Instruction Manual

AlprSDK Interface

Date: 09-04-2020

Doc Version: V1.0.1.25

English

Thank you for choosing our product. Please read the instructions carefully before operation. Follow these instructions to ensure that the

For further details, please visit our Company's website www.zkteco.com.

ZKTeco Headquarters

Address ZKTeco Industrial Park, No. 26, 188 Industrial Road,

Tangxia Town, Dongguan, China.

Phone +86 769 - 82109991

Fax +86 755 - 89602394

For business related queries, please write to us at: sales@zkteco.com.

To know more about our global branches, visit www.zkteco.com.

About the Manual

This manual introduces the operations of [Keywords]product.

All figures displayed are for illustration purposes only. Figures in this manual may not be exactly consistent with the actual products.

Document Conventions

Conventions used in this manual are listed below:

GUI Conventions

For Software	
Convention	**Description**
Bold font	Used to identify software interface names e.g. **OK**, **Confirm**, **Cancel**
>	Multi-level menus are separated by these brackets. For example, File > Create > Folder.
For Device	
Convention	**Description**
< >	Button or key names for devices. For example, press <OK>
[ ]	Window names, menu items, data table, and field names are inside square brackets. For example, pop up the [New User] window
/	Multi-level menus are separated by forwarding slashes. For example, [File/Create/Folder].

Symbols

Convention	Description
	This implies about the notice or pays attention to, in the manual
	The general information which helps in performing the operations faster

	The information which is significant
	Care taken to avoid danger or mistakes
	The statement or event that warns of something or that serves as a cautionary example.

Table of Contents

Table of Contents 5

1 Brief introduction 13

1.1 About Our Product 13
1.2 Features 13
1.3 Programming Model 13
1.4 About the value of Handle 16

2 AlprSDK Architecture 17

2.1 Supported Platform 17

3 Interface definition 18

3.1 Initialization SDK 18
3.2 Search the Device 18
3.3 Initialization of the Window Handle 19
3.4 Device Connection 19
3.4.1 Set Connection Timeout 19
3.4.2 Connect to Device 19
3.5 Callback Functions 20
3.5.1 License Plate Recognition 20
3.5.2 Capture Image 21
3.5.3 Device Status Callback 21
3.5.4 Offline Record Callback 21
3.5.5 Wiegand Card Data 22
3.6 Vehicle Feature Recognition 22
3.6.1 Vehicle Characteristics 22

3.6.2 Image Conversion····· 23
3.6.3 Identify Vehicle Color, Type and Logo····· 23
3.6.4 Delete Color Image····· 23
3.7 Video Functions····· 24
3.7.1 Video Initialization····· 24
3.7.2 Video Configurations····· 24
3.7.3 Video Size····· 24
3.7.4 Video Graphing Functions····· 25
3.7.5 Video Exit Functions····· 25
3.7.6 Exit Connections····· 25
3.8 Release Handle····· 26
3.9 Release SDK····· 26
3.10 Other Function Definition····· 26
3.10.1 Area and Virtual Coil····· 26
3.10.2 Device Functions····· 27
3.10.3 Device Control Operations····· 28
3.10.4 Information Parameter Settings····· 30
3.10.5 Camera Recognition Parameters····· 30
3.10.6 License Plate····· 32
3.10.7 Data Transmission····· 33
3.10.8 Voice Intercom····· 34
3.10.9 P2P Service····· 35
3.10.10 Manual identification····· 36
3.11 Parking Camera Interface····· 36

4 AlprSDK Functions 38

4.1 AlprSDK_Startup 38
4.2 AlprSDK_StartupWithPath 39
4.3 AlprSDK_SearchAllCameras 40
4.4 CallbackServerFind 41
4.5 AlprSDK_InitHandle 42
4.6 AlprSDK_ConnectDev 43
4.7 AlprSDK_SetConnectTimeout 44
4.8 AlprSDK_SendHeartBeat 46
4.9 AlprSDK_CreateRecogAllInfoTask 46
4.10 AlprSDK_ClearRecogAllInfoTask 47
4.11 RecogAllInfoCallback 48
4.12 AlprSDK_CaptureJpg 49
4.13 DeviceCaptureCallback 50
4.14 AlprSDK_CreateCaptureJpgCallback 50
4.15 AlprSDK_ClearCaptureJpgCallback 51
4.16 DevStatusCallback 52
4.17 AlprSDK_CreateDevStatusCallback 53
4.18 AlprSDK_ClearDevStatusCallback 54
4.19 AlprSDK_CreateRecordCallback 55
4.20 AlprSDK_ClearRecordCallback 57
4.21 RecordCallback 57
4.22 AlprSDK_CreateWiegandDataCallback 58
4.23 AlprSDK_ClearWiegandDataCallback 59
4.24 WiegandDataCallback 60
4.25 AlprSDK_StartVideo 61
4.26 AlprSDK_SetVideoConfig 62
4.27 AlprSDK_GetVideoConfig 63
4.28 AlprSDK_GetVideoSize 64
4.29 AlprSDK_RegCBOnDraw 65
4.30 HHCBOnDraw 67
4.31 AlprSDK_StopVideo 68
4.32 AlprSDK_DisConnectDev 69
4.33 AlprSDK_UnInitHandle 70
4.34 AlprSDK_Cleanup 70
4.35 AlprSDK_SetRoiEx 71
4.36 AlprSDK_GetRoiEx 73
4.37 AlprSDK_SetVirtualCoil 74

4.38 AlprSDK_GetVirtualCoil....75
4.39 AlprSDK_GetDeviceFunList....77
4.40 AlprSDK_GetDevLicenseInfo....78
4.41 AlprSDK_GetDevVerInfo....79
4.42 AlprSDK_SetSysTime....80
4.43 AlprSDK_OpenGate....81
4.44 AlprSDK_CloseGate....82
4.45 AlprSDK_OpenGateByJpgID....83
4.46 AlprSDK_RebootDevice....84
4.47 AlprSDK_SetGateCfg....85
4.48 AlprSDK_GetGateCfg....86
4.49 AlprSDK_OpenAuxOut....87
4.50 AlprSDK_SetAlprCfg....89
4.51 AlprSDK_GetAlprCfg....90
4.52 AlprSDK_SetOfflineParam....91
4.53 AlprSDK_GetOfflineParam....92
4.54 AlprSDK_SetOSD....93
4.55 AlprSDK_GetOSD....94
4.56 AlprSDK_SetBaseParam....95
4.57 AlprSDK_GetBaseParam....96
4.58 AlprSDK_OperateSDCard....97
4.59 AlprSDK_ResetFactory....98
4.60 AlprSDK_AddPlateListExt....99
4.61 AlprSDK_DelPlateListExt....101
4.62 AlprSDK_ClearPlateListExt....102
4.63 AlprSDK_ExportPlateListExt....103
4.64 AlprSDK_GetDeviceDataCount....104
4.65 AlprSDK_GetDeviceData....105
4.66 AlprSDK_SetDeviceData....107
4.67 AlprSDK_ClearDeviceData....108
4.68 AlprSDK_CommTransparentTransfer....109
4.69 AlprSDK_CommTransparentTransferByJpgID....110
4.70 AlprSDK_CarInfoRecogInit....112
4.71 AlprSDK_SwitchJpgToRGB....112
4.72 AlprSDK_CreateColorImageI....113
4.73 AlprSDK_SmartALPR_RecogColor....114
4.74 AlprSDK_SmartALPR_RecogCarType....115
4.75 AlprSDK_SmartALPR_RecogLogo....117
4.76 AlprSDK_DestroyColorImageI....118
4.77 AlprSDK_Trans2Voice....118

4.78 AlprSDK_Trans2Screen····120
4.79 AlprSDK_StartVoice····121
4.80 fVoiceDataCallBackV2····123
4.81 AlprSDK_InputAudio····124
4.82 AlprSDK_PlaySound····125
4.83 AlprSDK_SendVoiceData····126
4.84 AlprSDK_StopVoice····127
4.85 AlprSDK_SetP2PServerAddress····128
4.86 AlprSDK_EnableP2PReconnect····129
4.87 AlprSDK_DisableP2PReconnect····130
4.88 AlprSDK_SetNetworkParam····131
4.89 AlprSDK_GetNetworkParam····132
4.90 AlprSDK_SetVideoParam····134
4.91 AlprSDK_GetVideoParam····135
4.92 AlprSDK_GetDeviceParam····136
4.93 CarSpaceStateCallback····138
4.94 AlprSDK_CreateCarSpaceTask····138
4.95 AlprSDK_ClearCarSpaceTask····139
4.96 AlprSDK_SetCarSpaceConfig····140
4.97 AlprSDK_GetCarSpaceConfig····141
4.98 AlprSDK_RefreshCarSpaceState····143
4.99 AlprSDK_ModifyCameraNetInfo····错误！未定义书签。
4.100 AlprSDK_RecogByManual····143

5 Structure definition····**145**

5.1 RECOG_ALL_INFO····145
5.2 LICENSE_PLATE····145
5.3 PLATERESULT····146
5.4 DEVINFO····149
5.5 CAR_INFO····150
5.6 VEHICLEFEATURE····151
5.7 TWIEGANDDATA····152
5.8 CAR_SPACE_INFO····152
5.9 JPG_BYTES····153
5.10 CAR_SPACES····154
5.11 CARSPACEINFO····155
5.12 TCARSPACECONFIG····157
5.13 TALPRDEVNETINFO····158
5.14 ALPR_VIDEO_CONFIG····159

5.15 XRECT 160
5.16 TDEVLICENSE 161
5.17 ALPR_CONFIG 162
5.18 DEV_VER_INFO 164
5.19 TSYS_TIME 165
5.20 GATE_CONFIG 166
5.21 TOFFINEPARAM 167
5.22 OSD_CTRL 170
5.23 TBASE_PARAM 171
5.24 TSDCARDCAPACITY 173
5.25 PLATE_INFO_EXT 173
5.26 CCOLOR_IMAGE_I 174
5.27 CIMAGE_DATA_FORMAT 175
5.28 COLORRESULT 176
5.29 LOGORESULT 177
5.30 NETWORKPARAM 178
5.31 VIDEOPARAM 179
5.32 DEVICEPARAM 180
5.33 WEBAPIVER 181
5.34 DEVSTATUS 182

6 Enumeration definition 185

6.1 Macro Function Description 185
6.2 Type Definition Description 185
6.3 Enumeration Type Description 187
6.3.1 Country Code Description 187
6.3.2 Data Table Definition 197
6.3.3 Mode of Payment for Temporary Car 198
6.3.4 License Plate Recognition Strategy Definition 198
6.3.5 License Plate recognition Mode Definition 198
6.3.6 License Plate Recognition Trigger Mode Definition 199
6.3.7 Startup Method Definition 199

6.3.8 Ground State Definition····199

6.3.9 Gate Operation Status····200

6.3.10 Switch Status····200

6.3.11 ISP Switching Method Definition····201

6.3.12 Country Code····201

6.3.13 Offline Mode Definition····208

6.3.14 Online and Offline Definition····208

6.3.15 Gate Opening Method Definition····208

6.3.16 Event Type Definition····209

6.3.17 Entrance and Exit Definition····210

6.3.18 SD Card Status Information Definition····210

6.3.19 Country Code Definition for License Plate Recognition····211

6.3.20 Client Type Definition····212

6.3.21 List of Supported Features····212

6.3.22 Identified License Plate Color Definition····213

6.3.23 License Plate List Type Definition····213

6.3.24 Gate Work Definition····214

6.3.25 Image Upload Mode····214

6.3.26 Camera Recognition Computation····215

6.3.27 Scroll Mode Description····215

6.3.28 Display Color Definition····216

6.3.29 Car Type Definition····216

6.3.30 Vehicle Color Definition····220

6.3.31 Vehicle Type Definition····220

6.3.32 Camera Device Type Definition···· 221

6.3.33 Parking Status···· 221

6.3.34 Screen type definition····222

6.3.35 Other Record Definition····222

7 Appendixes····227

Appendix 1 - Error Code····227

Appendix 2 – RS485 Communication Protocol····229

Communication Data Format Definition····229

Communication Procedure····230

Communication Command···· 231

Communication Command – Error Code···· 235

Appendix 3 – Voice Segment Description····236

Voice Segment Description - Foreign····236

1 Brief introduction

1.1 About Our Product

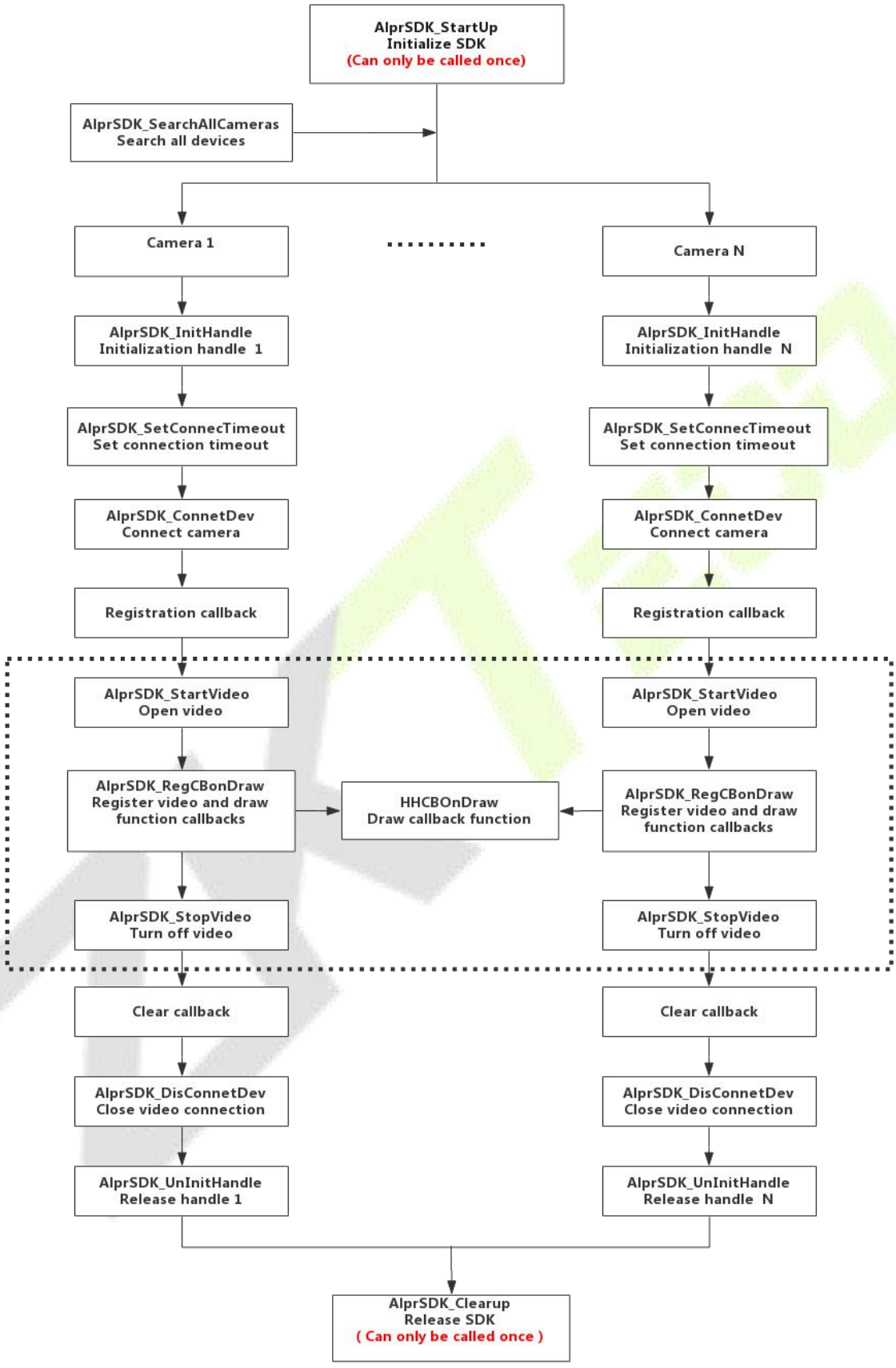
From the perspective of a developer, the key design objective of **AlprSDK** is compatibility and the ease of execution.

This Reference Manual contains the product development documentation for developers that describes the functions provided by the SDK and its related usage, which eases the development environment. In the following sections all the required information on how to integrate **AlprSDK** into a third-party application is provided.

1.2 Features

- **License Plate:** Gets the license plate, license plate image and the time to capture images by registering the license plate callback function.
- **Device Status:** Obtains the status of the online device, auxiliary camera, and the switch by registering the device status callback function.
- **Record**: Gets the real-time records by registering the record callback function.
- **Manual Capture**: Register manual capture callback. Call start capture function, can trigger callback.
- **Data management:** Support black and white list data management.
- **Remote control:** Supports remote opening function.
- **Display control:** Supports for transparent transmission: control of language and display

1.3 Programming Model



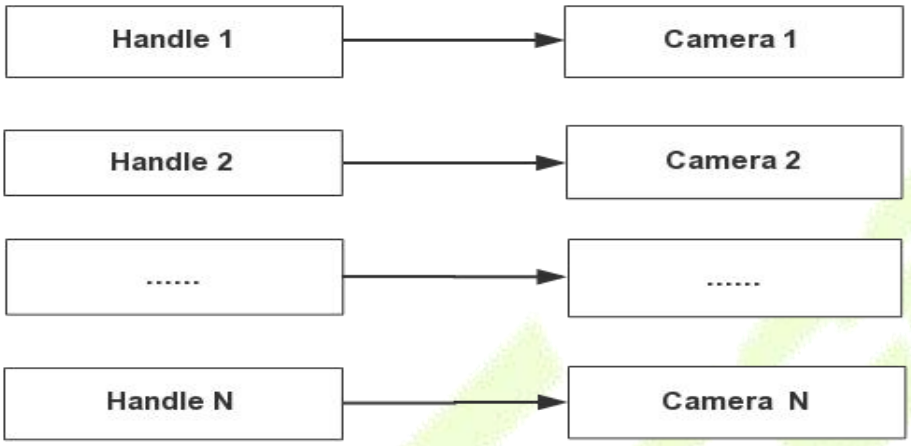
Note:

- License plate camera use: AlprSDK_CreateRecogAllInfoTask (the license plate callback function).
- Parking camera use: AlprSDK_CreateCarSpaceTask (parking callback function).
- Video-related interfaces are not supported on linux systems.

Callback function considerations :

- SDK interface functions cannot be called in this type of function.
- The total function call time cannot exceed 100ms.
- There cannot be a sleep function.

1.4 About the value of Handle



Notes :

- The software can customize the value of Assign handle, one handle corresponds to one device.
- If multiple videos need to be played at the same time, handle has a value ranging from 0 to 18. if no video needs to be played, handle has a value ranging from 0 to 1024
- This function is called in a multithreaded concurrent environment. if it is the same Handle id, the upper layer needs lock protection.

2 AlprSDK Architecture

2.1 Supported Platform

Operating System	SDK Package Path	Demo Package Path
Windows 32- or 64-bit system	license plate business SDK secondary development package / dynamic library / Windows /c#/dll	c# : license plate business SDK secondary development package / SDK/windows**/c#** **c++ :** license plate business SDK secondary development package / SDK/windows**/c++**
Linux 32- or 64-bit system	license plate business SDK secondary development package / SDK/ linux/dll	license plate business SDK secondary development package / SDK/ linux/c

Note:

- Windows provides 32-bit SDK and 64-bit SDK, which is compatible with 64-bit systems. And Video preview interface is not supported under 64-bit SDK。And the demo package provides C++ and C# source code.
- The SDK of Linux does not contain a video preview interface. But using the BS service architecture, the video preview plug-in can be called into the browser.
- This SDK document describes the commonly used function structure descriptions in detail. For other structure definitions, please refer to AlprSDK.h.

3 Interface definition

3.1 Initialization SDK

Under windows system :

The following functions initializes the SDK and starts up the Camera .Called before all interface calls and can only be called once.

Function	Description
AlprSDK_Startup	This function initializes the SDK Device in the windows system

Under Linux system :

The following functions initializes the SDK and starts up the Camera .Called before all interface calls and can only be called once.

Function	Description
AlprSDK_StartupWithPath	This function initializes the SDK Device in the linux system

3.2 Search the Device

This function facilitates in searching all the available camera Devices.

Function	Description
AlprSDK_SearchAllCameras	This interface is used to search for devices

3.3 Initialization of the Window Handle

Generally simplified as **hWnd** is a unique identifier assigned to each created window by Windows. This function performs the task to initialize the Window handles based on the availability of the devices.

Function	Description
AlprSDK_InitHandle	This function initializes the SDK Device

3.4 Device Connection

The following functions initiates the basic connection operations.

3.4.1 Set Connection Timeout

This function set the amount of time a connection waits to time out by using the Connect Timeout or Connection Timeout keywords in the connection string.

Function	Description
AlprSDK_SetConnectTimeout	This function sets the connection timeout.

3.4.2 Connect to Device

This function obtains the available device information and connects all the searched devices.

Function	Description

AlprSDK_ConnectDev	This function connects the available camera devices

Send HeartBeat

This function triggers a periodic signal generated by the device to indicate normal operation.

Usually a heartbeat is sent between devices at a regular interval in the order of seconds.

Function	Description
AlprSDK_SendHeartBeat	This function sends the heartbeat signals to get the device connection status.

3.5 Callback Functions

A callback function is a function that is passed as an argument to another function, to be "called back" at a later time.

3.5.1 License Plate Recognition

This function processes the License Plate Recognition callback function settings.

Function	Description
AlprSDK_CreateRecogAllInfoTask	License plate callback function settings.
AlprSDK_ClearRecogAllInfoTask	This function processes to clear the License Plate Callback function.

RecogAllInfoCallback	License Plate Recognition Information Callback Function

3.5.2 Capture Image

This function triggers the Camera Device to capture the image directly and return it to the application layer through the DeviceCaptureCallback callback function.

Function	Description
AlprSDK_CaptureJpg	To capture the image.
DeviceCaptureCallback	Image data capture callback function
AlprSDK_CreateCaptureJpgCallback	This function gets the Device capture callback settings.
AlprSDK_ClearCaptureJpgCallback	Image capture callback functions clear.

3.5.3 Device Status Callback

This function notifies the status of the Device by using a function pointer.

Function	Description
DevStatusCallback	Device Status Callback function.
AlprSDK_CreateDevStatusCallback	Device Status Callback
AlprSDK_ClearDevStatusCallback	Device status callback function clear.

3.5.4 Offline Record Callback

This function sets the offline record events by calling **RecordCallback** function.

Function	Description
AlprSDK_CreateRecordCallback	This function sets the offline logging event callbacks.
AlprSDK_ClearRecordCallback	offline logging event callback functions clear.
RecordCallback	Event record callback function

3.5.5 Wiegand Card Data

This function gets the Wiegand card data information by calling the Wiegand card number callback function.

Function	Description
AlprSDK_CreateWiegandDataCallback	Wiegand card data callback.
AlprSDK_ClearWiegandDataCallback	Wiegand data callback functions clear.
WiegandDataCallback	Gets the Wigan card number callback function

3.6Vehicle Feature Recognition

3.6.1 Vehicle Characteristics

This function processes to initialize the characteristic features of the vehicle.

Function	Description
AlprSDK_CarInfoRecogInit	Initializing vehicle features

3.6.2 Image Conversion

This function processes to covert the captured image to RGB format.

Function	Description
AlprSDK_SwitchJpgToRGB	Jpg Image conversion RGB Pixel
AlprSDK_CreateColorImageI	Create Color Image

3.6.3 Identify Vehicle Color, Type and Logo

This function processes the device to identify the vehicle color, vehicle type and the brand logo of the vehicle.

Function	Description
AlprSDK_SmartALPR_RecogColor	Identify Color
AlprSDK_SmartALPR_RecogCarType	Identify Type
AlprSDK_SmartALPR_RecogLogo	Identify Logo

3.6.4 Delete Color Image

This function processes to delete a color image.

Function	Description
AlprSDK_DestroyColorImageI	Delete Color Image

3.7 Video Functions

The following functions initializes the video and sets the essential video configurations.

3.7.1 Video Initialization

This function contains the task of initializing the vide operations.

Function	Description
AlprSDK_StartVideo	This function initiates the video.

3.7.2 Video Configurations

This function sets the parameters and gets the video configuration settings.

Function	Description
AlprSDK_SetVideoConfig	This function sets the video configurations.
AlprSDK_GetVideoConfig	This function gets the video configurations.

3.7.3 Video Size

This function processes the size of the video.

Function	Description
AlprSDK_GetVideoSize	This function gets the size of the video.

3.7.4 Video Graphing Functions

Register Drawing

This function registers the Drawing Callback function.

Function	Description
AlprSDK_RegCBOnDraw	This function registers the Drawing Callback.

3.7.5 Video Exit Functions

Close the Video

This function processes to exit the video settings.

Function	Description
AlprSDK_StopVideo	This function is to turn off the video.

Clear Rectangle Graph

This function processes to clear the rectangular graphing drawn on the video.

Function	Description
AlprSDK_ClearRectOnVideo	This function clears the rectangle drawn on the video.

3.7.6 Exit Connections

The following functions exits all the connections.

Device Disconnection

This function processes to disconnect from the device.

Function	Description
AlprSDK_DisConnectDev	Disconnect the device connection.

3.8 Release Handle

This function processes to release the handles connected to the camera devices.

Function	Description
AlprSDK_UnInitHandle	Release handle.

3.9 Release SDK

This function releases all the related reserved sources and connection and exits the AlprSDK.

Function	Description
AlprSDK_Cleanup	Close AlprSDK and release related resources.

3.10 Other Function Definition

3.10.1 Area and Virtual Coil

Set and Get Identification Region

This function processes to set and get the identification region operations.

Function	Description
AlprSDK_SetRoiEx	Setting identification region.
AlprSDK_GetRoiEx	Obtaining the identification area expansion.

Set and Get Virtual Coil

This function processes to set and get the virtual coil area.

Function	Description
AlprSDK_SetVirtualCoil	Set virtual coil area.
AlprSDK_GetVirtualCoil	Get virtual coil area.

3.10.2 Device Functions

Other Supported Functions

This function processes the device other supported operations.

Function	Description
AlprSDK_GetDeviceFunList	Get the functionality supported by the device.

Device License Information

This function processes the device license information.

Function	Description
AlprSDK_GetDevLicenseInfo	Get device license information.

Device Version Information

This function processes the device version information.

Function	Description
AlprSDK_GetDevVerInfo	Get device version information.

3.10.3 Device Control Operations

System Time

This function processes to set the system time.

Function	Description
AlprSDK_SetSysTime	Set system time.

Open Barrier

This function processes to open the Boom Barrier.

Function	Description
AlprSDK_OpenGate	Open the barrier.

Close Barrier

This function processes to Close the Boom Barrier.If you need to use this interface, you need to customize the firmware, which is not currently supported by mainstream firmware.

Function	Description
AlprSDK_CloseGate	Close the barrier.

Open Barrier with JPG ID

This function processes to open the Boom Barrier with JPG id.

Function	Description
AlprSDK_OpenGateByJpgID	Open the barrier with JPG id

RS485 transparent transmission

This function processes to connect the 485 transparent transmission

Function	Description
AlprSDK_CommTransparentTransfer	485 transparent transmission
AlprSDK_CommTransparentTransferByJpgID	485 transparent transmission (including jpg id)

Trans Voice and Screen

This function processes to process the voice through a camera and the display through a screen.

Function	Description
AlprSDK_Trans2Voice	Pass through voice through a camera
AlprSDK_Trans2Screen	Display content through the camera transmission screen

Device Reboot

This function processes to restart the Device.

Function	Description

AlprSDK_RebootDevice	Restart Device

3.10.4 Information Parameter Settings

Gate Configuration Settings

This function sets and gets the gate parameter channel configurations.

Function	Description
AlprSDK_SetGateCfg	Set the gate channel.
AlprSDK_GetGateCfg	Get the gate channel

Auxiliary Output

This function processes to control the auxiliary output.

Function	Description
AlprSDK_OpenAuxOut	Control auxiliary output.

3.10.5 Camera Recognition Parameters

Alpr Configuration Parameters

This function processes to set and get the Alpr configuration parameters

Function	Description
AlprSDK_SetAlprCfg	This function sets the identification parameters.
AlprSDK_GetAlprCfg	This function gets the camera recognition parameters

Offline Parameters

This function processes the offline parameters.

Function	Description
AlprSDK_SetOfflineParam	This function sets the offline parameters.
AlprSDK_GetOfflineParam	This function gets the offline parameters

OSD Parameters

This function processes to set and get the OSD parameters

Function	Description
AlprSDK_SetOSD	This function sets the OSD parameters.
AlprSDK_GetOSD	This function gets the OSD parameters

Basic Parameters

This function processes to set and get the basic parameters

Function	Description
AlprSDK_SetBaseParam	This function sets the basic parameters.
AlprSDK_GetBaseParam	This function gets the basic parameters

Operate SD Card

This function processes the SD Card operations.

Function	Description
AlprSDK_OperateSDCard	Operate SD Card.

Restore Device Parameters

This function processes to restore the device parameters.

Function	Description
AlprSDK_ResetFactory	Restore the device parameters.

3.10.6 License Plate

License Plate Information

This function processes to Add and Delete License Plate Information

Function	Description
AlprSDK_AddPlateListExt	Additional license plate information (if there is a duplicate license plate, it will be replaced directly) and user name will be added to the license plate list.
AlprSDK_DelPlateListExt	This function deletes the license plate information

Clear Cars from the list

This function processes to delete or clear the whitelist, blacklist and the fixed car list information.

Function	Description
AlprSDK_ClearPlateListExt	This function clears the license plate information (extension).

Import and Export License Plate Information

This function processes to import and export license plate information.

Function	Description
AlprSDK_ExportPlateListExt	This function exports the license plate information.

3.10.7 Data Transmission

Get Data Count

This function processes to get the total data count from the device.

Function	Description
AlprSDK_GetDeviceDataCount	Amount of data obtained from the device.

Device Data

This function processes to get the data from the device and send the data to the device.

Function	Description
AlprSDK_GetDeviceData	Get data from the device
AlprSDK_SetDeviceData	Send data to the device

Clear Data

This function processes to clear the data from the device.

Function	Description
AlprSDK_ClearDeviceData	Clear a table

3.10.8 Voice Intercom

Initialization

This function processes to initialize the voice intercom operation.

Function	Description
AlprSDK_StartVoice	Activate voice intercom

Decoding

This function processes to decode the voice data.

Function	Description
AlprSDK_InputAudio	Decoding speech data

Playback ON

This function processes to playback the voice data.

Function	Description
AlprSDK_PlaySound	Open to play the voice data

Send Voice Data

This function processes to send the voice data.

Function	Description
AlprSDK_SendVoiceData	Send voice data

Exit Voice Intercom

This function processes to end the voice intercom operation.

Function	Description
AlprSDK_StopVoice	Stop the voice intercom

3.10.9 P2P Service

Set P2P

This function processes to set the P2P server address.

Function	Description
AlprSDK_SetP2PServerAddress	Set the P2P server address

Reconnect

This function processes to enable and disable the p2p reconnection.

Function	Description
AlprSDK_EnableP2PReconnect	Turn on P2P reconnection (valid only for P2P connections)
AlprSDK_DisableP2PReconnect	Close P2P reconnect (valid only when P2P connection is active).

Network, Video and Device Parameters

This function processes to set and obtain the network, video and device parameter settings.

Function	Description
AlprSDK_SetNetworkParam	Set network parameters
AlprSDK_GetNetworkParam	Get network parameters
AlprSDK_SetVideoParam	Set the video parameters

AlprSDK_GetVideoParam	Get video parameters
AlprSDK_GetDeviceParam	Get device parameters

3.10.10 Manual identification

Manual identification

This interface is used for manual identification and is not affected by various strategies. It can directly identify the current picture.

Function	Description
AlprSDK_RecogByManual	Manual identification.

3.11 Parking Camera Interface

Parking Callback

This function sets the parking information by calling the parking space callback function.

Function	Description
CarSpaceStateCallback	Parking space information.
AlprSDK_CreateCarSpaceTask	Parking space callback function setting.
AlprSDK_ClearCarSpaceTask	Parking space callback function cleanup.

Set and Get the Parking Camera Parameters

This function processes to set up the parking camera parameter configuration settings.

Function	Description
AlprSDK_SetCarSpaceConfig	Set the parking camera parameter setting.
AlprSDK_GetCarSpaceConfig	Get the parking camera parameter setting

Update Space Information

This function processes to refresh and updates the parking space information in each interval.

Function	Description
AlprSDK_RefreshCarSpaceState	Actively trigger parking status upload.

4 AlprSDK Functions

4.1 AlprSDK_Startup

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_Startup
(
    HWND    hNotifyWnd,
    UINT    nCommandID
)
```

Description

This function initializes the SDK Device.

Parameters

Parameter	Description
`HWND hNotifyWnd`	**In:** A handle to the window.
`UINT nCommandID`	**In**: The command ID to be processed. **Setting ID**: Defined value is 0 x 500.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

Remarks

hNotifyWnd is recommended to be set to NULL.

nCommandID is recommended to be set to 0.

4.2 AlprSDK_StartupWithPath

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_StartupWithPath(const char* path);
```

Description

Start AlprSDK on Linux System.

Parameters

Parameter	Description
`const char* path`	**In:** Path to Key License File.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.3 AlprSDK_SearchAllCameras

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SearchAllCameras
(
      unsigned int   nTimeout,
            void *  callback
)
```

Description

This function searches for all Camera Devices.

Parameters

Parameter	Description
`unsigned int  nTimeout`	**In:** Waiting time for searching the DVS (in milliseconds).
`void *callback`	**In**: Search DVS callback function.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.4CallbackServerFind

Callback Function Description

```
typedef void(WINAPI *CallbackServerFind)
(
DVS_MACHINE_TYPE     nDeviceType,
          char *     pDeviceName,
          char *     pIP,
            BYTE     macAddr[6],
            WORD     wPortWeb,
            WORD     wPortListen,
          char *     pSubMask,
          char *     pGateway,
          char *     pMultiAddr,
          char *     pDnsAddr,
            WORD     wMultiPort,
              int    nChannelNum,
              int    nFindCount,
            DWORD    dwDeviceID
);
```

Callback Parameter Description

Parameter	Description
nDeviceType	Device model
pDeviceName	The name of the DVS
pIP	DVS IP address
macAddr	The physical address of the DVS to be set
wPortWeb	DVS web port number
wPortListen	DVS communication port

pSubMask	The subnet mask of DVS
pGateway	DVS ID Gateway
pMultiAddr	DVS multicast address
pDnsAddr	DVS DNS address
wMultiPort	DVS multicast port number
nChannelNum	Number of channels in DVS
nFindCount	Discover the number of DVS

Remarks

nFindCount: If one parameter does not have the value 0, then the previous parameter is invalid.

4.5 AlprSDK_InitHandle

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_InitHandle
(
    int     nHandleID,
    HWND    hVideoWnd
)
```

Description

This function initializes the window handle.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.
HWND hVideoWnd	**In**: Video display window handle set to NULL if no video is needed.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.6 AlprSDK_ConnectDev

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_ConnectDev
(
                int     nHandleID,
  const DEVINFO *       pDev,
    EAPIClientType      type
)
```

Description

This function connects the available Camera Devices.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`const DEVINFO * pDev`	**In**: Link information for pDev devices (device information returned by the function AlprSDK_SearchAllCameras).
`EAPIClientType type`	**In:** Type of connection

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.7 AlprSDK_SetConnectTimeout

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SetConnectTimeout
(
        int   nHandleID,
unsigned int  nTimeout
)
```

Description

This function sets the connection timeout.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles
`unsigned int  nTimeout`	**In**: Timeout (in milliseconds)

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.8 AlprSDK_SendHeartBeat

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SendHeartBeat(int   nHandleID )
```

Description

This function sends the heartbeat signals to get the device connection status.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.9 AlprSDK_CreateRecogAllInfoTask

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_CreateRecogAllInfoTask
(
                  int    nHandleID,
RecogAllInfoCallback    callback,
```

```
                void *     pUserData
    )
```

Description

License plate callback function settings.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`RecogAllInfoCallback callback`	**In:** Callback delegate
`void * pUserData`	**In:** User data

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.10 AlprSDK_ClearRecogAllInfoTask

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_ClearRecogAllInfoTask (int nHandleID )
```

Description

This function clears the License plate callback function.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.11 RecogAllInfoCallback

Function

```
typedef void  (WINAPI *RecogAllInfoCallback)
(
        RECOG_ALL_INFO *pRecogResult,
        void* pUserData
);
```

Description

Device status callback.

Parameters

Parameter	Description
`RECOG_ALL_INFO *pRecogResult`	**Out：** All the information about the license plate
`void *  pUserData`	**Out:** User defined data, Id that can be used to bind the camera.

4.12 AlprSDK_CaptureJpg

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_CaptureJpg(int  nHandleID )
```

Description

This function triggers the Camera Device to capture the image directly and return it to the application layer through the DeviceCaptureCallback callback function.

Parameters

Parameter	Description
`int  nHandleID`	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.13 DeviceCaptureCallback

Function

```
typedef void  (WINAPI *DeviceCaptureCallback)
(
     void *pBmpBuf,
     int *len
);
```

Description

Image data capture callback function.

Parameters

Parameter	Description
`void *pBmpBuf`	**In:** Picture data header address
`int *len`	**In:** Picture data length

4.14 AlprSDK_CreateCaptureJpgCallback

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_CreateCaptureJpgCallback
(
                          int     nHandleID,
        DeviceCaptureCallback     callback
)
```

Description

This function gets the Device capture callback settings.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`DeviceCaptureCallback callback`	**In:** Callback Function Pointer

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.15 AlprSDK_ClearCaptureJpgCallback

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_ClearCaptureJpgCallback (int nHandleID
    )
```

Description

This function clears the device captured images.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.16 DevStatusCallback

Function

```
typedef void  (WINAPI *DevStatusCallback)
(
```

```
        void *    pBuf,
         int *    len,
        void *    pUserData
    );
```

Description

Device status callback.

Parameters

Parameter	Description
`void *  pBuf`	**Out:** The actual type of the device status data first address (pBuf address refer to DevStatus).
`int *  len`	**Out:** Data length
`void *  pUserData`	**Out:** Custom Data Header Address

4.17 AlprSDK_CreateDevStatusCallback

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_CreateDevStatusCallback
(
    int nHandleID,
    DevStatusCallback callback,
    void* pUserData
);
```

Description

Create Device Status Callback.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`DevStatusCallback callback`	**In:** Callback function
`void * pUserData`	**In:** User Data Pointer

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.18 AlprSDK_ClearDevStatusCallback

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_ClearDevStatusCallback (int   nHandleID)
```

Description

This function clears the device status callback.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.19 AlprSDK_CreateRecordCallback

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_CreateRecordCallback
(
                    int     nHandleID,
        RecordCallback      callback,
                void *      pUserData
    )
```

Description

This function sets the offline logging event callback.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`RecordCallback callback`	**In:** Callback offline logging event callback
`void * pUserData`	**In:** User Data Pointer

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.20 AlprSDK_ClearRecordCallback

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_ClearRecordCallback(int nHandleID);
```

Description

Clear offline logging event callback.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.21 RecordCallback

Function

```
typedef void  (WINAPI *RecordCallback)
```

```
(
    int *nType,
    void *pBuf,
    int *len,
    void* pUserData
);
```

Description

Event record callback function.

Parameters

Parameter	Description
int *nType	event type
void *pBuf	Event data header address
int *len	Event data length
void* pUserData	User-defined data header address

4.22 AlprSDK_CreateWiegandDataCallback

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_CreateWiegandDataCallback
(
                        int    nHandleID,
        WiegandDataCallback    callback,
                    void *     pUserData
```

```
);
```

Description

Vroot card data callback.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`WiegandDataCallback callback`	**In:** Callback function
`void * pUserData`	**In:** User Data Pointer

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.23 AlprSDK_ClearWiegandDataCallback

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_ClearWiegandDataCallback ( int nHandleID
    )
```

Description

This function clears the Weigand data callback.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.24 WiegandDataCallback

Function

```
typedef void  (WINAPI *WiegandDataCallback)
(
```

```
    TWiegandData *pWiegandData,
    void* pUserData
);
```

Description

This function initiates the video.

Parameters

Parameter	Description
`TWiegandData *pWiegandData`	**In:** Wigan card data information
`void* pUserData`	**Out:** User defined data

4.25 AlprSDK_StartVideo

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_StartVideo   (int   nHandleID )
```

Description

This function initiates the video.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.26AlprSDK_SetVideoConfig

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SetVideoConfig
(
                int    nHandleID,
ALPR_VIDEO_CONFIG *   pVideoCtrl
)
```

Description

This function sets the video configurations.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.

ALPR_VIDEO_CONFIG * pVideoCtrl	**In:** Video parameters Supported Parameters： 0: 1080 p，3: D 1

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.27 AlprSDK_GetVideoConfig

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetVideoConfig
(
            int    nHandleID,
ALPR_VIDEO_CONFIG *  pVideoCtrl
)
```

Description

This function gets the video configurations.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.
ALPR_VIDEO_CONFIG * pVideoCtrl	**In:** Video parameters Supported Parameters： 0: 1080 p，3: D 1

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.28AlprSDK_GetVideoSize

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetVideoSize
(
    int     nHandleID,
    int *   pWidth,
    int *   pHeight
)
```

Description

This function gets the size of the video.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`int * pWidth`	**Out:** Video width
`int * pHeight`	**Out:** Video height

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.29 AlprSDK_RegCBOnDraw

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_RegCBOnDraw
(
        int     nHandleID,
HHCBOnDraw      callback,
    void *      pContext
    )
```

Description

This function registers the Drawing Callback.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`HHCBOnDraw callback`	**In:** Callback function
`void * pContext`	**In:** User data pointer

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

Remarks

- To draw a graph in the video area, use the API to register the callback function.
- Each time the video frame is refreshed, the callback function set by the **/ n** is automatically called.
- And any graph can be drawn through the **GDI** in callback function.

4.30HHCBOnDraw

Function

```
typedef int (WINAPI *HHCBOnDraw)(
      USHORT nPort,
      HDC hdc,
      int nWidth,
      int  nHeight,
      void *pContext
);
```

Description

Drawing callback function.

Parameters

Parameter	Description
USHORT nPort	Player number (0 ~ 127)
HDC hdc	HDC handle for device
int nWidth	The width of the paintable area
int nHeight	The height of the drawable area
void *pContext	Context-sensitive parameters

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.31 AlprSDK_StopVideo

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_StopVideo (  int nHandleID  )
```

Description

This function is to turn off the video.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.32AlprSDK_DisConnectDev

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_DisConnectDev    (   int nHandleID  )
```

Description

This function disconnects the device connection.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.33AlprSDK_UnInitHandle

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_UnInitHandle (   int nHandleID  )
```

Description

This function releases the handle.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.34 AlprSDK_Cleanup

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_Cleanup   ()
```

Description

This function is to close the AlprSDK and release the related resources.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.35 AlprSDK_SetRoiEx

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SetRoiEx
(
  int   nHandleID,
  const XPoint rgnRoi[4]
)
```

Description

This function is for setting identification region.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`const XPoint rgnRoi[4]`	**In:** Coordinate Area

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

XPoint Function Description

```
typedef struct _XPoint
{
    int x;
    int y;
    float Dist(_XPoint p)
    {
        int dx = x - p.x; dx = dx*dx;
        int dy = y - p.y; dy = dy * dy;
        return sqrt((float)dx + dy);
    }
}XPoint;
```

4.36AlprSDK_GetRoiEx

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetRoiEx
(
    int     nHandleID,
    XPoint  rgnRoi[4]
)
```

Description

This function is for obtaining the identification area expansion.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`const XPoint rgnRoi[4]`	**Out:** Coordinate region

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

XPoint Function Description

```
typedef struct _XPoint
{
    int x;
    int y;
    float Dist(_XPoint p)
    {
        int dx = x - p.x; dx = dx*dx;
        int dy = y - p.y; dy = dy * dy;
        return sqrt((float)dx + dy);
    }
}XPoint;
```

4.37 AlprSDK_SetVirtualCoil

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SetVirtualCoil
(
        int   nHandleID,
const XPoint  rgnRoi[4]
    )
```

Description

This function sets the virtual coil area.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.
const XPoint rgnRoi[4]	**In:** Virtual Coil Area

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

XPoint Function Description

```
typedef struct _XPoint
{
    int x;
    int y;
    float Dist(_XPoint p)
    {
        int dx = x - p.x; dx = dx*dx;
        int dy = y - p.y; dy = dy * dy;
        return sqrt((float)dx + dy);
    }
}XPoint;
```

4.38AlprSDK_GetVirtualCoil

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetVirtualCoil
(
        int     nHandleID,
    XPoint      rgnRoi[4]
)
```

Description

This function gets the virtual coil area.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`const XPoint rgnRoi[4]`	**In:** Virtual Coil Area

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

XPoint Function Description

```
typedef struct _XPoint
{
```

```
    int x;
    int y;
    float Dist(_XPoint p)
    {
        int dx = x - p.x; dx = dx*dx;
        int dy = y - p.y; dy = dy * dy;
        return sqrt((float)dx + dy);
    }
}XPoint;
```

4.39 AlprSDK_GetDeviceFunList

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetDeviceFunList
(
    int     nHandleID,
char *      devFunList,
    int     len
    )
```

Description

This function gets the functionality supported by the device.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.

`char *  devFunList`	**Out:** Upper layer allocation of 256-size buf, the meaning of the value corresponding to each byte refers to the enumeration eFunList
`int  len`	**In:** Upload fixed to 256

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.40 AlprSDK_GetDevLicenseInfo

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetDevLicenseInfo
(
        int     nHandleID,
PDevLicense     pDevLicense
    )
```

Description

This function gets the device license information.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`PDevLicense pDevLicense`	**Out:** License related information

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.41 AlprSDK_GetDevVerInfo

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetDevVerInfo
(
          int    nHandleID,
DEV_VER_INFO *    pDevVerInfo
    )
```

Description

This function gets the device version information.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`DEV_VER_INFO * pDevVerInfo`	**Out:** Device version information

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.42 AlprSDK_SetSysTime

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SetSysTime
(
        int     nHandleID,
 PSYS_TIME      pSysTime
    )
```

Description

This function sets the system time.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`PSYS_TIME pSysTime`	**In:** Temporal information

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.43 AlprSDK_OpenGate

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_OpenGate ( int nHandleID )
```

Description

This function opens the gate.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.44 AlprSDK_CloseGate

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_CloseGate(int nHandleID);
```

Description

Close the gate.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

Remarks

If you need to use this interface and need to customize firmware, please contact the relevant technical personnel.

4.45 AlprSDK_OpenGateByJpgID

Function

```
ALPRSDK_APIOS_Error WINAPI AlprSDK_OpenGateByJpgID
(
int    nHandleID,
char   pJpgID[JPG_ID_LENGTH]
);
```

Description

This function opens the gateway with the jpg ID.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`char pJpgID[JPG_ID_LENGTH]`	**In:** Picture ID. When receiving the picture id, if offline has been processed, the command will not be executed.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

Remarks

The picture ID will not be received if it is processed in the offline, and the command will not be executed

4.46 AlprSDK_RebootDevice

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_RebootDevice (   int    nHandleID  )
```

Description

This function restarts the Device.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.47 AlprSDK_SetGateCfg

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SetGateCfg
(
                int    nHandleID,
```

```
    const GATE_CONFIG *     pGate
      )
```

Description

This function sets the channel gate configuration parameters.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.
const GATE_CONFIG * pGate	**In:** Configuration parameters of gate.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.48 AlprSDK_GetGateCfg

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetGateCfg
(
          int    nHandleID,
GATE_CONFIG *    pGate
    )
```

Description

This function is to access the gateway configuration parameters.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`GATE_CONFIG * pGate`	**In:** Configuration parameters of gate

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.49 AlprSDK_OpenAuxOut

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_OpenAuxOut
(
    int nHandleID,
    int nIndex,
    int nTime
);
```

Description

This function controls the auxiliary output.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`int nIndex`	**In:** Auxiliary output number (value is 1, 2, 3,)
`int nTime`	**In:** Auxiliary output time (value 0, 2, 3, 255,)

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

Remarks

Mutually exclusive with traffic lights; secondary output interfaces cannot be used when the traffic light interfaces are being used.

4.50AlprSDK_SetAlprCfg

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SetAlprCfg
(
                int   nHandleID,
const ALPR_CONFIG *   pAlpr
    )
```

Description

This function sets the identification parameters.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.
const ALPR_CONFIG * pAlpr	**In:** Gate configuration parameters

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.51 AlprSDK_GetAlprCfg

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetAlprCfg
(
          int    nHandleID,
ALPR_CONFIG *    pAlpr
    )
```

Description

This function gets the camera recognition parameters.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.
ALPR_CONFIG * pAlpr	**Out:** Identification parameter

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.52AlprSDK_SetOfflineParam

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SetOfflineParam
(
                int   nHandleID,
const POfflineParam   pOfflineParam
    )
```

Description

This function sets the offline parameters.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles

`const POfflineParam pOfflineParam`	**In:** Offline charging reference

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.53 AlprSDK_GetOfflineParam

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetOfflineParam
(
        int    nHandleID,
 POfflineParam    pOfflineParam
   )
```

Description

This function gets the offline parameters

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles
POfflineParam pOfflineParam	**Out:** offline charge parameters

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.54 AlprSDK_SetOSD

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SetOSD
(
      int    nHandleID,
 OSD_CTRL *   pOsdCtrl
   )
```

Description

This function sets the OSD parameters.

Parameters

Parameter	Description

int nHandleID	**In:** Number of handles
OSD_CTRL * pOsdCtrl	**In:** OSD parameters

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.55 AlprSDK_GetOSD

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetOSD
(
        int     nHandleID,
OSD_CTRL *      pOsdCtrl
    )
```

Description

This function gets the OSD parameters.

Parameters

Parameter	Description

int nHandleID	**In:** Number of handles
OSD_CTRL * pOsdCtrl	**Out:** OSD parameters

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.56AlprSDK_SetBaseParam

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SetBaseParam
 (
              int    nHandleID,
 const PBASE_PARAM    pBase_Param
 )
```

Description

This function sets the basic parameters.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles
`const PBASE_PARAM pBase_Param`	**In:** basic parameters (whether to display identification area and virtual coil)

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.57 AlprSDK_GetBaseParam

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetBaseParam
(
        int    nHandleID,
PBASE_PARAM    pBase_Param
    )
```

Description

This function gets basic parameters.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles
`PBASE_PARAM pBase_Param`	**In:** Basic parameters (To display the identification area and the virtual coil)

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.58 AlprSDK_OperateSDCard

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_OperateSDCard
(
            int     nHandleID,
 SDCard_OpType_E    nOpType,
 PSDCardCapacity    pSDCardCapacity
   )
```

Description

This function is to operate the SD card.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles
`SDCard_OpType_E nOpType`	**In:** SD card operation type
`PSDCardCapacity pSDCardCapacity`	**In:** SD card capacity information

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.59 AlprSDK_ResetFactory

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_ResetFactory
(
    int  nHandleID,
    int  nResetType
)
```

Description

This function restores the device parameters.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles
`int nResetType`	**In:** Restart type (0: restore technical parameters, 1: restore network parameters, 2: restore all parameters)

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.60 AlprSDK_AddPlateListExt

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_AddPlateListExt
(
```

```
            int       nHandleID,
            int       t,
PLATE_INFO_EXT *      pPlateList,
          int         count
   )
```

Description

Additional license plate information (if there is a duplicate license plate, it will be replaced directly).

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles
`int t`	**In:** License plate type, value parameter enumeration link ePlateListType endlink
`PLATE_INFO_EXT * pPlateList`	**In:** Store data buffer, memory is allocated by the call layer
`int count`	**In:** Number of license plate data

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.61 AlprSDK_DelPlateListExt

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_DelPlateListExt
(
            int  nHandleID,
            int  t,
PLATE_INFO_EXT *  pPlateList,
            int  count
   )
```

Description

This function deletes the license plate information.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles
int t	**In:** License plate type, value parameter enumeration link ePlateListType endlink
PLATE_INFO_EXT * pPlateList	**In:** Store data buffer, memory is allocated by the call layer
int count	**In:** Number of license plate data

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.62AlprSDK_ClearPlateListExt

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_ClearPlateListExt
(
        int   nHandleID,
        int   t
)
```

Description

This function clears the license plate information (extension).

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles
int t	**In:** License plate type, 0: whitelist, 1: blacklist, 2: fixed

	car list.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.63 AlprSDK_ExportPlateListExt

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_ExportPlateListExt
(
                int  nHandleID,
                int  t,
    PLATE_INFO_EXT *  pPlateList,
               int *  pCount
)
```

Description

This function exports the license plate information.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles
int t	**In:** License plate type, value parameter enumeration link ePlateListType
PLATE_INFO_EXT * pPlateList	**In:** Store data buffer, memory is allocated by the call layer. The SDK puts the data returned by the device into the buffer.
int count	**Out:** Number of license plate data

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.64 AlprSDK_GetDeviceDataCount

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetDeviceDataCount
(
                    int    nHandleID,
        DATA_TABLE_TYPE_E    tType,
```

```
                    int    offset,
                int *    nCount
    )
```

Description

Amount of data obtained from the device.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles
DATA_TABLE_TYPE_E tType	**In:** Table type
int offset	**In:** Starts with an ID.
int * nCount	**Out:** Number of nCount records

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.65 AlprSDK_GetDeviceData

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetDeviceData
(
                        int    nHandleID,
        DATA_TABLE_TYPE_E    tType,
                        int    offset,
                    char *    pBuf,
                     int *    nCount
    )
```

Description

This function gets the data from the device.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles
DATA_TABLE_TYPE_E tType	**In:** Table type
int offset	**In:** Starts with an ID.
char * pBuf	**Out:** Data pointer
int * nCount	**Out:** The number of records/ devices that the nCount software wants to take is the number of records returned

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.66AlprSDK_SetDeviceData

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SetDeviceData
(
                int nHandleID,
 DATA_TABLE_TYPE_E tType,
               char *pBuf,
                int nCount
)
```

Description

This function sends the data to the device.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles
DATA_TABLE_TYPE_E tType	**In:** Table type
char * pBuf	**In:** Data pointer
int nCount	**Out:** Number of nCount rows

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.67 AlprSDK_ClearDeviceData

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_ClearDeviceData
(
                        int    nHandleID,
          DATA_TABLE_TYPE_E    tType
)
```

Description

This function clears a table.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles
DATA_TABLE_TYPE_E tType	**In:** Table type

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.68AlprSDK_CommTransparentTransfer

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_CommTransparentTransfer
(
            int    nHandleID,
unsigned char *   pdata,
            int    len
);
```

Description

485 transparent transmission.

Parameters

Parameter	Description
`int nHandleID`	**In:** Name of handles.
`unsigned char * pdata`	**In:** Data Head address. Please refer to Appendix 2
`int len`	**In:** Data Length

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

Remarks

- The length cannot exceed 255, and it must be in online mode to be valid, where it does not work in the offline mode.
- This API can directly transmit RS485 data.
- For example, the RS485 data for "Welcome" is AA 42 00 00 AE 1A 00 23 09 00 01 BB B6 D3 AD B9 E2 C1 D9 24 0B 00 00 32 30 31 38 2D 30 35 2D 32 38 F5 35.
- Just put the binary data into pdata.
- For the RS485 data combination method, please refer to the 485 dual color screen communication protocol.

4.69 AlprSDK_CommTransparentTransferByJpgID

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_CommTransparentTransferByJpgID
(
            int    nHandleID,
           char    pJpgID[JPG_ID_LENGTH],
 unsigned char *   pdata,
            int    len
);
```

Description

485 transparent transmission (including jpg id).

Parameters

Parameter	Description
`int nHandleID`	**In:** Name of handles.
`char pJpgID[JPG_ID_LENGTH]`	**In:** Picture ID
`unsigned char * pdata`	**In:** Data Head address.Please refer to Appendix 2
`int len`	**In:** Data Length

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

Remarks

- The picture ID will not be received if it is processed in the offline, and the command will not be executed.
- The length cannot exceed 255, and it must be in online mode to be valid, and it does not work in offline mode
- This API can directly transmit RS485 data.
- For example, the RS485 data for "Welcome" is AA 42 00 00 AE 1A 00 23 09 00 01 BB B6 D3 AD B9 E2 C1 D9 24 0B 00 00 32 30 31 38 2D 30 35 2D 32 38 F5 35

- Simply put the binary data into pdata.
- Note: RS485 data combination method can be found in 485 dual color screen communication protocol

4.70 AlprSDK_CarInfoRecogInit

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_CarInfoRecogInit();
```

Description

This function initializes vehicle features.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.71 AlprSDK_SwitchJpgToRGB

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SwitchJpgToRGB
(
const unsigned char *    pJpg,
     cCOLOR_IMAGE_I *    pstImageData
);
```

Description

This function converts Jpg Image to rbg Pixel.

Parameters

Parameter	Description
`const unsigned char *  pJpg`	**In:** Image Data,Obtaining jpg from License Plate Callback
`cCOLOR_IMAGE_I * pstImageData`	**In:** Image BGR data buffer

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.72 AlprSDK_CreateColorImageI

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_CreateColorImageI
(
    const char *  filename,
cCOLOR_IMAGE_I *  pstImageData);
```

Description

This function creates a color image.

Parameters

Parameter	Description
`const char *  filename`	**In:** Name of the file.
`cCOLOR_IMAGE_I *  pstImageData`	**In:** Image data.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.73 AlprSDK_SmartALPR_RecogColor

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SmartALPR_RecogColor
(
const cCOLOR_IMAGE_I *   pstImageData,
         const XRect *   pPlatePos,
       stColorResult *   pColorResult
);
```

Description

This function identifies the vehicle color.

Parameters

Parameter	Description
`const cCOLOR_IMAGE_I *  pstImageData`	**In:** Image Data
`const XRect * pPlatePos`	**In**: License plate location
`stColorResult *  pColorResult`	**In**: Color comparison results

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.74AlprSDK_SmartALPR_RecogCarType

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SmartALPR_RecogCarType
(
 const cCOLOR_IMAGE_I *   pstImageData,
          const XRect *   pPlatePos,
         stLogoResult *   pTypeResult
);
```

Description

This function identifies the vehicle type.

Parameters

Parameter	Description
`const cCOLOR_IMAGE_I *  pstImageData`	**In**: Image Data
`const XRect * pPlatePos`	**In**: License plate location
`stLogoResult *  pTypeResult`	**In**: Result Type

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.75 AlprSDK_SmartALPR_RecogLogo

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SmartALPR_RecogLogo
(
const cCOLOR_IMAGE_I  *  pstImageData,
          const XRect *  pPlatePos,
         stLogoResult *  pLogoResult
);
```

Description

This function identifies the car brand logo.

Parameters

Parameter	Description
`const cCOLOR_IMAGE_I *  pstImageData`	**In**: Image Data
`const XRect * pPlatePos`	**In**: License plate location
`stColorResult *  pColorResult`	**In**: Vehicle Logo

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.76 AlprSDK_DestroyColorImageI

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_DestroyColorImageI
(
const cCOLOR_IMAGE_I *pstImageData
);
```

Description

Clear color picture.

Parameters

Parameter	Description
`const cCOLOR_IMAGE_I *pstImageData`	**In**: Image Data

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.77 AlprSDK_Trans2Voice

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_Trans2Voice
(
          int    nHandleID,
          int    nScreenType,
 unsigned char   voiceIDList[],
          int    nCount
);
```

Description

Pass through voice through a camera.

Parameters

Parameter	Description
`int nHandleID`	**In:** Name of handles.
`int nScreenType`	**In:** Refers to the ScreenType, Note: Currently supports two-colour dual-line screen.
`unsigned char voiceIDList[]`	**In:** Voice code list refers to Appendix 3
`int nCount`	**In:** Number of voices.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.78 AlprSDK_Trans2Screen

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_Trans2Screen
(
        int   nHandleID,
        int   nScreenType,
        int   nFirstLineColor,
const char *  pFirstLineData,
        int   nSecondLineColor,
const char *  pSecondLineData,
        int   nThirdLineColor,
const char *  pThirdLineData,
        int   nFourthLineColor,
const char *  pFourthLineData
);
```

Description

This function displays the content through the camera transmission screen.

Parameters

Parameter	Description

int nHandleID	**In:** Name of handles.
int nScreenType	**In:** Refers to the ScreenType, Note: Currently supports two-colour dual-line screen.
int nFirstLineColor	**In:** This line displays the color, refer to ScreenShowColor
const char * pFirstLineData	**In:** This line shows the content utf8 encoding
int nSecondLineColor	**In:** This line displays the color, refer to ScreenShowColor
const char * pSecondLineData	**In:** This line shows the content utf8 encoding
int nThirdLineColor	**In:** This line displays the color, refer to ScreenShowColor
const char * pThirdLineData	**In:** This line shows the content utf8 encoding
int nFourthLineColor	**In:** This line displays the color, refer to ScreenShowColor
const char * pFourthLineData	**In:** This line shows the content utf8 encoding

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.79 AlprSDK_StartVoice

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_StartVoice
```

```
(
                    int     nHandleID,
                  DWORD     dwVoiceChan,
 fVoiceDataCallBackV2     fpVoiceDataCallBack,
                 LPVOID     lpUser,
                  DWORD     dwAudioType
);
```

Description

This function activates the voice intercom.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.
DWORD dwVoiceChan	**In:** Voice channel number (incoming 1)
fVoiceDataCallBackV fpVoiceDataCallBack	**In:** Process receives a callback for the opposite voice
LPVOID lpUser	**In:** Parameters for incoming callback
DWORD dwAudioType	**In**: It is an audio coding type range of 0 / OggVorbis ⩽ 1 ⩽ G711 / U / 2 / G711 / A / 3 / G726. Currently, only G711U and G711A types are supported, sampling rate 8000, bit rate 16, mono audio input and output data)

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.80fVoiceDataCallBackV2

fVoiceDataCallBackV2 Function Description

```
typedef void (CALLBACK *fVoiceDataCallBackV2)
(
    HANDLE hVoiceHandle,
      BYTE *lpBuffer,
      DWORD dwBufferSize,
      DWORD dwAudioFlag,
     LPVOID lpUser
);
```

Description

Voice intercom data callback function (extended callback)

Parameters

Parameter	Description
HANDLE hVoiceHandle	
BYTE *lpBuffer	
DWORD dwBufferSize	
DWORD dwAudioFlag	
LPVOID lpUser	

4.81 AlprSDK_InputAudio

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_InputAudio
(
              int     nHandleID,
 unsigned char *      pbuffer,
              int     nLen,
              int     ntype
);
```

Description

This function decodes the voice data.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`unsigned char * pbuffer`	**In:** Voice data
`int nLen`	**In:** Voice data length
`int ntype`	**In:** Voice data type

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.82 AlprSDK_PlaySound

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_PlaySound(int nHandleID);
```

Description

Open Play Voice.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.83 AlprSDK_SendVoiceData

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SendVoiceData
(
      int     nHandleID,
    DWORD     dwVoiceOpt,
   BYTE *     lpBuffer,
    DWORD     dwSize
);
```

Description

This function sends the voice data.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.
DWORD dwVoiceOpt	**In:** bit0~bit7: audio coding type. 0--OggVorbis 1--G711_U 2--G711_A 3--G726 bit8~bit15: sampling rate, 0 / 8000 bit 16 / 23: bit rate, 0 / 16, 1 / 32-bit 24 bit 32: Channel, 0-mono,

	1-double channel.
BYTE * lpBuffer	**In:** Voice data.
DWORD dwSize	**In:** Voice data length.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.84 AlprSDK_StopVoice

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_StopVoice(int nHandleID);
```

Description

This function stops the voice intercom.

Parameters

Parameter	Description

`int nHandleID`	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.85 AlprSDK_SetP2PServerAddress

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SetP2PServerAddress
(
    char *pP2PServerAddress
);
```

Description

This function sets the p2p server address.

Parameters

Parameter	Description
`char *pP2PServerAddress`	**In:** Address, can be domain name.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

Remarks

Considerations for functions with device handles.

4.86AlprSDK_EnableP2PReconnect

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_EnableP2PReconnect(int nHandleID);
```

Description

This function turns on the P2P reconnection (valid only for P2P connections).

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of the handle

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

Remarks

- When the reconnect function is enabled, the user's following interface is called.
- AlprSDK_DisConnectDev, AlprSDK_StopVideo is pre-conditional, the P2P reconnection function is disabled, otherwise the call will fail.
- The calling timing of this interface should be before AlprSDK_ConnectDev, AlprSDK_StartVideo.

4.87 AlprSDK_DisableP2PReconnect

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_DisableP2PReconnect(int nHandleID);
```

Description

This function closes the P2P reconnect (valid only when P2P connection is active).

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of the handle

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

Remarks

- When this interface is called, if the SDK is reconnecting, wait until the SDK reconnection is over.

4.88 AlprSDK_SetNetworkParam

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SetNetworkParam
(
```

```
                int     nHandleID,
    PNetworkParam     pNetworkParam
);
```

Description

This function sets the network parameters.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`PNetworkParam pNetworkParam`	**In:** Network Parameter.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.89 AlprSDK_GetNetworkParam

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetNetworkParam
(
                int    nHandleID,
    PNetworkParam pNetworkParam
);
```

Description

This function gets the network parameters.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`PNetworkParam pNetworkParam`	**Out:** Network Parameter.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.90AlprSDK_SetVideoParam

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SetVideoParam
(
            int   nHandleID,
    PVideoParam   pVideoParam
);
```

Description

This function sets the video parameters.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.
PVideoParam pVideoParam	**In:** Network Parameter.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.91 AlprSDK_GetVideoParam

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetVideoParam
(
            int    nHandleID,
    PVideoParam    pVideoParam
);
```

Description

This function gets the video parameters.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`PVideoParam pVideoParam`	**Out:** Network Parameter.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

Remarks

Considerations for functions with device handles.

4.92 AlprSDK_GetDeviceParam

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetDeviceParam
(
                int    nHandleID,
      PDeviceParam    pDeviceParam
);
```

Description

This function gets the device parameters.

Parameters

Parameter	Description
int nHandleID	**In:** Number of handles.
PDeviceParam pDeviceParam	**Out:** Device Parameter.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

Remarks

Considerations for functions with device handles.

4.93CarSpaceStateCallback

Function

```
typedef void  (WINAPI *CarSpaceStateCallback)
(
    CAR_SPACE_INFO *      pCarSpaceInfo,
                void*     pUserData
);
```

Description

Parking space information.

Parameters

Parameter	Description
CAR_SPACE_INFO * pCarSpaceInfo	**Out:** Parking space information.
void* pUserData	**Out:** User self-defining data.

Remarks

Considerations for functions with device handles.

4.94 AlprSDK_CreateCarSpaceTask

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_CreateCarSpaceTask
(
                        int    nHandleID,
    CarSpaceStateCallback    callback,
                       void*  pUserData
);
```

Description

This function sets the Parking space callback function.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`CarSpaceStateCallback callback`	**In:** Callback delegate.
`void* pUserData`	**In:** User data.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.95 AlprSDK_ClearCarSpaceTask

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_ClearCarSpaceTask(int nHandleID);
```

Description

This function clears the parking space callback function.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.96AlprSDK_SetCarSpaceConfig

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_SetCarSpaceConfig
(
    int nHandleID,
  const TCarSpaceConfig *carSpaceConfig
);
```

Description

This function sets the parking camera parameters.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`const TCarSpaceConfig *carSpaceConfig`	**Out:** Parking space parameters

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.97 AlprSDK_GetCarSpaceConfig

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_GetCarSpaceConfig
(
                  int     nHandleID,
    TCarSpaceConfig *     carSpaceConfig
);
```

Description

This function gets the parking space camera parameters.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.
`TCarSpaceConfig * carSpaceConfig`	**Out:** Parking space parameter. Please refer to AlprSDK_SetCarSpaceConfig for definition

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.98AlprSDK_RefreshCarSpaceState

Function

```
ALPRSDK_API OS_Error WINAPI AlprSDK_RefreshCarSpaceState(int nHandleID);
```

Description

This function refreshes and actively updates the parking space status.

Parameters

Parameter	Description
`int nHandleID`	**In:** Number of handles.

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

4.99AlprSDK_RecogByManual

Function

ALPRSDK_API OS_Error WINAPI AlprSDK_RecogByManual(int nHandleID);

Description

Manual identification, not affected by various strategies, directly identifies the current picture. After manual identification, a gate opening command needs to be configured before the gate can be opened.

Parameters

Parameter	Description
int nHandleID	The number of the handle

Returns

Successful: It returns 0.

Unsuccessful: Returns non-zero Error Code.

Refer: See Appendix 1 for Error Code.

5 Structure definition

5.1 RECOG_ALL_INFO

RECOG_ALL_INFO **Structure Description**

```
typedef struct RecogAllInfo
{
    LICENSE_PLATE        PlateInfo;
    struct Jpg_Bytes     JpgBytes;
    struct Jpg_Bytes     PlateJpgBytes;
    unsigned char        nReserve[4];
}RECOG_ALL_INFO;
```

Description

License Plate Recognition Information Definition

Parameters

Parameter	Description
LICENSE_PLATE PlateInfo	**Out:** Recognized license plate information
struct Jpg_Bytes JpgBytes	**Out:** Capture JPG picture information
struct Jpg_Bytes PlateJpgBytes	**Out:**Capture the license plate small picture information
unsigned char nReserve[4]	Reserved field

5.2 LICENSE_PLATE

LICENSE_PLATE **Structure Description**

```
typedef struct tagLICENSE_PLATE
{
    char              szTime[20];
    unsigned long     nProcessTime;
    int               nPlateNum;
    PLATERESULT       pPlate[4];
}LICENSE_PLATE;
```

Description

Identification of License Plate Information Definition.

Parameters

Parameter		Description
char	szTime[20]	**Out:** trigger time. format: 20161012163417050, i.e. 16: 34: 17: 050 ms on October 12, 2016
unsigned long	nProcessTime	**Out:** Processing Time of Current Picture
int	nPlateNum	**Out:**Number of plates currently recognized
PLATERESULT	pPlate[4]	You can identify up to 4 license plates at a time.

5.3 PLATERESULT

PLATERESULT **Structure Description**

```
typedef struct tagPLATERESULT
{
    unsigned char    szLicense[24];
    int              nLetterCount;
    float            fConfidence;
    XRect            rect;
    unsigned char    plateColor;
    unsigned char    bDoublePlates;
    unsigned char    nDirection : 3;
    unsigned char    bFakePlate : 1;
    unsigned char    bAlarmPlate : 1;
    unsigned char    bSecondRecog: 1;
    unsigned char    nReserve : 2;
    unsigned char    bReserve[1];
    int              nTypeResult;
    unsigned char    reserve[28];
} PLATERESULT;
```

Description

The result structure of the recognized license plate

Parameters

Parameter	Description
unsigned char szLicense[24]	**Out:** License plate number character
int nLetterCount	**Out:** Number of characters of license plate number
float fConfidence	The credibility of the license plate, the value range [1,100]. The smaller the value, the higher the credibility.
XRect rect	License plate location information
unsigned char plateColor	License plate color, please refer to ePlateColor for values

unsigned char bDoublePlates	Is there a double license plate
unsigned char nDirection : 3	For movement direction, please refer to MotionDir for values.
unsigned char bFakePlate : 1	False license plate
unsigned char bAlarmPlate : 1	Virtual police car license plate
unsigned char bSecondRecog: 1	Camera secondary identification record
unsigned char nReserve : 2	Reserved field
unsigned char bReserve[1]	Reserved field
int nTypeResult	Vehicle Type Identified by Equipment End: Large and Small Vehicles
unsigned char reserve[28]	Reserved field

5.4DEVINFO

`DEVINFO` **Structure Description**

```
Typedef struct _DEVINFO
 {
          char   szIP[32];
          char   szDevName[128];
          char   szDevUid[32];
 unsigned short   uUseP2PConn;
 unsigned short   u16Port;
          char   szUser[64];
          char   szPwd[64];
          char   szPicturesSavePath[256];
       unsigned  short u16AlprPort;
       unsigned  short lprDevType;
         HANDLE   hPullHandle;
 } DEVINFO.
```

Description

Camera device parameter definition.

Parameters

Parameter	Description
`char szIP[32]`	IP address
`char szDevName[128]`	Device name
`char szDevUid[32]`	Device serial number
`unsigned short uUseP2PConn`	Whether to use P2P connection method
`unsigned short u16Port`	Port number

char szUser[64]	Username
char szPwd[64]	Password
char szPicturesSavePath[256]	Picture local save path
unsigned short u16AlprPort	Alpr port number
unsigned short lprDevType	Camera device type, refer to LPR_DEV_TYPE_E
HANDLE hPullHandle	**pullsdk handle**: This sdk only manages its opening and closing, other software directly calls the pullsdk interface

5.5 CAR_INFO

CAR_INFO **Structure Description**

```
typedef struct tagCAR_INFO
{
    unsigned long      nProcessTime;
    int                nPlateNum;
    VEHICLEFEATURE     pCar[4];
}CAR_INFO;
```

Description

Vehicle characteristics.

Parameters

Parameter	Description
unsigned long nProcessTime	Processing Time of Current Picture

int	nPlateNum	Number of plates currently recognized
VEHICLEFEATURE	pCar[4]	You can identify up to 4 cars at a time.

5.6VEHICLEFEATURE

VEHICLEFEATURE **Structure Description**

```
typedef struct tagVehicleFeature
{
    stLogoResult typeResult;
    stLogoResult logoResult;
    stColorResult colorResult;
    unsigned char    reserve[64*8];
} VEHICLEFEATURE;
```

Description

Identified vehicle feature structure.

Parameters

Parameter		Description
stLogoResult	typeResult	Vehicle type
stLogoResult	logoResult	Vehicle logo
stColorResult	colorResult	Vehicle color
unsigned char	reserve[64*8]	Reserved field

5.7 TWIEGANDDATA

`TWiegandData` **Structure Description**

```
typedef struct _WiegandData
{
    unsigned long long   WiegandCardNo;
    unsigned char     Reserve[32];

}TWiegandData, *PWiegandData;
```

Description

Wigan card data.

Parameters

Parameter	Description
`unsigned long long  WiegandCardNo`	Wigan card number
`unsigned char  Reserve[32]`	Reserved field

5.8 CAR_SPACE_INFO

CAR_SPACE_INFO Function Description

```
typedef struct CarSpaceInfo
{
     CAR_SPACES       CarSpaces;
     Jpg_Bytes        JpgBytes;
     unsigned char    nReserve[32];
```

```
}CAR_SPACE_INFO;
```

Parameters

Parameter	Description
CAR_SPACES CarSpaces	Parking space information
Jpg_Bytes JpgBytes	Capture JPG picture information
unsigned char nReserve[32]	Reserved field

5.9 JPG_BYTES

Jpg_Bytes Function Description

```
typedef struct Jpg_Bytes
{
    char           szTime[20];
    int            nBytesLen;
    unsigned char  *pJpgBytes;
}JPG_BYTES;
```

Description

```
Capture JPG picture information definition.
```

Parameters

Parameter	Description
char szTime[20]	The time when the picture was captured is

	20161012163417050, i.e. 16: 34: 17: 050 ms on October 12, 2016
int nBytesLen	Jpg data length
unsigned char *pJpgBytes	Jpg data can be directly saved as jpg files.

5.10 CAR_SPACES

CAR_SPACES Function Description

```
typedef struct tagCAR_SPACES
{
          char    szTime[20];
unsigned long     nProcessTime;
           int    nCarSpaceNum;
  CARSPACEINFO    pCarSpaceInfo[6];
}CAR_SPACES;
```

Description

Definition of all parking space information

Parameters

Parameter	Description
char szTime[20]	Trigger time. Format: 20161012163417050, That is, 16: 34: 17: 050 ms. on October 12, 2016

unsigned long nProcessTime	Processing Time of Current Picture
int nCarSpaceNum	Current number of parking spaces
CARSPACEINFO pCarSpaceInfo[6]	Maximum support for 6 parking space

5.11 CARSPACEINFO

CARSPACEINFO Function Description

```
typedef struct tagCARSPACEINFO
    {
        unsigned char    szLicense[24];
        int              nLetterCount;
        float            fConfidence;
        XRect            rect;
        unsigned char    plateColor;
        unsigned char    bDoublePlates;
        unsigned char    nDirection : 3;
        unsigned char    bFakePlate : 1;
        unsigned char    bAlarmPlate : 1;
        unsigned char    nReserve : 3;
        unsigned char    bReserve[1];
        int              nCarSpaceID;
        int              nCarSpaceState;
#ifdef USE_JNI_INTERFACE
#else
        int typeIndex;
        int logoIndex;
        int colorIndex;
#endif
        unsigned char    reserve[32];
    } CARSPACEINFO;
```

Description

Definition of parking space information

Parameters

Parameter	Description
unsigned char szLicense[24]	license plate number character
int nLetterCount	Number of characters of license plate number
float fConfidence	Credibility of license plate, value range [1,100]. smaller value, higher credibility
XRect rect	License plate location information
unsigned char plateColor	License plate color, please refer to ePlateColor for values
unsigned char bDoublePlates	Is there a double license plate
unsigned char nDirection : 3	For movement direction, please refer to MotionDir for values.
unsigned char bFakePlate : 1	False license plate
unsigned char bAlarmPlate : 1	Virtual police car license plate
unsigned char nReserve : 3	Reserve
unsigned char bReserve[1]	Reserve
unsigned char nReserve [1]	Reserve
int nCarSpaceID	Parking space ID
int nCarSpaceState	Parking space status
int typeIndex	Reserve

int logoIndex	Reserve
int colorIndex	Reserve
unsigned char reserve[32]	Reserve

5.12 TCARSPACECONFIG

TCarSpaceConfig Function Description

typedef struct CarSpaceConfig

```
{
    Bool bEnable[MAX_CAR_SPACE_CNT];
    Bool bEnableCarCheck[MAX_CAR_SPACE_CNT];
    char nWatchRecogCnt;
    char nWatchComfirmCnt;
    char szReserve[256];
}TCarSpaceConfig, *PCarSpaceConfig;
```

Description

Parameter setting

Parameters

Parameter	Description
Bool bEnable[MAX_CAR_SPACE_CNT]	Whether parking space is enabled
Bool bEnableCarCheck[MAX_CAR_SPACE_CNT]	Whether vehicle detection is enabled
char nWatchRecogCnt;	Is the total number of vehicle detections (3-10)

char nWatchComfirmCnt	Whether vehicle detection is enabled (1-10)
char szReserve[256]	Reserved

5.13 TALPRDEVNETINFO

TAlprDevNetInfo Function Description

```
tYpedef struct _AlprDevNetInfo
{
        char szIP[16];
        char szMask[16];
        char szGateWay[16];
        char szDns1[16];
        char szDns2[16];
        BYTE byMacAddr[ALPR_MACADDR_LEN];
        BYTE byRes[14];
}TAlprDevNetInfo, *PAlprDevNetInfo;
```

Description

Equipment network information

Parameters

Parameter	Description
char szIP[16]	Device IP
char szMask[16]	Subnet mask
char szGateWay[16]	Default gateway

char szDns1[16]	Preferred DNS1
char szDns2[16]	Alternate DNS2
BYTE byMacAddr[ALPR_MACADDR_LEN]	MAC address
BYTE byRes[14]	Reserved field

5.14 ALPR_VIDEO_CONFIG

ALPR_VIDEO_CONFIG Function Description

```
typedef struct
{
            int  nEncFormat;
 unsigned char  nReserve[60];

}ALPR_VIDEO_CONFIG;
```

Description

Definition of video coding mode

Parameters

Parameter	Description
int nEncFormat	Only two are supported for the time being： 0:1080p，3:D1.
unsigned char nReserve[60]	Reserved field

5.15 XRECT

XRect Function Description

```
typedef struct _XRect
{
    int left;
    int right;
    int top;
    int bottom;
    int Width() { return right - left + 1; }
    int Height() { return bottom - top + 1; }
    int Area() {return Width()*Height();}
    void Zero() { left = right = top = bottom = 0; }

    void Union(_XRect &r2)
    {
        if (left > r2.left)left = r2.left;
        if(right < r2.right) right = r2.right;
        if(top > r2.top)top = r2.top;
        if(bottom < r2.bottom) bottom = r2.bottom;
    }

}XRect;
```

Description

Coordinate information, coordinates are currently based on 1024*72 resolution

Parameters

Parameter	Description
int left	left
int right	Reserved field

int top	
int bottom	

5.16 TDEVLICENSE

PDevLicense Function Description

```
Typedef struct _DevLicense
{
     unsigned int nLicenseType;
     unsigned int nLicenseStatus;
     char reserve[64];
}TDevLicense, *PDevLicense;
```

Description

Device license information

Parameters

Parameter	Description
unsigned int nLicenseType	0: old, 1: new
unsigned int nLicenseStatus	License Status 0: Authorization success 1: License file matching is incorrect

	2: License expired 3: Algorithm initialization failed or no license 4: Encrypted chip unlicensed content
`char reserve[64]`	reserved field

5.17 ALPR_CONFIG

ALPR_CONFIG Function Description

```
typedef struct
{
    unsigned char    nInstallDistID : 6;
    unsigned char    bRecogNewEmbassyLic : 1;
    unsigned char    bEnableMotionDetect : 1;
    unsigned char    nInstallDistID;
    unsigned char    nDefaultProvinceID;
    unsigned char    nLaneCount : 7;
    unsigned char    bOuputStableResult : 1;
    unsigned char    nLaneCount;
    unsigned char    nMaxRecogCount;
    unsigned char    nOutputInterval;
    unsigned char    nTriggerMode;
    unsigned char    nRecogMode;
    unsigned char    nUploadMode;
    unsigned char    nOutputMode;
    unsigned char    bRecogEmbassyLic : 1;
    unsigned char    bRecogDoubleLic : 1;
    unsigned char    bRecogSameLicOneTime : 1;
    unsigned char    bRecogOverlayChars : 1;
    unsigned char    bRecogOverlayRect : 1;
    unsigned char    bSamePosFilter : 1;
    unsigned char    bVirtualCoilTrigger : 1;
```

```
    unsigned char     bFilterStrip : 1;
    unsigned char     nCountryCode;
    unsigned char     nAlarmMaxRecogTime;
   unsigned short     nReserve2;
            float     fRecogThreshold;
}ALPR_CONFIG;
```

Parameters

Parameter	Description
`unsigned char nInstallDistID : 6`	By distance ID, the value is tInstallDist
`unsigned char bRecogNewEmbassyLic : 1`	Support for the new embassy license plate
`unsigned char bEnableMotionDetect : 1`	Start motion detection
`unsigned char nInstallDistID`	By distance ID, the value is tInstallDist
`unsigned char nDefaultProvinceID`	The default province, the value is tPLATE_SYM_ID
`unsigned char nLaneCount : 7`	Currently only supports 1
`unsigned char bOuputStableResult : 1`	Stable recognition trigger
`unsigned char nLaneCount`	Currently only supports 1
`unsigned char nMaxRecogCount`	maximum number of recognitions
`unsigned char nOutputInterval`	Same license plate entry interval in seconds
`unsigned char nTriggerMode`	Trigger mode, the value is tTriggerMode
`unsigned char nRecogMode`	Recognition mode, the value is tRecogMode
`unsigned char nUploadMode`	Upload mode, the value is tUploadMode
`unsigned char nOutputMode`	Input mode, the value is tOutputMode
`unsigned char bRecogEmbassyLic : 1`	Is it the embassy license plate

`unsigned char bRecogDoubleLic : 1`	Whether it is a double deck license plate
`unsigned char bRecogSameLicOneTime : 1`	1: The same license plate is only output once, 0 has no such limit
`unsigned char bRecogOverlayChars : 1`	Overlay OSD characters
`unsigned char bRecogOverlayRect : 1`	Overlay OSD Rectangular Box
`unsigned char bSamePosFilter : 1`	Is the license plate output at the same position, 1--no output, 0--output
`unsigned char bVirtualCoilTrigger : 1`	Virtual coil division 1--Use virtual coil, 0--Do not use virtual coil
`unsigned char bFilterStrip : 1`	Whether to enable stripe filtering, 1--enabled, 0--not enabled
`unsigned char nCountryCode`	Country code
`unsigned char nAlarmMaxRecogTime`	Maximum recognition time after ground trigger, in seconds.
`unsigned short nReserve2`	Reserved field
`float fRecogThreshold`	Identifies the threshold, which is [0, 1000.0f]. The smaller the value, the higher the confidence. The default value is 100.0f

5.18 DEV_VER_INFO

DEV_VER_INFO Function Description

```
typedef struct _DEV_VER_INFO
{
    char   Algorithm_Ver[VER_LEN];
    char   Firm_Ver[VER_LEN];
    char   SDK_Ver[VER_LEN];
}DEV_VER_INFO;
```

Description

Device version information definition

Parameters

Parameter	Description
char Algorithm_Ver[VER_LEN]	Algorithm Version.
char Firm_Ver[VER_LEN]	Firmware Version
char SDK_Ver[VER_LEN]	Sdk version

5.19 TSYS_TIME

TSYS_TIME Function Description

```
typedef struct _SYS_TIME
{
    unsigned char     year;
    unsigned char     month;
    unsigned char     day;
    unsigned char     week;
    unsigned char     hour;
    unsigned char     minute;
```

```
    unsigned char     second;
}TSYS_TIME, *PSYS_TIME;
```

Description

System time definition.

Parameters

Parameter	Description
unsigned char year	Year, excluding the current year 2000
unsigned char month	moon
unsigned char day	Day
unsigned char week	Week
unsigned char hour	Hour
unsigned char minute	Minute
unsigned char second	Seconds

5.20GATE_CONFIG

GATE_CONFIG Function Description

```
typedef struct _GATE_CONFIG
{
    int nWorkMode;
    int nLatency;
    int bInverseSignal;
    int nReverse;
```

```
}GATE_CONFIG;
```

Description

This function initializes the SDK Device.

Parameters

Parameter	Description
int nWorkMode	Operation mode of road gate
int nLatency	Delay in operation of gate
int bInverseSignal	Whether or not to reverse the signal. The value is 1 or 0.
int nReverse	Reserved field

5.21 TOFFINEPARAM

TOfflineParam Function Description

```
typedef struct _OfflineParam_
{
            char     sServerIP[16];
    unsigned int     nServerPort;
    unsigned int     nParkID;
   unsigned char     nRecordIsCover;
  unsigned char      nParkInOutFlag;
  unsigned char      nMonthcarAlarmDays;
  unsigned char      nRecognitionAccuracy;
  unsigned char      nRecordMatchAccuracy;
  unsigned char      nMonthCarToTempcarFlag;
```

```
        unsigned char       nMonthCarOpenType;
        unsigned char       nTempCarOpenType;
                float       fMinCharge;
        unsigned char       nTempCarForbiddenFlag;
        unsigned int        nSyncTimeFromMaster;
        unsigned char       nOnlineFlag;
        unsigned char       nOneChannelMode;
         unsigned int       nOneChannelWaitTime;
         unsigned int       nNormalModeWaitTime;
        unsigned char       nMinChargeFlag;
        unsigned char       nDisplayRefreshInterval;
                  char      sPropertyLogo[64];
        unsigned char       nScreenType;
        unsigned char       nZeroQuickOut;
        unsigned char       nReserve[30];
    }TOfflineParam, *POfflineParam;
```

Parameters

Parameter	Description
char sServerIP[16]	Primary server IP address
unsigned int nServerPort	Primary server port
unsigned int nParkID	parking lot number
unsigned char nRecordIsCover	Record (outbound and charge records) coverage, 1: coverage, 0: no coverage
unsigned char nParkInOutFlag	Garage entrance and exit identification, reference enumeration PARK_IN_OUT_E
unsigned char nMonthcarAlarmDays	Fixed vehicle warning days
unsigned char nRecognitionAccuracy	Fixed vehicle matching accuracy, 99: exact match
unsigned char nRecordMatchAccuracy	record matching precision, 99: exact match

unsigned char nMonthCarToTempcarFlag	Enable fixed car to temporary car, reference enumeration ENABLED_TYPE_E
unsigned char nMonthCarOpenType	Fixed car opening mode, reference enumeration OPEN_GATE_TYPE_E
unsigned char nTempCarOpenType	Temporary car opening mode, reference enumeration OPEN_GATE_TYPE_E
float fMinCharge	Minimum charge
unsigned char nTempCarForbiddenFlag	Temporary car is forbidden to enter the venue, 0: Admission is allowed, 1: No entry is allowed
unsigned int nSyncTimeFromMaster	Time point 23:00-->2300 -->0x08fc
unsigned char nOnlineFlag	Whether online mode, reference enumeration ONLINE_MODE_E
unsigned char nOneChannelMode	Whether to enable single channel mode, reference enumeration ENABLED_TYPE_E, 0: not enable single channel, 1: single channel mode
unsigned int nOneChannelWaitTime	Single channel repeat license plate waiting time in seconds
unsigned int nNormalModeWaitTime	Normal mode repeats license plate waiting time in seconds
unsigned char nMinChargeFlag	Whether to enable the minimum charge, refer to the enumeration ENABLED_TYPE_E, 0: not enabled, 1: enabled
unsigned char nDisplayRefreshInterval	The refresh interval of the offline display content, in seconds
char sPropertyLogo[64]	The corporate logo displayed by default on the offline display
unsigned char nScreenType	Display type, reference enumeration ScreenType
unsigned char nZeroQuickOut	Charge 0 yuan, whether to play quickly

unsigned char nReserve[30]	Reserved field

5.22OSD_CTRL

OSD_CTRL Function Description

```
typedef struct _OSD_CTRL
{
    OSD_PARAM     stDateOSD;
    OSD_PARAM     stTimeOSD;
    OSD_PARAM     stWeekOSD;
    OSD_PARAM     stBitrateOSD;
    OSD_PARAM     stTitleOSD;
         char     szTitle[64];
}OSD_CTRL;
```

Parameters

Parameter	Description
OSD_PARAM stDateOSD	Video date display location
OSD_PARAM stTimeOSD	Video time display position
OSD_PARAM stWeekOSD	Video week display position
OSD_PARAM stBitrateOSD	Video resolution display position
OSD_PARAM stTitleOSD	Video title display location
char szTitle[64]	video title

5.23TBASE_PARAM

PBASE_PARAM Function Description

```
typedef struct _BASE_PARAM
{
    unsigned char     bRgnArea : 1;
    unsigned char     bVirtualCoil : 1;
    unsigned char     lprAccuracyMatch : 6;
    unsigned char     nImageSwitchType;
    unsigned char     nFillLightBrightness;
    unsigned char     nAuxiliaryCameraFlag;
              char    sMasterCamIP[16];
    unsigned char     ledCharacterEncoding;
    unsigned char     onlyShowLic;
    unsigned short    nCountryCode;
    unsigned short    nStartTime;
    unsigned short    nEndTime;
    unsigned short    nPhotosensitiveThreshold;
     unsigned char    nOfflineLevel;
     unsigned char    nReserve[19];
     unsigned char    nReserve[30];
    }TBASE_PARAM, *PBASE_PARAM;
```

Parameters

Parameter	Description
`unsigned char bRgnArea : 1`	Whether to display the recognition area, 0: no display, 1: display
`unsigned char bVirtualCoil : 1`	Whether to display virtual coil, 0: no display, 1: display

unsigned char lprAccuracyMatch : 6	License plate matching accuracy
unsigned char nImageSwitchType	Day and night image parameter switching type, reference enumeration ImageSwitchType_E
unsigned char nFillLightBrightness	fill light brightness [0-100], 0: turn off the fill light, other: fill light brightness value
unsigned char nAuxiliaryCameraFlag	Whether it is a secondary camera, reference enumeration ENABLED_TYPE_E, 0: host, 1: auxiliary camera
char sMasterCamIP[16]	main camera IP address
unsigned char ledCharacterEncoding	Display character encoding, reference enumeration CharacterEncoding_E
unsigned char onlyShowLic	Does the display only show license plates, 0: other, 1: only send license plates
unsigned short nCountryCode	country code, reference enumeration COUNTRY_CODE_E
unsigned short nStartTime	Switch to daytime image parameters at 6:00-->600
unsigned short nEndTime	Switch to night image parameters at 18:00-->1800
unsigned short nPhotosensitiveThreshold	Photosensitive threshold [0-1024], which is greater than this value to switch to the night parameter, otherwise switch to daytime parameter
unsigned char nOfflineLevel	offline level, reference enumeration OFFLINE_LEVEL_E

unsigned char nReserve[19]	Reserved field
unsigned char nReserve[30]	Reserved field

5.24TSDCARDCAPACITY

PSDCardCapacity Function Description

```
Typedef struct _SDCardCapacity
{
        unsigned int      nTotalCapacity;
        unsigned int      nRemainCapacity;
    }TSDCardCapacity, *PSDCardCapacity;
```

Description

SD card capacity information definition

Parameters

Parameter	Description
unsigned int nTotalCapacity	Total capacity in megabytes
unsigned int nRemainCapacity	Remaining space, in MB

5.25PLATE_INFO_EXT

PLATE_INFO_EXT Function Description

```
typedef struct _PlateInfoExt
{
        char   szPlate[16];
        XDate  fromDate;
        XDate  toDate;
         char  szName[64];
unsiged long   long CardNo;
unsiged char   nPlateColor;
unsigned char  Reserve[31];
}PLATE_INFO_EXT;
```

Description

The list of license plate structures used in the country

Parameters

Parameter	Description
char szPlate[16]	license plate
XDate fromDate	date commenced
XDate toDate	End date
Char szName[64]	Name of owner
unsiged long long CardNo	License plate number
unsiged char nPlateColor	License plate color,0--black,20--green,30--bule,50--yellow,255--white
unsigned char Reserve[31]	Reserved field

5.26CCOLOR_IMAGE_I

cCOLOR_IMAGE_I Function Description

```
typedef struct
{
    cIMAGE_DATA_FORMAT    enPixelFormat;
                  int     Width;
                  int     Height;
        unsigned char *   pu8BGRData;
} cCOLOR_IMAGE_I;
```

Parameters

Parameter	Description
cIMAGE_DATA_FORMAT enPixelFormat	Yuv image pixel format
int Width	The width of the image
int Height	Image height
unsigned char * pu8BGRData	BGR data buffer for images

5.27CIMAGE_DATA_FORMAT

Function

```
typedef enum cIMAGE_DATA_FORMAT
    {
        E_YUV_PLANAR_422 = 0x01,
        E_YUV_SEMIPLANAR_422,
        E_YUV_SEMIPLANAR_420,
        E_YUV_PLANAR_420,
```

```
    E_BGR_DATA,
    E_YUV_SEMIPLANAR_420_NV21,
    E_YUV_PLANAR_420_YV12,
    E_MAX_IMG_FORMAT,
};
```

Description

Yuv image pixel formatis.

Parameters

Parameter	Description
E_YUV_PLANAR_422 = 0x01	YYY UUU VVV
E_YUV_SEMIPLANAR_422	[Y0 Y1 Y2] [U0V0 U1V1 U2V2]
E_YUV_SEMIPLANAR_420	[Y0 Y1 /Y2 Y3] [U0V0 U1V1...]
E_YUV_PLANAR_420	[Y0 Y1 ...] [U0U1..] [V0V1...]
E_BGR_DATA	BGR BGR BGR...
E_YUV_SEMIPLANAR_420_NV21	[Y0 Y1 /Y2 Y3] [V0U0 V1U1]
E_YUV_PLANAR_420_YV12	[Y0 Y1 ...] [V0V1...] [U0U1..]
E_MAX_IMG_FORMAT	

5.28COLORRESULT

stColorResult Function Description

```
typedef struct _tagColorResult
{
        String     label;
          char     label[24];
```

```
            int     index;
          float     score;
  unsigned char     u8Color;
  unsigned char     u8Reserve[3];
} stColorResult;
```

Description

Color comparison results.

Parameters

Parameter	Description
String label	
char label[24]	
int inde	
float score	The credibility of the vehicle target is within the range of [0,100]. The greater the value, the higher the credibility
unsigned char u8Color	License plate color, please refer to the value ePlateColor
unsigned char u8Reserve[3]	Reserved field

5.29LOGORESULT

stLogoResult Function Description

```
typedef struct _tagLogoResult
{
     char     szLogoDescr[24];
```

```
            int     s32LetterCount;
           char     label[24];
            int     index;
          float     score;
          XRect     rtPos;
  unsigned char     u8Color;
  unsigned char     u8Reserve[3];
      } stLogoResult;
```

Parameters

Parameter	Description
char szLogoDescr[24]	Car logo information reservation
int s32LetterCount	Number of characters
char label[24]	String label
int index	
float score	The credibility of the car logo, the value range [0,100]. The higher the value, the higher the credibility
XRect rtPos	Logo location information
unsigned char u8Color	License plate color, please refer to ePlateColor for value
unsigned char u8Reserve[3]	Reserved field

5.30NETWORKPARAM

PNetworkParam Function Description

```
typedef struct _NetworkParam
    {
        int    NetInterface;
       char    DVRIP[16];
```

```
    char    DVRIPMask[16];
    char    GatewayIpAddr[16];
    char    MACAddr[20];
    char    DnsServer1IpAddr[16];
    char    DnsServer2IpAddr[16];
    short   DvrPort;
}NetworkParam, *PNetworkParam;
```

Parameters

Parameter	Description
int NetInterface	Network interface. 1-10 M Base-T; 2-10 Base-T - full duplex; 3-100 Base-T X; 4-100 M - full duplex; 5-10 M/100 M - adaptation;
char DVRIP[16]	IP address
char DVRIPMask[16]	Subnet mask
char GatewayIpAddr[16]	Gateway address
char MACAddr[20]	Physical address
char DnsServer1IpAddr[16]	DNS address 1
char DnsServer2IpAddr[16]	DNS address 2
short DvrPort	Communication port, default 8000

5.31 VIDEOPARAM

PVideoParam Function Description

```
typedef struct _VideoParam
{
  int Mode;
  int DayStart;
  int DayEnd;
}VideoParam, *PVideoParam;
```

Description

Video parameter interface.

Parameters

Parameter	Description
int Mode	Mode, 0- External Trigger; 1-Automatic; 2- Color; 3- black and white; 4- time period
int DayStart	Daytime period start time (in seconds)=hour * 3600 + min * 60 + sec)
int DayEnd	Daytime period start time (in seconds)=hour * 3600 + min * 60 + sec)

5.32DEVICEPARAM

PVDeviceParam Function Description

```
typedef struct _DeviceParam
{

        char   softwareVer[24];
        char   hardwareVer[24];
        char   serialNum[24];
        char   dvrName[24];
```

```
    unsigned char   dvrType;
    unsigned char   alarmInPortNum;
             char   reverved[2];
       WebApiVer    webApiVer;
}DeviceParam, *PDeviceParam;
```

Parameters

Parameter	Description
char softwareVer[24]	Software version number (master version)
char hardwareVer[24]	Hardware version number
char serialNum[24]	serial number
char dvrName[24]	DVR name
unsigned char dvrType	DVR type. **1** - DVR, **2** - NVR, **3** - DVS/IPC
unsigned char alarmInPortNum	Number of DVR Alarm Inputs
char reverved[2]	Reserved
WebApiVer webApiVer	Web API Information

5.33WEBAPIVER

WebApiVer Function Description

```
typedef struct _WebApiVer
{

    char standard[24];
    char build[24];

}WebApiVer, *PWebApiVer;
```

Description

Web API Information.

Parameters

Parameter	Description
char standard[24]	Version number
char build[24]	Compilation date

5.34DEVSTATUS

DevStatus Function Description

```
typedef struct _DevStatus
    {
        AuxCameraStatus auxCameraStatusAarry[MAXAUXCAMERANUM];
        int         HeartBeatStatus;
        unsigned char GateSwitchStatus;
        unsigned char GateRunStatus;
        unsigned char FeelStatus;
        unsigned char Reserve1;
        unsigned short PhotosensitiveValue;
       unsigned char AuxInStatus[MAX_AUX_IN_NUM];
       unsigned char GateOpenStatus;
        unsigned char GateCloseStatus;
        unsigned char PrinterStatus;
        unsigned char TimeoutAlarm:1;
        unsigned char ScanCodeStatus:1;
        unsigned char Reserve2 : 6;
       unsigned char Reserve[7];
    }TDevStatus, *PDevStatus;
```

Description

Device status definition.

Parameters

Parameter	Description
AuxCameraStatus auxCameraStatusAarry[MAXAUXCAMERANUM]	Secondary camera status
int HeartBeatStatus	Heartbeat state
unsigned char GateSwitchStatus	Gate switch status, refer to enumeration GATE_SWITCH_STATUS_E
unsigned char GateRunStatus	Gate operation status, refer to enumeration GATE_RUN_STATUS_E
unsigned char FeelStatus	Earth-sensing status, refer to enumeration FEEL_STATUS_E
unsigned char Reserve1	Reserved field
unsigned short PhotosensitiveValue	Photosensitive signal
unsigned char AuxInStatus[MAX_AUX_IN_NUM]	Level state of auxiliary input
unsigned char GateOpenStatus	Level state of gate opening in place
unsigned char GateCloseStatus	Level state with gate closed in place
unsigned char PrinterStatus	Printer status, refer to enumeration Printer_Status_E
unsigned char TimeoutAlarm:1	After the receipt is printed, the parking space will leave and call the police after the scheduled time has passed.
unsigned char ScanCodeStatus:1	Scanning device status.

	0: no device detected, 1: device detected
`unsigned char Reserve2 : 6`	Reserved field
`unsigned char Reserve[7]`	Reserved field

6 Enumeration definition

6.1 Macro Function Description

Macro Name	Description
#define ALPR_SDK_VER "1.0.1.23"	SDK Version Number
#define ENCRYPT_LEN 33	Encryption Length
#define SERIAL_NUMBER_LEN 27	Serial Number Length
#define LIC_LEN 900	License Content Length
#define VER_LEN 33	Version Number Length
#define MAXAUXCAMERANUM (2)	Number of Secondary Cameras
#define MAX_WHITELIST_COUNT(1024)	Maximum number of supported License Plate Whitelists
#define MAX_BLACKLIST_COUNT(1024)	Maximum number of supported License Plate Blacklists
#define MAX_FIXEDLIST_COUNT (8192)	The maximum number of fixed license plates currently supported
#define FUN_LOG_IN_ID (1)	Number of equipment admission record
#define FUN_GET_FUN_LIST (1)	Get the function list of the Device

6.2 Type Definition Description

Type	Description
COUNTRY_CODE_E	typedef enum _APP_COUNTRY_CODE_E COUNTRY_CODE_E

Type	Description
	National coding definition
DATA_TABLE_TYPE_E	typedef enum _DATA_TABLE_TYPE_E DATA_TABLE_TYPE_E Data table definition
DEV_TYPE_E	typedef enum _DEV_TYPE_E DEV_TYPE_E Device type definition
EChargeMode	typedef enum _E_CHARGE_MODE_ EChargeMode Definition of temporary vehicle charging Mode
EExposureMode	typedef enum _E_EXPOSURE_MODE_ EExposureMode Exposure mode definition
EExposureRunMode	typedef enum _E_EXPOSURE_RUN_MODE_ EExposureRunMode Exposure operation mode definition
ELightMode	typedef enum _E_LIGHT_MODE_ ELightMode Fill light mode definition
EModelParamType	typedef enum _E_MODEL_PARAM_TYPE_ EModelParamType Model parameter type definition
ENABLED_TYPE_E	typedef enum _ENABLED_TYPE_E ENABLED_TYPE_E Startup mode definition
FEEL_STATUS_E	typedef enum _FEEL_STATUS_E FEEL_STATUS_E Ground state
GATE_RUN_STATUS_E	typedef enum _GATE_RUN_STATUS_E GATE_RUN_STATUS_E Gate operation status
GATE_SWITCH_STATUS_E	typedef enum _GATE_SWITCH_STATUS_E GATE_SWITCH_STATUS_E

Type	Description
	Switch state of the gate
ImageSwitchType_E	typedef enum _ImageSwitchType_E_ ImageSwitchType_E ISP switching method definition
N_COUNTRY_CODE	typedef enum _N_COUNTRY_CODE N_COUNTRY_CODE
OFFLINE_LEVEL_E	typedef enum _OFFLINE_LEVEL_E OFFLINE_LEVEL_E Offline definition
ONLINE_MODE_E	typedef enum _ONLINE_MODE_E ONLINE_MODE_E Online offline definition
OPEN_GATE_TYPE_E	typedef enum _OPEN_GATE_TYPE_E OPEN_GATE_TYPE_E Gate opening method definition
PARK_IN_OUT_E	typedef enum _PARK_IN_OUT_E PARK_IN_OUT_E Vehicle entrance and exit definition
ParkEvent	typedef enum _PARK_EVENT_ ParkEvent Event type definition
SDCard_OpType_E	typedef enum _SDCard_OpType_E SDCard_OpType_E SD card status information definition

6.3 Enumeration Type Description

6.3.1 Country Code Description

APP_COUNTRY_CODE_E

Code Value	Code	Description
CC_US	1	United States
CC_KZ	7	Kazakhstan
CC_EG	20	Egypt
CC_ZA	27	South Africa
CC_GR	30	Greece
CC_NL	31	Netherlands
CC_BE	32	Belgium
CC_FR	33	France
CC_ES	34	Spain
CC_IT	39	Italy
CC_UK	44	United Kingdom
CC_RO	40	Romania
CC_CH	41	Switzerland
CC_AT	43	Austria
CC_DK	45	Denmark
CC_SE	46	Sweden
CC_NO	47	Norway
CC_PL	48	Poland
CC_PE	51	Peru
CC_MX	52	Mexico

Code Value	Code	Description
CC_CU	53	Cuba
CC_AR	54	Argentina
CC_BR	55	Brazil
CC_CL	56	Chile
CC_CO	57	Colombia
CC_VE	58	Venezuela
CC_MY	60	Malaysia
CC_AU	61	Australia
CC_ID	62	Indonesia
CC_PH	63	Philippines
CC_NZ	64	New Zealand
CC_SG	65	Singapore
CC_TH	66	Thailand
CC_JP	81	Japan
CC_KR	82	Korea
CC_VN	84	Vietnam
CC_CN	86	China
CC_TR	90	Turkey
CC_IND	91	India
CC_PK	92	Pakistan

Code Value	Code	Description
CC_AF	93	Afghanistan
CC_LK	94	Sri Lanka
CC_MM	95	Myanmar
CC_IR	98	Iran
CC_MA	210	Morocco
CC_DZ	213	Algeria
CC_TN	216	Tunisia
CC_LY	218	Libya
CC_GM	220	Gambia
CC_SN	221	Senegal
CC_MR	222	Mauritania
CC_ML	223	Mali
CC_GN	224	Guinea
CC_CI	225	Cote d'Ivoire
CC_BF	226	Burkina Faso
CC_NE	227	Niger
CC_TG	228	Togo
CC_BJ	229	Benin
CC_MU	230	Mauritius
CC_LR	231	Liberia

Code Value	Code	Description
CC_SL	231	Sierra Leone
CC_GH	233	Ghana
CC_NG	234	Nigeria
CC_TD	235	Chad
CC_CF	236	Central African Republic
CC_CM	237	Cameroon
CC_CV	238	Cape Verde
CC_ST	239	Sao Tome and Principe
CC_GQ	240	Equatorial Guinea
CC_GA	241	Gabon
CC_CD	242	Congo
CC_AO	244	Angola
CC_GW	245	Guinea-Bissau
CC_SD	249	Sudan
CC_RW	250	Rwanda
CC_ET	251	Ethiopia
CC_SO	252	Somalia
CC_DJ	253	Kyrgyzstan
CC_KE	254	Kenya
CC_TZ	255	Tanzania

Code Value	Code	Description
CC_UG	256	Uganda
CC_BI	257	Burundi
CC_MZ	258	Mozambique
CC_ZM	260	Zambia
CC_MG	261	Madagascar
CC_RE	262	Reunion
CC_ZW	263	Zimbabwe
CC_NA	264	Namibia
CC_MW	265	Malawi
CC_LS	266	Lesotho
CC_BW	267	Botswana
CC_SZ	268	Swaziland
CC_KM	269	Comoros
CC_ER	291	Eritrea
CC_AW	297	Aruba
CC_HU	336	Hungary
CC_DE	349	Germany
CC_PT	351	Portugal
CC_LU	352	Luxembourg
CC_IE	353	Ireland

Code Value	Code	Description
CC_IS	354	Iceland
CC_AL	355	Albania
CC_MT	356	Malta
CC_CYP	357	Cyprus
CC_FI	358	Finland
CC_BG	359	Bulgaria
CC_LT	370	Lithuania
CC_LV	371	Latvia
CC_EE	372	Estonia
CC_MD	373	Moldova
CC_AM	374	Armenia
CC_BY	375	Belarus
CC_AD	376	Andorra
CC_UA	380	Ukraine
CC_RS	381	Serbia
CC_ME	382	Montenegro
CC_HR	385	Croatia
CC_SI	386	Slovenia
CC_BA	387	Bosnia and Herzegovina
CC_MK	389	Macedonia

Code Value	Code	Description
CC_VA	396	Vatican
CC_CZ	420	Czech Republic
CC_SK	421	Slovakia
CC_BZ	501	Belize
CC_GT	502	Guatemala
CC_SV	503	Salvador
CC_HN	504	Honduras
CC_NI	505	Nicaragua
CC_CR	506	Costa Rica
CC_PA	507	Panama
CC_HT	509	Haiti
CC_GP	590	Guadeloupe
CC_BO	591	Bolivia
CC_GY	592	Guyana
CC_EC	593	Ecuador
CC_GF	594	French Guiana
CC_PY	595	Paraguay
CC_MQ	596	Martinique
CC_SR	597	Suriname
CC_UY	598	Uruguay

Code Value	Code	Description
CC_AN	599	Netherlands Antilles
CC_GU	671	Guam
CC_TL	670	East Timor
CC_BN	673	Brunei Darussalam
CC_NR	674	Nauru
CC_PG	675	Papua New Guinea
CC_TO	676	Tonga
CC_SB	677	Solomon Islands
CC_VU	678	Vanuatu
CC_FJ	679	Fiji
CC_CK	682	Island
CC_WS	685	Samoa
CC_KI	686	Kiribati
CC_NC	687	New Caledonia
CC_TV	688	Tuvalu
CC_PF	689	French Polynesia
CC_FM	691	Micronesia
CC_MH	692	Marshall Islands
CC_KP	850	Korea
CC_HK	852	Hong Kong

Code Value	Code	Description
CC_MO	853	Macao
CC_KH	855	Cambodia
CC_LA	856	Laos
CC_BD	880	Bangladesh
CC_TW	886	Taiwan
CC_MV	961	Maldives
CC_LB	962	Lebanon
CC_JO	963	Jordan
CC_SY	964	Syria
CC_IQ	965	Iraq
CC_KW	966	Kuwait
CC_SA	967	Saudi Arabia
CC_YE	968	Yemen
CC_OM	989	Oman
CC_PS	970	Palestine
CC_AE	972	United Arab Emirates
CC_IL	973	Israel
CC_BH	974	Bahrain
CC_QA	975	Qatar
CC_BT	976	Bhutan

Code Value	Code	Description
CC_MN	977	Mongolia
CC_NP	978	Nepal
CC_TJ	992	Tajikistan
CC_TM	993	Turkmenistan
CC_AZ	994	Azerbaijan
CC_GE	995	Georgia
CC_KG	996	Djibouti
CC_UZ	998	Uzbekistan
CC_BB	1809	Barbados

6.3.2 Data Table Definition

_DATA_TABLE_TYPE_E

Enumeration Value	Code	Description
FCT_TEMP_CAR_CHARGE_RULE	3	Temporary vehicle charging rules（TempCarChargeRule）
FCT_GEN_CHARGE_RULE	4	General charge rule（GenChargeRule）
FCT_RECORD_IN_OUT_LOG	5	Record Table (RecordLog)
FCT_CHARGE_LOG	7	Fee Schedule (ChargeLog)
FCT_MODEL_PARAM	8	Model parameters (ModelParam)

6.3.3 Mode of Payment for Temporary Car

enum_E_CHARGE_MODE_

Temporary car charging mode definition.

Enumerate value	Code	Description
E_CHARGE_MODE_TIMEZONE	1	Charge by time period
E_CHARGE_MODE_PERIOD	2	Charge by cycle

6.3.4 License Plate Recognition Strategy Definition

enum tOutputMode

License plate recognition strategy definition.

Enumerate value	Code	Description
OutputAll	0	All output
OutputFilterGoAwayCar	1	Filtering away vehicles
OutputFilterComeCloseCar	2	Filtering away vehicles

6.3.5 License Plate recognition Mode Definition

enum tRecogMode

License plate recognition mode definition.

Enumerate value	Code	Description

RecogBySingleFrame	0	Single frame mode
RecogByMultiFrame	1	Multi-frame preferred mode

6.3.6 License Plate Recognition Trigger Mode Definition

enum tTriggerMode

License plate recognition trigger mode definition

Enumerate value	Code	Description
TriggerByVideo	0	Video trigger
TriggerBySensor	1	Ground trigger
TriggerByAll	2	Mixed trigger

6.3.7 Startup Method Definition

enum_ENABLED_TYPE_E

Startup mode definition

Enumerate value	Code	Description
NOT_ENABLED	0	Not enabled or not displayed
ENABLED	1	Enable or display

6.3.8 Ground State Definition

enum_FEEL_STATUS_E

Ground state

Enumerate value	Code	Description
FEEL_NONE	0	Landless
FEEL_HAVE	1	Sense of land
FEEL_STATUS_NO	255	no status

6.3.9 Gate Operation Status

enum_GATE_RUN_STATUS_E

Gate operation status

Enumerate value	Code	Description
GATE_NORMAL	0	Normal operation of the gate
GATE_OPEN_TIMEOUT	1	Gate open timeout
GATE_OFF_TIMEOUT	2	Gate closure timeout
GATE_RUN_NONE	255	no status

6.3.10 Switch Status

enum_GATE_SWITCH_STATUS_E

Switch state of the gate

Enumerate value	Code	Description

Enumerate value	Code	Description
GATE_CLOSED	0	The gate is closed
GATE_CLOSING	1	Gate gate
GATE_OPENING	2	Gate opening
GATE_OPENED	3	The gate is open
GATE_STOP	4	Stop at halfway
GATE_SWITCH_NONE	255	no status

6.3.11 ISP Switching Method Definition

enum_ImageSwitchType_E_

ISP switching method definition

Enumerate value	Code	Description
PHOTOSENSITIVE_THRESHOLD	0	Photosensitive threshold switching
PERIOD	1	Time period switching

6.3.12 Country Code

enum_N_COUNTRY_CODE

COUNTRY_CODE

Enumerate value	Code	Description
LOC_CODE_ESP	101	Europe Spain
LOC_CODE_PRT	102	Portugal
LOC_CODE_FRA	103	France
LOC_CODE_ITA	104	Italy
LOC_CODE_GBR	105	United Kingdom
LOC_CODE_GRC	106	Greece
LOC_CODE_IRL	107	Ireland (ROI)
LOC_CODE_DEU	108	Germany
LOC_CODE_AND	109	Andorra
LOC_CODE_POL	110	Poland
LOC_CODE_BGR	111	Bulgaria
LOC_CODE_NLD	112	Netherlands
LOC_CODE_EST	113	Estonia
LOC_CODE_BIH	114	Bosnia and Herzegovina
LOC_CODE_ROU	115	Romania
LOC_CODE_BEL	116	Belgium
LOC_CODE_NOR	117	Norway
LOC_CODE_DNK	118	Denmark
LOC_CODE_SWE	119	Sweden
LOC_CODE_FIN	123	Finland

Enumerate value	Code	Description
LOC_CODE_GBZ	121	Gibraltar
LOC_CODE_CHE	122	Switzerland
LOC_CODE_AUT	124	Austria
LOC_CODE_SVN	127	Slovenia
LOC_CODE_HUN	128	Hungary
LOC_CODE_CHL	201	South America and Central America Chile
LOC_CODE_COL	201	Colombia
LOC_CODE_BRA	203	Brazil
LOC_CODE_ARG	204	Argentina
LOC_CODE_MEX	205	Mexico
LOC_CODE_ECU	206	Ecuador
LOC_CODE_VEN	207	Venezuela
LOC_CODE_PER	208	Peru
LOC_CODE_SLV	209	El Salvador
LOC_CODE_BOL	210	Bolivia
LOC_CODE_URY	211	Uruguay
LOC_CODE_GTM	213	Guatemala
LOC_CODE_PAN	215	Panama
LOC_CODE_PRY	216	Paraguay
LOC_CODE_CRI	217	Costa Rica

Enumerate value	Code	Description
LOC_CODE_DOM	218	Dominican Republic
LOC_CODE_HND	219	Honduras
LOC_CODE_NIC	220	Nicaragua
LOC_CODE_BMU	221	Bermuda
LOC_CODE_TTO	222	Trinidad and Tobago
LOC_CODE_RUS	301	Asia Russia
LOC_CODE_TUR	302	Turkey
LOC_CODE_VNM	303	Vietnam
LOC_CODE_IDN	304	Indonesia
LOC_CODE_PHL	305	Philippines
LOC_CODE_MYS	306	Malaysia
LOC_CODE_SGP	307	Singapore
LOC_CODE_ISR	308	Israel
LOC_CODE_LBN	309	Lebanon
LOC_CODE_HKG	310	Hong-Kong
LOC_CODE_MAC	311	Macau
LOC_CODE_IND	312	India
LOC_CODE_TWN	313	Taiwan
LOC_CODE_BHR	314	Bahrain
LOC_CODE_ZAF	401	Africa South Africa

Enumerate value	Code	Description
LOC_CODE_MAR	402	Morocco
LOC_CODE_TUN	403	Tunisia
LOC_CODE_AGO	404	Angola
LOC_CODE_SEN	405	Senegal
LOC_CODE_NGA	406	Nigeria
LOC_CODE_UGA	407	Uganda
LOC_CODE_USA_WA	501	North America (mainly the United States and Canada) Washington (USA)
LOC_CODE_USA_FL	502	Florida (USA)
LOC_CODE_USA_MS	503	Mississippi (USA)
LOC_CODE_USA_NY	504	New York (USA)
LOC_CODE_USA_TX	505	Texas (USA)
LOC_CODE_USA_MA	506	Massachusetts (USA)
LOC_CODE_USA_IL	507	Illinois (USA)
LOC_CODE_USA_CA	508	California (USA)
LOC_CODE_USA_OK	508	Oklahoma (USA)
LOC_CODE_USA_LA	510	Louisiana (USA)
LOC_CODE_USA_NM	511	New Mexico (USA)
LOC_CODE_USA_AR	512	Arkansas (USA)
LOC_CODE_USA_NJ	513	New Jersey (USA)

Enumerate value	Code	Description
LOC_CODE_USA_DC	514	District of Columbia (USA)
LOC_CODE_USA_IN	515	Indiana (USA)
LOC_CODE_USA_ME	516	Maine (USA)
LOC_CODE_USA_NE	517	Nebraska (USA)
LOC_CODE_USA_MO	518	Missouri (USA)
LOC_CODE_USA_NV	519	Nevada (USA)
LOC_CODE_USA_MI	520	Michigan (USA)
LOC_CODE_USA_VA	521	Virginia (USA)
LOC_CODE_USA_NC	522	North Carolina (USA)
LOC_CODE_USA_GA	523	Georgia (USA)
LOC_CODE_USA_PA	524	Pennsylvania (USA)
LOC_CODE_USA_AK	525	Alaska (USA)
LOC_CODE_USA_MN	526	Minnesota (USA)
LOC_CODE_USA_KY	527	Kentucky (USA)
LOC_CODE_USA_ND	528	North Dakota (USA)
LOC_CODE_USA_OH	529	Ohio (USA)
LOC_CODE_USA_CO	530	Colorado (USA)
LOC_CODE_USA_AZ	531	Arizona (USA)
LOC_CODE_USA_UT	532	Utah (USA)
LOC_CODE_USA_WV	533	West Virginia (USA)

Enumerate value	Code	Description
LOC_CODE_USA_CT	534	Connecticut (USA)
LOC_CODE_USA_HI	535	Hawaii (USA)
LOC_CODE_USA_IA	536	Iowa (USA)
LOC_CODE_USA_KS	537	Kansas (USA)
LOC_CODE_USA_SC	538	South Carolina (USA)
LOC_CODE_USA_OR	539	Oregon (USA)
LOC_CODE_USA_MT	540	Montana (USA)
LOC_CODE_USA_AL	541	Alabama (USA)
LOC_CODE_USA_TN	542	Tennessee (USA)
LOC_CODE_USA_RI	543	Rhode Island (USA)
LOC_CODE_USA_NH	544	New Hampshire (USA)
LOC_CODE_USA_DE	545	Delaware (USA)
LOC_CODE_USA_VT	546	Vermont (USA)
LOC_CODE_USA_WI	547	Wisconsin (USA)
LOC_CODE_USA_SD	548	South Dakota (USA)
LOC_CODE_USA_MD	549	Maryland (USA)
LOC_CODE_USA_WY	550	Wyoming (USA)
LOC_CODE_USA_ID	551	Idaho (USA)
LOC_CODE_CAN	556	Canada

6.3.13 Offline Mode Definition

enum_OFFLINE_LEVEL_E

Offline mode definition

Enumerate Value	Code	Description
OFFLINE_LEVEL_NONE	0 x 00	Turn off offline
OFFLINE_LEVEL_SD_SAVE_LOG	0 x 01	Save the record to the SD card
OFFLINE_LEVEL_FLASH_SAVE_LOG	0 x 02	Save the record to FLASH
OFFLINE_LEVEL_SD_SAVE_JPG	0 x 04	Save the picture to the SD card
OFFLINE_LEVEL_ALL	0 x FF	Turn on offline all features

6.3.14 Online and Offline Definition

enum_ONLINE_MODE_E

Online offline definition

Enumerate Value	Code	Description
OFFLINE_MODE	0	Offline
ONLINE_MODE	1	Online

6.3.15 Gate Opening Method Definition

enum_OPEN_GATE_TYPE_E

Gate opening method definition

Enumerate Value	Code	Description
MANUAL_OPEN	0	Manual opening
AUTO_OPEN	1	Automatic opening

6.3.16 Event Type Definition

enum_PARK_EVENT_

Event type definition

Enumerate value	Code	Description
E_DEV_EVENT_START	100	Event begins
E_WHITE_LIST_CAR	101	whitelist
E_BLACK_LIST_CAR_REFUSED	102	Blacklist is forbidden to enter
E_TEMP_CAR	103	Temporary car
E_TEMP_CAR_REFUSED	104	Temporary car is forbidden to enter
E_WHITE_TO_TEMP_CAR	105	Whitelist to temporary car
E_WHITE_EXPIRED	106	Whitelist expired
E_TEMP_CAR_NO_MATCH_IN	107	Temporary car can't match
E_TEMP_CAR_MIN_CHARGE	108	Temporary car minimum charge
E_FIXED_LIST_CAR	109	Fixed car list
E_FIXED_TO_TEMP_CAR	110	Fixed car list to temporary car

E_FIXED_EXPIRED	111	Fixed car expired

6.3.17 Entrance and Exit Definition

enum_PARK_IN_OUT_E

Vehicle entrance and exit definition

Enumerate Value	Code	Description
PARK_IN	0	Garage entrance
PAKR_OUT	1	Depot exit

6.3.18 SD Card Status Information Definition

enum_SDCard_OpType_E

SD card status information definition

Enumerate value	Code	Description
SD_MOUNT	0	Mount SD card
SD_UMOUNT	1	Uninstall SD card
SD_FORMAT	2	Format SD card
SD_CAPACITY	3	Get SD card capacity

6.3.19 Country Code Definition for License Plate Recognition

enum CountryCode_E

Country Code Definition for License Plate Recognition

Enumerate value	Code	Description
SPAIN	0	Spain
USA	1	United States
MEXICO	2	Mexico
THAILAND	3	Thailand
ARGENTINA	4	Argentina
SOUTH_AFRICA	5	South Africa
CHILE	6	Chile
SAUDI	7	Saudi Arabia
COLOMBIA	8	Colombia
VIETNAM	9	Vietnam
PERU	10	Peru
TURKEY	11	Turkey
UAE	12	United Arab Emirates
BRAZIL	13	Brazil
MONGOLIA	14	Mongolia
CHINA	225	China

6.3.20 Client Type Definition

enum EAPIClientType

Client type definition

Enumerate value	Code	Description
E_CLIENT_NORMAL	0	Ordinary client
E_CLIENT_DEV_OCX	1	Web client
E_CLIENT_DEMO	2	Demo client

6.3.21 List of Supported Features

enum eFunList

List of features supported by the device

Enumerate value	Code	Description
FUN_LOG_ADD_IN_INFO	0	Admission records increase admission information
FUN_CARTEAM_MODE	1	Fleet mode
FUN_PLATE_LIST_EXT	2	License plate list information extension name, etc.
FUN_DEV_RECOG_CAR_INFO	3	Equipment identification vehicle characteristics

FUN_UPLOAD_OFFLINE_RECORD	4	Device actively uploads offline records
FUN_UPLOAD_EXT_GATE_VOICE_SCREEN	5	Support extended control gate, voice, display command (command with jpg id)
FUN_SCREEN_TYPE	6	Current device display type
FUN_CAR_SPACE	7	Parking camera function
FUN_NEURAL_ALGORITHM	254	Whether to support European algorithms

6.3.22 Identified License Plate Color Definition

enum ePlateColor

Recognized license plate color definition

Enumerate value	Code	Description
COLOR_BLACK	0	Black
COLOR_GREEN	20	Green
COLOR_BLUE	30	Blue
COLOR_RED	40	Red
COLOR_YELLOW	50	Yellow
COLOR_WHITE	255	white

6.3.23 License Plate List Type Definition

enum ePlateListType

License plate list type definition

Enumerate value	Code	Description
PL_TYPE_WHITE	0	whitelist
PL_TYPE_BLACK	1	Blacklist
PL_TYPE_FIXED	2	Fixed car

6.3.24 Gate Work Definition

GateWorkMode

enum GateWorkMode

Road gate work mode definition

Enumerate value	Code	Description
mOpenByRecog	0	Identification begins
mColoseByBlackList	1	Blacklist only does not open
mOpenByWhiteList	2	Whitelist only
mOpenByManual	3	Upper computer is turned on

6.3.25 Image Upload Mode

enum tUploadMode

Image uploading method definition after license plate recognition

Enumerate value	Code	Description
UploadByFTP	0	FTP upload
UploadBySDK	1	SDK upload
UploadByJSON	2	JSON upload
UploadByUDisk	3	U disk storage

6.3.26 Camera Recognition Computation

enum tInstallDist

Distance definition when the camera recognizes the license plate

Enumerate value	Code	Description
DistLT35	0	Less than 3.5 meters
DistLT45	1	3.5 meters - 4.5 meters
DistLT5	2	4.5 meters - 5 meters
DistLT6	3	5 meters - 6 meters
DistMT6	4	More than 6 meters

6.3.27 Scroll Mode Description

enum ScreenMoveDirection

Display scroll mode definition

Enumerate value	Code	Description
SCREEN_MOVE_UP	1	Data scrolling up
SCREEN_MOVE_DOWN	2	Data scrolling down
SCREEN_MOVE_LEFT	3	Data scrolls to the left
SCREEN_MOVE_RIGHT	4	Data scrolls to the right

6.3.28 Display Color Definition

enum ScreenShowColor

Display color definition

Enumerate value	Code	Description
SCREEN_COLOR_RED	0	Display data color is red
SCREEN_COLOR_GREEN	1	Display data color is green

6.3.29 Car Type Definition

typedef enum zkta_logo_enum

Car type

Enumerate value	Cod	Note
ZKTAA_LOGO_OTHRE_0	0	other
ZKTA_LOGO_OTHRE_1	1	Unknown (known category but not shown as
ZKTA_LOGO_DS	2	DS
ZKTA_LOGO_JAC	3	JAC

Enumerate value	Cod	Note
ZKTA_LOGO_Jeep	4	Jeep
ZKTA_LOGO_MG	5	MG
ZKTA_LOGO_MINI	6	MINI
ZKTA_LOGO_CNHTC	7	Cnhtc
ZKTA_LOGO_SANLING	8	Mitsubishi
ZKTA_LOGO_SHANGHAIHUIZHON	9	Shanghai Huizhong
ZKTA_LOGO_DONGNAN	10	southeast
ZKTA_LOGO_DONGFENG	11	Dongfeng
ZKTA_LOGO_ZHONGXIQICHE	12	Zhongxing Automobile
ZKTA_LOGO_ZHONGHUA	13	ZhongHua
ZKTA_LOGO_FENGTIAN	14	Toyota
ZKTA_LOGO_FENGTIANMARK	15	Toyota mark
ZKTA_LOGO_FENGTIANHUANGGU	16	Toyota crown
ZKTA_LOGO_WUSHILING	17	Isuzu
ZKTA_LOGO_WULING	18	Wuling
ZKTA_LOGO_ZHONGTAI	19	Zhongtai
ZKTA_LOGO_YIWEIKE	20	Iveco
ZKTA_LOGO_HX	21	Like Hx
ZKTA_LOGO_OTHER_2	22	Like a minivan_unknown
ZKTA_LOGO_OTHER_3	23	Like a triangle_unknown
ZKTA_LOGO_OTHER_4	24	Like a triangle plug a circle
ZKTA_LOGO_OTHER_5	25	Like big S_unknown
ZKTA_LOGO_HENGTONG	26	Hengtong bus
ZKTA_LOGO_ZHENGTENG	27	Zheng Teng
ZKTA_LOGO_WANGPAI	28	Ace card
ZKTA_LOGO_NANJUN	29	Nan Jun
ZKTA_LOGO_OTHER_6	30	Like a circle,unknown
ZKTA_LOGO_HUAPU	31	Wap
ZKTA_LOGO_YOUYIPAI	32	Friendship card
ZKTA_LOGO_SHAHUANGSHITONG	33	Three Rings Ten Pass
ZKTA_LOGO_YUTONG	34	Yutong
ZKTA_LOGO_CHANGCHENGJINDIE	35	Great Wall Jindir
ZKTA_LOGO_QUANQIUYING	36	Global Hawk
ZKTA_LOGO_KAIDILAKE	37	cadillac

Enumerate value	Cod	Note
ZKTA_LOGO_BIEKE	38	Buick
ZKTA_LOGO_LIFAN	39	Lifan
ZKTA_LOGO_BEIJINGFUTIAN	40	Beijing Futian
ZKTA_LOGO_BEIQI	41	Beiqi
ZKTA_LOGO_BEIQIZIZAO	42	Beiqi Manufacturing
ZKTA_LOGO_BEIQIHUANSU	43	Beiqi Magic Speed
ZKTA_LOGO_SHUANGHUANG	44	Double Ring
ZKTA_LOGO_JILI	45	Geely
ZKTA_LOGO_JILIDIHAO	46	Geely Emgrand
ZKTA_LOGO_QICHEN	47	Qichen
ZKTA_LOGO_HAFU	48	Haver
ZKTA_LOGO_HAFEI	49	Hafei
ZKTA_LOGO_XIALI	50	Xiali
ZKTA_LOGO_DAZHONG	51	Volkswagen
ZKTA_LOGO_QIRUI	52	Chery
ZKTA_LOGO_BENTENG	53	Pentium
ZKTA_LOGO_BENCHI	54	Benz
ZKTA_LOGO_AODI	55	Audi
ZKTA_LOGO_WEIWAN	56	Weiwang
ZKTA_LOGO_BAOWO	57	Baowo
ZKTA_LOGO_BAOMA	58	BMW
ZKTA_LOGO_BAOJUN	59	Bao Jun
ZKTA_LOGO_CHUANQIYEMA	60	Chuanqi hourse
ZKTA_LOGO_GUANGQI	61	Guangzhou Automobile
ZKTA_LOGO_AKIRUI	62	Karry
ZKTA_LOGO_SIMING	63	Si Ming
ZKTA_LOGO_SIBALU	64	Subaru
ZKTA_LOGO_SIKEDA	65	Skoda
ZKTA_LOGO_RICHAGN	66	Nissan
ZKTA_LOGO_CHANGHE	67	Chang he
ZKTA_LOGO_BENTIAN	68	Honda
ZKTA_LOGO_BIAOZHI	69	Peugeot
ZKTA_LOGO_OUBAO	70	Opel
ZKTA_LOGO_BIYADI	71	BYD

Enumerate value	Cod	Note
ZKTA_LOGO_JIANGHUAI	72	Jianghuai
ZKTA_LOGO_JIANGLING	73	Jiangling
ZKTA_LOGO_JIANGLINGJIASHENG	74	Jiang Ling Yusheng
ZKTA_LOGO_WOERWO	75	Volvo
ZKTA_LOGO_HAIMA	76	Ha/ma
ZKTA_LOGO_HIAMAZHENGZHOU	77	Hai Ma Zhengzhou
ZKTA_LOGO_MASHALADI	78	Maserati
ZKTA_LOGO_XIANDAI	79	modern
ZKTA_LOGO_FUTE	80	Ford
ZKTA_LOGO_FUTIAN	81	Futian
ZKTA_LOGO_FUDI	82	Fudi
ZKTA_LOGO_HONGQI		83
ZKTA_LOGO_NAZHIJIE	84	Na Zhijie
ZKTA_LOGO_YANGCHENGQICHE		
ZKTA_LOGO_YINGLUNQICHE		
ZKTA_LOGO_YINGFEINIDI		87
ZKTA_LOGO_RONGWEI		88
ZKTA_LOGO_YAFEITE	89	Fiat
ZKTA_LOGO_OUGE	90	Acura
ZKTA_LOGO_QIYA	91	Kia
ZKTA_LOGO_YUEJIN	92	Leap forward
ZKTA_LOGO_LUHU	93	Land-rover
ZKTA_LOGO_JINLUKECHE	94	Jinlv Bus
ZKTA_LOGO_JINBEI	95	Golden Cup
ZKTA_LOGO_JINNIU	96	Golden Bull
ZKTA_LOGO_JINLONGKECHE	97	Golden Dragon Bus
ZKTA_LOGO_LINGMU	98	Suzuk
ZKTA_LOGO_CHANGCHENG	99	Great Wall
ZKTA_LOGO_CHANGANSHANGYO	100	Changan Commercial
ZKTA_LOGO_CHANGANXIN	101	Changan New
ZKTA_LOGO_CHANGANJIU	102	Changan Old
ZKTA_LOGO_LUFEN G	103	Landwind
ZKTA_LOGO_XUEFULAN	104	Chevrolet
ZKTA_LOGO_XUETIELONG	105	Citroen

Enumerate value	Cod	Note
ZKTA_LOGO_LEIKESASI	106	Lexus
ZKTA_LOGO_LEILUO	107	Renault
ZKTA_LOGO_MAZIDA	108	Mazda
ZKTA_LOGO_HUANGHAI		109

6.3.30 Vehicle Color Definition

typedef enum zkta_color_enum

Color definition

Enumerate value	Code	Note
ZKTA_COLOR_White	0	white
ZKTA_COLOR_Gray	1	gray
ZKTA_COLOR_Golden	2	Gold
ZKTA_COLOR_Pink	3	Pink
ZKTA_COLOR_Red	4	red
ZKTA_COLOR_Purple	5	purple
ZKTA_COLOR_Green	6	green
ZKTA_COLOR_Blue	7	blue
ZKTA_COLOR_Brown	8	brown
ZKTA_COLOR_Black	9	black
ZKTA_COLOR_Silvery	10	Silver
ZKTA_COLOR_Yellow	11	yellow
ZKTA_COLOR_Other	12	other colors

6.3.31 Vehicle Type Definition

typedef enum zkta_type_enum

Vehicle Type

Enumerate value	Code	Note
ZKTA_TYPE_HOUCHUOHEZHEKECHE	0	Truck or bus (without car cover)
ZKTA_TYPE_XIAOXINGCHE	1	Small car (with car cover)

6.3.32 Camera Device Type Definition

typedef enum _LPR_DEV_TYPE_E

Camera device type definition

Enumerate value	Code	Note
LPR_DEV_UNKNOWN	0	Unknown device
LPR_DEV_JL	1	Dragon license plate equipment
LPR_DEV_GZ	2	Perceived license plate equipment
LPR_DEV_GZ_CAR_SPACE	3	Perceived parking space equipment

6.3.33 Parking Status

typedef enum _CS_EVENT_TYPE_E

Parking status

Parking status enumeration value	Code	Note
CS_EVENT_DISABLE	1	Parking space disabled
CS_EVENT_EMPTY	2	Empty parking space

CS_EVENT_HAVE_CAR	3	There is a car in the parking space
CS_EVENT_1_OCCUPY_N	4	Bicycles occupy more parking spaces
CS_EVENT_NO_LIC_CAR	5	Unlicensed car

6.3.34 Screen type definition

typedef enum ScreenType

Enumerate value	Code	Note
SIGNLE_COLOR_SCREEN	0	Monochrome screen
DOUBLE_COLOR_TWO_LINES_SCREEN		Two-color dual-line screen
DOUBLE_COLOR_FOUR_LINES_SCREEN		Two-color four-line screen

6.3.35 Other Record Definition

Maximum length of the IP address string:

#define MAX_EVENT_ADDR_LEN 16

#define MAX_CAR_NUMBER_LEN 32

#define MAX_PHOTO_PATH 256

Entry record definition

typedef struct _RecordInOutLog_

unsigned intID	Number
unsigned intTimeSecond	Time
char CarNumber[MAX_CAR_NUMBER_LEN]	License plate number
unsigned long long CardNo	Card number
unsigned char InOutState	Channel status, 0: In, 1: Out
char EventAddr[MAX_EVENT_ADDR_LEN]	Channel name, default is IP address
unsigned intEventType	101: Whitelist 102: Blacklist 103: Temporary car 104: Temporary car ban 105: Whitelist to temporary car 106: Whitelist expired 107: Temporary car can't match 108: Minimum charge for temporary vehicles 109: Fixed car 110: Fixed car to temporary car 111: Fixed car expired
char PhotoPath[MAX_PHOTO_PATH]	Image path, default string time, expandable time + license plate, etc.

unsigned char IsUploadFlag	iMage html IsUploadFlag.png 0[0000]: [Device Processing Business Mode] [Local Record] [Non-Repeat Entry Record] 1[0001]: [Device Processing Service Mode] [Local Record] [Non-Repeat Entry Record] [Send Event to Connected Online Monitoring Software] 2[0010]: [software processing business mode] [local record] [non-repeating admission record] [send license plate to software processing] 4[0100]: [Device Processing Business Mode] [Local Record] [Repeat Entry Record] 5[0101]: [Device Processing Business Mode] [Local Record] [Repeat Entry Record] [Send Event to Connected Online Monitoring Software] 6[0110]: [Software Processing Business Mode] [Local Record] [Repeat Entry Record] [Send License Plate to Software Processing]

	8[1000]: [Device processing business mode] [from other network synchronization records] [non-repeating admission records] 9[1001]: [Device processing business mode] [from other network synchronization records] [non-repeating admission records] [send events to connected online monitoring software] 10[1010]: [Software processing business mode] [from other network synchronization records] [non-repeating admission records] [send license plate to software processing] 12[1100]: [Device Processing Business Mode] [From Other Network Synchronization Records] [Repeat Entry Record] 13[1101]: [Device Processing Business Mode] [From Other Network Synchronization Records] [Repeat Entry Record] [Send Event to Connected Online Monitoring Software] 14[1110]: [Software processing business mode] [from other network synchronization

	records] [repeated admission record] [send license plate to software processing]
unsigned char IsLeave	Whether to play, 0: no appearance, 1: already played (for admission records)
#ifdef FUN_LOG_IN_ID	
unsigned int InID	Admission ID
char InAddr[MAX_EVENT_ADDR_LEN]	Admission IP address
#endif	

7 Appendixes

Appendix 1 - Error Code

Error type definition	Error code	Note
OS_NoErr = (OS_Error)0	0	Successful operation
OS_BadURLFormat=(OS_Error)-1000	-1000	error code
OS_NotEnoughSpace	-999	Not enough space
OS_CannotOpenSocket	-998	Failed to create socket
OS_CannotConnect	-997	Device disconnected
OS_InvalidArgument	-996	Invalid data
OS_CannotSendData	-995	Failed to send data
OS_TRYAGAIN	-994	The server is too busy, please try again
OS_OperationErr	-993	Function call error
OS_CannotOpenChn	-992	Failed to open channel
OS_VideoNotOpen	-991	Video channel is not open
OS_ConnectExist	-990	Connection already exists
OS_PullCannotConn	-989	Pull cannot be connected
OS_NotEnoughMem	-988	Not enough storage
OS_FileReadErr	-987	File read error
OS_FileWriteErr	-986	File write error
OS_CannotRecvData	-985	Read data timeout

Error type definition	Error code	Note
OS_SetTimeFail	-984	Set time failed
OS_DevCannotSupport	-983	The device does not support this command or interface.
OS_ExceedDevCapcity	-982	Exceeding device storage capacity
OS_PlaySoundFail	-981	Play sound failed
OS_StarVoiceFail	-980	Start the intercom failed
OS_SendVoiceFail	-979	Sending voice failed
OS_InputAudioFail	-978	Input Audio failed
OS_DevNotSupportP2P	-977	The device does not support P2P.
OS_JniError	-976	JNI error
OS_JsonParseError	-975	Json parsing error
OS_InitThirdSDKFaild	-974	Calling a third-party SDK failed
OS_GetHandleIDFail	-973	HandleID has been used up.
OS_DateFormatErr	-972	Error in date format, illegal date format
OS_InvalidDate	-971	Invalid date (start date is greater than end date)
OS_InvalidData	-970	invalid data
OS_VideoChannelException	-969	Video channel anomaly

Appendix 2 – RS485 Communication Protocol

RS485 communication protocol between camera and two-color screen (voice and display)

The content in this document is mainly used for the 485-communication protocol between the camera and the display and voice and follows the original embedded master-slave RS485 protocol standard.

Note

Display and Voice Device Address	66
Host Address	0
Display Screen	
The First row of the Data Type	35
The Second row of the Data Type	36

Communication Data Format Definition

	Start Tag	Target Address	Serial Number	Command Word	Total length of Data	Parameters and Data	CRC16 Check
Content	0xaa						

Length	1 Byte	1 Byte	2 Bytes	1 Byte	2 Bytes	N Byte	2 Bytes

Start Tag: Fixed to 0xaa, binary is 1011010, on the circuit as a neat peak, valley and valley waveform, easy to distinguish from interference data.

Serial Number: When the host computer sends data, it automatically increments by 1, and the lower computer responds with the same serial number.

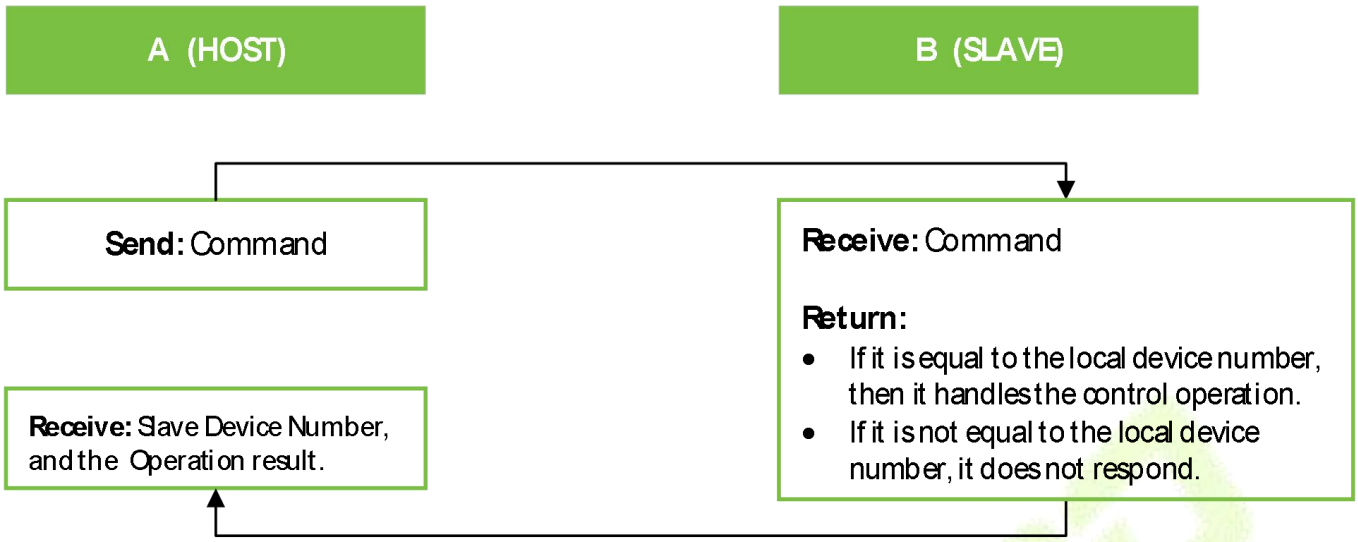
Target Address: Refers to the device ID number received by this data stream (that is, the 485 address of the slave). In principle, the ID number of all devices in a network or system cannot be repeated; the address of the broadcast command is off, and all received are executed.

Total Length of Data: The sum of the parameters and data length of this communication。

CRC Check: The result value of the CRC16 calculation is performed on the content from the "target address" to the end of "parameters and data".

Communication Procedure

Host - Slave Control:



Communication Command

Instructions to send the Real-Time Data.

	Start Tag	Target Address	Serial Number	Command Word	Total Length of Data	Parameters and Data	CRC16 Check
Content	0xaa	66	1	174			
Length	1 Byte	1 Byte	2 Bytes	1 Byte	2 Bytes	N Byte	2 Bytes

Data 1 Type	Data 1 Length	Data 1 Content		Data N Type	Data N Length	Data N Content
1 Byte	2 Byte	Data 1 Length		1 Byte	2 Bytes	Data N Length

Command Format:

Cmd=174，datalen=N，data[0]=data

The Parameters and the Data format are as follows,

	Data Type	Data Length	Data	Data Type	Data Length	Data
Content	First row of data (35)	N	DATA	Second row of data (36)	N	DATA
Length	1 Byte	2 Bytes	N Byte	1 Byte	2 Bytes	N Byte

Remarks:

Data Content: The first byte, the color of the data (0: red, 1: green), followed by the content that needs to be displayed.

Command – 176: Command that Interrupts the Last Voice Broadcast.

	Start Tag	Target Address	Serial Number	Command Word	Total Length of Data	Parameters and Data	CRC16 Check
Content	0xaa	66	1	176			
Length	1 Byte	1 Byte	2 Bytes	1 Byte	2 Bytes	N Byte	2 Bytes

Command Format:

Cmd=176，datalen=1，data[0]=

Command – 177: Command to send Voice Data.

	Start Tag	Target Address	Serial Number	Command Word	Total Length of Data	Parameters and Data	CRC16 Check
Content	0xaa	66	1	177		data	
Length	1 Byte	1 Byte	2 Bytes	1 Byte	2 Bytes	N Byte	2 Byte

Command Format:

Cmd=177，datalen=N，data= voice data

Command – 179: Version Number

	Start Tag	Target Address	Serial Number	Command Word	Total Length of Data	Parameters and Data	CRC16 Check
content	0xaa	66	1	179		data	
length	1 Byte	1 Byte	2 Bytes	1 Byte	2 Bytes	N Byte	2 Bytes

Command Format:

Cmd=179，datalen=N，data=0

Command – 200: Successfully executed Instructions

	Start Tag	Target Address	Serial Number	Command Word	Total Length of Data	Parameters and Data	CRC16 Check

Content	0xaa	0	1	200			
Length	1 Byte	1 Byte	2 Bytes	1 Byte	2 Bytes	N Byte	2 Bytes

Command Format:

Cmd=200，datalen=1，data=0

Target address: the address of the display screen or voice

The slave receives the command to obtain the version and replies the version information to the host.

Cmd=200，datalen= Version Number length，data= Version Number.

Command – 201: Instruction Execution Failed

	Start Tag	Destination Address	Serial number	Command Word	Total length of data	Data	CRC16 Check
Content	0xaa	0	1	201	1	Error code	
Length	1 Byte	1 Byte	2 Bytes	1 Byte	2 Bytes	1 Byte	2 Byte

Command Format:

Cmd=201，datalen=1，data[0]=Error code。

Destination address: The address of the Display or Voice.

Communication Command – Error Code

Error Value	Description
-1	Command failed to send
-2	Command did not respond
-3	Insufficient cache required
-4	Decompression failed
-5	The length of the read data is incorrect.
-6	The length of the decompression does not match the expected length
-7	Command repetition
-8	Connection not authorized
-9	CRC check failed
-10	Data API cannot be parsed
-11	Parameter error
-12	Command execution error
-13	Without this command
-14	Communication password error

Appendix 3 – Voice Segment Description

Voice Segment Description - Foreign

Voice Code	Description
157	One
158	Tow
159	Three
160	Four
161	Five
162	Six
163	Seven
164	Eight
165	Nine
166	Ten
171	Welcome
172	No extra parking spaces
173	No entrance record
174	Valid period
176	Day
179	See you next time
181	Please contact toll office

184	Please contact the administrator
185	Expired car
188	Unauthorized car

ZKTeco Industrial Park, No. 26, 188 Industrial Road,

Tangxia Town, Dongguan, China.

Phone : +86 769 - 82109991

Fax : +86 755 - 89602394