PLEXON
Neurotechnology Research Systems

User Guide

Software Version 2   January 2017

PlexStim™
Electrical Stimulator

Constant Current Stimulation System

Plexon Inc
6500 Greenville Avenue, Suite 700
Dallas, Texas 75206
USA

CAUTION

THIS STIMULATOR IS NOT FOR USE IN HUMANS.

CAUTION

READ THE ENTIRE MANUAL BEFORE ATTEMPTING TO OPERATE THIS EQUIPMENT.

IMPORTANT: READ THE VERSION AND COMPATIBILITY NOTICE IN SECTION 2 BEFORE ATTEMPTING TO OPERATE THIS EQUIPMENT.

CAUTION

OBSERVE THE FOLLOWING SEQUENCE DURING ALL EXPERIMENTS:

1. **TURN THE STIMULATOR POWER ON**
2. **LAUNCH THE USER INTERFACE PROGRAM**
3. **CONNECT THE STIMULATOR TO THE ELECTRODES**
4. **CONDUCT THE EXPERIMENT**
5. **CLOSE THE USER INTERFACE PROGRAM**
6. **DISCONNECT THE STIMULATOR FROM THE ELECTRODES**
7. **TURN THE STIMULATOR POWER OFF**

Contact Plexon support at +1 214-369-4957 or support@plexon.com if you would like additional information or instructions.

PlexStim™ Electrical Stimulator

Constant Current Stimulation System

User Guide

Document Number: STMMN0001b

Software Version 2

Date: January 2017

Publication History

January 2017

This user guide is based on PlexStim™ Electrical Stimulator, Version 2.3. It has been updated to incorporate the following corrections:

- The digital input/output latency is approximately 2 µs. It was previously listed in this document as less than 1 µs.
- The maximum number of points allowed in a fixed pattern loaded from the GUI is 999. It was previously listed in this document as 1,000.
- A variable pattern can hold up to 499 amplitude duration pairs. It was previously listed as 500.

The user guide is periodically updated and reissued, typically in conjunction with a new software or hardware release. You can see a summary of changes by accessing the Change Log for this product on the Plexon® website, www.plexon.com.

May 2015

This user guide is based on PlexStim™ Electrical Stimulator, Version 2.3.

Version 2.3 includes the following components:

- Hardware part number 14-20-A-10-F
- Firmware part number 14-20-A-07-A
- Software GUI Version 2.3.0.0

IMPORTANT NOTICE: Version 2.3.0.0 GUI will ONLY work with Plexon® stimulators that have the updated firmware (14-20-A-07-A). Stimulators that were manufactured prior to April 2015 with revision blank firmware (14-20-A-07) will generally not be recognized by GUI Version 2.3.0.0 as valid stimulators. See Section 2, "Version and Compatibility Notice" on page 2 for additional details and required actions.

This version incorporates the following changes:

- Up to 64 channels can be operated simultaneously by means of a software development kit (SDK) available on the Plexon website. (Two PlexStim Stimulator systems are needed to provide up to 32 channels. Three systems provide up to 48 channels. Four systems provide up to 64 channels.) See Section 1, "Introduction" on page 1.
- There is a change in the behavior of the stimulator in the "Level" triggered digital input mode. See Section 7.7, "Starting Stimulation from a Digital Input" on page 25 and Section 8.8.3, "Level Trigger Mode" on page 37 for a description of the new behavior.

April 2011

This is the first release of the user guide. It is based on PlexStimElectrical Stimulator, Version 2.0, which includes the following components:

- Hardware part number 14-20-A-10-A
- Firmware part number 14-20-A-07
- Software GUI part number 14-20-A-08

Contents

Publication History v

Introduction 1
Version and Compatibility Notice 2
System Requirements 2
System Components 3
Installation 4
  Software and Driver Installation 4
  Software Development Kits (SDKs) 6
  Hardware Installation 6
  Turning On the Power for the First Time 7
Getting Started 8
  Overview 8
  Launching the PlexStim Electrical Stimulator Software 9
  Creating Rectangular Pulses 10
  Verifying the Output on an Oscilloscope 12
Procedures and Examples 15
  Operational Flowchart 16
  Compliance Voltage and Stimulation Failure 17
  Automatic Electrode Discharge 18
  Arbitrary Waveforms and Complex Rectangular Pulses 19
  Impedance Measurement 21
  Working with Real Electrodes 24
  Starting Stimulation from a Digital Input 26
  Stopping Stimulation 28
GUI Function Reference 29
  Parameters Edit/Load 29
  Stimulation Stop/Start 29
  Rectangular Pulse Parameters 30
  Number of Repetitions and Pulse Rate 32
  Arbitrary Waveform Patterns 32
  Vmon Scaling and Z Conversion 36
  Function Stimulate / Function Z Test 37
  Trigger Mode 37

Edit/Load All 39
Start Mode 39
Monitor Channel 40
ABORT 40
File Open / File Save 42
Options: Discharge Mode 42
Options: Digital Output Mode 43
Input and Output Connectors 45
Power In 45
Digital In 45
USB 2.0 45
Stim Out 46
Digital Out 46
Current Monitor 47
Voltage Monitor 47
Stimulation Cable 49
Ground/Common Access Points 49
PlexStim Stimulator Limitations 50
Maximum Compliance Voltage Varies with Stimulation Amplitude 50
Power on/off Transients 50
Current Monitor with an Open Circuit 51
Impedance Test with an Open Circuit 52
Specifications 53
List of Related Documents 54

Appendix A
Introduction A-2
Hardware Compatibility A-2
Basic DLL Import A-3
Correcting Import Errors A-11
PS_GetArbPatternPointsX A-11
PS_GetArbPatternPointsY A-14
PS_GetArbPatternPoints A-15
Fix Error Reporting A-15
Improve Error Description Reporting (Optional) A-16
Special Note on Close Functions and Error Handling A-18
Updating User Programs A-19

PlexStim™ Electrical Stimulator

1 Introduction

The Plexon® PlexStim™ Electrical Stimulator (referred to in this user guide as the PlexStim Stimulator) is a 16-channel constant current stimulation system. It has 16 individually programmable constant current sources (channels) that share a common return path.

In Version 2.3 and later, up to 64 channels can be operated simultaneously by means of a software development kit (SDK) available on the Plexon website. (Two PlexStim Stimulator systems are needed to provide up to 32 channels. Three systems provide up to 48 channels. Four systems provide up to 64 channels.) See Section 5.2, "Software Development Kits (SDKs)" on page 6 for information on obtaining and using the SDKs.

The PlexStim Electrical Stimulator provides the following capabilities. Parameters for these capabilities can be configured in the graphical user interface (GUI):

- Define stimulation currents with 16-bit precision up to ±1 mA and deliver the specified currents with ±10 V compliance.
- Define bi-phasic rectangular pulses with 30nA precision and 1µsec temporal accuracy.
- Repeat bursts of pulses at a user defined rate.
- Load arbitrary waveform patterns from text files.
- Initiate playback of pulses and arbitrary stimulation waveforms from the GUI or allow playback to be triggered in response to external digital inputs.
- Initiate stimulation independently on each channel. (Each channel has a dedicated digital input that may be used in an edge-triggered or level-triggered [gated] mode to initiate stimulation with a latency of about 2 microseconds.)
- Provide signals to external devices that identify the precise time when the stimulation is occurring. (Each channel has a dedicated digital output for exporting this data.)
- Monitor the actual current and voltage delivered to any electrode and conduct impedance testing on each channel. (The system provides the current and voltage data through standard BNC connectors, allowing you to view this data on an oscilloscope. The PlexStim Electrical Stimulator impedance test mode provides nanoamp resolution and additional filtering and programmable gain for the monitor channel outputs so that the voltage elicited in response to sub-microamp currents may be resolved for impedance measurements.)

2 Version and Compatibility Notice

In April 2015, a new version of the PlexStim Stimulator GUI, Version 2.3.0.0, was released in conjunction with new firmware for the PlexStim Stimulator (Firmware PN 14-20-A-07-A). The Version 2.3.0.0 GUI will ONLY work with PlexStim Stimulators that have the updated Firmware. PlexStim Stimulators that were manufactured prior to April 2015 with revision blank firmware (14-20-A-07) will generally not be recognized by GUI Version 2.3.0.0 as valid systems.

PlexStim Stimulators of hardware Revision F (14-20-A-10-F) are the first stimulators originally manufactured with Revision A firmware. Examine the labels on the bottom of the stimulator to determine the firmware and hardware revision. If your stimulator is not Revision F or later or does not have Revision A or later firmware, contact Plexon support for information on upgrading the firmware in your stimulator. Please note that after the stimulator hardware and firmware have been upgraded you will be required to use the Revision 2.3.0.0 or later GUI (or Revision 2.3.16.0 SDK or later) to operate the stimulator. Prior versions of the GUI will not function properly with the updated firmware.

Technical Support

Plexon technical support is available by telephone at +1 214-369-4957 or by email at support@plexon.com.

3 System Requirements

A modern personal computer running Windows® 7 with a free USB 2.0 port is required to operate the system. An oscilloscope is highly recommended for monitoring the actual current and voltage waveforms during stimulation. However, note that many oscilloscope inputs are earth grounded. Normally the current outputs of the stimulator are isolated from earth ground. Connecting the stimulator monitor outputs to an earth grounded oscilloscope will cause the current outputs to not be isolated from earth ground.

4 System Components

When you receive your PlexStim Stimulator system, confirm that you have the following pieces:

1	USB memory with software & drivers	Plexon	14-20-A-14-C
2	AC power cord (7.5 ft)*	Volex	17250 10 B1
3	Power Supply	Plexon	08-06-A-37
4	USB Cable (2m)	Molex	88732-8902
5	Stimulator	Plexon	14-20-A-10-F
6	Stimulation Cable	Plexon	14-03-A-03
7	Test Board (model electrodes)	Plexon	14-04-A-03-A

* International customers: The stimulator power supply (**Item 3** in the above image) has an International Electrotechnical Commission (IEC) 60320 C14 inlet for AC power (shown below). The AC power cord supplied with the stimulator has an IEC 60320 C13 connector and a NEMA 5-15 plug compatible with the AC wall outlets in North America. Users in other regions must supply a power cord with an IEC 60320 C13 connector and a plug that is compatible with the AC wall outlets in the region of use.

5 Installation

Read the entire installation section before proceeding with the installation. Follow the installation steps in the order that they are presented. Do not connect the stimulator to an electrode implant or attempt to use the stimulator until you have read the entire manual.

You may also wish to read through Section 12, "List of Related Documents" on page 54.

5.1 Software and Driver Installation

Read the entire Software and Driver Installation section before proceeding with any of the steps. Note that you should be logged onto the computer as a user with administrative privileges before proceeding with the installation process. The installer will install the necessary drivers, the graphical user interface (GUI), and the software development kits (SDKs) onto the computer. At the end of the installation, the installer will ask you to reboot the computer.

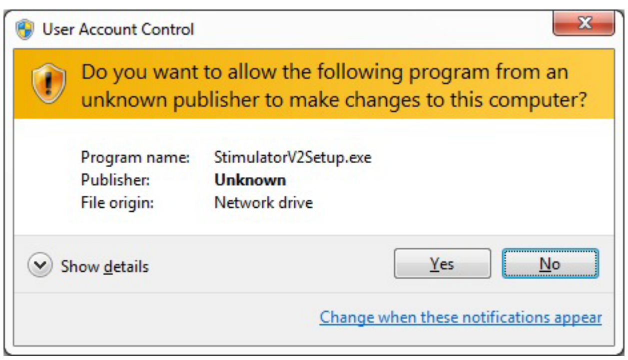
1 On the root of USB flash disk, locate and read the End User License Agreement (eula.pdf).

2 Log onto the target computer using an account that has administrative privileges.

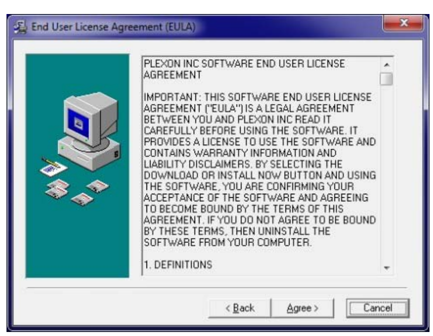
3 Browse to the folder \PlexStimV2\Installer. Copy the file StimulatorV2Setup.exe to the desktop.

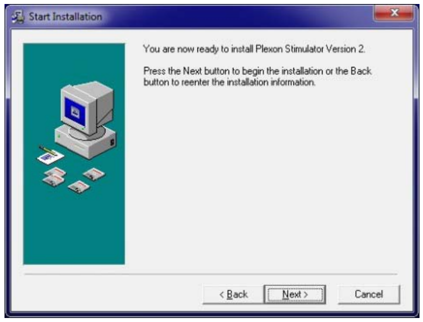
4 Right click on StimulatorV2Setup.exe and select "Run as administrator."

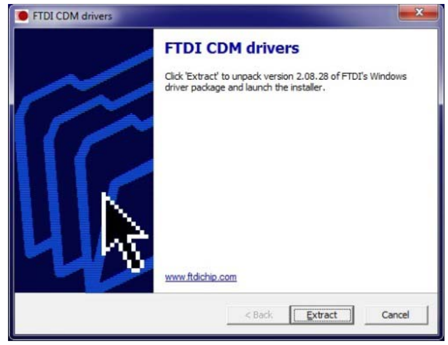
5 You may get a warning that the publisher of the software is unknown (to Microsoft®). Select **Yes** to proceed.

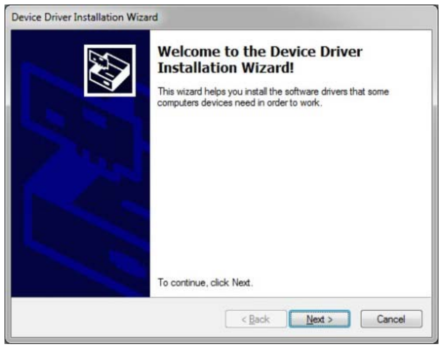


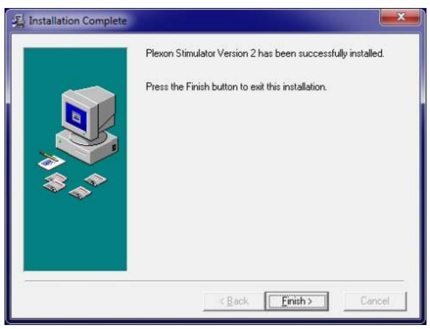
6 Follow the prompts from the installer. There are approximately eight dialogs that you will need to respond to:

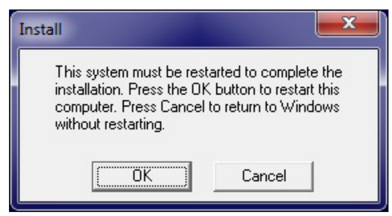












7 Note that the default installation location for the graphical user interface is C:\Program Files (x86)\Plexon Inc\Stim-2\Stim-2.exe.

5.2 Software Development Kits (SDKs)

In addition to installing the drivers and GUI, the installer also installs 32 bit and 64 bit SDKs for both C++ and MATLAB®. The default location for the SDKs is C:\PlexonSDKs. This manual does not specifically address the use of the SDKs. Nonetheless you are strongly encouraged to read the manual and familiarize yourself with the hardware and use of the GUI before attempting to use the SDKs.

See Section 12, "List of Related Documents” on page 54 for further supporting documentation.

Instructions for how to import the PlexStim DLL into LabVIEW™ can be found in Appendix A.

5.3 Hardware Installation

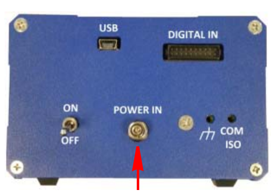
Read the entire Hardware Installation section before proceeding with any of the steps.

An oscilloscope is highly recommended for viewing the output of the monitor channels.

1 Ensure that the power switch on the stimulator is in the off position.

2 Connect the AC power cord between the AC outlet and the power supply.

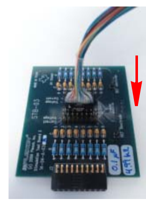
3 Plug the DC power cord into the stimulator.



4 Connect the USB cable between the stimulator and the computer.

5 IMPORTANT: Turn on the stimulator power and launch the stimulator GUI before connecting the stimulator to any electrodes.

6 Connect the stimulation cable between the stimulator and the test load board. Be sure to line up the white orientation dots on both connectors.

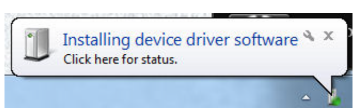


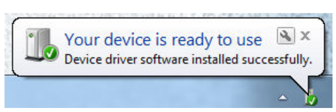
7 Connect BNC cables between the monitor channel outputs and an oscilloscope. Monitoring the current and voltage delivered to the electrode is an essential part of stimulation.

5.4 Turning On the Power for the First Time

1 Turn the power switch to the on position. The LED on the front of the stimulator should illuminate.

2 The first time you turn the stimulator on a “Found New Hardware” balloon will appear in the lower right hand corner of the computer screen. The balloon changes from “Installing device driver software” to “Your device is ready to use.”





3 If the “Found New Hardware Wizard” appears, then the drivers for the stimulator are *not* properly installed. Turn the stimulator power off and refer to section Section 5.1, “Software and Driver Installation” on page 4 for instructions on installing the drivers and software.

6 Getting Started

This section introduces the basic functions of the stimulator and guides you through basic stimulator operation using the model electrodes on the test board. It is highly recommended that you work through this section before attempting to do any other type of stimulation. Note that you will need an oscilloscope to see the output of the monitor channels.

Note: This Getting Started section is limited to applying a basic rectangular pulse pattern to the electrodes. The procedures for applying a wider range of stimulation patterns are presented in Section 7, "Procedures and Examples" on page 15.

IMPORTANT:
OBSERVE THE FOLLOWING SEQUENCE DURING ALL EXPERIMENTS:

1. TURN THE STIMULATOR POWER ON
2. LAUNCH THE USER INTERFACE PROGRAM
3. CONNECT THE STIMULATOR TO THE ELECTRODES*
4. CONDUCT THE EXPERIMENT
5. CLOSE THE USER INTERFACE PROGRAM
6. DISCONNECT THE STIMULATOR FROM THE ELECTRODES
7. TURN THE STIMULATOR POWER OFF

* Use ONLY the model electrodes on the test board while you are learning to operate the system.

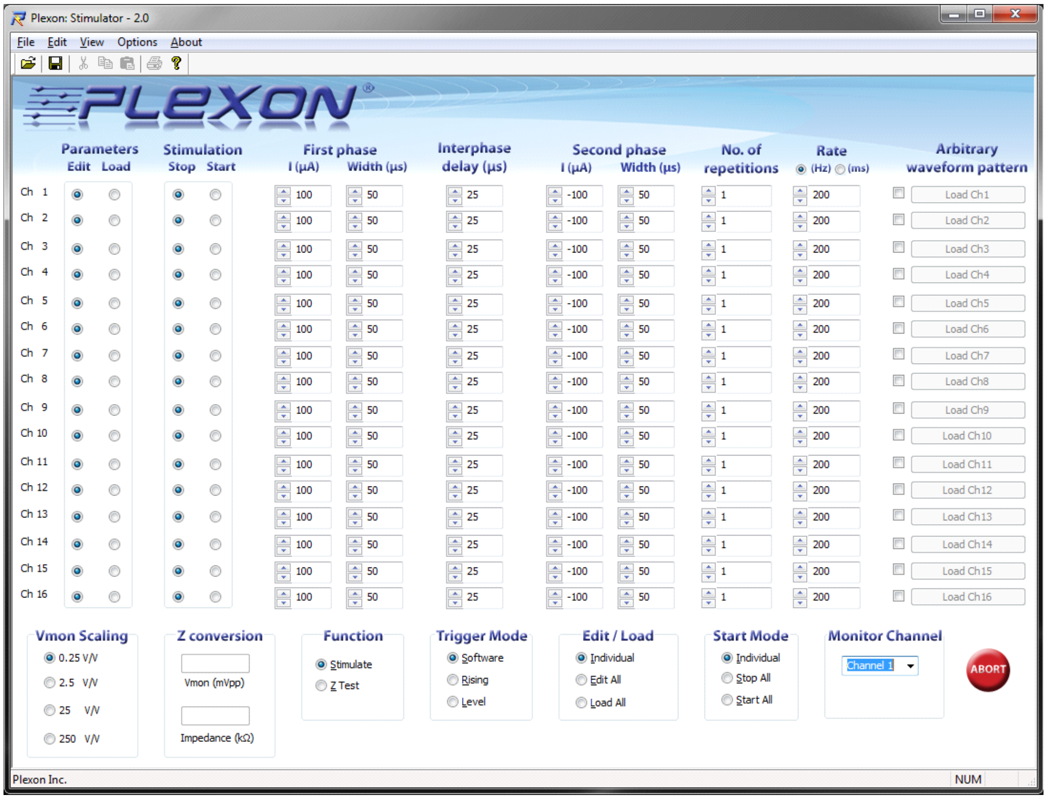
6.1 Overview

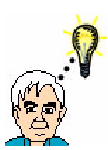
Every stimulation protocol begins with the definition of the stimulation pattern. The PlexStim Electrical Stimulator graphical user interface (GUI) provides a means for defining bi-phasic rectangular pulses and bursts of identical pulses that are repeated at a fixed rate. Once defined, the stimulation pattern is downloaded into the stimulator memory for playback. Playback can be initiated from the GUI or in response to a digital input to the stimulator hardware. During playback, the actual current and voltage delivered to the electrode can be observed on the monitor outputs. Monitoring the electrodes during stimulation is a critical component of successful stimulation.

To monitor the current and voltage waveforms, connect BNC cables between the PlexStim Electrical Stimulator and an oscilloscope.

6.2 Launching the PlexStim Electrical Stimulator Software

Make sure the PlexStim Electrical Stimulator is connected to the computer and turned on. Then launch the PlexStim Electrical Stimulator software by double clicking on the desktop icon . You should see the factory default graphical user interface as shown in the image below:





TIP

Load factory default configuration at any time

You may load the factory default configuration at any time by selecting **Open** from the **File** menu and then selecting the configuration file “Factory_Default.stm”. By default this file is installed in the directory C:\PlexonData\Stim-2\Configuration files.

Note that if the USB cable is not connected to the stimulator or if the stimulator power is turned off when the software is started, an error message will appear. Likewise, if the USB cable is disconnected or the stimulator power is turned off while the software is running, an error message will appear and the program will close:

6.3 Creating Rectangular Pulses

CAUTION

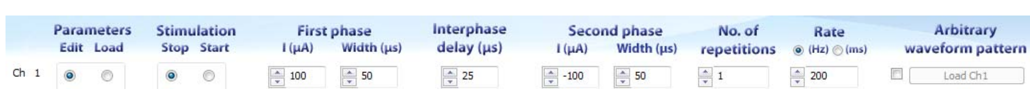
Use ONLY the model electrodes on the test board while you are learning to operate the system.

This section explains how to create and deliver a burst of rectangular pulses to the electrodes. Section 6.4, "Verifying the Output on an Oscilloscope” on page 12 explains how to monitor the actual voltage and current that are delivered.

For each channel there is a row of channel specific controls, and at the bottom of the interface there are additional global controls. (These global controls are described individually in detail in Section 8, "GUI Function Reference” on page 29.) In this section, only the controls required for generating rectangular pulses using the GUI controls are discussed.

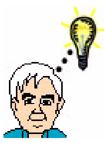
The first step is to configure the stimulation parameters for a single channel. We will start off configuring channel 1 to generate a bi-phasic rectangular pulse that repeats indefinitely at a rate of 200 Hz.

Examine the controls for channel 1:



The default configuration specifies a single biphasic pulse with a short inter-phase delay. The first phase is +100 µA for 50 µs, the inter-phase is 0 µA for 25 µs, and the second phase is –100 µA for 50 µs.

To make it easier to see on the oscilloscope, we want to generate a continuously repeating pulse instead of a single pulse. Locate the “No. of repetitions” control at the right hand side of the row of controls and type “INF” in the control:



TIP
Other methods of changing “No. of repetitions” to INF

If the No. of repetitions control is set to “1” and you press the down arrow, the control will change to “INF”. You can also type “0” in the control and when you click on any other control the 0 will change to INF.

The controls for channel 1 should now look like this:

The next step is to download the stimulation parameters into the stimulator hardware. Click on the **Load** button at the far the left of the row of controls:

Note that the GUI controls for configuring the pulse parameters become grayed out once the channel has been loaded. To change the parameters you must go back to “edit” mode.

Once the parameters for a channel have been loaded you may start the stimulation by clicking on the **Start** button:

The stimulation begins, and because the number of repetitions is set to infinite, the pulses will continue until the **Stop** (or **Edit**) button is pressed.

6.4 Verifying the Output on an Oscilloscope

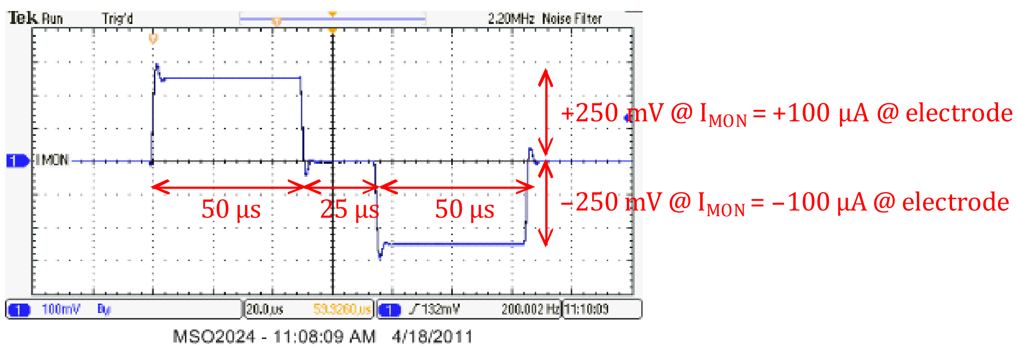
Now examine the current and voltage waveforms on an oscilloscope to see if they make sense. Note that both the current and voltage monitors output a voltage signal and that there is a scaling factor that relates what you see on the monitor output to what is happening at the electrode. The scale factor for the current monitor is 2.5 mV/µA and by default the scale factor for the voltage monitor is 0.25 V/V. That means that if there is a 100 µA current flowing into the electrode from the stimulator, the current monitor will read:

$$100\ \mu A \times\ 2.5\ \frac{mV}{\mu A} = 250\ mV$$

Likewise, if the stimulator was applying 1V to the electrode, the voltage monitor would read:

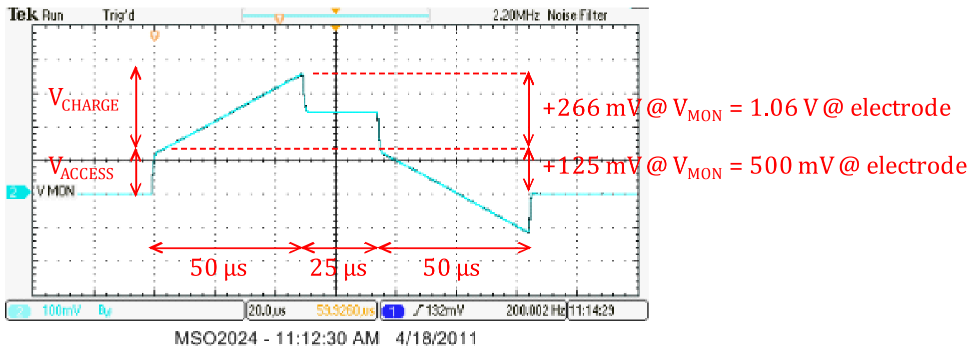
$$1V\ \times\ 0.25\ \frac{V}{V} = 250\ mV$$

The current monitor waveform you see should appear similar to the figure below:



At the beginning of the pulse, the current monitor (I_{MON}) jumps from 0 mV to +250 mV and stays at +250 mV for 50 µs. Then the current monitor drops back to 0 mV for 25 µs and finally drops to −250 mV for 50 µs. Considering the 2.5 mV / µA scaling factor on the current monitor, these 250 mV steps represent current steps of 100 µA as expected from our channel configuration settings. The durations also match our settings.

The output of the voltage monitor is a bit more interesting:



At the beginning of the pulse, the voltage monitor (V_{MON}) jumps from 0 mV to +125 mV, then linearly increases another 266 mV over the 50 µs duration of the first phase. Considering the default scaling of the voltage monitor, the voltage at the electrode initially jumps 500 mV and then increases another 1.06 V over the course of the 50 µs pulse.

The initial jump in the electrode voltage is due to a property of the electrode called “access resistance”. In order to drive a current I into the electrode, we must apply a voltage V to the electrode that satisfies Ohm’s law with respect to the access resistance of the electrode:

$$V_{ACCESS} = I_{ELEC} \times R_{ACCESS}$$

The subsequent rise in electrode voltage during the first phase of the pulse is due to charging of the electrode “capacitance.” By definition, the capacitance of the electrode relates the voltage on the electrode to the amount of charge deposited on the electrode:

$$C = \frac{Q}{V}$$

Therefore, the rate at which the electrode voltage changes during the pulse is directly proportional to the amount of current applied to the electrode and the capacitance of the electrode:

$$\frac{dV}{dt} = \frac{1}{C}\frac{dQ}{dt} = \frac{1}{C} I_{ELEC}$$

The change in electrode voltage during the first phase of the pulse is thus given by:

$$\Delta V = \frac{I_{ELEC} \times \Delta t}{C}$$

The access resistance and capacitance of the model electrodes on the test board are 4.99 kΩ and 4700 pF respectively. Therefore the expected initial voltage jump for a 100 µA current is:

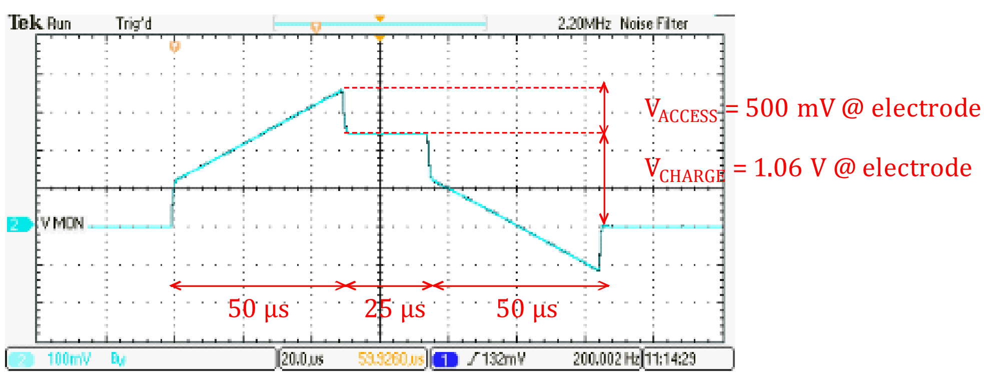
$$V_{ACCESS} = 100\ \mu A\ \times\ 4.99\ kOhm = 499\ \text{mV}$$

The expected voltage rise over the 50 µs duration 100 µA first phase of the pulse is:

$$V_{CHARGE} = \frac{100\ \mu A\ \times\ 50\ \mu s}{4700\ pF} = 1.06\ \text{V}$$

These calculated values are in agreement with the observations from the voltage monitor.

Note also that when the current stops flowing at the end of the first phase the voltage immediately drops by the same amount that it jumped at the beginning of the pulse (V_{ACCESS}). The current during the interphase period is zero. With no current flowing into the electrode, there is no voltage drop across the electrode access resistance. Therefore the voltage during the inter-phase period is the same as V_{CHARGE}. Since there is no current during the inter-phase period the voltage also remains constant during this period.



MSO2024 - 11:12:30 AM 4/18/2011

7 Procedures and Examples

This section provides procedures for operating the stimulator.

Note that you will need an oscilloscope to see the output of the monitor channels.

IMPORTANT:
OBSERVE THE FOLLOWING SEQUENCE DURING ALL EXPERIMENTS:

1. TURN THE STIMULATOR POWER ON
2. LAUNCH THE USER INTERFACE PROGRAM
3. CONNECT THE STIMULATOR TO THE ELECTRODES
4. CONDUCT THE EXPERIMENT
5. CLOSE THE USER INTERFACE PROGRAM
6. DISCONNECT THE STIMULATOR FROM THE ELECTRODES
7. TURN THE STIMULATOR POWER OFF

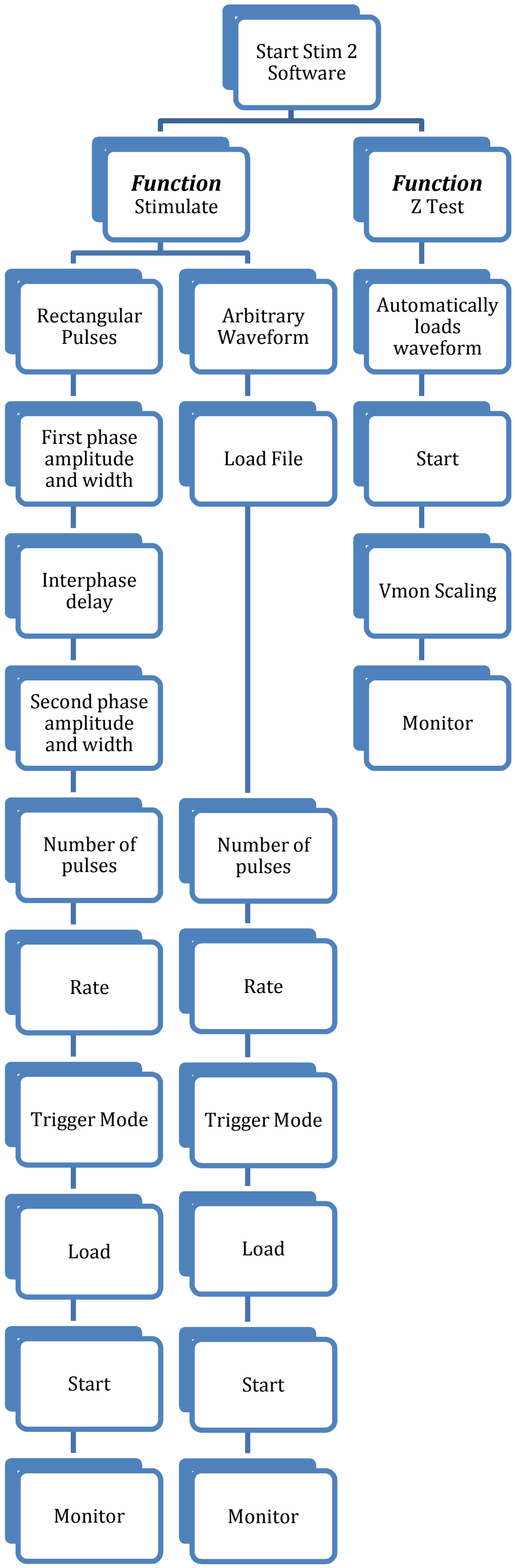
CAUTION

Use ONLY the model electrodes on the test board while you are learning to operate the system.

The flowchart in Section 7.1, "Operational Flowchart” on page 16 summarizes the basic operational steps for using the PlexStim Electrical Stimulator.

Note: If you have not already done so, it is recommended that you review the information and procedures in Section 6, “Getting Started” on page 8.

7.1 Operational Flowchart



7.2 Compliance Voltage and Stimulation Failure

It is important to monitor the electrode voltage during stimulation to verify that the desired stimulation pattern was successfully applied to the electrode. There is a maximum voltage, called the compliance voltage, that the stimulator can output. Once the voltage on the electrode reaches the compliance voltage, the stimulator can no longer drive current into the electrode. The successful delivery of a given stimulation protocol will depend on the amplitude of the current pulse, the duration of the pulse, the stimulator compliance limit, and the properties of the electrode.

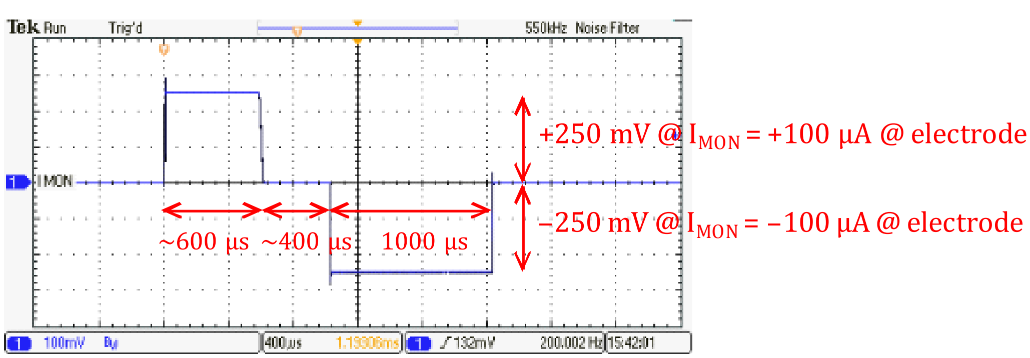
To see this effect, first stop the ongoing stimulation by pressing the **Edit** button for channel 1:

This stops the stimulation and returns the controls for channel to edit mode.

Next, increase the first phase width and second phase width parameters to 1000 µs:

Then **Load** and **Start** channel 1:

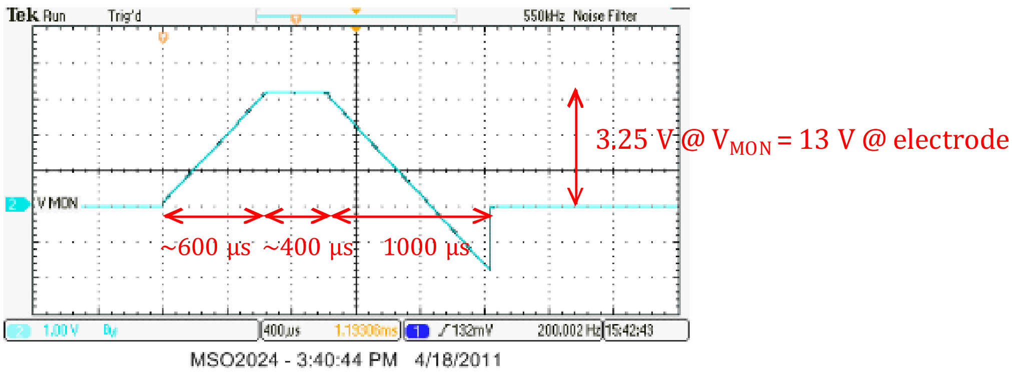
Observe the current monitor output on the oscilloscope:



MSO2024 - 3:40:02 PM 4/18/2011

Although the amplitude of the first phase still looks okay, the duration appears to be too short and the duration of the inter-phase period appears to be too long.

To understand what is happening it is necessary to examine the voltage monitor output:



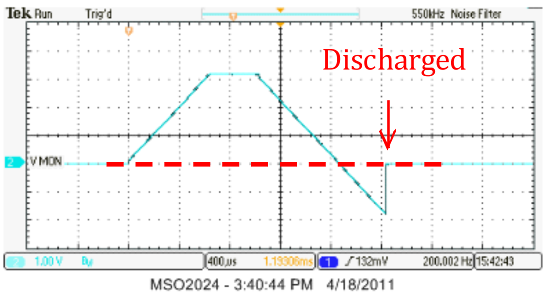
Notice that for the first ~ 600 µs, the voltage is increasing but that in the next ~ 400 µs the voltage has reached a plateau of approximately 13V at the electrode (3.25 V at the monitor channel). At this point, the compliance limit has been reached, and the output current, as seen on the current monitor, drops to zero. The voltage however stays at its maximum value just to maintain the electrode in its charged state.

This example clearly illustrates the importance of monitoring the electrode during stimulation. Monitoring the electrode is necessary to verify that the actual output from the stimulator matches the programmed response.

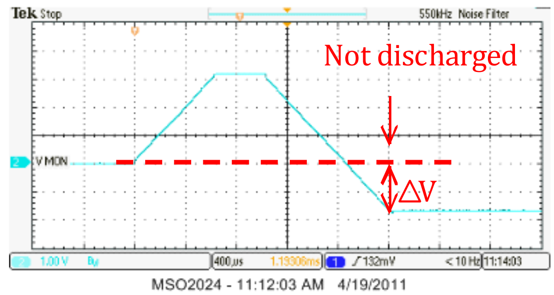
7.3 Automatic Electrode Discharge

Take another look at the current monitor output shown in Section 7.2, "Compliance Voltage and Stimulation Failure" on page 17. As we saw there, the first phase current delivery was cut short because the voltage hit the compliance limit. The amount of charge deposited on the electrode during this phase was less than requested. Now look at the amount of charge removed from the electrode during second phase of the pulse. The amount of charge removed from the electrode during the second phase exceeds the amount of charge deposited during the first phase. Consequently net charge was removed from the electrode during the stimulation pulse. Even though the pulse was designed to be charge balanced the actual delivery of the pulse was not balanced.

Take another look at the voltage monitor output. Notice that it returns to zero after the end of the pulse. Ordinarily, if the pulse delivery was not balanced and charge was left on the electrode then the voltage of the electrode would change after each pulse. The only reason that this does NOT happen in the figures above is that the stimulator has an automatic electrode discharge feature. By default the stimulator automatically discharges the electrode during the inter-pulse interval (in between pulses) and any time the channel is not stimulating (i.e. whenever the channel is in stop or edit mode). As an option for advanced users, the automatic electrode discharge during the inter-pulse interval may be disabled. With this feature turned off, the voltage on the electrode does in fact change after an unbalanced pulse delivery. This is easily seen on the voltage monitor as shown in the figure on the right below.



Automatic electrode discharge turned on (default)



Automatic electrode discharge turned off

Refer to Section 8.14, "Options: Discharge Mode" on page 42 for additional details.

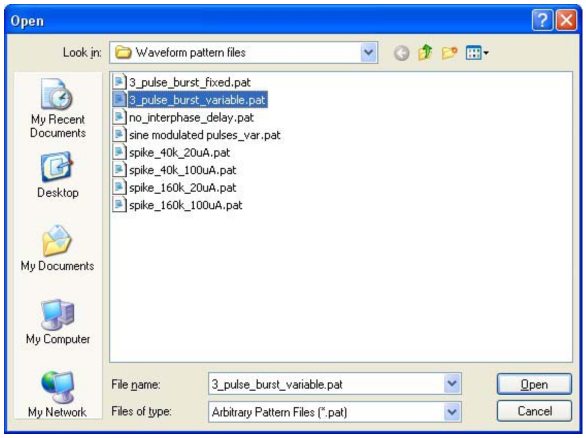
7.4 Arbitrary Waveforms and Complex Rectangular Pulses

The graphical user interface provides simple controls for creating bi-phasic rectangular pulses and bursts of identical pulses. More complicated stimulation patterns can be loaded from user-defined "arbitrary waveform pattern" text files. The arbitrary waveform pattern can be used to create a complex stimulation pattern like an action potential and can also be used to create more complicated patterns of rectangular pulses than can be defined using the front panel GUI controls. The format of the arbitrary waveform text file is defined in Section 8.5, "Arbitrary Waveform Patterns" on page 32.

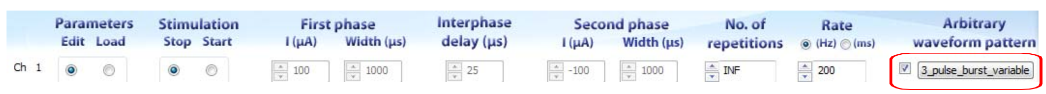
To load an arbitrary waveform for channel 1, make sure the channel is in the **Edit** mode:

Then click on the check box below the **Arbitrary waveform pattern** label:

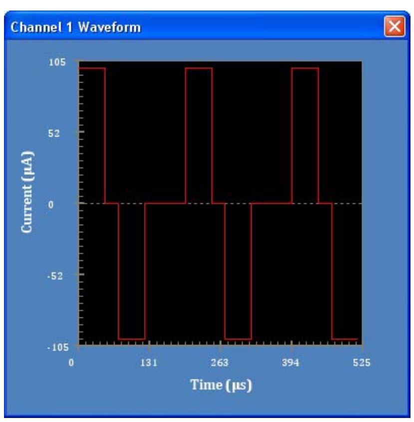
The GUI controls for defining a simple rectangular pulse become grayed out and the **Load Ch1** button becomes active. Click the **Load Ch1** button to open a file selection dialog and then select the file 3_pulse_burst_variable.pat:



The text in the button changes from “Load Ch1” to the name of the file selected:

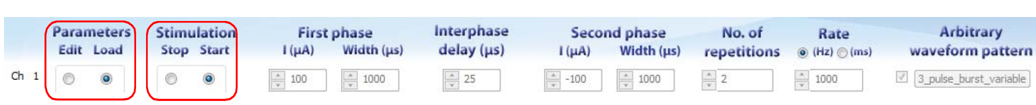


A graphical representation of the waveform is displayed:

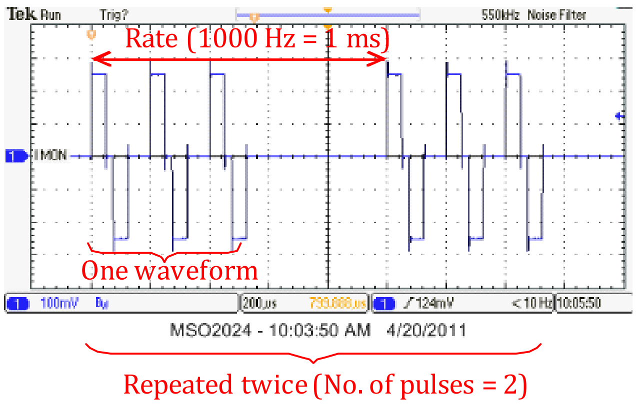


Change the **No. of repetitions** to be 2 and the **Rate** to 1000 Hz:

Then **Load** and **Start** the channel:



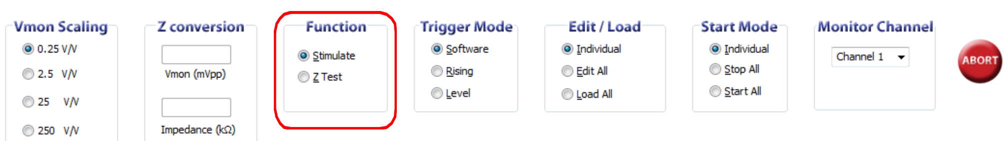
The output of the stimulator is:



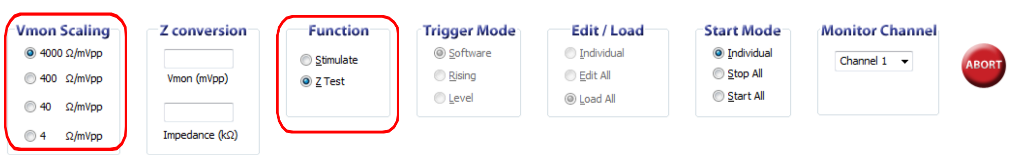
7.5 Impedance Measurement

Impedance measurement is a typical means of characterizing electrodes. Impedance measurement is typically accomplished by applying a very low amplitude sinusoidal current to the electrode and monitoring the resulting voltage developed across the electrode. The stimulator has a special impedance measurement (Z-test mode) that automatically generates a ±100 nA 1 kHz sinusoid for impedance testing.

Locate the **Function** control towards the bottom of the GUI:

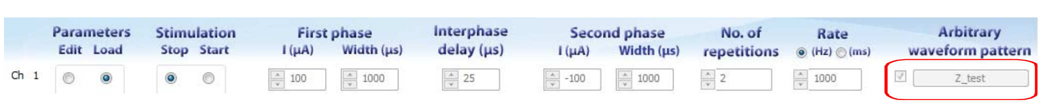


Press the **Z Test** button to put the stimulator in impedance test mode:

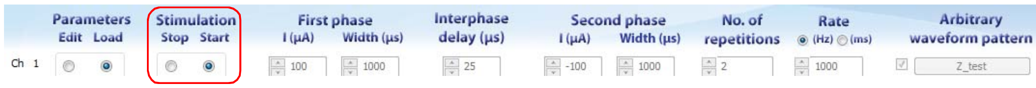


Notice that the Vmon scaling control changes and the default setting is now 4000 Ω/mV_{pp}.

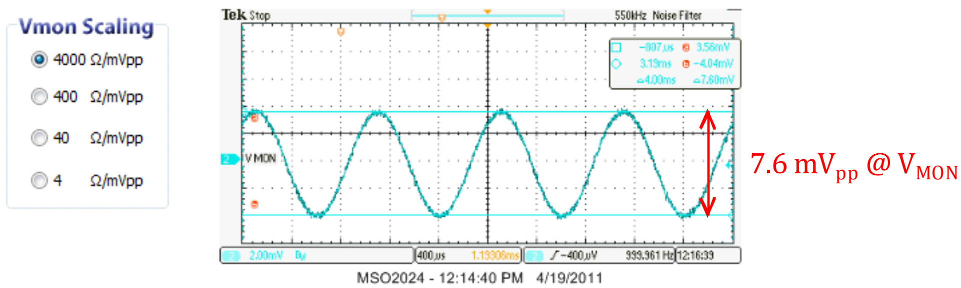
The controls for all channels become grayed out and a pre-defined arbitrary waveform pattern called "Z_test" that codes for a ±100 nA 1 kHz sinusoidal current is automatically loaded for every channel:



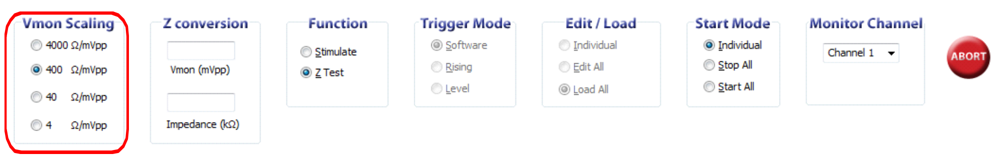
Click the start button to begin generating the sinusoidal current on channel 1:



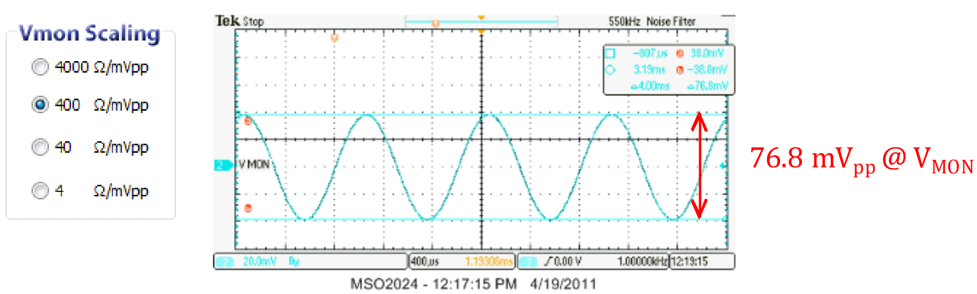
Observe the output of the voltage monitor on an oscilloscope. The voltage across the electrode is a sinusoid with a peak to peak amplitude of ~ 7.60 mV:



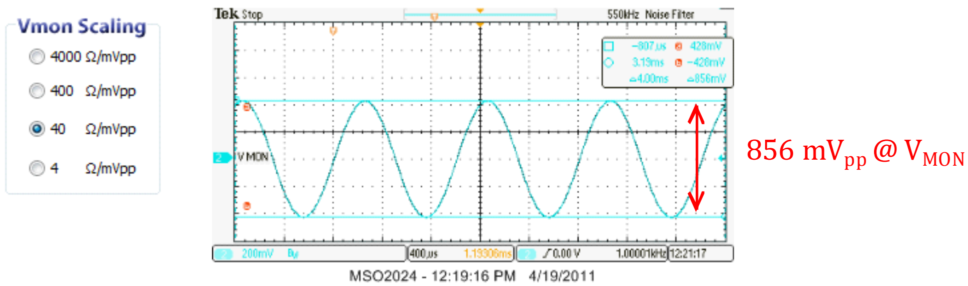
Try changing the Vmon scaling from 4000 Ω/mV_{pp} to 400 Ω/mV_{pp}:



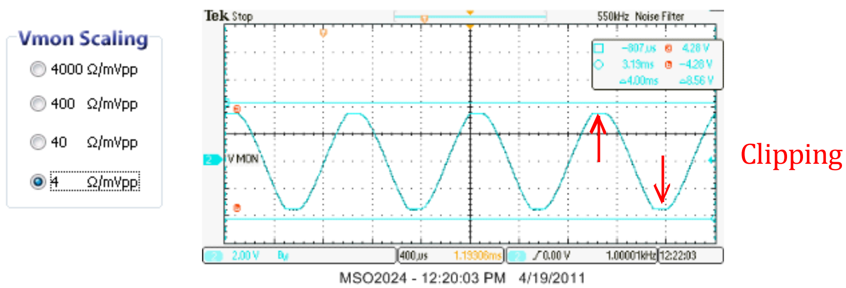
Now the output of the voltage monitor is a sinusoid with a peak to peak amplitude of ~ 76.8 mV:



Continue to change the resolution of the monitor channel to obtain the largest amplitude sinusoid on the monitor channel that is not clipping. Try 40 Ω/mV$_{pp}$:



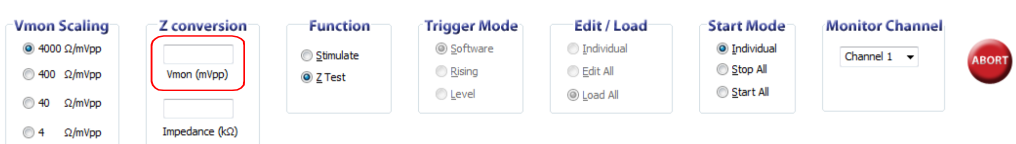
When the Vmon scaling is changed to 4 Ω/mV$_{pp}$ the output of the monitor channel starts to clip:



To calculate the impedance of the electrode, multiply the peak to peak amplitude observed on the voltage monitor by the Vmon scaling factor. In this case the peak to peak amplitude signal on the voltage monitor is 856 mV$_{pp}$ and the Vmon scaling is 40 Ω/mV$_{pp}$ so the impedance of the electrode is:

$$Z_{ELEC} = 856\ mV_{pp} \times \frac{40\ Ohm}{mV_{pp}} = 34.2\ \text{kOhm @ 1 kHz}$$

This calculation can be performed by the software if you enter the peak to peak amplitude (in millivolts) in the Vmon field under **Z conversion**.



To obtain the most accurate impedance reading, you should adjust the scale factor as described above to obtain the largest possible signal on the voltage monitor that does not clip.

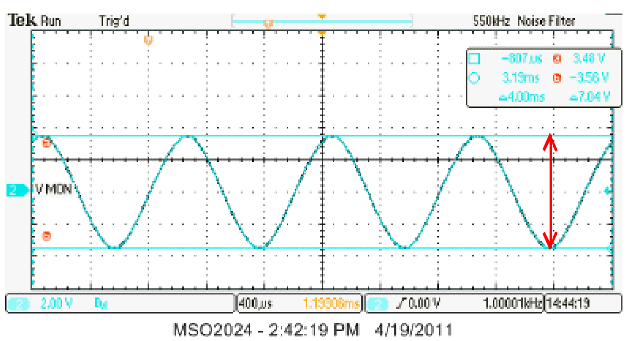
The model electrodes on the test board have a resistance of 4.99 kΩ in series with a 4700 pF capacitor. The expected impedance for this combination at 1 kHz is 34.2 kΩ in agreement with the measurement above.

7.6 Working with Real Electrodes

In a typical experimental situation, the properties of the electrode may not be well known and may vary over time. Characterizing the electrode in saline prior to using it in vivo can provide valuable information about the condition of the electrode. For example, measuring the impedance of the electrode before and after implantation can help determine if the electrode was physically damaged during the implantation process. Likewise estimating the impedance, access resistance, and capacitance of the electrode periodically can provide clues as to the stability of the electrode tissue interface over time.

This section shows some measurements obtained from a blunt cut 25 µm diameter platinum iridium micro-wire electrode in saline. All of the measurements are taken using the same procedures outlined in the preceding sections that were used with the model electrodes on the test board. If you have not already done so, practice using the test board before trying to work with electrodes in saline or trying to work with implanted electrodes.

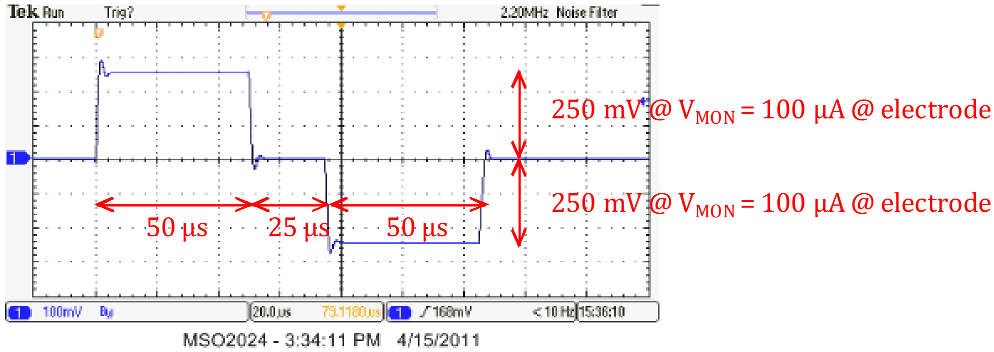
First measure the impedance of the electrode. With the Vmon scaling set to 40 Ω/mV$_{pp}$, the resulting voltage monitor output was 7.04 V$_{pp}$ implying an impedance of 282 kΩ:



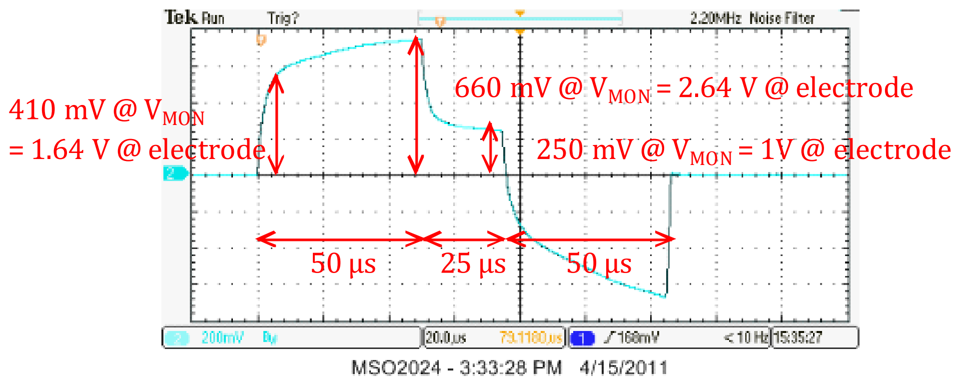
7040 mV$_{pp}$ @ V$_{MON}$ = 282 kOhm

Next observe the response of the electrode to the default 100 µA 50 µs stimulation pulse.

Current monitor:



Voltage monitor:



The current monitor output looks normal and the voltage remains well below the compliance limit. The requested stimulation was successfully delivered.

Similar to the model electrodes on the load board, the voltage of the real electrode shows a quick jump at the onset of the current followed by a more gradual increase over the duration of the first phase. Although these are not as distinct as they were with the model electrode, we can still interpret them in terms of the access resistance and capacitance of the electrode.

An initial jump of 1.64V in response to a current of 100 µA implies an access resistance of:

$$R_{ACCESS} = \frac{V}{I} = \frac{1.64\,V}{100\,\mu A} = 16.4\ \text{k}\Omega$$

And the subsequent increase in electrode voltage from 1.64 V to 2.64 V over the course of the 50 µs pulse implies a capacitance of:

$$C_{ELEC} = \frac{I \times \Delta t}{\Delta V} = \frac{100\ \mu A \times 50\ \mu s}{1\ V} = 5\ \text{nF}$$

It is tempting to try and relate the impedance of the electrode at 1 kHz to the stimulation properties of the electrode, but this can be problematic. Electrode impedance is typically measured at 1 kHz with extremely low currents while stimulation is typically carried out with constant (DC) current of a much larger amplitude. The properties of an electrode in saline are more correctly the properties of the electrode electrolyte interface and those properties can vary with frequency, applied voltage, time, and other factors.

For example, you can estimate the equivalent capacitance of the electrode from the impedance measurement. The impedance of a capacitor C at a frequency f is given by:

$$/Z/= \frac{1}{2\pi fC}$$

So the estimated capacitance of an electrode with impedance of 282 kΩ at 1 KHz is:

$$C_{est} = \frac{1}{2\pi f/Z/} = \frac{1}{2\pi(1\ \text{kHz})(282\ \text{k}\Omega)} = 0.56\ \text{nF}$$

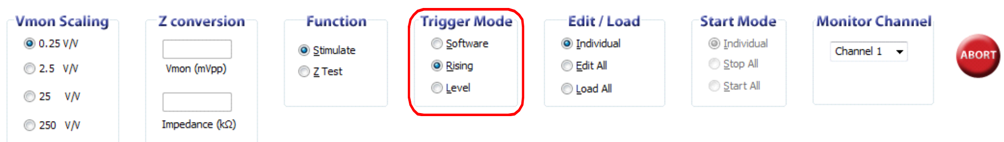
Note that the capacitance estimated from the impedance measurement is nearly an order of magnitude smaller than the capacitance of the electrode estimated from a constant (DC) current pulse.

7.7 Starting Stimulation from a Digital Input

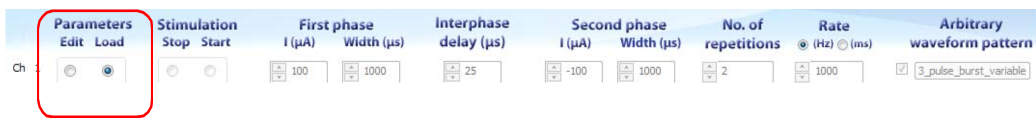
To start stimulation from a digital input first configure the pulse parameters or load an arbitrary waveform for the channel or channels you want to stimulate. Next, find the **Trigger Mode** control at the bottom of the screen. There are two digital input trigger modes, **Rising** and **Level**. In the Rising trigger mode, stimulation begins when the digital input for the channel transitions from low (~0V) to high (~5V). In the Level trigger mode, stimulation also begins when the digital input transitions from low to high, but in Level trigger mode, if the digital input is still high when the stimulation protocol completes, the stimulation protocol will begin again. Note that once the stimulation pattern is triggered by the digital input, it will play to completion even if the digital input goes low. This helps guard against unbalanced charge delivery.

Configure channel 1 using the “3_pulse_burst_variable” arbitrary pattern and set the **No. of repetitions** to 2 and the **Rate** to 1000 Hz:

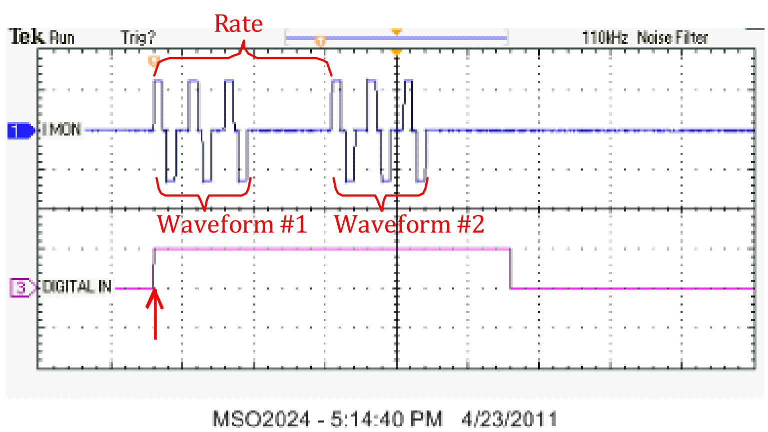
Now select the **Rising** Trigger Mode:



Download the stimulation parameters to the stimulator by selecting **Load**:

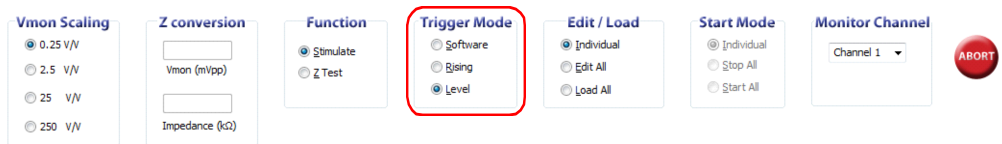


The stimulator is now armed and when the digital input for channel 1 goes high, the stimulation pattern consisting of two repeats of the “3 pulse burst” waveform starts and plays to completion:

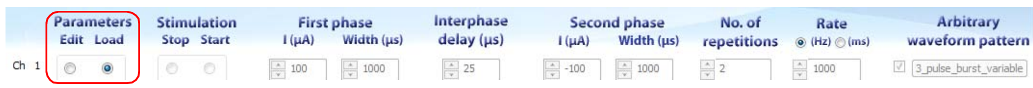


Compare that with the output in **Level** trigger mode. To switch to **Level** trigger mode, you must first change channel 1 to **Edit** mode:

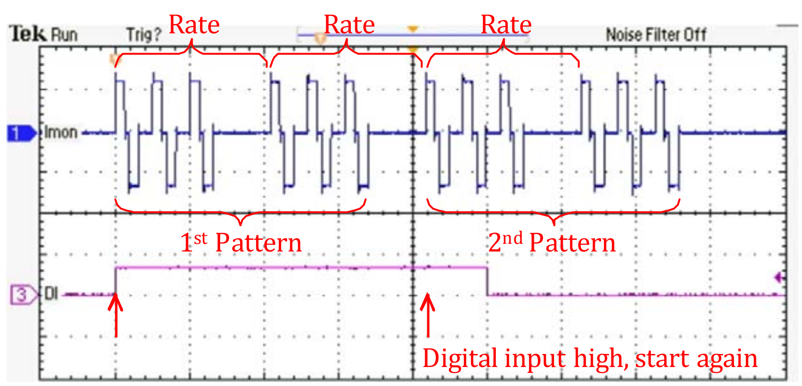
Then switch to **Level** trigger mode:



Finally load channel 1 again:



The stimulator is now armed and when the digital input for channel 1 goes high, the stimulation pattern starts and plays to completion:



In **Level** trigger mode however, if the digital input is high after the end of the first stimulation pattern, the stimulation pattern will play again in such a way that the frequency of the waveforms is maintained.

7.8 Stopping Stimulation

Most stimulation protocols are of finite duration and once they have been started will run to completion and then stop automatically. Some stimulation protocols however may be defined with very long intervals or with a very large or even infinite number of repetitions. It may become desirable to stop these protocols before completion. An ongoing stimulation protocol can be stopped by pressing the **Stop**, **Edit**, **Stop All**, **Edit All**, or **Abort** controls, or by closing the user interface. Pressing the **Stop** or **Edit** control for a particular channel causes the stimulation in progress on that channel to stop. Pressing the **Stop All**, **Edit All**, or **Abort** controls, or closing the user interface causes stimulation to stop on all channels.

If there is a pulse or waveform in progress when a **Stop**, **Edit**, **Stop All,** or **Edit All** command is issued, the stimulator will allow that pulse or waveform to complete before stopping the stimulation. Most rectangular pulse and arbitrary waveforms are designed to be charge balanced. That is, they are designed such that the net charge deposited on the electrode over the course of the pulse or waveform is zero. By allowing the pulse or waveform that is in progress to complete, the stop mechanism helps to preserve that charge balance.

Pressing the **Abort** button or closing the user interface causes the stimulation to stop immediately even if there is a pulse or arbitrary waveform playback in progress. Even though this temporarily results in an unbalanced stimulation, the automatic electrode discharge feature (see Section 7.3, "Automatic Electrode Discharge" on page 18) prevents long term charge accumulation on the electrode.

Refer to Section 8.12, "ABORT" on page 40 for some examples.

8 GUI Function Reference

8.1 Parameters Edit/Load

Note: The **Edit/Load** controls may be grayed out if the **Edit All** or **Load All** control is selected.

8.1.1 Edit

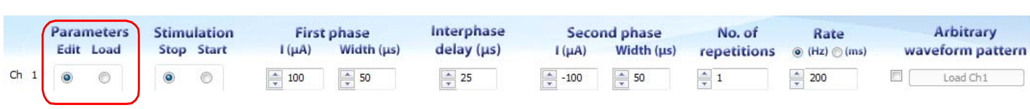
The stimulation parameters for a channel can only be changed when the channel is in edit mode. Press the **Edit** button to put the channel in edit mode.

Pressing the **Edit** button will cause any stimulation pattern that is in progress on the channel to stop.

8.1.2 Load

Once the stimulation parameters for a channel have been configured, press the **Load** button to download the configuration to the stimulator hardware. Once the stimulation parameters have been downloaded to the stimulator, the channel configuration controls become grayed out.

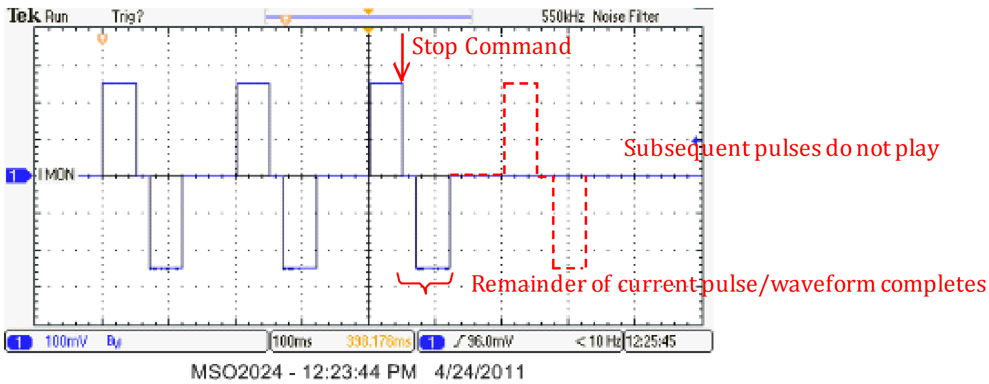
8.2 Stimulation Stop/Start



Note: The **Start/Stop** controls may be grayed out if the **Stop All** or **Start All** control is selected.

8.2.1 Stop

Press the **Stop** button to stop a stimulation pattern that is in progress. When **Stop** is clicked in the middle of a pulse or arbitrary waveform output, the remainder of that pulse or arbitrary waveform will play to completion. This helps preserve charge balance in the stimulation output. Any remaining pulses in the pattern will not be played.

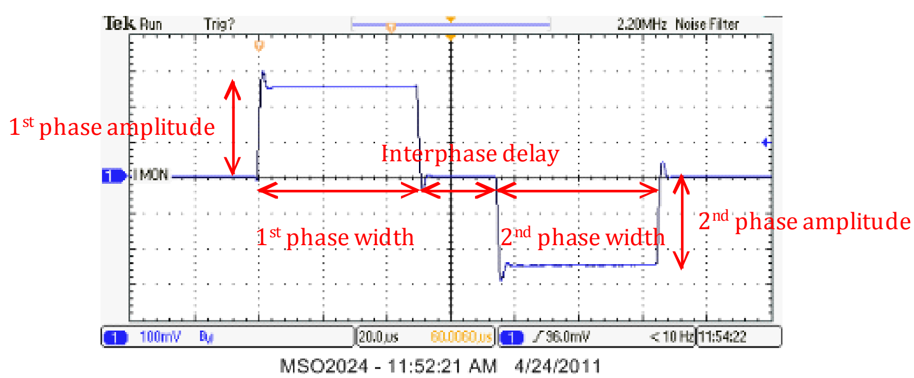


8.2.2 Start

Press the **Start** button to initiate playback of the stimulation pattern for the channel when the trigger mode is set to software. The stimulation parameters for the channel must be loaded before the start button can work.

8.3 Rectangular Pulse Parameters

The graphical user interface provides controls for defining the amplitude and duration parameters of a bi-phasic rectangular pulse:



8.3.1 First Phase Amplitude

The first phase amplitude may range from +1000 µA to -1000 µA in steps of 1 µA.

8.3.2 First Phase Width

The first phase width may range from 5 µs to 65535 µs in steps of 1 µs.

8.3.3 Interphase Delay

The interphase delay is the time between the first and second phases. The interphase delay may range from 5 µs to 65535µs in steps of 1 µs. The current output is zero during the interphase delay.

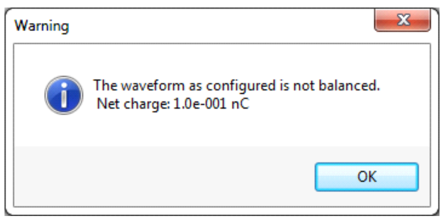
8.3.4 Second Phase Amplitude

The second phase amplitude may range from +1000 µA to -1000 µA in steps of 1 µA.

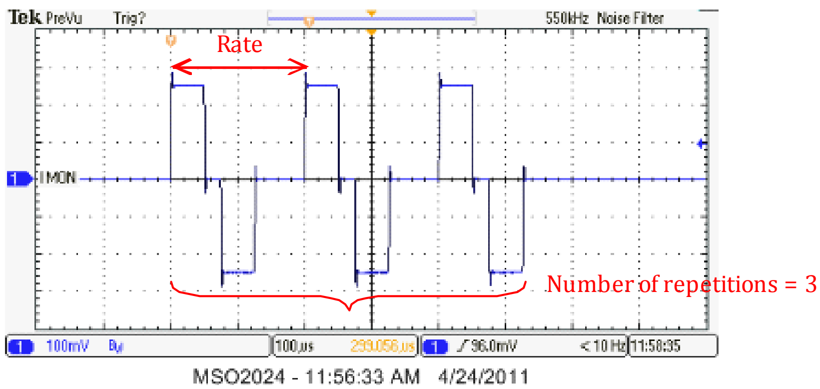
8.3.5 Second Phase Width

The second phase width may range from 5 µs to 65535 µs in steps of 1 µs.

If the charge delivered in the second phase is not equal and opposite to the charge delivered in the first phase, then a warning will pop up when you attempt to load the stimulation parameters. The net charge imbalance per pulse is also given in the warning.



8.4 Number of Repetitions and Pulse Rate



8.4.1 No. of Repetitions

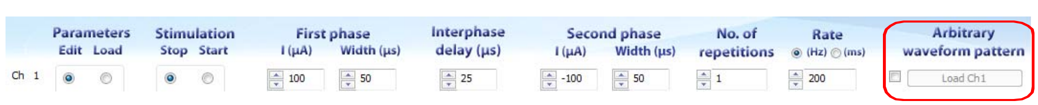
The number of repetitions is simply the number of times that the bi-phasic pulse (defined using the GUI controls) or the arbitrary waveform (loaded from a text file) is repeated. The number of repetitions can range from 1 to 32767. An infinite number of repetitions can also be generated by typing “0” or “INF” in the control. The defined number of pulses or waveforms will be generated every time the channel is started.

8.4.2 Rate

The rate is the frequency at which the defined pulse or arbitrary waveform is repeated, expressed in Hertz, or the time between the start of one pulse or arbitrary waveform and the start of the next pulse or arbitrary waveform, expressed in milliseconds. Use the radio buttons to select whether to express the rate in Hertz or in milliseconds. The maximum time between pulses is 125000 ms corresponding to a minimum rate of 0.008 Hz. The minimum time between pulses generally depends on the length of the pulse or arbitrary waveform that is being repeated. The time between pulses must include the duration of the pulse and at least 5 µs between pulses. For example a pulse that lasts for 200 µs must be repeated at a rate less than 4878 Hz since 1/(200 µs + 5 µs) ~ 4878 Hz. The rate parameter may be entered in steps of 0.001 ms (1 µs).

8.5 Arbitrary Waveform Patterns

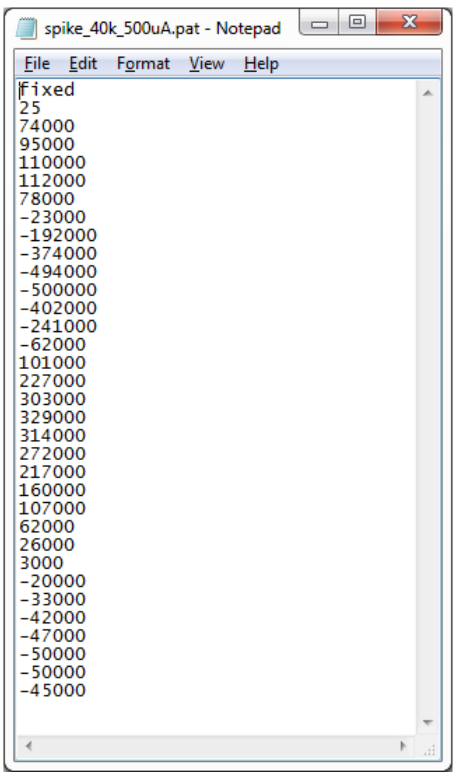
This section describes the arbitrary waveform pattern file formats (fixed sampling rate and variable sampling rate) and explains where to find several examples of these files.



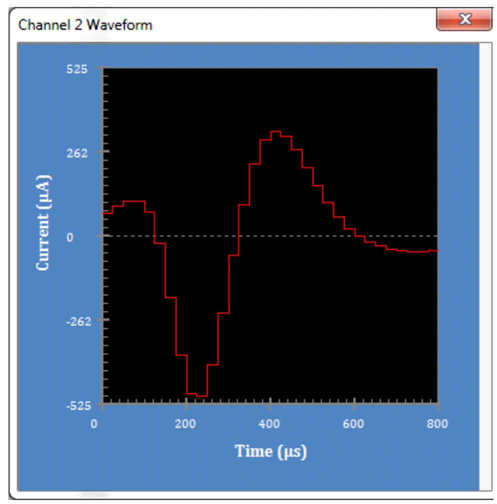
Arbitrary waveform patterns are stimulation patterns that are defined in text files with a .pat extension. Check the box at the far right of the row of controls next to where it says arbitrary waveform pattern to disable the rectangular pulse controls and enable the arbitrary waveform pattern load button. Click on the Load Ch N button to select and open an arbitrary waveform file. When the file is opened, a graphical representation of the arbitrary waveform is displayed.

8.5.1 Fixed Sampling Rate

In the fixed sampling rate file format, the first line of the file contains the keyword “fixed”. The second line of the file codes for a single duration parameter expressed in microseconds. The subsequent lines of the file contain a series of current amplitude values expressed in nanoamps. When the stimulation pattern is played back, each current amplitude from the file is played for the fixed amount of time specified by the duration parameter. The fixed sampling rate format is especially useful for coding continuously varying shapes such as the action potential waveform.



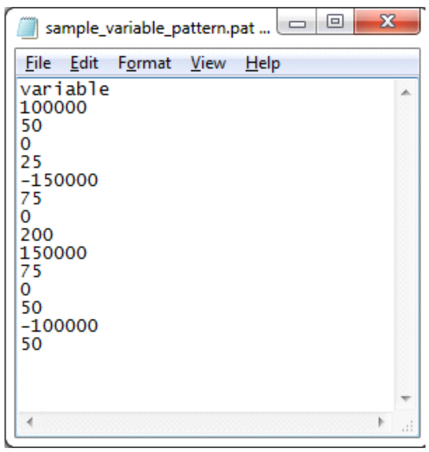

```
fixed
25
74000
95000
110000
112000
78000
-23000
-192000
-374000
-494000
-500000
-402000
-241000
-62000
101000
227000
303000
329000
314000
272000
217000
160000
107000
62000
26000
3000
-20000
-33000
-42000
-47000
-50000
-50000
-45000
```

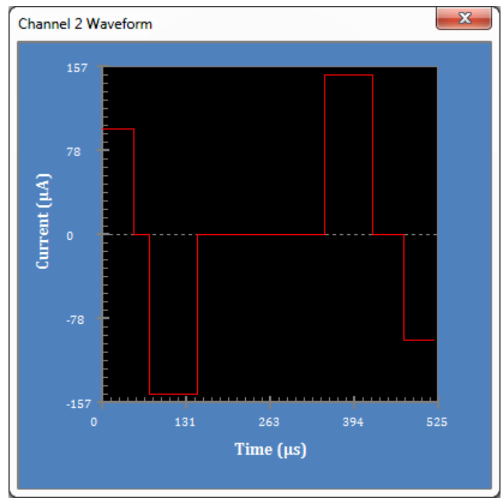


The file can contain up to 999 amplitude values. The duration parameter can range from 1 µs to 65535 µs. The current amplitudes can range from -1,000,000 nA to +1,000,000 nA in steps of 1 nA. Note however that the actual resolution of the stimulator output in stimulation mode is ~ 30 nA and all current amplitudes will be rounded to the nearest possible output value. For example an amplitude value of 15 nA will be rounded down to 0 nA and an amplitude value of 16 nA will be rounded up to 30 nA.

8.5.2 Variable Sampling Rate

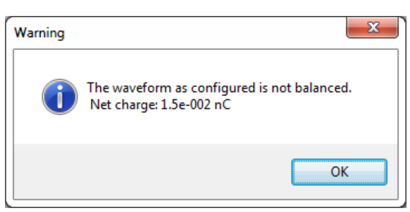
In the variable sampling rate file format, the first line of the file contains the keyword “variable”. The second line of the file codes for a current amplitude and the third line of the file codes for the duration that the amplitude on line two should be played Subsequent lines in the file code for additional amplitude duration pairs. When the stimulation pattern is played back, each current amplitude is played for the specific duration associated with that amplitude. The variable sampling rate format is especially useful for coding complex patterns of rectangular pulses.





The file can contain up to 499 amplitude duration pairs. The duration parameter can range from 1 µs to 65535 µs. The current amplitudes can range from -1,000,000 nA to +1,000, 000 nA in steps of 1 nA. Note however that the actual resolution of the stimulator output in stimulation mode is ~ 30 nA and all current amplitudes will be rounded to the nearest possible output value. For example an amplitude value of 15 nA will be rounded down to 0 nA and an amplitude value of 16 nA will be rounded up to 30 nA.

As with rectangular pulses, the arbitrary pattern should be charge balanced, meaning that the same amount of current is deposited and withdrawn from the electrode. The Stim2 software will automatically analyze the loaded file and give a warning if the net charge is not zero.



8.5.3 Example Arbitrary Waveform Pattern Files

Several example arbitrary waveform files are installed with the PlexStim Stimulator software. These files were designed to illustrate how to use arbitrary waveform files and some of the things you can accomplish using arbitrary waveform files. Each file is described briefly below. By default these files are installed in the directory “C:\PlexonData\Stim-2\Waveform pattern files.” The files can be opened with any text editor (e.g. Notepad).

“3_pulse_burst_fixed.pat” and “3_pulse_burst_variable.pat” both code for an identical sequence of three pulses. These files illustrate the two different arbitrary waveform file formats. One file is coded using the fixed sampling rate format described in Section 8.5.1, "Fixed Sampling Rate” on page 33 and the other is coded using the variable sampling rate format described in Section 8.5.2, "Variable Sampling Rate” on page 34. Creating an arbitrary waveform file containing multiple rectangular pulses is useful for creating complex temporal patterns of rectangular pulses that cannot be created using the GUI controls alone. For instance, a sequence consisting of bursts separated by idle periods can be created by defining the burst in an arbitrary waveform pattern file and then using the GUI controls to repeat the burst with the desired amount of time in between bursts. See Section 7.4, "Arbitrary Waveforms and Complex Rectangular Pulses” on page 19 for an example of such a waveform created using the “3_pulse_burst_variable.pat” arbitrary waveform pattern file.

When using the GUI controls to define rectangular pulses, the minimum interphase delay (the time between the two pulse phases) that can be entered is 5 µs. You can however define a pulse with zero interphase delay by using an arbitrary waveform file. This is illustrated in the file “no_interphase_delay.pat”.

The file “sine modulated pulses_var.pat” contains a sequence of pulses delivered at 150 Hz whose amplitudes are modulated by a 5 Hz sinusoidal envelope. Such amplitude modulated stimulation patterns can be used to mimic theta rhythm and other oscillatory brain patterns.

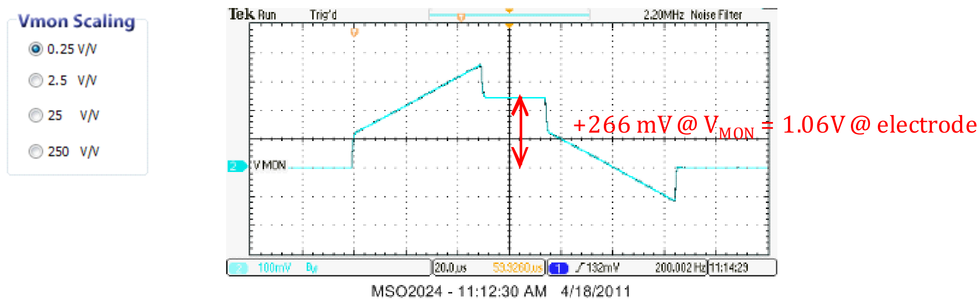
The files “spike_40k_20uA.pat” and “spike_40k_100uA.pat” are actual extracellular spike waveforms recorded with an acquisition system that was sampling at 40 kHz. These files illustrate the extreme flexibility of arbitrary waveform patterns. The two files represent two different scaling of the original extracellular voltage recording into current. In one the minimum voltage of the action potential was scaled to 20 µA and in the other it was scaled to 100 µA.

The files “spike_160k_20uA.pat” and “spike_160k_100uA.pat” illustrate the ability of the stimulator to play back waveforms at very high sampling rates. These files are based on the same data as their “40k” counterparts described above, but the original 40 kHz spike waveform has been interpolated and up-sampled to 167 kHz. This results in a noticeably smoother waveform on playback.

8.6 Vmon Scaling and Z Conversion

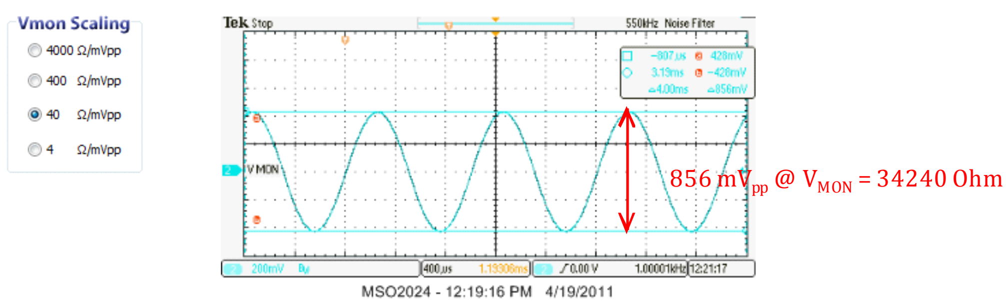
The Voltage monitor scaling control indicates the relationship between the voltage at the monitor channel and the voltage at the electrode. The behavior of the control depends on whether the stimulator is in stimulation mode or impedance test mode as described below.

8.6.1 Vmon Scaling (Stimulation Mode)



When the stimulator function is set to stimulate the voltage monitor scaling is expressed as the number of volts at the monitor channel output for each volt at the electrode. By default the scaling is 0.25 V/V meaning that a 1 V signal at the electrode will appear as 0.25 V on the monitor channel. This setting is appropriate for most electrodes. Adjusting the Vmon scaling might only be necessary to help resolve small voltages when stimulating with extremely low impedance electrodes. Divide the voltage observed on the monitor output by the scale factor to determine the voltage at the electrode. Note that if the Vmon scaling is set higher than 0.25V/V, then the output of the voltage monitor will saturate before the compliance limit is reached and the may not reflect the actual voltage at the electrode. See Section 9.7, "Voltage Monitor” on page 47 for additional details.

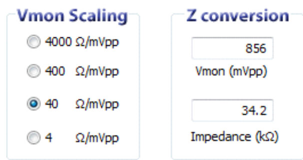
8.6.2 Vmon Scaling (Z Test Mode)



When the stimulator function is set to impedance test (Z test) a 1 kHz ±100 nA sinusoidal current is applied to the electrode and the resulting peak to peak voltage elicited across the electrode indicates the impedance of the electrode. Adjust the Vmon scaling control to obtain the largest peak to peak signal on the monitor channel that is not clipping. Measure the peak to peak amplitude of the signal on the voltage monitor in millivolts and multiply the reading by the scale factor to obtain the electrode impedance in Ohms:

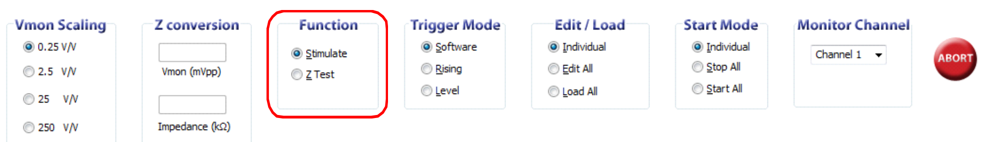
$$856\ mV_{pp}\ x\ \frac{40\Omega}{mV_{pp}} = 34240\ \Omega$$

8.6.3 Z Conversion



The **Z conversion** control calculates the electrode impedance from the voltage measurement you provide. Enter the peak to peak voltage expressed in millivolts that you observed on the voltage monitor output. The control calculates and displays the corresponding electrode impedance expressed in kΩ. The voltage monitor scale factor is handled automatically for you.

8.7 Function Stimulate / Function Z Test



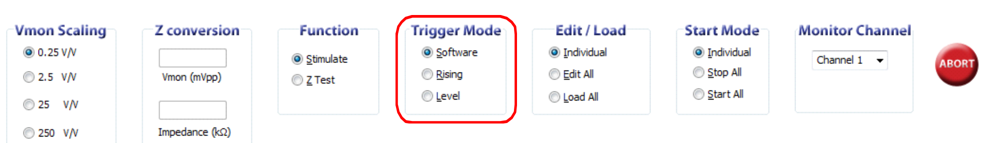
8.7.1 Stimulate Mode

Select stimulate mode for rectangular pulse or arbitrary waveform stimulation.

8.7.2 Impedance Test Mode

Select impedance test (Z Test) mode to perform impedance testing on electrodes. In impedance test mode, a 1 kHz ±100 nA sinusoidal stimulation pattern is automatically selected and loaded for each channel. Start stimulation on a channel and observe the voltage monitor output for that channel to determine the impedance of the electrode. See Section 7.5, "Impedance Measurement" on page 21 for an example.

8.8 Trigger Mode



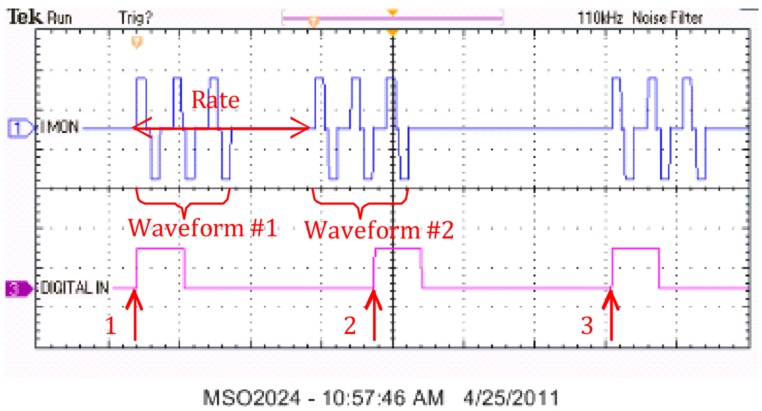
The trigger mode determines how stimulation is initiated. The three **Trigger Mode** options, **Software**, **Rising**, and **Level** are described below. In all cases, the stimulation pattern for the channel must be defined and loaded before it can be triggered.

8.8.1 Software Trigger Mode

In **Software** mode, stimulation is initiated by clicking on the **Start** or **Start All** button in the GUI. Each time **Start** or **Start All** is pressed, the stimulation pattern for the channel is initiated.

8.8.2 Rising Edge Trigger Mode

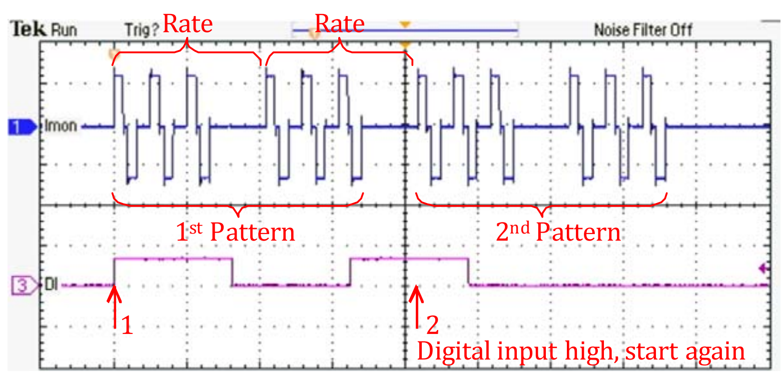
In the **Rising** trigger mode, stimulation begins when the digital input for the channel transitions from low (~0V) to high (~5V). The latency from digital input to stimulation onset is about 2 µs.



In the example above, the stimulation pattern consists of two repetitions of a 3-pulse burst arbitrary waveform. The stimulation pattern is initiated by the rising edge in the digital input marked 1. Once the stimulation pattern is triggered it will play to completion even if the digital input goes low during the stimulation. If another rising edge occurs during the stimulation pattern it is ignored (e.g. the rising edge marked 2 is ignored).

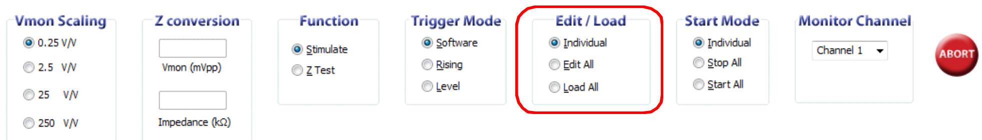
8.8.3 Level Trigger Mode

In the **Level** trigger mode, stimulation also begins when the digital input for the channel transitions from low (~0V) to high (~5V), but in level trigger mode, the digital input is checked again after the end of the stimulation pattern. If the digital input is high after the end of the first stimulation pattern, the pattern will play again in such a way that the frequency of the waveforms is maintained.



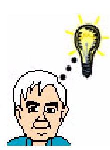
Note that the digital input does not have to stay high, it only has to be high at the appropriate time for the stimulation pattern to repeat. In the example above, the stimulation pattern consists of two repetitions of the “3 pulse burst” arbitrary waveform. The stimulation pattern is initiated by the rising edge in the digital input marked 1. Because the digital input is high after the first stimulation pattern (at the point marked 2) the stimulation pattern starts playing a second time. The fact that the digital input went low and then high again during the first pattern has no effect on the output.

8.9 Edit/Load All



When the **Edit/Load** control is in Individual mode, the mode of each channel can be configured using the individual Edit and Load controls in the row of controls dedicated to that channel. Pressing the **Edit All** button causes all channels to switch to parameter editing mode. Note that pressing **Edit All** will cause any stimulation pattern in progress on any channel to stop. Pressing the **Load All** button will cause the stimulation parameters for all channels to be downloaded into the stimulator.

Pressing **Edit All** or **Load All** will also cause individual channel edit and load controls to be grayed out. Press the **Individual** button to configure the mode of each channel independently of the others.

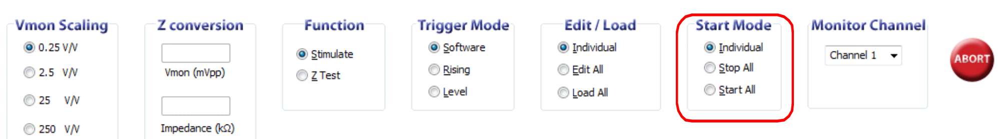


TIP

Resetting Edit/Load control

You probably want to return the **Edit/Load** control to **Individual** after pressing **Edit All** or **Load All**, and there is no reason not to do so.

8.10 Start Mode



When the **Start Mode** control is in Individual mode, each channel and can be started or stopped using the individual Start and Stop controls in the row of controls dedicated to that channel. Refer to Section 8.2, "Stimulation Stop/Start” on page 29 for a description of Stop and Start.

Pressing **Stop All** is a convenient way to stop the stimulation in progress on every channel. Pressing **Start All** is a convenient way to start the stimulation protocol on every channel. Pressing **Start All** is also the only way to start stimulation on multiple channels simultaneously from the GUI. Complex multichannel stimulation protocols can be initiated using the **Start All** function. First define and load the appropriate stimulation pattern for the channels you wish to stimulate and then initiate the multichannel stimulation pattern by pressing the **Start All** button.

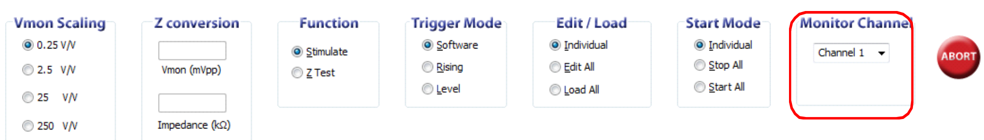
Note that pressing **Stop All** or **Start All** causes the individual channel **Stop** and **Start** controls to be grayed out. Press the **Individual** button to start or stop channels independently.

TIP

Resetting Start Mode

You probably want to return the **Start Mode** to **Individual** after pressing **Stop All** or **Start All**, and there is no reason not to do so.

8.11 Monitor Channel

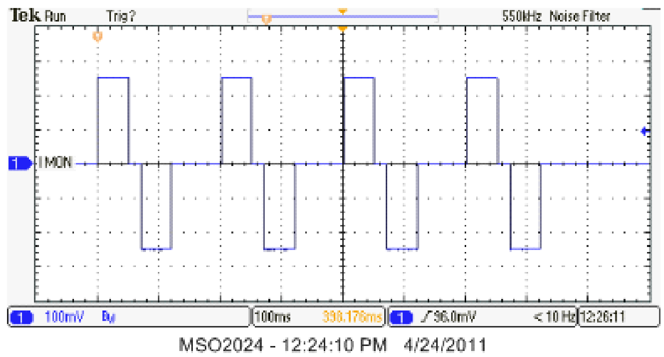


Use the dropdown menu to select one channel for display on the voltage and current monitor outputs. Only one channel may be monitored a time. Connect the current and voltage monitor outputs to an oscilloscope using standard BNC cables to see the stimulation pattern that is being applied to the selected electrode.

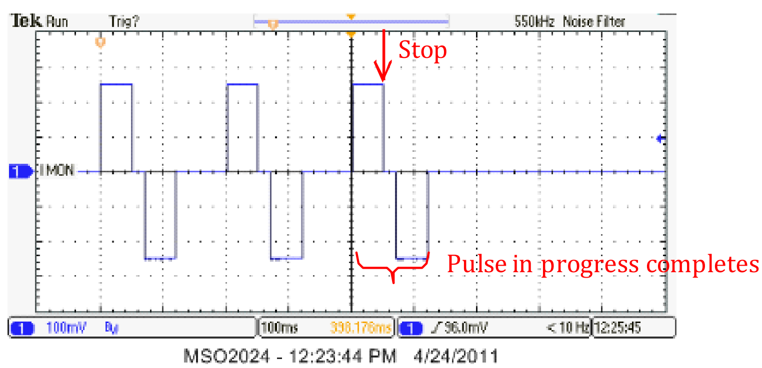
8.12 ABORT

Pressing the red **ABORT** button will cause all stimulation to stop immediately even if there is a pulse or arbitrary waveform in progress. This is in contrast to stopping stimulation by pressing **Stop**, **Stop All**, **Edit** or **Edit All**, in which a pulse or arbitrary waveform that is in progress will play to completion. Pressing **ABORT** also causes all channels to return to the edit mode. If the stimulator is in impedance test mode, it will return to stimulate mode.

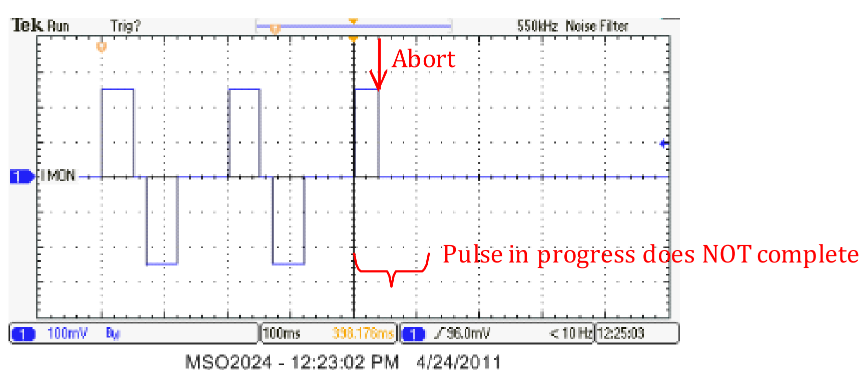
Consider a complete stimulation pattern that is composed of four pulses:



Pressing **Stop**, **Stop All**, **Edit** or **Edit All** in the middle of the third pulse causes the stimulation to end after the third pulse completes. Allowing the pulse to complete helps to preserve charge balance.



Pressing **Abort** in the middle of third pulse results in a truncated third pulse and an unbalanced stimulation, but is the fastest way to stop an ongoing stimulation.



8.13 File Open / File Save

The rectangular pulse parameters (phase amplitude, phase width, inter-phase delay, No. of repetitions, and rate) for all channels can be saved to a user named file for future use by clicking the Save icon or by selecting Save from the File menu. These configuration files have an extension of ".stm". The default location for these files is the directory "C:\PlexonData\Stim-2\Configuration files."

In addition, whenever you close the stimulator user interface, the current values of the rectangular pulse parameters are automatically saved to a file called "LastConfig.stm" in the directory "C:\PlexonData\Stim-2\Configuration files." These parameters are then automatically re-loaded the next time you open the user interface.

Note that if any channels have been configured to use an arbitrary waveform pattern, neither the fact that the channel has been configured to use an arbitrary waveform pattern nor the name of the arbitrary waveform pattern are saved to the configuration file. You must manually configure channels to use arbitrary waveform patterns and manually load the arbitrary waveform patterns for those channels.

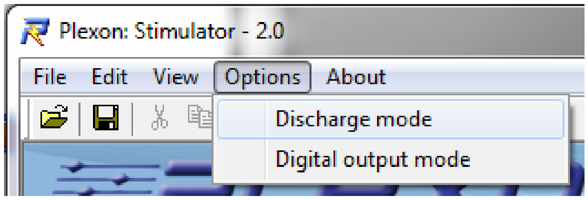
The default values of the rectangular pulse parameters are stored in a read only file called "Factory_Default.stm" in the directory "C:\PlexonData\Stim-2\Configuration files." If you open the "Factory_Default.stm" file just after launching the software you will restore the GUI controls to their factory fresh state.

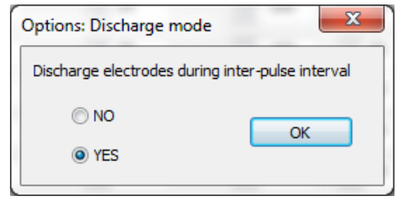
8.14 Options: Discharge Mode

CAUTION

Only advanced users who fully understand the implications should attempt to change the Discharge mode setting. If you have any questions do *NOT* change the Discharge mode setting.

Even when the stimulator is programmed to deliver a "balanced" stimulation in which equal amounts of charge are deposited on and removed from the electrode, a net accumulation of charge on the electrode can occur. This accumulation of charge can occur when the compliance limit is reached during the stimulation (see Section 7.2, "Compliance Voltage and Stimulation Failure" on page 17) or because the electrode response or circuit response is not perfectly linear. The accumulation of charge will result in a slow drifting of the electrode baseline voltage over time and can result in the electrode staying at excessive voltages for prolonged periods of time. In order to guard against unintentional electrode charging, all electrodes are automatically discharged between pulses and any time the channel is not stimulating. Advanced users ONLY may disable the automatic discharge of the electrodes that occurs between pulses. Note however, that disabling automatic electrode discharge can lead to excessive voltages developing on the electrode during extended stimulation patterns. It is essential to monitor the voltage of the electrodes frequently when the automatic electrode discharge feature is disabled. To disable this feature, select Discharge mode from the Options menu and then select NO.

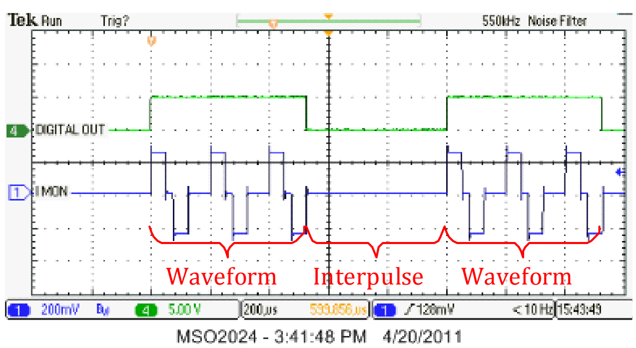


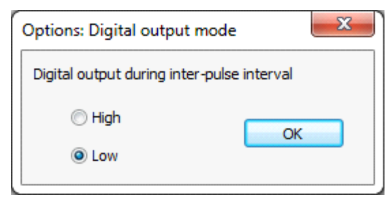


8.15 Options: Digital Output Mode

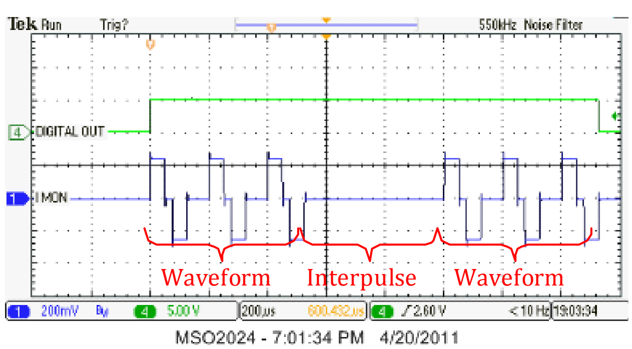
Each stimulator channel has a dedicated digital output that indicates when stimulation is occurring on that channel. The digital output is always high during the pulse or arbitrary waveform output, but the user can control the state of the digital output during the time in between pulses or arbitrary waveforms by selecting **Digital output mode** from the **Options** menu.

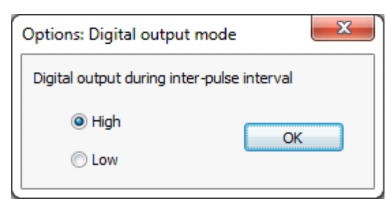
Digital output mode set to **Low** during inter-pulse interval:





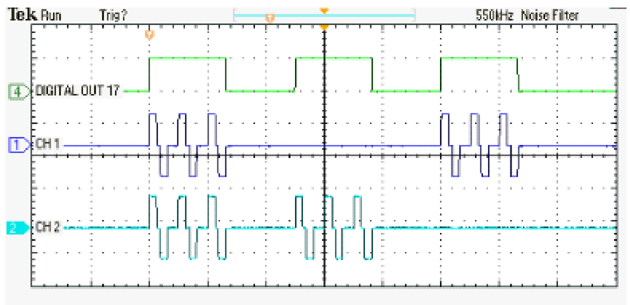
Digital output mode set to **High** during inter-pulse interval:

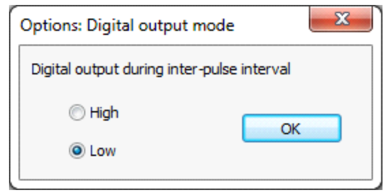




There is also a 17th digital output that goes high when ANY channel is playing back a stimulation waveform.

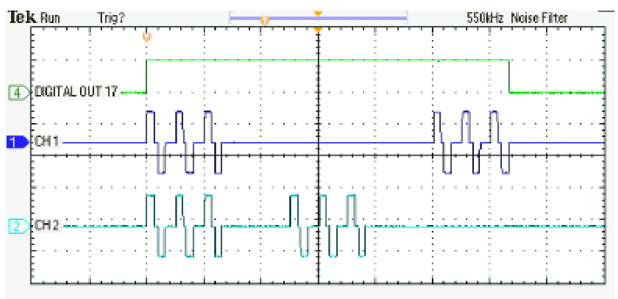
Digital output 17 with **Digital output mode** set to **Low** during inter-pulse interval:

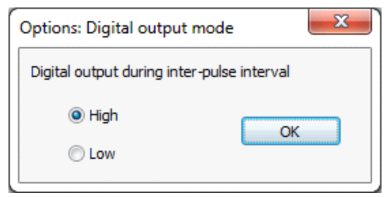




MSO2024 - 4:02:18 PM 4/20/2011

Digital output 17 with **Digital output mode** set to **High** during inter-pulse interval:





MSO2024 - 7:11:40 PM 4/20/2011

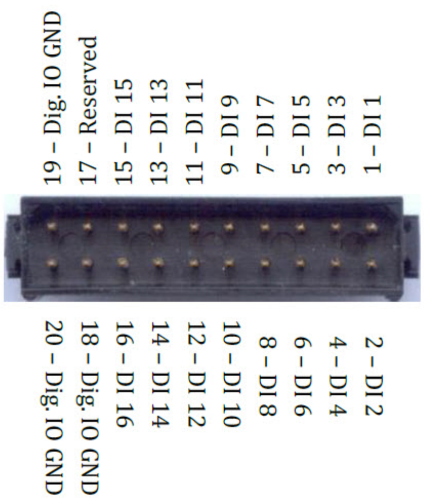
9 Input and Output Connectors

9.1 Power In

This information is provided for reference only. Use the recommended Plexon power supply and cable to power the stimulator. The power input connector is a 2.5mm inner/5.5mm outer barrel connector (Switchcraft L712RA). The stimulator operates from a +12V power supply.

Pin	Function
Center	+12V
Outside	Earth Ground

9.2 Digital In



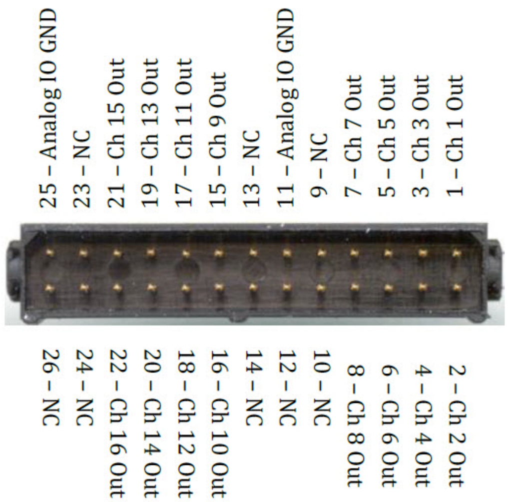
Each channel in the stimulator has a dedicated digital input (DI) that can initiate stimulation on that channel. See Section 7.7, "Starting Stimulation from a Digital Input" on page 26 and Section 8.8, "Trigger Mode" on page 37. The inputs are TTL compatible.

9.3 USB 2.0

The stimulator has a mini type B USB 2.0 receptacle for communications with the host computer. The USB receptacle on the stimulator is connected to digital IO ground. The USB connector on the host computer is typically connected to the AC wall outlet ground. Therefore, the USB cable typically connects the digital IO ground on the stimulator to the AC wall outlet ground.

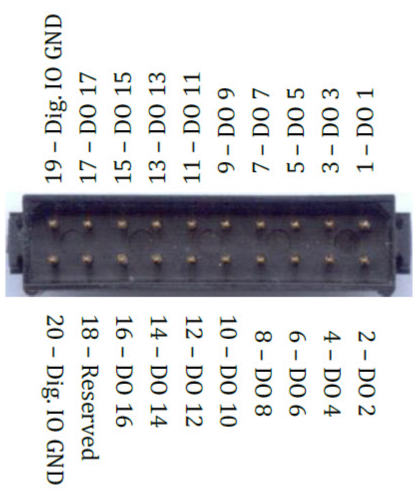
9.4 Stim Out

The stimulation output connector is where the stimulation currents exit the box. Note that the Analog IO ground is isolated from the Digital IO ground and from the AC wall outlet ground.

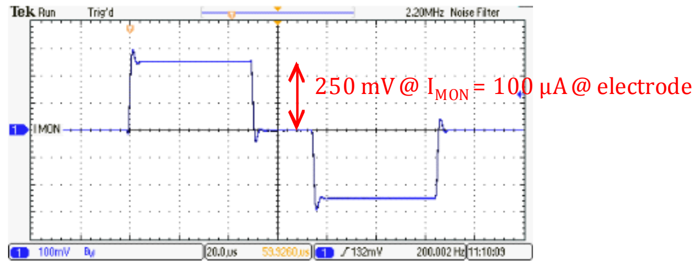


9.5 Digital Out

Digital outputs 1 – 16 indicate when the corresponding channels are stimulating. Digital output 17 goes high when any channel is stimulating. See Section 8.15, "Options: Digital Output Mode" on page 43, for additional details.



9.6 Current Monitor



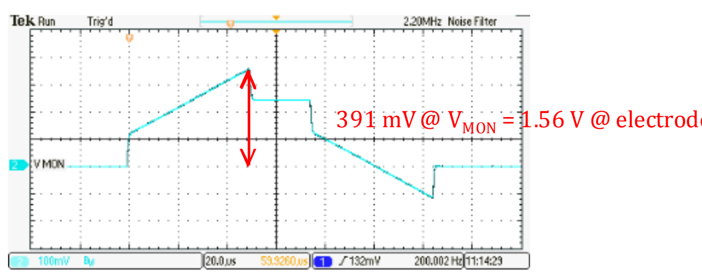
MSO2024 - 11:08:09 AM 4/18/2011

The current monitor displays a scaled representation of the actual current flowing into the selected electrode at any given time. The scaling factor for the current monitor output is 2.5 mV/µA. A 100 µA signal at the electrode will appear as a 250 mV signal on the oscilloscope. Use an oscilloscope to observe the current monitor output.

Note that the outer contact of the current monitor BNC connector is connected to analog IO ground. The outer contact of the BNC connectors on many oscilloscopes is connected to the AC wall outlet ground. Therefore connecting a BNC cable between the current monitor output and an oscilloscope will typically connect the stimulator analog IO ground to the AC wall outlet ground.

If the output of the current monitor does not appear to match the stimulation pattern you requested, refer to Section 6.4, "Verifying the Output on an Oscilloscope" on page 12, and Section 7.2, "Compliance Voltage and Stimulation Failure" on page 17.

9.7 Voltage Monitor



MSO2024 - 11:12:30 AM 4/18/2011

In stimulation mode, the voltage monitor outputs a voltage that is proportional to the voltage being applied to the selected electrode. Refer to Section 6.4, "Verifying the Output on an Oscilloscope" on page 12 for some examples. By default the Vmon scaling (see Section 8.6.1, "Vmon Scaling (Stimulation Mode)" on page 36) is set to 0.25 V/V and the voltage monitor output is one fourth the actual voltage at the electrode. This setting is appropriate for almost all electrodes. Note that the output range of the voltage monitor is ±3.25 V. If the Vmon scaling is set higher than 0.25V/V, then the output of the voltage monitor will saturate before the compliance limit is reached and the may not reflect the actual voltage at the electrode. For example, with the Vmon scaling set to 2.5 V/

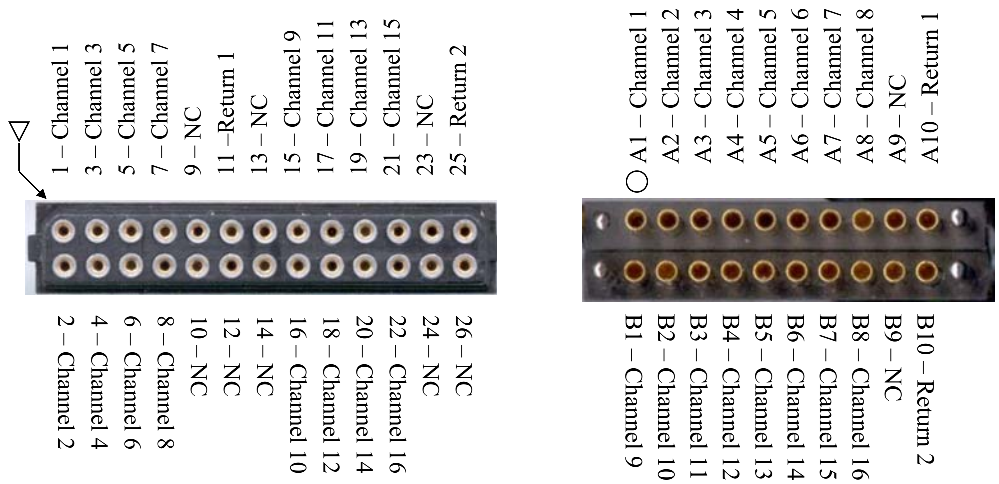
V, the voltage monitor will saturate when the electrode voltage is only ±1.3V, which is well below the maximum stimulator output voltage.

In Z test mode, the voltage monitor outputs a voltage that is proportional to the impedance of the electrode. See Section 7.5, "Impedance Measurement” on page 21 and Section 8.6, "Vmon Scaling and Z Conversion” on page 36 for additional details.

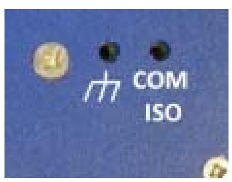
Note that the outer contact of the voltage monitor BNC connector is connected to analog IO ground. The outer contact of the BNC connectors on many oscilloscopes is connected to the AC wall outlet ground. Therefore connecting a BNC cable between the voltage monitor output and an oscilloscope will typically connect the stimulator analog IO ground to the AC wall outlet ground.

9.8 Stimulation Cable

The pinout for the stimulation cable, Plexon PN 14-03-A-03, is shown below.



9.9 Ground/Common Access Points

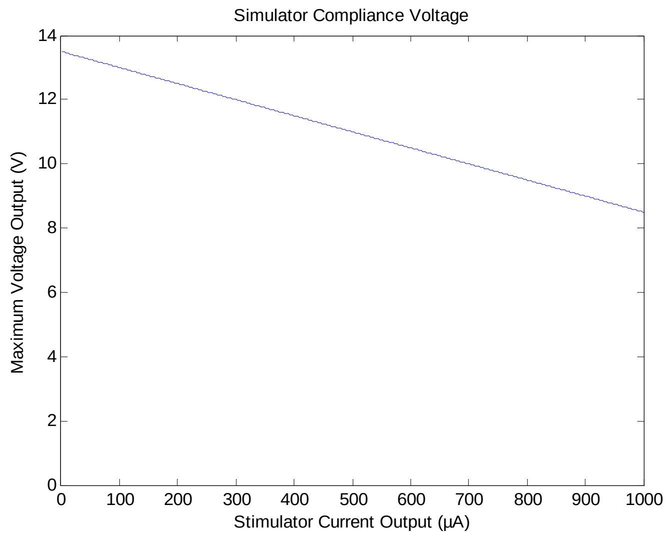


The PlexStim Electrical Stimulator is electrically isolated. This means that there is no relation between the return path of the stimulation current (Analog IO GND) and earth ground. There are connections on the back of the stimulator that give access to both earth ground (rake symbol) and the return path of the stimulation current, Analog IO GND (marked COM ISO). The plugs are designed to fit a 0.080” (2mm) connector (Johnson Components PN 105-0304-001). The connections can be used to connect the isolated return to the common of another isolated system, or to earth ground the stimulator. Note that connecting COM ISO to anything that is not electrically isolated will break the isolation of the stimulator.

10 PlexStim Stimulator Limitations

10.1 Maximum Compliance Voltage Varies with Stimulation Amplitude

The “compliance voltage” or maximum voltage that the stimulator will deliver to an electrode varies with the stimulation current as shown in the graph below. For very low stimulation currents the maximum voltage the stimulator will deliver is ±13.5 V. The compliance voltage decreases linearly with increasing stimulation current. At the maximum output current of ±1 mA, the compliance voltage is reduced to ±8.5 V.



There is a 5 kΩ resistor in series with the current output. As with any resistor, there is a voltage drop across the resistor that is proportional to the current going through the resistor (V = I*R). The voltage drop across this resistor is used to measure the actual output current, but reduces the voltage available to deliver to the electrode.

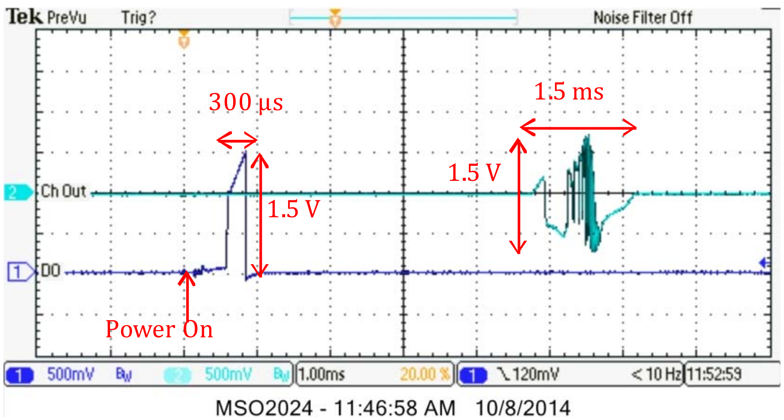
10.2 Power on/off Transients

To avoid transients at the stimulator outputs caused by turning the power on and off, the following sequence is recommended for conducting a stimulation experiment:

1. TURN THE STIMULATOR POWER ON
2. LAUNCH THE USER INTERFACE PROGRAM
3. CONNECT THE STIMULATOR TO THE ELECTRODES
4. CONDUCT THE EXPERIMENT
5. CLOSE THE USER INTERFACE PROGRAM
6. DISCONNECT THE STIMULATOR FROM THE ELECTRODES
7. TURN THE STIMULATOR POWER OFF

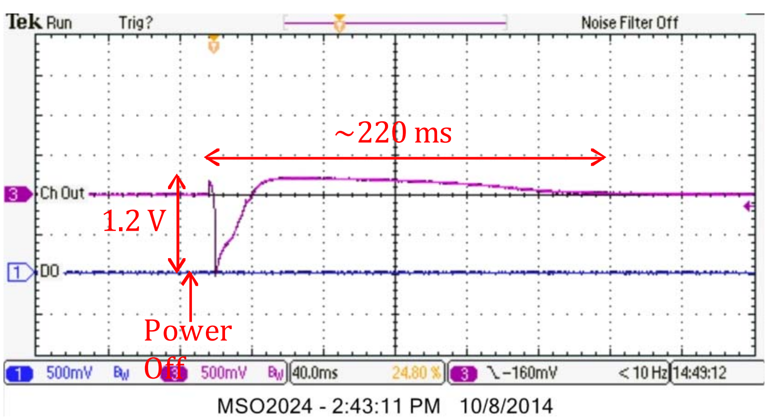
Do not turn the power supply to the stimulator on and off rapidly.

Transients may occur at the stimulator digital and current outputs when the stimulator power supply is turned on. A typical power on voltage transient is shown below:



A typical transient on the digital output may reach ~1.5 V and last ~300 µs. A typical transient on the current output may reach ~1.5 Vpp and last ~1.5 ms. For this reason, it is advisable to not connect the stimulator outputs to the electrode implant until after the stimulator power has been turned on.

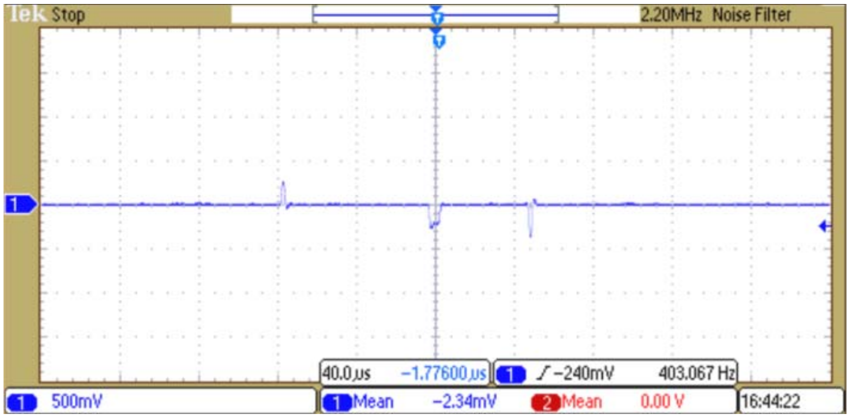
A transient may occur at the stimulator current output when the power is turned off. A typical power off voltage transient is shown below.



A typical transient on the current output may reach ~1.2Vpp and last ~220 ms. For this reason, it is advisable to disconnect the stimulator from the electrodes before turning the power off.

10.3 Current Monitor with an Open Circuit

Ideally, when no electrode is connected to the stimulator output, the current monitor should always read 0V. However, some artifacts may appear on the current monitor when the programmed current output changes rapidly. For example, stimulating with a rectangular pulse pattern when no electrode is connected will result in transients on the current monitor at the beginning and end of each phase of the pulse as shown below.



Whenever the current monitor does not appear to match the requested stimulation pattern, the voltage monitor should be examined. Refer to section Section 6.4, "Verifying the Output on an Oscilloscope" on page 12, and Section 7.2, "Compliance Voltage and Stimulation Failure" on page 17.

10.4 Impedance Test with an Open Circuit

Ideally, if no electrode is connected to the stimulator (or the connection to the electrode is broken) then the impedance test procedure would indicate infinite impedance. However, due to stray capacitance in the output of the stimulator and in the cabling to the electrode, the impedance test procedure will yield a finite but large impedance result. With no cable connected to the stimulator, this open circuit impedance will vary from channel to channel and will typically range from 5.5 MΩ to 11.5 MΩ. With the 14-03-A-03 stimulation cable connected to the stimulator, but no electrodes connected to the cable, the open circuit impedance will typically range from 1.5 MΩ to 5.75 MΩ.

11 Specifications

General	
Dimensions power supply stimulator	5.7 in x 2.5 in x 1.3 in 8.2 in x 2.8 in x 4.1 in
Weight power supply stimulator	0.9 lbs 2.0 lbs
Power requirements	100 – 240 VAC, 50/60 Hz, 2 A
Operating Systems	Windows 7
Interface	USB 2.0 (mini type B)
Analog outputs	
Stimulation mode	Current Control
Number of analog output channels	16 outputs and 1 return on each device. Channels are independent. Up to 64 channels if four devices are used simultaneously.
Maximum Current	±1 mA (Stimulation mode)
Resolution	16 bits
Output current resolution	30 nA (Stimulation mode)
Compliance voltage	10 V @ 700 µA (See Section 10.1)
Temporal resolution	1 µs
Output rise time	1.25 µs 100 µA, 10 kΩ load
Minimum pulse width	5 µs
Minimum pulse rate	0.008 Hz (125 s)
Maximum pulse rate	100 kHz
Digital inputs/outputs	
Number of digital inputs	16
Number of digital outputs	17
Digital input levels	TTL, Low < 0.8V, High > 2.0V
Digital output levels	HCT, Low < 0.33V, High > 3.84V
Digital input/output latency	~2 µs
Maximum trigger rate	100 kHz
Minimum trigger pulse width	1 µs
Arbitrary waveforms	
Maximum sample points	999
Maximum update rate	1 MHz (1 µs)

12 List of Related Documents

In addition to this user guide, the following PlexStim Stimulator documents are available on either the Software Downloads or the Documentation web pages (see www.plexon.com):

- PlexStim Electrical Stimulator Data Sheet
- PlexStim Electrical Stimulator DLL Guide
- PlexStim Electrical Stimulator MATLAB API Definitions
- PlexStim Electrical Stimulator Change Log

Instructions for how to import the PlexStim DLL into LabVIEW™ can be found in Appendix A.

Appendix A
Importing PlexStim DLL into LabVIEW

A.1 Introduction A-2
A.2 Hardware Compatibility A-2
A.3 Basic DLL Import A-3
A.4 Correcting Import Errors A-11
A.5 Fix Error Reporting A-15
A.6 Improve Error Description Reporting (Optional) A-16
A.7 Special Note on Close Functions and Error Handling A-18
A.8 Updating User Programs A-19

A.1 Introduction

The PlexStim dynamic link library (DLL) provides a mechanism for user-written programs to operate PlexStim Stimulator hardware. This document outlines the process of importing the Revision 2.3.17.0 PlexStim DLL into LabVIEW™ so that the stimulator can be controlled from user-written LabVIEW programs.

The basic process of importing a DLL into LabVIEW is fairly straightforward and automatic. Note however that during the automatic import process several of the functions will not import correctly and the VIs corresponding to these functions will need to be corrected. The LabVIEW error handling function for the imported library also requires a slight modification. Finally an optional step can be taken to provide LabVIEW with the English language descriptions of the error codes provided by the PlexStim Stimulator library.

PlexStim Stimulator Revision 2.3.17.0 DLL contains several changes that will not be compatible with programs written for earlier versions of the DLL. User programs will likely need to be modified before they can function with the Revision 2.3.17.0 DLL. In particular, the return value of all functions in the DLL is now an integer value that represents the error status of that function call. Furthermore all functions use a unified error reporting code. In previous versions of the DLL some functions returned nothing (void), an error code, a parameter value, or reported a combination of error code and parameter value. See the PlexStim DLL change log (14-20-D-18) for a complete listing of the functions that have changed and a description of the changes.

The revision 2.3.17.0 DLL will not recognize PlexStim Stimulators manufactured prior to April 2015 that have not been upgraded. See the Section 2, “Version and Compatibility Notice” on page 2 for additional details.

If you are running a 64-bit version of LabVIEW, you must import the 64-bit version of the PlexStim DLL and if you are running a 32-bit version of LabVIEW, you must import the 32-bit version of the PlexStim DLL.

As new LabVIEW software is issued by the owner of that product, it is possible that the process of loading the PlexStim DLL into a later version of LabVIEW might be somewhat different than the procedure presented in this Appendix. If you experience any difficulties loading the DLL, contact Plexon support by telephone at +1 214-369-4957 or by email at support@plexon.com.

A.2 Hardware Compatibility

PlexStim.dll Version 2.3.17.0 is designed to operate with PlexStim Stimulators running firmware Revision A (Firmware part number 14-20-A-07-A). Stimulators that were manufactured with revision blank firmware (14-20-A-07) will generally not be recognized by the 2.3.17.0 DLL as valid stimulators.

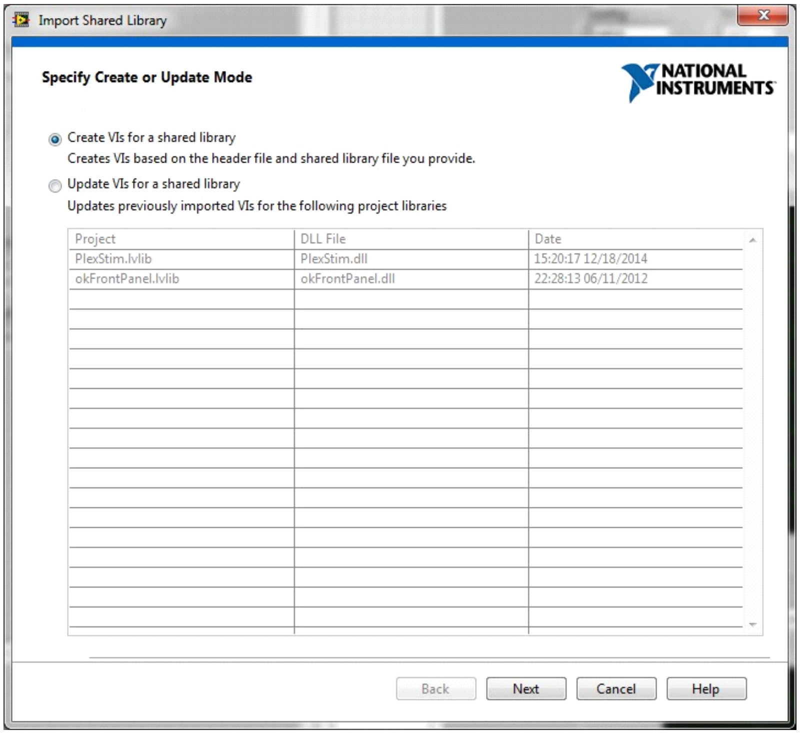
Stimulators of hardware Revision F (14-20-A-10-F) are the first stimulators originally manufactured with Revision A firmware. Examine the labels on the bottom of the stimulator to determine the firmware and hardware revision. If your stimulator is not revision F or later or does not have revision A or later firmware, contact Plexon support for information on upgrading the firmware in your stimulator. Please note that after the stimulator Hardware/Firmware has been upgraded you will be required to use the Revision 2.3.17.0 or later DLL to operate the stimulator as prior revisions of the DLL will not function properly with the upgraded hardware/firmware.

A.3 Basic DLL Import

If you have previously imported the PlexStim Stimulator library into LabVIEW and you are now updating the library, make sure to close any VIs that might make use of the library before proceeding. Ideally only the LabVIEW "Getting Started" screen should be open:

1 From the Tools menu, select import, shared library.

The Import Shared Library dialog opens.



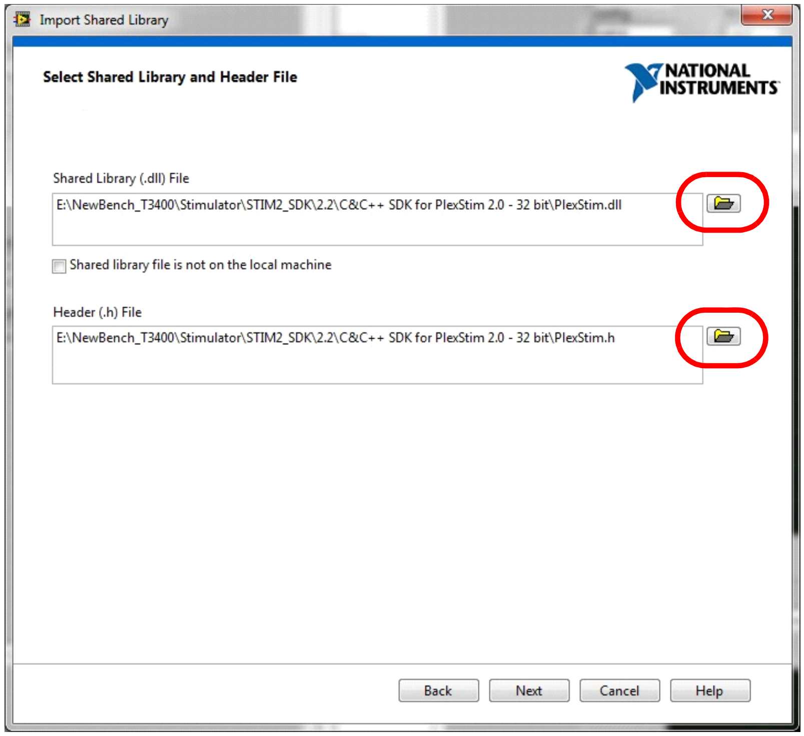
2 Note that there are two options available in this dialog:

- Create VIs for a shared library (default)
- Update VIs for a shared library

If this is the first time you are importing the PlexStim library into LabVIEW, the applicable option is already checked by default, so just click Next.

If you have previously imported an earlier version of the PlexStim library into LabVIEW, select this option and then select the PlexStim.lvlib from the list of libraries. Then click Next.

3 When the next dialog opens, click each of the folder icons to browse to the location of PlexStim.dll (or PlexStim64.dll if you are using a 64-bit version of LabVIEW) and PlexStim.h files on your computer.

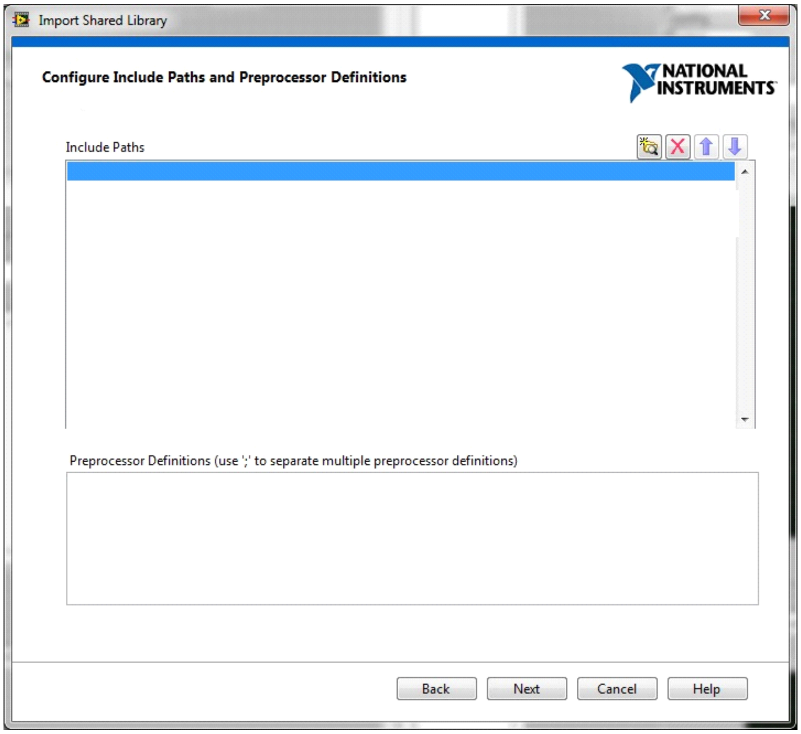


By default, LabVIEW assumes the PlexStim.h file is in the same path as the PlexStim.dll file. If you want to specify a different path, click the folder icon, then browse and select the folder containing the PlexStim.h file.

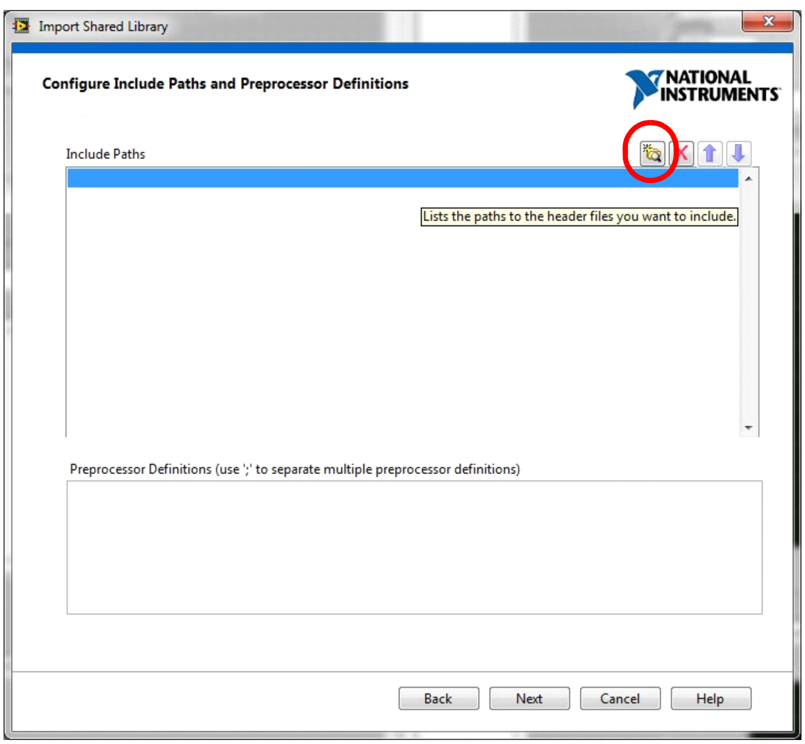
Note: In addition to PlexStim.h, the PlexStim library makes use of another header file PlexStimTypes.h. The path to this additional header file will be specified on the next screen.

4 Click Next.

A new dialog opens.



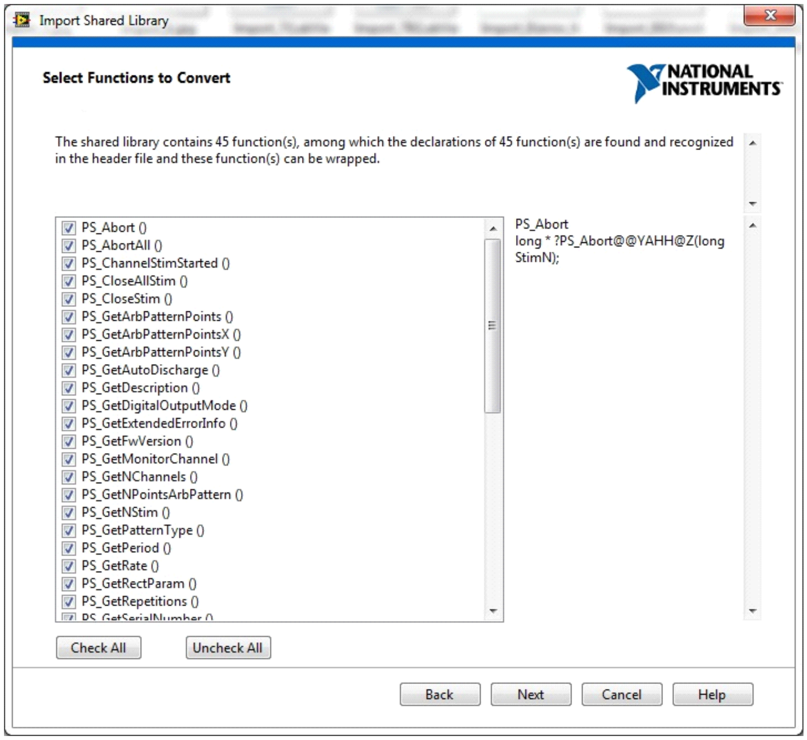
5 Click the new folder icon in the upper right portion of the window. This creates a line in the box labeled Include Paths. Select the folder that contains the PlexStimTypes.h file (not the file, the folder it is in). No Preprocessor Definitions are required.



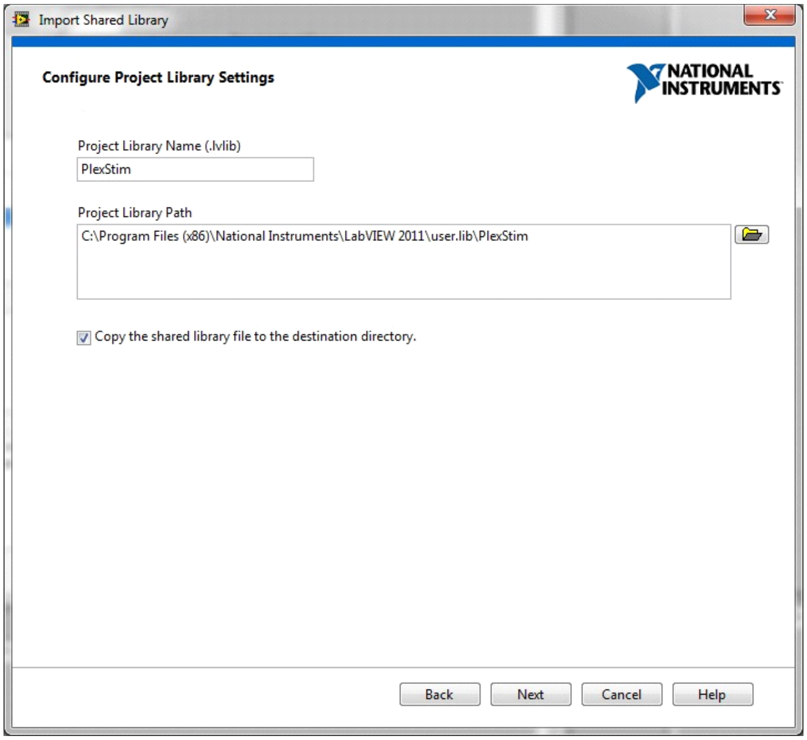
6 Click Next.

A new dialog opens.

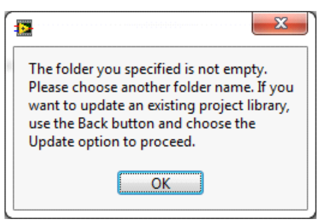
7 In this dialog, verify that LabVIEW recognizes 45 functions, all of which are selected for import by default. Accept the default selections, then click Next:



8 When the next dialog opens, make sure the "Copy the shared library" box is checked, then click Next



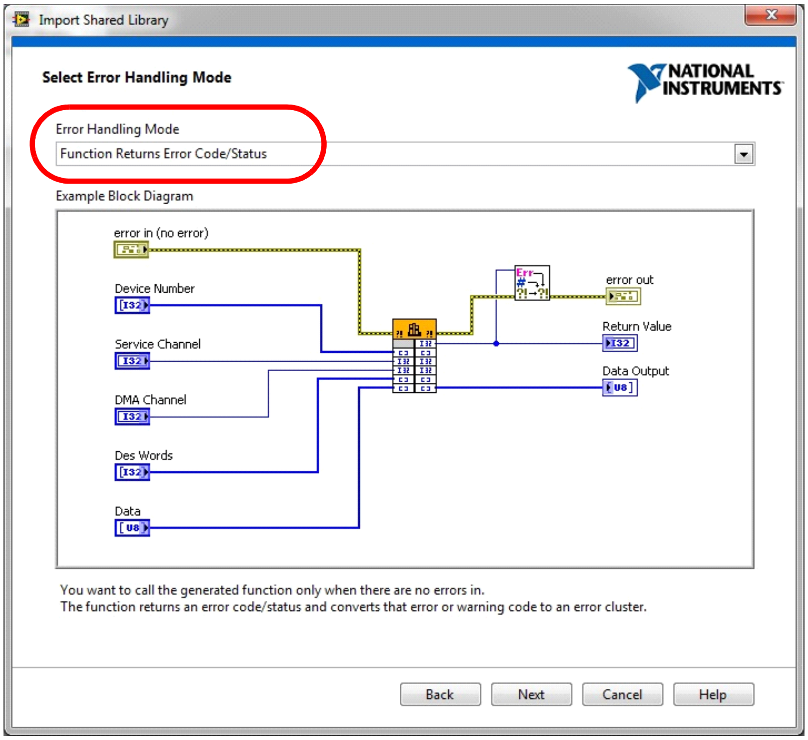
9 (If required) If you chose to "Update" an existing library at the beginning of the import process (in Step 2), you might now get an error stating that the folder you specified is not empty:



If you see this error notification, click OK, browse to and rename the indicated folder for the existing library (for example by adding "backup" at the end of the folder name). Then click Next again.

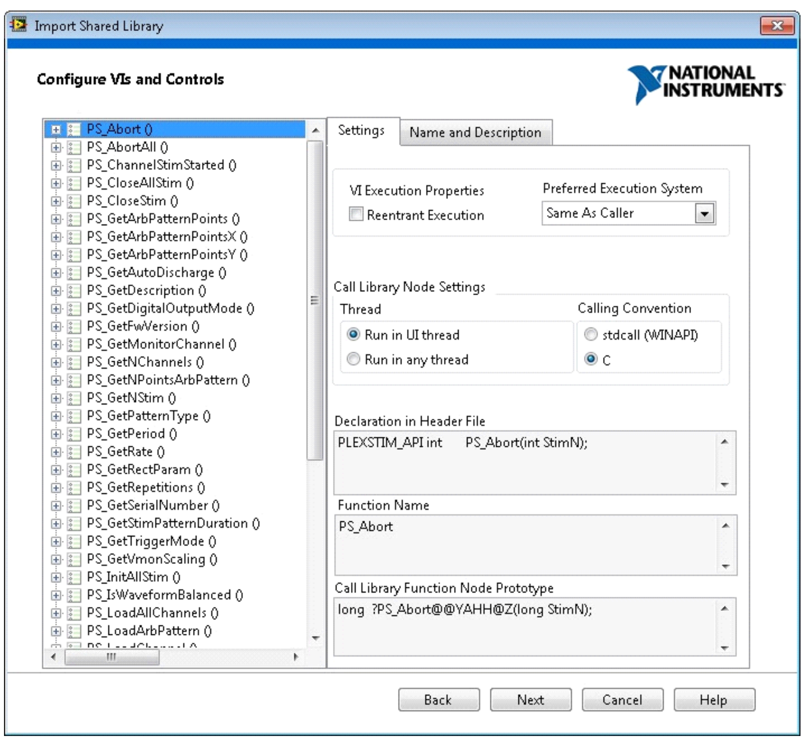
10 LabVIEW provides several modes for error handling, including "No Error Handling", "Simple Error Handling", "Function Returns Error Code/Status", and "Call Another Function to Check Errors." In the 2.3.17.0 version of the PlexStim DLL, the return value of all functions is an error/status code. This behavior meshes well with the

LabVIEW “Function Returns Error Code/Status” mode. Select this mode to have LabVIEW automatically convert the function return value into a LabVIEW error.

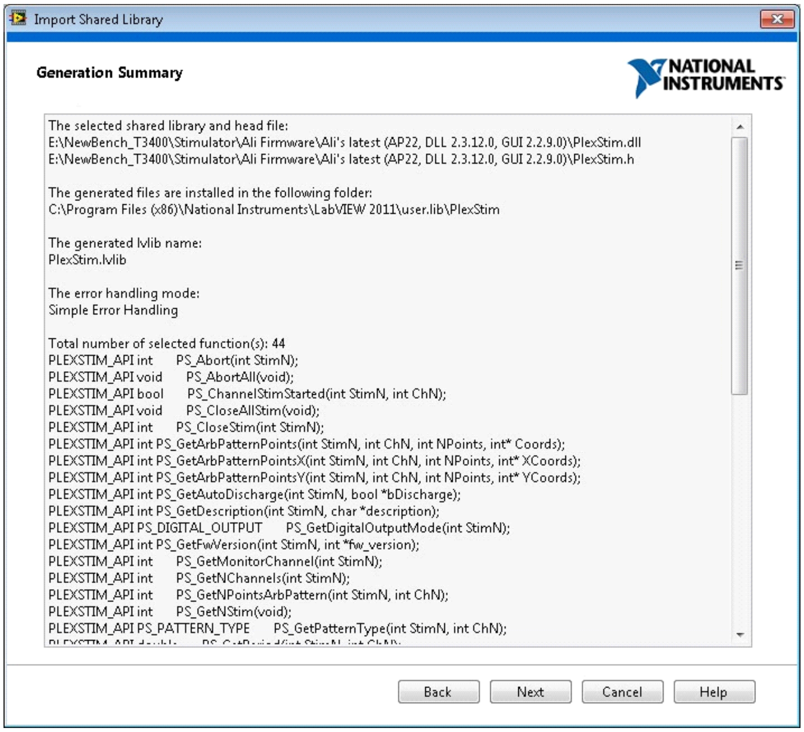


11 Click Next.

12 On the next screen, leave the settings as they are and just click Next.

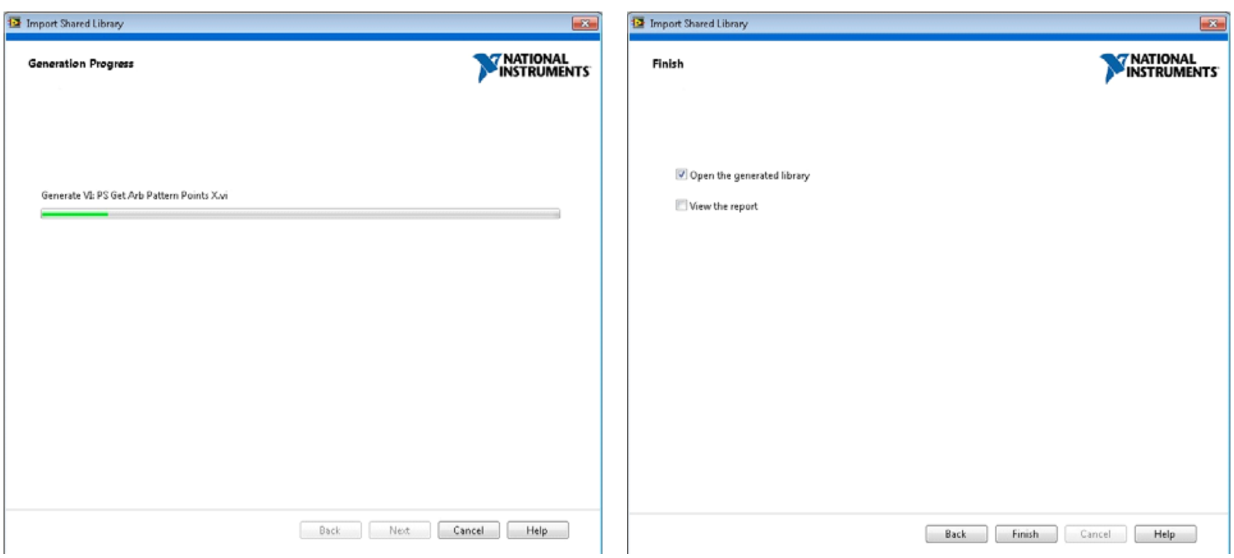


13 A summary is displayed, click Next to continue.



14 LabVIEW displays a progress bar (left) and then finally finishes (right)l.

Click Finish.



The library has now been imported into the user.lib subdirectory of the LabVIEW folder.

Note however, that several of the library functions do NOT get imported correctly and the LabVIEW error handling function also requires adjustment. The next sections provide instructions for addressing these issues.

A.4 Correcting Import Errors

Several functions in the PlexStim library return an array of values. This array of values is passed using a pointer. The user of the function passes a pointer to an array to the function and the function populates the memory area specified by that pointer with the requested values. Before calling such a function, however, it is necessary to allocate enough memory space to accommodate the returned values.

Unfortunately LabVIEW cannot allocate this memory until we tell it how much information will be returned by the function. Since LabVIEW did not have this information at the time the function was imported, it was not able to correctly allocate the required memory and the corresponding functions must be manually edited to supply this information.

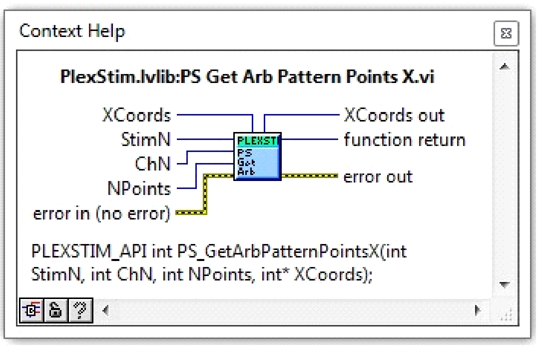
A.4.1 PS_GetArbPatternPointsX

As an example, consider the library function PS_GetArbPatternPointsX:

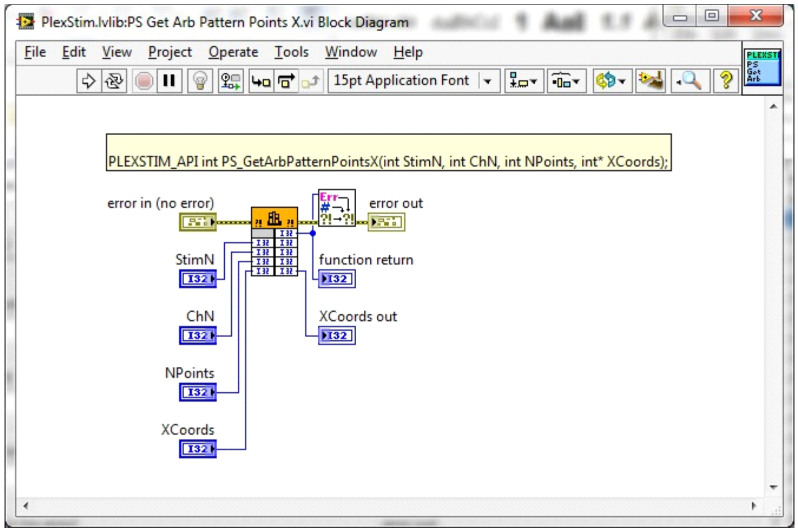
```
int PS_GetArbPatternPointsX(int StimN, int ChN, int NPoints, int* XCoords);
```

1 View the Context Help window for this library function.

Note that this function returns the X coordinates that would be useful for drawing the stimulation pattern associated with a particular channel. LabVIEW assigns both an input called XCoords and an output called XCoords out.

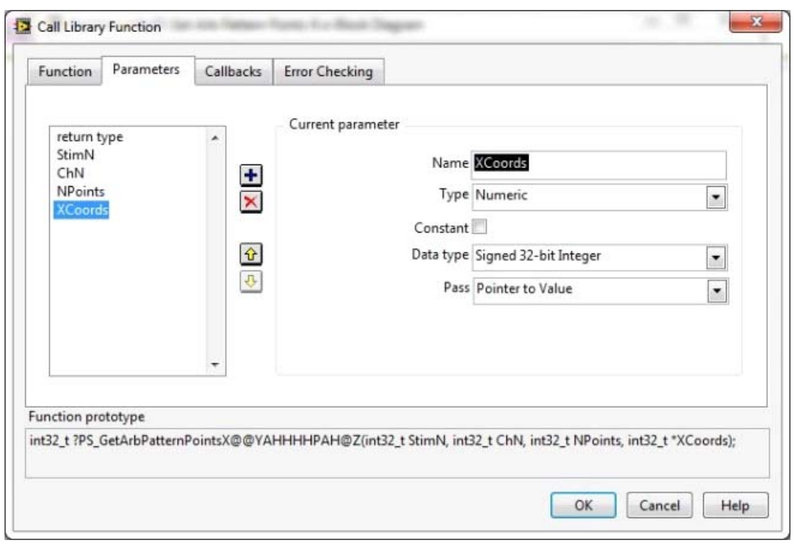


The block diagram for this function is shown below. Notice that the type of both "XCoords" and "XCoords out" is a single I32 (32 bit integer), not an array:

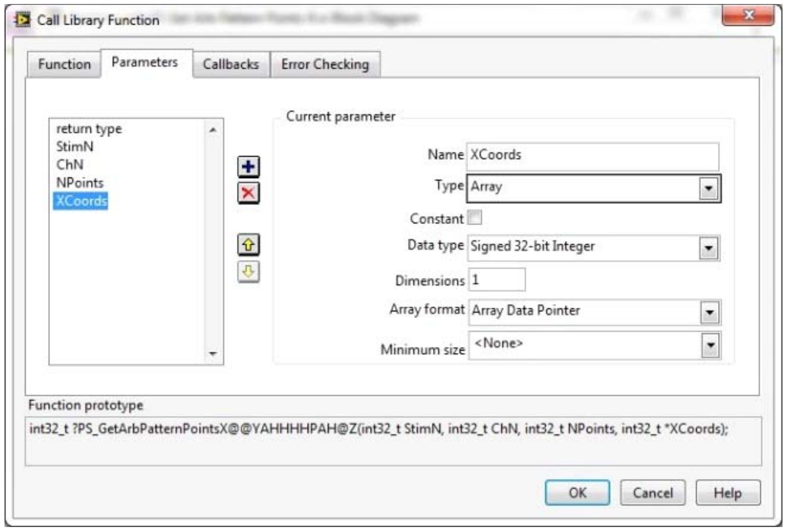


2 You do not need to access the XCoords input, but you do need to allocate sufficient memory to store the correct number of coordinates. You also need the output to be an array of I32 instead of a single value.

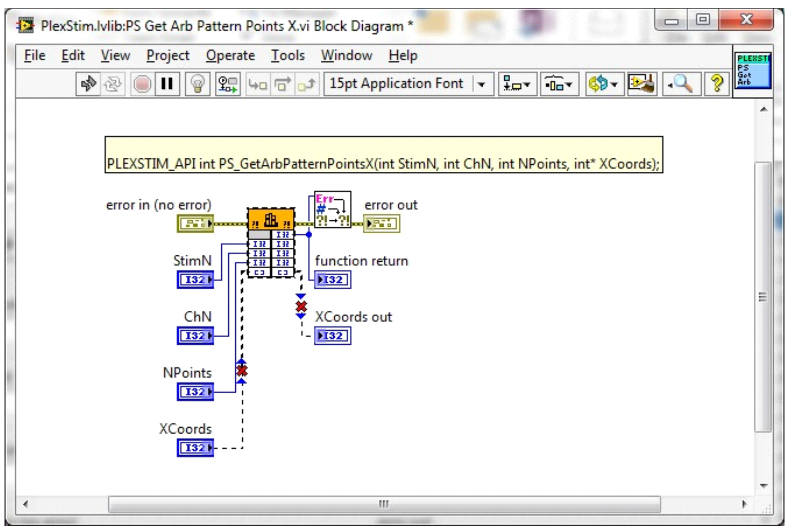
To accomplish this, double click on the "Call Library Function Node" icon to bring up the Call Library Function dialog, then click on the Parameters tab and select the XCoords parameter.



3 Change the Type from Numeric to Array.



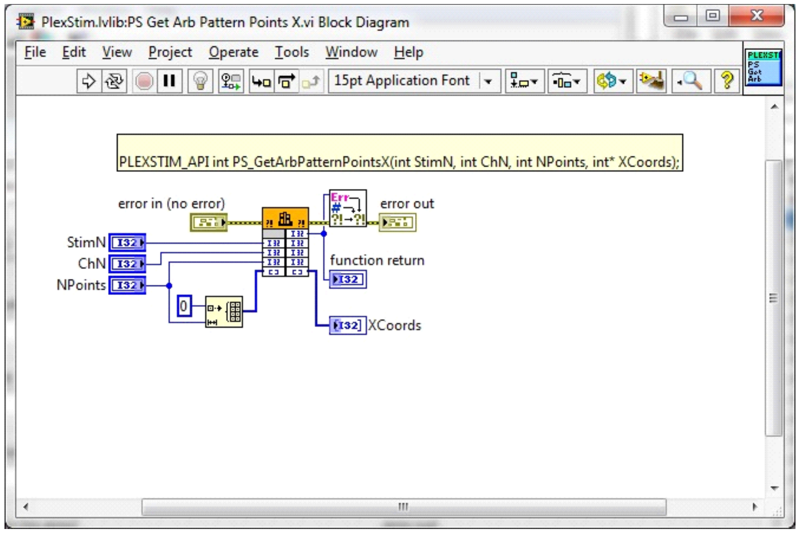
4 Click OK. The Block diagram of the function is now broken:



5 Delete the XCoords control and XCoords out indicator. Then right click on the broken wire on the lower right hand corner of the Call Library Function Node (where XCoords out used to be) and select create, indicator to create a new indicator for the array output.

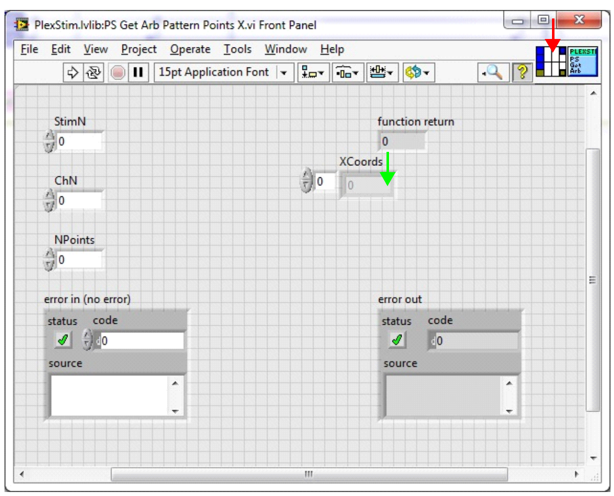
6 The function returns an array of NPoints in length. On the input (left) side of the Node, use the Initialize Array VI to create an array of I32 of size NPoints and connect

that to the input of the Node. The resulting block diagram should appear as shown below.



7 Next go to the front panel of the VI:

Re-link the new XCoords output array to the output terminal of the VI by first clicking on the terminal indicated by the red arrow in the figure below, and then clicking on the XCoords indicator (green arrow below). Then save the VI. The VI is ready to use.



A.4.2 PS_GetArbPatternPointsY

The function PS_GetArbPatternPointsY:

int PS_GetArbPatternPointsY(int StimN, int ChN, int NPoints, int* YCoords);

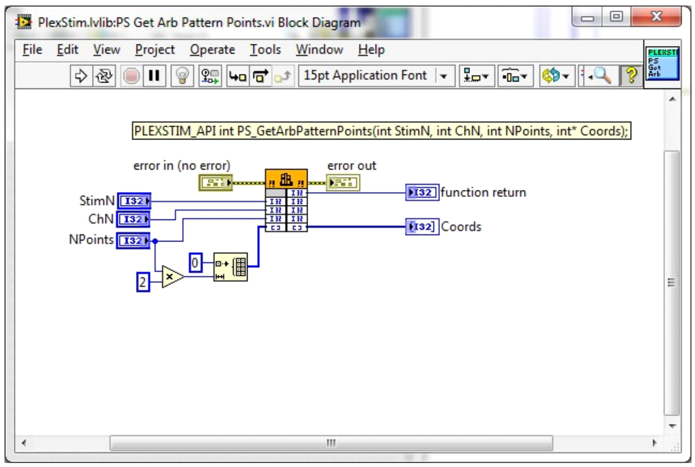
needs to be repaired in a similar fashion to PS_GetArbPatternPointsX as described in Section A.4.1, “PS_GetArbPatternPointsX” on page A-11. Follow similar steps to repair this function.

A.4.3 PS_GetArbPatternPoints

The function PS_GetArbPatternPoints:

```
int PS_GetArbPatternPoints(int StimN, int ChN, int NPoints, int* Coords);
```

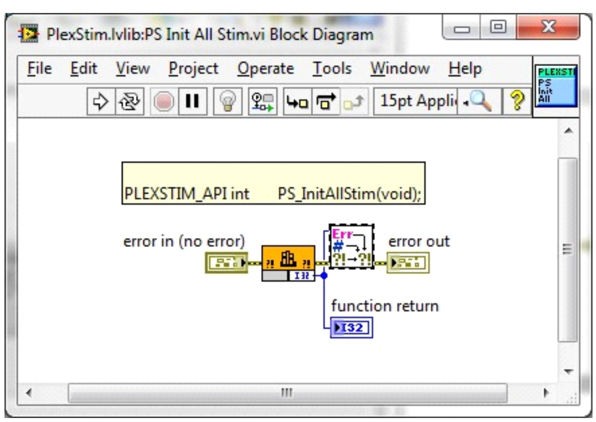
requires similar repairs as both PS_GetArbPatternPointsX and PS_GetArbPatternPointsY, however since it returns both the X and Y coordinates together, it returns twice as many point (2 *NPoints). Therefore the array we pass to the library node must be dimensioned twice as large. This is accomplished by multiplying NPoints by 2 as shown in the block diagram below:



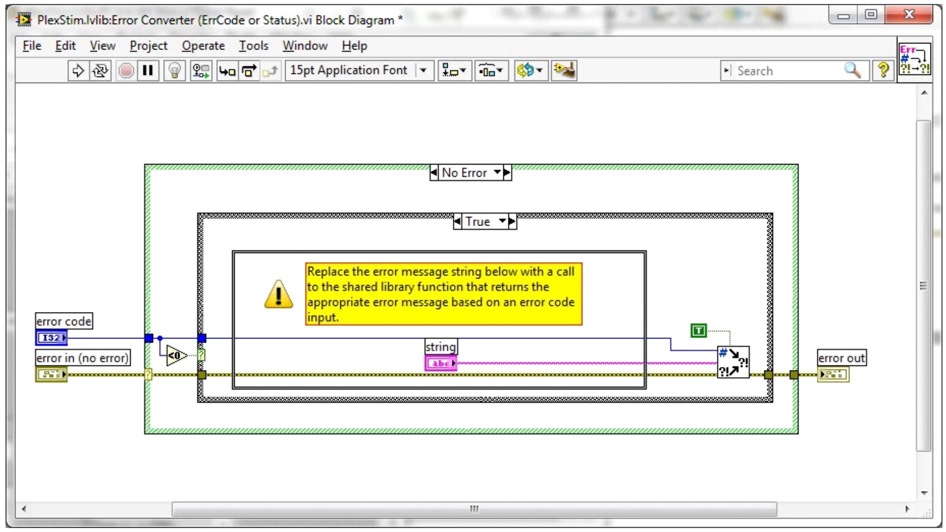
A.5 Fix Error Reporting

By default, when the PlexStim library is imported into LabVIEW with the “Function Returns Error Code/Status” error mode option described in Section A.3, “Basic DLL Import” on page A-3, LabVIEW will automatically generate an error whenever the value returned by the function is negative. However, as described in the PlexStimTypes.h header file, the PlexStim DLL returns both positive and negative error code values. Therefore it is necessary to tell LabVIEW to accept and process the positively valued error codes.

1 Open any VI from the library, for example PS_InitAllStim:



2 Open the “error converter” (red arrow in picture above):



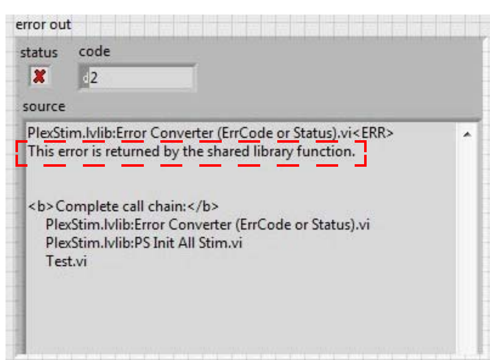
3 Notice that as the error code enters the VI, it is checked to see if it is <0. Right click the less than zero VI and select replace, comparison palette, not equal to zero (?0).

4 The error converter VI will now treat all non zero return values as errors. Save the error converter VI.

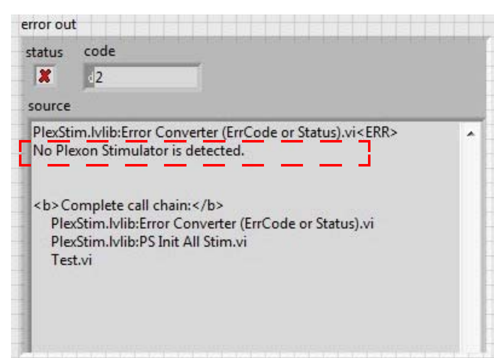
A.6 Improve Error Description Reporting (Optional)

After the library has been imported, the default behavior is for LabVIEW to report the numeric error code along with a generic error description “This error is returned by the shared library function.” However, it is possible to use the new PS_GetExtendedErrorInformation function in the PlexStim library to give LabVIEW access to a specific English Language description for each error code. An example is shown below:

Generic error description

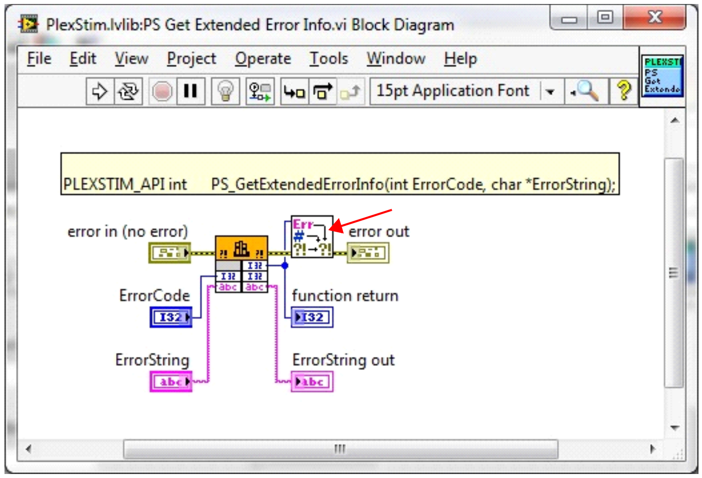


Improved error description

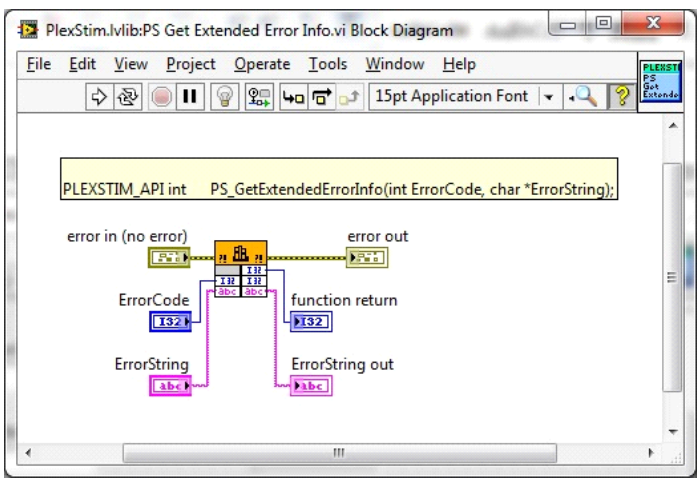


Follow the steps below to enable the English language description of the error to be passed through the LabVIEW error handler.

1 Open the PS_GetExtendedErrorInfo VI.



2 Delete the error converter (see the red arrow in the diagram above), then repair the error output wire and delete the broken function out wire as shown in the diagram below. (This is necessary to prevent infinitely recursive calling of the error converter VI, when PS_GetExtendedErrorInfo is added to the error converter VI in the steps below.)

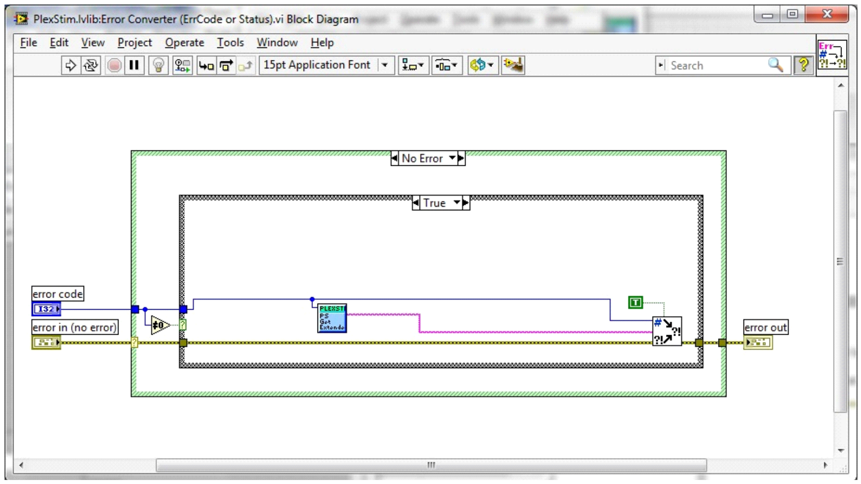


3 Save the PS_GetExtendedErrorInfo VI.

4 As described in Section A.5, “Fix Error Reporting” on page A-15, open any VI in the library and then open the error converter function (see Section A.5 for a picture). Carefully make the following changes:

- Delete the exclamation point and yellow text box.
- Delete the string input going to the “error cluster from error code” VI.
- Place the PS_GetExtendedErrorInfo VI on the error converter block diagram.
- Wire the string output of the PS_GetExtendedErrorInfo VI to the error cluster from error code VI as shown.

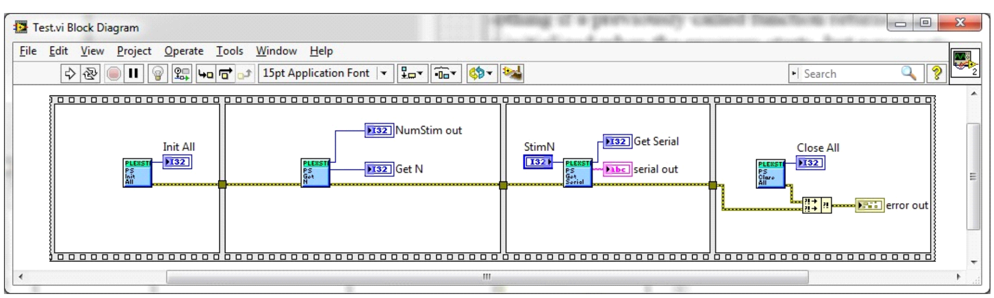
- Note carefully how the error code is wired in the picture below!
- The incoming error code should be passed to BOTH the PS_GetExtendedErrorInfo VI function and to the error cluster from error code. The error code output from PS_GetExtendedErrorInfo should not be used.



5 Save the error converter VI.

A.7 Special Note on Close Functions and Error Handling

The default LabVIEW error handling behavior is to do nothing if a previously called function returned an error. This can lead to a situation where a stimulator get initialized when the program starts, but never gets released because an error in the program in the middle of the program prevents the "CloseStim" or "CloseAllStim" function to be called at the end of the program. One simple way around this problem is to not connect the prior error stream to the input of the Close function as shown below:



This allows the PS_CloseAllStim function to execute even if there was an error earlier in the program.

A.8 Updating User Programs

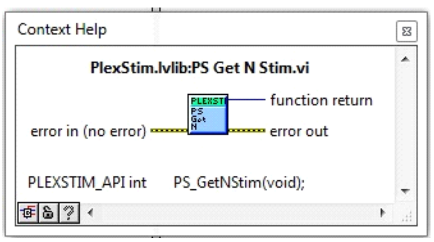
The return values of several of the functions in the PlexStim library have changed starting with Revision 2.3.17.0. See the PlexStim Electrical Stimulator DLL Guide on the Plexon Documentation web page (www.plexon.com) for a complete list of the changes. If you are calling any of the functions that have "major" changes as described in the PlexStim Electrical Stimulator DLL Guide, carefully examine each instance where these functions are called to see if the wiring around them needs updating.

As an example, the PS_GetNStim function's present and former behavior is shown below:

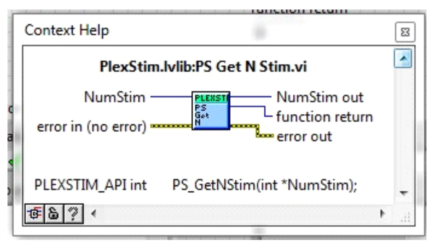
Current	int PS_GetNStim(int *NumStim);
Former	int PS_GetNStim(void);

Although the "function return" is still an integer value, this integer return value now represents an error code instead of the number of stimulators. The number of stimulators is now returned as a separate parameter called NumStim. The newly imported VI has a new "NumStim out" terminal that was not present before. This change might require corresponding changes to the user programs that call the function. Examine the PS_GetNStim VI before and after updating the library:

Before



After



Notice that before updating, the "function return" output is in the upper right hand corner of the VI. After updating, the "NumStim out" output is in the upper right hand corner of the VI and the "function out" is below it. Purely by coincidence, the importer assigned the new "NumStim out" output to the same terminal that was formerly used for the "function return". This is fortuitous, because the meaning of that output terminal was preserved—In both cases it carries information about the number of stimulators connected to the PC. The library importer assigned the function outputs to the terminals on the VI front panel in such a way that no change is required to the upper level program. However, you cannot always expect to be so fortunate.

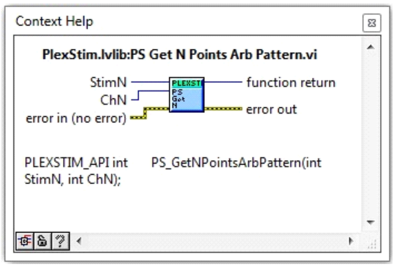
As a counter example, consider the PS_GenNPointsArbPattern function:

Current	int PS_GetNPointsArbPattern(int StimN, int ChN, int *points);
Former	int PS_GetNPointsArbPattern(int StimN, int ChN);

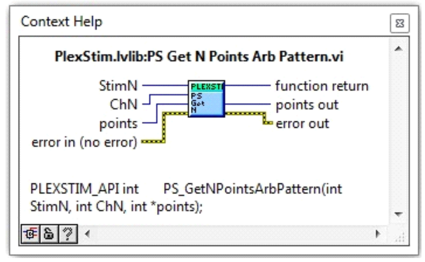
This function changed in a seemingly similar way as PS_GetNStim. The function return is still an integer, but the integer now represents an error code instead of the number of points in the waveform which is returned as a new parameter called “points.”

However, when we examine the PS_GenNPointsArbPattern VI before and after import, we see that LabVIEW handled this function in a different manner:

Before

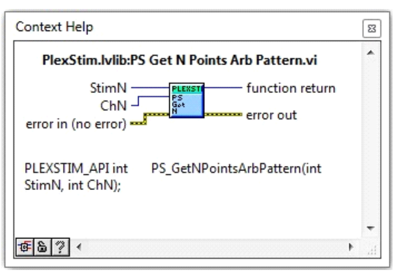


After



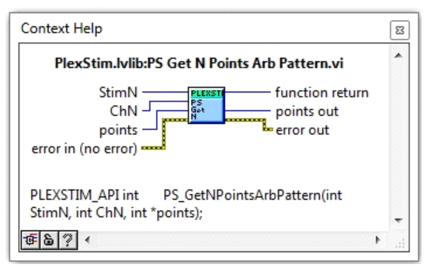
Notice that in this case, the function return stayed in the upper right corner and the new “points out” output was added below it. Programs that call this VI and make use of the number of points returned will have to be rewired. A simple example scenario is shown below where the number of points in the waveform is passed from PS_GetNPointsArbPattern to the function PS_GetArbPatternPoints. Even though LabVIEW does not report an error, after importing the DLL, we must move the wire connecting the two VIs such that it connects the “points out” output of PS_GetNPointsArbPattern to the “NPoints” input of PS_GetArbPatternPoints instead of connecting the function return (now an error code) to NPoints. Otherwise the error code will be passed from PS_GetNPointsArbPattern to PS_GetArbPatternPoints instead of the number of points in the waveform.

Before



(wire connected to function out)

After

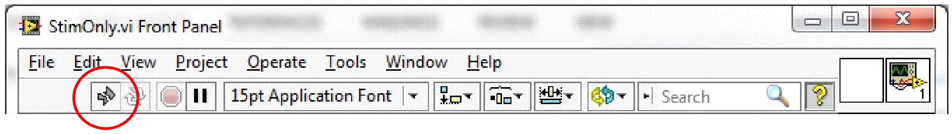


(wire connected to points only)

For other functions, the data type returned by the function has changed. As an example, the PS_IsWaveformBalanced function used to return a Boolean value and now returns an integer value:

Current	int PS_IsWaveformBalanced(int StimN, int ChN, bool *balanced);
Former	bool PS_IsWaveformBalanced(int StimN, int ChN);

This type of change is likely to cause to LabVIEW to indicate that the VI calling the function is “broken”. LabVIEW indicates this by showing a broken run arrow (circled in red below) at the top of the VI:



Clicking on the broken arrow will provide information on what LabVIEW considers to be broken. Reroute the wiring surrounding the indicated VIs to correct the problems as required.

PlexStim™
Electrical Stimulator

Constant Current Stimulation System

User Guide

Plexon Inc
6500 Greenville Avenue, Suite 700
Dallas, Texas 75206
Telephone: +1 (214) 369-4957
Fax: +1 (214) 369-1775

www.plexon.com

Plexon Inc Proprietary

Document Number: STMMN0001b

Software Version 2

Date: January 2017