Programmer's Manual

PROGRAMMER'S MANUAL ACE9600

For use with ACE9600
firmware versions later than 1.0

RF Test Equipment for Wireless Communications

Confidentially
This manual contains proprietary information of dBmCorp, Inc. It is provided under confidential custody for the sole purpose of specification, installation, maintenance and operation of dBm test system instruments and equipment and may not be used or disclosed to any person for any other purpose whatsoever. No portion of this manual or the information it contains may be reproduced, used, or disclosed to individuals not having a need to know consistent with its intended purpose, without the written permission of dBmCorp, Inc.

Disclaimer
dBmCorp, Inc. makes no representation or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantability for any particular purpose.

dBmCorp, Inc. reserves the right to revise this publication and to make changes from time to time in the content thereof without obligation of dBmCorp, Inc. to notify any person of such revision or change.

MAN_ACE_Prog Rev 1.04 06/17 Printed in U.S.A

Contents

Contacting dBm

We encourage you to contact us if you want more information or have any questions or concerns about this or any other dBm product or manual. Use any of the following methods:

Mail	dBm 32A Spruce Street Oakland, NJ 07436
Telephone	(201) 677-0008
Fax	(201) 677-9444
E-mail—Technical Support	Info@dbmcorp.com
www	http://www.dbmcorp.com

REMOTE OPERATION OVERVIEW

The ACE9600 device can be controlled remotely using its LAN interface. The instrument can be connected to any IEEE-802 network. It uses TCP/IP, and achieves transfer rates up to approximately 10/100 MBPS.

Programming control of the ACE9600 can be implemented by using the ACE9600 LAN client application called ***AceClient*** provided by dBm or by sending text commands directly via a valid socket connection to the control port.

In addition, parameter files may be downloaded or uploaded by the ACE9600 via the data port, or files may be deleted from the ACE9600 memory through text commands to the control port.

A complete description of both the ***AceClient*** and the ACE9600 text command syntax are given in this manual.

Text Command Overview

The text command syntax used to remotely control the ACE9600 was designed for ease of use, readability and tracking over socket monitoring programs. The commands begin with an action word, followed by an optional group name, channel number, parameter name and parameter value.

For more information about the ACE9600 text command syntax, see **USING REMOTE TEXT COMMANDS.**

Client Application Overview

AceClient is a Windows® application that interfaces with the ACE9600. The application provides graphical user interface (GUI) to:

- Display or modify parameters on the device
- Download, upload or delete dynamic data files from the device
- Store or recall settings that have been saved on the device

The application saves a list of recently used IP address between executions of the program using the Windows® INI file system, and has the ability to save or load complete configuration settings under a custom file name.

For more information about the features provided by ***AceClient***, see **CLIENT APPLICATION FEATURES.**

For detailed instructions on using the client application, see **USING THE CLIENT APPLICATION.**

USING REMOTE TEXT COMMANDS

This section details the text command syntax used to remotely control the ACE9600 over a LAN interface. The command descriptions are grouped by similar function. For a summary of the text command syntax in table form, see **Appendix B: Text Message** Format**.**

In general, all text commands begin with an action word,

e.g., "**`Get`**", "**`Abort`**", "**`Start`**"...

followed by an optional group name,

e.g., "**`Mode`**", "**`Param`**", "**`Dynamic`**" ...

channel number ,

e.g., "**`Ch1`**","**`Ch2`**" ...

parameter name ,

e.g., "**`DelayInterval`**", "**`Delay`**"...

or parameter setting,

e.g., "**`Enabled`**", "**`On`**", "**`1200.13`**" ...

Commands are case-insensitive, where each command word is separated by an ASCII space character.

Please note that all text commands and text responses are sent or received through the control port connection (5555). All binary data used for file download, file upload or capture data is through the data port connection (5556).

Connect and Open Commands

The commands described in this section are used for connecting, disconnecting, opening, and closing the control or data sockets.

Connecting to the ACE

Syntax:

Hello

Function:

After a socket connection has been established with the instrument's control port, this is first command that should be sent to the control port to verify that it is receiving and responding to text messages.

Returns:

Nak: <error message>

or

Ack: Hello

Where:

<error message> Descriptive text describing cause of error

Disconnecting from the ACE

Syntax:

Goodbye

Function:

This is the last command that should be sent to the control port to inform the instrument that remote communications are ending. The instrument will close the control port socket after transmitting the final return message and begin listening for a new socket connection request.

Returns:

Nak: <error message>

or

Ack: Goodbye

Where:

<error message> Descriptive text describing cause of error

Opening the Data Port

Syntax:

Open DataPort

Function:

Instructs the ACE to open the data port and await connection by the client.

Returns:

Nak: <error message>

or

Ack: DataPort Opened

Where:

<error message> Descriptive text describing cause of error

Example:

Sent command -

Open DataPort

Received command -

Ack: DataPort Opened

Notes:

- Send this command to the control port before uploading a file, downloading file or uploading capture data. After the command has been acknowledged, open a socket to the data port and begin file or capture transfers.

Closing the Data Port

Syntax:

Close DataPort

Function:

Instructs the ACE to close the data port.

Returns:

Nak: <error message>

or

Ack: DataPort Closed

Where:

<error message> Descriptive text describing cause of error

Example:

Sent command -

Close DataPort

Received command -

Ack: DataPort Closed

Notes:

- Send this command after finishing all file and capture transfers, as well as before disconnecting from the control port.
- If further file or capture transfers are needed, reopen the data port by sending the "Open DataPort" command and opening a socket to the data port.

Mode Commands

The commands described in this section are related to obtaining or changing the mode of the instrument.

Getting the Mode

Syntax:

Get Mode

Function:

Returns the current Mode of the instrument.

Returns:

Nak: <error message>

or

Ack: Mode <setting>

Where:

<error message> Descriptive text describing cause of error

<setting> **Static/Dynamic**

Example:

Sent command -

Get Mode

Received command -

Ack: Mode Static

Changing the Mode

Syntax:

Set Mode <setting >

Function:

Changes the Mode of the instrument to either the Static or Dynamic mode.

Returns:

Nak: <error message>

or

Ack: Mode <setting >

Where:

<error message> Descriptive text describing cause of error

<setting> **Static/Dynamic**

Example:

Sent command -

Set Mode Dynamic

Received command -

Ack: Mode Dynamic

Storing and Recalling Settings

The commands described in this section are related to storing and recalling settings in the instrument's on-board memory.

Storing a Current Setting on the Instrument

Syntax:

Store Settings <index>

Function:

Saves the current instrument settings in the on-board memory.

Returns:

Nak: <error message>

or

Ack: Settings <index>

Where:

<error message> Descriptive text describing cause of error

<index> **0** to **8**

Example:

Sent command -

Store Settings 3

Received command -

Ack: Settings 3

Notes:

- Setting 0 is the same as the "Preset" button on the instruments front panel.

Recalling a Setting on the Instrument

Syntax:

Recall Settings <index>

Function:

Recalls a saved setting from the on-board memory to replace the current instrument settings.

Returns:

Nak: <error message>

or

Ack: Settings <index>

Where:

<error message> Descriptive text describing cause of error

<index> **0** to **9**

Example:

Sent command -

Recall Settings 4

Received command -

Ack: Settings 4

Notes:

- Setting 0 is the same as the "Preset" button on the instruments front panel.
- Setting 9 is the default factory settings and cannot be changed.

Options Settings Commands

The commands described in this section are related to obtaining the factory settings of the instrument, enabling the Multipath Function, or changing the Capture Point.

Obtaining Version Related Settings

Syntax:

Get Version <type >

Function:

Returns the Software Version, Serial Number, or Model Number of the instrument.

Returns:

Nak: <error message>

or

Ack: Version <type > <string>

Where:

<error message>	Descriptive text describing cause of error
<type>	**SwVersion/SerialNumber/ModelNumber**
<string>	ASCII string

Example:

Sent command -

```
Get Version ModelNumber
```

Received command -

```
Ack: Version ModelNumber ACE9600-4-1125
```

Notes:

- These options cannot be changed without factory maintenance by dBm-Corp.

Obtaining Factory Option Related Settings

Syntax:

Get Options <channel> <type>

Function:

Returns the installation state for Fading license, AWGN license, channel board installation, or RF Slave configuration

Returns:

Nak: <error message>

or

Ack: Options <channel> <type> <status>

Where:

<error message>	Descriptive text describing cause of error
<channel>	**Ch1/Ch2/Ch3/Ch4**
<type>	**FadingLicense/AwgnLicense/ Installed/RfSlave**
<status>	**Enabled/Disabled**

Example:

Sent command -

```
Get Options Ch1 AwgnLicense
```

Received command -

```
Ack: Options Ch1 AwgnLicense Enabled
```

Notes:

- These options cannot be changed without factory maintenance by dBm-Corp.

Getting Non-Factory Option Related Settings

Syntax:

Get Options <channel> < type >

Function:

Returns the current setting for the Multipath and Capture Point options.

Returns:

Nak: <error message>

or

Ack: Options <channel> < type > < setting >

Where:

<error message>	Descriptive text describing cause of error
<type>	**Multipath/CapturePoint**
<channel>	**Ch1/Ch2/Ch3/Ch4**
<setting>	One of these valid choices:
	Multipath **Enabled/Disabled**
	CapturePoint **Before/After**

Example:

Sent command -

Get Options Ch1 Multipath

Received command -

Ack: Options Ch1 Multipath Enabled

Changing Setting for Non-Factory Option Related Settings

Syntax:

Set Options <channel> < type > <setting>

Function:

Changes the current setting for the Multipath and Capture Point Function.

Returns:

Nak: <error message>

or

Ack: Options <channel> < type > < setting >

Where:

<error message>	Descriptive text describing cause of error
<type>	**Multipath/CapturePoint**
<channel>	**Ch1/Ch2/Ch3/Ch4**
<setting>	One of these valid choices:
	Multipath **Enabled/Disabled**
	CapturePoint **Before/After**

Example:

Sent command -

Set Options Ch1 Multipath Enable

Received command -

Ack: Options Ch1 Multipath Enabled

Notes:

- The Multipath option allows for the entire multipath function to be disabled or enabled. This has the same effect as turning all of the paths off, but allows for the paths to be returned to their original settings by simply enabled the Multipath option.
- The Capture Point option allows the signal to be captured either before or after the delay is emulated.

Dynamic Mode Related Commands

The commands described in this section are related to parameters and actions that are valid in the Dynamic mode.

Starting a Dynamic Run

Syntax:

Start

Function:

Starts or restarts the dynamic run when in the dynamic mode while paused or ready.

Returns:

Nak: <error message>

or

Ack: Dynamic State <state>

Where:

<error message> Descriptive text describing cause of error

<state> **Running/Done/Armed**

Example:

Sent command -

Start

Received command -

Ack: Dynamic State Armed

Resetting a Dynamic Run

Syntax:

Reset

Function:

Resets the dynamic run when in the dynamic mode while running, paused or done.

Returns:

Nak: <error message>

or

Ack: Dynamic State Ready

Where:

<error message> Descriptive text describing cause of error

Example:

Sent command -

Reset

Received command -

Ack: Dynamic State Ready

Pausing a Dynamic Run

Syntax:

Pause

Function:

Pauses the dynamic run when in the dynamic mode while running.

Returns:

Nak: <error message>

or

Dynamic State <state>

Where:

<error message> Descriptive text describing cause of error

<state> **Paused/Done/Armed**

Example:

Sent command -

Pause

Received command -

Ack: Dynamic State Paused

Incrementing a Dynamic Run

Syntax:

StepFwd <setting>

Function:

Increments the dynamic run by the number of msec specified by <setting> when in the dynamic mode while paused or ready.

Returns:

Nak: <error message>

or

Ack: Dynamic ElapsedTime <setting>

Where:

<error message> Descriptive text describing cause of error

<setting> Time in msec

Example:

Sent command -

Step Fwd 10

Received command -

Ack: Dynamic ElapsedTime 50

Decrementing a Dynamic Run

Syntax:

StepBack <setting>

Function:

Decrements the dynamic run by the number of msec specified by <setting> when in the dynamic mode while paused or ready.

Returns:

Nak: <error message>

or

Ack: Dynamic ElapsedTime <setting>

Where:

<error message>	Descriptive text describing cause of error
<setting>	Time in msec

Example:

Sent command -

StepBack 10

Received command -

Ack: Dynamic ElapsedTime 40

Getting Dynamic Run Parameters

Syntax:

Get Dynamic <parameter>

Function:

Returns the Trigger, Time Reference, Loop, Update, Multi Chassis Sync, Delay Interval, RF Interval, Multipath Interval, Elapsed Time, Start Time, or State while in the Dynamic mode.

Returns:

Nak: <error message>

or

Ack: Dynamic < parameter > <setting>

Where:

<error message>	Descriptive text describing cause of error
<parameter>	<setting>
Trigger	**Internal/External**
Loop	**Single/Continuous**
MultiChassisSync	**StandAlone/Master/Slave**

DelayInterval	**1/2/5/10/20/50/100/200/ 500/1000** (in msec)
RfInterval	**100/200/500/1000** (in msec)
MpathInterval	**100/200/500/1000** (in msec)
Update	**Internal/External**
ElapsedTime	integer (in msec)
StartTime	integer (in msec)
State	**Static/Ready/Setup/Init/Running/ Paused/ Done/Armed**

Example:

Sent command -

Get Dynamic Loop

Received command -

Ack: Dynamic Loop Continuous

Changing Dynamic Run Parameters

Syntax:

Set Dynamic <parameter> <setting>

Function:

Changes the Trigger, Time Reference, Loop Type, Multi Chassis Sync, Update, Delay Interval, RF Interval, Multipath Interval, Elapsed Time, Start Time, or State while in the Dynamic mode.

Returns:

Nak: <error message>

or

Ack: Dynamic < parameter > <setting>

Where:

<error message>	Descriptive text describing cause of error
<parameter>	<setting>
Trigger	**Internal/External**
Loop	**Single/Continuous**
MultiChassisSync	**StandAlone/Master/Slave**
DelayInterval	**1/2/5/10/20/50/100/200/ 500/1000** (in msec)
RfInterval	**100/200/500/1000** (in msec)
MpathInterval	**100/200/500/1000** (in msec)
Update	**Internal/External**
StartTime	integer (in msec)

Example:

Sent command -

Set Dynamic DelayInterval 50

Received command -

Ack: Dynamic DelayInterval 50

Notes:

- The parameter **DelayInterval** sets the interval for Delay, Frequency Offset, Attenuation, Phase Offset, and Noise Density type files.

Beginning Dynamic Setup

Syntax:

Begin Dynamic Setup

Function:

Before changing dynamic run parameters or dynamic data file selections, the instrument must be placed in the Setup state. Once the state is acknowledged as **Setup**, changes in the dynamic run parameters or dynamic data file selections may begin. See **Changing Selected File for Dynamic** for more information about selecting files for dynamic data parameters.

Returns:

Nak: <error message>

or

Ack: Dynamic State Setup

Where:

<error message> Descriptive text describing cause of error

Beginning Dynamic Init

Syntax:

Begin Dynamic Init

Function:

After changing dynamic run parameters or selecting files for dynamic data parameters, the instrument must be placed in the Init state to initialize the dynamic run to the starting point. Once the initialization procedure finishes and the state is acknowledged as **Ready**, the command to Start may be sent. See **Changing Selected File for Dynamic** for more information about selecting files for dynamic data parameters.

Returns:

Nak: <error message>

or

Ack: Dynamic State Ready

Where:

<error message> Descriptive text describing cause of error

File Related Commands

The commands described in this section are related to setting or obtaining information about files that have been downloaded previously to the instrument. See **Getting the Delay Slew Boundary**

Syntax:

`Get DelaySlewBoundary`

Function:

Returns the current Delay Slew Boundary of the instrument.

Returns:

`Nak:` <error message>

or

`Ack: DelaySlewBoundary` <setting>

Where:

<error message>	Descriptive text describing cause of error
<setting>	value (in msec)

Example:

Sent command -

`Get DelaySlewBoundary`

Received command -

`Ack: DelaySlewBoundary 100`

Changing the Delay Slew Boundary

Syntax:

`Set DelaySlewBoundary` <setting >

Function:

Changes the Delay Slew Boundary of the instrument.

Returns:

`Nak:` <error message>

or

`Ack: DelaySlewBoundary` <setting >

Where:

<error message>	Descriptive text describing cause of error
<setting>	value (in msec)

Example:

Sent command -

`Set DelaySlewBoundary 100`

Received command -

`Ack: DelaySlewBoundary 100`

Calculating Noise Density

Syntax:

`Calculate`

Function:

Calculates the Noise Density parameter for each channel based on the Noise Ratio parameter and the current Signal Power.

Returns:

`Nak:` <error message>

or

`Ack: Calculated`

Where:

<error message> Descriptive text describing cause of error

Example:

Sent command -

`Calculate`

Received command -

`Ack: Calculated`

Notes:

- See **Notes:**
- **The Delay value** may also includes the status **`Slewing`**, which is separated from the value by an ASCII space character.
- Getting Settings for Noise Parameters about commands to retrieve the calculated status of each channel.
- The Noise Density parameter will only be calculated if the Noise Type for that channel is set to "EbNo".

Capturing Input Signals

Syntax:

`Capture` <channel>

Function:

Samples the incoming signal to a channel (or all channels) for retrieval later.

Returns:

`Nak:` <error message>

or

`Ack: Captured` <channel>

Where:

<error message> Descriptive text describing cause of error

<channel> **`Ch1/Ch2/Ch3/Ch4/All`**

Notes:

- The channel's incoming signal is captured when the command is received. See **Uploading Captured Data from a Channel** about commands to retrieve the captured data.

- The Capture Point option allows the signal to be captured either before or after the delay is emulated. See **Changing Setting for Non-Factory Option Related Settings** about commands to set the Capture Point.

Example:

Sent command -

`Capture Ch1`

Received command -

`Ack: Captured Ch1`

Downloading a Binary Dynamic Data File or **Uploading a Binary Dynamic Data File** for more information about commands to upload or download the dynamic files.

Getting the File Count

Syntax:

`Get File Count`

Function:

Returns the number of dynamic data files that have previously been downloaded to the instrument.

Returns:

`Nak:` <error message>

or

`Ack: File Count` <number>

Where:

<error message> Descriptive text describing cause of error

<number> number of downloaded files

Example:

Sent command -

`Get File Count`

Received command -

`Ack: File Count 5`

Deleting a Single File

Syntax:

`Delete File` <filename>

Function:

Removes a file from the file directory on the instrument.

Returns:

`Nak:` <error message>

or

`Ack: File` <filename> **`Deleted`**

Where:

<error message> Descriptive text describing cause of error

<filename> Filename with *.ace extension

Example:

Sent command -

Delete File Dly1.ace

Received command -

Ack: File Dly1.ace Deleted

Deleting All Files

Syntax:

Delete File All

Function:

Removes all files from the file directory on the instrument.

Returns:

Nak: <error message>

or

Ack: File All Deleted

Where:

<error message> Descriptive text describing cause of error

Example:

Getting Downloaded File Information

Syntax:

Get File Info <filename>

Function:

Retrieves file information for the file with the given name.

Returns:

Nak: <error message>

or

Ack: File Info <filename> <type> <length> <continuous> <timestamp>

Where:

Returns space delimited file information, where:

<error message>	Descriptive text describing cause of error
<filename>	Filename
<type>	**Delay/FreqOffset/Atten/ PhaseOffset/RfIn/NoiseNo/Mpf**
<length>	number of samples in the file
<continuous>	**CONTINUOUS/DISCONTINUOUS**
<timestamp>	YYYY-MM-DD_HH:MM:SS

Example:

Sent command -

Get File Info Dly1.ace

Received command -

Ack: File Info Dly1.ace Delay 10 Continuous 2015-09-23_07:05:32

Notes:

- **RfIn** type files may be used for either RF Input or RF Output file selections.

Getting Downloaded File Size in Bytes

Syntax:

Get File SizeInBytes <filename>

Function:

Retrieves file size in bytes for the file with the given name.

Returns:

Nak: <error message>

or

Ack: File SizeInBytes <filename> <size>

Where:

Returns space delimited file information, where:

<error message>	Descriptive text describing cause of error
<filename>	Filename
<size>	File size in bytes

Example:

Sent command -

Get File SizeInBytes Dly1.ace

Received command -

Ack: File SizeInBytes Dly1.ace 1024

Getting Selected File for Dynamic Data Parameters

Syntax:

Get File <channel> <parameter>

Function:

Returns the filename that is currently selected for Dynamic mode Delay, Frequency Offset, Attenuation, Phase Offset, RF Input, RF Output, Noise Density, or Multipath Fading.

Returns:

Nak: <error message>

or

Ack: File <channel> <parameter> <filename>

Where:

<error message>	Descriptive text describing cause of error
<channel>	**CH1/CH2/CH3/CH4**
<parameter>	**Delay/FreqOffset/Atten/ PhaseOffset/RfIn/RfOut/NoiseNo/Mpf**

<filename> Filename or **None**

Example:

Sent command -

Get File Ch1 Delay

Received command -

Ack: File Ch1 Delay None

Changing Selected File for Dynamic Data

Syntax:

Set File <channel> <parameter> <filename>

Function:

Selects the filename to be used for Dynamic mode Delay, Frequency Offset, Attenuation, Phase Offset, RF Input, RF Output, Noise Density, or Multipath Fading.

Returns:

Nak: <error message>

or

Ack: File <channel> <parameter> <filename>

Where:

<error message> Descriptive text describing cause of error

<channel> **CH1/CH2/CH3/CH4**

<parameter> **Delay/FreqOffset/Atten/ PhaseOffset/RfIn/RfOut/NoiseNo/Mpf**

<filename> Filename or **None**

Example:

Sent command -

Set File Ch1 Dly1.ace

Received command -

Ack: File Ch1 Delay Dly1.ace

Static Mode Related Commands

The commands described in this section are related to parameters and actions that are valid in the Static Mode.

Getting Settings for Link Parameters

Syntax:

Get Param <channel> <parameter>

Function:

Returns the current setting for Delay, Frequency Offset, Phase Offset, Attenuation, RF Input, RF Output and Signal Power.

Returns:

Nak: <error message>
or
Ack: Param <channel> <parameter> <setting>

Where:

<error message>	Descriptive text describing cause of error		
<channel>	**CH1/CH2/CH3/CH4**		
<parameter>	**Delay/FreqOffset/PhaseOffset/ Atten/RfIn/RfOut/SigPower**		
<setting>	Value with default units:		
	Delay	(in msec)	**Slewing** (only if slewing)
	FreqOffset	(in kHz)	
	PhaseOffset	(in deg)	
	Atten	(in dB)	
	RfIn	(in MHz)	
	RfOut	(in MHz)	
	SigPower	(in dBm)	**InRange/Overflow**

Example:

Sent command -

```
Get Param Ch1 Delay
```

Received command -

```
Ack: Param Ch1 1000.3400000
```

Notes:

- The Delay value may also includes the status **Slewing**, which is separated from the value by an ASCII space character.
- The Signal Power value also includes the status of the measurement as either **InRange** or **Overflow**, which is separated from the value by an ASCII space character.

Changing Settings for Link Parameters

Syntax:

Set Param <channel> <parameter> <setting>

Function:

Changes the current setting for Delay, Frequency Offset, Phase Offset, Attenuation, RF Input, and RF Output.

Returns:

Nak: <error message>
or
Ack: Param <channel> <parameter> <setting>

Where:

<error message>	Descriptive text describing cause of error
<channel>	**CH1/CH2/CH3/CH4**
<parameter>	**Delay/FreqOffset/PhaseOffset/**

	Atten/RfIn/RfOut		
<setting>	Value with default units:		
	Delay	(in msec)	**Slewing** (only if slewing)
	FreqOffset	(in kHz)	
	PhaseOffset	(in deg)	
	Atten	(in dB)	
	RfIn / RfOut	(in MHz)	

Example:

Sent command -

Set Param Ch3 Atten 10.5

Received command -

Ack: Param Ch3 Atten 10.5

Notes:

- The Delay value may also includes the status **Slewing**, which is separated from the value by an ASCII space character.

Getting Settings for Noise Parameters

Syntax:

Get Param <channel> <parameter>

Function:

Returns the current setting for Noise Density, Noise Bitrate, Noise Ratio, Noise Mode, Noise Enable Status, Calculated Status, and Range Status.

Returns:

Nak: <error message>

or

Ack: Param <channel> <parameter> <setting>

Where:

<error message>	Descriptive text describing cause of error	
<channel>	**CH1/CH2/CH3/CH4**	
<parameter>	**NoiseNo/NoiseBr/NoiseEbNo/ NoiseMode/NoiseEnable/ CalcStatus/RangeStatus**	
<setting>	Value with default units or one of valid choices:	
	NoiseNo	(in dBm/Hz)
	NoiseBr	(in MHz)
	NoiseEbNo	(in dB)
	NoiseMode	**No/EbNo**
	NoiseEnable	**Off/On**
	CalcStatus	**Calculated/NotCalculated**
	RangeStatus	**InRange/OutOfRange**

Example:

Sent command -

Get Param Ch2 Noise

Received command -

Ack: Param Ch2 Noise -90.50

Changing Settings for Noise Parameters

Syntax:

Set Param <channel> <parameter> <setting>

Function:

Changes the current setting for Noise Density, Noise Bitrate, Noise Ratio, Noise Mode and Noise Enable Status.

Returns:

Nak: <error message>

or

Ack: Param <channel> <parameter> <setting>

Where:

<error message>	Descriptive text describing cause of error
<channel>	**CH1/CH2/CH3/CH4**
<parameter>	**NoiseNo/NoiseBr/NoiseEbNo/ NoiseMode/NoiseEnable**
<setting>	Value with default units or one of valid choices:
	NoiseNo (in dBm/Hz)
	NoiseBr (in MHz)
	NoiseEbNo (in dB)
	NoiseMode **No/EbNo**
	NoiseEnable **Off/On**

Example:

Sent command -

Set Param Ch2 NoiseEnable On

Received command -

Ack: Param Ch2 NoiseEnable On

Getting Settings for Fading Parameters

Syntax:

Get Param <channel> **Mpf** <path> <parameter>

Function:

Returns the current setting for Multipath Delay, Doppler, Loss, Path Type, Correlation Path, Correlation Value, K Factor, AoA, Ratio and Std Dev.

Returns:

Nak: <error message>

or

Ack: Param <channel> **Mpf** <path> <parameter> <setting>

Where:

<error message>	Descriptive text describing cause of error
<channel>	**CH1/CH2/CH3/CH4**
<path>	**PATH1/PATH2/PATH3/PATH4/ PATH5/PATH6**
<parameter>	**Delay/Doppler/Loss/Type/CorrPath/ CorrVal/Kfactor/Aoa/Ratio/StdDev**
<setting>	Value with default units or one of valid choices:

Delay	(in usec)
Doppler	(in Hz)
Loss	(in dB)
Type	**Off/Cw/Ray/Rice**
CorrPath	**PATH1/PATH2/PATH3/PATH4/ PATH5/PATH6**
CorrVal	(in %)
Kfactor	(in dB)
Aoa	(in deg)
Ratio	(in Hz)
StdDev	(in dB)

Example:

Sent command -

Get Param Ch2 Mpf Path1 Aoa

Received command -

Ack: Param Ch2 Mpf Path1 Delay 45

Changing Settings for Fading Parameters

Syntax:

Set Param <channel> **Mpf** <path> <parameter> <setting>

Function:

Changes the current setting for Multipath Delay, Doppler, Loss, Path Type, Correlation Path, Correlation Value, K Factor, AoA, Ratio and Std Dev.

Returns:

Nak: <error message>

or

Ack: Param <channel> **Mpf** <path > <parameter> <setting>

Where:

<error message>	Descriptive text describing cause of error
<channel>	**CH1/CH2/CH3/CH4**
<path>	**PATH1/PATH2/PATH3/PATH4/ PATH5/PATH6**
<parameter>	**Delay/Doppler/Loss/Type/CorrPath/**

CorrVal/Kfactor/Aoa/Ratio/StdDev

<setting> Value with default units or one of valid choices:

Delay	(in usec)
Doppler	(in Hz)
Loss	(in dB)
Type	**Off/Cw/Ray/Rice**
CorrPath	**PATH1/PATH2/PATH3/PATH4/ PATH5/PATH6**
CorrVal	(in %)
Kfactor	(in dB)
Aoa	(in deg)
Ratio	(in Hz)
StdDev	(in dB)

Example:

Sent command -

Set Param Ch2 Mpf Path1 CorrPath Path6

Received command -

Ack: Param Ch2 Mpf Path1 CorrPath Path6

Getting the Delay Slew Boundary

Syntax:

Get DelaySlewBoundary

Function:

Returns the current Delay Slew Boundary of the instrument.

Returns:

Nak: <error message>

or

Ack: DelaySlewBoundary <setting>

Where:

<error message>	Descriptive text describing cause of error
<setting>	value (in msec)

Example:

Sent command -

Get DelaySlewBoundary

Received command -

Ack: DelaySlewBoundary 100

Changing the Delay Slew Boundary

Syntax:

Set DelaySlewBoundary <setting >

Function:

Changes the Delay Slew Boundary of the instrument.

Returns:

Nak: <error message>

or

Ack: DelaySlewBoundary <setting >

Where:

<error message> Descriptive text describing cause of error

<setting> value (in msec)

Example:

Sent command -

```
Set DelaySlewBoundary 100
```

Received command -

```
Ack: DelaySlewBoundary 100
```

Calculating Noise Density

Syntax:

Calculate

Function:

Calculates the Noise Density parameter for each channel based on the Noise Ratio parameter and the current Signal Power.

Returns:

Nak: <error message>

or

Ack: Calculated

Where:

<error message> Descriptive text describing cause of error

Example:

Sent command -

```
Calculate
```

Received command -

```
Ack: Calculated
```

Notes:

- See **Notes:**
- **The Delay value** may also includes the status **Slewing**, which is separated from the value by an ASCII space character.
- Getting Settings for Noise Parameters about commands to retrieve the calculated status of each channel.
- The Noise Density parameter will only be calculated if the Noise Type for that channel is set to "EbNo".

Capturing Input Signals

Syntax:

Capture <channel>

Function:

Samples the incoming signal to a channel (or all channels) for retrieval later.

Returns:

Nak: <error message>

or

Ack: Captured <channel>

Where:

<error message>	Descriptive text describing cause of error
<channel>	**Ch1/Ch2/Ch3/Ch4/All**

Notes:

- The channel's incoming signal is captured when the command is received. See **Uploading Captured Data from a Channel** about commands to retrieve the captured data.
- The Capture Point option allows the signal to be captured either before or after the delay is emulated. See **Changing Setting for Non-Factory Option Related Settings** about commands to set the Capture Point.

Example:

Sent command -

```
Capture Ch1
```

Received command -

```
Ack: Captured Ch1
```

Downloading a Binary Dynamic Data File

The commands described in this section are related to downloading a previously converted binary dynamic data file to the instrument. Text commands are sent through the control port and binary data is sent through the data port. For an example of downloading a binary data file, see **File Download Protocol Example**.

Beginning the File Download

Syntax:

Begin File Transfer <filename> <size in bytes>

Function:

Informs the instrument that downloading will begin.

Returns:

Nak: <error message>

or

Ack: File Transfer Beginning

Where:

<error message>	Descriptive text describing cause of error
<filename>	Filename

<size in bytes> Size of the entire binary data file in bytes

Notes:

- The **`Begin File Transfer`** command is sent through the control port of the instrument, and the acknowledgement or Nak comes through the control port.

Sending Binary Download Data

Syntax:

[binary data]

Function:

Binary data segments are sent through the data port of the instrument.

Returns:

`Nak:` <error message>

or

`Ack: File Transfer Next`

Where:

<error message> Descriptive text describing cause of error

Notes:

- The binary data is sent through the data port, and the acknowledgment or Nak comes through the control port.
- The binary data size for each segment is limited to 8192 bytes.

Ending the File Download

Syntax:

`End File Transfer` <filename>

Function:

Informs the instrument that downloading has ended normally.

Returns:

`Nak:` <error message>

or

`Ack: File Transfer Ending`

Where:

<error message> Descriptive text describing cause of error

<filename> Filename

Notes:

- The **`End File Transfer`** command is sent through the control port of the instrument, and the acknowledgement or Nak comes through the control port.
- Once the acknowledgment is received, another file transfer may begin.

Aborting the File Download

Syntax:

Abort File Transfer <filename>

Function:

Informs the instrument that downloading is terminating before completion.

Returns:

Nak: <error message>

or

Ack: File Transfer Aborted

Where:

<error message>	Descriptive text describing cause of error
<filename>	Filename

Notes:

- The **Abort File Transfer** command is sent through the control port of the instrument, and the acknowledgement or Nak comes through the control port.

File Download Protocol Example

The protocol (Figure 1 and Figure 31) for downloading a dynamic data file begins with the client application sending the **Open DataPort** command to the instrument's Control Port. Upon receiving acknowledgment back from the Control Port, the client application opens a socket and connects to the Data Port of the instrument. Upon successfully connecting to the Data Port, the client application then sends the **Begin File Transfer** command to the instrument's Control Port. Upon receiving an acknowledgment back from the Control Port, the client application sends segments of binary to the Data Port. Each data segment that is sent on the Data Port will generate a text response on the Control Port. After the last segment has been sent to the Data Port, the client application sends the **End File Transfer** command to the Control Port. Once the acknowledgment is received through the Control Port, another file transfer may begin. After completing all downloads, the client application sends the **Close DataPort** command to the Control Port. Upon receiving acknowledgment back from the Control Port, the client application must close the data port connection.

Uploading a Binary Dynamic Data File

The commands described in this section are related to uploading a binary dynamic data file from the instrument. Text commands are sent to the control port and binary data is sent from the data port.

Beginning the File Upload

Syntax:

Begin File Uptransfer <filename>

Function:

Informs the instrument that uploading will begin.

Returns:

Nak: <error message>
or
Ack: File Uptransfer Beginning

Where:

<error message> Descriptive text describing cause of error
<filename> Filename

Notes:

- The **Begin File Uptransfer** command is sent through the control port of the instrument, and the acknowledgement or Nak comes through the control port.

Receiving Binary Upload Data

Syntax:

Next File Uptransfer

Function:

Binary data segments are sent from the data port of the instrument.

Returns:

Nak: <error message>
or
[binary data]

Where:

<error message> Descriptive text describing cause of error

Notes:

- The binary data is sent from the data port as the acknowledgement or the Nak comes through the control port.
- The binary data size for each segment is limited to 8192 bytes.

Ending the File Upload

Syntax:

End File Uptransfer <filename>

Function:

Informs the instrument that uploading has ended normally.

Returns:

Nak: <error message>
or
Ack: File Uptransfer Ending

Where:

<error message> Descriptive text describing cause of error
<filename> Filename

Notes:

- The **End File Uptransfer** command is sent through the control port of the instrument, and the acknowledgement or Nak comes through the control port.
- Once the acknowledgment is received, another file transfer may begin.
- See **Getting Downloaded File Size in Bytes** about obtaining the total file size (in bytes) that is expected from the upload.

Aborting the File Upload

Syntax:

Abort File Uptransfer <filename>

Function:

Informs the instrument that uploading is terminating before completion.

Returns:

Nak: <error message>

or

Ack: File Uptransfer Aborting

Where:

<error message>	Descriptive text describing cause of error
<filename>	Filename

Notes:

- The **Abort File Uptransfer** command is sent through the control port of the instrument, and the acknowledgement or Nak comes through the control port.

File Upload Protocol Example

The protocol (Figure 2 and Figure 32) for uploading a dynamic data file begins with the client application sending the **Open DataPort** command through the instrument's Control Port. Upon receiving acknowledgment back through the Control Port, the client application opens and connects to the Data Port of the instrument. Upon successfully connecting to the Data Port, the client application then sends the **Begin File Uptransfer** command through the instrument's Control Port. Upon receiving an acknowledgment back through the Control Port, the client application sends the command **Next File Uptransfer** through the Control port which will generate a response of a binary data segment through the instrument's Data Port. After the last segment has been received through the Data Port, the client application sends the **End File Uptransfer** command through the Control Port. Once the acknowledgment is received through the Control Port, another file transfer may begin. After all files have been uploaded, the client application sends the **Close DataPort** command through the Control Port. Upon receiving acknowledgment back through the Control Port, the client application must close the data port connection.

Uploading Captured Data from a Channel

The commands described in this section are related to uploading binary capture data from the instrument. Text commands are sent through the control port and binary data is sent through the data port. Each binary data sample is contained in a 16-bit word for I (In-Phase component) followed by a 16-bit word for Q (Quadrature component). There are 4096 I/Q pairs, for a total data capture size of 16,384 bytes.

Beginning the Capture Upload

Syntax:

`Begin Capture Uptransfer` <channel>

Function:

Informs the instrument that uploading capture data will begin.

Returns:

`Nak:` <error message>

or

`Ack: Capture Uptransfer Beginning`

Where:

<error message> Descriptive text describing cause of error

<channel> **`CH1/CH2/CH3/CH4`**

Notes:

- The **`Begin Capture Uptransfer`** command is sent through the control port of the instrument, and the acknowledgement or Nak comes through the control port.

Receiving Binary Capture Data

Syntax:

`Next Capture Uptransfer`

Function:

Binary data segments are sent through the data port of the instrument.

Returns:

`Nak:` <error message>

or

[binary data]

Where:

<error message> Descriptive text describing cause of error

Notes:

- The binary data is sent through the data port as the acknowledgement or the Nak comes through the control port.
- The binary data size for each segment is limited to 8192 bytes.

Ending the Capture Upload

Syntax:

End Capture Uptransfer <channel>

Function:

Informs the instrument that uploading has ended normally.

Returns:

Nak: <error message>

or

Ack: Capture Uptransfer Ending

Where:

<error message> Descriptive text describing cause of error

<channel> **CH1/CH2/CH3/CH4**

Notes:

- The **End Capture Uptransfer** command is sent through the control port of the instrument, and the acknowledgement or Nak comes through the control port.
- Once the acknowledgment is received, another channel capture transfer may begin.
- The total data capture size expected by the upload is 16,384 bytes.

Aborting the Capture Upload

Syntax:

Abort Capture Uptransfer <channel>

Function:

Informs the instrument that uploading is terminating before completion.

Returns:

Nak: <error message>

or

Ack: Capture Uptransfer Aborting

Where:

<error message> Descriptive text describing cause of error

<channel> **CH1/CH2/CH3/CH4**

Notes:

- The **Abort Capture Uptransfer** command is sent through the control port of the instrument, and the acknowledgement or Nak comes through the control port.

Capture Upload Protocol Example

The protocol (Figure 3) for uploading captured data for a channel begins with the client application sending the **Open DataPort** command to the instrument's Control Port. Upon receiving acknowledgment back from the Control Port, the client application

opens and connects to the Data Port of the instrument. Upon successfully connecting to the Data Port, the client application then sends the **`Begin Capture Uptransfer`** command through the instrument's Control Port. Upon receiving an acknowledgment back through the Control Port, the client application sends the command **`Next Capture Uptransfer`** through the Control port, which will generate a response of a binary data segment through the instrument's Data Port. After the last segment has been received through the Data Port, another channel capture transfer may begin. After uploading all captured data, the client application sends the **`End Capture Uptransfer`** command through the Control Port. Once the acknowledgment is received through the Control Port, the client application sends the **`Close DataPort`** command through the Control Port. Upon receiving acknowledgment back through the Control Port, the client application must close the data port connection.

Uploading Dynamic File Directory List

The commands described in this section are related to uploading the list of dynamic data files that are stored on the instrument. Text commands are sent to and received through the control port.

Beginning the Directory List Upload

Syntax:

`Begin File Directory`

Function:

Informs the instrument that uploading the directory list will begin.

Returns:

`Nak:` <error message>

or

`Ack: File Directory Beginning`

Where:

<error message> Descriptive text describing cause of error

Receiving a Directory Item

Syntax:

`Next File DirItem`

Function:

Dynamic filenames are sent from the control port of the instrument.

Returns:

`Nak:` <error message>

or

`Ack: File DirItem` <filename>

Where:

<error message> Descriptive text describing cause of error

<filename> Filename

Ending the Directory List Upload

Syntax:

`End File Directory`

Function:

Informs the instrument that uploading has ended normally.

Returns:

`Nak:` <error message>

or

`Ack: File Directory Ending`

Where:

<error message> Descriptive text describing cause of error

Uploading Directory Protocol Example

The protocol (Figure 4) for uploading the directory of dynamic filenames begins with the client application sending the **`Begin file Directory`** command through the instrument's Control Port. Upon receiving an acknowledgment back through the Control Port, the client application sends the command **`Next File DirItem`** through the Control port which will generate a response of a filename through the instrument's Control Port. After the last filename has been received through the Control Port, the client application sends the **`End File Directory`** command through the Control Port. See **Getting the File Count** for more information about obtaining the total number of files stored on the instrument.

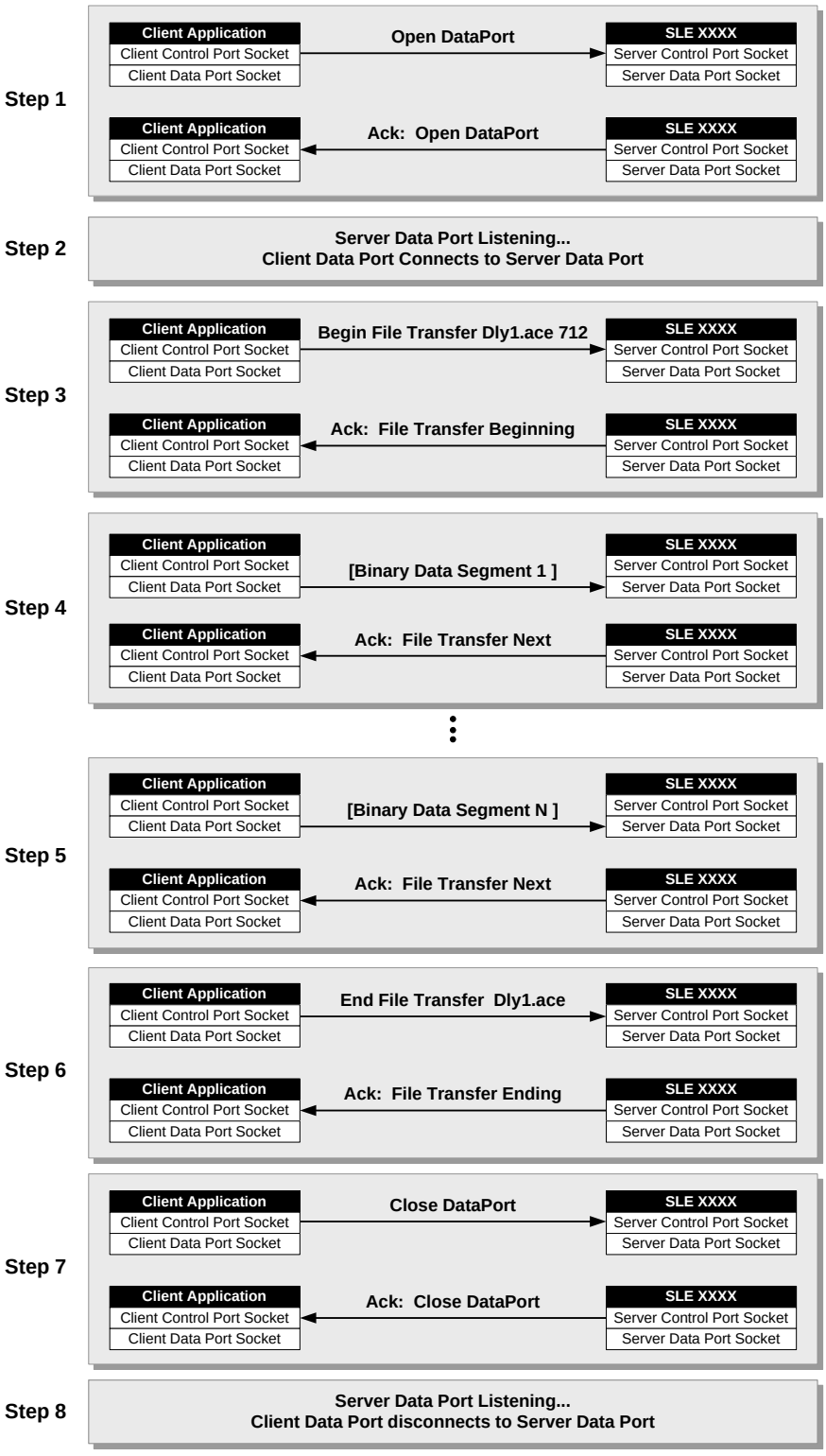
Step 1
Client Application
Client Control Port Socket
Client Data Port Socket
Open DataPort
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Client Application
Client Control Port Socket
Client Data Port Socket
Ack: Open DataPort
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Step 2
Server Data Port Listening...
Client Data Port Connects to Server Data Port
Step 3
Client Application
Client Control Port Socket
Client Data Port Socket
Begin File Transfer Dly1.ace 712
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Client Application
Client Control Port Socket
Client Data Port Socket
Ack: File Transfer Beginning
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Step 4
Client Application
Client Control Port Socket
Client Data Port Socket
[Binary Data Segment 1]
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Client Application
Client Control Port Socket
Client Data Port Socket
Ack: File Transfer Next
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Step 5
Client Application
Client Control Port Socket
Client Data Port Socket
[Binary Data Segment N]
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Client Application
Client Control Port Socket
Client Data Port Socket
Ack: File Transfer Next
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Step 6
Client Application
Client Control Port Socket
Client Data Port Socket
End File Transfer Dly1.ace
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Client Application
Client Control Port Socket
Client Data Port Socket
Ack: File Transfer Ending
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Step 7
Client Application
Client Control Port Socket
Client Data Port Socket
Close DataPort
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Client Application
Client Control Port Socket
Client Data Port Socket
Ack: Close DataPort
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Step 8
Server Data Port Listening...
Client Data Port disconnects to Server Data Port

Figure 1: File Download Protocol

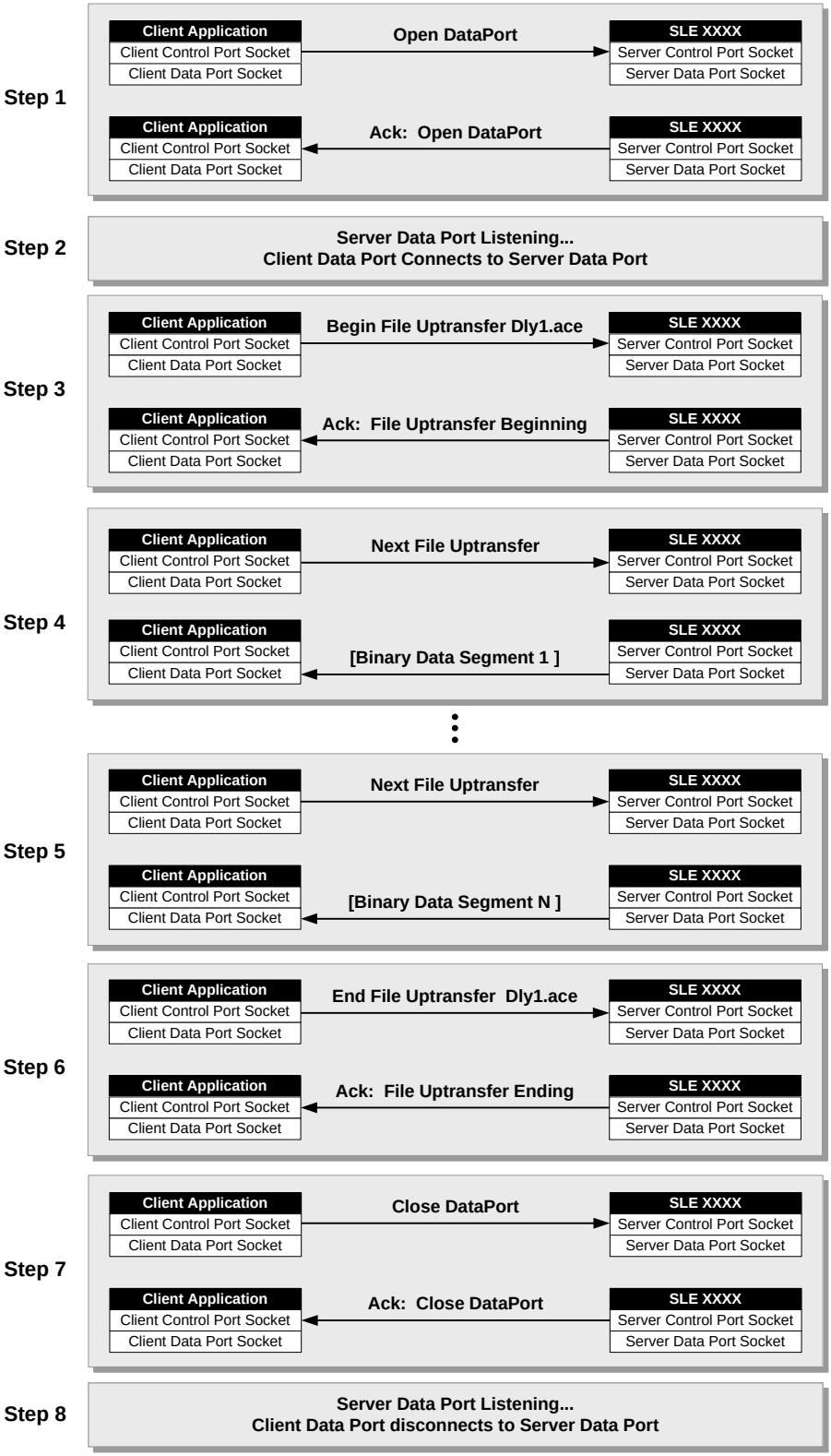
Step 1
Client Application
Client Control Port Socket
Client Data Port Socket
Open DataPort
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Client Application
Client Control Port Socket
Client Data Port Socket
Ack: Open DataPort
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Step 2
Server Data Port Listening...
Client Data Port Connects to Server Data Port
Step 3
Client Application
Client Control Port Socket
Client Data Port Socket
Begin File Uptransfer Dly1.ace
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Client Application
Client Control Port Socket
Client Data Port Socket
Ack: File Uptransfer Beginning
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Step 4
Client Application
Client Control Port Socket
Client Data Port Socket
Next File Uptransfer
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Client Application
Client Control Port Socket
Client Data Port Socket
[Binary Data Segment 1]
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Step 5
Client Application
Client Control Port Socket
Client Data Port Socket
Next File Uptransfer
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Client Application
Client Control Port Socket
Client Data Port Socket
[Binary Data Segment N]
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Step 6
Client Application
Client Control Port Socket
Client Data Port Socket
End File Uptransfer Dly1.ace
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Client Application
Client Control Port Socket
Client Data Port Socket
Ack: File Uptransfer Ending
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Step 7
Client Application
Client Control Port Socket
Client Data Port Socket
Close DataPort
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Client Application
Client Control Port Socket
Client Data Port Socket
Ack: Close DataPort
SLE XXXX
Server Control Port Socket
Server Data Port Socket
Step 8
Server Data Port Listening...
Client Data Port disconnects to Server Data Port

Figure 2: File Upload Protocol

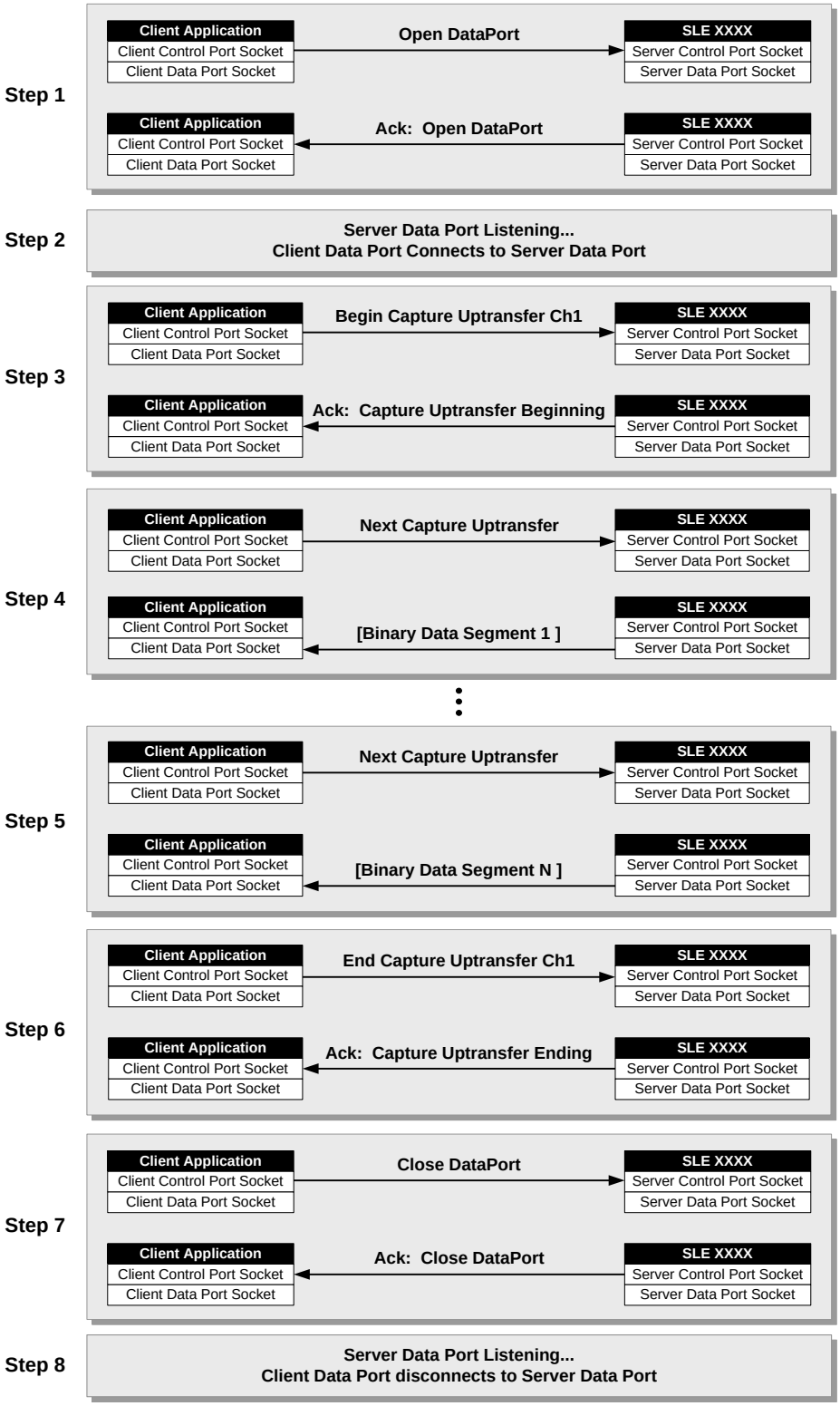

Figure 3: Capture Upload Protocol

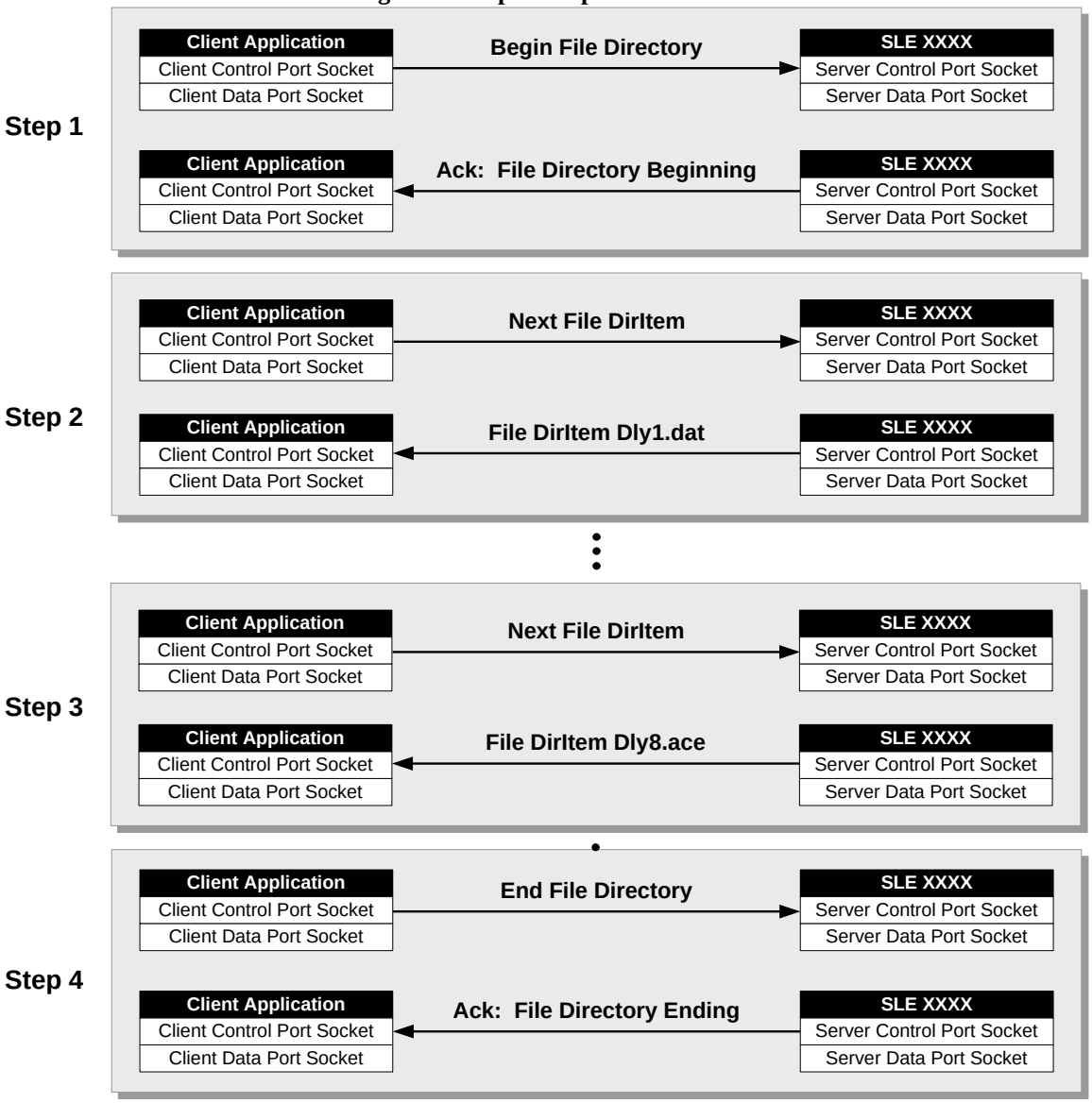

Figure 4: Uploading File Directory Protocol

CLIENT APPLICATION FEATURES

This section outlines the features provided by ***AceClient***, the advantages for customizing the user interface for different engineering objectives, and the computer requirements to run the software.

Summary of User Interface

AceClient provides an interface for ACE9600 test equipment that allows the user to customize the client layout according to individual preferences:

Multiple ACE control: A single ***AceClient*** application has the ability to communicate with a list of different ACE9600s, with the option to link the entire list for actions such as connecting, disconnecting, starting, pausing, resetting and stepping forward/back.

Save/Load Configuration: Custom configurations can be saved under a user-supplied filename. When loaded, the file will restore the list of ACE connections, all parameter settings, graph selections and connection states.

Customizable View Tabs: Individual users can manage which of the nine views appear on the screen during the current session. View preferences are saved between application executions in the Windows INI file system or by saving the entire configuration under a user-supplied filename.

Upload Preferences: Uploading dynamic data files from a connected ACE can be done manually through a file manager or automatically as each file is selected for a particular dynamic data parameter. In addition, local copies of the files can be used instead of requesting an upload from the ACE9600.

Store/Recall Settings: The Store/Recall capability available on the instrument's front panel is accessible from the client. Recalling a previously saved setting is faster compared to loading a saved configuration, which requires each parameter to be separately sent to the device.

Diverse Targeted Engineering Applications

The client application allows for customized views and preferences, depending on the engineering objective:

RF Engineers can take advantage of the Channel Summary, Link Parameter Summary and Multipath Parameter Summary views, which allows testing a system at a single "snapshot" in time and observing the change in spectrum from transmitter to receiver.

Communications Engineers will benefit from the dynamic data displayed on the Dynamic Overview, Graphs and Capture views of various communications links in a scenario.

Project Managers can setup multiple instrument settings and downloads, and then start running an entire scenario from start to finish.

Operating System Requirements

AceClient is designed to run on the Windows® operating system and uses standard controls to display and update parameters. The following requirements should be met to ensure optimal performance:

Screen Size: The client application window is a fixed size of 1278x995 pixels, therefore the monitor used to display the application should be large enough to display the entire area. If a smaller screen size is detected, scrollbars will allow access to all portions of the user interface.

Operating System: This is a PC based desktop application, and requires the following minimum configuration:

- Windows® XP or Windows® 7 operating system
- 512 MB RAM, 20 MB hard disk space

Other accessories: Other computer accessories needed include a keyboard and a two button mouse, laptop touch pad, or pen.

USING THE CLIENT APPLICATION

This section provides a detailed description of the client application's main window components, which includes the Control Bar, the Tabbed Views, and Editing the User Configuration.

Control Bar

The control bar appears on the right side of the client window. The top of the bar displays a Connection Panel that lists ACEs for the current configuration and their connection state. The middle of the bar contains the Static or Dynamic Panel, depending on the mode selected. The bottom of the bar has buttons to invoke dialogs to save the current configuration or load a previous configuration, as well as a control to allow an exit from the program.

Connection Panel

The Connection Panel appears in the upper right corner of the client window (Figure 5 and Figure 6).

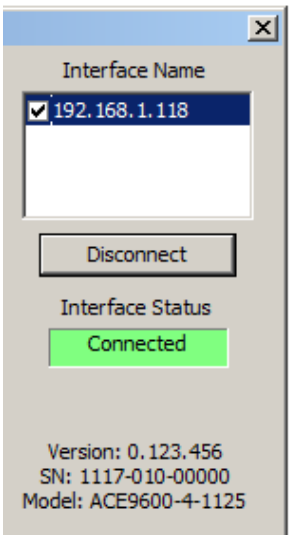

Figure 5: Panel with IP Addresses

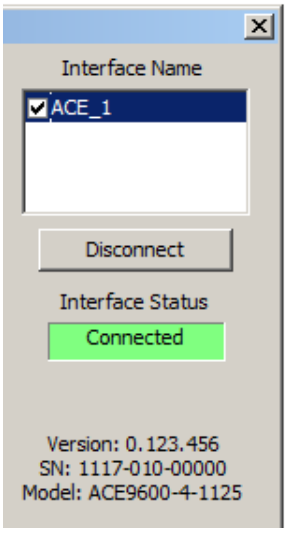

Figure 6: Panel with Instrument Names

Under the title "Interface Name", a list box contains either a descriptive name or an IP address of each instrument added to the current configuration. To add/remove items to the list or to change whether the name or IP address is displayed, see

Add/Remove Interface Property**.**

A button below the list box allows the user to connect or disconnect the currently selected ACE. The connect state also appears in a status box under the title "Interface Status". When connected, the version number, serial number, and model of the currently selected ACE will be displayed below the status.

The currently selected ACE will be highlighted with a contrasting background, and a check to the left indicates whether or not the client is currently connected to the device.

If the ACE is placed in the local mode via the instrument's front panel display, the client interface will be disabled until the "Return to Remote" button is pushed (Figure 7).

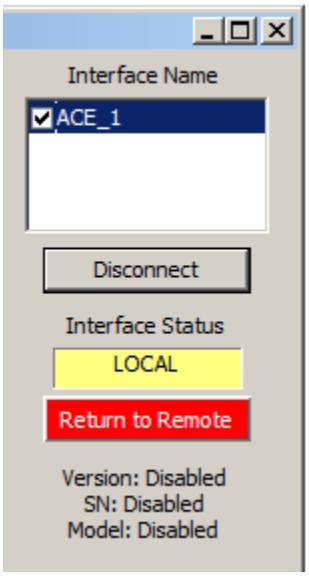

Figure 7: Panel in Local Mode

Double clicking with the left mouse button on a list box item invokes a dialog box that allows the connection information for that instrument to be changed (Figure 8).

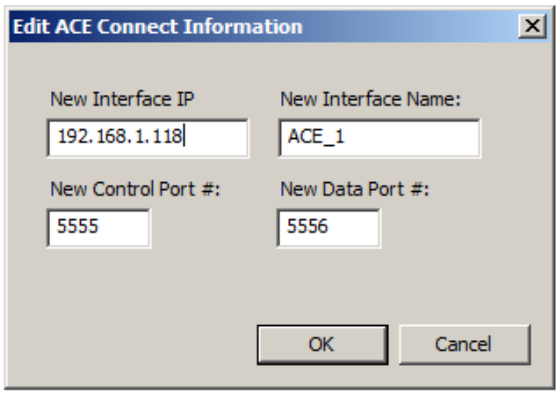

Figure 8: Edit Interface Dialog

Enter the new IP address, descriptive name, control port number and data port number in the edit boxes. Press the "OK" button to validate the connection settings and or the "Cancel" button to leave the settings unchanged.

Please note that the connection information can only be changed when the instrument is disconnected from the ACE

Static and Dynamic Panels

The Static Panel or Dynamic Panel appears in the middle right section of the client window (Figure 9 and Figure 10).

Figure 9: Static Panel

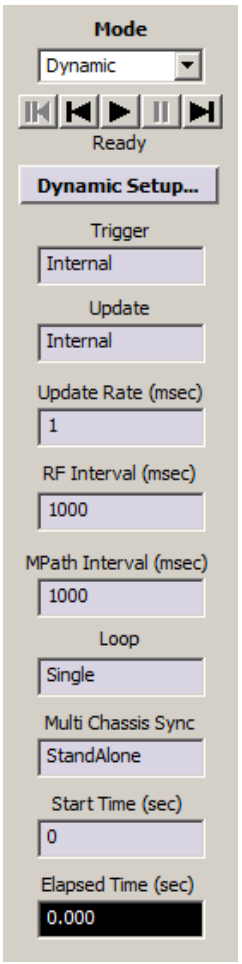

Figure 10: Dynamic Panel

Under the title "Mode", the mode of the currently selected interface appears in a drop list box. The mode can be changed to either Static or Dynamic mode if the control is enabled. Changing the mode will alter which panel is displayed in the area, as well as enable or disable other controls on the tabbed view.

In the Static Mode, the panel will contain a "Set Eb/No" button, as well as allow previously saved settings from the device to be recalled or allow the current settings to be stored.

In the Dynamic Mode, the panel contains a "Dynamic Setup..." button, controls to run a dynamic data file, and current dynamic parameter settings in status boxes.

The buttons to control a run through the dynamic data files appear as media-type play controls, including from left to right: Reset, StepBack, Play, Pause, and StepFwd. The current run State in the Dynamic Mode is displayed directly under these control buttons.

Pressing the "Dynamic Setup..." button invokes the Dynamic Run Setup dialog, which allows changes to the dynamic settings, as well as allowing for parameter file selection. Once the dialog closes, the ACE is ready to start the dynamic run (Figure 11).

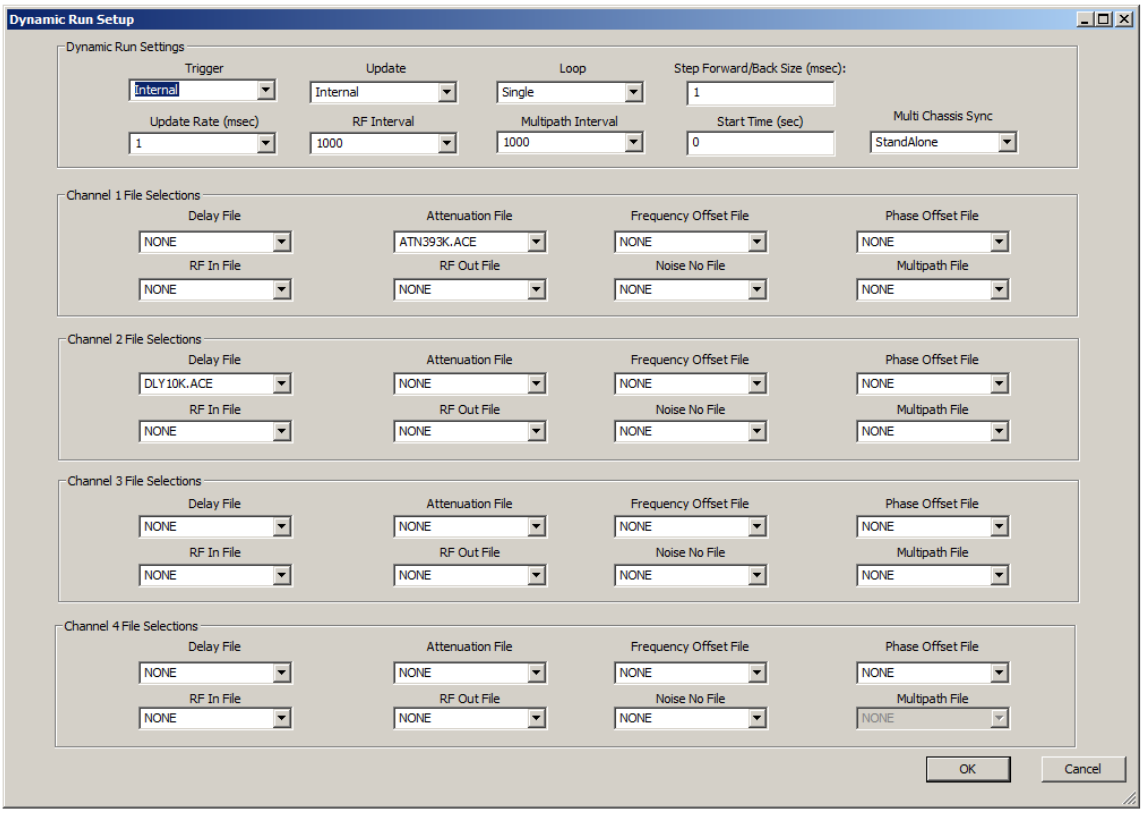

Figure 11: Dynamic Run Setup Dialog

Configuration Buttons

The "Load Config...", "Save Config...", and "Preferences..." buttons appear below the Static or Dynamic Panel on the right side of the client window (Figure 12).

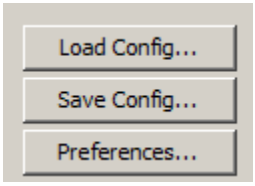

Figure 12: Load/Save Config Buttons

Pressing the "Load Config..." button will invoke a standard file dialog box to obtain the name of the previously saved configuration file. Navigate to the directory of the file and press the "Open" button to begin the load process.

Please note that loading a saved configuration will replace the current configuration completely and automatically connect to any ACE IP addresses that had been saved when connected.

A log of the commands executed during the loading of a configuration and any errors that occur will appear on the Activity Log View. See **Tab Views Property** for instructions to add the Activity Log to the tabbed views.

During the loading process, the progress in terms of percent completed will appear in the Interface Status display (Figure 13) and the "Load Config..." button changes text to "Cancel Load". The loading process can be canceled by pressing the "Cancel Load" button.

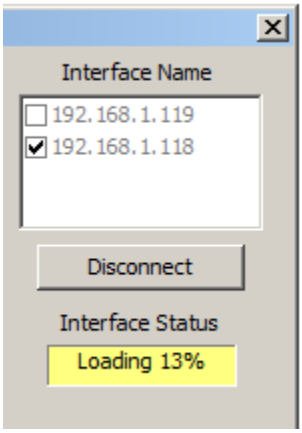

Figure 13: Load Percent Complete

Pressing the "Save Config..." button will invoke a standard file dialog box to obtain the name of the file in which to save all of the current configuration settings. Navigate to the directory of the file and press the "Save" button to begin.

Please note that the saved settings include all parameter settings, client preferences, graph settings and connection states, whereas the INI file system only saves client preferences.

Pressing the "Preferences..." button will invoke a property dialog to change the configuration settings. See **Editing the User Configuration** for more information about this dialog.

Exit Button and Scrolling

The "Exit" button and resize handle are located at the bottom right side of the client window (Figure 14).

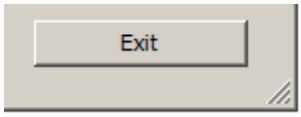

Figure 14: Exit Button

Pressing the "Exit" button will close all open socket connections and save the client preferences in the Windows INI file system that will be restored the next time the program is executed.

> *Please note that the main client window is resizable. Drag the edges of the window to resize the client, or use the resize handle to the lower right of the "Exit" button. If a screen size smaller than the client window is detected, horizontal and vertical scrollbars will appear to allow access the all controls.*

The program can also be closed by clicking the "dBm" icon in the upper, left corner of the title bar with the left mouse button and selecting "Close" (Figure 15).

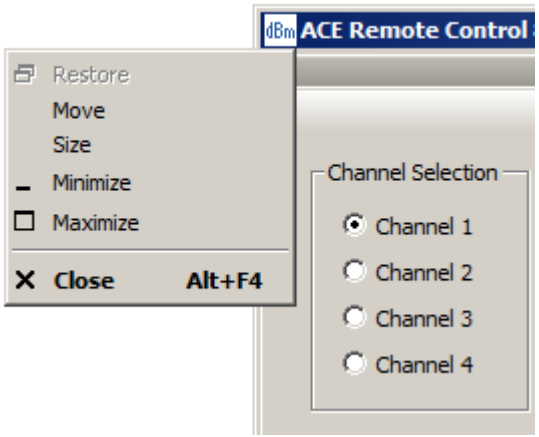

Figure 15: System Menu Minimize, Maximum and Close

Alternatively, the program can be closed by clicking the "X" cancel button in the upper, right corner of the title bar (Figure 16).

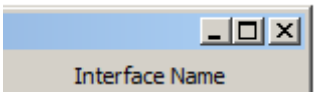

Figure 16: Minimize, Maximize and Close Buttons

Tabbed Views

A tab control on the left side of the client window is used to display preferred views. See **Tab Views Property** for instructions about how to add views to the tab control.

Channel Summary Tab

The Channel Summary tab displays all Static mode Link Parameters, AWGN Parameters, and Multipath Parameters for a given channel, as well as the Dynamic mode file selections for that channel (Figure 17). An overflow indicator for the Signal Power measurement is represented by a red LED icon next to the display.

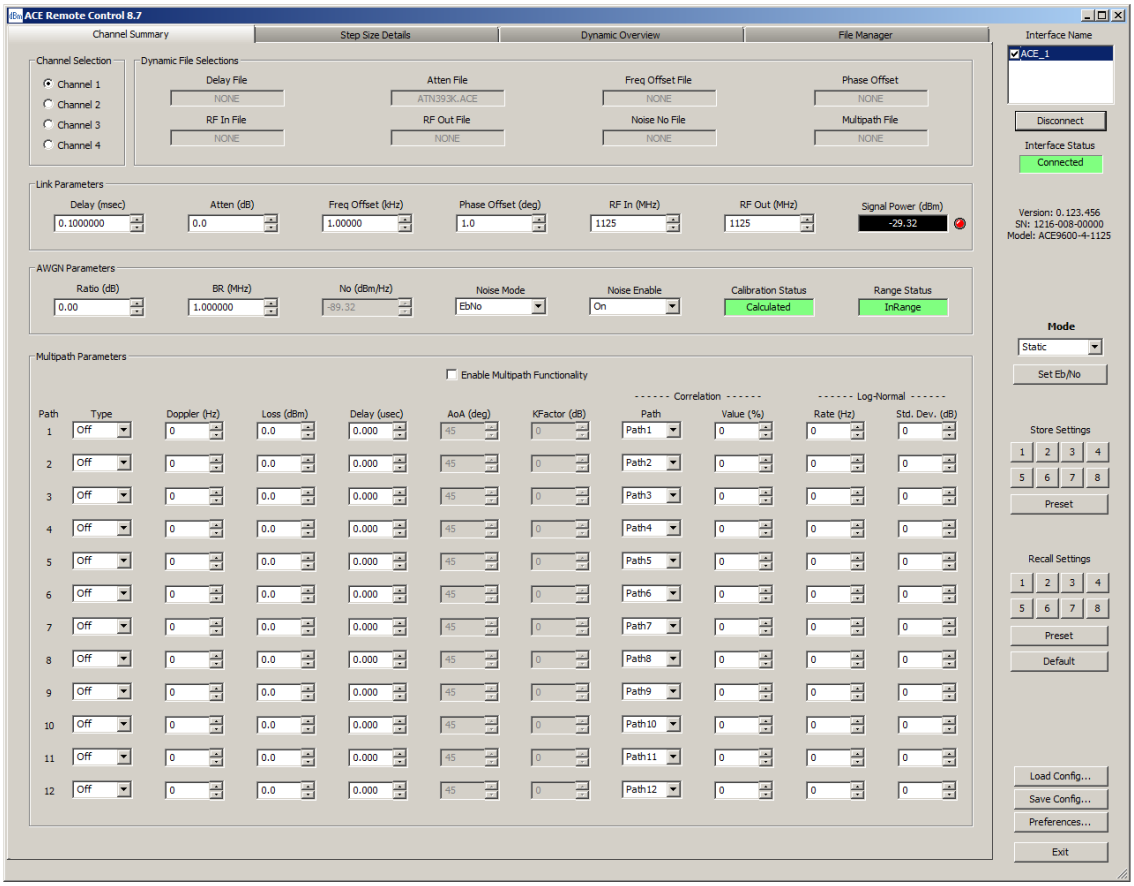

Figure 17: Channel Summary Tab

The currently selected channel is displayed with radio-type buttons on the top left of the tab. To view the summary of parameters for a different channel, click on the corresponding radio button next to the channel name.

The file selections for the channel appear in read-only boxes to the side of the channel buttons.

> *Please note that changes to Dynamic mode file selections must be made by pressing the "Dynamic Setup..." button to invoke the Dynamic Run Setup dialog. See **Static and Dynamic Panels** for more information about the Dynamic Run Setup dialog.*

Each Static mode parameter is displayed by an edit box, a drop list box of choices, or as a ready-only status box. Clicking the up/down buttons next to the edit box will increment or decrement the values by amounts that were set on the Step Size Details tab. See **Step Size Details Tab** for information about changing step values. Alternatively, a new parameter value can be entered using the keyboard, followed by the "Enter" key.

> *Please note that when the Delay value is changed, the delay control will be disabled until the ACE has completed slewing.*

A check box at the top of the Multipath Parameters section allows for the entire multipath function to be disable by un-checking the box. This has the same effect as turning all of the paths off, but allows for the paths to be returned to their original settings by simply checking the box.

> *Please note that AWGN and Multipath parameter controls will be disabled if the corresponding license is invalid. All controls will be disabled if the channel board is not present.*

Step Size Details Tab

The Step Size Details tab displays the size and units for incrementing or decrementing Static parameters, as well as the Max Delay Change with Slewing Enabled (Figure 18).

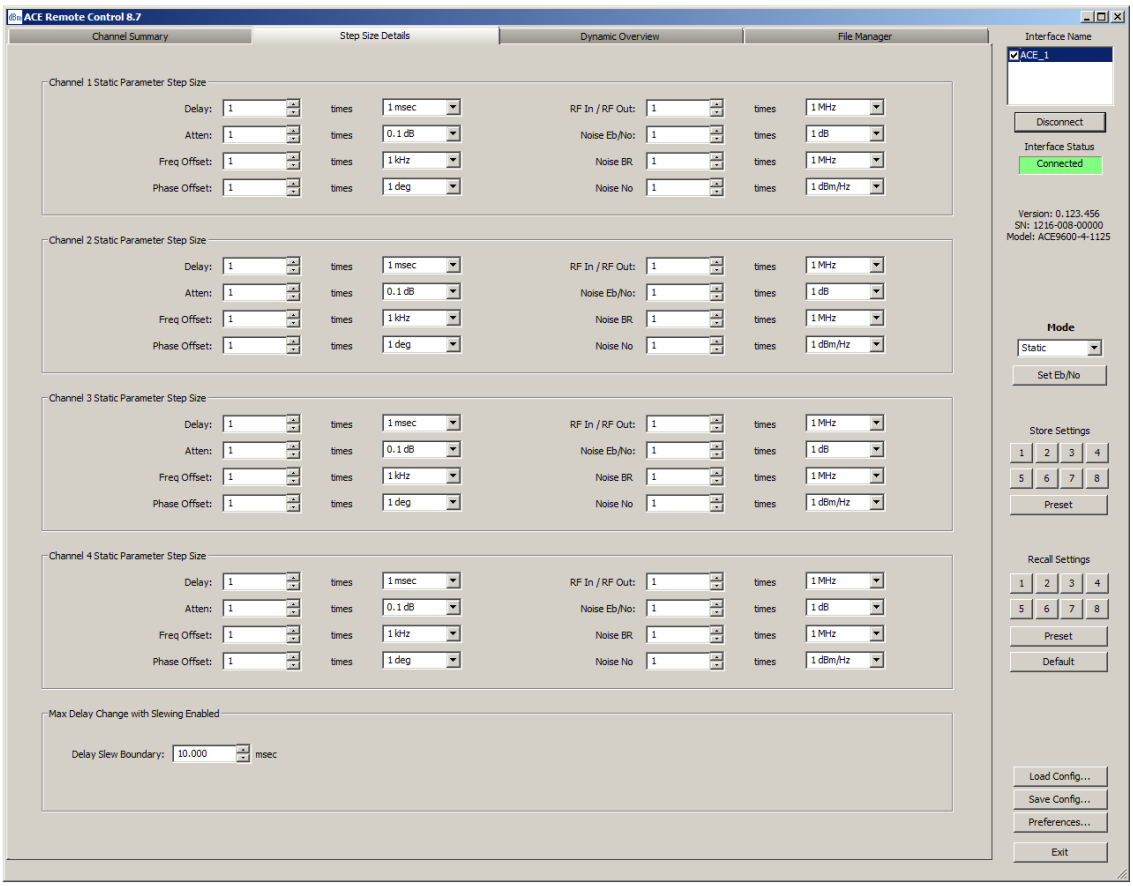

Figure 18: Details Tab

Step Sizes for Static mode parameters are displayed by an edit box, along with a drop list box of unit choices for Step Sizes. Clicking the up/down buttons next to the edit box will increase or decrease the Step Sizes set up by 1. Alternatively, a new Step Size value can be entered using the keyboard, followed by the "Enter" key.

The value for the Delay Slew Boundary is displayed by an edit box, along with the assumed units of msec. Clicking the up/down buttons next to the edit box will increase or decrease the Delay Slew Boundary by 1. Alternatively, a new Delay Slew Boundary value can be entered using the keyboard, followed by the "Enter" key.

Dynamic Overview Tab

The Dynamic Overview tab displays all the dynamic data parameters and selected graphs on a single page (Figure 19).

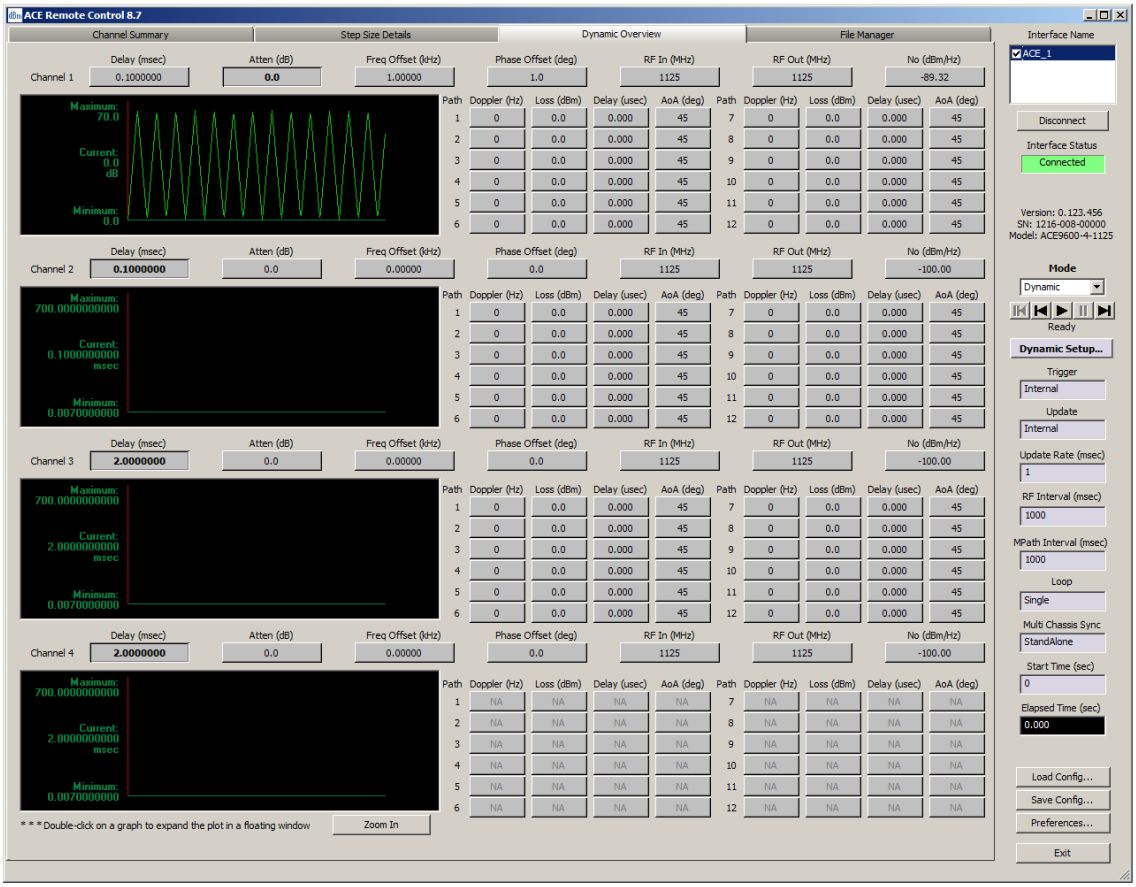

Figure 19: Dynamic Overview Tab

Current parameter values are displayed on two-state buttons. If no dynamic file has been selected for the parameter, the Static mode value is displayed.

Pressing a parameter button will display the graph for the dynamic data file that has been selected. For more information about uploading a data file, see

File Manager Tab**.**

File Manager Tab

The File Manager tab displays a list all files downloaded to ACE as well as a graph of the selected dynamic file (Figure 20).

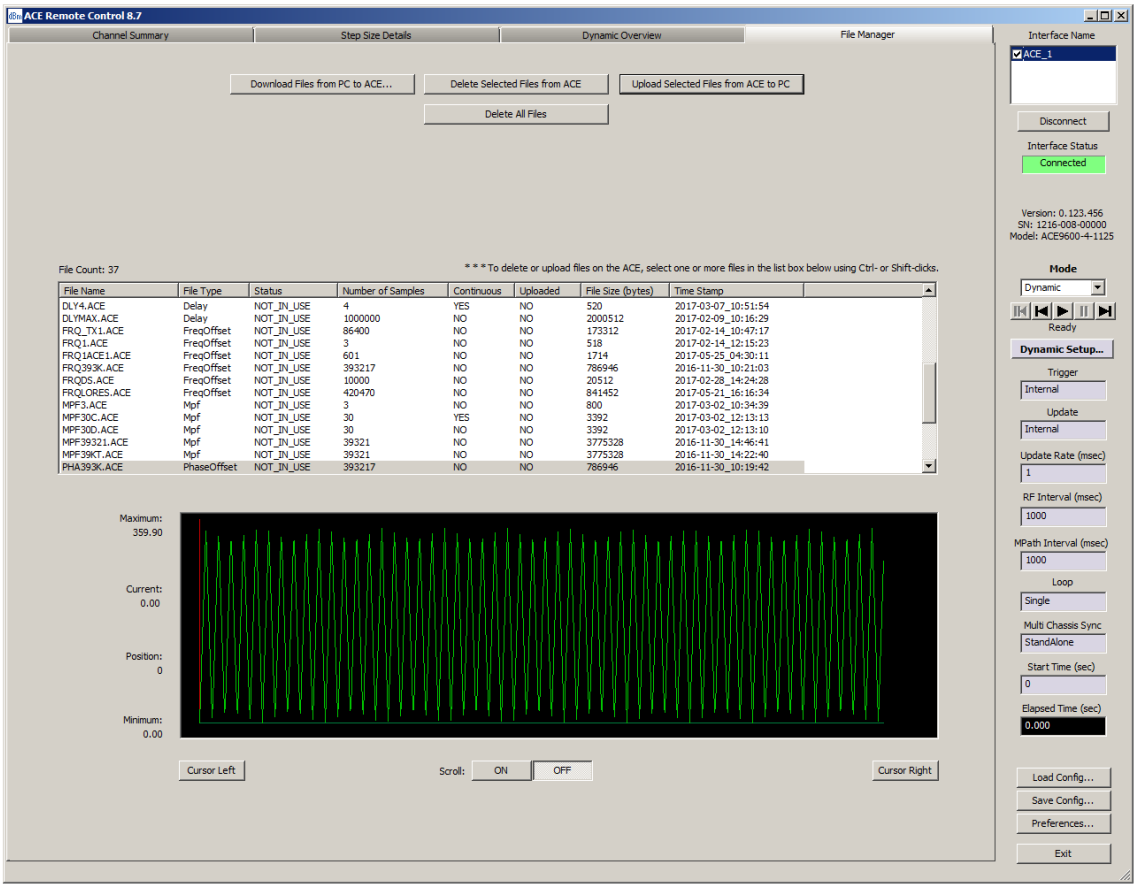

Figure 20: File Manager Tab

When connected, a list box in the center of the tab contains the file information for each file found on the selected instrument. The columns of the list box contain the preamble information that was saved during the file conversion process. The list also indicates if the file has been uploaded to the client for graphing or is currently being used by one or more channels.

A graph displays the currently selected file from the list if the dynamic data has been uploaded. The graph operates in one of two modes: compact viewing of the entire data set or detail viewing of each data sample. A scroll bar is attached to the detail view so that long data files can be examined. Press the Scroll "On"/"Off" button to toggle between graph viewing modes.

A red vertical cursor is located at the currently selected data point under observation with the corresponding dynamic value displayed on the left side of the graph. Change the cursor position by clicking on the graph or by using the "Cursor Left"/"Cursor Right" buttons.

Files can be deleted, uploaded or downloaded by pressing the buttons above the file list box. A status area is displayed while communicating with the ACE for these file tasks (Figure 21).

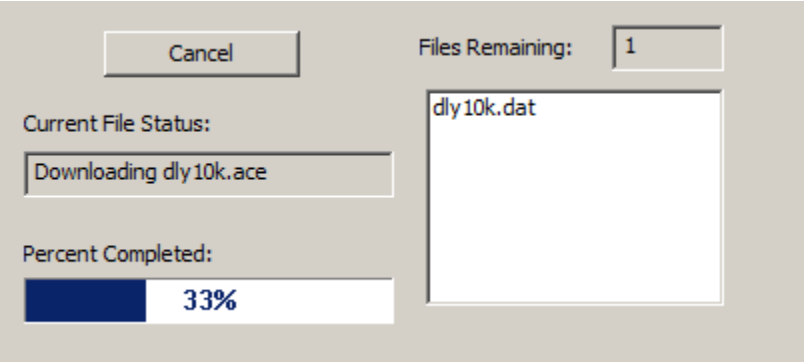

Figure 21: File Task Status

To delete one or more files, select the file name while holding down the keyboard "Ctrl" button, and then press the "Delete Selected Files" button. To delete all of the files, press the "Delete All Files" button. A message box will appear to confirm that all files will be deleted.

To upload one or more files from the ACE to the ***AceClient***, select the file name while holding down the keyboard "Ctrl" button, and then press the "Upload Selected Files" button. To change the preferences to automatically upload any file that has been selected for a channel instead of manually uploading, see **Options Property**.

To download one or more files to the currently selected ACE, press the "Download File..." button to invoke a file selection dialog box. Navigate to the directory where the files are located and select multiple files by holding the down keyboard "Ctrl" button, followed by pressing the "Open" button. See **Dynamic File Format** for more information about creating files for download. Open the ASCII text file under the name "download_log.txt" after downloading files to view a log of errors found in the file conversion process.

Link Parameter Summary Tab

The Link Parameter Summary tab displays all non-fading Static mode parameters and dynamic file selections on a single page (Figure 22).

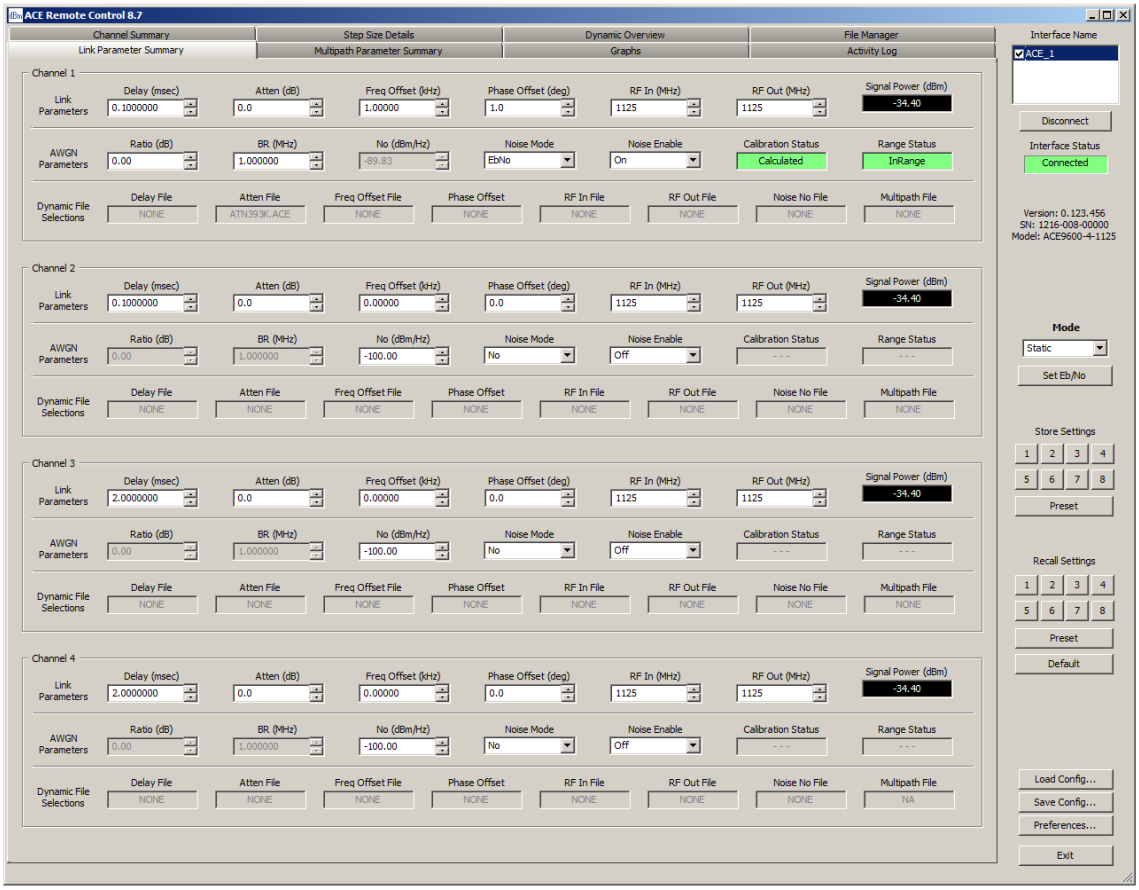

Figure 22: Link Parameter Summary Tab

Channel parameters are displayed by an edit box, a drop list box of choices, or as a read-only status box.

> *Please note that when the Delay value is changed, the delay control will be disabled until the ACE has completed slewing.*

Clicking the up/down buttons next to the edit box will increment or decrement the values set on the Step Size Details tab. See **Step Size Details Tab** for information about changing step values. Alternatively, a new parameter value can be entered using the keyboard, followed by the "Enter" key.

> *Please note that changes to the Dynamic mode file selections must be made by pressing the "Dynamic Setup..." button to invoke the Dynamic Run Setup dialog.* See ***Static and Dynamic Panels***Static

and Dynamic Panel *for more information about the Dynamic Run Setup dialog.*

Multipath Parameter Summary Tab

The Multipath Parameter Summary tab displays fading parameters for six paths at a time on a single page (Figure 23).

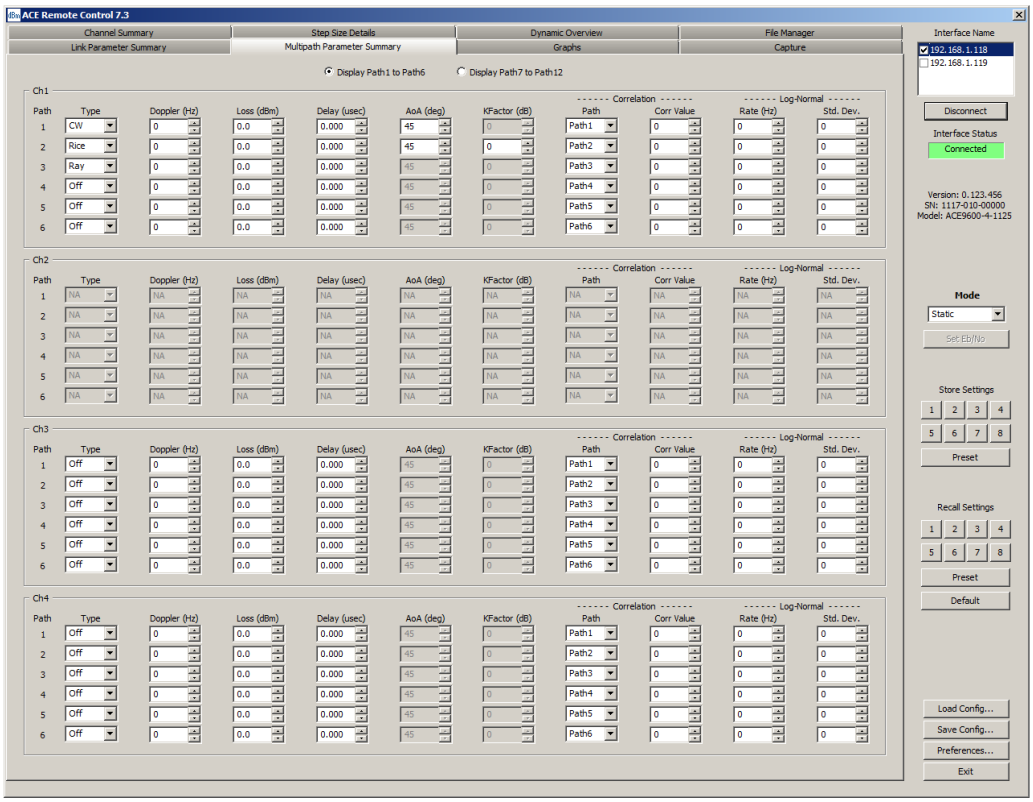

Figure 23: Summary Fading Tab

Two radio buttons appear at the top of the tab, allowing the user to view either the Paths 1-6 or Paths 7-12. Each fading parameter is represented by an edit box or a drop list box of choices.

Clicking the up/down buttons next to the edit box will increment or decrement the values by a fixed amount. Alternatively, a new parameter value can be entered using the keyboard, followed by the "Enter" key.

Graphs Tab

The Graphs tab allows for a customized display of selected graphs for dynamic data parameters on a single page (Figure 24).

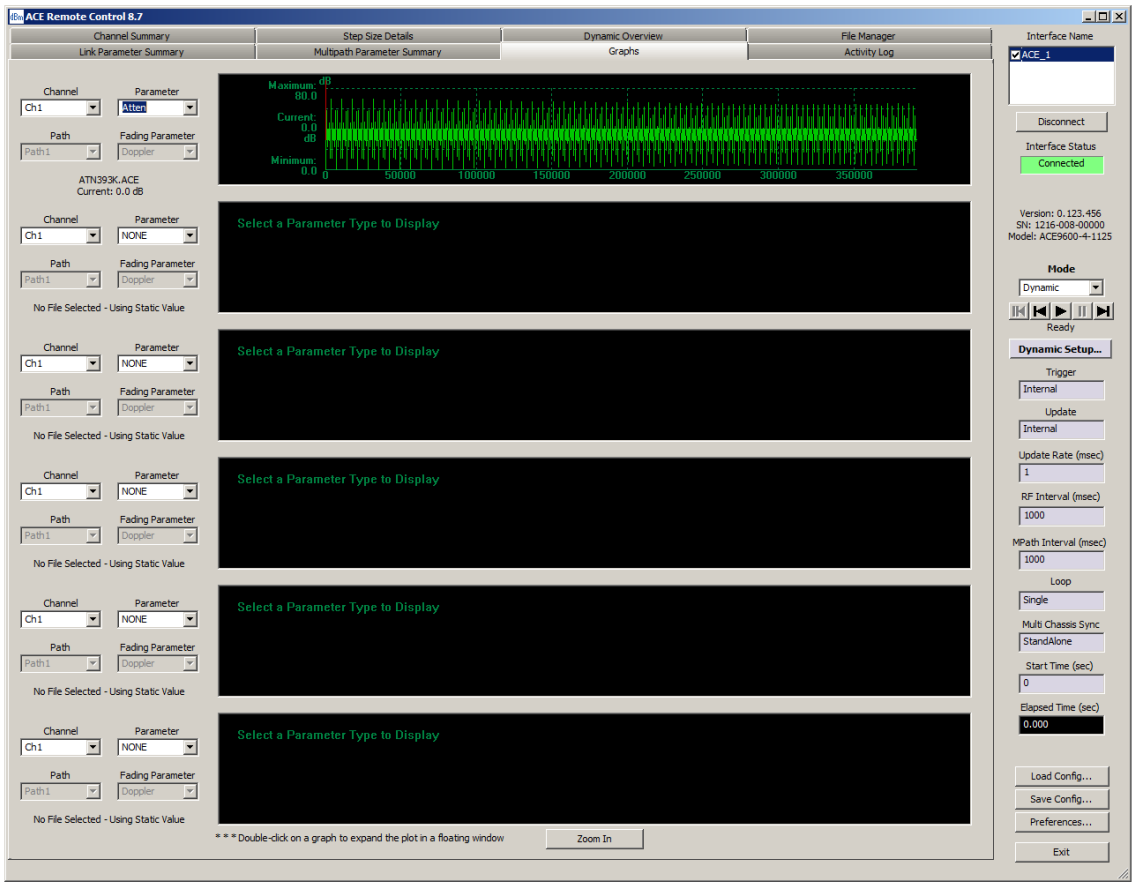

Figure 24: Graphs Tab

Each graph of uploaded data has an associated drop list box for selecting channel, parameter, path and fading parameter. If no dynamic file has been selected for the parameter, the Static mode value is graphed. Double-click on a graph to expand the plot in a floating window, or press the "Zoom In" button so that each sample in the file is displayed on the graph.

For more information about uploading file data, see

File Manager Tab**.**

Capture Tab

The Capture tab displays graphs of data that are sampled from the incoming signal to the instrument (Figure 25).

Please note that the controls for the Capture Tab are disabled when the device is operating in the Dynamic Mode.

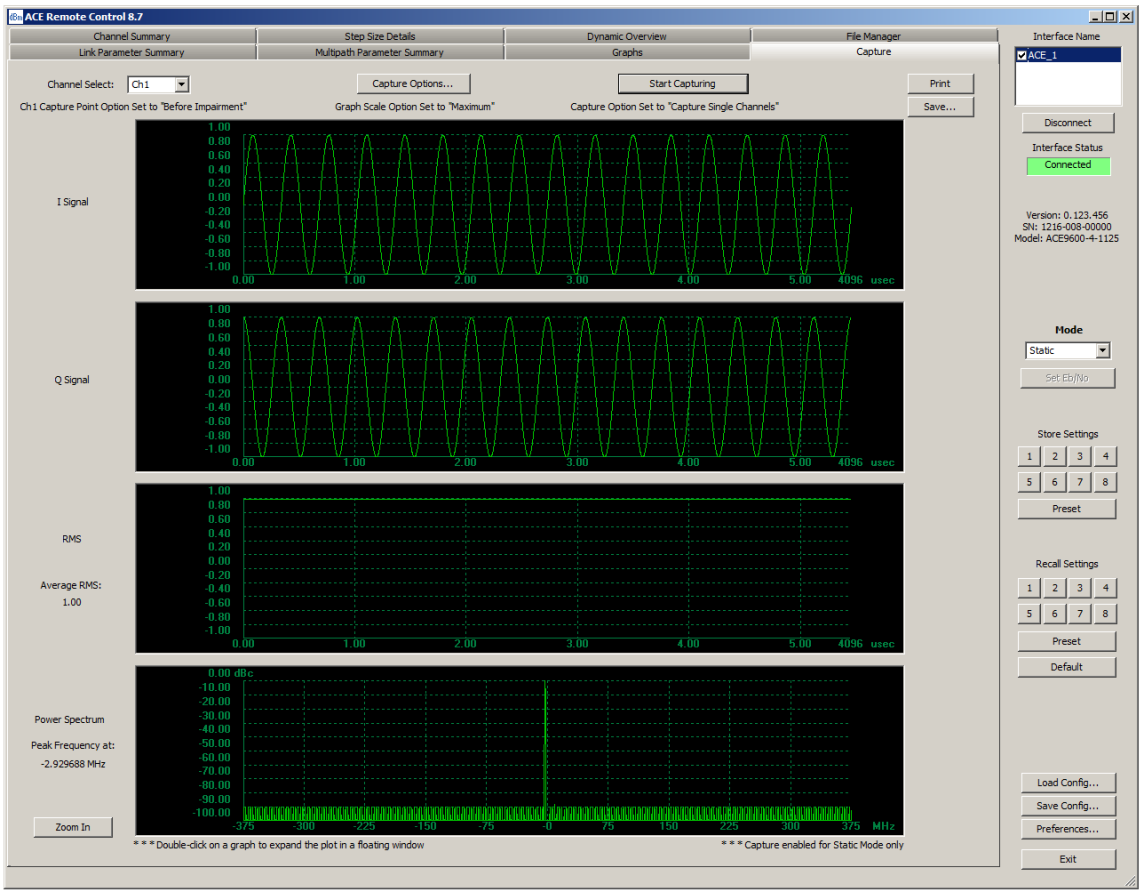

Figure 25: Capture Tab

The currently selected channel appears as drop list box at the top of the display area. The capture options for the currently selected channel appear above the graphs. To change the capture options, press the "Capture Options..." button (Figure 26).

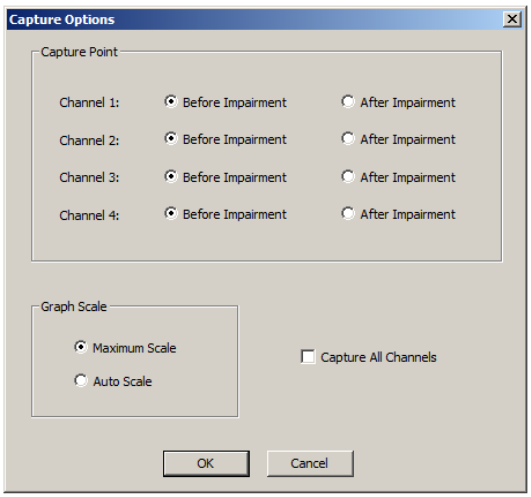

Figure 26: Capture Options Dialog

To capture samples of the input signal going into the ACE, press the "Start Capturing" at the top of the display area. If the "Capture All Channels" option is selected, the input signal from each channel will be captured. Otherwise, only the currently selected channel will be captured.

The input signal for the currently selected channel is displayed in four different graphs at the lower right of the tab. The I and Q components of the signal are displayed in the first two graphs.

The graph of the composite RMS representation of the signal appears in the third graph, along with the display of the Average RMS value computed from the sample set to the left of the graph.

The power spectrum of the signal is also computed and displayed in the bottom graph, with the peak frequency displayed to the left of the graph.

All captured graphs can be printed or saved by pressing the "Print" or "Save..buttons at the top right of the tab.

Activity Log Tab

The Activity Log tab displays the current settings of the selected ACE in text boxes, as well as text logs of communication between the ***AceClient*** to the ACE (Figure 27).

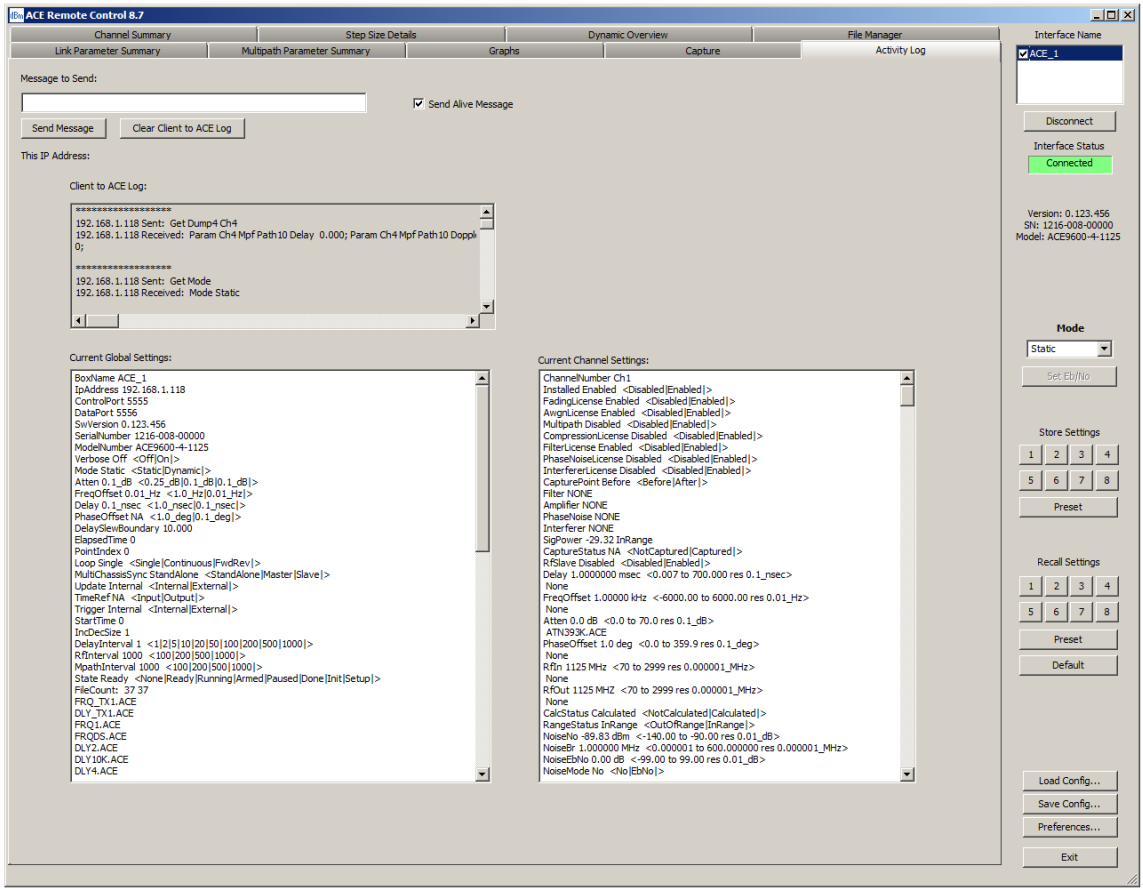

Figure 27: Activity Log Tab

All text boxes on the view have scrollbars to allow viewing long lists collected data.

Each text message from the client application to the ACE9600 has a corresponding response on either the control port or the data port.

Editing the User Configuration

The ***AceClient*** application can be customized for a variety of engineering applications by pressing the "Preferences..." button on the side Control Bar to invoke a property dialog.

Please note that none of the changes made to the configuration will take effect until the Edit Configuration dialog box is closed with the "OK" button.

Add/Remove Interface Property

The Add/Remove Interface property allows for adding, removing, or re-ordering the list of instruments in the current configuration (Figure 28).

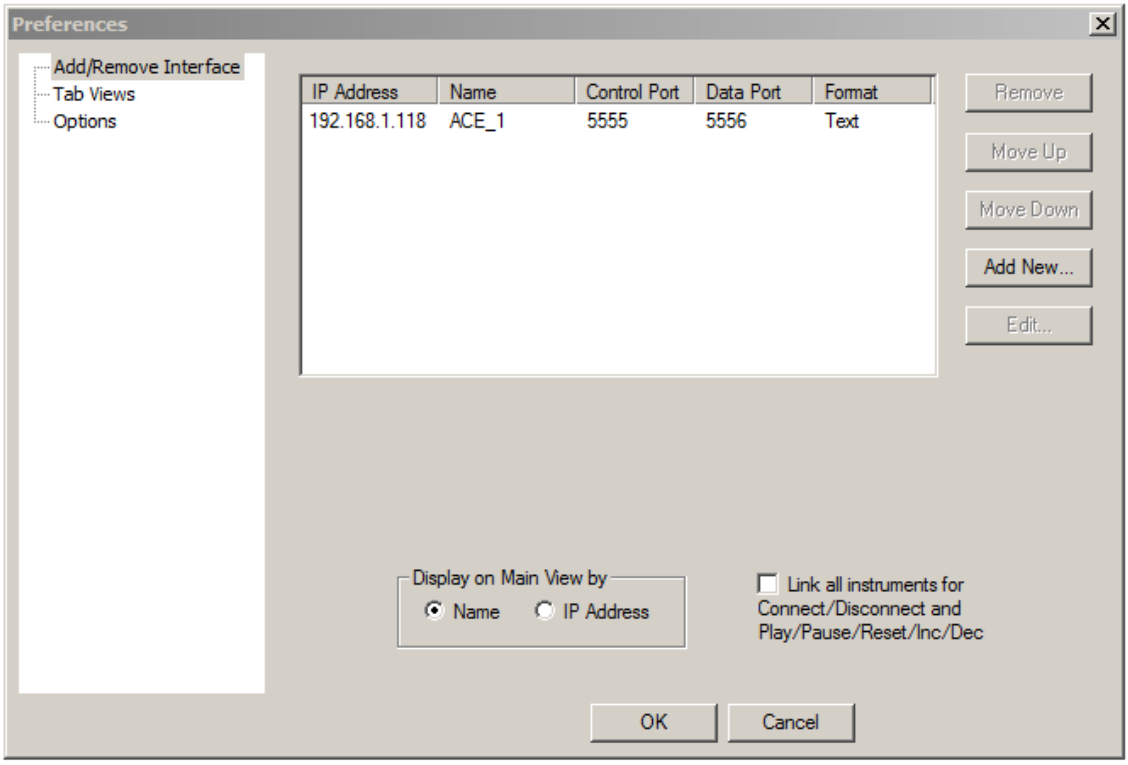

Figure 28: Add/Remove Interface Property

To add a new instrument to the configuration, press the "Add New..." button to invoke the "Add ACE Connect Information" dialog. Enter the IP address, descriptive name and port numbers in the edit boxes. Press the "OK" button to confirm the connection settings and add the new connection to the bottom of the list.

To remove an existing device in the configuration, select an item in the list and press the "Remove" button.

To move an existing device up or down in the list, select the item and press the "Move Up" or "Move Down" button until it is in the desired order.

To display either the descriptive name or IP address in the Control Bar connection list, click the corresponding radio button under the "Display on Main View by" title.

To connect the dynamic data run controls of all connected instruments, check the "Link all instruments" box.

Tab Views Property

The Tab Views property allows the selection of tabbed views shown on the main window tab control (Figure 29).

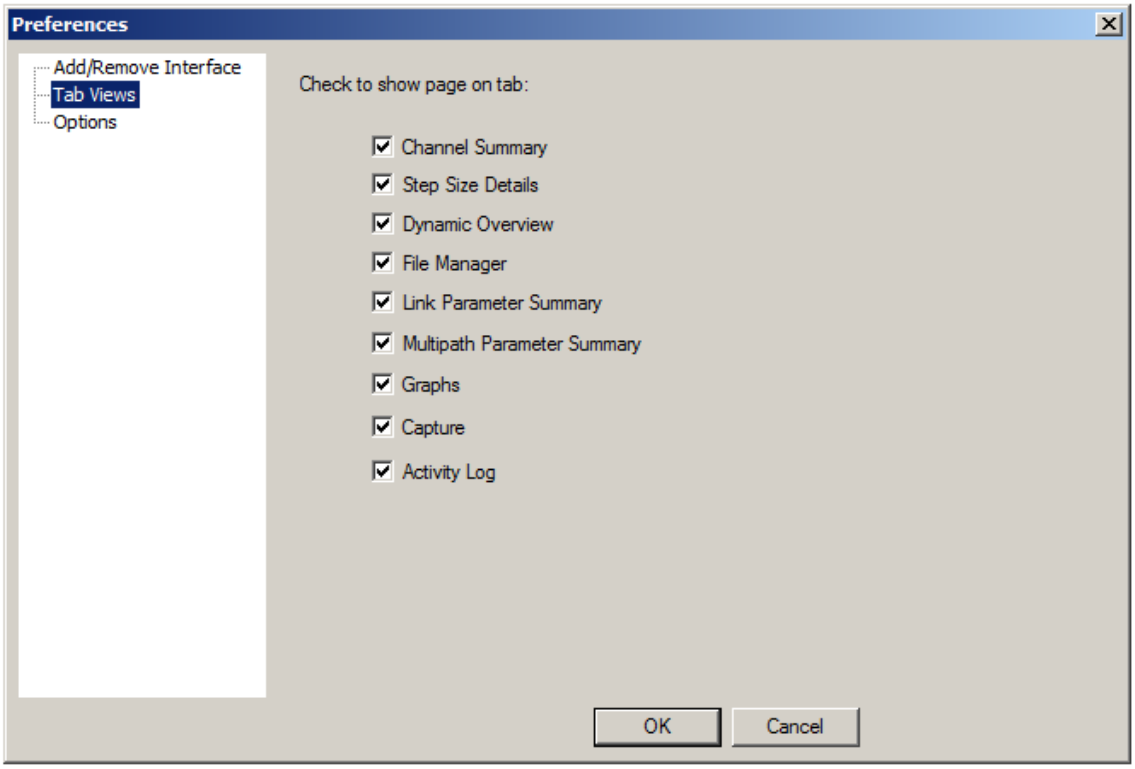

Figure 29: Tab Views Property

To include a view on the main window tab control, click the check box next to the page name until a check appears.

- The Channel Summary tab displays all Static mode Link Parameters, AWGN Parameters, and Multipath Parameters for a given channel, as well as the Dynamic mode file selections for that channel. For more information about this tab, see **Channel Summary Tab.**
- The Step Size Details tab displays the size and units for incrementing or decrementing Static parameters, as well as the Max Delay Change with Slewing Enabled. For more information about this tab, see **Step Size Details Tab.**
- The Dynamic Overview tab displays all the dynamic data parameters and selected graphs on a single page. For more information about this tab, see

- **Dynamic Overview** Tab.
- The File Manager tab displays a list all files downloaded to ACE as well as a graph of selected dynamic file data. For more information about this tab, see

- **File Manager** Tab**.**
- The Link Parameter Summary tab displays all non-fading Static mode parameters and dynamic file selections on a single page. For more information about this tab, see

- **Link Parameter** Summary Tab**.**
- The Multipath Parameter Summary tab displays fading parameters for six paths at a time on a single page. For more information about this tab, see **Multipath Parameter Summary** Tab.
- The Graphs tab allows for a customized display of selected graphs for dynamic data parameters on a single page. For more information about this tab, see

- **Graphs** Tab.
- The Capture tab displays graphs of data that are sampled from the incoming signal to the instrument. For more information about this tab, see

- **Capture** Tab**.**
- The Activity Log tab displays the current settings of the selected ACE in text boxes, as well as text box logs of communication between the ***AceClient*** to instrument. For more information about this tab, see

- **Activity Log** Tab**.**

Options Property

The Options property allows changes to preferences for uploading files (Figure 30).

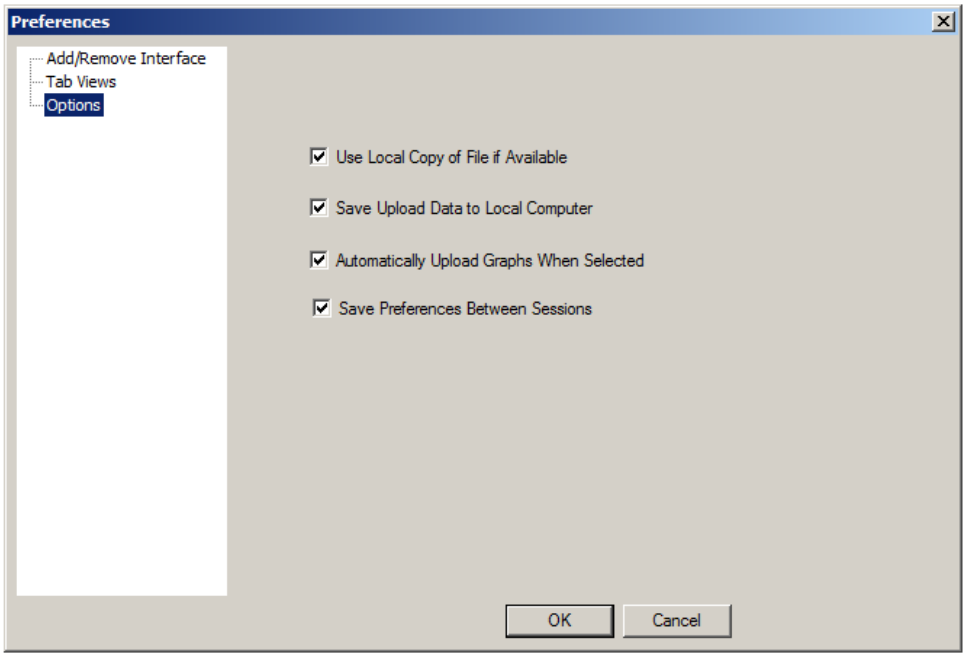

Figure 30: Preferences Property

Please note that the saved settings by pressing the "Save Config..." button include all parameter settings, client preferences, graph settings and connection states, whereas the INI file system only saves client preferences.

To enable a particular option, check the box next to the option.

- Each dynamic data file that is converted and downloaded by ***AceClient*** will contain a unique timestamp identifier in the preamble. A local copy of the converted file is saved to the execution file directory. To use this local copy of the converted file when uploading graph data, check the "Use Local Copy of File if Available" box. If the unique timestamp of the file does not match the preamble of the file residing on the instrument or is not found locally, the file will be uploaded from the device.
- To save a copy of the file uploaded from the instrument on the local computer executing ***AceClient***, check the "Save Upload Data to Local Computer" box. This may speed the upload process on subsequent executions of the client application.

- To automatically upload the dynamic data file when a file is selected on the tab views, check the "Automatically Upload Graphs When Selected" box. If this preference is not selected, then the File Manager tab must be used to manually upload dynamic data files for viewing. For more information about manual uploads, see

- **File Manager** Tab**.**
- To save the preferences to the INI file system between sessions, check the "Save Preferences Between Sessions" box. If this preference is not selected, one ACE interface with default settings will be entered into the interface list. Also, four default tab views will be on display (Channel Summary, Step Size Details, Dynamic Overview, and File Manager).

Dynamic File Format

The ACE distinguishes parameter file types by the first three letters in each file name. File names can be up to 8 alphanumeric characters with the file extension of ".dat":

DLYxxxxx.dat - designates a delay file
FRQxxxxx.dat - designates a frequency offset file
ATNxxxxx.dat - designates an attenuation file
PHAxxxxx.dat - designates a phase offset file
RFFxxxxx.dat – designates an RF frequency file
WGNxxxx.dat – designates a noise density file
MPFxxxxx.dat – designates a multipath fading file

Parameter data files must be generated in ASCII format. The first line in the file is a value that represents the number of sample points in the file. Delay and FreqOffset type files also have an optional resolution value. See **Delay Type File Format** and **Frequency Offset Type File Format** sections below for more information.

Each subsequent line will contain one data value, with lines separated by a carriage return. Multipath files have a different format. See the **Multipath Type File Format** section below for more information.

When the files are converted (optimized), the files have the same prefix, but the suffix becomes ".ACE". The converted files are then transferred over the LAN interface, and stored in flash memory. The converted files are in binary format (to optimize storage) with an ASCII header (so that information can be displayed on the front panel).

Delay Type File Format

Delay type filenames must begin with "DLY" and end with ".DAT". The first line in the file is a value that represents the number of sample points in the file. Delay type files also have an optional resolution value in msec which is separated by an ASCII space character from the number of sample points. If no resolution value exists in the data file, the assumed resolution is 0.0000001 msec (100 psec).

Valid resolutions in msec are :

- 0.0000001 (100 psec)
- 0.00000001 (10 psec)
- 0.000000001 (1 psec)
- 0.0000000001 (0.1 psec)

The maximum delta Delay between samples is limited by the resolution value:

Resolution in msec	Maximum Delta
0.0000001 (100 psec)	+/- 0.002000 msec
0.00000001 (10 psec)	+/- 0.00032767 msec
0.000000001 (1 psec)	+/- 0.000032767 msec

0.0000000001 (0.1 psec)	+/- 0.0000032767 msec

Each subsequent line will contain one data value in the assumed units of msec, with lines separated by a carriage return. The maximum range for the data values is determined by the minimum and maximum delay of the instrument.

A sample data file containing three sample points with a resolution of 10 psec is as follows:

```
3 0.00000001
100.00000000
100.00000100
100.00000000
```

Frequency Offset Type File Format

Frequency Offset type filenames must begin with “FRQ" and end with “.DAT”. The first line in the file is a value that represents the number of sample points in the file. Frequency Offset type files also have an optional resolution value in kHz which is separated by an ASCII space character from the number of sample points. If no resolution value exists in the data file, the assumed resolution is 0.001 kHz (1 Hz).

Valid resolutions in kHz are :

- 0.001 (1Hz)
- 0.0001 (10 mHz)

The maximum delta Frequency Offset between samples is limited by the resolution value:

Resolution in kHz	Maximum Delta
0.001 (1Hz)	+/- 32.767 kHz
0.0001 (10 mHz)	+/- 0.32767 kHz

Each subsequent line will contain one data value in the assumed units of kHz, with lines separated by a carriage return. The maximum range for the data values is determined by the minimum and maximum Frequency Offset of the instrument.

A sample data file containing three sample points with a resolution of 10 mHz is as follows:

```
3 0.00001
-4.659803
-4.659997
-4.660190
```

Phase Offset Type File Format

Phase Offset type filenames must begin with “PHA" and end with “.DAT”. The first line in the file is a value that represents the number of sample points in the file. The assumed resolution is 0.1 degrees.

Each subsequent line will contain one data value in the assumed units of degrees, with lines separated by a carriage return. The maximum range for the data values is 0.0 to 359.9 degrees.

A sample data file containing three sample points with a resolution of 0.1 degree is as follows:

```
3
179.9
150.5
0.2
```

Attenuation Type File Format

Attenuation type filenames must begin with “ATN" and end with “.DAT”. The first line in the file is a value that represents the number of sample points in the file. The assumed resolution is 0.1 dB.

Each subsequent line will contain one data value in the assumed units of degrees, with lines separated by a carriage return. The maximum range for the data values is determined by the minimum and maximum Attenuation of the instrument.

A sample data file containing three sample points with a resolution of 0.1 dB is as follows:

```
3
1.1
1.2
1.7
```

RF Type File Format

RF type filenames must begin with “RFF" and end with “.DAT”. The first line in the file is a value that represents the number of sample points in the file. The assumed resolution is 1 MHz.

Each subsequent line will contain one data value in the assumed units of degrees, with lines separated by a carriage return. The maximum range for the data values is determined by the minimum and maximum RF Input of the instrument.

A sample data file containing three sample points with a resolution of 1 MHz is as follows:

```
3
```

```
1127
1126
1125
```

Noise Type File Format

Noise type filenames must begin with “WGN" and end with “.DAT”. The first line in the file is a value that represents the number of sample points in the file. The assumed resolution is 0.01 dBm.

Each subsequent line will contain one data value in the assumed units of dBm, with lines separated by a carriage return. The maximum range for the data values is determined by the minimum and maximum Noise Density of the instrument.

A sample data file containing three sample points with a resolution of 0.01 dB is as follows:

```
3
-130.00
-90.25
-90.75
```

Multipath Type File Format

All of the twelve paths in a multipath channel can be driven dynamically by a single data file. During a dynamic run, the path values for path Doppler, Delay, Loss, and AoA are modified sequentially according to the fading file values.

Multipath type filenames must begin with “MPF" and end with “.DAT”. The first line in the file is a value that represents the number of sample points in the file.

The second through thirteenth line in the file represents the path Type, Correlation Path, Correlation Value, Rate, Std Dev, and K-factor settings, which remain fixed during a dynamic run.

Each subsequent line contains Doppler, Loss, Delay and AoA values for each of the 12 paths. Each of these path settings must begin with a semi-colon delimiter ";" and use an ASCII space character to separate the four values. If a path is off, the values between semi-colons are ignored.

The assumed resolution, range and units for each value is as follows:

Type	Units	Resolution	Range
CorrVal	%	1%	0 to 100%
Rate	Hz	1 Hz	0 to 100 Hz
StdDev	dB	1 dB	0 to 12 dB
Kfactor	dB	1 dB	-10 to 20 dB
Doppler	Hz	1 Hz	0 to 10,000Hz
Loss	dBm	0.1 dBm	0.0 to 30.0 dBm

Delay	usec	0.001 usec	0.000 to 9.996 usec
Aoa	degrees	1 degree	0 to 180 degrees

A sample data file containing three sample points with two paths on is as follows:

```
3
RICE 1 20 0 0 10
OFF
OFF
OFF
OFF
OFF
OFF
OFF
OFF
OFF
OFF
RICE 12 30 10 10 20
;0 0.000 0.0 0;;;;;;;;;;;11 0.011 1.1 11;
;12 0.012 1.2 12;;;;;;;;;;;23 0.023 2.3 23;
;24 0.024 2.4 24;;;;;;;;;;;35 0.035 3.5 35;
```

Example Code in Python

The following Python example shows the procedure for connecting with the control and data ports, as well as accessing and changing parameter settings:

```
import socket
import os

#----create list of commands
command_array = ['Set Mode Static',
'Set Param Ch1 Delay 2.0000000',
'Set Param Ch1 FreqOffset 0.00000',
'Set Param Ch1 Atten 0.0',
'Set Param Ch1 PhaseOffset 0.0',
'Set Param Ch1 RfIn 1125',
'Set Param Ch1 RfOut 1125',
'Set Param Ch1 NoiseEnable On',
'Set Param Ch1 NoiseMode No',
'Set Param Ch1 NoiseNo -100.00',
'Set Param Ch1 NoiseMode EbNo',
'Set Param Ch1 NoiseBr 1.000000',
'Set Param Ch1 NoiseEbNo 0.00',
'Set Param Ch1 NoiseMode No',
'Set Param Ch1 NoiseEnable Off',
'Set Param Ch1 Mpf Path1 Delay 0.000',
'Set Param Ch1 Mpf Path1 Doppler 0',
'Set Param Ch1 Mpf Path1 Loss 0.0',
'Set Param Ch1 Mpf Path1 Aoa 45',
'Set Param Ch1 Mpf Path1 CorrVal 0',
'Set Param Ch1 Mpf Path1 KFactor 0',
'Set Param Ch1 Mpf Path1 Type Rice',
'Set Param Ch1 Mpf Path1 CorrPath Path1',
'Set Param Ch1 Mpf Path1 Rate 0',
'Set Param Ch1 Mpf Path1 StdDev 0',
'Set Param Ch1 Mpf Path2 Type Off',
'Set Param Ch1 Mpf Path3 Type Off',
'Set Param Ch1 Mpf Path4 Type Off',
'Set Param Ch1 Mpf Path5 Type Off',
'Set Param Ch1 Mpf Path6 Type Off',
'Set Param Ch1 Mpf Path7 Type Off',
'Set Param Ch1 Mpf Path8 Type Off',
'Set Param Ch1 Mpf Path9 Type Off',
'Set Param Ch1 Mpf Path10 Type Off',
'Set Param Ch1 Mpf Path11 Type Off',
'Set Param Ch1 Mpf Path12 Type Off',
'Set Mode Dynamic',
'Begin Dynamic Setup',
'Set Dynamic Loop Single',
'Set Dynamic Update Internal',
'Set Dynamic Trigger Internal',
'Set Dynamic StartTime 0',
'Set Dynamic DelayInterval 1',
'Set Dynamic RfInterval 100',
'Set Dynamic MpathInterval 10',
```

```
'Set File Ch1 Delay dly10k.ace',
'Set File Ch1 FreqOffset None',
'Set File Ch1 Atten None',
'Set File Ch1 PhaseOffset None',
'Set File Ch1 RfIn None',
'Set File Ch1 RfOut None',
'Set File Ch1 NoiseNo None',
'Set File Ch1 Mpf None',
'Begin Dynamic Init',
'Start']

#----define port numbers
host = '192.168.1.118'
control_portnum = 5555
data_portnum = 5556
buffer_size = 1024

#----connect to control and data ports
try:
controlport = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
controlport.connect((host, control_portnum))
print ("Control port opened")
except:
print ("Failed to open control port")
input('Enter key to end ')
exit()

#----send initial hello command
message = 'Hello'
print ("Sending hello command")
controlport.send(message.encode('utf-8'))
data = controlport.recv(buffer_size)
print("Received - " + data.decode('utf-8') )

#----send command to open data port
message = 'Open DataPort'
print ("Sending Open DataPort command")
controlport.send(message.encode('utf-8'))
data = controlport.recv(buffer_size)
print("Received - " + data.decode('utf-8') )
try:
dataport = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
dataport.connect((host, data_portnum))
print ("Data port opened")
except:
print ("Failed to open data port")
input('Enter key to end ')
controlport.close()
exit()

print("--------")
print("Check to see if in Dynamic mode after connecting")
#----get mode and state
message = "Get Mode"
```

```
print ("Sending get mode command")
controlport.send(message.encode('utf-8'))
data = controlport.recv(buffer_size)
print("Received - " + data.decode('utf-8') )
mode = data.decode('utf-8').upper()

message = "Get Dynamic State"
print ("Sending get dynamic state command")
controlport.send(message.encode('utf-8'))
data = controlport.recv(buffer_size)
print("Received - " + data.decode('utf-8') )
state = data.decode('utf-8').upper()
testmode = "ACK: MODE DYNAMIC"
teststate = "ACK: DYNAMIC STATE READY"
if testmode in mode:
if teststate in state:
print("In ready state, able to continue")
else:
message = "Reset"
print ("Not in ready state, sending reset command")
controlport.send(message.encode('utf-8'))
data = controlport.recv(buffer_size)
print("Received - " + data.decode('utf-8') )
else:
print("In Static mode, able to continue")

print("--------")
print("Starting Profile Setup")
#----send profile setup
for commands in command_array:
print ("Sent - " + commands)
controlport.send(commands.encode('utf-8'))
data = controlport.recv(buffer_size)
print("Received - " + data.decode('utf-8') )
print("--------")

print("Done with Profile Setup")
#----get state
message = "Get Dynamic State"
print ("Sending get state command")
controlport.send(message.encode('utf-8'))
data = controlport.recv(buffer_size)
print("Received - " + data.decode('utf-8') )
print("--------")

#----send close data port
message = "Close DataPort"
print ("Sending Close DataPort command")
controlport.send(message.encode('utf-8'))
data = controlport.recv(buffer_size)
print("Received - " + data.decode('utf-8') )

print ("Closing data port")
dataport.close()
```

```
print ("Closing control port")
controlport.close()
```

Appendix A: Supplementary Download/Upload Protocol Diagrams

The following are supplementary depictions of the TCP communications sequencing for dynamic data file downloads and uploads.

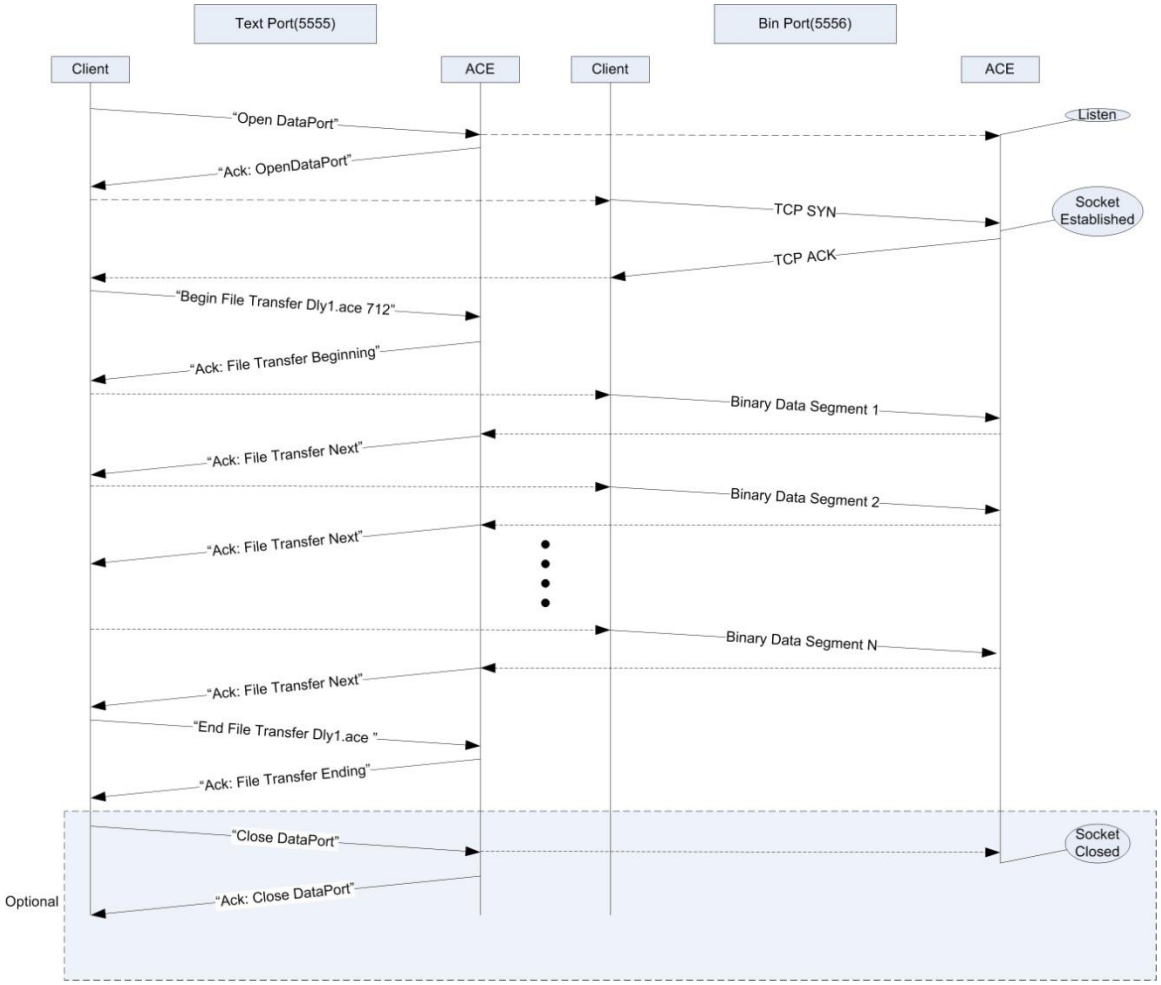

Figure 31: TCP Communications Sequence for Dynamic Data File Download

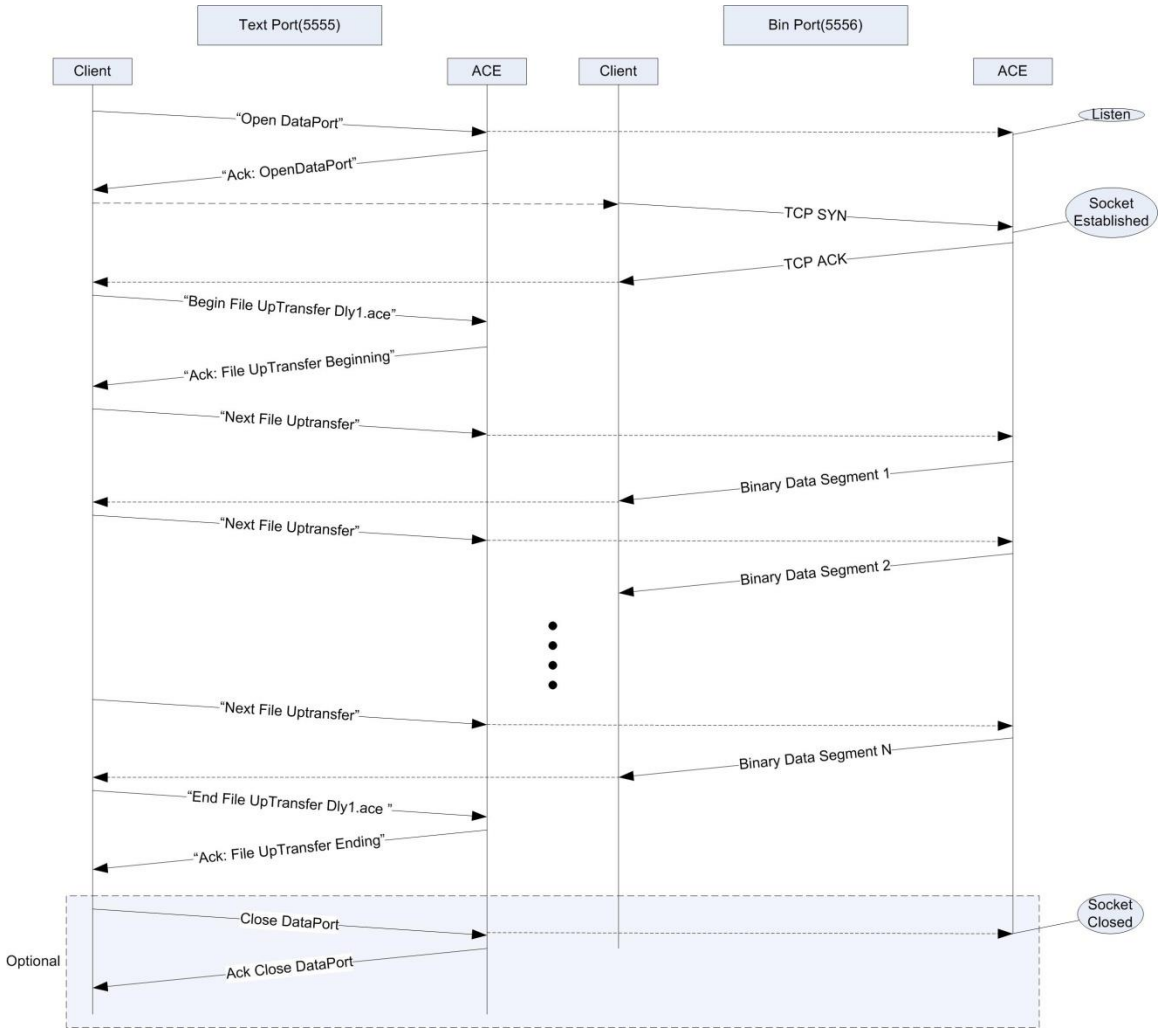

Figure 32: TCP Communications Sequence for Dynamic Data File Upload

Appendix B: Text Message Format

Send Message Format								Received Message Format							
Connect and Run Commands															
Action	Group	Channel	Parameter	Path	Fading Parameter	<value> or Choice List	Units	Ack/Nak	Group	Channel	Parameter	Path	Fading Parameter	<value> or Choice List	Units
Hello								Ack:	Hello						
Goodbye								Ack:	Goodbye						
Capture		Ch1 to Ch4, All						Ack:	Captured	Ch1 to Ch4, All					
Calculate								Ack:	Calculated						
Start								Ack:	Dynamic		State			Running, Done, Armed	
Pause								Ack:	Dynamic		State			Paused	
Reset								Ack:	Dynamic		State			Ready	
StepFwd			<setting>				msec	Ack:	Dynamic		ElapsedTime			<setting>	msec
StepBack			<setting>				msec	Ack:	Dynamic		ElapsedTime			<setting>	msec
Open	DataPort							Ack:	DataPort					Opened	
Close	DataPort							Ack:	DataPort					Closed	
Mode Commands															
Set	Mode					Static, Dynamic		Ack:	Mode					Static, Dynamic	
Get	Mode							Ack:	Mode					Static, Dynamic	
Set	Delay Slew Boundary					<setting>	msec	Ack:	Delay Slew Boundary					<setting>	msec
Get	Delay Slew Boundary						msec	Ack:	Delay Slew Boundary					<setting>	msec
Storing and Recall Commands															
Recall	Settings					0 to 9		Ack:	Settings					0 to 9	
Store	Settings					0 to 8		Ack:	Settings					0 to 8	
Factory Setting Commands															
Get	Version		SwVersion					Ack:	Version		SwVersion			<string>	
Get	Version		SerialNumber					Ack:	Version		SerialNumber			<string>	
Get	Version		ModelNumber					Ack:	Version		ModelNumber			<string>	

Get	Options	Ch1 to Ch4	FadingLicense					Ack:	Options	Ch1 to Ch4	FadingLicense			Enabled, Disabled	
Get	Options	Ch1 to Ch4	AwgnLicense					Ack:	Options	Ch1 to Ch4	AwgnLicense			Enabled, Disabled	
Get	Options	Ch1 to Ch4	RfSlave					Ack:	Options	Ch1 to Ch4	RfSlave			Enabled, Disabled	
Get	Options	Ch1 to Ch4	Installed					Ack:	Options	Ch1 to Ch4	Installed			Enabled, Disabled	
Set	Options	Ch1 to Ch4	Multipath			Enable/Disable		Ack:	Options	Ch1 to Ch4	Multipath			Enabled, Disabled	
Get	Options	Ch1 to Ch4	Multipath					Ack:	Options	Ch1 to Ch4	Multipath			Enabled, Disabled	
Set	Options	Ch1 to Ch4	CapturePoint			Before, After		Ack:	Options	Ch1 to Ch4	CapturePoint			Before, After	
Get	Options	Ch1 to Ch4	CapturePoint					Ack:	Options	Ch1 to Ch4	CapturePoint			Before, After	
Dynamic Related Commands															
Set	Dynamic		Trigger			Internal, External		Ack:	Dynamic		Trigger			Internal, External	
Set	Dynamic		Loop			Single, Continuous, FwdRev		Ack:	Dynamic		Loop			Single, Continuous	
Set	Dynamic		MultiChassisSync			StandAlone, Master, Slave		Ack:	Dynamic		MultiChassisSync			StandAlone, Master, Slave	
Set	Dynamic		Update			Internal, External		Ack:	Dynamic		Update			Internal, External	
Set	Dynamic		DelayInterval			1,2,5,10,20,50,100,200,500, 1000	msec	Ack:	Dynamic		DelayInterval			1,2,5,10,20,50,100,200,500, 1000	msec
Set	Dynamic		RfInterval			100,200,500,1000	msec	Ack:	Dynamic		RfInterval			100,200,500,1000	msec
Set	Dynamic		MpathInterval			100,200,500,1000	msec	Ack:	Dynamic		MpathInterval			100,200,500,1000	msec
Set	Dynamic		StartTime			<setting>	msec	Ack:	Dynamic		StartTime			<setting>	msec
Get	Dynamic		Trigger					Ack:	Dynamic		Trigger			Internal, External	
Get	Dynamic		Loop					Ack:	Dynamic		Loop			Single, Continuous, FwdRev	
Get	Dynamic		MultiChassisSync					Ack:	Dynamic		MultiChassisSync			StandAlone, Master, Slave	
Get	Dynamic		Update					Ack:	Dynamic		Update			Internal, External	
Get	Dynamic		DelayInterval					Ack:	Dynamic		DelayInterval			1,2,5,10,20,50,100,200,500, 1000	msec
Get	Dynamic		RfInterval					Ack:	Dynamic		RfInterval			100,200,500,1000	msec
Get	Dynamic		MpathInterval					Ack:	Dynamic		MpathInterval			100,200,500,1000	msec

Get	Dynamic		StartTime					Ack:	Dynamic		StartTime			<setting>	msec
Get	Dynamic		ElapsedTime					Ack:	Dynamic		ElapsedTime			<setting>	msec
Get	Dynamic		State					Ack:	Dynamic		State			Static, Ready, Init, Setup, Running, Paused, Done, Armed	
Begin	Dynamic		Setup					Ack:	Dynamic State Setup;						
Begin	Dynamic		Init					Ack:	Dynamic State Ready;						
File Related Commands															
Get	File		Count					Ack:	File		Count			<number>	
Delete	File					<filename>		Ack:	File					<filename> Deleted	
Get	File		SizeInBytes			<filename>		Ack:	File SizeInBytes <filename><size>						
Get	File		Info			<filename>		Ack:	File Info <filename> <type> <length> <continuous> <timestamp>						
Set	File	Ch1 to Ch4	Delay			<filename>		Ack:	File	Ch1 to Ch4	Delay			<filename>	
Set	File	Ch1 to Ch4	FreqOffset			<filename>		Ack:	File	Ch1 to Ch4	FreqOffset			<filename>	
Set	File	Ch1 to Ch4	Atten			<filename>		Ack:	File	Ch1 to Ch4	Atten			<filename>	
Set	File	Ch1 to Ch4	PhaseOffset			<filename>		Ack:	File	Ch1 to Ch4	PhaseOffset			<filename>	
Set	File	Ch1 to Ch4	RfIn			<filename>		Ack:	File	Ch1 to Ch4	RfIn			<filename>	
Set	File	Ch1 to Ch4	RfOut			<filename>		Ack:	File	Ch1 to Ch4	RfOut			<filename>	
Set	File	Ch1 to Ch4	NoiseNo			<filename>		Ack:	File	Ch1 to Ch4	NoiseNo			<filename>	
Set	File	Ch1 to Ch4	Mpf			<filename>		Ack:	File	Ch1 to Ch4	Mpf			<filename>	
Get	File	Ch1 to Ch4	Delay					Ack:	File	Ch1 to Ch4	Delay			<filename>	

Get	File	Ch1 to Ch4	FreqOffset					Ack:	File	Ch1 to Ch4	FreqOffset			<filename>
Get	File	Ch1 to Ch4	Atten					Ack:	File	Ch1 to Ch4	Atten			<filename>
Get	File	Ch1 to Ch4	PhaseOffset					Ack:	File	Ch1 to Ch4	PhaseOffset			<filename>
Get	File	Ch1 to Ch4	RfIn					Ack:	File	Ch1 to Ch4	RfIn			<filename>
Get	File	Ch1 to Ch4	RfOut					Ack:	File	Ch1 to Ch4	RfOut			<filename>
Get	File	Ch1 to Ch4	NoiseNo					Ack:	File	Ch1 to Ch4	NoiseNo			<filename>
Get	File	Ch1 to Ch4	Mpf					Ack:	File	Ch1 to Ch4	Mpf			<filename>
Channel Parameter Related Commands														
Set	Param	Ch1 to Ch4	Delay			<setting>	msec	Ack:	Param	Ch1 to Ch4	Delay			<setting>	msec
Set	Param	Ch1 to Ch4	FreqOffset			<setting>	kHz	Ack:	Param	Ch1 to Ch4	FreqOffset			<setting>	kHz
Set	Param	Ch1 to Ch4	Atten			<setting>	dB	Ack:	Param	Ch1 to Ch4	Atten			<setting>	dB
Set	Param	Ch1 to Ch4	PhaseOffset			<setting>	degrees	Ack:	Param	Ch1 to Ch4	PhaseOffset			<setting>	deg
Set	Param	Ch1 to Ch4	RfIn			<setting>	MHz	Ack:	Param	Ch1 to Ch4	RfIn			<setting>	MHz
Set	Param	Ch1 to Ch4	RfOut			<setting>	MHz	Ack:	Param	Ch1 to Ch4	RfOut			<setting>	MHz
Set	Param	Ch1 to Ch4	NoiseNo			<setting>	dBm	Ack:	Param	Ch1 to Ch4	NoiseNo			<setting>	dBm
Set	Param	Ch1 to Ch4	NoiseBR			<setting>	MHz	Ack:	Param	Ch1 to Ch4	NoiseBR			<setting>	MHz

Set	Param	Ch1 to Ch4	NoiseEbNo			<setting>	dB	Ack:	Param	Ch1 to Ch4	NoiseEbNo			<setting>	dB
Set	Param	Ch1 to Ch4	NoiseMode			No, EbNo		Ack:	Param	Ch1 to Ch4	NoiseMode			No, EbNo	
Set	Param	Ch1 to Ch4	NoiseEnable			On, Off		Ack:	Param	Ch1 to Ch4	NoiseEnable			On, Off	
Set	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Delay	<setting>	usec	Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Delay	<setting>	usec
Set	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Doppler	<setting>	Hz	Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Doppler	<setting>	Hz
Set	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Loss	<setting>	dB	Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Loss	<setting>	dB
Set	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Type	Off, Cw, Ray, Rice		Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Type	Off, Cw, Ray, Rice	
Set	Param	Ch1 to Ch4	Mpf	Path1 to Path12	CorrPath	Path1 to Path12		Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	CorrPath	Path1 to Path12	
Set	Param	Ch1 to Ch4	Mpf	Path1 to Path12	CorrVal	<setting>	%	Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	CorrVal	<setting>	%
Set	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Kfactor	<setting>	dB	Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Kfactor	<setting>	dB
Set	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Ratio	<setting>	Hz	Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Ratio	<setting>	Hz
Set	Param	Ch1 to Ch4	Mpf	Path1 to Path12	StdDev	<setting>	dB	Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	StdDev	<setting>	dB
Set	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Aoa	<setting>	degrees	Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Aoa	<setting>	deg

Get	Param	Ch1 to Ch4	Delay					Ack:	Param	Ch1 to Ch4	Delay			<setting>	msec
Get	Param	Ch1 to Ch4	FreqOffset					Ack:	Param	Ch1 to Ch4	FreqOffset			<setting>	kHz
Get	Param	Ch1 to Ch4	Atten					Ack:	Param	Ch1 to Ch4	Atten			<setting>	dB
Get	Param	Ch1 to Ch4	PhaseOffset					Ack:	Param	Ch1 to Ch4	PhaseOffset			<setting>	deg
Get	Param	Ch1 to Ch4	RfIn					Ack:	Param	Ch1 to Ch4	RfIn			<setting>	MHz
Get	Param	Ch1 to Ch4	RfOut					Ack:	Param	Ch1 to Ch4	RfOut			<setting>	MHz
Get	Param	Ch1 to Ch4	NoiseNo					Ack:	Param	Ch1 to Ch4	NoiseNo			<setting>	dBm
Get	Param	Ch1 to Ch4	NoiseBR					Ack:	Param	Ch1 to Ch4	NoiseBR			<setting>	MHz
Get	Param	Ch1 to Ch4	NoiseEbNo					Ack:	Param	Ch1 to Ch4	NoiseEbNo			<setting>	dB
Get	Param	Ch1 to Ch4	NoiseMode					Ack:	Param	Ch1 to Ch4	NoiseMode			No, EbNo	
Get	Param	Ch1 to Ch4	NoiseEnable					Ack:	Param	Ch1 to Ch4	NoiseEnable			On, Off	
Get	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Delay			Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Delay	<setting>	usec
Get	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Doppler			Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Doppler	<setting>	Hz
Get	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Loss			Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Loss	<setting>	dB

Get	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Type			Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Type	Off, Cw, Ray, Rice	
Get	Param	Ch1 to Ch4	Mpf	Path1 to Path12	CorrPath			Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	CorrPath	Path1 to Path12	
Get	Param	Ch1 to Ch4	Mpf	Path1 to Path12	CorrVal			Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	CorrVal	<setting>	%
Get	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Kfactor			Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Kfactor	<setting>	dB
Get	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Ratio			Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Ratio	<setting>	Hz
Get	Param	Ch1 to Ch4	Mpf	Path1 to Path12	StdDev			Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	StdDev	<setting>	dB
Get	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Aoa			Ack:	Param	Ch1 to Ch4	Mpf	Path1 to Path12	Aoa	<setting>	deg
Get	Param	Ch1 to Ch4	SigPower					Ack:	Param	Ch1 to Ch4	SigPower			<setting> InRange, Overflow	dBm
Get	Param	Ch1 to Ch4	CalcStatus					Ack:	Param	Ch1 to Ch4	CalcStatus			Calculated, NotCalculated	
Get	Param	Ch1 to Ch4	RangeStatus					Ack:	Param	Ch1 to Ch4	RangeStatus			InRange, OutOfRange	
Downloading a Binary Dynamic Data File															
Begin	File		Transfer			<filename> <size>		Ack:	File		Transfer			Beginning	
Binary								Ack:	File		Transfer			Next	
End	File		Transfer			<filename>		Ack:	File		Transfer			Ending	
Abort	File		Transfer			<filename>		Ack:	File		Transfer			Aborted	
Uploading a Binary Dynamic Data File															
Begin	File		UpTransfer			<filename>		Ack:	File		Transfer			Beginning	

Next	File		UpTransfer			<filename>		Nak: or	Binary						
End	File		UpTransfer			<filename>		Ack:	File		Transfer			Ending	
Abort	File		UpTransfer			<filename>		Ack:	File		Transfer			Aborted	
Uploading a Captured Data from Channel															
Begin	Capture		UpTransfer			Ch1 to Ch4		Ack:	Capture		UpTransfer			Beginning	
Next	Capture		UpTransfer					Nak: or	Binary						
End	Capture		UpTransfer			Ch1 to Ch4		Ack:	Capture		UpTransfer			Ending	
Abort	Capture		UpTransfer			Ch1 to Ch4		Ack:	Capture		UpTransfer			Aborted	
Uploading Dynamic File Directory List															
Begin	File		DirItem					Nak: or	File		DirItem			Next	
End	File		Directory					Ack:	File		Directory			Ending	
Next	File		DirItem					Nak: or	File		DirItem			Filename	