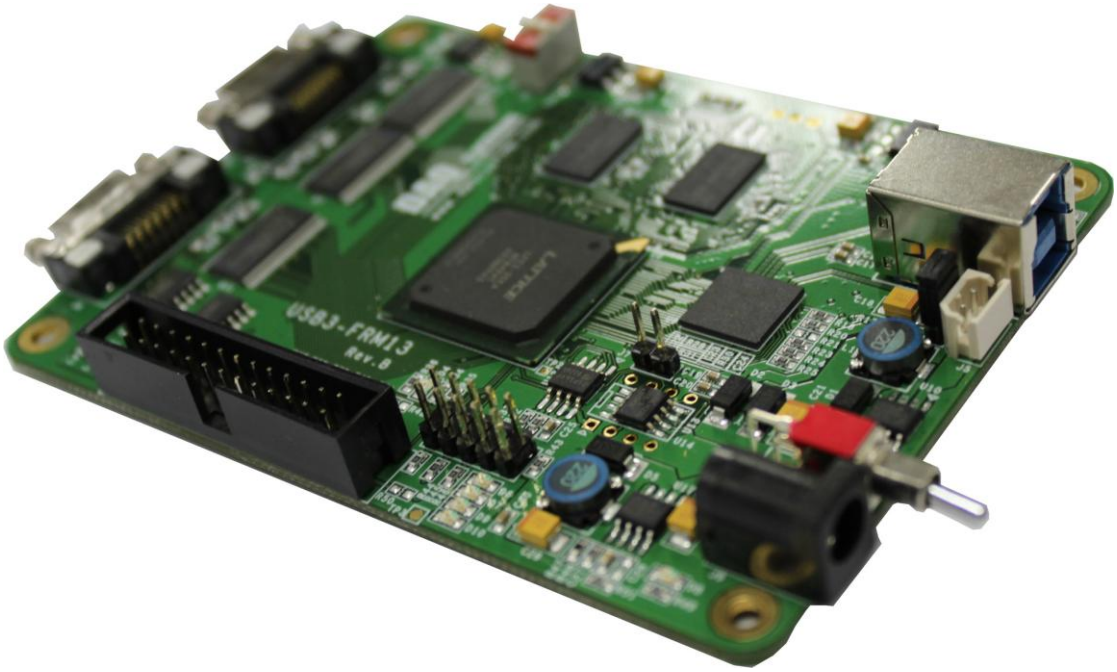
USB3-FRM13_K API Programming (Rev 1.0)



Board Level APIs

Overview

int **OpenDAQDevice (void)**
BOOL **ResetBoard (int nBoard)**
BOOL **CloseDAQDevice (void)**
int **GetBoardNum (void)**
BOOL **IsConnected(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.
(In the case of multi-board up to a max. 4)
If the function fails, the return value is 0, 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 after open device.

int GetBoardNum (void)

Parameters: None

Return Value:

Returns the number of boards installed.

(In the case of multi-board up to a max. 4)

IsConnected

This function confirms that the board is connected to USB.

BOOL IsConnected (void)

Parameters: None

Return Value:

If the function fail to close, it returns “FALSE”.

If the function succeed to close, it returns “TRUE”.

LVDS(Camera Link) APIs

Overview

BOOL	LVDS_Init (void)
BOOL	LVDS_Start (void)
BOOL	LVDS_GetFrame (DWORD* nCnt, unsigned char* buf)
BOOL	LVDS_Close (void)
BOOL	LVDS_GetResolution (DWORD *xRes, DWORD *yRes)
BOOL	LVDS_Stop (void)
BOOL	LVDS_SetDataMode (int nMode)
BOOL	LVDS_GetVersion (int *nFpgaVer, int *nFirmVer)
DWORD	LVDS_GetError (DWORD *dwStatuse)
BOOL	LVDS_BufferFlush (void)
BOOL	LVDS_SetDeUse (BOOL bUse)
BOOL	LVDS_SetHsPol (BOOL bPol)
BOOL	LVDS_CameraMode (int nMode)
BOOL	LVDS_SetReferrenceClock (int nClock)
BOOL	LVDS_ConfigureCc (DWORD dwCFG)
BOOL	LVDS_CcOutput (DWORD dwCC)
BOOL	LVDS_SetLineCount (DWORD dwCount)
BOOL	LVDS_SetPageStart (DWORD dwEvent)
BOOL	LVDS_SetPageDelay (DWORD dwCount)
BOOL	LVDS_ConfigureTrig1 (DWORD dwEvent, DWORD dwDelay, DWORD dwWidth, DWORD dwBlank)
BOOL	LVDS_ConfigureTrig2 (DWORD dwEvent, DWORD dwDelay, DWORD dwWidth, DWORD dwBlank)
BOOL	LVDS_SetRolling (BOOL bRolling)
BOOL	LVDS_SetPoCLDelay (int nMode)
BOOL	LVDS_ExtTrigEnable (BOOL bEn)
BOOL	LVDS_ExtTrigInv (BOOL bInv)
BOOL	LVDS_ExtTriConfigure (DWORD dwDelay, DWORD dwWidth, DWORD dwBlank)
BOOL	LVDS_EncAInv (BOOL bInv)
BOOL	LVDS_EncBInv (BOOL bInv)
BOOL	LVDS_EncZInv (BOOL bInv)
BOOL	LVDS_LineTrigInv (BOOL bInv)
BOOL	LVDS_PhaseTrigInv (BOOL bInv)

BOOL LVDS_TrigOutInv (int nBoard, int nInv)

LVDS_Init

This function initializes all resources 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

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

Caution) If it is not complete Frame data and return FALSE, returns with nCnt value 0.

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 (void)

Parameters: None.

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 (DWORD *xRes, DWORD *yRes)

Parameters:

*xRes : Width of image in pixels.

*yRes : Height of Image in pixels.

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 (void)

Parameters: None.

Return Value:

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

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

LVDS_SetDataMode

This function sets an image pixel data mode.

BOOL LVDS_SetDataMode (int nMode)

Parameters:

nMode : "0" : 8bit Mode, "1" : 16bit Mode

"2" : 32bit Mode, "3" : 64bit Mode

Return Value:

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

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

LVDS_GetVersion

This function gets a current FPGA version.

BOOL LVDS_GetVersion (int *nVersion)

Parameters:

*nVersion : FPGA version.

Return Value:

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

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

LVDS_GetError

This function gets a Frame (Image) Error.

DWORD LVDS_GetError (DWORD *dwStatus)

Parameters:

*dwStatus : Error status.

“1” : Overflow error

“2” : Read error

“4” : Size error

Return Value :

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

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

LVDS_BufferFlush

This function initializes the buffer.

BOOL LVDS_BufferFlush (void)

Parameters: None.

Return Value :

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

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

LVDS_SetDeUse

This function sets a Data Valid or Horizontal Synchronization signal.

BOOL LVDS_SetDUse (BOOL bUse)

Parameters:

bUse : If the value is “True”, use the DVAL(Data Validation).
If the value is “False”, use the HSYNC (Horizontal Synchronization).

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 (BOOL bPol)

Parameters:

bPol : If the value is “True”, use the Normal HSYNC signal,
If the value is “False”, use the Inverse HSYNC signal.

Return Value :

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

LVDS_CameraMode

This function sets a Camera Operation Mode.

BOOL LVDS_CameraMode (int nMode)

Parameters:

nMode : If the value is “0”, it is an Area Scan Camera mode(Default).
If the value is “Others”, it is a Line Scan Camera mode.

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 nClcok)

Parameters:

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”.

LVDS_ConfigureCc

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

BOOL LVDS_ConfigureCc (DWORD dwCFG)

Parameters:

dwCFG : 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 (DWORD dwCC)

Parameters:

dwCc : 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_SetLineCount

This function sets a Line Count.

BOOL LVDS_SetLineCount (DWORD dwCount)

Parameters:

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_SetPageStart

This function sets a Page Start Event.

BOOL LVDS_SetPageStart (DWORD dwEvent)

Parameters:

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_SetPageDelay

This function sets a Line Count.

BOOL LVDS_SetPageDelay (DWORD dwCount)

Parameters:

dwCount : How many clock need to be waited before one image acquisition as a page
Range 1 to 15.

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 (DWORD dwEvent, DWORD dwDelay, DWORD dwWidth, DWORD dwBlank)

Parameters:

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 (DWORD dwEvent, DWORD dwDelay, DWORD dwWidth, DWORD dwBlank)

Parameters:

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 :

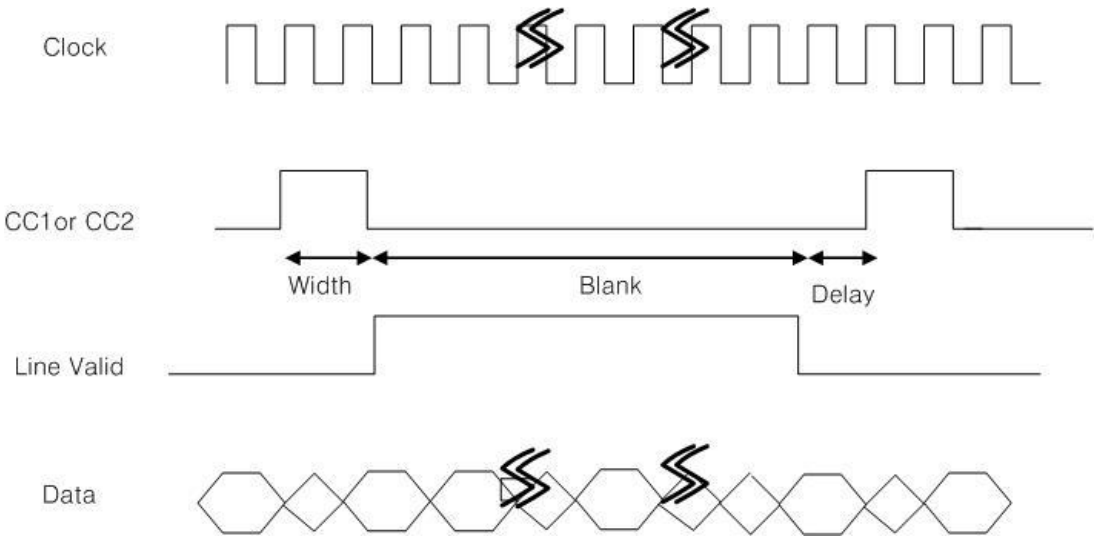
If the function call fails, it returns “FALSE”.

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

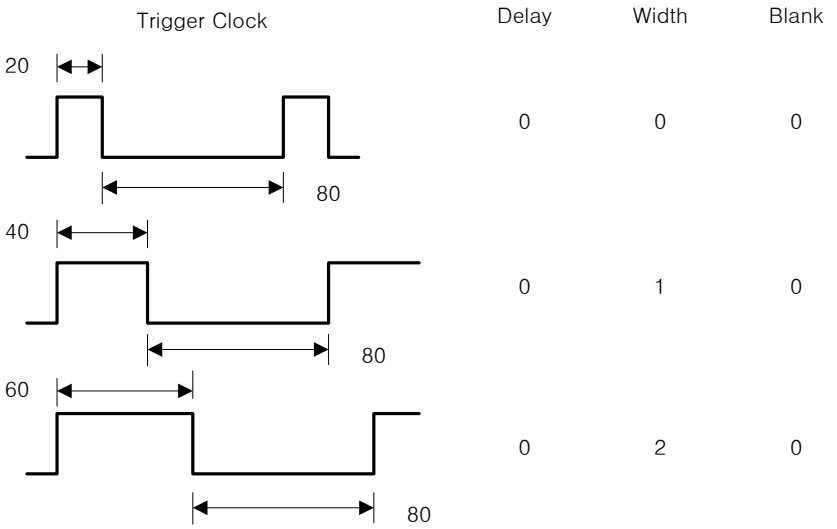
1. Note : Trigger # 1 (CC1), Trigger # 2 (CC2) of the following conditions, is set depending on the selected trigger Delay, Width, Blank.

Delay : 0 ~ 4095, Width & Blank : 0 ~ 65535

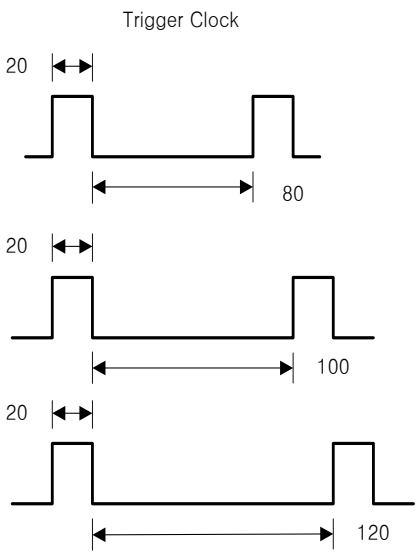
Minimum frequency can be set f=1/T , so 1 / ((4095 + 65535 + 65535) * 20ns) = 369.8Hz.



2. Note : If the default settings is Delay / Width / Blank of 0/0/0 100ns (20 +80), so the output is 10Mhz as shown in the figure below.



<When adjust the Width values>



Delay	Width	Blank				
0	0	0				
1	0	0	Or	0	0	1
2	0	0	Or	0	0	2
1	0	1				

<When adjust the Delay or Blank values>

LVDS_SetTrigger

This function starts a Trigger.

BOOL LVDS_SetTrigger (BOOL bUse)

Parameters:

bUse : “True” : Trigger State

“False” : Normal State

Return Value :

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

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

LVDS_SetRolling

This function updates the LVDS frame data without using the GetFrame function.

BOOL LVDS_SetRolling (BOOL bRolling)

Parameters:

bRolling : “True” : the frame data is updated regardless of the GetFrame function.

Return Value :

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

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

LVDS_SetPoCLDelay

When using a PoCL(Power over Camera Link), the user can select a delay time.
(DEFAULT is 0.5sec.)

BOOL LVDS_SetPoCLDelay (int nMode)

Parameters:

nMode : “0” : 0.5sec, “1” : 1sec, “2” : 1.5sec, “3” : 2sec
“4” : 2.5sec, “5” : 3sec, “6” : 3.5sec, “7” : 4sec

Return Value :

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

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

LVDS_ExtTrigEnable

This function selects an external trigger signal.

BOOL LVDS_ExtTrigEnable (BOOL bEn)

Parameters:

bEn : “True” : Use the External Trigger signal.
“False” : Use the Internal Trigger signal.

Return Value :

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

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

LVDS_ExtTrigInv

This function inverts the external trigger signal.

BOOL LVDS_ExtTrigInv (BOOL bInv)

Parameters:

bInv : "True" : It uses inversely converts the external trigger signal.
"False" : It uses inversely converts the internal trigger signal.

Return Value :

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

LVDS_ExtTrigConfigure

This function configures the external trigger signal.

BOOL LVDS_ExtTrigConfigure (DWORD dwDelay, DWORD dwWidth, DWORD dwBlank)

Parameters:

dwDelay : Number of reference clocks before external trigger signal goes high
dwWidth : Number of reference clocks for external trigger signal high
dwBlank : Number of reference clocks before external trigger signal returns from delay

Return Value :

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

LVDS_EncAInv

This function inverts the Encoder A signal.

BOOL LVDS_EncAInv (BOOL bInv)

Parameters:

bInv : "True" : Encoder A signal is inversely transformed and used.
"False" : Encoder A signal is used as it is.

Return Value :

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

LVDS_EncBInv

This function inverts the Encoder B signal.

BOOL LVDS_EncBInv (BOOL bInv)

Parameters:

bInv : “True” : Encoder B signal is inversely transformed and used.

“False” : Encoder B signal is used as it is.

Return Value :

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

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

LVDS_EncZInv

This function inverts the Encoder Z signal.

BOOL LVDS_EncZInv (BOOL bInv)

Parameters:

bInv : “True” : Encoder Z signal is inversely transformed and used.

“False” : Encoder Z signal is used as it is.

Return Value :

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

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

LVDS_LineTrigInv

This function inverts the Line Trigger signal.

BOOL LVDS_LineTrigInv (BOOL bInv)

Parameters:

bInv : “True” : Line Trigger signal is inversely transformed and used.

“False” : Line Trigger signal is used as it is.

Return Value :

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

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

LVDS_PhaseTrigInv

This function inverts the phase trigger signal.

BOOL LVDS_PhaseTrigInv (BOOL bInv)

Parameters:

bInv : “True” : Phase Trigger signal is inversely transformed and used.

“False” : Phase Trigger signal is used as it is.

Return Value :

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

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

LVDS_TrigOutInv

This function inverts the CC output waveform.

BOOL LVDS_TrigOutInv (int nInv)

Parameters:

nInv : “0” : CC1, CC2 Normal Signal

“1” : CC1 Inverse Signal

“2” : CC2 Inverse Signal

“Others” : CC1, CC2 Inverse Signal

Return Value :

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

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

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

This function initializes the resources that 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

This function receives the characters through the 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 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 (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 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 (void)

Parameters: None.

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 (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

This function clears an UART 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”.

CC(Camera Control) API Functions

Overview

DWORD DIO_Read (void)

BOOL DIO_Write (DWORD dwVal)

DIO_Read

This function reads the value of the input port.

DWORD 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

This function outputs the Camera Control values to port.

(Refer to each camera specification)

BOOL DIO_Write (DWORD dwVal)

Parameters:

dwVal : 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 LVDS(Camera Link) APIs

Overview

BOOL IsConnected_Mul (int nBoard)
BOOL LVDS_Init_Mul (int nBoard)
BOOL LVDS_Check_Mul (int nBoard)
BOOL LVDS_Start_Mul (int nBoard)
BOOL LVDS_GetFrame_Mul (int nBoard, DWORD* nCnt, unsigned char* buf)
BOOL LVDS_Close_Mul (int nBoard)
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_GetVersion_Mul (int nBoard, int *nVersion)
BOOL LVDS_GetError_Mul (int nBoard, DWORD *dwStatus)
BOOL LVDS_BufferFlush_Mul (int nBoard)
BOOL LVDS_SetDeUse_Mul (int nBoard, BOOL bUse)
BOOL LVDS_SetHsPol_Mul (int nBoard, BOOL bPol)
BOOL LVDS_CameraMode_Mul (int nBoard, int nMode)
BOOL LVDS_SetReferrenceClock_Mul (int nBoard, int nClock)
BOOL LVDS_ConfigureCc_Mul (int nBoard, DWORD dwCFG)
BOOL LVDS_CcOutput_Mul (int nBoard, DWORD dwCC)
BOOL LVDS_SetLineCount_Mul (int nBoard, DWORD dwCount)
BOOL LVDS_SetPageStart_Mul (int nBoard, DWORD dwEvent)
BOOL LVDS_SetPageDelay_Mul (int nBoard, DWORD dwCount)
BOOL LVDS_ConfigureTrig1_Mul (int nBoard, DWORD dwEvent,
DWORD dwDelay, DWORD dwWidth, DWORD dwBlank)
BOOL LVDS_ConfigureTrig2_Mul (int nBoard, DWORD dwEvent,
DWORD dwDelay, DWORD dwWidth, DWORD dwBlank)
BOOL LVDS_SetRolling_Mul (int nBoard, BOOL bRolling)
BOOL LVDS_SetPoCLDelay_Mul (int nBoard, int nMode)
BOOL LVDS_ExtTrigEnable_Mul (int nBoard, BOOL bEn)

BOOL LVDS_ExtTrigInv_Mul (int nBoard, BOOL bInv)

BOOL LVDS_ExtTriConfigure_Mul (int nBoard, DWORD dwDelay, DWORD dwWidth, DWORD dwBlank)

BOOL LVDS_EncAInv_Mul (int nBoard, BOOL bInv)

BOOL LVDS_EncBInv_Mul (int nBoard, BOOL bInv)

BOOL LVDS_EncZInv_Mul (int nBoard, BOOL bInv)

BOOL LVDS_LineTrigInv_Mul (int nBoard, BOOL bInv)

BOOL LVDS_PhaseTrigInv_Mul (int nBoard, BOOL bInv)

BOOL LVDS_TrigOutInv_Mul (int nBoard, int nInv)

IsConnected_Mul

This function confirms that it is connected to USB.

BOOL IsConnected _Mul (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_Init_Mul

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

BOOL LVDS_Init_Mul (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_Check_Mul

This function checks the completion of the received frame data.

BOOL LVDS_Check_Mul (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_Mul

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

BOOL LVDS_Start_Mul (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_GetFrame_Mul

This function acquires the image data from the frame buffer.

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

BOOL LVDS_GetFrame_Mul (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_Close_Mul

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_Mul (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_Mul

This function gets a currently configured camera’s frame resolution

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

Parameters:

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

*xRes : Width of image in pixels

*yRes : Height of Image in pixels

Return Value:

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

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

LVDS_Stop_Mul

This function stops a frame data capture.

BOOL LVDS_Stop_Mul (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_SetDataMode_Mul

This function sets an image pixel data mode.

BOOL LVDS_SetDataMode_Mul (int nBoard, int nMode)

Parameters:

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

nMode : “0” : 8bit Mode, “1” : 16bit Mode

“2” : 32bit Mode, “3” : 64bit Mode

Return Value:

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

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

LVDS_GetVersion_Mul

This function gets a current FPGA version.

BOOL LVDS_GetVersion_Mul (int nBoard, int *nVersion)

Parameters:

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

*nVersion : FPGA version.

Return Value:

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

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

LVDS_GetError_Mul

This function gets a Frame (Image) Error.

DWORD LVDS_GetError_Mul (int nBoard, DWORD *dwStatus)

Parameters:

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

*dwStatus : Error state.

“1” : Overflow error, “2” : Read error

“4” : Size error

Return Value :

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

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

LVDS_BufferFlush_Mul

This function initializes the buffer.

BOOL LVDS_BufferFlush_Mul (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_SetDeUse_Mul

This function sets a Data Valid or Horizontal Synchronization signal.

BOOL LVDS_SetDUse_Mul (int nBoard, BOOL bUse)

Parameters:

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

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

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

Return Value :

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

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

LVDS_SetHsPol_Mul

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

BOOL LVDS_SetHsPol_Mul (int nBoard, BOOL bPol)

Parameters:

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

bPol : If the value is “True”, Normal HSYNC use,

If the value is “False”, Inverse HSYNC use.

Return Value :

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

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

LVDS_CameraMode_Mul

This function sets a Camera Operation Mode.

BOOL LVDS_CameraMode_Mul (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 Scan Camera operation(Default).

If the value is “Others”, it is a Line Scan Camera operation.

Return Value :

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

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

LVDS_SetReferenceClock_Mul

This function sets an Internal or External clock.

BOOL LVDS_SetReferenceClock_Mul (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”, Internal Clock is used(Default).

If the value is “Others”, External Clock is used.

Return Value :

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

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

LVDS_ConfigureCc_Mul

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

BOOL LVDS_ConfigureCc_Mul (int nBoard, DWORD dwCFG)

Parameters:

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

dwCFG : 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_Mul

This function outputs a CC1 ~ CC4 value.

BOOL LVDS_CcOutput_Mul (int nBoard, DWORD dwCC)

Parameters:

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

dwCc : bit0(CC1 out) ➔ “0” : output “0” / “1”: output “1”

bit1(CC2 out) ➔ “0” : output “0” / “1”: output “1”

bit2(CC3 out) ➔ “0” : output “0” / “1”: output “1”

bit3(CC4 out) ➔ “0” : output “0” / “1”: output “1”

others : Reserved

Return Value :

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

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

LVDS_SetLineCount_Mul

This function sets a Line Count.

BOOL LVDS_SetLineCount_Mul (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_SetPageStart_Mul

This function sets a Page Start Event.

BOOL LVDS_SetPageStart_Mul (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_SetPageDelay_Mul

This function sets a Line Count.

BOOL LVDS_SetPageDelay_Mul (int nBoard, DWORD dwCount)

Parameters:

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

dwCount : How many clock need to be waited before one image acquisition as a page

Range 1 to 15.

Return Value :

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

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

LVDS_ConfigureTrig1_Mul

This function configures a Trigger 1.

BOOL LVDS_ConfigureTrig1_Mul (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 needed 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_Mul

This function configures a Trigger 2.

BOOL LVDS_ConfigureTrig2_Mul (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 needed 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_SetRolling_Mul

This function updates the LVDS frame data without using the GetFrame function.

BOOL LVDS_SetRolling_Mul (int nBoard, BOOL bRolling)

Parameters:

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

bRolling : "True" : the frame data is updated regardless of the GetFrame function.

Return Value :

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

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

LVDS_SetPoCLDelay_Mul

When using a PoCL(Power over Camera Link), the user can select a delay time.
(DEFAULT is 0.5sec.)

BOOL LVDS_SetPoCLDelay_Mul (int nBoard, int nMode)

Parameters:

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

nMode : “0” : 0.5sec, “1” : 1sec, “2” : 1.5sec, “3” : 2sec
“4” : 2.5sec, “5” : 3sec, “6” : 3.5sec, “7” : 4sec

Return Value :

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

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

LVDS_ExtTrigEnable_Mul

This function selects an external trigger signal.

BOOL LVDS_ExtTrigEnable_Mul (int nBoard, BOOL bEn)

Parameters:

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

bEn : “True” : Use the External Trigger signal.
“False” : Use the Internal Trigger signal.

Return Value :

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

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

LVDS_ExtTrigInv_Mul

This function inverts the external trigger signal.

BOOL LVDS_ExtTrigInv_Mul (int nBoard, BOOL bInv)

Parameters:

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

bInv : "True" : It uses inversely converts the external trigger signal.

"False" : It uses inversely converts the internal trigger signal.

Return Value :

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

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

LVDS_ExtTrigConfigure_Mul

This function configures the external trigger signal.

BOOL LVDS_ExtTrigConfigure_Mul (int nBoard, DWORD dwDelay,
DWORD dwWidth, DWORD dwBlank)

Parameters:

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

dwDelay : Number of reference clocks before external trigger signal goes high

dwWidth : Number of reference clocks for external trigger signal high

dwBlank : Number of reference clocks before external trigger signal returns from delay

Return Value :

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

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

LVDS_EncAInv_Mul

This function inverts the Encoder A signal.

BOOL LVDS_EncAInv_Mul (int nBoard, BOOL bInv)

Parameters:

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

bInv : "True" : Encoder A signal is inversely transformed and used.

"False" : Encoder A signal is used as it is.

Return Value :

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

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

LVDS_EncBInv_Mul

This function inverts the Encoder B signal.

BOOL LVDS_EncBInv_Mul (int nBoard, BOOL bInv)

Parameters:

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

bInv : "True" : Encoder B signal is inversely transformed and used.

"False" : Encoder B signal is used as it is.

Return Value :

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

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

LVDS_EncZInv_Mul

This function inverts the Encoder Z signal.

BOOL LVDS_EncZInv_Mul (int nBoard, BOOL bInv)

Parameters:

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

bInv : "True" : Encoder Z signal is inversely transformed and used.

"False" : Encoder Z signal is used as it is.

Return Value :

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

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

LVDS_LineTrigInv_Mul

This function inverts the Line Trigger signal.

BOOL LVDS_LineTrigInv_Mul (int nBoard, BOOL bInv)

Parameters:

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

bInv : "True" : Line Trigger signal is inversely transformed and used.

"False" : Line Trigger signal is used as it is.

Return Value :

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

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

LVDS_PhaseTrigInv_Mul

This function inverts the phase trigger signal.

BOOL LVDS_PhaseTrigInv_Mul (int nBoard, BOOL bInv)

Parameters:

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

bInv : "True" : Phase Trigger signal is inversely transformed and used.

"False" : Phase Trigger signal is used as it is.

Return Value :

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

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

LVDS_TrigOutInv_Mul

This function inverts the CC output waveform.

BOOL LVDS_TrigOutInv_Mul (int nBoard, int nInv)

Parameters:

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

nInv : "0" : CC1, CC2 Normal Signal, "1" : CC1 Inverse Signal,

"2" : CC2 Inverse Signal, "Others" : CC1, CC2 Inverse Signal

Return Value :

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

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

Multi-board 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

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

BOOL **UART_Init_Mul (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_Mul

This function receives the characters through the UART.

BOOL **UART_GetData_Mul (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_Mul

This function sends the characters through the UART.

BOOL UART_SendData_Mul (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_Mul

This function releases all resource that used for UART function.

BOOL UART_Close_Mul (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_Mul

This function sets an UART Baud.

BOOL UART_SetBaud_Mul (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_Mul

This function clears an UART Rx buffer.

BOOL UART_BufferFlush_Mul (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”.

CC(Camera Control) API Functions

Overview

BOOL DIO_Read_Mul (int nBoard, DWORD val)

BOOL DIO_Write_Mul (int nBoard, DWORD val)

DIO_Read

This function reads the value of the input port.

DWORD DIO_Read (void)

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”.

DIO_Write_Mul

This function outputs the Camera Control values to port.

(Refer to each camera specification)

BOOL DIO_Write_Mul (int nBoard, DWORD val)

Parameters:

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