HikVision Mpeg4 linux Player SDK

Version 4.3(build200800804)

（2008.08.05）

Introduction

HikVision Mpeg4/H264 Linux Player SDK is developed based on SDL(Simple DirectMedia Layer) 1.2.11。SDL Window is created under Hik_PlayM4_Init(INITINFO InitInfo). The size of SDL Window is decided by the INITINFO structure. You can create more YUV surfaces on SDL Window, video is displayed on YUV surfaces, the position and size of the YUV surfaces are specified by the RECT structure. If there is only one video is playing, you can double click or drag to modify size of video.

If you have questions or suggests, please contact us.

Requirements:

Hardware:

CPU: Pentium 3 or higher

Memory：256MB

Software:

Suggest SDL 1.2.11, `support audio better`. **http://www.libsdl.org/**

Update Description:

VER 4.3 (build 20080804)

Add: **int Hik_PlayM4_OpenFileEx();**
int Hik_PlayM4_SetIndexFileEx();
int Hik_PlayM4_SetDecCallBackMend();
int Hik_PlayM4_SetPlayedTime();

VER 4.2 (build20060829)

Add: **int Hik_PlayM4_GetKeyFramePos();**
int Hik_PlayM4_GetNextKeyFramePos();
int Hik_PlayM4_BeginMixSound();
int Hik_PlayM4_OpenSound();
int Hik_PlayM4_CloseSound();
int Hik_PlayM4_StopMixSound();
int Hik_PlayM4_GetAbsFrameNum();
int Hik_PlayM4_SetJpegQuality();
int Hik_PlayM4_ConvertToJpegFile();
int Hik_PlayM4_PlaySoundShare(int nPort);
int Hik_PlayM4_StopSoundShare(int nPort);
int Hik_PlayM4_CheckDiscontinuousFrameNum(int nPort, unsigned int nEnable);
int Hik_PlayM4_ResetBuffer(int nPort,unsigned int nBufType);

update: Support playing mixing sound, and playing multi-encode files. The sdk is independent with SDL while using decode callback function.

VER 4.1 (build20051025)

- **Improve the function for the forward play. Now could be set at 2x、4x、6x、10x.**
- **Fixed the bug that the time stamp error in the decode call back**
- **Remove the link of the gdk and gdk in the sdk**
- **Support playing audio only**

VER 4.0:

Add a new function Hik_PlayM4_SetIndexFile() to set the index file. Then the playsdk could get the index from the file and use it directly. These modified will decrease the time that spend for the file be opened to play. Sometimes there will have so many times for the index created for the large file due to high disk I/O usage.

Function Descriptions:

1. int Hik_PlayM4_Init(INITINFO InitInfo, int using_sdl)

Description：Initialize SDL，must be done before all other functions.

Argument:

InitInfo; InitInfo.uWidth 和 InitInfo.uHeight specify size of SDL window.

Using_sdl: 0 – User must call Hik_PlayM4_SetDecCallBack(), to get decoded data, and play it.

1 – SDKwill play audio and video.

Return :

-1 — fail

0 — success

```
typedef struct tagInitInfo{
 int    uWidth;
 int    uHeight;
}INITINFO;
```

2. int Hik_PlayM4_DeInit(void)

Description：Free SDL.

3. int Hik_PlayM4_SetDisplay(int width, int height)

Description: Sets the size of SDL window.

Return :

-1 — fail

0 — success

4. int Hik_PlayM4_OpenFile(int nPort,char * sFileName)

Description: Open a file.

Argument:

SfileName ; Filename.

Return :

-1 — fail

0 — success

5. int Hik_PlayM4_CloseFile(int nPort)

Description: Close the file.

Return :

-1 — fail

0 — success

6. int Hik_PlayM4_Play(int nPort, RECT rect)

Description: Start play. If vidio has played, this function only changes current play speed to normal..

Argument:

rect; specifies the poison and the size of video on SDL window, the video size can not exceed SDL window.

Return :

-1 — fail

0 — success

```
typedef struct tagRect{
    int x;
    int y;
    int uWidth;
    int uHeight;
}RECT;
```

7. int Hik_PlayM4_Stop(int nPort)

Description: Stop play.

Return :

-1 — fail

0 — success

8. int Hik_PlayM4_Pause(int nPort,int nPause)

Description: Pause/resume play.

Argument:

nPause; 1 pause, 0 resume.

Return :

-1 — fail

0 — success

9. int Hik_PlayM4_Fast(int nPort)

Description: Fast play. Will speed play one time, up to 3 times. Calling Hik_PlayM4_Play() to resume to normal play speed at current.

Return :

-1 — fail

0 — success

10. int Hik_PlayM4_Slow(int nPort)

Description:: Slow play. Will slow play one time, up to 3 times. Calling Hik_PlayM4_Play() to resume to normal play speed at current.

Return :

-1 — fail

0 — success

11. int Hik_PlayM4_SetPlayPos(int nPort, float percent)

Description: Set the relative position of play(by percentage).

Argument:

percent; range 0~100%.

Return :

-1 — fail

0 — success

12. float Hik_PlayM4_GetPlayPos(int nPort)

Description: Get the relative position of play.

Return:

-1 — fail

> 0 — relative position of play, range 0~100%

13. int Hik_PlayM4_SetFileEndCallBack(int nPort, void (*FileEndCallBack)())

Description: Set callback function at the end of the file. In SDK，the decoder thread will automaticlly end when the file end, but user need to do check up：***Application should call Hik_PlayM4_Stop(nPort) in callback function.***

Argument:

FileEndCallBack;; Pointer of callback function.

14. int Hik_PlayM4_SetDecCallBack(int nPort, void (*DecCallBack)(int nPort, char * pBuf,int nSize,FRAME_INFO * pFrameInfo, long nReserved1,long nReserved2));

Description: Set callback function, replace audio and audio video of the player, will be controlled by the user. This function is called before Hik_PlayM4_Play, and will automatically ineffective when call Hik_PlayM4_Stop. It has to be reset before you call Hik_PlayM4_Play next time., Decod operate will not control speed, when the user returned from callback function, decoder will decode next part of data. To use this function, the user must have enough understand of video and audio, otherwise, please use it carefully.

Argument:

DecCBFun recall function pointer, it can not be “NULL” when using_sdl is 0;

Parameters of callback function:

nPort: port number of the player

pBuf: decoded audio and video data

nSize: the length of decoded audio and video data pBuf

pFrameInfo: picture and sound information. Details see below.

nReserved1, nReserved2: reserved

```
typedef struct{
    long width;     // width of the picture, unit is pixel , if for audio, it’s 0.
    long height;    //height of the picture. for audio, it’s 0
    long stamp;     // time stamp information, unit is Milli-Second (ms)
    long type;      // Data type,  T_AUDIO16, T_YV12。
    long frame_rate;    // Frame rate for encoding
}FRAME_INFO;
```

T_AUDIO16: Single channel. Sample yield is 16000 samples per second, the unit of sample is 16bits.

T——YV12： YV12 format

15. int Hik_PlayM4_PlaySound(int nPort)

Description: Open sound. Only one channel sound at one time. It will close exist sound if an other sound is opened. Caution : the sound is closed for default.

Return :

-1 — fail

0 — success

16. int Hik_PlayM4_StopSound()

Description: Close sound.

Return :

-1 — fail

0 — success

17. int Hik_PlayM4_SetVolume(int nPort, int volume)

Description: Set the audio volume. Can be set before play, and initialization is maximum.

Argument:

Volume, range 0~100.

Return :

-1 — fail

0 — success

18. int Hik_PlayM4_OneByOne(int nPort)

Description: Play video frame by frame. Calling Hik_PlayM4_Play（）will resume to normal play.

Return :

-1 — fail

0 — success

19. int Hik_PlayM4_GetFileTime(int nPort)

Description: Get total time of the file. Unit: second.

Return:

-1 — fail

>0 — total time of the file。

20. int Hik_PlayM4_GetPlayedTime(int nPort)

Description: Get current playing time. Unit: second.

Return:

-1 — fail

>0 — current playing time

21. int Hik_PlayM4_GetFileFrames(int nPort)

Description: Get total frames of the file.

Return:

-1 — fail

>0 — total frames of the file

22. int Hik_PlayM4_GetPlayedFrames(int nPort)

Description: Get the decoded video frames.

Return:

-1 — fail

>0 — decode video frames

23. int Hik_PlayM4_GetCurrentFrameNum(int nPort)

Description: Get decoding frame number. The number is relative.

Return:

-1 — fail

>0 — decodeing frame number。

24. int Hik_PlayM4_GetCurrentFrameRate(int nPort)

Description: Get the frame rate from current stream.

Return:

-1 — fail

>0 — frame rate from current stream

25. int Hik_PlayM4_GetPictureSize(int nPort, int *pwidth, int *pheight)

Desription：Get original image size in stream. Set video display area based on this size, no need to do decrease or increase with video card. It’s very useful for the video card that does not support hardware decrease or increase.

Argument：

pwidth: width of original image. It’s 352 in PAL CIF format.

pheight: height of original image. It’s 288 in PAL CIF format.

Return :

-1 — fail

0 — success

26. int Hik_PlayM4_OpenStream(int nPort, char *pfileheadbuf, int size, int streambufsize)

Description: Open stream.（Similar open file）;

Argument：

pfileheadbuf — file header from the card

size — length of file header

streambufsize —Set the buffer size for data. Range is SOURCE_BUF_MIN~ SOURCE_BUF_MAX.

Return :

-1 — fail

0 — success

27. int Hik_PlayM4_InputData(int nPort, char *pBuf, int nSize)

Description : Input the data stream getting from the card; input data after open stream.

Argument:

pBuf – buffer address;

nSize - buffer size

Return :

-1 — fail, data not input

0 — success, data input success

28. int Hik_PlayM4_CloseStream(int nPort)

Description: Close data stream.

Return :

-1 — fail

0 — success

29. int Hik_PlayM4_SetDisplayCallBack(int nPort, void(*DisplayCallBack)(int nPort, char *pbuf, long size, long width, long height, long stamp, long type, long reserved));

Description: set display call back function; Should return as soon as possible. If you want stop call back, you could set DisplayCallBack as NULL. Once set call back function, the function will effective until the program exit. It can be called at any time.

Argument:

DisplayCallBack — display call back function, can be set as NULL.

Parameter of callback description:

nPort 通道号; channel number

pbuf; return the image data

size; return the image data size

width; width of the image, unit: pixel

heigh; height of the image

stamp; time stamp, unit : milli-second (ms)

type; data type， T_YV12。

Received; reserved

30. int Hik_playM4_SetDisplayFrames(int nPort, int num);

Description：Set size of video play buffer. Only support stream mode, will be called after Open and before Play.

Argument:

num; buffer size，range 1~50。Less size and less delay.

Return：-1 — fail；

0 — success.

31. int Hik_playM4_ConvertToBmpFile(char *pbuf, long size, long width, long height, long type, Char *filename);

Description: Save image data as BMP file. Convert function will cost cpu resource. No need use this function if do not need save the image.

Argument:

buf, size, width, height, type — Same as DisplayCallBack's parameters

filename — filename that yu want save as, the best is .bmp file.

32. int Hik_PlayM4_GetFileHeadLength();

Description: Get filehead length of the file which displayed by current version player. Major used under stream player mode, it will get filehead data for Hik_PlayM4_OpenStream() input parameter.

Return:

the filehead length for this version player

33. unsigned int Hik_PlayM4_GetSdkVersion()

Description : Get current player's sdk version and build number. If only modify bug, we will only upgrade build number.

Return:

High 16 bit is current build number, 9-16bit is major version, 1-8 is minor version number. For example, if return 0x08200202, that means: build number is 0820, version is 2.2.

34. int Hik_PlayM4_GetLastErrorCode()

Description: Get error code for current error. Users should use it to get details error information when call some functions failed.

Return:

Error code. See macro definitions.

35. int Hik_PlayM4_ResetSourceBuffer(int nPort)

Description：Clear data in source buffer at stream mode.

36. int Hik_PlayM4_GetSourceBufferRemain(int nPort)

Description：Get remain bytes in source buffer at stream mode.

Return：

Remain bytes in source buffer.

37. int Hik_PlayM4_ThrowBFrameNum(int nPort, unsigned int num)

Description：Set frame number which do not decode B frame，to reduce CPU usage. Setting the value will no effect if no B frame in code stream. Suggest calling the function when fast play or mutiport play, and CPU usage is high.

Argument:

num: frame number which do not decode B frame. Range 0~2.

38. int Hik_PlayM4_SetPicQuality(int nPort, unsigned int hightquality)

Description：Set video quality. When set high quality, video effect will be better, but CPU usage will be higher. When play multiport, set low quality to lower CPU usage. When play single port zoom out, set high quality to get better video effect..

Argument:

hightquality： 1 — high quality；

0 — low quality(default).

39. int Hik_PlayM4_GetPicQuality(int nPort, unsigned int *hightquality)

Description：Get video quality.

return：

highquality: 1 — high quality；

0 — low quality.

40. int Hik_PlayM4_SetPlayRect()(int nPort, PLAYRECT rect)

Description：Set play video size.

Argument：

rect; secifies the poison and the size of video on SDL window, the video size can not exceed SDL window.

Return：-1 — fail；

0 — success.

41. int Hik_PlayM4_SetFileRefCallBack(int nPort, void (*FileRefCallBack)(int nPort))

Description: Cretae callback function for file index, setup before open the file. We create file index when open the file, for precision position in the file. Fuctions using index mest be called after creating index finished. And other function will be no effect.

Argument:

FileRefCallBack: callback function.

Return：-1 — fail；

0 — success.

42. int Hik_PlayM4_OneByOneBack(int nPort)

Description: One frame callback. Play one frame every calling. ***The function must be called after file index created.*** Hik_PlayM4_SetCurrentFrameNum()also playback one frame, but efficiency is lower.

Return：-1 — fail；

0 — success.

43.int Hik_PlayM4_SetCurrentFrameNum(int nPort, unsigned int framenum)

Description: Locate the player to position of specifing frame number。 ***The function must be called after file index created.***

Argument:

framenum: the frame number that the user want to locate.

Return：-1 — fail；

0 — success.

44.int Hik_PlayM4_OpenStreamEx(int nPort, char *pfileheadbuf, int size, int streambufsize);

Descripstion：Open stream， separate audio/video stream.

Argument：

I pfileheadbuf — file header from the card

size — length of file header

streambufsize —Set the buffer size for data. Range is SOURCE_BUF_MIN~ SOURCE_BUF_MAX.

Return :

-1 — fail

0 — success

45．int Hik_PlayM4_InputVideoData(int nPort, char *pBuf, int nSize);

Description : Input the video stream getting from the card; input data after open stream.

Argument:

pBuf – buffer address;

nSize - buffer size

Return :

-1 — fail, video sata input failed

0 — success, video data input success

46.int Hik_PlayM4_InputAudioData(int nPort, char *pBuf, int nSize);

Description：Input the audio stream getting from the card; input data after open stream..

Argument:

pBuf – buffer address;

nSize - buffer size

Return :

-1 — input audio sata failed

0 — input audio data success

47.int Hik_PlayM4_CloseStreamEx(int nPort);

Description：Close stream, call it with Hik_PlayM4_OpenStreamEx().

Return: -1 — failed

0 — success

48.int Hik_PlayM4_SetOpenStreamMode(int nPort, int mode);

Description: Set stream play mode, must set befor play.

Argument:

Mode - STREAME_REALTIME(default)，STREAME_FILE

Macro description:

STREAME_REALTIME: for play realtime data from net, decoder will decode datas immediately.

STREAME_FILE: for input data from file as stream mode,. Note: user should retry while Hik_PlayM4_InputData() return FALSE.

Return: -1 — failed

0 — success

49.int Hik_PlayM4_GetStreamMode(int nPort);

Description: Get stream play mode

Return: -1 — failed

STREAME_REALTIME — realtime stream

STREAME_FILE — file stream

V4.0 New add

50．int Hik_PlayM4_SetIndexFile(int nPort, char* sIndexName);

Description: Create the index file while record the data. Then provide it to the play sdk before open file. These modified will decrease the time that spend for the file be opened to play. Sometimes there will have so many times for the index created for the large file due to high disk I/O usage.

Argument: sIndexName the name for index

Note: Please refer to the dsdemo and hikplay demo to understand this call.

V4.2 New add

51. int Hik_PlayM4_GetKeyFramePos(int nPort,unsigned int nValue, unsigned int nType, PFRAME_POS pframePos);

Description: Search the last I frame before the specified position. Decoding must start from I frame. If the file that user saved don't start with I frame, data before the next first I frame would be ignored. So user who wants to grab a part of the file should start saving from an I frame. The ending position isn't so important, and at most three frames will be lost.

Argument:

nValue	the specified pos, time pos or frame number, the type is set by nType
nType	set the type of nValue. nValue means frame number while nType is BY_FRAMENUM. And nValue means time pos while nType is BY_FRAMETIME.
nFramePos	The position , frame number, time of I frame which has been searched

Structure instruction:

```
typedef struct{
    long nFilePos;      // file position
    long nFrameNum;     // frame number
    long nFrameTime;    // time（ms）;
}FRAME_POS,*PFRAME_POS;
```

52. int Hik_PlayM4_GetNextKeyFramePos(int nPort,unsigned int nValue, unsigned int nType, PFRAME_POS pframePos);

Description: Search the first I frame after the specified pos.

Argument:

nValue	the specified pos, time pos or frame number, the type is set by nType
nType	set the type of nValue. nValue means frame number while nType is BY_FRAMENUM. And nValue means time pos while nType is BY_FRAMETIME.
nFramePos	The position , frame number, time of I frame which has been searched

53. int Hik_PlayM4_BeginMixSound();

Description: Start mix sound mode, and before that the single mode should be closed by Hik_PlayM4_StopSound().

54. int Hik_PlayM4_OpenSound(int nPort);

Description: Open the sound of channel nPort after start mix sound mode. Pay attention to the difference between Hik_PlayM4_OpenSound and Hik_PlayM4_PlaySound.

55. int Hik_PlayM4_CloseSound(int nPort);

Description: Close the sound of channel nPort after start mix sound mode. Pay attention to the difference between Hik_PlayM4_CloseSound and Hik_PlayM4_StopSound.

56. int Hik_PlayM4_StopMixSound();

Description: Stop mix sound mode;

Note:

Mix sound mode needs the support of sound card and its drivers. If the player could run fluently without mix sound, and it stops or comes into panic after start mix sound mode, which means the sound driver doesn't work well. Recommend to update the sound driver or switch to alsa driver

Calling order:

```
Hik_PlayM4_Init();
Hik_PlayM4_BeginMixSound();
....
Hik_PlayM4_Play(port1, rect1);
Hik_PlayM4_Play(port2, rect2);

Hik_PlayM4_OpenSound(port1);
Hik_PlayM4_OpenSound(port2);

Hik_PlayM4_SetVolume(port1, 80);
Hik_PlayM4_SetVolume(port2, 60);

Hik_PlayM4_CloseSound(port1);
Hik_PlayM4_CloseSound(port2);
Hik_PlayM4_StopMixSound();
```

57. int Hik_PlayM4_GetAbsFrameNum();

Description: Get absolute frame number of the current frame while playing, which is generated while encoding.

Return:-1 — failed

>0 — decoding number

58. int Hik_PlayM4_SetJpegQuality(long nQuality);

Description: To set the quality of the jpeg file.

Parameters:

nQuality: the quality of jpeg file, the greater its value is ,the better quality of the jpeg file is . It ranges from 0 to 100(80 by default)

Notes: the recommended value is between 75 and 90 .

59. int Hik_PlayM4_ConvertToJpegFile(char *pBuf,long nSize,long nWidth,long nHeight,long nType, char *sFileName);

Description: Convert the yuv image to a jpeg file.

Parameters :

pBuf , nSize , nWidht , nHeight , nType	as same as the parameters of the callback function of capture picture.
sFileName	the file name of jpeg file to be saved

Notes: the usage of this API is similar to the API **Hik_PlayM4_ConvertToBmpFile**, the only difference is that it is possible to set the quality of the jpeg file by calling API **Hik_PlayM4_SetJpegQuality**.

60. int Hik_PlayM4_PlaySoundShare(int nPort);

Description: Plays a sound specified by the channel with share mode. It doesn't stop others. **Notice**: its effect is the same with mix sound functions, suggest use this function

Parameters:

nPort The channel of the player. It ranges from 0 to c HIK_PLAYM4_MAX_SUPPORTS-1.

61.int Hik_PlayM4_StopSoundShare(int nPort);

Description: Stop a sound specified by the channel. Notice : The player must use the same mode to play or stop the same sound. **Notice**: its effect is the same with mix sound stop functions, suggest use this function

Parameters:

nPort : The channel of the player. It ranges from 0 to HIK_PLAYM4_MAX_SUPPORTS-1.

62.int Hik_PlayM4_CheckDiscontinuousFrameNum(int nPort, unsigned int nEnable);

Description: Discard the discontinuous frame number data .

Parameters: 0 — Not Discard

1 — Discard

63.int Hik_PlayM4_ResetBuffer(int nPort,unsigned int nBufType);

Description: Clear the specified buffer.

Parameters:

nBufType: buffer type.See the macro.

Description of the macro:

BUF_VIDEO_SRC The souce video buffer,only used at stream mode.

BUF_AUDIO_SRC The souce audio buffer,only used at stream mode.

BUF_VIDEO_RENDER The video render buffer.

BUF_AUDIO_RENDER The audio render buffer.

Macro Definitions：

M4PErr_NoError	No error；
M4PErr_SDLInitFail	Initial SDL failed；
M4PErr_SDLSetVideoFail	Return by Hik_PlayM4_Init(), can not create SDL surface;
M4PErr_CreateYUVOverlayFail	Return by Hik_PlayM4_OpenFile() or Hik_PlayM4_OpenStream (),

	create YUV surface on SDL surface failed;
M4PErr_SDLAddTimerFail	Initial a timer failed;
M4PErr_CreateMutexFail	Create a mutex failed;
M4PErr_InvalidParam	Invalid parameter;
M4PErr_M4SDKUnInit	Must call Hik_PlayM4_Init() first;
M4PErr_OpenFileFail	Operate file failed;
M4PErr_OrderFail	Error calling order;
M4PErr_AllocMemFail	Allocate memory failed;
M4PErr_ThreadCreateFail	Create a thread failed;
M4PErr_BadFileFormat	File too short , or can not recognize stream;
M4PErr_InitVideoDecoderFail	Initial video decoder failed;
M4PErr_InitAudioDecoderFail	Initial audio decoder failed;
M4PErr_DecoderVideoFail	Decode Video failed;
M4PErr_DecoderAudioFail	Decode audio failed;
M4PErr_AudioCreateFail	Open the audio device failed;
M4PErr_SurportFileOnly	Only support the file play;
M4PErr_BadFileHead	No file header;
M4PErr_VersionIncorrect	Decoder version is different from encoder version;
M4PErr_BufOverflow	Buffer overflow, input stream failed。
M4PErr_JpegCompressFail	encode jpeg error
M4PErr_OpenSDLFailed	open libsdl.so failed

V4.3 New add

64. int Hik_PlayM4_OpenFileEx(int nPort,char * sFileName, unsigned int bpos, unsigned int epos);

Description: open file for playing the specified segment

Parameters:		
	int nPort	Player channel number;
	char * sFileName	File name;
	unsigned int bpos	Start position, assign 0xFFFFFFF for the beginning of file
	unsigned int epos	The end position, assign 0xFFFFFFF for the end of file.

65. int Hik_PlayM4_SetIndexFileEx(int nPort, char* sIndexName, unsigned int bpos, unsigned int epos);

Description: This function works with Hik_PlayM4_OpenFileEx(). If Hik_PlayM4_SetIndexFileEx () is called, Hik_PlayM4_OpenFileEx() needn't create file index and will use the index inputted by Hik_PlayM4_SetIndexFileEx(). The user have to confirm the segment specified by Hik_PlayM4_OpenFileEx() matches the segment specified by index file. Otherwise, it will cause error. If user can't ensure this, we don't suggest using this API.

Parameters:		
	int nPort	Player channel number;
	char * sIndexName	Index file name;
	unsigned int bpos	Start position, assign 0xFFFFFFF for the beginning of file
	unsigned int epos	The end position, assign 0xFFFFFFF for the end of file.

66.int Hik_PlayM4_SetDecCallBackMend(int nPort, void (*DecCallBack) (int nPort, char*pBuf, int size, FRAME_INFO* pFrameInfo, long nUser, long reserved2), long nUser);

Description: Register a callback function to replace the displayed part. It is controlled by user. It is called before

Hik_PlayM4_Play and will be invalid automatically when invoking Hik_PlayM4_Stop. Also it need to be reset before recalling Hik_PlayM4_Play. Pay attention that the decoder does not control decoding speed and if user returns from call back function, the decoder will decode the next data. To use this function, user must understand video display and vocality play very well. Please refer to DirectX development package about the relative knowledge.

Parameters:

	int nPort	Player channel number;
	void (*DecCallBack)	Callback function pointer that can't be NULL;
	long nUser	User data;

Description of the callback function:

nPort	The channel of player ;
pBuf	Pointer to the buffer that receives the data decoded by the player.
nSzie	Specifies the number of bytes to be get from the player ;
pFrameInfo	Points to a FRAME_INFO structure to receive the information of the image or sound ;
nUser	User data;
nReserved2	Reserved;

Description of the structure :

```
typedef struct
{
    long nWidth;        //width of image in pixels. If it is audio data the width is 0.
    long nHeight;       // height of image in pixels , if it is audio data, the height is 0 ;
    long nStamp;        //time stamp in milliseconds.
    long nType;         //Received data type. It is must be T_AUDIO16，T_RGB32 or T_YV12.
    long nFrameRate;    //Suggest the frame rate to display.
}FRAME_INFO;
```

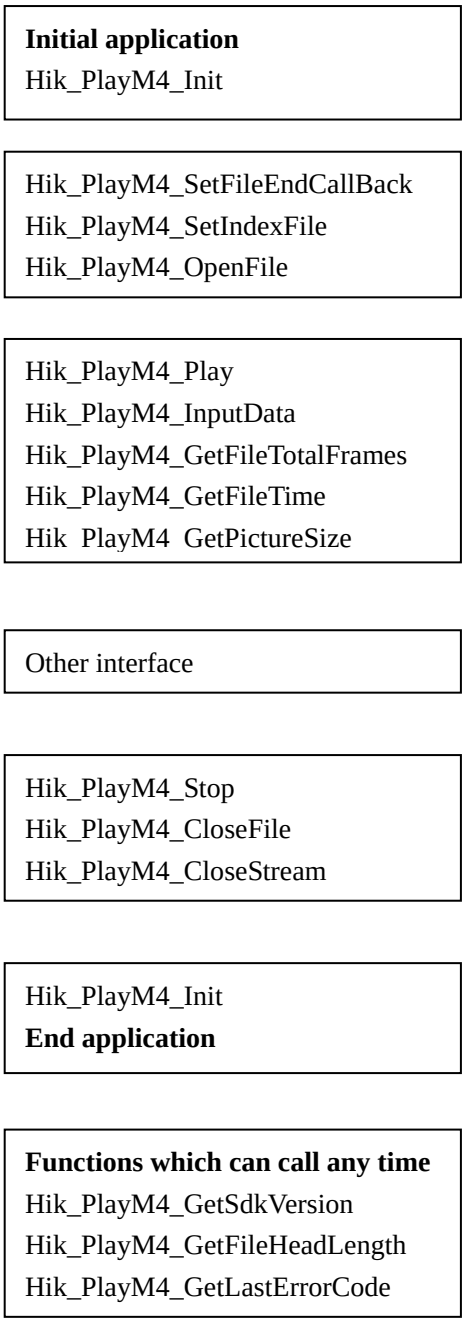
Please refer to Hik_PlayM4_SetDecCallBack() for more information.

67.int Hik_PlayM4_SetPlayedTime(int nPort, unsigned int nTime);

Description: Set the playing position basing the time information. This API will cost more time than Hik_PlayM4_SetPlayPos()

Parameters:

	int nPort	Player channel number;
	unsigned int nTime	The time to start playing, (unit: millisecond)



Technology support:
Tel: 86-571-88075998-8842
Mail: yutao@hikvision.com