VIVOTEK NETWORK DEVELPMENT PLATFORM

Data Packet Parser

Version 5.2.0.3

2017/01/20

TABLE OF CONTENTS

1. Overview

1.1 Introduction

This document describes the properties and methods supported by the VIVOTEK Data Packet Parser.

The Data Packet parser provides functions to parse and get the content of a Data Packet.

1.2 File Structure

File	Description
doc\VNDP_DataPacketParser_API.pdf	This manual document
lib\d_parsedatapacket.lib	Dynamic linking
lib\parsedatapacket.dll	Dynamic runtime library
inc\datapacketdef.h	Packet definition header file
inc\parsedatapacket.h	Packet parser header file

2. Sample Code

2.1 SaveAndLoadV3Packet

Description

User may want to save and load the TMediaDataPacketInfoV3, but the structure of TMediaDataPacketInfoV3 is more complex than TMediaDataPacketInfo. Here we will introduce how to save the packet into file and than re-load from file.

Sample Code

Save Packet

Step 1. Write out pbyTLVExt

```
if (ptPacket->tIfEx.tRv1.tExt.pbyTLVExt != NULL)
{
    fwrite(ptPacket->tIfEx.tRv1.tExt.pbyTLVExt, 1,
ptPacket->tIfEx.tRv1.tExt.dwTLVExtLen, fp);
}
else
{
    fwrite(&dwDummyData, 1, sizeof(dwDummyData), fp);
}
```

Step 2. Write out pbyBuff

```
if (ptPacket->tIfEx.tInfo.pbyBuff != NULL)
{
    fwrite(ptPacket->tIfEx.tInfo.pbyBuff, 1, ptPacket->tIfEx.tInfo.dwOffset +
    ptPacket->tIfEx.
    tInfo.dwBitstreamSize, fp);
}
else
{
    fwrite(&dwDummyData, 1, sizeof(dwDummyData), fp);
}
```

Step 3. Write out pby VExtBuf

```
if (ptPacket->pbyVExtBuf != NULL)
{
    fwrite(&ptPacket->dwVExtLen, 1, sizeof(DWORD), fp);
    fwrite(ptPacket->pbyVExtBuf, 1, ptPacket->dwVExtLen, fp);
}
else
{
    fwrite(&dwDummyData, 1, sizeof(dwDummyData), fp);
}
```

Load Packet

Step 1: initial packet

```
memset(ptPacket, 0, sizeof(TMediaDataPacketInfoV3));
ptPacket->tIfEx.tRv1.tExt.dwStructSize = sizeof(TMediaDataPacketInfoV3);
```

Step 2: read into pbyTLVExt

```
fread(&ptPacket->tIfEx.tRv1.tExt.dwTLVExtLen, 1, sizeof(DWORD), fp);
if (ptPacket->tIfEx.tRv1.tExt.dwTLVExtLen != 0)
{
    memcpy(pbyTLVExt, &ptPacket->tIfEx.tRv1.tExt.dwTLVExtLen, sizeof(DWORD));
    ptPacket->tIfEx.tRv1.tExt.dwTLVExtLen =
    HTONL(ptPacket->tIfEx.tRv1.tExt.dwTLVExtLen);
    fread(pbyTLVExt + sizeof(DWORD), 1, ptPacket->tIfEx.tRv1.tExt.dwTLVExtLen, fp);
    ptPacket->tIfEx.tRv1.tExt.pbyTLVExt = pbyTLVExt;
}
ptPacket->tIfEx.tRv1.tExt.dwTLVExtLen += sizeof(DWORD);
```

Step 3: read into pbyBuff

```
DWORD dwFrameSize = 0;
fread(&dwFrameSize, 1, sizeof(DWORD), fp);
if (dwFrameSize != 0)
{
    memcpy(pbyBuff, &dwFrameSize, sizeof(DWORD));
    dwFrameSize = HTONL(dwFrameSize);
    fread(pbyBuff + sizeof(DWORD), 1, dwFrameSize, fp);
```

```
    ptPacket->tIfEx.tInfo.pbyBuff = pbyBuff;
}
```

Step 4: read into pbyVExtBuf

```
DWORD dwVExtLen = 0;
fread(&dwVExtLen, 1, sizeof(DWORD), fp);
if (dwVExtLen)
{
    dwVExtLen = HTONL(dwVExtLen);
    fread(pbyVExtBuf, 1, dwVExtLen, fp);
    ptPacket->pbyVExtBuf = pbyVExtBuf;
}
```

Step 5: reconstruct packet

```
DataPacket_ParseV3(ptPacket);
```

Tips

Remember to set ptPacket->tIfEx.tRv1.tExt.dwStructSize = sizeof(**TMediaDataPacketInfoV3**), otherwise the **DataPacket_ParseV3** function won't parse the V3 information.

3. API Reference

3.1 Data Structure

The data structure is depicted here.

- TMediaDataPacketInfo
- TMediaDataPacketInfoV3
- TMediaDataPacketInfoEx
- TMediaPacketFirstReserved
- TVUExtInfo
- TMotionTriggerInfo
- TCaptureWinInfo
- TMediaPacketTimeZoneInfo
- TMediaAudioInfo
- TPoint
- TMotionTriggerInfoEx

3.1.1 TMediaDataPacketInfo

This structure provides the information in the Data Packet.

```
typedef struct {
                                    BYTE                *pbyBuff;
                                    DWORD               dwOffset;
                                    DWORD               dwBitstreamSize;
                                    DWORD               dwStreamType;
                                    DWORD               dwFrameNumber;
                                    TMediaDBFrameType   tFrameType;
                                    DWORD               dwFirstUnitSecond;
                                    DWORD               dwFirstUnitMilliSecond;
                                    BOOL                bFixedFrameSize;
                                    DWORD               dwAudioSamplingFreq;
                                    BYTE                byAudioChannelNum;
                                    DWORD               dwDIAlert;
                                    DWORD               dwDO;
                                    BOOL                bMotionDetection[3];
                                    BOOL                bMotionDetectionAlertFlag[3];
                                    BYTE                byMotionDetectionPercent[3];
                                    WORD                wMotionDetectionAxis[3][4];
                                    BOOL                bTimeModified;
                                    BOOL                bAudioDI;
                                    BOOL                bNoVideoSignal;
} TMediaDataPacketInfo;
```

Member

pbyBuff

The pointer of buffer containing this Data Packet.

dwOffset

The offset of buffer refer to the media bitstream, 32-bits align.

dwBitstreamSize

The size of media bitstream.

dwStreamType

The type of stream in the Data Packet, indicated by enumeration **EMediaCodecType**.

dwFrameNumber

The number of frame in the Data Packet.

tFrameType

The type of frame.

dwFirstUnitSecond

The second of first frame in the Data Packet (time in seconds from midnight, January 1, 1970).

dwFirstUnitMilliSecond

The millisecond of first frame in the Data Packet.

bFixedFrameSize

The flag indicates the frame size is fixed.

dwAudioSamplingFreq

Audio sampling frequency, if the stream has only one kind of sampling frequency, this field is ignored.

byAudioChannelNum

The channel number of audio, if the stream has only one kind of channel number, this field is ignored.

dwDIAlert

The digital input alert, each bit presents a digital input source. Bit 0 is for DI 0, bit 1 is for DI 1, bit 2 is for DI 2, and bit 3 is for ID 3. Other bits are reserved.

dwDO

Each bit is used to indicate the DO (H/L). It supports two digital output in the present. The LSB indicates the firest digital output.

bMotionDetection[3]

The flags of motion detection which indicate the motion detection windows are active or not.

bMotionDetectionAlertFlag[3]

The flags of motion detection alert.

byMotionDetectionPercent[3]

The percentages of motion detection.

wMotionDetectionAxis[3][4]

The window of motion detection, 1st element is the left coordinate of rectangular, 2nd element is the top coordinate of rectangular, 3rd element is the width of rectangular and 4th element is the height of rectangular.

bTimeModified

The flag indicates the time is modified according to timezone. If the value is FALSE, the time is dependent the time zone.

bAudioDI

The flag indicates audio packets take the DI Alert information.

bNoVideoSignal

The flag indicates the loss of video signal.

Remarks

Variable “dwDIMask” is removed in version 2.0.0.0. Variable “dwDO” is added in version 3.0.0.0. Variable “bTimeModified” and “bAudioDI” are added in version 4.0.0.0. Variable “bNoVideoSignal” is added in version 5.0.0.0.

Requirements

datapacketdef.h

3.1.2 TMediaDataPacketInfoV3

This structure provides the information in the new Data Packet (version 3). The most important part of the V3 is the capturing and cropping information, you could refer to the sample code “GetV3Information” for the detail.

```
typedef struct {
    TMediaDataPacketInfoEx      tIfEX;
    TMediaPacketTimeZoneInfo    tTZ;
    DWORD                       dwUTCTime;
    TVUExtInfo                  tVUExt;
    BYTE                        *pbyVExtBuf;
    DWORD                       dwVExtLen;
    BOOL                        bTemperatureAlert;
    BYTE                        byMotionNumber;
    TMotionTriggerInfoEx        *ptMotionInfoEx;
} TMediaDataPacketInfoV3;
```

Member

tIfEx

TMediaDataPacketInfoEx

tTZ

TMediaPacketTimeZoneInfo

dwUTCTime

The UTC time of this packet.This value could be got from dwFristUnitSecond and time zone information, too.

tVUExt

TVUExtInfo

pbyVExtBuf

Video further extension. If the value is null, it means this packet don’t have such extension.

dwVExtLen

The size of the pbyVExtBuf.

bTemperatureAlert

The flag of temperature alert.

byMotionNumber

Number of new motion window. For fisheye or the cameras with more than 3 motion windows, user should check this flag for getting correct motion window information.

ptMotionInfoEx

Pointer to the new motion window.

Requirements

datapacketdef.h

3.1.3 TMediaDataPacketInfoEx

This structure provides the information in the Data Packet.

```
typedef struct {
                                    TMediaDataPacketInfo    tInfo;
                                    UMediaPktReserved1      tRv1;
                                    DWORD                   dwWidth;
                                    DWORD                   dwHeight;
                                    DWORD                   dwWidthPadLeft;
                                    DWORD                   dwWidthPadRight;
                                    DWORD                   dwHeightPadTop;
                                    DWORD                   dwHeightPadBottom;
} TMediaDataPacketInfoEx;
```

Member

tInfo

TMediaDataPacketInfo

tRv1

UMediaPktReserved1. It is a union of TMediaPacketFirstReserved (tExt) and void*(apvReserved[4]). In most case, we use the tExt field. For example:PacketV3.tIfEx.tRv1.tExt.pbyTLVExt.

dwWidth

The width of the video frame if it's video (includes padding if any).

dwHeight

The height of the video frame if it's video (includes padding if any).

dwWidthPadLeft

Padded width of left, the value is usually zero.

dwWidthPadRight

Padded width of right, the value is usually zero.

dwHeightPadTop

Padded height of top, the value is usually zero.

dwHeightPadBottom

Padded height of bottom, the value is usually zero.

Requirements

datapacketdef.h

3.1.4 TMediaPacketFirstReserved

This structure provides the information of extension data of a frame.

```
typedef struct {
                                        BYTE            *pbyTLVExt;
                                        DWORD           dwTLVExtLen;
                                        DWORD           dwStructureSize;
} TMediaPacketFirstReserved;
```

Member

pbyTLVExt

The pointer to the tag/length/data extension of a frame. The pointer would point to the location after media data in pbyBuff.

dwTLVExtLen

The length of the pbyTLVExt.

dwStructureSize

The size of the structure. If this is 0, it means the packet is Ex only, else this maps the size of the overall packet structure.

Requirements

datapacketdef.h

3.1.5 TVUExtInfo

This structure provides the information in the Video Extension.

```
typedef struct {
                                    DWORD               dwPIR;
                                    DWORD               dwWLLed;
                                    TCaptureWinInfo     tCapWinInfo;
                                    DWORD               dwTamperingAlert;
                                    TMotionTriggerInfo  *ptMTI;
} TVUExtInfo;
```

Member

dwPIR

The PIR status, HIGH word means the PIR enabled flag, LOW word contains the values.

dwWLLed

The status for white light LED.

tCapWinInfo

The capture window info **TCaptureWinInfo.**

dwTamperingAlert

Tamperingalert info**.**

ptMTI

Point to the data in **TMediaDataPacketInfo.**

Requirements

datapacketdef.h

3.1.6 TCaptureWinInfo

This structure provides the information of Capture Window.

```
typedef struct {
    BOOL    bWithInfo;
    WORD    wCapW;
    WORD    wCapH;
    WORD    wOffX;
    WORD    wOffY;
    WORD    wCropW;
    WORD    wCropH;
} TCaptureWinInfo;
```

Member

bWithInfo

If the information is valid.

wCapW

Capture Width.

wCapH

Capture Height.

wOffX

Offset X.

wOffY

Offset Y.

wCropW

Cropping Width.

wCropH

Cropping Height.

Requirements

datapacketdef.h

3.1.7 TMediaPacketTimeZoneInfo

This structure provides the information of time zone and daylight saving.

```
typedef struct {
                                    BOOL        bTimeZone;
                                    long        lDLSaving;
                                    long        lOffsetSeconds;
} TMediaPacketTimeZoneInfo;
```

Member

bTimeZone

If the packet contains time zone information, if no, the following fields are from client machine.

lDLSaving

The daylight saving time in seconds (if 0 it means no day-light saving), -3600 for most case.

lOffsetSeconds

Offset of seconds from GMT time; for example Taipei will be 8 * 3600 = 28800.

Requirements

datapacketdef.h

3.1.8 TMediaAudioInfo

This structure is used to get the size of audio units in a Data Packet through the DataPacket_GetEveryAudioInfo function.

```
typedef struct {
                                        DWORD                   dwSize;
} TMediaAudioInfo;
```

Member

dwSize

The size of an audio unit.

Requirements

parsedatapacket.h

3.1.9 TPoint

This structure contains x-axis and y-axis of a certain position.

```
typedef struct {
                                    WORD        wX;
                                    WORD        wY;
} TPoint;
```

Member

wX

X-axis of the point.

wY

Y-axis of the point.

Requirements

datapacketdef.h

3.1.10 TMotionTriggerInfoEx

This structure provides the information of new motion window.

```
typedef struct {
    BYTE        byWindowNumber;
    BOOL        bMotionDetection;
    BOOL        bMotionDetectionAlertFlag;
    BYTE        byMotionDetectionPercent;
    TPoint      tPoint1;
    TPoint      tPoint2;
    TPoint      tPoint3;
    TPoint      tPoint4;
} TMotionTriggerInfoEx;
```

Member

byWindowNumber

The index of motion detection.

bMotionDetection

The flags of motion detection.

bMotionDetectionAlertFlag

The flags of motion detection alert.

byMotionDetectionPercent

The percentage of motion detection.

tPoint1

The first point of motion detection.

tPoint2

The second point of motion detection.

tPoint3

The third point of motion detection.

tPoint4

The fourth point of motion detection.

Requirements

datapacketdef.h

3.2 Enumeration

The enumeration used is depicted here.

- EMediaCodecType
- TMediaDBFrameType

3.2.1 EMediaCodecType

This enumeration indicates the media codec type.

```
typedef enum {
                                    mctJPEG         = 0x0001,
                                    mctH263         = 0x0002,
                                    mctMP4V         = 0x0004,
                                    mctH264         = 0x0008,
                                    mctHEVC         = 0x0010,
                                    mctG7221        = 0x0100,
                                    mctG729A        = 0x0200,
                                    mctAAC          = 0x0400,
                                    mctGAMR         = 0x0800,
                                    mctSAMR         = 0x1000,
                                    mctG711         = 0x2000,
                                    mctG711A        = 0x4000,
                                    mctG726         = 0x8000,
} EMediaCodecType;
```

Members

mctJPEG

The codec type is JPEG (image, video).

mctH263

The codec type is H.263 (video).

mctMP4V

The codec type is MPEG-4 video (video).

mctH264

The codec type is H.264 video (video).

mctHEVC

The codec type is H.265 video (video).

mctG7221

The codec type is G.722.1 (audio).

mctG729A

The codec type is G.729A (audio).

mctAAC

The codec type is AAC (audio).

mctGAMR

The codec type is AMR (audio).

mctSAMR

The codec type is SAMR (audio).

mctG711

The codec type is G.711 (audio).

mctG711A

The codec type is G.711A (audio).

mctG726

The codec type is G.726 (audio).

Requirements

mediatypedef.h

3.2.2 TMediaDBFrameType

This enumeration indicates the frame type of media.

```
typedef enum {
                                        MEDIADB_FRAME_INTRA     = 0,
                                        MEDIADB_FRAME_PRED      = 1,
                                        MEDIADB_FRAME_BIPRED    = 2
} TMediaDBFrameType;
```

Members

MEDIADB_FRAME_INTRA

The intra frame.

MEDIADB_FRAME_PRED

The prediction frame.

MEDIADB_FRAME_BIDIR

The bi-direction prediction frame.

Requirements

mediatypedef.h

3.3 API Definition

The API definition is depicted here.

- DataPacket_Parse
- DataPacket_ParseV3
- DataPacket_GetEveryAudioInfo
- DataPacket_ParseTemperatureAlert
- DataPacket_ParseDO
- DataPacket_AllocatePolygonMotion
- DataPacket_Free
- DataPacket_ParsePolygonMotion
- DataPacket_GetPolygonMotionNumber

3.3.1 DataPacket_Parse

Parse the Data Packet in the input buffer.

Syntax

```
SCODE DataPacket_Parse (
                                                    TMediaDataPacketInfo      *ptDataPacketInfo
);
```

Parameters

ptDataPacketInfo

[in/out] A pointer of data structure storing information of Data Packet.

Return Values

S_OK

Parse Data Packet successfully.

S_FAIL

Parse Data Packet failed.

Remarks

Before calling this function, the field of pbyBuff in the Data Packet structure must be valid.

Requirements

parsedatapacket.h

3.3.2 DataPacket_ParseV3

Parse the Data Packet in the input buffer into **TMediaDataPacketInfoV3**.

Syntax

```
SCODE DataPacket_ParseV3 (
                                                    TMediaDataPacketInfoV3    * ptDataPacketInfoV3
);
```

Parameters

ptDataPacketInfoV3

[in/out] A pointer of data structure storing information of Data Packet.

Return Values

S_OK

Parse Data Packet successfully.

S_FAIL

Parse Data Packet failed.

Remarks

The usage of **DataPacket_ParseV3** is more complex than **DataPacket_Parse**, please refer to the sample code "SaveAndLoadV3Packet" for the usage.

Requirements

parsedatapacket.h

3.3.3 DataPacket_GetEveryAudioInfo

This function gets the size and buffer address of every audio unit in a Data Packet.

Syntax

```
SCODE DataPacket_GetEveryAudioInfo (
                                        BYTE            *pbyInBuf,
                                        TMediaAudioInfo *ptAudioInfo
);
```

Parameters

pbyInBuf

[in] The pointer of buffer containing an audio Data Packet.

ptAudioInfo

[out] The pointer of structure to store the list of audio information, the size must be larger than audio number * sizeof(TMediaAudioInfo).

Return Values

S_OK

Get audio info successfully.

S_FAIL

Get audio info failed.

Requirements

parsedatapacket.h

3.3.4 DataPacket_ParseTemperatureAlert

This function parse the temperature alert flag in video extended.

Syntax

```
SCODE DataPacket_ParseTemperatureAlert (
                                        BYTE        *pbyData,
                                        DWORD       dwLength,
                                        BOOL        *pbTemperatureAlert
);
```

Parameters

pbyData

[in] The pointer to the video extended data.

dwLength

[in] The length of the extended data.

pbTemperatureAlert

[out] The output of parsed flag.

Return Values

S_OK

Parse the data successfully.

S_FAIL

Parse the data failed.

Requirements

parsedatapacket.h

3.3.5 DataPacket_ParseDO

s function parse the DO value in video extended for VS8xxx.

Syntax

```
SCODE DataPacket_ParseDO (
                                        BYTE        *pbyData,
                                        DWORD       dwLength,
                                        DWORD       *pdwDO
);
```

Parameters

pbyData

[in] The pointer to the video extended data.

dwLength

[in] The length of the extended data.

pbTemperatureAlert

[out] The output of DO value.

Return Values

S_OK

Parse the data successfully.

S_FAIL

Parse the data failed.

Requirements

parsedatapacket.h

3.3.6 DataPacket_AllocatePolygonMotion

This function parse motion window information and ALLOCATE the memory needed.

Syntax

```
SCODE DataPacket_AllocatePolygonMotion (
                                        BYTE            *pbyData,
                                        DWORD           dwLength,
                                        TMotionTriggerInfoEx    **pptMotion,
                                        DWORD           *pdwWinNumber
);
```

Parameters

pbyData

[in] The pointer to the video extended data.

dwLength

[in] The length of the extended data.

pptMotion

[out] The pointer to **TMotionTriggerInfoEx**.

pdwWinNumber

[out] The number of motion window.

Return Values

S_OK

Parse the data successfully.

S_FAIL

Parse the data failed.

Remarks

Application should call **DataPacket_Free** to free *ppMotion or it will cause memory leak.

Requirements

parsedatapacket.h

3.3.7 DataPacket_Free

This function frees the memory allocated by parsedatapacket.

Syntax

```
SCODE DataPacket_Free (
                                                    BYTE                    *pbyBuf
);
```

Parameters

pbyBuf

[in] The pointer to the memory.

Return Values

S_OK

Free the memory successfully.

Requirements

parsedatapacket.h

3.3.8 DataPacket_ParsePolygonMotion

This function parse polygon motion window information and fill the motion window information in TMediaDataPacketInfoV3::ptMotionInfoEx.

Syntax

```
SCODE DataPacket_ParsePolygonMotion (
                                        TMediaDataPacketInfoV3   *ptV3,
                                        DWORD                    dwInBufSize,
                                        DWORD                    *pdwOutMotionNumber
);
```

Parameters

ptV3

[in] The pointer to TMediaDataPacketInfoV3.

dwInputBufSize

[in] The number of buffer pointed by TMediaDataPacketInfoV3::ptMotionInfoEx.

pdwOutMotionNumber

[out] The number of motion window.

Return Values

S_OK

Parse the data successfully.

S_FAIL

Parse the data failed.

Remarks

Users can get the number of motion window through DataPacket_GetPolygonMotionNumber. Before calling this function, application should allocate enough memory and set this memory to TMediaDataPacketInfoV3::ptMotionInfoEx.

Requirements

parsedatapacket.h

3.3.9 DataPacket_GetPolygonMotionNumber

This function gets the number of motion window.

Syntax

```
SCODE DataPacket_GetPolygonMotionNumber (
                                        BYTE        *pbyData,
                                        DWORD       dwLength,
                                        DWORD       *pdwOutMotionNumber
);
```

Parameters

pbyData

[in] Pointer to the data.

dwLength

[in] The size of the data.

pdwOutMotionNumber

[out] The number of motion window.

Return Values

S_OK

Parse the data successfully.

S_FAIL

Parse the data failed.

Requirements

parsedatapacket.h