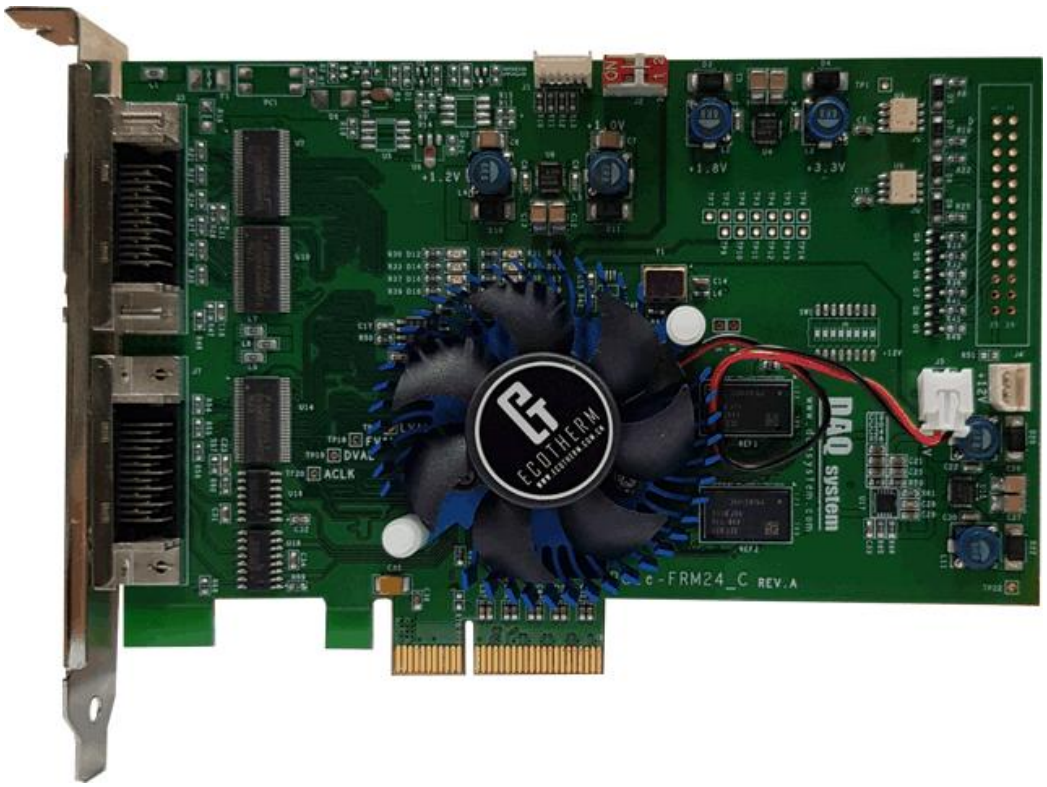
PCIe-FRM24_C API Programming (Rev 1.0)



Board Level APIs

Overview

int **OpenDAQDevice (void)**
BOOL **ResetBoard (int nBoard)**
BOOL **CloseDAQDevice (void)**
int **GetBoardNum (void)**
char* **GetDllVersion(void)**

OpenDAQDevice

This function initializes the device. You may call this function at the very first time you run the program.

BOOL **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

This function initializes a device at currently equipped system (PC).

BOOL **ResetBoard (int nBoard)**

Parameters:

nBoard : Numbers of discovered device. The board number is 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

This function closes all opened devices (boards). If using of device is finished, you must 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

This function 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.

GetDllVersion

This function notices a DLL version.

Char* GetDllVersion (void)

Parameters: None.

Return Value:

The date of the DLL is returned.

.

LVDS(Camera Link) APIs

Overview

BOOL **LVDS_GetVersion (int nBoard, int *nVersion)**

BOOL **LVDS_Init (int nBoard)**

BOOL **LVDS_Close (int nBoard)**

BOOL **LVDS_Start (int nBoard)**

BOOL **LVDS_Stop (int nBoard)**

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

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

BOOL **LVDS_GetFrameRate (int nBoard, DWORD* nFrameRate)**

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

BOOL **LVDS_SetHsPol (int nBoard, BOOL bInv)**

BOOL **LVDS_SetVsPol (int nBoard, BOOL bInv)**

BOOL **LVDS_SetDeVal (int nBoard, BOOL bUse)**

BOOL **LVDS_DdrEnable (int nBoard, BOOL bEn)**

BOOL **LVDS_CC_Configure (int nBoard, int nCCcfg)**

BOOL **LVDS_CC_Output (int nBoard, int nCCdata)**

BOOL **LVDS_ConfigureTrig1 (int nBoard, DWORD dwEvent, DWORD dwDelay, DWORD dwWidth, DWORD dwBlank)**

BOOL **LVDS_ConfigureTrig2 (int nBoard, DWORD dwEvent, DWORD dwDelay, DWORD dwWidth, DWORD dwBlank)**

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

BOOL **LVDS_SetPageStart (int nBoard, DWORD dwEvent)**

BOOL **LVDS_SetLineCount (int nBoard, DWORD dwCount)**

BOOL **LVDS_SetReferrenceClock (int nBoard, int nClock)**

LVDS_GetVersion

This function gets the version of the current program.

BOOL LVDS_GetVersion (int nBoard, int *nVersion)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

*nVersion : Current program version value.

Return Value:

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

LVDS_Init

This function initializes all resources for the LVDS sub-system, for example Interrupt and LVDS control register.

BOOL LVDS_Init (int nBoard)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

Return Value:

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

LVDS_Close

This function releases all resources that used for LVDS function.

At the end of the program, the application program calls this function.

BOOL LVDS_Close (int nBoard)

Parameters:

nBoard : Numbers of discovered device. The board number is 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

This function starts receiving frame data. After calling this function, by calling LVDS_GetFrame() function can be checked the complete data.

BOOL LVDS_Start (int nBoard)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

Return Value:

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

LVDS_Stop

This function stops the frame data capture.

BOOL LVDS_Stop (int nBoard)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

Return Value:

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

LVDS_GetResolution

This function gets currently configured camera’s frame resolution

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

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

*xRes : Horizontal resolution. It gets the width of the frame.

*yRes : Vertical resolution. It gets the height of the frame.

Return Value:

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

LVDS_GetFrame

This function acquires image data from the frame buffer.

The size of the buffer to receive the data should be informed.

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

Parameters:

nBoard : Numbers of discovered device. The board number is 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 : Pointer of first pixel of image data.

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_GetFrameRate

This function gets the actual frame rate from the sensor or camera.

BOOL LVDS_GetFrameRate (int nBoard, DWORD *nFrameRate)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

* nFrameRate : Actual Frame Rate value.

Return Value:

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

LVDS_SetDataMode

This function sets the image pixel data mode.

BOOL LVDS_SetDataMode (int nBoard, int nMode)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

nMode : “0” is 8bit Mode, “1” is 16bit Mode, "2" means 32bit Mode,
and "3" means 64bit Mode. If it is “4”, it is 80bit mode.

Return Value:

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

LVDS_SetHsPol

This function sets a normal or inverse HSYNC(Horizontal Synchronization) signal.

BOOL LVDS_SetHsPol (int nBoard, BOOL bInv)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

bInv : If the value is “0”, use the Normal HSYNC signal,
If the value is “1”, use the Inverse HSYNC signal.

Return Value :

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

LVDS_SetVsPol

This function sets a normal or inverse HSYNC(Horizontal Synchronization) signal.

BOOL LVDS_SetVsPol (int nBoard, BOOL bInv)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

bInv : If the value is “0”, use the Normal HSYNC signal,
If the value is “1”, use the Inverse HSYNC signal.

Return Value :

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

LVDS_SetDeVal

This function sets a Data Valid signal.

BOOL LVDS_SetDeVal (int nBoard, BOOL bUse)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

bUse : If the value is “1”, use the DVAL(Data Validation).

If the value is “0”, use the HSYNC (Horizontal Synchronization).

Return Value :

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

LVDS_DdrEnable

This function selects the using DDR Memory.

BOOL LVDS_DdrEnable (int nBoard, BOOL bEn)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

bEn : If the value is “1”, use the DDR memory.

If the value is “0”, don’t use the DDR memory.

Return Value :

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

LVDS_ConfigureCc

This function sets a configure value of the corresponding bits(CC1 ~ CC4).

BOOL LVDS_ConfigureCc (int nBoard, DWORD nCCcfg)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

nCCcfg : bit0(CC1 configure) ➔ "0" : digital out1 / "1" : Trigger1 output

bit1(CC2 configure) ➔ "0" : digital out2 / "1" : Trigger2 output

bit2(CC3 configure) ➔ "0" : digital out3 / "1" : Digital output

bit3(CC4 configure)➔ "0" : digital out4 / "1" : Reference clock output

others : Reserved

Return Value :

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

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

LVDS_CcOutput

This function outputs a CC1 ~ CC4 value.

BOOL LVDS_CcOutput (int nBoard, DWORD nCCdata)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

nCCData : bit0 = CC1 out, bit1 = CC2 out

bit2 = CC3 out, bit3 = CC4 out, others : Reserved

Return Value :

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

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

LVDS_ConfigureTrig1

This function configures a Trigger 1.

BOOL LVDS_ConfigureTrig1 (int nBoard, DWORD dwEvent, DWORD dwDelay, DWORD dwWidth, DWORD dwBlank)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

dwEvent : “0” : Continuous (Free running without any condition)

“1” : Rising edge on Line Trigger input

“2” : Rising edge on encoder A phase

“3” : Rising edge on encoder B phase

“4” : Rising edge encoder up clock using A and B phase

“5” : Rising edge encoder down clock A and B phase

dwDelay : How many reference clock need to be waited before to output trigger1 high.

dwWidth : How many reference clock need to output trigger1 high.

dwBlank : How many reference clock need to be waited before return to event wait state.

Return Value :

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

LVDS_ConfigureTrig2

This function configures a Trigger 2.

BOOL LVDS_ConfigureTrig2 (int nBoard, DWORD dwEvent, DWORD dwDelay, DWORD dwWidth, DWORD dwBlank)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

dwEvent : "0" : Continuous (Free running without any condition)
"1" : Rising edge on Line Trigger input
"2" : Rising edge on encoder A phase
"3" : Rising edge on encoder B phase
"4" : Rising edge encoder up clock using A and B phase
"5" : Rising edge encoder down clock A and B phase

dwDelay : How many reference clock need to be waited before to output trigger2 high.

dwWidth : How many reference clock need to output trigger2 high.

dwBlank : How many reference clock need to be waited before return to event wait state.

Return Value :

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

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

LVDS_CameraMode

This function selects the Camera Trigger Mode.

BOOL LVDS_CameraMode (int nBoard, int nMode)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

nMode : If the value is "0", it is an Area Mode(default).
If the value is "1", it is a Line Mode(Free Run).
If the value is "2", it is a Line Mode(Trigger from external port).
If the value is "3", it is a Line Mode(Internal Clock 33MHz with Trigger).

Return Value :

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

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

LVDS_SetPageStart

This function sets a Page Start Event.

BOOL LVDS_SetPageStart (int nBoard, DWORD dwEvent)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

dwEvent : “0” : Continuous (Free Running without any condition)

“1” : Rising edge in Page Trigger input

“2” : Rising edge on encoder z phase

“Others” : Reserved

Return Value :

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

LVDS_SetLineCount

This function sets a Line Count.

BOOL LVDS_SetLineCount (int nBoard, DWORD dwCount)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

dwCount : How many lines need to be acquired one image acquisition as a page.

Range 1 to 65535.

Return Value :

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

LVDS_SetReferenceClock

This function sets an Internal or External clock.

BOOL LVDS_SetReferenceClock (int nBoard, int nClcok)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

nClock : If the value is “0”, sets an Internal Clock(Default).

If the value is “Others”, sets an External Clock.

Return Value :

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

UART APIs

Overview

BOOL **UART_Init (int nBoard)**

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

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

BOOL **UART_Close (int nBoard)**

BOOL **UART_SetBaud (int nBoard, DWORD nBaud)**

BOOL **UART_BufferFlush (int nBoard)**

UART_Init

This function initializes all resources that used for the UART sub-system, for example Interrupt and UART control register.

BOOL **UART_Init (nBoard)**

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

Return Value:

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

UART_GetData

This function receives the characters through the UART.

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

Parameters:

nBoard : Numbers of discovered device. The board number is set up 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 1K byte.

buf : The buffer address.

Return Value:

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

UART _SendData

This function sends the characters through the UART.

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

Parameters:

nBoard : Numbers of discovered device. The board number is set up 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 1K byte.

buf : The buffer address.

Return Value:

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

UART _Close

This function releases all resource that used for UART function.

BOOL UART_Close (nBoard)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

Return Value:

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.

UART_SetBaud

This function sets an UART Baud.

BOOL UART_SetBaud (int nBoard, DWORD nBaud)

Parameters:

nBoard : Numbers of discovered device. The board number is set up 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

This function clears an UART Rx buffer.

BOOL UART_BufferFlush (int nBoard)

Parameters:

nBoard : Numbers of discovered device. The board number is set up by DIP switch.

Return Value:

If the function call fails, it returns “FALSE”.

If the function call succeeds, it returns “TRUE”.