Reading instruNet Files with 3rd Party Software.

Application Note #48, 5/12/2003

instruNet World supports the following different file types for waveform data:

Text - Data stored as a text file, with one file for each channel. Open the file with a word processor or spreadsheet to see how it's saved.

Text Merge - Same as text, yet includes an additional text file called "Excel Waveform Data.txt" which contains all channels in one file with one column devoted to each channel.

Binary - Data is stored in binary form, with one file for each channel. A binary header is stored at the beginning of the file, in the form of one GWI_file_header_struct struct (defined in file INET_INT.H), and this is followed by the wave data, stored in a one dimensional array of 32bit floating point numbers (i.e. C 'float' type). The 1st 4 bytes in the header is the length of the header in bytes (i.e. 516 = 0x0204).

Binary Merge - This is similar to Binary, yet all channels are stored in one file, in an interlaced format. BinaryMerge supports faster throughput rates than Binary, since the hard disk writing head stays in on place on the platter when writing multiple channels (as opposed to whipping back and forth between separate files for each channel). The beginning of the BinaryMerge file contains an array of GWI_file_header_struct structs (defined in file INET_INT.H) for the various channels, followed by the interlaced 32bit floating point data. The 1st 4 bytes in the header is the length of the header for 1 channel in bytes (i.e. 516 = 0x0204). To read the # of channels stored in the file, read the 'channelsPerFile' field of the 1st struct (e.g. to see how many headers are there). After the header, the data is saved in an interlaced form, where points are stored in the order that they are acquired in time. For example, data from 3 channels (A, B and C) at the same sample rate would be stored as:

```
A[0], B[0], C[0], A[1], B[1], C[1], A[2], B[2], C[2]......
```

If Channel B was stored at 1/N of the sample rate where N is 2 (N is ALWAYS and integer, and is defined in the 'sampleRate_Divider' field), the data would be packed in the following way:

```
A[0], B[0], C[0], A[1], C[1], A[2], B[1], C[2], A[3], C[3], A[4] ......
```

The binary data is stored in intel little endian (Windows) or 68k big endian (Macintosh) format, as noted by the 'file_endian' field in the header. Use the 'int32key', 'int32key_StructTest', 'int16key' and 'flt32key' fields to make sure you are reading things correctly. They are preset to specific values, as noted in file INET_INT.C. If you save the file on a MS-Windows computer, and load it on an MS-Windows computer, things should be ok. Byteswapping is necessay when saving on a bigEndian computer (e.g. Macintosh) and then loading on a littleEndian computer (e.g. Windows 95).

Using the instruNet Driver to Read/Write data to/from Disk

The instruNet driver can also be used to read/write data to/from disk. To read/write data to/from an instruNet RAM BUFFER (user or driver), one can send a command to the 'command' field within the FILE Setting Group of a channel. To learn more about this, please read the INET_INT.C documentation on 'fldNum_File_fileCmd' and 'ion_FileCmdPopup'; and also strudy ExerciseFileSaveLoadCode() in file INET_EX1.C. These routines fascillitate paging in data that was spooled to disk. For example, if you spool 1000 scans todisk with10K ptsPerScan (10M total pts) and need to load into ram one scan at a time, for post acquisition processing, then one could send fileCmd_FileToRamBuffer or fileCmd_FileToUserBuffer commands into the instruNet driver to fascillitate the paging in process.