DeltaLINK DL4Api SDK

Version 1.0, 25 March 2009

Contents

DeltaLINK DL4Api SDK

CD Contents

This DL4Api SDK contains the following files:

DeltaLINK DL4Api SDK.pdf (this document)

DL4ApiDemo Visual Studio 2008 project

This Includes the dependencies DL4Api.dll, DL4Api.h, DL4Api.lib, DL4Defs.h.

These are installed to a DL4Api SDK folder in your Documents folder.

Features of DL4Api

DL4Api SDK provides the following functionality to DL6 and GP1 logger:

Set Logger Time

Start and Stop logging

Download and save datasets from logger

Extract records from datasets

Take single or multiple real time readings from logger (as in DeltaLINK Sensors Tab)

DL4Api does not construct logging programs, these have to be created by DeltaLINK.

How to…

Open Logger Connection

To open a logger connection use DL4OpenLogger with the follow parameters:

pszPort, is the connection name, if using RS232 connection this will normally be COM1: or another COM port. This also could be a modem name and the number, the two being separated by ~~~ for example “standard 56k modem~~~012345567789”.

dwBaud will always be 115200.

pnSerNo is the serial number of the logger, that’s being connected to if know, if not then DL4SERNO_ANY is used.

pnIncarnation is the logger incarnation if known, otherwise INCARNATION_ANY. The incarnation changes every time a new program is applied to the logger.

pnFirmware is the firmware version of the logger that has be connected to, this is a output.

phDL4 is the handle for the logger that has be connected to.

To close the connection use DL4LoggerClose() with the logger handle to be closed.

Set logger clock

Use DL4LoggerSetTime(), where hDL4 is the logger handle, and time is the 32bit time_t time value.

Logging must be stopped to set the time, functions return DL4ERROR_LOGGING if logging.

If logger contains a dataset the time being set must not be <= to a time stamp of a existing record. Function returns DL4ERROR_INVALIDTIME if an existing record has a time stamp later the time being set.

Start and Stop Logging

To determine the logging status of the logger call DL4Logger Get St at us( ) Examine DL4St at us. pr og. bLoggi ng to determine the logging status of a logger.

Before starting and stopping logging, the start and stop tasks need to be found in the logger program. This is done by getting the program from the logger, and stepping through it identifying the various objects.

Call DL4Logger Get Pr ogr am( ) to retrieve the program from the logger, the size of pcPr og is found in DL4St at us. pr og. nSi z e.

Call DL4Pr ogr amGet I nf o( ) to get DL4Pr ogI nf o, which has the number of measures, variables and number of tasks, and the program name.

Now iterate through the tasks calling DL4Logger Get Obj ect I nf o( ) for each. Inspect DL4Obj ect . nObt ype bit field for DL4OB_START and DL4OB_STOP

Once the Start and Stop task handles have been found use DL4Logger Set Obj ect Val ue( ) to start or stop the logging process, with hObj ect as the task handle and val . t = 1.

Download and Save dataset from logger.

Use DL4Logger Get St at us( ) to get the size of the dataset as the number of segments, (DL4St at us. segs. nOccup). Create a Dataset with DL4Logger Cr eat eDat aset ( ) providing the size as nFi r st and nSegment s (normaly 0, and DL4St at us. segs. nOccup). Download the dataset with DL4Logger Get Dat aset Segment s( ) supplying nFi r st and nSegment s as with DL4Logger Cr eat eDat aset ( ).

Save the dataset to file with DL4Dat aset SaveFi l e( ) with the dataset handle, and the filename, flags is not used set to 0. This save the file in same format as dt6 files saved with DeltaLINK.

Close the dataset when done with DL4Dat aset Cl ose( )

Extract records from a dataset.

Once a dataset has been created by either downloading from logger with DL4Logger Cr eat eDat aset , and DL4Logger Get Dat aset Segment s( ), or by opening a dataset file with DL4Dat aset Open( ), the dataset can be walked through, to read the individual records.

Call DL4Dat aset Get I nf o( ) to get the information about the dataset.

Allocate a DL4Obj ect array with nSer i es elements.

Iterate through the DL4Obj ect array calling DL4Dat aset Get Ser i esI nf o( ) with nSer i es, and pObj ect [nSer i es] .

Allocate an array of DL4Val ue and an array of unsigned integer (pnSer i es), both being nMaxRecor d in size.

Now for each record call DL4Dat aset Get Recor d( ) .

For each series, 0 to nSer i es, iterate through pnSer i es to find the position in pVal ues for each series, then use pObj ect [ ] . nType to determine the type of value, for DL4TYPE_FLOAT use f , for DL4TYPE_I NT use i and for DL4TYPE_TI ME use t .

When finished close the dataset with DL4Dat aset Cl ose( )

Take instant readings

To take real time readings the handles to the measures (or channels) in the program have to be found.

Get a current DL4Status with DL4LoggerGetStaus(), use DL4LoggerGetProgram() to get the current logger program, then DL4ProgramGetInfo to get DL4ProgInfo.

Call DL4LoggerGetObjectInfo for each DL4ProgInfo.nMeasures, counting the number of Measure that DL4Object.nObtype has the DL4OB_PUBLIC bit set, to give then number of reading to take (nObjects).

Single instant reading

To take a single reading, once the number of readings has been found (nObjects), then allocate an array of DL4Result's of length nObjects.

Then call DL4LoggerReadObjectValues(), where nAction is DL4API READ_ONCE, nObjects is the number of readings obtained eairler, phObjects is an unsigned integer array of nObjects in size, and pResults and array of DL4Result's with the length of nObjects.

To get the readings, iterate through phObjects call DL4LoggerGetObjectInfo(hDL4, phObjects[x], &ob).

ob.pszName, contains the name for the channel.

To determine the type of each reading inspect ob.nType, and the reading is found in pResult array, for DL4TYPE_FLOAT use f, for DL4TYPE_INT use i and for DL4TYPE_TIME use t.

Multiple instant readings

To take multiple readings, the logger is first setup to take multiple readings, and then polled for each reading. To do this get the number of readings as above, and call DL4LoggerReadObjectValues, with nAction value as DL4API READ_SETUP, and to get the readings call DL4LoggerReadObjectValues, with nAction value as DL4API READ_POLL. DL4LoggerReadObjectValues() does not return until the reading has completed, the reading are done on 1 second intervals, as in DeltaLINKs sensor tab.

Once the number of readings required have be taken, then call DL4LoggerReadObjectValues(), with nAction values as `.

To get access to the reading is the same process as in the single reading process.