

# Windows SDK

## 说明手册(V1.23)



## GTSP.LDLL 函式库支持开发平台/语言

开发平台/语言	范例代码说明
C Sharp	<a href="#">第 25~34 页</a>
Java	<a href="#">第 34~46 页</a>
Javascript	<a href="#">第 46~48 页</a>
ASP.NET	<a href="#">第 48~50 页</a>
VB.NET	<a href="#">第 50~51 页</a>
Access-VBA	<a href="#">第 52~53 页</a>
Excel-VBA	<a href="#">第 54 页</a>
Visual C++	<a href="#">第 55~65 页</a>
FoxPro	<a href="#">第 66~67 页</a>
Borland C++ Builder 6	待开发
Delphi	待开发

# GTSP.L.DLL 函式库使用说明(V1.23)

## 1. openport(name)、openport\_USB()、openports\_USB(PrinterModel)、openport\_Ethernet(IP, port)

■ 函式说明：指定并开启计算机端的输出端口

■ 参数说明：

➔ name：字符串型别

(1) 单机打印时，请指定打印机驱动程序名称

(2) 若连接网络打印机，请指定打印机路径及共享打印机名称

如：\\192.168.1.104\ Printer Model

➔ PrinterModel：字符串型别，打印机机型名称

➔ IP：字符串型别，打印机 IP 位址

➔ port：int 型别，打印机连接埠号码

## 2. closeport()、closeport\_USB()、closeport\_Ethernet()

■ 函式说明：关闭输出埠

■ 参数说明：无

## 3. detectUSB\_USB()

■ 函式说明：侦测打印机 USB 接口插拔状况

■ 参数说明：无

■ 回传说明：字符串数组型别，以 USB 连接所有开启的打印机

## 4. detectUSBStr\_USB()

■ 函式说明：侦测打印机 USB 接口插拔状况

■ 参数说明：无

■ 回传说明：字符串型别，以 USB 连接所有开启的打印机

## 5. sendcommand(command)、sendcommand\_USB(command)、sendcommand\_Ethernet(command)

■ 函式说明：将内建命令传送至打印机(结尾会自动加入\r\n 命令结束用语)

■ 参数说明：

➔ command：字符串型别，设定指令内容，详细指令请参考 TSPL 使用手册

## 6. `sendcommand_Array(command)`、`sendcommand_Array_USB(command)`、

### `sendcommand_Array_Ethernet(command)`

■ 函式说明：将内建命令传送至打印机(结尾会自动加入\r\n 命令结束用语)

■ 参数说明：

➔ `command`：字节数组型别，设定指令内容，详细指令请参考 TSPL 使用手册

## 7. `sendcommand_noCIRf(command)`、`sendcommand_noCIRf_USB(command)`、

### `sendcommand_noCIRf_Ethernet(command)`

■ 函式说明：将内建命令传送至打印机(结尾不会自动加入\r\n 命令结束用语)

■ 参数说明：

➔ `command`：字符串型别，设定指令内容，详细指令请参考 TSPL 使用手册

## 8. `clearbuffer()`、`clearbuffer_USB()`、`clearbuffer_Ethernet()`

■ 函式说明：清除图像缓冲

■ 参数说明：无

## 9. `setup(width, height, speed, density, sensor, sensorDistance, sensorOffset)`、

### `setup_USB(width, height, speed, density, sensor, sensorDistance, sensorOffset)`、

### `setup_Ethernet(width, height, speed, density, sensor, sensorDistance, sensorOffset)`

■ 函式说明：设定卷标的宽度、高度、打印速度、打印热度、传感器类别、间隙/黑标垂直间距、间隙/黑标偏移距离

■ 参数说明：

参数	型别	说明
<code>width</code>	字符串	设定标签宽度，单位 mm
<code>height</code>	字符串	设定标签高度，单位 mm
<code>speed</code>	字符串	设定打印速度，1~15，代表每秒 1~15 吋打印速度(随机型不同会有不同打印最高上限，最高为每秒 15 吋打印速度)
<code>density</code>	字符串	设定打印浓度，0~15，数字越大打印结果越黑

<b>sensor</b>	字符串	设定使用传感器之类别： 0：表示使用间隙传感器(gap sensor) 1：表示使用黑标传感器(black mark sensor)
<b>sensorDistance</b>	字符串	设定间隙/黑标垂直间距高度，单位 mm
<b>sensorOffset</b>	字符串	设定间隙/黑标垂直间距高度，单位 mm，此参数若使用一般标签时均设为 0

## 10. barcode( x, y, type, height, readable, rotation, narrow, wide, content)、

barcode\_USB( x, y, type, height, readable, rotation, narrow, wide, content)、

barcode\_Ethernet(x, y, type, height, readable, rotation, narrow, wide, content)

■ 函式说明：使用打印机内建条形码打印

■ 参数说明：

参数	型别	说明
<b>x</b>	字符串	条形码 X 方向起始点，以点(dot)表示
<b>y</b>	字符串	条形码 Y 方向起始点，以点(dot)表示
<b>type</b>	字符串	设定条形码类型(Code Type) ， <a href="#">请参考附件</a>
<b>height</b>	字符串	设定条形码高度，高度以点来表示
<b>readable</b>	字符串	设定是否打印条形码码文 0:不打印 1:打印条形码码文置左 2:打印条形码码文置中 3:打印条形码码文置右
<b>rotation</b>	字符串	设定条形码旋转角度 0：旋转0度 90：旋转90度 180：旋转180度 270：旋转270度
<b>narrow</b>	字符串	设定条形码窄 bar 比例因子， <a href="#">请参考附件</a>
<b>wide</b>	字符串	设定条形码宽 bar 比例因子， <a href="#">请参考附件</a>
<b>content</b>	字符串	设定欲打印之条形码内容

# 11. qrcode(x, y, ECCLevel, cellWidth, mode, rotation, content)、

qrcode\_USB(x, y, ECCLevel, cellWidth, mode, rotation, content)、

qrcode\_Ethernet(x, y, ECCLevel, cellWidth, mode, rotation, content)

■ 函式说明：使用打印机打印 QRcode

■ 参数说明：

参数	型别	说明
<b>x</b>	字符串	QRCode X 方向起始点，以点(dot)表示
<b>y</b>	字符串	QRCode Y 方向起始点，以点(dot)表示
<b>ECCLevel</b>	字符串	容错率 L : 7% M : 15% Q : 25% H : 30%
<b>cellWidth</b>	字符串	设定 QRCode 大小，1~10
<b>mode</b>	字符串	设定自动或手动编码 A : 自动 M : 手动
<b>rotation</b>	字符串	设定QRCode旋转角度 0 : 旋转0度 90 : 旋转90度 180 : 旋转180度 270 : 旋转270度
<b>content</b>	字符串	设定数据内容 数据内容限制： 1) 数字数据: (数字 0~9) 2) 字母数据 数字 0-9 大写字母 A-Z 9 种其它字符: 空格, \$ % * + - . / : ) *如果” A” 是数据内容的第一个字符，那么数据内容将会被设置为字母数据。 *如果” N” 是数据内容的第一个字符，那么数据内容将会被设置为数字数据。 * “！” 用来转换数据的格式，” N” 、” A” 等数据类型可通过” ！” 来转换。

## 12. `printerfont(x, y, size, rotation, x_scale, y_scale, content)`、

`printerfont_USB(x, y, size, rotation, x_scale, y_scale, content)`、

`printerfont_Ethernet(x, y, size, rotation, x_scale, y_scale, content)`

■ 函式说明：使用打印机内建文字打印

■ 参数说明：

参数	型别	说明
<b>x</b>	字符串	文字 X 方向起始点，以点(dot)表示
<b>y</b>	字符串	文字 Y 方向起始点，以点(dot)表示
<b>size</b>	字符串	内建字型名称 1: 8*/12 dots 2: 12*20 dots 3: 16*24 dots 4: 24*32 dots 5: 32*48 dots TST24.BF2: 繁体中文24*24 TST16.BF2: 繁体中文16*16 TSS24.BF2: 简体中文24*24 TSS16.BF2: 简体中文16*16
<b>rotation</b>	字符串	设定文字旋转角度 0: 旋转0度 90: 旋转90度 180: 旋转180度 270: 旋转 270 度
<b>x_scale</b>	字符串	设定文字 X 方向放大倍率，1~10
<b>y_scale</b>	字符串	设定文字Y方向放大倍率，1~10
<b>content</b>	字符串	设定欲打印之文字内容

## 13. `formfeed()`、`formfeed_USB()`、`formfeed_Ethernet()`

■ 函式说明：跳页，该函式需在 `setup` 后使用

■ 参数说明：无

#### 14. nobackfeed()、nobackfeed\_USB()、nobackfeed\_Ethernet()

- 函式说明：设定纸张不回吐
- 参数说明：无

#### 15. printlabel (set, copy)、printlabel \_USB(set, copy)、printlabel \_Ethernet(set, copy)

- 函式说明：打印标签内容
- 参数说明：
  - ➔ set：字符串型别，设定打印标签式数(set)
  - ➔ copy：字符串型别，设定打印标签份数(copy)

#### 16. downloadpcx (filename,memoryname)、downloadpcx\_USB (filename,memoryname)、downloadpcx\_Ethernet(filename,memoryname)

- 函式说明：下载单色 PCX 格式图文件至打印机
- 参数说明：
  - ➔ filename：字符串型别，文件名(可包含路径)
  - ➔ memoryname：下载至打印机内存内之文件名(请使用大写档名)

#### 17. downloadbmp (filename, memoryname)、downloadbmp\_USB (filename, memoryname)、downloadbmp\_Ethernet (filename, memoryname)

- 函式说明：下载单色 BMP 格式图文件至打印机
- 参数说明：
  - ➔ filename：字符串型别，文件名(可包含路径)
  - ➔ memoryname：下载至打印机内存内之文件名(请使用大写档名)



## 18. windowsfont (x,y,height,rotation,fontstyle,underline,fontname,content)

windowsfont\_USB (x,y,height,rotation,fontstyle,underline,fontname,content)

windowsfont\_Ethernet(x,y,height,rotation,fontstyle,underline,fontname,content)

■ 函式说明：使用 Windows TTF 字型打印文字

■ 参数说明：

参数	型别	说明
<b>x</b>	int	文字 X 方向起始点，以点(dot)表示
<b>y</b>	int	文字 Y 方向起始点，以点(dot)表示
<b>height</b>	int	字体高度，以点(dot)表示
<b>rotation</b>	int	设定旋转角度，逆时钟方向旋转 0：旋转0度 90：旋转90度 180：旋转180度 270：旋转 270 度
<b>fontstyle</b>	int	设定字体外型 0：标准(Normal) 1：斜体(Italic) 2：粗体(Bold) 3：粗斜体(Bold and Italic)
<b>underline</b>	int	设定底线 0：无底线 1：加底线
<b>fontname</b>	字符串	Windows字体名称，如: Arial, Times new Roman, 细明体, 标楷体
<b>content</b>	字符串	设定欲打印之文字内容

## 19. getDLLVersion (returnWay)、getDLLVersion\_USB(returnWay)、getDLLVersion\_Ethernet(returnWay)

■ 函式说明：回传此 SDK 版本号

■ 参数说明：

➔ retrunWay: int 型别，输入 0 除返回 SDK 版本号外，会跳出 SDK 版本讯息

## 20. printerstatus\_USB ()、printerstatus\_Ethernet()

- 函式说明：回传打印机状态，需用字符串变量接收回传讯息
- 参数说明：无
- 回传字符串说明：

回传字符串	打印机状态
00	就绪
01	上盖开启
02	卡纸
03	卡纸且上盖开启
04	标签用尽
05	标签用尽且上盖开启
08	碳带用尽
09	碳带用尽且上盖开启
0A	碳带用尽且卡纸
0B	碳带用尽、卡纸且上盖开启
0C	碳带用尽且标签用尽
0D	碳带用尽、标签用尽且上盖开启
10	暂停
20	打印中
80	其他错误

## 21. writeUHF (dataFormat,startBlockNo,byteSize,Gen2MemoryBank,datastring)

**writeUHF\_USB (dataFormat,startBlockNo,byteSize,Gen2MemoryBank,datastring)**

**writeUHF\_Ethernet (dataFormat,startBlockNo,byteSize,Gen2MemoryBank,datastring)**

- 函式说明：将数据写入 UHF Gen2/UHF GJB 卷标内存中
- 参数说明：

参数	型别	说明
<b>dataFormat</b>	string	设定字符串数据编码格式，默认为 H A: ASCII

		H: Hexadecimal
<b>startBlockNo</b>	int	设定数据区块起始位置，Gen2 默认为 2，GJB 默认为 1
<b>byteSize</b>	int	设定写入数据byte长度，默认为1
<b>Gen2Memory Bank</b>	string	设定 Gen2/GJB 数据区段，默认为 E R: 保留 E: EPC T: TID(Tag ID) U: User
<b>datastring</b>	string	欲写入之字符串数据

## 22. EPCPWD\_Action (action, password)、EPCPWD\_Action\_USB (action, password)、

### EPCPWD\_Action\_Ethernet (action, password)

- 函式说明：将 UHF GNE2 的 EPC 资料区块上锁或解锁
- 参数说明：

参数	型别	说明
<b>action</b>	string	设定执行动作 U: 解锁资料区块 L: 上锁资料区块 O: 永久解锁资料区块 P: 永久上锁资料区块
<b>password</b>	string	密码，应为 8 hex 字符(0~9,A,B,C,D,E,F)

## 23. TIDPWD\_Action(action, password)、TIDPWD\_Action\_USB (action, password)、

### TIDPWD\_Action\_Ethernet(action, password)

- 函式说明：将 UHF GNE2 的 TID 资料区块上锁或解锁
- 参数说明：

参数	型别	说明
<b>action</b>	string	设定执行动作 U: 解锁资料区块 L: 上锁资料区块 O: 永久解锁资料区块

		P: 永久上锁资料区块
<b>password</b>	string	密码，应为 8 hex 字符(0~9,A,B,C,D,E,F)

## 24. USERPWD\_Action (action, password)、USERPWD\_Action\_USB (action, password)、

### USERPWD\_Action\_Ethernet (action, password)

- 函式说明：将 UHF GNE2 的 USER 资料区块上锁或解锁
- 参数说明：

参数	型别	说明
<b>action</b>	string	设定执行动作 U: 解锁资料区块 L: 上锁资料区块 O: 永久解锁资料区块 P: 永久上锁资料区块
<b>password</b>	string	密码，应为 8 hex 字符(0~9,A,B,C,D,E,F)

## 25. AccessPWD\_Action (action, password)、AccessPWD\_Action\_USB (action, password)、

### AccessPWD\_Action\_Ethernet (action, password)

- 函式说明：将 UHF GNE2/UHF GJB 的访问密码进行设定、上锁或解锁
- 参数说明：

参数	型别	说明
<b>action</b>	string	设定执行动作 UHF GNE2 : U: 解锁访问密码 L: 上锁访问密码 O: 永久解锁访问密码 P: 永久上锁访问密码 S: 设定访问密码 UHF GJB : U: 解锁访问密码 S: 设定访问密码
<b>password</b>	string	密码，应为 8 hex 字符(0~9,A,B,C,D,E,F)

## 26. KillPWD\_Action (action, password)、KillPWD\_Action\_USB (action, password)、

### KillPWD\_Action\_Ethernet (action, password)

- 函式说明：将 UHF GNE2 的删除密码进行设定、上锁或解锁
- 参数说明：

参数	型别	说明
action	string	设定执行动作 U: 解锁删除密码 L: 上锁删除密码 O: 永久删除访问密码 P: 永久删除访问密码 S: 设定删除密码
password	string	密码，应为 8 hex 字符(0~9,A,B,C,D,E,F)

## 27. Set\_RFIDPorcedure (tagType, rw\_position, void\_printout, tryEncodie\_times,error\_handle, speed, retry\_times)

### Set\_RFIDPorcedure\_USB (tagType, rw\_position, void\_printout, tryEncodie\_times,error\_handle, speed, retry\_times)

### Set\_RFIDPorcedure\_Ethernet (tagType, rw\_position, void\_printout, tryEncodie\_times,error\_handle, speed, retry\_times)

- 函式说明：RFID 设定
- 参数说明：

参数	型别	说明
tagType	int	设定卷标类型，1~10，默认值为 8 1：EPC Class 1 Generation 2-Q，8：EPC Class 1 Generation 2-R，10：UHF-J
rw_position	int	设标签读写位置(卷标顶部起算)，范围为 0~9999(dot)，预设 0
void_printout	int	设定无效打印长度(dot)，范围为0~卷标长度，默认为卷标长度
tryEncodie_times	string	设定最大无效标签数，范围为 0~10，预设 3
error_handle	string	设定无效时采取的动作，预设 N

		N: No action(继续) P: Pause mode(暂停) E: Error mode(停止)
<b>speed</b>	int	设定无效打印速度，范围 2~10(IPS)，默认值 2(IPS)
<b>retry_times</b>	int	设定标签重试次数，范围 0~10，默认值 6

**28. Set\_RFIDPorcedure (tagType, rw\_position, void\_printout, tryEncodie\_times,error\_handle, speed, retry\_times,dpi)**

**Set\_RFIDPorcedure\_mm\_USB (tagType, rw\_position, void\_printout, tryEncodie\_times,error\_handle, speed, retry\_times,dpi)**

**Set\_RFIDPorcedure\_mm\_Ethernet (tagType, rw\_position, void\_printout, tryEncodie\_times,error\_handle, speed, retry\_times,dpi)**

■ 函式说明：RFID 设定

■ 参数说明：

参数	型别	说明
<b>tagType</b>	int	设定卷标类型，1~10，默认值为 8 1：EPC Class 1 Generation 2-Q, 8：EPC Class 1 Generation 2-R, 10：UHF-J
<b>rw_position</b>	double	设卷标读写位置(卷标顶部起算)， 范围为 203dpi:0 ~ 1251 (mm)、 300dpi:0 ~ 846 (mm)、 600dpi:0 ~ 423 (mm)，预设为 0
<b>void_printout</b>	double	设定无效打印长度(mm)，范围为0~卷标长度，默认为卷标长度
<b>tryEncodie_times</b>	int	设定最大无效标签数，范围为 0~10，预设为 3
<b>error_handle</b>	字符串	设定无效时采取的动作，预设为 N N：No action(继续) P：Pause mode(暂停) E：Error mode(停止)
<b>speed</b>	int	设定无效打印速度，范围 2~10(IPS)，默认值 2(IPS)
<b>retry_times</b>	int	设定标签重试次数，范围 0~10，默认值 6
<b>dpi</b>	字串	设定打印机的 DPI 203: 203 dpi 300: 300 dpi 600: 600 dpi

## 29. writeHF (dataFormat,startBlockNo,byteSize,datastring)

**writeHF\_USB (dataFormat,startBlockNo,byteSize,datastring)**

**writeHF\_Ethernet (dataFormat,startBlockNo,byteSize,datastring)**

■ 函式说明：将数据写入 HF 卷标内存中

■ 参数说明：

参数	型别	说明
<b>dataFormat</b>	string	设定字符串数据编码格式，默认为 H A: ASCII H: Hexadecimal
<b>startBlockNo</b>	int	设定数据区块起始位置，默认为 2
<b>byteSize</b>	int	设定写入数据byte长度，默认为1
<b>datastring</b>	string	欲写入之字符串数据

## 30. printerfontblock (x, y, width, height, fontname, rotation, x\_scale, y\_scale, space, align, content) 、

**printerfontblock\_USB(x, y, width, height, fontname, rotation, x\_scale, y\_scale,space, align,content) 、**

**printerfontblock\_Ethernet(x, y, width, height, fontname, rotation, x\_scale, y\_scale, space, align, content)**

■ 函式说明：打印段落文字内容

■ 参数说明：

参数	型别	说明
<b>x</b>	字符串	文字 X 方向起始点，以点(dot)表示
<b>y</b>	字符串	文字 Y 方向起始点，以点(dot)表示
<b>width</b>	字符串	设定段落区块宽度，以点(dot)表示
<b>height</b>	字符串	设定段落区块高度，以点(dot)表示
<b>fontname</b>	字符串	内建字型名称 1: 8*/12 dots 2: 12*20 dots 3: 16*24 dots 4: 24*32 dots 5: 32*48 dots TST24.BF2: 繁体中文24*24 TST16.BF2: 繁体中文16*16

		TSS24.BF2: 简体中文24*24 TSS16.BF2: 简体中文16*16
<b>rotation</b>	字符串	设定旋转角度，逆时针方向旋转 0: 旋转0度 90: 旋转90度 180: 旋转180度 270: 旋转 270 度
<b>x_scale</b>	字符串	设定文字 X 方向放大倍率，1~10
<b>y_scale</b>	字符串	设定文字Y方向放大倍率，1~10
<b>space</b>	字符串	行距，以点(dot)表示
<b>align</b>	字符串	对齐位置 0: 预设(置左) 1: 置左 2: 置中 3: 置右
<b>content</b>	字符串	欲写入之字符串数据

### 31. readUHF\_USB (dataFormat,startBlockNo,byteSize,Gen2MemoryBank)

#### readUHF\_Ethernet (dataFormat,startBlockNo,byteSize,Gen2MemoryBank)

■ 函式说明：读取 UHF Gen2/UHF GJB 卷标内存数据

■ 参数说明：

参数	型别	说明
<b>dataFormat</b>	string	设定字符串数据编码格式，默认为 H A: ASCII H: Hexadecimal
<b>startBlockNo</b>	int	设定数据区块起始位置，默认为 0
<b>byteSize</b>	int	设定读取数据byte长度，默认为1
<b>Gen2Memory Bank</b>	string	设定 Gen2/GJB 数据区段，默认为 E R: 保留 E: EPC T: TID(Tag ID) U: User

■ 回传字符串说明(标签资料):



dataFormat	回传字符串(范例)
A	标签资料以 ASCII 显示 (ex: 24051324000103456400)
H	标签资料以 Hexadecimal 显示 (ex: 3234303531333234303030313033343536343030)

■ 回传字符串说明(错误代码)：

回传字符串	错误代码说明
64000000000000000000000000000000	其他错误
65000000000000000000000000000000	超过记忆体空间
66000000000000000000000000000000	记忆体被锁住
67000000000000000000000000000000	读取功率不足
68000000000000000000000000000000	非特定的错误
69000000000000000000000000000000	CRC错误
6A000000000000000000000000000000	写入中若发生错误时，回覆已写入多少 words 数
6B000000000000000000000000000000	写入中若 Tag 标签回覆错误时，错误码加上已写入多少 words 数
6C000000000000000000000000000000	没有标签存在
6D000000000000000000000000000000	指令格式错误
6E000000000000000000000000000000	设定电源强度失败
6F000000000000000000000000000000	设定法规失败

### 32. setPWD\_Action(passwordArea, action, NewPassword, WritePassword)

setPWD\_Action\_USB(passwordArea, action, NewPassword, WritePassword)

setPWD\_Action\_Ethernet(passwordArea, action, NewPassword, WritePassword)

■ 函式说明：设定 UHF GJB 各密码区域新密码

■ 参数说明：

参数	型别	说明
passwordArea	string	设定密码区域，预设为 W K：Kill W：Write R：Read S：Status
action	string	设定动作

		S : Set Password
<b>NewPassword</b>	string	设定密码区域的新密码，应为8 hex字元(0~9,A,B,C,D,E,F)
<b>WritePassword</b>	string	写入密码，应为 8 hex 字元(0~9,A,B,C,D,E,F)

33. **writeGJB\_UHF(dataFormat, startBlockNo, byteSize, Gen2MemoryBank, datastring, WritePassword)**  
**writeGJB\_UHF\_USB(dataFormat, startBlockNo, byteSize, Gen2MemoryBank, datastring, WritePassword)**  
**writeGJB\_UHF\_Ethernet(dataFormat, startBlockNo, byteSize, Gen2MemoryBank, datastring, WritePassword)**

■ 函式说明：将资料写入 UHF GJB 标签记忆体中

■ 参数说明：

参数	型别	说明
<b>dataFormat</b>	string	设定字串资料编码格式，预设 H A : ASCII H : Hexadecimal
<b>startBlockNo</b>	int	设定资料区块起始位置，GJB 预设 1
<b>byteSize</b>	int	设定写入资料byte长度，预设1
<b>Gen2MemoryBank</b>	string	设定 GJB 资料区段，预设 E R : 安全区 E : EPC T : TID(Tag ID) U : User
<b>datastring</b>	string	欲写入之字串资料
<b>WritePassword</b>	string	写入密码，应为 8 hex 字元(0~9,A,B,C,D,E,F)

34. **readGJB\_UHF\_USB(dataFormat, startBlockNo, byteSize, Gen2MemoryBank, ReadPassword)**  
**readGJB\_UHF\_Ethernet(dataFormat, startBlockNo, byteSize, Gen2MemoryBank, ReadPassword)**

■ 函式说明：读取 UHF GJB 标签记忆体资料，需用字符串变数接收回传讯息

■ 参数说明：

参数	型别	说明
<b>dataFormat</b>	string	设定字串资料编码格式，预设 H

		A : ASCII H : Hexadecimal
<b>startBlockNo</b>	int	设定资料区块起始位置，预设 0
<b>byteSize</b>	int	设定写入资料byte长度，预设 1
<b>Gen2MemoryBank</b>	string	设定 GJB 资料区段，预设 E R : 安全区 E : EPC T : TID(Tag ID) U : User
<b>ReadPassword</b>	string	读取密码，应为 8 hex 字元(0~9,A,B,C,D,E,F)

■ 回传字符串说明：

dataFormat	回传字符串(范例)
<b>A</b>	标签资料以 ASCII 显示 (ex: 24051324000103456400)
<b>H</b>	标签资料以 Hexadecimal 显示 (ex: 3234303531333234303030313033343536343030)

### 35. statusGJB\_UHF(Gen2MemoryBank, action, StatusPassword)

statusGJB\_UHF\_USB(Gen2MemoryBank, action, StatusPassword)

statusGJB\_UHF\_Ethernet(Gen2MemoryBank, action, StatusPassword)

■ 函式说明：设定 UHF GJB 各资料区块读写状态

■ 参数说明：

参数	型别	说明
<b>Gen2MemoryBank</b>	string	设定 GJB 资料区段，预设 E F : 安全区 E : EPC T : TID(Tag ID) U : User
<b>action</b>	string	设定状态，预设 A A=Lock0(可读可写) B=Lock1(可读不可写) C=Lock2(不可读可写) D=Lock3(不可读不可写)
<b>StatusPassword</b>	string	状态密码，应为 8 hex字元(0~9,A,B,C,D,E,F)

### 36. killGJB\_UHF(KillPassword)

**killGJB\_UHF\_USB(KillPassword)**

**killGJB\_UHF\_Ethernet(KillPassword)**

■ 函式说明：删除 UHF GJB 标签

■ 参数说明：

参数	型别	说明
<b>KillPassword</b>	string	删除密码，应为 8 hex 字元(0~9,A,B,C,D,E,F)

### 37. query\_UHF\_USB(dataFormat, PCReturnStatus, CRCReturnStatus)

**query\_UHF\_Ethernet (dataFormat, PCReturnStatus, CRCReturnStatus)**

■ 函式说明：以 Q 指令读取 UHF Gen2 标签记忆体 EPC 资料区段资料，需用字串变数接收回传讯息

■ 参数说明：

参数	型别	说明
<b>dataFormat</b>	string	设定字串资料编码格式，预设 H A：ASCII H：Hexadecimal
<b>PCReturnStatus</b>	int	PC 返回状态，预设 0 0：不回传 PC 值 1：回传 PC 值
<b>CRCReturnStatus</b>	int	CRC-16 返回状态，预设 0 0：不回传 CRC-16 1：回传CRC-16

■ 回传字符串说明(标签资料)：

以 query\_UHF\_USB("A", 1, 1)和 query\_UHF\_USB("H", 1, 1)为例：(PC 值和 CRC-16 皆回传)

dataFormat	回传字符串(范例)
<b>A</b>	标签资料以 ASCII 显示 (ex: 24051324000103456400)
<b>H</b>	标签资料以 Hexadecimal 显示 (ex: 3234303531333234303030313033343536343030)

以 query\_UHF\_USB("A", 0, 0)和 query\_UHF\_USB("H", 0, 0)为例：(PC 值和 CRC-16 皆不回传)

dataFormat	回传字符串(范例)
------------	-----------

<b>A</b>	标签资料以 ASCII 显示 (ex: 0513240001034564)
<b>H</b>	标签资料以 Hexadecimal 显示 (ex: 30353133323430303031303334353634)

■ 回传字符串说明(错误代码)：

回传字符串	错误代码说明
<b>64000000000000000000000000000000</b>	其他错误
<b>65000000000000000000000000000000</b>	超过记忆体空间
<b>66000000000000000000000000000000</b>	记忆体被锁住
<b>67000000000000000000000000000000</b>	读取功率不足
<b>68000000000000000000000000000000</b>	非特定的错误
<b>69000000000000000000000000000000</b>	CRC错误
<b>6A000000000000000000000000000000</b>	写入中若发生错误时，回覆已写入多少 words 数
<b>6B000000000000000000000000000000</b>	写入中若 Tag 标签回覆错误时，错误码加上已写入多少 words 数
<b>6C000000000000000000000000000000</b>	没有标签存在
<b>6D000000000000000000000000000000</b>	指令格式错误
<b>6E000000000000000000000000000000</b>	设定电源强度失败
<b>6F000000000000000000000000000000</b>	设定法规失败

### 38. RFIDAutoCalibration()、RFIDAutoCalibration\_USB()、RFIDAutoCalibration\_Ethernet()

- 函式说明：进行 RFID 标签自动校准至最佳位置
- 参数说明：无

### 39. setDirectionAndMirror(direction,mirror)、setDirectionAndMirror\_USB(direction,mirror)、setDirectionAndMirror\_Ethernet(direction,mirror)

- 函式说明：设定标签打印时的出纸方向与是否使用镜像打印
- 参数说明：

参数	型别	说明
<b>direction</b>	int	设定出纸方向，预设 0 0：顶端出纸 1：底端出纸
<b>mirror</b>	int	设定是否镜像打印

		0：否 1：是
--	--	------------

#### 40. setShift(shiftY)、setShift\_USB(shiftY)、setShift\_Ethernet(shiftY)

- 函式说明：设定图像垂直位移距离，数值为正时，图像会往打印方向移动，数值为负时，图像会背离打印方向
- 参数说明：
  - ➔ shiftY：int 型别，垂直位移距离，单位为 dot

#### 41. printReverse(x\_start, y\_start, x\_width, y\_height)、printReverse\_USB(x\_start, y\_start, x\_width, y\_height)、printReverse\_Ethernet(x\_start, y\_start, x\_width, y\_height)

- 函式说明：将指定的区域于打印时反白
- 参数说明：

参数	型别	说明
x_start	int	指定 X 起始坐标位置，以点(dot)表示
y_start	int	指定 Y 起始坐标位置，以点(dot)表示
x_width	int	指定 X 坐标宽度，以点(dot)表示
y_height	int	指定 Y 坐标高度，以点(dot)表示

#### 42. setOffset(offset)、setOffset\_USB(offset)、setOffset\_Ethernet(offset)

- 函式说明：设定每次出纸后额外偏移的距离(通常与剥纸模式和裁切模式组合使用)
- 参数说明：
  - ➔ offset：double 型别，额外的出纸偏移，单位为 mm

#### 43. setCutMode(mode, piece)、setCutMode\_USB(piece)、setCutMode\_Ethernet(mode, piece)

- 函式说明：设定裁切模式与张数
- 参数说明：

参数	型别	说明
mode	int	设定裁切方式，预设为 1 0：反切 1：正切
piece	int	设定裁切张数

#### 44. setAfterPrintAction(mode)、setAfterPrintAction\_USB(mode)、setAfterPrintAction\_Ethernet(mode)

- 函式说明：设定打印后动作
- 参数说明：

参数	型别	说明
<b>mode</b>	int	设定打印后动作，预设 1 0：停在原地 1：撕纸 2：剥纸 3：裁切

#### 45. genericDefault ()、genericDefault\_USB ()、genericDefault\_Ethernet()

- 函式说明：将打印机之一般设定值初始化
- 参数说明：无

#### 46. sensorDefault ()、sensorDefault\_USB ()、sensorDefault\_Ethernet ()

- 函式说明：将打印机之传感器设定值初始化
- 参数说明：无

#### 47. rfidSetupDefault ()、rfidSetupDefault\_USB ()、rfidSetupDefault\_Ethernet ()

- 函式说明：将 RFID 设定值初始化
- 参数说明：无

#### 48. WifiFrequency(Frequency)、WifiFrequency\_USB(Frequency)、WifiFrequency\_Ethernet (Frequency)

- 函式说明：使用兼容 5G 频段 WIFI 模块时，可用于切换使用频段
- 参数说明：

参数	型别	说明
<b>Frequency</b>	string	设定模块频段 2.4G：使用 2.4G 频段 5G：使用 5G 频段 BOTH：使用双频频段

## 49. Bitmap(x,y, width,height,mode,filename)

**Bitmap\_USB(x,y, width,height,mode,filename)**

**Bitmap\_Ethernet(x,y, width,height,mode,filename)**

■ 函式说明：将图片转为单色点阵图，使用打印机直接打印

■ 参数说明：

参数	型别	说明
<b>x</b>	string	文字 X 方向起始点，以点(dot)表示
<b>y</b>	string	文字 Y 方向起始点，以点(dot)表示
<b>width</b>	int	图片宽度，以字节(byte)表示
<b>height</b>	int	图片高度，以点(dot)表示
<b>mode</b>	int	图片格式 0: OVERWRITE 1: OR 2: XOR
<b>filename</b>	string	档案名称(可包含路径) 图档仅支援以下格式： 1. BMP (Bitmap)：位图格式 2. JPG (JPEG)：压缩的图像格式 3. PNG (Portable Network Graphics)：无损压缩的图像格式 4. GIF (Graphics Interchange Format)：支援多张图片的格式，通常用于动画 5. TIFF (Tagged Image File Format)：高品质的无损压缩图像格式 6. ICO (Icon)：图示格式，用于显示档案、程式或资料夹的图示 7. WMF (Windows Metafile)：Windows 绘图文件格式 8. EMF (Enhanced Metafile)：扩展的 Windows 绘图文件格式



## 50. compressBitmap(x,y, width,height,filename)

**compressBitmap\_USB(x,y, width,height,filename)**

**compressBitmap\_Ethernet(x,y, width,height,filename)**

- 函式说明：将图片转为单色点阵图，压缩后再使用打印机打印
- 参数说明：

参数	型别	说明
<b>x</b>	string	文字 X 方向起始点，以点(dot)表示
<b>y</b>	string	文字 Y 方向起始点，以点(dot)表示
<b>width</b>	int	图片宽度，以字节(byte)表示
<b>height</b>	int	图片高度，以点(dot)表示
<b>filename</b>	string	档案名称(可包含路径) 图档仅支援以下格式： 1. BMP (Bitmap)：位图格式 2. JPG (JPEG)：压缩的图像格式 3. PNG (Portable Network Graphics)：无损压缩的图像格式 4. GIF (Graphics Interchange Format)：支援多张图片的格式，通常用于动画 5. TIFF (Tagged Image File Format)：高品质的无损压缩图像格式 6. ICO (Icon)：图示格式，用于显示档案、程式或资料夹的图示 7. WMF (Windows Metafile)：Windows 绘图文件格式 8. EMF (Enhanced Metafile)：扩展的 Windows 绘图文件格式

## ● C#(.Netframework)

1.需先将 GTSPL\_SDK.dll 加入参考

2.汇入 GTSPL\_SDK:

```
using GTSPL_SDK;
```

3.范例程序:

```
//单机打印
```

```
Driver driver = new Driver();
```

```
driver.openport("Printer Model");
```

```
driver.setup("54", "30", "2", "3", "0", "3", "0");
```

```
driver.sendcommand("DIRECTION 1");
```

```
driver.setDirectionAndMirror(1, 1);
```

```
driver.setShift(50);
```

```
driver.setAfterPrintAction(2);
```

```
driver.setOffset(20);
```

```
driver.setCutMode(0,2);
```

```
driver.clearbuffer();
```

```
driver.sendcommand("TEXT 100,100,\"3\",0,1,1,\"REVERSE\");
```

```
driver.printReverse(90, 90, 128, 40);
```

```
driver.barcode("30", "30", "128", "100", "1", "0", "2", "2", "barcode1234567");
```

```
driver.qrcode("220", "260", "M", "3", "A", "0", "AABCB03abcN123");
```

```
driver.printerfont("50", "10", "3", "0", "1", "1", "Print Font 123456");
```

```
driver.printerfontblock("35", "15", "790", "90", "1", "0", "8", "8", "0", "1", "We stand behind our products  
with one of the most comprehensive support programs in the Auto-ID industry.");
```

```
string file = Environment.CurrentDirectory;
```

```

driver.downloadbmp(file + "\\CIRCLE.BMP", "CIRCLE.BMP");

driver.sendcommand("PUTBMP 150,30,\"CIRCLE.BMP\"");

driver.windowfont(100, 20, 48, 0, 0, 0, "arial", "C# Driver test");

//BitmapResult 为 Bitmap 处理结果，成功为 1；失败为 0

int BitmapResult = driver.Bitmap("-500", "70", 400, 350, 1, "Thunder.png ");

BitmapResult = driver.compressBitmap("-600", "70", 666, 357, "Cat1.jpg");

driver.printlabel("1", "1");

driver.genericDefault();

driver.sensorDefault();

driver.WifiFrequency("5G");

driver.writeUHF("H", 2, 12, "E", "414142424343444445454646"); //startBlockNo: Gen2 默认为 2

driver.printlabel("1", "1");

driver.EPCPWD_Action("U", "12345678");

driver.TIDPWD_Action("L", "12345678");

driver.USERPWD_Action("L", "12345678");

driver.AccessPWD_Action("S", "12345678");

driver.KillPWD_Action("L", "12345678");

driver.Set_RFIDPorcedure(8, 8, 32, 3, "N", 2, 2);

driver.Set_RFIDPorcedure_mm(5, 40, 32, 5, "N", 5, 5, "203");

driver.writeHF("H", 2, 12, "414142424343444445454646");

driver.printlabel("1", "1");

//RFID

driver.RFIDAutoCalibration();

driver.rfidSetupDefault();

//UHF GJB 写入资料

```

```
driver.writeGJB_UHF("H", 1, 12, "E", "414142424343444445454646", "12345678");
```

\*带入写入密码，写入资料(startBlockNo: GJB 预设为 1)

//UHF GJB 设定各密码区域新密码

```
driver.setPWD_Action("S", "S", "11112222", "12345678"); //带入写入密码，设定新状态密码
```

```
driver.setPWD_Action("R", "S", "33334444", "12345678"); //带入写入密码，设定新读取密码
```

```
driver.setPWD_Action("K", "S", "55556666", "12345678"); //带入写入密码，设定新删除密码
```

```
driver.setPWD_Action("W", "S", "87654321", "12345678"); //带入旧写入密码，设定新写入密码
```

//UHF GJB 设定资料区块状态

```
driver.statusGJB_UHF("E", "B", "11112222"); //带入状态密码，设定状态
```

```
driver.printlabel("1", "1");
```

//UHF GJB 删除标签

```
driver.killGJB_UHF ("55556666"); //带入删除密码，删除标签
```

```
driver.printlabel("1", "1");
```

```
driver.closeport();
```

//指定 USB 传输接口

```
USB usb = new USB();
```

```
usb.openport_USB();
```

```
string[] PrinterName = usb.detectUSB_USB(); //假设侦测打印机有一台以上
```

```
string[] PrinterName = usb.detectUSBStr_USB() == null ? null : usb.detectUSBStr_USB().Split('\n');
```

\*先判断 `usb.detectUSBStr_USB()` 是否为 null，若不是，再以 '\n' 区分各个机型

```
usb.openports_USB(PrinterName[0]); //利用侦测到的第一台打印机连线
```

```
usb.setup_USB("54", "30", "2", "3", "0", "3", "0");
```

```
usb.sendcommand_USB("DIRECTION 1");
```

```
usb.setDirectionAndMirror_USB(1,1);
```

```
usb.setShift_USB(50);
```

```

usb.setAfterPrintAction_USB(2);

usb.setOffset_USB(20);

usb.setCutMode_USB(0,2);


usb.clearbuffer_USB();

usb.sendcommand_USB("TEXT 100,100,\"3\",0,1,1,\"REVERSE\"");

usb.printReverse_USB(90,90,128,40);

usb.barcode_USB("30", "30", "128", "100", "1", "0", "2", "2", "barcode1234567");

usb.qrcode_USB("220", "260", "M", "3", "A", "0", "AABCB03abcN123");

usb.printerfont_USB("50", "10", "3", "0", "1", "1", "Print Font 123456");

string file = Environment.CurrentDirectory;

usb.downloadpcx_USB(file + "\\UL.PCX", "UL.PCX");

usb.windowfont_USB(10, 100, 48, 0, 0, 0, "arial", "C# WIN TEST");

//BitmapResult 为 Bitmap 处理结果，成功为 1；失败为 0

int BitmapResult = usb.Bitmap_USB("-500", "70", 400, 350, 1, "Thunder.png");

BitmapResult=usb.compressBitmap_USB("-600", "70", 666, 357, "Cat1.jpg");

usb.printlabel_USB("1", "1");

var status = usb.printerstatus_USB();

usb.genericDefault_USB();

usb.sensorDefault_USB();

usb.WifiFrequency_USB("5G");

//简中打印

string stString="默认简体中文测试";

usb.clearbuffer_USB();

usb.printerfont_USB ("100", "10", "TSS24.BF2", "0", "1", "1", stString);

usb.printlabel_USB (1, 1, this);

```

//繁中打印

```
string ttString="默認繁體中文測試";

usb.clearbuffer_USB();

usb.printerfont_USB("100", "10", " TST24.BF2", "0", "1", "1", ttString);

usb.printlabel_USB(1, 1, this);
```

//BLOCK 打印

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

usb.clearbuffer_USB();

usb.printerfontblock_USB("35","15","790","90","1","0","8","8","0","1", ttString);

usb.printlabel_USB(1, 1, this);
```

//RFID

```
usb.RFIDAutoCalibration_USB();

usb.rfidSetupDefault_USB();
```

//UHF GEN2

```
usb.writeUHF_USB("H", 2, 12, "E", "414142424343444445454646"); //startBlockNo: Gen2 默认为 2
string data = usb.readUHF_USB("H", 0, 12, "E");
string data_Q = usb.query_UHF_USB("A", 0, 0); //以 Q 指令读取 EPC 资料
usb.EPCPWD_Action_USB("L", "12345678");
usb.TIDPWD_Action_USB("L", "12345678");
usb.USERPWD_Action_USB("U", "12345678");
usb.AccessPWD_Action_USB("L", "12345678");
usb.KillPWD_Action_USB("L", "12345678");
usb.Set_RFIDPorcedure_USB(8,8,32,3,"N",2,2);
usb.Set_RFIDPorcedure_mm_USB(5, 40, 32, 5, "N", 5, 5, "203");
```

```
usb.writeHF_USB("H", 2, 12, "414142424343444445454646");
```

```
usb.printlabel_USB("1", "1");
```

```
//UHF GJB 写入资料
```

```
usb.writeGJB_UHF_USB ("H", 1, 12, "E", "414142424343444445454646", "12345678");
```

\*带入写入密码，写入资料(startBlockNo: GJB 默认为 1)

```
//UHF GJB 设定各密码区域新密码
```

```
usb.setPWD_Action_USB ("S", "S", "11112222", "12345678"); //带入写入密码，设定新状态密码
```

```
usb.setPWD_Action_USB ("R", "S", "33334444", "12345678"); //带入写入密码，设定新读取密码
```

```
usb.setPWD_Action_USB ("K", "S", "55556666", "12345678"); //带入写入密码，设定新删除密码
```

```
usb.setPWD_Action_USB ("W", "S", "87654321", "12345678"); //带入旧写入密码，设定新写入密码
```

```
//UHF GJB 设定资料区块状态
```

```
usb.statusGJB_UHF_USB ("E", "B", "11112222"); //带入状态密码，设定状态
```

```
usb.printlabel_USB ("1", "1");
```

```
//UHF GJB 读取资料
```

```
string data =usb.readGJB_UHF_USB("H", 0, 12, "E", "33334444"); //带入读取密码，读取标签资料
```

```
//UHF GJB 删除标签
```

```
usb.killGJB_UHF_USB ("55556666"); //带入删除密码，删除标签
```

```
usb.printlabel_USB ("1", "1");
```

```
usb.closePort_USB(1000);
```

```
usb.getDLLVersion_USB(1);
```

```
//指定网口传输介面
```

```
SocketConnect socketconnect = new SocketConnect();
```

```
socketconnect.openport_Ethernet("192.168.66.177", 9100);
```

```
socketconnect.setup_Ethernet ("54", "30", "2", "3", "0", "3", "0");
```

```
socketconnect.sendcommand_Ethernet ("DIRECTION 1");
```

```

socketconnect.setDirectionAndMirror_Ethernet(1,1);

socketconnect.setShift_Ethernet(50);

socketconnect.setAfterPrintAction_Ethernet(2);

socketconnect.setOffset_Ethernet(20);

socketconnect.setCutMode_Ethernet(0,2);

socketconnect.clearbuffer_Ethernet ();

socketconnect.sendcommand_Ethernet("TEXT 100,100,\"3\",0,1,1,\"REVERSE\"");

socketconnect.printReverse_Ethernet(90, 90, 128, 40);

socketconnect.barcode_Ethernet ("30", "30", "128", "100", "1", "0", "2", "2", "barcode1234567");

socketconnect.qrcode_Ethernet ("220", "260", "M", "3", "A", "0", "AABCB03abcN123");

socketconnect.printerfont_Ethernet ("50", "10", "3", "0", "1", "1", "Print Font 123456");

string file = Environment.CurrentDirectory;

socketconnect.downloadpcx_Ethernet (file + "\\UL.PCX", "UL.PCX");

socketconnect.windowfont_Ethernet (10, 100, 48, 0, 0, 0, "arial", "C# WIN TEST");

//BitmapResult 为 Bitmap 处理结果，成功为 1；失败为 0

int BitmapResult = socketconnect.Bitmap_Ethernet("-500", "70", 400, 350, 1, "Thunder.png");

BitmapResult = socketconnect.compressBitmap_Ethernet("-600", "70", 666, 357, "Cat1.jpg");

socketconnect.printlabel_Ethernet ("1", "1");

var status = socketconnect.printerstatus_Ethernet ();

socketconnect.genericDefault_Ethernet();

socketconnect.sensorDefault_Ethernet();

socketconnect.WifiFrequency_Ethernet("5G");

//简中打印

string stString="默认简体中文测试";

socketconnect.clearbuffer_Ethernet ();

socketconnect.printerfont_Ethernet ("100", "10", "TSS24.BF2", "0", "1", "1", stString);

```



```
socketconnect.printlabel_Ethernet (1, 1, this);
```

```
//繁中打印
```

```
string ttString="默認繁體中文測試";
```

```
socketconnect.clearbuffer_Ethernet ();
```

```
socketconnect.printerfont_Ethernet ("100", "10", " TST24.BF2", "0", "1", "1", ttString);
```

```
socketconnect.printlabel_Ethernet (1, 1, this);
```

```
//BLOCK 打印
```

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

```
socketconnect.clearbuffer_Ethernet ();
```

```
socketconnect.printerfontblock_Ethernet ("35","15","790","90","1","0","8","8","0","1", ttString);
```

```
socketconnect.printlabel_Ethernet (1, 1, this);
```

```
//RFID
```

```
socketconnect.RFIDAutoCalibration_Ethernet();
```

```
socketconnect.rfidSetupDefault_Ethernet();
```

```
//UHF GEN2
```

```
socketconnect.writeUHF_Ethernet("H", 2, 12,"E","414142424343444445454646"); //startBlockNo :
```

```
Gen2 默认为 2
```

```
socketconnect.printlabel_Ethernet ("1", "1");
```

```
string data = socketconnect.readUHF_Ethernet ("H", 0, 12, "E");
```

```
string data_Q = socketconnect.query_UHF_Ethernet ( "A", 0, 0); //以 Q 指令读取 EPC 资料
```

```
socketconnect.EPCPWD_Action_Ethernet ("L","12345678");
```

```
socketconnect.TIDPWD_Action_Ethernet ("L", "12345678");
```

```
socketconnect.USERPWD_Action_Ethernet ("L", "12345678");
```

```
socketconnect.AccessPWD_Action_Ethernet ("U", "12345678");
```

```

socketconnect.KillPWD_Action_Ethernet ("U", "12345678");

socketconnect.Set_RFIDPorcedure_Ethernet (8,8,32,3,"N",2,2);

socketconnect.Set_RFIDPorcedure_mm_Ethernet(5, 40, 32, 5, "N", 5, 5, "203");

socketconnect.writeHF_Ethernet ("H", 0, 12, "414142424343444445454646");

socketconnect.printlabel_Ethernet ("1", "1");

//UHF GJB 写入资料

socketconnect. writeGJB_UHF_Ethernet ("H", 1, 12, "E", "414142424343444445454646", "12345678");

*带入写入密码，写入资料(startBlockNo: GJB 默认为 1)

//UHF GJB 设定各密码区域新密码

socketconnect. setPWD_Action_Ethernet ("S", "S", "11112222", "12345678"); //带入写入密码，设定
新状态密码

socketconnect. setPWD_Action_Ethernet ("R", "S", "33334444", "12345678"); //带入写入密码，设定
新读取密码

socketconnect. setPWD_Action_Ethernet ("K", "S", "55556666", "12345678"); //带入写入密码，设定
新删除密码

socketconnect. setPWD_Action_Ethernet ("W", "S", "87654321", "12345678"); //带入旧写入密码，设定
新写入密码

//UHF GJB 设定资料区块状态

socketconnect. statusGJB_UHF_Ethernet ("E", "B", "11112222"); //带入状态密码，设定状态

socketconnect. printlabel_Ethernet ("1", "1");

//UHF GJB 读取资料

string data = socketconnect. readGJB_UHF_Ethernet ("H", 0, 12, "E", "33334444"); //带入读取密码，
读取标签资料

//UHF GJB 删除标签

socketconnect. killGJB_UHF_Ethernet ("55556666"); //带入删除密码，删除标签

```

```

socketconnect.printlabel_Ethernet ("1", "1");

socketconnect.closePort_Ethernet ();

socketconnect.getDLLVersion_Ethernet (1);

```

## ● Java

1.需先汇入 jan-5.5.0.jar:

```

import com.sun.jna.Library;

import com.sun.jna.Native;

```

2.建立调用 dll 的 class 调用 dll

```

class GTSPL {

    interface GTSPL_SDK extends Library {

        GTSPL_SDK INSTANCE = (GTSPL_SDK) Native.load("GTSPL_SDK_C", GTSPL_SDK.class);

        int openport_USB();

        int openport(String PrinterName);

        ....

    }
}

```

3.范例程序:

```

System.load(System.getProperty("user.dir") + "\\GTSPL_SDK_C.dll");

//单机打印

GTSPL.GTSPL_SDK.INSTANCE.openport("Printer Model");

GTSPL.GTSPL_SDK.INSTANCE.setup("54", "30", "2", "3", "0", "3", "0");

GTSPL.GTSPL_SDK.INSTANCE.sendcommand("DIRECTION 1");

GTSPL.GTSPL_SDK.INSTANCE.setDirectionAndMirror(1, 1);

GTSPL.GTSPL_SDK.INSTANCE.setShift(50);

GTSPL.GTSPL_SDK.INSTANCE.setAfterPrintAction(2);

GTSPL.GTSPL_SDK.INSTANCE.setOffset(20);

GTSPL.GTSPL_SDK.INSTANCE.setCutMode(0,2);

```

```

GTSP.LGTSP.L_SDK.INSTANCE.clearbuffer();

GTSP.LGTSP.L_SDK.INSTANCE.sendcommand("TEXT 100,100,\"3\",0,1,1,\"REVERSE\");

GTSP.LGTSP.L_SDK.INSTANCE.printReverse(90, 90, 128, 40);

GTSP.LGTSP.L_SDK.INSTANCE.barcode("30", "30", "128", "100", "1", "0", "2", "2",
"barcode1234567");

GTSP.LGTSP.L_SDK.INSTANCE.qrcode("50", "100", "H", "4", "A", "0", "QRcode1234567");

GTSP.LGTSP.L_SDK.INSTANCE.printerfont("50", "10", "2", "0", "1", "1", "Print Font 123456");

GTSP.LGTSP.L_SDK.INSTANCE.downloadbmp(System.getProperty("user.dir") + "\\CIRCLE.BMP",
"CIRCLE.BMP");

GTSP.LGTSP.L_SDK.INSTANCE.windowfont(5, 150, 48, 0, 0, 0, "arial", "JAVA WIN DRIVER");

GTSP.LGTSP.L_SDK.INSTANCE.sendcommand("PUTBMP 150,30,\"CIRCLE.BMP\");

GTSP.LGTSP.L_SDK.INSTANCE.printerfontblock("15", "15", "790", "90", "0", "0", "8", "8", "20",
"2", "We stand behind our products with one of the most comprehensive support programs in the
Auto-ID industry.");

```

//初始化

```

GTSP.LGTSP.L_SDK.INSTANCE.genericDefault();

GTSP.LGTSP.L_SDK.INSTANCE.sensorDefault();

GTSP.LGTSP.L_SDK.INSTANCE.rfidSetupDefault();

```

//修改 Wifi 频段

```

GTSP.LGTSP.L_SDK.INSTANCE.WifiFrequency("5G");

```

//BLOCK 打印

```

WString str = new WString("-- 默认简体中文测试：海天米醋白米醋 1 瓶 450ml --");

GTSP.LGTSP.L_SDK.INSTANCE.printerfontblockUnicode("15", "15", "790", "90", "TSS24.BF2", "0", "1",

```

```

"1", "0", "1", str);

GTSPL.GTSPL_SDK.INSTANCE.printlabel("1", "1");

GTSPL.GTSPL_SDK.INSTANCE.closeport();

GTSPL.GTSPL_SDK.INSTANCE.getDLLVersion(0);

//UHF GEN2

GTSPL.GTSPL_SDK.INSTANCE.writeUHF("H", 2, 12, "E", "11223344556677889900AABB");

GTSPL.GTSPL_SDK.INSTANCE.printlabel("1", "1");

GTSPL.GTSPL_SDK.INSTANCE.EPCPWD_Action("L", "12345678");

GTSPL.GTSPL_SDK.INSTANCE.TIDPWD_Action("L", "12345678");

GTSPL.GTSPL_SDK.INSTANCE.USERPWD_Action("L", "12345678");

GTSPL.GTSPL_SDK.INSTANCE.AccessPWD_Action("U", "12345678");

GTSPL.GTSPL_SDK.INSTANCE.KillPWD_Action("U", "12345678");

GTSPL.GTSPL_SDK.INSTANCE.Set_RFIDPorcedure(8,8,32,"3","N",2,2);

GTSPL.GTSPL_SDK.INSTANCE.Set_RFIDPorcedure_mm(5, 40, 32, 5, "N", 5, 5, "203");

GTSPL.GTSPL_SDK.INSTANCE.writeHF("H", 2, 12, "414142424343444445454646");

GTSPL.GTSPL_SDK.INSTANCE.printlabel("1", "1");

//UHF GJB 写入资料

GTSPL.GTSPL_SDK.INSTANCE.writeGJB_UHF("H", 1, 12, "E", "414142424343444445454646",
"12345678");

*带入写入密码，写入资料(startBlockNo: GJB 默认为 1)

//UHF GJB 设定各密码区域新密码

GTSPL.GTSPL_SDK.INSTANCE.setPWD_Action("S", "S", "11112222", "12345678");

//带入写入密码，设定新状态密码

GTSPL.GTSPL_SDK.INSTANCE.setPWD_Action("R", "S", "33334444", "12345678");

//带入写入密码，设定新读取密码

GTSPL.GTSPL_SDK.INSTANCE.setPWD_Action("K", "S", "55556666", "12345678");

```

//带入写入密码，设定新删除密码

```
GTSP.LGTSP.L_SDK.INSTANCE.setPWD_Action("W", "S", "87654321", "12345678");
```

//带入旧写入密码，设定新写入密码

//UHF GJB 设定资料区块状态

```
GTSP.LGTSP.L_SDK.INSTANCE.statusGJB_UHF("E", "B", "11112222"); //带入状态密码，设定状态
```

```
GTSP.LGTSP.L_SDK.INSTANCE.printlabel("1", "1");
```

//UHF GJB 删除标签

```
GTSP.LGTSP.L_SDK.INSTANCE.killGJB_UHF ("55556666"); //带入删除密码，删除标签
```

```
GTSP.LGTSP.L_SDK.INSTANCE.printlabel("1", "1");
```

```
GTSP.LGTSP.L_SDK.INSTANCE.closeport()
```

//简中打印

**WString str=new WString("默认简体中文测试"); //需 jni 的 WString 才能正确传送中文**

```
GTSP.LGTSP.L_SDK.INSTANCE.clearbuffer();
```

```
GTSP.LGTSP.L_SDK.INSTANCE.printerfontUnicode("100", "10", "TSS24.BF2", "0", "1", "1", str);
```

```
GTSP.LGTSP.L_SDK.INSTANCE.printlabel(1, 1);
```

//繁中打印

**WString str=new Wstring("默認繁體中文測試");//需 jni 的 WString 才能正确传送中文**

```
GTSP.LGTSP.L_SDK.INSTANCE.clearbuffer();
```

```
GTSP.LGTSP.L_SDK.INSTANCE. printerfontUnicode("100", "10", " TST24.BF2", "0", "1", "1", str);
```

```
GTSP.LGTSP.L_SDK.INSTANCE.printlabel(1, 1);
```

```
GTSP.LGTSP.L_SDK.INSTANCE.closeport();
```

```
GTSP.LGTSP.L_SDK.INSTANCE.getDLLVersion(0);
```

//RFID 自动校准

```
GTSP.LGTSP.L_SDK.INSTANCE.clearbuffer();
```

```
GTSP.LGTSP.L_SDK.INSTANCE.RFIDAutoCalibration();
```

```
GTSP.LGTSP.L_SDK.INSTANCE.closeport();
```

//指定 USB 传输接口

//侦测多台打印机

```
String PrinterName = GTSPL.GTSPL_SDK.INSTANCE.detectUSBStr_USB();
```

```
String[] PrinterNameStringArray = PrinterName.split("\\n+");
```

```
GTSPL.GTSPL_SDK.INSTANCE.openport_USB("Printer Model");
```

//单机打印

```
GTSPL.GTSPL_SDK.INSTANCE.openport_USB();
```

```
GTSPL.GTSPL_SDK.INSTANCE.setup_USB("54", "30", "2", "3", "0", "3", "0");
```

```
GTSPL.GTSPL_SDK.INSTANCE.sendcommand_USB("DIRECTION 1");
```

```
GTSPL.GTSPL_SDK.INSTANCE.setDirectionAndMirror_USB(1, 1);
```

```
GTSPL.GTSPL_SDK.INSTANCE.setShift_USB(50);
```

```
GTSPL.GTSPL_SDK.INSTANCE.setAfterPrintAction_USB(2);
```

```
GTSPL.GTSPL_SDK.INSTANCE.setOffset_USB(20);
```

```
GTSPL.GTSPL_SDK.INSTANCE.setCutMode_USB(0,2);
```

```
GTSPL.GTSPL_SDK.INSTANCE.clearbuffer_USB();
```

```
GTSPL.GTSPL_SDK.INSTANCE.barcode_USB("30", "30", "128", "100", "1", "0", "2", "2",  
"barcode1234567");
```

```
GTSPL.GTSPL_SDK.INSTANCE.qrcode_USB("50", "100", "H", "4", "A", "0", "QRcode1234567");
```

```
GTSPL.GTSPL_SDK.INSTANCE.printerfont_USB("50", "10", "2", "0", "1", "1", "Print Font 123456");
```

```
GTSPL.GTSPL_SDK.INSTANCE.windowfont_USB(5, 150, 48, 0, 0, 0, "arial", "JAVA WIN DRIVER");
```

```
GTSPL.GTSPL_SDK.INSTANCE.downloadpcx_USB(System.getProperty("user.dir")+ "\\UL.PCX",  
"UL.PCX");
```

```
GTSPL.GTSPL_SDK.INSTANCE.sendcommand_USB("PUTPCX 50,10,\"UL.PCX\");
```

```
GTSPL.GTSPL_SDK.INSTANCE.printerfontblock_USB("15", "15", "790", "90", "0", "0", "8", "8", "20",  
"2", "We stand behind our products with one of the most comprehensive support programs in the
```

```
Auto-ID industry.");
```

```
//初始化
```

```
GTSP.LGTSP.L_SDK.INSTANCE.genericDefault_USB();
```

```
GTSP.LGTSP.L_SDK.INSTANCE.sensorDefault_USB();
```

```
GTSP.LGTSP.L_SDK.INSTANCE.rfidSetupDefault_USB();
```

```
//修改 Wifi 频段
```

```
GTSP.LGTSP.L_SDK.INSTANCE.WifiFrequency_USB ("5G");
```

```
//BLOCK 打印
```

```
WString str = new WString("-- 默认简体中文测试：海天米醋白米醋 1 瓶 450ml --");
```

```
GTSP.LGTSP.L_SDK.INSTANCE.printerfontblockUnicode_USB("15", "15", "790", "90", "TSS24.BF2", "0",  
"1", "1", "0", "1", str);
```

```
GTSP.LGTSP.L_SDK.INSTANCE.printlabel_USB("1", "1");
```

```
//UHF GEN2
```

```
String status = GTSP.LGTSP.L_SDK.INSTANCE.printerstatus_USB();
```

```
GTSP.LGTSP.L_SDK.INSTANCE.writeUHF_USB("H", 2, 12, "E", "11223344556677889900AABB");
```

```
GTSP.LGTSP.L_SDK.INSTANCE.printlabel_USB("1", "1");
```

```
String status = GTSP.LGTSP.L_SDK.INSTANCE.readUHF_USB("A", 0, 12, "E");
```

```
GTSP.LGTSP.L_SDK.INSTANCE.EPCPWD_Action_USB("L", "12345678");
```

```
GTSP.LGTSP.L_SDK.INSTANCE.TIDPWD_Action_USB("L", "12345678");
```

```
GTSP.LGTSP.L_SDK.INSTANCE.USERPWD_Action_USB("L", "12345678");
```

```
GTSP.LGTSP.L_SDK.INSTANCE.AccessPWD_Action_USB("U", "12345678");
```

```
GTSP.LGTSP.L_SDK.INSTANCE.KillPWD_Action_USB("U", "12345678");
```



```

GTSP.LGTSP.L_SDK.INSTANCE.Set_RFIDPorcedure_USB(8,8,32,"3","N",2,2);

GTSP.LGTSP.L_SDK.INSTANCE.Set_RFIDPorcedure_mm_USB(5, 40, 32, 5, "N", 5, 5, "203");

GTSP.LGTSP.L_SDK.INSTANCE.writeHF_USB("H", 2, 12, "414142424343444445454646")

GTSP.LGTSP.L_SDK.INSTANCE.printlabel_USB("1", "1");

//UHF GJB 写入资料

GTSP.LGTSP.L_SDK.INSTANCE.writeGJB_UHF_USB("H", 1, 12, "E", "414142424343444445454646",
"12345678");

*带入写入密码，写入资料(startBlockNo: GJB 默认为 1)

//UHF GJB 设定各密码区域新密码

GTSP.LGTSP.L_SDK.INSTANCE.setPWD_Action_USB ("S", "S", "11112222", "12345678");

//带入写入密码，设定新状态密码

GTSP.LGTSP.L_SDK.INSTANCE.setPWD_Action("R", "S", "33334444", "12345678");

//带入写入密码，设定新读取密码

GTSP.LGTSP.L_SDK.INSTANCE.setPWD_Action("K", "S", "55556666", "12345678");

//带入写入密码，设定新删除密码

GTSP.LGTSP.L_SDK.INSTANCE.setPWD_Action("W", "S", "87654321", "12345678");

//带入旧写入密码，设定新写入密码

//UHF GJB 设定资料区块状态

GTSP.LGTSP.L_SDK.INSTANCE.statusGJB_UHF_USB ("E", "B", "11112222");

//带入状态密码，设定状态

GTSP.LGTSP.L_SDK.INSTANCE.printlabel_USB( ("1", "1");

//UHF GJB 读取资料

String data = GTSP.LGTSP.L_SDK.INSTANCE. readGJB_UHF_USB("H", 0, 12, "E", "33334444");

//带入读取密码，读取标签资料

//UHF GJB 删除标签

```

```
GTSP.LGTSP.L_SDK.INSTANCE.killGJB_UHF ("55556666"); //带入删除密码，删除标签
```

```
GTSP.LGTSP.L_SDK.INSTANCE.printlabel_USB( "1", "1");
```

```
GTSP.LGTSP.L_SDK.INSTANCE.closePort_USB(1000);
```

```
GTSP.LGTSP.L_SDK.INSTANCE.getDLLVersion_USB(1);
```

```
//简中打印
```

```
WString str=new WString("默认简体中文测试");//需 jni 的 WString 才能正确传送中文
```

```
GTSP.LGTSP.L_SDK.INSTANCE.clearbuffer_USB();
```

```
GTSP.LGTSP.L_SDK.INSTANCE. printerfontUnicode_USB ("100", "10", "TSS24.BF2", "0", "1", "1", str);
```

```
GTSP.LGTSP.L_SDK.INSTANCE.printlabel_USB (1, 1);
```

```
//繁中打印
```

```
WString str=new Wstring("默认繁体中文测试");//需 jni 的 WString 才能正确传送中文
```

```
GTSP.LGTSP.L_SDK.INSTANCE.clearbuffer_USB();
```

```
GTSP.LGTSP.L_SDK.INSTANCE. printerfontUnicode_USB ("100", "10", " TST24.BF2", "0", "1", "1", str);
```

```
GTSP.LGTSP.L_SDK.INSTANCE.printlabel_USB(1, 1);
```

```
GTSP.LGTSP.L_SDK.INSTANCE.closeport_USB();
```

```
GTSP.LGTSP.L_SDK.INSTANCE.getDLLVersion_USB(0);
```

```
//RFID 自动校准
```

```
GTSP.LGTSP.L_SDK.INSTANCE.clearbufferr_USB();
```

```
GTSP.LGTSP.L_SDK.INSTANCE.RFIDAutoCalibrationr_USB();
```

```
GTSP.LGTSP.L_SDK.INSTANCE.closeportr_USB();
```

```
//指定 Ethernet 传输接口
```

```
GTSP.LGTSP.L_SDK.INSTANCE.openport_Ethernet(IP,Port);
```

```
GTSP.LGTSP.L_SDK.INSTANCE.setup_Ethernet("54", "30", "2", "3", "0", "3", "0");
```

```
GTSP.LGTSP.L_SDK.INSTANCE.sendcommand_Ethernet("DIRECTION 1");
```

```
GTSP.LGTSP.L_SDK.INSTANCE.setOffset_Ethernet(20);
```

```

GTSP.LGTSP.L_SDK.INSTANCE.setCutMode_ Ethernet(0,2);

GTSP.LGTSP.L_SDK.INSTANCE.setDirectionAndMirror_ Ethernet(1, 1);

GTSP.LGTSP.L_SDK.INSTANCE.setShift_ Ethernet(50);

GTSP.LGTSP.L_SDK.INSTANCE.setAfterPrintAction_ Ethernet(2);

GTSP.LGTSP.L_SDK.INSTANCE.clearbuffer_ Ethernet();

GTSP.LGTSP.L_SDK.INSTANCE.sendcommand_ Ethernet("TEXT 100,100,\"3\",0,1,1,\"REVERSE\"");

GTSP.LGTSP.L_SDK.INSTANCE.printReverse_ Ethernet(90, 90, 128, 40);

GTSP.LGTSP.L_SDK.INSTANCE.barcode_ Ethernet("30", "30", "128", "100", "1", "0", "2", "2",
"barcode1234567");

GTSP.LGTSP.L_SDK.INSTANCE.qrcode_ Ethernet("50", "100", "H", "4", "A", "0", "QRcode1234567");

GTSP.LGTSP.L_SDK.INSTANCE.printerfont_ Ethernet("50", "10", "2", "0", "1", "1", "Print Font
123456");

GTSP.LGTSP.L_SDK.INSTANCE.windowfont_ Ethernet(5, 150, 48, 0, 0, 0, "arial", "JAVA WIN DRIVER");

GTSP.LGTSP.L_SDK.INSTANCE.downloadpcx_ Ethernet(System.getProperty("user.dir")+ "\\UL.PCX",
"UL.PCX");

    GTSP.LGTSP.L_SDK.INSTANCE.sendcommand_ Ethernet("PUTPCX 50,10,\"UL.PCX\"");

GTSP.LGTSP.L_SDK.INSTANCE.printerfontblock_ Ethernet("15", "15", "790", "90", "0", "0", "8", "8",
"20", "2", "We stand behind our products with one of the most comprehensive support programs in
the Auto-ID industry.");

```

//初始化

```

GTSP.LGTSP.L_SDK.INSTANCE.genericDefault_ Ethernet();

GTSP.LGTSP.L_SDK.INSTANCE.sensorDefault_ Ethernet();

GTSP.LGTSP.L_SDK.INSTANCE.rfidSetupDefault_ Ethernet();

```

//修改 Wifi 频段

```
GTSP.LGTSP.L_SDK.INSTANCE.WifiFrequency("5G");
```

```
//BLOCK 打印
```

```
WString str = new WString("-- 默认简体中文测试：海天米醋白米醋 1 瓶 450ml --");
GTSP.LGTSP.L_SDK.INSTANCE.printerfontblockUnicode_Ethernet("15", "15", "790", "90", "TSS24.BF2",
"0", "1", "1", "0", "1", str);
GTSP.LGTSP.L_SDK.INSTANCE.printlabel_Ethernet("1", "1");
String status = GTSP.LGTSP.L_SDK.INSTANCE.printerstatus_Ethernet();
```

```
//UHF GEN2
```

```
GTSP.LGTSP.L_SDK.INSTANCE.writeUHF_Ethernet("H", 0, 12, "E", "11223344556677889900AABB");
GTSP.LGTSP.L_SDK.INSTANCE.printlabel_Ethernet("1", "1");
String status = GTSP.LGTSP.L_SDK.INSTANCE.readUHF_Ethernet("A", 0, 12, "E");
GTSP.LGTSP.L_SDK.INSTANCE.EPCPWD_Action_Ethernet("L", "12345678");
GTSP.LGTSP.L_SDK.INSTANCE.TIDPWD_Action_Ethernet("L", "12345678");
GTSP.LGTSP.L_SDK.INSTANCE.USERPWD_Action_Ethernet("L", "12345678");
GTSP.LGTSP.L_SDK.INSTANCE.AccessPWD_Action_Ethernet("U", "12345678");
GTSP.LGTSP.L_SDK.INSTANCE.KillPWD_Action_Ethernet("U", "12345678");
GTSP.LGTSP.L_SDK.INSTANCE.Set_RFIDPorcedure_Ethernet(8, 8, 32, "3", "N", 2, 2);
GTSP.LGTSP.L_SDK.INSTANCE.Set_RFIDPorcedure_mm_Ethernet(5, 40, 32, 5, "N", 5, 5, "203");
GTSP.LGTSP.L_SDK.INSTANCE.writeHF_Ethernet("H", 0, 12, "4141424243434444445454646")
GTSP.LGTSP.L_SDK.INSTANCE.printlabel_Ethernet("1", "1");
```

```
//UHF GJB 写入资料
```

```
GTSP.LGTSP.L_SDK.INSTANCE.writeGJB_UHF_Ethernet("H", 1, 12, "E",
"4141424243434444445454646", "12345678");
```

\*帶入寫入密碼，寫入資料(startBlockNo: GJB 預設為 1)

// UHF GJB 設定各密碼區域新密碼

```
GTSP.LGTSP.LSDK.INSTANCE.setPWD_Action_Ethernet("S", "S", "11112222", "12345678");
```

//帶入寫入密碼，設定新狀態密碼

```
GTSP.LGTSP.LSDK.INSTANCE.setPWD_Action_Ethernet("R", "S", "33334444", "12345678");
```

//帶入寫入密碼，設定新讀取密碼

```
GTSP.LGTSP.LSDK.INSTANCE.setPWD_Action_Ethernet("K", "S", "55556666", "12345678");
```

//帶入寫入密碼，設定新刪除密碼

```
GTSP.LGTSP.LSDK.INSTANCE.setPWD_Action_Ethernet("W", "S", "87654321", "12345678");
```

//帶入舊寫入密碼，設定新寫入密碼

// UHF GJB 設定資料區塊狀態

```
GTSP.LGTSP.LSDK.INSTANCE.statusGJB_UHF_Ethernet("E", "B", "11112222");
```

//帶入狀態密碼，設定狀態

```
GTSP.LGTSP.LSDK.INSTANCE.printlabel_Ethernet( ("1", "1");
```

// UHF GJB 讀取資料

```
String data = GTSP.LGTSP.LSDK.INSTANCE. readGJB_UHF_Ethernet("H", 0, 12, "E", "33334444");
```

//帶入讀取密碼，讀取標籤資料

// UHF GJB 刪除標籤

```
GTSP.LGTSP.LSDK.INSTANCE.killGJB_UHF_Ethernet("55556666"); //帶入刪除密碼，刪除標籤
```

```
GTSP.LGTSP.LSDK.INSTANCE.printlabel_Ethernet( ("1", "1");
```

```
GTSP.LGTSP.LSDK.INSTANCE.closePort_Ethernet();
```

```
GTSP.LGTSP.LSDK.INSTANCE.getDLLVersion_Ethernet(1);
```

//簡中打印

WString str=new WString("默认简体中文测试");//需 jni 的 WString 才能正確傳送中文

```
GTSP.LGTSP.LSDK.INSTANCE.clearbuffer_Ethernet();
```

```

GTSP.LGTSP.L_SDK.INSTANCE.printerfontUnicode_Ethernet("100", "10", "TSS24.BF2", "0", "1", "1",
str);

GTSP.LGTSP.L_SDK.INSTANCE.printlabel_Ethernet(1, 1);

//繁中打印
WString str=new Wstring("默認繁體中文測試");//需 jni 的 WString 才能正確傳送中文
GTSP.LGTSP.L_SDK.INSTANCE.clearbuffer_Ethernet();

GTSP.LGTSP.L_SDK.INSTANCE. printerfontUnicode_Ethernet("100", "10", " TST24.BF2", "0", "1", "1",
str);

GTSP.LGTSP.L_SDK.INSTANCE.printlabel_Ethernet(1, 1);

GTSP.LGTSP.L_SDK.INSTANCE.closeport_Ethernet();

GTSP.LGTSP.L_SDK.INSTANCE.getDLLVersion_Ethernet(0);

//RFID 自动校准
GTSP.LGTSP.L_SDK.INSTANCE.clearbufferr_Ethernet();

GTSP.LGTSP.L_SDK.INSTANCE.RFIDAutoCalibrationr_Ethernet();

GTSP.LGTSP.L_SDK.INSTANCE.closeport_Ethernet();

```

#### 4.DLL 放置位置:

〔 利用 IDE 开发时 〕

GTSP.L\_SDK.dll 放在 jdk 的 bin 文件夹下

如: C:\Program Files\Java\jdk1.8.0\_221\bin

GTSP.L\_SDK\_C.dll 放在 java 项目文件夹下

〔 未安装 jdk 直接执行.jar 档时 〕

GTSP.L\_SDK.dll 放在 jre 的 bin 文件夹下

如: C:\Program Files\Java\jre1.8.0\_251\bin

GTSP.L\_SDK\_C.dll 放在.jar 档相同路径下

.jar 档相同路径下，还需要有 lib 文件夹，且里面要有 jna-5.5.0.jar

5.Driver 模式下，需要安装打印机的驱动才能执行。

## ● Javascript

1.使用管理员身分开启命令提示字符

2.注册 dll

将 x86 的 dll 放到 Windows/Syswow64，x64 的 dll 放在 Windows/System32

进入 microsoft.net framework 的安装路径下

如：cd C:\Windows\Microsoft.NET\Framework64\v4.0.30319

\*x64 的 dll 注册在 C:\Windows\Microsoft.NET\Framework64\v4.0.30319

\*x86 的 dll 注册在 C:\Windows\Microsoft.NET\Framework\v4.0.30319

输入指令 regasm.exe /register /tlb <dll path>

输入指令 regasm.exe <dll path> /codebase

如：regasm.exe /register /tlb C:\Users\User\Desktop\Javascript\_Sample\_Code\GTSPK\_SDK.dll

regasm.exe C:\Users\User\source\repos\product\GTSPK\_SDK\GTSPK\_SDK.dll /codebase

3.范例程序：

```
<script language="javascript" type="text/javascript">
```

```
//单机打印
```

```
var driverObject = new ActiveXObject("GTSPK_SDK.Driver");
```

```
driverObject.openport("Printer Model");
```

```
driverObject.setup("54", "30", "2", "3", "0", "3", "0");
```

```
driverObject.sendcommand("DIRECTION 1");
```

```
driverObject.clearbuffer();
```

```
driverObject.barcode("30", "30", "128", "100", "1", "0", "2", "2", "barcode1234567");
```

```
driverObject.printerfont("50", "10", "3", "0", "1", "1", "Print Font 123456");
```

```
var file = getFilePath();

driverObject.downloadpcx(file + "\\UL.PCX", "UL.PCX");

driverObject.sendcommand("PUTPCX 50,10,\"UL.PCX\"");

driverObject.windowfont(50, 20, 48, 0, 0, 0, "arial", "Windows Font Test123467");

driverObject.getDLLVersion(1)

driverObject.printlabel("1", "1");

driverObject.closeport();


//指定 USB 传输接口

var usbObject = new ActiveXObject("GTSPL_SDK.USB");

usbObject.openport_USB();

usbObject.setup_USB("54", "30", "2", "3", "0", "3", "0");

usbObject.sendcommand_USB("DIRECTION 1");

usbObject.clearbuffer_USB();

usbObject.barcode_USB("30", "30", "128", "100", "1", "0", "2", "2", "barcode1234567");

usbObject.printerfont_USB("50", "10", "3", "0", "1", "1", "Print Font 123456");

var file = getFilePath();

usbObject.downloadbmp_USB(file + "\\CIRCLE.BMP", "CIRCLE.BMP");

usbObject.sendcommand_USB("PUTBMP 150,30,\"CIRCLE.BMP\"");

usbObject.windowfont_USB(10, 100, 48, 0, 0, 0, "arial", "Windows Font Test");

var status = usbObject.printerstatus_USB();

usbObject.printlabel_USB("1", "1");


//简中打印

string stString="默认简体中文测试";

usbObject.clearbuffer_USB();
```



```

usbObject.printerfont_USB("100", "10", "TSS24.BF2", "0", "1", "1", stString);

usbObject.printlabel_USB(1, 1,);

//繁中打印
string ttString="默認繁體中文測試";

usbObject.clearbuffer_USB();

usbObject.printerfont_USB ("100", "10", " TST24.BF2", "0", "1", "1", ttString);

usbObject.printlabel_USB (1, 1);

usbObject.getDLLVersion_USB(1);

usbObject.closeport_USB();

usbObject.getDLLVersion_USB(0)

</script>

```

## ● Asp.Net

- 1.加入 x86 的 dll 进项目
- 2.范例程序:

```

//单机打印

GTSPL_SDK.Driver driver = new GTSPL_SDK.Driver();

driver.openport("Printer Model");

driver.setup("54", "30", "2", "3", "0", "3", "0");

driver.sendcommand("DIRECTION 1");

driver.clearbuffer();

driver.barcode("30", "30", "128", "100", "1", "0", "2", "2", "barcode1234567");

string path = HttpContext.Current.Server.MapPath("~/");

driver.downloadpcx(path + "\\CIRCLE.BMP", "CIRCLE.BMP");

driver.sendcommand("PUTBMP 150,30,\"CIRCLE.BMP\"");

```

```

driver.windowfont(50, 20, 48, 0, 0, 0, "impact", "Windows Font ASP.NET");

driver.getDLLVersion(1);

driver.printlabel("1", "1");

driver.closeport();

```

//指定 USB 传输接口

```

GTSPL_SDK.USB usb = new GTSPL_SDK.USB();

usb.openport_USB();

usb.setup_USB("54", "30", "2", "3", "0", "3", "0");

usb.sendcommand_USB("DIRECTION 1");

usb.clearbuffer_USB();

usb.barcode_USB("30", "30", "128", "100", "1", "0", "2", "2", "barcode1234567usb");

usb.printerfont_USB("50", "10", "3", "0", "1", "1", "Print Font usb Asp");

usb.downloadpcx_USB(path + "\\UL.PCX", "UL.PCX");

usb.sendcommand_USB("PUTPCX 50,10,\"UL.PCX\"");

usb.windowfont_USB(10, 100, 48, 0, 0, 0, "impact", "Windows Font ASP USB");

usb.printlabel_USB("1", "1");

var status = usb.printerstatus_USB();

usb.getDLLVersion_USB(1);

```

//简中打印

```

string stString="默认简体中文测试";

usb.clearbuffer_USB();

usb.printerfont_USB("100", "10", "TSS24.BF2", "0", "1", "1", stString);

usb.printlabel_USB(1, 1);

```

//繁中打印

```
string ttString="默認繁體中文測試";

usb.clearbuffer_USB();

usb.printerfont_USB ("100", "10", " TST24.BF2", "0", "1", "1", ttString);

usb.printlabel_USB (1, 1);

usb.closeport_USB();
```

## ● VB.Net

### 1.加入 dll 进项目

### 2.范例程序:

```
Dim path = Environment.CurrentDirectory

//单机打印

Dim driver As New GTSPL_SDK.Driver

driver.openport("Printer Model")

driver.setup("54", "30", "2", "3", "0", "3", "0")

driver.sendcommand("DIRECTION 1")

driver.clearbuffer()

driver.barcode("30", "30", "128", "100", "1", "0", "2", "2", "barcode1234567")

driver.printerfont("50", "10", "3", "0", "1", "1", "PrintFont in vb.net")

driver.downloadbmp(path + "\\CIRCLE.BMP", "CIRCLE.BMP")

driver.sendcommand("PUTBMP 50,10, ""CIRCLE.BMP""")

driver.windowfont(50, 20, 48, 0, 0, 0, "impact", "VB Imapct Driver")

driver.printlabel("1", "1")

driver.closeport()
```

//指定 USB 传输接口

```
Dim usb As New GTSPL_SDK.USB
```

```
usb.openport_USB()
```

```
usb.setup_USB("54", "30", "2", "3", "0", "3", "0")
```

```
usb.sendcommand_USB("DIRECTION 1")
```

```
usb.clearbuffer_USB()
```

```
usb.barcode_USB("30", "30", "128", "100", "1", "0", "2", "2", "barcode1234567")
```

```
usb.downloadbmp_USB(path + "\\CIRCLE.BMP", "CIRCLE.BMP")
```

```
usb.sendcommand_USB("PUTBMP 50,10,\"\"CIRCLE.BMP\"\"")
```

```
usb.windowfont_USB(50, 20, 48, 0, 0, 0, "impact", "VB Imapct Driver")
```

```
Dim status = usb.printerstatus_USB()
```

```
Dim vcode = usb.getDLLVersion_USB("1")
```

```
usb.printlabel_USB("1", "1")
```

```
usb.closeport_USB()
```

//简中打印

```
usb.clearbuffer_USB();
```

```
usb.printerfont_USB("100", "10", "TSS24.BF2", "0", "1", "1", "默认简体中文测试");
```

```
usb.printlabel_USB(1, 1);
```

//繁中打印

```
usb.clearbuffer_USB();
```

```
usb.printerfont_USB ("100", "10", " TST24.BF2", "0", "1", "1", "默认繁体中文测试");
```

```
usb.printlabel_USB (1, 1);
```

```
usb.closeport_USB();
```

## ● Access VBA

1.使用管理员身分开启命令提示字符

2.注册 dll

将 x86 的 dll 放到 Windows/Syswow64

进入 microsoft.net framework 的安装路径下

如: cd C:\Windows\Microsoft.NET\Framework\v4.0.30319

输入指令 regasm.exe /register /tlb <dll path>

输入指令 regasm.exe <dll path> /codebase

如: regasm.exe /register /tlb C:\Users\User\Desktop\Javascript\_Sample\_Code\GTSP\_SDK.dll

regasm.exe C:\Users\User\source\repos\product\GTSP\_SDK\GTSP\_SDK.dll /codebase

3.范例程序:

```
Dim usb As New GTSP_SDK.usb
```

```
Dim path As String
```

```
usb.openport_USB()
```

```
Call usb.clearbuffer_USB
```

```
Call usb.setup_USB("54", "39", "4", "7", "0", "3", "0")
```

```
Call usb.clearbuffer_USB
```

```
Call usb.printerfont_USB("10", "20", "2", "0", "1", "1", "Product Management")
```

```
Call usb.printerfont_USB("10", "40", "2", "0", "1", "1", "Product No. : ")
```

```
Call usb.barcode_USB("10", "60", "128", "50", "1", "0", "1", "1", ProductNo)
```

```
Call usb.printerfont_USB("10", "140", "2", "0", "1", "1", "Product Name : " + ProductName)
```

```
Call usb.printerfont_USB("10", "160", "2", "0", "1", "1", "Unit Price : " + UnitPrice)
```

```
Call usb.printerfont_USB("10", "180", "2", "0", "1", "1", "Amount : " + Amount)
```

```
Call usb.printerfont_USB("10", "200", "2", "0", "1", "1", "Total Price : " + TotalPrice)
```

```
Call usb.sendcommand_USB("PUTBMP 10,230,\"\"LOGO.BMP\"\"")
```

```
Call usb.printlabel_USB("1", "1")
```

Call `usb.closeport_USB`

## ● Excel VBA

1.使用管理员身分开启命令提示字符

2.注册 dll

将 x86 的 dll 放到 Windows/Syswow64

进入 microsoft.net framework 的安装路径下

如: cd C:\Windows\Microsoft.NET\Framework\v4.0.30319

输入指令 regasm.exe /register /tlb <dll path>

输入指令 regasm.exe <dll path> /codebase

如: regasm.exe /register /tlb C:\Users\User\Desktop\Javascript\_Sample\_Code\GTSP\_L\_SDK.dll

regasm.exe C:\Users\User\source\repos\product\GTSP\_L\_SDK \GTSP\_L\_SDK.dll /codebase

3.范例程序:

```
Dim driver As New GTSP_L_SDK.driver
```

```
Dim path As String
```

```
openport = driver.openport("Printer Model")
```

```
Call driver.setup("54", "20", "2", "3", "0", "3", "0")
```

```
Call driver.sendcommand("DIRECTION 1")
```

```
Call driver.clearbuffer
```

```
Call driver.barcode("30", "30", "128", "100", "1", "0", "2", "2", "barcode1234567")
```

```
Call driver.printerfont("50", "10", "3", "0", "1", "1", "PrintFont in vba_xls")
```

```
Call driver.downloadpcx(path + "\\UL.PCX", "UL.PCX")
```

```
Call driver.sendcommand("PUTPCX 50,10,\"\"UL.PCX\"\"")
```

```
Call driver.windowfont(50, 20, 48, 0, 0, 0, "impact", "VBAEX Driver")
```

```
Call driver.printlabel("1", "1")
```

```
Call driver.closeport
```

## ● Visual C++

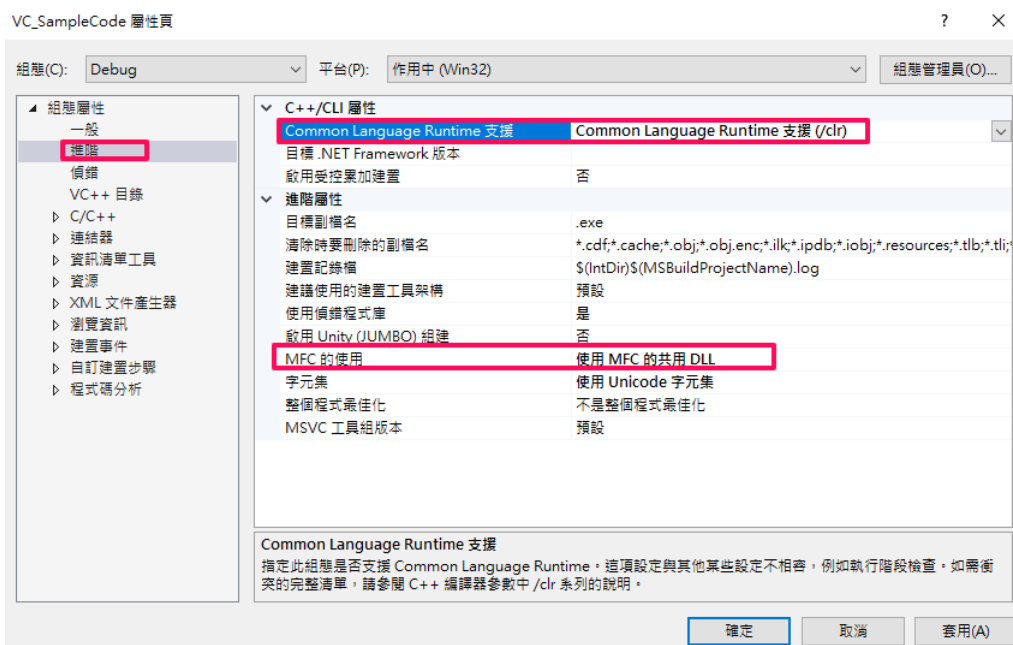
### 1. 设定项目属性

要支持 dll 必须将 Common Language Runtime 支持启动

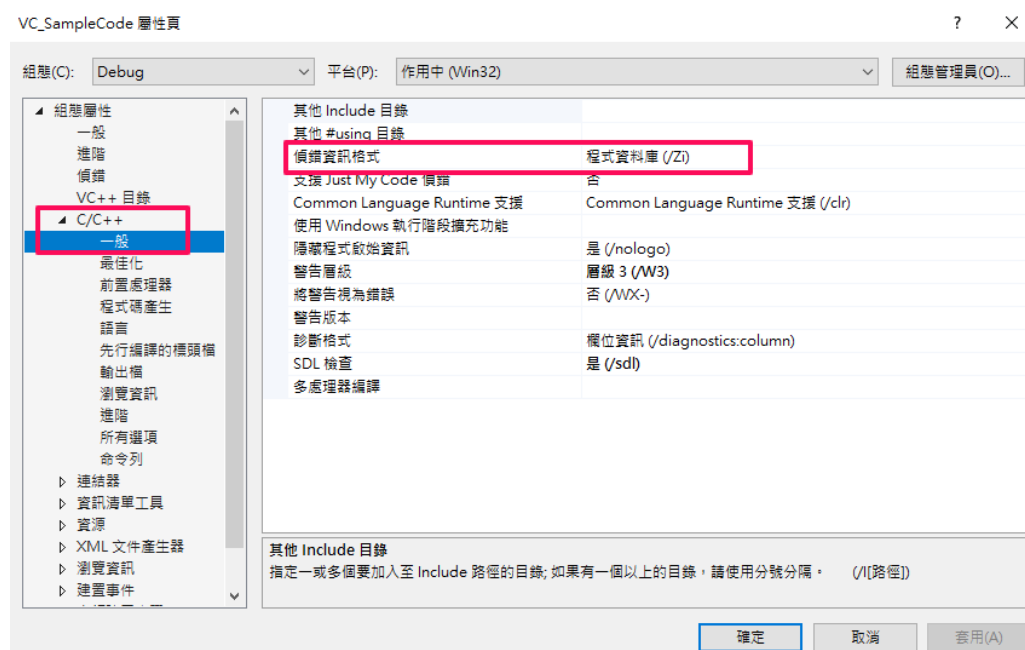
=>开启项目属性

=>组态属性=>进阶=>Common Language Runtime 支持设定为(/clr)

=>组态属性=>进阶=>MFC 的使用设定为使用 MFC 的共享 DLL



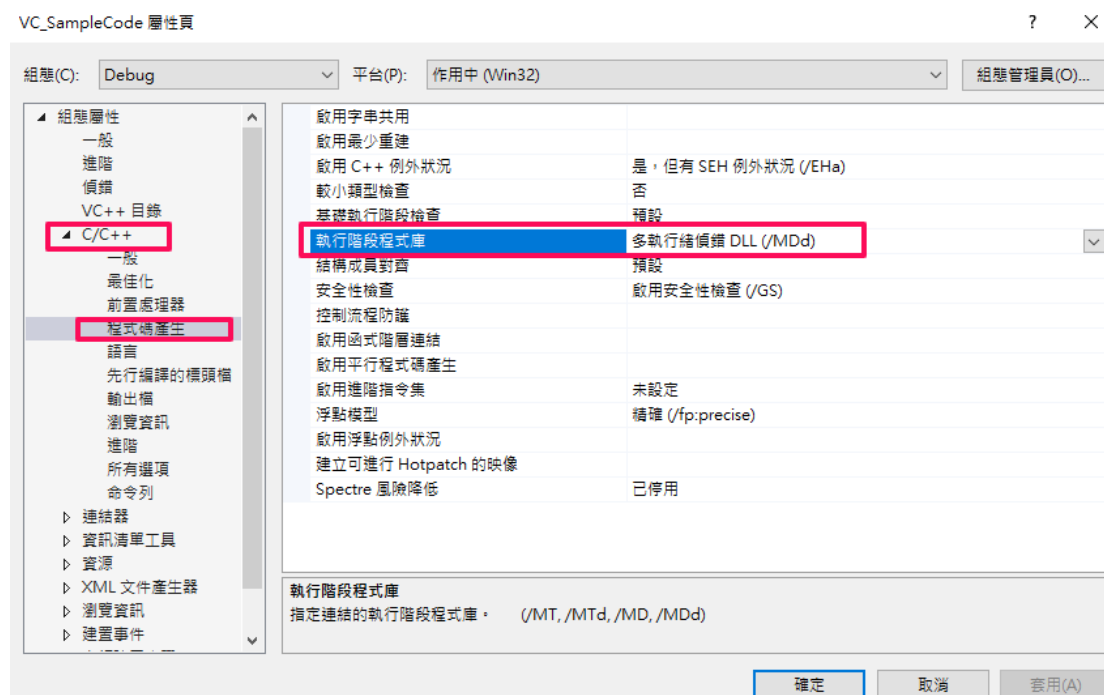
属性=>C/C++=>一般=>侦错信息格式设定为"程序数据库/Zi"(不可以设定为/zi)





属性=>C/C++=>程序代码产生

=>运行时间链接库设定为"多执行续侦错 DLL(/MDd)"或"多执行续(/MD)"



### 3.范例程序:

```
#using "GTSPL_SDK.dll"

using namespace System;

using namespace GTSPL_SDK;

//单机打印

Driver^ driver = gcnew Driver();

String^ printerName = gcnew String(str);

driver->openport(printerName);

driver->setup("30", "30", "2", "3", "0", "3", "0");

driver->clearbuffer();

driver->barcode("30", "30", "128", "100", "1", "0", "2", "2", "barcode VC5678 ");

driver->printerfont("10", "150", "2", "0", "1", "1", "Print Font VC DRIVER");

driver->downloadpcx(_T("../VC_SampleCode/UL.PCX"), "UL.PCX");

driver->sendcommand("PUTPCX 50,10,\"UL.PCX\"");
```

```

driver ->downloadbmp (_T("../VC_SampleCode/CIRCLE.BMP"), "CIRCLE.BMP");

driver ->sendcommand ("PUTBMP 150,30,\"CIRCLE.BMP\"");

driver->windowsfont(10, 120, 48, 0, 0, 0, "impact", "VC Driver test");

driver->printlabel("1", "1");

driver->sendcommand("DIRECTION 1");

driver->qrcode("220", "260", "M", "3", "A", "0", "AABCB03abcN123");

driver->printerfontblock ("35","15","790","90","1","0","8","8","0","1", ttString);

driver ->setDirectionAndMirror(1,1);

driver ->setShift(50);

driver ->setAfterPrintAction(2);

driver ->setOffset(20);

driver ->setCutMode(0,2);

driver ->printReverse (90,90,128,40);

driver->printlabel("1", "1");

driver->sendcommand("DIRECTION 1");

driver->genericDefault();

driver->sensorDefault();

// RFID

driver-> rfidSetupDefault();

driver->RFIDAutoCalibration();

//UHF GEN2

driver->writeUHF("H", 2, 12, "E", "414142424343444445454646"); // startBlockNo: Gen2 預設
為 2

driver->printlabel("1", "1");

driver->EPCPWD_Action("U", "12345678");

driver->TIDPWD_Action("L", "12345678");

```

```

driver->USERPWD_Action("L", "12345678");

driver->AccessPWD_Action("S", "12345678");

driver->KillPWD_Action("L", "12345678");

driver->Set_RFIDPorcedure(8, 8, 32, 3, "N", 2, 2);

driver->Set_RFIDPorcedure_mm(5, 40, 32, 5, "N", 5, 5, "203");

driver->writeHF("H", 0, 12, "414142424343444445454646");

driver->printlabel("1", "1");

//UHF GJB 写入资料

driver->writeGJB_UHF("H", 1, 12, "E", "414142424343444445454646", "12345678");

*帶入寫入密碼，寫入資料(startBlockNo: GJB 預設為 1)

///UHF GJB 设定各密码区域新密码

driver->setPWD_Action("S", "S", "11112222", "12345678"); //帶入写入密码，设定新状态密码

driver->setPWD_Action("R", "S", "33334444", "12345678"); //帶入写入密码，设定新读取密码

driver->setPWD_Action("K", "S", "55556666", "12345678"); //帶入写入密码，设定新删除密码

//帶入旧写入密码，设定新写入密码

driver->setPWD_Action("W", "S", "87654321", "12345678");

//UHF GJB 设定资料区块状态

driver->statusGJB_UHF("E", "B", "11112222"); //帶入状态密码，设定状态

driver->printlabel("1", "1");

//UHF GJB 删除标签

driver->killGJB_UHF ("55556666"); //帶入删除密码，删除标签

driver->printlabel("1", "1");

driver->closeport();

//指定 USB 传输接口

USB^ usb = gcnew USB();

```

```

usb->openport_USB();

usb->setup_USB("54", "30", "2", "3", "0", "3", "0");

usb->sendcommand_USB("DIRECTION 1");

usb->clearbuffer_USB();

usb->barcode_USB("10", "80", "128", "100", "1", "0", "2", "2", "barcode vc");

usb->printerfont_USB("50", "100", "2", "0", "1", "1", "Print Font VC123");

usb->downloadbmp_USB(_T("../VC_SampleCode/CIRCLE.BMP"), "CIRCLE.BMP");

usb->sendcommand_USB("PUTBMP 150,30,\"CIRCLE.BMP\"");

usb ->downloadpcx_ USB (_T("../VC_SampleCode/UL.PCX "), "UL.PCX");

usb ->sendcommand_ USB ("PUTPCX 50,10,\"UL.PCX\"");

usb->windowsfont_USB(10, 20, 48, 0, 0, 0, "impact", "VC WIN TEST");

usb ->qrcode_USB("220", "260", "M", "3", "A", "0", "AABCB03abcN123");

usb ->printerfontblock_USB("35", "15", "790", "90", "1", "0", "8", "8", "0", "1", ttString);

usb->getDLLVersion_USB(1);

CString status;

status=usb->printerstatus_USB();

usb->printlabel_USB("1", "1");

```

//簡中打印

```

usb->clearbuffer_USB();

usb->printerfont_USB("10", "110", "TSS24.BF2", "0", "1", "1", L"默认简体中文测试: 海天米醋白
米醋 1 瓶 450ml");

usb->printlabel_USB("1", "1");

```

//繁中打印

```

usb->clearbuffer_USB();

usb->printerfont_USB("10", "110", "TST24.BF2", "0", "1", "1", L"默认繁体中文测试: 海天米醋白

```

```

米醋 1 瓶 450ml");

usb ->setDirectionAndMirror_USB(1,1);

usb ->setShift_USB(50);

usb ->setAfterPrintAction_USB(2);

usb ->setOffset_USB(20);

usb ->setCutMode_USB(0,2);

usb ->printReverse_USB(90,90,128,40);

usb ->printlabel_USB("1", "1");

usb ->sendcommand_USB("DIRECTION 1");

usb ->genericDefault_USB();

usb ->sensorDefault_USB();

// RFID

usb -> rfidSetupDefault_USB();

usb ->RFIDAutoCalibration_USB();

//UHF GEN2

usb ->writeUHF_USB("H", 2, 12, "E", "4141424243434444445454646"); // startBlockNo: Gen2
預設為 2

usb ->printlabel_USB("1", "1");

string data = usb->readUHF_USB("H", 0, 12, "E");

usb->EPCPWD_Action_USB("L","12345678");

usb->TIDPWD_Action_USB("L", "12345678");

usb->USERPWD_Action_USB("L", "12345678");

usb->AccessPWD_Action_USB("U", "12345678");

usb->KillPWD_Action_USB("U", "12345678");

usb->Set_RFIDPorcedure_USB(8,8,32,3,"N",2,2);

usb->Set_RFIDPorcedure_mm_USB(5, 40, 32, 5, "N", 5, 5, "203");

```

```

usb->writeHF_USB("H", 0, 12, "414142424343444445454646");

usb->printlabel_USB("1", "1");

//UHF GJB 写入资料

usb ->writeGJB_UHF_USB("H", 1, 12, "E", "414142424343444445454646", "12345678");

*帶入寫入密碼，寫入資料(startBlockNo: GJB 預設為 1)

//UHF GJB 设定各密码区域新密码

//帶入写入密码，设定新状态密码

usb ->setPWD_Action_USB ("S", "S", "11112222", "12345678");

//帶入写入密码，设定新读取密码

usb ->setPWD_Action_USB ("R", "S", "33334444", "12345678");

//帶入写入密码，设定新删除密码

usb ->setPWD_Action_USB ("K", "S", "55556666", "12345678");

//帶入旧写入密码，设定新写入密码*

usb ->setPWD_Action_USB ("W", "S", "87654321", "12345678");

//UHF GJB 设定资料区块状态

usb ->statusGJB_UHF_USB ("E", "B", "11112222"); //帶入状态密码，设定状态

usb ->printlabel_USB ("1", "1");

//UHF GJB 读取资料

//帶入读取密码，读取标签资料

string data =usb->readGJB_UHF_USB("H", 0, 12, "E", "33334444");

//UHF GJB 删除标签

usb ->killGJB_UHF_USB ("55556666"); //帶入删除密码，删除标签

usb ->printlabel_USB ("1", "1");

usb ->closeport_USB ();

usb->getDLLVersion_USB(1);

//指定网口传输介面

```

```

SocketConnect socketconnect = gcnew SocketConnect();

socketconnect->openport_Ethernet("xxx.xxx.xx.xxx", 9100);

socketconnect->setup_Ethernet ("54", "30", "2", "3", "0", "3", "0");

socketconnect->sendcommand_Ethernet ("DIRECTION 1");

socketconnect->clearbuffer_Ethernet ();

socketconnect->barcode_Ethernet ("30", "30", "128", "100", "1", "0", "2", "2",
"barcode1234567");

socketconnect->qrcode_Ethernet ("220", "260", "M", "3", "A", "0", "AABCB03abcN123");

socketconnect->downloadpcx_Ethernet (_T("../VC_SampleCode/UL.PCX "), "UL.PCX");

socketconnect->sendcommand_Ethernet ("PUTPCX 50,10,\"UL.PCX\"");

socketconnect->downloadbmp_Ethernet (_T("../VC_SampleCode/ CIRCLE.BMP "),
"CIRCLE.BMP");

socketconnect->sendcommand_Ethernet ("PUTBMP 150,30,\"CIRCLE.BMP\"");

socketconnect->windowsfont_Ethernet (10, 100, 48, 0, 0, 0, "arial", "VC WIN TEST");

socketconnect->qrcode_Ethernet("220", "260", "M", "3", "A", "0", "AABCB03abcN123");

socketconnect->printerfontblock_Ethernet ("35", "15", "790", "90", "1", "0", "8", "8", "0", "1",
ttString);

socketconnect->printlabel_Ethernet ("1", "1");

string status = socketconnect->printerstatus_Ethernet ();

//简中打印
string stString="默认简体中文测试";

socketconnect->clearbuffer_Ethernet ();

socketconnect->printerfont_Ethernet ("100", "10", "TSS24.BF2", "0", "1", "1", stString);

socketconnect->printlabel_Ethernet (1, 1, this);

//繁中打印
string ttString="默认繁体中文测试";

```

```
socketconnect->clearbuffer_Ethernet ();
```

```
socketconnect->printerfont_Ethernet ("100", "10", " TST24.BF2", "0", "1", "1", ttString);
```

```
socketconnect->printlabel_Ethernet (1, 1, this);
```

```
//BLOCK 打印
```

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

```
socketconnect->clearbuffer_Ethernet ();
```

```
socketconnect->printerfontblock_Ethernet ("35","15","790","90","1","0","8","8","0","1",  
ttString);
```

```
socketconnect ->setShift_Ethernet (50);
```

```
socketconnect ->setAfterPrintAction_Ethernet (2);
```

```
socketconnect ->setOffset_Ethernet (20);
```

```
socketconnect ->setCutMode_Ethernet (0,2);
```

```
socketconnect ->printReverse_Ethernet (90,90,128,40);
```

```
socketconnect ->printlabel_Ethernet ("1", "1");
```

```
socketconnect ->sendcommand_Ethernet ("DIRECTION 1");
```

```
socketconnect ->genericDefault_Ethernet ();
```

```
socketconnect ->sensorDefault_Ethernet ();
```

```
// RFID
```

```
socketconnect -> rfidSetupDefault_Ethernet ();
```

```
socketconnect ->RFIDAutoCalibration_Ethernet ();
```

```
//UHF GEN2
```

```
socketconnect->writeUHF_Ethernet("H", 2, 12,"E","414142424343444445454646");
```

```
//startBlockNo : Gen2 预设为 2
```

```
socketconnect->printlabel_Ethernet ("1", "1");
```

```
string data = socketconnect->readUHF_Ethernet ("H", 0, 12, "E");
```



```

string data_Q = socketconnect->query_UHF_Ethernet ( "A", 0, 0); //以 Q 指令读取 EPC 资料

socketconnect->EPCPWD_Action_Ethernet ("L", "12345678");

socketconnect->TIDPWD_Action_Ethernet ("L", "12345678");

socketconnect->USERPWD_Action_Ethernet ("L", "12345678");

socketconnect->AccessPWD_Action_Ethernet ("U", "12345678");

socketconnect->KillPWD_Action_Ethernet ("U", "12345678");

socketconnect->Set_RFIDPorcedure_Ethernet (8,8,32,3,"N",2,2);

socketconnect->Set_RFIDPorcedure_mm_Ethernet(5, 40, 32, 5, "N", 5, 5, "203");

socketconnect->writeHF_Ethernet ("H", 0, 12, "414142424343444445454646");

socketconnect->printlabel_Ethernet ("1", "1");

//UHF GJB 写入资料

socketconnect->writeGJB_UHF_Ethernet ("H", 1, 12, "E", "414142424343444445454646",
"12345678");

*带入写入密码，写入资料(startBlockNo: GJB 预设 为 1)

//UHF GJB 设定各密码区域新密码

socketconnect->setPWD_Action_Ethernet("S", "S", "11112222", "12345678"); //带入写入密码，
设定新状态密码

socketconnect->setPWD_Action_Ethernet ("R", "S", "33334444", "12345678");//带入写入密码，
设定新读取密码

socketconnect->setPWD_Action_Ethernet ("K", "S", "55556666", "12345678"); //带入写入
密码，设定新删除密码

socketconnect->setPWD_Action_Ethernet("W", "S", "87654321", "12345678"); //带入旧写入
密码，设定新写入密码

//UHF GJB 设定资料区块状态

socketconnect->statusGJB_UHF_Ethernet ("E", "B", "11112222"); //带入状态密码，设定状态

socketconnect->printlabel_Ethernet ("1", "1");

```

//UHF GJB 读取资料

```
string data = socketconnect->readGJB_UHF_Ethernet ("H", 0, 12, "E", "33334444");
```

//带入

读取密码，读取标签资料

//UHF GJB 删除标签

```
socketconnect->killGJB_UHF_Ethernet ("55556666"); //带入删除密码，删除标签
```

```
socketconnect->printlabel_Ethernet ("1", "1");
```

```
socketconnect->closePort_Ethernet (1000);
```

```
socketconnect->getDLLVersion_Ethernet (1);
```

1.使用管理员身分开启命令提示字符

2.注册 dll

将 x86 的 dll 放到 Windows/Syswow64, x64 的 dll 放在 Windows/System32

进入 microsoft.net framework 的安装路径下

如: cd C:\Windows\Microsoft.NET\Framework64\v4.0.30319

\*x64 的 dll 注册在 C:\Windows\Microsoft.NET\Framework64\v4.0.30319

\*x86 的 dll 注册在 C:\Windows\Microsoft.NET\Framework\v4.0.30319

输入指令 regasm.exe /register /tlb <dll path>

输入指令 regasm.exe <dll path> /codebase

如: regasm.exe /register /tlb C:\Users\User\Desktop\Javascript\_Sample\_Code\GTSPL\_SDK.dll

regasm.exe C:\Users\User\source\repos\product\GTSPL\_SDK\GTSPL\_SDK.dll /codebase

3.范例程序:

LOCAL driver as GTSPL\_SDK.Driver

LOCAL filePath

LOCAL printerStatus

filePath= JUSTPATH(SYS(16,0))

driver =CREATEOBJECT("GTSPL\_SDK.Driver")

driver.openport("Printer Model ")

driver.setup("54", "40", "2", "3", "0", "3", "0")

driver.sendcommand("DIRECTION 1")

driver.clearbuffer()

driver.barcode("30", "2", "128", "50", "1", "0", "2", "2", "barcode Foxpro Driver")

driver.printerfont("30", "75", "3", "0", "1", "1", "Print Font Foxpro Driver")

driver.downloadbmp(filePath+"\CIRCLE.BMP", "CIRCLE.BMP")

driver.sendcommand("PUTBMP 200,150,"+CHR(34)+"CIRCLE.BMP"+CHR(34))

```
driver.downloadbmp(filePath+"\\impact.ttf", "impact.ttf")  
driver.windowfont(10, 100, 36, 0, 0, 0, "impact", "Foxpro Driver")  
driver.printlabel("1", "1")  
driver.getDLLVersion(0)  
driver.closeport()
```

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		