DVP-7010
4 Channel PCI-bus Video Grabber Card
User's Manual

Acknowledgments
DVP-7010 is a trademark of Advantech Co., Ltd. IBM and PC are trademarks of International Business Machines Corporation. MS-DOS, Windows, Microsoft Visual C++ and Visual BASIC are trademarks of Microsoft Corporation. Intel and Pentium are trademarks of Intel Corporation. Delphi and C++ Builder are trademarks of Inprise Corporation.

CE notification
The DVP-7010, 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

Part No. XXXXXX 1st Edition
Printed in Taiwan January 2004

Contents

CHAPTER 1

General Information

Chapter 1 General Information

Thank you for buying the Advantech DVP-7010. The DVP-7010 is a 4 channels input, PCI bus digital video grabber card. It supporting NTSC / PAL (NTSC-M, NTSC-Japan, PAL-B, PAL-D, PAL-G, PAL-H, PAL-I, PAM-M, PAL-N)/SECAM video input also built-in Watchdog timer function to prevent system crash. The DVP-7010 captures 30 fps/channel (PAL format: 25 fps/channel). The DVP-7010 has extensive functionalities such as multiple resolutions, providing high-quality and raising reliability to a new level, allowing OEMs and system integrators to build powerful, yet cost-effective surveillance solutions.
The following sections of this chapter will provide further information about features of the multifunction cards, a Quick Start for installation, together with some brief information on software and accessories for the DVP-7010 card.

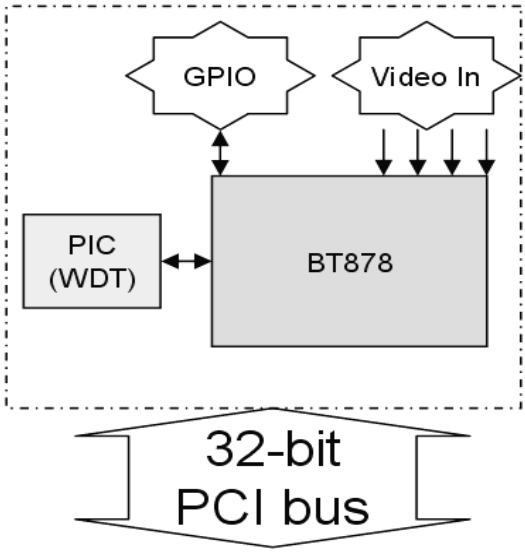
1.1 Hardware Requirement

- Intel Pentium III 1GHz or above (The CPU speed is depends on the video frame rate, channels and resolution)
- 128MB RAM or above
- Free PCI slot(s)
- CD-ROM
- Hard disk with 1GB free space

1.2 Software Requirement

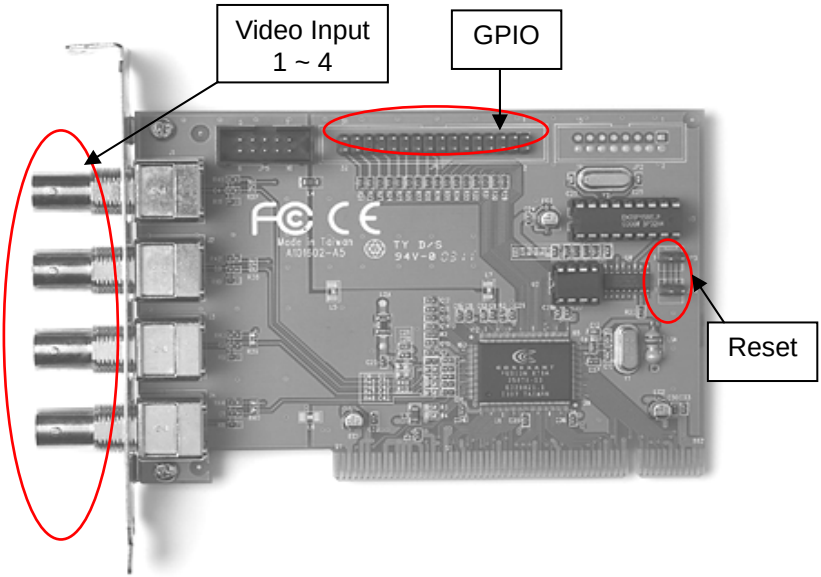
- Microsoft Windows 98/ME/2000/XP with DirectX 8.1 or above

1.3 Block Diagram



(DVP-7010)

1.4 Jumper/Connector Location



1.5 Packing List

- DVP-7010 / 7020 PCI-bus Video Capture card
- CD Disk for manual / driver / SDK
- User's Manual
- Reboot Cable for Watchdog function

1.6 Watchdog Function

The Watchdog function is a fail-protection system which built-in DVP-7010 / 7020. There are two connectors on our boards. One is connecting to the RESET switch on chassis and the others to RESET-pin on main board. Please reference the programming function description of Adv_VAPI_SetWDT for detail information

1.7 GPIO Function

The GPIO (JP1) are TTL compatible signal, programmer can uses these I/O for alarm detecting or output.

I/O connector signal descriptions

Pin #	Function	Pin #	Function	Pin #	Function	Pin #	Function
1	GPIO_8	9	GPIO_12	17	GND	25	GND
2	GND	10	GND	18	GPIO_7	26	GPIO_3
3	GPIO_9	11	GPIO_13	19	GND	27	GND
4	GND	12	GND	20	GPIO_6	28	GPIO_2
5	GPIO_10	13	GPIO_14	21	GND	29	GND
6	GND	14	GND	22	GPIO_5	30	GPIO_1
7	GPIO_11	15	GPIO_15	23	GND	31	GND
8	GND	16	GND	24	GPIO_4	32	GPIO_0

1.8 Hardware Installation

1. Turn off your computer and unplug the power cord.
2. Remove the cover of your computer.
3. Remove the slot cover on the back panel of your computer.
4. Touch the metal part on the surface of your computer to neutralize the static electricity that might be on your body.

5. Insert the DVP-7010 / 7020 card into an unused PCI slot. Hold the card only by its edges and carefully align it with the slot. Insert the card firmly into place. Use of excessive force must be avoided; otherwise the card might be damaged.
6. Fasten the bracket of the PCI card on the back panel rail of the computer with screws.
7. Connect appropriate accessories (Video cable to camera. if necessary) to the PCI card.
8. Replace the cover of your computer chassis.
9. Plug in the power cord and turn on the computer...
10. Remove the original BT878 driver before the card installed if it is present.

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.

1.9 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-7010 board are located on the software installation CD. The auto-run function of the driver

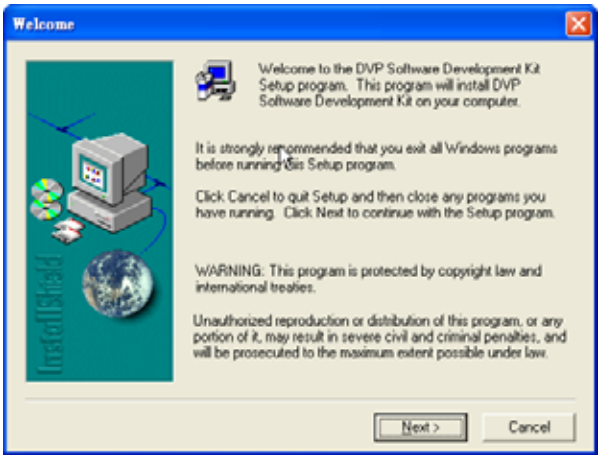
CD will guide and link you to the utilities and device drivers under Windows system.

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.

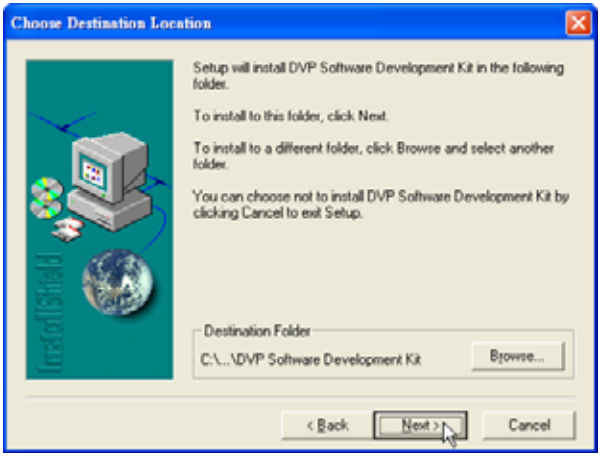
Installing

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.

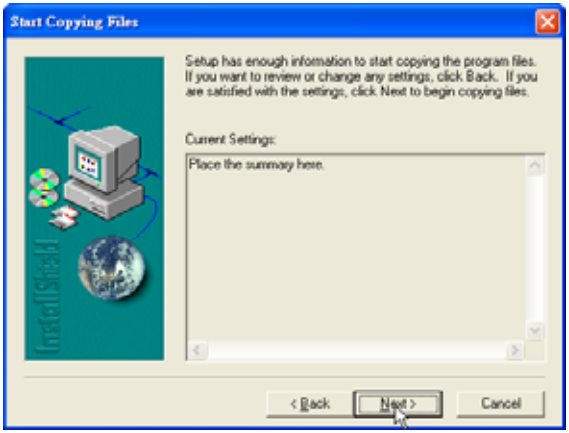
2. Click "Next" when you see the following message.



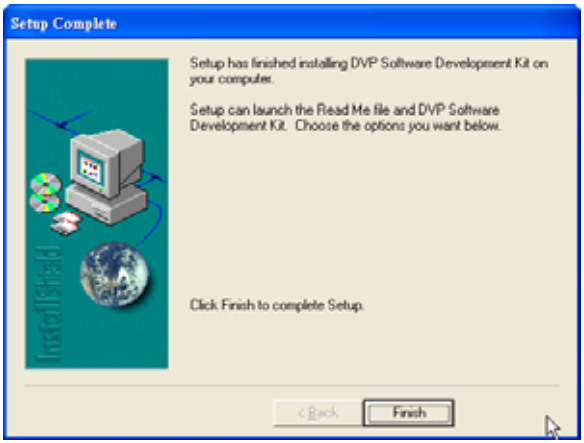
3. Click "Next" when you see the following message.



4. Click "Next" when you see the following message.



5. When the following message appears, click "Finish" to complete the installation and restart Windows.



CHAPTER 2

Functions Library

Chapter 2 Functions Library

2.1 Summary Tables

The following table summaries the functions that belong to Advantech VAPI (Video Application Program Interface) library. Functions are grouped by tasks you might wish to perform.

Initialized and Close Functions:

Name	Description
Adv_VAPI_Init	Initialize the library
Adv_VAPI_Close	Close the library

Port Configurative Functions:

Name	Description
Adv_VAPI_SetCurrentPort	Set the current card number and port number
Adv_VAPI_GetCurrentCard	Get current card number
Adv_VAPI_GetCurrentPort	Get current port number
Adv_VAPI_SetPortType	Assign a device to the current port
Adv_VAPI_GetPortType	Get the current device on the current port
Adv_VAPI_SetVideoFormat	Set the video format on the current port
Adv_VAPI_SetCaptureSize	Set the video capturing size on the current port
Adv_VAPI_GetVideoFormat	Get the video format on the current port
Adv_VAPI_GetCaptureSize	Get the video capturing size on the current port

Name	Description
Adv_VAPI_SetBrightness	Set the video brightness value for a specified channel
Adv_VAPI_SetSaturation	Set the video saturation value for a specified channel
Adv_VAPI_SetContrast	Set the video contrast value for a specified channel
Adv_VAPI_SetHue	Set the video hue value for a specified channel
Adv_VAPI_GetBrightness	Get the video brightness value for a specified channel
Adv_VAPI_GetSaturation	Get the video saturation value for a specified channel
Adv_VAPI_GetContrast	Get the video contrast value for a specified channel
Adv_VAPI_GetHue	Get the video hue value for a specified channel

Channel Capture Setting Functions:

Name	Description
Adv_VAPI_SetCaptureWindow	Set the video capturing window handle to a specified channel
Adv_VAPI_SetCaptureBuf	Set the video capturing buffer address to a specified channel
Adv_VAPI_SetCaptureFile	Set the video capturing file handle to a specified channel
Adv_VAPI_SetCaptureCallback	Set the video capturing callback function to a specified channel
Adv_VAPI_GetCaptureStatus	Get the capturing status of a specified channel

Port Capture Functions:

Name	Description
Adv_VAPI_SetCaptureSource	Set the video capture source on the current port
Adv_VAPI_GetCaptureSource	Get the video capture source on the current port
Adv_VAPI_CaptureStart	Start the video capture on the current port
Adv_VAPI_CaptureStop	Stop the video capture on the current port
Adv_VAPI_EnableVideoSource	Switch the video source on or off
Adv_VAPI_GetVideoSignal	Check the video signal available

Port GPIO Functions:

Name	Description
Adv_VAPI_GPIOInit	Initialize the direction of GPIO on the current port
Adv_VAPI_GPIOGetDirect	Get the direction of GPIO on the current port
Adv_VAPI_SetGPIOData	Set the GPIO data on the current port
Adv_VAPI_GetGPIOData	Get the GPIO data on the current port

Port Watchdog Functions:

Name	Description
Adv_VAPI_SetWDT	Set the watchdog state on the current port

Error Debug Functions:

Name	Description

Adv_VAPI_GetLastError	Get the last error code
Adv_VAPI_ErrorMsg	Display the last error message if failed

2.2 Programming Functions Reference

There are three programming languages for your reference: C++, Delphi and Visual BASIC, and also we provides sample program within our bundle CD.

Adv_VAPI_Init

Syntax

[C++]

*BOOL Adv_VAPI_Init(DWORD *pdwVer)*

[Delphi]

function Adv_VAPI_Init(var dwVer:Longword):Boolean;

[VB]

function Adv_VAPI_Init (ByRef pdwVer As Long) As Boolean

Parameters

pdwVer : Specifies the pointer of DWORD. It will be assigned to the version number of this library. For example, if assigned value 0x10000, it represents Version 1.00. Now the version is 0x10000,

Return Value

TRUE : Initialization is successful

FALSE : Initialization is failed

Description

This function will initialize and allocate the library variables and resources. You must call this function before calling other functions.

Example

DWORD dwVer;

```
        …
if (Adv_VAPI_Init(&dwVer))
{ // Success
      …
}
else // Fail
                  Adv_VAPI_GetLastError(NULL);
```

See Also

Adv_VAPI_Close

Adv_VAPI_Close

Syntax

[C++]

void Adv_VAPI_Close(void)

[Delphi]

procedure Adv_VAPI_Close;

[VB]

sub Adv_VAPI_Close ()

Parameters

None

Return Value

None

Description

This function will release the library variables and functions. You must call this function before closing the application.

See Also

Adv_VAPI_Init

Adv_VAPI_SetCurrentPort

Syntax

[C++]

BOOL Adv_VAPI_SetCurrentPort(int nCardID,int nPortID)

[Delphi]

function Adv_VAPI_SetCurrentPort(nCardID,nPortID:Integer):Boolean;

[VB]

function Adv_VAPI_SetCurrentPort (ByVal nCardID As Integer, ByVal nPortID As Integer) As Boolean

Parameters

nCardID : Set a card number to current virtual card number. Its range is from 0 to 3.

nPortID : Set a port number to current virtual port number. Its range is from 0 to 3.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will assign a card number and port number to current virtual port. With DVP-7010 card serials, a physical card has only one port. With DVP-7020 card serials, a physical card has four ports. There are 1~4 video channels on each physical port.

See Also

Adv_VAPI_GetCurrentCard, Adv_VAPI_GetCurrentPort

Adv_VAPI_GetCurrentCard

Syntax

[C++]

int Adv_VAPI_GetCurrentCard(void)

[Delphi]

function Adv_VAPI_GetCurrentCard:Integer;

[VB]

function Adv_VAPI_GetCurrentCard () As Integer

Parameters

None

Return Value

The current card number, its range is 0 to 3.

Description

This function will return the current card number.

See Also

Adv_VAPI_SetCurrentPort, Adv_VAPI_GetCurrentPort

Adv_VAPI_GetCurrentPort

Syntax

[C++]

int Adv_VAPI_GetCurrentPort(void)

[Delphi]

function Adv_VAPI_GetCurrentPort:Integer;

[VB]

function Adv_VAPI_GetCurrentPort () As Integer

Parameters

None

Return Value

The current port number, the range is 0 to 3.

Description

This function will return the current port number.

See Also

Adv_VAPI_SetCurrentPort, Adv_VAPI_GetCurrentCard

Adv_VAPI_SetPortType

Syntax

[C++]

BOOL Adv_VAPI_SetPortType(DEVICE_TYPE deviceType)

[Delphi]

function Adv_VAPI_SetPortType(deviceType:Integer):Boolean;

[VB]

function Adv_VAPI_SetPortType (ByVal deviceType As Integer) As Boolean

Parameters

deviceType : Specifies the physical port type. Its valid values are the following list

V_TYPENONE	:	no device
V_ADV7010_0 :		DVP-7010 Card 0
V_ADV7010_1 :		DVP-7010 Card 1
V_ADV7010_2 :		DVP-7010 Card 2
V_ADV7010_3 :		DVP-7010 Card 3
V_ADV7020_0_0	:	DVP-7020 Card 0 Port 0
V_ADV7020_0_1	:	DVP-7020 Card 0 Port 1
V_ADV7020_0_2	:	DVP-7020 Card 0 Port 2
V_ADV7020_0_3	:	DVP-7020 Card 0 Port 3
V_ADV7020_1_0	:	DVP-7020 Card 1 Port 0
V_ADV7020_1_1	:	DVP-7020 Card 1 Port 1
V_ADV7020_1_2	:	DVP-7020 Card 1 Port 2
V_ADV7020_1_3	:	DVP-7020 Card 1 Port 3
V_ADV7020_2_0	:	DVP-7020 Card 2 Port 0
V_ADV7020_2_1	:	DVP-7020 Card 2 Port 1
V_ADV7020_2_2	:	DVP-7020 Card 2 Port 2

V_ADV7020_2_3 : DVP-7020 Card 2 Port 3
V_ADV7020_3_0 : DVP-7020 Card 3 Port 0
V_ADV7020_3_1 : DVP-7020 Card 3 Port 1
V_ADV7020_3_2 : DVP-7020 Card 3 Port 2
V_ADV7020_3_3 : DVP-7020 Card 3 Port 3

Return Value

TRUE : Function is successful
FALSE : Function is failed

Description

This function will assign a physical port to current virtual port.

Example

```
DWORD dwVer;
…
if (Adv_VAPI_Init(&dwVer))
{ // Success
    if (!Adv_VAPI_SetCurrentPort(0,0)) // Set virtual port as (0,0)
            Adv_VAPI_GetLastError(NULL);
    if (!Adv_VAPI_SetPortType(V_ADV7010_3)) // Set DVP-7010 Card 3 to current virtual port(0,0)
            Adv_VAPI_GetLastError(NULL);
    if (!Adv_VAPI_SetCurrentPort(0,1)) // Set virtual port as (0,1)
            Adv_VAPI_GetLastError(NULL);
    if (!Adv_VAPI_SetPortType(V_ADV7010_2)) // Set DVP-7010 Card 2 to current virtual port(0,1)
            Adv_VAPI_GetLastError(NULL);
        …
}
else // Fail
            Adv_VAPI_GetLastError(NULL);
```

See Also

Adv_VAPI_SetVideoFormat, Adv_VAPI_SetCaptureSize, Adv_VAPI_GetVideoFormat, Adv_VAPI_GetCaptureSize

Adv_VAPI_GetPortType

Syntax

[C++]

*BOOL Adv_VAPI_GetPortType(DEVICE_TYPE *pDeviceType)*

[Delphi]

function Adv_VAPI_GetPortType(var DeviceType:Integer):Boolean;

[VB]

function Adv_VAPI_GetPortType (ByRef pDeviceType As Integer) As Boolean

Parameters

pDeviceType : Specifies the address of returned physical port type. It pointers to valid values are the one of which are the following list

V_TYPENONE	:	no device
V_ADV7010_0 :		DVP-7010 Card 0
V_ADV7010_1 :		DVP-7010 Card 1
V_ADV7010_2 :		DVP-7010 Card 2
V_ADV7010_3 :		DVP-7010 Card 3
V_ADV7020_0_0	:	DVP-7020 Card 0 Port 0
V_ADV7020_0_1	:	DVP-7020 Card 0 Port 1
V_ADV7020_0_2	:	DVP-7020 Card 0 Port 2
V_ADV7020_0_3	:	DVP-7020 Card 0 Port 3
V_ADV7020_1_0	:	DVP-7020 Card 1 Port 0
V_ADV7020_1_1	:	DVP-7020 Card 1 Port 1
V_ADV7020_1_2	:	DVP-7020 Card 1 Port 2
V_ADV7020_1_3	:	DVP-7020 Card 1 Port 3
V_ADV7020_2_0	:	DVP-7020 Card 2 Port 0
V_ADV7020_2_1	:	DVP-7020 Card 2 Port 1

V_ADV7020_2_2 : DVP-7020 Card 2 Port 2
V_ADV7020_2_3 : DVP-7020 Card 2 Port 3
V_ADV7020_3_0 : DVP-7020 Card 3 Port 0
V_ADV7020_3_1 : DVP-7020 Card 3 Port 1
V_ADV7020_3_2 : DVP-7020 Card 3 Port 2
V_ADV7020_3_3 : DVP-7020 Card 3 Port 3

Return Value

TRUE : Function is successful
FALSE : Function is failed

Description

This function will assign a physical port to current virtual port.

See Also

Adv_VAPI_SetVideoFormat, Adv_VAPI_SetCaptureSize, Adv_VAPI_GetVideoFormat, Adv_VAPI_GetCaptureSize

Adv_VAPI_SetVideoFormat

Syntax

[C++]

BOOL Adv_VAPI_SetVideoFormat(AnalogVideoStandard vFormat)

[Delphi]

function Adv_VAPI_SetVideoFormat(vFormat:Integer):Boolean;

[VB]

function Adv_VAPI_SetVideoFormat (ByVal vFormat As Integer) As Boolean

Parameters

vFormat : Specifies the video format on current physical port. Its values are as follows:

AnalogVideo_None	:	None,
AnalogVideo_NTSC_M:		NTSC format
AnalogVideo_PAL_B :		PAL format

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will assign the video format to the current port. The default value of video format is **AnalogVideo_NTSC_M**.

See Also

Adv_VAPI_SetPortType, Adv_VAPI_SetCaptureSize, Adv_VAPI_GetVideoFormat, Adv_VAPI_GetCaptureSize

Adv_VAPI_SetCaptureSize

Syntax

[C++]

BOOL Adv_VAPI_SetCaptureSize(VIDEO_SIZE vcSize)

[Delphi]

function Adv_VAPI_SetCaptureSize(vcSize:Integer):Boolean;

[VB]

function Adv_VAPI_SetCaptureSize (ByVal vcSize As Integer) As Boolean

Parameters

vcSize : Specifies the video capturing size and colors (BPP : bit per pixel) on current physical port. Its values are as follows:
SIZE640X240_16BPP: size resolution as 640x240 16-BPP
SIZE320X240_16BPP: size resolution as 320x240 16-BPP
SIZE160X120_16BPP: size resolution as 160x120 16-BPP
SIZE320X240_24BPP: size resolution as 320x240 24-BPP
SIZE160X120_24BPP: size resolution as 160x120 24-BPP
SIZE640X240_24BPP: size resolution as 640x240 24-BPP

Return Value

TRUE : Function is successful
FALSE : Function is failed

Description

This function will assign the video capturing size to the current port. The default value of video capturing size is **SIZE640X240_16BPP**.

See Also

Adv_VAPI_SetPortType, Adv_VAPI_SetVideoFormat, Adv_VAPI_GetVideoFormat, Adv_VAPI_GetCaptureSize

Adv_VAPI_GetVideoFormat

Syntax

[C++]

AnalogVideoStandard Adv_VAPI_GetVideoFormat(void)

[Delphi]

function Adv_VAPI_GetVideoFormat:Integer;

[VB]

function Adv_VAPI_GetVideoFormat () As Integer

Parameters

None

Return Value

The video format, its value is one of the following:

AnalogVideo_None	:	None,
AnalogVideo_NTSC_M format	:	NTSC
AnalogVideo_PAL_B format	:	PAL

Description

This function will return the video format of the current port.

See Also

Adv_VAPI_SetPortType, Adv_VAPI_SetCaptureSize, Adv_VAPI_SetVideoFormat, Adv_VAPI_GetCaptureSize

Adv_VAPI_GetCaptureSize

Syntax

[C++]

VIDEO_SIZE Adv_VAPI_GetCaptureSize(void)

[Delphi]

function Adv_VAPI_GetCaptureSize:Integer;

[VB]

function Adv_VAPI_GetCaptureSize () As Integer

Parameters

None

Return Value

The video capturing size, its value is one of the following:

SIZE640X240_16BPP: size resolution as 640x240 16-Bpp
SIZE320X240_16BPP: size resolution as 320x240 16-Bpp
SIZE160X120_16BPP: size resolution as 160x120 16-Bpp
SIZE320X240_24BPP: size resolution as 320x240 24-Bpp
SIZE160X120_24BPP: size resolution as 160x120 24-Bpp
SIZE640X240_24BPP: size resolution as 640x240 24-Bpp

Description

This function will return the video capturing size of the current port.

See Also

Adv_VAPI_SetPortType, Adv_VAPI_SetCaptureSize, Adv_VAPI_GetVideoFormat, Adv_VAPI_SetVideoFormat

Adv_VAPI_SetBrightness

Syntax

[C++]

BOOL Adv_VAPI_SetBrightness(int nChannel,int nValue)

[Delphi]

function Adv_VAPI_SetBrightness(nChannel,nValue:Integer):Boolean;

[VB]

function Adv_VAPI_SetBrightness (ByVal nChannel As Integer, ByVal nValue As Integer) As Boolean

Parameters

nChannel : Specifies the video channel number on current physical port. The range is 0~3.

nValue : Specifies the brightness of the video channel on current physical port. The range is 0~10000. 0 means deep dark and 10000 means more brightness.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will assign the brightness of video channel on the current port. The default brightness value of a video channel is **5000**.

See Also

Adv_VAPI_SetSaturation, Adv_VAPI_SetContrast, Adv_VAPI_SetHue

Adv_VAPI_SetSaturation

Syntax

[C++]

BOOL Adv_VAPI_SetSaturation(int nChannel,int nValue)

[Delphi]

function Adv_VAPI_SetSaturation(nChannel,nValue:Integer):Boolean;

[VB]

function Adv_VAPI_SetSaturation (ByVal nChannel As Integer, ByVal nValue As Integer) As Boolean

Parameters

nChannel : Specifies the video channel number on current physical port. Its range is 0~3.

nValue : Specifies the saturation of the video channel on current physical port. Its range is 0~10000. 0 means deep dark and 10000 means more saturation.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will assign the saturation of video channel on the current port. The default saturation value of a video channel is **5000**.

See Also

Adv_VAPI_SetBrightness, Adv_VAPI_SetContrast, Adv_VAPI_SetHue

Adv_VAPI_SetContrast

Syntax

[C++]

BOOL Adv_VAPI_SetContrast(int nChannel,int nValue)

[Delphi]

function Adv_VAPI_SetContrast(nChannel,nValue:Integer):Boolean;

[VB]

function Adv_VAPI_SetContrast (ByVal nChannel As Integer, ByVal nValue As Integer) As Boolean

Parameters

nChannel : Specifies the video channel number on current physical port. Its range is 0~3.

nValue : Specifies the contrast of the video channel on current physical port. Its range is 0~10000. 0 means deep dark and 10000 means more contrast.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will assign the contrast of video channel on the current port. The default contrast value of a video channel is **5000**.

See Also

Adv_VAPI_SetSaturation, Adv_VAPI_SetBrightness, Adv_VAPI_SetHue

Adv_VAPI_SetHue

Syntax

[C++]

BOOL Adv_VAPI_SetHue(int nChannel,int nValue)

[Delphi]

function Adv_VAPI_SetHue(nChannel,nValue:Integer):Boolean;

[VB]

function Adv_VAPI_SetHue (ByVal nChannel As Integer, ByVal nValue As Integer) As Boolean

Parameters

nChannel : Specifies the video channel number on current physical port. Its range is 0~3.

nValue : Specifies the hue of the video channel on current physical port. Its range is 0~10000. 0 means deep dark and 10000 means more hue.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will assign the hue of video channel on the current port. The default hue value of a video channel is **5000**.

See Also

Adv_VAPI_SetSaturation, Adv_VAPI_SetContrast, Adv_VAPI_SetBrightness

Adv_VAPI_GetBrightness

Syntax

[C++]

*BOOL Adv_VAPI_GetBrightness(int nChannel,int *pnValue)*

[Delphi]

function Adv_VAPI_GetBrightness(nChannel:Integer; var nValue:Integer):Boolean;

[VB]

function Adv_VAPI_GetBrightness (ByVal nChannel As Integer, ByRef pnValue As Integer) As Boolean

Parameters

nChannel	: Specifies the video channel number on current physical port. Its range is 0~3.
pnValue	: Specifies the address of returned brightness value of the video channel on current physical port. The range of its value is 0~10000. 0 means deep dark and 10000 means more brightness.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will return the brightness value of video channel on the current port. The default brightness value of a video channel is **5000**.

See Also

Adv_VAPI_GetSaturation, Adv_VAPI_GetContrast, Adv_VAPI_GetHue

Adv_VAPI_GetSaturation

Syntax

[C++]

*BOOL Adv_VAPI_GetSaturation(int nChannel,int *pnValue)*

[Delphi]

function Adv_VAPI_GetSaturation(nChannel:Integer; var nValue:Integer):Boolean;

[VB]

function Adv_VAPI_GetSaturation (ByVal nChannel As Integer, ByRef pnValue As Integer) As Boolean

Parameters

nChannel : Specifies the video channel number on current physical port. Its range is 0~3.

pnValue : Specifies the address of returned saturation value of the video channel on current physical port. Its range of value is 0~10000. 0 means deep dark and 10000 means more saturation.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will return the saturation value of video channel on the current port. The default saturation value of a video channel is **5000**.

See Also

Adv_VAPI_GetBrightness, Adv_VAPI_GetContrast, Adv_VAPI_GetHue

Adv_VAPI_GetContrast

Syntax

[C++]

*BOOL Adv_VAPI_GetContrast(int nChannel,int *pnValue)*

[Delphi]

function Adv_VAPI_GetContrast(nChannel:Integer; var nValue:Integer):Boolean;

[VB]

function Adv_VAPI_GetContrast (ByVal nChannel As Integer, ByRef pnValue As Integer) As Boolean

Parameters

nChannel : Specifies the video channel number on current physical port. Its range is 0~3.

pnValue : Specifies the address of returned contrast of the video channel on current physical port. Its range of the value is 0~10000. 0 means deep dark and 10000 means more contrast.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will return the contrast value of video channel on the current port. The default contrast value of a video channel is **5000**.

See Also

Adv_VAPI_GetSaturation, Adv_VAPI_GetBrightness, Adv_VAPI_GetHue

Adv_VAPI_GetHue

Syntax

[C++]

*BOOL Adv_VAPI_GSetHue(int nChannel,int *pnValue)*

[Delphi]

function Adv_VAPI_GetHue(nChannel:Integer; var nValue:Integer):Boolean;

[VB]

function Adv_VAPI_GetHue (ByVal nChannel As Integer, ByRef pnValue As Integer) As Boolean

Parameters

nChannel	: Specifies the video channel number on current physical port. Its range is 0~3.
pnValue	: Specifies the address of returned hue of the video channel on current physical port. Its range of the value is 0~10000. 0 means deep dark and 10000 means more hue.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will return the hue of video channel on the current port. The default hue value of a video channel is **5000**.

See Also

Adv_VAPI_GetSaturation, Adv_VAPI_GetContrast, Adv_VAPI_GetBrightness

Adv_VAPI_SetCaptureWindow

Syntax

[C++]

BOOL Adv_VAPI_SetCaptureWindow(int nChannelNo,HWND hWnd,int nFrameCount,Adv_CompleteProc pCall)

[Delphi]

function Adv_VAPI_SetCaptureWindow(nChannelNo:Integer; hWnd:HWND; nFrameCount:Integer; pCall:Adv_CompleteProc):Boolean;

[VB]

function Adv_VAPI_SetCaptureWindow (ByVal nChannelNo As Integer, ByVal Hwnd As Long, ByVal nFrameCount As Integer, ByVal pCall As Long) As Boolean

Parameters

nChannelNo : Specifies the video channel number on current physical port. Its range is 0~3.

hWnd : Specifies the capture window handle for the video channel on current physical port. If the handle is NULL, the capture window of video channel will be canceled.

nFrameCount : Specifies total frame number to capture. If the value is ALWAYS_CAPTURE, the capture frame is endless.

pCall : Pointer to callback function to notify the capture completed. The function should be declared as follows:

BOOL CALLBACK *YourCompleteProc*(DWORD dwStatus)

The value of the parameter dwStatus has one of the following:

DEST_HWND defines windows capture completed.

DEST_BUF defines buffer capture completed.

DEST_FILE defines file capture completed.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will assign a window handle to display the data of video capture for a channel of video.

See Also

Adv_VAPI_SetCaptureBuf, Adv_VAPI_SetCaptureFile, Adv_VAPI_SetCaptureCallback, Adv_VAPI_GetCaptureStatus

Adv_VAPI_SetCaptureBuf

Syntax

[C++]

BOOL Adv_VAPI_SetCaptureBuf(int nChannelNo,PBYTE pBuf,DWORD dwSize,int nFrameCount,Adv_CompleteProc pCall)

[Delphi]

function Adv_VAPI_SetCaptureBuf(nChannelNo:Integer; pBuf:POINTER; dwSize:Longword; nFrameCount:Integer; pCall:Adv_CompleteProc):Boolean;

[VB]

function Adv_VAPI_SetCaptureBuf (ByVal nChannelNo As Integer, ByVal pBuf As Long, ByVal dwSize As Long, ByVal nFrameCount As Integer, ByVal pCall As Long) As Boolean

Parameters

nChannelNo : Specifies the video channel number on current physical port. Its range should be 0~3.

pBuf : Specifies the capture buffer for the video channel on current physical port. If the buffer address is NULL, the capture buffer of video channel will be canceled.

dwSize : Specifies total number of buffer. You should allocate enough space to buffer.

nFrameCount : Specifies total frame number to capture. If the value is ALWAYS_CAPTURE, the capture frame is endless.

pCall : Pointer to callback function to notify the capture completed. The function should be declared as follows:

BOOL CALLBACK *YourCompleteProc*(DWORD dwStatus)

The value of the parameter dwStatus has one of the following:

DEST_HWND defines windows capture completed.

DEST_BUF defines buffer capture completed.

DEST_FILE defines file capture completed.

Return Value

TRUE : Function is successful
FALSE : Function is failed

Description

This function will assign a buffer to store the data of video capture for a channel of video.

See Also

Adv_VAPI_SetCaptureWindow, Adv_VAPI_SetCaptureFile, Adv_VAPI_SetCaptureCallback, Adv_VAPI_GetCaptureStatus

Adv_VAPI_SetCaptureFile

Syntax

[C++]

*BOOL Adv_VAPI_SetCaptureFile(int nChannelNo,TCHAR *fileName,int nFrameCount,FILE_FORMAT fileFormat,Adv_CompleteProc pCall)*

[Delphi]

function Adv_VAPI_SetCaptureFile(nChannelNo:Integer; fileName:string; nFrameCount,fileFormat:Integer; pCall:Adv_CompleteProc):Boolean;

[VB]

function Adv_VAPI_SetCaptureFile (ByVal nChannelNo As Integer, ByVal fileName As String, ByVal nFrameCount As Integer, ByVal fileFormat As Integer, ByVal pCall As Long) As Boolean

Parameters

nChannelNo : Specifies the video channel number on current physical port. Its range should be 0~3.

fileName : Specifies the filename for the video channel on current physical port. If filName is NULL, the capture file of video channel will be canceled.

nFrameCount : Specifies total frame number to capture. If the value is ALWAYS_CAPTURE, the capture frame is endless.

fileFormat : Defines the capture file format. There are two types here:

RAW_DATA stream raw data of video channel

BMP_DATA bitmap file for one frame of video channel

pCall : Pointer to callback function to notify the capture completed. The function should be declared as follows:

BOOL CALLBACK *YourCompleteProc*(DWORD dwStatus)

The value of the parameter dwStatus has one of the following:

DEST_HWND defines windows capture

completed.

DEST_BUF	defines buffer capture completed.
DEST_FILE	defines file capture completed.

Return Value

TRUE : Function is successful
FALSE : Function is failed

Description

This function will assign a window handle to the channel of video.

See Also

Adv_VAPI_SetCaptureBuf, Adv_VAPI_SetCaptureWindow, Adv_VAPI_SetCaptureCallback, Adv_VAPI_GetCaptureStatus

Adv_VAPI_SetCaptureCallback

Syntax

[C++]

BOOL Adv_VAPI_SetCaptureCallback(int nChannelNo, Adv_CallBackProc pCall, PVOID lpParam,int nFrameCount)

[Delphi]

function Adv_VAPI_SetCaptureCallback(nChannelNo:Integer; pCall:Adv_CallBackProc; lpParam:Pointer;nFrameCount:Integer):Boolean;

[VB]

function Adv_VAPI_SetCaptureCallback (ByVal nChannelNo As Integer, ByVal pCall As Long, ByVal lpParam As Long, ByVal nFrameCount As Integer) As Boolean

Parameters

nChannelNo : Specifies the video channel number on current physical port. Its range should be 0~3.

pCall : Pointer to callback function to notify the capture one frame. The function should be declared as follows:

HRESULT CALLBACK *YourCallBackProc*(BYTE *pImgBuffer, BYTE *pVdoSrc, LPVOID lpParam)

pImgBuffer pointer of capture frame buffer,

pVdoSrc reserved,

lpParam pointer to the lpParam of this function

lpParam : Specifies the callback procedure parameter 3. You can pass useful pointer to your callback procedure.

nFrameCount : Specifies total frame number to capture. If the value is ALWAYS_CAPTURE, the capture frame is endless.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will define the callback procedure to capture the frame data of a channel of video.

See Also

Adv_VAPI_SetCaptureBuf, Adv_VAPI_SetCaptureFile, Adv_VAPI_SetCaptureWindow, Adv_VAPI_GetCaptureStatus

Adv_VAPI_GetCaptureStatus

Syntax

[C++]

DWORD Adv_VAPI_GetCaptureStatus(int nChannelNo)

[Delphi]

function Adv_VAPI_GetCaptureStatus(nChannelNo:Integer):Longword;

[VB]

function Adv_VAPI_GetCaptureStatus (ByVal nChannelNo As Integer) As Long

Parameters

nChannelNo : Specifies the video channel number on current physical port. Its range is 0~3.

Return Value

The value is DEST_NONE or the combination of the following values:

DEST_BUF : capture buffer function is not completed.

DEST_CALLBACK : capture callback function is not completed.

DEST_FILE : capture file function is not completed.

DEST_HWND : capture window function is not completed.

DEST_RUN : capture function for the port is still active..

Description

This function will return the capture status of a video channel.

See Also

Adv_VAPI_SetCaptureBuf, Adv_VAPI_SetCaptureFile, Adv_VAPI_SetCaptureCallback, Adv_VAPI_SetCaptureWindow

Adv_VAPI_SetCaptureSource

Syntax

[C++]

BOOL Adv_VAPI_SetCaptureSource(VIDEO_SOURCE nChannelNo)

[Delphi]

function Adv_VAPI_SetCaptureSource(nChannelNo:Integer):Boolean;

[VB]

function Adv_VAPI_SetCaptureSource (ByVal nChannelNo As Integer) As Boolean

Parameters

nChannelNo : Specifies the video channel number on current physical port. Its valid value is the following:

V_CVBS_0 : set video source as channel 0,
V_CVBS_1 : set video source as channel 1,
V_CVBS_2 : set video source as channel 2,
V_CVBS_3 : set video source as channel 3,
V_POLLING_ALL : set video source polling from channel 0 to 3

Return Value

TRUE : Function is successful
FALSE : Function is failed

Description

This function will assign a video source on the current physical port. The default video source is **V_POLLING_ALL**.

See Also

Adv_VAPI_CaptureStart, Adv_VAPI_CaptureStop

Adv_VAPI_GetCaptureSource

Syntax

[C++]

*BOOL Adv_VAPI_GetCaptureSource(VIDEO_SOURCE *pnChannelNo)*

[Delphi]

function Adv_VAPI_GetCaptureSource(var nChannelNo:Integer):Boolean;

[VB]

function Adv_VAPI_GetCaptureSource (ByRef pnChannelNo As Integer) As Boolean

Parameters

pnChannelNo : Specifies the return address of video channel number on current physical port. It will return a valid value as one of the following:

V_CVBS_0 : set video source as channel 0,

V_CVBS_1 : set video source as channel 1,

V_CVBS_2 : set video source as channel 2,

V_CVBS_3 : set video source as channel 3,

V_POLLING_ALL : set video source polling from channel 0 to 3

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will return a video source on the current physical port. The default video source is **V_POLLING_ALL**.

See Also

Adv_VAPI_CaptureStart, Adv_VAPI_CaptureStop

Adv_VAPI_CaptureStart

Syntax

[C++]

BOOL Adv_VAPI_CaptureStart(int nFrameCount)

[Delphi]

function Adv_VAPI_CaptureStart(nFrameCount:Integer):Boolean;

[VB]

function Adv_VAPI_CaptureStart (ByVal nFrameCount As Integer) As Boolean

Parameters

nFrameCount : Specifies total frame number to capture. If the value is ALWAYS_CAPTURE, the capture frame is endless.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will begin the capture function on current physical port.

See Also

Adv_VAPI_SetCaptureSource, Adv_VAPI_CaptureStop

Adv_VAPI_EnableVideoSource

Syntax

[C++]

BOOL Adv_VAPI_EnableVideoSource(VIDEO_SOURCE nChannelNo,BOOL bEnable);

[Delphi]

function Adv_VAPI_EnableVideoSource(nChannelNo:Integer; bEnable:Boolean):Boolean

[VB]

function Adv_VAPI_EnableVideoSource((ByVal nChannelNo As Integer, ByVal bEnable As Boolean) As Boolean

Parameters

nChannelNo : Specify the video source channel number. The valid value is one of the following:

V_CVBS_0	channel number 0,
V_CVBS_1	channel number 1,
V_CVBS_2	channel number 2,
V_CVBS_3	channel number 3

bEnable : Switch the video source on or off. If the value is TRUE, specified video source is on, otherwise is off.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will switch the video source channel on or off on current physical port.

Remark

This function is only for DVP-7010 card 0~3.

See Also

Adv_VAPI_CaptureStart, Adv_VAPI_SetCaptureSource

Adv_VAPI_GetVideoSignal

Syntax

[C++]

*BOOL Adv_VAPI_GetVideoSignal(BOOL *pbHasSignal)*

[Delphi]

function Adv_VAPI_GetVideoSignal(var pbHasSignal:Boolean):Boolean

[VB]

function Adv_VAPI_GetVideoSignal ((ByRef, pbHasSignal As Boolean) As Boolean

Parameters

pbHasSignal : Specify the pointer of returned video signal value. If the value id TRUE, the physical port has signal from input, otherwise the physical port has not signal from input

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will get the video signal from input on current physical port.

Remark

This function is only for DVP-7020 card 0~3, port 0~3.

See Also

Adv_VAPI_CaptureStart, Adv_VAPI_SetCaptureSource

Adv_VAPI_CaptureStop

Syntax

[C++]

BOOL Adv_VAPI_CaptureStop(void)

[Delphi]

function Adv_VAPI_CaptureStop:Boolean;

[VB]

function Adv_VAPI_CaptureStop () As Boolean

Parameters

None

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will stop the capture function on current physical port.

See Also

Adv_VAPI_CaptureStart, Adv_VAPI_SetCaptureSource

Adv_VAPI_GPIOInit

Syntax

[C++]

BOOL Adv_VAPI_GPIOInit(DWORD dwDirect)

[Delphi]

function Adv_VAPI_GPIOInit(dwDirect:Longword):Boolean;

[VB]

function Adv_VAPI_GPIOInit (ByVal dwDirect As Long) As Boolean

Parameters

dwDirect :	Specifies direction of GPIO pins. In the BT878, there are 16 GPIO pins. Every bit means a GPIO pin. If the bit is set to 1, the GPIO pin is set to output. If the bit is set to 0, the GPIO pin is set to input.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will initialize direction of the GPIO pins. For example,

Adv_VAPI_GPIO(0xff00)

set the bit 8~15 as output pins, bit 0~7 as input pins.

See Also

Adv_VAPI_SetGPIOData, Adv_VAPI_GetGPIOData

Adv_VAPI_GPIOGetDirect

Syntax

[C++]

*BOOL Adv_VAPI_GPIOGetDirect(DWORD *pdwDirect)*

[Delphi]

function Adv_VAPI_GPIOGetDirect(var dwDirect:Longword):Boolean;

[VB]

function Adv_VAPI_GPIOGetDirect (ByRef pdwDirect As Long) As Boolean

Parameters

pdwDirect : Specifies the address of returned direction of GPIO pins. In the BT878, there are 16 GPIO pins. Every bit means a GPIO pin. If the bit is set to 1, the GPIO pin is set to output. If the bit is set to 0, the GPIO pin is set to input.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will return direction of the GPIO pins.

See Also

Adv_VAPI_SetGPIOData, Adv_VAPI_GetGPIOData

Adv_VAPI_SetGPIOData

Syntax

[C++]

BOOL Adv_VAPI_SetGPIOData(DWORD dwValue)

[Delphi]

function Adv_VAPI_SetGPIOData (var dwValue:Longword):Boolean;

[VB]

function Adv_VAPI_SetGPIOData (ByVal dwValue As Long) As Boolean

Parameters

dwValue : Set the value of GPIO output pins.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will set the value of GPIO output pins. For example,

Adv_VAPI_GPIOInit(0xff00);

Adv_VAPI_SetGPIOData(0x00??)

set the bit 8~15 as output pins and their states are on low.

See Also

Adv_VAPI_GPIOInit, Adv_VAPI_GetGPIOData

Adv_VAPI_GetGPIOData

Syntax

[C++]

*BOOL Adv_VAPI_GetGPIOData(DWORD *pdwValue)*

[Delphi]

function Adv_VAPI_GetGPIOData(var dwValue:Longword):Boolean;

[VB]

function Adv_VAPI_GetGPIOData (ByRef pdwValue As Long) As Boolean

Parameters

pdwValue : Pointer to state value of GPIO input pins.

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will get the value of GPIO input pins. For example,

```
Adv_VAPI_GPIOInit(0xff00);
Adv_VAPI_GetGPIOData(&dwValue)
```

set the bit 0~7 as input pins and get their states.

See Also

Adv_VAPI_GPIOInit, Adv_VAPI_SetGPIOData

Adv_VAPI_SetWDT

Syntax

[C++]

BOOL Adv_VAPI_SetWDT(WDT_STATE nSet)

[Delphi]

function Adv_VAPI_SetWDT(nSet:Integer):Boolean;

[VB]

function Adv_VAPI_SetWDT (ByVal nSet As Integer) As Boolean

Parameters

nSet : Specify the watchdog state. Its value is as follows:

WDT_START	: Start the watchdog function
WDT_CLEAR	: Trigger the watchdog signal
WDT_STOP	: Stop the watchdog function

Return Value

TRUE : Function is successful

FALSE : Function is failed

Description

This function will set watchdog state. If you set the watchdog function to start, you must call clear function in every 5 seconds until you call stop function.

Adv_VAPI_ErrorMsg

Syntax

[C++]

void Adv_VAPI_ErrorMsg(HWND hWnd)

[Delphi]

procedure Adv_VAPI_ErrorMsg(hWwnd:HWND);

[VB]

sub Adv_VAPI_ErrorMsg (ByVal Hwnd As Long)

Parameters

hWnd : Specify parent window of the display message window.

Return Value

None

Description

This function will display the DialogBox to show error message.

See Also

Adv_VAPI_GetLastError

Adv_VAPI_GetLastError

Syntax

[C++]

DWORD Adv_VAPI_GetLastError(void)

[Delphi]

function Adv_VAPI_GetLastError:Longword;

[VB]

function Adv_VAPI_GetLastError () As Long

Parameters

None

Return Value

Return the last error code if function call failed. The error code is one of the following:

ERROR_VAPI_SUCCESS
ERROR_VAPI_UNINITIALIZE
ERROR_VAPI_BINDTOFILTER
ERROR_VAPI_SYSTEMDEVICEENUM
ERROR_VAPI_CREATECLASSENUM
ERROR_VAPI_NODEVICEFORCAPTURE
ERROR_VAPI_NOMATCHFILTER
ERROR_VAPI_COINITIALIZE
ERROR_VAPI_INITIALIZEPV
ERROR_VAPI_UNINTIALIZE
ERROR_VAPI_FILTERGRAPH
ERROR_VAPI_CAPTUREGRAPHBUILDER
ERROR_VAPI_CALLBACKRENDER
ERROR_VAPI_FILTERADDCALLBACKRENDER
ERROR_VAPI_QUERYCALLBACKRENDER
ERROR_VAPI_QUERYMEDIACONTROL
ERROR_VAPI_QUERYMEDIAEVENT
ERROR_VAPI_SETFILTERGRAPH
ERROR_VAPI_PORTID
ERROR_FAILED
ERROR_VAPI_DEVICECHANGED
ERROR_VAPI_DEVICEUSED
ERROR_VAPI_FINDMEDIAANALOG
ERROR_VAPI_FINDMEDIASTREAM

ERROR_VAPI_ADDFILTER
ERROR_VAPI_VDOUNINIT
ERROR_VAPI_GETTVFORMAT
ERROR_VAPI_TVFORMATUNSUPPORT
ERROR_VAPI_TVFORMATSETFAIL
ERROR_VAPI_VSCUNINIT
ERROR_VAPI_VSCGETFORMAT
ERROR_VAPI_MEDIACONTROLUNINIT
ERROR_VAPI_CARDID
ERROR_VAPI_CHANNELINVALID
ERROR_VAPI_MEDIACONTROLRUN
ERROR_VAPI_FILENAMEINVALID

Description

This function will return a error code if function call failed.

See Also

Adv_VAPI_ErrorMsg