

# Gainscha Android Bluetooth/Ethernet/USB

## Function Library Instructions

### 1. GTSPL\_openPort()

#### ★ Bluetooth

##### **GTSPL\_openPort(MacAddress)**

■ Description: Start the printer spool.

■ Parameter:

➔ address: String Type, Specifies the Bluetooth MacAddress(BR/EDR MacAddress).

example: " DC:1D:30:00:1D:87"

#### ★ Ethernet

##### **GTSPL\_openPort(IP, Port,time)**

■ Description: Start the printer spool.

■ Parameter:

➔ IP: String Type, IP Address, example: "192.168.1.109"

➔ Port: int Type, Port Number, example: 8899

➔ time: int Type, Delay Time,1000 = 1 second, example: GTSPL\_closePort(2000)

#### ★ USB

##### **GTSPL\_openPort(manager, device)**

■ Description: Start the printer spool.

■ Parameter:

➔ **manager**: USBManager Class, pass USBManager variables.

➔ **device**: USBDevice Class, pass USBDevice variables.

## 2. GTSPL\_closePort()

- Description: Close printer spool.
- Parameter: None

## 3. GTSPL\_closePort(time)

- Description: Close printer spool.
- Parameter:
  - ➔ time: int Type, delay time, 1000=1 second, example: GTSPL\_closePort(2000)

## 4. GTSPL\_setCmdSendMode(mode)

- Description: Sent the built-in commands to the printer or file.
- Parameter:
  - ➔ mode: String Type

F: Send the built-in commands to file.

(File will be saved in InternalStorage/android/data/packageName/files)

P: Send the built-in commands to the printer.

## 5. GTSPL\_setup(width, height, speed, density, sensor, sensorDistance, sensorOffset, context)

- Description: Set up label width, label height, print speed, print density, sensor type, gap/black mark vertical distance and gap/black mark offset distance.
- Parameter:

Parameter	Type	Description
<b>width</b>	int	Set up label width; unit: mm.
<b>height</b>	int	Set up label height; unit: mm.
<b>speed</b>	int	Set up print speed (1~15: print speed at 1"~15"/sec). Selectable print speeds depend on different printer models, and maximum speed is 15"/sec.
<b>density</b>	int	Set up print density(0~15);If the number is bigger, the printout will be darker.

<b>sensor</b>	int	Set up the sensor type. 0: Gap sensor 1: Black mark sensor
<b>sensorDistance</b>	int	Set up vertical gap height of the gap/black mark; unit: mm
<b>sensorOffset</b>	int	Set up offset distance of the gap/black mark, unit: mm, this parameter is set to 0 when the general label is used.
<b>context</b>	Context	Pass current context.

## 6. **GTSPSetup(width, height, speed, density, sensor, sensorDistance, sensorOffset, context)**

- Description: Set up label width, label height, print speed, print density, sensor type, gap/black mark vertical distance and gap/black mark offset distance.

- Parameter:

Parameter	Type	Description
<b>width</b>	Double	Set up label width; unit: mm.
<b>height</b>	Double	Set up label height; unit: mm.
<b>speed</b>	int	Set up print speed (1~15: print speed at 1"~15"/sec). Selectable print speeds depend on different printer models, and maximum speed is 15"/sec.
<b>density</b>	int	Set up print density(0~15);If the number is bigger, the printout will be darker.
<b>sensor</b>	int	Set up the sensor type. 0: Gap sensor 1: Black mark sensor
<b>sensorDistance</b>	Double	Set up vertical gap height of the gap/black mark; unit: mm
<b>sensorOffset</b>	Double	Set up offset distance of the gap/black mark, unit: mm, this parameter is set to 0 when the general label is used.
<b>context</b>	Context	Pass current context.

## 7. **GTSPSetDirectionAndMirror(direction, mirror, context)**

- Description: Set direction and mirror.

- Parameter:

Parameter	Type	Description
<b>direction</b>	int	Set direction, default:0

		0 : Top out 1 : Bottom out
<b>mirror</b>	int	Set mirror 0 : No 1 : Yes
<b>context</b>	Context	Pass current context.

## 8. GTSPL\_setShift (shiftY, context)

- Description: Set the vertical displacement distance, when value is positive, it will shift in the printing direction, otherwise, it will shift in opposite direction.
- Parameter:
  - ➔ shiftY : int type, vertical displacement distance, the unit is dot.
  - ➔ context : Context Class, Pass current context.

## 9. GTSPL\_printReverse( x\_start, y\_start, x\_width, y\_height, context)

- Description: Reverse the designated area.
- Parameter:

Parameter	Type	Description
<b>x_start</b>	int	The x-coordinate of the area(in dots)
<b>y_start</b>	int	The y-coordinate of the area (in dots)
<b>x_width</b>	int	The width of area in dots
<b>y_height</b>	int	The height of area in dots
<b>context</b>	Context	Pass current context.

## 10.GTSPL\_setOffset(offset, context)

- Description: Set feed offset(Usually use with peel mode and cut mode)
- Parameter:
  - ➔ offset : double type, extra feed offset, the unit is mm
  - ➔ context : Context Class, Pass current context.

## 11.GTSPL\_setCutMode(mode, piece, context)

- Description: Set cut mode and cut number

- Parameter:

Parameter	Type	Description
<b>mode</b>	int	Set cut mode, default:1 0 : Backward 1 : Forward
<b>piece</b>	int	Set cut number
<b>context</b>	Context	Pass current context.

## 12.GTSPL\_setAfterPrintAction(mode, context)

- Description: Set action after print

- Parameter:

Parameter	Type	Description
<b>mode</b>	int	Set action after print, default:1 0 : Normal 1 : Tear Mode 2 : Peel Mode 3 : Cut Mode
<b>context</b>	Context	Pass current context.

## 13.GTSPL\_genericDefault (context)

- Description: Initialize the general setting value

- Parameter:

➔ context : Context Class, Pass current context.

## 14.GTSPL\_sensorDefault (context)

- Description: Initialize the sensor setting value

- Parameter:

➔ context : Context Class, Pass current context.

## 15. GTSPL\_clearBuffer(context)

- Description: Clear the image buffer.

- Parameter:

➔ context: Context Class, Pass current context.

#### 16. GTSPL\_barcode( x, y, type, height, readable, rotation, narrow, wide, content, context)

■ Description: Use built-in barcode formats to print barcodes.

■ Parameter:

Parameter	Type	Description
<b>x</b>	int	Specify the x-coordinate bar code on the label, Unit: dot
<b>y</b>	int	Specify the y-coordinate bar code on the label, Unit: dot
<b>type</b>	String	Set up Code Type · <a href="#">refer to Appendix</a>
<b>height</b>	int	Set up bar code height (in dots)
<b>readable</b>	int	Set up whether to print human recognizable interpretation (text) or not. 0: Do not print 1: Print barcode document left 2: Print barcode code document in the center 3: Print barcode document right
<b>rotation</b>	int	Set up barcode rotation 0 : No rotation 90 : Rotate 90 degrees clockwise 180 : Rotate 180 degrees clockwise 270 : Rotate 270 degrees clockwise
<b>narrow</b>	int	Set up narrow bar ratio (in dots), <a href="#">refer to Appendix</a>
<b>wide</b>	int	Set up wide bar ratio (in dots), <a href="#">refer to Appendix</a>
<b>content</b>	String	Content of barcode. Please note that the maximum number of digits of bar code content. <a href="#">refer to Appendix</a>
<b>context</b>	Context	Pass current context.

#### 17. GTSPL\_formFeed(context)

■ Description: Feed label to the top of next label.

■ Parameter:

➔ context: Context Class, Pass current context.

#### 18. GTSPL\_noBackFeed(context)

■ Description: Set the paper not to back feed.

■ Parameter:

➔ context: Context Class, Pass current context.

#### 19. GTSPL\_sendCommand (context, command)

■ Description: Sends built-in commands to the printer.

■ Parameter:

➔ command: String Type, refer to TSPL programming manual commands for details.

➔ context: Context Class, Pass current context.

#### 20. GTSPL\_printerFont(x, y, size, rotation, x\_scale, y\_scale, content, context)

■ Description: Use printer built-in fonts to print.

■ Parameter:

Parameter	Type	Description
<b>x</b>	int	The x-coordinate of the text
<b>y</b>	int	The y-coordinate of the text
<b>size</b>	String	Built-in font type 1: 8*/12 dots 2: 12*20 dots 3: 16*24 dots 4: 24*32 dots 5: 32*48 dots TST24.BF2: Traditional Chinese 24*24 TST16.BF2: Traditional Chinese 16*16 TSS24.BF2: Simplified Chinese 24*24 TSS16.BF2: Simplified Chinese 16*16
<b>rotation</b>	int	The rotation angle of text 0 : No rotation 90: degrees, in clockwise direction

		180 : degrees, in clockwise direction 270 : degrees, in clockwise direction
<b>x_scale</b>	int	Horizontal multiplication, Available factors: 1~10
<b>y_scale</b>	int	Vertical multiplication, Available factors: 1~10
<b>content</b>	String	Content of text string
<b>context</b>	Context	Pass current context.

## 21. GTSP\_L\_qrcode(x, y, size, ECCLevel, cellWidth, mode, rotation, content)

- Description : Use built-in QR code formats to print QR code
- Parameter:

parameter	Type	illustrate
<b>x</b>	int	The upper left corner x-coordinate of the QR code
<b>y</b>	int	The upper left corner y-coordinate of the QR code
<b>ECCLevel</b>	String	Error correction recovery level  L : 7% 、 M : 15% 、  Q : 25% 、 H : 30%
<b>cellWidth</b>	int	QR code shape width 1~10
<b>mode</b>	int	Set QR code mode : A : Auto 、 M : Manual
<b>rotation</b>	int	Set up QR code rotation :  0 : 0 degree 、 90 : 90 degree 、  180 : 180 degree 、 270 : 270 degree
<b>content</b>	String	The encodable character set is described as below, *Encodable character set: 1) Numeric data: (digits 0~9) 2) Alphanumeric data Digits 0-9 Upper case letters A-Z Nine other characters: space, \$ % * + - . / : ) 3) 8-bit byte data JIS 8-bit character set (Latin and Kana) in accordance with JIS X 0201 4) Kanji characters Shift JIS values 8140HEX –9FFCHEX and E040HEX –EAA4 HEX. These are values shifted from those of JIS X 0208. Refer to JIS X 0208



		Annex 1 Shift Coded Representation for detail.
<b>context</b>	Context	Pass current context.

## 22. GTSPL\_printLabel(set, copy, context)

- Description: Print the label format currently stored in the image buffer.
- Parameter:
  - ➔ set: int Type, Specifies how many sets of labels will be printed.  
 $1 \leq \text{set} \leq 999999999$
  - ➔ copy: int Type, Specifies how many copies should be printed for each particular label set.  
 $1 \leq \text{copy} \leq 999999999$
  - ➔ context: Context Class, Pass current context.

## 23. GTSPL\_downloadPCX(filename, context)

- Description: Download the monochrome PCX file to main board flash memory
- Parameter:
  - ➔ filename: String Type, PCX file name  
(File has to be saved in InternalStorage/android/data/packageName/files)
  - ➔ context: Context Class, Pass current context.

## 24. GTSPL\_downloadBMP(filename, context)

- Description: Download the monochrome BMP file to main board flash memory
- Parameter:
  - ➔ filename: String Type, BMP file name.  
(File has to be saved in InternalStorage/android/data/packageName/files)
  - ➔ context: Context Class, Pass current context.

## 25. GTSPL\_download\_Not1BitDepthBMP (filename, context)

- Description: Download the non-monochrome file to main board flash memory.
- Parameter:
  - ➔ filename: String Type, File name  
(File has to be saved in InternalStorage/android/data/packageName/files)
  - ➔ context : Context Class, Pass current context.

## 26. GTSPL\_downloadTTF(filename, context)

- Description: Download the True Type Font file to main board flash memory.
- Parameter:
  - ➔ filename: String Type, File name  
(File has to be saved in InternalStorage/android/data/packageName/files)
  - ➔ context: Context Class, Pass current context.

## 27. GTSPL\_printerStatus(delaytime)

- Description: Obatin the printer status.
- Parameter:
  - ➔ delaytime: int Type, Set up delay time.
- Return Value(String Type):

Return Value	Printer Status
00	Normal
01	Head opened
02	Paper Jam
03	Paper Jam and head opened
04	Out of paper
05	Out of paper and head opened
08	Out of ribbon
09	Out of ribbon and head opened
0A	Out of ribbon and paper jam
0B	Out of ribbon, paper jam and head opened
0C	Out of ribbon and out of paper

<b>0D</b>	Out of ribbon, out of paper and head opened
<b>10</b>	Pause
<b>20</b>	Printing
<b>80</b>	Other error

## 28. GTSPL\_getSDKVersion (returnWay, context)

- Description: Response the SDK version.
- Parameter:
  - ➔ retrunWay: int Type, pass entering 0 will bounce out the SDK version message in addition to returning the SDK.
  - ➔ context: Context Class, Pass current context.

## 29. GTSPL\_writeUHF (dataFormat,startBlockNo,byteSize,Gen2MemoryBank,datastring, context)

- Description: Write data to UHF tag memory.
- Parameter:

Parameter	Type	Description
<b>dataFormat</b>	string	Define data format · default is“H”  A : ASCII  H : Hexadecimal
<b>startBlockNo</b>	int	Secify the 16-bit starting block number · Default : 2
<b>byteSize</b>	int	Values: 1 to n, where n is the maximum number of bytes for the tag. Default: 1
<b>Gen2Memory Bank</b>	string	Select Gen2 memory bank  R : Reserved  E : EPC (Default)

		T : TID(Tag ID)  U : User
<b>datastring</b>	string	Data string.
<b>context</b>	Context	Pass current context.

### 30. GTSPL\_EPCPWD\_Action(action, password, context)

- Description: Lock or unlock EPC memory with password for UHF GEN2 tag.
- Parameter:

Parameter	Type	Description
<b>action</b>	string	Action type  U : unlock EPC memory bank  L : lock EPC memory bank  O : permanently unlock EPC memory bank  P : permanently lock EPC memory bank
<b>password</b>	string	password · 8 HEX characters. (0~9, A,B,C,D,E,F)
<b>context</b>	Context	Pass current context.

### 31. GTSPL\_TIDPWD\_Action(action, password, context)

- Description: Lock or unlock TID memory with password for UHF GEN2 tag.
- Parameter:

Parameter	Type	Description
<b>action</b>	string	Action type  U : unlock TID memory bank  L : lock TID memory bank

		O : permanently unlock TID memory bank  P : permanently lock TID memory bank
<b>password</b>	string	password ·  8 HEX characters. (0~9, A,B,C,D,E,F)
<b>context</b>	Context	Pass current context.

### 32. GTSPL\_USERPWD\_Action(action, password, context)

- Description: Lock or unlock USER memory with password for UHF GEN2 tag.
- Parameter:

Parameter	Type	Description
<b>action</b>	String	Action type  U : unlock USER memory bank  L : lock USER memory bank  O : permanently unlock USER memory bank  P : permanently lock USER memory bank
<b>password</b>	String	password ·  8 HEX characters. (0~9, A,B,C,D,E,F)
<b>context</b>	Context	Pass current context.

### 33. GTSPL\_AccessPWD\_Action (action, password, context)

- Description: Lock or unlock access password with password for UHF GEN2 tag.
- Parameter:

Parameter	Type	Description
<b>action</b>	string	Action type  U : unlock the access password*  L : lock the access password*

		O : permanently unlock the access password  P : permanently lock the access password  S : Set Password
<b>password</b>	string	password ·  8 HEX characters. (0~9, A,B,C,D,E,F)
<b>context</b>	Context	Pass current context.

### 34. GTSPL\_KillPWD\_Action (action, password, context)

- Description: Lock or unlock kill password with password for UHF GEN2 tag.
- Parameter:

Parameter	Type	Description
<b>action</b>	string	Action type  U : unlock the kill password*  L : lock the kill password*  O : permanently unlock the kill password  P : permanently lock the kill password  S : Set Password
<b>password</b>	string	password ·  8 HEX characters. (0~9, A,B,C,D,E,F)
<b>context</b>	Context	Pass current context.

### 35. GTSPL\_Set\_RFIDPorcedure (tagType, rw\_position, void\_printout, tryEncodie\_times,error\_handle, speed, retry\_times, context)

- Description: Set RFID procedure
- Parameter:

Parameter	Type	Description
<b>tagType</b>	int	Set Tag type · accepted value:1~10 ·  For UHF: 1 = ISO 18000 6C/Class 1 Gen2 (Q command) 8 = ISO 18000 6C/Class 1 Gen 2 (default)  For HF 10 = UHF-J
<b>rw_position</b>	int	Move the media to the specified position on the label, measured in dot rows from the label top, before encoding  Accept value: 0~9999(mm) · default is 0
<b>void_printout</b>	int	Set the length of the void printout in vertical (Y axis) dot rows. Accepted values: 0 to label length Default: label length
<b>tryEncodie_times</b>	string	The number of labels that will be attempted in case of read/encode failure. Accepted values: 1 to 10 Default: 3
<b>error_handle</b>	string	If an error persists after the specified number of labels are tried, perform this error handling action.  N : No action (Default)  P : Pause mode  E : Error mode
<b>speed</b>	int	If a label is voided, the speed at which "VOID" will be printed across the label. Accepted value: 2~10(IPS), Default is 2.
<b>retry_times</b>	int	The retry times of a tag that will be attempted in case of read/encode failure.  Accepted value:0~10 · Default is 6
<b>context</b>	Context	Pass current context.

### 36. GTSPL\_Set\_RFIDPorcedure (tagType, rw\_position, void\_printout, tryEncodie\_times,error\_handle, speed, retry\_times, dpi, context)

■ Description: Set RFID procedure

■ Parameter:

Parameter	Type	Description
<b>tagType</b>	int	Set Tag type, accepted value:1~10, For UHF: 1 = ISO 18000 6C/Class 1 Gen2 (Q command) 8 = ISO 18000 6C/Class 1 Gen 2 (default) For HF 10 = UHF-J
<b>rw_position</b>	int	Move the media to the specified position on the label, measured in dot rows from the label top, before encoding Accept value: 203dpi:0 ~ 1251 (mm)、 300dpi:0 ~ 846 (mm)、 600dpi:0 ~ 423 (mm), default is 0
<b>void_printout</b>	int	Set the length of the void printout in vertical (Y axis) dot rows. Accepted values: 0 to label length Default: label length
<b>tryEncodie_times</b>	string	The number of labels that will be attempted in case of read/encode failure. Accepted values: 1 to 10 Default: 3
<b>error_handle</b>	string	If an error persists after the specified number of labels are tried, perform this error handling action. N: No action (Default) P: Pause mode E: Error mode
<b>speed</b>	int	If a label is voided, the speed at which "VOID" will be printed across the label. Accepted value: 2~10(IPS), Default is 2.
<b>retry_times</b>	int	The retry times of a tag that will be attempted in case of read/encode failure. Accepted value:0~10, Default is 6
<b>dpi</b>	String	Set DPI of the printer 203: 203 dpi 300: 300 dpi 600: 600 dpi



<b>context</b>	Context	Pass current context.
----------------	---------	-----------------------

### 37. GTSPL\_writeHF (dataFormat,startBlockNo,byteSize,datastring, context)

- Description: Write data to HF tag memory.

- Parameter:

Parameter	Type	Description
<b>dataFormat</b>	string	Define data format · default is“H”  A : ASCII  H : Hexadecimal
<b>startBlockNo</b>	int	Secify the 16-bit starting block number · Default : 2
<b>byteSize</b>	int	Values: 1 to n, where n is the maximum number of bytes for the tag. Default: 1
<b>datastring</b>	string	Data string
<b>context</b>	Context	Pass current context.

### 38. GTSPL\_printerFontBlock (x, y, width, height, fontname, rotation, x\_scale, y\_scale, space, align, content, context)

- Description: Use printer built-in fonts to print paragraph.

- Parameter:

Parameter	Type	Description
<b>x</b>	string	The x-coordinate of the text (in dots)
<b>y</b>	string	The y-coordinate of the text (in dots)
<b>width</b>	string	The width of block for the paragraph in dots
<b>height</b>	string	The height of block for the paragraph in dots
<b>fontname</b>	string	Built-in font type 1: 8*/12 dots 2: 12*20 dots 3: 16*24 dots 4: 24*32 dots 5: 32*48 dots

		TST24.BF2: Traditional Chinese 24*24 TST16.BF2: Traditional Chinese 16*16 TSS24.BF2: Simplified Chinese 24*24 TSS16.BF2: Simplified Chinese 16*16
<b>rotation</b>	string	The rotation angle of text 0 : No rotation 90 : degrees, in clockwise direction 180 : degrees, in clockwise direction 270 : degrees, in clockwise direction
<b>x_scale</b>	string	Horizontal multiplication, Available factors: 1~10
<b>y_scale</b>	string	Vertical multiplication, Available factors: 1~10
<b>space</b>	string	Add or delete the space between lines (in dots)
<b>align</b>	string	Text alignment 0 : default (Left) 1 : Left 2 : Center 3 : Right
<b>content</b>	string	Data in block. The maximum data length is 4092 bytes.
<b>context</b>	Context	Pass current context.

### 39. GTSPL\_readUHF(dataFormat,startBlockNo,byteSize,Gen2MemoryBank,context)

- Description: Read data from UHF tag memory (R command)
- Parameter:

Parameter	Type	Description
<b>dataFormat</b>	string	Setting callback returned data format ·  A : ASCII  H : Hexadecimal (Default)
<b>startBlockNo</b>	int	Secify the 16-bit starting block number to read · Default is 0
<b>byteSize</b>	int	Secify the data lengths to read · default is 1
<b>Gen2Memory Bank</b>	string	Gen2 memory bank ·  R = Reserved

		E = EPC T = TID U = UESR  Default : E
<b>context</b>	Context	Pass current context.

■ Return Value(String Type):

dataFormat	Return string(example)
<b>A</b>	Label data is displayed in ASCII (ex: 24051324000103456400)
<b>H</b>	Label data is displayed in Hexadecimal (ex: 3234303531333234303030313033343536343030)
* An error occurs and an error code is returned , For error code description, <a href="#">refer to Appendix 2</a>  UHFReaderE710 <a href="#">refer to Appendix 3</a>	

#### 40. GTSPL\_readUHF(dataFormat,startBlockNo,byteSize,Gen2MemoryBank, delaytime,context)

■ Description: Read data from UHF tag memory (R command)(only supports USB)

■ Parameter:

Parameter	Type	Description
<b>dataFormat</b>	string	Setting callback returned data format ,  A : ASCII  H : Hexadecimal (Default)
<b>startBlockNo</b>	int	Secify the 16-bit starting block number to read , Default is 0
<b>byteSize</b>	int	Secify the data lengths to read , default is 1
<b>Gen2Memory Bank</b>	string	Gen2 memory bank ,  R = Reserved E = EPC T = TID U = UESR

		Default : E
<b>delaytime</b>	int	Setting read delaytime
<b>context</b>	Context	Pass current context.

■ Return Value(String Type):

<b>dataFormat</b>	<b>Return string(example)</b>
<b>A</b>	Label data is displayed in ASCII (ex: 24051324000103456400)
<b>H</b>	Label data is displayed in Hexadecimal (ex: 3234303531333234303030313033343536343030)
* An error occurs and an error code is returned , For error code description, <a href="#">refer to Appendix 2</a>  UHFReaderE710 <a href="#">refer to Appendix 3</a>	

#### 41. GTSPL\_readUHFQ(dataFormat, PCReturnStatus, CRCReturnStatus,context)

■ Description: Read data from UHF tag memory (Q command)

■ Parameter:

<b>Parameter</b>	<b>Type</b>	<b>Description</b>
<b>dataFormat</b>	string	Set up callback return data format .  A : ASCII  H : Hexadecimal (Default)
<b>PCReturnStatus</b>	int	enable/disable PC value returned  0 : read epc data not include PC value  1 : read epc data include PC value
<b>CRCReturnStatus</b>	int	enable/disable CRC-16 value returned  0 : read epc data not include CRC-16 value  1 : read epc data include CRC-16 value
<b>context</b>	Context	Pass current context.

- Return Value(String Type):

dataFormat	Return string(example)
A	Label data is displayed in ASCII (ex: 24051324000103456400)
H	Label data is displayed in Hexadecimal (ex: 3234303531333234303030313033343536343030)
* An error occurs and an error code is returned · For error code description, <a href="#">refer to Appendix 2</a> , UHFReaderE710 <a href="#">refer to Appendix 3</a>	

#### 42. GTSPL\_LabelCalibration (Context context)

- Description: Auto calibration for RFID label
- Parameter:
  - ➔ context: Context Class, Pass current context.

#### 43. GTSPL\_rfidSetupDefault (Context context )

- Description: Initialize the RFID setting value
- Parameter:
  - ➔ context: Context Class, Pass current context.

#### 44. GTSPL\_writeGJB(String dataFormat, int startBlockNo, int byteSize, String GJBMemoryBank, String datastring, String writePWD, Context context)

- Description:Write Write data to UHF GJB tag memory
- Parameter:

Parameter	Type	Description
dataFormat	string	Define data format · default is“H”  A : ASCII  H : Hexadecimal
startBlockNo	int	Secify the 16-bit starting block number · Default : 1

<b>byteSize</b>	int	Values: 1 to n, where n is the maximum number of bytes for the tag. Default: 1
<b>Gen2MemoryBank</b>	string	Select GJB memory bank to write R = Reserved E = EPC T = TID U = User 2 = User 2 3 = User 3  Default : E
<b>datastring</b>	string	Data string
<b>writePWD</b>	string	Writing password ·  8 HEX characters. (0~9, A,B,C,D,E,F)
<b>context</b>	Context	Pass current context.

#### 45. GTSPL\_readGJB(String dataFormat, int startBlockNo, int byteSize, String GJBMemoryBank, String ReadPWD, Context context)

■ Description: Read data from UHF GJB tag.

■ Parameter:

Parameter	Type	Description
<b>dataFormat</b>	string	Set up callback returned data format ·  A : ASCII  H : Hexadecimal (Default)
<b>startBlockNo</b>	int	Secify the 16-bit starting block number to read · Default is 0
<b>byteSize</b>	int	Secify the data lengths to read · default is 1
<b>Gen2MemoryBank</b>	string	Select GJB memory bank to read ·  E = EPC T = TID U = User

		2 = User 2 3 = User 3  Default : E
<b>ReadPWD</b>	string	Reading password ·  8 HEX characters. (0~9, A,B,C,D,E,F)
<b>context</b>	Context	Pass current context.

■ Return Value(String Type):

dataFormat	Return string(example)
<b>A</b>	Label data is displayed in ASCII (ex: 24051324000103456400)
<b>H</b>	Label data is displayed in Hexadecimal (ex: 32343035313332343030303130333343536343030)

#### 46. GTSPL\_Set\_GJB\_Pwd(String pwdArea, String action, String pwdSet, String writePWD, Context context)

■ Description: Set Write/Read/Status/Kill Password to UHF GJB tag.

■ Parameter:

Parameter	Type	Description
<b>passwordArea</b>	string	Set password area K=Kill, W=Write ( Default), R=Read, S=Status
<b>action</b>	string	S=Set Password
<b>pwdSet</b>	string	New password for password area setting above ·  8 HEX characters. (0~9, A,B,C,D,E,F)
<b>writePWD</b>	string	Writing password ·  8 HEX characters. (0~9, A,B,C,D,E,F)
<b>context</b>	Context	Pass current context.

#### 47. GTSPL\_Set\_GJB\_Status(String GJBMemoryBank, String action, String statusPWD, Context context)

■ Description: Set memory status with password for UHF GJB tag.

■ Parameter:

Parameter	Type	Description
<b>Gen2MemoryBank</b>	string	Select GJB memory bank for set, E = EPC (default) T = TID U = User 2 = User 2 3 = User 3
<b>action</b>	string	Memory Status type  A : Lock0( readable and writable )  B : Lock1(read only)  C : Lock2(write only)  D : Lock3( non-readable and non-writable )  *Each Memory support status : <ul style="list-style-type: none"> <li>EPC area : A:read and write   B: read only</li> <li>USER area(include User 、 User2 、 User3) : A:read and write   B:read only C: write only   D: not allow access</li> <li>TID area : B:read only   D: not allow access</li> <li>SAFE area : C: write only   D: not allow access</li> </ul>
<b>statusPWD</b>	string	Status password ,  8 HEX characters. (0~9, A,B,C,D,E,F)
<b>context</b>	Context	Pass current context.

#### 48. GTSPL\_Kill\_GJB\_Tag(String kill\_PWD, Context context)

■ Description: Kill UHF GJB tag.

■ Parameter:

Parameter	Type	Description
<b>kill_PWD</b>	string	Killing password ,  8 HEX characters. (0~9, A,B,C,D,E,F)



<b>context</b>	Context	Pass current context.
----------------	---------	-----------------------

#### 49. GTSPL\_WifiFrequency (String Frequency, Context context)

- Description: When using a compatible 5G frequency band WiFi module, it can be used to switch between frequency bands.

- Parameter:

Parameter	Type	Description
<b>Frequency</b>	string	Module Frequency Setting: 2.4G: Use the 2.4GHz frequency band. 5G: Use the 5GHz frequency band. BOTH: Use dual-band frequency
<b>context</b>	Context	Pass current context.

#### 50. GTSPL\_printBMP (String x, String y, int width, int height, int mode, String filename, Context context)

- Description: Convert the image into a monochrome bitmap and directly print using the printer.

- Parameter:

Parameter	Type	Description
<b>x</b>	string	The x-coordinate of the text (in dots)
<b>y</b>	string	The y-coordinate of the text (in dots)
<b>width</b>	int	The width of the image is represented in bytes.
<b>height</b>	int	The height of the image is represented in dots (pixels).
<b>mode</b>	int	Image format 0: OVERWRITE 1: OR 2: XOR
<b>filename</b>	string	The file name . (File has to be saved in InternalStorage/android/data/packageName/files) The supported image formats are as follows: 1. BMP (Bitmap): A bitmap format. 2. JPG (JPEG): Compressed image format. 3. PNG (Portable Network Graphics): Lossless compressed image format. 4. GIF (Graphics Interchange Format): Format that supports multiple images, commonly used for animations. 5. TIFF (Tagged Image File Format): High-quality lossless compressed

		image format. 6. ICO (Icon): Icon format used for displaying icons of files, programs, or folders. 7. WMF (Windows Metafile): Windows graphics file format. EMF (Enhanced Metafile): Enhanced Windows graphics file format.
<b>context</b>	Context	Pass current context.

## 51. GTSPL\_printBMP\_Compression (String x, String y, int width, int height, String filename,

### Context context)

- Description: Convert the image into a monochrome bitmap, compress it, and then print using the printer.

- Parameter:

Parameter	Type	Description
<b>x</b>	string	The x-coordinate of the text (in dots)
<b>y</b>	string	The y-coordinate of the text (in dots)
<b>width</b>	int	The width of the image is represented in bytes.
<b>height</b>	int	The height of the image is represented in dots (pixels).
<b>filename</b>	string	The file name. (File has to be saved in InternalStorage/android/data/packageName/files) The supported image formats are as follows: 1. BMP (Bitmap): A bitmap format. 2. JPG (JPEG): Compressed image format. 3. PNG (Portable Network Graphics): Lossless compressed image format. 4. GIF (Graphics Interchange Format): Format that supports multiple images, commonly used for animations. 5. TIFF (Tagged Image File Format): High-quality lossless compressed image format. 6. ICO (Icon): Icon format used for displaying icons of files, programs, or folders. 7. WMF (Windows Metafile): Windows graphics file format. EMF (Enhanced Metafile): Enhanced Windows graphics file format.
<b>context</b>	Context	Pass current context.

## 52. GTSPL\_autoConnect (MacAddress, isManual)

- Description: If it is not closed manually, start the printer spool after 7 seconds. (Only supports BT)
- Only supports Printer:GE2408DE1168
- Parameter:

➔ **MacAddress** : String Type, Specifies the Bluetooth MacAddress(BR/EDR MacAddress).

example: " DC:1D:30:00:1D:87"

➔ **isManual** : Boolean Type, confirm whether it is closed manually

## 53. GTSPL\_labelCalibration(width, height, sensor, sensorDistance, tagType, context)

- Description : Set up label width, label height, sensor type, gap/black mark vertical distance, tag type, after setting, auto calibration for RFID label
- Only supports Printer:GE2408DE1168
- Parameter :

Parameter	Type	Description
<b>width</b>	Double	Set up label width; unit: mm.
<b>height</b>	Double	Set up label height; unit: mm.
<b>sensor</b>	int	Set up the sensor type. 0: Gap sensor 1: Black mark sensor
<b>sensorDistance</b>	Double	Set up vertical gap height of the gap/black mark; unit: mm
<b>tagType</b>	int	Set Tag type , accepted value:1~10 , For UHF: 1 = ISO 18000 6C/Class 1 Gen2 (Q command) 8 = ISO 18000 6C/Class 1 Gen 2 (default) For HF 10 = UHF-J
<b>context</b>	Context	Pass current context.

#### 54. GTSPL\_setBicolor(color, density, context)

- Description : Set two-color command
- Only supports some models, please contact your agent for details
- Parameter :

Parameter	Type	Description
<b>color</b>	string	Set color R : Red B : Black
<b>density</b>	int	Set up print density(0~15);If the number is bigger, the printout will be darker. (Set 0 when color is black)
<b>context</b>	Context	Pass current context.

# Android Bluetooth Examples

1.Add the following permissions in AndroidManifest.xml first:

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

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

2. Import GTSPS\_SDK SDK:

```
import com.gainscha.gtspl_sdk.GTSPLActivity;
```

3.Example:

```
public class MainActivity extends AppCompatActivity {  
  
    GTSPLActivity mGtsplCmdTest = new GTSPLActivity();  
  
    protected void onCreate(Bundle savedInstanceState) {  
  
        super.onCreate(savedInstanceState);  
  
        setContentView(R.layout.activity_main);  
  
        mGtsplCmdTest.GTSPL_setCmdSendMode ("P");  
  
        mGtsplCmdTest.GTSPL_openPort ("DC:1D:30:00:1D:87");  
  
        mGtsplCmdTest.GTSPL_autoConnect("DC:1D:30:00:1D:87", false);  
  
        mGtsplCmdTest.GTSPL_setup(62, 45, 2, 6, 0, 3, 0, this);  
  
        mGtsplCmdTest.GTSPL_setup(62.0, 45.0, 2, 6, 0, 3.0, 0.0, this);  
  
        mGtsplCmdTest.GTSPL_sendCommand(this, "DIRECTION 1\n\n");  
  
        mGtsplCmdTest.GTSPL_clearBuffer(this);  
  
        mGtsplCmdTest.GTSPL_printerFont(100, 100, "3", 0, 1, 1, "Print Font 123456", this);  
  
        mGtsplCmdTest.GTSPL_barcode(30, 30, "128", 100, 1, 0, 2, 2, "barcode1234567", this);  
  
        mGtsplCmdTest.GTSPL_qrcode(300, 100, "H", 4, "A", 0, "ABCabc123", this);  
  
        mGtsplCmdTest.GTSPL_downloadBMP("CIRCLE.BMP",  this);  
  
        mGtsplCmdTest.GTSPL_sendCommand(this, "PUTBMP 150, 30, \"CIRCLE.BMP\\\"\\r\\n");  
  
        mGtsplCmdTest.GTSPL_download_Not1BitDepthBMP("printTest4.BMP", MainActivity.this);  
  
        mGtsplCmdTest.GTSPL_sendCommand(MainActivity.this,  
  
        "PUTBMP10,10,\"PrintTest4.BMP\\\"\\r\\n");  
    }  
}
```

```

        String sBlock = "We stand behind our products with one of the most
comprehensive support programs in the Auto-ID industry.";

mGtsplCmdTest.GTSPL_printFontBlock("15", "15", "790", "90", "0", "0", "8", "8", "20",
"2",sBlock,this);

mGtsplCmdTest.GTSPL_printLabel(1, 1, this);


// initialization

mGtsplCmdTest.GTSPL_genericDefault(); // Initialize the general setting value
mGtsplCmdTest.GTSPL_sensorDefault(); // Initialize the sensor setting value
mGtsplCmdTest.GTSPL_rfidSetupDefault(); // Initialize the RFID setting value


// Modify Wifi frequency band

mGtsplCmdTest.GTSPL_WifiFrequency("5G");


//Set print function(It needs to use with print)

mGtsplCmdTest.GTSPL_setDirectionAndMirror(1, 1); // Set direction and mirror.
mGtsplCmdTest.GTSPL_setShift(50); // Set the vertical displacement distance
mGtsplCmdTest.GTSPL_printReverse(10, 10, 160, 160); //Set reverse
mGtsplCmdTest.GTSPL_setOffset(20); // Set feed offset
mGtsplCmdTest.GTSPL_setCutMode(0, 2); // Set cut mode and cut number
mGtsplCmdTest.GTSPL_setAfterPrintAction(2); // Set action after print
mGtsplCmdTest.GTSPL_printLabel(1, 1);


//Bitmap Print

mGtsplCmdTest.GTSPL_printBMP(-500,30,400,300,1,"CIRCLE.BMP ",this);

mGtsplCmdTest.GTSPL_printBMP_Compression(-500,30,400,300," CIRCLE.BMP ",this);

mGtsplCmdTest.GTSPL_printLabel(1, 1);

```

```
//Bicolor Print
```

```
mGtsplCmdTest.GTSPL_setBicolor("R",7, this);
```

```
//GEN2 RFID
```

```
mGtsplCmdTest.GTSPL_writeUHF("H", 2, 12, "E", "414142424343444445454646", this);
```

```
mGtsplCmdTest.GTSPL_GTSPL_EPCPWD_Action("U", "12345678", this);
```

```
mGtsplCmdTest.GTSPL_GTSPL_TIDPWD_Action("L", "12345678", this);
```

```
mGtsplCmdTest.GTSPL_USERPWD_Action("L", "12345678", this);
```

```
mGtsplCmdTest.GTSPL_AccessPWD_Action("S", "12345678", this);
```

```
mGtsplCmdTest.GTSPL_KillPWD_Action("S", "12345678", this);
```

```
mGtsplCmdTest.GTSPL_Set_RFIDPorcedure(8, 8, 32, 3, "N", 2, 2, this);
```

```
mGtsplCmdTest.GTSPL_Set_RFIDPorcedure(8, 8, 32, 3, "N", 2, 2, "203", this);
```

```
mGtsplCmdTest.GTSPL_writeHF("H", 2, 12, "414142424343444445454646", this);
```

```
mGtsplCmdTest.GTSPL_printLabel(1, 1, this);
```

```
String uhfData = mGtsplCmdTest.GTSPL_readUHF("H",2,12,"E", this);
```

```
String uhfData = mGtsplCmdTest.GTSPL_readUHFQ("H",0,0, this);
```

```
//GJB RFID set password
```

```
// Use write password, set a new read password
```

```
mGtsplCmdTest.GTSPL_Set_GJB_Pwd("R","S","87654321","12345678",this);
```

```
// Use write password, set a new write password
```

```
mGtsplCmdTest.GTSPL_Set_GJB_Pwd("W","S","87654321","12345678",this);
```

```
// Use write password, set a new kill password
```

```
mGtsplCmdTest.GTSPL_Set_GJB_Pwd("K","S","87654321","12345678",this);
```

```
// Use write password, set a new status password
```

```
mGtsplCmdTest.GTSPL_Set_GJB_Pwd("S","S","87654321","12345678",this);
```

```

mGtsplCmdTest.GTSPL_printLabel(1, 1, this);

//GJB RFID set status
mGtsplCmdTest.GTSPL_Set_GJB_Status("E","C","11112222",this);
mGtsplCmdTest.GTSPL_printLabel(1, 1, this);

//GJB RFID write EPC data
mGtsplCmdTest.GTSPL_writeGJB("H",2,12,"E","404041414242434344444545","12345678",this);

mGtsplCmdTest.GTSPL_printLabel(1, 1, this);

//GJB RFID read EPC data
String GJBData=mGtsplCmdTest.GTSPL_readGJB("H",2,12,"E","33334444",MainActivity.this);

//GJB RFID kill tag
mGtsplCmdTest.GTSPL_Kill_GJB_Tag("11224455",this);
mGtsplCmdTest.GTSPL_printLabel(1,1,this);

// Print Simplified Chinese
String stString="默认简体中文测试";
mGtsplCmdTest.GTSPL_clearBuffer(this);
mGtsplCmdTest.GTSPL_printerFont(100, 10, "TSS24.BF2", 0, 1, 1, stString, this);
mGtsplCmdTest.GTSPL_printLabel(1, 1, this);

// Print traditional Chinese
String ttString="默認繁體中文測試";
mGtsplCmdTest.GTSPL_clearBuffer(this);

```



```

mGtsplCmdTest.GTSPL_printerFont(100, 10, " TST24.BF2", 0, 1, 1, ttString, this);

mGtsplCmdTest.GTSPL_printLabel(1, 1, this);

String status = mGtsplCmdTest.GTSPL_printersStatus(1000);

mGtsplCmdTest.GTSPL_closePort(1000);

String version= mGtsplCmdTest .GTSPL_getSDKVersion(0,this);


//RFID auto calibration

mGtsplCmdTest.GTSPL_LabelCalibration(this);


//UHFReaderE710 auto calibration

mGtsplCmdTest.GTSPL_labelCalibration(62.0, 45.0, 0, 3.0, 8,this)

}

}

```

# Android Ethernet Examples

1.Add the following permissions in AndroidManifest.xml first:

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

2.Import GTSP\_L\_SDK SDK:

```
import com.gainscha.gtspl_sdk.GTSPLWiFiActivity;
```

3.Example:

```
public class MainActivity extends AppCompatActivity {

    GTSPLWiFiActivity mGtsplWiFiCmdTest = new GTSPLWiFiActivity();

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_main);

        mGtsplWiFiCmdTest.GTSPL_setCmdSendMode("P");

        mGtsplWiFiCmdTest.GTSPL_openPort("192.168.1.109", 8899);

        mGtsplWiFiCmdTest.GTSPL_setup(62, 45, 2, 6, 0, 3, 0, this);

        mGtsplWiFiCmdTest.GTSPL_setup(62.0, 45.0, 2, 6, 0, 3.0, 0.0, this);

        mGtsplWiFiCmdTest.GTSPL_sendCommand(this, "DIRECTION 1\n\n");

        mGtsplWiFiCmdTest.GTSPL_clearBuffer(this);

        mGtsplWiFiCmdTest.GTSPL_printerFont(100,10,"5",0,1,1,"Print Font 123456",this);

        mGtsplWiFiCmdTest.GTSPL_barcode(30, 30, "128", 100, 1, 0, 2, 2, "barcode1234567", this);

        mGtsplWiFiCmdTest.GTSPL_qrcode(300, 100, "H", 4, "A", 0, "ABCabc123", this);

        mGtsplWiFiCmdTest.GTSPL_downloadBMP("CIRCLE.BMP", this);

        mGtsplWiFiCmdTest.GTSPL_sendCommand(this, "PUTBMP 150,30,\"CIRCLE.BMP\"\n\n");

        mGtsplWiFiCmdTest.GTSPL_download_Not1BitDepthBMP("printTest4.BMP",

MainActivity.this);

        mGtsplWiFiCmdTest.GTSPL_sendCommand(MainActivity.this, "PUTBMP

10,10,\"PrintTest4.BMP\"\n\n");

        String sBlock = "We stand behind our products with one of the most comprehensive support
```

```

programs in the Auto-ID industry.";

mGtsplWIFICmdTest.GTSPL_printFontBlock("15", "15", "790", "90", "0", "0", "8", "8", "20",
"2",sBlock,this);

mGtsplWIFICmdTest.GTSPL_printLabel(1, 1, this);


// initialization

mGtsplWIFICmdTest.GTSPL_genericDefault(); // Initialize the general setting value
mGtsplWIFICmdTest.GTSPL_sensorDefault(); // Initialize the sensor setting value
mGtsplWIFICmdTest.GTSPL_rfidSetupDefault(); // Initialize the RFID setting value


//Modify Wifi frequency band

mGtsplWIFICmdTest.GTSPL_ WifiFrequency("5G");


//Set print function(It needs to use with print)

mGtsplWIFICmdTest.GTSPL_setDirectionAndMirror(1, 1); // Set direction and mirror
mGtsplWIFICmdTest.GTSPL_setShift(50); // Set the vertical displacement distance
mGtsplWIFICmdTest.GTSPL_printReverse(10, 10, 160, 160); //Set reverse
mGtsplWIFICmdTest.GTSPL_setOffset(20); // Set feed offset
mGtsplWIFICmdTest.GTSPL_setCutMode(0, 2); // Set cut mode and cut number
mGtsplWIFICmdTest.GTSPL_setAfterPrintAction(2); // Set action after print
mGtsplWIFICmdTest.GTSPL_printLabel(1, 1);


//Bitmap Print

mGtsplWIFICmdTest.GTSPL_printBMP(-500,30,400,300,1,"CIRCLE.BMP ",this);
mGtsplWIFICmdTest.GTSPL_printBMP_Compression(-500,30,400,300," CIRCLE.BMP ",this);
mGtsplWIFICmdTest.GTSPL_printLabel(1, 1);

```

```
//Bicolor Print
```

```
mGtsplWIFICmdTest.GTSPL_setBicolor("R",7, this);
```

```
//GEN2 RFID
```

```
mGtsplWIFICmdTest.GTSPL_writeUHF("H", 2, 12, "E", "414142424343444445454646", this);
```

```
mGtsplWIFICmdTest.GTSPL_EPCPWD_Action("L", "12345678", this);
```

```
mGtsplWIFICmdTest.GTSPL_TIDPWD_Action("L", "12345678", this);
```

```
mGtsplWIFICmdTest.GTSPL_USERPWD_Action("L", "12345678", this);
```

```
mGtsplWIFICmdTest.GTSPL_AccessPWD_Action("S", "12345678", this);
```

```
mGtsplWIFICmdTest.GTSPL_KillPWD_Action("S", "12345678", this);
```

```
mGtsplWIFICmdTest.GTSPL_Set_RFIDPorcedure(8, 8, 32, 3, "N", 2, 2, this);
```

```
mGtsplWIFICmdTest.GTSPL_Set_RFIDPorcedure(8, 8, 32, 3, "N", 2, 2, "203", this);
```

```
mGtsplWIFICmdTest.GTSPL_writeHF("H", 2, 12, "414142424343444445454646", this);
```

```
mGtsplWIFICmdTest.GTSPL_printLabel(1, 1, this);
```

```
String uhfData = mGtsplWIFICmdTest.GTSPL_readUHF("H",2,12,"E", this);
```

```
String uhfData = mGtsplWIFICmdTest.GTSPL_readUHFQ("H",0,0,this);
```

```
//GJB RFID set password
```

```
// Use write password, set a new read password
```

```
mGtsplWIFICmdTest.GTSPL_Set_GJB_Pwd("R","S","87654321","12345678",this);
```

```
// Use write password, set a new write password
```

```
mGtsplWIFICmdTest.GTSPL_Set_GJB_Pwd("W","S","87654321","12345678",this);
```

```
// Use write password, set a new kill password
```

```
mGtsplWIFICmdTest.GTSPL_Set_GJB_Pwd("K","S","87654321","12345678",this);
```

```
// Use write password, set a new status password
```

```
mGtsplWIFICmdTest.GTSPL_Set_GJB_Pwd("S","S","87654321","12345678",this);
```

```
mGtsplWIFICmdTest.GTSPL_printLabel(1, 1, this);
```

```

//GJB RFID set status

mGtsplWIFICmdTest.GTSPL_Set_GJB_Status("E","C","11112222",this);

mGtsplWIFICmdTest.GTSPL_printLabel(1, 1, this);


//GJB RFID write EPC data

mGtsplWIFICmdTest.GTSPL_writeGJB("H",2,12,"E","404041414242434344444545","123456
78",this);

mGtsplWIFICmdTest.GTSPL_printLabel(1, 1, this);


//GJB RFID read EPC data

String GJBData =

mGtsplWIFICmdTest.GTSPL_readGJB("H",2,12,"E","33334444",MainActivity.this);


//GJB RFID kill tag

mGtsplWIFICmdTest.GTSPL_Kill_GJB_Tag("11224455",this);

mGtsplWIFICmdTest.GTSPL_printLabel(1,1,this);


// Print Simplified Chinese

String stString="默认简体中文测试";

mGtsplWIFICmdTest.GTSPL_clearBuffer(this);

mGtsplWIFICmdTest.GTSPL_printerFont(100, 10, "TSS24.BF2", 0, 1, 1, stString, this);

mGtsplWIFICmdTest.GTSPL_printLabel(1, 1, this);


// Print traditional Chinese

String ttString="默認繁體中文測試";

mGtsplWIFICmdTest.GTSPL_clearBuffer(this);

```

```
mGtspiWIFICmdTest.GTSPL_printerFont(100, 10, " TST24.BF2", 0, 1, 1, ttString, this);

mGtspiWIFICmdTest.GTSPL_printLabel(1, 1, this);

String status = mGtspiWIFICmdTest.GTSPL_printersStatus(1000);

mGtspiWIFICmdTest.GTSPL_closePort();

String version=mGtspiWIFICmdTest.GTSPL_getSDKVersion(0,this);


//RFID auto calibration

mGtspiWIFICmdTest.GTSPL_LabelCalibration(this);


//UHFReaderE710 auto calibration

mGtspiWIFICmdTest.GTSPL_labelCalibration(62.0, 45.0, 0, 3.0, 8,this)
}
}
```

# Android USB Examples

1.Import GTSPL\_SDK:

```
import com.gainscha.gtspl_sdk.GTSPLUsbActivity;
```

2.Example:

```
public class MainActivity extends AppCompatActivity {

    GTSPLUsbActivity mUSB = new GTSPLUsbActivity();

    private static final String ACTION_USB_PERMISSION ="com.android.example.USB_PERMISSION";

    private static UsbManager mUsbManager;

    private static PendingIntent mPermissionIntent;

    private static boolean hasPermissionToCommunicate = false;

    private static UsbDevice mDevice;


    private final BroadcastReceiver mUsbReceiver = new BroadcastReceiver() {

        public void onReceive(Context context, Intent intent) {

            String action = intent.getAction();

            if (ACTION_USB_PERMISSION.equals(action)) {

                synchronized (this) {

                    UsbDevice device = intent.getParcelableExtra(UsbManager.EXTRA_DEVICE);

                    if (intent.getBooleanExtra(UsbManager.EXTRA_PERMISSION_GRANTED, false)) {

                        if (device != null) {hasPermissionToCommunicate = true;}}

                }

            }

        }

    };

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);
```

```

setContentView(R.layout.activity_main);

mUsbManager = (UsbManager) getSystemService(Context.USB_SERVICE);

mPermissionIntent = PendingIntent.getBroadcast(this, 0, new
Intent(ACTION_USB_PERMISSION), 0);

IntentFilter filter = new IntentFilter(ACTION_USB_PERMISSION);

registerReceiver(mUsbReceiver, filter);

HashMap<String, UsbDevice> deviceList = mUsbManager.getDeviceList();

Iterator<UsbDevice> deviceIterator = deviceList.values().iterator();

while (deviceIterator.hasNext()) {

    mDevice = deviceIterator.next();

    if (mDevice.getVendorId() == 1137) {break;}

}

mPermissionIntent = PendingIntent.getBroadcast(MainActivity.this, 0, new
Intent(ACTION_USB_PERMISSION), PendingIntent.FLAG_ONE_SHOT);

mUsbManager.requestPermission(mDevice, mPermissionIntent);

mUSB.GTSPL_setCmdSendMode("P");

mUSB.GTSPL_openPort(mUsbManager, mDevice);

mUSB.GTSPL_setup(62, 45, 2, 3, 0, 3, 0, this);

mUSB.GTSPL_setup(62.0, 45.0, 2, 3, 0, 3.0, 0.0, this);

mUSB.GTSPL_sendCommand(this, "DIRECTION 1\r\n");

mUSB.GTSPL_clearBuffer(this);

mUSB.GTSPL_barcode(30, 30, "128", 100, 1, 0, 2, 2, "barcode9463521", this);

mUSB.GTSPL_qrcode(300, 100, "H", 4, "A", 0, "ABCabc123", this);

mUSB.GTSPL_printerFont(100, 50, "2", 0, 1, 1, "PrintFontTest123", this);

mUSB.GTSPL_downloadBMP("LOGO.BMP", this);

mUSB.GTSPL_sendCommand(this, "PUTBMP 100,80,\"LOGO.BMP\"\\r\\n");

mUSB.GTSPL_download_Not1BitDepthBMP("printTest4.BMP", MainActivity.this);

```



```

mUSB.GTSPL_sendCommand(MainActivity.this, "PUTBMP10,10,\"PrintTest4.BMP\"\\r\\n");

String sBlock = "We stand behind our products with one of the most comprehensive
support programs in the Auto-ID industry.";

mUSB.GTSPL_printFontBlock("15", "15", "790", "90", "0", "0", "8", "8", "20",
"2",sBlock,this);

mUSB.GTSPL_printLabel(1, 1, this);

// initialization

mUSB.GTSPL_genericDefault(); // Initialize the general setting value
mUSB.GTSPL_sensorDefault(); // Initialize the sensor setting value
mUSB.GTSPL_rfidSetupDefault(); // Initialize the RFID setting value

//Modify Wifi frequency band

mUSB.GTSPL _ WifiFrequency("5G");

//Set print function (It needs to use with print)

mUSB.GTSPL_setDirectionAndMirror(1, 1); // Set direction and mirror.
mUSB.GTSPL_setShift(50); // Set the vertical displacement distance
mUSB.GTSPL_printReverse(10, 10, 160, 160); //Set reverse
mUSB.GTSPL_setOffset(20); // Set feed offset
mUSB.GTSPL_setCutMode(0, 2); // Set cut mode and cut number
mUSB.GTSPL_setAfterPrintAction(2); // Set action after print

mUSB.GTSPL_printLabel(1, 1);

//Bitmap Print

mUSB.GTSPL_printBMP(-500,30,400,300,1,"CIRCLE.BMP ",this);

```

```

mUSB.GTSPL_printBMP_Compression(-500,30,400,300," CIRCLE.BMP ",this);

mUSB.GTSPL_printLabel(1, 1);


//Bicolor Print

mUSB.GTSPL_setBicolor("R",7, this);


//GEN2 RFID

mUSB.GTSPL_writeUHF("H", 2, 12, "E", "414142424343444445454646", this);

mUSB.GTSPL_EPCPWD_Action("L", "12345678", this);

mUSB.GTSPL_TIDPWD_Action("L", "12345678", this);

mUSB.GTSPL_USERPWD_Action("L", "12345678", this);

mUSB.GTSPL_AccessPWD_Action("S", "12345678", this);

mUSB.GTSPL_KillPWD_Action ("S", "12345678", this);

mUSB.GTSPL_Set_RFIDPorcedure(8, 8, 32, 3, "N", 2, 2, this);

mUSB.GTSPL_Set_RFIDPorcedure(8, 8, 32, 3, "N", 2, 2, "203", this);

mUSB.GTSPL_writeHF("H", 2, 12, "414142424343444445454646", this);

mUSB.GTSPL_printLabel(1, 1, this);

String uhfData = mUSB.GTSPL_readUHF("H",2,12,"E", this);

String uhfData = mUSB.GTSPL_readUHF("H",2,12,"E", 4000, this);

String uhfData = mUSB.GTSPL_readUHFQ("H",0,0,this);


//GJB RFID set password

// Use write password, set a new read password

mUSB.GTSPL_Set_GJB_Pwd("R","S","87654321","12345678",this);

// Use write password, set a new read password

mUSB.GTSPL_Set_GJB_Pwd("W","S","87654321","12345678",this);

// Use write password, set a new kill password

```

```

mUSB.GTSPL_Set_GJB_Pwd("K","S","87654321","12345678",this);

// Use write password, set a new status password

mUSB.GTSPL_Set_GJB_Pwd("S","S","87654321","12345678",this);

mUSB.GTSPL_printLabel(1, 1, this);


//GJB RFID Set status

mUSB.GTSPL_Set_GJB_Status("E","C","11112222",this);

mUSB.GTSPL_printLabel(1, 1, this);


//GJB RFID write EPC data

mUSB.GTSPL_writeGJB("H",2,12,"E","404041414242434344444545","12345678",this);

mUSB.GTSPL_printLabel(1, 1, this);


//GJB RFID read EPC data

String GJBData = mUSB.GTSPL_readGJB("H",2,12,"E","33334444",MainActivity.this);


//GJB RFID kill tag

mUSB.GTSPL_Kill_GJB_Tag("11224455",this);

mUSB.GTSPL_printLabel(1,1,this);


// Print Simplified Chinese

String stString="默认简体中文测试";

mUSB.GTSPL_clearBuffer(this);

mUSB.GTSPL_printerFont(100, 10, "TSS24.BF2", 0, 1, 1, stString, this);

mUSB.GTSPL_printLabel(1, 1, this);


// Print traditional Chinese

```

```
String ttString="默認繁體中文測試";

mUSB.GTSPL_clearBuffer(this);

mUSB.GTSPL_printerFont(100, 10, " TST24.BF2", 0, 1, 1, ttString, this);

mUSB.GTSPL_printLabel(1, 1, this);

String status = mUSB.GTSPL_printersStatus(1000);

mUSB.GTSPL_closePort();

String version= mUSB.GTSPL_getSDKVersion(0,this);


//RFID auto calibration

mUSB.GTSPL_LabelCalibration(this);


//UHFReaderE710 auto calibration

mUSB.GTSPL_labelCalibration(62.0, 45.0, 0, 3.0, 8,this)

}
```

## Appendix 1

Code Type	Description	Narrow : Width					Max. data length
		1:1	1:2	1:3	2:5	3:7	
<b>128</b>	Code 128, switching code subset automatically.	V					
<b>128M</b>	Code 128, switching code subset manually.	V					
<b>EAN128</b>	EAN128, switching code subset automatically.	V					
<b>EAN128M</b>	EAN128M, switching code subset manually.	V					
<b>25</b>	Interleaved 2 of 5.		V	V	V		Length is even
<b>25C</b>	Interleaved 2 of 5 with check digit.		V	V	V		Length is odd
<b>25S</b>	Standard 2 of 5.		V	V	V		
<b>25I</b>	Industrial 2 of 5.		V	V	V		
<b>39</b>	Code 39, switching standard and full ASCII mode automatically.		V	V	V		
<b>39C</b>	Code 39 with check digit.		V	V	V		
<b>93</b>	Code 93.			V			
<b>EAN13</b>	EAN 13.	V					12
<b>EAN13+2</b>	EAN 13 with 2 digits add-on.	V					14
<b>EAN13+5</b>	EAN 13 with 5 digits add-on.	V					17
<b>EANB</b>	EAN 8.	V					7
<b>EANB+2</b>	EAN 8 with 2 digits add-on.	V					96
<b>EANB+5</b>	EAN 8 with 5 digits add-on.	V					12
<b>CODA</b>	Codabar.		V	V	V		
<b>POST</b>	Postnet.	V					5,9,11
<b>UPCA</b>	UPC-A.	V					11
<b>UPCA+2</b>	UPC-A with 2 digits add-on.	V					13
<b>UPA+5</b>	UPC-A with 5 digits add-on.	V					16
<b>UPCE</b>	UPC-E.	V					6
<b>UPCE+2</b>	UPC-E with 2 digits add-on.	V					8
<b>UPE+5</b>	UPC-E with 5 digits add-on.	V					11
<b>MSI</b>	MSI.		V	V	V		
<b>MSIC</b>	MSI with check digit.		V	V	V		
<b>PLESSEY</b>	PLESSEY.		V	V	V		

<b>CPOST</b>	China post.					V	
<b>ITF14</b>	ITF14.		V	V	V		13
<b>EAN14</b>	EAN14.	V					13
<b>11</b>	Code 11.		V	V	V		
<b>TELEPEN</b>	Telepen. *Since V6.89EZ.		V	V	V		
<b>TELEPENN</b>	Telepen number. *Since V6.89EZ.		V	V	V		
<b>PLANET</b>	Planet. *Since V6.89EZ.	V					
<b>CODE49</b>	Code 49. *Since V6.89EZ.	V					
<b>DPI</b>	Deutsche Post Identcode. *Since V6.91EZ.		V	V	V		11
<b>DPL</b>	Deutsche Post Leitcode. *Since V6.91EZ.		V	V	V		13
<b>LOGMARS</b>	A special use of Code 39. *Since V6.88EZ.		V	V	V		

## Appendix 2

Code Type	Description
<b>100</b>	other errors
<b>101</b>	memory space exceeded
<b>102</b>	memory is locked
<b>103</b>	Insufficient read power
<b>104</b>	non-specific error
<b>105</b>	CRC error
<b>106</b>	If an error occurs during writing , Reply how many words have been written
<b>107</b>	If the Tag tag replies with an error during writing , Error code plus how many words have been written
<b>108</b>	no tag exists
<b>109</b>	command format error
<b>110</b>	Failed to set power strength
<b>111</b>	Failed to set regulations

## Appendix 3

錯誤代碼	錯誤代碼說明
00	No electronic tag found
05	Wrong access password
FA	There is an electronic tag, but the communication is poor and the operation failed.
FB	No electronic tag is operable.
FC	Electronic tag returned error
FD	The command length is incorrect.
FE	Illegal command.
FF	Parameter error.