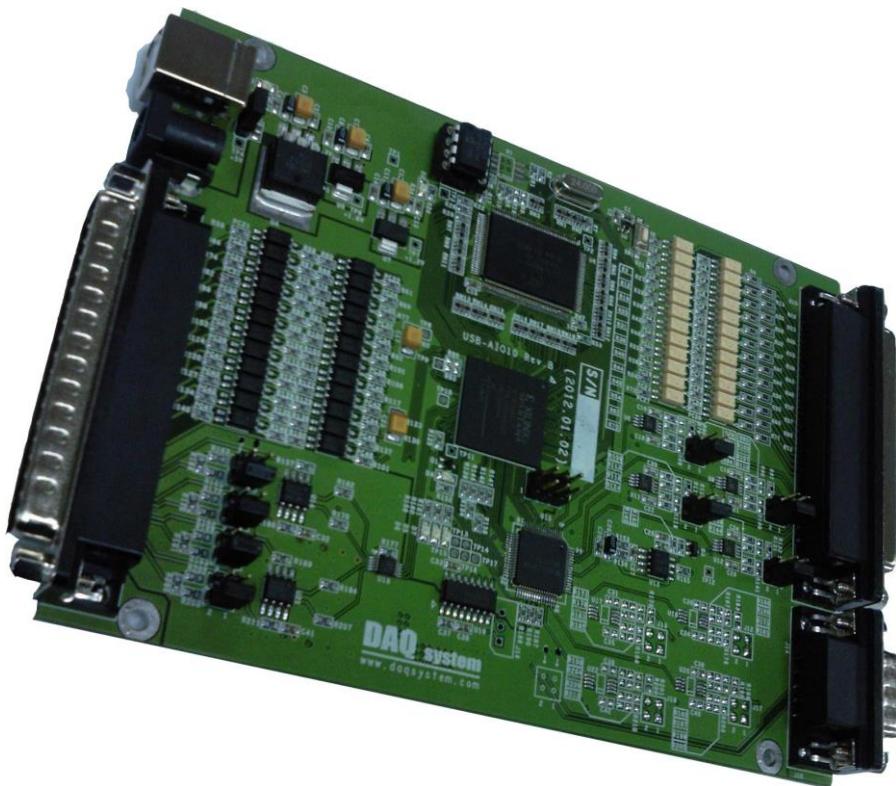


USB-AIO10 API Programming (Rev 1.0d)



Windows, Windows2000, Windows NT, Windows XP and Windows 7 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 © 2012 DAQ system Co., LTD. All rights reserved.

API

API (Application Programming Interface) to use the functions of the USB-AIO10 board will be described.

```
//////////  
// DLL import header file for USB-AIO10  
// (c) 2012 DAQ system  
//////////  
  
#define SAMPLE_RATE_32K      0      // OSC = 33.554432MHz, ADC 32768sps(=33554432/2/512)  
#define SAMPLE_RATE_16K      1      // ADC 16384sps  
#define SAMPLE_RATE_8K       2      // ADC 8192sps  
#define SAMPLE_RATE_4K       3      // ADC 4096sps  
#define SAMPLE_RATE_2K       4      // ADC 2048sps  
#define SAMPLE_RATE_1K       5      // ADC 1024sps  
#define SAMPLE_RATE_512      6      // ADC 512sps  
#define SAMPLE_RATE_256      7      // ADC 256sps  
#define SAMPLE_RATE_128      8      // ADC 128sps  
#define SAMPLE_RATE_64       9      // ADC 64sps  
  
#define DAC_CH_0            0      // DAC CHANNEL  
#define DAC_CH_1            1      // DAC CHANNEL  
#define DAC_CH_2            2      // DAC CHANNEL  
#define DAC_CH_3            3      // DAC CHANNEL  
#define DAC_CH_ALL          7      // DAC CHANNEL  
  
//*****  
// Board level API functions  
//*****  
extern "C" __declspec(dllimport) BOOL __stdcall OpenDAQDevice(void);  
extern "C" __declspec(dllimport) BOOL __stdcall CloseDAQDevice(void);  
extern "C" __declspec(dllimport) BOOL __stdcall COM_PortOpen(int port_num);  
extern "C" __declspec(dllimport) BOOL __stdcall COM_PortClose(void);  
  
//*****  
// Digital IO API  
//*****  
extern "C" __declspec(dllimport) BOOL __stdcall DIO_Set(int data);  
extern "C" __declspec(dllimport) BOOL __stdcall DIO_Get(int *data);  
extern "C" __declspec(dllimport) BOOL __stdcall DIO_SetVerify(int *data);  
extern "C" __declspec(dllimport) BOOL __stdcall COM_DIO_Set(int data);  
extern "C" __declspec(dllimport) BOOL __stdcall COM_DIO_Get(int *data);  
  
//*****  
// Analog IN API  
//*****  
extern "C" __declspec(dllimport) BOOL __stdcall ADC_SetSampleRate(int type, int sample_rate);  
extern "C" __declspec(dllimport) BOOL __stdcall ADC_StartBufferedRead(void);  
extern "C" __declspec(dllimport) BOOL __stdcall ADC_StopBufferedRead(void);  
extern "C" __declspec(dllimport) int __stdcall ADC_GetBufferedData(int nCount, int* AinData);  
extern "C" __declspec(dllimport) int __stdcall ADC_GetBufferedPointer(int* WrPtr, int* RdPtr);  
extern "C" __declspec(dllimport) BOOL __stdcall ADC_GetData(int *data);  
extern "C" __declspec(dllimport) BOOL __stdcall COM_ADC_GetData(int *data);  
  
//*****  
// Analog OUT API  
//*****  
extern "C" __declspec(dllimport) BOOL __stdcall DAC_Set(int channels, int value);  
extern "C" __declspec(dllimport) BOOL __stdcall COM_DAC_SetData(int channels, int value);
```

Board Level API Functions

Overview

BOOL	OpenDAQDevice(void)
BOOL	CloseDAQDevice(void)
BOOL	COM_PortOpen(int port_num)
BOOL	COM_PortClose(void)

OpenDAQDevice

Check the registration of being use the USB device in the system, the board can be used the USB interface. USB function can be called only on board the normal registration.

BOOL OpenDAQDevice(void)

Parameters: None

Return Value:

If the function fail to reset, it returns “FALSE”. If the function succeed to reset, it returns “TRUE”.

CloseDAQDevice

Stop the USB interface function of the board, and terminate the device using in the system.

BOOL CloseDAQDevice(void)

Parameters: None

Return Value:

If the function fail to reset, it returns “FALSE”. If the function succeed to reset, it returns “TRUE”.

COM_PortOpen

Open the COM port for using the RS-232 interface function.

BOOL COM_PortOpen(int port_num)

Parameters:

port_num : Set the COM port number.

Return Value:

If the function fail to reset, it returns “FALSE”. If the function succeed to reset, it returns “TRUE”.

COM_PortClose

Close the opened COM port.

BOOL COM_PortClose(void)

Parameters: None

Return Value:

If the function fail to reset, it returns “FALSE”. If the function succeed to reset, it returns “TRUE”.

DIO(Digital Input/Output) API Functions

Overview

BOOL	DIO_Set (int data)
BOOL	DIO_Get(int *data)
BOOL	DIO_SetVerify (int *data)
BOOL	COM_DIO_Set(int data)
BOOL	COM_DIO_Get(int *data)

DIO_Set

Set the Digital Output by USB interface.

BOOL	DIO_Set (int data)
-------------	---------------------------

Parameters:

data : Output Setup Value, 24-bit data.

When set to “1”, the current flows between DOUT pin and DOUT_COM.

Return Value:

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

DIO_Get

Ceeck the Digital Input by USB interface.

BOOL	DIO_Get (int *data)
-------------	----------------------------

Parameters:

***data** : Input Setup Value, 24-bit data.

When read to “1”, the current flows the loop formed DIN_COM and DIN.

Return Value:

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

DIO_SetVerify

Check the setup value at Digital Output by USB interface.

BOOL DIO_SetVerify (int *data)

Parameters:

***data** : Output Setup Value, 24-bit data.

Return Value:

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

COM_DIO_Set

Set the Digital Output by RS-232 interface.

BOOL COM_DIO_Set (int data)

Parameters:

data : Output Setup Value, 24-bit data.

When set to “1”, the current flows between DOUT pin and DOUT_COM.

Return Value:

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

COM_DIO_Get

Check the Digital Input by RS-232 interface.

BOOL COM_DIO_Get (int *data)

Parameters:

***data** : Input Stste Value, 24-bit data.

When read to “1”, the current flows the loop formed DIN_COM and DIN.

Return Value:

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

ADC(Analog to Digital Convertor) API Functions

Overview

BOOL	ADC_SetSamplerate(int type, int sample_rate)
BOOL	ADC_StartBufferedRead(void)
BOOL	ADC_StopBufferedRead(void)
int	ADC_GetBufferData(int nCount, int *AinData)
int	ADC_GetBufferedPointer(int *WrPtr, int *RdPtr)
BOOL	ADC_GetData(int *data)
BOOL	ADC_GetSingleData(int *data)
BOOL	COM_ADC_GetData(int *data)

ADC_SetSamplerate

Set the ADC data sampling rate by USB interface.

BOOL ADC_SetSamplerate(int type, int sample_rate)

Parameters:

type : Reserved.

Sample_rate : Sampling Rate Code. Sampling frequency and code is as follows.

SAMPLE_RATE_32K	0	// ADC 32768sps
SAMPLE_RATE_16K	1	// ADC 16384sps
SAMPLE_RATE_8K	2	// ADC 8192sps
SAMPLE_RATE_4K	3	// ADC 4096sps
SAMPLE_RATE_2K	4	// ADC 2048sps
SAMPLE_RATE_1K	5	// ADC 1024sps
SAMPLE_RATE_512	6	// ADC 512sps
SAMPLE_RATE_256	7	// ADC 256sps
SAMPLE_RATE_128	8	// ADC 128sps
SAMPLE_RATE_64	9	// ADC 64sps

Sampling Rate [sps] = 32,768/(2^Setup value)

Return Value:

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

ADC_StartBufferedRead

Start the AD Data collection by USB interface. The collected data is stored to the library buffer. Buffer that stored to the library can save the 8M data.

BOOL ADC_StartBufferedRead(void)

Parameters: None

Return Value:

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

ADC_StopBufferedRead

Stop the AD Data collection by USB interface.

BOOL ADC_StopBufferedRead(void)

Parameters: None

Return Value:

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

ADC_GetBufferData

Read the collected AD Data by USB interface from the library. The data value is 16-bit integer, the range is 0~5V(20mA).

int ADC_GetBufferData(int nCount, int *AinData)

Parameters:

nCount : Number of data to be read. Max. 4,194,034(4M).

***AinData** : Buffer pointer, the read data will be stored.

Return Value:

AinData : Actual stored data is returned. nCount is equal or smaller.

ADC_GetBufferedPointer

Read the buffer pointer of the library by USB interface. *WrPtr is a saved pointer that obtained from USB, *RdPtr is a read pointer from the program.

int ADC_GetBufferedPointer(int *WrPtr, int *RdPtr)

Parameters:

*WrPtr : Pointer that read and save the data from the USB.

*RdPtr : Data pointer that read from the application.

Return Value:

Returns the number of data that can be read.

ADC_GetData

Read the current Analog Input AD data by USB interface.

BOOL ADC_GetData(int *data)

Parameters

*data : Buffer pointer to save the 4 channel data. Storage space shall not be less than 4.

First channel save to the Pointer 0 in turn.

Return Value:

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

ADC_GetSingleData

Read the current ADC data.

BOOL ADC_GetSingleData(int *data)

Parameters

*data : ADC 4 channels of AD conversion data is stored in an integer array..

4 data saves to the Pointer 0 in turn.

Return Value:

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

COM_ADC_GetData

Read the current Analog Input AD data by RS-232 interface.

BOOL COM_ADC_GetData(int *data)

Parameters

***data** : Buffer pointer to save the 4 channel data. Storage space shall not be less than 4.

First channel save to the Pointer 0 in turn.

Return Value:

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

DAC(Digital to Analog Convertor) API Functions

Overview

BOOL DAC_Set(int channels, int value)
BOOL COM_DAC_SetData(int channels, int value)

DAC_Set

Set the DA output by USB interface. 0~5V(20mA) output set by 16-bit data.

BOOL DAC_Set(int channels, int value)

Parameters:

channels : Select the desired channel.

DAC_CH_0	0	// DAC CHANNEL
DAC_CH_1	1	// DAC CHANNEL
DAC_CH_2	2	// DAC CHANNEL
DAC_CH_3	3	// DAC CHANNEL
DAC_CH_ALL	7	// DAC CHANNEL

value : Set the output value. 0~65,535(Voltage 0~5V, Current 0~20mA)

Return Value:

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

COM_DAC_Set

Set the DA output by RS-232 interface. 0~5V(20mA) output set by 16-bit data.

BOOL COM_DAC_Set(int channels, int value)

Parameters:

channels : Select the desired channel.

DAC_CH_0	0	// DAC CHANNEL
DAC_CH_1	1	// DAC CHANNEL
DAC_CH_2	2	// DAC CHANNEL
DAC_CH_3	3	// DAC CHANNEL
DAC_CH_ALL	7	// DAC CHANNEL

value : Set the output value. 0~65,535(Voltage 0~5V, Current 0~20mA)

Return Value:

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