RACAL INSTRUMENTS
3156B
200 MS/s DUAL CHANNEL
WAVEFORM SYNTHESIZER

PUBLICATION NO. 980897

EADS North America Defense Test and Services, Inc.
4 Goodyear, Irvine, CA 92618
Tel: (800) 722-2528, (949) 859-8999; Fax: (949) 859-7139

info@eads-nadefense.com
sales@eads-nadefense.com
helpdesk@eads-nadefense.com
http://www.eads-nadefense.com

PUBLICATION DATE: August 24, 2005

THANK YOU FOR PURCHASING THIS EADS NORTH AMERICA DEFENSE TEST AND SERVICES PRODUCT

For this product, or any other EADS North America Defense Test and Services, Inc. product that incorporates software drivers, you may access our web site to verify and/or download the latest driver versions. The web address for driver downloads is:

http://www.eads-nadefense.com/downloads

If you have any questions about software driver downloads or our privacy policy, please contact us at:

info@eads-nadefense.com

WARRANTY STATEMENT

All EADS North America Defense Test and Services, Inc. products are designed and manufactured to exacting standards and in full conformance to EADS ISO 9001:2000 processes.

For the specific terms of your standard warranty, or optional extended warranty or service agreement, contact your EADS North America Defense Test and Services, Inc. customer service advisor. Please have the following information available to facilitate service.

1. Product serial number
2. Product model number
3. Your company and contact information

You may contact your customer service advisor by:

E-Mail:	Helpdesk@eads-nadefense.com	
Telephone:	+1 800 722 3262	(USA)
Fax:	+1 949 859 7309	(USA)

RETURN of PRODUCT

Authorization is required from EADS North America Defense Test and Services, Inc. before you send us your product for service or calibration. Call or contact the Customer Support Department at 1-800-722-3262 or 1-949-859-8999 or via fax at 1-949-859-7139. We can be reached at: helpdesk@eads-nadefense.com.

FOR YOUR SAFETY

Before undertaking any troubleshooting, maintenance or exploratory procedure, read carefully the **WARNINGS** and **CAUTION** notices.

CAUTION
RISK OF ELECTRICAL SHOCK
DO NOT OPEN

This equipment contains voltage hazardous to human life and safety, and is capable of inflicting personal injury.

If this 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.

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/two-prong adapter. This will defeat the protective feature of the third conductor in the power cord.

CAUTION
SENSITIVE ELECTRONIC DEVICES
DO NOT SHIP OR STORE NEAR STRONG ELECTROSTATIC, ELECTROMAGNETIC, MAGNETIC OR RADIOACTIVE FIELDS

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.

Before operating this instrument:

1. Ensure the proper fuse is in place for the power source to operate.

2. Ensure all other devices connected to or in proximity to this instrument are properly grounded or connected to the protective third-wire earth ground.

If the instrument:

- fails to operate satisfactorily
- shows visible damage
- has been stored under unfavorable conditions
- has sustained stress

Do not operate until, performance is checked by qualified personnel.

Racal Instruments

EC Declaration of Conformity

We

Racal Instruments Inc.
4 Goodyear Street
Irvine, CA 92618

declare under sole responsibility that the

3156B Arbitrary Waveform Generator, P/N 407862-001/-011

They conform to the following Product Specifications:

EMC: EN61000-6-3:2001 and EN61000-6-1:2001

Supplementary Information:

The above specifications are met when the product is installed in a Racal Instruments certified mainframe with faceplates installed over all unused slots, as applicable

The product herewith complies with the requirements of the Low Voltage Directive 73/23/EEC and the EMC Directive 89/336/EEC (modified by 93/68/EEC).

Irvine, CA, May 26, 2005 [illegible]
Engineering Director

Table of Contents

List of Figures

List of Tables

Chapter 1
GETTING STARTED

What's in This Chapter

This chapter contains a general description of the VXIbus 3156B Waveform Generators and an overall functional description of the instrument. It also describes the front panel connectors and indicators.

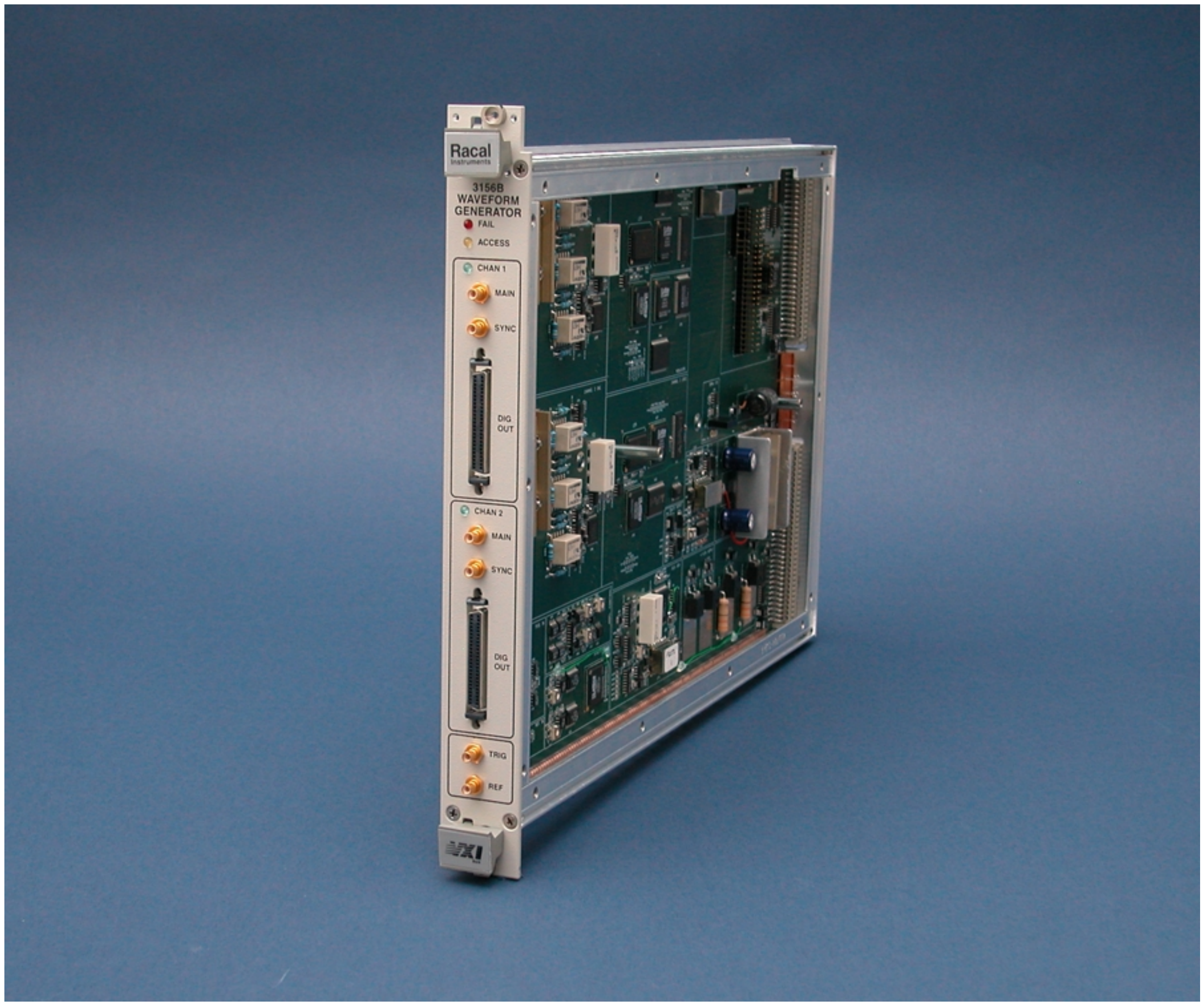


Figure 1-1, The 3156B

Introduction

A detailed functional description is given following the general description of the features, functions, and options available with the 3156B.

The 3156B is a VXIbus, single slot, C-size, dual-channel synthesized Waveform Generator, a high performance instrument that provides six powerful functions in one small package. The 3156B generates an array of standard waveforms from a built-in waveform library as well as arbitrary, sequenced and modulated waveforms. The 3156B can also be used as a dual-channel 12-bit parallel pattern generator up to 100Mbits/s and as video stroke generator. The generator outputs 16-bit waveforms from two channels at up to 200MS/s with different waveform properties. The dynamic range is greatly improved over 12-bit designs providing increased dynamic range and lower “noise floor” making it ideal for the generation of multi-tone signals and I&Q modulation.

Direct Digital Synthesis (DDS) technology, utilized in the design of the 3156B, allows flexibility in usage of features like FM, FSK, sweep and frequency hopping. For example, the FM feature can be stimulated by an internal source, or arbitrary FM waveform allowing the production of customized chirp signals. Included WaveCAD software can be used to breadboard custom frequency modulation profiles graphically.

Sample rates up to 200MS/s are available with memory size up to 1Meg. Channels A and B are both synchronized to the same sampling clock however, each channel can output a different waveform shape and length.

Based entirely on digital design, the 3156B has no analog functions resident in its hardware circuits. Data has to be downloaded to the instrument for it to start generating waveforms. The instrument can compute and generate a number of standard functions such like sine, square triangle and others however, complex waveforms must be converted to an appropriate format and downloaded to the 3156B as waveform coordinates. Dedicated waveform memory stores waveforms in memory segments and allows playback of a selected waveform when required. The waveforms are kept in the memory as long as the power is on.

Frequency accuracy of the output waveform is determined by the clock reference. Using CLK10 as the reference oscillator provides 100ppm accuracy and stability over time and temperature. If higher accuracy and/or stability are required, one may select between two options: Purchase the optional TCXO reference or, connect his own reference oscillator to a front panel input and use this input as the reference for the 3156B. Frequency may be is programmed with up to 10 digits, so using an external reference is recommended, if you intend to utilize the full resolution provided by the instrument.

Output amplitude for each of the channels may be programmed separately from 200mV to 20V peak-to-peak into an open circuit, and 100mV to 10V into 50Ω. Amplitude and offsets are programmed with 4 digits of resolution.

Besides its normal continuous mode, the 3156B responds to a variety of trigger sources. The output waveform may be gated, triggered, or may generate a counted burst of waveforms. A built-in re-trigger generator with a programmable period can be used as a replacement of an external trigger source. The 3156B generates arbitrary waveforms with 12 bits of vertical resolution. Any waveform it generates must first be downloaded to waveform memory.

The arbitrary waveform memory is a bank of 16-bit words. Each word represents a point on the horizontal waveform scale. Each word has a horizontal address that can range from 0 to 1Meg and a vertical address that can range from -32767 to +32768 (16 bits). Using a high speed clocking circuit, the digital contents of the arbitrary waveform memory are extracted and routed to the Digital to Analog Converter (DAC). The DAC converts the digital data to an analog signal, and the output amplifier completes the task by amplifying or attenuating the signal at the output connector. 16-bit, or 12-bit waveforms are available, depending on the requirement and instrument setting.

There is no need to use the complete memory bank every time an arbitrary waveform is generated. Waveform memory can be divided into up to 16k smaller segments and different waveforms can be loaded into each segment. The various segments may then be loaded into a sequence table to generate long and complex waveforms. The sequence table can link and loop up to 4096 segments in user defined order. Each channel has its own sequence generator.

The 3156B is a register-based product and therefore, software drivers or user generated programs must be used to access and program its registers. The product is supplied with Plug & Play driver and Soft Front Panel (SFP) example. The *SFP* driver simulate an array of mechanical front panels with the necessary push buttons, displays and dials to operate the 3156B as if it is a bench-top instrument.

Also available is WaveCAD 3.0 – Waveform Creation and Editing utility, which is very similar to the SFP however, provides complete access to all functions and features of the 3156B. It also allows on-screen creation and editing of complex waveforms and patterns to drive the 3156B various outputs.

It is highly recommended that the user become familiar with the 3156B front panel, its basic features, functions and programming concepts as described in this and the following chapters.

Options

As standard, the 3156B connects to the backplane CLK10 that provides a 10MHz signal for all devices in the chassis that need to synchronize to the same reference clock. The accuracy and stability of CLK10 is normally 100ppm however, it can be improved by connecting a better source to the controller. In cases where external reference is not available, but better accuracy and stability of the output frequency is required, a TCXO time base (1ppm) is offered as an option for the 3156B. The TCXO option is not field installable and therefore, it must be ordered with the product. Compare the option number below with the number printed on your instrument to check if the TCXO option is installed in your equipment.

407xxx-001 – 3156B, 200MS/s Dual-Channel Waveform Generator

407xxx-002 – 3156B, 200MS/s Dual-Channel Waveform Generator, TCXO reference

Manual Changes

Technical corrections to this manual (if any) are listed in the back of this manual on an enclosed MANUAL CHANGES sheet.

Safety Considerations

The 3156B has been manufactured according to international safety standards. The instrument meets EN61010-1 and UL1244 standards for safety of commercial electronic measuring and test equipment for instruments with an exposed metal chassis that is directly connected to earth via the chassis power supply cable.

WARNING:

Do not remove instrument covers when operating or when the chassis power cord is connected to the mains.

Any adjustment, maintenance and repair of an opened, powered-on instrument should be avoided as much as possible, but when necessary, should be carried out only by a skilled person who is aware of the hazard involved.

Supplied Accessories

The instrument is supplied with an Instruction Manual. The manual includes disks with VXI*plug&play* drivers along with WaveCAD for Win2000/XP. A Service Manual is available upon request.

Specifications

Instrument specifications are listed in Appendix A. These specifications are the performance standards or limits against which the instrument is tested. Specifications apply under the following conditions: output terminated into 50Ω after 30 minutes of warm up time, and within a temperature range of 0^{o}C to 37.7^{o}C.

Functional Description

A detailed functional description is given in the following paragraphs. The description is divided into logical groups: input and output connectors, operating modes, output type, output state, synchronization, and front panel indicators.

Input and Output Connectors

The 3156B has 6 SMB connectors on its front panel: two main outputs, two SYNC outputs, two digital outputs, trigger and 10MHz reference clock input. The connectors are described in the following paragraphs.

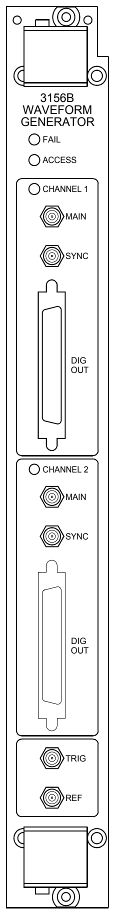


Figure 1-2, 3156B Front Panel

Main Output - Channel 1 and 2

The main output connectors output fixed (standard) waveforms to 25MHz, user (arbitrary) and sequenced waveforms with sampling clock to 200MS/s. Output source impedance is 50Ω, hence the cable connected to this output should be terminated with 50Ω load resistance. If the output is connected to a different load resistance, determine the actual amplitude from the following equation:

$$V_{out} = 2V_{prog}\left(\frac{50\Omega}{50\Omega + R_L}\right)$$

The output amplitude is doubled when the output impedance is above roughly 10kΩ.

SYNC Output - Channel 1 and 2

The SYNC outputs generate a single or multiple TTL pulses for synchronizing other instruments (i.e., an oscilloscope) to the output waveform. The SYNC signal always appears at a fixed point relative to the waveform. The location of the signal along the waveform is programmable. The SYNC outputs are used as marker outputs when the 3156B is programmed to one of the modulation functions.

Trigger Input

In general, the trigger input is used for stimulating an output waveform at the main output connector. Trigger level and edge sensitivity are programmable for the rigger input. For example, if your trigger signal rides on a dc level, you can offset the trigger level to the same level as your trigger signal, thus assuring correct threshold for the trigger signal.

The trigger input has different functionality, depending on the run mode of the instrument. For example, in continuous mode, the 3156B will start generating waveforms only when an enable command is true. The enable command can be selected from software or hardware. When hardware option is selected, the instrument will output waveform only after a valid trigger signal is applied to the trigger input. The trigger input is also used for enabling waveforms when the 3156B is placed in trigger, burst, or mixed run modes.

In trigger and burst modes, the trigger input is edge sensitive, i.e., it senses transitions from high to low or from low to high to trigger the 3156B. The direction of the transition is programmable. In gated mode, two trigger transitions are required to gate on and off the output waveform.

Reference Clock Input

The reference clock input accepts a 10MHz, TTL level signal to replace the CLK10 reference. The reference clock input is active only when selected.

Run Modes

The 3156B can be programmed to operate in one of four operating modes: continuous, triggered, gated and counted burst. These modes are described below. In some cases, the two channels can be used with different run modes however, to avoid setting conflicts; it is recommended that both channels share the same run mode. A list of setting conflicts resulting from incorrect setting of the various run modes and output functions is given in Chapter 3 of this manual.

Continuous

In normal continuous mode, the selected waveform is generated continuously at the selected frequency, amplitude and offset. Two conditions are required for the 3156B to output waveforms in continuous run mode: 1) Output is on and 2) Output is enabled by either a software or hardware enable command. The waveform at the output connector can be stopped (only on the last point of the waveform) with a software enable off command. Chapter 3 lists the various options of enabling and disabling the output waveform in the various run modes and enable sources.

Triggered

In triggered mode, the 3156B circuits are armed to generate one output waveform. The trigger circuit is sensitive to transitions at the trigger input. Select between positive or negative transitions to trigger the instrument. You may also program the trigger level to the desired threshold level. When triggered, the generator outputs one waveform cycle and remains idle at the last point of the waveform. Two conditions are required for the 3156B to output waveforms in triggered run mode: 1) Output is on and 2) Output is enabled by either a software or hardware enable command.

The 3156B can be triggered from a number of sources: trigger signal from a front panel connector, trigger signal on one of the VXI backplane TTLTRG<n> or software trigger.

The trigger signal, whether it comes from the front panel or from the VXIbus, is routed through some electrical circuits. These circuits cause some small delay known as system delay. System delay cannot be eliminated completely. It is, however, minimized in the 3156B to approximately 150ns. System delay is a factor that must be considered when applying a trigger signal. It defines how long it will take from a valid trigger edge to the moment that the output reacts.

Delayed Trigger

The delayed trigger run mode is exactly the same as the trigger mode except a programmable delay inhibits signal output for a programmable period after a valid trigger. The delay time defines the time that will lapse from a valid trigger (hardware or software) to output. The delay is programmable in steps of 20ns from 500ns to 21 seconds. The trigger delay can be applied to all run modes: continuous, trigger and burst. Delayed trigger does not affect the gated run mode.

Four conditions are required for the 3156B to output waveforms in delayed triggered run mode: 1) Output is on 2) Output is enabled by either a software or hardware enable command 3) Delayed trigger is on and 4) Delay value is programmed.

Burst

The burst mode is an extension of the triggered mode where the 3156B can be programmed to output a pre-determined number of waveforms. The source to trigger the counted burst cycle can be selected from a front panel connector, VXI backplane trigger lines, or software command. Trigger delay and re-trigger delay modes apply to the burst run mode.

Two conditions are required for the 3156B to output bursts of waveforms: 1) Output is on 2) Output is enabled by either a software or hardware enable command.

Mixed Triggers

Mixed trigger run mode define a special sequence where the first trigger source is accepted from a software command only and then subsequent triggers are accepted from a hardware source only. In this mode, trigger delay apply to the first software command only, subsequent triggers will generate waveforms without delays.

Re-Trigger

The Re-trigger run mode requires only one trigger command to start a sequence of triggered or counted burst of signals. The re-trigger delay defines the time that will lapse from the end of a signal to the start of the next signal. Re-trigger delay is programmable in steps of 20ns from 500ns to 21 seconds. Re-trigger delay operates in conjunction with triggered and counted burst modes only.

Four conditions are required for the 3156B to output waveforms in delayed triggered run mode: 1) Output is on 2) Output is enabled by either a software or hardware enable command 3) Re-trigger is on and 4) Re-trigger delay value is programmed.

Gated

In gated mode, the 3156B generates output waveforms between two gating signal. Only hardware triggers can be used to open and close the gate. The gate opens on the first trigger transition and closes on the second transition. Trigger level and trigger slope are programmable. Trigger delay and re-trigger do not apply to the gated run mode.

Output Type

The 3156B can output six types of waveforms: standard, arbitrary, sequenced, modulated, digital patterns and video stroke waveforms. In some cases, different waveform types can be assigned to each channel however, these combinations are limited and should be observed otherwise they could result in setting conflicts and commands lockout. Possible combination between channel output type and run modes are listed in Chapter 3 of this manual.

Standard Waveforms

The 3156B can generate an array of standard waveforms. The waveforms are generated electronically from standard equations and converted to waveform coordinates that are downloaded to the working memory. Unlike analog function generators that use electrical circuits to produce the wave shapes, the 3156B must compute the waveform coordinates every time a new function is selected or every time the parameters of the function change.

The 3156B can produce 9 standard waveforms: sine, triangle and square, ramp and pulse, sinc, gaussian and exponential pulses and dc. Some of the waveforms parameters can be modified such as start phase for sine and triangle, duty cycle for square, zero crossings for sinc etc. The standard waveforms are the most commonly used wave shapes and therefore were collected to a library of standard waveforms that can be used without the need to compute and adjust waveform coordinates.

The repetition rate of the standard waveforms is given in units of Hz. Both channels share the same clock source and therefore, when a standard function shape is selected for re-play for both channels, the frequency of the waveforms is the same at the output connectors of both channels. Also, when standard waveforms are used, both channels share the same run mode, trigger and re-trigger delay settings. On the other hand, each channel can have a unique set of waveform, amplitude, offset and waveform parameters without interference between the channels.

When both channels are programmed for standard waveforms, the skew between the channels is minimal. Refer to Appendix A for skew between channels specification.

Half-Cycle Waveforms

Besides the 9 standard waveforms, the 3156B can generate half-cycle waveforms from sine, triangle and square shapes. The operation of the half-cycle function is different from the operation of the standard waveforms in the sense that half-cycle delay must be programmed for these functions, which determines the time lapse between the end of the first half-cycle and the start of the second half-cycle. Both channels can be programmed to produce half-cycle waveforms however, if one channel only is required to output half-cycle, the other channel must be programmed to output continuous waveforms only.

The half-cycle delay is programmable from 500ns to 21 seconds. If it is programmed to 0, the half cycle function automatically defaults to triggered run mode, which forces the other channel to default to the same run mode. In other words, when half-cycle function is selected and delay is set to 0, both channels must be programmed to triggered run mode.

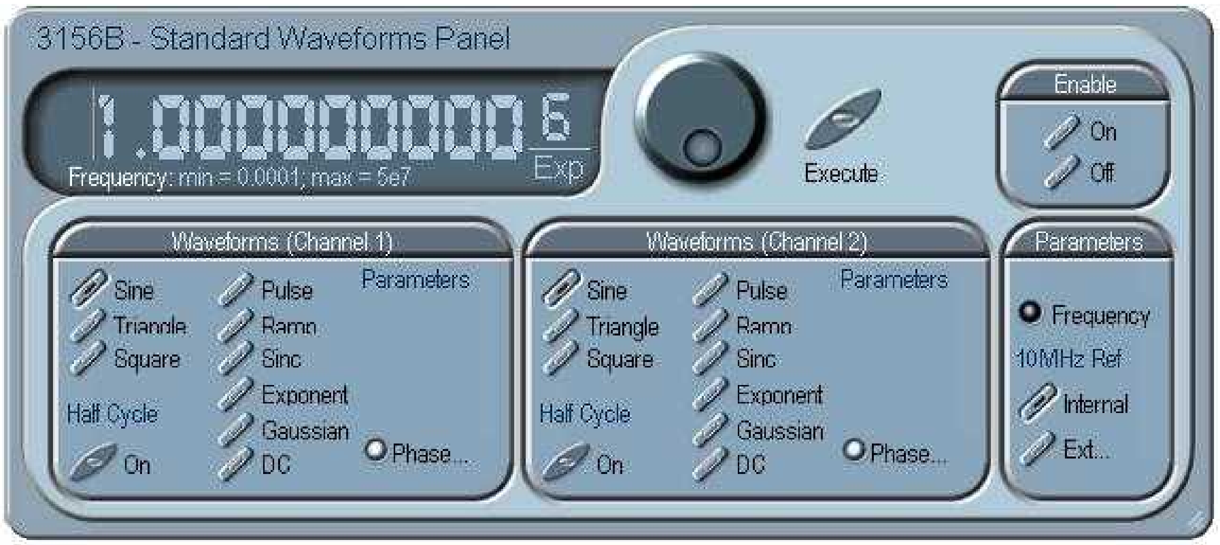


Figure 1-3, WaveCAD Example – Standard Waveforms Panel

Arbitrary Waveforms

One of the main functions of the 3156B is generating real-life waveforms. These are normally not sinewaves and squares but user specific waveforms. Generating such waveforms require external utilities such as MatLAB or even spreadsheets but having the program alone is not enough for the 3156B; Once the waveform is computed and defined, it must be converted to a format which the instrument can accept and downloaded to the generator memory for re-play.

Arbitrary waveforms are stored as digital XY coordinates in a special memory, normally referred to as working memory and each coordinate is referred to as waveform point, or waveform sample. The waveform is better defined if it has many waveform points. For example, with only 8 point, sine waveform will hardly resemble the shape of a sinewave and will look more like an up-down staircase, but with 100 points, the same sine waveform will look almost perfect.

The final shape of the waveform is produced by a DAC (Digital to Analog Converter) The waveform samples are clocked to the DAC at the rate defined by the sample clock frequency. The output of the DAC converts the digital data to analog levels and passes on the signal to the output amplifier. The shape of the function is more or less the same as it comes out of the DAC except it could be amplified or attenuated, depending on the require amplitude level.

The size of the working memory is limited to the way the hardware was designed. The 3156B has 1Meg points available to build one or more waveforms. There is no need to use the entire memory for only one waveform; The memory can be divided into smaller segments loaded with different waveforms while the instrument can be programmed to output one segment at a time.

The 3156B has separate arbitrary waveform memories for each channel and each channel can be loaded with different waveforms. Channels are not limited by the number of segments and by the shape of the waveforms regardless if the channels operate in the same or in different modes.

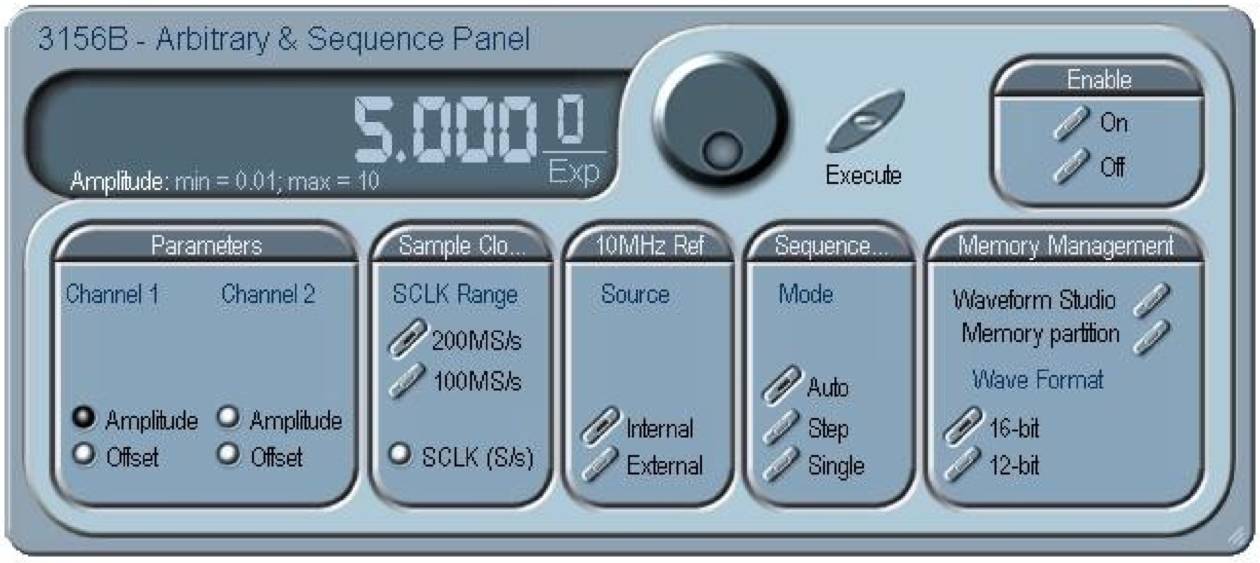


Figure 1-4, WaveCAD Example – Arbitrary & Sequenced Waveforms Panel

Sequenced Waveforms

The sequence generator is a very powerful tool that lets you link and loop segments in any way you desire. As a simple example of a sequenced waveform, look at **Figures 1-5 through 1-7**. The waveforms shown in these figures were placed in memory segments 1, 2 and 3, respectively. The sequence generator takes these three waveforms links and loops them in a predefined order to generate the waveform shown in **Figure 1-8**.

The sequence circuit is useful for generating long waveforms with repeated sections. The repeated waveform has to be programmed once and the repeater will loop on this segment as many times as selected. When in sequenced mode, there is no time gap between linked or looped segments.

The 3156B has two separate sequence generators – one for each channel. Each sequence generator is dedicated to its own channel.

Sequence tables must be loaded to the generator before sequenced waveforms can be generated. The data for the sequence table is first prepared on an external platform, then downloaded to the generator. **Figure 1-9** shows an example how to define a sequence using WaveCAD.

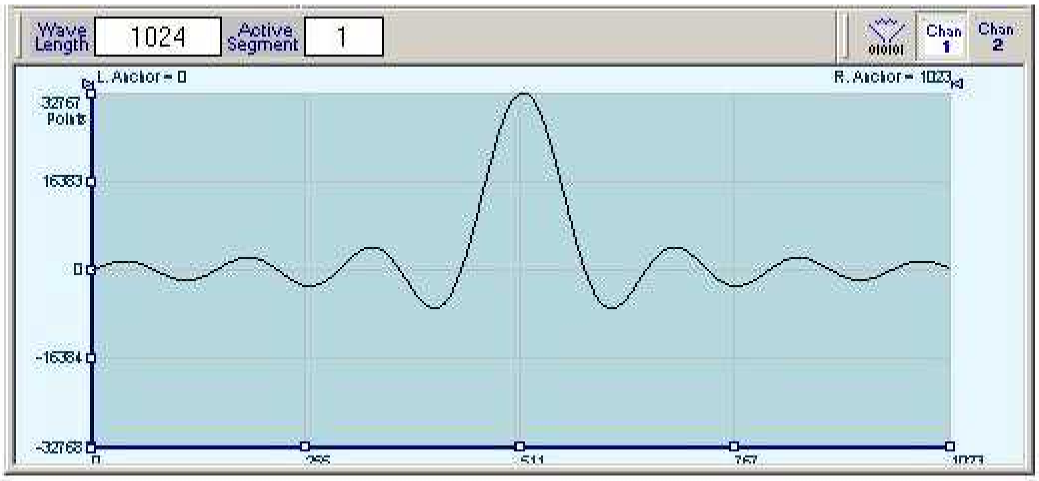


Figure 1-5, Segment 1 Waveform - Sinc

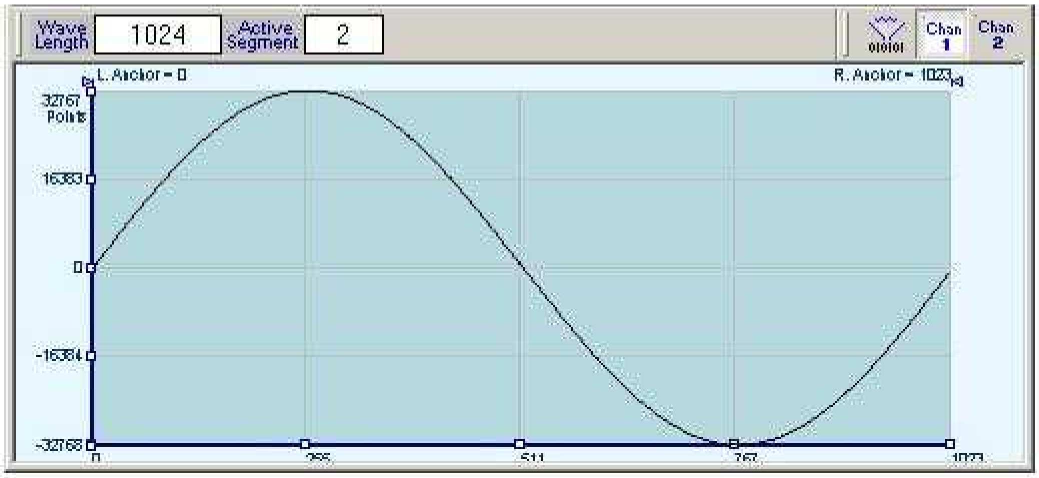


Figure 1-6, Segment 2 Waveform - Sine

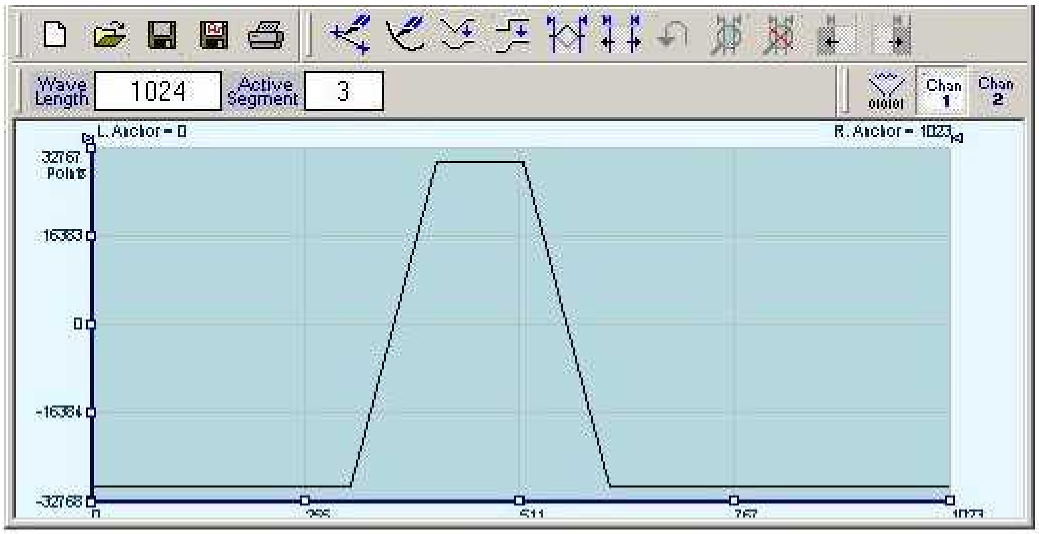


Figure 1-7, Segment 3 Waveform – Pulse

The following sequence was made of segment 2 repeated twice, segment 1 repeated four times, and segment 3 repeated two times.

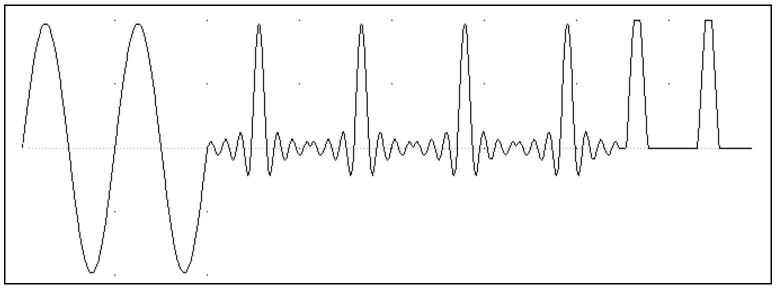
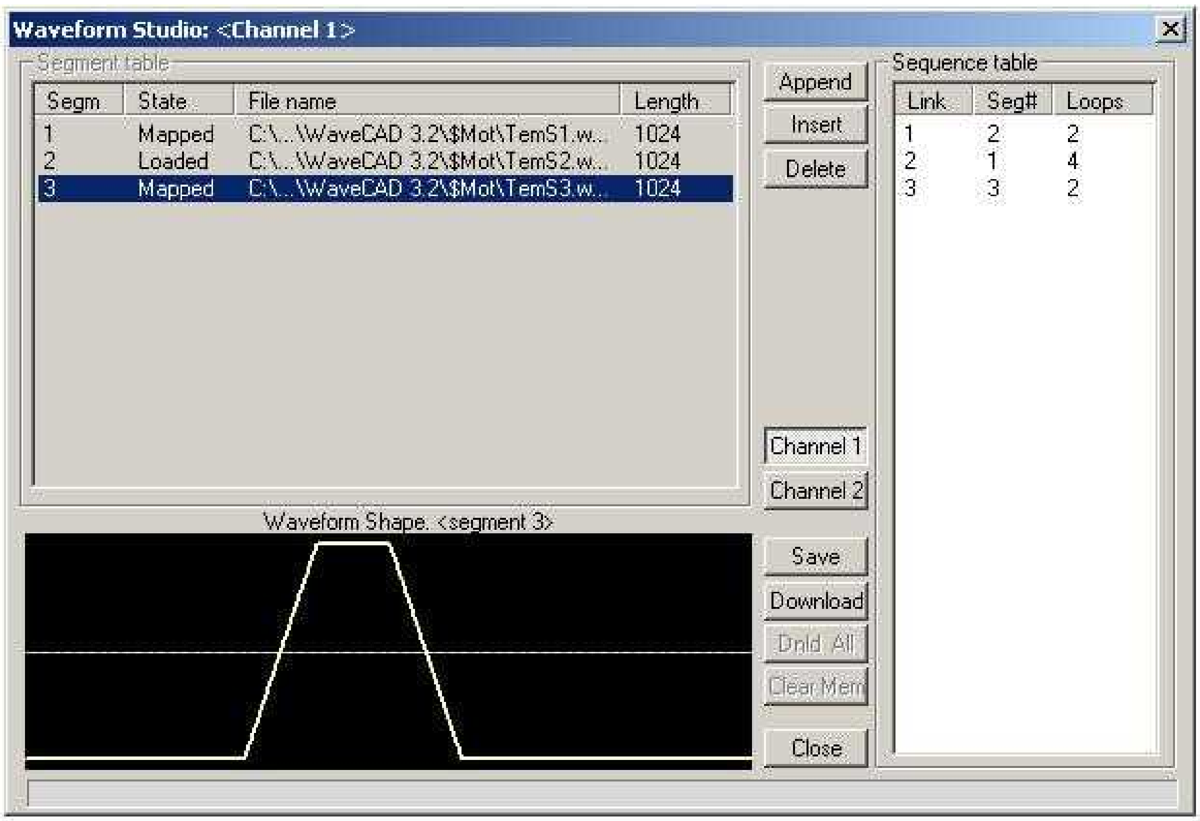


Figure 1-8, Sequenced Waveform

The table below from WaveCAD's Waveform Studio, shows how the same sequence can be defined using WaveCAD.



Segment table

Segm	State	File name	Length
1	Mapped	C:\...\WaveCAD 3.2\$Mot\TemS1.w...	1024
2	Loaded	C:\...\WaveCAD 3.2\$Mot\TemS2.w...	1024
3	Mapped	C:\...\WaveCAD 3.2\$Mot\TemS3.w...	1024

Sequence table

Link	Seg#	Loops
1	2	2
2	1	4
3	3	2

Figure 1-9, WaveCAD's Waveform Studio

Modulated Waveforms

Using the latest DDS technology, the 3156B is capable of producing an array of modulation which, places this generator in-line with high performance, modulation generators. The 3156B can produce: Sweep, FSK, Frequency Hops, AM, FM and the most advanced modulation function - Arbitrary FM. When modulation is used from one channel, the other channel can either be in the same modulation mode, or generate an AC continuous function such as standard or arbitrary waveforms. Description of the various modulation functions is given below.

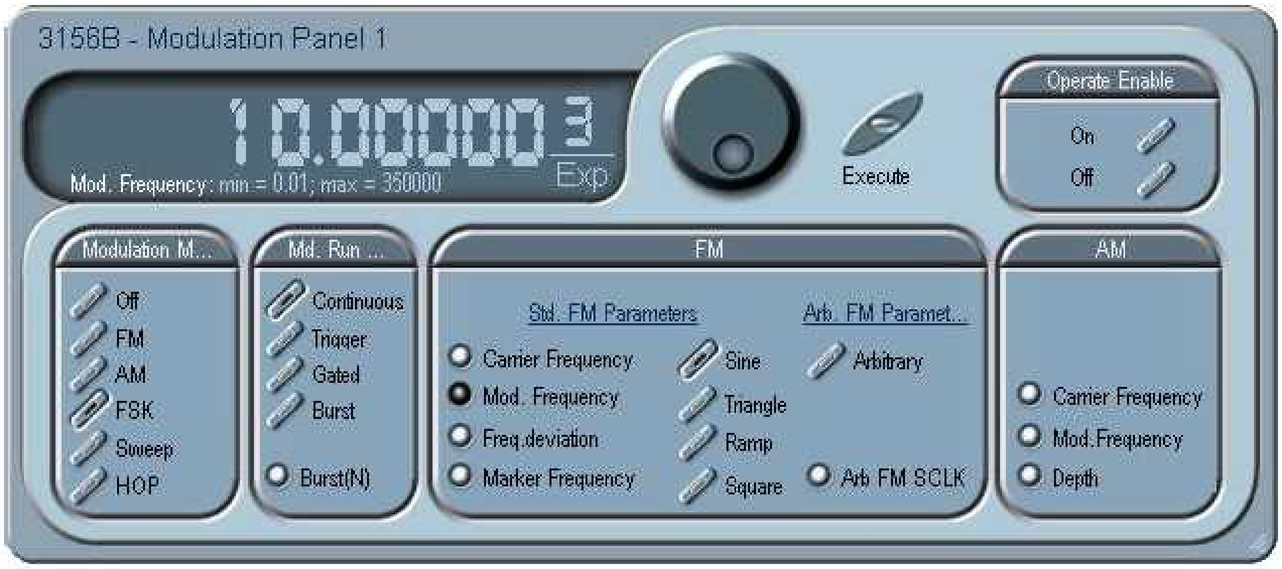


Figure 1-10, WaveCAD Example - Modulation panel

Sweep

Sweep modulation allows carrier waveform (CW) to sweep from one frequency, defined by the sweep start parameter to another frequency, defined by the sweep stop parameter. Note that CW is sinewave only. The start and stop frequencies can be programmed with 10 digits throughout the entire frequency range of the instrument, from 100μHz to 25MHz. There are other parameters that control how the waveforms are swept, these are:

Sweep Time – defines the time that will lapse from sweep start to sweep stop frequencies. Sweep time is programmable from 1.4μs to 40s.

Sweep Step – selects between linear or logarithmic steps.

Sweep Direction – defines if the carrier will sweep from start to stop or from stop to start frequencies.

Marker Position – programs a unique frequency where the SYNC output generates a pulse to mark this frequency.

The 3156B starts to sweep following an enable command. The source of the enable command is selectable from software, front panel trigger input, or from a backplane TTLTrg(n) line. When sweep function is selected but an enable command has not been received yet, the 3156B outputs continuous carrier frequency.

FSK

FSK (Frequency Shift keying) modulation allows frequency hops between two pre-programmed frequencies: Carrier Waveform Frequency and Shifted Frequency. Note that CW is sinewave only. The CW and shifted frequencies can be programmed with 10 digits throughout the entire frequency range of the instrument, from 100μHz to 25MHz. There are other parameters that control how the frequencies are shifted, these are:

Baud Rate – defines the rate of which the frequencies are toggled. The rate can be programmed within the range of 1bits/s to 10Mbits/s.

FSK Date – defines the sequence of which the frequencies will toggle. FSK data is stored in an external table. The length of the table is limited from 1 to 4096 toggle sequences.

The 3156B requires an enable command to start shifting frequencies. The source of the enable command is selectable from software, front panel trigger input, or from a backplane TTLTrg(n) line. When FSK function is selected but an enable command has not been received yet, the 3156B outputs continuous carrier frequency.

Frequency Hop

Frequency hop modulation allows frequency hops throughout the entire range of the instrument. The base frequency is always CW (sine waveform). Frequencies are programmed with 10 digits resolution from 100μHz to 25MHz. There are other parameters that control how the frequencies are hoped, these are:

Hop Sequence – defined using a hop table. The hop table is a list of “0”s and “1”s which determine the sequence. “0” defines CW and “1” defines shifted frequency. The hop table is limited to 4096 frequency hops.

Dwell Time – defines the lapse of time for a hop step. There are two options: fixed or variable. Using the variable time option, each step can be programmed to have a unique dwell time value. Dwell time is programmable in steps of 20ns from 500ns to 21 seconds.

The 3156B requires an enable command to start hops between frequencies. The source of the enable command is selectable from software, front panel trigger input, or from a backplane TTLTrg(n) line. When frequency hop function is selected but an enable command has not been received yet, the 3156B outputs continuous carrier frequency.

AM

The AM function allows amplitude modulation of a carrier waveform (CW). The carrier waveform is sinewave and it is being modulated by an internal waveform, normally referred to as envelop waveform. The envelop waveform is also sinewave. Carrier waveforms are programmed with 10 digits resolution from 1Hz to 25MHz. There are other parameters that control how the frequencies are hoped, these are:

Envelop Frequency – defines the frequency of the modulating waveform. The modulating waveform is programmed from 10mHz to 100kHz.

Modulation depth – programmed in units of % and defines the depth of the modulating envelop. Modulation depth is programmed from 0% to 100%.

The 3156B requires an enable command to start amplitude modulation. The source of the enable command is selectable from software, front panel trigger input, or from a backplane TTLTrg(n) line. When AM function is selected but an enable command has not been received yet, the 3156B outputs continuous carrier frequency.

FM

The FM function allows frequency modulation of a carrier waveform (CW). The carrier waveform is sinewave and it is being modulated by an internal waveform, normally referred to as modulating waveform. The modulating waveform can be selected from sine, triangle or square waveforms. Carrier waveforms are programmed with 10 digits resolution from 1Hz to 25MHz. There are other parameters that control how the frequencies are hoped, these are:

Modulation Frequency – defines the frequency of the modulating waveform. The modulating waveform is programmed from 10mHz to 100kHz.

Peak Deviation – defines the range of frequencies of which the modulation will go through. The peak value is symmetrical around the value of the carrier waveform frequency.

Marker Position – programs a unique frequency where the SYNC output generates a pulse to mark this frequency.

The 3156B requires an enable command to start frequency modulation. The source of the enable command is selectable from software, front panel trigger input, or from a backplane TTLTrg(n) line. When FM function is selected but an enable command has not been received yet, the 3156B outputs continuous carrier frequency.

Arbitrary FM

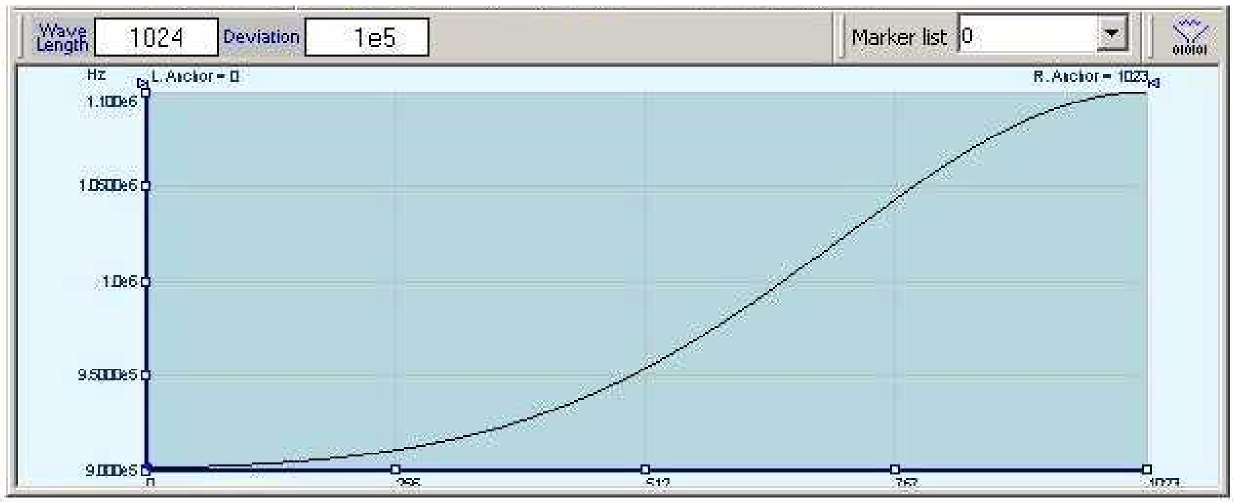
The Arbitrary FM function is very similar to the standard FM function except the modulating waveform is user programmable. The arbitrary waveform to modulate the carrier is generated using the FM composer panel from WaveCAD. It also can be generated by the user using other utilities. The arbitrary FM waveform has an array size of 20000 frequency points. If you look at the following FM composer example, you'll see that the vertical scale is made of frequency points, The change in frequency will follow the curve shown in the FM composer panel.

The frequency of the arbitrary FM wave is computed in the same way as a standard arbitrary waveform.

Frequency = Sample Clock / Number of waveform points

In the example below, the frequency of the waveform will change from 900kHz to 1.1MHz. The sample clock for the modulating waveform can be programmed from 1S/s to 5MS/s while the carrier waveform frequency can be selected from 1Hz to 25MHz. Note that the CW wave is always sine waveform however, the modulating waveform can take any shape defined by the FM composer panel.

The 3156B requires an enable command to start arbitrary frequency modulation. The source of the enable command is selectable from software, front panel trigger input, or from a backplane TTLTrg(n) line. When FM function is selected but an enable command has not been received yet, the 3156B outputs continuous carrier frequency.



Wave Length 1024 Deviation 1e5 Marker list 0

Figure 1-11, WaveCAD Example - Arbitrary FM Composer Panel

Modulation Run Modes

Modulation can be generated by the 3156B using the same run modes as for standard, arbitrary and sequenced waveforms except when in modulation function, where run mode options take different meaning. Modulation can be programmed to use one of the following: continuous, triggered, mixed triggers, burst, delayed trigger and re-trigger.

As a two-channel instrument, each channel can be programmed to generate different function. The following rules must be observed:

1. Both channels can be programmed to output the same modulation function, using the same modulation run mode.

2. When one channel is generating continuous modulation, the other channel can be programmed to output a non-modulated, continuous AC signal such as standard or arbitrary waveform.

3. When one channel is modulated and the other not, they must share the same run mode. For example, if channel 1 generates triggered AM, channel 2 can be set to output triggered pulse. Trigger parameters such as trigger source, trigger level and slope are common to both modulated and non-modulated functions.

Description of the various modulation run modes is given below.

Continuous

In continuous modulation, the carrier waveform is modulated continuously. Two conditions are required for the 3156B to generate continuous modulation: 1) Output is on and 2) Output is enabled by either a software or hardware enable command. Modulation can be stopped (always at the end of the last cycle of the modulating waveform) with a software enable off command. Chapter 3 lists the various options of enabling and disabling the output waveform in the various run modes and enable sources. When not enabled, the 3156B outputs carrier waveform (sine) with frequency value defined by the carrier frequency parameter.

Triggered

In triggered mode, the 3156B circuits are armed to generate a single cycle of modulated waveform. The trigger circuit is sensitive to transitions at the trigger input. Select between positive or negative transitions to trigger the instrument. You may also program the trigger level to the desired threshold level.

When not enabled, the 3156B outputs carrier waveform (sine) with frequency value defined by the carrier frequency parameter. When enabled, the generator outputs one modulation cycle and resumes carrier frequency at the end of the modulating waveform.

Two conditions are required for the 3156B to generate triggered modulation: 1) Output is on and 2) Output is enabled by either a software or hardware enable command. The 3156B can be enabled from a number of sources: trigger signal from a front panel connector, trigger signal on one of the VXI backplane TTLTRG<n> or software command.

Delayed Trigger

The delayed trigger modulation run mode is exactly the same as the trigger mode except a programmable delay inhibits the modulation for a programmable period after a valid trigger. The delay time defines the time that will lapse from a valid trigger (hardware or software) to the first modulation cycle. The delay is programmable in steps of 20ns from 500ns to 21 seconds. The delayed trigger can be applied to all modulation run modes: continuous, trigger and burst.

Four conditions are required for the 3156B to output waveforms in delayed trigger modulation run mode: 1) Output is on 2) Output is enabled by either a software or hardware enable command 3) Delayed trigger is on and 4) Delay value is programmed.

Burst

The modulation burst mode is an extension of the triggered modulation mode where the 3156B can be programmed to output a pre-determined number of modulated cycles. The source to trigger the counted burst cycle can be selected from a front panel connector, VXI backplane trigger lines, or software command. Trigger delay and re-trigger delay modes apply to the modulation burst run mode.

Two conditions are required for the 3156B to output bursts of modulated waveforms: 1) Output is on 2) Output is enabled by either a software or hardware enable command.

Mixed Triggers

Mixed trigger run mode define a special sequence where the first trigger source is accepted from a software command only and then subsequent triggers are accepted from a hardware source only. In this mode, trigger delay apply to the first software command only, subsequent triggers will generate modulated waveforms without delays.

Re-Trigger

The Re-trigger run mode requires only one trigger command to start a sequence of triggered or counted burst of modulated waveforms. The re-trigger delay defines the time that will lapse from the end of a modulation cycle to the start of the next modulation cycle. Re-trigger delay is programmable in steps of 20ns from 500ns to 21 seconds. Re-trigger delay operates in conjunction with triggered and counted burst modes only.

Four conditions are required for the 3156B to output modulated waveforms in delayed trigger modulation run mode: 1) Output is on 2) Output is enabled by either a software or hardware enable command 3) Re-trigger is on and 4) Re-trigger delay value is programmed.

Digital Patterns

Digital patterns are generated through a special front-panel connector, one for each channel. Patterns are user programmable using 12-bit hex words and are stored in pattern tables, as shown in the following figure. The 3156B generates two pattern types: Free-run and Stimulus. Both types are very similar except the free-run has hold time parameter where each pattern can be designed with its own unique hold time. Each channel has its own digital output. The digital lines are differential, terminated into 50Ω to -2V.

Pattern update frequency is 100μHz to 100MHz and hold time can vary from 1 to 10e9. The number of patterns a table can hold is 512k.

Data lines for the digital outputs are routed from the DAC inputs through special buffers, directly to the output connectors and therefore, these outputs are always active, regardless if the digital patterns mode is selected or not. All run modes that are available for the standard and arbitrary waveforms are also available for the digital patterns outputs.

Video Stroke Waveforms

Video stroke waveforms are used for analog monitors where one channel drives the “X” axis and the other channel drives the “Y” axis. There are ten standard video stroke characters built in the 3156B software however, using the arbitrary capability of the instrument, one can build and generate multiple characters to match any application. The standard, built-in characters are: Cross Locator, Cross Hair, Positioned Square, Vertical Marker Line, Horizontal Marker Line, Right Hand Arrow, Left Hand Arrow, Diamond Overlay, Inverted Triangle and Upright Triangle.

The video stroke generator can easily be adopted for testing XY recorders. Using the automatically stepped offset generator, the offset motion covers a span of -4.995V to +4.995V with step increments as small as ±1mV (up to ±9.99V).

Output State

The main outputs can be turned on or off. The internal circuit is disconnected from the output connector by a mechanical switch (relay). This feature is useful for connecting the main outputs to an analog bus. For safety reasons, when power is first applied to the chassis, the main output is always off.

Enable State

In general, the 3156B starts generating output functions only after an enable command has been received. The source of the enable command is selectable from software command, front panel trigger input or from a backplane TTLTrg(n) line. The output waveform, in some cases, can be stropped using the enable off command. Summary of run modes and their respective enable sources is given in Chapter 3 of this manual.

Front Panel Indicators

There are four LED's on the front panel. The FAIL LED (Red) illuminates at power up until the 3156B has passed self-test. The FAIL LED remains on if the self-test fails.

The ACCESS LED (Amber) illuminates each time a command has been received by the 3156B. This light remains on during shared memory data transfer.

When the output state is on, the OUTPUT LED (Green) light illuminates. There are two LED's, one for each channel.

Programming the 3156B

The 3156B has no controls on its front panel. Instrument functions, parameters, and modes can only be accessed through VXIbus commands. There are a number of ways to "talk" to the instrument. They all require that an appropriate software driver be installed in the Resource Manager (slot 0). The rest is a matter of practice and knowledge of the language in use. There are other system considerations like address selection that have to be settled before programming the instrument. These topics are discussed in later chapters.

Low level programming of the 3156B is done using function call. Programming aspects are covered in Chapters 3 and 4.

High level drivers like VXI*plug&play* panels and WaveCAD are beyond the scope of this manual. Contact your EADS North America Defense Test and Services, Inc. representative for more information about high level drivers for the 3156B.

Chapter 2
CONFIGURING THE INSTRUMENT

Installation Overview

This chapter contains information and instructions necessary to prepare the 3156B for operation. Details are provided for initial inspection, grounding safety requirements, repackaging instructions for storage or shipment, logical address selection and installation information.

Unpacking and Initial Inspection

Unpacking and handling of the generator requires normal precautions and procedures applicable to handling of sensitive electronic equipment. The contents of all shipping containers should be checked for included accessories and certified against the packing slip to determine that the shipment is complete.

Safety Precautions

The following safety precautions should be observed before using this product and associated computer. Although some instruments and accessories would normally be used with non-hazardous voltages, there are situations where hazardous conditions may be present.

This product is intended for use by qualified persons who recognize shock hazards and are familiar with the safety precautions required to avoid possible injury. Read the operating information carefully before using the product.

Exercise extreme caution when a shock hazard is present. Lethal voltage may be present on cables, connector jacks, or test fixtures. The American National Standard Institute (ANSI) states that a shock hazard exists when voltage levels greater than 30V RMS, 42.4V peak or 60 VDC are present.

WARNING:

For maximum safety, do not touch the product, test cables, or any other instrument parts while power is applied to the circuit under test. ALWAYS remove power from the entire test system before connecting cables or jumpers, installing or removing cards from the computer, or making internal changes such as changing the module address.

Do not touch any object that could provide a current path to the common side of the circuit under test or power line (earth) ground. Always keep your hands dry while handling the instrument.

When using test fixtures, keep the lid closed while power is applied to the device under test. Carefully read the Safety Precautions instructions that are supplied with your test fixtures.

Before performing any maintenance, disconnect the line cord and all test cables. Only qualified service personnel should perform maintenance.

Performance Checks

The instrument has been inspected for mechanical and electrical performance before shipment from the factory. It is free of physical defects and in perfect electrical order. Check the instrument for damage in transit and perform the electrical procedures outlined in the section entitled **Unpacking and Initial Inspection.**

Grounding Requirements

To ensure the safety of operating personnel, the U.S. O.S.H.A. (Occupational Safety and Health) requirement and good engineering practice mandate that the instrument panel and enclosure be "earth" grounded. Although BNC housings are isolated from the front panel, the metal part is connected to earth ground.

WARNING:

Do not attempt to float the output(s) from ground as it may damage the 3156B and your equipment.

Long Term Storage or Repackaging For Shipment

If the instrument is to be stored for a long period of time or shipped immediately, proceed as directed below. If you have any questions, contact your local EADS North America Defense Test and Services,Inc. Customer Service Department.

1. Repack the instrument using the wrappings, packing material and accessories originally shipped with the unit. If the original container is not available, purchase replacement materials.
2. Be sure the carton is well sealed with strong tape or metal straps.
3. Mark the carton with the model and serial number. If it is to be shipped, show sending and return address on two sides of the box.

NOTE:

If the instrument is to be shipped to EADS North America Defense Test and Services, Inc. for calibration or repair, attach a tag to the instrument identifying the owner. Note the problem, symptoms, and service or repair desired. Record the model and serial number of the instrument. Show the work authorization order as well as the date and method of shipment. ALWAYS OBTAIN A RETURN AUTHORIZATION NUMBER FROM THE FACTORY BEFORE SHIPPING THE INSTRUMENT TO EADS North America Defense Test and Services, Inc.

Preparation for Use

Preparation for use includes removing the instrument from the container box, selecting the required logical address and installing the module in a VXIbus chassis.

Logical Address Selection

The VXIbus Resource Manager identifies modules in the system by the module’s address. VXIbus logical addresses can range from 0 to 255, however, addresses 1 to 254 **only** are reserved for VXIbus modules. Logical address 0 is reserved for the Resource Manager. Logical address 255 permits the Resource Manager to dynamically configure the module logical address.

To change the logical address, use the 8-position DIP switch accessible from the top right side of the module near the rear end of the case (switch S1). The switches are marked with numbers 1 to 8. The 3156B uses binary values (2^0 to 2^7) to set the logical address using the active low address switch. A switch is active when its arm is placed in the ON position.

EADS North America Defense Test and Services, Inc. ships the 3156B with the logical address set to 2.

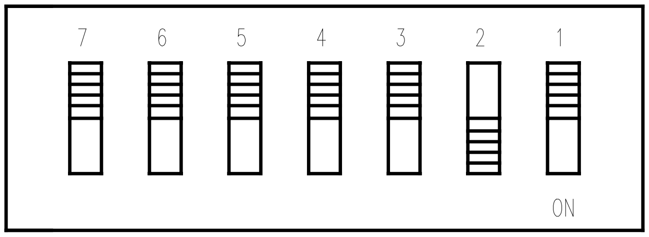


Figure 2-1, Logical Address Set to "2"

Installation

The instrument can be installed in any slot except slot 0 in a VXIbus mainframe. When inserting the instrument into the mainframe, it should be gently rocked back and forth to seat the connectors into the backplane receptacle. The ejectors will be at right angles to the front panel when the instrument is properly seated into the backplane. Use two captive screws above and below the ejectors to secure the instrument into the chassis.

After installation, perform an initial checkout and operational verification. The instrument can be installed in any slot except slot 0 in a VXIbus mainframe. When inserting the instrument into the mainframe, it should be gently rocked back and forth to seat the connectors into the backplane receptacle. The ejectors will be at right angles to the front panel when the instrument is properly seated into the backplane. Use two captive screws above and below the ejectors to secure the instrument into the chassis.

After installation, install the software and drivers on the supplied CD and perform an initial checkout and operational verification. The CD has the following utilities: Plug&Play driver, SFP's (soft front panels), WaveCAD 3.2, manual and links to service centers.

Chapter 3
USING THE INSTRUMENT

Overview

This chapter contains information about how to operate the 3156B. Unlike bench-type instruments, the 3156B must be programmed to turn on functions, change parameters and configure various operating modes. Two graphical interfaces are available for programming the instrument: VXIbus Plug&Play SFP's (soft front panels) and WaveCAD 3.2. Plug&Play drivers and a set of function calls are available for the experienced programmers. Function calls for the 3156B are listed in **Table 4-1**. The following paragraphs describe how to set up the 3156B for different modes and operations with the aid of WaveCAD examples.

Inter-Channel Dependency

The 3156B has two output channels. In general, each channel can generate different signals however, there are some limitations that should be observed that are not acceptable for the instrument. These conditions, when applied will cause setting conflicts. Inter-channel dependency limitations are summarized in the table below. When both channels are placed in waveform generation mode that does not present a setting conflict, each channel can separately have its own waveform and parameters such as amplitude, offset etc.

Having a single sample clock source, the two channels are tightly synchronized. Both channels start the waveforms from the first point and at the same time. Also, the start phase of the two channels is the same except if programmed to a different value.

Table 3-1, Inter-Channel Dependencies

Channel A Waveform	Channel B Waveform	3156B Run Mode	Setting Conflict
Standard	Standard	All	No
	Sequenced	All	Yes
	Modulated	Continuous	No
	Modulated	Triggered	Yes
	Modulated	Gated	Yes

Table 3-1, Inter-Channel Dependencies (continued)

Channel A Waveform	Channel B Waveform	3156B Run Mode	Setting Conflict
Standard	Modulated	Burst	Yes
	Video Stroke	All	Yes
	Digital Patterns	All	Yes
Arbitrary	Standard	All	Yes
	Arbitrary	All	No
	Sequenced	Continuous	No
	Sequenced	Triggered	No
	Sequenced	Gated	No
	Sequenced	Burst	Yes
	Modulated	Continuous	No
	Modulated	Triggered	Yes
	Modulated	Gated	Yes
	Modulated	Burst	Yes
	Video	All	Yes
	Digital	All	No
Sequenced	Standard	All	Yes
	Arbitrary	Continuous	No
	Arbitrary	Triggered	No
	Arbitrary	Gated	No
	Arbitrary	Burst	Yes
	Sequenced	Continuous	No
	Sequenced	Triggered	No
	Sequenced	Gated	No
	Sequenced	Burst	Yes
	Modulated	Continuous	No
	Modulated	Triggered	Yes
	Modulated	Gated	Yes
	Modulated	Burst	Yes
	Video	All	Yes
	Digital	Continuous	No
	Digital	Triggered	No
	Digital	Gated	No
	Digital	Burst	Yes

Table 3-1, Inter-Channel Dependencies (continued)

Channel A Waveform	Channel B Waveform	3156B Run Mode	Setting Conflict
Modulated	Standard	Continuous	No
	Standard	Triggered	Yes
	Standard	Gated	Yes
	Standard	Burst	Yes
	Arbitrary	Continuous	No
	Arbitrary	Triggered	Yes
	Arbitrary	Gated	Yes
	Arbitrary	Burst	Yes
	Sequenced	Continuous	No
	Sequenced	Triggered	Yes
	Sequenced	Gated	Yes
	Sequenced	Burst	Yes
	Modulated	Continuous	No
	Modulated	Triggered	Yes
	Modulated	Gated	Yes
	Modulated	Burst	Yes
	Video	All	Yes
	Digital	Continuous	No
	Digital	Triggered	Yes
	Digital	Gated	Yes
	Digital	Burst	Yes
Video	Standard	All	Yes
	Arbitrary	All	Yes
	Sequenced	All	Yes
	Modulated	All	Yes
	Modulated	Triggered	Yes
	Video	Continuous	No
	Video	Triggered	Yes
	Video	Gated	Yes
	Video	Burst	Yes

Table 3-1, Inter-Channel Dependencies (continued)

Channel A Waveform	Channel B Waveform	3156B Run Mode	Setting Conflict
Digital	Standard	All	Yes
	Arbitrary	All	No
	Sequenced	Continuous	No
	Sequenced	Triggered	No
	Sequenced	Gated	No
	Sequenced	Burst	Yes
	Modulated	Continuous	No
	Modulated	Triggered	Yes
	Modulated	Gated	Yes
	Modulated	Burst	Yes
	Video	All	Yes
	Digital	All	No

Output Termination

During use, output connectors must be properly terminated to minimize signal reflection or power loss due to impedance mismatch. Proper termination is also required for an accurate amplitude level at the main output connector. Use 50Ω cables and terminate the main and SYNC cables with terminating resistors. Always place the 50Ω termination at the far end of the cables.

Input / Output Protection

The 3156B provides protection for internal circuitry connected to input and output connectors. Refer to the specifications in Appendix A to determine the level of protection associated with each input or output connector.

Power On/Reset Defaults

At power-on or as a result of a software reset, the instrument defaults to some factory pre-selected conditions. A complete list of all parameters, their default values, as well as their maximum and minimum values is given in Chapter 4.

Use the following function call to place the instrument in its default state:

```
ri3156B_reset
```

You can also use the Utility panel from WaveCAD to perform the same function. Note that by turning the output on, the 3156B is still not enabled to output waveforms. The output on command activates a mechanical relay that connects the output connector to the internal circuits. When the output on command is activated, the output source impedance is 50Ω.

Enabling Output Waveforms

As was discussed before, the output on command does not enable generation of waveforms. To enable waveforms, one must use the enable on command from WaveCAD or use the appropriate function call.

As an example, look at the Main Panel picture below. The panel shown has button positions as are set by the default state. The Output Control group has three sub-groups of buttons:

State – toggles the output on and off. Remember that the action of the output state button Is to activate a mechanical relay that connects the output connector to the internal circuit.

Enable Source – selects the source of the enable command. The options are: Software, Hardware and Mixed. These are discussed separately in the next topic.

Enable – On, enables the instrument to generate waveforms, Off disables generation of the waveforms. Note, however, that in some cases, the Enable Off command has no effects on the waveform. These cases are summarized in **Table 3-2**.

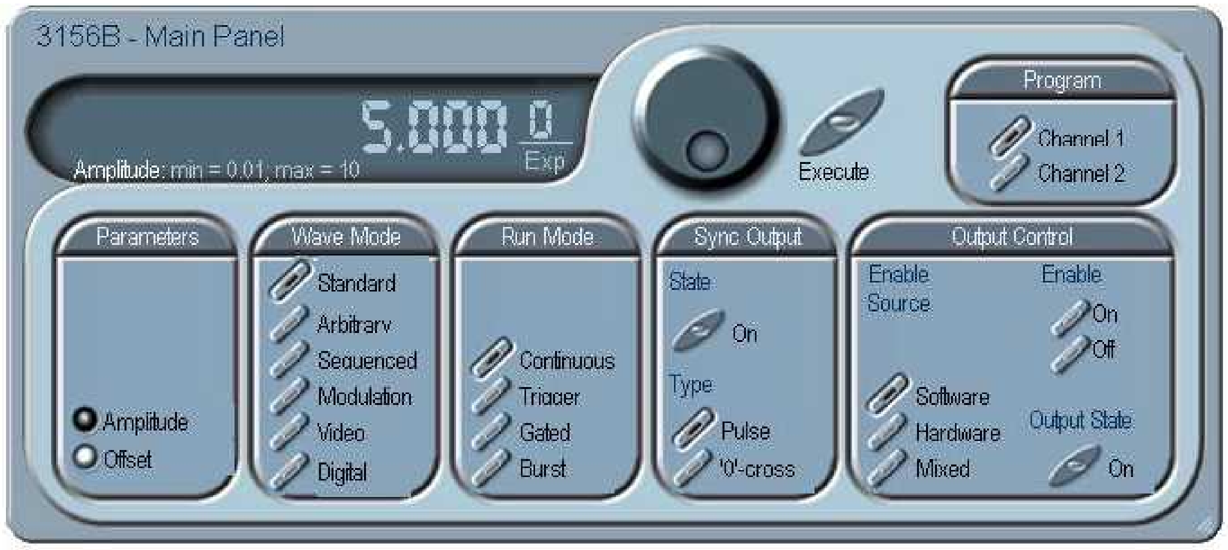


Figure 3-1, Enabling Output Waveforms

By default, the Output State is Off and the Enable Source is software and therefore, the 3156B will output waveforms only if:

1) The State button has been turned On and

2) The Enable has been turned On

The equivalent function calls are:

1) ri3156Bset_output (1)
2) ri3156Bset_enable (0)

If you try this example, the 3156B should output a sine waveform from its Channel 1 output. The sine parameters are by default: 1MHz, 5Vp-p. Note that the enable command affects both channels simultaneously and therefore, channel 2 will output waveforms immediately after the output state is turned on. To control channel 2 parameters, press the Channel 2 button in the Program group.

Selecting an Output Enable Source

An output enable command is necessary for the 3156B to start generating waveforms at the output connectors. It is like a trigger signal except it is not just used for triggered run mode; It is also necessary to have an enable commands for run modes like continuous. Note that the enable command has different meaning for standard/arbitrary and modulated waveforms.

In standard waveforms mode, before an enable command is applied, the output generates DC level, normally around the start of the requested waveform. When an enable command is received, the output generates the waveform. In continuous mode, the output is continuous. In triggered or burst modes, the output resumes idle position at the end of the signal.

In modulation functions, before an enable command is applied, the output generates continuous CW (sinewave) at CW frequency. In continuous mode, an enable command initiates continuous modulation. In triggered or burst modes, the output resumes CW frequency at the end of the modulating wave.

The enable signal can be applied to the 3156B from a number of sources as described below:

Software – defines an enable command is expected from a software source.

Hardware – defines an enable command is expected from either a front panel trigger input or from a backplane TTLTrg(n) signal. One of these options is selected from the Trigger panel.

Mixed – defines a special enable command where the first enable is software while hardware triggers are ignored and subsequent triggers are hardware while software triggers are ignored.

Under some conditions, signals can be turned off at the output connector, using the enable off command. Summary of run modes and their respective enable and disable source is given in **Table 3-2**. Note that in some cases different enable sources are used to turn the signal on and off. The function call to disable the output is:

ri3156Bset_enable (0)

Also note that by using the enable off command, the output circuits are still connected to the output connector and therefore, if you want to assure high impedance at the output connector, use the following command:

ri3156Bset_output (0)

Table 3-2, Enable/Disable Source with Different Run Modes

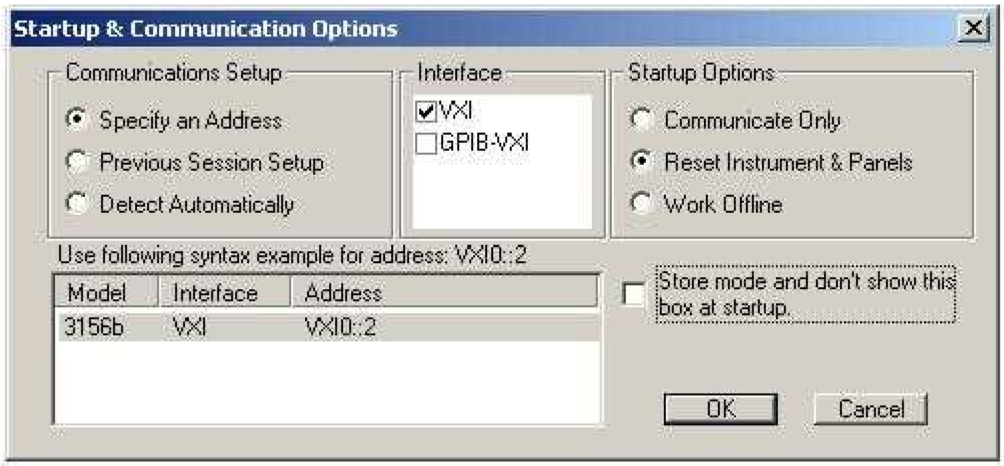
Index	**3156B Run Mode**	**Trigger Delay**	**Re-Trigger**	**Enable Source**	**Disable Source**
1	Continuous	Off	x	Software	Software
2		Off	x	Hardware	Software
3		On	x	Software	Software
4		On	x	Hardware	Software
5	Triggered	Off	Off	Software	None
6		Off	Off	Hardware	None
7		On	Off	Software	None
8		On	Off	Hardware	None
9		Off	On	Software	Software
10		Off	On	Hardware	Software
11		On	On	Software	Software
12		On	On	Hardware	Hardware
22	Triggered, Mixed	Off	x	Software	Software
22		On	x	Software	Software
13	Burst	Off	Off	Software	None
14		Off	Off	Hardware	None
15		On	Off	Software	None
16		On	Off	Hardware	None
17		Off	On	Software	Software
18		Off	On	Hardware	Software
19		On	On	Software	Software
20		On	On	Hardware	Hardware
23	Burst, Mixed	Off	x	Software	Software
23		On	x	Software	Software
21	Gated	x	x	Hardware	Hardware

(x) Don't care

Generating Simple Waveforms

To best way to learn how to control the 3156B is by first practicing with simple waveforms and then step by step building up the knowledge and level of confidence in the performance of the instrument. The first example would be to make the 3156B generate sine waves. Here is how it's done:

Assuming that the chassis is already set up and the resource manager invoked, connect two cables, one from the channel 1 main output and the other from the channel 1 SYNC output to an oscilloscope. Set up the oscilloscope to trigger of the SYNC signal. Invoke WaveCAD and check the appropriate buttons as shown in **Figure 3-2**.



Use following syntax example for address: VXI0::2

Model	Interface	Address
3156b	VXI	VXI0::2

Figure 3-2, Startup & Communications options

To start communicating with your instrument select one of the Communications Setup options. Select the Detect Automatically option to let WaveCAD detect your instrument identification in the system, or select Previous Session Setup if this is not the first time you turn on WaveCAD. Select the Specify an Address option if you know the address however, in this case, you'll have to type in the address using the syntax as shown in **Figure 3-2**.

In the Interface window, you have two options: Select VXI if you are using MXI II, LAN or similar controllers, or select GPIB-VXI for GPIB controllers.

The Startup Options allows you to just initiate communications, reset the instrument and WaveCAD panels, or work offline without communicating with your device.

If you check the Store mode and don't show this box at startup, you'll never have to see this dialog box when you initialize WaveCAD however, you can later select to see this box from the View menu. Press OK to launch WaveCAD, or Cancel to abort. If communication is established with the instrument, the Main WaveCAD panel as shown in **Figure 3-3** will open. Note that the menu bar is displayed on top of the Main menu however, the two parts can be separated and placed in different places on the screen.

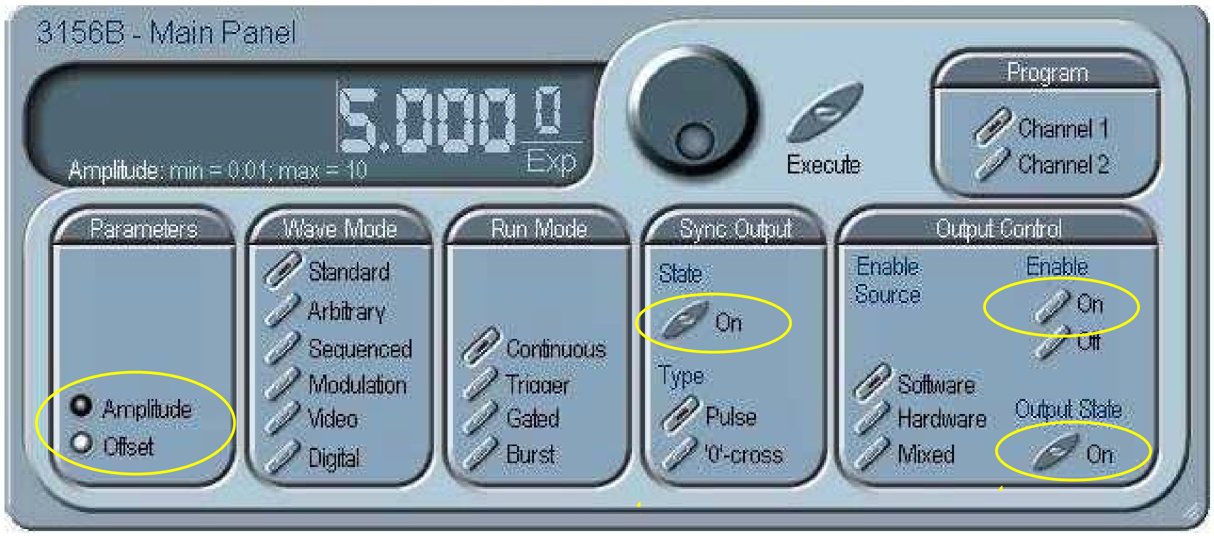


Figure 3-3, WaveCAD Main Panel

Note:

The link in Figure 3-3 shows 3156B, Offline. If such link is established, there is no communications between the 3156B and your controlling device. The link field should show 3156B, VXI0::2 if communications has been established between the instrument and the controlling device. The manual pictures were taken with WaveCAD operating in demo mode and therefore, all references to Offline operation should be replaced with the address setting of your 3156B.

Use the following sequence to start seeing waveforms on your oscilloscope:

1. In the Output Control group, point and click on the Output State On button, the button will change its shape and show a red mark at its center, depicting an LED and designating the output is on.

2. In the Sync Output group, point and click on the State On button, the button will change its shape and show a red mark at its center, depicting an LED and designating the Sync output is on.

Note:

Step 2 turns on the SYNC output only and is optional, in case you want to synchronize your oscilloscope. This step is not required to have a waveform generated from the main output connector.

3. In the Output Control group, point and click on the Enable Output State On button, the button will change its shape and show a red mark at its center, depicting an LED and designating the output enable is on.

Set your oscilloscope and observe the 3156B generates a sine waveform with the following properties: Frequency is 1MHz, offset is 0V and amplitude is 5V. If your amplitude is off by a factor of 2, it could only mean that the instrument has not been terminated properly. The output of the 3156B is calibrated when the signal is applied to a load impedance of 50Ω and therefore either add a feedthrough terminating resistor or modify your oscilloscope setting to have 50Ω termination.

Tip:

When the Output State is off, the front-panel channel 1 and 2 OUT connectors are disconnected from the electrical circuit by a mechanical relay. This is the default state of the 3156B after power is applied to the instrument and is designed this way for safety reason, to block low impedance path to the outside connector. An Enable command or external trigger is required to start generating waveforms. Table 3-2 lists the available enable source for the various run modes of the instrument

If you want to change the amplitude and/or offset of the signal, do the following:

4. In the Parameters group, point and click on the Amplitude text, the LED next to it will change its color and show a dark mark at its center, depicting an illuminating LED and designating that the amplitude parameter has been selected. The value that is associated with the lit LED is displayed on the digital display. Use the dial, keyboard, or the [↑] [↓} keys to adjust the readout to the required setting. After you modify the reading, press Execute to update the 3156B with the new reading. You can modify the offset is a similar way.

Note:

Normal color of the digital reading is dark blue. If you modify the reading, the color changes to a lighter shade of blue, indicating that the 3156B has not been updated yet with the new parameter. Pressing Execute will update the instrument and will restore the color of the digital readout to dark blue, indicating that the displayed value is the same as the generator setting.

Also note that the digital readout has an autodetect mechanism for the high and low limits. You cannot exceed the limits if you are using the dial but only if you use the keypad. In case you do, the program will not let you download an illegal parameter and you'll be requested to correct your setting.

5. Till now you programmed channel 1 only. You can proceed to program the second channel if you point and click on the Channel 2 button in the Program group. When this button is depressed, anything that you program on the MAIN panel will be programmed to the second channel. Note however, that is you selected the Output Enable ON button, you do not have to do it again for the second channel because the enable command works automatically for both channels.

If you are not using WaveCAD, you can execute the same sequence using the following function calls:

ri3156B_init	Initializes the session at the specified address and assigns a handle to the 3156B
ri3156B_reset	Restores factory defaults after power up. Complete listing of the 3156B defaults is given in **Table 5-1**.
ri3156Bset_output (1)	Activate hardware connection to the front panel
ri3156Bset_enable (1)	Output is enabled. This is a software enable command. If you intend to use hardware enable you have to select the hardware enable source with the ri3156B_set_enable_source () function
ri3156B_set_amplitude()	Is used to set up the amplitude level
ri3156B_set_offset()	Is used to set up the offset level

Optional - If you want to program the second channel to output the same waveform, amplitude and offset, use the function call ri3156B_set_active_channel (2) to select the second channel for programming. Subsequent function calls that are defined as independent effect the active channel only. Repeat the above function call setting output on, and amplitude and offset parameters.

ri3156B_close	(Optional) Closes the session at the specified address and releases the handle to the system

Amplitude-Offset Interaction

Amplitude and offsets may be programmed freely for each channel as long as the following relationship is observed:

$$\frac{Amplitude}{2} + |Offset| \leq 10Vp-p \text{ (-5V to +5V peaks)}$$

Amplitude-offset settings outside the above limits will generate "settings conflict" errors.

Selecting and Modifying Std. Waveforms

The paragraphs above described how you can generate a standard sine waveform and how to adjust its amplitude and offset. The next step is to select another waveform from the built-in library of standard waveforms and how to modify their parameters. The WaveCAD panel to use is the STD panel. There was no need to use this panel to select the sine waveform because this is the default waveform after power up. To invoke the STD panel, point and click on the STC button on the panels bar. The panel as shown in **Figure 3-4** will display. You can turn this panel off and on by clicking on the STD button. If the STD panel is already on the screen but hide after another panel, you can bring it to front by clicking on the panel or by clicking on the STD button.

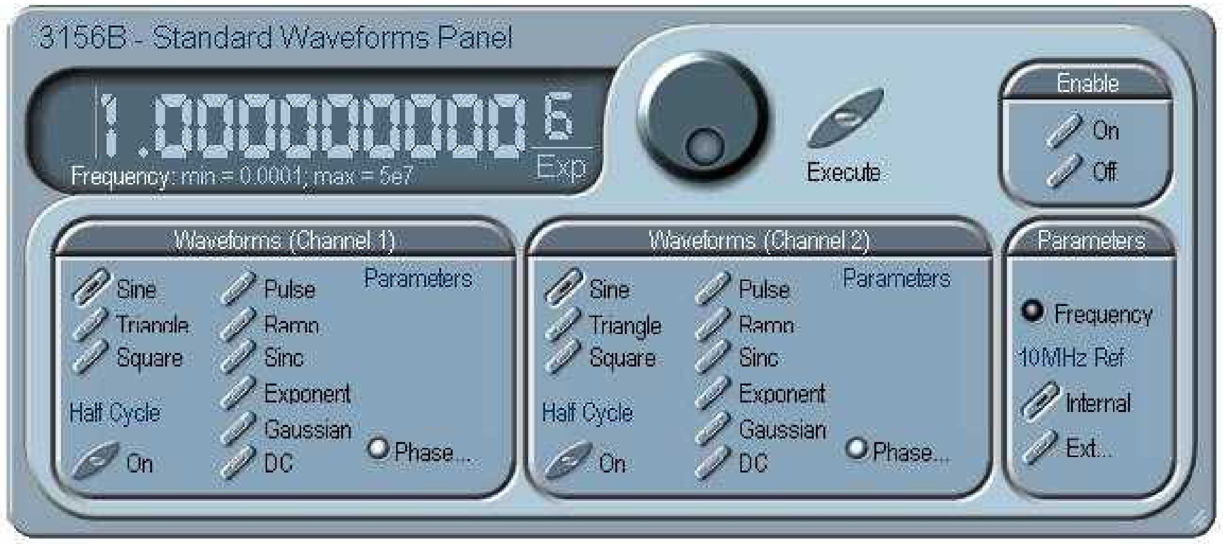


Figure 3-4, The Standard Waveforms Panel

The Standard Waveforms Panel lets you select and modify waveforms, their parameters and the output frequency. You may need to modify parameters in both the MAIN and the STD panels; It will be easier if you select the Tile option from the View menu.

Change waveform by pointing and clicking on the required waveform button. Each channel has its own set of waveform button so modifying one channel does not affect the other. Note that when you change waveforms, the output is updated automatically to the selected waveform.

Waveform parameters are listed below the Parameters title. They appear and disappear on the panel depending on the selected waveform. The example above shows the sine selected for both channels and the phase parameter under the Parameters heading. This is the default setting of WaveCAD after power on reset.

Parameters and frequency are modified in the same way as you modified the amplitude and offset before. Just point and click on the required parameter and then change the display setting and press the Execute button to update the 3156B. The 10MHz reference buttons let you select the source for the internal clock. If you leave as is, the default is Internal. The External option removes the 10MHz reference from the internal source and lets you connect an external reference to the front panel connector. Note however, that if you select the External option and do not connect a source to the front panel, the accuracy of the instrument will degrade without control.

If you are not using WaveCAD, you can execute the same sequence using the following function calls:

ri3156B_set_standard_waveform () will select the function shape

ri3156B_set_sine_wave_phase () or similar function call will program waveform specific parameters

ri3156B_set_frequency () will program the output frequency

ri3156B_set_reference_oscillator () will select between internal or external 10MHz reference source

Using the apply Command

The apply function call provides a shortcut for setting up a waveform, its parameters and other associated functions such as frequency, amplitude and offset without the need to program each of the parameters individually. This also sets the waveform as the active function shape at the output connector; It does not, however, eliminate the need for turning on and enabling the output.

The following is an example of using the apply function call that will generate a square waveform, both channels, at the end of the programming sequence:

ri3156B_init	Initializes the session at the specified address and assigns a handle to the 3156B.
ri3156B_reset	Restores factory defaults after power up. Complete listing of the 3156B defaults is given in **Table 5-1**.
ri3156Bset_output (1)	Turn on channel 1 output.
ri3156B_apply_square_wave ()	This will set square waveform as the active function for channel 1 and will simultaneously program the frequency, amplitude, offset and duty cycle parameters.
ri3156B_set_active_channel (2)	This will select channel 2 as the active channel for subsequent programming routines.
ri3156Bset_output (1)	Turn on channel 2 output.
ri3156B_apply_square_wave ()	This will set square waveform as the active function for channel 2 and will simultaneously program the frequency, amplitude, offset and duty cycle parameters. Note that channel 1 and 2 frequency settings must be identical as there is only one clock source in the system.
ri3156Bset_enable (1)	Output is enabled on both channels.
ri3156B_close	(Optional) Closes the session and releases the handle to the system.

Using the above example, you can use the apply function calls to generate almost any waveform that is possible to generate with the 3156B, including modulation, digital patterns and video strokes. Complete listing of the apply function calls is given in **Table 5-3**.

Selecting the Active Channel

On power up, the instrument defaults to channel 1. That means that each command sent to the generator affects channel 1 parameters and operation only. If you want to send commands to the second channel, you must send a command that will shot down commands stream to channel 1 and open the gateway to channel 2. After you select another active channel, all subsequent commands will affect the selected channel only. Use the following function call to select the channel you want to program:

ri3156B_set_active_channel ()

Subsequent programming routines will affect the active channel only however, function calls that their channel dependency is marked Common, will affect both channels simultaneously.

You can query the active channel using the following function call:

ri3156B_query_active_channel ()

The WaveCAD Main Panel, shown in **Figure 3-2**, demonstrates high-level implementation of channel control.

Selecting an Output Type

There are six main types of waveforms that the 3156B can produce:

1. Standard, from a built-in library of waveforms.
2. Arbitrary, from waveform coordinates that are downloaded to the instrument from a host computer.
3. Sequenced, made of pre-loaded waveform segments that can be linked and looped as configured in a sequence table.
4. Modulated, from a built-in library of modulation functions such as FM, AM, FSK, Sweep and Frequency hops.
5. Digital patterns which provide 12-bit differential outputs through separate connectors.
6. Video Stroke characters, from either a built in library of 10 standard characters or from character coordinates that are downloaded to the instrument from a host computer, similar to arbitrary waveform coordinates.

The 3156B was designed in a way to allow different waveforms and functions to be output from each channel, for example, if channel 1 is programmed to output an arbitrary waveform, the other channel can be programmed to output sequenced waveforms. However, since there is only one clock source in the instrument, there are some function combinations that will not be possible to run simultaneously on both channels and if programmed so will issue a setting conflict error message. The inter-channel dependencies or the conditions that cause setting conflicts are summarized in **Table 3-1**.

When programming the waveform type, you also must consider the run mode as some combinations legal waveform types may cause conflicts with run mode types. For example, arbitrary and sequenced waveforms may not conflict in continuous run mode but if run mode is changed to burst, an error will occur. The conditions that will cause the run mode to generate a settings conflict errors are summarized in **Table 3-1** as well.

There are three function calls that are available for programming the output type. The most important points before you start writing your programming routines is to decide if both channels will be programmed to output two different output types or if they will share the same output type and run modes. Here are the three options:

1. The *ri3156b_set_global_operating_mode ()* - has five variables: Channel 1 waveform, Channel 2 waveform, Carrier Run Mode and Operate Enable Source. This function call is the best to use if channels need to be programmed with different functionality.

2. The *ri3156b_set_common_waveform_mode ()* - has only one variable that sets both channels simultaneously to the same waveform mode. This function call would be the best to use if channels need to be programmed with the same functionality.

3. The *ri3156b_set_common_waveform_mode ()* - allows separate programming of each channel to a different waveform mode. From all functions, this is the least recommended function to be used as it may cause setting conflict errors should one not fully understand the limitation of the product. For example, the default waveform mode is Standard. If you use this function call to change the waveform mode to arbitrary, the 3156B will immediately generate an error because it has only one sample clock source while Standard and Arbitrary waveforms use two different sample clock settings and therefore, this call will generate an error. The Set Active Channel command is required to program each channel separately.

The WaveCAD Main Panel can be used for selecting the output type. Note however, that WaveCAD is using the function call *ri3156b_set_common_waveform_mode ()* to set up both channel simultaneously so if you need to set up the 3156B to output different waveform types from each channel, the only way to use WaveCAD is from its function call editor, as explained in Chapter 4.

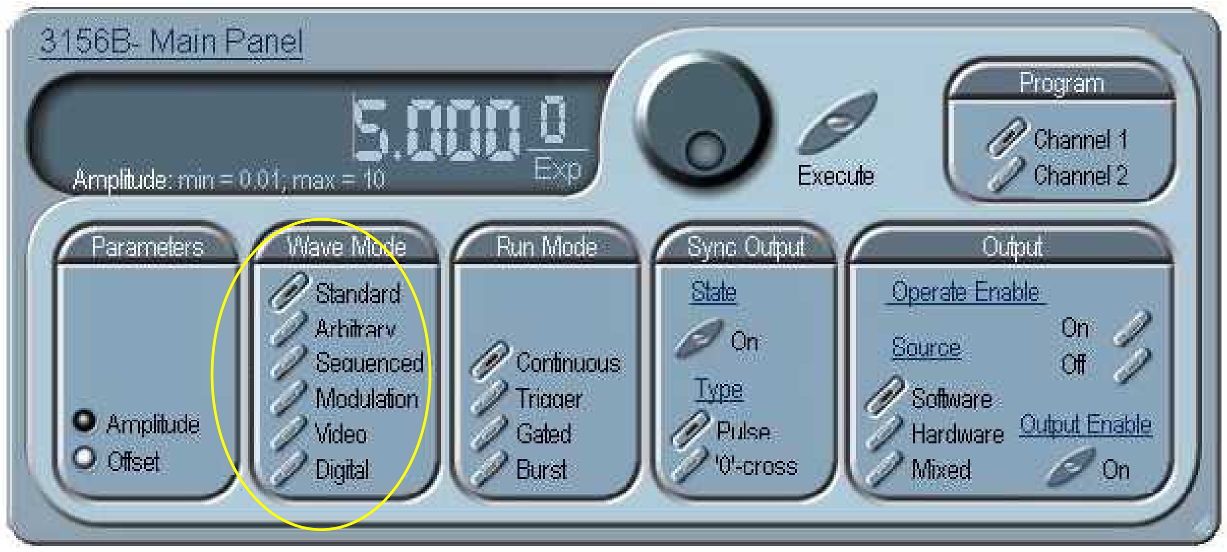


Figure 3-5, Using WaveCAD Main Panel to Select a Waveform Type

Looking at the WaveCAD Main Panel in **Figure 3-5**, the Wave Mode group has buttons associated with each waveform type. Point and click on the required function and the output of the 3156B will by updated immediately with the new type. Note that if channel 2 output is on and enabled, the selection will automatically be associated with the other channels as well.

After you select the waveform type, you can click on the Panels bar to open other panels that contain parameters which are related to the selected waveform type. For example, if you select the Arbitrary waveform type, open the ARB panel to modify and program waveform segments and sample clock settings.

Generating Standard Waveforms

Standard waveforms are built into the instrument's program. Nine standard function shapes are available: Sine, Triangle, Square, Pulse, Ramp, Sinc and Exponential decaying pulses, Gaussian pulse, and dc. Every time a standard function is selected the coordinates for this function are re-computed and placed in the waveform memory. Therefore, there is a minimal delay from when you select the function until the output starts generating the waveform. Use the following function call to select the waveform shape:

ri3156B_set_standard_waveform ()

The function variables are from 0 to 8. Upon power up, the 3156B defaults to 0 (sine waveform).

0	selects sine waveform
1	selects triangle waveform
2	selects square waveform
3	selects pulse waveform

4 selects ramp waveform
5 selects sinc waveform
6 selects exponential waveform
7 selects gaussian waveform
8 selects dc waveform

After you select the required waveform type and shape, you can proceed with modifying the waveform parameters. The function calls that are available for programming standard waveform parameters are summarized in Chapter 5.

Note:

The number of points used to define each Standard Waveform varies according to the programmed frequency and therefore, some parameters may not have any effect on the waveform because too few points are available to generate the waveform mutation.

As was stated in the note above, the number of points may vary depending on the output frequency. The reason for this variation is that even standard waveforms are built from waveform coordinates, similar to arbitrary waveforms except the standard waveforms are stored in a special library for immediate use without the need to compute or download such waveforms. At low frequencies, the number of points for each standard waveform is 1000 so waveform modification is possible to within 1/1000 increments. For example, if you want to modify the duty cycle of a square waveform, your best resolution is 0.001%.

It is different at higher frequencies as the number of points that are being used for generating the waveforms are decreased according to the following relationship:

Output Frequency = Sample Clock Frequency / Waveform Points

Sine the sample clock frequency has final value of 200MS/s, the only way to increase frequency is by reducing the number of waveform points. The reduction in number of points at higher frequency may have a result on the resolution of which you intend to modify the standard waveforms. For example, at 20MHz square wave, the number of points that are available is only 10 and therefore the duty cycle resolution is decreased to 10% increments.

There are two functions that allow you to find out the values of the sample clock and the number of points that are used for a specific function shape at certain frequency, these are:

ri3156B_query_std_sample_clock_freq

ri3156B_query_std_waveform_numb_points

The first function will provide information on the sample clock frequency and the second, on the number of points in the waveform. Note that the sample clock frequency setting cannot be changed when you output a standard waveform from the built in library and therefore this function provides entry to a query only

Generating Arbitrary Waveforms

The 3156B cannot generate arbitrary waveforms without first loading them into its working memory. A description of the arbitrary waveform function and an explanation of how to load waveforms into memory are given in the following paragraphs.

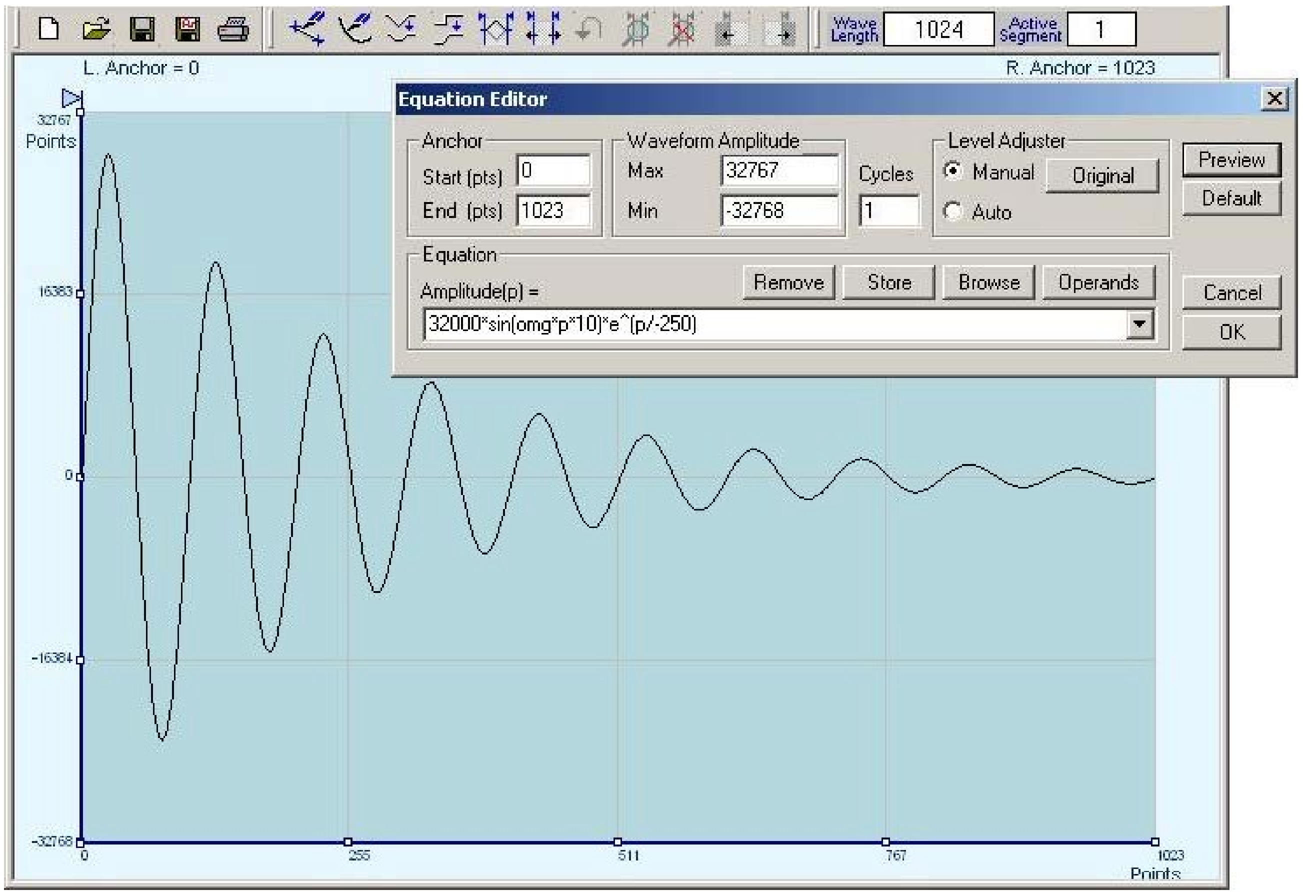
What Are Arbitrary Waveforms?

Arbitrary waveforms are generated from digital data points which are stored in memory. Each data point has a vertical resolution of 16 bits (65536 points), i.e., each sample is placed on the vertical axis with a precision of 1/65536.

The 3156B has waveform memory capacity of 1Meg point. Each horizontal point has a unique address - the first being 00000 and the last depends on the size of the memory. In cases where smaller waveform length is required, the waveform memory can be divided into smaller segments.

When the instrument is programmed to output arbitrary waveforms, the clock samples the data points, one at a time, from address 0 to the last address. The rate at which each sample is replayed is defined by the sample clock rate parameter. The 3156B provides programmable sample clock rates from 1S/s to 200MS/s.

Unlike the built-in library of standard waveforms, arbitrary waveforms must first be loaded into the instrument's memory. Generation of waveform coordinates can be done in multiple ways but the easiest is using WaveCAD for on-screen creation and editing of waveform, no matter how complex. **Figure 3-6** shows an example of a complex waveform that has been created using the Wave Composer. Instructions how to use WaveCAD are given in Chapter 4.



Wave Length: 1024
Active Segment: 1

Anchor
Start (pts): 0
End (pts): 1023

Waveform Amplitude
Max: 32767
Min: -32768

Cycles: 1

Level Adjuster
[✓ Manual] Auto

Equation
Amplitude(p) =
32000*sin(omg*p*10)*e^(p/-250)

Figure 3-6, WaveCAD Example of a Complex Waveform

Arbitrary Memory Management

Correct memory management is required for best utilization of the arbitrary memory. An explanation of how to manage the arbitrary waveform memory is given below.

The arbitrary memory in comprised of a finite number of words. The maximum size of a single waveform that can be loaded into memory is 1Meg long. However, waveforms are created using small sections of the arbitrary memory. The memory can be partitioned into smaller segments (up to 16k) and different waveforms can be loaded into each segment, each having a unique length. Information on how to partition the memory is given in the following paragraphs.

Memory Management Commands

Arbitrary memory can be divided into smaller segments; up to 16k different arbitrary waveforms can be stored in working memory of the 3156B. The length of each segment and its associated sample clock rate are left for the user to program. Segment length are defined using the following function call:

ri3156B_define_arb_segment ()

This function call has two variables: segment number and size. Note that numbers, not names, are assigned to segments. Segment numbers can range from 1 through 16k. The order of assignment is not important as long as segment size does not alter after it has been defined.

You cannot query the segment definition parameters so make sure you keep good track if you intend to partition the memory into many segments. There is a easier way to create a memory partition table for waveform segments, this time creating the complete table from one array. Use the command:

ri3156B_load_segment_table ()

This function has two arrays, one array of segment numbers and another array of size. Using this function, the complete memory partition table will be downloaded to the instrument in one shot. Note however, that you still need to load the memory segments with waveform coordinates before they can be generated at the outputs.

If a mistake is made and removal of one or more segments from the active directory is needed, use the following command:

ri3156B_delete_segment (n)

where <n> is the segment number to be removed from memory. Note that if a segment is deleted, the memory portion that belonged to this segment is no longer accessible. The next segment that is defined will be placed after the last defined memory segment. However, if the last segment is deleted, the next downloaded segment will be written on top of the deleted one. There is danger that by using the delete function calls too often, large portions of memory will remain unused. It is, therefore, suggested that you periodically clear the entire memory and only reload waveforms that you intend to use.

To delete the entire memory partition table use the following function call:

ri3156B_delete_segment (0)

where the variable <0> defines a delete all segments command.

Tip:

This function will delete the segment partition table from the waveform however, the waveform coordinates will remain in the destroy and therefore, if you made a mistake and want to restore the memory segments, all you have to do is write the memory partition table again.

Loading Arbitrary Waveforms

The easiest way to download waveforms to the 3156B is with WaveCAD. Using this program you can define, create and download memory segments to the instrument without thinking about low level commands. Regardless, it is understood that not all can use WaveCAD and therefore, the only other way to download waveform coordinates is by using the following function calls sequence:

First, define the work area. Define the segment number and its associated length using the function call:

ri3156B_define_arb_segment ()

or, you can define the entire memory partition table using the following function call:

ri3156B_load_segment_table ()

Next, make a specific segment number is made the active segment. The active segment must be selected because as waveforms are loaded, the 3156B must be notified as to where to place the data it receives. Select the active segment using the following function call:

ri3156B_set_active_segment ()

The next step is to transfer data to the active segment. Use the following function call:

ri3156B_load_arb_data ()

This function call has three variables:

Segment number –	defines the specific segment where you intend to load the data
Data points array –	provides an array of data points, or waveform coordinates that will be sampled by the instrument to generate the required waveform. The data array is constructed from 16-bit words, sent in words and is placed in the buffer is two bytes, high byte first. The order of the bytes can be changed, if required by swapping the high and low bytes

Number of points – which define and must match the number of waveform coordinate in the above array

Using the 12-bit Download mode

Data for the data point array is stored in 16 bit words. In cases where only 12 bit data arrays are available, for example, waveform file that were used previously on other 12-bit waveform generators, the 3156B can be adjusted to accept 12-bit data. Regardless, the data is stored in 16 bit words and therefore, the 12-bit data must be multiplied by 8 so that the MSB bit is shifted 4 places to the left. The data format can be switched by using the following function call:

ri3156B_set_wave_format ()

Reversing Byte Order

Data is placed in the data points array in byte-high byte-low order. This order can be reversed using the following function call

ri3156B_set_byte_order ()

The default is high-byte, low-byte

Changing the Sample Clock Frequency

Users should be careful not to confuse waveform frequency with sample clock frequency. The waveform frequency parameter is valid for standard waveforms only and controls waveform frequency at the output connector. On the other hand, the sample clock frequency parameter is valid for arbitrary and sequenced waveforms only and defines the frequency at which the generator clocks data points.

Standard waveform frequency is measured in units of Hertz (Hz). Arbitrary waveform sample clock frequency is measured in units of Samples per second (S/s). The frequency of a given arbitrary waveform at the output connector must be computed using the sample clock frequency and the number of data points. Use the following equation for computing the frequency of (one cycle of) an arbitrary waveform:

Frequency = Sample Clock / Number of data points

For example, using a sample clock of 80 MS/s with a 1000-point waveform will generate 80kHz waveform at the output connector.

The following function call sets the sample clock frequency for the arbitrary and sequenced waveforms:

ri3156B_set_arb_sampling_freq ()

Sampling clock frequency can normally be programmed throughout the range of 1S/s to 200MS/s however, on one condition, that the number of points in any waveform is a multiple of 2 points. For example, waveform length of 25804 can be used throughout the entire range but if you increase the number of point by 1, it will only work with a special sample clock range that operates from 1S/s to 100MS/s.

Before you decide which of the sample clock ranges you can use, check all of your waveform files. For even number of points, use the normal range, for odd number of points you can use the limited range only. Select the sample frequency range using the following function call:

ri3156B_set_arb_sampling_freq_range ()

Figure 3-7 demonstrates how easy it is, using a graphic interface, to select the sample clock range (1), select waveform format (2) and program the sample clock frequency (3), all are done with a click of a mouse button.

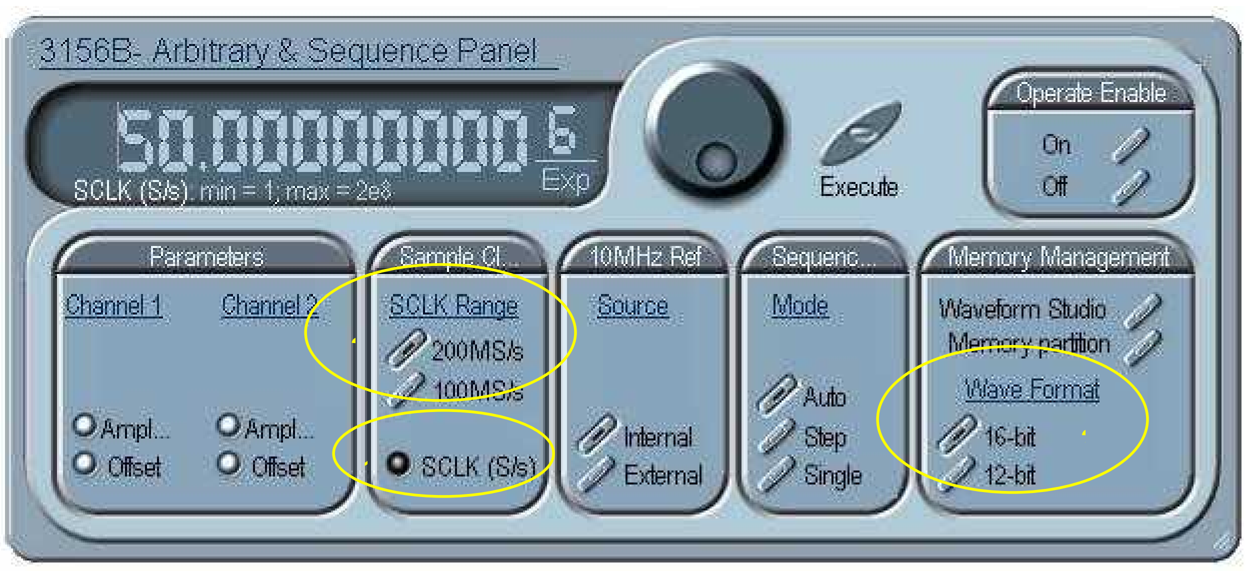


Figure 3-7, WaveCAD Arbitrary and Sequence Panel

Generating Sequenced Waveforms

What Are Sequenced Waveforms?

Sequenced waveforms are made of a number of arbitrary waveforms, which can be linked and repeated in user-programmable order. Sequenced waveforms are generated from waveforms stored in a library of memory segments (sequence table). Before using a sequence of waveforms, load the arbitrary memory with the required waveforms. Information on how to partition the memory and load waveforms is given in the section entitled **Generating Arbitrary Waveforms**.

An example of how sequenced waveforms work is demonstrated in **Figure 1-4**. The sequence generator lets you link and loop segments in user-defined order. **Figure 1-4** shows a sequence of waveforms that were stored in three different memory segments.

Sequence Commands

The following is an overview of how to define and program a sequence of arbitrary waveforms.

A sequence is made of steps. A step can stand on its own or link to another step. It is possible to have only one step in a sequence but the output will look like a continuous waveform. If only one step is specified and the 3156B is placed in Triggered mode, the output will behave as it would in Burst mode where the repeat number replaces the burst count parameter.

Aside from step numbers, each step has two other parameters: segment number and repeat counter. The segment number specifies which segment will be linked, and the repeat counter specifies how many times the segment will loop.

The easiest way to create a sequence table is with WaveCAD. Using this program you can define, create and download memory segments to the instrument without thinking about low level commands, and then use the sequence table to design a sequence. Regardless, it is understood that not all can use WaveCAD and therefore, the only other way to generate such sequence table is by using the following function calls sequence:

Use the following function call repeatedly until you complete the table:

ri3156B_define_sequence_step ()

or, you can define the entire sequence table using the following function call:

ri3156B_load_sequence_table ()

This function call has three variables:

Step number –	defines an index step in the sequence table
Segments array –	provides an array of segments that will be placed in the sequence table, and
Repeats array –	provides an array of repeats that are associated with the segment and index number

Note that the same segment number can be used for different sequence steps. These commands do not change the mode of operations to sequence. The table will affect the 3156B only if you change the waveform function to sequenced.

The sequence generator goes through its steps in descending order. In continuous run mode, the sequence is repeated automatically after the last step has been completed. When the generator is set to operate in Triggered mode, the output stops at the last point of the last waveform in the sequence. In Gated mode, the sequence is always completed after the gate stop signal.

If a mistake is made and removal of one or more steps from the sequence table is needed, use the following command:

ri3156B_delete_sequence_step (n)

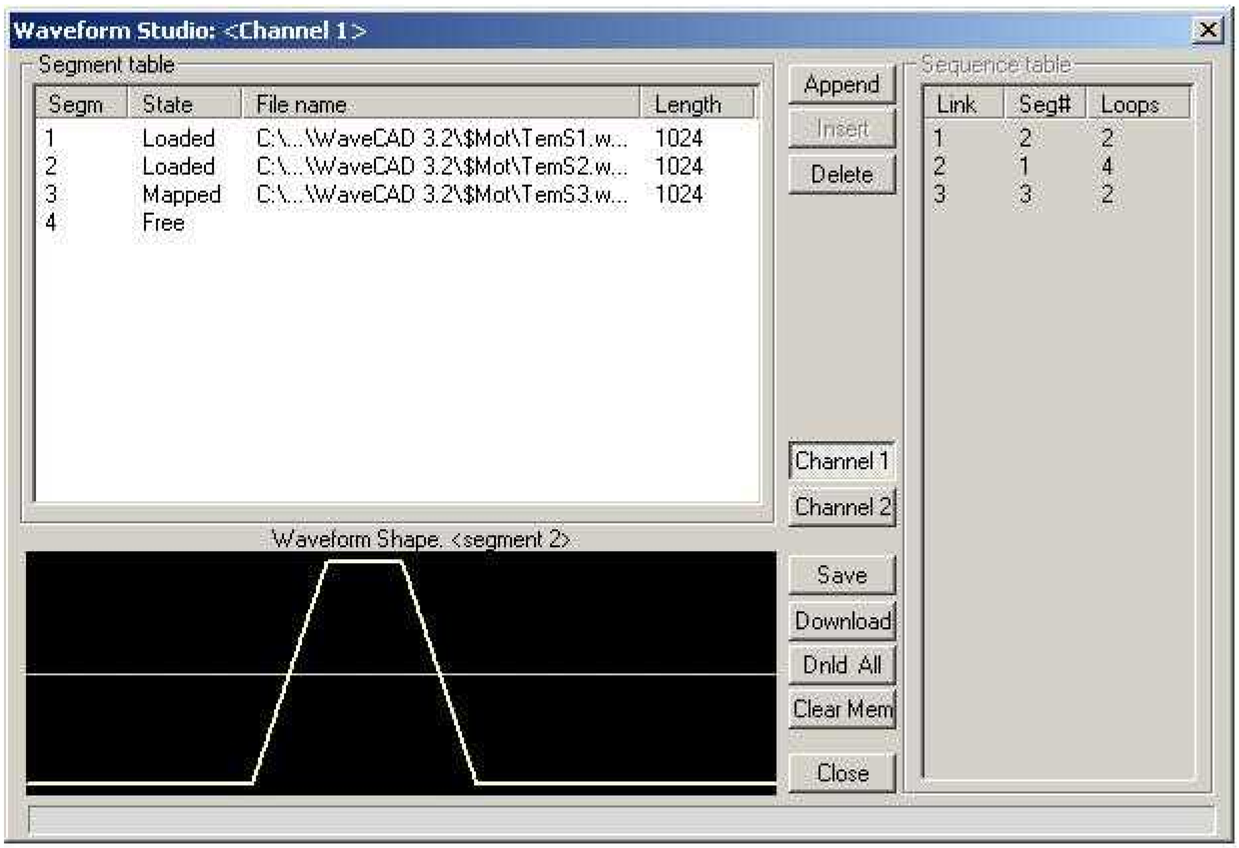
where <n> is the step number to be removed from table. To delete the entire sequence table and start from fresh use the following function call:

ri3156B_delete_sequence_step (0)

where the variable <0> defines a delete all segments command

CAUTION:

The above command is destructive to the sequence table. Once executed, there is no restore procedure.



Segm	State	File name	Length
1	Loaded	C:\...\WaveCAD 3.2\$Mot\TemS1.w...	1024
2	Loaded	C:\...\WaveCAD 3.2\$Mot\TemS2.w...	1024
3	Mapped	C:\...\WaveCAD 3.2\$Mot\TemS3.w...	1024
4	Free		

Link	Seg#	Loops
1	2	2
2	1	4
3	3	2

Figure 3-8, WaveCAD example of a Segment and Sequence Tables

## Controlling the Sequence Advance

The way the instrument advances through the sequence links and the source of the event causing sequence advance can be specified by the user. Use the following command to control how the sequence advances through the sequence table steps:

ri3156B_set_sequence_mode ()

There are three advance options: Auto, Stepped and Single. These modes are described below.

AUTO specifies continuous advance where the generator steps continuously to the end of the sequence table and repeats the sequence from the start. For example, if a sequence is made of three segments – 1, 2, and 3, and AUTO mode is used, the sequence will generate an infinite number of 1,2,3,1,2,3,1,2,3…waveforms. Segments will loop depending on the loop counter which is specified in the sequence table.

In **STEP** advance mode, the sequence is advanced to the next waveform only when a valid trigger is received. The output of the 3156B generates the first segment continuously until a trigger signal advances the sequence to the next segment. If repeats were specified in the sequence table, they are ignored in STEP advance mode.

In **SINGLE** advance mode, the 3156B idles between steps until a valid trigger signal is sensed. This advance mode is available when the 3156B is in triggered run mode only. An attempt to select the SINGLE advance mode when the 3156B is in continuous run mode will generate a setting conflict error. After a trigger, the generator outputs one (or more) waveform cycle(s). Then, the output level idles at a DC level equal to the last point of the last generated waveform. If loops (repeats) were programmed, the segment is repeated n times automatically before it begins idling. After executing all of the programmed loops the sequencer will step to the next segment in the sequence when it receives its next valid trigger.

Advance source can be specified using the following function call:

ri3156B_set_carrier_run_mode ()

Note that some run modes collide may sequence operation and cause setting conflict errors; These conditions are summarized in **Table 3-1**. In general, all trigger advance sources are available for advancing the sequence. You can use the software enable, front-panel TRIG IN connector, or the backplane TTLTrg0-7 lines.

Using the Modulated Waveforms

The 3156B can be programmed to produce various modulated waveforms such as: FM, AM, FSK, Sweep and frequency hops. Note that when one channel is programmed to generate modulation, the other channel can be programmed to either generate the same modulation type or produce AC signal only. The legal conditions that are legal for both channels if one is producing modulation are summarized in **Table 3-1**.

To first step to place the instrument in modulation mode. Use one of the following function calls:

1. The *ri3156b_set_global_operating_mode ()* – use this function call if each channels needs to be programmed with its own functionality
2. The *ri3156b_set_common_waveform_mode ()* – the best function call to use if both channels need to be programmed with the same functionality.
3. The *ri3156b_set_common_waveform_mode ()* - allows separate programming of each channel to a different waveform mode. From all functions, this is the least recommended function to be used as it may cause setting conflict errors should one not fully understand the limitation of the product.

More information on these function calls is given in the Selecting Output Type section of this chapter.

The next step after placing the generator in modulation is to select the modulation type. Use the following function call:

ri3156B_set_modulation_mode ()

There are 6 argument options available:

0 – places the 3156B in modulation OFF mode where the output is connected to the modulation source but the modulation is not selected or inactive yet. In this case, the output generates carrier frequency with the following properties: waveform is sinewave and frequency is set by the carrier frequency parameter. Other parameters such as amplitude and offset affect this function as well.

1 – selects FM. Depending on the modulation run mode, the output idles on carrier frequency until a valid enable signal is applied to the generator. The following FM parameters are available for modification: Carrier frequency, Modulation frequency, Modulating waveform shape, Deviation frequency and Marker position. The FM function calls are described in Chapter 4.

2 – selects AM. Depending on the modulation run mode, the output idles on carrier frequency until a valid enable signal is applied to the generator. The following AM parameters are available for modification: Carrier frequency, Modulation frequency and Modulation depth. The AM function calls are described in Chapter 4.

3 – selects FSK. Depending on the modulation run mode, the output idles on carrier frequency until a valid enable signal is applied to the generator then, hops from carrier to shifted frequency according to a sequence listed in the FSK table. The following FSK parameters are available for modification: Carrier frequency, Shifted frequency, Shift rate and Marker position. The FSK table is programmable as well. The FSK function calls are described in Chapter 4.

4 – selects Sweep. Depending on the modulation run mode, the output idles on carrier frequency until a valid enable signal is applied to the generator. The following Sweep parameters are available for modification: Start and Stop frequency range, Sweep type and direction, Sweep time and marker frequency. The Sweep function calls are described in Chapter 4.

5 – selects Frequency hops. Depending on the modulation run mode, the output idles on last hop frequency until a valid enable signal is applied to the generator then the frequency hops from frequency to frequency at a rate determined by the baud rate and sequenced in the hop table. The following Frequency hops parameters are available for modification: Hop mode, dwell time and marker position. Hop table is programmable as well. The Frequency hops function calls are described in Chapter 4.

Hints:

1. **The carrier waveform in modulation is always sinewave**
2. **Modulation run modes are separate and different than the non-modulated run modes**
3. **An operate enable command initiates modulation however, once the output on command is used, carrier is immediately available at the output**
4. **Following completion of trig or burst cycles, the output resumes carrier frequency**

As was hinted above, the modulation run modes are different than the run modes for the non-modulated waveforms. The difference is in the way the output generates the modulated waveforms. Use the following function call to program the modulation run mode:

ri3156B_set_modulation_run_mode ()

The following options are available as modulation run modes:

Continuous –	sine waveform idles on carrier frequency, modulation starts after an enable command and continues indefinitely till terminated
Triggered –	sine waveform idles on carrier frequency. Each trigger generates a single modulation cycle. The trigger source is selected using the ri3156B_set_trigger_source() function
Gated –	sine waveform idles on carrier frequency. An external signal enables modulation. First output cycle is synchronous with the active slope of the trigger signal. Last cycle of modulated waveform is always completed. The trigger source is selected using the ri3156B_set_trigger_source() function
Burst –	sine waveform idles on carrier frequency. Each trigger generates a single burst of modulation cycles. The trigger source is selected using the ri3156B_set_trigger_source() function

The WaveCAD MOD1 and MOD2 panels can be used for selecting the modulation type and modes. Note however, that WaveCAD is using the function call *ri3156b_set_common_waveform_mode ()* to set up both channel simultaneously so if you need to set up the 3156B to output different modulation types from each channel, the only way to use WaveCAD is from its function call editor, as explained in Chapter 4. **Figure 3-9** shows an example of high level programming using the Modulation Panel 1.

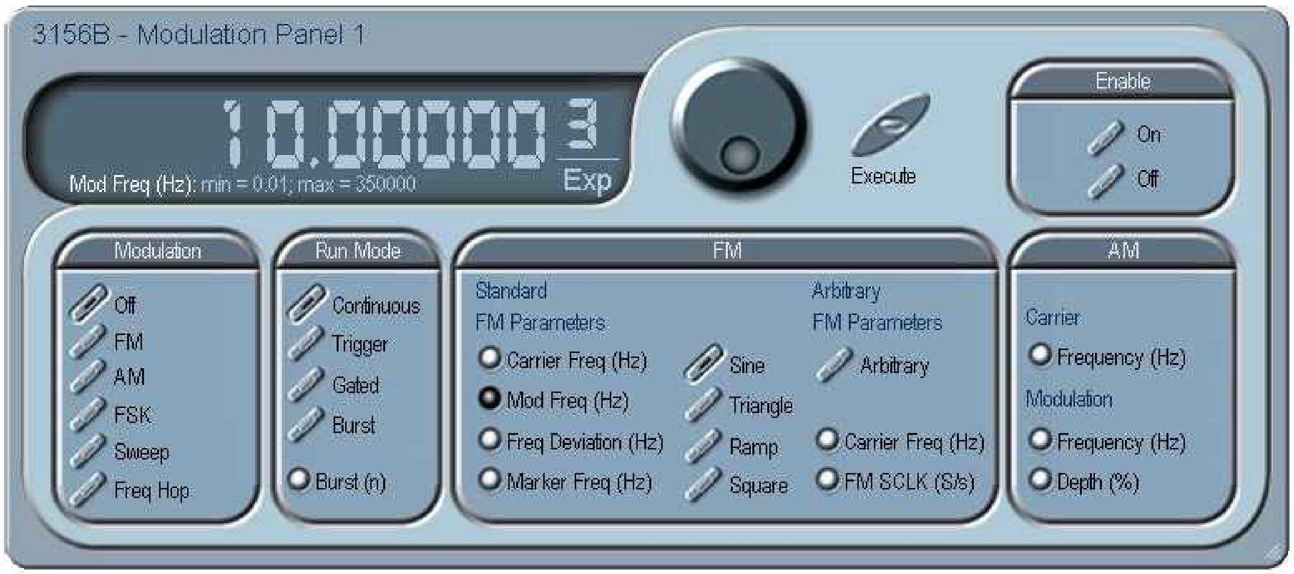


Figure 3-9, WaveCAD Example, modulation Programming

Using the Digital Outputs

The 3156B has two special connectors of which can be used to generate digital patterns. Each channel generates different patterns. The inter-channel dependencies and run mode rules apply to the digital outputs as well. The conditions that are legal for both channels if one is producing modulation are summarized in **Table 3-1**.

To first step to place the instrument in digital mode. Use one of the following function calls:

1. The *ri3156b_set_global_operating_mode ()* – use this function call if each channels needs to be programmed with its own functionality.

2. The *ri3156b_set_common_waveform_mode ()* – the best function call to use if both channels need to be programmed with the same functionality.

3. The *ri3156b_set_common_waveform_mode ()* - allows separate programming of each channel to a different waveform mode. From all functions, this is the least recommended function to be used as it may cause setting conflict errors should one not fully understand the limitation of the product.

More information on these function calls is given in the Selecting Output Type section of this chapter.

The next step after placing the generator in digital mode is to select the output type. Use the following function call:

ri3156B_set_digital_mode ()

There are 2 options available: Freerun and Stimulus. The difference between the two types is as follows:

Freerun patterns can be programmed with a programmable hold count for each pattern. The following parameters are available for programming the Freerun patterns: Rate range, Step rate and hold count. Freerun patterns are stored and executed from pattern tables.

Stimulus patterns steps through the patterns at a rate defined by the sample clock frequency. The following parameters are available for programming the Freerun patterns: Rate range and Step rate. Stimulus patterns are stored and executed from pattern tables.

Run modes apply to the digital patterns the same way they are being applied to the standard and arbitrary waveforms.

As was hinted above, the digital patterns run modes are the same as for standard waveforms. Use the following function call to program the modulation run mode:

ri3156B_set_carrier_run_mode ()

The following options are available as modulation run modes:

Continuous –	following an enable command, patterns are generated continuously and indefinitely till terminated
Triggered –	each trigger generates a single sequence of patterns. The trigger source is selected using the ri3156B_set_trigger_source() function
Gated –	an external signal enables patterns. First output cycle is synchronous with the active slope of the trigger signal. Last cycle of patterns is always completed. The trigger source is selected using the ri3156B_set_trigger_source() function

Burst –	each trigger generates a single burst of patterns cycles. The trigger source is selected using the ri3156B_set_trigger_source() function

The WaveCAD DIG panel can be used for selecting the digital patterns type and modes. Note however, that WaveCAD is using the function call *ri3156b_set_common_waveform_mode ()* to set up both channel simultaneously so if you need to set up the 3156B to output different waveform types from each channel, the only way to use WaveCAD is from its function call editor, as explained in Chapter 4. **Figure 3-10** shows an example of high level programming using the Digital and Video Panel.

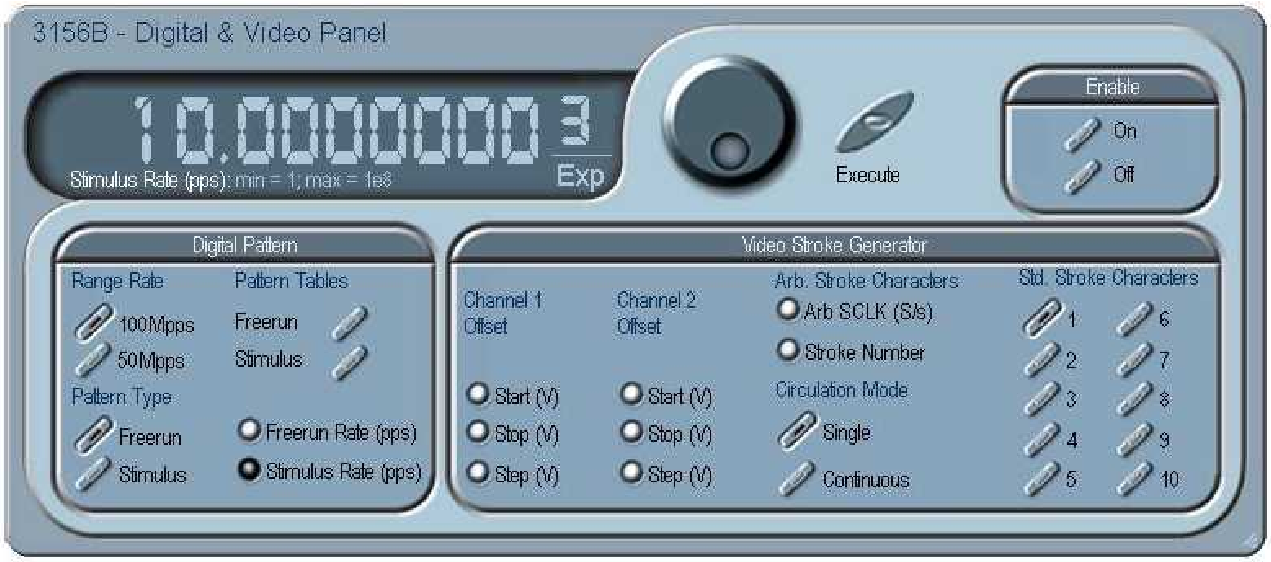


Figure 3-10, WaveCAD Example of Digital Programming

Using the Video Stroke Generator

The 3156B can generate two types of video stroke characters: Standard characters, from a built-in library, or user characters that can be loaded to the generator the same way as arbitrary waveforms are loaded. The inter-channel dependencies and run mode rules apply to the video stroke generator as well. The legal conditions that are available for both channels if one is producing video characters are summarized in **Table 3-1**.

To first step to place the instrument in video mode. Since video requires that both channels generate characters, the only function call to use for placing the 3156B in video stroke mode is the following:

ri3156b_set_common_waveform_mode ()

The next step after placing the generator in video mode is to select the character type. If you want to use one of the 10 standard characters that are built in the 3156B, use the following function call:

ri3156B_set_video_character ()

There are 10 options available: Cross locator, Cross hair, Positioned square, Vertical marker line, Horizontal marker line, Right hand arrow, Left hand arrow, Diamond overlay, Inverted triangle and upright triangle. The following parameters are available for video stroke programming: Point-stroke frequency, Offset start and stop range, Offset step and circulation mode.

If you do not intend to use the built in characters and want to download to the 3156B memory your own characters, you will be using the instrument in a similar way as you would be using the arbitrary function, hence downloading character data to segments in the arbitrary memory. The functions that are available for user video stroke characters are summarized in Chapter 4.

Use the following function call to program the video stroke circulation type:

ri3156b_set(query)_video_stroke_circ_type ()

There are two circulation modes available for video stroke: Single and Continuous.

Single – require trigger signals to move the video stroke character about the screen from the start point which is determined by the offset start value to the end point which is determined by the offset stop value. The rate of movement is determined by the trigger rate and the increment of the movement is determined by the offset step value. The character will end up at the end point and will idle there until the function is re-selected.

Continuous – require trigger signals to move the video stroke character about the screen from the start point which is determined by the offset start value to the end point which is determined by the offset stop value. The rate of movement is determined by the trigger rate and the increment of the movement is determined by the offset step value then the character jumps to the start point. The sequence is repeated continuously as long as triggers are applied to the instrument.

The WaveCAD DIG panel can be used for selecting the video stroke type and characters. **Figure 3-10** shows an example of high level programming using the Digital and Video Panel.

Selecting Run Modes

The 3156B offers four run modes: Continuous, Triggered, Gated and Burst. The selected waveform is repeated continuously when the instrument is set to operate in continuous mode and an operate enable command has been issued from the proper source.

Note:

Regardless of the selected run mode, the 3156B will not start generating waveforms before an output enable command is issued. The enable command can be software, hardware or mixed. Information on the various enable options is given in the section entitled Enabling Output Waveforms in this chapter.

Triggered, Gated, and Burst modes require a single or multiple enable (trigger) signals to initiate output cycles. Information on how to trigger, gate or output a burst of waveforms is given in the following paragraphs. The WaveCAD Trigger control panel in **Figure 3-11** shows a high-level example of selecting the trigger mode. Description of the various run modes is given below.

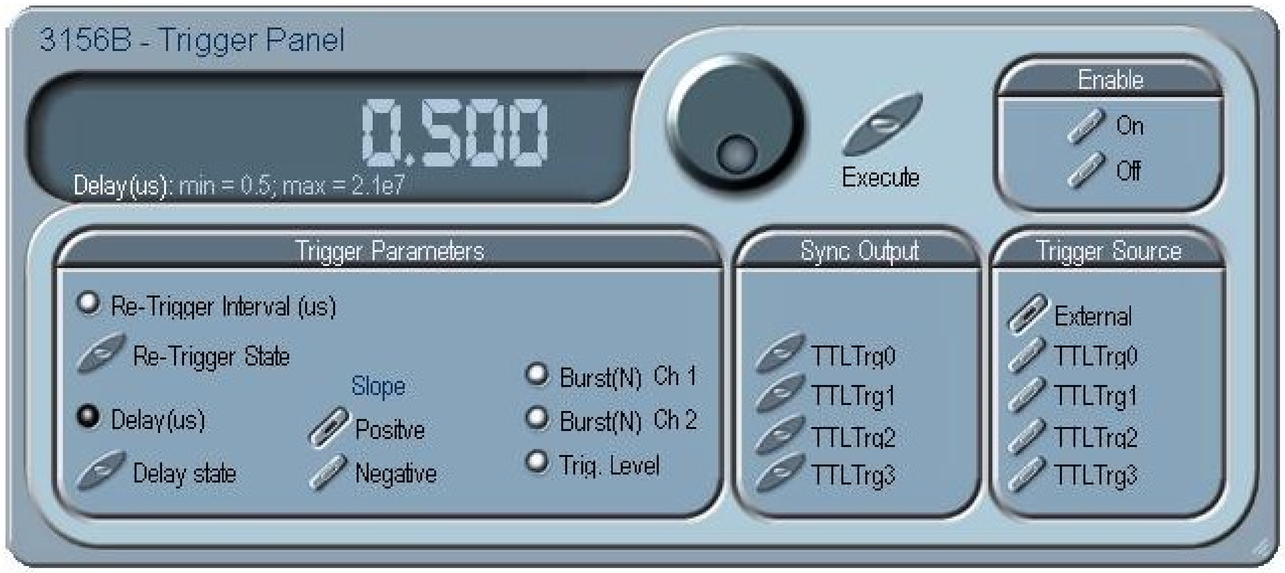


Figure 3-11, WaveCAD Example of Trigger Control

Continuous Run Mode

The following function calls need to be executed to generate continuous waveforms:

ri3156Bset_output (1)	Turns output on by activating hardware connection to the front panel

ri3156Bset_enable (1)	Output is enabled. This is a software enable command. If you intend to use hardware enable you have to select the hardware enable source with the ri3156B_set_enable_source () function

Note that the above sequence is valid for non-modulated waveforms only. Also, the continuous run mode is the default run mode and therefore, after reset or power up, there is no need to specifically select this run mode.

Triggered Run Mode

In Triggered run mode, the output remains at a certain DC level until the operate enable (or trigger) signal initiates a single output cycle. If hardware source is selected for the enable command, the edge sensitivity can be program for either the rising or the falling edge of the input signal. Each time a transition at the trigger input occurs, the 3156B generates one complete output waveform. At the end of the output cycle, the output resumes position at a DC level that is equal to the amplitude of the last point of the waveform.

To place the 3156B in Triggered mode, use the following function call:

ri3156B_set(query)_carrier_run_mode (1)

Variable number 1 selects the trigger run mode.

The 3156B can also be programmed to re-trigger itself, after a valid trigger command, to run continuously in triggered mode with a delay between triggers, which is set with the re-trigger delay parameter. Use the following function call to place the instrument in re-trigger mode:

ri3156B_set_re_trigger_delay_state (1)

The variable 1 turns the re-trigger delay state on. Use the following function call to program the re-trigger delay interval:

ri3156B_set_re_trigger_delay ()

The delay is programmable in units of μs from 0.5μs to 21s.

Gated Run Mode

In Gated run mode, the output remains at a certain DC level until the operate enable (or trigger) signal opens the gate. A subsequent operate enable command closes the gate. Only hardware signal can open and close the gate. Edge sensitivity of the trigger signal can be program for either the rising or the falling edge of the signal. At the end of the last output cycle, the output resumes position at a DC level that is equal to the amplitude of the last point of the waveform.

To place the 3156B in Gated mode, use the following function call:

ri3156B_set(query)_carrier_run_mode (2)

Variable number 2 selects the gated run mode.

Burst Run Mode

Burst mode is very similar to Triggered mode with the exception that only one trigger signal is needed to generate a counted number of output waveforms. In Burst mode, the output remains at a certain DC level until the operate enable (or trigger) signal initiates a single output cycle. If hardware source is selected for the enable command, the edge sensitivity can be program for either the rising or the falling edge of the input signal. Each time a transition at the trigger input occurs, the 3156B generates a counted burst of output waveforms. At the end of the burst, the output resumes position at a DC level that is equal to the amplitude of the last point of the waveform. The burst counter is programmable from 1 to 1Meg.

The 3156B can also be programmed to re-trigger itself, after a valid trigger command, to run continuously in triggered burst mode with a delay between bursts, which is set with the re-trigger delay parameter. Use the following function call to place the instrument in re-trigger mode:

ri3156B_set_re_trigger_delay_state (1)

The variable 1 turns the re-trigger delay state on. Use the following function call to program the re-trigger delay interval:

ri3156B_set_re_trigger_delay ()

The delay is programmable in units of μs from 0.5μs to 21s.

Selecting the Trigger Slope

The trigger slope command toggles edge sensitivity for the selected trigger input. The 3156B can be made sensitive to either positive or negative transitions. The inputs that will be affected by this command are: Front-panel TRIG IN connector, TTLTrg lines 0 through 7 and ECLTrg line 0. Use the following function call to select slope sensitivity for the trigger signal:

ri3156B_set(query)_trigger_slope ()

Positive going transitions trigger the 3156B when variable 0 is selected. Negative transitions trigger the 3156B when variable 1 is selected. Positive is the default slope sensitivity.

Selecting the Trigger Level

The trigger level command sets the threshold level at the trigger input connector only. Trigger level is adjustable from -5V to +5V. Use the following function call to set the trigger level for the trigger signal:

ri3156B_set(query)_trigger_level ()

This sets the trigger level for the signal, which is applied at the TRIG IN connector. The default value is 1.6 V so there is no need to modify this value if you apply a TTL level signal to the trigger input.

Using the Trigger Delay

The trigger delay value designates the time that will lapse from a valid trigger signal to the first transition of an output waveform at the output connector. The delay is programmable in units of μs from 0.5μs to 21s. Use the following function call to turn on trigger delay:

ri3156B_set(query)_trigger_delay_state (1)

The variable 1 turns the trigger delay state on. Use the following function call to program the trigger delay interval:

ri3156B_set_trigger_delay ()

Activating the Backplane TTLTrg Lines

The 3156B can receive triggers on the backplane TTLTrg 0 through 7 and ECLTrg0 lines. The instrument can also be made outputs and transmit the Sync signal on the same lines, except not on the ECLTrg0 line. Use the following function call to generate triggers on the backplane trigger lines:

ri3156B set_TTLTRG_n_output_state (1)

<n> designates the required TTLTrg line and can take values from 0 to 7. When a specific backplane trigger line is programmed to output the sync signal, make sure that no other module on the bus is programmed to generate trigger signals on the same lines at the same time. If you want to make any of the TTLTrg lines an input, make sure its output state is turned off and select the required line using the following function call:

ri3156B_set(query)_trigger_source ()

Variable options 1 through 8 select TTLTrg lines 0 through 7. The variable 9 selects the ECLTrg0 as an input source for the trigger signal.

Chapter 4
WaveCAD

What's in This Chapter

This Chapter contains information how to install, invoke and use WaveCAD. Introduction to WaveCAD and examples how to program instrument controls and parameters and how to generate waveforms and download them to the 3156B are also given in the following sections.

Introduction to WaveCAD

In general, WaveCAD is a utility program that serves as an aid for programming the 3156B. WaveCAD has many functions and features of which all of them share a common purpose – controlling 3156B functions from remote. As minimum, to use WaveCAD, you'll need the following tools:

1. Computer, Pentium III or better
2. Windows 2000/XP, or higher
3. High resolution screen, at least, 1024 x 768 pixels
4. Pointing device, mouse or ball
5. Visa 2.6, or higher installation
6. Last, but not least, some basic knowledge how to operate computers and Windows-based programs.

WaveCAD operation is divided into three main functions: 1) Front panel control, 2) Waveform generation and editing and 3) FM waveform generation and editing. These operating options are described in this chapter however, you must install WaveCAD before you can use it. The next paragraphs describe installation and first steps before going into in-depth operation.

Installing WaveCAD

The installation program installs WaveCAD on a logical drive of your choice. The default is drive C. It automatically creates a new directory and copies the files that are required to run the program. Before you install WaveCAD, make sure that there is at least 5 megabytes of available memory on your hard disk drive.

To install WaveCAD, insert the distribution disk in the A: drive.

Invoke Run and type:

```
A:\Setup
```

The install program does the complete job far you and creates a workgroup and icons to start WaveCAD.

Quitting WaveCAD

Before you start roaming through menus and editing commands, we strongly recommend that you make yourself familiar with WaveCAD basics and concept. For now quit the program and spend some more time with this section of the manual. Point the mouse cursor to the File menu and press the left mouse button. Move the mouse cursor to the Exit command and press the left mouse button.

For the New and Advanced Users

For the New User
Learning to use WaveCAD is easy, intuitive and quick, even if you have never used such programs before. After you have installed WaveCAD on your computer read the following paragraphs to learn how to find your way around WaveCAD's menus.

Once you are familiar with the basics, you'll continue to learn about features, programming, and editing commands. If you can't find the answer to a question in this guide, call your distributor or the LeCroy customer support service near you and we'll gladly assist you with your problems.

For the Advanced User
If you are already familiar with computer conventions and have basic knowledge of Windows programming, you may want to skip some of the following paragraphs.

Conventions Used in This Manual

This manual uses certain typographical conventions to make it easier for you to follow instructions. These conventions are described in the following:

[Enter, or ↵] Press the Enter or Return key.

[Esc] Press the Escape key.

[Alt-F] Press the Alt key and the key that follows, simultaneously. In this example the key that follows is F.

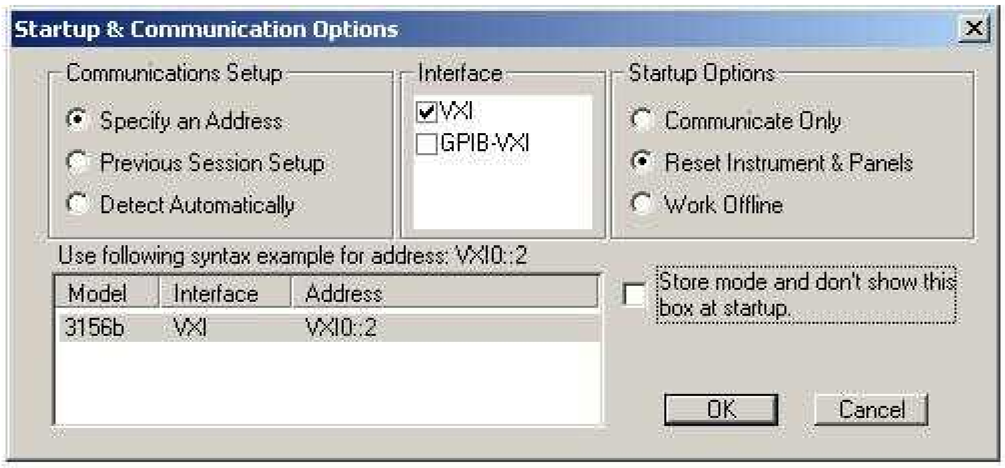
[Ctrl-S] Press the Control key and the letter that follows, simultaneously. In this example, the letter is S. The control key also appears in the menus as a target sign.

[↑] [↓] [→] [←] Press the Arrow key with the symbol pointing in the direction specified (i.e., up, down, left, or right).

<+> Press the key for the character or word enclosed in angle brackets. In this case, the Plus sign key.

The Opening Screen

Invoke WaveCAD by double clicking on the icon. If you cannot find the icon on your desktop, click on Start, Programs and WaveCAD. The opening screen will show. If you installed the program correctly, your screen should look as shown in **Figure 4-1**.



Startup & Communication Options

Communications Setup
[✓ Specify an Address]
Previous Session Setup
Detect Automatically

Interface
[✓ VXI]
GPIB-VXI

Startup Options
Communicate Only
[✓ Reset Instrument & Panels]
Work Offline

Use following syntax example for address: VXI0::2

Model	Interface	Address
3156b	VXI	VXI0::2

Store mode and don't show this box at startup.

Figure 4-1, Startup & Communication Options

The Startup & Communication Options dialog box is displayed. You can check the "Store and don't show…" so next time you invoke WaveCAD, this dialog box will not be displayed. The purpose of this dialog box is to update the program in the way you intend to use it. For example, if you are using the VXI module address 2, then you, then you can click on the Specify an Address option and type in the required address so the next time you use WaveCAD, the program will automatically resume communication with the same address as was originally detected.

If you chose to hide this dialog box, you can still access and change the options from the System command, at the top of the screen.

Make your selection and click OK. The Startup & Communication Updater dialog box will be removed from the screen. And two panels: the Main and the standard Waveform panel will now be accessible. But before we go into panel operation, let's look at the toolbars at the left top of the screen as shown in **Figure 5-2**.

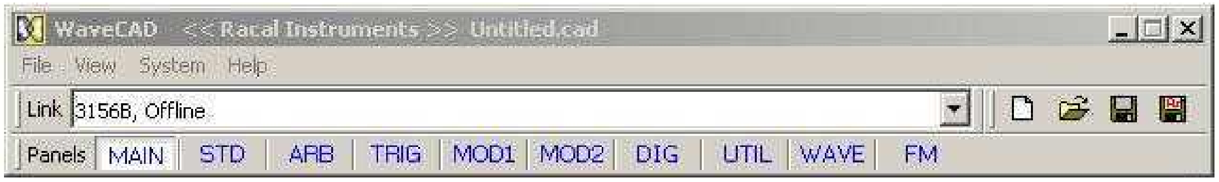

WaveCAD << Racal Instruments >> Untitled.cad
File View System Help
Link 3156B, Offline
Panels MAIN STD ARB TRIG MOD1 MOD2 DIG UTIL WAVE FM

Figure 4-2, WaveCAD's Toolbars

The standard Windows **Menu Bar** is the top bar. It provides access to main system controls like saving files, and viewing or removal of screen images.

The second bar is called **Link** bar. It provides direct access to different instruments that are active on the VXI bus. WaveCAD can control a number of 3156B units simultaneously. If the instruments were connected to the VXIbus system while invoking WaveCAD, they will automatically be detected by the program and will be placed in the Link pull-down window. The active instrument is displayed with its associated address. If you run WaveCAD in offline mode, the Link bar will show 3156B, Offline.

The **Panels** bar provides direct access to instrument control panels. The individual control panels are explained later in this chapter. The MAIN, STD, ARB, TRIG, MOD1 and the other buttons will bring up to the screen panels that are associated with these names. The WAVE and FM buttons will open the waveform and FM waveform composers. The first time you launch WaveCAD, the opening screen will have the Main panel open. Click on other buttons and interactively get the feel how WaveCAD opens and closes control panels.

WaveCAD Features

WaveCAD’s main purpose is controlling 3156B functions and parameters. The 3156B can generate standard waveforms from a built-in library, arbitrary waveforms from user-downloaded coordinates, modulated waveforms, digital patterns and much more. The only way to access all of these features is through software utilities such as Plug & Play drivers, and soft front panels. WaveCAD is built to provide complete control over the 3156B.

WaveCAD has three main screens: 1) Control panels, 2) Waveform composer and 3) FM composer. The various screens along with instructions how to access and use them are described below in detail.

The Control Panels

The control panels look and feel just as if you would operate an instrument from its front panel. They even look like a front panel of an instrument, so operating function and changing parameters are very intuitive. Let’s look at the first panel that shows at the opening screen. This panel, as shown in **Figure 4-3**, is called the Main Panel.

To begin with, let’s explore the panel controls to see how they feel, react and what they do. All other panels share almost the same feel, so the description of how to operate the Main Panel can serve as general guide for controlling the rest of the panels.

Looking at the panel you can identify the following controls: Push buttons, LED’s, radio buttons, Dial and Digital display. The function of each control is described below.

Push Buttons – These are used for toggling a function on and off. For example, the Output Enable button in the Output group toggles the output on and off. The first mouse click will push the button inwards and will turn on a red bar at the center of the button, indicating that the function is on. The second mouse click will turn the function off.

Radio Buttons – Are used for changing operating modes, or selecting between mode options. One of the radio buttons is always on with a red dot in its center, indicating its state condition.

LED’s – The LED’s indicate which of the parameters are displayed on the Digital Display. Red LED indicates that the parameter name next to this LED is selected. Only one LED can be ON at a time.

HINT:

LED’s are turned on by clicking on the LED or the text next to it. The selected parameter is flagged by a darker LED shade.

Dial – Use the dial to modify displayed reading. To use the dial, press and hold the mouse cursor on the dial and move the mouse in a clockwise circle to increase the number, or counterclockwise circle to decrease the displayed number. The dial modifies digits at the cursor position and will allow modification within the legal range of the displayed parameter. If you reach the end of the range, the dial will have no further effect on the display. If you do not want to use the dial, you can still change the display reading by using the [↑], or [↓] keys, or simply type the required number using the standard keyboard features.

NOTE:

After you change the displayed readout, the 3156B will be updated with the new parameter only after you press the Execute button.

Digital Display – The display is used for displaying and reading various 3156B parameters, just as you would use it on your instrument.

The Main Panel

The Main Panel, as shown in **Figure 4-3**, is the first panel you see after invoking WaveCAD. Notice how buttons and LED's are grouped; this is done specifically so that common parameters are placed in functional groups. The Main Panel groups allow (from left to right) adjustment of amplitude and offset, selection of waveform mode, selection of run mode and control over SYNC and Main output parameters. Controls, where applicable, are provided for each channel separately however, on the main panel, you must select the active channel from the Program group before programming its parameters.

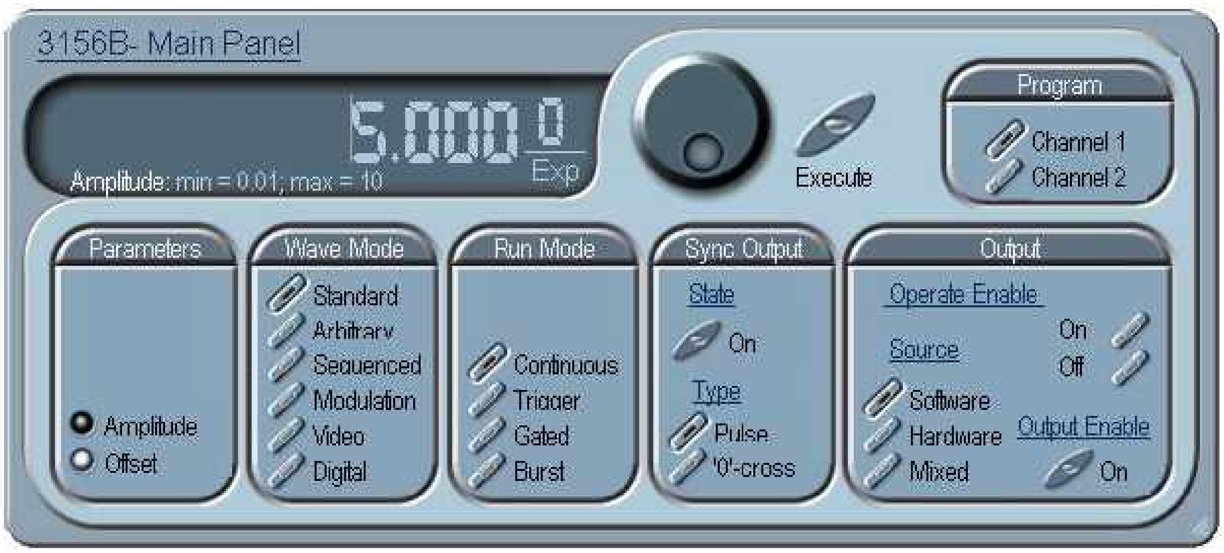


Figure 4-3, The Main Panel

If you are connected properly to a VXI chassis and WaveCAD has detected your instrument, then every time you press a button, you are getting an immediate action on the 3156B. It is different if you are changing parameters on the display; Doing this, you'll have to press the Execute button for the command to update the instrument.

The functional groups in the Main Panel are explained below.

Program

The Program group identifies which of the channels is being programmed. Except the Wave Mode and the Run Mode groups, all other parameters are channel-dependent and therefore, before programming parameters and output states, make sure the correct channel setting is enabled.

Parameters

The Parameters group has two parameters: Amplitude and Offset. To access the required parameter, click on the LED or the text next to it to display the required parameter. The value that is associated with the lit LED is displayed on the digital display. You can use the dial, keyboard, or the [↑] [↓} keys to adjust the readout to the required setting. After you modify the reading, press Execute to update the 3156B with the new reading.

Note:

Normal color of the digital reading is dark blue. If you modify the reading, the color changes to a lighter shade of blue, indicating that the 3156B has not been updated yet with the new parameter. Pressing Execute will update the instrument and will restore the color of the digital readout to dark blue, indicating that the displayed value is the same as the generator setting.

Also note that the digital readout has an autodetect mechanism for the high and low limits. You cannot exceed the limits if you are using the dial but only if you use the keypad. In case you do, the program will not let you download an illegal parameter and you'll be requested to correct your setting.

Wave Mode

The Wave Mode group is used for selecting between waveform types. The 3156B provides six types of waveforms: Standard, Arbitrary, Sequenced, Modulated, Video and Digital. By pressing one of these buttons output waveform will change to the selected option. The default waveform type is Standard. If you want to change standard waveform parameters, you can select this panel from the Panels bar.

Run Mode

The Run Mode group is used for selecting the active run mode for the instrument. You can select between continuous, triggered, gated and burst modes. There is no additional panel associated with the continuous mode, but if you press one of the other run mode options, you'll be able to adjust the trigger parameters from the Trigger Panel.

SYNC Output

The SYNC Output group controls SYNC type and state. The SYNC output is enabled when the State button is ON. The SYNC Type toggles between Pulse and "0" Cross. In Sequenced wave mode, the SYNC output automatically reverts to LCOM (loop Complete) output.

Output

The Output group is used for controlling the state of the 3156B channel 1 and 2 outputs and allows selection of the enable source. It also provides access to software enable commands. The default Operate Enable source is Software and therefore, if the output state is on, clicking on the operate Enable button will stimulate the output.

TIP:

The 3156B requires two conditions to start generating waveforms: 1) Output State is ON and 2) Operate Enable is ON. The Output State button activates a mechanical relay that connects/disconnects the output terminal to the output circuit. The Operate Enable ON command stimulates the 3156B to generate signals.

Enable

The enable group affects the 3156B only when the operate enable source is set to software (software is the default enable source). If the output state is on, clicking on the operate Enable button will stimulate the output. In some cases, the disable can stop the output. The various options to enable and disable the output waveforms, using the enable commands are given in **Table 3-2**.

The Standard Waveforms Panel

The Standard Waveforms panel, as shown in **Figure 4-4**, is accessible after you click on the STD button in the Panels bar. The functional groups in the Standard Waveforms Panel are described below.

Waveforms (Channel 1 and 2)

The Waveforms group provides access to a library of built-in standard waveforms. The library includes: Sine, Triangle, Square, Pulse Ramp, Sinc, Exponential, Gaussian and DC waveforms. Each waveform has one or more parameters that can be adjusted for the required characteristics of the output. For example, phase start can be adjusted for the sine and triangle waveforms and duty-cycle can be adjusted for the square waveform. The pulse waveform can be adjusted for rise and fall time as well as width and delay. Parameters that are associated with each waveform are automatically displayed when the waveform is selected.

When the Half Cycle button is depressed, the Sine, Triangle and Square waveforms produce half cycles each time an enable command is received. In continuous run mode, the time lapse between two half-cycles is adjustable using the re-trigger delay parameter.

Note that by clicking a button in this group, you are immediately updating the 3156B output with this waveform shape.

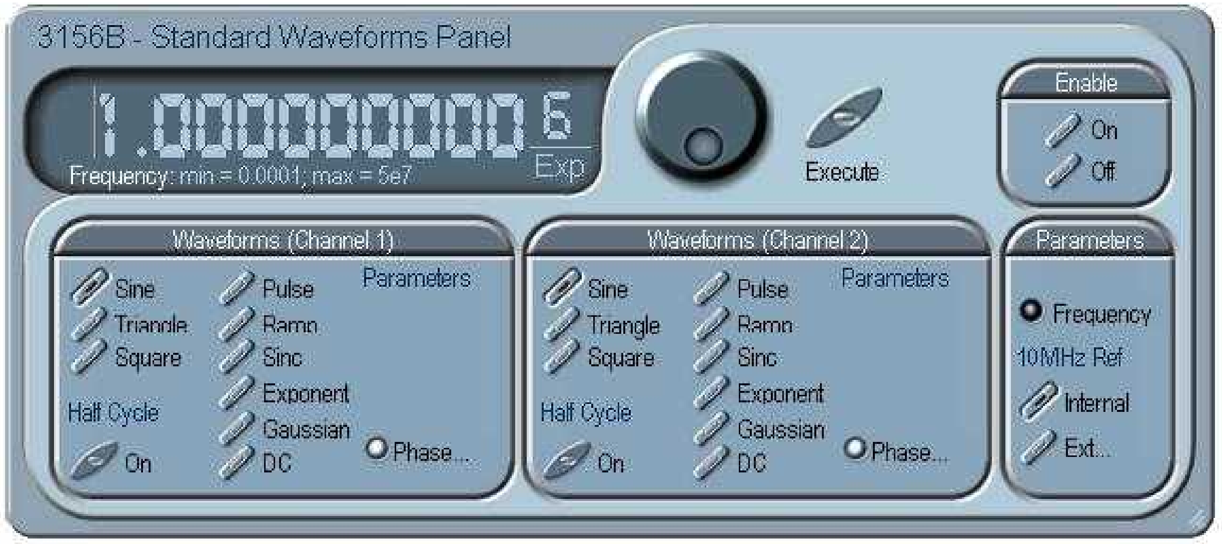


Figure 4-4, The Standard Waveforms Panel

Parameters
The Frequency control lets you program the output frequency of the selected waveform shape. The frequency parameter may be modified when the LED illuminates. You can use the dial, keyboard, or the [↑] [↓} keys to adjust the readout to the required setting. After you modify the reading, press Execute to update the 3156B with the new reading.

Note:

Normal color of the digital reading is dark blue. If you modify the reading, the color changes to a lighter shade of blue, indicating that the 3156B has not been updated yet with the new parameter. Pressing Execute will update the instrument and will restore the color of the digital readout to dark blue, indicating that the displayed value is the same as the generator setting.

Also note that the digital readout has an autodetect mechanism for the high and low limits. You cannot exceed the limits if you are using the dial but only if you use the keypad. In case you do, the program will not let you download an illegal parameter and you'll be requested to correct your setting.

The 10 MHz Ref sub-group provides selection between internal or external references. The 3156B has two reference options: Backplane reference and TCXO. The backplane reference is routed to the VXI controller. Normal reference is 100ppm however, if an external reference is applied to the controller, the reference is automatically routed to all modules that feed off the backplane clock reference. The TCXO replaces the backplane reference for cases were 1ppm is sufficient for the application. An external reference input is provided for applications requiring better accuracy and stability. Click on the Internal button to select the Internal reference, or the External button to activate the external reference input.

WARNING:

By selecting an external reference you are disabling the internal reference circuit. If you do not have a 10 MHz reference connected to the instrument, the output will generate erroneous frequencies.

Enable

The enable group affects the 3156B only when the operate enable source is set to software (software is the default enable source). If the output state is on, clicking on the operate Enable button will stimulate the output. In some cases, the disable can stop the output. The various options to enable and disable the output waveforms, using the enable commands are given in **Table 3-2**.

The Arbitrary & Sequence Panel

The Arbitrary & Sequence panel, as shown in **Figure 5-5**, is invoked by pressing the ARB button on the Panels bar. Note that if you invoke the Arbitrary & Sequence Panel from the Panels menu, the 3156B will not change its output type. On the other hand, if you select the arbitrary, or the sequenced options from the Main Panel, the 3156B will immediately change its output to the selected waveform type. The functional groups in the Arbitrary Waveforms Panel are described below.

Parameters

The Parameters group contains two parameters for each channel: Amplitude and Offset. Actually, the values exhibited in this group are exactly the same as in the Main Panel, so every time you change amplitude and offset in the Parameters group, the other panels are updated automatically.

To access the required parameter, click on the parameter name. The LED next to the required parameter turns on. The value that is associated with the lit LED is displayed on the digital display. You can use the dial, keyboard, or the [↑] [↓} keys to adjust the readout to the required setting. After you modify the reading, press Execute to update the 3156B with the new reading.

Note:

Normal color of the digital reading is dark blue. If you modify the reading, the color changes to a lighter shade of blue, indicating that the 3156B has not been updated yet with the new parameter. Pressing Execute will update the instrument and will restore the color of the digital readout to dark blue, indicating that the displayed value is the same as the generator setting.

Also note that the digital readout has an autodetect mechanism for the high and low limits. You cannot exceed the limits if you are using the dial but only if you use the keypad. In case you do, the program will not let you download an illegal parameter and you'll be requested to correct your setting.

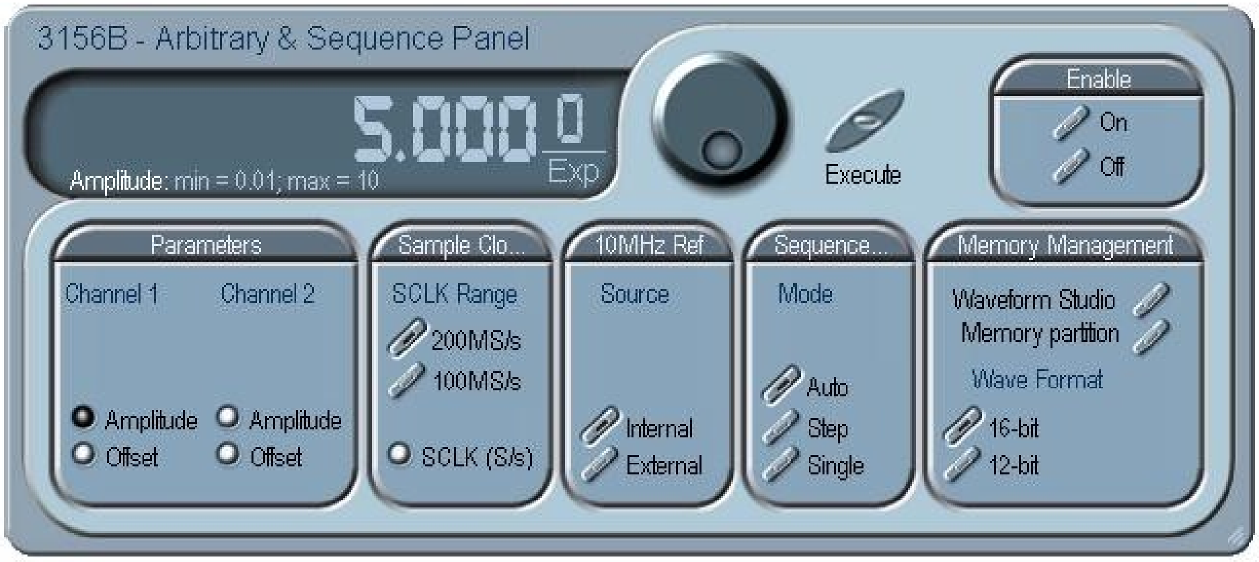


Figure 4-5, The Arbitrary & Sequence Panel

Sample Clock

The Sample Clock group is comprised of parameters that control the sample clock mode and frequency. The 3156B has two sample clock ranges: 200MS/s and 100MS/s. Using the 200MS/s sample clock frequency, the number of waveform points is restricted to even number only. If you use waveform data files that contain data that does not divide by 2, then the 100MS/s sample clock range must be used. Select the appropriate sample clock range by clicking on the 200MS/s or 100MS/s buttons.

Select the SCLK light to adjust the sample clock rate. Note that the sample clock rate is programmed in units of S/s (samples per second) and will affect the instrument only when it is programmed to output arbitrary or sequenced waveforms. The SCLK parameter has no effect on standard waveforms.

To access the required parameter, click on the button until the LED next to the required parameter turns on. The value that is associated with the lit LED is displayed on the digital display. You can use the dial, keyboard, or the [↑] [↓} keys to adjust the readout to the required setting. After you modify the reading, press Execute to update the 3156B with the new reading.

10MHz Ref

The 10 MHz Ref group is a duplicate of the same group in the Main Panel. It provides selection between internal or external references. The 3156B has two reference options: Backplane reference and TCXO. The backplane reference is routed to the VXI controller. Normal reference is 100ppm however, if an external reference is applied to the controller, the reference is automatically routed to all modules that feed off the backplane clock reference. The TCXO replaces the backplane reference for cases were 1ppm is sufficient for the application. An external reference input is provided for applications requiring better accuracy and stability. Click on the Internal button to select the Internal reference, or the External button to activate the external reference input.

WARNING:

By selecting an external reference you are disabling the internal reference circuit. If you do not have a 10 MHz reference connected to the instrument, the output will generate erroneous frequencies.

Sequence Advance

The Sequence Advance group provides control over advance modes for the sequence generator. Advance options are: Auto, Stepped and Single. Refer to the 3156B manual to find out more when and how to use these advance modes. You should be careful while selecting modes because it is possible to cause settings conflict, for example, if you select the Single option and you forgot to change your run mode to Triggered.

Memory Management

The memory management group manages waveform format and provides access to the memory partition and waveform studio screens. The wave format buttons are important to set before you download waveforms to the memory. The 16-bit and 12-bit define for the 3156B how many vertical bits are used for waveform generation. If 12-bit data points are used, the 3156B automatically adjusts the internal resolution to the input file.

The Waveform Partition button opens a screen as shown in **Figure 4-6** and the Waveform Studio button opens a screen as shown in **Figure 4-7**. Information how to use these screens is given in the following paragraphs.

Enable

The enable group affects the 3156B only when the operate enable source is set to software (software is the default enable source). If the output state is on, clicking on the operate Enable button will stimulate the output. In some cases, the disable can stop the output. The various options to enable and disable the output waveforms, using the enable commands are given in **Table 3-2**.

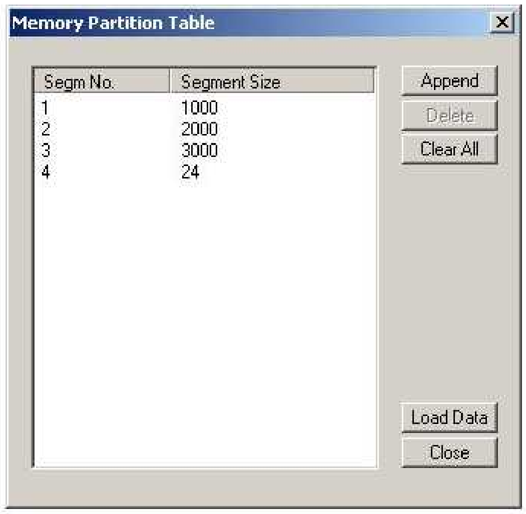
Using the Memory Partition Table

If you want to learn more about waveform memory and segment control, you should refer to section 3 of this manual. In general, the 3156B can generate arbitrary waveforms but, before it can generate anything, waveforms must be downloaded to the instrument from a host computer. Waveforms are downloaded to the instrument as coordinates and are stored in the 3156B in a place designated as “waveform memory”. The waveform memory has a finite size of 1Meg.

Having such long memory does not necessarily mean that you have to use the entire memory every time you download a waveform. On the contrary, the 3156B allows segmentation of the memory so that up to 4096 smaller waveforms could be stored in this memory. There are two ways to divide the waveform memory to segments: 1) Define a segment and load it with waveform data, define the next and load with data, then the third etc. or 2) Use what WaveCAD has to offer and that is to make up one long waveform that contains many smaller segments, download it to the instrument in one shot and then download a memory partition table that splits the entire waveform memory into the required segment sizes.

Want to use it? Here is how it is done.

Point and click on the Memory Partition. A dialog box as shown in **Figure 4-6** will pop up.



Segm No.	Segment Size
1	1000
2	2000
3	3000
4	24

Figure 4-6, The Memory Partition Table

The two main fields in the segment table are Segment number and segment size. The **Seg No** (segment number) is an index field that can have values only, from 1 to 4096. The **Segment Size** is always associated with the segment number. You can program any segment size from 1 to 512k or from 2 to 1Meg, depending on the SCLK setting.

Use the **Append** key to add a segment at the end of the segment list. If you highlighted a segment, the Append key turns automatically to insert Use the **Insert** key to insert a segment at the cursor location. The **Delete** key is used for deleting a segment at the cursor position.

The **Clear All** key will remove all segments from the table and will let you start designing your segment table from fresh.

Click on the **Close** to discard of the contents of the dialog box without saving your last actions and to remove the Segment Table from the screen.

The **Load Data** key has double action, it will download the segment table to the instrument and will store the contents of your segment table.

TIP:

The Memory Partition table does not download waveform. Use the memory partition table only if you merged a few waveforms to one. The partition table then divides the memory to the individual and original size of each waveform. If you download waveforms using the waveform studio or used ri3156B_load_arb_data function, it already contain segment size variable and there is no need for further use of the memory partition table.

Using the Waveform Studio

The Waveform Studio, as shown in **Figure 4-7** has two parts: 1) Segment Table and 2) Sequence Table. The purpose of the waveform studio is to provide access to waveform files that are already resident in the system. These files can be delegated to various segments and later be used as individual waveforms or combined into complex sequences.

The Segment Table

Using the Segment Table you may list and download waveform files that were previously stored on the computer. The table shows the segment number and its associated file name, length and its download status. There are other means to download waveforms to memory segments such as the Wave Composer and individual function calls; The waveform studio makes it easier by combining multiple and complex commands into one simple dialog box.

To access the Segment table, click anywhere on the Segment Table area. If it was not yet, it will turn white as opposed to the Sequence Table area that turns gray. The Segment Table area is divided into three parts: the table area, the waveform shape area and control buttons. When you point and click on one of the waveforms, its shape is shown in the Waveform Shape window.

The Segment Table has four fields:

The **Seg** field contains numbers from 1 through 4096, designating the programmed memory segment. Note that memory segments are numbered from 1 to 4096.

The **State** field shows the current status of the memory segment. It can be *Free*, if no file has yet been assigned to this segment number, or *Mapped*, if file name has been assigned to the segment but the Download button has not been used yet to move the file to the 3156B memory, or *Loaded*, if the process has been completed by pressing either the Download button or the All (download all) button.

The **File** field is an edit field that lets you browse and select file names to be applied to a specific memory segment. To change or add file name, point and click on the File name field and either type your path or browse to the file location and let Windows find the right path.

The **Length** field displays the length of the selected memory segment. Memory segments size may be programmed from 1 to 512k or from 2 to 1Meg, depending on the SCLK setting. Note that the length field is not accessible and shown for reference purpose only.

TIP:

Point and click on one of the segments to show its shape in the Waveform Shape window.

Description of the various buttons in the Segment Table is given below.

Append – adds segment number at the end of the table

Insert – adds a segment above a highlighted segment line

Delete – removes a highlighted segment

Channel 1 – shows segment table for channel 1 only

Channel 2 – shows segment table for channel 2 only

Save – saves current table settings

Download – downloads a highlighted segment only to the 3156B memory

All – downloads the complete table to the 3156B memory

Clear Mem – wipes out the entire memory and clears the table for fresh settings

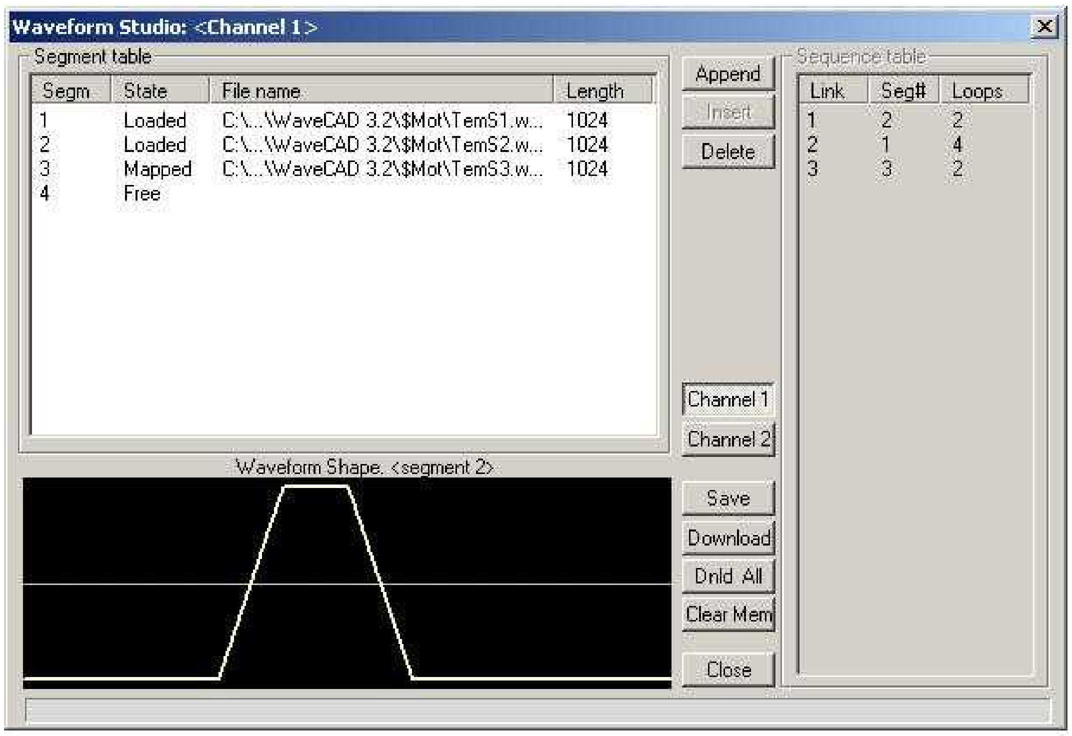
Close – removes the Waveform Studio from the screen. If you have not saved your work, the table setting will be lost.

The Sequence Table

As was explained in the above, the waveform memory can be divided into smaller segments and up to 4096 segments can be defined and used as individual arbitrary waveforms. Having a limited size of waveform memory can, for some applications, pose a limitation however, if sections of the waveform are repetitive, one may use the sequence generator to take these segments and replay them as part of the complete waveform without loosing valuable memory space and without scarifying waveform coherences, or integrity. The tool for using repetitive and multiple segments in one long waveform is called Sequence Generator. The 3156B has two separate sequence generators, one for each channel and WaveCAD has a special dialog box where sequences are designed. This tool is called – Sequence Table.

Using the Sequence table you can use waveforms that you downloaded to the 3156B from the Segment table to link and loop in random order to create one long and complex waveform that combines the individual memory segments.

The Sequence Table is demonstrated in **Figure 4-7**. To access the Sequence table, click anywhere on the Sequence Table area. If it was not yet, it will turn white as opposed to the Segment Table area that turns gray.



Segment table

Segm	State	File name	Length
1	Loaded	C:\...\WaveCAD 3.2\$Mot\TemS1.w...	1024
2	Loaded	C:\...\WaveCAD 3.2\$Mot\TemS2.w...	1024
3	Mapped	C:\...\WaveCAD 3.2\$Mot\TemS3.w...	1024
4	Free		

Sequence table

Link	Seg#	Loops
1	2	2
2	1	4
3	3	2

Figure 4-7, The Sequence Table

There are three major elements that you should consider while programming a sequence table. They are: Link, Seg# and Loops. These terms are explained below.

Link - This parameter defines an index array for the sequence generator. When generating sequences, the instrument steps though the links in descending order therefore, make sure that you enter your waveform segments in exactly the order you would like them at the output.

Seg # - This parameter associates waveform segments with links. You can use different segments for different links or you can use the same segment for a number of links. There are no limitations how you associate links to segments, except you cannot program in the sequence table segments that were not defined earlier.

Loops – This parameter define how many times the segment will loop for the selected link. For example, if you program 2, the waveform will cycle twice through the same segment before transitioning to the next link.

Figure 4-7 shows an example of a 3-step sequence of which the first waveform is made of segment 2, which will loop 2 times; segment 1, looping 4 times and segment 3 looping 2.

HINT:

The 3156B has two separate sequence generators, one for each channel. If the 3156B is programmed to continuous run mode, make sure both channels have the same sequence length for inter-channel synchronization. For triggered run mode, each channel can be programmed for a unique sequence length.

The control buttons on the left of the Sequence Table have the same functionality as for the Segment Table.

Use the **Append** key to add a step at the end of the sequence list. Use the **Insert** key to insert a step at the cursor location. The **Delete** key is used for deleting a step at the cursor position.

The **Clear All** key will remove all steps from the table and will let you start designing your sequence table from scratch.

Click on the **Close** to discard of the contents of the dialog box without saving your last actions and to remove the sequence Table from the screen but click on the **Save** key if you want just to save your work before you close the dialog box.

The **Download** key has double action, it will download the sequence table to the instrument and will save the contents of your table so the next time you open this table, it will have the same contents as you saved in your previous session.

The Trigger Panel

The Trigger panel, as shown in **Figure 4-8**, is invoked by pressing the TRIG button on the Panels bar. Note that if you invoke the Trigger Panel from the Panels menu, the 3156B will not change its trigger mode. To modify the instrument run mode, use the Main Panel. The trigger parameters and setting in the Trigger Panel will have an effect on the 3156B only if an appropriate run mode setting has been selected. The functional groups in the Trigger Panel are described below.

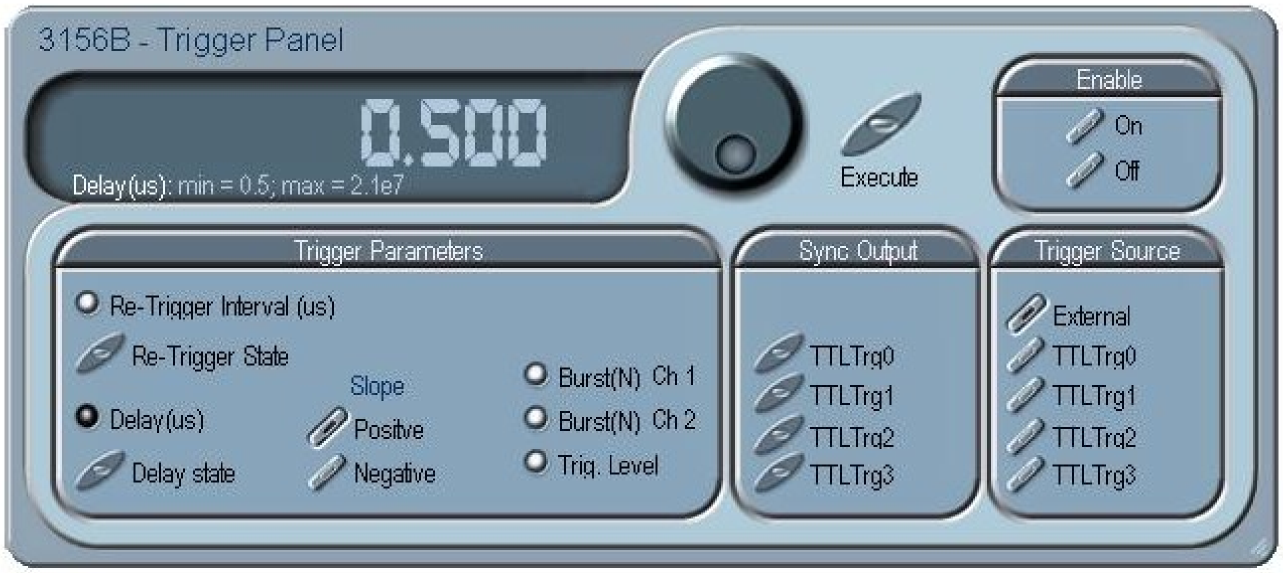


Figure 4-8, The Trigger Panel

Trigger Parameters

Trigger Parameters

The Trigger Parameters group provides access to Slope, trigger level and burst count. Note that different burst count can be programmed for each channel while the rest of the parameters in this group are common to both channels.

The delay state button toggles trigger delay on and off and the re-trigger delay toggles the re-trigger function on and off. Each button has an associated parameter that can be adjusted when one of the lights are on. The delay and re-trigger parameters are adjusted in units of μs from 0.5μs to 21s.

The Slope sub-group lets you select edge sensitivity for the trigger input of the 3156B. If you click on Positive, the instrument will trigger on the rising edge of the trigger signal. Likewise, if you click on Negative, the instrument will trigger on the falling edge of the trigger signal.

To change burst or trigger level values, point and click on one of these parameters. The value that is associated with the lit LED is displayed on the digital display. You can use the dial, keyboard, or the [↑] [↓} keys to adjust the readout to the required setting. After you modify the reading, press Execute to update the 3156B with the new reading.

Note:

Normal color of the digital reading is dark blue. If you modify the reading, the color changes to a lighter shade of blue, indicating that the 3156B has not been updated yet with the new parameter. Pressing Execute will update the instrument and will restore the color of the digital readout to dark blue, indicating that the displayed value is the same as the generator setting.

Also note that the digital readout has an autodetect mechanism for the high and low limits. You cannot exceed the limits if you are using the dial but only if you use the keypad. In case you do, the program will not let you download an illegal parameter and you'll be requested to correct your setting.

Sync Output

The 3156B front panel SYNC OUT connector generates signal that is synchronous with the main output. This signal can be turned on and off from the main panel. The same sync signal can be routed, in parallel, to one or more backplane TTLTrg lines. Note that the TTLTrg lines, depending on instrument settings, can be selected to either output signals or receive signals and therefore it is extremely important to be careful not to drive two adjacent modules as outputs. Due to space restrictions, only TTLTrg 0 through 3 are available on the trigger panel.

Trigger Source

The Main Panel output control group defines if the enable source is software, hardware, or mixed. If hardware or mixed enable options are selected, the 3156B must be told from where the trigger or enable signal will arrive. The instrument can be programmed to receive such signals from the front panel TRIG IN connector or from one of 8 backplane TTLTrg trigger lines. Due to space restrictions, only TTLTrg 0 through 3 are available on the trigger panel.

Enable

The Enable Group affects the 3156B only when the operate enable source is set to software (software is the default enable source). If the output state is on, clicking on the operate Enable button will stimulate the output. In some cases, the disable can stop the output. The various options to enable and disable the output waveforms, using the enable commands are given in **Table 3-2**.

The Modulation Panels

The Modulation functions were designed over two separate panels, as shown in **Figures 4-9 and 4-10**. The panels are invoked by pressing the MOD1 or MOD2 buttons on the panels menu. These panels provide access to all modulation functions and their respective run modes and parameters. The modulation functions that are available on both panels are: FM (frequency modulation), AM (amplitude modulation), FSK (frequency shift keying), Sweep and Frequency Hops. Using WaveCAD, modulation is programmed simultaneously for both channels however, using external utilities, each channel can be programmed to output different waveform types. The legal options for programming both channels are summarized in **Table 3-1**. When modulation is Off, the 3156B generates carrier signal (sinewave) at a frequency set by the carrier frequency parameter.

Besides the Modulation Mode and Enable groups which are common to both panels, each pf the modulation panels has a different set of functions and parameters. These groups are described below.

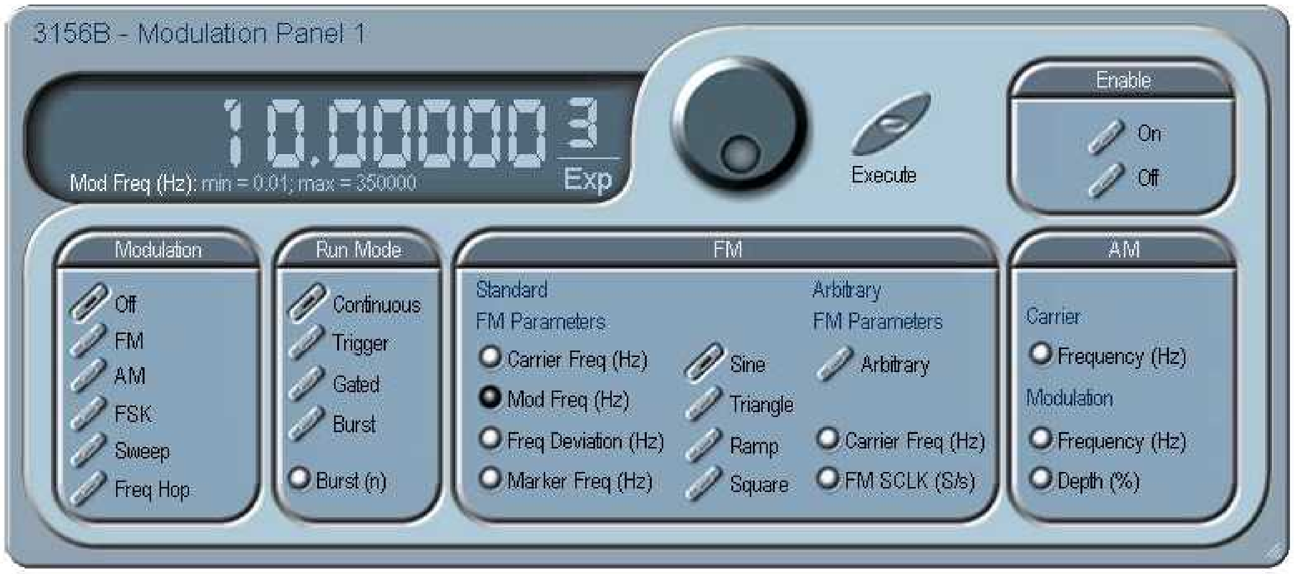


Figure 4-9, The Modulation Panel 1

Modulation

The Modulation Group provides selection of modulation function. When the Off button is pressed, the instrument generates carrier signal (sinewave) at a frequency set by the carrier frequency parameter. Note that the modulation group buttons also appear on the Modulation Panel 2. It doesn't really matter if you select the function from either panel 1 or 2.

Run Mode

The Run Mode group provides access to the various trigger modes of the modulation functions. Note that the run modes that are shown on this panel behave differently than the Main Panel run mode. For example, placing the 3156B in triggered FM, the instrument idles on carrier frequency until triggered. None modulated functions do not output any signal until triggered.

The burst parameter in this group programs the burst count for the modulation functions. To change burst count value, point and click on the burst parameter. The value that is associated with the lit LED is displayed on the digital display. You can use the dial, keyboard, or the [↑] [↓} keys to adjust the readout to the required setting. After you modify the reading, press Execute to update the 3156B with the new reading.

Note:

Normal color of the digital reading is dark blue. If you modify the reading, the color changes to a lighter shade of blue, indicating that the 3156B has not been updated yet with the new parameter. Pressing Execute will update the instrument and will restore the color of the digital readout to dark blue, indicating that the displayed value is the same as the generator setting.

Also note that the digital readout has an autodetect mechanism for the high and low limits. You cannot exceed the limits if you are using the dial but only if you use the keypad. In case you do, the program will not let you download an illegal parameter and you'll be requested to correct your setting.

FM

The FM Group is divided to two sub groups: Standard FM and Arbitrary FM. The difference between the two FM types is that with the standard FM, one can select between built-in standard waveforms to modulate the carrier. For arbitrary FM, the modulating waveforms are designed in the FM composer and can take any arbitrary shape as required by the application.

Press on Sine, Triangle, Ramp, or Square to modulate the carrier waveform with one of the built-in waveforms. Press Arbitrary to allow the FM composer generate custom waveforms to modulate your carrier signal.

There are 4 parameters that control standard FM. Point and click on Carrier Frequency to adjust the frequency of the carrier signal. The Modulation frequency parameter defines the frequency of the modulating waveform, which could be one of sine, triangle, ramp or square waves. Point and click on the Frequency deviation parameter to program the deviation range. The Marker Frequency defines a frequency point where the SYNC output on the front panel will output a marker signal.

The arbitrary FM becomes active when the Arbitrary button is depressed. When this function is selected, the standard FM parameters, except the carrier frequency which is common to both functions, have no effect anymore.

The Arbitrary FM has two parameters: Carrier frequency and FM SCLK. The carrier frequency is the frequency of the sine waveform before it is being modulated. The FM SCLK sets the sample clock frequency of the arbitrary waveform that modulates the carrier. The arbitrary waveform is built using the FM composer. Not that the arbitrary FM modulating waveform memory is separate from the main arbitrary memory and function. The specifications for the FM arbitrary sample clock and size is given in Appendix A.

To change the arbitrary FM parameters point and click on the Arbitrary FM parameters. The value that is associated with the lit LED is displayed on the digital display. You can use the dial, keyboard, or the [↑] [↓} keys to adjust the readout to the required setting. After you modify the reading, press Execute to update the 3156B with the new reading.

AM

The AM Group has parameters that control the amplitude modulation function. The carrier frequency is the frequency of the sine waveform before it is being modulated. Modulation Frequency and Modulation Depth determine the properties of the modulation envelop.

To change the AM parameters point and click on the AM parameters. The value that is associated with the lit LED is displayed on the digital display. You can use the dial, keyboard, or the [↑] [↓} keys to adjust the readout to the required setting. After you modify the reading, press Execute to update the 3156B with the new reading.

Enable

The Enable Group affects the 3156B only when the operate enable source is set to software (software is the default enable source). If the output state is on, clicking on the operate Enable button will stimulate the output. In some cases, the disable can stop the output. The various options to enable and disable the output waveforms, using the enable commands are given in **Table 3-2**. Note that when the 3156B has not been enabled, the output generates sine waveforms at frequency value set by the carrier frequency parameter.

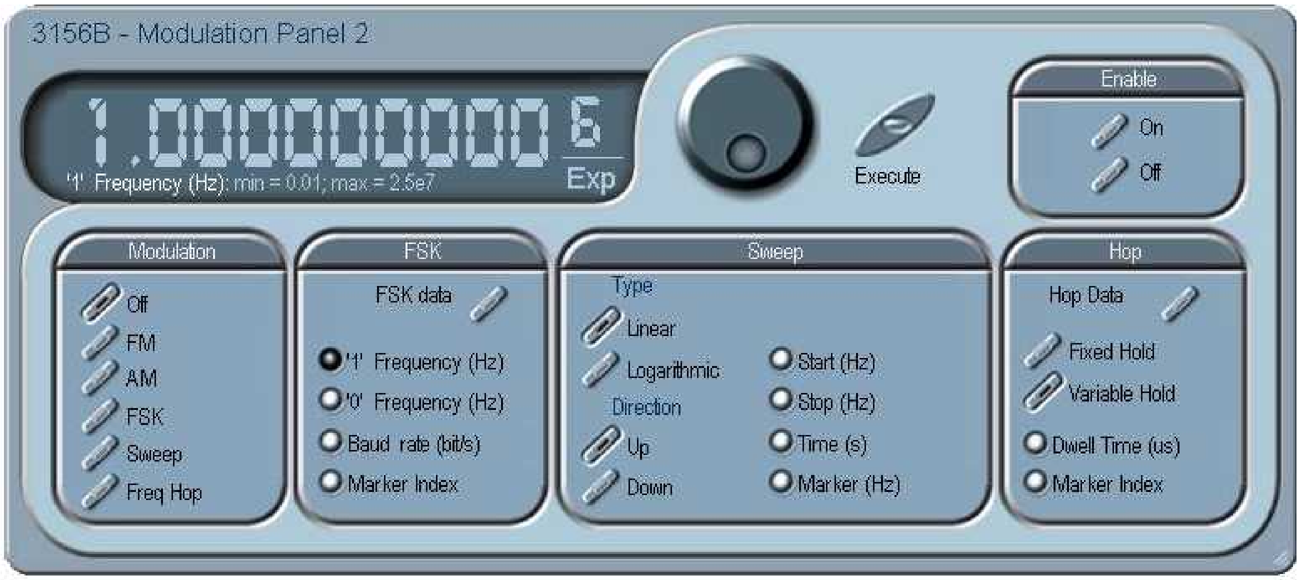


Figure 4-10, The Modulation Panel 2

The Modulation Panel 2, as shown in **Figure 4-10**, controls FSK, Sweep and Frequency Hop modulation. The Modulation and Enable groups are duplicated from the Modulation Panel 1. Description of the controls in this panel is given below.

FSK
The FSK group contains parameters for controlling the FSK function. The “0” Frequency represents the carrier frequency and the “1” Frequency represents the frequency to where the carrier will shift. The data which is required for the FSK sequence is programmed and stored in the FSK data table and the rate of which the frequency advances through the data is programmed using the Baud Rate parameter. Finally, the marker index points to a bit in the FSK sequence of which, when reached, the SYNC out will generate a marker pulse.

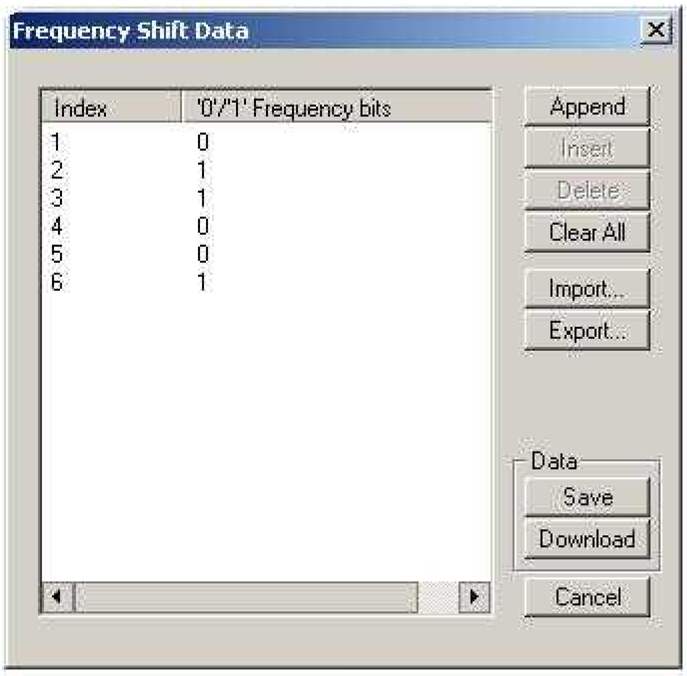
To change the FSK parameters point and click on one of the FSK parameters. The value that is associated with the lit LED is displayed on the digital display. You can use the dial, keyboard, or the [↑] [↓} keys to adjust the readout to the required setting. After you modify the reading, press Execute to update the 3156B with the new reading.

Note:

Normal color of the digital reading is dark blue. If you modify the reading, the color changes to a lighter shade of blue, indicating that the 3156B has not been updated yet with the new parameter. Pressing Execute will update the instrument and will restore the color of the digital readout to dark blue, indicating that the displayed value is the same as the generator setting.

Also note that the digital readout has an autodetect mechanism for the high and low limits. You cannot exceed the limits if you are using the dial but only if you use the keypad. In case you do, the program will not let you download an illegal parameter and you'll be requested to correct your setting.

The FSK Data table, as shown in **Figure 4-11**, programs a sequence of bits that are associated to an index number. The 3156B steps through the programmed index number and shifts or returns the carrier frequency according to the "0" and "1" list. When the index line is "0", the instrument generates sine waveforms with frequency set by the carrier frequency parameter. When the index line is "1", the generator modifies the sine waveform frequency to the shifted frequency setting.



Index	'0'/'1' Frequency bits
1	0
2	1
3	1
4	0
5	0
6	1

Figure 4-11, The FSK Data Table

The Append, insert and Delete buttons operate on individual line entries. The Clear All clears the FSK data table so you can start from a fresh list.

The Import… and Export… buttons let you save or retrieve files. The list is stored exactly in the same format as would be used by external utilities. The Save button stores the table in a temporary location for future uses by WaveCAD. The Download button updates the 3156B with the FSK table data.

Point and click on Cancel to discard of any changes you made to the table and to remove the FSK Data table from the screen.

Sweep

The Sweep group contains parameters for controlling the frequency sweep function. The Start and Stop parameters program the range of which the carrier will sweep and the Time defines how time that will lapse for one sweep cycle. The marker setting points to a specific frequency of which, when reached, the SYNC output generates a marker pulse.

The “0” Frequency represents the carrier frequency and the “1” Frequency represents the frequency to where the carrier will shift. The data which is required for the FSK sequence is programmed and stored in the FSK data table and the rate of which the frequency advances through the data is programmed using the Baud Rate parameter. Finally, the marker index points to a bit in the FSK sequence of which, when reached, the SYNC out will generate a marker pulse.

To change the sweep parameters point and click on one of the sweep parameters. The value that is associated with the lit LED is displayed on the digital display. You can use the dial, keyboard, or the [↑] [↓} keys to adjust the readout to the required setting. After you modify the reading, press Execute to update the 3156B with the new reading.

There are two sub-groups in the sweep block – Type and Direction.

Type – is used to select sweep the step. The two options are: Linear and Logarithmic.

Direction – is used to program sweep direction. Up select sweep from start to stop setting and Down selects sweep from the stop to start setting.

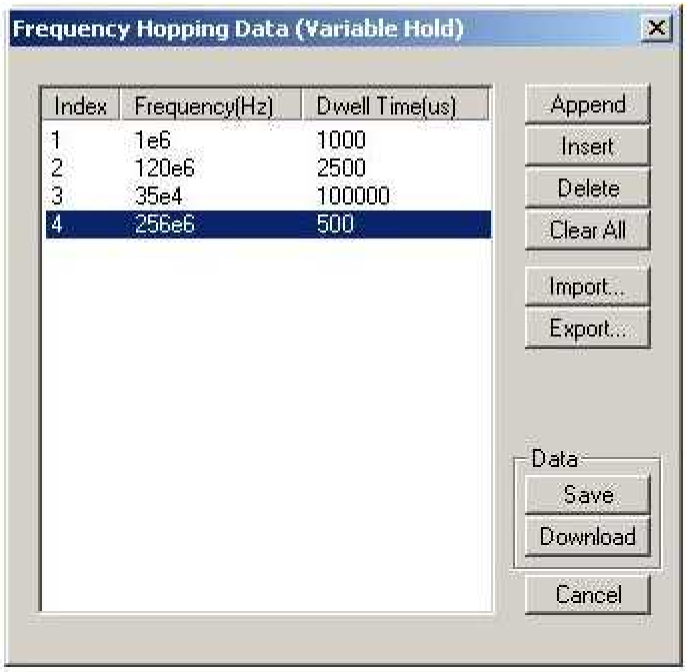
Frequency Hop

The Hop group contains parameters for controlling the frequency hop function. The 3156B has two hop modes: 1) Frequency hopes with fixed hold time and 2) Frequency hops with variable hold time. The hold, or dwell time determines the time that will lapse before the carrier will hop to the next frequency step.

The Dwell Time parameter is used in conjunction with the fixed hold frequency hop function. The Marker Index points to a specific step in the frequency hop table that will generate a marker pulse through the SYNC output.

To change the frequency hop parameters point and click on one of the hop parameters. The value that is associated with the lit LED is displayed on the digital display. You can use the dial, keyboard, or the [↑] [↓} keys to adjust the readout to the required setting. After you modify the reading, press Execute to update the 3156B with the new reading.

Frequency hops are generated using a hop data table, similar to that shown in **Figure 4-12**. Note that for the Fixed Hold mode, the Dwell Time parameter is omitted in the data table.



Index	Frequency(Hz)	Dwell Time(us)
1	1e6	1000
2	120e6	2500
3	35e4	100000
4	256e6	500

Figure 4-12, The Variable Hold Time Frequency Hop Table

The Append, insert and Delete buttons operate on individual line entries. The Clear All clears the data table so you can start from a fresh list.

The Import… and Export… buttons let you save or retrieve files. The list is stored exactly in the same format as would be used by external utilities. The Save button stores the table in a temporary location for future uses by WaveCAD. The Download button updates the 3156B with the frequency hop table data.

Point and click on Cancel to discard of any changes you made to the table and to remove the Data table from the screen.

The Digital and Video Panel

The Digital and Video panel is shown in **Figure 4-13**. The panel is invoked by pressing the DIG button on the panels menu. The panel provides access to all digital patterns and video stroke functions and their respective parameters. The Digital Patterns and Video waveform modes are selected from the Main Panel. Note that the video function mode utilizes both channels to operate in XY mode for the video drive. Using an external utility, the digital patterns can be used with another function on the other channel. The legal options for programming both channels are summarized in **Table 3-1**.

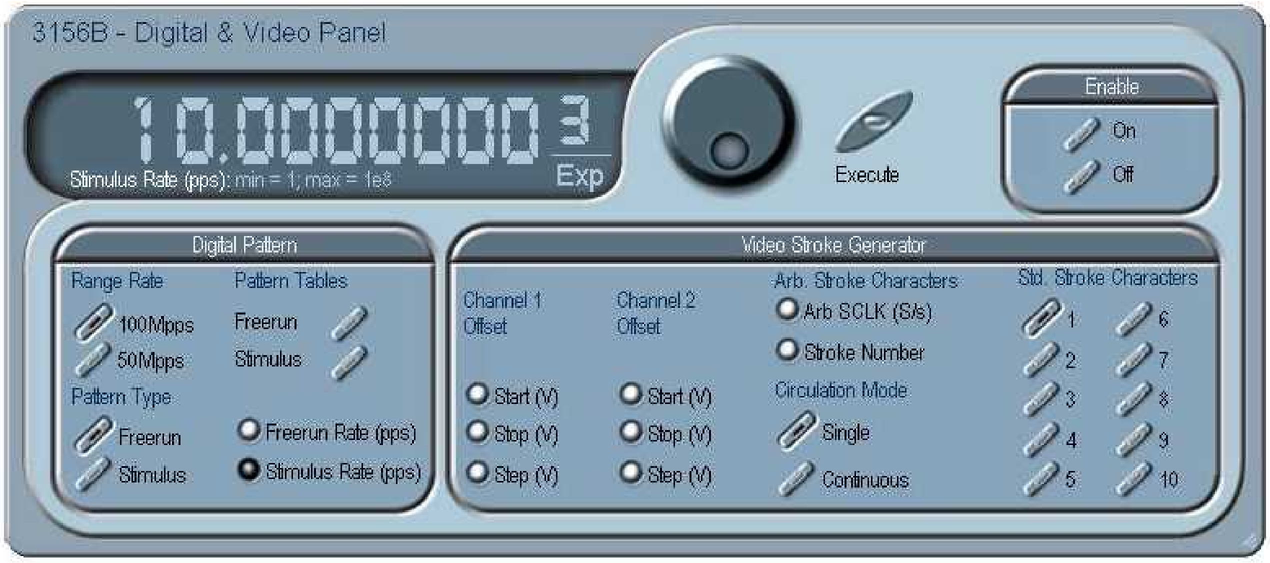


Figure 4-13, The Digital & Video Panel

Digital Patterns

The Digital Patterns Group provides selection of pattern type - Freerun or Stimulus and Pattern Rate Range - 100Mpps or 50Mpps. The Freerun range and the stimulus rate parameters define the rate of which the patterns change at the output however, the final Freerun rate may have a different response since it depends on the hold count as set in the Freerun Pattern Table.

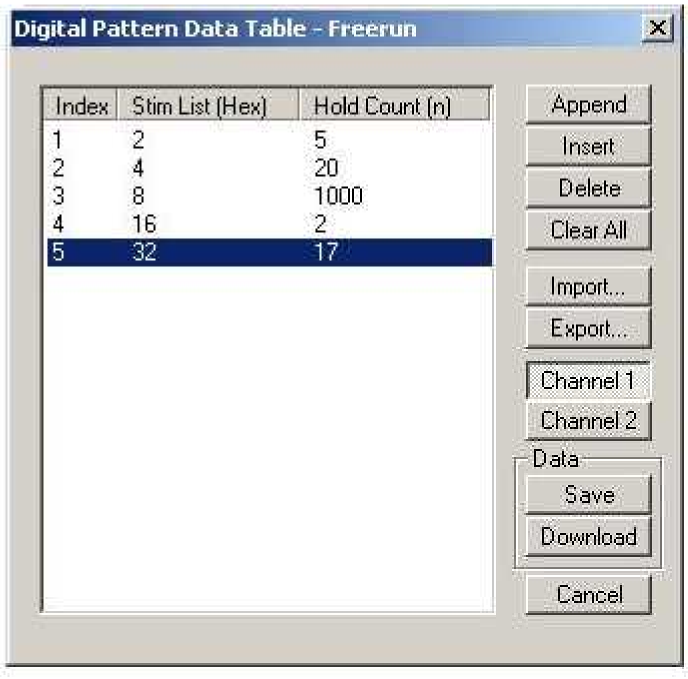
To change digital pattern rate, point and click on the parameter. The value that is associated with the lit LED is displayed on the digital display. You can use the dial, keyboard, or the [↑] [↓} keys to adjust the readout to the required setting. After you modify the reading, press Execute to update the 3156B with the new reading.

Note:

Normal color of the digital reading is dark blue. If you modify the reading, the color changes to a lighter shade of blue, indicating that the 3156B has not been updated yet with the new parameter. Pressing Execute will update the instrument and will restore the color of the digital readout to dark blue, indicating that the displayed value is the same as the generator setting.

Also note that the digital readout has an autodetect mechanism for the high and low limits. You cannot exceed the limits if you are using the dial but only if you use the keypad. In case you do, the program will not let you download an illegal parameter and you'll be requested to correct your setting.

The Freerun Pattern Data Table, as shown in **Figure 4-14** defines the patterns at the digital output connector. The Index column defines pattern steps along with their associated Stim List and Hold Count. The Stim List defines 12-bit pattern values in hex and the Hold Count defines sample clock dividers for the indexed step.



Index	Stim List (Hex)	Hold Count (n)
1	2	5
2	4	20
3	8	1000
4	16	2
5	32	17

Figure 4-14, The Digital Pattern Data Table - Freerun

If you look at the example in **Figure 4-14**, with pattern rate set to 1Mpps (mega patterns per second), step one will generate the following pattern at the digital output connector:

000000000010	Stable for 5μs (5 pattern rate clocks periods)
000000000100	Stable for 20μs (20 pattern rate clocks periods)
000000001000	Stable for 1ms (1000 pattern rate clocks periods)
000000010000	Stable for 2μs (2 pattern rate clocks periods)
000000100000	Stable for 17μs (17 pattern rate clocks periods)

Each channel can be designed to have its own digital patterns so program channel 1 table with the Channel 1 button depressed. When ready to program the 2nd channel, point and click on the Channel 2 button.

The Append, insert and Delete buttons operate on individual line entries. The Clear All clears the data table so you can start from a fresh list.

The Import… and Export… buttons let you save or retrieve files. The list is stored exactly in the same format as would be used by external utilities. The Save button stores the table in a temporary location for future uses by WaveCAD. The Download button updates the 3156B with the frequency hop table data.

Point and click on Cancel to discard of any changes you made to the table and to remove the Data table from the screen.

Hint:

The Freerun and the Stimulus pattern tables are almost identical except the Freerun has variable hold time. If you do not intend to use the variable hold time feature, it will be easier to program the Stimulus pattern table and have a single pattern rate value to program.

Video Stroke Generator

The Video Stroke Generator Group provides access to and control over generation of either standard, built-in or arbitrary, user-definable video stroke characters. The video function mode is selectable from the Main Panel. When this mode is selected, both channels must share the same mode because channel 1 is used for driving one video axis while the other channel drives the second axis.

There are two video stroke modes:

1) Standard stroke Characters – Using this mode, one can use one of 10 built-in stroke character: Cross locator, Crosshair, Positioned square, Vertical marker line, Horizontal marker line, Right hand arrow, Left hand arrow, Diamond overlay, Upright triangle and Inverted triangle. To select one of the built-in characters, point and click on one of the buttons which are numbered from 1 to 10.

2) Arbitrary Stroke Characters – Using this mode, one can use video files that are stored in memory segments, similar to arbitrary waveform segments. The arbitrary stroke generator is automatically selected when the arbitrary waveform mode is selected. In fact, the Arb SCLK parameter is a duplicate of the SCLK setting in the ARB panel and the Stroke Number duplicates the segment number in the same panel. The Offset Start, Stop and Step are unique for each channel as they simulate character movement speed and direction on the video monitor.

To change the video stroke parameters, point and click on the parameter. The value that is associated with the lit LED is displayed on the digital display. You can use the dial, keyboard, or the [↑] [↓} keys to adjust the readout to the required setting. After you modify the reading, press Execute to update the 3156B with the new reading.

The Circulation Mode has two options: Single and Continuous. Both options require an enable command to move the characters about the screen. The Single option moves a single step for each trigger and the Continuous option requires one enable signal to trigger one complete movement sequence.

Enable
The Enable Group affects the 3156B only when the operate enable source is set to software (software is the default enable source). If the output state is on, clicking on the operate Enable button will stimulate the output. In some cases, the disable can stop the output. The various options to enable and disable the output waveforms, using the enable commands are given in **Table 3-2**. Note that for video mode, only external triggers enable stroke movement and therefore, the Enable group has no effect on this mode.

The Utility Panel

The Utility panel, as shown in **Figure 4-15**, is invoked by pressing the UTIL button. The Utility panel provides access to general instrument references that are not directly related to waveform generation. Through this panel you can read information that is available specifically for service and information purposes

The *Serial Number* returns the same number as printed on your serial number label

The *Installed Option* returns either “0” for no option or “1” for TCXO option

The *Calibration Date* returns a date of which the instrument was last calibrated. Calibration cycle is normally 3 years so if the current date is over three years since last calibrated, make sure the 3156B is returned to the factory for calibration.

The Address Space returns A24 or A32, depending on your internal switch setting. The factory default is A32.

The *Firmware Revision* code should be compared to the latest firmware revision as is published from time to time by the factory. Firmware upgrades are provided by the factory.

To display the utility parameters, point and click on the required parameter. The value that is associated with the lit LED is displayed on the digital display.

Also available in this panel are some system commands such as Reset Instrument, Query Errors and Clear Error Queue. The Reset Instrument button is needed in places where you get stuck with instrument programming and want to start from a fresh and know state. The default parameters are described in the programming section of this manual.

The Dial and Execute buttons are on this panel for service purpose and therefore are not accessible for normal user operation.

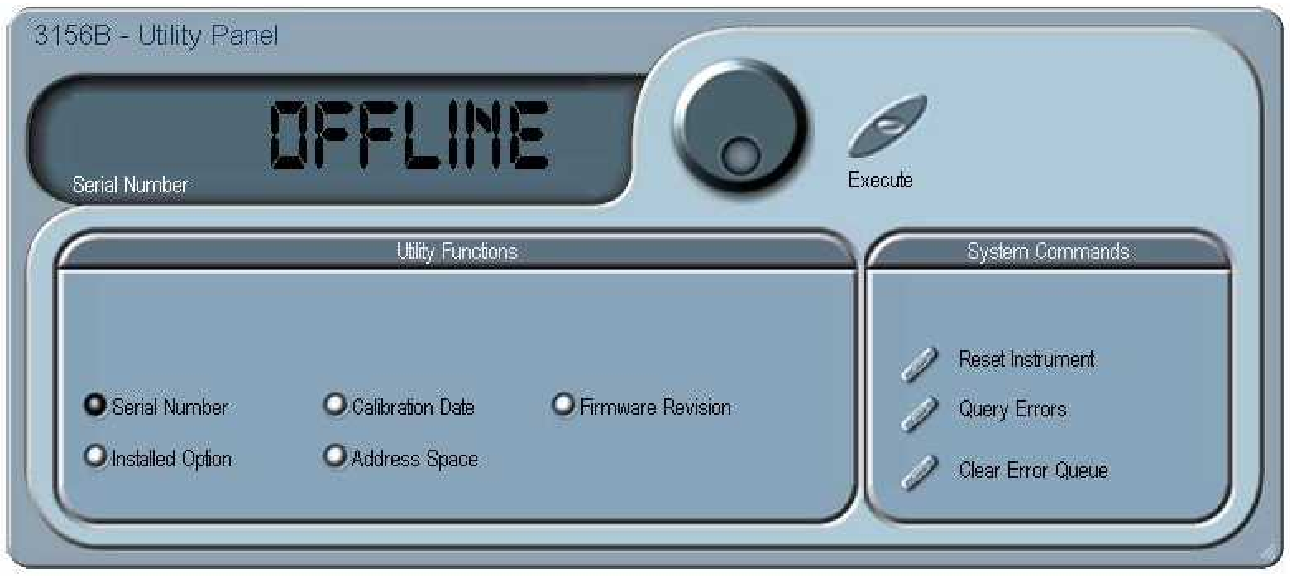


Figure 4-15, The Utility Panel

The Function Calls Editor

The Function Calls Editor is an excellent tool for learning low level programming of the 3156B. If you are not fully familiar with the syntax or the controlling functions, you can click on the Show List button and the complete list of functions will display in a separate dialog box as shown in **Figure 4-17**.

Invoke the Function Calls Editor from the System command at the top of the screen. Dialog box, as shown in **Figure 4-16** will pop up. If you press the Download button, the function call in the Function Call field will be sent to the instrument.

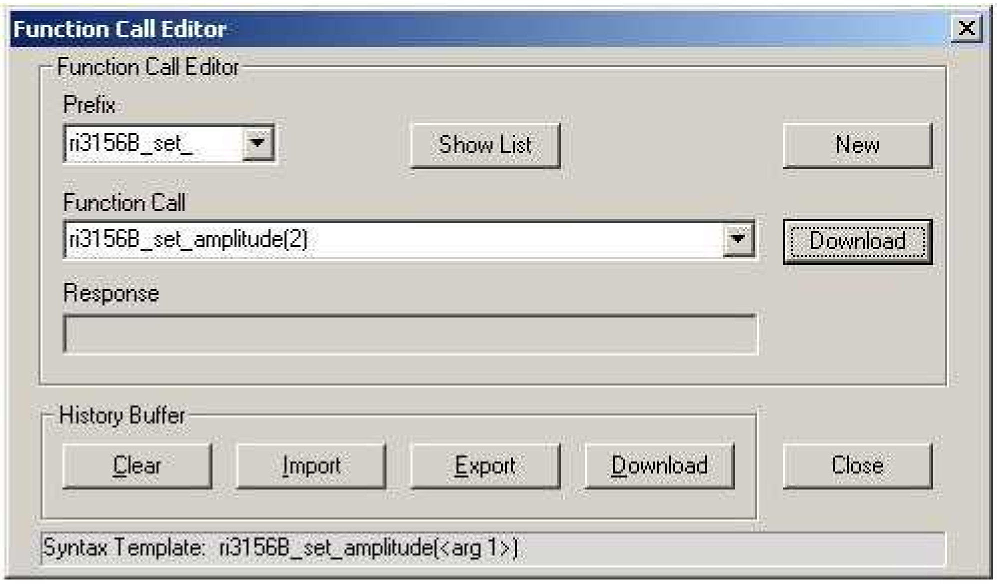


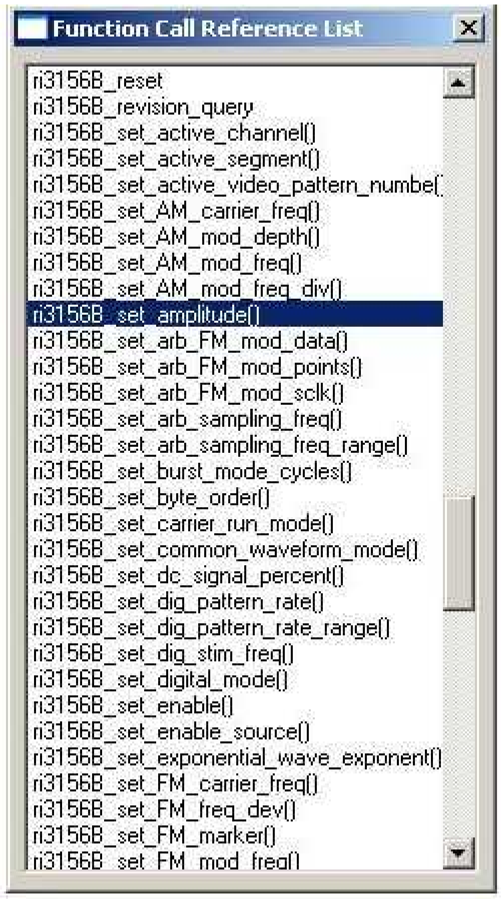
Figure 4-16, The Command Editor

The New button will clear the Function Call field and will replace it with a new Function Call Prefix. If you are familiar with the function call list, you can type in the function and its parameters. Note that, as you type the function name, a list if opened below the field showing all functions that are available in alphabetic order. If you are not familiar with the name of the functions, press Show List and the list will be displayed for your reference.

The Download button sends one command at a time to the instrument. If you send multiple commands, they are accumulated in a history buffer. You can watch this buffer if you click on the pointer at the right hand of the Function Calls field. Click on Clear to clear the history buffer, or Download to send the complete history buffer to the instrument. You can also Export history buffers for future repetitive operations using the Import function.

The Response field will display response to queries. The format for each response is given in the programming section of this manual.

Also note the syntax examples at the bottom of the dialog box. An example is given for each group of function call. Refer to the programming section of this manual if you are not sure about the syntax of the function calls.


Function Call Reference List

```
ri3156B_reset
ri3156B_revision_query
ri3156B_set_active_channel()
ri3156B_set_active_segment()
ri3156B_set_active_video_pattern_numbe(
ri3156B_set_AM_carrier_freq()
ri3156B_set_AM_mod_depth()
ri3156B_set_AM_mod_freq()
ri3156B_set_AM_mod_freq_div()
ri3156B_set_amplitude()
ri3156B_set_arb_FM_mod_data()
ri3156B_set_arb_FM_mod_points()
ri3156B_set_arb_FM_mod_sclk()
ri3156B_set_arb_sampling_freq()
ri3156B_set_arb_sampling_freq_range()
ri3156B_set_burst_mode_cycles()
ri3156B_set_byte_order()
ri3156B_set_carrier_run_mode()
ri3156B_set_common_waveform_mode()
ri3156B_set_dc_signal_percent()
ri3156B_set_dig_pattern_rate()
ri3156B_set_dig_pattern_rate_range()
ri3156B_set_dig_stim_freq()
ri3156B_set_digital_mode()
ri3156B_set_enable()
ri3156B_set_enable_source()
ri3156B_set_exponential_wave_exponent()
ri3156B_set_FM_carrier_freq()
ri3156B_set_FM_freq_dev()
ri3156B_set_FM_marker()
ri3156B_set_FM_mod_freq()
```

Figure 4-17, The Function Calls Reference List

The Wave Composer

Being an arbitrary waveform generator, the 3156B has to be loaded with waveform data before it can start generating waveforms. The waveform generation and editing utility is part of WaveCAD and is called – The Waveform Composer. This program gives you tools to create definitions for arbitrary waveforms. It can also convert coordinates from other products, such as, oscilloscopes and use them directly as waveform data. The program is loaded with many features and options so use the following paragraphs to learn how to create, edit and download waveforms to the 3156B using the Waveform Composer.

You can invoke the Waveform Composer program from two places: On the Panels bar click on WAVE, or from the Arbitrary & Sequence Panel click on "To Wave Composer". **Figure 4-18** shows an example of the wave composer.

The Wave Composer has three sections: Commands bar, Toolbar and Waveform screen. Refer to **Figure 4-18** throughout the description of these parts.

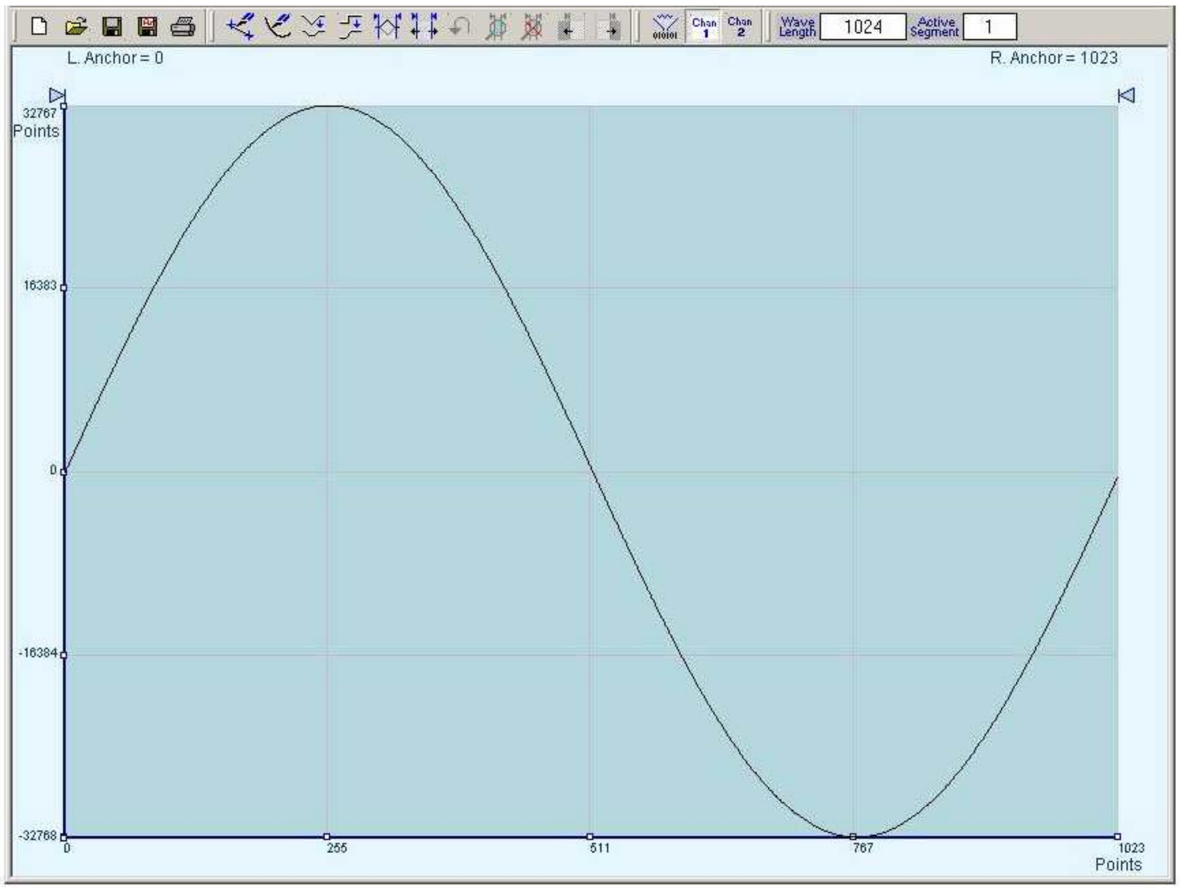


Figure 4-18, The Wave Composer Opening Screen

The Commands bar

The commands bar provides access to standard Windows commands such as File and View. In addition, there are WaveCAD-specific commands such as Edit, Wave and System.

In general, clicking on one of the commands opens a dialog box with an additional list of commands. Then, clicking on an additional command, may open a dialog box, or generate an immediate action. For example, Clicking on File and then Exit will cause an immediate termination of the Wave Composer. On the other hand, clicking on Wave and then on Sine, will open a Sine Wave dialog box that lets you program and edit sine wave parameters. The various commands in the Commands bar are listed and described below.

File Commands

The File command has 4 command lines that control waveform files. Also use this command to print the active waveform, or exit the wave composer program. Description of the various commands under File is given below.

New Waveform

The New Waveform (Ctrl+N) command will remove the waveform from the screen. If you made changes to the waveform area and use the New Waveform command, you should save your work before clearing the screen. The New Waveform command is destructive to the displayed waveform.

Open Waveform…

The Open Waveform… (Ctrl+O) command will let you browse your disk for previously saved waveform files and load these waveforms to the waveform area. This command is also very useful for converting waveform files to format that is acceptable by the Wave Composer. The Open Waveform command can convert ASCII. *CSV (comma delimited text), *PRN (space delimited text) and *.0* (LeCroy binary format). The Open dialog box in **Figure 4-19** shows the various file extensions that can be opened into the Wave Composer environment. The file that is opened is automatically converted to *.wav format and can later be saved as a standard WaveCAD file.

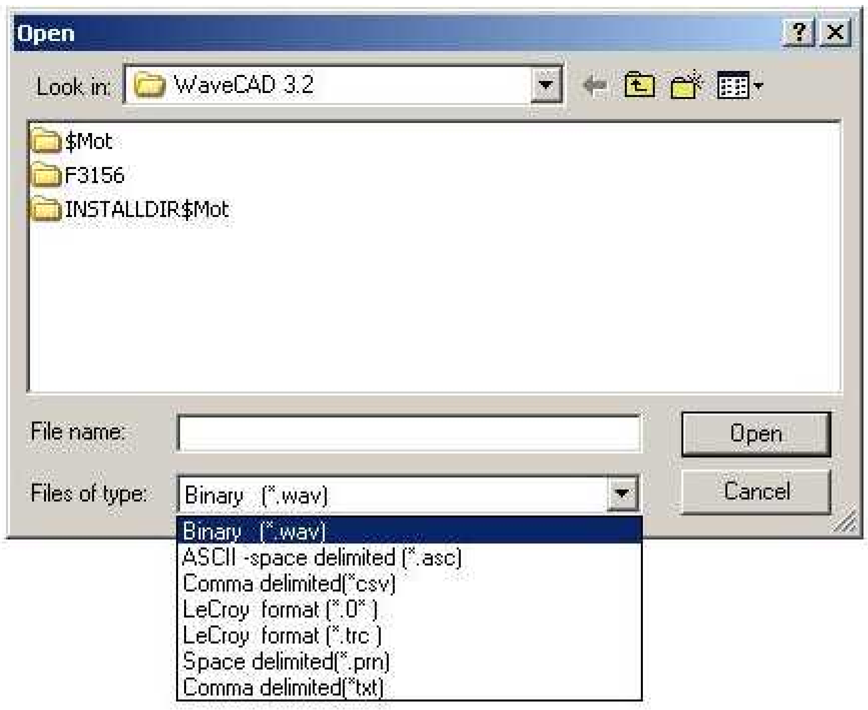


Figure 4-19, The Open Waveform Dialog Box

Save Waveform

The Save Waveform (Ctrl+S) command will store your active waveform in your 3156B directory, as a binary file with an *.wav extension. If this is the first time you save your waveform, the Save Waveform As… command will be invoked automatically, letting you select name, location and format for your waveform file.

Save Waveform As…

Use the Save Waveform As… command the first time you save your waveform. It will let you select name, location and format for your waveform file.

Print

With this command you may print the active Waveform Window. The standard printer dialog box will appear and will let you select printer setup, or print the waveform page.

Exit

The Exit command ends the current Wave Composer session and takes you back to the Panels screen. If you made changes to your waveform since it was last saved, the Wave Composer will prompt you to Save or Abandon changes these changes.

Edit Commands

The Edit commands are used for manipulating the waveform that is drawn on the screen. The editing commands are explained in the following paragraphs.

Autoline

The Autoline command lets you draw straight-line segments. To draw a line the left mouse button at the start point. Click again at the next point and then click on the right mouse button to terminate this operation.

Sketch

The Sketch command lets you draw free-hand segments. To draw a line using this command click and hold the left mouse button at the start point. Release the mouse button when you want to stop and then click on the right mouse button to terminate this operation.

Smooth

The Smooth command lets you smooth out rough transitions on your waveform. This is done mathematically by multiplying waveform coordinates by the non-linear portion of a cubic parabola.

The Smooth operation is done on segments of the waveform that are bound by anchors. Anchor operation is described later in this chapter. Place the anchors on the left and right of your waveform segment and select the Smooth command. The waveform will change its shape immediately to follow the mathematical pattern of a parabolic curve.

Note that small segments with fast transitions, when combined with parabolic expressions have tendencies to generate even larger transitions. Therefore, make sure you omit such sections of the waveform when you use this operation.

Filter
The Filter used with this command is moving average. This is done by recalculating each point as an average of symmetrical number of adjacent points. When you select the Filter command, a dialog box pops up, letting you program the filter spacing in number of adjacent points. You can filter the entire waveform, or you may chose to filter a segment of the waveform by placing the anchors as boundaries on the left and right of the segment.

Invert
The Invert command lets you invert the entire waveforms, or marked segments of waveforms. The waveform is inverted about the 0-point axis.

Trim Left
The trim left command lets you trim waveforms to the left of the anchor point. This command is grayed out if the left anchor was not moved from its original left position. The waveform is trimmed and the point at the left anchor point becomes the first point of the waveform.

Trim Right
The trim right command lets you trim waveforms to the right of the anchor point. This command is grayed out if the right anchor was not moved from its original right position. The waveform is trimmed and the point at the right anchor point becomes the last point of the waveform.

Unmark
The unmark command removes the anchors from the waveform screen and resets anchor positions to point 0 and the last waveform point.

Undo
The Undo command undoes the last editing operation.

View Commands

The View commands have commands that let you view various sections of the waveform area. The View commands include: Zoom In, Zoom Out, Hide/Show Toolbars and Channel ½ waveforms. Description of the view commands is given in the following.

Zoom In
The zoom in command operates between anchors. Anchors are marked as left and right hand triangles. The default position of the anchors is the start and the end of the waveform. To move an anchor to a new location, click and hold on the triangle and drag the anchor to left or right as required. If you move the left anchor to the right and the right anchor to the left, the area between the anchor will zoom in as you select this command.

Looking at the Waveform Map, as shown in **Figure 4-20**, you'll see that the white portion is the zoomed area. Click and hold on the white area and move your cursor around and the waveform screen will be updated accordingly.

While zoomed in you can perform autoline and sketch editing, or zoom-in further by clicking and holding the mouse at one corner and releasing the mouse button at the other corner.

Zoom Out

The zoom out restores the screen to display the complete waveform.

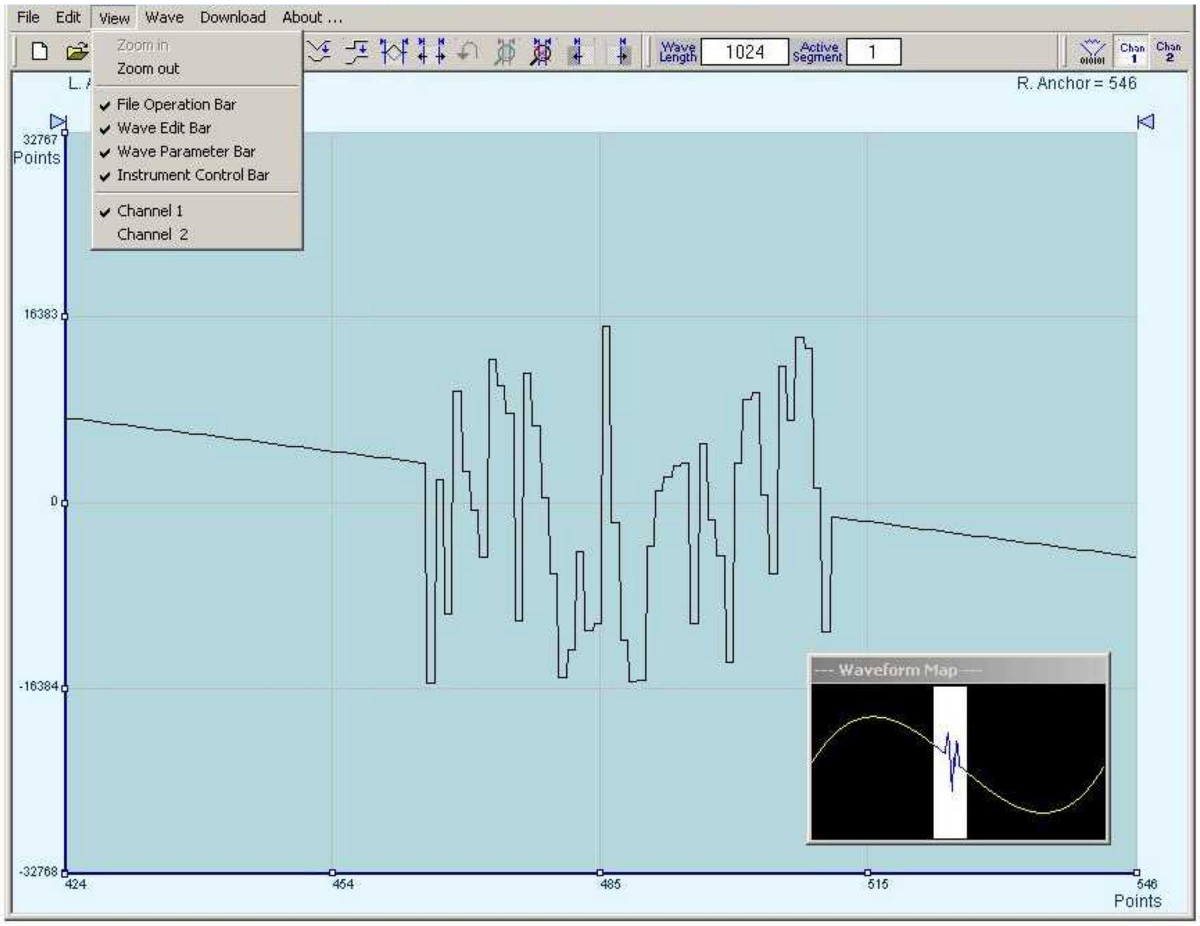


Figure 4-20, Zooming In on Waveform Segments

Channel 1

The Channel 1 Waveform command updates the waveform screen with the Channel 1 waveform. If you have not yet generated a waveform for channel 1, the waveform screen will show a dc level at vertical point 0.

Channel 2

The Channel 2 command updates the waveform screen with the Channel 2 waveform. If you have not yet generated a waveform for channel 2, the waveform screen will show a dc level at vertical point 0.

Wave Commands

The Wave commands let you create waveforms on the screen. The Wave command has a library of 8 waveforms: Sine, Sawtooth, Square, Sinc, Gaussian, Exponent, Pulse, and Noise. It also lets you create waveforms using the Equation Editor. Information how to create waveforms using the Wave commands is given below.

Creating Waveforms From the Built-in Library

You can create any waveform from the built-in library using the Wave command. Clicking on one of the Wave options will open a dialog box. An example of the Sine waveform dialog box is shown in **Figure 4-22**. This dialog box is representative of the rest of the waveforms, so other waveforms will not be described.

Creating Sine Waveforms

Use the following procedure to create sine waveforms from the built-in library. Click on Wave, then sine… the dialog box as shown in **Figure 4-22** will appear. You can now start programming parameters that are available in this box.

Start Point – Defines the first point where the created wave will start. Note that if you change the start point the left anchor will automatically adjust itself to the selected start point. The example shows start point set at point 0.

End Point – Defines where the created waveform will end. Note that as you change the end point the right anchor will automatically adjust itself to the selected end point. The example shows end point set at point 499.

Cycles – The Cycles parameter defines how many sine cycles will be created within the specified start and end points. The example below shows five sine cycles.

Amplitude – 16-bit of vertical define 32,768 incremental steps. The Amplitude parameter defines how many of these steps are used for generating the sine. The example is showing sine waveform with maximum peak-to-peak amplitude. Any number below the maximum will generate an attenuated sine.

Start Phase – The start phase parameter defines the angle of which the sine will start. The example shows start phase of 90°.

Power – The example shows sine cubed. Sine to the power of 1 will generate a perfect sine. Power range is from 1 through 9.

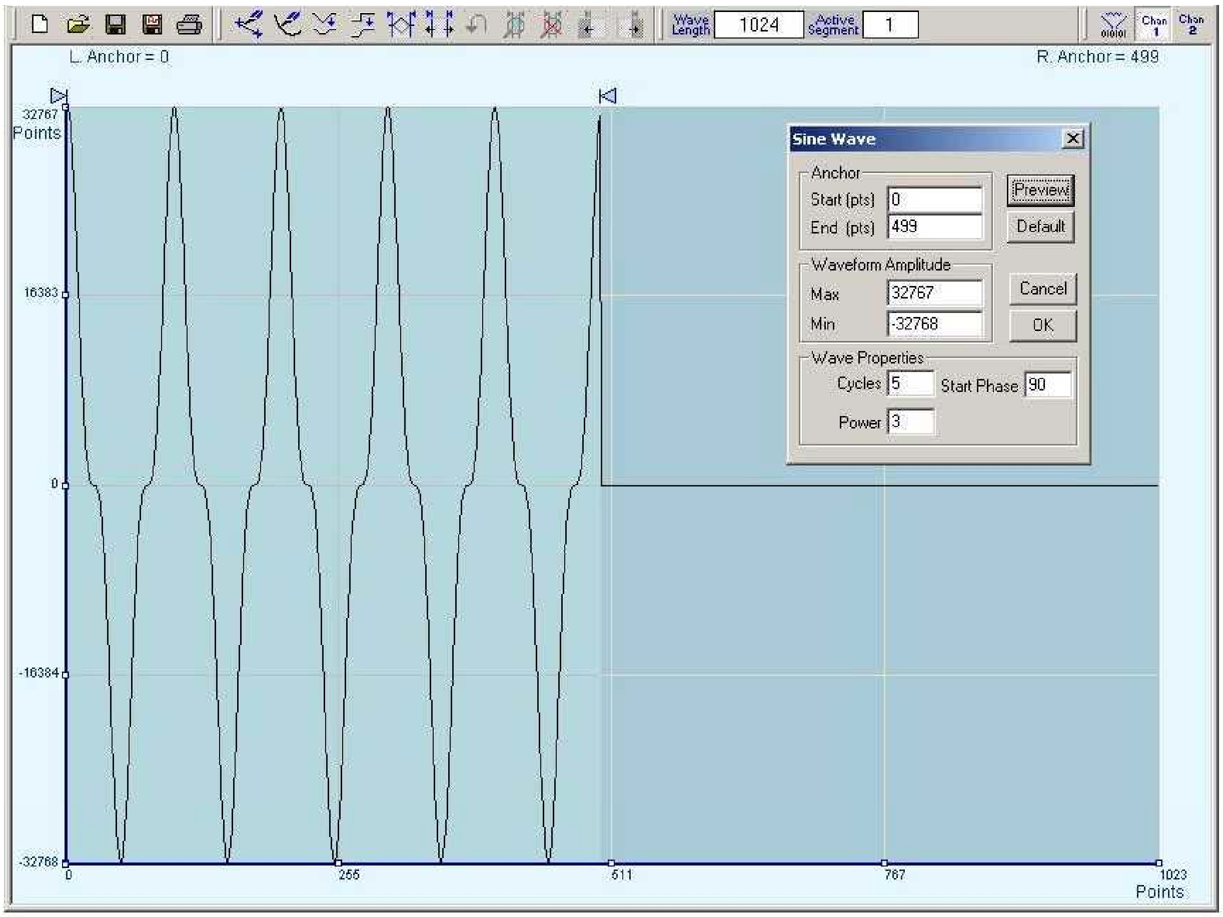


Figure 4-21, Generating Distorted Sinewaves from the built-in Library

The Toolbar

The toolbar contains icons for editing the waveform screen, icons for saving and loading waveforms, fields for selecting an active channel and for adjusting segment length and more. The Toolbar is shown in **Figure 4-23**. For the individual icons, refer to the descriptions above of the Wave Composer Menus.

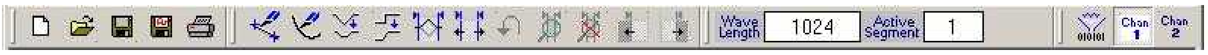


Figure 4-22, The Toolbar Icons

The Waveform Screen

Waveforms are created and edited on the waveform screen. **Figure 4-24** shows an example of a waveform created using the equation editor and the anchors to limit generation of the waveform between points 100 and 900. The various elements of the waveform screen are described below.

The waveform screen has two axes – vertical and horizontal. Both axes are divided into points.

The vertical axis is labeled from –16, 383 through 16,384 for a total of 32,768 point. This number represents 16 bits of vertical resolution and cannot be changed because it is critical to the range of which the 3156B operates.

The horizontal axis, by default has 1000 points (from point 0 to 999). This number can be changed using the Wave Length field in the Toolbar. The maximum length depends on the option installed in your instrument. The wave composer will let you define the horizontal axis to a maximum of 1Meg words).

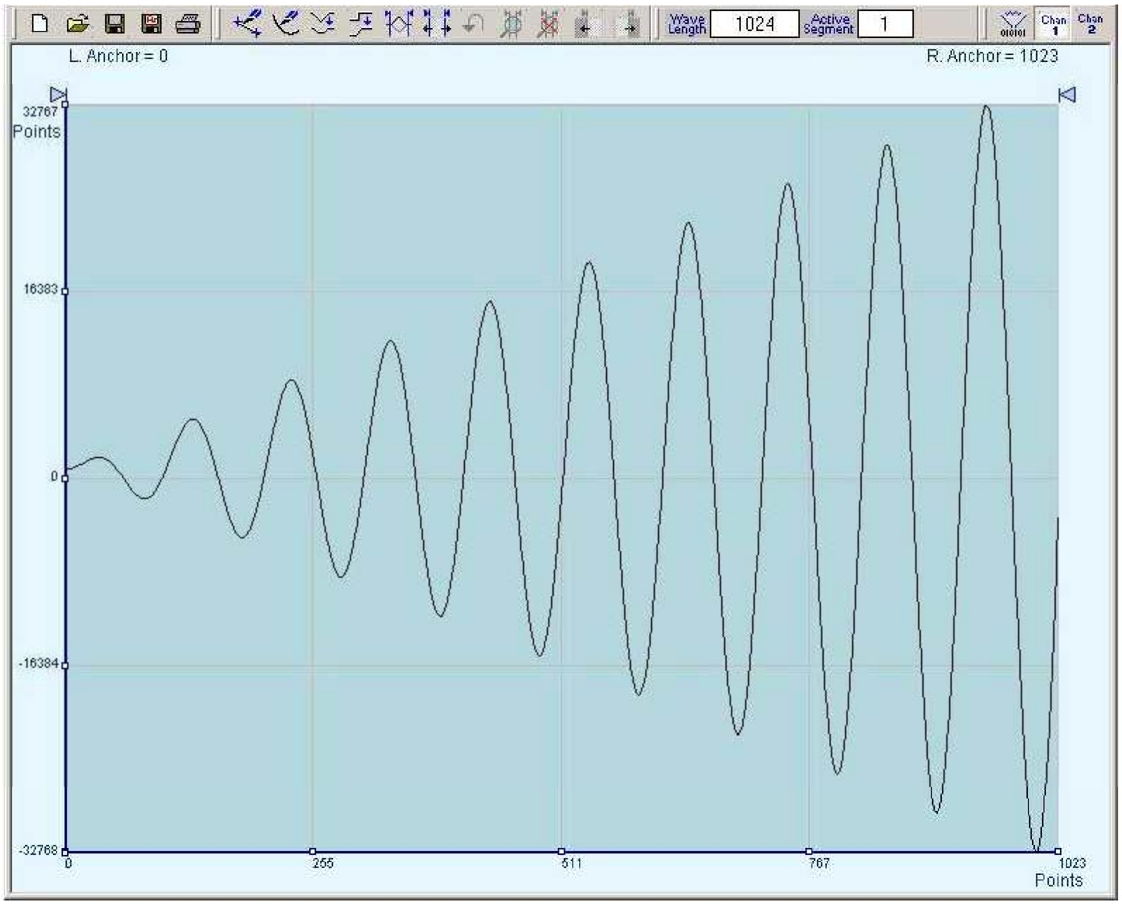


Figure 4-23, The Waveform Screen

Notice on the left top and on the right top there are two triangles pointing to the center of the screen. These are the anchors. The anchors are used as the start and end pointers where your waveform will be created. For example, if you want to create a sine waveform between point 100 and point 500, you place the left anchor at point 100 and the right at point 500 and then generate the sine from the built-in library.

There are two ways to control anchor placements.

1) Click and hold your mouse cursor on the left anchor triangle and then drag the curtain to the left position. Do the same for the right anchor. Notice the X and Y coordinates at the top of the waveform screen and how they change to correspond to your anchor placement.

2) You can also place your anchors in a more precise manner from the waveform library by programming the start and end points for the waveform. An example of anchor placement using the sine dialog box is shown in **Figure 4-21**.

Finally, when you are done creating and editing your waveform, you can save your work to a directory of your choice. The name at the title will show you the name you selected for storing your waveform and its path.

The FM Composer

The FM Composer looks and feels almost like the waveform composer except there is a major difference in what it does. If you look at the opening screen as shown in **Figure 4-24**, you'll see that the vertical axis is marked with frequencies. You'll see later that as you draw waveforms on the FM composer screen, these waveforms represent frequency changes and not amplitude changes as are generated by the waveform composer.

The FM composer is a great tool for controlling frequency agility by generating the agility curve as an arbitrary waveform. For example, if you create a sine waveform, the 3156B will generate frequency-modulated signal that will follow the sine pattern. The resolution and accuracy of the modulated waveform is unsurpassed and can only be duplicated by mathematical simulation. The FM composer is loaded with many features and options so use the following paragraphs to learn how to create and download modulating waveforms to the 3156B using the FM Composer.

Invoke the FM Composer from Panels bar. The Wave Composer has three sections: Commands bar, Toolbar and Waveform screen. Refer to **Figure 4-24** throughout the description of these parts.

The Commands bar

The commands bar is exact duplication of the commands bar in the Wave Composer. It provides access to standard Windows commands such as File and View.

In general, clicking on one of the commands opens a dialog box with an additional list of commands. Then, clicking on an additional command, may open a dialog box, or generate an immediate action. For example, Clicking on File and then Exit will cause an immediate termination of the FM Composer. On the other hand, clicking on Wave and then on Square, will open a Square Wave dialog box that lets you program and edit square wave parameters. The various commands in the Commands bar are listed and described below.

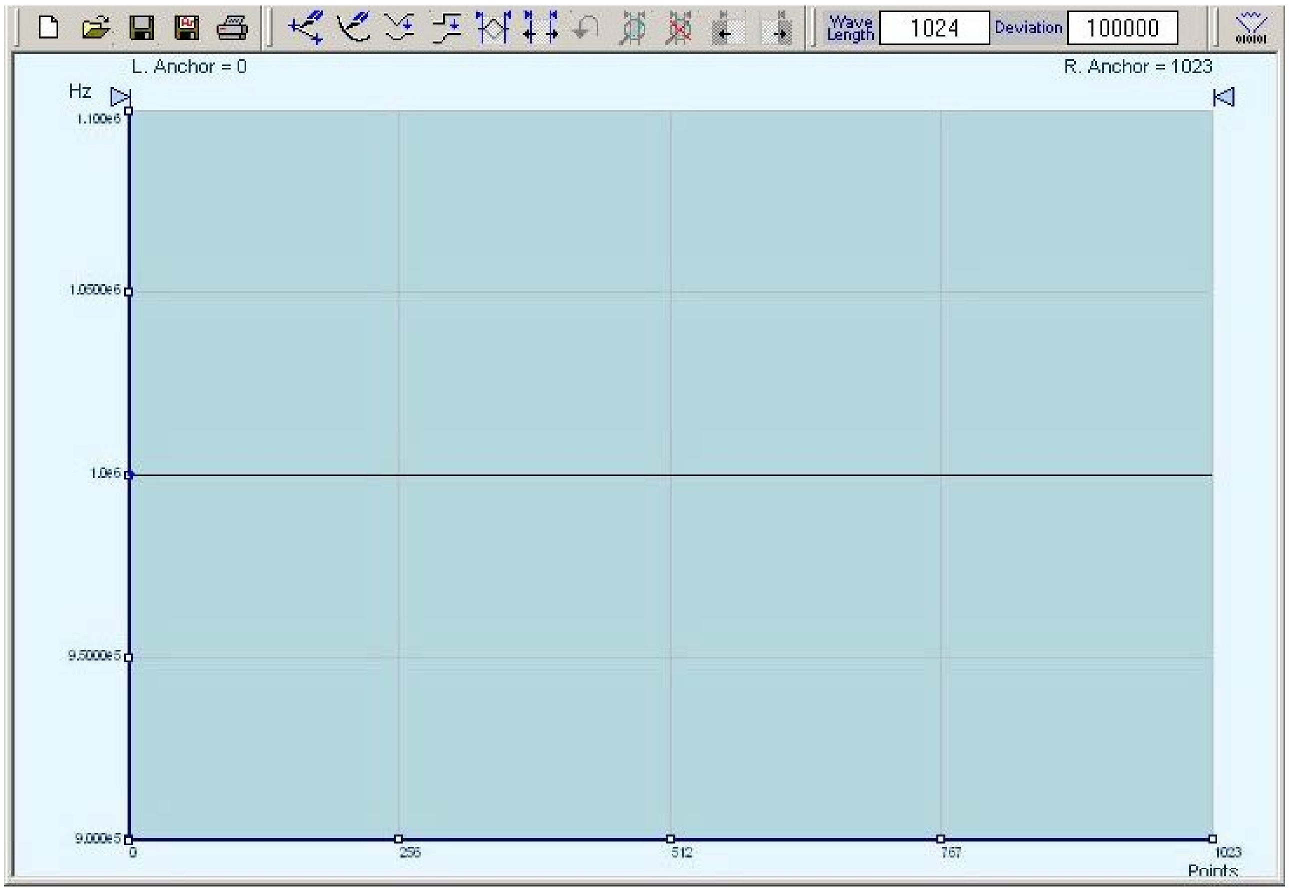


Figure 4-24, The Fm Composer opening Screen

File Commands

The File command has 4 command lines that control waveform files. Also use this command to exit the FM composer program. Description of the various commands under File is given below.

New Waveform

The New Waveform command will remove the waveform from the screen. If you made changes to the waveform area and use this command, you should save your work before clearing the screen. The New Waveform command is destructive to the displayed waveform.

Open Waveform…

The Open Waveform… command will let you browse your disk for previously saved waveform files and load these waveforms to the waveform area. This command is also very useful for converting waveform files to format that is acceptable by the Wave Composer.

Save Waveform

The Save Waveform command will store your active waveform in your 3156B directory, as a binary file with a *.wvf extension. If this is the first time you save your waveform, the Save Waveform As… command will be invoked automatically, letting you select name, location and format for your waveform file.

Save Waveform As…

Use the Save Waveform As… command the first time you save your waveform. It will let you select name, location and format for your waveform file.

Print

With this command you may print the active Waveform Window. The standard printer dialog box will appear and will let you select printer setup, or print the waveform page.

Exit

The Exit command ends the current FM Composer session and takes you back to the Panels screen. If you made changes to your waveform since it was last saved, make sure to Save your work before you use this command.

Wave Commands

The Wave commands let you create waveforms on the screen. The Wave command has a library of 6 waveforms: Sine, Triangle, Square, Exponent, Pulse, and Noise. It also lets you create waveforms using an Equation editor. Information how to create waveforms using the Wave commands is given below.

Creating Waveforms From the Built-in Library

You can create any waveform from the built-in library using the Wave command. Clicking on one of the Wave options will open a dialog box. An example of the Sine waveform dialog box is shown in **Figure 4-25**. This dialog box is representative of the rest of the waveforms, so other waveforms will not be described.

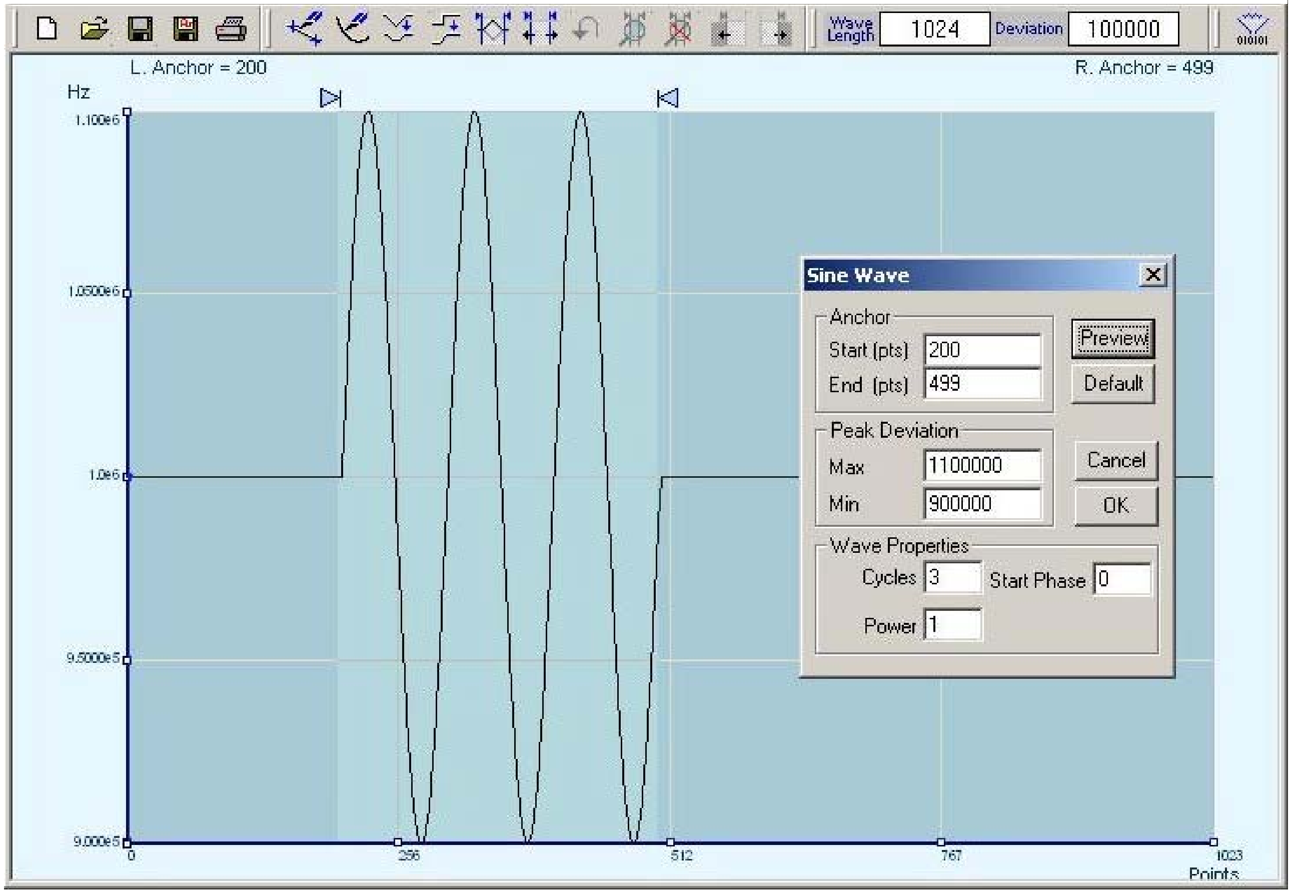
Creating Sine Waveforms

Use the following procedure to create sine waveforms from the built-in library. Click on Wave, then sine… the dialog box as shown in **Figure 4-25** will appear. You can now start programming parameters that are available in this box.

Start Point Anchor – Defines the first point where the created wave will start. Note that if you change the start point the left anchor will automatically adjust itself to the selected start point. The example shows start point set at point 200.

End Point Anchor – Defines where the created waveform will end. Note that as you change the end point the right anchor will automatically adjust itself to the selected end point. The example shows end point set at point 499.

Max. Peak Deviation – This parameter defines the forward peak deviation. Note that the forward peak deviation cannot exceed the pre-defined Deviation parameter as shown on the Toolbar. In case you need to exceed the pre-defined peak value you must quit this box and modify the Deviation parameter to provide sufficient range for the forward peak deviation range.



Sine Wave

Anchor
Start (pts) 200
End (pts) 499

Peak Deviation
Max 1100000
Min 900000

Wave Properties
Cycles 3
Start Phase 0
Power 1

Figure 4-25, Generating Sine Modulation Using the FM Composer

Min. Peak Deviation – This parameter defines the backwards peak deviation. Note that the backwards peak deviation cannot exceed the pre-defined Deviation parameter as shown on the Toolbar. In case you need to exceed the pre-defined peak value you must quit this box and modify the Deviation parameter to provide sufficient range for the backwards peak deviation range.

Cycles – The Cycles parameter defines how many sine cycles will be created within the specified start and end anchor points. The example below shows three sine cycles.

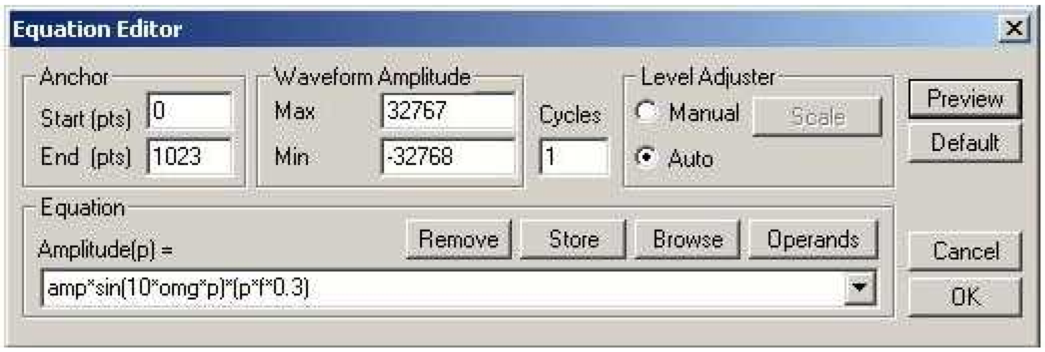
Start Phase – The start phase parameter defines the angle of which the sine will start. The example shows 0° start phase.

Power – Sine to the power of 1 will generate a perfect sine. Power range is from 1 through 9.

Generating Waveforms Using the Equation Editor

One of the most powerful feature within WaveCAD and probably the feature that will be used most is the Equation Editor. The Equation Editor let you write equations the same way as you would do on a blank piece of paper. The equations are then translated to sequential points that form waveforms and are displayed on the waveform screen. The Equation Editor will detect and inform you on syntax errors and, with its self adjusting feature, will automatically adjust your parameters so that none of the points on your waveform will exceed the maximum scale limits.

When you invoke the Equation Editor, the dialog box, as shown in **Figure 4-27** will display. Use the following paragraphs to learn how to use this dialog box and how to write your equations.



Start (pts) 0
End (pts) 1023
Max 32767
Min -32768
Cycles 1
Manual [✓ Auto]
Amplitude(p) =
amp*sin(10*omg*p)*(p*f*0.3)

Figure 4-26, The Equation Editor Dialog Box

There are four sub-group parameters in the equation editor plus control buttons and equation field. These parts are described below.

Anchor

The anchors define start and end point of which the equation will be generated. By default the anchors are placed at the start and the end of the horizontal (time) scale however, the equation can be limited to a specific time scale by moving the anchor points from their default locations.

Start – defines the first point where the created wave will start. Note that if you change the start point the left anchor will automatically adjust itself to the selected start point.

End – defines where the created waveform will end. Note that as you change the end point the right anchor will automatically adjust itself to the selected end point.

Waveform Amplitude

The vertical axis of the Wave Composer represents 16-bits of vertical resolution. That means that the equation is computed, resolved and generated with 1/65,536 increments and accuracy. The Waveform Amplitude fields in the Equation Editor are used in two cases: 1) when the “amp” parameter is used in the equation or 2 if the Level Adjuster is set to Auto. Information on these two operations is given later.

Max – defines the positive peak of the vertical axis

Min – defines the negative peak of the vertical axis

Cycles

The Cycles parameter defines how many waveform cycles will be created within the specified start and end anchor points.

Level Adjuster

The Level Adjuster is a convenient tool that helps you adjust the amplitude and offset without modifying your equation. The Level Adjuster mode does not interfere with your calculations and displays the waveform as computed from your equation. The only difference is that your final calculations are stretched or shrunk or offset on the vertical scale to fit the new amplitude and offset boundaries.

For example, look at the equation that is shown in **Figure 4-26**. This equation will generate the waveform as shown in **Figure 4-23**.

If you change the Max and Min setting in the Waveform Amplitude fields and press the Adjust key, your waveform will offset immediately without changing the equation. The same way, you can also change amplitude only or both amplitude and offset. If you check the Manual option, you’ll have to click on the Adjust button for the Waveform Amplitude parameters to take effect. The Adjust button name will change to Restore and back to Adjust if you click on it again. If you check the Auto option, your waveform will be created automatically with the new Amplitude setting.

Equation

The Equation group has four buttons and the equation field. You will be using the Equation field for writing your equations. Equation syntax and conventions are discussed in the following paragraphs. The *Remove* button clears the equation field so you can start typing a new equation. Click on the *Store* button to store your equation if you intend to use it again. The Browse button provides access to waveform pre-stored files in your computer for combining them in new equations. The *Operands* button expands the bottom of the dialog box to show the operands you can use with your equation.

While you type and store equations, they are collected in a history file and can be used again by expanding the history log from the equation field.

Control Buttons

There are four control buttons at the right corner of the dialog box. Use the *Preview* button to preview an image of your equation, or use the *OK* button to place your waveform on the waveform screen and to leave the dialog box on the screen. The *Default* button restores the parameters in the equation editor to their original factory default values. The *Cancel* button will remove the dialog box from the screen and will discard of any waveforms that you previewed with your Equation Editor.

Writing Equations

The Equation Editor lets you process mathematical expressions and convert them into waveform coordinates. As you probably already know, waveforms are made of vertical samples. The number of samples on your waveform is determined by the wavelength parameter. For example, if you have 1024 horizontal points, your equation will be computed along 1024 points as a function of the vertical scale. Each vertical sample is computed separately and placed along the horizontal axis. The points are graphically connected to form a uniform and continuous waveform shape however, if you zoom in on a waveform line, you'll see that the points are connected like a staircase. In reality, the 3156B generates its waveforms exactly as shown on the screen but, if the waveform has many horizontal points, the steps get smaller and harder to see without magnification.

Equations are always computed as a function of the vertical (Amplitude) axis therefore the left side of your equation will always look as Amplitude(p)=, where “p” is the equation variables in units of waveform points. You can write equations with up to 256 characters. If the equation is too long to fit in the visible field, parts to the left or right will scroll off the ends.

Equation Conventions

The following paragraphs describe the conventions that are used for writing an equation. To avoid errors, it is extremely important that you make yourself familiar with these conventions before you plan your waveforms.

Equations are written in conventional mathematical notation. You may only enter the right part of the equation. The only limitation is that the equation must be of a single variable that is directly related to the current horizontal axis setting. Case is not important and spaces are ignored. Numbers are entered in scientific notation. All calculations are done with double-digit precision. For the trigonometric functions, all angles are expressed in radians.

A number of constants are provided: e, which is the base of the natural logarithm; pi, which is the circumference of a unit-diameter circle; per, which equals the programmed horizontal range; f, which equals 1 /per; omg, which equals 2 * pi / per, and numerals in the range of -1E^20 to 1E^20.

There are three classes of precedence: ^ (raise to power) has the highest precedence; (multiply) and / (divide) come second; + and - have the lowest precedence. Parentheses may be used to change the order of precedence. The following table summarize the mathematical expressions and their respective abbreviated commands that can be used with the Equation Editor.

Equation Editor Operands

^	Raise to the power
*	Multiply
/	Divide
+	Add
Ä	Subtract
( )	Parentheses
e	Base of natural Logarithm
pi (π)	Circumference of unit-diameter circle
per	Horizontal wavelength in points
f	I/per
omg (Ω)	2*π / per
amp	Amplitude in units of points or seconds
sin(x)	The sine of x(*)
cos(x)	The cosine of x
tan(x)	The tangent of x
ctn(x)	The cotangent of x
log(x)	The base IO logarithm of x
In(x)	The natural (base e) logarithm of x
abs(x)	The absolute value of x
-1E^20<>1E^20	Numerals, equation constants

(*)x = argument mathematical expression

After you get familiar with the operands and conventions, you can commence with a few simple equations and see what they do to your waveform screen. Once you'll get the feel, you'll be able to explore your own creativity to generate much more complicated and complex waveforms.

Typing Equations

If you remember from your old high school studies, the simplest curve of Y as a function of X is defined by the equation Y=aX+b. You can use the same “technique” to generate straight lines with the Equation Editor. Assuming first that p=0, try this:

Amplitude(p)=1000

Press [Preview] and see what you get. Of course, you get an uninteresting line that runs parallel to the X-axis. Now, lets give the line some angle by typing:

Amplitude(p)=-2*p+2000

Press [Preview] and see that the line slopes down. It may still be not very interesting however, pay close attention to the convention that is used in this equation. You cannot type: Amplitude(p)=-2p+1000, like you would normally do in your notebook; You must use the * (multiply) sign, otherwise you'll get a syntax error. Now we'll try to generate a simple sine waveform. Try this:

Amplitude(p)=sin(10)

Press [Preview] and… sorry, you still get nothing on the screen. The Wave Composer did not make a mistake! The sine of 10 in radians is exactly what it shows. You are unable to see the result because the line on your screen running across the 0 vertical point.

REMEMBER:

The equation must be a function of a single variable and that variable must be directly related to the Horizontal axis Scale setting.

Now try this:

Amplitude(p)=sin(omg*p)

Still no good, but now press the [Adjust] button and here is your sinewave. So what's wrong? Well, if you'll give it a little amplitude it might help so, do it now exactly as follows:

Amplitude(p)=24000*sin(omg*p)

There you go. You should now see a perfect sine waveform with a period of 1000 points. This is because you have asked the Equation Editor to compute .the sine along p points (“p” is the equation variable, remember?). If you want to create 10 sine waveforms, you should multiply p by 10. Try this:

Amplitude(p)=6000*sin(omg*p*10)

Equation Samples

So far, you have learned how to create two simple waveforms: straight lines and trigonometric functions. Let’s see if we can combine these waveforms to something more interesting. Take the straight line equation and add it to the sinewave equation:

Amplitude(p)=24000*sin(omg*p*l0)-8*p+8000

Press [Preview]. Your screen should look like **Figure 4-27**.

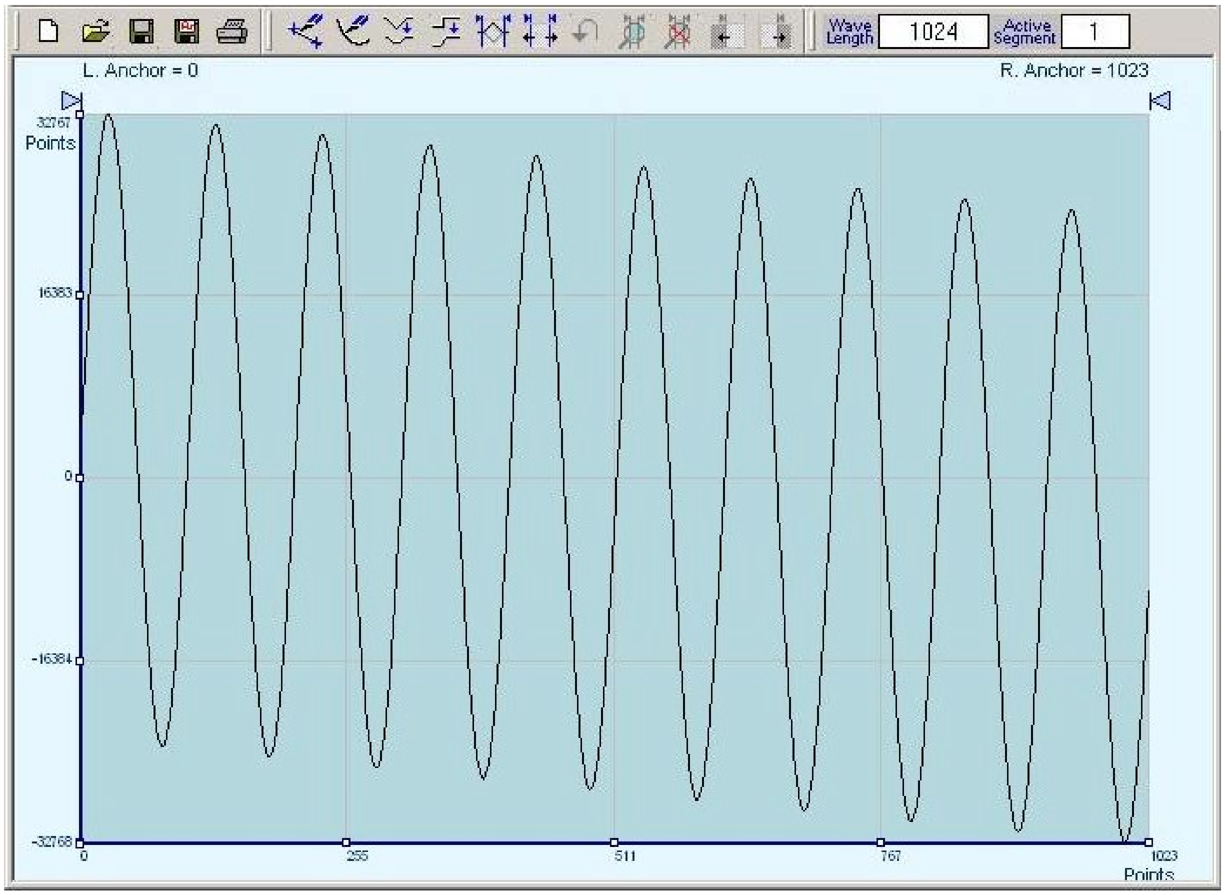


Figure 4-27, An Equation Editor Example

Now let’s try to modulate two sinewaves with different periods and different start phase. Type this:

Amplitude(p)= 32000*sin(omg*p)*cos(omg*p*30)

Press [Preview]. Your screen should look like **Figure 4-28**.

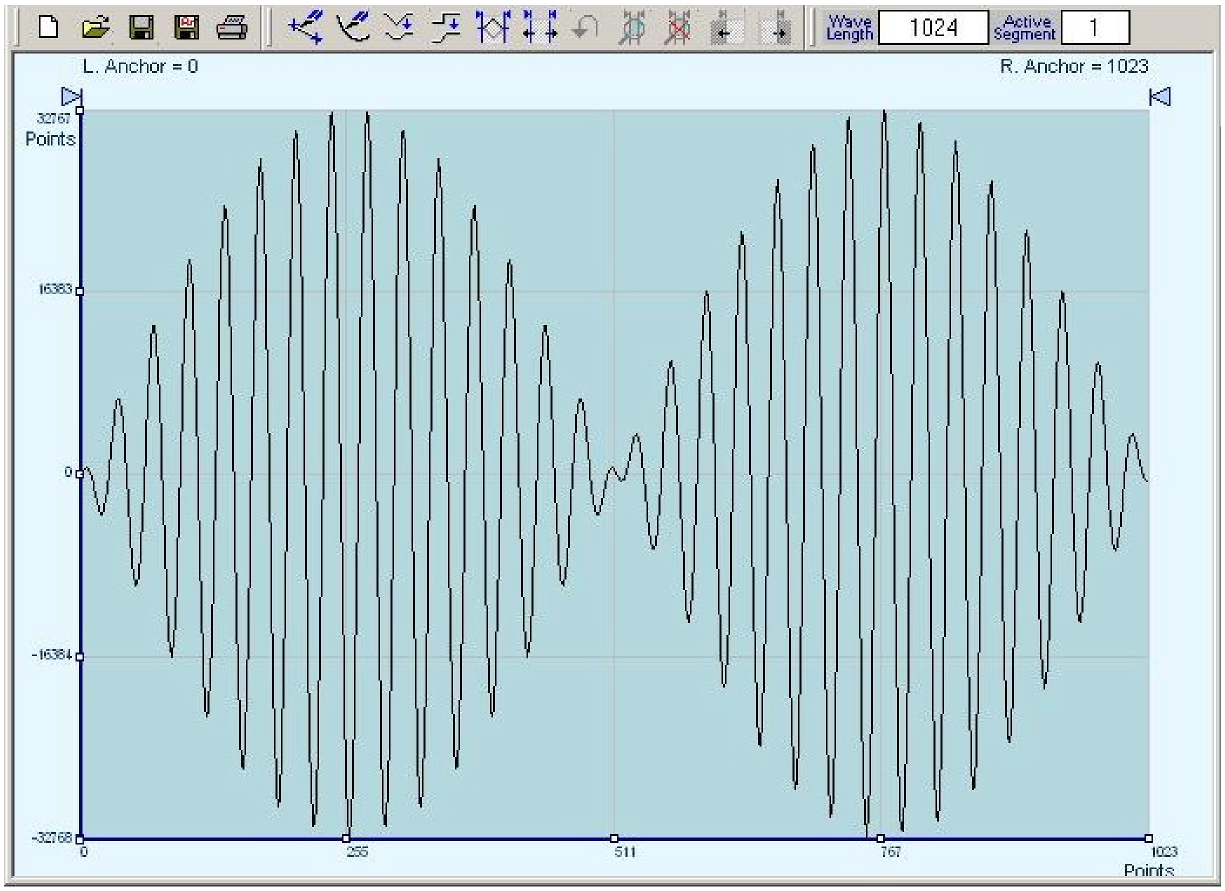


Figure 4-28, Using the Equation Editor to Modulate Sine Waveforms

In the following example, as shown in **Figure 4-29**, 20% second harmonic distortion has been added to a standard sinewave. The original waveform had a peak-to-peak value of 20000 points so 20% second harmonic is equivalent to 4000 points. The frequency of the second harmonic is obviously double that of the fundamental, so term +4000*sin(2*omg*p) is added to the original sine wave equation. Use the following equation:

Amplitude(p)=20000*sin(omg*p)+4000*sine(2*omg*p)

Press [Preview]. Your screen should look like **Figure 4-29**.

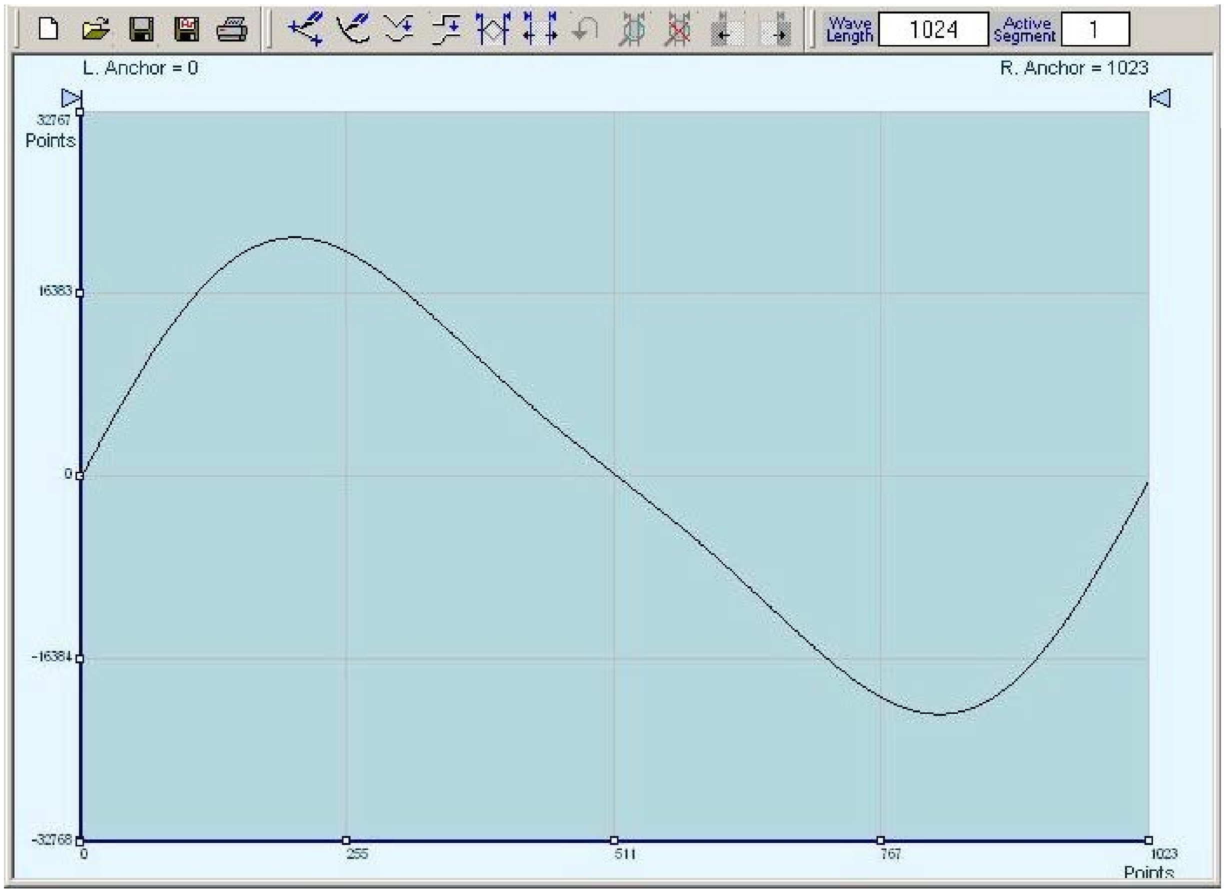


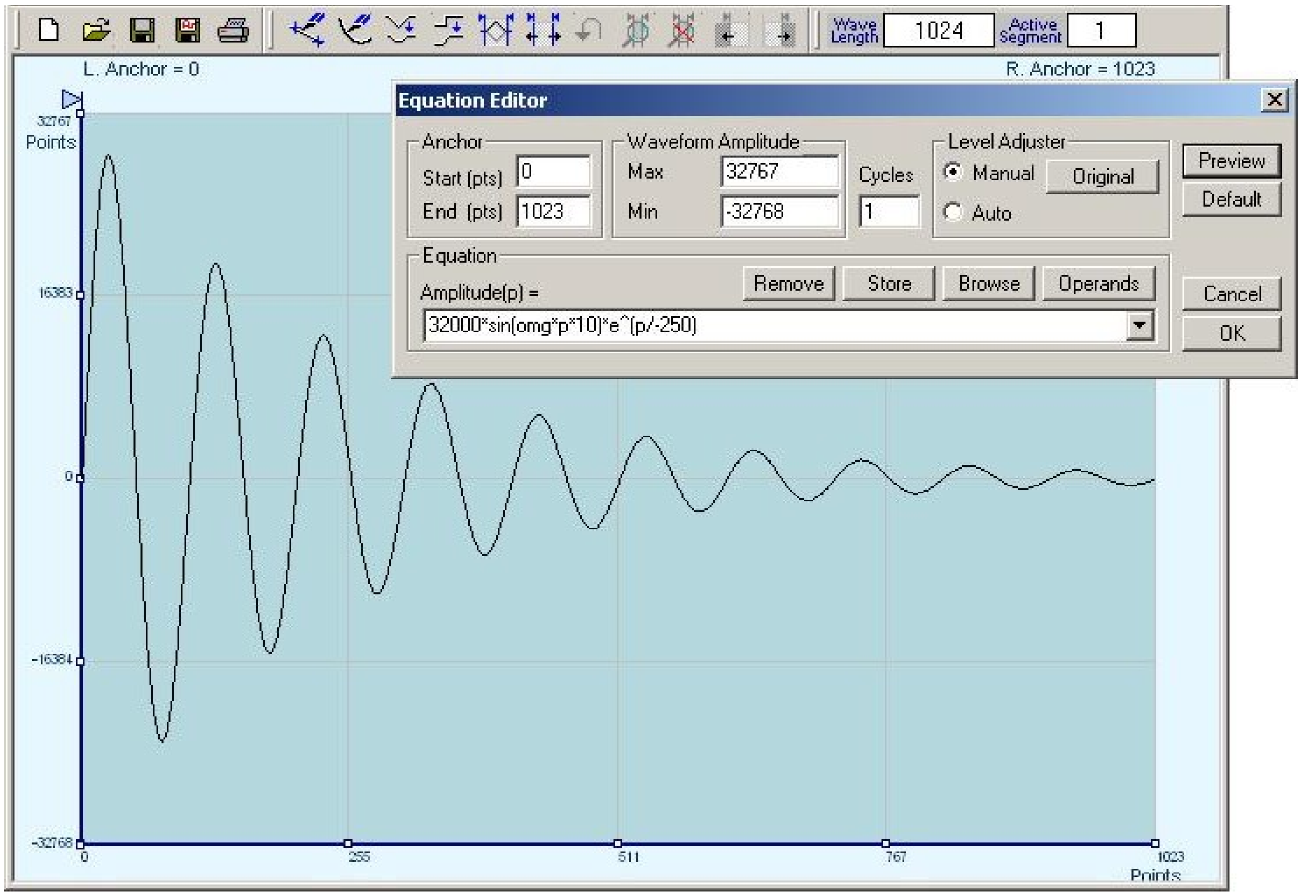
Figure 4-29, Using the Equation Editor to Add Second Harmonic Distortion.

In **Figure 4-30** we created 10 cycles of sinewave made to decay exponentially. The original expression for a standard sinewave is multiplied by the term e^(-p/250). Increasing the value of the divisor (200 in this case) will slow down the rate of decay.

Use the following equation:

Amplitude(p)=32000*sin(omg*p*10)^e(p/-250)

Press [Preview]. Your screen should look like **Figure 4-30**.



Wave Length: 1024
Active Segment: 1

Anchor
Start (pts): 0
End (pts): 1023

Waveform Amplitude
Max: 32767
Min: -32768

Cycles: 1

Level Adjuster
[✓ Manual] Auto

Equation
Amplitude(p) =
32000*sin(omg*p*10)*e^(p/-250)

Figure 4-30, Using the Equation Editor to Generate Exponentially Decaying Sinewave

The last example as shown in **Figure 4-31** is the most complex to be discussed here. Here, 100 cycles of sinewave are amplitude modulated with 10 cycles of sine wave with a modulation depth of 20%. To achieve this, the upper and lower sidebands are defined separately and added to the fundamental or carrier. The upper sideband is produced by the expression 100*cos(110*omg*p) and the lower sideband by the term 100*cos(90*omg*p).

Use the following equation:

Ampl(p)=20000*sin(100*omg*p)+4000*cos(110*omg*p)-4000*cos(90*omg*p)

Press [Preview]. Your screen should look like **Figure 4-31**.

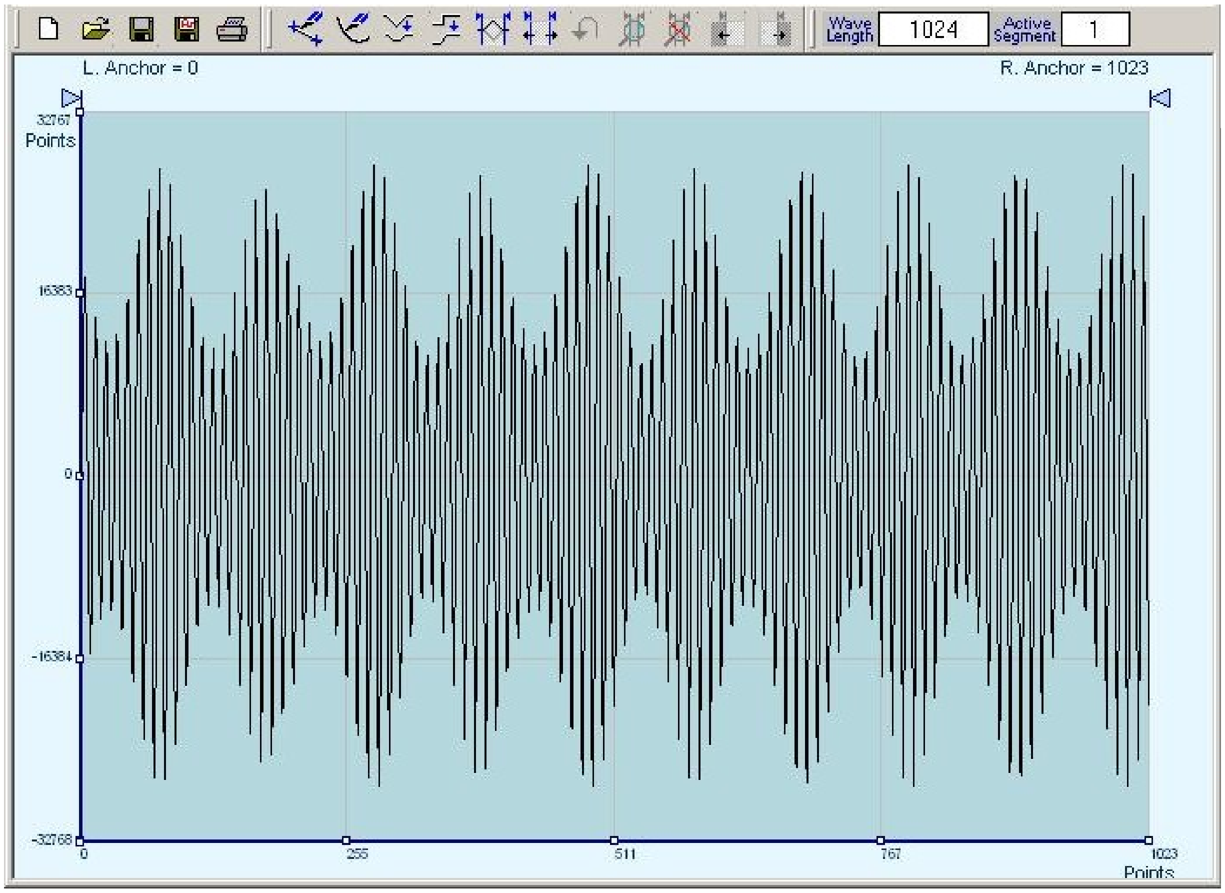


Figure 4-31, Using the Editor to Build Amplitude Modulated Signal With Upper and Lower Sidebands

Combining Waveforms

The last but not least powerful feature allows you to combine waveforms which you previously stored on your hard disc. You can write mathematical expressions that contain waveforms, simple operands and trigonometric functions similar to the example given below. If you want to use waveforms in your equations, you must first generate these waves and store them on your hard disk. You identify waveforms by adding the *.wav extension as shown in the example below.

Amplitude(p)= Sine.wav*sin(omg*p*10)*Noise.wav/1000

The above equation will generate amplitude-modulated waveform with added noise. The following steps demonstrate how to create, store and combine waveforms using this equation.

Step 1 – Create and store sine.wav. Invoke the Equation Editor and type the following equation:

Ampl(p)= 40000*sin(omg*p)

Press OK and then select the Save Waveform As… from the File command. Save this file using the name Sine.wav

Step 2 – Create and store Noise.wav. From the Wave command select Noise. Click OK and watch your waveform screen draw noisy signal. From the File menu select Save Waveform As… and save this waveform using the name Noise.wav.

Step 3 – Write and compute the original equation:

Amplitude(p)= Sine.wav*sin(omg*p*5)*Noise.wav/16000

If you did not make any mistakes, your waveform screen should look as shown in **Figure 4-32**.

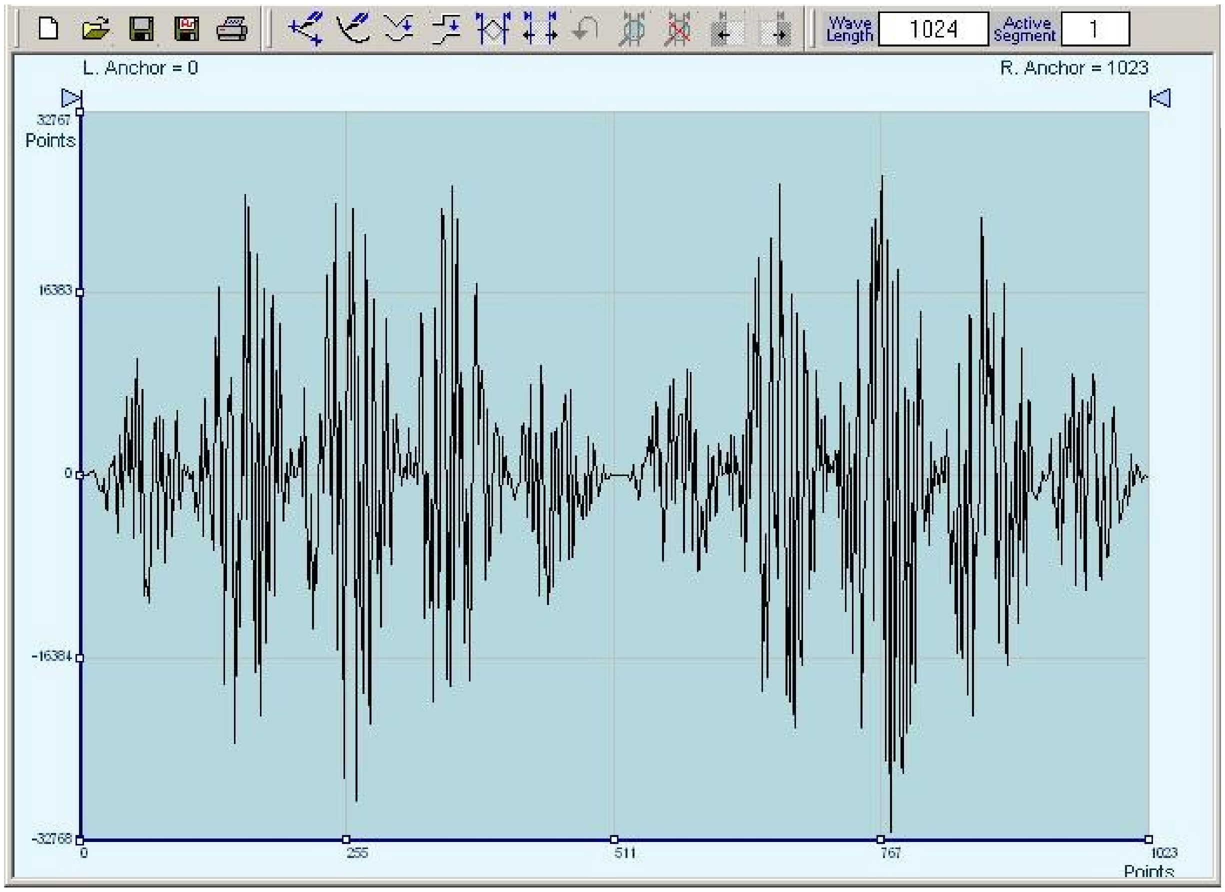


Figure 4-32, Combining Waveforms into Equations

Chapter 5
PROGRAMMING REFERENCE

What's in This Chapter

This Chapter lists and describes the set of driver function calls that are used to operate the 3156B. To provide familiar formatting for users who have previously used such reference documentation, function call descriptions are dealt with in a similar manner. In general, each group starts with a short description, followed by a table showing the complete set of function calls in the group; finally the effects of individual function and its arguments are described. Complete listing of all driver function calls that are used to program the 3156B is given in **Table 5-1**.

Introduction

This instrument driver provides programming support for 3156B Arbitrary Waveform Generator. It contains functions for opening, configuring, loading and generating waveforms, and closing the instrument.

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

1. The instrument is installed in the VXI mainframe and you are using one of the following controller options:

   Embedded controller

   MXI

   MXI2
2. The logical address supplied to the initialize function must match the logical address of the instrument.
3. The 3156B driver is installed on your controller

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. The general meaning of the status code is as follows:

Value	Meaning
0	Success
Positive Values	Warnings
Negative Values	Errors

Below is a description of possible error codes and their interpretation. Your program should examine the status code from each call to an instrument driver function to determine if an error occurred.

VISA Errors

These error codes are defined by the VISA I/O library function standards. These errors are defined in the header file "visa.h". These error codes are typically in the range 0xBFFF0000L to 0xBFFFFFFFL (-1073807360 to -1073741825).

RI3156B Driver Errors

Errors returned from the RI3156B drivers be between 0xBFFC0800 and 0xBFFC0FFF are described in the source code for ri3156B_error_message() function. This function may be used to convert the error code to a meaningful error message.

How to Use This Document

Use this document as a programming reference manual. It describes each function in the 3156B Arbitrary Waveform Generator instrument. The functions appear in grouped order, with a description of the function and its C syntax, a description of each parameter, and a list of possible error codes.

Table 5-1, Driver Function Calls Summary

3156B Function Name	Parameter(s)	Range	Default
Initialization			
ri3156B_init	ViRsrc instrDescriptor	2 to 256	2
	ViBoolean IDQuery	0, 1 (OFF, ON)	
	ViBoolean resetDevice	0, 1 (OFF, ON)	
	ViSession *instrHandle		VXI init
ri3156B_reset	None		Default
ri3156B_close	None		
Common Control Functions			
ri3156B_set(query)_active_channel	ViInt16 channel	1 to 2	1
ri3156B_set(query)_output	ViBoolean outputSwitch	0, 1 (OFF, ON)	0
ri3156B_set(query)_enable	ViBoolean outputSwitch	0, 1 (OFF, ON)	0
ri3156B_set(query)_enable_source	ViInt16 enableSource	0 to 2 (SOFT, HARD, MIX)	0
ri3156B_set(query)_amplitude	ViReal64 amplitude	10e-3 to 10	5
ri3156B_set(query)_offset	ViReal64 offset	-4.995 to 4.995	0
ri3156B_set(query)_global_operating_mode	ViInt16 ch1_waveformMode	0 to -5 (STD, ARB, SEQ, MOD, VIDEO, DIGITAL)	0
	ViInt16 ch2_waveformMode	0 to 5 (STD, ARB, SEQ, MOD, VIDEO, DIGITAL)	0
	ViInt16 CarrierRunMode	0 to 3 (CONT, TRIG, GATE, BURST)	0

Table 5-2, Driver Function Calls Summary (continued)

3156B Function Name	Parameter(s)	Range	Default
	ViInt16 OperateEnableSource	0, 1 (SOFT, HARD, MIX)	0
ri3156B_set(query)_common_waveform_mode	ViInt16 commonWaveformMode	0 to 5 (STD, ARB, SEQ, MOD, VIDEO, DIGITAL)	0
ri3156B_set(query)_waveform_mode	ViInt16 waveformMode	0 to 5 (STD, ARB, SEQ, MOD, VIDEO, DIGITAL)	0
ri3156B_set(query)_carrier_run_mode	ViInt16 CarrierRunMode	0 to 3 (CONT, TRIG, GATE, BURST)	0
ri3156B_set(query)_modulation_mode	ViInt16 modulationMode	0 to 5 (OFF, FM, AM, FSK, SWEEP, HOP)	0
ri3156B_set(query)_modulation_run_mode	ViInt16 modulationRunMode	0 to 3 (CONT, TRIG, GATE, BURST)	0
ri3156B_set(query)_digital_mode	ViBoolean digitalMode	0, 1 (FREERUN, STIMULUS)	0
ri3156B_set(query)_reference_oscillator	ViInt16 referenceOscillator	0, 1 (INT (CLK10 or TCXO), EXT)	0
Standard Waveforms Programming			
ri3156B_set(query)_standard_waveform	ViInt16 standardWaveform	0 to 9 (Sine, Triangle, Square, Pulse, Ramp, SINC, Exp, Gaussian, DC	0
ri3156B_set(query)_frequency	ViReal64 frequency	0.01 to 25e6	1e6
ri3156B_query_std_sample_clock_freq	ViPReal64 stdSampleClockFrequency		
ri3156B_query_std_waveform_numb_points	ViPInt16 standardWaveformPoints		
Sine Wave Functions			
ri3156B_set(query)_sine_wave_phase	ViReal64 phase	0 to 359.95	0
ri3156B_apply_sine_wave	ViReal64 frequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 phase	0 to 359.95	0
Triangle Wave Functions			
ri3156B_set(query)_triangular_wave_phase	ViReal64 phase	0 to 359.95	0
ri3156B_apply_triangular_wave	ViReal64 frequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 phase	0 to 359.95	0
Square Wave Functions			
ri3156B_set(query)_square_wave_duty_cycle	ViReal64 dutyCycle	0 to 99.99	50
ri3156B_apply_square_wave	ViReal64 frequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 dutyCycle	0 to 99.99	50

Table 5-3, Driver Function Calls Summary (continued)

3156B Function Name	Parameter(s)	Range	Default
Half Cycle Wave Functions			
ri3156B_set(query)_half_cycle_mode	ViBoolean HalfCycleMode	0, 1 (OFF, ON)	0
ri3156B_set(query)_half_cycle_delay	ViReal64 HalfCycleDelay	0, 500e-9 to 21	1e-6
ri3156B_apply_half_cycle_sine_wave	ViReal64 frequency	0.01 to 1e6	500e3
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 delay	0, 500e-9 to 21	1e-6
	ViInt16 Waveform	0, 2 (Sine, Triangle, Square)	0
	ViReal64 phase	0 to 359.95	0
	ViReal64 dutyCycle	0 to 99.99	50
Pulse Wave Functions			
ri3156B_set(query)_pulse_wave_PRF	ViReal64 pulsePRF	0.01 to 25e6	1e6
ri3156B_set(query)_pulse_wave_high_time	ViReal64 highTime	0 to 99.99	10
ri3156B_set(query)_pulse_wave_delay	ViReal64 delayTime	0 to 99.99	10
ri3156B_set(query)_pulse_wave_rise_time	ViReal64 riseTime	0 to 99.99	10
ri3156B_set(query)_pulse_wave_fall_time	ViReal64 fallTime	0 to 99.99	10
ri3156B_apply_pulse_wave	ViReal64 pulsePRF	0.01 to 25e6	1e
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 highTime	0 to 99.99	10
	ViReal64 delayTime	0 to 99.99	10
	ViReal64 riseTime	0 to 99.99	10
	ViReal64 fallTime	0 to 99.99	10
Ramp Wave Functions			
ri3156B_set(query)_ramp_wave_slope	ViReal64 rampSlope	0 to 99.99	10
ri3156B_apply_ramp_wave	ViReal64 frequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 rampSlope	0 to 99.99	10
Sinc Wave Functions			
ri3156B_set(query)_sinc_wave_num_cycles	ViInt16 numberofCycles	4 to 100	10
ri3156B_apply_sinc_wave	ViReal64 frequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViInt16 numberofCycles	4 to 100	10
Exponential Wave Functions			
ri3156B_set(query)_exponential_wave_exponent	ViInt16 exponent	-100 to 100	-1
ri3156B_apply_exponential_wave	ViReal64 frequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViInt16 exponent	-100 to 100	-1

Table 5-4, Driver Function Calls Summary (continued)

3156B Function Name	Parameter(s)	Range	Default
Gaussian Wave Functions			
ri3156B_set(query)_gaussian_wave_exponent	ViInt16 exponent	10 to 200	10
ri3156B_apply_gaussian_wave	ViReal64 frequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10mV to 10V	5
	ViReal64 offset	-4.995 to 4.995	0
	ViInt16 exponent	10 to 200	10
DC Signal Functions			
ri3156B_set(query)_dc_signal_percent	ViReal64 percentAmplitude	-100 to 100	100
ri3156B_apply_dc_signal	ViReal64 percentAmplitude	-100 to 100	100
Arbitrary Waveforms Programming			
ri3156B_set(query)_arb_sampling_freq_range	ViBoolean samplingClockRange	0, 1	0
ri3156B_set(query)_arb_sampling_freq	ViReal64 samplingClock	1 to 200	50e6
Arbitrary Waveform Functions			
ri3156B_define_arb_segment	ViInt16 segmentNumber	1 to 16k	
	ViInt32 segmentSize	1 to 512e3	
		2 to 1e6	
ri3156B_delete_segment	ViInt16 segmentNumber	1 to 16k	
ri3156B_set(query)_active_segment	ViInt16 segmentNumber	1 to 16k	1
ri3156B_load_arb_data	ViInt16 segmentNumber	1 to 16k	
	ViInt16 dataPointArray[]	(array of 12 or 16 bit data)	
	ViInt32 numberofPoints	1 to 512e3	
		2 to 1e6	
ri3156B_load_segment_table	ViInt 16numberOfSegments	1 to 16k	
	ViInt32 waveSize[]	1 to 512e3	
		2 to 1e6	
ri3156B_load_ascii_file	ViInt16 segmentNumber	1 to 16k	
	ViString fileName		
	ViInt32 numberofPoints	1 to 512e3	
		2 to 1e6	
ri3156B_set(query)_wave_format	ViBoolean waveFormat	0, 1 (16-bit or 12-bit)	0
ri3156B_set(query)_byte_order	ViBoolean byteOrder	0, 1 (Hi-Lo, Lo Hi)	0
ri3156B_apply_arb_waveform	ViInt16 segmentNumber	1 to 16k	1
	ViReal64 samplingClock	1 to 200e6	50e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0

Table 5-5, Driver Function Calls Summary (continued)

3156B Function Name	Parameter(s)	Range	Default
Sequenced Waveforms Programming			
ri3156B_define_sequence_step	ViInt16 numberofStep	1 to 4096	1
	ViInt16 segmentNumber	1 to 16k	
	ViInt32 repeatSegment	1 to 1e6	
ri3156B_set(query)_sequence_mode	ViInt16 sequenceMode	0 to 2 (AUTO, STEP, SINGLE)	0
ri3156B_delete_sequence_step	ViInt16 sequenceStep	0, 1 to 4096 (0 means delete all)	
ri3156B_load_sequence_table	ViInt16 numberofSteps	1 to 4096	1
	ViInt16 segment[]	array of values 1 to 16k	
	ViInt32 repetitions[]	array of values 1 to 1Meg	
ri3156B_apply_sequence_waveform	ViReal64 samplingClock	1 to 200e6	50e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViInt16 sequenceMode	0 to 2	0
Modulated Waveforms Programming			
Sweep Programming			
ri3156B_set(query)_sweep_type	ViInt16 sweepType	0, 1 (LIN, LOG)	0
ri3156B_set(query)_sweep_start	ViReal64 startFrequency	0.01 to 25e6	10e3
ri3156B_set(query)_sweep_stop	ViReal64 stopFrequency	0.01 to 25e6	1e6
ri3156B_set(query)_sweep_range	ViReal64 startFrequency	0.01 to 25e6	10e3
	ViReal64 stopFrequency	0.01 to 25e6	1e6
ri3156B_set(query)_sweep_time	ViReal64 sweepTime	1.4e-6 to 40	1
ri3156B_set(query)_sweep_direction	ViInt16 SweepDirection	0, 1 (UP,DOWN)	0
ri3156B_set(query)_sweep_marker	ViReal64 markerFrequency	<sweep range>	(stop-start)/2
ri3156B_apply_sweep_waveform	ViReal64 startFrequency	0.01 to 25e6	10e3
	ViReal64 stopFrequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViInt16 sweepType	0, 1 (LIN, LOG)	0
	ViReal64 sweepTime	1.4e-6 to 40	1
	ViInt16 SweepDirection	0, 1 (UP,DOWN)	0
	ViReal64 markerFrequency	<sweep range>	(stop-start)/2
FM Programming			
ri3156B_set(query)_FM_carrier_freq	ViReal64 FMcarrierFrequency	0.01 to 25e6	1e6
ri3156B_set(query)_FM_mod_freq	ViReal64 FMmodulationFrequency	0.01 to 350e3	10e3
ri3156B_set(query)_FM_mod_waveform	ViInt16 FMmodulationWaveform	0 to 4 (Sine,Triangle,Ramp, Square, Arb)	0

Table 5-6, Driver Function Calls Summary (continued)

3156B Function Name	**Parameter(s)**	**Range**	**Default**
ri3156B_set(query)_FM_freq_dev	ViReal64 FMfrequencyDeviation	< carrier frequency	100e3
ri3156B_set(query)_FM_marker	ViReal64 FMmarkerFrequency	< deviation frequency>	Carrier freq
ri3156B_apply_FM_waveform	ViReal64 FMcarrierFrequency	0.01 to 25e6	1e6
	ViReal64 Amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 FMmodulationFrequency	0.01 to 350e3	10e3
	ViReal64 FMfrequencyDeviation	< carrier frequency	100e3
	ViInt15 FMmodulationWaveform	0 to 4	0
	ViReal64 FMmarkerFrequency	<carrier+/- deviation frequency/2	Carrier freq
ri3156B_set(query)_arb_FM_mod_sclk	ViReal64 FMarbModulationSCLK	1 to 5e6	1e6
ri3156B_load_arb_FM_mod_data	ViReal64 FMdataPointArray[]	array of values 0.01 to 25e6	
	ViInt32 FMnumberofPoints	10 to 32768	
ri3156B_apply_arb_FM_waveform	ViReal64 FMarbModulationSCLK	1 to 5e6	1e6
	ViReal64 Amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
Frequency Hopping Programming			
ri3156B_set(query)_hop_mode	ViBoolean hopMode	0,1	0
ri3156B_set(query)_hop_dwell_time	ViReal64 hopDwellTime	500e-9 to 21	500e-9
ri3156B_set(query)_hop_marker	ViInt16 hopMarkerIndex	0 to 4095	Last hop
ri3156B_load_fix_hop_freq_list	ViReal64 hopFreqList[]	array of values 0.01 to 25e6	
	ViInt16 hopFreqListSize	Hopping frequency list length 1 to 4096	
ri3156B_load_var_hop_freq_list	ViReal64 hopFreqList[]	array of values 0.01 to 25e6	
	ViInt16 hopFreqListSize	Hopping frequency list length 1 to 4096	
	ViReal64 hopDwellTimeList[]	array of dwell time list	0=default/prog dwell time
ri3156B_apply_hop_waveform	ViReal64 hopMode	0,1	0
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 hopDwellTime	500e-9 to 21	500e-9
	ViInt16 hopMarkerIndex	0 to 4095	Last hop

Table 5-7, Driver Function Calls Summary (continued)

3156B Function Name	Parameter(s)	Range	Default
FSK Programming			
ri3156B_set(query)_FSK_one_frequency	ViReal64 FSKoneFrequency	0.01 to 25e6	1e6
ri3156B_set(query)_FSK_zero_frequency	ViReal64 FSKzeroFrequency	0.01 to 25e6	100e3
ri3156B_load_FSK_data	ViInt16 FSKwordLength	8 to 4096	8
	ViBoolean FSKdata[]	0, 1	
ri3156B_set(query)_FSK_word_rate	ViReal64 FSKbaudRate	1 to 10e6	10e3
ri3156B_set(query)_FSK_marker	ViInt16 FSKmarkerIndex	1 to 4096	Last hop
ri3156B_apply_FSK_waveform	ViReal64 FSKbaudRate	1 to 10e6	10e3
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 FSKoneFrequency	0.01 to 25e6	1e6
	ViReal64 FSKzeroFrequency	0.01 to 25e6	100e3
	ViInt16 FSKmarkerIndex	1 to 4096	last hop
AM Programming			
ri3156B_set(query)_AM_carrier_freq	ViReal64 AMcarrierFrequency	0.01 to 25e6	1e6
ri3156B_set(query)_AM_mod_freq	ViReal64 AMmodulationFrequency	0.01 to 100e3	10e3
ri3156B_set(query)_AM_mod_freq_div	ViInt16 AMmodulationFreqDiv	2 to 4096	100
ri3156B_set(query)_AM_mod_depth	ViReal64 AMmodulationDepth	0 to 100	50
ri3156B_apply_AM_waveform	ViReal64 AMcarrierFrequency	0.01 to 25e6	1e6
	ViReal64 Amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViInt16 AMmodulationFreqDiv	2 to 4096	100
	ViReal64 AMmodulationDepth	0 to 100	50
Patterns - Free Running Programming			
ri3156B_set(query)_dig_pattern_rate_range	ViInt16 digitalPatternRateRange	0, 1 (100Mpps, 50Mpps)	100e6
ri3156B_set(query)_dig_pattern_rate	ViReal64 digitalPatternRate	1 to 50/100e6	10e3
ri3156B_load_dig_pattern_stim_list	ViInt16 digitalPatternStimList[]	array of values 0 to 0xFFF	
	ViInt32 digPatternStimListSize	Digital Pattern Stim List length, 1 to 512k	
	ViInt32 digitalPatternHoldCountList[]	array of values 1 to 1.049B < 50MS/s, 2 to 2.1B 50MS/s-100MS/s	
ri3156B_apply_digital_pattern	ViReal64 digitalPatternRate	1 to 50/100e6	10e3
	ViInt16 digitalPatternRateRange	0, 1	0
Patterns - Stimulus Programming			
ri3156B_set(query)_dig_stim_freq	ViReal64 digitalDataFrequency	1 to 100e6	10e3
ri3156B_load_dig_data_stim_list	ViInt16 digitalDataStimList[]	array of values 0-0xFFF	
	ViInt32 digitalDataStimListSize	Digital Data Stim List Length. 1 to 512e3	
ri3156B_apply_digital_data	ViReal64 digitalDataFrequency	1 to 100e6	10e3

Table 5-8, Driver Function Calls Summary (continued)

3156B Function Name	**Parameter(s)**	**Range**	**Default**
Video Stroke Generator Programming			
ri3156B_set(query)_video_stroke_point_freq	ViReal64 videoStrokePointFrequency	1 to 100e6	10e3
ri3156B_set(query)_video_offset_start	ViReal64 videoStrokeOffsetStart	-4.995 to 4.995	0
ri3156B_set(query)_video_offset_stop	ViReal64 videoStrokeOffsetStop	-4.995 to 4.995	0
ri3156B_set(query)_video_offset_range	ViReal64 videoStrokeOffsetStart	-4.995 to 4.995	0
	ViReal64 videoStrokeOffsetStop	-4.995 to 4.995	0
ri3156B_set(query)_video_offset_step	ViReal64 videoStrokeOffsetStep	±1m to ±9.99	1e-3
ri3156B_set(query)_video_stroke_circ_type	ViBoolean videoStrokeCircType	0, 1 (SINGLE, CONTINUOUS)	0
ri3156B_set(query)_video_character	ViInt16 videoStrokeCharacter	0 to 9 (CROSS-LOCATOR, CROSS-HAIR, POSITIONED-SQUARE, VERTICAL-MARKER-LINE, HORIZONTAL-MARKER-LINE, RIGHT-HAND-ARROW, LEFT-HAND-ARROW, DIAMOND-OVERLAY, INVERTED-TRIANGLE, UPRIGHT-TRIANGLE)	0
ri3156B_set(query)_active_video_pat_number	ViInt16 ActivevideoStrokePatternNumber	1 to 16k	1
ri3156B_load_video_str_pattern_data	ViInt16 videoStrokePatternNumber	1 to 16k	1
	ViInt16 dataPointArray_ch1[]	array of 16 bit data	
	ViInt16 dataPointArray_ch2[]	array of 16 bit data	
	ViInt32 videoStrokePatternSize	1 to 512k	1
ri3156B_apply_video_str_character	ViReal64 videoStrokePointFrequency	1 to 100e6	10e3
	ViReal64 videoStrokeOffsetStart	-4.995 to 4.995	0
	ViReal64 videoStrokeOffsetStop	-4.995 to 4.995	0
	ViReal64 videoStrokeOffsetStep	±1e-3 to ±9.99	1e-3
	ViBoolean videoStrokeCircType	0, 1	0
	ViInt16 videoStrokeCharacter	0 to 9	0
ri3156B_apply_video_stroke_pattern	ViReal64 videoStrokePointFrequency	1 to 100e6	10e3
	ViReal64 videoStrokeOffsetStart	-4.995 to 4.995	0
	ViReal64 videoStrokeOffsetStop	-4.995 to 4.995	0
	ViReal64 videoStrokeOffsetStep	±1e-3 to ±9.99	1e-3
	ViBoolean videoStrokeCircType	0, 1	0
	ViInt16 videoStrokePatternNumber	1 to 16k	1

Table 5-9, Driver Function Calls Summary (continued)

3156B Function Name	Parameter(s)	Range	Default
Trigger Inputs and Outputs			
ri3156B_set(query)_trigger_source	ViInt16 triggerSource	0 to9 (EXT, TTLTrg0-7, ECLT0)	0
ri3156B_set(query)_trigger_delay	ViReal64 triggerDelay	500e-9 to 21	500e-9
ri3156B_set(query)_trigger_delay_state	ViBoolean triggerDelayState	0, 1 (OFF,ON)	0
ri3156B_set(query)_burst_mode_cycles	ViInt32 numberofCycles	1 to 1e6	1
ri3156B_set(query)_Mod_burst_mode_cycles	ViInt32 ModulatNumberofCycles	1 to 1e6	1
ri3156B_set(query)_trigger_slope	ViBoolean triggerSlope	0, 1 (POS, NEG)	0
ri3156B_set(query)_trigger_level	ViReal64 triggerLevel	-5 to +5	1.6
ri3156B_set(query)_re_trigger_delay_state	ViBoolean retriggerDelayState	0, 1 (OFF,ON)	0
ri3156B_set(query)_re_trigger_delay	ViReal64 retriggerDelay	500e-9 to 21	500 ns
Sync Outputs			
ri3156B_set(query)_sync_output_type	ViInt16 SYNCPulseType	0, 1 PULSE, ZERO-CROSS	0
ri3156B_set(query)_sync_output_state	ViBoolean syncState	0, 1 (OFF, ON)	0
ri3156B set(query)_TTLTRG_n_output_state	VIBoolean TTLTRG_out_state	0, 1 (OFF, ON)	0
	VIInt 16 TTLTRG_n	0 to 7 (TTLTRG0-7)	0
WaveCAD Support			
ri3156B_load_wavecad_wave_file	ViInt16 segmentNumber	1 to 4096	
	ViString WaveCadWaveformFileName		
	ViInt32 segmentSize		
	ViBoolean fileResolution	0, 1 (16-bit,12-bit)	
ri3156B_load_wavecad_FM_wave_file	ViString WaveCadFM_DataFileName	fm frequency values - 0.01 to 25e6	
	ViInt32 FMnumberOfPoints	10 to 32768	
ri3156B_load_wavecad_HOP_freq_list_file	ViString WaveCadHopDataFileName	hop frequency values - 0.01 to 25e6 dwell time values - 500e-9 to 21	
	ViInt16 hopFreqListSize	Hopping frequency list length 1 to 4096	
ri3156B_load_wavecad_FSK_data_file	ViString WaveCadFskDataFileName	fsk data - 0-1	
	ViInt16 FSKwordLength	8 to 4096	
ri3156B_load_wavecad_dig_patt_stim_list_file	ViString WaveCadDigitalPatternDataFileName	digital stim list values - 0-0xFFF hold count values - 1-1.049B < 50MS/s, 54-2.1B 50MS/s-100MS/s	
	ViInt32 digPatternStimListSize	Digital Pattern Stim List length, 1-512k	

Table 5-10, Driver Function Calls Summary (continued)

3156B Function Name	Parameter(s)	Range	Default
ri3156B_load_wavecad_video_data_file	ViInt16 videoStrokePatternNumber	1 to 16e3	
	ViString VideoDataCh1FileName		
	ViString VideoDataCh2FileName		
	ViInt32 videoStrokePatternSize	1 to 512e3	
Utility Functions			
ri3156B_clear	None		
ri3156B_get_option	ViInt16 optionInstalled	0, 1 (CLK10,TCXO)	0
ri3156B_revision_query	ViChar driverRevision[]		
	ViChar firmwareRevision[]		
ri3156B_error_query	ViInt32 error		
	ViChar errorMessage		
ri3156B_error_message	ViStatus errorReturnValue		
	ViChar errorMessage[]		
Ri3156B_read_serial_number	ViChar SerialNumber		
Ri3156B_read_last_cal_date	ViChar LastCalDate		
Ri3156B_query_addr_space	ViInt16 *addrSpace	0, 1 (A24,A32)	0

The Initialization Group

This group is used to initialize the 3156B. The initialization process requires instrument identification, assigning a handle and resetting the instrument to its default state. Factory defaults after reset are given as well. The communication session is closed with the close function. Parameters range is given where applicable.

Initialization			
3156B Function Name	**Parameter(s)**	**Range**	**Default**
ri3156B_init	ViRsrc instrDescriptor	2-256	2
	ViBoolean IDQuery	0-1 (OFF, ON)	
	ViBoolean resetDevice	0-1 (OFF, ON)	
	ViSession *instrHandle		VXI init
ri3156B_reset	None		Default
ri3156B_close	None		

ri3156b_init

Description

Initializes the instrument and returns and "instrument handle". The instrument handle must be used with all of the other functions of this driver. The initialize call allows the instrument to be queried to ensure that it is a 3156B Waveform Synthesizer. It also resets the 3156B to the power-up state if the "Reset" parameter is True (ON). Note that for each "ri3156B_init()" call, a new unique instrument handle is returned. Thus, if four calls are made to the initialize call in succession, four unique instrument handles will be returned. This driver supports 10 instances of instrument handles, so that if this function is called 11 times consecutively (with no call to ri3156B_close()),the 11th call will fail.

C Syntax

ViStatus ri3156b_init (ViRsrc instrDescriptor, ViBoolean IDQuery, resetDevice, ViSession instrHandle)

Parameters

Name	Variable Type	Description
instrDescriptor	ViRsrc	Specifies which remote instrument to establish a communication session with. Based on the syntax of the Instr Descriptor, the Initialize function configures the I/O interface and generates an Instr Handle Default Value: "VXI::2::INSTR" Based on the Instrument Descriptor, this operation establishes a communication session with a device. The grammar for the Instrument Descriptor is shown below. Optional parameters are shown in square brackets ([]). The default value is for a VXI interface for logical address 2. For a GPIB-VXI interface with the instrument set to logical address 2, the value should be: "GPIB-VXI::2::INSTR" *Interface Grammar* VXI - VXI[board]::VXI logical address[::INSTR] GPIB-VXI - GPIB-VXI[board][::GPIB-VXI primary address]::VXI logical address[::INSTR] The VXI keyword is used for VXI instruments via either embedded or MXIbus controllers. The GPIB-VXI keyword is used for a GPIB-VXI controller. The default values for optional parameters are shown below. *Optional Parameter* — *Default Value* Board — 0 secondary address — none - 31 GPIB-VXI primary address — 1
IDQuery	ViBoolean	Specifies if an ID Query is sent to the instrument during the initialization procedure. Valid Range: 1 = Yes 0 = No Default Value: 1 - Yes NOTE: Under normal circumstances the ID Query insures that the instrument initialized over the bus is the type supported by this driver. However circumstances may arise where it is undesirable to send an ID Query to the instrument. In those cases; set this control to Skip Query and this function will initialize the bus and the Command arrays in the driver, without doing an ID Query.
resetDevice	ViBoolean	Specifies if the instrument is to be reset to its power-on settings during the initialization procedure. Valid Range: 1 = Yes, 0 = No Default Value: 1 - Yes

		NOTE: If you do not want the instrument reset, set this control to No while initializing the instrument.
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the same model type is used, this Handle will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument. NOTE: A new (unique) handle will be returned EACH time the Initialize function is called. The ri3156B_close() should be used for EACH handle returned by the ri3156B_init() function.

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_reset

Description

Resets the instrument to the power-up, factory defaults state

C Syntax

ViStatus ri3156b_reset (ViSession instrHandle)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_close

Description

Closes the instrument and de-allocates the resources allocated by the call to the initialization function. This should be called once for EACH instrument handle returned by the ri3156B_init() function. This should be called prior to terminating the application program.

C Syntax
ViStatus ri3156b_close (ViSession instrHandle)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.

Return Values
Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

The Common Control Functions Group

This group is used to control macro 3156B functions. These commands set the instrument to its various operating, waveform and run modes. Functions from this group also control the reference source and the amplitude and offset settings of the instrument. Factory defaults after reset are given as well. The communication session is closed with the close function. Parameters range is given where applicable.

Common Control Functions			
3156B Function Name	**Parameter(s)**	**Range**	**Default**
ri3156B_set(query)_active_channel	ViInt16 channel	1-2	1
ri3156B_set(query)_output	ViBoolean outputSwitch	0-1 (OFF, ON)	0
ri3156B_set(query)_enable	ViBoolean outputSwitch	0-1 (OFF, ON)	0
ri3156B_set(query)_enable_source	ViInt16 enableSource	0-2 (SOFT, HARD, MIX)	0
ri3156B_set(query)_amplitude	ViReal64 amplitude	10e-3 to 10	5
ri3156B_set(query)_offset	ViReal64 offset	-4.995 to 4.995	0
ri3156B_set(query)_global_operating_mode	ViInt16 ch1_waveformMode	0-5 (STD, ARB, SEQ, MOD, VIDEO, DIGITAL)	0
	ViInt16 ch2_waveformMode	0-5 (STD, ARB, SEQ, MOD, VIDEO, DIGITAL)	0
	ViInt16 CarrierRunMode	0-3 (CONT, TRIG, GATE, BURST)	0
	ViInt16 OperateEnableSource	0-1 (SOFT, HARD, MIX)	0
ri3156B_set(query)_common_waveform_mode	ViInt16 commonWaveformMode	0-5 (STD, ARB, SEQ, MOD, VIDEO, DIGITAL)	0

ri3156B_set(query)_waveform_mode	ViInt16 waveformMode	0-5 (STD, ARB, SEQ, MOD, VIDEO, DIGITAL)	0
ri3156B_set(query)_carrier_run_mode	ViInt16 CarrierRunMode	0-3 (CONT, TRIG, GATE, BURST)	0
ri3156B_set(query)_modulation_mode	ViInt16 modulationMode	0-5 (OFF, FM, AM, FSK, SWEEP, HOP)	0
ri3156B_set(query)_modulation_run_mode	ViInt16 modulationRunMode	0-3 (CONT, TRIG, GATE, BURST)	0
ri3156B_set(query)_digital_mode	ViBoolean digitalMode	0-1 (FREERUN, STIMULUS)	0
ri3156B_set(query)_reference_oscillator	ViInt16 referenceOscillator	0-1 (INT (CLK10 or TCXO), EXT)	0

ri3156b_set(query)_active_channel

Description

This selects the 3156B Channel. The channel selected by this function is used for all subsequent function calls until either:

1) This function is called again to change the channel
2) The ri3156B_reset() function is called (selects channel 1)

C Syntax

ViStatus ri3156b_set_active_channel (ViSession instrHandle, ViInt16 channel)
ViStatus ri3156b_query_active_channel (ViSession instrHandle, ViInt16 * channel)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
channel	ViInt16	This control determines which of the channels is being programmed for the 3156B. This command must be applied at the beginning of the string as all subsequent commands will apply to the selected channel only. Valid Range: 1, 2 Default Value: 1

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_set(query)_output

Description

This activates a relay that connects the output connector to the output circuit. Set Enable ON is required to start generating waveforms. The active channel is selected from the Set Active Channel panel. Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_output (ViSession instrHandle, ViBoolean outputSwitch)
ViStatus ri3156b_query_output (ViSession instrHandle, ViBoolean * outputSwitch)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
outputSwitch	ViBoolean	This function activates a mechanical switch (relay) that connects or disconnects the output connector to or from the electrical circuit. In the OFF mode, the inner side of the output connector is floating, thus representing high impedance to the external device. In the ON position, the output is driven from a 50Ω source. This function alone will not cause the output to generate signals but must be applied to have the signal path routed to the output connector. Each channel has its own switch and therefore each channel switch must be turned on separately. Signal is generated at the output connector only after an enable command is set true, in continuous mode or a valid trigger signal is applied, in one of the interrupted modes. Valid Range: 0, 1 (Off, On) Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_set(query)_enable

Description
This panel is used for enabling or disabling output waveforms. Set Output ON is required to have the signal connected to the output connector. Channel Dependency: Common

C Syntax
ViStatus ri3156b_set_enable (ViSession instrHandle, ViBoolean outputSwitch)
ViStatus ri3156b_query_enable (ViSession instrHandle, ViBoolean * outputSwitch)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
outputSwitch	ViBoolean	Enables the output to generate signals in continuous run mode. Note that Set Output ON/OFF function must be used to connect the output connector to the output circuit path before the Set Enable function will affect the signal. The Set Enable function is not used for the 3156B interrupted modes (Trigger, Gated, and Burst). The Set Enable is global thus activating both channels simultaneously. In all continuous operating modes, the enable ON acts like a software gate-open signal and the enable OFF like a gate-close signal. Valid Range: 0, 1 (Off, On) Default Value: 0

Return Values
Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_set(query)_enable_source

Description
This panel is used for enabling or disabling output waveforms. Set Output ON is required to have the signal connected to the output connector. Channel Dependency: Common

C Syntax
ViStatus ri3156b_set_enable_source (ViSession instrHandle, ViInt16 enableSource)

ViStatus ri3156b_query_enable_source (ViSession instrHandle, ViInt16 * enableSource)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
enableSource	ViInt16	Enables This switch defines the source for the operation enable signal. Note that possible setting conflicts could occur if you set up illegal combination of operating mode, run mode and operate enable source. Possible setting conflicts are listed in chapter 3 Valid Range: 0-2 0 Output enable command is expected as a software command - Set Enable ON 1 Output enable command is expected from one of the following: front panel TRIG IN, TTLTRG(n), or ECLTRG0. The active trigger input is set using the Set Trigger Source command 2 First output cycle is initiated using the Set Output ON command, subsequent output cycles are initiated using one of the following: front panel TRIG IN, TTLTRG(n), or ECLTRG0. The active trigger input is set using the Set Trigger Source command Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_set(query)_amplitude

Description

Sets the amplitude for the presently selected waveform. The active channel is selected from the Set Active Channel panel. Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_amplitude (ViSession instrHandle, ViReal64 amplitude)
ViStatus ri3156b_query_amplitude (ViSession instrHandle, ViReal64 * amplitude)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
amplitude	ViReal64	Description: Selects the amplitude (Volts peak-to-peak). Each channel may be programmed separately to have different amplitude level. Amplitude and offset can be set independently as long as the following relationship is observed: +/-(Vp-p/2 + offset) <= Amplitude Window Amplitude window is from -5V to +5V into 50 ohms or double into high impedance Valid Range: 10e-3 to 10 (V into 50Ω) Default Value: 5

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_set(query)_offset

Description

Sets the offset for the presently selected waveform. The active channel is selected from the Set Active Channel panel. Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_offset (ViSession instrHandle, ViReal64 offset)
ViStatus ri3156b_query_offset (ViSession instrHandle, ViReal64 * offset)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
offset	ViReal64	Selects the offset. Each channel may be programmed separately to have different offset level. Amplitude and offset can be set independently as long as the following

		relationship is observed:+/-(Vp-p/2 + offset) <= Amplitude Window Amplitude window is from -5V to +5V into 50Ω or double into high impedance Valid Range: -4.995 to +4.995(V into 50Ω) Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_set(query)_global_operating_mode

Description

This function allows the user to globally select the 3156B operating mode for both channels, as well as, run mode and the operation enable source: Setting up correctly the global operating parameters of the 3156B is critical to avoid setting conflicts between channels and their respective operating and run modes and the source of the operation enable signal. Chapter 3 lists possible setting conflicts for global settings in conjunction with the various run modes.

The 3156B driver provides three function call options to program waveform mode for the 3156B:

4. The ri3156b_set_global_operating_mode - has five variables: Channel 1 waveform, Channel 2 waveform, Carrier Run Mode and Operate Enable Source. This function call is the best to use if channels need to be programmed with different functionality

5. The ri3156b_set_common_waveform_mode - has only one variable that sets both channels simultaneously to the same waveform mode. This function call would be the best to use if channels need to be programmed with the same functionality.

6. The ri3156b_set_common_waveform_mode - allows separate programming of each channel to a different waveform mode. From all functions, this is the least recommended function to be used as it may cause setting conflict errors should one not fully understand the limitation of the product. For example, the default waveform mode is Standard. If you use this function call to change the waveform mode to arbitrary, the 3156B will immediately generate an error because it has only one sample clock source while Standard and Arbitrary waveforms use two different sample clock settings and therefore, this call will generate an error. The Set Active Channel command is required to program each channel separately.

C Syntax

ViStatus ri3156b_set_global_operating_mode (ViSession instrHandle, ViInt16 ch1_waveformMode, ViInt16 ch2_waveformMode, ViInt16 carrierRunMode, ViInt16 operateEnableSource)
ViStatus ri3156b_query_global_operating_mode (ViSession instrHandle, ViInt16 * ch1_waveformMode, ViInt16 * ch2_waveformMode, ViInt16 * carrierRunMode, ViInt16 * operateEnableSource)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.

ch1_waveformMode	ViInt16	Setting up this function is mandatory for correct operation of the instrument. The 3156B has a lot of capability and functionality built into it and, at time, incorrect initialization of the main functions may cause numerous setting conflicts. It is therefore recommended to always use this function as part of the initialization process. The following table shows possible setting conflicts for channel 1: Valid Range: 0-5 0 STD - Use this standard mode to generate one waveform from a list of built in waveforms, such as Sine, Square, Triangle, Half Cycle, etc. 1 ARB - Use this arbitrary mode to generate an arbitrary waveform. Arbitrary waveform(s) must first be loaded into one or more memory segments. Once loaded, the segments become a library of waveform that can be selected individually to be generated at the output connector. 2 SEQ - The sequenced mode is used for generating a sequence of memory segments. The definition of the sequence is stored in a sequence table where segments can be linked and looped in a user defined fashion. 3 MOD - The modulation mode is used for generating modulated waveforms. The modulated carrier signal is always sine waveform. The carrier can be modulated using one of the following modulation modes: FM, AM, FSK, SWEEP and HOP 4 VIDEO - The video stoke is used for generating special video signals for driving TV's and analog monitors. In this mode, both channels operate simultaneously and in conjunction with each other, similar to XY mode in oscilloscopes. Therefore, any setting of global operation mode other than both channels set to video will result in setting conflict error 5 DIGITAL - The digital output mode cannot be turned off! Whenever there is a signal at the output connector, there is signal on the digital output lines as they are connected in parallel and buffered from the 12 bits (out of 16) that drive the DAC. If you specifically select the Digital mode, you also need to select if you want to operate in free-running or stimulus run mode. Default Value: 0
Ch2_waveformMode	ViInt16	Setting up this function is mandatory for correct operation of the instrument. The 3156B has a lot of capability and functionality built into it and, at time, incorrect initialization of the main functions may cause numerous setting conflicts. It is therefore recommended to always use this function as part of the initialization process. The following table shows possible setting conflicts for channel 1:

		Valid Range: 0-5 0 STD. Use this standard mode to generate one waveform from a list of built in waveforms, such as Sine, Square, Triangle, Half Cycle, etc. 1 ARB. Use this arbitrary mode to generate an arbitrary waveform. Arbitrary waveform(s) must first be loaded into one or more memory segments. Once loaded, the segments become a library of waveform that can be selected individually to be generated at the output connector. 2 SEQ. The sequenced mode is used for generating a sequence of memory segments. The definition of the sequence is stored in a sequence table where segments can be linked and looped in a user defined fashion. 3 MOD. The modulation mode is used for generating modulated waveforms. The modulated carrier signal is always sine waveform. The carrier can be modulated using one of the following modulation modes: FM, AM, FSK, SWEEP and HOP 4 VIDEO - The video stoke is used for generating special video signals for driving TV's and analog monitors. In this mode, both channels operate simultaneously and in conjunction with each other, similar to XY mode in oscilloscopes. Therefore, any setting of global operation mode other than both channels set to video will result in setting conflict error 5 DIGITAL - The digital output mode cannot be turned off! Whenever there is a signal at the output connector, there is signal on the digital output lines as they are connected in parallel and buffered from the 12 bits (out of 16) that drive the DAC. If you specifically select the Digital mode, you also need to select if you want to operate in free-running or stimulus run mode. Default Value: 0
carrierRunMode	ViInt16	This switch defines the run mode for the 3156B. Select one of CONT, TRIG, GATED and BURST. Setting up the global run mode is mandatory for correct operation of the instrument. The 3156B has a lot of capability and functionality built into it and, at time, incorrect initialization of its global operating and run modes may cause numerous setting conflicts. It is therefore recommended to always use this function as part of the initialization process. Possible setting conflicts are listed in Chapter 3. Note that Set Output On command alone will not enable the 3156B to generate signal at its output connectors. The Set Output On command activates a relay that connects the output circuit to the output connector. To start generating waveforms, the 3156B must be enabled using either a software enable or hardware enable commands, or combination of both. Hardware commands

		are valid trigger signals that are applied to the appropriate trigger input Valid Range: 0-3 0 Continuous - continuous Waveform is generated following a valid Operate Enable signal or command, Trigger Delay ON will delay the first output by the amount set with the trigger delay parameter. Re-trigger delay has no effect on the continuous operating mode. 1 Triggered - Single waveform cycle is initiated following a valid Operate Enable signal or command, Trigger Delay ON will delay the first cycle by the amount set with the trigger delay parameter. With the Re-trigger State ON, only one valid Operate Enable signal or command is needed to initiate continuous single-cycle waveforms; The Re-trigger Delay value defines the time that will elapse from the completion of one cycle to the start of the next. Re-trigger delay state and value has no effect if Operate Enable Source is set to MIX however, in Mixed mode, hardware triggers will be ignored and ONLY the first software operate enable command will initiate the first waveform cycle then, only hardware triggers will initiate waveform cycles. 2 Gated - continuous waveform cycles are initiated following a valid hardware Operate Enable signal. Another valid hardware Operate Enable command will stop the waveform. Last cycle is always completed. Trigger Delay State and value and Re-trigger Delay State and value have no effect on gated operation. Set Trigger Slope defines the active transition direction. 3 Burst - has the same functionality as the trigger mode except a preset number of up to 1M cycles is triggered Default Value: 0
operateEnableSource	ViInt16	This switch defines the source for the operation enable signal. Note that possible setting conflicts could occur if you set up illegal combination of operating mode, run mode and operate enable source. Possible setting conflicts are listed in this panel description. Valid Range: 0-3 0 SOFT - Output enable command is expected as a software command - Set Enable ON 1 HARD - Output enable command is expected from one of the following: front panel TRIG IN, TTLTRG(n), or ECLTRG0. The active trigger input is set using the Set Trigger Source command 2 MIX - First output cycle is initiated using the Set Output ON command, subsequent output cycles are

initiated using one of the following: front panel TRIG IN, TTLTRG(n), or ECLTRG0. The active trigger input is set using the Set Trigger Source command

Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_set(query)_common_waveform_mode

Description

This function allows the user to globally select the 3156B operating mode for both channels, as well as, run mode and the operation enable source: Setting up correctly the global operating parameters of the 3156B is critical to avoid setting conflicts between channels and their respective operating and run modes and the source of the operation enable signal. Chapter 3 lists possible setting conflicts for global settings in conjunction with the various run modes.

The 3156B driver provides three function call options to program waveform mode for the 3156B:

1. The ri3156b_set_global_operating_mode - has five variables: Channel 1 waveform, Channel 2 waveform, Carrier Run Mode and Operate Enable Source. This function call is the best to use if channels need to be programmed with different functionality

2. The ri3156b_set_common_waveform_mode - has only one variable that sets both channels simultaneously to the same waveform mode. This function call would be the best to use if channels need to be programmed with the same functionality.

3. The ri3156b_set_common_waveform_mode - allows separate programming of each channel to a different waveform mode. From all functions, this is the least recommended function to be used as it may cause setting conflict errors should one not fully understand the limitation of the product. For example, the default waveform mode is Standard. If you use this function call to change the waveform mode to arbitrary, the 3156B will immediately generate an error because it has only one sample clock source while Standard and Arbitrary waveforms use two different sample clock settings and therefore, this call will generate an error. The Set Active Channel command is required to program each channel separately.

C Syntax

ViStatus ri3156b_set_common_waveform_mode (ViSession instrHandle, ViInt16 commonWaveformMode)
ViStatus ri3156b_query_common_waveform_mode (ViSession instrHandle, ViInt16 * commonWaveformMode)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
commonWaveformMode	ViInt16	Programs both channels simultaneously to generate the same waveform type: Standard, Arbitrary, Sequenced,

Modulated, Video Stroke and Digital Patterns. The Set Active Channel command is not required to execute the Set Common Waveform Mode command. Channel dependency: common

Valid Range: 0-5

0 STD - Use this standard mode to generate one waveform from a list of built in waveforms, such as Sine, Square, Triangle, Half Cycle, etc.

1 ARB - Use this arbitrary mode to generate an arbitrary waveform. Arbitrary waveform(s) must first be loaded into one or more memory segments. Once loaded, the segments become a library of waveform that can be selected individually to be generated at the output connector.

2 SEQ - The sequenced mode is used for generating a sequence of memory segments. The definition of the sequence is stored in a sequence table where segments can be linked and looped in a user defined fashion.

3 MOD - The modulation mode is used for generating modulated waveforms. The modulated carrier signal is always sine waveform. The carrier can be modulated using one of the following modulation modes: FM, AM, FSK, SWEEP and HOP

4 VIDEO - The video stoke is used for generating special video signals for driving TV's and analog monitors. In this mode, both channels operate simultaneously and in conjunction with each other, similar to XY mode in oscilloscopes. Therefore, any setting of global operation mode other than both channels set to video will result in setting conflict error

5 DIGITAL - The digital output mode cannot be turned off! Whenever there is a signal at the output connector, there is signal on the digital output lines as they are connected in parallel and buffered from the 12 bits (out of 16) that drive the DAC. If you specifically select the Digital mode, you also need to select if you want to operate in free-running or stimulus run mode.

Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_waveform_mode

Description

This function allows the user to globally select the 3156B operating mode for both channels, as well as, run mode and the operation enable source: Setting up correctly the global operating parameters of the 3156B is critical to avoid setting conflicts between channels and their respective operating and run modes and the source of the operation enable signal. Chapter 3 lists possible setting conflicts for global settings in conjunction with the various run modes.

The 3156B driver provides three function call options to program waveform mode for the 3156B:

1. The ri3156b_set_global_operating_mode - has five variables: Channel 1 waveform, Channel 2 waveform, Carrier Run Mode and Operate Enable Source. This function call is the best to use if channels need to be programmed with different functionality

2. The ri3156b_set_common_waveform_mode - has only one variable that sets both channels simultaneously to the same waveform mode. This function call would be the best to use if channels need to be programmed with the same functionality.

3. The ri3156b_set_waveform_mode - allows separate programming of each channel to a different waveform mode. From all functions, this is the least recommended function to be used as it may cause setting conflict errors should one not fully understand the limitation of the product. For example, the default waveform mode is Standard. If you use this function call to change the waveform mode to arbitrary, the 3156B will immediately generate an error because it has only one sample clock source while Standard and Arbitrary waveforms use two different sample clock settings and therefore, this call will generate an error. The Set Active Channel command is required to program each channel separately.

C Syntax

ViStatus ri3156b_set_waveform_mode (ViSession instrHandle, ViInt16 channelWaveformMode)
ViStatus ri3156b_query_waveform_mode (ViSession instrHandle, ViInt16 * channelWaveformMode)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
channelWaveformMode	ViInt16	Programs the waveform mode for each channel separately. The Set Active Channel command should be used to selectively program each channel. Channel dependency: independent Valid Range: 0-5 0 STD - Use this standard mode to generate one waveform from a list of built in waveforms, such as Sine, Square, Triangle, Half Cycle, etc. 1 ARB - Use this arbitrary mode to generate an arbitrary waveform. Arbitrary waveform(s) must

first be loaded into one or more memory segments. Once loaded, the segments become a library of waveform that can be selected individually to be generated at the output connector.

2 SEQ - The sequenced mode is used for generating a sequence of memory segments. The definition of the sequence is stored in a sequence table where segments can be linked and looped in a user defined fashion.

3 MOD - The modulation mode is used for generating modulated waveforms. The modulated carrier signal is always sine waveform. The carrier can be modulated using one of the following modulation modes: FM, AM, FSK, SWEEP and HOP

4 VIDEO - The video stoke is used for generating special video signals for driving TV's and analog monitors. In this mode, both channels operate simultaneously and in conjunction with each other, similar to XY mode in oscilloscopes. Therefore, any setting of global operation mode other than both channels set to video will result in setting conflict error

5 DIGITAL - The digital output mode cannot be turned off! Whenever there is a signal at the output connector, there is signal on the digital output lines as they are connected in parallel and buffered from the 12 bits (out of 16) that drive the DAC. If you specifically select the Digital mode, you also need to select if you want to operate in free-running or stimulus run mode.

Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_carrier_run_mode

Description

This function allows the user to select carrier run mode. The 3156B offers four run modes: Continuous, Triggered, Gated, and Burst. The selected waveform is repeated continuously when the instrument is set to operate in continuous mode. In this mode, the 3156B requires a ri3156B_set_enable (1) call to stimulate output cycles. The default operating mode of the instrument is continuous.

Triggered, Gated, and Burst modes require an external signal, or software command to initiate output cycles. See the description for the function ri3156B_set_global_opearting_mode() for a discussion on selecting the appropriate 3156B Run Mode.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_carrier_run_mode (ViSession instrHandle, ViInt16 carrierRunMode)
ViStatus ri3156b_query_carrier_run_mode (ViSession instrHandle, ViInt16 * carrierRunMode)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
carrierRunMode	ViInt16	Selects one of the 3156B run modes: CONT, TRIG, GATED and BURST. Valid Range: 0-3 0 Continuous - continuous waveform is generated, following a ri3156B_set_enable (1) call 1 Triggered - each trigger generates a single waveform cycle. The trigger source is selected using the ri3156B_set_trigger_source() function 2 Gated - An external signal enables output waveforms. First output cycle is synchronous with the active slope of the trigger signal. Last waveform cycle is always completed. 3 Burst - A preset number of up to 1M cycles is triggered by either a backplane trigger or by the internal timer. This mode is available in either STD and User modes only. The source of the trigger is programmed using the ri3156B_trigger_source() call. The number of cycles is programmed using the function ri3156B_burst_mode(). Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_modulation_mode

Description

This function allows the user to turn on the modulation function and select a modulation mode. Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_modulation_mode (ViSession instrHandle, ViInt16 modulationMode)
ViStatus ri3156b_query_modulation_mode (ViSession instrHandle, ViInt16 * modulationMode)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
modulationMode	ViInt16	Selects one of the modulation modes: OFF, FM, AM, FSK, SWEEP and HOP. It is recommended that modulation parameters be pre-programmed before the required modulation mode is turned on. Note the following behavior of the 3156B: 1. Set Output ON, Set Enable ON and selection of one of the modulation modes must be used before a modulation function becomes available at the output connector. The Set Enable ON is not required if run mode other than continuous is selected. 2. All modulation functions share modulation run mode setting such as continuous, trigger, gated and burst. However, if one channel is set to modulation, the other channel automatically reverts to continuous AC mode. 3. Using the global operating Mode function, it is possible to have both channels set to one of the modulation modes. In this case, the modulation mode must be the same for both channels and the outputs will have 90 degrees phase shift. 4. Modulation run mode and Global Operating Mode do not share the same values. 5. If the selected modulation run mode is trigger or gated and valid trigger signal is not available, or trigger signal is false, the remains at a DC level. After the first trigger or gate, the output generates carrier, non-modulated frequency. Carrier frequency is either programmable or available as default for each function. Channel dependency: independent Valid Range: 0-5 0 Modulation off 1 Frequency modulation (FM) 2 Amplitude modulation (AM) 3 Frequency shift keying modulation (FSK)

		4 Sweep modulation
		5 Frequency hopping modulation
		Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_modulation_run_mode

Description

This function allows the user to select modulation run mode. Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_modulation_run_mode (ViSession instrHandle, ViInt16 modulationRunMode)
ViStatus ri3156b_query_modulation_run_mode (ViSession instrHandle, ViInt16 * modulationRunMode)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
modulationRunMode	ViInt16	Selects one of the modulation run modes: CONT, TRIG, GATED or BURST. Note that modulation run mode options, although they have the same names as global operating mode run modes, have different meaning. Valid Range: 0-3 0 Continuous - continuous waveform is generated 1 Triggered - each trigger generates a single modulation cycle. The trigger source is selected using the ri3156B_set_trigger_source() function 2 Gated - An external signal enables modulation. First output cycle is synchronous with the active slope of the trigger signal. Last cycle of modulated waveform is always completed. 3 Burst - each trigger generates a single burst of modulation cycles. The trigger source is selected using the ri3156B_set_trigger_source() function Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_digital_mode

Description

Use this function to select between freerun or stimulus modes. Channel Dependency: common.

C Syntax

ViStatus ri3156b_set_digital_mode (ViSession instrHandle, ViBoolean digitalMode)
ViStatus ri3156b_query_digital_mode (ViSession instrHandle, ViBoolean * digitalMode)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
digitalMode	ViBoolean	Selects one of the digital modes: freerun or stimulus. The digital output connector has 12 differential data lines that are ECL compatible and must be terminated to -2V through 50 ohms resistors. The 3156B is a 16-bit generator however, only 12 bits are available for this output therefore, data to the digital outputs must be multiplied by 16 to shift the data to the correct bits. 3156B rules and run modes apply to the digital output. Valid Range: 0-1 0 freerun digital patterns are generated following a ri3156B_set_enable (1) call. Hold count is programmable for each pattern step 1 stimulus digital patterns are generated following a ri3156B_set_enable (1) call. Hold count is fixed for each pattern step Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_reference_oscillator

Description

This function selects the reference oscillator source for the sample clock synthesizer circuit. Channel Dependency: common.

C Syntax

ViStatus ri3156b_set_reference_oscillator (ViSession instrHandle, ViInt16 referenceOscillator)
ViStatus ri3156b_query_reference_oscillator (ViSession instrHandle, ViInt16 * referenceOscillator)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
referenceOscillator	ViInt16	Sets the reference oscillator to either Internal or External. The internal reference could be either CLK10 (back-plane 10 MHz, 100ppm source) or optional TCXO. The TCXO (temperature compensated crystal oscillator) has better stability than the back plane source however, it has to be bought separately and installed by the factory before shipment. If this option is installed, the CLK10 source is not available anymore. In cases where better reference source is needed, system clock can be applied to a front-panel connector and the reference source set to external. Valid Range: 0-1 0 select CLK10 or TCXO 1 selects external source Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

The Standard Waveforms Functions Group

This group is used to control the 3156B standard waveform shapes, their respective parameters, frequency, amplitude and offset. If both channels are set to operate in standard waveforms mode, each channel can be programmed independently to output different waveform shape. Parameter ranges and their defaults are given where applicable. The communication session is terminated with the close function.

Standard Waveforms Programming			
3156B Function Name	**Parameter(s)**	**Range**	**Default**
ri3156B_set(query)_standard_waveform	ViInt16 standardWaveform	0-9	0
ri3156B_set(query)_frequency	ViReal64 frequency	0.01 to 25e6	1e6
ri3156B_query_std_sample_clock_freq	ViPReal64 stdSampleClockFrequency		
ri3156B_query_std_waveform_numb_points	ViPInt16 standardWaveformPoints		

Sine Wave Functions			
ri3156B_set(query)_sine_wave_phase	ViReal64 phase	0 to 359.95	0
ri3156B_apply_sine_wave	ViReal64 frequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 phase	0 to 359.95	0
Triangle Wave Functions			
ri3156B_set(query)_triangular_wave_phase	ViReal64 phase	0 to 359.95	0
ri3156B_apply_triangular_wave	ViReal64 frequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 phase	0 to 359.95	0
Square Wave Functions			
ri3156B_set(query)_square_wave_duty_cycle	ViReal64 dutyCycle	0 to 99.99	50
ri3156B_apply_square_wave	ViReal64 frequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 dutyCycle	0 to 99.99	50
Half Cycle Wave Functions			
ri3156B_set(query)_half_cycle_mode	ViBoolean HalfCycleMode	0-1 (OFF, ON)	0
ri3156B_set(query)_half_cycle_delay	ViReal64 HalfCycleDelay	0, 500e-9 to 21	1e-6
ri3156B_apply_half_cycle_sine_wave	ViReal64 frequency	0.01 to 1e6	500e3
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 delay	0, 500e-9 to 21	1e-6
	ViInt16 Waveform	0-2	0
3156B Function Name	**Parameter(s)**	**Range**	**Default**
	ViReal64 phase	0 to 359.95	0
	ViReal64 dutyCycle	0 to 99.99	50
Pulse Wave Functions			
ri3156B_set(query)_pulse_wave_PRF	ViReal64 pulsePRF	0.01 to 25e6	1e6
ri3156B_set(query)_pulse_wave_high_time	ViReal64 highTime	0 to 99.99	10
ri3156B_set(query)_pulse_wave_delay	ViReal64 delayTime	0 to 99.99	10
ri3156B_set(query)_pulse_wave_rise_time	ViReal64 riseTime	0 to 99.99	10
ri3156B_set(query)_pulse_wave_fall_time	ViReal64 fallTime	0 to 99.99	10
ri3156B_apply_pulse_wave	ViReal64 pulsePRF	0.01 to 25e6	1e
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 highTime	0 to 99.99	10
	ViReal64 delayTime	0 to 99.99	10
	ViReal64 riseTime	0 to 99.99	10
	ViReal64 fallTime	0 to 99.99	10
Ramp Wave Functions			
ri3156B_set(query)_ramp_wave_slope	ViReal64 rampSlope	0 to 99.99	10

ri3156B_apply_ramp_wave	ViReal64 frequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 rampSlope	0 to 99.99	10
Sinc Wave Functions			
ri3156B_set(query)_sinc_wave_num_cycles	ViInt16 numberofCycles	4 to 100	10
ri3156B_apply_sinc_wave	ViReal64 frequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViInt16 numberofCycles	4 to 100	10
Exponential Wave Functions			
ri3156B_set(query)_exponential_wave_exponent	ViInt16 exponent	-100 to 100	-1
ri3156B_apply_exponential_wave	ViReal64 frequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViInt16 exponent	-100 to 100	-1
Gaussian Wave Functions			
ri3156B_set(query)_gaussian_wave_exponent	ViInt16 exponent	10 to 200	10
ri3156B_apply_gaussian_wave	ViReal64 frequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10mV to 10V	5
	ViReal64 offset	-4.995 to 4.995	0
	ViInt16 exponent	10 to 200	10
DC Signal Functions			
ri3156B_set(query)_dc_signal_percent	ViReal64 percentAmplitude	-100 to 100	100
ri3156B_apply_dc_signal	ViReal64 percentAmplitude	-100 to 100	100

Ri3156b_set(query)_standard_waveform

Description

The Standard Waveform commands control the various parameters of the active Standard Waveform. The waveform mode should be set to "Standard" for the selected waveform to be applied to the channel. See the function ri3156B_set_waveform_mode() for a description of how to select the standard waveform mode.

The number of points used to define each Standard Waveform varies according to the programmed frequency. Thus, some parameters may not have any effect on the waveform because too few points are available to generate the waveform.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_standard_waveform (ViSession instrHandle, ViInt16 standardWaveform)
ViStatus ri3156b_query_standard_waveform (ViSession instrHandle, ViInt16 * standardWaveform)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver

		and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
standardWaveform	ViInt16	Selects a Standard Waveform Valid Range: 0-8 0 selects sine waveform 1 selects triangle waveform 2 selects square waveform 3 selects pulse waveform 4 selects ramp waveform 5 selects sinc waveform 6 selects exponential waveform 7 selects gaussian waveform 8 selects dc waveform Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_frequency

Description

Sets the frequency for the presently selected standard waveform. Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_frequency (ViSession instrHandle, ViReal64 frequency)
ViStatus ri3156b_query_frequency (ViSession instrHandle, ViReal64 * frequency)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
frequency	ViReal64	Programs the frequency for the presently selected waveform. Valid Range: 0.01 to 25e6 Default Value: 1e6

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_query_std_sample_clock_freq

Description

Queries the internal sample clock setting when the 3156B generates standard waveforms. Note that this parameter is not programmable and is automatically set by the 3156B for the appropriate standard waveform frequency This query can be used for reference purpose only

C Syntax

ViStatus ri3156b_query_std_sample_clock_freq (ViSession instrHandle, ViPReal64 * stdSampleClockFrequency)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
stdSampleClockFrequency	ViReal64	Returns the present setting of the sample clock frequency. Note that this parameter is not programmable and is automatically set by the 3156B for the appropriate standard waveform frequency, depending on the number of points the 3156B defines for generating the standard waveform shape.

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_query_std_waveform_numb_points

Description

Queries the number of points the instrument is using internally when the 3156B generates standard waveforms. Note that this parameter is not programmable and is automatically set by the 3156B for the appropriate standard waveform frequency This query can be used for reference purpose only.

C Syntax

ViStatus ri3156b_query_stand_wav_numb_points (ViSession instrHandle, ViPInt16 * standartWaveformPoints)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
standartWaveformPoints	ViInt16	Returns the present (internal) setting of the number of points used by the instrument for generating the programmed standard waveform. Note that this parameter is not programmable and is automatically set by the 3156B for the selected waveform, depending on the sample clock frequency the 3156B defines for generating the output waveform.

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_sine_wave_phase

Description

Programs only the phase of the standard sine wave. This function does NOT select the sine wave as the active waveform. Note: The frequency, amplitude, and offset may be varied by using the following functions:

ri3156B_set_frequency()

ri3156B_set_amplitude()

ri3156B_set_offset()

You may set all sine wave parameters at once, by using the ri3156B_apply_sine_wave() function, or by using the individual functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_sine_wave_phase (ViSession instrHandle, ViReal64 phase)
ViStatus ri3156b_query_sine_wave_phase (ViSession instrHandle, ViReal64 * phase)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
phase	ViReal64	Selects the starting phase of the sine wave. Valid range: 0 to 359.95 (°) Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_sine_wave

Description

Programs frequency, amplitude, offset, and phase of the sine wave function. Also sets the Sine waveform as the active waveform. Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_sine_wave (ViSession instrHandle, ViReal64 frequency, ViReal64 amplitude, ViReal64 offset, ViReal64 phase)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
frequency	ViReal64	Programs the frequency for the selected function Valid range: 0.01 to 25e6 (Hz) Default: 1e6
amplitude	ViReal64	Programs the amplitude for the selected function

		Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0
phase	ViReal64	Selects the starting phase of the selected function Valid range: 0 to 359.95 (°) Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_triangular_wave_phase

Description

Programs only the phase of the standard triangular wave. This function does NOT select the triangular wave as the active waveform. Note: The frequency, amplitude, and offset may be varied by using the following functions:

ri3156B_set_frequency()

ri3156B_set_amplitude()

ri3156B_set_offset()

You may set all triangular wave parameters at once, by using the ri3156B_apply_triangular_wave() function, or by using the individual functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_triangular_wave_phase (ViSession instrHandle, ViReal64 phase)
ViStatus ri3156b_query_triangular_wave_phase (ViSession instrHandle, ViReal64 * phase)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.

phase	ViReal64	Selects the starting phase of the triangular wave. Valid range: 0 to 359.95 (°) Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_triangular_wave

Description

Programs frequency, amplitude, offset, and phase of the triangular wave function. Also sets the triangular waveform as the active waveform. Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_triangular_wave (ViSession instrHandle, ViReal64 frequency, ViReal64 amplitude, ViReal64 offset, ViReal64 phase)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
frequency	ViReal64	Programs the frequency for the selected function Valid range: 0.01 to 25e6 (Hz) Default: 1e6
amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0
phase	ViReal64	Selects the starting phase of the selected function Valid range: 0 to 359.95 (°) Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_square_wave_duty_cycle

Description

Programs only the duty cycle of the standard square wave. This function does NOT select the square wave as the active waveform. Note: The frequency, amplitude, and offset may be varied by using the following functions:

ri3156B_set_frequency()

ri3156B_set_amplitude()

ri3156B_set_offset()

You may set all square wave parameters at once, by using the ri3156B_apply_square_wave() function, or by using the individual functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_square_wave_duty_cycle (ViSession instrHandle, ViReal64 dutyCycle)
ViStatus ri3156b_query_square_wave_duty_cycle (ViSession instrHandle, ViReal64 * dutyCycle)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
dutyCycle	ViReal64	Selects the duty cycle of the square wave. Valid range: 0 to 99.99 (%) Default: 50

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_square_wave

Description

Programs frequency, amplitude, offset, and duty cycle of the square wave function. Also sets the square waveform as the active waveform. Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_square_wave (ViSession instrHandle, ViReal64 frequency, ViReal64 amplitude, ViReal64 offset, ViReal64 dutyCycle)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
frequency	ViReal64	Programs the frequency for the selected function Valid range: 0.01 to 25e6 (Hz) Default: 1e6
amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0
dutyCycle	ViReal64	Selects the duty cycle of the square wave. Valid range: 0 to 99.99 (%) Default: 50

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_half_cycle_mode

Description

Programs the half cycle mode. Half cycles are available for sine, triangle and square waveforms only. This function selects the half cycle as the active mode only if the 3156B is programmed to output one of these three waveforms. Note: The frequency, amplitude, and offset may be varied by using the following functions:

ri3156B_set_frequency()

ri3156B_set_amplitude()

ri3156B_set_offset()

You may set all half cycle wave parameters at once, by using the ri3156B_apply_half_cycle_wave() function, or by using the individual functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_half_cycle_mode (ViSession instrHandle, ViBoolean halfCycleMode)
ViStatus ri3156b_query_half_cycle_mode (ViSession instrHandle, ViBoolean * halfCycleMode)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
halfCycleMode	ViBoolean	This function splits sine triangle and square waveforms to two symmetrical sections. Valid range: 0, 1 (Off, On Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_half_cycle_delay

Description

Programs the amount of time that will elapse between the end of the delivery of the first one-half cycle and the beginning of the second one-half of the cycle. Note: The frequency, amplitude and offset may be varied by using the following functions:

ri3156B_set_frequency()

ri3156B_set_amplitude()

ri3156B_set_offset()

You may set all half cycle sine wave parameters at once, by using the ri3156B_apply_half_cycle_wave() function, or by using the individual functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_half_cycle_delay (ViSession instrHandle, ViReal64 halfCycleSineDelay)
ViStatus ri3156b_query_half_cycle_delay (ViSession instrHandle, ViReal64 * halfCycleSineDelay)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
halfCycleSineDelay	ViReal64	Programs the half cycle delay time. ***NOTE***: Half-cycle delay time is programmed in units of µs (microsecond). For example, to program 1.37ms delay time, enter the value as follows: 1370 Valid range: 0.5 to 21e6 (µs) Default: 0.5

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_half_cycle_wave

Description

Programs frequency, amplitude, offset, delay, waveform shape and phase or duty cycle of the half cycle wave function. Also sets the half cycle mode as the active mode but only if the selected waveform is one of sine, triangle or square.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_half_cycle_wave (ViSession instrHandle, ViReal64 frequency, ViReal64 amplitude, ViReal64 offset, ViReal64 delay, ViInt16 waveform, ViReal64 phase/DutyCycle)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
frequency	ViReal64	Programs the frequency for the selected function Valid range: 0.01 to 1e6 (Hz) Default: 500e3
amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0
delay	ViReal64	Programs the half cycle delay time. ***NOTE***: Half-cycle delay time is programmed in units of μs (microsecond). For example, to program 1.37ms delay time, enter the value as follows: 1370 Valid range: 0.5 to 21e6 (μs) Default: 0.5

waveform	ViInt16	Selects a standard waveform for the half cycle mode. Valid range: 0 to 2 0 selects sine waveform 1 selects triangle waveform 2 selects square waveform Default: 0
phase, or dutyCycle	ViReal64	Selects the start phase if the waveform is sine or triangle, or duty cycle, if the waveform is square. Valid range: 0 to 359.95 (°) start phase for sine and triangle waves Default: 0 Valid range: 0 to 99.99 (%) duty cycle for square wave Default: 50

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_pulse_wave_PRF

Description

Programs only the pulse repetition of the standard pulse wave. This function does NOT select the pulse wave as the active waveform. Note: The frequency, amplitude, and offset may be varied by using the following functions:

ri3156B_set_frequency()

ri3156B_set_amplitude()

ri3156B_set_offset()

You may set all pulse wave parameters at once, by using the ri3156B_apply_pulse_wave() function, or by using the individual functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_pulse_wave_PRF (ViSession instrHandle, ViReal64 pulsePRF)
ViStatus ri3156b_query_pulse_wave_PRF (ViSession instrHandle, ViReal64 * pulsePRF)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
pulsePRF	ViReal64	Programs the pulse repetition frequency for the pulse waveform. Valid range: 0.01 to 25e6 (Hz) Default: 1e6

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_pulse_high _time

Description

Programs only the high time of the standard pulse wave. This function does NOT select the pulse wave as the active waveform. Note: The frequency, amplitude, and offset may be varied by using the following functions:

ri3156B_set_frequency()

ri3156B_set_amplitude()

ri3156B_set_offset()

You may set all pulse wave parameters at once, by using the ri3156B_apply_pulse_wave() function, or by using the individual functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_pulse_wave_high_time (ViSession instrHandle, ViReal64 highTime)
ViStatus ri3156b_query_pulse_wave_high_time (ViSession instrHandle, ViReal64 * highTime)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver

		and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
highTime	ViReal64	Programs the high time portion of the pulse waveform. Valid range: 0 to 9.99 (%) Default: 10

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_pulse_delay _time

Description

Programs only the delay time of the standard pulse wave. This function does NOT select the pulse wave as the active waveform. Note: The frequency, amplitude, and offset may be varied by using the following functions:

ri3156B_set_frequency()

ri3156B_set_amplitude()

ri3156B_set_offset()

You may set all pulse wave parameters at once, by using the ri3156B_apply_pulse_wave() function, or by using the individual functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_pulse_wave_delay_time (ViSession instrHandle, ViReal64 delayTime)
ViStatus ri3156b_query_pulse_wave_delay_time (ViSession instrHandle, ViReal64 * delayTime)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
delayTime	ViReal64	Programs the delay time portion of the pulse waveform. Valid range: 0 to 9.99 (%) Default: 10

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_pulse_rise _time

Description

Programs only the rise time of the standard pulse wave. This function does NOT select the pulse wave as the active waveform. Note: The frequency, amplitude, and offset may be varied by using the following functions:

ri3156B_set_frequency()

ri3156B_set_amplitude()

ri3156B_set_offset()

You may set all pulse wave parameters at once, by using the ri3156B_apply_pulse_wave() function, or by using the individual functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_pulse_wave_rise_time (ViSession instrHandle, ViReal64 riseTime)
ViStatus ri3156b_query_pulse_wave_rise_time (ViSession instrHandle, ViReal64 * riseTime)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
riseTime	ViReal64	Programs the rise time portion of the pulse waveform. Valid range: 0 to 9.99 (%) Default: 10

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_pulse_fall _time

Description

Programs only the fall time of the standard pulse wave. This function does NOT select the pulse wave as the active waveform. Note: The frequency, amplitude, and offset may be varied by using the following functions:

ri3156B_set_frequency()

ri3156B_set_amplitude()

ri3156B_set_offset()

You may set all pulse wave parameters at once, by using the ri3156B_apply_pulse_wave() function, or by using the individual functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_pulse_wave_fall_time (ViSession instrHandle, ViReal64 fallTime)
ViStatus ri3156b_query_pulse_wave_fall_time (ViSession instrHandle, ViReal64 * fallTime)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
fallTime	ViReal64	Programs the fall time portion of the pulse waveform. Valid range: 0 to 9.99 (%) Default: 10

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_pulse_wave

Description

Programs PRF (pulse repetition frequency), amplitude, offset, high, delay, rise and fall time of the pulse wave function. Also sets the pulse waveform as the active waveform. Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_pulse_wave (ViSession instrHandle, ViReal64 pulsePRF, ViReal64 amplitude, ViReal64 offset, ViReal64 highTime, ViReal64 delay, ViReal64 rise, ViReal64 fallTime)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
pulsePRF	ViReal64	Programs the PRF (pulse repetition frequency) for the selected function Valid range: 0.01 to 25e6 (Hz) Default: 1e6
amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0
highTime	ViReal64	Programs the high time portion of the pulse waveform. Valid range: 0 to 9.99 (%) Default: 10
delayTime	ViReal64	Programs the delay time portion of the pulse waveform. Valid range: 0 to 9.99 (%) Default: 10

riseTime	ViReal64	Programs the rise time portion of the pulse waveform. Valid range: 0 to 9.99 (%) Default: 10
fallTime	ViReal64	Programs the fall time portion of the pulse waveform. Valid range: 0 to 9.99 (%) Default: 10

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_ramp_wave_slope

Description

Programs only the ramp time of the standard ramp wave. This function does NOT select the square wave as the active waveform. Note: The frequency, amplitude, and offset may be varied by using the following functions:

ri3156B_set_frequency()

ri3156B_set_amplitude()

ri3156B_set_offset()

You may set all ramp wave parameters at once, by using the ri3156B_apply_ramp_wave() function, or by using the individual functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_ramp_wave_slope (ViSession instrHandle, ViReal64 rampSlope)
ViStatus ri3156b_query_ramp_wave_slope (ViSession instrHandle, ViReal64 * rampSlope)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.

rampSlope	ViReal64	Selects the slope of the ramp wave. Valid range: 0 to 99.99 (%) Default: 50

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_ramp_wave

Description

Programs frequency, amplitude, offset, and duty cycle of the ramp wave function. Also sets the ramp waveform as the active waveform. Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_ ramp _wave (ViSession instrHandle, ViReal64 frequency, ViReal64 amplitude, ViReal64 offset, ViReal64 rampSlope)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
frequency	ViReal64	Programs the frequency for the selected function Valid range: 0.01 to 25e6 (Hz) Default: 1e6
amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0

rampSlope	ViReal64	Selects the slope of the ramp wave. Valid range: 0 to 99.99 (%) Default: 50

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_sinc_wave_numb_cycles

Description

Programs only the number of “0” crossing cycles of the standard sinc wave. This function does NOT select the sinc wave as the active waveform. Note: The frequency, amplitude, and offset may be varied by using the following functions:

ri3156B_set_frequency()

ri3156B_set_amplitude()

ri3156B_set_offset()

You may set all sinc wave parameters at once, by using the ri3156B_apply_sinc_wave() function, or by using the individual functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_sinc_wave_numb_cycles (ViSession instrHandle, ViInt16 numberofCycles)
ViStatus ri3156b_query_sinc_wave_numb_cycles (ViSession instrHandle, ViInt16* numberofCycles)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
numberofCycles	ViInt16	Selects the “0” crossings for the sinc wave function. Valid range: 4 to 100 Default: 4

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_sinc_wave

Description

Programs frequency, amplitude, offset, and number of cycles of the sinc wave function. Also sets the sinc waveform as the active waveform. Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_sinc_wave (ViSession instrHandle, ViReal64 frequency, ViReal64 amplitude, ViReal64 offset, ViInt16 numberofCycles)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
frequency	ViReal64	Programs the frequency for the selected function Valid range: 0.01 to 25e6 (Hz) Default: 1e6
amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0
numberofCycles	ViInt16	Selects the “0” crossings for the sinc wave function. Valid range: 4 to 100 Default: 4

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_exponential_wave_exponent

Description

Programs only the exponent of the standard exponential wave. This function does NOT select the exponential wave as the active waveform. Note: The frequency, amplitude, and offset may be varied by using the following functions:

ri3156B_set_frequency()

ri3156B_set_amplitude()

ri3156B_set_offset()

You may set all exponential wave parameters at once, by using the ri3156B_apply_exponential_wave() function, or by using the individual functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_exponential_wave_exponent (ViSession instrHandle, ViInt16 exponent)
ViStatus ri3156b_query_exponential_wave_exponent (ViSession instrHandle, ViInt16* exponent)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
exponent	ViInt16	Selects the exponent for the exponential wave function. Valid range: -100 to 100 Default: -1

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_exponential_wave

Description

Programs frequency, amplitude, offset, and exponent of exponential wave function. Also sets the exponential waveform as the active waveform. Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_exponential_wave (ViSession instrHandle, ViReal64 frequency, ViReal64 amplitude, ViReal64 offset, ViInt16 exponent)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
frequency	ViReal64	Programs the frequency for the selected function Valid range: 0.01 to 25e6 (Hz) Default: 1e6
amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0
exponent	ViInt16	Selects the exponent for the exponential wave function. Valid range: -100 to 100 Default: -1

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_gaussian_wave_exponent

Description

Programs only the exponent of the standard gaussian wave. This function does NOT select the gaussian wave as the active waveform. Note: The frequency, amplitude, and offset may be varied by using the following functions:

ri3156B_set_frequency()

ri3156B_set_amplitude()

ri3156B_set_offset()

You may set all gaussian wave parameters at once, by using the ri3156B_apply_gaussian_wave() function, or by using the individual functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_gaussian_wave_exponent (ViSession instrHandle, ViInt16 exponent)
ViStatus ri3156b_query_gaussian_wave_exponent (ViSession instrHandle, ViInt16* exponent)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
exponent	ViInt16	Selects the exponent for the gaussian wave function. Valid range: -100 to 100 Default: -1

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_exponential_wave

Description

Programs frequency, amplitude, offset, and exponent of exponential wave function. Also sets the exponential waveform as the active waveform. Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_exponential_wave (ViSession instrHandle, ViReal64 frequency, ViReal64 amplitude, ViReal64 offset, ViInt16 exponent)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
frequency	ViReal64	Programs the frequency for the selected function Valid range: 0.01 to 25e6 (Hz) Default: 1e6
amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0
exponent	ViInt16	Selects the exponent for the exponential wave function. Valid range: -100 to 100 Default: -1

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_dc_signal_percent

Description

Programs only the percent level of the standard dc wave. This function does NOT select the dc wave as the active waveform. Note: The frequency, amplitude, and offset may be varied by using the following functions:

ri3156B_set_frequency()

ri3156B_set_amplitude()

ri3156B_set_offset()

You may set all dc wave parameters at once, by using the ri3156B_apply_dc_wave() function, or by using the individual functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_dc_signal_percent (ViSession instrHandle, ViInt16 percentAmplitude)
ViStatus ri3156b_query_dc_signal_percent (ViSession instrHandle, ViInt16* percentAmplitude)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
percentAmplitude	ViReal64	Selects the percent level for the dc function. Valid range: -100 to 100 Default: 100

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_exponential_wave

Description

Programs frequency, amplitude, offset, and exponent of dc function. Also sets the dc waveform as the active waveform. Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_dc_signal (ViSession instrHandle, ViReal64 frequency, ViReal64 amplitude, ViReal64 offset, ViReal64 percentAmplitude)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
percentAmplitude	ViReal64	Selects the percent level for the dc function. Valid range: -100 to 100 Default: 100

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

The Arbitrary Waveforms Functions Group

This group is used to control the 3156B arbitrary waveform shapes, their respective parameters, sample clock frequency, amplitude and offset. Also, within this group you find function calls that select, define and load active segments. If both channels are set to operate in arbitrary waveforms mode, each channel can be programmed independently to output different waveform shape. Parameter ranges and their defaults are given where applicable. The communication session is terminated with the close function.

Arbitrary Waveforms Programming			
3156B Function Name	**Parameter(s)**	**Range**	**Default**
ri3156B_set(query)_arb_sampling_freq_range	ViBoolean samplingClockRange	0-1	0
ri3156B_set(query)_arb_sampling_freq	ViReal64 samplingClock	1 to 200	50e6
Arbitrary Waveform Functions			
ri3156B_define_arb_segment	ViInt16 segmentNumber	1 to 16k	
	ViInt32 segmentSize	1 to 512e3, or	
		2 to 1e6	
ri3156B_delete_segment	ViInt16 segmentNumber	1 to 16k	
ri3156B_set(query)_active_segment	ViInt16 segmentNumber	1 to 16k	1
ri3156B_load_arb_data	ViInt16 segmentNumber	1 to 16k	
	ViInt16 dataPointArray[]	12 or 16 bit array	
	ViInt32 numberofPoints	1 to 512e3	
		2 to 1e6	
ri3156B_load_segment_table	ViInt16 numberOfSegments	1 to 16k	
	ViInt32 waveSize[]	1 to 512e3, or	
		2 to 1e6	
ri3156B_load_ascii_file	ViInt16 segmentNumber	1 to 16k	
	ViString fileName		
	ViInt32 numberofPoints	1 to 512e3, or	
		2 to 1e6	
ri3156B_set(query)_wave_format	ViBoolean waveFormat	0-1	0
ri3156B_set(query)_byte_order	ViBoolean byteOrder	0-1	0
ri3156B_apply_arb_waveform	ViInt16 segmentNumber	1 to 16k	1
	ViReal64 samplingClock	1 to 200e6	50e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0

Ri3156b_set(query)_arb_sampling_freq_range

Description

Proper setting of sample clock frequency range is essential for correct programming of the arbitrary waveform mode as it defines if incremental size of the waveform can be set with 1 or 2 points. The 3156B has two sample clock ranges 100MS/s and 200 MS/s. If you use waveform files that their size is divisible by 2, then you can use the 3156B with sample clock rates to 200 Ms/s. Waveform file that has an odd number of points can be used with sample clock rates to 100 MS/s only.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_arb_sampling_freq_range (ViSession instrHandle, ViBoolean samplingClockRange)
ViStatus ri3156b_query_arb_sampling_freq_range (ViSession instrHandle, ViBoolean * samplingClockRange)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
samplingClockRange	ViBoolean	Programs the sample clock range for the 3156B. Valid range: 0,1 0 – 1S/s to 200MS/s 1 – 1S/s to 100MS/s Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_arb_sampling_freq

Description

Sets the sample clock frequency. Users should be careful not to confuse waveform frequency with sample clock frequency. The waveform frequency parameter is valid for standard waveforms only and controls waveform frequency at the output connector. On the other hand, the sample clock frequency parameter is valid for arbitrary waveforms only and defines the frequency of which the generator clocks data points.

Standard waveform frequency is measured in units of Hz. Arbitrary waveform sample clock frequency is measured in units of S/s (samples per second). The frequency of a given arbitrary waveform at the output connector is computed using the sample clock frequency and the number of data points.

Use the following equation for computing the frequency of an arbitrary waveform:

Waveform Frequency = Sample Clock / Number of Data Points

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_arb_sampling_freq (ViSession instrHandle, ViReal64 samplingClock)
ViStatus ri3156b_query_arb_sampling_freq (ViSession instrHandle, ViReal64 * samplingClock)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
samplingClock	ViReal64	Programs the sample clock frequency for the 3156B. Valid range: 1 to 200e6 (S/s) Default: 50e6

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_define_arb_segment

Description

Use this function to attach size to a specific memory segment. The final size of the arbitrary memory is 1Meg points. The memory can be partitioned to smaller segments, up to 16k segments. This function allows definition of segment size. Total length of memory segments cannot exceed the size of 1Meg.Segment size is sample clock frequency dependent. Odd number of points can be used to 100 MS/s, even number of points can be used through the entire range of the product.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_define_arb_segment (ViSession instrHandle, ViInt16 segmentNumber, ViInt32 segmentSize)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
segmentNumber	ViInt16	Select the segment number. Valid range: 1 to 16k Default: 1

segmentSize	ViInt32	Select the segment size. Valid range: 1 to 523264 with SCLK setting of 100MS/s; 2 to 1046528 with SCLK setting of 200MS/s

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_delete_segment

Description

Deletes the specified arbitrary waveform memory segment or ALL segments (if the segment number specified is 0). Note that if a segment is deleted, the memory portion that belonged to this segment is no longer accessible. The next segment that is defined will be placed after the last defined memory segment. However, if the last segment is deleted, the next downloaded segment will be written on top of the deleted one. There is danger that by deleting segments often, large portions of memory will remain unused. It is therefore suggested that you periodically delete the entire memory and only reload waveforms that will be used.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_delete_segment (ViSession instrHandle, ViInt16 segmentNumber)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
segmentNumber	ViInt16	Select the segment number to be deleted. Selecting Segment 0 deletes ALL segments Valid range: 1 to 16k Default: 1

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_active_segment

Description

This function selects the memory segment to be used as the active segment for generating arbitrary waveforms. If the waveform mode is Arbitrary, this selects the waveform that is output by the 3156B.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_active_segment (ViSession instrHandle, ViInt16 segmentNumber)
ViStatus ri3156b_queryve_segment (ViSession instrHandle, ViInt16 * segmentNumber)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
segmentNumber	ViInt16	Select the segment number to become active Valid range: 1 to 16k Default: 1

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_arb_data

Description

Selects the arbitrary data segment and loads data into the segment. This function supports both 12-bit (normal) and 16-bit (user) downloads. The format of the data is selected using the ri3156B_set_wave_format () function. The byte order of the data is selected using the ri3156B_set_byte_order() function.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_load_arb_data (ViSession instrHandle, ViInt16 segmentNumber, ViInt16 _VI_FAR dataPointArray[],ViInt32 number_ofPoints)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
segmentNumber	ViInt16	Select the segment number to become active Valid range: 1 to 16k Default: 1
dataPointArray	ViInt16[]	Select an array of data to be downloaded to a specific memory segment. Data range: 0 to 0xFFF (12-bit) or 0 to 0xFFFF (16-bit)
number_ofPoints	ViInt32	Select the size of the segment to be loaded. Size must match the number of data points in the data array. Valid range: 1 to 523264 with SCLK setting of 100MS/s; 2 to 1046528 with SCLK setting of 200MS/s

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_segment _table

Description

This loads the segment table using the fast binary download method. Use this function only after you have loaded the 3156B memory with all of your waveform segments. This function call does not load waveforms to the memory; It is used to divide the memory to multiple segments in one function call. With this function, the programmer merely states the number of segments he wants to use and the size (in points) of each wave. This function is an alternative to calling the following sequence multiple times:

ri3156B_define_arb_segment() for segment #1

ri3156B_load_arb_data() (or equivalent) for wave #1

ri3156B_define_arb_segment() for segment #2

ri3156B_load_arb_data() for wave #2

ri3156B_define_arb_segment() for segment #N

ri3156B_load_arb_data() for wave #N

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_load_segment_table (ViSession instrHandle, ViInt16 number_ofSegments, ViInt32 _VI_FAR waveSize[])

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
number_ofSegments	ViInt16[]	Loads an array of numbers which define the segments that will be placed in the segment table Valid range: 1 to 16k
waveSize	ViInt32[]	Loads an array of numbers which define the respective sizes of the segment in the number_ofSegment array. Valid range: 1 to 523264 with SCLK setting of 100MS/s; 2 to 1046528 with SCLK setting of 200MS/s

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_ASCII_file

Description

This selects a specific segment and loads an external file name that contains waveform points in ASCII. Note that waveform points are normally loaded to the 3156B in binary. This function call does an automatic conversion of ASCII to binary before the data is loaded to the working memory and therefore accuracy of the ASCII file is expected to avoid conversion errors.

The file should be composed of ASCII numbers (integers only) separated by one or more white space characters (including end-of-line). Each number must be in the range -2048 to 2047 (for 12-bit download setting) or -32768 to 32767 (for 16-bit download setting). The value of -32768 will produce the minimum (most negative) voltage, and the value of 32767 will produce the maximum (most positive) voltage. The minimum and maximum voltages are determined by the "amplitude" and "offset" parameters of the "ri3156B_set_amplitude()" and "ri3156B_set_offset()" functions.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_load_ascii_file (ViSession instrHandle, ViInt16 segmentNumber, ViString fileName, ViInt32 number_ofPoints, ViBoolean fileResolution)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
fileName	ViString	Provides a complete path to the file location.
number_ofPoints	ViInt32	Select the size of the segment to be loaded. Size must match the number of data points in the ASCII file name. Valid range: 1 to 523264 with SCLK setting of 100MS/s; 2 to 1046528 with SCLK setting of 200MS/s
fileResolution	ViBoolean	Sets the waveform format for either 12-bit or 16-bit data file. Valid range: 0,1 0 – selects 16-bit; 1 – selects 12-bit Default: 0

Return Values
Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_wave_format

Description

This selects between 12-bit or 16-bit waveform formats. 16-bit is the default format as it gives an excellent resolution of 1/65536 vertical increments. 16-bit format is recommended for normal operation however, 12-bit format can be selected if you already have such files prepared and stored in your waveform library.

Channel Dependency: Independent

C Syntax
ViStatus ri3156b_set_wave_format (ViSession instrHandle, ViBoolean waveFormat)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be

		used to associate multiple instrument handles with a single instrument.
waveFormat	ViBoolean	Sets the waveform format for either 12-bit or 16-bit data file. Valid range: 0,1 0 – selects 16-bit; 1 – selects 12-bit Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_byte_order

Description

This specifies the byte order in which the binary data is sent to the device. By default, the binary data is sent in byte-high byte-low order. This function is useful only for binary block transfer.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_byte_order (ViSession instrHandle, ViBoolean byteOrder)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
byteOrder	ViBoolean	Sets the order of the binary data block. Valid range: 0,1 0 – selects high-low; 1 – selects low-high Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_arb_waveform

Description

Programs the generator to output an arbitrary waveform. This command lets you specify segment number, sampling clock, amplitude and offset and sets arbitrary to be the active waveform mode. You cannot use this function to load waveforms to memory segments.

Channel Dependency: Independent

C Syntax

```
ViStatus ri3156b_apply_arb_waveform (ViSession instrHandle, ViInt16 segmentNumber,
ViReal64 samplingClock, ViReal64 amplitude, ViReal64 offset)
```

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
segmentNumber	ViInt16	Select the segment number to become active Valid range: 1 to 16k Default: 1
samplingClock	ViReal64	Programs the sample clock frequency for the 3156B. Valid range: 1 to 200e6 (S/s) Default: 50e6
amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

The Sequenced Waveforms Functions Group

Sequence is built from arbitrary waveforms that are already resident in the working memory. The sequence is designed in a table – Sequence Table that defines steps, links and loop. The 3156B can link up to 4096 segments in one sequence table and loop each segment up to 1 million times. This group is used to control the 3156B sequenced waveform. Using these functions, you can define a sequence and manipulate sequence steps and table data. If both channels are set to operate in sequenced waveforms mode, each channel can be programmed independently to output different waveforms sequence. Parameter ranges and their defaults are given where applicable. The communication session is terminated with the close function.

Sequenced Waveforms Programming			
3156B Function Name	**Parameter(s)**	**Range**	**Default**
ri3156B_define_sequence_step	ViInt16 numberofStep	1 to 4096	1
	ViInt16 segmentNumber	1 to 16k	
	ViInt32 repeatSegment	1 to 1e6	
ri3156B_set(query)_sequence_mode	ViInt16 sequenceMode	0 to 2	0
ri3156B_delete_sequence_step	ViInt16 sequenceStep	0, 1 to 4096 (0 means delete all)	
ri3156B_load_sequence_table	ViInt16 numberofSteps	1 to 4096	1
	ViInt16 segment[]	1 to 16k	
	ViInt32 repetitions[]	1 to 1Meg	
ri3156B_apply_sequence_waveform	ViReal64 samplingClock	1 to 200e6	50e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViInt16 sequenceMode	0 to 2	0

Ri3156b_define_sequence_step

Description

Sequenced waveforms are made of a number of arbitrary waveforms, which can be linked and looped in user-programmable order. Sequenced waveforms are generated form waveforms stored in a library of memory segments (sequence table). Before using a sequence of waveforms, load arbitrary memory with the required waveforms. Note that this function call defines one step at a time. To define the sequence table in one shot, use the ri3156B_load_sequence_table() function call.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_define_sequence_step (ViSession instrHandle, ViInt16 number_ofStep, ViInt16 segmentNumber, ViInt32 repeatSegment)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
number_ofStep	ViInt16	Selects the step (link) number to be programmed in the sequence table. Valid range: 1 to 4096
segmentNumber	ViInt16	Assigns a waveform segment to the selected step (link) number. Valid range: Valid range: 1 to 16k
repeatSegment	ViInt32	Assigns a repeat count (loops) to the selected step (link) number. Valid range: Valid range: 1 to 1M

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_sequence_mode

Description

This function programs the manner in which the instrument advances through the sequence links and loops, and the source of the event causing sequence advance. There are three advance modes for the sequence table: Automatic (AUTO), Stepped (STEP) and Single.

AUTO specifies continuous advance, where the generator steps continuously to the end of the sequence table and repeats the sequence from the start. For example, if a sequence is made of there segments - 1, 2, and 3, and AUTO mode is used, the sequence will generate an infinite number of 1,2,3,1,2,3 ... waveforms. Of course, each link (segment) can be programmed with its associated loop (repeat) number and each segment can be programmed with a unique sample clock rate divider. AUTO is the default sequence advance mode.

In STEP advance mode, the sequence is advanced to the next waveform only when a valid trigger is received. The output of the 3156B generates the first segment continuously until a trigger signal advances the sequence to the next segment. If repeats were selected for a segment, the loop counter is executed automatically.

In Single advance mode, the 3156B idles between steps until a valid trigger signal is sensed. This mode operates with triggered mode only. An attempt to select the Single advance mode when the 3156B is in continuous operating mode will generate an error. After receiving a trigger, the generator outputs N loops, where N is the number of repeats programmed for that step. Then, the output level idles at a DC level equal to the last point of the last generated waveform. After completing the present step, the 3156B will accept a new trigger and advance to the next step.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_sequence_mode (ViSession instrHandle, ViInt16 sequenceMode)
ViStatus ri3156b_query_sequence_mode (ViSession instrHandle, ViInt16 * sequenceMode)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
sequenceMode	ViInt16	Selects the sequence advance mode. Valid range: 1 to 2 0 – defines AUTO 1 – defines STEP 2 – defines SINGLE Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_delete_sequence_step

Description

Deletes the specified step in a sequence table or ALL steps in one shot (if the segment number specified is 0).

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_delete_sequence_step (ViSession instrHandle, ViInt16 sequenceStep)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.

sequenceStep	ViInt16	Select the sequence step number to be deleted. Selecting 0 deletes ALL sequence steps and resets the sequence table Valid range: 1 to 4096

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_sequence _table

Description

This loads the sequence table using the fast binary download method. Use this function only after you have loaded the 3156B memory with all of your waveform segments. This function call does not load waveforms to the memory; It is used to build the sequence table only in one function call. With this function, the programmer merely states the number of links and their respective segment number and loop count.. This function is an alternative to calling the ri3156B_define_sequence_step() for multiple times:

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_load_sequence_table (ViSession instrHandle, ViInt16_numberofSteps, ViInt16_segment[], ViInt32_repetitions[])

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
numberofSteps	ViInt16	Defines how many steps (links) will be programmed in the sequence table Valid range: 1 to 4096
segment	ViInt16[]	Loads an array of numbers which define the segments that will be used for the sequence table. Valid range: 1 to 16k
repetition	ViInt32[]	Loads an array of numbers which define the respective number of repetitions (loops) for each segment that is placed in the sequence table. Valid range: 1 to 1M

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_sequence_waveform

Description

Programs the generator to output a sequence of waveforms. This command lets you specify sampling clock, amplitude, offset and advance mode. It also sets sequence to be the active waveform mode. You cannot use this function to load waveforms to the sequence table.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_sequence_waveform (ViSession instrHandle, ViReal64 samplingClock, ViReal64 amplitude, ViReal64 offset, ViInt16 sequenceMode)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
samplingClock	ViReal64	Programs the sample clock frequency for the 3156B. Valid range: 1 to 200e6 (S/s) Default: 50e6
amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0
sequenceMode	ViInt16	Selects the sequence advance mode. Valid range: 1 to 2 0 – defines AUTO 1 – defines STEP 2 – defines SINGLE Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

The Modulated Waveforms Functions Group

Use this group to define modulation type and their respective parameters and run modes. The 3156B can generate the following modulation: Sweep, FM, Frequency hops, FSK and AM. These functions are listed below. Note that if one channel is programmed to generate modulated waveform, the other channel must either generate the same modulation type or generate an ac, none-interrupted signal only. In modulation mode, both channels share the same run mode. Parameter ranges and their defaults are given where applicable. The communication session is terminated with the close function.

Modulated Waveforms Programming			
3156B Function Name	**Parameter(s)**	**Range**	**Default**
Sweep Programming			
ri3156B_set(query)_sweep_type	ViInt16 sweepType	0,1	0
ri3156B_set(query)_sweep_start	ViReal64 startFrequency	0.01 to 25e6	10e3
ri3156B_set(query)_sweep_stop	ViReal64 stopFrequency	0.01 to 25e6	1e6
ri3156B_set(query)_sweep_range	ViReal64 startFrequency	0.01 to 25e6	10e3
	ViReal64 stopFrequency	0.01 to 25e6	1e6
ri3156B_set(query)_sweep_time	ViReal64 sweepTime	1.4e-6 to 40	1
ri3156B_set(query)_sweep_direction	ViInt16 SweepDirection	0,1	0
ri3156B_set(query)_sweep_marker	ViReal64 markerFrequency	<sweep range>	(stop-start)/2
ri3156B_apply_sweep_waveform	ViReal64 startFrequency	0.01 to 25e6	10e3
	ViReal64 stopFrequency	0.01 to 25e6	1e6
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViInt16 sweepType	0,1	0
	ViReal64 sweepTime	1.4e-6 to 40	1
	ViInt16 SweepDirection	0,1	0
	ViReal64 markerFrequency	<sweep range>	(stop-start)/2
FM Programming			
ri3156B_set(query)_FM_carrier_freq	ViReal64 FMcarrierFrequency	0.01 to 25e6	1e6
ri3156B_set(query)_FM_mod_freq	ViReal64 FMmodulationFrequency	0.01 to 350e3	10e3
ri3156B_set(query)_FM_mod_waveform	ViInt16 FMmodulationWaveform	0 to 4	0
ri3156B_set(query)_FM_freq_dev	ViReal64 FMfrequencyDeviation	< carrier frequency	100e3
ri3156B_set(query)_FM_marker	ViReal64 FMmarkerFrequency	< deviation frequency>	Carrier freq
ri3156B_apply_FM_waveform	ViReal64 FMcarrierFrequency	0.01 to 25e6	1e6
	ViReal64 Amplitude	10e-3 to 10	5

	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 FMmodulationFrequency	0.01 to 350e3	10e3
	ViReal64 FMfrequencyDeviation	< carrier frequency	100e3
	ViInt15 FMmodulationWaveform	0 to 4	0
	ViReal64 FMmarkerFrequency	<carrier+/- dev frequency/2	Carrier freq

Modulated Waveforms Programming (continued)

3156B Function Name	**Parameter(s)**	**Range**	**Default**
ri3156B_set(query)_arb_FM_mod_sclk	ViReal64 FMarbModulationSCLK	1 to 5e6	1e6
ri3156B_load_arb_FM_mod_data	ViReal64 FMdataPointArray[]	0.01 to 25e6	
	ViInt32 FMnumberofPoints	10 to 32768	
ri3156B_apply_arb_FM_waveform	ViReal64 FMarbModulationSCLK	1 to 5e6	1e6
	ViReal64 Amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
Frequency Hopping Programming			
ri3156B_set(query)_hop_mode	ViBoolean hopMode	0,1	0
ri3156B_set(query)_hop_dwell_time	ViReal64 hopDwellTime	500e-9 to 21	500e-9
ri3156B_set(query)_hop_marker	ViInt16 hopMarkerIndex	0 to 4095	Last hop
ri3156B_load_fix_hop_freq_list	ViReal64 hopFreqList[]	0.01 to 25e6	
	ViInt16 hopFreqListSize	1 to 4096	
ri3156B_load_var_hop_freq_list	ViReal64 hopFreqList[]	0.01 to 25e6	
	ViInt16 hopFreqListSize	1 to 4096	
	ViReal64 hopDwellTimeList[]	dwell time list	
ri3156B_apply_hop_waveform	ViReal64 hopMode	0,1	0
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 hopDwellTime	500e-9 to 21	500e-9
	ViInt16 hopMarkerIndex	0 to 4095	Last hop
FSK Programming			
ri3156B_set(query)_FSK_one_frequency	ViReal64 FSKoneFrequency	0.01 to 25e6	1e6
ri3156B_set(query)_FSK_zero_frequency	ViReal64 FSKzeroFrequency	0.01 to 25e6	100e3
ri3156B_load_FSK_data	ViInt16 FSKwordLength	8 to 4096	8
	ViBoolean FSKdata[]	0,1	
ri3156B_set(query)_FSK_word_rate	ViReal64 FSKbaudRate	1 to 10e6	10e3
ri3156B_set(query)_FSK_marker	ViInt16 FSKmarkerIndex	1 to 4096	Last hop
ri3156B_apply_FSK_waveform	ViReal64 FSKbaudRate	1 to 10e6	10e3
	ViReal64 amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViReal64 FSKoneFrequency	0.01 to 25e6	1e6
	ViReal64 FSKzeroFrequency	0.01 to 25e6	100e3
	ViInt16 FSKmarkerIndex	1 to 4096	last hop

Modulated Waveforms Programming (continued)			
3156B Function Name	**Parameter(s)**	**Range**	**Default**
AM Programming			
ri3156B_set(query)_AM_carrier_freq	ViReal64 AMcarrierFrequency	0.01 to 25e6	1e6
ri3156B_set(query)_AM_mod_freq	ViReal64 AMmodulationFrequency	0.01 to 100e3	10e3
ri3156B_set(query)_AM_mod_freq_div	ViInt16 AMmodulationFreqDiv	2 to 4096	100
ri3156B_set(query)_AM_mod_depth	ViReal64 AMmodulationDepth	0 to 100	50
ri3156B_apply_AM_waveform	ViReal64 AMcarrierFrequency	0.01 to 25e6	1e6
	ViReal64 Amplitude	10e-3 to 10	5
	ViReal64 offset	-4.995 to 4.995	0
	ViInt16 AMmodulationFreqDiv	2 to 4096	100
	ViReal64 AMmodulationDepth	0 to 100	50

Ri3156b_set(query)_sweep_type

Description

This specifies the sweep step type. Two options are available: logarithmic or linear. In linear, the incremental steps between the frequencies are uniform throughout the sweep range. Logarithmic type defines logarithmic spacing throughout the sweep range.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_sweep_type (ViSession instrHandle, ViInt16 sweepType)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
SweepType	ViInt16	Sets the sweep step type. Valid range: 0,1 0 – selects linear step; 1 – selects logarithmic step Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_sweep_start

Description

This specifies the sweep start frequency. The 3156B will normally sweep from start to stop frequencies however, if the sweep direction is revered, the output will sweep from stop to start frequencies. To reverse sweep direction use the ri3156B_set_sweep_direction() function. The start and stop frequencies may be programmed freely throughout the frequency of the standard waveform frequency range.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_sweep_start (ViSession instrHandle, ViReal64 startFrequency)
ViStatus ri3156b_query_sweep_start (ViSession instrHandle, ViReal64 * startFrequency)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
startFrequency	ViReal64	Programs the sweep start frequency. Valid range: 0.01 to 25e6 (Hz) Default: 10e3

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_sweep_stop

Description

This specifies the sweep stop frequency. The 3156B will normally sweep from start to stop frequencies however, if the sweep direction is revered, the output will sweep from stop to start frequencies. To reverse sweep direction use the ri3156B_set_sweep_direction() function. The start and stop frequencies may be programmed freely throughout the frequency of the standard waveform frequency range.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_sweep_stop (ViSession instrHandle, ViReal64 stopFrequency)
ViStatus ri3156b_query_sweep_stop (ViSession instrHandle, ViReal64 * stopFrequency)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
stopFrequency	ViReal64	Programs the sweep stop frequency. Valid range: 0.01 to 25e6 (Hz) Default: 10e3

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_sweep_range

Description

This specifies simultaneously the sweep start and stop frequency. The 3156B will normally sweep from start to stop frequencies however, if the sweep direction is revered, the output will sweep from stop to start frequencies. To reverse sweep direction use the ri3156B_set_sweep_direction() function. The start and stop frequencies may be programmed freely throughout the frequency of the standard waveform frequency range.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_sweep_range (ViSession instrHandle, ViReal64 startFrequency, ViReal64 stopFrequency)
ViStatus ri3156b_query_sweep_range (ViSession instrHandle, ViReal64 * startFrequency, ViReal64 * stopFrequency)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
startFrequency	ViReal64	Programs the sweep start frequency. Valid range: 0.01 to 25e6 (Hz) Default: 10e3

stopFrequency	ViReal64	Programs the sweep stop frequency. Valid range: 0.01 to 25e6 (Hz) Default: 10e3

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_sweep_time

Description

This specifies the time that will take the 3156B to sweep from start to stop frequencies. The time does not depend on the sweep boundaries as it is automatically adjusted by the software to the required interval. At the end of the sweep cycle the output waveform maintains the sweep stop frequency setting.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_sweep_time (ViSession instrHandle, ViReal64 sweepTime)
ViStatus ri3156b_query_sweep_time (ViSession instrHandle, ViReal64 sweepTime)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
sweepTime	ViReal64	Programs the sweep time parameter. Valid range: 1.4E-6 to 40 (s) Default: 1

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_sweep_direction

Description

This specifies if the 3156B output will sweep from start-to-stop (UP) or from stop-to-start (DOWN) frequencies. Sweep time does not affect the sweep direction and frequency limits. At the end of the sweep cycle the output waveform normally maintains the sweep stop frequency setting but will maintain the start frequency, if the DOWN option is selected.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_sweep_direction (ViSession instrHandle, ViInt16 sweepDirection)
ViStatus ri3156b_query_sweep_direction (ViSession instrHandle, ViInt16 * sweepDirection)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
sweepDirection	ViInt16	Programs the sweep direction parameter. Valid range: 0,1 0 – selects sweep UP 1 – selects sweep DOWN Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_sweep_marker

Description

This function programs marker frequency position. Sweep marker can be placed in between the start and the stop frequencies. The marker pulse is output from the SYNC output connector. It can also be place on one of the TTLTrg0-7 lines using the appropriate ri3156B_set_TTLTRG_n_output_state () function.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_sweep_marker (ViSession instrHandle, ViReal64 markerFrequency)
ViStatus ri3156b_query_sweep_marker (ViSession instrHandle, ViReal64 * markerFrequency)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
markerFrequency	ViReal64	Programs the sweep marker frequency parameter. Valid range: Within the sweep start and stop frequencies Default: (Stop freq - Start freq) / 2

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_sweep_waveform

Description

Programs the generator to output a swept waveform. This command lets you specify all parameters to execute the sweep: start and stop frequencies, amplitude, offset, sweep type, time and direction as well as marker frequency. It also sets sweep to be the active waveform mode.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_sweep_waveform (ViSession instrHandle, ViReal64 startFrequency, ViReal64 stopFrequency, ViReal64 amplitude, ViReal64 offset, ViInt16 sweepType, ViReal64 sweepTime, ViInt16 sweepDirection, ViReal64 markerFrequency)

Parameters

Name	Variable Type	Description
startFrequency	ViReal64	Programs the sweep start frequency. Valid range: 0.01 to 25e6 (Hz) Default: 10e3
stopFrequency	ViReal64	Programs the sweep stop frequency. Valid range: 0.01 to 25e6 (Hz) Default: 10e3
amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5

offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0
SweepType	ViInt16	Sets the sweep step type. Valid range: 0,1 0 – selects linear step; 1 – selects logarithmic step Default: 0
sweepTime	ViReal64	Programs the sweep time parameter. Valid range: 1.4E-6 to 40 (s) Default: 1
sweepDirection	ViInt16	Programs the sweep direction parameter. Valid range: 0,1 0 – selects sweep UP 1 – selects sweep DOWN Default: 0
markerFrequency	ViReal64	Programs the sweep marker frequency parameter. Valid range: Within the sweep start and stop frequencies Default: (Stop freq - Start freq) / 2

Ri3156b_set(query)_FM_carrier_freq

Description

Sets the frequency for the FM carrier waveform. Carrier waveform is sinusoidal.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_FM_carrier_freq (ViSession instrHandle, ViReal64 FMcarrierFrequency)

ViStatus ri3156b_query_FM_carrier_freq (ViSession instrHandle, ViReal64 * FMcarrierFrequency)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be

		used to associate multiple instrument handles with a single instrument.
FMcarrierFrequency	ViReal64	Programs the FM carrier frequency. Valid Range: 0.01 to 25e6 Default Value: 1e6

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_FM_mod_freq

Description

Sets the frequency of the modulating waveform. The modulating waveform can be selected from 4 built-in waveforms: Sine, Triangle, Square and Ramp. An arbitrary waveform can also frequency modulate the carrier however, this waveform must be designed externally and downloaded to the arbitrary FM memory before it can be used for modulating the carrier.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_FM_mod_freq (ViSession instrHandle, ViReal64 FMmodulationFrequency)

ViStatus ri3156b_query_FM_mod_freq (ViSession instrHandle, ViReal64 * FMmodulationFrequency)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
FMmodulationFrequency	ViReal64	Programs the modulating waveform frequency. Valid Range: 0.01 to 350e3 Default Value: 10e3

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_FM_mod_waveform

Description

The modulating waveform can be selected from 4 built-in waveforms: Sine, Triangle, Ramp and Square. In addition, One may use and arbitrary waveform to modulate the carrier wave; In this case, an arbitrary waveform data has to be downloaded to the 3156B using the ri3156B_load_arb_FM_mod_data function. Use the description in the ri3156B_load_arb_FM_mod_data function to properly prepare and download data to the arbitrary FM waveform memory.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_FM_mod_waveform (ViSession instrHandle, ViReal64 FMmodulationWaveform)

ViStatus ri3156b_query_FM_mod_waveform (ViSession instrHandle, ViReal64 * FMmodulationWaveform)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
FMmodulationWaveform	ViInt16	Selects one of 4 built-in waveforms, or arbitrary waveform to modulate the carrier. Valid range: 0 to 4 0 – selects sine 1 – selects triangle 2 – selects ramp 3 – selects square 4 – selects arbitrary Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_FM_freq_dev

Description

This programs the deviation range around the carrier frequency. The deviation range is always symmetrical about the carrier frequency. If you need non-symmetrical deviation range, you can use the arbitrary FM composer screen or an external utility to design such waveforms.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_FM_freq_dev (ViSession instrHandle, ViReal64 FMfrequencyDeviation)

ViStatus ri3156b_query_FM_freq_dev (ViSession instrHandle, ViReal64 * FMfrequencyDeviation)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
FMfrequencyDeviation	ViReal64	Programs the deviation frequency range. Valid range: Valid Range: 0<(Carrier Frequency ± deviation frequency / 2)<25MHz. Default: 100e3

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_FM_marker

Description

This function programs marker frequency position. FM marker can be placed inside the following range: (carrier frequency ± deviation frequency / 2). The marker pulse is output from the SYNC output connector. It can also be place on one of the TTLTrg0-7 lines using the ri3156B_set_TTLTRG_n_output_state () function.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_FM_marker (ViSession instrHandle, ViReal64 FMmarkerFrequency)
ViStatus ri3156b_query_FM_marker (ViSession instrHandle, ViReal64 * FMmarkerFrequency)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
FMmarkerFrequency	ViReal64	Programs the FM marker frequency parameter. Valid range: Within (carrier frequency ± deviation frequency / 2) Default: Carrier frequency setting

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_FM_waveform

Description

Programs the generator to output frequency modulated waveform. This command lets you specify all parameters to execute the carrier frequency, deviation range, amplitude, offset, modulating waveform shape and its frequency as well as frequency of the marker. It also sets FM to be the active waveform mode.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_sweep_waveform (ViSession instrHandle, ViReal64 startFrequency, ViReal64 stopFrequency, ViReal64 amplitude, ViReal64 offset, ViInt16 sweepType, ViReal64 sweepTime, ViInt16 sweepDirection, ViReal64 markerFrequency)

Parameters

Name	Variable Type	Description
FMcarrierFrequency	ViReal64	Programs the FM carrier frequency. Valid Range: 0.01 to 25e6 Default Value: 1e6
amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0

FMmodulationFrequency	ViReal64	Programs the modulating waveform frequency. Valid Range: 0.01 to 350e3 Default Value: 10e3
FMfrequencyDeviation	ViReal64	Programs the deviation frequency range. Valid range: Valid Range: 0<(Carrier Frequency ± deviation frequency / 2)<25MHz. Default: 100e3
FMmodulationWaveform	ViInt16	Selects one of 4 built-in waveforms, or arbitrary waveform to modulate the carrier. Valid range: 0 to 4 0 – selects sine 1 – selects triangle 2 – selects ramp 3 – selects square 4 – selects arbitrary Default: 0
FMmarkerFrequency	ViReal64	Programs the FM marker frequency parameter. Valid range: Within (carrier frequency ± deviation frequency / 2) Default: Carrier frequency setting

Return Values
Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_arb_FM_mod_sclk

Description
This programs the modulating wave sample clock frequency. This parameter affects the 3156B when set to operate in arbitrary FM mode only. The arbitrary FM sample clock frequency parameter does not modify the sample clock of the arbitrary waveform function. The frequency of the modulating signal can be determined from the following relationship:

Modulation frequency = FM modulation SCLK / Number of Points in the arbitrary modulating waveform

The modulating waveform can be created by WaveCAD or generated as a data file array using the function ri3156B_load_arb_FM_mod_data

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_arb_FM_mod_sclk (ViSession instrHandle, ViReal64 FMArbModulationSCLK)

ViStatus ri3156b_query_arb_FM_mod_sclk (ViSession instrHandle, ViReal64 * FMArbModulationSCLK)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
FMArbModulationSCLK	ViReal64	Programs the sample clock for the arbitrary modulating signal. Valid Range: 1 to 5e6 (S/s) Default Value: 1e6

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_arb_FM_mod_data

Description

This command will download FM modulating waveform data to the arbitrary FM memory. Below you can see how such data array is constructed. Downloading data to the arbitrary FM waveform memory is very different than loading arbitrary waveform data. Waveform data programs amplitude domain therefore, every point programs an amplitude level. On the other hand, FM modulating waveform data programs frequency domain therefore, every point sets different sample clock frequency.

Arbitrary FM waveform sample file. This file has 80 frequency points of that will create the following output modulation:
1. Stable 1MHz signal for 11 arbitrary FM SCLK intervals,
2. Sine shaped waveform, minimum frequency is 900047Hz and maximum frequency is 1.09995e+006Hz
3. Stable 1MHz signal for 19 arbitrary FM SCLK intervals.

The frequency of the modulating signal is computed from the following equation:

Modulation frequency = FM modulation SCLK / Number of Points in the arbitrary modulating waveform

The modulating waveform sample clock is programmed using the ri3156B_set_arb_FM_mod_sclk

Channel Dependency: Independent

Sample Arbitrary Modulating Waveform Data Array

1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1.01229e+006 1.02439e+006 1.03612e+006 1.04731e+006 1.05778e+006 1.06737e+006 1.07594e+006 1.08336e+006 1.08952e+006 1.09432e+006 1.09768e+006 1.09957e+006 1.09995e+006 1.09882e+006 1.09618e+006 1.09209e+006 1.0866e+006 1.0798e+006 1.07179e+006 1.06269e+006 1.05264e+006 1.0418e+006 1.03032e+006 1.01837e+006 1.00616e+006 993844 981625 969685 958204 947357 937308 928209 920198 913397 907909 903817 901183 900047 900427 902315 905685 910484 916640 924060 932630 942223 952691 963876 975609 987711 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006

C Syntax

ViStatus ri3156b_set_arb_FM_mod_data (ViSession instrHandle, ViReal64 FMDataPointArray[], ViInt32 FMNumber_ofPoints)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
FMDataPointArray	ViReal64[]	Points to an array of waveform points. Valid Range: array of values from 0.01 to 25e6 (Hz)
FMNumber_ofPoints	ViInt32	Defines the number of points for the arbitrary modulating signal. This number must match the number of frequencies in the data array. Valid Range: 10 to 32768 (waveform points)

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_arb_FM_waveform

Description

Programs the generator to output frequency modulated waveform where the modulating signal is an arbitrary waveform. This command lets you specify all parameters to execute the arbitrary FM sample clock, amplitude, and offset. Marker frequency can be programmed using the ri3156B_set_FM_marker() function. It also sets arbitrary FM to be the active waveform mode.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_arb_FM_waveform (ViSession instrHandle, ViReal64 amplitude, ViReal64 offset, ViReal64 arbModulationSCLK)

Parameters

Name	Variable Type	Description
amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0
FMArbModulationSCLK	ViReal64	Programs the sample clock for the arbitrary modulating signal. Valid Range: 1 to 5e6 (S/s) Default Value: 1e6

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_hop_mode

Description

This selects between fixed or variable dwell-time for the frequency hops. Select the fixed option if you want each frequency to dwell equally on each step. The variable option lets you program different dwell times for each frequency hop. The 3156B output hops from one frequency to the next according to a sequence given in a hope table. The variable dwell time table contains dwell time data for each step however, the fixed dwell time table does not contain and dwell time information and therefore, use the ri3156B_set_hop_dwell_time() function for the fixed mode.

Channel Dependency: independent

C Syntax

ViStatus ri3156b_set_hop_mode (ViSession instrHandle, ViBoolean hopMode)

ViStatus ri3156b_query_hop_mode (ViSession instrHandle, ViBoolean * hopMode)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
hopMode	ViBoolean	Selects between two frequency hops options: Fixed dwell time and Variable dwell time. Valid range: 0,1 0 – selects fixed dwell time frequency hops mode 1 – selects variable dwell time frequency hops mode Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_hop_dwell_time

Description

This selects the dwell time for frequency hops when the selected mode is Fixed dwell time hops. The dwell time table in this case does not contain the dwell time per step parameters and therefore, the value which is programmed with this function remains constant for the entire hop sequence.

Channel Dependency: independent

C Syntax

ViStatus ri3156b_set_hop_dwell_time (ViSession instrHandle, ViReal64 hopDwellTime)

ViStatus ri3156b_query_hop_dwell_time (ViSession instrHandle, ViReal64 * hopDwellTime)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
hopDwellTime	ViReal64	Programs dwell time for the fixed dwell-time frequency hop function. The same dwell time will be valid for each frequency hop. ***NOTE***: Dwell time is programmed in units of μs (microsecond). For example, to program 1.37ms dwell time, enter the value as follows: 1370 Valid range: 0.5 to 21e6 (μs) Default: 0.5

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_hop_marker

Description

Programs where on the frequency list the 3156B will generate a pulse, designated as Hop marker, or index point. The marker pulse is generated at the SYNC output connector. It can also be place on one of the backplane trigger lines using the ri3156B_ set_TTLTRG_n_output_state function call.

Channel Dependency: independent

C Syntax

ViStatus ri3156b_set_hop_marker (ViSession instrHandle, ViInt16 hopMarkerIndex)

ViStatus ri3156b_query_hop_marker (ViSession instrHandle, ViInt16 * hopMarkerIndex)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.

hopMarkerIndex	ViInt16	Selects the hop step index for the marker output. Valid range: 1 to 4096 (without exceeding the physical length of the hop table) Default: Frequency equal to the last hop frequency

Return Values
Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_fix_hop_freq_list

Description
This command will download the data array that will cause the instrument to hop through the frequency list. The dwell time for each frequency list item is fixed and can be programmed using the ri3156B_set_hop_dwell_time() function call. Note that if you intend to program marker position, you must do it first and then load the frequency hops list.

Below you can see how a hop table is constructed. The file sample below shows a list of 10 frequencies. The 3156B will hop through this list, outputting the next frequency each time it hops. Note that the carrier waveform is always sinewave and that the last cycle is always completed even if the dwell time is shorter than the period of the waveform. For example, if you program dwell time of 1ms and the frequency step has frequency of 1Hz (1s period), the frequency step will last 1 second although the dwell time is 1ms.

Channel Dependency: Independent

Sample Frequency Hops Data Array
1e+6 2e+6 3e+3 4e+6 5e+5 6e+2 7e+1 8e+6 9e+3 10e+5

C Syntax
ViStatus ri3156b_load_fix_hop_freq_list (ViSession instrHandle, ViReal64 hopFrequencyList[], ViInt16 hopFreqListSize)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
hopFrequencyList	ViReal64[]	Points to an array of frequency values. Valid Range: Array of values from 0.01 to 25e6 (Hz)
hopFreqListSize	ViInt16	Defines the number of frequency steps are included in the frequency hops table. This number must match the number of frequencies in the data array. Valid Range: 1 to 4096

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_var_hop_freq_list

Description

This command will download the data array that will cause the instrument to hop through the frequency list. The dwell time for each frequency list item is variable and is supplied as an array of dwell time values. Note that the ri3156B_set_hop_dwell_time() function call has no effect on this sequence. Note that if you intend to program marker position, you must do it first and then load the frequency hops list.

Below you can see how a hop table is constructed. The file sample below shows a list of 10 frequencies and their associated dwell times. The 3156B will hop through this list, outputting the next frequency each time it hops. Note that the carrier waveform is always sinewave and that the last cycle is always completed even if the dwell time is shorter than the period of the waveform. For example, if you program dwell time of 1ms and the frequency step has frequency of 1Hz (1s period), the frequency step will last 1 second although the dwell time is 1ms.

Channel Dependency: Independent

Sample Frequency Hops Data Arrays

Frequency Hops Array: 1e+6 2e+6 3e+3 4e+6 5e+5 6e+2 7e+1 8e+6 9e+3 10e+5
Dwell Time Array: 100 2000 3e4 40 5e3 6000 0.7 8e2 90 1000

C Syntax

ViStatus ri3156b_load_fix_hop_freq_list (ViSession instrHandle, ViReal64 _hopFrequencyList[], ViInt16 hopFreqListSize), ViReal64 _hopDwellTimeList[]

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
hopFrequencyList	ViReal64[]	Points to an array of frequency values. Valid Range: Array of values from 0.01 to 25e6 (Hz)
hopFreqListSize	ViInt16	Defines the number of frequency steps are included in the frequency hops table. This number must match the number of frequencies in the data array and in the dwell times array. Valid Range: 1 to 4096

hopDwellTimeList	ViReal64[]	Points to an array of dwell time values. Each dwell time is associated with a different frequency hop and therefore, the array size must match the number of frequency hops. ***NOTE***: Dwell time is programmed in units of μs (microsecond). For example, to program 1.37ms dwell time, enter the value as follows: 1370 Valid range: 0.5 to 21e6 (μs) Default: 0.5

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_hop_waveform

Description

Programs the generator to output frequency hops. This command lets you specify all parameters to execute the frequency hop mode, amplitude, offset and marker frequency It also sets frequency hops to be the active waveform mode.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_hop_waveform (ViSession instrHandle, ViBoolean hopMode, ViReal64 amplitude, ViReal64 offset, ViReal64 hopDwellTime, ViInt16 hopMarkerIndex)

Parameters

Name	**Variable Type**	**Description**
hopMode	ViBoolean	Selects between two frequency hops options: Fixed dwell time and Variable dwell time. Valid range: 0,1 0 – selects fixed dwell time frequency hops mode 1 – selects variable dwell time frequency hops mode Default: 0
amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0

hopDwellTime	ViReal64	Programs dwell time for the fixed dwell-time frequency hop function. The same dwell time will be valid for each frequency hop. ***NOTE***: Dwell time is programmed in units of μs (microsecond). For example, to program 1.37ms dwell time, enter the value as follows: 1370 Valid range: 0.5 to 21e6 (μs) Default: 0.5
hopMarkerIndex	ViInt16	Selects the hop step index for the marker output. Valid range: 1 to 4096 (without exceeding the physical length of the hop table) Default: Frequency equal to the last hop frequency

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_FSK_one_frequency

Description

This programs the shifted frequency. The frequency shifts when the pointer in the data array points to "1".

Channel Dependency: independent

C Syntax

ViStatus ri3156b_set_FSK_one_frequency (ViSession instrHandle, ViReal64 FSKOneFrequency)

ViStatus ri3156b_query_FSK_one_frequency (ViSession instrHandle, ViReal64 * FSKOneFrequency)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
FSKOneFrequency	ViReal64	Selects the frequency where the 3156B will shift when the controlling bit is one. Valid range: 0.01 to 25e6 (Hz) Default: 1e6

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_FSK_zero_frequency

Description

This programs the carrier frequency. The frequency remains at this frequency as long as the pointer in the data array points to “0”.

Channel Dependency: independent

C Syntax

ViStatus ri3156b_set_FSK_zero_frequency (ViSession instrHandle, ViReal64 FSKzeroFrequency)

ViStatus ri3156b_query_FSK_zero_frequency (ViSession instrHandle, ViReal64 * FSKzeroFrequency)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
FSKzeroFrequency	ViReal64	Selects the frequency where the 3156B will remain when the controlling bit is zero. Valid range: 0.01 to 25e6 (Hz) Default: 1e6

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_FSK_data

Description

Loads the data stream that will cause the 3156B to hop from carrier to shifted frequency and visa versa. Data format is a string of "0" and "1" which define when the output generates carrier frequency and when it shifts frequency to the FSK value. "0" defines carrier frequency,"1" defines shifted frequency. Note that if you intend to program marker position, you must do it before you load the FSK data list.

Below you can see how an FSK data table is constructed. The sample below shows a list of 10 shifts. The 3156B will step through this list, outputting either carrier or shifted frequencies, depending on the data list: Zero will generate carrier frequency and One will generate shifted frequency. Note that the waveform is always sinewave and that the last cycle is always completed.

Channel Dependency: Independent

Sample FSK Data Array

0 1 1 1 0 1 0 0 0 1

C Syntax

ViStatus ri3156b_load_FSK_data (ViSession instrHandle, ViInt16 FSKWordLength, ViBoolean FSKData[])

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
FSKWordLength	ViInt16	Defines the length of the FSK data array. This number must match the number of shifts in the data array Valid Range: 8 to 4096 Default: 8
FSKData	ViBoolean[]	Defines a stream of data, similar to the example above, which will cause the 3156B to hop from carrier frequency to shifted frequency value. Valid Range: 0 or 1

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_FSK_word_rate

Description

This allows the user to select FSK word rate. The word rate is the interval of which the bit streams in the FSK data array are clocked causing the output frequency is hop from carrier to shifted frequency values and visa versa.

Channel Dependency: independent

C Syntax

ViStatus ri3156b_set_FSK_word_rate (ViSession instrHandle, ViReal64 FSKBaudRate)

ViStatus ri3156b_query_FSK_word_rate (ViSession instrHandle, ViReal64 * FSKBaudRate)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
FSKBaudRate	ViReal64	Programs the FSK baud rate. Valid range: 1 to 10e6 (bit/s) Default: 10e3

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_FSK_marker

Description

Programs where on the data stream the 3156B will generate a pulse, designated as FSK marker, or index point. The marker pulse is generated at the SYNC output connector. It can also be place on one of the backplane trigger lines using the ri3156B_ set_TTLTRG_n_output_state function call.

Channel Dependency: independent

C Syntax

ViStatus ri3156b_set_FSK_marker (ViSession instrHandle, ViInt16 FSK_marker_index)

ViStatus ri3156b_query_FSK_marker (ViSession instrHandle, ViInt16 * FSK_marker_index)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.

FSK_marker_index	ViInt16	Programs a marker pulse at an index bit position. Valid range: 1 to 4096 (without exceeding the physical length of the FSK data stream) Default: Frequency equal to the last hop frequency

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_FSK_waveform

Description

Programs the generator to output FSK output. This command lets you specify all parameters to execute the FSK baud rate, amplitude, offset carrier and shifted frequencies and marker index. It also sets FSK to be the active waveform mode.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_FSK_waveform (ViSession instrHandle, ViReal64 FSK_baudRate, ViReal64 amplitude, ViReal64 offset, ViReal64 FSKOneFrequency, ViReal64 FSKZeroFrequency, ViInt16 FSKMarkerIndex)

Parameters

Name	Variable Type	Description
hopMode	ViBoolean	Selects between two frequency hops options: Fixed dwell time and Variable dwell time. Valid range: 0,1 0 – selects fixed dwell time frequency hops mode 1 – selects variable dwell time frequency hops mode Default: 0
FSKBaudRate	ViReal64	Programs the FSK baud rate. Valid range: 1 to 10e6 (bit/s) Default: 10e3
amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0

FSKOneFrequency	ViReal64	Selects the frequency where the 3156B will shift when the controlling bit is one. Valid range: 0.01 to 25e6 (Hz) Default: 1e6
FSKzeroFrequency	ViReal64	Selects the frequency where the 3156B will remain when the controlling bit is zero. Valid range: 0.01 to 25e6 (Hz) Default: 1e6
FSK_marker_index	ViInt16	Programs a marker pulse at an index bit position. Valid range: 1 to 4096 (without exceeding the physical length of the FSK data stream) Default: Frequency equal to the last hop frequency

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_AM_carrier_frequency

Description

This programs the carrier frequency for the amplitude modulated function.

Channel Dependency: independent

C Syntax

ViStatus ri3156b_set_AM_carrier_freq (ViSession instrHandle, ViReal64 AMCarrierFrequency)

ViStatus ri3156b_query_AM_carrier_freq (ViSession instrHandle, ViReal64 * AMCarrierFrequency)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
AMCarrierFrequency	ViReal64	Programs the carrier frequency. Valid range: 0.01 to 25e6 (Hz) Default: 1e6

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_AM_mod_frequency

Description

This programs a frequency divider. The ratio parameter divides the carrier frequency. The divider accepts integers only. The resultant value is used as the frequency of the modulating waveform. Note, however, that it is possible to program the modulating frequency directly, using the ri3156B_set_AM_mod_freq_div() parameter. The 3156B will accept the last parameter sent to it and therefore refrain from sending two parameters as only the last will be accepted.

Channel Dependency: independent

C Syntax

ViStatus ri3156b_set_AM_mod_freq_div (ViSession instrHandle, ViInt16 AMModulationFrequencyDivider)

ViStatus ri3156b_query_AM_mod_freq_div (ViSession instrHandle, ViInt16 * AMModulationFrequencyDivider)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
AMModulationFrequencyDivider	ViInt16	Programs the frequency divider ratio for the modulating frequency parameter. Valid range: 2 to 4096 and (Carrier Frequency/Frequency Divider) within Modulation Frequency range Default: 100

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_AM_mod_freq_div

Description

Programs a frequency divider for the modulating waveform. The ratio parameter divides the carrier frequency. The divider accepts integers only. The resultant value is used as the frequency of the modulating waveform. Note, however, that it is possible to program the modulating frequency directly, using the ri3156B_set_AM_mod_freq() parameter. The 3156B will accept the last parameter sent to it and therefore refrain from sending two parameters as only the last will be accepted.

This allows the user to program the modulation frequency. Note, however, that it is possible to program the modulating frequency using the parameter. The 3156B will accept the last parameter sent to it and therefore refrain from sending two parameters as only the last will be accepted.

Channel Dependency: independent

C Syntax

ViStatus ri3156b_set_AM_mod_freq_div (ViSession instrHandle, ViInt16 AMModulationFrequencyDivider)

ViStatus ri3156b_query_AM_mod_freq_div (ViSession instrHandle, ViInt16 * AMModulationFrequencyDivider)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
AMModulationFrequencyDivider	ViInt16	Programs the frequency divider ratio for the modulating frequency parameter. Valid range: 2 to 4096 Default: 100

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_AM_mod_depth

Description

This programs the modulation depth for the amplitude modulation programs.

Channel Dependency: independent

C Syntax

ViStatus ri3156b_set_AM_mod_depth (ViSession instrHandle, ViReal64 AMModulationDepth)

ViStatus ri3156b_query_AM_mod_depth (ViSession instrHandle, ViReal64 * AMModulationDepth)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
AMModulationDepth	ViReal64	Programs the modulation depth for the amplitude modulation. Valid range: 0 to 100 (%) Default: 50

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_AM_waveform

Description

Programs the generator to output amplitude modulated waveforms. This command lets you specify all parameters to execute the carrier frequency, amplitude, offset modulation frequency divider and modulation depth. It also sets AM to be the active waveform mode.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_AM_waveform (ViSession instrHandle, ViReal64 AMCarrierFrequency, ViReal64 amplitude, ViReal64 offset, ViInt16 AMModulationFrequencyDivider, ViReal64 AMModulationDepth)

Parameters

Name	Variable Type	Description
AMCarrierFrequency	ViReal64	Programs the carrier frequency. Valid range: 0.01 to 25e6 (Hz) Default: 1e6

amplitude	ViReal64	Programs the amplitude for the selected function Valid range: 10e-3 to 10 (V, into 50Ω) Default:5
offset	ViReal64	Programs the offset for the selected function Valid range: -4.995 to 4.995 (V, into 50Ω) Default: 0
AMModulationFrequencyDivider	ViInt16	Programs the frequency divider ratio for the modulating frequency parameter. Valid range: 2 to 4096 Default: 100
AMModulationDepth	ViReal64	Programs the modulation depth for the amplitude modulation. Valid range: 0 to 100 (%) Default: 50

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

The Digital Patterns Functions Group

Use this group to generate digital patterns from the 3156B. Although the instrument can generate 16-bit waveforms, the digital patterns are generated from 12 bits only (each channel). The digital outputs are always active however, if no termination is hooked on the outputs, the signal will not be seen at the terminals. The digital outputs generate 100k series ECL level, terminated into 50Ω to -2V.

Parameter ranges and their defaults are given where applicable. The communication session is terminated with the close function.

Digital Patterns Programming			
3156B Function Name	**Parameter(s)**	**Range**	**Default**
Patterns - Free Running Programming			
ri3156B_set(query)_dig_pattern_rate_range	ViInt16 digitalPatternRateRange	0, 1 (100Mpps, 50Mpps)	100e6
ri3156B_set(query)_dig_pattern_rate	ViReal64 digitalPatternRate	1 to 50/100e6	10e3
ri3156B_load_dig_pattern_stim_list	ViInt16 digitalPatternStimList[]	array of values 0 to 0xFFF	
	ViInt32 digPatternStimListSize	Digital Pattern Stim List length, 1 to 512k	
	ViInt32 digitalPatternHoldCountList[]	array of values 1 to 1.049B < 50MS/s, 2 to 2.1B 50MS/s-100MS/s	
ri3156B_apply_digital_pattern	ViReal64 digitalPatternRate	1 to 50/100e6	10e3

	ViInt16 digitalPatternRateRange	0, 1	0
Patterns - Stimulus Programming			
ri3156B_set(query)_dig_stim_freq	ViReal64 digitalDataFrequency	1 to 100e6	10e3
ri3156B_load_dig_data_stim_list	ViInt16 digitalDataStimList[]	array of values 0-0xFFF	
	ViInt32 digitalDataStimListSize	Digital Data Stim List Length. 1 to 512e3	
ri3156B_apply_digital_data	ViReal64 digitalDataFrequency	1 to 100e6	10e3

Ri3156b_set(query)_dig_pattern_rate_range

Description

Depending on the minimum pattern size and resolution, Freerun digital patterns have two rate ranges: 1pps to 50Mpps and 1pps to 100Mpps. The two ranges are characterized as follows:

1pps to 50Mpps can generate patterns that have odd number of patterns and minimum number of patterns is 1;

1pps to 100Mpps generate patterns that have even number of patterns and minimum number of patterns is 2.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_dig_pattern_rate_range (ViSession instrHandle, ViInt16 digitalPatternRateRange)

ViStatus ri3156b_query_dig_pattern_rate_range (ViSession instrHandle, ViInt16 * digitalPatternRateRange)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
digitalPatternRateRange	ViInt16	Programs the freerun digital patterns rate range Valid range: 0, 1 0 – selects the 1pps to 100Mpps range 1 – selects the 1pps to 50Mpps range Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_dig_pattern_rate

Description

This programs the rate of which the Freerun digital patterns will change. Note that the rate depends on the rate range. The rate range can be programmed using the Ri3156b_set_dig_pattern_rate_range function call.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_dig_pattern_rate (ViSession instrHandle, ViReal64 digitalPatternRate)

ViStatus ri3156b_query_dig_pattern_rate (ViSession instrHandle, ViReal64 * digitalPatternRate)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
digitalPatternRate	ViReal64	Programs the Freerun digital patterns rate Valid range: 1 to 50/100e6 (pps) Default: 10e3

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_dig_pattern_stim_list

Description

This command will download the data arrays for the Freerun digital patterns. For freerun digital patterns, one can program variable hold count for each pattern. The hold count defines the dwell time per pattern.

Below you can see how a Freerun digital pattern table is constructed. The sample below shows a list of 10 patterns and their associated hold counts. The 3156B will step through this list, outputting the next pattern each time it hop to the next step at a rate programmed by the ri3156B_set_dig_patter_rate() function call and the hold count set by the ViInt32 digitalPatternHoldCountList[]) argument.

Channel Dependency: Independent

Sample Digital Patters Arrays – Freerun Mode

Digital Pattern Array: 0x001 0x100 0x203 0x400 0x805 0xD00 0xA07 0x118 0x0FD 0x010
Hold Count Array: 100 200 333 44 50 600 707 8 90 1000

C Syntax

ViStatus ri3156b_load_dig_pattern_stim_list (ViSession instrHandle, ViInt16 digitalPatternStimList[],ViInt32 digPatternStimListSize, ViInt32 digitalPatternHoldCountList[])

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
digitalPatternStimList	ViInt16[]	Points to an array of patterns. Pattern values are in hex format. Valid Range: 0 to 0xFFF (12-bit)
digPatternStimListSize	ViInt32	Defines the number of digital pattern steps are included in the table. This number must match the number of patterns in the data and in the hold count arrays. Valid Range: 1 to 523264
digitalPatternHoldCountList	ViInt32[]	Points to an array of hold count values. Each hold count is associated with a pattern step and therefore, the array size must match the number of patterns. Valid range: Hold count depends on the rate range and the programmed channel. The range is 1 to 1.049B if rate range 50Mpps and 2 to 2.1B if rate range 100Mpps.In channel 2 max hold count = 16000000 if rate range is 100Mpps and 8000000 if rate range is 50Mpps

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_digital_pattern

Description

Programs all digital pattern parameters simultaneously: Digital Pattern Rate Range, Digital Pattern Rate. It also sets active channel Waveform Mode to Digital Pattern and Digital Mode to Freerun.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_digital_pattern (ViSession instrHandle, ViReal64 digitalPatternRate, ViInt16 digitalPatternRateRange)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
digitalPatternRate	ViReal64	Programs the Freerun digital patterns rate Valid range: 1 to 50/100e6 (pps) Default: 10e3
digitalPatternRateRange	ViInt16	Programs the freerun digital patterns rate range Valid range: 0, 1 0 – selects the 1pps to 100Mpps range 1 – selects the 1pps to 50Mpps range Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_dig_stim_freq

Description

This programs the rate of which the Stimulus digital patterns will change.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_dig_stim_freq (ViSession instrHandle, ViReal64 digitalDataFrequency)

ViStatus ri3156b_query_dig_stim_freq (ViSession instrHandle, ViReal64 * digitalDataFrequency)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
digitalDataFrequency	ViReal64	Programs the stimulus digital patterns frequency. Valid range: 1 to 100e6 (pps) Default: 10e3

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_dig_data_stim_list

Description

This command will download the data arrays for the Stimulus digital patterns. For stimulus digital patterns, the hold time is fixed at the rate set by the frequency parameter. For variable hold count you may want to use the Freerun digital patterns option.

Below you can see how a Stimulus digital pattern table is constructed. The sample below shows a list of 10 patterns. The 3156B will step through this list, outputting the next pattern each time it hop to the next step at a rate programmed by the ri3156B_set_stim_freq () function call.

Channel Dependency: Independent

Sample Digital Patters Arrays – Stimulus Mode

Digital Pattern Array: 0x001 0x100 0x203 0x400 0x805 0xD00 0xA07 0x118 0x0FD 0x010

C Syntax

ViStatus ri3156b_load_dig_data_stim_list (ViSession instrHandle, ViInt16 digitalDataStimList[], ViInt32 digitalDataStim_listSize)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
digitalDataStimList	ViInt16[]	Points to an array of patterns. Pattern values are in hex format. Valid Range: 0 to 0xFFF (12-bit)
digitalDataStim_listSize	ViInt32	Defines the number of digital pattern steps are included in the table. This number must match the number of patterns in the data array. Valid Range: 1 to 523264

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_digital_data

Description

Programs all stimulus digital pattern parameters simultaneously. It also sets active channel Waveform Mode to Digital Pattern and Digital Mode to Stimulus.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_apply_digital_data (ViSession instrHandle, ViReal64 digitalDataFrequency)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
digitalDataFrequency	ViReal64	Programs the stimulus digital patterns frequency. Valid range: 1 to 100e6 (pps) Default: 10e3

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

The Video Stroke Functions Group

Use this group to generate video stroke characters. There are 10 built-in characters built into the 3156B library. Other patterns can be generated the same way arbitrary waveforms are being generated. The video stroke generator requires that both channels output video data; the first channel generates the Y coordinates while the second generates the X coordinates.

Parameter ranges and their defaults are given where applicable. The communication session is terminated with the close function.

3156B Function Name	**Parameter(s)**	**Range**	**Default**
Video Stroke Generator Programming			
ri3156B_set(query)_video_stroke_point_freq	ViReal64 videoStrokePointFrequency	1 to 100e6	10e3
ri3156B_set(query)_video_offset_start	ViReal64 videoStrokeOffsetStart	-4.995 to 4.995	0
ri3156B_set(query)_video_offset_stop	ViReal64 videoStrokeOffsetStop	-4.995 to 4.995	0
ri3156B_set(query)_video_offset_range	ViReal64 videoStrokeOffsetStart	-4.995 to 4.995	0
	ViReal64 videoStrokeOffsetStop	-4.995 to 4.995	0
ri3156B_set(query)_video_offset_step	ViReal64 videoStrokeOffsetStep	±1m to ±9.99	1e-3
ri3156B_set(query)_video_stroke_circ_type	ViBoolean videoStrokeCircType	0, 1	0
ri3156B_set(query)_video_character	ViInt16 videoStrokeCharacter	0 to 9	0
ri3156B_set(query)_active_video_pat_number	ViInt16 ActivevideoStrokePatternNumber	1 to 16e3	1
ri3156B_load_video_str_pattern_data	ViInt16 videoStrokePatternNumber	1 to 16e3	1
	ViInt16 dataPointArray_ch1[]	array of 16 bit data	
	ViInt16 dataPointArray_ch2[]	array of 16 bit data	
	ViInt32 videoStrokePatternSize	1 to 512e3	1

ri3156B_apply_video_str_character	ViReal64 videoStrokePointFrequency	1 to 100e6	10e3
	ViReal64 videoStrokeOffsetStart	-4.995 to 4.995	0
	ViReal64 videoStrokeOffsetStop	-4.995 to 4.995	0
	ViReal64 videoStrokeOffsetStep	±1e-3 to ±9.99	1e-3
	ViBoolean videoStrokeCircType	0, 1	0
	ViInt16 videoStrokeCharacter	0 to 9	0
ri3156B_apply_video_stroke_pattern	ViReal64 videoStrokePointFrequency	1 to 100e6	10e3
	ViReal64 videoStrokeOffsetStart	-4.995 to 4.995	0
	ViReal64 videoStrokeOffsetStop	-4.995 to 4.995	0
	ViReal64 videoStrokeOffsetStep	±1e-3 to ±9.99	1e-3
	ViBoolean videoStrokeCircType	0, 1	0
	ViInt16 videoStrokePatternNumber	1 to 16e3	1

Ri3156b_set(query)_video_stroke_point_freq

Description

This programs the frequency of which stroke point will be sampled. This is parameter is very similar to the sample clock frequency where each clock samples a point in the video waveform file.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_video_stroke_point_freq (ViSession instrHandle, ViReal64 videoStrokePointFrequency)

ViStatus ri3156b_query_video_stroke_point_freq (ViSession instrHandle, ViReal64 * videoStrokePointFrequency)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
videoStrokePointFrequency	ViReal64	Programs the video waveform sample clock frequency. Valid range: 1 to 100e6 (pps) Default: 10e3

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_set(query)_video_offset_start

Description

This programs the video stroke offset start point. Use this function to set the initial position of the video stroke pattern/character

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_video_offset_start (ViSession instrHandle, ViReal64 videoStrokeOffsetStart)
ViStatus ri3156b_query_video_offset_start (ViSession instrHandle, ViReal64 * videoStrokeOffsetStart)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
videoStrokeOffsetStart	ViReal64	Selects the video stroke offset start level. Valid Range: -4.995 to +4.995(V into 50Ω) Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_set(query)_video_offset_stop

Description

This programs the video stroke offset stop point. Use this function to set the end position of the video stroke pattern/character

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_video_offset_stop (ViSession instrHandle, ViReal64 videoStrokeOffsetStop)
ViStatus ri3156b_query_video_offset_stop (ViSession instrHandle, ViReal64 * videoStrokeOffsetStop)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
videoStrokeOffsetStop	ViReal64	Selects the video stroke offset stop level. Valid Range: -4.995 to +4.995(V into 50Ω) Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_set(query)_video_offset_range

Description

This programs the video stroke offset range. Use this function to simultaneously set initial and final position of the video stroke pattern/character on the video monitor.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_video_offset_range (ViSession instrHandle, ViReal64 videoStrokeOffsetStart, ViReal64 videoStrokeOffsetStop)
ViStatus ri3156b_query_video_offset_range (ViSession instrHandle, ViReal64 * videoStrokeOffsetStart, ViReal64 * videoStrokeOffsetStop)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
videoStrokeOffsetStart	ViReal64	Selects the video stroke offset start level. Valid Range: -4.995 to +4.995(V into 50Ω) Default Value: 0

videoStrokeOffsetStop	ViReal64	Selects the video stroke offset stop level. Valid Range: -4.995 to +4.995(V into 50Ω) Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_set(query)_video_offset_step

Description

This programs the video stroke offset step increment. The video character will move about the screen at a rate set by the ri3156B_set_video_stroke_point_freq () and in step increments set by this function call.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_video_offset_step (ViSession instrHandle, ViReal64 videoStrokeOffsetStep)
ViStatus ri3156b_query_video_offset_step (ViSession instrHandle, ViReal64 * videoStrokeOffsetStep)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
videoStrokeOffsetStep	ViReal64	Selects the video stroke offset step increment. Valid Range: ±1e-3 to ±9.99 (V into 50Ω) Default Value: 1e-3

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_set(query)_video_stroke_circ_type

Description

This programs the circulation type for the video stroke generator. There are two options, both operate from an external trigger (or enable) signal: Single will move the pattern from start to stop offset and will leave the video character at the end of the offset range; Continuous will move the pattern from start to stop offset and will repeat the cycle as long as triggers are applied to the instrument. The circulation range is defined by the ri3156B_set_video_offset_range () function

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_video_stroke_circ_type (ViSession instrHandle, ViBoolean videoStrokeCircType)
ViStatus ri3156b_query_video_stroke_circ_type (ViSession instrHandle, ViBoolean * videoStrokeCircType)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
videoStrokeCircType	ViBoolean	Selects the circulation type for the video stroke mode. Valid Range: 0,1 0 – selects single 1 – selects continuous Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_set(query)_video_character

Description

This selects a video character from a built-in library of 10 standard characters. The characters are moved about the video screen in increments set by the ri3156B_set_video_offset_step () function through the range set by the ri3156B_set_video_offset_range () function. Both channels drive the XY coordinate simultaneously.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_video_character (ViSession instrHandle, ViInt16 videoStrokeCharacter)
ViStatus ri3156b_query_video_character (ViSession instrHandle, ViInt16 * videoStrokeCharacter)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
videoStrokeCharacter	ViInt16	Selects a video character to be output. Valid Range: 0 to 9 0 – selects cross locator character 1 – selects cross hair character 2 – selects positioned square character 3 – selects vertical marker line character 4 – selects horizontal marker line character 5 – selects right-hand arrow character 6 – selects left-hand arrow character 7 – selects diamond overlay character 8 – selects inverted triangle character 9 – selects upright triangle character Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

ri3156b_set(query)_active_video_pat_number

Description

This selects a video pattern from preloaded library of up to 16k characters. The character will be generated only if the 3156B was programmed to output video stroke patterns. The pattern is loaded to the instrument using the ri3156B_load_video_stroke_pattern_data function. The characters are moved about the video screen in increments set by the ri3156B_set_video_offset_step () function through the range set by the ri3156B_set_video_offset_range () function. Both channels drive the XY coordinate simultaneously.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_active_video_pat_number (ViSession instrHandle, ViInt16 videoStrokePatternNumber)
ViStatus ri3156b_query_active_video_pat_number (ViSession instrHandle, ViInt16 * videoStrokePatternNumber)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
videoStrokePatternNumber	ViInt16	Selects a pre-loaded video pattern number. Valid Range: 1 to 16374 Default Value: 1

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_video_str_pattern_data

Description

This command will download the data arrays to generate video character. Data is loaded to both channels simultaneously. Note that the pattern number is equivalent to the segment number that you download for arbitrary waveform segment and therefore, be careful, if you use the same numbers, video patterns may not share the same numbers as you use for arbitrary waveforms. The pattern number is selected using the ri3156B_set_active_video_pattern_number () function. The character will be generated only if the 3156B was programmed to output video stroke patterns.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_load_video_str_pattern_data (ViSession instrHandle, ViInt16 videoStrokePatternNumber, ViInt16 dataPointArrayCh1[],ViInt16 dataPointArrayCh2[],ViInt32 videoStrokePatternSize)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
videoStrokePatternNumber	ViInt16	Selects a pre-loaded video pattern number. Valid Range: 1 to 16374 Default Value: 1

dataPointArrayCh1	ViInt16[]	Select an array of data to be downloaded to a specific memory segment. Data is loaded to channel 1 patterns Data range: 0 to 0xFFFF (16-bit)
dataPointArrayCh2	ViInt16[]	Select an array of data to be downloaded to a specific memory segment. Data is loaded to channel 2 patterns Data range: 0 to 0xFFFF (16-bit)
digPatternStimListSize	ViInt32	Defines the number of points in the video character array. This is similar to defining the number of points for an arbitrary waveform segment. The size of the array must match the number of data points in the datPointArrays. Valid Range: 1 to 523264

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_video_str_character

Description

Programs all video pattern parameters simultaneously. It also sets video stroke as the active output function. Note that this function can be used for the built-in library of video characters only. If you design your own characters, you can use the Ri3156b_apply_video_stroke_pattern function instead.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_apply_video_str_character (ViSession instrHandle, ViReal64 videoStrokePointFrequency, ViReal64 videoStrokeOffsetStart, ViReal64 videoStrokeOffsetStop, ViReal64 videoStrokeOffsetStep, ViBoolean videoStrokeCircType, ViInt16 videoStrokeCharacter)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
videoStrokePointFrequency	ViReal64	Programs the video waveform sample clock frequency. Valid range: 1 to 100e6 (pps) Default: 10e3

videoStrokeOffsetStart	ViReal64	Selects the video stroke offset start level. Valid Range: -4.995 to +4.995(V into 50Ω) Default Value: 0
videoStrokeOffsetStop	ViReal64	Selects the video stroke offset stop level. Valid Range: -4.995 to +4.995(V into 50Ω) Default Value: 0
videoStrokeOffsetStep	ViReal64	Selects the video stroke offset step increment. Valid Range: ±1e-3 to ±9.99 (V into 50Ω) Default Value: 1e-3
videoStrokeCircType	ViBoolean	Selects the circulation type for the video stroke mode. Valid Range: 0,1 0 – selects single 1 – selects continuous Default Value: 0
videoStrokeCharacter	ViInt16	Selects a video character to be output. Valid Range: 0 to 9 0 – selects cross locator character 1 – selects cross hair character 2 – selects positioned square character 3 – selects vertical marker line character 4 – selects horizontal marker line character 5 – selects right-hand arrow character 6 – selects left-hand arrow character 7 – selects diamond overlay character 8 – selects inverted triangle character 9 – selects upright triangle character Default Value: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_apply_video_stroke_pattern

Description

Programs all preloaded video pattern parameters simultaneously. It also sets video stroke as the active output function. Note that this function can be used for an externally-built video characters only. If you want to use the standard characters, you can use the Ri3156b_apply_video_str_character function instead.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_apply_video_stroke_pattern (ViSession instrHandle, ViReal64 videoStrokePointFrequency, ViReal64 videoStrokeOffsetStart, ViReal64 videoStrokeOffsetStop, ViReal64 videoStrokeOffsetStep, ViBoolean videoStrokeCircType, ViInt16 videoStrokePatternNumber)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
videoStrokePointFrequency	ViReal64	Programs the video waveform sample clock frequency. Valid range: 1 to 100e6 (pps) Default: 10e3
videoStrokeOffsetStart	ViReal64	Selects the video stroke offset start level. Valid Range: -4.995 to +4.995(V into 50Ω) Default Value: 0
videoStrokeOffsetStop	ViReal64	Selects the video stroke offset stop level. Valid Range: -4.995 to +4.995(V into 50Ω) Default Value: 0
videoStrokeOffsetStep	ViReal64	Selects the video stroke offset step increment. Valid Range: ±1e-3 to ±9.99 (V into 50Ω) Default Value: 1e-3
videoStrokeCircType	ViBoolean	Selects the circulation type for the video stroke mode. Valid Range: 0,1 0 – selects single 1 – selects continuous Default Value: 0
videoStrokePatternNumber	ViInt16	Selects a pre-loaded video pattern number. Valid Range: 1 to 16374 Default Value: 1

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

The Trigger Functions Group

Use this group to synchronize device operations with external events. These functions control the run modes of the 3156B. The generator can be placed in Triggered, Gated or Burst mode, with or without trigger and re-trigger delays. Trigger source is selectable from a number of external sources or a software trigger.

Parameter ranges and their defaults are given where applicable. The communication session is terminated with the close function.

Trigger Programming			
3156B Function Name	**Parameter(s)**	**Range**	**Default**
Trigger Inputs and Outputs			
ri3156B_set(query)_trigger_source	ViInt16 triggerSource	0 to 9	0
ri3156B_set(query)_trigger_delay	ViReal64 triggerDelay	500e-9 to 21	500e-9
ri3156B_set(query)_trigger_delay_state	ViBoolean triggerDelayState	0, 1	0
ri3156B_set(query)_burst_mode_cycles	ViInt32 numberofCycles	1 to 1e6	1
ri3156B_set(query)_Mod_burst_mode_cycles	ViInt32 ModulatNumberofCycles	1 to 1e6	1
ri3156B_set(query)_trigger_slope	ViBoolean triggerSlope	0, 1	0
ri3156B_set(query)_trigger_level	ViReal64 triggerLevel	-5 to +5	1.6
ri3156B_set(query)_re_trigger_delay_state	ViBoolean retriggerDelayState	0, 1	0
ri3156B_set(query)_re_trigger_delay	ViReal64 retriggerDelay	500e-9 to 21	500 ns
Sync Outputs			
ri3156B_set(query)_sync_output_type	ViInt16 SYNCPulseType	0, 1	0
ri3156B_set(query)_sync_output_state	ViBoolean syncState	0, 1	0
ri3156B set(query)_TTLTRG_n_output_state	VIBoolean TTLTRG_out_state	0, 1	0
	VIInt 16 TTLTRG_n	0 to 7	0

Ri3156b_set(query)_trigger_source

Description

The 3156B accepts triggers (or enables) from one of four locations:

1. Front panel trigger input connector;
2. Backplane TTLTrg lines 0 through 7;
3. Back plan ECLTrg0, or
4. Software call

When a front panel source is not available, the operator has the option of using either the TTLTRG<n> or ECLTRG0. A software commands can be used when none of the external sources is available.

Note: TTLTrg0-7 lines are bi-directional. When you program a line as an output, it cannot be programmed simultaneously as an input. Also note, the ECLTrg0 line is an input source only.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_trigger_source (ViSession instrHandle, ViInt16 triggerSource)

ViStatus ri3156b_query_trigger_source (ViSession instrHandle, ViInt16 * triggerSource)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
triggerSource	ViInt16	Select the source to trigger the instrument: External selects the front panel TRIG IN connector as the active trigger source. One or more of TTLTRG 0 through 7 or ECLTRG0 selects the backplane lines as the active trigger source. To use software trigger or enable commands, the 3156B must be set to external trigger source. Valid range: 0 to 9 0 selects the front panel TRIG IN connector as the active trigger source. Also, use this option when external trigger source is not available and software trigger is expected 1 selects TTLTrg0 as the active trigger source 2 selects TTLTrg1 as the active trigger source 3 selects TTLTrg2 as the active trigger source 4 selects TTLTrg3 as the active trigger source 5 selects TTLTrg4 as the active trigger source 6 selects TTLTrg5 as the active trigger source 7 selects TTLTrg6 as the active trigger source 8 selects TTLTrg7 as the active trigger source 9 selects ECLTrg0 as the active trigger source Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_trigger_delay

Description

The trigger delay parameter defines the time that will elapse from a valid trigger signal to the initiation of the first output waveform. Trigger delay can be turned ON and OFF using the ri3156B_set_trigger_delay_state () function.

Note: System delay must always be considered when using an external trigger. System delay is measured from a valid trigger input to the transition of the first waveform point. It has a fixed period that adds to the programmed trigger delay value. System delay is 1 sample clock cycle +100ns.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_trigger_delay (ViSession instrHandle, ViReal64 triggerDelay)

ViStatus ri3156b_query_trigger_delay (ViSession instrHandle, ViReal64 * triggerDelay)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
triggerDelay	ViReal64	Selects trigger delay value. ri3156B_set_trigger_delay (1) function call is required for this value to affect the operation of the 3156B. ***NOTE***: trigger delay time is programmed in units of µs (microsecond). For example, to program 1.37ms delay time, enter the value as follows: 1370 Valid range: 0.5 to 21e6 (µs) Default: 0.5

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_trigger_delay_state

Description

This panel toggles trigger delay ON and OFF.

Note: System delay must always be considered when using an external trigger. System delay is measured from a valid trigger input to the transition of the first waveform point. It has a fixed period that adds to the programmed trigger delay value. System delay is 1 sample clock cycle +100ns.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_trigger_delay_state (ViSession instrHandle, ViBoolean triggerDelayState)

ViStatus ri3156b_query_trigger_delay_state (ViSession instrHandle, ViBoolean * triggerDelayState)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
triggerDelayState	ViBollean	Toggles the trigger delay On and Off Valid range: 0, 1 (Off, On) Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_burst_mode_cycles

Description

This function sets the number of cycles when the Burst Mode is on. Use the ri3156B_set_carrier_run_mode() function to select the Burst Mode.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_burst_mode_cycles (ViSession instrHandle, ViInt32 number_ofCycles)

ViStatus ri3156b_query_burst_mode_cycles (ViSession instrHandle, ViInt32 * number_ofCycles)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
number_ofCycles	ViInt32	Select the number of cycles to be output per burst. Valid range: 1 to 1000000 Default: 1

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_mod_burst_mode_cycles

Description

This function sets the number of modulation cycles when a modulation function is active and burst is on. Use the ri3156B_set_modulation_run_mode() function to select the Burst Mode.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_mod_burst_mode_cycles (ViSession instrHandle, ViInt32 number_ofCycles)

ViStatus ri3156b_query_mod_burst_mode_cycles (ViSession instrHandle, ViInt32 * number_ofCycles)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
number_ofCycles	ViInt32	Select the number of modulation cycles to be output per burst. Valid range: 1 to 1000000 Default: 1

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_trigger_slope

Description

The trigger slope function selects the sensitive edge of the trigger signal that is applied to the TRIG IN connector or to the backplane TTLTrg lines. The 3156B can be made sensitive to either the positive or negative transitions. Positive going transitions will trigger the 3156B when the POS option is selected. Negative transitions will trigger the 3156B when the NEG option is selected. In Gated mode, two transitions in the same direction are required to gate on and off the output.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_trigger_slope (ViSession instrHandle, ViBoolean triggerSlope)

ViStatus ri3156b_query_trigger_slope (ViSession instrHandle, ViBoolean * triggerSlope)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
triggerSlope	ViBoolean	Selects the trigger edge for the external trigger signal. Valid range: 0, 1 0 – selects positive transitions 1 – selects negative transitions Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_trigger_level

Description

The trigger level command sets the threshold level at the trigger input connector. Trigger levels are adjustable from -5V to 5V.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_trigger_level (ViSession instrHandle, ViReal64 triggerLevel)

ViStatus ri3156b_query_trigger_level (ViSession instrHandle, ViReal64 * triggerLevel)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
triggerLevel	ViBoolean	Programs the trigger level. Valid range: -5 to +5 (V) Default: 1.6

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_re_trigger_delay

Description

In trigger run mode, for standard, arbitrary and digital operating modes, this parameter specifies the amount of time that will elapse between the end of the delivery of the waveform cycle and the beginning of the next waveform cycle. In burst run mode, this parameter specifies the amount of time that will elapse between the end of the delivery of the burst cycles and the beginning of the next burst cycles. The re-trigger delay has no effect on other run modes. In trigger run mode, for Sweep, FM, FSK and Frequency Hops, this parameter specifies the amount of time that will elapse between the end of the delivery of a modulated waveform cycle and the beginning of the next modulated waveform cycle. In burst run mode, this parameter specifies the amount of time that will elapse between the end of the delivery of the burst of modulated cycles and the beginning of the next burst of modulation cycles. The re-trigger delay has no effect on other run modes.

Re-trigger run mode is initiated from one of the following sources: External (front panel TRIG IN connector), TTLTRG(n) or ECLTRG0 (backplane trigger lines), or software command (Immediate trigger).

Re-trigger delay can be turned ON and OFF using the ri3156B_set_re_trigger_delay_state () function.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_re_trigger_delay (ViSession instrHandle, ViReal64 reTriggerDelay)

ViStatus ri3156b_query_re_trigger_delay (ViSession instrHandle, ViReal64 * reTriggerDelay)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
reTriggerDelay	ViReal64	Selects re-trigger delay value. ri3156B_set_re_trigger_delay (1) function call is required for this value to affect the operation of the 3156B. ***NOTE***: trigger delay time is programmed in units of μs (microsecond). For example, to program 1.37ms delay time, enter the value as follows: 1370 Valid range: 0.5 to 21e6 (μs) Default: 0.5

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_re-trigger_delay_state

Description

This toggles re-trigger delay ON and OFF. Re-trigger delay value can be set using the ri3156b-set_re_trigger_delay () function. Re-trigger run mode is initiated from one of the following sources: External (front panel TRIG IN connector), TTLTRG(n) or ECLTRG0 (backplane trigger lines), or software command (Immediate trigger).

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_re_trigger_delay_state (ViSession instrHandle, ViBoolean reTriggerDelayState)

ViStatus ri3156b_query_re_trigger_delay_state (ViSession instrHandle, ViBoolean * reTriggerDelayState)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
reTriggerDelayState	ViBoolean	Toggles the re-trigger delay On and Off Valid range: 0, 1 (Off, On) Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_sync_output_type

Description

This selects between two sync options: Pulse and Zero crossing. The pulse option generates a single pulse at the start of the waveform and the zero crossing option generates a pulse every time the output waveform transitions through the 0V level.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_sync_output_type (ViSession instrHandle, ViInt16 SYNCPulseType)

ViStatus ri3156b_query_sync_output_type (ViSession instrHandle, ViInt16 * SYNCPulseType)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.

SYNCPulseType	ViInt16	Selects the sync pulse type Valid range: 0, 1 0 – selects pulse 1 – selects zero crossing Default: 0

Return Values
Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_sync_output_state

Description

This toggles the SYNC signal, at the SINC OUT connector only, ON and OFF. Sync type can be set using the ri3156b-set_sync_type () function. This command does not affect the TTLTRG(n) state.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_set_sync_output_state (ViSession instrHandle, ViBoolean syncState)

ViStatus ri3156b_query_sync_output_state (ViSession instrHandle, ViBoolean * syncState)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
syncState	ViBoolean	Toggles the SYNC output On and Off Valid range: 0, 1 (Off, On) Default: 0

Return Values
Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_set(query)_TTLTRG_n_output_state

Description

This programs one or more of the eight backplane TTLTrg lines as an output. The signal that is placed on this output is the same signal as generated at the front-panel SYNC output connector. The synchronization signal is selected using the ri3156B_set_sync_output_type () function; The ri3156B_set_sync_output_state () has no effect on the backplane trigger lines.

Note: TTLTrg0-7 lines are bi-directional. When you program a line as an output, it cannot be programmed simultaneously as an input. Also note, the ECLTrg0 line is an input source only.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_set_TTLTRG_n_output_state (ViSession instrHandle, ViBoolean TTLTRG_output_state, ViInt16 TTLTRG_n)

ViStatus ri3156b_query_TTLTRG_n_output_state (ViSession instrHandle, ViBoolean * TTLTRG_output_state, ViInt16 TTLTRG_n)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
TTLTRG_output_state	ViBoolean	Toggles a specific TTLTrg line as an output on and off. Valid range: 0, 1 0 Turns a specific TTLTrg line output OFF 1 Turns a specific TTLTrg line output ON Default: 0
TTLTRG_n	ViInt16	Selects a specific TTLTrg line as an output. Valid range: 0, 7 (TTLTrg0-7) Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

The WaveCAD Support Functions Group

Use the functions in this group to point to a path where WaveCAD files were already checked and stored. There are similar functions that were already described in the above that point to arrays of data but in this case, you have to prepare the data on your own while the function call merely points to the location of your data array in your program.

WaveCAD is a high level utility that lets you generate, edit and download waveforms without accessing low level programming. Using WaveCAD, you can prepare waveforms and data arrays on the screen and save as files for either later use or for embedding in function call that are summarized in this group.

The files are saved in structures that are compatible with the function calls below. Each file has different purpose and the way to differentiate between the files is their extension. The following file extensions are being used:

***.wav** – used for storing coordinates of arbitrary waveforms. The same extension is used for files that contain video characters

***.wfm** – used for storing coordinates of arbitrary waveforms that will be used for frequency modulation

***.hop** – stores hop table data – variable dwell time

***.hpf** – stores hop table data – fixed dwell time

***.fsk** – stores the strings to generate FSK

***.pat** – stores data for digital patterns - freerun

***sti** – stores data for digital patterns – stimulus

Samples of the files structure is given in the description of each function call. Parameter ranges and their defaults are given where applicable. The communication session is terminated with the close function.

WaveCAD Support Programming			
3156B Function Name	**Parameter(s)**	**Range**	**Default**
ri3156B_load_wavecad_wave_file	ViInt16 segmentNumber	1 to 4096	
	ViString WaveCadWaveformFileName	File name path	
	ViInt32 segmentSize	1 to 512k	
	ViBoolean fileResolution	0, 1	
ri3156B_load_wavecad_FM_wave_file	ViString WaveCadFM_DataFileName	File name path	
	ViInt32 FMnumberOfPoints	10 to 32768	

WaveCAD Support Programming (continued)			
3156B Function Name	**Parameter(s)**	**Range**	**Default**
ri3156B_load_wavecad_HOP_freq_list_file	ViString WaveCadHopDataFileName	Name path	
	ViInt16 hopFreqListSize	1 to 4096	
ri3156B_load_wavecad_FSK_data_file	ViString WaveCadFskDataFileName	File name path	
	ViInt16 FSKwordLength	8 to 4096	
ri3156B_load_wavecad_dig_patt_stim_list_file	ViString WaveCadDigitalPatternDataFileName	File name path	
	ViInt32 digPatternStimListSize	1 to 512k	
ri3156B_load_wavecad_video_data_file	ViInt16 videoStrokePatternNumber	1 to 16e3	
	ViString VideoDataCh1FileName	File name path	
	ViString VideoDataCh2FileName	File name path	
	ViInt32 videoStrokePatternSize	1 to 512e3	

Ri3156b_load_wavecad_wave_file

Description

This function is similar to the Ri3156b_load_arb_data except it points to a file name that was already prepared and stored on your computer. The complete path to the file location is required. You can use this function to load files in either 12-bit or 16-bit format, depending on how you define the format of your data with the fileResolution variable.

Channel Dependency: Independent

C Syntax

ViStatus ri3156b_load_wavecad_wave_file (ViSession instrHandle, ViInt16 segmentNumber, ViString waveCADWaveformFileName, ViInt32 segmentSize, ViBoolean fileResolution)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
segmentNumber	ViInt16	Select the segment number to become active Valid range: 1 to 16k Default: 1
waveCADWaveformFileName	ViString	Points to a file name already stored in your computer's memory. Full path is required. File extension is *.wav. The file structure should have waveform coordinates in the range of: 0 to 0xFFF (12-bit) or 0 to 0xFFFF (16-bit). The number of points in this file must match the number of points that are defined for this segment with the segmentSize variable.

segmentSize	ViInt32	Select the size of the segment to be loaded. Size must match the number of data points in the data file. Valid range: 1 to 523264 with SCLK setting of 100MS/s; 2 to 1046528 with SCLK setting of 200MS/s
fileResolution	ViBoolean	Defines the data file format: for either 12-bit or 16-bit Valid range: 0,1 0 – selects 16-bit; 1 – selects 12-bit Default: 0

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_wavecad_FM_file

Description

This function is similar to the Ri3156b_load_arb_FM_mod_data except it points to a file name that was already prepared and stored on your computer. The complete path to the file location is required. This command will download FM modulating waveform data to the arbitrary FM memory. Below you can see how such data files are constructed. Downloading data to the arbitrary FM waveform memory is very different than loading arbitrary waveform data. Waveform data programs amplitude domain therefore, every point programs an amplitude level. On the other hand, FM modulating waveform data programs frequency domain therefore, every point sets different sample clock frequency.

Arbitrary FM waveform sample file. This file has 80 frequency points of that will create the following output modulation:

1. Stable 1MHz signal for 11 arbitrary FM SCLK intervals,
2. Sine shaped waveform, minimum frequency is 900047Hz and maximum frequency is 1.09995e+006Hz
3. Stable 1MHz signal for 19 arbitrary FM SCLK intervals.

The frequency of the modulating signal is computed from the following equation:

Modulation frequency = FM modulation SCLK / Number of Points in the arbitrary modulating waveform

The modulating waveform sample clock is programmed using the ri3156B_set_arb_FM_mod_sclk

Channel Dependency: Independent

Sample Arbitrary Modulating Waveform Data File

```
1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1.01229e+006
1.02439e+006 1.03612e+006 1.04731e+006 1.05778e+006 1.06737e+006 1.07594e+006 1.08336e+006
1.08952e+006 1.09432e+006 1.09768e+006 1.09957e+006 1.09995e+006 1.09882e+006 1.09618e+006
1.09209e+006 1.0866e+006 1.0798e+006 1.07179e+006 1.06269e+006 1.05264e+006 1.0418e+006
1.03032e+006 1.01837e+006 1.00616e+006 993844 981625 969685 958204 947357 937308 928209
920198 913397 907909 903817 901183 900047 900427 902315 905685 910484 916640 924060 932630
942223 952691 963876 975609 987711 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006
1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006 1e+006
```

C Syntax

ViStatus ri3156b_load_wavecad_FM_wave_file (ViSession instrHandle, ViString waveCADFMWaveFileName, ViInt32 FMNumber_ofPoints)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
waveCADFMWaveFileName	ViString	Points to a file name already stored in your computer's memory. Full path is required. File extension is *.wfm. The file structure should have waveform coordinates in the range of 0.01 to 25e6 (Hz)
FMNumber_ofPoints	ViInt32	Defines the number of points for the arbitrary modulating signal. This number must match the number of frequencies in the data array. Valid Range: 10 to 32768 (waveform points)

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_var_HOP_freq_list

Description

This function is similar to the ri3156b_load_var_hop_freq_list and ri3156b_load_var_hop_freq_list functions except it points to a file name that was already prepared and stored on your computer. The complete path to the file location is required. This command will download the data file that will cause the instrument to hop through the frequency list. The same function is used for both the variable dwell time and fixed dwell time hops. The

Below you can see how a hop file is constructed. The file sample below shows a list of 10 frequencies and their associated dwell times. The 3156B will hop through this list, outputting the next frequency each time it hops. Note that the carrier waveform is always sinewave and that the last cycle is always completed even if the dwell time is shorter than the period of the waveform. For example, if you program dwell time of 1ms and the frequency step has frequency of 1Hz (1s period), the frequency step will last 1 second although the dwell time is 1ms. For fixed dwell time, remove from the table the dwell time values and leave just the frequency values.

Channel Dependency: Independent

Sample Data File - Frequency Hops with Variable Dwell Time

```
1e+6 100 2e+6 2000 3e+3 3e4 4e+6 40 5e+5 5e3 6e+2 6000 7e+1 0.7 8e+6 8e2 9e+3 90 10e+5 1000
```

Sample Data File - Frequency Hops with Fixed Dwell Time

1e+6 2e+6 3e+3 4e+6 5e+5 6e+2 7e+1 8e+6 9e+3 10e+5

C Syntax

ViStatus ri3156b_load_wavecad_HOP_freq_list_file (ViSession instrHandle, ViString waveCADHOPFreqListFileName, ViInt16 hopFreqListSize)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
waveCADHOPFreqListFileName	ViString	Points to a file name already stored in your computer's memory. Full path is required. File extension is *.hop for variable dwell time files and *.hpf for fixed dwell time files.
hopFreqListSize	ViInt16	Defines the number of frequency steps are included in the frequency hops table. This number must match the number of frequencies in the data array and in the dwell times array. Valid Range: 1 to 4096
hopDwellTimeList	ViReal64[]	Points to an array of dwell time values. Each dwell time is associated with a different frequency hop and therefore, the array size must match the number of frequency hops. ***NOTE***: Dwell time is programmed in units of μs (microsecond). For example, to program 1.37ms dwell time, enter the value as follows: 1370 Valid range: 0.5 to 21e6 (μs) Default: 0.5

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_FSK_data_file

Description

This function is similar to the ri3156b_load_FSK_data function except it points to a file name that was already prepared and stored on your computer. The complete path to the file location is required. This function loads the data stream that will cause the 3156B to hop from carrier to shifted frequency and visa versa. Data format is a string of "0" and "1" which define when the output generates carrier frequency and when it shifts frequency to the FSK value. "0" defines carrier frequency,"1" defines shifted frequency. Note that if you intend to program marker position, you must do it before you load the FSK data list.

Below you can see how an FSK data table is constructed. The sample below shows a list of 10 shifts. The 3156B will step through this list, outputting either carrier or shifted frequencies, depending on the data list: Zero will generate carrier frequency and One will generate shifted frequency. Note that the waveform is always sinewave and that the last cycle is always completed.

Channel Dependency: Independent

Sample FSK Data File

0 1 1 1 0 1 0 0 0 1

C Syntax

ViStatus ri3156b_load_FSK_data_file (ViSession instrHandle, ViString FSKFileName, ViInt16 FSK_word_length)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
FSKFileName	ViString	Points to a file name already stored in your computer's memory. Full path is required. File extension is *.fsk Valid Range: 0 or 1
FSK_Word_Length	ViInt16	Defines the length of the FSK data array. This number must match the number of shifts in the data array Valid Range: 8 to 4096 Default: 8

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_dig_pattern_stim_list

Description

This function is similar to the ri3156b_load_FSK_data function except it points to a file name that was already prepared and stored on your computer. The complete path to the file location is required. This function will download the data file for the Freerun digital patterns. For freerun digital patterns, one can program variable hold count for each pattern. The hold count defines the dwell time per pattern.

Below you can see how a digital pattern sample files are constructed. The sample below shows a list of 10 patterns and their associated hold counts. The 3156B will step through this list, outputting the next pattern each time it hop to the next step at a rate programmed by the ri3156B_set_dig_patter_rate() function call.

Channel Dependency: Independent

Sample File Digital Patters Arrays – Freerun Mode

0x001 100 0x100 200 0x203 333 0x400 44 0x805 50 0xD00 600 0xA07 707 0x118 8 0x0FD 90 0x010 1000

Sample File Digital Patters Arrays – Stimulus Mode

0x001 0x100 0x203 0x400 0x805 0xD00 0xA07 0x118 0x0FD 0x010

C Syntax

ViStatus ri3156b_load_dig_pattern_stim_list (ViSession instrHandle, ViString waveCADdigitalPatternDataFileName,ViInt32 digPatternStimListSize

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
waveCADdigitalPatternDataFileName	ViString	Points to a file name already stored in your computer's memory. Full path is required. File extension is *.pat for freerun mode and *.sti for stimulus mode
digPatternStimListSize	ViInt32	Defines the number of digital pattern steps are included in the table. This number must match the number of patterns in the data and in the hold count arrays. Valid Range: 1 to 523264

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_load_wavecad_video_data_file

Description

This function is similar to the ri3156b_load_video_str_pattern_data function except it points to a file name that was already prepared and stored on your computer. The complete path to the file location is required. This function will download the data arrays to generate video character. Data is loaded to both channels simultaneously. Note that the pattern number is equivalent to the segment number that you download for arbitrary waveform segment and therefore, be careful, if you use the same numbers, video patterns may not share the same numbers as you use for arbitrary waveforms. The pattern number is selected using the videoStrokePatterNumber variable. The character will be generated only if the 3156B was programmed to output video stroke patterns.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_load_wavecad_video_data_file (ViSession instrHandle, ViInt16 videoStrokePatternNumber, ViString waveCADPatternDataCh1File, ViString waveCADPatternDataCh2File, ViInt32 videoStrokePatternSize)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
videoStrokePatternNumber	ViInt16	Selects a pre-loaded video pattern number. Valid Range: 1 to 16374 Default Value: 1
waveCADPatternDataCh1File	ViIString	Points to a file name already stored in your computer's memory. Full path is required. File extension is *.wav
waveCADPatternDataCh2File	ViIString	Points to a file name already stored in your computer's memory. Full path is required. File extension is *.wav
videoStrokePatternSize	ViInt32	Defines the number of points in the video character file. This is similar to defining the number of points for an arbitrary waveform segment. The number of points must match the number of data points in the data files. Valid Range: 1 to 523264

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

The Utility Functions Group

Functions in this group do not directly control instrument operation but come to assist in extracting important information from the 3156B such as when was the last calibration date, what option was installed in the factory and more.

Parameter ranges and their defaults are given where applicable. The communication session is terminated with the close function.

Utility Functions			
3156B Function Name	**Parameter(s)**	**Range**	**Default**
ri3156B_clear	None		
ri3156B_get_option	ViInt16 *optionInstalled	0, 1	0
ri3156B_revision_query	ViChar driverRevision[]		
	ViChar firmwareRevision[]		
ri3156B_error_query	ViInt32 *error		
	ViChar errorMessage		
ri3156B_error_message	ViStatus errorReturnValue		
	ViChar errorMessage[]		
Ri3156B_read_serial_number	ViChar* SerialNumber		
Ri3156B_read_last_cal_date	ViChar* LastCalDate		
Ri3156B_query_addr_space	ViInt16 *addrSpace	0, 1 (A24,A32)	0

Ri3156b_clear

Description

Performs a VXI word serial clear command. This results in clearing the input and output buffers of the 3156B. This can be useful for re-establishing communication with the 3156B and for ensuring no partial commands lie in the instrument's input buffer.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_clear (ViSession instrHandle)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.

Return Values
Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_get_option

Description
This interrogates the 3156B for the installed options. There is only one option available for the 3156B – TCXO time base. The TCXO reference option increases the accuracy of the generator output from 100ppm to 1ppm.

Channel Dependency: Common

C Syntax
ViStatus ri3156b_get_option (ViSession instrHandle, ViPInt16 optionInstalled)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
optioninstalled	ViInt16	This returns a number designating if there is an option installed in the instrument. Valid Range: 0,1 (no option, option installed)

Return Values
Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_revision_query

Description
This interrogates the 3156B's driver and firmware version. The latest revision levels are published on EADS North America Defense Test and Services, Inc. web site. Compare these numbers with the values you receive from this function call.

Channel Dependency: Common

C Syntax
ViStatus ri3156b_revision_query (ViSession instrHandle, ViChar * driverRevision,ViChar * firmwareRevision)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
driverRevision	ViChar[]	This holds a string upon return from the function, designating the revision level of the driver.
firmwareRevision	ViChar[]	This holds a string upon return from the function, designating the revision level of the firmware.

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_error_query

Description

This interrogates the 3156B for programming error. Programming errors could occur from setting conflicts, programming parameters out of their legal ranges others. The function returns two parameters: Error number and Error message

Channel Dependency: Common

C Syntax

```
ViStatus ri3156b_error_query (ViSession instrHandle, ViPInt32 * error, ViChar * errorMessage)
```

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
error	ViInt32	Any number returned from this function other than 0 designates an error. “0” designates no error.
errorMessage	ViChar[]	This holds a string upon return from the function which is associated with the error number.

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_error_message

Description

This Converts a numeric error code returned by one of the functions of this driver into a descriptive error message string.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_error_message (ViSession instrHandle, ViStatus * errorReturnValue, ViChar * errorMessage)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
errorReturnValue	ViStatus	Accepts the error codes which were returned by one of the functions in this instrument driver.
errorMessage	ViChar[]	This holds a string upon return from the function which is associated with the error number.

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_read_serial_number

Description

This reads the serial number of the instrument as was assigned by the factory. The returned value must match the number printed on the serial number label on the side panel of the 3156B.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_read_serial_number (ViSession instrHandle, ViChar * serialNumber)

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the

		model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
serialNumber	ViChar[]	This holds the serial number of the instrument upon return from the function.

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_read_last_cal_date

Description

This reads the date of which the 3156B was calibrated the last time. Normal calibration cycle is 3 year however, if the unit has been repaired or checked to be out of its specified ranges, then the instrument will get calibrated and this value updated accordingly.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_read_last_cal_date (ViSession instrHandle, ViChar * lastCalibrationDate

Parameters

Name	**Variable Type**	**Description**
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
lastCalibrationDate	ViChar[]	This holds the last calibration date upon return from the function. The format is: dd-mm-yyyy.

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Ri3156b_query_addr_space

Description

This reads the address space the 3156B was programmed to use. Two options are used A24 or A32 address space. Configuration is made in the factory. The recommended configuration is A32.

Channel Dependency: Common

C Syntax

ViStatus ri3156b_query_addr_space (ViSession instrHandle, ViPInt16 * addrSpace)

Parameters

Name	Variable Type	Description
instrHandle	ViSession	The Instrument Handle is used to identify the unique session or communication channel between the driver and the instrument. If more than one instrument of the model type is used, this will be used to differentiate between them. Note that the initialize function may be used to associate multiple instrument handles with a single instrument.
addrSpace	ViInt16	This holds the address space upon return from the function. Valid Range: 0,1 (A24, A32)

Return Values

Displays the return status of the function call. If the function was successful, it will return a status of VI_SUCCESS, otherwise it will return an error code. Passing the error code into the function "ri3156B_error_message()" will return a string describing the error.

Chapter 6
MAINTENANCE AND PERFORMANCE CHECKS

What's in This Chapter

This chapter provides maintenance and service information, performance tests, and the procedures necessary to adjust and troubleshoot the 3156B Waveform Synthesizer.

WARNING:

The procedures described in this section are for use only by qualified service personnel. Many of the steps covered in this section may expose the individual to potentially lethal voltages that could result in personal injury or death if normal safety precautions are not observed.

CAUTION:

ALWAYS PERFORM DISASSEMBLY, REPAIR AND CLEANING AT A STATIC SAFE WORKSTATION.

Disassembly Instructions

If it is necessary to troubleshoot the instrument or replace a component, use the following procedure to remove the side panels:

1. Using a Phillips head screwdriver, remove the screws on each side of the instrument that secures the side panels.

2. Carefully lift the cover off the instrument. Use the same procedure to remove the other side panel. After removing the side panels from the instrument, access the component side for calibration and checks, and the solder side when replacing components.

3. When replacing the side panels, reverse the above procedure.

Special Handling of Static Sensitive Devices

CMOS devices are designed to operate at very high impedance levels for low power consumption. As a result, any normal static charge that builds up on your person or clothing may be sufficient to destroy these devices if they are not handled properly. When handling such devices, use the precautions described below to avoid damaging them:

1. CMOS IC's should be transported and handled only in containers specially designed to prevent static build-up. Typically, these parts are received in static-protected containers of plastic or foam. Keep these devices in their original containers until ready for installation.

2. Ground yourself with a suitable wrist strap. Remove the devices from the protective containers only at a properly grounded workstation.

3. Remove a device by grasping the body; do not touch the pins.

4. Any printed circuit board into which the device is to be inserted must also be grounded to the bench or table.

5. Use only anti-static type solder suckers.

6. Use only grounded soldering irons.

Once the device is installed on the PC board, the device is adequately protected and normal handling may resume.

Cleaning

The 3156B should be cleaned as often as operating conditions require. To clean the instrument, use the following procedure:

1. Thoroughly clean the inside and outside of the instrument.

2. When cleaning inaccessible areas, remove dust with low-pressure compressed air or a vacuum cleaner.

3. Use alcohol applied with a cleaning brush to remove accumulation of dirt or grease from connector contacts and component terminals.

4. Clean the exterior of the instrument and the front panel with a mild detergent mixed with water, applying the solution with a soft, lint-free cloth.

Repair and Replacement

Repair and replacement of electrical and mechanical parts must be accomplished with great care. Printed circuit boards can become warped, cracked or burnt from excessive heat or mechanical stress. The following repair techniques are suggested to avoid inadvertent destruction or degradation of parts and assemblies:

1. Use a 60/40 solder and temperature-controlled 35 - 40 watt pencil-type soldering iron on the circuit board. The tip of the iron should be clean and properly tinned for best heat transfer to the solder joint. A higher wattage soldering iron may separate the circuit from the base material.

2. Keep the soldering iron in contact with the PC board for a minimum time to avoid damage to the components or printed conductors.

3. To desolder components, use a commercial "solder sucker" or a solder-removing SOLDER - WICK, size 3.

4. Always replace a component with an exact duplicate as specified in the parts list.

Performance Checks

The following performance checks verify proper operation of the instrument and should normally be used:

1. As a part of the incoming inspection of the instrument specifications;

2. As part of the troubleshooting procedure;

3. After any repair or adjustment before returning the instrument to regular service.

Environmental Conditions

Tests should be performed under conditions of an ambient temperature of $0^{o}C$ to $37.7^{o}C$ and at a relative humidity of less than 80%. If the instrument has been subjected to conditions outside these ranges, allow at least one additional hour for the instrument to stabilize before beginning the adjustment procedure.

Warm-up Period

Most equipment is subject to a small amount of drift when it is first turned on. To ensure accuracy, turn on the power to the 3156B and allow it to warm-up for at least 30 minutes before beginning the performance test procedure.

Initial Instrument Setting

To avoid confusion as to which initial setting is to be used for each test, it is required that the instrument be reset to factory default values prior to each test. To reset the 3156B to factory defaults, use the WaveCAD Utility panel to reset the instrument.

Recommended Test Equipment

Recommended test equipment for troubleshooting, calibration and performance checking is listed below. Test instruments other than those listed may be used only if their specifications equal or exceed the required characteristics.

Equipment	Model No.	Manufacturer
Oscilloscope (with jitter package)	LT342	LeCroy
Distortion Analyzer	6900B	Krohn Hite
Digital Multimeter	2000	Keithley
Freq. Counter	2201 w/ OPT.42	EADS North America Defense Test and Services, Inc.
Spectrum Analyzer	E4411	HP
Function Generator (with manual trigger)	33120A or Similar	Agilent

Performance Check Procedures

Use the following procedures to check the 3156B against the specifications. A complete set of specifications is listed in Appendix A. The following paragraphs show how to set up the instrument for the test, what the specifications for the tested function are, and what acceptable limits for the test are. If the instrument fails to perform within the specified limits, the instrument must be calibrated or tested to find the source of the problem.

Frequency Accuracy

Equipment: Counter

Preparation:

1. Configure the counter as follows:
   Termination: 50Ω, DC coupled
2. Connect the 3156B Channel 1 output to the counter input – channel A
3. Configure the 3156B, channel 1 as follows:
   Reset
   Frequency: As specified in **Table 3-1**
   Waveform: Squarewave
   Amplitude: 2V
   Output: On
   Enable: On

Test Procedure

1. Perform frequency Accuracy tests using **Table 5-1**

Note. If the 3156B under test is equipped with a TCXO option, replace the ±100ppm Error Limits in **Table 5-1** with ±1ppm.

Table 6-1, Frequency Accuracy

3156B Setting	Error Limits	Counter Reading	Pass	Fail
10.0000000 Hz	±100ppm			
100.000000 Hz	±100ppm			
1.00000000 KHz	±100ppm			
10.0000000 KHz	±100ppm			
100.000000 KHz	±100ppm			
1.00000000 MHz	±100ppm			
10.0000000 MHz	±100ppm			
25.0000000 MHz	±100ppm			

Amplitude Accuracy

DAC Waveforms

Equipment: DMM

Preparation:

1. Configure the DMM as follows:
   Termination: 50Ω feedthrough at the DMM input
   Function: ACV
2. Connect 3156B Channel 1/2 output to the DMM input
3. Configure the 3156B channels 1/2 as follows:
   Reset
   Frequency: 1KHz
   Amplitude: As specified in **Table 5-2**
   Output: On
   Enable: On

Test Procedure

1. Perform amplitude Accuracy tests on both channels using **Table 3-2**

Table 6-2, Amplitude Accuracy, DAC Waveforms

3156B Amplitude		DMM Reading			
Setting	Error Limits	CH 1	CH 2	Pass	Fail
10.00 V	3.534 V, ± 60.0 mV				
5.000 V	1.767 V, ± 42 mV				
1.000 V	353.4 mV, ± 35 mV				
100.0 mV	35.34 mV, ± 6 mV				
10.00 mV	3.534 mV, ± 2.1 mV				

Modulation Waveforms

Equipment: DMM

Preparation:

1. Configure the DMM as follows:
   Termination: 50Ω feedthrough at the DMM input
   Function: ACV
2. Connect 3156B Channel 1/2 output to the DMM input
3. Configure the 3156B channels 1/2 as follows:
   Reset
   Waveform Mode: MOD
   Modulation Mode: OFF
   CW Frequency:1kHz
   Amplitude : As specified in **Table 5-3**
   Output: On

Test Procedure

1. Perform amplitude Accuracy tests on both channels using **Table 5-3**

Table 6-3, Amplitude Accuracy, Modulation Waveforms

3156B Amplitude		**DMM Reading**			
Setting	**Error Limits**	**CH 1**	**CH 2**	**Pass**	**Fail**
5.000 V	1.767 V, ± 42 mV				

Offset Accuracy

DAC Waveforms

Equipment: DMM

Preparation:

1. Configure the DMM as follows:
   Termination: 50Ω feedthrough at the DMM input
   Function: DCV
2. Connect 3156B Channel 1/2 output to the DMM input
3. Configure the 3156B channels 1/2 as follows:
   Reset
   Frequency: 1MHz
   Amplitude: 10mV
   Offset: As specified in **Table 3-3**
   Output: On
   Enable: On

Test Procedure

1. Perform Offset Accuracy tests on both channels using **Table 5-4**

Table 6-4, Offset Accuracy, DAC Waveforms

3156B Offset		DMM Reading			
Setting	**Error Limits**	**CH 1**	**CH 2**	**Pass**	**Fail**
+4.000V	4.000V ±45 mV				
0.000V	0V ±5 mV				
-4.000V	-4.000V ±45 mV				

2. Modify 3156B Amplitude setting to 5V and offset setting to 0V
3. Continue the Offset tests using **Table 5-5**

Table 6-5, Offset Accuracy, DAC Waveforms (continued)

3156B Offset		DMM Reading			
Setting	**Error Limits**	**CH 1**	**CH 2**	**Pass**	**Fail**
0.000 V	0 ± 55 mV				

Modulation Waveforms

Equipment: DMM

Preparation:

1. Configure the DMM as follows:
   Termination: 50Ω feedthrough at the DMM input
   Function: DCV
2. Connect 3156B Channel 1/2 output to the DMM input
3. Configure the 3156B channels 1/2 as follows:
   Reset
   Waveform Mode: MOD
   Modulation Mode: OFF
   CW Frequency: 1MHz
   Amplitude: 5V
   Output: On

Test Procedure

1. Perform offset Accuracy tests on both channels using **Table 5-6**

Table 6-6, Offset Accuracy, Modulation Waveforms

		DMM Reading			
3156B Setting	**Error Limits**	**CH 1**	**CH 2**	**Pass**	**Fail**
0.000 V	0V ± 55 mV				

Squarewave Characteristics

Equipment: Oscilloscope

Preparation:

1. Configure the Oscilloscope as follows:

Termination:	50Ω feedthrough at the oscilloscope input
Setup:	As required for the test

2. Connect 3156B Channel 1/2 output to the oscilloscope input
3. Configure the 3156B channels 1/2 as follows:

Reset

Frequency:	1MHz
Waveform:	Squarewave
Amplitude:	5V
Output:	On
Enable:	On

Test Procedure

1. Perform Squarewave Characteristics tests on both channels using **Table 5-7**

Table 6-7, Squarewave Characteristics Tests

Parameter		**Oscilloscope Reading**			
Tested	**Error Limits**	**CH 1**	**CH 2**	**Pass**	**Fail**
Rise/Fall Time	**<8ns**				
Ringing	**<8% + 10mV**				
Over/undershoot	**<8% + 10mV**				

Sine Wave Characteristics

Distortion, DAC Waveforms

Equipment: Distortion Analyzer, WaveCAD

Preparation:

1. Connect 3156B Channel 1/2 output to the distortion analyzer input. Use 50Ω feedthrough termination at the distortion analyzer input
2. Configure the 3156B channels 1/2 as follows:

Reset

SCLK: As required by the test

Waveform:	Arbitrary
Amplitude:	5V

Output:	On
Enable:	On

3. Using WaveCAD prepare and download the following waveform:

Wavelength:	As required by the test
Waveform:	Sinewave

Test Procedure

1. Perform Sinewave distortion tests on both channels using **Table 5-8**

Table 6-8, Sinewave Distortion, DAC Waveforms Tests

3156B SCLK	**Sinewave**	**3156B**	**Reading**	**Distortion Reading**			
Settings	**Points**	**Frequency**	**Limits**	**CH 1**	**CH 2**	**Pass**	**Fail**
40 KS/s	4000	10.00 Hz	< 0.1%				
400 KS/s	4000	100.0 Hz	< 0.1%				
4 MS/s	4000	1.000 KHz	< 0.1%				
40 Ms/s	4000	10.00 KHz	< 0.1%				
100 Ms/s	2000	50.00 KHz	< 0.1%				
200 Ms/s	2000	100.00 KHz	< 0.1%				

Sinewave Spectral Purity , DAC Waveforms

Equipment: Spectrum Analyzer

Preparation:

1. Connect 3156B Channel 1/2 output to the spectrum analyzer input. Use 50Ω and 20dB feedthrough termination at the spectrum analyzer input
2. Configure the 3156B channels 1/2 as follows:

Reset	
Frequency:	As required by the test
Waveform:	Sinewave
Amplitude:	5V
Output:	On
Enable:	On

Test Procedure

1. Perform sinewave spectral purity, DAC waveforms tests using **Table 5-9**

Table 6-9, Sinewave Spectral Purity, DAC Waveforms Test

3156B Freq	**Reading**	**Spectrum Analyzer, Settings & Results**					
Settings	**Limits**	**Start**	**Stop**	**CH 1**	**CH 2**	**Pass**	**Fail**
1 MHz	>55 dBc	100K	10M				
5 MHz	>40 dBc	1M	20M				
10 MHz	>35 dBc	1M	100M				
25 MHz	>30 dBc	1M	150M				

Sinewave Spectral Purity, Modulation Waveforms

Equipment: Spectrum Analyzer

Preparation:

1. Connect 3156B Channel 1/2 output to the spectrum analyzer input. Use 50Ω and 20dB feedthrough termination at the spectrum analyzer input
2. Configure the 3156B channels 1/2 as follows:

Reset	
Waveform Mode:	MOD
Modulation Mode:	OFF
CW Frequency:	As required by the test
Amplitude	5V
Output:	On

Test Procedure

1. Perform sinewave spectral purity, DDS Waveforms tests on both channels using **Table 5-10**

Table 6-10, Sinewave Spectral Purity, Modulation Waveforms Tests

3156B	Reading	Spectrum Analyzer, Settings & Results					
CW Freq	**Limits**	**Start**	**Stop**	**CH 1**	**CH 2**	**Pass**	**Fail**
1 MHz	>55 dBc	100K	10M				
5 MHz	>40 dBc	1M	20M				
10 MHz	>35 dBc	1M	100M				
25 MHz	>30 dBc	1M	150M				

Flatness, DAC Waveforms

Equipment: Oscilloscope

Preparation:

1. Configure the Oscilloscope as follows:

Termination:	20dB, 50Ω feedthrough attenuator
Setup:	As required for the test

2. Connect 3156B Channel 1/2 output to the oscilloscope input. Place the feedthrough attenuator at the oscilloscope inputs
3. Configure the 3156B channels 1/2 as follows:

Reset	
Frequency:	Initially, 1kHz then, as required by the test
Amplitude:	6V
Output:	On
Enable:	On

Test Procedure

1. Adjust the vertical controls of the Oscilloscope to get 6 division of display
2. Perform Sine flatness, DAC waveforms tests on both channels using **Table 5-11**

Table 6-11, Sinewave Flatness, DAC Waveforms Tests

3156B Sine		**Oscilloscope Reading**			
Frequency	**Error Limits**	**CH 1**	**CH 2**	**Pass**	**Fail**
1kHz	6 Divisions	Reference	Reference	X	X
1MHz	6 ±0.2 Divisions				
10MHz	6 ±0.6 Divisions				
25MHz	6 ±0.9 Divisions				

Flatness, Modulation Waveforms

Equipment: Oscilloscope

Preparation:

1. Configure the Oscilloscope as follows:
   Termination: 20dB, 50Ω feedthrough attenuator
   Setup: As required for the test
2. Connect 3156B Channel 1/2 output to the oscilloscope input. Place the feedthrough attenuator at the oscilloscope inputs
3. Configure the 3156B channels 1/2 as follows:
   Reset
   Waveform Mode: MOD
   Frequency: Initially, 1kHz then, as required by the test
   Amplitude: 6V
   Output: On

Test Procedure

1. Adjust the vertical controls of the Oscilloscope to get 6 division of display
2. Perform Sine flatness, Modulation waveforms tests on both channels using **Table 5-12**

Table 6-12, Sinewave Flatness, Modulation Waveforms Tests

3156B Sine		**Oscilloscope Reading**			
Frequency	**Error Limits**	**CH 1**	**CH 2**	**Pass**	**Fail**
1kHz	6 Divisions	Reference	Reference	X	X
1MHz	6± 0.2 Divisions				
10MHz	6± 0.6 Divisions				
25MHz	6± 0.9 Divisions				

Trigger Operation

Trigger, Gate and Burst

Equipment: Oscilloscope, function generator, counter

Preparation:

1. Configure the Oscilloscope as follows:
   Termination: 20dB, 50Ω feedthrough attenuator
   Setup: As required for the test

2. Connect 3156B Channel 1/2 output to the oscilloscope input. Place the feedthrough attenuator at the oscilloscope inputs
3. Configure the function generator as follows:

Frequency:	10kHz
Run Mode:	As required by the test
Wave:	TTL level Square from the main output.

4. Connect the function generator output to the 3156B TRIG IN connector
5. Configure the 3156B channels 1/2 as follows:

Reset	
Frequency:	1.5625MHz
Waveform:	Sinewave
Burst Count:	1e6 counts, each channel
Amplitude:	5V
Enable Source:	Hardware
Output:	On
Enable:	On

Test Procedure

1. Perform trigger and gate tests using **Tables 5-13**
2. Configure the counter to TOTB Measurements and perform burst tests using **Tables 5-13**

Table 6-13, Trigger, gate, and burst Characteristics

3156B Run	**External Trigger**	**Oscilloscope Reading**			
Mode	**Pulse**	**CH 1**	**CH 2**	**Pass**	**Fail**
Triggered	10kHz, Continuous	Triggered waveform	Triggered waveform		
Gated	10kHz, Continuous	Gated Waveform	Gated Waveform		
Burst	10kHz, Single shot	Burst, 106 waveforms	Burst, 106 waveforms		

Delayed Trigger

Equipment: Function generator, 50Ω “T” connector, Counter, Wave CAD

Preparation:

1. Configure the Function generator as follows:

Amplitude:	1V
Frequency:	1MHz
Trigger Mode:	Triggered.
Wave:	Squarewave

2. Place the “T” connector on the output terminal of the function generator. Connect one side of the “T” to the 3156B TRIG IN connector and the other side of the “T” to the channel A input of the counter
3. Connect the 3156B output to channel B input of the counter
4. Configure the counter to TI A to B measurements
5. Using WaveCAD prepare and download the following waveform:

Wavelength:	100 points
Waveform:	Pulse, Delay = 0.1, Rise/Fall = 0, High Time = 99.99

6. Configure the 3156B channels 1/2 as follows:

Reset	
SCLK:	200MHz
Waveform:	Arbitrary
Run Mode:	Trigger
Trigger Level	0V
Trigger Delay:	On
Delay:	As required for the test
Amplitude:	5V
Enable Source:	Hardware
Output:	On

Test Procedure

1. Perform trigger delay tests using **Table 5-14**

Table 6-14, Trigger Delay Tests

3156B Delay Setting	Error Limits	Counter Reading	Pass	Fail
1μs	1μs ±150ns			
1ms	1ms ±50μs			
1s	1s ±50ms			
10s	10s ±500ms			

Re-Trigger

Equipment: Counter, WaveCAD

Preparation:

1. Configure the counter to pulse width measurements as follows:

Function:	Pulse Width Measurement
Channel A Slope:	Negative

2. Connect the counter channel A to the 3156B output
3. Using WaveCAD prepare and download the following waveform:

Wavelength:	100 points
Waveform:	Pulse, Delay = 0.1, Rise/Fall = 0, High Time = 99.99

4. Configure the 3156B channel 1 only as follows:

Reset	
SCLK:	200MHz
Waveform:	Arbitrary
Amplitude:	5V
Run Mode:	Trigger
Trigger Level	0V
Re-trigger:	On
Re-trigger Delay	As required for the test
Output:	On

Test Procedure

1. Perform re-trigger delay tests using **Table 5-14**

Table 6-15, Re-Trigger Delay Tests

3156B Re-trigger Setting	Error Limits	Counter Reading	Pass	Fail
1μs	1μs ±150ns			
1ms	1ms ±50 μs			
1s	1s ±50ms			
10s	10s ±500ms			

Trigger Slope

Equipment: Oscilloscope, function generator

Preparation:

1. Configure the Oscilloscope as follows:

Termination:	20dB, 50Ω feedthrough attenuator
Setup:	As required for the test
Trigger Source:	External

2. Connect 3156B Channel 1/2 output to the oscilloscope input. Place the feedthrough attenuator at the oscilloscope inputs
3. Configure the function generator as follows:

Frequency	10kHz
Run Mode:	Continue
Waveform:	TTL Output

4. Connect the function generator TTL output to the 3156B TRIG IN connector
5. Connect the function generator main output to the 2nd channel of the oscilloscope
6. Configure the 3156B channels 1/2 as follows:

Reset

Frequency:	1MHz
Waveform:	Sine wave
Run Mode:	Trigger
Enable Source:	Hardware
Output:	On
Enable:	On

Test Procedure

1. Toggle 3156B trigger slope from positive to negative visa versa
2. Verify on the oscilloscope that the 3156B transitions are synchronized with the slope of the trigger

Test Results	Pass		Fail	

Trigger Level

Equipment: Oscilloscope, function generator

Preparation:

1. Configure the Oscilloscope as follows:

Termination:	20dB, 50Ω feedthrough attenuator
Setup:	As required for the test
Trigger Source:	External

2. Connect 3156B Channel 1 output to the oscilloscope input. Place the feedthrough attenuator at the oscilloscope inputs
3. Configure the function generator as follows:

Frequency	10kHz
Run Mode:	Continuous
Waveform:	Squarewave.
Amplitude:	1V

4. Connect the function generator output to the 3156B TRIG IN connector
5. Configure the 3156B channel 1 as follows:

Reset	
Frequency:	1MHz
Waveform:	Sine wave
Run Mode:	Trigger
Trigger level:	0V
Enable Source:	Hardware
Output:	On

Test Procedure

1. Verify that the 3156B outputs triggered waveforms spaced at 0.1ms
2. Modify the function generator offset to +4V and change the 3156B trigger level to +4V. Verify that the 3156B outputs triggered waveforms spaced at 0.1ms
3. Modify the function generator offset to -4V and change the 3156B trigger level to -4V. Verify that the 3156B outputs triggered waveforms spaced at 0.1ms

Test Results	Pass		Fail	

Sequence Operation

Automatic Advance

Equipment: Counter

Preparation:

1. Configure the counter as follows:

Function:	TOTB Measurement

2. Connect the counter channel B to the 3156B output
3. Configure the 3156B channels 1/2 as follows:

Reset	
SCLK	200MS/s
Waveform:	Sequence
Run Mode:	Trigger
Amplitude:	2V
Output:	On

4. Using WaveCAD prepare and download the following waveform:

Segments:	1 to 5
Wavelength:	100 points
Waveform:	1 cycle square

5. Using WaveCAD, build and download the following sequence table:

Step 1:	Segment 1, loop 100,000
Step 2:	Segment 2, loop 100,000
Step 3:	Segment 3, loop 100,000
Step 4:	Segment 4, loop 100,000
Step 5:	Segment 5, loop 100,000

Test Procedure

1. From WaveCAD, select Enable On and observe that counter reading is 500,000 counts. Reset counter and repeat the test a few times. Every time the counter reading should be 500,000 counts exactly

Test Results	Pass		Fail	

2. Remove the cable from 3156B channel 1 and connect to channel 2
3. Repeat the test procedure as above for channel 2

Test Results	Pass		Fail	

Step Advance

Equipment: Oscilloscope, function generator

Preparation:

1. Configure the Oscilloscope as follows:

Termination:	20dB, 50Ω feedthrough attenuator
Setup:	As required for the test

2. Connect 3156B Channel 1 output to the oscilloscope input
3. Configure the function generator as follows

Frequency	10kHz
Run Mode:	Triggered
Waveform:	Squarewave.
Amplitude:	Adjust for TTL level on 50Ω

4. Connect the function generator output to the 3156B TRIG IN connector
5. Connect 3156B Ch1 to the Oscilloscope input
6. Configure the 3156B channels 1/2 as follows:

Reset	
SCLK	200MS/s
Waveform:	Sequence
Seq Advance:	Step
Amplitude:	2V
Enable Source:	Hardware
Output:	On

7. Using WaveCAD prepare and download the following waveform:

Segment 1:	Sine, 1000 points
Segment 2:	Triangle, 1000 points
Segment 3:	Square, 1000 points
Segment 4:	Sinc, 1000 points
Segment 5:	Gaussian Pulse, 1000 points

8. Using WaveCAD, build and download the following sequence table:

Step 1:	Segment 1, loop 1
Step 2:	Segment 2, loop 1
Step 3:	Segment 3, loop 1
Step 4:	Segment 4, loop 1
Step 5:	Segment 5, loop 1

Test Procedure

1. Press the manual trigger button on the function generator and observe that the waveforms advance through the sequence table repeatedly

Test Results	Pass		Fail	

2. Remove the cable from 3156B channel 1 and connect to channel 2
3. Repeat the test procedure as above for channel 2

Test Results	Pass		Fail	

NOTE:

Leave the same setup for the next test

Step Advance

Equipment: Oscilloscope, function generator

Preparation:

1. Use the same preparations as for step advance, except change mode to single sequence advance
2. Change Oscilloscope configuration to single:

Test Procedure

1. Press the manual trigger button on the function generator and observe that one cycle waveform advances through the sequence table repeatedly with each external trigger signal. Note that you need to press the Single mode on the oscilloscope for each trigger advance

Test Results	Pass		Fail	

2. Remove the cable from 3156B channel 1 and connect to channel 2
3. Repeat the test procedure as above for channel 2

Test Results	Pass		Fail	

Modulated Waveforms Characteristics

FSK

Equipment: Oscilloscope (LeCroy LT342, fitted with jitter package)

Preparation:

1. Configure the oscilloscope as follows:

Time Base:	0.1 ms
Sampling Rate:	50MS/s at least.
Trace A View:	Jitter, Type: FREQ, CLK.
Trigger source:	Channel 2, positive slope
Amplitude:	1V/div

2. Connect 3156B Channel 1 output to the oscilloscope input, channel 1
3. Connect the 3156B SYNC output to the oscilloscope input, channel 2
4. Configure the 3156B channels 1/2 as follows:

Reset	
Wave Mode:	Modulation
Modulation Mode:	FSK
CW Frequency:	2MHz
Shift Frequency:	4MHz
Baud Rate:	10kHz
Sync:	On
Output:	On
Enable:	On

5. Using WaveCAD, prepare and download 10-step FSK list with alternating “0” and “1”

Test Procedure

1. Verify FSK operation on the oscilloscope as follows:

Waveform:	Squarewave
Period:	0.1ms
Max A:	4MHz
Min A:	2MHz

Test Results	Pass		Fail	

2. Remove the cable from 3156B channel 1 and connect to chan 2
3. Repeat the test procedure as above for channel 2

Test Results	Pass		Fail	

Sweep

Equipment: Oscilloscope (LeCroy LT342, fitted with jitter package)

Preparation:

1. Configure the oscilloscope as follows:

Time Base:	0.1 ms
Sampling Rate:	50MS/s at least.
Trace A View:	Jitter, Type: FREQ, CLK.
Trigger source:	Channel 2, positive slope
Amplitude:	1V/div

2. Connect 3156B Channel 1 output to the oscilloscope input, channel 1
3. Connect the 3156B SYNC output to the oscilloscope input, channel 2
4. Configure the 3156B channels 1/2 as follows:

Reset	
Wave Mode:	Modulation
Modulation Mode:	Sweep
Start Frequency:	1MHz
Stop Frequency:	2MHz
Sweep Time:	1ms
Sweep Type:	Linear
Sync:	On
Output:	On
Enable:	On

Test Procedure

1. Verify FSK operation on the oscilloscope as follows:

Waveform:	Ramp Up
Period:	10kHz
Max A:	2MHz
Min A:	1MHz

Test Results	Pass		Fail	

2. Move 3156B sweep marker position to 1.5MHz and verify marker position at the middle of the ramp

Test Results	Pass		Fail	

3. Reverse between Start and Stop frequencies and verify oscilloscope reading as before except the ramp is down

Test Results	Pass		Fail	

4. Change sweep step to logarithmic and verify oscilloscope exponential down waveform with properties as in 7 above

Test Results	Pass		Fail	

5. Remove the cable from 3156B channel 1 and connect to chan 2
6. Repeat the test procedure as above for channel 2

Test Results	Pass		Fail	

FM – Std Waveforms, Continuous Run Mode

Equipment: Oscilloscope (LeCroy LT342, fitted with jitter package)

Preparation:

1. Configure the oscilloscope as follows:

Time Base:	0.1 ms
Sampling Rate:	50MS/s at least.
Trace A View:	Jitter, Type: FREQ, CLK.
Trigger source:	Channel 2, positive slope
Amplitude:	1V/div

2. Connect 3156B Channel 1 output to the oscilloscope input, channel 1
3. Connect the 3156B SYNC output to the oscilloscope input, channel 2
4. Configure the 3156B channels 1/2 as follows:

Reset	
Wave Mode:	Modulation
Modulation Mode:	FM
Carrier Frequency:	1MHz
Modulating Frequency:	10kHz
Deviation:	500kHz
Sync:	On
Output:	On
Enable:	On

Test Procedure

1. Verify FM operation on the oscilloscope as follows:

Waveform:	Sine
Period:	10kHz
Max A:	1.5MHz
Min A:	500kHz

Test Results	Pass		Fail	

2. Modify 3156B modulating waveform to triangle, then square and ramp and verify FM waveforms as selected

Test Results	Pass		Fail	

3. Move 3156B marker position to 1.5MHz and verify marker position

Test Results	Pass		Fail	

4. Remove the cable from 3156B channel 1 and connect to chan 2
5. Repeat the test procedure as above for channel 2

Test Results	Pass		Fail	

FM – Std Waveforms, Triggered Run Mode

Equipment: Oscilloscope (LeCroy LT342, fitted with jitter package), function generator

Preparation:

1. Configure the oscilloscope as follows:

Time Base:	0.1 ms
Sampling Rate:	50MS/s at least.
Trace A View:	Jitter, Type: FREQ, CLK.
Trigger source:	Channel 2, positive slope
Amplitude:	1V/div

2. Connect 3156B Channel 1 output to the oscilloscope input, channel 1
3. Connect the 3156B SYNC output to the oscilloscope input, channel 2
4. Configure the function generator as follows:

Frequency	1kHz
Run Mode:	Continuous
Waveform:	Squarewave.
Amplitude:	2V
Offset:	1V

5. Connect the function generator output connector to the 3156B TRIG IN connector
6. Configure the 3156B channels 1/2 as follows:
   Reset
   Wave Mode: Modulation
   Modulation Mode: FM
   Modulation Run Mode: Triggered
   Carrier Frequency: 1MHz

Modulating Frequency:	10kHz
Deviation:	500kHz
Enable Source:	Hardware
Sync:	On
Output:	On

Test Procedure

1. Verify triggered FM operation on the oscilloscope as follows:

Waveform:	Triggered Sine
Sine Frequency:	10kHz
Trigger Period:	1ms
Max A:	1.5MHz
Min A:	500kHz

Test Results	Pass		Fail	

FM – Std Waveforms, Burst Run Mode

Equipment: Oscilloscope (LeCroy LT342, fitted with jitter package), function generator

Preparation:

1. Configure the oscilloscope as follows:

Time Base:	0.1 ms
Sampling Rate:	50MS/s at least.
Trace A View:	Jitter, Type: FREQ, CLK.
Trigger source:	Channel 2, positive slope
Amplitude:	1V/div

2. Connect 3156B Channel 1 output to the oscilloscope input, channel 1
3. Connect the 3156B SYNC output to the oscilloscope input, channel 2
4. Configure the function generator as follows:

Frequency	1kHz
Run Mode:	Continuous
Waveform:	Squarewave.
Amplitude:	Adjust to TTL level on 50Ω

5. Connect the function generator output connector to the 3156B TRIG IN connector
6. Configure the 3156B channels 1/2 as follows:

Reset

Wave Mode:	Modulation
Modulation Mode:	FM
Modulation Run Mode:	Burst
Burst Count:	5
Carrier Frequency:	1MHz
Modulating Frequency:	10kHz
Deviation:	500kHz
Enable Source:	Hardware
Sync:	On
Output:	On

Test Procedure

1. Verify burst FM operation on the oscilloscope as follows:

Waveform:	Burst of 5 Sine cycles
Sine Frequency:	10kHz
Burst Period:	1ms
Max A:	1.5MHz
Min A:	500kHz

Test Results	Pass		Fail	

FM – Std Waveforms, Gated Run Mode

Equipment: Oscilloscope (LeCroy LT342, fitted with jitter package), function generator

Preparation:

1. Configure the oscilloscope as follows:

Time Base:	0.1 ms
Sampling Rate:	50MS/s at least.
Trace A View:	Jitter, Type: FREQ, CLK.
Trigger source:	Channel 2, positive slope
Amplitude:	1V/div

2. Connect 3156B Channel 1 output to the oscilloscope input, channel 1
3. Connect the 3156B SYNC output to the oscilloscope input, channel 2
4. Configure the function generator as follows:

Frequency	1kHz
Run Mode:	Continuous
Waveform:	Squarewave.
Amplitude:	Adjust to TTL level on 50Ω

5. Connect the function generator output connector to the 3156B TRIG IN connector
6. Configure the 3156B channels 1/2 as follows:

Reset

Wave Mode:	Modulation
Modulation Mode:	FM
Modulation Run Mode:	Gate
Carrier Frequency:	1MHz
Modulating Frequency:	10kHz
Deviation:	500kHz
Enable Source:	Hardware
Sync:	On
Output:	On

Test Procedure

1. Verify Gated FM operation on the oscilloscope as follows:

Waveform:	Burst of 5 Sine cycles
Sine Frequency:	10kHz
Gated Period:	1ms

Gate Duration:	0.5ms
Max A:	1.5MHz
Min A:	500kHz

Test Results	Pass		Fail	

FM – Std Waveforms, Re-triggered Burst Run Mod

Equipment: Oscilloscope (LeCroy LT342, fitted with jitter package), function generator

Preparation:

1. Configure the oscilloscope as follows:

Time Base:	0.1 ms
Sampling Rate:	50MS/s at least.
Trace A View:	Jitter, Type: FREQ, CLK.
Trigger source:	Channel 2, positive slope
Amplitude:	1V/div

2. Connect 3156B Channel 1 output to the oscilloscope input, channel 1
3. Connect the 3156B SYNC output to the oscilloscope input, channel 2
4. Configure the function generator as follows:

Frequency	1kHz
Run Mode:	Continuous
Waveform:	Squarewave.
Amplitude:	Adjust to TTL level on 50Ω

5. Connect the function generator output connector to the 3156B TRIG IN connector
6. Configure the 3156B channels 1/2 as follows:

Reset

Wave Mode:	Modulation
Modulation Mode:	FM
Modulation Run Mode:	Burst
Burst Count:	5
Carrier Frequency:	1MHz
Modulating Frequency:	10kHz
Deviation:	500kHz
Sync:	On
Re-trigger:	On
Re-trigger Delay:	200μs
Output:	On

Test Procedure

1. Using WaveCAD, select Enable On
2. Verify re-triggered FM burst operation on the oscilloscope as follows:

Waveform:	Repetitive bursts of 5-cycle sine waveforms
Sine Frequency:	10kHz
Re-trigger Delay:	200μs
Max A:	1.5MHz
Min A:	500kHz

Test Results	Pass		Fail	

FM – Arb Waveforms, Continuous Run Mode

Equipment: Oscilloscope (LeCroy LT342, fitted with jitter package)

Preparation:

1. Configure the oscilloscope as follows:

Time Base:	0.1 ms
Sampling Rate:	50MS/s at least.
Trace A View:	Jitter, Type: FREQ, CLK.
Trigger source:	Channel 2, positive slope
Amplitude:	1V/div

2. Connect 3156B Channel 1 output to the oscilloscope input, channel 1
3. Connect the 3156B SYNC output to the oscilloscope input, channel 2
4. Configure the 3156B channels 1/2 as follows:

Reset	
Wave Mode:	Modulation
Modulation Mode:	FM
Modulating Wave:	Arbitrary
Carrier Frequency:	1MHz
Modulating Frequency:	10kHz
FM SCLK:	5MS/s
Sync:	On
Output:	On
Enable:	On

5. Using WaveCAD, open the FM Composer and download the following waveform:

Wavelength:	20000 points
Waveform:	20 cycles sinewave
Deviation**:**	0.5MHz

Test Procedure

1. Verify FM operation on the oscilloscope as follows:

Waveform:	Sine
Period:	5kHz
Max A:	1.5MHz
Min A:	500kHz

Test Results	Pass		Fail	

2. Remove the cable from 3156B channel 1 and connect to chan 2
3. Repeat the test procedure as above for channel 2

Test Results	Pass		Fail	

Variable Dwell Time Frequency Hops

Equipment: Oscilloscope (LeCroy LT342, fitted with jitter package)

Preparation:

1. Configure the oscilloscope as follows:

Time Base:	0.1 ms
Sampling Rate:	50MS/s at least.
Trace A View:	Jitter, Type: FREQ, CLK.
Trigger source:	Channel 2, positive slope
Amplitude:	1V/div

2. Connect 3156B Channel 1 output to the oscilloscope input, channel 1
3. Connect the 3156B SYNC output to the oscilloscope input, channel 2
4. Configure the 3156B channels 1/2 as follows:

Reset	
Wave Mode:	Modulation
Modulation Mode:	HOP
Hop Mode:	Variable
Sync:	On
Output:	On
Enable:	On

5. Using WaveCAD, open the Hop Table composer and download the following table (both channels):

Frequency	***Dwell Time***
1.0e6	50e-6
1.2e6	100e-6
1.4e6	150e-6
1.6e6	200e-6
1.8e6	250e-6
2.0e6	300e-6
2.2e6	350e-6
2.4e6	400e-6
2.6e6	450e-6
2.8e6	500e-6

Test Procedure

1. Verify hop operation on the oscilloscope as follows:

Waveform:	Frequency steps, increasing dwell time from 50μs to 500μs
Period:	2750μs
Max A:	2.8MHz
Min A:	1.0MHz

Test Results	Pass		Fail	

2. Remove the cable from 3156B channel 1 and connect to chan 2
3. Repeat the test procedure as above for channel 2

Test Results	Pass		Fail	

Fixed Dwell Time Frequency Hops

Equipment: Oscilloscope (LeCroy LT342, fitted with jitter package)

Preparation:

1. Configure the oscilloscope as follows:

Time Base:	0.5 ms
Sampling Rate:	50MS/s at least.
Trace A View:	Jitter, Type: FREQ, CLK.
Trigger source:	Channel 2, positive slope
Amplitude:	1V/div

2. Connect 3156B Channel 1 output to the oscilloscope input, channel 1
3. Connect the 3156B SYNC output to the oscilloscope input, channel 2
4. Configure the 3156B channels 1/2 as follows:

Reset

Wave Mode:	Modulation
Modulation Mode:	HOP
Hop Mode:	Fixed
Sync:	On
Output:	On
Enable:	On

5. Using WaveCAD, open the Hop Table composer and download the following table (both channels):

Frequency
1.0e6
1.2e6
1.4e6
1.6e6
1.8e6
2.0e6
2.2e6
2.4e6
2.6e6
2.8e6

Test Procedure

1. Verify hop operation on the oscilloscope as follows:

Waveform:	Frequency steps, fixed dwell time of 50μs
Period:	500μs
Max A:	2.8MHz
Min A:	1.0MHz

Test Results	Pass		Fail	

2. Remove the cable from 3156B channel 1 and connect to chan 2
3. Repeat the test procedure as above for channel 2

Test Results	Pass		Fail	

AM

Equipment: Oscilloscope

Preparation:

1. Configure the oscilloscope as follows:
   Time Base: 0.5 ms
   Trigger source: Channel 2, positive slope
   Amplitude: 1V/div
2. Connect 3156B Channel 1 output to the oscilloscope input, channel 1
3. Connect the 3156B SYNC output to the oscilloscope input, channel 2
4. Configure the 3156B channels 1/2 as follows:
   Reset
   Wave Mode: Modulation
   Modulation Mode: AM
   Carrier Frequency: 1MHz
   Modulating Frequency: 1kHz
   Modulation Depth: 500kHz
   Sync: On
   Output: On
   Enable: On

Test Procedure

1. Verify AM operation on the oscilloscope as follows:
   Waveform: Amplitude Modulated Sine
   Modulation Depth: 10kHz

Test Results	Pass		Fail	

2. Remove the cable from 3156B channel 1 and connect to chan 2
3. Repeat the test procedure as above for channel 2

Test Results	Pass		Fail	

Digital Pattern Generator Characteristics

Digital Patterns - Freerun

Equipment: Oscilloscope, Digital pattern test board, Dual DC power supply

Preparation:

1. Turn chassis power OFF
2. Connect +5V and -5V to the test board
3. Hook up the test board on the channel 1 digital output connector
4. Turn chassis power ON
5. Turn power supply power ON

NOTE:

Do not attempt to connect the test board to the 3156B connector while power is ON as this may result in permanent damage to the 3156B. Always turn chassis power OFF before connecting or disconnecting the test board to the 3156B.

6. Connect 3156B Channel 1 output to the oscilloscope input, channel 1
7. Configure the 3156B channel 1 as follows:

   Reset

Pattern Rate:	10Mpps
Wave Mode:	Digital
Digital Mode:	Freerun
Enable:	On

8. Using WaveCAD, open the Digital Pattern Stim List Table, prepare and download the following table:

Index	*Stim List (hex)*	*Hold Count*
1	1	10000000
2	2	10000000
3	4	10000000
4	8	10000000
5	10	10000000
6	20	10000000
7	40	10000000
8	80	10000000
9	100	10000000
10	200	10000000
11	400	10000000
12	800	10000000

Test Procedure

1. Watch the LED's on the test board. The 12 MSD LED's light in sequence. Each LED lights for about 1 second
2. Set up the oscilloscope and check the output level. Verify output level is ECL

Test Results	Pass		Fail	

3. Repeat step 8 but change hold count to 100 for the entire list
4. Set up the oscilloscope and verify digital sequence of 10μs per pattern step

Test Results	Pass		Fail	

5. Turn chassis power off and remove the cable from 3156B channel 1
6. Repeat the test procedure as above for channel 2

Test Results	Pass		Fail	

Digital Patterns – Simulated Bursts

Equipment: Oscilloscope, Digital pattern test board, Dual DC power supply

Preparation:

1. Turn chassis power OFF
2. Connect +5V and -5V to the test board
3. Hook up the test board on the channel 1 digital output connector
4. Turn chassis power ON
5. Turn power supply power ON

NOTE:

Do not attempt to connect the test board to the 3156B connector while power is ON as this may result in permanent damage to the 3156B. Always turn chassis power OFF before connecting or disconnecting the test board to the 3156B.

6. Connect 3156B Channel 1 output to the oscilloscope input, channel 1

7. Configure the 3156B channel 1 as follows:

Reset	
Data Frequency:	1pps
Wave Mode:	Digital
Digital Mode:	Stimulus
Run Mode:	Burst
Burst Count:	5

8. Using WaveCAD, open the Digital Pattern Stim List Table, prepare and download the following table:

Index	***Stim List (hex)***
1	1
2	2
3	4
4	8
5	10
6	20
7	40
8	80
9	100
10	200
11	400
12	800

Test Procedure

1. Using WaveCAD, press the Enable On button
2. Watch the LED's on the test board. The 12 MSD LED's light in sequence. Each LED lights for about 1 second. The sequence is repeated for 5 times, then stops on the last step

Test Results	Pass		Fail	

3. Turn chassis power off and remove the cable from 3156B channel 1
4. Repeat the test procedure as above for channel 2

Test Results	Pass		Fail	

Video Stroke Characteristics

Equipment: Oscilloscope, function generator

Preparation:

1. Configure the oscilloscope as follows:

Time Base:	As required by the test
Amplitude:	2V/div

2. Connect 3156B Channel 1 output to the oscilloscope input, channel 1. Use 50Ω at the oscilloscope input

3. Connect 3156B Channel 2 output to the oscilloscope input, channel 2. Use 50Ω at the oscilloscope input
4. Configure the function generator as follows:

Frequency	10kHz
Run Mode:	Continuous
Waveform:	Squarewave.
Amplitude:	2V
Offset:	1V

5. Connect the function generator output connector to the 3156B TRIG IN connector
6. Configure the 3156B channel 1 and 2 controls as follows:

Reset	
Sample Clock:	100MS/s
Wave Mode:	Video
Start Offset:	-4V
Stop Offset:	+4V
Offset Step:	0.1V
Amplitude:	10mV
Run Mode:	Continuous
Burst Count:	5
Output:	On

Test Procedure

1. Verify trace on the oscilloscope for both channels show a ramp from -4V to +4V

Test Results	Pass		Fail	

2. Modify external function generator run mode to burst; burst count 80; manual trigger
3. Modify 3156B run mode to single
4. Observe trace reading of -4V
5. Manual trigger the function generator and observe that the oscilloscope trace changes to +4V
6. Note that subsequent external triggers do not change the trace level

Test Results	Pass		Fail	

7. Change 3156B configuration on both channels as follows:

Start Offset:	0V
Stop Offset:	0V
Offset Step:	10mV
Amplitude:	2V
Run Mode:	Single

8. Change oscilloscope configuration to XY mode and note cross-locate stroke character
9. Scroll through the stroke characters and observe the response on the oscilloscope

Test Results	Pass		Fail	

Adjustments

Introduction

This document contains the calibration procedure for the 200MS/s dual–channel Arbitrary Waveform Generator 3156B. A list of specifications is given in Appendix A of the Operations Manual. The calibration procedures that are described in this document are for use by qualified service personnel only. Do not perform these procedures unless qualified to do so. This procedure is intended to be used once before complete and final performance verification to verify that the 3156B meets or exceeds its published specifications. The calibration interval of the 3156B is 3 years.

Description

The 3156B is a VXI module. The output can generate a standard set of waveforms. These waveforms include Sine, Triangle, Square, Pulse, Ramp, Sinc, Gaussian, Exponential Decaying/Rising Pulse, Noise, and DC. Arbitrary waveforms are generated by downloading data to memory. Large waveform memories are provided so that multiple waveforms can be loaded at once. Powerful sequencing allows the waveform segments to be generated in any order. Communicating with the 3156B is done via the VXI bus.

Environmental Conditions

The 3156B can operate from 0°C to 50°C. Calibration should be performed under conditions of an ambient temperature of 0°C to 37.7°C and at a relative humidity of less than 80%. Turn on the power to the 3156B and allow it to warm up for at least 30 minutes before beginning the adjustment procedure. If the instrument has been subjected to conditions outside these ranges, allow at least one additional hour for the instrument to stabilize before beginning the adjustment procedure.

Initial Instrument Setting

To avoid confusion as to what initial setting is to be used for each calibration, it is required that the instrument be reset to factory default values prior to each adjustment.

Required Equipment

Recommended equipment for calibration is listed below. Instruments other than those listed may be used only if their specifications equal or exceed the required minimal characteristics. Also listed below are accessories required for calibration.

SMB to BNC cable

Dual banana to BNC adapter

50Ω Feedthrough termination

Equipment	Model No.	Manufacturer
Digital Multimeter	2000	Keithley
Oscilloscope	DSO	LeCroy
Counter	2201 w/ OPT. 42	EADS North America Defense Test and Services, Inc.

3156B Remote Calibration Procedure

Perform remote calibration on instruments that were prepared for this operation only. Follow the procedure as described below:

1. Invoke WaveCAD and click on the UTIL button. Click on the Calibration button.

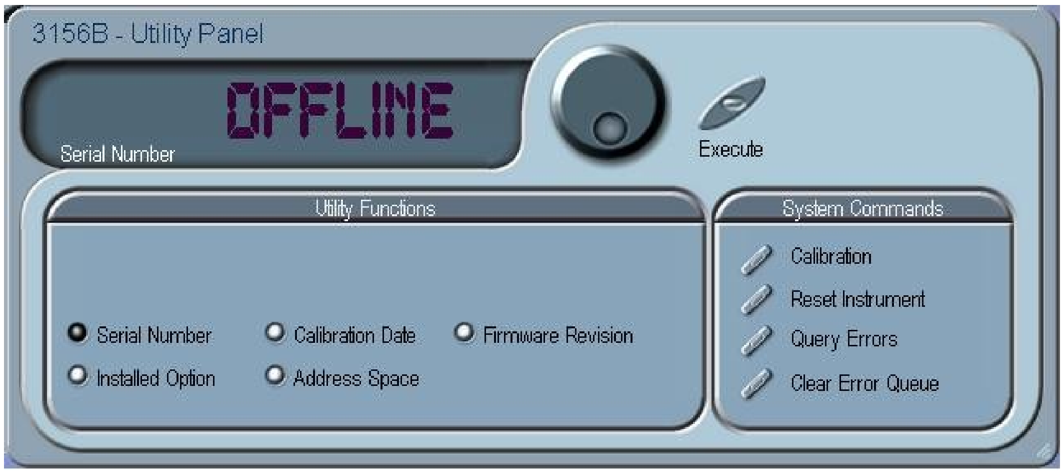


Figure 6-1, New Calibration Button

2. An Enter Password dialog box opens, as shown in **Figure 6-2**.

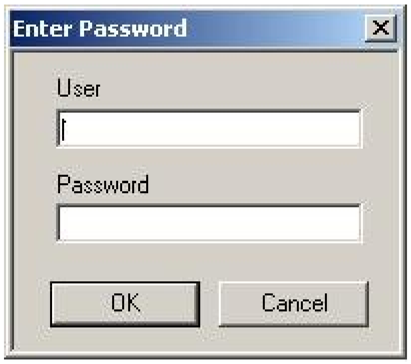


Figure 6-2, Enter Password Dialog Box

3. Type in the following

   User Name: Racal

   Password: 3333

4. The panel as shown in **Figure 6-3** will open

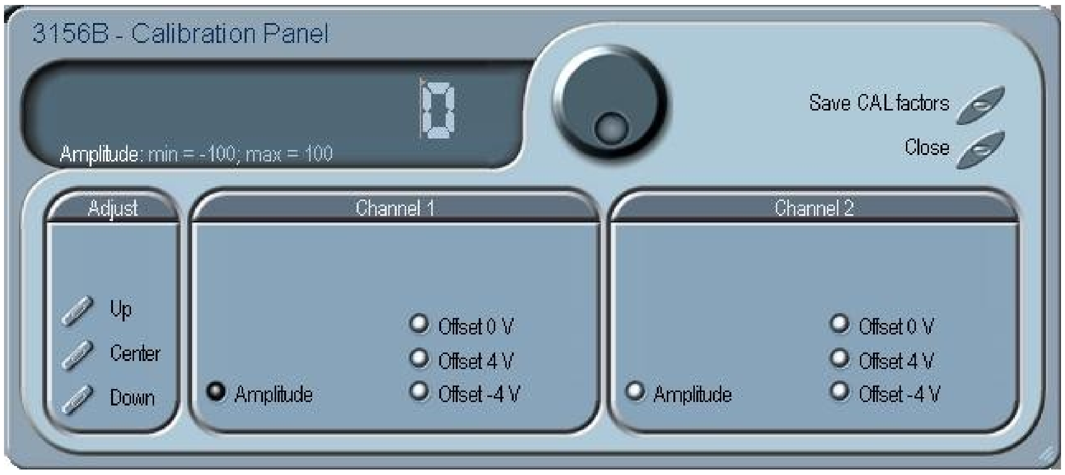


Figure 6-3, Calibration Pane

5. Program the following setup for both channels:

Output	On
Enable	On

Verify the channel output LED's on the UUT are on.

Amplitude Adjustment Channel 1

1. Click on Amplitude in the Channel 1 group
2. Click on Center in the Adjust group
3. Connect a DMM to the 3156B Channel 1 output. Set the DMM to AC, 10V measurements
4. Using the Dial or the Up and Down buttons in the Adjust group, adjust for DMM reading of 2.121V, ±10mV

Amplitude Adjustment Channel 2

1. Click on Amplitude in the Channel 2 group
2. Click on Center in the Adjust group
3. Connect a DMM to the 3156B Channel 2 output. Set the DMM to AC, 10V measurements
4. Using the Dial or the Up and Down buttons in the Adjust group, adjust for DMM reading of 2.121V, ±10mV

0” Offset Adjustment Channel 1

1. Click on Offset 0V in the Channel 1 group
2. Click on Center in the Adjust group
3. Connect a DMM to the 3156B Channel 1 output. Set the DMM to DC, 100mV measurements
4. Using the Dial or the Up and Down buttons in the Adjust group, adjust for DMM reading of 0V, ±20mV

“0” Offset Adjustment Channel 2

1. Click on Offset 0V in the Channel 2 group
2. Click on Center in the Adjust group
3. Connect a DMM to the 3156B Channel 2 output. Set the DMM to DC, 100mV measurements
4. Using the Dial or the Up and Down buttons in the Adjust group, adjust for DMM reading of 0V, ±20mV

“+4V” Offset Adjustment Channel 1

1. Click on Offset 4V in the Channel 1 group
2. Click on Center in the Adjust group
3. Connect a DMM to the 3156B Channel 1 output. Set the DMM to DC, 10V measurements
4. Using the Dial or the Up and Down buttons in the Adjust group, adjust for DMM reading of 4V, ±40mV. Note the DMM reading

“-4V” Offset Adjustment Channel 1

1. Click on Offset -4V in the Channel 1 group
2. Click on Center in the Adjust group
3. Connect a DMM to the 3156B Channel 1 output. Set the DMM to DC, 10V measurements
4. Using the Dial or the Up and Down buttons in the Adjust group, adjust for DMM reading of -4V, ±40mV
5. Repeat +4V and -4V Offset Adjustments until error is the same on both settings

“+4V” Offset Adjustment Channel 2

1. Click on Offset 4V in the Channel 2 group
2. Click on Center in the Adjust group
3. Connect a DMM to the 3156B Channel 2 output. Set the DMM to DC, 10V measurements
4. Using the Dial or the Up and Down buttons in the Adjust group, adjust for DMM reading of 4V, ±40mV. Note the DMM reading

“-4V” Offset Adjustment Channel 2

1. Click on Offset -4V in the Channel 2 group
2. Click on Center in the Adjust group
3. Connect a DMM to the 3156B Channel 2 output. Set the DMM to DC, 10V measurements
4. Using the Dial or the Up and Down buttons in the Adjust group, adjust for DMM reading of -4V, ±40mV
5. Repeat +4V and -4V Offset Adjustments until error is the same on both settings

!!!Warning!!!

The procedure above completes the field adjustments. However, make sure that you exit the Calibration panel only after you press the Save CAL Factors button. Press the Close button only after you saved the calibration factors to the 3156B memory.

Chapter 7
PRODUCT SUPPORT

Product Support

EADS North America Defense Test and Services, Inc. has a complete Service and Parts Department. If you need technical assistance or should it be necessary to return your product for repair or calibration, call 1-800-722-3262. If parts are required to repair the product at your facility, call 1-949-859-8999 and ask for the Parts Department.

When sending your instrument in for repair, complete the form in the back of this manual.

For worldwide support and the office closest to your facility, refer to the website for the most complete information http://www.eads-nadefense.com.

Warranty

Use the original packing material when returning the 3156B to EADS North America Defense Test and Services, Inc. for calibration or servicing. The original shipping container and associated packaging material will provide the necessary protection for safe reshipment.

If the original packing material is unavailable, contact EADS North America Defense Test and Services, Inc. Customer Service at 1-800-722-3262 for information.

REPAIR AND CALIBRATION REQUEST FORM

To allow us to better understand your repair requests, we suggest you use the following outline when calling and include a copy with your instrument to be sent to the EADS North America Defense Test and Service, Inc. Repair Facility.

Model____________________Serial No._________________________Date_______________________

Company Name_________________________Purchase Order #________________________________

Billing Address____

City

____

State/Province Zip/Postal Code Country

Shipping Address____

City

____

State/Province Zip/Postal Code Country

Technical Contact________________________Phone Number (   )______________________________
Purchasing Contact_______________________Phone Number (   )______________________________

1. Describe, in detail, the problem and symptoms you are having. Please include all set up details, such as input/output levels, frequencies, waveform details, etc.

____
____
____
____

2. If problem is occurring when unit is in remote, please list the program strings used and the controller type.

____
____
____
____

3. Please give any additional information you feel would be beneficial in facilitating a faster repair time (i.e., modifications, etc.)

____
____
____
____

4. Is calibration data required?   Yes   No   (please circle one)

Call before shipping Note: We do not accept “collect” shipments.	Ship instruments to nearest support office.

Appendix A SPECIFICATIONS

VXIbus General Information

Module Form	Single slot VXIbus C-size module
Connectors	P1, P2
Protocol	A24/A32/D16, Register-Based
VXIbus Revision	2.0
SCPI Revision	1993.0
Logical Address Settings	1 - 255, configured via DIP switches
Interrupt Level Settings	1 - 7, configured dynamically (no DIP switch)
Drivers	LabVIEW, LabWindows/CVI, VXIplug&play (WIN2000/XP)
Waveform Creation and Control Software	WaveCAD (WIN2000/XP)
Calibration Interval	3 years

Module General Information

Configuration	Two semi-independent channels
Channel Dependencies	Each channel uses its own sample clock and trigger.
Function Modes	Common to both channels except when in modulation and half cycle, where second channel, if not set to the same function, outputs continuous AC signal
Run modes	Common to both channels
Common parameters	Sample clock frequency, reference source, trigger source
Separate parameters	Amplitude, offset, waveform shape and its parameters, SYNC output and output disable

Skew between channels	
Sample Clock Mode 100MS/s	
1S/s to 25MS/s	3ns
25MS/s to 75MS/s	½ SCLK +3ns
75MS/s to 100MS/s	1 SCLK +3ns
Sample Clock Mode 200MS/s	
1S/s to 50MS/s	3ns
50MS/s to 150MS/s	1 SCLK +3ns
150MS/s to 200MS/s	2 SCLK +3ns

Sampling Clock

Internal Source Range	1Hz to 200MHz
Resolution	10 digits
Accuracy and Stability	Same as reference
Reference Clock	
CLK10	100ppm, backplane clock, or Internal: 1 ppm, 0 to 50° (with optional TCXO)
External	Front Panel SMB

Amplitude Characteristics

Amplitude	20mV to 20Vpk-pk, output open circuit
	10mV to 10Vpk-pk, into 50Ω
Resolution	4 digits
Accuracy (measured at 1kHz)	1V to 10Vpk-pk: ±(1% + 25mV)
	100mV to 1Vpk-pk: ±(1% + 5mV)
	10mV to 100mVpk-pk: ±(1% + 2mV)
DC Offset Range	0 to ±4.995V

DC Offset Resolution	1mV
DC Offset Accuracy	±(1% ± 1% from Amplitude ±5mV)

Run Modes (applies for Standard, Arbitrary, Sequencer and Modulated waveforms)

Continuous Mode	Continuous output of a waveform after a software or hardware Enable ON command. Continuous mode disabled with software only Enable OFF command
Triggered Mode	Output of one waveform cycle following an Enable ON command. Last cycle always completed
Burst Mode (does not operate in	
Sequencer Mode)	Output of a single or multiple pre-programmed number of waveform cycles (burst) starting after a software or hardware Enable ON command.
Counted Burst Cycles	1 to 1Meg, programmable
Burst Limitations	For segments below 8 points: Segment duration must be >500ns
	For segments Above 8 points: Segment duration must be >40ns
Gated Mode	Hardware or backplane transition enables or disables generator output. Last cycle always completed
Run Modes Enable Source	Software: Enable ON/OFF command
Hardware	Front panel TRIG IN, VXI Backplane: TTLTrg0-7
Mixed	Output of one cycle following a software Enable ON command. Subsequent outputs enabled by hardware, or backplane triggers

Trigger Characteristics

Input Sources	External: Front panel SMB
	VXI Backplane: TTLTrg0-7
Trigger Level	Range: ±5V
Resolution	1mV
Input Frequency Range	DC to 5MHz
Pulse Width	10ns, min

Slope	Positive/Negative transitions, selectable
Trigger Out	VXI Backplane: TTLTrg0-7
System Delay (Trigger I/P to waveform O/P)	6 sample clock cycles+150ns
Trigger Delay (Enable cmd to event O/P)	0; 500ns to 21s
Trigger Delay Error	6 sample clock cycles+150ns ± 5% of delay setting
Retrigger Delay (Event O/P end to event O/P restart)	500ns to 21s
Retrigger Delay Error	3 sample clock cycles+150ns ± 5% of retrigger delay setting
Trigger/Retrigger Delay Resolution	20ns
Trigger Jitter	±1 SCLK

Standard Waveforms

Frequency Range	100µHz to 25MHz
Frequency Resolution	10 digits
Accuracy & Stability	Same as frequency standard

Sine

Start Phase Range	0-359.95°
Start Phase Resolution	0.05°
Sine Total Harmonic Distortion	0.3% to 100kHz
Harmonics and Spurious (max amplitude)	30dBc, <25MHz
	40dBc, <10MHz
	45dBc, <5MHz
	55dBc, <1MHz
Flatness	5% to 10MHz
	10% to 25MHz

Triangle

Start Phase Range	0-359.95°
Start Phase Resolution	0.05°

Square

Duty Cycle Range	0% to 99.9%
Rise/Fall Time (10%-90%)	<8ns
Aberration	<5%+10mV

Pulse and Ramp Functions

Delay, Rise/Fall Time, High Time Ranges	0%-99.9% of period (each independently)
Gaussian Pulse Time Constant Range	10-200
Sinc Pulse “Zero Crossings” Range	4-100
Exponential Pulse Time Constant Range	-100 to 100

DC Output Function

Range	-100% to 100% of amplitude

Half-Cycle Waveforms

Function Shape (other channel either in half-cycle mode or AC continuous signal)	Sine, Triangle, Square
Frequency Range	100□Hz to 2MHz
Phase Start Range (Sine and triangle only)	0° to 359.95°
Start Phase Resolution	0.05°
Run Modes	Continuous, Triggered
Delay Between Half Cycles (Applies to continuous run mode only)	500ns to 21s
Delay Resolution	20ns

Arbitrary Waveforms

Vertical Resolution	12 or 16 bits, user selectable
Waveform Segmentation	Permits division of waveform memory into smaller segments. Segments shorter than 8 points occupy location of 4 segments (32 points)
Number of Memory Segments	1 to 16k, if all segments are longer than 8 points; 1 to 4096, if all segments are less than 8 points
Waveform Segments, size and resolution	
Sclk Mode - 1S/s to 100MS/s	1 point size increments from 1 to 512k points
Sclk Mode - 1S/s to 200MS/s	2 points size increments from 2 to 1Meg points,
Custom Waveform Creation Software	WaveCAD software allows instrument control and creation of custom waveforms either freehand, with equations or built-in functions or with imported waveforms

Sequenced Waveforms

Operation	Segments may be linked and repeated in a user-selectable fashion to generate extremely long waveforms. Segments are advanced using either a command or a trigger
Advance Modes	
Automatic Sequence Advance	No trigger required to step from one segment to the next. Sequence is repeated continuously per a pre-programmed sequence table.
Stepped Sequence Advance	Current segment is sampled continuously until a trigger advances the sequence to the next programmed segment and sample clock rate.
Single Sequence Advance	Current segment is sampled the specified number of repetitions and then idles at the end of the segment. Next trigger samples the next segment the specified repeat count, and so on.
Sequencer Steps	1 to 4096
Segment Loops	1 to 1Meg
Minimum Segment Duration	500ns
Minimum Segment Size in a Sequence	8 points

Modulated Waveforms

Run Modes	Continuous, Triggered, Burst and Gated
Run Mode Advance Source	Software commands, Front panel TRIG IN, Backplane TTLTrg0-7
Trigger Delay Range (Enable cmd to modulated O/P)	0, 500ns to 21s
Re-trigger Delay Range (Modulated O/P end to modulated O/P restart)	500ns to 21s
Trigger Parameters	All trigger parameters such as level, slope, jitter, etc. apply

Sweep

Swept Waveform	Sine wave
Sweep Step	Linear or log
Sweep Direction	Up or Down
Sweep Range	100μHz to 25MHz
Sweep Time	1.4μs to 40s
Marker Output	Programmable marker at a selected frequency.

FM

Modulated Waveform	Sine wave
Modulating Waveforms	Sine, square, triangle
Carrier Frequency Range	1Hz to 25MHz
Modulating Frequency Range	10mHz to 100kHz
Peak Deviation	Up to 25MHz
Marker Position	Programmable at selectable a frequency

ARBITRARY FM

Operation	Operated from and external utility only such as WaveCAD
Modulated Waveform	Sine wave
Carrier Frequency Range	1Hz to 25MHz

Modulating Waveform	Arbitrary waveform
Modulating Waveform Sampling Clock	1S/s to 5MS/s
Frequency Array Size	4 to 20000 frequencies
Marker Output	Programmable at a selected frequency

AM

Modulated Waveform	Sine wave
Carrier Frequency Range	1Hz to 25MHz
Envelop Waveform	Sine wave
Envelop Frequency	10mHz to 100kHz
Modulation Depth	0% to 100%

FSK

Shifted Waveform	Sine wave
Carrier/Shifted Frequency Range	100μHz to 25MHz
Baud Rate Range	1bits/sec to 10Mbits/sec
FSK Data Bits Length	1-4096
Marker Output	Programmable marker at a selected frequency

FREQUENCY HOPPING

Hopped Waveform	Sine wave
Hop Table Size	1 to 4096
Dwell Time Mode	Fixed or Programmable for each step
Dwell Time	500 ns to 21 s
Dwell Time Resolution	20 ns
Hop Frequency Range	100μHz to 25MHz
Resolution	10 digits
Marker Position	Programmable on a selected frequency step

Video Stroke Generation

Built-in Patterns	Cross-Locator, Cross-Hair, Positioned-Square, Vertical-Marker-Line, Horizontal-Marker-Line, Right-Hand-Arrow, Left-Hand-Arrow, Diamond-Overlay, Inverted-Triangle, Upright-Triangle
DC Offset Range	-4.995V to +4.995V
Step Size	±1mV to ±9.99V
Step Rate	1 step per trigger
Run Mode	Continuous, Single
Pattern Generation Period	167ns to 100s
Minimum Trigger Period	Pattern Period + 8 clock cycles

Digital Pattern Outputs

Pattern Size	12-bits, ECL levels differential, each channel
Update Frequency	100µHz to 100MHz
Number of Patterns	1 to 512K
Output Mode	
Free-Run	Has programmable hold time for each pattern
Stimulus	Has fixed hold time for all steps
Hold Time Range	1 to 1,000,000,000 to 50MS/s (hold time >1080ns and <21s); 54 to 2,000,000,000 to 100MS/s

Front Panel I/O’s

Main Outputs

	Connector: SMB, each channel
	Impedance: 50□ ±1%
Protection	Short Circuit to Case Ground, 10s max
Standby	Output On or Off (Output Disconnected)

Sync Outputs

Connector	SMB, each channel
Level	TTL

Sync Type:	Zero Cross A/B or Pulse with Arbitrary and Standard Waves; LCOM in Sequence and Burst Modes (including Burst Modulation); Marker with Modulation Mode only, programmable position
Digital Word Outputs	
Connector	50-pin VHDC
Level	Differential ECL externally terminated into 50Ω to –2V
Trigger Input	
Connector	SMB
Impedance	50Ω ±1%
Slope	Positive or Negative (selectable)
Programmable Level	±5V
Sensitivity	100mVp-p
Damage Level	±8V
Pulse Width	10ns minimum
External Reference Input	
Connector	SMB
Frequency	10MHz
Impedance	50Ω ±5%
Level	0dBm, sine
Damage Level	1V rms

Environmental

Temperature	Operating: 0°C-50°C
	Storage: -40°C-70°C
Spec Compliance	20°C-30°C
Humidity (non-condensing)	
	11°C-30°C: 95% ±5%
	31°C-40°C: 75% ±5%
	41°C-50°C: 45% ±5%

Altitude	
	Operating: 10,000ft
	Storage: 15,000ft
Cooling (10°C Rise)	3.7l/s @ 0.5mm H_2O
Total Power	<50Watts
Vibration (non-operating)	2g at 55Hz
Shock (non-operating)	30g, 11ms, half sine pulse
Weight	31bs. 8oz. (1.6kg)