SFC3KPC

OCX 2.6.2
Specification

CAMURATECH

CONTENTS

1. GetLastError

[FUNCTION]

Get Last Error.
With this function, be able to know 'erro' message when a call of the course of event'OCX function is failed.

[Format]

```
boolean GetLastError(
       long* dwErrorCode
       );
```

[Parameter]

dwErrorCode : long value pointer that storages error values

Name	Value	Explanation
Z_SERIES_ERR_SUCCESS	0	No error
Z_SERIES_ERR_COM_CREATE	1	Cannot open COM PORT
Z_SERIES_ERR_COM_SETINFO	2	Cannot set up COM PORT
Z_SERIES_ERR_SOCK_CREATE	3	Cannot create Windows Socket
Z_SERIES_ERR_SOCK_SETOPTION	4	Cannot set Windows Socket option
Z_SERIES_ERR_SOCK_CONNECT	5	Cannot connect
Z_SERIES_ERR_SOCK_RECONNECT	6	Cannot reconnect (correspond COMPORT)
Z_SERIES_ERR_SOCK_INVALID_PASS	7	Communication password does not match
Z_SERIES_ERR_WRITE_FAIL	101	Write fail
Z_SERIES_ERR_READ_FAIL	102	Read fail
Z_SERIES_ERR_INVALID_PARAM	103	Parameter fail
Z_SERIES_ERR_INVALID_DATA	104	Data fail
Z_SERIES_ERR_NON_CARRYOUT	501	Cannot operate
Z_SERIES_ERR_LOG_END	502	Finish reading data
Z_SERIES_ERR_MULTIUSER	503	Fingerprint data or card data has been duplicated

[ReturnValue]

No meaning.

2. ConnectSerial

[FUNCTION]

USB OR COM PORT Connection.

[Format]

```
boolean ConnectSerial(
       long dwMachineNumber,
       long dwCommPort,
       long dwBaudRate
       );
```

[Parameter]

dwMachineNumber : Controller Address (1-16)
dwCommPort : USB OR COM PORT Number

Name	Value
USB	0
COM1	1
COM2	2
COM3	3
COM4	4

dwBaudRate : Baud rate (bps)

Value
9600
19200
38400
57600
115200

[ReturnValue]

TRUE – Success, FALSE – Fail. (TRUE = 0 Non value, FALSE = 0)

3. ConnectTcpip

[FUNCTION]

TCP/IP Connection.

[Format]

```
boolean ConnectTcpip(
       long dwMachineNumber,
       BSTR lpszIPAddress,
       long dwPortNumber,
       long dwPassWord
       );
```

[Parameter]

dwMachineNumber : Controller Address
lpszIPAddress : IP Address (Example : "192.168.1.224")
dwPortNumber : Port Number, (Controller start value 5005)
dwPassWord : Communication Password, (Controller start value 0)

[ReturnValue]

TRUE – Success, FALSE – Fail.

4. Disconnect

[FUNCTION]

Disconnect connection using either ConnectSerial or ConnectTcpip

[Format]

```
void Disconnect();
```

[Parameter]

None

[ReturnValue]

None.

5. EnableDevice

[FUNCTION]

Convert controller either enable or able.
If the controller becomes enable, it shows "=PC INTERFACE=" so that manipulation is not possible. It can be applied to communicate with PC. Even if the controller becomes enable, it will be applied to communicate with PC. But there is no indication of communication sing with PC.

[Format]

```
boolean EnableDevice(
       long dwMachineNumber,
       BOOL bFlag
       );
```

[Parameter]

dwMachineNumber : Controller Address
bFlag : FALSE – It converts controller to enable.
TRUE – It converts controller to able.

[ReturnValue]

TRUE – Success, FALSE – Fail.

6. GetEnrollData

[FUNCTION]

Get enroll data from the controller.

[Format]

```
boolean GetEnrollData(
      long dwMachineNumber,
      long dwEnrollNumber,
      long* dwEnrollData
      );
```

[Parameter]

dwMachineNumber : Controller address
dwEnrollNumber : User ID (0-999999)
dwEnrollData : EnrollDataType variable pointer, when actually call, it will be converted to long variable pointer.

```
Public Const FP_SIZE_COM As Long = (1404 + 12)              //Fingerprint data size (Byte)
Public Const ENROLL_DATA_SIZE As Long = (4 * 8 + FP_SIZE_COM * 2)  //EnrollDataType size
Type EnrollDataType
    dwValidTempId As Long               //1-User Temp ID
    dwValidCard As Long                 //1-Card data exist
    dwValidFp1 As Long                  //1-First fingerprint data exist
    dwValidFp2 As Long                  //1-Second fingerprint data exist
    dwTempIdNumber As Long              //user Temp ID (It shows actual accessed number)
    dwManager As Long                   //0-user, 1-administrator
    dwCardData(2 - 1) As Long           //Card data, here dwCardData(0) – low number 32bit
    byFpData1(FP_SIZE_COM - 1) As Byte  //First fingerprint data
    byFpData2(FP_SIZE_COM - 1) As Byte  //Second fingerprint data
End Type
```

[ReturnValue]

TRUE – Success, FALSE – Fail.

7. SetEnrollData

[FUNCTION]

Sending enrolled data to controller.

[Format]

```
boolean SetEnrollData(
      long dwMachineNumber,
      long dwEnrollNumber,
      long* dwEnrollData,
      long dwEnrollMode
      );
```

[Parameter]

dwMachineNumber : Controller address
dwEnrollNumber : User ID
dwEnrollData : EnrollDataType variable pointer, when actually call, it wil be converted long variable pointer.
dwEnrollMode : Transmittion mode of users' registration informaiton
0 – Register only controller(D SERIES /FPC-301SF)

1 - Register controller and telecop main device at once(KTL801/KTT801NS)

[Warning]
dwEnrollData.dwValidTempId, dwEnrollData.dwValidCard, dwEnrollData.dwValidFp1, dwEnrollData.dwValidFp2, if all are 0, delete pertinent user ID.

When dwEnrollData.dwValidTempId is 1, dwEnrollData.dwTempIdNumber will be applied, and the rest will not be applied.

[ReturnValue]
TRUE – Success, FALSE – Fail.

8. GetUserInfo

[Function]
Get user info from controller.

[Format]
```
boolean GetUserInfo(
       long dwMachineNumber,
       long dwEnrollNumber,
       long* dwUserInfo
       );
```

[Parameter]
dwMachineNumber : Controller address
dwEnrollNumber : User ID
dwUserInfo : UserInfoType variable pointer, when actually call, it will be converted to long variable pointer.

```
Public Const USER_INFO_SIZE As Long = 4 //UserInfoType's size (Long number)
Type UserInfoType
    dwTzone1 As Long       //access time 1  1-256-> number of time, 0-All, 257-None
    dwTzone2 As Long       //access time 2  1-256-> number of time, 0-All, 257-None
    dwAccessMode As Long   //0-Normal, 1-AnyOne
    dwLimitTime As Long    //reaccess prohibition time (minutes) 0-600
End Type
```

[ReturnValue]
TRUE – Success, FALSE – Fail.

9. SetUserInfo

[Function]
Set user info from controller.

[Format]
```
boolean SetUserInfo(
       long dwMachineNumber,
       long dwEnrollNumber,
       long* dwUserInfo
       );
```

[Parameter]
dwMachineNumber : Controller address
dwEnrollNumber : User ID
dwUserInfo : UserInfoType variable pointer, when actually call, it will be converted to long variable pointer.

[ReturnValue]
TRUE – Success, FALSE – Fail.

10. DeleteEnrollData

[Function]

Delete one user from controller.

[Format]

```
boolean DeleteEnrollData(
       long dwMachineNumber,
       long dwEnrollNumber,
       long dwDeleteMode
       );
```

[Parameter]

dwMachineNumber : Controller address
dwEnrollNumber : User ID
dwDeleteMode : Delete mode of users' registration information
0 – Delete only controller(D SERIES /FPC-301SF)
1 – Delete controller and telecop main device at once(KTL801/KTT801NS)

[ReturnValue]

TRUE – Success, FALSE – Fail.

11. EmptyEnrollData

[Function]

Delete all user from controller.

[Format]

```
boolean EmptyEnrollData(
       long dwMachineNumber
       long dwEmptyMode
       );
```

[Parameter]

dwMachineNumber : Controller address
dwEmptyMode : Delete mode of users' registration information
0 – Delete only controller(D SERIES /FPC-301SF
1 – Delete controller and telecop main device at once(KTL801/KTT801NS)

[ReturnValue]

TRUE – Success, FALSE – Fail.

12. ReadAllUserID

[Function]

All information in the controller will be stroaged in inner structure of OCX. GetAllUserID will abtain saved information one at a time. (actural card information or fingerprint information do not save)

[Format]

```
boolean ReadAllUserID(
       long dwMachineNumber
       );
```

[Parameter]

dwMachineNumber : Controller address

[ReturnValue]

TRUE – Success, FALSE – Fail.

13. GetAllUserID

[Function]

ReadAllUserID will abtain saved information in inner structure of OCX one at a time.

[Format]

```
boolean GetAllUserID(
       long dwMachineNumber,
       long* dwEnrollNumber,
       long* dwEnrollData
       );
```

[Parameter]

dwMachineNumber : Controller address
dwEnrollNumber : long variable pointer that storage User ID
dwEnrollData : EnrollDataType variable pointer that will storage registration information, when actually call, it will be converted to long variable pointer.
[Warning]
dwEnrollData.dwTempIdNumber, dwEnrollData.dwCardData, dwEnrollData.byFpData1, dwEnrollData.byFpData1

[ReturnValue]

TRUE – Succes, FALSE – Fail.
When FALSE GetLastError value is SFC3KPCERR_LOG_END, it means every information is read.

14. ModifyPrivilege

[Function]

Modify user privilege.

[Format]

```
boolean ModifyPrivilege(
       long dwMachineNumber,
       long dwEnrollNumber,
       long dwMachinePrivilege
       );
```

[Parameter]

dwMachineNumber : Controller address
dwEnrollNumber : User ID
dwMachinePrivilege : Privilege, 0-User, 1-Administrator

[ReturnValue]

TRUE – Success, FALSE – Fail.

15. ReadSuperLogData

[Function]

Storage all managing materials in inner structure of OCX
And reading point moves to the end.
Storaged contents will be abtained one at a time by GetSuperLogData

[Format]

```
boolean ReadSuperLogData(
       long dwMachineNumber
       );
```

[Parameter]

dwMachineNumber : Controller address

[ReturnValue]

TRUE - Success, FALSE - Fail.

16. GetSuperLogData

[Function]

Abtain deposited archives in inner structure of OCX one at a time.

[Format]

```
boolean GetSuperLogData(
       long  dwMachineNumber,
       long* dwEnrollNumber,
       long* dwDevice,
       long* dwManipulation,
       long* dwYear,
       long* dwMonth,
       long* dwDay,
       long* dwHour,
       long* dwMinute,
       long* dwSecond
       );
```

[Parameter]

dwMachineNumber : Controller address
dwEnrollNumber : long variable pointer to save user ID
(dwManipulation meaning changes as the value changes.)
dwDevice : Where manipulation takes place 0-Controller, 1-PC
dwManipulation : Manipulation (refer to the table below)
dwYear, dwMonth, dwDay, dwHour, dwMinute, dwSecond :Manipulation progress time

Possible dwManipulationis value is the same as below.

Name	Value	Explanation	dwEnrollNumber Value Meaning
ENROLL_USER_CARD	1	Enroll user card	Enrolled ID
ENROLL_MGR_CARD	2	Enroll manager card	"
ENROLL_USER_FP	3	Enroll user FP	"
ENROLL_MGR_FP	4	Enroll manager FP	"
ENROLL_TEMP	5	Enroll user temporary FP	"
ENROLL_FP_EDIT	6	Fingerprint edit	"
ENROLL_CARD_EDIT	7	Change card	
ENROLL_DELETE_ONE	8	Delete one enrolled user	"
ENROLL_DELETE_ALL	9	Delete all enrolled user	No Meaning
		(Nomal chage log)	(for detailed new setting value, please refer to SetDeviceInfo)
SET_DOOR_RELAY	10	Door Relay setup	Correspond setting value
SET_DOOR_MODE	11	Operation method(Boot-time) setup	" (Boot-time and Run-time)
SET_DOOR_TIME	12	Lock lock time setup	"
SET_DOOR_ALARM	13	DOOR open bell setup	"
SET_CD_TYPE	14	Wiegand method setup	"
SET_REACCESS	15	Verification security setup	"
SET_CD2_FUN	16	External reader function key setup	"
SET_ALARM_RELAY_FUN	17	External alarm function key setup (relay)	"
SET_ALARM_BUZZ_FUN	18	External alarm function key setup (buzzer)	"
SET_ANTIPASS	19	Anti-Pass Back setup	"

SET_FIRE	20	Fire watch/intrution watch	"
SET_IDSECRET	21	ID Security setup	"
SET_DOOR_LIMIT	22	Setup a limit access time	"
SET_LANGUAGE	23	Language setup	"
SET_VOICE_OUT	24	Voice out setup	"
		(communication installation material change log)	" (specific setup value SetDeviceInfo)
COMM_MACHINE_ID	25	Controller address setup	"
COMM_BAUDRATE	26	COM PORT Baudrate setup	"
COMM_PASSWORD	27	TCP/IPcommunication password setup	" (0-999999), 0- no passwod Start value – 0
COMM_PORTNUM	28	TCP/IP PORT number setup	" (1-9999) Start value – 5005
COMM_ISDHCP	29	TCP/IP DHCP setup	0- DHCP enable 1- DHCP able
COMM_IP	30	TCP/IP IP address setup	Start value 192.168.1.224 (example : a upper rank bite 192 The second bite 168 The third bite 1 A low rank bite 224)
COMM_SUBNET_MASK	31	TCP/IP subnet mask setup	Start value 255.255.255.0
COMM_DEF_GATEWAY	32	TCP/IP gateway setup	Start value 192.168.1.1
		(sechedule mangaement installation manual change log)	
OCX_DAYLIGHT	33	Change a terminal in a certain time	No meaning
OCX_HOLIDAY	34	Holiday setup	"
OCX_TZONE	35	Install access time zone	"
OCX_TMODE	36	Install operation mode in a certain time	"
OCX_BELL_TIME	37	Bell control setup	"
OCX_AUTODOOR	38	Autodoor control setup	"
OCX_AUTOKEY	39	Auto function key setup	"
OCX_NOACTKEY	40	Relay non-functionkey installation	"
		(extra log)	
_TIME	41	Controller time setup	1–TIME_GENERAL (when setup a time through menu or OCX) 2-TIME_DAYLIGHT (time setup by DayLight)
_BRI_ADJUST	42	Adjustment of fingerprint sensor brightness	No meaning
_DOOR_MODE	43	Operation door mode by a cetain time zone (Run-time) (when differ from Boot-Time mode)	"
_DOOR_OPEN	44	Opening the door	1-OPEN_EXIT (by Exitsignal) 2-OPEN_AUTODOOR (by automatic door signal) 3-OPEN_HOPEN (LOCKCTRL_HOPEN-by commend, refer to the manual of door mode) 4-OPEN_FIRE (by fire watch)
_DOOR_CLOSE	45	Closing the door	1-CLOSE_HCLOSE

			(LOCKCTRL_HCLOSE) 2-CLOSE_INVASION- by commend, refer to the manual of door mode (by intrution watch)
_ALARM	46	Alarm	1-ALARM_FIRE (fire watch) 2-ALARM_INVASION (intrution watch) 3-ALARM_FORCED_OPEN (Forced open) 4-ALARM_OPEN_OVER (over time for opening door) 5-ALARM_FKEY (external alarm function key) 6-ALARM_BELL (bell control) 7-ALARM_COVER_OPEN (Cover Open – Tamper When switch comes off)
_POWER_ON	47	controller operating time	No meaning
_MENU_ACCESS	48	Menu access successful	Admin ID When -1, the administer is not resistered but 1234.
_MENU_FAIL	49	Menu access fail	entered ID
_MENU_EXIT	50	Menu exit	No meaning
_MEMORY_ALLDEL	51	Register material, Event all delete	"
_EVENT_ALLDEL	52	Event data all delete	"

[ReturnValue]

TRUE – success, FALSE – Fail.
When FALSE, GetLastError value is SFC3KPCERR_LOG_END, it means it has done reading to the end.

17. DeleteReadSuperLogData

[Function]

From the point of start to current reading point in regards to controller, it deletes management record materials.

[Format]

```
boolean DeleteReadSuperLogData(
	long dwMachineNumber
	);
```

[Parameter]

dwMachineNumber : Controller address

[ReturnValue]

TRUE – Success, FALSE – Fail.

18. Start Read Super Log Data

[Function]

From controller, move reading point to start point. It helps find alredy read materials.

[Format]

```
boolean StartReadSuperLogData(
       long dwMachineNumber
       );
```

[Parameter]

dwMachineNumber : Controller address

[ReturnValue]

TRUE – Success, FALSE – Fail.

19. Empty Super Log Data

[Function]

Delete all log data from controller.

[Format]

```
boolean EmptySuperLogData(
       long dwMachineNumber
       );
```

[Parameter]

dwMachineNumber : Controller address

[ReturnValue]

TRUE – Success, FALSE – Fail.

20. ReadGeneralLogData

[Function]

From the point of start to current reading point in regards to controller , they will be stored in inner structure of OCX.
And the reading point moves to the end.
Abtain deposited cotents through GetGeneralLogData.

[Format]

```
boolean ReadGeneralLogData(
       long dwMachineNumber
       );
```

[Parameter]

dwMachineNumber : Controller address

[ReturnValue]

TRUE – Success, FALSE – Fail.

21. GetGeneralLogData

[Function]

Abtain access record material from OCX through ReadGeneralLogData

[Format]

```
boolean GetGeneralLogData(
       long dwMachineNumber,
       long* dwEnrollNumber,
       long* dwGranted,
       long* dwMethod,
       long* dwDoorMode,
       long* dwFunNumber,
       long* dwSensor,
       long* dwYear,
```

```
	long* dwMonth,
	long* dwDay,
	long* dwHour,
	long* dwMinute,
	long* dwSecond
	);
```

[Parametr]

dwMachineNumber : Controller address
dwEnrollNumber : long variable point to save User ID
if -1, it means user ID is not clear.
dwGranted : User verification result, 0-Access denied 1-Access granted
dwMethod : Verification method

verify mode will have this bit structure as below(a low rank 8 bit is only valid)

Bit Number :

7	6	5	4	3	2	1	0
reservation	If 1, Prohibition by TIME ZONE	If 1, prohibition by ANTI PASS	If 1, LIMIT TIME (prohibition by a certain time zone)	R.V	If,1 Fingerprint verification	If, 1 Card verification	If, 1 ID verification

And if 0,1,2 bits are all 0, it means card verification through terminal installed outisde

dwDoorMode : operating mode when verified (Run-time)
dwFunNumber : function key when verified

Functionkey number = (F number – 1) * 10 + number.
Example) F3-4's number (3-1)*10 + 4 = 24
If function key number is 40, it means any function key number is not pressed.

dwSensor: In the state of being open the door when verified, 0-close, 1-open
dwYear, dwMonth, dwDay, dwHour, dwMinute, dwSecond : a time that verification is done

[ReturnValue]

TRUE – Success, FALSE – Fail.
When FALSE, if GetLastError value is SFC3KPCERR_LOG_END, it means it read until the end.

22. DeleteReadGeneralLogData

[Function]

From the point of start to current reading point in regards to controller , deletes access information.

[Format]

```
boolean DeleteReadGeneralLogData(
	long dwMachineNumber
	);
```

[parameter]

dwMachineNumber : Controller address

[ReturnValue]

TRUE – Success, FALSE – Fail.

23. StartReadGeneralLogData

[Function]

From controller, move reading point to starting point. It can be used when access information needs to be read again.

[Format]

```
boolean StartReadGeneralLogData(
	long dwMachineNumber
	);
```

[Parameter]

dwMachineNumber : Controller address

[ReturnValue]

TRUE – Success, FALSE – Fail.

24. EmptyGeneralLogData

[Function]

Delete all general log data from the controller.

[Format]

```
boolean EmptyGeneralLogData(
	long dwMachineNumber
	);
```

[Parameter]

dwMachineNumber : Controller address

[ReturnValue]

TRUE – Success, FALSE – Fail.

25. ClearKeeperData

[Function]

Delete all information such as recorded material, access information, etc.

[Format]

```
boolean ClearKeeperData(
	long dwMachineNumber
	);
```

[Parameter]

dwMachineNumber : Controller address

[ReturnValue]

TRUE – Success, FALSE – Fail.

26. GetProductCode

[Function]

Get product code from controller.
Setup product information as initial setup program

[Format]

```
boolean GetProductCode(
	long dwMachineNumber,
	BSTR* lpszProductCode
	);
```

[Parameter]

dwMachineNumber : Controller address
lpszProductCode : variable character that saves product information (Max. 32 letters)

[ReturnValue]

TRUE – Success, FALSE – Fail.

27. GetSerialNumber

[Function]

Get serial number from the controller.
The serial number is installed in the setup program.

[Format]

```
boolean GetSerialNumber(
       long dwMachineNumber,
       BSTR* lpszSerialNumber
       );
```

[parameter]

dwMachineNumber : Controller address
lpszProductCode : variable letters (Max; 32 letters)

[ReturnValue]

TRUE – success, FALSE – Fail.

28. GetDeviceStatus

[Function]

Get device status from controller.

[Format]

```
boolean GetDeviceStatus(
       long dwMachineNumber,
       long dwStatus,
       long* dwValue
       );
```

[Parameter]

dwMachineNumber : Controller address
dwStatus : indicating number for information
dwValue : long variable pointer

Name	dwStatus value	dwValue value
UserCount	1	Registered user ID number (TEMP ID included)
ManagerCount	2	Registered administer ID number
FpCount	3	Currentlyregisterd fingprint number
CardCount	4	Registered card number
SLogTotal	5	Registered terminal number
SLogRead	6	Management datareading point
GLogTotal	7	Recorded access number
GLogRead	8	Access reading point
CurDoorMode	9	Run-time operating mode

[ReturnValue]

TRUE – Success, FALSE – Fail.

29. GetDeviceInfo

[Function]

Get device information from controller.

[Format]

```
boolean GetDeviceInfo(
       long  dwMachineNumber,
       long  dwInfo,
       long* dwValue
       );
```

[Parameter]

dwMachineNumber : controller address
dwInfo : device information
dwValue : long variable pointer

Name	Explanation	dwInfovalue	dwValue value	
DoorRelay	Door Relay	1	1	Relay 1 – start value
			2	Relay 2
DoorMode	Boot-time Operating mode	2	0	[ANY MODE] – start value
			1	[FINGER]
			2	[CD] or [FP]
			3	[ID&FP]or[CD]
			4	[ID&FP]or[ID&CD]
			5	[ID&FP]or[CD&FP]
			6	[OPEN]
			7	[CLOSE]
			8	[CD]
			9	[ID] or [FP]
			10	[ID] or [CD]
			11	[ID & CD]
			12	[CD & FP]
			13	[ID & FP]
			14	[ID & CD & FP]
DoorTime	Lock Locking time	3	1-99 Sec., start value is 3 seconds.	
DoorAlarm	Door Opening bell	4	0 – enable (start value) 1-99 secod	
CdType	Wiegand method	5	always 1, 48bit Wiegand	
ReAccess	Verification security	6	0-9 , start value is 0	
Cd2Fun	External reader function key	7	40 – enable (start value) 0-39 (function key value's caculating method is GetGeneralLogData)	
AlmRelayFun	External alarm function key (Relay)	8	"	
AlmBuzzFun	External alarm (Buzzer)	9	"	
AntiPass	AntiPass	10	0	No (start value)
			1	Yes
Fire	Fire watch/ Intrution watch	11	0	Intrution watch(start value)
			1	Fire watch
IdSecret	IDsecurity	12	0	Off (start value)
			1	On
Limit	Prohibit in a certain time	13	0 – enable (start value) 1-600 minutes	

			0	English
			1	Korean (start value)
Language	Language	14	2	Chinese
			0	Off
VoiceOut	Voice out	15	1	On (start value)
MachineID	Controller address	16	1-16 (start value is 1.)	
			0	9600 bps
			1	19200 bps
			2	38400 bps
			3	57600 bps
Baudrate	Baudrate	17	4	115200 bps (start value)
Machine Type	Device type	100	Value for device's type(Annex-B)	

[ReturnValue]

TRUE – Success, FALSE – Fail.

30. SetDeviceInfo

[Function]

Change device information.

[Format]

```
boolean SetDeviceInfo(
       long dwMachineNumber,
       long dwInfo,
       long dwValue
       );
```

[Parameter]

dwMachineNumber : Controller address
dwInfo : device information
dwValue : device info. value (refer to GetDeviceInfo)

[ReturnValue]

TRUE – Success, FALSE – Fail.

31. Get Device Long Info

[Function]

Abtain controller's schedule setup information

[Format]

```
boolean GetDeviceLongInfo(
       long dwMachineNumber,
       long dwInfo,
       long* dwValue
       );
```

[Parameter]

dwMachineNumber : Controller address
dwInfo : (Schedule)device information
dwValue : (Schedule) long variable pointer

[Attention] Type that means dwValue is different according to the value of dwInfo(See below paragraph). It is changed yb pointer of long type value when actually calls.

```
'================================================================================
Public Const DB_HOLIDAY_SIZE As Long = 4              // DB_HOLIDAY size (Long number)
Public Const DB_HOLIDAY_MAX As Long = 256             // possible number of holidays
Type DB_HOLIDAY                                       //  one holiday information
    Valid As Long     '0-Invalid, 1-Valid             // validity
    Month As Long     '1-12                           // month
    Day   As Long     '1-31, 1-7                      // day
    Number As Long    '0-Day is Day, 1-5-Day is Weekday // reservation
End Type
Public DbHolidayArray(DB_HOLIDAY_MAX - 1) As DB_HOLIDAY    //every holiday information
variable
Public DbHolidayArrayByte(DB_HOLIDAY_MAX * DB_HOLIDAY_SIZE - 1) As Long

'================================================================================
Public Const DB_TZONE_SIZE As Long = 6                // DB_TZONE size(Long number)
Public Const DB_TZONE_MAX As Long = 2048              // possible access time zone
Type DB_TZONE                                         // one access time zone
    Valid As Long      '0-Invalid, 1-Valid            // validity
    DMask As Long      '1-255-Day Mask                // day mask
    SHour As Long      '0-23-Start Hour               // start hour
    SMin  As Long      '0-59-Start Minute             // start minute
    EHour As Long      '0-23-End Hour                 // end hour
    EMin  As Long      '0-59-End Minute               // end minute
End Type
Public DbTzoneArray(DB_TZONE_MAX - 1) As DB_TZONE     //all access time zone variable
Public DbTzoneArrayByte(DB_TZONE_MAX * DB_TZONE_SIZE - 1) As Long
```

※ Enterance time zone No. n is included from (n-1)*8 to n*8-1 that the arrangement is shown.

ex) it is included from DbTzoneArray(16) to DbTzoneArray(23) if enterance time zone is 3.

(Enterance time zone No. is actually from 1 to 256.)

```
'================================================================================
Public Const DB_DAYLIGHT_SIZE As Long = 6             // DB_DAYLIGHT size(Long number)
Public Const DB_DAYLIGHT_MAX As Long = 4              // possible number

Type DB_DAYLIGHT                                      // DayLight
    Valid As Long       '0-Invalid, 1-Valid           //validity
    Year  As Long       '0-99                         //year
    Month As Long       '1-12                         //month
    Day   As Long       '1-31                         //day
    Hour  As Long       '0-23                         //hour
    Min   As Long       '0-59                         //minute
End Type
Public DbDayLightArray(DB_DAYLIGHT_MAX - 1) As DB_DAYLIGHT //all DayLight information
variable
                                                      // if DbDayLightArray(0),
```

```
                                                    //    it    will    be    changed    to
DbDayLightArray(1)
                                                    // if DbDayLightArray(2)
                                                    //    it    will    be    changed    to
DbDayLightArray(3).
Public DbDayLightArrayByte(DB_DAYLIGHT_MAX * DB_DAYLIGHT_SIZE - 1) As Long

'================================================================================
Public Const DB_TMODE_SIZE As Long = 7              // DB_TMODE size(Long number)
Public Const DB_TMODE_MAX As Long = 10              // possible time zone operating mode
number

Type DB_TMODE                                       // one time zone operating mode number
    Valid  As Long       '0-Invalid, 1-Valid        // validity
    DMask  As Long       '1-255-Day Mask            // day mask
    SHour  As Long       '0-23-Start Hour           // start hour
    SMin   As Long       '0-59-Start Minute         // start minute
    EHour  As Long       '0-23-End Hour             // end hour
    EMin   As Long       '0-59-End Minute           // end minute
    Mode   As Long       '0-14-Door Mode            // door mode
                                                    // (Run-time only work in door mode.)
End Type
Public DbTmodeArray(DB_TMODE_MAX - 1) As DB_TMODE      // all time zone operating mode
information variable
Public DbTmodeArrayByte(DB_TMODE_MAX * DB_TMODE_SIZE - 1) As Long

'================================================================================
Public Const DB_BELLTIME_SIZE As Long = 5           // DB_BELLTIME size (Long number)
Public Const DB_BELLTIME_MAX As Long = 20           // possible bell control number

Type DB_BELLTIME                                    // one bell control information
    Valid  As Long       '0-Invalid, 1-Valid        // validity
    DMask  As Long       '1-255-Day Mask            // day mask
    SHour  As Long       '0-23-Start Hour           // start hour
    SMin   As Long       '0-59-Start Minute         // start minute
    Sec    As Long       '1-255-Interval            // Bell on time (sec.)
End Type
Public DbBellTimeArray(DB_BELLTIME_MAX - 1) As DB_BELLTIME //all bell informaiton
Public DbBellTimeArrayByte(DB_BELLTIME_MAX * DB_BELLTIME_SIZE - 1) As Long

'================================================================================
Public Const DB_AUTODOOR_SIZE As Long = 6           //   DB_AUTODOOR   material   size(Long
number)
Public Const DB_AUTODOOR_MAX As Long = 6            // possible autodoor control number

Type DB_AUTODOOR                                 // one automatic door control information
material
    Valid  As Long       '0-Invalid, 1-Valid        // validity
    DMask  As Long       '0-Invalid, 1-255-Day Mask // day mask
    SHour  As Long       '0-23-Start Hour           // start hour
    SMin   As Long       '0-59-Start Minute         // start minute
    EHour  As Long       '0-23-End Hour             // end hour
    EMin   As Long       '0-59-End Minute           // end minute
End Type
Public DbAutoDoorArray(DB_AUTODOOR_MAX - 1) As DB_AUTODOOR //All automatic door control
information variable
Public DbAutoDoorArrayByte(DB_AUTODOOR_MAX * DB_AUTODOOR_SIZE - 1) As Long

'================================================================================
Public Const DB_AUTOKEY_SIZE As Long = 7            // DB_AUTOKEY size (Long number)
Public Const DB_AUTOKEY_MAX As Long = 5             // possible autokey number

Type DB_AUTOKEY                                     // one automatic function key
    Valid  As Long       '0-Invalid, 1-Valid        // validity
    DMask  As Long       '0-Invalid, 1-255-Day Mask // day mask
    SHour  As Long       '0-23-Start Hour           // start hour
    SMin   As Long       '0-59-Start Minute         // start minute
    EHour  As Long       '0-23-End Hour             // end hour
    EMin   As Long       '0-59-End Minute           // end minute
```

```
    FKey   As Long       '0-39-Function Key          // function key
End Type
Public DbAutoKeyArray(DB_AUTOKEY_MAX - 1) As DB_AUTOKEY //all automatic function key
information variable
Public DbAutoKeyArrayByte(DB_AUTOKEY_MAX * DB_AUTOKEY_SIZE - 1) As Long

'=============================================================================
Public Const DB_FKEY_NONE As Long = 40 '0-39, none - 40 //ineffective function key's
invariable
Public Const DB_NOACTKEY_SIZE As Long = 1      // one Relay enable operation function key is
Long
Public Const DB_NOACTKEY_MAX As Long = 4       // up to 4 Relay  enable operation function
key is possible.

Public DbNoActKeyArray(DB_NOACTKEY_MAX - 1) As Long  // all Relay enable operation function
key information variable
Public DbNoActKeyArrayByte(DB_NOACTKEY_MAX * DB_NOACTKEY_SIZE - 1) As Long
```

Above date mask is same as belwo

bitnumber

7	6	5	4	3	2	1	0
If 1, Saturday	If 1, Friday	If 1, Thursday	If 1, Wednesday	If 1, Tuesday	If 1, Monday	If 1, Sunday	If 1, holiday

ex) Put OxFC that is set 1 from bit No. 2, 3, 4, 5, 6, 7 if appreas <from Monday to Friday>

GetDeviceLongInfo's parameter will be same as below

Name	Explanation	dwInfoValue	dwValue (variable name)
Holiday	holiday	1	DbHolidayArray
Tzone	Access time zone	2	DbTzoneArray
DayLight	Day Light	3	DbDayLightArray
Tmode	Time operation mode	4	DbTmodeArray
BellTime	bell control	5	DbBellTimeArray
AutoDoors	autodoor control	6	DbAutoDoorArray
AutoKey	Auto key	7	DbAutoKeyArray
NoActKey	Relayno operation function key	8	DbNoActKeyArray

[returnvalue]

TRUE - Success, FALSE - Fail.

32. SetDeviceLongInfo

[Function]

Change controller's setup schedule .

[format]

```
boolean SetDeviceLongInfo (
       long dwMachineNumber,
       long dwInfo,
       long* dwValue
       );
```

[Parameter]

dwMachineNumber : Controller address
dwInfo : (schedule) setup information number
dwValue : (schedule) setup information material structure pointer (refer to GetDeviceLongInfo)

[returnvalue]

TRUE - Success, FALSE - Fail.

33. GetDoorStatus

[Function]

Abtain controller's door status.

[format]

```
boolean GetDoorStatus(
       long dwMachineNumber,
       long* dwStatus,
       long* dwDelay
       );
```

[Parameter]

dwMachineNumber : Controller address
dwStatus : long variable pointer for storing the door status
dwDelay : long variable pointer that saves currently the door status's delay time (second)
(if -1, indiates the time for standby

dwStatus's possible value is same as below

Name	Value	Explanation
LOCKSTATUS_CLOSE	1	Close status – already confirmed Door open bell setup will be applied with this status No meaning of dwDelay If dwDelay = -1, Run-time operation mode shows [CLOSE]
LOCKSTATUS_OPEN	2	Open status – can be access if verification is succeed dwDelay is Locking time or -1. If dwDelay = -1, Run-time operation mode [OPEN]
LOCKSTATUS_HCLOSE	3	Conmpulsory close door – Run-time operation mode is [close]. It is not changed.
LOCKSTATUS_HOPEN	4	Compulsory open status- Run-time operation mode is [OPEN]. It is not changed.
LOCKSTATUS_INVASION	5	It is related to intrution, and It doesn't affect Run-time operation mode.
LOCKSTATUS_FIRE	6	It is related to fire alarm, and It doesn't affect Run-time operation mode.

Operation mode is Boot-time and Run-time .
Boot-time operation mode is related to Door Mode from manu of "13>. When electricity power is on from the controller, Run-time operation mode will be applied to Boot-time operation mode.
Run-time operation mode is changed by time zone. Currently if there is no operation mode applied to time zone, automatically boot –time operation mode is applied to Run-time operation mode Boot-time.
But LOCKSTATUS_HCLOSE will be fixed as [CLOSE] in regardless of Run-time operation mode.
And LOCKSTATUS_HOPEN will be fixed as [OPEN] in regardless of Run-time operation mode..

[returnvalue]

TRUE – success, FALSE – Fail.

34. SetDoorStatus

[Function]

Set door status.

[Format]

```
boolean SetDoorStatus(
       long dwMachineNumber,
       long dwStatus,
       long dwDelay
       );
```

[Parameter]

dwMachineNumber : Controller address
dwStatus : Set door status
dwDelay : delay time to setup(sec)
(if -1, wait infinately.)
Execute automatically LOCKSTATUS_CLOSE after finish delay time.
If delay time is 0, execute instantly LOCKSTATUS_CLOSE.
(if alarm is on and delay time is 0 , stop alarm.)

dwStatus's value is same as below

Name	Value	Explanation
LOCKCTRL_CLOSE	1	Execute LOCKSTATUS_CLOSE dw but Delay has not meaning.
LOCKCTRL_OPEN	2	During Locking time, becomes LOCKSTATUS_OPEN. (same as verified) There is not meaning of dwDelay. Either Run-time operation mode is [CLOSE] or if door status is LOCKSTATUS_HCLOSE LOCKSTATUS_HOPEN LOCKSTATUS_INVASION LOCKSTATUS_FIRE , are not working.
LOCKCTRL_HCLOSE	3	Once Run-time operation mode [CLOSE] is changed, execute LOCKSTATUS_HCLOSE.
LOCKCTRL_HOPEN	4	Change Run-time operating mode[OPEN], and execute LOCKSTATUS_HOPEN.
LOCKCTRL_INVASION	5	Execute LOCKSTATUS_INVASION (check intrution alarm)
LOCKCTRL_FIRE	6	Execute LOCKSTATUS_FIRE (check fire alarm)
LOCKCTRL_EXIT	7	Same as operation LOCKCTRL_OPEN (check EXIT signal.)
LOCKCTRL_BELL_BUZZ	8	Check buzzer alarm
LOCKCTRL_BELL_RELAY	9	Check Relay alarm

[returnvalue]

TRUE – Success, FALSE – Fail.

35. Get Device Time

[Function]

Get device time.

[Format]

```
boolean GetDeviceTime(
       long dwMachineNumber,
       long* dwYear,
       long* dwMonth,
       long* dwDay,
       long* dwHour,
       long* dwMinute,
       long* dwSecond,
       long* dwDayOfWeek
       );
```

[Parameter]

dwMachineNumber : Controller address
dwYear, dwMonth, dwDay, dwHour, dwMinute, dwSecond : present time
dwDayOfWeek : Present day

value	day
1	Sunday
2	Monday
3	Tuesday
4	Wednesday
5	Thursday
6	Friday
7	Saturday

[returnvalue]

TRUE – Success, FALSE – Fail.

36. SetDeviceTime

[Function]

Setting the controller time to be the same as the PC time.

[Format]

```
boolean SetDeviceTime(
       long dwMachineNumber
       );
```

[parameter]

dwMachineNumber : Controller address

[returnvalue]

TRUE – Success, FALSE – Failure.

37. PoweroffDevice

[Function]

Power off device

[Format]

```
boolean PowerOffDevice(
       long dwMachineNumber
       );
```

[Parameter]
dwMachineNumber : Controller Address

[returnvalue]
TRUE – Success, FALSE – Fail.

38. FirmwareUpgrade

[Function]
Firmeware upgrade. (includes bootloader.)
Once firmware is upgrade, controller must be restarted.

[Format]
```
boolean FirmwareUpgrade(
       long dwMachineNumber,
       BSTR lpszFirmwareImageFilePath
       );
```

[parameter]
dwMachineNumber : controller address
lpszFirmwareImageFilePath : firmware image file route.
[returnvalue]
TRUE – Success, FALSE – Fail.

39. ~~ReadAllEnrollData~~

[Function]
All users's registration information that is registered to the controller is saved to the internal memory of OCX reading on one time.
It is read whole users's registration information included of card inforation, registration No. fingerprint data and etc.
[Format]
```
boolean ReadAllEnrollData(
       long  dwMachineNumber,
       long* dwEnrollCount
       );
```

[Parameter]
dwMachineNumber : Controller address
dwEnrollCount : Long type value pointer to save registrated Users No. to controller

[returnvalue]
TRUE – Success, FALSE – Failure.

40. ~~GetEnrollDataFromIndex~~

[Function]
ReadAllEnrollData It can get users' registration information of designated index position that is read users' registration information that is called function.
Read users' whole registration information such as card data, registration No., fingerprint data and etc..
[Format]
```
boolean GetEnrollDataFromIndex(
       long  dwMachineNumber,
       long  dwUserInfoIndex,
```

```
      long* dwEnrollNumber,
      long* dwEnrollData
      );
```

[Parameter]

dwMachineNumber : Controller Address
dwUserInfoIndex : Index of users' information getting in the users' information that is saved on the memory of OCX
Index starts from 0.
dwEnrollNumber : Pointer of long type variable to be saved users' ID
dwEnrollData : It is changed pointer of variable of EnrollData Type, pointer of variable of long Type when actually calls.(Note 6.GetEnrollData)
[Attention]
The value of dwUserInfoIndex must be less than dwUserCount that is got calling ReadAllEnrollData function.

[returnvalue]

TRUE – SUCCESS, FALSE – FAILURE.

41. ~~SetEnrollDataToIndex~~

[Function]

Users' registration information that is read the DB of PC is saved at the index position that is designated to the internal memory of OCX.
40. It is a function against GetEnrollDataFromIndex function.
Read users' whole registration information such as card data, registration No., fingerprint data and etc..

[Format]

```
boolean SetEnrollDataToIndex(
      long  dwMachineNumber,
      long  dwUserInfoIndex,
      long  dwEnrollNumber,
      long* dwEnrollData
      );
```

[Parameter]

dwMachineNumber : Controller Address
dwUserInfoIndex : Index of users' information to get users' information that is saved on a memory of OCX
Index starts from 0.
dwEnrollNumber : Pointer of long type variable that is be saved users' ID
dwEnrollData : Pointer of variable of EnrollDataType, it is changed long type variable pointer when is actually called.
(Note 7.SetEnrollData, 40.GetEnrollDataFromIndex)
[Attention]
This function has to call when index(dwUserInfoIndex) increases 1 each.

[returnvalue]

TRUE – SUCCESS, FALSE – FAILURE.

42. ~~WriteAllEnrollData~~

[Function]

41. It makes to register to send designated amount to the controller saved users' registration information at the internal memory of OCX by SetEnrollDataToIndex function.
39.Reply ReadAllEnrollData function.

[Format]

```
boolean WriteAllEnrollData(
      long dwMachineNumber,
      long dwEnrollCount,
      long dwEnrollMode
      );
```

[Parameter]

dwMachineNumber : Controller Address
dwEnrollCount : Registration information No. to transfer to the controller
dwEnrollMode : Transfer Mode for users' registration data
0 – Register only controller(D SERIES /FPC-301SF)
1 – Register controller and Telecop main device(KTL801/KTT801NS) together
(7.SetEnrollData Noate function)
[Attention]
This function sends only registration data of dwEnrollCount from No.0 index at the registration data saved the internal memory of OCX.

[returnvalue]

TRUE – SUCCESS, FALSE – FAILURE.

43. Progressing

[Function]

It is the event to notify the rate of process for PC programme duing progressing thg large communication between OCX and controller.
It is sent the notice at the programme to calculate progress rate of the function at the OCX when runs ReadAllEnrollData function and 42.WriteAllEnrollData function.

[Format]

```
void Progressing(
	short ActionType,
	short Percent,
	long  EnrollNumber
	);
```

[Parameter]

ActionType : Mean reading or writing(which function runs)
0 – 39.ReadAllEnrollData function running
1 – 42. WriteAllEnrollData function running
Percent : Rate(0~100%)
EnrollNumber : User registration No. of present transmission(receiption)

[Attention]
This event must be definited event handling part to be different the value to call from OCX user programme.

44. SetUserName

[Function]

Set user name. .
UNICODE 22letters (English and Korean)

[Format]

```
boolean SetUserName(
	long  dwMachineNumber,
	long  dwEnrollNumber,
	BSTR* lpszName
	);
```

[parameter]

dwMachineNumber : controller address
dwEnrollNumber : User enroll number to enroll whose name.
lpszName : User name string

[Warning]
If dwEnrollNumber is not already enrolled number, this function will be failed
Name should be Unicode and available for English and Korean.

[returnvalue]

TRUE – Success, FALSE – Failure

45. GetUserName

[Function]

Get user name
This function get user name who enrolled the name using function 44.SetUserName
Name : UNICODE STRING .

[Format]

```
boolean GetUserName(
	long  dwMachineNumber,
	long  dwEnrollNumber,
	BSTR* lpszName
	);
```

[parameter]

dwMachineNumber : Controller address
dwEnrollNumber : User enroll number to enroll whose name.
lpszName : long variable pointer to save user ID

[Warning]
If dwEnrollNumber is not already enrolled number, this function will be failed

[returnvalue]

TRUE – Success, FALSE – Failure

46. ~~GetMachineIP~~

[Function]

Get IP of controller.
This function is to get IP of controller according to NBNS protocol.
Getting IP is Unicode character string.
It can use the IP of the controller both static IP and dynamic IP that is alloted designated DHCP.

[Format]

```
boolean GetMachineIP(
	BSTR  lpszProduct,
	long  dwMachineNumber,
	BSTR* lpszIPBuf
	);
```

[Parameter]

lpszProduct : Product name character string
dwMachineNumber : Controller address to get IP
lpszIPBuf : Character variable point to save IP address character string of controller

[Attention]
Product name character string parametre must be 'D SERIES ' in case of and D SERIES and 'D SERIES' in case of D SERIES.

[returnvalue]

TRUE – SUCCESS, FALSE – FAILURE.

47. GetDeviceNetworkStatus

[Function]

Controller network status
When popup dialogue is on, OCX communication is impossible. Under this condition, controller tries to communicate, PC comes to hanging for a while. It recognize such status and used to avoid PC hangings.

[Format]

```
long GetDeviceNetworkStatus(
	BSTR lpszIPAddress
```

```
);
```

[parameter]

lpszIPAddress : IP address of the Controller

[returnvalue]

long value showing network status

[explanation]

Return value	Explanation
0	Controller is not connected
1	Controller is connected but no respond to get device network status.
2	Controller is connected but not able to communicate. (Pop-up is on)
3	Controller is connected and able to communicate

48. GetPhotoLogData

[Function]

Get Picture event matched designated condition

[Format]

```
boolean GetPhotoLogData(
	long dwMachineNumber,
	long dwEnrollNumber,
	long dwYear,
	long dwMonth,
	long dwDay,
	long dwHour,
	long dwMinute,
	long dwSecond,
	long* dwSize,
	long* dwData,
	);
```

[Parameter]

```
dwMachineNumber     : controller address
dwEnrollNumber      : User ID
                      if -1, it means user ID appears not clear.
dwYear … dwSecond   : Event time
dwSize              : Variable pointer to preserve data size
dwData              : Variable pointer to preserve picture data
                      Max. 8Kbyte(8192)
```

49. SetUserPhoto

[Function]

Set user’s picture.

[Format]

```
boolean SetUserPhoto(
	long dwMachineNumber,
	long dwEnrollNumber,
	long dwSize,
	long* dwData,
	);
```

[Parameter]

```
dwMachineNumber     : Controller Address
dwEnrollNumber      : User’s ID to set a picture
dwSize              : Picture data size
dwData              : Picture data
```

[Attention]
It can only display 160*120 jped format at Y SERIES
Picture data size must not over 8Kbyte(Max).

50. GetUserPhoto

[Function]
Get user's picture
[Format]

```
boolean GetUserPhoto(
	long dwMachineNumber,
	long dwEnrollNumber,
	long* dwSize,
	long* dwData,
	);
```

[Parameter]
dwMachineNumber : Controller Address
dwEnrollNumber : User ID to set picture
dwSize : Variable pointer to preserve picture data size
dwData : Vatiable pointer to preserve picture data
[Attention]
The registered video at Y SERIES device is 160*120, jpeg format and the size is 8Kbyte(Max).

51. VoIP_Init

[Function]
Initialize VoIP feature.
[Format]

```
boolean VoIP_Init(
	LPCTSTR lpszRingWaveFile,
	BSTR* lpszErrorMessage,
	long dwReserved
	);
```

[Parameter]
lpszRingWaveFile : Absolute path of calling sound file that is played calling from the controller.
The file format must be 44.1Khz, 16Bit, Stereo (Windows PCM format).
lpszErrorMessge : Pointer to save error message when fails initialization
dwReserverd : Reservation
[returnvalue]
TRUE – SUCCESS, FALSE – FAILURE.

[Attention]

52. VoIP_SendCommand

[Function]
Send VoIP Command.
[Format]

```
void VoIP_SendCommand(
	long dwCommand,
	long dwParam,
	long dwReserved
```

```
      );
```

[Parameter]

dwCommand : Command Code
dwParam : Parametre following to command
dwReserverd : Reservation

[returnvalue]

None.

[explanation]

dwCommand parametre had various meaning according to the present line state.

Line State	Command	Explanation
Calling from controller	0	Accept Calling
	1	Refuse Calling
On the call	0	Calling disconnection
	1	Open door(At this time, dwParam is pointed out opening time by second unit. dwParam becomes unlimited door opening in case of 0xFFFF.)

53. VoIP_GetStatus

[Function]

Get VoIP condition.

[Format]

```
long VoIP_GetStatus();
```

[Parameter]

None

[returnvalue]

The value of long type to refer to the present state.

[Explanation]

The meaning of return value is as below.

Return value	Explanation
0	No change state
1	calling disconnection
2	Calling from the controller
3	Connect call

54. VoIP_GetClientInfo

[Function]

Get the information of connected controller or the present tried calling.

[Format]

```
void VoIP_GetClientInfo(
       long *dwParams
       );
```

[Parameter]

dwParams : Arrangement variable pointer to save the information of the device

[returnvalue]

None

[Explanation]

The meaning of arrangement variable is as below.

Index	Explanation
0	Controller address(Machine ID)
1~4	IP of Controller, Each bite value of IP4v replys each arrangement element.

55. VoIP_GetImageExt

[Function]
Get camera image of the present connected controller. Return the original video size information of image sent to the device.

[Format]

```
boolean VoIP_GetImageExt(
	long *dwImage,
	long *dwWidth,
	long *dwHeight
	);
```

[Parameter]
dwImage : variable pointer to save camera image
dwWidth : Width pixel of camera image
dwHeight : Height pixel of camera image

[returnvalue]
[TRUE]-Success, [TRUE]-Failure

[Explanation]
Getting camera image is 160*120 or 320*240 size and each pixel is 3 bite(RGB channel) .
Namely, the size of variable is over 320*240*3 bite.

56. VoIP_DeInit

[Function]
Do deinit VoIP

[Format]

```
boolean VoIP_DeInit(
	long dwReserved
	);
```

[Parameter]
dwReserved : Reservation

[returnvalue]
[TRUE]-Success, [TRUE]-Failure

[Explanation]
This function must call before finishing programme.

57. OnVoipMessage

[Function]
It is the event to happen trying calling at the different controller on the call.

[Format]

```
void OnVoipMessage(
	long dwMessageType,
	long dwParam1,
	long dwParam2
	);
```

[Parameter]
dwMessageType : Event Type
dwParam1 : Parametre 1
dwParam2 : Parametre 2

[returnvalue]
[TRUE]-Success, [TRUE]-Failure

[Explanation]
The meaning of event type and event parameter is as below.

Event Type	Event Meaning	Parametre1	Parametre2
1	Try to call from different controller during holding on	Controller address trying to call	

58. GetCompanyName

[Function]

Get company name character string and other indication data.
This function is to get the information to set up by 63.SetCompanyName.

[Format]

```
boolean GetCompanyName(
       long  dwMachineNumber,
       BSTR* lpszCompanyName,
       long  *dwTopOffset,
       long  *dwColor,
       );
```

[Parameter]

dwMachineNumber : Controller address
lpszCompanyName : Character string variable pointer to save the company name
dwTopOffset : Pointer that is to save Y shaft position(Height position) to apprear company character string at the screen (Range: 25 ~ 180 at the Y SERIES)
dwColor : Pointer to save the colour value(definited the value at the RGM model) information to mark company name character string

[returnvalue]

TRUE – SUCCESS, FALSE – FAILURE.

59. SetCompanyName

[Function]

Set company name character string and other disply informaiton.
Reflect directly if the length of company name character string is not 0.

[Format]

```
boolean SetCompanyName_Ext(
       long  dwMachineNumber,
       BSTR* lpszCompanyName,
       long  dwTopOffset,
       long dwLeftOffset,
       long  dwColor,
       BSTR* lpszFontFace,
       long dwFontHeight,
       long dwDecorationFlag,
       long dwSecond,
       long dwType
       );
```

[Parameter]

dwMachineNumber : Controller address
lpszCompanyName : Company name character string variable pointer
dwTopOffset : Y shaft position(Height position) to display company name character string at the screen
dwLeftOffset : X shaft deviaton pixel to display company name character string at the screen
dwColor : Colour value to mark company name chatacter string(deginated value at the RGM Model)
lpszFontFace : Character string of font name to mark company name (Note VB6.0 sample)
dwFontHeight : Text size that marks company name
dwDecorationFlag : Variable to designate various style for emphasis font, *Italic* font and etc. It has meaning by bit.

Bit No. 31 ..... 4 3 2 1 0

Reserved						**Bold**	*Italic*

dwSecond : Apprear company name to display for several seconds. Permanent sign if 0xFFFFFFFF.

dwType : Set display form. If 1, it flows from left to right of the screen and fixed shown if 0.

[Attention]
Company name is 32 characters by 32 unicode and cut if it is over.
it is processed the bottom limit or top limit value if dwTopOffset is designated over.
if the set up character string is not empty, it directly reflects and flows from right to left of screen slowly
it does not apprear if the empty chatacter string. It is not flowed if dwType is 0.
it is cut if the maximum valid height of character string of all section is over 34.

[returnvalue]
TRUE – SUCCESS, FALSE – FAILURE.

60. SetBackgroundImage

[Function]
Set background image of main screen of the device.

[Format]

```
boolean GetFunckeyName(
	long dwMachineNumber,
	BSTR lpszBitmapFile
	);
```

[Parameter]
dwMachineNumber : Controller address
dwKeyIndex : Function key No.
lpszKeyName : Pointer of character string to save function keys

[Attention]
This feature only supports at Y SERIES Full type device.
The format of bite map file that is designated lpszBitmapFile parametre must be 272*480 pixel and 24bit colour bite map.

[returnvalue]
TRUE – SUCCESS, FALSE – FAILURE

Appendix-A. XML Command

[Function]
Feature to send command to XML character string for incarnating various features not add the new function at the OCX.

[Format]

```
boolean GeneralOperationXML(
	BSTR* lpszXML
	);
```

[Parameter]
lpszXML : XML command character string

[XML command character string format]

```
<REQUEST>REQUEST_STRING</REQUEST>
<MSGTYPE>request</MSGTYPE>
<MachineID>MACHINE_ID</MachineID>
```

Tags following to other REQEUST type

[returnvalue]

TRUE - SUCCESS, FALSE - FAILURE.

A-1. SetDeviceTime

[Function]

Function to set time of device

[Format]

```
<REQUEST>SetDeviceTime</REQUEST>
<MSGTYPE>request</MSGTYPE>
<MachineID>MACHINE_ID</MachineID>
<Year>YEAR</Year>
<Month>MONTH</Month>
<Day>DAY</Day>
<Hour>HOUR</Hour>
<Minute>MINUTE</Minute>
<Second>SECOND</Second>
```

[returnvalue]

TRUE - SUCCESS, FALSE - FAILURE.

A-2. GetLockStatus

[Function]

Get the door sensor state of device.

[Format]

```
<REQUEST>GetLockStatus</REQUEST>
<MSGTYPE>request</MSGTYPE>
<MachineID>MACHINE_ID</MachineID>
```

[returnvalue]

TRUE - SUCCESS, FALSE - FAILURE.

[if success Return XML]

```
<DoorSensor>Door sensor condition(If open 1, If not 0)</DoorSensor>
< ControlStatus >Logial control state</ControlStatus>
<ControlTime>Condition continue time</ControlTime>
```

A-3. SetEventFiltering

[Function]

Set the event filter. Bring only the event of set date that ReadGneralLogData/ReadSuperLogData value is called setting event filter. If the event filter is set, the devcice sends filtered event that is in setting time of whole event but not filtered at the new event. Date parametre is valid if the value of <EnableFiltering> is only 1. Once the process to get event after setting filter is same as ReadGeneralLogData-> GetGeneralLogData(Note example programme)

[Format]

```
<REQUEST>SetEventFiltering</REQUEST>
<MSGTYPE>request</MSGTYPE>
<MachineID>MACHINE_ID</MachineID>
<EnableFiltering> if 1, filter application, if 0, filter clear</EnableFiltering>
<StartYear></StartYear>
<StartMonth></StartMonth>
<StartDay></StartDay>
<EndYear></EndYear>
<EndMonth></EndMonth>
<EndDay></EndDay>
```

[returnvalue]

TRUE - SUCCESS, FALSE - FAILURE.

A-4. BeginEventTransaction

[Function]

Start event transaction. After the application gets event from the device, the device and application could be happened the event transmission discordance by unexpected erroes adn etc. To prenvent this, it institutes event transaction because of the possibility of 'event loss' from the application.

Once event transaction starts, event transmission is settled by completion(=close-out) getting evincive command that is finished transaction after the application takes event from the device. Namely, application sends event transaction start command before sending event and then event transaction finish command after processing to take event(save database).

[Format]

```
<REQUEST>BeginEventTransaction</REQUEST>
<MSGTYPE>request</MSGTYPE>
<MachineID>MACHINE_ID</MachineID>
<LogType> if 1, General Log, if 2, Super Log</LogType>
```

[returnvalue]

TRUE – SUCCESS, FALSE – FAILURE.

A-5. FinishEventTransaction

[Function]

Finish event transaction. It must finish sending this command after calling BeingEventTransaction command and application processes bringing all event from the device.

[Format]

```
<REQUEST>FinishEventTransaction</REQUEST>
<MSGTYPE>request</MSGTYPE>
<MachineID>MACHINE_ID</MachineID>
<LogType> if 1, General Log, if 2, Super Log</LogType>
```

[returnvalue]

TRUE – SUCCESS, FALSE – FAILURE.

A-6. GetManagerPCInfo

[Function]

Get administration computer IP to be used VOIP

[Format]

```
<REQUEST>GetManagerPCInfo</REQUEST>
<MSGTYPE>request</MSGTYPE>
<MachineID>MACHINE_ID</MachineID>
```

[returnvalue]

TRUE – SUCCESS, FALSE – FAILURE.

[Return XML]

```
<IPAddress>IP Address of Manager PC<IPAddress>
```

A-7. SetManagerPCInfo

[Function]

Set administration computer IP to be used VOIP

[Format]

```
<REQUEST>SetManagerPCInfo</REQUEST>
<MSGTYPE>request</MSGTYPE>
<MachineID>MACHINE_ID</MachineID>
<IPAddress>IP Address of Manager PC</IPAddress>
```

[returnvalue]

TRUE – SUCCESS, FALSE – FAILURE.

A-8. SetBackgroundImageIndex

[Function]

Set background image index of main screen.

[Format]

```
<REQUEST>SetBackgroundImageIndex</REQUEST>
<MSGTYPE>request</MSGTYPE>
<MachineID>MACHINE_ID</MachineID>
<ImageIndex>Index number of background image</ImageIndex>
```

[Attention]

Set default index when the index is out of the setting range.

[returnvalue]

TRUE – SUCCESS, FALSE – FAILURE.

A-9. GetBackgroundImageIndex

[Function]

Get background image index of main screen.

[Format]

```
<REQUEST>GetBackgroundImageIndex</REQUEST>
<MSGTYPE>request</MSGTYPE>
<MachineID>MACHINE_ID</MachineID>
```

[returnvalue]

TRUE – SUCCESS, FALSE – FAILURE.

[Return XML]

```
<ImageIndex>Index number of background image</ImageIndex>
```

Appendix -B. Device category and unsupportable function list

B-1. Device Pattern

The device type can know to judge the value from 29.GetDeviceInfo function.
This value is to find the device pattern and not to use the value that is not supported according to the type.

[Attention] :
Mark 16 antilogarithm and do not mark 10 antilogarithm because it is just marked.
XXXX part can become the temporary value.

No.	Value	Device Type	Explanation
1	0	Z SERIES	No support
2	2500	D SERIES	
3	3300	X SERIES	
4	3400	Q-CMR SERIES	
5	3401	Q-CMR SERIESPlus	
6	0x3380XXXX	X SERIES Plus	The board that made NUC951 CPU
7	0x3500XXXX	Y SERIES Full	The board that made NUC951 CPU
8	0x3501XXXX	Y SERIES Half	The board that made NUC951 CPU
9	0x3520XXXX	Y SERIES Full	The board that made ATMEL AT91SAM9G35 CPU
10	0x3521XXXX	Y SERIES Half	The board that made ATMEL AT91SAM9G35 CPU

B-2. Device category and unsupportable function list

O : Support
X : Not Support

No	Value(or XMLCommand)	Z SERIES	D SERIES	X SERIES	X SERIES+	Q-CMR SERIES+	Y SERIES Half	Y SERIES Full	
1	SetUserName	X	X	O	O	O	O	O	
2	GetUserName	X	X	O	O	O	O	O	
3	GetDeviceNetworkStatus	X	X	O	O	O	O	O	
4	GetPhotoLogData	X	X	X	X	X	O	O	
5	SetUserPhoto	X	X	X	X	X	O	O	
6	GetUserPhoto	X	X	X	X	X	O	O	
7	Voip_Init	X	X	X	X	X	O	O	
8	VoIP_SendCommand	X	X	X	X	X	O	O	
9	VoIP_GetStatus	X	X	X	X	X	O	O	
10	VoIP_GetClientInfo	X	X	X	X	X	O	O	
11	VoIP_GetImageExt	X	X	X	X	X	O	O	
12	VoIP_DeInit	X	X	X	X	X	O	O	
13	GetManagerPCInfo	X	X	X	X	X	O	O	
14	SetManagerPCInfo	X	X	X	X	X	O	O	
14	SetBackgroundImageIndex	X	X	X	X	X	O	O	
16	GetBackgroundImageIndex	X	X	X	X	X	O	O	
17	SetBackgroundImage	X	X	X	X	X	X	O	