```
/*****************************************************************************
*
*  HYTEK Automation
*  iUSBDAQ driver API for Linux
*****************************************************************************
* FileName:        iUSBDAQ.h
* Dependencies:    None
* Compiler:        C++
* Company:         Copyright (C) 2007 by HYTEK Automation, Inc.
*
* Software License Agreement
*
* The software supplied herewith by HYTEK Automation Incorporated
* (the Company) for its iUSBDAQ. The software is owned by the
* Company, and is protected under applicable copyright laws.
* All rights are reserved.
*
* THIS SOFTWARE IS PROVIDED IN AN AS IS CONDITION. NO WARRANTIES,
* WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED
* TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
* PARTICULAR PURPOSE APPLY TO THIS SOFTWARE. THE COMPANY SHALL NOT,
* IN ANY CIRCUMSTANCES, BE LIABLE FOR SPECIAL, INCIDENTAL OR
* CONSEQUENTIAL DAMAGES, FOR ANY REASON WHATSOEVER.
*
* Date:       11/18/2006    Original version 1.10 completed
*****************************************************************************/

/*****************************************************************************
* Please see the iUSBDAQ User's Guide for the documentation of the
* interface functions defined in this file.
*
*****************************************************************************/
#ifndef HYTEK_IUSBDAQ_API_HEADER
#define HYTEK_IUSBDAQ_API_HEADER

#include <usb.h>
#define BYTE    char
#define DevSession usb_dev_handle*
#define DWORD   int

#define IUSBDAQ_API
#define IUSBDAQ_STD_API

typedef enum
{
        iUSBDAQ_NOERROR=0,
```

```
	iUSBDAQ_Unknown=1,
	iUSBDAQ_InvalidDevIndex,    //device index exceed the max device number of that type
	iUSBDAQ_WrongType,  //No such iUSBDAQ module type or wrong device type
	iUSBDAQ_HandleError,    //Open device session error
	iUSBDAQ_DeviceNotAvailable,//device maybe used by other application
                                //or session already been opened
	iUSBDAQ_WriteError, //Write error, it could be the device disconnected
	iUSBDAQ_ReadError,  //Read error, it could be the device disconnected
	iUSBDAQ_NULLSession,    //NULL session
	iUSBDAQ_RcvLength,   // Partially failed, incorrect receive length
	iUSBDAQ_InvalidByteCount,//the byte count number is invalid
	iUSBDAQ_InvalidChannel,//channel number is incorrect
	iUSBDAQ_InvalidNofChs,//number of channels invalid
	iUSBDAQ_InputRange, //Wrong input voltage range
	iUSBDAQ_InvalidDutyCycle,//Duty cycle value should be between 0- 1023
	iUSBDAQ_InvalidDIOWrite,//The IO channel is not an output channel
	iUSBDAQ_InvalidPeriod,//wrong PWM period

	iUSBDAQ_TooManyScans, // iUSBDAQ_AIGetScans function asks too many scans to get, max
6400/NrOfChannels
	iUSBDAQ_ReadSamplesTimeout,
	iUSBDAQ_LowNofSamples,
	iUSBDAQ_HighNofSamples,
	iUSBDAQ_ReadWritePassAllowedAddr,
	iUSBDAQ_PMW_Period_too_big,
	iUSBDAQ_PMW_Period_too_small,
   /* first used in linux driver */
   iUSBDAQ_UsbConfig_Err,
	iUSBDAQ_MAX_ERR
} iUSBDAQ_RETURN_CODE;

iUSBDAQ_RETURN_CODE iUSBDAQ_EnumerateDev (unsigned long DevType, unsigned long*
Count);

iUSBDAQ_RETURN_CODE iUSBDAQ_OpenDevice (unsigned long DevType, unsigned long DevIndex,
DevSession* Session);

iUSBDAQ_RETURN_CODE iUSBDAQ_ReleaseDevice (DevSession Session);

iUSBDAQ_RETURN_CODE iUSBDAQ_GetDeviceSerialNo (DevSession Session, unsigned long*
SerialNumber);


iUSBDAQ_RETURN_CODE iUSBDAQ_ReadSingleAnalogIn ( DevSession Session, int Channel,
                        int InputRange, float * Voltage,
                        int * reserved);
```

```
iUSBDAQ_RETURN_CODE iUSBDAQ_ReadCounter (DevSession Session, int Reset,
                    unsigned int *Count);

iUSBDAQ_RETURN_CODE iUSBDAQ_DIO (DevSession Session,
            BYTE Port1to8dir,
            BYTE Port9to16dir,
            BYTE *Port1to8state,
            BYTE *Port9to16state,
            int updateDio);

iUSBDAQ_RETURN_CODE iUSBDAQ_ReadMultiAnalogIn(DevSession Session,
                    int StartCh, int NofChs, int InputRange,
                    float *Voltages, int * reserved);

float iUSBDAQ_BitsToVolt (int Bits);

int iUSBDAQ_VoltToBits ( float Voltage);

iUSBDAQ_RETURN_CODE iUSBDAQ_GetCredits (DevSession Session, BYTE* data);

iUSBDAQ_RETURN_CODE iUSBDAQ_GetFirmwareVersion (DevSession Session, unsigned long *
Version);

void iUSBDAQ_GetErrorDes (int ErrorCode, char* ErrorString);//max 256 characters

iUSBDAQ_RETURN_CODE iUSBDAQ_Reset (DevSession Session);

iUSBDAQ_RETURN_CODE iUSBDAQ_PWMOut (DevSession Session, BYTE Channel, int DutyCycle, int
Period);

iUSBDAQ_RETURN_CODE iUSBDAQ_STOPPWM (DevSession Session, int Channel);

iUSBDAQ_RETURN_CODE iUSBDAQ_ReadMemory (DevSession Session, BYTE ByteCount, BYTE
Address, BYTE* Data);

iUSBDAQ_RETURN_CODE iUSBDAQ_WriteMemory (DevSession Session, BYTE ByteCount, BYTE
Address, BYTE* Data);

iUSBDAQ_RETURN_CODE iUSBDAQ_ReadIUSB_DEVID (DevSession Session, BYTE *IUSB_DevID);

iUSBDAQ_RETURN_CODE iUSBDAQ_10bitPWMOut (DevSession Session, BYTE Channel, int
DutyCycle, int period);

void iUSBDRV_LoadDLL(void);
```

#endif