M3 MOBILE

Android SDK Guide book

Release date: 3, June, 2020

Copyright and Agreement

M3 Mobile does not guarantee the quality and performance of the programs written in unsupported programming language. For supported development tools and languages, please refer to Development Tool and Requirements section.

Requirements

The following software must be installed

Microsoft Microsoft© Windows 10 (32-bit and 64-bit)
Java Development Kit (JDK) v7u7 or higher
Android Developer Tools (ADT) v22.6.0 or higher
Android SDK 4.3.1 (API Version 18)

Official site provides the Android SDK

http://developer.android.com/

Devices Supported

The following device has been used for validation:
SM15 7.1.1 (Nougat, M3 OS ver. 1.2.0), SM15 8.1, UL20 8.1, UL20 9.0, SL10 8.1

	Key Remap	Scanner	LongRange Scanner	UHF RFID Reader
SM15	O	O	O	O
UL20	X	O	X	X
SL10	X	O	X	X

Revision History

Version 0.0.1 Release date. 2018-03-29

- Supported SM15(Android 7.1.1)

Version 0.0.2 Release date. 2018-04-20

- Included UHF RFID Reader SDK (SM15 only)

Version 0.0.3 Release date. 2018-07-03

- Added Intelligent Document Capture

Version 1.0.0 Release date. 2018-10-08

- Added AIDL Scanner SDK
- Added Decode Count Intent Extra
- Added IDC Symbology QR Code Value

Version 1.1.0 Release date. 2019-01-10

- Added Honeywell Scanner SDK

Version 1.1.2 Release date. 2019-06-13

- Added LongRange Scanner SDK

Version 1.1.3 Release date. 2019-07-29

- Added UHF RFID AIDL Callback SDK

Version 1.2.0 Release date. 2019-11-15

- Supported AIDL SDK for install type ScanEmul
- Need to use the package name 'net.m3mobile.app.ScanEmul' for AIDL

Version 1.2.1 Release date. 2019-11-20

- Added UHF_DEMO source code description
- Added Examples of Honeywell Code Setting

Version 1.2.2 Release date. 2020-03-03

- Added ImageCaputre SDK

Version 1.2.3 Release date. 2020-03-05

- Added Xamarin.Forms Scanner SDK Tutorial
- Added getting scanner information

Version 1.2.4 Release date. 2020-06-03

- Added Long-range Scanner Snapshot

Android SDK Manual Index

1. Key Remap

a. Key Remap SDK

Classes - APIs

- KeyRemap
  - Member classes
    - KeyLScan LScan
    - KeyRScan RScan
    - KeyAction Action
    - KeyCam Cam
    - KeyVolUp VolUp
    - KeyVolDown VolDown
    - KeyBack Back
    - KeyHome Home
    - KeyMenu Menu
  - Common Functions
    - getDefaultKey()
    - setDefaultKey()
    - setKey(int)
    - getKey()
- KeyLScan
- KeyRScan
- KeyCam
- KeyVolUp
- KeyVolDown
- KeyBack
- KeyHome
- KeyMenu
- KeyAction

Tutorial

1. Initialization

```
import com.m3.sdk.key.KeyRemap;

private KeyRemap mKey = new KeyRemap();
```

2. Get/Set Key code

```
// get
int nCode = 0;
nCode = mKey.VolUp.getKey();

// set
nCode = KeyRemap.getDisableKeyCode();
mKey.VolUp.setKey(nCode);
```

3. Get/Set Default Key code

```
// get
int nCode = 0;
nCode = mKey.VolUp.getDefaultKey();

// set
mKey.VolUp.setDefaultKey();
```

APIs

int getDefaultKey()

Get 'Default Key code'.

Parameter

None

Return

Int

Key Code

Example

```
// get
int nCode = 0;
nCode = mKey.VolUp.getDefaultKey();
```

boolean setDefaultKey()

Set the key as 'Default Key Code'.

Parameter

None

Return

boolean

true or false

Example

```
// set
mKey.VolUp.setDefaultKey();
```

boolean setKey(int setKeyCode)

Change the key setting.

Parameter

int Desired Key Code

Return

boolean true or false

Example

```
nCode = KeyRemap.KEY_DISABLE;
mKey.VolUp.setKey(nCode);
```

int getKey()

Get 'Key Code'.

Parameter

None

Return

int

Assigned key code

Example

```
int nCode = 0;
nCode = mKey.VolUp.getKey();
```

2. SCANNER

a. Scanner API SDK

Classes - APIs

- Barcode
  - void setScanner(Boolean enable)
  - void scanStart()
  - void scanDispose()
- BarcodeBroadcast
- BarcodeListener
  - void onBarcode(String barcode)
  - void onBarcode(String barcode, String codeType)
- BarcodeManager
  - void addListener(BarcodeListener bl)
  - void removeListener(BarcodeListener bl)
  - void dismiss()

Tutorial

1. Initialization

```
import com.m3.sdk.scannerlib.Barcode;
import com.m3.sdk.scannerlib.BarcodeListener;
import com.m3.sdk.scannerlib.BarcodeManager;
import com.m3.sdk.scannerlib.Barcode.Symbology;

private Barcode mBarcode = null;
private BarcodeListener mListener = null;
private BarcodeManager mManager = null;
private Symbology mSymbology = null;

mBarcode = new Barcode(this);
mManager = new BarcodeManager(this);
mSymbology = mBarcode.getSymbologyInstance();
mBarcode.setScanner(true);

mListener = new BarcodeListener() {
        @Override
        public void onBarcode(String strBarcode) {
                Log.i("ScannerTest","result="+strBarcode);
        }

        @Override
        public void onBarcode(String barcode, String codeType) {
                Log.i("ScannerTest","result="+barcode);
                mTvResult.setText("data: " + barcode + " type: " + codeType);
        }
};

mManager.addListener(mListener);
```

- AndroidManifest.xml

```
<uses-permission android:name="android.permission.WRITE_SETTINGS" />
```

2. Start and Stop reading barcode

```
public void onClick(View vw) {
        int id = vw.getId();

        if(id == R.id.startread){
                mBarcode.scanStart();
        }else if(id == R.id.stopread){
                mBarcode.scanDispose();
        }

}
```

3. Close

```
mManager.removeListener(mListener);
mManager.dismiss();
mBarcode.setScanner(false);
```

4. Getting Scanner Information

a. Request information

```
Intent intent = null;
intent = new Intent(ConstantValues.SCANNER_ACTION_IS_ENABLE);
mContext.sendOrderedBroadcast(intent, null);
```

b. Receive information by BroadcastReceiver

```
if(intent.getAction().equals(ConstantValues.SCANNER_ACTION_STATUS)){
    String strMessage = "";
    int nStatus = intent.getIntExtra(ConstantValues.SCANNER_EXTRA_STATUS, 0);
    Log.i(TAG, "Status: " + nStatus + (nStatus &
ConstantValues.SCANNER_STATUS_SCANNER_OPEN_SUCCESS));
    if((nStatus & ConstantValues.SCANNER_STATUS_SCANNER_OPEN_SUCCESS) > 0){
        strMessage += "Scanner Open Success\n";
    }else if((nStatus & ConstantValues.SCANNER_STATUS_SCANNER_OPEN_FAIL) > 0){
        strMessage += "Scanner Open Fail\n";
    }else if((nStatus & ConstantValues.SCANNER_STATUS_SCANNER_CLOSE_SUCCESS) > 0){
        strMessage += "Scanner Close Success\n";
    }else if((nStatus & ConstantValues.SCANNER_STATUS_SCANNER_CLOSE_FAIL) > 0){
        strMessage += "Scanner Open Fail\n";
    }

    String module =
intent.getStringExtra(ConstantValues.SCANNER_EXTRA_MODULE_TYPE);
    strMessage += "Scanner: " + module;
}
```

APIs

- **Barcode class**

void setScanner(boolean enable)

Enable Scanner status or disable.

Parameter

enable

Set Scanner status.

Return

Void

void scanStart()

Shooting the beam for barcode reading

Parameter

None

Return

v*oid*

void scanDispose()

Stop the beam

Parameter

None

Return

v*oid*

- **BarcodeManager class**

void addListener(BarcodeListener bl)

Description

Add User Instance of BarcodeListener class for both Barcode reading result and Symbology set result.

Parameter

bl BarcodeListener class instance to get callback event

Return

void

void removeListener(BarcodeListener bl)

Description

Remove the user instance of BarcodeListener class added.

Parameter

bl BarcodeListener class instance to be removed.

Return

void

void dismiss()

Description

Terminate BarcodeManager.

Parameter

None

Return

void

b. Scanner Intent SDK

1. Intent Constant Values

```
// Scanner Setting
SCANNER_ACTION_SETTING_CHANGE = "com.android.server.scannerservice.settingchange";
// Setting parameter and value
SCANNER_ACTION_PARAMETER = "android.intent.action.SCANNER_PARAMETER";
// Scanner enable or Disable
SCANNER_ACTION_ENABLE = "com.android.server.scannerservice.m3onoff";
// Decoding start
SCANNER_ACTION_START = "android.intent.action.M3SCANNER_BUTTON_DOWN";
// Decoding Stop
SCANNER_ACTION_CANCEL = "android.intent.action.M3SCANNER_BUTTON_UP";
// Receiving decoding result and setting parameter result
SCANNER_ACTION_BARCODE = "com.android.server.scannerservice.broadcast";
// request to get information
SCANNER_ACTION_IS_ENABLE = "com.android.server.scannerservice.m3onoff.ison";
// Receive a scanner Module and Status
SCANNER_ACTION_STATUS = "scanemul.action.status"

// extra about scanner enable or disable
SCANNER_EXTRA_ENABLE = "scanneronoff";
// Decoding result of Barcode
SCANNER_EXTRA_BARCODE_DATA = "m3scannerdata";
// decoding result of Barcode type
SCANNER_EXTRA_BARCODE_CODE_TYPE = "m3scanner_code_type";
SCANNER_EXTRA_MODULE_TYPE = "m3scanner_module_type";
SCANNER_EXTRA_BARCODE_DEC_COUNT = "m3scanner_dec_count"; // 2D Only
SCANNER_EXTRA_STATUS = "scanemul.extra.status";
```

2. Register IntentFilter

```
IntentFilter filter = new IntentFilter();
filter.addAction(SCANNER_ACTION_BARCODE);
registerReceiver(BarcodeIntentBroadcast,filter);
```

3. Send Intent

Scanner Enable/Disable

Intent Action

SCANNER_ACTION_ENABLE

Intent Extra

Name	Value	Description
SCANNER_EXTRA_ENABLE	1	Enable
	0	Disable

Scanner Read Start

Intent Action

SCANNER_ACTION_START

Intent Extra

None

Scanner Read Stop

Intent Action

SCANNER_ACTION_CANCEL

Intent Extra

None

Getting Scanner Parameter

Intent Action

SCANNER_ACTION_PARAMETER

Intent Extra

Name	Value	Description
"symbology"	*Parameter number	
"value"	-1	

Setting Scanner Parameter

Intent Action

SCANNER_ACTION_PARAMETER

Intent Extra

Name	Value	Description

“symbology”	*Parameter number
“value”	*Value number

Sound

Intent Action

SCANNER_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	description
“setting”	“sound”	
“sound_mode”	0	Sound none
“sound_mode”	1	Beep sound
“sound_mode”	2	Ding dong sound

Vibration

Intent Action

SCANNER_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	description
“setting”	“vibration”	
“vibration_value”	1	Vibration On
	0	Vibration Off

Scanner Button Enable/Disable

Intent Action

SCANNER_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	description
“setting”	“key_press”	
“key_press_value”	1	Can press the Scanner key

	0	Can't press the key

Read Mode

Intent Action

SCANNER_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	description
"setting"	"read_mode"	
"read_mode_value"	0	Async read mode
	1	Sync read mode
	2	Continuous read mode

Output Mode

Intent Action

SCANNER_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	description
"setting"	"output_mode"	
"output_mode_value"	0	Copy and Paste
	1	Keyboard Emulation
	2	None (Copy to clipboard)

Scanner End Character

Intent Action

SCANNER_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	description
"setting"	"end_char"	
"end_char_value"	0	Enter

	1	Space
	2	Tab
	3	Keyboard Enter
	4	Keyboard space
	5	Keyboard tab
	6	None

Prefix/Postfix

Intent Action

SCANNER_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	description
"setting"	"prefix"	
"prefix_value"	String value	Prefix value
"postfix_value"	String value	Sufix value

Scanner Module

Description

Can check scanner module type by sending 'SCANNER_ACTION_IS_ENABLE' intent. M3 ScanEmul will send the 'SCANNER_ACTION_STATUS'. It can be receive using BroadcastReceiver.

Send Intent Action

SCANNER_ACTION_IS_ENABLE

"com.android.server.scannerservice.m3onoff.ison"

Receive Intent Action

SCANNER_ACTION_STATUS

"scanemul.action.status"

Intent Extra

Name	Value	description
SCANNER_EXTRA_MODULE_TYPE	"SE4710", "SE4750", "SE4850",..	Scanner Module Type

SCANNER_EXTRA_STATUS	SCANNER_STATUS_SCANNER_OPEN_FAIL = 1; SCANNER_STATUS_SCANNER_CLOSE_FAIL = 2; SCANNER_STATUS_SCANNER_OPEN_SUCCESS = 4; SCANNER_STATUS_SCANNER_CLOSE_SUCCESS = 8;

IDC folder

Intent Action

SCANNER_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	description
"setting"	"idc_folder"	
"folder_value"	String value	Folder value

IDC name

Intent Action

SCANNER_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	description
"setting"	"idc_name"	
"name_value"	String value	Name value

c. AIDL SDK

getAIDLPackageName method lets you get the package name

```
private String mServiceName;

public void bindScannerService(){
    Intent intent = null;
    intent = new Intent("net.m3mobile.app." + mServiceName + ".start");
    intent.setPackage(getAIDLPackageName(mServiceName));
    boolean bBind = bindService(intent,this, Context.BIND_AUTO_CREATE |
Context.BIND_ABOVE_CLIENT);
    Log.d(TAG, "bindScannerService " + bBind);
}

private String getAIDLPackageName(String serviceName) {
    String packageName = "net.m3mobile.app.scanemul";
    PackageManager manager = getPackageManager();
    List<ApplicationInfo> list = manager.getInstalledApplications(PackageManager.GET_META_DATA);
    for (ApplicationInfo info : list) {
        try {
            if(info.packageName.equals("net.m3mobile.app." + serviceName))
                packageName = "net.m3mobile.app." + serviceName;
        } catch (Exception e) {
            Log.e(TAG, "getPackageType Exception : " + e.getMessage());
        }
    }
    return packageName;
}
```

String getAIDLPackageName(String serviceName);

Parameter

serviceName

Return

String

Example

intent.setPackage(getAIDLPackageName(mServiceName));

ScannerType	serviceName
1D Scanner (Zebra)	"scannerservicezebra1d"
2D Scanner (Zebra)	"scannerservicezebra2d"
2D Scanner (Honey)	"scannerservicehoney"

1D Scanner (Zebra)

void setScanner(boolean bEnable);

Enable Scanner status or disable.

Parameter

bEnable

Set Scanner status.

Return

Void

Example

m1DService.setScanner(true);

void decodeStart();

Shooting the beam for barcode reading

Parameter

None

Return

void

Example

m1DService.decodeStart();

void decodeStop();

Stop the beam

Parameter

None

Return

void

Example

m1DService.decodeStop();

void setEndCharMode(int mode);

Set a ending character after barcode reading

Parameter

mode	int	
	0	Enter
	1	SPACE
	2	TAB
	3	Keyboard stroke Enter
	4	Keyboard stroke Space
	5	Keyboard stroke tab
	6	None

Return

void

Example

m1DService.setEndCharMode(0);

void setOutputMode(int mode);

Set the output Mode of reading result.

Parameter

mode	int	
	0	Print directly
	1	Keyboard stroke
	2	Copy to clipboard

Return

void

Example

m1DService.setOutputMode(1);

void setPrefix(String prefix);

Set barcode prefix string.

Parameter

prefix String

Return

void

Example

m1DService.setPrefix("pre_");

void setPostfix(String postfix);

Set barcode postfix string.

Parameter

postfix String

Return

void

Example

m1DService.setPostfix("_suffix");

void setSoundMode(int nMode);

Set mode of the sound after reading.

Parameter

nMode int

0	None
1	BEEP
2	Ding-Dong

Return

void

Example

m1DService.setSoundMode(1);

void setVibration(boolean isOn);

Set the Vibration to on and off.

Parameter

isOn boolean

Return

void

Example

m1DService.setVibration(true);

void setReadMode(int nMode);

Set a bar-code reading mode

Parameter

nMode int

0	reading asynchronous
1	reading synchronous
2	reading continuous

Return

void

Example

m1DService.setReadMode(1);

void setScannerTriggerMode(int nMode);

Set a mode of Scanner trigger key.

Parameter

nMode int

0 Enable

1 Can't use scanner beam

2 Using only calling API.

Return

void

Example

m1DService.setScannerTriggerMode(0);

int setScanParameter(int num, int val);

Set a scanner parameter.

Parameter

num

A number of scanner parameter for setting

val

A value for option setting.

Return

int

0 if a function completed successfully

Example

int error_num = 0;

error_num = m1DService.setScanParameter (3,1);

// Please refer to Appenidx – 1D Scanner Parameters for parameter numbers and values.

int getScanParameter(int num);

Get a value of a parameter.

Parameter

num

Parameter number

Return

int

the value of a specified parameter number.

Example

int nNumber = m1DService.getScanParameter (3)

// Please refer to Appenidx – 1D Scanner Parameters for parameter numbers and values.

byte[] setParamRequest(byte[] params, byte[] val);

Set a scanner parameter.

Parameter

params

Scanner parameter for setting

val

A value for option setting.

Return

byte[]

0 if a function completed successfully

Example

byte[] result;

result = m1DService.setParamRequest (new byte[]{0x03}, new byte[]{0x01});

// Please refer to Appenidx – 1D Scanner Parameters for parameter numbers and values.

byte[] getParamRequest(byte[] params);

Get a value of a parameter.

Parameter

params

Parameters

Return

byte[]

the value of a specified parameter

Example

byte[] values = m1DService.getParamRequest(new byte[]{0x03});

// Please refer to Appenidx – 1D Scanner Parameters for parameter numbers and values.

2D Scanner (Zebra)

void setScanner(boolean bEnable);

Enable Scanner status or disable.

Parameter

bEnable

Set Scanner status.

Return

Void

Example

m2DService.setScanner(true);

void decodeStart();

Shooting the beam for barcode reading

Parameter

None

Return

void

Example

m2DService.decodeStart();

void decodeStop();

Stop the beam

Parameter

None

Return

void

Example

m2DService.decodeStop();

void setEndCharMode(int mode);

Set a ending character after barcode reading

Parameter

mode	int	
	0	Enter
	1	SPACE
	2	TAB
	3	Keyboard stroke Enter
	4	Keyboard stroke Space
	5	Keyboard stroke tab
	6	None

Return

void

Example

m2DService.setEndCharMode(0);

void setOutputMode(int mode);

Set the output Mode of reading result.

Parameter

mode	int	
	0	Print directly
	1	Keyboard stroke
	2	Copy to clipboard

Return

void

Example

m2DService.setOutputMode(1);

void setPrefix(String prefix);

Set barcode prefix string.

Parameter

prefix String

Return

void

Example

m2DService.setPrefix("pre_");

void setPostfix(String postfix);

Set barcode postfix string.

Parameter

postfix String

Return

void

Example

m2DService.setPostfix("_suffix");

void setSoundMode(int nMode);

Set mode of the sound after reading.

Parameter

nMode int

0	None
1	BEEP
2	Ding-Dong

Return

void

Example

m2DService.setSoundMode(1);

void setVibration(boolean isOn);

Set the Vibration to on and off.

Parameter

isOn boolean

Return

void

Example

m2DService.setVibration(true);

void setReadMode(int nMode);

Set a bar-code reading mode

Parameter

nMode int

0	reading asynchronous
1	reading synchronous
2	reading continuous

Return

void

Example

m2DService.setReadMode(1);

void setScannerTriggerMode(int nMode);

Set a mode of Scanner trigger key.

Parameter

nMode int

0 Enable

1 Can't use scanner beam

2 Using only calling API.

Return

void

Example

m2DService.setScannerTriggerMode(0);

int setScanParameter(int num, int val);

Set a scanner parameter.

Parameter

num

A number of scanner parameter for setting

val

A value for option setting.

Return

int

0 if a function completed successfully

Example

int error_num = 0;

error_num = m2DService. setScanParameter (292,1);

// Please refer to Appendix – 2D Scanner Parameter for parameter numbers and values.

int getScanParameter(int num);

Get a value of a parameter.

Parameter

num

Parameter number

Return

int

the value of a specified parameter number.

Example

int nNumber = m2DService. getScanParameter (339)

// Please refer to Appendix – 2D Scanner Parameter for parameter numbers and values.

2D Scanner (Honeywell)

Class

```
public class SymbolConfig {
    public int symID;
    public int Mask;
    public int Flags;
    public int MinLength;
    public int MaxLength;
}
```

Fields

Flags

Logical OR of valid flags for the given symbology

Mask

Logical OR of valid masks

MaxLength

Maximum length for valid barcode string for this symbology

MinLength

Minimum length for valid barcode string for this symbology

symID

Symbology ID

Constant Value

Symbology ID

```
public static final int SYM_AZTEC = 0;
public static final int SYM_CODABAR = 1;
public static final int SYM_CODE11 = 2;
public static final int SYM_CODE128 = 3;
public static final int SYM_CODE39 = 4;
public static final int SYM_CODE93 = 6;
public static final int SYM_COMPOSITE = 7;
public static final int SYM_DATAMATRIX = 8;
public static final int SYM_EAN8 = 9;
public static final int SYM_EAN13 = 10;
public static final int SYM_INT25 = 11;
```

```
public static final int SYM_MAXICODE = 12;
public static final int SYM_MICROPDF = 13;
public static final int SYM_PDF417 = 15;
public static final int SYM_QR = 17;
public static final int SYM_RSS = 18;
public static final int SYM_UPCA = 19;
public static final int SYM_UPCE0 = 20;
public static final int SYM_UPCE1 = 21;
public static final int SYM_ISBT = 22;
public static final int SYM_BPO = 23;
public static final int SYM_CANPOST = 24;
public static final int SYM_AUSPOST = 25;
public static final int SYM_IATA25 = 26;
public static final int SYM_CODABLOCK = 27;
public static final int SYM_JAPOST = 28;
public static final int SYM_PLANET = 29;
public static final int SYM_DUTCHPOST = 30;
public static final int SYM_MSI = 31;
public static final int SYM_TLCODE39 = 32;
public static final int SYM_TRIOPTIC = 33;
public static final int SYM_CODE32 = 34;
public static final int SYM_STRT25 = 35;
public static final int SYM_MATRIX25 = 36;
public static final int SYM_CHINAPOST = 38;
public static final int SYM_KOREAPOST = 39;
public static final int SYM_TELEPEN = 40;
public static final int SYM_COUPONCODE = 43;
public static final int SYM_USPS4CB = 44;
public static final int SYM_IDTAG = 45;
public static final int SYM_GS1_128 = 47;
public static final int SYM_HANXIN = 48;
public static final int SYM_POSTALS = 50;
public static final int SYM_US_POSTALS1 = 51;
public static final int SYMBOLOGIES = 52;
public static final int SYM_ALL = 100;
```

Masks

```
/*Mask Combination*/
// Enable Symbology bit
public static final int SYMBOLOGY_ENABLE = 1;
// Enable usage of check character
public static final int SYMBOLOGY_CHECK_ENABLE = 2;
// Send check character
public static final int SYMBOLOGY_CHECK_TRANSMIT = 4;
// Include the start and stop characters in result string
public static final int SYMBOLOGY_START_STOP_XMIT = 8;
// Code39 append mode
/** @deprecated */
public static final int SYMBOLOGY_ENABLE_APPEND_MODE = 16;
// Enable Code39 Full ASCII
public static final int SYMBOLOGY_ENABLE_FULLASCII = 32;
// UPC-A/UPC-E send NUM Sys
public static final int SYMBOLOGY_NUM_SYS_TRANSMIT = 64;
// Enable 2-digit Addenda (UPC & EAN)
public static final int SYMBOLOGY_2_DIGIT_ADDENDA = 128;
// Enable 5-digit Addenda (UPC & EAN)
public static final int SYMBOLOGY_5_DIGIT_ADDENDA = 256;
// Only allow codes with addenda (UPC & EAN)
public static final int SYMBOLOGY_ADDENDA_REQUIRED = 512;
// Include Addenda separator space in returned string
public static final int SYMBOLOGY_ADDENDA_SEPARATOR = 1024;
// upca to ean13
public static final int SYMBOLOGY_UPCA_TRANSLATE_TO_EAN13 = 2097152;
// Extended UPC-E
public static final int SYMBOLOGY_EXPANDED_UPCE = 2048;
// UPC-E1 enable (use SYMBOLOGY_ENABLE for UPC-E0)
public static final int SYMBOLOGY_UPCE1_ENABLE = 4096;
// Enable UPC composite codes
public static final int SYMBOLOGY_COMPOSITE_UPC = 8192;
// Include Australian postal bar data in string
public static final int SYMBOLOGY_AUSTRALIAN_BAR_WIDTH = 65536;
// Enable RSE Symbology bit
public static final int SYMBOLOGY_RSE_ENABLE = 8388608;
// Enable RSL Symbology bit
public static final int SYMBOLOGY_RSL_ENABLE = 16777216;
```

```
// Enable RSS Symbology bit
public static final int SYMBOLOGY_RSS_ENABLE = 33554432;
// Enable all RSS versions
public static final int SYMBOLOGY_RSX_ENABLE_MASK = 58720256;
// Telepen Old Style mode
public static final int SYMBOLOGY_TELEPEN_OLD_STYLE = 67108864;
// Codabar concatenate
public static final int SYMBOLOGY_CODABAR_CONCATENATE = 536870912;
// Numeric N Table
public static final int SYMBOLOGY_AUS_POST_NUMERIC_N_TABLE = 1048576;
// Alphanumeric C Table
public static final int SYMBOLOGY_AUS_POST_ALPHANUMERIC_C_TABLE = 2097152;
// Combination N and C Tables
public static final int SYMBOLOGY_AUS_POST_COMBINATION_N_AND_C_TABLES = 4194304;

/*Flag Combination*/
// Flags are valid
public static final int SYM_MASK_FLAGS = 1;
// Min Length valid
public static final int SYM_MASK_MIN_LEN = 2;
// max Length Valid
public static final int SYM_MASK_MAX_LEN = 4;
// All flags are valid
public static final int SYM_MASK_ALL = 7;
```

API

```
void setScanner(boolean bEnable);
```

Enable Scanner status or disable.

Parameter

bEnable

Set Scanner status.

Return

Void

Example

mHService.setScanner(true);

void decodeStart();

Shooting the beam for barcode reading

Parameter

None

Return

void

Example

mHService.decodeStart();

void decodeStop();

Stop the beam

Parameter

None

Return

void

Example

mHService.decodeStop();

void setEndCharMode(int mode);

Set a ending character after barcode reading

Parameter

mode	int	
	0	Enter
	1	SPACE
	2	TAB

3 Keyboard stroke Enter
4 Keyboard stroke Space
5 Keyboard stroke tab
6 None

Return

void

Example

mHService.setEndCharMode(0);

void setOutputMode(int mode);

Set the output Mode of reading result.

Parameter

mode int
0 Print directly
1 Keyboard stroke
2 Copy to clipboard

Return

void

Example

mHService.setOutputMode(1);

void setPrefix(String prefix);

Set barcode prefix string.

Parameter

prefix String

Return

void

Example

mHService.setPrefix("pre_");

void setPostfix(String postfix);

Set barcode postfix string.

Parameter

postfix String

Return

void

Example

```
mHService.setPostfix("_suffix");
```

void setSoundMode(int nMode);

Set mode of the sound after reading.

Parameter

nMode int

0	None
1	BEEP
2	Ding-Dong

Return

void

Example

```
mHService.setSoundMode(1);
```

void setVibration(boolean isOn);

Set the Vibration to on and off.

Parameter

isOn boolean

Return

void

Example

mHService.setVibration(true);

void setReadMode(int nMode);

Set a bar-code reading mode

Parameter

nMode	int	
	0	reading asynchronous
	1	reading synchronous
	2	reading continuous

Return

void

Example

mHService.setReadMode(1);

void setScannerTriggerMode(int nMode);

Set a mode of Scanner trigger key.

Parameter

nMode	int	
	0	Enable
	1	Can't use scanner beam
	2	Using only calling API.

Return

void

Example

mHService.setScannerTriggerMode(0);

int getMultiReadCount();

Get number of barcodes read in a single image

Parameter

None

Return

Int

Number of barcodes read

Example

```
// get
int nMulti = 0;
nMulti = mHService.getMultiReadCount();
```

void setMultiReadCount(in int MultiReadCount);

Set number of barcodes to read in a single image

Parameter

MultiReadCount

number of barcodes to read (1~20)

Return

void

Example

```
mHService.setMultiReadCount(1);
```

SymbolConfig getSymbologyConfig(in int symbologyID);

Get Symbology configuration for setting symbology settings.

Parameter

symbologyID

ID of symbology

Return

SymbolConfig

SymbologyConfig to retrieve

Example

```
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(symConfig.symID);
```

(Appendix c. has more examples.)

boolean setSymbologyConfig(in SymbolConfig symbol);

Set the current configuration of the passed in structure.

Parameter

symbol

SymbologyConfig to set

Return

boolean

true if the success to set

Example

```
SymbolConfig symConfig = new SymbolConfig(4);
symConfig.Flags = SymbologyFlags.SYMBOLOGY_ENABLE;
symConfig.MinLength = 5;
symConfig.MaxLength = 55;
mHService.setSymbologyConfig(symConfig);
```

(Appendix c. has more examples.)

d. LongRange Scanner SDK

1) Intent Constant Values

```
LRSCANNER_ACTION_SETTING_CHANGE = "com.android.server.lrscannerservice.settingchange";
LRSCANNER_ACTION_ENABLE = "com.android.server.lrscannerservice.m3onoff";
LRSCANNER_ACTION_START = "android.intent.action.LRSCANNER_BUTTON_DOWN";
LRSCANNER_ACTION_CANCEL = "android.intent.action.LRSCANNER_BUTTON_UP";
SCANNER_ACTION_BARCODE = "com.android.server.scannerservice.broadcast";

SCANNER_EXTRA_ENABLE = "scanneronoff";
SCANNER_EXTRA_BARCODE_DATA = "m3scannerdata";
SCANNER_EXTRA_BARCODE_CODE_TYPE = "m3scanner_code_type";
SCANNER_EXTRA_MODULE_TYPE = "m3scanner_module_type";

LRSCANNER_ACTION_TAKE_PICTURE = "android.intent.action.LRSCANNER_TAKE_PICTURE";
LRSCANNER_ACTION_TAKE_PICTURE_RESULT =
"android.intent.action.LRSCANNER_TAKE_PICTURE.result";
LRSCANNER_EXTRA_TAKE_PICTURE_RESULT = "take_picture_result";
```

2) Register IntentFilter

```
IntentFilter filter = new IntentFilter();
filter.addAction(SCANNER_ACTION_BARCODE);
filter.addAction(LRSCANNER_ACTION_TAKE_PICTURE_RESULT);
registerReceiver(BarcodeIntentBroadcast,filter);
```

3) Send Intent

Scanner Enable/Disable

Intent Action

LRSCANNER_ACTION_ENABLE

Intent Extra

Name	Value	Description
SCANNER_EXTRA_ENABLE	1	Enable
	0	Disable

Scanner Read Start

Intent Action

LRSCANNER_ACTION_START

Intent Extra

None

Scanner Read Stop

Intent Action

LRSCANNER_ACTION_CANCEL

Intent Extra

None

Sound

Intent Action

LRSCANNER_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	description
"setting"	"sound"	
"sound_mode"	0	Sound none
"sound_mode"	1	Beep sound
"sound_mode"	2	Ding dong sound

Vibration

Intent Action

LRSCANNER_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	description
"setting"	"vibration"	
"vibration_value"	1	Vibration On
	0	Vibration Off

Output Mode

Intent Action

LRSCANNER_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	description
"setting"	"output_mode"	
"output_mode_value"	0	Copy and Paste
	1	Keyboard Emulation
	2	None (Copy to clipboard)

Scanner End Character

Intent Action

LRSCANNER_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	description
"setting"	"end_char"	
"end_char_value"	0	Enter
	1	Space
	2	Tab
	3	Keyboard Enter
	4	Keyboard space
	5	Keyboard tab
	6	None

Prefix/Postfix

Intent Action

LRSCANNER_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	description
"setting"	"prefix"	
"prefix_value"	String value	Prefix value

“postfix_value”	String value	Sufix value

Take Capture

Intent Action

LRSCANNER_ACTION_TAKE_PICTUR = "android.intent.action.LRSCANNER_TAKE_PICTURE";

Receive Intent Action

LRSCANNER_ACTION_TAKE_PICTURE_RESULT =

“android.intent.action.LRSCANNER_TAKE_PICTURE.result”;

Intent Extra

Name	**Value**	**description**
“take_picture_path”	Ex) /sdcard/test1.jpg	Setting save path
“take_picture_result”	boolean	the result of capture

e. Scanner Xamarin.Forms SDK

Tutorial

A. **Interface**

Interface was made in common Xamarin part.

```
public interface IM3Scanner
{
    void SetEnable(bool bEnable);
    void DecodeStart();
    void DecodeStop();
    void RegisterReceiver();
    void UnregisterReceiver();
    void GetScanParam(int param);
    void SetScanParam(int param, int value);
    void SetKeyDisable(bool bDisable);
}
```

B. **Implement**

Xamarin Scanner can implement using the Intent Scanner SDK. You can reference on this manual. This was implemented on Xamarin.Android part. It need to use 'dependency'.

```
[assembly:Dependency(typeof(Class_Scan))]
```

```
public class Class_Scan : IM3Scanner
{
    // action scanner Settings
    private const String SCANNER_ACTION_SETTING_CHANGE =
"com.android.server.scannerservice.settingchange";
    // Action Scanner Setting Parameter and Value
    private const String SCANNER_ACTION_PARAMETER = "android.intent.action.SCANNER_PARAMETER";
    // action Scanner Enable or Disable
    private const String SCANNER_ACTION_ENABLE = "com.android.server.scannerservice.m3onoff";
    // Extra about scanner enable or Disable
    private const String SCANNER_EXTRA_ENABLE = "scanneronoff";

    // Action start decoding
    private const String SCANNER_ACTION_START = "android.intent.action.M3SCANNER_BUTTON_DOWN";
    // action Stop decoding
    private const String SCANNER_ACTION_CANCEL = "android.intent.action.M3SCANNER_BUTTON_UP";

    // Action to receive decoding result and setting parameter result
    private const String SCANNER_ACTION_BARCODE =
"com.android.server.scannerservice.broadcast";
    // Extra about decoding result
    public const String SCANNER_EXTRA_BARCODE_DATA = "m3scannerdata";
    // Extra about code type
    public const String SCANNER_EXTRA_BARCODE_CODE_TYPE = "m3scanner_code_type";
```

```
    private ScanReceiver _scanReceiver = new ScanReceiver();

    public void DecodeStart()
    {
        Intent intent = new Intent(SCANNER_ACTION_START);
        Android.App.Application.Context.SendBroadcast(intent);
    }

    public void DecodeStop()
    {
        Android.App.Application.Context.SendBroadcast(new Intent(SCANNER_ACTION_CANCEL));
    }

    public void RegisterReceiver()
    {
        // for getting decoding result and setParam, getParam result.
        System.Diagnostics.Debug.WriteLine(String.Format("RegisterReceiver"));
        IntentFilter filter = new IntentFilter();
        filter.AddAction(SCANNER_ACTION_BARCODE);
        Android.App.Application.Context.RegisterReceiver(_scanReceiver, filter);
    }

    public void UnregisterReceiver()
    {
        System.Diagnostics.Debug.WriteLine(String.Format("UnregisterReceiver"));
        Android.App.Application.Context.UnregisterReceiver(_scanReceiver);
    }

    public void SetEnable(bool bEnable)
    {
        // Scanner enable or disable. If you want to prevent only decoding, we recommend
'SetKeyDisable'
        Intent intent = new Intent(SCANNER_ACTION_ENABLE);
        intent.PutExtra(SCANNER_EXTRA_ENABLE, bEnable ? 1 : 0);
        Android.App.Application.Context.SendBroadcast(intent);
    }

    public void GetScanParam(int param)
    {
        Intent intent = new Intent(SCANNER_ACTION_PARAMETER);
        intent.PutExtra("symbology", param);
        intent.PutExtra("value", -1);
        Android.App.Application.Context.SendBroadcast(intent);
    }

    public void SetScanParam(int param, int value)
    {
        Intent intent = new Intent(SCANNER_ACTION_PARAMETER);
        intent.PutExtra("symbology", param);
        intent.PutExtra("value", value);
        Android.App.Application.Context.SendBroadcast(intent);
    }
```

```
    public void SetKeyDisable(bool bDisable)
    {
        Intent intent = new Intent(SCANNER_ACTION_SETTING_CHANGE);
        intent.PutExtra("setting", "key_press");
        intent.PutExtra("key_press_value", bDisable ? 0 : 1);
        Android.App.Application.Context.SendBroadcast(intent);
    }

    public class ScanReceiver : BroadcastReceiver
    {
        private String barcode;
        private String type;
        private App scanApp = new App();

        public override void OnReceive(Context context, Intent intent)
        {
            Android.Util.Log.Info("ScanReceiver", "onReceiver: " + intent.Action);
            if (intent.Action.Equals(SCANNER_ACTION_BARCODE))
            {
                barcode = intent.GetStringExtra(SCANNER_EXTRA_BARCODE_DATA);
                if(barcode != null)
                {
                    // Send Barcode Data
                    type = intent.GetStringExtra(SCANNER_EXTRA_BARCODE_CODE_TYPE);
                    MessagingCenter.Send<App, string>(scanApp, "barcode", barcode);
                    System.Diagnostics.Debug.WriteLine(String.Format("OnReceive barcode: " +
barcode + " type: " + type));
                }
                else
                {
                    // Send Parameter data
                    int nParam = intent.GetIntExtra("symbology", -1);
                    int nValue = intent.GetIntExtra("value", -1);
                    MessagingCenter.Send<App, int>(scanApp, "value", nValue);
                    System.Diagnostics.Debug.WriteLine(String.Format("OnReceive param: " +
nParam + " value: " + nValue));
                }
            }
        }
    }

}
```

C. **Calling functions**

This part need to make on Xamarin common part. The scanner can be called by DependencyService and MessagingCenter. The SetParameter can be reference by zebra parameter list in this manual.

```
public partial class MainPage : ContentPage
{
    private ObservableCollection<string> _scanned { get; set; }
```

```
public MainPage()
{
    InitializeComponent();
    _scanned = new ObservableCollection<string>();
    listView_scanned.ItemsSource = _scanned;
}

protected override void OnAppearing()
{
    base.OnAppearing();

    var scan = DependencyService.Get<IM3Scanner>();
    scan.RegisterReceiver();

    MessagingCenter.Subscribe<App, string>(this, "barcode", (sender, arg) =>
    {
        _scanned.Add("Data: " + arg);
    });

    MessagingCenter.Subscribe<App, int>(this, "value", (sender, arg) =>
    {
        edValue.Text = "" + arg;
    });
}

protected override void OnDisappearing()
{
    var scan = DependencyService.Get<IM3Scanner>();
    MessagingCenter.Unsubscribe<App, string>(this, "barcode");
    scan.UnregisterReceiver();
    base.OnDisappearing();
}

private void Button_Clicked_Start(object sender, EventArgs e)
{
    var scan = DependencyService.Get<IM3Scanner>();

    scan.DecodeStart();
}

private void Button_Clicked_Stop(object sender, EventArgs e)
{
    var scan = DependencyService.Get<IM3Scanner>();

    scan.DecodeStop();
}

private void Button_Clicked_Enable(object sender, EventArgs e)
{
    var scan = DependencyService.Get<IM3Scanner>();
    scan.SetEnable(true);
}
```

```
    private void Button_Clicked_Disable(object sender, EventArgs e)
    {
        var scan = DependencyService.Get<IM3Scanner>();
        scan.SetEnable(false);
    }

    private void Button_Clicked_GetParam(object sender, EventArgs e)
    {
        int nParam = Int32.Parse(edParam.Text);
        var scan = DependencyService.Get<IM3Scanner>();
        scan.GetScanParam(nParam);
        // return by MessagingCenter
    }


    private void Button_Clicked_SetParam(object sender, EventArgs e)
    {
        int nParam = Int32.Parse(edParam.Text);
        int nValue = Int32.Parse(edValue.Text);
        var scan = DependencyService.Get<IM3Scanner>();
        scan.SetScanParam(nParam, nValue);
    }

    private void chkKeyDisable_CheckedChanged_KeyDisable(object sender,
CheckedChangedEventArgs e)
    {
        var scan = DependencyService.Get<IM3Scanner>();
        scan.SetKeyDisable(chkKeyDisable.IsChecked);
    }
}
```

3. UHF RFID Reader

a. UHF RFID Intent SDK

Intent Constant Values

```
public static final String SCANNER_ACTION_BARCODE =
"com.android.server.scannerservice.broadcast";

public static final String UGR_ACTION_ENABLE = "com.android.server.ugrservice.m3onoff";
public static final String UGR_EXTRA_ENABLE = "ugronoff";

public static final String UGR_ACTION_EPC = "com.android.server.ugrservice.broadcast";
public static final String UGR_EXTRA_EPC_DATA = "m3ugrdata";
public static final String UGR_ACTION_START = "android.intent.action.M3UGR_BUTTON_DOWN";
public static final String UGR_ACTION_CANCEL = "android.intent.action.M3UGR_BUTTON_UP";
public static final String UGR_ACTION_IS_READING = "com.android.server.ugrservice.isreading";
public static final String UGR_EXTRA_IS_READING = "m3ugr_is_reading";

public static final String UGR_ACTION_GET_SETTING = "com.android.server.ugrservice.getsetting";
public static final String UGR_ACTION_SETTING = "com.android.server.ugrservice.setting";

public static final String UGR_EXTRA_POWER = "m3ugr_power";
public static final String UGR_EXTRA_REGION_OEM = "m3ugr_region_oem";
public static final String UGR_EXTRA_DLL_VERSION = "m3ugr_dll_version";
public static final String UGR_EXTRA_FIRM_VERSION = "m3ugr_firm_version";
public static final String UGR_ACTION_SETTING_CHANGE =
"com.android.server.ugrservice.settingchange";

public static final String UGR_ACTION_MEMORY_READING = "com.android.server.ugrservice.reading";
public static final String UGR_ACTION_MEMORY_WRITING = "com.android.server.ugrservice.writing";
public static final String UGR_ACTION_MEMORY_RESPONSE =
"com.android.server.ugrservice.memory.response";

public static final String UGR_EXTRA_MEMORY = "m3ugr_memory";
public static final String UGR_ACTION_LOCK = "com.android.server.ugrservice.lock";
public static final String UGR_ACTION_KILL = "com.android.server.ugrservice.kill";
public static final String UGR_ACTION_LOCK_RESPONSE =
"com.android.server.ugrservice.lock.response";
public static final String UGR_ACTION_KILL_RESPONSE =
"com.android.server.ugrservice.kill.response";
```

1. Register IntentFilter

```
//RFID
IntentFilter filter = new IntentFilter();
filter.addAction(UGRApplication.UGR_ACTION_EPC);
filter.addAction(UGRApplication.UGR_ACTION_IS_READING);
registerReceiver(resultReceiver, filter);

//Barcode
mBarcodeFilter = new IntentFilter();
mBarcodeFilter.addAction(UGRApplication.SCANNER_ACTION_BARCODE);
registerReceiver(mCodeReceiver, mBarcodeFilter);
```

Send Intent

RFID Enable/Disable

Intent Action

UGR_ACTION_ENABLE

Intent Extra

Name	Value	Description
UGR_EXTRA_ENABLE	1	Enable
	0	Disable

RFID Module reset

Intent Action

UGR_ACTION_ENABLE

Intent Extra

Name	Value	Description
"module_reset"	true	UGR Enable 시 RFID Module 전원을 Off -> On 한다.
	False	Enable 시 RFID Module 전원을 reset 하지 않는다.

RFID Read Start

Intent Action

UGR_ACTION_START

Intent Extra

None

RFID Read Stop

Intent Action

UGR_ACTION_CANCEL

Intent Extra

None

Region OEM

Intent Action

UGR_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	Description
"setting"	"region_oem"	
"region_oem_value"	0	KOREA_NEW
	1	KOREA_WEAK
	2	KOREA_OLD
	3	USA
	4	EURO
	5	EURO_NEW
	6	JAPAN
	7	CHINA
	8	AUSTRALIA
	9	BRAZIL
	10	MALAYSIA
	11	TAIWAN

Power

Intent Action

UGR_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	Description
"setting"	"power"	
"power_value"	0 ~ 300	

Inventory Mode

Intent Action

UGR_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	Description
"setting"	"read_mode"	
"read_mode_value"	0	Overlap
	1	Multiple
	2	Once

read_mode_value

Overlap : 중복여부와 관계없이 다수의 TAG를 read합니다.

Multiple : 중복을 제외한 다수의 TAG를 read합니다.

Once : 단일 TAG를 read합니다.

Trigger Mode

Intent Action

UGR_ACTION_SETTING_CHANGE

Intent Extra

Name	Value	Description
"setting"	"trigger_mode"	
"trigger_mode_value"	0	RFID Only
	1	Scanner Only
	2	Both

Get Setting

Intent Action

UGR_ACTION_GET_SETTING

Intent Extra

Name	Value	Description
"setting"	"power"	
	"region_oem"	
	"version"	

Memory Reading

Intent Action

UGR_ACTION_MEMORY_READING

Intent Extra

Name	Value	Description
"memory_bank"	0	RESERVED
	1	EPC
	2	TID
	3	USER
"offset"		
"length"		
"password"		

Memory Writing

Intent Action

UGR_ACTION_MEMORY_WRITING

Intent Extra

Name	Value	Description
"memory_bank"	0	RESERVED
	1	EPC
	2	TID
	3	USER
"offset"		
"length"		
"data"		
"password"		

- 기능을 수행하기 전 하나의 TAG만 지정할 수 있도록 power를 조정하십시오.
- Memory read/write는 word단위로 수행됩니다.
- Offset은 시작 위치를 가리키며 0부터 시작합니다.
- Offset, length 값은 정수입니다.

  Ex) EPC값이 123456780123 인 TAG를 000056780123으로 write하려면(이때 password는 초기값이라고 가정하고 lock되지 않았을 경우를 상정)

```
Intent intent = new Intent(UGR_ACTION_MEMORY_WRITING);
intent.putExtra("offset", 0);
intent.putExtra("length", 1);
intent.putExtra("data", "0000");
intent.putExtra("password", "000000");
```

sendBroadcast(intent);

RFID Lock

Intent Action

UGR_ACTION_LOCK

Intent Extra

Name	Value	Description
"acc_permission"		0 : ACCESIBLE
"kill_permission"		1 : ALWAYS ACCESSIBLE
"epc_permission"		2 : SECURED ACCESSIBLE
"tid_permission"		3 : ALWAYS NOT ACCESSIBLE
"user_permission"		4 : NO CHANGE
"acc_pwd"		

RFID Kill

Intent Action

UGR_ACTION_KILL

Intent Extra

Name	Value	Description
"kill_pwd"		

Receive Intent

Inventory EPC Data

Intent Action

UGR_ACTION_EPC

Intent Extra

Name	Type	Description
UGR_EXTRA_EPC_DATA	String	

- Since reading speed has been dramatically increased since UHFEmul version 1.2.3 and later, we recommend using AIDL instead of Intent Receiver

Inventory Reading Flag

Intent Action

UGR_ACTION_IS_READING

Intent Extra

Name	Type	Description
UGR_EXTRA_IS_READING	boolean	

READ가 진행중일 때 true를 반환합니다

Power

Intent Action

UGR_ACTION_SETTING

Intent Extra

Name	value	Description
"setting"	"power"	
UGR_EXTRA_POWER	0 ~ 300	

Region OEM

Intent Action

UGR_ACTION_SETTING

Intent Extra

Name	Value	Description
"setting"	"region_oem"	
UGR_EXTRA_REGION_OEM	0	KOREA_NEW
	1	KOREA_WEAK
	2	KOREA_OLD
	3	USA
	4	EURO
	5	EURO_NEW
	6	JAPAN
	7	CHINA
	8	AUSTRALIA
	9	BRAZIL
	10	MALAYSIA

	11	TAIWAN

Version

Intent Action

UGR_ACTION_SETTING

Intent Extra

Name	value	Description
"setting"	"version"	
UGR_EXTRA_DLL_VERSION		
UGR_EXTRA_FIRM_VERSION		

Access Response

Intent Action

UGR_ACTION_MEMORY_RESPONSE

Intent Extra

Name	Type	Description
UGR_EXTRA_MEMORY	String	
"success"	Boolean	

read/write에 대한 성공여부를 반환합니다.

Lock/Kill Response

Intent Action

UGR_ACTION_LOCK_RESPONSE, UGR_ACTION_KILL_RESPONSE

Intent Extra

Name	Type	Description
"success"	Boolean	

lock/kill 에 대한 성공여부를 반환합니다.

b. UHF RFID AIDL Callback SDK

- Since reading speed has been dramatically increased since UHFEmul version 1.2.3 and later, we recommend using AIDL instead of Intent Receiver

```
// Disable inventory data intent
Intent intent = new Intent(UGR_ACTION_SETTING_CHANGE);
intent.putExtra("setting", "intent_enable");
intent.putExtra("intent_enable_value", false);
sendOrderedBroadcast(intent, null);
```

```
boolean registerUHFServiceCallback(IUHFServiceCallback callback);
```

Register uhf emul callback.

Parameter

callback

Callback interface

Return

boolean

Example

```
boolean bSuccess = m_remoteSvc.registerUHFServiceCallback(m_remoteCallback);

IUHFServiceCallback m_remoteCallback = new IUHFServiceCallback.Stub() {
        @Override
        public void onInventory(String epc) throws RemoteException {
            // Handle epc data from UI thread
        }

        @Override
        public void onIsReading(boolean isReading) throws RemoteException {
            // You can check the read status
        }
    };
```

```
boolean unregisterUHFServiceCallback(IUHFServiceCallback callback);
```

Unregister uhf emul callback.

Parameter

callback

Callback interface

Return

boolean

Example

boolean bSuccess = m_remoteSvc.unregisterUHFServiceCallback(m_remoteCallback);

c. UHF_DEMO source code 해설서

Intent를 사용하는 방식이 ResultWindow.java에, AIDL을 사용하는 방식은 ResultWindow_aidl.java 에 각각 구현되어 있습니다.

Intent filter와 AIDL callback, Broadcast receiver등의 차이가 있으므로 요주의 입니다.

또한 1.2.3 이상의 버전을 사용한다면 AIDL로 구현하는 것이 매우 좋습니다.

register broadcast receiver

```
// inventory 결과값과 Key입력을 받기위한 receiver입니다.
resultReceiver = new ResultWindowReceiver();

// Intent 방식으로 결과값을 얻을 때만 호출하십시오
IntentFilter filter = new IntentFilter();
filter.addAction(UGRApplication.UGR_ACTION_EPC);
filter.addAction(UGRApplication.UGR_ACTION_IS_READING);
registerReceiver(resultReceiver, filter);

// 스캐너를 활용한 barcode 값을 얻기 위한 receiver 입니다. RFID TAG 만 필요한 경우 생략할 수 있습니다.
mCodeReceiver = new BarcodeReceiver();
mBarcodeFilter = new IntentFilter();
mBarcodeFilter.addAction(UGRApplication.SCANNER_ACTION_BARCODE);
registerReceiver(mCodeReceiver, mBarcodeFilter);
```

AIDL service connection 구현

```
if(m_UHFSvcConnection == null) {
    m_UHFSvcConnection = new ServiceConnection() {
        @Override
        public void onServiceConnected(ComponentName name, IBinder service) {
            m_remoteSvc = IUGRTestService.Stub.asInterface(service);
            Log.d(ResultWindow_aidl.class.getSimpleName(), "Service is Connected");
            try {
                if(m_remoteSvc.registerUHFServiceCallback(m_remoteCallback))
                    Log.d(ResultWindow_aidl.class.getSimpleName(), "Callback was registered");
                else
                    Log.d(ResultWindow_aidl.class.getSimpleName(), "Registering Callback was
failed");
            } catch (RemoteException e) {
                e.printStackTrace();
            }

          // Service 정상적으로 연결됐을 때 Enable 합니다
            RFIDEnable(true);

        }

        @Override
        public void onServiceDisconnected(ComponentName name) {
            m_remoteSvc = null;
            Log.d(ResultWindow_aidl.class.getSimpleName(), "Service is Disconnected");
```

```
        }
    };
}
```

AIDL callback 구현

```
m_remoteCallback = new IUHFServiceCallback.Stub() {
    @Override
    public void onInventory(String epc) throws RemoteException {
        UIHandler.sendMessage(UIHandler.obtainMessage(UGRApplication.MSG_HANDLE_DATA, epc));
    }

    @Override
    public void onIsReading(boolean isReading) throws RemoteException {
        mIsReading = isReading;
        UIHandler.sendMessage(UIHandler.obtainMessage(UGRApplication.MSG_IS_READING));
    }
};
```

Bind Service

```
Intent intent = new Intent("net.m3mobile.ugremul.start");
intent.setPackage("net.m3mobile.ugremul");
bindService(intent,m_UHFSvcConnection, Context.BIND_AUTO_CREATE);
```

Intent disable

```
// AIDL callback을 사용할 때는 intent로 값을 받지않으므로 intent disable 합니다
intent = new Intent(UGRApplication.UGR_ACTION_SETTING_CHANGE);
intent.putExtra("setting", "intent_enable");
intent.putExtra("intent_enable_value", false);
sendOrderedBroadcast(intent, null);
```

Set read mode

```
// list view 등으로 여러 값을 한 번에 받을 경우 read mode 를 0(Overlap)으로 설정합니다.
Intent intent = new Intent(UGRApplication.UGR_ACTION_SETTING_CHANGE);
intent.putExtra("setting", "read_mode");
intent.putExtra("read_mode_value", 0);
sendOrderedBroadcast(intent, null);
```

Set trigger mode

```
// trigger mode 의 기본값은 2(Both)입니다. RFID 만 사용할 때는 0, Scanner 만 사용할 때는 1 로 설정하고
특별한 경우가 아닌 경우 기본값 2 로 되돌려 둬야합니다.(onDestroy, onResume 등에서 2 로 설정하십시오)
intent = new Intent(UGRApplication.UGR_ACTION_SETTING_CHANGE);
intent.putExtra("setting", "trigger_mode");
intent.putExtra("trigger_mode_value", 2);
sendOrderedBroadcast(intent, null);
```

Unregister broadcast receiver

```
// onDestroy등에서 구현하십시오.
unregisterReceiver(resultReceiver);
unregisterReceiver(mCodeReceiver);
```

Trigger mode control

```
// Demo app 에서는 trigger mode 를 radio button 으로 구현하였습니다.
// OnClickListener 이므로 클릭하기 이전, 즉 화면에 진입했을 때는 호출되지않으니 요주의 입니다.
RadioButton.OnClickListener OnTriggerClickListener2 = new RadioButton.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent triggerIntent = new Intent(UGRApplication.UGR_ACTION_SETTING_CHANGE);
        triggerIntent.putExtra("setting", "trigger_mode");

        switch (view.getId()) {
            case R.id.radio_trigger_rfid:
                triggerIntent.putExtra("trigger_mode_value", 0);
                mLastTriggerMode = 0;
                break;
            case R.id.radio_trigger_scanner:
                triggerIntent.putExtra("trigger_mode_value", 1);
                mLastTriggerMode = 1;
                break;
            case R.id.radio_trigger_both:
                triggerIntent.putExtra("trigger_mode_value", 2);
                mLastTriggerMode = 2;
                break;
        }

        sendBroadcast(triggerIntent, null);
    }
};
```

Inventory start/stop

```
// 실제 소스코드에는 타이머와 관련된 코드가 있으나 생략할 수 있습니다.
private void inventory(boolean bStart) {
    Intent intent;
    if(bStart) {
        intent = new Intent(UGRApplication.UGR_ACTION_START, null);
    } else {
        intent = new Intent(UGRApplication.UGR_ACTION_CANCEL, null);
    }
    sendOrderedBroadcast(intent, null);
}
```

RFID Emul enable/disable

```
private void RFIDEnable(boolean bOn) {
    int nExtra;
```

```
    if(bOn)
        nExtra = 1;
    else
        nExtra = 0;
    Intent intent = new Intent(UGRApplication.UGR_ACTION_ENABLE, null);
    intent.putExtra(UGRApplication.UGR_EXTRA_ENABLE, nExtra);
    intent.putExtra("module_reset", bOn);
    sendOrderedBroadcast(intent, null);
}
```

4. ImageCaputre

a. ImageCaputre Intent SDK

Intent Constant Values

```
//ImageCapture
public static final String SCANNER_ACTION_TAKE_PICTURE =
"android.intent.action.SCANNER_TAKE_PICTURE"; //Take picture Intent.
public static final String SCANNER_ACTION_IMAGE_CAPUTRE_SETTINGS = "image_capture_settings";
public static final String SCANNER_ACTION_TAKE_PICTURE_PATH =
"android.intent.action.SCANNER_TAKE_PICTURE_PATH";
//After take picture, response the action as path.

public static final String SCANNER_EXTRA_TAKE_PICTURE_PATH = "take_picture_path";
//Saved image path
public static final String SCANNER_ACTION_TAKE_PICTURE_RESPONSE =
"android.intent.action.SCANNER_TAKE_PICTURE_RESPONSE";
```

Tutorial

1. How to take picture

```
//take picture
Intent intent = new Intent(ConstantVariables.SCANNER_ACTION_TAKE_PICTURE);
intent.putExtra("save_path", ConstantVariables.SCANEMUL_PATH); //Path to save Image.
sendBroadcast(intent, null);
//if you send broadcast this Intent action, It's going to call the takePicture() method.
```

2. How to receive an image file

```
private BroadcastReceiver mReceiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        Log.i(TAG, "onReceive - "+action);
        if(ConstantVariables.SCANNER_ACTION_TAKE_PICTURE_PATH.equals(action)){
          String strPath =intent.getStringExtra(ConstantVariables.SCANNER_EXTRA_TAKE_PICTURE_PATH);
          // This is the path to be saved., You can set the save path.
        }
    }
};
```

3. Register and Unregister Receiver

```
//add ation to IntentFilter and register
IntentFilter filter = new IntentFilter();
filter.addAction(ConstantVariables.SCANNER_ACTION_TAKE_PICTURE_PATH);
registerReceiver(mReceiver, filter);
```

```
//Unregister Receiver
Protected void onDestroy() {
    super.onDestroy();
    unregisterReceiver(mReceiver);
}
```

NOTE : You can also set the image options. Such as ImageResolution, Iilumination, Image enhancement.. ETC.
If you need futher information about set the image options, refer to the "2D Imaging Options" on page 143.

TakePicture

Intent Action

SCANNER_ACTION_TAKE_PICTURE

Intent Extra

None

Getting TakePicture Path

Intent Action

SCANNER_ACTION_TAKE_PICTURE_PATH

Intent Extra

Name	Value	Description
"take_picture_path"	*saved path (String)	

b. Preview AIDL

Using AIDL Method

1. Methods for use preview

```
// IScannerServiceZebra2D.aidl
package net.m3mobile.app.scannerservicez2d;
import net.m3mobile.app.scanemul.IScannerCallback;

// Declare any non-default types here with import statements

interface IScannerServiceZebra2D {
          .........
    void previewStart();
    void previewStop();

           .........
    void registerScannerCallback(IScannerCallback callback);
    void unregisterScannerCallback(IScannerCallback callback);
}
```

You should call previewStart(), previewStop() method in "IScannerServiceZebra2D" to use preview.

2. callback method for aidl

```
// IScannerCallback.aidl
package net.m3mobile.app.scanemul;
import android.graphics.Bitmap;
// Declare any non-default types here with import statements

interface IScannerCallback {
   oneway void onZebraPreview(in Bitmap bitmap);
}
```

You are able to callback bitmap data with onZebraPreview() method in IScannerCallback.aidl

Tutorial

1. **To get the Service instance**

```
private IScannerServiceZebra2D mService;  // Member variable
@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.scanner);
    bindScannerService();

}

public void bindScannerService(){
   Intent intent = null;
   intent = new Intent("net.m3mobile.app.scannerservicezebra2d.start");
   intent.setPackage( "com.zebra.scanner");
   boolean bBind = bindService(intent,this, Context.BIND_AUTO_CREATE |
Context.BIND_ABOVE_CLIENT);
   Log.d(TAG, "bindScannerService " + bBind);
}


@Override
public void onServiceConnected(ComponentName componentName, IBinder iBinder) {
    ……..
   m2DService = IScannerServiceZebra2D.Stub.asInterface(iBinder);
   ……..
  try {
      m2DService.previewStart();
  } catch (RemoteException e) {
    e.printStackTrace();
  }



}
```

1) Bind the Service by call the bindScannerService() method
2) You can get the service instance with onServiceConnected()
3) After get the service instance, start the m2DService.previewStart()

2. To call the previewStart() and previewStop() of AIDL mehod

```
// It is recommended to use it as a method.
void startPreview() {

        if (mService != null) {
            try {
                Log.d(TAG, "previewStart() Try call...");
                mService.previewStart();

            } catch (RemoteException e) {
                e.printStackTrace();
            }
        } else {
                Log.d(TAG, "Failed previewStart()...");
        }
}
void stopPreview() {
        if (mService != null) {
            try {
                mService.previewStop();
            } catch (RemoteException e) {
                e.printStackTrace();
            }
        }
}
```

```
// call the startPreview() at onResume()

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

}
// call the stopPreview() at onPause()

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

3. To receive a callback

```
private IScannerCallback.Stub callback = new IScannerCallback.Stub() {
    @Override
    public void onZebraPreview(Bitmap bitmap) throws RemoteException {
        Log.d(TAG, "Callback recived");

        mImageView.setImageBitmap(bitmap);
    }//AIDL Callback
};
```

```
// Register callback

public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_preview);
    getWindow().setFormat(PixelFormat.UNKNOWN);

     ..................................
      try {
        mService.registerScannerCallback(callback);
    } catch (RemoteException e) {
        e.printStackTrace();
    }
}
```

```
// Unregister callback

@Override
protected void onDestroy() {
    super.onDestroy();
    try {
        mService.unregisterScannerCallback(callback);
    } catch (RemoteException e) {
        e.printStackTrace();
    }
}
```

void previewStart();

Start the preview

Parameter

None

Return

void

Example

mService.previewStart();

void previewStop();

Stop the preview

Parameter

None

Return

void

Example

mService.previewStop();

registerScannerCallback(IScannerCallback callback);

Register preview callback.

Parameter

callback

Callback interface

Return

void

Example

mService.registerScannerCallback(callback);

unregisterScannerCallback(IScannerCallback callback);

Unregister preview callback.

Parameter

callback

Callback interface

Return

void

Example

mService.unregisterScannerCallback(callback);

5. Appendix – Scanner Parameters

a. 1D Symbology Parameter

Scan Angle

Parameter Number	Value	description
191	*2	Wide Angle (47°)
	0	Narrow Angle (10°)
	1	Medium Angle (35°)

NOTE Narrow scan angle is not supported by Class 1 scan engines.

This parameter sets the scan angle to narrow, medium, or wide.

Timeout Between Decodes, Same Symbol

Parameter Number	Value	description
137	*10	Timeout Between Same Symbol

In **Continuous** and **Blink** triggering modes and when **Continuous Bar Code Read** is enabled, this parameter sets the minimum time that must elapse before the scan engine decodes a second bar code identical to one just decoded. This reduces the risk of accidently scanning the same symbol twice. It is programmable in 0.1 second increments from 0.0 to 9.9 seconds.

NOTE The Timeout between Decodes, Same Symbol must be greater than the Timeout Between Decodes, Different Symbols.

Timeout Between Decodes, Different Symbols

Parameter Number	Value	description
144	*2	Timeout Between Decodes, Different Symbols

In **Continuous** and **Blink** triggering modes and when **Continuous Bar Code Read** is enabled, this parameter sets the minimum time that must elapse before the scan engine decodes a second bar code different from the one just decoded. It is programmable in 0.1 second increments from 0.1 to 9.9 seconds.

NOTE The Timeout between Decodes, Different Symbols cannot be greater than or equal to the Timeout Between Decodes, Same Symbol.

Linear Code Type Security Level

Parameter Number	Value	description
78	*1	Linear Security Level 1
	2	Linear Security Level 2
	3	Linear Security Level 3
	4	Linear Security Level 4

The SE-965HP-X20XR offers four levels of decode security for linear code types (e.g. Code 39, Interleaved 2 of 5). Select higher security levels for decreasing levels of bar code quality. As security levels increase, the scan engine's aggressiveness decreases.

Select the security level appropriate for your bar code quality

Linear Security Level 1

The following code types must be successfully read twice before being decoded

Code Type	Length
Codabar	All
MSI	4 or less
D 2 of 5	8 or less
I 2 of 5	8 or less

Linear Security Level 2

All code types must be successfully read twice before being decoded.

Linear Security Level 3

Code types other than the following must be successfully read twice before being decoded. The following codes must be read three times

Code Type	Length

MSI	4 or less
D 2 of 5	8 or less
I 2 of 5	8 or less

Linear Security Level 4

All code types must be successfully read three times before being decoded.

Bi-directional Redundancy

Parameter Number	Value	description
67	*0	Disable Bi-directional Redundancy
	1	Enable Bi-directional Redundancy

UPC/EAN

UPC-A

Parameter Number	Value	description
1	*1	Enable UPC-A
	0	Disable UPC-A

Enable/Disable UPC-E

Parameter Number	Value	description
2	*1	Enable UPC-E
	0	Disable UPC-E

Enable/Disable UPC-E1

Parameter Number	Value	description
12	*0	Disable UPC-E1
	1	Enable UPC-E1

NOTE UPC-E1 is not a UCC (Uniform Code Council) approved symbology

Enable/Disable EAN-8/JAN-8

Parameter Number	Value	description
4	*1	Enable EAN-8/JAN-8
	0	Disable EAN-8/JAN-8

Enable/Disable EAN-13/JAN-13

Parameter Number	Value	description
3	*1	Enable EAN-13/JAN-13
	0	Disable EAN-13/JAN-13

Enable/Disable Bookland EAN

Parameter Number	Value	description
83	*0	Ignore Supplementals
	1	Decode UPC/EAN/JAN Only With Supplementals
	2	Autodiscriminate UPC/EAN/JAN Supplementals
	4	Enable 378/379 Supplemental Mode
	5	Enable 978/979 Supplemental Mode
	7	Enable 977 Supplemental Mode
	6	Enable 414/419/434/439 Supplemental Mode
	8	Enable 491 Supplemental Mode
	3	Enable Smart Supplemental Mode
	9	Supplemental User-Programmable Type 1
	10	Supplemental User-Programmable Type 1 and 2
	11	Smart Supplemental Plus User-Programmable 1
	12	Smart Supplemental Plus User-Programmable 1 and 2

Supplementals are bar codes appended according to specific format conventions (e.g., UPC A+2, UPC E+2, EAN 13+2). The following options are available:

- If you select Decode UPC/EAN/JAN with Supplementals, the engine only decodes UPC/EAN symbols with supplemental characters, and ignores symbols without supplementals.
- If you select Ignore UPC/EAN/JAN Supplementals, and the engine is presented with a UPC/EAN plus supplemental symbol, the engine decodes UPC/EAN and ignores the supplemental characters.
- If you select Autodiscriminate UPC/EAN/JAN Supplementals, the engine decodes UPC/EAN symbols with supplemental characters immediately. If the symbol does not have

a supplemental, the engine must decode the bar code the number of times set via Decode UPC/EAN Supplemental Redundancy before transmitting its data to confirm that there is no supplemental.

- If you select one of the following Supplemental Mode options, the engine immediately transmits EAN-13 bar codes starting with that prefix that have supplemental characters. If the symbol does not have a supplemental, the engine must decode the bar code the number of times set via Decode UPC/EAN Supplemental Redundancy before transmitting its data to confirm that there is no supplemental. The engine transmits UPC/EAN bar codes that do not have that prefix immediately.
- Enable 378/379 Supplemental Mode.
- Enable 978/979 Supplemental Mode

NOTE If you select 978/979 Supplemental Mode and are scanning Bookland EAN bar codes, see Enable/Disable Bookland EAN to enable Bookland EAN, and select a format using Bookland ISBN Format.

- Enable 977 Supplemental Mode.
- Enable 414/419/434/439 Supplemental Mode.
- Enable 491 Supplemental Mode.
- Enable Smart Supplemental Mode - applies to EAN-13 bar codes starting with any prefix listed previously
- Supplemental User-Programmable Type 1 - applies to EAN-13 bar codes starting with a 3-digit user-defined prefix. Set this 3-digit prefix using User-Programmable Supplementals.
- Supplemental User-Programmable Type 1 and 2 - applies to EAN-13 bar codes starting with either of two 3-digit user-defined prefixes. Set the 3-digit prefixes using User-Programmable Supplementals.
- Smart Supplemental Plus User-Programmable 1 - applies to EAN-13 bar codes starting with any prefix listed previously or the user-defined prefix set using User-Programmable Supplementals.
- Smart Supplemental Plus User-Programmable 1 and 2 - applies to EAN-13 bar codes starting with any prefix listed previously or one of the two user-defined prefixes set using User-Programmable Supplementals.

NOTE To minimize the risk of invalid data transmission, select either to decode or ignore supplemental characters.

Decode UPC/EAN/JAN Supplementals

Parameter Number	Value	description
16	*0	Disable Bookland EAN
	1	Enable Bookland EAN

Decode UPC/EAN Supplemental Redundancy

Parameter Number	Value	description
80	*7	

With Autodiscriminate UPC/EAN Supplementals selected, this option adjusts the number of times a symbol without supplementals are decoded before transmission. The range is from 2 to 30 times. Five or above is recommended when decoding a mix of UPC/EAN symbols with and without supplementals, and the autodiscriminate option is selected.

Transmit UPC-A Check Digit

Parameter Number	Value	description
40	*1	Transmit UPC-A Check Digit
	0	Do Not Transmit UPC-A Check Digit

The check digit is the last character of the symbol used to verify the integrity of the data. Scan the appropriate bar code below to transmit the bar code data with or without the UPC-A check digit. It is always verified to guarantee the integrity of the data.

Transmit UPC-E Check Digit

Parameter Number	Value	description
41	*1	Transmit UPC-E Check Digit
	0	Do Not Transmit UPC-E Check Digit

The check digit is the last character of the symbol used to verify the integrity of the data. Scan the appropriate bar code below to transmit the bar code data with or without the UPC-E check digit. It is always verified to guarantee the integrity of the data.

Transmit UPC-E1 Check Digit

Parameter Number	Value	description
42	*1	Transmit UPC-E1 Check Digit
	0	Do Not Transmit UPC-E1 Check Digit

The check digit is the last character of the symbol used to verify the integrity of the data. Scan

the appropriate bar code below to transmit the bar code data with or without the UPC-E1 check digit. It is always verified to guarantee the integrity of the data

UPC-A Preamble

Parameter Number	Value	description
34	*1	System Character (<SYSTEM CHARACTER> <DATA>)
	0	No Preamble (<DATA>)
	2	System Character & Country Code (< COUNTRY CODE> <SYSTEM CHARACTER> <DATA>)

Preamble characters are part of the UPC symbol, and include Country Code and System Character. There are three options for transmitting a UPC-A preamble to the host device: transmit System Character only, transmit System Character and Country Code (“0” for USA), and transmit no preamble. Select the appropriate option to match the host system.

UPC-E Preamble

Parameter Number	Value	description
35	*1	*System Character (<SYSTEM CHARACTER> <DATA>)
	0	No Preamble (<DATA>)
	2	System Character & Country Code (< COUNTRY CODE> <SYSTEM CHARACTER> <DATA>)

Preamble characters are part of the UPC symbol, and include Country Code and System Character. There are three options for transmitting a UPC-E preamble to the host device: transmit System Character only, transmit System Character and Country Code (“0” for USA), and transmit no preamble. Select the appropriate option to match the host system.

UPC-E1 Preamble

Parameter Number	Value	description
36	*1	*System Character (<SYSTEM CHARACTER> <DATA>)
	0	No Preamble (<DATA>)
	2	System Character & Country Code (< COUNTRY CODE> <SYSTEM CHARACTER> <DATA>)

Preamble characters are part of the UPC symbol, and include Country Code and System Character. There are three options for transmitting a UPC-E1 preamble to the host device: transmit System Character only, transmit System Character and Country Code (“0” for USA), and transmit no preamble. Select the appropriate option to match the host system.

Convert UPC-E to UPC-A

Parameter Number	Value	description
37	*0	Do Not Convert UPC-E to UPC-A (Disable)
	1	Convert UPC-E to UPC-A (Enable)

Enable this to convert UPC-E (zero suppressed) decoded data to UPC-A format before transmission. After conversion, the data follows UPC-A format and is affected by UPC-A programming selections (e.g., Preamble, Check Digit).

When disabled, UPC-E decoded data is transmitted as UPC-E data, without conversion.

Convert UPC-E1 to UPC-A

Parameter Number	Value	description
38	*0	Do Not Convert UPC-E1 to UPC-A (Disable)
	1	Convert UPC-E1 to UPC-A (Enable)

Enable this to convert UPC-E1 decoded data to UPC-A format before transmission. After conversion, the data follows UPC-A format and is affected by UPC-A programming selections (e.g., Preamble, Check Digit).

When disabled, UPC-E1 decoded data is transmitted as UPC-E1 data, without conversion.

EAN-8/JAN-8 Extend

Parameter Number	Value	description
39	*0	Disable EAN/JAN Zero Extend
	1	Enable EAN/JAN Zero Extend

When enabled, this parameter adds five leading zeros to decoded EAN-8 symbols to make them compatible in format to EAN-13 symbols.

When disabled, EAN-8 symbols are transmitted as is.

UPC/EAN Security Level

Parameter Number	Value	description
77	*1 - UPC/EAN Security Level 1	As bar code quality levels diminish, certain characters become prone to misdecodes before others (i.e., 1, 2, 7, 8). If misdecodes of poorly printed bar codes occur, and the misdecodes are limited to these characters, select this security level.
	0 - UPC/EAN Security Level 0	This setting allows the scan engine to operate in its most aggressive state, while providing sufficient security in decoding most "in-spec" UPC/EAN bar codes.
	2 - UPC/EAN Security Level 2	If misdecodes of poorly printed bar codes occur, and the misdecodes are not limited to characters 1, 2, 7, and 8, select this security level.
	3 - UPC/EAN Security Level 3	If misdecodes still occur after selecting Security Level 2, select this security level. Be advised, selecting this option is an extreme measure against mis-decoding severely out of spec bar codes. Selection of this level of security significantly impairs the decoding ability of the scan engine. If this level of security is

		necessary, try to improve the quality of the bar codes.

The SE965HP offers four levels of decode security for UPC/EAN bar codes. Increasing levels of security are provided for decreasing levels of bar code quality. Select higher levels of security for decreasing levels of bar code quality. Increasing security decreases the scan engine's aggressiveness, so choose only that level of security necessary for the application.

UCC Coupon Extended Code

Parameter Number	Value	description
85	*0	Disable UCC Coupon Extended Code
	1	Enable UCC Coupon Extended Code

The UCC Coupon Extended Code is an additional bar code adjacent to a UCC Coupon Code. To enable or disable UCC Coupon Extended Code, scan the appropriate bar code below.

Code 128

Parameter Number	Value	description
8	*1	Enable Code 128
	0	Disable Code 128

Set Lengths for Code 128

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Set lengths for Code 128 to any length, one or two discrete lengths, or lengths within a specific range

NOTE When setting lengths for different bar code types, enter a leading zero for single digit numbers.

Parameter Number	Value	description
209	Any Length	Length1
210		Length2

GS1-128 (formerly UCC/EAN-128)

Parameter Number	Value	description

14	*1	Enable GS1-128
	0	Disable GS1-128

ISBT 128

Parameter Number	Value	description
84	*1	Enable ISBT 128
	0	Disable ISBT 128

ISBT Concatenation Redundancy

Parameter Number	Value	description
223	*10	

If you set ISBT Concatenation to Autodiscriminate, use this parameter to set the number of times the engine must decode an ISBT symbol before determining that there is no additional symbol.

Code 39

Parameter Number	Value	description
0	*1	Enable Code 39
	0	Disable Code 39

Trioptic Code 39

Parameter Number	Value	description
13	*0	Disable Trioptic Code 39
	1	Enable Trioptic Code 39

NOTE Trioptic Code 39 and Code 39 Full ASCII cannot be enabled simultaneously. If an error beep sounds when enabling Trioptic Code 39, disable Code 39 Full ASCII and try again.

Convert Code 39 to Code 32 (Italian Pharma Code)

Parameter Number	Value	description
86	*0	Disable Convert Code 39 to Code 32
	1	Enable Convert Code 39 to Code 32

Code 32 is a variant of Code 39 used by the Italian pharmaceutical industry. Scan the appropriate bar code below to enable or disable converting Code 39 to Code 32.

NOTE Code 39 must be enabled for this parameter to function

Code 32 Prefix

Parameter Number	Value	description
231	*0	Disable Code 32 Prefix
	1	Enable Code 32 Prefix

Enable this parameter to add the prefix character "A" to all Code 32 bar codes. Convert Code 39 to Code 32 (Italian Pharma Code) must be enabled for this parameter to function.

Set Lengths for Code 39

Parameter Number	Value	description
18	Length within Range:	Length1
19	2 - 55	Length2

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Set lengths for Code 39 to any length, one or two discrete lengths, or lengths within a specific range. If Code 39 Full ASCII is enabled, Length Within a Range or Any Length are the preferred options

NOTE When setting lengths for different bar code types by scanning single digit numbers, single digit numbers must always be preceded by a leading zero.

Code 39 Check Digit Verification

Parameter Number	Value	description
48	*0	Disable Code 39 Check Digit
	1	Enable Code 39 Check Digit

When this feature is enabled, the scan engine checks the integrity of all Code 39 symbols to verify that the data complies with specified check digit algorithm. Only those Code 39 symbols which include a modulo 43 check digit are decoded. Only enable this feature if your Code 39 symbols contain a module 43 check digit.

Transmit Code 39 Check Digit

Parameter Number	Value	description
43	*0	Do Not Transmit Code 39 Check Digit (Disable)
	1	Transmit Code 39 Check Digit (Enable)

Scan a bar code below to transmit Code 39 data with or without the check digit.

NOTE Code 39 Check Digit Verification must be enabled for this parameter to function

Code 39 Full ASCII Conversion

Parameter Number	Value	description
17	*0	Disable Code 39 Full ASCII
	1	Enable Code 39 Full ASCII

NOTE Trioptic Code 39 and Code 39 Full ASCII cannot be enabled simultaneously. If you get an error beep when enabling Code 39 Full ASCII, disable Trioptic Code 39 and try again.

Code 93

Parameter Number	Value	description
9	*0	Disable Code 93
	1	Enable Code 93

Set Lengths for Code 93

Parameter Number	Value	description
26	Length within Range:	Length1
27	4 - 55	Length2

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Lengths for Code 93 may be set for any length, one or two discrete lengths, or lengths within a specific range. To set lengths via serial commands, see Setting Code Lengths Via Serial Commands.

Code 11

Parameter Number	Value	description
10	*0	Disable Code 11
	1	Enable Code 11

Set Lengths for Code 11

Parameter Number	Value	description
28	Length within Range:	Length1
29	4 - 55	Length2

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Set lengths for Code 11 to any length, one or two discrete lengths, or lengths within a specific range.

Code 11 Check Digit Verification

Parameter Number	Value	description
52	*0	Disable
	1	One Check Digit
	2	Two Check Digits

This feature allows the scan engine to check the integrity of all Code 11 symbols to verify that the data complies with the specified check digit algorithm. This selects the check digit mechanism for the decoded Code 11 bar code. The options are to check for one check digit, check for two check digits, or disable the feature.

To enable this feature, scan the bar code below corresponding to the number of check digits encoded in the Code 11 symbols.

Transmit Code 11 Check Digits

Parameter Number	Value	description
47	*0	Do Not Transmit Code 11 Check Digit(s) (Disable)
	1	Transmit Code 11 Check Digit(s) (Enable)

This feature selects whether or not to transmit the Code 11 check digit(s).

NOTE Code 11 Check Digit Verification must be enabled for this parameter to function.

Interleaved 2 of 5

Parameter Number	Value	description
6	*1	Enable Interleaved 2 of 5
	0	Disable Interleaved 2 of 5

To enable or disable Interleaved 2 of 5, scan the appropriate bar code below, and select an Interleaved 2 of 5 length from the following pages.

Set Lengths for Interleaved 2 of 5

Parameter Number	Value	description
22	One Length: 14	Length1
23		Length2

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Lengths for I 2 of 5 may be set for any length, one or two discrete lengths, or lengths within a specific range. To set lengths via serial commands, see Setting Code Lengths Via Serial Commands.

NOTE When setting lengths, include a leading zero for single digit numbers.

I 2 of 5 Check Digit Verification

Parameter Number	Value	description
49	*0	Disable
	1	USS Check Digit
	2	OPCC Check Digit

When enabled, this parameter checks the integrity of an I 2 of 5 symbol to ensure it complies with a specified algorithm, either USS (Uniform Symbology Specification), or OPCC (Optical Product Code Council).

Transmit I 2 of 5 Check Digit

Parameter Number	Value	description

44	*0	Do Not Transmit I 2 of 5 Check Digit (Disable)
	1	Transmit I 2 of 5 Check Digit (Enable)

Scan the appropriate bar code below to transmit I 2 of 5 data with or without the check digit.

Convert I 2 of 5 to EAN-13

Parameter Number	Value	description
82	*0	Do Not Convert I 2 of 5 to EAN-13 (Disable)
	1	Convert I 2 of 5 to EAN-13 (Enable)

Enable this parameter to convert 14-character I 2 of 5 codes to EAN-13, and transmit to the host as EAN-13. To accomplish this, the I 2 of 5 code must be enabled, and the code must have a leading zero and a valid EAN-13 check digit.

Discrete 2 of 5

Parameter Number	Value	description
5	*0	Disable Discrete 2 of 5
	1	Enable Discrete 2 of 5

Set Lengths for Discrete 2 of 5

Parameter Number	Value	description
20	One Length: 12	Length1
21		Length2

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Lengths for D 2 of 5 may be set for any length, one or two discrete lengths, or lengths within a specific range. To set lengths via serial commands, see Setting Code Lengths Via Serial Commands.

Codabar

Parameter Number	Value	description
7	*0	Disable Codabar
	1	Enable Codabar

Set Lengths for Codabar

Parameter Number	Value	description
24	Length within Range:	Length1
25	5 - 55	Length2

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Lengths for Codabar may be set for any length, one or two discrete lengths, or lengths within a specific range. To set lengths via serial commands, see Setting Code Lengths Via Serial Commands.

CLSI Editing

Parameter Number	Value	description
54	*0	Disable CLSI Editing
	1	Enable CLSI Editing

When enabled, this parameter strips the start and stop characters and inserts a space after the first, fifth, and tenth characters of a 14-character Codabar symbol.

NOTE Symbol length does not include start and stop characters

NOTIS Editing

Parameter Number	Value	description
55	*0	Disable NOTIS Editing
	1	Enable NOTIS Editing

When enabled, this parameter strips the start and stop characters from decoded Codabar symbol

MSI

Parameter Number	Value	description
11	*0	Disable MSI
	1	Enable MSI

Set Lengths for MSI

Parameter Number	Value	description
30	Length within Range:	Length1
31	6 - 55	Length2

The length of a code refers to the number of characters (i.e., human readable characters) the code contains, and includes check digits. Lengths for MSI can be set for any length, one or two discrete lengths, or lengths within a specific range.

MSI Check Digits

Parameter Number	Value	description
50	*0	One MSI Check Digit
	1	Two MSI Check Digits

These check digits at the end of the bar code verify the integrity of the data. At least one check digit is always required. Check digits are not automatically transmitted with the data. If two check digits are selected, also select an MSI Check Digit Algorithm

Transmit MSI Check Digit

Parameter Number	Value	description
46	*0	Do Not Transmit MSI Check Digit(s) (Disable)
	1	Transmit MSI Check Digit(s) (Enable)

MSI Check Digit Algorithm

Parameter Number	Value	description
51	*1	MOD 10/MOD 10
	0	MOD 10/MOD 11

Two algorithms are possible for the verification of the second MSI check digit. Select the bar code below corresponding to the algorithm used to encode the check digit.

Transmit Code ID Character

Parameter Number	Value	description
45	*0	None
	1	AIM Code ID Character

	2	Symbol Code ID Character

A code ID character identifies the code type of a scanned bar code. This can be useful when decoding more than one code type. The code ID character is inserted between the prefix character (if selected) and the decoded symbol.

Select no code ID character, a Symbol Code ID character, or an AIM Code ID character. For Symbol and AIM code ID characters

b. 2D Symbology Parameter

In this section, * indicates the default option.

UPC-A

Enable or disable UPC-A

Parameter Number	Value	description
1	*1	Enable UPC-A
	0	Disable UPC-A

UPC-E

Enable or disable UPC-E

Parameter Number	Value	description
2	*1	Enable UPC-E
	0	Disable UPC-E

UPC-E1

Enable or disable UPC-E1. UPC-E1 is disabled by default

Parameter Number	Value	description
12	*0	Enable UPC-E1
	1	Disable UPC-E1

NOTE UPC-E1 is not a UCC (Uniform Code Council) approved symbology.

EAN-8/JAN 8

Enable or disable EAN-8/JAN-8

Parameter Number	Value	description
4	*1	Enable EAN-8/JAN-8
	0	Disable EAN-8/JAN-8

EAN-13/JAN 13

Enable or disable EAN-13/JAN-13

Parameter Number	Value	description

3	*1	Enable EAN-13/JAN-13
	0	Disable EAN-13/JAN-13

Bookland EAN

Enable or disable Bookland EAN:

Parameter Number	Value	description
83	*0	Disable Bookland EAN
	1	Enable Bookland EAN

NOTE If you enable Bookland EAN, select a Bookland ISBN Format. Also select either Decode UPC/EAN Supplementals, Autodiscriminate UPC/EAN Supplementals, or Enable 978/979 Supplemental Mode in Decode UPC/EAN/JAN Supplementals.

Decode UPC/EAN/JAN Supplementals

Supplementals are bar codes appended according to specific format conventions (e.g., UPC A+2, UPC E+2, EAN 13+2). Select one of the following options

Parameter Number	Value	description
16	*0	Ignore UPC/EAN with Supplementals
	1	Decode UPC/EAN with Supplementals
	2	Autodiscriminate UPC/EAN Supplementals
	4	Enable 378/379 Supplemental Mode
	5	Enable 978/979 Supplemental Mode
	7	Enable 977 Supplemental Mode
	6	Enable 414/419/434/439 Supplemental Mode
	8	Enable 491 Supplemental Mode
	3	Enable Smart Supplemental Mode
	9	Supplemental User-Programmable Type 1
	10	Supplemental User-Programmable Type 1 and 2
	11	Smart Supplemental Plus User-Programmable 1
	12	Smart Supplemental Plus User-Programmable 1 and 2

User-Programmable Supplementals

If you selected a Supplemental User-Programmable option from Decode UPC/EAN/JAN Supplementals, select User-Programmable Supplemental 1 to set the 3-digit prefix. Select User-Programmable Supplemental 2 to set a second 3-digit prefix.

Parameter Number	Value	description
579		Supplemental 1
580		Supplemental 2

UPC/EAN/JAN Supplemental Redundancy

With Autodiscriminate UPC/EAN/JAN Supplementals selected, this option adjusts the number of times a symbol without supplementals is decoded before transmission. The range is from two to 30 times. Five or above is recommended when decoding a mix of UPC/EAN/JAN symbols with and without supplementals, and the autodiscriminate option is selected. The default is set at 10.

Parameter Number	Value	description
80	*10	

UPC/EAN/JAN Supplemental AIM ID Format

Select an output format when reporting UPC/EAN/JAN bar codes with Supplementals with Transmit Code ID Character set to AIM Code ID Character

Parameter Number	Value	description
672	*1	Combined
	0	Separate
	2	Separate

UPC Reduced Quiet Zone

Enable or disable decoding UPC bar codes with reduced quiet zones. If you select Enable, select a 1D Quiet Zone Level.

Parameter Number	Value	description
1289	*0	Disable UPC Reduced Quiet Zone
	1	Enable UPC Reduced Quiet Zone

Transmit UPC-A Check Digit

The check digit is the last character of the symbol used to verify the integrity of the data. Select whether to transmit the bar code data with or without the UPC-A check digit. It is always verified to guarantee the integrity of the data.

Parameter Number	Value	description
40	*1	Transmit UPC-A Check Digit

	0	Do Not Transmit UPC-A Check Digit

Transmit UPC-E Check Digit

The check digit is the last character of the symbol used to verify the integrity of the data. Select whether to transmit the bar code data with or without the UPC-E check digit. It is always verified to guarantee the integrity of the data.

Parameter Number	Value	description
41	*1	Transmit UPC-E Check Digit
	0	Do Not Transmit UPC-E Check Digit

Transmit UPC-E1 Check Digit

The check digit is the last character of the symbol used to verify the integrity of the data. Select whether to transmit the bar code data with or without the UPC-E1 check digit. It is always verified to guarantee the integrity of the data.

Parameter Number	Value	description
42	*1	Transmit UPC-E1 Check Digit
	0	Do Not Transmit UPC-E1 Check Digit

UPC-A Preamble

Preamble characters are part of the UPC symbol, and include Country Code and System Character. There are three options for transmitting a UPC-A preamble to the host device. Select the appropriate option to match the host system

Parameter Number	Value	description
34	*1	Transmit System Character Only
	2	Transmit System Character and Country Code
	0	Transmit no preamble

UPC-E Preamble

Preamble characters are part of the UPC symbol, and include Country Code and System Character. There are three options for transmitting a UPC-E preamble to the host device. Select the appropriate option to match the host system

Parameter Number	Value	description
35	*1	Transmit System Character Only
	2	Transmit System Character and Country Code

	0	Transmit no preamble

UPC-E1 Preamble

Preamble characters are part of the UPC symbol, and include Country Code and System Character. There are three options for transmitting a UPC-E1 preamble to the host device. Select the appropriate option to match the host system.

Parameter Number	Value	description
36	*1	Transmit System Character Only
	2	Transmit System Character and Country Code
	0	Transmit no preamble

Convert UPC-E to UPC-A

Enable this to convert UPC-E (zero suppressed) decoded data to UPC-A format before transmission. After conversion, the data follows UPC-A format and is affected by UPC-A programming selections (e.g., Preamble, Check Digit). When disabled, UPC-E decoded data is transmitted as UPC-E data, without conversion.

Parameter Number	Value	description
37	*0	Do Not Convert UPC-E to UPC-A (Disable)
	1	Convert UPC-E to UPC-A (Enable)

Convert UPC-E1 to UPC-A

Enable this to convert UPC-E1 decoded data to UPC-A format before transmission. After conversion, the data follows UPC-A format and is affected by UPC-A programming selections (e.g., Preamble, Check Digit). When disabled, UPC-E1 decoded data is transmitted as UPC-E1 data, without conversion.

Parameter Number	Value	description
38	*0	Do Not Convert UPC-E to UPC-A (Disable)
	1	Convert UPC-E to UPC-A (Enable)

EAN-8/JAN-8 Extend

Enable this parameter to add five leading zeros to decoded EAN-8 symbols to make them compatible in format to EAN-13 symbols. Disable this to transmit EAN-8 symbols as is.

Parameter Number	Value	description
39	*0	Disable EAN/JAN Zero Extend

	1	Enable EAN/JAN Zero Extend

Bookland ISBN Format

If you enabled Bookland EAN using Enable/Disable Bookland EAN, select one of the following formats for Bookland data

Parameter Number	Value	description
576	*0	Bookland ISBN-10
	1	Bookland ISBN-13

NOTE For Bookland EAN to function properly, first enable Bookland EAN using Enable/Disable Bookland EAN, then select either Decode UPC/EAN Supplementals, Autodiscriminate UPC/EAN Supplementals, or Enable 978/979 Supplemental Mode in Decode UPC/EAN/JAN Supplementals.

UCC Coupon Extended Code

Enable this parameter to decode UPC-A bar codes starting with digit '5', EAN-13 bar codes starting with digit '99', and UPC-A/EAN-128 Coupon Codes. UPCA, EAN-13, and EAN-128 must be enabled to scan all types of Coupon Codes.

Parameter Number	Value	description
85	*0	Disable UCC Coupon Extended Code
	1	Enable UCC Coupon Extended Code

NOTE Use the Decode UPC/EAN Supplemental Redundancy parameter to control autodiscrimination of the EAN128 (right half) of a coupon code.

Coupon Report

Traditional coupon symbols (old coupon symbols) are composed of two bar codes: UPC/EAN and Code128. A new coupon symbol is composed of a single Databar Expanded bar code. The new coupon format offers more options for purchase values (up to $999.99) and supports complex discount offers such as a second purchase requirement.

An interim coupon symbol also exists that contains both types of bar codes: UPC/EAN and Databar Expanded. This format accommodates both retailers that do not recognize or use the additional information included in the new coupon symbol, as well as those who can process new coupon symbols.

Select one of the following options for decoding coupon symbols

Parameter Number	Value	description
730	*1	New Coupon Symbols
	0	Old Coupon Symbols
	2	Both Coupon Formats

ISSN EAN

Enable or disable ISSN EAN.

Parameter Number	Value	description
617	*0	Disable ISSN EAN
	1	Enable ISSN EAN

Code 128

Enable or disable Code 128

Parameter Number	Value	description
8	*0	Enable Code 128
	1	Disable Code 128

Set Lengths for Code 128

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Assign lengths for Code 128 to decode either one or two discrete lengths, or a length within a specific range.

Parameter Number	Value	description
209	Any Length	Length1 [Range: 0..55]
210		Length2 [Range: 0..55]

One Discrete Length - To limit the decoding of Code 128 to one specific length, assign this length to the Length1 parameter and 0 to the Length2 parameter. For example, for fixed length 14, set Length1 = 14, Length2 = 0.

Two Discrete Lengths - To limit the decoding of Code 128 to either of two specific lengths, assign the greater length to the Length1 parameter and the lesser to Length2. For example, to decode Code 128 codes of either 2 or 14 characters only, set Length1 = 14, Length2 = 2.

Length Within Range - To decode only Code 128 codes that fall within a specific length range, assign the lesser length to the Length1 parameter and the greater to the Length2 parameter. For example, to decode Code 128 codes of length 4 through 12 characters, set Length1 = 4, Length2 = 12.

GS1-128 (formerly UCC/EAN-128)

Enable or disable GS1-128

Parameter Number	Value	description
14	*1	Enable GS1-128
	0	Disable GS1-128

ISBT 128

ISBT 128 is a variant of Code 128 used in the blood bank industry. Enable or disable ISBT 128. If necessary, the host must perform concatenation of the ISBT data.

Parameter Number	Value	description
84	*1	Enable ISBT 128
	0	Disable ISBT 128

ISBT Concatenation

Select an option for concatenating pairs of ISBT code types

Parameter Number	Value	description
577	*0 - Disable ISBT Concatenation	The device does not concatenate pairs of ISBT codes it encounters.
	1 - Enable ISBT Concatenation	There must be two ISBT codes in order for the device to decode and perform concatenation. The device does not decode single ISBT symbols.
	2 - Autodiscriminate ISBT Concatenation	The device decodes and concatenates pairs of ISBT codes immediately. If only a single ISBT symbol is present, the device must decode the symbol the number of times set

		via ISBT Concatenation Redundancy before transmitting its data to confirm that there is no additional ISBT symbol.

Check ISBT Table

The ISBT specification includes a table that lists several types of ISBT bar codes that are commonly used in pairs. If you enable ISBT Concatenation, enable Check ISBT Table to concatenate only those pairs found in this table. Other types of ISBT codes are not concatenated.

Parameter Number	Value	description
578	*1	Enable Check ISBT Table
	0	Disable Check ISBT Table

ISBT Concatenation Redundancy

With ISBT Concatenation set to Autodiscriminate, this option sets the number of times the device must decode an ISBT symbol before determining that there is no additional symbol. The range is from two to 20 times. The default is 10.

Parameter Number	Value	description
223	*10	

Code 128 Reduced Quiet Zone

Enable or disable decoding Code 128 bar codes with reduced quiet zones. If you select Enable, select a 1D Quiet Zone Level.

Parameter Number	Value	description
1208	*0	Disable Code 128 Reduced Quiet Zone
	1	Enable Code 128 Reduced Quiet Zone

Ignore Code 128<FNC4>

This feature applies to Code 128 bar codes with an embedded <FNC4> character.

Parameter Number	Value	description
1254	*0 - Disable Ignore Code 128<FNC4>	The <FNC4> character is not transmitted but the following character has 128 added to it.
	1 - Enable Ignore Code 128<FNC4>	This strips the <FNC4> character

		from the decode data. The remaining characters do not change.

Code 39

Enable or disable Code 39

Parameter Number	Value	description
0	*1	Enable Code 39
	0	Disable Code 39

Trioptic Code 39

Trioptic Code 39 is a variant of Code 39 used in the marking of computer tape cartridges. Trioptic Code 39 symbols always contain six characters. Enable or disable Trioptic Code 39.

Parameter Number	Value	description
13	*0	Disable Code 39
	1	Enable Code 39

NOTE Trioptic Code 39 and Code 39 Full ASCII cannot be enabled simultaneously

Convert Code 39 to Code 32

Code 32 is a variant of Code 39 used by the Italian pharmaceutical industry. Enable or disable converting Code 39 to Code 32.

Parameter Number	Value	description
86	*0	Disable Convert Code 39 to Code 32
	1	Enable Convert Code 39 to Code 32

NOTE Code 39 must be enabled for this parameter to function.

Code 32 Prefix

Enable or disable adding the prefix character "A" to all Code 32 bar codes.

Parameter Number	Value	description
231	*0	Disable Code 32 Prefix
	1	Enable Code 32 Prefix

NOTE Convert Code 39 to Code 32 must be enabled for this parameter to function.

Set Lengths for Code 39

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Assign lengths for Code 39 to decode either one or two discrete lengths, or a length within a specific range.

Parameter Number	Value	description
18	*2	Length1 [Range: 0..55]
19	*55	Length2 [Range: 0..55]

One Discrete Length - To limit the decoding of Code 39 to one specific length, assign this length to the Length1 parameter and 0 to the Length2 parameter. For example, for fixed length 14, set Length1 = 14, Length2 = 0.

Two Discrete Lengths - To limit the decoding of Code 39 to either of two specific lengths, assign the greater length to the Length1 parameter and the lesser to Length2. For example, to decode Code 39 codes of either 2 or 14 characters only, set Length1 = 14, Length2 = 2.

Length Within Range - To decode only Code 39 codes that fall within a specific length range, assign the lesser length to the Length1 parameter and the greater to the Length2 parameter. For example, to decode Code 39 codes of length 4 through 12 characters, set Length1 = 4, Length2 = 12.

Code 39 Check Digit Verification

Enable this to check the integrity of all Code 39 symbols to verify that the data complies with specified check digit algorithm. Only Code 39 symbols which include a modulo 43 check digit are decoded. Enable this feature if the Code 39 symbols contain a Modulo 43 check digit.

Parameter Number	Value	description
48	*0	Disable Code 39 Check Digit Verification
	1	Enable Code 39 Check Digit Verification

Transmit Code 39 Check Digit

Select whether to transmit Code 39 data with or without the check digit.

Parameter Number	Value	description

43	*0	Do Not Transmit Code 39 Check Digit (Disable)
	1	Transmit Code 39 Check Digit (Enable)

NOTE Code 39 Check Digit Verification must be enabled for this parameter to function.

Code 39 Full ASCII Conversion

Code 39 Full ASCII is a variant of Code 39 which pairs characters to encode the full ASCII character set. Enable or disable Code 39 Full ASCII

Parameter Number	Value	description
17	*0	Disable Code 39 Full ASCII
	1	Enable Code 39 Full ASCII

NOTE Trioptic Code 39 and Code 39 Full ASCII cannot be enabled simultaneously. Code 39 Full ASCII to Full ASCII Correlation is host-dependent.

Code 39 Reduced Quiet Zone

Enable or disable decoding Code 39 bar codes with reduced quiet zones. If you select Enable, select a 1D Quiet Zone Level.

Parameter Number	Value	description
1209	*0	Disable Code 39 Reduced Quiet Zone
	1	Enable Code 39 Reduced Quiet Zone

Code 93

Enable or disable Code 93

Parameter Number	Value	description
9	*0	Disable Code 93
	1	Enable Code 93

Set Lengths for Code 93

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Assign lengths for Code 93 to decode either one or two discrete lengths, or a length within a specific range.

Parameter Number	Value	description

26	*4	Length1 [Range: 0..55]
27	*55	Length2 [Range: 0..55]

One Discrete Length - To limit the decoding of Code 93 to one specific length, assign this length to the Length1 parameter and 0 to the Length2 parameter. For example, for fixed length 14, set Length1 = 14, Length2 = 0.

Two Discrete Lengths - To limit the decoding of Code 93 to either of two specific lengths, assign the greater length to the Length1 parameter and the lesser to Length2. For example, to decode Code 93 codes of either 2 or 14 characters only, set Length1 = 14, Length2 = 2.

Length Within Range - To decode only Code 93 codes that fall within a specific length range, assign the lesser length to the Length1 parameter and the greater to the Length2 parameter. For example, to decode Code 93 codes of length 4 through 12 characters, set Length1 = 4, Length2 = 12.

Code 11

Enable or disable Code 11

Parameter Number	Value	description
10	*0	Disable Code 11
	1	Enable Code 11

Set Lengths for Code 11

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Assign lengths for Code 11 to decode either one or two discrete lengths, or a length within a specific range.

Parameter Number	Value	description
28	*4	Length1 [Range: 0..55]
29	*55	Length2 [Range: 0..55]

One Discrete Length - To limit the decoding of Code 11 to one specific length, assign this length to the Length1 parameter and 0 to the Length2 parameter. For example, for fixed length 14, set Length1 = 14, Length2 = 0.

Two Discrete Lengths - To limit the decoding of Code 11 to either of two specific lengths, assign the greater length to the Length1 parameter and the lesser to Length2. For example, to decode

Code 11 codes of either 2 or 14 characters only, set Length1 = 14, Length2 = 2.

Length Within Range - To decode only Code 11 codes that fall within a specific length range, assign the lesser length to the Length1 parameter and the greater to the Length2 parameter. For example, to decode Code 11 codes of length 4 through 12 characters, set Length1 = 4, Length2 = 12.

Code 11 Check Digit Verification

This feature allows the decoder to check the integrity of all Code 11 symbols to verify that the data complies with the specified check digit algorithm. This selects the check digit mechanism for the decoded Code 11 bar code. To enable this feature, set the number of check digits encoded in the Code 11 symbols

Parameter Number	Value	description
52	*0	Disable Code 11 Check Digit Verification
	1 - 1	Check Digit
	2 - 2	Check Digits

Transmit Code 11 Check Digits

Select whether or not to transmit the Code 11 check digit(s).

Parameter Number	Value	description
47	*0	Do Not Transmit Code 11 Check Digit(s) (Disable)
	1	Transmit Code 11 Check Digit(s) (Enable)

NOTE Code 11 Check Digit Verification must be enabled for this parameter to function.

Interleaved 2 of 5

Enable or disable Interleaved 2 of 5

Parameter Number	Value	description
6	*1	Enable Interleaved 2 of 5
	0	Disable Interleaved 2 of 5

Set Lengths for Interleaved 2 of 5

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Assign lengths for I 2 of 5 to decode either one or two discrete lengths, or a length within a specific range.

Parameter Number	Value	description
22	*14	Length1 [Range: 0..55]
23	*0	Length2 [Range: 0..55]

One Discrete Length - To limit the decoding of I 2 of 5 to one specific length, assign this length to the Length1 parameter and 0 to the Length2 parameter. For example, for fixed length 14, set Length1 = 14, Length2 = 0.

Two Discrete Lengths - To limit the decoding of I 2 of 5 to either of two specific lengths, assign the greater length to the Length1 parameter and the lesser to Length2. For example, to decode I 2 of 5 codes of either 2 or 14 characters only, set Length1 = 14, Length2 = 2.

Length Within Range - To decode only I 2 of 5 codes that fall within a specific length range, assign the lesser length to the Length1 parameter and the greater to the Length2 parameter. For example, to decode I 2 of 5 codes of length 4 through 12 characters, set Length1 = 4, Length2 = 12.

Any Length - To decode I 2 of 5 codes of any length, set the values of Length1 and Length2 parameters to 0.

NOTE Due to the construction of the I 2 of 5 symbology, it is possible for a scan line covering only a portion of the code to be interpreted as a complete scan, yielding less data than is encoded in the bar code. To prevent this, select specific lengths (one or two discrete lengths) for I 2 of 5 applications.

I 2 of 5 Check Digit Verification

Enable this feature to check the integrity of all I 2 of 5 symbols to verify the data complies with either the specified Uniform Symbology Specification (USS), or the Optical Product Code Council (OPCC) check digit algorithm.

Parameter Number	Value	description
49	*0	Disable
	1	USS Check Digit
	2	OPCC Check Digits

Transmit I 2 of 5 Check Digit

Select whether to transmit I 2 of 5 data with or without the check digit

Parameter Number	Value	description
44	*0	Do Not Transmit I 2 of 5 Check Digit (Disable)
	1	Transmit I 2 of 5 Check Digit (Enable)

Convert I 2 of 5 to EAN-13

Enable this parameter to convert 14-character I 2 of 5 codes to EAN-13, and transmit to the host as EAN-13. To accomplish this, the I 2 of 5 code must be enabled, and the code must have a leading zero and a valid EAN-13 check digit.

Parameter Number	Value	description
82	*0	Do Not Convert I 2 of 5 to EAN-13 (Disable)
	1	Convert I 2 of 5 to EAN-13 (Enable)

I 2 of 5 Security Level

Interleaved 2 of 5 bar codes are vulnerable to misdecodes, particularly when I 2 of 5 Lengths is set to Any Length. The scanner offers four levels of decode security for Interleaved 2 of 5 bar codes. There is an inverse relationship between security and scanner aggressiveness. Increasing the level of security can reduce scanning aggressiveness, so select only the level of security necessary

Parameter Number	Value	description
1121	*1	I 2 of 5 Security Level 1
	0	I 2 of 5 Security Level 0
	2	I 2 of 5 Security Level 2
	3	I 2 of 5 Security Level 3

NOTE Selecting this option is an extreme measure against mis-decoding severely out-of-spec bar codes. Selecting this level of security significantly impairs the decoding ability of the scanner. If this level of security is required, try to improve the quality of the bar codes.

I 2 of 5 Reduced Quiet Zone

Enable or disable decoding I 2 of 5 bar codes with reduced quiet zones. If you select Enable, select a 1D Quiet Zone Level.

Parameter Number	Value	description
82	*0	Disable I 2 of 5 Reduced Quiet Zone
	1	Enable I 2 of 5 Reduced Quiet Zone

Discrete 2 of 5

Enable or disable Discrete 2 of 5

Parameter Number	Value	description
5	*0	Disable Discrete 2 of 5
	1	Enable Discrete 2 of 5

Set Lengths for Discrete 2 of 5

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Assign lengths for D 2 of 5 to decode either one or two discrete lengths, or a length within a specific range.

Parameter Number	Value	description
20	*12	Length1 [Range: 0..55]
21	*0	Length2 [Range: 0..55]

One Discrete Length - To limit the decoding of D 2 of 5 to one specific length, assign this length to the Length1 parameter and 0 to the Length2 parameter. For example, for fixed length 14, set Length1 = 14, Length2 = 0.

Two Discrete Lengths - To limit the decoding of D 2 of 5 to either of two specific lengths, assign the greater length to the Length1 parameter and the lesser to Length2. For example, to decode D 2 of 5 codes of either 2 or 14 characters only, set Length1 = 14, Length2 = 2.

Length Within Range - To decode only D 2 of 5 codes that fall within a specific length range, assign the lesser length to the Length1 parameter and the greater to the Length2 parameter. For example, to decode D 2 of 5 codes of length 4 through 12 characters, set Length1 = 4, Length2 = 12

NOTE Due to the construction of the D 2 of 5 symbology, it is possible for a scan line covering only a portion of the code to be interpreted as a complete scan, yielding less data than is encoded in the bar code. To prevent this, select specific lengths (one or two discrete lengths) for D 2 of 5 applications.

Codabar

Enable or disable Codabar

Parameter Number	Value	description
7	*0	Disable Codabar
	1	Enable Codabar

Set Lengths for Codabar

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Assign lengths for Codabar to decode either one or two discrete lengths, or a length within a specific range.

Parameter Number	Value	description
24	*5	Length1 [Range: 0..55]
25	*55	Length2 [Range: 0..55]

One Discrete Length - To limit the decoding of Codabar to one specific length, assign this length to the Length1 parameter and 0 to the Length2 parameter. For example, for fixed length 14, set Length1 = 14, Length2 = 0.

Two Discrete Lengths - To limit the decoding of Codabar to either of two specific lengths, assign the greater length to the Length1 parameter and the lesser to Length2. For example, to decode Codabar codes of either 2 or 14 characters only, set Length1 = 14, Length2 = 2.

Length Within Range - To decode only Codabar codes that fall within a specific length range, assign the lesser length to the Length1 parameter and the greater to the Length2 parameter. For example, to decode Codabar codes of length 4 through 12 characters, set Length1 = 4, Length2 = 12.

CLSI Editing

Enable this parameter to strip the start and stop characters and insert a space after the first, fifth, and tenth characters of a 14-character Codabar symbol. Enable this if the host system requires this data format.

Parameter Number	Value	description
54	*0	Disable CLSI Editing
	1	Enable CLSI Editing

NOTE Symbol length does not include start and stop characters.

NOTIS Editing

Enable this parameter to strip the start and stop characters from a decoded Codabar symbol. Enable this if the host system requires this data format.

Parameter Number	Value	description
55	*0	Disable NOTIS Editing
	1	Enable NOTIS Editing

MSI

Enable or disable MSI.

Parameter Number	Value	description
11	*0	Disable MSI
	1	Enable MSI

Set Lengths for MSI

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Assign lengths for MSI to decode either one or two discrete lengths, or a length within a specific range.

Parameter Number	Value	description
30	*4	Length1 [Range: 0..55]
31	*55	Length2 [Range: 0..55]

One Discrete Length - To limit the decoding of MSI to one specific length, assign this length to the Length1 parameter and 0 to the Length2 parameter. For example, for fixed length 14, set Length1 = 14, Length2 = 0.

Two Discrete Lengths - To limit the decoding of MSI to either of two specific lengths, assign the greater length to the Length1 parameter and the lesser to Length2. For example, to decode MSI codes of either 2 or 14 characters only, set Length1 = 14, Length2 = 2.

Length Within Range - To decode only MSI codes that fall within a specific length range, assign the lesser length to the Length1 parameter and the greater to the Length2 parameter. For example, to decode MSI codes of length 4 through 12 characters, set Length1 = 4, Length2 = 12.

NOTE Due to the construction of the MSI symbology, it is possible for a scan line covering only a portion of the code to be interpreted as a complete scan, yielding less data than is encoded in the bar code. To prevent this, select specific lengths (one or two discrete lengths) for MSI

applications.

MSI Check Digits

With MSI symbols, one check digit is mandatory and always verified by the reader. The second check digit is optional. If the MSI codes include two check digits, select Two MSI Check Digits to enable verification of the second check digit

Parameter Number	Value	description
50	*0	One MSI Check Digit
	1	Two MSI Check Digits

Transmit MSI Check Digit(s)

Select whether to transmit MSI data with or without the check digit.

Parameter Number	Value	description
46	*0	Do Not Transmit MSI Check Digit(s) (Disable)
	1	Transmit MSI Check Digit(s) (Enable)

MSI Check Digit Algorithm

Select one of two algorithms for the verification of the second MSI check digit

Parameter Number	Value	description
51	*1	MOD 10/MOD 10
	0	MOD 10/MOD 11

Chinese 2 of 5

Enable or disable Chinese 2 of 5

Parameter Number	Value	description
408	*0	Disable Chinese 2 of 5
	1	Enable Chinese 2 of 5

Korean 3 of 5

Enable or disable Korean 3 of 5

Parameter Number	Value	description
581	*0	Disable Korean 3 of 5
	1	Enable Korean 3 of 5

NOTE The length for Korean 3 of 5 is fixed at 6.

Matrix 2 of 5

Enable or disable Matrix 2 of 5.

Parameter Number	Value	description
618	*0	Disable Matrix 2 of 5
	1	Enable Matrix 2 of 5

Set Lengths for Matrix 2 of 5

The length of a code refers to the number of characters (i.e., human readable characters), including check digit(s) the code contains. Assign lengths for Matrix 2 of 5 to decode either one or two discrete lengths, or a length within a specific range.

Parameter Number	Value	description
619	*14	Length1 [Range: 0..55]
620	*0	Length2 [Range: 0..55]

One Discrete Length - To limit the decoding of Matrix 2 of 5 to one specific length, assign this length to the Length1 parameter and 0 to the Length2 parameter. For example, for fixed length 14, set Length1 = 14, Length2 = 0.

Two Discrete Lengths - To limit the decoding of Matrix 2 of 5 to either of two specific lengths, assign the greater length to the Length1 parameter and the lesser to Length2. For example, to decode Matrix 2 of 5 codes of either 2 or 14 characters only, set Length1 = 14, Length2 = 2.

Length Within Range - To decode only Matrix 2 of 5 codes that fall within a specific length range, assign the lesser length to the Length1 parameter and the greater to the Length2 parameter. For example, to decode Matrix 2 of 5 codes of length 4 through 12 characters, set Length1 = 4, Length2 = 12

Matrix 2 of 5 Redundancy

Enable or disable Matrix 2 of 5 redundancy

Parameter Number	Value	description
621	*0	Disable Matrix 2 of 5 Redundancy
	1	Enable Matrix 2 of 5 Redundancy

Matrix 2 of 5 Check Digit

The check digit is the last character of the symbol used to verify the integrity of the data. Select whether to transmit the bar code data with or without the Matrix 2 of 5 check digit

Parameter Number	Value	description
622	*0	Disable Matrix 2 of 5 Check Digit
	1	Enable Matrix 2 of 5 Check Digit

Transmit Matrix 2 of 5 Check Digit

Select whether to transmit Matrix 2 of 5 data with or without the check digit.

Parameter Number	Value	description
623	*0	Do Not Transmit Matrix 2 of 5 Check Digit
	1	Transmit Matrix 2 of 5 Check Digit

Inverse 1D

Set the 1D inverse decoder setting

Parameter Number	Value	description
586	*0	Regular Only
	1	Inverse Only
	2	nverse Autodetect

US Postnet

Enable or disable US Postnet

Parameter Number	Value	description
89	*1	Enable US Postnet
	0	Disable US Postnet

US Planet

Enable or disable US Planet

Parameter Number	Value	description
90	*1	Enable US Planet
	0	Disable US Planet

Transmit US Postal Check Digit

Select whether to transmit US Postal data, which includes both US Postnet and US Planet, with or without the check digit

Parameter Number	Value	description
95	*1	Transmit US Postal Check Digit
	0	Do Not Transmit US Postal Check Digit

UK Postal

Enable or disable UK Postal

Parameter Number	Value	description
91	*1	Enable UK Postal
	0	Disable UK Postal

Transmit UK Postal Check Digit

Select whether to transmit UK Postal data with or without the check digit

Parameter Number	Value	description
96	*1	Transmit UK Postal Check Digit
	0	Do Not Transmit UK Postal Check Digit

Japan Postal

Enable or disable Japan Postal

Parameter Number	Value	description
290	*1	Enable Japan Postal
	0	Disable Japan Postal

Australia Post

Enable or disable Australia Post

Parameter Number	Value	description
291	*1	Enable Australia Post
	0	Disable Australia Post

Australia Post Format

Select one of the following formats for Australia Post

Parameter Number	Value	description
718	*0	Autodiscriminate
	1	Raw Format
	2	Alphanumeric Encoding
	3	Numeric Encoding

Netherlands KIX Code

Enable or disable Netherlands KIX Code

Parameter Number	Value	description
326	*1	Enable Netherlands KIX Code
	0	Disable Netherlands KIX Code

USPS 4CB/One Code/Intelligent Mail

Enable or disable USPS 4CB/One Code/Intelligent Mail

Parameter Number	Value	description
592	*0	Disable USPS 4CB/One Code/Intelligent Mail
	1	Enable USPS 4CB/One Code/Intelligent Mail

UPU FICS Postal

Enable or disable UPU FICS Postal

Parameter Number	Value	description
611	*0	Disable UPU FICS Postal
	1	Enable UPU FICS Postal

GS1 DataBar-14

Enable or disable GS1 DataBar-14

Parameter Number	Value	description
338	*1	Enable GS1 DataBar-14
	0	Disable GS1 DataBar-14

GS1 DataBar Limited

Enable or disable GS1 DataBar Limited

Parameter Number	Value	description
339	*0	Disable GS1 DataBar Limited
	1	Enable GS1 DataBar Limited

GS1 DataBar Limited Security Level

There are four levels of decode security for GS1 DataBar Limited bar codes. There is an inverse relationship between security and scanner aggressiveness. Increasing the level of security may result in reduced aggressiveness in scanning, so only choose the level of security necessary.

Parameter Number	Value	description
728	*3	Security level reflects newly proposed GS1 standard that requires a 5X trailing clear margin
	1	No clear margin required. This complies with the original GS1 standard, yet might result in erroneous decoding of the DataBar Limited bar codes when scanning some UPC symbols that start with the digits "9" and "7".
	2	Automatic risk detection. This level of security may result in erroneous decoding of DataBar Limited bar codes when scanning some UPC symbols. If a misdecode is detected, the scanner operates in Level 3 or Level 1.
	4	Security level extends beyond the standard required by GS1. This level of security requires a 5X leading and trailing clear margin.

GS1 DataBar Expanded

Enable or disable GS1 DataBar Expanded

Parameter Number	Value	description
340	*0	Disable GS1 DataBar Expanded
	1	Enable GS1 DataBar Expanded

Convert GS1 DataBar to UPC/EAN

This parameter only applies to GS1 DataBar-14 and GS1 DataBar Limited symbols not decoded as part of a Composite symbol. Enable this to strip the leading 010 from GS1 DataBar-14 and GS1 DataBar Limited symbols encoding a single zero as the first digit, and report the bar code as EAN-13.

For bar codes beginning with two or more zeros but not six zeros, this parameter strips the leading'0100 and reports the bar code as UPC-A. The UPC-A Preamble parameter that transmits the system character and country code applies to converted bar codes. Note that neither the system character nor the check digit can be stripped.

Parameter Number	Value	description
397	*0	Disable Convert GS1 DataBar to UPC/EAN
	1	Enable Convert GS1 DataBar to UPC/EAN

Composite CC-C

Enable or disable Composite bar codes of type CC-C

Parameter Number	Value	description
341	*0	Disable CC-C
	1	Enable CC-C

Composite CC-A/B

Enable or disable Composite bar codes of type CC-A/B

Parameter Number	Value	description
342	*0	Disable CC-A/B
	1	Enable CC-A/B

Composite TLC-39

Enable or disable Composite bar codes of type TLC-39

Parameter Number	Value	description
371	*0	Disable TLC39
	1	Enable TLC39

UPC Composite Mode

Select an option for linking UPC symbols with a 2D symbol during transmission as if they were one symbol

Parameter Number	Value	description
344	*0	UPC Never Linked
	1	UPC Always Linked
	2	Autodiscriminate UPC Composites

GS1-128 Emulation Mode for UCC/EAN Composite Codes

Enable or disable this mode

Parameter Number	Value	description
371	*0	Disable GS1-128 Emulation Mode for UCC/EAN Composite Codes
	1	Enable GS1-128 Emulation Mode for UCC/EAN Composite Codes

PDF417

Enable or disable PDF417

Parameter Number	Value	description
15	*1	Enable PDF417
	0	Disable PDF417

MicroPDF417

Enable or disable MicroPDF417.

Parameter Number	Value	description
227	*0	Disable MicroPDF417
	1	Enable MicroPDF417

Code 128 Emulation

Enable this parameter to transmit data from certain MicroPDF417 symbols as if it was encoded in Code 128 symbols. Transmit AIM Symbology Identifiers must be enabled for this parameter to work.

Parameter Number	Value	description
123	*0	transmits these MicroPDF417 symbols with one of the following prefixes:]L3 if the first codeword is 903-905]L4 if the first codeword is 908 or 909]L5 if the first codeword is 910 or 911
	1	transmits these MicroPDF417 symbols with one of the following prefixes:]C1 if the first codeword is 903-905]C2 if the first codeword is 908 or 909

		]C0 if the first codeword is 910 or 911

NOTE Linked MicroPDF codewords 906, 907, 912, 914, and 915 are not supported. Use GS1 Composites instead.

Data Matrix

Enable or disable Data Matrix

Parameter Number	Value	description
292	*1	Enable Data Matrix
	0	Disable Data Matrix

Maxicode

Enable or disable Maxicode

Parameter Number	Value	description
294	*1	Enable Maxicode
	0	Disable Maxicode

QR Code

Enable or disable QR Code

Parameter Number	Value	description
293	*1	Enable QR Code
	0	Disable QR Code

MicroQR

Enable or disable MicroQR

Parameter Number	Value	description
573	*1	Enable MicroQR
	0	Disable MicroQR

Aztec

Enable or disable Aztec

Parameter Number	Value	description
574	*1	Enable Aztec

	0	Disable Aztec

Han Xin

Enable or disable Han Xin

Parameter Number	Value	description
1167	*0	Disable Han Xin
	1	Enable Han Xin

IDC Operating Mode

Select the operating mode of the Intelligent Document Capture firmware

- **Off** - Disables the IDC feature.
- **Anchored** - Requires a bar code decode. The image capture region is based off this bar code.
- **Free-Form** - A printed border or page edge defines the image capture region. A bar code is optional.
- **Linked** - A printed border or page edge defines the image capture region. A bar code is required.

Parameter Number	Value	description
594	*0	Off
	1	Anchored
	2	Free-Form
	3	Linked

IDC Symbology

Select the bar code type(s) to use when Document Capture mode is not set to Off. To enable more than one symbology at a time, simply add the values together. For example, to enable PDF417, Data Matrix, and Code 39 write a value of 98 (32 + 64 + 2).

Parameter Number	Value	description
655	1	Code 128
	2	Code 39
	4	I 2 of 5
	8	D 2 of 5
	16	Codabar
	32	PD 417
	64	Data Matrix
	128	EAN 128
	512	QR Code

- Value 512(QR Code) : SM15 OS version 1.1.1 or later

IDC X Coordinate

Specify the horizontal offset to the top left corner of the region to capture relative to the center of the bar code. Negative values move toward the left. This parameter only applies when IDC Operating Mode is set to Anchored.

Parameter Number	Value	description
596	+/- 1279	Default : -151

IDC Y Coordinate

Specify the vertical offset to the top left corner of the region to capture relative to the center of the bar code. Negative values move toward the top. This parameter only applies when IDC Operating Mode is set to Anchored.

Parameter Number	Value	description
597	+/- 1023	Default : -050

IDC Width

Specify the width of the region to capture. This parameter only applies when IDC Operating Mode is set to Anchored.

Parameter Number	Value	description
598	0010 to 1279	Default : 0300

IDC Height

Specify the height of the region to capture. This parameter only applies when IDC Operating Mode is set to Anchored.

Parameter Number	Value	description
599	0010 to 1023	Default : 0050

2D User Preferences

In this section, * indicates the default option.

Picklist Mode

Picklist mode enables the decoder to decode only bar codes aligned under the center of the laser aiming pattern. Select one of the following picklist modes

Parameter Number	Value	description
402	*0 - Disabled Always	Picklist mode is always disabled.
	2 - Enabled Always	Picklist mode is always enabled.

NOTE Picklist mode works via an approximation of the aiming pattern center. In most cases this approximation is fully accurate. However, decodes can occur when the target bar code is near but not directly under the center of the aiming pattern.

Decode Session Timeout

Parameter Number	Value	description
136	99	9.9 Sec

Set the maximum time decode processing continues during a scan attempt, available in 0.1 second increments from 0.5 to 9.9 seconds. The default timeout is 9.9 seconds.

For example, to set a decode session timeout of 0.5 seconds, set this parameter to a value of 5. To set a timeout of 2.5 seconds, enter the value 25.

Transmit Code ID Character

A Code ID character identifies the code type of a scanned bar code. This is useful when decoding more than one code type. In addition to any single character prefix already selected, the Code ID character is inserted between the prefix and the decoded symbol.

NOTE If you enable Symbol Code ID Character or AIM Code ID Character, and enable Transmit "No Read" Message, the decoder appends the code ID for Code 39 to the NR message.

Select one of the following Code ID options

Parameter Number	Value	description
45	*0	None
	1	AIM Code ID Character
	2	Symbol Code ID Character

Mobile Phone/Display Mode

This mode improves bar code reading performance on mobile phones and electronic displays. Enabling this mode improves accuracy by reducing the probability of no-decodes or mis-decodes, but may increase decode time.

Select one of the following options

Parameter Number	Value	description
716	*0	Disable Mobile Phone/Display Mode
	1	Enable Mobile Phone/Display Mode

Multi Decode Mode

This mode enables decoding multiple bar codes within the scanner's field of view. Select one of the following options

Parameter Number	Value	description
900	*0	Disable Multi Decode Mode
	1	Enable Multi Decode Mode

Bar Codes to Read

This parameter sets the number of bar codes to read when Multi Decode Mode is enabled. The range is 1 to 10 bar codes. The default is 1.

Parameter Number	Value	description
902	***1**	

Full Read Mode

Select when to generate a decode event to the calling application when Multi Decode Mode is enabled

Parameter Number	Value	description
901	***1**	when at least the number of bar codes set in Bar Codes to Read are decoded.
	0	Generate a decode event after one or more bar codes are decoded.

2D Imaging Options

In this section, * indicate the default option.

Illumination Power Level

Parameter Number	Value	description
764	10	

This parameter sets the level of illumination by altering laser/LED power. The default is 10, which is maximum illumination. For values from 0 to 10, illumination varies from lowest to highest level. This parameter affects both decoding and motion illumination.

Decoding Illumination

Enable or disable illumination

Parameter Number	Value	description
298	*1 - Enable Decoding Illumination	the decoder turns on illumination every image capture to aid decoding.
	0 - Disable Decoding Illumination	the decoder does not use decoding illumination.

Enabling illumination usually results in superior images. The effectiveness of illumination decreases as the distance to the target increases.

Decode Aiming Pattern

This parameter only applies in Decode Mode.

Parameter Number	Value	description
306	*1 - Enable Decode Aiming Pattern	this projects the aiming pattern during bar code capture.
	0 - Disable Decode Aiming Pattern	this turns off the aiming pattern.

Image Capture Illumination

Enable or disable image capture illumination.

Parameter Number	Value	description
361	*1 - Enable Image Capture Illumination	the decoder turns on illumination during every image capture.
	0 - Disable Image Capture Illumination	prevents the decoder from using image capture illumination.

Enabling illumination usually results in superior images. The effectiveness of illumination decreases as the distance to the target increases.

Image Cropping

Enable or disable the Image Cropping:

Parameter Number	Value	description
301	*0 - Disable Image Cropping	presents the full 1280 x 960 for the SE4750, 1280 x 800 for the SE4710,
	1 - Enable Image Cropping	crops the image to the pixel addresses set

NOTE The decoder has a cropping resolution of 4 pixels. Setting the cropping area to less than 3 pixels transfers the entire image.

Crop to Pixel Addresses

Parameter Number
315 (Top)
316 (Left)
317 (Bottom)
318 (Right)

If you selected Enable Image Cropping, set the pixel addresses to crop to. Specify four values for Top, Left, Bottom, and Right, where Top and Bottom correspond to row pixel addresses, and Left and Right correspond to column pixel addresses.

Engine	Value Range	Column Numbering	Row Numbering
SE4750	(0, 0) to (1279, 959)	0 to1279	0 to 959
SE4710	(0, 0) to (1279, 799)	0 to 1279	0 to 799

For example, for a 4 row x 8 column image in the extreme bottom-right section of the image, set the following values:
SE4750: Top = 955, Bottom = 959, Left = 1271, Right = 1279
SE4710: Top = 795, Bottom = 799, Left = 1271, Right = 1279

Image Resolution

This option alters image resolution before compression. Rows and columns are removed from the image,
resulting in a smaller image containing the original content with reduced resolution.
Select one of the following values:

Parameter Number	Value	Description	
302	*0 (Full)	1280 x 960 (SE4750)	1280 x 800 (SE4710)
	1 (1/2)	640 x 480 (SE4750)	640 x 400 (SE4710)
	3 (1/4)	320 x 240 (SE4750)	320 x 200 (SE4710)

Enabling illumination usually results in superior images. The effectiveness of illumination decreases as the distance to the target increases.

Image Enhancement

This feature uses a combination of edge sharpening and contrast enhancement to produce an image that is visually pleasing. Select a level of image enhancement:

Parameter Number	Value	description
564	*0	Off
	1	Low
	2	Medium
	3	High

Exposure Time

Enable or disable the Image Cropping:

Parameter Number	Value	description
567	0~960 (The default is 0)	

NOTE The maximum exposure time is based on the configured Frame Rate. For example, for a frame rate of 60 fps, the maximum exposure time allowed is 15ms. Setting exposure time to a larger value than the framerate allows sets the value to the maximum allowed exposure time. As exposure time lengthens, aim brightness decreases.

Set the Exposure Time to a value from 0 to 960, where 960 represents the highest quality image. The default is 0. Each integer value represents 100 µs worth of exposure.

c. The examples of Honeywell 2D Symbology

In this section, See examples of Honeywell scanner symbology setting.

Aztec Setting

```
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_AZTEC);

boolean aztec_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

// default      1, min      1, max   3832
int min = symConfig.MinLength;
// default 3832, min      1, max   3832
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= aztec_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0; // sym_upca_enable: if you want to disable the symbol, then upca_enable = false;

symConfig.symID = SymbologyID.SYM_AZTEC;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN | SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

CODABAR Setting

```
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_CODABAR);

// Enable : default      1
// Check Character         : default      0, min      0, max       1
// Start Stop Transmit           : default      0, min      0, max       1
// Concatenate           : default      0, min      0, max       1
// MinLength       : default      2, min      2, max      60
// MaxLength       : default     60, min      2, max      60

boolean codabar_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
boolean codabar_check_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_CHECK_ENABLE) > 0 ? true : false);
boolean codabar_check_transmit = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT) > 0 ? true : false);
```

```
boolean codabar_start_stop = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_START_STOP_XMIT) > 0 ? true : false);
boolean codabar_concatenate = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_CODABAR_CONCATENATE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= codabar_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0; // "sym_codabar_enable"
flags |= codabar_check_enable ? SymbologyFlags.SYMBOLOGY_CHECK_ENABLE : 0; // "sym_codabar_check_enable"
flags |= codabar_check_transmit ? SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT : 0; // "sym_codabar_check_transmit_enable"
flags |= codabar_start_stop ? SymbologyFlags.SYMBOLOGY_START_STOP_XMIT : 0; // sym_codabar_start_stop_transmit_enable
flags |= codabar_concatenate ? SymbologyFlags.SYMBOLOGY_CODABAR_CONCATENATE : 0; // sym_codabar_concatenate_enable

symConfig.symID = SymbologyID.SYM_CODABAR;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN | SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

CODE11 Setting

```
// GET
// Enable          : default   0, min   0, max    1
// DoubleCheck     : default   1, min   0, max    1
// MinLength       : default   4, min   1, max   80
// MaxLength       : default  80, min   1, max   80

SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_CODE11);

boolean code11_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
boolean code11_check_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_CHECK_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= code11_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;
flags |= code11_check_enable ? SymbologyFlags.SYMBOLOGY_CHECK_ENABLE : 0;
```

```
symConfig.symID = SymbologyID.SYM_CODE11;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

CODE128 Setting

```
// GET
// Enable : default 1, min 0, max 1
// MinLength : default 0, min 0, max 80
// MaxLength : default 80, min 0, max 80
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_CODE128);

boolean code128_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= code128_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_CODE128;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

GS1 128 Setting

```
// GET
// gs1_128Enable : default 1, min 0, max 1
// gs1_128MinLength : default 0, min 0, max 80
// gs1_128MaxLength : default 80, min 0, max 80

SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_GS1_128);

boolean gs1128_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
```

```
int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= gs1128_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_GS1_128;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

CODE39 Setting

```
// GET
// Enable : default 1, min 0, max 1
// Start Stop Xmit : default 0, min 0, max 1
// Append On : default 0, min 0, max 1
// Full ASCII On : default 0, min 0, max 1
// Min Length : default 2, min 0, max 48
// Max Length : default 48, min 0, max 48

SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_CODE39);

boolean code39_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
boolean code39_check_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_CHECK_ENABLE) > 0 ? true : false);
boolean code39_check_transmit = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT) > 0 ? true : false);
boolean code39_start_stop = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_START_STOP_XMIT) > 0 ? true : false);
boolean code39_append_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE_APPEND_MODE) > 0 ? true :
false);
boolean code39_fullascii = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE_FULLASCII) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= code39_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0; // sym_code39_enable
flags |= code39_check_enable ? SymbologyFlags.SYMBOLOGY_CHECK_ENABLE : 0; // sym_code39_check_enable
flags |= code39_check_transmit ? SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT : 0; //
sym_code39_check_transmit_enable
flags |= code39_start_stop ? SymbologyFlags.SYMBOLOGY_START_STOP_XMIT : 0; //
sym_code39_start_stop_transmit_enable
```

```
flags |= code39_append_enable ? SymbologyFlags.SYMBOLOGY_ENABLE_APPEND_MODE : 0; // sym_code39_append_enable
flags |= code39_fullascii ? SymbologyFlags.SYMBOLOGY_ENABLE_FULLASCII : 0; // sym_code39_fullascii_enable

symConfig.symID = SymbologyID.SYM_CODE39;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN | SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

CODE93 Setting

```
// GET
// Enable : default 0, min 0, max 1
// MinLength : default 0, min 0, max 80
// MaxLength : default 80, min 0, max 80
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_CODE93);

boolean code93_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= code93_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_CODE93;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN | SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

COMPOSITE Setting

```
// GET
// CompositeEnable : default 0, min 0, max 1
// CompsOnUpcEan : default 0, min 0, max 1
// CompMinLength : default 1, min 1, max 300
// CompMaxLength : default 300, min 1, max 300

SymbolConfig symConfig = new SymbolConfig(0);
```

```
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_COMPOSITE);

boolean composite_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
boolean composite_upc_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_COMPOSITE_UPC) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= composite_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0; // sym_composite_enable
flags |= composite_upc_enable ? SymbologyFlags.SYMBOLOGY_COMPOSITE_UPC : 0; // sym_composite_upc_enable

symConfig.symID = SymbologyID.SYM_COMPOSITE;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

DataMatrix Setting

```
// Enable : default 0, min 0, max 1
// MinLength : default 1, min 1, max 3166
// MaxLength : default 3166, min 1, max 3166
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_DATAMATRIX);

boolean datamatrix_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= datamatrix_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_DATAMATRIX;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

EAN8 Setting

```
// ean8Enable          : default      0, min      0, max       1
// ean8ChkXmit         : default      0, min      0, max       1
// ean8AddendSep       : default      0, min      0, max       1
// ean8Addend2         : default      0, min      0, max       1
// ean8Addend5         : default      0, min      0, max       1
// ean8AddendReq       : default      0, min      0, max       1

// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_EAN8);

boolean ean8_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
boolean ean8_check_transmit = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT) > 0 ? true : false);
boolean ean8_addenda_separator = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ADDENDA_SEPARATOR) > 0 ? true :
false);
boolean ean8_2digit_addenda = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_2_DIGIT_ADDENDA) > 0 ? true : false);
boolean ean8_5digit_addenda = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_5_DIGIT_ADDENDA) > 0 ? true : false);
boolean ean8_addenda_required = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ADDENDA_REQUIRED) > 0 ? true :
false);



// SET
int flags = 0;
flags |= ean8_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0; // sym_ean8_enable
flags |= ean8_check_transmit ? SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT : 0; // sym_ean8_check_transmit_enable
flags |= ean8_addenda_separator ? SymbologyFlags.SYMBOLOGY_ADDENDA_SEPARATOR : 0; //
sym_ean8_addenda_separator_enable
flags |= ean8_2digit_addenda ? SymbologyFlags.SYMBOLOGY_2_DIGIT_ADDENDA: 0; // sym_ean8_2_digit_addenda_enable
flags |= ean8_5digit_addenda ? SymbologyFlags.SYMBOLOGY_5_DIGIT_ADDENDA : 0; // sym_ean8_5_digit_addenda_enable
flags |= ean8_addenda_required ? SymbologyFlags.SYMBOLOGY_ADDENDA_REQUIRED : 0; //
sym_ean8_addenda_required_enable

symConfig.symID = SymbologyID.SYM_EAN8;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;

mHService.setSymbologyConfig(symConfig);
```

EAN13 Setting

```
// ean13Enable         : default      1, min      0, max       1
// ean13ChkXmit        : default      0, min      0, max       1
// ean13AddendSep      : default      0, min      0, max       1
// ean13Addend2        : default      0, min      0, max       1
// ean13Addend5        : default      0, min      0, max       1
// ean13AddendReq      : default      0, min      0, max       1
```

```
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_EAN13);

boolean ean13_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
boolean ean13_check_transmit = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT) > 0 ? true : false);
boolean ean13_addenda_separator = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ADDENDA_SEPARATOR) > 0 ? true : false);
boolean ean13_2digit_addenda = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_2_DIGIT_ADDENDA) > 0 ? true : false);
boolean ean13_5digit_addenda = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_5_DIGIT_ADDENDA) > 0 ? true : false);
boolean ean13_addenda_required = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ADDENDA_REQUIRED) > 0 ? true : false);

// SET
int flags = 0;
flags |= ean13_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0; // sym_ean13_enable
flags |= ean13_check_transmit ? SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT : 0; // sym_ean13_check_transmit_enable
flags |= ean13_addenda_separator ? SymbologyFlags.SYMBOLOGY_ADDENDA_SEPARATOR : 0; // sym_ean13_addenda_separator_enable
flags |= ean13_2digit_addenda ? SymbologyFlags.SYMBOLOGY_2_DIGIT_ADDENDA: 0; // sym_ean13_2_digit_addenda_enable
flags |= ean13_5digit_addenda ? SymbologyFlags.SYMBOLOGY_5_DIGIT_ADDENDA : 0; // sym_ean13_5_digit_addenda_enable
flags |= ean13_addenda_required ? SymbologyFlags.SYMBOLOGY_ADDENDA_REQUIRED : 0; // sym_ean13_addenda_required_enable

symConfig.symID = SymbologyID.SYM_EAN13;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;

mHService.setSymbologyConfig(symConfig);
```

Interleaved 2 of 5 Setting

```
// GET
// MinLength      : default    4, min     2, max     80
// MaxLength      : default   80, min     2, max     80

SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_INT25);

boolean int25_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
boolean int25_check_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_CHECK_ENABLE) > 0 ? true : false);
boolean int25_check_transmit = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;
```

```
// SET
int flags = 0;
flags |= int25_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0; // sym_int25_enable
flags |= int25_check_enable ? SymbologyFlags.SYMBOLOGY_CHECK_ENABLE : 0; // sym_int25_check_enable
flags |= int25_check_transmit ? SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT : 0; // sym_int25_check_transmit_enable

symConfig.symID = SymbologyID.SYM_INT25;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

Maxicode Setting

```
// Enable : default 0, min 0, max 1
// MinLength : default 1, min 1, max 150
// MaxLength : default 150, min 1, max 150

// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_MAXICODE);

boolean maxicode_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= maxicode_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;
symConfig.symID = SymbologyID.SYM_MAXICODE;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

PDF417 Setting

```
// Enable : default 1, min 0, max 1
// MinLength : default 1, min 1, max 2750
// MaxLength : default 2750, min 1, max 2750
```

```
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_PDF417);

boolean pdf417_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= pdf417_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_PDF417;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

QR Setting

```
// Enable : default 0, min 0, max 1
// MinLength : default 1, min 1, max 7089
// MaxLength : default 7089, min 1, max 7089

// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_QR);

boolean qr_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= qr_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_QR;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;
```

```
mHService.setSymbologyConfig(symConfig);
```

Hanxin Setting

```
// Enable              : default      0, min      0, max       1
// MinLength          : default      1, min      1, max    6000
// MaxLength          : default 6000, min      1, max    6000
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_HANXIN);

boolean hanxin_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= hanxin_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_HANXIN;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

RSS Setting

```
// rss Enable          : default      0, min      0, max       1
// rss Lim Enable      : default      0, min      0, max       1
// rss Exp Enable      : default      0, min      0, max       1
// MinLength   : default      1, min      1, max      80
// MaxLength   : default     80, min      1, max      80

// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_RSS);

boolean rss_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_RSS_ENABLE) > 0 ? true : false);
boolean rsl_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_RSL_ENABLE) > 0 ? true : false);
boolean rse_transmit = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_RSE_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
```

```
int flags = 0;
flags |= rss_enable ? SymbologyFlags.SYMBOLOGY_RSS_ENABLE : 0; // sym_rss_rss_enable
flags |= rsl_enable ? SymbologyFlags.SYMBOLOGY_RSL_ENABLE : 0; // sym_rss_rsl_enable
flags |= rse_transmit ? SymbologyFlags.SYMBOLOGY_RSE_ENABLE : 0; // sym_rss_rse_enable

symConfig.symID = SymbologyID.SYM_RSS;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

UPCA Setting

```
// Enable : default 1, min 0, max 1
// Check Xmit : default 0, min 0, max 1
// Num Sys Xmit : default 1, min 0, max 0
// Addend Sep : default 0, min 0, max 1
// Addend2 : default 0, min 0, max 1
// Addend5 : default 0, min 0, max 1
// Addend Required : default 0, min 0, max 1

// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_UPCA);

boolean upca_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
boolean upca_transmit = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT) > 0 ? true : false);
boolean upca_num_transmit = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_NUM_SYS_TRANSMIT) > 0 ? true : false);
boolean upca_separator = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ADDENDA_SEPARATOR) > 0 ? true : false);
boolean upca_2_digit = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_2_DIGIT_ADDENDA) > 0 ? true : false);
boolean upca_5_digit = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_5_DIGIT_ADDENDA) > 0 ? true : false);
boolean upca_required = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ADDENDA_REQUIRED) > 0 ? true : false);
boolean upca_translate_ean13 = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_UPCA_TRANSLATE_TO_EAN13) > 0 ?
true : false);

// SET
int flags = 0;
flags |= upca_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0; // sym_upca_enable: if you want to disable the symbol,
then upca_enable = false;
flags |= upca_transmit ? SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT : 0; // sym_upca_check_transmit_enable
flags |= upca_num_transmit ? SymbologyFlags.SYMBOLOGY_NUM_SYS_TRANSMIT : 0; //
sym_upca_sys_num_transmit_enable
flags |= upca_separator ? SymbologyFlags.SYMBOLOGY_ADDENDA_SEPARATOR : 0; // sym_upca_addenda_separator_enable
flags |= upca_2_digit ? SymbologyFlags.SYMBOLOGY_2_DIGIT_ADDENDA : 0; // sym_upca_2_digit_addenda_enable
flags |= upca_5_digit ? SymbologyFlags.SYMBOLOGY_5_DIGIT_ADDENDA : 0; // sym_upca_5_digit_addenda_enable
```

```
flags |= upca_required ? SymbologyFlags.SYMBOLOGY_ADDENDA_REQUIRED : 0; // sym_upca_addenda_required_enable
flags |= upca_translate_ean13 ? SymbologyFlags.SYMBOLOGY_UPCA_TRANSLATE_TO_EAN13 : 0; // upc-a to ean13

symConfig.symID = SymbologyID.SYM_UPCA;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;
mHService.setSymbologyConfig(symConfig);
```

UPC-E1 Setting

```
// Enable        : default      0, min      0, max       1
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_UPCE1);

boolean upce1_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_UPCE1_ENABLE) > 0 ? true : false);

// SET
int flags = 0;
flags |= upce1_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_UPCE1;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;
mHService.setSymbologyConfig(symConfig);
```

UPC-E0 Setting

```
// Enable        : default      0, min      0, max       1
// Expand        : default      0, min      0, max       1
// ChkXmit       : default      0, min      0, max       1
// Num Sys Xmit      : default      0, min      0, max       1
// AddendSep     : default      0, min      0, max       1
// Addend2       : default      0, min      0, max       1
// Addend5       : default      0, min      0, max       1
// AddendReq     : default      0, min      0, max       1
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_UPCE0);

boolean upce0_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
boolean upce0_expanded = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_EXPANDED_UPCE) > 0 ? true : false);
boolean upce0_check_transmit = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT) > 0 ? true : false);
boolean upce0_num_sys = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_NUM_SYS_TRANSMIT) > 0 ? true : false);
boolean upce0_addenda_separator = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ADDENDA_SEPARATOR) > 0 ?
true : false);
boolean upce0_2digit = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_2_DIGIT_ADDENDA) > 0 ? true : false);
boolean upce0_5digit = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_5_DIGIT_ADDENDA) > 0 ? true : false);
boolean upce0_addenda_required = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ADDENDA_REQUIRED) > 0 ? true :
```

```
false);

// SET
int flags = 0;
flags |= upce0_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;
flags |= upce0_expanded ? SymbologyFlags.SYMBOLOGY_EXPANDED_UPCE : 0;
flags |= upce0_check_transmit ? SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT : 0;
flags |= upce0_num_sys ? SymbologyFlags.SYMBOLOGY_NUM_SYS_TRANSMIT : 0;
flags |= upce0_addenda_separator ? SymbologyFlags.SYMBOLOGY_ADDENDA_SEPARATOR : 0;
flags |= upce0_2digit ? SymbologyFlags.SYMBOLOGY_2_DIGIT_ADDENDA : 0;
flags |= upce0_5digit ? SymbologyFlags.SYMBOLOGY_5_DIGIT_ADDENDA : 0;
flags |= upce0_addenda_required ? SymbologyFlags.SYMBOLOGY_ADDENDA_REQUIRED : 0;

symConfig.symID = SymbologyID.SYM_UPCE0;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;
mHService.setSymbologyConfig(symConfig);
```

ISBT Setting

```
// Enable          : default      0, min      0, max       1
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_ISBT);

boolean isbt_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

// SET
int flags = 0;
flags |= isbt_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_ISBT;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;
mHService.setSymbologyConfig(symConfig);
```

IATA25 Setting

```
// Enable       : default      0, min      0, max       1
// MinLength    : default      4, min      4, max      80
// MaxLength    : default     80, min      0, max      80
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_IATA25);

boolean iata25_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;
```

```
// SET
int flags = 0;
flags |= iata25_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_IATA25;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

CodaBlock Setting

```
// Enable : default 0, min 0, max 1
// MinLength : default 0, min 0, max 2048
// MaxLength : default 2048, min 0, max 2048
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_CODABLOCK);

boolean codablock_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= codablock_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_CODABLOCK;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

MSI Setting

```
// Enable : default 0, min 0, max 1
// CheckTransmit : default 0, min 0, max 1
// MinLength : default 4, min 4, max 48
// MaxLength : default 48, min 4, max 48
// GET
SymbolConfig symConfig = new SymbolConfig(0);
```

```
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_MSI);

boolean msi_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
boolean msi_check_transmit = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= msi_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;
flags |= msi_check_transmit ? SymbologyFlags.SYMBOLOGY_CHECK_TRANSMIT : 0;

symConfig.symID = SymbologyID.SYM_MSI;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

TLCODE-39 Setting

```
// tlc39Enable : default 0, min 0, max 1
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_TLCODE39);

boolean tlcode39_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

// SET
int flags = 0;
flags |= tlcode39_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_TLCODE39;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;
mHService.setSymbologyConfig(symConfig);
```

Matrix 2 of 5 Setting

```
// Enable : default 0, min 0, max 1
// MinLength : default 4, min 4, max 80
// MaxLength : default 80, min 4, max 80
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_MATRIX25);
```

```
boolean matrix25_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= matrix25_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_MATRIX25;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

Korea Post Setting

```
// korpostEnable      : default      0, min      0, max       1
// korpostMinLength : default      4, min      4, max      48
// korpostMaxLength : default     48, min      4, max      48
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_CODABLOCK);

boolean codablock_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= codablock_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_CODABLOCK;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

CodaBlock Setting

```
// Enable         : default      0, min      0, max       1
// MinLength     : default      0, min      0, max    2048
```

```
// MaxLength    : default 2048, min      0, max   2048
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_KOREAPOST);

boolean korea_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= korea_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_KOREAPOST;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

Trioptic Setting

```
// triopEnable      : default      0, min      0, max      1
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_TRIOPTIC);

boolean trioptic_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

// SET
int flags = 0;
flags |= trioptic_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_TRIOPTIC;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;
mHService.setSymbologyConfig(symConfig);
```

Code-32 Setting

```
// c32Enable        : default      0, min      0, max      1
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_CODE32);

boolean code32_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
```

```
// SET
int flags = 0;
flags |= code32_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_CODE32;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;
mHService.setSymbologyConfig(symConfig);
```

Straight 2 of 5 Setting

```
// Enable          : default    0, min    0, max    1
// MinLength       : default    4, min    4, max    48
// MaxLength       : default   48, min    4, max    48
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_STRT25);

boolean strt25_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= strt25_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_STRT25;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

China Post Setting

```
// Enable       : default    0, min    0, max    1
// MinLength    : default    4, min    4, max   80
// MaxLength    : default   80, min    4, max   80
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_CHINAPOST);

boolean chinapost_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

int min = symConfig.MinLength;
```

```
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= chinapost_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_CHINAPOST;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

Telepen Setting

```
// teleEnable : default 0, min 0, max 1
// teleOldStyle : default 0, min 0, max 1
// teleMinLength : default 1, min 1, max 60
// teleMaxLength : default 60, min 1, max 60
// GET
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_TELEPEN);

boolean telepen_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
boolean telepen_old_style = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_TELEPEN_OLD_STYLE) > 0 ? true : false);

int min = symConfig.MinLength;
int max = symConfig.MaxLength;

// SET
int flags = 0;
flags |= telepen_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;
flags |= telepen_old_style ? SymbologyFlags.SYMBOLOGY_TELEPEN_OLD_STYLE : 0;

symConfig.symID = SymbologyID.SYM_TELEPEN;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS | SymbologyFlags.SYM_MASK_MIN_LEN |
SymbologyFlags.SYM_MASK_MAX_LEN;
symConfig.Flags = flags;
symConfig.MinLength = min;
symConfig.MaxLength = max;

mHService.setSymbologyConfig(symConfig);
```

Coupon Code Setting

```
// upcaCouponCode : default 0, min 0, max 1
// GET
```

```
SymbolConfig symConfig = new SymbolConfig(0);
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_COUPONCODE);

boolean couponCode_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);

// SET
int flags = 0;
flags |= couponCode_enable ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;

symConfig.symID = SymbologyID.SYM_COUPONCODE;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;
mHService.setSymbologyConfig(symConfig);
```

Postal Code Setting

!! Only one postal symbol can be used among 10 codes

- **Austria, Canadian, British, ID-Tag, USPS4CB, US Postal, Dutch Post, Planet Code, Japan Post, Postnet**

```
private void selectPostal(int postalID) throws RemoteException {
        int flags = 0;
        SymbolConfig symConfig = new SymbolConfig(0);

        // !! Only one postal symbol can be used among below.!!
        // postnet
        // GET
        symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_POSTNET);
        boolean postnet_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
        // SET
        flags |= (postalID == SymbologyID.SYM_POSTNET) ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;
        symConfig.symID = SymbologyID.SYM_POSTNET;
        symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
        symConfig.Flags = flags;
        mHService.setSymbologyConfig(symConfig);

        // Japan post
        // GET
        symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_JAPOST);
        boolean japan_post_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
        // SET
        flags |= (postalID == SymbologyID.SYM_JAPOST) ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;
        symConfig.symID = SymbologyID.SYM_JAPOST;
        symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
        symConfig.Flags = flags;
        mHService.setSymbologyConfig(symConfig);
```

```
// planet code post
// GET
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_PLANET);
boolean planet_code_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
// SET
flags |= (postalID == SymbologyID.SYM_PLANET) ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;
symConfig.symID = SymbologyID.SYM_PLANET;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;
mHService.setSymbologyConfig(symConfig);

// dutch post
// GET
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_DUTCHPOST);
boolean dutch_post_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
// SET
flags |= (postalID == SymbologyID.SYM_DUTCHPOST) ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;
symConfig.symID = SymbologyID.SYM_DUTCHPOST;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;
mHService.setSymbologyConfig(symConfig);

// US Postals1
// GET
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_US_POSTALS1);
boolean us_post_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
// SET
flags |= (postalID == SymbologyID.SYM_US_POSTALS1) ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;
symConfig.symID = SymbologyID.SYM_US_POSTALS1;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;
mHService.setSymbologyConfig(symConfig);

// USPS4CB
// GET
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_USPS4CB);
boolean usps4cb_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
// SET
flags |= (postalID == SymbologyID.SYM_USPS4CB) ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;
symConfig.symID = SymbologyID.SYM_USPS4CB;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;
mHService.setSymbologyConfig(symConfig);

// IDTAG
// GET
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_IDTAG);
boolean idtag_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
```

```
// SET
flags |= (postalID == SymbologyID.SYM_IDTAG) ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;
symConfig.symID = SymbologyID.SYM_IDTAG;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;
mHService.setSymbologyConfig(symConfig);

// British Post
// GET
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_BPO);
boolean bpo_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
// SET
flags |= (postalID == SymbologyID.SYM_BPO) ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;
symConfig.symID = SymbologyID.SYM_BPO;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;
mHService.setSymbologyConfig(symConfig);

// Canadian Post
// GET
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_CANPOST);
boolean canadian_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
// SET
flags |= (postalID == SymbologyID.SYM_CANPOST) ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;
symConfig.symID = SymbologyID.SYM_CANPOST;
symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
symConfig.Flags = flags;
mHService.setSymbologyConfig(symConfig);


// Austria Post
// GET
symConfig = mHService.getSymbologyConfig(SymbologyID.SYM_AUSPOST);
boolean aus_enable = ((symConfig.Flags & SymbologyFlags.SYMBOLOGY_ENABLE) > 0 ? true : false);
// SET
flags |= (postalID == SymbologyID.SYM_AUSPOST) ? SymbologyFlags.SYMBOLOGY_ENABLE : 0;
int nInterpretMode = 0; // None
switch(nInterpretMode){
        case 1: // Numeric N Table
                flags |= SymbologyFlags.SYMBOLOGY_AUS_POST_NUMERIC_N_TABLE;
                break;
        case 2: // Alphanumeric C Table
                flags |= SymbologyFlags.SYMBOLOGY_AUS_POST_ALPHANUMERIC_C_TABLE;
                break;
        case 3: // Combination N & C Tables
                flags |= SymbologyFlags.SYMBOLOGY_AUS_POST_COMBINATION_N_AND_C_TABLES;
                break;
        default:      // None
```

```
                                    break;
                }
                symConfig.symID = SymbologyID.SYM_AUSPOST;
                symConfig.Mask = SymbologyFlags.SYM_MASK_FLAGS;
                symConfig.Flags = flags;
                mHService.setSymbologyConfig(symConfig);
}
```