VIVOTEK NETWORK DEVELPMENT PLATFORM

Packet Maker

Version 2.3.0.0

2017/01/20

TABLE OF CONTENTS

1. Overview

1.1 Introduction

This document describes the properties and methods supported by the VIVOTEK PacketMaker module.

1.2 Getting Started with PacketMaker Module

The main purpose of PacketMaker module is to generate a packet from encoded audio or video data.

1.3 File Structure

File	Description
doc\VNDP_PacketMaker_API.doc	This manual document
lib\d_PacketMaker.lib	The dynamic linking library
lib\PacketMaker.dll	The dynamic runtime library
inc\PacketMaker.h	Header file
inc\datapacketdef.h	Data packet definition file

2. Programmer's Guide

2.1 Using PacketMaker Module

You can generate a MJPEG packet from the output of AVSynchronizer decoder channel. This module can also trans-code audio packet from PCM to G729A. Generated packets can be sent to two-way servers, processed by AVSynchronizer or stored in database.

The audio capture channel is used to capture audio from sound cards of the computer and pack raw data stream directly into G729A encoded packet. This function wraps low-level audio capture mechanism, hence users don't need to write their own capture subroutine.

2.2 Application Sample Code

The sample code of PacketMaker is integrated in the DataBroker sample code, please refer to the distribution package for the TalkWithWAVFile sample code.

3. API Reference

This chapter describes the API functions for the PacketMaker

3.1 Enumeration

The enumeration used is depicted here.

- EMPMCHOPTION
- EPMUPDATEOPTION

3.1.1 EMPMCHOPTION

This enumeration lists the setting values which are used in PacketMaker_SetChannelOption.

```
typedef enum {
                                        eChOptAudioCodec        = 1,
                                        eChOptAudioSampleFreq   = 2,
                                        eChOptAudioChannelNum   = 3,
                                        eChOptAudioSizePerFrame = 4,
                                        eChOptAudioMute         = 5,
                                        eChOptAudioCodec        = 6,
                                        eChOptBitsPerSample     = 7,
                                        eChOptPacketCallback    = 8,
                                        eChOptStatusCallback    = 9,
} EMPMCHOPTION;
```

Values

eChOptAudioCodec

Update the new audio codec for channel. Currently only G729A and G711 are supported. The dwParam1 is the new codec.

eChOptAudioSampleFreq

Set the audio sample frequency. Currently only 8000 is supported. The dwParam1 is the new frequency.

eChOptAudioChannelNum

Set the audio channel number. The dwParam1 is new channel number.

eChOptAudioFramePerPacket

Set the number of frames per audio packet. The dwParam1 is the new number.

eChOptAudioSizePerFrame

Set the un-decoded audio data size for a frame. The module will use this value to partition the raw data and encode them to be a frame. The dwParam1 contains the size to be set.

eChOptAudioMute

Set the audio mute flag. If audio mute is set, the channel will generate packet contains data that have zero volume (PCM value 0). The dwParam1 contains Boolean value to indicate the mute state.

eChOptBitsPerSample

Set the bits per sample value. Currently only 16 is supported. The dwParam1 contains the new bits per sample.

eChOptPacketCallback

Set the callback function for packet callback. This is only meaningful for capture channel. The dwParam1 is the function pointer (of type **LPACAPPACKETCALLBACK**) function. Caller could use the context to locate object.

eChOptStatusCallback

Set the callback function for reporting capture status. This is only meaningful for capture channel. The dwParam1 is the function pointer (of type **LPACAPSTATUSCALLBACK**) and dwParam2 is the context that would be set as the first parameter for the callback function. Caller could use the context to locate object.

Remarks

These setting is able to be changed during channel operating, but it is recommend to setup the settings before operating starts.

Requirements

PacketMaker.h

3.1.2 EPMUPDATEOPTION

This enumeration lists the setting value that is to be set when calling **PacketMaker_UpdateChannelSettings**.

```
typedef enum {
                                        eUpChOptAudioVolume          = 1,
} EPMUPDATEOPTION;
```

Values

eUpChOptAudioVolume

Update the audio volume of encoder. The dwParam1 is the volume whose value is 0 ~100.

Remarks

Requirements

PacketMaker.h

3.2 Callback Function

The Callback function is depicted here.

- LPACAPSTATUSCALLBACK
- LPACAPPACKETCALLBACK

3.2.1 LPACAPSTATUSCALLBACK

This callback function is used to report the capture status for capture channel.

Syntax

```
typedef SCODE (__stdcall *LPACAPSTATUSCALLBACK)
                                        DWORD       dwContext,
                                        DWORD       dwStatusCode,
                                        DWORD       dwParam1,
                                        DWORD       dwParam2
);
```

Parameters

dwContext

[in] The context value of the status callback function.

dwStatusCode

[in] The status code.

dwParam1

[in] The first extra parameter for corresponding status code.

dwParam2

[in] The second extra parameter for corresponding status code.

Return Values

S_OK

Remarks

Requirements

PacketMaker.h

3.2.2 LPACAPPACKETCALLBACK

This callback function is used to notify the new coming of captured packet. The buffer is temporary, so when notified, the application should copy the packet if it needs to handle it later.

Syntax

```
typedef SCODE (__stdcall * LPACAPPACKETCALLBACK) (
                                                  DWORD                dwContext,,
                                                  TMediaDataPacketInfo *ptMediaPacket,
                                                  DWORD                dwDataTimePeriod,
);
```

Parameters

dwContext

[in] The context value of the status callback function

***ptMediaPacket**

[in] The packet that contains the encoded captured audio data.

dwDataTimePeriod

[in] The time length for the packet, it means the time period needs when playing the data back. This would help the next gate to control data rate.

Return Values

S_OK

Remarks

The capture mechanism in each sound card is not always the same, some cards would capture faster and the others would be slower. If the card captures faster than standard speed (for example it captures 16008 bytes when specified with 8000 samples per second and 2 bytes per sample), that's why the dwDataTimePeriod is needed here. It provides the next gate to control the data rate.

Requirements

PacketMaker.h

3.3 Data Structure

The data structure is depicted here.

- TPacketDateTimeInfo
- TPMCHANNELOPTION
- TPMACAPCHANNELOPTION

3.3.1 TPacketDateTimeInfo

This structure stores the date and time information.

```
typedef struct{
                                    WORD        wYear;
                                    WORD        wMonth;
                                    WORD        wMonthDay;
                                    WORD        wWeekDay;
                                    WORD        wHour;
                                    WORD        wMinute;
                                    WORD        wSecond;
                                    WORD        wMilliSecond;
} TPacketDateTimeInfo;
```

Members

wYear

Year in A.D. which is from 1970 to 2034.

wMonth

Month of year which is from 1 to 12.

wMonthDay

Day of month which is from 1 to 31.

wWeekDay

Date of week which is from 0(Sun) to 6(Sat).

wHour

Hour of day which is from 0 to 23.

wMinute

Minute of hour which is from 0 to 59.

wSecond

Second of minute which is from 0 to 59.

wMilliSecond

MilliSecond of second.

Remarks

Requirements

PacketMaker.h

3.3.2 TPMCHANNELOPTION

This structure collects the settings of the channel. When creating channel, fill this structure to setup the channel.

```
typedef struct{
                                    DWORD       dwFlag;
                                    DWORD       dwAudioCodec;
                                    DWORD       dwAudioSampleFreq;
                                    DWORD       dwAudioChannelNum;
                                    DWORD       dwAudioFramesPerPacket;
                                    DWORD       dwAudioSizePerFrame;
} TPMCHANNELOPTION;
```

Members

dwFlags

Indicate which field of this structure is valid. Currently it should be 0. It's reserved for future use.

dwAudioCodec

This is the audio codec for audio packetizer. If users do not need audio packetizing, please set this value to 0 and all the following fields are ignored.

dwAudioSampleFreq

This is the sampling frequency for the audio data. This value could not be 0 if dwAudioCodec is not 0. Currently only '8000' is valid.

dwAudioChannelNum

This is the channel number for the audio data. This value could not be 0 if dwAudioCodec is not 0. Currently only '1' is valid.

dwAudioFramesPerPacket

This is the frame number that will be assembled in an audio packet. This value could not be 0 if dwAudioCodec is not 0. The proper value range is from 5~20, larger value will have obvious delay. Smaller value will have less efficient for network usage.

dwAudioSizePerFrame

This the raw audio packet size for a frame. This value could not be 0 if dwAudioCodec is not 0. Currently only '160' is valid.

Remarks

Requirements

PacketMaker.h

3.3.3 TPMACAPCHANNELOPTION

This structure collects the settings of the audio capture channel. When creating capture channel, fill this structure to setup the channel.

```
typedef struct{
                                DWORD                   dwFlag;
                                DWORD                   dwAudioCodec;
                                DWORD                   dwAudioSampleFreq;
                                DWORD                   dwAudioChannelNum;
                                DWORD                   dwAudioFramesPerPacket;
                                DWORD                   dwAudioSizePerFrame;
                                DWORD                   dwBitsPerSample;
                                DWORD                   dwStatusContext;
                                DWORD                   dwPacketContext;
                                LPACAPSTATUSCALLBACK    pfStatusCB;
                                LPACAPPACKETCALLBACK    pfPacketCB;
} TPMACAPCHANNELOPTION;
```

Members

dwFlags

Indicate which field of this structure is valid. Currently it should be 0. It's reserved for future use.

dwAudioCodec

This is the audio codec for audio packetizer. This value could not be 0 if dwAudioCodec is not 0. Currently only 'mctG729A' is valid.

dwAudioSampleFreq

This is the sampling frequency for the audio data. This value could not be 0. Currently only '8000' is valid.

dwAudioChannelNum

This is the channel number for the audio data. This value could not be 0. Currently only '1' is valid.

dwAudioFramesPerPacket

This is the frame number that will be assembled in an audio packet. This value could not be 0. The proper value range is from 5~20, larger value will have obvious delay. Smaller value will have less efficient for network usage.

dwAudioSizePerFrame

This the raw audio packet size for a frame. This value could not be 0. Currently only '160' is valid.

dwBitsPerSample

This is one of the capture settings. It specifies the bits per sample. Currently only '16' is valid.

dwStatusContext

This is the context for status callback.

dwPacketContext

This is the context for packet callback.

pfStatusCB

This is the function pointer for the status callback.

pfPacketCB

This is the function pointer for the packet callback.

Remarks

Requirements

PacketMaker.h

3.4 API Definition

The API definition is depicted here.

3.4.1 PacketMaker_Initial

This function initializes the PacketMaker Module. You must call this function before using this module.

Syntax

```
SCODE PacketMaker_Initial(
                                    HANDLE          *phPacketMaker,
                                    DWORD           dwFlag,
                                    DWORD           dwVersion
);
```

Parameters

***phPacketMaker**

[out] The pointer to receive the handle of this PacketMaker object.

dwFlag

[in] Must be 0 now.

dwVersion

[in] The version of the PacketMaker. Assign the value PACKET_MAKER_VERSION to this parameter.

Return Values

S_OK

Initializes this module ok.

ERR_INVALID_VERSION

You are using an incompatible version.

PACKETMAKER_E_OUT_OF_MEMORY

There is not enough memory.

S_FAIL

Fail to initialize the module.

Remarks

Requirements

PacketMaker.h

See Also

PacketMaker_Release

3.4.2 PacketMaker_Release

Call this function to release the PacketMaker object.

Syntax

```
SCODE PacketMaker_Release (
                                        HANDLE              *phPacketMaker,
);
```

Parameters

***phPacketMaker**

[in] The address of the pointer to the PacketMaker object, returned by **PacketMaker_Initial**.

Return Values

S_OK

Release the object successfully.

ERR_INVALID_HANDLE

The handle can't be NULL.

Remarks

Requirements

PacketMaker.h

See Also

PacketMaker_Initial

3.4.3 PacketMaker_CreateChannel

Create a channel to pack video or audio data into a data packet. The packed packet is compatible to those received from IP camera or video server.

Syntax

```
SCODE PacketMaker_CreateChannel (
                                    HANDLE              hPacketMaker,
                                    HANDLE              *phChannel,
                                    TPMCHANNELOPTION    *ptChannelOptions
);
```

Parameters

hPacketMaker

[in] The handle of the PacketMaker, which created by PacketMaker_Initial.

***phChannel**

[out] The pointer to receive the handle of this channel.

***ptChannelOptions**

[in] A structure contains the settings of the channel. See TPMCHANNELOPTION.

Return Values

S_OK

Create the channel successfully.

ERR_INVALID_HANDLE

The handle can't be NULL.

ERR_INVALID_ARG

The settings are incorrect.

PACKETMAKER_E_OUT_OF_MEMORY

There is not enough memory.

PACKETMAKER_E_CHANNEL_EXCEEDED

You can't create more than 64 channels.

S_FAIL

Fail to create the channel due to the audio encoder initialized failed.

Remarks

Requirements

PacketMaker.h

See Also

PacketMaker_DeleteChannel, **TPMCHANNELOPTION**

3.4.4 PacketMaker_DeleteChannel

Delete the channel created by **PacketMaker_CreateChannel**.

Syntax

```
SCODE PacketMaker_DeleteChannel (
                                        HANDLE          *phChannel,
);
```

Parameters

***phChannel**

[in] The address of the handle of channel created by **PacketMaker_CreateChannel**.

Return Values

S_OK

Delete the channel successfully.

ERR_INVALID_HANDLE

The handle can't be NULL.

PACKETMAKER_E_CHANNEL_NOT_FOUND

The channel you specified does not exist.

Remarks

Requirements

PacketMaker.h

See Also

PacketMaker_CreateChannel

3.4.5 PacketMaker_GetMaxHeader

Call this function to get the upper bound the packet header will occupy. This value is possible to be changed if the packet format changes in the future, so please do not hardcode this value.

Syntax

```
SCODE PacketMaker_GetMaxHeader (
                                        HANDLE          hPacketMaker,
                                        DWORD           *pdwMaxHeader,
);
```

Parameters

hPacketMaker

[in] The handle of the PacketMaker, which created by **PacketMaker_Initial**.

***pdwMaxHeader**

[out] The maximum bytes a packet header will occupy.

Return Values

S_OK

The maximum value is retrieved successfully.

ERR_INVALID_ARG

The pdwMaxHeader could not be NULL.

ERR_INVALID_HANDLE

The handle can't be NULL.

Remarks

Requirements

PacketMaker.h

See Also

3.4.6 PacketMaker_SetChannelOption

Call this function to update the channel option after channel has been created.

Syntax

```
SCODE PacketMaker_SetChannelOption (
                                        HANDLE          hChannel,
                                        DWORD           dwOption,
                                        DWORD           dwParam1
                                        DWORD           dwParam2
);
```

Parameters

hChannel

[in] the handle of the channel created by PacketMaker_CreateChannel.

dwOption

[in] the channel option to be updated. The value must be one of the values in **EMPMCHOPTION**.

dwParam1

[in] the first parameter that each option needs. The corresponding value could be found in **EMPMCHOPTION**.

dwParam2

[in] the second parameter that each option needs. The corresponding value could be found in **EMPMCHOPTION**.

Return Values

S_OK

The option is set successfully.

ERR_INVALID_ARG

The option and parameters combination is wrong.

ERR_INVALID_HANDLE

The handle can't be NULL.

PACKETMAKER_E_OUT_OF_MEMORY

There is not enough memory.

S_FAIL

Fail to initialize audio encoder.

Remarks

Requirements

PacketMaker.h

See Also

3.4.7 PacketMaker_MakeVideoPacket

Call this function to generate a video packet from the encoded video data.

Syntax

```
SCODE PacketMaker_MakeVideoPacket (
                                        HANDLE                  hChannel,
                                        BYTE                    *pbyBuffer,
                                        DWORD                   dwBuffSize,
                                        DWORD                   dwDataOffset,
                                        TMediaDataPacketInfo    *ptMediaPacket
);
```

Parameters

hChannel

[in] The handle of the channel, which created by PacketMaker_CreateChannel.

***pbyBuffer**

[in] The buffer that contains the encoded video data and the extra space reserved for the packet header. Note: the video data must start from an offset of this buffer. The value of the offset is specified in dwDataOffset.

dwBuffSize

[in] The total size of the buffer specified in pbyBuffer, this includes the reserved header space (dwDataOffset).

dwDataOffset

[in] The offset value between pbyBuffer and the starts of the video data. This value should be large enough to hold the packet header. You could specify the value returned by PacketMaker_MakeVideoPacket.

***ptMediaPacket**

[in/out] Hold the returned packet. When input, you must specify the following structure members: dwStreamType, tFrameType, dwFirstUnitSecond, and dwFirstUnitMilliSecond.

If you want the generated packet to contain DI/DO or motion information, you could also specify the following members: dwDIAlert, dwDO, bMotionDetection, bMotionDetectionAlertFlag, byMotionDetectionPercent, wMotionDetectionAxis. It's a good idea to copy the packet information of the source packet to this parameter.

Return Values

S_OK

The maximum value is retrieved successfully.

ERR_INVALID_ARG

The pdwMaxHeader could not be NULL.

ERR_INVALID_HANDLE

The handle can't be NULL.

Remarks

Requirements

PacketMaker.h

See Also

PacketMaker_CreateChannel

3.4.8 PacketMaker_MakeAudioPacket

Call this function to generate an audio packet from the raw audio data (PCM).

Syntax

```
SCODE PacketMaker_MakeAudioPacket (
                                    HANDLE                  hChannel,
                                    BYTE                    * pbySrcBuffer,
                                    DWORD                   dwBuffSize,
                                    DWORD                   dwTargetBuffSize,
                                    DWORD                   *pdwUsedSize,
                                    TMediaDataPacketInfo    *ptMediaPacket
);
```

Parameters

hChannel

[in] The handle of the channel, which created by PacketMaker_CreateChannel.

pbySrcBuffer

[in] The buffer that contains the raw audio data.

dwBuffSize

[in] This is the data size that the raw audio data buffer occupies.

dwTargetBuffSize

[in] The memory size that is available to hold the encoded audio data and the packet header. This value must be large enough to hold the whole packet.

pdwUsedSize

[out] This is the size of the raw audio data that is used in a function call. Each call to this function will generate only one audio packet, so if the given data length is larger than needed, only *pdwUsedSize will be used. Caller must call again to pack the remained data.

ptMediaPacket

[in/out] Hold the returned packet. When input, you must set the pbyBuff to a valid buffer that will hold the encoded frames as well as the packet header. When returned, all packet fields are valid.

Return Values

S_OKS_OK

The maximum value is retrieved successfully.

ERR_INVALID_ARG

The following conditions are matched:

- The ptMediaPacket is NULL
- The input data size is smaller than the required data length for one packet, that is the size is smaller than audio size per frame * audio frame per packet.

ERR_INVALID_HANDLE

The handle can’t be NULL.

S_FAIL

Fail to encode audio data.

Remarks

Requirements

PacketMaker.h

See Also

PacketMaker_CreateChannel

3.4.9 PacketMaker_CreateAudioCaptureChannel

Create a audio capture channel that could pack the audio into a media packet.

Syntax

```
SCODE PacketMaker_CreateAudioCaptureChannel
(
                                    HANDLE                  hPacketMaker,
                                    HANDLE                  *phChannel,
                                    TPMACAPCHANNELOPTION    *ptChannelOptions
);
```

Parameters

hPacketMaker

[in] the handle of the PacketMaker, which created by PacketMaker_Initial.

***phChannel**

[out] the pointer to receive the handle of this channel.

***ptChannelOptions**

[in] a structure contains the settings of the channel. See TPMACAPCHANNELOPTION.

Return Values

S_OK

Create the channel successfully.

ERR_INVALID_HANDLE

The handle can't be NULL.

ERR_INVALID_ARG

The settings are incorrect.

PACKETMAKER_E_OUT_OF_MEMORY

There is not enough memory.

PACKETMAKER_E_CHANNEL_EXCEEDED

You can’t create more than 64 channels.

PACKETMAKER_E_INIT_AUDIOIN

Fail to initialize audio capture device.

S_FAIL

Fail to create the channel due to the audio encoder initialized failed.

Remarks

Requirements

PacketMaker.h

See Also

PacketMaker_DeleteAudioCaptureChannel

3.4.10 PacketMaker_DeleteAudioCaptureChannel

Delete the channel created by **PacketMaker_CreateAudioCaptureChannel**.

Syntax

```
SCODE PacketMaker_DeleteAudioCaptureChannel (
                                                    HANDLE                  *phChannel,
);
```

Parameters

***phChannel**

[in] the address of the handle of channel created by **PacketMaker_CreateAudioCaptureChannel**.

Return Values

S_OK

Delete the channel successfully.

ERR_INVALID_HANDLE

The handle can't be NULL.

PACKETMAKER_E_CHANNEL_NOT_FOUND

The channel you specified does not exit.

Remarks

Requirements

PacketMaker.h

See Also

PacketMaker_CreateAudioCaptureChannel

3.4.11 PacketMaker_SetAudioCaptureChannelOption

Call this function to update the channel option after audio capture channel has been created.

Syntax

```
SCODE PacketMaker_SetAudioCaptureChannelOption (
                                                HANDLE      hChannel,
                                                DWORD       dwOption
                                                DWORD       dwParam1
                                                DWORD       dwParam2
);
```

Parameters

hChannel

[in] the handle of the channel created by PacketMaker_CreateAudioCaptureChannel.

dwOption

[in] the channel option to be updated. The value must be one of the values in **EMPMCHOPTION**.

dwParam1

[in] the first parameter that each option needs. The corresponding value could be found in **EMPMCHOPTION**.

dwParam2

[in] the second parameter that each option needs. The corresponding value could be found in **EMPMCHOPTION**.

Return Values

S_OK

The option is set successfully.

ERR_INVALID_ARG

The option and parameters combination is wrong.

ERR_INVALID_HANDLE

The handle can't be NULL.

PACKETMAKER_E_OUT_OF_MEMORY

There is not enough memory.

PACKETMAKER_E_INIT_AUDIOIN

Fail to initialize audio capture device.

S_FAIL

Fail to initialize audio encoder.

Remarks

Requirements

PacketMaker.h

See Also

3.4.12 PacketMaker_SetPacketTime

Call this function to set packet's date and time information.

Syntax

```
SCODE PacketMaker_SetPacketTime (
                                        TMediaDataPacketInfo    *ptMediaPacket,
                                        TPacketDateTimeInfo     tpktTime
);
```

Parameters

ptMediaPacket

[in] pointer to the packet

tpktTime

[in] structure stores the date and time information.

Return Values

S_OK

Set the packet time information successfully.

ERR_INVALID_HANDLE

The handle can't be NULL.

Remarks

Requirements

PacketMaker.h

See Also

3.4.13 PacketMaker_StartAudioCapture

Call this function to start the audio capture. The captured data will start to be called back by packet callback after the capture starts.

Syntax

```
SCODE PacketMaker_StartAudioCapture (
                                                    HANDLE          hChannel,
);
```

Parameters

hChannel

[in] the handle of the channel, which created by PacketMaker_CreateAudioCaptureChannel.

Return Values

S_OK

The capture channel starts successfully.

ERR_INVALID_HANDLE

The handle can't be NULL.

Remarks

Requirements

PacketMaker.h

See Also

PacketMaker_CreateAudioCaptureChannel

3.4.14 PacketMaker_StopAudioCapture

Call this function to stop the audio capture. The callback will stop after this function called.

Syntax

```
SCODE PacketMaker_StopAudioCapture (
                                                    HANDLE          hChannel,
);
```

Parameters

hChannel

[in] the handle of the channel, which created by PacketMaker_CreateAudioCaptureChannel.

Return Values

S_OK S_OK

The capture channel stops successfully.

ERR_INVALID_HANDLE

The handle can't be NULL.

Remarks

Requirements

PacketMaker.h

See Also

PacketMaker_CreateAudioCaptureChannel

3.4.15 PacketMaker_UpdateChannelSettings

Call this function to update the channel settings.

Syntax

```
SCODE PacketMaker_UpdateChannelSettings (
                                        HANDLE      hChannel,
                                        DWORD       dwOption,
                                        DWORD       dwParam1,
                                        DWORD       dwParam2
);
```

Parameters

hChannel

[in] the handle of the channel, which created by PacketMaker_CreateAudioCaptureChannel.

dwOption

[in] The option listed in **EPMUPDATEOPTION**.

dwParam1

[in] the first parameter if needed. Please refer to **EPMUPDATEOPTION** for more information.

dwParam2

[in] the second parameter if needed. Please refer to **EPMUPDATEOPTION** for more information.

Return Values

S_OK S_OK

Set the Connection options successfully.

ERR_INVALID_HANDLE

The handle can't be NULL.

Remarks

Requirements

PacketMaker.h

See Also