

PCIe-FRM11 API Programming (Rev 1.2)



Windows, Windows2000, Windows NT and Windows XP are trademarks of **Microsoft**. We acknowledge that the trademarks or service names of all other organizations mentioned in this document as their own property.

Information furnished by DAQ system is believed to be accurate and reliable. However, no responsibility is assumed by DAQ system for its use, nor for any infringements of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or copyrights of DAQ system.

The information in this document is subject to change without notice and no part of this document may be copied or reproduced without the prior written consent.

Copyrights © 2005 DAQ system, All rights reserved.

Board Level APIs

Overview

int	OpenDAQDevice (void)
BOOL	ResetBoard (int nBoard)
BOOL	CloseDAQDevice (void)
int	GetBoardNum (void)

OpenDAQDevice

It opens a device. You may call this function at the very first time you run the program and some suspicious operation.

int OpenDAQDevice (void)

Parameters: None .

Return Value:

If the function succeeds, it returns the number of boards which were detected.

If the function fails, the return value is -1, it means there is no device in the system.

ResetBoard

It initializes a device at currently equipped system (PC).

BOOL ResetBoard (int nBoard)

Parameters:

nBoard : It informs a board number at currently equipped system.

The board number set up by DIP switch.

Return Value:

It returns TRUE in case of the success of reset and initialization.

If you get FALSE you should not call any API functions with the board and call the **CloseDAQDevice()** instead.

CloseDAQDevice

The CloseDAQDevice function closes all opened devices (boards). If use of device is finished, it can certainly close a device for making it other programs so as usable.

BOOL CloseDAQDevice (void)

Parameters: None.

Return Value:

If the function fail to close, it returns "FALSE".

If the function succeed to close, it returns "TRUE".

GetBoardNum

Returns currently detected board number in the system.

int GetBoardNum void)

Parameters: None

Return Value:

The number of boards, The Board number is set by dip switch.

LVDS APIs

Overview

BOOL	LVDS_Init (void)
BOOL	LVDS_Start (void)
BOOL	LVDS_Check(void)
BOOL	LVDS_GetFrame (DWORD* nCnt, unsigned char* buf)
BOOL	LVDS_Close (void)
BOOL	LVDS_SetModel (int Model)
BOOL	LVDS_GetModel (int *Model)
BOOL	LVDS_SetResolution (DWORD xRes, DWORD yRes)
BOOL	LVDS_GetResolution (DWORD *xRes, DWORD *yRes)
BOOL	LVDS_Stop (void)
BOOL	LVDS_SetDataMode (int nMode)
BOOL	LVDS_SetFilter (DWORD dwValue)
BOOL	LVDS_GetFilter (DWORD *dwValue)
BOOL	LVDS_SetHsCount (int nCount)
BOOL	LVDS_GetHsCount (int *nCount)
BOOL	LVDS_GetVersion (int *nVersion)
BOOL	LVDS_GetError (int *nHZ, int *nResX, int *nResY)

LVDS_Init

Initialize resources used for the LVDS sub-system, for example interrupt and LVDS control register.

BOOL **LVDS_Init (void)**

Parameters: None.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_Start

Start receiving frame data.

BOOL **LVDS_Start (void)**

Parameters: None.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_Check

Check out the frame data acquisition has been completed.

BOOL **LVDS_Check (void)**

Parameters: none.

Return Value:

If the frame data has been completed, It returns "True".

If the frame data has not been completed. It returns "FALSE"

LVDS_GetFrame

Get Frame data from the frame buffer.

BOOL **LVDS_GetFrame (DWORD* nCnt, unsigned char* buf)**

Parameters:

nCnt : It is the address which contains the number of data to be received in byte size. Specifies the size buffer when the function is called, and read the values of the variables after a call to find out how many actually read. The data size is in bytes.

buf : Frame buffer pointer.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, check the values of the size that you want to read nCnt.

LVDS_Close

Release all resource were used for LVDS function.

BOOL LVDS_Close (void)

Parameters: None.

Return Value :

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_SetModel

Set up the Camera model for LVDS interface.

BOOL LVDS_SetModel (int Model)

Parameters:

Model : Model number '0' selects 2048 x 1560 resolution Camera,
otherwise selects 3160 x 2560 resolution Camera.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_GetModel

Get a setup value of the Camera model for LVDS interface.

BOOL LVDS_GetModel (int *Model)

Parameters:

Model: Address value for camera model number

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_SetResolution

Set camera resolution for the specific camera Model.

BOOL **LVDS_SetResolution (DWORD xRes, DWORD yRes)**

Parameters:

xRes : Value of the horizontal Camera resolution

yRes : Value of the vertical Camera resolution

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_GetResolution

Gets currently configured camera's frame resolution

BOOL **LVDS_GetResolution (DWORD *xRes, DWORD *yRes)**

Parameters:

xRes : Address pointer to receive horizontal Camera resolution

yRes : Address pointer to receive vertical Camera resolution

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_Stop

Stop the frame data capture.

BOOL **LVDS_Stop (void)**

Parameters: None.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_SetDataMode

Set image pixel data mode.

BOOL LVDS_SetDataMode (int nMode)

Parameters:

nMode : If the value is 2, the pixel data be expressed by 24bits, others be 16bits.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_SetFilter

The normal state is not occurred, but if the cable is missed, if the broken camera occurs signal without Vertical Synchronization signal, the system will be damaged from this occasion. In this case, set the hardware filter Vsync signal can eliminate the abnormal signal.

BOOL LVDS_SetFilter (DWORD dwValue)

Parameters:

dwValue : The valid range of values is from 0 to 65535, the unit is about 15nSEC. The default value is 160. If you set up a filter, you should keep in mind a Front porch and Back porch.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_GetFilter

Get the Vsync filter current value.

BOOL LVDS_GetFilter (DWORD *dwValue)

Parameters:

dwValue: Valuable address of set filter values.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_SetHsCount

Set the value of a Horizontal sync count for image checking.

BOOL LVDS_SetHsCount (int nCount)

Parameters:

nCount: The minimum value of the frame's Horizontal sync.
The frame is useless smaller than this value.

Return Value:

If the function call fails, it returns "FALSE".
If the function call succeeds, it returns "TRUE".

LVDS_GetHsCount

Get the value of a Horizontal sync count.

BOOL LVDS_GetHsCount (int *nCount)

Parameters:

*nCount: Valuable of Horizontal count.

Return Value:

If the function call fails, it returns "FALSE".
If the function call succeeds, it returns "TRUE".

LVDS_GetVersion

Get FPGA version.

BOOL LVDS_GetVersion (int *nVersion)

Parameters:

nVersion : The pointer of the FPGA version. Some API are only supported by boards which have the FPGA version number 2 or more.

Return Value:

If the function call fails, it returns "FALSE".
If the function call succeeds, it returns "TRUE".

LVDS_GetError

Get the error state.

BOOL **LVDS_GetError (int *nHZ, int *nResX, int *nResY)**

Parameters:

nHZ : real Pixel frequency to check with measured value
nResX : real Horizontal resolution to check with measured value
nResY : real Vertical resolution to check with measured value

Return Value :

0 : OK.
If bit0 is '1' : PCLK error
If bit1 is '1' : HSYNC error
If bit2 is '1' : VSYNC error

UART APIs

Overview

BOOL	UART_Init (void)
BOOL	UART_GetData (DWORD* nCnt, unsigned char* buf)
BOOL	UART_SendData (DWORD* nCnt, unsigned char* buf)
BOOL	UART_Close (void)
BOOL	UART_SetBaud (DWORD nBaud)
BOOL	UART_BufferFlush (void)

UART_Init

Initialize resources used for the UART sub-system, for example interrupt and UART control register.

BOOL **UART_Init (void)**

Parameters: None.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

UART_GetData

Receive characters through the differential UART.

BOOL **UART_GetData (DWORD* nCnt, unsigned char* buf)**

Parameters:

nCnt : The address which contains the number of characters to be received.

The maximum number of characters to be received is limited to 4Kbyte(4096).

buf : The buffer address.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

UART_SendData

Sent characters through the differential UART.

BOOL UART_SendData (DWORD* nCnt, unsigned char* buf)

Parameters:

nCnt : The address which contains the number of characters to be sent.

The maximum number of characters to be sent is limited to 4K byte(4096).

buf : The buffer address.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

UART_Close

Release all resource were used for UART function.

BOOL UART_Close (void)

Parameters: None.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

UART_SetBaud

Set the UART Baud.

BOOL UART_SetBaud (DWORD nBaud)

Parameters:

nBaud : 0: 9600, 1: 19200, 2: 38400, 3:57600, 4:115200

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

UART_BufferFlush

Clear the UART Rx buffer.

BOOL **UART_BufferFlush (void)**

Parameters: None.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

DIO(Digital Input Output) API Functions

Overview

BOOL **DIO_Read (void)**

BOOL **DIO_Write (DWORD val)**

DIO_Read

Read the Digital Input state.

BOOL **DIO_Read (void)**

Parameters: None.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

DIO_Write

Set up the Digital Output pin state.

BOOL **DIO_Write (DWORD val)**

Parameters:

val : The value to be written to the port.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

Multi Board support APIs

Notes) All single board API can be used with the system which have only one board installed, but multi board APIs must be used with the system which have more than two boards installed.

Multi board API can be used with the board which have the FPGA version #2 or more.

LVDS(DVI) APIs

Overview

BOOL	LVDS_Init_Mul (int nBoard)
BOOL	LVDS_Start_Mul (int nBoard)
BOOL	LVDS_Check_Mul (int nBoard)
BOOL	LVDS_GetFrame_Mul (int nBoard, DWORD* nCnt, unsigned char* buf)
BOOL	LVDS_Close_Mul (int nBoard)
BOOL	LVDS_SetModel_Mul (int nBoard, int Model)
BOOL	LVDS_GetModel_Mul (int nBoard, int *Model)
BOOL	LVDS_SetResolution_Mul (int nBoard, DWORD xRes, DWORD yRes)
BOOL	LVDS_GetResolution_Mul (int nBoard, DWORD *xRes, DWORD *yRes)
BOOL	LVDS_Stop_Mul (int nBoard)
BOOL	LVDS_SetDataMode_Mul (int nBoard, int nMode)
BOOL	LVDS_SetFilter_Mul (int nBoard, DWORD dwValue)
BOOL	LVDS_GetFilter_Mul (int nBoard, DWORD *dwValue)
BOOL	LVDS_SetHsCount_Mul (int nBoard, int nCount)
BOOL	LVDS_GetHsCount_Mul (int nBoard, int *nCount)
BOOL	LVDS_GetVersion_Mul (int nBoard, int *nVersion)
BOOL	LVDS_GetError_Mul (int nBoard, int *nHZ, int *nResX, int *nResY)

LVDS_Init_Mul

Initialize resources used for the LVDS sub-system, for example interrupt and LVDS control register.

BOOL **LVDS_Init_Mul (int nBoard)**

Parameters:

nBoard : It informs a board number at currently equipped system.
The board number set up by DIP switch.

Return Value:

If the function call fails, it returns "FALSE".
If the function call succeeds, it returns "TRUE".

LVDS_Start_Mul

Start receiving frame data.

BOOL **LVDS_Start_Mul (int nBoard)**

Parameters:

nBoard : It informs a board number at currently equipped system.
The board number set up by DIP switch.

Return Value:

If the function call fails, it returns "FALSE".
If the function call succeeds, it returns "TRUE".

LVDS_Check_Mul

Check out the frame data acquisition has been completed.

BOOL **LVDS_Check_Mul (int nBoard)**

Parameters:

nBoard : It informs a board number at currently equipped system.
The board number set up by DIP switch.

Return Value:

If the frame data has been completed, It returns "True".
If the frame data has not been completed. It returns "FALSE"

LVDS_GetFrame_Mul

Get Frame data from the frame buffer.

BOOL **LVDS_GetFrame_Mul (int nBoard, DWORD* nCnt, unsigned char* buf)**

Parameters:

nBoard : It informs a board number at currently equipped system.

The board number set up by DIP switch.

nCnt : It is the address which contains the number of data to be received in byte size. Specifies the size buffer when the function is called, and read the values of the variables after a call to find out how many actually read. The data size is in bytes.

buf : The buffer address.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, check the values of the size that you want to read nCnt.

LVDS_Close_Mul

Release all resource were used for LVDS function.

BOOL **LVDS_Close_Mul (int nBoard)**

Parameters:

nBoard : It informs a board number at currently equipped system.

The board number set up by DIP switch.

Return Value :

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_SetModel_Mul

Set up the Camera model for LVDS interface.

BOOL **LVDS_SetModel_Mul (int nBoard, int Model)**

Parameters:

nBoard : It informs a board number at currently equipped system.

The board number set up by DIP switch.

Model : Model number '0' selects 2048 x 1560 resolution Camera,

otherwise selects 3160 x 2560 resolution Camera.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_GetModel_Mul

Get a setup value of the Camera model for LVDS interface.

BOOL LVDS_GetModel_Mul (int nBoard, int *Model)

Parameters:

nBoard : It informs a board number at currently equipped system.

The board number set up by DIP switch.

Mode I: Address value for camera model number

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_SetResolution_Mul

Set camera resolution for the specific camera Model.

BOOL LVDS_SetResolution_Mul (int nBoard, DWORD xRes, DWORD yRes)

Parameters:

nBoard : It informs a board number at currently equipped system.

The board number set up by DIP switch.

xRes : Value of the horizontal Camera resolution

yRes : Value of the vertical Camera resolution

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_GetResolution_Mul

Gets currently configured camera's frame resolution

BOOL **LVDS_GetResolution_Mul (int nBoard, DWORD *xRes, DWORD *yRes)**

Parameters:

nBoard : It informs a board number at currently equipped system.

The board number set up by DIP switch.

xRes : Address pointer to receive horizontal Camera resolution

yRes : Address pointer to receive vertical Camera resolution

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_Stop_Mul

Stop the frame data capture.

BOOL **LVDS_Stop_Mul (int nBoard)**

Parameters:

nBoard : It informs a board number at currently equipped system.

The board number set up by DIP switch.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_SetDataMode_Mul

Set image pixel data mode.

BOOL **LVDS_SetDataMode_Mul (int nBoard, int nMode)**

Parameters:

nBoard : It informs a board number at currently equipped system.

The board number set up by DIP switch.

nMode : If the value is 2, the pixel data be expressed by 24bits, others be 16bits.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_SetFilter_Mul

The normal state is not occurred, but if the cable is missed, if the broken camera occurs signal without Vertical Synchronization signal, the system will be damaged from this occasion. In this case, set the hardware filter Vsync signal can eliminate the abnormal signal.

BOOL LVDS_SetFilter_Mul (int nBoard, DWORD dwValue)

Parameters:

nBoard : It informs a board number at currently equipped system.

The board number set up by DIP switch.

dwValue : The valid range of values is from 0 to 65535, the unit is about 15nSEC. The default value is 160. If you set up a filter, you should keep in mind a Front porch and Back porch.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_GetFilter_Mul

Get the Vsync filter current value.

BOOL LVDS_GetFilter_Mul (int nBoard, DWORD *dwValue)

Parameters:

nBoard : It informs a board number at currently equipped system.

The board number set up by DIP switch.

dwValue: Valuable address of set filter values.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_SetHsCount_Mul

Set the value of a Horizontal sync count for image checking.

BOOL LVDS_SetHsCount_Mul (int nBoard, int nCount)

Parameters:

nBoard : It informs a board number at currently equipped system.

The board number set up by DIP switch.

nCount: The minimum value of the frame's Horizontal sync.

The frame is useless smaller than this value.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_GetHsCount_Mul

Get the value of a Horizontal sync count.

BOOL LVDS_GetHsCount_Mul (int nBoard, int *nCount)

Parameters:

nBoard : It informs a board number at currently equipped system.

The board number set up by DIP switch.

*nCount: Valuable address of Horizontal count.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

LVDS_GetVersion_Mul

Get FPGA version.

BOOL LVDS_GetVersion_Mul (int nBoard, int *nVersion)

Parameters:

nBoard : It informs a board number at currently equipped system.

The board number set up by DIP switch.

nVersion : The pointer of the FPGA version.

Some API are only supported by boards which have the FPGA version number 2 or more.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

If the function call succeeds, it returns "TRUE".

LVDS_GetError_Mul

Get the error state.

BOOL **LVDS_GetError_Mul** (int nBoard, int *nHZ, int *nResX, int *nResY)

Parameters:

nBoard : It informs a board number at currently equipped system.

 The board number set up by DIP switch.

nHZ : real Pixel frequency to check with measured value

nResX : real Horizontal resolution to check with measured value

nResY : real Vertical resolution to check with measured value

Return Value :

0 : OK.

If bit0 is '1' : PCLK error

If bit1 is '1' : HSYNC error

If bit2 is '1' : VSYNC error

UART APIs

Overview

BOOL	UART_Init_Mul (int nBoard)
BOOL	UART_GetData_Mul (int nBoard, DWORD* nCnt, unsigned char* buf)
BOOL	UART_SendData_Mul (int nBoard, DWORD* nCnt, unsigned char* buf)
BOOL	UART_Close_Mul (int nBoard)
BOOL	UART_SetBaud_Mul (int nBoard, DWORD nBaud)
BOOL	UART_BufferFlush_Mul (int nBoard)

UART_Init_Mul

Initialize resources used for the UART sub-system, for example interrupt and UART control register.

BOOL UART_Init_Mul (int nBoard)

Parameters:

nBoard : The Board number is set by dip switch.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

UART_GetData_Mul

Receive characters through the differential UART.

BOOL UART_GetData_Mul (int nBoard, DWORD* nCnt, unsigned char* buf)

Parameters:

nBoard : The Board number is set by dip switch.

nCnt : The address which contains the number of characters to be received.

The maximum number of characters to be received is limited to 4Kbyte(4096).

buf : The buffer address.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

UART_SendData_Mul

Sent characters through the differential UART.

BOOL UART_SendData_Mul (int nBoard, DWORD* nCnt, unsigned char* buf)

Parameters:

nBoard : The Board number is set by dip switch.

nCnt : The address which contains the number of characters to be sent.

The maximum number of characters to be sent is limited to 4K byte(4096).

buf : The buffer address.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

UART_Close_Mul

Release all resource were used for UART function.

BOOL UART_Close_Mul (int nBoard)

Parameters:

nBoard : The Board number is set by dip switch.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

UART_SetBaud_Mul

Sets UART Baud rates

BOOL UART_SetBaud_Mul (int nBoard, DWORD nBaud)

Parameters:

nBoard : The Board number is set by dip switch.

nBaud : 0: 9600, 1: 19200, 2: 38400, 3:57600, 4: 115200

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

UART_BufferFlush_Mul

Flushes UART RX Buffer

BOOL **UART_BufferFlush_Mul (int nBoard)**

Parameters:

nBoard : The Board number is set by dip switch.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

DIO(Digital Input Output) API Functions

Overview

DWORD **DIO_Read_Mul (int nBoard)**

BOOL **DIO_Write_Mul (int nBoard, DWORD val)**

DIO_Read_Mul

Reads from input port

DWORD **DIO_Read_Mul (int nBoard)**

Parameters:

nBoard : The Board number is set by dip switch.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".

DIO_Write_Mul

Writes to output port

BOOL **DIO_Write_Mul (int nBoard, DWORD val)**

Parameters:

nBoard : The Board number is set by dip switch.

val : The value to be written to the port.

Return Value:

If the function call fails, it returns "FALSE".

If the function call succeeds, it returns "TRUE".