Model 2583
Frequency Response Analyzer

PUBLICATION NUMBER: R-OM-2583

RACAL INSTRUMENTS

United States

(Corporate Headquarters and Service Center)
4 Goodyear Street, Irvine, CA 92618
Tel: (800) 722-2528, (949) 859-8999; Fax: (949) 859-7139

5730 Northwest Parkway Suite 700, San Antonio, TX 78249
Tel: (210) 699-6799; Fax: (210) 699-8857

Europe

(European Headquarters and Service Center)
18 Avenue Dutartre, 78150 LeChesnay, France
Tel: +33 (0)1 39 23 22 22; Fax: +33 (0)1 39 23 22 25

29-31 Cobham Road, Wimborne, Dorset BH21 7PF, United Kingdom
Tel: +44 (0) 1202 872800; Fax: +44 (0) 1202 870810

Via Milazzo 25, 20092 Cinisello B, Milan, Italy
Tel: +39 (0)2 6123 901; Fax: +39 (0)2 6129 3606

Technologie Park, Friedrich Ebert Strasse, 51429 Bergisch Gladbach, Germany
Tel: +49 (0) 2204 844200; Fax: +49 (0) 2204 844219

info@racalinstruments.com
sales@racalinstruments.com
helpdesk@racalinstruments.com
http://www.racalinstruments.com/

PUBLICATION DATE: 13 June, 2001

WARRANTY STATEMENT

Products sold by Racal Instruments are warranted to be free from defects in workmanship or materials. Racal Instruments will, at its option, either repair or replace any hardware products that prove to be defective during the warranty period. You are a valued customer. Our mission is to make any necessary repairs in a reliable and timely manner.

Duration of Warranty

The warranty period for this Racal Instruments hardware is one year, except software and firmware products designed for use with Racal Instruments hardware, which is warranted not to fail to execute its programming instructions due to defect in materials or workmanship for a period of ninety (90) days from the date of delivery to the initial end user.

Return of Product

Authorization is required from Racal Instruments before you send us your product for service or calibration. Call your nearest Racal Instruments support facility. A list is located on the first page of this manual. If you are unsure where to call, contact Racal Instruments, Customer Support Department, USA: 1-800-722-3262 or 1-949-859-8999 or FAX via 1-949-859-7309.
UK: 018706-080134 or FAX via 01753-791290.

We can be reached at helpdesk@racalinstruments.com

Limitation of Warranty

Racal Instruments shall be released from all obligations under this warranty in the event repairs or modifications are made by persons other than authorized Racal Instruments service personnel or without the written consent of Racal Instruments.

Racal Instruments expressly disclaims any liability to its customers, dealers and representatives and to users of its product, and to any other person or persons, for special or consequential damages of any kind and from any cause whatsoever arising out of or in any way connected with the manufacture, sale, handling, repair, maintenance, replacement or use of said products.

Representations and warranties made by any person including dealers and representatives of Racal Instruments which are inconsistent or in conflict with the terms of this warranty (including but not limited to the limitations of the liability of Racal Instruments as set forth above), shall not be binding upon Racal Instruments unless reduced to writing and approved by an officer of Racal Instruments.

Except as stated above, Racal Instruments makes no warranty, express or implied (either in fact or by operation of law), statutory or otherwise; and except to the extent stated above, Racal Instruments shall have no liability under any warranty, express or implied (either in fact or by operation of law), statutory or otherwise.

PROPRIETARY NOTICE

SAFETY PRECAUTIONS

SYMBOLS AND HEADINGS

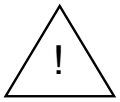
The following symbols and headings are used in this manual to indicate Safety hazards. Personnel using this equipment must read this manual and familiarize themselves with each safety requirement before operating the equipment.

WARNING:

A WARNING indicates a hazard that affects personnel. The instructions in a WARNING must be observed; if the WARNING is ignored, injury or loss of life may result.

CAUTION:

A CAUTION indicates a hazard that affects the equipment. The instructions in a CAUTION must be observed; if the CAUTION is ignored, damage may be caused to the equipment.



This symbol is used on the equipment to indicate that it is necessary to refer to, and comply with, all instructions in this manual regarding the use of such marked facilities.

GENERAL SAFETY PRECAUTIONS

WARNINGS:

1. This instrument has been designed and tested in accordance with EN61010-1:1993/A2:1995, SAFETY REQUIREMENTS FOR ELECTRONIC MEASURING APPARATUS, and has been supplied in a safe condition. This manual contains some information and warnings that must be followed by the user to ensure safe operation and to retain the instrument in safe condition. The instrument has been designed for indoor use.

2. Before any connections are made to the front panel of the instrument and under all permitted conditions of usage the instrument must be fully inserted into the chassis and retained securely in place by the front panel locking screws.

3. To ensure safe operation under all permitted conditions of usage the chassis should be connected to a suitable safety earth point.

4. The system into which the instrument is installed should be fitted with a switch or circuit breaker, located within easy reach of the operator, to enable the system to be disconnected from the mains supply in the event of a hazard arising. The switch or circuit breaker should be clearly marked as the disconnecting device.

5. The environmental operating conditions specified for the instrument must be observed. Do not allow the instrument to become wet, and do not allow water to enter the instrument. Do not operate the instrument when wet because, in this condition, the safety of the instrument may be degraded.

6. The instrument must be kept clean and free from contamination.

7. Any deviation from the instructions provided in this manual might cause the protection provided by the instrument to be impaired.

8. If the instrument is to be powered from the AC line (mains) through an autotransformer, ensure the common connector is connected to the neutral (earth pole) of the power supply.

9. Before operating the unit, ensure the conductor (green wire) is connected to the ground (earth) conductor of the power outlet. Do not use a two-conductor extension cord or a three-prong to two-prong adapter. This will defeat the protective feature of the third conductor in the power cord.

10. Maintenance and calibration procedures sometimes call for operation of the unit with power applied and protective covers removed. Read the procedures and heed warnings to avoid "live" circuit points.

11. Before operating this instrument:
    a. Ensure the instrument is configured to operate on the voltage at the power source. See the Installation Section (Chapter 2 – Installation Instructions).
    b. Ensure the correct fuse is in place for the power source to operate.
    c. Ensure all other devices connected to or in proximity to, this instrument are properly grounded or connected to the protective third-wire earth ground.

12. If the instrument:
    - fails to operate satisfactorily…
    - shows visible damage…
    - has been stored under unfavorable conditions…
    - has sustained stress…

    …do not operate until qualified personnel have checked its performance.

Always operate the product in accordance with the instructions in this manual.

Meets EN61010-1:1993/A2:1995, when used as directed. Suitable for indoor use.

OVERVOLTAGE CATEGORY 1 (EN61010-1).

POLLUTION DEGREE 2 (IEC664)

Safety Precautions Associated with Front Panel Components

The following inputs are available on the front panel of the instrument.

The safety observations should be considered at all times:

Channel 1 Input:
- Maximum Common Mode Voltage: 500V
- Insulation Category II
- **Always use insulated BNC leads for common mode input potentials above 30V rms or 42V pk or 60V DC.**

Channel 2 Input:
- Maximum Common Mode Voltage: 500V
- Insulation Category II
- **Always use insulated BNC leads for common mode input potentials above 30V rms or 42V pk or 60V DC.**

Carrier 1 Input:
- Maximum High or Low to Ground: 350V
- Insulation Category II
- **Always use insulated BNC leads for common mode input potentials above 30V rms or 42V pk or 60V DC.**

Carrier 2 Input:
- Maximum High or Low to Ground: 350V
- Insulation Category II
- **Always use insulated BNC leads for common mode input potentials above 30V rms or 42V pk or 60V DC.**

Generator Output:
- Maximum High or Low to Ground: 150V
- Insulation Category II
- **Always use insulated BNC leads for common mode input potentials above 30V rms or 42V pk or 60V DC.**

Synchronizer Input:
- Maximum High or Low to Ground: 350V
- Insulation Category II
- **Always use insulated BNC leads for common mode input potentials above 30V rms or 42V pk or 60V DC.**

Multipole Input:
- Maximum Core High to ground, 500V
- Insulation Category II

Note: The “Caution Triangle” is used to remind users of special precautions detailed in this manual.

It is placed next to input terminals that are sensitive to over-voltage conditions, as detailed above.

Table of Contents

List of Figures

List of Tables

Chapter 1 – General Information

1.1 INTRODUCTION

The Racal Instruments Model 2583, Two Channel Frequency Response Analyzer is a 'C' sized, single-width VXIbus module that may be fitted into any compatible VXIbus mainframe. This high performance instrument may be used for the accurate measurement of phase and gain relative to either an internally generated signal or reference signal from an external source.

The measured gain and phase values from the two input channels are made using a single sine correlation technique that enables measurements to be made at only the fundamental frequency of excitation. This fundamental frequency is determined to be either the current frequency of the internal signal generator or the measured frequency value of the synchronizer input. If required, a harmonic analysis of the input signal may be performed for orders 2 to 16, provided that the frequency of the highest measured order does not exceed 100kHz. Two carrier inputs are provided that allow modulation of the generator output, utilizing the frequency component of the currently selected carrier input.

Figure 1–1 The 2583 Frequency Response Analyzer VXIbus Module

Operation of the instrument is achieved through a series of low-level message commands. In addition, a VXI*plug&play* driver is provided which contains command calls for each of these commands. This simplifies the creation of user specific driver software or applications. In addition, a Soft Front Panel interface is provided which may be used to initiate full operation from any PC system, provided that appropriate communications with the VXIbus mainframe are established. The Soft Front Panel also contains the functionality to allow automated frequency sweeps to be run and includes the ability to create closed loop control over one of the measurement channels. It provides automated calibration of the instrument and allows up issuing of the instrument's firmware.

1.2 PERFORMANCE SPECIFICATION

This section details the performance specification for the Model 2583 Frequency Response Analyzer.

1.2.1 Generator

Frequency

Range	:	10µHz to 100kHz
Resolution	:	1 part in 65,535
Accuracy	:	Better than 99.99%

Amplitude

Range	:	10mV to 10.3V rms.
Resolution	:	1 part in 65,535
Accuracy	:	>99% ± 1mV
Increment	:	Linear

DC Offset

Range	:	± 10.3V
Resolution	:	1 in 65,535 (<0.4mV)
Accuracy	:	>99% ± 10mV
Increment	:	Linear

Maximum Output Voltage (High to low) : 25 V peak

Output Impedance : 50Ω +0% / -2%

Maximum Voltage (Low to ground) : 150V

Impedance (Low to Ground) : 100kΩ, <100pF

Waveform : Sine, Square and Triangle

Programmed Stop : At 0º, 90º, 180º, 270º and Instantaneous

Distortion (Sinewave) : <1%

Output is short circuit proof

1.2.2 Analyzers

Two Independent channels, operating in parallel.

Frequency	:	10µHz to 100kHz
Input Configuration		
FSD Range	:	30mV, 300mV, 3V, 30V, 300V rms Full Autorange (20% over-range)
Input Common Mode	:	30V (30mV, 300mV and 3V ranges) 500V (30V and 300V ranges)
Common Mode Rejection	:	>65dB up to 50V peak up to 100 Hz (AC coupled specified over 50Hz) >60dB to over 50V peak up to 100 Hz
Protection	:	300V rms
Input Type	:	Differential
Coupling	:	DC or AC (Nominally –3dB @ 0.5Hz)
Impedance (High or low to ground)	:	1MΩ
BNC	:	< 70 pF
Multipole	:	< 100pF
Channel Isolation	:	>85dB @ 1kHz
Integration Time		
Minimum	:	1 cycle
Maximum	:	100,000 cycles
Auto-integration		
Minimum	:	3 cycles
Maximum	:	100,000 cycles
Measurement Delay		
Minimum	:	Zero
Maximum	:	100,000 cycles
Harmonics	:	2 to 16

1.2.3 Measurement Accuracy (Channel to Channel)

For:
Temperature: 20ºC ± 10ºC
Integration: > 20 Cycles
Input: > 10% FSD

		Amplitude	Phase
Up to 50Hz	:	>99.8%	0.1º
50Hz to 1kHz	:	>99.8%	0.25º
1kHz to 5kHz	:	>99.7%	0.5º
5kHz to 20kHz	:	>99.5%	1.0º
20kHz to 50kHz	:	>99.3%	3.0º
Over 50kHz	:	>99.0%	5.0º

Both channels measure simultaneously

1.2.4 Modulator/Demodulator

Two independent carrier inputs

Input Type : Differential

Coupling : AC

Impedance : >100kΩ, <100pF
(High or low to ground)

Common Mode Rejection : >50dB
(<100Hz)

Maximum Common Mode : 300V

Maximum Input : 350V Peak
(High or low to ground)

Carrier Frequency Range : 48Hz to 20kHz

Voltage Ranges : 0.6V to 25V rms
6V to 250V rms

Generator Output Carrier Phase Shift

50Hz to 300Hz	:	<3º
300Hz to 3kHz	:	<1º
3kHz to 20kHz	:	<6º

Analyzer Quadrature Rejection : >26dB

Additional Error when Demodulating
(Modulation frequency = 0.05 x carrier, Input > 10% full scale, Integration 200ms)

Magnitude	:	<0.5% reading
Phase	:	<0.5º

1.2.5 Synchronizer

Input Type ..... : Differential

Coupling ..... : DC or AC (Nominally -3dB @ 3Hz)

Input Impedance ..... : 200kΩ, <100pF
High or low to ground

Maximum Input ..... : 350V peak
High or low to ground

Frequency Range ..... : 1mHz to 100kHz

Trigger Level ..... : Programmable ±25V, Resolution 0.1V

Trigger Edge ..... : Programmable +ve or –ve

Maximum Time to Sync ..... : < 5Hz: 4 cycles
> 5Hz: 500ms + 1 cycle

Ratio Mode Range ..... : 0.001 to 1000 times input frequency

1.2.6 VXIbus Interface

Device Logical Address

Selection ..... : Dynamic or static via 2 rotary switches, accessible from rear:

Dynamic ..... : Set to 255 (address dynamically assigned by VXIbus resource manager)

Static, available range ..... : 1 to 254

Status LEDs ..... : POWER, FAILED, OVERRANGE and MESSAGE

Interrupt Configuration

Selection ..... : Selected manually via rotary switch, accessible from rear

Interrupt levels ..... : 1 to 7, or off

Compatibility ..... : Fully compatible with VXIbus System Specification, Revision 1.4

Control ..... : Message based servant

Protocol ..... : Word Serial

1.2.7 General

Safety ..... : BS EN 61010-1:1993/A2 1995

EMC

Emission ..... : EN61326-1:1997 + A1:1998, Class B

Immunity ..... : EN61326-1:1997 + A1:1998, Table 1

Temperature

Operating ..... : 0° to +55°C

Some specified limits ..... : +20°C ± 10°C

Storage ..... : -40° to +70°C

Humidity ..... : Non-condensing 93.3% at +40°C

Vibration ..... : Designed to conform to IEC68-2-6

Bump ..... : Designed to conform to IEC68-2-29

Drop/Topple ..... : Designed to conform to IEC68-2-31

Dry Heat ..... : Designed to conform to IEC68-2-2

Low Temperature ..... : Designed to conform to IEC68-2-1

Steady State Humidity ..... : Designed to conform to IEC68-2-3

Power Requirements

Voltage	Current (mA) I_{Pm}	Current (mA) I_{Dm}
-2V DC	0	0
+5V DC	3000	3000
+5V DC (standby)	0	0
-5.2V DC	0	0
+12V DC	450	300
-12V DC	50	50
+24V DC	525	500
-24V DC	525	500

Total Power ..... : < 50W

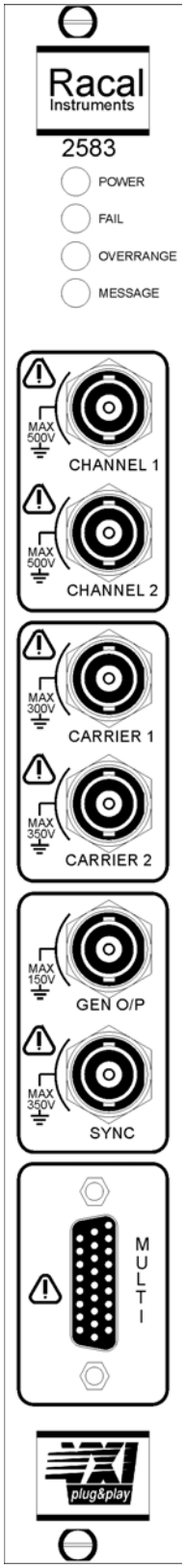
Cooling ..... : 3.0 l/sec @ 0.5mm H_2O
(10°C maximum rise)

Packaging ..... : Single slot width, 'C' size VXIbus module
262.0mm (H) x 30.18 (W) x 355.0mm (D)

Weight ..... : ~1kg

MTBF ..... : 14,000 hours

FRONT PANEL COMPONENTS



The module front panel is comprised of 4 display LEDs, 6 BNC signal connectors and 1 ‘26 way' multipole connector.

The following LEDs are provided and have the following purposes

POWER : Illuminated when all power supply lines to the VXIbus module are within the module specification.

FAILED : Illuminated during system initialization at power on and extinguished when the module is ready to commence VXIbus operation.

OVERRANGE : Illuminated either upon the failure of a module self test or upon an input over-range condition at one of the Channel or Carrier inputs.

MESSAGE : Illuminated during communication with the VXIbus.

The following signal inputs/outputs are available for normal operation:

CHANNEL 1 : Differential measurement channel. Full operation detailed in Section 3.1.2 below

CHANNEL 2 : As Channel 1.

CARRIER 1 : Differential carrier input channel. Full operation detailed in Section 3.1.3 below

CARRIER 2 : As Carrier 1.

GEN O/P : Programmable signal generator. Full operation detailed in Section 3.1.1 below

SYNC : Differential synchronizer input. Full operation detailed in Section 3.1.1 below

MULTI : If required, the operation of all inputs and outputs may be switched through this connector to ease/tidy the connection of permanent ATE setups.

Wiring details for this connector are shown in Figure 1–3 and *Table 1—1* below.

Figure 1–2 2583 Front Panel

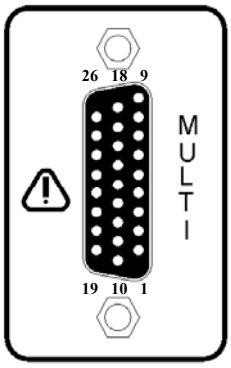


Figure 1–3 Multipole Connector Pin Layout

Functional Description	Pin Number
Channel 1 HI	9
Channel 1 LO	26
Channel 1 GND	18
Channel 2 HI	7
Channel 2 LO	24
Channel 2 GND	16
Carrier 1 HI	5
Carrier 1 LO	22
Carrier 1 GND	14
Carrier 2 HI	4
Carrier 2 LO	21
Carrier 2 GND	13
Generator HI	2
Generator LO	19
Generator GND	11
Synchronizer HI	3
Synchronizer LO	20
Synchronizer GND	12

Table 1—1 Multipole Connector Pin Assignment

Chapter 2 – Installation Instructions

2.1 UNPACKING AND INSPECTION

The following instructions should be followed when unpacking and inspecting the instrument prior to first use:

1. Before unpacking the 2583 FRA, check the exterior of the shipping carton for any signs of damage. All irregularities should be noted on the shipping bill.

2. Carefully remove the instrument from the factory packaging and be sure to preserve it for future use.

3. Inspect the instrument for any defects or damage. Immediately notify the carrier if any damage is apparent.

4. Before use, have qualified personnel perform a safety check.

NOTE:

Proper ESD handling procedures must always be used when packing, unpacking or installing any module. Failure to do so may cause damage to the unit.

2.2 INTERRUPT LEVEL & LOGICAL ADDRESS SETTING

The 'Priority Interrupt' and 'Logical Address' settings of the Model 2583 are selected manually via the three rotary switches at the rear of the unit (Figure 2–1).

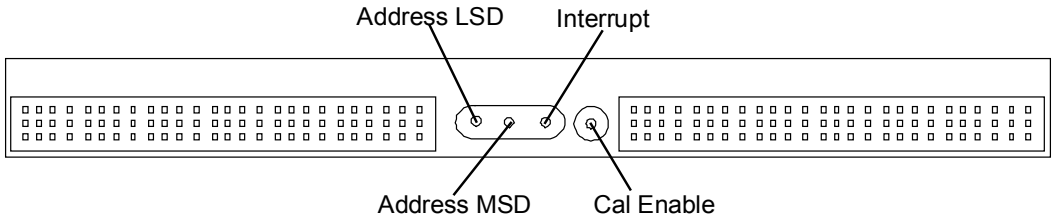


Figure 2–1 Address / Interrupt Switch Location

These settings should be made **before** the installation of the instrument in the mainframe rack. The following steps should be followed:

- Place the unit on its side with two connectors facing forward; the three rotary switches detailed above will become apparent *(*Figure 2–1*)*. The calibration switch will also be seen, although a calibration seal may cover this.

- The address LSD and MSD switches are used to set the logical address of the unit. Any logical address between 1 and 255 is permitted. The LSD switch sets the least significant address bit, whilst the MSD switch sets the most significant address bit.

  Note: The Address MSD + LSD = F + F = Dynamic Configuration (255)

- The default 'Priority Interrupt' level of the module may be set. If the 'Interrupt' switch is set to '0' then priority interrupt will be disabled. Otherwise interrupts on levels 1 to 7 are permitted.

2.3 INSTALLATION IN MAINFRAME

Before installation, a visual inspection of the instrument should be performed. In particular, inspect connectors P1 and P2 at the rear of the unit for bent, damaged or missing pins. Qualified personnel should repair any damage before proceeding.

Refer to the product identification label on the side of the module for system integration information relating to voltage, power and cooling requirements to be supplied by the VXIbus chassis.

To install the unit into a 'C' sized VXIbus mainframe the following procedure should be followed:

1. Verify that connectors P1 and P2, located at the rear of the instrument, are orientated to mate with the corresponding connectors on the mainframe back-plane.
2. Align the instrument with the card guides for the slot selected, and slide the instrument into the mainframe.
3. Push the instrument into the mainframe, using a firm even pressure to ensure that the connectors are mated properly. **DO NOT** use undue force that could lead to damage of the connectors.
4. Secure the instrument into the mainframe using the captive screws provided.
5. Poor mechanical alignment of the rear connectors P1 and P2 may require the unit to be re-seated in the VXIbus mainframe.

2.4 APPLYING POWER TO THE MAINFRAME

When power is applied to the mainframe a module self-test is performed, the following LED sequence may be verified on the front of the instrument in order to determine the status of the module:

1. Upon application of power, the 'Power', 'Fail' and 'Over-range' LEDs will become illuminated.
2. The module will then commence to test the internal memory and interface circuits. This test lasts for approximately 2 seconds and the 'Fail' LED will turn off upon completion.
3. The 'Power' and 'Over-range' LEDs will remain illuminated, until the system controller initializes the instrument. When initialized, the module tests the internal analogue circuits. During this test, the measurement, carrier, synchronizer and generator circuits are tested. It is possible to hear the internal relays clicking for the duration of this test, which lasts for approximately 1 second. Upon the successful completion of this test the over-range LED will go out and the module will be released to the mainframe.

Should either section of the self-test fail, the over-range LED will continue to flash at a rate of approximately twice per second. Should this occur, the instrument should be returned to the vendor for servicing following the instructions in Section 6.2 below. It must be noted that there are no user-serviceable parts within the 2583 module.

2.5 INSTALLING THE VXI*plug&play* DRIVER

To install the VXI*plug&play* driver successfully the computer must be running Windows 95 or Windows NT.

There are two 3.25" diskettes for this driver installation. Insert Disk 1 of 2 and press "Start" on the bottom taskbar. Then select "Settings" and "Control Panel". In the Control Panel select "Add/Remove Programs" and press "Install…". Follow the instructions, inserting Disk 2 when requested to do so.

2.6 CHECKING THAT THE MODULE IS OPERATIONAL

Having powered up the system and checked that the sequence described in Section 2.4 above completed successfully then:

- Initialize the system as appropriate (e.g. for a PCI MXI-2 system, run RESMAN)
- Run the Self-Test from the VXI*plug&play* soft front panel and check that a "PASS" is indicated.

2.7 REMOVAL FROM MAINFRAME

To remove the Model 2583 from the VXIbus mainframe, the following procedure should be followed:

1. Power-down the mainframe and release the captive screws that secure the Model 2583 into the mainframe.
2. Eject the Model 2583 from the mainframe using the plastic levers provided on the top and bottom edges.
3. Pull the Model 2583 forward in the card guides, until it is released from the mainframe.

Chapter 3 – System Operation

3.1 OPERATION OF HARDWARE

The following overview gives detail of the major hardware components of the 2583 module and the operational parameters that may be set by the user. A brief description of the operation of each of these parameters is given, based upon the low-level Message Commands detailed in Chapter 4 below, Sections 4.1 and 4.2. In addition, all functionality described here may also be achieved through use of the supplied DLL driver and the soft front panel (Section 4.3).

3.1.1 Generator and Synchronizer

The generator of the 2583 FRA may be programmed to operate over the frequency range of 10μHz to 100kHz. The generator is normally used to excite the system or device under test and is the reference from which phase measurements will be taken. Alternately it may be preferable to use an external generator for this purpose, in which case an input synchronizer channel is provided as a phase reference. The synchronizer may accept an input signal over the frequency range of 1mHz to 100kHz.

The following generator/synchronizer parameters may be set:

- Generator Frequency
- Generator Amplitude
- Generator Bias
- Generator Waveform
- Generator On/Off
- Generator Soft Start
- Generator Soft Stop
- Generator Hold
- Synchronizer Select
- Synchronizer Lock State
- Synchronizer Edge
- Synchronizer Level
- Synchronizer Input Ratio
- Synchronizer Coupling Select

3.1.1.1 Generator Frequency

The frequency of the signal generator may be set anywhere in the range of 1μHz to 100kHz. The new generator frequency will be set immediately, irrespective of the current status of the generator output.

Message Command: GFR

3.1.1.2 Generator Amplitude

The amplitude of the signal generator is always set in volts rms with an upper limit of 10.3V. The new generator output amplitude will be set immediately, irrespective of the current status of the generator output.

Message Command: GAM

3.1.1.3 Generator Bias

A DC bias level or offset may be applied to the generator output in the range of –10.3V to +10.3V. Any bias applied will remain present at the output at all times and is independent of the *Generator ON* and *Generator OFF* commands.

Message Command: GBI

3.1.1.4 Generator Waveform

The current generator waveform may be selected to be of either the Sinusoidal, Square or Triangle types. The new waveform will be set immediately, irrespective of the current status of the generator output.

Message Command: GWF

3.1.1.5 Generator ON/OFF

These two commands are used to enable and disable the output of the integral signal generator. When enabled, the current generator frequency, amplitude and waveform will be applied. Any selected generator bias level will remain active at all times, irrespective of the status of the generator output.

Message Command: GON/GOF

3.1.1.6 Generator Soft Start/Stop

These two commands may be used to enable a gradual ramp up or down of the generator output when it is turned on or off. This is particularly useful in applications when large electro-mechanical devices are being used, such as shaker tables or hydraulic jacks where the sudden application of the excitation signal may otherwise cause damage to the machine.

Message Command: GSN/GSF

3.1.1.7 Generator Hold

The generator hold function allows the output of the function generator to be held at any of the four quadrant boundaries of the output signal (0º, 90º, 180º and 270º). Additionally, there is an instantaneous hold option.

When the generator output is held, normal operation may be resumed either by selecting a new hold point, in which case the generator will continue until the requested hold condition is met, or alternately the generator ON command may be issued which will resume the operation of the generator until either a new hold point is selected or the generator is disabled.

Message Command: GHO

3.1.1.8 Synchronizer Select

In certain applications, it may not be possible to use the internal signal generator to stimulate the system under test. In this case, the FRA may make phase measurements from an external synchronizer signal. The synchronizer signal can be set anywhere in the range of 1mHz to 100kHz. Once the synchronizer is enabled, the generator may be used to provide a phase locked waveform at a user-specified magnitude and type. If the synchronizer is used, either a loose or a tight waveform lock may be specified. If a loose

lock is requested, the module will track the period of the incoming waveform over one complete cycle and use the calculated frequency as the phase reference. If a tight lock is specified, the instrument will continue to track the incoming waveform for further cycles until is has ensured that full synchronization to the incoming signal has been achieved. It may be advantageous to use a loose lock if the quality of the incoming signal is poor, and is unable to be reliably triggered with a tight lock.

When using external synchronization, it is generally recommended that channel to channel measurements are used wherever possible, since these will not be affected by any error in the phase synchronization.

Message Command: *SYS*

3.1.1.9 Synchronizer Lock State

This returns a value to identify whether or not the synchronizer has successfully 'locked' onto the input signal. The Lock State will become active only if a 'tight' lock has been achieved, as specified in the 'SYS' command.

Message Command: *SLO?*

3.1.1.10 Synchronizer Edge Detect

When the synchronizer is used as the measurement phase reference, either the positive or negative edge of the input signal may be used as the trigger. As the synchronizer uses an edge detection technique to determine the phase reference point, it is important that the rate of change of the input signal is as high as possible.

Message Command: *SYE*

3.1.1.11 Synchronizer Trigger Level

The module's synchronizer may accept an input signal anywhere in the range of ±350V. The exact point at which the synchronizer registers the incoming reference is called the trigger level and this must be set by the user.

Message Command: *SYL*

3.1.1.12 Synchronizer Input Ratio Selection

Although the synchronizer is generally used to provide the required measurement frequency, this is not always possible. By changing the Ratio Selection, any ratio between the Synchronization frequency and Measurement frequency may be requested. When the Ratio is not set to zero, it is not possible for phase locking to occur; therefore, absolute phase measurements are not possible.

The value entered is a ratio of the input synchronizer frequency and the desired reference signal frequency. For example, if a value of 0.25 is entered, and the synchronizer input reads a frequency of 1kHz, the measurement frequency used by the module will be 250Hz. The entered ratio may be either greater than, or less than 1, provided that the resultant frequency for phase measurement calculation is within the bandwidth of the instrument.

Message Command: *SYM*

3.1.1.13 Synchronizer Coupling Select

The input coupling of the synchronizer channel may be specified as either AC or DC

coupled.

Message Command: *SYC*

3.1.2 Analyzers

The 2583 has two high performance analyzer channels that operate over the frequency range of 1µHz to 100kHz. Each differential analyzer channel is capable of making absolute magnitude and relative phase measurements [referenced to the signal generator or synchronizer] for input magnitudes of up to 300V rms. If required, a harmonic analysis of the input signal may also be performed for measurement orders 2 to 16, provided that the highest order frequency does not exceed 100kHz.

As the FRA employs a sine correlation technique for result measurement, only the fundamental frequency of the generator (or selected harmonic) is considered during the measurement process. This measurement process will inherently reject noise. However, the extent of this rejection is dependent upon the measurement integration time that may be set.

The following analyzer parameters may be set:

- Integration Time
- Measurement Delay
- Auto-integration
- Auto Integrate Channel Selection
- Measurement Channel Range
- Measurement Channel Coupling
- Measurement Channel Harmonic Selection

3.1.2.1 Integration Time

The integration time is the time spent by the analyzers in calculating the signal levels during each measurement. The longer that the analyzers are allowed to measure, the better the noise rejection of the instrument becomes and, therefore, the more accurate the measurement. The integration time may be specified as either a time, in seconds, or as a number of generator/synchronizer cycles.

Message Command: *AIT*

3.1.2.2 Measurement Delay

The module is capable of making a measurement just a few milliseconds after a new generator excitation setting has been made. In some circumstances, this may be too fast for the system under test to stabilize. Therefore, a measurement delay time may be given. The measurement delay time may be specified as either a time, in seconds, or as a number of generator cycles.

Message Command: *AMD*

3.1.2.3 Auto Integrate

If the degree of noise on the measurement signals is unknown, auto-integration may be used to improve the accuracy of the measurement. If auto-integration is enabled, the module will continue to take measurements until the statistical result on each measurement indicates the required accuracy has been achieved. Two levels of auto-integration are

provided:

- Short – Standard Deviation of Measurements <10% of Result
- Long – Standard Deviation of Measurements <1% of Result

Message Command: AAI

3.1.2.4 Auto Integrate Channels

If auto-integration is selected, it will operate only on the channels specified through this command.

Message Command: AIC

3.1.2.5 Measurement Channel Range

The range of the measurement channels may be set as either 30mV, 300mV, 3V, 30V or 300V rms. Alternately, auto-range may be selected. In this case, the module will set the optimum range for each input channel prior to each measurement being taken. If the user sets the input range of the analyzer channel a slight increase in measurement speed may be achieved.

An over-range indication LED is provided on the front of the unit which becomes active when the magnitude of the incoming signal exceeds 20% of the current channel range setting.

Message Command: ACR

3.1.2.6 Measurement Channel Coupling

The input coupling of each measurement channel may be specified as either AC or DC coupled. The –3dB point when AC coupling is selected lies at approximately 0.5Hz.

Message Command: ACC

3.1.2.7 Measurement Channel Harmonic Selection

If required, the module may take magnitude and phase measurements of integer harmonics of the fundamental frequency in the range of the 2^{nd} to the 16^{th} harmonic. The fundamental frequency is the current frequency of the generator/synchronizer.

Message Command: HAR

3.1.3 The Carriers

The 2583 module incorporates two carrier channels that may be used as a frequency source for output generator modulation. Both carrier amplitude and modulation amplitude settings may be applied to the generator output using either one of the carrier inputs, each having an input frequency range of 48Hz to 20kHz.

The following carrier parameters may be set:

- Generator Modulation
- Generator Modulation Amplitude
- Generator Modulation Carrier
- Channel Demodulation
- Carrier Input Range Selection

3.1.3.1 Generator Modulation

If modulation of the generator output is required, the carrier input channel to be used may be specified.

Message Command: GMO

3.1.3.2 Generator Modulation Amplitude

If generator output modulation is selected, the modulation amplitude is specified in volts rms. The maximum modulation amplitude value allowed is 10.3V rms.

Message Command: GMA

3.1.3.3 Generator Modulation Carrier

If generator output modulation is selected, the modulation carrier amplitude may be specified. The maximum modulation carrier amplitude allowed is 10.3V.

Message Command: GMC

3.1.3.4 Channel Demodulation

If generator output modulation is selected, demodulation may be required on the measurement channels in order to make measurements on the input signal. Demodulation may be selected for either of the input channels for either one of the two carrier channels.

Message Command: ACM

3.1.3.5 Carrier Input Range Selection

The range of the carrier input channels might be manually set as either 30V or 300V rms.

Message Command: CAR

3.1.4 General/Results Output

The 2583 module allows measurements to be taken and output as part of an ASCII message. Additionally, the module may report on its current status and give details of reported system errors.

The following interface commands may be issued:

- Perform Single Measurement
- Stop Measuring
- Output Last Results Set
- Output Instrument Status
- Return Error From Error Queue
- Input Connector Select
- Return Calibration Date

3.1.4.1 Perform Single Measurement

This command is used to perform a single measurement. The result from the measurement will be held in module memory until the completion of the next measurement. The result comprises of the frequency of measurement and the real and imaginary components of the result from Channels 1 and 2.

Message Command: MSI

3.1.4.2 Stop Measuring

If the current measurement is not required, this command may be issued to abort it. This is useful when taking measurements at low frequencies where each measurement may take a considerable time to complete. If a measurement is aborted, no result will be available.

Message Command: MST

3.1.4.3 Output Last Results Set

This command outputs the result from the last complete measurement in a text format. The returned information includes the frequency of measurement and the real and imaginary components of the result from Channels 1 and 2. It must be noted that if the command is issued as a new measurement is being taken, the result from the last complete measurement will be returned.

Message Command: ?ODC

3.1.4.4 Output Instrument Status

This command returns the current status of the instrument. The returned information details the current frequency of the generator and the current measurement status.

Message Command: ?STS

3.1.4.5 Return Error From Error Queue

If any errors have occurred, they will be stored within the error queue. This command may be used to return the last error stored in the error queue.

Message Command: ERR?

3.1.4.6 Input Connector Select

Signal measurements may be taken through either the BNC or the multipole connectors on the front panel. This command is used to switch between the input connector types.

Message Command: ICS

3.1.4.7 Return Last Calibration Date

This allows the date of the last module calibration to be returned.

Message Command: CDA

Chapter 4 – Control Interfaces

4.1 COMMAND LEVEL INTERFACE

This is the lowest level of system command. At this level the module may be controlled in the standard Message Command format. Two sets of commands are available to the user; these are commands that are generic to all IEEE 488.2 compatible instruments and commands that are specific to the Model 2583, Frequency Response Analyzer.

Each command consists of a simple ASCII string that is sent to the instrument. The commands are constructed from three components: an operand and up to three control parameters or arguments. The command operand is usually a simple mnemonic of the required function, e.g. 'Generator On' = GON.

The general format of commands is as follows:-

CCC P1,P2,P3;CCC P1,P2...

Where: CCC is the command mnemonic

P1 is the first optional parameter, separated by a space from the mnemonic

P2 and P3 are further optional parameters, separated from each other by commas

; Separates commands on the same line from each other

All available commands are tabulated below. If a command is listed as having a 'Response' then the current setting of that parameter may be requested by preceding the mnemonic with a '?', e.g. '?GFR' will return the current generator frequency.

4.1.1 Device Specific Interface Commands

The following commands are specific to the 2583 device and are used to control the instrument functionality:

GFR – Generator Frequency

Allow the current generator frequency to be set.

Command Mnemonic	GFR
Number of Arguments	1
Argument 1	Frequency in Hz (Floating Point)
Response Available	Yes

GAM - Generator Amplitude

Allow the current generator amplitude to be set.

Command Mnemonic	GAM
Number of Arguments	1
Argument 1	Amplitude in Volts rms (Floating Point)
Response Available	Yes

GBI - Generator Bias

Allow the current generator bias level to be set.

Command Mnemonic	GBI
Number of Arguments	1
Argument 1	Bias Voltage in Volts (Floating Point)
Response Available	Yes

GWF - Generator Waveform

Allow selection of the current generator waveform type.

Command Mnemonic	GWF
Number of Arguments	1
Argument 1	0 = Sine Waveform (Integer) 1 = Square Waveform 2 = Triangle Waveform
Response Available	Yes

GON - Generator On

Enable the generator output.

Command Mnemonic	GON
Number of Arguments	0
Response Available	No

GOF - Generator Off

Disable the generator output.

Command Mnemonic	GOF
Number of Arguments	0
Response Available	No

GSN - Soft Start

Enable/disable soft start for the generator output.

Command Mnemonic	GSN
Number of Arguments	1
Argument 1	0 = Enabled 1 = Disabled
Response Available	Yes

GSF – Soft Stop

Enable/disable soft stop for the generator output.

Command Mnemonic	GSF
Number of Arguments	1
Argument 1	0 = Enabled (Integer) 1 = Disabled
Response Available	Yes

GHO - Generator Hold

Enable the hold facility on the generator output.

Command Mnemonic	GHO
Number of Arguments	1
Argument 1	0 = Off (Integer) 1 = Stop at 0° 2 = Stop at 90° 3 = Stop at 180° 4 = Stop at 270° 5 = Instantaneous
Response Available	Yes

GMO - Generator Modulation

Allow selection of source carrier channel for generator output modulation.

Command Mnemonic	GMO
Number of Arguments	1
Argument 1	0 = Off 1 = On using Carrier Input 1 2 = On using Carrier Input 2
Response Available	Yes

GMA - Generator Modulation Amplitude

Allow the generator amplitude modulation level to be set.

Command Mnemonic	GMA
Number of Arguments	1
Argument 1	Modulation Amplitude in Volts
Response Available	Yes

GMC - Generator Modulation Carrier

Allow the generator carrier voltage to be set.

Command Mnemonic	GMC
Number of Arguments	1
Argument 1	Modulation Carrier in Volts
Response Available	Yes

SYS – Synchronizer Select

Enable/Disable the use of the synchronizer as the phase reference.

Command Mnemonic	SYS
Number of Arguments	1
Argument 1	0 = Disabled (Integer) 1 = Loose Lock 2 = Tight Lock
Response Available	Yes

SYE – Synchronizer Edge

Allow selection of either the negative or positive going edge for synchronizer operation.

Command Mnemonic	SYE
Number of Arguments	1
Argument 1	0 = Positive Edge (Integer) 1 = Negative Edge
Response Available	Yes

SYC – Synchronizer Coupling Select

Allow selection of either AC or DC coupling on the synchronizer input.

Code	SYC
Number of Arguments	1
Argument 1	0 = DC Coupled Input 1 = AC Coupled Input
Response Available	Yes

SYL – Synchronizer Level

Allow the input trigger level of the synchronizer to be set.

Command Mnemonic	SYL
Number of Arguments	1
Argument 1	Trigger Level in Volts (Floating Point)
Response Available	Yes

SYM – Synchronizer Input Ratio

Allow selection of the synchronizer input ratio.

Command Mnemonic	SYM
Number of Arguments	1
Argument 1	Measurement Frequency as Ratio of Synchronizer Frequency (Floating Point)
Response Available	Yes

SLO? – Synchronizer Lock State

Return the current lock state of the synchronizer.

Code	SLO?
Number of Arguments	0
Response Available	Yes, 1 = Locked, 0 = Unlocked

AIT - Integration Time

Allow a measurement integration time to be set.

Command Mnemonic	AIT
Number of Arguments	2
Argument 1	Integration Time (Floating Point)
Argument 2	S = Argument 1 in seconds (Character) C = Argument 1 in Cycles
Response Available	Yes

AMD - Measurement Delay

Allow a measurement delay time to be set.

Command Mnemonic	AMD
Number of Arguments	2
Argument 1	Measurement Delay Time (Floating Point)
Argument 2	S = Argument 1 in seconds (Character) C = Argument 1 in Cycles
Response Available	Yes

AAI - Auto-integration

Allow the duration of auto-integration to be specified.

Command Mnemonic	AII
Number of Arguments	1
Argument 1	0 = Disabled (Integer) 1 = Short Integration 2 = Long Integration
Response Available	Yes

AIC - Auto Integrate Channels

Enable/disable auto-integration for the specified channels.

Command Mnemonic	AIC
Number of Arguments	2
Argument 1	A = Channel A (Character) B = Channel B
Argument 2	0 = Disabled (Integer) 1 = Enabled
Response Available	Yes

ACR - Channel Range

Allow the input range for each measurement channel to be set.

Command Mnemonic	ACR
Number of Arguments	2
Argument 1	A = Channel A (Character) B = Channel B
Argument 2	0 = Auto (Integer) 1 = 30mVolts 2 = 300mVolts 3 = 3 Volts 4 = 30 Volts 5 = 300 Volts
Response Available	Yes

ACC - Channel Coupling

Allow the input coupling type each measurement channel to be set.

Command Mnemonic	ACC
Number of Arguments	2
Argument 1	A = Channel A (Character) B = Channel B
Argument 2	0 = AC Coupled (Integer) 1 = DC Coupled
Response Available	Yes

ACM – Channel Demodulation

Set demodulation for the specified input channel using the specified carrier channel.

Command Mnemonic	ACM
Number of Arguments	2
Argument 1	A = Channel A (Character) B = Channel B
Argument 2	0 = Off 1 = On using Carrier Input 1 2 = On using Carrier Input 2
Response Available	Yes

CAR – Carrier Input Full Scale

Set the full scale input of the carrier channels

Code	CAR
Number of Arguments	2
Argument 1	A = Channel A (Character) B = Channel B
Argument 2	0 = 300 V rms 1 = 30 V rms
Response Available	Yes

HAR - Harmonics

Select the harmonic of the measured response signal for which magnitude and phase results are to be obtained.

Command Mnemonic	HAR
Number of Arguments	1
Argument 1	1 = Off (Integer) 2 = 2nd Harmonic …. 16 = 16th Harmonic
Response Available	Yes

MST - Stop Measuring

Abort the current measurement, without obtaining a result.

Command Mnemonic	MST
Number of Arguments	0
Response Available	No

MSI - Single Measurement

Perform a single measurement on both channels.

Command Mnemonic	MSI
Number of Arguments	0
Response Available	No

ORR? – Report Out Of Range Conditions

Perform a single measurement on both channels.

Code	ORR?
Number of Arguments	1
Argument 1	Mask of Out of Range Conditions
Response Available	Yes

Return a bit mask containing the Out of Range that are stored. This value is cleared by the ORR? Command, the *CLS command or by commencing a new measurement.

The bits in the returned value are as follows:-

Bit	Decimal	Description
Bit 0	1	Channel 1 Common Mode Overrange
Bit 1	2	Channel 2 Common Mode Overrange
Bit 2	4	Carrier 1 Level Overrange
Bit 3	8	Carrier 2 Level Overrange
Bit 4	16	Not Used
Bit 5	32	Carrier 1 Common Mode Overrange
Bit 6	64	Carrier 2 Common Mode Overrange
Bit 7	128	Power Supply out of range Overrange
Bit 14	16384	Carrier 1 Level Underrange
Bit 15	32768	Carrier 2 Level Underrange
Bit 16	65536	Channel 1 Dynamic Range Overrange
Bit 17	131072	Channel 2 Dynamic Range Overrange

ORE – Out Of Range Mask Enable

Set a mask of bits that will cause the Out of Range Summary Status bit to be set in the Status Byte register. The bits are as described for the ORR? Command.

For example, value of 5 will cause Channel 1 Common Mode or Carrier 1 Level Overrange conditions to cause the Out of Range Summary Status bit to be set to 1.

Code	ORE
Function	Out of Range Mask Enable
Number of Arguments	1
Argument 1	Mask of Out of Range Conditions that will set the Out of Range Summary Status bit in the Status Byte Register
Response Available	Yes

?STS – Instrument Status

Return the current status of the instrument.

Command Mnemonic	?STS
Number of Arguments	3 (Response Only, no arguments for request)
Argument 1	GFR ffff (ffff is the current Frequency in Hz)
Argument 2	MEA S (Measurement Stopped) MEA 1 (Single Measurement in Process)
Argument 3	SWE 0 (Sweep not supported on VXIbus)
Response Available	Yes

?ODC Last Data Set

Return the result from the last complete measurement.

Command Mnemonic	?ODC
Number of Arguments	8 (Response Only, no arguments for request)
Argument 1	Frequency in Hz (Float)
Argument 2	Unused (Integer)
Argument 3	1 (Channel Identifier)
Argument 4	Channel 1 Real Component (Floating Point)
Argument 5	Channel 1 Complex Component (Floating Point)
Argument 6	2 (Channel Identifier)
Argument 7	Channel 2 Real Component (Floating Point)
Argument 8	Channel 2 Complex Component (Floating Point)
Response Available	Yes

ICS - Input Connector Select

Select the input signal source for either the BNC or Multipole connectors.

Command Mnemonic	ICS
Number of Arguments	1
Argument 1	0 = BNC Connectors (Integer) 1 = Multipole Connector
Response Available	Yes

ERR? - Return Error from Error Queue

Return the last error stored in the error queue. A full list of errors is available in Appendix A - Message Command Error Codes

Command Mnemonic	ERR?
Function	Returns Earliest Stored Error
Number of Arguments	nnn,EXPLANATORY_TEXT Where nnn is the internal error Command Mnemonic 000 denotes no errors available 999 denotes that an overflow occurred in the error list (Response only, no arguments for request)
Argument 1	See Appendix A.
Response Available	Yes

CDA – Report Date Of Last Calibration

Return the date of the last module calibration.

Example, The response '1,21,1,2000,12,45,30,0,0,0,0' shows calibration active, and a calibration date of 21st January 2000 at 12:45:30.

Code	CDA?
Function	Report Date of Last Calibration
Number of Arguments	1
Argument 1 (Inquiry)	0 – Calibration Disabled 1 – Calibration Enabled If Enabled, Returns 10 integer values. The first 6 give the day, month, year, hour, minute and second that the unit was last calibrated. The remaining 4 are currently undefined.
Response Available	Yes

4.1.2 Generic Message Commands

The following commands are defined in the IEEE-488.2 standard and are generic to all compliant instruments.

****CLS - Clear Event Register***

This command will clear the Standard Event Status Register, and the Error Queue.

Command Mnemonic	*CLS
Number of Arguments	0
Response Available	No

****ESE - Set the Standard Event Status Enable Register***

Set the Standard Event Status Enable Register

Command Mnemonic	*ESE
Number of Arguments	1
Argument 1	1 – Bit Mask as follows:- Bit 0 – Operation Complete Bit 1 – Unused Bit 2 – Query Error Bit 3 – Device Dependent Error Bit 4 – Execution Error Bit 5 – Command Error Bit 6 – Unused Bit 7 – Unused
Response Available	Yes

****ESR? - Query and Clear the Standard Event Status Register***

Read and Clear the Standard Event Status Register

Command Mnemonic	*ESR?
Number of Arguments	1 (Response Only)
Argument 1	1 – Bit Mask as follows:- Bit 0 – Operation Complete Bit 1 – Unused Bit 2 – Query Error Bit 3 – Device Dependent Error Bit 4 – Execution Error Bit 5 – Command Error Bit 6 – Unused Bit 7 – Power On
Response Available	Yes

**IDN? - Identify*

Return Instrument Identification String

Command Mnemonic	*IDN?
Number of Arguments	1 consisting of 4 comma separated fields <Manufacturers Command Mnemonic>,<Model Number>,<Serial No>,<Firmware Version> (Response only, no arguments for request) All fields may be text or numeric
Field 1	Racal Instruments
Field 2	2583
Field 3	Variable (Serial number of unit)
Argument 4	Variable (Dependent upon firmware version fitted)
Response Available	Yes

**OPC – Operation Complete*

This will set the OPC bit in the Event Status Register when all pending operations have been completed. The operation query command (*OPC?) will return a 1 when all pending operations have been carried out.

Command Mnemonic	*OPC
Number of Arguments	1 (Response Only, No Arguments for Command)
Argument 1	1 when all pending operations completed
Response Available	Yes

**RST - Reset*

This has the same functionality as the INI message.

Command Mnemonic	*RST
Number of Arguments	0
Response Available	No

**SRE - Set the Service Request Enable Register*

Set the Service Request Enable Register

Command Mnemonic	*SRE
Number of Arguments	1
Argument 1	1 – Bit Mask as follows:- Bit 0 – Unused Bit 1 – Unused Bit 2 – Error Queue Has Data Bit 3 – Not Used Bit 4 – Message Available Bit 5 – Event Summary Status Bit 6 – 0 Bit 7 – Unused
Response Available	Yes

**STB? - Query the Status Byte Register*

Query the Status Byte Register

Command Mnemonic	*STB?
Number of Arguments	1 (Response Only, No Argument for Command)
Argument 1	1 – Bit Mask as follows:- Bit 0 – Unused Bit 1 – Unused Bit 2 – Error Queue Has Data Bit 3 – Not Used Bit 4 – Message Available Bit 5 – Event Summary Status Bit 6 – Master Summary Status Bit 7 – Unused
Response Available	Yes

**TST? – Self Test*

Perform Internal Self Test. This will result in any generator output being disabled.

Command Mnemonic	*TST?
Number of Arguments	1 (Response Only, No Arguments for Command)
Argument 1	0 = Self test Passed See Appendix for other valid responses.
Response Available	Yes

**WAI – Wait for Operation Complete*

Suspend further processing of commands until the OPC bit is set in the Standard Event Status Register.

Command Mnemonic	*WAI
Function	Wait for OPC
Number of Arguments	0
Response Available	No

4.2 DRIVER LEVEL INTERFACE

In order to ease the task of programming the Racal Instruments 2583 FRA system for specific custom tasks, a VXI*plug&play* driver has been created that includes function calls for each of the command level functions detailed in Section 4.1 above. This driver DLL may be used with common programming applications such as 'C' or Visual BASIC' in order to create a fully integrated test solution.

Details of the driver calls are in the driver documentation, which is included in its entirety in Appendix D - VXI*plug&play* Driver Interface User Manual

In addition, two examples of the use of the DLL are given in Appendix C - Driver Interface Function Examples. The first demonstrates how to establish a connection to the instrument, set the generator frequency and amplitude, perform a single measurement on channels 1 and 2 and then turn the generator off. The second demonstrates how to apply generator amplitude and sweep settings, how to run the frequency sweep and then turn the generator off.

4.3 SOFT FRONT PANEL INTERFACE

As well as the command and driver levels of interface, full instrument functionality may also be obtained through the soft front panel. This is a PC based application that may be used to drive the instrument from the desktop of a PC system. Other than the standard settings, the soft front panel allows more advanced functionality such as the ability to perform frequency sweeps and initiate closed loop control over specified measurement channels.

The calibration of the instrument and update of instrument firmware may also be achieved at this level. (By appropriate personnel.)

4.3.1 Software Installation

The soft front panel driver software is supplied on two floppy disks and may be installed by using the following instructions:

1. Insert 'Disk 1' into the PC's floppy drive and run the program titled 'setup.exe'.

2. The installation wizard will now guide the user through the installation procedure. It is recommended that a full installation be selected for most users.

3. Once installed, the application may be run by selecting the appropriate icon from the start-up tree, provided that a connection to a VXIbus mainframe slot zero controller is available.

4.3.2 Getting Started

4.3.2.1 Familiarization

Selecting the '2583' icon on the PC start-up tree will start the application. If multiple 2583 modules are present in the mainframe, a pull-down selection menu will be displayed enabling the user to connect to the required module. Once the application has been started, the display screen shown in Figure 4–1 is presented.

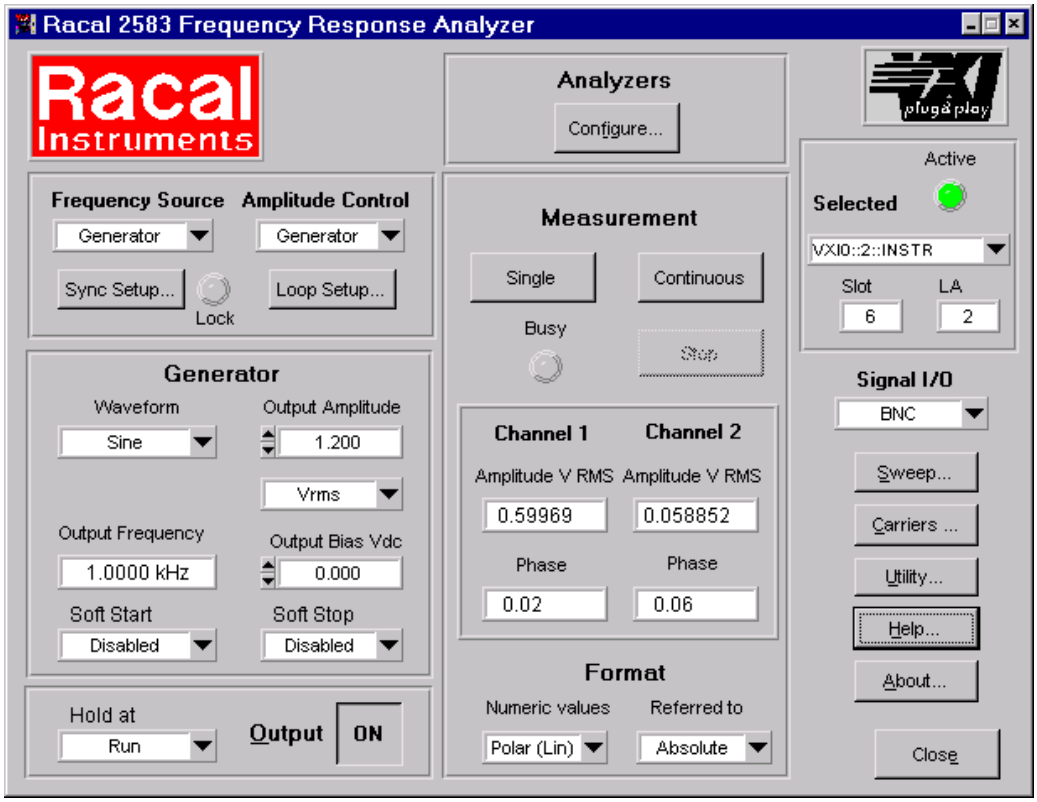


Figure 4–1 Soft Front Panel, Front Page

The 'soft front panel' front page is divided into the following distinct sections to allow simple operation of the system:

Generator Set-up: This section, to the left of the panel allows the generator 'Waveform', 'Bias', 'Amplitude' and 'Frequency' settings to be made. These settings are always used to determine the characteristics of the generator output signal, with the exception of the frequency setting, which may be over-ridden if an automated frequency sweep is performed. If a parameter is changed, the effect of the change will be applied immediately. Selection of alternative sources for frequency and amplitude control is also available through the ability to select the 'Synchronizer' and 'Carrier' input channels

Analyzer Set-up: This section contains a single 'Configure' button which leads to a subsequent panel allowing the configuration of global and channel specific analyzer parameters. Global parameters include the 'Measurement Delay', 'Integration Period', 'Auto-integration Period' and 'Measured Harmonic' settings. Channel specific parameters include the 'Channel Range', 'Channel Coupling' and the ability to enable or disable auto-integration.

Measurement: This section is used for the display of measurement information. The FRA may be required to perform either a single measurement, or may be used to measure continually using the current generator settings. In either case, the measurement results for both measurement channels will be updated simultaneously in the selected format, which may be either 'Cartesian', 'Polar (Linear)' or 'Polar (Logarithmic)'.

As well as these sections, sub-menus are available for the configuration of 'Sweeps', 'Carrier Channels' and 'Additional Utilities' such as instrument calibration, firmware update and instrument self test. The ability to select between the front panel connector types is also available.

The instrument status is shown in the top right hand corner of the display. If multiple 2583 modules are fitted to the mainframe rack, a pull down menu is provided in order to switch between the control of each system. The slot that the module occupies within the mainframe and the logical address of the device is also shown. (Preceded by "D" in demo mode.)

4.3.2.2 Automated Sweep Execution

The soft front panel enables the automated execution of either a frequency sweep whereby the selected generator output frequency is overridden by the sweep frequency being applied, or a harmonic sweep whereby the selected measurement harmonic value is overridden by the sweep harmonic being applied. Upon selection of the 'Sweep' button from the right hand side of the 'Soft Front Panel' front panel, the sweep dialog detailed in Figure 4–2 is displayed. This dialog enables the configuration of both of the sweep types and, for convenience, details the current generator settings as detailed on the front page. Once configured, the sweep is initiated by using the six function buttons on the top right hand corner of the dialog. These buttons allow the sweep to progress automatically, or allow each step of the sweep to be applied manually.

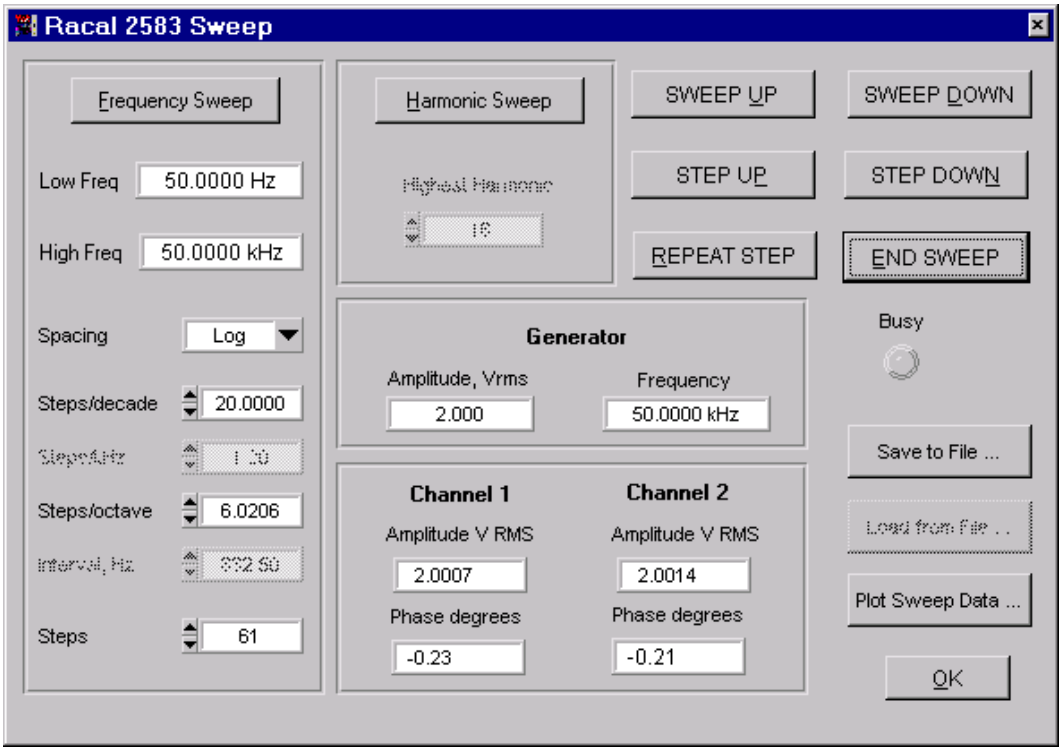


Figure 4–2 Sweep Configuration Dialog

Once a sweep has been run, options exist for the sweep results data to be either plotted to the PC printer or written to a data file in a comma-delimited format file (.csv), which may be read directly into common spreadsheet applications. The format of the data file output is as follows:

Row 1:	Sweep Type
Row 2 → n+1, Column 1:	Generator Amplitude (V rms)
Row 2 → n+1, Column 2:	Generator Frequency (Hz)
Row 2 → n+1, Column 3:	Channel 1 Magnitude
Row 2 → n+1, Column 4:	Channel 1 Phase
Row 2 → n+1, Column 5:	Channel 2 Magnitude
Row 2 → n+1, Column 6:	Channel 2 Phase

Where n is the end test in the sweep configuration.
Sweep Type 0 = Frequency sweep; Sweep Type 1 = Harmonic sweep

If the sweep data is to be plotted, a further dialog is displayed which allows the configuration of the display of the sweep data. Configuration parameters that may be modified include the type of graph (Nyquist/Bode), scale settings of the x-axis and y-axis, the axes to be displayed (magnitude/phase), the display type of each axes (logarithmic/linear) and the graph title. Once displayed, the graph may be output to the PC system plotter if required.

4.3.2.3 Performing A Single Measurement *Example*

The following example allows the user to set a fixed frequency and amplitude at the generator output, specify both of the analyzer channels for auto-ranging input, turn the generator on, perform a single measurement on channels 1 and 2 and then turn the generator off.

1. From the ‘soft front panel’ front panel *(*Figure 4–1*)*, enter ‘1.0’ in the ‘Output Amplitude’ field under the ‘Generator’ section, which will specify a generator output amplitude of 1V. This may be specified as 1V either rms, pk or pk-pk with the associated pull-down menu.

2. Enter a value of ’50.0’ in the ‘Output Frequency’ field under the ‘Generator’ section, which will specify a generator output frequency of 50Hz.

3. Select the ‘Configure’ button under the ‘Analyzers’ section to enter the analyzer’s configuration dialog. Now select the ‘Channel 1’ followed by the ‘Channel 2’ buttons in turn, to configure the input ranges of each channel to auto-ranging.

4. Using a BNC T-piece and two BNC to BNC leads, connect the ‘Generator Output’ to both the ‘Channel 1’ and ‘Channel 2’ measurement channels.

5. From the bottom of the ‘Generator’ section, use the ‘Output’ button to turn the generator ‘ON’.

6. In the measurement section, use the ‘Single’ button to perform a single measurement on both measurement channels. The results will be displayed under the ‘Channel 1’ and ‘Channel 2’ headings in the selected ‘Format’.

7. Once the results have been displayed, turn the generator ‘OFF’ by using the ‘Output’ button once again.

4.3.2.4 Performing a Frequency Sweep *Example*

The following example allows the user to configure and run a frequency sweep at a fixed generator amplitude of 2V rms over the frequency range of 50Hz to 50kHz using a sweep rate

of 20 steps/decade in an upwards direction. The sweep data will then be output both to a Bode plot hard copy and to a data file in a comma-delimited file format (*.csv).

1. From the 'soft front panel' front page *(*Figure 4–1*)*, enter '2.0' in the 'Output Amplitude' field under the 'Generator' section, which will specify a generator output amplitude of 2V. This must be then specified as 2V rms by selecting 'rms' from the associated pull down menu.
2. Using a BNC T-piece and two BNC to BNC leads, connect the 'Generator Output' to both the 'Channel 1' and 'Channel 2' measurement channels.
3. From the bottom of the 'Generator' section, use the 'Output' button to turn the generator 'ON'.
4. From the right hand side of the front panel, select the 'Sweeps' button to display the sweeps configuration dialog. This is shown in Figure 4–2
5. From the top left-hand corner of the dialog, select the 'Frequency Sweep' button to configure this type of sweep.
6. In the 'Low Freq.' field, enter the value of '50'. In the 'High Freq.' field, enter the value of '50,000'. Under the 'Spacing' filed, select 'Log' for a logarithmic sweep increment. Enter the value of '20' in the 'Steps/Decade' field; it may be noted that the 'Steps/Octave' and 'Steps' fields will automatically update upon this entry being made.
7. Select the 'Sweep Up' button from the top right hand corner of the dialog. This will start the automated frequency sweep. If required it may be stopped at any time by using the 'End Sweep' button. Note that, as the sweep progresses, the 'Generator' section is updated to show the current generator frequency and amplitude settings and that the 'Channel 1' and 'Channel 2' fields update to show the current recorded values. The 'Busy' indicator will also remain illuminated until the sweep is over.
8. Once the sweep has completed, the 'Busy' indicator will extinguish. The options to 'Save to file' and 'Plot sweep data' will become available. In order to save the data to a file, press the 'Save to File' button.
9. The 'Save sweep file' dialog will be displayed. To save the sweep data simply specify a file name and the target data directory and the file will be stored in a .csv format that is compatible with most spreadsheet applications.
10. To plot the sweep data; press the 'Plot Sweep Data' button, which will display the "Plot Setup Dialog" as detailed in Figure 4–3
11. At the 'Plot type' field select 'Bode (Lin)' and from the 'Plot data' field select 'Channel 1'. The Bode plot options section will now be active and manual-scaling values can be entered. At the 'Plot type' field, select 'Mag & Phase' and set the 'Frequency axis' type to 'Log'.
12. If required a title may also be entered into the 'Graph Title' field.
13. To display the plot; select the 'Display Plot' button and a plot similar to that shown in Figure 4–4 will be shown.
14. To produce a hard copy; simply select the print button in the bottom left hand corner of the dialog.

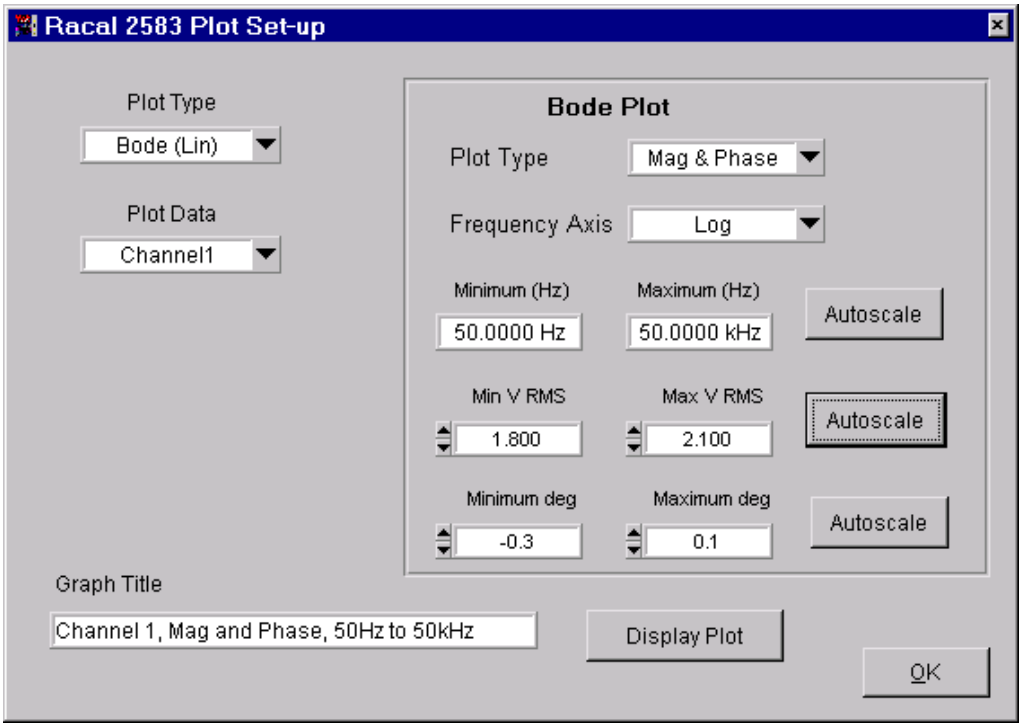


Figure 4–3 Plot Setup Dialog

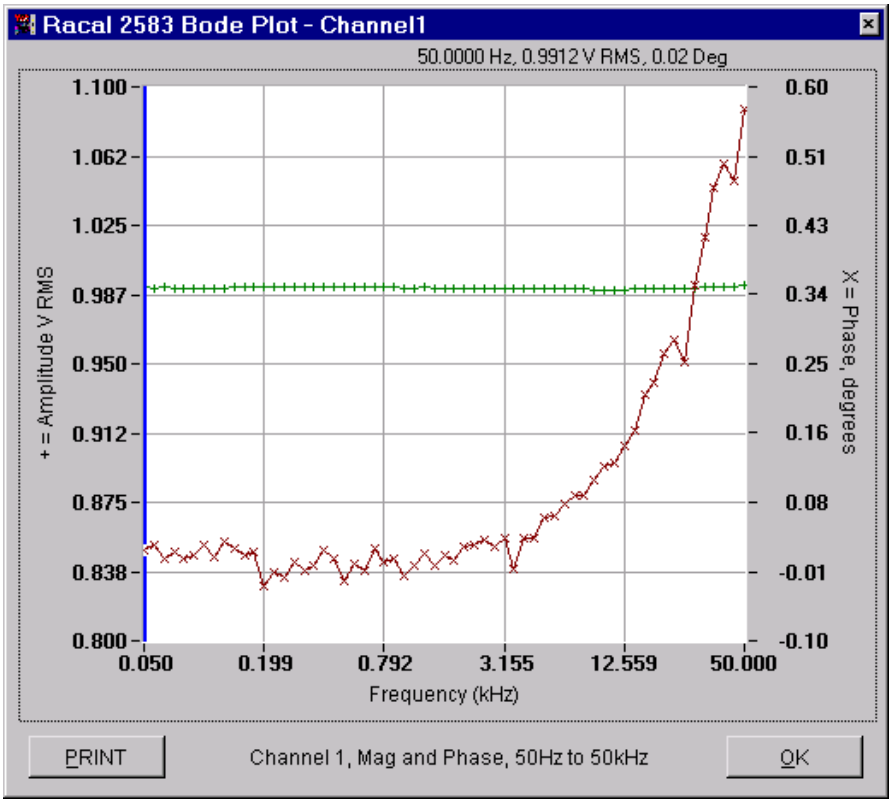


Figure 4–4 Frequency Sweep Plot

Chapter 5 – Utilities

5.1 MODULE CALIBRATION

The 2583 FRA module has no internal manual adjustments. Calibration of the 2583 FRA is fully automated and may be performed only with the use of the soft front panel interface.

Two options are available: either to 'Calibrate' or 'Verify' the calibration of the instrument:

- 'Verify' is selected - the full calibration routine is executed and a report is generated to detail deviance from the module specification at the end of testing. The report may be output to a text file in either a .txt or .rtf format. If a failure is detected, it will be signified with an asterisk next to the measured result. With the .rtf output file type, failures will also be identified in red.
- 'Calibrate' is selected - the system will first verify that the calibration switch on the rear panel is enabled. (See Figure 2–1). If it is not, an error message is displayed, and the calibration sequence aborted. Optionally, a pre-calibration verification may be run which allows a direct comparison of the calibration of the instrument both before and after the newly calculated correction values are applied. After all measurements are taken, the option to produce a hardcopy of the calibration status of the instrument is given, before the revised calibration factors are set. This allows the user to determine whether or not to apply the new calibration correction factors, upon inspection of the test data. If this option is used, the hardcopy will clearly show that the correction factors calculated for the instrument had not been applied at the time the hardcopy was generated. If the calibration is accepted, the option for a hardcopy is given once again, the hardcopy will now indicate that the calibration was applied.

In order to perform either a verification or calibration, the PC must be fitted with an IEEE 488 (referred to as GPIB, i.e. General Purpose Interface Bus) interface, for connection to a Wavetek 1271 or 1281 DMM. An external frequency generator is also required, the use of either a Racal Instruments 3151, 3152 or another 2583 FRA will allow a fully automated calibration process. If one of these instruments is not available, any other frequency source may be used; however, it will be necessary to enter the required frequency values manually by the operator.

The following steps may be performed in order to perform a verification/calibration on the instrument:

1. Load the 2583 soft front panel application onto the desktop of the PC system, connected to the controller of the VXIbus mainframe. The Wavetek 1271 DMM must be connected to the GPIB interface on the PC.

2. Once a connection to the target 2583 FRA has been established, select the 'Utility' button on the right hand side of the front page to display the list of available system utilities.

3. From this menu, select either the 'Verify' or 'Calibrate' option. If the calibrate option is requested, the user will be asked if a pre-calibration verification test is to be conducted, this optional test will allow the calibration status of the module to be recorded both prior to and after the new calibration correction factors are set.

4. Once the appropriate test has been selected, the system will automatically locate the

Wavetek 1271/1281 reference DMM. If this device is not connected to the GPIB interface, the calibration of the instrument cannot continue.

5. Once the DMM has been located, the system will automatically locate a compatible Racal Instruments 3151, 3152 or 2583 signal source through the VXIbus. If more than one compatible device is available a list will be issued from which the user may choose the desired signal source. If no suitable frequency generator is located the option to use a manual frequency source will be given.

6. Upon selection of the required test apparatus the necessary detail is displayed. The following connections must be made:
   - Connect the Generator output of the 2583 to the 'Channel 1' and 'Channel 2' inputs and to the voltage measurement input of the Wavetek 1271/1281 DMM.
   - Connect the output of the Racal Instruments 3151/3151/2583 signal source to the 2583 'Synchronizer', 'Carrier 1' and 'Carrier 2' inputs.
   - Connect to the 1271/1281 Wavetek via the GPIB interface.

   Important Note: In order to optimize the calibration process, it is important that each BNC interconnect cable is no longer than 1 meter. The use of longer cables may adversely affect the accuracy of the calibration.

7. When the required connections have been made and the dialog is accepted, the system will run a connectivity check and report if any errors are detected.

8. Assuming that the signal connections have been made according to the required detail, the calibration of the instrument will commence automatically. If a non-compatible frequency generator is used as the frequency source, the system will prompt for generator settings when required.

9. The full calibration routine takes approximately 40 minutes. Upon its completion, a dialog will be displayed detailing that either 'All measurements were within the specification' or that 'n calibration failures were recorded'. At this point three options are presented:
   - To produce a hard copy of the calibration results (prior to the results actually being applied)
   - To apply the calibration factors
   - To exit and abort the calibration routine. If a hard copy is produced, the reported values will be shown using the calculated correction factors, however, it is clearly indicated that the calibration was NOT applied. If the calibration is applied, the calculated correction factors will be immediately initiated.

10. Upon the application of the calibration factors, a further dialog will be produced; again the option will be given to make a hardcopy of the calibration results. If a hardcopy is produced, the calibration results will be output and the calibration time of calibration will be shown.

Unless the Calibration is 'Applied', no changes are made to the internal calibration of the instrument. The date of last calibration is displayed on the 'About' panel of the Soft Front Panel.

If calibration errors are detected, the module must be returned to the vendor for maintenance as detailed in Chapter 6 – Product Support. It must be noted that there are NO user-serviceable parts within the 2583 VXIbus module.

5.2 FIRMWARE UPDATE

When firmware updates become available, it is possible for them to be simply loaded into the module by the user. The firmware update is released on a PC format image file (*.ima) and may be downloaded to the flash memory devices within the FRA. (Only with the use of the soft front panel interface.)

The following steps may be performed in order to achieve this:

1. Load the 2583 soft front panel application onto the desktop of the PC system, connected to the controller of the VXIbus mainframe.

2. Once a connection to the target 2583 FRA has been established, select the ‘Utility’ button on the right hand side of the page to display the list of available utilities.

3. From the ‘Utilities’ page, select the ‘Download’ button in the ‘Firmware Update’ section. The source firmware image file may now be selected for downloading to the FRA module.

4. Upon selection of the source image file, the application will prompt the user to accept the download of the new firmware.

5. After accepting the download a message box ‘Downloading Firmware’ will be displayed throughout the download operation. When the operation is complete, a further confirmation dialog will be displayed.

6. Upon successful downloading of the instrument firmware the instrument must be restarted for the changes to be implemented. This may be achieved by resetting the mainframe rack. This may also require the controller to be restarted.

7. The firmware revision of the instrument may be verified by selecting the ‘About’ button on the bottom right hand side of the soft front panel after restarting the controller.

Chapter 6 – Product Support

6.1 PRODUCT SUPPORT

Racal Instruments has a complete Service and Parts Department. If you require technical assistance or should it be necessary to return your product for calibration or repair, contact Racal Instruments, Customer Support Department:

USA:	1-800-722-3262 or 1-949-859-8999	or FAX via 1-949-859-7309.
UK:	+44 (0)8706-080134	or FAX via +44 (0)1753-791290.

We can also be reached at helpdesk@racalinstruments.com or visit our website at http://www.racalinstruments.com for further information regarding your local Sales and Service Centers.

6.2 RESHIPMENT INSTRUCTIONS

Authorization is required from Racal Instruments before you send us your product for service or calibration. Call your nearest Racal Instruments support facility.

Use the original packing material when returning the module to Racal Instruments for calibration or servicing. The original shipping carton and the instrument's plastic foam will provide the necessary support for safe reshipment.

If the original packing material is unavailable, contact Racal Instruments Customer Service for information.

Reship in either the original or a new shipping carton.

Chapter 7 - Appendices

7.1 Appendix A - Message Command Error Codes

Error Code	*Description*
0	No Error
1	Command Has No Reply
2	Unknown Command
3	Missing Command Argument
4	Invalid Command Argument
5	Auto-integration Failed To Complete
6	Generator Frequency Is Invalid
7	Generator Harmonic Is Invalid
8	Generator Amplitude Is Invalid
9	Generator Bias Is Invalid
10	Generator Waveform Is Invalid
11	Generator Soft Start Parameter Is Invalid
12	Generator Soft Stop Parameter Is Invalid
13	Generator Output Hold Parameter Is Invalid
14	Synchronizer Select Parameter Is Invalid
15	Synchronizer Edge Is Invalid
16	Synchronizer Level Is Invalid
17	Measurement Integration Time Is Invalid
18	Measurement Integration Cycles Are Invalid
19	Integration Parameter Is Invalid
20	Measurement Delay Parameter Is Invalid
21	Measurement Delay Time Is Invalid
22	Measurement Delay Cycles Are Invalid
23	Auto-integration Type Is Invalid
24	Auto-integration Channel Is Invalid
25	Auto-integration Parameter Is Invalid
26	Measurement Range Channel Is Not Valid
27	Measurement Range Parameter Is Not Valid
28	Measurement Coupling Channel Is Not Valid

Table 7—1 Message Command Error Codes

Error Code	*Description*
29	Measurement Coupling Parameter Is Not Valid

30	Input Select Parameter Is Invalid
31	Generator Modulation Parameter Is Invalid
32	Demodulation Channel Is Invalid
33	Demodulation Parameter Is Invalid
34	Synchronizer Range Is Invalid
35	Calibration Type Is Not Valid
36	Unable To Allocate Memory For Calibration Request
37	Calibration Is Protected
38	Unable To Write Calibration
39	Synchronizer Enabled But No Synchronizer Signal
40	Unable To Allocate Internal Timer
41	Synchronizer Ratio Mode Gave Invalid Measurement Frequency
42	Flash Memory Record Checksum Incorrect
43	No Completed Flash Image Has Been Downloaded
44	Invalid Format Of Flash Download Record
45	Flash Download Without Start Command
46	Flash Image Too Large For Download
47	Flash Programming Failed
48	Carrier Channel Invalid
49	Carrier Range Invalid
50	Synchronizer Coupling Invalid
51	Internal Error – Measurement Failed
52	Measurement Frequency Not Allowed At This Harmonic
53	Failure During Initial Self Test
54	Failure During Self Test
55	An Internal Measurement Error Occurred
56	Hardware Failure On Relay Circuit
57	Synchronizer Ratio Value Out Of Range
58	Requested Carrier And Modulation Exceeded Maximum Amplitude
59	Synchronizer Input Frequency Change Too Large During Measurement
999	Error Queue Is Full

Table 7—2 Message Command Error Codes (cont.)

7.2 Appendix B - Self Test Failure Error Messages

Upon the failure of a self-test (low level command *TST?) an error message will be issued according to the result.

Error Code	*Description*
0	Self Test Passed
1	Channel 1 Failure in the 30mV Range
2	Channel 2 Failure in the 30mV Range
4	Channel 1 Failure in the 300mV Range
8	Channel 2 Failure in the 300mV Range
16	Channel 1 Failure in the 3V Range
32	Channel 2 Failure in the 3V Range
64	Channel 1 Failure in the 30V Range
128	Channel 2 Failure in the 30V Range
256	Channel 1 Failure in the 300V Range
512	Channel 2 Failure in the 300V Range
1024	Synchronizer Failure
2048	Internal Relays Failure

Table 7—3 Self Test Failure Error Messages

7.3 Appendix C - Driver Interface Function Examples

```
//====================================================================
// 2583Test.c             Example 1, Code for 2583 Driver
// ----------             -------------------------------
//
// Basic Generator Functions
//
// Makes a simple connection to a Racal 2583 FRA, then commands the 2583
// to set frequency and amplitude. Turns generator on and Performs a
// measurement on channels 1 and 2 then turns the Generator Off.
//
//====================================================================

#include <cvirte.h>
#include <stdio.h>
#include <stdlib.h>
#include <ri2583.h>

// Function to initialise the CVIRTE Interface
static void SetupCVIRTE(void)
{
    if(InitCVIRTE(0, 0, 0)==0)
    {
        printf("Cannot initialise CVIRTE.\n");
        exit(-1);
    }

    return;
}


//Function to Display Error Message
static void Display2583Error(ViSession FRAHandle, ViStatus status)
{
    ViChar errormessage[256];

    // Determine if Error Code is Valid.
    if (ri2583_error_message(FRAHandle, status, errormessage) == VI_SUCCESS)
        // If Error code is valid display Error Message
        printf("- Error Code: %s\n", errormessage);
    else
        // If Error code is not valid display Unknown Error Message
        printf("- Unknown Error message, code %x\n", status);

    return;
}


//Function to Initialise 2583
static ViSession    Locate2583(ViInt16 BoardNumber)
{
    ViStatus   status;
    ViSession  fraHandle;
    ViBoolean  unitsFound;

    printf ("Initialising 2583 Number %d\n", BoardNumber);

    // Locate and Initialise 2583 Modules
    status = ri2583_autoInitialize (BoardNumber, &unitsFound, &fraHandle);
```

```
    // Determine if 2583 can be located.
    If (status == VI_SUCCESS)
    {
        printf("Success, ");
        if (unitsFound == VI_TRUE)
        {
            // If 2583 module can be located Display Success Message
            printf("2583 Module(s) Located\n");
            return(fraHandle);
        }

        // If no 2583 module can be located Display Error Message
        printf("No 2583 Module(s) Located\n");
        CloseCVIRTE();
        exit(1);
    }

    Display2583Error(fraHandle, status);
    CloseCVIRTE();
    exit(1);
}


//Function to perform a measurement on channels 1 and 2.
static void MeasureQuery2583(ViSession FRAHandle)
{
    ViStatus   status;
    ViReal64   Hertz;
    ViReal64   Amplitude1;
    ViReal64   Phase1;
    ViReal64   Amplitude2;
    ViReal64   Phase2;
    ViInt32    ORRStatus;

    // Take a Measurement.
      status = ri2583_measureQuery(FRAHandle, &Hertz, &Amplitude1, &Phase1, &Amplitude2,
                    &Phase2, &ORRStatus);

    // Verify that measurement was successful
    if (status == VI_SUCCESS)
        // Display Measurements Taken.
        printf("Measured Freq %fHz\nCh 1 %fVrms at %fDeg\nCh 2 %fVrms at %fDeg\n", Hertz,
                    Amplitude1, Phase1, Amplitude2, Phase2);
    else
    {
        // If an Error has Occurred Display Error Message
        printf("- Reply:- NO RESPONSE\n");
        Display2583Error(FRAHandle, status);
    }

    return;
}


// Main Function.
int    main(void)
{
    ViSession  FRAHandle;
    ViReal64   Amplitude;
    ViReal64   Frequency;

    // Define Variables
    Amplitude = 2;
    Frequency = 10000;

    // Initialise the CVIRTE Interface
    SetupCVIRTE();

    // Initialise 2583(s)
    FRAHandle = Locate2583(1);
```

```
    // Set Frequency to 10kHz
    printf("Set Generator Frequency to 1kHz.\n");
    ri2583_generatorFrequency (FRAHandle, Frequency);

    // Set Amplitude to 2 Vrms
    printf("Set Generator Output to 2 Vrms.\n");
    ri2583_generatorAmplitude (FRAHandle, Amplitude);

    // Turn Generator On
    printf("Turn On Generator - \n\nOutput from Generator should be 10 kHz, 2V rms
                    Sinewave.\n");
    ri2583_generatorOutput(FRAHandle, RI2583_ON);

    // Take a Measurement.
    printf("Press Enter to take a measurement.\n");
    getchar();

    MeasureQuery2583(FRAHandle);

    // Turn Generator OFF
    printf("Press Enter to Turn Generator Off.\n");
    getchar();

    ri2583_generatorOutput(FRAHandle, RI2583_OFF);

    // Close 2583 Interface
    printf("Press Enter to Close Current Session.\n");
    getchar();

    ri2583_close(FRAHandle);

    // Close the CVIRTE Interface
    CloseCVIRTE();

    return(0);
    // end function main()
}
```

```
//====================================================================
// 2583Test.c             Example 2, Code for 2583 Driver
// ----------             -------------------------------
//
// Basic Frequency Sweep
//
// Makes a simple connection to a Racal 2583 FRA, then commands the 2583
// to set the amplitude and sweep settings. Turns the generator on and
// performs a frequency sweep, then turns the Generator Off.
//
//====================================================================

#include <cvirte.h>
#include <stdio.h>
#include <stdlib.h>
#include <ri2583.h>

// Define Varibles

#define NUMBEROFSTEPS 10


// Function to initialise the CVIRTE Interface
static void   SetupCVIRTE(void)
{
    if(InitCVIRTE(0, 0, 0)==0)
    {
        printf("Cannot initialise CVIRTE.\n");
        exit(-1);
    }

    return;
}


//Function to Display Error Message
static void Display2583Error(ViSession FRAHandle, ViStatus status)
{
    ViChar   errormessage[256];

    // Determine if Error Code is Valid.
    if (ri2583_error_message(FRAHandle, status, errormessage) == VI_SUCCESS)
    {
        // If Error code is valid display Error Message
        printf("- Error Code: %s\n", errormessage);
    }
    else
    {
        // If Error code is not valid display Unknown Error Message
        printf("- Unknown Error message, code %x\n", status);
    }

    return;
}


//Function to Initialise 2583
static ViSession Locate2583(ViInt16 BoardNumber)
{
    ViStatus  status;
    ViSession fraHandle;
    ViBoolean unitsFound;

    printf ("Initialising 2583 Number %d\n", BoardNumber);

    // Locate and Initialise 2583 Modules
    status = ri2583_autoInitialize (BoardNumber, &unitsFound, &fraHandle);
```

```
    // Determine if 2583 can be located.
    If (status == VI_SUCCESS)
    {
        printf("Success, ");
        if (unitsFound == VI_TRUE)
        {
            // If 2583 module can be located Display Success Message
            printf("2583 Module(s) Located\n");
            return(fraHandle);
        }

        // If no 2583 module can be located Display Error Message
        printf("No 2583 Module(s) Located\n");
        CloseCVIRTE();
        exit(1);
    }

    Display2583Error(fraHandle, status);
    CloseCVIRTE();
    exit(1);
}


//Function to Perform a Frequency Sweep
static void FrequencySweep2583(ViSession FRAHandle, ViInt16 mode)
{
    ViStatus   status;
    ViReal64   Frequency[NUMBEROFSTEPS];
    ViReal64   Amplitude1[NUMBEROFSTEPS];
    ViReal64   Amplitude2[NUMBEROFSTEPS];
    ViReal64   Phase1[NUMBEROFSTEPS];
    ViReal64   Phase2[NUMBEROFSTEPS];
    ViInt32    ORRStatus;
    ViChar     outofrangeMessage[256];
    Int        i;

    //Perform Sweep Then Display Results if Success
    status = ri2583_sweepFrequencyQuery(FRAHandle, mode, Frequency,  Amplitude1, Phase1,
              Amplitude2, Phase2, &ORRStatus);

    If (status == VI_SUCCESS)
    {
        for (i = 0; I != NUMBEROFSTEPS; i++)
        {
            printf("%f, %f, %f, %f, %f\n", Frequency[i], Amplitude1[i], Phase1[i],
                    Amplitude2[i], Phase2[i]);
        }

        while (ORRStatus != 0)
        {
            ri2583_outofrange_message(&ORRStatus, outofrangeMessage);
            printf("Overrange %s\n", outofrangeMessage);
        }
    }
    else
        Display2583Error(FRAHandle, status);

    return;
}


// Main Function.
int    main(void)
{
    // Define Variables
    ViSession  FRAHandle;
    ViReal64   Amplitude = 2;
    ViReal64   Frequency1 = 10;
    ViReal64   Frequency2 = 1000;

    // Initialise the CVIRTE Interface
    SetupCVIRTE();
```

```
    // Initialise 2583(s)
    FRAHandle = Locate2583(1);

    // Set up Frequency sweep
    printf("Set Frequency Sweep to sweep from %f Hz to %f Hz in %d Steps.\n", Frequency1,
            Frequency2, NUMBEROFSTEPS);
    ri2583_frequencySweepSetup(FRAHandle, Frequency1, Frequency2, NUMBEROFSTEPS,
            RI2583_LINEAR);

    // Set Amplitude to 2 Vrms
    printf("Set Generator Output to 2 Vrms.\n");
    ri2583_generatorAmplitude (FRAHandle, Amplitude);

    // Turn Generator On
    printf("Turn On Generator.\n");
    ri2583_generatorOutput(FRAHandle, RI2583_ON);

    // Take a Measurement
    printf("Press Enter to Perform a Sweep.\n");
    getchar();

    FrequencySweep2583(FRAHandle, RI2583_SWEEPUP);

    // Turn Generator OFF
    printf("Press Enter to Turn Generator Off.\n");
    getchar();

    ri2583_generatorOutput(FRAHandle, RI2583_OFF);

    // Close 2583 Interface
    printf("Press Enter to Close Current Session.\n");
    getchar();

    ri2583_close(FRAHandle);

    // Close the CVIRTE Interface
    CloseCVIRTE();

    return(0);
} // end function main()
```

7.4 Appendix D - VXI*plug&play* Driver Interface User Manual

Racal 2583, Frequency Response Analyzer

Introduction:

This instrument driver provides programming support for Racal 2583, Frequency Response Analyzer.

It contains functions for opening, configuring, taking measurements from, and closing the instrument.

Assumptions:

To use this module successfully, the following conditions must be met:

For GPIB instrument drivers:

- the instrument is connected to the GPIB.
- the GPIB address supplied to the initialize function must match the GPIB address of the instrument.

For VXI instrument drivers:

- the instrument is installed in the VXI mainframe and you are using one of the following controller options:
  - Embedded controller
  - MXI
  - MXI2
  - GPIB-VXI
- the logical address supplied to the initialize function must match the logical address of the instrument.

For RS-232 instrument drivers:

- the instrument is connected to the RS-232 interface.
- the COM port, baud rate, parity, and timeout supplied to the initialize function must match the settings of the instrument.

Error and Status Information:

Each function in this instrument driver returns a status code that either indicates success or describes an error or warning condition.
Your program should examine the status code from each call to an instrument driver function to determine if an error occurred.
The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

The description of each instrument driver function lists possible error codes and their meanings

---

How To Use This Document:

Use this document as a programming reference manual.
It describes each function in the

Racal 2583, Frequency Response Analyzer

instrument. The functions appear in alphabetical order, with a description of

the function and its C syntax, a description of each parameter, and a list of possible error codes.

Function Tree Layout:

Class/Panel Name:	Function Name:
Connection	
Auto Initialize	ri2583_autoInitialize
Get Resource Name	ri2583_nameEntry
Initialize	ri2583_init
Location	ri2583_location
Close	ri2583_close
Configuration	
Analyzer Autointegrate Mode	ri2583_analyzerAutoIntegrate
Analyzer Delay	ri2583_analyzerDelay
Analyzer Harmonic	ri2583_analyzerHarmonic
Analyzer Integration Period	ri2583_analyzerIntegrate
Carrier Range	ri2583_carrierRange
Channel Autointegrate Mode	ri2583_channelAutoIntegrate
Channel Coupling	ri2583_channelCouple
Channel Demodulation	ri2583_channelDemodulate
Channel Voltage Range	ri2583_channelRange
Closed Loop Enable	ri2583_closedLoopEnable
Closed Loop Setup	ri2583_closedLoopSetup
Frequency Sweep Setup	ri2583_frequencySweepSetup
Generator Amplitude	ri2583_generatorAmplitude
Generator Bias	ri2583_generatorBias
Generator Carrier Amplitude	ri2583_generatorCarrierAmplitude
Generator Frequency	ri2583_generatorFrequency
Generator Function	ri2583_generatorFunction
Generator Modulation	ri2583_generatorModulate
Generator Modulation Amplitude	ri2583_generatorModulationAmplitude
Generator Soft Start	ri2583_generatorSoftStart
Generator Soft Stop	ri2583_generatorSoftStop
Generator Sweep Frequency	ri2583_generatorSweepFrequency
Select Front-panel Connector	ri2583_selectConnector
Synchronization Configure	ri2583_syncConfigure
Synchronization Enable	ri2583_syncEnable
Action/Measurement Functions	
Analyzer Cartesian Query	ri2583_analyzerCartesianQuery
Analyzer Polar Query	ri2583_analyzerPolarQuery
Analyzer Trigger	ri2583_analyzerTrigger
Generator Hold	ri2583_generatorHold
Generator Output	ri2583_generatorOutput
Perform Complete Measurement	ri2583_measureQuery
Perform Frequency Sweep	ri2583_sweepFrequencyQuery
Perform Harmonic Sweep	ri2583_sweepHarmonicQuery
Status Functions	
Analyzer Autointegrate Query	ri2583_analyzerAutoIntegrateQuery
Analyzer Delay Query	ri2583_analyzerDelayQuery
Analyzer Harmonic Query	ri2583_analyzerHarmonicQuery
Analyzer Integration Query	ri2583_analyzerIntegrateQuery
Analyzer Status Query	ri2583_analyzerStatusQuery
Carrier Amplitude Query	ri2583_carrierAmplitudeQuery
Carrier Range Query	ri2583_carrierRangeQuery
Channel Autointegration Query	ri2583_channelAutoIntegrateQuery
Channel Input Coupling Query	ri2583_channelCoupleQuery
Channel Demodulation Query	ri2583_channelDemodulateQuery
Channel Range Query	ri2583_channelRangeQuery
Check for Measurement Complete	ri2583_measureCompleteQuery
Closed Loop Query	ri2583_closedLoopQuery
Frequency Sweep Query	ri2583_frequencySweepQuery
Generator Amplitude Query	ri2583_generatorAmplitudeQuery
Generator Bias Query	ri2583_generatorBiasQuery

```
        Generator Frequency Query          ri2583_generatorFrequencyQuery
        Generator Function Query           ri2583_generatorFunctionQuery
        Generator Hold Query               ri2583_generatorHoldQuery
        Generator Output Query             ri2583_generatorOutputQuery
        Generator Soft Start Query         ri2583_generatorSoftStartQuery
        Generator Soft Stop Query          ri2583_generatorSoftStopQuery
        Modulation Amplitude Query         ri2583_modulationAmplitudeQuery
        Modulation Source Query            ri2583_modulateQuery
        Select Connector Query             ri2583_selectConnectorQuery
        Sync Configure Query               ri2583_syncConfigureQuery
        Sync Enable Query                  ri2583_syncEnableQuery
        Sync Locked Query                  ri2583_syncLockedQuery
    Utilities
        Error Message Decode               ri2583_error_message
        Error Query                        ri2583_error_query
        Out Of Range Decode                ri2583_outofrange_message
        Query                              ri2583_query_read
        Query Event Status Register        ri2583_queryESR
        Query Out of Range Register        ri2583_queryORR
        Query Status Byte                  ri2583_querySTB
        Read Message                       ri2583_read
        Reset                              ri2583_reset
        Revision Query                     ri2583_revision_query
        Self-Test                          ri2583_self_test
        Send Clear Status                  ri2583_sendCLS
        Send Operation Complete            ri2583_sendOPC
        Serial Number Query                ri2583_serial_query
        Write Message                      ri2583_write
```

Racal 2583, Frequency Response Analyzer

Racal Instruments Limited
480 Bath Road
Burnham
Slough
Berkshire
SL1 6BE

```
Filename                      : RI2583.FP
Software Number               : 60-0109
Issue                         : 1.2
Minimum Instrument Revision   : 1.02
Date                          : 25-May-2001
```

This instrument driver provides programming support for the Racal Instruments 2583, 2-channel frequency response analyzer.

The driver is divided into the following function classes:

(1) Connection: (Class)

Used to connect and disconnect from RI2583 VXI modules – via the allocation and deallocation of VISA instrument sessions.

Auto-Initializing an instrument sets it to a default configuration.

(2) Configuration Functions: (Class)

Configure the instrument signal generation and measurement settings.

(3) Action/Measurement Functions: (Class)

Turn the signal generator output ON and OFF, initiate measurements, and read measurement results.

(4) Status Functions: (Class)

Report the current configuration of the instrument signal generation and measurement settings.

(5) Utility Functions: (Class)

Implement useful IEEE 488.2 Common Commands, and provide means for communicating ASCII strings to and from the instrument.

The following functions are in alphabetical order.

---

ri2583_analyzerAutoIntegrate

ViStatus ri2583_analyzerAutoIntegrate (ViSession instrumentHandle, ViInt16 mode);

Purpose

Select or deselect Auto-integration.

Auto-integration is a technique whereby the analyzer determines the standard deviation of the samples taken, and integrates until the estimate of deviation at the 90% confidence level is:
10% of sample mean for 'SHORT' auto-integration selected.
1% of sample mean for 'LONG' auto-integration selected.

Integration time for any given signal will vary between 3 and 30000 cycles of generator fundamental.

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

mode

Variable Type ViInt16

Select or deselect the Auto-integration mode:

RI2583_OFF defined as 0
RI2583_SHORT defined as 1
RI2583_LONG defined as 2

Return Value

The status code returned by the function call:

VI_SUCCESS (0) - No error, the instrument has been re-configured.

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_AUTOINT - parameter 'mode' has an unrecognized value.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_analyzerAutoIntegrateQuery

ViStatus ri2583_analyzerAutoIntegrateQuery (ViSession instrumentHandle, ViPInt16 mode);

Purpose

Report on the status of analyzer Auto-integration.

```
Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    mode
        Variable Type       ViInt16 (passed by reference)

        Reflect the currently selected Auto-integration mode:

        RI2583_OFF     defined as 0
        RI2583_SHORT   defined as 1
        RI2583_LONG    defined as 2

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)         - No error.

        RI2583_ERROR_HANDLE    - does not correspond to an open session
        RI2583_ERROR_POINTER   - parameter 'mode' is referenced by a null
                                 pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_analyzerCartesianQuery

    ViStatus ri2583_analyzerCartesianQuery (ViSession instrumentHandle,
            ViPReal64 frequency, ViPReal64 real1, ViPReal64 complex1,
            ViPReal64 real2, ViPReal64 complex2);

Purpose
    Read the most recent measurement from the analyzer.

Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    frequency
        Variable Type       ViReal64 (passed by reference)

        The frequency setting of the generator in Hz.

    real1
        Variable Type       ViReal64 (passed by reference)

        The real (in-phase) rms voltage measured on channel 1.

    complex1
        Variable Type       ViReal64 (passed by reference)

        The complex (quadrature) rms voltage measured on channel 1.

    real2
        Variable Type       ViReal64 (passed by reference)

        The real (in-phase) rms voltage measured on channel 2.

    complex2
        Variable Type       ViReal64 (passed by reference)

        The complex (quadrature) rms voltage measured on channel 1.

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)         - No error, values have been returned.

        RI2583_ERROR_HANDLE    - does not correspond to an open session
```

```
        RI2583_ERROR_POINTER   - one or more output parameters is referenced by
                                 a null pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_analyzerDelay

    ViStatus ri2583_analyzerDelay (ViSession instrumentHandle, ViInt16 mode,
            ViReal64 delay);

  Purpose
    Specify the delay between any change in generator status and the start of
    measurement.

  Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    mode
        Variable Type       ViInt16

        Determine how delay is specified:

        RI2583_SECONDS defined as 0
        RI2583_CYCLES  defined as 1

    delay
        Variable Type       ViReal64

        Set the delay in seconds or cycles of generator fundamental according
        to parameter 'mode'.

        Acceptable values when converted to cycles of fundamental are between
        0 and 100,000

        The maximum limit of 100,000 cycles applies even if the period is
        specified in seconds. The unit will use the least of the specified
        time, or 100,000 cycles.

  Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)           - No error, the instrument has been re-
                                   configured.

        RI2583_ERROR_HANDLE      - does not correspond to an open session
        RI2583_ERROR_DELAY_MODE  - parameter 'mode' has an unrecognized value.
        RI2583_ERROR_DELAY_TIME  - parameter 'delay' is outside the valid
                                   range.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_analyzerDelayQuery

    ViStatus ri2583_analyzerDelayQuery (ViSession instrumentHandle, ViPInt16
            mode, ViPReal64 delay);

  Purpose
    Report the delay between any change in generator status and the start of
    measurement.

  Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.
```

```
    mode
        Variable Type       ViInt16 (passed by reference)

        Return the Units of the Measurement Delay:

        RI2583_SECONDS defined as 0
        RI2583_CYCLES  defined as 1

    delay
        Variable Type       ViReal64 (passed by reference)

        Return the delay in seconds or cycles of generator fundamental.

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)          - No error, the instrument has been re-
                                  configured.

        RI2583_ERROR_HANDLE     - does not correspond to an open session
        RI2583_ERROR_POINTER    - parameter 'delay' is referenced by a null
                                  pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_analyzerHarmonic

    ViStatus ri2583_analyzerHarmonic (ViSession instrumentHandle, ViInt16
            harmonic);

Purpose
    Configure the analyzer to measure an harmonic of the generator frequency.

Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    harmonic
        Variable Type       ViInt16

        Determine the harmonic to be measured, in the range 1 (fundamental) to
        16 (highest harmonic) inclusive.

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)          - No error (the call was successful).

        RI2583_ERROR_HANDLE     - does not correspond to an open session
        RI2583_ERROR_HARMONIC   - parameter 'nvalue' is outside the permitted
                                  range.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_analyzerHarmonicQuery

    ViStatus ri2583_analyzerHarmonicQuery (ViSession instrumentHandle,
            ViPInt16 harmonic);

Purpose
    Report which harmonic of the generator frequency the analyzer is
    configured to measure.

    In normal use this will be 1.

Parameter List
    instrumentHandle
        Variable Type       ViSession
```

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

harmonic
Variable Type ViInt16 (passed by reference)

Return the harmonic to be measured, in the range 1 (fundamental) to 16 (highest harmonic) inclusive.

Return Value
The status code returned by the function call:

VI_SUCCESS (0)	- No error (the call was successful).
RI2583_ERROR_HANDLE	- does not correspond to an open session
RI2583_ERROR_POINTER	- parameter 'harmonic' is referenced by a NULL pointer.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_analyzerIntegrate

```
ViStatus ri2583_analyzerIntegrate (ViSession instrumentHandle, ViInt16
        mode, ViReal64 period);
```

Purpose
Specify the period over which the analyzer channels perform the measurement.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

mode
Variable Type ViInt16

Determine how period is specified:

RI2583_SECONDS defined as 0
RI2583_CYCLES defined as 1

period
Variable Type ViReal64

Set the integration period in seconds or cycles of generator fundamental according to parameter 'mode'.

Acceptable values when converted to cycles of fundamental are between 1 and 100,0000

The maximum limit of 100,000 cycles applies even if the period is specified in seconds. The unit will use the least of the specified time, or 100,000 cycles.

Return Value
The status code returned by the function call:

VI_SUCCESS (0)	- No error, the instrument has been re-configured.
RI2583_ERROR_HANDLE	- does not correspond to an open session
RI2583_ERROR_INT_MODE	- parameter 'mode' has an unrecognized value.
RI2583_ERROR_INT_TIME	- parameter 'period' is outside the valid range.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_analyzerIntegrateQuery

```
ViStatus ri2583_analyzerIntegrateQuery (ViSession instrumentHandle,
        ViPInt16 mode, ViPReal64 period);
```

Purpose

Report the specified analyzer integration period in seconds or cycles of generator fundamental.

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

mode

Variable Type ViInt16 (passed by reference)

period

Variable Type ViReal64 (passed by reference)

Return the integration period in seconds or cycles of generator fundamental.

Return Value

The status code returned by the function call:

VI_SUCCESS (0)	- No error, the instrument has been re-configured.
RI2583_ERROR_HANDLE	- does not correspond to an open session
RI2583_ERROR_POINTER	- parameter 'period' is referenced by a null pointer.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_analyzerPolarQuery

```
ViStatus ri2583_analyzerPolarQuery (ViSession instrumentHandle, ViPReal64
        frequency, ViPReal64 amplitude1, ViPReal64 phase1, ViPReal64
        amplitude2, ViPReal64 phase2);
```

Purpose

Read the most recent measurement from the analyzer.

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

frequency

Variable Type ViReal64 (passed by reference)

The frequency setting of the generator in Hz.

amplitude1

Variable Type ViReal64 (passed by reference)

The amplitude in V rms measured on channel 1.

phase1

Variable Type ViReal64 (passed by reference)

The phase in degrees measured on channel 1.

amplitude2

Variable Type ViReal64 (passed by reference)

The amplitude in V rms measured on channel 2.

```
    phase2
        Variable Type       ViReal64 (passed by reference)

        The phase in degrees measured on channel 2.

  Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)          - No error, values have been returned.

        RI2583_ERROR_HANDLE     - does not correspond to an open session
        RI2583_ERROR_POINTER    - one or more output parameters is referenced by
                                  a null pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

---

ri2583_analyzerStatusQuery

```
    ViStatus ri2583_analyzerStatusQuery (ViSession instrumentHandle, ViPInt16
            state);

  Purpose
    Determine whether a measurement result is available, or imminent because
    of a recent trigger.

  Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    state
        Variable Type       ViInt16 (passed by reference)

        Return the analyzer state.

        RI2583_READY denotes that the instrument is not currently making a
        measurement.

        RI2583_BUSY denotes that the instrument is currently making a
        measurement.

        RI2583_READY  defined as 0
        RI2583_BUSY   defined as 1

  Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)              - No error, the reply is valid.

        RI2583_ERROR_HANDLE         - does not correspond to an open session
        RI2583_ERROR_POINTER        - parameter 'state' is referenced by a null
                                      pointer.
        RI2583_ERROR_REPLY_FORMAT - invalid reply string from instrument.

        To translate errors into text string form use the function
        ri2583_error_message().
```

---

ri2583_analyzerTrigger

```
    ViStatus ri2583_analyzerTrigger (ViSession instrumentHandle, ViInt16
            mode);

  Purpose
    Trigger or abort the analyzer reading process.

    The most recently completed reading may be recovered via a call to
    ri2583_analyzerCartesianQuery() or ri2583_analyzerPolarQuery(), even after
    aborting the current reading process.
```

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

mode

Variable Type ViInt16

Determine whether the reading process is initiated or cancelled.

Permitted values:

RI2583_CANCEL defined as 0
RI2583_TRIGGER defined as 1

Return Value

The status code returned by the function call:

VI_SUCCESS (0) - No error.

RI2583_ERROR_HANDLE - does not correspond to an open session.
RI2583_ERROR_TRIG_MODE - parameter 'mode' outside permitted range.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_autoInitialize

ViStatus ri2583_autoInitialize (ViInt16 unitNumber, ViPBoolean unitsFoundPtr, ViPSession ptrThisSession);

Purpose

Automatically attempt to locate and connect to a unit.

It will find all units in the system and will connect to and return a handle to the unit specified by the unit number.

Notes:
(1) Each time this function is invoked a Unique Session is opened.
(2) It is not advisable to have more than one session open for the same resource.

Parameter List

unitNumber

Variable Type ViInt16

Indicate to which unit to connect in the case where there are multiple units in the system.

If a value greater than zero is passed, then the corresponding 2583 module will be connected to.

If -1 is passed, then no connection will be made. However, all units will be located, for identification with the ri2583_nameEntry() function.

unitsFoundPtr

Variable Type ViBoolean (passed by reference)

This returns VI_TRUE if any ri2583 unit was found in the system, VI_FALSE if none.

ptrThisSession

Variable Type ViSession (passed by reference)

Return an Instrument Handle that is used in all subsequent function calls to differentiate between sessions of this instrument driver.

Notes:
(1) Each time this function is invoked a Unique Session is opened.
(2) It is not advisable to have more than one session open for the same resource.

```
Return Value
       The status code returned by the function call:

       VI_SUCCESS (0)          - No error, fraHandle is a valid handle.

       RI2583_WARN_RESOURCE    - a session to this resource is already open

       RI2583_ERROR_SESSIONS   - 8 sessions are already open
       RI2583_ERROR_NOT_FOUND  - no 2583 at this unit number.
       RI2583_ERROR_POINTER    - a return parameter is referenced by a NULL
                                 pointer.

       To translate errors into text string form use the function
       ri2583_error_message().
```

```
    ri2583_carrierAmplitudeQuery

    ViStatus ri2583_carrierAmplitudeQuery (ViSession instrumentHandle,
            ViPReal64 volts);

Purpose
    Report the currently programmed amplitude of the output carrier when the
    generator is in modulation mode.

Parameter List
    instrumentHandle
       Variable Type       ViSession

       The Instrument Handle returned by ri2583_init() or
       ri2583_autoInitialize() to open this instrument driver session.

    volts
       Variable Type       ViReal64 (passed by reference)

       Return the carrier amplitude in V rms.

Return Value
       The status code returned by the function call:

       VI_SUCCESS (0)          - No error (the call was successful).

       RI2583_ERROR_HANDLE     - does not correspond to an open session
       RI2583_ERROR_POINTER    - parameter 'volts' is referenced by a null
                                 pointer.

       To translate errors into text string form use the function
       ri2583_error_message().
```

```
    ri2583_carrierRange

    ViStatus ri2583_carrierRange (ViSession instrumentHandle, ViInt16 carrier,
            ViInt16 range);

Purpose
    Select range attenuation of specified carrier input channel. Two ranges
    are available, 250V rms and 25V rms.

Parameter List
    instrumentHandle
       Variable Type       ViSession

       The Instrument Handle returned by ri2583_init() or
       ri2583_autoInitialize() to open this instrument driver session.

    carrier
       Variable Type       ViInt16

       Determine which carrier input is to be configured:

       RI2583_MODINPUT1  defined as 1
       RI2583_MODINPUT2  defined as 2
```

```
    range
       Variable Type       ViInt16

       Select appropriate range attenuation for the specified carrier input.

       RI2583_MAX250V   defined as 0
       RI2583_MAX25V    defined as 1

  Return Value
       The status code returned by the function call:

       VI_SUCCESS (0)           - No error, the instrument has been re-
                                  configured.

       RI2583_ERROR_HANDLE      - does not correspond to an open session
       RI2583_ERROR_MODULATION  - parameter 'carrier' has an unrecognised
                                  value.
       RI2583_ERROR_MAXIN       - parameter 'range' has an unrecognised
                                  value.

       To translate errors into text string form use the function
       ri2583_error_message().
```

---

ri2583_carrierRangeQuery

```
    ViStatus ri2583_carrierRangeQuery (ViSession instrumentHandle, ViInt16
            carrier, ViPInt16 state);

  Purpose
     Report which input range is selected for the specified channel.

  Parameter List
     instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

     carrier
        Variable Type       ViInt16

        Determine which carrier input is to be configured:

        RI2583_MODINPUT1  defined as 1
        RI2583_MODINPUT2  defined as 2

     state
        Variable Type       ViInt16 (passed by reference)

        Return range state for the specified carrier input.

        RI2583_MAX300V   defined as 0
        RI2583_MAX30V    defined as 1

  Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)          - No error.

        RI2583_ERROR_HANDLE     - does not correspond to an open session
        RI2583_ERROR_MODULATION - parameter 'carrier' has an unrecognised
                                  value.
        RI2583_ERROR_POINTER    - parameter 'range' is referenced by a NULL
                                  pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

---

ri2583_channelAutoIntegrate

ViStatus ri2583_channelAutoIntegrate (ViSession instrumentHandle, ViInt16 channel, ViInt16 auto);

Purpose
Enable or disable analyzer Auto-integration of measurements on the specified channel.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

channel
Variable Type ViInt16

Determine which channel is to be configured:

RI2583_CHANNEL1 defined as 1
RI2583_CHANNEL2 defined as 2

auto
Variable Type ViInt16

Disable or enable Auto-integration for the specified channel.

RI2583_OFF defined as 0
RI2583_ON defined as 1

Return Value
The status code returned by the function call:

VI_SUCCESS (0) - No error, the instrument has been re-configured.

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_CHANNEL - parameter 'channel' has an unrecognised value.
RI2583_ERROR_ON_OFF - parameter 'auto' has an unrecognised value.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_channelAutoIntegrateQuery

ViStatus ri2583_channelAutoIntegrateQuery (ViSession instrumentHandle, ViInt16 channel, ViPInt16 state);

Purpose
Report whether the specified channel is configured for analyzer Auto-integration.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

channel
Variable Type ViInt16

Determine which channel is to be reported:

RI2583_CHANNEL1 defined as 1
RI2583_CHANNEL2 defined as 2

state
Variable Type ViInt16 (passed by reference)

Return Auto-integration state for the specified channel.

```
        RI2583_OFF   defined as 0
        RI2583_ON    defined as 1

  Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)          - No error, the instrument has been re-
                                  configured.

        RI2583_ERROR_HANDLE     - does not correspond to an open session
        RI2583_ERROR_CHANNEL    - parameter 'channel' has an unrecognised value.
        RI2583_ERROR_POINTER    - parameter 'auto' is referenced by a NULL
                                  pointer

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
     ri2583_channelCouple

     ViStatus ri2583_channelCouple (ViSession instrumentHandle, ViInt16
             channel, ViInt16 couple);

  Purpose
     Configure the AC/DC coupling of the specified channel.

  Parameter List
     instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

     channel
        Variable Type       ViInt16

        Determine which channel is to be configured:

        RI2583_CHANNEL1   defined as 1
        RI2583_CHANNEL2   defined as 2

     couple
        Variable Type       ViInt16

        Select the measurement couple for the specified channel.

        RI2583_COUPLEAC   defined as 0
        RI2583_COUPLEDC   defined as 1

  Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)          - No error, the instrument has been re-
                                  configured.

        RI2583_ERROR_HANDLE     - does not correspond to an open session
        RI2583_ERROR_CHANNEL    - parameter 'channel' has an unrecognised value.
        RI2583_ERROR_COUPLING   - parameter 'couple' has an unrecognised value.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
     ri2583_channelCoupleQuery

     ViStatus ri2583_channelCoupleQuery (ViSession instrumentHandle, ViInt16
             channel, ViPInt16 couple);

  Purpose
     Report the state of input AC/DC coupling of the specified channel.

  Parameter List
     instrumentHandle
        Variable Type       ViSession
```

```
        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    channel
        Variable Type       ViInt16

        Determine which channel is to be reported:

        RI2583_CHANNEL1   defined as 1
        RI2583_CHANNEL2   defined as 2

    couple
        Variable Type       ViInt16 (passed by reference)

        Return the currently selected input coupling for the specified
        channel.

        RI2583_COUPLEDC   defined as 0
        RI2583_COUPLEAC   defined as 1

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)          - No error, the instrument has been re-
                                  configured.

        RI2583_ERROR_HANDLE     - does not correspond to an open session
        RI2583_ERROR_CHANNEL    - parameter 'channel' has an unrecognised value.
        RI2583_ERROR_POINTER    - parameter 'couple' is referenced by a null
                                  pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_channelDemodulate

    ViStatus ri2583_channelDemodulate (ViSession instrumentHandle, ViInt16
            channel, ViInt16 modSource);

Purpose
    Select or deselect demodulation of the signal on the specified channel.

Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    channel
        Variable Type       ViInt16

        Determine which channel is to be configured:

        RI2583_CHANNEL1   defined as 1
        RI2583_CHANNEL2   defined as 2

    modSource
        Variable Type       ViInt16

        Select the demodulation source (if any) for the specified channel.

        RI2583_OFF         defined as 0
        RI2583_MODINPUT1   defined as 1
        RI2583_MODINPUT2   defined as 2

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)           - No error, the instrument has been re-
                                   configured.

        RI2583_ERROR_HANDLE      - does not correspond to an open session
        RI2583_ERROR_CHANNEL     - parameter 'channel' has an unrecognised
                                   value.
```

```
        RI2583_ERROR_MODULATION -  parameter 'modSource' has an unrecognised
                                   value.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_channelDemodulateQuery

    ViStatus ri2583_channelDemodulateQuery (ViSession instrumentHandle,
            ViInt16 channel, ViPInt16 modSource);

  Purpose
    Report whether Demodulation is selected for the specified channel.

  Parameter List
    instrumentHandle
       Variable Type       ViSession

       The Instrument Handle returned by ri2583_init() or
       ri2583_autoInitialize() to open this instrument driver session.

    channel
       Variable Type       ViInt16

       Determine which channel is to be reported:

       RI2583_CHANNEL1   defined as 1
       RI2583_CHANNEL2   defined as 2

    modSource
       Variable Type       ViInt16 (passed by reference)

       Return the demodulation source (if any) for the specified channel.

       RI2583_OFF         defined as 0
       RI2583_MODINPUT1   defined as 1
       RI2583_MODINPUT2   defined as 2

  Return Value
       The status code returned by the function call:

       VI_SUCCESS (0)         - No error, the instrument has been re-
                                configured.

       RI2583_ERROR_HANDLE    - does not correspond to an open session
       RI2583_ERROR_CHANNEL   - parameter 'channel' has an unrecognised value.
       RI2583_ERROR_POINTER   - parameter 'modSource' is referenced by a null
                                pointer.

       To translate errors into text string form use the function
       ri2583_error_message().
```

```
    ri2583_channelRange

    ViStatus ri2583_channelRange (ViSession instrumentHandle, ViInt16 channel,
            ViInt16 range);

  Purpose

    Configure the voltage measurement range of the specified channel.

  Parameter List
    instrumentHandle
       Variable Type       ViSession

       The Instrument Handle returned by ri2583_init() or
       ri2583_autoInitialize() to open this instrument driver session.

    channel
       Variable Type       ViInt16

       Determine which channel is to be configured:

       RI2583_CHANNEL1    defined as 1
```

```
        RI2583_CHANNEL2    defined as 2

    range
        Variable Type       ViInt16

        Select the measurement range for the specified channel.

        RI2583_AUTORANGE    defined as 0
        RI2583_30MVRANGE    defined as 1
        RI2583_300MVRANGE   defined as 2
        RI2583_3VRANGE      defined as 3
        RI2583_30VRANGE     defined as 4
        RI2583_300VRANGE    defined as 5

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)           - No error, the instrument has been re-
                                   configured.

        RI2583_ERROR_HANDLE      - does not correspond to an open session
        RI2583_ERROR_CHANNEL     - parameter 'channel' has an unrecognised
                                   value.
        RI2583_ERROR_VOLTS_RANGE - parameter 'range' has an unrecognised
                                   value.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_channelRangeQuery

    ViStatus ri2583_channelRangeQuery (ViSession instrumentHandle, ViInt16
            channel, ViPInt16 range);

Purpose
    Report the voltage measurement range of the specified channel.

Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    channel
        Variable Type       ViInt16

        Determine which channel is to be reported:

        RI2583_CHANNEL1   defined as 1
        RI2583_CHANNEL2   defined as 2

    range
        Variable Type       ViInt16 (passed by reference)

        Return the currently selected measurement range for the specified
        channel.

        RI2583_AUTORANGE    defined as 0
        RI2583_30MVRANGE    defined as 1
        RI2583_300MVRANGE   defined as 2
        RI2583_3VRANGE      defined as 3
        RI2583_30VRANGE     defined as 4
        RI2583_300VRANGE    defined as 5

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)         - No error, the instrument has been re-
                                 configured.

        RI2583_ERROR_HANDLE    - does not correspond to an open session
        RI2583_ERROR_CHANNEL   - parameter 'channel' has an unrecognised value.
```

```
        RI2583_ERROR_POINTER   - parameter 'range' is referenced by a null
                                 pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_close

    ViStatus ri2583_close (ViSession instrumentHandle);

  Purpose
    Close a session that was opened with ri2583_init() or
    ri2583_autoInitialize(), calling viClose(instrSession) and
    viClose(rmSession).

  Parameter List
    instrumentHandle
       Variable Type       ViSession

       The Instrument Handle returned by ri2583_init() or
       ri2583_autoInitialize() to open this instrument driver session.

  Return Value
       The status code returned by the function call:

       VI_SUCCESS (0)         - No error.

       RI2583_WARN_HANDLE     - does not correspond to an open session

       To translate errors into text string form use the function
       ri2583_error_message().
```

```
    ri2583_closedLoopEnable

    ViStatus ri2583_closedLoopEnable (ViSession instrumentHandle, ViInt16
            channel);

  Purpose
    Enable or disable closed-loop control of generator amplitude to maintain a
    target amplitude on the specified channel.

    When enabled, every measurement is preceded by an iterative adjustment of
    generator amplitude to meet the target established by the last call to
    closedLoopSetup()which configured the specified channel.

    Closed loop can only be selected on one channel at a time.

  Parameter List
    instrumentHandle
       Variable Type       ViSession

       The Instrument Handle returned by ri2583_init() or
       ri2583_autoInitialize() to open this instrument driver session.

    channel
       Variable Type       ViInt16

       Disable closed-loop control, or select the control channel:

       RI2583_OFF        defined as 0
       RI2583_CHANNEL1   defined as 1
       RI2583_CHANNEL2   defined as 2

  Return Value
       The status code returned by the function call:

       VI_SUCCESS (0)               - No error, the instrument has been re-
                                      configured.

       RI2583_ERROR_HANDLE          - does not correspond to an open session
       RI2583_ERROR_CHANNEL_AND_OFF - parameter 'channel' has an unrecognised
                                      value.
```

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_closedLoopQuery

ViStatus ri2583_closedLoopQuery (ViSession instrumentHandle, ViInt16 channel, ViPReal64 targetVrms, ViPReal64 tolerance, ViPReal64 safetyLevel, ViPInt16 enable);

Purpose

Report closed-loop control parameters of the specified channel.

Parameter List

instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

channel
Variable Type ViInt16

Determine which channel is to be reported:

RI2583_CHANNEL1 defined as 1
RI2583_CHANNEL2 defined as 2

targetVrms
Variable Type ViReal64 (passed by reference)

Return the target value for the amplitude on the selected channel in Vrms.

tolerance
Variable Type ViReal64 (passed by reference)

Return the fractional tolerance on the value of target.

safetyLevel
Variable Type ViReal64 (passed by reference)

Return the maximum allowable generator amplitude in Vrms.

enable
Variable Type ViInt16 (passed by reference)

Report whether closed loop control is enabled on the selected channel.

RI2583_NO defined as 0
RI2583_YES defined as 1

Return Value

The status code returned by the function call:

VI_SUCCESS (0) - No error, the instrument has been re-configured.

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_CHANNEL - parameter 'channel' has an unrecognised value.
RI2583_ERROR_POINTER - a return parameter is referenced by the NULL pointer

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_closedLoopSetup

ViStatus ri2583_closedLoopSetup (ViSession instrumentHandle, ViInt16 channel, ViReal64 targetVrms, ViReal64 tolerance, ViReal64 safetylimit);

Purpose

Configure closed-loop control of generator amplitude to maintain a target amplitude on the specified channel.

When enabled by a call to ri2583_closedLoopEnable(), every trigger of the analyzer is followed by iterative adjustment of generator amplitude to bring the amplitude on the specified channel to the required level.

If this cannot be achieved without exceeding a generator amplitude of 'safety', an error will be generated.

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

channel

Variable Type ViInt16

Determine which channel is to be configured:

RI2583_CHANNEL1 defined as 1
RI2583_CHANNEL2 defined as 2

targetVrms

Variable Type ViReal64

Set the target value for the amplitude on the selected channel in Vrms.

The value must be between 0.001 and 300.0

tolerance

Variable Type ViReal64

Set the fractional tolerance on the value of target, between 0.001 and 0.1 inclusive.

safetylimit

Variable Type ViReal64

Set the maximum allowable generator amplitude in Vrms.

Return Value

The status code returned by the function call:

VI_SUCCESS (0)	- No error, the instrument has been re-configured.
RI2583_ERROR_HANDLE	- does not correspond to an open session
RI2583_ERROR_CHANNEL	- parameter 'channel' has an unrecognised value.
RI2583_ERROR_LOOP_TARGET	- parameter 'target' is outside the permitted range
RI2583_ERROR_LOOP_TOL	- parameter 'tolerance' is outside the permitted range
RI2583_ERROR_LOOP_LIMIT	- parameter 'safetylimit' is outside the permitted range

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_error_message

```
ViStatus ri2583_error_message (ViSession instrumentHandle, ViStatus
        errorCode, ViChar _VI_FAR errorMessage[]);
```

Purpose

Return an explanatory text string for the Status Code previously returned by a function from this instrument driver.

```
Parameter List
   instrumentHandle
      Variable Type       ViSession

      The Instrument Handle returned by ri2583_init() or
      ri2583_autoInitialize() to open this instrument driver session.

   errorCode
      Variable Type       ViStatus

      A Status Code previously returned from an instrument driver function.

   errorMessage
      Variable Type       ViChar[]

      Return the explanatory text string.

      Note: The calling application must assign storage for at least 256
      elements.

Return Value
      The status code returned by the function call:

      VI_SUCCESS            - text string returned.

      RI2583_ERROR_POINTER  - return parameter is referenced by a null
                              pointer.
```

```
   ri2583_error_query

   ViStatus ri2583_error_query (ViSession instrumentHandle, ViPInt32
           errorCode, ViChar _VI_FAR errorMessage[]);

Purpose
   Read the next message in the unit's error queue.

   If no messages remain, the response is 0,"NO ERROR IN QUEUE"

Parameter List
   instrumentHandle
      Variable Type       ViSession

      The Instrument Handle returned by ri2583_init() or
      ri2583_autoInitialize() to open this instrument driver session.

   errorCode
      Variable Type       ViInt32 (passed by reference)

      Return the error code portion of the response to the ERR? query.

   errorMessage
      Variable Type       ViChar[]

      Return the string portion of the response to the ERR? query.

      Note: The calling application must assign storage for at least 256
      elements.

Return Value
      The status code returned by the function call:

      VI_SUCCESS (0)        - No error, an error code and message have been
                              retrieved.

      RI2583_ERROR_HANDLE   - does not correspond to an open session
      RI2583_ERROR_POINTER  - return parameter is referenced by a null
                              pointer.

      To translate errors into text string form use the function
      ri2583_error_message().
```

---

ri2583_frequencySweepQuery

```
ViStatus ri2583_frequencySweepQuery (ViSession instrumentHandle, ViPReal64
        frequency1, ViPReal64 frequency2, ViPInt16 steps, ViPInt16
        interval);
```

Purpose

Report the current setup of frequency sweep.

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

frequency1

Variable Type ViReal64 (passed by reference)

The lowest frequency of the sweep, in Hertz.

frequency2

Variable Type ViReal64 (passed by reference)

The highest frequency of the sweep, in Hertz.

steps

Variable Type ViInt16 (passed by reference)

The number of steps in the sweep. Value will be in range 3 to 4000.

interval

Variable Type ViInt16 (passed by reference)

The type of interval between frequency steps:

RI2583_LINEAR	defined as 0
RI2583_LOGARITHMIC	defined as 1

Return Value

The status code returned by the function call:

VI_SUCCESS (0)	- No error.
RI2583_ERROR_HANDLE	- does not correspond to an open session
RI2583_ERROR_POINTER	- a return parameter is referenced by a NULL pointer.
RI2583_ERROR_SWEEP_STEP	- parameter 'steps' outside valid range

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_frequencySweepSetup

```
ViStatus ri2583_frequencySweepSetup (ViSession instrumentHandle, ViReal64
        frequency1, ViReal64 frequency2, ViInt16 steps, ViInt16 interval);
```

Purpose

Configure a sweep of frequencies from 'hertz1' to 'hertz2' or vice versa.

The sweep may be performed in either direction and the results returned by function ri2583_sweepFrequencyQuery().

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

frequency1

Variable Type ViReal64

```
        The first frequency of the sweep in Hz, in the range 10 uHz to 100 kHz
        inclusive.

    frequency2
        Variable Type       ViReal64

        The last frequency of the sweep in Hz, in the range 10 uHz to 100 kHz
        inclusive.

    steps
        Variable Type       ViInt16

        The number of steps in the sweep, between 3 and 4000 inclusive.

    interval
        Variable Type       ViInt16

        RI2583_LINEAR       defined as 0
        RI2583_LOGARITHMIC  defined as 1

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)           - No error, the instrument has been re-
                                   configured.

        RI2583_ERROR_HANDLE      - does not correspond to an open session
        RI2583_ERROR_FREQUENCY   - parameter 'hertz1' or 'hertz2' is outside
                                   the permitted range
        RI2583_ERROR_SWEEP_STEPS - parameter 'steps' is outside the permitted
                                   range
        RI2583_ERROR_SWEEP_TYPE  - parameter 'interval' has an unrecognised
                                   value.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_generatorAmplitude

    ViStatus ri2583_generatorAmplitude (ViSession instrumentHandle, ViReal64
            volts);

Purpose
    Define the amplitude of the generator output waveform as an rms voltage.

Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    volts
        Variable Type       ViReal64

        Set the output amplitude in V rms. Value must not exceed 10.3 V rms.

        When 'volts' is converted into an equivalent peak voltage for the
        currently selected waveform,  the result should not exceed 14.5 V
        Peak.

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)          - No error, the instrument has been re-
                                  configured.

        RI2583_ERROR_HANDLE     - does not correspond to an open session
        RI2583_ERROR_VOLTRMS    - parameter 'volts' outside permitted range.
        RI2583_ERROR_VOLTPK     - Vpeak equivalent of 'volts' outside permitted
                                  range.

        To translate errors into text string form use the function
        ri2583_error_message().
```

---

ri2583_generatorAmplitudeQuery

```
ViStatus ri2583_generatorAmplitudeQuery (ViSession instrumentHandle,
        ViPReal64 volts);
```

Purpose
Report the currently programmed amplitude of the generator signal.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

volts
Variable Type ViReal64 (passed by reference)

Return the selected amplitude in V rms

Return Value
The status code returned by the function call:

VI_SUCCESS (0)	- No error (the call was successful).
RI2583_ERROR_HANDLE	- does not correspond to an open session
RI2583_ERROR_POINTER	- parameter 'volts' is referenced by a null pointer.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_generatorBias

```
ViStatus ri2583_generatorBias (ViSession instrumentHandle, ViReal64
        dcbias);
```

Purpose
Define the DC offset of the generator output waveform.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

dcbias
Variable Type ViReal64

Set the output DC offset in volts.

Value must be in range -10.0 to +10.0 inclusive.

Return Value
The status code returned by the function call:

VI_SUCCESS (0)	- No error, the instrument has been re-configured.
RI2583_ERROR_HANDLE	- does not correspond to an open session
RI2583_ERROR_BIAS	- parameter 'bias' is outside permitted range.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_generatorBiasQuery

```
ViStatus ri2583_generatorBiasQuery (ViSession instrumentHandle, ViPReal64
        volts);
```

Purpose
Report the currently selected DC offset of the generator signal.

```
Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    volts
        Variable Type       ViReal64 (passed by reference)

        Return the selected offset in volts.

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)          - No error (the call was successful).

        RI2583_ERROR_HANDLE     - does not correspond to an open session
        RI2583_ERROR_POINTER    - parameter 'volts' is referenced by a null
                                  pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_generatorCarrierAmplitude

    ViStatus ri2583_generatorCarrierAmplitude (ViSession instrumentHandle,
            ViReal64 volts);

Purpose
    Define the maximum amplitude of a modulated generator output waveform as
    an rms voltage.

Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    volts
        Variable Type       ViReal64

        Set the output amplitude in V rms. Value must not exceed 10.3 V rms.

        When 'volts' is converted into an equivalent peak voltage for the
        currently selected waveform, the result should not exceed 14.5 V Peak.

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)          - No error, the instrument has been re-
                                  configured.

        RI2583_ERROR_HANDLE     - does not correspond to an open session
        RI2583_ERROR_VOLTRMS    - parameter 'volts' outside permitted range.
        RI2583_ERROR_VOLTPK     - Vpeak equivalent of 'volts' outside permitted
                                  range.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_generatorFrequency

    ViStatus ri2583_generatorFrequency (ViSession instrumentHandle, ViReal64
            frequency);

Purpose
    Set the output frequency of the generator.
```

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

frequency
Variable Type ViReal64

The output frequency of the generator in Hertz.

Acceptable values are between 10 uHz and 100 kHz inclusive.

Return Value
The status code returned by the function call:

VI_SUCCESS (0) - No error (the call was successful).

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_FREQUENCY - parameter 'frequency' is outside the permitted range.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_generatorFrequencyQuery

```
ViStatus ri2583_generatorFrequencyQuery (ViSession instrumentHandle,
        ViPReal64 frequency);
```

Purpose
Report the currently selected frequency of the generator signal.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

frequency
Variable Type ViReal64 (passed by reference)

Return the selected frequency in Hertz.

Return Value
The status code returned by the function call:

VI_SUCCESS (0) - No error (the call was successful).

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_POINTER - parameter 'frequency' is referenced by a null pointer.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_generatorFunction

```
ViStatus ri2583_generatorFunction (ViSession instrumentHandle, ViInt16
        shape);
```

Purpose
Define the shape of the generator output waveform.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

```
    shape
       Variable Type       ViInt16

       Select the shape of the generated function.

       RI2583_SINEWAVE      defined as 0
       RI2583_SQUAREWAVE    defined as 1
       RI2583_TRIANGLEWAVE  defined as 2

  Return Value
       The status code returned by the function call:

       VI_SUCCESS (0)         - No error, the instrument has been re-
                                configured.

       RI2583_ERROR_HANDLE    - does not correspond to an open session
       RI2583_ERROR_FUNCTION  - parameter 'shape' has an unrecognised value.

       To translate errors into text string form use the function
       ri2583_error_message().
```

```
    ri2583_generatorFunctionQuery

    ViStatus ri2583_generatorFunctionQuery (ViSession instrumentHandle,
            ViPInt16 shape);

  Purpose
    Report the currently selected shape of the generator waveform.

  Parameter List
    instrumentHandle
       Variable Type       ViSession

       The Instrument Handle returned by ri2583_init() or
       ri2583_autoInitialize() to open this instrument driver session.

    shape
       Variable Type       ViInt16 (passed by reference)

       Return the shape of the generator signal.

       RI2583_SINEWAVE      defined as 0
       RI2583_SQUAREWAVE    defined as 1
       RI2583_TRIANGLEWAVE  defined as 2

  Return Value
       The status code returned by the function call:

       VI_SUCCESS (0)         - No error (the call was successful).

       RI2583_ERROR_HANDLE    - does not correspond to an open session
       RI2583_ERROR_POINTER   - parameter 'shape' is referenced by a null
                                pointer.

       To translate errors into text string form use the function
       ri2583_error_message().
```

```
    ri2583_generatorHold

    ViStatus ri2583_generatorHold (ViSession instrumentHandle, ViInt16 state);

  Purpose
    Select or de-select holding the generator output at its instantaneous
    state, or at a particular phase angle.

    Any programmed DC offset will be present at the output.

  Parameter List
    instrumentHandle
       Variable Type       ViSession

       The Instrument Handle returned by ri2583_init() or
       ri2583_autoInitialize() to open this instrument driver session.
```

```
    state
        Variable Type       ViInt16

        Determine the state of the generator output.

        RI2583_OFF        defined as 0
        RI2583_PHASE0     defined as 1
        RI2583_PHASE90    defined as 2
        RI2583_PHASE180   defined as 3
        RI2583_PHASE270   defined as 4
        RI2583_PHASENOW   defined as 5

  Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)         - No error (the call was successful).

        RI2583_ERROR_HANDLE    - does not correspond to an open session
        RI2583_ERROR_HOLD      - parameter 'state' has an unrecognised value.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_generatorHoldQuery

    ViStatus ri2583_generatorHoldQuery (ViSession instrumentHandle, ViPInt16
            state);

  Purpose
    Report the hold status of the generator output as configured by
    ri2583_generatorHold()

  Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    state
        Variable Type       ViInt16 (passed by reference)

        Return the state of the generator hold as one of the following.

        RI2583_OFF        defined as 0
        RI2583_PHASE0     defined as 1
        RI2583_PHASE90    defined as 2
        RI2583_PHASE180   defined as 3
        RI2583_PHASE270   defined as 4
        RI2583_PHASENOW   defined as 5

  Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)         - No error (the call was successful).

        RI2583_ERROR_HANDLE    - does not correspond to an open session
        RI2583_ERROR_POINTER   - parameter 'state' is referenced by a NULL
                                 pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_generatorModulate

    ViStatus ri2583_generatorModulate (ViSession instrumentHandle, ViInt16
            source);

  Purpose
    Select or de-select amplitude modulation of the generator output.
```

```
Parameter List
    instrumentHandle

        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    source
        Variable Type       ViInt16

        The available sources are:

        RI2583_OFF        defined as 0
        RI2583_MODINPUT1  defined as 1
        RI2583_MODINPUT2  defined as 2

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)          - No error (the call was successful).

        RI2583_ERROR_HANDLE     - does not correspond to an open session
        RI2583_ERROR_MODULATION - parameter 'modSource' has an unrecognised
                                  value.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_generatorModulationAmplitude

    ViStatus ri2583_generatorModulationAmplitude (ViSession instrumentHandle,
            ViReal64 volts);

Purpose
    Define the amplitude of modulation of a modulated generator output in
    Vrms.

Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    volts
        Variable Type       ViReal64

        Set the modulation amplitude in V rms. Value must not exceed 10.3Vrms.

        When 'volts' is converted into an equivalent peak voltage for the
        currently selected waveform, the result should not exceed 14.5 V Peak.

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)          - No error, the instrument has been re-
                                  configured.

        RI2583_ERROR_HANDLE     - does not correspond to an open session
        RI2583_ERROR_VOLTRMS    - parameter 'volts' outside permitted range.
        RI2583_ERROR_VOLTPK     - Vpeak equivalent of 'volts' outside permitted
                                  range.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_generatorOutput

    ViStatus ri2583_generatorOutput (ViSession instrumentHandle, ViInt16
            state);
```

```
Purpose
    Set the generator output to one of the following states:

    OFF zero AC amplitude
    ON  generating selected function at programmed amplitude and frequency

    Any programmed DC offset will be present at the output in both of these
    states.

Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    state
        Variable Type       ViInt16

        Determine the state of the generator output.

        RI2583_OFF   defined as 0
        RI2583_ON    defined as 1

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)         - No error (the call was successful).

        RI2583_ERROR_HANDLE    - does not correspond to an open session
        RI2583_ERROR_OUTPUT    - parameter 'state' has an unrecognised value.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_generatorOutputQuery

    ViStatus ri2583_generatorOutputQuery (ViSession instrumentHandle, ViPInt16
            state);

Purpose
    Report the state of the generator output as configured by
    ri2583_generatorOutput().

Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    state
        Variable Type       ViInt16 (passed by reference)

        Return the state of the generator output as one of the following.

        RI2583_OFF   defined as 0
        RI2583_ON    defined as 1

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)         - No error (the call was successful).

        RI2583_ERROR_HANDLE    - does not correspond to an open session
        RI2583_ERROR_POINTER   - parameter 'state' is referenced by a NULL
                                 pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

---

ri2583_generatorSoftStart

ViStatus ri2583_generatorSoftStart (ViSession instrumentHandle, ViInt16 state);

Purpose

Select or deselect 'Soft Start' of the generator.

When selected, calling ri2583_generatorOutput(fraHandle, RI2583_ON) will ramp up the generator output over several seconds, instead of turning it on abruptly.

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

state

Variable Type ViInt16

Determine the status of 'Soft Start'.

RI2583_OFF defined as 0
RI2583_ON defined as 1

Return Value

The status code returned by the function call:

VI_SUCCESS (0) - No error (the call was successful).

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_ON_OFF - parameter 'state' has an unrecognised value.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_generatorSoftStartQuery

ViStatus ri2583_generatorSoftStartQuery (ViSession instrumentHandle, ViPInt16 state);

Purpose

Report the status of 'Soft Start' of the generator.

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

state

Variable Type ViInt16 (passed by reference)

Determined by the status of 'Soft Start':

RI2583_OFF defined as 0
RI2583_ON defined as 1

Return Value

The status code returned by the function call:

VI_SUCCESS (0) - No error (the call was successful).

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_POINTER - parameter 'state' is referenced by a null pointer.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_generatorSoftStop

ViStatus ri2583_generatorSoftStop (ViSession instrumentHandle, ViInt16 state);

Purpose

Select or deselect 'Soft Stop' of the generator.

When selected, calling ri2583_generatorOutput(fraHandle, RI2583_OFF) will ramp down the generator output over several seconds instead of turning it off abruptly.

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

state

Variable Type ViInt16

Determine the status of 'Soft Stop'.

RI2583_OFF defined as 0
RI2583_ON defined as 1

Return Value

The status code returned by the function call:

VI_SUCCESS (0) - No error (the call was successful).

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_ON_OFF - parameter 'state' has an unrecognised value.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_generatorSoftStopQuery

ViStatus ri2583_generatorSoftStopQuery (ViSession instrumentHandle, ViPInt16 state);

Purpose

Report the status of 'Soft Stop' of the generator.

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

state

Variable Type ViInt16 (passed by reference)

Determined by the status of 'Soft Stop':

RI2583_OFF defined as 0
RI2583_ON defined as 1

Return Value

The status code returned by the function call:

VI_SUCCESS (0) - No error (the call was successful).

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_POINTER - parameter 'state' is referenced by a null pointer.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_generatorSweepFrequency

ViStatus ri2583_generatorSweepFrequency (ViSession instrumentHandle,
ViInt16 step);

Purpose
Compute the frequency corresponding to the step number (according to parameters sent by ri2583_frequencySweepSetup()), and program the generator to the corresponding frequency.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

step
Variable Type ViInt16

Must be less than the number of steps specified in the most recent call to ri2583_frequencySweepSetup(). Lowest frequency step is 0.

Return Value
The status code returned by the function call:

VI_SUCCESS (0) - No error, the measurements have been taken.

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_SWEEP_STEP - parameter 'step' outside valid range.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_init

ViStatus ri2583_init (ViRsrc resourceName, ViBoolean IDQuery, ViBoolean resetDevice, ViPSession instrumentHandle);

Purpose
Open a session to the Default Resource Manager resource and a session to the instrument specified by the resourceName parameter.

If enabled, confirm that the instrument is a Racal 2583.

If enabled, configure the instrument to a known state.

Send the CLEAR command to flush the instrument's input and output buffers

Return an Instrument Handle which is used to differentiate between sessions of this instrument driver.

Each time this function is invoked a Unique Session is opened.

It is not advisable to have more than one session open for the same instrument.

Parameter List
resourceName
Variable Type ViRsrc

Specify the interface and address of the device that is to be initialized (Instrument Descriptor). The exact syntax is shown in the note below.

Upper-case denotes literals, lower-case integer arguments Optional parameters are shown in square brackets [].

VXI/MXI Interface:
VXI[board]::vxi_logical_address[::INSTR]

GPIB-VXI Interface:
GPIB-VXI[board][::gpib_vxi_primary_address]::vxi_logical_address[::IN

The default value for optional parameters are:

Board	0
gpib_vxi_primary_address	1

IDQuery
Variable Type ViBoolean

Specify if an ID Query is sent to the instrument during the initialization procedure.

VI_FALSE (0) - Skip Query
VI_TRUE (1) - Do Query

resetDevice
Variable Type ViBoolean

Specify if the instrument is to be reset to its power-on settings during the initialization procedure. The instrument can always be reset later using the reset function.

VI_FALSE (0) - Don't Reset
VI_TRUE (1) - Reset Device

instrumentHandle
Variable Type ViSession (passed by reference)

Return an Instrument Handle that is used in all subsequent function calls to differentiate between sessions of this instrument driver.

Return Value
The status code returned by the function call:

VI_SUCCESS (0)	- No error (the call was successful).
RI2583_WARN_RESOURCE	- a session to this resource is already open
RI2583_ERROR_SESSIONS	- already 8 open sessions to 2583 resources
RI2583_ERROR_POINTER	- parameter 'fraHandle' is referenced by a null pointer.
RI2583_ERROR_BAD_BOOLEAN	- parameter IDQuery or Reset_Device are not set to VI_TRUE or VI_FALSE
VI_ERROR_FAIL_ID_QUERY	- valid 2583 response not obtained

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_location

```
ViStatus ri2583_location (ViSession instrumentHandle, ViPInt16
        interfaceType, ViPInt16 board, ViPInt16 slot, ViPInt16
        logicalAddress);
```

Purpose
Report the interface type, board, slot number, and logical address of the instrument that this session controls.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

interfaceType
Variable Type ViInt16 (passed by reference)

This is the returned interface type used to communicate with the ri2583 unit, either:

VI_INTF_GPIB_VXI
VI_INTF_VXI (for both embedded VXI or MXI-VXI)

board
Variable Type ViInt16 (passed by reference)

The returned interface board number.

slot
Variable Type ViInt16 (passed by reference)

The returned VXI slot number (or GPIB primary address in the case where interfaceType == VI_INTF_GPIB_VXI).

logicalAddress
Variable Type ViInt16 (passed by reference)

The returned VXI logical address (or GPIB secondary address in the case where interfaceType == VI_INTF_GPIB_VXI).

Return Value
The status code returned by the function call:

VI_SUCCESS (0) - No error (the call was successful).

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_POINTER - one or more parameters is referenced by a null pointer.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_measureCompleteQuery

```
ViStatus ri2583_measureCompleteQuery (ViSession instrumentHandle, ViPInt32
        ORRStatus);
```

Purpose
Check whether a measurement has completed. The measurement must have been started with the ri2583_analyzerTrigger command.

The function will only report measurement complete the first time it is called after a measurement has finished.

The function always returns immediately, allowing the calling program to perform other functions while the measurement is in progress.

The value returned shows measurement in progress, measurement complete, or an error.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

ORRStatus
Variable Type ViInt32 (passed by reference)

Return the current out of range status, if the function returns RI2583_WARNING_OVERRANGE

The ri2583_outofrange_message function may be used to decode the returned values.

Return Value
Measurement Not Completed Codes
RI2583_WARN_NOT_COMPLETE - End of measurement not received

Measurement Completed Codes
VI_SUCCESS (0) - No error, measurement completed

Error Codes
RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_POINTER - one or more output parameters is referenced by a null pointer.

```
        Additional Errors Codes during Closed Loop operation
        RI2583_ERROR_LOOP_SAFETY  - Output rose to safety limit
        RI2583_ERROR_LOOP_SETTLE  - Unable to converge to target level.
        RI2583_ERROR_LOOP_TOLOW   - Output fell too low
        RI2583_ERROR_LOOP_NOINPUT - No input measured on target channel
        RI2583_ERROR_NO_GEN       - Generator cannot be initially set to zero
                                    volts.

        To translate errors into text string form use the function
        ri2583_error_message().
```

---

ri2583_measureQuery

```
ViStatus ri2583_measureQuery (ViSession instrumentHandle, ViPReal64
        frequency, ViPReal64 amplitude1, ViPReal64 phase1, ViPReal64
        amplitude2, ViPReal64 phase2, ViPInt32 ORRStatus);
```

Purpose

Trigger the analyzer, wait for measurement process to complete, and read the analyzer in polar format. Cartesian format readings may be obtained by following the call with a call to ri2583_analyzerCartesianQuery().

If closed-loop control of generator amplitude is enabled, the measurement process consists of iterative adjustments to generator amplitude until the analyzer reading is within-tolerance of the target value.

Parameter List

instrumentHandle
Variable Type: ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

frequency
Variable Type: ViReal64 (passed by reference)

The frequency setting of the generator in Hz.

amplitude1
Variable Type: ViReal64 (passed by reference)

The amplitude in V rms measured on channel 1.

phase1
Variable Type: ViReal64 (passed by reference)

The phase in degrees measured on channel 1.

amplitude2
Variable Type: ViReal64 (passed by reference)

The amplitude in V rms measured on channel 2.

phase2
Variable Type: ViReal64 (passed by reference)

The phase in degrees measured on channel 2.

ORRStatus
Variable Type: ViInt32 (passed by reference)

A mask indicating any out of range conditions that may have occurred.

Return Value

The status code returned by the function call:

VI_SUCCESS (0)	- No error, values have been returned.
VI_WARN_OVERRANGE	- Values have been returned, but are suspect because one or more overrange flags are set.
RI2583_ERROR_HANDLE	- does not correspond to an open session

```
        RI2583_ERROR_LOOP_SAFETY -  closed-loop reading out of tolerance at
                                    generator safety limit
        RI2583_ERROR_LOOP_SETTLE -  closed-loop reading out of tolerance after
                                    10 iterations
        RI2583_ERROR_NO_READING  -  reading unavailable or incomplete.
        RI2583_ERROR_POINTER     -  one or more output parameters is referenced
                                    by a null pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_modulateQuery

    ViStatus ri2583_modulateQuery (ViSession instrumentHandle, ViPInt16
            modSource);

  Purpose
    Report the currently selected modulation source applied to the generator
    signal.

  Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    modSource
        Variable Type       ViInt16 (passed by reference)

        Return the currently selected modulation of the generator signal:

        RI2583_OFF         defined as 0
        RI2583_MODINPUT1   defined as 1
        RI2583_MODINPUT2   defined as 2

  Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)         - No error

        RI2583_ERROR_HANDLE    - does not correspond to an open session
        RI2583_ERROR_POINTER   - parameter 'modSource' is referenced by a null
                                 pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_modulationAmplitudeQuery

    ViStatus ri2583_modulationAmplitudeQuery (ViSession instrumentHandle,
            ViPReal64 volts);

  Purpose
    Report the currently programmed amplitude of the output modulation when
    the generator is in modulation mode.

  Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    volts
        Variable Type       ViReal64 (passed by reference)

        Return the modulation amplitude in V rms.

  Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)         - No error (the call was successful).
```

```
        RI2583_ERROR_HANDLE    - does not correspond to an open session
        RI2583_ERROR_POINTER   - parameter 'volts' is referenced by a null
                                 pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_nameEntry

    ViStatus ri2583_nameEntry (ViInt32 unit, ViChar _VI_FAR resourceName[]);

Purpose
    The function ri2583_autoInitialize() will create a table with a maximum of
    8 entries, containing resource names returned by VISA for the first 8
    units of 2583 located in the VXI address space.

    This function allows an application program to obtain the resource name
    corresponding to each unit of 2583 found by ri2583_autoInitialize. This
    resource name is needed if the user wishes to open a session to a specific
    unit using ri2583_init().

Parameter List
    unit
        Variable Type       ViInt32

        The sequence number in the table of the target unit, in range 1-8.

    resourceName
        Variable Type       ViChar[]

        The function returns the VISA resource name as a null-terminated
        string.

        The user must ensure that there is room for up to 256 bytes (including
        null terminator).

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)         - No error, a resource name has been returned.

        RI2583_ERROR_POINTER   - resourceName is a null pointer.
        RI2583_ERROR_NOT_FOUND - the value of 'unit' does not correspond to a
                                 list entry with a valid resource name.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_outofrange_message

    ViStatus ri2583_outofrange_message (ViPInt32 outofrangeCode, ViChar
            _VI_FAR outofrangeMessage[]);

Purpose
    Examine an out-of-range Code previously returned by the instrument driver.

    Each call will return an explanatory text string for a non-zero bit, and
    clear that bit in the out-of-range Code.

Parameter List
    outofrangeCode
        Variable Type       ViInt32 (passed by reference)

        An Out-of-Range Code previously returned from instrument driver
        function ri2583_queryORR().

        Each call to ri2583_outofrange_message() will clear one non-zero-bit
        of this parameter and return the corresponding text message in array
        'outofrangeMessage'

    outofrangeMessage
        Variable Type       ViChar[]

        Return the explanatory text as a null-terminated string.
```

Note: The calling application must assign storage for at least 256 elements.

Return Value
The status code returned by the function call:

VI_SUCCESS - text string returned.

RI2583_WARN_NO_MESSAGE - no out-of-range bit was set.
RI2583_ERROR_POINTER - return parameter is referenced by a null pointer.

---

ri2583_queryESR

ViStatus ri2583_queryESR (ViSession instrumentHandle, ViPInt32 value);

Purpose
Send the "*ESR?" query command to the instrument, and return the value of the Event Status Register.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

value
Variable Type ViInt32 (passed by reference)

Return the integer value of the reply. Individual non-zero bit values represent:

1	Operation complete (*OPC previously sent)
4	IEE488.2 Query Error
8	IEE488.2 Device-dependent Error
16	IEE488.2 Execution Error
32	IEE488.2 Command Error
128	IEE488.2 Power-on

Return Value
The status code returned by the function call:

VI_SUCCESS (0) - No error.

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_POINTER - parameter 'value' is referenced by a NULL pointer.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_queryORR

ViStatus ri2583_queryORR (ViSession instrumentHandle, ViInt32 mask, ViPInt32 value);

Purpose
Send the "ORR?" query command to the instrument and return the value of the Out of Range Register.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

mask
Variable Type ViInt32

A mask to be ANDed with the instrument reply to generate output parameter 'value'.

```
    value
        Variable Type       ViInt32 (passed by reference)

        Return the instrument reply value ANDED with 'mask'.

        Individual non-zero bit values represent:

              1  Channel 1 Common Mode Overrange
              2  Channel 2 Common Mode Overrange
              4  Carrier 1 Level Overrange
              8  Carrier 2 Level Overrange
             16  Synchronizer Common Mode Overrange
             32  Carrier 1 Common Mode Overrange
             64  Carrier 2 Common Mode Overrange
            128  Power Supply out of range Overrange
            256  Synchronizer Dynamic Overrange
          16384  Carrier 1 Level Underrange
          32768  Carrier 2 Level Underrange
          65536  Channel 1 Dynamic Overrange
         131072  Channel 2 Dynamic Overrange

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)          - No error.

        RI2583_ERROR_HANDLE     - does not correspond to an open session
        RI2583_ERROR_POINTER    - parameter 'value' is referenced by a NULL
                                  pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_querySTB

    ViStatus ri2583_querySTB (ViSession instrumentHandle, ViPInt32 value);

Purpose
    Send the "*STB?" query command to the instrument, and return the value of
    the Status Byte.

Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    value
        Variable Type       ViInt32 (passed by reference)

        Return the integer value of the reply. Individual non-zero bits
        signify:

          4  Error Queue has data
          8  Overrange Register summary bit
         16  Message available
         32  Event status summary bit
         64  Master summary bit

Return Value
        The status codes returned by the function call:

        VI_SUCCESS (0)          - No error.

        RI2583_ERROR_HANDLE     - does not correspond to an open session
        RI2583_ERROR_POINTER    - parameter 'value' is referenced by a NULL
                                  pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

---

ri2583_query_read

```
ViStatus ri2583_query_read (ViSession instrumentHandle, ViInt32 qlength,
        ViChar _VI_FAR query[], ViPInt32 rlength, ViChar _VI_FAR reply[],
        ViInt32 replylength);
```

Purpose

Send a query command to the instrument and return the reply string.

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

qlength

Variable Type ViInt32

The length of the query command (excluding terminating null).

query

Variable Type ViChar[]

The text of the query command as a null terminated string.

rlength

Variable Type ViInt32 (passed by reference)

Return the number of bytes (excluding terminating null) in reply[].

reply

Variable Type ViChar[]

The bytes returned by the instrument, stripped of any terminating newline characters, and null-terminated.

replylength

Variable Type ViInt32

The maximum length that can be stored in reply (including terminating NULL)

Return Value

The status code returned by the function call:

VI_SUCCESS (0) - No error.

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_POINTER - one or more output parameters are referenced by a NULL pointer.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_read

```
ViStatus ri2583_read (ViSession instrumentHandle, ViInt32 bufferSize,
        ViChar _VI_FAR reply[], ViPInt32 returnCount);
```

Purpose

Read an output string from the instrument (which must have previously been sent a query command).

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

bufferSize
Variable Type ViInt32

The maximum length of reply string (including terminating null) that can be accommodated in array response[].

reply
Variable Type ViChar[]

The bytes returned by the instrument, stripped of any terminating newline characters, and null-terminated.

returnCount
Variable Type ViInt32 (passed by reference)

The number of bytes (excluding terminating null) in 'reply'.

Return Value
The status codes returned by the function call:

VI_SUCCESS (0) - No error.

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_POINTER - one or more output parameters are referenced by a NULL pointer.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_reset

```
ViStatus ri2583_reset (ViSession instrumentHandle);
```

Purpose
Reset the instrument to a known state by sending the *RST common command defined by IEEE 488.2.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

Return Value
The status code returned by the function call:

VI_SUCCESS (0) - No error (the call was successful).

RI2583_ERROR_HANDLE - does not correspond to an open session

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_revision_query

```
ViStatus ri2583_revision_query (ViSession instrumentHandle, ViChar _VI_FAR
        instrumentDriverRevision[], ViChar _VI_FAR firmwareRevision[]);
```

Purpose
Return the revision numbers of the instrument driver and instrument firmware.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

instrumentDriverRevision
Variable Type ViChar[]

Return the Instrument Driver Software Revision in format AA.BB, where AA is a two-ASCII-numeric issue number, BB a two-ASCII-numeric subsidiary revision number. This is followed by the null character (ASCII code = 0).

Note: The calling application must have assigned sufficient storage to hold these characters.

firmwareRevision
Variable Type ViChar[]

Return the Instrument Firmware Revision.

Note: The target string should be capable of receiving at least 40 characters.

Return Value
The status code returned by the function call:

VI_SUCCESS (0) - No error (the call was successful).

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_POINTER - one or more output parameters are referenced by a NULL pointer.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_selectConnector

```
ViStatus ri2583_selectConnector (ViSession instrumentHandle, ViInt16
        connector);
```

Purpose
Switch the front-panel signal inputs and outputs between the individual BNC connectors and the multipole D-connector.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

connector
Variable Type ViInt16

Determine which front-panel terminals are connected to the instrument circuits.

RI2583_BNC defined as 0
RI2583_MULTI defined as 1

Return Value
The status code returned by the function call:

VI_SUCCESS (0) - No error (the call was successful).

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_CONNECTOR - parameter 'connector' has an unrecognised value.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_selectConnectorQuery

```
ViStatus ri2583_selectConnectorQuery (ViSession instrumentHandle, ViPInt16
        select);
```

Purpose
Report the status of the front-panel connector selection (BNC connectors or multi-way D-connector).

```
Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    select
        Variable Type       ViInt16 (passed by reference)

        Determined by the current front-panel connector selection:

        RI2583_BNC    defined as 0
        RI2583_MULTI  defined as 1

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)         - No error (the call was successful).

        RI2583_ERROR_HANDLE    - does not correspond to an open session
        RI2583_ERROR_POINTER   - parameter 'select' is referenced by a null
                                 pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_self_test

    ViStatus ri2583_self_test (ViSession instrumentHandle, ViPInt16
            selfTestResult, ViChar _VI_FAR selfTestMessage[]);

Purpose
    Invoke the unit's self-test and return the 16-bit result code.

    To invoke the self-test the *TST? common query defined by IEEE 488.2 is
    sent to the unit.

    No ASCII error message is returned thus the returned message will always
    be a NULL string.

Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    selfTestResult
        Variable Type       ViInt16 (passed by reference)

        This control contains the value returned from the unit's self test.

        Zero means success; non-zero results indicate a failure.

        The result can be interpreted with the aid of the unit's operator's
        manual.

    selfTestMessage
        Variable Type       ViChar[]

        The string returned is always NULL.

Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)         - No error (the call was successful).

        RI2583_ERROR_HANDLE    - does not correspond to an open session
        RI2583_ERROR_POINTER   - one or more output parameters are referenced
                                 by a NULL pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

---

ri2583_sendCLS

```
ViStatus ri2583_sendCLS (ViSession instrumentHandle);
```

Purpose
Send the "*CLS" command to clear the Event Status Register, Over-range Register, and Error Queue.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

Return Value
The status code returned by the function call:

VI_SUCCESS (0) - No error

RI2583_ERROR_HANDLE - does not correspond to an open session

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_sendOPC

```
ViStatus ri2583_sendOPC (ViSession instrumentHandle);
```

Purpose
Send the "*OPC" command to enable setting of the Operation Complete bit in the Event Status Register when all programmed tasks have been completed by the instrument.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

Return Value
The status code returned by the function call:

VI_SUCCESS (0) - No error

RI2583_ERROR_HANDLE - does not correspond to an open session

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_serial_query

```
ViStatus ri2583_serial_query (ViSession instrumentHandle, ViChar _VI_FAR
        serial[]);
```

Purpose
Send the "*IDN?" query command to the instrument, and extract the instrument serial number as a string.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

serial
Variable Type ViChar[]

Return the serial number as a null-terminated string. A minimum of 40 characters should be available.

Return Value

The status code returned by the function call:

VI_SUCCESS (0) - No error.

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_POINTER - parameter 'serial' is referenced by a NULL pointer.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_sweepFrequencyQuery

ViStatus ri2583_sweepFrequencyQuery (ViSession instrumentHandle, ViInt16 mode, ViReal64 _VI_FAR frequency[], ViReal64 _VI_FAR amplitude1[], ViReal64 _VI_FAR phase1[], ViReal64 _VI_FAR amplitude2[], ViReal64 _VI_FAR phase2[], ViPInt32 ORRStatus);

Purpose

Perform a frequency sweep according to the parameters passed by ri2583_frequencySweepSetup().

Results are returned via the array pointers passed in the call. It is the calling program's responsibility to ensure that sufficient storage has been allocated for the arrays.

Sweeps may be performed in ascending or descending frequency order, according to parameter 'mode', but results are always stored with the lowest frequency corresponding to the first element [0] of each array.

Parameter List

instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

mode
Variable Type ViInt16

Select direction of sweep (increasing or decreasing frequency respectively:

RI2583_SWEEPUP defined as 0
RI2583_SWEEPDOWN defined as 1

frequency
Variable Type ViReal64[]

The frequency settings of the generator.

amplitude1
Variable Type ViReal64[]

The amplitudes in V rms measured on channel 1.

phase1
Variable Type ViReal64[]

The phases in degrees measured on channel 1.

amplitude2
Variable Type ViReal64[]

The amplitudes in V rms measured on channel 2.

phase2
Variable Type ViReal64[]

The phases in degrees measured on channel 2.

ORRStatus
Variable Type ViInt32 (passed by reference)

A mask indicating any out of range conditions that may have occurred.

Return Value
The status code returned by the function call:

VI_SUCCESS (0) - No error, values have been returned.

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_POINTER - one or more output parameters is referenced by a null pointer.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_sweepHarmonicQuery

```
ViStatus ri2583_sweepHarmonicQuery (ViSession instrumentHandle, ViInt16
        harmonic1, ViInt16 harmonic2, ViReal64 _VI_FAR amplitude1[],
        ViReal64 _VI_FAR phase1[], ViReal64 _VI_FAR amplitude2[], ViReal64
        _VI_FAR phase2[], ViPInt32 ORRStatus);
```

Purpose
Measure at all harmonics of the generator fundamental between 'harmonic1' and 'harmonic2' inclusive, and return measurement results via the array pointers in the call.

Parameter List
instrumentHandle
Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

harmonic1
Variable Type ViInt16

The first harmonic of the sweep, in range 1 to 16 inclusive.

harmonic2
Variable Type ViInt16

The last harmonic of the sweep, in range 1 to 16 inclusive

amplitude1
Variable Type ViReal64[]

The rms amplitudes measured on channel 1.

phase1
Variable Type ViReal64[]

The phases in degrees measured on channel 1.

amplitude2
Variable Type ViReal64[]

The rms amplitudes measured on channel 2.

phase2
Variable Type ViReal64[]

The phases in degrees measured on channel 2.

ORRStatus
Variable Type ViInt32 (passed by reference)

A mask indicating any out of range conditions that may have occurred.

Return Value
The status code returned by the function call:

VI_SUCCESS (0) - No error, the instrument has been re-configured.

RI2583_ERROR_HANDLE - does not correspond to an open session

```
        RI2583_ERROR_POINTER   - one or more output parameters is referenced by
                                 a null pointer.
        RI2583_ERROR_HARMONIC  - parameter 'harmonic1' or 'harmonic2' is
                                 outside the permitted range

        To translate errors into text string form use the function
        ri2583_error_message().
```

---

ri2583_syncConfigure

```
ViStatus ri2583_syncConfigure (ViSession instrumentHandle, ViInt16 slope,
        ViReal64 level, ViInt16 couple, ViReal64 ratio);
```

Purpose

Configure the external synchronization triggering conditions, ratio between output frequency and sync trigger frequency.

Note that control of analyzer triggering (generator frequency or sync input) is selected by ri2583_syncEnable.

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

slope

Variable Type ViInt16

Select the slope for level triggering.

```
RI2583_RISING      defined as 1
RI2583_FALLING     defined as 2
```

level

Variable Type ViReal64

Set the signal level in V DC on the selected slope at which synchronization is triggered.

Acceptable values are between + and - full range of the selected range.

couple

Variable Type ViInt16

Select the input coupling of the sync input.

```
RI2583_COUPLEDC    defined as 0
RI2583_COUPLEAC    defined as 1
```

ratio

Variable Type ViReal64

Program the ratio of analyzer fundamental measurement frequency to synchronizer input frequency. A ratio greater than 1 specifies that the measurement frequency is higher than the synchronizer input frequency by the specified factor.

Return Value

The status code returned by the function call:

```
VI_SUCCESS (0)           - No error, the instrument has been re-
                           configured.

RI2583_ERROR_HANDLE      - does not correspond to an open session
RI2583_ERROR_SYNC_EDGE   - parameter 'slope' is outside the valid
                           range.
RI2583_ERROR_SYNC_LEVEL  - parameter 'level' is outside the valid
                           range.
RI2583_ERROR_COUPLING    - parameter 'level' is outside the valid
                           range.
```

```
        RI2583_ERROR_SYNC_RATIO -  parameter 'ratio' is outside the valid
                                   range.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_syncConfigureQuery

    ViStatus ri2583_syncConfigureQuery (ViSession instrumentHandle, ViPInt16
            slope, ViPReal64 level, ViPInt16 couple, ViPReal64 ratio);

  Purpose
    Report the external synchronization parameters currently programmed.

  Parameter List
    instrumentHandle
        Variable Type       ViSession

        The Instrument Handle returned by ri2583_init() or
        ri2583_autoInitialize() to open this instrument driver session.

    slope
        Variable Type       ViInt16 (passed by reference)

        Return selected edge for level triggering.

        RI2583_RISING      defined as 1
        RI2583_FALLING     defined as 2

    level
        Variable Type       ViReal64 (passed by reference)

        Return the signal level in Vdc on the selected slope at which
        synchronization is triggered.

    couple
        Variable Type       ViInt16 (passed by reference)

        Return the currently selected input coupling for the sync input.

        RI2583_COUPLEDC    defined as 0
        RI2583_COUPLEAC    defined as 1

    ratio
        Variable Type       ViReal64 (passed by reference)

        Return the programmed ratio of analyzer trigger frequency to sync
        input frequency.

  Return Value
        The status code returned by the function call:

        VI_SUCCESS (0)          - No error, the instrument has been re-
                                  configured.

        RI2583_ERROR_HANDLE     - does not correspond to an open session
        RI2583_ERROR_POINTER    - parameter 'state', 'trigger' or 'level' is
                                  referenced by a null pointer.

        To translate errors into text string form use the function
        ri2583_error_message().
```

```
    ri2583_syncEnable

    ViStatus ri2583_syncEnable (ViSession instrumentHandle, int state);

  Purpose
    Select or de-select whether the instrument is synchronised to an external
    signal.

    If selected:
      Generator output is disabled.
```

Analyzer fundamental or harmonic frequencies are determined with respect to the sync input.

Phase is determined with respect to the defined trigger-point.

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

state

Variable Type int

Select or deselect external synchronization

RI2583_SYNC_OFF defined as 0
RI2583_SYNC_LOOSE defined as 1
RI2583_SYNC_TIGHT defined as 2

Return Value

The status code returned by the function call:

VI_SUCCESS (0) - No error, the instrument has been re-configured.

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_SYNC_ENABLE - parameter 'state' has an unrecognised value.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_syncEnableQuery

```
ViStatus ri2583_syncEnableQuery (ViSession instrumentHandle, ViPInt16
        state);
```

Purpose

Determine whether the output frequency is controlled by the programmed generator frequency or the external synchronization input.

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

state

Variable Type ViInt16 (passed by reference)

Return status of external sync control of generator frequency.

RI2583_OFF defined as 0
RI2583_ON defined as 1

Return Value

The status code returned by the function call:

VI_SUCCESS (0) - No error, the instrument has been re-configured.

RI2583_ERROR_HANDLE - does not correspond to an open session
RI2583_ERROR_POINTER - parameter 'state' is referenced by a null pointer.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_syncLockedQuery

```
ViStatus ri2583_syncLockedQuery (ViSession instrumentHandle, ViPBoolean
        state);
```

Purpose

Indicate whether the instrument has sucessfully locked to the Synchronizer input signal.

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

state

Variable Type ViBoolean (passed by reference)

Return status of external sync lock, TRUE indicates the instrument is phase and frequency locked to Synchronizer input

FALSE defined as 0
TRUE defined as 1

Return Value

The status code returned by the function call:

VI_SUCCESS (0)	- No error, the instrument has been re-configured.
RI2583_ERROR_HANDLE	- does not correspond to an open session
RI2583_ERROR_POINTER	- parameter 'state' is referenced by a null pointer.

To translate errors into text string form use the function ri2583_error_message().

---

ri2583_write

```
ViStatus ri2583_write (ViSession instrumentHandle, ViInt32 count, ViChar
        _VI_FAR message[]);
```

Purpose

Send an ASCII message to the unit. The message may contain any device-specific or IEEE 488.2 common command.

Parameter List

instrumentHandle

Variable Type ViSession

The Instrument Handle returned by ri2583_init() or ri2583_autoInitialize() to open this instrument driver session.

count

Variable Type ViInt32

The length of the message (excluding terminating null).

message

Variable Type ViChar[]

The text of the message as a null-terminated string.

Return Value

The status code returned by the function call:

VI_SUCCESS (0)	- No error.
RI2583_ERROR_HANDLE	- does not correspond to an open session
RI2583_ERROR_POINTER	- one or more output parameters are referenced by a NULL pointer.

```
To translate errors into text string form use the function
ri2583_error_message().
```