New SDK API

No. ：XX-XX-XX-XX

Date created：2019-01-18

Version No. ：0.0.0.28

Logs Changed

Date Changed	User	Description	Version	
2018-01-19	HU	Create	0.0.0.1	
2018-03-28	Dai	Add client connunication mode and 2.24--2.38 I/O	0.0.0.14	
2018-04-03	Dai	Improve 2.37 ,2.38I/O 2.35change data format	0.0.0.15	
2018-05-18	Dai	Add Face device I/O,	0.0.0.16	
2018-05-25	Dai	Add 2.39--2.58 I/O	0.0.0.17	
2018-06-07	Dai	Add 2.59--2.60 I/O	0.0.0.18	
2018-07-09	Dai	Add customize function and improve demo	0.0.0.19	
2018-07-23	Dai	Add 2.61--2.77 I/O	0.0.0.21	
2018-08-23	Dai	Increase The Customize function	0.0.0.23	
2018-09-20	Dai	Update the function for automatic add setup port of the server	0.0.0.25	
2018-11-29	Dai	Update the user ID up to 8 digit	0.0.0.26	
2019-01-18	Dai	Add the Log manage	0.0.0.28	

Content

1 Overview

1.1 Introduction

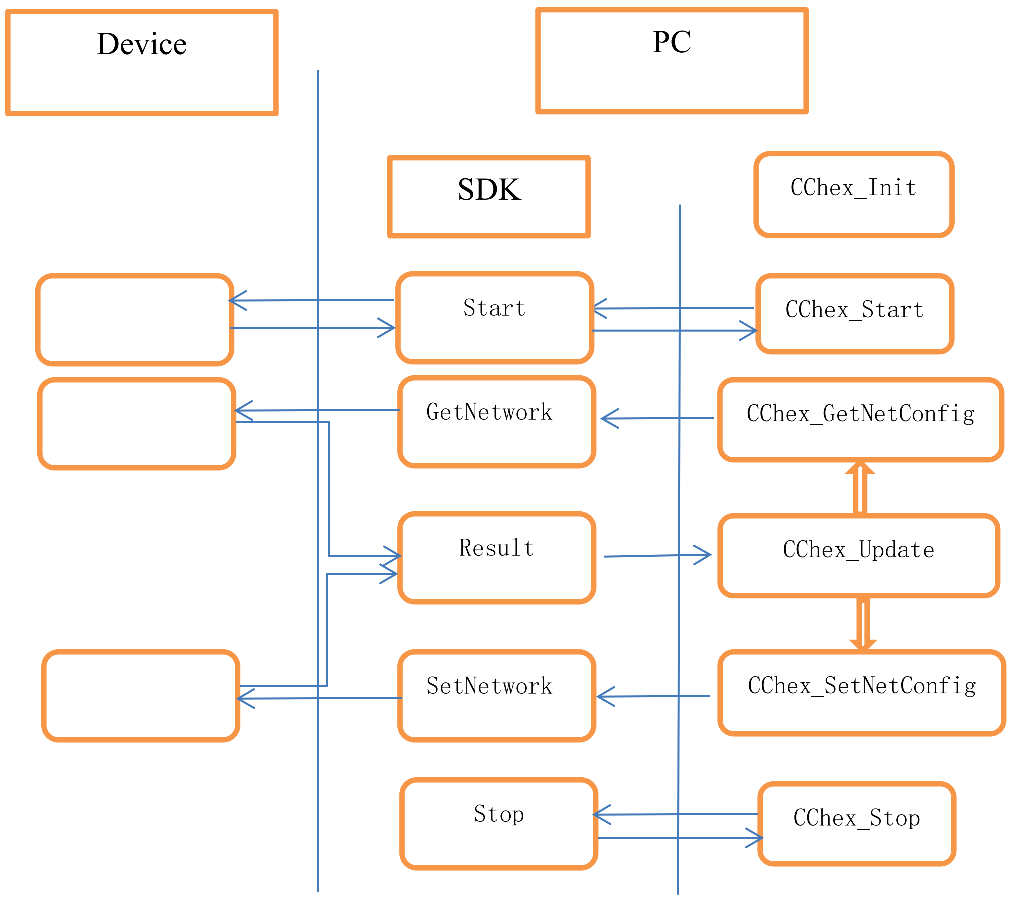
This API document mainly defines the development interface of the Anviz SDK, which provides reference for developers to connect and operate related devices.

This document is mainly related to the software development engineers, software testing engineers and project managers.

1.2 Overview of functions

This API document mainly provides some interfaces of the client/server to the PC end, getting the parameters of the time attendance/access control, and setting the configuration and parameters of the time attendance/access control device. In order to support multiple devices, the interfaces between queries and settings exist asynchronously. SDK is compiled into the form of a dynamic library.

1.3 System logic structure



1.4 Brief introduction of interface functions

```
uint CChex_Version();

void CChex_Init();

IntPtr CChex_Start();

void CChex_Stop(IntPtr CchexHandle);

int CChex_Update(IntPtr CchexHandle, int[] DevIdx, int[] Type, IntPtr Buff, int Len);


int CChex_GetNetConfig(IntPtr CchexHandle, int DevIdx);

int CChex_SetNetConfig(IntPtr CchexHandle, int DevIdx, ref CCHEX_NETCFG_INFO_STRU Config);
int CChex_MsgGetByIdx(IntPtr CchexHandle, int DevIdx, byte Idx);

int CChex_MsgDelByIdx(IntPtr CchexHandle, int DevIdx, byte Idx);

int CChex_MsgAddNew(IntPtr CchexHandle, int DevIdx, byte[] Data, int Len);

int CChex_MsgGetAllHead(IntPtr CchexHandle, int DevIdx);


int CChex_RebootDevice(IntPtr CchexHandle, int DevIdx);

int CChex_SetTime(IntPtr CchexHandle, int DevIdx, int Year, int Month, int Day, int Hour, int
Min, int Sec);

int CChex_GetSNConfig(IntPtr CchexHandle, int DevIdx);


int CChex_DownloadAllRecords(IntPtr CchexHandle, int DevIdx);

int CChex_DeleteRecordInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX_DEL_RECORD_INFO_STRU
Config);


int CChex_GetBasicConfigInfo(IntPtr CchexHandle, int DevIdx);

int CChex_SetBasicConfigInfo(IntPtr CchexHandle, int DevIdx, ref
CCHEX_SET_BASIC_CFG_INFO_STRU Config);


int CChex_ListPersonInfo(IntPtr CchexHandle, int DevIdx);

int CChex_ModifyPersonInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX_RET_PERSON_INFO_STRU
personlist, byte person_num);

int CChex_DeletePersonInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX_DEL_PERSON_INFO_STRU
Config);


int CChex_DownloadFingerPrint(IntPtr CchexHandle, int DevIdx, byte[] EmployeeId, byte
FingerIdx);

int CChex_UploadFingerPrint(IntPtr CchexHandle, int DevIdx, byte[] EmployeeId, byte FingerIdx,
byte[] FingerData, int DataLen);



int CChex_GetTime(IntPtr CchexHandle, int DevIdx);//(24)


int CChex_InitUserArea(IntPtr CchexHandle, int DevIdx);

int CChex_InitSystem(IntPtr CchexHandle, int DevIdx);


int CChex_GetBasicConfigInfo2(IntPtr CchexHandle, int DevIdx);

int CChex_SetBasicConfigInfo2(IntPtr CchexHandle, int DevIdx, ref
CCHEX_GET_BASIC_CFG_INFO2_STRU_EXT_INF config);


int CChex_GetPeriodTime(IntPtr CchexHandle, int DevIdx, byte SerialNumbe);

int CChex_SetPeriodTime(IntPtr CchexHandle, int DevIdx, ref
CCHEX_SET_PERIOD_TIME_STRU_EXT_INF config);


int CChex_GetTeamInfo(IntPtr CchexHandle, int DevIdx, byte TeamNumbe);

int CChex_SetTeamInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX_SET_TEAM_INFO_STRU_EXT_INF
config);


int CCHex_AddFingerprintOnline(IntPtr CchexHandle, int DevIdx, ref
CCHEX_ADD_FINGERPRINT_ONLINE_STRU_EXT_INF Param);


int CCHex_ForcedUnlock(IntPtr CchexHandle, int DevIdx, ref CCHEX_FORCED_UNLOCK_STRU_EXT_INF
Param);


int CCHex_Udp_Search_Dev(IntPtr CchexHandle);
int CCHex_Udp_Set_Dev_Config(IntPtr CchexHandle, ref CCHEX_UDP_SET_DEV_CONFIG_STRU_EXT_INF
config);


int CCHex_ClientConnect(IntPtr CchexHandle, byte[] Ip, int Port);

int CCHex_ClientDisconnect(IntPtr CchexHandle, int DevIdx);
```

1.5 Message Type

```
CChex_Update(anviz_handle, dev_idx, Type, pBuff, len);
```

You need to loop the query to see if there is an asynchronous return, and if you have a different Type of data parsing, get the data you want.public enum MsgType:int

```
    {

        CCHEX_RET_RECORD_INFO_TYPE = 1

        , CCHEX_RET_DEV_LOGIN_TYPE

        , CCHEX_RET_DEV_LOGOUT_TYPE

        , CCHEX_RET_DLFINGERPRT_TYPE = 4

        , CCHEX_RET_ULFINGERPRT_TYPE

        , CCHEX_RET_MODIFY_PERSON_INFO_TYPE = 8

        , CCHEX_RET_LIST_PERSON_INFO_TYPE

        , CCHEX_RET_MSGGETBYIDX_INFO_TYPE = 12

        , CCHEX_RET_MSGGETBYIDX_UNICODE_INFO_TYPE

        , CCHEX_RET_MSGADDNEW_INFO_TYPE

        , CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

        , CCHEX_RET_MSGDELBYIDX_INFO_TYPE // 15

        , CCHEX_RET_MSGGETALLHEAD_INFO_TYPE

        , CCHEX_RET_REBOOT_TYPE

        , CCHEX_RET_DEV_STATUS_TYPE
```

```
, CCHEX_RET_MSGGETALLHEADUNICODE_INFO_TYPE

, CCHEX_RET_SETTIME_TYPE // 20

, CCHEX_RET_UPLOADFILE_TYPE

, CCHEX_RET_GETNETCFG_TYPE

, CCHEX_RET_SETNETCFG_TYPE

, CCHEX_RET_GET_SN_TYPE  //24

, CCHEX_RET_GET_BASIC_CFG_TYPE = 29

, CCHEX_RET_SET_BASIC_CFG_TYPE

, CCHEX_RET_DEL_PERSON_INFO_TYPE = 31

, CCHEX_RET_DEL_RECORD_OR_FLAG_INFO_TYPE = 33


 ,CCHEX_RET_GET_BASIC_CFG2_TYPE = 37

 ,CCHEX_RET_SET_BASIC_CFG2_TYPE = 38

 ,CCHEX_RET_GETTIME_TYPE = 39

 ,CCHEX_RET_INIT_USER_AREA_TYPE = 40

 ,CCHEX_RET_INIT_SYSTEM_TYPE = 41

 ,CCHEX_RET_GET_PERIOD_TIME_TYPE = 42

 ,CCHEX_RET_SET_PERIOD_TIME_TYPE = 43

 ,CCHEX_RET_GET_TEAM_INFO_TYPE = 44

 ,CCHEX_RET_SET_TEAM_INFO_TYPE = 45

 ,CCHEX_RET_ADD_FINGERPRINT_ONLINE_TYPE = 46

 ,CCHEX_RET_FORCED_UNLOCK_TYPE = 47

 ,CCHEX_RET_UDP_SEARCH_DEV_TYPE = 48

 ,CCHEX_RET_UDP_SET_DEV_CONFIG_TYPE = 49
```

```
,CCHEX_RET_GET_INFOMATION_CODE_TYPE = 50

,CCHEX_RET_SET_INFOMATION_CODE_TYPE = 51

,CCHEX_RET_GET_BELL_INFO_TYPE = 52

,CCHEX_RET_SET_BELL_INFO_TYPE = 53

,CCHEX_RET_LIVE_SEND_ATTENDANCE_TYPE = 54

,CCHEX_RET_GET_USER_ATTENDANCE_STATUS_TYPE = 55

,CCHEX_RET_SET_USER_ATTENDANCE_STATUS_TYPE = 56

,CCHEX_RET_CLEAR_ADMINISTRAT_FLAG_TYPE = 57

,CCHEX_RET_GET_SPECIAL_STATUS_TYPE = 58

,CCHEX_RET_GET_ADMIN_CARD_PWD_TYPE = 59

,CCHEX_RET_SET_ADMIN_CARD_PWD_TYPE = 60

,CCHEX_RET_GET_DST_PARAM_TYPE = 61

,CCHEX_RET_SET_DST_PARAM_TYPE = 62

,CCHEX_RET_GET_DEV_EXT_INFO_TYPE = 63

,CCHEX_RET_SET_DEV_EXT_INFO_TYPE = 64

,CCHEX_RET_GET_BASIC_CFG3_TYPE = 65

,CCHEX_RET_SET_BASIC_CFG3_TYPE = 66

,CCHEX_RET_CONNECTION_AUTHENTICATION_TYPE = 67

,CCHEX_RET_GET_RECORD_NUMBER_TYPE = 68

,CCHEX_RET_GET_RECORD_BY_EMPLOYEE_TIME_TYPE = 69

,CCHEX_RET_GET_RECORD_INFO_STATUS_TYPE = 70

,CCHEX_RET_GET_NEW_RECORD_INFO_TYPE = 71


,CCHEX_RET_ULEMPLOYEE2W2_INFO_TYPE            = 72,

CCHEX_RET_GET_BASIC_CFG5_TYPE               = 73,
        CCHEX_RET_SET_BASIC_CFG5_TYPE             = 74,

        CCHEX_RET_GET_CARD_ID_TYPE                = 75,

        CCHEX_RET_SET_DEV_CURRENT_STATUS_TYPE     = 76,

        CCHEX_RET_GET_URL_TYPE                    = 77,

        CCHEX_RET_SET_URL_TYPE                    = 78,

        CCHEX_RET_GET_STATUS_SWITCH_TYPE          = 79,

        CCHEX_RET_SET_STATUS_SWITCH_TYPE          = 80,

        CCHEX_RET_GET_STATUS_SWITCH_EXT_TYPE      = 81,

        CCHEX_RET_SET_STATUS_SWITCH_EXT_TYPE      = 82,

        CCHEX_RET_UPDATEFILE_STATUS_TYPE          = 83,


        ,CCHEX_RET_CLINECT_CONNECT_TYPE = 200

        ,CCHEX_RET_CLINECT_DISCONNECT_TYPE = 201

    };
```

1.6 Setup label and print setup Description

Generic version:

Default Setup: After download new records mark as:0(NO) Print LOG:0(NO)

W2 Version：

Default Setup: After download new records mark as:1(Yes) Print LOG:0(NO)

Customize(S) Version：

Default Setup: After download new records mark as:1(Yes) Print LOG:0(NO)

Customize（F）Version：：

Default Setup: After download new records mark as:0(NO) Print LOG:0(NO)

Change Configuration path:

1：Defining the configuration file：tc-b_new_sdk.ini

```
[LogFile]

LogFile = 1             1：yes 0：no

[SetRecordFlag]

SetRecordFlag = 1 1：yes 0：no

[Debug]

Debug = 0
```

2:Call the function interface：void CChex_SetSdkConfig(void *CchexHandle,int SetRecordflag,int SetLogFile);

Parameter：

int SetRecordflag	1：yes 0：no
int SetLogFile	1：yes 0：no

Priority level:

High： Call the function interface

High： Defining the configuration file

Low： Defining Setup

Notice：When without any “Call the function interface” and “Defining the configuration file as “Defining Setup”

2 Description interface

2.1 CChex_Version

2.1.1 Description functions

【Function】Query SDK Version.

2.1.2 Request

【Mode】`uint CChex_Version()`

【Parameter】None

2.1.3 Response

【Return value】Get SDK version, Returns the version number of an integer number.

2.1.4 Sample

```
int sdk_ver = CChex_Version();
```

2.1.5 Notice

2.2 CChex_Init

2.2.1 Description functions

【Function】 The SDK function initializes and initializes the socket application interface.

2.2.2 Request

【Mode】`void CChex_Init()`

【Parameter】None

2.2.3 Response

【Return value】None

2.2.4 Sample

```
CChex_Init();
```

2.2.5 Notice

None

2.3 CChex_Start

2.3.1 Description functions

【Function】 Start the SDK, allocate space, and establish communication with the Anviz device.

2.3.2 Request

【Mode】`IntPtr CChex_Start()`

【Parameter】None

2.3.3 Response

【Return value】Return the handle of SDK.

2.3.4 Sample

```
IntPtr sdk_handle = CChex_Start();
```

```
if (sdk_handle != null)
{
    MessageBox.Show("Startup OK");
}  else
{
     MessageBox.Show("Startup errors,Please restart the program.");
}
```

2.3.5 Notice

1. Make sure that CChex_Init is initialized before run.

2.4 CChex_Stop

2.4.1 Description functions

【Function】Stop the SDK, release the space, close the socket chain and so on.

2.4.2 Request

【Mode】`void CChex_Stop(IntPtr CchexHandle)`

【Parameter】`CchexHandle,CChex_Start successfully create the handle`

2.4.3 Response

【Return value】None

2.4.4 Sample

```
CChex_stop();
```

2.4.5 Notice

1. Make sure that CChex_Start has been started successfully before run.

2.5 CChex_Update

2.5.1 Description functions

【Function】The Return value get or set asynchronously.。

2.5.2 Request

【Mode】`int CChex_Update(IntPtr CchexHandle, int[] DevIdx, int[] Type, IntPtr Buff, int Len);`

【Parameter】CchexHandle,CChex_Start successfully create the handle，Input[Parameter];

DevIdx,Device index returned asynchronously,Output[Parameter];

Buff,Returned data section,Output [Parameter];

Len,Returns the length of the data section,Input[Parameter];

2.5.3 Response

【Return value】> 0：Successful asynchronous ; Return value == 0：Invalid Return value;

< 0：buffer space is not enough，based on Return value, then re-apply the space.

2.5.4 Sample

```
 int ret = CChex_Update(anviz_handle, dev_idx, Type, pBuff, len);
if (ret > 0)
{
       switch(Type)
       {
       case CCHEX_RET_DEV_LOGIN_TYPE: // Connect the Anviz device successfully and return the
Device ID, Device IP address, Software version, Device type and software version, etc.
```

```
                break;
            case CCHEX_RET_GET_BASIC_CFG_TYPE: // Some basic information about the Anviz device,
such as administrator password, firmware version, volume, etc.
                break;
            case CCHEX_RET_DEV_STATUS_TYPE: // Total number of employees, total number of
fingerprints, total number of passwords, total number of cards, total attendance records, total
new attendance record.
            default:
                break;
            }
        }else if (ret == 0)
        {
            // invalid data
        }else
        {
            // Buff is not enough,
        }
```

2.5.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.6 CChex_GetNetConfig

2.6.1 Description functions

【Function】Query Anviz device’s network Configuration

2.6.2 Request

【Mode】 int CChex_GetNetConfig(IntPtr CchexHandle, int DevIdx);

【Parameter】 CChex_Start successfully create the handle, Input[Parameter];

DevIdx,search the device Input[Parameter];

2.6.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】 CChex_Update, Type return CCHEX_RET_GETNETCFG_TYPE

Data Part：

```
uint MachineId;       // Device ID
int Result;           // 0:OK, -1:error
byte [4]IpAddr;// IP address
byte [4]IpMask;// Mask
byte [6]MacAddr;// MAC
byte [4]GwAddr;// Gateway
byte [4]ServAddr;// Sever IP address
byte RemoteEnable;//Standby
byte [2]Port;// Port
byte Mode;// Mode：0: Server, 1:Client
byte DhcpEnable;// DHCP, 0:Disable, 1:Enable
```

2.6.4 Sample

```
int ret  = CChex_GetNetConfig();
```

2.6.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.7 CChex_SetNetConfig

2.7.1 Description functions

【Function】Set the network configuration of the Anviz device.

2.7.2 Request

【Mode】 `int CChex_SetNetConfig(IntPtr CchexHandle, int DevIdx, ref CCHEX_NETCFG_INFO_STRU Config);`

【Parameter】CchexHandle: ,CChex_Start successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

Config:

```
byte [4]IpAddr;// IP Address

byte [4]IpMask;// Mask

byte [6]MacAddr;// MAC address

byte [4]GwAddr;// Gateway

byte [4]ServAddr;// Server IP

byte RemoteEnable;//Standby

byte [2]Port;// Port

byte Mode;// Mode: 0:Server, 1:Client

byte DhcpEnable;// DHCP, 0:Disable, 1:Enable
```

2.7.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update,Type return CCHEX_RET_SETNETCFG_TYPE

```
Data part：
   uint MachineId;       // Device ID
   int Result;           // 0:OK, -1:error
```

2.7.4 Sample

int ret = (anviz_handle, dev_idx, ref dev_info);

2.7.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.8 CChex_MsgGetByIdx

2.8.1 Description functions

【Function】Reads the designated short message's start date, end date, and message content .

2.8.2 Request

【Mode】`int CChex_MsgGetByIdx(IntPtr CchexHandle, int DevIdx, byte Idx);`

【Parameter】CchexHandle：,CChex_Start successfully create the handle，Input[Parameter];

DevIdx: Index of device，Input[Parameter];

Idx,Read the short message, Input[Parameter];

2.8.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGGETBYIDX_UNICODE_INFO_TYPE

```
Data part:

   uint MachineId;        // Device ID

   int Result;            // 0:OK, -1:error

   int Len;               // The length of the following data

   byte[5] EmployeeId;    // User ID, 0 means public short message

   byte StartYear;        // Start time Year

   byte StartMonth;       // Start time Month

   byte StartDay;         // Start time Day

   byte EndYear;          // End time Year

   byte EndMonth;         // End time Month

   byte EndDay;           // End time Day

   byte [48*2]content     // Content of short message,UNICODE
```

2.8.4 Sample

```
int ret = CChex_MsgGetByIdx(anviz_handle, dev_idx, msg_idx);
```

2.8.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.9 CChex_MsgDelByIdx

2.9.1 Description functions

【Function】Delete the designated short message

2.9.2 Request

【Mode】 `int CChex_MsgDelByIdx(IntPtr CchexHandle, int DevIdx, byte Idx);`

【Parameter】 CChex_Start， successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Idx,Delete the designed short message,Input[Parameter]

2.9.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】 CChex_Update， Type return CCHEX_RET_MSGDELBYIDX_INFO_TYPE

Data Part：

```
uint MachineId;        // Device ID
int Result;            // 0:OK， -1:error
```

2.9.4 Sample

```
int ret =CChex_MsgDelByIdx(anviz_handle, dev_idx, msg_idx);
```

2.9.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.10 CChex_MsgAddNew

2.10.1 Description functions

【Function】 Add short message.

2.10.2 Request

【Mode】int CChex_MsgAddNew(IntPtr CchexHandle, int DevIdx, byte[] Data, int Len);

【Parameter】CChex_Start，successfully create the handle，Input[Parameter];

DevIdx: Index of device，Input[Parameter];

Data,Short message, Input[Parameter];

```
    byte[5] EmployeeId;    // User ID，0 means public short message
    byte StartYear;        // Start time Year
    byte StartMonth;       // Start time Month
    byte StartDay;         // Start time Day
    byte EndYear;          // End time Year
    byte EndMonth;         // End time Month
    byte EndDay;           // End time Day
    byte [48*2]content     // Content of short message,UNICODE
```

Len,the length of short message，Input[Parameter];

2.10.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

Data Part：

```
    uint MachineId;        // Device ID
    int Result;            // 0:OK, -1:error
```

2.10.4 Sample

int ret = CChex_MsgAddNew(anviz_handle, dev_idx, send_buff, send_len);

2.10.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.11 CChex_MsgGetAllHead

2.11.1 Description functions

【Function】Read all short message headers.

2.11.2 Request

【Mode】int CChex_MsgGetAllHead(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

2.11.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

Data Part:

```
uint MachineId;        // Device ID

int Result;            // 0:OK, -1:error

int Len;               // All 50 SMS users, start and end time.

byte[5] EmployeeId;    // User ID, 0 means public SMS
```

```
byte StartYear;       // Start time Year
byte StartMonth;      // Start time Month
byte StartDay;        // Start time Day
byte EndYear;         // End time Year
byte EndMonth;        // End time Month
byte EndDay;          // End time Day
  ...Total 50 records
```

2.11.4 Sample

int ret =CChex_MsgGetAllHead(anviz_handle, dev_idx);

2.11.5 Notice

1.Make sure that CChex_Start has been started successfully before run.已经成功启动;

2.12 CChex_RebootDevice

2.12.1 Description functions

【Function】Restart the Anviz device

2.12.2 Request

【Mode】int CChex_RebootDevice(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle:,successfully create the handle, Input[Parameter];

DevIdx: Index of device，Input[Parameter];

2.12.3 Response

【Return value】1： The command was executed successfully.； Minus： Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

Data Part：

```
uint MachineId;        // Device ID

int Result;            // 0:OK, -1:error
```

2.12.4 Sample

int ret = CChex_RebootDevice(anviz_handle, dev_idx);

2.12.5 Notice

1.Make sure that CChex_Start has been started successfully before run.已经成功启动;

2.13 CChex_SetTime

2.13.1 Description functions

【Function】Set date/time of the Anviz device

2.13.2 Request

【Mode】int CChex_SetTime(IntPtr CchexHandle, int DevIdx, int Year, int Month, int Day, int Hour, int Min, int Sec);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter];

DevIdx: Index of device，Input[Parameter];

Year、Month、Day、Hour、 Min、 Sec，Input[Parameter]

2.13.3 Response

【Return value】1： The command was executed successfully.； Minus： Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
Data Part：
  uint MachineId;       // Device ID
  int Result;           // 0:OK, -1:error
```

2.13.4 Sample

```
int ret = CChex_SetTime(anviz_handle, dev_idx, 2018, 1, 25, 10, 10, 10);
```

2.13.5 Notice

1.Make sure that CChex_Start has been started successfully before run.已经成功启动;

2.14 CChex_GetSNConfig

2.14.1 Description functions

【Function】Get the SN of the Anviz device

2.14.2 Request

【Mode】`int CChex_GetSNConfig(IntPtr CchexHandle, int DevIdx);`

【Parameter】CchexHandle:，successfully create the handle，Input[Parameter]句柄，Input[Parameter]；

DevIdx: Index of device, Input[Parameter];

2.14.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
Data Part:

uint MachineId;       // Device ID

int Result;           // 0:OK, -1:error

byte[16] sn;          // SN, Each number, and it needs to convert into characters
```

2.14.4 Sample

int ret = CChex_GetSNConfig(anviz_handle, dev_idx);

2.14.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.15 CChex_DownloadAllRecords

2.15.1 Description functions

【Function】Download all attendance records.

2.15.2 Request

【Mode】`int CChex_DownloadAllRecords(IntPtr CchexHandle, int DevIdx);`

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

2.15.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
Data Part:

uint MachineId;       // Device ID

byte NewRecordFlag;   // New or not record?

byte[5] EmployeeId;   // User ID,

byte[4] Date;         // date/time, The number of seconds apart from 2000.

byte BackId;          // Backup ID, the 3 digits; Card, the 2 digits; Password,
the 1~0; 11, 7~4 digits; 10 Fingerprints (1~10) digits

byte RecordType       // Record type, 7 digits; 1: open the door , 0: can't open
the door ; 3~0 digits attendance status

byte[3] WorkType;     // Workcode

byte Rsv              // Reserved
```

2.15.4 Sample

int ret = CChex_DownloadAllRecords(anviz_handle, dev_idx);

2.15.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.16 CChex_DeleteRecordInfo

2.16.1 Description functions

【Function】Delete all records, or clear all/ some new records flag.

2.16.2 Request

【Mode】int CChex_DeleteRecordInfo(IntPtr CchexHandle, int DevIdx, ref

CCHEX_DEL_RECORD_INFO_STRU Config);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter];

DevIdx: Index of device，Input[Parameter];

Config,Input[Parameter];

byte del_type;   //0:Delete all records; 1：Clear all new record flag; 2：Clear some records flag，and the deleted number based on del_count

uint del_count;   // del_type =2 时，assigned clear new records number

2.16.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update，Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

Data Part：

```
uint MachineId;       // Device ID
int Result;           // 0:OK，-1:error
uint deleted_count;   // How many records or new records flag be deleted?
```

2.16.4 Sample

int ret = CChex_DeleteRecordInfo(anviz_handle， dev_idx， ref delete_record);

2.16.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.17 CChex_GetBasicConfigInfo

2.17.1 Description functions

【Function】Get the basic configuration of the Anviz device

2.17.2 Request

【Mode】int CChex_GetBasicConfigInfo(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

2.17.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
Data Part:

uint MachineId;       // Device ID

int Result;           // 0:OK, -1:error

CCHEX_GET_BASIC_CFG_INFO_STRU Cfg;

    byte[8] software_version;  // firmware version,

    uint password;             // Password, longest support 6 digits

    byte delay_for_sleep;      // Delay sleep time, 0~250 Mins, 0 don’t sleep

    byte volume;               // Volume, 0~5, 0: mute, 5: Max
    byte language;             //Language, 0: Simplified Chinese, 1: Traditional
Chinese, 2: English, 3.French, 4: Spanish, 5: Portuguese

    byte date_format;          // Date format, 0:Chinese, 1: UK, 2: USA
    byte time_format;          // Time format, 0: 24Houes, 1: 12Hours

    byte machine_status;       // Attendance Status, 0~15
```

```
                byte modify_language;      // Modify Language，0x10 You can modify the
language of the device.
                byte cmd_version;          // Instruction version
```

2.17.4 Sample

int ret = CChex_GetBasicConfigInfo(anviz_handle, dev_idx);

2.17.5 Notice

1. 1.Make sure that CChex_Start has been started successfully before run.;

2.18 CChex_SetBasicConfigInfo

2.18.1 Description functions

【Function】Set the basic configuration of the Anviz device

2.18.2 Request

【Mode】int CChex_SetBasicConfigInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX_SET_BASIC_CFG_INFO_STRU Config);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Config,Input[Parameter]；Unmodified fields，0xFF

```
                uint password;             // Password，String convert into number
                byte pwd_len;              // Password length
                byte delay_for_sleep;      // Delay sleep time，0~250 Mins，0 don’t sleep
                byte volume;               // Volume，0~5，0：mute，5：Max
                byte language;             //Language，0:Simplified Chinese，1:Traditional
```

```
Chinese, 2: English, 3.French, 4: Spanish, 5: Portuguese
                    byte date_format;          // Date format, 0:Chinese, 1: UK, 2: USA
                    byte time_format;          // Time format, 0: 24Houes, 1: 12Hours
                    byte machine_status;       // Attendance Status, 0~15
                    byte modify_language;      // Modify Language, 0x10 You can modify the
language of the device.
                    byte rsv;                  // Reserved
```

2.18.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
Data Part:
uint MachineId;       // Device ID
int Result;           // 0:OK, -1:error
```

2.18.4 Sample

```
int ret = CChex_SetBasicConfigInfo(anviz_handle, dev_idx, ref set_basic_cfg);
```

2.18.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.19 CChex_ListPersonInfo

2.19.1 Description functions

【Function】Get all user information.

2.19.2 Request

【Mode】`int CChex_ListPersonInfo(IntPtr CchexHandle, int DevIdx);`

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

2.19.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update，Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

Data Part：

```
uint MachineId;         // Device ID

int CurIdx;             // Current index number

int TotalCnt;           // The total number of users

byte [5]EmployeeId;     // User ID，5 bytes,Longest 12digits

byte password_len;      // Password length

byte max_password;      // Maximum length of password,= 6，unchangeable

int password;           // Password

byte max_card_id;       // The maximum length of the card number，= 6 (3bytes)
or 10 （4bytes）, unchangeable

uint card_id;           // Card number

byte max_EmployeeName;      // The maximum length of a user's name，= 10、20 、
64 and160,unchangeable；=160 ,The user name is stored in EmployeeName2.

byte[64] EmployeeName;      // User name
```

```
                byte DepartmentId;          // Department ID

                byte GroupId;               // Group ID

                byte Mode;                  // Attendance mode

                uint  Fp_Status;            // Fingerprint register status, 0~9:fp; 10:face; 
11:iris1; 12:iris2

                byte Rserved1;              // for 22

                byte Rserved2;              // for 72 and 22

                byte Special;               // Special information


                // DR info

                byte[160] EmployeeName2;    // Employee Name2

                byte[13] RFC;               // RFC Information

                byte[18] CURP;              // CURP Information
```

2.19.4 Sample

int ret = CChex_ListPersonInfo(anviz_handle, dev_idx);

2.19.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.20 CChex_ModifyPersonInfo

2.20.1 Description functions

【Function】Modify the relevant user's information.

2.20.2 Request

【Mode】int CChex_ModifyPersonInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX_RET_PERSON_INFO_STRU personlist, byte person_num);

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

person_num, How many users were changed, Input[Parameter];

Personlist, user details information, Input[Parameter];

```
uint MachineId;       // Device ID
int CurIdx;           // Current index number
int TotalCnt;         // The total number of users
byte [5]EmployeeId;   // User ID, 5 bytes,Longest 12digits
byte password_len;    // Password length
byte max_password;    // Maximum length of password,= 6, unchangeable
int password;         // Password
byte max_card_id;     // The maximum length of the card number, = 6 (3bytes) or 10 (4bytes), unchangeable
uint card_id;         // Card number
byte max_EmployeeName;    // The maximum length of a user's name, = 10、20 、64 and160,unchangeable; =160 ,The user name is stored in EmployeeName2.
byte[64] EmployeeName;    // User name
byte DepartmentId;        // Department ID
byte GroupId;             // Group
byte Mode;                // Attendance mode
uint  Fp_Status;          // Fingerprint register status, 0~9:fp; 10:face; 11:iris1; 12:iris2
byte Rserved1;            // for 22
byte Rserved2;            // for 72 and 22
byte Special;              // Special information


// DR info

byte[160] EmployeeName2;   // User name 2

byte[13] RFC;              // RFC Information

byte[18] CURP;             // CURP Information
```

2.20.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

Data Part：

```
uint MachineId;       // Device ID

int Result;           // 0:OK, -1:error
```

2.20.4 Sample

int ret = `CChex_ModifyPersonInfo(anviz_handle, dev_idx, ref item, 1);`

2.20.5 Notice

1.Make sure that CChex_Start has been started successfully before run；

2.21 CChex_DeletePersonInfo

2.21.1 Description functions

【Function】Deletes the assigned person's information.

2.21.2 Request

【Mode】int CChex_DeletePersonInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX_DEL_PERSON_INFO_STRU Config);

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

Config,Input[Parameter];

```
byte []EmployeeId;    // User ID
```

byte operation;   // Which information be deleted ? The 3 digits: Card; The 2 digits: Password; The 1~0 digits: 1~0 digits : 11; 7~4 digits: 10 Fingerprints (1~10) ; 0xFF: Delete all user' s information .

2.21.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

Data Part:

```
uint MachineId;    // Device ID
int Result;        // 0:OK, -1:error
```

2.21.4 Sample

int ret = CChex_DeletePersonInfo(anviz_handle, dev_idx, ref delete_item);

2.21.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.22 CChex_DeletePersonInfo_VER_4_NEWID

2.22.1 Description functions

【Function】Delete the user’s information

(Version:User ID for Chat DevTypeFlag & 0XFF ==DEV_TYPE_VER_4_NEWID)

2.22.2 Request

【Mode】int CChex_DeletePersonInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX_DEL_EMPLOYEE_INFO_STRU_EXT_INF_ID_VER_4_NEWID Config);

【Parameter】CchexHandle：successfully create by CChex_Start the handle，Input[Parameter]；；

DevIdx: Index of device，Input[Parameter]；

Config: Input[Parameter]；

```
    byte [28]EmployeeId;  // User ID for the String

    byte operation;       // Delete user Information the 3 bit is card, The 2 bit is
Password ,The 0~1 byte is 11 and the 7~14 bit is number of the fingerprints(1~10)；  0xFF，
delete all user information
```

2.22.3 Response

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_DEL_EMPLOYEE_INFO_TYPE

Data Part：

```
uint      MachineId;        // Device ID

int       Result;           // 0:OK, -1:error

byte [28] EmployeeId; // User ID for the String
```

2.22.4 Sample

int ret = CChex_DeletePersonInfo(anviz_handle, dev_idx, ref delete_item);

2.22.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.23 CChex_DownloadFingerPrint

2.23.1 Description functions

【Function】Download fingerprint template from Anviz device

2.23.2 Request

【Mode】int CChex_DownloadFingerPrint(IntPtr CchexHandle, int DevIdx, byte[] EmployeeId, byte FingerIdx);

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx,Search the device, Input[Parameter];

EmployeeId, User ID, Input[Parameter];

FingerIdx,Fingerprint Indexing,1~10:means fingerprint number 1~10,Input[Parameter];

2.23.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

Data Part:

```
uint MachineId;        // Device ID
int Result;            // 0:OK, -1:error
byte []EmployeeId;     // User ID
byte FpIdx;            //Fingerprint indexing,1~10:means fingerprint number 1~10
uint fp_data_len;      // The length of the fingerprint data.
byte []Data;           // fingerprint data
```

2.23.4 Sample

int ret = CChex_DownloadFingerPrint(anviz_handle, dev_idx, EmployeeID, FpIdx);

2.23.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.24 CChex_DownloadFingerPrint_VER_4_NEWID

2.24.1 Function

【Function】Download the fingerprint information from the device

(Version:device version DevTypeFlag & 0XFF ==DEV_TYPE_VER_4_NEWID)

2.24.2 Request

【Mode】`int CChex_DownloadFingerPrint_VER_4_NEWID(IntPtr CchexHandle, int DevIdx, byte[] EmployeeId, byte FingerIdx);`

【Parameter】CchexHandle: successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

EmployeeId: byte[28]User ID, Input[Parameter]; User ID for the String

FingerIdx:Fingerprint indexing,1~10:means fingerprint number 1~10,Input[Parameter];

2.24.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex_Update Type Return CCHEX_RET_DLFINGERPRT_TYPE

```
Data Part：

  uint MachineId;        // Device ID

  int Result;            // 0:OK, -1:error

  byte [28]EmployeeId;   // User ID //User ID for the String ID byte[28]

  byte FpIdx;            //Fingerprint indexing,1~10:means fingerprint number 1~10

  uint fp_data_len;      // The length of the fingerprint data.

  byte []Data;           // fingerprint data
```

2.24.4 Sample

int ret = CChex_DownloadFingerPrint_VER_4_NEWID(anviz_handle, dev_idx, EmployeeID, FpIdx);

2.24.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.25 CChex_UploadFingerPrint

2.25.1 Description functions

【Function】Upload fingerprint template to Anviz device

2.25.2 Request

【Mode】int CChex_UploadFingerPrint(IntPtr CchexHandle, int DevIdx, byte[] EmployeeId, byte FingerIdx, byte[] FingerData, int DataLen);

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

EmployeeId, User ID, Input[Parameter];

FingerId; Fingerprint indexing, 1~10: means fingerprint number 1~10, Input[Parameter];

FingerData, fingerprint data, Input[Parameter];

DataLen, The length of the fingerprint data, Input[Parameter];

2.25.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

Data Part:

```
uint MachineId;        // Device ID
int Result;            // 0:OK, -1:error
```

2.25.4 Sample

int ret = CChex_DownloadFingerPrint(anviz_handle, dev_idx, EmployeeID, FpIdx);

2.25.5 Notice

1. 1.Make sure that CChex_Start has been started successfully before run.

2.26 CChex_UploadFingerPrint_VER_4_NEWID

2.26.1 Function

【Function】Upload the fingerprint template to device (Device Type:"FACEPASS7" For :FaceTemplate download)

(Version:Device Version Model DevTypeFlag & 0XFF ==DEV_TYPE_VER_4_NEWID)

2.26.2 Request

【Mode】`int CChex_UploadFingerPrint_VER_4_NEWID(IntPtr CchexHandle, int DevIdx, byte[] EmployeeId, byte FingerIdx, byte[] FingerData, int DataLen);`

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；；

DevIdx: Index of device，Input[Parameter]；

EmployeeId: User ID，Input[Parameter]； //User ID for the String ID byte[28]

FingerIdx: Fingerprint indexing，1~10：means fingerprint number 1~10，Input[Parameter]；

FingerData，fingerprint data，Input[Parameter]；

DataLen，The length of the fingerprint data，Input[Parameter]；

2.26.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex_Update Type return CCHEX_RET_ULFINGERPRT_TYPE

Data Part：

```
uint MachineId;        // Device ID
int Result;            // 0:OK, -1:error
byte []EmployeeId;     // /User ID for the String ID byte[28]
```

```
                byte FpIdx;          // Fingerprint indexing, 1~10: means fingerprint number
1~10, Input[Parameter];
                uint fp_data_len;    // The length of the fingerprint data
                byte []Data;         // Fingerprint Data
```

2.26.4 Sample

int ret = `CChex_DownloadFingerPrint_VER_4_NEWID(anviz_handle, dev_idx, EmployeeID, FpIdx);`

2.26.5 Notice

1. Make sure that CChex_Start has been started successfully before run.

2.27 CChex_GetTime

2.27.1 Description functions

【Function】Get the date/time of the Anviz device.

2.27.2 Request

【Mode】`int CChex_GetTime(IntPtr CchexHandle, int DevIdx);`

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

2.27.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
Data Part:

 uint MachineId;        // Device ID

 int Result;            // 0:OK, -1:error

 CCHEX_MSG_GETTIME_STRU Cfg;

     uint Year;             //Year

     uint Month;            //Month

     uint Day;              //Day

     uint Hour;             //Hour

     uint Min;              //Minute

     uint Sec;              //Second
```

2.27.4 Sample

int ret = `CChex_GetTime(anviz_handle, dev_idx);`

2.27.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.28 CChex_InitUserArea

2.28.1 Description functions

【Function】Initial user area

Initialize all user data areas， Clear all user data, fingerprint data, password/card data.

2.28.2 Request

【Mode】`int CChex_InitUserArea(IntPtr CchexHandle, int DevIdx);`

【Parameter】 CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

2.28.3 Response

【Return value】1： The command was executed successfully.； Minus： Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
Data Part：
 uint MachineId;        // Device ID
 int Result;            // 0:OK, -1:error
```

2.28.4 Sample

```
int ret = CChex_InitUserArea(anviz_handle, dev_idx);
```

2.28.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.29 CChex_InitSystem

2.29.1 Description functions

【Function】initial device

initialize the Anviz device to restore factory Default.

2.29.2 Request

【Mode】`int CChex_InitSystem(IntPtr CchexHandle, int DevIdx);`

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter];

DevIdx: Index of device，Input[Parameter];

2.29.3 Response

【Return value】1： The command was executed successfully.; Minus： Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
Data Part:
 uint MachineId;       // Device ID
 int Result;           // 0:OK, -1:error
```

2.29.4 Sample

int ret = `CChex_InitSystem(anviz_handle, dev_idx);`

2.29.5 Notice

1. 1.Make sure that CChex_Start has been started successfully before run.

2.30 CChex_GetBasicConfigInfo2

2.30.1 Description functions

【Function】Get the Anviz device's configuration2

Get device's identification precision, fixed wiegand head, wiegand mode, workcode, real-time attendance records , schedule bell , lock control delay, record overflow warning, repeated attendance record time , door sensor delay, schedule bell delay.

2.30.2 Request

【Mode】int CChex_GetBasicConfigInfo2(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter];

DevIdx: Index of device，Input[Parameter];

2.30.3 Response

【Return value】1： The command was executed successfully.； Minus： Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
        Data Part：

         uint MachineId;        // Device ID

         int Result;            // 0:OK, -1:error

         CCHEX_GET_BASIC_CFG_INFO2_STRU_EXT_INF Param;

                    byte compare_level;         //identification precision

                    byte wiegand_range;         //fixed wiegand head

                    byte wiegand_type;          //wiegand mode

                    byte work_code;             //Workcode function, enable or disable?

                    byte real_time_send;        //Realtime function ,enable or disable?

                    byte auto_update;           //Auto-update function, enable or
disable?

                    byte bell_lock;             //Schedule bell function, enable or
disable?

                    byte lock_delay;            //Lock time delay

                    uint record_over_alarm;     //Record overflow alarm
```

```
byte re_attendance_delay;   //Repeat attendance time delay

byte door_sensor_alarm;     //Door sensor time delay

byte bell_delay;            //Schedule bell time delay

byte correct_time;          //Time calibration
```

2.30.4 Sample

int ret = CChex_GetBasicConfigInfo2(anviz_handle, dev_idx);

2.30.5 Notice

1. 1.Make sure that CChex_Start has been started successfully before run.

2.31 CChex_SetBasicConfigInfo2

2.31.1 Description functions

【Function】Set up the Anviz device's configuration information2

2.31.2 Request

【Mode】int CChex_SetBasicConfigInfo2(IntPtr CchexHandle, int DevIdx, ref CCHEX_GET_BASIC_CFG_INFO2_STRU_EXT_INF config);

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

config: Input[Parameter];

```
byte compare_level;         //identification precision

byte wiegand_range;         //fixed wiegand head

byte wiegand_type;          //wiegand mode
byte work_code;              //Workcode function, enable or disable?
byte real_time_send;         //Realtime function ,enable or disable?
byte auto_update;        //Auto-update function, enable or disable?
byte bell_lock;              //Schedule bell function, enable or disable?
byte lock_delay;             //Lock time delay
uint record_over_alarm;      //Record overflow alarm
byte re_attendance_delay;    //Repeat attendance time delay
byte door_sensor_alarm;      //Door sensor time delay
byte bell_delay;             //Schedule bell time delay
byte correct_time;           //Time calibration
```

2.31.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update，Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
Data Part：
 uint MachineId;       // Device ID
 int Result;           // 0:OK, -1:error
```

2.31.4 Sample

int ret = `CChex_SetBasicConfigInfo2(anviz_handle, dev_idx,ref config);`

2.31.5 Notice

1. 1.Make sure that CChex_Start has been started successfully before run.

2.32 CChex_GetPeriodTime

2.32.1 Description functions

【Function】Get time zone from device

Read the timezone information，Total 32 timezones。

2.32.2 Request

【Mode】`int CChex_GetPeriodTime(IntPtr CchexHandle, int DevIdx, byte SerialNumbe);`

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

SerialNumbe,timezone number，Input[Parameter]；

2.32.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update，Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
Data Part：
 uint MachineId;       // Device ID
 int Result;           // 0:OK，-1:error
 CCHEX_GET_PERIOD_TIME_ONE_STRU_EXT_INF day[7]; //One week 7 days,28Byte
            byte StartHour;    //Start hour
            byte StartMin;     //Start mins
            byte EndHour;      //End hour

            byte EndMin;       //End mins
```

2.32.4 Sample

int ret = CChex_GetPeriodTime(anviz_handle, dev_idx,SerialNumbe);

2.32.5 Notice

1. 1.Make sure that CChex_Start has been started successfully before run.

2.33 CChex_SetPeriodTime

2.33.1 Description functions

【Function】Set timezone

Total 32 timezones。

2.33.2 Request

【Mode】int CChex_SetPeriodTime(IntPtr CchexHandle, int DevIdx, ref CCHEX_SET_PERIOD_TIME_STRU_EXT_INF config);

【Parameter】CchexHandle：,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

Config: Input[Parameter];

byte SerialNumbe;

CCHEX_GET_PERIOD_TIME_ONE_STRU_EXT_INF day[7]; //one week with 7 days,so should be have 7 groups,28Byte

byte StartHour;   //Start Hour

```
byte StartMin;     //Start Mins

byte EndHour;      //End Hours

byte EndMin;       //End Mins
```

2.33.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update，Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
Data Part：

 uint MachineId;       // Device ID

 int Result;           // 0:OK, -1:error
```

2.33.4 Sample

int ret = `CChex_SetPeriodTime(anviz_handle, dev_idx,ref config);`

2.33.5 Notice

1. 1.Make sure that CChex_Start has been started successfully before run；

2.34 CChex_GetTeamInfo

2.34.1 Description functions

【Function】Get Group information

Read one group information, group number 0-16, group0 = normal close; group1=normal open. Only read group2 - group16.

2.34.2 Request

【Mode】`int CChex_GetTeamInfo(IntPtr CchexHandle, int DevIdx, byte TeamNumbe);`

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

TeamNumbe:Group number,Input[Parameter]；

2.34.3 Response

【Return value】1： The command was executed successfully.； Minus： Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

Data Part：

```
uint MachineId;            // Device ID

int Result;                // 0:OK, -1:error

byte[4]   PeriodTimeNumber;//4Group timezone number(refer to 2.29)
```

2.34.4 Sample

int ret = `CChex_GetTeamInfo(anviz_handle, dev_idx,TeamNumbe);`

2.34.5 Notice

2. 1.Make sure that CChex_Start has been started successfully before run.

2.35 CChex_SetTeamInfo

2.35.1 Description functions

【Function】Setup group

Setup one group, group number 0- 16, group 0= normal close; group1= normal open, only setup group2 -group16.

2.35.2 Request

【Mode】`int CChex_SetTeamInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX_SET_TEAM_INFO_STRU_EXT_INF config);`

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

```
DevIdx: Index of device, Input[Parameter];

Config:         Input[Parameter];

        byte      TeamNumbe;           //Group Number

        byte[4]    PeriodTimeNumber;    //4Group ,timezone
```

2.35.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
Data Part:

uint MachineId;      // Device ID

int Result;          // 0:OK, -1:error
```

2.35.4 Sample

int ret = `CChex_SetTeamInfo(anviz_handle, dev_idx,ref config);`

2.35.5 Notice

1. 1.Make sure that CChex_Start has been started successfully before run.

2.36 CCHex_AddFingerprintOnline

2.36.1 Description functions

【Function】Register fingerprint online

2.36.2 Request

【Mode】`int CCHex_AddFingerprintOnline(IntPtr CchexHandle, int DevIdx, ref CCHEX_ADD_FINGERPRINT_ONLINE_STRU_EXT_INF Param);`

【Parameter】CchexHandle：,successfully create the handle, Input[Parameter];

```
DevIdx: Index of device, Input[Parameter];

Param:          Input[Parameter];

        byte[5]   EmployeeId;      //User ID

        byte      BackupNum;       //Backup number
```

2.36.3 Response

【Return value】1： The command was executed successfully.； Minus： Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
Data Part:

 uint MachineId;       // Device ID
```

```
        int Result;          // 0:OK, -1:error
```

2.36.4 Sample

int ret = `CCHex_AddFingerprintOnline(anviz_handle, dev_idx,ref Param);`

2.36.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2. After the recall, the device will register the fingerprint twice, timeout is 10 seconds, the registration error (timeout, operation failure, the user already exists, the fingerprint exists) returned -1, and successfully returns 0.

2.37 CCHex_ForcedUnlock

2.37.1 Description functions

【Function】Open the door by software (Force open the lock)

2.37.2 Request

【Mode】`int CCHex_ForcedUnlock(IntPtr CchexHandle, int DevIdx, ref CCHEX_FORCED_UNLOCK_STRU_EXT_INF Param);`

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param:　　　Input[Parameter]；(Param =IntPtr.Zerohe ; default null pointer , Malaysia panasonic project customization needs to enter Param)；

```
    byte      LockCmd;          //Lock command
    byte[5]   EmployeeId;       //User ID
```

2.37.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update，Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
Data Part：
 uint MachineId;       // Device ID
 int Result;           // 0:OK, -1:error
```

2.37.4 Sample

int ret = CCHex_ForcedUnlock(anviz_handle, DevIdx,IntPtr.Zero);

2.37.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;
2. If Not Malaysia Panasonic project customization, Param none pointer..

2.38 CCHex_Udp_Search_Dev

2.38.1 Description functions

【Function】UDP Research the Anviz device

2.38.2 Request

【Mode】int CCHex_Udp_Search_Dev(IntPtr CchexHandle);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter];

2.38.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

Data Part：(Data length =sizeof(int)+DevNum*sizeof(CCHEX_UDP_SEARCH_STRU_EXT_INF))

```
    int DevNum;                              // quantity of device, ( =0, No
device,data length =4)

    CCHEX_UDP_SEARCH_STRU_EXT_INF[] dev_net_info ;        //Device Data(Single
180Byte)

              uint         MachineId;       //Device ID

              int          Result;          //0: ok -1: fail

              int          DevHardwareType; //device type(0:Normal device;1:support
    DNS;2:support wifi)

              byte[167]    Data;            //Data(Convert based on device type)

              Byte         Padding;         //Data alignment, padding

    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

```
byte[167] Data: Data type definition

Device type = 0 :Normal device

         CCHEX_UDP_SEARCH_STRU BasicSearchInfo;           //normal device 63Byte

                  byte[10]   DevType;            //device type

                  byte[16]   DevSerialNum;       //device serial numebr

                  byte[4]   IpAddr;              //IP Address

                  byte[4]   IpMask;              //MASK

                  byte[4]   GwAddr;              //Gateway

                  byte[6]   MacAddr;             //MAC

                  byte[4]   ServAddr;            //Server IP

                  byte[2]   Port;                //Port
```

```
                byte     NetMode;              //network mode
                byte[8]   Version;             //Firmware version
                byte[4]   Reserved;            //Reserved
Device type = 1 :Support DNS firmware
        CCHEX_UDP_SEARCH_WITH_DNS_STRU DnsSearchInfo;            //Support DNS firmware 167Byte
                CCHEX_UDP_SEARCH_STRU BasicSearchInfo;      //device type 0:normal device
            63Byte structure
                byte[4]   Dns;                          //DNS
                byte[100]   Url;                        //URL

Device type = 2 : Support wifi
        CCHEX_UDP_SEARCH_TWO_CARD_STRU TwocardSearchInfo;        \\ support wifi    102Byte
                byte[10]   DevType;            //Device type
                byte[16]   DevSerialNum;       //Device serial number
                byte[4]   ServAddr;            //Server IP
                byte[2]   Port;                //Port
                byte     NetMode;              //Network mode
                byte[8]   Version;             //Firmware version
                byte[4]   Reserved;            //Reserved
                byte     CardNumber;           //Network card number
                CCHEX_UDP_SEARCH_CARD_STRU CardInfo[2];     //2group network card information:
            single network card information = 28Byte
                byte[10]   CardName;      //Network card name
                        byte[4]   IpAddr;          //IP Address
                        byte[4]   IpMask;          //MASK
                        byte[4]   GwAddr;          //Gateway
```

```
byte[6]   MacAddr;         //MAC Address
```

~~~~~~

2.38.4 Sample

int ret = `CCHex_Udp_Search_Dev(anviz_handle);`

2.38.5 Notice

1. Make sure that CChex_Start has been started successfully before run.

2.39 CCHex_Udp_Set_Dev_Config

2.39.1 Description functions

【Function】UDP setting device configuration

2.39.2 Request

【Mode】`int CCHex_Udp_Set_Dev_Config(IntPtr CchexHandle, ref CCHEX_UDP_SET_DEV_CONFIG_STRU_EXT_INF config);`

【Parameter】CchexHandle:，successfully create the handle，Input[Parameter];

```
Config:      Input[Parameter];
    byte[4]   IpAddr;             //IP Address
    byte[4]   IpMask;             //MASK
    byte[4]   GwAddr;             //Gateway
    byte[6]   MacAddr;            //MAC address
    byte[4]   ServAddr;           //Server IP
    byte[2]   Port;               //Port
    byte      NetMode;            //Network mode
```
~~~~~~

```
        byte[3]   Padding;            //structure alignment,padding 3byte
        uint      NewMachineId;       //New Device ID
        byte[4]   Reserved;           //Reserved
        byte[12]   DevUserName;       //User name
        byte[12]   DevPassWord;       //Password
        int        DevHardwareType;   //Device type 0:Normal device ;1:support DNS
    device
        byte[4]   Dns;                //DNS    device type = 0 Dns , unchangeable
        byte[100]   Url;              //URL    device type = 0 Url,unchangeable
```

2.39.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

Data Part：

```
uint MachineId;        // Device ID
int Result;            // 0:OK, -1:error
```

2.39.4 Sample

int ret = `CCHex_Udp_Set_Dev_Config(anviz_handle,ref config);`

2.39.5 Notice

1. 1.Make sure that CChex_Start has been started successfully before run.

2.40 CCHex_ClientConnect

2.40.1 Description functions

【Function】The client actively connects to the server.

2.40.2 Request

【Mode】`int CCHex_ClientConnect(IntPtr CchexHandle, byte[] Ip, int Port);`

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

Ip: Server IP address，Input[Parameter]；

Port: Server Port, Input[Parameter]；

2.40.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】Failure:CChex_Update,Type return CCHEX_RET_CLINECT_CONNECT_TYPE;

```
  Data Part：
      int Result;            //-1 err
      byte[24] Addr;         //IP:Port  如(192.168.100.100:5010)
Successful:CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE
  Data Part：
      int DevIdx;            //
      uint MachineId;        //Device ID
      byte[24] Addr;         //IP:Port   For example(192.168.100.100:5010)
      byte[8] Version;       //Version Number
      byte[8] DevType;       //Device type
```

```
int DevTypeFlag;      //Type flag
```

2.40.4 Sample

int ret = `CCHex_ClientConnect(anviz_handle,Ip,Port);`

2.40.5 Notice

1. 1.Make sure that CChex_Start has been started successfully before run.

2.41 CCHex_ClientDisconnect

2.41.1 Description functions

【Function】The client actively disconnects the server.

2.41.2 Request

【Mode】`int CCHex_ClientDisconnect(IntPtr CchexHandle, int DevIdx);`

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

2.41.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】Failure:CChex_Update，Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

Data Part：

```
int Result;           // -1:error

int DevIdx;           //
```

Successful:CChex_Update, Type return CCHEX_RET_MSGADDNEW_UNICODE_INFO_TYPE

```
Data Part:

int DevIdx;                //Index of device, Input[Parameter];

uint MachineId;            //Device ID

uint Live;                 //

byte[24] Addr;             //ip:Port   如(192.168.100.100:5010)

byte[8] Version;           //Version number

byte[8] DevType;           //Device type
```

2.41.4 Sample

int ret = CCHex_ClientDisconnect(anviz_handle,DevIdx);

2.41.5 Notice

1.Make sure that CChex_Start has been started successfully before run.

2.42 CChex_GetInfomationCode

2.42.1 Description function

【Function】Gets the factory information

Read the factory information ANSI version data:10Byte,UNICODE version data:20Byte

2.42.2 Request

【Mode】 `int CChex_GetInfomationCode(IntPtr CchexHandle, int DevIdx);`

【Parameter】 CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

2.40.3 Response

【Return value】 1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】 CChex_Update, Type return CCHEX_RET_GET_INFOMATION_CODE_TYPE

```
Data Part:
   uint MachineId;       // Device ID
   int Result;           // 0:OK, -1:error
   int fp_len;       //ANSI VERSION:fp_len == 10   UNICODE VERSION:fp_len == 20
   byte[20] data;    //
```

2.42.4 Sample

int ret = `CChex_GetInfomationCode(anviz_handle, dev_idx);`

2.42.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.43 CChex_SetInfomationCode

2.43.1 Description function

【Function】Modify the factory information

Modify the factory information ANSI version data:10Byte ,UNICODE version data:20Byte

2.43.2 Request

【Mode】`int CChex_SetInfomationCode(IntPtr CchexHandle, int DevIdx ,byte[] Data, int DataLen);`

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Data: Byte[10] or Byte[20] parameter，Input[Parameter]；

DataLen: Write the parameters based on the version, , ANSI version = 10,UNICODE version = 20, Input[Parameter]；

2.43.3Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_SET_INFOMATION_CODE_TYPE

Data Part：

```
uint MachineId;       // Device ID

int Result;           // 0:OK, -1:error
```

2.43.4 Sample

int ret = CChex_SetInfomationCode(anviz_handle, dev_idx,Data,DataLen);

2.43.5 Notice

1. Make sure that CChex_Start has been started successfully before run.；

2.44 CChex_GetBellInfo

2.44.1 Description function

【Function】Get the Schedule bell information

Read all time of schedule bell，totally 30.

2.44.2 Request

【Mode】`int CChex_GetBellInfo(IntPtr CchexHandle, int DevIdx);`

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

2.44.3 Response

【Return value】1： The command was executed successfully.； Minus： Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_GET_BELL_INFO_TYPE

```
Data Part：
  uint MachineId;       // Device ID
  int Result;           // 0:OK, -1:error
  CCHEX_RET_GET_BELL_TIME_POINT time_point[30];
          byte  hour;
          byte  minute;
         byte  flag_week; //Week mark/flag(Binary 0000000 means：Sat Fri Thu Wed
```

Tue Mon Sun)

Byte[2] padding; //data structure alignment , invalid data

2.44.4 Sample

int ret = CChex_GetBellInfo(anviz_handle, dev_idx);

2.44.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.45 CChex_SetBellInfo

2.45.1 Description function

【Function】Setup the schedule bell information

Setup the time of schedule bell

2.45.2 Request

【Mode】int CChex_SetBellInfo(IntPtr CchexHandle, int DevIdx ,byte BellTimeNum,byte Hour,byte Min,byte FlagWeek);

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

BellTimeNum: The time to ring the bell, Input[Parameter];

Hour: Hour, Input[Parameter];

Min: Minute, Input[Parameter];

FlagWeek: Week mark/flag(Binary 0000000means: Sat Fri Thu Wed Tue Mon Sun), Input[Parameter];

2.45.3 Response

【Return value】1： The command was executed successfully.； Minus： Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_SET_BELL_INFO_TYPE

```
Data Part：

    uint MachineId;        // Device ID

    int Result;            // 0:OK, -1:error
```

2.45.4 Sample

int ret = `CChex_SetBellInfo(CchexHandle,DevIdx ,BellTimeNum,Hour,Min,FlagWeek);`

2.45.5 Notice

1. Make sure that CChex_Start has been started successfully before run.；

2.46 CChex_GetUserAttendanceStatusInfo

2.46.1 Description function

【Function】Gets the self-defining attendance status table

Read the information of self-defining attendance status.

2.46.2 Request

【Mode】`int CChex_GetUserAttendanceStatusInfo(IntPtr CchexHandle,int DevIdx);`

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

2.46.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_GET_USER_ATTENDANCE_STATUS_TYPE

```
Data Part:

    uint MachineId;       // Device ID

    int Result;           // 0:OK, -1:error

    int fp_len;    //ANSI VERSION: fp_len == 80  UNICODE VERSION   fp_len == 160

    Byte atten_status_number;   //Number of attendance status == 8 default

    Byte[160] data_info;              //data format:  ANSI VERSION: unsigned char
[8][10]  UNICODE VERSION: unsigned char[8][20] 。

    Byte[3] padding;          //data structure alignment, invalid data
```

2.46.4 Sample

```
int ret = CChex_GetUserAttendanceStatusInfo(anviz_handle, dev_idx);
```

2.46.5 Notice

1. Make sure that CChex_Start has been started successfully before run.；

2.47 CChex_SetUserAttendanceStatusInfo

2.47.1 Description function

【Function】Setup the self-defining attendance status

Setup the self-defining attendance status.

2.47.2 Request

【 Mode 】 int CChex_SetUserAttendanceStatusInfo(IntPtr CchexHandle, int DevIdx,ref CCHEX_SET_USER_ATTENDANCE_STATUS_STRU Param);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param：8 groups self-defining status,Input[Parameter]；

```
        uint fp_len;//ANSI VERSION   fp_len = 80   UNICODE VERSION   fp_len = 160
        Byte atten_status_number;           //Number of attendance status 8
  default
        Byte[160] data_info;//ANSI VERSION:8][10]，UNICODE VERSION:[8][20]
        Byte[3] padding;                    //alignment, invalid data
```

2.47.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_SET_USER_ATTENDANCE_STATUS_TYPE

Data Part：

```
uint MachineId;       // Device ID
int Result;           // 0:OK, -1:error
```

2.47.4 Sample

int ret = CChex_SetUserAttendanceStatusInfo(CchexHandle,DevIdx ,ref Param);

2.47.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.48 CChex_ClearAdministratFlag

2.48.1 Description function

【Function】Clear the administrator flag

Clear all administrator flag.

2.48.2 Request

【Mode】`int CChex_ClearAdministratFlag(IntPtr CchexHandle,int DevIdx);`

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

2.48.3 Response

【Return value】1： The command was executed successfully.; Minus： Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_CLEAR_ADMINISTRAT_FLAG_TYPE

Data Part：

```
uint MachineId;       // Device ID
int Result;           // 0:OK, -1:error
```

2.48.4 Sample

```
int ret = CChex_ClearAdministratFlag(anviz_handle, dev_idx);
```

2.48.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.49 CChex_GetSpecialStatus

2.49.1 Description function

【Function】Get the special status

Get the current special status. Only for VF30/VP30/T60+

2.49.2 Request

【Mode】`int CChex_GetSpecialStatus(IntPtr CchexHandle, int DevIdx);`

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

2.49.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_GET_SPECIAL_STATUS_TYPE

Data Part:

```
uint MachineId;       // Device ID

int Result;           // 0:OK, -1:error

Byte status;         //Byte1: Door alarm status 0-normal ,status 1-alarm status; Byte5: Door status 0-close , 1-open; Byte6: Door sensor status 0-close, 1-open; Byte7: Lock status 0-close, 1-open

Byte[7] reserved;         //reserved  useless in the temporary
```

2.49.4 Sample

int ret = CChex_GetSpecialStatus(anviz_handle, dev_idx);

2.49.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.50 CChex_GetAdminCardnumberPassword

2.50.1 Description function

【Function】Read administrator card number or password. Only for T5/M5/T50m

Get administrator card number or password.

2.50.2 Request

【Mode】`int CChex_GetAdminCardnumberPassword(IntPtr CchexHandle, int DevIdx);`

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

2.50.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_GET_ADMIN_CARD_PWD_TYPE

Data Part:

```
uint MachineId;       // Device ID
int Result;           // 0:OK, -1:error
```

```
Byte[13] data;        //device type : T5A,T50

 * Parameter :data[13]  : if the device type is T5A, so:

 DATA    Add card ;delete card ;stress/duress card; special information  Byte
 1-4     5-8        9-12         13

The special information define as below:

  Byte 0: Add card

  Byte 1: Delete card

  Byte 2: Stress/duress

  If the device type is T50, so:

 DATA        Password length+ passwod       Reserved

 Byte           1-3                      4-13

 Password length = Byte(1) >> 4

Byte[3] padding;      //data structure alignment, invalid data
```

2.50.4 Sample

int ret = CChex_GetAdminCardnumberPassword(anviz_handle, dev_idx);

2.50.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.51 CChex_SetAdminCardnumberPassword

2.51.1 Description function

【Function】Setup administrator card or password. Only for T5/M5/T50M

Setup T5A/M5A admin card and T50M admin password.

2.51.2 Request

【Mode】int CChex_SetAdminCardnumberPassword(IntPtr CchexHandle, int DevIdx,byte[] Data, int DataLen);

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

Data[13]: Parameter refer to 2.47 :data[13] description , Input[Parameter];

DataLen: length = 13, Input[Parameter];

2.51.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_SET_ADMIN_CARD_PWD_TYPE

Data Part:

```
uint MachineId;        // Device ID
int Result;            // 0:OK, -1:error
```

2.51.4 Sample

int ret = CChex_SetAdminCardnumberPassword(CchexHandle,DevIdx,Data,DataLen);

2.51.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.52 CChex_GetDSTParam

2.52.1 Description function

【Function】Read the daylight saving time parameters

Read the flag of daylight saving time and time zone .

2.52.2Request

【Mode】`int CChex_GetDSTParam(IntPtr CchexHandle,int DevIdx);`

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

2.52.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update，Type return CCHEX_RET_GET_DST_PARAM_TYPE

```
Data Part：
  uint MachineId;       // Device ID
  int Result;           // 0:OK， -1:error
  CCCHEX_SET_DST_PARAM_STRU param;  //16byte
      byte enabled;        //0-Disable      1-enable;
      byte ate_week_type;   //1-Date format   2-week format;
      GET_DST_PARAM_TIME start_time;  //7 byte
          byte  month;
          byte  day;
          byte  week_num; //Weekly definition：0x01-0x04： First 1-4 week
      0x81-0x82：Last1-2 week
          byte flag_week;       //Week flag: flag_week 0-6:(Binary 0000000 means：
      Sat Fri Thu Wed Tue Mon Sun)
```

```
                byte  hour;
                byte  minute;
                byte  sec;
            GET_DST_PARAM_TIME special_time;  //7 byte
```

2.52.4 Sample

int ret = `CChex_GetDSTParam(anviz_handle, dev_idx);`

2.52.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.53 CChex_SetDSTParam

2.53.1 Description function

【Function】Setup daylight saving time

Setup daylight saving time flag and time zone

2.53.2 Request

【Mode】`int CChex_SetDSTParam(IntPtr CchexHandle, int DevIdx, ref CCHEX_SET_DST_PARAM_STRU Param);`

【Parameter】`CchexHandle: ,successfully create the handle, Input[Parameter];`

```
        DevIdx: Index of device, Input[Parameter];
        Param: 16byte, Input[Parameter];
                byte enabled;        //0-disable      1-enable;
                byte ate_week_type;   //1-Date format   2-week format;
```

```
GET_DST_PARAM_TIME start_time;  //7 byte

    byte  month;

    byte  day;

    byte  week_num; //Weekly definition: 0x01-0x04:  First 1-4 week
0x81-0x82: Last1-2 week

    byte flag_week;      //Week flag: flag_week 0-6:(Binary 0000000 means:
Sat Fri Thu Wed Tue Mon Sun)

    byte  hour;

    byte  minute;

    byte  sec;

GET_DST_PARAM_TIME special_time;  //7 byte
```

2.53.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_SET_DST_PARAM_TYPE

```
Data Part:

  uint MachineId;       // Device ID

  int Result;           // 0:OK, -1:error
```

2.53.4 Sample

int ret = `CChex_SetDSTParam(CchexHandle,DevIdx,ref Param);`

2.53.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.54 CChex_GetDevExtInfo

2.54.1 Description function

【Function】Gets the device extension information code

Read the factory name/Tax number/factory Address

2.54.2 Request

【Mode】`int CChex_GetDevExtInfo(IntPtr CchexHandle,int DevIdx);`

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

2.54.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_GET_DEV_EXT_INFO_TYPE

```
Data Part：
   uint MachineId;        // Device ID
   int Result;            // 0:OK, -1:error
   CCHEX_SET_DEV_EXT_INFO_STRU param;  //320byte
       Byte[50]  manufacturer_name;        //Factory name
       Byte[100] manufacturer_addr;        //Address
       Byte[15]  duty_paragraph;           //Tax number
       Byte[155] reserved;                 //Reserved
```

2.54.4 Sample

int ret = `CChex_GetDevExtInfo(CchexHandle,DevIdx);`

2.54.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.55 CChex_SetDevExtInfo

2.55.1 Description function

【Function】Modify the device extension information code

Modify the factory name/Tax number/factory Address

2.55.2 Request

【Mode】int CChex_SetDevExtInfo(IntPtr CchexHandle, int DevIdx,ref CCHEX_SET_DEV_EXT_INFO_STRU Param);

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

Param: 320byte, Input[Parameter];

```
Byte[50]  manufacturer_name;      //factory name
Byte[100] manufacturer_addr;      //factory address
Byte[15]  duty_paragraph;         //Tax number
Byte[155] reserved;               //Reserved
```

2.55.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_SET_DEV_EXT_INFO_TYPE

```
Data Part:

    uint MachineId;       // Device ID

    int Result;           // 0:OK, -1:error
```

2.55.4 Sample

int ret = `CChex_SetDevExtInfo(CchexHandle,DevIdx,ref Param);`

2.55.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.56 CChex_GetBasicConfigInfo3

2.56.1Description function

【Function】Get the device configuration information3

Read Wiegand mode。

2.56.2 Request

【Mode】`int CChex_GetBasicConfigInfo3(IntPtr CchexHandle,int DevIdx);`

【Parameter】CchexHandle:,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

2.56.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_GET_BASIC_CFG3_TYPE

```
Data Part:

  uint MachineId;       // Device ID

  int Result;           // 0:OK, -1:error

  CCHEX_SET_BASIC_CFG_INFO3_STRU param;  //15 byte

      Byte wiegand_type;         //wiegand mode

      Byte online_mode;          //online mode

      Byte collect_level;        //Acquisition threshold

      Byte pwd_status;           //communication password status  =0 ,connect
don' t need the communication password by TCP/IP; communication password status  =1,
send 0x04 command, verify the communication password by TCP/IP

      Byte sensor_status;           //=0  don' t report the door sensor status;
=1 activate report the door sensor status (the device will activate send the response
of 0x2F command )

      Byte[8]  reserved;         //Reserved

      Byte  independent_time;    //User independent time limit

      Byte  m5_t5_status;         //= 0  disable;  = 1 enable; Out =2, disable
In  =4  disable, out =5  disable, In

  Byte padding;     //alignment  invalid data
```

2.56.4 Sample

int ret = CChex_GetBasicConfigInfo3(CchexHandle,DevIdx);

2.56.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.57 CChex_SetBasicConfigInfo3

2.57.1 Description function

【Function】Setup the device configuration information3

Read Wiegand mode

2.57.2 Request

【 Mode 】 int CChex_SetBasicConfigInfo3(IntPtr CchexHandle, int DevIdx, ref CCHEX_SET_BASIC_CFG_INFO3_STRU Config);

【Parameter】CchexHandle：, successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

Config: 15byte, Input[Parameter];

```
Byte wiegand_type;          //wiegand mode

Byte online_mode;          //Online mode

Byte collect_level;        //Acquisition threshold

Byte pwd_status;            //communication password status  =0 ,connect
```

don’t need the communication password by TCP/IP; communication password status =1, send 0x04 command, verify the communication password by TCP/IP

```
Byte sensor_status;            //=0  don’t report the door sensor status;
```

=1 activate report the door sensor status (the device will activate send the response of 0x2F command)

```
Byte[8]  reserved;         //Reserved

Byte  independent_time;    //User independent time limit

Byte  m5_t5_status;         //= 0  disable;  = 1 enable; Out =2, disable
```

In =4 disable, out =5 disable, In

2.57.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_SET_BASIC_CFG3_TYPE

```
Data Part:
    uint MachineId;       // Device ID
    int Result;           // 0:OK, -1:error
```

2.57.4 Sample

int ret = CChex_SetBasicConfigInfo3(CchexHandle,DevIdx,ref Config);

2.57.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.58 CChex_ConnectionAuthentication

2.58.1 Description function

【Function】connection identification

connection identification, identification successful, then response the other command; if the identification successful and no data transfer in 5 mins , recover to dis-identification mode

2.58.2 Request

【 Mode 】 int CChex_ConnectionAuthentication(IntPtr CchexHandle, int DevIdx,ref CCHEX_CONNECTION_AUTHENTICATION_STRU Param);

【Parameter】CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

Param: 24byte, Input[Parameter];

```
        Byte[12] username;

        Byte[12] password;

    //Iris user name “admin”, password “admin”,

   //Other devices, don’t identify user name, only identify communication password.
```

2.58.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_CONNECTION_AUTHENTICATION_TYPE

```
Data Part：

   uint MachineId;        // Device ID

   int Result;            // 0:OK, -1:error
```

2.58.4 Sample

int ret = `CChex_ConnectionAuthentication(CchexHandle,DevIdx, ref Param);`

2.58.5 Notice

1. Make sure that CChex_Start has been started successfully before run.；

2.59 CChex_GetRecordNumByEmployeeIdAndTime

2.59.1 Description function

【Function】Get the record quantity by user ID and time. Only support A20/972 hardware platform devices.

Get the record quantity by user ID and time.

2.59.2 Request

【 Mode 】 int CChex_GetRecordNumByEmployeeIdAndTime(IntPtr CchexHandle, int DevIdx, ref CCHEX_GET_RECORD_INFO_BY_TIME Param);

【Parameter】 CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

Param: 13byte, Input[Parameter];

```
    Byte[5] EmployeeId;       //User ID
    Byte[4] start_date;       //The number of seconds after 2000.1.2
Byte[4] end_date;         //The number of seconds after 2000.1.2
```

User ID = 0xFF means all users

Start date = 0xFF, means unlimit the start date

End date = 0xFF, means unlimit the end date

2.59.3 Response

【Return value】 1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】 CChex_Update, Type return CCHEX_RET_GET_RECORD_NUMBER_TYPE

Data Part:

```
uint MachineId;       // Device ID
int Result;           // 0:OK, -1:error
int record_num;       // record quantity
```

2.59.4 Sample

int ret = CChex_GetRecordNumByEmployeeIdAndTime(CchexHandle,DevIdx, ref Param);

2.59.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.60 CChex_DownloadRecordByEmployeeIdAndTime

2.60.1 Description function

【Function】Get the record by user ID and time. Only support A20/972 hardware platform devices.

Get the record by user ID and time。

2.60.2 Request

【 Mode 】 int CChex_DownloadRecordByEmployeeIdAndTime(IntPtr CchexHandle, int DevIdx, ref CCHEX_GET_RECORD_INFO_BY_TIME Param);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter];

```
DevIdx: Index of device，Input[Parameter];

Param：13byte，Input[Parameter];

      Byte[12] EmployeeId;      //user ID

      Byte[4] start_date;       //The number of seconds after 2000.1.2

      Byte[4] end_date;         //The number of seconds after 2000.1.2

   User ID = 0xFF means all users

   Start date = 0xFF, means unlimit the start date

   End date = 0xFF, means unlimit the end date
```

2.60.3Response

【Return value】1：The command was executed successfully.; Minus：Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_GET_RECORD_BY_EMPLOYEE_TIME_TYPE

```
Data Part:

    uint MachineId;             // Device ID

    int Result;                 // 0:OK, -1:error

    Byte[5]  EmployeeId;        //User ID

    Byte[4]  char date;         //date time

    Byte     char back_id;      //back ID

    Byte     ecord_type;        //record type

    Byte[3]  work_type;     //work code

    Byte[2]  padding;       //alignment invalid data
```

Note: The number of times this type of data is returned based on the number of records recorded during the output download time

2.60.4 Sample

int ret = `CChex_DownloadRecordByEmployeeIdAndTime(CchexHandle,DevIdx,ref Param);`

2.60.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.61 Send attendance records in real time

2.61.1 Description function

【Function】Send the records in real time

After verification, the attendance information will be output automatically, and only the response information will be available.

2.61.2 Request

【Mode】

【Parameter】

2.61.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_LIVE_SEND_ATTENDANCE_TYPE

```
Data Part：

    uint MachineId;          // Device ID

    int Result;              // 0:OK, -1:error

    byte[5]  EmployeeId;     //user ID

    byte[4]  timestamp;      //The number of seconds after 2000.1.2

    Byte     backup;         //backup ID

    byte     record_type;    //record type

    byte[3]  work_type[3];   //workcode

    byte[2]  padding[2];
```

2.61.4Sample

2.61.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;
2. Set whether the device configuration information 2 is sent in real time

2.62 CChex_GetRecordInfo

2.62.1 Description function

【Function】Get the record information

Get the record information, contains already register user quantity, already register FP quantity, already register password quantity, all record quantity and new record quantity

2.62.2 Request

【Mode】int CChex_GetRecordInfoStatus(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

2.62.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update，Type return CCHEX_RET_GET_RECORD_INFO_STATUS_TYPE

Data Part：

```
uint MachineId;          // Device ID

uint EmployeeNum;        //User quantity
uint FingerPrtNum;       //FP quantity
uint PasswdNum;
uint CardNum;
uint TotalRecNum;
uint NewRecNum;
```

2.62.4 Sample

int ret = CChex_GetRecordInfoStatus(CchexHandle,DevIdx);

2.62.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.63 CChex_DownloadAllNewRecords

2.63.1 Description function

【Function】Download all record

2.63.2Request

【Mode】`int CChex_DownloadAllNewRecords(IntPtr CchexHandle, int DevIdx);`

【Parameter】CchexHandle：,successfully create the handle，Input[Parameter];

DevIdx: Search the device，Input[Parameter]

2.63.3 Response

【Return value】1：The command was executed successfully.; Minus：Execute command failure

【Actual Data】CChex_Update, Type return CCHEX_RET_GET_NEW_RECORD_INFO_TYPE

Data Part：

```
uint MachineId;       // Device ID
byte NewRecordFlag;   // new record or no
byte[5] EmployeeId;   // User ID,
byte[4] Date;         // Time, The number of seconds after 2000.1.2
byte BackId;          //Backup ID, 3 digits: Card, 2 digits: , 1~0: 11, digits
```

7~4: 10 FP templates (1~10)

byte RecordType // Record type, 7 digits: 1: open the door, 0: can' t open the door; 3~0 digits attendance status

byte[3] WorkType; // workcode

byte Rsv // reserved

2.63.4 Sample

int ret = CChex_DownloadAllNewRecords(anviz_handle, dev_idx);

2.63.5 Notice

1. Make sure that CChex_Start has been started successfully before run.;

2.64 Upload user information

【Function】upload user information

2.64.2 Request

【Mode】call the universal interface **2.20** CChex_ModifyPersonInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX_RET_PERSON_INFO_STRU personlist, byte person_num);

Call by the device type: dev->DevTypeFlag & 0xff

Universal 1: dev->DevTypeFlag & 0xff == 0x01

CChex_UploadEmployeeInfo(IntPtr *CchexHandle, int DevIdx, ref CCHEX_EMPLOYEE_INFO_STRU EmployeeList, byte EmployeeNum);

ASCII universal: dev->DevTypeFlag & 0xff == 0x02

CChex_UploadEmployee2Info(IntPtr *CchexHandle, int DevIdx, ref CCHEX_EMPLOYEE2_INFO_STRU

```
 EmployeeList, byte  EmployeeNum);

UNICODE universal: dev->DevTypeFlag & 0xff == 0x04

CChex_UploadEmployeeInfo(IntPtr *CchexHandle, int DevIdx, ref CCHEX_EMPLOYEE2UNICODE_INFO_STRU

EmployeeList, byte  EmployeeNum);

UNICODE_W2 universal: dev->DevTypeFlag & 0xff == 0x20

CChex_UploadEmployeeInfo(IntPtr *CchexHandle, int DevIdx, ref CCHEX_EMPLOYEE2W2_INFO_STRU

 EmployeeList, byte  EmployeeNum);
```

【Parameter】 CchexHandle: ,successfully create the handle, Input[Parameter];

DevIdx: Search the device, Input[Parameter]

EmployeeNum: Increase the number of employees (8 or 12 per upload per device type and customization)

Universal 1: dev->DevTypeFlag & 0xff == 0x01

```
 CCHEX_EMPLOYEE_INFO_STRU                //28byte

             byte[5] EmployeeId;         //user id  only one mark

             byte[3] Passwd;             //password

             byte[3] CardId;             //card ID

             byte[10] EmployeeName;      //name

             byte DepartmentId;          //department ID

             byte GroupId;               //Group ID

             byte Mode;                  //Attendance mode

             byte[2] FpStatus;           //FP registration status

             byte Special;               //Count of the in formations

             byte Padding;               //Invalid structure alignment
```

ASCII Universal: dev->DevTypeFlag & 0xff == 0x02

```
CCHEX_EMPLOYEE2_INFO_STRU               //32byte
             byte[5] EmployeeId;        //user id  only one mark
             byte[3] Passwd;            //Password
             byte[4] CardId;            //Card Number
             byte[10] EmployeeName;     //Name
             byte DepartmentId;         //Department ID
             byte GroupId;              //Group ID
             byte Mode;                 //Time attendance mode
             byte[2] FpStatus;          //Fingerprint Status
             byte PwdH8bit;             //password high 8 bits
             byte Rserved;              //Save
             byte Special;              //Count of the in formations
             Byte[2] Padding;           //Invalid structure alignment
```

UNICODE Universal: dev->DevTypeFlag & 0xff == 0x04

```
CCHEX_EMPLOYEE2UNICODE_INFO_STRU        //40byte
             byte[5] EmployeeId;        //user id  only one mark
             byte[3] Passwd;            //Password
             byte[4] CardId;            //Card Number
             byte[20] EmployeeName;     //Name
             byte DepartmentId;         //Department ID
             byte GroupId;              //Group ID
             byte Mode;                 //Time attendance mode
```

```
              byte[2] FpStatus;          //Fingerprint Status

              byte PwdH8bit;             //Password high 8 bits

              byte Rserved;              //Save

              byte Special;              //Count of the in formations


UNICODE_W2 Universal: dev->DevTypeFlag & 0xff == 0x20

CCHEX_EMPLOYEE2W2_INFO_STRU

CCHEX_EMPLOYEE2UNICODE_INFO_STRU         //40byte

              byte[5] EmployeeId;        //user id  only one mark

              byte[3] Passwd;            //Password

              byte[4] CardId;            //Card Number

              byte[20] EmployeeName;     //Name

              byte DepartmentId;         //Department ID

              byte GroupId;              //Group ID

              byte Mode;                 //Time attendance mode

              byte[2] FpStatus;          //Fingerprint Status

              byte PwdH8bit;             //Password high 8 bits

              byte Rserved;              //Save

              byte Special;              //Count of the in formations

                  byte[4]  start_date        //Employee starting time, The number of seconds after
              2000.1.2


                  byte[4]  end_date          //Employee ending time, The number of seconds after
              2000.1.2
```

2.64.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update, Type return

```
Universal 1：dev->DevTypeFlag & 0xff == 0x01

CCHEX_RET_ULEMPLOYEE_INFO_TYPE



ASCII Universal ：dev->DevTypeFlag & 0xff == 0x02

CCHEX_RET_ULEMPLOYEE2_INFO_TYPE

UNICODE Universal：dev->DevTypeFlag & 0xff == 0x04

CCHEX_RET_ULEMPLOYEE2UNICODE_INFO_TYPE

UNICODE_W2 Universal：dev->DevTypeFlag & 0xff == 0x20

CCHEX_RET_ULEMPLOYEE2W2_INFO_TYPE



 Data Part：

                  uint MachineId;       // Device ID

                  int Result;           // 0:OK, -1:error
```

2.64.4 Sample

int ret = `CChex_UploadEmployeeInfo(anviz_handle, dev_idx，EmployeeList，1);`

2.64.5 Notice

1. Make sure that CChex_Start has been started successfully before run.；

2.65 Chex_GetBasicConfigInfo5

2.65.1 Function

【Function】获取考勤机配置信息 5（Bolid 定制）

2.65.2 Request

【Mode】int CChex_GetBasicConfigInfo5(IntPtr CchexHandle, int DevIdx);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；参;

2.65.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex_Update Type Return CCHEX_RET_GET_BASIC_CFG5_TYPE

Data Part：

```
uint      MachineId;        // Device ID
Int       Result;           // 0:Succes  -1: Failure
byte      fail_alarm_time;  // 勤失败报警次数: 0: 不报警 1~10 N 次失败报警
byte      tamper_alarm;     // 防拆报警: 0: 关闭 1: 开启
byte[94] reserved;          // reserved
```

2.65.4 Sample

int ret = CChex_GetBasicConfigInfo5(anviz_handle, dev_idx);

2.65.5 Notice

1. Make sure that CChex_Start has been started successfully before run.

2.66 CChex_SetBasicConfigInfo 5

2.66.1 Function

【Function】Setup Time attendance device information 5（Bolid Customize）

2.66.2 Request

【Mode】int CChex_SetBasicConfigInfo5(IntPtr CchexHandle, int DevIdx,ref CCHEX_SET_BASIC_CFG_INFO5_STRU Param);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

Param: //96 byte （0xFF means do not setup）

```
byte     fail_alarm_time;  // Times of Attendance Fail Alarm: 0: Do not alarm out 1~10 times of alarm out
byte     tamper_alarm;     // Tamper alarm: 0: Close1: Active
byte[94] reserved;         // reserved
```

2.66.3 Respond

【Return value】1： The command was executed successfully； Minus： Execute command failure

【Actual Data】CChex_Update Type Return CCHEX_RET_SET_BASIC_CFG5_TYPE

Data Part：

```
uint     MachineId;       // Device ID
Int      Result;          // 0:Succes  -1: Failure
```

2.66.4 Sample

int ret = CChex_SetBasicConfigInfo5(anviz_handle, dev_idx,ref Param);

2.66.5 Notice

1. Make sure that CChex_Start has been started successfully before run.

Function

2.67 CChex_GetCardNo

2.67.1 Function

【Function】Query T5S swipe card information（Bolid Customize）

Notice: 1：The first time call the function ：Setting the device as swipe card mode

2：Swipe card over time > 5sec；

3：The second time call the function ：Return the last Card number before called the function.

2.67.2 Request

【Mode】`int Chex_GetCardNo(IntPtr CchexHandle, int DevIdx);`

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

2.67.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex_Update Type Return CCHEX_RET_GET_CARD_ID_TYPE

Data Part：

```
    uint     MachineId;        // Device ID
    Int      Result;           // 0:Succes  -1: Failure
    Uint    card_id;           //Card Number
```

2.67.4 Sample

int ret = Chex_GetCardNo(anviz_handle, dev_idx);

2.67.5 Notice

1. Make sure that CChex_Start has been started successfully before run.

2.68 CChex_SetDevCurrentStatus

2.68.1 Function

【Function】User to modify the device status by temporary（Bolid Customize）

2.68.2 Request

【Mode】int CChex_SetDevCurrentStatus(IntPtr CchexHandle, int DevIdx,ref CCHEX_SET_DEV_CURRENT_STATUS_STRU Param);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter];

DevIdx: Index of device，Input[Parameter];

```
Param:   //96 byte                （0xFF means do not setup）
    byte     alarm_stop;      //Disable the Alarm except 0XFF
    byte     door_status      //0: Setting the access control as normal 1: Setting
the access control as normal Open   2: Setting the access control as normal Close
    byte[94] reserved;        // reserved
```

2.68.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex_Update Type Return CCHEX_RET_SET_DEV_CURRENT_STATUS_TYPE

Data Part:

```
uint     MachineId;       // Device ID
Int      Result;          // 0:Succes  -1: Failure
```

2.68.4 Sample

int ret = `CChex_SetDevCurrentStatus(anviz_handle, dev_idx,ref Param);`

2.68.5 Notice

1. Make sure that CChex_Start has been started successfully before run.

2.69 CChex_GetServiceURL

2.69.1 Function

【Function】Get the server's URL

2.69.2 Request

【Mode】`int CChex_GetServiceURL(IntPtr CchexHandle, int DevIdx);`

【Parameter】CchexHandle: successfully create the handle, Input[Parameter];

DevIdx: Search the device, Input[Parameter]

2.69.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update return CCHEX_RET_GET_URL_TYPE

Data Part:

```
    uint        MachineId;        // device ID
    Int         Result;           // 0:Success  -1: Fail
    Byte[4]     Dns               //dns address
    byte[100]   Url;          //URL address
```

2.69.4 Sample

int ret = `CChex_GetServiceURL(anviz_handle, dev_idx);`

2.69.5 Notice

1. Make sure the `CChex_Start already active before running`

2.70 CChex_SetServiceURL

2.70.1 Function

【Function】Setuphe server's URL

2.70.2 Request

【Mode】`int CChex_SetServiceURL(IntPtr CchexHandle, int DevIdx,ref CCHEX_SET_URL_STRU Param);`

【Parameter】CchexHandle: successfully create the handle, Input[Parameter];

```
DevIdx: Search the  device, Input[Parameter]
Param:   //104 byte
    Byte[4]     Dns           //dns Address
    byte[100]   Url;          //URL Address
```

2.70.3 Response

【Return value】1： The command was executed successfully.； Minus： Execute command failure

【Actual Data】CChex_Update Type return CCHEX_RET_SET_URL_TYPE

```
Data Part：

    uint        MachineId;       // Device ID

    Int         Result;          // 0:Success  -1: Fail
```

2.70.4 Sample

int ret = `CChex_SetServiceURL(anviz_handle, dev_idx,ref Param);`

2.70.15 Notice

1.Make sure the `CChex_Start already active before running`

2.71 CChex_UploadFile

2.71.1 Function

【Function】Update firmware, Image and Voice

2.71.2 Request

【Mode】`int CChex_UploadFile(IntPtr CchexHandle, int DevIdx, byte FileType, byte[] FileName, byte[] Buff, int Len);`

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Search the device，Input[Parameter]

FileType:	= 0 Firmware，= 1 Image，= 2 Voice，= 3 Language
FileName:	The name update to the device (MAX_len = 10)；
Buff:	The uploaded file is read as a string;
Len：	The length of the string of the uploaded file.

2.71.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update Type return CCHEX_RET_UPLOADFILE_TYPE

Data Part：

```
uint        MachineId;        // Device ID
int         Result;           // 0:Success  -1：Fail
uint        TotalBytes;       // Total length of the uploaded string
uint        SendBytes;        // Length already uploaded
```

Notice：Every 32K of data is uploaded will return the type once SendBytes == TotalBytes，the upload is completed.

2.71.4 Sample

int ret = CChex_UploadFile(CchexHandle,DevIdx,FileType,FileName,Buff,Len);

2.71.5 Notice

1. Make sure the CChex_Start already active before running

2.72 CChex_UpdateDevStatus

2.72.1 Function

【Function】ask device the status of the firmware updating

Notice: After the upgrade package is uploaded, it is called multiple times. When the result is: 1: Check completion 2: The verification is successful. You can restart the upgrade

2.72.2 Request

【Mode】`int CChex_UpdateDevStatus(IntPtr CchexHandle, int DevIdx);`

【Parameter】CchexHandle: successfully create the handle, Input[Parameter];

DevIdx: Search the device, Input[Parameter]

2.72.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update Type return CCHEX_RET_UPDATEFILE_STATUS_TYPE

Data Part:

```
uint        MachineId;        // Device ID
Int         Result;           // 0:Success  -1: Fail
int         verify_status;    // 0:verifying ;1: Verify Success
int         verify_ret;       // 0:Verify Success;1:Verify Fail
```

2.72.4 Sample

int ret = CChex_UpdateDevStatus(anviz_handle, dev_idx);

2.72.5 Notice

1. Make sure the CChex_Start already active before running

2.73 CChex_GetStatusSwitch

2.73.1 Function

【Function】Get status switch information, total with 16 status

Notice: Device TypeDevTypeFlag & 0x200000 = = 1 can be called

2.73.2 Request

【Mode】`int CChex_GetStatusSwitch(IntPtr CchexHandle, int DevIdx,byte GroupId);`

【Parameter】CchexHandle: successfully create the handle, Input[Parameter];

DevIdx: Search the device, Input[Parameter]

GroupId: Edit Group ID

2.73.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update Type Return CCHEX_RET_GET_STATUS_SWITCH_TYPE

Data Part:

```
    uint        MachineId;      // Device IDInt          Result;          //
0:Success  -1: Fail
```

```
byte          group_id;         //get the group number
CCHEX_GET_PERIOD_TIME_ONE_STRU_EXT_INF day_week[7]; //7 groups as 7day for one
                                                     week. One  group
                                                     with 28Byte


            byte StartHour;    //Starting Hour
            byte StartMin;     //Starting Mins
            byte EndHour;      //Ending Hour
            byte EndMin;       //Ending Mins
byte          status_id;        //Setup status number
Byte[2]       padding;          //Invalid data, fill in the structure
```

2.73.4 Sample

int ret = CChex_GetStatusSwitch(anviz_handle, dev_idx,GroupId);

2.73.5 Notice

1. Make sure the CChex_Start already active before running

2.74 CChex_SetStatusSwitch

2.74.1 Function

【Function】Setup status switch information, total with 16 status

Notice: Device Type DevTypeFlag & 0x200000 = = 1 can be called

2.74.2 Request

【Mode】`int CChex_SetStatusSwitch(IntPtr CchexHandle, int DevIdx, ref`

CCHEX_SET_STATUS_SWITCH_STRU Param);

【Parameter】CchexHandle: successfully create the handle, Input[Parameter];

```
DevIdx: Search the  device, Input[Parameter]

Param:   //   32 byte

    byte          group_id;         //get the group number

    CCHEX_GET_PERIOD_TIME_ONE_STRU_EXT_INF day_week[7]; // 7 groups as 7day for one
                                                         week. One  group
                                                         with 28Byte

                byte StartHour;    //Starting Hour

                byte StartMin;     //Starting Mins

                byte EndHour;      //Ending Hour

                byte EndMin;       //Ending Mins

    byte          status_id;        //Setup status number

    Byte[2]       padding;          //Invalid data, fill in the structure
```

2.74.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update Type Return CCHEX_RET_SET_STATUS_SWITCH_TYPE

```
 Data Part:

    uint          MachineId;        // Device IDInt           Result;          //
0:Success  -1: Fail
```

2.74.4 Sample

int ret = CChex_SetStatusSwitch(anviz_handle, dev_idx,ref Param);

2.74.5 Notice

1. Make sure the CChex_Start already active before running

2.75 CChex_GetStatusSwitch_EXT

2.75.1 Function

【Function】Get status switch information

Notice: Device TypeDev TypeFlag & 0x100000 = = 1 can be called

2.75.2 Request

【Mode】`int CChex_GetStatusSwitch_ext(IntPtr CchexHandle, int DevIdx,byte FlagWeek);`

【Parameter】CchexHandle: successfully create the handle, Input[Parameter];

DevIdx: Search the device, Input[Parameter]

FlagWeek: Week day as 0-6, to represent Sun. to Sat., Such as Monday : 00000010

2.75.3 Response

【Return value】1: The command was executed successfully.; Minus: Execute command failure

【Actual Data】CChex_Update Type return CCHEX_RET_GET_STATUS_SWITCH_EXT_TYPE

Data Part:

```
   uint        MachineId;       // Device IDInt          Result;          //
0:Success  -1: Fail
   byte        flag_week;       //get the group number
   CCHEX_ONE_TIMER_STATUS one_time[8]; //8 time zone 40Byte
```

```
            byte StartHour;    //Starting Hour
            byte StartMin;     //Starting Mins
            byte EndHour;      //Ending Hour
            byte EndMin;       //Ending Mins
            byte status_id;    //Setup status number
    Byte[3]      padding;      //Invalid data, fill in the structure
```

2.75.4 Sample

int ret = `CChex_GetStatusSwitch_EXT(anviz_handle, dev_idx,flag_week);`

2.75.5 Notice

2. Make sure the CChex_Start already active before running

2.76 CChex_SetStatusSwitch_EXT

2.76.1 Function

【Function】Setup time attendance status information

Notic: Device type DevTypeFlag & 0x100000 = = 1 can be call

2.76.2 Request

【Mode】`int CChex_SetStatusSwitch_EXT(IntPtr CchexHandle, int DevIdx, ref CCHEX_SET_STATUS_SWITCH_STRU_EXT Param);`

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Search the device，Input[Parameter]

Param: // 44 byte

```
byte          flag_week;        //get the group number
CCHEX_ONE_TIMER_STATUS one_time[8]; //8 timezone  40Byte
            byte StartHour;     //Starting Hour
            byte StartMin;      //Starting Mins
            byte EndHour;       //Ending Hour
            byte EndMin;        //Ending Mins
            byte status_id;     //Setup status number
Byte[3]       padding;          //Invalid data, fill in the structure
```

2.76.3 Response

【Return value】1：The command was executed successfully.；Minus：Execute command failure

【Actual Data】CChex_Update Type return CCHEX_RET_SET_STATUS_SWITCH_EXT_TYPE

```
  Data Part：
    uint        MachineId;       // Device IDInt          Result;          //
0:Success  -1：Fail
```

2.76.4 Sample

int ret = `CChex_SetStatusSwitch_EXT(anviz_handle, dev_idx,ref Param);`

2.76.5 Notice

1. Make sure the CChex_Start already active before running

2.77 CChex_Get_Service_Port

2.77.1 Function

【Function】Setup the current server port

2.77.2 Request

【Mode】`int CChex_Get_Service_Port(IntPtr CchexHandle);`

【Parameter】`CchexHandle: successfully create the handle, Input[Parameter];`

2.77.3 Sample

int ret = `CChex_Get_Service_Port(CchexHandle);`

【Return value】Return`ret as server port`

2.78 CChex_SetSdkConfig

2.78.1 Function

【Function】Monitor the records status, the new record will automatic download and after download new records automatic to make a flag and SDK will print out to the LOG folder.

* after CChex_Start();

* para :

* SetRecordflag = 1 set recordflag after download new record;else = 0

* SetLogFile = 1 set some info log to file for find problem ;else = 0

* if do not set "CChex_SetSdkConfig(void *CchexHandle, int SetAutoDownload,int SetRecordflag,int SetLogFile)", config is default

* ANVIZ_DEFAULT:

* W2 : SetRecordflag = 1,SetLogFile = 0,SetAutodownload = 1;

* SEATS : SetRecordflag = 1,SetLogFile = 0,SetAutodownload = 1;

* DR : SetRecordflag = 0,SetLogFile = 0,SetAutodownload = 1;

* COMMON : SetRecordflag = 0,SetLogFile = 0,SetAutodownload = 1;

* Bolid : SetRecordflag = 1,SetLogFile = 0,SetAutodownload = 1;

2.78.2 Request

【Mode】void CChex_SetSdkConfig(IntPtr CchexHandle, int SetAutoDownload, int SetRecordflag, int SetLogFile);

【Parameter】SetAutodownload: = 1: Monitor records status , when get new records automatic download record, others do not download automatic

SetRecordflag: = 1:After download new records automatic to make a flag.

SetLogFile: = 1:SDK Print out to Log file

2.78.3 Sample

int ret = CChex_SetSdkConfig(CchexHandle, 1, 1, 0);

;

2.79 CChex_UploadRecord

2.79.1 Function

【Function】Upload Time Attendance Records

2.79.2 Request

【Mode】int CChex_UploadRecord(IntPtr CchexHandle, int DevIdx, ref CCHEX_UPLOAD_RECORD_INFO_STRU Param);

【Parameter】CchexHandle: successfully create the handle, Input[Parameter];

DevIdx: Index of device, Input[Parameter];

Param: //

```
Byte[5]  EmployeeId;      //Employee Id
Byte[4]  char date;       //Date
```

```
Byte      char back_id;     //Backup ID

Byte      ecord_type;       //Record Type

Byte[3]  work_type;    //Work Code
```

2.79.3 Respond

【Return value】1： The command was executed successfully；Minus： Execute command failure

【Actual Data】CChex_Update Type Return CCHEX_RET_UPLOAD_RECORD_TYPE   //87

```
    Data Part：

     uint          MachineId;         // Device ID

     Int           Result;            // 0:Succes  -1: Failure

     Byte[5]       EmployeeId;            // digital of Employee Id

     Byte[3]       padding;           // 无效数据结构对齐
```

2.79.4 Sample

int ret = CChex_UploadRecord(anviz_handle, dev_idx,ref Param);

2.79.5 Notice

2. Make sure that CChex_Start has been started successfully before run.

2.80 CChex_UploadRecord_VER_4_NEWID

2.80.1 Function

【Function】Upload Time Attendance Records

(Version: Device Type DevTypeFlag & 0XFF ==DEV_TYPE_VER_4_NEWID)

2.80.2 Request

【Mode】 int CChex_UploadRecord_VER_4_NEWID(IntPtr CchexHandle, int DevIdx, ref CCHEX_UPLOAD_RECORD_INFO_STRU_VER_4_NEWID Param);

【Parameter】 CchexHandle: successfully create the handle, Input[Parameter];

```
DevIdx: Index of device, Input[Parameter];

Param:   //

    Byte[28]  EmployeeId;      // String of Employee Id

    Byte[4]  date;         //Date

    Byte     back_id;      //Backup ID

    Byte     ecord_type;       //Record Type

    Byte[3]  work_type;    //Work Code
```

2.80.3 Respond

【Return value】 1: The command was executed successfully; Minus: Execute command failure

【Actual Data】 CChex_Update Type Return CCHEX_RET_UPLOAD_RECORD_TYPE   //87

```
Data Part:

  uint         MachineId;        // Device ID

  Int          Result;           // 0:Succes  -1: Failure

  Byte[28]     EmployeeId;           // String of Employee Id
```

2.80.4 Sample

int ret = CChex_UploadRecord_VER_4_NEWID(anviz_handle, dev_idx,ref Param);

2.80.5 Notice

3. Make sure that CChex_Start has been started successfully before run.

2.81 CChex_GetOnePersonInfo

(Version :digital of Employee Id)

CChex_GetOnePersonInfo_VER_4_NEWID

(Version:

String of Employee Id DevTypeFlag & 0XFF ==DEV_TYPE_VER_4_NEWID)

2.81.1 Function

【Function】According to employee Id to get user information

2.81.2 Request

【Mode】`int CChex_GetOnePersonInfo(IntPtr CchexHandle, int DevIdx, ref CCHEX_GET_ONE_EMPLOYEE_INFO_STRU Param);`

【Parameter】CchexHandle: successfully create the handle, Input[Parameter];

```
DevIdx: Index of device, Input[Parameter];

Param:   //

    Byte[5]  EmployeeId;      //digital of Employee Id
```

【Mode】`int CChex_GetOnePersonInfo_VER_4_NEWID(IntPtr CchexHandle, int DevIdx, ref CCHEX_GET_ONE_EMPLOYEE_INFO_STRU_VER_4_NEWID Param);`

(Version :Device Version DevTypeFlag & 0XFF ==DEV_TYPE_VER_4_NEWID)

【Parameter】CchexHandle: successfully create the handle, Input[Parameter];

```
DevIdx: Index of device, Input[Parameter];
```

Param： //

```
Byte[28]  EmployeeId;      //String of Employee Id
```

2.81.3 Respond

【Return value】1： The command was executed successfully； Minus： Execute command failure

【Actual Data】CChex_Update Type Return CCHEX_RET_GET_ONE_EMPLOYEE_INFO_TYPE   //88

Data Part：

digital of Employee Id ,Without employee valid time:

```
uint MachineId;       // Device ID
int CurIdx;           // Current index ID
int TotalCnt;         // Total number of User
byte [5]EmployeeId;   // User ID 5 bytes
byte password_len;    // Length of the Password
byte max_password;    // Max length of the Password, the value is 6  Can not modify
int password;         // Password
byte max_card_id;     // Max length of the Card Number, the value is 6(3 byte)
or 10(3 byte) Can not modify
uint card_id;         // Card Number
byte max_EmployeeName;    // Max length of the User Name, the value is 10,20,64
or 64(3 byte) When the value is 160 the user name store in the EmployeeName2
byte[64] EmployeeName;    // User Name
byte DepartmentId;        // Department ID
byte GroupId;             // Group ID
byte Mode;                // Time Attendance Mode
```

```
                  uint  Fp_Status;          // The status of FP enroll, 0~9:fp; 10:face; 11:iris1;
12:iris2

                  byte Rserved1;            // for 22

                  byte Rserved2;            // for 72 and 22

                  byte Special;             // Special information


                  // DR info

                  byte[160] EmployeeName2;  // User Name2

                  byte[13] RFC;             // RFC Information

                  byte[18] CURP;            // CURP Information
```

digital of Employee Id ,With employee valid time:

```
                  uint MachineId;       // Device ID

                  int CurIdx;           // Current index ID

                  int TotalCnt;         // Total number of User

                  byte [5]EmployeeId;   // User ID 5 bytes

                  byte password_len;    // Length of the Password

                  byte max_password;    // Max length of the Password, the value is 6  Can not modify

                  int password;         // Password

                  byte max_card_id;     // Max length of the Card Number, the value is 6(3 byte)
or 10(3 byte) Can not modify

                  uint card_id;         // Card Number

                  byte max_EmployeeName;    // Max length of the User Name, the value is 10,20,64
or 64(3 byte) When the value is 160 the user name store in the EmployeeName2

                  byte[64] EmployeeName;    // User Name

                  byte DepartmentId;        // Department ID

                  byte GroupId;             // Group ID
                  byte Mode;                  // Time Attendance Mode

                  uint  Fp_Status;            // The status of FP enroll, 0~9:fp; 10:face; 11:iris1;
12:iris2

                  byte Rserved1;              // for 22

                  byte Rserved2;              // for 72 and 22

                  byte Special;               // Special information


                  // DR info

                  byte[160] EmployeeName2;   // User Name2

                  byte[13] RFC;               // RFC Information

                  byte[18] CURP;              // CURP Information

                  byte[4] start_date;                  // RFC Information

                  byte[4] end_date;           // CURP Information
```

String of Employee Id , with employee valid time:

```
                  uint MachineId;        // Device ID

                  int CurIdx;            // Current index ID

                  int TotalCnt;          // Total number of User

                  byte [28]EmployeeId;   // Enployee ID,28bytes

                  byte password_len;     // Length of the Password

                  byte max_password;     // Max length of the Password, the value is 6  Can not modify

                  int password;          // Password

                  byte max_card_id;      // Max length of the Card Number, the value is 6(3 byte)
or 10(3 byte) Can not modify

                  uint card_id;          // Card Number

                  byte max_EmployeeName;     // Max length of the User Name, the value is 10,20,64
or 64(3 byte) When the value is 160 the user name store in the EmployeeName2
```

```
byte[64] EmployeeName;     // User Name
byte DepartmentId;         // Department ID
byte GroupId;              // Group ID
byte Mode;                 // Time Attendance Mode
uint  Fp_Status;           // The status of FP enroll, 0~9:fp; 10:face; 11:iris1;
12:iris2
byte Rserved1;             // for 22
byte Rserved2;             // for 72 and 22
byte Special;              // Special information

// DR info
byte[160] EmployeeName2;   // User Name2
byte[13] RFC;              // RFC Information
byte[18] CURP;             // CURP Information
byte[4] start_date;                 // RFC Information
byte[4] end_date;          // CURP Information
```

2.81.4 Sample

```
int ret = CChex_UploadRecord(anviz_handle, dev_idx,ref Param);
int ret = CChex_UploadRecord_VER_4_NEWID(anviz_handle, dev_idx,ref Param);
```

2.81.5 Notice

4. Make sure that CChex_Start has been started successfully before run.

2.82 CChex_GetMachineId

2.82.1 Function

【Function】Read the device communication device ID

2.82.2 Request

【Mode】`int CChex_GetMachineId(IntPtr CchexHandle, int DevIdx);`

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

2.82.3 Respond

【Return value】1： The command was executed successfully；Minus： Execute command failure

【Actual Data】CChex_Update Type Return CCHEX_RET_GET_MACHINE_ID_TYPE   //84

```
Data Part：

  uint        MachineId;       // Device ID

  Int         Result;          // 0:Succes  -1: Failure

  uint     cur_machineid;          //Communicatrion device ID
```

2.82.4 Sample

int ret = `CChex_GetMachineId(anviz_handle, dev_idx);`

2.82.5 Notice

1. Make sure that CChex_Start has been started successfully before run.

2.83 CChex_SetMachineId

2.83.1 Function

【Function】Setting device Communication device ID

2.83.2 Request

【Mode】int CChex_SetMachineId(IntPtr CchexHandle, int DevIdx,uint MachineId);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter]；

DevIdx: Index of device，Input[Parameter]；

MachineId: device Communication device ID;

2.83.3 Respond

【Return value】1： The command was executed successfully； Minus： Execute command failure

【Actual Data】CChex_Update Type Return CCHEX_RET_SET_MACHINE_ID_TYPE   //85

Data Part：

```
uint        MachineId;        // Device ID
Int         Result;           // 0:Succes  -1: Failure
uint     cur_machineid;           //New Communicatrion device ID
uint     old_machineid;           //Old Communicatrion device ID
```

2.83.4 Sample

```
int ret = CChex_SetMachineId(anviz_handle, dev_idx,MachineId);
```

2.83.5 Notice

1. Make sure that CChex_Start has been started successfully before run.

2.84 CChex_ManageLogRecord

2.84.1 Function

【Function】Setting device Communication device ID

2.84.2 Request

【Mode】int CChex_ManageLogRecord(IntPtr CchexHandle, int DevIdx, ref CCHEX_MANAGE_LOG_RECORD Param);

【Parameter】CchexHandle：successfully create the handle，Input[Parameter];

```
DevIdx: Index of device，Input[Parameter];

Param:   //

    Byte[4]  start_date;      // The second since of  after 2000 years, 2000.1.2

    Byte[4]  end_date;        //The second since of  after 2000 years, 2000.1.2

    Byte     CmdType;     //Extended command :   0x00 Get the total number of current
time periods

                          0x01 Download the current time period log record

                          0x02 delete all records in the current time period

                          0x03 Set real-time report log record

                          0x04 Get the status of real-time reporting records

   Byte    AutoFlag;        //0:Close real-time report log record 1: Enable real-time
report log record
```

2.84.3 Respond

【Return value】1：The command was executed successfully；Minus：Execute command failure

【Actual Data】CChex_Update Type Return CCHEX_RET_MANAGE_LOG_RECORD_TYPE   //90

```
  Data Part：

    uint         MachineId;        // Device ID

    uint         CmdType;          // Extended command

                                     0x00 :Result judge success,TotalNum is Total
                                   number of the log

                                      0x02 :Result is valid Delete success or not

                                      0x03 Set real-time report log record

                                      0x04 Get the status of real-time reporting
records

                                        0x05 Real-time reporting record

    Int          Result;           // 0:Succes  -1: Failure

    uint         IsAuto;           // Auto-tagging

    uint         TotalNum;         // Total number of the log

                                   CmdType == 0x00:Result==0  Total number of the
log

                                   CmdType == 0x01:Result==0 Total download number
                              of the log

    uint         CurNum;           //  The log id of the current logByte[5]
    EmployeeId;       //Employee Id

    Byte[4]      Date;            //Event date The second since of  after 2000 years,
2000.1.2

    Byte[2]      LogType;          //Log Type 0x0001 open door 0x0002 close door
```

```
                                    0x0003 door sensor 0x0004 Tamper alarm
                                    0x0005 Exit button  0x0006 burst open the
                                door
Byte[2]      LogLen;            //Log content length
Byte[3]      padding;           //Fill alignment invalid data
```

2.84.4 Sample

int ret = `CChex_ManageLogRecord(anviz_handle, dev_idx,ref Param);`

2.84.5 Notice

1. Make sure that CChex_Start has been started successfully before run.

3 Return value

1：Success；

-1：Parameter Error；

-2：Lack of device resources；

-3：Unknown Error；