IEI Technology Corp.

MODEL:

HDC-3x Series SDK (Windows)

A SDK software development kit for the HDC-3x Series

User Manual

Rev. 2.00 – 27 November, 2012

Revision

Date	Version	Changes
27 November, 2012	2.00	Updated for new software version v2.00
4 May, 2011	1.02	Added information for the HDC-302E
13 April, 2011	1.01	Added information for HDC-301 and HDC-301E and renamed the manual to HDC-3x Series
13 January, 2011	1.00	Initial release

Copyright

COPYRIGHT NOTICE

The information in this document is subject to change without prior notice in order to improve reliability, design and function and does not represent a commitment on the part of the manufacturer.

In no event will the manufacturer be liable for direct, indirect, special, incidental, or consequential damages arising out of the use or inability to use the product or documentation, even if advised of the possibility of such damages.

This document contains proprietary information protected by copyright. All rights are reserved. No part of this manual may be reproduced by any mechanical, electronic, or other means in any form without prior written permission of the manufacturer.

TRADEMARKS

All registered trademarks and product names mentioned herein are used for identification purposes only and may be trademarks and/or registered trademarks of their respective owners.

Table of Contents

Chapter

1

Driver and SDK Installation

1.1 Overview

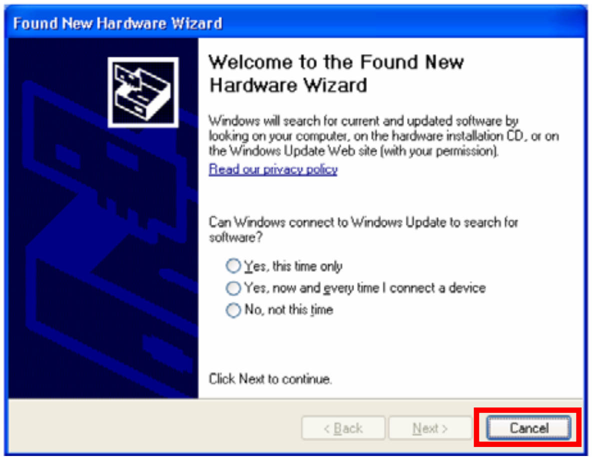
A CD is shipped with the video capture card. The CD contains a driver for the video capture controllers on the card. When the video capture card is installed on the system, the driver must be installed. Failure to install the driver means that that video capture card cannot be detected by the system.

This manual includes SDK information for the HDC-3x Series, which includes:

- HDC-301
- HDC-301E
- HDC-302E
- HDC-304E

NOTE:

The Found New Hardware Wizard will automatically start when the system detects the video capture card (see the image below). Click **Cancel** to exit the wizard and follow the steps described in this chapter to install the driver and the HDCapture SDK.



Found New Hardware Wizard

Welcome to the Found New Hardware Wizard

Windows will search for current and updated software by looking on your computer, on the hardware installation CD, or on the Windows Update Web site (with your permission).

Read our privacy policy

Can Windows connect to Windows Update to search for software?

○ Yes, this time only

○ Yes, now and every time I connect a device

○ No, not this time

Click Next to continue.

1.2 Driver Installation

To install the HDC-3x Series SDK (Windows) driver, please follow the steps below: If the HDC-3x Series SDK (Windows) driver is already installed, please refer to **Section 1.2.1** to uninstall the driver first.

NOTE:

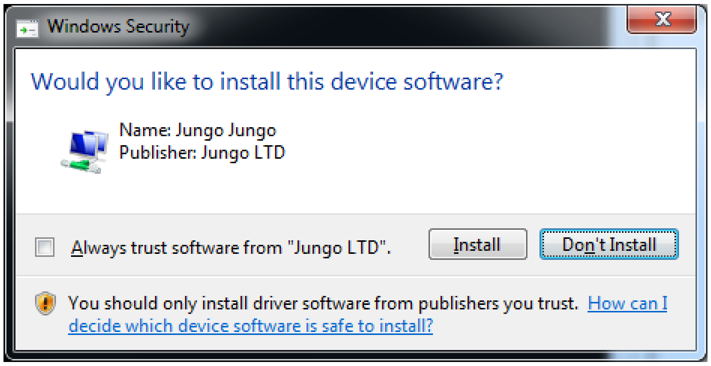
If the **User Access Control** dialog box appears during installation, click **Yes** to continue.

Step 1: Make sure to log in the system as the administrator.

Step 2: Insert the driver CD.

Step 3: Locate the "Driverinstaller.bat" file in the driver CD. Double click it.

Step 4: A confirmation window appears. Click **Install**.



Windows Security

Would you like to install this device software?

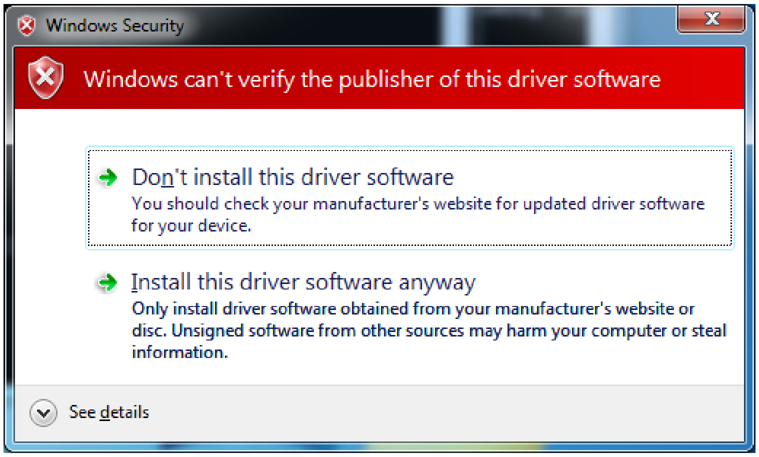
Name: Jungo Jungo
Publisher: Jungo LTD

□ Always trust software from "Jungo LTD".

You should only install driver software from publishers you trust. How can I decide which device software is safe to install?

Figure 1-1: Windows Security

Step 5: If the following window appears, click **Install this driver software Anyway**.



Windows Security

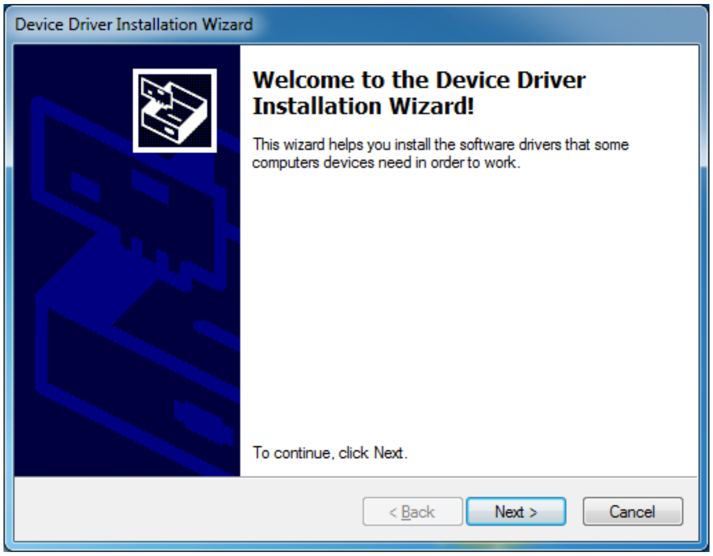
Windows can't verify the publisher of this driver software

Don't install this driver software
You should check your manufacturer's website for updated driver software for your device.

Install this driver software anyway
Only install driver software obtained from your manufacturer's website or disc. Unsigned software from other sources may harm your computer or steal information.

Figure 1-2: Windows Warning Window

Step 6: The Device Driver Installation Wizard appears. Click **Next** to start.



Device Driver Installation Wizard

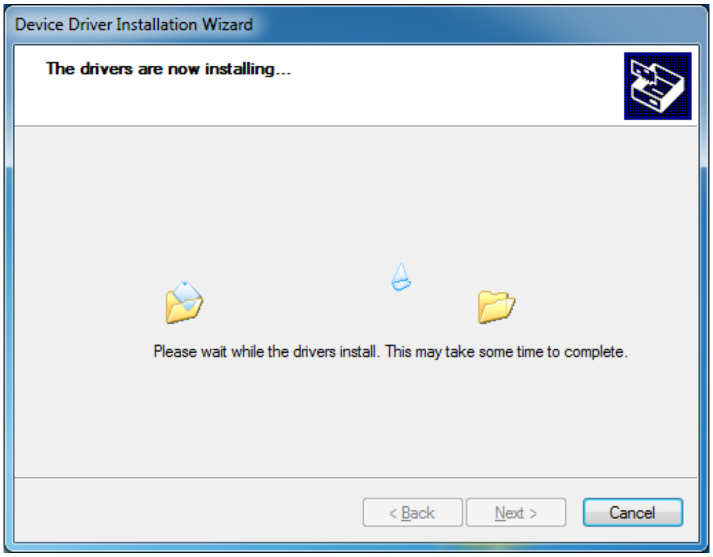
Welcome to the Device Driver Installation Wizard!

This wizard helps you install the software drivers that some computers devices need in order to work.

To continue, click Next.

Figure 1-3: Device Driver Installation Wizard

Step 7: The driver starts to install and the screen in **Figure 1-4** appears.



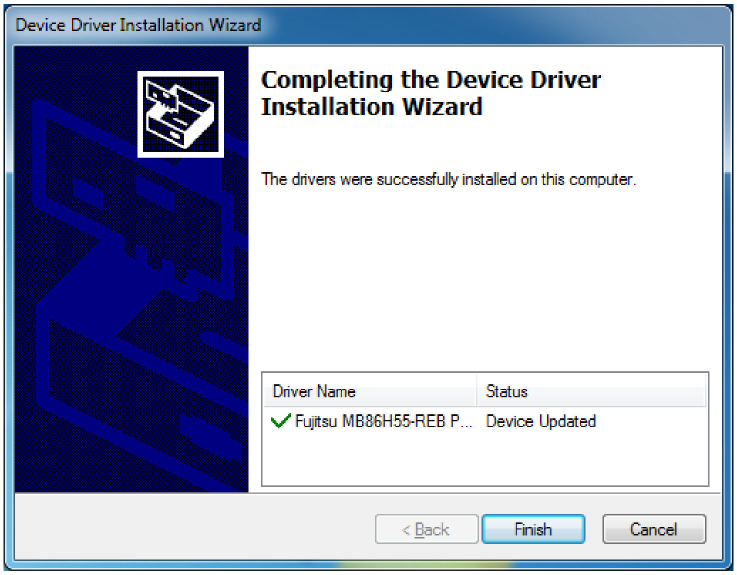
Device Driver Installation Wizard

The drivers are now installing...

Please wait while the drivers install. This may take some time to complete.

Figure 1-4: Driver Installing

Step 8: When the driver installation is complete, the screen in **Figure 1-5** appears. Click the **Finish** button to finish driver installation.



Device Driver Installation Wizard

Completing the Device Driver Installation Wizard

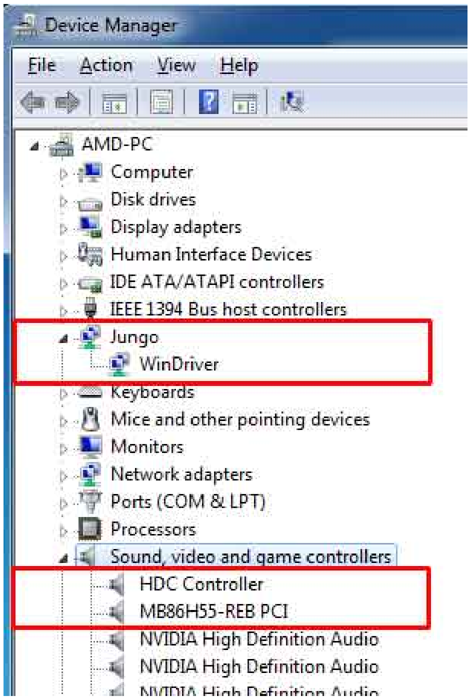
The drivers were successfully installed on this computer.

Driver Name	Status
✓ Fujitsu MB86H55-REB P...	Device Updated

Figure 1-5: Driver Installation Complete

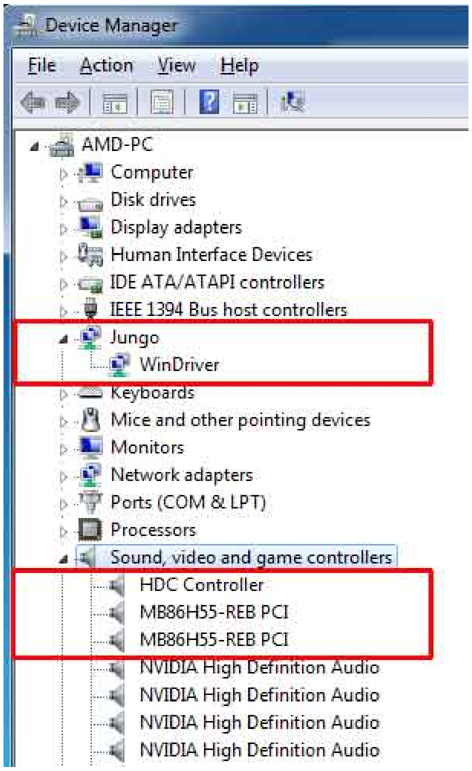
Step 9: Check the device manager in the Windows control panel to ensure the driver (MB86H55-REB PCI, HDC controller and WinDriver) has been properly installed. The installed driver is slightly different on different models. See **Figure 1-6** for the details.

HDC-301 and HDC-301E



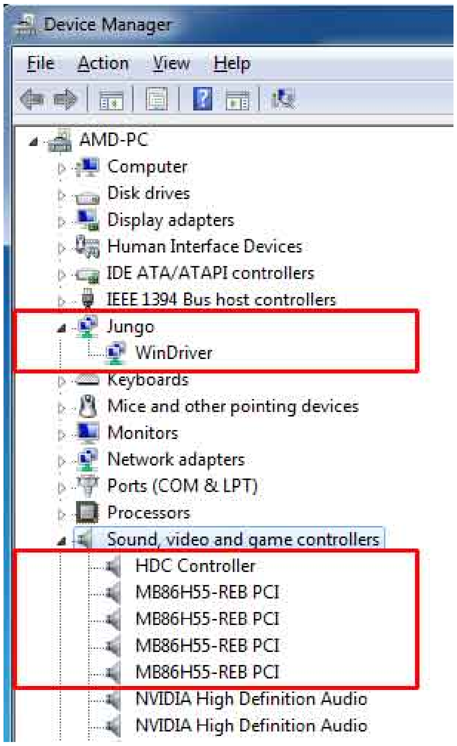
AMD-PC
Computer
Disk drives
Display adapters
Human Interface Devices
IDE ATA/ATAPI controllers
IEEE 1394 Bus host controllers
Jungo
WinDriver
Keyboards
Mice and other pointing devices
Monitors
Network adapters
Ports (COM & LPT)
Processors
Sound, video and game controllers
HDC Controller
MB86H55-REB PCI
NVIDIA High Definition Audio
NVIDIA High Definition Audio
NVIDIA High Definition Audio

HDC-302E



AMD-PC
Computer
Disk drives
Display adapters
Human Interface Devices
IDE ATA/ATAPI controllers
IEEE 1394 Bus host controllers
Jungo
WinDriver
Keyboards
Mice and other pointing devices
Monitors
Network adapters
Ports (COM & LPT)
Processors
Sound, video and game controllers
HDC Controller
MB86H55-REB PCI
MB86H55-REB PCI
NVIDIA High Definition Audio
NVIDIA High Definition Audio
NVIDIA High Definition Audio
NVIDIA High Definition Audio

HDC-304E


Device Manager

File Action View Help

AMD-PC
Computer
Disk drives
Display adapters
Human Interface Devices
IDE ATA/ATAPI controllers
IEEE 1394 Bus host controllers
Jungo
WinDriver
Keyboards
Mice and other pointing devices
Monitors
Network adapters
Ports (COM & LPT)
Processors
Sound, video and game controllers
HDC Controller
MB86H55-REB PCI
MB86H55-REB PCI
MB86H55-REB PCI
MB86H55-REB PCI
NVIDIA High Definition Audio
NVIDIA High Definition Audio

Figure 1-6: Device Manager

1.2.1 Driver Installation in 64-bit Windows 7 OS

NOTE:

The HDC-301/301E series does not support Windows 7 64-bit operating system. Please install the HDC-301/301E series in a system with other OS, such as Windows 7 32-bit.

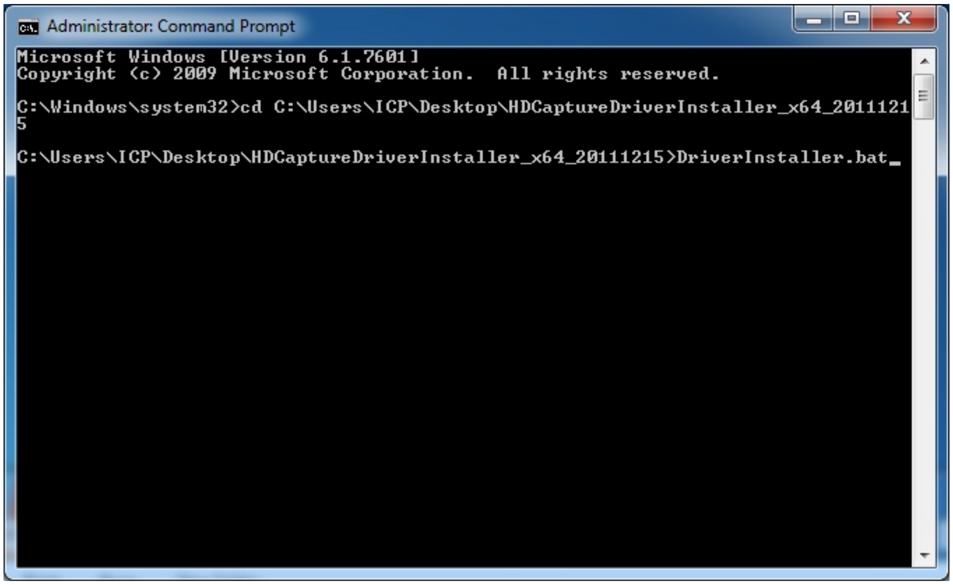
To install the driver in a 64-bit Windows 7 operating system, please do the followings:

Step 1: Make sure to log in the system as the administrator.

Step 2: Insert the driver CD.

Step 3: Launch the Command Prompt application in Windows 7 as an administrator (right click the Command Prompt and select “Run as administrator”).

Step 4: In the Command Prompt window, specify the 64-bit driver directory. Then, type **DriverInstaller** to install the driver to the system.



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>cd C:\Users\ICP\Desktop\HDCaptureDriverInstaller_x64_2011121
5

C:\Users\ICP\Desktop\HDCaptureDriverInstaller_x64_20111215>DriverInstaller.bat_
```

Figure 1-7: Command Prompt – Driver Installation

Step 5: Follow **Step 5** ~ **Step 8** in **Section 1.2** to complete installing the driver to a 64-bit Windows 7 operating system.

Step 6: Check the device manager in the Windows control panel to ensure the driver (MB86H55-REB PCI, DEVICE and WinDriver) has been properly installed. See **Figure 1-8** for the details.

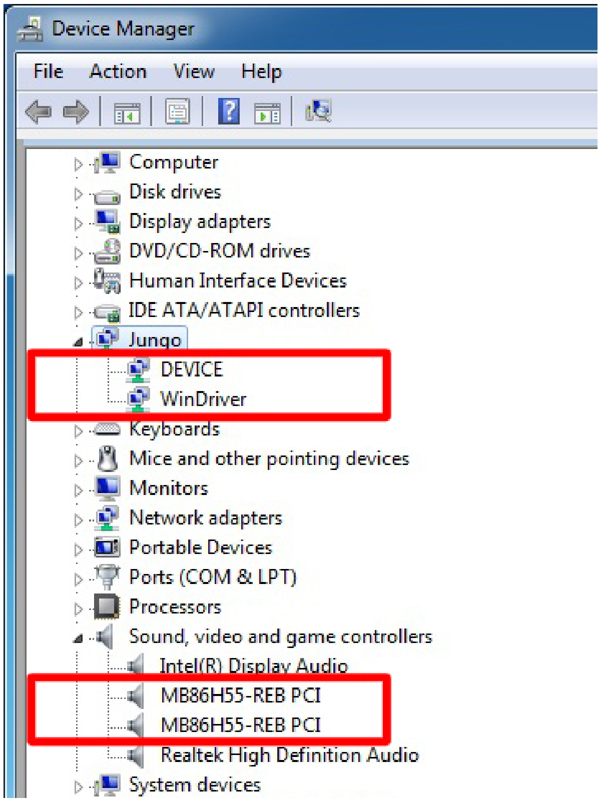


Figure 1-8: Device Manager – 64-bit OS

1.2.2 Uninstall Driver

To uninstall the driver, please follow the steps below.

Step 1: Make sure to login the system as the administrator.

Step 2: Locate the "Driveruninstaller.bat" file in the driver CD. Double click it to uninstall the driver.

The console window pop-up and all drivers will be uninstalled.

1.3 Software Installation

The HDC-3x series comes with a video capture application – HDCapture SDK. This section describes how to install the application in Windows environment.

1.3.1 System Requirements

The supported OS versions are listed below:

- Microsoft Windows XP SP2 32-bit
- Microsoft Windows 7 32-bit
- Microsoft Windows 7 64-bit (not supported by the HDC-301/301E series)

After installing the driver, the following programs must be installed in order to use the HDCapture SDK:

- Microsoft .NET Framework 3.0/3.5/4.0
- Microsoft DirectX 9.0c
- Win7DSFilterTweaker tool (for Windows 7 OS only)
- Visual C++ 2005 & 2008 Redistributable

Please download the setup files of these programs from the official websites and install these programs in the system. For detailed setup procedures for some of the above programs, please refer to **Appendix A**.

NOTE:

For the 64-bit Windows 7 operating system, the Microsoft .NET Framework 4.0 must be installed.

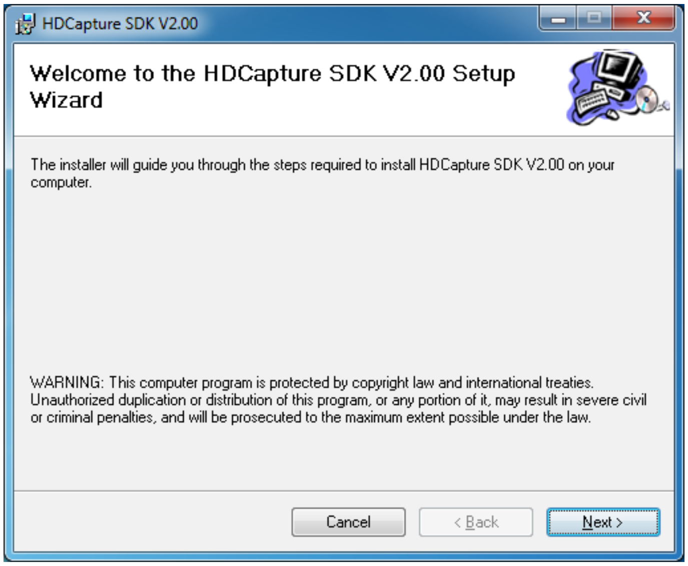
1.3.2 HDCapture SDK Installation

To install the HDCapture SDK, please follow the steps below.

Step 1: Insert the driver CD.

Step 2: Locate the **HDCaptureSDK_x86_Vxxxx.msi** file in the driver CD (HDCaptureSDK_x86_VxxxxR.msi R: released version; HDCaptureSDK_x86_VxxxxD.msi D: debug version). Double click the setup file to start the installation. The user can also download the latest setup file from IEI website.

Step 3: The HDCapture Setup Wizard welcome window appears. Click **Next** to start.



HDCapture SDK V2.00

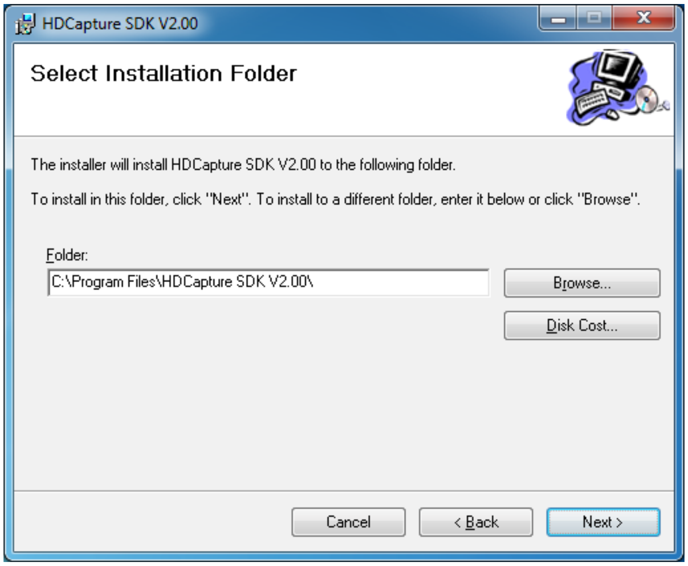
Welcome to the HDCapture SDK V2.00 Setup Wizard

The installer will guide you through the steps required to install HDCapture SDK V2.00 on your computer.

WARNING: This computer program is protected by copyright law and international treaties. Unauthorized duplication or distribution of this program, or any portion of it, may result in severe civil or criminal penalties, and will be prosecuted to the maximum extent possible under the law.

Figure 1-9: HDCapture Setup Wizard

Step 4: Select a folder for HDCapture installation in **Figure 1-10**. Click **Next** to continue.


HDCapture SDK V2.00

Select Installation Folder

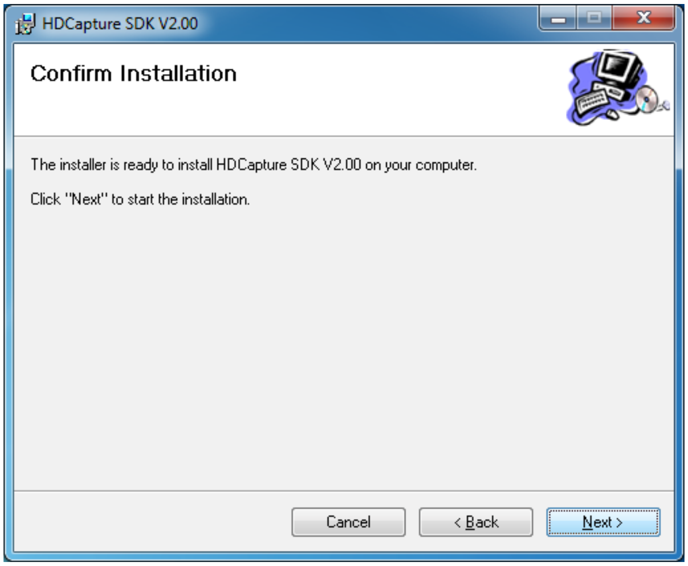
The installer will install HDCapture SDK V2.00 to the following folder.

To install in this folder, click "Next". To install to a different folder, enter it below or click "Browse".

Folder:
C:\Program Files\HDCapture SDK V2.00\

Figure 1-10: Select Installation Folder

Step 5: The following screen appears. Click **Next** to confirm the installation.


HDCapture SDK V2.00

Confirm Installation

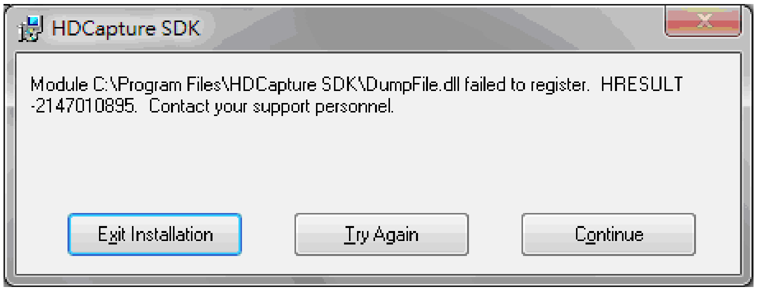
The installer is ready to install HDCapture SDK V2.00 on your computer.

Click "Next" to start the installation.

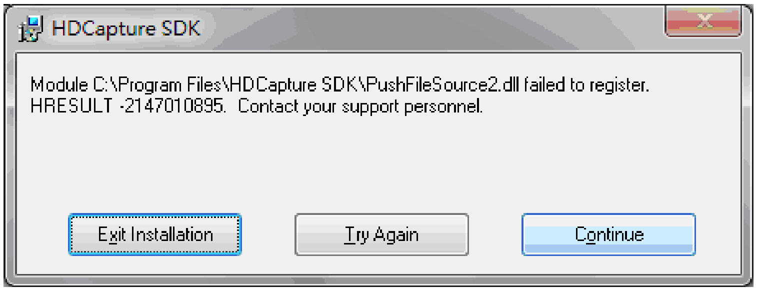
Figure 1-11: Confirm Installation

Step 6: The system starts installing the HDCapture.

Step 7: If an error happens during the installation (as shown in **Figure 1-12**), click **Continue** to continue the installation.



Module C:\Program Files\HDCapture SDK\DumpFile.dll failed to register. HRESULT -2147010895. Contact your support personnel.



Module C:\Program Files\HDCapture SDK\PushFileSource2.dll failed to register. HRESULT -2147010895. Contact your support personnel.

Figure 1-12: Installation Error Messages

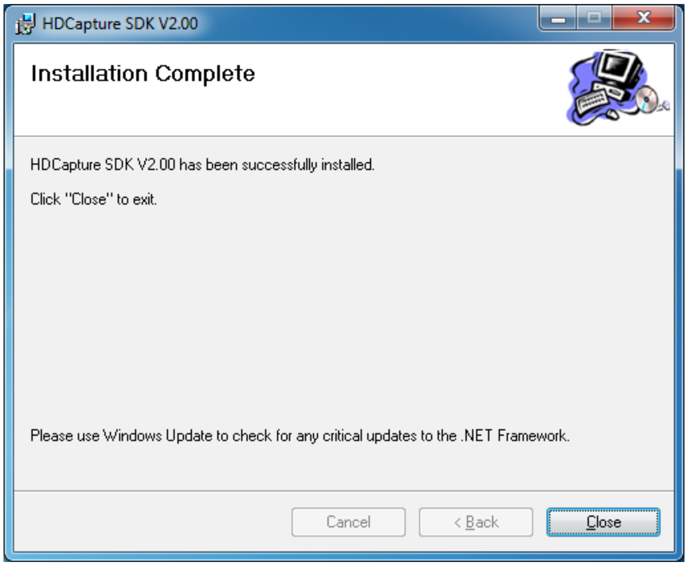
NOTE:

Since "DumpFile.dll" and "PushFileSource2.dll " are DirectShow filters, the user must register them before use. If the error described in **Step 7** occurs, please use one of the following methods to register after the HDCapture SDK installation:

1. Start Menu -> Programs -> HDCapture SDK V1.01 -> InstallFilter. or

2. Go to the installation folder and click **InstallFilter.bat**.

Step 8: When the HDCapture SDK is successfully installed, the following window appears. Click **Close** to exit.



HDCapture SDK V2.00 has been successfully installed.

Click "Close" to exit.

Please use Windows Update to check for any critical updates to the .NET Framework.

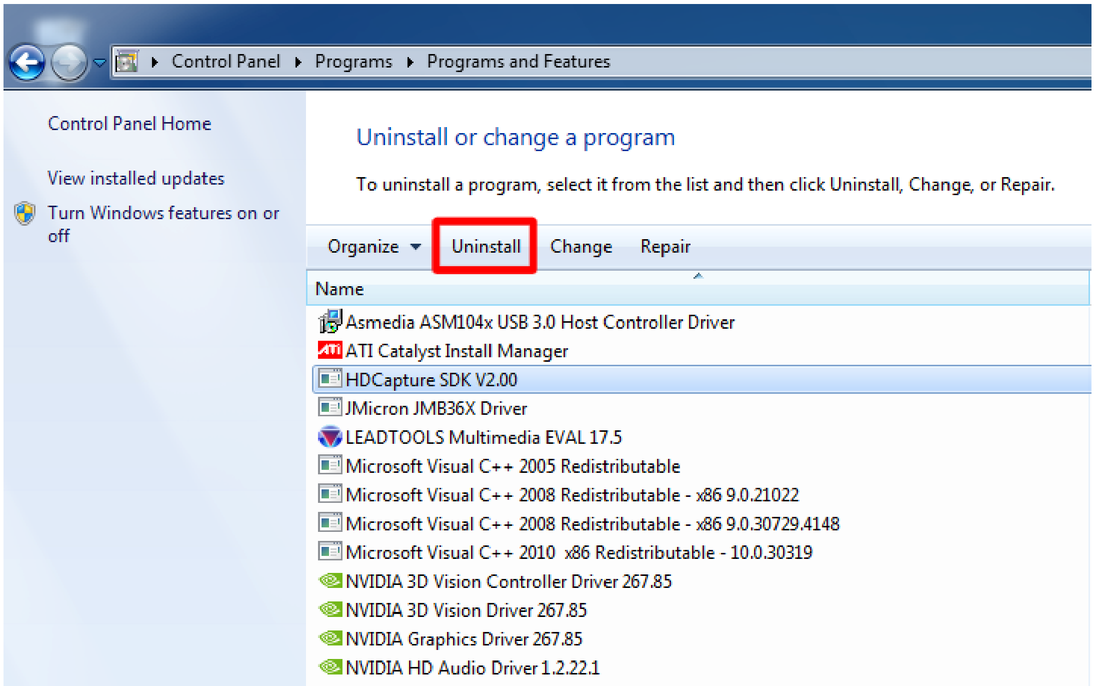
Figure 1-13: Installation Complete

1.3.3 Uninstall HDCapture SDK

To uninstall the HDCapture SDK, follow the steps below.

Step 1: Select **Control Panel** → **Programs** → **Programs and Features**.

Step 2: Select HDCapture SDK and click the **Uninstall** button to uninstall the HDCapture SDK (**Figure 1-14**).


Control Panel ▸ Programs ▸ Programs and Features
Control Panel Home
View installed updates
Turn Windows features on or off
Organize
Uninstall
Change
Repair
Name

Uninstall or change a program

To uninstall a program, select it from the list and then click Uninstall, Change, or Repair.

- Asmedia ASM104x USB 3.0 Host Controller Driver
- ATI Catalyst Install Manager
- HDCapture SDK V2.00
- JMicron JMB36X Driver
- LEADTOOLS Multimedia EVAL 17.5
- Microsoft Visual C++ 2005 Redistributable
- Microsoft Visual C++ 2008 Redistributable - x86 9.0.21022
- Microsoft Visual C++ 2008 Redistributable - x86 9.0.30729.4148
- Microsoft Visual C++ 2010 x86 Redistributable - 10.0.30319
- NVIDIA 3D Vision Controller Driver 267.85
- NVIDIA 3D Vision Driver 267.85
- NVIDIA Graphics Driver 267.85
- NVIDIA HD Audio Driver 1.2.22.1

Figure 1-14: Uninstall HDCapture SDK

Step 3: A confirmation window appears. Click **Yes** to uninstall the HDCapture SDK.

Chapter

2

HDCapture SDK Application

2.1 HDCapture SDK Overview

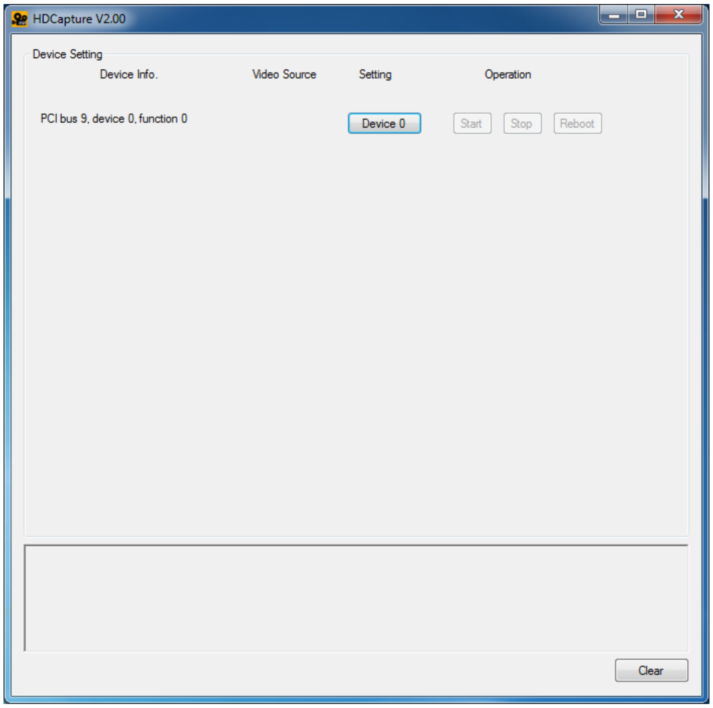
The HDCapture SDK is a video capture tool that allows user to capture video through the HDMI input ports in Windows environment. The HDCapture SDK also includes decoding function that decodes the video signal for video output to the HDMI-enabled display device.

2.2 Video Capture

To use the HDCapture SDK to capture video, follow the steps below. If the older version of the HDCapture SDK is already installed, please refer to **Section 1.3.3** to uninstall it.

Step 1: Launch the HDCapture SDK. The Device Setting on the right side panel of the HDCapture SDK is varied based on the installed video capture card as shown in **Figure 2-1**. The best resolution to view HDCapture SDK is 1280 x 1024 or above.

HDC-301 and HDC-301E



HDCapture V2.00

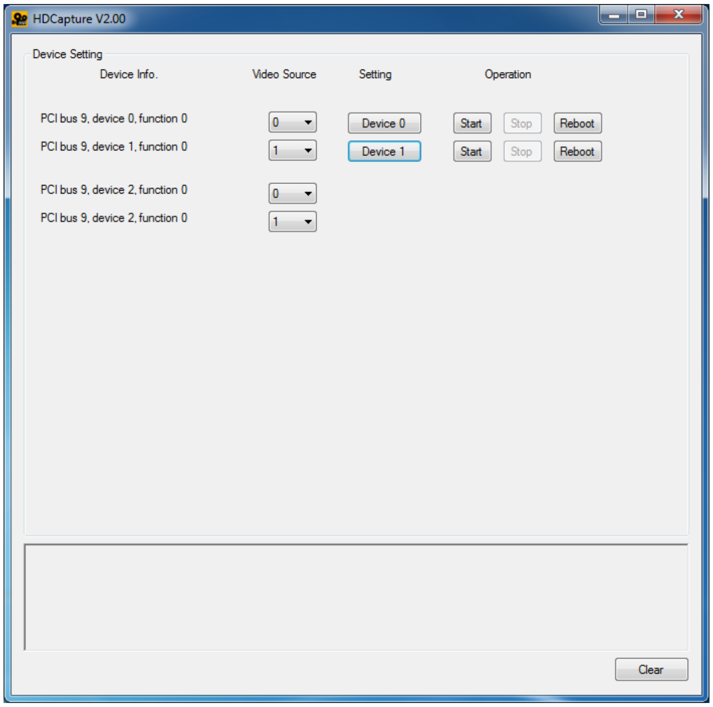
Device Setting

Device Info.: PCI bus 9, device 0, function 0

Video Source

Setting: Device 0

HDC-302E


HDCapture V2.00
Device Setting
Device Info.
Video Source
Setting
Operation
PCI bus 9, device 0, function 0 | 0 | Device 0 | Start | Stop | Reboot
PCI bus 9, device 1, function 0 | 1 | Device 1 | Start | Stop | Reboot
PCI bus 9, device 2, function 0 | 0
PCI bus 9, device 2, function 0 | 1
Clear

HDC-304E

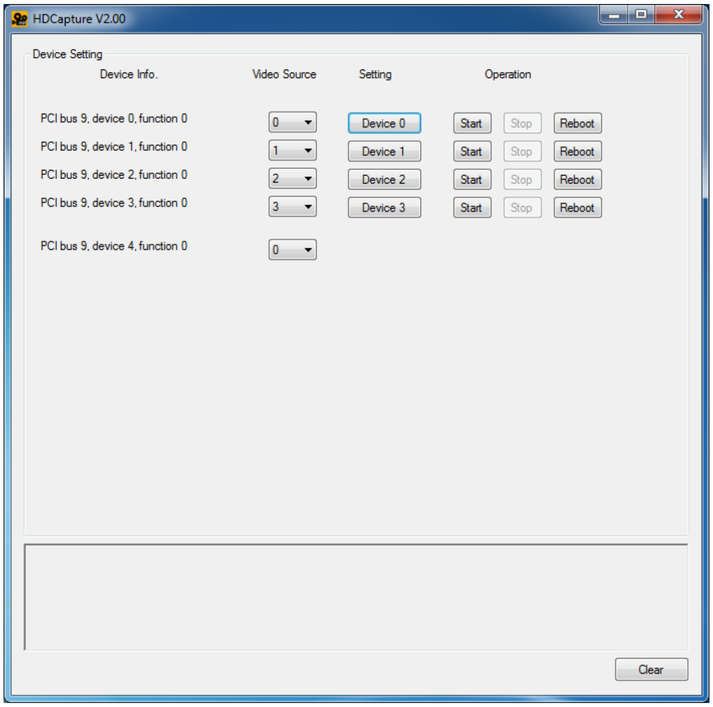
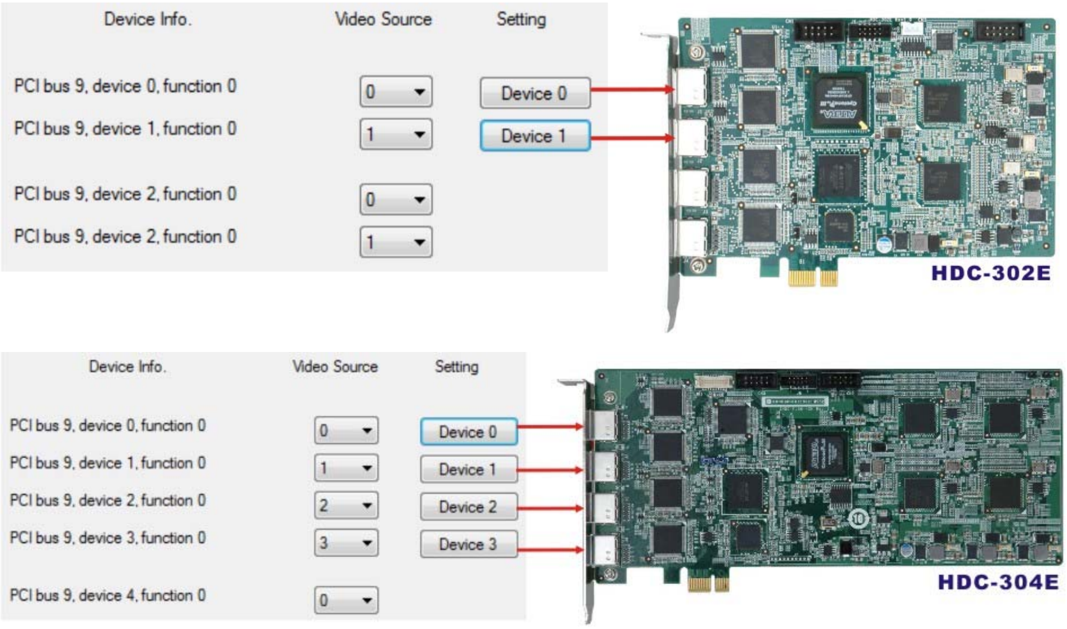

HDCapture V2.00
Device Setting
Device Info.
Video Source
Setting
Operation
PCI bus 9, device 0, function 0 | 0 | Device 0 | Start | Stop | Reboot
PCI bus 9, device 1, function 0 | 1 | Device 1 | Start | Stop | Reboot
PCI bus 9, device 2, function 0 | 2 | Device 2 | Start | Stop | Reboot
PCI bus 9, device 3, function 0 | 3 | Device 3 | Start | Stop | Reboot
PCI bus 9, device 4, function 0 | 0
Clear

Figure 2-1: HDCapture SDK

Step 2: Enable and configure the device settings by clicking the Device # (0, 1, 2, 3) buttons. The device number is decided by which port the device is installed. If the HDC-301 series is installed, there will be only one Device 0 button to choose.



Device Info.	Video Source	Setting
PCI bus 9, device 0, function 0	0	Device 0
PCI bus 9, device 1, function 0	1	Device 1
PCI bus 9, device 2, function 0	0	
PCI bus 9, device 2, function 0	1	

Device Info.	Video Source	Setting
PCI bus 9, device 0, function 0	0	Device 0
PCI bus 9, device 1, function 0	1	Device 1
PCI bus 9, device 2, function 0	2	Device 2
PCI bus 9, device 3, function 0	3	Device 3
PCI bus 9, device 4, function 0	0	

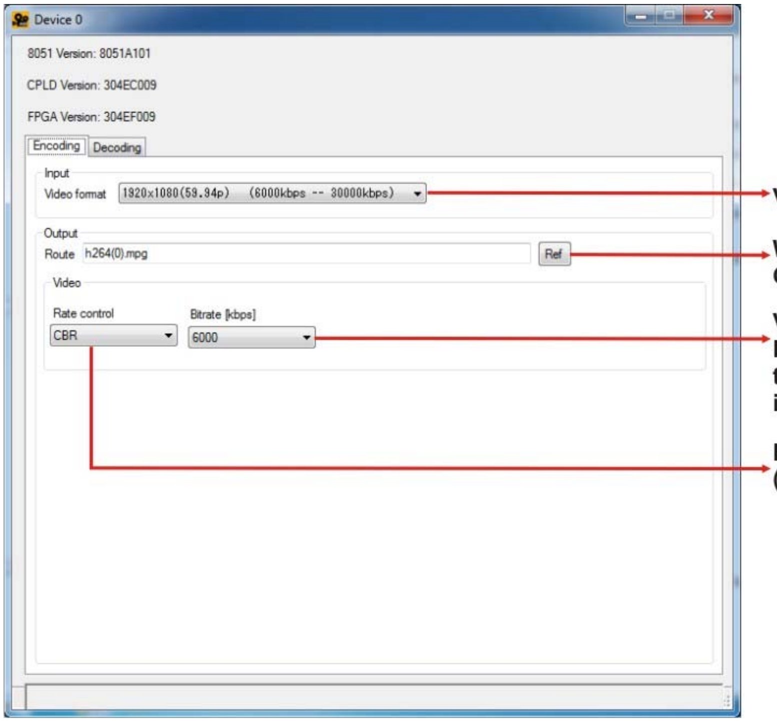
Figure 2-2: HDC-302E and HDC-304E Device Ports

Step 3: Click the Device # button. The Encoding window appears (**Figure 2-3**). Choose the video input format which depends on the video device. **The video format selected here must be same with the HDMI input video format.** The available options include:

1920x1080 (60p)	(6000kps – 30000kps)
1920x1080 (59.94p)	(6000kps – 30000kps)
1920x1080 (50p)	(6000kps – 30000kps)
1920x1080 (60i)	(6000kps – 24000kps)
1920x1080 (59.94i)	(6000kps – 24000kps)
1920x1080 (50i)	(6000kps – 24000kps)
1280x720 (60p)	(4000kps – 24000kps)
1280x720 (59.94p)	(4000kps – 24000kps)
1280x720 (50p)	(4000kps – 24000kps)

720x480 (60i)	(2000kps – 10000kps)
720x480 (59.94i)	(2000kps – 10000kps)
720x576 (50i)	(2000kps – 10000kps)

Step 4: Configure the encoding settings, including encoding file directory (click Ref button to choose the directory), rate control (CBR or VBR) and video encoding bitrate (must be in the range of video format). When "CBR" is selected, the "Bitrate" text box is displayed. When "VBR" is selected, the "Average bitrate" and "Peak bitrate" text boxes are displayed. Close the window to save the settings.



Device 0

8051 Version: 8051A101

CPLD Version: 304EC009

FPGA Version: 304EF009

Video format: 1920x1080(59.94p) (6000kbps -- 30000kbps)

Route: h264(0).mpg

Rate control: CBR

Bitrate [kbps]: 6000

Video Input Format

Where to Save the Output File

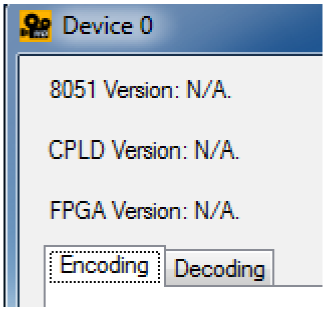
Video Output Bitrate (must be in the range of video input format

Rate Control (CBR or VBR)

Figure 2-3: Encoding Settings

NOTE:

If the HDC-301/301E series is installed, the version information shown in the encoding/decoding setting window will be N/A (as below).



Device 0

8051 Version: N/A.

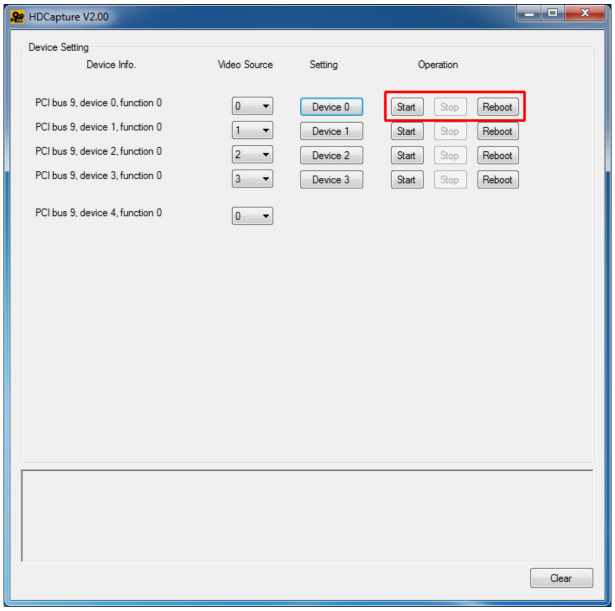
CPLD Version: N/A.

FPGA Version: N/A.

Step 5: Repeat **Step 2 ~ Step 4** to configure the connected input devices if necessary.

Step 6: Click Start Start to start capture the video. Click Stop Stop to stop capture.

Click Reboot to reboot the device.



HDCapture V2.00

Device Setting

Device Info.	Video Source	Setting	Operation
PCI bus 9, device 0, function 0	0	Device 0	Start Stop Reboot
PCI bus 9, device 1, function 0	1	Device 1	Start Stop Reboot
PCI bus 9, device 2, function 0	2	Device 2	Start Stop Reboot
PCI bus 9, device 3, function 0	3	Device 3	Start Stop Reboot
PCI bus 9, device 4, function 0	0		

Figure 2-4: Video Capture Control

2.3 Video Decoding

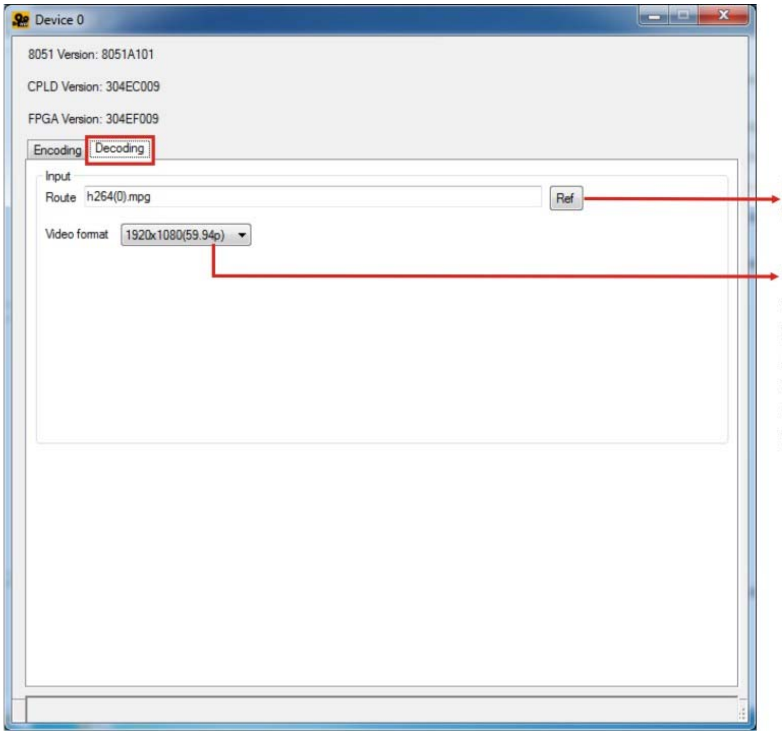
The HDCapture SDK also includes decoding function that decodes the video signal for video output to the HDMI-enabled display device. To decode a captured video clip, follow the steps below.

Step 1: Launch the HDCapture SDK.

Step 2: Bring up the Decoding page by clicking one of the Device # buttons. Click the Decoding tab to access the decoding page.

Step 3: Click Ref button Ref to locate a video file in the computer to decode.

Step 4: Select the video format of the selected video clip. The video format selected here must be the same with the video format of the file selected in the previous step.



8051 Version: 8051A101

CPLD Version: 304EC009

FPGA Version: 304EF009

Route: h264(0).mpg

Video format: 1920x1080(59.94p)

Locate a video file to decode

The video format of the video. (The video format selected here must be the same with the video format of the file selected in the field above)

Figure 2-5: Decoding Settings

Step 5: Close the window to save the settings.

Step 6: **This step is only required for the HDC-302E**. Select the transmitter bus (2 or 3) (**Figure 2-7**). For example, when the Device0 button is selected to decode the encoded file, please select 2 from the device2 (transmitter bus) drop-down menu to display the video on HDMI display device via video output port. Please refer to the following table:

Device Port	Device4 (Transmitter Bus) Setting
Device0	2
Device1	3

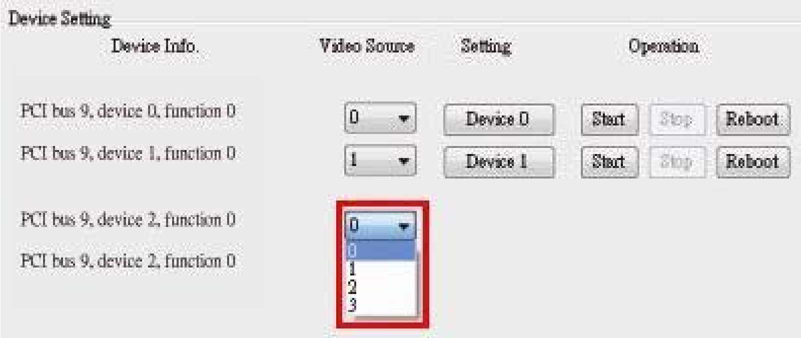

Device Setting
Device Info.
Video Source
Setting
Operation
PCI bus 9, device 0, function 0
PCI bus 9, device 1, function 0
PCI bus 9, device 2, function 0
PCI bus 9, device 2, function 0
0
1
Device 0
Device 1
Start
Stop
Reboot
Start
Stop
Reboot
0
0
1
2
3

Figure 2-6: Select Transmitter Bus for HDC-302E

Step 7: **This step is only required for the HDC-304E**. Select the transmitter bus from 4 to 7 (**Figure 2-7**). For example, when the Device0 button is selected to decode the encoded file, please select 4 from the device4 (transmitter bus) drop-down menu to display the video on HDMI display device by output kit. Please refer to the following table:

Device Port	Device4 (Transmitter Bus) Setting
Device0	4
Device1	5
Device2	6
Device3	7

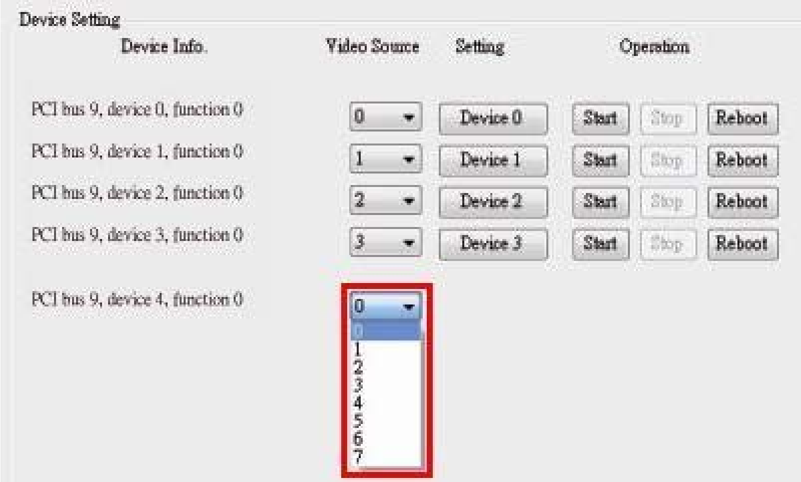


Figure 2-7: Select Transmitter Bus for HDC-304E

Step 8: Click Operation Start Start to start decoding the selected video.

2.4 Video Bypass

To view the video input source on HDMI display device in real time, please follow the steps below.

Step 1: Launch the HDCapture SDK.

Step 2: Connect the HDMI cable from HDMI display device to the HDMI output port or the HDMI output kit (HDC-304E).

Step 3: Connect the video input source to the HDMI input port of the HDC-3x Series SDK (Windows).

Step 4: **This step is only required for the HDC-302E**. Select the transmitter bus (0 or 1) (**Figure 2-9**). For example, to view the video input source 1 on HDMI display device, select the transmitter bus 0. Please refer to the following table:

Video Input Source	Transmitter Bus
Source 1	0
Source 2	1

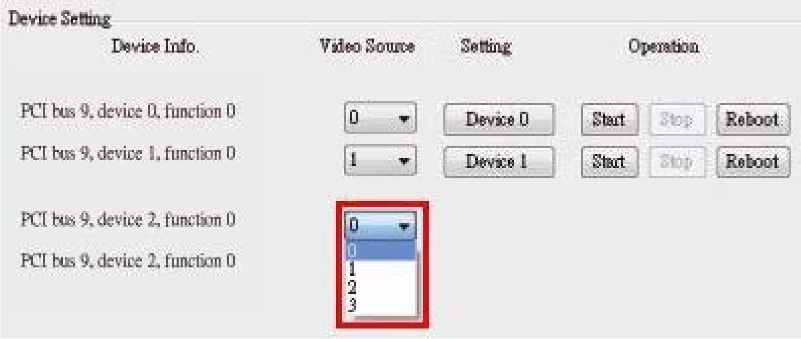

Device Setting
Device Info.
Video Source
Setting
Operation
PCI bus 9, device 0, function 0
PCI bus 9, device 1, function 0
PCI bus 9, device 2, function 0
PCI bus 9, device 2, function 0
0
1
Device 0
Device 1
Start
Stop
Reboot
Start
Stop
Reboot
0
0
1
2
3

Figure 2-8: Select Transmitter Device for HDC-302E

Step 5: **This step is only required for the HDC-304E**. Select the transmitter bus from 0 to 3 (**Figure 2-9**). For example, to view the video input source 1 on HDMI display device, select the transmitter bus 0. Please refer to the following table:

Video Input Source	Transmitter Bus
Source 1	0
Source 2	1
Source 3	2
Source 4	3

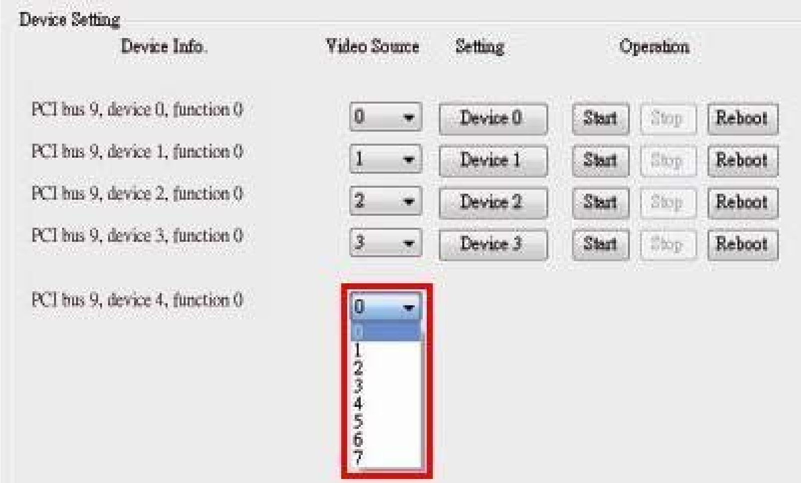


Figure 2-9: Select Transmitter Device for HDC-304E

NOTE:

For the HDC-304E users:

The video input source 1 is set to default for video bypass. Thus, when turn on the computer, video bypass will be enabled and the video from video input 1 will show on the HDMI display, if

1. the HDMI output kit from the HDC-3x Series SDK (Windows) is connecting to HDMI display, and

2. the video input source is connecting to the video input 1 connector of the HDC-3x Series SDK (Windows).

Chapter

3

API Introduction

3.1 Build Environment

The API build environment requirements are listed below. If build environment is not Microsoft Visual Studio 2005 SP1 or latter, you need to install Microsoft Visual C++ 2005 SP1 Redistributable Package (x86).

- Microsoft Windows XP SP2 32-bit
- Microsoft Windows 7 32-bit/64-bit
- DirectX SDK – August 2007
- Windows SDK for Windows Vista (6.0.6000)
- Microsoft .NET Framework 2.0/3.0/3.5/4.0 32-bit/64-bit
- Microsoft Visual Studio 2005 SP1

NOTE:

The DumpFile.dll and PushFileSource2.dll are filters of DirectShow.

You must register them before using them. Otherwise, you will get an error.

3.2 API Introduction

NOTE:

If API usage in document is different from API usage in SDK source code, the API usage in SDK source code is CORRECT.

3.2.1 DeviceMan API Introduction

There are one enum, one structure and two functions in DeviceMan.dll. The source codes are listed below for reference.

```
typedef struct _CardList_T
{
        // Card category.
        int   iCategory;
        // UI No, usually is the slot No.
        int   iUINo;
        // Bus No.
        int   iBusNo;
        // Device number.
        int   iDeviceNum;
        // Transmitter number.
        int   iTransmitterNum;
        // Device No of each device.
        int   iDeviceNo[4];
        // Device information of each device.
        char cDeviceInfo[4 * MAX_BUFFER_SIZE];
        // Transmitter information of each device.
        char cTransmitterInfo[4 * MAX_BUFFER_SIZE];
} CardList_T;
```

and the MAX_BUFFER_SIZE is 512.

```
enum
{
        DEVICE_MAN_RESULT_SUCCESS = 0,
        DEVICE_MAN_RESULT_NULL_ADDRESS,
        // ASCII to Unicode failed.
        DEVICE_MAN_RESULT_ATOU_FAILED,
        // Unicode to ASCII failed.
        DEVICE_MAN_RESULT_UTOA_FAILED,
        DEVICE_MAN_RESULT_INVALID_HANDLE,
        DEVICE_MAN_RESULT_BUF_ERR_MAXIMUM,
        DEVICE_MAN_RESULT_BUF_ERR_LENGTH,
        DEVICE_MAN_RESULT_BUF_ERR_OVER_MAX,
        // Input parameter error.
        DEVICE_MAN_RESULT_PARAMETER_ERROR,
        // Memory allocate failed.
        DEVICE_MAN_RESULT_MEM_ALLOC_FAILED,
        // No capture card.
        DEVICE_MAN_RESULT_NO_CARD,
        // Get UI No. failed.
        DEVICE_MAN_RESULT_GET_UI_NO_FAILED,
        // Get bus No. failed.
        DEVICE_MAN_RESULT_GET_BUS_NO_FAILED,
        // Get information failed.
        DEVICE_MAN_RESULT_GET_INFO_FAILED,
        // CPLD check failed.
        DEVICE_MAN_RESULT_CPLD_FAILED,
        DEVICE_MAN_RESULT_UNKNOWN_ERROR
};
```

1. DeviceManGetVersion(int* ot_ipVerYear,int* ot_ipVerMonth,int* ot_ipVerDay)

   Description: Get DeviceMan.dll verion.

   Parameter:

   ot_ipVerYear : Integer pointer of year version.

   ot_ipVerMonth: Integer pointer of month version.

   ot_ipVerDay: Integer pointer of day version.

   Return:

   An integer, see enum type.

2. DeviceManGetCardList(int* ot_ipCardNum, void** ot_ppCardList)

   Description:

   Get capture card list.

Parameter:

ot_ipCardNum: Integer pointer of card number.

ot_ppCardList: Void pointer of card list.

Return:

An integer, see enum type.

3.2.2 CPLDMan API Introduction

The CPLDMan.dll is the same with the DeviceMan.dll. The detail usage can be found in the source code.

```
enum
{
        CPLD_RESULT_SUCCESS = 0,
        CPLD_RESULT_MEM_ALLOC_FAILED,
        CPLD_RESULT_LIB_INITIALIZED,
        CPLD_RESULT_LIB_UNINITIALIZED,
        CPLD_RESULT_LIB_INITIALIZE_FAILED,
        CPLD_RESULT_LIB_UNINITIALIZE_FAILED,
        CPLD_RESULT_OPENED_NUMBER_OVER,
        CPLD_RESULT_OPEN_FAILED,
        CPLD_RESULT_INVALID_CERTIFICATE,
        CPLD_RESULT_INVALID_PARAMETER,
        CPLD_RESULT_VIDEO_SOURCE_GET_FAILED,
        CPLD_RESULT_VIDEO_SOURCE_SET_FAILED,
        CPLD_RESULT_VIDEO_RESOLUTION_NO_OUTPUT,
        CPLD_RESULT_VIDEO_RESOLUTION_NO_HDMI,
        CPLD_RESULT_VIDEO_RESOLUTION_INVALID,
        CPLD_RESULT_VIDEO_RESOLUTION_GET_FAILED,
        CPLD_RESULT_VERSION_8051_GET_FAILED,
        CPLD_RESULT_VERSION_CPLD_GET_FAILED,
        CPLD_RESULT_VERSION_FPGA_GET_FAILED
};
```

1. CPLDManGetVersion(int* ot_ipVerYear,int* ot_ipVerMonth, int* ot_ipVerDay)

   Description:

   Get CPLDMan.dll version.

   Parameter:

   ot_ipVerYear : Integer pointer of year version.

   ot_ipVerMonth: Integer pointer of month version.

   ot_ipVerDay: Integer pointer of day version.

   Return:

   An integer, see enum type.

2. CPLDManInitialize();

   Description:

   Initialize CPLD library.

   Parameter:

   N/A.

   Return:

   An integer, see enum type.

3. CPLDManUninitialize();

   Description:

   Uninitialize CPLD library.

   Parameter:

   N/A.

   Return:

   An integer, see enum type.

4. CPLDManOpen(int in_iBusNo)

   Description:

   Open CPLD.

   Parameter:

   in_iBusNo: Bus No. of CPLD.

   Return:

   An integer, see enum type.

5. CPLDManClose(int in_iBusNo)

Description:

Close CPLD.

Parameter:

in_iBusNo: Bus No. of CPLD.

Return:

An integer, see enum type.

6. CPLDManCodecVideoSrcGet(int in_iBusNo, int in_iCodecNo, int* ot_ipValue)

Description:

Get video source of codec.

Parameter:

iBusNo: Bus No. of CPLD.

in_ iCodecNo: Codec No.

ot_ipValue: Integer pointer of video source, used in get funcion.

Return:

An integer, see enum type.

7. CPLDManCodecVideoSrcSet(int in_iBusNo, int in_iCodecNo, int in_iValue)

Description:

Set video source of codec.

Parameter:

iBusNo: Bus No. of CPLD.

in_ iCodecNo: Codec No.

in_iValue: Video source, used in set function.

Return:

An integer, see enum type.

8. CPLDManTXVideoSrcGet(int in_iBusNo, int in_iTXNo, int* ot_ipValue)

Description:

Get video source of transmitter.

Parameter:

iBusNo: Bus No. of CPLD.

in_ iTXNo: Transmitter No.

ot_ipValue: Integer pointer of video source, used in get funcion.

Return:

An integer, see enum type.

9. CPLDManTXVideoSrcSet(int in_iBusNo, int in_iTXNo, int in_iValue)

   Description:

   Set video source of transmitter.

   Parameter:

   iBusNo: Bus No. of CPLD.

   in_ iTXNo: Transmitter No.

   in_iValue: Video source, used in set function.

   Return:

   An integer, see enum type.

10. CPLDMan8051Version(int in_iBusNo, int* ot_ipValue);

11. CPLDManCPLDVersion(int in_iBusNo, int* ot_ipValue);

12. CPLDManFPGAVersion(int in_iBusNo, int* ot_ipValue);

    Description:

    Get firmware version of 8051 / CPLD / FPGA.

    Parameter:

    in_iBusNo: Bus No. of CPLD.

    ot_ipValue: Integer pointer of firmware version.

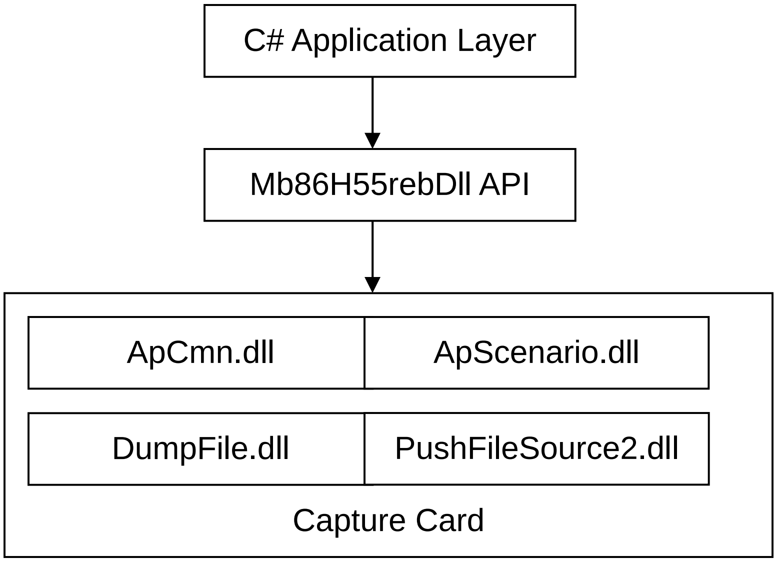
    Return:

    An integer, see enum type.

3.2.3 Mb86H55rebDll API Introduction

The Mb86H55rebDll API only has C# version now. The detail usage can be found in the source code.

3.2.4 Role of Mb86H55rebDll API



The application can use Mb86H55rebDll API to control capture card.

3.2.5 Using Mb86H55rebDll API

Step 1: Put the “ApCmn.dll”, “ApScenario.dll”, “DumpFile.dll”, “Mb86H55rebDll.dll” and “PushFileSource2.dll” in the folder where execution file exist.

NOTE:

The DumpFile.dll and PushFileSource2.dll are filters of DirectShow. The user must register them before using them, otherwise an error will occur.

Step 2: Use name space:

```
using Mb86H55rebDll;
```

Step 3: Declare variable to control MB86H55 as below:

```
Mb86H55reb mb86h55reb = new Mb86H55reb;
```

Step 4: Add the following event handler:

```
protected override void WndProc(ref Message m)
{
    DoMb86h55Events(ref m);
    base.WndProc(ref m);
}
private void DoMb86h55Events(ref Message m)
{
    Mb86H55reb.AsyncEventResult result;
    string comment;
    result = mb86h55reb.OnMsg(ref m, out comment);
    UpdateScreenAfterEvents(result, comment);
}
```

Step 5: In the function UpdateScreenAfterEvents(),other control functions can be added according to the purpose. For example: Error message report function.

```
private void UpdateScreenAfterEvents(Mb86H55reb.AsyncEventResult result,
string comment)
{
    switch (result)
    {
    case Mb86H55reb.AsyncEventResult.OperationComplete:
        break;
    case Mb86H55reb.AsyncEventResult.OperationCompleteStop:
        mb86h55reb.Reset();
        break;
    case Mb86H55reb.AsyncEventResult.OperationCompleteAutoStop:
        mb86h55reb.Reset();
        break;
    case Mb86H55reb.AsyncEventResult.OperationCancel:
        break;
    case Mb86H55reb.AsyncEventResult.Warning:
        break;
    case Mb86H55reb.AsyncEventResult.SeriousError:
        break;
    case Mb86H55reb.AsyncEventResult.HdmiCableStatusChanged:
        break;
    case Mb86H55reb.AsyncEventResult.OperationContinue:
        break;
    case Mb86H55reb.AsyncEventResult.AudioStatusChanged:
        break;
    default:
        break;
    }
}
void SystemEvents_PowerModeChanged(object sender,
Microsoft.Win32.PowerModeChangedEventArgs e)
{
switch (e.Mode)
    {
case Microsoft.Win32.PowerModes.Suspend:
        mb86h55reb.Close();
        break;
case Microsoft.Win32.PowerModes.Resume:
mb86h55reb.DirectShowEnabled(miChipNo, mbDirectShowEnabled);
mbIsMb86h55rebOpened = mb86h55reb.Open(miChipNo,this.Handle);
```

```
mb86h55reb.SetCanvasHandle(mPnlCanvas.Handle);
mb86h55reb.ApplyGpio();
mb86h55reb.RebootFirm();
SetScreenMode(ScreenMode.Processing);
mb86h55reb.Reset();
break;
    }
}
```

Step 6: Before using MB86H55REB, it must be initialized:

```
mb86h55reb.Close();
mb86h55reb.DirectShowEnabled(miChipNo, mbDirectShowEnabled);
mbIsMb86h55rebOpened = mb86h55reb.Open(miChipNo,this.Handle);
mb86h55reb.SetCanvasHandle(mPnlCanvas.Handle);
mb86h55reb.ApplyGpio();
mb86h55reb.RebootFirm();
mb86h55reb.Reset();
```

Step 7: Refer the following function for detail:

```
frmMain_Load()
SystemEvents_PowerModeChanged()
cmbBoardSelection_SelectedIndexChanged()
```

3.2.6 Mb86H55rebDll API Description

Simplify description of Mb86H55rebDll variable, interface and API. Refer to the source code to get the detail usage.

Variable:

1. string h264FileName

Encode / decode file name.

Interface

1. FMBVideoFormatEnum h264VideoFormat

    Video formate.

```
      enum FMBVideoFormatEnum
      {
FMBEnmVideoFmt1920x1080,
      FMBEnmVideoFmt1440x1080,
      FMBEnmVideoFmt1280x720,
      FMBEnmVideoFmt720x480,
      FMBEnmVideoFmt720x576,
      EnmVideoNumofFmt
};
```

2. FMBVideoFrameEnum h264VideoFrame

    Video frame rate.

```
      enum FMBVideoFrameEnum
      {
FMBEnmVideoFrm_60p,
      FMBEnmVideoFrm_5994p,
      FMBEnmVideoFrm_50p,
      FMBEnmVideoFrm_60i,
      FMBEnmVideoFrm_5994i,
      FMBEnmVideoFrm_50i,
      EnmVideoNumofFrm
};
```

3. FMBVideoRateCtlEnum h264VideoRateCtl

    Video rate control.

```
      enum FMBVideoRateCtlEnum
      {
FMBEnmVideoRateCtlCbr,
FMBEnmVideoRateCtlVbr,
};
```

4. int h264VideoBitrateCbr

   Video CBR bitrate value.

5. int h264VideoBitrateAverage

   Video average bitrate for VBR.

6. int h264VideoBitratePeak

   Video peak bitrate for VBR.

7. int[] h264Pids = new int[(int)PidTypeEnum.EnmPidNumofPid];

   PID value array.

```
enum PidTypeEnum
{
      EnmPidVideo,
      EnmPidAudio,
      EnmPidPmt,
      EnmPidSit,
      EnmPidPcr,
      EnmPidNumofPid
};
```

8. FMBFuncModeEnum operationMode

   Operation mode.

```
enum FMBFuncModeEnum
{
   FMBEnmFuncModeEnc,
   FMBEnmFuncModeDec,
};
```

9. int pciNo

   Get current PCI / chip No.

10. bool isStreamRunning

    Get is stream runnging.

API

1. bool Open(int pciNoArg, IntPtr hWnd)

    Description:

    Open device.

    Parameter:

    pciNoArg: Device (chip) No.

    hWnd: Window handle.

2. void Close()

    Description:

    Close device.

3. void Encode()

    Description:

    The encode is begun.

4. void Decode()

    Description:

    The decode is begun.

5. void Stop()

    Description:

    The stop is begun.

6. void Reset()

    Description:

    The reset is begun.

7. AsyncEventResult OnMsg(ref Message m, out string comment)

    Description:

It is processed to receive the message.

Parameter:

m: Value of message

comment: Comment form me

Return:

Value of AsyncEventResult

```
public enum AsyncEventResult
{
UnknownEvent,
OperationContinue,
OperationComplete,
OperationCompleteStop,
OperationCompleteAutoStop,
OperationCancel,
Warning,
SeriousError,
HdmiCableStatusChanged,
AudioStatusChanged,
}
```

8. bool Equals(ref Mb86H55reb target)

Description:

Oneself is compared with the argument.

Parameter:

target: target

Return:

true:equal, false:not equal.

9. void CommitProperty()

Description:

The change in property is committed.

10. void ApplyGpio()

Description:

Property is applied to the GPIO device.

11. void RebootFirm()

    Description:

    Firm is rebooted.

12. void SetChipNo(int in_iChipNo)

    Description:

    Set device (chip) No.

    This function will change the chip ID, use it be carefully.

    Parameter:

    in_iChipID: Chip ID.

    in_iBusNumber: Bus No.

    in_iDevNumber: Device No.

13. void DirectShowEnabled(int in_iChipNo, bool in_bFlag)

    Description:

    Enable / disable DirectShow.

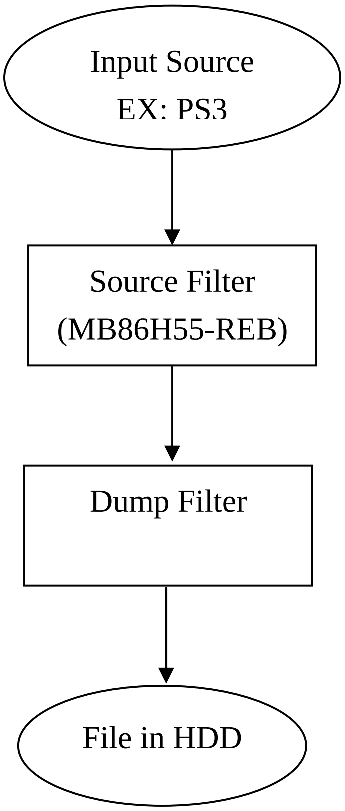
    Parameter:

    in_iChipNo: Chip No.

    in_bFlag: true is enabled, false is disabled.

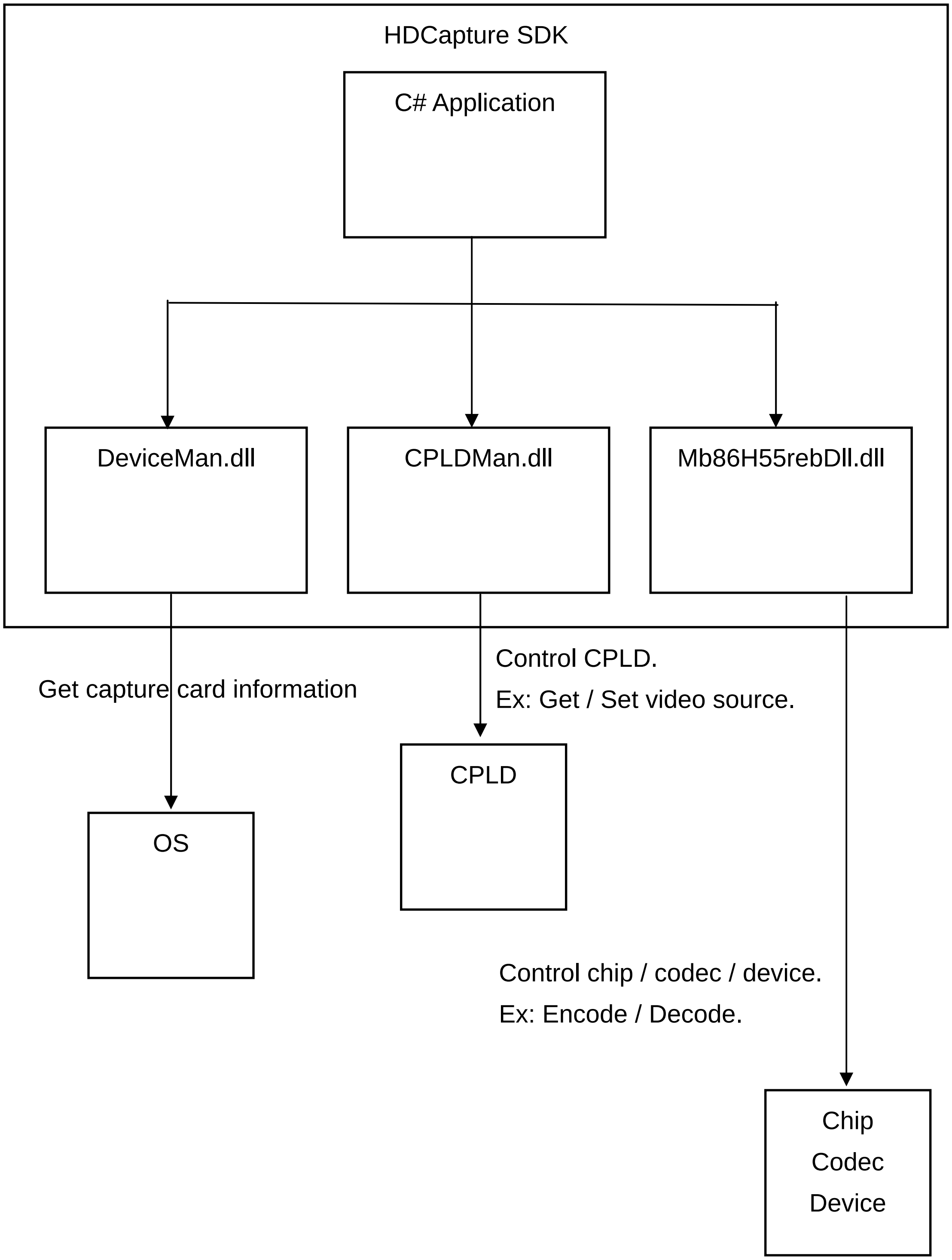
3.3 DirectShow Graph

3.3.1 Encoding Graph



3.4 Architecture of SDK

Chip / codec / device usually means the same thing.



Chapter

4

FAQ

Q: Capture card, driver, application and input source are ready, but the recorded video is not displayed or displayed incorrectly.

A: In this situation, a correct video codec may be needed to display the recorded video -- the ffdshow codec for example.

Q: How do I check the current DirectX version?

A: In Windows, navigate to Start → Run → Type 'dxdiag' → Enter. The current version is displayed in the DirectX Diagnostic Tool window.

Q: Input source and encoding are both set to 1080 60p, but the application cannot encode.

A: The chip is critical for input stream timing. Make sure the input stream frequency is 1080 60p.

Q: How do I obtain the latest driver?

A: Go to http://www.ieiworld.com/. You can always find and download the latest drivers from the "Support" pages.

Appendix A

Error Code

A.1 Error Code Overview

Error register (M_ERROR_INFO_H and M_ERROR_INFO_L)

Name	M_ERROR_INFO_H		M_ERROR_INFO_L
Bit	15……..…..…..8 7……………..0		15…………………………...0
Field	ERROR_ MODULE[7:0]	ERROR_STATUS[23:0]	

A.2 ERROR_MODULE[7:0]

Module where the error occurred.

The table below outlines the relationship between values and modules.

Value	Module Name	Function	Mode
0x00	HOSTCMD	Host communication library	ENC, DEC
0x01	ETOP	Recorder-wide controller	ENC
0x02	DTOP	Player-wide controller	DEC
0x03	BACKGROUND	Interrupt controller	ENC, DEC
0x04	VREC	Video input controller	ENC
0x05	VMUX	Video multiplex controller	ENC
0x06	AREC	Audio input controller	ENC
0x07	AMUX	Audio multiplex controller	ENC
0x08	SMUX	System multiplex controller	ENC
0x09	VPLAY	Video playback controller	DEC
0x0A	VDMX	Video decode controller	DEC
0x0B	APLAY	Audio playback controller	DEC
0x0C	ADMX	Audio decode controller	DEC
0x0D	SDMX	System stream controller	DEC
0x0E	SAPI	Serial communication controller	ENC, DEC

A.3 ERROR_STATUS[23:0]

Detailed error cause. The relationship between the values and error causes is described in the following sections.

A.3.1 IDLE

ERROR_MODULE[7:0]		ERROR_MODULE[23:0]		Outline
Value	Name	Value	Name	
0x00	HOSTCMD	0x00_0001	INVALID_SCMD_CMD_ID	The system command parameter cmd_id is invalid
		0x00_0002	SCMD_CC_NOT_CONTINUOUS	The system command parameter continuity_counter values are not consecutive
		0x00_0003	HOSTCMD_ERR_SCMD_ UNACCEPTABLE_FIRMWARE	Incorrect firmware

A.3.2 ENC

ERROR_MODULE[7:0]		ERROR_MODULE[23:0]		Outline
Value	Name	Value	Name	
0x00	HOSTCMD	0x00_0001	INVALID_SCMD_CMD_ID	The system command parameter cmd_id is invalid
		0x00_0002	SCMD_CC_NOT_CONTINUOUS	The system command parameter continuity_counter values are not consecutive
		0x00_0003	HOSTCMD_ERR_SCMD_ UNACCEPTABLE_FIRMWARE	Incorrect firmware
0x01	ETOP	0x00_0001	FIFO_OVERFLOW	The event queue overflowed
		0x00_0002	INVALID_SCMD_INIT_PARAM	An invalid value is specified in initialization dedicated parameter register M
		0x00_0003	INVALID_VCMD_INIT_PARAM	An invalid value is specified in initialization dedicated parameter register V
		0x00_0004	INVALID_ACMD_INIT_PARAM	An invalid value is specified in initialization dedicated parameter register A
		0x00_0005	UNACCEPTABLE_EVENT	An event that cannot be handled by the ETOP was received
		0x00_0006	INVALID_SCMD_CMD_ID	The system command parameter cmd_id is invalid
		0x00_0007	ANOTHER_SCMD_ BEFORE_SCMD_ACK	A subsequent system command was received before an acknowledge was returned
		0x00_0008	SCMD_CC_NOT_ CONTINUOUS	The system command parameter continuity_counter values are not consecutive
		0x00_0009	INVALID_SMES_ACK_CMD_ID	The system command parameter cmd_id is invalid

		0x00_000A	ANOTHER_SMES_ACK_ BEFORE_SMES	An acknowledge was received althought no subsequent system message was sent
		0x00_000B	SMES_ACK_CC_ NOT_CONTINUOUS	The system command parameter continuity_counter values are not consecutive
		0x00_000C	SMES_ACK_NOT_RECIEVED	Before reception of a system message acknowledge, the next message was generated
		0x00_000D	INNER_ERROR	ETOP internal error
		0x00_000E	EVENT_QUEUE_OVERFLOW	The event queue overflowed
0x03	BACKGROUND			
0x04	VREC	0x00_0001	INVALID_VCMD_INIT_PARAM	The value of the initialization dedicated parameter register V is invalid
		0x00_0002	VIDEO_CPU_ACCESS	There is a problem concerning communication with the video section
		0x00_0003	FIFO_OVERFLOW	Some idx_fifo overflowed
		0x00_0004	FIFO_EMPTY	Some idx_fifo became empty
		0x00_0005	UNACCEPTABLE_ HOSTCMD_EVENT	Invalid HOSTCMD event
		0x00_0006	UNACCEPTABLE_EVENT	Invalid event
		0x00_0007	INVALID_IDX	The VRAW _idx value became invalid
		0x00_0008	INNER_ERROR	VREC internal error
		0x00_0009	EVENT_QUEUE_OVERFLOW	The event queue overflowed
0x05	VMUX	0x00_0001	NG	Unclassified VMUX internal error
		0x00_0002	UNDERFLOW	A video stream buffer underflow was detected
		0x00_0003	VBV_BOC	VBV discontinuity was detected

		0x00_0004	STRM_BUF_OVERWRITTEN	A stream buffer overwrite was detected
		0x00_0005	INVALID_HOST_CMD	Invalid HOSTCMD event
		0x00_0006	INVALID_EVENT	Invalid event
		0x00_0007	FIFO_OVERFLOW	Some idx_fifo overflowed
		0x00_0008	FIFO_EMPTY	Some idx_fifo became empty
		0x00_0009	INVALID_MUXCMD	Invalid command to the multiplexing section
		0x00_000A	INVALID_PARAM	Invalid parameter
0x06	AREC	0x00_0001	INVALID_INIT_PARAM	Invalid initialization parameter value
		0x00_0002	AUDIO_CPU_ACCESS	There is a problem concerning communication with the audio section
		0x00_0003	AUDI_IN	An error occurred during audio input access
		0x00_0004	FIFO_OVERFLOW	Some idx_fifo overflowed
		0x00_0005	FIFO_EMPTY	Some idx_fifo became empty
		0x00_0006	UNACCEPTABLE_HOSTCMD_EVENT	Invalid HOSTCMD event
		0x00_0007	UNACCEPTABLE_EVENT	Invalid event
		0x00_0008	INVALID_IDX	The ARAW_idx value became invalid
		0x00_0009	INNER_ERROR	AREC internal error
		0x00_000A	EVENT_QUEUE_OVERFLOW	The event queue overflowed
0x07	AMUX	0x00_0001	NG	Unclassified AMUX internal error
		0x00_0002	INVALID_HOST_CMD	Invalid HOSTCMD event
		0x00_0003	INVALID_EVENT	Invalid event
		0x00_0004	FIFO_OVERFLOW	Some idx_fifo overflowed
		0x00_0005	FIFO_EMTPY	Some idx_fifo became empty
		0x00_0006	INVALID_MUXCMD	Invalid command to the multiplexing section

0x08	SMUX	0x00_0001	NG	Unclassified SUX internal error
		0x00_0002	INVALID_HOST_CMD	Invalid HOSTCMD event
		0x00_0003	INVALID_EVENT	Invalid event
		0x00_0004	INVALID_EVENT_SOURCE	Invalid event issuer
		0x00_0005	INVALID_EVENT_PARAM	Invalid event parameter
		0x00_0006	INVALID_PARAM	Invalid parameter
		0x00_0007	START_STC	STC start processing error
		0x00_0008	FIRST_PCR	FIRST_PCR processing error
		0x00_0009	FIRST_PAT	FIRST_PAT processing error
		0x00_000A	INVALID_ VMUX_AMUX_STATE	VMUX or AMUX state transition error
		0x00_000B	AUTO_NULL_ON	NULL output processing error
		0x00_000C	AUTO_NULL_OFF	NILL output stop processing error
		0x00_000D	STOP	Stop processing error
		0x00_000E	INVALID_MUXCMD	Invalid command to the multiplexing section
0x0E	SAPI	0x00_0001	OVERFLOW	A buffer overflow occurred
		0x00_0002	UNEXPECTED_ TRANS_DATA_IRQ	A data transmission completion interrupt was received when it should not have been
0xFF	(Special, tentative)	0x00_0001		Error notification from the video section (details are displayed in the error register V_ERROR_INFO)
		0x00_0002		Error notification from the audio section (details are displayed in the error register A_ERROR_INFO)

A.3.3 DEC

ERROR_MODULE[7:0]		ERROR_MODULE[23:0]		Outline
Value	Name	Value	Name	
0x00	HOSTCMD	0x00_0001	INVALID_SCMD_CMD_ID	The system command parameter cmd_id is invalid
		0x00_0002	SCMD_CC_NOT_CONTINUOUS	The system command parameter continuity_counter values are not consecutive
		0x00_0003	HOSTCMD_ERR_SCMD_UNACCEPTABLE_FIRMWARE	Incorrect firmware
0x02	DTOP	0x00_0001	FIFO_OVERFLOW	An FIFO overflow occurred
		0x00_0002	INVALID_SCMD_INIT_PARAM	The value specified in the initialization dedicated parameter register M is invalid
		0x00_0003	INVALID_VCMD_INIT_PARAM	The value specified in the initialization dedicated parameter register V is invalid
		0x00_0004	INVALID_ACMD_INIT_PARAM	The value specified in the initialization dedicated parameter register A is invalid
		0x00_0005	UNACCEPTABLE_STRM_INPUT_EVENT	An invalid stream input control event was received
		0x00_0006	UNACCEPTABLE_STATE_CHANGE_EVENT	An invalid state transition notification event was received
		0x00_0007	INVALID_SCMD_CMD_ID	The system command parameter cmd_id is invalid
		0x00_0008	INVALID_SCMD_SUB_CMD_ID	The system command parameter sub_cmd_id is invalid
		0x00_0009	ANOTHER_SCMD_BEFORE_SCMD_ACK	A subsequent system command was received before an acknowledge was returned

		0x00_000A	SCMD_CC_NOT_CONTINUOUS	The system command parameter continuity_counter values are not consecutive
		0x00_000B	INVALID_VIDEO_CPU_STATE	The state of the video section is invalid
		0x00_000C	INVALID_AUDIO_CPU_STATE	The state of the audio section is invalid
		0x00_000D	ERROR_NOTIFIED_ FROM_VIDEO_CPU	Error notification from the video section (details are displayed in the error register V_ERROR_INFO)
		0x00_000E	ERROR_NOTIFIED_ FROM_AUDIO_CPU	Error notification from the audio section (details are displayed in the error register A_ERROR_INFO)
		0x00_000F	INNER_ERROR	DTOP internal error
0x03	BACKGROUND	0x04_xxxx		An error occurred during processing of an interrupt from the AUDIO_SPDIF output
		0x05_xxxx		An error occurred during processing of an interrupt from the audio output
		0x09_xxxx		An error occurred during processing of an interrupt from the video output
		0x0D_xxxx		An error occurred during processing of an interrupt from the stream splitter
0x09	VPLAY	0x00_0001	FIFO_OVERFLOW	An FIFO overflow occurred
		0x00_0002	UNACCEPTABLE_ HOSTCMD_EVENT	An invalid state transition instruction event was received
		0x00_0003	INVALID_PARAM	Invalid argument

		0x00_0004	INVALID_INIT_PARAM	The initialization dedicated parameter register V is invalid
		0x00_0005	INVALID_VIDEO_OUT_STATE	The state of the video output hardware is invalid
		0x00_0006	INVALID_AUDIO_OUT_STATE	The state of the audio output hardware is invalid
		0x00_0007	INVALID_ VIDEO_OUT_STC_STATE	The STC state of the video output hardware is invalid
		0x00_0008	INVALID _AUDIO_OUT_STC+STATE	The STC state of the audio output hardware is invalid
		0x00_0009	INNER_ERROR	VPLAY module internal error
		0x00_000A	VIDOE_OUT_INNER_ERROR	The internal state of the video output hardware is invalid
		0x00_000B	AUDIO_OUT_INNER_ERROR	The internal state of the audio output hardware is invalid
		0x00_000C	UNIMPLEMENTED	No corrective measure has been implemented
		0x00_000D	NO_VALID_VIDEO_ES_INPUT	An urgent action was take because a valid video ES input delay was detected
0x0A	VDMX	0x00_0001	FIFO_OVERFLOW	An FIFO overflow occurred
		0x00_0002	INVALID_PARAM	Invalid argument
		0x00_0003	UNACCEPTABLE_ HOSTCMD_EVENT	An invalid HOSTCMD event was received
		0x00_0004	INVALID_VIDEO_CPU_STATE	The state of the video section is invalid
		0x00_0005	INNER_ERROR	VDMX module internal error
		0x00_0006	DEMUX_INNER_ERROR	Stream splitting hardware internal error
0x0B	APLAY	0x00_0001	FIFO_OVERFLOW	An FIFO overflow occurred

		0x00_0002	INVALID_INIT_PARAM	The initialization dedicated parameter register A is invalid
		0x00_0003	UNACCEPTABLE_HOSTCMD_EVENT	Invalid HOSTCMD event
		0x00_0004	UNACCEPTABLE_AUDIO_OUT_CONTROL_EVENT	Invalid AUDIO_OUTPUT_CONTROL event
		0x00_0005	UNACCEPTABLE_AUDIO_OUT_DONE_EVNET	Invalid AUDIO_OUTPUT_DONE event
		0x00_0006	INVALID_AUDIO_OUT_STATE	The state of the audio output hardware is invalid
		0x00_0007	INVALID_AUDIO_OUT_STC_STATE	The STC state of the audio output hardware is invalid
		0x00_0008	INVALID_MUTE_CONTROL	Invalid mute control was used
		0x00_0009	NEXT_AFRAME_IDX_CONFLICT	The specification of the next frame was repeated
		0x00_000A	INNER_ERROR	The internal state of the APLAY module is invalid
		0x00_000B	AUDIO_OUT_INNER_ERROR	The internal state of the audio output hardware is invalid
		0x00_000C	UNIMPLEMENTED	No corrective measure has been implemented
		0x00_000D	INVALID_AUDIO_OUT_SPDFI_STATE	The state of the audio output hardware (SPDIF) is invalid
		0x00_000E	AUDIO_OUT_SPDIF_INNER_ERROR	The internal state of the audio output hardware (SPDIF) is invalid
0x0C	ADMX	0x00_0001	FIFO_OVERFLOW	An FIFO overflow occurred
		0x00_0002	INVALID_PARAM	Invalid argument
		0x00_0003	UNACCEPTABLE_HOSTCMD_VENT	An invalid HOSTCMD event was received
		0x00_0004	INVALID_AUDIO_CPU_STATE	The state of the audio section is invalid

		0x00_0005	INNER_ERROR	VDMX module internal error
		0x00_0006	DEMUX_INNER_ERROR	Stream splitting hardware internal error
0x0D	SDMX	0x00_0001	FIFO_OVERFLOW	An FIFO overflow occurred
		0x00_0002	INVALID_PARAM	Invalid argument
		0x00_0003	INVALID_INIT_PARAM	The initialization dedicated parameter register M is invalid
		0x00_0004	INVALID_DEMUX_STATE	The state of the stream splitting hardware is invalid
		0x00_0005	SEQ_NUM_UNCNAHGED	seq_num is the same as the value previously specified
		0x00_0006	UNACCEPTABLE_ HOSTCMD_EVENT	An invalid HOSTCMD event was received
		0x00_0007	INNER_ERROR	SDMX module internal state error
		0x00_0008	UNACCEPTABLE_STRM_ INPUT_CONTROL_EVENT	An invalid STRM_ INPUT_CONTROL event was received
		0x00_0009	DEMUX_INNER_ERROR	The internal state of the stream splitting hardware is invalid
		0x00_000A	INVALID_SECTION_FROMAT	The PSI section is in an invalid format
0x0E	SAPI	0x00_0001	OVERFLOW	A buffer overflow occurred
		0x00_0002	UNEXPECTED_ TRANS_DATA_IRQ	A data transmission completion interrupt was received when it should not have been