Android™ OPN2002i / OPN3002i

Series SDK Users Guide

OPN2002i OPN3002i

OPN2002i / OPN3002i Series

All information subject to change without notice.

Document History

Model Number:	OPN2002 Series / OPN3002 Series	**Specification Number:**	SI12026
Edition:	1.2	**Original Spec Number:**	
Date:	2/8/2013		

Limited Warranty and Disclaimers
PLEASE READ THIS MANUAL CAREFULLY BEFORE INSTALLING OR USING THE PRODUCT.

Serial Number
A serial number appears on all Opticon products. This official registration number is directly related to the device purchased. Do not remove the serial number from your Opticon device. Removing the serial number voids the warranty.

Warranty
Unless otherwise agreed in a written contract, all Opticon products are warranted against defects in materials and workmanship for two years after purchase. Opticon will repair or, at its option, replace products that are defective in materials or workmanship with proper use during the warranty period. Opticon is not liable for damages caused by modifications made by a customer. In such cases, standard repair charges will apply. If a product is returned under warranty and no defect is found, standard repair charges will apply. Opticon assumes no liability for any direct, indirect, consequential or incidental damages arising out of use or inability to use both the hardware and software, even if Opticon has been informed about the possibility of such damages.

Packaging
The packing materials are recyclable. We recommend that you save all packing material to use should you need to transport your scanner or send it for service. Damage caused by improper packaging during shipment is not covered by the warranty.

Trademarks
Trademarks used are the property of their respective owners.

Opticon Inc. and Opticon Sensors Europe B.V. are wholly owned subsidiaries of OPTOELECTRONICS Co., Ltd., 12-17, Tsukagoshi 4-chome, Warabi-shi, Saitama, Japan 335-0002. TEL +81-(0) 48-446-1183; FAX +81-(0) 48-446-1184

SUPPORT

USA

Phone: 800-636-0090
Email: support@opticonusa.com
Web: www.opticonusa.com

Europe

Email: support@opticon.com
Web: www.opticon.com

Revision Record

DOC_ID: SI12026

Version: 1.2

Model Number: OPN2002i Series/OPN3002i Series

Version	Date	Description of Changes	Content
1.0	8/10/2012	—	New document
1.1	10/22/2012	1, 2	Added paragraph numbers
		1.1	Deleted supported Android versions 2.2-2.3.2
		1.2	Changed “opn-spp.jar” to “bluetoothservice.jar”
		2.4	Changed the device search for ”OPN2002i” from all matches to the top match
		2.6.1	Added Edit Processing for scanned data
		2.6.3	Added enableAckNack () to the Completing Connection process
1.2	2/8/2013	Cover	Added the OPN3002i Series
		3.1-3.4	Added OPN3002i Specific Settings items

Contents

Forward

This document is primarily used for aiding the development of Android applications that utilize the OPN2002i and OPN3002i data collectors. For details regarding device use or technical information necessary for Android development, please refer to reference documents that cover those subjects.

All details contained within this document pertain to both the OPN2002i and OPN3002i, excluding OPN3002i specific settings, which apply only to the OPN3002i.

Explicit details regarding the API within this SDK are not contained in this document. Please refer to "OPNBluetoothKit API Reference" for more information, or the API calls, parameters, etc.

1. Development Environment

1.1 Execution Environment

Android applications developed with this SDK are created for the OS versions below:

OS	API Level
Android 4.0.3	15
Android 4.0-4.0.2	14
Android 3.2	13
Android 3.1	12
Android 3.0	11
Android 2.3.3-2.3.7	10

1.2 Development Environment Procedure for Using the SDK

It is important to do the following when using the SDK:

* Add bluetoothservice.jar to your Android application project.

* Enable the use of Bluetooth® via AndroidManifest.xml.

This guide operates under the following assumptions:

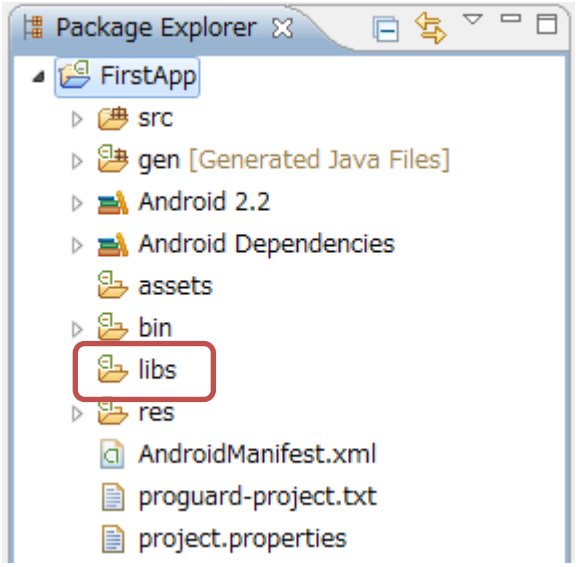
- The Eclipse development environment is up to date and equipped to build Android applications.

- The projects to utilize this SDK are already in a state of development.

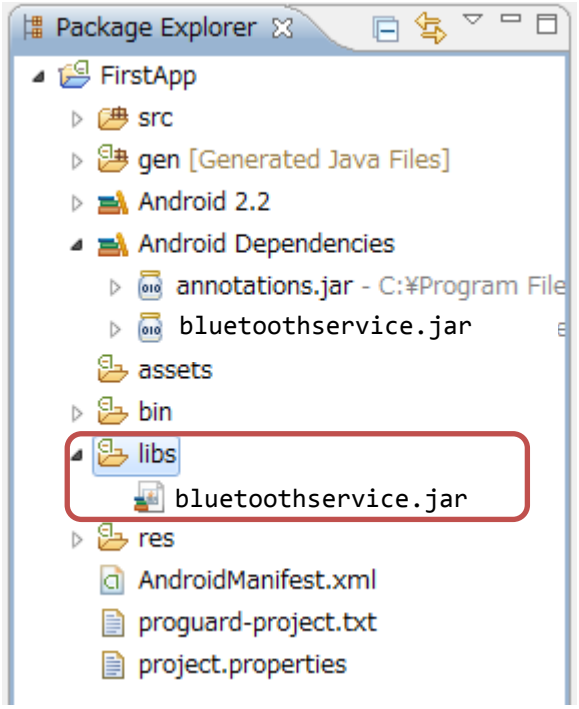
This document explains the procedure for integrating and using the SDK, beginning from the project "FirstApp" as an example.

1.2.1 Adding the "bluetoothservice.jar" file to the Project

① First, create the libs folder inside of the project.



② Copy the “bluetoothservice.jar” file into the libs folder.



→ [Android Dependencies] will be added automatically.

1.2.2 Enabling Bluetooth via “AndroidManifest.xml”

① Add Bluetooth-enabling settings via AndroidManifest.xml.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest ・・・・>

    ・・・・

    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN"
/>
    <uses-permission android:name="android.permission.BLUETOOTH" />

    ・・・・
```

2. API Usage

The sample application FirstApp (which comes with the SDK) demonstrates how to use the API, along with the following section.

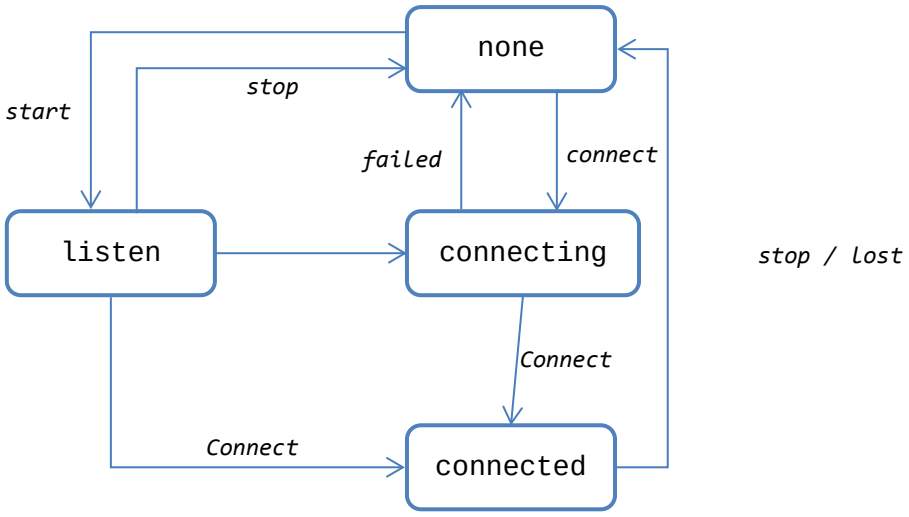
2.1 Creating/Returning the Opn2002BluetoothService

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    mBTService = Opn2002BluetoothService.getInstance();
```

All Bluetooth transmissions to the data collector are performed via the OPN2002Bluetooth Service Class. In this sample, Opn2002BluetoothService-class instances are created and returned within onCreate, then stored in the field.

2.2 Opn2002BluetoothService Status



Opn2002BluetoothService has four different status types, all of which perform transitions similarly to the above. The characteristics for each are described in the following table.

None	SPP Connection is not being conducted.
Connecting	Android is set to Master Mode and attempts SPP connection to the data collector.
Listen	Android is set to Slave Mode and awaits a connection attempt from the data collector.
Connected	SPP Connection is currently implemented. Receiving data is possible.

States are returned via calling the Opn2002BluetoothService#getState () method. The return value of this method is a BluetoothServiceState-class (Enum-type) object.

2.3 Android-side awaiting connection attempt from the Data Collector

```
@Override
protected void onStart() {
    super.onStart();

    if(mBTService != null){

        . . . . .

        if(mBTService.getState() == BluetoothServiceState.none){
            mBTService.start();
        }
    }
}
```

Call the Opn2002BluetoothServce#start () method to switch to “waiting for data collector connection” status (BluetoothServiceState.listen). The connected () method is called once the request is received and connection confirmed. Once connection is established, the process will cease functioning automatically (waiting for connection process). This method is implemented when the data collector is set to Master Mode. As shown in the next section "Connecting to the data collector from the Android-side", it is possible to connect via pressing and holding the function key even while in Master Mode.

2.4 Connecting to the data collector from the Android-side

```
// Press Connect Button
Button btnConn = (Button)findViewById(R.id.button_connect);
btnConn.setOnClickListener(new OnClickListener() {
    public void onClick(View v) {
       mBTService.stop();  //Stops Listen Processing
        //One paired device with "OPN" in its name is returned and connected.
       Set<BluetoothDevice> set = mBTService.getPairedDevices();
       Iterator<BluetoothDevice> it = set.iterator();
       if(set.size() > 0) {
           while(it.hasNext()){
              BluetoothDevice device = it.next();
              if(device.getName() != null &&
                   device.getName().indexOf("OPN")>=0){
                   //Begins connecting.
                   mBTService.connect(device, false);
                }
           }
          if (mBTService.getState()!=BluetoothServiceState.connected)
             mBTService.start(); //Listen resumes if connection fails
        }
    }
});
```

Connect to the data collector by calling the Opn2002BluetoothService#connect () method (BluetoothServiceState.connecting status). The connected () method is called when successfully connecting to the device (ref. 2.6.3). Alternatively, the connectFailed () method is called if the connection is unsuccessful (ref. 2.6.4).

In this sample, when the Connect button is pressed, one paired device (with "OPN" in the title) is returned and begins connection testing. If there is more than one device with "OPN" in the title, please make changes so that the desired device is selected. This method is called by holding the function key to connect when the data collector set to Slave Mode.

2.5 Disconnecting

```
// Press Disconnect Button
Button btnDisconn = (Button)findViewById(R.id.button_disconnect);
btnDisconn.setOnClickListener(new OnClickListener() {
    public void onClick(View v) {
       mBTService.stop();
    }
});
```

Disconnect from the data collector by calling the Opn2002BluetoothService#stop () method (BluetoothServiceState.connecting status). In this sample, the procedure begins once the Disconnect button is pressed.

2.6 Processing Events created via Communication with the Data Collector

```
public class FirstAppActivity extends Activity
implements IBluetoothObserver {

    . . . . .
}
```

Events such as "Receive Data", "Disconnect", "Connection Completed", and "Connect Failure" are created by transmitting between the Opn2002BluetoothService and the data collector. In order to receive and process these events, first the BluetoothObserver interface must be called via the Event Processing-class. In this sample, the FirstAppActivy-class calls the necessary interface.

```
@Override
protected void onResume() {
    super.onResume();

    if(mBTService != null){
        mBTService.addObserver(this);
    }
    refreshStatusLabel();
}
```

Next, the Opn2002BluetoothService#addObserver () method is conducted, with the object (Observer) called with BluetoothObserver-class as its argument. Thus, the interface-implemented method will be called whenever an event is created. In this sample, Observer is registered within onResume.

```
@Override
protected void onPause(){
    super.onPause();

    if(mBTService != null){
            mBTService.removeObserver(this);
    }
}
```

An Observer registered via Opn2002BluetoothService#addObserver () method may have that registration removed by calling a removeObserver () method of the same class. Within this sample, Observer is registered within onPause.

2.6.1 Receive: Receiving Data

```
@Override
public void receive(String data) {
    if(mTextView != null && data != null){
       /* Replaces the line feed code.   */
       data = data.replace((char)0x0D, (char)0x0A);
       /* Deleted if the command's ACK/NAK is entered.   */
       data = data.replaceAll(""+(char)0x6, "");
       data = data.replaceAll(""+(char)0x15, "");
       mTextView.append(data);
    }
}
```

This processes the "Receive Data" event created when a barcode is scanned by the data collector. This data is then passed on as a string-type argument. Because the line feed code is 0xD by default, the scanned data is changed to 0xA. Also, if the command is executed both before and afterwards, it will be removed due to the possibility of ACK/NAK connecting. In this sample, content of string-type argument data is added to TextView.

2.6.2 ConnectionLost: Disconnecting

```
@Override
public void connectionLost() {
    Toast.makeText(this, "connection lost", Toast.LENGTH_LONG).show();
    refreshStatusLabel();
}
```

This processes the "Disconnect" event created when connection is severed from the data collector. In this sample, Toast is implemented to inform the user of the event. Also, the connection status display is updated via the refreshStatusLabes () method.

2.6.3 Connected: Completing Connection

```
@Override
public void connected(BluetoothDevice arg0) {
    Toast.makeText(this, "connected", Toast.LENGTH_LONG).show();
    mBTService.enableAckNak();
    // Caution!  When using Opn2002BluetoothService, be sure to call
enableAckNak () when connecting.
    refreshStatusLabel();
}
```

This processes the "Completed Connection" event that is created when connecting to the data collector. In this sample, Toast is utilized to inform the user of completion. The connection status display is updated via the refreshStatusLabes () method.

This SDK receives the ACK/NAK to control transmissions as a command response. Because command response is the OPN default, when connecting it is important to call enableAckNak() to enable command response functionality.

2.6.4 ConnectFailed: Connection Failure

```
@Override
public void connectFailed() {
    Toast.makeText(this, "connection failed ", Toast.LENGTH_LONG).show();
}
```

Processes the event created when connection to the data collector has failed. Toast is used in this sample to inform the user of the failure.

2.7 Running API Standard Commands

```
// Press Version Button
Button btnCommand = (Button)findViewById(R.id.button_command);
btnCommand.setOnClickListener(new OnClickListener() {
    public void onClick(View v) {
       mBTService.getFirmwareVersion(new GetFirmwareVersionCallback() {
           @Override
           protected void callbackExecute(String arg0) {
              receive(arg0);      //Displays results on the screen.
           }
       });
    }
});
```

Among commands related to the data collector, there are several provided as Opn2002iBluetoothService-class methods. The format regularly used for issuing the above commands is shown below:

public void Method Name(Argument 1, Argument 2, …, Callback)

Because this method is called asynchronously, it is necessary to specify the callback retrieved when asynchronous processing is completed. Each callback is an object for an IBluetoothCallback-outfitted class, and the callbackExecute () method is called when asynchronous processing is completed. In this sample, Opn2002BluetoothService's getFirmwareVersion () method is called when the Version button is pressed. The callback specifies IBluetoothCallback-inputted GetFirmwareVersionCallback class's anonymous class and the callbackExecute () method is called when processing is completed.

2.8 Running Other Commands

```
// Buzzer ON
Button btnBuzzerOn = (Button)findViewById(R.id.button_buzzer_on);
btnBuzzerOn.setOnClickListener(new OnClickListener() {
    public void onClick(View v) {
       mBTService.write("W8", new VoidCallback() {

           @Override
           protected void callbackExecute(Void arg0) {
              if(getIsError()){
                     showErrorMessage(); //Displays error message.
              }
           }
       });
    }
});
```

As an Opn2002BluetoothService-class method, the Opn2002BluetoothService#write () method must be called in order to conduct methods that have not been provided by this SDK. These are called asynchronously via specifying the callback and command string to be implemented into the write () method argument. The callback has the same thinking process as API Standard Commands. In this sample, the Opn2002BluetoothService-class's write () method is called when the “Buzzer ON” button is pressed. The callback specifies the IBluetoothCallback-inputted VoidCallback class's anonymous class and callbackExecute () method is called when processing is completed. For further details regarding usable commands, please refer to "3. Command References".

"3.5 Character String Setting" commands are used to input character strings set via PIN(]PINS), Connection Partner Address(]BDAS), Device Name([E65), Prefix, Suffix, or Error Message(TH, TI).

* Example 1: Setting commands when the device name is "OPN-2002"
  “[E650O0P0N5NQ2Q0Q0Q2[E66”
* Example 2: Setting commands when setting UPC-A's prefix as "UPCA"
  “N10U0P0C0A”

3. Command References

3.1 Decoder Settings

Setting Item	Parameters	Setting Value	Command
UPC-A/E Scanning	Enable	1	R1
	Disable	0	[X4B
UPC-A/E Addon2 Scanning	Enable	1	R2
	Disable	0	[X4C
UPC-A/E Addon5 Scanning	Enable	1	R3
	Disable	0	[X4D
JAN/EAN-13/8 Scanning	Enable	1	R4
	Disable	0	[X4E
JAN/EAN-13/8 Addon2 Scanning	Enable	1	R5
	Disable	0	[X4F
JAN/EAN-13/8 Addon5 Scanning	Enable	1	R6
	Disable	0	[X4G
Code 39 Scanning	Enable	1	B2
	Disable	0	VB
Tri-Optic Scanning	Enable	1	JZ
	Disable	0	[DDJ
NW-7 Scanning	Enable	1	B3
	Disable	0	VC
Industrial 2 of 5 Scanning	Enable	1	R7
	Disable	0	V7
Interleaved 2 of 5 Scanning	Enable	1	R8
	Disable	0	V8
S-Code Scanning	Enable	1	R9
	Disable	0	[DDK
Matrix 2 of 5 Scanning	Enable	1	BB
	Disable	0	[DDL
Code 93 Scanning	Enable	1	B5
	Disable	0	VD
Code 128 Scanning	Enable	1	B6
	Disable	0	VE
EAN-128 Scanning	Enable if Possible	2	OG
	Enable EAN-128 Only	1	JF
	Disable Code 128 as Output	0	OF
MSI/Plessey Scanning	Enable	1	B7
	Disable	0	VF
IATA Scanning	Enable	1	B4
	Disable	0	VH
UK/Plessey Scanning	Enable	1	B1
	Disable	0	VA
Telepen Scanning	Enable	1	B9
	Disable	0	VG
GS1 DataBar (RSS-14)	Enable	1	JX
	Disable	0	SJ
GS1 DataBar	Enable	1	JY

Limited (RSS-Limited)	Disable	0	SK
GS1 DataBar Expanded (RSS-Expanded)	Enable	1	DR
	Disable	0	SL
PDF-417 Scanning	Enable	1	[BCF
	Disable	0	[BCR
MicroPDF-417 Scanning	Enable	1	[BCG
	Disable	0	[BCS
Code 11 Scanning	Enable	1	[BLC
	Disable	0	[BLA
Code 3 of 5 Scanning	Enable	1	WH
	Disable	0	WI
UPC-A Format (Transmit CD)	13-digit (Leading Zero and Transmit CD)	b2=1, b4=1	E2
	12-digit (No Leading Zero)	b2=1, b4=0	E3
	12-digit (Do Not Transmit CD)	b2=0, b4=1	E4
	11-digit (No Leading Zero and Do Not Transmit CD)	b2=0, b4=0	E5
UPC-E Format (Transmit CD)	8-digit (Leading Zero and Transmit CD)	b2=1, b4=1	E6
	7-digit (No Leading Zero)	b2=1, b4=0	E7
	7-digit (Not Transmit CD)	b2=0, b4=1	E8
	6-digit (No Leading Zero and Do Not Transmit CD)	b2=0, b4=0	E9
JAN/EAN-13 Format (Transmit CD)	Transmit CD	1	6K
	Do Not Transmit CD	0	6J
JAN/EAN-8 Format (Transmit CD)	Transmit CD	1	6I
	Do Not Transmit CD	0	6H
Transmit Code 39 CD	Transmit CD	1	D9
	Do Not Transmit CD	0	D8
Transmit NW-7 CD	Transmit CD	1	H8
	Do Not Transmit CD	0	H9
Transmit Industrial 2 of 5/ Interleaved 2 of 5	Transmit CD	1	E0
	Do not transmit CD	0	E1
Transmit MSI/Plessey CD	CD Transmit CD1	1	4E
	CD Transmit CD1 and CD2	2	4F
	Do Not Transmit CD	0	4G
Transmit IATA CD	Transmit CD	1	4L
	Do Not Transmit CD	0	4M
Transmit GS1 DataBar Family CD	Transmit CD	1	DL
	Do Not Transmit CD	0	DM
Calculate WPC (UPC/EAN/JAN) CD	Calculate CD	1	[XEE
	Do Not Calculate CD	0	[XEF
Calculate Code 39 CD	Calculate CD	1	C0
	Do Not Calculate CD	0	C1
Calculate NW-7 CD	Calculate CD Mod10/W1, 2 spec1	1	[XF8

	Calculate CD Mod16	2	H6
	Calculate CD 7 check	3	[XFB
	Calculate CD Mod11	4	[XFC
	Do Not Calculate CD	0	H7
Calculate Industrial 2 of 5/ Interleaved 2 of 5	Calculate CD	1	G1
	Do Not Calculate CD	0	G0
Calculate Code 93 CD	Calculate CD	1	AC
	Do Not Calculate CD	0	9Q
Calculate Code 128/EAN-128 CD	Calculate CD	1	ME
	Do Not Calculate CD	0	MF
Calculate MSI/Plessey CD	Calculate CD CD1 Only (Mod10)	1	4B
	Calculate CD (Mod10/Mod10)	2	4C
	Calculate CD (Mod10/Mod11)	3	4D
	Calculate CD (Mod11/Mod10)	4	4R
	Do Not Calculate CD	0	4A
Calculate IATA CD	Calculate CD (CPN+FORM SERIAL)	1	4J
	Calculate CD (FORM SERIAL)	2	4I
	Calculate CD (ALL DATA)	3	4K
	Do Not Calculate CD	0	4H
Transmit Code 39 Start Stop	Transmit	1	D0
	Do not transmit	0	D1
Transmit NW-7 Start Stop	Transmit ABCD/TN*E	1	F1
	Transmit abcd/tn*e	2	F2
	Transmit ABCD	3	F3
	Transmit abcd	4	F4
	Transmit DC1DC2DC3DC4	5	HJ
	Do Not Transmit	0	F0
Error Message Upon Scanning Failure	Do Not Transmit	0	TG
	Transmit	9	TH,TI
Setting for the Character String Transmitted When Scanning Fails (Cannot Find Label)	char x 4-digits		TH
Setting for the Character String Transmitted When Scanning Fails (Decode Failed)	char x 4-digits		TI
OPN3002i Specific Settings			
Intelligent Mail Scanning	Enable	1	[D5F
	Disable	0	[D5G
Postnet Scanning	Enable	1	[D6A
	Disable	0	[D6B
Japanese Postal Scanning	Enable	1	[D5P
	Disable	0	[D5Q
CodablockF Scanning	Enable	1	[D4P
	Disable	0	[D4Q
Data Matrix (ECC200) Scanning	Enable	1	[BCC
	Disable	0	[BCO
Data Matrix (ECC000-140) Scanning	Enable	1	[BG0
	Disable	0	[BG1

Aztec Code Scanning	Enable	1	[BCH
	Disable	0	[BCT
Aztec Runes Scanning	Enable	1	[BF2
	Disable	0	[BF3
Chinese Sensible Code Scanning	Enable	1	[D4L
	Disable	0	[D4M
QR code Scanning	Enable	1	[BCD
	Disable	0	[BCP
MicroQR Scanning	Enable	1	[D2U
	Disable	0	[D2V
Maxi Code Scanning	Enable	1	[BCE
	Disable	0	[BCQ
Composite on GS1Databar Scanning	Enable	1	[BHE
	Disable	0	[BHF
Composite on UPC/EAN Scanning	Enable	1	[D1V
	Disable	0	[D1W

3.2 Scanner/Device Settings

Scanning Modes	Single Read	1	S0
	Multiple Read	2	S1
	Successive Read	3	S2
Scan Duration	Unlimited	0	YM
	0 sec	-1	Y0
	1 sec	50	Y1
	2 sec	100	Y2
	3 sec	150	Y3
	4 sec	200	Y4
	5 sec	250	Y5
	6 sec	300	Y6
	7 sec	350	Y7
	8 sec	400	Y8
	9 sec	450	Y9
	Scan Duration x10	-	YL
Number of Checks	1 Scan 0 Checks	0	X0
	2 Scan 1 Checks	1	X1
	3 Scan 2 Checks	2	X2
	4 Scan 3 Checks	3	X3
	5 Scan 4 Checks	4	BS
	6 Scan 5 Checks	5	BT
	7 Scan 6 Checks	6	BU
	8 Scan 7 Checks	7	BV
	9 Scan 8 Checks	8	BW
	10 Scan 9 Checks	9	[XBT
	11 Scan 10 Checks	10	[XBU
	12 Scan 11 Checks	11	[XBV
	13 Scan 12 Checks	12	[XBW

	14 Scan 13 Checks	13	[XBX
	15 Scan 14 Checks	14	[XBY
	16 Scan 15 Checks	15	[XBZ
Multiple Read Reset Timer	Unlimited	0	AG
	50ms	3	AH
	100ms	5	AI
	200ms	10	AJ
	300ms	15	AK
	400ms	20	AL
	500ms	25	AM
	600ms	30	AN
Add On Timer	Nothing	0	XA
	250ms	13	XB
	500ms	25	XC
	750ms	38	XD
Buzzer Volume	Maximum	127	T0
	High	32	T1
	Medium	8	T2
	Small	1	T3
LED Lighting Duration	Disable	0	T4
	200ms	10	T5
	400ms	20	T6
	600ms	30	T7
Trigger Mode	Disable Trigger	0	S7
	Enable Trigger	1	S8
Trigger Repeat	Disable	0	/K
	Enable	1	/M
Buzzer Sound	Disable	0	W0
	Enable	1	W8
Buzzer Tone	Single Tone	0	W1
	High	1	W2
	Low	2	W3
	4.5KHz	3	[XTS
	2.2KHz-2KHz	4	[X%Q
Buzzer Vibration Length	100ms	5	W4
	200ms	10	W5
	400ms	20	W6
	50ms	2	W7
Buzzer Vibration Timing	Pre-Transmit Buzzer	0	VY
	Post-Transmit Buzzer	1	VZ
Connected Partner Address		char×12\0	
	Start Address Settings	-	]BDAS
	Finish Address Settings	-	]BDAE
Authentication	Do Not Authenticate	0	]AUTD
	Authenticate (Every Time)	1	]AUTE
	Authenticate (Paired When Not Supported)	7	]AUTO

Encryption	Disable	0	]ENCD
	Enable	1	]ENCE
Command Response	YES (ACK/NAK)	1	WC
	Adheres to ACK/NAK Control Settings	0	WD
PIN Code		char x 16¥0	
	Start PIN Settings	-	]PINS
	Finish PIN Settings	-	]PINE
Trigger Connection	Disable	0	]TSCD
	Enable	1	]TSCE
Enable/Disable Connection Processing via Address Barcodes	Disable	0	]DIAU
	Enable	1	]ENAU
ACK/NAK Controls	N/A	0	[XP5
	YES	1	P3
	YES (No Response)	2	P4
Connection Mode	SPP Master	0	]BCMA
	SPP Slave	1	]BCSA
	HID	2	[C02
Slave Connection Wait Time	30 sec	1500	]SWT0
	1 min	3000	]SWT1
	2 min	6000	]SWT2
	3 min	9000	]SWT3
	4 min	12000	]SWT4
Effective Auto Reconnect Time	Disable	0	]CA00
	1 min	3000	]CA01
	2 min	6000	]CA02
	3 min	9000	]CA03
	4 min	12000	]CA04
	5 min	15000	]CA05
	6 min	18000	]CA06
	7 min	21000	]CA07
	8 min	24000	]CA08
	9 min	27000	]CA09
	10 min	30000	]CA10
	11 min	33000	]CA11
	12 min	36000	]CA12
	13 min	39000	]CA13
	14 min	42000	]CA14
	15 min	45000	]CA15
Auto Disconnect Time	Disable	0	]AD00
	10 min	30000	]AD01
	20 min	60000	]AD02
	30 min	90000	]AD03
	40 min	120000	]AD04
	50 min	150000	]AD05
	60 min	180000	]AD06
	1 min	3000	]ADM1
	2 min	6000	]ADM2

	3 min	9000	]ADM3
	4 min	12000	]ADM4
	5 min	15000	]ADM5
	6 min	18000	]ADM6
	7 min	21000	]ADM7
	8 min	24000	]ADM8
	9 min	27000	]ADM9
	10 sec	500	]ADS1
	20 sec	1000	]ADS2
	30 sec	1500	]ADS3
	40 sec	2000	]ADS4
	50 sec	2500	]ADS5
Trigger Connect Press Time	Disable Trigger Connect	0	]PC00
	1 sec	50	]PC01
	2 sec	100	]PC02
	3 sec	150	]PC03
	4 sec	200	]PC04
	5 sec	250	]PC05
	6 sec	300	]PC06
	7 sec	350	]PC07
	8 sec	400	]PC08
	9 sec	450	]PC09
Trigger Disconnect Press Time	Disable Trigger Disconnect	0	]PD00
	1 sec	50	]PD01
	2 sec	100	]PD02
	3 sec	150	]PD03
	4 sec	200	]PD04
	5 sec	250	]PD05
	6 sec	300	]PD06
	7 sec	350	]PD07
	8 sec	400	]PD08
	9 sec	450	]PD09
ACK/NAK Wait Time	100ms	5	[XI5
	500ms	25	[XI6
	1s	50	[XI7
Outside-Range Memory	Disable	0	]DTMD
	Enable	1	]DTME
Data Collect	Disable	0	[BM0
	Enable	1	[BM1
Barcode Read Auto Connect	Disable	0	]ARCD
	Enable	1	]ARCE
Data Collector Disconnect Buzzer	Disable	0	]DSSD
	Enable	1	]DSSE
Connection Partner Disconnect Buzzer	Disable	0	]DSPD
	Enable	1	]DSPE
Memory Data Output Method	Outputs Immediately When Connecting	0	[EBB
	Outputted via Function Key	1	[EBC

	Press or Commands		
	Data Output Commands		[EBD
COM Communication when connected to USB	Disable	0	[C10
	Enable	1	[C11
Function Press Output	Start Settings		[C23
	HT	0x09	[$09
	LF	0x0A	[$0A
	CR	0x0D	[$0D
	CAN	0x18	[$18
	ESC	0x1B	[$1B
	Do Not Output	0xFF	[$FF
	iPhone Soft Keyboard Display	0xA6	[$A6
	ENTER	0xB2	[$B2
Bluetooth Device Name	Start Device Settings	-	[E65
	Finish Device Settings	-	[E66
Previous Slave Connection Partner Address (12 characters + NUL)			
OPN3002i Specific Settings			
Good Read Vibrate	Enable	1	[EBI
	Disable	0XFF	[EBH

3.3 Prefix

UPC-A Prefix			N1
UPC-A Add On Prefix			M0
UPC-E Prefix			N2
UPC-A Add On Prefix			M1
JAN/EAN-13 Prefix			N3
JAN/EAN-13 Add On Prefix			M2
JAN/EAN-8 Prefix			N4
JAN/EAN-8 Add On Prefix			M3
Code 39 Prefix			M4
Tri-Optic Prefix			MC
NW-7 Prefix			M5
D. 2 of 5 Prefix			M6
I. 2 of 5 Prefix			M7
S Code Prefix			MB
Matrix 2 of 5 Prefix			GL
Code 93 Prefix			M8
Code 128 Prefix			M9
MSI/Plessey Prefix			N0 (zero)
IATA Prefix			I8
UK/Plessey Prefix			MA
Telepen Prefix			L8
RSS Prefix			OE
PDF-417 Prefix			OC

Micro PDF-417 Prefix			OD
Code 11 Prefix			[BLD
Code 3 of 5 Prefix			*$
EAN-128 Prefix			[XMX
Common Prefix Settings			MZ
All Code Prefix Settings			RY
OPN3002i Specific Settings			
Intelligent Mail Prefix			[D5I
Postnet Prefix			[D6D
Japanese Postal Prefix			[D5S
CodablockF Prefix			[D4S
Data Matrix (ECC200, ECC000-140) Prefix			MD
Aztec Code/Aztec Runes Prefix			[BF0
Chinese Sensible Code Prefix			[D4N
QR/MicroQR Prefix			MK
Maxi Code Prefix			ML
Composite (on GS1Databar, UPC/EAN) Prefix			RR

3.4 Suffix

UPC-A Suffix			N6
UPC-A Add On Suffix			O0 (O Zero)
UPC-E Suffix			N7
UPC-A Add On Suffix			O1
JAN/EAN-13 Suffix			N8
JAN/EAN-13 Add On Suffix			O2
JAN/EAN-8 Suffix			N9
JAN/EAN-8 Add On Suffix			O3
Code-39 Suffix			O4
Tri-Optic Suffix			PN
NW-7 Suffix			O5
D. 2 of 5 Suffix			O6
I. 2 of 5 Suffix			O7
S Code Suffix			OB
Matrix 2 of 5 Suffix			GM
Code 93 Suffix			O8
Code-128 Suffix			O9
MSI/Plessey Suffix			N5
IATA Suffix			I9
UK/Plessey Suffix			OA
Telepen Suffix			L9
RSS Suffix			PQ
PDF-417 Suffix			PY
Micro PDF-417 Suffix			PZ
Code 11 Suffix			[BLE

Code 3 of 5 Suffix			*%
EAN-128 Suffix			[XOX
Common Suffix			PS
All Code Suffix Settings			RZ
OPN3002i Specific Settings			
Intelligent Mail Suffix			[D5J
Postnet Suffix			[D6E
Japanese Postal Suffix			[D5T
CodablockF Suffix			[D4T
Data Matrix (ECC200, ECC000-140) Suffix			PO
Aztec Code/Aztec Runes Suffix			[BF1
Chinese Sensible Code Suffix			[D4O
QR/MicroQR Suffix			PW
Maxi Code Suffix			PX
Composite (on GS1Databar, UPC/EAN) Suffix			RS

3.5 Character String Settings

Setting Characters	Command
0	Q0
1	Q1
2	Q2
3	Q3
4	Q4
5	Q5
6	Q6
7	Q7
8	Q8
9	Q9
A	0A
B	0B
C	0C
D	0D
E	0E
F	0F
G	0G
0H	0H
0I	0I
J	0J
K	0K
L	0L
M	0M
N	0N
O	0O

P	0P
Q	0Q
R	0R
S	0S
T	0T
U	0U
V	0V
W	0W
X	0X
Y	0Y
Z	0Z
a	$A
b	$B
c	$C
d	$D
e	$E
f	$F
g	$G
h	$H
i	$I
j	$J
k	$K
l	$L
m	$M
n	$N
o	$O
p	$P
q	$Q
r	$R
s	$S
t	$T
u	$U
v	$V
w	$W
x	$X
y	$Y
z	$Z
<SPACE>	5A
!	5B
"	5C
#	5D
$	5E
%	5F
&	5G
	5H
(	5I
)	5J
*	5K

+	5L
,	5M
-	5N
.	5O
/	5P
:	6A
;	6B
<	6C
=	6D
>	6E
?	6F
@	6G
[	7A
\	7B
]	7C
^	7D
_	7E
`	7F
{	9T
|	9U
}	9V
~	9W
^@ (NULL)	9G
^A (SOH)	1A
^B (STX)	1B
^C (ETX)	1C
^D (EOT)	1D
^E (ENQ)	1E
^F (ACK)	1F
^G (BEL)	1G
^H (BS)	1H
^I (HT)	1I
^J (LF)	1J
^K (VT)	1K
^L (FF)	1L
^M (CR)	1M
^N (SO)	1N
^O (SI)	1O
^P (DLE)	1P
^Q (DC1)	1Q
^R (DC2)	1R
^S (DC3)	1S
^T (DC4)	1T
^U (NAK)	1U
^V (SYN)	1V
^W (ETB)	1W
^X (CAN)	1X
^Y (EM)	1Y

^Z (SUB)	1Z
^[(ESC)	9A
^＼(FS)	9B
^] (GS)	9C
^^ (RS)	9D
^_ (US)	9E
DEL (ASCII 127)	9F
Year	[$YR
Month	[$MO
Day	[$DY
Hour	[$HR
Minute	[$MI
Second	[$SC
Scan Count	[$CT
Barcode Type	[$BT
Barcode Data Length	[$BL
Battery Voltage	[$BV
BD Address	[$AR
Terminal ID	[$ID
Terminal Name	[$NM

Model Name: OPN2002i Series
OPN3002i Series
Revision Number: 1.2
DOC_ID: SI12026