iMON Display API

Version	1.00.0929
Release date	September 29, 2010

Revision History

Version	Release date	Remark
1.00.0929	Sep. 29, 2010	First version of iMON Display API

SOUNDGRAPH, Inc.

1. About iMON Display API

SoundGraph have supplied the various remote control solutions and display devices to Retail market and our ODM/OEM partners. And there have been lots of requests from our users and 3rd party developers to open API for iMON display devices.

For this reason we release iMON Display API now and will try to release the update of this API later after considering the request from users and 3rd party developers.

iMON Display API will help 3rd party developer to make their application to control iMON Display easily. It uses Display Plug-in mode feature of iMON, newly added in iMON 7.91.0929 Beta version. In order to make this API work properly, iMON 7.91.0929 Beta version or the later version should be installed and because this API must communicate with iMON, iMON should be always running. Currently it supports our VFD products and LCD products.

2. Using iMON Display API

Overview

iMON Display API can be used by Windows application. It is made by a shared MFC dll for C/C++ programmer. In order to make this API work properly, iMON 7.91.0929 Beta version or the later version should be always running.

Typical implementation

1) Initialize API by calling *IMON_Display_Init* function with a window handle and a message identifier. API will send notification to this window handle with this message identifier. When the caller application calls this function, API requests Display Plug-in Mode to iMON.
2) If Display Plug-in Mode is initialized successfully (API has the control for VFD/LCD display), the caller application can display text and equalizer data on VFD/LCD module and turn on/off icons on LCD module with API function calls.
3) When the caller application doesn't use display anymore, it should uninitialize API by calling *IMON_Display_Uninit* function.

How to use this API

In the folder which is unzipped from downloaded API file, there're 3 sub-folders;

API

iMONDisplayAPI.h – API interface definition file.

iMONDisplayDefines.h – API using enums and structs are defined.

iMONDisplay.dll – API dynamic linked library file

iMONDisplay.lib – API static linked library file.

Document

iMON Display API.doc – This guide document.

iMONDisplayApiSample

DisplayTest.sln – Visual C++ 8.0 (Visual Studio 2005) solution file

DisplayTest*DisplayTest.vcproj* – Visual C++ 8.0 (Visual Studio 2005) project file

Other files and folder – Sample project source files

As shown in iMONDisplayApiSample project, application needs to include *iMONDisplayAPI.h* header file (this header file includes *iMONDisplayDefines.h*) and link iMON Display API library file. Application can include library statically or can load library dynamically. The files *iMONDisplay.dll* should be placed at the folder where application execute(.exe) file located.

3. References

1) Enumerations

These enumerations are defined in *iMONDisplayDefines.h* file.

DSPResult

These enumeration values represent the returned result for iMON Display API function calls. All iMON Display API function calls return one of this result values.

```
enum DSPResult
{
        DSP_SUCCEEDED = 0,
        DSP_E_FAIL,
        DSP_E_OUTOFMEMORY,
        DSP_E_INVALIDARG,
        DSP_E_NOT_INITED,
        DSP_E_POINTER,

        DSP_S_INITED = 0x1000,
        DSP_S_NOT_INITED,,
        DSP_S_IN_PLUGIN_MODE,
        DSP_S_NOT_IN_PLUGIN_MODE,
}
```

Each value has meaning as below

Value	Meaning
DSP_SUCCEEDED	Function Call Succeeded Without Error
DSP_E_FAIL	Unspecified Failure
DSP_E_OUTOFMEMORY	Failed to Allocate Necessary Memory
DSP_E_INVALIDARG	One or More Arguments Are Not Valid
DSP_E_NOT_INITED	API is Not Initialized
DSP_E_POINTER	Pointer is Not Valid
DSP_S_INITED	API is Initialized
DSP_S_NOT_INITED	API is Not Initialized
DSP_S_IN_PLUGIN_MODE	API Can Control iMON Display (Display Plug-in Mode)
DSP_S_NOT_IN_PLUGIN_MODE	API Can't Control iMON Display

DSPNInitResult

These enumeration values represent the result status for requesting Display Plug-in Mode to iMON. iMON Display API notifies one of this result values to the caller application after requesting Display Plug-in Mode to iMON.

```
enum DSPNInitResult
{
        DSPN_SUCCEEDED = 0,
        DSPN_ERR_IN_USING = 0x0100,
        DSPN_ERR_HW_DISCONNECTED,
        DSPN_ERR_NOT_SUPPORTED_HW,
        DSPN_ERR_PLUGIN_DISABLED,
        DSPN_ERR_IMON_NO_REPLY,
        DSPN_ERR_UNKNOWN = 0x0200,
}
```

Each value has meaning as below

Value	Meaning
DSPN_ SUCCEEDED	Display Plug-in Mode is Initialized Successfully
DSPN_ERR_IN_USED	Display Plug-in is Already Used by Other Application
DSPN_ERR_HW_DISCONNECTED	iMON HW is Not Connected
DSPN_ERR_NOT_SUPPORTED_HW	The Connected HW doesn't Support Display Plug-in
DSPN_ERR_PLUGIN_DISABLED	Display Plug-in Mode Option is Disabled
DSPN_ERR_IMON_NO_REPLY	The Latest iMON is Not Installed or iMON Not Running
DSPN_ERR_UNKNOWN	Unknown Failure

DSPType

These enumeration values represent display type. Currently iMON Display API supports VFD and LCD products.

```
enum DSPType
{
        DSPN_DSP_NONE = 0,
        DSPN_DSP_VFD = 0x01,
        DSPN_DSP_LCD = 0x02,
}
```

Value	Meaning
0	No Display HW
0x01 (DSPN_DSP_VFD)	VFD products
0x02 (DSPN_DSP_LCD)	LCD products
0x03	VFD products and LCD products

DSPNotifyCode

These enumeration values represent the notification codes. iMON Display API will send or post message to the caller application. The caller application should assign the message and the winodw handle to receivce message with IMON_Display_Init fucntion. These enumeration values are used with WPARAM parameter of the message.

```
enum DSPNotifyCode
{
        DSPNM_PLUGIN_SUCCEED = 0,
        DSPNM_PLUGIN_FAILED,
        DSPNM_IMON_RESTARTED,
        DSPNM_IMON_CLOSED,
        DSPNM_HW_CONNECTED,
        DSPNM_HW_DISCONNECTED,
        DSPNM_LCD_TEXT_SCROLL_DONE,
}
```

For each notification code, please refer the following explanation.

Notification code	Remark
DSPNM_PLUGIN_SUCCEED	When API succeeds to get the control for the display, API will post the caller-specified message with DSPNM_PLUGIN_SUCCEED as WPARAM parameter. LPARAM represents DSPType. This value can be 0x01 (VFD), 0x02 (LCD) or 0x03 (VFD+LCD).
DSPNM_PLUGIN_FAILED	When API fails to get the control for the display, API will post the caller-specified message with DSPNM_PLUGIN_FAILED as WPARAM parameter. LPARAM represents error code with DSPNResult.
DSPNM_IMON_RESTARTED	When iMON starts, API will post the caller-specified message with DSPNM_IMON_RESTARTED as WPARAM parameter. LPARAM represents DSPType. This value can be 0 (No Display HW), 0x01 (VFD), 0x02 (LCD) or 0x03 (VFD+LCD).
DSPNM_IMON_CLOSED	When iMON closed, API will post the the caller-specified message with DSPNM_IMON_CLOSED as WPARAM parameter. LPARAM is not used.

DSPNM_HW_CONNECTED	When iMON HW newly connected, API will post the caller-specified message with DSPNM_IMON_RESTARTED as WPARAM parameter. LPARAM represents DSPType. This value can be 0 (No Display HW), 0x01 (VFD), 0x02 (LCD) or 0x03 (VFD+LCD).
DSPNM_HW_DISCONNECTED	When iMON HW disconnected, API will post the caller-specified message with DSPNM_HW_DISCONNECTED as WPARAM parameter. LPARAM is DSPNResult value, DSPN_ERR_HW_DISCONNECTED.
DSPNM_LCD_TEXT_SCROLL_DONE	When iMON LCD finishes scrolling Text, API will post the caller-specified message with DSPNM_LCD_TEXT_SCROLL_DONE as WPARAM parameter. The caller application may need to know when text scroll is finished, for sending next text.

2) Structure

This struct is defined in the *iMONDisplayDefines.h* file.

DspEqData

This structure contains Equalizer data for 16 bands.

```
struct DspEqData
{
        int BandData[16];

} DSPEQDATA, *PDSPEQDATA;
```

The member has meaning as below;

Member	Meaning
BandData	It represents Equalizer data for 16 bands. Its range is from 0 to 100.

3) Interface functions

These interfaces are defined in the *iMONDisplayAPI.h* file.

IMON_Display_Init

This function should be called to use other functions in iMON Display API. When the caller application calls this function, API tries to request Display Plug-in Mode to iMON.

```
DSPResult
IMON_Display_Init
(
        HWND hwndNoti,
        UINT uMsgNotification
)
```

Parameter	Remark
HWND hwndNoti	API will send/post message to this handle.
UINT uMsgNotification	API will send/post message to hwndNoti with this message identifier.

This function will return one of DSPResult enumeration value. DSP_SUCCEEDED will be returned if succeeded. DSP_E_INVALIDARG or DSP_E_OUTOFMEMORY can be returned when error occurs.

IMON_Display_Uninit

This function should be called when the caller application need not use this API any more. If this function call is missed, iMON can't display other information.

```
DSPResult
IMON_Display_Uninit
(
        VOID
)
```

This function will return one of DSPResult enumeration value. DSP_SUCCEEDED will be returned if succeeded.

IMON_Display_IsInited

This function can be used when the caller application wants to know if API is initialized.

```
DSPResult
IMON_Display_IsInited
(
        VOID
)
```

This function will return one of DSPResult enumeration value. If API is initialized, this call will return DSP_S_INITED. Otherwise DSP_S_NOT_INITED will be returned.

IMON_Display_IsPluginModeEnabled

This function can be used when the caller application wants to know if API can control iMON display.

```
DSPResult
IMON_Display_IsPluginModeEnabled
(
        VOID
)
```

This function will return one of DSPResult enumeration value. If API can control iMON display, this call will return DSP_S_IN_PLUGIN_MODE. Otherwise DSP_S_NOT_IN_PLUGIN_MODE will be returned.

IMON_Display_SetVfdText

This function can be used when the caller application wants to display text data on VFD module.

```
DSPResult
IMON_Display_SetVfdText
(
        LPCTSTR lpsz1stLine,
        LPCTSTR lpsz2ndLine
)
```

Parameter	Remark
LPCTSTR lpsz1stLine	This string data will be displayed on the 1st line of VFD module. It doesn't support multi-byte character and if string data is longer than 16 characters, it displays 16 characters from the first.
LPCTSTR lpsz2ndLine	This string data will be displayed on the 2nd line of VFD module. It doesn't support multi-byte character and if string data is longer than 16 characters, it displays 16 characters from the first.

This function will return one of DSPResult enumeration value. DSP_SUCCEEDED will be returned if succeeded. DSP_E_POINTER, DSP_E_NOT_INITED or DSP_E_FAIL can be returned if failed.

IMON_Display_SetVfdEqData

This function can be used when the caller application wants to display equalizer data on VFD module.

```
DSPResult
IMON_Display_SetVfdEqData
(
        PDSPEQDATA pEqData
)
```

Parameter	Remark
PDSPEQDATA pEqData	Pointer of DSPEQDATA structure. The caller application should fill this structure with the equalizer data for 16 bands.

This function will return one of DSPResult enumeration value. DSP_SUCCEEDED will be returned if succeeded. DSP_E_POINTER, DSP_E_NOT_INITED or DSP_E_FAIL can be returned if failed.

IMON_Display_SetLcdText

This function can be used when the caller application wants to display text data on LCD module.

```
DSPResult
IMON_Display_SetLcdText
(
        LPCTSTR lpszText
)
```

Parameter	Remark
LPCTSTR lpszText	This string data will be displayed on the LCD module. It supports multi-byte character and if string data is longer than display area, it will start to scroll. When text scrolling is finished, API will notify it with DSPNotifyCode enumeration value, DSPNM_LCD_TEXT_SCROLL_DONE.

This function will return one of DSPResult enumeration value. DSP_SUCCEEDED will be returned if succeeded. DSP_E_POINTER, DSP_E_NOT_INITED or DSP_E_FAIL can be returned if failed.

IMON_Display_SetLcdEqData

This function can be used when the caller application wants to display equalizer data on LCD module.

```
DSPResult
IMON_Display_SetLcdEqData
(
        PDSPEQDATA pEqDataL,
        PDSPEQDATA pEqDataR
)
```

Parameter	Remark
PDSPEQDATA pEqDataL	Pointer of DSPEQDATA structure. This parameter represents equalizer data of left channel. The caller application should fill this structure with the equalizer data of left channel for 16 bands.
PDSPEQDATA pEqDataR	Pointer of DSPEQDATA structure. This parameter represents equalizer data of right channel. The caller application should fill this structure with the equalizer data of right channel for 16 bands.

This function will return one of DSPResult enumeration value. DSP_SUCCEEDED will be returned if succeeded. DSP_E_POINTER, DSP_E_NOT_INITED or DSP_E_FAIL can be returned if failed.

IMON_Display_SetLcdAllIcons

This function can be used when the caller application wants to turn on/off all icons on LCD module.

```
DSPResult
IMON_Display_SetLcdAllIcons
(
        BOOL bOn
)
```

Parameter	Remark
BOOL bOn	If this value is TRUE, iMON will turn on all icons. Otherwise, iMON will turn off all icons.

This function will return one of DSPResult enumeration value. DSP_SUCCEEDED will be returned if succeeded. DSP_E_NOT_INITED or DSP_E_FAIL can be returned if failed.

IMON_Display_SetLcdOrangeIcon

This function can be used when the caller application wants to turn on/off orange shaped disk icons on the upper left part of LCD module. Disk icons consist of 8 pieces of orange and orange peel.

```
DSPResult
IMON_Display_SetLcdOrangeIcon
(
        BYTE btIconData1,
        BYTE btIconData2
)
```

Parameter	Remark
BYTE btIconData1	Each bit represents one of icons shaped the piece of orange. MSB is used for the piece placed on top and the remaining bits are for the piece placed in CCW from top.
BYTE btIconData2	MSB represents the orange peel shaped icon. Other bits are not used.

This function will return one of DSPResult enumeration value. DSP_SUCCEEDED will be returned if succeeded. DSP_E_NOT_INITED or DSP_E_FAIL can be returned if failed.

IMON_Display_SetLcdMediaTypeIcon

This function can be used when the caller application wants to turn on/off media type icons on the upper part of LCD module.

```
DSPResult
IMON_Display_SetLcdMediaTypeIcon
(
        BYTE btIconData
)
```

Parameter	Remark
BYTE btIconData	Each bit represents one of media type icons. From MSB each bit represents MUSIC, MOVIE, PHOTO, CD/DVD, TV, WEBCASTING and NEWS/WEATHER icon.

This function will return one of DSPResult enumeration value. DSP_SUCCEEDED will be returned if succeeded. DSP_E_NOT_INITED or DSP_E_FAIL can be returned if failed.

IMON_Display_SetLcdSpeakerIcon

This function can be used when the caller application wants to turn on/off speaker icons on the upper right part of LCD module.

```
DSPResult
IMON_Display_SetLcdSpeakerIcon
(
        BYTE btIconData1,
        BYTE btIconData2
)
```

Parameter	Remark
BYTE btIconData1	Each bit represents one of speaker icons.\nFrom MSB each bit represents L, C, R, SL, LFE, SR, RL and SPDIF icon.
BYTE btIconData2	MSB represents RR icon. Other bits are not used.

This function will return one of DSPResult enumeration value. DSP_SUCCEEDED will be returned if succeeded. DSP_E_NOT_INITED or DSP_E_FAIL can be returned if failed.

IMON_Display_SetLcdVideoCodecIcon

This function can be used when the caller application wants to turn on/off codec icons for video file on the lower part of LCD module.

```
DSPResult
IMON_Display_SetLcdVideoCodecIcon
(
        BYTE btIconData
)
```

Parameter	Remark
BYTE btIconData	Each bit represents one of video codec icons. From MSB each bit represents MPG, DIVX, XVID, WMV, MPG, AC3, DTS and WMA icon.

This function will return one of DSPResult enumeration value. DSP_SUCCEEDED will be returned if succeeded. DSP_E_NOT_INITED or DSP_E_FAIL can be returned if failed.

IMON_Display_SetLcdAudioCodecIcon

This function can be used when the caller application wants to turn on/off codec icons for audio file on the lower part of LCD module.

```
DSPResult
IMON_Display_SetLcdAudioCodecIcon
(
        BYTE btIconData
)
```

Parameter	Remark
BYTE btIconData	Each bit represents one of audio codec icons. From MSB each bit represents MP3, OGG, WMA and WAV icon.

This function will return one of DSPResult enumeration value. DSP_SUCCEEDED will be returned if succeeded. DSP_E_NOT_INITED or DSP_E_FAIL can be returned if failed.

IMON_Display_SetLcdAspectRatioIcon

This function can be used when the caller application wants to turn on/off aspect ratio icons on the lower right part of LCD module.

```
DSPResult
IMON_Display_SetLcdAspectRatioIcon
(
        BYTE btIconData
)
```

Parameter	Remark
BYTE btIconData	Each bit represents one of aspect ratio icons. From MSB each bit represents SRC, FIT, TV, HDTV, SCR1 and SCR2 icon.

This function will return one of DSPResult enumeration value. DSP_SUCCEEDED will be returned if succeeded. DSP_E_NOT_INITED or DSP_E_FAIL can be returned if failed.

IMON_Display_SetLcdEtcIcon

This function can be used when the caller application wants to turn on/off icons on the lower left part of LCD module.

```
DSPResult
IMON_Display_SetLcdEtcIcon
(
        BYTE btIconData
)
```

Parameter	Remark
BYTE btIconData	Each bit represents icon. From MSB each bit represents REPEAT, SHUFFLE, ALARM, REC, VOL and TIME icon.

This function will return one of DSPResult enumeration value. DSP_SUCCEEDED will be returned if succeeded. DSP_E_NOT_INITED or DSP_E_FAIL can be returned if failed.

IMON_Display_SetLcdProgress

This function can be used when the caller application wants to display progress bar on the upper and lower left part of text area of LCD module.

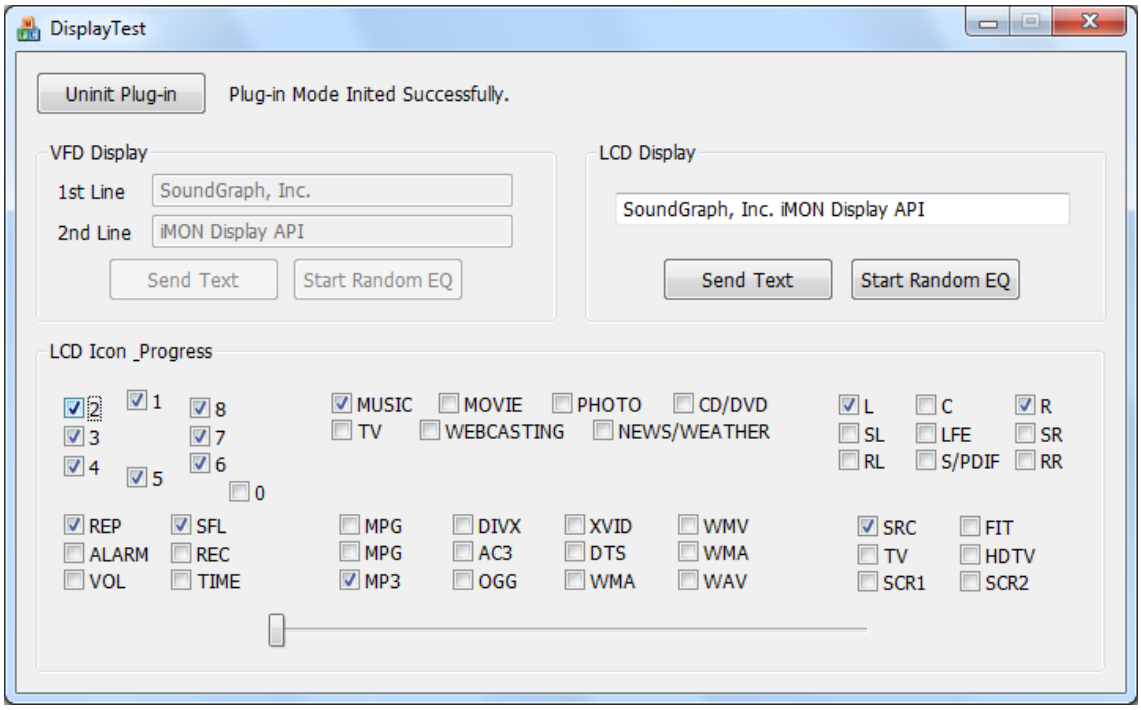
```
DSPResult
IMON_Display_SetLcdProgress
(
        int nCurPos,
        int nTotal
)
```

Parameter	Remark
int nCurPos	It represents the current position of progress bar.
int nTotal	It represents the total length of progress bar.

This function will return one of DSPResult enumeration value. DSP_SUCCEEDED will be returned if succeeded. DSP_E_NOT_INITED or DSP_E_FAIL can be returned if failed.

4. iMONDisplayApiSample

In the folder *iMONDisplayApiSample*, sample project exists. This sample is created in Visual C++ 8.0 (Visual Studio 2005). This sample shows how to initialize/uninitialize iMON Display API, how to display text and equalizer data on VFD/LCD module and how to turn on/off icons on LCD module.



This figure shows that Display Plug-in Mode is initialized successfully and this sample program has the control for LCD display. It sent text data and turned on some icons.