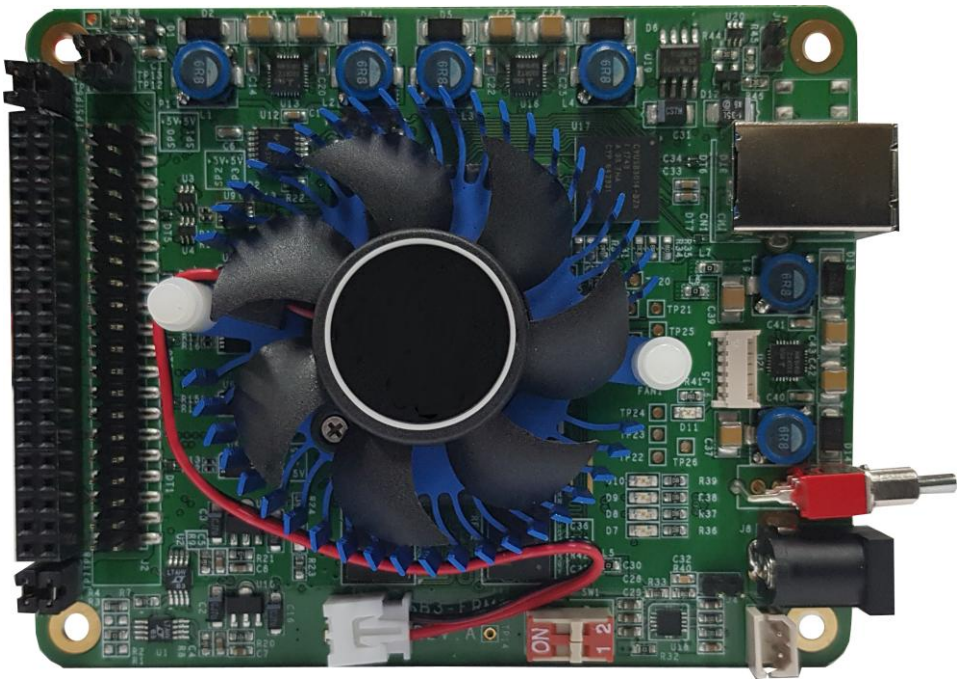
USB3-FRM02 API Programming (Rev 1.0)



Board Level APIs

Overview

Int	**OpenDAQDevice (void)**
BOOL	**ResetBoard (int nBoard)**
BOOL	**CloseDAQDevice (void)**
Int	**GetBoardNum (void)**
char*	**GetDllVersion(void)**
BOOL	**SetFrameSize (DWORD dwSize)**
BOOL	**IsConnected(int nBoard)**

OpenDAQDevice

디바이스를 Open한다.

프로그램에서 초기에 반드시 한번 함수를 호출하여 디바이스를 Open 하여야 한다.

Int **OpenDAQDevice (void)**

Parameters: 없음.

Return Value:

함수 호출에 성공한 경우, 설치된 보드의 번호를 리턴한다.

함수 호출에 실패한 경우, "-1"을 리턴한다. 이것의 의미는 시스템에 장치가 없다는 의미이다.

ResetBoard

현재 시스템(PC)에 장착된 디바이스를 초기화 한다.

BOOL **ResetBoard (int nBoard)**

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.

보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

Return Value:

함수 호출에 성공일 경우 "TRUE"을 리턴함.

Open된 보드가 없을 경우 "FALSE" 를 리턴한다.

CloseDAQDevice

Open된 디바이스를 Close한다. 장치의 사용이 끝나게 되면, 반드시 장치를 Close 하여 다른 프로그램에서 사용할 수 있도록 한다.

BOOL **CloseDAQDevice (void)**

Parameters: 없음.

Return Value:

디바이스 Open에 성공할 경우 TRUE를 실패할 경우 FALSE를 리턴한다.

GetBoardNum

시스템에서 설치된 보드 개수를 알려준다.

int **GetBoardNum (void)**

Parameters: 없음.

Return Value:

설치된 보드의 개수를 리턴한다.

보드 넘버는 장착되어있는 DIP 스위치(SW1)의 설정 값이 넘어온다.

GetDllVersion

DLL 버젼을 알려준다.

Char* **GetDllVersion (void)**

Parameters: 없음.

Return Value:

설치되어 있는 DLL의 날짜가 넘어 온다.

SetFrameSize

Frame Buffer 크기를 설정한다.

BOOL **SetFrameSize (DWORD dwSize)**

Parameters:

dwSize : Frame Buffer Size (1M ~ 128M), default 48M

Return Value:

함수 호출에 실패할 경우 “FALSE” 성공일 경우 “TRUE”을 리턴함.

IsConnected

USB에 연결이 되어 있는지를 알려준다.

BOOL IsConnected (int nBoard)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

Return Value:

함수 호출에 실패할 경우 “FALSE” 성공일 경우 “TRUE”을 리턴함.

LVDS API Functions

Overview

BOOL LVDS_GetVersion (int nBoard, DWORD *dwFpga, DWORD *dwFirm)
BOOL LVDS_Init (int nBoard)
BOOL LVDS_Check(int nBoard)
BOOL LVDS_SetDataMode (int nBoard, int nMode)
BOOL LVDS_GetResolutuion (int nBoard, DWORD *xRes, DWORD *yRes)
BOOL LVDS_Start (int nBoard)
BOOL LVDS_Stop (int nBoard)
BOOL LVDS_GetFrame (int nBoard, DWORD* nCnt, unsigned char* buf)
BOOL LVDS_Flush (int nBoard)
BOOL LVDS_SelectInput (int nBoard, int nInput)
BOOL LVDS_SetSyncFlag (int nBoard, BOOL bFlag)
BOOL LVDS_GetSyncCount (int nBoard, DWORD *dwVsync, DWORD *dwHsync)

LVDS_GetVersion

FPGA 와 Firmware version을 가져온다.

BOOL LVDS_GetVersion (int nBoard, DWORD *dwFpga, DWORD *dwFirm)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP스위치를 이용하여 설정한다.

dwFpga : FPGA 버전.

dwFirm : Firmware 버전

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

LVDS_Init

LVDS sub-system의 자원, 예를 들어 interrupt와 LVDS control register을 초기화한다.

BOOL LVDS_Init (int nBoard)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP스위치를 이용하여 설정한다.

Return Value:

함수 호출에 실패할 경우 "FALSE" 성공일 경우 "TRUE"을 리턴함.

LVDS_Check

Device가 정상적으로 연결 되어있는지 Check하는 함수이다.

BOOL LVDS_Check (int nBoard)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP스위치를 이용하여 설정한다.

Return Value:

프레임 데이터를 완벽하게 받았다면 "TRUE"을 리턴함.

LVDS_SetDataMode

입력되는 프레임(이미지) 데이터 모드를 설정한다.

BOOL LVDS_SetDataMode (int nBoard, int nMode)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP스위치를 이용하여 설정한다.
nMode : "0" 이면 8bit Mode이고, "1" 이면 16bit Mode
"2" 이면 32bit Mode 이다.

Return Value :

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

LVDS_GetResolution

Video 입력의 해상도를 가져온다.

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

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP스위치를 이용하여 설정한다.

*nXres : 수평해상도 값을 얻어올 번지.

*nYres : 수직해상도 값을 얻어올 번지.

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

LVDS_Start

LVDS_Start 함수 호출이 되면 Board에서 DLL로 Data를 전송 시작한다.
(USB Board의 경우 Thread를 통하여 DLL의 Buffer에 Data를 쌓고, LVDS_GetFrame 함수로 DLL -> APP 로 copy 한다.)

BOOL LVDS_Start (int nBoard)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP스위치를 이용하여 설정한다.

Return Value:

함수 호출에 실패할 경우 "FALSE" 성공일 경우 "TRUE"을 리턴함.

LVDS_Stop

프레임 데이터 Capture를 중지한다.

BOOL LVDS_Stop (int nBoard)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP스위치를 이용하여 설정한다.

Return Value :

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

LVDS_GetFrame

프레임 데이터가 완성이 되었는가를 검사하고 완성이 되었으면 프레임 데이터를 가져온다. 이때 데이터를 받아올 버퍼 크기를 반드시 알려주어야 한다.

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

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP스위치를 이용하여 설정한다.

nCnt : 프레임 데이터를 받아 올 버퍼의 크기가 저장 되어있는 변수의 번지이다. 함수를 호출할때 버퍼크기를 지정하고 호출한 후에는 변수 값을 읽어서 실제로 읽어 온 개수를 확인한다. 데이터 크기는 바이트 단위이다.

buf : 프레임 버퍼 포인터.

Return Value:

함수 호출에 실패할 경우 “FALSE” 성공일 경우 “TRUE”을 리턴함.
호출이 성공하면 nCnt 값을 확인하여 원하는 크기 만큼 읽어 졌는가를 확인한다.

㈜ Open이 되지 않았거나, DLL Thread가 돌지 않는 경우, USER가 입력한 nCnt값이 Buffer Size보다 큰 경우 nCnt를 “0”으로 Return 하고,
아직 프레임 데이터가 덜 쌓인 경우에도 return “FALSE”만 합니다.

LVDS_Flush

버퍼를 초기화 한다.

BOOL LVDS_Flush (int nBoard)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP스위치를 이용하여 설정한다.

Return Value :

함수 호출에 실패한 경우 ‘FALSE’, 성공일 경우 ‘TRUE’를 return한다.

LVDS_SelectInput

LVDS 입력 모드를 설정한다.

BOOL LVDS_SelectInput (int nBoard, int nInput)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP스위치를 이용하여 설정한다.

nMode : "0" 이면 MIPI 입력이고 "1" 이면 Parallel 입력이다.

Return Value :

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

LVDS_SetSyncFlag

하드웨어에서 1초당 Vsync/Hsync 카운팅을 시작한다.

BOOL LVDS_SetSyncFlag (int nBoard, BOOL bFlag)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP스위치를 이용하여 설정한다.

bFlag : : "0" 이면 False 이고 "1" 이면 Vsync/Hsync 카운팅을 한다.

Return Value:

함수 호출에 실패할 경우 "FALSE" 성공일 경우 "TRUE"을 리턴함.

LVDS_GetSyncCount

Vsync, Hsync를 카운트 한다.

BOOL LVDS_GetSyncCount (int nBoard, DWORD *dwVsync, DWORD *dwHsync)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP스위치를 이용하여 설정한다.

dwVsync : Vsync 수.

dwHsync : Hsync 수

Return Value:

함수 호출에 실패할 경우 "FALSE" 성공일 경우 Vsync, Hsync의 개수를 리턴함.

Clock API Functions

Overview

BOOL **CLK_Select (int nBoard, int nSelect)**

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

CLK_Select

센서에 제공하는 Clock On/Off 를 설정한다.

BOOL **CLK_Select (int nBoard, int nSelect)**

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

nSelect : "0" 이면 Off, "Others" 이면 On

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

CLK_Set

Clock의 주파수를 설정한다. (1039Hz ~ 68Mhz까지 세팅할 수 있다.)

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

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

val : 1039Hz ~ 68Mhz.

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

MISC API Functions

Overview

BOOL **I2C_SEN_Off (int nBoard, BOOL bOff)**
BOOL **SEN_Reset (int nBoard, BOOL bReset)**
BOOL **SEN_Enable (int nBoard, BOOL bEnable)**
BOOL **PWR_IO_On (int nBoard, BOOL bOn)**
BOOL **PWR_IO_Voltage (int nBoard, float fVoltage)**
BOOL **PWR_IO2_On (int nBoard, BOOL bOn)**
BOOL **PWR_IO2_Voltage (int nBoard, float fVoltage)**

I2C_SEN_Off

Sensor의 I2C 전원을 disable 시킨다.

BOOL **I2C_SEN_Off (int nBoard, BOOL bOff)**

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

bOff : 참이면 Off, 거짓이면 On

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

SEN_Reset

sensor를 초기화시킨다.

BOOL **SEN_Reset (int nBoard, BOOL bReset)**

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

bReset: 참이면 Reset, 거짓이면 기존 상태 유지

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

SEN_Enable

Sensor를 Enable 시킨다.

BOOL SEN_Enable (int nBoard, BOOL bEnable)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

bEnable: 참이면 Enable, 거짓이면 기존 상태 유지

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

PWR_IO_On

MIPI I/O Power을 enable 시킨다.

BOOL PWR_IO_On (int nBoard, BOOL bOn)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

bOn: 참이면 On, 거짓이면 Off

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

PWR_IO_Voltage

MIPI I/O level을 설정한다. (I2C/Sensor Enable 등 MIPI 신호 Level 1.5 ~ 3.3V 설정).

BOOL PWR_IO_Voltage (int nBoard, float fVoltage)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

fVoltage : 1.5 ~ 3.3V 전압 값.

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

PWR_IO2_On

Parallel GPIO Power를 enable 시킨다.

BOOL PWR_IO2_On (int nBoard, BOOL bOn)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

bOn: 참이면 On, 거짓이면 Off

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

PWR_IO2_Voltage

Parallel GPIO 전압 Level을 설정한다. (1.5 ~ 3.3V 설정).

BOOL PWR_IO2_Voltage (int nBoard, float fVoltage)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

fVoltage : 1.5 ~ 3.3V 전압 값.

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

I2C API Functions

Overview

BOOL I2C_SYS_Reset (int nBoard)

BOOL I2C_SYS_Set_Clock (int nBoard, int nClock)

BOOL I2C_SYS_Read (int nBoard, Byte slAddr, DWORD nAddrLen, DWORD nAddr, DWORD nCnt, unsigned char* buf)

BOOL I2C_SYS_Write (int nBoard, Byte slAddr, DWORD nAddrLen, DWORD nAddr, DWORD nCnt, unsigned char* buf)

I2C_SYS_Reset

시스템 모듈의 I2C system의 자원을 초기화 한다.

BOOL I2C_SYS_Reset (int nBoard)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

I2C_SYS_Set_Clock

시스템 모듈의 I2C system의 clock을 설정한다.

BOOL I2C_SYS_Set_Clcok (int nBoard, int nClock)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

nClock : I2C clock의 개수 (default :100000)

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

I2C_SYS_Read

시스템 모듈의 I2C를 통하여 특정 어드레스 번지 값을 정해진 개수만큼 읽어 온다.

BOOL I2C_SYS_Read (int nBoard, Byte slAddr, DWORD nAddrLen, DWORD nAddr, DWORD nCnt, unsigned char* buf)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

slAddr : I2C address

nAddrLen : Address Lengh

nAddr : Register Address

nCnt : 읽어 올 데이터의 바이트 개수.

buf : 읽어 올 데이터의 버퍼 포인터.

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

I2C_SYS_Write

시스템 모듈의 I2C를 통하여 특정 어드레스 번지 값을 정해진 개수만큼 쓴다.

BOOL I2C_SYS_Write (int nBoard, Byte slAddr, DWORD nAddrLen, DWORD nAddr, DWORD nCnt, unsigned char* buf)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

slAddr : I2C address

nAddrLen : Address Lengh

nAddr : Register Address

nCnt : 기록할 데이터의 바이트 개수.

buf : 기록할 데이터의 버퍼 포인터.

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

Parallel / Virtual Mode API Functions

Overview

BOOL MIPI_VirtualMode (int nBoard, BOOL bEn)
BOOL Virtual_VsyncDelay (int nBoard, DWORD dwDelay)
BOOL Parallel_BT656Mode (int nBoard, BOOL bEn)
BOOL Parallel_VsyncPol (int nBoard, BOOL bEn)
BOOL Parallel_PclkPol (int nBoard, BOOL bEn)
BOOL FAN_Control (int nBoard, BOOL bOn, BYTE bySpeed)

MIPI_VirtualMode

Virtual Mode를 On/Off 시킨다.

BOOL MIPI_VirtualMode (int nBoard, BOOL bOff)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

bEn : 참이면 Virtual Mode, 거짓이면 MIPI mode

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

Virtual_VsyncDelay

Virtual Mode 일 경우 Vsync 의 Delay(지연) 값을 줄 수 있다.

BOOL Virtual_VsyncDelay (int nBoard, DWORD dwDelay)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

dwDelay : 0 to 0x00FFFFFF (268435455)

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

Parallel_BT656Mode

Parallel / BT656 Mode를 선택한다.

BOOL Parallel_BT656Mode (int nBoard, BOOL bEn)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

bEn : 참이면 BT656 Mode, 거짓이면 Parallel Mode

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

Parallel_VsyncPol

Parallel 및 BT656 모드 Vsync의 극성을 선택한다.

BOOL Parallel_VsyncPol (int nBoard, BOOL bEn)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

bEn : 참이면 Inverse , 거짓이면 Original

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

Parallel_PclkPol

Parallel 및 BT656 모드 Pixel Clock의 극성을 선택한다.

BOOL Parallel_PclkPol (int nBoard, BOOL bEn)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

bEn : 참이면 Inverse , 거짓이면 Original

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

FAN_Control

FAN 을 On/OFF하고 속도를 제어한다.

BOOL FAN_Control (int nBoard, BOOL bOn, BYTE bySpeed)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

bEn : 참이면 Fan On , 거짓이면 Fan Off

bySpeed: FAN Speed

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

Parallel GPIO Input/Output API Functions

Overview

BOOL DIO_SetDirection (int nBoard, DWORD dwVal)
BOOL DIO_GetDirection (int nBoard, DWORD *dwVal)
DWORD DIO_Read (int nBoard)
BOOL DIO_Write (int nBoard, DWORD dwVal)
BOOL DIO_GetWrite (int nBoard, DWORD *dwVal)

DIO_SetDirection

각각의 포트를 입력으로 사용할지 출력으로 사용할지 설정한다.
(Parallel GPIO setting 하위 8비트)

BOOL DIO_SetDirection (int nBoard, DWORD dwVal)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

dwVal : 입/출력 direction 설정 값.
각 포트에 '1'이면 출력 / '0'이면 입력

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

DIO_GetDirection

현재 설정된 direction 값을 읽어 온다.

BOOL DIO_GetDirection (int nBoard, DWORD *dwVal)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

*dwVal : 입/출력 direction을 읽어올 변수.
각 포트에 '1'이면 출력 / '0'이면 입력

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

DIO_Read

입력 값을 읽어 온다.

DWORD DIO_Read (int nBoard)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

DIO_Write

원하는 값을 출력포트에 출력한다.

BOOL DIO_Write (int nBoard, DWORD dwVal)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

dwVal : 출력 포트에 기록할 값.

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

DIO_GetWrite

출력된 현재 값을 적는다.

BOOL DIO_GetWrite (int nBoard, DWORD *dwVal)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

*dwVal : 출력 포트의 현재 값을 쓸 변수이다.

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

MIPI Sensor GPIO Input/Output API Functions

Overview

BOOL **Sensor_GPIO_SetDirection (int nBoard, DWORD dwVal)**
BOOL **Sensor_GPIO _GetDirection (int nBoard, DWORD *dwVal)**
DWORD **Sensor_GPIO _Read (int nBoard)**
BOOL **Sensor_GPIO _Write (int nBoard, DWORD dwVal)**
BOOL **Sensor_GPIO _GetWrite (int nBoard, DWORD *dwVal)**

Sensor_GPIO_SetDirection

각각의 MIPI GPIO 포트를 입력으로 사용할지 출력으로 사용할지 설정한다.

BOOL **Sensor_GPIO_SetDirection (int nBoard, DWORD dwVal)**

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

dwVal : 입/출력 direction 설정 값.
각 포트에 '1'이면 출력 / '0'이면 입력

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

Sensor_GPIO_GetDirection

현재 설정된 MIPI GPIO direction 값을 읽어 온다.

BOOL **Sensor_GPIO_GetDirection (int nBoard, DWORD *dwVal)**

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

*dwVal : 입/출력 direction을 읽어올 변수.
각 포트에 '1'이면 출력 / '0'이면 입력

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

Sensor_GPIO_Read

MIPI GPIO 입력 값을 읽어 온다.

DWORD Sensor_GPIO_Read (int nBoard)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

Sensor_GPIO_Write

MIPI GPIO 원하는 값을 출력포트에 출력한다.

BOOL Sensor_GPIO_Write (int nBoard, DWORD dwVal)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

dwVal : 출력 포트에 기록할 값.

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

Sensor_GPIO_GetWrite

MIPI GPIO 현재 값을 출력한다.

BOOL Sensor_GPIO_GetWrite (int nBoard, DWORD *dwVal)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

*dwVal : 출력 포트의 현재 값을 쓸 변수이다.

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

SPI(Serial Peripheral Interface) API Functions

Overview

BOOL SPI_Write (int nBoard, BYTE byCmd, int AddrLen,
BYTE *pAddr, int nDataLen, BYTE *pData)

BOOL SPI_Read (int nBoard, BYTE byCmd, int AddrLen,
BYTE *pAddr, int nDataLen, BYTE *pData)

SPI_Write

Serial Peripheral Interface 버스에 데이터를 쓴다.
(AddrLen + nDataLen 은 7바이트를 넘지 않는다)

BOOL SPI_Write (int nBoard, BYTE byCmd, int AddrLen,
BYTE *pAddr, int nDataLen, BYTE *pData)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

byCmd: 명령어

AddrLen: 어드레스 길이

pAddr : 데이터의 어드레스 주소

nDataLen : 전송할 데이터 길이

pData : 전송할 데이터

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.

SPI_Read

Serial Peripheral Interface 버스의 데이터를 읽어 온다.
(AddrLen + nDataLen 은 7바이트를 넘지 않는다)

BOOL SPI_Read (int nBoard, BYTE byCmd, int AddrLen,
BYTE *pAddr, int nDataLen, BYTE *pData)

Parameters:

nBoard : 현재 시스템에 장착되어 있는 보드 번호를 알려준다.
보드 번호는 보드의 DIP 스위치를 이용하여 설정한다.

byCmd: 명령어

AddrLen: 어드레스 길이

pAddr : 데이터의 어드레스 주소

nDataLen : 전송할 데이터 길이

pData : 전송할 데이터

Return Value:

함수 호출에 실패한 경우 'FALSE', 성공일 경우 'TRUE'를 return한다.