LabVIEW support for the AvaSpec

The AvaSpec DLL uses Windows messages by preference to signal the arrival of new data to the user program. Unfortunately, LabVIEW does not support these Windows Messages very well. The (old) 203639.zip archive from the NI website lets you use Windows messages (on 32 bit Windows), but our tests show no performance improvements at all over polling.
The AvaSpec DLL also supplies a function that allows you to poll for the arrival of new data, called AVS_PollScan. This function should be called from a loop with a short delay in it, in order to let Windows handle updating of variables. If you call the AVS_PollScan function from a tight loop, it will always return false, even though new data has arrived.
The three polling demos and the LabViewSingleChan demo use the AVS_PollScan function. The delay in the loop is always present, either explicit, or in the form of the Event Timeout of the main event structure that is used.

We have received some requests for faster processing than is possible with the AVS_PollScan function. We have also received some reports of problems with this function, mainly on fast PCs with 32 bits Windows.
For this reason, we have looked into other methods to signal data besides polling.

Another disadvantage of LabVIEW, when working with the AvaSpec DLL is the use of several nested, byte aligned structures in the DLL that are poorly supported by LabVIEW. These structures have to be translated to and from a linear array of bytes, to prevent data corruption as LabVIEW will pad these structures to get 16 or 32 bit alignment. A more elegant solution to this issue would be to handle these structures in an intermediate DLL rather than in LabVIEW itself.

Available are the following new solutions:

1) The use of a custom user event. This is implemented with a separate AVS_MeasureLV function. This function will trigger a user event, that can be received in your LabVIEW program. Please note that the function is only available in the AvaSpec DLL, or the Ethernet series of the AS5216 DLL (the 8.x series). This version of the DLL is written in Microsoft C++, which is necessary to link the library from NI that supplies the PostLVUserEvent function used. The standard AS5216 DLL (the 2.x series) is written in Borland/Embarcadero C++, which is not compatible with this library. Three demos using the event are supplied in the 'events' subdirectory, for one, two and four spectrometers.

2) An intermediate DLL that handles Windows messages. Versions for such a library in Delphi are supplied, both for a single channel setup and for a multichannel setup. This DLL does not signal back to the LabVIEW program, but handles the data acquisition directly. In the present implementation, it waits for all channels to have received data, before reading and transferring the data back to LabVIEW. The intermediate DLL also reduces the number of calls necessary for data acquisition. You can also set the measurement parameters separately, instead of setting them all at once. The source code of the intermediate DLLs is supplied. Two demos are supplied, for one and two spectrometers, in the 'messaging' subdirectory.

3) An intermediate DLL that uses the custom user event to signal the arrival of new data, while using polling itself. This DLL was written in Microsoft C++, but it can be used in combination with the standard AS5216 DLL. Separate functions to set the different measurement parameters are available. The source code of the intermediate DLL is supplied. Two demos are supplied, for one and two spectrometers, in the 'intermediate' subdirectory.

Reference section

1) Custom user event implemented with separate AVS_MeasureLV function

```
Function: int AVS_MeasureLV   (
                                    AvsHandle a_hDevice,
                                    LVUserEventRef *msg,
                                    int param,
                                    short a_Nmsr
                              )
```

Group:	Non-Blocking data write function	
Description:	Starts measurement on the spectrometer, triggers LabVIEW user event	
Parameters:	a_hDevice:	device identifier returned by AVS_Activate
	msg:	user event reference from LabVIEW
	param:	not used, e.g. pass a constant 0 here.
	a_Nmsr:	number of measurements to do after one single call to AVS_MeasureLV (-1 is infinite)
Return:	On success:	ERR_SUCCESS
	On error:	ERR_OPERATION_PENDING
		ERR_DEVICE_NOT_FOUND
		ERR_INVALID_DEVICE_ID
		ERR_INVALID_PARAMETER
		ERR_INVALID_STATE

Please note that the AVS_MeasureLV function is only present in the AvaSpec DLL and the Ethernet version (the 8.x series) of the AS5216 DLL.

2) Intermediate DLL that uses Windows messages to signal arrival of new data

Functions:

For a single channel setup:

```
init
done
getnumpix (var numpix:word)
getlambda (var lambda:pixelarray)
prepare(paramname:pChar;value:double)
measure
getdata(var timelabel:DWord;var data:pixelarray)
stop
```

For a multichannel setup:

```
init(var ldevices:handlearray)
done
getnumpix(hDevice:integer;var numpix:word)
getlambda(hDevice:integer;var lambda:pixelarray)
prepare (hDevice:integer;paramname:pChar;value:double)
measure (hDevice:integer)
dataready
getdata (hDevice:integer;var timelabel:DWord;var data:pixelarray)
stop (hDevice:integer)
```

pixelarray is an array of 4096 doubles
paramname is one of the following names of parameters to set:

```
INTEGRATIONTIME
NRAVERAGES
STARTPIXEL
STOPPIXEL
INTEGRATIONDELAY
CORDYNDARK
SMOOTHING
TRIGGERMODE
SATURATIONDETECTION
NRSTORETORAM
```

Handlearray is an array of 10 integers

LabVIEW translations of these functions are available in the samples. Source code in Delphi version 6 is supplied. C style rather than pascal style prototypes can be found as well. If you click in the 'Call Library Function' icon in each subvi, a C style function prototype will be shown.

3) Intermediate DLL that uses custom user event to signal arrival of new data

Functions:

init
done
getnumpix
getlambda
setintegrationtime
setnumaverages
measure
getdata
stopscan
setstartpixel
setstoppixel
setintegrationdelay
setcordyndark
setsmoothing
settriggermode
settriggersource
settriggersourcetype
setsaturationdetection
setnrstoretoram
setsyncmode
deactivate
activate
getsaturated

LabVIEW translations of these functions are available with the demos. Both a demo for a single spectrometer setup and a dual spectrometer setup are supplied. Source code in Microsoft Visual C++ 2008 is supplied.
The debug version of the DLL will show debug messages in a separate window. Use this to debug your program. Use the release version to dispose of the debug window.