3DISC FireCR Dental SDK

Reference Manual

Version 1.1.2

Doc No.: TM-xxx-EN
Rev 1.11, June 15, 2018
Part No.: CR-xxx-xx-xxx-EN

3DISC and FireCR Dental are trademarks of ***3D Imaging & Simulations Corp***, South Korea, and its affiliates. All other trademarks are held by their respective owners and are used in an editorial fashion with no intention of infringement. The data in this publication are for illustration purposes only and do not necessarily represent standards or specifications, which must be met by ***3D Imaging & Simulations Corp.*** All information contained herein is intended for guidance purposes only, and characteristics of the products and services described in this publication can be changed at any time without notice. Products and services may not be available in your local area. Please contact your local sales representative for availability information. ***3D Imaging & Simulations Corp.*** strives to provide as accurate information as possible, but shall not be responsible for any typographical error.

Contact

3DISC
IMAGING

3D Imaging & Simulations Corp.
Bldg.1, 48, Yuseong-daero 1184 beon-gil,
Yuseong-gu, Daejeon, 305-345 Korea

Tel : 82-42-931-2100
Fax : 82-42-931-2299
Website : www.3DISCimaging.com
E-mail : info@3DISCimaging.com

3DISC Americas
22560 Glenn Dr, Suite 116
Sterling, VA 20164 USA
Tel : 1-703-430-6080
E-mail : info@3DISCimaging.com

EC REP

3DISC Europe
Gydevang, 39-41, 3450 Alleroed, Denmark
Tel : 45-88-276-650
E-mail : info@3DISCimaging.com

Medical Device Security

Users must take steps to secure their networks and protect their Medical Information Systems which includes a risk assessment strategy, network defense in depth strategy, business continuity planning, etc.

- ✓ User Authentication
  Only authorized users should log on to computers on which medical information systems are installed.

- ✓ Password Security
  In today's world, passwords can be compromised in literally seconds by using a wide variety of tools and techniques. To lower the possibility of a compromised password, it is vital to adhere to a set of protocols.
  - Choose a password between 7 ~ 10 characters using both alpha and numeric characters.
  - Do not share the password.
  - Do not base the password on a pet's name, a relative's name or any dictionary word.
  - Do not write down the password.
  - Do not leave the account logged on.

- ✓ User Access Control
  Configure the workstation to prompt for logon after coming out of stand-by mode.

- ✓ Internet Usage
  Accessing to the Internet exposes the computer to a plethora of vulnerabilities such as:
  - Viruses
  - Spyware
  - Trojans
  - Hostile Codes

  It is not recommended to install any unauthorized software on the computer. Peer-to-peer software can expose your entire hard drive to any individual running the same type of software.

- ✓ Antivirus Products
  Use of antivirus software can increase CPU and memory usage, which can cause a slight degradation in the performance of the system. However, functionality should not be affected.

- ✓ Physical Security

  It is recommended that the user employs some method of physical security when dealing with the system to ensure that only authorized personnel have access to the product.

  There are several vulnerabilities a malicious user could exploit locally. Some examples are:

  - Theft of equipment
  - Local password cracking
  - Installation of hardware key loggers

1 Overview

This manual describes the structure, operation, and functions of the **Fire CR Dental SDK** that are used for image acquisition from **Fire CR Dental**. The manual explains the SDK concepts, as well as specific data type definitions and detailed descriptions of the functions.

1.1 What's New

This Reference Manual documents **Fire CR Dental SDK** release.

New Features:

Simple

- Complex modes (USB, 1:1, 1:N, N:N) is simplified to (USB, LAN). User can connect scanner by manually or Fire ID at any configuration in LAN mode.
- Calibration is not required any more

Versatile

- No limit for the number of IP address
- Window x64 version is also provided

Stable

- Fixed all issues of Standard and Advanced SDK
- Abundant log and error codes

New functions:

- Connect: Connect scanner manually (LAN only)
- Disconnect: Disconnect scanner manually connected (LAN only)
- GetFirmwareVresion: Get firmware version
- GetErrorCode: Get last error code
- GetErrorMessage: Get last error message

Deprecated functions:

- LockScanner

- UnlockScanner
- FreeImageBuffer
- SetRFIDListening
- GetrRFIDListening
- DoRun
- SetResolution
- SetAutoContrast
- FreeUID
- GetScannerError
- GetImageList
- GetPatientID

1.2 Required Environment

Item	**Description**
Operating System	Microsoft Windows 7 32bit/64bit or higher Mac OS High Sierra 10.13.4 or higher
CPU	Intel Celeron, Pentium, Core and Zeon
Memory	1GB free space or higher
Hard Disk	100MB free space or higher
Sample Project	Microsoft Visual Studio 2015 sample for Windows XCode sampe for Mac OS

*** Win32 Memory Usage Caution**: SDK consumes up to 500MB memory during image processing after image transferring. Win32 process is known to have limitation for allocating memory up to 1.2GB. Application has to be designed not to use over 700MB memory during SDK image processing. If more than 700MB memory is already used by application, separated process is recommended for DLL loading.

1.3 Windows Installation

Extract **FireCR_Dental_SDK_Windows_1.1.X.zip** file to appropriate folder. Uncompressed folder has following contents

Folder	File Name	Description
bin	x64/3discipp.dll	x64 Image processing DLL
	x64/calibration.dat	Calibration file (same with x86)
	x64/CRSwing.dll	x64 Dynamic Link Library
	x86/3discipp.dll	x86 Image processing DLL
	x86/calibration.dat	Calibration file (same with x64)
	x86/CRSwing.dll	x86 Dynamic Link Library
document	FireCR Dental SDK Manual.pdf	This document
	Release note.txt	Release note
driver	FireCRDriver_overwrite_xxxx.exe	Fire CR Dental USB driver
	CDM v2.10.00 WHQL Certified.exe	Fire ID Driver
	FT_Prog_v2.8.2.0 Installer	Fire ID Driver
Firmware	DentalCR_SystemUpdater_xxxx.exe	Fire CR Dental firmware updater
include	SDK.h	Library header file
lib	x64/CRSwing.lib	x64 Lib file for DLL
	x86/CRSwing.lib	x86 Lib file for DLL
sample	Visual Studio 2015 sample	

1.4 Mac OS Installation

Extract **FireCR_Dental_SDK_MacOS_1.1.X.zip** file to appropriate folder.

Folder/File	File Name	Description
FireCRDentalOSXDemo	XCode sample	Execution file
FireCRDentalOSXSDK.Framework		SDK Framework
FirmwareUpdater	DentalCR_OSX_SystemUpdater	Firmware updater

2. Work Flow

This chapter describes the work flow of scanning on USB and LAN mode and also describes functions and events required for work flow.

2.1 USB mode

Scanner is connected to PC with USB. Fire ID is not required and ignored.

2.1.1 Scan

Step	Procedure	Calling Function	Events Posted
1	Initialize SDK	OpenScannerSDK()	
2	Connect scanner	OpenScanner()	TOnScannerStatusEvent::esnConnected
3	Set patient ID (optional)	SetPatientID()	
4	Place IP on the tray		TOnScannerNotifyEvent::esnChangeFormat TOnRFIDNotifyEvent::esnFromScannner Call GetIPSize() to get IP Size Call GetUID() to get IP UID.
5	Start scan		TOnScannerNotifyEvent::esnScanStarted TOnScannerStatus::estScannnig
6	Transferring image		TOnScannerStatus::estTransferring TOnScannerNotifyEvent::esnProgress Call GetProgress() to get progress position
7	Image transferred		TOnScannerStatusEvent::esnDisconnected TOnScannerNotifyEvent::esnPreprocessed Call GetImageBuffer(), GetImageWidth(), GetImageHeight(), GetIPSize(), GetUID(), GetResolution() to get image information
8	Disconnect scanner	CloseScanner()	TOnScannerStatusEvent::esnDisconnected
9	Finalize SDK	CloseScannerSDK()	

2.1.2 Download image from scanner

Step	Procedure	Calling Function	Events Posted
1	Initialize SDK	OpenScannerSDK()	
2	Connect scanner	OpenScanner()	TOnScannerStatusEvent::esnConnected
3	Get image list	RequestImageList()	
4	Request image	RequestImageFromList	
5	Transferring image		TOnRFIDNotifyEvent::ernFromImage TOnScannerStatus::estTransferring TOnScannerNotifyEvent::esnProgress Call GetProgress() to get progress position
6	Image transferred		TOnScannerNotifyEvent::esnPreprocessed TOnScannerStatusEvent::esnDisconnected TOnScannerNotifyEvent::esnPreprocessed Call GetImageBuffer(), GetImageWidth(), GetImageHeight(), GetIPSize(), GetUID(), GetResolution() to get image information
7	Disconnect scanner	CloseScanner()	TOnScannerStatusEvent::esnDisconnected
8	Finalize SDK	CloseScannerSDK()	

2.2 LAN mode

2.2.1 Scan with Fire ID

Scanner is connected to PC with LAN automatically. Scanner and PC have to be on same network. Fire ID is required.

Step	Procedure	Calling Function	Events Posted
1	Initialize SDK	OpenScannerSDK()	TOnRFIDStatusEvent::ersConnected
2	Start waiting	OpenScanner()	
3	Tag IP at Fire ID		TOnRFIDStatusEvent::ernFromFireID Call GetUID() to get IP UID
4	Place IP on the tray		TOnScannerStatusEvent::esnConnected TOnScannerNotifyEvent::esnChangeFormat TOnRFIDNotifyEvent::esnFromScannner Call GetIPSize() to get IP Size

			Call GetUID() to get IP UID
5	Set patient ID (optional)	SetPatientID()	
6	Start scan		TOnRFIDNotifyEvent::ernFromImage TOnScannerNotifyEvent::esnScanStarted TOnScannerStatus::estScannnig
7	Transferring image		TOnScannerStatus::estTransferring TOnScannerNotifyEvent::esnProgress Call GetProgress() to get progress position
8	Image transferred		TOnScannerStatusEvent::esnDisconnected TOnScannerNotifyEvent::esnPreprocessed Call GetImageBuffer(), GetImageWidth(), GetImageHeight(), GetIPSize(), GetUID(), GetResolution() to get image information
9	Stop waiting	CloseScanner()	
10	Finalize SDK	CloseScannerSDK()	

2.2.2 Scan without Fire ID

Scanner is connected to PC with LAN manually. Scanner and PC have to be on same network. Fire ID is not required

Step	**Procedure**	**Calling Function**	**Events Posted**
1	Initialize SDK	OpenScannerSDK()	
2	Start waiting	OpenScanner()	
3	Get scanner list	RequestScannerList()	
4	Connect scanner	Connect()	TOnScannerStatusEvent::esnConnected
5	Set patient ID (optional)	SetPatientID()	
6	Place IP on the tray		TOnScannerNotifyEvent::esnChangeFormat TOnRFIDNotifyEvent::esnFromScannner Call GetIPSize() to get IP Size Call GetUID() to get IP UID
7	Start scan		TOnRFIDNotifyEvent::ernFromImage TOnScannerNotifyEvent::esnScanStarted TOnScannerStatus::estScannnig
7	Transferring image		TOnScannerStatus::estTransferring TOnScannerNotifyEvent::esnProgress

			Call GetProgress() to get progress position
8	Image transferred		TOnScannerNotifyEvent::esnPreprocessed TOnScannerStatusEvent::esnDisconnected Call GetImageBuffer(), GetImageWidth(), GetImageHeight(), GetIPSize(), GetUID(), GetResolution() to get image information
9	Disconnect scanner	Disconnect()	TOnScannerStatusEvent::esnDisconnected
10	Stop waiting	CloseScanner()	
11	Finalize SDK	CloseScannerSDK()	

2.2.3 Download image from scanner

Step	Procedure	Calling Function	Events Posted
1	Initialize SDK	OpenScannerSDK()	TOnRFIDStatusEvent::ersConnected
2	Start waiting	OpenScanner()	
3	Get scanner list	RequestScannerList()	
4	Get image list	GetImageList()	
5	Request image	RequestImageFromList	
7	Transferring image		TOnRFIDNotifyEvent::ernFromImage TOnScannerStatus::estTransferring TOnScannerNotifyEvent::esnProgress Call GetProgress() to get progress position
8	Image transferred		TOnScannerStatusEvent::esnDisconnected TOnScannerNotifyEvent::esnPreprocessed Call GetImageBuffer(), GetImageWidth(), GetImageHeight(), GetIPSize(), GetUID(), GetResolution() to get image information
9	Stop waiting	CloseScanner()	
10	Finalize SDK	CloseScannerSDK()	

3. Concepts

This chapter describes the enumerators, structures and error codes used in SDK

3.1 Enumerators

The EScannerStatus enumeration defines the status of the scanner.

```
enum EScannerStatus
{
    estNone,
    estDisconnected,
    estWaiting,
    estSynchronizing,
    estConnected,
    estSleeping,
    estTransferring,
    estScanning,
    estErasing,
};
```

The EUnitStatus enumeration defines the internal status of the scanner.

```
enum EUnitStatus
{
    ssNone,
    ssReady,
    ssSleep,
    ssScanning,
    ssErasing,
    ssTransferring,
    ssDoorJam,
    ssFlushing,
    ssNoResponse
};
```

The EScannerNotify enumeration defines the notify event of the scanner.

```
enum EScannerNotify
```

```
{
    esnNone,
    esnScanStarted,
    esnProgress,
    esnPreprocessed,
    esnChangeFormat,
    esnScannerError,
    esnCalPassed,
    esnCalFailed,
    esnChangePID
};
```

The ERFIDStatus enumeration defines the status of the Fire ID

```
enum ERFIDStatus
{
    ersNone,
    ersDisconnected,
    ersConnected
};
```

The ERFIDNotify enumeration defines the notify event of the Fire ID

```
enum ERFIDNotify
{
    ernNone,
    ernFromFireID,
    ernFromImage
    ernFromScanner,
    ernFromOtherSDK
};
```

The EScanResolution enumeration defines the scan resolution of the scanner

```
enum EScanResolution
{
    esrSD,
    esrHD
};
```

The EIPSize enumeration defines the IP Size

```
enum EIPSize
{
    eisNone
    eisIP0,
    eisP1,
    eisP2,
    eisP3,
    eisP4C,
    eisIP4
};
```

The EConnectionType enumeration defines the connection type

```
enum EConnectionType
{
    ectNone,
    ect1to1,
    ectNtoN,
    ectUSB
};
```

The enumeration EMsgEventName (Mac OS Only)

```
enum EMsgEventName;
{
    emsScannerStatus,
    emsScannerNotify,
    emsRFIDStatus,
    emsRFIDNotify,
    emsScannerList,
    emsImageList
};
```

3.2 Structures

The union TIPAddress for storing the IP Address is defined as

```
typedef union
{
    unsigned long   quad;
    unsigned char   bytes[4];
    public:
    void fromString(const char* ip);
    string toString();
} TIPAddress;
```

The union TScannerItem for storing the scanner information is defined as

```
union TScannerItem
{
    int params[14];
    struct {
        TIPAddress scannerIP;
         TIPAddress hostIP;
         EUnitStatus unitStatus;
         char scannerName[20];
         char hostName[20];
         int   imageCount;
    };
    bool operator==(TScannerItem& rValue)
    {
        return (scannerIP.quad == rValue.scannerIP.quad);
    }
    TScannerItem& operator=(TScannerItem& rValue)
    {
        for(int i = 0;i < 14;i++)
            params[i] = rValue.params[i];
        return (*this);
    }
};
```

The union TImageItem for storing the image information is defined as

```
union TImageItem
{
    char filler[512];
    struct
    {
        char                    header[4];
        TIPAddress              owner;
        long                    index;
        unsigned short          width;
        unsigned short          height;
        EScanResolution         resolution;
        EIPSize                 ipSize;
        TUID                    uid;
        double                  timeSaved;
        double                  timeServed;
        char                    pid[DCM_LONG_STRING_SIZE];
        PIXEL_FORMAT*            thumbs;
    };
    bool operator==(TImageItem& rValue)
    {
         return (owner.quad == rValue.owner.quad && index == rValue.index);
    }
    TImageItem& operator=(TImageItem& rValue)
    {
         for (int i = 0; i < 512; i++)
             filler[i] = rValue.filler[i];
         return (*this);
    }
};
```

The structure TCallBackSet for storing the callback function is defined as (Windows Only)

```
typedef struct TCallBackSet
{
    TOnScannerStatusEvent    scannerStatusEvent;
    TOnScannerNotifyEvent    scannerNotifyEvent;
    TOnRFIDStatusEvent          rfidStatusEvent;
    TOnRFIDNotifyEvent          rfidNotifyEvent;
    TOnScannerListEvent         scannerListEvent;
    TOnImageListEvent            imageListEvent;
};
```

3.3 Error Code

Call GetErrorCode() to get error code and call GetErrorMessage() to get error code string.

Error Code	Error String
NERR_SUCCESS	Success
NERR_FAIL_TO_CREATE_WINDOW	Failed to create window
NERR_FAIL_TO_LOAD_CAL_FILE	Failed to load calibration data
NERR_FAIL_TO_START_WSA	Failed to start up window socket
NERR_FAIL_TO_START_UDP_LISTEN	Failed to start UDP listening
NERR_NIC_NOT_EXIST	Network adapter is not found
NERR_CONNECTED_ALREADY	Scanner is already connected
NERR_FAIL_TO_CONNECT	Failed to connect to the scanner
NERR_REQUEST_TIMEOUT	Request time out
NERR_NO_CONNNECTED_SCANNER	No connected scanner exists
NERR_NOT_READY_STATUS	Scanner is not ready status
NERR_FAIL_TO_SEND_PACKET	Failed to send packet to the scanner
NERR_INVALID_SCANNER_STATUS	Scanner status is not valid to perform requested operation
NERR_ CONNECTED_BY_FIREID	Scanner is connected by Fire ID
NERR_NOT_LAN_TYPE	Connection type is not LAN
NERR_PROC_PRE_REQ	Processing pre-request
NERR_SDK_OPENED_ALREADY	SDK is already opened
NERR_NEED_NULL_CHAR	NULL character is required at end of string
NERR_SDK_NOT_OPEN	SDK is not opened
NERR_OUT_OF_INDEX	Out of index
NERR_SCANNER_NOT_EXIST	Scanner does not exist
NERR_NO_SUPPORT_1to1	Does not support 1 to 1 connection
NERR_INVALID_ARGUMENT	Invalid argument
NERR_INVALID_SIZE	SetOption function parameter valSize is invalid

NERR_INVALID_OPTION	SetOption function parameter is invalid

4. Functions

This chapter describes the **Fire CR Dental SDK** functions that perform image acquisition operations. Many solutions and hints for use of these functions can be found in Demo Sample.

* Mac OS rules
- "BOOL" is "bool"
- "TRUE" is true
- "FALSE" is false

OpenScannerSDK

Initialize the SDK

Syntax

Window: BOOL OpenScannerSDK(HWND hwnd, TCallBackSet * callbackSet, char* configPath);
Mac: OS bool OpenScannerSDK(wchar_t* configPath);

Parameters

hwnd	Deprecated
callbackSet	Pointer to the callback
configPath	Deprecated

Description

OpenScannerSDK allocates and initializes the internal resources, and also register callback function. OpenScannerSDK has to be called at the start of the application or start point of the scan procedure.

Return Values

TRUE	Indicates no error
FALSE	Indicates an error

CloseScannerSDK

Free the SDK

Syntax

void CloseScannerSDK();

Description

CloseScannerSDK frees all internal resources. CloseScannerSDK has to be called at the end of the application or end point of the scan procedure

OpenScanner

Connect to scanner in USB mode and wait for connection in LAN mode

Syntax

BOOL OpenScanner();

Description

OpenScanner try to connect scanner using USB first, if failed open network and wait for connecton.

Return Values

TRUE	Indicates no error
FALSE	Indicates an error

OpenScanner2

Connect to scanner using connection type

Syntax

BOOL OpenScanner2(EConnectionType type);

Parameters

type	Use ectUSB or ectNtoN.

Description

OpenScanner connect scanner using user specified connection type.

Return Values

TRUE	Indicates no error

FALSE	Indicates an error

CloseScanner

Disconnect scanner in USB mode and stop waiting in LAN mode

Syntax

BOOL CloseScanner();

Description

Disconnect scanner

RequestScannerList

Get scanner list

Syntax

TScannerItem * RequestScannerList(int * count)

Parameters

count	Number of scanners

Description

RequestScannerList get scanner information located on the same network. Number of scanner can be get by count parameter.

Return Values

Not NULL	Pointer to the TScanerItem. If the member hostName or hostIP is blank, it means the scanner is connected by that host and can’t be used.
NULL	Indicates an error

RequestImageList

Get image list

Syntax

TImageItem * RequestImageList(int * count)

Parameters

count	Number of images

Description

RequestImageList get image information saved on scanner's internal memory. Number of images are always 100.

Return Values

Not NULL	Pointer to the TImageItem. Member timeSaved and timeServied is COleDateTime format. ex: COleDateTime dt(item.timeSaved);
NULL	Indicates an error

RequestImageFromList

Request image using Image List

Syntax

BOOL RequestImageFromList(TIPAddress ipAddr, int index)

Parameters

ipAddr	NULL for USB connection. Scanner's IP address for Lan Connection
index	index of the image (from 0 to 99)

Description

RequestImageFromList requests image transfer to scanner. ScannerNotify callback function is called with notify parameter esnPreprocessed when image transfer is finished.

Return Values

TRUE	Indicates no error
FALSE	Indicates an error

GetImageWidth

Get image width

Syntax

int GetImageWidth()

Description

Get last transferred image's width.

Return Values

Not 0	Indicates no error
0	Indicates an error

GetImageHeight

Get image height

Syntax

int GetImageHeight()

Description

Get last transferred image's height

Return Values

Not 0	Indicates no error
0	Indicates an error

GetImageResolution

Get current scan resolution

Syntax

EScanResolution GetImageResolution()

Description

Get current scan resolution

Return Values

erSD	Scanner is set as SD resolution
erHD	Scanner is set as HD resolution

GetIPSize

Get current IP’s size

Syntax

EIPSize GetIPSize()

Description

Get current IP’s size

Return Values

eisNone	IP is not on the scanner
eisIP0	IP is Size 0
eisIP1	IP is Size 1
eisIP2	IP is Size 2
eisIP3	IP is Size 3
eisIP4C	IP is Size 4C

GetImageBuffer

Get last scanned image data

Syntax

PIXEL_FORMAT * GetImageBuffer ()

Description

Get last scanned image data. Image data type is unsigned short.

Return Values

Not NULL	Indicates no error
NULL	Indicates an error

ShowCalibrationDialog

Shows calibration dialog box

Syntax

BOOL ShowCalibrationDialog(BOOL visible)

Description

ShowCalibrationDialog shows calibration dialog box.

Parameters

visible	Not used

Return Values

TRUE	Indicates no error
FALSE	Indicates an error

ShowScannerControlDialog

Shows scanner control dialog box

Syntax

void ShowScannerControlDialog(BOOL visible)

Description

ShowScannerControlDialog shows scanner control dialog box.

Parameters

visible	Not used

GetProgress

Get image transferring progress

Syntax

int GetProgress()

Description

GetProgress get progress of the image transfer. Progress range is 0 to 1000.

Return Values

0~1000	Current progress

GetScannerStatus

Get scanner's status

Syntax

EScannerStatus GetScannerStatus ()

Description

GetScannerStatus get connected scanner status.

Return Values

estNone	Scanner is not connected
estDisconnected	Scanner is disconnected from this computer
estWaiting	This computer is ready to connect to network scanner
estSynchronizing	Not used
estConnected	Scanner is connected (to this computer)
estSleeping	Scanner is sleeping
estTransferring	Scanner is transferring image
estScanning	Scanner is scanning
estErasing	Not used

GetRFIDReaderStatus

Get Fire ID's status

Syntax

ERFIDStatus GetRFIDReaderStatus ()

Description

GetRFIDReaderStatus get Fire ID's status.

Return Values

estConnected	Fire ID is connected to this computer
estDisconnected	Fire ID is not found on this computer

GetUID

Get image's unique identifier

Syntax

TUID GetUID(ERFIDNotify device)

Description

GetUID get image UID connected to the device. Image UID is 64bit integer value.

Parameters

device	ernFromFireID: Fire ID ernFromScanner: Scanner tray ernFromImage: Scanned image

Return Values

Not 0	Image's UID
0	Image is not on the device

GetScannerConnectionType

Get scanner connection type

Syntax

EConectionType GetScannerConnectionType()

Description

GetScannerConnectionType get connection type

Return Values

ectNone	Not connected to scanner
ectNtoN	Scanner is connected via network
ectUSB	Scanner is connected vis USB

ClearImages

Delete all images on the scanner's internal memory

Syntax

BOOL ClearImages()

Description

ClearImages delete all images of the connected scanner. Scanner reserves 100 last images

Parameters

device	ernFromFireID: Fire ID ernFromScanner: Scanner tray ernFromImage: Scanned image

Return Values

Not NULL	Patient ID in character string
NULL	Patient ID not exist

SetPatientID

Set scanner's patient ID

Syntax

BOOL SetPatientID(const char * pid, int size)

Description

SetPatientID set scanner's current patient id. Maximum number of patient id's length is 64

Parameters

pid	Patient id in character string
size	Not used

Return Values

TRUE	Indicates no error
FALSE	Indicates an error

SleepScanner

Put scanner to sleep or wake scanner up

Syntax

BOOL SleepScanner(BOOL dosleep)

Description

SleepScanner puts scanner to sleep or wake scanner up

Parameters

dosleep	TRUE: Put to sleep FALSE: Wake up

Return Values

TRUE	Indicates no error
FALSE	Indicates an error

Connect

Connect scanner (LAN)

Syntax

BOOL Connect(const char * ip)

Description

Connect connects to network scanner with specified IP address

Parameters

ip	IP address string

Return Values

TRUE	Indicates no error
FALSE	Indicates an error

Disconnect

Disconnect scanner (LAN)

Syntax

const char * Disconnect()

Description

Disconnect scanner

Return Values

Not NULL	IP address of the disconnected scanner
NULL	Indicates an error

GetErrorCode

Get last error code

Syntax

ErroCodes GetErrorCode()

Description

GetErroCode get last error code

Return Values

ErrorCodes

GetErrorMessage

Get last error code string

Syntax

const char * GetErrorMessage()

Description

GetErroMessage get last error code

Return Values

Not NULL	Message string
NULL	No Message

GetScannerPN

Get scanner’s UDI string

Syntax

int GetScannerPN(char * buffer, int buffer_size)

Description

GetScannerPN get connected scanner’s UDI string if scanner is manufactured after 2016, otherwise get serial string.

Parameters

buffer	Character string where UDI will be written
buffer_size	size of buffer

Return Values

0	Indicates no error
Not 0	Indicates error

GetOption

Set scanner option

Syntax

bool SetOption(int opt, int flag, void * optval, int valsize)

Description

SetOption set various options of the scanner.

Parameters

opt	option type
flag	option flag
optval	option value
valsize	sizeof optval

Return Values

TRUE	Indicates no error
FALSE	Indicates an error

Option List

Type	Description	flag	optval
CSO_PAINT_CORNER	Turn on/off the empty corner area paint option	NA	true: enable false: disable

5. Event

Windows uses callback function to notify the SDK event. Callback functions are defined in TCallBackSet structure.t Mac OS uses notification center to notify the SDK event. messages are defined in EMsgEventName enumeration and message strings are stored in msgEventName[].

Event Type	**Windows TCallBackSet**	**Mac OS EMsgEventName**
Scanner Status	TOnScannerStatusEvent	emsScannerStatus
Scanner Notify	TOnScannerNotifyEvent	emsScannerNotify
RFID Status	TOnRFIDStatusEvent	emsRFIDStatus
RFID Notify	TOnRFIDNotifyEvent	emsRFIDNotify
Scanner List	TOnScannerListEvent	emsScannerList
Image List(Deprecated)	TOnImageListEvent	emsImageList

Mac OS uses “dentalCR-msg” for NSNotificationName. Sample code is as following

```
[[NSNotificationCenter defaultCenter] addObserver:self
                                      selector:@selector(didReceivedSDKNotification:)
                                      name:@”dentalCR-msg”
                                      object:nil];
```

Use msgEventName[] for the keyName of NSNotification. Sample code is as following

```
- (void)didReceiveSDKNotificaiton(NSNotification *)notification
{
    NSString * keyName = [NSString stringWithFormat:@”%s”, msgEventName[emsScannerStatus]];
    if([notification.unserInfo objectForKey:keyName] != nil)
    {
        EScannerStatus iMsg;
        iMsg = (EScannerStatus)[notification.userInfo objectForKey:keyName] integerValue];
        switch(iMsg){
        case estConnected:
            break;
        case estDisconnected:
            break;;
        …
```

Scanner Status

Posted when scanner status is changed

Description

Scanner Status event is posted when scanner status is changed.

Parameters

estNone	Initial status
estDisconnected	Scanner is disconnected from this PC
estWaiting	This PC is waiting for scanner connection (network)
estSyncronizing	Deprecated
estConnected	Scanner is connected to this PC
estSleeping	Scanner is sleeping
estTransferring	Scanner is transferring data
estScanning	Scanner is scanning
estErasing	Deprecated

Scanner Notify

Posted when scanner status is changed

Description

Scanner Notify event is posted when scanner notification is generated.

Parameters

esnProgress	Transferring is progressing. Use GetProgress() to get progress position
esnPreprocessed	Image is transferred to this PC
esnChangeFormat	Scan resolution or IP size is changed. Use GetResolution() and GetIPSize() function to get resolution and IP size.
esnScanStarted	Scan started
esnScannerError	Deprecated
esnCalPassed	Deprecated
esnCalFailed	Deprecated

esnImageList	Requested image list is ready (network only)
esnChangePID	Scanner patient ID is changed. Use GetPatientID() to get current patient ID

RFID Status

Posted when Fire ID is attached or detached

Description

RFID Status event is posted when Fire ID is attached or detached. ersConnected is also called If Fire ID is connected and OpenScannerSDK is called.

Enumerations

ersConnected	IP is tagged on Fire ID
ersDisconnected	Scan started

RFID Notify

Posted when IP is tagged on Fire ID, placed on scanner or scan is started

Description

RFID Notify event is posted when IP is tagged on Fire ID or placed on scanner or scan is started. Use GetUID() function to get image UID.

Enumerations

ernFromFireID	IP is tagged on Fire ID
ernFromImage	Scan started
ernFromScanner	IP is placed on scanner
ernFromOtherSDK	Deprecated

Scanner List

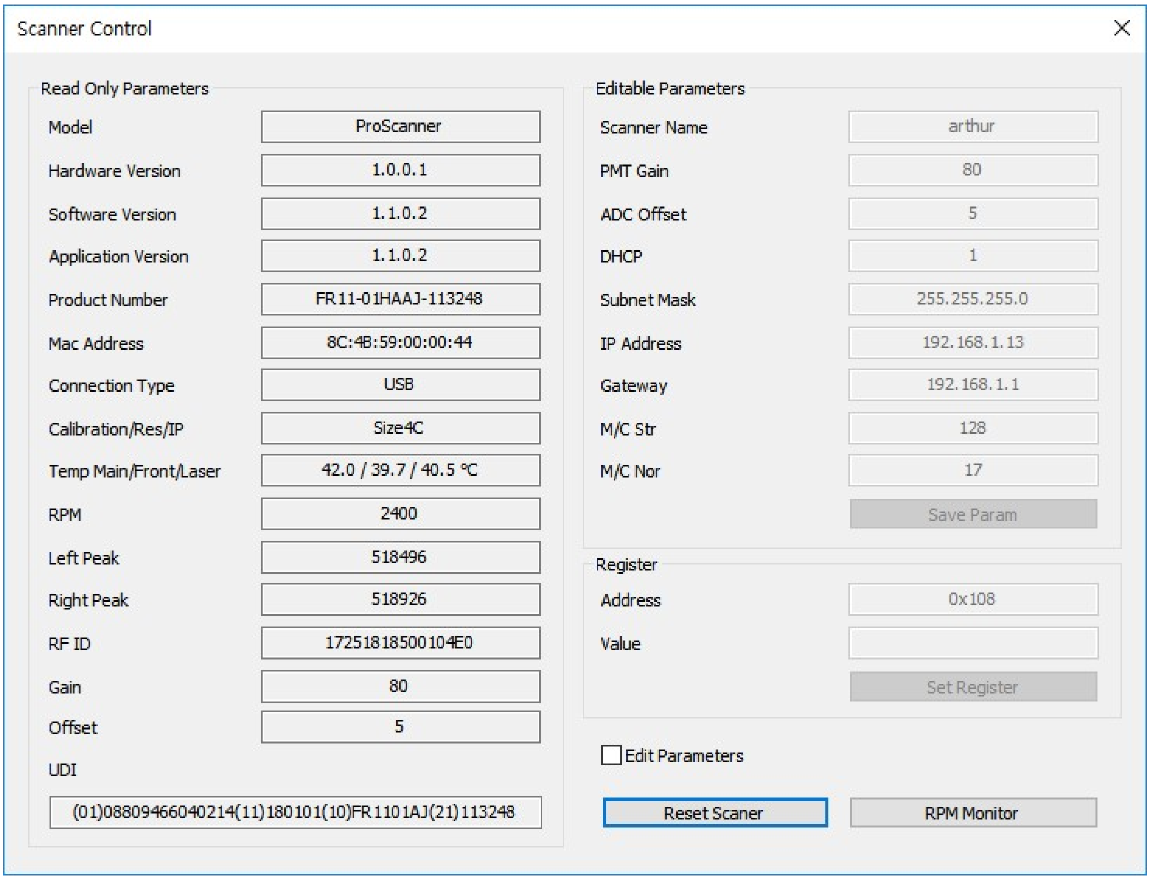
Posted when scanner list is changed

Description

Scanner List event is posted when new scanner is turned on or exist scanner turned off on the network.

6. Scanner Control

Scanner control can be used when PC is connected to scanner. Manual connection or automatic connection using Fire ID (and place on tray) is required in LAN mode.



Read Only Parameters

Model	ProScanner
Hardware Version	1.0.0.1
Software Version	1.1.0.2
Application Version	1.1.0.2
Product Number	FR11-01HAAJ-113248
Mac Address	8C:4B:59:00:00:44
Connection Type	USB
Calibration/Res/IP	Size4C
Temp Main/Front/Laser	42.0 / 39.7 / 40.5 ℃
RPM	2400
Left Peak	518496
Right Peak	518926
RF ID	17251818500104E0
Gain	80
Offset	5
UDI	(01)08809466040214(11)180101(10)FR1101AJ(21)113248

Editable Parameters

Scanner Name	arthur
PMT Gain	80
ADC Offset	5
DHCP	1
Subnet Mask	255.255.255.0
IP Address	192.168.1.13
Gateway	192.168.1.1
M/C Str	128
M/C Nor	17

Register

Address	0x108
Value	

Read Only Parameters

Displays read only scanner parameters.

Editable Parameters

Disaply editable scanner parameters. Check “Edit Paramters” to edit parameters. Click “Save Param” button to send parameters to scanner. Scanner rebooting is requried to apply the new parameters.

Register

Used to set special internal register values.

Reset Scanner

Click “Reset Scanner” button to reboot the scanner.

RPM Monitor

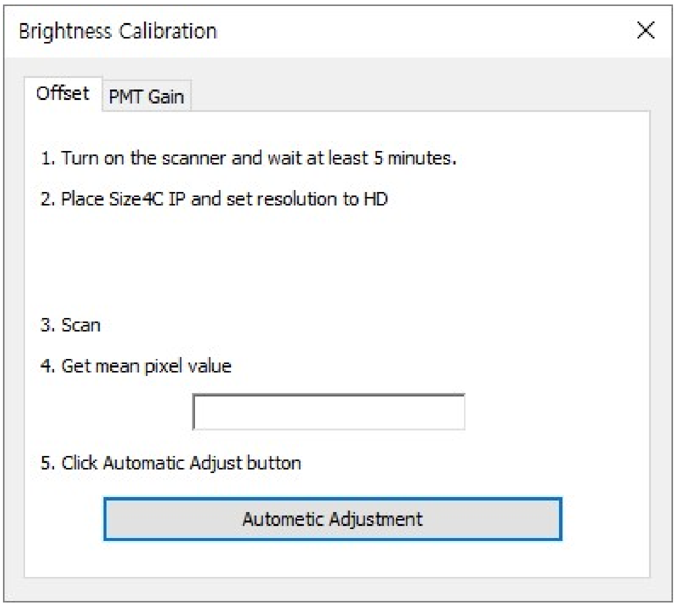
Click “RPM Monitor” button to monitor RPM variations.

7. Brightness Calibration

SDK version 1.0.0.x and 1.0.1.x requires device dependent calibration file which has to be created by calibration process or downloaded from scanner. But version 1.1.0.x uses device independent calibration file and do not need any calibration process. Name of calibration file is "calibration.dat" and distributed with SDK files. *Brightness calibration is required only if PMT or laser module is replaced. For other cases, it is not required.* Offset and PMT gain have to be adjusted if calibration is required.

7.1 Offset Calibration.

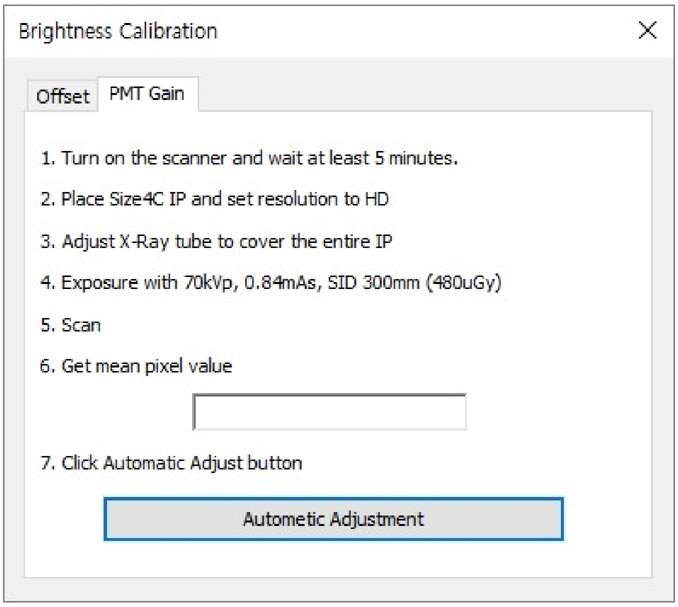
Follow the instructions.



1. Turn on the scanner and wait at least 5 minutes.
2. Place Size4C IP and set resolution to HD
3. Scan
4. Get mean pixel value
5. Click Automatic Adjust button

2. PMT Gain Calibration

Follow the instructions.



1. Turn on the scanner and wait at least 5 minutes.
2. Place Size4C IP and set resolution to HD
3. Adjust X-Ray tube to cover the entire IP
4. Exposure with 70kVp, 0.84mAs, SID 300mm (480uGy)
5. Scan
6. Get mean pixel value
7. Click Automatic Adjust button