DVP-7010AX

4 Cannel PCI-bus Video Capture Card
Copyright
This documentation and the software included with this product are copyrighted in 2004 by Advantech Co., Ltd. All rights are reserved. Advantech Co., Ltd. reserves the right to make improvements in the products described in this manual at any time without notice. No part of this manual may be reproduced, copied, translated or transmitted in any form or by any means without the prior written permission of Advantech Co., Ltd. Information provided in this manual is intended to be accurate and reliable. ADVANTECH CO., LTD. assumes no responsibility for its use, nor for any infringements of the rights of third parties which may result from its use.

Acknowledgments
IBM and PC are trademarks of International Business Machines Corporation. MS-DOS, Windows, Microsoft Visual C++ and Visual BASIC are trade-marks of Microsoft Corporation. Intel and Pentium are trademarks of Intel Corporation. Delphi and C++ Builder are trademarks of Inprise Corporation.

CE notification
The DVP-7010A, developed by ADVANTECH CO., LTD., has passed the CE test for environmental specifications when shielded cables are used for external wiring. We recommend the use of shielded cables. This kind of cable is available from Advantech. Please contact your local supplier for ordering information.

On-line Technical Support
For technical support and service, please visit our support website at: http://www.advantech.com/support
Contents

CHAPTER 1  GENERAL INFORMATION ........................................ 2

1.1 HARDWARE REQUIREMENT ............................................. 2
1.2 SOFTWARE REQUIREMENT ............................................. 2
1.3 BLOCK DIAGRAM ...................................................... 3
   Figure 1.1 System diagram ........................................... 3
1.4 PACKING LIST .......................................................... 3
1.5 DIMENSIONS ............................................................ 4
   Figure 1.2 Dimensions ................................................. 4
1.6 JUMPER/CONNECTOR LOCATION .................................. 4
   Figure 1.3 Jumper & connector location .......................... 4
1.7 CARD ID SELECTION & PIN DEFINITION ....................... 5
   1.7.1 Card ID selection(SW1) & LED indicator ............... 5
      Table 1.1 Card ID .................................................... 5
      Figure 1.4 LED indicator location ............................ 5
   1.7.2 GPIO: BH1 ......................................................... 6
      Table 1.2 GPIO pin definition ................................... 6
      Figure 1.5 GPIO(BH1) pin definition ......................... 6
   1.7.3 Auto wake-up & WDT reset function: J2 ................. 6
      Table 1.3 Auto wake-up & reset(J2) Pin definition ....... 7
      Figure 1.6 Auto wake-up & reset(J2) Pin def (1) .......... 7
      Figure 1.7 Auto wake-up & reset(J2) Pin def (2) .......... 7
   1.7.4 External trigger: J3 .............................................. 8
      Table 1.4 External trigger(J3)Pin definition ............... 8
      Figure 1.8 External trigger(J3)Pin def (1) ................. 8
      Figure 1.9 The description for active triggers of J3 ...... 8
   1.7.5 Standby power input : J4 ...................................... 9
      Table 1.5 Standby power input(J4)Pin definition .......... 9
      Figure 1.10 Standby power input(J4)Pin def (1) .......... 9
   1.7.6 Jumper: J7 ........................................................ 9
      Table 1.6 Jumper (J7)Pin definition .......................... 9
1.8 BATTERY ............................................................... 10
   Table 1.7 Battery Spec ............................................... 10
1.9 HARDWARE INSTALLATION .......................................... 10
1.10 SOFTWARE / DRIVER INSTALLATION .............................. 11
<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adv_DVPAPI_CreateSDKInstance</td>
<td>24</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetNumberOfDevices</td>
<td>25</td>
</tr>
<tr>
<td>Adv_DVPAPI_InitSDK</td>
<td>26</td>
</tr>
<tr>
<td>Adv_DVPAPI_CloseSDK</td>
<td>27</td>
</tr>
<tr>
<td>Adv_DVPAPI_Start</td>
<td>28</td>
</tr>
<tr>
<td>Adv_DVPAPI_Stop</td>
<td>29</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetCapState</td>
<td>30</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetCurFrameBuffer</td>
<td>31</td>
</tr>
<tr>
<td>Adv_DVPAPI_SetNewFrameCallback</td>
<td>32</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetVideoFormat</td>
<td>33</td>
</tr>
<tr>
<td>Adv_DVPAPI_SetVideoFormat</td>
<td>34</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetFrameRate</td>
<td>35</td>
</tr>
<tr>
<td>Adv_DVPAPI_SetFrameRate</td>
<td>36</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetVideoResolution</td>
<td>37</td>
</tr>
<tr>
<td>Adv_DVPAPI_SetVideoResolution</td>
<td>38</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetVideoInput</td>
<td>39</td>
</tr>
<tr>
<td>Adv_DVPAPI_SetVideoInput</td>
<td>40</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetBrightness</td>
<td>41</td>
</tr>
<tr>
<td>Adv_DVPAPI_SetBrightness</td>
<td>42</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetContrast</td>
<td>43</td>
</tr>
<tr>
<td>Adv_DVPAPI_SetContrast</td>
<td>44</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetHue</td>
<td>45</td>
</tr>
<tr>
<td>Adv_DVPAPI_SetHue</td>
<td>46</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetSaturation</td>
<td>47</td>
</tr>
<tr>
<td>Adv_DVPAPI_SetSaturation</td>
<td>48</td>
</tr>
<tr>
<td>Adv_DVPAPI_GPIOGetData</td>
<td>49</td>
</tr>
<tr>
<td>Adv_DVPAPI_GPIOSetData</td>
<td>50</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetWDTTimeout</td>
<td>51</td>
</tr>
<tr>
<td>Adv_DVPAPI_SetWDTTimeout</td>
<td>52</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetUCFlag</td>
<td>53</td>
</tr>
<tr>
<td>Adv_DVPAPI_SetUCFlag</td>
<td>54</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetPoweronEvent</td>
<td>55</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetAlarm</td>
<td>56</td>
</tr>
<tr>
<td>Adv_DVPAPI_SetAlarm</td>
<td>57</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetChecksum</td>
<td>58</td>
</tr>
<tr>
<td>Adv_DVPAPI_GetEEData</td>
<td>59</td>
</tr>
</tbody>
</table>
Adv_DVPAPI_SetEEData ....................................................... 60
Adv_DVPAPI_GetRTCData ................................................... 61
Adv_DVPAPI_SetRTCData .................................................... 62
CHAPTER 1

General Information
Chapter 1  General Information

DVP-7010AX is 4 channel video capture card by share-frame technology and captures up to D1 resolution with 30/25 fps frame rate totally.

DVP-7010AX allows up to 4 cards to be installed on one PC system by an onboard DIP switch setting and identifies card ID by LED indicators. DVP-7010AX supports NTSC/PAL composite video input through BNC connectors and digitizes the data to PC through PCI bus.

DVP-7010AX has an Auto wake-up function which provides 10 sets of indicated times to wake up system from an off state. This function can be also be triggered from an outer signal by 2 digital trigger pins. It also has a built-in Watch dog timer to reset the system when any unknown error or system crash occurs. DVP-7010AX is also designed with a programmable software protection key function for software copy protection.

1.1 Hardware Requirement

- Intel Pentium III 1GHz or above (CPU speed depends on video frame rate, channels and resolution)
- 256 MB RAM or above
- Free PCI slot(s)
- CD-ROM
- Hard disk with 1G free space

1.2 Software Requirement

- Microsoft Windows 2000/XP with DirectX 8.1 or above
1.3 Block Diagram

![Block Diagram](image)

**Figure 1.1 System diagram**

1.4 Packing List

- DVP-7010AX PCI capture card  \( \times 1 \)
- Utility CD (SDK, Manual, Datasheet) \( \times 1 \)
- Standby power cable (DVP-7010AX-S001) \( \times 1 \)
- Connection cable for WDT \( \times 1 \)
- Connection cable for power switch (DVP-7010AX-S001) \( \times 1 \)
1.5 Dimensions

Figure 1.2 Dimensions

1.6 Jumper/connector location

Figure 1.3 Jumper & connector location
1.7 Card ID selection & Pin definition

1.7.1 Card ID selection(SW1) & LED indicator

DVP-7010AX provides the probability for 16 cards on one board. The SW1 is for Card ID selection and LED indicator is for corresponding expression which as follows

<table>
<thead>
<tr>
<th>Card</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Card 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Card 2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Card 3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Card 4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Card 5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Card 6</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Card 7</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Card 8</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Card 9</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Card 10</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Card 11</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Card 12</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Card 13</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Card 14</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Card 15</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Card 16</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 1.1 Card ID

Figure 1.4 LED indicator location
1.7.2 GPIO: BH1

- 8 bit TTL/CMOS level Digital I/O.

<table>
<thead>
<tr>
<th>Pin no.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin 1</td>
<td>GPIO Pin 0</td>
</tr>
<tr>
<td>Pin 2</td>
<td>GPIO Pin 4</td>
</tr>
<tr>
<td>Pin 3</td>
<td>GPIO Pin 1</td>
</tr>
<tr>
<td>Pin 4</td>
<td>GPIO Pin 5</td>
</tr>
<tr>
<td>Pin 5</td>
<td>GPIO Pin 2</td>
</tr>
<tr>
<td>Pin 6</td>
<td>GPIO Pin 6</td>
</tr>
<tr>
<td>Pin 7</td>
<td>GPIO Pin 3</td>
</tr>
<tr>
<td>Pin 8</td>
<td>GPIO Pin 7</td>
</tr>
<tr>
<td>Pin 9</td>
<td>GND</td>
</tr>
<tr>
<td>Pin 10</td>
<td>GND</td>
</tr>
</tbody>
</table>

*Table 1.2 GPIO pin definition*

![Figure 1.5 GPIO(BH1) pin definition](image)

1.7.3 Auto wake-up & WDT reset function: J2

Auto wake-up and WDT reset function are set by J2 with 8 pins. The pins of 5,6,7,8 are for auto wake-up function (DVP-7010AX) and 1,2,3,4 are for WDT function. Its definitions are shown in Table 1.3, Figure 1.6 and 1.7. Auto wake-up function needs ATX motherboard and power.
<table>
<thead>
<tr>
<th>Pin no.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin 1</td>
<td>GND</td>
</tr>
<tr>
<td>Pin 2</td>
<td>GND</td>
</tr>
<tr>
<td>Pin 3</td>
<td>Reset Pin on Motherboard</td>
</tr>
<tr>
<td>Pin 4</td>
<td>Reset Pin on Panel</td>
</tr>
<tr>
<td>Pin 5</td>
<td>GND</td>
</tr>
<tr>
<td>Pin 6</td>
<td>GND</td>
</tr>
<tr>
<td>Pin 7</td>
<td>Power SW on Motherboard</td>
</tr>
<tr>
<td>Pin 8</td>
<td>Power SW on Panel</td>
</tr>
</tbody>
</table>

Table 1.3 Auto wake-up & reset(J2) Pin definition

Figure 1.6 Auto wake-up & reset(J2) Pin def (1)

Figure 1.7 Auto wake-up & reset(J2) Pin def (2)
1.7.4 External trigger: J3

DVP-7010A provides 2 pins to receive outer triggers to wake up system. These external triggers are set by J3 and its pin definitions are shown in Table 1.4 and Figure 1.8. The outer triggers can be sent through pin1 or 2 and the signals must be low-actived. (Figure 1.9)

<table>
<thead>
<tr>
<th>Pin no.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin 1</td>
<td>Trigger Pin 0</td>
</tr>
<tr>
<td>Pin 2</td>
<td>Trigger Pin 1</td>
</tr>
<tr>
<td>Pin 3</td>
<td>GND</td>
</tr>
</tbody>
</table>

*Table 1.4 External trigger(J3) Pin definition*

*Figure 1.8 External trigger(J3) Pin def (1)*

*Figure 1.9 The description for active triggers of J3*
1.7.5 Standby power input : J4

This pin should be connected to 5V standby power when user wants to enable the auto wake-up function.

<table>
<thead>
<tr>
<th>Pin no.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin 1</td>
<td>5V/100mA standby voltage</td>
</tr>
<tr>
<td>Pin 2</td>
<td>GND</td>
</tr>
<tr>
<td>Pin 3</td>
<td>Not connected</td>
</tr>
</tbody>
</table>

*Table 1.5 Standby power input(J4) Pin definition*

![Figure 1.10 Standby power input(J4) Pin def (1)](image)

1.7.6 Jumper: J7

Auto wake-up function need to set up the Jumper which is as follow table:

<table>
<thead>
<tr>
<th>Jumper (J7) pin definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin status.</td>
</tr>
<tr>
<td>Pin1-Pin2</td>
</tr>
<tr>
<td>Pin2-Pin3</td>
</tr>
</tbody>
</table>

*Table 1.6 Jumper (J7) Pin definition*
1.8 Battery

DVP-7010AX has a lithium battery for the Real Time Clock (RTC) function to keep the time running precisely. This battery is wildly used and its specifications are as follows;

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model name</td>
<td>CR2032</td>
</tr>
<tr>
<td>Normal Voltage</td>
<td>3.0V</td>
</tr>
<tr>
<td>Standard capacity</td>
<td>220 mA (on continuous discharge at 20°C under 15K ( \Omega ) load to 2.5V end-voltage)</td>
</tr>
<tr>
<td>Standard weight</td>
<td>3.1g</td>
</tr>
<tr>
<td>Terminals</td>
<td>Materials of Positive electrode: Nickel-plated stainless steel</td>
</tr>
<tr>
<td></td>
<td>Materials of Negative electrode: Nickel-plated stainless steel</td>
</tr>
</tbody>
</table>

*Table 1.7 Battery Spec.*

1.9 Hardware Installation

1. Turn off your computer and unplug the power cord.
2. Remove the cover of your computer.
3. Touch the metal part on the surface of your computer to neutralize the static electricity that might be on your body.
4. Setting the SW1 (Card ID) while you want.
5. Place the DVP-7010AX into Mother Board PCI slot.
6. Connect appropriate accessories (Video cable to camera. if necessary) to the DVP-7010AX.
7. Replace the cover of your computer chassis.
8. Plug in the power cord and turn on the computer.

*Note: Keep the anti-static bag for future use. You might need the original bag to store the card if you have to remove the card from the PC or transport it elsewhere.*
1.10 **Software / Driver Installation**

**Before you begin**

To facilitate the installation of the enhanced display device drivers and utility software, you should read the instructions in this chapter carefully before you attempt installation. The device drivers for the DVP-7010AX board are located on the software installation CD. The auto-run function of the driver CD will guide and link you to the utilities and device drivers under Windows system. Before you begin, it is important to note that most display drivers need to have the relevant software application already installed in the system prior to installing the enhanced display drivers. In addition, many of the installation procedures assume that you are familiar with both the relevant software applications and operating system commands. Review the relevant operating system commands and the pertinent sections of your application software user’s manual before performing the installation.
Installing
1. Insert the driver CD into your system's CD-ROM drive. In a few seconds, the software installation main menu appears. Move the mouse cursor over the "Manual" button under the "SETUP" heading, a message pops up telling you to start the installation.
2. Please read the following License Agreement. Press the PAGE DOWN key to see the rest of the agreement and Click "Yes" to continue the installation.
3. Choose destination location on your system disc then click "Next" when you see the following message.
4. Click "Next" when you see the following message.
5. Please fill in the information and Click "Next".
6. When the following message appears, click "Finish" to complete the installation and restart Windows or your computer.
CHAPTER 2

Functions Library
Chapter 2  DVP-7010AX Functions

Library

SDK Initialize and close
   Adv_DVPAPI_CreateSDKInstence
   Adv_DVPAPI_GetNoOfDevices
   Adv_DVPAPI_InitSDK
   Adv_DVPAPI_CloseSDK

Capture control
   Adv_DVPAPI_Start
   Adv_DVPAPI_Stop
   Adv_DVPAPI_GetCapState
   Adv_DVPAPI_SetNewFrameCallback
   Adv_DVPAPI_GetCurFrameBuffer

Capture setting
   Adv_DVPAPI_GetVideoFormat
   Adv_DVPAPI_SetVideoFormat
   Adv_DVPAPI_GetFrameRate
   Adv_DVPAPI_SetFrameRate
   Adv_DVPAPI_GetResolution
   Adv_DVPAPI_SetResolution
   Adv_DVPAPI_GetVideoInput
   Adv_DVPAPI_SetVideoInput

Sensor Control
   Adv_DVPAPI_GetBrightness
   Adv_DVPAPI_SetBrightness
   Adv_DVPAPI_GetContrast
   Adv_DVPAPI_SetContrast
   Adv_DVPAPI_GetHue
   Adv_DVPAPI_SetHue
   Adv_DVPAPI_GetSaturation
   Adv_DVPAPI_SetSaturation
GPIO
   Adv_DVPAPI_GPIOGetData
   Adv_DVPAPI_GPIOSetData

Micro Controller
   Adv_DVPAPI_GetWDDTTimeout
   Adv_DVPAPI_SetWDDTTimeout
   Adv_DVPAPI_GetUCFlag
   Adv_DVPAPI_SetUCFlag
   Adv_DVPAPI_GetPoweronEvent
   Adv_DVPAPI_GetAlarm
   Adv_DVPAPI_SetAlarm
   Adv_DVPAPI_GetChecksum
   Adv_DVPAPI_GetEEData
   Adv_DVPAPI_SetEEData
   Adv_DVPAPI_GetRTCData
   Adv_DVPAPI_SetRTCData
2.1 Functions Reference

**Struct**

**TimeStruct**

typedef struct{
    BYTE second;
    BYTE minute;
    BYTE hour;
    BYTE day;
    BYTE date;
    BYTE month;
    BYTE year;
} TimeStruct;

**Description**

A struct stores time setting.

**AlarmStruct**

typedef struct{
    BOOL enable;
    BYTE type;
    TimeStruct AlarmT;
} AlarmStruct;

**Parameters**

enable: Enable or disable alarm setting.
type:   Type of alarm:
        HOURLY_ALARM
        DAILY_ALARM
        WEEKLY_ALARM
        MONTHLY_ALARM
        YEARLY_ALARM
        ONCE_ALARM
AlarmT:   Time setting for this alarm.

**Description**

A struct stores alarm time setting.

**UCFlag**
typedef struct{
bool EnableWDT;
bool EnableAlarm;
bool EnableTrigger0;
bool EnableTrigger1;
} UCFlag;

**Parameters**
EnableWDT: Enable or disable watch dog timer.
EnableAlarm: Enable or disable alarm.
EnableTrigger0: Enable or disable trigger0 on board to boot the system.
EnableTrigger1: Enable or disable trigger1 on board to boot the system.

**Description**
A struct stores system boot setting.
Method
Adv_DVPAPI_CreateSDKInstance

Syntax
int Adv_DVPAPI_CreateSDKInstance(void **pp)

Parameters
pp: A pointer to the SDK.

Return Value
SUCCEEDED: Function succeeded.
PARAMERROR: Parameter error.
SDKINITFAILED: Failed to initialize SDK.

Description
This function creates SDK instance.
**Adv_DVPAPI_GetNumberOfDevices**

**Syntax**
```
int Adv_DVPAPI_GetNoOfDevices(void)
```

**Parameters**
None

**Return Value**
Number of DVP7010AX Capture Devices

**Description**
This function gets number of DVP7010AX Capture Devices in the system. At most 16 channels are available in a DVP7010AX integrated system.
Adv_DVPAPI_InitSDK

**Syntax**
```
int Adv_DVPAPI_InitSDK(int NoOfDevs, int* IDList)
```

**Parameters**
- **NoOfDevs**: Number of devices.
- **IDs**: An array pointer stores all board IDs. Negative value identifies inactive channel.

**Return Value**
- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Failed to get board ID or duplicate board ID.
- **NODEVICES**: No devices found.

**Description**
This function initializes all DVP7010AX capture devices in the system and gets all board IDs. After initializing each device, the capture status would be set as “STOPPED”.

**See Also**
- Adv_DVPAPI_GetNoOfDevices
- Adv_DVPAPI_GetCapState
- Adv_DVPAPI_CloseSDK
Adv_DVPAPI_CloseSDK

**Syntax**
```
int Adv_DVPAPI_CloseSDK(void)
```

**Parameters**
None

**Return Value**
- **SUCCEEDED**: Function succeeded.
- **PARAMERROR**: Parameter error.
- **SDKINITFAILED**: SDK not initialized.

**Description**
This function cleans all instances of capture devices and closes up the SDK.

**See Also**
Adv_DVPAPI_InitSDK
Adv_DVPAPI_Start

Syntax
int Adv_DVPAPI_Start(int BoardID, int SwitchingChans, HWND Main, HWND hwndPreview)

Parameters
BoardID: Specifies the board ID number(0~15).
SwitchingChans: Single video input or switching between video muxes.
0 single channel.
2 channels (mux0, mux1).
3 channels (mux0, mux1, mux2).
4 channels (mux0, mux1, mux2, mux3).
Main: A main window handle.
hwndPreview: A windows handle for display area.

Return Value
SUCCEEDED: Function succeeded.
FAILED: Function failed.
BOARDIDERROR: Invalid board ID.
SDKINITFAILED: SDK not initialized.

Description
This function starts video capturing on a specified capture board. The capture state would be set as "RUNNING" after a successful start.

See Also
Adv_DVPAPI_Stop
Adv_DVPAPI_GetCapState
Adv_DVPAPI_Stop

**Syntax**

```c
int Adv_DVPAPI_Stop(int BoardID)
```

**Parameters**

BoardID: Specifies the board ID number(0~15).

**Return Value**

SUCCEEDED: Function succeeded.
FAILED: Function failed.
BOARDIDERROR: Invalid board ID.
SDKINITFAILED: SDK not initialized.

**Description**

This function stops video capturing on a specified capture board. The capture state would be set as “STOPPED” after a successful stop.

**See Also**

Adv_DVPAPI_Start
Adv_DVPAPI_GetCapState
Adv_DVPAPI_GetCapState

**Syntax**

```c
int Adv_DVPAPI_GetCapState(int BoardID)
```

**Parameters**

BoardID: Specifies the board ID number (0~15).

**Return Value**

- BOARDIDERROR: Invalid board ID.
- SDKINITFAILED: SDK not initialized.

**Description**

This function gets capture state of a specified capture board.

- **STOPPED** = 1,
- **RUNNING** = 2,
- **UNINITIALIZED** = -1,
- **UNKNOWNSTATE** = -2

**See Also**

- Adv_DVPAPI_InitSDK
- Adv_DVPAPI_Start
- Adv_DVPAPI_Stop
Adv_DVPAPI_GetCurFrameBuffer

Syntax
int Adv_DVPAPI_GetCurFrameBuffer(int BoardID, long* bufSize, BYTE* buf, int VMux)

Parameters
- **BoardID**: Specifies the board ID number (0~15).
- **bufSize**: Frame buffer size.
- **buf**: Frame buffer.
- **VMux**: Video mux.

Return Value
- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.
- **NOSAMPLE**: No buffer sample.

Description
This function gets current frame buffer of a specified capture board. Start capturing before the function is called.

See Also
Adv_DVPAPI_Start
Adv_DVPAPI_SetNewFrameCallback

**Syntax**
```
int Adv_DVPAPI_SetNewFrameCallback(int BoardID, int callback)
```

**Parameters**
- **BoardID**: Specifies the board ID number (0~15).
- **callback**: Callback function.
- Callback function type:
  ```
  int (int lParam, int nID, int BoardID, int VMux, int bufsize, BYTE* buf);
  ```

**Return Value**
- **SUCCEEDED**: Function succeeded.
- **BOARDIDERROR**: Invalid board ID.
- **SDKINITFAILED**: SDK not initialized.

**Description**
This function sets a callback function to SDK. When new frame arrived, messages and frame information will be sent to callback function.

**See Also**
Adv_DVPAPI_GetVideoFormat

**Syntax**
```c
int Adv_DVPAPI_GetVideoFormat(int BoardID, AnalogVideoFormat* vFormat)
```

**Parameters**
- **BoardID**: Specifies the board ID number (0~15).
- **Vformat**: A pointer to get video format. Video_None, Video_NTSC_M, Video_NTSC_M_J, Video_PAL_B, Video_PAL_M, Video_PAL_N, Video_SECAM_B

**Return Value**
- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**
This function gets video input format of a specified capture board.

**See Also**
Adv_DVPAPI_SetVideoFormat
Adv_DVPAPI_SetVideoFormat

Syntax
int Adv_DVPAPI_SetVideoFormat(int BoardID,
AnalogVideoFormat* vFormat)

Parameters
BoardID: Specifies the board ID number (0~15).
Vformat: video format:
Video_None,
Video_NTSC_M,
Video_NTSC_M_J,
Video_PAL_B,
Video_PAL_M,
Video_PAL_N,
Video_SECAM_B

Return Value
SUCCEEDED: Function succeeded.
FAILED: Function failed.
BOARDIDERROR: Invalid board ID.
SDKINITFAILED: SDK not initialized.

Description
This function sets video input format a specified capture board. This function should be called before “Adv_DVPAPI_Start”.

See Also
Adv_DVPAPI_GetVideoFormat
Adv_DVPAPI_GetFrameRate

Syntax
int Adv_DVPAPI_GetFrameRate(int BoardID, double *FrameRate)

Parameters
BoardID: Specifies the board ID number(0~15).
FrameRate: A pointer to get video frame rate.

Return Value
SUCCEEDED: Function succeeded.
FAILED: Function failed.
BOARDIDERROR: Invalid board ID.
PARAMERROR: Invalid parameter.
SDKINITFAILED: SDK not initialized.

Description
This function gets frame rate of a specified capture board.

See Also
Adv_DVPAPI_SetFrameRate
Adv_DVPAPI_SetFrameRate

Syntax
int Adv_DVPAPI_SetFrameRate(int BoardID, double FrameRate)

Parameters
BoardID: Specifies the board ID number (0~15).
FrameRate: A value to set frame rate. (0.0<FrameRate<=30.0, Default value is 30.0)

Return Value
SUCCEEDED: Function succeeded.
FAILED: Function failed.
BOARDIDERROR: Invalid board ID.
PARAMERROR: Invalid parameter.
SDKINITFAILED: SDK not initialized.

Description
This function sets frame rate of a specified capture board.
This function should be called before “Adv_DVPAPI_Start”.

See Also
Adv_DVPAPI_GetFrameRate
Adv_DVPAPI_GetVideoResolution

Syntax
int Adv_DVPAPI_GetResolution(int BoardID, VideoSize *Size)

Parameters
BoardID: Specifies the board ID number(0~15).
Size: A pointer to get video resolution.
  SIZED1 (NTSC: 720x480, PAL: 720x576),
  SIZEVGA (640x480),
  SIZEQVGA (320x240),
  SIZESUBQVGA (160x120),

Return Value
SUCCEEDED: Function succeeded.
FAILED: Function failed.
BOARDIDERROR: Invalid board ID.
PARAMERROR: Invalid parameter.
SDKINITFAILED: SDK not initialized.

Description
This function gets video resolution of a specified capture board.

See Also
Adv_DVPAPI_SetResolution
Adv_DVPAPI_SetVideoResolution

Syntax
int Adv_DVPAPI_SetResolution(int BoardID, VideoSize Size)

Parameters
BoardID: Specifies the board ID number(0~15).
Size: A value to set video resolution. SIZE1 (NTSC: 720x480, PAL: 720x576), SIZEVGA (640x480), SIZEQVGA (320x240), SIZESUBQVGA (160x120),

Return Value
SUCCEEDED: Function succeeded.
FAILED: Function failed.
BOARDIDERROR: Invalid board ID.
SDKINITFAILED: SDK not initialized.

Description
This function sets video resolution of a specified capture board. This function should be called before “Adv_DVPAPI_Start”.

See Also
Adv_DVPAPI_GetResolution
Adv_DVPAPI_ GetVideoInput

**Syntax**

```c
int Adv_DVPAPI_GetVideoInput(int BoardID, int* input)
```

**Parameters**

- **BoardID**: Specifies the board ID number (0~15).
- **input**: A pointer to get video input mux.

**Return Value**

- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**

This function gets video input mux of a specified capture board. It returns “FAILED” when argument “SwitchingChans” of Adv_DVPAPI_Start was not set to 0. (This function works for no video mux automatically switching.)

**See Also**

- Adv_DVPAPI_Start
- Adv_DVPAPI_SetVideoInput
Adv_DVPAPI_SetVideoVideoInput

**Syntax**

```c
int Adv_DVPAPI_SetVideoInput(int BoardID, int input)
```

**Parameters**

- **BoardID**: Specifies the board ID number (0~15).
- **input**: A value to set video input mux (0~3).

**Return Value**

- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**

This function sets video input mux of a specified capture board. It returns “FAILED” when argument “SwitchingChans” of Adv_DVPAPI_Start was not set to 0. (This function works for no video mux automatically switching.)

**See Also**

- Adv_DVPAPI_Start
- Adv_DVPAPI_GetVideoInput
Adv_DVPAPI_GetBrightness

**Syntax**

```c
int Adv_DVPAPI_GetBrightness(int BoardID, long *pnValue)
```

**Parameters**

- **BoardID**: Specifies the board ID number (0~15).
- **pnValue**: A long pointer to get brightness value.

**Return Value**

- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**

This function gets brightness value of a specified capture board.

**See Also**

Adv_DVPAPI_SetBrightness
**Adv_DVPAPI_SetBrightness**

**Syntax**

```c
int Adv_DVPAPI_SetBrightness(int BoardID, long nValue)
```

**Parameters**

- **BoardID:** Specifies the board ID number (0~15).
- **nValue:** A value to set brightness (0~100).

**Return Value**

- **SUCCEEDED:** Function succeeded.
- **FAILED:** Function failed.
- **BOARDIDERROR:** Invalid board ID.
- **PARAMERROR:** Invalid parameter.
- **SDKINITFAILED:** SDK not initialized.

**Description**

This function sets brightness value of a specified capture board.

**See Also**

Adv_DVPAPI_GetBrightness
Adv_DVPAPI_GetContrast

**Syntax**

```c
int Adv_DVPAPI_GetContrast(int BoardID, long *pnValue)
```

**Parameters**

- **BoardID**: Specifies the board ID number (0~15).
- **pnValue**: A long pointer to get contrast value.

**Return Value**

- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**

This function gets contrast value of a specified capture board.

**See Also**

Adv_DVPAPI_SetContrast
Adv_DVPAPI_SetContrast

Syntax
int Adv_DVPAPI_SetContrast(int BoardID, long nValue)

Parameters
BoardID: Specifies the board ID number (0~15).
nValue: A value to set contrast (0~100).

Return Value
SUCCEEDED: Function succeeded.
FAILED: Function failed.
BOARDIDERROR: Invalid board ID.
PARAMERROR: Invalid parameter.
SDKINITFAILED: SDK not initialized.

Description
This function sets contrast value of a specified capture board.

See Also
Adv_DVPAPI_GetContrast
Adv_DVPAPI_GetHue

**Syntax**

```c
int Adv_DVPAPI_GetHue(int BoardID, long *pnValue)
```

**Parameters**

- **BoardID**: Specifies the board ID number (0~15).
- ** pnValue**: A long pointer to get hue value.

**Return Value**

- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**

This function gets hue value of a specified capture board.

**See Also**

Adv_DVPAPI_SetHue
Adv_DVPAPI_SetHue

**Syntax**

```c
int Adv_DVPAPI_SetHue(int BoardID, long nValue)
```

**Parameters**

- **BoardID**: Specifies the board ID number (0~15).
- **nValue**: A value to set hue (0~100).

**Return Value**

- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**

This function sets hue value of a specified capture board.

**See Also**

- Adv_DVPAPI_GetHue
Adv_DVPAPI_GetSaturation

Syntax
int Adv_DVPAPI_GetSaturation(int BoardID, long *pnValue)

Parameters
BoardID: Specifies the board ID number (0~15).
pnValue: A long pointer to get saturation value.

Return Value
SUCCEEDED: Function succeeded.
FAILED: Function failed.
BOARDIDERROR: Invalid board ID.
PARAMERROR: Invalid parameter.
SDKINITFAILED: SDK not initialized.

Description
This function gets saturation value of a specified capture board.

See Also
Adv_DVPAPI_SetSaturation
Adv_DVPAPI_SetSaturation

**Syntax**

```c
int Adv_DVPAPI_SetSaturation(int BoardID, long nValue)
```

**Parameters**

- **BoardID**: Specifies the board ID number (0~15).
- **nValue**: A value to set saturation (0~100).

**Return Value**

- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**

This function sets saturation value of a specified capture board.

**See Also**

Adv_DVPAPI_GetSaturation
Adv_DVPAPI_GPIOGetData

Syntax
int Adv_DVPAPI_GPIOGetData(int BoardID, int Pin, BOOL* value)

Parameters
BoardID: Specifies the board ID number(0~15).
Pin: GPIO pin.
value: A pointer to get specified pin value.

Return Value
SUCCEEDED: Function succeeded.
FAILED: Function failed.
BOARDIDERROR: Invalid board ID.
PARAMERROR: Invalid parameter.
SDKINITFAILED: SDK not initialized.

Description
This function gets value of specified GPIO pin on a selected board.

See Also
Adv_DVPAPI_GPIOSetData
Adv_DVPAPI_GPIOSetData

Syntax
int Adv_DVPAPI_GPIOSetData(int BoardID, int Pin, BOOL value)

Parameters
BoardID: Specifies the board ID number(0~15).
Pin: GPIO pin.
nValue: A value to set specified pin value.

Return Value
SUCCEEDED: Function succeeded.
FAILED: Function failed.
BOARDIDERROR: Invalid board ID.
PARAMERROR: Invalid parameter.
SDKINITFAILED: SDK not initialized.

Description
This function sets value of specified GPIO pin on a selected board.

See Also
Adv_DVPAPI_GPIOGetData
Adv_DVPAPI_GetWDTTimeout

Syntax
int Adv_DVPAPI_GetWDTTimeout(int BoardID, BOOL *EnableWDT, int *timeout)

Parameters
BoardID: Specifies the board ID number(0~15).
EnableWDT: A pointer to get watch dog timer state.
timeout: A pointer to get watch dog timer.

Return Value
SUCCEEDED: Function succeeded.
FAILED: Function failed.
BOARDIDERROR: Invalid board ID.
PARAMERROR: Invalid parameter.
SDKINITFAILED: SDK not initialized.

Description
This function gets state and value of watch dog timer on a selected board.

See Also
Adv_DVPAPI_SetWDTTimeout
Adv_DVPAPI_SetWDTTimeout

**Syntax**
```
int Adv_DVPAPI_SetWDTTimeout(int BoardID, BOOL EnableWDT, int timeout)
```

**Parameters**
- **BoardID**: Specifies the board ID number (0~15).
- **EnableWDT**: Enable or disable. Watch dog timer.
- **timeout**: A pointer to get watch dog timer.

**Return Value**
- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**
This function sets state and value of watch dog timer on a selected board.

**See Also**
Adv_DVPAPI_GetWDTTimeout
Adv_DVPAPI_GetUCFlag

**Syntax**

```c
int Adv_DVPAPI_GetUCFlag(int BoardID, BOOL *enableAlarm, BOOL *enableTrig0, BOOL *enableTrig1)
```

**Parameters**

- **BoardID**: Specifies the board ID number (0~15).
- **EnableWDT**: A pointer to get watchdog timer (enable or disable).
- **EnableAlarm**: A pointer to get alarm (enable or disable).
- **EnableTrigger0**: A pointer to get trigger0 (enable or disable).
- **EnableTrigger1**: A pointer to get trigger1 (enable or disable).

**Return Value**

- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**

This function gets settings (enable or disable) of WDT, alarm, and triggers on a selected board.

**See Also**

Adv_DVPAPI_SetUCFlag
UCFlag
Adv_DVPAPI_SetUCFlag

**Syntax**

```c
int Adv_DVPAPI_SetUCFlag(int BoardID, BOOL enableAlarm, BOOL enableTrig0, BOOL enableTrig1)
```

**Parameters**

- **BoardID**: Specifies the board ID number (0~15).
- **EnableWDT**: A value to enable or disable watchdog timer.
- **EnableAlarm**: A value to enable or disable alarm.
- **EnableTrigger0**: A value to enable or disable trigger0.
- **EnableTrigger1**: A value to enable or disable trigger1.

**Return Value**

- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**

This function enables or disables WDT, alarm, and triggers on a selected board.

**See Also**

- Adv_DVPAPI_GetUCFlag
- UCFlag
**Adv_DVPAPI_GetPoweronEvent**

**Syntax**

```c
int Adv_DVPAPI_GetPoweronEvent(int BoardID, POWERON_EVENT *powerEvent)
```

**Parameters**

- **BoardID**: Specifies the board ID number (0~15).
- **powerEvent**: A pointer to get system current boot type.
  - BY_USER,
  - BY_ALARM,
  - BY_TRIGGER0,
  - BY_TRIGGER1

**Return Value**

- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**

This function gets the type of system boot after setting.

**See Also**

55
Adv_DVPAPI_GetAlarm

**Syntax**

```c
int Adv_DVPAPI_GetAlarm(int BoardID, int index, AlarmStruct* alarm)
```

**Parameters**
- **BoardID**: Specifies the board ID number(0~15).
- **index**: Specifies the alarm number(0~9).
- **alarm**: A AlarmStruct pointer to get alarm setting.

**Return Value**
- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**
This function gets setting of specified alarm on a selected board.

**See Also**
- Adv_DVPAPI_SetAlarm
- AlarmStruct
Adv_DVPAPI_SetAlarm

**Syntax**
```c
int Adv_DVPAPI_SetAlarm(int BoardID, int index, AlarmStruct* alarm)
```

**Parameters**
- **BoardID**: Specifies the board ID number (0~15).
- **index**: Specifies the alarm number (0~9).
- **alarm**: Alarm setting.

**Return Value**
- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**
This function sets setting of specified alarm on a selected board.

**See Also**
- Adv_DVPAPI_GetAlarm
- AlarmStruct
Adv_DVPAPI_GetChecksum

Syntax
int Adv_DVPAPI_GetChecksum(int BoardID, BYTE* key, BYTE* checksum)

Parameters
BoardID: Specifies the board ID number (0~15).
key: Input key for checksum value.
checksum: A pointer to get checksum value.

Return Value
SUCCEEDED: Function succeeded.
FAILED: Function failed.
BOARDIDERROR: Invalid board ID.
PARAMERROR: Invalid parameter.
SDKINITFAILED: SDK not initialized.

Description
This function gets checksum value with a “KEY” input on a selected board.

See Also
Adv_DVPAPI_GetEEData

**Syntax**

```c
int Adv_DVPAPI_GetEEData(int BoardID, BYTE wordAddr, BYTE* pData)
```

**Parameters**

- **BoardID**: Specifies the board ID number (0~15).
- **wordAddr**: Specifies the word address (0~127).
- **pData**: A pointer to get byte value stored in EE.

**Return Value**

- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**

This function reads the value at specified EE word address on a selected board.

**See Also**

Adv_DVPAPI_SetEEData
Adv_DVPAPI_SetEEData

Syntax
int Adv_DVPAPI_SetEEData(int BoardID, BYTE wordAddr, BYTE* pData)

Parameters
BoardID: Specifies the board ID number(0~15).
wordAddr: Specifies the word address(0~127).
pData: A value to set the byte value in EE.

Return Value
SUCCEEDED: Function succeeded.
FAILED: Function failed.
BOARDIDERROR: Invalid board ID.
PARAMERROR: Invalid parameter.
SDKINITFAILED: SDK not initialized.

Description
This function writes the value at specified EE word address on a selected board.

See Also
Adv_DVPAPI_GetEEData
Adv_DVPAPI_GetRTCDData

**Syntax**

```c
int Adv_DVPAPI_GetRTCData(int BoardID, TimeStruct* time)
```

**Parameters**

- **BoardID**: Specifies the board ID number (0~15).
- **time**: A TimeStruct pointer to get RTC

**Return Value**

- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**

This function gets RTC settings on a selected board.

**See Also**

- Adv_DVPAPI_SetRTCData
- TimeStruct
Adv_DVPAPI_SetRTCData

**Syntax**
int Adv_DVPAPI_SetRTCData(int BoardID, TimeStruct* time)

**Parameters**
- **BoardID**: Specifies the board ID number (0~15).
- **time**: A TimeStruct pointer to set RTC

**Return Value**
- **SUCCEEDED**: Function succeeded.
- **FAILED**: Function failed.
- **BOARDIDERROR**: Invalid board ID.
- **PARAMERROR**: Invalid parameter.
- **SDKINITFAILED**: SDK not initialized.

**Description**
This function sets RTC settings on a selected board.

**See Also**
- Adv_DVPAPI_GetRTCData
- TimeStruct