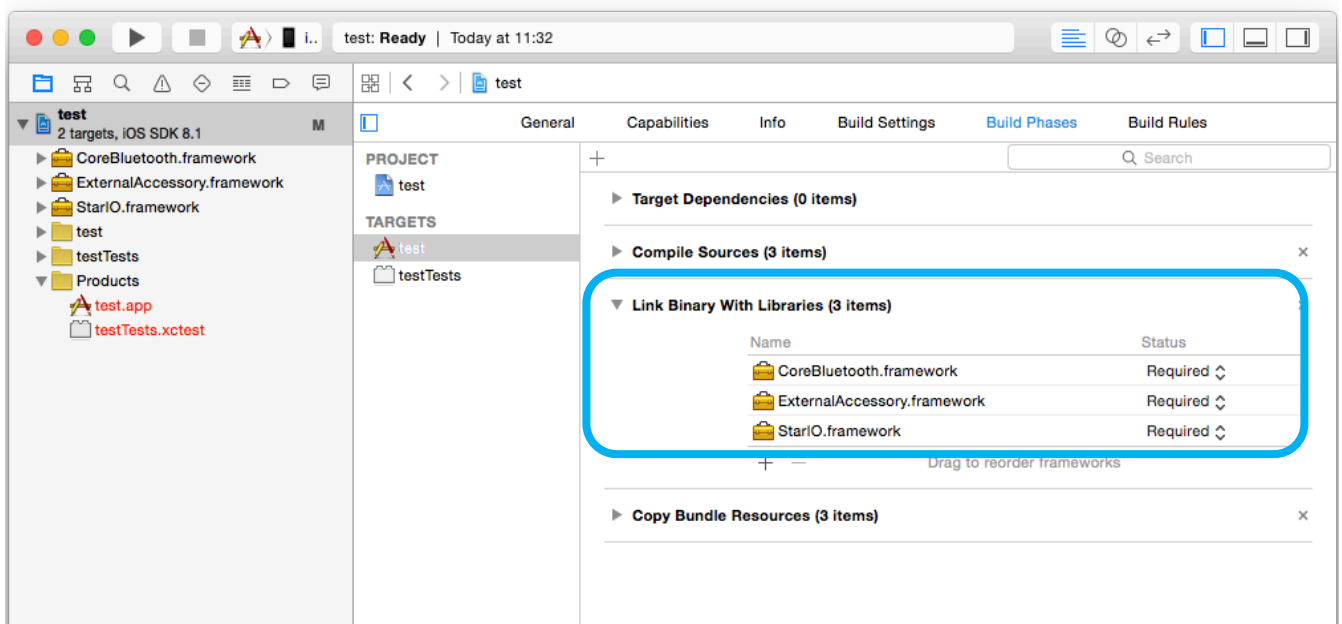
1. Click the created project.



2. Open a target, click the Build Phases tab, click the + of Link Binary With Libraries.



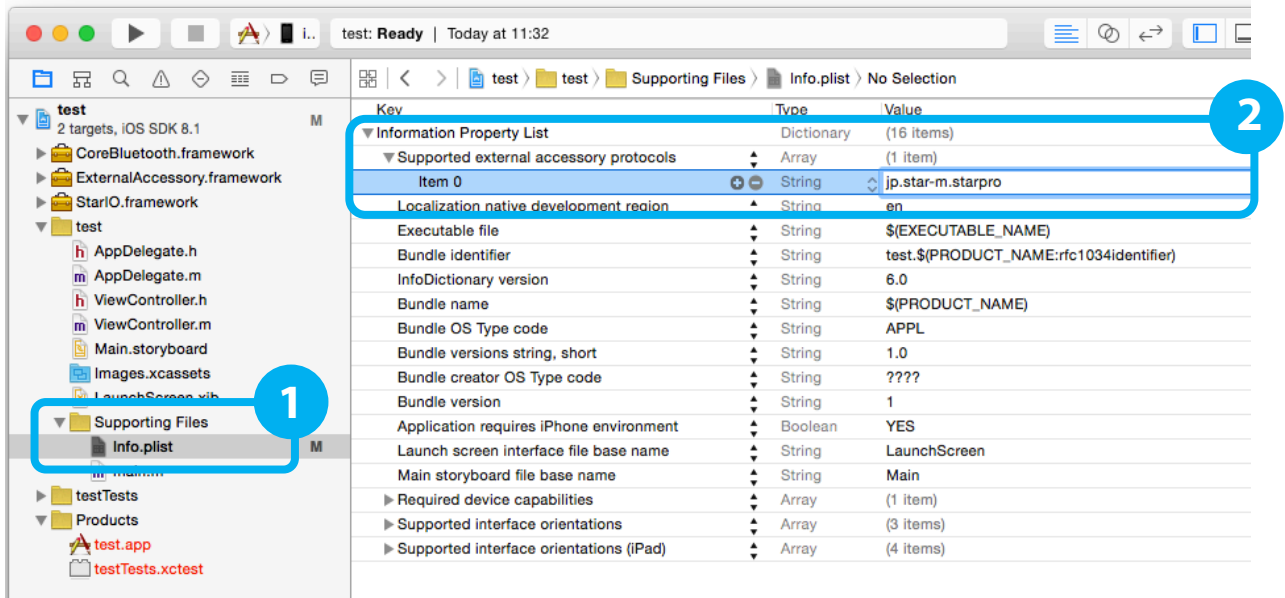
3. External Accessory framework and Core Bluetooth framework are added respectively. Select the framework, click the Add button.



4. Check if the necessary framework has been added.

### 3. Edit information property list (Bluetooth Interface only)

*Note:* Please do not apply this, if you are not using Bluetooth ineterface.



1. Click on the information property list file (default: Info.plist file).
2. Add the Supported external accessory protocols key. Click the triangle of this key and set the value for the Item 0 to jp.star-m.starpro.
4. You have finished editing the information property list.

### ◆Version up of StarIO.framework

---

1. Delete StarIO.framework from your project.
2. Copy new StarIO.framework
3. Clean the Xcode project.  
-Open the Xcode project and select [Build]-[Clean] from the menu.
4. Build the Xcode project.



To refer to the new StarIO.framework without deleting the existing StarIO.framework, surely confirm the “framework search path” setting of the Xcode project.

If the old path of the StarIO.framework remains in front of the “framework search path”, the previous StarIO.framework will be used.

## The StarIO Methods Overview

### SMPort Class:

#### ● Property

<b>portName</b>	Acquires the printer port name.
<b>portSettings</b>	Acquires the port settings.
<b>timeoutMillis</b>	Acquires and specifies the timeout time for internal control and API.
<b>endCheckedBlockTimeoutMillis</b>	Acquires and specifies the timeout time for endCheckedBlock method.

`- (NSString *)portName`

Specifies the port of the printer.

`- (NSString *)portSettings`

Acquires the port settings.

`- (u_int32_t)timeoutMillis`

Acquires and specifies the timeout time for internal control and API. (unit: millisecond)

`@property(assign, readwrite) u_int32_t endCheckedBlockTimeoutMillis`

It obtains and sets endCheckedBlock method timeout value [unit: ms]

If it takes long time to print, stand-by time for print completion in endCheckedBlock method can be extended by increasing this value.

Default value is the timeout value designated by getPort method.

#### Precautions in case the button security is enabled (North America only)

If the button security is utilized, please call beginCheckedBlock prior to and endCheckedBlock posterior to the drawer open command transference.

Please do not change this property, since it is automatically adjusted optimally by beginCheckedBlock method.

## ●Method

### getPort

```
+ (SMPort *) getPort: (NSString *) portName : (NSString *) portSettings : (uint32_t) TimeoutMillis)
```

GetPort is what you will be using to “open” the port to the printer.

#### Parameters:

portName        - Specify the communication port to the printer.

*Ex. @"TCP:192.168.1.2" ( In Wired LAN / Wireless LAN )*

*@"BT:StarMicronics" ( In Bluetooth )*

*@"BT:00:11:62:1b:4d:f4"(To specify the MAC address in Bluetooth)*

***Note: iOS6 is required to specify the MAC address in Bluetooth.  
Other iOS versions cannot use this function.***

[Use share printer function with Apple AirPort Express]

Set AirPort Express IP Address for portName.

*Ex. @"TCP:192.168.1.2"*

portSettings    - when using a wireless LAN, specify @";wl".  
                  - when using a connection other than a wireless LAN, specify a empty string (@"").

*Ex. @"" ( for Wired LAN or Bluetooth )*

*@"wl" ( for Wireless LAN )*

timeoutMillis    - Specify the timeout time for internal control and API.

***Note: this parameter guarantees that all of the below APIs will  
complete in a bounded amount of time, but does NOT guarantee  
the exact timeout length).***

[Use share printer function with Apple AirPort Express]

Set port number for portSettings.

Increase the port number in sequential order from 9100 to 9109 until  
communication is successful.

*Ex. @"9100"*



## Returns:

An instance of SMPort class. It returns "nil" if it fails to generate communication port.



After executing `getPort`, please do not forget `releasePort` before executing the next `getPort`.

Otherwise the communication may return nil.

*//The following would be an actual usage of getPort:*

```
SMPort *port = nil;
NSString *portName = @"TCP:192.168.0.5";
NSString *portSettings = @"";
@try
{
    port = [SMPort getPort:portName :portSettings :10000];
}
@catch (PortException)
{
    //There was an error opening the port
}
```



Always use a `try`, `catch` when using `getPort`. If the port cannot be opened because of connection problems, your program will crash unless you use a `try`, `catch` like the above example.

## searchPrinter

```
+ (NSArray *) searchPrinter;

+ (NSArray *) searchPrinter: (NSString *) target
```

searchPrinter detects DK-AirCashes in LAN and returns search result as NSArray.

NSArray of return value includes instance of PortInfo Class.

PortInfo class of return value includes, PortName, MAC address, ModelName and you can get them by portName, macAddress, and modelName property.

And you can use Port Name as Argument value of getPort.

### Parameters:

- Target - When @"TCP:" is specified, Wired LAN / Wireless LAN DK-AirCashes will be detected.
- When @"BT:" is specified, Bluetooth DK-AirCashes will be detected.



This API do not guarantee the discovery of devices.

iOS6 is required to specify the MAC address in Bluetooth.

Other iOS versions cannot use this function.

//The following would be an actual usage of searchPrinter:

```
NSArray *portArray = [[SMPort searchPrinter] retain];
for (int i = 0; i < portArray.count; i++) {
    PortInfo *port = [portArray objectAtIndex:i];
    NSLog(@"Port Name: %@", port.portName);
    NSLog(@"MAC Address : %@", port.macAddress);
    NSLog(@"Model Name: %@", port.modelName);
}
[portArray release];
```

The above example shows DK-AirCashes being detected and search result being output to the log.

## readPort

```
- (u_int32_t) readPort: (u_int8_t *) readBuffer : (u_int32_t *) offset : (u_int_32_t) size;
```

This method reads data from the device. Only use this if you really need to read raw bytes from the DK-AirCash.



**Do not use this method to read status.**

Use getParsedStatus:: for getting status.

### Parameters:

`readbuffer` – A Byte Array buffer into which data is read.

`offset` - specifies where to begin writing data into the `readBuffer[]`

`size` – Total number of bytes to read.

### Returns:

The number of bytes that were actually read. Under some interface types, this function will succeed even when no data was read in. Your application should call this function a limited number of times until the expected data has been read in or until an application determined retry threshold has been reached.

### Throws:

`PortException` - when a communication failure occurs

## releasePort

```
+ (void) releasePort: (SMPort *) port;
```

This function closes a connection to the port specified.

### Parameters:

`port` - `StarIOPort` type representing a previously initialized port.



After executing `getPort`, please do not forget `releasePort` before executing the next `getPort`.

Otherwise the communication may return nil.

## writePort

```
-(u_int32_t) writePort: (u_int8_t const *) writeBuffer: (u_int32_t) offset: (u_int32_t) size;
```

This method writes data to the device. Use this to send commands to the DK-AirCash.

To check the completion of command transfer, run `beginCheckedBlock` before and `endCheckedBlock` after this method.

See the sample code [here](#).

※Remember to use a Try, Catch for safe programming practices.

The SDK has code in “PrintTextWithPortName” that will show you how to verify data transmission to the DK-AirCash.

### Parameters:

`writeBuffer` - Contains the output data in a byte array.

`offset` - Specifies where to begin pulling data from `writeBuffer` .

`size` - Number of bytes to write.

### Returns:

The number of bytes that were actually written. Under some interface types, this function will succeed even when no data was written out. Your application should call this function a limited number of times until all the data has been written out or until an application determined retry threshold has been reached.

### Throws:

`PortException` - when a communication failure occurs

## getParsedStatus

```
-(void) getParsedStatus: (void *) starPrinterStatus: (u_int32_t) level;
```

This method retrieves detailed status from the DK-AirCash with StarIO.

**Returns:**

[StarPrinterStatus](#) structure giving the current DK-AirCash status

**Throws:**

[PortException](#) - when a communication failure occurs

This method uses a class structure that is included with StarIO called [StarPrinterStatus](#). This structure gives the printer's status in both boolean and binary form. Create the [StarPrinterStatus](#) object in your project by doing the following:

```
StarPrinterStatus_2 printerStatus;
[port getParsedStatus:&printerStatus :2];
if (printerStatus.offline == SM_TRUE)
{
    if (printerStatus.coverOpen == SM_TRUE) {
        //There was a cover opne error
    }
    else if (printerStatus.receiptPaperEmpty == SM_TRUE) {
        //There was a receipt paper empty error
    }
    else {
        //There was a offline error
    }
}
else {
    //If False, then the printer is online.
}
```

Status List of the class structure **StarPrinterStatus**

mamber name	contents	Type	Detail	DK-AirCash
<b>blackMarkError</b>	Black Mark Error	SM_BOOLEAN	DK-AirCashes are not supported.	
<b>compulsionSwitch</b>	Compulsion SW	SM_BOOLEAN	You can check status of CashDrawer (Open or Close) SM_TRUE : Compulsion SW is pressed. SM_FALSE : Compulsion SW is not pressed.	✓
<b>coverOpen</b>	Cover Status	SM_BOOLEAN	DK-AirCashes are not supported.	
<b>cutterError</b>	Auto-cutter Error	SM_BOOLEAN	DK-AirCashes are not supported.	
<b>etbAvailable</b>	ETB available or not	SM_BOOLEAN	SM_TRUE : available to use SM_FALSE : not available to use	✓
<b>etbCounter</b>	ETB Counter	UCHAR	You can get current value of ETB	✓
<b>headThermistorError</b>	Head Thermistor Error	SM_BOOLEAN	DK-AirCashes are not supported.	
<b>offline</b>	ONLINE/OFFLINE Status	SM_BOOLEAN	You can check status of Online or offline. SM_TRUE : Printer is Offline. SM_FALSE : Printer is Online	✓
<b>overTemp</b>	Stopped by high head temperature	SM_BOOLEAN	DK-AirCashes are not supported.	
<b>presenterPaperJamError</b>	Presenter Paper Jam Error	SM_BOOLEAN	DK-AirCashes are not supported.	
<b>presenterState</b>	Presenter Paper Position	UCHAR	DK-AirCashes are not supported.	
<b>raw</b>	Byte column of status	UCHAR[63]	Byte column of status (example : HEX 23 86 00 00 00 00 00 00)	✓
<b>rawLength</b>	raw length	CHAR	raw length	✓
<b>receiptPaperEmpty</b>	Paper end	SM_BOOLEAN	DK-AirCashes are not supported.	
<b>receiptPaperNearEmptyInner</b>	Paper Near-end (Inner Side)	SM_BOOLEAN	DK-AirCashes are not supported.	
<b>receiveBufferOverflow</b>	Receive Buffer Overflow	SM_BOOLEAN	DK-AirCashes are not supported.	
<b>unrecoverableError</b>	Non-recoverable Error	SM_BOOLEAN	SM_TRUE : Unrecoverable error occurs. SM_FALSE : Unrecoverable error does not occur. Unrecoverable error : Head Thermistor Error, Auto-cutter Error, Electric Voltage Error and etc.)	✓
<b>voltageError</b>	Electric Voltage Error	SM_BOOLEAN	DK-AirCashes are not supported.	

## beginCheckedBlock

```
-(void) beginCheckdBlock: (void *) starPrinterStatus: (u_int32_t) level;
```

This method is used in combination with endCheckedBlock and checks the completion of data transmission. beginCheckedBlock must be run just before sending data.

### Parameters:

starPrinterStatus - a pointer to StarPrinterStatus structure

(Possible to specify StarPrinterStatus, StarPrinterStatus\_1 of StarPrinterStatus\_2. Normally StarPrinterStatus\_2 is specified.)

level - the level of StarPrinterStatus structure

(Possible to specify a value of 0,1 or 2. Normally 2 is specified.)

See the sample code [here](#).

### In case the button security is enabled (North America only)

beginCheckedBlock reads the timeout setting of the DK-AirCash and sets setEndCheckedBlockTimeoutMillis to the proper value automatically.

## endCheckedBlock

```
-(void) endCheckdBlock: (void *) starPrinterStatus: (u_int32_t) level;
```

This method is used together with the beginCheckedBlock method in a set.

It monitors DK-AirCash status and when the transferred data is processed completely by DK-AirCash, returns control.

In case the transfer of data is not completed before the timeout (\*1) or an error occurs during command execution, it returns PortException.

(\*1) To timeout value, endCheckedBlockTimeoutMillis property is applied. Default value is the timeout value designated by getPort. Please adjust the endCheckedBlockTimeoutMillis value to be longer than printing time.

Timeout length is specified by getPort, endCheckedBlockTimeoutMillis or is 10 seconds if specified less than 10 seconds.

### Parameters:

starPrinterStatus - a pointer to StarPrinterStatus structure

(Possible to specify StarPrinterStatus, StarPrinterStatus\_1 of StarPrinterStatus\_2. Normally StarPrinterStatus\_2 is specified.)

level - the level of StarPrinterStatus structure

(Possible to specify a value of 0,1 or 2. Normally 2 is specified.)

### Returns:

StarPrinterStatus structure giving the current device status

### Throws:

PortException - when a communication failure\* occurs

\*Examples) - An error sending the command (such as Off-Line)  
- No response for the completion from the DK-AirCash within the timeout

### In case the button security is enabled (North America only)

If this method is used after having transferred drawer open command, it returns control either when button is pressed or timeout is over before the button is pressed.

You can confirm the drawer status by getParsedStatus method 150ms after control is returned.



```

unsigned char command[] = {0x07};
uint bytesWritten = 0;

StarPrinterStatus_2 starPrinterStatus;
SMPort *port = nil;

@try
{
    port = [SMPort getPort:@"TCP:192.168.1.10" :@"":10000];

    //Start checking the completion of printing
    [port beginCheckedBlock:&starPrinterStatus :2];

    if (starPrinterStatus.offline == SM_TRUE)
    {
        //There was an error writing to the port
    }

    while (bytesWritten < sizeof (command)) {
        bytesWritten += [port writePort: command : bytesWritten : sizeof (command) - bytesWritten];
    }

    //End checking the completion of printing
    [port endCheckedBlock:&starPrinterStatus :2];

    if (starPrinterStatus.offline == SM_TRUE)
    {
        //There was an error writing to the port
    }
}
@catch (PortException)
{
    //There was an error writing to the port
}
@finally
{
    [SMPort releasePort:port];
}

```

## disconnect

```
-(BOOL) disconnect
```

This method disconnects the specified Bluetooth device.

After the disconnection, the Bluetooth device can be connected by other iOS terminals.

This method fails in the following cases:

- when the disconnection has not been completed within the timeout specified by `getPort`
- when the disconnection function is not supported by a printer (such like portable printers).

This method has no effect on Wired LAN / Wireless LAN devices.

### Returns:

It returns YES when succeeded and NO when failed.

It always returns YES when it was run with the Wired LAN / Wireless LAN device.

## getFirmwareInformation

```
-(NSDictionary *) getFirmwareInformation:
```

This method gets a firmware Information of the printer.

### Returns:

It returns `NSDictionary` as an acquisition result.

Gets a model name from the return value by setting the Key to `@modelName`.

Gets a firmware version from the return value by setting the Key to `@firmwareVersion`.

### Throws:

`PortException` - when a communication failure occurs

### Note:

- If it failed to get information, it returns an empty string.

## StarIOVersion

```
+(NSString *) StarIOVersion
```

This method gets the StarIO version.

**Returns:**

StarIO version

## getDipSwitchInformation

```
-(NSDictionary *) getDipSwitchInformation
```

This method gets a Dip Switch Information of the DK-AirCash.

**Returns:**

It returns [NSDictionary](#) as an acquisition result.

Gets a Dip Switch information from the return value.

**Throws:**

[PortException](#) - when a communication failure occurs

**Note:**

- If it failed to get information, it returns an empty string.
- DK-AirCash firmware Version 2.0 or later is required for this method.

## SMBluetoothManager Class:

SMBluetoothManager Class specifies various settings of the Bluetooth interface.  
It can not be used with SMPort Class.

### ●Property

<b>portName</b>	Acquires the portName of the device to be connected.
<b>deviceType</b>	Acquires the type of the device to be connected.
<b>opened</b>	Shows whether the port is opened.
<b>deviceName</b>	Acquires and specifies the current Bluetooth device name.
<b>iOSPortName</b>	Acquires and specifies the port name to be used with the StarIO.
<b>autoConnect</b>	Acquires and specifies the setting (Valid or Invalid) of the autoconnection function.
<b>security</b>	Acquires the Bluetooth security setting (SPP or PIN Mode)
<b>pinCode</b>	Specifies the PIN Code to be used for pairing.

```
@property(nonatomic, readonly) NSString *portName
```

Creates an instance of SMBluetoothManager.

```
@property(nonatomic, readonly) SMDeviceType deviceType
```

Acquires the type of the device to be connected.

```
@property(nonatomic, readonly) BOOL opened
```

Shows whether the port is opened.

It returns YES if the open method was successful.

Then it will return NO when the close method is called.

```
@property(nonatomic, retain) NSString *deviceName
```

Acquires and specifies the current Bluetooth device name.

The current setting is read when the loadSetting method is called.

To set it, run the apply method after changing this property.

*Valid characters: 0-9, a-z, A-Z*

`@property(nonatomic, retain) NSString *iOSPortName`

Acquires and specifies the iOS port name to be used with the StarIO.  
The current setting is read when the loadSetting method is called.  
To set it, run the apply method after changing this property.

***Valid characters:***

0-9 a-z A-Z ; : ! ? # \$ % & , . @ \_ - = Space / \* + ~ ^ [ { ( ) } ) | \

`@property(nonatomic, assign) BOOL autoConnect`

Acquires and specifies the setting of the auto connection function.  
The current setting is read when the loadSetting method is called.  
To set it, run the apply method after changing this property.



Set to NO when the security setting is set to PIN code mode.

`@property(nonatomic, assign) SMBluetoothSecurity security`

Acquires and specifies the Bluetooth security setting(SSP or PIN code mode).  
The current setting is read when the open method is called.  
To set it, run the apply method after changing this property.



Set the autoConnect property to NO when specifies the PIN code mode.

`@property(nonatomic, retain) NSString *pinCode`

Specifies the PIN code of the Bluetooth interface.  
It can not acquire the current setting.  
Set to nil when the PIN code is not changed.

***Valid characters:***        0-9, a-z, A-Z

## ●Method

### initWithPortName

```
-(id) initWithPortName: (NSString *) portName deviceType: (SMDeviceType) deviceType
```

This method is used to create an instance of SMBluetoothManager.

#### Parameters:

- portName            - the port name of the device to be connected  
                      Ex. "BT:Star Micronic"
- deviceType          - the type of the device to be connected  
                      SMDeviceTypeDesktopPrinter

#### Returns:

It returns Instance of SMBluetoothManager when succeeded.  
It returns nil when failed.

### open

```
-(BOOL) open
```

This method is used to open connection to the Bluetooth printer.

#### Returns:

It returns YES when succeeded and NO when failed.

## apply

-(BOOL) apply

This method is used to apply the property values of deviceName, iOSPortName, autoConnect, security and pinCode.

### Returns:

It returns YES when succeeded and NO when failed.



The values applied with this method are effective after turning the device off and on and pairing again.

## close

-(void) close

This method is used to close communication with the printer.

## loadSetting

-(BOOL) loadSetting

This method is gets the value specified from the star Bluetooth device.

### Returns:

It returns YES when succeeded and NO when failed.

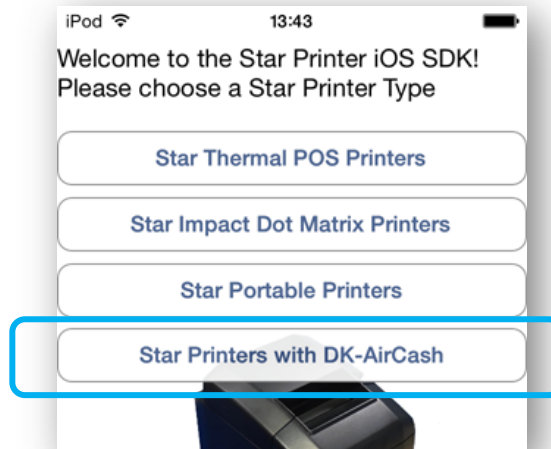
## StarIO iOS SDK Functionality

Overview of this SDK functionality and StarIO Printer Commands

All of these commands can be found in the Star Line Mode Command Manual.

This SDK also has page and section references to the Line Mode Manual for more information so please download and study it if you need more detail on a specific command.

Choosing a Printer and Communication Type



1. Tap "Star Printers with DK-AirCash".



## Supported Samples by Command Type

### Command Samples Include:

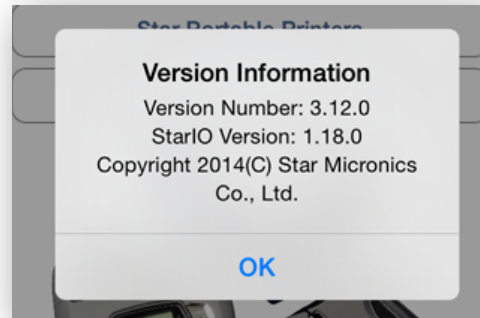
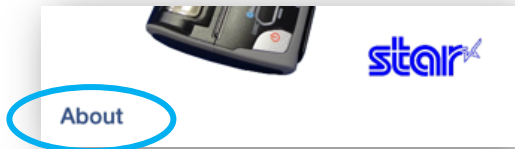
- Get StarIO Version
- Port Discovery
- Get DK-AirCash Firmware Information
- Get DK-AirCash Dip Switch Information
- Get Printer Status
- Get DK-AirCash Status
- Sample Receipt + Open Cash Drawer via DK-AirCash
- Open Cash Drawer via DK-AirCash
- Bluetooth Pairing + Connect
- Bluetooth Disconnect
- DK-AirCash Bluetooth Setting

## Help



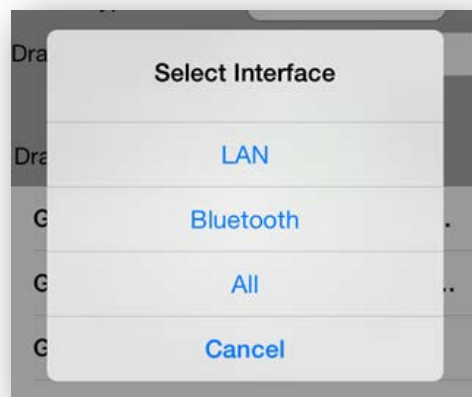
Help produces information on network port settings.

## Get StarIO Version



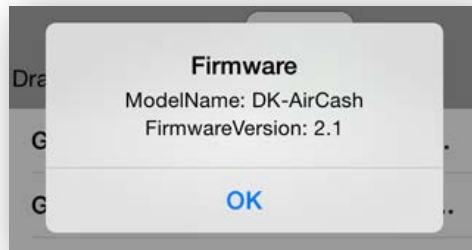
Tap “About”, displays StarIO version.

## Port Discovery



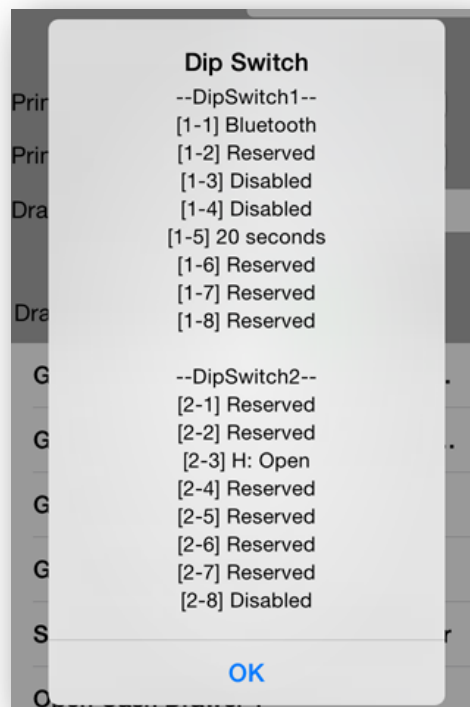
Automatically detects which DK-AirCash are connected to the network. Tap the device to connect to it. [This feature is documented in greater detail here.](#)

## Get Cash Drawer Firmware Information

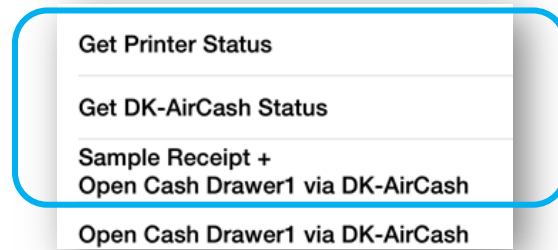


Displays firmware information of the DK-AirCash specified by Drawer Port Name.

## Get DK-AirCash Dip Switch Information

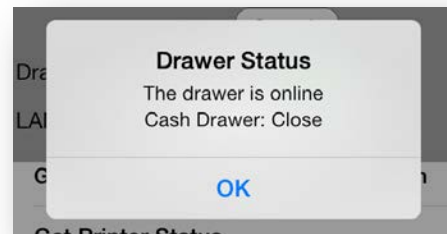
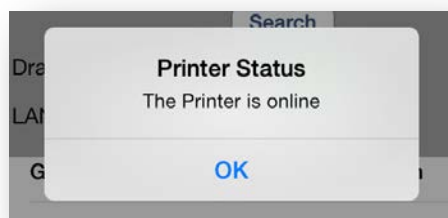


Displays Dip Switch setting information of the DK-AirCash specified by Drawer Port Name.



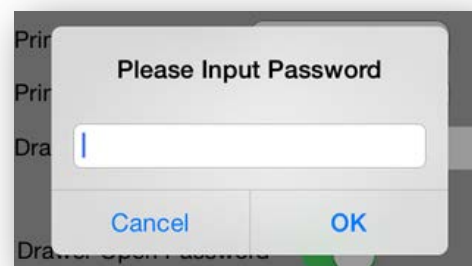
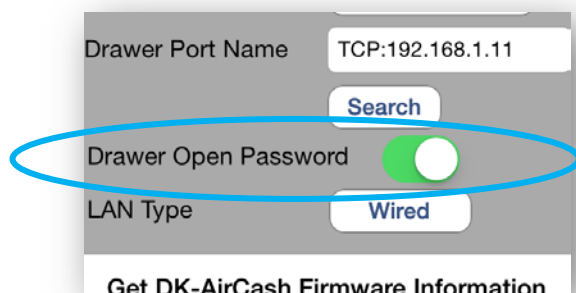
## Get Printer Status / Get Cash Drawer Status

The status of the connected printer or cash drawer is displayed.



## Sample Receipt + Open Cash Drawer

When a printer is connected, it prints a sample receipt and a cash drawer is opened if connected. When Drawer Open Password is set to ON, a password is required to open the cash drawer. The default password is "1234"



When a printer is not connected, a cash drawer does not work.

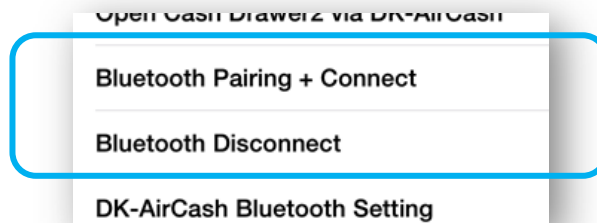


## Open Cash Drawer1 via DK-AirCash / Open Cash Drawer 2 via DK-AirCash

A cash drawer is opened if connected. When Drawer Open Password is set to ON, a password is required to open the cash drawer. The default password is "1234"



You cannot execute Open Cash Drawer1 and Open Cash Drawer2 simultaneously.



## Bluetooth Pairing + Connect

All Bluetooth devices you can connect to are displayed.

You can pair with and connect to a device by tapping it, when the security setting is set to SSP (Default). When a PIN code is used for a secured setting, only the connection function is available. You must pair with a device in advance before using this function.

No StarIO functionality is used.

*Note: iOS6 or later is required for this function.*

## Bluetooth Disconnect

All devices you have paired with and connected to are displayed. You can disconnect a device to allow it to be connected to other host devices.

This function uses the disconnect method.



This function does not support mobile printers.  
(It supports POS printers and the DK-AirCash only.)

## Cash Drawer Bluetooth Setting

iPod 13:14

Device Name DK-AirCash

iOS Port Name DK-AirCash

Auto Connect ☒

Security SSP

Change PIN Code No Change

New PIN Code

Back Apply

All Bluetooth devices you can connect to are displayed.

You can change the Bluetooth interface setting.



The values applied with this method are effective after turning the device off and on and pairing again.

## Tips for App Development when using StarIO

Star Micronics prides itself as the industry leader in great POS products and with great power comes great responsibility. Below is a tips section just to help you get on the fast track to software development with StarIO.

**TIP #1:** If you are going to be coding a large project, create a class to abstract all the printing methods into class(s) instead of having the code reside in the main code block. This will help with code reusability and will also save you time in the long run from having to find one line of code in the main code. By having StarIO only reside in the class(s), you will be fully taking advantage of object oriented programming.

**TIP #2:** Know what the differences and definitions of (ASCII & Unicode), (Hex & Decimal), and (Byte & Char) are. A byte is normally 8-bits long which would be 8 digits of binary (1s and 0s). These bytes are just 8 bits of binary data but bytes can also be int or char. The three different variable types basically hold the data in the same way but there are slight differences. Try to code with Bytes instead of Chars, ints, or strings when choosing a variable to contain your print job data. ASCII to Unicode and vice versa conversions are sometimes unsecure so make sure you know what and how the encoding class works with these. Big mistakes made in Unicode are culture-sensitive search and casing, surrogate pairs, combining characters, and normalization.

**TIP #3:** HEX DUMP MODE! If you are debugging and your application seems to have a bug in it use hex dump mode on the device. This is the best way to verify what is being sent out of the computer is being received correctly. To put the device in hex dump mode, turn the device off, open the cover to the paper, hold the feed button down, turn the device back on, close the cover, let go of the feed button. Hex dump mode is a sure fire way to verify hex data is sent correctly. When in hex dump mode, device functions will not work.

**TIP #4:** Do not waste time trying to reverse engineer StarIO command codes. All the available StarIO commands are available in the Thermal Line Mode Spec Manual and that is the best resource to use when researching a specific StarIO command. This SDK & Manual was built to help you (The Developer) have a very easy job ahead of you to program for Star Devices.

**TIP #5:** If there is a command that is not covered in this SDK but you wish to see a code snippet of that command in use then visit our Developers' section for a possible code block that matches your needs.

**TIP #6:** Looking for an Android printing SDK? Visit our [Developers section](#) to get access to Star developer tools for these environments.

## Additional Resources

This section will share resources that will help you develop good software with StarIO.

Please get the programmers manual for Star Devices from the link below.

### [Star Micronics Developers Network](#)

Browse Star Micronics' FAQs, look up information, etc.

The Developers Network gets you access to:

- Updated Versions of this Manual and Source Code
- Getting Started Advice and Industry Information
- Star Micronics Device Drivers
- Technical Questions/Support

### [Apple Developer Site](#)

The official Apple development resource.

### [Apple Developer Site Resources](#)

Peruse Apple's library of documentation for developers.

### [Unicode.org](#)

The Unicode Consortium - Good place to learn more about Unicode.

### [1D Barcodes](#)

Barcode Island is a great resource for specs on 1D barcodes.

### [2D Barcodes](#)

Great place for information on 2D Barcodes, [QR Codes](#), and [PDF417](#)

### [Code Pages](#)

Learn about Code Pages here.



## ASCII Table Resource

ASCII Hex Symbol	ASCII Hex Symbol	ASCII Hex Symbol	ASCII Hex Symbol
0 0 NUL	16 10 DLE	32 20 (space)	48 30 0
1 1 SOH	17 11 DC1	33 21 !	49 31 1
2 2 STX	18 12 DC2	34 22 "	50 32 2
3 3 ETX	19 13 DC3	35 23 #	51 33 3
4 4 EOT	20 14 DC4	36 24 \$	52 34 4
5 5 ENQ	21 15 NAK	37 25 %	53 35 5
6 6 ACK	22 16 SYN	38 26 &	54 36 6
7 7 BEL	23 17 ETB	39 27 '	55 37 7
8 8 BS	24 18 CAN	40 28 (	56 38 8
9 9 TAB	25 19 EM	41 29 )	57 39 9
10 A LF	26 1A SUB	42 2A *	58 3A :
11 B VT	27 1B ESC	43 2B +	59 3B ;
12 C FF	28 1C FS	44 2C ,	60 3C <
13 D CR	29 1D GS	45 2D -	61 3D =
14 E SO	30 1E RS	46 2E .	62 3E >
15 F SI	31 1F US	47 2F /	63 3F ?

ASCII Hex Symbol	ASCII Hex Symbol	ASCII Hex Symbol	ASCII Hex Symbol
64 40 @	80 50 P	96 60 `	112 70 p
65 41 A	81 51 Q	97 61 a	113 71 q
66 42 B	82 52 R	98 62 b	114 72 r
67 43 C	83 53 S	99 63 c	115 73 s
68 44 D	84 54 T	100 64 d	116 74 t
69 45 E	85 55 U	101 65 e	117 75 u
70 46 F	86 56 V	102 66 f	118 76 v
71 47 G	87 57 W	103 67 g	119 77 w
72 48 H	88 58 X	104 68 h	120 78 x
73 49 I	89 59 Y	105 69 i	121 79 y
74 4A J	90 5A Z	106 6A j	122 7A z
75 4B K	91 5B [	107 6B k	123 7B {
76 4C L	92 5C \	108 6C l	124 7C
77 4D M	93 5D ]	109 6D m	125 7D }
78 4E N	94 5E ^	110 6E n	126 7E ~
79 4F O	95 5F _	111 6F o	127 7F □

Use this to compare hex values to symbol (ASCII) values.

## SDK Package Version History

Release Date	SDK Package Version	Update
Mar. 2016	3.16.0	- Add support devices
Apr. 2015	3.14.0	- iOS 5.x End of support
Mar. 2015	3.13.1	- Deleted a 32-bit separate build settings.
Oct. 2014	3.13.0	- Added Wireless LAN support
Sep. 2014	3.12.0	- Added StarIO Version method, getDipSwitchInformation method
Jul. 2014	3.10.3	-Fixed a problem with multiple execution of getPort method in Bluetooth connection.
Mar. 10 2014	3.10.0	-Added getFirmwareInformation method -Added Open Cash Drawer2 function
Nov. 25 2013	3.9.0	-Added SMBluetoothManager Class -Added endCheckedBlockTimeoutMillis Property -iPad Air, iPad mini(2nd Generation), iPhone5S, iPhone 5C support -iOS 4.3 End of support -Added 64-bit build
Sep. 19 2013	3.8.0	-Added iOS7 support
Aug. 28 2013	3.7.2	-MFi support
Jul. 3 2013	3.7.1	- Initial Release



Star Micronics is a global leader in the manufacturing of small printers. We apply over 50 years of knowhow and innovation to provide elite printing solutions that are rich in stellar reliability and industry-respected features. Offering a diverse line of Thermal, Hybrid, Mobile, Kiosk and Impact Dot Matrix printers, we are obsessed with exceeding the demands of our valued customers every day.

We have a long history of implementations into Retail, Point of Sale, Hospitality, Restaurants and Kitchens, Kiosks and Digital Signage, Gaming and Lottery, ATMs, Ticketing, Labeling, Salons and Spas, Banking and Credit Unions, Medical, Law Enforcement, Payment Processing, and more!

High Quality POS Receipts, Interactive Coupons with Triggers, Logo Printing for Branding, Advanced Drivers for Windows, Mac and Linux, Complete SDK Packages, Android, iOS, Blackberry Printing Support, OPOS, JavaPOS, POS for .NET, Eco-Friendly Paper and Power Savings with Reporting Utility, ENERGY STAR, MSR Reading, *future*PRNT, StarPRNT... How can Star help you fulfill the needs of your application?

Don't just settle on hardware that won't work as hard as you do. Demand everything from your printer. Demand a Star!

### Star Micronics Worldwide

Star Micronics Co., Ltd.  
536 Nanatsushinya  
Shimizu-ku, Shizuoka 424-0066  
Japan  
+81-54-347-2163  
<http://www.star-m.jp/eng/index.htm>

Star Micronics America, Inc.  
65 Clyde Road. Suite G  
Somerset, NJ 08873  
USA  
1-848-216-3300  
<http://www.starmicronics.com>

Star Micronics EMEA  
Star House  
Peregrine Business Park, Gomm Road  
High Wycombe, Buckinghamshire HP13 7DL  
UK  
+44-(0)-1494-471111  
<http://www.star-emea.com>

Star Micronics Southeast Asia Co., Ltd.  
Room 2902C. 29th Fl. United Center Bldg.  
323 Silom Road, Silom Bangrak, Bangkok 10500  
Thailand  
+66-2-631-1161 x 2  
<http://www.starmicronics.co.th/>