

# **Aclas Barcode Label Scale and Touch Scale**

---

Programming Manual

V2.0

2019/1/11

## CONTENT

1. Introduction .....	1
2. Dynamic Link Library Interface.....	2
2.1 Dynamic Link Library Calling Process.....	2
2.2 Dynamic Link Library Function Declaration.....	3
3. Command Line Interface .....	6
3.1 Command Line Parameter Description .....	6
3.2 Command Line Usage Examples.....	6
4. Type Definition.....	8
4.1 Protocol Type (ProtocolType).....	8
4.2 Operation Type (ProcType) .....	8
4.3 Data Type (DataType).....	8
4.4 Error Code (ErrorCode).....	9
4.5 Device Information Structure Definition (TASSDKDeviceInfo).....	10
5. Data File Format.....	11
5.1 File Format Description .....	11
5.2 File Examples .....	11
5.3 File Encoding Format.....	11
5.4 File Format Details.....	12
5.4.1 PLU File (PLU).....	12
5.4.2 Note File (Note1/2/3/4).....	14
5.4.3 Department File (Department).....	15
5.4.4 Group File (Group).....	15
5.4.5 Hotkey File (HotKey) .....	16
5.4.6 Discount Schedule File (Discount).....	16
5.4.7 Advertising Information File (AdverisementInfo).....	17
5.4.8 Label File (Label) .....	17
5.4.9 Sale Record (SaleRecord) .....	18
5.4.10 Traceability File (Trace) .....	18
5.4.11 Information (Message1).....	19



# 1. Introduction

This document will introduce how to call Aclas Scale SDK interface to communicate with Aclas Barcode Label Scale.

Supported devices: Aclas LH51 Series, LS M3 Series Barcode Label Scale and TS Series Touch Barcode Label Scale

Supported operating systems: Windows and Linux

Interface modes: Dynamic link library interface (DLL or SO) and Command line interface (Console)

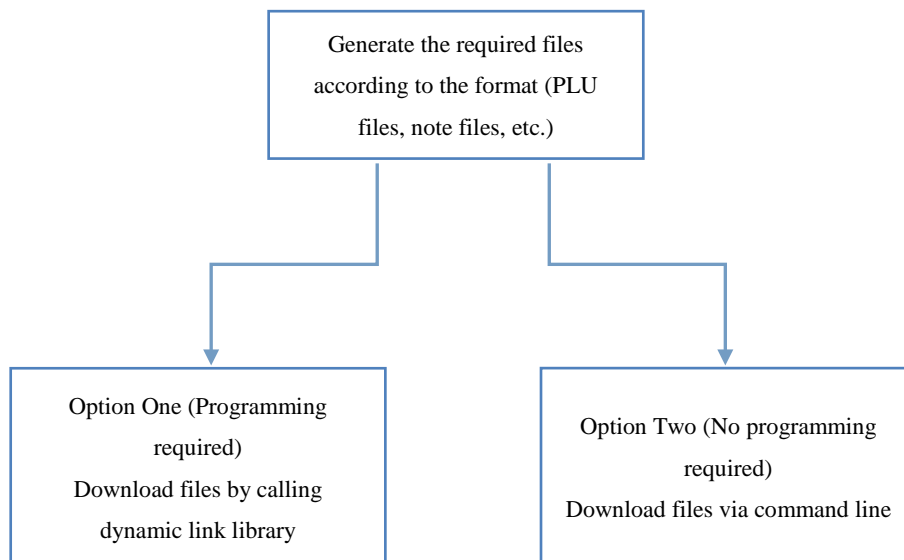
File List:

- |                       |                                  |
|-----------------------|----------------------------------|
| 1、AclasSDK.dll        | ----Windows dynamic link library |
| 2、AclasSDKConsole.exe | ----Windows command line program |
| 3、AclasSDK.so         | ----Linux dynamic link library   |
| 3、AclasSDKConsole     | ----Linux command line program   |

The command line program relies on the dynamic link library, for example, if called by a console interface, it should attach dynamic link library simultaneous. However, dynamic link library can be called without command line program.

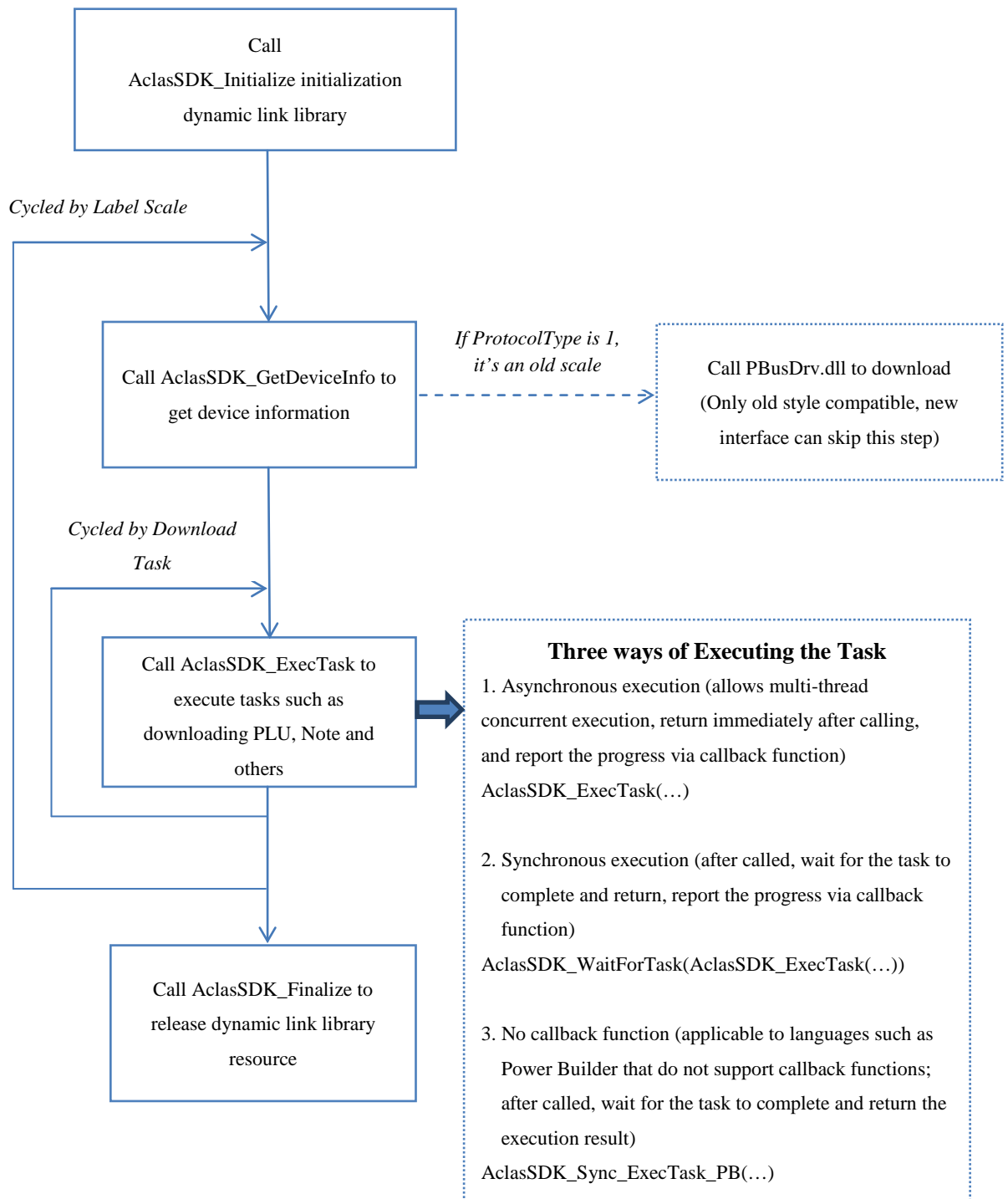
PLU files and note files can be downloaded via SDK interface generally. For touch scale, department files and group files' downloading are also required. For other infrequent or one-time configuration data, to finish via Aclas Label Scale upper-computer software Link66 are recommended.

Interface Calling Process:



## 2. Dynamic Link Library Interface

### 2.1 Dynamic Link Library Calling Process



## 2.2 Dynamic Link Library Function Declaration

### 2.2.1 Initialization

Pascal	function AclasSDK_Initialize(Adjuct: Pointer): Boolean; stdcall;		
Description	Used to initialize the dynamic link library.		
	Parameter	Type	Description
Input	Adjuct	Pointer	Reserved character, transmit nil or null
Return		Boolean	True for success, False for fail.

### 2.2.2 Release

Pascal	procedure AclasSDK_Finalize; stdcall;		
Description	Used to release resources allocated by dynamic link library.		

### 2.2.3 Get device protocol type

Pascal	function AclasSDK_GetDeviceType(Addr, Port, ProtocolType: UInt32): Integer;		
Description	Used to get device protocol type. The return value can be used to distinguish scale types. If the return value is “1”, it should be Aclas old type label scale, please call old label dynamic link library PBusDrv.dll.		
	Parameter	Type	Description
Input	Addr	UInt32	Device IP address (integer)
	Port	UInt32	Device port No., input 0
	ProtocolType	UInt32	Protocol type, input 0
Return		Integer	0: Unknown, other return value see 4.1

Explanation: Integer IP address: The integer (UInt32) expression format of IP address.

E.g. “192.168.2.208” converts to integer format is:

**192**\*256\*256\*256+**168**\*256\*256+**2**\*256+**208**=3232236240

### 2.2.4 Get device information

Pascal	function AclasSDK_GetDeviceInfo(Addr, Port, ProtocolType: UInt32): TASSDKDeviceInfo; C# can adopt functions below function AclasSDK_GetDeviceInfo(Addr, Port, ProtocolType: UInt32; var DeviceInfo TASSDKDeviceInfo):Boolean;		
Description	Used to get device information.		
	Parameter	Type	Description
Input	Addr	UInt32	Device IP address (integer)
	Port	UInt32	Device port No., input 0
	ProtocolType	UInt32	Protocol type, see 4.1
Return		TASSDKDeviceInfo	See 4.5

### 2.2.5 Execute task

Pascal	function AclasSDK_ExecTask(Addr, Port, ProtocolType, ProcType, DataType: UInt32; FileName: PWideChar; OnProgress: TASSDKOnProgressEvent; UserData: Pointer): THandle; stdcall;		
--------	--	--	--

	// Ansi version function AclasSDK_ExecTaskA(Addr, Port, ProtocolType, ProcType, DataType: UInt32; FileName: PAnsiChar; OnProgress: TASSDKOnProgressEvent; UserData: Pointer): THandle; stdcall;		
Description	Used to download upload data. With executing asynchronous, the function will return immediately after calling, and report specific progress via callback function. 1. If task need to be cancelled, please call AclasSDK_StopTask. 2. If need to execute synchronous, please call it in the form of AclasSDK_WaitForTask(AclasSDK_ExecTask(...)). 3. If the language used does not support callback function (E.g. PowerBuilder), please use AclasSDK_Sync_ExecTask_PB.		
	Parameter	Type	Description
Input	Addr	UInt32	Device IP address (integer)
	Port	UInt32	Device port No., input 0
	ProtocolType	UInt32	Protocol type, see 4.1
	ProcType	UInt32	Operation type, see 4.2
	DataType	UInt32	Data type, see 4.3
	FileName	string	Upload, modify, download the absolute address of the file AclasSDK_ExecTask is WideString AclasSDK_ExecTaskA is AnsiString
	OnProgress	TASSDKOnProgressEvent	Progress callback function
	UserData	Pointer	User defined data pointer, SDK does not perform any operation on this value, and remain unchanged to return when callback. This parameter aims to distinguish tasks in callback function when multiple threads execute multiple tasks asynchronously.
Return		THandle	Task handle, when returned -1, task fail.

#### 2.2.6 Stop task

Pascal	procedure AclasSDK_StopTask(TaskHandle: THandle = 0); stdcall;		
Description	Used to cancel a task.		
	Parameter	Type	Description
Input	TaskHandle	THandle	Task handle, 0 stands for stop all tasks.

#### 2.2.7 Waiting for task completion

Pascal	procedure AclasSDK_WaitForTask(TaskHandle: THandle); stdcall;		
Description	Used to wait for task complete before returning, means execute task synchronously.		
	Parameter	Type	Description
Input	TaskHandle	THandle	Task handle

#### 2.2.8 Progress callback function

Pascal	TASSDKOnProgressEvent = procedure(nErrorCode, Index, Total: UInt32; lpUserData: Pointer); stdcall;		
	Parameter	Type	Description
Input	nErrorCode	UInt32	Error code, see 4.4
	Index	UInt32	Current progress
	Total	UInt32	Sum

2.2.9 Execute task (applicable to programming languages without callback function such as Power Builder, referable for other languages)

Pascal	function AclasSDK_Sync_ExecTask_PB(Addr: PWideChar; Port, ProtocolType, ProcType, DataType: UInt32; FileName: PWideChar): Integer; stdcall;  // Ansi version function AclasSDK_Sync_ExecTaskA_PB(Addr: PAnsiChar; Port, ProtocolType, ProcType, DataType: UInt32; FileName: PAnsiChar): Integer; stdcall;  //Power Builder please use Ansi version, function declaration of Power Builder shows below function int AclasSDK_Sync_ExecTaskA_PB(ref string Addr, uint Port, uint ProtocolType, uint ProcType, uint DataType, ref string FileName)		
	Parameter	Type	Description
Input	Addr	string	Device IP address, e.g.: 192.168.0.2
	Port	UInt32	Device port No., input 0
	ProtocolType	UInt32	Protocol type, see 4.1
	ProcType	UInt32	Operation type, see 4.2
	DataType	UInt32	Data type, see 4.3
	FileName	string	Upload, modify, download the absolute address of the file AclasSDK_ExecTask is WideString AclasSDK_ExecTaskW is WideString AclasSDK_ExecTaskA is AnsiString
Return		Integer	0: Execution success. Other error code, see 4.4



## 3. Command Line Interface

Call AclasSDKConsole.exe and bring in the parameters to execute the operation by using the command line calling method.

### 3.1 Command Line Parameter Description

Parameter	Function	Value	Description
-h	Device IP address	E.g.: "192.168.2.208"	Assigned device IP address
-p	Protocol type	The default is None, means automatic detection protocol. See 4.1	
-t	Operation type	See 4.2	
-b	Data type	See 4.3	
-d	Data sequence number	Mainly used to transmit the label sequence number, default value is 0. See 5.4.8	
-n	Filename	E.g.: "d:\plu.txt"	Filename

Return value (PostQuitMessage):

- 0: Execution success
- 1: Device offline
- 2: Parameter error
- 3: Lack parameter
- 4: Execution error

### 3.2 Command Line Usage Examples

Function: Download the PLU data d:\PLU.txt to the device on 192.168.2.208

Command: AclasSDKConsole.exe -h "192.168.2.208" -t Down -b PLU -n "d:\PLU.txt"

Function: Upload PLU data from the device of 192.168.2.208 to d:\PLU.txt

Command: AclasSDKConsole.exe -h "192.168.2.208" -t Up -b PLU -n "d:\PLU.txt"

Function: Clear the PLU data on device of 192.168.2.208

Command: AclasSDKConsole.exe -h "192.168.2.208" -t Del -b PLU -n "\*"

Function: Download the Note1 data d:\Note1.txt to the device on 192.168.2.208

Command: AclasSDKConsole.exe -h "192.168.2.208" -t Down -b Note1 -n "d:\Note1.txt"

Function: Download the Hotkey data d:\hotkey.txt to the device on 192.168.2.208

Command: AclasSDKConsole.exe -h "192.168.2.208" -t Down -b Hotkey -n "d:\hotkey.txt"

Function: Download the Hotkey images d:\900001.jpg to the Touch Scale on 192.168.2.208

Command: AclasSDKConsole.exe -h "192.168.2.208" -t Down -b KeyImage -n "d:\900001.jpg "

Function: Synchronize device time in 192.168.2.208

Command: AclasSDKConsole.exe -h "192.168.2.208" -t Down -n "2008-08-08 08:08:08" -b Time

Function: Download the label format file d:\label.lf to the No. 1 label of device on 192.168.2.208.

Command: AclasSDKConsole.exe -h "192.168.2.208" -t Down -b LabelFormat -d 0 -n "d:\label.lf"

Function: Download the label shading file d:\label.lm to the No. 1 label of device on 192.168.2.208.

Command: AclasSDKConsole.exe -h "192.168.2.208" -t Down -b LabelMap -d 0 -n "d:\label.lm"

Function: Download the label design file d:\label.tbl to the No. 1 label of device on 192.168.2.208.

Command: AclasSDKConsole.exe -h "192.168.2.208" -t Down -b LabelDesign -d 0 -n "d:\label.tbl"

Function: Upload the label design file of No. 1 label from the device on 192.168.2.208 to d:\label.tbl.

Command: AclasSDKConsole.exe -h "192.168.2.208" -t Up -b LabelDesign -d 0 -n "d:\label.tbl"

## 4. Type Definition

### 4.1 Protocol Type (ProtocolType)

ProtocolType(-p)		
DLL value	Console command	Description
0	None	Automatic detection
1	Pecr	Pecr protocol
2	Hecr	Hecr protocol
3	TSecr	TSecr protocol
Pecr protocol for LH51 or LH51 mode of M3 Hecr protocol for M3 series TSecr protocol for TS series Set “None”, the SDK will automatically detect the model protocol, and preset protocol parameters to save the detection time before transmission.		

### 4.2 Operation Type (ProcType)

ProcType(-t)		
DLL value	Console command	Description
0	Down	Download
1	UP	Upload
2	Edit	Edit
3	Del	Delete
4	List	TS model to get images, videos and file list.
Set task operation type		

### 4.3 Data Type (DataType)

DataType(-b)				
DLL value (Decimalism)	DLL value (Hexadecimal)	Console command	Description	Text Format
0	0x0000	PLU	PLU data	Text file
12	0x000C	Note1	Note 1	Text file
13	0x000D	Note2	Note 2	Text file
14	0x000E	Note3	Note 3	Text file
28	0x001C	Note4	Note 4	Text file
2	0x0002	Department	Department	Text file

4	0x0004	Group	Grouping	Text file
3	0x0003	HotKey	Hotkey	Text file
5	0x0005	Discount	Discount schedule	Text file
21	0x0015	AdvertisementInfo	Advertising information	Text file
32	0x0020	LabelFormat	Label format file	Binary file
33	0x0021	LabelMap	Label shading file	Binary file
34	0x0022	LabelDesign	Label design file	Binary file
261	0x0105	Message1	Information	Text file
263	0x0107	SaleRecord	Sale record	Text file
8193	0x2001	Time	Time	None
The following is exclusive to LS				
288	0x0120	TraceStatus	Traceability configuration file	Text file
289	0x0121	TraceMap	Traceability index file	Text file
290	0x0122	TraceData	Traceability data file	Text file
The following is exclusive to TS				
4101	0x1005	KeyImage	Hotkey image	JPG file
4357	0x1005	BatchKeyImage	Batch hotkey image	ZIP file

## 4.4 Error Code (ErrorCode)

ErrorCode		
DLL value (Decimalism)	DLL value (Hexadecimal)	Description
0	0	Normal
1	0x0001	Progress event
2	0x0002	Manual stop
256	0x0100	Initialized
257	0x0101	Uninitialized
258	0x0102	Device does not exist
259	0x0103	Unsupported protocol type
260	0x0104	This data type does not support this operation
261	0x0105	Not support this data type
264	0x0108	Unable to open input file
265	0x0109	The number of fields does not match the number of content
266	0x010A	Communication data exception
267	0x010B	Parsing data exception
268	0x010C	CodePage error
269	0x010D	Unable to create output file

## 4.5 Device Information Structure Definition (TASSDKDeviceInfo)

TASSDKDeviceInfo			
Parameter	Type	Size	Description
ProtocolType	UInt32	4	Protocol type
Addr	UInt32	4	Address
Port	UInt32	4	Port
Name	AnsiChar	16	Device name
ID	UInt32	4	Device ID
Version	UInt32	4	Device software version
Country	UInt8	1	Country category
DepartmentID	UInt8	1	Department ID
KeyType	UInt8	1	Keyboard type
PrinterDot	UInt64	8	Points printed by printer head
PrnStartDate	TDateTime	8	Printer head activation time
LabelPage	UInt32	4	Number of labels printed by the printer head
PrinterNo	UInt32	4	Printer head series number
PLUStorage	UInt16	2	PLU storable quantity
HotKeyCount	UInt16	2	Number of hotkeys supported
NutritionStorage	UInt16	2	Nutrition information storable quantity
DiscountStorage	UInt16	2	Discount schedule storable quantity
Note1Storage	UInt16	2	Note1 storable quantity
Note2Storage	UInt16	2	Note2 storable quantity
Note3Storage	UInt16	2	Note3 storable quantity
Note4Storage	UInt16	2	Note4 storable quantity
Adjunct	AnsiChar	177	Reserved fields
The entire structure size is 256			

## 5. Data File Format

### 5.1 File Format Description

- The first line is the field name, and the official data begins from the second line. The order and the number of fields are not mandatory, but the data needs to correspond to the fields one by one.
- Each field or data is separated by a TAB (0x09). Separate each line with a carriage return (0x0D 0x0A).
- The above separator appears in the data, which is escaped according to the following rules.

Original character		Replace with
0x09	<->	{ \$09 }
0x0A	<->	{ \$0A }
0x0D	<->	{ \$0D }

If there are { \$09 }, { \$0A }, { \$0D }, etc. in the original string, please escape it yourself.

### 5.2 File Examples

The following is the example of PLU file:

ID	DepartmentID	GroupID	Name1	Price	BarcodeType1
90001	20	1	Apple	9.99	40
90002	20	1	Pear	8.88	40
90003	20	2	Banana	3.00	40

### 5.3 File Encoding Format

**Input file:** Refers to the file specified when calling interface downloads data to the scale. Because this is the time to read an existing file, the DLL will automatically determine the file encoding format based on the text file encoding header. For example, in Ansi format, CodePage is parsed with the default CodePage of the current operating system. The UTF8 format is recommended.

**Output file:** Refers to the file specified when calling interface uploads data from the scale. Because this is the time to create a new file, the DLL will default to use the Ansi format in order to ensure versatility. If other encoding format needed, please specify it by adding "? Encoding format" after the file name.

Ansi: 0

Unicode: 1200

BigEndianUnicode: 1201

UTF8: 65001

E.g.: plu.txt?65001 will create a text file named plu.txt with UTF8 encoding format.

## 5.4 File Format Details

Field name with bold is the primary key.

### 5.4.1 PLU File (PLU)

The fields in the PLU file are very rich, but these fields are not all necessary: the underlined fields are required fields. Please include these fields in the PLU file; other fields can be selected or not included as needed.

PLU																								
Field name	Type	Range	Description																					
<u>ID</u>	Int	999999	LFCODE, unique identification PLU																					
<u>ItemCode</u>	Str	16	Item No., major used to construct the barcode, and is generally the same as the LFCODE. Please refer to Appendix Barcode Coding Comparison Table																					
<u>DepartmentID</u>	Int	0..99	Department ID. Major used to construct the barcode. Please refer to Appendix: Barcode Coding Comparison Table <u>For 1 digit department, 2 is recommended; for 2 digits department, 20 or 22 is recommended.</u>																					
<u>GroupID</u>	Int	0..9999	Group ID. This field is required for the touch scale, associated with the ID in the Group table and not necessary for normal touch-tone label scale.																					
<u>Name1</u>	Str	40	Name 1																					
<u>Name2</u>	Str	40	Name 2																					
<u>Name3</u>	Str	40	Name 3																					
<u>Price</u>	Float		Unit price, up to 2 decimal places, up to 999999.99																					
<u>UnitID</u>	Int		<u>If not familiar with this part, please use proposed value: 4 (kg) for weighing, and 10 (PCS) for counting.</u> Weighing unit number, value means: 0-50g; 1-g; 2-10g; 3-100g; 4-kg; 5-oz; 6-lb; 7-500g; 8-600g; 9-pcs(g); 10-pcs(kg); 11-pcs(oz); 12-pcs(lb)																					
<u>BarcodeType1</u>	Int	0..255	Barcode type 1. 0-149: built-in; 150-255: user-defined. Please refer to Appendix: Barcode Coding Comparison Table <u>If not familiar with this part, please refer to following suggestions:</u> <table><tr><th>Item No. Digit</th><th>Barcode Type</th><th>Proposed Value</th></tr><tr><td>5</td><td>18 Code</td><td>94</td></tr><tr><td>5</td><td>13 Code with total price</td><td>2</td></tr><tr><td>5</td><td>13 Code with weight</td><td>7</td></tr><tr><td>6</td><td>18 Code</td><td>30</td></tr><tr><td>6</td><td>13 Code with total price</td><td>22</td></tr><tr><td>6</td><td>13 Code with weight</td><td>27</td></tr></table>	Item No. Digit	Barcode Type	Proposed Value	5	18 Code	94	5	13 Code with total price	2	5	13 Code with weight	7	6	18 Code	30	6	13 Code with total price	22	6	13 Code with weight	27
Item No. Digit	Barcode Type	Proposed Value																						
5	18 Code	94																						
5	13 Code with total price	2																						
5	13 Code with weight	7																						
6	18 Code	30																						
6	13 Code with total price	22																						
6	13 Code with weight	27																						
<u>BarcodeType2</u>	Int	0..255	Barcode type 2. 0-149: built-in; 150-255: user-defined																					

Label1ID	Int	0..32	Label 1 Number. 0: Do not print
Label2ID	Int	0..32	Label 2 Number. 0: Do not print
ProducedDate	Str		Production date. According to system time format, e.g.: YYYY/MM/DD hh:mm:ss
FreshnessDate	Int	0..999	Fresh days (hours). The calculation method of freshness period is detailed in [FreshnessDateFrom] definition.
<u>ValidDate</u>	Int	0..999	Valid day. The calculation method of valid days is detailed in [ValidDateFrom] definition.
PackageType	Int	0..4	Packaging type. 0: Normal; 1: Fixed weight; 2:Fixed price; 3:Fixed weight and price; 4: Gift basket
PackageWeight	Float		Package weight, or limit sales weight
PackagePrice	Float		Package price
PackageRange	Int	0..99	Package error range. 0-99%
PackageDays	Int	0..999	Package days. The calculation method of package days is detailed in [PackageDateFrom] definition.
PackageHours	Int	0..99	Package hours. The calculation method of package days is detailed in [PackageDateFrom] definition.
DiscountID	Int	0..99	Discount table ID. 0: No discount; 1-99: Associate the ID in Discount table Choose between DiscountID and DiscountRate
DiscountRate	Float	0..100	Discount Rate; 90.05%-->9005
TareID	Int	0..99	Tare. 0: Invalid; 1-99 Associate the ID of Tare table Choose between TareID and TareValue
TareValue	Float		Tare value.
LimitPrice	Float		Highest unit price (no price ceiling if value=0)
<u>Flag1</u>	Byte		<u>If not familiar with this part, please use proposed value 60</u> <u>(Hexadecimal:0x3C, Binary:00111100)</u> Bit0: Allow to change price. 0: Yes; 1: No Bit1: Fresh day, valid day, sales day or hours calculated by (Day/Hour). 0: by day; 1: by hour Bit2: Print package date. 0: No; 1: Yes Bit3: Print freshness date. 0: No; 1: Yes Bit4: Print valid days. 0: No; 1: Yes Bit5: Print Note3. 0: No; 1: Yes Bit6: N/A, the default is 0 Bit7: N/A, the default is 0
<u>Flag2</u>	Byte		<u>If not familiar with this part, please use proposed value 240</u> <u>(Hexadecimal: 0xF0, Binary: 11110000)</u> Bit0: Print PLU text 1. 0: No; 1: Yes Bit1: Print PLU text 2. 0: No; 1: Yes Bit2: Print PLU text 3. 0: No; 1: Yes Bit3: Print PLU text 4. 0: No; 1: Yes Bit4: Print nutrition information 0: No; 1: Yes Bit5: Print Note 1. 0: No; 1: Yes



			Bit6: Print Note 2. 0: No; 1: Yes Bit7: Print production date. 0: No; 1: Yes
Flag3	Byte		Bit0: Traceability item. 0: No; 1: Yes Bit1: Traceability information sequence. 0: first-in, first-out; 1: last-in, first-out Bit2: Ceiling price. 0: No limit; 1: Do not exceed highest unit price Bit3: N/A, the default is 0 Bit4: Print PLU text 5. 0: No; 1: Yes Bit5: Print PLU text 6. 0: No; 1: Yes Bit6: Print PLU text 7. 0: No; 1: Yes Bit7: Print PLU text 8. 0: No; 1: Yes
ProducedDataRule	Int		Production date definition rule 0: Production date = System date 1: Production date = [ProducedDate]
FreshnessDateFrom	Int		Freshness date base time 0: Freshness date = System date + [FreshnessDate] 1: Freshness date = Production date + [FreshnessDate] 2: Freshness date = Package date + [FreshnessDate]
ValidDateFrom	Int		Valid date base time 0: Valid date = System date + [ValidDate] 1: Valid date = Production date + [ValidDate] 2: Valid date = Package date + [ValidDate]
PackageDateFrom	Int		Package date base time 0: Package date = System date + [PackageDays] + [PackageHours] 1: Package date = Production date + [PackageDays] + [PackageHours]
Message1	Int		Quoted information number
IceValue	Int		Ice-containing ratio

### Description of the Flag field

Flag field is typically a set of switch flags. One byte has 8 bits which named bit0~bit7. Each bit can correspond to a switch flags. Therefore, one flag field can contain up to 8 flags. The value of the flag field can be calculated by following method:

$$\text{Flag field value} = \text{bit0} * 2^0 + \text{bit1} * 2^1 + \text{bit2} * 2^2 + \text{bit3} * 2^3 + \text{bit4} * 2^4 + \text{bit5} * 2^5 + \text{bit6} * 2^6 + \text{bit7} * 2^7$$

Thereinto, each of the bits has only value 0 and 1.

^ stands for exponentiation

## 5.4.2 Note File (Note1/2/3/4)

- Each PLU can have 4 notes, each of which can hold up to 1000 characters.
- Note is downloaded via a separate command and associated with the PLU LFCODE by PLUID.

### Note1

Field name	Type	Size	Description
<b>PLUID</b>	Int		LFCODE
Value	Str	1000	Material information

Note2			
Field name	Type	Size	Description
<b>PLUID</b>	Int		LFCODE
Value	Str	1000	Additional information

Note3			
Field name	Type	Size	Description
<b>PLUID</b>	Int		LFCODE
Value	Str	1000	Additional information

Note4			
Field name	Type	Size	Description
<b>PLUID</b>	Int		LFCODE
Value	Str	1000	Additional information

### 5.4.3 Department File (Department)

The concept of the department corresponds to the concept of Department/Category Group/Large Category in the background of the Mis.

Department			
Field name	Type	Size	Description
<b>ID</b>	Int		Department ID
Name	Str	30	Name

There is a department field in the PLU file, which is only used to form the barcode, and has no relationship with the department here.

### 5.4.4 Group File (Group)

The concept of grouping corresponds to the concept of Category/Small Category in the background of the Mis, and its superior is the department.

Associated with the department by DepartmentID, that is, each group belongs to a certain department.

Group			
Field name	Type	Size	Description
<b>ID</b>	Int		Group ID
Name	Str	30	Name
DepartmentID	Int		Department ID
ShowPosition	Int		Show grouping position

There is a column in the PLU that associates the PLU with the packet, that is, each PLU belongs to a

certain group.

## 5.4.5 Hotkey File (HotKey)

Hotkey files are available in two formats:

Label scale (\*.key):

HotKey			
Field name	Type	Size	Description
ButtonIndex	Int		Hotkey number, starting from 1, the total number is the number of hotkeys (e.g.: LS2 is 224)
ButtonValue	Int		Hotkey value, input PLU LFCODE

Touch scale (\*.tsk):

HotKey			
Field name	Type	Size	Description
PLUID	Int		LFCODE
GroupID	Int		Group ID
IsShowTouch1	Int		Display or not
Position1	Int		Show position
IsShowTouch2	Int		Reserve
Position2	Int		Reserve

## 5.4.6 Discount Schedule File (Discount)

Discount			
Field name	Type	Size	Description
ID	Int		Discount schedule No.
Name	Str	50	Discount schedule name, this field may be used in Touch Scale
ItemIndex	int	0..2	Discount schedule subitem No.
Mode	Int	1..99	Discount mode
DiscountRate	Float		Discount rate
DiscountAmt	Float		Discount value
BeginDateTime	Datetime		Start time
EndDateTime	Datetime		End time
MinWeight	Float		Min. weight
MaxWeight	Float		Max. weight
MinTotalPrice	Float		Min. total price
MaxTotalPrice	Float		Max. total price
Flag1	Byte		Bit0: N/A, the default is 0 Bit1: N/A, the default is 0 Bit2: N/A, the default is 0

			Bit3: Time range mode. 0: within that period of each day; 1: Start time to End time Bit4: Enable the total price range. 0: No; 1: Yes Bit5: Enable the weight range. 0: No; 1: Yes Bit6: Enable the time range. 0: No; 1: Yes Bit7: Enable the schedule subitem. 0: No; 1: Yes
--	--	--	--

## 5.4.7 Advertising Information File (AdverisementInfo)

Used to show advertising information on the display, which is only one line of data.

AdverisementInfo			
Field name	Type	Size	Description
Value	Str	256	Advertising information

## 5.4.8 Label File (Label)

The label file is a binary file. A set of label file is divided into 3 files which are respectively:

Label shading file (LabelMap), file type is \*.LM

Label format file (LabelFormat), file type is \*.LF

Label design file (LabelDesign), file type is \*.TBL

When saved as \*.TBL in the Label Designer, it will auto-save \*.LM and \*.LF files with the same name. \*.TBZ format can also be saved under Label Designer. It's actually packing the above three format files into a ZIP file.

The three files need to be downloaded to label scale simultaneously, while only label design file (LabelDesign) need to be uploaded when upload.

Aclás barcode label scale supports multiple labels, which are distinguished by the label number. Generally, No.1 label can meet the ordinary demand.

**Attention: The label number used in the interface for downloading the label = Label number -1, that is, the label number of the first label is 0, the label number of the second label is 1, and so on.**

How to transmit label sequence number?

➤ Dynamic link library interface

The label sequence number is represented by the high byte of the operation type (ProcType).

E.g.: the value of ProcType for downloading the No. 8 label is:

ProcType=7\*256+0;

E.g.: the value of ProcType for uploading the No. 5 label is:

ProcType=4\*256+1;

➤ Command line interface

Specify the label number by the -d parameter. If the label number is 0, this parameter can be omitted.

See 3.2 Command Line Usage Examples

### 5.4.9 Sale Record (SaleRecord)

SaleRecord			
Field name	Type	Size	Description
TypeID	Str	1	When the field value is “D”, it means that this piece of data is a label sales PLU; When the field value is “E”, it means that the piece of data is a single PLU sold by the receipt.
IP	Int		Scale IP address
ReceiptNO	Int		When the field value is “E”, this value is the serial number of the receipt.
PLUID	Int		PLU LFCODE
UnitPrice	Int		PLU unit price
TotalPrice	Int		PLU total price
DiscountAmt	Int		PLU discount amount
UnitID	Int		Weighing unit number, the meaning of the value is: 0-50g; 1-g; 2-10g; 3-100g; 4-kg; 5-oz; 6-lb; 7-500g; 8-600g; 9-pcs(g); 10-pcs(kg); 11-pcs(oz); 12-pcs(lb)
Weight	Int		When sold by PCS, it means the value of 1 PCS is 1000; if not sold by PCS, the value refers to the actual weighing weight.
SaleTime	Datetime		Sales date and time
OnlineTime	Datetime		Date and time when the sales data was last cleared
Clerk	Str	8	Clerk No.
Tracecode	Str	25	Traceability code
Pcode	Int		Traceability code batch code
LinkF	Int		0x10: Networking with price change 0x11: Networking without price change 0x01: Non-networked with price change 0x00: Non-networked without price change

### 5.4.10 Traceability File (Trace)

The traceability file contains three interrelated file, respectively:

Traceability configuration file (TraceStatus), file name TraceStatus.txt is recommended

Traceability index file (TraceMap), file name TraceMap.txt is recommended

Traceability data file (TraceData), file name TraceData.txt is recommended

- Traceability data is time-sensitive, so each download is a full coverage download.
- Many-to-many relationship exists between LFCODE and traceability code, that is, one LFCODE

may have multiple traceability codes, and a traceability code may be shared by multiple LFCODE.

- Traceability configuration file (TraceStatus) has only one line of data.
- For simple traceability code application scenarios, such as an LFCODE with only one traceability code, can also be directly traced in the note (Note 1/2/3/4) to achieve.

TraceStatus			
Field name	Type	Size	Description
MapCount	Int		Corresponding to the number of data in TraceMap.txt
DataCount	Int		Corresponding to the number of data in TraceData.txt
Http	Str	128	Traceability code URL

Example

MapCount      DataCount      Http  
 2      2      http://www.xxxtrace.com/search.html?code=

TraceMap			
Field name	Type	Size	Description
PLUID	Int		LFCODE
Index	Int		Index No. (that is, the traceability code corresponding to this LFCODE is the line number in the TraceData.txt file, and the line number is counted from 0)

Example

PluID      Index  
 601      0  
 602      1

TraceData			
Field name	Type	Size	Description
GroupID	Int		Reserve, input 0
TraceCode	Str	30	Traceability code

Example

GroupID      TraceCode  
 0      41110321000000013407  
 0      41110321000000013507

## 5.4.11 Information (Message1)

- Different with Note, the information is maintained separately, that is, not corresponding to PLU one by one.
- There is a Message1 field in the PLU, which fills in the information number (ID) here, indicating that the PLU references this information.

Message1			
Field name	Type	Size	Description

ID	Int		Information number
Val	Str	245	Information content

## Appendix: Barcode Coding Comparison Table

Barcode type	Dept.	Item No.	Total price	Weight	Checksum
00~09: Ean13 Code, first two codes print department code:					
00	DD(2)	IIIIIIII(10)	X	X	C
01	DD(2)	IIIII(6)	PPPP(4)	X	C
02	DD(2)	IIII(5)	PPPPP(5)	X	C
03	DD(2)	III(4)	PPPPPP(6)	X	C
04	DD(2)	II(3)	PPPPPPP(7)	X	C
05	DD(2)	IIIII(6)	X	W.WWW(4)	C
06	DD(2)	IIIII(6)	X	WW.WW(4)	C
07	DD(2)	IIII(5)	X	WW.WWW(5)	C
08	DD(2)	IIII(5)	X	WWWW.W(5)	C
09	DD(2)	IIII(5)	X	WWWWW(5)	C
10~19: Ean13 Code, first two codes print fixed code:					
10	20(2)	IIIIIIII(10)	X	X	C
11	21(2)	IIIII(6)	PPPP(4)	X	C
12	22(2)	IIII(5)	PPPPP(5)	X	C
13	23(2)	III(4)	PPPPPP(6)	X	C
14	24(2)	II(3)	PPPPPPP(7)	X	C
15	25(2)	IIIII(6)	X	W.WWW(4)	C
16	26(2)	IIIII(6)	X	WW.WW(4)	C
17	27(2)	IIII(5)	X	WW.WWW(5)	C
18	28(2)	IIII(5)	X	WWWW.W(5)	C
19	29(2)	IIII(5)	X	WWWWW(5)	C
20: Do not print barcodes					
21~29: Ean13 Code, first code print department code:					
21	D(1)	IIIIII(7)	PPPP(4)	X	C
22	D(1)	IIIII(6)	PPPPP(5)	X	C
23	D(1)	IIII(5)	PPPPPP(6)	X	C
24	D(1)	III(4)	PPPPPPP(7)	X	C
25	D(1)	IIIIII(7)	X	W.WWW(4)	C
26	D(1)	IIIII(7)	X	WW.WW(4)	C
27	D(1)	IIIII(6)	X	WW.WWW(5)	C
28	D(1)	IIIII(6)	X	WWWW.W(5)	C
29	D(1)	IIIII(6)	X	WWWWW(5)	C

30~35: 18 Code, first code print department code:					
30&33	D(1)	IIIII(6)	PPPPP(5)	WW.WWW(5)	C
31&34	D(1)	IIIII(6)	PPPPP(5)	WWWW.W(5)	C
32&35	D(1)	IIIII(6)	PPPPP(5)	WWWWW(5)	C

Barcode type	Dept.	Item No.	Unit price	Weight	Checksum
40~45: 18 Code, first code print department code:					
40&43	D(1)	IIIII(6)	UUUUU(5)	WW. WWW(5)	C
41&44	D(1)	IIIII(6)	UUUUU (5)	WWWW.W(5)	C
42&45	D(1)	IIIII(6)	UUUUU (5)	WWWWW(5)	C

Barcode type	Dept.	Item No.	Total price	Weight	Checksum
46: 18 Code					
46	DD(2)	IIIII(6)	PPPPP(5)	WWWWW(5)	X
50~55: 8 Code					
50	X	IIIII(7)	X	X	C
51	D(1)	IIIII(6)	X	X	C
52	DD(2)	IIII(5)	X	X	C
53	X	IIIIII(8)	X	X	X
54	D(1)	IIIII(7)	X	X	X
55	DD(2)	IIIII(6)	X	X	X

Barcode type	Dept.	LFCode	Batch No.	Discount	Weight
36-38: 18 Code, first code print department code, for fresh batch management dedicated					
36	D(1)	LLLLL(6)	BBBB(4)	RR(2)	WW.WWW(5)
37	D(1)	LLLLL(6)	BBBB(4)	RR(2)	WWWW.W(5)
38	D(1)	LLLLL(6)	BBBB(4)	RR(2)	WWWWW(5)

Barcode type	Dept.	LFCode	Batch No.	Discount	Weight	Checksum
66~68: 18 Code, for fresh batch management dedicated						
66	D(1)	LLLLL(5)	BBBB(4)	RR(2)	WW.WWW(5)	C
67	D(1)	LLLLL(5)	BBBB(4)	RR(2)	WWWW.W(5)	C
68	D(1)	LLLLL(5)	BBBB(4)	RR(2)	WWWWW(5)	C

Barcode type	Dept.	Item No.	Total price or unit price	Weight	Checksum
60~65: ISBN Code, one kind of 18 Code					
60	D(1)	IIIII(6)	PPPPP(5)	WW.WWW(5)	C
61	D(1)	IIIII(6)	PPPPP(5)	WWWW.W(5)	C
62	D(1)	IIIII(6)	PPPPP(5)	WWWWW(5)	C
63	D(1)	IIIII(6)	UUU.UU(5)	WW.WWW(5)	C
64	D(1)	IIIII(6)	UUU.UU(5)	WWWW.W(5)	C




65	D(1)	IIIII(6)	UUU.UU(5)	WWWWW(5)	C
----	------	----------	-----------	----------	---


Label type	Dept.	Item No.	Qty. (Weight)	Total price or unit price	Checksum
79~85, 87: 18 Code					
79	DD	IIIII(6)	WW.WWW	PPPPP	X
80	D	IIIII(6)	WW.WWW	PPPPP	C
81	D	IIIII(6)	WWWW.W	PPPPP	C
82	D	IIIII(6)	WWWWW	PPPPP	C
83	D	IIIII(6)	WW.WWW	UUU.UU	C
84	D	IIIII(6)	WWWW.W	UUU.UU	C
85	D	IIIII(6)	WWWWW	UUU.UU	C
87	DD	IIII(5)	WW.WWW	PPPPP	C


Barcode type	Dept.	Item No.	Total price	Qty. (Weight)	Unit price	Checksum
90~95: 18 Code, first two code print department code:						
90	DD(2)	IIIII(6)		WW.WWW	UU.UU(4)	C
91	DD(2)	IIIII(6)		WWWW.W	UU.UU(4)	C
92	DD(2)	IIIII(6)		WWWWW	UU.UU(4)	C
93	D(1)	IIIII(6)	PPPPP(5)		UUU.UU(5)	C
94	DD(2)	IIII(5)	PPPPP(5)	WW.WWW		C
95	DD(2)	IIIII(6)		WWWWW	UU.UU(4)	C
Barcode type	Dept.	Item No.	Qty. (Weight)		Total price	Unit price
96	DD(2)	IIIII(6)	W.WWW	PPPPP(5)		C
Barcode type	Dept.	Item No.	Total price	Qty. (Weight)	Unit price	Checksum
97	DD(2)	IIII(4)	PPPPPP(6)	WW.WWW		C
Barcode type	Dept.	Item No.	Unit price	Qty. (Weight)	Total price	Checksum
98	DD(2)	IIII(5)	UUUUU(5)	WW.WWW		C

The meaning of letters in the table show as below:


C: Checksum D: Department code 2: Fixed number "2" I: Item No. L: LFCODE  
P: Total price U: Unit price R: Discount W: Weight X: None


 Ean13 Code = DEPARTMENT+ CODE+ [TOTAL PRICE] + [WEIGHT] +C Thereinto, with [ ] item means that there is no such item in some encoding methods.


 For the Ean13 code, if the price barcode is used, the barcode type is usually 2 (or 22); if the weight barcode is used, the barcode type is usually 7 (or 27).


 CHECKSUM is automatically calculated by the Label Scale, and the user does not need to input this item in the PLU manager.


 30-32 format and 33-35 format are different in the calculate method of CHECKSUM.

 The difference between 0-45 and 30-35 formats is that the price is the total price in 30-35, and the price is the unit price in 40-45.

 36-38, 66~68 is the barcode type that can be used for fresh batch management. The commodity batch number of the product is filled in the item number field, and the item number is filled in the fresh code field.

 60-65 is the ISBN Code.

 Computing method of EAN13 code check code Z:  $(\text{sum of even digits} * 3) + \text{sum of odd digits} + Z = \text{multiple of 10}$

 Two methods for calculating the 18 Code check code Z:

Method 1: Same as 13 Code (Barcode type: 30-32, 40-42)

Method 2:  $(\text{sum of odd digits} * 3) + \text{sum of even digits} + Z = \text{multiple of 10}$   
(Barcode type: 33-35, 43-45)