Installation and Operation Manual
ProSeries Model SPS390
Dynamic Signal Analyzer
Part Seven

Legacy Manual

COGNITIVE VISION, INC.
7220 Trade Street, Suite 101
San Diego, CA 92121-2325 USA

instruments@cognitivevision.com
www.cognitivevision.com

Tel: 1.858.578.3778 / Fax: 1.858.578.2778
In USA: 1.800.VIB.TEST (842.8378)

LEGACY MANUAL POLICY

Cognitive Vision Legacy manuals are those product manuals and documentation that accompanied earlier products and product lines which have since been discontinued ("Legacy Products"). Over the past thirty years, these include products that were sold by Spectral Dynamics, Scientific Atlanta and Smiths Industries.

Cognitive Vision, Inc. provides downloadable copies of these manuals strictly as a courtesy to its customers who continue to use Legacy Products. IMPORTANT: Please read the following Disclaimer carefully. Any use of this manual indicates your express agreement with this policy.

If you have any questions regarding this policy, or for additional information regarding the serviceability of any Legacy Product(s), please call our service department.

DISCLAIMER

IN DOWNLOADING THIS MANUAL, THE USER UNDERSTANDS AND EXPRESSLY AGREES THAT COGNITIVE VISION MAKES NO WARRANTIES WHATSOEVER, EITHER EXPRESS OR IMPLIED, INCLUDING ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN USING THIS MANUAL, THE USER ACKNOWLEDGES THAT ALL PREVIOUS PRODUCT WARRANTIES ISSUED BY SPECTRAL DYNAMICS, SCIENTIFIC ATLANTA AND SMITHS INDUSTRIES FOR LEGACY PRODUCTS HAVE SINCE EXPIRED.

IN PROVIDING THIS MANUAL, COGNITIVE VISION ASSUMES NO LIABILITY OR RESPONSIBILITY WHATSOEVER TO THE USER OF THIS MANUAL, THE USER'S AGENTS AND/OR CUSTOMERS, OR ANY OTHER PARTY, FOR ANY CLAIMED INACCURACY IN THIS MANUAL, OR FOR DAMAGE CAUSED OR ALLEGED TO BE CAUSED DIRECTLY OR INDIRECTLY BY ANY USE OF THIS MANUAL, REGARDLESS OF WHETHER COGNITIVE VISION WAS INFORMED ABOUT THE POSSIBIILITY OF SUCH DAMAGES, OR FOR ANY CLAIM MADE AGAINST THE USER'S ORIGINAL PRODUCT WARRANTY.

FURTHER, COGNITIVE VISION SHALL NOT BE RESPONSIBLE FOR ANY INTERRUPTION OF SERVICE, LOSS OF BUSINESS, ANTICIPATORY PROFITS, CONSEQUENTIAL DAMAGES, OR INDIRECT OR SPECIAL DAMAGES ARISING UNDER ANY CIRCUMSTANCES, OR FROM ANY CAUSE OF ACTION WHATSOEVER INCLUDING CONTRACT, WARRANTY, STRICT LIABILITY OR NEGLIGENCE.

NOTWITHSTANDING THE ABOVE, IN NO EVENT SHALL COGNITIVE VISION'S LIABILITY TO THE USER EXCEED AN AMOUNT EQUAL TO THE REPLACEMENT COST OF THIS MANUAL.

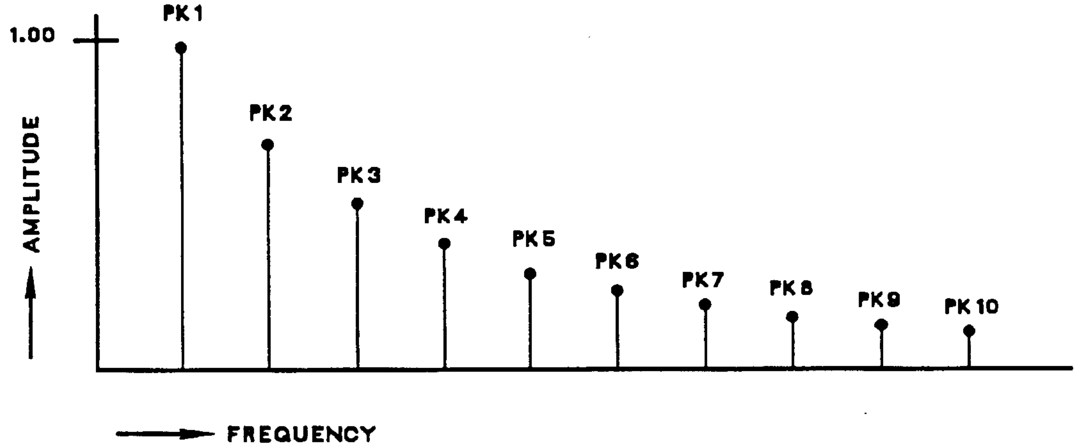
COGNITIVE VISION, INC.
7220 Trade Street, Suite 101
San Diego, CA 92121-2325 USA

analyzers@cognitivevision.com
www.cognitivevision.com

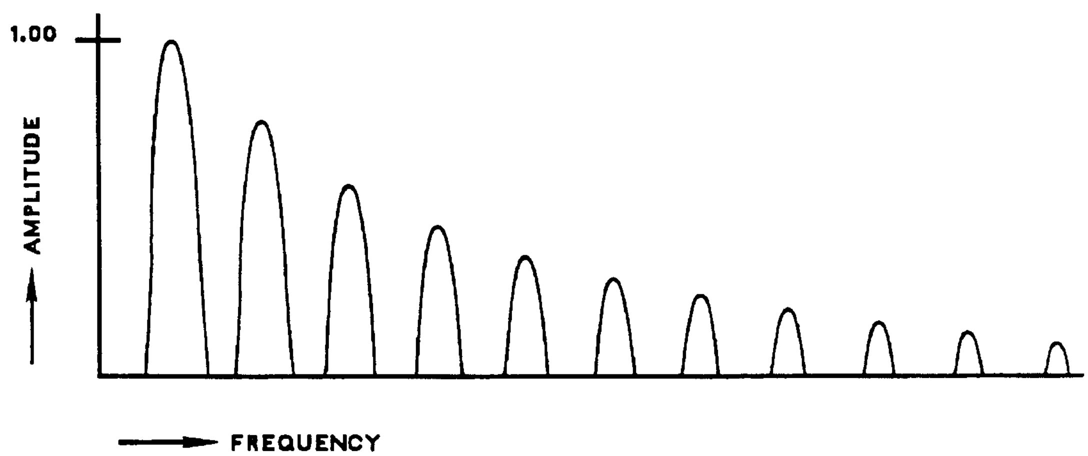
Telephone: 1.858.578.3778 / Fax: 1.858.578.2778
IN USA: 1.800.VIB.TEST (842.8378)

4–1.2 FFT Weighting

The earliest spectrum (or harmonic) analyzers consisted of tunable, high Q filters with a frequency dial and a standard analog voltmeter connected to the output of the filter. As the operator swept across the frequency band of interest, the meter would "peak" at various frequencies. The operator, typically, would plot (by hand) the frequency and magnitude of the peaks. The plot may have looked like the following figure.

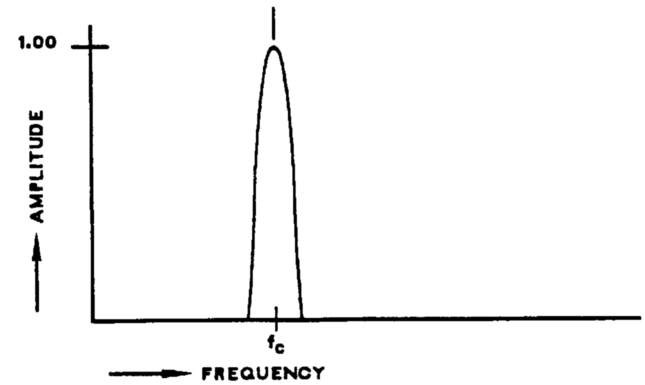


The previous plot, however, is not a true representation of a spectrum plot. If the frequency dial and voltmeter readings of the analyzer were connected to the X– and Y–axis of an X–Y recorder, the obtained plot would resemble the following figure.



The harmonic plot of the previous figure is the result of tuning a filter past a stationary signal that contains more than one frequency component. If the filter is tuned to a frequency and a swept frequency signal is applied, the resultant filter response would resemble the following figure.

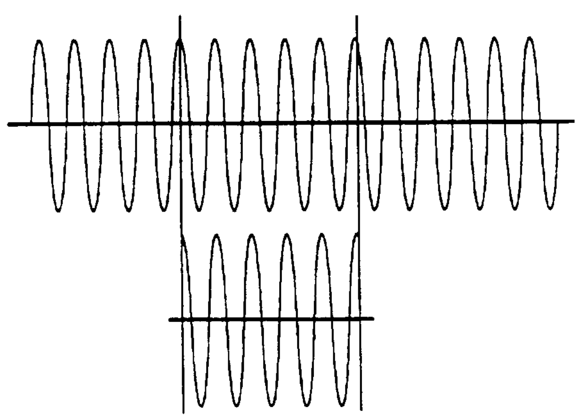
Filter Response of a High Q Filter



A digital spectrum analyzer carries the spectrum analysis concept even further. It simultaneously displays the output of up to 1600 equally spaced, digital filters. This is accomplished by using a high speed discrete Fourier Transform called the Fast Fourier Transform (FFT). FFT analysis allows the rapid transformation of discrete time domain data into frequency domain data. The end result is up to 1600 values representing the simultaneous output of each high Q filter.

Since the FFT is performed on a finite sample, incongruities such as side lobes, spurious noise, etc., can cause false data to be acquired. Weighting is a technique for treating data in the time domain, before the FFT, to reduce the effect of side lobes, etc. The FFT of the signal shown in the following figure is performed on the data in the sample window.

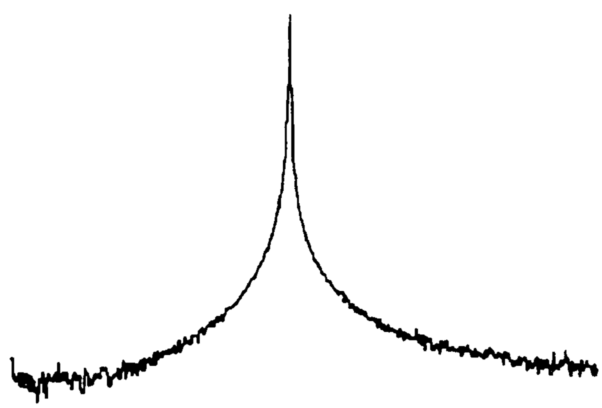
Sample Window



The following illustrations show the difference between "true" spectrum data and FFT data.

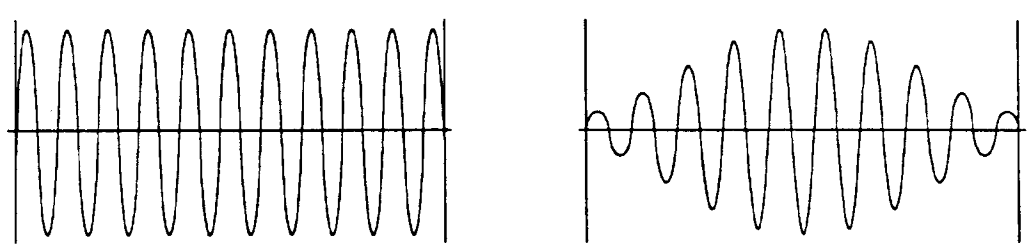
"True" Spectrum of the Sample Window Data

FFT of the Sample Window Data



This phenomenon is called "flaring" or "leakage" and can be caused by "Rectangular" weighting, since uniform weighting is applied to the data before the FFT is performed. Weighting eliminates the flaring by smoothing the transitions on each side of the sample window.

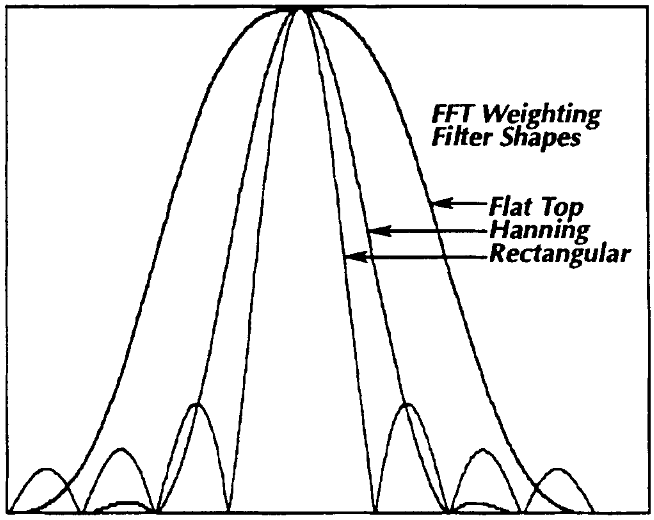
Weighting Effects on the Sample Window Signal



Sample Window ***Weighting "Envelope"***

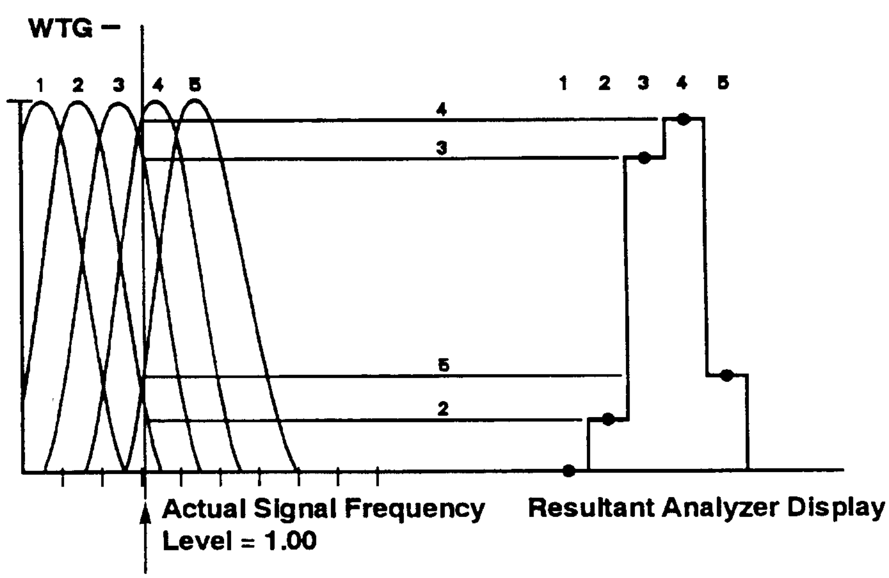
An additional effect weighting has on the data is the effective shape of the digital filters that produce the display. This filter shape could be measured by slowly sweeping a signal through the filter and measuring the response. A "swept sine" plot of a single filter would produce the following filter shapes for Rectangular, Hanning, and Flat Top weighting:

Filter Shapes of the Various Weighting Functions



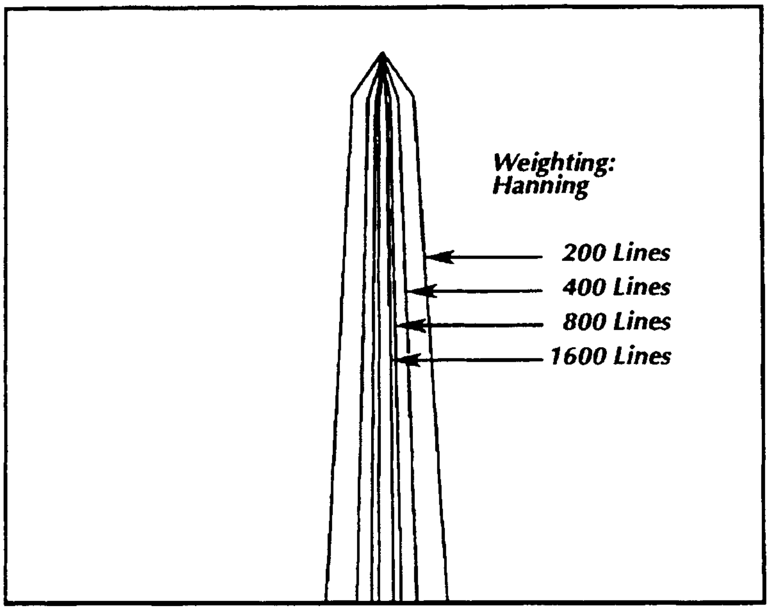
There are up to 1600 filters overlapping each other on the SPS390 display. The displayed data are actually the vectors between the output data points at the applied frequency. The following figure shows the analyzer display of a signal measured with Hanning weighting.

Generation of a Weighted Analyzer Display

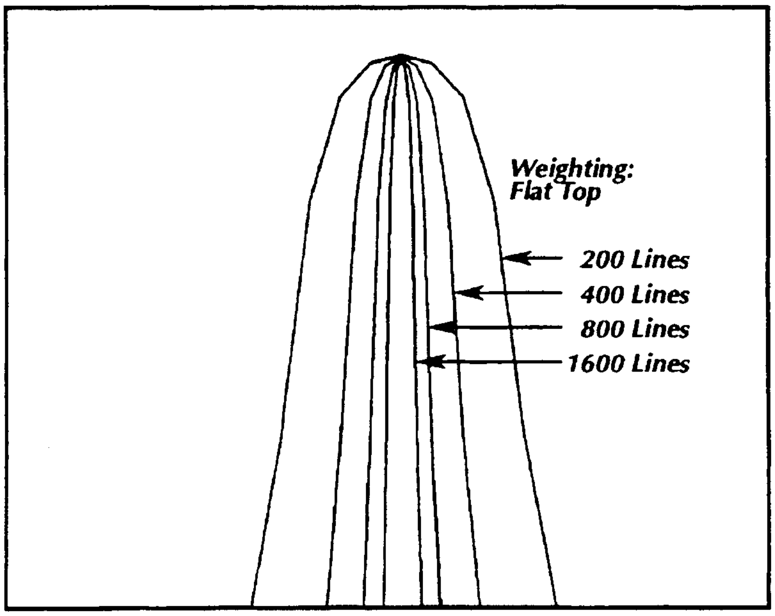


The following three figures are examples of the first three weighting functions from the **Process Control** menu. The applied input signal was identical for all three weighting functions. In addition, the actual width of the filters depends upon the selected resolution (number of lines). Each weighting function shown in the following three illustrations also shows how resolution selection affects filter width.

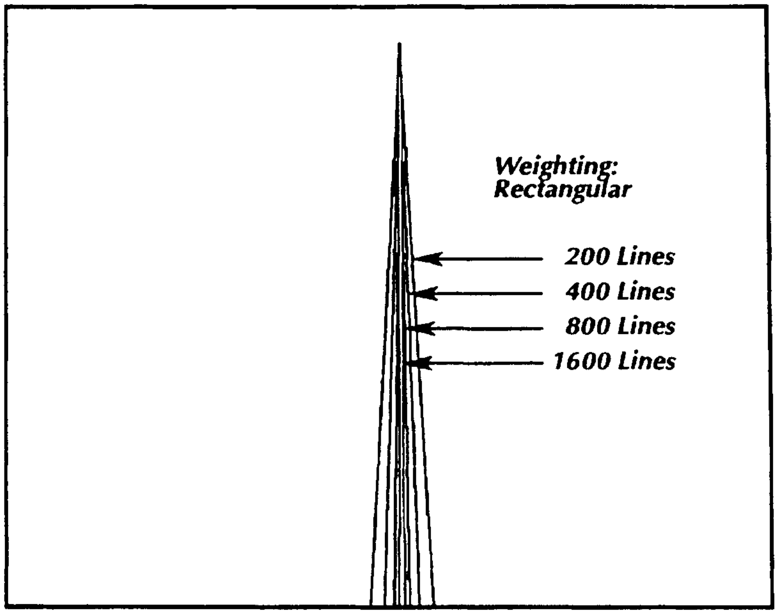
Hanning Weighting Example



Flat Top Weighting Example



Rectangular Weighting Example

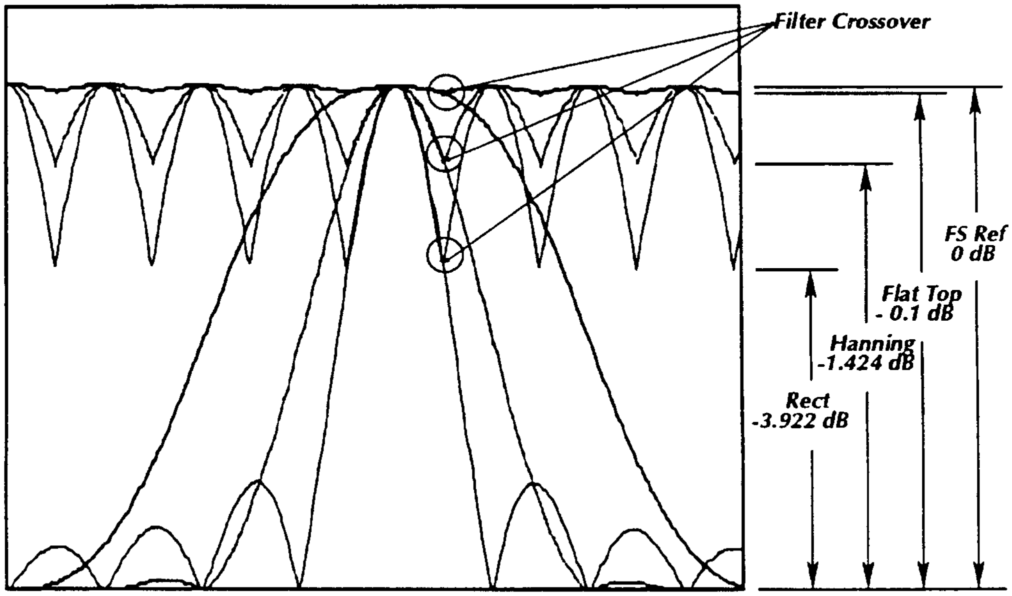


If we were to plot the measured signal peak versus input frequency across the weighting filters, we would notice an effect called "scalloping." It (scalloping) reflects the variation in amplitude of a measured tonal as it passes into and out of the digital filter centers. Examples of this effect are shown in the following figure. Note that as the tonal passes from the center of one filter to the center of the next filter, the amplitude of the tonal will decrease as it passes between the filter centers (the point at which the filters overlap) and then increase as it approaches the next filter center. This variation in amplitude reflects the percentage of scalloping for each weighting-function filter shape.

In an environment where the amplitude of a signal (tonal) is extremely important, Flat Top weighting can insure that filter shape does not affect measured amplitude. For example, if you were to sweep a signal across a particular band, you would notice that the "next filter" would rise to signal strength before the "current filter" started to descend.

Harris Flat Top weighting (the method of Flat Top weighting the SPS390 uses) gives you the necessary flat response without sacrificing any more frequency resolution than necessary.

Hanning, Flat Top and Rectangular Scalloping Examples

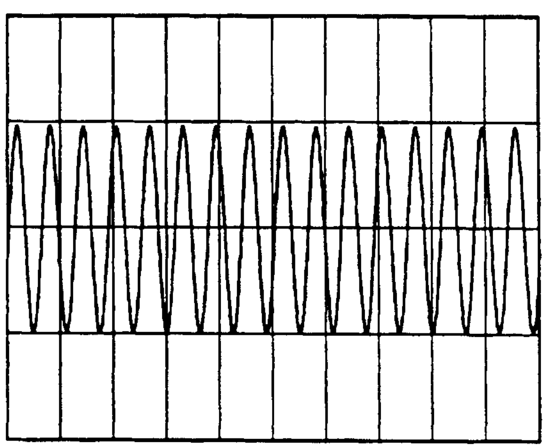


4–1.2.1 Response Weighting

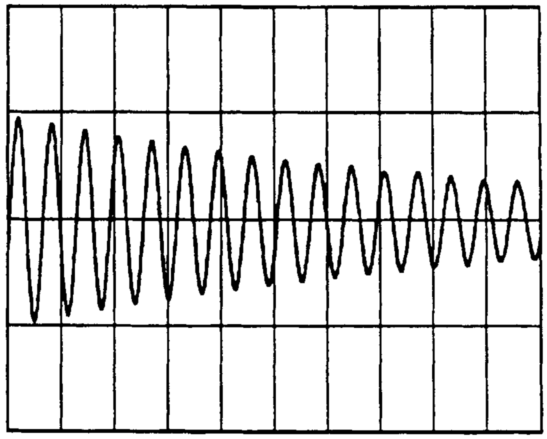
The remaining selection on the **Process Weighting** menu, Response, is a special weighting algorithm. This weighting algorithm enhances signal analysis for impact testing and other types of transient response testing.

Response Weighting applies an exponential damping envelope to the data. The TAU menu that appears when Response Weighting is selected allows for choosing different amounts of damping. The following illustrations are examples of TAU selections 1 through 4 and show the effects of the response window when applied to SPS390 time domain displays. This allows you to visually ensure the proper TAU for the unit under test.

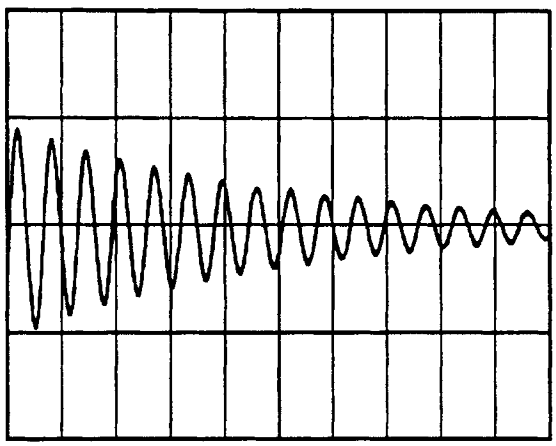
Example of a Signal Before Response Weighting Is Applied



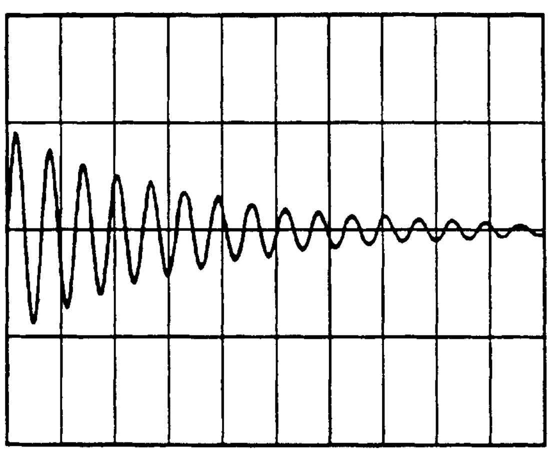
Response Weighting With a TAU of 1. Last Cell Is Attenuated 1/e



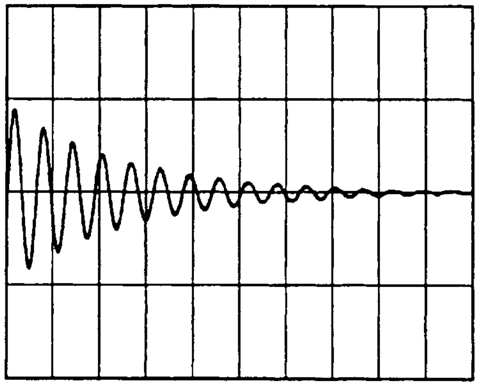
Response Weighting With a TAU of 2. Last Cell Is Attenuated $1/e^2$



Response Weighting With a TAU of 3. Last Cell Is Attenuated $1/e^3$



Response Weighting With a TAU of 4. Last Cell Is Attenuated $1/e^4$



The primary application for response weighting is narrow-width impulse type data. This type of signal data may have a poor signal-to-noise characteristic. Noise information that is 60 dB down may be significant when compared to an impulse which may have a total area on the order of one or two cells. Even though the noise information has an amplitude at any one cell of about 1/1000th of the amplitude of the impulse, the noise occupies all 1024 cells. Hence the noise rms is on the same order as the impulse rms.

Using trigger delays to position the impulse close to the beginning of the Memory Period, Response Weighting can be used to zero out most of the noise and vastly improve the signal-to-noise characteristic of the measurement.

4–1.2.2 FFT Weighting Summary

Weighting selections are totally dependent on the desired data. Some basic rules of thumb to follow:

- Hanning → Good compromise.
- Rectangular → Superb frequency resolution, poor amplitude accuracy.
- Flat Top → Poor frequency resolution, superb amplitude accuracy.

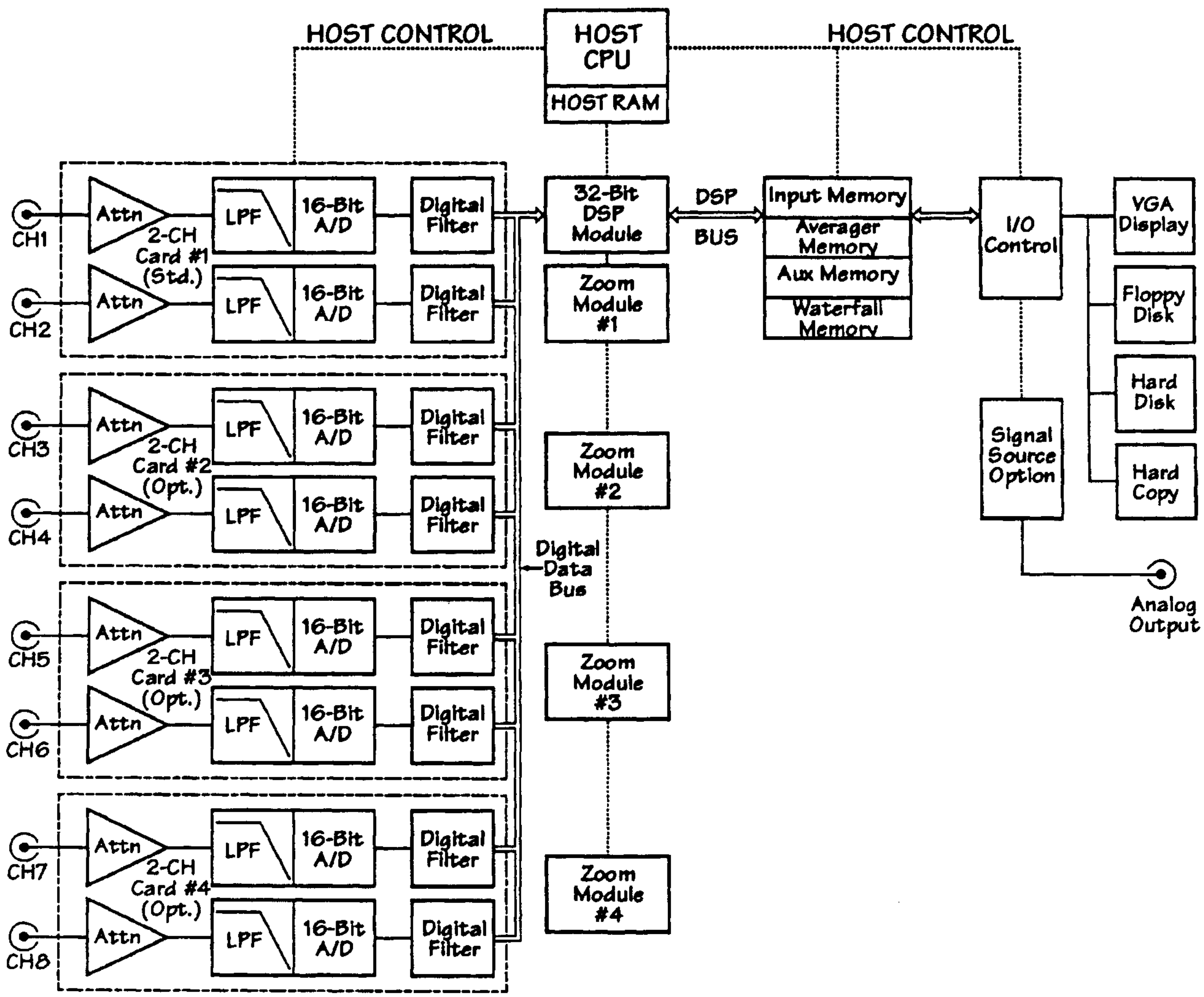
If you're not sure, start with Hanning, check Rectangular, and then possibly Flat Top if the signal is periodic.

- Response → Always used for impact testing for Transfer Function or Cross Properties data.

With the introduction of frequency and amplitude interpolation of spectrum peaks when using the Hanning window, Hanning can now offer the frequency resolution advantages of Hanning plus the amplitude accuracy of Flat Top.

4–2 FUNCTIONAL BLOCK DIAGRAM

SPS390 Basic Block Diagram



Notes

5–1 CLEANING

5–1.1 Chassis

CAUTION

Always turn off the unit power and disconnect the power cord prior to cleaning any part of the chassis or components

Use a clean, damp, lint-free cloth and avoid introducing any foreign matter or liquid inside the unit or inside externally exposed connectors.

Cooling Fan Filter Cleaning

Avoid overheating of the unit by cleaning the fan filter periodically. The frequency of cleaning is dependent on the amount of dust and airborne particles present in the ambient operating environment.

Remove the filter clean dirt by rinsing the filter in water containing a mild detergent. Rinse the filter in clean water and thoroughly dry it before reinstallation on the unit.

Display Screen Cleaning

Wipe the display glass with a clean, damp. Lint-free soft cloth. If a cleaner is necessary, use a commercial grade glass cleaner liquid. Avoid harsh chemical cleaners or any abrasive cleaner. Use a clean, lint-free cloth to dry the class after cleaning.

5–2 FIELD CALIBRATION VERIFICATION TEST

This test is used to verify that the SPS390 measures an input signal within its specified accuracy limits.

The SPS390 does not contain any user-adjustable calibration parts. If the unit does not meet its specified accuracy, contact the factory.

This calibration test supports SPS390 versions 3.1 and above. Select **About SA390** from the **Help** pull-down menu to determine the version of the unit to be tested.

5–2.1 Test Equipment

The following test equipment is required for running the Calibration Verification Tests. If other test equipment is substituted, it must meet or exceed the specifications of the required test equipment.

Description	Model	Manufacturer
SPS390 Calibration Test Disk	2-F3-018	SPS
Synthesizer/Function Generator with 40 Vp-p high Voltage option (002	HP3325B	Hewlett-Packard
Keyboard	AT compatible 101 enhanced	
Assorted 50-ohm coax cables, terminators, BNC tee connectors and shorting plugs		

The signal generator output settings must be set to within ±1.0% of the values specified in the following steps.

5–2.2 Calibration Verification Test Definition

The Calibration Verification Test is used to determine the integrity of the SPS390. The test ensures noise floor, filter flatness, input signal amplitudes, input signal frequency, etc. It also verifies the proper operation of the user controls and internal display.

The SPS390 must be configured with the test configuration files contained on the Signal Processing Systems 3.5" floppy disk (2-F3-018). This disk contains a unique set of files for each hardware configuration of 2, 4, 6, and 8 channels units. Upon initial execution of the test, the operator will be prompted to indicate the number of channels, and then the appropriate test configuration files will be loaded onto the internal hard drive of the SPS390. The test configuration files are listed in the figure 5-1.

CAUTION

Any existing configuration files with the same name as those listed in figure 5-1 will be destroyed. Protect any such files as necessary.

Actual measurement results may be stored for archival purposes. This is not required but is a feature that would facilitate verification of field measurements should data validity questions arise. It is recommended that pre-measurement calibration verification data files be stored on properly formatted 3.5" removable-floppy disks and not on the internal hard drive.

If there is a requirement for an NIST or MIL SPEC calibration, contact Signal Processing Systems' customer service department.

Figure 5-1. Test Configuration files

2 Channels 17 Files	4 Channels 18 Files	6 Channels 27 Files	8 Channels 28 Files
test001.cfg	**test001.cfg**	**test001.cfg**	**test001.cfg**
test002.cfg	**test002.cfg**	**test002.cfg**	**test002.cfg**
test005.cfg	**test005.cfg**	**test005.cfg**	**test005.cfg**
test011.cfg	**test011.cfg**	**test011.cfg**	**test011.cfg**
test012.cfg	**test012.cfg**	**test012.cfg**	**test012.cfg**
test019.cfg	**test019.cfg**	**test019.cfg**	**test019.cfg**
test021.cfg	**test021.cfg**	**test021.cfg**	**test021.cfg**
test022.cfg	**test022.cfg**	**test022.cfg**	**test022.cfg**
test024.cfg	**test024.cfg**	**test024.cfg**	**test024.cfg**
test025.cfg	**test025.cfg**	**test025.cfg**	**test025.cfg**
test035.cfg	**test035.cfg**	**test035.cfg**	**test035.cfg**
test203.cfg	**test035a.cfg**	**test035a.cfg**	**test035a.cfg**
test203a.cfg	**test203.cfg**	**test101.cfg**	**test101.cfg**
test204.cfg	**test203a.cfg**	**test102.cfg**	**test102.cfg**
test204a.cfg	**test204.cfg**	**test105.cfg**	**test105.cfg**
test205.cfg	**test204a.cfg**	**test111.cfg**	**test111.cfg**
test206.cfg	**test205.cfg**	**test112.cfg**	**test112.cfg**
	test206.cfg	**test119.cfg**	**test119.cfg**
		test121.cfg	**test121.cfg**
		test125.cfg	**test125.cfg**
		test135.cfg	**test135.cfg**
		test203.cfg	**test135a.cfg**
		test203a.cfg	**test203.cfg**
		test204.cfg	**test203a.cfg**
		test204a.cfg	**test204.cfg**
		test205.cfg	**test204a.cfg**
		test206.cfg	**test205.cfg**
			test206.cfg

5–2.3 Calibration Setup

Perform the following setup steps prior to running the verification test.

Step	Action
1.	Make sure that the SPS390 to be tested is not powered up.
2.	Attach the keyboard to the KEYBOARD port.
3.	Turn the SPS390 power on and wait for the unit to boot and the SPS390 program screen to be displayed.
4.	Close the SPS390 program and select the PROGRAM MANAGER.
5.	Exit the Windows environment and return to the DOS prompt.
6.	Insert the SPS390 Calibration Test Disk into the floppy drive and change to the A drive by entering "A:[Enter]" at the DOS prompt.
7.	Enter "**CAL390¿**" at the DOS prompt.
8.	When prompted, enter the number of installed channels of the SPS390 and the appropriate configuration files will be loaded.
9.	After the files are loaded remove the SPS390 Calibration Test Disk from the floppy drive. Cycle the SPS390 power off and then on to reboot the unit.

5–2.4 Testing Channels 1 through 4

To test channels 1 through 4, perform the following:

Step	Action
1.	Attach the synthesizer to the Channel 1 and Channel 2 input BNC connectors. If the SPS390 has more than 2 channels installed, also attach the synthesizer to the Channel 3 and Channel 4 input BNC connectors. Enable the 40 Vp-p option on the synthesizer and do not terminate the line in 50 ohms. If the 40 Vp-p option is not enabled, all measurements will be off by 6 dB.
2.	Load test setup **test001.cfg** as follows: a. Select the File menu on the top bar of the display. b. Select the Open Configuration item from the File menu. c. Select the test001.cfg from the Open Configuration menu and then select OK. d. Set the synthesizer for a 1 volt rms sine wave at 2000 Hz. e. Verify that the display shows a signal at 2000 Hz and between -0.40 dBV and +0.30 dBV on all displayed channels.
3.	Load test setup **test002.cfg**. Change the synthesizer to 10 kHz. Verify that the display shows a signal at 10 kHz and between –0.40 dBV and +0.30 dBV on all displayed channels.
4.	Load test setup **test005.cfg**. change the synthesizer to 2 kHz. Verify that the display shows a signal at 2 kHz and between –0.40 dBV and +0.30 dBV on all displayed channels.

Step	Action
5.	Load test setup **test011.cfg**. Start the averager by selecting the START function key (F2) on the SPS390. Verify that all tonals other than the 1 kHz primary tonal are at a maximum level of -70 dB on all displayed channels.
6.	Load test setup **test012.cfg**. Start the averager by selecting the START function key (F2) on the SPS390. Verify that all tonals other than the 1 kHz primary tonal are at a maximum level of -70 dB on all displayed channels.
7.	Load test setup **test019.cfg**. Set the synthesizer for 1.25 Vrms at 1 kHz. a. Verify that all displayed channels show an overload indication (indicated by the corresponding Channel Overload Indicator going to reverse video). b. Decrease the input level to 1.0 Vrms. c. Verify that all channels are no longer in overload
8.	Load test setup **test021.cfg**. Set the synthesizer for 38 kHz at 1 Vrms. For each of the displayed channels, one at a time, attach the synthesizer to one of the channels and shorting plugs to the other one or three channel(s). Verify that all shorted channels have no tonal at 38 kHz that exceeds -70 dB.
9.	Load test setup **test022.cfg**. Set the synthesizer for 95 kHz at 1 Vrms and attach it to the Channel 1 input BNC connector. Attach a shorting plug to the Channel 2 input BNC connector. a. Verify that any tonal appearing at 95 kHz on Channel 2 is at a maximum signal level of -60 dB. b. Attach the synthesizer to the Channel 2 input BNC connector and attach the shorting plug to the Channel 1 input BNC connector. c. Verify that any tonal appearing at 95 kHz on Channel 1 is at a maximum signal level of -60 dB.
10.	Load test setup **test024.cfg**. Set the synthesizer for 250 Hz at 0.60 Vrms. Remove all shorting plugs. Attach the synthesizer to the Channel 1 input BNC connector. a. Verify that the display is updating (indicated by slight flicker and/or movement of the displayed trace). b. Set the synthesizer to 0.55 Vrms. c. Verify that the display is no longer updating. d. Set the synthesizer to 1 Vrms. e. Verify that the display is updating. f. In the ACQUISITION menu, change the trigger source to external. g. Verify that the "Data Not Available" message is displayed. h. Connect a parallel connection from the Channel 1 input BNC connector to the EXT TRIG input. i. Verify that the display is updating. j. Remove the signal from the EXT TRIG input. k. Verify that the display is no longer updating.

Step	Action
11.	Load test setup **test025.cfg**. Set the synthesizer for 1 kHz at 20 mVrms. Attach the synthesizer to the inputs of all displayed channels. a. Verify that all channels show a 1 kHz tonal between 19.0 mVrms and 21.0 mVrms. b. Use the Channel Full Scale Controls to change the input full Scale for all displayed channels to each of the following levels: 50 mVrms, 100 mVrms, 200 mVrms, 500 mVrms, 1 Vrms, 2 Vrms, 5 Vrms, 10 Vrms and 20 Vrms. c. Verify, at each level, that the 1 kHz tonal appears at a level between 19.0 mVrms and 21.0 mVrms.
12.	Load test setup **test035.cfg**. Attach the synthesizer to Channels 1 and 2 input BNC connectors. Set the synthesizer for 0-100 kHz swept frequency with a sweep time of 60 seconds and an amplitude of 1 Vrms. On the synthesizer start the sweep function. On the SPS390, start the averager by selecting the START function key (F2) and allow it to complete its average (60 seconds). a. At 10 kHz and 30 kHz (±2 kHz in each case): b. Verify that the Magnitude transfer function is between 0.97 and 1.03, the phase transfer function is between -1 and +1 degrees and the magnitude of each channel is between -0.9 dBV and +0.1 dBV. c. At 50 kHz, 70 kHz, and 90 kHz (±2 kHz in each case): d. Verify that the Magnitude transfer function is between 0.94 and 1.06, the phase transfer function is between -3 and +3 degrees and the magnitude of each channel is between -1.0 dBV and +0.2 dBV.
13.	If the SPS390 has only 2 channels installed, skip this step; otherwise: Load test setup **test035a.cfg**. Attach the synthesizer to Channels 3 and 4. On the synthesizer start the sweep function. On the SPS390, start the averager by selecting the START function key (F2) and allow it to complete its average (60 seconds). a. At 10 kHz and 30 kHz (±2 kHz in each case): Verify that the Magnitude transfer function is between 0.97 and 1.03, the phase transfer function is between -1 and +1 degrees and the magnitude of each channel is between -0.9 dBV and +0.1 dBV. b. At 50 kHz, 70 kHz, and 90 kHz (±2 kHz in each case): Verify that the Magnitude transfer function is between 0.94 and 1.06, the phase transfer function is between -3 and +3 degrees and the magnitude of each channel is between -1.0 dBV and +0.2 dBV.
14.	If the SPS390 has more than 4 channels then continue with the testing of channels 5 through 8; otherwise proceed to the section 5–2.6, Test System Feature

5–2.5 Testing Channels 5 through 8

To test channels 5 through 8, perform the following:

Step	Action
1.	If the SPS390 has no more than 4 channels installed, then skip this section. Otherwise, attach the synthesizer to the Channel 5 and Channel 6 input BNC connectors. If the SPS390 has more than 6 channels installed, also attach the synthesizer to the Channel 7 and Channel 8 input BNC connectors.
2.	Load test setup **test101.cfg**. Set the synthesizer for a 1 volt rms sine wave at 2000 Hz. Verify that the display shows a signal at 2000 Hz and between -0.40 dBV and +0.30 dBV on all displayed channels.
3.	Load test setup **test102.cfg**. Change the synthesizer to 10 kHz. Verify that the display shows a signal at 10 kHz and between -0.40 dBV and +0.30 dBV on all displayed channels.
4.	Load test setup **test105.cfg**. Change the synthesizer to 2 kHz. Verify that the display shows a signal at 2 kHz and between -0.40 dBV and +0.30 dBV on all displayed channels.
5.	Load tést setup **test111.cfg**. Change the synthesizer to 1 kHz. Start the averager by selecting the START function key (F2) on the SPS390. Verify that all tonals other than the 1 kHz primary tonal are at a maximum level of -70 dB on all displayed channels.
6.	Load test setup **test112.cfg**. Start the averager by selecting the START function key (F2) on the SPS390. Verify that all tonals other than the 1 kHz primary tonal are at a maximum level of -70 dB on all displayed channels.
7.	Load test setup **test119.cfg**. Set the synthesizer for 1.25 Vrms at 1 kHz. a. Verify that all displayed channels show an overload indication (indicated by the corresponding Channel Overload Indicator going to reverse video). Decrease the input level to 1.0 Vrms. Verify that all channels are no longer in overload.
8.	Load test setup **test121.cfg**. Set the synthesizer for 38 kHz at 1 Vrms. For each of the displayed channels, one at a time, attach the synthesizer to one of the channels and shorting plugs to the other one or three channel(s). Verify that all channels which are shorted have no tonal at 38 kHz which exceeds -70 dB.
9.	Load test setup **test125.cfg**. Set the synthesizer for 1 kHz at 20 mVrms. Attach the synthesizer to the inputs of all displayed channels. a. Verify that all channels show a 1 kHz tonal between 19.0 mVrms and 21.0 mVrms. Use the Channel Full Scale Controls to change the input Full Scale for all displayed channels to each of the following levels: 50 mVrms, 100

Step	Action
	mVrms, 200 mVrms, 500 mVrms, 1 Vrms, 2 Vrms, 5 Vrms, 10 Vrms and 20 Vrms. b. Verify at each level that the 1 kHz tonal appears at a level between 19.0 mVrms and 21.0 mVrms.
10.	Load test setup **test135.cfg**. Attach the synthesizer to Channels 5 and 6 input BNC connectors. Set the synthesizer for 0-100 kHz swept frequency with a sweep time of 60 seconds and an amplitude of 1 Vrms. On the synthesizer start the sweep function. On the SPS390, start the averager by selecting the START function key (F2) and allow it to complete its average (60 seconds). a. At 10 kHz and 30 kHz (±2 kHz in each case): Verify that the Magnitude transfer function is between 0.97 and 1.03, the phase transfer function is between -1 and +1 degrees and the magnitude of each channel is between -0.9 dBV and +0.1 dBV. b. At 50 kHz, 70 kHz, and 90 kHz (±2 kHz in each case): Verify that the Magnitude transfer function is between 0.95 and 1.06, the phase transfer function is between -3 and +3 degrees and the magnitude of each channel is between -1.0 dBV and +0.2 dBV.
11.	If the SPS390 has only 6 channels installed skip this step, otherwise: Load test setup **test135a.cfg**. Attach the synthesizer to Channels 7 and 8 input BNC connectors. On the synthesizer start the sweep function. On the SPS390, start the averager by selecting the START function key (F2) and allow it to complete its average (60 seconds). a. At 10 kHz and 30 kHz (±2 kHz in each case): Verify that the Magnitude transfer function is between 0.94 and 1.06, the phase transfer function is between -1 and +1 degrees and the magnitude of each channel is between -0.9 dBV and +0.1 dBV. b. At 50 kHz, 70 kHz, and 90 kHz (±2 kHz in each case): Verify that the Magnitude transfer function is between 0.94 and 1.06, the phase transfer function is between -3 and +3 degrees and the magnitude of each channel is between -1.0 dBV and +0.2 dBV.

5–2.6 Testing System Features

To exercise the test system features, perform the following steps as applicable.

Step	Action
1.	Perform this step only if the SPS390 has the SSG option. Load test setup **test203.cfg**. Disconnect the synthesizer and connect the SSG output to Channel 2. In the SETUP menu, select the Noise Gen Option. When the Noise Generator Setup menu is displayed, select OK. Start the noise function by selecting the NOISE START function key (F7) on the SPS390. Verify, using the Spectrum display, that the overall value (OA) is between 0.95 Vrms and 1.10 Vrms.
2.	Perform this step only if the SPS390 has the SSG option. Load test setup **test203a.cfg**. In the SETUP menu, select the Noise Gen Option. On the Noise Generator Setup menu, select OK. Start the noise function by selecting the NOISE START function key (F7) on the SPS390. Verify, using the Spectrum display, that the overall value (OA) is between 0.95 Vrms and 1.10 Vrms.
3.	Perform this step only if the SPS390 has the SSG option. Load test setup **test204.cfg**. In the SETUP menu, select the Sine Gen Option. On the Sine Generator Setup window, select OK. In the Spectrum display select the S function. In the AXES SETUP window ensure that the scale for the Y Axis Max is set to 1 and then select OK. Start the sine function by selecting the SINE START function key (F7) on the SPS390. a. Verify that there is no overload for Channel 2. b. Verify, using the Spectrum display, that only one tonal appears and that it has a minimum of 900 millivolts across the entire sweep. c. Verify that the time required for the tonal to make one complete sweep is between 39 and 41 seconds.
4.	Perform this step only if the SPS390 has the SSG option. Load test setup **test204a.cfg**. In the SETUP menu, select the Sine Gen Option. On the Sine Generator Setup window, select OK. Start the sine function by selecting the SINE START function key (F7) on the SPS390. a. Verify that there is no overload for Channel 2. b. Verify, using the Spectrum display, that only one tonal appears and that it has a minimum of 850 millivolts across the entire sweep. c. Verify that the time required for the tonal to make one complete sweep is between 78 and 82 seconds.
5.	Load test setup **test205.cfg**. For all available channels attach one 50 ohm terminator per channel pair and connect Channel 1 to Channel 2, Channel 3 to Channel 4, Channel 5 to Channel 6, and Channel 7 to Channel 8. Verify that all displayed DC amplitudes are between 175 millivolts and 225 millivolts.

Step	Action
6.	Load test setup **test206.cfg**. Verify that all displayed DC amplitudes are between 175 millivolts and 225 millivolts.

This concludes the Field Calibration Verification Test.

6– EXTERNAL INTERFACE GUIDE

6–1 SPS390 INTERNAL FILE FORMAT

Each of the data files of the SPS390 have the following basic format, a section for the configuration parameter that determine the data characteristics and a section containing the raw data.

SPS390 Internal Configuration Parameters	**0** **Data Index**
SPS390 Data	

Since the 3.3B release of the SPS390 software, the location of the block of data has been fixed to a certain value according to the file's type. The size of the configuration header is fixed by padding to the end.

File Type	File Offset
Average (AVG)	5120
Extended recorded (XRC)	5120
Trace (TRC) Version 3.4 and Before	558
Trace (TRC) Version 3.5 and After	676
Waterfall (WFL)	5120

6–1.1 SPS390 Interrupt and Address Usage:

The SPS390 uses the following hardware interrupts:

Description	**IRQ**
Timer	0
Keyboard	1
Chained to 2nd int controller	2
COM2 (Mouse/Trackball)	3
COM1	4
Front Panel Keypad (I/O board)	5
Floppy Disk Controller	6
LPT1	7
Real Time Clock	8
Redirected from IRQ2	9
Channel Overload (DSP board)	10
DSP32C (DSP board)	11
DSP32C (I/O board)	12
Math Coprocessor	13
Hard Disk	14
Unassigned	15

The SPS390 uses the following hardware addresses:

Description	Address
DSP32C (I/O board)	100 - 11E
Analog board	200 - 202
I/O board registers	204 - 206
DSP board registers	208 - 20A
IEEE/GPIB controller	2E0 - 2E7
COM 2 - Mouse/Trackball	2F8 - 2FF
DSP32C (DSP board)	300 - 326
Front Panel Keypad (I/O board)	378 - 37F
LPT1 (can be disabled)	3BD - 3BF
COM 3 (selected by jumper)	3E8 - 3EF
COM 1 (selected by jumper)	3F8 - 3FF

The following table describes the configuration header for the AVG, XRC, WFL file formats:

Field Name	Type	Offset	Length	Description
Comment	String	0	80	Description what type of SPS390 file
Timestamp	String	80	30	Date and Time of file creation
File Signature	Long	110	4	See *File Signature Section*
Acquisition PB	Struct	114	400	See *Configuration Header Description*
Analysis PB	Struct	514	132	See *Configuration Header Description*
Display Positions	Struct	646	72	The X/Y positions for all display windows
Display PB	Struct	718	3816	SPS390 Display information for each window. (X/Y min/max and units)
SMS PB*	Struct	4534	224	SMS Data Structure
SSG PB*	Struct	4758	52	Signal Source Generator Data Structure
Padding	-	-	-	Padding to standardized header size

* Optional data structures which are not always present.

The following table describes the configuration header for the TRC file format:

Field Name	Type	Offset	Length	Description
Comment	String	0	80	Description what type of SPS390 file
Timestamp	String	80	30	Date and Time of file creation
File Signature	Long	110	4	See File Signature Section
Display PB	Struct	114	444	SPS390 Display information for trace (X/Y min/max and units)
Trace Data in Volts RMS	Float	558	804 1604 3204 6404	200 Lines (201 data points) 400 Lines (401 data points) 800 Lines (801 data points) 1600 Lines (1601 data points)

6–1.1 SPS390 Data Section

6–1.1.1 Average File Type: (*.avg)

The average data section is stored in internal ATT DSP32C floating point format in Vrms2 units (see Decoding the Average (AVG)).

The data is organized into sequence blocks of data each active channel. Note, only the active channels set in the Analysis Dialog are saved.

For example, an average file may contain four blocks of data for each active channel, 2, 5, 6, 7.

Block of Channel 2 Data
Block of Channel 5 Data
Block of Channel 6 Data
Block of Channel 7 Data

The format of each block is determined by the operating mode of the SPS390:

The **Sync Spectrum** mode stores the time block component of the acquired data. The size of this block is determined by the time block size set in the Analysis Dialog.

The **Spectrum, Sync Spectrum** and **Octave** mode only store the AVG spectrum (FFT) component of the acquired data. The size of this block is determined by the lines of resolution set in the Analysis Dialog.

The **Cross Spectrum** mode stores the AVG spectrum, and the CO and QUAD sections. The size of each sections is determined by the lines of resolution set in the ANALYSIS dialog. These sections are ordered into following manner:

AVG spectrum (FFT)
CO section for 1st reference
QUAD section for 1st reference
CO section for 2nd reference (if MIMO is activated)
QUAD section for 2nd reference (if MIMO is activated)

Example of Cross Spectrum Averager Data Section:

Channels 1 and 2 active with Channel 1 as the reference.

Block #	Description
1	Average FFT Spectrum (Auto Spectrum) for Channel 1
2	Average FFT Spectrum (Auto Spectrum) for Channel 2
3	Cross Spectrum Real (CO) of Channel 1 with Channel 1 as a reference
4	Cross Spectrum Real (CO) of Channel 2 with Channel 1 as a reference
5	Cross Spectrum Imaginary (QUAD) of Channel 1 with Channel 1 as a reference
6	Cross Spectrum Imaginary (QUAD) of Channel 2 with Channel 1 as a reference

Blocks 3 and 5 are useless for analysis purposes; however, they are used to pad to provide a simple indexing scheme into the data for every channel combination. Therefore, an eight channel averager file would contain 24 blocks in the same order (Auto Spectrum, Cross Spectrum Real, and Cross Spectrum Imaginary).

6–1.1.2 Extended Recorder File Type: (*.xrc)

The extended recorder file is divided into two section as defined above a configuration header and a data block. The data section is stored in a 16 bit ATT32C integer format, which can be decoded into the following steps:

Step	Action
1.	Open XRC data file.
2.	Read the configuration header to get full scale voltage, number of channels, and number of samples.

Additional information see Configuration Header and Variable Offsets.

3.	Move file pointer to data section (byte 5120).
4.	Read in 16-bit integer samples.
5.	Scale data according to the following equation: $$\text{scaled value} = \frac{\text{16 bit sample} * \text{full scale voltage}}{16180000}$$
6.	Close XRC data file.

Additional detail information see Decoding the XRC data file

6–1.1.3 Trace File Type: (*.trc)

The trace data is stored according to the trace type. The MODE of the SPS390 determined which display types that are available for the user to display and store to disk. The binary data stored in the trace files is in Vrms units.

Format for the SPS390 trace file with the following display types:

If the display type is any magnitude displays such as **Magnitude Spectrum** or **Transfer Function Magnitude**, the data consists of the Cross Spectrum magnitude information for the number of bins of resolutions specified in the ANALYSIS dialog.

If the display type is any phase displays such as **Phase** or **Transfer Function Phase**, the data consists of the Cross Spectrum phase information for the number of bins of resolutions specified in the ANALYSIS dialog.

If the display type is **Orbit** or **Nyquist**, the data consists of the 2 sets of values with each set having the number of bins of resolutions specified in the ANALYSIS dialog. For **Orbit**, the imaginary data set comes before the real data set. For Nyquist, the Y values come before the X values.

If the display type is an **Octave**, the data consists of the octave magnitude information for the number of octaves specified in the ANALYSIS dialog plus a extra floating point value for the Overall Magnitude that directly follows the trace data.

6–1.1.4 Waterfall File Type: (*.wfl)

The waterfall data file consists of a contiguous block of data from each active channel starting with the lowest channel and ending with the highest channel such as one to eight.

Each channel block is a sequence of waterfall records arranged according to their order of acquisition. However, if waterfall stop method is set to CONTINUOUS, the acquired waterfall records will wrap-around. Therefore, the first acquired record does not always correspond to Record #1.

The user can change the type of data allowed into the waterfall by going to the **Waterfall** dialog box and selecting a waterfall type. The size of each record is determined by number of lines of resolution set in the **Analysis** dialog plus 5 additional floating point values. These additional values are used to store the RPM, time-tag and overall RMS values as seen in the following table:

Record	Byte Offset	Data
0 to n	0 to (n*4)	Vrms data
n + 1	(n + 1) * 4	Overall RMS for record
n + 2	(n + 2) * 4	Time (divide floating point value by 100 to get into seconds)
n + 3	(n + 3) * 4	RPM (only present if tach activated)
n + 4	(n + 4) * 4	Not Used
n + 5	(n + 5) * 4	Not Used

6–1.2 SPS390 File Signature

Each SPS390 data file contains a file signature which determines the type of data storage and the version of SPS390 that supports the file.

Each version of the SPS390 operating software has a distinct base signature and each file type has a distinct type signature as seen in the following tables:

Base Version Signatures:

Version	Base Signature	AVG	CFG	WFL	TRC	XRC
1.0	0x39000101	N/A	0x39001101	N/A	0x39002101	0x39002101
1.5	0x39002259	N/A	0x39003259	N/A	0x39004259	0x39005259
2.0	0x3900517D	N/A	0x3900617D	N/A	0x3900717D	0x3900817D
3.0	0x39001578	N/A	0x39002578	0x39005578	0x39003578	0x39004578
3.1	0x3900C92E	N/A	0x3900D92E	0x3901092E	0x3900E92E	0x3900F92E
3.2	0x39003E58	N/A	0x39004E58	0x39007E58	0x39005E58	0x39006E58
3.3	0x39005968	0x3900A968	0x39006968	0x39009968	0x39007968	0x39008968
3.4	0x3900512B	0X3900A12B	0x3900612B	0x3900912B	0x3900712B	0x3900812B
3.5	0x390010E1	0x390060E1	0x390020E1	0x390050E1	0x390030E1	0x390040E1
3.6	0x390068CB	0x3900B8CB	0x390078CB	0x3900A8CB	0x390088CB	0x390098CB
3.6B	0x3900DE93	0x39012E93	0x3900EE93	0x39011E93	0x3900FE93	0x39010E93

Base File Type Signatures:

AVG	CFG	WFL	TRC	XRC
0x5000	0x1000	0x4000	0x2000	0x3000

After Version 1.5, a file signature is created by adding the Base Version Signature with the Base Type Signature.

Example: Version 3.6 Trace File = 0x390068CB + 0x2000 = 0x390088CB

6–1.3 Converting from DSP32C Floating Point to IEEE Floating Point

The ATT DSP32C processor uses an optimized floating point format. In order to use the SPS390 data stored in the internal format, bit-swapping needs to be performed to get the floating point data into IEEE format, which is the standard for x86 processors.

The description of the bit formats use the following letters to denote how each bit is used.

s = sign bit

f = mantissa bits

e = exponent bits

DSP32 Floating Point Bit Format

sfffffff ffffffff ffffffff eeeeeeee

N = (-1)s * 1.f * 2(e-127)

IEEE Floating Point Bit Format

seeeeeee efffffff ffffffff ffffffff

N = [(-2)s * 0.f] * 2(e-128)

The DSP ATT32C stores negative numbers in Two's Complement. To interpret negative numbers, first complement the bits and then add 1.

6–1.4 SPS390 Floating Point Format

The data stored internally to the SPS390 uses the 32-bit floating point format that conforms to the Institute of Electrical and Electronics Engineers (IEEE) standard 754 format. However, when the SPS390 data is stored into a file, the data is byte and word swapped. This swapping occurs is a result of how the Intel-based X86 hardware stores data in internal memory. This is not a problem if the data is read by a machine having the same Intel architecture. However, if this data is exported to another hardware system (such as a RISC based machine like the 680x0), the data may have to be unswapped to interpret the data correctly.

A example of how to unswap the floating point data:

This is a section of a hex dump of 32-bit floating point data stored in a MSDOS datafile. The data corresponds to the floating point numbers (0.0, 1.0, 2.0, 3.0, and 4.0).

0.0	1.0	2.0	3.0	4.0
0000 0000	0000 803F	0000 0040	0000 4040	0000 8040

The correct IEEE representation of 1.0 is 3F800000. Therefore, the two 16-bit words need to be swapped and then, the bytes in each word need to be swapped.

0000 803F

(Swap words)

803F 0000

(Swap bytes)

3F80 0000

6–1.5 Decoding the Average(AVG) File

The following C program demonstrates how perform to decode the SPS390 average format and how to convert from the DSP32C floating point format to the IEEE floating point format.

```
// Scientific Atlanta Inc. (c) 1994
// Date: August 29, 1994

//++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Example source code - Compiled using Microsoft C7.0
// An examples on how to decode the SPS390 XRC files.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++

#include <math.h>
#include <io.h>
#include <conio.h>
#include <stdio.h>
#include <fcntl.h>        /* _O_ constant definitions */

#define   BYTE   unsigned char
void
main( int argc, char *argv[] )
{
        int       iFile;            // file descriptor
        int       I;                // loop counter
        float     rVrmsSqr;         // floating number in Vrms^2
        float     rVrms;            // floating number in Vrms
        float     rScaledValue;     // scaled floating number
        float     rFSVolt[8];       // full scale voltage for each channel
        int       iSign;                      // sign of the value ( -1 or + 1 )
        float     rMantissa;                  // mantissa part of the value
        float     rExponent;                  // exponent part of the value
        BYTE      b1, b2, b3, b4; // four byte used to decode value
        long      l3, l4;                     // two temporary long values

        // Open SPS390 Average File using binary read
        if( ( iFile = _open( argv[1], _O_BINARY | _O_RDONLY )) == - 1 )
            exit(0);

        for ( i = 0; i < 1; i++ )
        {
            _lseek( iFile, 114 + 4 + 28 * i, 0 );
            _read( iFile, &rFSVolt[i], sizeof( float ) );
        }

        // Seek to start of data section
        _lseek( iFile, 5120, 0 );

        // Read 100 ensembles
        for ( i = 0; i < 100; i++ )
        {
            _read( iFile, &b1, sizeof( BYTE ) );
            _read( iFile, &b2, sizeof( BYTE ) );
            _read( iFile, &b3, sizeof( BYTE ) );
            _read( iFile, &b4, sizeof( BYTE ) );

        if ( b4 & 0x80 )     // Check sign bit which the MSD in byte 4
        {
                  b4 = b4 & 0x7f;             // clear sign bit
            b2 = ~b2;               // invert mantissa byte
            b3 = ~b3;               // invert mantissa byte
            b4 = ~b4;               // invert mantissa byte
            iSign = -1;             // Set sign to negative
```

```
            }
                else
                {
                b4 = b4 | 0x80;        // Supply "1" for "1.f" by setting MSD
                iSign = 1;             // Set to positive
            }

            // Shift before adding the three mantissa bytes
                l4 = (long) b4 << 16;  // align number shifting over 16 bytes
            l3 = (long) b3 << 8;       // align number shifting over 8 bytes

                rMantissa = (l4 + l3 + b2) / pow (2, 23); // 1.f
                rExponent = pow( 2.0f, b1 - 127 );

                // Number = sign * 1.f * 2 (e - 127 )
                rVrmsSqr = iSign * ( rMantissa * rExponent ) / 2;

                // Take square-root to change the Vrms units
                rVrms = (float) sqrt( (double) rVrmsSqr );

                // Output routine goes here!!!
            }
            _close( iFile );
   exit( 0 );
}
```

6–1.6 Decoding the Extended Recorder (XRC) File

The following C program demonstrates how to decode a SPS390 XRC file.

```
// Scientific Atlanta Inc. (c) 1994      August 29, 1994
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Example source code - Compiled with Microsoft C7.0
// An examples on how to decode the SPS390 XRC files.
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++

#include <math.h>
#include <io.h>
#include <conio.h>
#include <stdio.h>
#include <fcntl.h>          /* _O_ constant definitions */

#define OFFSET_DATA_SECTION  5120
#define OFFSET_NUM_CHAN      516
#define OFFSET_XRC_SIZE      526

void
main( int argc, char *argv[] )
{
            int         iFile;          // file descriptor
            long        i, x;           // loop counters
            float       rScaledValue;   // scaled floating number
            float       rFSVolt[8];     // full scale voltage for each channel
            int         i1;             // 16-bit integer value
            int         iNChans;        // Number of channels
            long        lXTimeSize;     // Size of time samples per channel

            // Open SPS390 Extended Recorder File using binary read
  if( ( iFile = _open( argv[1], _O_BINARY | _O_RDONLY )) == - 1 )
  {
                printf( "Cannot open SPS390 XRC file!\n\n" );
                exit(0);
  }

  // Read the full scale voltage for all the channels
            for ( i = 0; i < 8; i++ )
            {
                _lseek( iFile, 114 + 4 + 28 * i, 0 );
                _read( iFile, &rFSVolt[i], sizeof( float ) );
            }

            // Determine the number of channels
            _lseek( iFile, OFFSET_NUM_CHAN, 0 );
            _read( iFile, &iNChans, sizeof( short ) );

            // Determine the size of XRC
            _lseek( iFile, OFFSET_XRC_SIZE, 0 );
            _read( iFile, &lXTimeSize, sizeof( long ) );

            // Seek to start of data section
            _lseek( iFile, 5120, 0 );

            for ( x = 0; x < iNChans; x++ )
            {
                // Read data points for each channel
                for ( i = 0; i < lXTimeSize; i++ )
                {
                      read( iFile, &i1, sizeof( int ) );

                      // For XRC data, need to scale data by following equation
                      rScaledValue = ( (float) i1 * rFSVolt[x] ) / 16180000.0f;

                      // Put Output Routine Here!!!!
                }
            }
            _close( iFile );
            exit( 0 )
}
```

6–1.7 Configuration Header Description

Each SPS390 has a file header that contains the information about the data stored into that file. The averager (AVG), waterfall (WFL), and extended recorder (XRC) each have a detailed configuration header that stored both the data characteristics and acquisition and analysis settings. The following table contains a list of the useful variables and their corresponding offset into the data file.

Field Name	Offset	Length	C Data Type	Notes
Acquisition Parameters				
Channel 1 Information				
mv/EU	114	4	float	
Full Scale Voltage	118	4	float	
DB Reference	122	4	float	
Reference Voltage for DB	126	4	float	
EU Label	134	8	char	
Channel 2 Information				
mv / EU	142	4	float	
Full Scale Voltage	146	4	float	
DB Reference	150	4	float	
Reference Voltage for DB	154	4	float	
EU Label	162	8	char	
Channel 3 Information				
mv / EU	170	4	float	
Full Scale Voltage	174	4	float	
DB Reference	178	4	float	
Reference Voltage for DB	182	4	float	
EU Label	190	8	char	
Channel 4 Information				
mv / EU	198	4	float	
Full Scale Voltage	202	4	float	
DB Reference	206	4	float	
Reference Voltage for DB	210	4	float	
EU Label	218	8	char	
Channel 5 Information				
mv / EU	226	4	float	
Full Scale Voltage	230	4	float	
DB Reference	234	4	float	
Reference Voltage for DB	238	4	float	
EU Label	246	8	char	
Channel 6 Information				
mv / EU	254	4	float	
Full Scale Voltage	258	4	float	
DB Reference	262	4	float	
Reference Voltage for DB	266	4	float	

Field Name	**Offset**	**Length**	**C Data Type**	**Notes**
Acquisition Parameters				
EU Label	274	8	char	
Channel 7 Information				
mv / EU	282	4	float	
Full Scale Voltage	286	4	float	
DB Reference	290	4	float	
Reference Voltage for DB	294	4	float	
EU Label	302	8	char	
Channel 8 Information				
mv / EU	310	4	float	
Full Scale Voltage	314	4	float	
DB Reference	318	4	float	
Reference Voltage for DB	322	4	float	
EU Label	330	8	char	
Sampling Method	338	2	integer	1 = Internal 2 = SRA 3 = External
Band	340	2	integer	0 = Base Band 1 = Zoom Band 2 = Base Zoom Band
Acquisition Mode	342	2	WORD (unsigned integer)	0 = Continuous 1 = Stop When Full
Tach Present	344	2	BOOL (integer)	
Frequency Span	346	4	float	For Base and Zoom
Center Frequency	350	4	float	For Zoom
Trigger Source	406	2	integer	-1 = External 0 = Free Run 1-8 = channel #
Trigger Threshold	408	2	integer	-99% to 99%
Trigger Slope	410	2	integer	0 = Positive 1 = Negative
Trigger Type	412	2	BOOL (integer)	TRUE = Single FALSE = Repetitive
SRA Full Scale Orders	430	2	WORD (unsigned integer)	Tracking Orders
SRA Trigger RPM	438	4	long	
SRA End RPM	442	4	long	
SRA Pulses per Revolution	434	4	float	
SRA Delta RPM	458	4	float	

Field Name	Offset	Length	C Data Type	Notes
Acquisition Parameters				
Number of Analog Boards Present	484	2	integer	0 to 8
Analyzer Mode	514	2	WORD (unsigned integer)	2 = Spectrum 4 = Sync Spectrum 8 = Cross Properties 16 = Octave 64 = Correlation
Number of Active Channels	516	2	integer	1 to 8
Active Channel List	518	2	integer	bit 1 = Channel 1 bit 2 = Channel 2 bit 3 = Channel 3 bit 4 = Channel 4 bit 5 = Channel 5 bit 6 = Channel 6 bit 7 = Channel 7 bit 8 = Channel 8
Reference Channel List	520	2	integer	Same as Active List
Extended Recorder Size per Channel	526	4	long	Depends on Installed DSP Memory
Size of Time Blocks	530	2	integer	512, 1024, 2048, 4096
Number of Lines Resolution for FFT	532	2	integer	200, 400, 800, 1600
FFT Weighting Window	534	2	integer	0 = Rectangle 1 = Hanning 2 = Harris Flat Top 4 = Response
Overlap	536	2	integer	0 = None 1 = 25% 2 = 50% 3 = 75% 4 = 90% 5 = MAX
Response Tau	538	2	integer	1 to 15
Response Offset	540	2	integer	10% to 80%
Octave Mode	582	2	WORD (unsigned integer)	1 = 1/1 Octave 3 = 1/3 Octave
Octave Weighting	584	2	WORD (unsigned integer)	1 = A 2 = B 3 = C 4 = D 5 = Flat
Octave Number of Bins of Resolution	586	2	WORD (unsigned integer)	

Field Name	Offset	Length	C Data Type	Notes
Acquisition Parameters				
Number of Calculated Octaves	588	2	WORD (unsigned integer)	
Waterfall Active Flag	590	2	BOOL (integer)	TRUE = Active FALSE = Inactive
Waterfall Source	592	2	WORD (unsigned integer)	1 = Real Time 2 = Averager
Number Waterfall Channels	594	2	WORD (unsigned integer)	1 to 8
Waterfall Channel List	596	2	WORD (unsigned integer)	Same as Active List (see above)
Waterfall Reference Channel	598	2	WORD (unsigned integer)	1 to 8
Maximum Waterfall Records	600	2	WORD (unsigned integer)	
Actual Number of Waterfall Records	602	2	WORD (unsigned integer)	
Number of Waterfall Functions	604	2	WORD (unsigned integer)	
Waterfall Function List	606	4	long	
Waterfall Load Control	610	2	WORD (unsigned integer)	1 = Single Block 2 = 75% Overlap 3 = 50% Overlap 4 = 25% Overlap 5 = 10% Overlap 6 = MAX 7 = Single 8 = AVG Recycle 9 = Delta Time 10 = Delta RPM 11 = % Amplitude
Waterfall Memory Load Control	612	2	WORD (unsigned integer)	0 = Continuous 1 = Stop When Full
Waterfall RPM Start	614	4	long	
Waterfall RPM End	618	4	long	
Waterfall RPM Increment	622	4	long	
Waterfall Speed Control	624	2	WORD (unsigned integer)	1 = RPM Up 2 = RPM Down 3 = RPM Up and Down

Field Name	Offset	Length	C Data Type	Notes
Acquisition Parameters				
Waterfall Time Increment	626	4	float	Delta Time
Waterfall % Amplitude	630	2	integer	% Amplitude Change
Averager Mode	634	2	integer	0 = Linear 1 = Exponential 2 = Peak
Averager Stop Mode	636	2	integer	0 = Count 1 = Time 2 = Manual
Averager Target Count	638	2	integer	
Averager Current Count	640	2	integer	
Averager Target Time	642	4	long	in Seconds

6–1.8 SMS Data File Format

The SPS390 stores the SMS data into four different data files, Frequency Response Function (FRF), Coherence (COH), Auto Power Spectrum (APS), and Cross Power Spectrum (CPS).

The following table describes the SMS elements stores into the data file by SPS390:

Description	Variable	Format	Value
General Header	header	headert	See *headert* structure
STAR Revision Code	revcode	int	2831
Header Length	headerlen	int	16
Measurement Header	msmheader	msmheadert	See *msmheadert* structure
Primary Measurement Type	datatype	int	0 = FRF 12 = APS 2 = COH 13 = CPS
# of Complex Data Pairs	noelements	int	Analysis Resolution of 200, 400, 800, or 1600 lines
Measurement ID Label	measid	char[128]	Date and Time Label dd:yy:hh:mm:ss
Analyzer ID Label	analyzerid	char[24]	"SD390"
APS Amplitude Compensation	ispeak	char	3 = Hanning compensation of Channel A 5 = Hanning compensation of Channel B(only used for datatype APS)
X Axis Range Information	xaxisrange	xaxisranget	See *xaxisranget* structure
Starting Frequency	xlow	float	Any valid frequency
Delta Frequency	deltax	float	Any valid frequency

Description	Variable	Format	Value
Ending Frequency	xhigh	float	Any valid frequency
Center Frequency	xoffset	float	Any valid frequency
Zoom Code	zoom	integer	0 = Baseband 1 = Zoom 2 = Octave 3 = 1/3 Octave 4 = 1/6 Octave If the zoom code indicates octave analysis, xlow, deltax, xhigh, and xoffset are not used. The complex data is the data for each octave band.
Reference Measurement Info	channel1	channelt	See *channelt* structure
Response Measurement Info	channel2	channelt	See *channelt* structure
Point and Direction	point	int	Direction information is encoded as follows: 0 = No Direction 1 = X Rectangular 2 = Y Rectangular 3 = Z Rectangular, Cylindrical 4 = R Cylindrical, Spherical 5 = T Cylindrical, Spherical 6 = P Spherical
User Units	units	int	0 = User Units
Real/Imag Parts of the Measurement	msmdata[]	complex	See *complex* structure

6–1.8.1 How to calculate the "point" variable

point = (10 * abs(point_number) + direction) * sign(point_number)

Examples: 33Z = (10 * 33 + 3) = 36,-28R = -(10 * 28 + 4) = 284

For COH, FRF and CPS Channel 1 is the Reference Channel and Channel 2 is the Response Channel.

For APS Channel 1 and Channel2 are the same point.

For COH, FRF and CPS the complex data includes the real component followed by the imaginary component for each data point.

For APS, the imaginary component is 0, and the real component is the amplitude.

SMS Header C Data Structures

The SMS data files are defined by the following definitions and structures. Only the commented variables are used by the SPS390. All others variables are set to a NULL value.

```
#define MAXBSIZE 1601

typedef struct
{
        int   revcode;        /* STAR revision code */
        int   headerlen;      /* header length */
        int   entries;
        int   pline;
        int   sentries;
        int   spline;
        int   tentries;
        int   tpline;
} headert;

typedef struct
{
        int   datatype;       /* primary measurement type */
        int   misctype;
        int   noelements;             /* number or complex data pairs */
        float msmcalval;
        char caltrace[16];
        char measid[128];             /* measurement id label */
        char labletype[120];
        char xlabel[16];
        char ylabel[16];
        char msmdate[16];
        char msmtime[16];
        char analyzerid[24];          /* analyzer id label */
        int   noave;
        int   windowtype;
        char miscwindow[16];
        float wdwvalue;
        float spacing;
        int   surface;
        float minreal;
        float maxreal;
        float minimag;
        float maximag;
        float maxmag;
        int   mmdefined;
        char micpair;
        char ispeak;    /* APS amplitude compensation */
        float excamp;
```

```
} msmheadert:

typedef struct
{
        float xlow:         /* start frequency  */
        float deltax:       /* delta frequency  */
        float xhigh:        /* end frequency    */
        float xoffset:      /* center frequency */
        int   zoom:         /* zoom code        */
        int   code:
} xaxisranget:

typedef struct
{
        int   point.        /* point and direction */
        int   units:        /* user units          */
        char unitslabel[8].
        char xducerid[24]:
        float calfactor:
        char ampid[24]:
        float gain:
        char adcid[24]:
        float range:
        int   couplingtype:
} channelt:

typedef struct
{
        float real:
        float imag:
} complex:      /* complex data type */

typedef struct
{
        headert      header:                  /* general header */
        msmheadert   msmheader:    /* measurement header */
        xaxisranget  xaxisrange:   /* X axis range info  */
        channelt     channel1.     /* Reference measurement info */
        channelt     channel2:     /* Response measurement info  */
        complex      msmdata[MAXBSIZE]: /* real/imag parts of the measurement*/
} MSMT:
```

However, only N (not MAXBSIZE] complex data pairs are written to the file, where N is the resolution of the analysis.

The following tables contain the values for various internal SPS390 configuration items:

Display Types	Value
Compressed Time	0
Time Trace	1
Orbit	2
Magnitude Spectrum	5
Phase Spectrum	6
Real Spectrum	7
Imaginary Spectrum	8
TF Mag Accelerance	11
TF Mag Mobility	12
TF Mag Compliance	13
TF Mag Stiffness	14
TF Mag Impedance	15
TF Mag Dynmass	16
TF Phase	17
TF Phase Accelerance	18
TF Phase Mobilty	19
TF Phase Compliance	20
TF Phase Stiffness	21
TF Phase Impedance	22
TF Phase Dynmass	23
TF Coherence	24
TF Real	25
TF Imaginary	26
TF Nyquist	27
Cross Spectrum Magnitude	28
Cross Spectrum Phase	29
Cross Spectrum Real	30
Cross Spectrum Imaginary	31
Cross Spectrum Nyquist	32
Sync Spec Magnitude	35
Sync Spec Phase	36
Sync Spec Real	37
Sync Spec Imaginary	38
Octave	39
Auto Correlation	40
Cross Correlation	41
Multiple Display	42

Analyzer Mode	Value
Spectrum	0x02
Syn Spectrum	0x04
Cross Properties	0x08
Octave	0x10
Correlation	0x40

Waterfall Type	Value
Single	0x01
Cascade	0x02
Peak Hold	0x04
Profile	0x08
Profile OA	0x10

Math Functions Types	Value
None	0
Add Constant	1
Subtract Constant	2
Multiply Constant	3
Divide Constant	4
Inverse	5
Square	6
Square Root	7
Binary Add	20
Binary Subtract	21
Binary Multiply	22
Binary Divide	23

6.2 DYNAMIC DATA EXCHANGE (DDE)

6–2.1 DDE Description

This section provides how to implement the SPS390's DDE (Dynamic Data Exchange) interface. DDE allows another MS Windows application to exchange data and control the SPS390 analyzer. These DDE links can be established both at an application level such as through Microsoft Excel, or at a programming level using Visual Basic or C++.

To invoke a DDE conversation between the SPS390 and another application, the server name, topic and item must be specified. The server name for the SPS390 is the string "SD390".

An example of SPS390 DDE request

Server:	SD390
Topic:	Display_A
Item:	Cursor_3

In Excel, this DDE link can be established by entering the following into an Excel cell location:

= SD390|Display_A!Cursor_3

In Visual Basic, this DDE link can be created by setting the DDE properties of a static control as shown in the following example:

```
Label1.LinkMode = 0
Label1.LinkTopic = "SD390|DISPLAY_A"      'Server Name and Topic
Label1.LinkItem = "CURSOR_3"              ' Item Name
Label1.LinkMode = 1                       ' Establish Hot Link
```

In C or C++, a DDE conversion must be created using the Windows API and the server, topic and item. Refer to the ***DDEML*** Windows API library that provides the high level functionality to initialize a conservation, create string handles, and create the DDE callback procedure.

6–2.2 DISPLAY Topic

Having the ability to get the cursor values from a SPS390 display is a powerful and quick way to get the information for the analyzer.

Topic Syntax:	**DISPLAY_A | DISPLAY_B | DISPLAY_C | DISPLAY_D | DISPLAY_E | DISPLAY_F | DISPLAY_G | DISPLAY_H | DISPLAY_I**
Item Syntax:	**CURSOR_1 | CURSOR_2 | CURSOR_3**
Description:	Requests a cursor value from a specified display. The number after "Cursor" corresponds to the position of the readout value beneath the SPS390 display. Generally, CURSOR_1 will be the X value, CURSOR_2 will be the Y value and the CURSOR_3 will be either OA or the Z value.
Example:	Get the Y cursor value of display B. Server = **SPS390**, Topic = **DISPLAY_B**, Item = **CURSOR_2**

6–2.3 COMMAND topic

The **COMMAND** topic provides the ability to control the SPS390 analyzer from another applications. It provides the capability of simulating the SPS390 button presses and performs various operating tasks such as loading in a configuration file, starting the averager, and saving a trace to a file.

In Visual Basic, the **COMMAND** topic is executed via the ***LinkExecute*** command. To initiate a DDE link, set the LinkTopic with SD390 server and topic and call the LinkExecute with the item strings documented below.

In C, this topic is performed via the ***DdeClientTransaction*** using the ***XTYP_EXECUTE*** transaction type.

Topic Syntax: **COMMAND**

These are the group of DDE items that are used to control the SPS390 operations.

Item Syntax:	**CTRL ACQU [UPDATE I HOLD]** **CTRL AVGR [START I STOP]** **CTRL WATR [START I STOP I RESUME I SINGLE]** **CTRL SSG [START I STOP]** **CTRL SRA** **CTRL ARNG**
Description:	Set a specified SPS390 control or process to a desired setting.
Example:	Inform SPS390 to start to acquire data into the input buffer. Server = SD390, Topic = **COMMAND**, Item = **"CTRL ACQU UPDATE"**

Item Syntax:	**SAVE** <type> <filename>
Arguments:	type = { trace | **CNFG IAVGR I WATR I EXTR }** filename = a string that contains a valid DOS filename with extension.
Description:	Save a specified area to SPS390 disk storage under a filename.
Example:	Server = **SD390**, Topic = **COMMAND**, Item = "**SAVE CNFG testfile.cfg**"

Syntax:	**LOAD** <type> <filename>
Arguments:	type = { trace **I CNFG I AVGR I WATR I EXTR }** filename = a string that contains a valid DOS filename with extension.
Description:	Load specified area from SPS390 disk storage under a filename.
Example:	Load configuration from a file. Server = **SD390**, Topic = **COMMAND** Item = "**LOAD CNFG testfile.cfg**"

Syntax:	**WAIT** <time>
Description:	Waits the specified duration in seconds while SPS390 continues to process information.
Example:	Allow SPS390 to process data from the specified duration of 3 seconds. Server = **SD390,** Topic = **COMMAND**, Item = "**WAIT 3**"

Syntax:	**INFO** <information string>
Description:	Displays a message string inside an information box on the SPS390.
Example:	Display a test message to the SPS390 Server = **SD390**, Topic = **COMMAND**, **Item** = "**INFO Starting Testing...**"

6–2.4 STATUS topic

This is a group of DDE items that provide the user will the status of the SPS390.

Topic Syntax:	**STATUS**
Item Syntax:	**RPM**
Description:	Returns the tach value in **RPM** .
Example:	Retrieve the current **RPM** from the tach. Server = **SD390**, Topic = **STATUS**, Item = **RPM**
Item Syntax:	**INFO**
Description:	Returns the analyzer's current status and configuration in a 16 bit unsigned integer.

BIT #	Field Description	Values
0	Error Bit	1 - Yes, 0 - No
1 2 3	Mode Bits	000 - Cross Properties 001 - Octave 010 - Spectrum 011 - Sync Spectrum 100 - Correlation
4	Acquisition Processing	1 - Yes, 0 - No
5	NOT USED	
6	Averager Processing	1 - Yes, 0 - No
7	Averager Count Reached	1 - Yes, 0 - No
8	SRA Reprocessing Active	1 - Yes, 0 - No
9	Waterfall Processing	1 - Yes, 0 - No
10	NOT USED	
11	Playback Active	1 - Yes, 0 - No
12	Tach Active	1 - Yes, 0 - No
13	Channel Overload	1 - Yes, 0 - No
14	WFL Mode Active	1 - Yes, 0 - No
15	Auto-Range in Process	1 - Yes, 0 - No

Example: Retrieve the current status information.
Server = **SD390**, Topic = **STATUS**, Item = **INFO**

6–2.5 CHANNEL topic

This is a specialized group of DDE topics that provide machine diagnostic data information about analyzed spectrum data.

Topic Syntax: **CHANNEL_1 | CHANNEL_2 | CHANNEL_3 | CHANNEL_4**
CHANNEL_5 | CHANNEL_6 | CHANNEL_7 | CHANNEL_8
Item Syntax: **1XMAG**
Description: Get fundamental (1X) vibration in unscaled Vrms.

Use must have a complex waterfall and averager running.

Example: Get the 1X vibration from channel two.
Server = **SD390**, Topic = **CHANNEL**_2, Item = **1XMAG**

Item Syntax: **1XPHASE**
Description: Get fundamental (1X) phase in degrees .

Use must have a complex waterfall and averager running.

Example: Get the 1X phase from channel two.
Server = **SD390,** Topic = **CHANNEL_2,** Item = **1XPHASE**

Item Syntax: **MISALIGN**
Description: Get the misalignment (2X) vibration of a selected channel.

Use must have a complex waterfall and averager running.

Example: Get the misalignment vibration of channel two.
Server = **SD390**, Topic = **CHANNEL_2**, Item = **MISALIGN**

Item Syntax:	**OA**
Description:	Get overall amplitude of a selected channel.

Use must have a complex waterfall and averager running.

Example:	Get the overall amplitude of channel two. Server = **SD390,** Topic = **CHANNEL_2,** Item = **OA**