ThinkRF WSA5000

Wireless Signal Analyzer
Version 3.3.2
Programmer's Guide

October 15, 2014
Document no. 75-0014-140915

Important notice

Trademarks

ThinkRF Corp

390 March Road
Kanata, ON K2K 0G7
(613) 369-5104

HARDWARE WARRANTY AND LIMITATION OF LIABILITY

Read this warranty carefully before you use the product.

WSA5000 Wireless Signal Analyzers are warranted for workmanship and materials for a period of one (1) year from the date of shipment as identified by the Customer's packing slip or carrier waybill. ThinkRF reserves the right to void the warranty on any equipment that has been altered or damaged due to Customer negligence, unauthorized repair, misuse of equipment, evidence of physical or environmental damage, transportation abuse or removal of any ThinkRF identification labels or serial numbers.

It will remain the responsibility of the Customer, having obtained a Return Material Authorization (RMA) and shipping instructions from ThinkRF, to return, at the Customer's expense, the defective unit to ThinkRF's repair facilities. ThinkRF will incur shipping charges for the return of warranty repaired equipment. The RMA number can be secured by calling ThinkRF Customer Service and Support (1-613-369-5104). If the product does not fall within ThinkRF's warranty period or the product is found to be functioning as designed, then under the terms of ThinkRF's warranty policy, all costs of repairs and shipping will be charged directly to the Customer. ThinkRF will warrant repaired units for a period of 90 days from date of shipment from ThinkRF to the Customer. If the remaining period on the original hardware warranty is greater than 30 days, then ThinkRF will honor this remaining warranty period.

THINKRF EXPRESSLY DISCLAIMS ALL OTHER WARRANTIES AND CONDITIONS, WHETHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, WARRANTIES, CONDITIONS OR REPRESENTATIONS OF WORKMANSHIP, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, DURABILITY, OR THAT THE OPERATION OF THE HARDWARE OR LICENSED SOFTWARE WILL BE ERROR FREE. IN NO EVENT WILL THINKRF BE LIABLE FOR INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES.

USE OF PRODUCTS IN HIGH RISK ACTIVITIES

THINKRF PRODUCTS ARE INTENDED FOR STANDARD INDOOR COMMERCIAL USE. WITHOUT THE APPROPRIATE NETWORK DESIGN ENGINEERING, THEY MUST NOT BE USED FOR ANY "HIGH RISK ACTIVITY", as described in this paragraph. Customer acknowledges and agrees that the products supplied hereunder are not fault-tolerant and are not designed, manufactured or intended for use or resale as on-line control equipment in hazardous environments requiring fail safe performance including but not limited to the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines, or weapons systems, in which the failure of products could lead directly to death, personal injury, or severe physical or environmental damage, all of which are examples of "High Risk Activity". THINKRF AND ITS SUPPLIERS EXPRESSLY DISCLAIM ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS FOR HIGH RISK ACTIVITIES.

GNU General Public License

Table of Contents

Abbreviations

ADC	Analog-to-Digital Converter
API	Application Programming Interface
CIC	Cascaded Integrator-Comb
DC	Direct Current
DD	Direct Digitizer
DDC	Digital Down Converter
DDS	Direct Digital Synthesizer
DSP	Digital Signal Processing
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
FPGA	Field-Programmable Gate Array
GPIO	General Purpose Input/Ouput
HDR	High Dynamic Range
IBW	Instantaneous Bandwidth
IEEE	Institute of Electrical and Electronics Engineers
IF	Intermediate Frequency
IQ	In-phase and Quadrature
IQIN	External I and Q Input
LAN	Local Area Network
MB	Mega-Bytes
MSB	Most Significant Byte
NB	Narrowband
NCO	Numerically Controlled Oscillator
NSH	Narrow Super-Het
PLL	Phase-Locked Loop
PSD	Power Spectrum Density
RF	Radio Frequency
RFE	Receiver Front-End
Sa/s	Samples-per-Second
SCPI	Standard Commands for Programmable Instruments
SH	Super-Heterodyne
TCP/IP	Transmission Control Protocol/Internet Protocol
TD	Time Domain
TSF	TimeStamp-Fractional
TSI	TimeStamp-Integer
TSM	TimeStamp Mode
UTC	Coordinated Universal Time
VCO	Voltage Control Oscillator
VRT	VITA-49 Radio Transport
WB	Wideband
WSA	Wireless Signal Analyzer
ZIF	Zero Intermediate Frequency

List of Figures

List of Tables

Preface

This preface describes the audience for, the organization of, and conventions used in this document. It also identifies related documentation and explains how to access electronic documentation.

Audience

This document is written for software developers wishing to develop and/or maintain a software interface to the WSA5000 and who have a basic understanding, familiarity and experience with network test and measurement equipment.

Conventions

This section describes the conventions used in this document.

Grayed-out Font
Indicates a command or a feature is not yet available in the current release.

`Courier Font`
Illustrates this is an example for a command or a concept.

Light Blue Font
Contains hyperlink that can be clicked on to go to the source that is being referenced.

Normal Bold Font
When used within a sentence or a paragraph, it emphasizes an idea to be paid attention to particularly.

Red Font
Conveys special information of that section.

Note: This symbol means **take note**. Notes contain helpful suggestions or references to additional information and material.

Caution: This symbol means **be careful**. In this situation, you might do something that could result in equipment damage or loss of data.

Warning: This symbol means **danger**. You are in a situation that could cause bodily injury. Before you work on any equipment, be aware of the hazards involved with electrical circuitry and be familiar with the standard practices for preventing accidents.

Obtaining Documentation and Releases

You can access the most current ThinkRF documentation at http://www.thinkrf.com/resources and release bundles at http://www.thinkrf.com/releases.

Document Feedback

Please send your comments about this document or our other documentation to support@thinkrf.com.

Thank you, we appreciate your comments.

Obtaining Technical Assistance

The ThinkRF Support website provides online documents for resolving technical issues with ThinkRF products at this URL: http://thinkrf.com/documentation/.

For all customers who hold a valid end-user license, ThinkRF provides technical assistance 9 AM to 5 PM Eastern Time, Monday to Friday. Contact us at **support@thinkrf.com** or by calling **+1.613.369.5104**.

Before contacting Support, please have the following information available:

- WSA5000's serial number. The serial number S/N is located on the identification label on the WSA5000's underside;
- version of ThinkRF firmware you are using, potentially including version of PyRF and/or API libraries to third-party applications; and
- the operating system you are using.

WSA5000 Functional Overview

This section overviews the WSA5000's functionality and protocols used, and summarizes the SCPI command sets for controlling the individual functions.

Note: This is a living and evolving document. We welcome your feedback.

The features and functionality described in this section **may** exist in the current product firmware release or are scheduled for a future product firmware release (grayed out commands and/or text). Please refer to Appendix F: SCPI Commands Quick Reference for the complete list of commands and the availability information. No hardware upgrade is required at each feature release (unless specified though unlikely).

System Overview

The WSA5000 Wireless Signal Analyzer is a high-performance software-defined RF receiver, digitizer and analyzer, as illustrated in Figure 1. With patent-pending software-defined RF receiver technology, the WSA provides industry leading combined sensitivity, tuning range, instantaneous bandwidth (IBW) and scan rate. Additionally, it provides real-time sophisticated triggering and capture control.

The WSA5000 is designed for stand-alone, remote and/or distributed wireless signal analysis. It is ideal for monitoring, management and surveillance of transmitters, whether they are in-building or spread across a geographic area. Applications include, but are not limited to:

- spectrum analysis, wireless network management and interference mitigation;
- cognitive radio and white space spectrum sensing, enterprise wireless signal intrusion detection (WSID);
- government spectrum licensing monitoring and enforcement;
- technical security counter measures (TSCM) and military communications and signals intelligence (COMINT/SIGINT and CEW).

The WSA5000 hardware largely consists of:

- a hybrid super-heterodyne and direct-conversion RF receiver front-end (RFE);
- receiver front end inputs and outputs to support clock synchronization, direct digitization input, and IF output for high-end digitization;
- a 125 MSample/sec 12-bit (or 14-bit as a population variant) wideband (WB) ADC with a dynamic range of about 70 dB;
- a 300 kSample/sec 24-bit narrowband (NB) ADC with a dynamic range in excess of 100 dB;
- a large Xilinx FPGA with embedded MicroBlaze microprocessor, Gigabit Ethernet interface and custom embedded digital signal processing (DSP) logic;
- 128 or 256 MB of DDR3 for real-time caching of digitized data; and
- a general purpose input/output (GPIO) port.

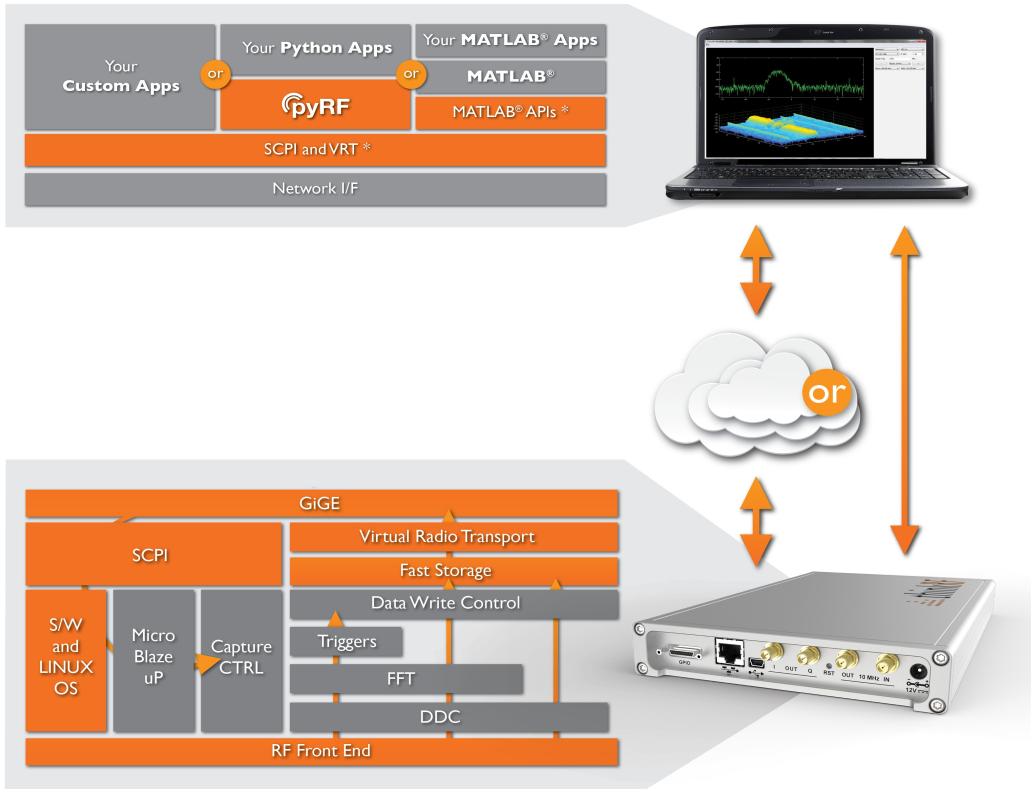


Figure 1: WSA5000 Functional Block Diagram

ThinkRF's products conform with standardized protocols for interoperability. ThinkRF provides application programming interfaces (APIs) designed for easy integration with third-party applications. Standard protocols include the Standard Commands for Programmable Instruments (SCPI) protocol for controlling and obtaining status from the WSA and the VITA-49 Radio Transport (VRT) protocol for digitized data and its associated context information.

In addition, API libraries, written in C/C++, Python and MATLAB, are provided for quick interfacing, data acquisition and as well as for spectral analysis with MATLAB® applications. The Python API is built within the PyRF development framework and is open-source under BSD licensing. PyRF handles the low-level details of real-time acquisition, signal processing and visualization, and provides feature rich libraries, example applications and source code, all specific to the requirements of signal analysis. Usage examples are provided through the available source codes of the Graphical User Interfaces (GUI) or any applications included in each release package.

Refer to Appendix A for how to connect to a WSA and Appendix B for the protocol on how to find any WSAs available on the local network. The source code provided for the aforementioned APIs and GUIs/applications would serve as examples.

The WSA5000 provides system level control and status commands as defined in Table 1.

Table 1: System Level Control/Status Commands

SCPI Command	Description
:SYSTem	
:VERSion?	Returns the SCPI version number that the instrument complies with
:CAPability?	Returns a list of the WSA5000's capabilities including firmware versions and installed hardware options
:OPTions?	Returns comma separated 3-digit values to represent the hardware option(s) or features available with a particular WSA model
:ABORt	Aborts the current data capturing process and puts the WSA system into a normal manual mode (i.e. sweep, trigger, and streaming will be aborted)
:FLUSh	Clears the WSA5000's internal data storage buffer of any remaining data that has not transferred out of the WSA
:CAPTure	
:MODE?	Gets the current capture mode of the WSA (i.e. sweeping, streaming or block mode)
:LOCK	
:REQuest?	Requests the WSA5000 to provide a lock on a specific task such that only the application that has the lock can perform the task
:HAVE?	Returns the current lock state of the task specified
:SYNC	
:MASTer[?]	Sets a WSA unit to be the master or slave for a synchronization trigger system with multiple units. Affects :TRIGger:TYPE PULSe or WORD.
:WAIT[?]	Sets the delay time in nanoseconds that the system must wait after receiving the trigger signal before performing data capture
:DATE[?]	Sets/reads date
:TIME[?]	Sets/reads time
:ADJust	Adjust the system time relative to it's current time
:SYNC[?]	Sets/ gets the System time synchronization source via network or SCPI, or disable
:MODE[?]	Synchronize one time only or continuously
:STATus?	Status of the time synchronization
:COMMunicate	
:LAN<commands>	Subset of commands for configuring/querying WSA's LAN settings
:ERRor	Returns the error code and messages from the SCPI error/event queue
[:NEXT]?	
:ALL?	
:STATus	
:OPERation	Returns the standard Operation Status Register (OSR) for any event
[:EVENt]?	
:CONDition?	
:ENABle[?]	
:PRESET	Presets the WSA5000 (similar to *RST)
:QUEStionable	Returns the standard Questionable Status Register (QSR) for any event
[:EVENt]?	
:CONDition?	

SCPI Command	Description
:ENABle[?]	
:TEMPerature?	Returns the WSA5000's internal ambient temperature

See SCPI Command Set section (page 44 onward) for further details on the commands.

Caution pertaining to multi-user: The current firmware version of the WSA5000 allows multiple applications to connect to the unit simultaneously but it does not support independent sessions. Therefore, the actions of one user may over-write those of another. This could potentially damage the unit for instance if the front-end's gain were incorrectly set. If multiple applications are connecting to the unit, it is advised that only one of those is controlling the unit at any time.

The Architecture

The WSA5000 is an integrated wireless radio receiver and digitizer/analyzer. It has an embedded capture controller that enables users to:

- define and execute real-time and sophisticated triggers, traces and sweeps;
- configure the radio RFE and DSP in association with those traces or sweeps; and
- time-stamping and data output for captures.

Traces and sweeps are controlled by the capture controller as illustrated in the lower portion of Figure 2. A trace and a sweep are defined as a single (block or continuously streamed) capture and a series of captures, respectively, each with their associated hardware configurations.

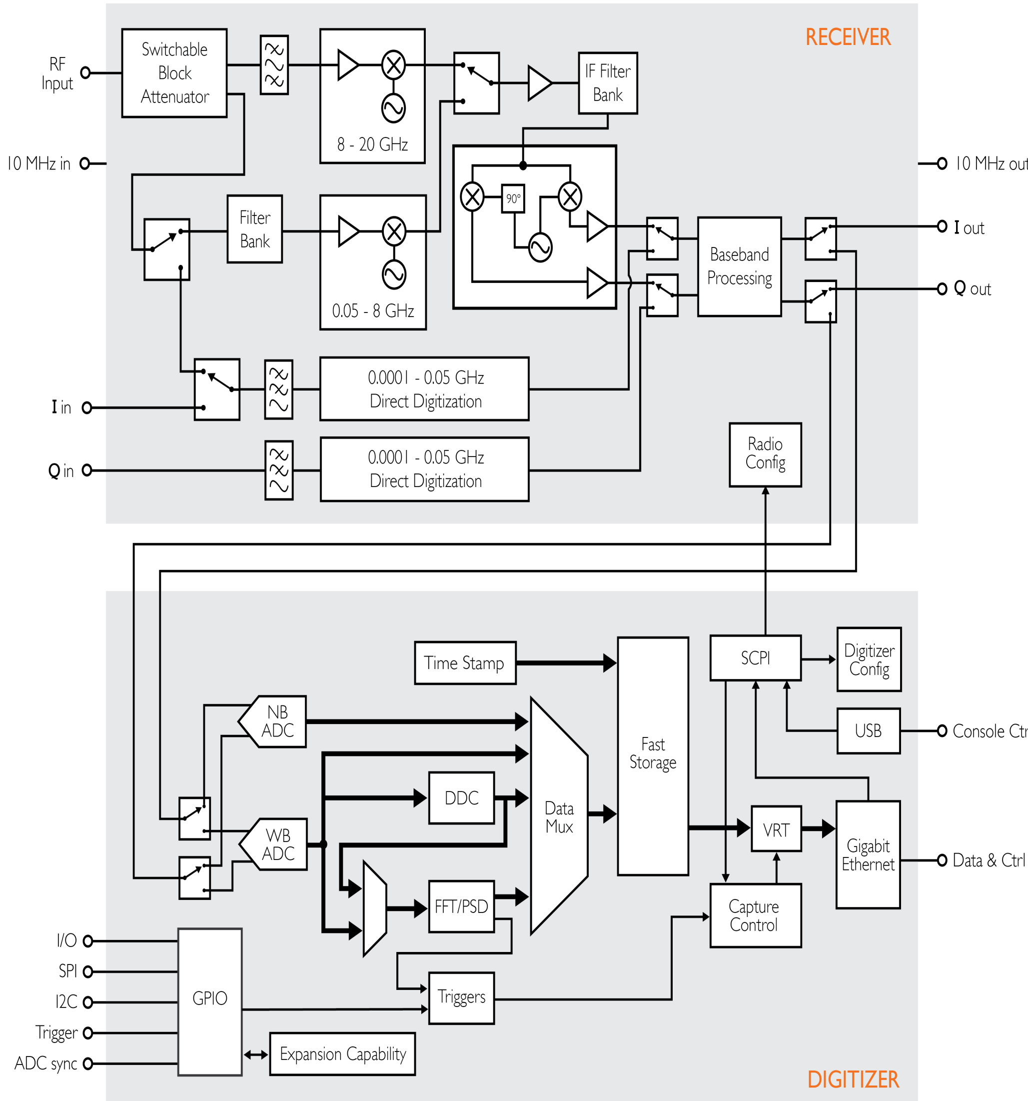


Figure 2: RF Receiver Front-end and Capture Controller Functional Block Diagram

The WSA5000 supports different RFE modes of operation and subsequent DSP capabilities as per Table 2 and as described in the following subsections.

Table 2: Radio RFE Modes and DSP Data Output Formats

Mode[0]	Description	Freq Range (MHz)	IBW (MHz)	DSP Data Output Format[1]			
				None	CIC/Dec	Frequency Shift	logPSD
ZIF	Zero-IF Receiver	50 - max	100	I_{14} Q_{14}	I_{14} Q_{14}	I_{14} Q_{14}	PSD_8
SH	Super-Heterodyne Receiver	50 - max	40[2]	I_{14}	I_{14} Q_{14}[4]	I_{14} Q_{14}	PSD_8
SHN[3]	SH Receiver with narrower BW	50 - max	10	I_{14}	I_{14} Q_{14}[4]	I_{14} Q_{14}	PSD_8
HDR	High Dynamic Range Receiver	50 - max	0.1	I_{24}	-	-	-
DD	Direct Digitization Receiver	0.1 - 50	50	I_{14}	I_{14} Q_{14}[4]	I_{14} Q_{14}	PSD_8
IQIN	External IQ Input	0.1 - 50	100	I_{14} Q_{14}	I_{14} Q_{14}	I_{14} Q_{14}	PSD_8

[0] The RFE Mode availability is product dependent. WSA5000-108, for instance, does not have ZIF, SH and IQIN mode as this model does not handle wideband ADC.

[1] The WSA5000 supports a 12-bit or 14-bit WB ADC as a manufacturing population variant. The least significant bits of the 14-bit data representations are zeroed when the WSA is populated with the 12-bit WB ADC, hence, the subscript of 14 for IQ or I.

[2] The 40 MHz SH is only available in WSA5000 hardware revision 3. Revision 2 hardware value varies between 30 or 35 MHz, contact ThinkRF's Support for further details. See *IDN? to find out your hardware version (or the Administrative web-console to the box).

[3] SHN mode is only available in WSA5000 hardware revision 3. See *IDN? to find out your hardware version (or the Administrative web-console to the box).

[4] For SH and SHN modes, when the decimation is used has to be used, a frequency shift of 35 MHz will be applied automatically to bring the WSA5000's center frequency back to the zero IF. Thus, the data output will be I and Q.

WSA5000 complies to VRT protocol for sending digitized IF data packets and their associated context information depending on the capture mode. It is very important to follow the VRT's IF Data Packet Class section (page 37) for the exact VRT data output formats as well as packing method.

RF Receiver Front-End

The upper portion of Figure 2 shows a block diagram of the RFE within the WSA5000. The architecture consists of a super-heterodyne (SH) front-end with a back-end that utilizes an I/Q mixer similar to that in a direct-conversion (or zero-IF) receiver.

Depending on the frequency of the signals being analyzed, one of the three receiver signal processing paths is selected. Signals in the frequency range 100 kHz to 50 MHz are directly digitized, while all other signals are translated to the frequencies of the first IF block via one of the other two signal processing paths. The IF block consists of a bank of multiple SAW filters. SAW filter selection depends on the frequency of the input signal. The output of the SAW filter feeds the I/Q mixer.

The three signal processing paths are further classified into different modes of operation for the capture engine as shown in Table 2. The radio modes ZIF, SH, SHN and HDR support tuning the center frequency from 50 MHz to the maximum frequency supported by the particular product variant (8 GHz for the WSA5000-108 and -208, and 20 GHz for the WSA5000-220).

The ZIF, SH and SHN radio modes support a tuning resolution of 100 kHz. Digital frequency shifting can then be used to enhance the tuning resolution to the nearest 1 Hz (±0.23 Hz) The frequency shifting technology used is an embedded Numerically Controlled Oscillator (NCO) (a Direct Digital Synthesizer or DDS) with as described in the Digital Down Converter subsection (page 21).

The HDR radio mode supports a tuning resolution of 100 kHz. No further fine tuning is available.

The remaining two radio modes (IQIN and DD) support 50 MHz IBW direct digitization of the baseband from the external RF IN or I and Q IN ports. Hence, neither of these modes support frequency tuning of the radio although the DSP's frequency shift mode may be applied.

Direct-Conversion Receiver Technology

Direct-conversion (or ZIF) receivers are ideal for signal analysis of wideband waveforms, such as 4G/LTE, Wi-Fi and Bluetooth. With that benefit comes the drawback of both IQ and DC offsets which are inherent to direct-conversion technology.

DC Offset Correction

The WSA5000's WB ADC sampling rate is 125 MSa/s, intermediate frequency (IF) is 0 and the entire IF bandwidth is 125 MHz. The analog filter results in an amplitude roll-off at approximately ±50 MHz around the center frequency Fc, as illustrated in Figure 3.

Direct-conversion receivers have a DC offset at the center of the band. The offset is primarily compensated for in real-time in the receiver hardware but there always is some residual offset that (depending on the application and bandwidth of interest) might need to be compensated for in software. Several options such as calibration or dynamic offset compensation in software have been described in the open literature.

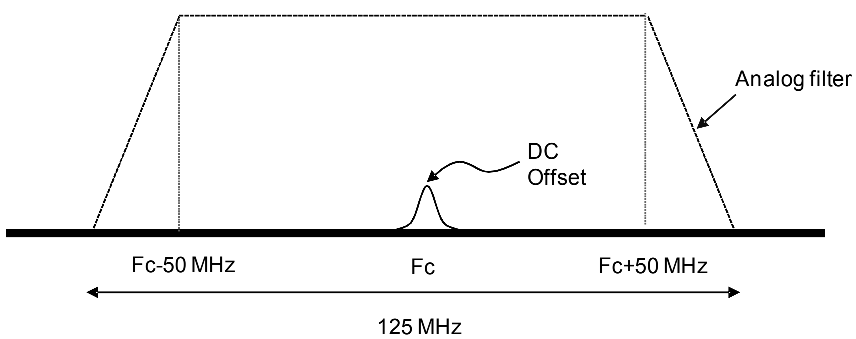


Figure 3: DC Offset with Amplitude Roll-Off at ±50 MHz

If the application only needs to utilize up to 50 MHz of IBW, a simple alternative to DC offset compensation is to use the SH mode of operation.

IQ Offset Correction

Direct-conversion receivers have phase and/or amplitude offsets between in-phase (I) and quadrature (Q) components of the baseband signal. Due to this, when an FFT is performed on digitized baseband data where there is a signal tone present, there will be an 'image' at the same frequency offset from the center frequency as the tone itself. This is illustrated in Figure 4.

To compensate for this, the raw I and Q data must be processed according to the following "calibrateIQ" routine, illustrated using the following MATLAB® code. When an FFT is performed on the output of calibrateIQ, the image will disappear. This process has no impact on the accuracy or precision of the data.

```
%%%%%
%  MATLAB code for IQ Offset Correction
%%%%%

function [calibratedQ] = calibrateIQ(iData, qData)
  numberOfSamples = size(iData, 1);
  sumOfSquaresI = sum(iData.^2);
  sumOfSquaresQ = sum(qData.^2);
  amplitude = sqrt(sumOfSquaresI * 2 / numberOfSamples);
  ratio = sqrt(sumOfSquaresI / sumOfSquaresQ);
  p = (qData/amplitude) * ratio .* (iData/amplitude);
  sinphi = 2 * sum(p) / numberOfSamples;
  phi_est = -asin(sinphi);

  calibratedQ = ((sin(phi_est) * iData) + (ratio * qData)) / cos(phi_est);
end
```

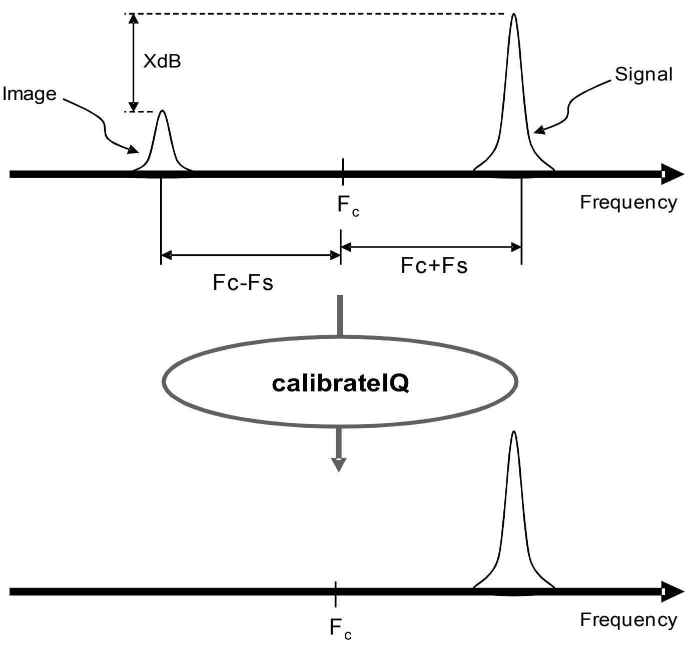


Figure 4: IQ Offset Correction

Table 3: RF Front-End Control/Status Commands

SCPI Command	Description
:CALibrate	
:RUN	Starts a calibration of the WSA5000 system
:STATus?	

SCPI Command	Description
:ABORt	
:INPut	
:MODE[?]	Selects the receiver mode of operation
:ATTenuator[?]	Enables/disables the front-end's 20 dB attenuation
:FILTer	
:PRESelect[?]	Enables/disables the use of preselect filtering
:GAIN	
:IF[?]	Sets the variable IF gain
:HDR[?]	Sets gain level for the NB ADC of of the HDR signal path
:SOURce	
:REFerence	
:PLL[?]	Selects the 10 MHz reference clock source
:RESET	Resets the 10 MHz reference selection to INTernal source
:CLOCk	
:ADC[?]	Selects variable or fix input clock type to the WB ADC
:RATE[?]	Sets the clock rate for the WB ADC
[:SENSe]	
:CORRection	
:DCOFfset[?]	Enables/disables the ADC's DC-offset correction system
:FREQuency	
:CENTer[?]	Sets the center frequency of the RFE
:LOSCillator?	Gets the frequency to be set for the external LO 1 or 2 in corresponding to current the WSA's center frequency
:SHIFt[?]	Sets the frequency shift value (not available for HDR mode)
:RESolution?	Gets the Analog PLL tuning resolution
:INVersion?	Queries if a spectral inversion is required at a given frequency
:DECimation[?]	Sets the decimation rate as an exponent of 2 (i.e. rate = 2^{level} where level = 0, 1, 2 - 10)
:LOCK	
:REFerence?	Queries the lock status of the PLL reference clock
:RF?	Queries the lock status of the RFE's PLL
:OUTput	
:IQ	
:MODE[?}	Selects the IQ output path to be from the external connector or the digitizer

See SCPI Command Set section (page 44 onward) for further details on each set of commands.

Digital Signal Processing

The WSA5000 has embedded DSP blocks to provide further signal processing capabilities, such DDC with up to 10 levels of decimation, FFT computation and logarithmic PSD output.

Digital Down Converter

The DDC block takes the frequency band of interest and shifts it down in frequency, then provides decimation of the sampling rate to one that is lower and consistent with the bandwidth of the signal of interest. This enables channelization of signals having bandwidth smaller than the IBW.

Referring to Figure 5, the DDC has two major elements, an NCO (DDS) and a down sampling with filtering. The NCO generates a complex sinusoid, which is mixed with the IQ input using a complex multiplier, to shift or offset the signal spectrum from the selected carrier frequency. This process provides the frequency fine-tuning (and shifting) feature as mentioned in the previous subsections.

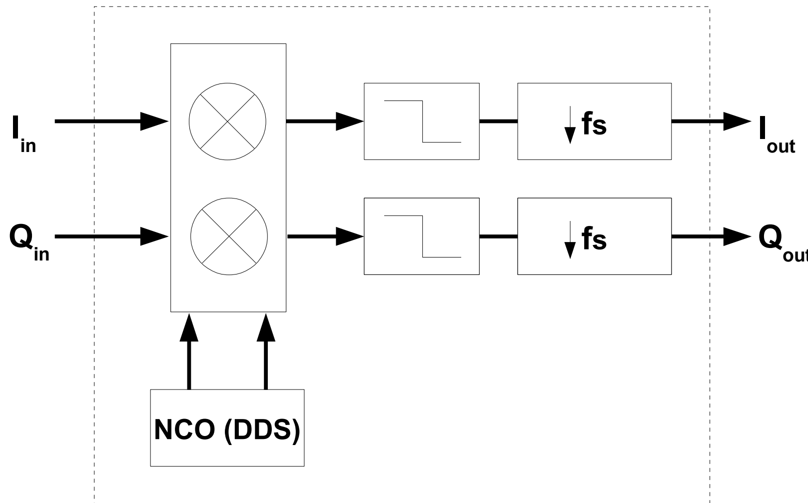


Figure 5: DDC Functional Block Diagram

The complex multiplication is then followed by either a finite impulse response (FIR) filter or cascaded integrator-comb (CIC) filters with a FIR filter combined. The CIC filter has a 'droop' associated with it in the passband. In order to compensate for this droop, the CIC filter is followed by a compensating FIR filter. Each filter type has its own decimator. This whole process effectively reduces the sample rate and filters the signal to remove adjacent channels, minimize aliasing, and maximize the received signal-to-noise ratio.

Note: The use of the NCO converts the real signal output (I data) of the receiver's DD, SH and SHN processing paths to complex I and Q data output. See Table 2.

Power Spectrum Density

Lastly, the resulting signal from the ZIF, SH, SHN, DD and IQIN radio process paths may be digitally signal processed to obtain the signal's power spectral density. The time domain signal is first transformed to the frequency domain by an an embedded fast Fourier transform (FFT) engine. The resulting IQ components are then converted to PSD data in logarithmic form using the following equation:

$$logPSD = 10 * \log_{10}(I_{fft}^2 + Q_{fft}^2)$$

The logarithm of PSD provides a mean of offloading the calculation to the WSA5000's embedded real-time hardware while reducing the amount of data required to represent the signal of interest.

Note: While providing higher data throughput rate, the tradeoff of logPSD output is that there is no IQ offset correction (refers to page 18) being applied to the IQ data within the embedded hardware. The signal 'image', therefore, might present in the PSD_8 data, particularly for the ZIF receiver mode.

Commands related to this feature are not available in this release.

Triggers

Triggers provide a means of qualifying the storage of captured time domain IQ data based on an external, periodic or frequency domain event. Triggering can be considered a means of filtering signals of interest for the purposes of subsequent visualization and/or analysis.

The following describes the different types of triggers and their common controls. Selection of different types is mutually exclusive.

Frequency Domain Triggering

Frequency domain triggering relies on the embedded real-time FFT mechanism to transform the sampled signal from the time domain to the frequency domain. The WSA5000 uses a 1024 point real-time FFT core embedded within the FPGA to transform 1024 time domain IQ samples to 1024 frequency domain FFT bins. Each bin is an average of the spectral activity over a range of 125 MHz divided by the DDC decimation rate divided by the 1024 FFT points.

The frequency domain triggering supported by WSA5000 is a level trigger type, used to capture any signal above the noise floor within a specified frequency range. The user defines a single amplitude level within a frequency range. The frequency range encompasses all FFT bins with center frequencies within the range defined by START and STOP. If the sampled signal amplitude exceeds the defined trigger level at any single sample within the defined frequency range, the trigger will occur and the subsequent IQ data capture will proceed.

Figure 6 illustrates the association of the time domain and the frequency domain. The internal frequency domain data lags the time domain data by 1024 samples at the rate of 125 MSa/s. After a trigger event is detected, the subsequent time domain IQ data is then stored to memory.

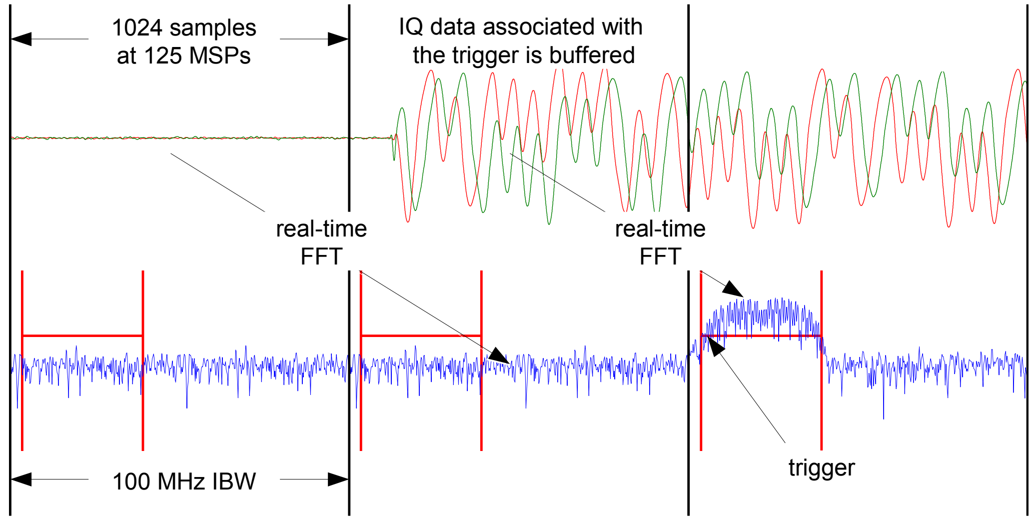


Figure 6: Association between Time and Frequency Domain

The measurable range of the input signal, and the corresponding allowable trigger level range, varies depending on the selected center frequency, the calibrated reference level and the attenuation setting. When the attenuation is in the circuit (:INPut:ATTenuator ON), the maximum trigger level to use is -10 dBm; and when the attenuation is out (:INPut:ATTenuator OFF), the maximum is -30 dBm. The threshold error is approximately -3 dBm or less when the trigger level is 20 dBm above the noisefloor. When the level is within 20 dBm of the noisefloor, the threshold error increases as the signal gets closer to the noisefloor.

Note: The threshold error is relative to the measured input signal level, which is dependent on the calibrated reference level. The reference levels could be custom calibrated to best fit a user's application, see the "AppNote – How to calibrate WSA5000" document listed on the ThinkRF website for more information.

See TRIGger Commands section (page 70) or SWEep's trigger (page 82) for further details.

Periodic Triggering

Periodic triggering provides a means of capturing a defined amount of IQ data on a periodic basis. Periodic triggering is typically used for statistical analysis of the captured signal.

Commands related to this feature are not available in this release.

External Triggering

External triggering provides a means of synchronized triggering based on the receiving of a trigger signal provided via the WSA5000's GPIO. The trigger "signal" could be a single pulse or a sync-word. See Synchronized Sweep (page 27) for additional details.

Caution: Contact ThinkRF's Support for details on how to use the GPIO port prior to connecting anything to the port.

Table 4: Trigger Control/Status Commands

SCPI Command	Description
:TRIGger	
:STATus?	Returns the status of the active trigger as to whether it is pending or has occurred
:TYPE[?]	Sets or disable the trigger type including LEVel | PERiodic | PULSe | WORD | NONE
:LEVel[?]	Sets the frequency range and amplitude of a frequency domain level trigger
:PERiodic[?]	Sets the time period of a periodic trigger

See TRIGger Commands section (page 70) or SWEep's trigger (page 82) for further details.

Capture Controller

The Capture Controller provides a means of defining and performing simple traces and complex sweeps. For example, it allows for:

- the definition and execution of a complex sweep;
- the interruption of that sweep;
- the execution of a specific trace; and
- the resumption of the previous sweep.

Caution: The configurations of the capture engine associated with :TRACe and :SWEep commands are fully independent of each other. A :TRACe command uses the configurations of the capture engine based on the root :INPut, :SENSe and :TRIGger commands. It does not use the configurations based on the :SWEep command subset.

Trace Capture Control

The :TRACe capture control initiates the capture, storage and conditionally the sending of IQ data through triggering. It supports both streaming and block mode capture.

The :TRACe:BLOCk command initiates a block capture of continuous IQ data (available to be "pulled" from the WSA5000 per command issued). Once it is issued, data will be stored instantly (conditional on triggering), contiguously and reliably and are available to be read. The maximum size of a block is limited by the memory device in the WSA.

The :TRACe:STReam command initiates the streaming of IQ data (which is "pushed" from the WSA5000). Once it is issued, data packets will be sent instantly (conditional on triggering) and continuously on best effort basis (in other words, data might not be continuous from one packet to the next once the internal buffer is full).

The execution of the trace capture could be conditioned by the triggering. The triggering may be enabled or disabled via the :TRIGger:TYPE command, thereby, supporting free-run or triggered signal searches.

Table 5: Trace Capture Control Commands

SCPI Command	Description
:TRACe	
:BLOCk	
[:DATA]?	Initiates the sending of the IQ data captured
:PACKets[?]	Sets the number of IQ data packets to be captured per block (a block = :PACKets * SPP)
:STReam	
:STARt	Initiates the capture, storage and streaming of IQ data
:STOP	Stops streaming
:SPPacket[?]	Defines the number of samples per VRT packet
:FORMat[?]	Sets the data output type to be time domain (IQ, I) or frequency domain (logPSD)

See TRACe Commands section (page 71) for further details.

Sweep Capture Control

The :SWEep capture control provides the ability to define and execute simple or complex sweeps. A sweep setup consists of defining a list or multiple lists and executing one of the defined lists, with each list consisting of one or more entries storing different capture engine configurations. A list may be edited, deleted and/or executed using the :SWEep:LIST command set.

The :SWEep:ENTRy commands provide the ability to define the capture engine configurations equivalent to most of :INPut, :SENSe, :TRACe and :TRIGger commands for each sweep entry. Sweep entries are identified by an index number and may be inserted, edited and/or deleted like rows in a table or spreadsheet.

There are slight differences between the configuration options for trace versus sweep captures. The sweep allows for definition of a range of center frequencies whereby the center frequency is incremented in frequency by a step value. Level triggers may be defined over the entire range of center frequencies. Sweeping does not support time delayed triggers.

In addition, sweep mode data packets, whether VRT context or digitized data, are “streamed” or “pushed” from the WSA (similar to :TRACe:STReam).

Table 6: Sweep Capture Control/Status Interface

SCPI Command	Description
:SWEep	
:LIST	
:CREAte	Creates a new list identified by a unique string identifier

SCPI Command	Description
:DELETE	Deletes the current list
:EDIT[?]	Sets the current list of which all subsequent :LIST commands pertain to
:STARt	Begins execution of the current sweep list from the first entry
:STOP	Stops execution of the current sweep list
:ITERations[?]	Defines the number of times the list is repeated during execution
:STATus?	
:ENTRy	*All entry commands operate on the current list*
:NEW	Sets the sweep entry settings to default values
:COPY	Copies the settings of an existing sweep entry into the current settings for quick editing
:SAVE	Saves the current editing entry to the end of the list or before the specified ID location in the list when the integer value is given
:DELETE	Deletes the specified entry or all entries
:READ?	Gets the settings of an existing sweep entry
:COUNt?	Gets the number of entries available in the list
:FORMat[?]	Sets the data output type to be time domain (IQ, I) or frequency domain (logPSD)
:MODE	As defined in :INPut:MODE, page 62
:ATTenuator	Enables/disables the front-end's 20 dB attenuation
:FILTer	
:PRESelect[?]	As defined in :INPut:FILTer:PRESelect, page 63
:GAIN	
:IF[?]	As defined in :INPut:GAIN:IF, page 63
:HDR[?]	As defined in :INPut:GAIN:HDR, page 64
:DECimation[?]	As defined in [:SENSe]:DECimation, page 65
:FREQuency	
:CENTer[?]	Defines the center frequency of the RFE or a range of frequencies that are stepped by the value defined by the :FREQuency:STEP
:STEP[?]	Defines the amount of frequency that the center frequency is stepped by
:SHIFt[?]	As defined in [:SENSe]:FREQuency:SHIFt, page 67
:TRIGger	
:TYPE[?]	As defined in :TRIGger:TYPE, page 70
:LEVel[?]	As defined in :TRIGger:LEVel, page 70
:PERiodic[?]	As defined in :TRIGger:PERiodic, page 71
:PPBlock[?]	Same as :TRACe:BLOCk:PACKets, page 73
:SPPacket[?]	As defined in :TRACe:SPPacket, page 72
:DWELl[?]	Sets the maximum amount of time waited for a trigger to occur after which the trigger is aborted

See SWEep Commands section (page 75) for further details.

Synchronized Sweep

The WSA5000 supports a synchronized sweep function for the purposes of comparing the same signal received via multiple WSA5000s.

Synchronized sweep is an extension of the external trigger capability. One of the WSA5000s in a network is configured to be the master (:SYSTem:SYNC:MASTer ON) and the other WSA5000s are configured as slaves (:SYSTem:SYNC:MASTer OFF). The master and slaves are configured with a sweep list, in which each sweep entry has a synchronization trigger type (:SWEep:ENTRy:TRIGger:TYPE PULSE | WORD). The synchronization trigger is generated and delivered from the master's GPIO to that of the slaves to indicate the beginning of a capture.

Figure 7 provides a synchronization trigger example using sync-word. The master sends the sync-word when the setup of its front-end has been completed. Master and slaves are also individually configured with a delay variable (:SYSTem:SYNC:WAIT <nsec> with a resolution of 8 nsec). This delay wait time accounts for the typical worst-case front-end setup time and for differences in the synchronization cable length. Master and slaves then begin the capture upon the expiration of the wait (or delay).

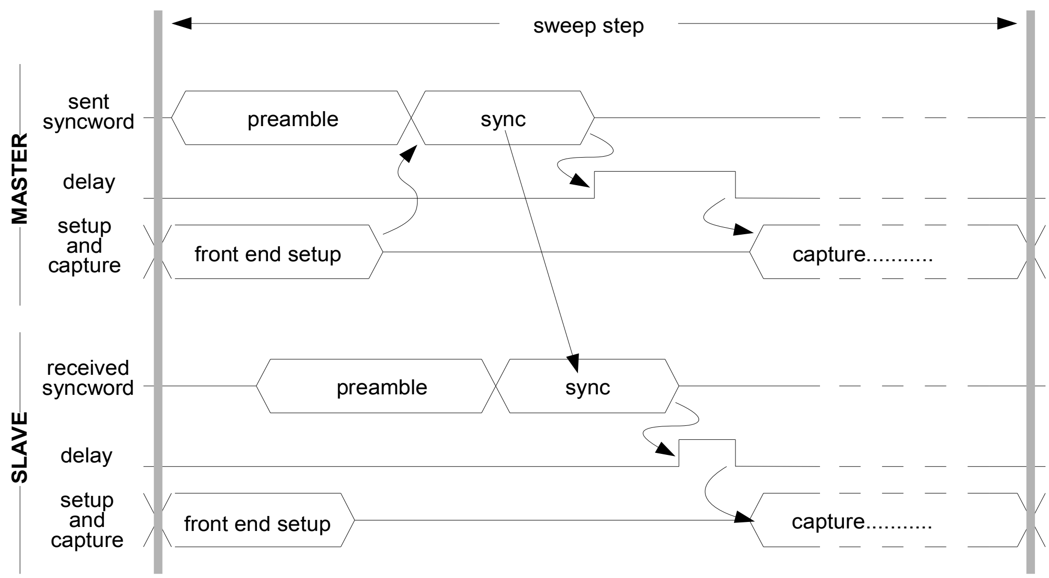


Figure 7: Synchronized Sweep using Sync-Word

The front-end setup time is typically of approximately 200 usec but is variable due to the embedded running processes. Referring to Figure 8, if the front-end setup time on one (or more) of the slaves is longer than the combined duration of the master's setup time plus the sync-word plus the slave's delay, then the slave will miss the beginning of the capture. The host-side application that is collating the capture data may recognize the missed capture by noting the timestamps and/or frequency of the capture data within the associated VRT Receiver Context packets. The rate of sweep versus the amount of missed captures may be balanced by adjusting the delay values.

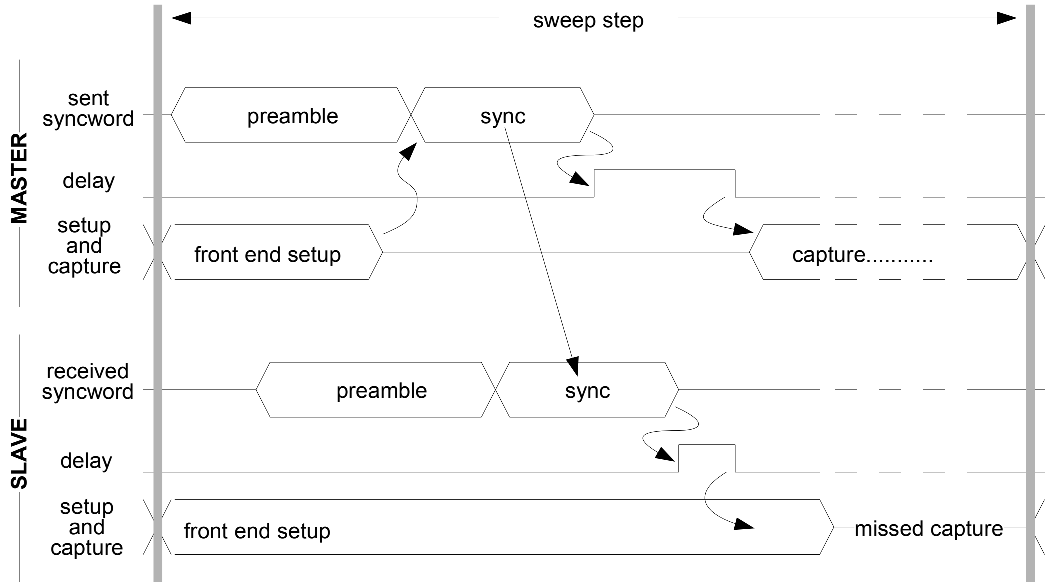


Figure 8: Synchronized Sweep with a Missed Capture

See SWEep Commands section (page 75) for further interface details.

VITA-49 Radio Transport Protocol

The section describes the WSA5000's VRT Information Class as per the "VITA Radio Transport (VRT) Draft Standard" Specification VITA-49.0 – 2007 Draft 0.21.

Purpose

Convey an arbitrary 100 MHz of IF data and associated information from the WSA5000 to another equipment using an industrial standard.

WSA5000's VRT Overview

ThinkRF's VRT supports four different packet streams of information defined and organized as shown in Figure 9 and Table 7. The streams of packets are sent when the data capture is started. The context packets carry the WSA5000 settings information associated with the immediate following IF data packets.

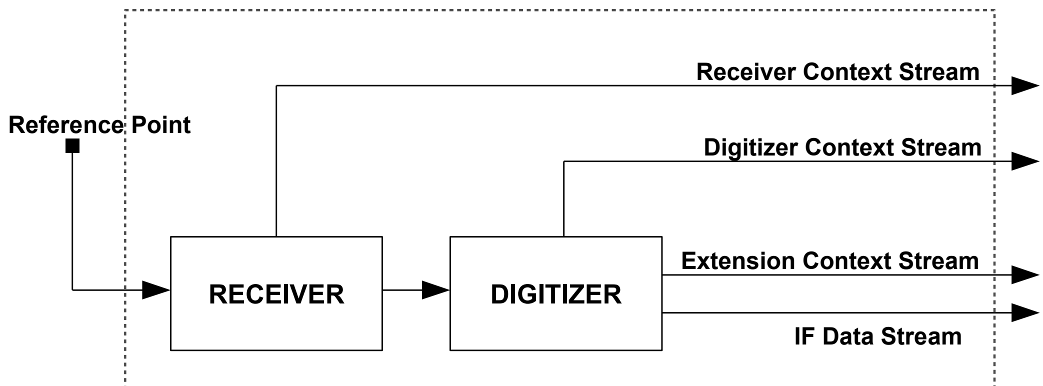


Figure 9: Connectivity and 4 Different Packet Streams Supported by WSA5000

Table 7: The Categories of VRT Packet Streams Supported by ThinkRF's WSA5000

Contents	Standard Formats	Custom Formats
Context	**IF Context Packet Stream** conveys metadata concerning IF Data Packet Stream and the settings *- Digitizer Context Packet Class Stream* *- Receiver Context Packet Class Stream*	**Extension Context Packet Stream** conveys additional Context concerning IF or Extension Data Packet Stream *- Extension Context Packet Class Stream*
Data	**IF Data Packet Stream** conveys discrete time sampled signal data *- IF Data Packet Class Stream*	**Extension Data Packet Stream** conveys any signal or data derived from a signal *- Currently not used*

Receiver Context Packet Class Stream

The Receiver Context Packet Class Stream is used to convey informational messages about changes in the configuration and status of the RF receiver in the WSA5000.

Digitizer Context Packet Class Stream

The Digitizer Context Class Stream is used to convey information messages about changes in the configuration and status of the IF digitizer in the WSA5000.

Extension Context Packet Class Stream

The Extension Context Packet Class Stream is used to convey metadata for the IF Data Packet Stream, which no provision has been made in the IF Context Packet Stream.

IF Data Packet Class Stream

The IF Data Packet Stream is used to convey complex IQ samples from the digitizer to devices external to the WSA5000.

Table 8 summarizes numerically the list of Stream Identifiers used by ThinkRF for different Packet Class Stream. Each ID will be mentioned in the subsequent corresponding Packet Class sections.

Table 8: A List of Stream Identifiers As Used by ThinkRF for Different Packet Classes

Stream Identifier	Packet Class
0x90000001	Receiver Context
0x90000002	Digitizer Context
0x90000003	IF Data – $\{I_{14}Q_{14}\}$ Format
0x90000004	Extension Context
0x90000005	IF Data – $\{I_{14}\}$ Format
0x90000006	IF Data – $\{I_{24}\}$ Format
0x90000007	IF Data – $\{PSD_{8}\}$ Format

Packet Classes and Streams

This section describes in details the rules and structure of those Packet Classes and Streams. By definition, a series of packets instantiated from the same Packet Class form a Packet Stream.

Note: All data words in each VRT packets are in big-endian order, and sent MSB first.

Receiver Context Packet Class

This Packet Class is a type of IF Context Packet Class. The packet information conveys changes in the configuration and status of the WSA5000's RF receiver.

Table 9: Receiver Context Packet Class Structure

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="4">Pkt Type
0 1 0 0</td><td>C</td><td colspan="2">R</td><td>T
S
M</td><td colspan="2">TSI</td><td colspan="2">TSF</td><td colspan="4">Pkt Count</td><td colspan="16">Pkt Size</td></tr>
<tr><td colspan="32">Stream Identifier (1 word)</td></tr>
<tr><td colspan="32">Timestamp - Integer Seconds (1 word)</td></tr>
<tr><td colspan="32">Timestamp - Integer Picoseconds (2 words)</td></tr>
<tr><td colspan="32">Context Indicator Field (1 word)</td></tr>
<tr><td colspan="32">Context Fields (Variable Size)</td></tr>
</table>

1. **Pkt Type** shall be set to **0100** to indicate this is context packet.
2. **C** shall be set to **0** to indicate there is no Class Identifier in the packet.
3. **R** shall be set to **00**, because they are reserved bits.
4. **TSM** (TimeStamp Mode) shall be set to **0**, indicating that context packet timestamps are precise.
5. **TSI** (TimeStamp-Integer) field shall be set to **01**, indicating that integer (seconds) part of the timestamps are in Coordinated Universal Time (UTC).
6. **TSF** (TimeStamp-Fractional) field shall be set to **10**, indicating that the fractional part of the timestamp measures in real time picosecond resolution.
7. **Pkt Count** shall start at 0000 and increment once for each context packet, until reaching 1111 (or 15), where it shall rollover to 0000 on the next count.
8. **Pkt Size** indicates the total number of 32-bit words in the entire context packet, including all headers, the context indicator field and context sections.
9. **Stream Identifier** shall be the 32-bit word, **0x90000001**
10. **Timestamp - Integer Seconds** shall be in UTC format and will represent the number of seconds occurred since Midnight, January 1, 1970, GMT.
11. **Timestamp - Integer Picoseconds** shall count the number of picoseconds past since the last increment of the Timestamp seconds field. See the Picosecond Timestamp Words Format section for the format.
12. The **Context Indicator Field** shall follow the format indicated in Table 10.

Table 10: Receiver Context Indicator Field Positions

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>I</td><td>R</td><td colspan="2">-</td><td>F</td><td colspan="3">-</td><td>G</td><td colspan="4">-</td><td>T</td><td colspan="18">-</td></tr>
</table>

13. The **Context Fields** section shall contain a context field for every field that is indicated to be present in the Context Indicator Field. *The fields shall be ordered in the identical order of their occurrence in the Context Indicator Field.* See Table 11 for the definition and associated value of each field.

Table 11: Receiver Context Field Definition and Values

Bit Name	Context Field	# of Words in Field	Period of Validity
I	Context Field Change Indicator	0	N/A
R	Reference Point	1	Persistent
F	RF Reference Frequency	2	Persistent
G	Gain	1	Persistent
T	Temperature	1	Persistent

Context Field Change Indicator

The Context Field Change Indicator is used to indicate when some context value of the system has changed. One or more of the other bits in the indicator field will be also set, indicating which values have been changed and have their updated values in the context fields that follow. It is possible that a context packet may be sent where the Context Field Change Indicator is set to 0, indicating that no change has occurred.

Reference Point

The Reference Point is used to indicate a point at which other context packets can be taken from as a reference. In this system, it is used to indicate which antenna port has been selected to receive signals.
When field R is marked in Receiver Context Indicator Field, one of the following values is stored in the Context Field to indicate which the signal source used:

- 0x01000001 – reference point P1 (antenna port 1),
- 0x01000002 – reference point P2 (antenna port 2),
- 0x01000004 – internal calibration signal.

RF Reference Frequency

The RF Reference Frequency communicates the frequency of origin for the signal. The value of the RF Reference Frequency shall be expressed in units of Hertz. The RF Reference Frequency sub-field shall use the 64-bit, two's-complement format as shown in Table 12. This field has an integer and a fractional part, with the radix point to the right of bit 20 in the second 32-bit word. This gives the RF Reference Frequency a range of ±8.79 THz with a resolution of 0.95 μHz.

Table 12: RF Reference Frequency Word Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Integer RF Reference Value (43..12), Hz																															
Integer RF Ref. Value (11..0), Hz												Fractional RF Reference Value(19..0)																			

Gain

The gain is a 32-bit value that is split into two 16-bit values, representing the Stage 1 and Stage 2 gain values. The Stage 1 gain represents the amount of gain in the front-end system, the RF gain. The Stage 2 gain represents the amount of gain in the back-end system, the IF gain.

Each gain value is a signed two's-complement number, having two sub-fields, bits 15:7 being the integer value, and 6:0 being the fractional value. This gives each gain figure a range of ±256 dB with a resolution of 1/128 dB (0.0078125 dB).

Table 13: Gain Field Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Integer IF									Fractional IF							Integer RF									Fractional RF						

Temperature

This field is not yet available.

The WSA5000 has a temperature sensor and will report changes in temperature to the system. The value of the Temperature field shall be expressed in units of degrees Celsius (°C). The Temperature field shall use the 32-bit format shown in Table 14 with the upper 16 bits reserved and shall be set to zero. The Temperature value shall be expressed in signed two's-complement format in the lower 16 bits of this field. This field has an integer and a fractional part, with the radix point to the right of bit 6.

Table 14: Temperature Field Format

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="16">Reserved</td><td colspan="10">Integer Temperature</td><td colspan="6">Fractional Temp.</td></tr>
</table>

The valid range of the Temperature field is -273.15 °C to +511.984375 °C. The resolution of the Temperature field is 0.015625 °C (1/64 °C).

For examples, a Temperature field value of:

- 0x0040 represents +1 °C,
- 0xFFC0 represents -1 °C,
- 0x0001 represents +0.015625 °C, and
- 0xFFFF represents -0.015625 °C.

Digitizer Context Packet Class

This Packet Class is a type of IF Context Packet Class. The packet information conveys changes in the configuration and status of the WSA5000's IF digitizer.

Table 15: Digitizer Context Packet Class Structure

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="4">Pkt Type
0 1 0 0</td><td>C</td><td colspan="2">R</td><td>T
S
M</td><td colspan="2">TSI</td><td colspan="2">TSF</td><td colspan="4">Pkt Count</td><td colspan="16">Pkt Size</td></tr>
<tr><td colspan="32">Stream Identifier (1 word)</td></tr>
<tr><td colspan="32">Timestamp - Integer Seconds (1 word)</td></tr>
<tr><td colspan="32">Timestamp - Integer Picoseconds (2 words)</td></tr>
<tr><td colspan="32">Context Indicator Field (1 word)</td></tr>
<tr><td colspan="32">Context Section (Variable Size)</td></tr>
</table>

1. **Pkt Type** shall be set to **0100** to indicate this is context packet.
2. **C** shall be set to **0** to indicate there is no Class Identifier in the packet.
3. **R** shall be set to **00**, because they are reserved bits.
4. **TSM** shall be set to **0**, indicating that context packet timestamps are precise.
5. **TSI** field shall be set to **01**, indicating that integer (seconds) part of the timestamps are in UTC.
6. **TSF** field shall be set to **10**, indicating that the fractional part of the timestamp measures real time picosecond resolution.
7. **Pkt Count** shall start at 0000 and increment once for each context packet, until reaching 1111 (or 15), where it shall rollover to 0000 on the next count.
8. **Pkt Size** indicates the size of the entire context packet, including all headers, the context indicator field and context sections.
9. **Stream Identifier** shall be the 32-bit word, **0x90000002**.

10. **Timestamp - Integer Seconds** shall be in UTC format and will represent the number of seconds occurred since Midnight, January 1, 1970 GMT.
11. **Timestamp - Integer Picoseconds** shall count the number of picoseconds past since the last increment of the Timestamp seconds field. See the Picosecond Timestamp Words Format section for the format.
12. The **Context Indicator Field** shall follow the format indicated in Table 16.

Table 16: Digitizer Context Indicator Field Bit Positions

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I		B			O		R																								

13. The **Digitizer Context Fields** section shall contain a context field for every field that is indicated to be present in the Context Indicator Field. The fields shall be ordered in the identical order of their occurrence in the Context Indicator Field. See Table 17 for the definition and associated value of each field.

Table 17: Digitizer Context Field Values

Bit Name	Context Field	# of Words in Field	Period of Validity
I	Context Field Change Indicator	0	N/A
B	Bandwidth	2	Persistent
O	RF Frequency Offset	2	Persistent
R	Reference Level	1	Persistent

Context Field Change Indicator

The Context Field Change Indicator is used to indicate when some context value of the system has changed. One or more of the other bits in the indicator field will be also set, indicating which values have been changed and have their updated values in the context fields that follow. It is possible that a context packet may be sent where the Context Field Change Indicator is set to 0, indicating that no change has occurred.

Bandwidth

The bandwidth is used to indicate that the amount of spectrum that is currently viewable due to decimation settings that have been enabled.

The Bandwidth field shall use the 64-bit, two's-complement format as shown in Table 18. This field has an integer and a fractional part, with the radix point to the right of bit 20 in the second 32-bit word. This gives the RF Reference Frequency a range of ±8.79 THz with a resolution of 0.95 μHz.

Table 18: Bandwidth Word Format

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="32">Integer Bandwidth (43..12), Hz</td></tr>
<tr><td colspan="12">Integer Bandwidth (11..0), Hz</td><td colspan="20">Fractional Bandwidth (19..0)</td></tr>
</table>

Reference Level

The Reference Level indicator provides a power level reference so that the magnitude of the received data can be calculated by a user. The reference level provided in the context packet is adjusted according to the RFE's gain setting. However, the reference levels must also be adjusted accordingly relative to the antenna (the Reference Point) and to any other conditions.

The absolute power level P (in dBm) is then computed using the following formula:

$$P = R + 20 * \log(IQ_{measured})$$

with

$$IQ_{measured} = \sqrt{(I_{fft}^2 + Q_{fft}^2)}$$

where:

R = the reference level provided in the VRT context packet, dBm
$IQ_{measured}$ = as shown in the formula above. The $IQ_{measured}$ formula, however, is a simplified example as it doesn't include any corrections, such as IQ imbalance, DC offset or windowing
I_{fft} = the real component of the FFT (Fast Fourier Transform) computation applied on the VRT I data, which is normalized by dividing each Q by $2^{bit\_size - 1}$
Q_{fft} = the imaginary component of the FFT computation applied on the VRT Q data (when used), which is normalized by dividing each Q by $2^{bit\_size - 1}$

The Reference Level field shall use the 32-bit format shown in Table 19 with the upper 16 bits reserved and shall be set to zero. The Reference Level field value shall be expressed in signed two's-complement format in the lower 16 bits of this field. This field has an integer and a fractional part, with the radix point to the right of bit 7.

Table 19: Reference Level Field Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																Integer Ref. Level									Frac. Ref. Level						

The value of the Reference Level field has a range of nearly ±256 dBm with a resolution of 1/128 dBm (0.0078125 dBm).

For examples, a Reference Level field value of:

- 0x0080 represents a reference level of +1 dBm,
- 0xFF80 represents -1 dBm,
- 0x0001 represents +0.0078125 dBm, and
- 0xFFFF represents -0.0078125 dBm.

RF Frequency Offset

The RF Frequency Offset indicator specifies the amount of frequency in Hz the received data has been shifted.

The RF Frequency Offset field shall use the 64-bit, two's-complement format as shown in Table 20. This field has an integer and a fractional part, with the radix point to the right of bit 20 in the second 32-bit word. This gives the RF Reference Frequency a range of ±8.79 THz with a resolution of 0.95 µHz.

Table 20: RF Frequency Offset Word Format

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="32">Integer RF Reference Value (43..12), Hz</td></tr>
<tr><td colspan="12">Integer RF Ref. Value (11..0), Hz</td><td colspan="20">Fractional RF Reference Value(19..0)</td></tr>
</table>

Extension Context Packet Class

This Packet Class conveys metadata concerning IF Data Packet Class that cannot be communicated in the IF Context Packet Class. See Table 21 for the organization of this context packet class.

Table 21: Extension Context Packet Class Structure

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="4">Pkt Type
0 1 0 1</td><td>C</td><td colspan="2">R</td><td>T
S
M</td><td colspan="2">TSI</td><td colspan="2">TSF</td><td colspan="4">Pkt Count</td><td colspan="16">Pkt Size</td></tr>
<tr><td colspan="32">Stream Identifier (1 word)</td></tr>
<tr><td colspan="32">Timestamp - Integer Seconds (1 word)</td></tr>
<tr><td colspan="32">Timestamp - Integer Picoseconds (2 words)</td></tr>
<tr><td colspan="32">Context Indicator Field (1 word)</td></tr>
<tr><td colspan="32">Context Fields (Variable Size)</td></tr>
</table>

1. **Pkt Type** shall be set to **0101** to indicate this is context packet.
2. **C** shall be set to **0** to indicate there is no Class Identifier in the packet.
3. **R** shall be set to **00**, because they are reserved bits.
4. **TSM** shall be set to **0**, indicating that context packet timestamps are precise.
5. **TSI** field shall be set to **01**, indicating that integer (seconds) part of the timestamps are in UTC.
6. **TSF** field shall be set to **10**, indicating that the fractional part of the timestamp measures in real time picosecond resolution.
7. **Pkt Count** shall start at 0000 and increment once for each context packet, until reaching 1111, where it shall rollover to 0000 on the next count.
8. **Pkt Size** indicates the total number of 32-bit words in the entire context packet, including all headers, the context indicator field and context sections.
9. **Stream Identifier** shall be the 32-bit word, **0x90000004**
10. **Timestamp - Integer Seconds** shall be in UTC format and will represent the number of seconds occurred since Midnight, January 1, 1970, GMT.
11. **Timestamp - Integer Picoseconds** shall count the number of picoseconds past since the last increment of the Timestamp seconds field. See the Picosecond Timestamp Words Format section for the format.
12. The **Context Indicator Field** shall follow the format indicated in Table 22.

Table 22: Extension Context Indicator Field Positions

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>I</td><td colspan="29"></td><td>St</td><td>Sw</td></tr>
</table>

13. The **Context Fields** section shall contain a context field for every field that is indicated to be present in the Context Indicator Field. The fields shall be ordered in the identical order of their occurrence in the Context Indicator Field. See Table 23 for the definition and associated value of each field.

Table 23: Receiver Context Field Definition and Values

Bit Name	Context Field	# of Words in Field	Period of Validity
I	Context Field Change Indicator	0	N/A
St	New Stream Start ID	1	Persistent
Sw	New Sweep Start ID	1	Persistent

Context Field Change Indicator

The Context Field Change Indicator is used to indicate when some context value of the system has changed. One or more of the bits in the indicator field will then be set, indicating which values have been changed and have their updated values in the context field(s) that follow. It is possible that a context packet may be sent where the Context Field Change Indicator is set to 0, indicating that no change has occurred.

New Stream Start ID

The New Stream Start ID indicator indicates a new stream capture has started, any packets following this Context Packet belong to this new stream capture.

The value of the New Stream Start ID field shall use the 32-bit unsigned integer format shown in Table 24.

Table 24: New Stream Start ID Field Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
New Stream Start ID																															

New Sweep Start ID

The New Sweep Start ID indicator indicates a new sweep has started, any packets following this Context Packet belong to this new sweep.

The value of the New Sweep Start ID field shall use the 32-bit unsigned integer format shown in Table 25.

Table 25: New Sweep Start ID Field Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
New Sweep Start ID																															

IF Data Packet Class

The IF Data Packet Class conveys digitized IF Data from the digitizer to devices external to the WSA5000. The payload data is dependent on the RFE modes of operation (:INPut:MODE or :SWEep:ENTRy:MODE) and capture output format (:TRACe:FORMat or :SWEep:ENTRy:FORMat). In addition to Table 2 (page 17), the following Table 26 describes the output data width and packing method for the different data type in order comply with VRT's 32-bit word output format:

Table 26: Output Data Width and Packing Method for Different Data Formats

Original Data Format	Binary Format Per Data Component	Signed Extension	Per 32-bit Word Packing Method
$\{I_{14}Q_{14}\}$	Signed 2-complement	$\{I_{16}Q_{16}\}$	$\{I_{16}Q_{16}\}$
$\{I_{14}\}$	Signed 2-complement	$\{I_{16}\}$	$\{I\mathbf{1}_{16}I\mathbf{2}_{16}\}$
$\{I_{24}\}$	Signed 2-complement	$\{I_{32}\}$	$\{I_{32}\}$
$\{PSD_8\}$	Positive 8-bit Integer	-	$\{PSD\mathbf{1}_8PSD\mathbf{2}_8PSD\mathbf{3}_8PSD\mathbf{4}_8\}$

The different Stream Identifier values will be used to indicate these different formats.

The order of the fields in an IF Data packet is organized as shown in Table 27. Packets is transmitted in big-endian byte order.

Table 27: IF Data Class Field Values

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="4">Pkt Type</td><td>C</td><td>T</td><td colspan="2">-</td><td colspan="2">TSI</td><td colspan="2">TSF</td><td colspan="4">Pkt Count</td><td colspan="16">Pkt Size</td></tr>
<tr><td colspan="32">Stream Identifier (1 word)</td></tr>
<tr><td colspan="32">Timestamp - Integer Seconds (1 word)</td></tr>
<tr><td colspan="32">Timestamp - Integer Picoseconds (2 words)</td></tr>
<tr><td colspan="32">Data Payload (Variable Size)</td></tr>
<tr><td colspan="32">Trailer (1 word)</td></tr>
</table>

1. **Pkt Type** shall be **0001**, indicating that a stream identifier is present.
2. **C** shall be set to **0**, indicating that there is no class identifier present.
3. **T** shall be set to **1**, indicating there is a trailer word in the packet.
4. **TSI** field shall be set to **01**, indicating that integer (seconds) part of the timestamps are in UTC.
5. **TSF** field shall be set to **10**, indicating that the fractional part of the timestamp measures in real time picosecond resolution.
6. **Pkt Count** shall start at 0000, and be incremented once for each IF Data packet that is received, until reaching 1111, when it then wraps back to 0000 on the next count.
7. **Pkt Size** shall be the number of 32-bit words that are present in the packet, including all headers, data payload and trailer if included.
8. **Stream Identifier** shall have the values as shown in the following Table 28.

Table 28: Stream Identifier Values for Different Data Output Formats

Data Output Format	Stream Identifier
$\{I_{14}Q_{14}\}$	0x90000003
$\{I_{14}\}$	0x90000005
$\{I_{24}\}$	0x90000006
$\{PSD_8\}$	0x90000007

9. **Timestamp - Integer Seconds** shall be in UTC format and will represent the number of seconds occurred since Midnight, January 1st, 1970, GMT.
10. **Timestamp - Integer Picoseconds** shall count the number of picoseconds past since the last increment of the Timestamp seconds field. See Table 29.

11. **Data Payload** shall contain the IF data from the WSA5000, arranged in the format indicated in Table 30 to Table 33.
12. **Trailer** shall be included and be arranged in the format described in Table 34.

Picosecond Timestamp Words Format

The two 32-bit words timestamp allotted for picoseconds are arranged as below.

Table 29: 64-bit or Two Words Picosecond Timestamp Format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Timestamp – picoseconds, Word 1 (63..32)																															
Timestamp – picoseconds, Word 2 (31..0)																															

Data Payload Format

The data payload of an IF Data packet contains a contiguous sequence of the Data Samples from an IF Data Sample stream. The number of words in the data payload is variable from packet to packet, and can be determined at the receiving end of the link from the Packet Size by subtracting the number of words dedicated to the header, trailer, and other additional fields. The presence or absence of these fields can be determined entirely from information in the header.

1. The maximum number of data payload 32-bit words shall be 2^{16}-16 and must be a multiple of 16. Limitation due to embedded data transferring engine.
2. The data payload shall consist of an integer number of contiguous 32-bit words.
3. IF Data Packets convey either the time domain in-phase (I or real) and/or quadrature (Q or imaginary) components forming the Complex Cartesian samples or the frequency domain logPSD data.

{$I_{14}Q_{14}$} Data Payload Format

4. Each I or Q data is a signed two's-complement 14-bit data with signed extended into 16-bit. Thus, each component is an integer ranging from -8192 to +8191 (or $\pm 2^{13}$).
5. The I-component is in the upper 16-bit of each data word followed by the Q-component in the lower 16-bit, as seen in Table 30.

Table 30: {$I_{14}Q_{14}$} Data Payload Arrangement with Upper 2-bit of Each Item Signed Extended to {I_{16} Q_{16}}

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0bxx		Item 1 (Sample 1 I_{14})														0bxx		Item 2 (Sample 1 Q_{14})													
0bxx		Item 3 (Sample 2 I_{14})														0bxx		Item 4 (Sample 2 Q_{14})													
0bxx		Item 5 (Sample 3 I_{14})														0bxx		Item 6 (Sample 3 Q_{14})													
⋮		⋮														⋮		⋮													

Example conversion, given the big-endian bytes 0x0018FFFE received:

- Split into two data items (i = 0x0018, q = 0xFFFE)
- Parse signed two's complement (i = 24, q = -2)
- Compute fractional value if needed: $i/2^{13}$ and $q/2^{13}$

{I_{14}} Data Payload Format

6. Each I data is a signed two's-complement 14-bit sample with signed extended into 16-bit. Thus, each component is an integer ranging from -8192 to +8191 (or $\pm 2^{13}$).
7. The first I sample is in the upper 16-bit of each data word follows by the second I sample in the lower 16-bit, as seen in Table 31.

Table 31: {I_{14}} Data Payload Arrangement with Upper 2-bit Signed Extended to {I_{16}}

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">0bxx</td><td colspan="14">Item 1 (Sample 1 I_{14})</td><td colspan="2">0bxx</td><td colspan="14">Item 2 (Sample 2 I_{14})</td></tr>
<tr><td colspan="2">0bxx</td><td colspan="14">Item 3 (Sample 3 I_{14})</td><td colspan="2">0bxx</td><td colspan="14">Item 4 (Sample 4 I_{14})</td></tr>
<tr><td colspan="2">0bxx</td><td colspan="14">Item 5 (Sample 5 I_{14})</td><td colspan="2">0bxx</td><td colspan="14">Item 6 (Sample 6 I_{14})</td></tr>
<tr><td colspan="2">⋮</td><td colspan="14">⋮</td><td colspan="2">⋮</td><td colspan="14">⋮</td></tr>
</table>

Same conversion example as {$I_{14}Q_{14}$}.

{I_{24}} Data Payload Format

8. Each data word is one I-component as seen in Table 32.
9. Each I data is a signed two's-complement 24-bit sample with signed extended into 32-bit. Thus, each component is an integer ranging from -8388608 to +8388607 (or $\pm 2^{23}$).

Table 32: {I_{24}} Data Payload Arrangement with Upper 8-bit Signed Extended to {I_{32}}

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">0bxxxxxxxx</td><td colspan="24">Item 1 (Sample 1 I_{24})</td></tr>
<tr><td colspan="8">0bxxxxxxxx</td><td colspan="24">Item 2 (Sample 2 I_{24})</td></tr>
<tr><td colspan="8">0bxxxxxxxx</td><td colspan="24">Item 3 (Sample 3 I_{24})</td></tr>
<tr><td colspan="8">⋮</td><td colspan="24">⋮</td></tr>
</table>

Examples conversion:

- Given the big-endian bytes 0x0018FFFE, then I_{24} = 0x18FFFFE.
- Given the big-endian bytes 0xFF800034, then I_{24} = 0x800034 (or -8388556).
- Compute fractional value if needed: $i/2^{23}$.

{PSD_8} Data Payload Format

10. Each data word consists of 4 logPSD components, with the first PSD_8 in the upper 8-bit MSB, follows by the second PSD_8, then the third and then the fourth as seen in Table 33.
11. Each logPSD data is a positive 8-bit data. Thus, each component is an integer ranging from 0 to 256 (or 2^8).

Table 33: {PSD_8} Data Payload Arrangement Into 32-bit Word

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Item 1 (Sample 1 PSD_8)</td><td colspan="8">Item 2 (Sample 2 PSD_8)</td><td colspan="8">Item 3 (Sample 3 PSD_8)</td><td colspan="8">Item 4 (Sample 4 PSD_8)</td></tr>
<tr><td colspan="8">Item 5 (Sample 5 PSD_8)</td><td colspan="8">Item 6 (Sample 6 PSD_8)</td><td colspan="8">Item 7 (Sample 7 PSD_8)</td><td colspan="8">Item 8 (Sample 8 PSD_8)</td></tr>
<tr><td colspan="8">⋮</td><td colspan="8">⋮</td><td colspan="8">⋮</td><td colspan="8">⋮</td></tr>
</table>

Example conversions, given the word 0x0018FFFE:

- Split into 4 PSD8 components: $PSD1_8$ = 0x00, $PSD2_8$ = 0x18, $PSD3_8$ = 0xFF, and $PSD4_8$ = 0xFE)
- Compute fractional value if needed: $i/2^8$ and $q/2^8$

Trailer Word Format

Table 34: Trailer Word Format

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="12">Enables</td><td colspan="12">State and Event Indicators</td><td rowspan="2">E</td><td colspan="7" rowspan="2">Associated Context Packet Count</td></tr>
<tr><td></td><td>DV</td><td>RL</td><td></td><td></td><td>SI</td><td>OR</td><td>SL</td><td></td><td></td><td></td><td></td><td></td><td>DV</td><td>RL</td><td></td><td></td><td>SI</td><td>OR</td><td>SL</td><td></td><td></td><td></td><td></td></tr>
</table>

State and Event Indicators and the associated Enable bits shall be positions as indicated in Table 35.

1. For each Indicator bit in the State and Event Indicators field, there is a corresponding Enable bit at the same position in the Enables field.
2. When an Enable bit is set to 1, the corresponding indicator shall function as shown in Table 35. Otherwise, the corresponding indicator shall not be considered valid.
3. Unused bits in the Enables field and the Indicators field shall be set to 0.
4. The E field shall be set to 0 to specify the Associated Context Packet Count field as undefined.
5. The Associated Context Packet Count field is unused and shall be set to 0.

Table 35: Trailer Indicator and Enable Bits

Enable Bit Position	Indicator Bit Position	Indicator Name
30	18	Valid Data Indicator
29	17	Reference Lock Indicator
26	14	Spectral Inversion Indicator
25	13	Over-range Indicator
24	12	Sample Loss Indicator

6. The **Valid Data Indicator**, when set to 1, shall indicate that the data in the packet is valid. When set to zero, it shall indicate that some condition exists that may invalidate the data.
7. The **Reference Lock Indicator,** when set to 1, shall indicate all PLLs in the system are locked and stable, and when set to 0, shall indicate one or all of the PLLs is not locked or unstable. It is very crucial to check this indicator bit.
8. The **Spectral Inversion Indicator**, when set to 1, shall indicate that the signal conveyed in the data payload has an inverted spectrum with respect to the spectrum of the signal at the system Reference Point. When processing the data

payload, for plotting purpose for instance, follow the suggested solution in Table 36 to properly display the spectrum.

Important Note: When using :OUTput:IQ:MODE CONNector with :INPut:MODE SH or SHN, the spectral inversion indicator is available through the GPIO port. Contact ThinkRF's Support for further details (or see “Synchronized Sweep with IQout” AppNote for important information).

9. The **Over-range Indicator** shall be set to 1 if any data value in the packet has reached full scale at the input of the digitizer.
10. The **Sample Loss Indicator**, when set to 1, shall indicate that data overflow occurs **after** the current captured VRT packet, **not within** the packet. In other words, the samples of the immediate packet following the current packet with the sample loss indicator bit high are not continuous from those of the current packet.

Table 36 lists the conditions in which an indicator would signal an abnormal state and the suggested resolutions.

Table 36: Conditions Causing Abnormal Indicator State and Suggested Resolution

Indicator Name	Abnormal State	Conditions	Suggested Resolution
Valid Data	0	1) One or more PLLs failed to lock. 2) In HDR mode, the NB ADC's filter has not settled.	1) Try to reset the frequency or restart the WSA. If the condition persists, contact ThinkRF's Support. 2) Discard the data packet and re-acquire the data. This condition is unlikely as the settle time is within hundred of nanoseconds.
Reference Lock	0	One or more PLLs failed to lock.	Same as 1) above
Spectral Inversion	1	Spectral inversion occurs when the frequency of the local oscillator exceeds that of the RF input signal being processed. At some signal frequency ranges input to the WSA5000, the IF output spectrum is inverted. Figure 10 illustrates an example.	Either swap each {I,Q} data point when both {I,Q} components are available or invert the output data bins of the computed spectral power. The latter suggestion is recommended since SH and SHN mode without the frequency shift particularly have {I} only data.
Over-range	1	The over-range threshold is the absolute full-scale of I or Q data. For WB ADC, the over-range threshold is at Vpeak = 1.0 V; and for NB ADC, Vpeak = 1.6 V.	- Enable the :ATTentuator if it is not yet on. - If :ATTentuator is already on, reduce the input level or the gain settings.
Sample Loss	1	This condition occurs only in Stream mode when the internal buffer is full.	Use a decimation value such that the transfer rate matches that of the capture rate.

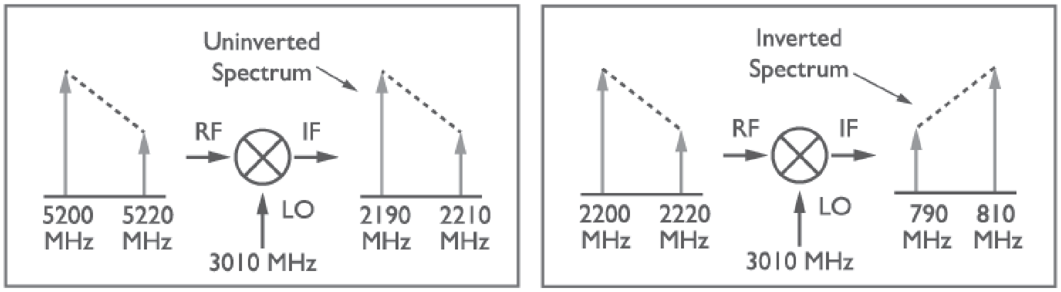


Figure 10: An Example Illustrating Uninverted and Inverted Spectrums

SCPI Command Set

This section is a SCPI reference guide for controlling the ThinkRF WSA5000 Wireless Signal Analyzer. The WSA5000 supports the Standard Commands for Programmable Instruments (SCPI) standard version 1999.0 as described in the following sections. SCPI lends itself to a command line interface and scripting, is supported by the major instrument vendors and provides a high level of familiarity for instrument users.

Note: ThinkRF's version of SCPI does not provide commands for network connection. The WSA5000 receives SCPI commands and sends query responds over port 37001. See Appendix A: Connecting to WSA for more details.

SCPI Language Overview

In the early 1990s, a group of instrument manufacturers developed Standard Commands for Programmable Instrumentation (SCPI) for controlling programmable instruments via a communication link, such as RS232, USB, LAN, etc. SCPI specifies the command structure and syntax using ASCII characters to provide some basic standardization and consistency to the control commands. SCPI commands, hence, lend themselves to communications with equipments via command line interface, scripting and/or programming languages such as C/C++, MATLAB®, Python, etc.

The SCPI language is based on a hierarchical or tree structure as illustrated in Figure 11 an example command set. The top level of the tree is the root node, which is followed by one or more lower-level nodes.

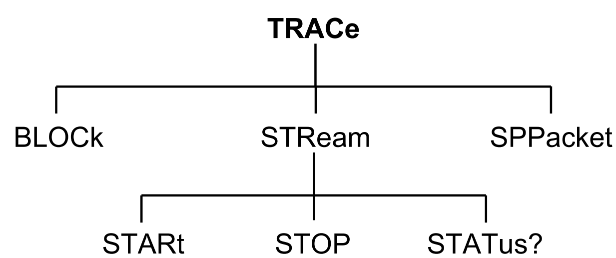


Figure 11: SCPI Language Hierarchical or Tree Structure Example

SCPI defines a measurement function block that is directly applicable to the ThinkRF WSA. The measurement function converts a physical signal into an internal data form that is available for formatting into bus data. It may perform the additional tasks of signal conditioning and post-conversion calculation. The measurement function box is subdivided into three distinct parts: INPut, SENSe, and CALCulate as seen in Figure 12.

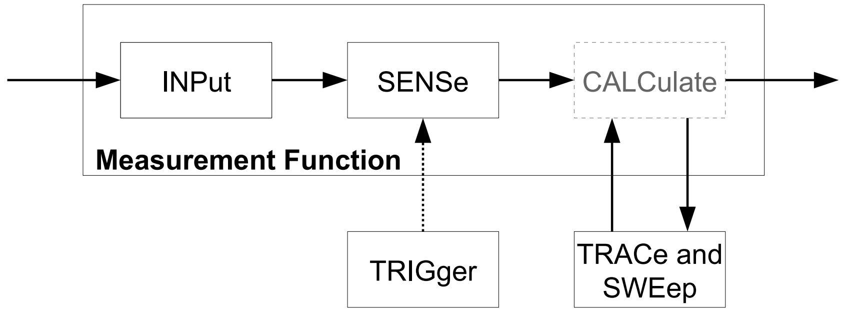


Figure 12: SCPI Measurement Function Block

Refer to the Appendix C: SCPI Command Syntax section for the general SCPI command syntax format and usage details.

IEEE Mandated SCPI Commands

These commands control and query the communication event/error and status registers as defined in the Appendix D: SCPI Status and Event Registers section. They are mandated by the IEEE.

Caution: The mandated IEEE SCPI commands are not affected by *RST command.

*CLS

The Clear Status (CLS) command clears the Status Byte register (STB), the standard Event Status register (ESR), the standard Questionable status register, the standard Operation Status register, and the error/event queue.

Syntax	*CLS
Parameter/Response	None

*ESE/*ESE?

*ESE command enables bits in the ESE register. The decimal integer value entered is the binary equivalent of the desired 8-bit mask. Bits enabled in ESE and set in ESR register will result in the Standard Event Status Summary bit (bit 5) in the STB register being set. This then allows the reading of ESR by using the *ESR? query command to determine the cause.

*ESE? query returns the decimal sum of the enabled bits in the ESE register. The decimal sum is the binary equivalent of the 8-bit mask.

See Figure 13 for the ESE/ESR register bits mapping.

Syntax	*ESE <integer> *ESE?
Parameter/Response	<integer>
Allowable Values	0 - 255

*ESR?

Query the standard Event Status Register (ESR), which returns the decimal sum of the bits set in the ESR. The decimal sum is the binary equivalent of the 8-bit mask. Any specific bit in the ESR will only appear set if and only if its event has occurred and the corresponding bit in the ESE is also enabled.

See Figure 13 for the ESR register bits mapping.

Caution: This is a destructive read. Once queried, the register is cleared.

Syntax	*ESR?
Parameter	None
Response	<integer>
Description	Refer to the Appendix D: SCPI Status and Event Registers section for the ESR register bit definition

*IDN?

Returns the WSA5000's identification information string.

Syntax	*IDN?
Parameter	None
Response	"<Manufacturer>,<Model>,<Serial number>,<Firmware version>"
Data Type	string

*OPC/*OPC?

*OPC (Operation Complete) command sets to confirm bit 0 in the ESR to 1 when all commands received before *OPC or *OPC? have been completed.

*OPC? returns the ASCII character 1 in the Standard Event register indicating completion of all pending operations. The query also stops any new commands from being processed until the current processing is complete.

Syntax	*OPC *OPC?
Parameter	None
Query Response	1

*RST

Resets the WSA5000 to its default settings. This includes stopping any running capture mode and trigger mode, and also performs :SYSTem:FLUSh.

*RST does not affect the registers or queues associated with the IEEE mandated commands. Each non-IEEE mandated command description in this reference shows the *RST value when affected.

Syntax *RST
Parameter/Response None

*SRE/*SRE?

The *SRE (Service Request Enable) command enables bits in the SRE register. The decimal integer value entered is the binary equivalent of the desired 8-bit mask to be enabled. Bits enabled in this register allow accessing the equivalent bits status in the STB register.

*SRE? query returns the decimal sum of the enabled bits in the SRE register. The decimal sum is the binary equivalent of the 8-bit mask.

See Figure 13 for the SRE/STB register bits mapping.

Syntax *SRE <integer>
*SRE?
Parameter/Response <integer>

*STB?

*STB? (Status Byte) query returns the decimal sum of the bits set in the STB register without erasing its content. The decimal sum is the binary equivalent of the 8-bit mask. Any specific bit in the STB will only appear set if and only if its event has occurred and the corresponding bit in the SRE is also enabled.

See Figure 13 for the ESE/ESR register bits mapping and the Status Byte Register (SBR) section of the Appendix D for the bits definition.

Syntax *STB?
Parameter None
Response <integer>

*TST?

*TST? (self-test) query initiates the device's internal self-test and returns one of the following results:

- 0 - all tests passed.
- 1 - one or more tests failed.

Syntax *TST?
Parameter None
Response 0 | 1
Data Type Integer

*WAI

*WAI (Wait-to-Continue) command suspends the execution of any further commands or queries until all operations for pending commands are completed.

Syntax	*WAI
Parameter/Response	None

SYSTem Commands

These commands control and query the communication event and status registers as defined in the Appendix D: SCPI Status and Event Registers. They are the minimal :SYSTem sets required in all SCPI instruments.

:SYSTem:VERSion?

This query returns the SCPI version number that the instrument software complies with.

Syntax	:SYSTem:VERSion?
Parameter	None
Response	<NR2>
Output Data Type	Decimal number YYYY.V
Example	:SYST:VERS

:SYSTem:CAPability?

This query returns the WSA5000's capabilities including firmware versions and installed hardware options The output is a list of SCPI-defined basic functionality of the WSA5000 and the additional capabilities it has in parallel (a&b) and singularly (a|b).

Further information will be provided in a future revision of this document.

Syntax	:SYSTem:CAPability?
Parameter	None
Response	<character>
Example	:SYST:CAP?

:SYSTem:OPTions?

This command queries the hardware option(s) or features that a particular WSA model supported. The response string contains comma separated 3-digit values to represent the options. See Table 37 for the translated list. Example, WSA5000-208-001 would has code 001 returned in the response string to indicate “external local oscillators” option.

Syntax :SYSTem:OPTions?

Parameter None

Response <xxx>{,<xxx>}

For WSA5000: <RF>,<Mixer>,<Digital>

Data Type Comma separated 3-digit value (ex: 000, 001, 019)

*RST State None

Example `:SYST:OPT?`

Table 37: WSA Option Codes and the Corresponding Description

Option Code	Description	Related SCPI Command
000	No Special Option	
001	External Local Oscillator Input[1]	[:SENSe]:FREQuency:LOSCillator?

[1] External Local Oscillator Input is a special WSA5000 variant, not available on all WSA5000s. Contact ThinkRF for more details.

:SYSTem:ABORt

This command will cause the WSA5000 to stop the data capturing, whether in the manual trace block capture, triggering or sweeping mode. The WSA5000 will be put into the manual mode; in other words, process such as streaming, trigger and sweep will be stopped. The capturing process does not wait until the end of a packet to stop, it will stop immediately upon receiving the command.

Syntax :SYSTem:ABORt

Parameter/Response None

*RST State N/A

Example `:SYST:ABORT`

:SYSTem:FLUSh

This command clears the WSA5000's internal data storage buffer of any data that is waiting to be sent. Thus, it is recommended that the flush command should be used when switching between different capture modes to clear up the remnants of packet within the WSA.

Caution: Issuing :SYSTem:FLUSh any time during streaming or sweeping mode will cause the stream or sweep capture to stop (abort) and switch automatically to block capture mode.

Note: Flush command only handles the WSA's internal buffer storage. The host application should ensure that the socket buffer is also cleared up of any potential data in the socket buffer. This can be done by calling the receive socket (non-blocking) until no data is returned. With Streaming or Sweeping, the start ID in a VRT extension packet marks the beginning of packets belonging to the new stream or sweep. This helps to distinct old packets from new packets.

Syntax	:SYSTem:FLUSh
Parameter/Response	None
*RST State	N/A
Examples	`:SYST:FLUSH`

:SYSTem:CAPTure:MODE?

This command returns what the current WSA data capture mode is (i.e. sweeping, streaming or block mode).

When stream or sweep mode is stopped, block mode will resume.

Syntax	:SYSTem:CAPTure:MODE?
Parameter	None
Response	BLOCK | STREAMING | SWEEPING
Output Data Type	Character
*RST State	BLOCK
Example	`:SYST:CAPTURE:MODE?`

:SYSTem:LOCK:REQuest?

This query attempts to attain the lock on the WSA5000 for a specific task, such as data acquisition. The query returns 1 when lock is successful or 0 if it fails.

Attaining a lock is equivalent to having the sole ownership for that task. This prevents multiple connected applications from doing the same task that would result in an erroneous operation or feedback from the WSA5000. The WSA5000's system lock ownership works in the following manner:

- The first application to connect to WSA5000 will automatically have the lock. The next application will need to perform this query request to attain the lock.
- When there is only one application connected (or the last one remained), that application will automatically has the lock.
- The last application that requested successfully has the lock until another application attained it.

Any application that doesn't have the specific lock will not be able to perform that task.

Note: When a TCP/IP socket connection is not exited properly, that socket might continue to exist in the WSA5000 server for a few minutes. This could affect a situation when only one application is used to connect to the WSA5000 as reconnection by that application might not get the lock. This application would then need to request the lock.

Syntax	:SYSTem:LOCK:REQuest? ACQuisition
Parameter	ACQuisition
Input Data Type	Character
Response	1 | 0 1 – Successfully locked 0 – Failed to lock
Output Data Type	Boolean
*RST State	N/A
Example	`:SYST:LOCK:REQ? ACQ`

:SYSTem:LOCK:HAVE?

This query returns the current lock state of the specified task.

Syntax	:SYSTem:LOCK:HAVE? ACQuisition
Parameter	ACQuisition
Input Data Type	Character
Response	1 | 0 1 – Have the lock 0 – Does not have the lock
Output Data Type	Boolean
*RST State	N/A
Examples	`:SYST:LOCK:HAVE? ACQ`

:SYSTem:SYNC:MASTer

This command sets the WSA unit to be the master or slave for a synchronization trigger system with multiple units, in which **only one unit** can be the master.

The master sends a sync-word or pulse (set through :TRIGger:TYPE or :SWEep:ENTRy:TRIGger:TYPE) via its GPIO to that of the slaves to indicate the beginning of a capture. The master WSA itself will has an internal loop-back of the synchronization signal it sent out.

Syntax	:SYSTem:SYNC:MASTer <Boolean> :SYSTem:SYNC:MASTer?
Parameter	ON | OFF | 1 | 0
Input Data Type	Integer | Character
Query Response	0 | 1
Output Data Type	Integer
*RST State	0
Examples	`:SYSTem:SYNC:MASTER ON` `:SYSTem:SYNC:MAST?`

:SYSTem:SYNC:WAIT

This command sets the delay time in nanoseconds that a WSA system must wait after receiving the satisfying trigger signal and before performing data capture. The delay time should be a multiple of 8 nsec as the WSA system runs with a 125 MHz clock.

Syntax	:SYSTem:SYNC:WAIT <integer> :SYSTem:SYNC:WAIT?
Parameter/Response	<integer>
Allowable Values	0 – 4294967295 (or 2^{32}-1)
*RST State	0
Examples	`:SYSTem:SYNC:WAIT 120` `:SYSTem:SYNC:WAIT?`

:SYSTem:DATE

This command set or queries the current date of the WSA5000. When the date is set, the change is applied to the real time clock (RTC) of the WSA5000 system, and the :SYSTem:TIME:SYNC field is changed to DISable automatically. The date returned is representative of the current time mode that is UTC.

This command is not affected by a power-on, factory reset, or *RST command.

Syntax	:SYSTem:DATE <integer>,<integer>,<integer> :SYSTem:DATE?
Parameters/Response	<year>,<month>,<date>
I/O Data Type	Comma separated integers
Allowable Values	Year: YYYY - requires a four digit integer Month: 1 - 12 Date: 1 - 31
*RST State	N/A
Examples	`:SYST:DATE 2012,12,2`

:SYSTem:TIME

This command set or queries the current time of the WSA5000. When the time is set, the change is applied to the RTC of the WSA5000 system, and the :SYSTem:TIME:SYNC field is changed to DISable automatically.

The time returned is representative of the current time mode that is UTC.

This command is not affected by a power-on, factory reset, or *RST command.

Syntax	:SYSTem:TIME <integer>,<integer>,<integer>[,<integer>] :SYSTem:TIME?
Parameters/Response	<hour>,<minute>,<second>[,<millisecond>]
I/O Data Type	Comma separated integers
Allowable Values	Hour: 0 - 23 Minute: 0 - 59 Second: 0 – 59 Millisecond: 0 - 999
*RST State	N/A
Examples	`:SYST:TIME 10,30,15`

:SYSTem:TIME:ADJust

This command adjusts the system time relative to its current time.

Further information will be provided in a future revision of this document.

Syntax	:SYSTem:TIME:ADJust <integer> [unit] :SYSTem:TIME:ADJust?
Parameters	<second or sub-second> [unit]
Response	<integer>
Allowable Values	0 – 4294967295 (or 2^{32} - 1)
Default I/O unit	ns
*RST State	0
Examples	`:SYST:TIME:ADJUST 10 ns` `:SYSTEM:TIME:ADJUST?`

:SYSTem:TIME:SYNC

This command selects the time synchronization source for WSA5000 and the query returns the source selected. Choosing NTP (Network Time Protocol) as the synchronization source will impact the system real time clock (RTC), causing it to update either at a continuous interval or one time only. When :SYSTem:DATE and/or :SYSTem:TIME commands are used to change the time, the source will automatically be changed to DISable.

*RST does not affect this command. At factory install, the synchronization is defaulted to disabled.

Syntax	:SYSTem:TIME:SYNC DISable | NTP,{ONCE | CONTinuous} :SYSTem:TIME:SYNC?
Parameter	DISable | NTP,{ONCE | CONTinuous}
Response	DISabled | NTP,{ONCE | CONTinuous}
I/O Data Type	Character | Comma separated characters
*RST State	DISable

Examples	`:SYST:TIME:SYNC NTP,ONCE`
	`:SYST:TIME:SYNC DISABLE`
	`:SYSTEM:TIME:SYNC?`

:SYSTem:TIME:SYNC:STATus?

This command returns the current status of the time synchronization.

Further information will be provided in a future revision of this document.

Syntax	:SYSTem:TIME:SYNC:STATus?
Parameter	None
Response	TBD
Data Type	TBD
Examples	`:SYST:TIME:SYNC:STAT?`

:SYSTem:COMMunicate:LAN:CONFigure

The set command will store the new LAN configuration type to be applied to the WSA. This command does not take effect until :SYSTem:COMMunicate:LAN:APPLy is sent (please refer to the Caution note of the :APPLy command). Once the option is applied, it is not affected by power-on, :STATus:PRESET, or *RST.

The query will return the option set or that of the actual current configuration if one is not set. The CURRENT query will return what is currently and actually used by the WSA's LAN interface.

Note: *RST command cannot be used to set the box to its manufacturing default state of DHCP. To set the box back to DHCP from a working STATIC mode, use this command or the web-browser as mentioned in the WSA User's Guide.

Syntax	SYSTem:COMMunicate:LAN:CONFigure DHCP | STATIC SYSTem:COMMunicate:LAN:CONFigure? [CURRENT]
Parameter	Set: DHCP | STATIC Query: [CURRENT]
Response	DHCP | STATIC
I/O Data Type	Character
*RST State	N/A

Examples	`:SYST:COMM:LAN:CONF DHCP`
	`:SYSTEM:COMMUNICATE:LAN:CONF?`
	`:SYST:COMM:LAN:CONF? CURRENT`

:SYSTem:COMMunicate:LAN:DNS

The set command will store the new LAN DNS server address(es) to be applied to the WSA. This command does not take effect until :SYSTem:COMMunicate:LAN:APPLy is sent (please refer to the Caution note of the :APPLy command). Once the setting is applied, it is not affected by power-on, :STATus:PRESET, or *RST.

The query will return the LAN DNS address(es) set or that of the actual current configuration if one is not issued. The CURRENT query will return what is currently and actually used by the WSA's LAN interface.

Syntax SYSTem:COMMunicate:LAN:DNS <main DNS>[,alternative DNS]
SYSTem:COMMunicate:LAN:DNS? [CURRENT]

Parameter Set: D.D.D.D[,D.D.D.D] where D = 0 – 255
Query: [CURRENT]

Response D.D.D.D[,D.D.D.D]

*RST State N/A

Examples
```
SYSTEM:COMMUNICATE:LAN:DNS 208.67.110.0
SYST:COMM:LAN:DNS 208.67.110.0,208.67.100.10
SYSTEM:COMMUNICATE:LAN:DNS?
SYST:COMM:LAN:DNS? CURRENT
```

:SYSTem:COMMunicate:LAN:GATEway

The set command will store the new LAN gateway to be applied to the WSA. This command does not take effect until :SYSTem:COMMunicate:LAN:APPLy is sent (please refer to the Caution note of the :APPLy command). Once the setting is applied, it is not affected by power-on, :STATus:PRESET, or *RST.

The query will return the gateway address set or that of the actual current configuration if one is not issued. The CURRENT query will return what is currently and actually used by the WSA's LAN interface.

Syntax SYSTem:COMMunicate:LAN:GATEway <IPv4 address>
SYSTem:COMMunicate:LAN:GATEway? [CURRENT]

Parameter Set: D.D.D.D where D = 0 – 255
Query: [CURRENT]

Response D.D.D.D

*RST State N/A

Examples
```
SYST:COMM:LAN:GATEWAY 102.101.0.13
SYSTEM:COMMUNICATE:LAN:GATEWAY?
SYST:COMM:LAN:GATE? CURRENT
```

:SYSTem:COMMunicate:LAN:IP

The set command will store the new LAN IP to be applied to the WSA. This command does not take effect until :SYSTem:COMMunicate:LAN:APPLy is sent (please refer to the Caution note of the :APPLy command). Once the setting is applied, it is not affected by power-on, :STATus:PRESET, or *RST.

The query will return the IP address set or that of the actual current configuration if one is not issued. The CURRENT query will return what is currently and actually used by the WSA's LAN interface.

Syntax SYSTem:COMMunicate:LAN:IP <IPv4 address>
SYSTem:COMMunicate:LAN:IP? [CURRENT]

Parameter Set: D.D.D.D where D = 0 – 255
Query: [CURRENT]

Response D.D.D.D

*RST State N/A

Examples
```
SYST:COMM:LAN:IP 101.125.1.16
SYSTEM:COMM:LAN:IP?
SYST:COMM:LAN:IP? CURRENT
```

:SYSTem:COMMunicate:LAN:NETMask

The set command will store the new LAN netmask address to be applied to the WSA. This command does not take effect until :SYSTem:COMMunicate:LAN:APPLy is sent (please refer to the Caution note of the :APPLy command). Once the setting is applied, it is not affected by power-on, :STATus:PRESET, or *RST.

The query will return the maskaddress set or that of the actual current configuration if one is not issued. The CURRENT query will return what is currently and actually used by the WSA's LAN interface.

Syntax SYSTem:COMMunicate:LAN:NETMask <address>
SYSTem:COMMunicate:LAN:NETMask? [CURRENT]

Parameter Set: D.D.D.D where D = 0 – 255
Query: [CURRENT]

Response D.D.D.D

*RST State N/A

Examples
```
SYST:COMM:LAN:NETMASK 255.255.255.0
SYSTEM:COMMUNICATE:LAN:NETM?
SYST:COMM:LAN:NETM? CURRENT
```

:SYSTem:COMMunicate:LAN:APPLy

This command will apply the changes to the LAN settings and the embedded system will automatically reconfigure the Ethernet to put in effect the new LAN setting. Once the LAN settings are applied, they are not affected by power-on, :STATus:PRESET, or *RST.

Caution: When changing from DHCP to STATIC mode, this command should to be sent only when all the required LAN settings are set using the appropriate :SYSTem:COMMunicate:LAN commands.

Syntax :SYSTem:COMMunicate:LAN:APPLy

Parameter/Response None

*RST State N/A

Examples
```
:SYST:COMM:LAN:APPLY
```

:SYSTem:ERRor[:NEXT]?

This query returns the oldest uncleared error code and message from the SCPI error/event queue. When there are no error messages, the query returns 0,"No error". *RST does not affect the error queue.

Note: It is recommended to do a :SYSTem:ERRor[:NEXT]? (or :SYSTem:ERRor:ALL?) query command after each non-query command is sent to ensure that the non-query command is executed without error. Since each error message is queued into a buffer, if multiple commands have been sent follow by only one :SYSTem:ERRor[:NEXT]? command, it would be uncleared which command has resulted in which error.

Syntax	:SYSTem:ERRor[:NEXT]?
Parameter	None
Response	<integer>,<string> ::= <error code>,<description>
Description	Refer to the Appendix D: SCPI Status and Event Registers section
Example	`:SYST:ERR?`

:SYSTem:ERRor:ALL?

This query returns all the uncleared error codes and messages from the SCPI error/event queue. If there are no error messages, the query returns 0,"No error".

Syntax	:SYSTem:ERRor:ALL?
Parameter	None
Response	<integer>,<string>{,<integer>,<string>} ::= <error code>,<description>{,<error code>,<description>}
Description	Refer to the Appendix E: SCPI Error Codes Used section
Example	`:SYST:ERR:ALL?`

STATus Commands

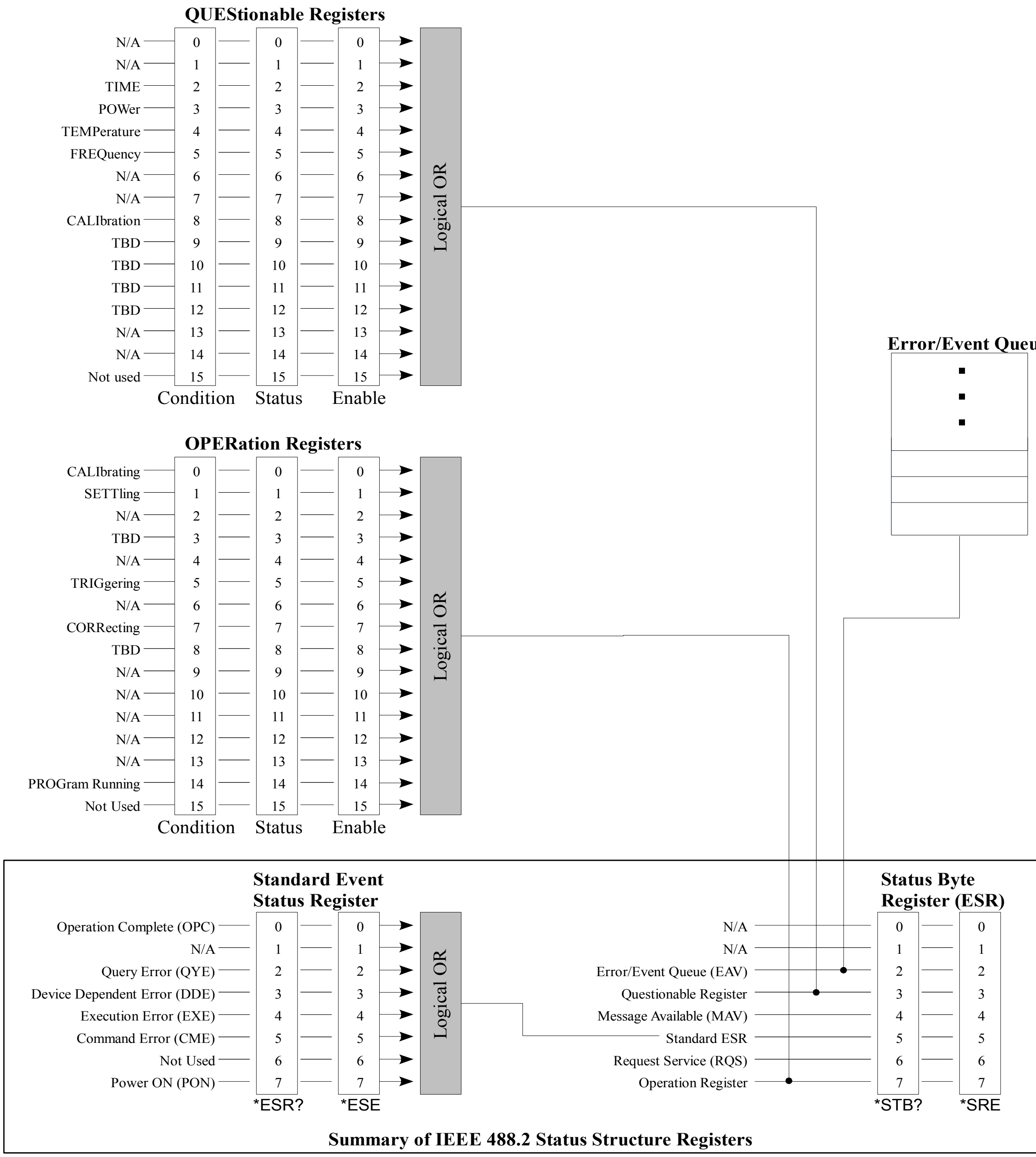


Figure 13: Status Reporting Structure with Status & Enable Registers

The STATus commands control the SCPI-defined status-reporting structures as illustrated in Figure 13.

SCPI defines the QUEStionable, OPERation, Instrument SUMmary and INSTrument registers in addition to those in IEEE 488.2. These registers conform to the IEEE 488.2

specification and each may be comprised of a condition register, an event register, an enable register, and negative and positive transition filters.

SCPI also defines an IEEE 488.2 queue for status. The queue provides a human readable record of instrument events. The application programmer may individually enable events into the queue. :STATus:PRESET enables errors and disables all other events. If the summary of the queue is reported, it shall be reported in bit 2 of the status byte register. A subset of error/event numbers is defined by SCPI.

:STATus:OPERation[:EVENt]?

This command queries the standard Operation Status Register (OSR) for any event. The query returns the decimal sum of the bits set in the OSR. The decimal sum is the binary equivalent of the 16-bit mask. The last bit is unused. Any specific bit in the OSR will only appear set if and only if its event has occurred and the corresponding bit in the OSE is also enabled (refer to Appendix D: SCPI Status and Event Registers).

Caution: This query clears all bits in the register to 0.

See Figure 13 for the Operation Status register bits mapping.

Syntax	:STATus:OPERation[:EVENt]?
Parameter	None
Response	<integer>
Allowable Values	0 – 32767 (2^{15}-1)
*RST State	None
Example	`:STAT:OPER?`

:STATus:OPERation:CONDition?

This command queries the standard Operation Condition Register (OCR) for any questionable event. The query returns the decimal sum of the bits set in the OCR. The decimal sum is the binary equivalent of the 16-bit mask. The last bit is unused. Any specific bit in the OCR will only appear set if and only if its event has occurred and the corresponding bit in the OSE is also enabled (see :STATus:OPERation:ENABle section). The content of the OCR remains unchanged after it is read.

The data in this register is continuously updated to reflect the most current conditions.

See Figure 13 for the Operation Condition register bits mapping.

Syntax	:STATus:OPERation:CONDition?
Parameter	None
Response	<integer>
Allowable Values	0 – 32767 (2^{15}-1)
*RST State	None
Example	`:STAT:OPER:COND?`

:STATus:OPERation:ENABle

This command enables or queries bits in the Operation Enable register (OER). The decimal integer value entered is the binary equivalent of the desired 16-bit mask to be enabled. Bits enabled in this register allow accessing the equivalent bits status in the OSR and OCR registers.

Bits enabled in OER and set in OSR/OCR register will result in the Standard Operation Status Summary bit (bit 7) in the STB register being set. See Figure 13.

Syntax	:STATus:OPERation:ENABle <integer> :STATus:OPERation:ENABle?
Parameter/Response	<integer>
Allowable Values	0 – 32767 (2^{15}-1)
*RST State	0
Examples	`:STAT:OPER:ENAB 256` `:STAT:OPER:ENAB?`

:STATus:PRESET

This command presets the WSA5000 (similar to *RST), and OSE and QSE to zero.

Syntax	:STATus:PRESET
Parameter/Response	None

:STATus:QUEStionable[:EVENt]?

This command queries the standard Questionable Status Register (QSR) for any event. The query returns the decimal sum of the bits set in the QSR. The decimal sum is the binary equivalent of the 16-bit mask. The last bit is unused. Any specific bit in the QSR will only appear set if and only if its event has occurred and the corresponding bit in the QSE is also enabled (refer to Appendix D: SCPI Status and Event Registers).

Caution: This query clears all bits in the register to 0.

See Figure 13 for the Questionable Status register bits mapping.

Syntax	:STATus:QUEStionable[:EVENt]?
Parameter	None
Response	<integer>
Allowable Values	0 – 32767 (2^{15}-1)
*RST State	None
Example	`:STAT:QUES?`

:STATus:QUEStionable:CONDition?

This command queries the standard Questionable Condition Register (QCR) for any questionable event. The query returns the decimal sum of the bits set in the QCR. The decimal sum is the binary equivalent of the 16-bit mask. The last bit is unused. Any specific bit in the QCR will only appear set if and only if its event has occurred and the corresponding bit in the QSE is also enabled (refer to Appendix D: SCPI Status and Event Registers). The content of the QCR remains unchanged after it is read.

The data in this register is continuously updated to reflect the most current conditions.

See Figure 13 for the Questionable Condition register bits mapping.

Syntax	:STATus:QUEStionable:CONDition?
Parameter	None
Response	<integer>
Allowable Values	0 – 32767 (2^{15}-1)
*RST State	None
Example	`:STAT:QUES:COND?`

:STATus:QUEStionable:ENABle

This command enables bits in the Questionable Enable register (QER). The decimal integer value entered is the binary equivalent of the desired 16-bit mask to be enabled. Bits enabled in this register allow accessing the equivalent bits status in the QSR and QCR registers.

Bits enabled in QER and set in QSR/QCR register will result in the Standard Questionable Status Summary bit (bit 3) in the STB register being set. See Figure 13.

Syntax	:STATus:QUEStionable:ENABle <integer> :STATus:QUEStionable:ENABle?
Parameter/Response	<integer>
Allowable Values	0 – 32767 (2^{15}-1)
*RST State	0
Examples	`:STAT:QUES:ENAB 256` `:STAT:QUES:ENAB?`

:STATus:TEMPerature?

This command queries the WSA's internal temperature provided by one or more temperature sensors. The response field varies depending on how many sensors are available in a WSA model. The WSA5000 model, for instance, returns comma separated values for the sensors at the RF, Mixer and Digital sections.

Syntax	:STATus:TEMPerature?
Parameter	None
Response	<NRf>{,<NRf>} For WSA5000: <RF>,<Mixer>,<Digital>
Data Type	Float
Unit	degrees Celsius
*RST State	None

CALibrate Commands

:CALibrate:RUN

This command starts the calibration of the WSA5000 system.

Syntax	:CALibrate:RUN
Parameter/Response	None
*RST State	Calibration off

:CALibrate:STATus?

This command queries the status of the calibration. 'On' indicates the calibration process is running and 'off' means either it has done calibrating or was not initiated.

Syntax	:CALibrate:STATus?
Response	ON | OFF
*RST State	None

:CALibrate:ABORt

This command aborts the calibration process of the WSA5000 system.

Syntax	:CALibrate:ABORt
Parameter/Response	None
*RST State	None

INPut Commands

:INPut:MODE

This command sets or queries the WSA5000's RFE mode of operation.

Notes: The RFE modes affect the data packing method due to the different output data width. The type of DSP applied would also change the data output of some of the modes as well. For example, SH mode with frequency shift would change from I_{14} only data to $I_{14}Q_{14}$ data output. The VRT's Stream ID would identify the format accordingly. See Table 2: Radio RFE Modes and DSP Data Output Formats (page 17) and VRT's IF Data Packet Class (page 37).

- It is also important to see Table 2 for the IBW of each mode and the related notes.

- If :OUTput:IQ:MODE CONNector is to be used with SH or SHN mode, see the Important Note listed under that command.

Syntax	:INPut:MODE ZIF | DD | HDR | IQIN | SH | SHN[1] :INPut:MODE?
Parameter/Response	ZIF | DD | HDR | IQIN | SH | SHN
Data Type	Character
*RST State	*Product verion dependent*[2]
Examples	`:INP:MODE HDR` `:INPUT:MODE?`

[1] SHN mode is only available in WSA5000 hardware revision 3. See *IDN? to find out your hardware version (or the Administrative web-console to the box).
[2] The RFE Mode availability is product dependent. WSA5000-108, for instance, does not have ZIF and IQIN mode. Hence, the *RST state and the initial power-up default would be different depending on the product version.

:INPut:ATTenuator

This command enables, disables or queries the WSA5000's RFE 20 dB attenuation.

Syntax	:INPut:ATTenuator <Boolean> :INPut:ATTenuator?
Parameter	ON | OFF | 1 | 0
Input Data Type	Integer | Character
Query Response	1 | 0
Output Data Type	Integer
*RST State	1
Examples	`:INP:ATT ON` `:INPUT:ATT?`

:INPut:FILTer:PRESelect

This command sets or queries the RFE preselect filter selection.

Syntax	:INPut:FILTer:PRESelect <Boolean> :INPut:FILTer:PRESelect?
Parameter	ON | OFF | 1 | 0
Input Data Type	Integer | Character
Query Response	0 | 1
Output Data Type	Integer
*RST State	0
Examples	`:INP:FILT:PRES ON` `:INP:FILTER:PRES?`

:INPut:GAIN:IF

This command sets or queries variable IF gain stages of the RFE.

Syntax	:INPut:GAIN:IF <NR1 [unit]> :INPut:GAIN:IF? [MAX | MIN]
Parameter	<NR1 [unit]>
Input Data Type	Signed integer [character]
Query Response	<NR1>
Output Data Type	Signed integer
Allowable Values	0 to 30
Default I/O unit	dB
*RST State	0 dB

Examples	`:INPUT:GAIN:IF -2`
	`:INP:GAIN:IF 20 dB`
	`:INP:GAIN:IF?`

:INPut:GAIN:HDR

This command sets or queries variable NB IF gain of the HDR signal path.

Syntax	:INPut:GAIN:HDR <NR1 [unit]> :INPut:GAIN:HDR? [MAX | MIN]
Parameter	<NR1 [unit]>
Input Data Type	Signed integer [character]
Query Response	<NR1>
Output Data Type	Signed integer
Allowable Values	-10 to 30
Default I/O unit	dB
*RST State	-10 dB

Examples	`:INPUT:GAIN:HDR -5`
	`:INP:GAIN:HDR 20 dB`
	`:INP:GAIN:HDR?`

SOURce Commands

:SOURce:REFerence:PLL

This command selects and queries the 10 MHz reference clock source, whether it be via the internal source or through the external SMA connector.

Caution: When the external 10 MHz reference is used, its reference level **must** be between **-10dBm** and **0 dBm**. Exceeding the level of 0 dBm will result in permanent damage to the internal clock circuit. Additionally, the 10 MHz reference must be powered down prior to powering down the WSA5000.

Syntax	:SOURce:REFerence:PLL INT | EXT :SOURce:REFerence:PLL?
Parameter/Response	INT | EXT
Data Type	Character

*RST State	INT
Examples	`:SOURCE:REF:PLL INT` `:SOUR:REF:PLL?`

:SOURce:REFerence:PLL:RESET

This command resets the 10 MHz reference clock in to the internal source.

Syntax	:SOURce:REFerence:PLL:RESET
Parameter/Response	None
*RST State	INT
Examples	`:SOURCE:REF:PLL:RESET`

SENSe Commands

[:SENSe]:CORRection:DCOFfset

This command sets or queries the ADC's DC-offset correction state.

Syntax	[:SENSe]:CORRection:DCOFfset <Boolean> [:SENSe]:CORRection:DCOFfset?
Parameter	ON | OFF | 1 | 0
Input Data Type	Integer | Character
Query Response	0 | 1
Output Data Type	Integer
*RST State	1
Examples	`:SENS:CORR:DC OFF` `CORR:DC?`

[:SENSe]:DECimation

This command sets or queries the rate of decimation of samples in a trace capture. When the rate is set to 1 (or OFF), no decimation is performed on the trace capture. The decimation range varies depending on the RFE modes as described below.

In HDR mode, the decimation is done directly by the on-board NB ADC. The decimation value supported by this mode is 1, 2 and 4.

In the remaining RFE modes, WSA5000 uses DDC to provide 10 levels of decimation of values 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 (i.e. decimation rate = 2^{level} where level = 1, 2 – 10). The decimation process consists of CIC filters and FIR filters, each type of filters with its own decimator. The decimator captures one sample at every <integer> number of samples. The filters are arranged in the following manner:

- For the decimation rate of 4, only a FIR filter with a fixed decimation by 4 is used, CIC filter is bypassed.
- For the decimation rates of 8 to 1024, a 4-stage CIC of rate 4 to 512 is applied first for each I and/or Q data. The resulting I and/or Q data pipes are then passed to a

FIR filter with a fixed decimation of 2 to arrive at the rate set (i.e. if the rate is set to 16 for example, I and Q data will first pass-through the CIC filters with a decimation rate of 8; then, the CIC output will be further decimated by 2 by the FIR filter with the fixed decimation rate of 2).

Note: When in SH/SHN mode, the WSA5000 will automatically be shifted by 35 MHz to the zero IF when decimation is applied. This implies the VRT data output will be I and Q for SH/SHN with decimation.

Syntax	:SENSe:DECimation OFF | <integer> :SENSe:DECimation? [MAX | MIN]
Parameter	OFF | <integer>
Allowable Values	In HDR mode: OFF, 1, 2, 4 In other RFE modes: OFF, 1, 2, 4, 8, 16, 32, 64, 128, 256, 512 and 1024 1 Equivalents to decimation off.
Input Data Type	Integer | Character
Query Response	In HDR mode: 1, 2, 4 In other RFE modes: 1, 2, 4, 8, 16, 32, 64, 128, 256, 512 and 1024
Output Data Type	Integer
*RST State	1
Examples	`:DEC 16` `:SENSE:DEC OFF`

[:SENSe]:FREQuency:CENTer

This command sets or queries the tuned center frequency of the WSA.

The frequency resolution varies depending on the RFE modes of operation. ZIF, SH, SHN, DD and IQIN signal paths utilize the WB ADC; thus, the frequency resolution is 100 kHz. For those four receiver modes, the resolution can be down to the nearest 1 Hz resolution (±0.23 Hz) using [:SENSe]:FREQuency:SHIFt command. While for HDR receiver mode, the frequency resolution is 100 kHz without further frequency shifting capability. Refer to RF Receiver Front-End (page 17) for more details.

For example, the system is in ZIF mode, to tune to a frequency of 2441.16 MHz require the sending of two commands: [:SENSe]:FREQuency:CENTer 2441.1 MHz and [:SENSe]:FREQuency:SHIFt 6 KHz. The set values can be verified by querying. If a valid frequency with an inappropriate resolution is set, the frequency value will be rounded down to the nearest valid resolution, no error is set.

In addition, depending on the product models, the allowable range of programmable frequencies varies. Check with your product's data sheet. For example, WSA5000-108 and 208 have a range of 0.1 to 8 GHz, while WSA5000-220 has 0.1 to 20 GHz.

Syntax	[:SENSe]:FREQuency:CENTer <NRf [unit]> [:SENSe]:FREQuency:CENTer? [MAX | MIN]
Parameters	<center frequency [unit]>
Input Data Type	Double [character]
Allowable Values	***Varies depending on the product model***
Query Response	<integer>
Default I/O Unit	Hz
*RST State	2400 MHz

Examples	`:FREQ:CENTER 2441.5 MHz`
	`SENSE:FREQ:CENT 200000000`
	`:FREQ:CENT 2.01 GHZ`
	`SENSe:FREQ:CENTer?`

[:SENSe]:FREQuency:LOSCillator?

This command queries the frequency to be set for the external local oscillator (LO) 1 or 2 in corresponding to current the WSA's center frequency.

Note: This command ONLY works with WSA5000s that have the external LO option (see :SYSTem:OPTions? command). And when a WSA5000 supports the external LO input mode, the RFE's IQIN mode will not be available. Issuing :INPut:MODE IQIN will result in a SCPI error being returned. Please see "The Use of ThinkRF Products with External Local Oscillators" AppNote for more important details.

Syntax	[:SENSe]:FREQuency:LOSCillator? <1 | 2>
Parameter	None
Query Response	<integer>
Default Output Unit	Hz
Example	`:FREQ:LO? 2`

[:SENSe]:FREQuency:SHIFt

This command sets or queries the frequency shift value. A negative shift value corresponding to a left shifting.

This command is also used in additional to [:SENSe]:FREQuency:CENTer to fine tune the WSA down to 1 Hz resolution.

Note: Frequency shift mode is not available for some RFE modes of operation. Also, when enabled, it would affect the data output format of some RFE modes. See Table 2: Radio RFE Modes and DSP Data Output Formats (page 17).

Syntax	[:SENSe]:FREQuency:SHIFt <NRf [unit]> [:SENSe]:FREQuency:SHIFt? [MAX | MIN]
Parameters	<Frequency [unit]>
Input Data Type	Float [character]

Allowable Values	-62.5 – 62.5 MHz
Query Response	<integer>
Default I/O Unit	Hz
*RST State	0 Hz

Examples	`:FREQ:SHIF -10.5 MHz`
	`SENSE:FREQ:SHIFT 20000000.0`
	`SENSe:FREQ:SHIFT?`
	`FREQ:SHIFT? MAX`

[:SENSe]:FREQuency:RESolution?

This command queries the frequency resolution value.

Syntax	[:SENSe]:FREQuency:RESolution?
Parameter	None
Query Response	<integer>
Default Output Unit	Hz
Example	`:FREQ:RES?`

[:SENSe]:LOCK:REFerence?

This command queries the lock status of the PLL reference clock in the digital card.

Syntax	[:SENSe]:LOCK:REFerence?
Parameter	None
Query Response	0 | 1 1 Reference PLL is locked 0 Reference PLL is not locked
Data Type	Integer
Example	`LOCK:REF?`

[:SENSe]:LOCK:RF?

This command queries the lock status of the RF VCO (Voltage Control Oscillator) in the RFE.

Syntax	[:SENSe]:LOCK:RF?
Parameter	None
Query/Response	0 | 1 1 RF PLL is locked 0 RF PLL is not locked
Data Type	Integer
Example	`LOCK:RF?`

OUTput Commands

:OUTput:IQ:MODE

This command sets or queries the WSA5000's IQ output path to use the digitizer section for data output or the direct output to the IQ connector port of the WSA.

Important Notes:

- When the CONNector option is selected:

+ all commands affecting the digitizer data path of the WSA5000 will not apply, these include TRACe Commands, level triggering (:TRIGger:LEVel) and digital signal processing ([:SENSe]:DECimation and :SWEep:ENTRy:DECimation, [:SENSe]:FREQuency:SHIFt and :SWEep:ENTRy:FREQuency:SHIFt).

+ no VRT context packets will be sent out from the WSA.

+ when use with :INPut:MODE SH or SHN, the spectral inversion solution as suggested in Table 36 is required depending on the frequency input. The spectral inversion indicator is available through the GPIO port. Contact ThinkRF's Support for further details.

- The CONNector mode only works on certain WSA5000 model (ex WSA5000-108 does not support this mode). Verify that your product support this method. A system error will be returned if the model does not accept this option.

When using CONNector option with :INPut:MODE SH or SHN, it is highly recommended to use the output Q channel for the

Syntax	:OUTput:IQ:MODE :OUTput:IQ:MODE?
Parameter/Response	CONNector | DIGitizer
Data Type	Character
*RST State	DIGitizer
Examples	`:OUT:IQ:MODE CONNECTOR` `:OUTPUT:IQ:MODE?`

:OUTput:IQ:CONNector:INVersion?

This query only command is used to determine if a spectral inversion is required on the data output at the IQ OUT connector at a given frequency, regardless of the DSP mode enabled. See Table 36 for more information on spectral inversion.

This command is not available for the sweep entry subset of commands. However, this command can be used iteratively during say initialization stage to query the frequency range of interest. The results can than be stored in a look-up table, for example.

Syntax	:OUTput:IQ:CONNector:INVersion? [NRf [unit]]
Parameters	Optional [center frequency [unit]]

Input Data Type	[Double [character]]
Default Input Unit	Hz
Allowable Values	***Varies depending on the product model***
Query Response	1 | 0 where 1 is equivalent to the inversion is required
*RST State	N/A
Examples	`:OUT:IQ:CONN:INV? 2441.5 MHz` `OUT:IQ:CONN:INV?`

TRIGger Commands

:TRIGger:TYPE

This command sets or queries the type of trigger event. Setting the :TRIGger:TYPE to NONE is equivalent to disabling the trigger execution, while setting to any other type will enable the trigger engine.

The LEVel trigger type is condition by the start and stop frequencies range and the amplitude level. See the :TRIGger:LEVel command.

The PULSe and WORD trigger types belong to the external synchronization trigger through a GPIO port (see External Triggering, page 23).

Syntax	:TRIGger:TYPE LEVel | PERiodic | PULSe | WORD | NONE :TRIGger:TYPE?
Parameter/Response	LEVEL | PERIODIC | PULSE | WORD | NONE
Data Type	Character
*RST State	NONE
Examples	`:TRIG:TYPE LEVEL` `:TRIG:TYPE?`

:TRIGger:LEVel

This command sets or queries the frequency range and amplitude of a frequency domain level trigger. If the sampled signal amplitude exceeds the defined trigger level at any single sample within the defined frequency range then the trigger will occur and the associated IQ data will be stored.

The frequency range encompasses all FFT bins of which their center frequencies are within the range defined by START and STOP. The defined START and STOP frequencies may exceed, but only affect, the range defined by the IBW (with consider of the DDC decimation) centered around the [:SENSe]:FREQuency:CENTer value.

Refer to the Frequency Domain Triggering section for more information.

Syntax	:TRIGger:LEVel <NRf [unit]>,<NRf [unit]>,<NR1 [unit]> :TRIGger:LEVel?
Parameters	<start>,<stop>,<level>

Input Data Type	Comma separated values with: Frequency: Double [character] Level: signed integer value
Allowable Values	Frequency: See [:SENSe]:FREQuency:CENTer Levels: Dependent on the attenuation. When the attenuator is + off, -30 dBm maximum + on, -10 dBm maximum
Query Response	<integer>,<integer>,<signed integer>
Default I/O Units	Hz,Hz,dBm
*RST State	N/A (Trigger is off)

Examples	`:TRIG:LEVEL 2000 MHZ,2100 MHZ,-70 DBM`
	`:TRIG:LEVEL 15000000,15050000,-50`
	`:TRIG:LEVEL?`

:TRIGger:PERiodic

Further information will be provided in a future revision of this document.

:TRIGger:STATus?

This command returns the status of the current enabled trigger as to whether it is pending or has occurred. It is cleared once the data buffer is cleared out.

Syntax	:TRIGger:STATus?
Parameter	None
Query Response	0 | 1 1 Trigger event occurred 0 No trigger event
Data Type	Integer
*RST State	0

TRACe Commands

A "trace capture" consists of a set of continuous data samples, ranging from 128 samples to a maximum determined by the WSA5000 version (see :TRACe:BLOCk:PACKets and :TRACe:SPPacket). Each data word is 32-bit wide, arranged differently depending on the :INPut:MODE and :TRACe:FORMat (see VRT's Data Payload Format, page 39).

ThinkRF's WSA5000 data packet returned through a network is complied with the industry standard VRT protocol. Therefore, every data packet returned is encapsulated with a VRT header and a VRT trailer. In addition, the VRT packet format sets a limit on the maximum number of samples per packet. Refer to the "Receiver Context Class" subsection of the VITA-49 Radio Transport Protocol section for further details on the VRT packet organization.

To do a single **block** capture of continuous data, the total number of samples captured is determined by the number of samples per packet (:TRACe:SPPacket) and the number of packets per block (:TRACe:BLOCk:PACKets). When the block data capture command (:TRACe:BLOCk[:DATA]?) is issued, the WSA5000 will capture and store the total number of samples into a buffer. Hence, the samples within a single block capture is continuous from one packet to the other, but not necessary between successive block capture commands issued.

In **streaming** mode, the number of samples per packet (:TRACe:SPPacket) must be set to determine the size of each packet coming back. The samples from one packet to another will be continuous until the sample loss indicator (aka overflow indicator) is detected within the trailer of the data packet. When this indicator is high in the current VRT packet, it indicates that data overflow occurs **after** the current captured packet, not within the packet. In other words, the samples of the immediate packet following after the current packet that has the sample loss indicator bit high are not continuous from those of the current packet.

Note: The :DECimation command can be used to slow down the capture rate, thus, effectively lowers the rate of discontinuity between packets to provide contiguous data stream of data.

The WSA5000 can stored up to 32 MSa continuous data.

:TRACe:SPPacket

This command sets or queries the number of Samples Per Packet (SPPacket). In block capture mode, it is used in conjunction with the :TRACe:BLOCk:PACKets command to set the total number of samples to capture.

The upper bound of the samples is limited by the VRT's 16-bit Packet Size field less the VRT's headers and any 32-bit word wide optional fields (i.e. Class ID, Timestamps, and trailer). However, the 32-bit VRT words must be a multiple of 16. Therefore, the maximum SPP is limited by “65520 (or 2^{16} – 16) * number of samples-per-word”. The SPP values must also be a multiple of “16 * number of samples-per-word”. Table 38 summarizes the minimum and maximum SPP sizes and the required multiple values for different data output format.

Table 38: Minimum and Maximum SPP Sizes and the Required Multiples for Different Data Output Format

Format	Samples-per-word	Minimum Size	Maximum Size	Required Multiples
{$I_{14}Q_{14}$}	1	128	65520	16
{I_{14}}	2	128 * 2 = 256	65520	16 * 2 = 32
{I_{24}}	1	128	65520	16
{PSD_8}	4	128 * 4 = 512	65520 * 4 = 262080	16 * 4 = 64

Syntax	:TRACe:SPPacket <integer> :TRACe:SPPacket? [MAX | MIN]
Parameter	<integer> [MAX | MIN] for query
Input Data Type	Integer | Character
Allowable Values	See Table 38 above for the Min and Max values and the **multiples requirement**
Query Response	<integer>
*RST State	1024
Examples	`:TRACE:SPP 4096` `:TRAC:SPP?`

:TRACe:BLOCk[:DATA]?

This command will start the single block capture and the return of all trace packets set by :TRACe:BLOCk:PACKets command, with each packet of the size set through :TRACe:SPPacket command. The data within a single block capture trace is continuous from one packet to the other, but not necessary between successive block capture commands issued.

Syntax	:TRACe:BLOCk[:DATA]?
Parameter	None
Response	Control port 37001: empty string Data port 37000: <NRr>
Output Data Type	Hexadecimal bytes
*RST State	N/A
Examples	`:TRACE:BLOCK?` `:TRACE:BLOCK:DATA?`

Note: The status of the query will be return through the control port 37001 as usual, however the data will be returned through the data port 37000. Once the :TRACe:BLOCk[:DATA]? command is issued, a block of SPP * PACKets of data will be returned. In other words, :TRACe:BLOCk[:DATA]? needs to be sent only once to get SPP * PACKets block of data.

The returned data in each VRT packet is presented in continuous hexadecimal chunk, as shown here:

```
Response <NRr> ::= <VRT header bytes>{<data payload bytes>}
                   [<4 bytes VRT trailer>]
```

Further description on the VRT data output formats can be found in the VRT's IF Data Packet Class section, page 37.

:TRACe:BLOCk:PACKets

This command sets or queries the total number of packets set in the WSA. The maximum is limited by the storage capacity of a WSA5000 and the samples per packet (SPP) size set through :TRACe:SPPacket. Therefore, when :TRACe:BLOCk:PACKets?

MAX query command is sent, the returned value will vary depending on the SPP value of a WSA and the data output format. For example, the WSA5000 has 128 MBytes storage capacity, if SPP is 32768 with $I_{14}Q_{14}$ output format, then the maximum packet size is 1023 (or 128 MB / (4 bytes-per-sample * (32768 + 6))). If I_{14} is the output format, then the maximum is 2047 (or 128 MB * / (2 bytes-per-sample * (32768 + 6))).

In single block capture mode, this command is used in conjunction with the :TRACe:SPPacket command to set the total number of samples to capture. In other words, the data from one packet to the next within a single block capture mode is continuous.

Syntax	:TRACe:BLOCk:PACKets <integer> :TRACe:BLOCk:PACKets? [MAX | MIN]
Parameter	<integer> [MAX | MIN] for query1 – (WSA's max storage storage capacity ÷ (# bytes-per-sample * (SPP value + 6 Header and trailer words)))
Input Data Type	Integer | Character
Allowable Values	1 – (WSA's max storage storage capacity ÷ (# bytes-per-sample * (SPP value + 6 Header and trailer words)))
Query Response	<integer>
*RST State	1
Examples	`:TRACE:BLOC:PACK 100` `:TRACE:BLOCK:PACK?`

:TRACe:STReam:STARt

This command begins the execution of the stream capture. It will also initiate data capturing. Data packets will be streamed (or pushed) from the WSA5000 whenever data is available.

Through the sending of a VRT Extension Context Packet carrying the ID value, the use of an ID in this command is to indicate the beginning of new data packets belonging to a new stream start. Even though the start ID value is optional, a VRT Extension Context Packet with the New Stream Start ID (page 37) value will **always** be sent out after this command is received and before data packets of the new stream become available. When no ID value is provided, the default ID value 0 is returned in the Context Packet.

Note: Once :TRACe:STReam:STARt is issued, the WSA will not accept any setting changes. Changes can be sent after :TRACe:STReam:STOP command is issued.

Syntax	:TRACe:STReam:STARt [ID]
Parameter	Unsigned 32-bit integer
Response	None
*RST State	0 (Stream stopped)
Examples	`:TRAC:STREAM:START 1` `:TRACE:STR:START`

:TRACe:STReam:STOP

This command stops the stream capture. After receiving the command, the WSA system will stop when the current capturing VRT packet is completed with the required samples (as opposed to :SYSTem:ABORt).

Note: After :TRACe:STReam:STOP command is issued, :SYSTem:FLUSh command should be issued as well as to clear up any data remained in the internal memory.

Syntax	:TRACe:STReam:STOP
Parameter/Response	None
*RST State	N/A (Stream stopped)
Examples	`:TRACE:STREAM:STOP` `:TRAC:STREAM:STOP`

:TRACe:FORMat

This command sets or queries the WSA data capture output format, whether in time domain (TD) or frequency domain. TD data refers to the IQ or I only samples depending on the selected RFE mode (:INPut:MODE or :SWEep:ENTRy:MODE) and [:SENSe]:FREQuency:SHIFt or :SWEep:ENTRy:FREQuency:SHIFt setting. Frequency domain data refers to the logPSD (LPSD) data. LPSD is the logarithm of the PSD data computed on the TD samples of the selected RFE mode in order to reduced the $I_{fft}^2 + Q_{fft}^2$ results to 8-bit wide:

$$logPSD = 10 * \log_{10}(I_{fft}^2 + Q_{fft}^2)$$

Notes:

- The capture output format along with the RFE mode (:INPut:MODE or :SWEep:ENTRy:MODE) affect the 32-bit word packing method due to the different data width. See VRT's IF Data Packet Class, page 37.

- logPSD is not available for HDR receiver mode.

Syntax	:TRACe:FORMat :TRACe:FORMat?
Parameter/Response	TD | LPSD
Data Type	Character
*RST State	TD
Examples	`:TRAC:FORM LPSD` `:TRACE:FORMAT?`

SWEep Commands

Note: Currently, only one single sweep list is supported. Thus, some description on list in this section might not apply. For example, the string identifier is not needed yet, neither is list editing as there is only one list. The entries, however, can be configured as described.

A sweep control setup consists of defining one or more sweep lists and one or more entries for each list. The sweep execution is controlled by issuing the commands (such as start, stop or resume) listed under :LIST.

A sweep list can be thought of as being similar to a spreadsheet or table where the columns define the different specific capture engine configurations (such as :ANTenna, :FREQuency and :DECimation), and the rows as sweep entries with each consisting of a sweep frequency or range and its associated capture engine configurations.

A :SWEep:LIST is created and identified using a unique string identifier set by the user. A list may be edited, deleted and/or executed using the :SWEep:LIST command set. Each list is executed indefinitely or a finite number of time as determined by the :ITERations command.

More information will be provided in the future revision of this document for multiple lists handling.

The :SWEep:ENTRy commands provide the ability to define the capture engine configurations for each sweep entry including the equivalent of :INPut, :SENSe and :TRIGger commands. There may be any number of entries in a sweep list for up to 500. Sweep entries are identified by an index number and may be inserted, edited and/or deleted like rows in a table or spreadsheet. A sweep entry is created by using either :NEW or :COPY and :SAVE command. The entry will not be part of a list until :SAVE is issued.

If trigger is defined for an entry, captured data is returned only if a trigger event occurred. Otherwise, when the :DWEll time is reached, the trigger is aborted and the next sweep entry will be executed.

During sweeping, the WSA internal buffer might be overflown, at which point the sweep engine will pause. The engine will resume sweeping once there are enough space for the next “block” of data or more.

The engine will stop when the iterations have been reached or either a :SYSTem:ABORt or :SWEep:LIST:STOP command has been issued.

Notes:

- Unlike with [:SENSe]:FREQuency:CENTer, the center frequency command of a sweep entry can take a frequency range and the step size as the parameters.

- Unlike :TRACe:BLOCk capture, sweep mode data packets, whether VRT context or digitized data, are “streamed” (similar to :TRACe:STReam). As soon as :SWEep:LIST:STARt command is issued, this will initiate also the data capturing and data packets will be “pushed” from the WSA5000 when available.

- When sweep is stopped, the WSA will retain the settings of the last performed sweep entry when :STOP command is received and executed. Any non-sweep commands can then be operated on the WSA. When the :SWEep is resumed (:STARt), the settings as per the sweep entries are executed.

- When the WSA is sweeping, any non-sweep commands sent will resulted in an error and are not executed. The sweep will not be affected and keep on running. However, sweep related settings can still be changed while sweep is running.

:SWEep:LIST:STARt

This command begins the execution of the current sweep list from the first entry.

This command will also initiate data capturing. Data packets will be streamed (or pushed) from the WSA5000 whenever it is available.

Through the sending of a VRT Extension Context Packet carrying the ID value, the use of an ID in this command is to indicate the beginning of new data packets belonging to a new sweep start. Even though the start ID value is optional, a VRT Extension Context Packet with the New Sweep Start ID (page 37) value will **always** be sent out after this command is received and before data packets of the new sweep become available. When no ID value is provided, the default ID value 0 is returned in the Context Packet.

Syntax	:SWEep:LIST:STARt [ID]
Parameter	Unsigned 32-bit integer
Response	None
*RST State	0 (Sweep stopped)
Examples	`:SWEEP:LIST:STAR` `:SWE:LIST:START`

:SWEep:LIST:STOP

This command stops the sweeping and stores the entry index where it is stopped. The WSA retains the settings of the last performed sweep entry when :STOP command is executed.

Syntax	:SWEep:LIST:STOP
Parameter/Response	None
*RST State	N/A (Sweep stopped)
Examples	`:SWEEP:LIST:STOP` `:SWE:LIST:STOP`

Note: :SWEep:LIST:STOP should be issued to clear the WSA5000's data buffer of any data that has not been sent from the WSA5000 prior to setting up any other capturing process.

:SWEep:LIST:ITERations

This command sets or queries the number of times the sweep list is repeated.

Syntax	:SWEep:LIST:ITERations <integer> :SWEep:LIST:ITERations? [MAX | MIN]
Parameter	<integer> [MAX | MIN] for query
Allowable Values	0 – 4294967295 (or 2^{32}-1) 0 := infinity
Query Response	<integer>
I/O Data Type	Integer
*RST State	0
Examples	`:SWEEP:LIST:ITER 10` `:SWE:LIST:ITER? MAX`

:SWEep:LIST:STATus?

This query returns the current status of the sweep engine.

Syntax	:SWEep:LIST:STATus?
Query Response	RUNNING | STOPPED
Data Type	Character
*RST State	STOPPED
Examples	`:SWEEP:LIST:STATUS?` `:SWE:LIST:STAT?`

:SWEep:ENTRy:NEW

This commands will populate all the capture engine configurations under :SWEep:ENTRy with default values. No new entry is created until :SWEep:ENTRy:SAVE command is issued.

Syntax	:SWEep:ENTRy:NEW
Parameter/Response	None
*RST State	N/A
Examples	`:SWEEP:ENTRY:NEW`

:SWEep:ENTRy:COPY

This commands will copy and populate all the capture engine configurations under :SWEep:ENTRy with values from the sweep entry of the specified index. No new entry is created until :SWEep:ENTRy:SAVE command is issued and any changes will not affect the existing entry.

Syntax	:SWEep:ENTRy:COPY <integer>
Parameter	Sweep entry integer index
Query Response	None
Allowable Values	If :COUNt? returns non-zero, 1 to :COUNt? value If :COUNt? returns zero, an execution error is returned
*RST State	N/A
Examples	`:SWEEP:ENTR:COPY` `:SWE:ENTRY:COPY`

:SWEep:ENTRy:SAVE

This command saves a new entry into the current editing list with all the current capture engine configurations under :SWEep:ENTRy. The saving is done by inserting either the new entry **before** the specified index value or to the end of the list when no index value is given.

When saved, a new entry is given an index value. Index value starts from 1. When an index value is specified along with the :SAVE command, the new entry will take the index of that value and all other following indexes will be incremented by one accordingly, just as rows in a spreadsheet. Otherwise, the new index will be one up from the index of the last sweep entry in the list.

When there are no existing entries and an index value other than 1 is specified, an error will be returned. Similarly for non-existing index location except if the index value is equal to the value returned by :SWEep:ENTRy:COUNt? plus one.

Syntax	:SWEep:ENTRy:SAVE [integer]
Parameter	[Entry index value]
Input Data Type	Integer
Allowable Values	:COUNt? value + 1
*RST State	N/A
Examples	`:SWEEP:ENTR:SAVE` `:SWE:ENTR:SAVE 5`

:SWEep:ENTRy:DELETE

This commands delete one or all the entries. When an entry is deleted, the following indexes if existed will be reduced by one accordingly, just as rows in a spreadsheet.

Syntax	:SWEep:ENTRy:DELETE <integer> | ALL
Parameter	<Entry index value> | ALL
Input Data Type	Integer | Character
Allowable Values	1 to COUNt? value
*RST State	N/A
Examples	`:SWEEP:ENTR:DELETE 5` `:SWE:ENTR:DELETE ALL`

:SWEep:ENTRy:READ?

This query command returns the current configuration settings of a sweep entry.

Syntax	:SWEep:ENTRy:READ? <integer>
Parameter	[Entry index value]
Input Data Type	Integer
Allowable Values	1 to COUNt? value
Query Response	<integer>,<{ingeger | char}> :== <RFE mode>,<freq start>,<freq stop>,<freq step>,<freq shift>, <decimation>, <attenuator>,<IF gain>,<HDR gain>, <SPPacket>,<packets>,<dwell:second>,<dwell:microsecond>, <trigger type: NONE | PULSe | WORD | <LEVel,freq start,freq stop, amplitude>>
Output Data Type	Comma separated integer and character values
*RST State	N/A
Examples	`:SWEEP:ENTR:READ? 5` `:SWE:ENTR:READ? 1`

:SWEep:ENTRy:COUNt?

This query command returns the number of entries available in a list.

Syntax	:SWEep:ENTRy:COUNt?
Query Response	<integer>
Output Data Type	Integer
*RST State	N/A
Examples	`:SWEEP:ENTR:COUNT?`

:SWEep:ENTRy:FORMat

Refers to the :TRACe:FORMat section (page 75) for the definition of this command.

Examples	`:SWEEP:ENTRY:FORMAT LPSD` `:SWE:ENTR:FORM?`

:SWEep:ENTRy:MODE

Refers to the :INPut:MODE section (page 62) for the definition of this command.

Examples	`:SWEEP:ENTRY:MODE ZIF` `:SWE:ENTR:MODE?`

:SWEep:ENTRy:ATTenuator

Refers to the :INPut:ATTenuator section (page 63) for the definition of this command.

Examples
```
:SWEEP:ENTRY:ATTENUATOR ON
:SWEEP:ENTR:ANT?
```

:SWEep:ENTRy:FILTer:PRESelect

Refers to the :INPut:FILTer:PRESelect section (page 63) for the definition of this command.

Examples
```
:SWEEP:ENTR:FILT:PRES ON
:SWEEP:ENTRY:FILTER:PRES?
```

:SWEep:ENTRy:GAIN:IF

Refers to the :INPut:GAIN:IF section (page 63) for the definition of this command.

Examples
```
:SWEEP:ENTR:GAIN:IF -10
:SWEEP:ENTRY:GAIN:IF?
```

:SWEep:ENTRy:GAIN:HDR

Refers to the :INPut:GAIN:HDR section (page 64) for the definition of this command.

Examples
```
:SWEEP:ENTR:GAIN:HDR -10
:SWEEP:ENTRY:GAIN:HDR?
```

:SWEep:ENTRy:DECimation

Refers to the [:SENSe]:DECimation section (page 65) for the definition of this command.

Examples
```
:SWEEP:ENTR:DEC 16
:SWEEP:ENTRY:DEC?
```

:SWEep:ENTRy:FREQuency:CENTer

This command or query defines the center frequency or a range of center frequencies to sweep. When a range is provided, the sweep will step through the center frequencies with the value provided by :SWEep:ENTRy:FREQuency:STEP.

Syntax	:SWEep:ENTRy:FREQuency:CENTer <NRf [unit]>[,<NRf [unit]>] :SWEep:ENTRy:FREQuency:CENTer?
Parameter	<start freq [unit]>[,<stop freq [unit]>]
Input Data Type	Double [character] | Comma separated doubles [character]
Allowable Values	**Varies depending on the product model**
Query Response	<integer>,<integer>
Default I/O Units	Hz
*RST State	2400 MHz, 2480 MHz

Examples	`:SWEEP:ENTRY:FREQ:CENT 0,10 GHZ`
	`:SWE:ENTRY:FREQ:CENT 2400 MHZ,6 GHZ`
	`:SWE:ENTR:FREQ:CENT 2400000000`
	`:SWEEP:ENTRY:FREQ:CENTER?`

:SWEep:ENTRy:FREQuency:STEP

This command or query defines the frequency step size for the sweep center frequency range specified by :SWEep:ENTRy:FREQuency:CENTer command. If a range is not given, the step size is ignored.

Syntax	:SWEep:ENTRy:FREQuency:STEP <NRf [unit]> :SWEep:ENTRy:FREQuency:STEP?
Parameter	<freq [unit]>
Input Data Type	Double [character]
Allowable Values	0 – 10 GHz (values are WSA5000 version dependent)
Query Response	<integer>
Default I/O Units	Hz
*RST State	100 MHz

Examples	`:SWEEP:ENTRY:FREQ:STEP 10.5 MHZ`
	`:SWE:ENTRY:FREQ:STEP 4000 KHZ`
	`:SWEEP:ENTR:FREQ:STEP 100000000`
	`:SWEEP:ENTR:FREQ:STEP?`

:SWEep:ENTRy:FREQuency:SHIFt

Refers to the [:SENSe]:FREQuency:SHIFt section (page 67) for the definition of this command.

Examples	`:SWEEP:ENTR:FREQ:SHIFT 25 MHZ` `:SWEEP:ENTRY:FREQ:SHIF?`

:SWEep:ENTRy:TRIGger:LEVel

Refers to the :TRIGger:LEVel section (page 70) for the definition of this command.

Examples	`:SWEEP:ENTR:TRIG:LEV 2400 MHZ,2900 MHZ,-60` `:SWEEP:ENTRY:TRIGGER:LEVEL?`

:SWEep:ENTRy:TRIGger:TYPE

Refers to the :TRIGger:TYPE section (page 70) for the definition of this command.

Examples
```
:SWEEP:ENTR:TRIG:TYPE LEVEL
:SWEEP:ENTRY:TRIG:TYPE?
```

:SWEep:ENTRy:PPBlock

This command (where PPBlock is defined as Packets per block) has the same functionality as the :TRACe:BLOCk:PACKets command since at each sweep frequency step of an entry, a block of data can be captured.

Refers to the :TRACe:BLOCk:PACKets section (page 73) for the definition of this command.

Examples
```
:SWEEP:ENTR:PPB 10
:SWEEP:ENTRY:PPB?
```

:SWEep:ENTRy:SPPacket

Refers to the :TRACe:SPPacket section (page 72) for the definition of this command.

Examples
```
:SWEEP:ENTR:SPP 16384
:SWEEP:ENTRY:SPP?
```

:SWEep:ENTRy:DWELl

This command or query defines the maximum amount of time to wait for the trigger of a sweep entry to occur, after which the trigger is aborted and the next sweep entry, if existed, will run. However, when the required amount of data has been captured before the dwell time has been reached, the sweep engine will move onto the next entry.

Note that, the default dwell time is 0 second, 0 microsecond. This is equivalent to an infinite dwell time. In this case, the sweep engine will move on as soon as the current data capture amount has been met (as explained in the previous paragraph).

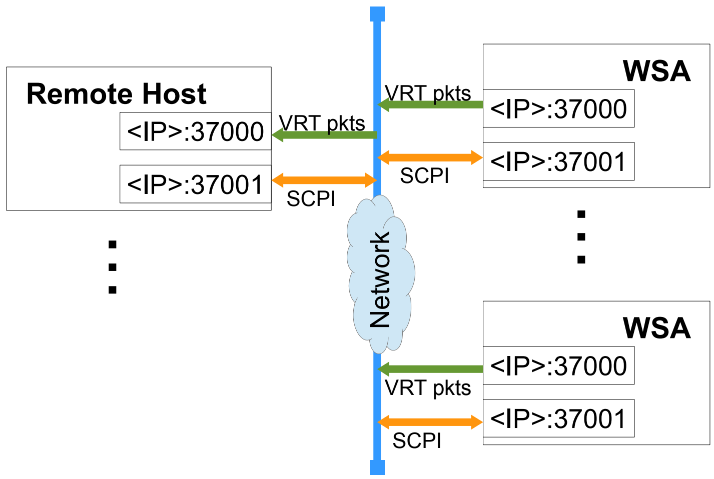
When the trigger type is NONE, dwell time is ignored.

Syntax	:SWEep:ENTRy:DWELl <integer>[,<integer>] :SWEep:ENTRy:DWELl?
Parameter	<second>[,<microsecond>]
Allowable Values	0 – 4294967295 (or $2^{32} – 1$) 0,0 := infinity
Query Response	<integer>,<integer>
I/O Data Type	Integer | Comma separated ingeters
*RST State	0,0 (infinite dwell time)

Examples	`:SWEEP:ENTR:DWEL 5,30`
	`:SWEEP:ENTR:DWELL 2`
	`:SWEEP:ENTR:DWELL?`

Appendix A: Connecting to WSA

ThinkRF's WSAs are network ready devices conveying control commands and data using TCP/IP protocol. Each WSA receives SCPI commands and sends query responds over port 37001, and sends VRT context and data packets over port 37000, as illustrated in the following figure:



A WSA, when powered up, will have a dynamic or preassigned static IP address, which when bind with a port will form a network socket. To successfully establish a connection to a WSA, **both** <IP>:37000 and <IP>:37001 sockets must be created one right after the other, the order is not important.

In addition, refer to the "Connecting to the WSA5000" of the *WSA5000 User Guide* (v3.6 or later) for more information on how to connect to WSA5000 and to determine its IP address.

Appendix B: Protocol for Discovering WSA

ThinkRF uses a simple broadcast UDP protocol for discovering any WSAs available on the same local network as the host computer. This protocol can not be used to find any WSAs on a different network.

The remote host computer would first send out a UDP message of broadcast type to port 18331. The message contains a query request code followed by query discovery version in big-endian order as follows:

<request code><discovery version>

where each field is:

Name	Data Type	Length	Required Value
request code	32-bit unsigned integer	1	0x93315555
discovery version	32-bit unsigned integer	1	2

The discovery version is used to determine how to parse the response message. Note that the <> bracket is for clarity of the explanation purpose only, not to be included in the message.

A WSA with the discovery version 2 would respond with the following data:

<response code><discovery version><WSA model><WSA S/N><firmware version>

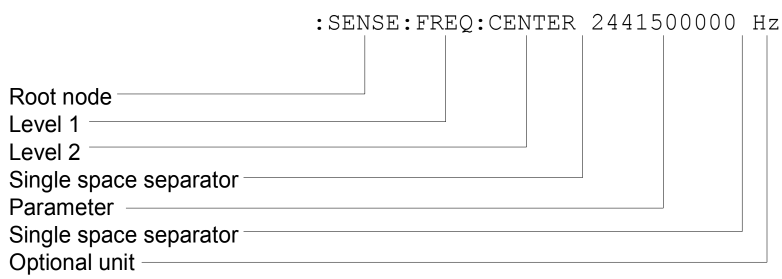
where each field is:

Name	Data Type	Length	Response Value
response code	32-bit unsigned integer	1	0x93316666
discovery version	32-bit unsigned integer	1	2
WSA model	ASCII character, nul-padded	16	WSAXXXX-XXX (ex: WSA5000-220)
WSA S/N	ASCII character, nul-padded	16	XXXXXX-XXX (ex: 120600-020)
firmware version	ASCII character, nul-padded	20	vX.X.X (ex: v1.0.0)

The IP address of a WSA can be retrieved from the responding socket. The WSA may be identified by matching the serial number (S/N) in the response message with the S/N on the label of the WSA.

Appendix C: SCPI Command Syntax

Each SCPI command consists of a root node, one or more lower level nodes, follow by applicable parameters and separators:



Entering Commands

SCPI commands have both a long and short version, such as :SOURCE and :SOUR. The SCPI interface responds to either version, but will not respond to variations of either version. The interface does not differentiate between upper-case and lower-case letters but only the long or short form of a command.

An example correct and incorrect SCPI entry format for :SOURce command:

	Command Entry		
Correct Entry	:SOURCE	:SOURce	:source
	:SOUR	:sour	
Incorrect Entry	:SOU	:SOURC	
	:sourc		

Note: At the end of each SCPI command string, whether a single command or multiple commands separated by semicolons “;”, *a new line-feed or carriage return is required.* Example in C: “:FREQ 2400 MHZ\n” or “FREQ 2400 MHZ;INP:ANT 1\n”.

Notation

<table>
<tr><th>Notation</th><th>Description</th></tr>
<tr><td>:</td><td>Links command keywords together</td></tr>
<tr><td>;</td><td>Separates multiple commands entered together on a single program message</td></tr>
<tr><td>single space</td><td>Uses to separate a parameter from a command or unit from a parameter</td></tr>
<tr><td>,</td><td>Uses to separate multiples parameters of a command</td></tr>
<tr><td>[]</td><td>Uses to optionally enclose zero or more parameters</td></tr>
<tr><td>{.} or {.}*</td><td>The enclosed item maybe included zero or more times</td></tr>
<tr><td>{.}+</td><td>The enclosed items occurs one or more times</td></tr>
<tr><td>{.|.|.}</td><td>One and only one of the two or more enclosed items separated by | maybe included</td></tr>
<tr><td><></td><td>Uses to enclose required parameter descriptions</td></tr>
<tr><td>?</td><td>Indicates query command, use where applicable</td></tr>
<tr><td>|</td><td>Indicates “or” and is used to separate alternative parameter options</td></tr>
<tr><td>::=</td><td>Means “is defined as”</td></tr>
</table>

Parameter types

This section defines different SCPI parameter data type.

<table>
<tr><th>Parameter Type</th><th>Description</th></tr>
<tr><td><boolean></td><td>ON | OFF | 1 | 0
Boolean parameters are always returned as 1 or 0 in NR1 format by query commands</td></tr>
<tr><td><integer>
<int></td><td>Unsigned integer of NR1 format
Ex: 1 or 3432</td></tr>
<tr><td><NR1></td><td>Signed integer without a decimal point (implied radix point)
Ex: -25 or 0</td></tr>
<tr><td><NR2></td><td>Signed number with an explicit radix point
Ex: -1.234 or 1.0 or 0.0</td></tr>
<tr><td><NR3></td><td>Scaled explicit decimal point numeric value with and an exponent
Ex: 2.73e+2 or 2.351e2</td></tr>
<tr><td><NRf></td><td><NR1>|<NR2>|<NR3></td></tr>
<tr><td><NRr></td><td>Non-decimal numeric value such as hexadecimal, octal or binary</td></tr>
<tr><td><char>
<character></td><td>Character program data
Ex: MAXimum or MEDium</td></tr>
<tr><td><string></td><td>ASCII string surrounded by single or double quotes
Ex: “This is an example”</td></tr>
</table>

Default Units

Parameter	Default Unit
frequency	Hz
time	s or ns where applicable
voltage	V
absolute amplitude	dBm
relative amplitude	dB

Units other than the default may be specified. If units are not specified then the default units apply. Note the following examples, which are all equivalent.

Example	`:FREQ:CENTer 2441.5 MHz`
is equivalent to	`:FREQ:CENTer 2441500000`
is equivalent to	`:FREQ:CENTer 2441500000 Hz`
is equivalent to	`:FREQ:CENTer 244150 kHz`
is equivalent to	`:FREQ:CENTer 2441.5e6`

Appendix D: SCPI Status and Event Registers

The WSA's SCPI interface has a status and event reporting system that enables the user to handle device events. The interface conforms to IEEE Std 488.2-1987 and SCPI 1999.0. This section discusses these status registers, status register enable masks, event queues and event handling.

Status Byte Register (SBR)

The SBR is used to determine the specific nature of the event or condition. It is read by issuing a *STB? command. The contents of the SBR are clear by issuing either a *STB? or *CLS command.

Bits in the SBR will be set only when the corresponding bits in the Service Request Enable Register are set.

Bit	Name	Description
0	not used	This bit is not used and is always 0.
1	not used	This bit is not used and is always 0.
2	Error / Event Available (EAV)	This bit is set if there are any unread error or event in the System Error queue. It is read using the SYSTem:ERRor? query.
3	Questionable Register Summary	This bit is not used and is always 0.
4	Message Available (MAV)	This bit is set if there is any unread data in the Output queue.
5	Standard Event Status Bit (ESB)	This bit is set if there is any unread or non-cleared data in the Standard Event Status register.
6	Request Service	Summary of the Request Service register.
7	Operation Condition Register Summary	Summary of the Operation Status register

Standard Event Status Register (ESR)

The ESR is used to determine the nature of the status and error conditions. It is read by issuing a *ESR? command. The contents of the ESR are clear by issuing either a *ESR? or *CLS command.

Bits in the ESR will be set only when the corresponding bits in the Standard Events Status Enable Register are set.

Bit	Name	Description
0	Operation Complete (OPC)	Set to indicate that all pending operations are complete and the WSA5000 is ready to accept another command, or that query results are available.
1	Request Control (RQC)	This bit is not used and is always 0.
2	Query Error (QYE)	Set to indicate that a query has been made for which no

		response is available. Query errors have SCPI error codes from –499 to –400.
3	Device Dependent Error (DDE)	Set to indicate that a device-dependent error has occurred. Device-dependent errors have SCPI error codes from –399 to –300 and 1 to 32767.
4	Execution Error (E)	Set to indicate that a parameter exceeds its allowed range. Execution errors have SCPI error codes from –299 to –200.
5	Command Error (CME)	Set to indicate that a command error has occurred. Command errors have SCPI error codes from –199 to –100.
6	not used	This bit is always 0.
7	Power ON (PON)	Set once upon power-up. This bit has no effect on the Error / Event Available (EAV) bit in the Status Byte Register.

Operational Status (OSR) Register

The OSR is a 16-bit register that is used to determine the state of operation. It is read by issuing a :STATus:OPERation[:EVENt]? command.

Bit	Name	Description
0-3	not used	These bits are not used and is always 0.
4	Measuring (MEAS)	Set to indicate that a query has been made for which no response is available. Query errors have SCPI error codes from –499 to –400.
5-15	not used	These bits are not used and is always 0.

Output Queue

The WSA5000 has an Output FIFO Queue that is structured as a FIFO and holds the response messages to queries. The SBR's MAV bit is set when there are messages in the queue. The unread results of a previous command are cleared from the queue when a new command or query is received.

Error and Event Queue

The WSA5000 has an Error and Event FIFO Queue that holds up to 16 errors and events. It is queried using the :SYSTem:ERRor[:NEXT]? command. The *CLS command clears all entries from the queue.

Appendix E: SCPI Error Codes Used

Code	Message	Description
0	No error	
Command error, range [-199, -100]		
-144	Character data too long	The character data contained more than 12 characters.
-171	Invalid expression	The command syntax was incorrect.
Execution error, range [-299, -200]		
-200	Execution error	A generic execution error for which more specific information is not available.
-210	Trigger error	
-220	No matched module	The specific operation is not installed.
-221	Settings conflict	Indicates that a legal program data element was parsed but could not be executed due to the current device state
-222	Data out of range	A parameter was of the proper type but outside of the defined range for the specific command.
-223	Too much data	A parameter was received that contained more data than the device could handle.
-224	Illegal parameter value	A parameter was received that is NOT allowed for the particular command.
-230	Data corrupt or stale	Possibly invalid data; new reading started but not completed since last access.
-240	Hardware error	Indicates that a legal program command or query could not be executed because of a hardware problem in the device.
-241	Hardware missing	Indicates that a legal program command or query could not be executed because of missing device hardware.
Device specific error, range [-399, -300]		
-321	Out of memory	An internal operation needed more memory than that was available.
-330	Self test failed	
-340	Calibration failed	
-350	Query overflow	The SCPI remote interface error queue overflowed.
Query error, range [-499, -400]		
-410	Query INTERRUPTED	A condition causing an INTERRUPTED query error occurred
WSA5000 Specific, range [-999, -900]		
-901	No data	Read trace command issued while there is no data available.
-911	Please upgrade firmware	The current firmware needs upgrading.
-912	Invalid option license	The option could not be installed because of invalid license.

Appendix F: SCPI Commands Quick Reference

This section summarizes the SCPI commands available for interfacing with WSA5000. The commands are listed alphabetically based on the main node, then sub-nodes, so on. See Appendix C's Notation (page 87) for details.

The Release column indicates from which **firmware** release version that the commands are available. **Grayed-out** commands are not yet implemented.

Keyword	Parameter	Description	Release
IEEE Mandated		*Page 45*	
*CLS		Clear all status registers	v1.0
*ESE	<integer>	Event Status Enable register	v1.0
*ESE?		Query ESE register	v1.0
*ESR?		Query Event Status Register	v1.0
*IDN?		Query device identification	v1.0
*OPC		Operation Complete	TBD
*OPC?		Query OC	TBD
*RST		Reset to factory default	v1.0
*SRE	<integer>	Service Request Enable bits	v1.0
*SRE?		Query SRE register	v1.0
*STB?		Query Status Byte register	v1.0
*TST?		Query self-test status	v1.0
*WAI		Wait-to-Continue	TBD
:CALibrate		*Page 62*	
:ABORt			TBD
:RUN		Starts a calibration of the WSA5000 system	TBD
:STATus?			TBD
:INPut		*Page 62*	
:ATTenuator	ON | OFF | 1 | 0	Enables/disables the front-end's 20 dB attenuation	v3.0
:ATTenuator?			
:FILTer			
:PRESelect	ON | OFF | 1 | 0	Enables/disables the use of preselect filtering	TBD
:PRESelect?			
:GAIN			
:IF	<NR1 [unit]>	Selects the variable IF gain stages	TBD
:IF?			
:HDR		Sets gain level for the narrow-band ADC of the HDR signal path	v3.1 – :NB v3.2.1 – :HDR
:HDR?			

Keyword	Parameter	Description	Release
:MODE	ZIF | DD | HDR | IQIN | SH | SHN	Selects the receiver mode of operation	v3.0 – ZIF v3.1 – HDR v3.2 – SH v3.2.1 – IQIN, SHN v3.2.2 – DD
:MODE?			
:OUTput		*Page 69*	
:IQ			
:MODE	CONNector | DIGitizer	Selects the IQ output path type	v3.1
:MODE?			
:CONNector			
:INVersion?	[NRf [unit]]	Query if spectral inversion is required at the given frequency	v4.1.0
[:SENSe]		*Page 65*	
:CORRection			
:DCOFfset	ON | OFF | 1 | 0	Enables/disables the wideband ADC's DC-offset correction	TBD
:DCOFfset?			
:DECimation	OFF | <integer>	Sets the decimation rate as an exponent of 2 (i.e. rate = 2^{level} where level = 0, 1, 2 - 10)	v3.0
:DECimation?			
:FREQuency			
:CENTer	<NRf [unit]>	Sets the center frequency of the RFE	v3.0
:CENTer?	[MAX | MIN]		
:LOSCillator?	<1 | 2>	Gets the frequency to be set for the external LO 1 or 2 in corresponding to current the WSA's center frequency	v3.2.1
:SHIFt	<NRf [unit]>	Sets the frequency shift value (not available for HDR mode)	v3.1
:SHIFt?	[MAX | MIN]		v3.1
:RESolution?		Gets the Analog PLL tuning resolution	TBD
~~:INVersion?~~	~~<NRf [unit]>~~	~~Query if a spectral inversion is required at the given frequency~~ Replaced with	~~v3.2.3~~ v4.1.0 - Deprecated
:LOCK			
:REFerence?		Queries the lock status of the PLL reference clock	v3.0
:RF?		Queries the lock status of the RFE's RF PLL	v3.0
:SOURce		*Page 64*	
:CLOCk			
:ADC	FIX | VAR	Selects variable or fix input clock type to the WB ADC	TBD
:ADC?			
:RATE		Sets the clock rate (in MHz)	TBD
:RATE?			
:REFerence			
:PLL	INT | EXT	Selects the 10 MHz reference clock source	v3.0
:PLL?			

Keyword	Parameter	Description	Release
:RESET		Resets the 10 MHz reference selection to INTernal source	v3.0
:STATus		*Page 58*	
:OPERation		Returns the standard Operation Status Register (OSR) for any event	
[:EVENt]?			TBD
:CONDition?			TBD
:ENABle	<integer>		v3.0
:ENABle?			
:PRESET		Presets the WSA5000 (similar to *RST)	v3.0
:QUEStionable		Returns the standard Questionable Status Register (QSR) for any event	
[:EVENt]?			TBD
:CONDition?			TBD
:ENABle	<integer>		v3.0
:ENABle?			
:TEMPerature?		Returns the WSA5000's internal ambient temperature	v3.2.1
:SWEep		*Page 75* *The sub-nodes are grouped and listed alphabetically based on functionality*	
:ENTRy			
:COPY	<integer>	Copies the settings of an existing sweep entry into the current settings for quick editing	v3.1
:COUNt?		Gets the number of entries available in the list	v3.1
:DELETE	<integer> | ALL	Deletes a specified entry or all entries	v3.1
:FORMat	As defined in :TRACe:FORMat, page 75		
:FORMat?			
:NEW		Sets the sweep entry's capture configuration settings to default values	v3.1
:READ?	<integer>	Gets the settings of an existing sweep entry	v3.1
:SAVE	[integer]	Saves the current editing entry to the end of the list or before the specified ID location in the list when the integer value is given	v3.1
:ATTenuator	As defined in :INPut:ATTenuator, page 63		v3.1
:ATTenuator?			
:DECimation	As defined in [:SENSe]:DECimation, page 65		v3.0
:DECimation?			
:DWELl	<integer>[,<integer>] ::= <sec>[,<microsec>]	Sets the maximum amount of time to wait for the trigger of a sweep entry to occur, after which the trigger is aborted and the next sweep entry if existed will run. When the trigger type is NONE, dwell time is ignored. Default 0.0 sec.	v3.0
:DWELl?			
:FILTer			
:PRESelect	As defined in :INPut:FILTer:PRESelect, page 63		TBD
:PRESelect?			
:FREQuency			

Keyword	Parameter	Description	Release
:CENTer	<NRf [unit]>[,<NRf [unit]>] ::= <start freq>[,<stop freq>]	Sets the center frequency or a range of center frequencies that are stepped by the value defined by :SWEep:ENTRy:FREQuency:STEP	v3.0
:CENTer?			
:SHIFt	As defined in [:SENSe]:FREQuency:SHIFt, page 67		v3.1
:SHIFt?			v3.1
:STEP	<NRf [unit]>	Sets the amount of frequency that the center frequency is stepped by	v3.0
:STEP?			v3.0
:GAIN			
:IF	As defined in :INPut:GAIN:IF, page 63		TBD
:IF?			
:HDR	As defined in :INPut:GAIN:HDR, page 64		v3.1 – :NB v3.2.1 – :HDR
:HDR?			
:MODE	As defined in :INPut:MODE, page 62		v3.1
:MODE?			v3.1
:PPBlock	Same as :TRACe:BLOCk:PACKets, page 73		v3.0
:PPBlock?			
:SPPacket	As defined in :TRACe:SPPacket, page 72		v3.0
:SPPacket?			
:TRIGger			
:LEVel	As defined in :TRIGger:LEVel, page 70		v4.1.0
:LEVel?			
:TYPE	As defined in :TRIGger:TYPE, page 70		v3.1.2 – PULSE | NONE v3.2.0 – WORD
:TYPE?			
:LIST			
:CREAte		Creates a new list identified by a unique string identifier	TBD
:DELETE		Deletes the current list	TBD
:EDIT[?]		Sets the current list of which all subsequent :LIST commands pertain to	TBD
:ITERations	<integer>	Defines the number of times the list is repeated during execution	v3.1
:ITERations?			v3.1
:STARt	[integer]	Begins execution of the current sweep list from the first entry	v3.1
:STATus?			v3.1
:STOP		Stops execution of the current sweep list	v3.1
:SYSTem		*Page 48*	
:ABORt		Aborts the current data capturing process and puts the WSA system into a normal manual mode (i.e. sweep, trigger, and streaming will be aborted)	v3.0

Keyword	Parameter	Description	Release
:CAPability?		Returns a list of the WSA5000's firmware and hardware capability	TBD
:OPTions?		Returns comma separated 3-digit values to represent the hardware option(s) or features available with a particular WSA model	v3.2.1
:DATE	<integer>,<integer>,<integer> ::= <year>,<month>,<date>	Sets the date	TBD
:DATE?			TBD
:CAPTure			
:MODE?		Gets the current capture mode of the WSA (i.e. sweeping, streaming or block mode)	v3.0
:ERRor			v1.0
[:NEXT]?		Returns the SCPI error/event queue	v1.0
:ALL?		Returns all the errors in the queue	TBD
:FLUSh		Clears the WSA5000's internal data storage buffer of any remaining old data that has not been transferred out of the WSA.	v3.0
:LOCK			
:REQuest?	ACQuisition	Request the WSA5000 to provide a lock on a specific task such that only the application that has the lock can perform the task	v3.0
:HAVE?	ACQuisition	Returns the current lock state of the task specified	v3.0
:SYNC			
:MASTer	ON | OFF | 1 | 0	Sets a WSA unit to be the master or slave for a synchronization trigger system with multiple units. Affects :TRIG:TYPE PULSe or WORD	v3.1.2
:MASTer?			
:WAIT	<integer>	Sets the delay time in nanoseconds that the system must wait after receiving the trigger signal before performing data capture	v3.1.2
:WAIT?			
:TIME	<integer>,<integer>,<integer> [,<integer>] | <char> ::= <hr>,<min>,<s>[,<ms>]	Sets the time	TBD
:ADJust	<integer>	Adjust the system time relative to it's current time	
:MODE		Synchronize one time only or continuously	
:MODE?			
:STATus?		Status of the time synchronization	
:SYNC	DISable | NTP,{ONCE | CONTinuous}	Selects the synchronization source and mode	
:SYNC?			
:STATus?		Returns the status of the time synchronization	
:TIME?			v3.2.1
:COMMunicate			
:LAN			
:CONFigure	DHCP | STATIC	Set the WSA's LAN to use DHCP or STATIC configuration type	v3.2.3
:CONFigure?	[CURRENT]		v3.2.3
:DNS	<main DNS>[,alt DNS]	Set the WSA's LAN DNS address(es)	v3.2.3
:DNS?	[CURRENT]		v3.2.3

Keyword	Parameter	Description	Release
:GATEway	<IPv4 address>	Set the WSA's LAN Gateway address	v3.2.3
:GATEway?	[CURRENT]		v3.2.3
:IP	<IPv4 address>	Set the new IPv4 address for the WSA's LAN	v3.2.3
:IP?	[CURRENT]		v3.2.3
:NETMask	<IPv4 address>	Set the WSA's LAN netmask address	v3.2.3
:NETMask?	[CURRENT]		v3.2.3
:APPLy		Apply the new WSA's LAN settings from the commands above, which will then take effect. This command should be applied only once all the required LAN settings have been set.	v3.2.3
:VERSion?		Returns the SCPI compliance version	v3.0
:TRACe		*Page 71*	
:BLOCk			
[:DATA]?		Initiates the sending of the IQ data captured	v3.0
:PACKets	<integer>	Sets the number of IQ data packets to be captured per block (a block = :PACKets * SPP)	v3.0
:PACKets?	[MAX | MIN]		
:FORMat	TD | LPSD	Sets the data output type to be time domain (IQ, I) or frequency domain (logPSD)	TBD
:FORMat?			
:SPPacket	<integer>	Defines the number of IQ samples per VRT packet, and must be a multiple of 16	v3.0
:SPPacket?	[MAX | MIN]		
:STReam			
:STARt	[integer]	Initiates the capture, storage and streaming of IQ data	v3.1
:STOP		Stops streaming	v3.1
:TRIGger		*Page 70*	
:LEVel	<NRf [unit]>,<NRf[unit]>,<NRf [unit]> ::= <start>,<stop>,<level>	Sets the frequency range and amplitude of a frequency domain level trigger	v4.1.0
:LEVel?			
:PERiodic	<integer [unit]>	Sets the time period of a periodic trigger	TBD
:PERiodic?			
:STATus?		Returns the status of the active trigger as to whether it is pending or has occurred	v4.1.0
:TYPE	LEVel | PERiodic | PULSe | WORD | NONE	Sets or disables the trigger type	v3.1.2 – PULSE | WORD | NONE
:TYPE?			

References

1. “Standard Commands for Programmable Instruments (SCPI)”, SCPI Consortium, May 1999, version 1999.0, http://www.spiconsortium.org
2. "VITA Radio Transport (VRT) Draft Standard" VITA-49.0 – 2007, VITA Standard Organization, 31 October 2007, Draft 0.21, http://www.vita.com/
3. "IEEE Standard Codes, Formats, Protocols, and Common Commands", ANSI/IEEE Standard 488.2-1992, http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?tp=&isnumber=5581&arnumber=213762&punumber=2839

Document Revision History

This section summarizes document revision history.

Document Version[1]	Release Date	Revisions and Notes
v2.0 – v2.5	May.2012 – March.2013	- First release of this document for WSA4000 - Modified Stream and Trigger commands - Added details to the Sweep section for single list sweep and synchronized sweep with external trigger - Added :SYSTem:LOCK command subset and note on mandatory command :SYSTem:LOCK:REQuest? ACQuisition for starting data acquisition - Added :SYSTem:ABORt and :SYSTem:FLUSh with caution note - Added VRT Extension context packet class 0x90000004 with a custom context packet defined for sweep start Ids
v2.6	March.11.2013	- SPPacket value is now limited to be a multiple of 16 only with range within 128 to (2^{16} – 16) inclusive - Added :SYSTem:CAPTure:MODE?
v2.7	August.20.2013	Removed the note on IF gain change settling time. It is no longer applicable as the issue is resolved.
v3.0	September.03.2013	- Updated this document to correspond with the capabilities of WSA5000. Contact Support for list of changes from WSA4000 to WSA5000.
v3.1	October.01.2013	- Added: + :OUTput:IQ:MODE command + Table 8 to list the different VRT Stream Ids + Stream IDs to Table 28 (was Table 27) + Table 36 to list the conditions leading to VRT trailer's indicator abnormal state and resolution suggestions + Appendix A: Connecting to WSA and Appendix B: Protocol for Discovering WSA - Made available: + :INPut:GAIN:HDR command + :INPut:MODE and :SWEep:ENTRy:MODE command + Stream feature (see Table 5 for available commands) + Sweep feature (see Table 6 for available commands) + Over-range indicator in the Trailer word - Deprecated: + :TRIGger:SYNC and :SWEep:LIST:TRIGger:SYNC (replaced with :SYSTem:SYNC:MASTer) + :TRIGger:DELay and :SWEep:LIST:TRIGger:DELay (replaced with :SYSTem:SYNC:WAIT) - Changed SYSTem:FTUNe back to [:SENSe]:FREQuency:SHIFt and :SWEep:ENTRy:FREQuency:SHIFt as it was originally. And corrected the allowable range to be -62.5 MHz to 62.5 MHz - Corrected the :SWEep:ENTRy:READ? output response - Changed Figure 11 and 12

Document Version[1]	Release Date	Revisions and Notes
		- Updated :TRACe:SPPacket definition and Table 38 (was Table 35)
v3.1.1	December.01. 2013	- Corrected :SWEep:ENTRy:READ? output response fields - Made available the decimation rate of 1024
v3.1.2	December.20. 2013	- Made available: + :SYSTem:SYNC:MASTer and :SYSTem:SYNC:WAIT + :TRIGger:TYPE and :SWEep:ENTRy:TRIGger:TYPE commands for the PULSE | NONE modes only
v3.1.3	January.10. 2014	- Updated the explanation for [:SENSe]:DECimation - Made available WORD mode for :TRIGger:TYPE and :SWEep:ENTRy:TRIGger:TYPE
v3.2.0	January.30. 2014	- Made available SH mode for :INPut:MODE and updated the IBW from 40 MHz to 30 MHz - Added to VRT's Trailer Word Format section a new Spectral Inversion Indicator bit that is used with SH mode - Added important notes for :OUTput:IQ:MODE CONNector usage
v3.2.1	February.28. 2014	- Made available :SWEep:ENTRy:TRIGger:TYPE WORD - Enabled IQIN option for :INPut:MODE command - Clarified the $IQ_{measured}$ parameter in the VRT's Reference Level section - *RST value for :INPut:GAIN:NB is changed from 0 to -10 dB
v3.2.2	April.14.2014	- Enabled :STATus:TEMPerature? and :SYSTem:TIME?query only commands - Added: + New commands :SYSTem:OPTions? and [:SENSe]:FREQuency:LOSCillator? to support WSAs with the external local oscillator mode + <HDR gain> field to :SWEep:ENTRy:READ? returned string, right after <IF gain> + SHN mode to :INPut:MODE and :SWEep:ENTRy:MODE, as well as in Table 2 + New details/definition to [:SENSe]:DECimation command + A note to :SOURce:REFerence:PLL to see a related AppNote + New notes under Table 2 regarding the IBW of the SH and SHN modes - Changed: + :INPut:GAIN:NB to :INPut:GAIN:HDR for consistency + The **Important Notes** relating to :OUTput:IQ:MODE CONNector mode. Data capture through :TRACe commands is no longer available and the spectral inversion indicator for SH or SHN mode is now available through GPIO port + :PRESet keyword to :PRESET to avoid conflict with the :PRESelect keyword + :DELete to :DELETE to avoid conflict with :DELay + :RESet to :RESET to avoid conflict with :RESume - Removed the following commands as they are unnecessary + :SWEep:LIST:RESume + :INPut:FILTer:SAW

Document Revision History

Document Version[1]	Release Date	Revisions and Notes
v3.2.3	May.01.2014	- Changed maximum SPP for all RFE modes to 65520 - Added a note to Table 37 of :SYSTem:OPTions? regarding external local oscillator option only available to specific WSA5000 variant - Enabled RFE's DD mode available for package release v3.2.2
v3.2.4	May.19.2014	- Added + :SYSTem:COMMunicate:LAN:CONFigure and others LAN commands to change the WSA LAN settings + [:SENSe]:FREQuency:INVersion? command for determining when spectral inversion is required for a given frequency + New notes to Table 2, :INPut:MODE, and :OUTput:IQ:MODE regarding RFE mode availability due to product model dependency
v3.2.5	June.06.2014	- Added USB console control connection and some correction to Figure 2: RF Receiver Front-end and Capture Controller Functional Block Diagram - New note to Table 2 regarding SH/SHN modes and the decimation usage
v3.2.6	June.10.2014	- Corrected the tuning resolution for HDR to be 100 kHz
v3.3.0	July.21.2014	- Data output type for SH/SHN mode with Decimation will now be I and Q. See Table 2 for details. - Replace [:SENSe]:FREQuency:INVersion? With :OUTput:IQ:CONNector:INVersion? - Enable frequency level trigger command :TRIGger:LEVel - Caution note for using external 10 MHz refence source in the :SOURce:REFerence:PLL command
v3.3.1	Aug.11.2014	- Corrected the equation for calculating the absolute power level in the Reference Level section - Corrected the level parameter in :TRIGger:LEVel to be a signed integer type instead of double - Added limitation to the maximum allowable trigger level for the Frequency Domain Triggering basing on the attenuation setting; and removed the mentioning of the dependency on the gain settings.
v3.3.2	Oct.15.2014	- Corrected :TRIGger:STATus? to be unavailable - Corrected the absolute power level formula in the Reference Level section - Updated the Conventions section with the meaning of different fonts used in the document

[1] Document Version is not the same as the firmware Release Version as mentioned in Appendix F: SCPI Commands Quick Reference.