设置缩放字体或点阵字体

```
/**
 *  设置缩放字体或点阵字体
 set scalable font or bitmapped font
 *
 *  @param fontName         (A~Z),(0~9)
 *  @param fieldOrientation (N,R,I,B)
 *  @param characterHeight  (10~32000)
 *  @param width            (10~32000)
 */
- (void)A_SetFont:(NSString *)fontName
 fieldOrientation:(NSString *)fieldOrientation
  characterHeight:(NSInteger)characterHeight
            width:(NSInteger)width;

- (void)A_SetFont:(NSString *)fontName fieldOrientation:(NSString
*)fieldOrientation;

- (void)A_SetFont:(NSString *)fontName
  characterHeight:(NSInteger)characterHeight
            width:(NSInteger)width;

- (void)A_SetFont:(NSString *)fontName;
```

使用字体名调用字体。

```
/**
 *  ^A@ 使用字体名调用字体。（The ^A@ command uses the complete name of a font, rather than the character designation used in ^A. Once a value for ^A@ is defined, it represents that font until a new font name is specified by ^A@.）
 *
 *  @param orientation  (N,R,I,B)
 *  @param location (R, E, B, A) defualt:R
 */
- (void)A_SetFontWithOrientation:(NSString *)orientation
                          height:(NSInteger)height
                           width:(NSInteger)width
                        location:(NSString *)location
                        fontName:(NSString *)fontName
                       extension:(NSString *)extension;

- (void)A_SetFontWithOrientation:(NSString *)orientation
                          height:(NSInteger)height
                           width:(NSInteger)width;
```

^B0 Aztec 条码

```
/**
 *  The ^B0 command creates a two-dimensional matrix symbology made up of square modules arranged around a bulls-eye pattern at the center.
 *
 *  @param orientation         (N,R,I,B)
 *  @param magnificationFactor (1~10)
 *  @param isContainECIC       (Y,N),default is N
 *  @param errorAndSymbol      (0),(01~99),(101~104),(201~232),(300)
 *  @param isMenuSymbol        (Y,N),default is N
 *  @param appendSymbolNumber  (1~26),default is 26
 *  @param appendOptionalID    (<= 24 characters)
 */
- (void)B0_BacodeAztecWithOrientation:(NSString *)orientation
                  magnificationFactor:(NSInteger)magnificationFactor
                        isContainECIC:(NSString *)isContainECIC
                       errorAndSymbol:(NSInteger)errorAndSymbol
                         isMenuSymbol:(NSString *)isMenuSymbol
                   appendSymbolNumber:(NSInteger)appendSymbolNumber
                     appendOptionalID:(NSString *)appendOptionalID;
```

^B1 Code 11条码

```
/**
 *  ^B1 Code 11条码（The ^B1 command produces the Code 11 bar code, also known
as USD-8 code. In a Code 11 bar code, each character is composed of three bars
and two spaces, and the character set includes 10 digits and the hyphen (-).）
 *
 *  @param orientation             (N,R,I,B)
 *  @param checkDigit              (Y,N)
 *  @param barcodeHeight           (1~32000)
 *  @param interpretationLine (Y,N) 是否打印注释行
 *  @param aboveCode               (Y,N) 注释行是否在条码上方
 */
- (void)B1_BacodeCode11WithOrientation:(NSString *)orientation
                            checkDigit:(NSString *)checkDigit
                         barcodeHeight:(NSInteger)barcodeHeight
                    interpretationLine:(NSString *)interpretationLine
                             aboveCode:(NSString *)aboveCode;
```

^B3 Code 39 码

```
/**
 *  ^B3 Code 39 码。（The Code 39 bar code is the standard for many industries,
including the U.S. Department of Defense. It is one of three symbologies
identified in the American National Standards Institute (ANSI) standard
MH10.8M-1983. Code 39 is also known as USD-3 Code and 3 of 9 Code.）
 *
 *  @param orientation             (N,R,I,B)
 *  @param checkDigit              (Y,N)
 *  @param barcodeHeight           (1~32000)
 *  @param interpretationLine (Y,N)
 *  @param aboveCode               (Y,N)
 */
- (void)B3_BacodeCode39WithOrientation:(NSString *)orientation
                            checkDigit:(NSString *)checkDigit
                         barcodeHeight:(NSInteger)barcodeHeight
                    interpretationLine:(NSString *)interpretationLine
                             aboveCode:(NSString *)aboveCode;
```

^B4 Code 49 码

```
/**
 *  ^B4 Code 49 码。
 *
 *  @param orientation        (N,R,I,B)
 *  @param barcodeHeight      (1~32000)
 *  @param interpretationLine (Y,N)
 *  @param aboveCode          (Y,N)
 */
- (void)B5_BacodePlanetCodeWithOrientation:(NSString *)orientation
                             barcodeHeight:(NSInteger)barcodeHeight
                        interpretationLine:(NSString *)interpretationLine
                                 aboveCode:(NSString *)aboveCode;
```

^B7 PDF417 码

```
/**
 *  ^B7 PDF417 码。(The ^B7 command produces the PDF417 bar code, a two-dimensional, multirow, continuous, stacked symbology. PDF417 is capable of encoding over 1,000 characters per bar code. It is ideally suited for applications requiring large amounts of information at the time the bar code is read.)
 *
 *  @param orientation   (N, R, I, B)
 *  @param barcodeHeight (1 to height of label)
 *  @param securityLevel (1-8)
 *  @param columns       (1-30)
 *  @param rows          (3-90)
 *  @param truncation    (N,Y)
 */
- (void)B7_BarcodePDF417CodeWithOrientation:(NSString *)orientation
                              barcodeHeight:(NSInteger)barcodeHeight
                              securityLevel:(NSInteger)securityLevel
                                    columns:(NSInteger)columns
                                       rows:(NSInteger)rows
                                 truncation:(NSString *)truncation;
```

^B8 EAN-8 条码

```
/**
 *  ^B8 EAN-8 条码。 (The ^B8 command is the shortened version of the EAN-13 bar code. EAN is an acronym for European Article Numbering. Each character in the EAN-8 bar code is composed of four elements: two bars and two spaces.)
 *
 *  @param orientation        (N,R,I,B)
 *  @param barcodeHeight      (1~32000)
 *  @param interpretationLine (Y,N)
 *  @param aboveCode          (Y,N)
 */
- (void)B8_BacodeEAN8WithOrientation:(NSString *)orientation
                       barcodeHeight:(NSInteger)barcodeHeight
                  interpretationLine:(NSString *)interpretationLine
                           aboveCode:(NSString *)aboveCode;
```

^B9 UPC-E 条码

```
/**
 *  ^B9 UPC-E 条码。 (The ^B9 command produces a variation of the UPC symbology used for number system 0. It is a shortened version of the UPC-A bar code, where zeros are suppressed, resulting in codes that require less printing space. The 6 dot/mm, 12 dot/mm, and 24 dot/mm printheads produce the UPC and EAN symbologies at 100 percent of their size. However, an 8 dot/mm printhead produces the UPC and EAN symbologies at a magnification factor of 77 percent.)
 *
 *  @param orientation        (N,R,I,B)
 *  @param barcodeHeight      (1~32000)
 *  @param interpretationLine (Y,N)
 *  @param aboveCode          (Y,N)
 *  @param checkDigit         (Y,N)
 */
- (void)B9_BarcodeUPCE8CodeWithOrientation:(NSString *)orientation
                             barcodeHeight:(NSInteger)barcodeHeight
                        interpretationLine:(NSString *)interpretationLine
                                 aboveCode:(NSString *)aboveCode
                                checkDigit:(NSString *)checkDigit
```

^BA Code 93 条码

```
/**
 *  ^BA Code 93 条码。 (The ^BA command creates a variable length, continuous
symbology. The Code 93 bar code is used in many of the same applications as
Code 39. It uses the full 128- character ASCII set. ZPL II, however, does not
support ASCII control codes or escape sequences. It uses the substitute
characters shown below.)
 *
 *  @param orientation        (N,R,I,B)
 *  @param barcodeHeight      (1~32000)
 *  @param interpretationLine (Y,N)
 *  @param aboveCode          (Y,N)
 *  @param checkDigit         (Y,N)
 */
- (void)BA_BarcodeCode93WithOrientation:(NSString *)orientation
                          barcodeHeight:(NSInteger)barcodeHeight
                     interpretationLine:(NSString *)interpretationLine
                              aboveCode:(NSString *)aboveCode
                             checkDigit:(NSString *)checkDigit
```

^BB CodeBlock 条码

```
/**
 *  ^BB CodeBlock 条码。 (The ^BB command produces a two-dimensional, multirow,
stacked symbology. It is ideally suited for applications that require large
amounts of information.)
 *
 *  @param orientation      (N,R,I,B)
 *  @param barcodeHeight    (1~32000)
 *  @param securityLevel    (Y,N)
 *  @param perRowCharacters (2-62 characters)
 *  @param rows             (1-22)
 *  @param mode             (A,E,F)
 */
- (void)BB_BarcodeCodeBlockWithOrientation:(NSString *)orientation
                             barcodeHeight:(NSInteger)barcodeHeight
                             securityLevel:(NSInteger)securityLevel
                          perRowCharacters:(NSInteger)perRowCharacters
                                      rows:(NSInteger)rows
                                      mode:(NSString *)mode;
```

^BC Code 128 条码

```
/**
 *  ^BC Code 128 条码。 (The ^BC command creates the Code 128 bar code, a high-density, variable length, continuous, alphanumeric symbology. It was designed for complexly encoded product identification.)
 *
 *  @param orientation        (N,R,I,B)
 *  @param barcodeHeight      (1~32000)
 *  @param interpretationLine (Y,N)
 *  @param aboveCode          (Y,N)
 *  @param checkDigit         (Y,N)
 *  @param mode               (N,U,A,D)
 */
- (void)BC_BarcodeCode128WithOrientation:(NSString *)orientation
                           barcodeHeight:(NSInteger)barcodeHeight
                      interpretationLine:(NSString *)interpretationLine
                               aboveCode:(NSString *)aboveCode
                              checkDigit:(NSString *)checkDigit
                                    mode:(NSString *)mode;
```

^BD UPS Maxicode 条码

```
/**
 *  ^BD UPS Maxicode 条码。 (The ^BD command creates a two-dimensional, optically read (not scanned) code. This symbology was developed by UPS (United Parcel Service).)
 *
 *  @param mode               (2-6) default:2
 *  @param symbolNumber       (1-8) default:1
 *  @param totalSymbolNumbers (1-8) default:1
 */
- (void)BD_BarcodeUPSMaxicodeWithMode:(NSInteger)mode
                         symbolNumber:(NSInteger)symbolNumber
                   totalSymbolNumbers:(NSInteger)totalSymbolNumbers;
```

^BE EAN-18 条码

```
/**
 *  ^BE EAN-18 条码。 (The ^BE command is similar to the UPC-A bar code. It is
widely used throughout Europe and Japan in the retail marketplace.)
 *
 *  @param orientation        (N,R,I,B)
 *  @param barcodeHeight      (1~32000)
 *  @param interpretationLine (Y,N)
 *  @param aboveCode          (Y,N)
 */
- (void)BE_BacodeEAN13WithOrientation:(NSString *)orientation
                        barcodeHeight:(NSInteger)barcodeHeight
                   interpretationLine:(NSString *)interpretationLine
                            aboveCode:(NSString *)aboveCode;
```

^BF 微型 PDF417 条码

```
/**
 *  ^BF 微型 PDF417 条码。 (The ^BF command creates a two-dimensional, multi-row,
continuous, stacked symbology identical to PDF417, except it replaces the 17-
module-wide start and stop patterns and left/right row indicators with a unique
set of 10-module-wide row address patterns. These reduce overall symbol width
and allow linear scanning at row heights as low as 2X.)
 *
 *  @param orientation        (N,R,I,B)
 *  @param barcodeHeight      (1~32000)
 *  @param mode               (0~33)
 */
- (void)BF_BarcodeMicroPDF417WithOrientation:(NSString *)orientation
                               barcodeHeight:(NSInteger)barcodeHeight
                                        mode:(NSInteger)mode;
```

^BI 工业二五码

```
/**
 *  ^BI 工业二五码。 (The ^BI command is a discrete, self-checking, continuous
numeric symbology. The Industrial 2 of 5 bar code has been in use the longest
of the 2 of 5 family of bar codes. Of that family, the Standard 2 of 5 (^BJ)
and Interleaved 2 of 5 (^B2) bar codes are also available in ZPL II.)
 *
 *  @param orientation        (N,R,I,B)
 *  @param barcodeHeight      (1~32000)
 *  @param interpretationLine (Y,N)
 *  @param aboveCode          (Y,N)
 */
- (void)BI_BacodeIndustrial2of5WithOrientation:(NSString *)orientation
                                 barcodeHeight:(NSInteger)barcodeHeight
                            interpretationLine:(NSString *)interpretationLine
                                     aboveCode:(NSString *)aboveCode;
```

^BJ 标准二五码

```
/**
 *  ^BJ 标准二五码。 (The ^BJ command is a discrete, self-checking, continuous
numeric symbology.)
 *
 *  @param orientation        (N,R,I,B)
 *  @param barcodeHeight      (1~32000)
 *  @param interpretationLine (Y,N)
 *  @param aboveCode          (Y,N)
 */
- (void)BJ_BacodeStandard2of5WithOrientation:(NSString *)orientation
                               barcodeHeight:(NSInteger)barcodeHeight
                          interpretationLine:(NSString *)interpretationLine
                                   aboveCode:(NSString *)aboveCode;
```

^BK ANSI Codebar条码

```
/**
 *  ^BK ANSI Codebar条码。（The ANSI Codabar bar code is used in a variety of information processing applications such as libraries, the medical industry, and overnight package delivery companies. This bar code is also known as USD-4 code, NW-7, and 2 of 7 code. It was originally developed for retail price labeling.）
 *
 *  @param orientation        (N,R,I,B)
 *  @param checkDigit         (N)
 *  @param barcodeHeight      (1-32000)
 *  @param interpretationLine (N,Y)
 *  @param aboveCode          (N,Y)
 *  @param startCharacter     (A,B,C,D)
 *  @param stopCharacter      (A,B,C,D)
 */
- (void)BK_BarcodeANSICodebarWithOrientation:(NSString *)orientation
                                  checkDigit:(NSString *)checkDigit
                               barcodeHeight:(NSInteger)barcodeHeight
                          interpretationLine:(NSString *)interpretationLine
                                   aboveCode:(NSString *)aboveCode
                              startCharacter:(NSString *)startCharacter
                               stopCharacter:(NSString *)stopCharacter;
```

^BL LOGMARS 条码

```
/**
 *  ^BL LOGMARS 条码。（The ^BL command is a special application of Code 39 used by the Department of Defense. LOGMARS is an acronym for Logistics Applications of Automated Marking and Reading Symbols.）
 *
 *  @param orientation        (N,R,I,B)
 *  @param barcodeHeight      (1~32000)
 *  @param interpretationLine (Y,N)
 *  @param aboveCode          (Y,N)
 */
- (void)BL_BacodeLOGMARSWithOrientation:(NSString *)orientation
                          barcodeHeight:(NSInteger)barcodeHeight
                     interpretationLine:(NSString *)interpretationLine
                              aboveCode:(NSString *)aboveCode;
```

^BQ QRcode 二维码

```
/**
 *  ^BQ QRcode 二维码。（The ^BQ command produces a matrix symbology consisting of an array of nominally square modules arranged in an overall square pattern. A unique pattern at three of the symbol’s four corners assists in determining bar code size, position, and inclination.）
 *
 *  @param orientation      (N,R,I,B)
 *  @param model            (1,original),(2,enhanced)
 *  @param magnification    (1-10)
 *  @param reliabilityLevel (H,Q,M,L)
 */
- (void)BQ_BarcodeQRcodeWithOrientation:(NSString *)orientation
                                  model:(NSInteger)model
                          magnification:(NSInteger)magnification
                       reliabilityLevel:(NSString *)reliabilityLevel;
```

^BS UPCE-EAN 码

```
/**
 *  The ^BS command is the two-digit and five-digit add-on used primarily by publishers to create bar codes for ISBNs (International Standard Book Numbers). These extensions are handled as separate bar codes.
 *
 *  @param orientation        (N,R,I,B)
 *  @param barcodeHeight      (1~32000)
 *  @param interpretationLine (Y,N)
 *  @param aboveCode          (Y,N)
 */
- (void)BS_BacodeUPCEANExtensionsWithOrientation:(NSString *)orientation
                                   barcodeHeight:(NSInteger)barcodeHeight
                              interpretationLine:(NSString *)interpretationLine
                                       aboveCode:(NSString *)aboveCode;
```

^BU UPC-A 条码

```
/**
 *  ^BU UPC-A 条码。 (The ^BU command produces a fixed length, numeric
symbology. It is primarily used in the retail industry for labeling packages.
The UPC-A bar code has 11 data characters. The 6 dot/mm, 12 dot/mm, and 24
dot/mm printheads produce the UPC-A bar code (UPC/EAN symbologies) at 100
percent size. However, an 8 dot/mm printhead produces the UPC/EAN symbologies
at a magnification factor of 77 percent.)
 *
 *  @param orientation        (N,R,I,B)
 *  @param barcodeHeight      (1~32000)
 *  @param interpretationLine (Y,N)
 *  @param aboveCode          (Y,N)
 *  @param checkDigit         (Y,N)
 */
- (void)BU_BarcodeUPCAWithOrientation:(NSString *)orientation
                        barcodeHeight:(NSInteger)barcodeHeight
                   interpretationLine:(NSString *)interpretationLine
                            aboveCode:(NSString *)aboveCode
                           checkDigit:(NSString *)checkDigit;
```

^BY 条码宽度、高度、比例调整

```
/**
 *  The ^BY command is used to change the default values for the module width
(in dots), the wide bar to narrow bar width ratio and the bar code height (in
dots). It can be used as often as necessary within a label format.
 *
 *  @param moduleWidth   (1-10)
 *  @param ratio         (2.0-3.0) default:3.0
 *  @param barcodeHeight (10)
 */
- (void)BY_BarcodeFieldDefaultWithModuleWidth:(NSInteger)moduleWidth
                                        ratio:(NSInteger)ratio
                                barcodeHeight:(NSInteger)barcodeHeight;

- (void)BY_BarcodeFieldDefaultWithModuleWidth:(NSInteger)moduleWidth;
```

改变格式指令前缀

```
/**
 *  改变格式指令前缀。缺省前缀是^(脱字符)。
  （The ^CC command is used to change the format command prefix. The default prefix is the caret (^).）
 *
 *  @param charactor (any ASCII character）
 */
- (void)CC_ChangeCaret:(NSString *)charactor;
```

改变 ZPL 分隔符

```
/**
 *  改变 ZPL 分隔符，缺省分隔符是‘,’。（The ^CD and ~CD commands are used to change the delimiter character. This character is used to separate parameter values associated with several ZPL II commands. The default delimiter is a comma (,)）
 *
 *  @param charactor (any ASCII character)
 */
- (void)CD_ChangeDelimiter:(NSString *)charactor;
```

改变默认字体

```
/**
 *  改变默认字体。（The ^CF command sets the default font used in your printer. You can use the ZPL Commands ^CF command to simplify your programs.）
 *
 *  @param font   (A-Z),(0~9)
 *  @param width  (0~32000) 不设置宽度：-1
 *  @param height (0~32000) 不设置宽度：-1
 */
- (void)CF_ChangeDefaultFont:(NSString *)font
                       width:(NSInteger)width
                      height:(NSInteger)height;

- (void)CF_ChangeDefaultFont:(NSString *)font;

- (void)CF_ChangeDefaultFontWidth:(NSInteger)width height:(NSInteger)height;
```

改变国际字体、编码

```
/**
 *  改变国际字体、编码。 (international character sets: U.S.A.1, U.S.A.2, UK,
Holland, Denmark/Norway, Sweden/Finland, Germany, France 1, France 2, Italy,
Spain, and several other sets, including the Unicode character set.)
 *
 *  @param characterSet (0-36)
 *  @param parameter    (0-255，多个参数以','分隔)
 */
- (void)CI_ChangeInternationalCharacterSet:(NSString *)characterSet
                                 parameter:(NSString *)parameter;
```

^CV CodeValidation

```
/**
 *  The ^CV command acts as a switch to turn the code validation function on
and off. When this command is turned on, all bar code data is checked for these
error conditions:
 *
 *  @param codeValidation (Y,N)
 */
- (void)CV_CodeValidation:(NSString *)codeValidation;
```

^CW 字体标识

```
/**
 *   字体标识。All built-in fonts are referenced using a one-character
identifier. The ^CW command assigns a single alphanumeric character to a font
stored in DRAM, memory card, EPROM, or Flash.
 *
 *  @param fontName          (A~Z),(0~9)
 *  @param fontDriveLocation (R, E, B, A)
 *  @param downloadedFont    (<= 8 characters)
 *  @param extension         (FNT,TTF,TTE)
 */
- (void)CW_FontIdentifier:(NSString *)fontName
        fontDriveLocation:(NSString *)fontDriveLocation
           downloadedFont:(NSString *)downloadedFont
                extension:(NSString *)extension;
```

^DF 下载格式

```
/**
 *  下载格式。The ^DF command saves ZPL II format commands as text strings to be later merged using ^XF with variable data. The format to be stored might contain field number (^FN) commands to be referenced when recalled.
 *
 *  @param deviceToStoreImage (R, E, B, A)
 *  @param imageName          (1 to 8 alphanumeric characters)
 *  @param extension          (.ZPL)
 */
- (void)DF_DownloadFormatWithDevice:(NSString *)deviceToStoreImage
                          imageName:(NSString *)imageName
                          extension:(NSString *)extension;
```

~DG 下载图像

```
/**
 *  下载图像。The ~DG command downloads an ASCII Hex representation of a graphic image. If .GRF is not the specified file extension, .GRF is automatically appended.
 *
 *  @param storeDevice  (R, E, B, A)
 *  @param imageName    (1 to 8 alphanumeric characters)
 *  @param extension    (.GRF)
 */
- (void)DG_DownloadGraphicsWithStoreDevice:(NSString *)storeDevice
                                 imageName:(NSString *)imageName
                                 extension:(NSString *)extension
                                imageWidth:(NSInteger)imageWidth
                                 imageData:(NSData *)imageData;
```

从 DRRAM 中删除所有图像

```
/**
 *  从 DRRAM 中删除所有图像
 */
- (void)EG_EraseDownloadGraphics;
```

^FD 字段数据

```
/**
 *  字段数据，使用“\&” 回车换行，使用“\(*)”作为软连字符（单词断开时使用），当连字符在一行的最后，会被打印出来，(*)表示任意的数字或字母
 *
 *  @param fieldData (up to 3072 bytes)
 */
- (void)FD_FieldData:(NSString *)fieldData;
```

^FH 字段的十六进制表示

```
/**
 *  字段的十六进制表示。The ^FH command allows you to enter the hexadecimal value for any character directly into the ^FD statement. The ^FH command must precede each ^FD command that uses hexadecimals in its field.
 *
 *  @param hexadecimalIndicator (any character except current format and control prefix (^ and ~ by default))
 */
- (void)FH_FieldHexadecimalIndicator:(NSString *)hexadecimalIndicator;
- (void)FH_FieldHexadecimal;
```

^FN 字段编号指令

```
/**
 *  字段编号指令。
 *
 *  @param fieldNumber 字段编号
 *  @param optional    (0-9999), default 1
 */
- (void)FN_FieldNumber:(NSInteger)fieldNumber
              optional:(NSString *)optional;

- (void)FN_FieldNumber:(NSInteger)fieldNumber;
```

^FO 字段位置

```
/**
 * 字段位置。The ^FO command sets a field origin, relative to the label home (^LH) position. ^FO sets the upper-left corner of the field area by defining points along the x-axis and y-axis independent of the rotation.
 *
 *  @param XAxis         (0~32000)
 *  @param YAxis         (0~32000)
 *  @param justification (0,1,2),当不设置对齐项时，传入-1
 */
- (void)FO_FieldOriginWithXAxis:(NSInteger)XAxis
                          YAxis:(NSInteger)YAxis
                  justification:(NSInteger)justification;

- (void)FO_FieldOriginWithXAxis:(NSInteger)XAxis
                          YAxis:(NSInteger)YAxis;
```

^FP 字段参数

```
/**
 *  字段参数。The ^FP command allows vertical and reverse formatting of the font field, commonly used for printing Asian fonts.
 *
 *  @param direction    (H,left to right),(V,top to bottom),(R, reverse printing), default is H
 *  @param characterGap (0~9999)
 */
- (void)FP_FieldParameterWithDirection:(NSString *)direction
                          characterGap:(NSInteger)characterGap;
```

^FR 字段反相打印

```
/**
 *  字段反相打印
 */
- (void)FR_FieldReversePrint;
```

^FS 字段分隔

```
/**
 *  字段分隔
 */
- (void)FS_FieldSeparator;
```

^FT 字段排版

```
/**
 *  字段排版。The ^FT command sets the field position, relative to the home position of the label designated by the ^LH command. The typesetting origin of the field is fixed with respect to the contents of the field and does not change with rotation.
 *
 *  @param XAxis         (0~32000)
 *  @param YAxis         (0~32000)
 *  @param justification (0,1,2)
 */
- (void)FT_FieldTypesetWithXAxis:(NSInteger)XAxis
                           YAxis:(NSInteger)YAxis
                   justification:(NSInteger)justification;

- (void)FT_FieldTypesetWithXAxis:(NSInteger)XAxis
                           YAxis:(NSInteger)YAxis;

- (void)FT_FieldTypeset;
```

^FV 可改变数据段

```
/**
 *  可改变数据段。^FV replaces the ^FD (field data) command in a label format when the field is variable.
 *
 *  @param variableFieldData (0~3072 byte string)
 */
- (void)FV_FieldVariable:(NSString *)variableFieldData;
```

^FW FieldOrientation Justification

```
/**
 *  The ^FW command sets the default orientation for all command fields that
have an orientation (rotation) parameter (and in x.14 sets the default
justification for all commands with a justification parameter). Fields can be
rotated 0, 90, 180, or 270 degrees clockwise by using this command. In x.14,
justification can be left, right, or auto.
 *
 *  @param fieldOrientation (N, R, I, B)
 *  @param justification    (0,1,2)
 */
- (void)FW_FieldOrientation:(NSString *)fieldOrientation
              justification:(NSInteger)justification;
- (void)FW_FieldOrientation:(NSString *)fieldOrientation;
```

^FX 字段注释

```
/**
 *  字段注释。The ^FX command is useful when you want to add non-printing
informational comments or statements within a label format. Any data after the
^FX command up to the next caret (^) or tilde (~) command does not have any
effect on the label format. Therefore, you should avoid using the caret (^) or
tilde (~) commands within the ^FX statement.
 *
 *  @param comment 注释字符
 */
- (void)FX_FieldComment:(NSString *)comment;
```

^GB 打印矩形

```
/**
 *  打印矩形。The ^GB command is used to draw boxes and lines as part of a label
format. Boxes and lines are used to highlight important information, divide
labels into distinct areas, or to improve the appearance of a label. The same
format command is used for drawing either boxes or lines.
 *
 *  @param width        (thickness~32000)
 *  @param height       (thickness~32000)
 *  @param thickness    (1~32000)
 *  @param lineColor    (B,black),(W,white) default:B
 *  @param cornorRoundingDegree (0~8)   default:0
 */
- (void)GB_GraphicBoxWithWidth:(NSInteger)width
                        height:(NSInteger)height
                     thickness:(NSInteger)thickness
                     lineColor:(NSString *)lineColor
          cornorRoundingDegree:(NSInteger)cornorRoundingDegree;

- (void)GB_GraphicBoxWithWidth:(NSInteger)width
                        height:(NSInteger)height
                     thickness:(NSInteger)thickness
          cornorRoundingDegree:(NSInteger)cornorRoundingDegree;

- (void)GB_GraphicBoxWithWidth:(NSInteger)width
                        height:(NSInteger)height
                     thickness:(NSInteger)thickness;
```

^GC 打印圆形

```
/**
 *  打印圆形。The ^GC command produces a circle on the printed label. The
command parameters specify the diameter (width) of the circle, outline
thickness, and color. Thickness extends inward from the outline.
 *
 *  @param diameter  (3~4095) default:3
 *  @param thickness (2~4095) default:1
 *  @param lineColor (B,W),black,white default:B
 */
- (void)GC_GraphicCircleWithDiameter:(NSInteger)diameter
                           thickness:(NSInteger)thickness
                           lineColor:(NSString *)lineColor;
```

^GD 打印斜线

```
/**
 *  The ^GD command produces a straight diagonal line on a label. This can be ZPL Commands Parameters used in conjunction with other graphic commands to create a more complex figure.
 *
 *  @param width        (thickness~32000)
 *  @param height       (thickness~32000)
 *  @param thickness    (1~32000)
 *  @param lineColor    (B,black),(W,white)
 *  @param orientation  (R,L)
 */
- (void)GD_GraphicDiagonalLineWithWidth:(NSInteger)width
                                 height:(NSInteger)height
                              thickness:(NSInteger)thickness
                              lineColor:(NSString *)lineColor
                            orientation:(NSString *)orientation;
```

^GE 打印椭圆

```
/**
 *  The ^GE command produces an ellipse in the label format.
 *
 *  @param width        (3~4095)
 *  @param height       (3~4095)
 *  @param thickness    (2~4095)
 *  @param lineColor    (B,black),(W,white)
 */
- (void)GE_GraphicEllipseWithWidth:(NSInteger)width
                            height:(NSInteger)height
                         thickness:(NSInteger)thickness
                         lineColor:(NSString *)lineColor;
```

^GS Graphic Symbol

```
/**
 *  The ^GS command enables you to generate the registered trademark, copyright
symbol, and other symbols.
 *
 *  @param orientation (N,R,I,B)
 *  @param height      (0-32000)
 *  @param width       (0-32000)
 */
- (void)GS_GraphicSymbolWithOrientation:(NSString *)orientation
                                 height:(NSInteger)height
                                  width:(NSInteger)width;

- (void)GS_GraphicSymbol;
```

^ID Image Delete

```
/**
 *  The ^ID command deletes objects, graphics, fonts, and stored formats from
storage areas. Objects can be deleted selectively or in groups. This command
can be used within a printing format to delete objects before saving new ones,
or in a stand-alone format to delete objects.
 *
 *  @param objectLocation (R, E, B, A)
 *  @param objectName     (any 1 to 8 character name)
 *  @param extension      (.GRF)
 */
- (void)ID_ImageDeleteWithObjectLocation:(NSString *)objectLocation
                              objectName:(NSString *)objectName
                               extension:(NSString *)extension;
```

^IL Image Load

```
/**
 *  The ^IL command is used at the beginning of a label format to load a stored
image of a format and merge it with additional data. The image is always
positioned at ^FO0,0.
 *
 *  @param objectLocation (R, E, B, A)
 *  @param objectName     (1 to 8 alphanumeric characters)
 *  @param extension      (.GRF, .PNG)
 */
- (void)IL_ImageLoadWithObjectLocation:(NSString *)objectLocation
                            objectName:(NSString *)objectName
                             extension:(NSString *)extension;
```

^IM Image Move

```
/**
 *  The ^IM command performs a direct move of an image from storage area into the bitmap. The command is identical to the ^XG command (Recall Graphic), except there are no sizing parameters.
 *
 *  @param objectLocation (R, E, B, A)
 *  @param objectName     (1 to 8 alphanumeric characters)
 *  @param extension      (.GRF, .PNG)
 */
- (void)IM_ImageMoveWithObjectLocation:(NSString *)objectLocation
                            objectName:(NSString *)objectName
                             extension:(NSString *)extension;
```

^IS Image Save

```
/**
 *  The ^IS command is used within a label format to save that format as a graphic image, rather than as a ZPL II script. It is typically used toward the end of a script. The saved image can later be recalled with virtually no formatting time and overlaid with variable data to form a complete label.
 *
 *  @param objectLocation (R, E, B, A)
 *  @param objectName     (1 to 8 alphanumeric characters)
 *  @param extension      (.GRF, .PNG)
 *  @param printAfterSorting (Y,N)
 */
- (void)IS_ImageSaveWithObjectLocation:(NSString *)objectLocation
                            objectName:(NSString *)objectName
                             extension:(NSString *)extension
                     printAfterSorting:(NSString *)printAfterSorting;
```

~JA Cancel All

```
- (void)JA_CancelAll;
```

~JD Enable Communications Diagnostics

```
- (void)JD_EnableCommunicationsDiagnostics;
```

~JE Disable Diagnostics

```
- (void)JE_DisableDiagnostics;
```

^JZ Set Reprin tAfter Error

```
/**
 *
 *
 *  @param enable (Y/N)
 */
- (void)JZ_SetReprintAfterError:(NSString *)enable;
```

^LH LabelHome

```
/**
 *  @param XPos (0~32000)
 *  @param YPos (0~32000)
 */
- (void)LH_LabelHomeWithXPos:(NSInteger)XPos
                        YPos:(NSInteger)YPos;
```

^LL LabelLength

```
- (void)LL_LabelLength:(NSInteger)length;
```

^LR Label Reverse Print

```
/**
 *  The ^LR command reverses the printing of all fields in the label format. It
allows a field to appear as white over black or black over white.
 *
 *  @param reverse (Y,N)
 */
- (void)LR_LabelReversePrint:(NSString *)reverse;
```

^LS Label Shift

```
/**
 *  @param shift (-9999~9999)
 */
- (void)LS_LabelShift:(NSInteger)shift;
```

^LT LabelTop

```
/**
 *  The ^LT command moves the entire label format a maximum of 120 dot rows up
or down from its current position, in relation to the top edge of the label. A
negative value moves the format towards the top of the label; a positive value
moves the format away from the top of the label.
 *
 *  @param top (-120~120)
 */
- (void)LT_LabelTop:(NSInteger)top;
```

^MC MapClear

```
/**
 *  In normal operation, the bitmap is cleared after the format has been
printed. The ^MC command is used to retain the current bitmap. This applies to
current and subsequent labels until cleared with ^MCY.
 *
 *  @param clear (Y,N)
 */
- (void)MC_MapClear:(NSString *)clear;
```

^ML Maximum Label Length

```
/**
 *  The ^ML command lets you adjust the maximum label length
 *
 *  @param length 0 to maximum length of label
 */
- (void)ML_MaximumLabelLength:(NSInteger)length;
```

^MT Media Type

```
/**
 *  The ^MT command selects the type of media being used in the printer. These
are the choices for this command:
 *
 *  @param type (T = thermal transfer media),(D = direct thermal media)
 */
- (void)MT_MediaType:(NSString *)type;
```

^MT Set Media Type

```
/**
 *  @param type (T/D)
 */
- (void)MT_SetMediaType:(NSString *)type;
```

^PM Print Label Mirror Image

```
/**
 *  The ^PM command prints the entire printable area of the label as a mirror image. This command flips the image from left to right.
 *
 *  @param mirror (Y,N)
 */
- (void)PM_PrintLabelMirrorImage:(NSString *)mirror;
```

^PO Print Orientation

```
/**
 *  The ^PO command inverts the label format 180 degrees. The label appears to be printed upside down. If the original label contains commands such as ^LL, ^LS, ^LT and ^PF, the inverted label output is affected differently.
 *
 *  @param orientation (N,I), normal,invert
 */
- (void)PO_PrintOrientation:(NSString *)orientation;
```

^PP ~PP Programmable Pause

```
- (void)PP_ProgrammablePause;
```

^PQ Print Quantity

```
/**
 *  The ^PQ command gives control over several printing operations. It controls the number of labels to print, the number of labels printed before printer pauses, and the number of replications of each serial number.
 *
 *  @param quantity       (1-99 999 999)
 *  @param pauseValue     (1-99 999 999)
 *  @param replicateValue (0-99 999 999)
 *  @param overrided      (N,Y)
 */
- (void)PQ_PrintQuantity:(NSInteger)quantity
              pauseValue:(NSInteger)pauseValue
          replicateValue:(NSInteger)replicateValue
               overrided:(NSString *)overrided;

- (void)PQ_PrintQuantity:(NSInteger)quantity;
```

^PW Print Width

```
/**
 *  @param width (0~864)
 */
- (void)PW_PrintWidth:(NSInteger)width;
```

~SD Set Darkness

```
/**
 *  The ~SD command allows you to set the darkness of printing. ~SD is the equivalent of the darkness setting parameter on the control panel display.
 *
 *  @param darkness (0~30)
 */
- (void)SD_SetDarkness:(NSInteger)darkness;
```

^SF Serialization Field

```
/**
 *  The ^SF command allows you to serialize a standard ^FD string. The maximum
size of the mask and increment string is 3K combined.
 *
 *  @param maskString      (D,H,O,A,N)
 *  @param incrementString ()
 */
- (void)SF_SerializationFieldWithMaskString:(NSString *)maskString
                            incrementString:(NSString *)incrementString;
```

^TO Transfer Object

```
/**
 *  The ^TO command is used to copy an object or group of objects from one
storage device to another. It is similar to the copy function used in PCs.
 *
 *  @param sourceDevice          (R, E, B, A)
 *  @param destinationDevice     (R, E, B, A)
 */
- (void)TO_TransferObjectWithSourceDevice:(NSString *)sourceDevice
                         sourceObjectName:(NSString *)sourceObjectName
                          sourceExtension:(NSString *)sourceExtension
                        destinationDevice:(NSString *)destinationDevice
                    destinationObjectName:(NSString *)destinationObjectName
                     destinationExtension:(NSString *)destinationExtension;
```

^WC Print Configuration Label

```
- (void)WC_PrintConfigurationLabel;
```

^XA Format Start

```
- (void)XA_FormatStart;
```

^XF Recall Format

```
/**
 *  The ^XF command recalls a stored format to be merged with variable data. There can be multiple ^XF commands in one format, and they can be located anywhere within the code.
 *
 *  @param sourceDevice (R, E, B, A)
 *  @param imageName    1 to 8 alphanumeric characters
 *  @param extension    (.ZPL)
 */
- (void)XF_RecallFormatWithSourceDevice:(NSString *)sourceDevice
                              imageName:(NSString *)imageName
                              extension:(NSString *)extension;
```

^XG Recall Graphic

```
/**
 *  The ^XG command is used to recall one or more graphic images for printing. This command is used in a label format to merge graphics, such as company logos and piece parts, with text data to form a complete label.
 *
 *  @param sourceDevice       (R, E, B, A)
 *  @param imageName          1 to 8 alphanumeric characters
 *  @param extension          (.GRF)
 *  @param XAxisMagnification (1~10)
 *  @param YAxisMagnification (1~10)
 */
- (void)XG_RecallGraphicWithSourceDevice:(NSString *)sourceDevice
                               imageName:(NSString *)imageName
                               extension:(NSString *)extension
                      XAxisMagnification:(NSInteger)XAxisMagnification
                      YAxisMagnification:(NSInteger)YAxisMagnification;
```

^XZ Format End

```
- (void)XZ_FormatEnd;
```

^PR Speed

```
- (void)PR_SetSpeed:(NSInteger)speed
          slewSpeed:(NSInteger)slewSpeed
      backfeedSpeed:(NSInteger)backfeedSpeed;
```