280x USB Driver DLL

The 280x USB Driver DLL was created as a dual mode driver. It can be used by standard C/C++/LabView application as well as a referenced object in Visual Basic and other languages. Depending on the mode used, there are two sets of function calls. Below outlines the two different modes.

In order to utilize this driver a 280x device must be plugged in to the PC. Once the PC enumerates this device, this DLL will be able to locate it within the system. There are two function calls to assist you.

C/C++/LabView (mode)

```
char *USB_Get_First_DeviceName()
char *USB_Get_ALL_Available_DeviceName()
```

Visual Basic (mode)

```
Get_First_DeviceName() as string
Get_ALL_Available_DeviceName() as string
```

Each of this functions returns string that represents the internal name of the device. Since more that one device can be plugged in to the PC at the same time, each device is given its own name.

USB_Get_First_DeviceName/Get_First_DeviceName returns the first name in the list of devices while the USB_Get_ALL_Available_DeviceName/ Get_ALL_Available_DeviceName returns a string array, delimited by a CR, of all the available devices. To use device 2, 3 or so on, the application will need to provide the user with a list of devices to choose from.

This device name will be used by the other functions calls as a reference to the physical device.

To send a command to the device, use the USB_Write/Device_Write function.

C/C++/LabView (mode)

```
long USB_Write(char *DeviceName, char *Buffer, long Buffer_Len )
```

Visual Basic (mode)

```
Device_Write(DeviceName As String, Value As String) As Long
```

This function returns one of two values, 0 indicates a failure and a non zero indicates success.

Depending on the Command sent to the device, there are two possible channels the results can be returned on, Interrogative or Formatted. Using the appropriate read function to obtain the desired results.

Both functions act the same. Pass Device Name and a pointer to buffer. Note: The Buffer needs to allocate to the largest possible response. This function does not allocate buffer space; it simply copies the buffer from the device driver to your buffer. The function will return the number of bytes read.

C/C++/LabView (mode)

```
long USB_Read_Formatted(char * DeviceName, char *Buffer)
long USB_Read_Interrogative(char * DeviceName, char *Buffer)
```

Visual Basic (mode)

```
Device_Read_Interrogative(DeviceName As String) As String
Device_Read_Formatted(DeviceName As String) As String
```