April 16, 2012

WinLTP Manual

Version, 2.01

by William W. Anderson, Ph.D.

WinLTP Ltd.
support@winltp.com

and

MRC Centre for Synaptic Plasticity
University of Bristol
Bristol BS8 1TD, England
w.w.anderson@bristol.ac.uk
Tel: 0117-331-3063

Table of Contents

CHAPTER 1 – Introduction

1.1 WinLTP Capabilities

WinLTP is a stimulation, data acquisition and on-line analysis program for studying Long-Term Potentiation (LTP), Long-Term Depression (LTD) and other synaptic events such as epileptiform bursts. WinLTP records synaptic activity in extracellular, current-clamp or voltage-clamp modes (at up to 40 KHz/channel). WinLTP is a multitasking program that can run up to three tasks simultaneously, 1) repetitive Stimulation/Acquisition Sweeps (similar to the DOS LTP Program Anderson and Collingridge, 2001), and 2) Continuous Acquisition 'tape recording' (Fig. 1.1.1). In addition, the Stimulation/Acquisition Sweeps task now has a Protocol Builder to produce complex protocols, changeable at run time.

WinLTP runs on Windows PCI bus computers and uses National Instruments PCI/PCIe M-or X-Series boards and Axon Instruments' Digidata 1320A and 1322A data acquisition boards. Other software that can use the X-Series and M-Series boards includes WaveMetrics' IGOR, National Instruments' LabView, John Dempster's Strathclyde Electrophysiology Suite (WinWCP and WinEDR), and Silver lab's Nclamp.

The basic design philosophy behind WinLTP is to provide ever more complex stimulation protocols using the Protocol Builder, do sufficient online analysis to let you modify the experiment as it runs (such as changing baselines or protocol flow), do simultaneous continuous acquisition, (eventually) do simultaneous capture and analysis of spontaneous events, and use inexpensive but excellent data acquisition boards (National Instruments M- and X-Series boards). WinLTP on/off-line analyses include basic analyses of synaptic potentials (Peak Amplitude, Latency, Slope, Area, Duration, Rise/Decay Time, Coastline, PopSpike Amplitude and Latency, Average Amplitude), and also Cell Input Resistance (Rm), and Patch Electrode Series Resistance (Rs). However, WinLTP is not designed to do every possible synaptic analysis, and instead is designed to work with other very good, and often inexpensive analysis programs. For example, additional synaptic event analyses such as synaptic exponential decay time can be done by separate analysis programs including Synaptosoft's MiniAnalysis and Christoph Schmidt-Hieber's StimFit32, and spontaneous synaptic events in continuous acquisition gap-free abf files can be analyzed by Bill Heitler's DataView, Synaptosoft's MiniAnalysis, Axograph Scientific's AxoGraphX and Molecular Device's ClampFit.

WinLTP was written with Borland C++ Builder using Win32 VCL components. The program is available at low or no cost from www.winltp.com.

More complex WinLTP functionality includes:

1. **Multitasking**
   a. Repeat sweep stimulation, acquisition and analysis
   b. Tape recorder (**Continuous Acquisition** of 2 AD channels down to 25 µsec sampling interval, saved to a gap-free Axon Binary File)
2. **Protocol Builder** for writing complex protocols.
3. **Automated perfusion control** for extracellular slice and patch-clamp experiments
3. Fast Repeat (LTD) Sweep Stimulation **with no time between sweeps**

4. On and off-line calculation and plotting of several waveform parameters including:
    a. DC Baseline
    b. Peak Amplitude
    c. Latency
    d. Slope and Maximum Slope
    e. Area
    f. Duration
    g. Rise Time
    h. Decay Time
    i. Coastline
    j. PopSpike Amplitude
    k. PopSpike Latency
    l. Average Amplitude
    m. Cell Input Resistance (Rm)
    n. Patch Electrode Series Resistance (Rs)
5. Analyze all S0- and S1-evoked postsynaptic responses in both AD channels in a sweep
6. Special analyses of trains including:
    a. Analyze every pulse in train, but use the baseline of the first pulse as the baseline for each pulse
    b. Analyze whole train by analyzing only first pulse in train but detecting whole train
    c. Analyze train using baseline of the first pulse and response of the last pulse
7. **Automatic blanking of stimulus artifacts** to allow accurate determination of peaks and areas in a train
8. Measurement of Patch Electrode Series Resistance (Rs) using Rs peak, or Rs single or **double exponential** curve fitting

Simpler WinLTP functionality includes:

1. WinLTP records synaptic activity in extracellular, intracellular or patch clamp modes
2. 2 AD channel acquisition (down to 25 usec sample interval)
3. Two extracellular stimulation outputs (S0 and S1)
4. Two simultaneous patch-clamp recordings using two analog outputs
5. Analog stimulation including analog trains and ramps (loop within loop stimulation)
6. Repetitive sweeps with simultaneous data acquisition (up to 1,000,000 samples and 100 sec) and stimulation (using two extracellular pathway stimulation, S0 and/or S1, and epoch-like digital and intracellular analog stimulation).
5. The protocols for the basic LTP experiment are either slow single pathway S0 stimulation, or slow alternating dual pathway (S0 then S1) stimulation.
6. The sweep data can be signal averaged and digitally filtered on-line and off-line
7. LTP induction can be produced by:
    a. Single train
    b. Repetitive train (theta burst stimulation)
    c. Primed burst stimulation (limited implementation)
8. LTD stimulation and analysis can be performed using fast repetitive sweeps (at up to 10 Hz) with no delay between sweeps, or a single sweep lasting several minutes for faster repetitive
9. Patch sealtest protocol implemented
10. Save an ADsweep graph as a **Windows Enhanced Metafile**
11. Save your Spreadsheet/ AmpFile data to an **Excel XLS file**
12. Reanalyze straight ASCII files (skip header)
13. Automatic data folder creation at start-up
14. WinLTP Reanalysis works on Macs with Intel processors
15. On-line acquisition runs in Windows XP, Vista and **Windows 7** (M- and X-Series boards) and Windows 2000 and XP (Digidata 132x boards)

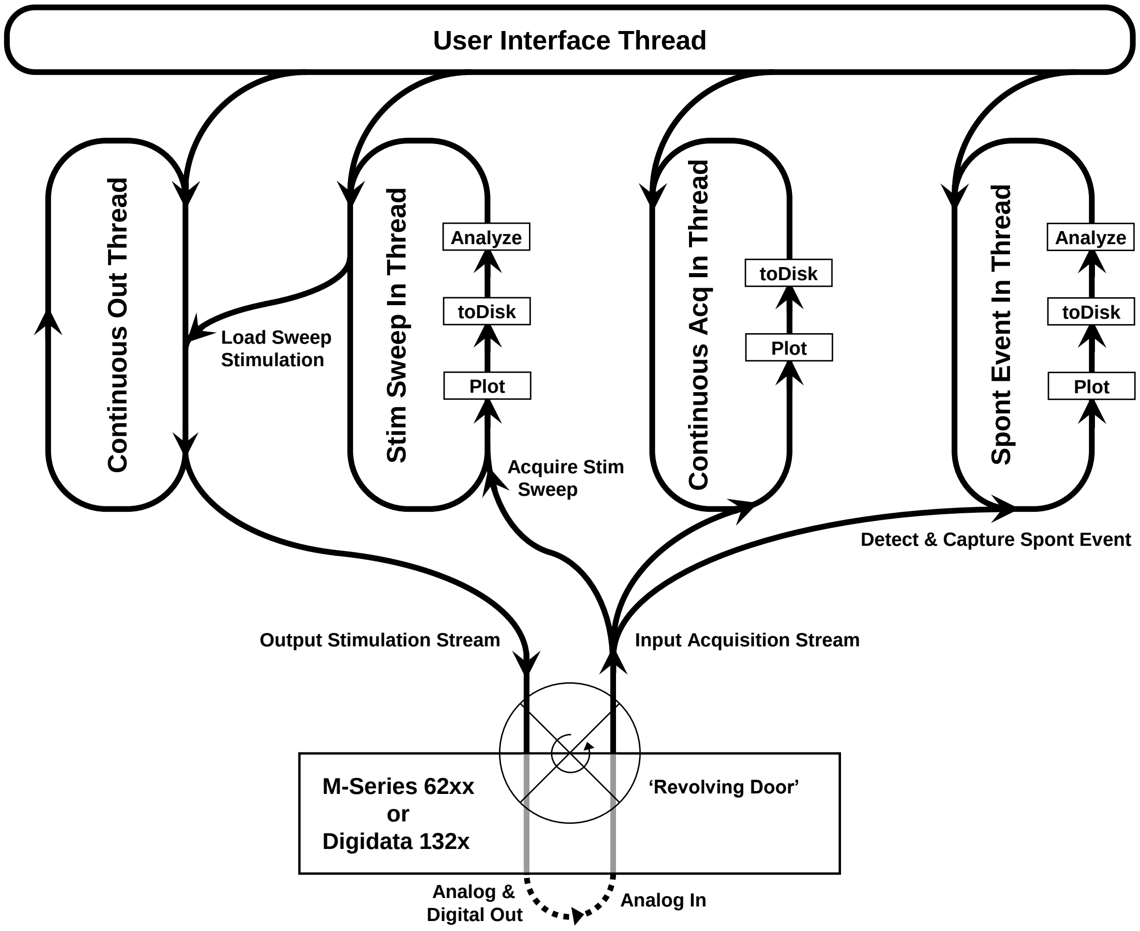


Fig. 1.1.1. WinLTP is a multitasking program that has three tasks - Stimulation/Acquisition Sweeps, Continuous Acquisition, and Capturing Spontaneous Events. These three tasks are produced by six processor threads – the Stim Sweep In and Continuous Out Threads produce the Stimulation/Acquisition Sweeps, the Continuous In Thread produces Continuous Acquisition, and the Spont Event Thread captures and analyzes Spontaneous Events. And for all tasks, the User Interface Thread captures user input and outputs screen graphics, and the Digidata Thread drives the Digidata 132x board.

As a user, I strongly think that if a program is any good, you shouldn't have to read the manual, just start using the program. On the other hand, as a programmer/manual writer, I know that when a user doesn't read the manual they are unaware of a lot of the program functionality. Because I can see both sides of the argument, I strongly recommend that you, at minimum, read the installation section Getting Started (Chapter 2) and Limitations To WinLTP (APPENDIX B) because without reading these sections you could be using the program incorrectly and giving you results that aren't what you think they are. Then scan the rest of the manual and look at the figures to see the other capabilities of the program. Then look at the Known Bugs in WinLTP (APPENDIX A). Hopefully after this you'll carefully read the rest of the manual (ha!).

1.2 Appropriate Equipment

1.2.1 Data Acquisition Boards

WinLTP currently uses the Axon **Digidata 132x boards** (the 1320A and 1322A), National Instruments **M-Series PCI, PCIe (PCIexpress) and USB 2.0 boards,** and also the newer National Instruments **X-Series PCIe and USB 2.0 boards.**

Importantly, the M- and X-Series boards have a 0.5 sec keyboard response delay (compared to 5.0 sec for the Digidata 132x boards).

If you are planning to use an National Instruments M- or X-Series board, see Section 2.3 for the appropriate AD board, cable and connector box to buy.

NOTE: there is currently a problem with X-Series PCIe and USB boards in that the MainProtocol stops running after 1hr 29min. This is due to a mistake in NIDAQmx code in versions 9.4 and earlier, and should be fixed shortly in version 9.5. In the meantime, there is a workaround solution using Protocol Linking (see Section 8.4).

1.2.2 Computers

1.2.2.1 Computer Makes

If I had to recommend one motherboard maker it would be INTEL as they are the processor maker and the ultimate decider on standards.

Basically, you should be able to use any Windows computer. The only current problem computers are **DELL Optiplex computers *with* RAID hard drives**, which do not work correctly with WinLTP. However, Dell Optiplex computers ***without*** **RAID** hard drives do work correctly. I would therefore **recommend staying away from computers using RAID hard drives.**

1.2.2.2 Bus

For the National Instruments M-Series board, WinLTP requires a computer with a PCI or PCIexpress bus. For the new National Instruments X-Series board, WinLTP requires a computer with a PCIe bus.

The USB 2.0 bus can also be used for M- and X-Series USB boards (see Section 2.3.1).

For the **Axon Digidata 132x board**, WinLTP requires a **PCI bus** computer or **laptop** computer (using a SCSI card inserted into the PCM-CIA slot).

1.2.2.3 Processor and Speed

For **National Instruments M- and X-Series boards**, WinLTP requires at least a computer with a **2.8 GHz** Intel Pentium 4 or higher speed processor **with HyperThreading**, the faster the better. (Note: some 2.93 and 3.06 GHz Pentium 4 processors do not have HyperThreading, and will not work with the M- or X-Series boards.)

For **Axon Digidata 132x boards**, WinLTP prefers a computer with at least a **3 GHz** or higher processor. However, in contrast to the M- and X-Series boards, the Digidata 132x boards can work with somewhat slower ca. 2 GHz computers (e.g. processors without HyperThreading), but this speed is really not recommended.

Both the M- and X-Series boards or the Digidata 132x boards have been tested on the Intel dual-core processors, and work well, and the M- and X-Series boards have been tested on the newer Intel quad-core processors, and work well.

The WinLTP Reanalysis program worked fine on all Intel Pentium 4 processors and above and is not limited to any minimum GHz processor speed.

1.2.2.4 RAM memory

When using either the M- or X-Series or Digidata board, at least **2048 MB** of RAM memory is recommended, although simple LTP experiments can probably be run on computers with a minimum of 1024 MB of RAM memory.

1.2.3 Monitors

WinLTP prefers monitors having **1280x1024** pixels or more, particularly if many AD channels are used, if the Protocol Builder is used, and/or if capturing spontaneous events is used. 1024x768 pixels is now too small – not all the perfusion channel information will be visible.

1.2.4 Operating Systems

For the M- and X-Series boards, WinLTP requires the **Windows XP, Windows Vista or Windows 7** operating system (which support HyperThreading and multi-core processors, Windows 2000 does not). However, Windows Vista has largely been bypassed by most users and institutions, and in our WinLTP tests, Windows 7 seems superior to Windows Vista. Windows Vista users should strongly consider upgrading to Windows 7.

When running on Windows 7, make sure the computer does NOT GO TO SLEEP. This will cause WinLTP to hang-up! To stop your computer from going to sleep change:

ControlPanel -> SmallIcons -> PowerOptions -> Balanced -> ChangePlanSettings

Put the computer to sleep [30 minutes] change to -> [Never]

Note that the automatic turning off of the display does NOT cause problems.

For the Axon Digidata 132x board, WinLTP requires the **Windows 2000 or XP** operating system. My understanding is that Windows Vista and Windows 7 drivers are not available for the SCSI card used by the Digidata 132x board, and therefore we have not tested WinLTP with the Digidata 132x board on Vista or Windows 7. Apparently you can buy a very expensive SCSI card that will work with Windows Vista and 7, but we think you may as well buy a National Instruments M- or X-Series board instead. Therefore we do not support the Digidata 132x board on Windows Vista or 7.

1.3 Overview of WinLTP – Basic LTP/LTD Experiment

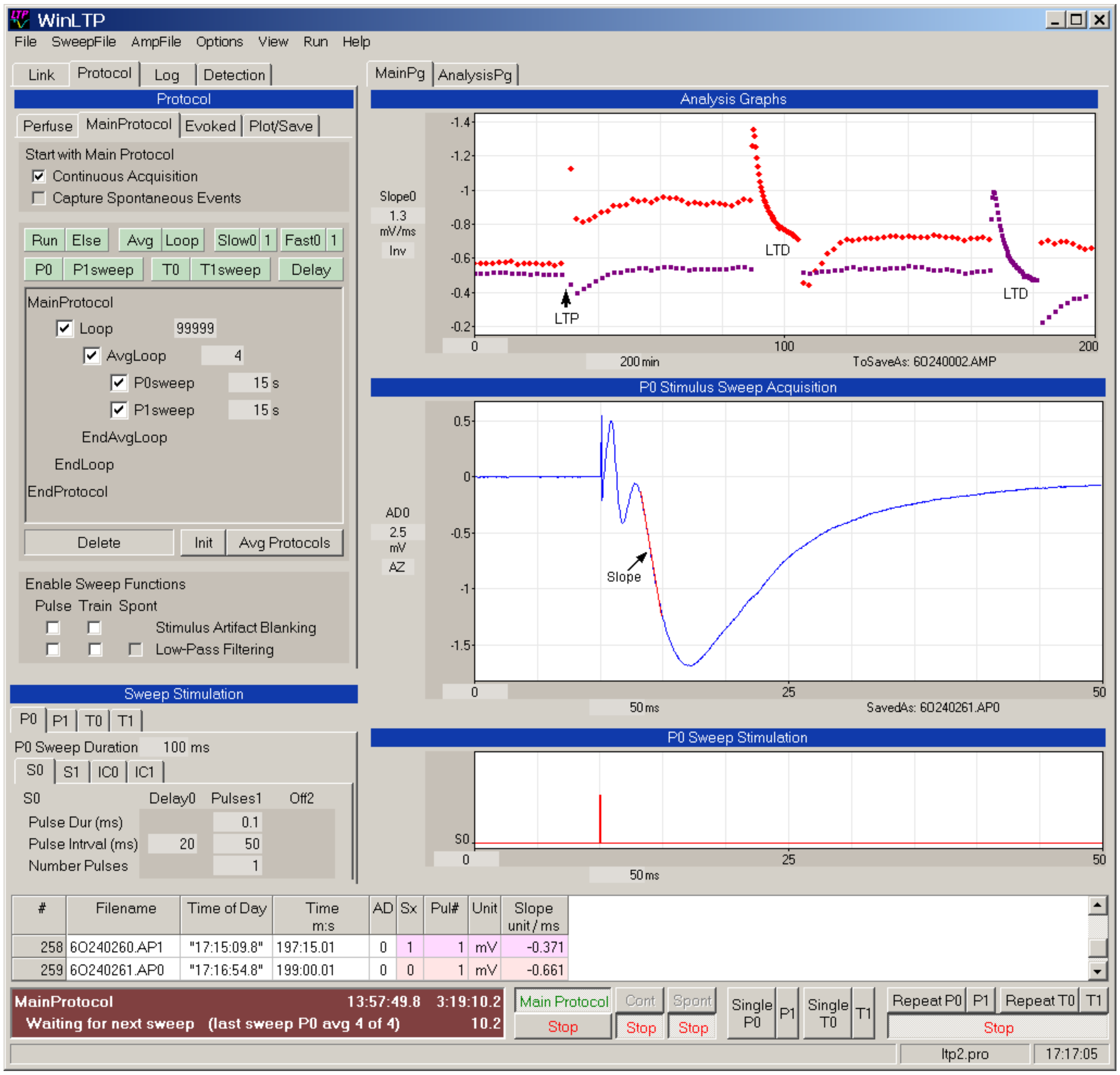
The protocols of WinLTP for running the basic LTP experiment consists of either repetitive slow single extracellular pathway stimulation by one electrode (S0), or slow alternating dual extracellular pathway stimulation by two electrodes (S0 then S1). Single train, theta burst, or primed burst stimulation induces LTP. Low frequency stimulation (e.g., 900 pulses at 1-2 Hz) induces LTD induction.

Fig. 1.3.1 shows the layout of the WinLTP program showing Protocol and Detection fields (upper left panel), Sweep Stimulation fields and graphs (lower left and right panels), Analysis graphs (one slope graph) (top right panel), Sweep Acquisition (middle right panel), and the Spreadsheet and Run Buttons (bottom panels).

Fig. 1.3.1 illustrates the basic LTP experiment of slow alternating dual pathway stimulation, in this case using signal averaging. The right middle graph shows an extracellular synaptic response from the CA1 region of the hippocampus, averaged from four sweeps, and produced by single extracellular S0 stimulus pulses, 10 ms from the start of the sweep. Superimposed on the synaptic waveform are red lines to show where calculations were made for the S0 slope. The slope 'calculation lines' are color-coded and are shown in red for an S0-evoked fEPSP and magenta for an S1-evoked fEPSP. The right top graph in Fig. 1.3.1 shows calculations for slope produced by S0 stimulation (red triangles) and S1 stimulation (magenta squares).

In WinLTP, alternating dual pathway stimulation (S0 then S1) of the experiment in Fig. 1.3.1 is achieved by producing dual alternating sweeps (Pulse Sweep P0 then Pulse Sweep P1) in which Pulse Sweep P0 has one pathway stimulation by one extracellular electrode, S0, and Pulse Sweep P1 has one pathway stimulation by extracellular electrode S1. The S0 and S1 stimulation outputs trigger stimulus isolation units that are connected to extracellular stimulation electrodes S0 and S1.

WinLTP is actually capable of generating four different sweep stimulations with different stimulation capabilities on each. Two sweep stimulations are Pulse Sweeps P0 and P1, and these are usually used for single pulse stimulation, can be repeated at set time intervals, and the sweep data can be signal averaged. The other two sweep stimulations are Train Sweeps T0 and T1, and these are evoked as single, nonrepetitive sweeps that are usually used for train stimulation.


WinLTP
File SweepFile AmpFile Options View Run Help
Link Protocol Log Detection
MainPg AnalysisPg
Protocol
Perfuse MainProtocol Evoked Plot/Save
Run Else Avg Loop Slow0 1 Fast0 1
P0 P1sweep T0 T1sweep Delay
Delete Init Avg Protocols
Analysis Graphs
Slope0
1.3
mV/ms
Inv
LTP
LTD
LTD
200 min
ToSaveAs: 60240002.AMP
P0 Stimulus Sweep Acquisition
AD0
2.5
mV
AZ
Slope
50 ms
SavedAs: 60240261.AP0
P0 Sweep Stimulation
S0
50 ms
Sweep Stimulation
P0 P1 T0 T1
S0 S1 IC0 IC1
Main Protocol Cont Spont
Stop Stop Stop
Single P0 P1 Single T0 T1
Repeat P0 P1 Repeat T0 T1
Stop

Start with Main Protocol
☑ Continuous Acquisition
☐ Capture Spontaneous Events

```
MainProtocol
    ☑ Loop            99999
        ☑ AvgLoop         4
            ☑ P0sweep       15 s
            ☑ P1sweep       15 s
        EndAvgLoop
    EndLoop
EndProtocol
```

Enable Sweep Functions

Pulse	Train	Spont	
☐	☐		Stimulus Artifact Blanking
☐	☐	☐	Low-Pass Filtering

P0 Sweep Duration 100 ms

S0	Delay0	Pulses1	Off2
Pulse Dur (ms)		0.1	
Pulse Intrval (ms)	20	50	
Number Pulses		1	

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	Slope unit / ms
258	60240260.AP1	"17:15:09.8"	197:15.01	0	1	1	mV	-0.371
259	60240261.AP0	"17:16:54.8"	199:00.01	0	0	1	mV	-0.661

MainProtocol 13:57:49.8 3:19:10.2
Waiting for next sweep (last sweep P0 avg 4 of 4) 10.2

ltp2.pro 17:17:05

Fig. 1.3.1. WinLTP layout for a basic LTP/LTD experiment showing the Protocol fields (upper left panel), analysis graphs (in this case only one slope graph, top right panel), Sweep Acquisition (middle right panel), Sweep Stimulation fields and graphs (lower left and right panels), and the Spreadsheet and Run Buttons (bottom panels). Detection fields to change synaptic potential detection values are hidden behind.

The MainProtocol panel shows the alternating P0sweep every 30 sec, then P1sweep every 30 sec, in an AvgLoop of 4 to produce an average every 4 sweeps.

Pulse Sweep P0 stimulation producing only one S0 extracellular stimulation pulse (left lower panel and the stimulation graph in the right lower panel). With this simple stimulation, rapid repeating of Pulse Sweep P0 at 1/sec produces rapid 900 pulse LTD S0 stimulation at the same frequency.

The Sweep Acquisition graph shows a fEPSP evoked by S0 stimulation (10 ms after the start of the sweep, averaged from 4 sweeps) with a red line showing the slope. The slope graph shows calculations of slope for S0-evoked fEPSPs (red triangles) and S1-evoked fEPSPs (magenta squares) caused by alternating S0/S1 pathway stimulation produced by alternating P0/P1 sweeps. Calculations are normally made on 4 averaged sweeps, but 20 averaged sweeps are used during LTD stimulation. Numerical values for slope ("Slope") are also shown on the spreadsheet panel below the graphs.

In the spreadsheet, "Time of Day" shows the time the sweep began, "Time m:s" shows the time of the stimulus pulse from when analysis starts, "Sx" shows whether S0 or S1 stimulation was used to evoke the synaptic response, "Pul#" shows the number of the S0 or S1 pulse that evokes the synaptic response, and "Slope" shows the calculated slope of the evoked response.

The induction of LTP by S0 stimulation (indicated by 'LTP' and up arrow below red triangles in the right top panel of Fig 1.3.1) is produced by evoking by clicking the ‘Single T0’ Run Button which produces a single Train Sweep of 100 S0 pulses at 100 Hz (not shown). The induction of LTD by S0 stimulation (indicated by 'LTD' below red triangles in the top panel of Fig. 1.3.1) is initiated by clicking the ’Repeat P0’ Run Button which produces rapidly repeating Pulse P0 Sweeps for a set number of times (900 here), once a second here. Since the Pulse P0 Sweep produces only 1 S0 pulse per sweep, this generates 900 S0 pulses at 1 Hz. (the LTD stimulation fields are actually shown in the Protocol Panel, **EvokedEvents** tabsheet.

1.4 Technical Support

Technical support can be obtained by directly contacting the author at WinLTP Ltd.:

Dr. William W. Anderson
Email: support@winltp.com
Tel: 0117-331-1968 (from outside the UK dial +44-117-331-1968)

Alternatively, if I do not respond in 1 or 2 days, then please contact my WinLTP Ltd. colleague Steve Fitzjohn at Fitzjohn_Stephen_NonLilly@Lilly.com.

1.5 Acknowledgements

I would like to thank my colleagues in the Bashir, Bortollotto, Cho and Collingridge labs for help with the testing of WinLTP, for both putting up with crashes, and rewriting your protocol files for yet another ("The Last" ha!) version. In particular, I would like to thank Mascia Amici, Douglas Caruana, Paul Farrow, Jihoon Jo, Minos Kritikos, Sarah Lucas, Stephane Peineau, Patrick Tidball and Steve Fitzjohn. I would also thank Tim Benke for giving me his exponential curve fitting C code, and John Dempster for helping me get started with Borland C++ Builder Windows data acquisition programming. And finally, I would like to thank Graham Collingridge for providing the support for WinLTP, without which it would not exist.

1.6 Conditions of Use

CONDITIONS FOR USING WinLTP

At the sole discretion WinLTP Ltd., academic users can freely run WinLTP Acquisition in the Basic Mode. However, academic users must purchase an Advanced Mode license to run WinLTP Acquisition in the Advanced Mode after the initial Demotrial period. Commercial users must purchase an Advanced Mode license to run WinLTP Acquisition in both the Basic and Advanced Modes. All users can freely run the WinLTP Reanalysis program for all reanalysis functions. However, some Reanalysis program file conversion functions will require a Advanced Mode license key.

WARRANTY AND LIMITATIONS FOR USING THIS SOFTWARE

1. Ownership of Software
   a. WinLTP Ltd. and the University of Bristol have shared copyright ownership of this software. The user is granted a nonexclusive license to use this software, but does not own it.

2. Warranty and Liability
   a. There is NO WARRANTY implied concerning the fitness of this software or documentation for the user's purpose. The software and documentation is supplied 'AS FOUND'.
   b. This is experimental research software. As such it has been well tested in our research group and many groups around the world, and is largely bug-free in these particular experiments. However, it is solely up to the user to determine if this software is suitable for his or her purpose, to determine if its limitations are acceptable, and to test that it is operating correctly.
   c. In particular, WinLTP Ltd. disclaims all liability for any direct, indirect, consequential or indirect damages resulting from the use of the program, including the costs of rectifying incorrectly obtained research results.
   d. For users running WinLTP Acquisition in the free Basic Mode or during the Demotrial Period, and running the free WinLTP Reanalysis reanalysis functions, the maximum liability to WinLTP Ltd. is zero.

3. Support
   a. Although WinLTP Ltd. will strenuously try to fix all bugs for the foreseeable future, it cannot promise when they will be fixed (or whether they will actually be fixed, particularly if they occur intermittently on a remote site computer).

4. Animal Research Guidelines and Human Subjects
   a. This software can only be used in research that meets the Society for Neuroscience guidelines for animal research.
   b. It is forbidden to use this software on HUMAN subjects.

ADDITIONAL WARRANTY AND LIMITATIONS FOR USING THIS SOFTWARE IN THE ADVANCED MODE

1. Warranty and Liability
   a. For users running WinLTP Acquisition in the Advanced Mode using a purchased Advanced Mode license, the maximum liability to WinLTP Ltd. is the cost of the WinLTP Advanced Mode license purchased.

2. Sale of Advanced Mode License
   a. At the sole discretion WinLTP Ltd., academic users can freely run WinLTP Acquisition in the Basic Mode. However, academic users must purchase an Advanced Mode license to run WinLTP Acquisition in the Advanced Mode after the initial Demotrial period. Commercial users must purchase an Advanced Mode license to run WinLTP Acquisition in both the Basic and Advanced Modes. All users can freely run the WinLTP Reanalysis program for all reanalysis functions. However, some Reanalysis program file conversion functions will require a Advanced Mode license key.
   b. Currently, there are no plans for to charge in future versions of WinLTP for running in the Basic Mode. However future changes to this policy cannot be eliminated.

3. Support
   a. WinLTP Ltd. will provide continuing support on how to experimentally use the WinLTP program for the foreseeable future. However, the promise of this experimental support by WinLTP Ltd. is limited to 1 year after purchase.

4. Upgrade Policy
   a. Minor upgrades and bug fixes for Advanced Mode functions will be free to the user. Whether to charge for major upgrades of WinLTP will be up to WinLTP Ltd.

5. Restrictions on Use
   a. You may not attempt in any way to overcome the copy protection mechanisms for the Advanced Mode of WinLTP Acquisition, or the file conversion routines of WinLTP Reanalysis. You are **not allowed** to:
      1) Reverse engineer, debug or decompile WinLTP or the License Key.
      2) Clock back the computer date or reinstall the program in an attempt to overcome the copy protection.
      3) Distribute your License Key outside your lab group.
   b. WinLTP may not be resold or included with any other product without written permission from WinLTP Ltd.

REQUESTS

1. If you are running WinLTP Acquisition in the Basic or Advanced Mode, I would appreciate it if you would reference using this software by name (e.g. as WinLTP).

2. If you are running WinLTP Acquisition in the free Basic Mode, I would also appreciate it if you would reference the WinLTP paper:

Anderson WW and Collingridge GL (2007) Capabilities of the WinLTP data acquisition program extending beyond basic LTP experimental functions. *J. Neurosci. Meth.,* 162:346-356.

3. BUGS: I would appreciate it if you would let me know of any bugs you have found. If you don't tell me, they won't get fixed!

CHAPTER 2 – Getting Started

2.1 Upgrade notice

If you are upgrading from earlier versions of WinLTP, you may have to write new *.pro protocol files. The newer WinLTP program will not load protocol files you made using a previous WinLTP if the protocol file size has changed. However,

Note that the WinLTP Reanalysis program can analyze the same ADsweep files make with all earlier versions of WinLTP and also the earlier DOS LTP Program.

2.2 Install WinLTP

Install WinLTP by running Install_WinLTP200.exe. By default, WinLTP is installed in the folder C:\WinLTP. Two Acquisition programs will be installed, one to run the Digidata 1322x boards (still WinLTPd111.exe) and one to run the M- and X-Series boards (WinLTPm200.exe), and a Reanalysis program (still WinLTPr111.exe) will be installed.

2.3 Install the National Instrument PCI M- or X-Series board

NOTE: The M- and X-Series PCI, PCIe (PCIexpress) and USB 2.0 boards have a 0.5 sec keyboard response delay (compared to 5.0 sec for the Digidata 132x boards).

2.3.1 M- and X-Series Data Acquisition Boards

Bottom Line – what NI stuff should I buy?

The most common and cost effective configuration currently is:

1) AD board:	**PCIe-6321**
2) Cable:	**SHC68-68-EPM cable, 2 meters**
3) Connector Box:	**BNC-2110**

This assumes you:

1) Have a computer with a PCIe slot available
2) Need only 2 output channels (that's all WinLTP2.01 supports but will increase to 4 in future WinLTP releases)
3) Need only 8 high-speed digital outputs (that's all WinLTP 2.01 uses for now)
4) If you are using WinLTP for automated extracellular slice perfusion, the BNC-2110 connector box does not have the P2.7 (PFI 15) digital output, so you can't use channel 16 with the standard

16-channel perfusion system system, or channel 8 with the pre-flush 8-channel system (see Section 9.2.4). A BNC-2090A may be a better choice then.

Recently, National Instruments has introduced the X-Series of boards along side the M-Series boards. The X-series boards are slightly higher performance than the M-Series boards. I thought the PCI and PCIe M-Series boards were/are great. The X-Series PCIe boards may be even slightly better. Note that the PCIe bus is twice as fast as the PCI bus, so if your computer has a PCIe slot (and most do), it is better to buy a PCIe board. More importantly, some computers are now down to only 1 PCI slot, and may soon go down to none, so you will have to buy a PCIe board. WinLTP 2.01 also supports USB 2.0 M- and X-Series boards. Athough the USB boards with BNC connectors are more expensive than the PCI/PCIe boards and BNC connector boxes, the portablility and ability to use with laptops may favor their purchase. Also, the USB boards with screw teminals are close to comparable cost of the PCI/PCIe boards and connector boxes.

NOTE: There is currently a problem with X-Series PCIe and USB board in that the MainProtocol stops running after 1hr 29min. This is due to a mistake in NIDAQmx code in versions 9.4 and earlier, and **should** be fixed shortly in version 9.5. In the meantime, there is a workaround solution using Protocol Linking (see Section 8.4).

PCI boards to consider include these M-Series boards:

AD board	Price*	#AnalogInChs	AnalogInSpeed	#AnalogOutChs	#HighSpeedDigOutChs
PCI-6221	$557	16**	250 kSamples/sec	2	8
PCI-6229	$746	32**	250 kSamples/sec	4	32
PCI-6251	$971	16**	1 MSamples/sec	2	8
PCI-6259	$1304	32**	1 MSamples/sec	4	32

PCIe boards to consider include these X-Series boards.

AD board	Price*	#AnalogInChs	AnalogInSpeed	#AnalogOutChs	#HighSpeedDigOutChs
PCIe-6321	$557	16	250 kSamples/sec	2	8
PCIe-6323	$746	32	250 kSamples/sec	4	32
PCIe-6351	$971	16	1 MSamples/sec	2	8
PCIe-6353	$1304	32	1 MSamples/sec	4	32

If you wish to buy a USB 2.0 board and want to have BNC connectivity you can purchase the relatively pricey USB-6221-BNC or USB-6229-BNC M-Series boards, or for screw terminal connectivity the more reasonably priced USB-6341 or USB-6343:

AD board	Price*	#AnalogInChs	AnalogInSpeed	#AnalogOutChs	#HighSpeedDigOutChs
USB-6341	$1061	16**	500 kSamples/sec	2	8
USB-6343	$1394	32**	500 kSamples/sec	4	32
USB-6221-BNC	$1574	8***	250 kSamples/sec	2	8
USB-6229-BNC	$1943	16***	250 kSamples/sec	4	32

* Prices are for October, 2011, are in US dollars, and include a 10% academic discount in units of 1. There is also a 25% academic discount if buying in units of 5.

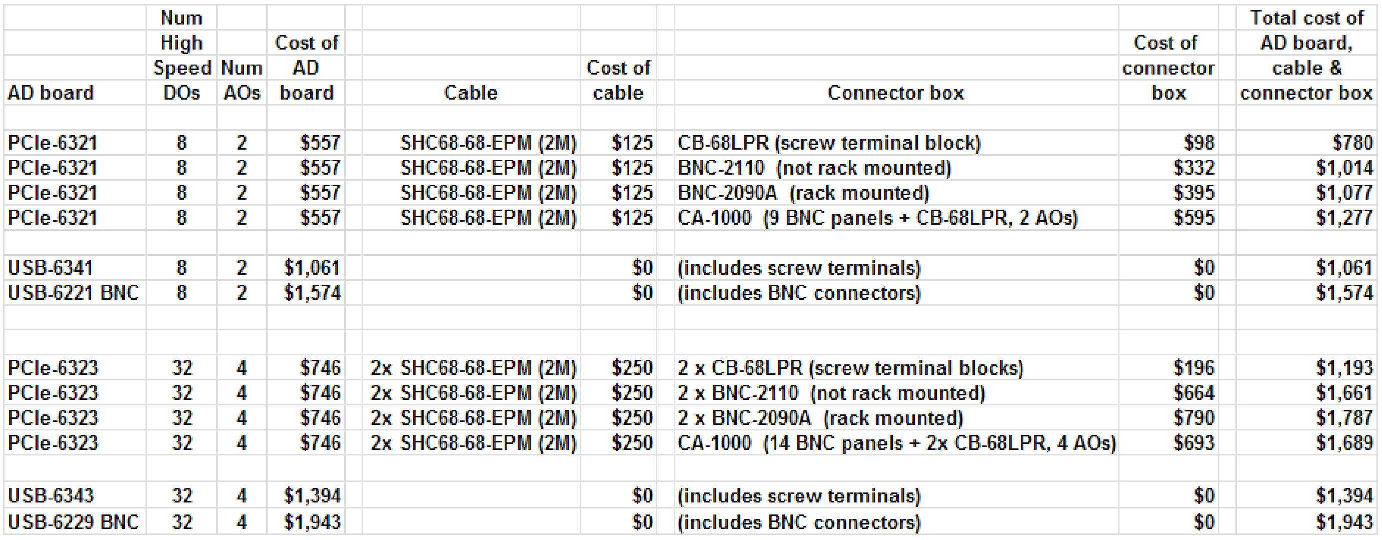
** The #AnalogInChs are for Non-Referenced Single Ended (NRSE) or Referenced Single-Ended (RSE) recording modes. For Differential recording mode divide this analog input channel number by two.

** Differential recording channels.

Note that the current 2.01 version of WinLTP samples at 200 kSamples/sec, so there is need to necessarily buy 500 kSamples/sec or 1 MSamples/sec boards. Future versions of WinLTP may utilize 1 MSamples/second capability, but this is not guaranteed.

I do not recommend buying the PCI-6221 (37 pin) version because it has only 2 rather than 8 high speed digital outputs, and is therefore only sufficient for S0 and S1, but nothing else such as the 4 digital outputs or the additional extracellular outputs in WinLTP 2.01 for automated perfusion control.

Fig. 2.3.1.2 shows the total cost for National Instrument PCIe data acquisition boards, cables and connector boxes, and for USB boards for price comparison. Also incuded are CB-68LPR screw terminal connector blocks for the absolute minimum cost of PCIe-6221 and PCIe-6223's boards and connectors, and for direct comparison with the USB-6341 and USB-6343 which use screw terminals.



AD board	Num High Speed DOs	Num AOs	Cost of AD board	Cable	Cost of cable	Connector box	Cost of connector box	Total cost of AD board, cable & connector box
PCIe-6321	8	2	$557	SHC68-68-EPM (2M)	$125	CB-68LPR (screw terminal block)	$98	$780
PCIe-6321	8	2	$557	SHC68-68-EPM (2M)	$125	BNC-2110 (not rack mounted)	$332	$1,014
PCIe-6321	8	2	$557	SHC68-68-EPM (2M)	$125	BNC-2090A (rack mounted)	$395	$1,077
PCIe-6321	8	2	$557	SHC68-68-EPM (2M)	$125	CA-1000 (9 BNC panels + CB-68LPR, 2 AOs)	$595	$1,277
USB-6341	8	2	$1,061		$0	(includes screw terminals)	$0	$1,061
USB-6221 BNC	8	2	$1,574		$0	(includes BNC connectors)	$0	$1,574
PCIe-6323	32	4	$746	2x SHC68-68-EPM (2M)	$250	2 x CB-68LPR (screw terminal blocks)	$196	$1,193
PCIe-6323	32	4	$746	2x SHC68-68-EPM (2M)	$250	2 x BNC-2110 (not rack mounted)	$664	$1,661
PCIe-6323	32	4	$746	2x SHC68-68-EPM (2M)	$250	2 x BNC-2090A (rack mounted)	$790	$1,787
PCIe-6323	32	4	$746	2x SHC68-68-EPM (2M)	$250	CA-1000 (14 BNC panels + 2x CB-68LPR, 4 AOs)	$693	$1,689
USB-6343	32	4	$1,394		$0	(includes screw terminals)	$0	$1,394
USB-6229 BNC	32	4	$1,943		$0	(includes BNC connectors)	$0	$1,943

Fig 2.3.1.2. Commonly used configurations of PCIe data acquisiton boards, cables and connector boxes, and two USB BNC boards for comparison. Prices include a 10% academic discount for units of 1. There is also a 25% academic discount if buying in units of 5. Prices are for October 2011.

2.3.2. Cables

All PCI M- and X-Series boards SHC68-68-EPM cable. It is available in 1, 2 or 3 meter lengths.

2.3.3 Connector Boxes

All PCI M- and X-Series boards with 2 analog outputs such as the PCI-6221, PCI-6251, PCIe-6321 and PCIe-6351 require using either the BNC-2110 or BNC-2120 for non-rack mounting, the BNC-2090A for rack mounting, or the CA-1000 enclosure (with BNC panelettes and one CB-68LPR connector block).

All PCI M- and X-Series boards with 4 analog outputs such as the PCI-6229, PCI-6259, PCIe-6323 and PCIe-6353 require either two BNC-2110's, two BNC-2120's, two BNC-2090A's or using the CA-1000 enclosure (with BNC panelettes and two CB-68LPR connector blocks). (Note however that the four analog outputs and > 8 high-speed digital outputs are not supported with WinLTP111 but will be in subsequent versions – so there is no need to buy the second cable and additional connections now.)

You could also use other connector boxes such as the SCB-68 Shielded Connector Box, but there is little point, as it is about the same cost as the others, and only has screw terminal outputs.

NOTE: WinLTP110 and earlier was only set to run in the NRSE (Non-Referenced Single-Ended) mode. WinLTP111 can now be set up in Differential and RSE (Referenced Single-Ended) modes as well as the NRSE mode.

When using M- or X-Series board and connector boxes, the recording mode must be set on program startup to either Differential, Non-Referenced Single-Ended (NRSE) or Referenced Single-Ended (RSE) as described in Section 2.7. For certain boxes, such as the BNC-2110 and the BNC-2120, the preferred recording mode (Differential in these cases) is already chosen. For other connector boxes such as the BNC-2090A or the CA-1000 connector box with All Pins Available, you can choose Differential, NRSE or RSE. If you wash to use the BNC-2110 or BNC-2120 in NRSE or RSE mode, choose the 'Connector box with All Pins Available'.

Also, for some connector boxes, such as the BNC-2110 and the BNC-2120, each channel in the connector box can be set to record from Floating Source (FS) Signals such as battery powered biological amplifiers NOT connected to mains ground, or from Ground Source (GS) Signals such as biological amplifiers connected to mains ground to connector boxes.

For information on connecting Floating Source (FS) Signals and Ground Source (GS) Signal and using Differential, NRSE or RSE recording modes, see Fig. 2.3.3.1. Also see the detailed discussion in Field Wiring and Noise Considerations for Analog Signals – both from the National Instruments website.

NOTE: Even though the Resources tabsheet is specifically set up for BNC-2110, BNC-2120 and BNC-2090A connector boxes, if you choose the "Connector Box with All Pins Available", you can set up connections in any recording mode to any connector box. This includes a USB M-Series board and box, a CA-1000 box with your own custom connections, or even the BNC-2110, BNC-2120 boxes in any recording mode.

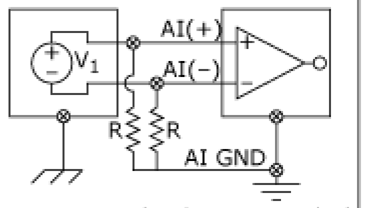
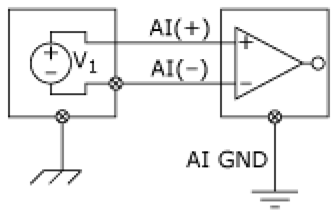
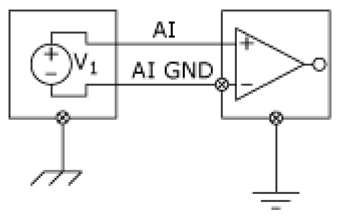
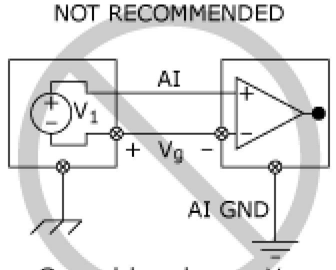
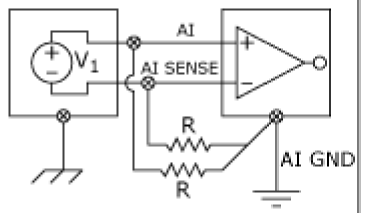
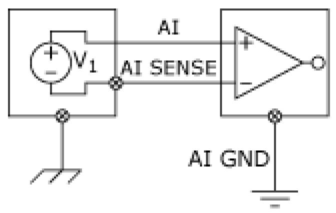
Input Configuration	Signal Source Type	
	Floating Signal Source (Not Connected to Building Ground)	**Grounded Signal Source**
	Examples • Thermocouples • Signal Conditioning with Isolated Outputs • Battery Devices	Examples • Plug-in Instruments with Nonisolated Inputs
Differential (DIFF)	 Two resistors (10 kΩ<R<100 kΩ) provide return paths to ground for bias currents	
Single-Ended – Ground Referenced (RSE)		NOT RECOMMENDED  Ground-loop losses, V_g, are added to measured signal.
Single-Ended – Nonreferenced (NRSE)		

Fig. 2.3.3.1. Making measurements in Differential, Non-Referenced Single-Ended (NRSE), and Referenced Single-Ended mode (RSE) modes. Use Differential, NRSE or RSE mode to connect a Floating Signal (FS) Source such as a battery powered biological amplifier NOT connected to mains ground. Usually use Differential or NRSE to connect a Grounded Signal (GS) Source such as a biological amplifier connected to mains ground (Copyright National Instruments).

Use Differential, NRSE or RSE mode to connect a Floating Signal (FS) Source such as a battery powered biological amplifier NOT connected to mains ground. Usually use Differential or NRSE to connect a Grounded Signal (GS) Source such as a biological amplifier connected to mains ground (Copyright National Instruments).

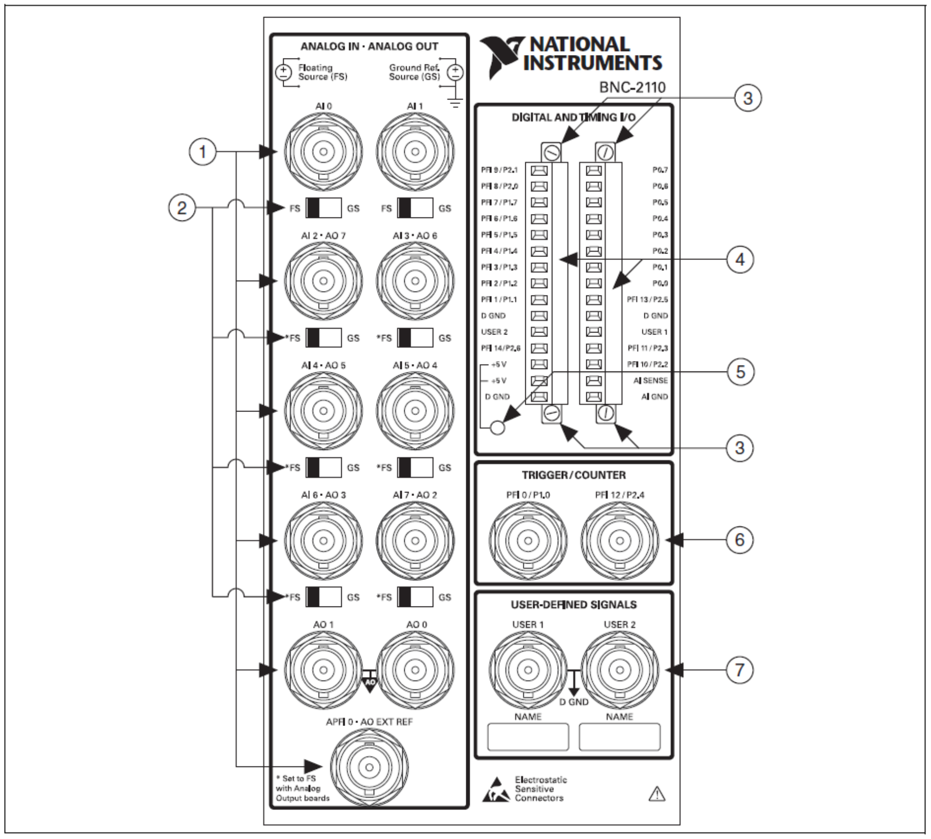
2.3.3.1 BNC-2110 Connector Box

The BNC-2110 (Fig. 3.3.1.1) is the smallest and simplest connector box to install.

Analog Inputs 0 and 1 are used as normal (plugged into with a shielded BNC cable) and are recorded in Differential mode (see Section 2.7 and Fig. 2.7.1). Thus for a signal from either a battery powered biological amplifier or a mains powered and grounded biological amplifier, the center pin signal and the BNC shield are recorded differentially. You do NOT have to connect anything to the AI SENSE or AI GND pins.

For a Floating Signal (FS) Source such as a signal from a battery powered biological amplifier, switch the FS/GS switch to FS. For a Ground Signal (GS) Source such as a signal from a mains powered and grounded biological amplifier, switch the FS/GS switch to GS.

Analog Outputs 0 and 1 are used as normal (plugged into with a shielded BNC cable).



1	Analog Input/Analog Output BNC Connectors	5	Power Indicator Light
2	FS/GS Switches	6	Trigger/Counter BNC Connectors
3	Terminal Block Retaining Screws	7	User-Defined Signals BNC Connectors
4	Digital and Timing I/O Spring Terminal Blocks		

Fig. 3.3.1.1. Photograph of the BNC-2110 (left), and the AI (AnalogIn), AO (AnalogOut), and User1 and User2 BNC connectors, the FS/GS switches for each analog input channel, and the pin connections for 8 high-speed digitial outputs (P0.0 to P0.7) (right). For ease of use, a wire is connected from the P0.0 pin to the User1 pin, and from P0.1 pin to the User2 pin so that the Stimulus Isolation Unit for signal S0 can be connected by a BNC cable to User1, and the Stimulus Isolation Unit for signal S1 can be connected by a BNC cable to User2. (Copyright National Instruments)

To use the extracellular stimulation outputs, connect a wire from pin P0.0 to User1 (so that the User1 BNC can be S0 output), and connect a wire from pin P0.1 to User2 (so that the User2 BNC can be S1 output).

To use the four Digital Outputs (D2 to D5), directly connect wires to the P0.2 to P0.5, respectively.

2.3.3.2 BNC-2120 Connector Box

NOTE: In WinLTP 1.10 and earlier, the BNC-2120 was used in NRSE recording mode and required making connections to AI SENSE and AI GND pins. In WinLTP 1.10 onward the default is now the Differential recording mode, and no connections need to be made to AI SENSE or AI GND pins. If AI SENSE is still connected to AI GND after upgrading from WinLTP 1.11, this conntection need not be removed. AI SENSE is not used in the Differential Mode. However, if your BNC shield is connected to AI SENSE or AI GND, you should disconnect it to get a true differential recording between the cable center pin and cable shield.

We actually now favour the BNC-2110 Connector Box over the similar BNC-2120 because it is smaller, cheaper and slightly simpler.

For the BNC-2120, Analog Inputs 0 and 1 are used as normal (plugged into with a shielded BNC cable) and are recorded in Differential mode (see Section 2.7 and Fig. 2.7.3). Thus for a signal from either a battery powered biological amplifier or a mains powered and grounded biological amplifier, the center pin signal and the BNC shield are recorded differentially.

For a Floating Signal (FS) Source such as a signal from a battery powered biological amplifier, switch the FS/GS switch to FS. For a Ground Signal (GS) Source such as a signal from a mains powered and grounded biological amplifier, switch the FS/GS switch to GS.

Analog Outputs 0 and 1 are used as normal (plugged into with a shielded BNC cable).

To use the extracellular stimulation outputs, connect a wire from pin P0.0 to User1 (so that the User1 BNC can be S0 output), and connect a wire from pin P0.1 to User2 (so that the User2 BNC can be S1 output).

To use the four Digital Outputs (D2 to D5), directly connect wires to the P0.2 to P0.5, respectively.

2.3.3.3 BNC-2090A Connector Box

NOTE: In WinLTP 1.10 and earlier, the BNC-2090A was used in NRSE recording mode and required setting the SE/DIFF switches to SI, and the RSE/NRSE to NRSE (Non-Referenced Single Ended). In WinLTP 1.11 onward the recording mode can now be Differential, NRSE or RSE depending what is chosen in the Choose Recording Mode radiobutton group (**Fig. 2.7.4**).

Analog Inputs 0 and 1 (ACH0 and ACH1) are used as normal (plugged into with a shielded BNC cable).

For recording in the **Differential** mode (Fig. 2.7.4, top), set the **SE/DIFF** switch to **DIFF**. It does not matter what the RSE/NRSE switch is set to because it is not used. Thus for a signal from either a battery powered biological amplifier or a mains powered and grounded biological amplifier, the center pin signal and the BNC shield are recorded differentially.

There is no Floating Signal (FS) Source / Ground Signal (GS) Source FS/GS switch in the BNC-2090A. If you are recording in **Differential** mode from a **Floating Signal Source** you will have to have a 10–100 kΩ resistor to AI GND on one input if DC-coupled, or both inputs if AC-coupled (see the BNC-29090A User Manual for more information).

If you are recording in **Differential** mode from a **Ground Signal Source** you do not have to make any special connections – just use the factory default configuration (see the BNC-29090A User Manual for more information).

When recording in **Single-Ended** mode from a **Floating Signal Source**, configure the BNC-2090A to supply a ground reference by placing the device in **Referenced Single-Ended (RSE)** mode. For recording in the **Referenced Single-Ended (RSE)** mode (Fig. 2.7.4, lower right), set the **SE/DIFF** switch to **SE**, and set the **RSE/NRSE** switch to **RSE**.

When recording in **Single-Ended** mode from a **Ground Signal Source**, the external signal supplies its own reference ground point and the BNC-2090A should not supply one. Therefore, configure the BNC-2090A in the **Non-Referenced Single-Ended (NRSE)** mode (see the BNC-29090A User Manual for more information). For recording in the **Non-Referenced Single-Ended (NRSE)** mode (Fig. 2.7.4, lower left), set the **SE/DIFF** switch to **SE**, and set the **RSE/NRSE** switch to **NRSE**.

Analog Outputs 0 and 1 (DAC0 and DAC1) are used as normal (plugged into with a shielded BNC cable).

To use the extracellular stimulation outputs, connect a wire from pin DIO0 to User1 (so that the User1 BNC can be S0 output), and connect a wire from pin DIO1 to User2 (so that the User2 BNC can be S1 output).

To use the four Digital Outputs (D2 to D5), directly connect wires to DIO2 to DIO5, respectively.

2.3.3.4 CA-1000 Connector Box

For the current WinLTP program (version 2.01) using 2 analog inputs, 2 analog outputs, and 6 high-speed digital outputs, it makes little sense to put together a custom CA-1000 enclosure with a CB-68LPR connector block and wire up the BNC panelettes. This is particularly true since they recently doubled the cost of the BNC panelettes, making a wired up CA-1000 box not nearly as cost effective as it was several years ago (see Fig. 2.3.1.2).

However, future versions of WinLTP will include at least 5 analog inputs, 4 analog outputs, 8 to 24 high-speed digital outputs, and maybe 8 low-speed digitial outputs. For this you can use either two BNC-2110's, two BNC-2120's, two BNC-2090A's or using the CA-1000 enclosure (with BNC panelettes and two CB-68LPR connector blocks). So wiring up a CA-1000 is not completely unreasonable.

Installation is pretty easy and requires only a supplied screwdriver. Install 9 panelettes in front, and the CB-68LPR connector block in the back left or right position. Connect the wires as shown using the PCI-62xx data sheet pinout information.

If you are recording in **Differential** mode (Fig. 2.7.5, top), then you can:

1) connect the central pin of the AnalogInput 0 BNC to AI0 (pin 68), and
2) connect the shield wire of the BNC to AI8 (pin 34)

and so forth.

If you are recording in **Non-Referenced Single-Ended (NRSE)** mode (Fig. 2.7.5, lower left) (which was the recording mode of WinLTP110 and earlier), you can:

1) connect the central pin of the AnalogInput 0 BNC to AI0 (pin 68).

Then you can either:

2a) connect the shield wire of the BNC to AI SENSE (pin J62). This is to remove any ground loops and is theoretically the better way, or,

2b) connect AI GND (say pin J29) to AI SENSE (pin J62). Then connect the shield wire of the BNC to an AI GND pin (say J67). You may have to disconnect the grounded shield of the Analog Input 0 coaxial cable from the Analog Input 0 BNC ground to prevent a ground loop – alternatively in this situation leave the shield wire of the BNC unconnected).

2.3.4. Install NI-DAQmx 9.x

You first have to install NI-DAQmx from the CD included with your M- or X-Series board. For these boards, version 8.8 or higher must be used because of the code added to support USB M- and X-Series boards. If you need to upgrade, you might as well download and use the latest Version 9.4 from **www.ni.com**. NOTE: the NI-DAQmx driver file is over 1 GB, and can take over 2 hrs to download.

Because of the current problem with X-Series PCIe and USB board in that the MainProtocol stops running after 1hr 29min, which is due to a mistake in NIDAQmx code in versions 9.4 and earlier, if using an X-Series board, you should upgrade to 9.5 when it becomes available.

Follow the installation instructions. Although you can use the installation defaults, only the NI-DAQmx and NI Measurement and Automation Explorer must be installed, the Microsoft Visual C Support, the OPC Support, and the VI Logger are not required.

Then **restart** the computer and the **Found New Hardware wizard** will run. When asked about Windows connecting to Windows Update, respond

(*) No not at this time

Then choose

{*} Install the software automatically (recommended)

And the wizard should say that it has finished installing the software for the PCI-62xx board.

2.3.5. Run the Measurement and Automation Explorer (MAX) and test the board

Click on ‘Devices and Interfaces’, then click on ‘NI-DAQmx Devices’ and your M- or X-Series PCI or PCIe board should appear (such as ‘NI PCI-6221 “Dev1”’ or ‘NI PCIe-6321 “Dev1”’).

If the name of the device is not the default “Dev1”, perform a right mouse click and Rename the device to “Dev1”. It is essential that this device is called “Dev1”.

Then perform a right mouse click and run the Self-Test.

Then click on the Test Panels. (Make sure AO0 is connected to AI0.) Set Analog Output mode to Sinewave Generator, Transfer Mechanism to DMA, and click Start. Set Analog Input Mode to Continuous and Configuration to NRSE, and click Start. A sinewave should appear in the graph.

NOTE: **If there is a problem, first try moving the PCI M-Series board to another PCI slot** (I know this from experience!). Otherwise contact National Instruments support. Nothing in MAX depends on WinLTP, and WinLTP will not work if the testing in MAX doesn’t work.

2.3.6 Run more than one M- or X-Series board in one computer

In WinLTP111, we have added the capability of simultaneously running more than one M- or X-Series board in one computer. As far as WinLTP is concerned, you can theoretically install up to 9 National Instrument boards into one computer (“Dev1” to “Dev9”), and have up to 9 boards operating simultaneously. However, we have only tested simultaneously running up to 3 boards for now. The number of boards you can simultaneously run depends on the speed of your computer (it should at least be a dual-core), and the rate at which you run and save sweeps during an experiment (be sure to test beforehand).

To install a second M- or X-Series board in your computer perform the following steps:

1) Make sure the WinLTP program that uses the “Dev1” board is not running.

2) Install a second National Instruments M- or X-Series board into your computer. Run National Instruments Measurement and Automation Explorer program (MAX) and make sure that this second board is called **“Dev2”**. If it is not, then rename it in MAX.

3) Copy the contents of the \WinLTP folder into a second folder and rename the folder, say to
\WinLTP_Dev2
You do not need to copy any of the subfolders of \WinLTP.

4) Make a shortcut for the WinLTPm111.exe file in the \WinLTP_Dev2 folder, put it on the Desktop, and give it a new title of say,
WinLTP M,Xseries Dev2

5) Then run the WinLTP program that is to use the “Dev2” board. Once it has started up, use the menu commands (Fig. 3.2.4):
Option -> Set M,X-Series Board Device Number…
to call up the **Set M,X-Series Board Device Number** dialog box (Fig. 2.3.6.1).

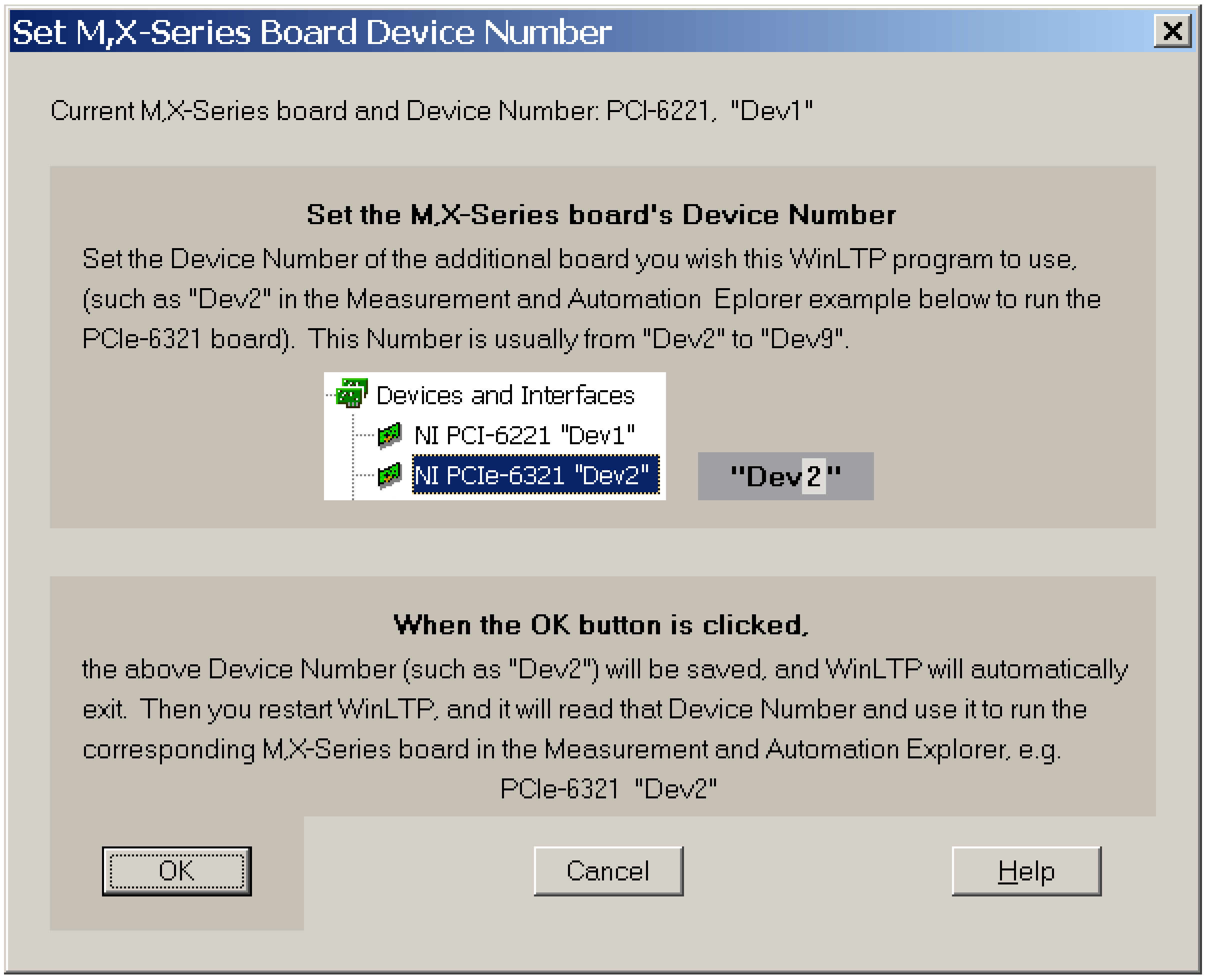
The current M,X-Series board and Device Number is presented at the top of the dialog box. - in the example in Fig. 2.3.6.1 it is:

PCI-6221, “Dev1”

Then you set the second M,X-Series board to the desired Device Number that is found in MAX, which in Fig. 2.3.6.1 is:
PCIe-6321, “Dev2”
so you would enter **2** into the “Dev ” field box.

Then you click the ‘OK’ button. This will automatically save the new Device Number (in Fig. 2.3.6.1 it is “Dev2”) to a “Mseries_DeviceNUmber.ini” file in the \WinLPT folder, and then WinLTP will **automatically exit**. Then **you have to manually restart WinLTP.** During startup, WinLTP will read the new Device Number from the “Mseries_DeviceNumber.ini” file, and use this Device Number to initialize WinLTP for the correct second board found in MAX (in the example in Fig. 2.3.6.1 its PCIe 6321).

WinLTP has to be manually restarted because WinLTP will only correctly initialize the new Device Number and board during startup.



Set M,X-Series Board Device Number

Current M,X-Series board and Device Number: PCI-6221, "Dev1"

Set the M,X-Series board's Device Number

Set the Device Number of the additional board you wish this WinLTP program to use, (such as "Dev2" in the Measurement and Automation Eplorer example below to run the PCIe-6321 board). This Number is usually from "Dev2" to "Dev9".

Devices and Interfaces
NI PCI-6221 "Dev1"
NI PCIe-6321 "Dev2"

"Dev 2"

When the OK button is clicked,

the above Device Number (such as "Dev2") will be saved, and WinLTP will automatically exit. Then you restart WinLTP, and it will read that Device Number and use it to run the corresponding M,X-Series board in the Measurement and Automation Explorer, e.g.
PCIe-6321 "Dev2"

Fig 2.3.6.1. The Set M,X-Series Board Device Number dialog box used to set the Device Number of an additional M- or X-Series board.

2.4 Install the Axon Digidata 1320A or 1322A board

NOTE: If you are installing just the WinLTP Reanalysis program, you do not have to first install AxoScope or pClamp, inotherwords (e.g., you can skip steps 2.4.1, 2.4.2 and 2.4.3).

Put the SCSI card into the computer and connect the Digidata 1320A or 1322A board to the SCSI card according to the instructions from Axon Instruments included with the board. Do not power up the board, but turn on the computer to see that the SCSI card is recognized.

2.4.1 Install AxoScope 9 (or pClamp 9)

Axon Instruments' AxoScope 9 (or pClamp 9) must be installed before WinLTP can be run because the WinLTP acquisition program uses one library file included only in AxoScope 9 (or pClamp 9) to run the Digidata 132x board (axdd132x.dll). AxoScope 9 is included with the Digidata 1322A board. If you have an older version of Axoscope obtained with the Digidata 1320A, contact Molecular Devices at www.moleculardevices.com. Molecular Devices has not given me permission to distribute this file with the WinLTP package.

NOTE: you must use axdd132x.dll from AxoScope (or pClamp) verision 9, NOT AxoScope (or pClamp) version 10 - axdd132x.dll from version 10 WILL NOT WORK!!!

2.4.2 Check that AxoScope 9 (or pClamp 9) is running correctly

It is advisable, although not required, that you check that AxoScope 9 (or pClamp 9) are running correctly. Turn off the computer. Turn on the Digidata 132x board. Then turn on the computer. NOTE: Always turn on the Digidata board before turning on the computer.

Test that AxoScope (or pClamp) recognizes the Digidata 132x board, and test that it is basically working correctly. Then exit AxoScope or pClamp.

2.4.3 Make the required Axon library file (axdd132x.dll) accessible to WinLTP

Copy the Axon axdd132x.dll library file from your AxoScope 9 (or pClamp 9) folder to the WinLTP folder. (Axon will not let me distribute this file with my WinLTP program, but this will not be a problem provided you have AxoScope 9 or pClamp 9.)

DO NOT INSTALLL axdd132x.dll from pClamp 10 or AxoScope 10 (or higher). It is guaranteed NOT TO WORK!!! WinLTP should issue a warning message if you try to do this.

2.5 Start WinLTP

Start WinLTP acquisition by clicking on either the 'WinLTP Digidata' icon to run the Digidata 132x board, or the 'WinLTP M,X-Series' to run the M- or X-Series board (Fig 2.5.1). You may as well delete the acquisition program icon you are not using.

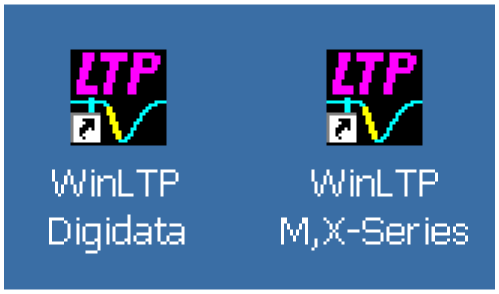


Fig 2.5.1.

When WinLTP starts up, the initial 'splash screen' comes up almost immediately indicating that the program in "Loading…", which takes at least 15 seconds. After this period the "Loading…" message goes away and the initial Data Root Folder is shown (Fig. 2.6.1).

NOTE: There may be a video related BUG with early versions of Windows XP. If WinLTP hangs up during start-up (start-up can take at least 15 seconds!), try changing your video to Classic (Windows 2000) mode.

NOTE: If you are trying to run an M or X-Series board, and you get an ERROR message saying that
"dynamic link library NICAIU.DLL" has not been found
this means that you either haven't installed NI-DAQmx on your computer.

If you get an error like:

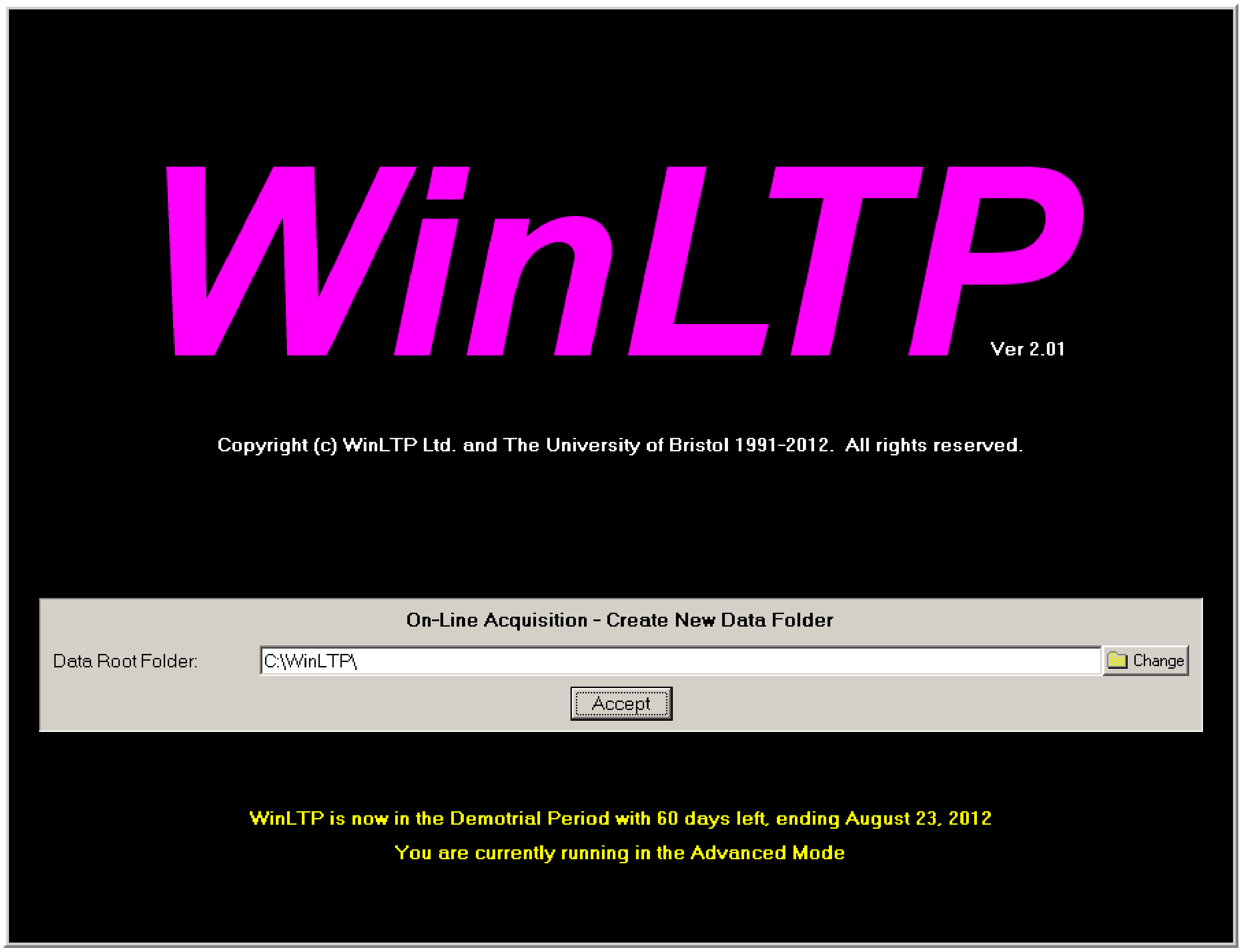
WinLTPm111.exe – Entry Point Not Found
The procedure entry point DAQmxGetAIUsbXferReqSize could not be located in the dynamic link library NICAIU.DLL

this means that you have an NI-DAQmx version earlier than 8.8 installed. You must install version 8.8 or later because of the code added to support USB M- and X-Series boards. You might as well download and use the latest Version 9.4 from www.ni.com. NOTE: the NI-DAQmx driver file is over 1 GB, and can take over 2 hrs to download (see Section 2.3.4).

2.6 Automatic Data Folder Creation at Start-up

The Data Root Folder is the folder off of which all subsequent data folders will be created. In Fig. 2.6.1, the data drive folder is C: and the data root folder is the folder containing the WinLTP program, \WinLTP. The WinLTP program folder C:\WinLTP is the default data root folder that is presented upon initial start-up. Therefore if today was June 24, 2010, then the data folder into which the *.P0 sweep and *.AMP files would be written would be C:\WinLTP\100624, and this data folder would be automatically created. The first two characters of the data folder denote decade and year, the second two characters denote month and the third two characters denote day of the month. With this method, folders are automatically sorted year/month/day when viewed with the Windows Explorer.

To change the Data Root Folder either type in a new one in the edit box, or click on the change button to bring up the Change Data Folder Dialog box (Fig. 2.6.2), and either choose a different existing Data Root Folder, or make a new one by clicking on the 'Make a New Folder' button. Once you click on the Accept button, the final Data Root Folder and the Data Read/Write Folder (which is the Data Root Folder plus a subfolder with today's data where data will be written) are created as shown in Fig. 2.6.3. You can also change the Data Read/Write Folder while running an experiment (Section 4.14).


WinLTP

Ver 2.01

Copyright (c) WinLTP Ltd. and The University of Bristol 1991-2012. All rights reserved.

On-Line Acquisition - Create New Data Folder

Data Root Folder: C:\WinLTP\

WinLTP is now in the Demotrial Period with 60 days left, ending August 23, 2012

You are currently running in the Advanced Mode

Fig. 2.6.1. 'Splash screen' showing the initial Data Root Folder. This 'splash screen' is the one shown if running in the Demotrial Period showing the number of days left, the ending date, and the fact that you can currently run WinLTP in the Advanced Mode.

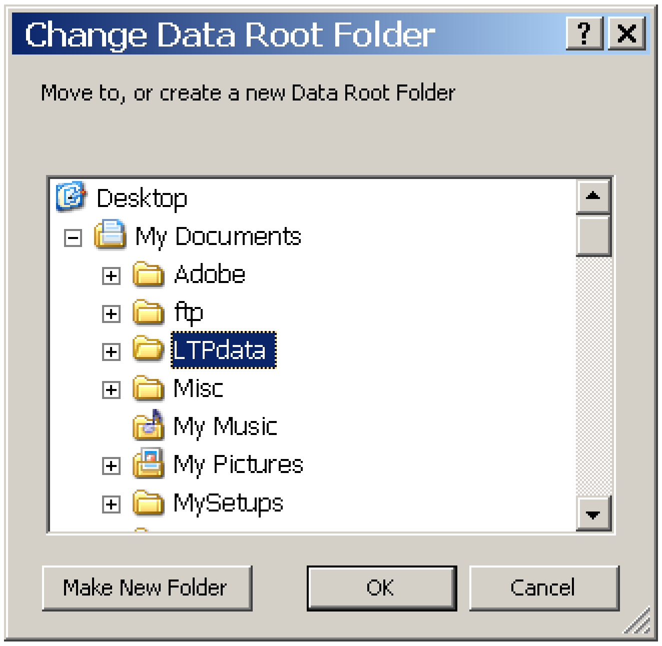
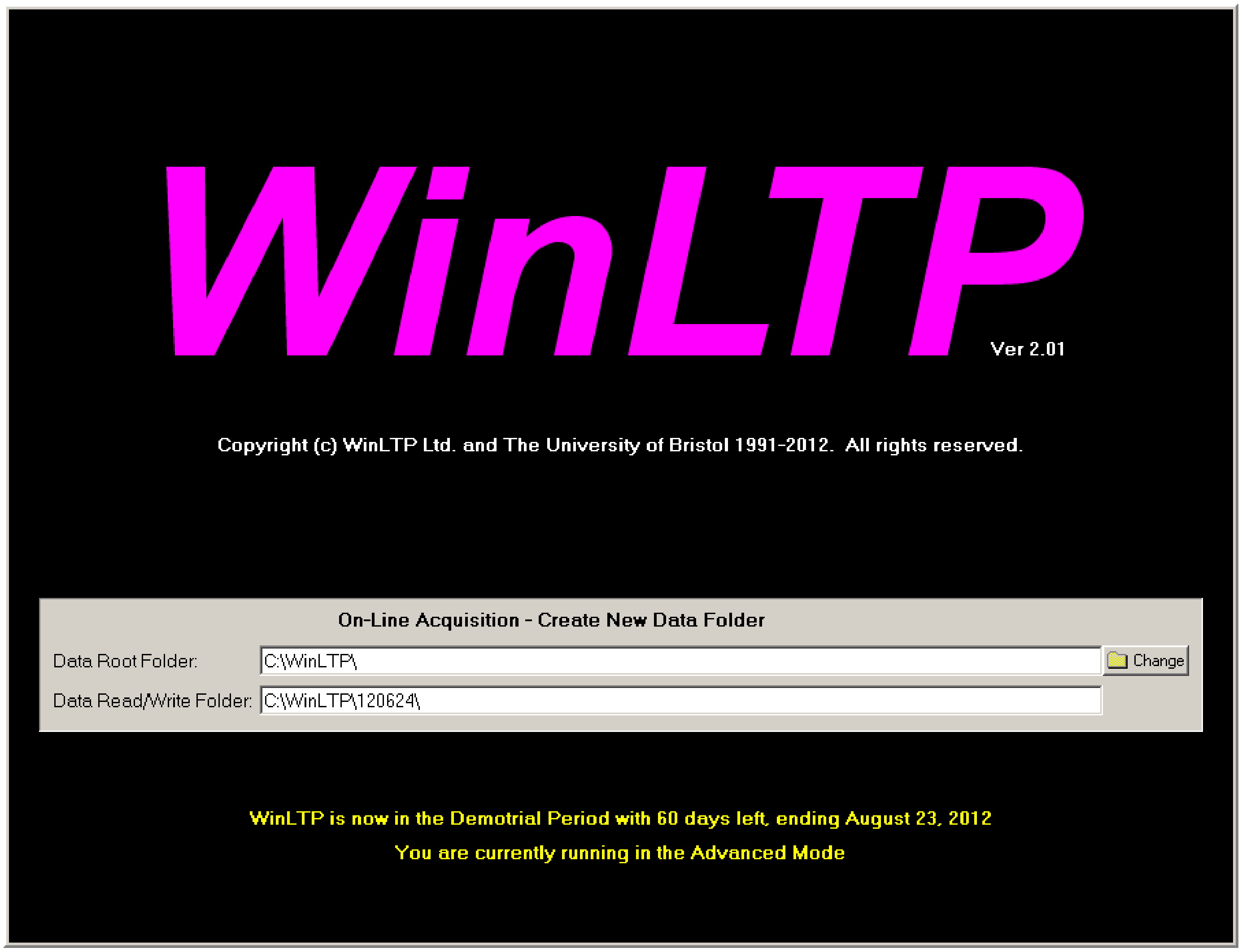


Fig. 2.6.2 The Change Data Root Folder dialog box.



Copyright (c) WinLTP Ltd. and The University of Bristol 1991-2012. All rights reserved.

On-Line Acquisition - Create New Data Folder

Data Root Folder: C:\WinLTP\ Change

Data Read/Write Folder: C:\WinLTP\120624\

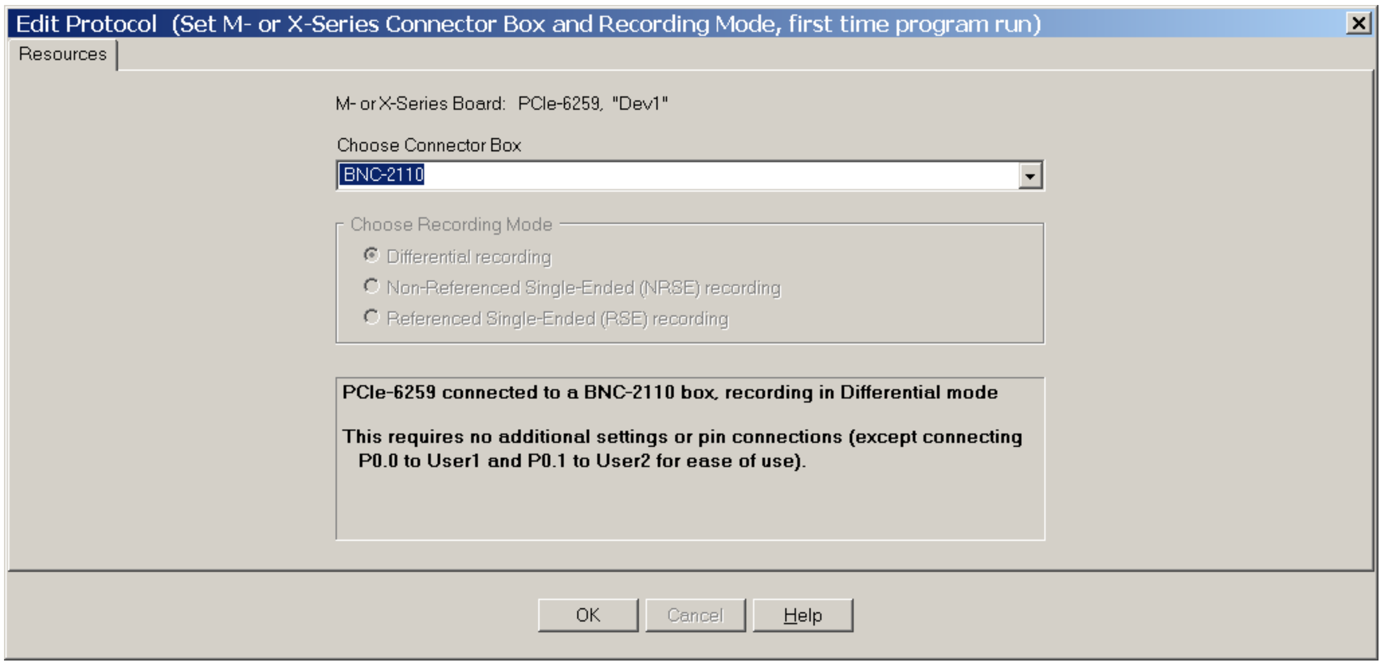
WinLTP is now in the Demotrial Period with 60 days left, ending August 23, 2012

You are currently running in the Advanced Mode

Fig. 2.6.3. ‘Splash screen’ showing the final Data Root Folder, and the Data Read/Write Folder into which data will be written during acquisition and analysis.

2.7 Choose the Connector Box and Recording Mode for M-,X-Series boards

If this is the first time WinLTP has been run on this M or X-Series board, an 'Edit Protocol (Set M- or X-Series Connector Box and Recording Mode, first time program run)' dialog box will appear (Fig. 2.7.1).


Edit Protocol (Set M- or X-Series Connector Box and Recording Mode, first time program run)

Resources

M- or X-Series Board: PCIe-6259, "Dev1"

Choose Connector Box

BNC-2110

Choose Recording Mode

[✓ Differential recording]

Non-Referenced Single-Ended (NRSE) recording

Referenced Single-Ended (RSE) recording

PCIe-6259 connected to a BNC-2110 box, recording in Differential mode

This requires no additional settings or pin connections (except connecting P0.0 to User1 and P0.1 to User2 for ease of use).

Fig. 2.7.1. The Resources tabsheet in the Edit Protocol (Set M- or X-Series Connector Box and Recording Mode, first time program run)' dialog box showing the choice of a BNC-2110 Connector Box.

The top line of the Resources tabsheet of this dialog box shows the National Instruments data aquistion board you have installed, in this case a PCIe-6259.

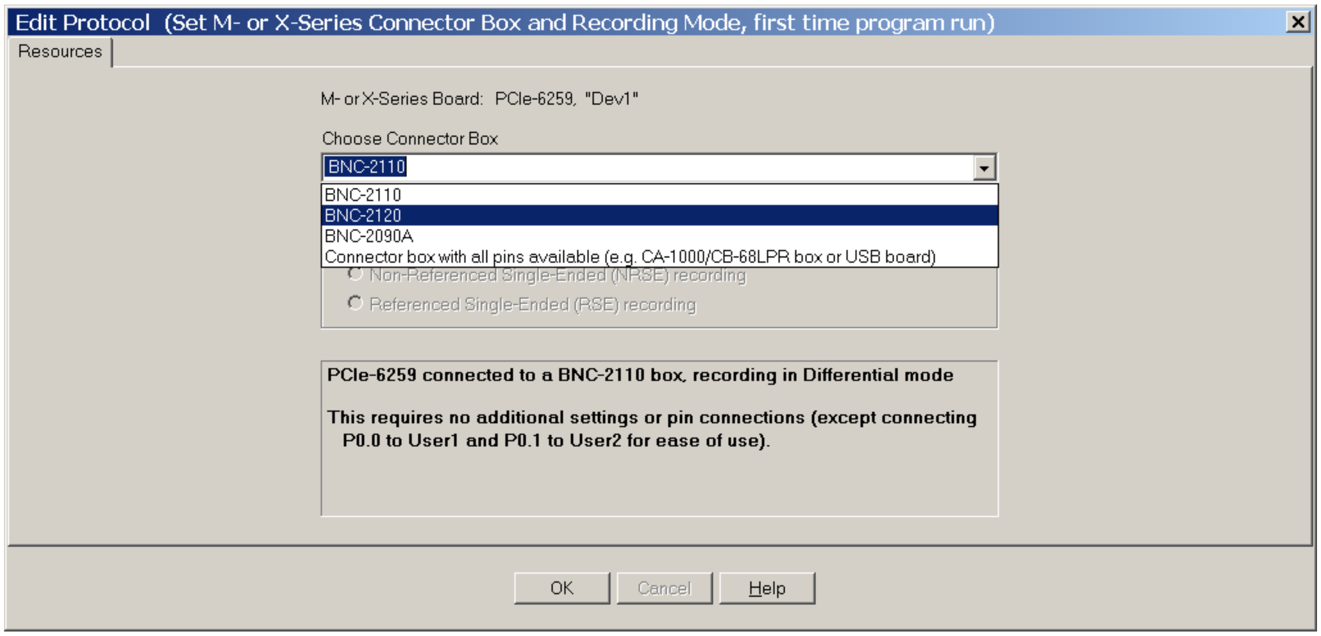
The section below that requests you choose from a drop-down list a Connector Box that is connected to this data acquisition board (Fig. 2.7.2). The current Connector Boxes to choose from include the BNC-2110, BNC-2120, BNC-2090A and a connector box that has all input/output pins available.

NOTE: Choose the Connector Box you installed in Section 2.3.3.

This is then followed by a 'Choose Recording Mode' radiobutton group which is either set to a particular appropriate value for a particular Connector Box (for example Differential recording from the BNC-2110 Connector Box (Fig. 2.7.1), or the BNC-2210 Connector Box (Fig. 2.7.3). Or for certain Connector Boxes (such as the BNC-2090A, Fig. 2.7.4) or a connector box that has all input/output pins available (Fig. 2.7.5)) you can choose Differential, Non-Referenced Single-Ended (NRSE), or Referenced Single-Ended (RSE) recording mode.

Below the 'Choose Recording Mode' radiobutton group is an Information Panel. The top line shows the data acquisition board, connector box and recording mode. The middle lines show any additional switch settings or pin-to-pin wire connections that need to be made. The bottom lines (if any) give a WARNING (in yellow) for any setting need be changed when upgrading from WinLTP 1.10 or earlier.

NOTE: After you have chosen the Connector Box you installed in Section 2.3.3 and have chosen the Recording Mode, set any switches mentioned in the Information Panel, and any pin-to-pin wire connections required. This is particularly true if the bottom lines give a WARNING (in yellow) for any setting need be changed when upgrading from WinLTP 1.10 or earlier.


Edit Protocol (Set M- or X-Series Connector Box and Recording Mode, first time program run)
Resources
OK
Cancel
Help

M- or X-Series Board: PCIe-6259, "Dev1"

Choose Connector Box: BNC-2110

- BNC-2110
- BNC-2120
- BNC-2090A
- Connector box with all pins available (e.g. CA-1000/CB-68LPR box or USB board)

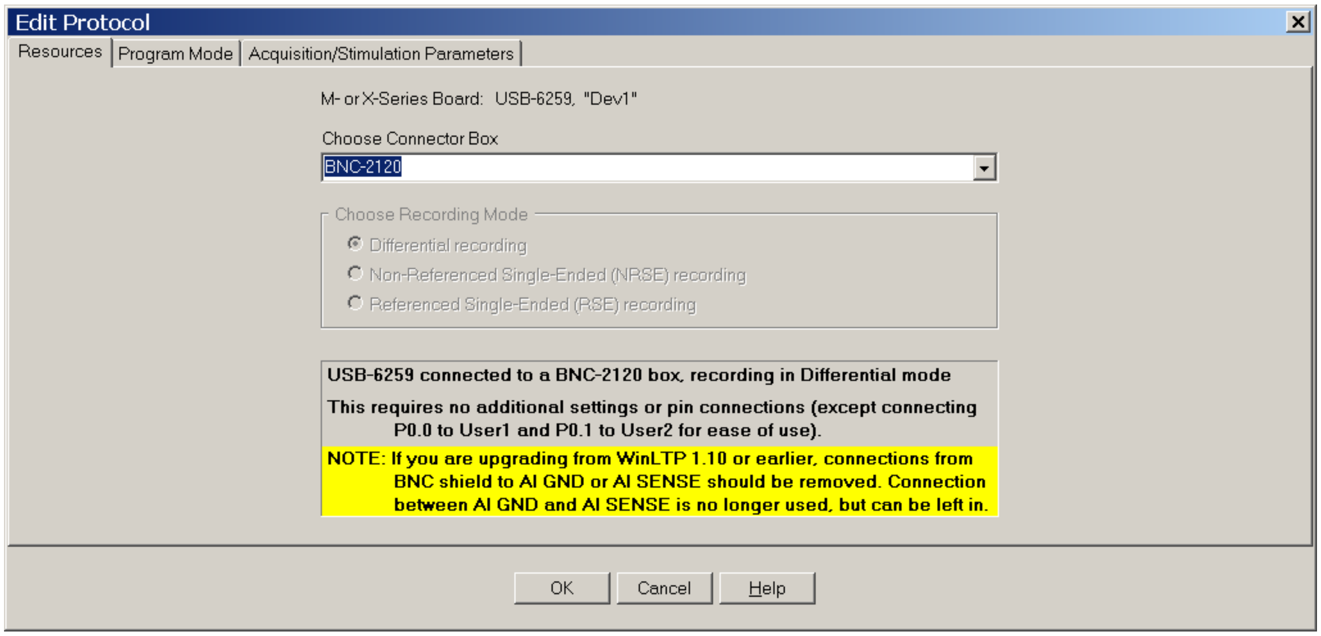
Non-Referenced Single-Ended (NRSE) recording

Referenced Single-Ended (RSE) recording

PCIe-6259 connected to a BNC-2110 box, recording in Differential mode

This requires no additional settings or pin connections (except connecting P0.0 to User1 and P0.1 to User2 for ease of use).

Fig. 2.7.2. The Resources tabsheet in the Edit Protocol (Set M- or X-Series Connector Box and Recording Mode, first time program run)' dialog box showing the choices Connector Boxes: BNC-2110, BNC-2120, BNC-2090A and a Connector Box that has all input/output pins available such as the CA-1000 enclosure with a CB-68LPR connector block. You would also choose 'Connector box with all pins available' to use a USB board.



M- or X-Series Board: USB-6259, "Dev1"

Choose Connector Box: BNC-2120

Choose Recording Mode

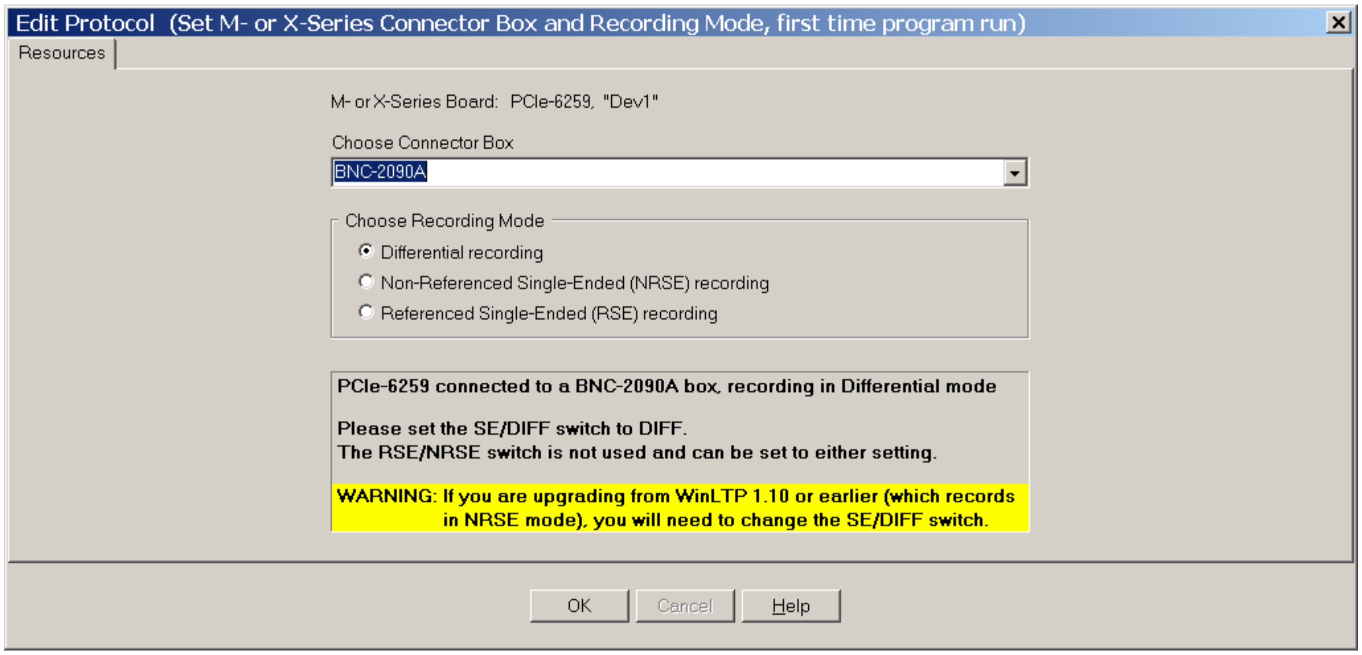
- [✓ Differential recording]
- Non-Referenced Single-Ended (NRSE) recording
- Referenced Single-Ended (RSE) recording

USB-6259 connected to a BNC-2120 box, recording in Differential mode

This requires no additional settings or pin connections (except connecting P0.0 to User1 and P0.1 to User2 for ease of use).

NOTE: If you are upgrading from WinLTP 1.10 or earlier, connections from BNC shield to AI GND or AI SENSE should be removed. Connection between AI GND and AI SENSE is no longer used, but can be left in.

Fig. 2.7.3. The Resources tabsheet in the Edit Protocol (Set M- or X-Series Connector Box and Recording Mode, first time program run)' dialog box showing the choice of a BNC-2120 Connector Box.


Edit Protocol (Set M- or X-Series Connector Box and Recording Mode, first time program run)
Resources
OK
Cancel
Help

M- or X-Series Board: PCIe-6259, "Dev1"

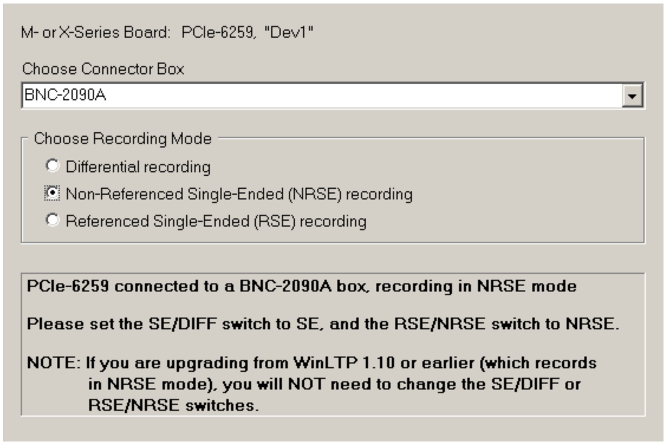
Choose Connector Box: BNC-2090A

Choose Recording Mode
- [✓ Differential recording]
- Non-Referenced Single-Ended (NRSE) recording
- Referenced Single-Ended (RSE) recording

PCIe-6259 connected to a BNC-2090A box, recording in Differential mode

Please set the SE/DIFF switch to DIFF.
The RSE/NRSE switch is not used and can be set to either setting.

WARNING: If you are upgrading from WinLTP 1.10 or earlier (which records in NRSE mode), you will need to change the SE/DIFF switch.



M- or X-Series Board: PCIe-6259, "Dev1"

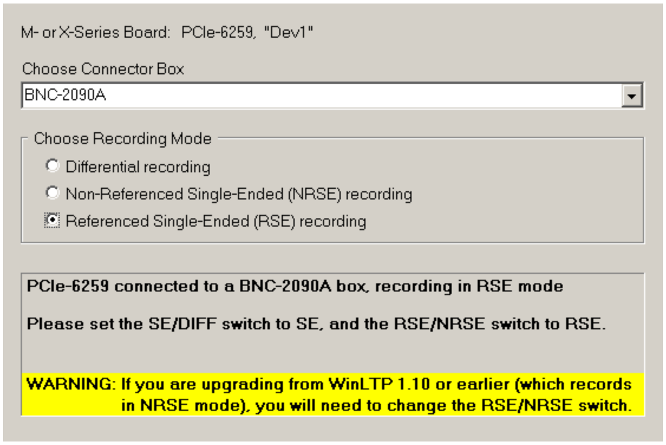
Choose Connector Box: BNC-2090A

Choose Recording Mode
- Differential recording
- [✓ Non-Referenced Single-Ended (NRSE) recording]
- Referenced Single-Ended (RSE) recording

PCIe-6259 connected to a BNC-2090A box, recording in NRSE mode

Please set the SE/DIFF switch to SE, and the RSE/NRSE switch to NRSE.

NOTE: If you are upgrading from WinLTP 1.10 or earlier (which records in NRSE mode), you will NOT need to change the SE/DIFF or RSE/NRSE switches.



M- or X-Series Board: PCIe-6259, "Dev1"

Choose Connector Box: BNC-2090A

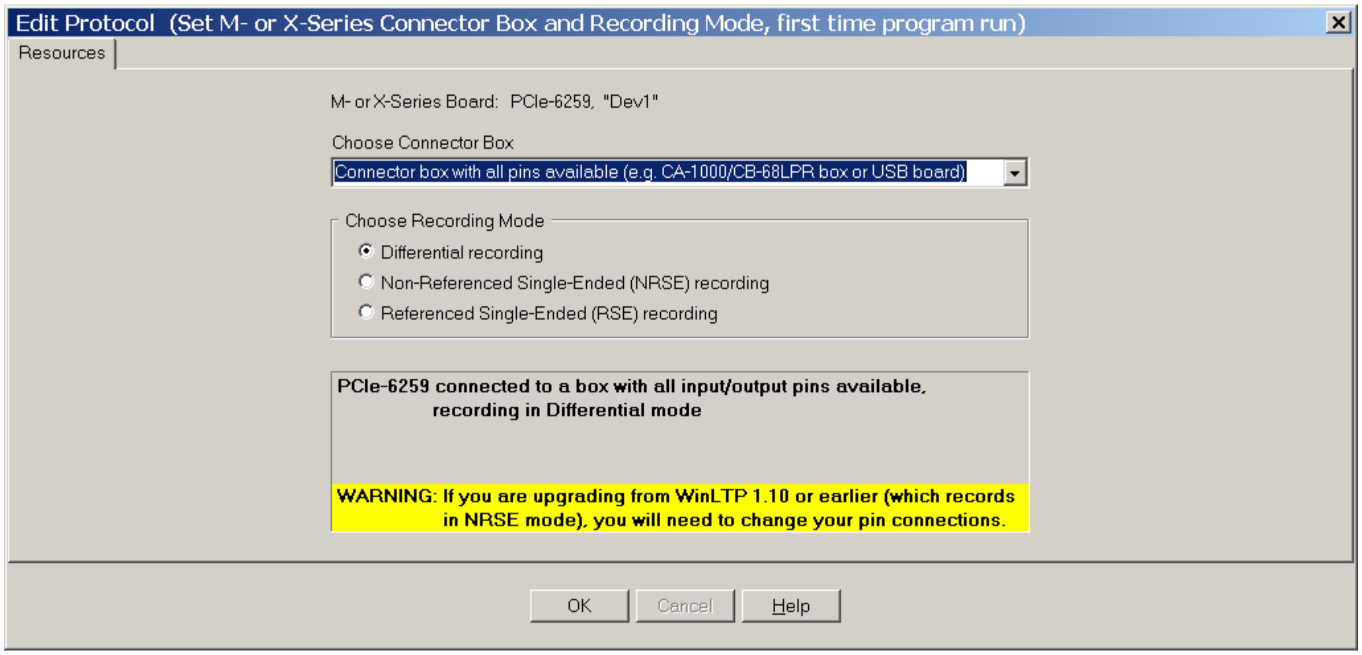
Choose Recording Mode
- Differential recording
- Non-Referenced Single-Ended (NRSE) recording
- [✓ Referenced Single-Ended (RSE) recording]

PCIe-6259 connected to a BNC-2090A box, recording in RSE mode

Please set the SE/DIFF switch to SE, and the RSE/NRSE switch to RSE.

WARNING: If you are upgrading from WinLTP 1.10 or earlier (which records in NRSE mode), you will need to change the RSE/NRSE switch.

Fig. 2.7.4. The Resources tabsheet in the Edit Protocol (Set M- or X-Series Connector Box and Recording Mode, first time program run)' dialog box showing the choice of a BNC-2090A Connector Box. With the BNC-2090A you can choose between Differential recording mode (top panel) , Non-Referenced Single-Ended (NRSE) mode (lower left panel), and Referenced Single-Ended (RSE) mode (lower right panel).


Edit Protocol (Set M- or X-Series Connector Box and Recording Mode, first time program run)

M- or X-Series Board: PCIe-6259, "Dev1"

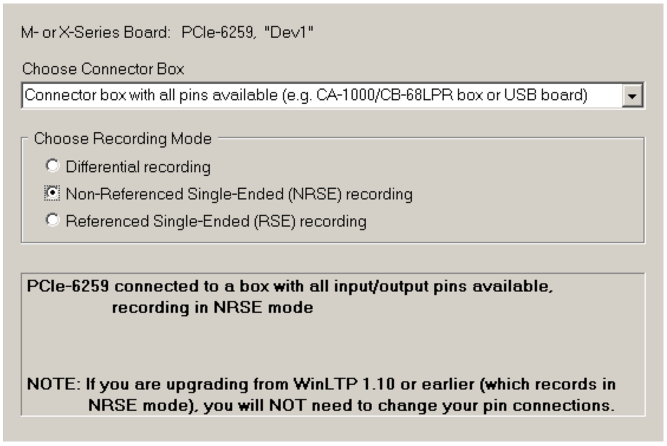
Choose Connector Box: Connector box with all pins available (e.g. CA-1000/CB-68LPR box or USB board)

Choose Recording Mode

- [✓ Differential recording]
- Non-Referenced Single-Ended (NRSE) recording
- Referenced Single-Ended (RSE) recording

PCIe-6259 connected to a box with all input/output pins available, recording in Differential mode

WARNING: If you are upgrading from WinLTP 1.10 or earlier (which records in NRSE mode), you will need to change your pin connections.


M- or X-Series Board: PCIe-6259, "Dev1"

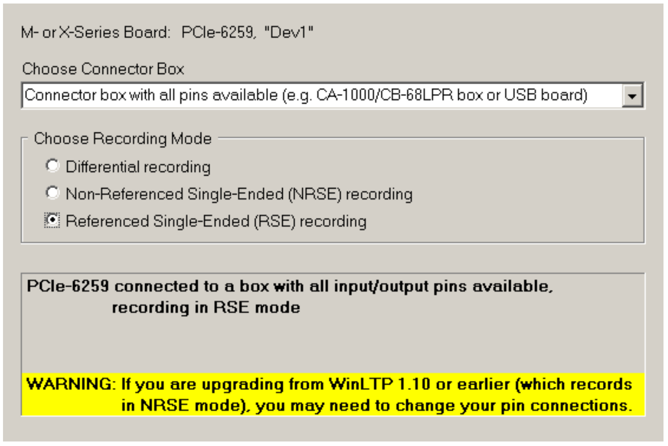
Choose Connector Box: Connector box with all pins available (e.g. CA-1000/CB-68LPR box or USB board)

Choose Recording Mode

- Differential recording
- [✓ Non-Referenced Single-Ended (NRSE) recording]
- Referenced Single-Ended (RSE) recording

PCIe-6259 connected to a box with all input/output pins available, recording in NRSE mode

NOTE: If you are upgrading from WinLTP 1.10 or earlier (which records in NRSE mode), you will NOT need to change your pin connections.


M- or X-Series Board: PCIe-6259, "Dev1"

Choose Connector Box: Connector box with all pins available (e.g. CA-1000/CB-68LPR box or USB board)

Choose Recording Mode

- Differential recording
- Non-Referenced Single-Ended (NRSE) recording
- [✓ Referenced Single-Ended (RSE) recording]

PCIe-6259 connected to a box with all input/output pins available, recording in RSE mode

WARNING: If you are upgrading from WinLTP 1.10 or earlier (which records in NRSE mode), you may need to change your pin connections.

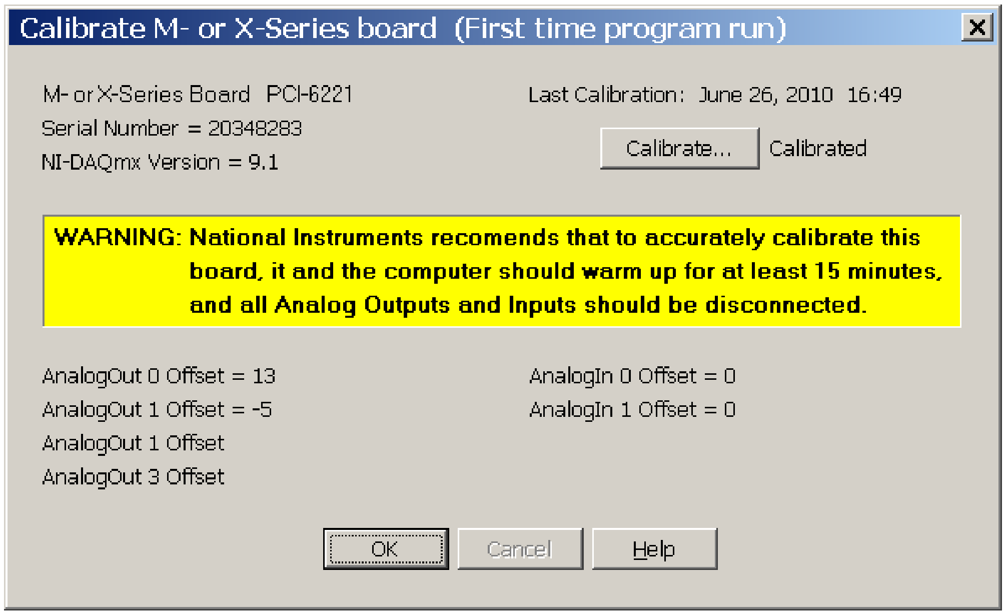
Fig. 2.7.5. The Resources tabsheet in the Edit Protocol (Set M- or X-Series Connector Box and Recording Mode, first time program run)' dialog box showing the choice of a connector box with all pins available such as the CA-1000 enclosure with a CB-68LPR connector block. With the connector box with all pins available, you can choose between Differential recording mode (top panel) , Non-Referenced Single-Ended (NRSE) mode (lower left panel), and Referenced Single-Ended (RSE) mode (lower right panel.

2.8 Calibrate Data Acquisition Board When Running the First Time

If this is the first time WinLTP has been run on **this** M or X-Series board or **this** Digidata 132x board, a 'Calibrate M- or X-Series board' or 'Calibrate Digidata 132x' dialog box will appear. Make sure to remove the connections to the Analog Outputs before calibrating. To calibrate, simply click on the 'Calibrate…' button. Figs. 2.8.1 and 2.8.2 show the calibration results for M or X-Series and Digidata 132x boards after the 'Calibrate…' button has been pressed and calibration has successfully concluded.

For M or X-Series boards, National Instruments recommends that the computer and board have warmed up for at least 15 minutes. Also, National Instruments recommends that you periodically self-calibrate the board by clicking on:

Options -> Recalibrate Data Acquisition Board

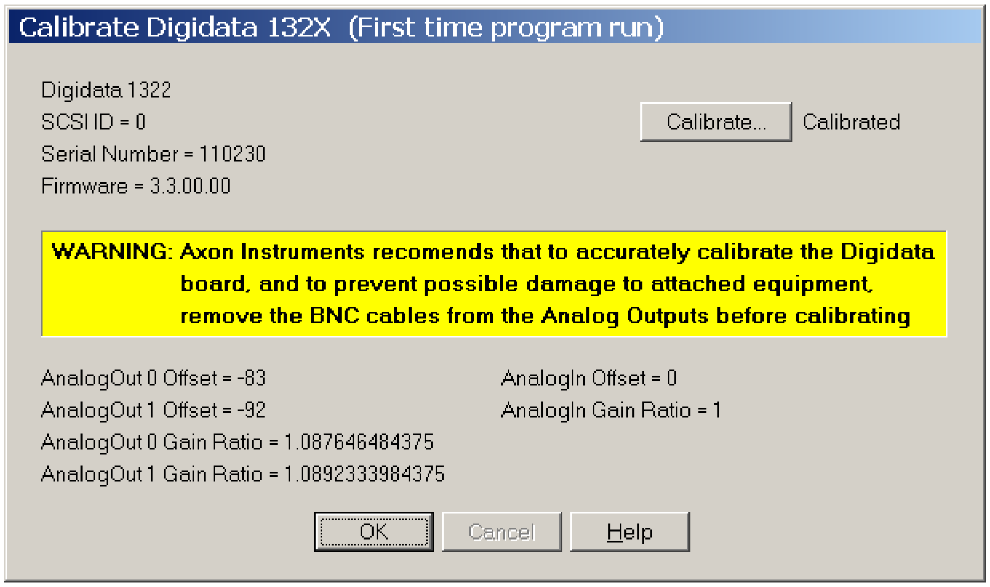


M- or X-Series Board PCI-6221
Serial Number = 20348283
NI-DAQmx Version = 9.1
Last Calibration: June 26, 2010 16:49
Calibrated

WARNING: National Instruments recomends that to accurately calibrate this board, it and the computer should warm up for at least 15 minutes, and all Analog Outputs and Inputs should be disconnected.

AnalogOut 0 Offset = 13
AnalogOut 1 Offset = -5
AnalogOut 1 Offset
AnalogOut 3 Offset
AnalogIn 0 Offset = 0
AnalogIn 1 Offset = 0

Fig. 2.8.1. The Calibrate M-Series dialog box after calibration has successfully concluded.



Digidata 1322
SCSI ID = 0
Serial Number = 110230
Firmware = 3.3.00.00
Calibrated

WARNING: Axon Instruments recomends that to accurately calibrate the Digidata board, and to prevent possible damage to attached equipment, remove the BNC cables from the Analog Outputs before calibrating

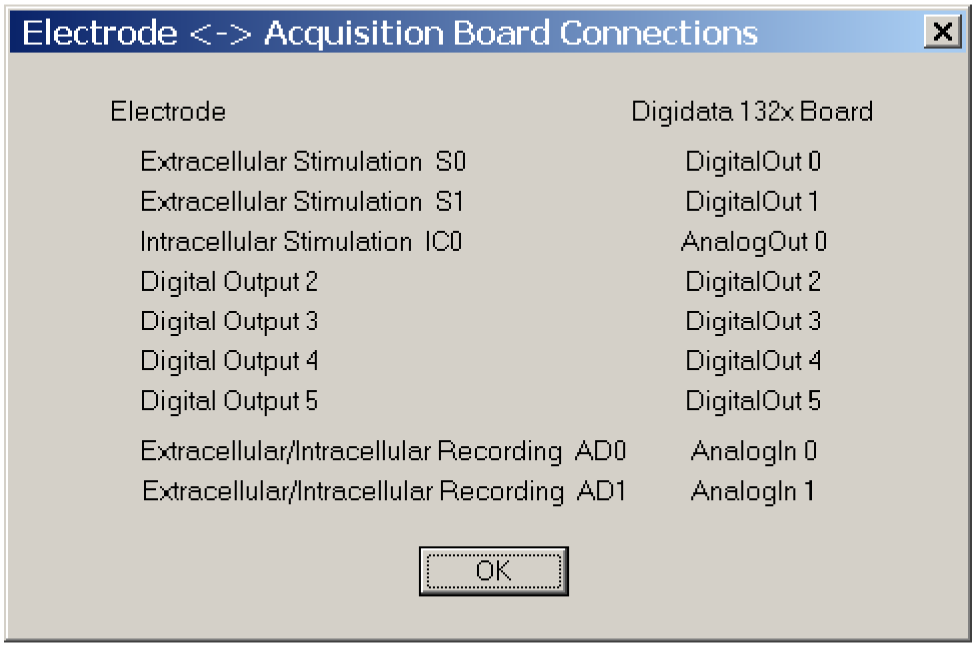
AnalogOut 0 Offset = -83
AnalogOut 1 Offset = -92
AnalogOut 0 Gain Ratio = 1.087646484375
AnalogOut 1 Gain Ratio = 1.0892333984375
AnalogIn Offset = 0
AnalogIn Gain Ratio = 1

Fig. 2.8.2. The Calibrate Digidata 132x dialog box after calibration has successfully concluded.

2.9 Connect the amplifier and stimulus isolation unitsto the data acquisition board

To connect the amplifier(s) analog inputs and outputs, and the stimulus isolation units (SIUs) use the main menu command

Help -> Electrode / Data Acquisition Board Connections…
to call up the Electrode <-> Acquisiton Board Connections dialog box (Fig. 2.9.1).


Electrode <-> Acquisition Board Connections

Electrode	Digidata 132x Board
Extracellular Stimulation S0	DigitalOut 0
Extracellular Stimulation S1	DigitalOut 1
Intracellular Stimulation IC0	AnalogOut 0
Digital Output 2	DigitalOut 2
Digital Output 3	DigitalOut 3
Digital Output 4	DigitalOut 4
Digital Output 5	DigitalOut 5
Extracellular/Intracellular Recording AD0	AnalogIn 0
Extracellular/Intracellular Recording AD1	AnalogIn 1

Fig. 2.9.1. Electrode <-> Acquisition Board Connections dialog box to connect the extracellular stimulation SIUs, recording amplifier analog inputs and outputs, and digital outputs to the Digidata 132x data acquisition board.

This dialog shows where to connect the extracellular S0/S1 stimulation SIUs, intracellular IC0 analog stimulation, digital output stimulation and extracellular/intracellular AD0/AD1 recording to the Digidata 132x data acquisition board.

2.10 WinLTP is running in either the Basic or Advanced Mode

When WinLTP is started it is running in either the Basic or Advanced Mode. Whether it is running in Basic or Advanced Mode is dependent on several factors.

2.10.1 DemoTrial Period - Advanced Mode

If WinLTP is running in the 2 month DemoTrial Period, it is running by default in the Advanced Mode, and the Advanced Mode capabilities can be freely used (Fig. 2.6.1).

2.10.2 Post-DemoTrial Period - Basic Mode

If no Temporary or Permanent License Key file is installed, after this 2 month DemoTrial Period, WinLTP will automatically run in the Basic Mode. When WinLTP is started at this time, the 'splash screen' will display the following message saying when the DemoTrial Period ended and that WinLTP is running in the Basic Mode (Fig. 2.10.2.1).

Running WinLTP in the post-DemoTrial Period in the Basic Mode is free and has no time constraints. However, in order to protect against running WinLTP in the Advanced Mode and saving data after the Demotrial Period without purchasing a Permanent License Key, copy protection has been added. You are **not allowed** to:

1) Reverse engineer, debug or decompile WinLTP or the License Key.
1) Clock back the computer date or reinstall the program in an attempt to overcome the copy protection.
3) Distribute your Permanent License Key outside your lab group

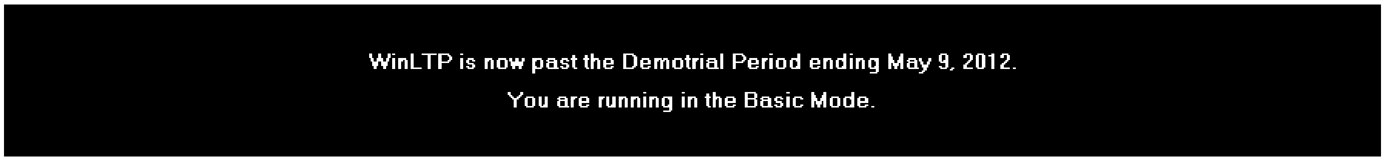

WinLTP is now past the Demotrial Period ending May 9, 2012.
You are running in the Basic Mode.

Fig. 2.10.2.1. If the 2 month DemoTrial Period is past, this 'splash screen' says when the DemoTrial Period ended and that WinLTP is running in the Basic Mode.

2.10.3 Temporary License Key – Advanced Mode

However, beginning with WinLTP 1.00, a Temporary or a purchased Permanent License Key file is required to run WinLTP in the Advanced Mode. If there is a reason, WinLTP Ltd. can send you a time-limited Temporary License Key file which you put into the \WinLTP program folder to allow you to run WinLTP temporarily in the Advanced mode. If you are using a Temporary License Key file, you will see the following message at the bottom of the beginning 'splash screen' (Fig. 2.10.3.1).

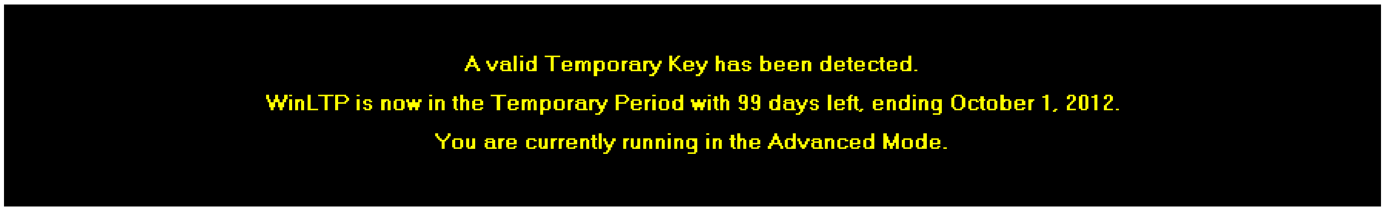

A valid Temporary Key has been detected.
WinLTP is now in the Temporary Period with 99 days left, ending October 1, 2012.
You are currently running in the Advanced Mode.

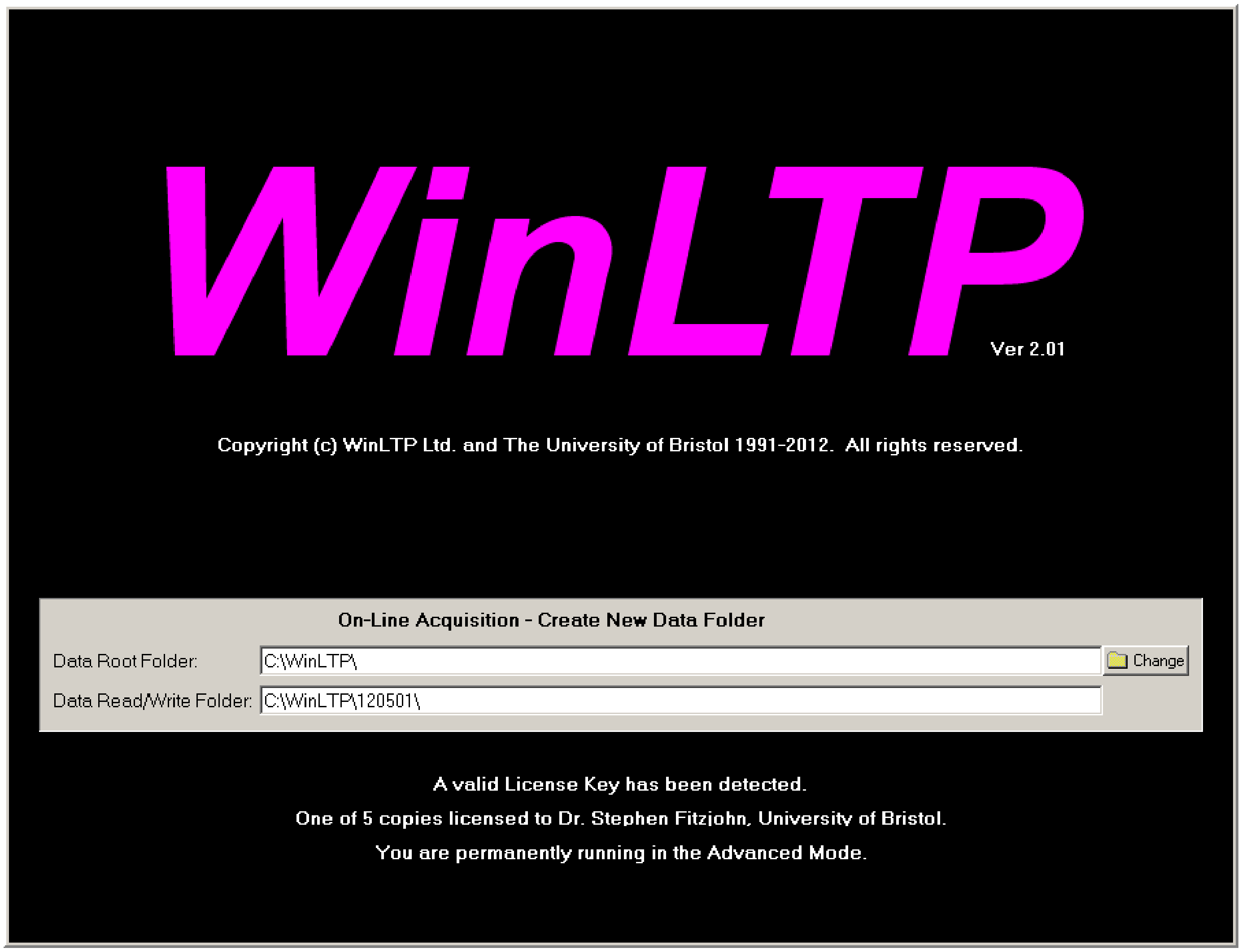
Fig. 2.10.3.1. Partial 'splash screen' information when a Temporary Key is placed in the \WinLTP folder to temporarily use WinLTP in the Advanced Mode. It shows the number of days left, the ending date, and the fact that you can temporarily run WinLTP in the Advanced Mode.

2.10.4 Permanent License Key – Advanced Mode

As mentioned above, beginning with WinLTP 1.00, a Temporary or a purchased Permanent License Key file is required to run WinLTP in the Advanced Mode.

To purchase a Permanent License Key see WinLTP's webpage (www.winltp.com). Put your purchased Permanent License Key file into the \WinLTP program folder to allow you to run WinLTP in the Advanced mode. When using a Permanent License Key file, you will see the following message at the bottom of the beginning 'splash screen' (Fig. 2.10.4.1). This tells you that the key is permanent (i.e. time-unlimited), and that you can permanently run WinLTP in the Advanced Mode.

Usually this purchased Permanent License key allows you to run up to 5 copies on 5 computers simultaneously. If you plan to run more than 5 copies on 5 computers, please purchase additional 5 licenses.



Copyright (c) WinLTP Ltd. and The University of Bristol 1991-2012. All rights reserved.

Data Root Folder: C:\WinLTP\

Data Read/Write Folder: C:\WinLTP\120501\

A valid License Key has been detected.

One of 5 copies licensed to Dr. Stephen Fitzjohn, University of Bristol.

You are permanently running in the Advanced Mode.

Fig. 2.10.4.1. 'Splash screen' showing that a Permanent License Key file has been detected in the \WinLTP program folder, that it was licensed to run up to 5 copies on up to 5 computers simultaneously, who and where it was licensed to, and that you are permanently running WinLTP in the Advanced Mode.

2.11 Basic and Advanced Mode Capabilities

When you enter the WinLTP program for the first time in the DemoTrial period you are running in the Advanced Mode with a fully functioning Protocol builder (Fig. 2.11.1, left). In this mode you can write any number of advanced protocols including this automated perfusion protocol using 'Slow0' and 'Slow1 Perfuse' events. All the Protocol Builder events can be used and are shown in green, including the 'Run', 'ElseRun', 'AvgLoop', and 'Loop' events, the 'Slow0', 'Slow1', 'Fast0' and 'Fast1' Perfuse events, the P0, P1, T0 and T1sweep events, and the Delay event. The Advanced Mode in general is required to:

1) have a fully functional Protocol Builder
2) run fully Automated Perfusion Control (which uses the Perfuse events Protocol Builder)
3) for Full Multitasking, to save Continuous Acquisition files while running the Main Protocol
4) save the Experimental Log file
5) convert WinLTP ADsweep files to Axon Binary (*.abf) files

If you have ordered an Advanced Mode license you will continue with the fully functional Protocol Builder (Fig. 2.11.1, left).

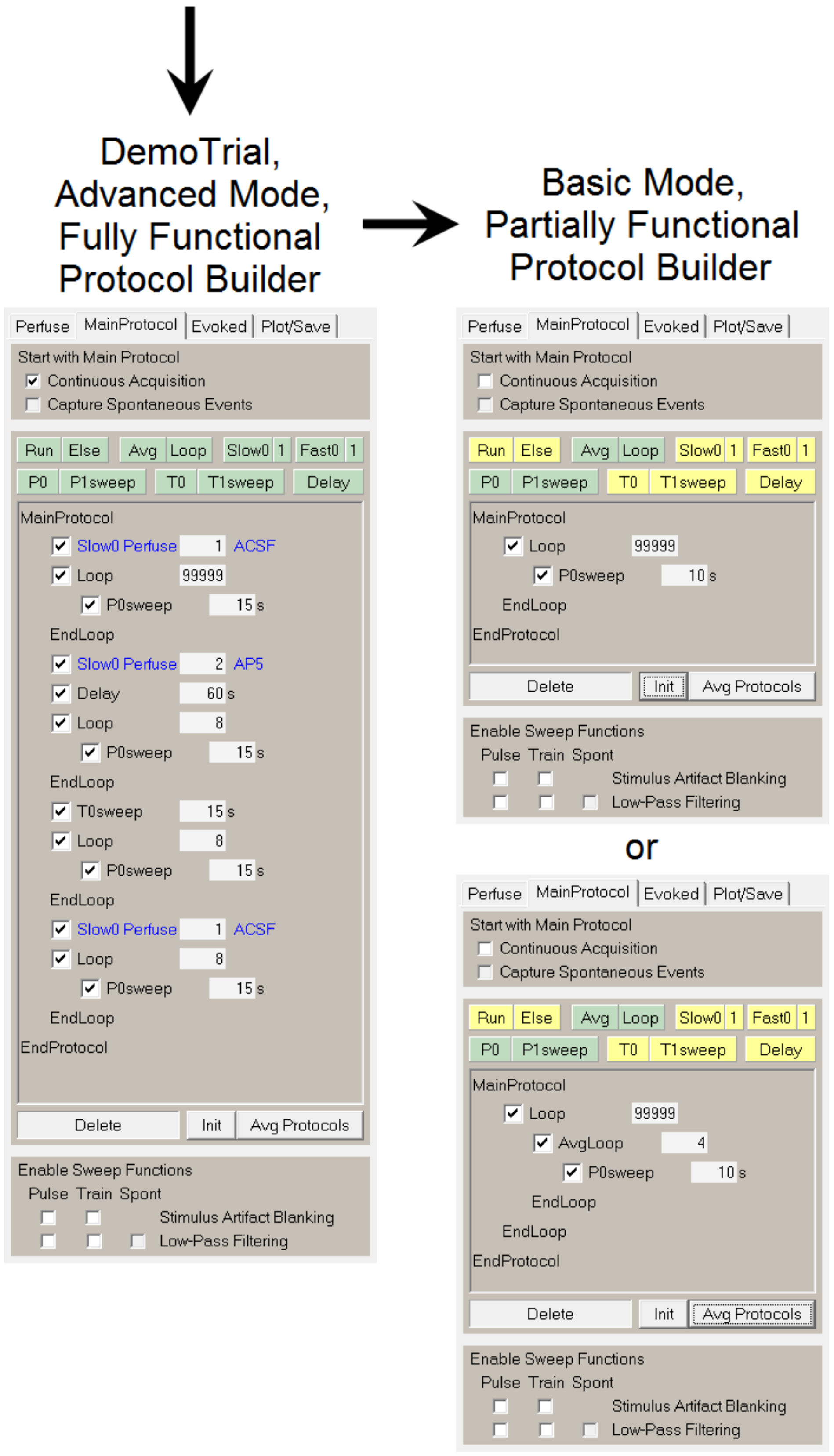
Alternatively, if you have not ordered an Advanced Mode license, at the end of the 2 month DemoTrial period you will automatically enter the Basic Mode partially functional Protocol Builder (Fig. 2.11.1, right). Only the green Insert Event buttons can be used and still save ADsweep data. These include 'AvgLoop', 'Loop', 'P0sweep' and P1sweep events. If you use the yellow Insert Event buttons, the 'Run', 'ElseRun', 'Tosweep' 'T1sweep' and 'Delay' events, **your protocol will run perfectly OK, except that the ADsweep data will not be saved.** This allows you to easily test the Advanced Mode functions to see if it is worthwhile upgrading to the Advanced Version.

Even with the Basic Mode, partially functional Protocol Builder, you can run basic LTP experiments. By clicking the **'Init' Protocol button** (top panel), you can immediately do repetitive P0sweeps, by clicking on the **Init 'Avg Protocols' button** (bottom panel)or you could do repetitive P0sweeps with Signal Averaging.

You can also get alternating P0/P1sweeps you press down the LeftMouseButton to click on the 'P1sweep' Insert button, hold the LeftMouseButton down to **drag the P1sweep down** to just below the P0sweep in the MainProtocol script, and then release the LeftMouseButton to insert the P1sweep just below the P0sweep.

And by clicking on the **Evoked 'Single' and 'Repetitive Sweeps' buttons** while the MainProtocol is running, you can evoke Single or Repetitive P0, P1, T0 and T1 sweeps, which allows you to run basic LTP type experiments.

For now just click on the 'Init' Protocol button (Fig. 2.11.1, right top panel), to be able to run repetitive P0sweeps.



Start with Main Protocol
☑ Continuous Acquisition
☐ Capture Spontaneous Events

```
MainProtocol
  ☑ Slow0 Perfuse     1  ACSF
  ☑ Loop          99999
      ☑ P0sweep       15 s
  EndLoop
  ☑ Slow0 Perfuse     2  AP5
  ☑ Delay            60 s
  ☑ Loop              8
      ☑ P0sweep       15 s
  EndLoop
  ☑ T0sweep          15 s
  ☑ Loop              8
      ☑ P0sweep       15 s
  EndLoop
  ☑ Slow0 Perfuse     1  ACSF
  ☑ Loop              8
      ☑ P0sweep       15 s
  EndLoop
EndProtocol
```

Enable Sweep Functions
Pulse Train Spont
☐ ☐ Stimulus Artifact Blanking
☐ ☐ ☐ Low-Pass Filtering

Start with Main Protocol
☐ Continuous Acquisition
☐ Capture Spontaneous Events

```
MainProtocol
  ☑ Loop          99999
      ☑ P0sweep       10 s
  EndLoop
EndProtocol
```

Enable Sweep Functions
Pulse Train Spont
☐ ☐ Stimulus Artifact Blanking
☐ ☐ ☐ Low-Pass Filtering

Start with Main Protocol
☐ Continuous Acquisition
☐ Capture Spontaneous Events

```
MainProtocol
  ☑ Loop          99999
      ☑ AvgLoop        4
          ☑ P0sweep   10 s
      EndLoop
  EndLoop
EndProtocol
```

Enable Sweep Functions
Pulse Train Spont
☐ ☐ Stimulus Artifact Blanking
☐ ☐ ☐ Low-Pass Filtering

Fig. 2.11.1. Left) When you enter the WinLTP program for the first time in the DemoTrial period you are running in the Advanced Mode with a fully functioning Protocol builder. All the Insert Event buttons can be used and are marked in green. Right) In the Basic Mode, the Protocol Builder is partially function and only the green Insert Events can be used, not those in yellow. Clicking on the 'Init' button writes a continuous loop script containing one P0sweep. Clicking on the 'Avg Protocols' button writes an averaging loop script containing one P0sweep.

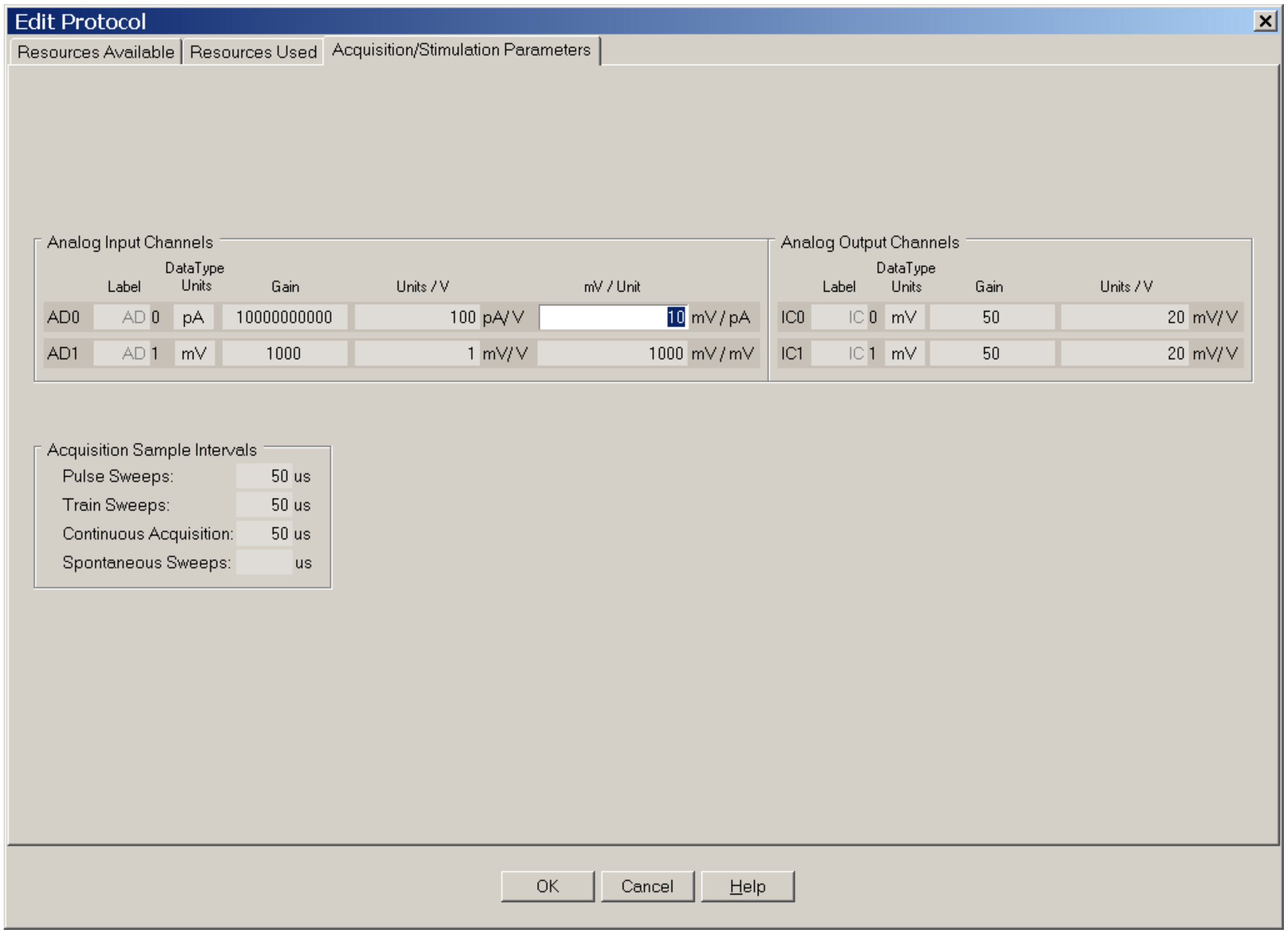
2.12 Set the AD Gain, DataType, Sample Interval and Other Parameters

2.12.1 Set the AD Gain, Data Type, and Sample Interval in the Edit Protocol dialog box

Once in the Edit Protocol dialog box, click on the Acquisition/Stimulation Parameters tabsheet (Fig. 2.12.1.1).

The Data Type Units determines whether an acquisition channel is in extracellular or intracellular current clamp mode, or in patch clamp voltage clamp mode.

Mode	DataType Unit
Extracellular	mV
Intracellular current clamp	mV
Patch clamp voltage clamp mode	pA



Analog Input Channels

	Label	DataType Units	Gain	Units / V	mV / Unit
AD0	AD 0	pA	10000000000	100 pA/V	10 mV / pA
AD1	AD 1	mV	1000	1 mV/V	1000 mV / mV

Analog Output Channels

	Label	DataType Units	Gain	Units / V
IC0	IC 0	mV	50	20 mV/V
IC1	IC 1	mV	50	20 mV/V

Acquisition Sample Intervals

Pulse Sweeps:	50 us
Train Sweeps:	50 us
Continuous Acquisition:	50 us
Spontaneous Sweeps:	us

Fig. 2.12.1.1. Edit Protocol dialog box showing the Acquisition/Stimulation Parameters tabsheet. Note the new addition of the Analog Input Channels mV/Unit fields.

First, set the channel AD0 **DataType** to "mV".

Second, change the AD0 **Gain** to be equal to the total amplification gain from the electrode to the AD board connection. For example, if an Axon Instruments AxoClamp is used in current clamp mode with an internal x10 output gain, which then goes into an external x100 amplifier and then into channel AD0, the AD0 Gain would be 1000.

Alternatively if an Axon Instruments AxoPatch 1D or 200B is used in whole cell voltage clamp mode (with a CV-04 1/100 headstage where beta = 1), with an output gain of alpha = 10, and with the output going channel AD0, the channel AD0 Gain would be the following:

$$\text{AD0 Gain} = \frac{V_{out}}{I_{in}} = \frac{\text{Beta * Alpha mV}}{\text{pA}} = \frac{1 * 10 * 10^{-3}}{10^{-12}} = 10^{10} = 10000000000$$

(Yes that's 10 0's!) If the output gain is changed to = 20, the AD0 Gain increases to 20000000000. Then set the channel AD0 DataType to "pA".

Alternatively, the gain can be changed by entering it in **Units/V** (Fig. 2.12.1). While maybe it is clearer to just type an AD Gain of 1000 for an extracellular or intracellular experiment in "mV", with a patch clamp experiment in "pA" with enormous gain, a AD Units/V of 100 pA/V is much clearer than an AD Gain of 10000000000.

Furthermore, there is now also a **mV/Unit** field which as been added so that the patch-clamp gain can now be set as mV/pA. This is to improve clarity of setting patch-clamp gain in mV/Unit (as in 10 mV/pA) because many patch-clamps set gain in mV/pA. This gain is included with the normal strait gain (as in Gain = 10000000000) and gain in Units/V (as in 100 pA/V).

Third, set the Train and PulseSweep Acquisition Sample Intervals to what you want, say 50 µsec (Fig. 2.12.1).

2.12.2 Set the Digital-to-Analog Output

Don’t worry about this for now. (Setting Digital-to-Analog output discussed in Section 4.6.1.)

2.12.3 Set the PulseSweep P0 Stimulation Values

Don’t worry about setting P0sweep stimulation values in the Sweep Stimulation Panel for now, just keep the default values which are for single S0 pulse stimulation. (Setting sweep stimulation is discussed in Section 4.7.1.)

2.13 Analog Filtering of the Signal Before Digitization

The waveform signal data should be filtered before being digitized with an analog filter set to half or less of the digitization frequency. For example, if you are acquiring an AD sample at 50 µsec intervals (e.g. at 20 KHz sampling frequency), the analog filter should be set to at maximum 10 kHz (1/2 the sampling frequency), or preferably to 5 kHz or lower (except if doing an Rs exponential fit, then keep it at 10 KHz). If the analog filter is set to higher than half the sampling frequency (for example to 20 kHz in the above example), mysterious things can begin to happen due to anti-aliasing such as loosing the capacitative transients (e.g. the Rs measurement) that occur during the voltage clamp pulse when patch clamping.

2.14 Click the 'MainProtocol' Button to Start Stimulating and Acquiring Data

In WinLTP, protocols are initiated either by clicking a Button, pressing a Function Key or by using the RunMenu menu commands. To start the Main Protocol either click the 'MainProtocol' button or press the F1 key, or pull down the Run menu and click on MainProtocol.

NOTE: If you want to change something (like the number of stimulus pulses, the Sweep Period, Delay time etc, you have to do it at least 5 sec before the event starts. This is because in this version there is substantial stimulus output buffering (3 sec in the Digidata board, and 2 sec in the WinLTP program. You have to plan ahead a bit. (Eventually I would like to get it down to 2-3 seconds for the Digidata 132x board, and 0.5-1 second for another board I have my eye on.) But this delay allows me to do multitasking Continuous Acquisition and Capturing Spontaneous Events (not implemented yet), so I think its worth it.

2.15 Check that the Data Acquisition and Stimulation are Working Correctly

Put in known amplitude and duration signals to check that WinLTP acquisition is working correctly, and record WinLTP output on an oscilloscope to make sure the WinLTP stimulation is working correctly.

2.16 Use the Windows Task Manager to check CPU and Memory Usage

In order to get an idea of how WinLTP is using your computer's resources, primarily CPU usage and memory usage you can use the Windows Task Manager. Press Ctrl-Alt-Del key combination to run the Windows Task Manager.

Click on the **Process** tab (Fig. 2.16.1). Then use the menu command:

View -> Update Speed -> High

to update CPU usage as quick as possible.

Then use the menu command:

View -> Select Columns…

to call up the Select Columns dialog box and then select CPU Usage, Memory Usage, Peak Memory Usage and any other you want such as, in this example, Base Priority and the number of Threads.

Fig. 2.16.1 shows a relatively low **CPU usage** of about **9%** when running WinLTP with a Digidata 1322A board on a 3.2 GHz computer (2AD channels at 50us sampling interval, Repeat Sweeps of **1 sec** duration with **no delay** between them, and Continuous Acquisition, with plotting on screen).

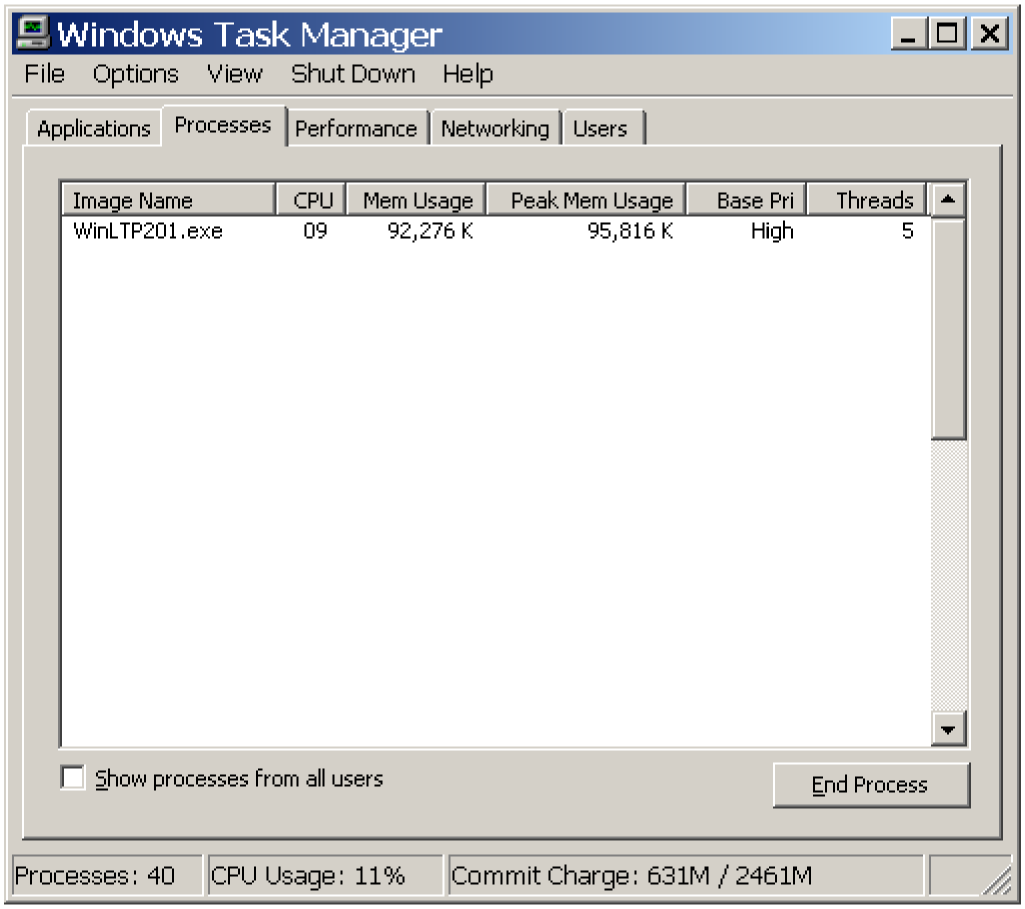


Image Name	CPU	Mem Usage	Peak Mem Usage	Base Pri	Threads
WinLTP201.exe	09	92,276 K	95,816 K	High	5

Processes: 40 | CPU Usage: 11% | Commit Charge: 631M / 2461M

Fig. 2.16.1. Windows Task Manager showing CPU usage and Memory Usage, Peak Memory Usage (and in this example also Base Priority and the number of Threads WinLTP is using). Note: all the other processes have be ‘whited out’ for clarity.

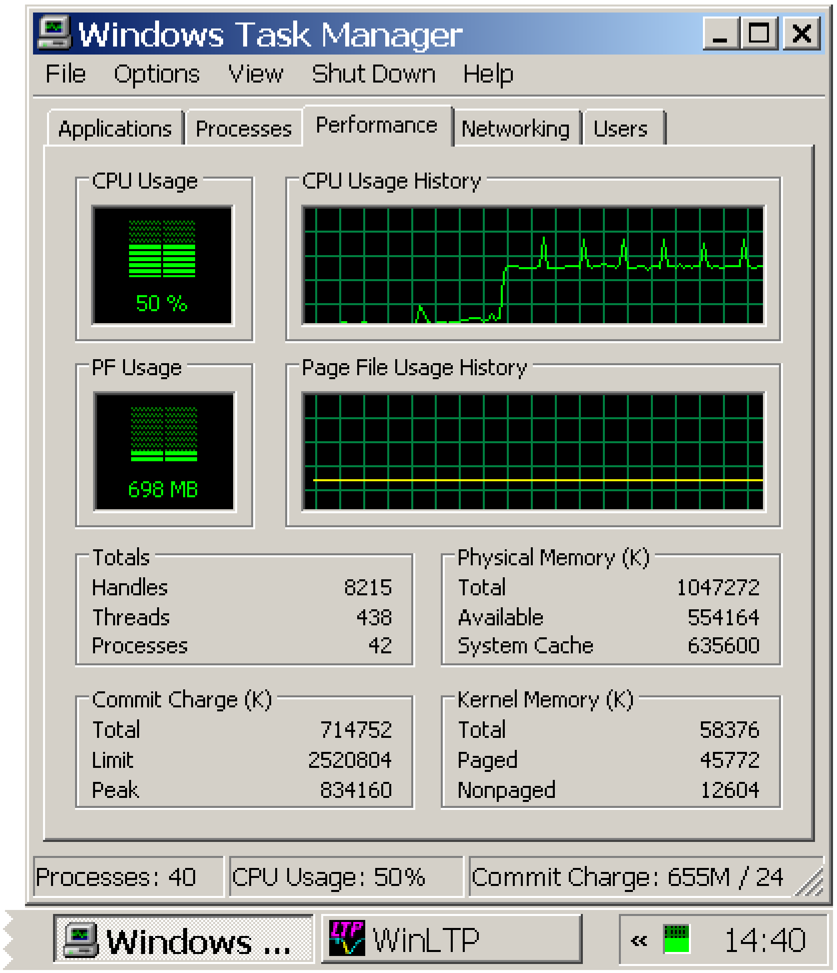
CPU usage can also be seen graphically by clicking on the **Performance** tab (Fig. 2.16.2). Then use the menu command:

View -> CPU History -> One Graph, All CPUs

to make sure the CPU Usage history graphs is one graph (with Windows XP, Vista or Windows 7, a Pentium CPU with HyperThreading can appear as two CPUs, but it is in fact one CPU).

Fig. 2.16.2 shows CPU usage increase when the Main Protocol is started and reaches a relatively high **CPU usage** of about **50%** when running WinLTP with a Digidata 1322A board on a 3.2 GHz computer (2AD channels at 50us sampling interval, Repeat Sweeps of longer **5 sec** duration with **no delay** between them, and Continuous Acquisition, with plotting on screen).

Also note the **small green CPU Usage Graph** on the right of the Task Bar which concisely shows CPU usage as a % full usage. **This can be kept on screen all the time to show CPU usage** (with the Task Manager minimized).



Totals	
Handles	8215
Threads	438
Processes	42

Physical Memory (K)	
Total	1047272
Available	554164
System Cache	635600

Commit Charge (K)	
Total	714752
Limit	2520804
Peak	834160

Kernel Memory (K)	
Total	58376
Paged	45772
Nonpaged	12604

Processes: 40 CPU Usage: 50% Commit Charge: 655M / 24

Fig. 2.16.2. Windows Task Manager showing CPU usage graphically after the MainProtocol was started. Note the small green CPU Usage Graph on the right of the Task Bar.

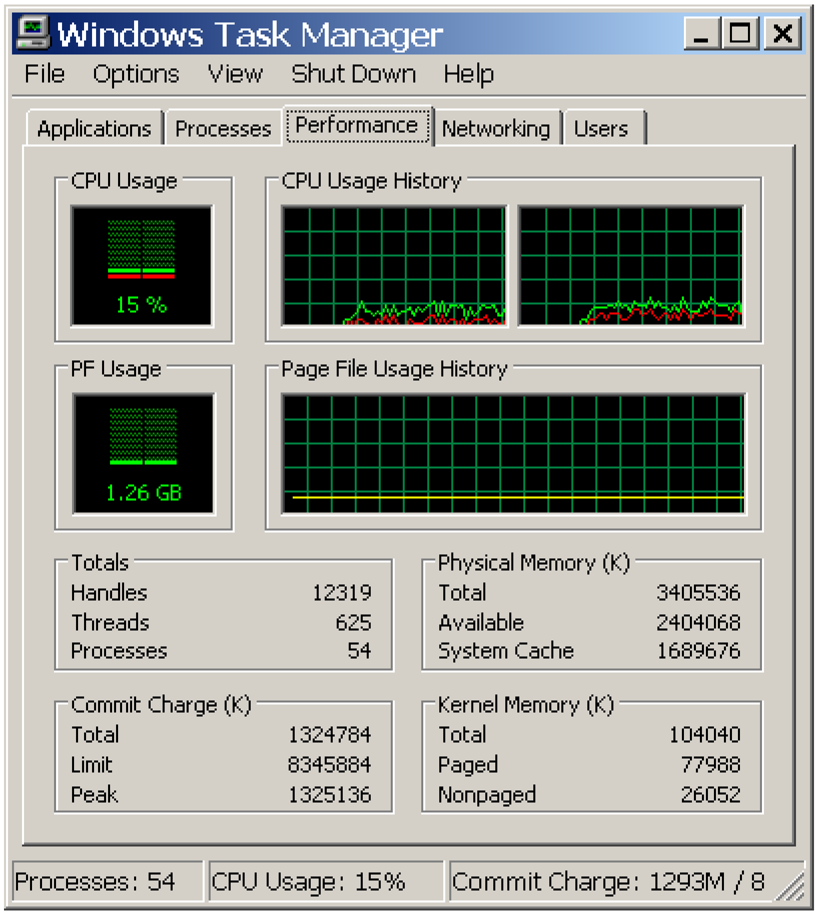
Alternatively, CPU usage can also be seen graphically by clicking on the **Performance** tab (Fig. 2.16.2) and then use the menu command:

View -> CPU History -> One Graph per CPU

to view how each CPU of a multicore processor is being used (again, Windows XP, Vista or Windows 7, a Pentium CPU with HyperThreading will appear as two CPUs).

Fig. 2.16.3 shows CPU usage increase when the Main Protocol is started and reaches a moderately low total **CPU usage** of about **15%** when running WinLTP with a PCI M-Series board on a 2.4 GHz dual-core computer (2AD channels at 50us sampling interval, Repeat Sweeps of 0.1 sec duration with **no delay** between them, and Continuous Acquisition, with plotting on screen). Note the substantially smaller CPU usage compared to Fig. 2.16.2.

If a quad-core processor was used, the CPU Usage History would contain 4 panels, one for each CPU.



Totals

Handles	12319
Threads	625
Processes	54

Physical Memory (K)

Total	3405536
Available	2404068
System Cache	1689676

Commit Charge (K)

Total	1324784
Limit	8345884
Peak	1325136

Kernel Memory (K)

Total	104040
Paged	77988
Nonpaged	26052

Processes: 54 CPU Usage: 15% Commit Charge: 1293M / 8

Fig. 2.16.3. Windows Task Manager showing usage of 2 CPUs in a dual-core processor graphically after the MainProtocol was started.

2.17 Ways to speed up WinLTP on slower computers

There are a few ways of speeding up WinLTP on slower computers without buying a new computer. Basically, do a dry run of your experimental protocol including all stimulation and saving data to disk before actually running the experiment. This is particularly important if you are using capabilities of WinLTP that are particularly taxing to the computer: a) if you are doing LTD type experiments where ADsweeps are saved rapidly, and there is no delay between sweeps, or b) if you are analyzing a large number of synaptic potentials suing fast repetitive sweeps.

1. Make sure you are running only those capabilities of WinLTP that you need. **Only plot and/or save Continuously Acquired data if you need to!** Continuous Acquisition plotting generally takes more CPU power than saving to disk. Only sample at the frequency you need, which may be less than the default 20 KHz sampling rate. Only plot and save those AD channels you need.

2. When you analyze a large number of synaptic potentials (say over 100) using fast repetitive sweeps on-line, this is particularly taxing for the computer. So, perhaps, **analyze only the first synaptic potential while on-line, and then later reanalyze all the synaptic potentials off-line**.

3. Unhook your Ethernet cable connecting your computer to the network, and then **temporarily disable the On-Access capability of your anti-virus program.** If you look at CPU usage by different processes

(e.g. Fig. 2.16.1), you may find that your anti-virus program may be using half the CPU power to check that the WinLTP files being saved (*.ABF and *.P0 etc. files) do not contain a virus. The chances that they do are practically nil, particularly for the ASCII *.P0 etc. files. If disabling your anti-virus program makes a difference, you can re-enable the On-Access capability of your anti-virus program, but **exclude anti-virus scanning of specific file types ABF and P0 etc.** with your anti-virus program.

4. If you have only 512 MB of RAM memory, try **increasing RAM memory to 2 or even 4 GB**.

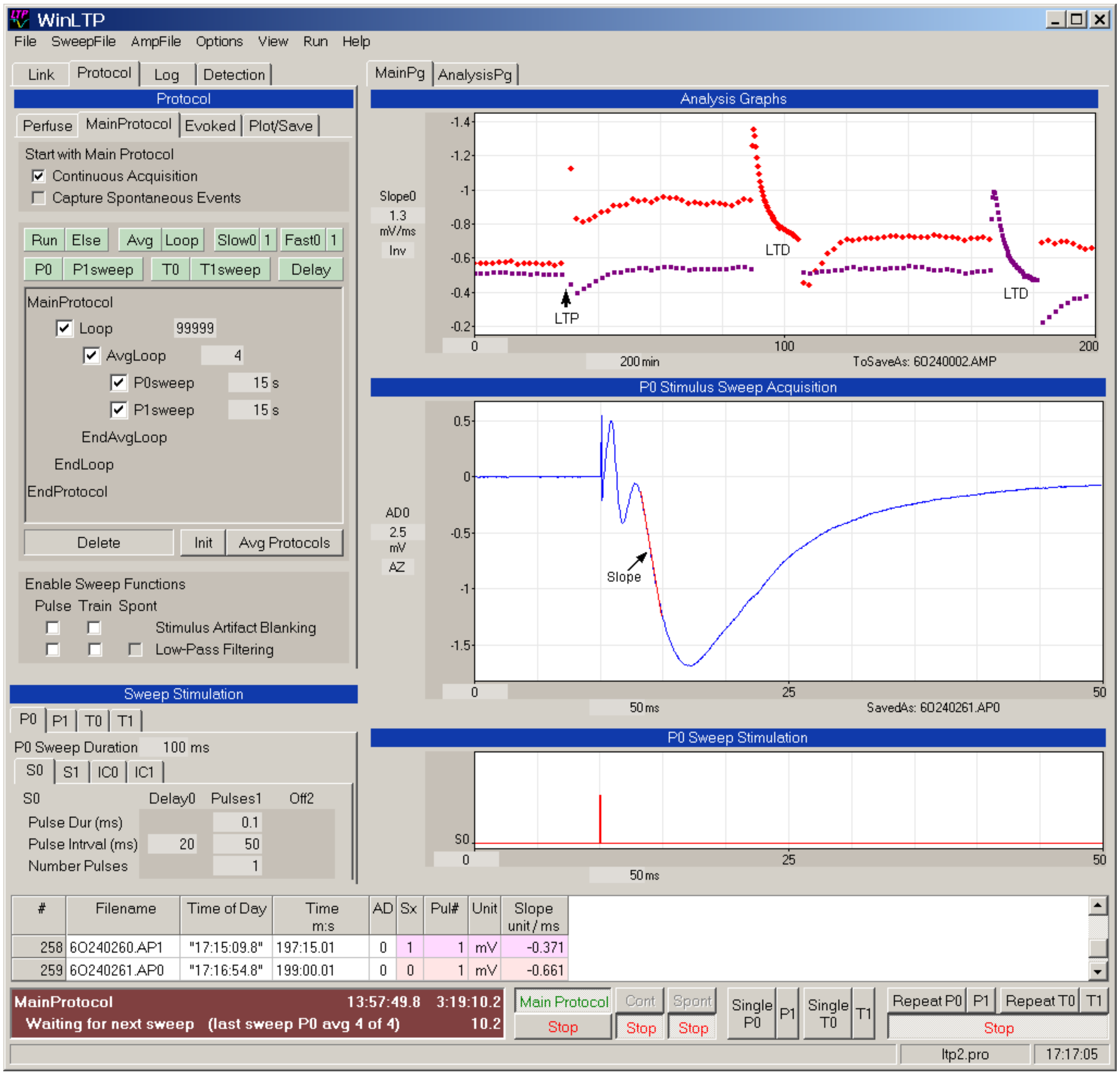
Basically, I determine if a computer is **minimally** fast enough by seeing if it can keep up with Continuous Acquisition on, 2AD channels used, and at 20 KHz, and with Repeat Sweeps on using fast **1 Hz Repeat Sweeps of 1 sec duration** so there is no delay between sweeps, 2 AD channels used, and at 20 KHz, with 1 pulse stimulation and analysis. This is basically the default.pro protocol, but with the P0 sweep period decreased to 1 sec. You may need quite a bit less.

A further speed test I now use to determine if the computer is substantially fast enough is by seeing if it can keep up with Continuous Acquisition on, 2AD channels used, and at 20 KHz, and with Repeat Sweeps on using very fast **10 Hz Repeat Sweeps of 0.1 sec duration** so there is no delay between sweeps, 2 AD channels used, and at 20 KHz, with 1 pulse stimulation and analysis.

CHAPTER 3 – Organization of WinLTP

3.1 Tabsheet and Panel Areas

WinLTP is organized in multiple tabsheets and panels. Fig. 3.1.1 shows the basic WinLTP tabsheet and panel layout for a basic LTP experiment. Win LTP is divided into seven areas, the Menu area (top), the Protocol/Detection tabsheet area (upper left), the MainPg/AnalysisPg tabsheet area (upper right), the Sweep Field Stimulation area (middle left), the Spreadsheet area (below the Sweep Field Stimulation and MainPg/AnalysisPg areas), the Run Panel/Button area (below the Spreadsheet area), and the Status Bar area (bottom)



Start with Main Protocol
- [✓] Continuous Acquisition
- [ ] Capture Spontaneous Events

```
MainProtocol
    [✓] Loop          99999
        [✓] AvgLoop       4
            [✓] P0sweep    15 s
            [✓] P1sweep    15 s
        EndAvgLoop
    EndLoop
EndProtocol
```

Enable Sweep Functions

Pulse	Train	Spont	
☐	☐		Stimulus Artifact Blanking
☐	☐	☐	Low-Pass Filtering

P0 Sweep Duration 100 ms

S0	Delay0	Pulses1	Off2
Pulse Dur (ms)		0.1	
Pulse Intrval (ms)	20	50	
Number Pulses		1	

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	Slope unit / ms
258	60240260.AP1	"17:15:09.8"	197:15.01	0	1	1	mV	-0.371
259	60240261.AP0	"17:16:54.8"	199:00.01	0	0	1	mV	-0.661

MainProtocol 13:57:49.8 3:19:10.2
Waiting for next sweep (last sweep P0 avg 4 of 4) 10.2

Fig. 3.1.1. Basic WinLTP tabsheet and panel layout (in a basic LTP experiment).

3.1.1 MainPg/AnalysisPg Tabsheet Area

The upper right corner of the program shows the MainPg and AnalysisPg tabsheets (Fig. 3.1.1.1). The MainPg tabsheet consists of three areas, the Analysis Graphs panel (top here, but can also be placed on the right) containing (currently) one to four Analysis graphs, the Acquisition area (middle) showing here only one (P0) Stimulus Sweep Acquisition, although it can also show Continuous Acquisition, Spontaneous Sweep Acquisition, and some or all the other stimulation sweeps (P1, T0 and T1 sweeps), and the graph of the stimulation in the sweep (bottom).

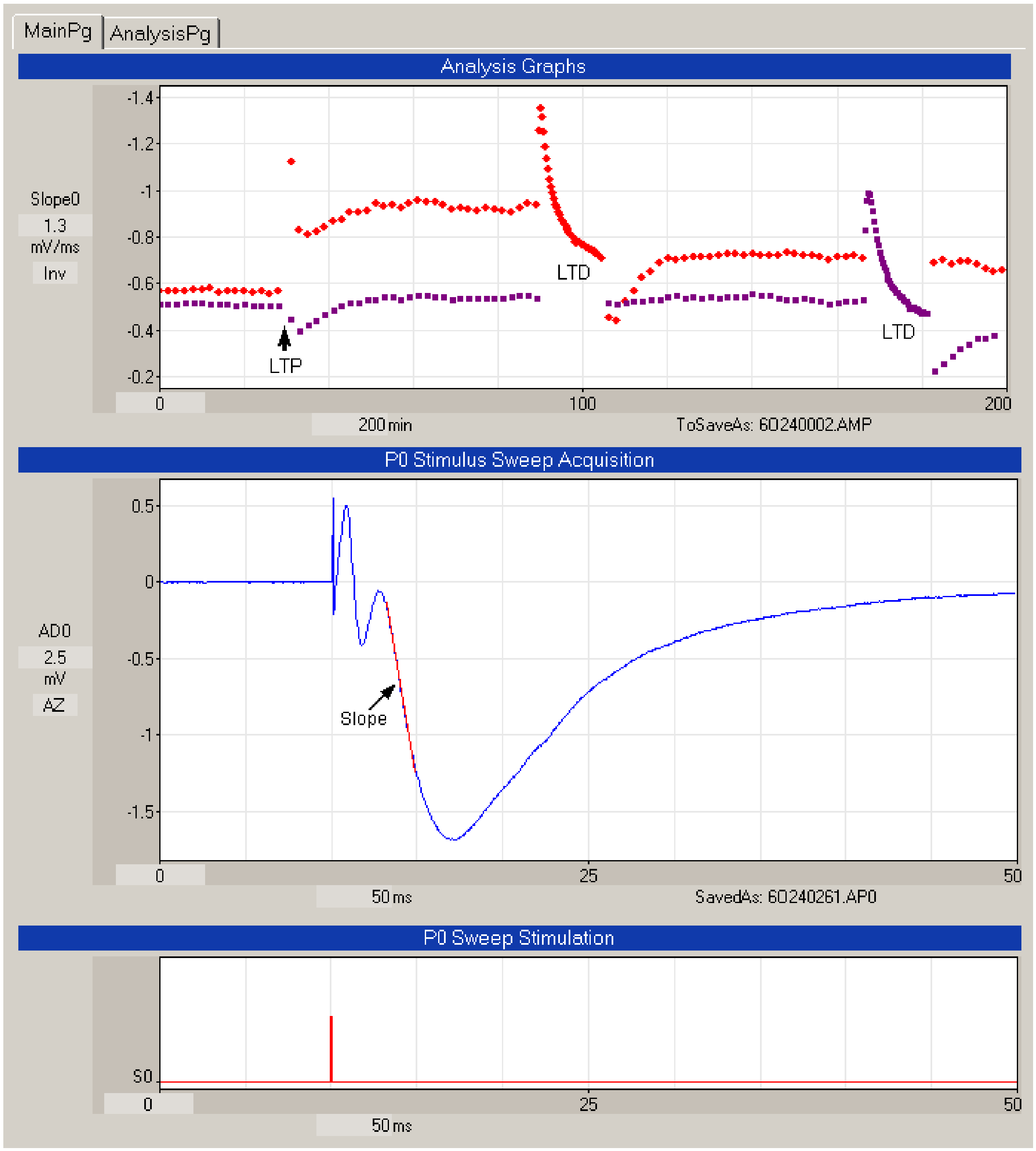
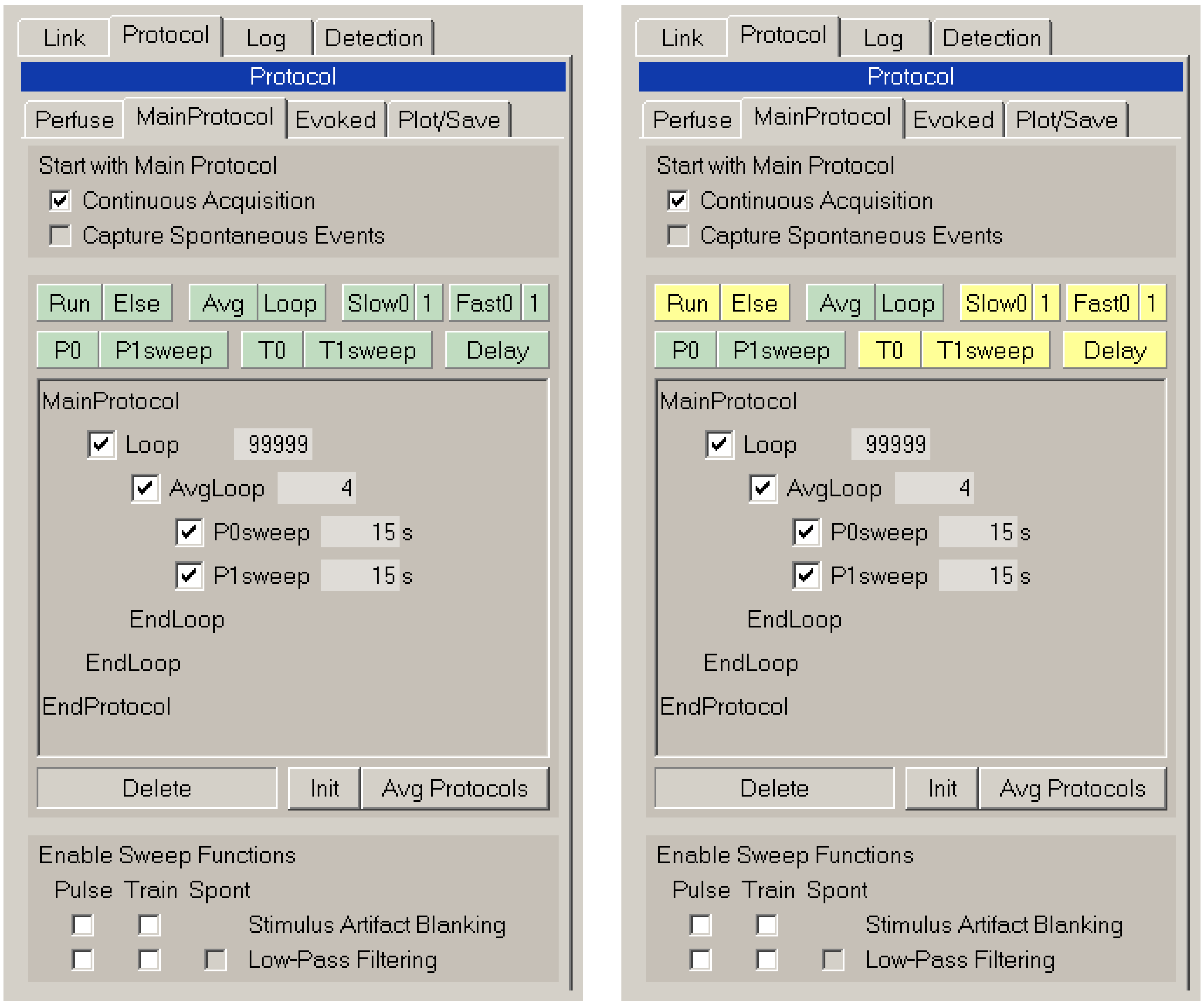


Fig. 3.1.1.1. The MainPg/AnalysisPg tabsheet area. The MainPg contains Analysis graphs (top here, although the can be on the right), Stimulus Sweep Acquisition graphs (middle), and Sweep Stimulation graphs (bottom).

The AnalysisPg contains one or two columns of Analysis graphs with one to four Analysis graphs in each column (however, in this version the AnalysisPg is not implemented).

3.1.2 Link/Protocol/Log/Detection Tabsheet Area

The upper left corner of the program shows the Link/Protocol/Log/Detection tabsheet area containing the **Link** tabsheet (Fig. 8.1.1), the **Protocol** tabsheet (Fig. 3.1.2.1), the **Log** tabsheet (Fig. 10.1.1), and the **Detection** tabsheet (Fig. 3.1.2.3).



Start with Main Protocol
☑ Continuous Acquisition
☐ Capture Spontaneous Events

```
MainProtocol
    ☑ Loop        99999
        ☑ AvgLoop      4
            ☑ P0sweep    15 s
            ☑ P1sweep    15 s
        EndLoop
    EndLoop
EndProtocol
```

Enable Sweep Functions

Pulse	Train	Spont	
☐	☐		Stimulus Artifact Blanking
☐	☐	☐	Low-Pass Filtering

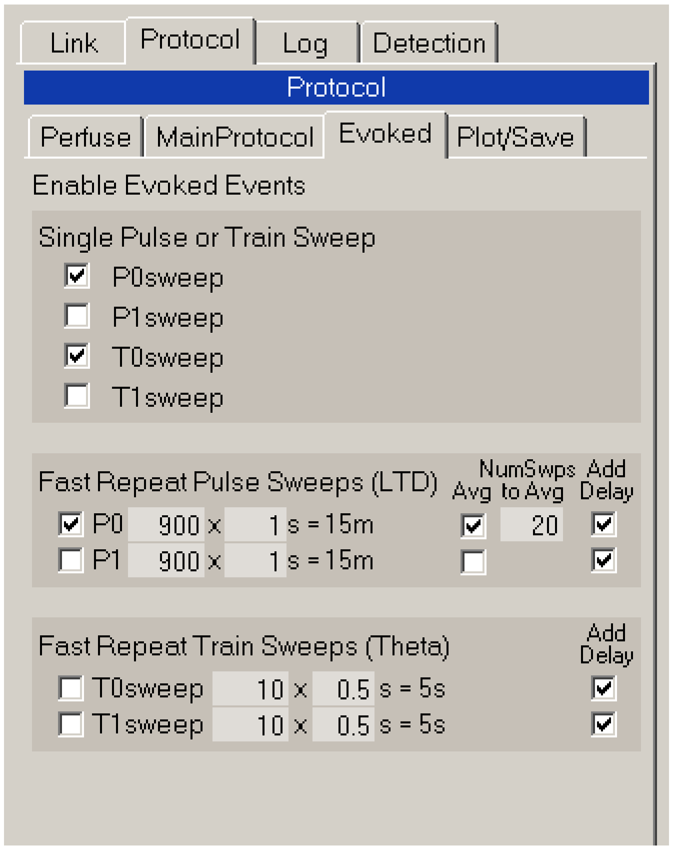
Fig. 3.1.2.1. The Protocol tabsheet in the Advanced Mode (including DemoTrial period) fully functional ‘Protocol Builder’ (left), and (with the same protocol) in the Basic Mode partially functional ‘Protocol Builder’ (right).

The **Link** tabsheet is discussed in Chapter 8.

The **Protocol** tabsheet then has four 'sub' tabsheets consisting of the **Perfuse** tabsheet (Fig. 9.2.7.2), the **MainProtocol** tabsheet (Fig. 3.1.2.1), the **Evoked** Events tabsheet (Fig. 3.1.2.2, left), and the **Plot/Save** tabsheet (Fig. 3.1.2.2, right).

The MainProtocol tabsheet then has three panels the **'Start with Main Protocol'** panel (top), the **'Protocol Builder'** panel (middle), and the **'Enable Sweep Functions'** panel (bottom).

The Protocol Builder panel in turn contains **Insert Buttons** (top), the **Script area** (middle) and a **Delete/Init buttons** panel at the bottom. In this bottom panel, clicking on an **Init** Protocols button initializes the Protocol Builder script to continuous non-averaging loops, and clicking on an Init **Avg Protocols** button initializes the script to continuous averaging loops (as shown in Fig. 3.1.2.1). Note that the Protocol Builder can appear in two modes, the **fully functional** Protocol Builder with **green** Insert buttons in the Advanced Mode (including during the DemoTrial period) (Fig. 3.1.2.1, left), and a partially functional Protocol Builder with **green and yellow** Insert buttons in the Basic Mode (Fig. 3.1.2.1, right).



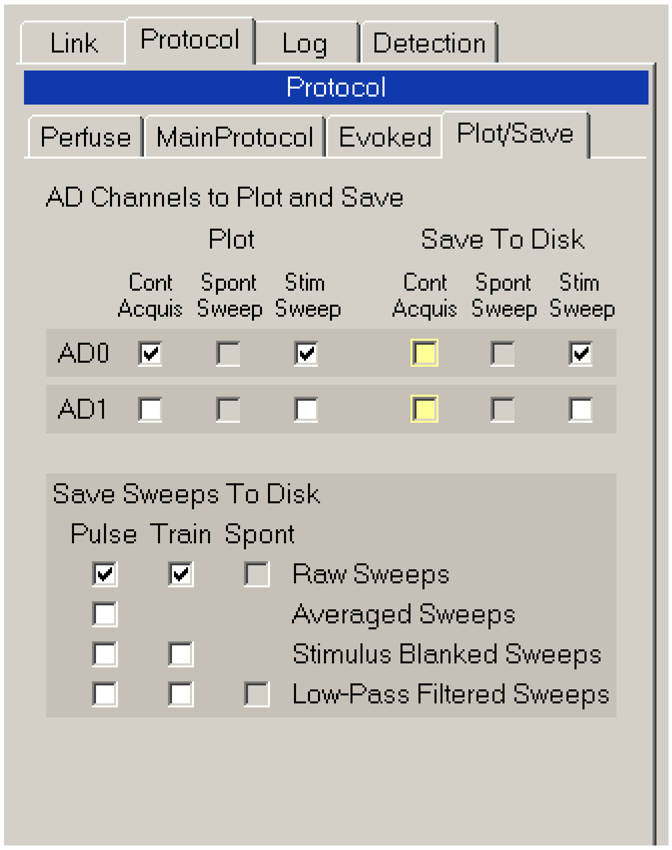
Enable Evoked Events

Single Pulse or Train Sweep

✓	P0sweep
☐	P1sweep
✓	T0sweep
☐	T1sweep

Fast Repeat Pulse Sweeps (LTD)				Avg	NumSwps to Avg	Add Delay
✓	P0	900 x	1 s = 15m	✓	20	✓
☐	P1	900 x	1 s = 15m	☐		✓

Fast Repeat Train Sweeps (Theta)				Add Delay
☐	T0sweep	10 x	0.5 s = 5s	✓
☐	T1sweep	10 x	0.5 s = 5s	✓



AD Channels to Plot and Save

	Plot			Save To Disk		
	Cont Acquis	Spont Sweep	Stim Sweep	Cont Acquis	Spont Sweep	Stim Sweep
AD0	✓	☐	✓	🟨	☐	✓
AD1	☐	☐	☐	🟨	☐	☐

Save Sweeps To Disk

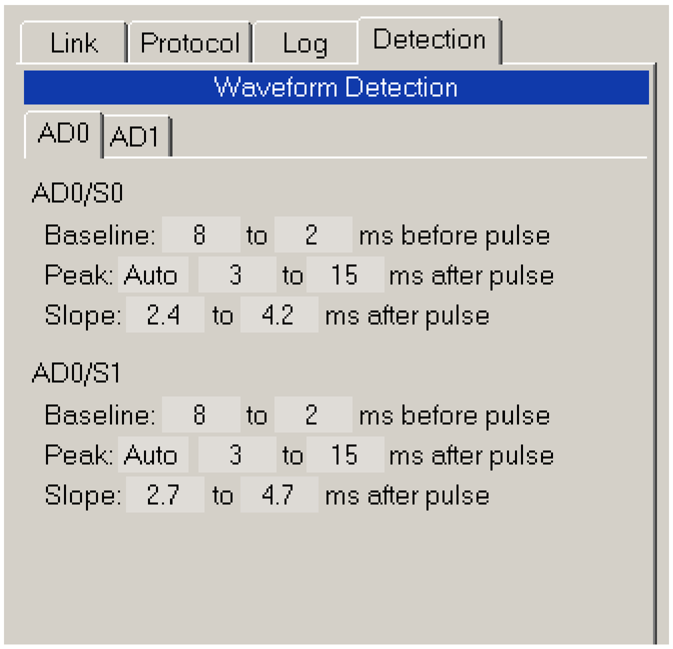
Pulse	Train	Spont	
✓	✓	☐	Raw Sweeps
☐			Averaged Sweeps
☐	☐		Stimulus Blanked Sweeps
☐	☐	☐	Low-Pass Filtered Sweeps

Fig. 3.1.2.2. The EvokedEvents 'sub' tabsheet (left), and the Plot/Save 'sub' tabsheet (right) of the Protocol tabsheet. Note that in the Basic Mode, the 'Cont Acquis' 'Save To Disk' checkboxes are marked in yellow indicating that Continuous Acqusition data cannot be saved in the Basic Mode. In the Advanced Mode, , the 'Cont Acquis' 'Save To Disk' checkboxes are colored normally.

The **EvokedEvents** 'sub' tabsheet has three panels, the enable **Single Train or Pulse Sweep** panel (top), the **Fast Repeat Pulse Sweeps** panel for evoking LTD stimulation (middle), and the **Fast Repeat Train Sweeps** panel for evoking theta burst stimulation.

The **Plot/Save** ‘sub’ tabsheet has two panels, which **AD Channels to Plot and Save** panel (top) and a **Save Sweeps to Disk** panel (bottom). The AD Channels to Plot and Save panel determines which AD channels (AD0 and/or AD1) and for which modes (Continuous Acquisition, Spontaneous Sweep and/or Stimulation/Acquisition Sweep) for either Plotting and/or Saving To Disk. The Save Sweeps to Disk panel determines for Pulse/Train Stimulation/Acquisition sweeps whether to save Raw, Averaged, Stimulus Blanked and/or Low-Pass Filtered Sweeps. For Spontaneous Sweeps it determines whether to save Raw and/or Low-Pass Filtered Sweeps.

The **Detection** tabsheet (Fig. 3.1.2.3) has two ‘sub’ tabsheets, the **AD0** and **AD1** tabsheets. The AD0 and AD1 tabsheets are divided into **S0** and **S1** areas for detecting stimulation on channels AD0 and AD1 for S0 and S1 extracellular stimulation pulses. These can include Baseline, Peak and Slope lines shown in Fig. 3.1.2.3, but can also include Duration, Coastline, PopSpike, and Average Amplitude lines (see also Section 4.9).



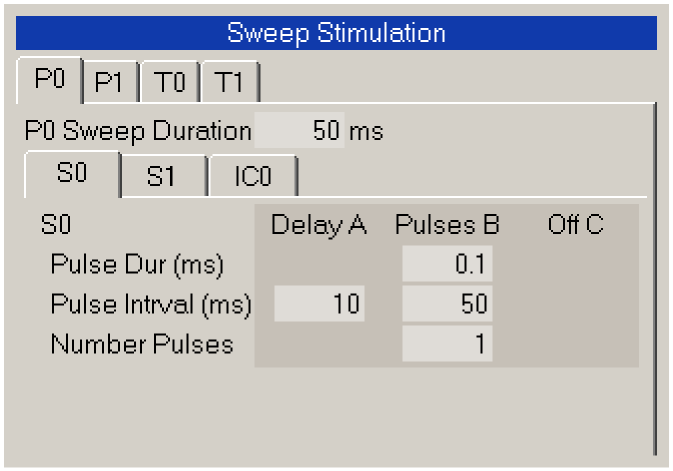
AD0/S0
Baseline: 8 to 2 ms before pulse
Peak: Auto 3 to 15 ms after pulse
Slope: 2.4 to 4.2 ms after pulse

AD0/S1
Baseline: 8 to 2 ms before pulse
Peak: Auto 3 to 15 ms after pulse
Slope: 2.7 to 4.7 ms after pulse

Fig. 3.1.2.3.. Detection tabsheet containing both the AD0 and AD1 ‘sub’ tabsheets.

3.1.3 Sweep Stimulation Field Area

The middle left area of the program shows the Sweep Stimulation field area, the area where value fields that control stimulation values for each sweep, P0, P1, T0 and T1, are located. For instance, Fig. 3.1.3.1 shows P0 Sweep Stimulation that includes Sweep Duration and S0 Pulse Duration, Interval and Number in the S0 ‘sub’ tabsheet. In addition to the S0 ‘sub’ tabsheet, there is the S1 and IC0 (IntraCellular Ch 0) ‘sub’ tabsheet stimulations.


Sweep Stimulation

P0 | P1 | T0 | T1

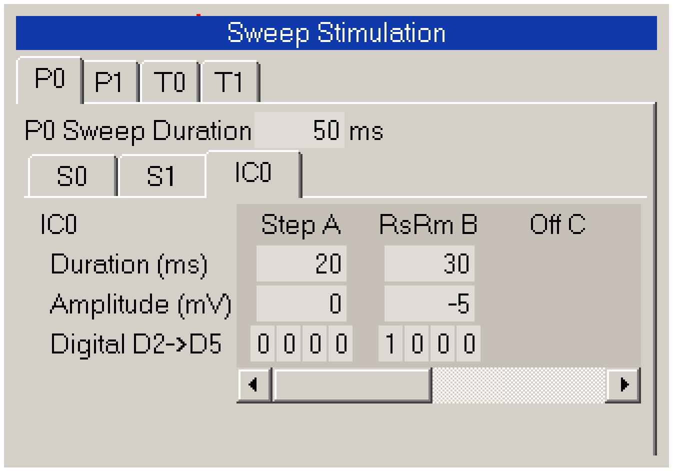
P0 Sweep Duration 50 ms

S0 | S1 | IC0

S0	Delay A	Pulses B	Off C
Pulse Dur (ms)		0.1	
Pulse Intrval (ms)	10	50	
Number Pulses		1	

Fig. 3.1.3.1. Sweep Stimulation Field Area showing S0 pulse stimulation in a P0sweep.

A different example of IC0 stimulation (not related to the Fig. 3.1.1 example) is shown in Fig. 3.1.3.2.


Sweep Stimulation

P0 | P1 | T0 | T1

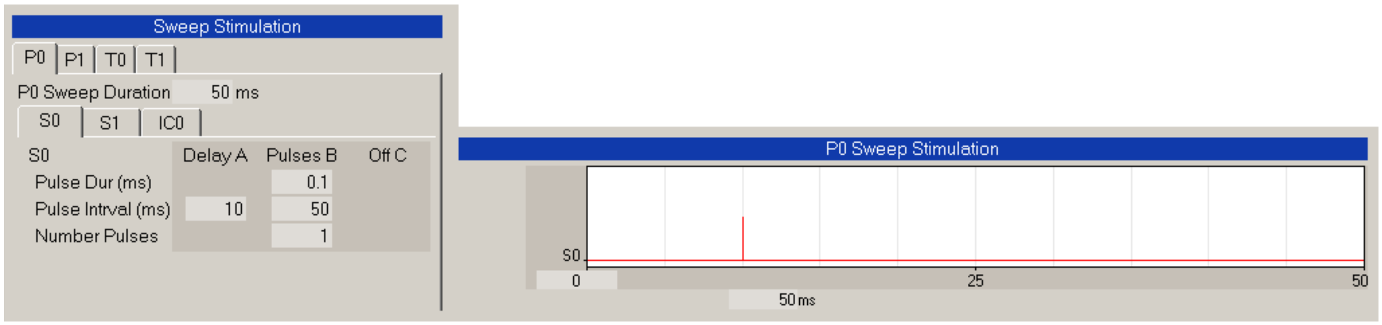
P0 Sweep Duration 50 ms

S0 | S1 | IC0

IC0	Step A	RsRm B	Off C
Duration (ms)	20	30	
Amplitude (mV)	0	-5	
Digital D2->D5	0 0 0 0	1 0 0 0	

Fig. 3.1.3.2. IC0 stimulation fields.

Note that the Field Sweep Stimulation area is functionally coupled with the Graph Stimulation area, so that when you click on the P0 tabsheet in the Field Sweep Stimulation area, the P0 Sweep Stimulation graph comes up (Fig. 3.1.3.3). And when you click on the T0 tabsheet in the Field Sweep Stimulation area, the T0 Sweep Stimulation graph comes up, and so forth.


Sweep Stimulation

P0 | P1 | T0 | T1

P0 Sweep Duration 50 ms

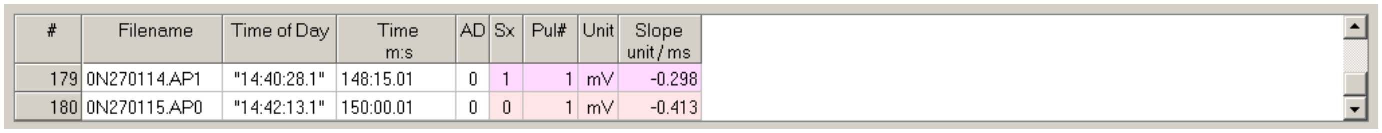
S0 | S1 | IC0

S0	Delay A	Pulses B	Off C
Pulse Dur (ms)		0.1	
Pulse Intrval (ms)	10	50	
Number Pulses		1	

Fig. 3.1.3.3. Field and Graph Sweep Stimulation Areas are coupled.

3.1.4 Spreadsheet Area

The Spreadsheet contains from zero number of lines to the maximum number of lines that can be visible (Fig. 3.1.4.1).



#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	Slope unit / ms
179	0N270114.AP1	"14:40:28.1"	148:15.01	0	1	1	mV	-0.298
180	0N270115.AP0	"14:42:13.1"	150:00.01	0	0	1	mV	-0.413

Fig. 3.1.4.1. Spreadsheet. Only the Slope analysis column is shown.

The columns include

a) Calculation Number
b) **Filename** of the ADsweep data file
c) **Time Of Day** (in hr:min:sec) where hr increases to 25 after 24, not 0
d) **Time** on the Analysis graph the data point was taken (in Min:Sec.UpToTenthsOfMsec)
e) AD channel number (0 or 1)
f) Extracellular Stimulation S0 or S1 (0 or 1)
g) **Pulse Number** (starting with 1)
h) **Unit or DataType** (mV or pA) for those analyses using units
g) Analyses (from none to all the following columns)
  1) DC Baseline "DC"
  2) Peak Amplitude "PkAmp"
  3) Latency "Latency ms"
  4) Area "Area unit * ms"
  5) Duration "Dur ms”
  6) Rise Time "RiseTm ms”
  7) Decay Time "DecTm ms"
  8) Coastline "CoastLn"
  9) PopSpike Amplitude "PSamp"
  10) PopSpike Latency "PSlat ms"
  11) Slope "Slope unit/ms"
  12) Average Amplitude "AvgAmp"
  13) Patch Electrode Series Resistance "Rs Mohm"
  14) Membrane or Input Resistance "Rm Mohm"

3.1.5 Run Panel / Run Button Area

Acquisition activity can be started by either clicking on different Run Buttons on the Run Panel (Fig. 3.1.5.1). Click on 1) the ‘Main Protocol’ Run Button to start the Main Protocol with Stimulation/Acquisition sweeps with or without Continuous Acquisition, 2) click on the ‘Cont’ Run Button to start Continuous Acquisition only, or 3) click on a ‘Single P0, ‘P1’, ‘Single T0’ or ‘T1’ Run Button to immediately acquire a single sweep for viewing with no analyzing or saving to disk. Once the Main Protocol has started, clicking a ‘Single P0, ‘P1’, ‘Single T0’ or ‘T1’ Run Button now evokes the sweep, but also may analyze it and save it to disk. Once the Main Protocol has been started, clicking on a ‘Repeat P0’, ‘P1’, ‘Repeat T0’, ‘T1’ Run Button now evokes repeating sweeps that also may be analyzed and saved to disk.

Once the Main Protocol is started, the "Protocol STOPPED" top Run Line changes to "Main Protocol" followed by the **Time of Day the protocol was started** (in hr:min:sec:tenthsec) followed by the **time the protocol was running** (in hr:min:sec:tenthsec). The bottom Run Line shows "Waiting for next sweep", followed by the last sweep taken and possibly how many sweeps out of how many total sweeps were averaged if in an averaging loop, followed by the **time after the last sweep** (in hr:min:sec:tenthsec) on the right side of the lower Run Line.

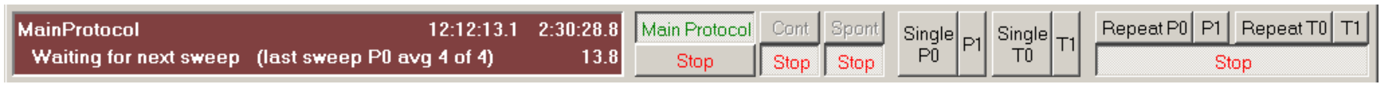

MainProtocol 12:12:13.1 2:30:28.8
Waiting for next sweep (last sweep P0 avg 4 of 4) 13.8

Fig. 3.1.5.1. Run Panel (left) and Run Buttons (right).

3.1.6 Status Bar

The Status Bar area (Fig. 3.1.6.1) is divided into three regions: 1) the left **Status or Message Line** (Green means successful operation message, Yellow means warning message, Red means error message), 2) the protocol file name running (*.pro), and 3) the Time of Day.

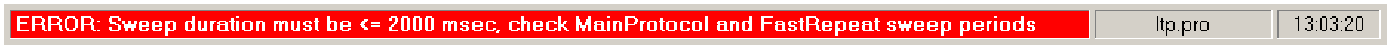

ERROR: Sweep duration must be <= 2000 msec, check MainProtocol and FastRepeat sweep periods
ltp.pro
13:03:20

Fig. 3.1.6.1. Status Bar.

3.2 Menus

Win LTP uses the following Menus:

1) Protocol **<u>F</u>ile** Menu (Fig. 3.2.1)
2) **<u>S</u>weepFile** Menu for loading, saving and printing individual ADsweep files (Fig. 3.2.2)
3) Amplitude or **<u>A</u>mpFile** Menu for performing sweep analysis and reanalysis (Fig. 3.2.3)
4) **<u>O</u>ptions** Menu (Fig. 3.2.4)
5) **<u>V</u>iew** Menu (Fig. 3.2.5)
5) **<u>R</u>un** Menu - an alternative to running protocols by pushing function keys (Fig. 3.2.6)
6) **<u>H</u>elp** Menu (Fig. 3.2.7)

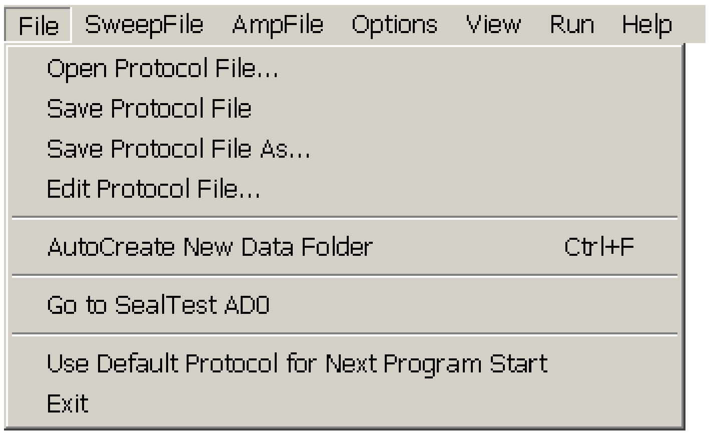


Fig. 3.2.1. Protocol File Menu

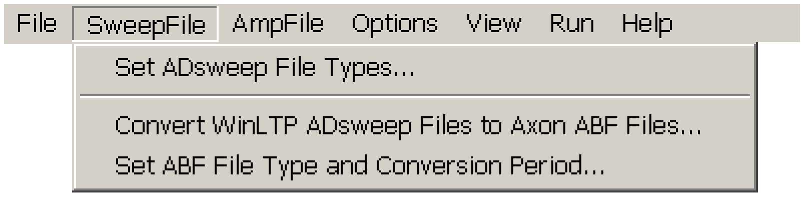

File
SweepFile
AmpFile
Options
View
Run
Help
Set ADsweep File Types...
Convert WinLTP ADsweep Files to Axon ABF Files...
Set ABF File Type and Conversion Period...

Fig. 3.2.2. SweepFile Menu.

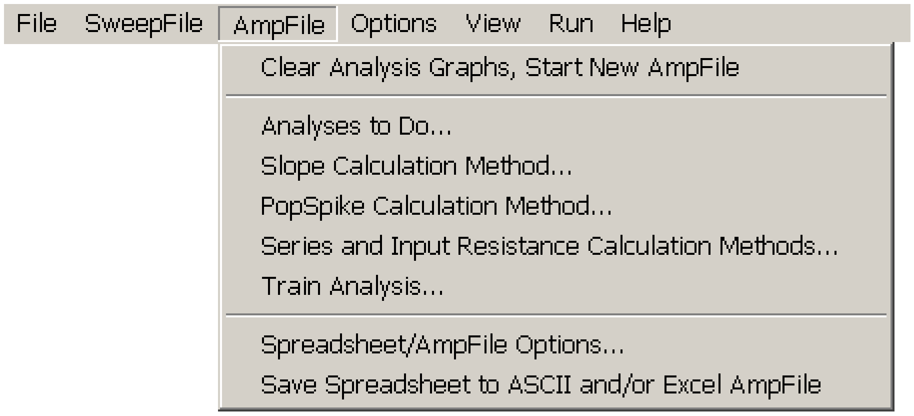

File
SweepFile
AmpFile
Options
View
Run
Help
Clear Analysis Graphs, Start New AmpFile
Analyses to Do...
Slope Calculation Method...
PopSpike Calculation Method...
Series and Input Resistance Calculation Methods...
Train Analysis...
Spreadsheet/AmpFile Options...
Save Spreadsheet to ASCII and/or Excel AmpFile

Fig. 3.2.3. Amplitude File Menu.

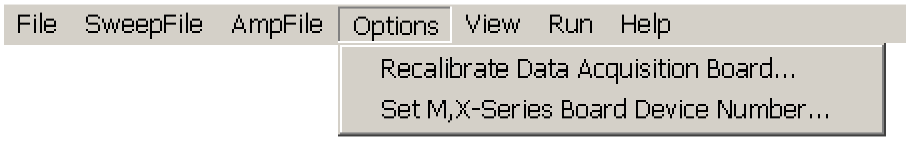

File
SweepFile
AmpFile
Options
View
Run
Help
Recalibrate Data Acquisition Board...
Set M,X-Series Board Device Number...

Fig. 3.2.4. Options Menu (The Set M,X-Series Board Device Number only appears when running the M- or X-Series National Instruments boars).

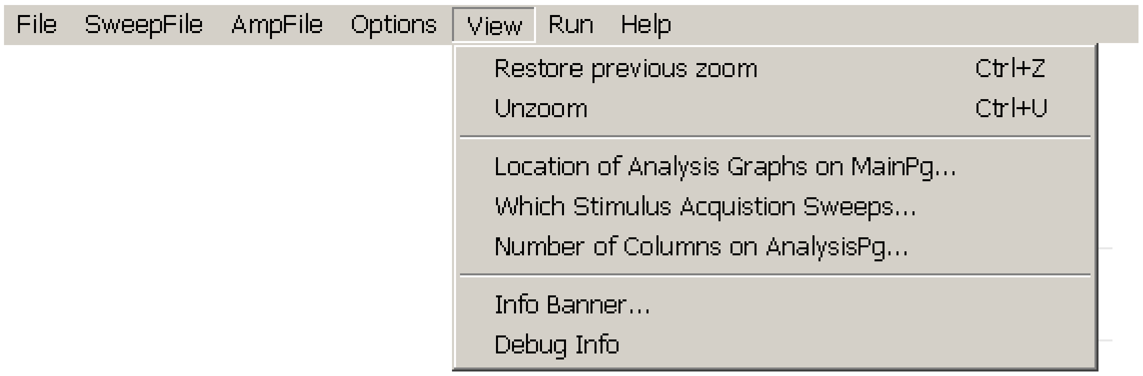

File
SweepFile
AmpFile
Options
View
Run
Help
Restore previous zoom Ctrl+Z
Unzoom Ctrl+U
Location of Analysis Graphs on MainPg...
Which Stimulus Acquistion Sweeps...
Number of Columns on AnalysisPg...
Info Banner...
Debug Info

Fig. 3.2.5. View Menu.

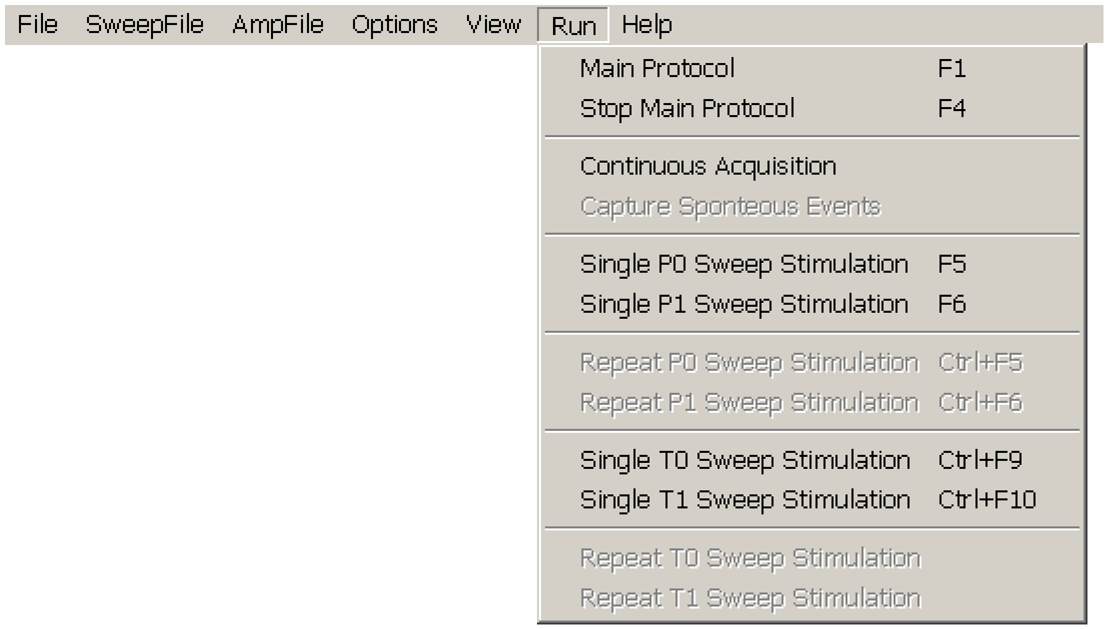

File
SweepFile
AmpFile
Options
View
Run
Help
Main Protocol F1
Stop Main Protocol F4
Continuous Acquisition
Capture Sponteous Events
Single P0 Sweep Stimulation F5
Single P1 Sweep Stimulation F6
Repeat P0 Sweep Stimulation Ctrl+F5
Repeat P1 Sweep Stimulation Ctrl+F6
Single T0 Sweep Stimulation Ctrl+F9
Single T1 Sweep Stimulation Ctrl+F10
Repeat T0 Sweep Stimulation
Repeat T1 Sweep Stimulation

Fig. 3.2.6. Run Menu.

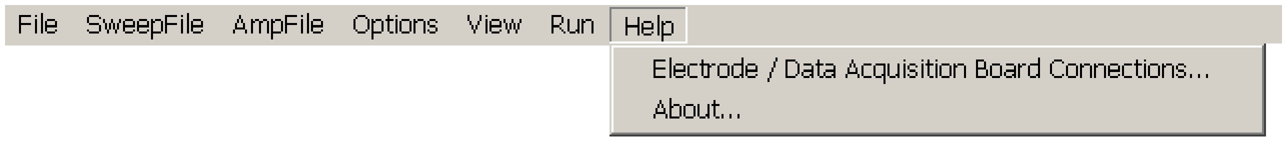


Fig. 3.2.7. Help Menu.

3.3 Running Protocols using Run Buttons, Function Keys and Run Menus

Protocols can be started by either clicking a Run Button, pressing a Function Key, or by choosing the correct line in the **Run** menu (Fig. 3.2.6). The Run menu gives a description of the protocol on the left side of the line, and the shortcut Function Key that can be alternatively pressed to start that protocol.

3.4 Fields – Changing Values

Fields can be selected by Tabbing or Shift-Tabbing (reverse) into them, and are indicated selected by turning dark blue. Selected field values can be changed by:

1) Entering values using the keyboard and pressing the Enter key
2) Pressing the **Left** Mouse Button to **increment** a value, and pressing the Right Mouse Button to decrement a value.
3) Moving the Mouse Thumbwheel **forward** to **increment** a value, and moving the Mouse Thumbwheel backwards to decrement a value.
4) Also, the **amount** of increment and decrement caused by pressing the Left or Right Mouse Button, or moving the Mouse Thumbwheel forward or backward can be:
   a) **Increased** by also simultaneously pressing the **Shift** key, and
   b) **Decreased** by also simultaneously pressing the **Ctrl** key

3.5 Graphs – Changing Axis Values, Dragging, Zooming

It is important to briefly discuss how the **AZ/DC** field in the ADsweep graphs operates. When the field is AZ, or AutoZeroed, the waveform is not high pass filtered, but the first 20th of the waveform (or the first 100 msec, whichever is shorter) is displayed at 0 mV or 0 pA. This produces an ersatz 'AC coupling' or AutoZeroing of the waveform, and allows small signals with large slow voltage fluctuations to be displayed in the graph, but the waveform is not at all distorted by high pass filtering and is plotted DC ('Direct Coupled') except for the superimposed offset. When the field is DC, no offsets are imposed on the display of the waveform. The AZ/DC field only affects the **display** of the data and does not affect the data values saved in the ADsweep (*.T0, *.T1, *.P0, *.P1, *.AP0 or *.AP1) files, which are saved as DC values.

In addition to changing the X and Y coordinate values on the graphs by changing the **associated field** values, the graph X, Y coordinates can be changed by using the mouse:

1) Click on the graph data window with the **Left** Mouse Button, hold it down, and **drag** the graph **vertically up or down**.
2) Click on the graph data window with the **Right** Mouse Button, hold it down, and **drag** the graph **horizontally left or right**.

Furthermore, ADsweep and Analysis graphs can be Zoomed and Unzoomed.

1) To **Zoom**, first hold the **Ctrl** key down, then click on the graph data window with the **Left** Mouse Button, hold it down, drag it across the graph **horizontally to the right and vertically down,** and then **release** it to zoom in on a section of the graph. The graph goes from an initial N**on-Zoomed state** with no red label in the lower left axis region (Fig 3.5.1a) to a **Zoomed state** as indicated by the red **“Zoomed”** label in the lower left axis region (Fig. 3.5.1b). This zooming by mouse can be repeated any number of times to zoom further in on the area of interest. And you can further refine the X and Y coordinate values on the graphs by changing the **associated field** values.
2) To **Unzooom**, you can either:

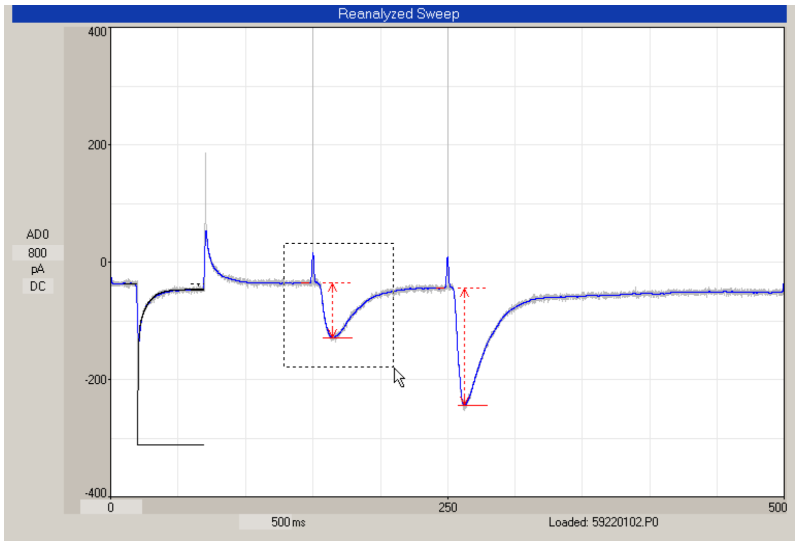
a) first hold the **Ctrl** key down, then click on the graph data window with the **Left** Mouse Button, hold it down, drag it across the graph **horizontally to the left and vertically up,** and then **release** it. This will restore the graph X and Y coordinates to what they were before the zooming took place. The graph will now be in an Unzoomed state as as indicated by the black **“Unzoomed”** label in the lower left axis region (Fig. 3.5.1c).

b) Or, more simply, for ADsweep graphs you can press the **Ctrl-U** key, or click the **View -> Unzoom** menu item. You cannot do this for Analysis graphs.

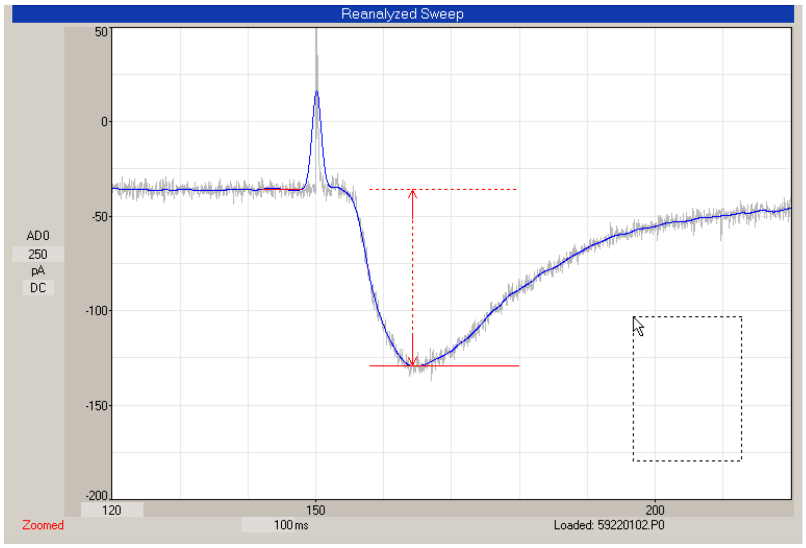
3) For ADsweep graphs you can also toggle back and forth between the **Zoomed** and **Unzoomed** states by pressing either the **Ctrl-Z** key (or the **View -> Restore Previous Zoom** menu item) to enter the Zoomed state, or press the **Ctrl-U** key (or the **View -> Unzoom** menu item) to enter the **Unzoomed** state.

For the ADsweep, the zoomed and unzoomed states will be kept even when new ADsweeps are obtained Online by acquisition, or in Reanalysis by loading in a new ADsweep file. The ContinuousAcquisition graphs cannot be zoomed.

a



b



c

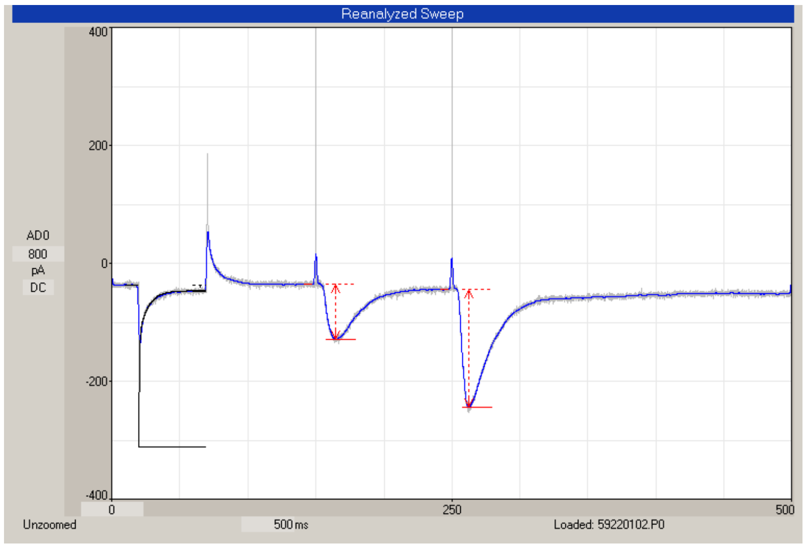


Fig. 3.5.1. ADsweep graph Zooming and Unzooming. a) Non-Zoomed state. There is no “Unzoomed” label on the lower left axis. The mouse cursor has been dragged from the upper-left to the lower right around the first EPSC (but not released) and this ADsweep graph is **about to be** zoomed. b) Zoomed state. There is a red “Zoomed” label on the lower left axis. The mouse cursor has been dragged from the lower-right to the upper-left right around nothing (but not realeased), and this ADsweep graph is **about to be** unzoomed. c) Unzoomed state. There is a black “Unzoomed” label on the lower left axis.

3.6 Coding of Synaptic Waveform Detection

Different colors are chosen in the program depending on whether a synaptic potential was stimulated by S0 or S1 stimulation.

Red denotes extracellular electrode **S0** stimulation in the Train and Pulse Sweep Stimulation panels (Fig. 3.1.3.1). Red marks the superimposed DC Baseline, Peak Amplitude, Slope etc superimposed calculation lines for the first S0 stimulation in the Pulse ADsweep graphs in the Pulse Detection panel (Fig. 3.1.2.3) and MainPg ADsweep graphs (Fig. 3.1.1.1, middle panel). Red also denotes the S0 DC Baseline, Peak Amplitude, Slope etc. calculations in the points plotted in the Amplitude/Calculation graphs (Fig. 3.1.1.1, top panel), and in the Spreadsheet (Fig. 3.1.4.1).

Magenta denotes extracellular electrode **S1** stimulation in the Stimulation Panels, Detection Panels, ADsweep graphs, Analysis graphs and Spreadsheet.

Black denotes **intracellular** electrode stimulation for measurements of membrane or cell input resistance (Rm) and patch electrode series resistance (Rs) in the Stimulation Panels, Detection Panel, ADsweep graphs, Analysis graphs and Spreadsheet.

CHAPTER 4 – Running a Basic LTP Experiment

In the Getting Started section (Chapter 2) an initial configuration of the program has been performed. This includes:

1) Installing WinLTP
2) Installing the data acquisition board.
3) Starting WinLTP
4) Making the appropriate connections to the data acquisition hardware from the recording amplifier and stimulus isolation units (SIUs).
5) Setting up data acquisition and stimulation parameters
6) Acquiring a sweep of data

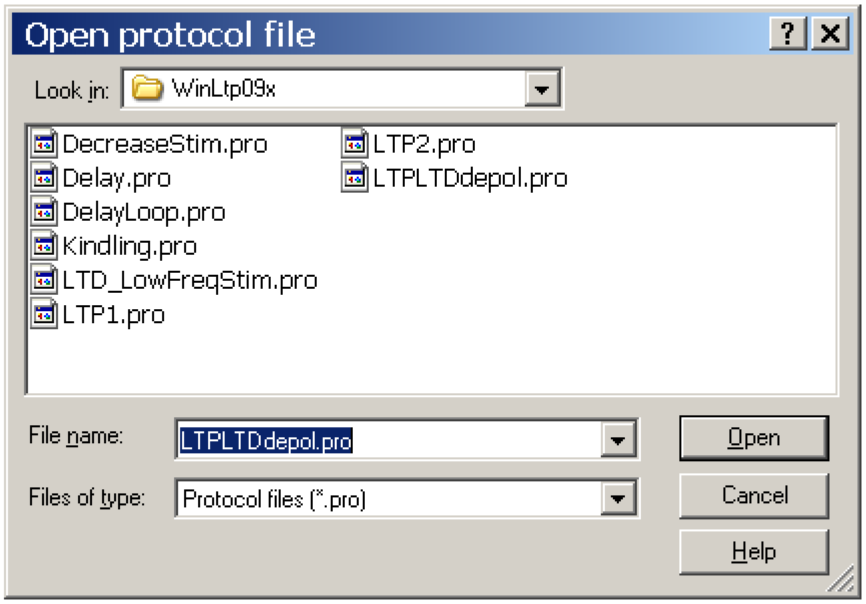
4.1 Automatic and Manual Loading of the Protocol (*.pro) File from Disk

If you have already developed custom protocols to run your particular experiment, the last protocol file used will be the one automatically loaded when the program is later restarted. However, if there was never any protocol file saved, then only the integral default values will be initially operating (as indicated by the default.pro name in the Status Bar.

Alternatively, if you wish to load a different protocol file, use the menu command

File -> Open

to open the Protocol File Dialog Box (Fig. 4.1.1).



File name: LTPLTDdepol.pro

Files of type: Protocol files (*.pro)

Fig. 4.1.1. The Open Protocol File dialog box.

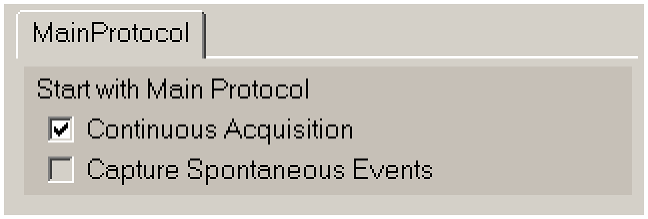
If a custom Protocol File to run your particular experiment has not yet been developed, the following procedures should be performed to fully implement the protocol. Essentially this involves:

1) Choose whether or not to use Continuous Acquisition
2) Writing the script in the Protocol Builder, often to just continuous looping or continuous looping with Signal Averaging
3) Setting up the data acquisition values.
4) Choosing the Stimulation Protocols.
5) Setting Train and Pulse stimulation values.
6) Choosing what synaptic response calculations to do (e.g. DC Baseline, Peak Amplitude, Peak Latency, Area, Duration, Rise Time, Decay Time, Coastline, PopSpike Amplitude, PopSpike Latency, Slope, Average Amplitude), and what Rm and Rs calculations to do.
7) Setting the synaptic potential and Rs detection criteria.
8) Saving this new protocol file.
9) Running the experiment.
10) Saving Spreadsheet / Analysis graph data at the end of the experiment.

Note that after the initial 2 month DemoTrial Period, an Advanced Mode License Key must be purchased from WinLTP Ltd. to the Protocol Builder in the fully functional Advanced Mode (see Section 2.11).

4.3 Choosing Continuous Acquisition

Next, you have to choose whether to run Continuous Acquisition and Capturing Spontaneous Events (not implemented in this version) along with the Main Protocol as simultaneous tasks. To do this go to the Main Protocol 'sub' tabsheet in the Protocol tabsheet and to the 'Start with Main Protocol' panel and either check or uncheck the Continuous Acquisition and/or Capturing Spontaneous Events check boxes (Fig. 4.3.1, see also Fig. 3.1.2.1). The Capturing Spontaneous Events is not currently available.


MainProtocol

Start with Main Protocol

☑ Continuous Acquisition

☐ Capture Spontaneous Events

Fig. 4.3.1. Start with Main Protocol panel in the MainProtocol tabsheet.

4.4 Writing a Simple Script in the Protocol Builder – Scripting 101

As discussed in Section 2.11, when you enter the WinLTP program for the first time in the DemoTrial period you are running in the Advanced Mode with a fully functioning Protocol builder. In this mode you can write any number of advanced protocols using all the Protocol Builder events (marked in green) including the 'Run', 'ElseRun', 'AvgLoop', and 'Loop' events, the 'Slow0', 'Slow1', 'Fast0' and 'Fast1' Perfuse events, the P0, P1, T0 and T1sweep events, and the Delay event.

Alternatively, if you have not ordered an Advanced Mode license, at the end of the 2 month DemoTrial period you will automatically enter the Basic Mode partially functional Protocol Builder. Only the green

Insert Event buttons can be used and still save ADsweep data. These include ‘AvgLoop’, ‘Loop’, ‘P0sweep’ and P1sweep events. If you use the yellow Insert Event buttons, the ‘Run’, ‘ElseRun’, ‘Tosweep’ ‘T1sweep’ and ‘Delay’ events, **your protocol will run perfectly OK, except that the ADsweep data will not be saved. This allows you to easily test the Advanced Mode functions to see if it is worthwhile upgrading to the Advanced Version.**

4.4.1 Slow Repetitive Pulse Sweeps with and without Signal Averaging

Even with the Basic Mode, partially functional Protocol Builder, you can run basic LTP experiments. In basic LTP experiments, a P0sweep normally generates S0 pulse stimulation only, and P1sweep normally generates S1 pulse stimulation only. Therefore, slow repetitive P0sweeps produces slow repetitive S0 pulse stimulation, and alternating P0 and P1sweeps produces alternating S0 and S1 pulse stimulation.

If you:

1) click on the **‘Init’ Protocol button**

you immediately write a continuous loop containing one P0sweep which produces continuous, repetitive P0sweeps (with no signal averaging) (Fig. 4.4.1.1A, top). The loop is continuous because it loops for 99999 times which is essentially longer than the experiment.

If you want to produce repetitively alternating P0/P1sweeps, you:

1) press down the LeftMouseButton to click on the **‘P1sweep’ Insert button**,
2) hold the LeftMouseButton down to **drag the P1sweep down** to just below the P0sweep in the MainProtocol script, and then
3) **release the LeftMouseButton** to insert the P1sweep just below the P0sweep (see red line/arrow in Fig. 4.4.1.1A, bottom).

If you want to signal average, you:

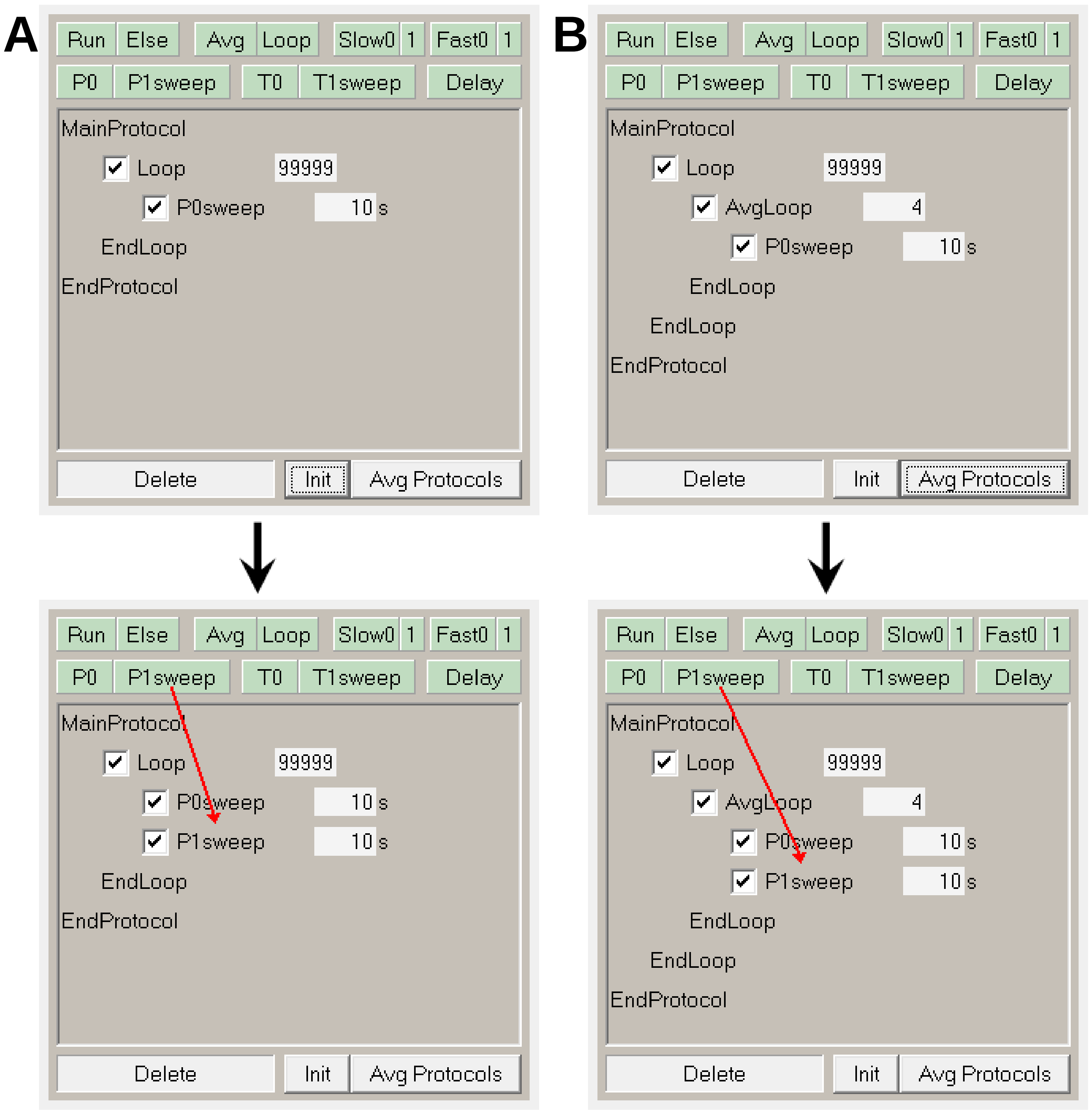
1) click on the **Init ‘Avg Protocols’ button**

to write a continuous outer loop containing an AvgLoop of 4, which in turn contains one P0sweep (Fig. 4.4.1.1B, top). This is a standard protocol for signal averaging every 4 P0sweeps, while continuously repeating this averaging.

However, rather than clicking the **Init ‘Avg Protocols’ button** (Fig. 4.4.1.1B, top) to write the basic signal averaging protocol, you could:

1) click on the **‘Init’ Protocol button** to get the basic continuous loop protocol (Fig. 4.4.1.1A, top),
2) click on the P0sweep in the loop with the LeftMouseButton, drag it down to the delete box, and release the LeftMouseButton **to remove the P0sweep from the continuous loop**,
3) click on the ‘Avg’Loop Insert button, drag it down between the Loop and EndLoop events, and release the LeftMouse button **to insert an AvgLoop in the continuous Loop**, and finally
4) click on the ‘P0’sweep Insert button, drag it down between the AvgLoop and EndLoop events and release it **to insert a P0sweep in the AvgLoop.**

After you do this you have the protocol in Fig. 4.4.1.1B (top) that was alternatively achieved by clicking the **Init Avg Protocols’** button. This shows that even running just a basic LTP experiment protocol is actually just using a simple Protocol Builder script. Hopefully, it has also demonstrated that writing Protocol Builder scripts is incredibly simple, or as the British say, “Bloody Obvious”.



A (top):

```
MainProtocol
    ☑ Loop          99999
        ☑ P0sweep      10 s
    EndLoop
EndProtocol
```

A (bottom):

```
MainProtocol
    ☑ Loop          99999
        ☑ P0sweep      10 s
        ☑ P1sweep      10 s
    EndLoop
EndProtocol
```

B (top):

```
MainProtocol
    ☑ Loop          99999
        ☑ AvgLoop          4
            ☑ P0sweep      10 s
        EndLoop
    EndLoop
EndProtocol
```

B (bottom):

```
MainProtocol
    ☑ Loop          99999
        ☑ AvgLoop          4
            ☑ P0sweep      10 s
            ☑ P1sweep      10 s
        EndLoop
    EndLoop
EndProtocol
```

Fig. 4.4.1.1. Writing simple Protocol Builder scripts to do A) slow Repetitive P0 Pulse Sweeps once every 10 sec (top) and repetitive alternating P0 then P1sweeps every 20 sec. B) Slow Repetitive P0sweeps with signal averaging, and repetitive alternating P0 then P1 sweeps, one averaged P0sweep also every 80 sec. Obviously you would change the period times to something more suitable – 10 sec is the initial sweep period value.

Finally, if you want to produce repetitively alternating P0/P1sweeps with signal averaging, again you

1) press down the LeftMouseButton to click on the **'P1sweep' Insert button**,
2) hold the LeftMouseButton down to **drag the P1sweep down** to just below the P0sweep in the MainProtocol script, and then
3) **release the LeftMouseButton** to insert the P1sweep just below the P0sweep (see red line/arrow in Fig. 4.4.1.1B, bottom).

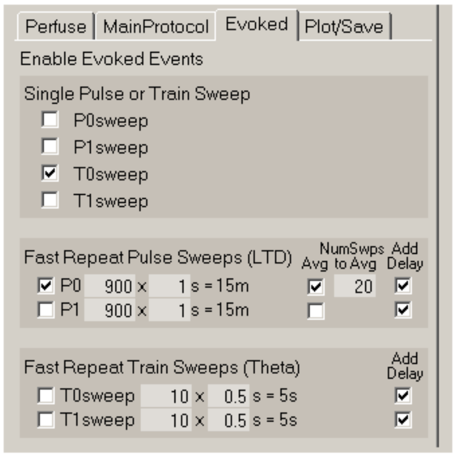
Furthermore, by clicking on the **Evoked 'Single' and 'Repetitive Sweeps' buttons** while the MainProtocol is running, you can evoke Single or Repetitive P0, P1, T0 and T1 sweeps, which allows you to run basic LTP type experiments.

For more information on how to write more advanced scripts in the Protocol Builder, see Chapter 6, Chapter 7 and Chapter 9.

For now just click on the 'Init' Protocol button (Fig. 2.11.1, right top panel), to be able to run repetitive P0sweeps.

4.4.2 Evoked Sweep Stimulation

The EvokedEvents tabsheet (Fig. 4.4.2.1, see also Fig. 3.1.2.2) controls the enabling of evoked single and repetitive sweep stimulation as well as setting the values of evoked repetitive sweep stimulation.

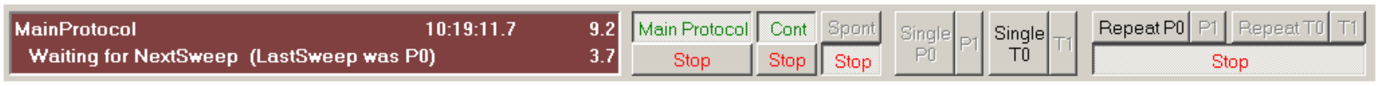


Enable Evoked Events

Single Pulse or Train Sweep
- ☐ P0sweep
- ☐ P1sweep
- ☑ T0sweep
- ☐ T1sweep

Fast Repeat Pulse Sweeps (LTD)			Avg	NumSwps to Avg	Add Delay
☑ P0	900 x	1 s = 15m	☑	20	☑
☐ P1	900 x	1 s = 15m	☐		☑

Fast Repeat Train Sweeps (Theta)			Add Delay
☐ T0sweep	10 x	0.5 s = 5s	☑
☐ T1sweep	10 x	0.5 s = 5s	☑



```
MainProtocol                                  10:19:11.7     9.2
 Waiting for NextSweep  (LastSweep was P0)                   3.7
```

Fig. 4.4.2.1. EvokedEvents tabsheet showing which Single Sweeps and which Repeat Sweeps are enabled by checking the checkbox on the left. In this example, the Single T0sweep and Repeat P0sweeps are enabled and only the 'Single T0' and 'Repeat P0' Run Buttons at the bottom are enabled and ready to be clicked on.

4.4.2.1 Enabling Evoked Single Sweep Stimulation

The check boxes in the ‘Single Train or Pulse Sweep’ panel in Fig. 4.4.2.1 (top of top panel) determine whether these single sweeps can be evoked by clicking single Run Buttons, Function Keys or Run Menu items (Sections 3.1.5 and 3.3, and Figs. 3.1.5.1 and 3.2.6). Fig. 4.4.2.1 shows a Single T0sweep is enabled and only the ‘Single T0’ button of all the Single Sweep Run Buttons in the bottom panel is enabled and ready to be clicked on.

4.4.2.2 Fast Repetitive Pulse Sweep (LTD) Stimulation

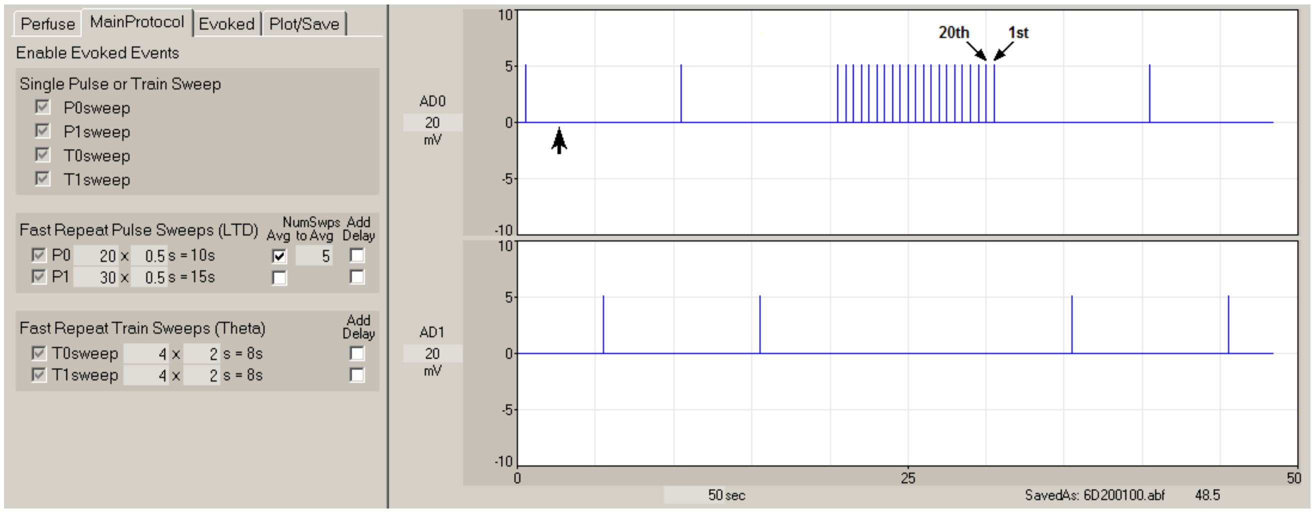
The check boxes in the ‘Fast Repeat Pulse Sweeps (LTD)’ panel in Fig. 4.4.2.1 (middle of the top panel) determine whether these repeat pulse sweeps (P0 and P1) can be evoked by clicking single Run Buttons, Function Keys or Run Menu items (Sections 3.1.5 and 3.3, and Figs. 3.1.5.1 and 3.2.6). Fig. 4.4.2.1 shows Repeat P0 Sweeps is enabled and only the ‘Repeat P0’ button of all the Repeat Sweeps Run Buttons in the bottom panel is enabled and ready to be clicked on.

The first field to the right of the check box determines the number of times a sweep is repeated. The next field to the right is the period of the sweep (in seconds). The text following the equal sign is the total time of the sweep in min:sec.

Next, the ‘Avg’ check box determines whether the Pulse Sweeps are to be signal averaged. If the ‘Avg’ check box is checked, then another field on the right, the ‘NumSweeps to Avg’ field comes into view.

Finally, on the right, if the ‘Add Delay’ check box is checked, an additional delay period (equal to the **preceding** normal slow repeat sweep period) between the end of the LTD stimulation and the resumption of the normal slow repeat sweep stimulation (Fig.4.4.2.2.1). Otherwise there will be no added delay, and it is possible that the first normal sweep will occur immediately after the LTD stimulation without any obvious gap.

A



Enable Evoked Events

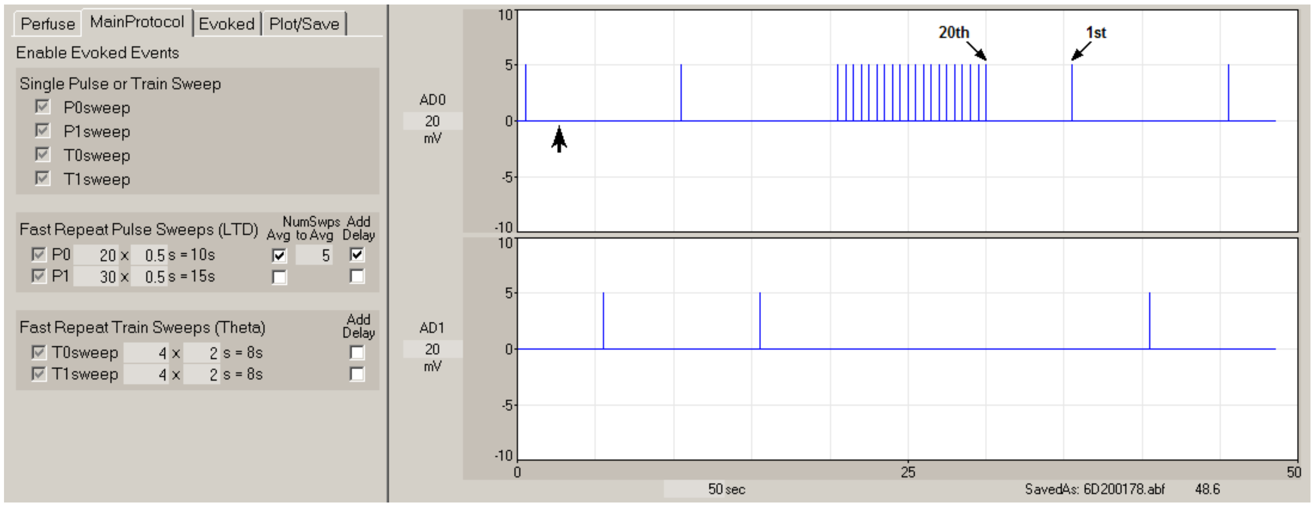
Single Pulse or Train Sweep
- ✓ P0sweep
- ✓ P1sweep
- ✓ T0sweep
- ✓ T1sweep

Fast Repeat Pulse Sweeps (LTD)			Avg	NumSwps to Avg	Add Delay
✓ P0	20 x	0.5 s = 10s	✓	5	☐
✓ P1	30 x	0.5 s = 15s	☐		☐

Fast Repeat Train Sweeps (Theta)			Add Delay
✓ T0sweep	4 x	2 s = 8s	☐
✓ T1sweep	4 x	2 s = 8s	☐

SavedAs: 6D200100.abf 48.5

B



Enable Evoked Events

Single Pulse or Train Sweep
- ✓ P0sweep
- ✓ P1sweep
- ✓ T0sweep
- ✓ T1sweep

Fast Repeat Pulse Sweeps (LTD)			Avg	NumSwps to Avg	Add Delay
✓ P0	20 x	0.5 s = 10s	✓	5	✓
✓ P1	30 x	0.5 s = 15s	☐		☐

Fast Repeat Train Sweeps (Theta)			Add Delay
✓ T0sweep	4 x	2 s = 8s	☐
✓ T1sweep	4 x	2 s = 8s	☐

SavedAs: 6D200178.abf 48.6

Fig. 4.4.2.2.1. Add one extra Delay period after an LTD stimulation. A) LTD stimulation with ***no*** extra Delay period. At the arrow, the 'Repeat P0' sweep button was clicked and a 20 pulse LTD stimulation (1 pulse/sweep) started after the last AvgLoop was exited. The 1st post-LTD pulse looked like 21st pulse of LTD stimulation. B) LTD stimulation with ***one*** extra Delay period. At the arrow, the 'Repeat P0' sweep button was clicked and a 20 pulse LTD stimulation started after the last AvgLoop was exited. The 1st post-LTD pulse was now clearly separated from the 20th pulse of the LTD stimulation.

If a single pulse is in 'Pulse' sweeps, repetitive pulse sweeps will produce LTD (Long-Term Depression) stimulation.

In normal LTP experiments, a Pulse P0 Sweep generates S0 pulse stimulation only, and Fast Slow Repetitive P0 Sweeps produces fast repetitive S0 pulse LTD stimulation. Similarly, a Pulse P1 Sweep generates S1 pulse stimulation only, Fast Slow Repetitive P1 Sweeps produces fast repetitive S1 pulse LTD stimulation.

4.4.2.3 Fast Repetitive Train Sweep (Theta) Stimulation

The check boxes in the 'Fast Repeat Train Sweeps (Theta)' panel in Fig. 4.4.2.1 (bottom of top panel) determine whether these repeat train sweeps (T0 and T1) can be evoked by clicking single Run Buttons, Function Keys or Run Menu items ((Sections 3.1.5 and 3.3, and Figs. 3.1.5.1 and 3.2.6). These enabling check boxes work the same way as those for the Repeat P0 Sweeps.

The first field to the right of the check box determines the number of times a sweep is repeated. The next field to the right is the period of the sweep (in seconds). Finally, the text following the equal sign is the total time of the sweep in min:sec.

Again, on the right, if the 'Add Delay' check box is checked, an additional delay period (equal to the **preceding** normal slow repeat sweep period) between the end of the theta stimulation and the resumption of the normal slow repeat sweep stimulation. Otherwise there will be no added delay, and it is possible that the first normal sweep will occur immediately after the theta stimulation without any obvious gap.

If single trains are in 'Train' sweeps, repetitive sweeps of trains will then produce a theta stimulation.

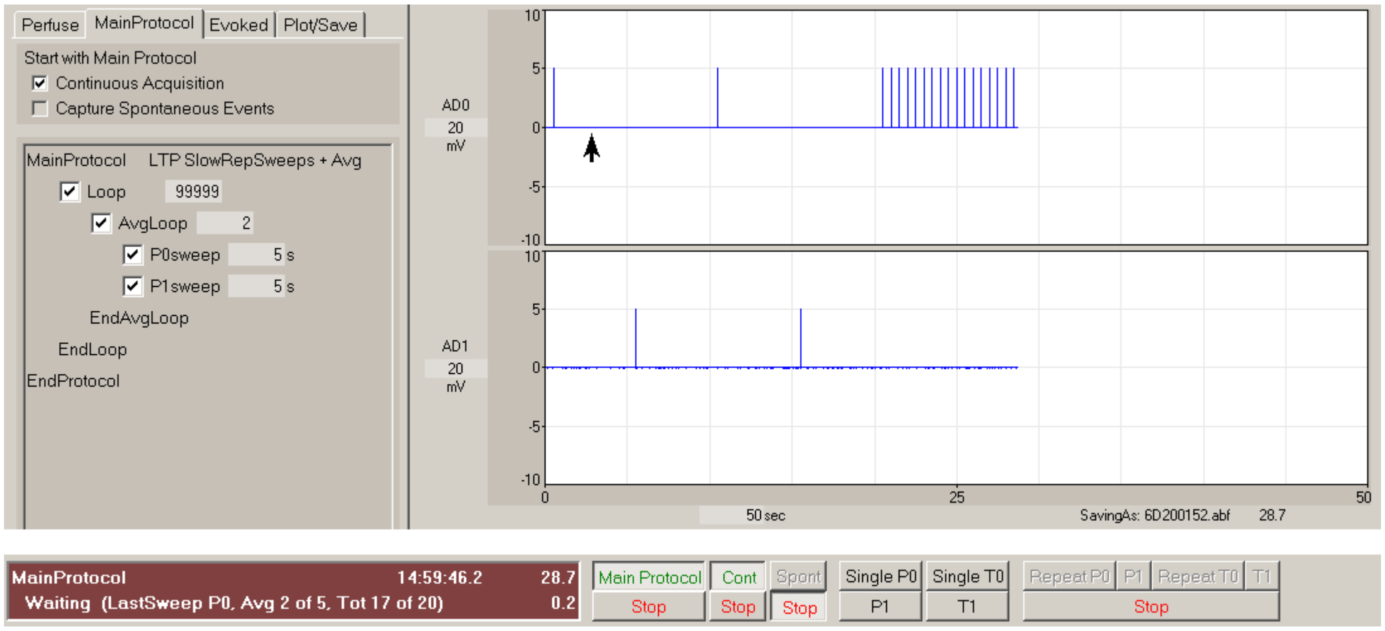
In normal LTP experiments, a Train T0 Sweep generates many (say 100) S0 pulse stimulations, and a eliciting a single Train T0 Sweep would deliver an S0 train stimulation to the preparation. Similarly, a Train T1 Sweep generates many S1 pulse stimulations, and a eliciting a single Train T1 Sweep would deliver an S1 train stimulation to the preparation.

As described in Section 4.7.1, the stimulations in Pulse P0 and P1 Sweeps and Train T0 and T1 Sweeps can be quite varied. For instance, a Train or Pulse Sweep can generate repetitive trains (theta burst) and primed burst stimulation. In fact, a Pulse Sweep can generate train stimulation, and a Train Sweep can generate single pulse stimulation.

4.4.2.4 Evoked Single and Repeat PulseSweeps cannot occur in an Average Loop

In the previous versions of WinLTP from WinLTP090 to WinLTP094, evoked Single and Repeat PulseSweeps could occur in an Average Loop. This would cause a serious error in the calculation of averaged PulseSweep values such as Peak Amplitude because in an Averaging Loop the P0sweeps and the P1sweeps are being averaged by adding to an array **in each pulse sweep** that sums all the sweeps and then is divided by the number of loops. If Single or Repeat P0 or P1 sweeps are evoked, this adds to the sum array but does not change the number of loops – hence the average would have been **wrong!**

This bug was **fixed** in WinLTP095 changing WinLTP so that when Single and Repeat PulseSweeps are evoked during an Average Loop, they will not be run during the Average Loop, but only after it is finished. This is shown in Fig. 4.4.2.4.1, where LTD stimulation was evoked by cliking the 'Repeat P0' sweep Run Button during an Average Loop, but the actual LTD stimulation was delayed until after 2nd Average Loop was exited.



Start with Main Protocol
[✓] Continuous Acquisition
[ ] Capture Spontaneous Events

```
MainProtocol    LTP SlowRepSweeps + Avg
  [✓] Loop        99999
    [✓] AvgLoop       2
      [✓] P0sweep     5 s
      [✓] P1sweep     5 s
    EndAvgLoop
  EndLoop
EndProtocol
```

SavingAs: 6D200152.abf 28.7

```
MainProtocol                              14:59:46.2    28.7
 Waiting  (LastSweep P0, Avg 2 of 5, Tot 17 of 20)        0.2
```

Fig. 4.4.2.4.1. Evoked Repeat PulseSweeps can not occur in an Average Loop. At arrow, the ‘Repeat P0’ sweep button was clicked evoking the LTD stimulation which only started after 2nd Average Loop exited. Also note that For LTD stimulation with averaging, the current and total number of sweeps evoked is now written on the RunLine.

However, evoking Single or Repeat T0 or T1 sweeps in an Averaging Loop is fine because these do not have a sum array and averaging is never done on them. And if you are using the Protocol Builder, only one P0sweep and only one P1sweep is allowed in an AverageLoop – e.g. this possible bug has been prohibited in the Protocol Builder.

4.4.3 Signal Averaging, Stimulus Artifact Blanking and Low-Pass Digital Filtering

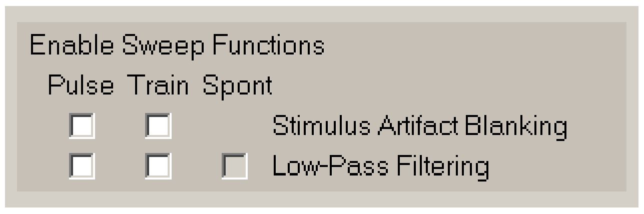
In addition to capturing and analyzing raw sweeps, the WinLTP can also do on- and off-line signal averaging of these sweeps, blank out the stimulus artifacts if required, and low-pass filter the sweeps.

Section 4.4.1 and Fig. 4.4.1.1B have already shown you how to do signal averaging in WinLTP.

The ‘Enable Sweep Functions’ panel in the MainProtocol tabsheet (Fig. 4.4.3.1A, also see Fig. 3.1.2.1) is used to set the stimulus artifact blanking and digital filtering options:

1) **Stimulus Artifact Blanking** enables the stimulus artifacts to be removed on either the Pulse and/or Train Sweeps, and occurs after averaging. Stimulus Artifact Blanking cannot be performed on Spontaneous Sweeps.
2) **Low-Pass Filtering** enables digital filtering on either the Pulse, Train and/or Spontaneous Sweeps, and occurs after averaging and stimulus artifact blanking.

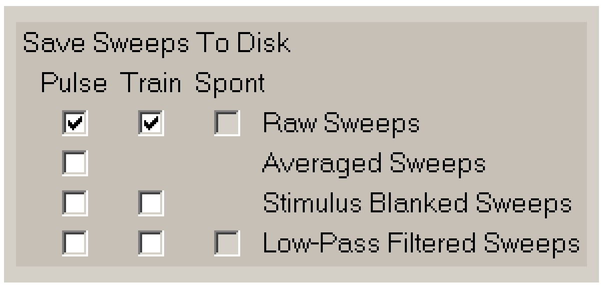
A



Enable Sweep Functions

Pulse	Train	Spont	
☐	☐		Stimulus Artifact Blanking
☐	☐	☐	Low-Pass Filtering

B



Save Sweeps To Disk

Pulse	Train	Spont	
✓	✓	☐	Raw Sweeps
☐			Averaged Sweeps
☐	☐		Stimulus Blanked Sweeps
☐	☐	☐	Low-Pass Filtered Sweeps

Fig. 4.4.3.1. A) Enable Sweep Functions panel in the MainProtocol tabsheet. B) Save Sweeps To Disk panel in the Plot/Save tabsheet.

Single raw sweeps can either be (i) low-pass filtered, (ii) stimulus artifact blanked, or (iii) stimulus artifact blanked and then filtered (Fig. 4.4.3.2a) (but not first filtered and then stimulus artifact blanked). The insets in Fig. 4.4.3.2a show a patch-clamp recording of an EPSC from one raw sweep (left trace) showing substantial noise and a large stimulus artifact at the left of the trace, the sweep that has been digitally filtered to reduce the noise (note the large filtered artifact, right top trace), the sweep with the stimulus artifact removed (middle trace), and the stimulus artifact blanked sweep that has then been filtered (right bottom trace).

Alternatively, raw sweeps can first be (i) signal averaged, then this signal averaged sweep can either be (ii) low-pass filtered, (iii) stimulus artifact blanked, or (iv) stimulus artifact blanked and then filtered (Fig. 4.4.3.2b) (but not first filtered and then stimulus artifact blanked). The insets in Fig. 4.4.3.2b show EPSCs from two raw sweeps showing substantial noise and a large stimulus artifact (left traces), the signal averaged sweep also with a large stimulus artifact (2nd trace) obtained from the two raw sweeps, the signal averaged sweep that has been digitally filtered (right top trace), the signal averaged sweep with the stimulus artifact removed (3rd trace), and the averaged, stimulus artifact blanked sweep that that has been filtered (right bottom trace).

All these traces can be shown on the screen and saved to a data file. Calculations of slopes and peaks are made on the latest processed sweep. For example, if signal averaging, stimulus artifact blanking and digital filtering are being used, then the averaged, blanked and filtered sweep is the one that will be analyzed.

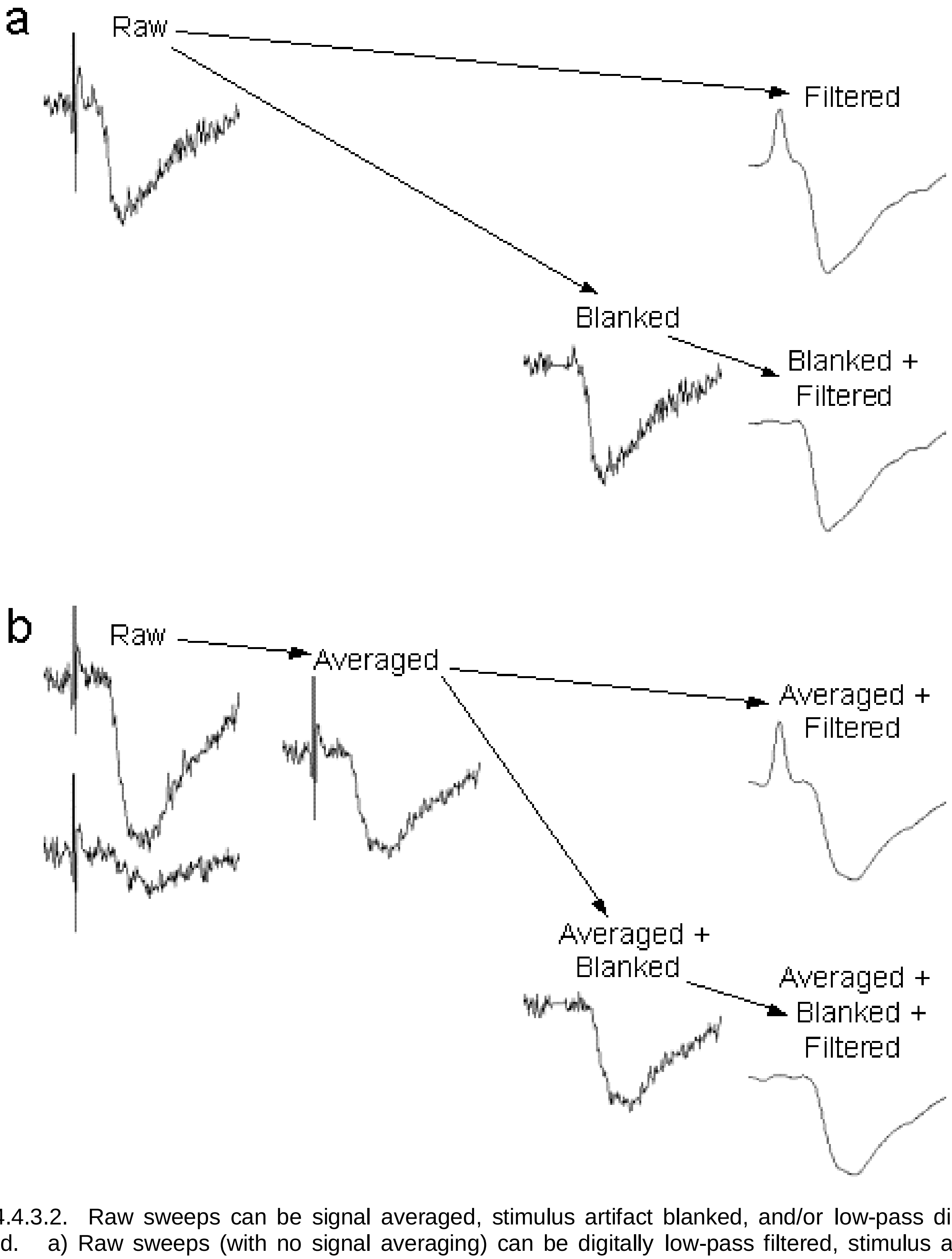


Fig. 4.4.3.2. Raw sweeps can be signal averaged, stimulus artifact blanked, and/or low-pass digitally filtered. a) Raw sweeps (with no signal averaging) can be digitally low-pass filtered, stimulus artifact blanked, or stimulus artifact blanked first and then filtered. The insets show the raw sweep (left trace), the filtered sweep (right top trace), the stimulus artifact blanked sweep (middle trace), and the blanked and filtered sweep (right bottom trace). b) Raw sweeps can also be signal averaged, then low-pass filtered, stimulus artifact blanked, or stimulus artifact blanked first and then filtered. The insets show two raw sweeps (left traces), the signal averaged sweep obtained from the two raw sweeps (2nd trace), the averaged and filtered sweep (right top trace), the averaged and stimulus artifact blanked sweep (3rd trace), and the averaged, blanked and filtered sweep (right bottom trace).

4.4.3.1 Signal Averaging

If **Signal Averaging** is used, each sweep is first acquired and plotted in Gray, then the average of the sweeps is then plotted in LightBlue. The setting for the number of sweeps to average for the **Slow Repeat Sweeps** is set by the **AvgLoop number of loops field** - see Fig. 3.1.2.1 for a simple averaging protocol, and Figs. 6.4.1.1, 6.4.9.1 and 6.4.10.1 for more complex averaging protocols.

You can also signal average during Fast Repetitive Pulse Sweep (LTD) Stimulation which is set by the **NumSweeps to Avg** field (Fig. 4.4.2.1).

4.4.3.2 Stimulus Artifact Blanking

Fig 4.4.3.2.1 and Fig 4.11.2.1 show the effect of stimulus artifact blanking. If **Stimulus Artifact Blanking** is chosen, stimulus artifact blanking is enabled, and a S0 or S1 **Blank field** appears on the Detection page (see Fig. 4.4.3.2.1). Beginning at the time of the start of the S0 or S1 pulse to the time set in the S0 or S1 Blank field (1 ms in Fig. 4.4.3.2.1), the stimulus artifact data in the ADsweep array is (sometimes) set to the **Average** of the 1 data point value just before blanking begins, and one point value just after blanking ceases (Fig. 4.4.3.2.1a). If the time in the S0 or S1 Blank field is set too short, then the stimulus artifact is not entirely blanked, if it is set too long (but not ridiculously long) generally there aren't any problems as long as the peak amplitude is not clipped.

WinLTP can take the **Average** of the point just before blanking begins and the point just after blanking ceases (Fig. 4.4.3.2.1a). However, WinLTP can also take the **Slope** between the point just before blanking begins and the point just after blanking ceases (Fig. 4.4.3.2.1b), and can also take the point just before blanking begins and then **Hold** that voltage until to the point just after blanking ceases (Fig. 4.4.3.2.1c). Setting the **Average**, **Slope** or **Hold** blanking method is accomplished by entering **A**, **S** or **H** into the **Average/Slope/Hold field** next to Blank.

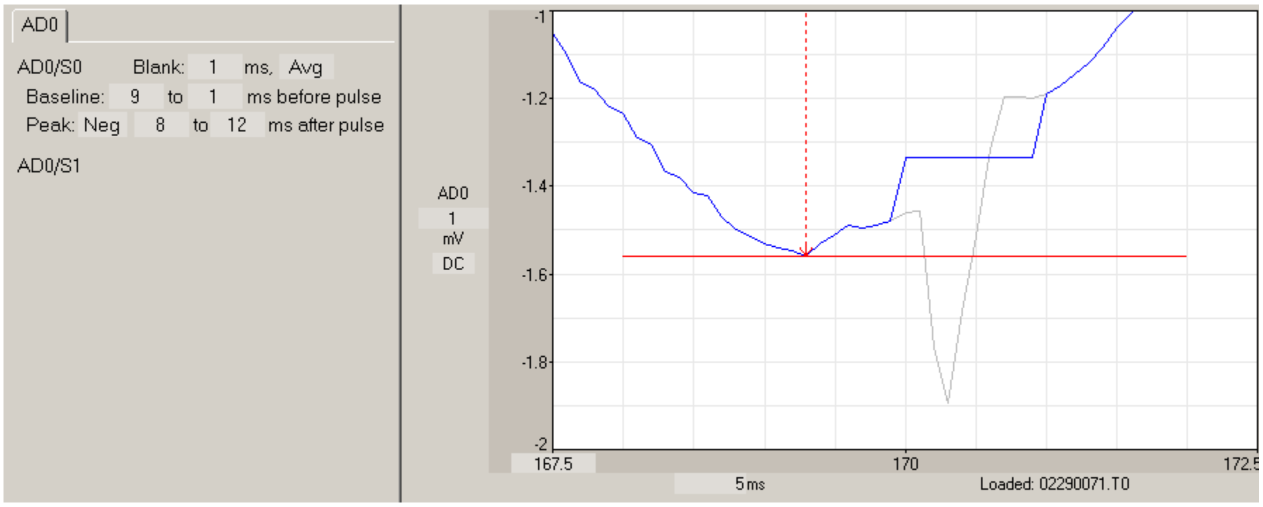
Stimulus artifact blanking is useful for determining the **peak amplitudes of individual EPSPs** when the stimulus artifacts are riding on top of the previous EPSP, or when trying to determine the **area or peak amplitude of a whole train** which could be seriously distorted by the stimulus artifact (Fig 4.11.2.1).

Stimulus artifact blanking has also turned out to be useful when trying to **fit exponential curves to the decay phase** of closely spaced EPSCs when the artifact for the next EPSC occurs during this decay phase. This approach has been used for exponential curve fitting of synaptic trains by the Synaptosoft MiniAnalysis Program (www.synaptosoft.com). It's interesting how removal of stimulus artifacts allows EPSPs and EPSCs to be analyzed as spontaneous events. The **Slope** method has proved particularly useful for fitting exponential curves to synaptic potentials by having a smoother decay phase; in fact it is almost impossible to see where the stimulus artifact was.

The **Hold** method has proved useful for blanking stimulus artifacts where the point just after blanking ceases varies widely (such as blanking the stimulation in extracellular CA3 recording where the ntidromic spike occurs right after the stimulus artifact).

a

Average


AD0

AD0/S0 Blank: 1 ms, Avg

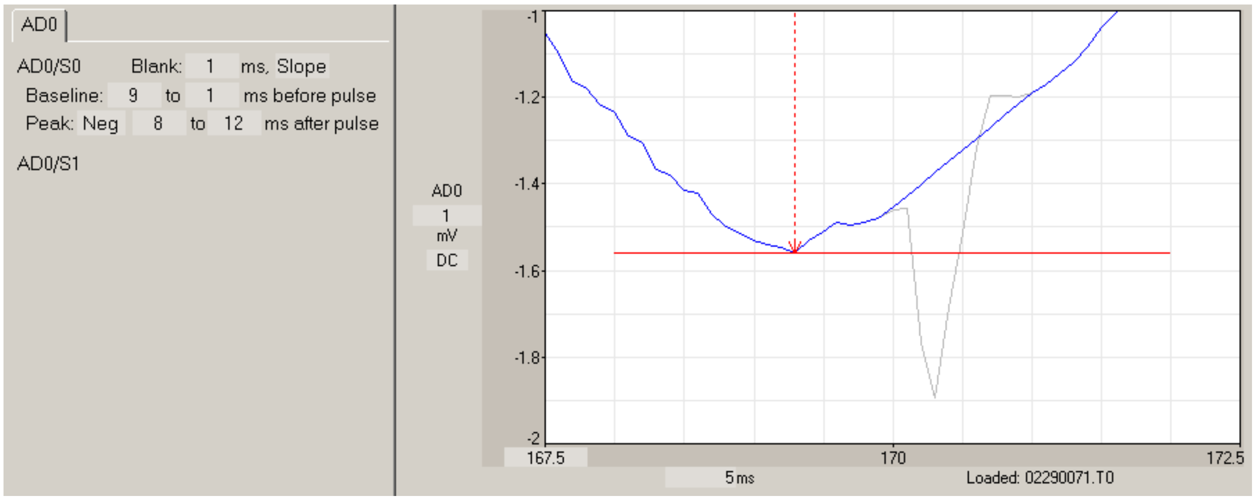
Baseline: 9 to 1 ms before pulse

Peak: Neg 8 to 12 ms after pulse

AD0/S1

b

Slope


AD0

AD0/S0 Blank: 1 ms, Slope

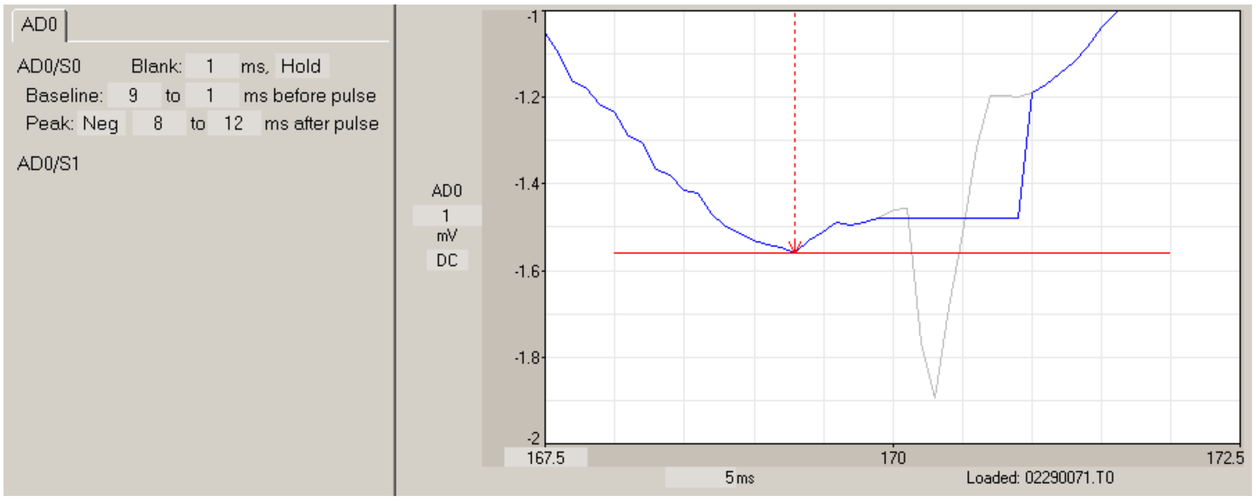
Baseline: 9 to 1 ms before pulse

Peak: Neg 8 to 12 ms after pulse

AD0/S1

c

Hold


AD0

AD0/S0 Blank: 1 ms, Hold

Baseline: 9 to 1 ms before pulse

Peak: Neg 8 to 12 ms after pulse

AD0/S1

Fig. 4.4.3.2.1. The a) Average, b) Slope and c) Hold methods of blocking stimulus artifacts.

4.4.3.3 Low-Pass Digital Filtering

Low-pass digital filtering is done using a Gaussian digital filter (Colquhoun D, and Sigworth, FJ, Fitting and statistical analysis of single channel records. In B. Sakmann and E. Neher editors. Single Channel Recording. Plenum Press, London, 191-263, 1983).

If **Low-Pass Filtering** is chosen, each sweep is first acquired and plotted in Gray, and then digitally filtered at an appropriate frequency and plotted in LightBlue. The setting for digital filtering frequency is set by the **Filter cut-off frequency** field (-3 dB) in the Pulse Detection panel (Figure 4.4.3.3). Note in Fig 4.4.3.3 and in the filtered traces in Fig. 4.4.3.2 how filtering the stimulus artifact can really distort the waveform around the stimulus artifact, and that blanking the stimulus artifact removes the distortion around the artifact.

Digital Filtering can be very useful for reanalyzing **Peak Amplitude** because the algorithm to measure peak amplitude merely picks out the most positive or negative ADsample. Therefore, if there is a lot of noise, the Peak Amplitude measurement will be artificially increased by the extraneous noise. Appropriate levels of digital filtering can reduce this noise and give a more accurate Peak Amplitude measurement.

Digital filtering shouldn't be necessary for analyzing the **Slope** since the slope is a linear regression line (least squares fit) through the data points. Interestingly, digital filtering can reduce the sweep-to-sweep jitter in the slope data points, possibly because the digital filtering effectively extends the time over which the slope is calculated by including sample points before and after the time at which the slope is calculated.

When digital filtering is on, the amplitude/slope calculation analysis is done on the filtered ADsweeps only.

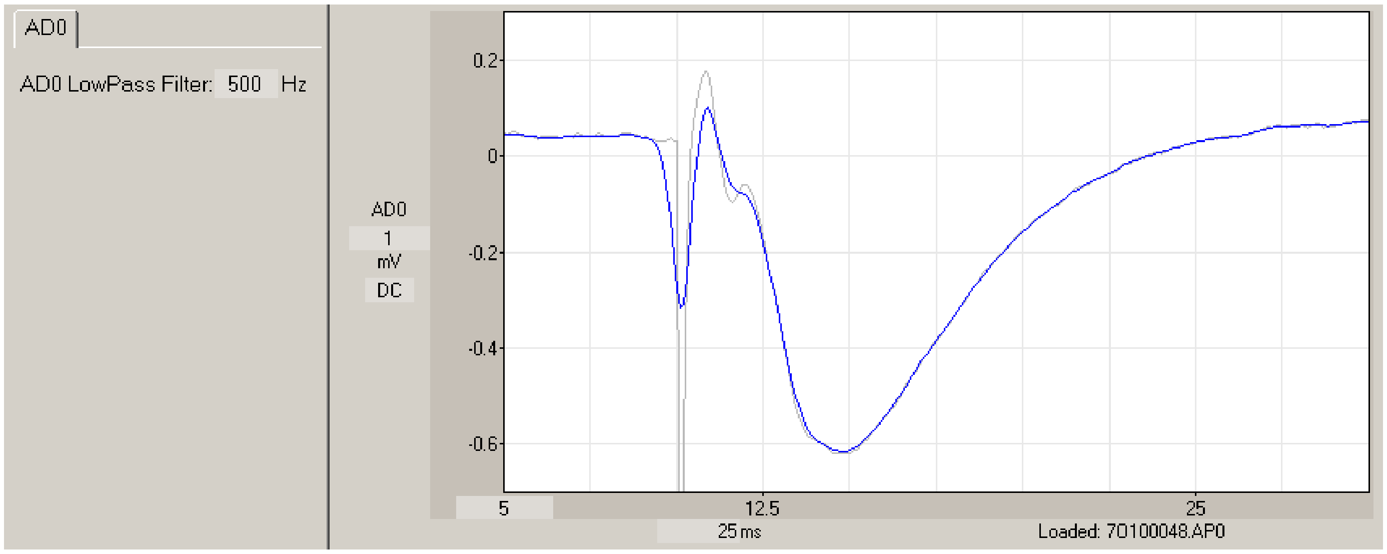
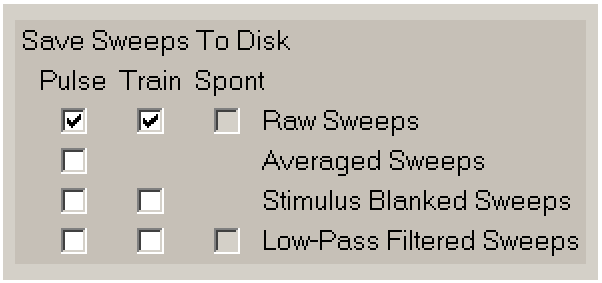


Fig. 4.4.3.3. Digitally filtering an ADsweep waveform at the **Filter cut-off freq** of 500Hz (-3dB cut-off frequency) in the Pulse Waveform Detection Panel. The raw, unfiltered ADsweep is shown in Gray and the filtered data is shown in LightBlue.

4.4.4 Setting Which Sweeps to Save to disk

The 'Save Sweeps To Disk' panel (Fig. 4.4.4.1) in the Plot/Save tabsheet (Fig. 4.4.3.1B, also see Fig. 3.1.2.2, right) sets which sweeps will be saved:

1) **Raw Sweeps** enables saving raw Pulse, Train or Spontaneous Sweeps to disk
2) **Averaged Sweeps** enables saving only averaged Pulse Sweeps to disk (Trains Sweeps, because they cannot be averaged, cannot be saved as averaged sweeps).
3) **Stimulus Artifact Blanked Sweeps** enables saving stimulus artifact blanked Pulse and/or Train Sweeps to disk
4) **Low-Pass Filtered Sweeps** enables saving digitally filtered Pulse and/or Train Sweeps to disk.



Save Sweeps To Disk

Pulse	Train	Spont	
✓	✓	☐	Raw Sweeps
☐			Averaged Sweeps
☐	☐		Stimulus Blanked Sweeps
☐	☐	☐	Low-Pass Filtered Sweeps

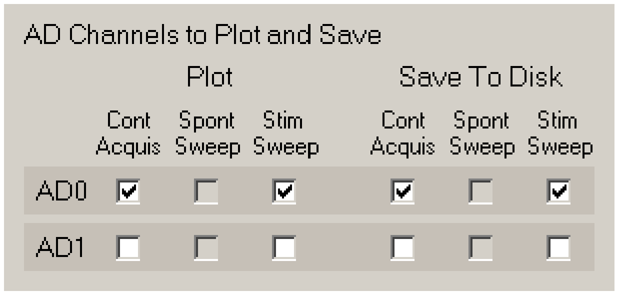
Fig. 4.4.4.1. Set which sweeps to save to disk. Saving Spontaneous sweeps is disabled for now.

The file extensions of the saved sweep files are as follows:

	Pulse	Train
Raw	*.P0, *.P1	*.T0, *.T1
Blanked	*.BP0, *.BP1	*.BT0, *.BT1
Filtered	*.FP0, *.FP1	*.FT0, *.FT1
Blanked & **F**iltered	*.FP0, *.FP1	*.FT0, *.FT1
Averaged	*.AP0,*.AP1	
averaged & b**L**anked	*.LP0, *.LP1	
averaged & f**I**ltered	*.IP0, *.IP1	
averaged, blanked & f**I**ltered	*.IP0, *.IP1	

4.5 Set Which AD Channels to Plot and Save, and Which Sweeps to Save

Win LTP can acquire data on one channel (AD0 or AD1), or on two channels (AD0 + AD1). To set which AD channels will be plotted and saved go to the AD Channels to Plot and Save panel (Fig. 4.5.1 and Fig. 3.1.2.2, right).



AD Channels to Plot and Save

	Plot			Save To Disk		
	Cont Acquis	Spont Sweep	Stim Sweep	Cont Acquis	Spont Sweep	Stim Sweep
AD0	✓		✓	✓		✓
AD1						

Fig. 4.5.1. AD Channels to Plot and Save Panel.

Then check which AD channels you want to plot for Continuous Acquisition and Stimulation Sweeps, and which AD channels you want to save to disk for Continuous Acquisition and Stimulation Sweeps. Note that Plotting and saving to Spontaneous Sweeps is currently disabled.

4.6 Set the Data Acquisition Values

Setting Analog Input values has already been discussed in Section 2.12.1.

4.6.1 Setting Analog Output Values

The lower left panel of the Edit Protocol dialog box and the Acquisition/Stimulation Parameters tabsheet is the Analog Output Channels panel (see Fig. 2.12.1.1).

This currently sets IC0 output (IC1 not implemented yet). The output can either be in straight volts '**V**', in '**mV**' for patch clamp voltage-clamp mode, or '**nA**' or '**pA**' for current-clamp mode. It is set in the **DataTypeUnits** field.

If the output is in **volts**, the **DataTypeUnit** is set to '**V**', the **Gain** field is set to '**1**', and the **Units/V** field becomes '**1 V/V**'.

For patch clamp **voltage-clamping**, the **DataTypeUnit** is set to '**mV**', and the Gain and Units/V fields depend on the output gain of the patch clamp amplifier. For example, with Axon Instruments patch clamp amplifier 200B, the sensitivity is 20 mV cell voltage to 1 volt input voltage, or 20 mV/V, and the **Units/V** field is therefore set to '**20 mV/V**'. The **Gain** field then becomes '**50**'. This is in fact the default setting.

For current clamping, the **DataTypeUnit** is set to either 'nA' or 'pA', and we will set it to '**nA**' in this example. For the Axon 200B, the output to the cell in nA is (2/β) nA per volt input, and where β = 1, this becomes 2 nA/V, and the **Units/V** field is therefore set to '**2 nA/V**'. The Gain field then becomes '**500000000**'.

4.7 Choosing Pulse/Train Sweep Stimulation Protocols and Setting Stimulation Values

The complete stimulation output of WinLTP is a combination of the output of the P0, P1, T0 and T1sweeps in the Protocol Builder (Chapters 6, 7 and 9) plus the Evoked Single and Repeat Sweeps.

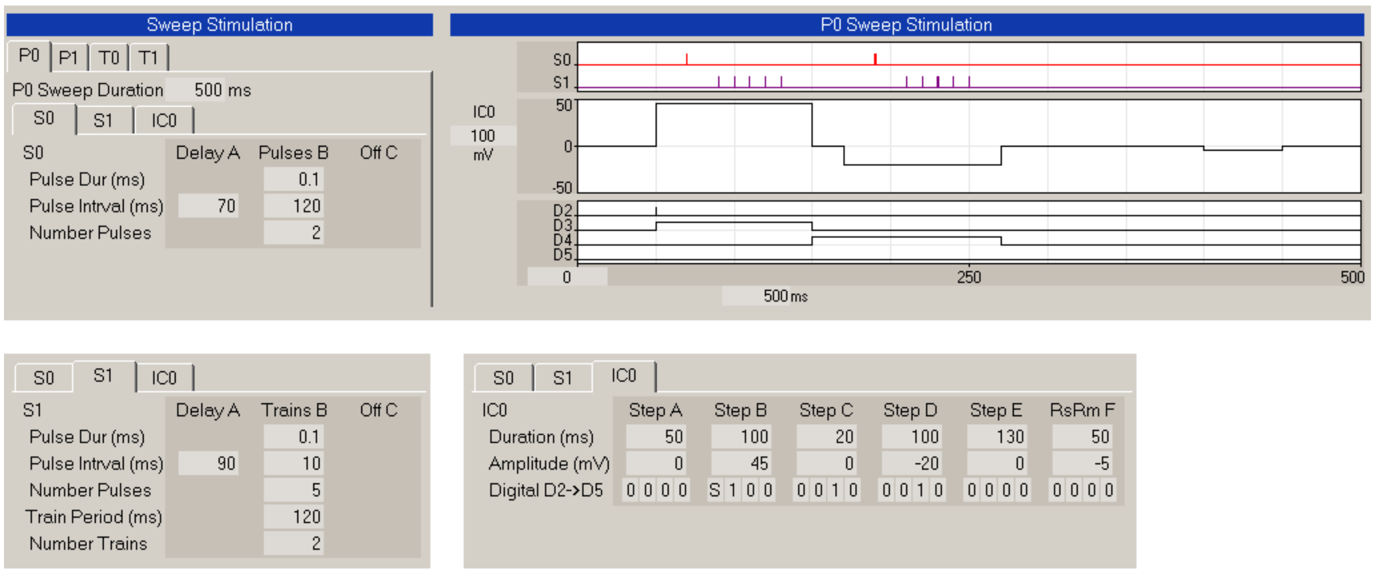
4.7.1 Choosing the Sweep Stimulation Protocol

The extracellular, intracellular and digital stimulation in each sweep is controlled by the fields in the P0, P1, T0 or T1 tabsheet in the Sweep Stimulation area (Fig. 4.7.1.1, also see Section 3.1.3 and Figs. 3.1.3.1, 3.1.3.2 and 3.1.3.3).

The stimulations that can be controlled by the Sweep Stimulation area include:

1) Sweep Duration
2) S0 and S1 extracellular stimulation (organized in terms of pulses and trains)
3) IntraCellular (IC) Analog Output stimulation organized as epochs (e.g. up to 6 sequential pulses, PulseA to PulseE)
4) Four Digital Outputs that can also be produced during the Intracellular (IC) epochs.

To set the sweep stimulation protocol, first choose the desired sweep by clicking on the P0, P1, T0 or T1 tabsheet in the Sweep Stimulation area.



Sweep Stimulation

P0 | P1 | T0 | T1

P0 Sweep Duration 500 ms

S0 | S1 | IC0

S0	Delay A	Pulses B	Off C
Pulse Dur (ms)		0.1	
Pulse Intrval (ms)	70	120	
Number Pulses		2	

S0 | S1 | IC0

S1	Delay A	Trains B	Off C
Pulse Dur (ms)		0.1	
Pulse Intrval (ms)	90	10	
Number Pulses		5	
Train Period (ms)		120	
Number Trains		2	

S0 | S1 | IC0

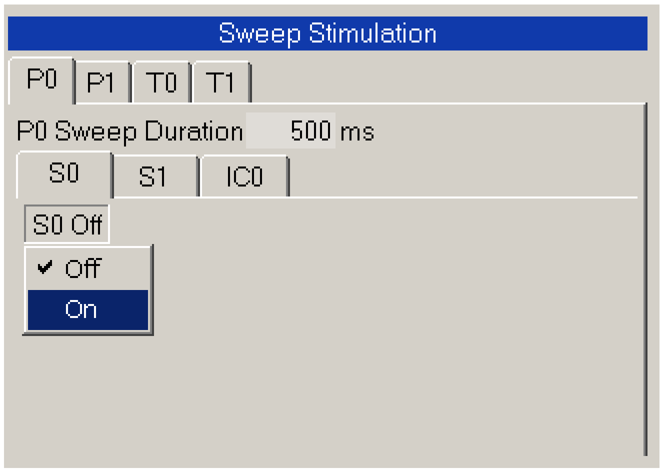
IC0	Step A	Step B	Step C	Step D	Step E	RsRm F
Duration (ms)	50	100	20	100	130	50
Amplitude (mV)	0	45	0	-20	0	-5
Digital D2->D5	0 0 0 0	S 1 0 0	0 0 1 0	0 0 1 0	0 0 0 0	0 0 0 0

Fig. 4.7.1.1. P0sweep stimulation consisting of two S0 pulses (left side of top panel), two S1 trains (left bottom panel), and IntraCellular analog output channel 0 (IC0), Digital **S**ync output (see the '**S**' in DO2), and Digital Out step output (see the '**1**'s in DO3 and DO4) (right bottom panel).

As discussed in Section 3.1.3, the Field Sweep Stimulation area is functionally coupled with the Graph Stimulation area, so that, for example, when you click on the P0 tabsheet in the Field Sweep Stimulation area, the P0 Sweep Stimulation graph comes up (Fig. 3.1.3.3).

Then choose whether you want to change S0 extracellular stimulation, S1 extracellular stimulation or Intracellular analog channel 0 (and possibly digital) stimulation by clicking on the S0, S1 or IC0 tabsheet.

S0 and S1 stimulation can be either ‘Off’ or ‘On’ as set by the pop-up menu that pops up when the mouse cursor is clicked on the S0 or S1 Label/Button in the S0 or S1 tabsheet (Fig. 4.7.1.2).



P0 Sweep Duration 500 ms

Fig. 4.7.1.2. Setting S0 pulses ‘Off’ or ‘On’ by clicking on the S0 Label/Button that raises up when the mouse cursor is moved over it.

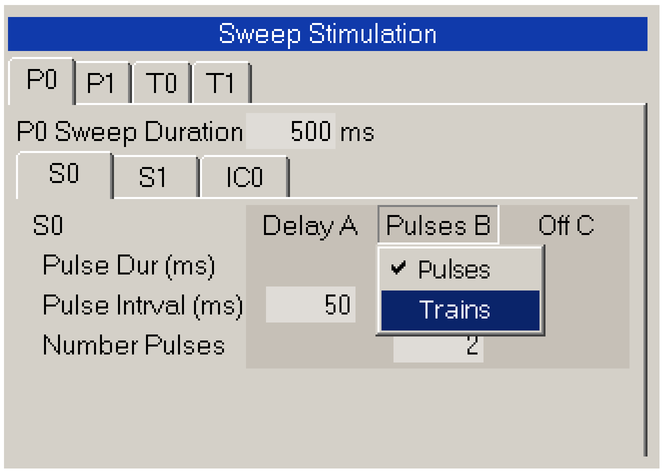
If S0 or S1 extracellular stimulation is ‘On’, the output for P0, P1, T0, or T1 Sweeps can be either:

1) Pulses
2) Trains

which is set by changing EochB to PulsesB or TrainsB (Fig. 4.7.1.3).

Note that in this version, the DOS LTP Program Dual Trains is not supported, so stimulations such as prime burst stimulation cannot be generated by only one extracellular output. However, this will be supplanted by up to 14 more Pulses/Trains (‘PulsesO/TrainsO’) with one level of looping among the pulses/trains, so this will easily include prime burst stimulation.

Note that there does not need to be any S0 or S1 pulse stimulation in a sweep, nor does there need to be any IC analog or digital output. So actually, there needs to be no stimulation in a sweep.

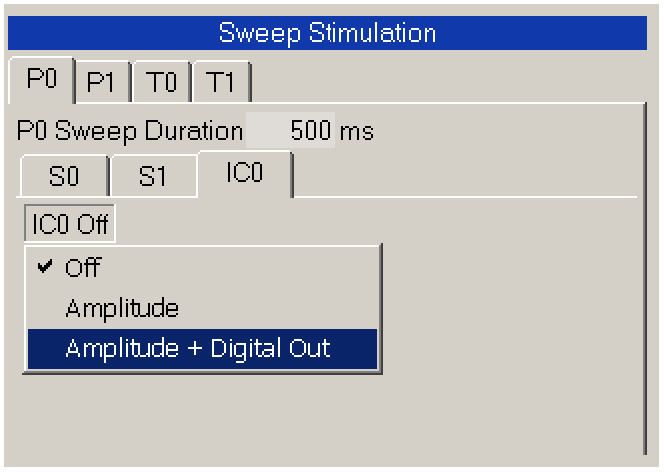


P0 Sweep Duration 500 ms

S0			
Pulse Dur (ms)			
Pulse Intrval (ms)	50		
Number Pulses		2	

Fig. 4.7.1.3. Setting S0 output to ‘Pulses’ or ‘Trains’ by clicking on the Pulses/TrainsB Label/Button that raises up when the mouse cursor is moved over it.

IC0 stimulation can be either ‘Off’, ‘Amplitude’ On, or ‘Amplitude + Digital Out’ On as set by the pop-up menu that pops up when the mouse cursor is clicked on the S0 or S1 Label/Button in the S0 or S1 tabsheet (Fig. 4.7.1.4).



P0 Sweep Duration 500 ms

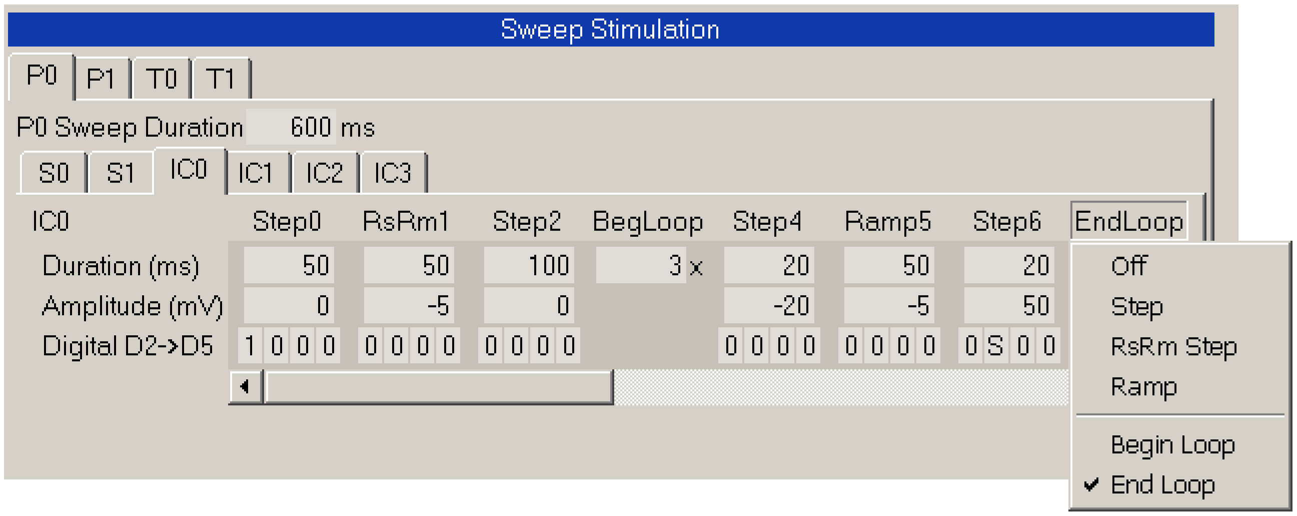
Fig. 4.7.1.4. Setting Intracellular Epoch output to ‘Off’, Analog Out ‘Amplitude’ On, or Analog Out ‘Amplitude + Digital Out’ On, by clicking on the IC0 Label/Button that raises up when the mouse cursor is moved over it.

If IC0 Analog Out ‘Amplitude’ or ‘Amplitude + Digital Out’ stimulation is On, the output for each Epoch0 to Epoch19 in P0, P1, T0, or T1 Sweeps can be either:

1) Off
2) Step
3) RsRm Step
4) Ramp
5) Begin Loop
6) End Loop

which is set by changing Eoch1 through Epoch18 (Fig. 4.7.1.5). The RsRm Step is the step where patch electrode series resistance and cell input resistance are measured relative to the previous step.

Note that Epoch0 only can go to ‘Off’, ‘Step’ or ‘Begin Loop’, not ‘RsRm Step’, ‘Ramp’ or ‘EndLoop’ because there is no previous step with which to use in the Rs/Rm measurements, you need a Step before a Ramp to set the initial amplitude, and a loop cannot begin on an EndLoop.


Sweep Stimulation

P0 | P1 | T0 | T1

P0 Sweep Duration 600 ms

S0 | S1 | IC0 | IC1 | IC2 | IC3

IC0	Step0	RsRm1	Step2	BegLoop	Step4	Ramp5	Step6	EndLoop
Duration (ms)	50	50	100	3 x	20	50	20	
Amplitude (mV)	0	-5	0		-20	-5	50	
Digital D2->D5	1 0 0 0	0 0 0 0	0 0 0 0		0 0 0 0	0 0 0 0	0 S 0 0	

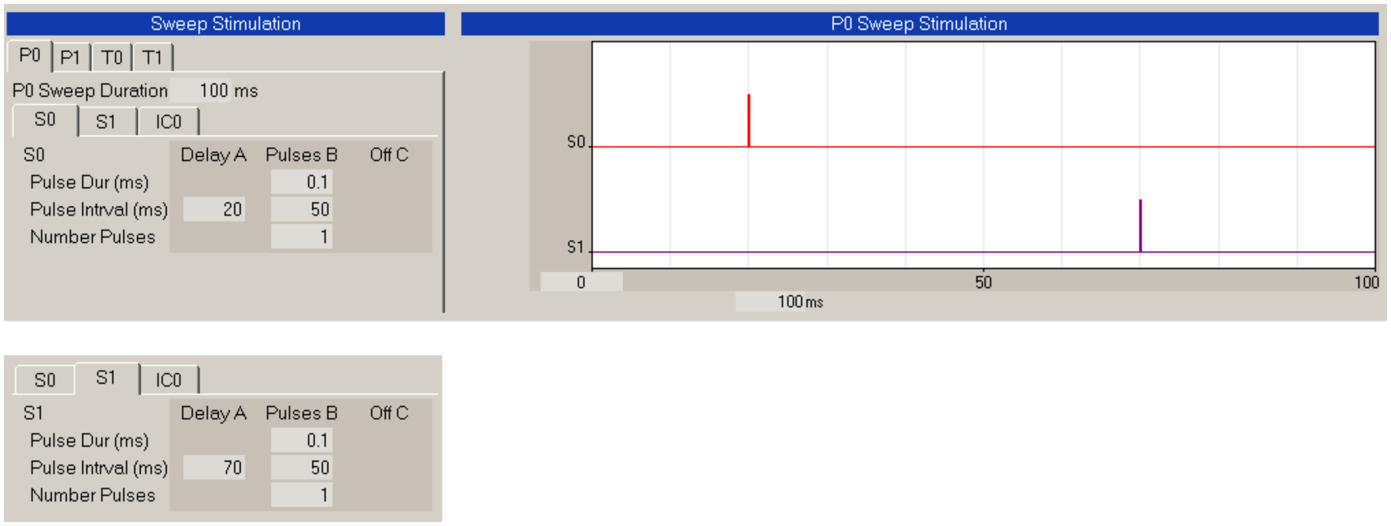
Fig. 4.7.1.5. Setting IC0 Epoch output to ‘Off’, ‘Step’, ‘RsRm Step’, ‘Ramp’, ‘Begin Loop’ or ‘End Loop’ by clicking on the Off/Step/RsRm/Ramp/BegLoop/EndLoop Epoch Label/Button that raises up when the mouse cursor is moved over it.

Analog stimulation also has the capability of producing **ramps**, and generating **analog sequential single trains** using BeginLoop/EndLoop constructs (Fig. 4.7.2.5) and **analog sequential multiple trains** using Loops within Loops constructs (Fig. 4.7.2.6).

4.7.2 Examples of Sweep Stimulation

In addition to the **full stimulation capability** of this current version shown in Fig. 4.7.1.1, the following examples show many simpler but useful sweep stimulation examples.

Fig. 4.7.2.1 shows that **pathway independence** can be tested by comparing heterosynaptic paired pulse stimulation with one pulse on S0 and one pulse on S1 with the effects of homosynaptic paired pulse stimulation (see S0 trace in Fig. 4.7.1.1). Fig. 4.7.2.1 is produced by Pulses on both S0 and S1, and Intracellular and Digital Out Off.


Sweep Stimulation

P0 | P1 | T0 | T1

P0 Sweep Duration 100 ms

S0 | S1 | IC0

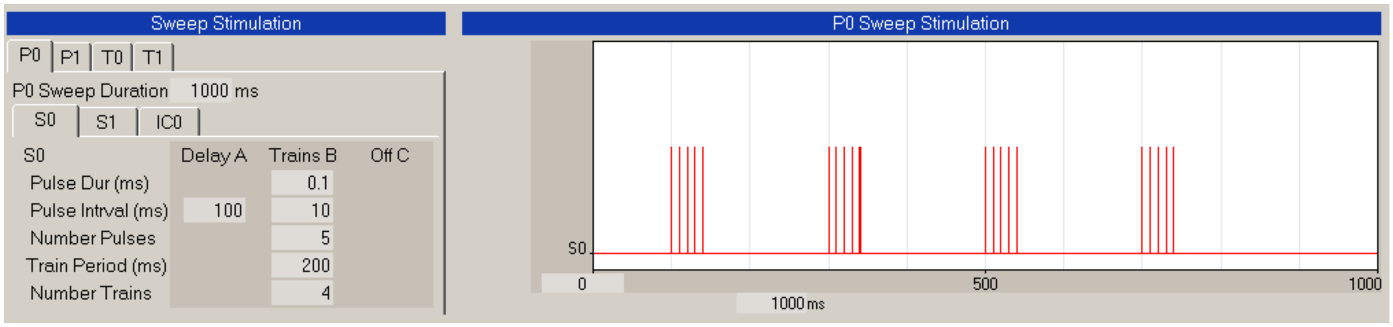
S0	Delay A	Pulses B	Off C
Pulse Dur (ms)		0.1	
Pulse Intrval (ms)	20	50	
Number Pulses		1	

S0 | S1 | IC0

S1	Delay A	Pulses B	Off C
Pulse Dur (ms)		0.1	
Pulse Intrval (ms)	70	50	
Number Pulses		1	

Fig. 4.7.2.1. Heterosynaptic paired-pulse stimulation can help test for pathway independence.

Theta burst stimulation in a single sweep. Fig. 4.7.2.2 shows a theta burst stimulation in a single sweep (rather than single trains in repeating sweeps) capable of inducing LTP. S0 Epoch B is set to Trains, and the NumTrains is set to greater than 1 to get repetitive train stimulation.


Sweep Stimulation

P0 | P1 | T0 | T1

P0 Sweep Duration 1000 ms

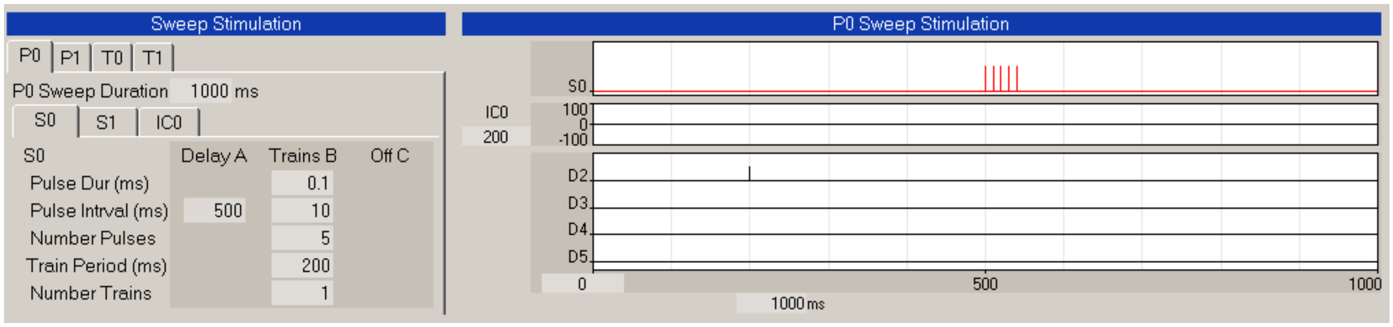
S0 | S1 | IC0

S0	Delay A	Trains B	Off C
Pulse Dur (ms)		0.1	
Pulse Intrval (ms)	100	10	
Number Pulses		5	
Train Period (ms)		200	
Number Trains		4	

Fig. 4.7.2.2. Theta burst stimulation for LTP induction consisting of repeating trains in a single sweep.

Primed burst stimulation. Fig. 4.7.2.3 shows primed burst stimulation capable of inducing LTP. However, since S0 or S1 is capable of producing only pulse, trains, or repeating trains, the version of WinLTP can only produce primed burst stimulation by outputting two extracellular pathways. It could be a single pulse on S1 and a single train on S0, but if S1 is also being used, it has to be a Digital Out pulse (on D2 here) and a single train on S0.

Then the two digital outputs S0 and D2 have to go into the same Stimulus Isolation Unit either by adding an **OR chip** before it, or using two **diodes** to stop current flow in the wrong direction (this is what we do). **Check with your Stimulus Isolation Unit seller on how to do this.**



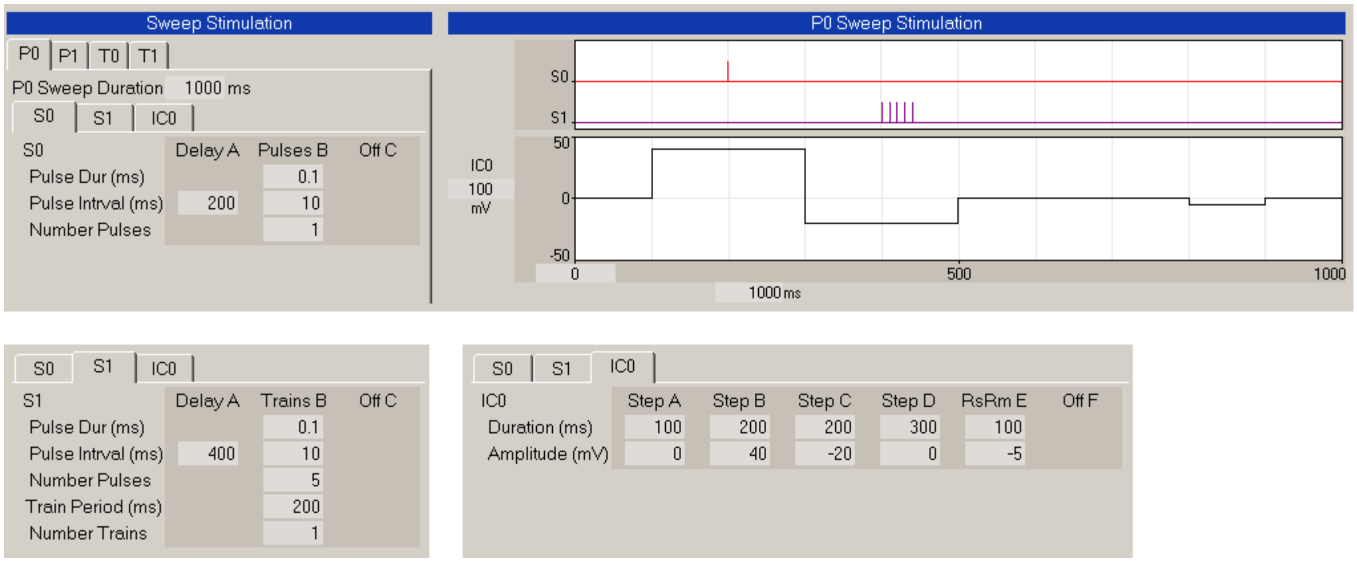
P0 Sweep Duration 1000 ms

S0 | S1 | IC0

S0	Delay A	Trains B	Off C
Pulse Dur (ms)		0.1	
Pulse Intrval (ms)	500	10	
Number Pulses		5	
Train Period (ms)		200	
Number Trains		1	

Fig. 4.7.2.3. Primed burst stimulation for LTP induction consisting of a single pulse on one digital output (D2), and a single train on another digital output (S0) which then go to trigger the same Stimulus Isolation Unit by using an OR chip or two diodes.

Fig. 4.7.2.4 shows how intracellular stimulation can be coincident with extracellular pulse and train stimulation. In this example S0, S1 and Intracellular with RsRm stimulation are On. In this example an intracellular depolarization occurs during the one priming S0 pulse, and hyperpolarization occurs during the S1 train. Intracellular voltage changes could have interesting effects on the ability of extracellular pulses and trains to induce LTP or LTD.



P0 Sweep Duration 1000 ms

S0 | S1 | IC0

S0	Delay A	Pulses B	Off C
Pulse Dur (ms)		0.1	
Pulse Intrval (ms)	200	10	
Number Pulses		1	

S0 | S1 | IC0

S1	Delay A	Trains B	Off C
Pulse Dur (ms)		0.1	
Pulse Intrval (ms)	400	10	
Number Pulses		5	
Train Period (ms)		200	
Number Trains		1	

S0 | S1 | IC0

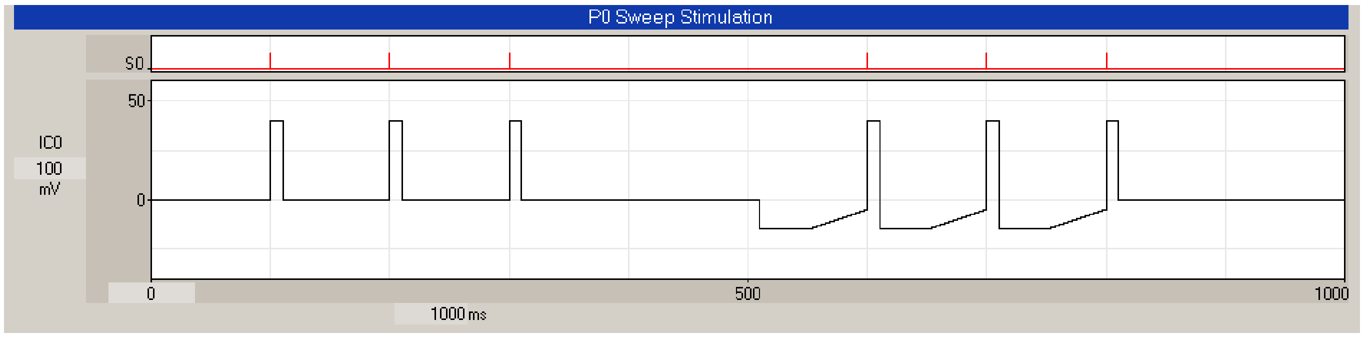
IC0	Step A	Step B	Step C	Step D	RsRm E	Off F
Duration (ms)	100	200	200	300	100	
Amplitude (mV)	0	40	-20	0	-5	

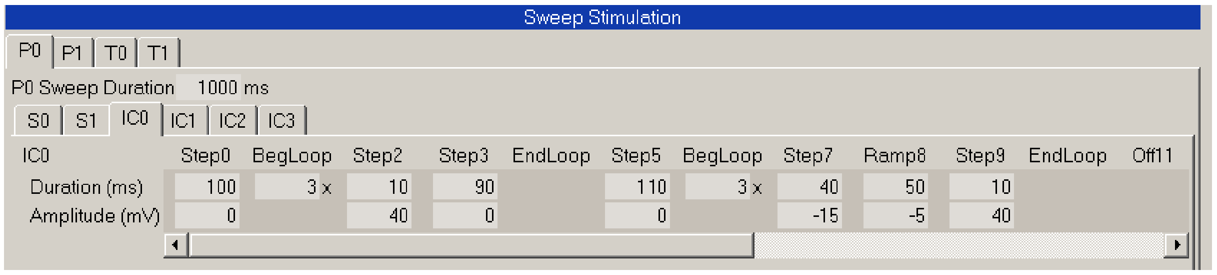
Fig. 4.7.2.4. Extracellular heterosynaptic primed burst stimulation (S0 single pulse, S1 train) and coincident intracellular (IC) depolarization and hyperpolarization stimulation with an RsRm test pulse.

It is important to understand that what you see in the Train and Pulse Stimulation graphs is what stimulus pulses you will generate. If not all of your stimulus pulses ‘fit’ in the stimulation graphs, they will not be generated and no error message will be produced to tell you that. It is up to you to see that this does not occur.

Analog stimulation now has the capability of producing **ramps** that were generated within Train1 Loop (Fig. 4.7.2.5).

Analog stimulation also now has the capability of generating **analog sequential single trains** using BeginLoop/EndLoop constructs (Fig. 4.7.2.5) and **analog sequential multiple trains** using Loops within Loops constructs (Fig. 4.7.2.6).




Sweep Stimulation

P0 | P1 | T0 | T1

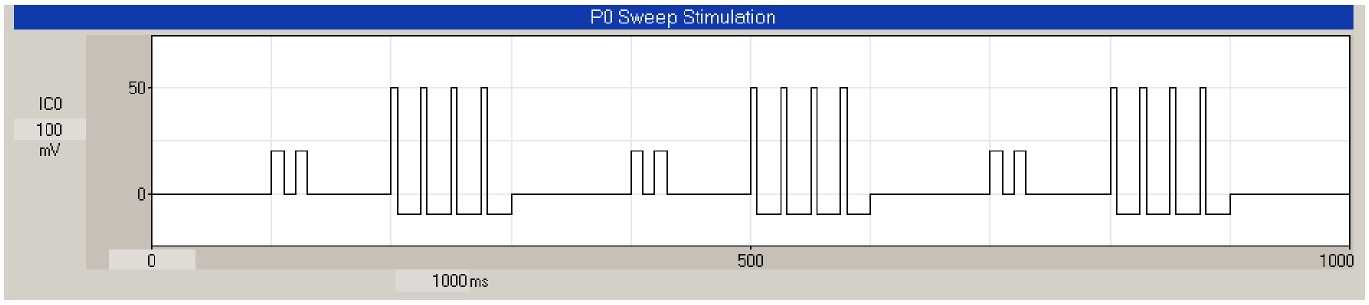
P0 Sweep Duration 1000 ms

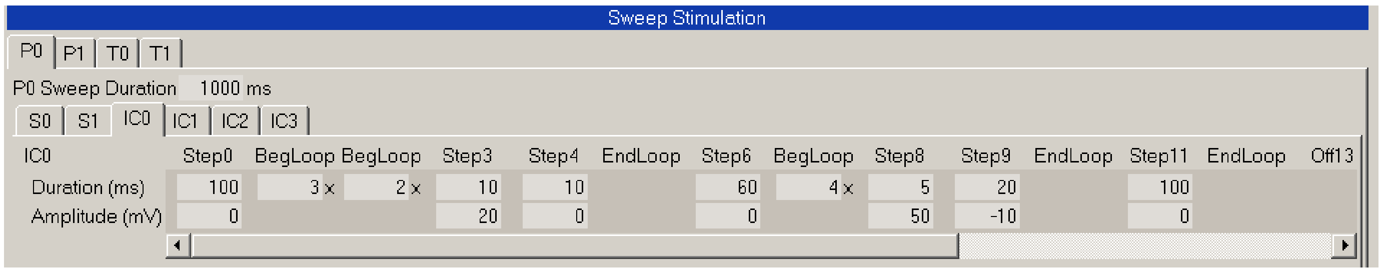
S0 | S1 | IC0 | IC1 | IC2 | IC3

IC0	Step0	BegLoop	Step2	Step3	EndLoop	Step5	BegLoop	Step7	Ramp8	Step9	EndLoop	Off11
Duration (ms)	100	3 x	10	90		110	3 x	40	50	10		
Amplitude (mV)	0		40	0		0		-15	-5	40		

|-----------Train0 Loop-----------| **|----------------Train1 Loop----------------|**

Fig. 4.7.2.5. Sequential Single Trains using BegLoop/EndLoop contructs for Train0 Loop and Train1 Loop. Also note that ramps are generated within the Train1 Loop.




Sweep Stimulation

P0 | P1 | T0 | T1

P0 Sweep Duration 1000 ms

S0 | S1 | IC0 | IC1 | IC2 | IC3

IC0	Step0	BegLoop	BegLoop	Step3	Step4	EndLoop	Step6	BegLoop	Step8	Step9	EndLoop	Step11	EndLoop	Off13
Duration (ms)	100	3 x	2 x	10	10		60	4 x	5	20		100		
Amplitude (mV)	0			20	0		0		50	-10		0		

|------- Train0 Inner Loop -------| **|------- Train1 Inner Loop -------|**

|--- Outer Loop ---|

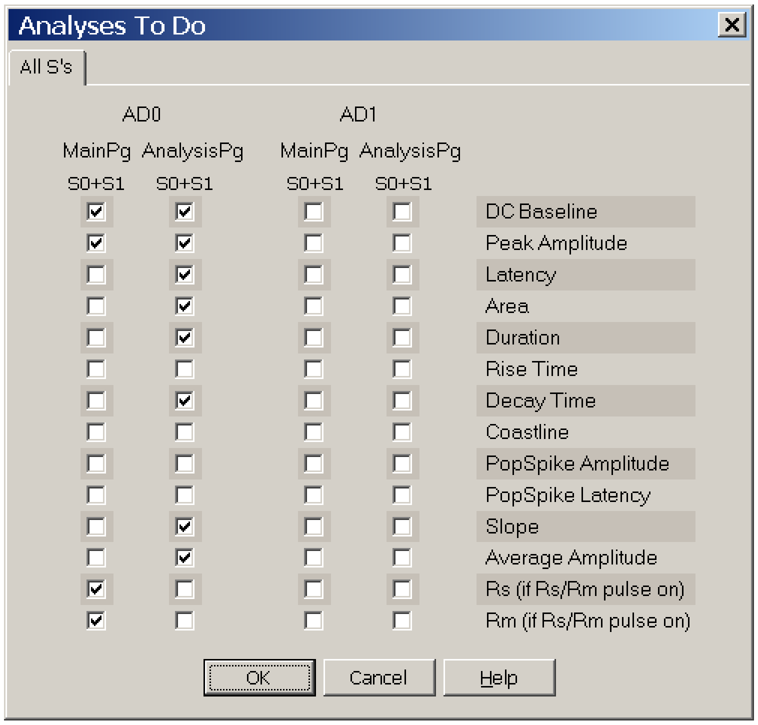
Fig. 4.7.2.6. Sequential Multiple Trains using multiple Train0 and Train1 Inner Loops within an Outer Loop.

4.8 Choosing the Analyses To Do

After setting the sweep stimulations, you have to decide which calculations you want to do. To do this call up the Amplitude/Slope **Analyses To Do** Dialog Box (Fig. 4.8.1A).

On the right the dialog box shows which calculations can be done: DC Baseline before of each Stimulus Pulse, the Peak Amplitude, Latency, Area, Duration, Rise Time, Decay Time, Coastline, PopSpike Amplitude, PopSpike Latency, Slope and/or Average Amplitude of synaptic potentials produced by S0 and S1 stimulation, and if the Rm pulse is On, patch pipette series resistance (Rs) and membrane or cell input resistance (Rm).

A

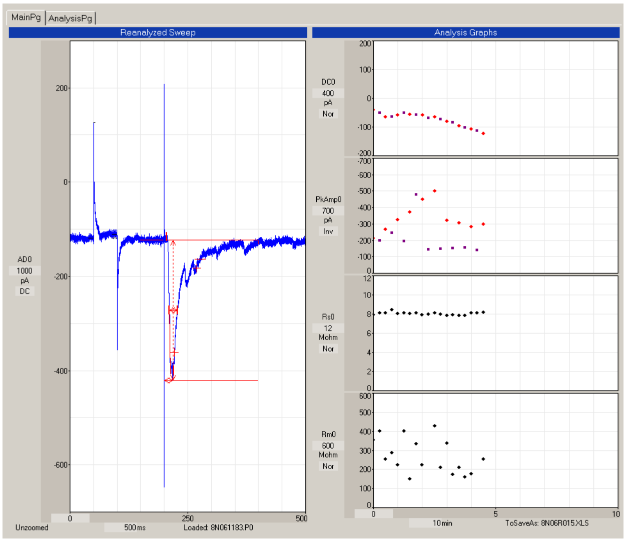


Analyses To Do

All S's

AD0 MainPg S0+S1	AD0 AnalysisPg S0+S1	AD1 MainPg S0+S1	AD1 AnalysisPg S0+S1	
✓	✓	☐	☐	DC Baseline
✓	✓	☐	☐	Peak Amplitude
☐	✓	☐	☐	Latency
☐	✓	☐	☐	Area
☐	✓	☐	☐	Duration
☐	☐	☐	☐	Rise Time
☐	✓	☐	☐	Decay Time
☐	☐	☐	☐	Coastline
☐	☐	☐	☐	PopSpike Amplitude
☐	☐	☐	☐	PopSpike Latency
☐	✓	☐	☐	Slope
☐	✓	☐	☐	Average Amplitude
✓	☐	☐	☐	Rs (if Rs/Rm pulse on)
✓	☐	☐	☐	Rm (if Rs/Rm pulse on)

B



C

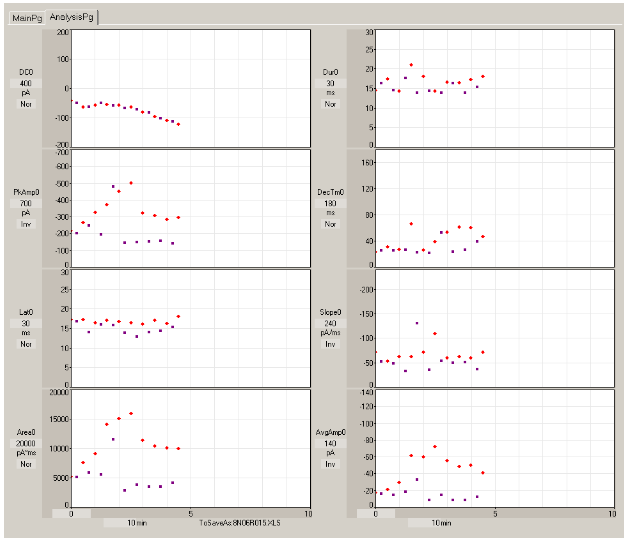


Fig. 4.8.1. A) Amplitude/Slope Analyses To Do dialog box. In this example only analyses on channel AD0 were chosen , and B) four were placed on the MainPg, and C) eight on the AnalysisPg.

The top line in the dialog box shows whether the Analysis To Do will be performed on channel AD0 and /or AD1 (only AD0 in this example). The next line shows where these calculations will be plotted or displayed, either on the **MainPg** or the AnalysisPg. More than **four** MainPg calculations checked will not be accepted. More than **eight** AnalysisPg calculations checked will not be accepted. If an Analysis To Do is chosen, say Slope, then Slope will also appear as one of the columns in the spreadsheet, and will also be saved to the Amplitude analysis (*.Amp) file. WinLTP can analyze all S0- and S1-evoked synaptic responses in all channels in both Pulse and Train sweeps (see Section 4.10).

Adding the four Analysis Graphs on the MainPg with the eight Analysis Graphs on the Analysis Page gives a total of twelve Analysis Graphs that can be viewed during online acquisition and reanalysis.

4.9 Setting the Calculation Detection Criteria

The fields that set the ranges for detecting the various calculations are set in the Pulse Waveform Detection Panel (Fig. 3.1.2.3). However, only if at least one of the MainPg or AnalysisPg check boxes is chosen for a particular waveform calculation row in the Amp/Slope Analyses To Do dialog box (Fig. 4.8.1) will the particular calculation be plotted and saved to the Amplitude/Calculation file as a non-zero value.

4.9.1 DC Baseline

If DC Baseline, Peak Amplitude, Peak Latency, Area, Duration, Rise Time, Decay Time, Average Amplitude, or Slope (for Low% -> High% Peak Amplitude) is chosen, then the DC Baseline value will be calculated.

The

BaselineS0: __ to __ ms before pulse

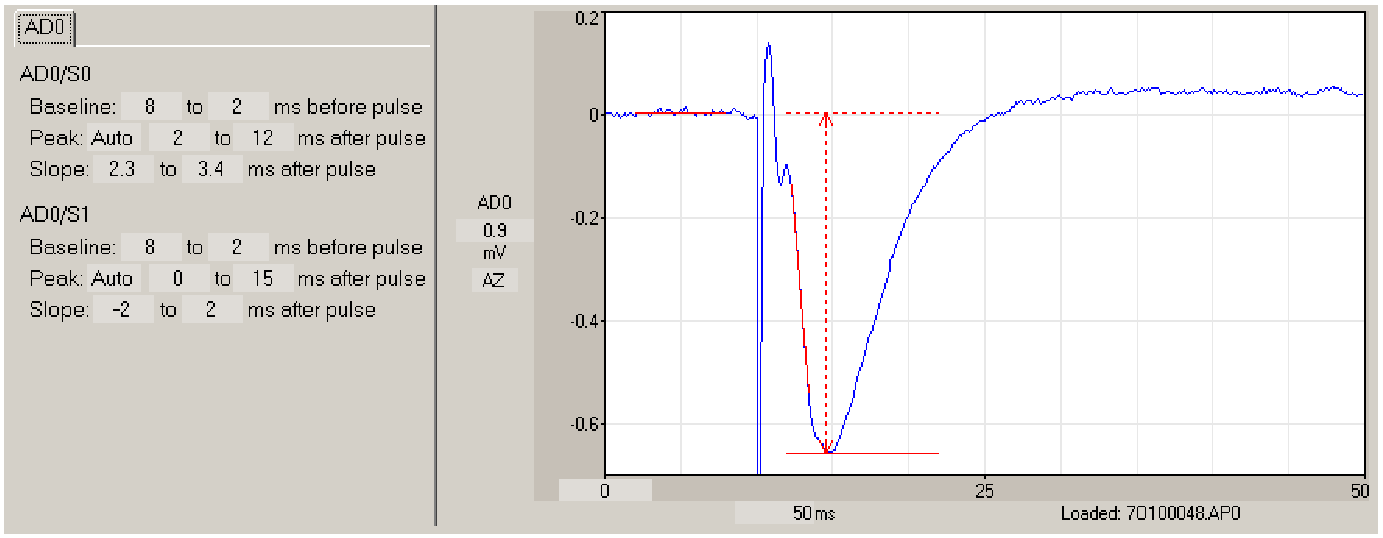
time fields shown in the Pulse Detection Panel in Fig. 4.9.2.1 set the pre-stimulus pulse baseline to be between these two ‘Baseline’ time values, and both are relative to the stimulus pulse.

4.9.2 Peak Amplitude

The Peak Amplitude is the difference between the DC Baseline value and the calculated peak. The peak will be measured between the time fields in

Peak: Auto/Pos/Neg ___ to ___ ms after pulse

and is shown by the PkAmp solid line of the Pulse Detection Panel in Fig. 4.9.2.1. The first ‘Peak’ time value must be before the Peak Amplitude, and the second ‘Peak’ time value must be after the Peak Amplitude, and both are relative to the stimulus pulse.


AD0

AD0/S0
Baseline: 8 to 2 ms before pulse
Peak: Auto 2 to 12 ms after pulse
Slope: 2.3 to 3.4 ms after pulse

AD0/S1
Baseline: 8 to 2 ms before pulse
Peak: Auto 0 to 15 ms after pulse
Slope: -2 to 2 ms after pulse

AD0
0.9
mV
AZ

Loaded: 70100048.AP0

Fig. 4.9.2.1. Detection of extracellular synaptic waveform parameters (DC Baseline, Peak Amplitude and Slope).

The Auto/Pos/Neg field determines whether the peak will be Automatically (Auto) determined to be positive or negative, forced to be Positive (Pos), or forced to be Negative (Neg). The normal value is A, automatic. Automatic calculates whether the average of the points between the Peak: ___ to ___ms after pulse time fields is more positive than the baseline average value. If so, the peak is positive; otherwise the peak is negative.

4.9.3 Latency

Latency is now measured only at 100% of Peak, and is therefore Peak Latency. Peak Latency is the time between the occurrence of stimulus pulse and the peak. In later versions, Latency will be able to be determined as a % of peak.

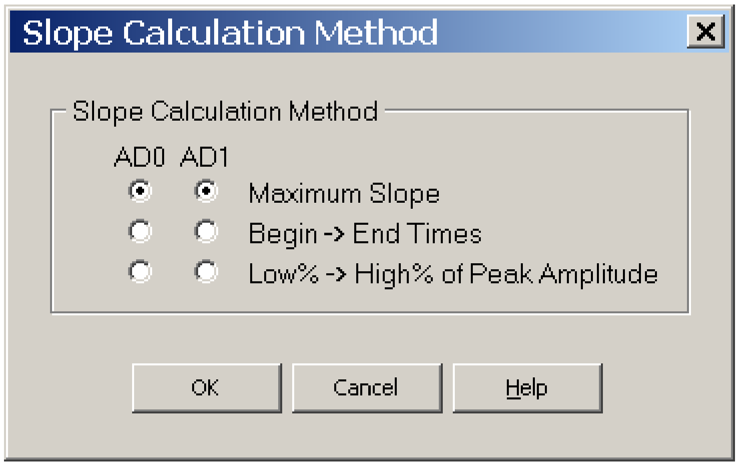
4.9.4 Slope

The Slope is calculated by taking all the waveform voltage/current points from the slope beginning time point to the slope end time point, and using these points to calculate a **linear regression line (least squares fit)** through the data.

When analyzing the Slope of the EPSP/EPSC there are several ways to determining the slope beginning time point and the slope end time point. These Slope Calculation Methods are chosen by using the menu commands:

AmpFile -> Slope calculation method...

to bring up the Slope Calculation Methods dialog box with the following choices (Fig. 4.9.4.1).



Slope Calculation Method

Slope Calculation Method

AD0	AD1	
●	●	Maximum Slope
○	○	Begin -> End Times
○	○	Low% -> High% of Peak Amplitude

Fig. 4.9.4.1. Slope Calculation Method Dialog Box.

Often there is no one best method of measuring slope, and using two methods is better. For example, when there are large latency shifts in the synaptic potential, and the amplitude of the synaptic potential sometimes goes to zero (ie with synaptic failures), a combination of Maximum Slope measurement (with no low pass filtering) to measure the large amplitude synaptic potential slopes, and Begin -> End Times slope measurement to measure the zero and very low amplitude potential slopes will give the best slope measurement.

4.9.4.1 Maximum Slope (with no Low Pass Digital Filtering)

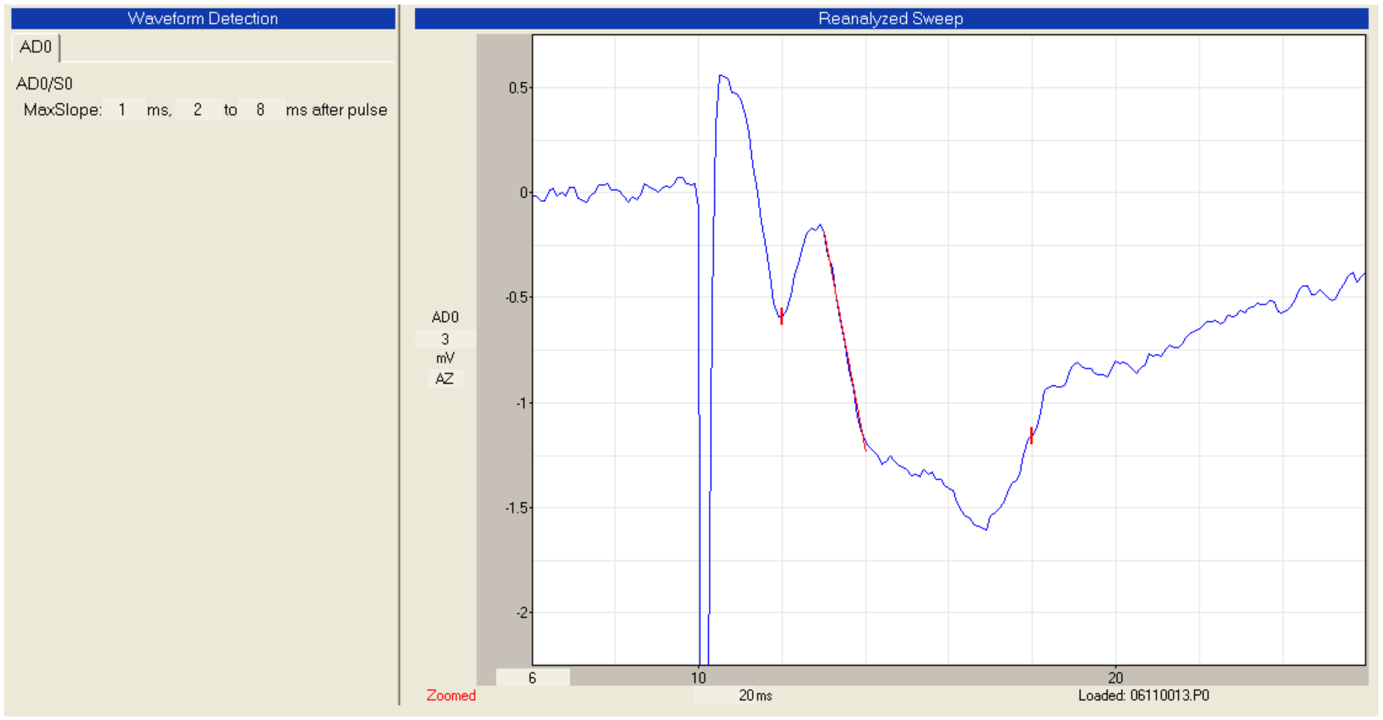
The first method, the Maximum Slope method (with no low-pass filtering), was added as a more reliable way of measuring slope for conditions where the latency shifts in the synaptic potential and the slope of the synaptic potential is small (close to noise level). The problem of this method is when there is no synaptic potential (ie synaptic failure), the slope still measures the maximum slope of the noise and is therefore never zero.

To use the MaximumSlope measurement (Fig. 4.9.4.1.1), set the MaxSlope time field ("1 ms" in this figure, marked by a red slope line) so that it is within the larger BegTime/EndTime range ("2 to 8 ms after pulse" in this example, marked by solid vertical lines on the trace).

MaxSlope: ___ ms, ___ to ___ ms after pulse

In this example, the MaxSlope algorhythm will start at 2 ms after the pulse and calculate the value of a 1 ms slope every 0.1 ms until the EndTime point is reached at a slope between 7 and 8 ms after the pulse. The **absolute largest positive or negative slope** will be the one chosen and plotted using the red slope line).

This is the easiest method to use on-line and is therefore the default method.



AD0/S0
MaxSlope: 1 ms, 2 to 8 ms after pulse

Fig. 4.9.4.1.1. The Maximum Slope Method using no low-pass digital filtering.

4.9.4.2 Maximum Slope (using Low Pass Digital Filtering)

Although the Slope fits a straight line to the “straight” part of the rising phase of the EPSP/EPSC, the rising phase is usually not a straight line, and is more accurately measured by getting the maximum slope of a **shorter** straight line. But there is a tradeoff between getting the correct maximum slope using a short straight line, and measuring a slope of noise. To help get a more correct maximum slope measurement using a short straight line, the trace should be low-pass digitally filtered (Section 4.4.3.3) to an empirically determined level such that the peak amplitude is not attenuated.

This is probably best used during Reanalyis when different levels of low-pass digital filtering and different levels of MaxSlope time can be empirically tried. And then compare these results to Maximum Slope (with no Low Pass Digital Filtering) and slope measured with Beg -> End Times. To my surprise, when analyzing synaptic potentials the results using longer Maximum Slope (using Low Pass Digital Filtering) were about identical with the shorter Maximum Slope (using Low Pass Digital Filtering).

4.9.4.3 Begin -> End Times

The third method, the **Begin -> End Times**, merely sets the slope beginning time point and the slope end time points. This method is a much more reliable way of measuring slope when the amplitude of the synaptic potential can be zero (synaptic failures) and there are **no** latency shifts in the synaptic potential.

If this method is chosen, the Pulse Detection Panel appears as in Fig. 4.9.2.1. The slope beginning and end time points are the time fields:

Slope: ___ to ___ ms after pulse

4.9.4.4 Low% -> High% of Peak Amplitude

The fourth method, the Low% -> High% Peak Amplitude calculates the slope beginning time point by using the time where the voltage/amperage was say 20% of the Low% Peak Amplitude value. It calculates the slope end time point by using the time where the voltage/amperage was say 80% of the High% Peak Amplitude value. If this method is chosen, the Pulse Detection Panel appears with the following fields:

Slope: ___ to ___ % peak amplitude

where the first % field is the Low% field and the second % field is the High% field.

All methods have their advantages, but in general the Maximum Slope method is best (with or without low-pass filtering). If the latency between the stimulus pulse and the slope shifts with time, the Maximum Slope (with or without low-pass filtering) or the Low% -> High% Peak Amplitude methods are best. **However, when the EPSP/EPSC amplitude approaches 0, the Low% -> High% method begins to calculate slopes made of noise and therefore gives spurious result. In contrast, the Begin -> End Times method continues to accurately measure the slope when the EPSP/EPSC amplitude approaches 0. This problem does not occur as much with the Maximum Slope method.**

With the proper amount of low-pass digital filtering, a short Maximum Slope will yield the most accurate measurement of the maximum slope, and will best deal with shifts in EPSP/EPSC latency and very low EPSP/EPSC amplitudes.

Slope detections in our group are usually of 0.6 to 2.0 msec duration. When sampling every 100 µsec, this is 7 to 21 AD samples, respectively. Without low-pass digital filtering, the longer the slope duration the better, provided the slope still remains on the (somewhat) ‘straight’ part of the EPSP/EPSC. On-line signal averaging will also decrease slope error measurement.

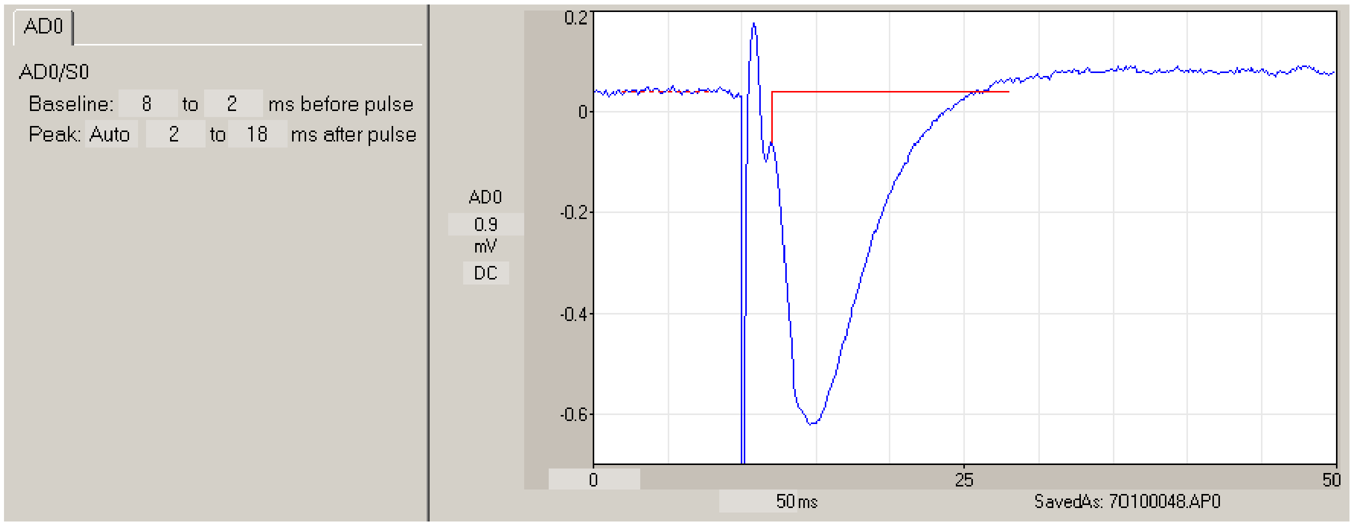
4.9.5 Area

Area calculates the area of the peak more negative or positive than the pre-pulse DC Baseline and is measured in mV*ms or pA*ms. The Area is measured between the

Peak: Auto/Pos/Neg ___ to ___ ms after pulse

time fields shown by the solid horizontal Area line of the Pulse Detection Panel in Fig. 4.9.5.1. Just as with the Peak Amplitude measurement, the Auto/Pos/Neg field determines whether the peak will be Automatically (Auto) determined to be positive or negative, forced to be Positive (Pos), or forced to be Negative (Neg).

Notice, that when the waveform goes to the opposite polarity of the peak, those values are not calculated in the area (for example, in Fig. 4.9.5.1, when the waveform goes positive after 16 ms, the area is only between the first ‘Peak’ time field and up to 16 ms after the pulse.


AD0

AD0/S0

Baseline: 8 to 2 ms before pulse

Peak: Auto 2 to 18 ms after pulse

Fig. 4.9.5.1. Detection of Area of the peak more negative than the pre-pulse baseline. The Area is measured in the range of 2 and 18 ms after the stimulus pulse (solid horizontal line). However, because the waveform goes positive at 16 ms, the area is only measured between 2 and 16 ms after the pulse.

4.9.6 Duration

Duration calculates the duration of the peak or peaks measured between the

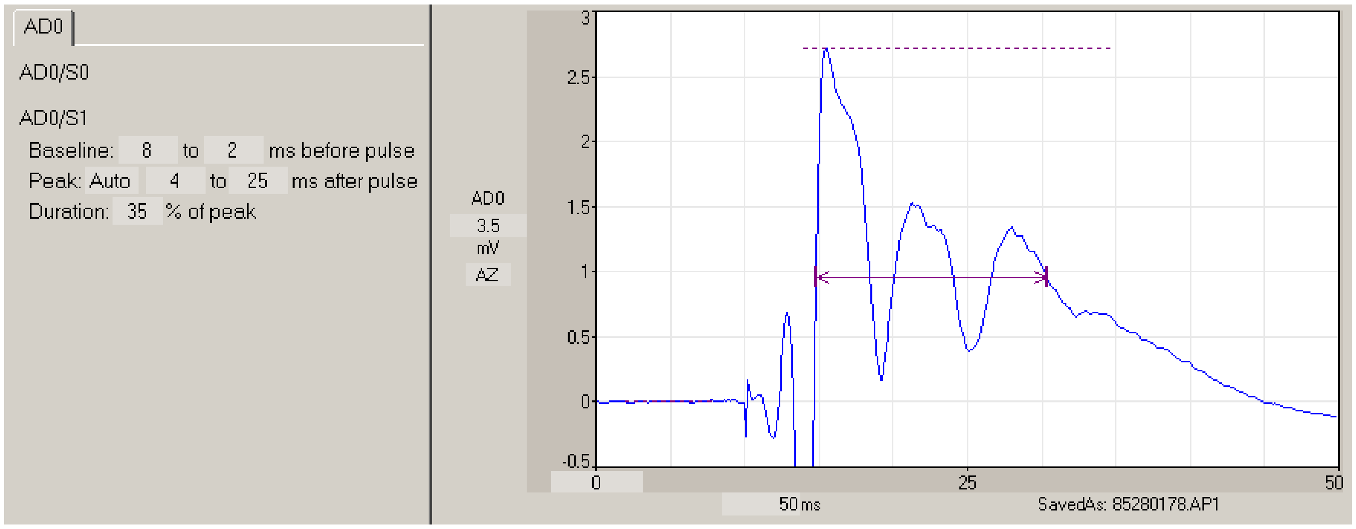
Peak: <u>Auto/Pos/Neg</u> ___ to ___ ms after pulse

time fields shown by the dotted horizontal peak line in the Pulse Detection Panel (Fig. 4.9.6.1). Just as with the Peak Amplitude measurement, the Auto/Pos/Neg field determines whether the peak will be Automatically (Auto) determined to be positive or negative, forced to be Positive (Pos), or forced to be Negative (Neg).

The Duration is measured at a certain percentage of the amplitude between the DC baseline and peak

Dur: ___ % of peak

Duration can measure the duration of bursts (multiple spikes or peaks), and is therefore particularly useful in epilepsy studies for measuring the duration of epileptiform bursts and electrographic seizures.


AD0

AD0/S0

AD0/S1

Baseline: 8 to 2 ms before pulse

Peak: Auto 4 to 25 ms after pulse

Duration: 35 % of peak

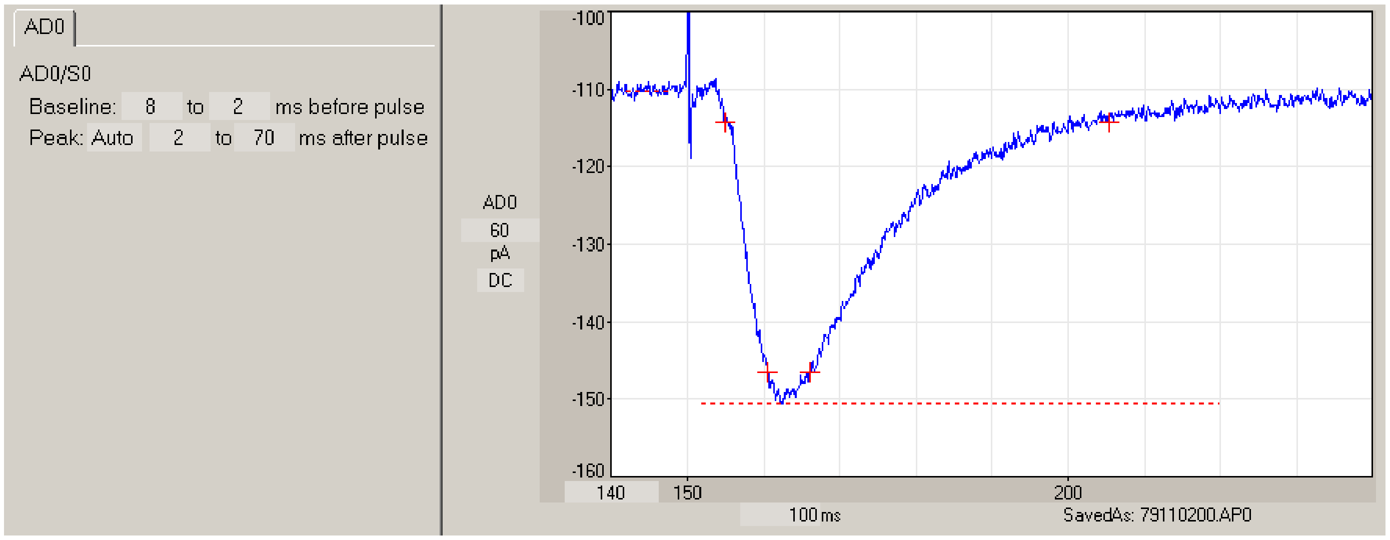
Fig. 4.9.6.1. Detection of Duration in the range of 4 and 25 ms after the stimulus pulse (dotted line). The Duration is measured at 35% of the peak amplitude and is between the arrows (solid line).

4.9.7 Rise Time and Decay Time

Rise Time calculates the time between 10% and 90% of DC baseline to peak on the rising phase of the peak and is shown by + 's. Decay Time calculates the time between 10% and 90% of DC baseline to peak for the falling phase of the peak and is also shown by + 's. Therefore the Rise Time and Decay Time depend on DC baseline settings. The Rise Time and Decay Time is measured between the

Peak: Auto/Pos/Neg ___ to ___ ms after pulse

time fields shown by the dotted horizontal peak line in the Pulse Detection Panel in Fig. 4.9.7.1. Just as with the Peak Amplitude measurement, the Auto/Pos/Neg field determines whether the peak will be Automatically (Auto) determined to be positive or negative, forced to be Positive (Pos), or forced to be Negative (Neg).


AD0

AD0/S0

Baseline: 8 to 2 ms before pulse

Peak: Auto 2 to 70 ms after pulse

Fig. 4.9.7.1. Detection of 10-90% Rise Time and 10-90% Decay Time. The 10% and 90% Rise Times are denoted by the first and second **+** 's. The 10% and 90% Decay Times are denoted by the third and fourth **+** 's. The range for detecting the Rise and Decay Times (shown by the dotted line) was set to 2 to 70 ms after the stimulus pulse.

4.9.8 Coastline

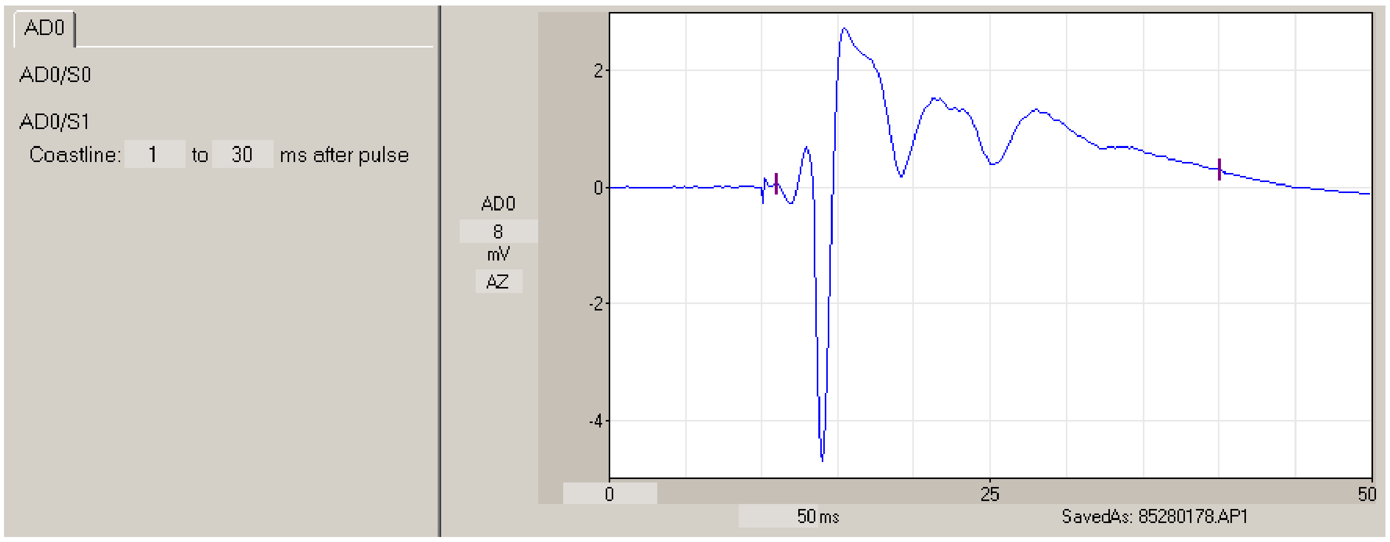
Coastline calculates the amount of vertical deflection between the

CoastLn: ___ to ___ ms after pulse

time fields and is shown to occur between the left and right brackets on the waveform (Fig. 4.9.8.1). (Alternatively, if Peak Amplitude is also being calculated, Peak: Auto/Pos/Neg ___to ___ms after pulse indicates the time fields). The Coastline is measured in mV or pA. Coastline does not depend upon DC baseline.

For example, the coastline of an EPSP of 1 mV amplitude would be 2mV. Coastline is indicative of, and sensitive to, the addition of extra population spikes in an epileptiform burst and can therefore be useful in epilepsy studies.

However, coastline will also increase with extraneous high frequency noise and you should be sure that sufficient external low-pass analog (Section 2.13) or internal low-pass digital filtering (Section 4.4.3.3) is applied to the waveform when the Coastline measurement is made.



Coastline: 1 to 30 ms after pulse

SavedAs: 85280178.AP1

Fig. 4.9.8.1. Detection of Coastline. The Coastline is measured between the left and right bracket located on the waveform.

4.9.9 PopSpike Amplitude and Latency

The PopSpike Amplitude is calculated as the amplitude from the PopSpike peak to the intersection with an interpolated tangent dotted line drawn between the pre-PopSpike peak to the post-PopSpike peak (shown be the solid vertical line in Fig. 4.9.9.1). PopSpike Latency is calculated as the time between the occurrence of stimulus pulse and the PopSpike peak. PopSpike Amplitude and PopSpike Latency do not depend upon DC baseline or Peak Amplitude.

In order to use this tangent autodetection method correctly you must **first set the PopSpike to be positive or negative** by setting the Pos/Neg field in

PSamp: Pos/Neg ___ to ___ ms after pulse

If Peak Amplitude is also to be calculated then the above line is just

PSamp: Pos/Neg

Next, you must set the time range the PopSpike will be detected over by the "__to__ms after pulse" fields above. (If Peak Amplitude it to be calculated then use the time fields in the "Peak: Auto/Pos/Neg __ to __ ms after pulse) This time range is shown between the left and right bracket located on the waveform in Fig. 4.9.9.1.

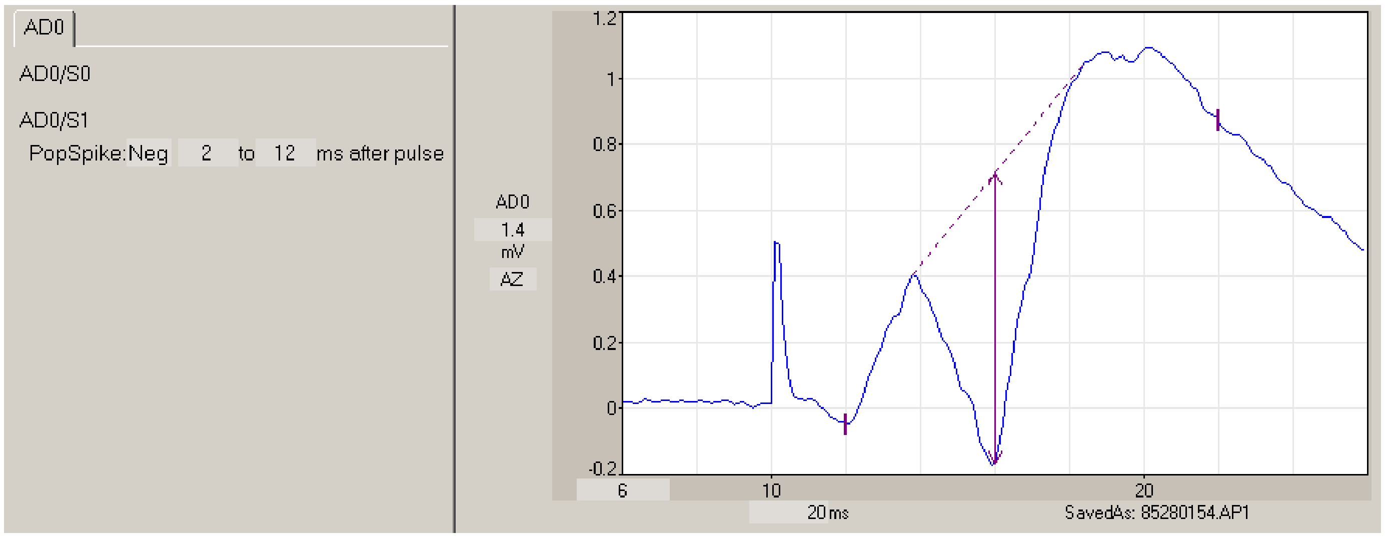

AD0
AD0/S0
AD0/S1
PopSpike:Neg 2 to 12 ms after pulse

Fig. 4.9.9.1. Detection of Population Spike Amplitude and Population Spike Latency. Detection occurs between the left and right brackets on the waveform. The solid vertical line is the PopSpike Amplitude, and the time between the stimulus pulse and the solid vertical line is the PopSpike Latency.

4.9.10 Average Amplitude

The Average Amplitude is the difference between the DC Baseline value and the averaged values between the

AvgAmp : ___ to ___ ms after pulse

time fields and is shown in the AvgAmp solid line of the Pulse Detection Panel (see Fig. 4.9.10.1).

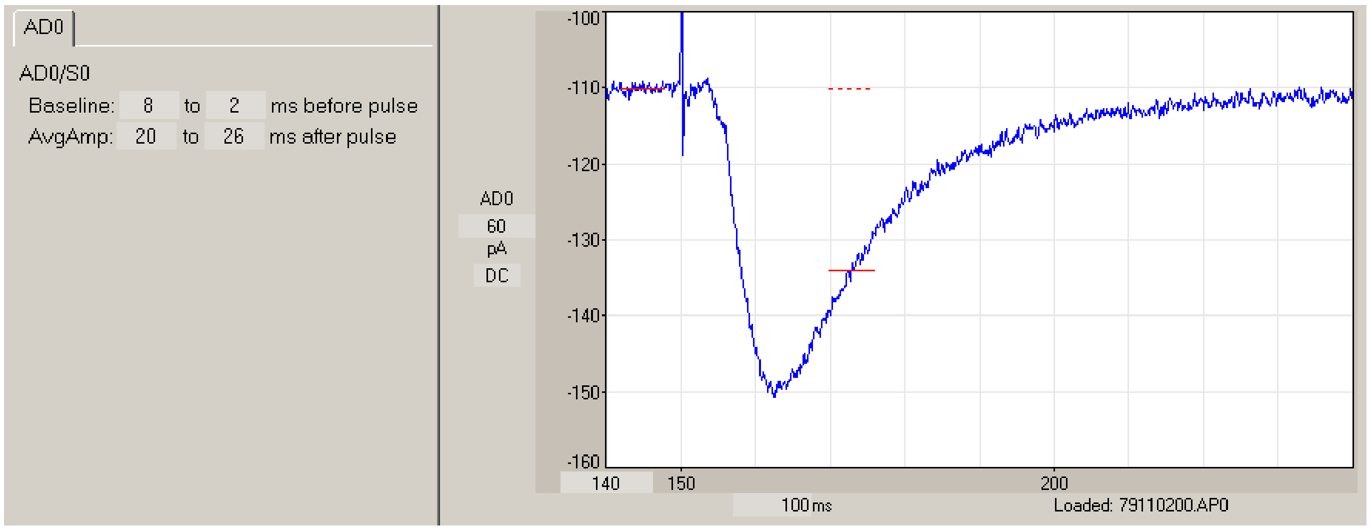
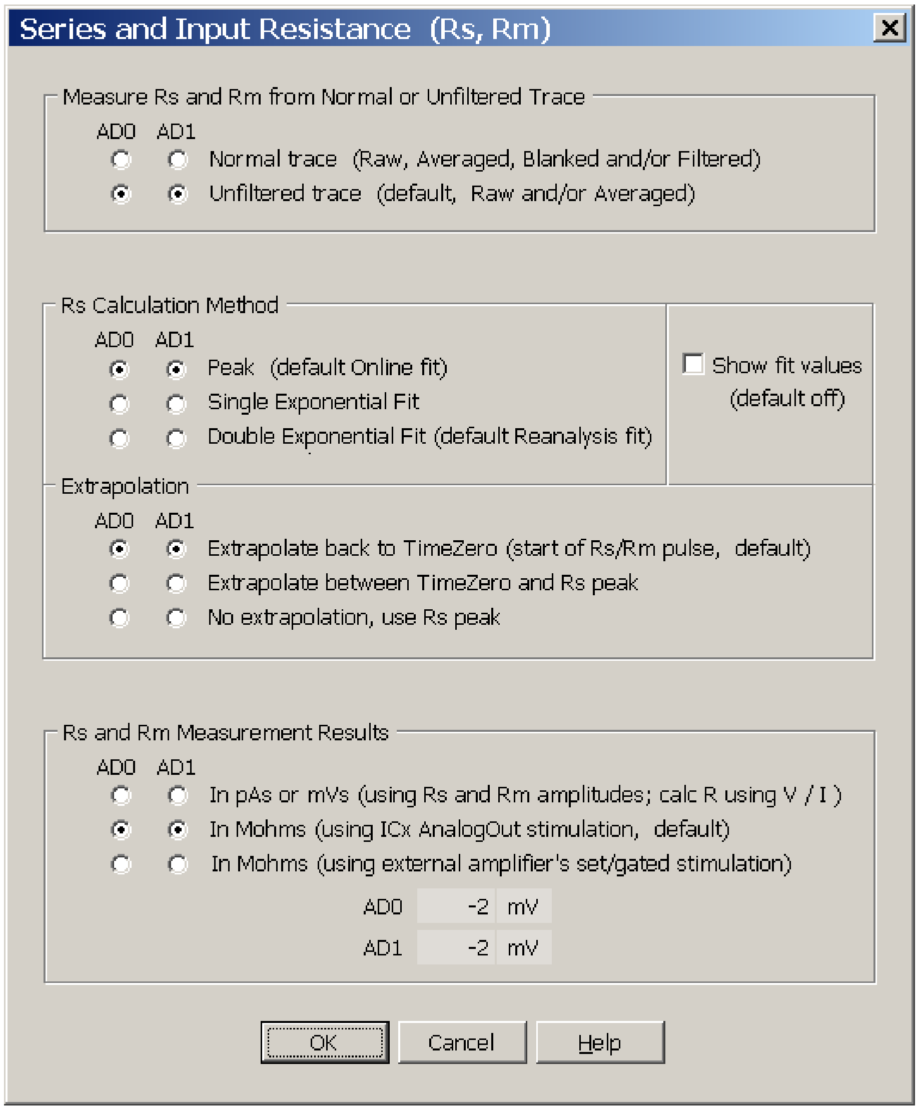

AD0
AD0/S0
Baseline: 8 to 2 ms before pulse
AvgAmp: 20 to 26 ms after pulse

Fig. 4.9.10.1. Detection of Average Amplitude between 23 and 27 ms after the stimulus pulse (solid line) relative to the pre-pulse baseline (dotted line).

4.9.11 Patch Electrode Series Resistance (Rs)

The patch electrode series resistance (Rs) can either be calculated from the 1) **peak** of the capacitative transient, 2) the extrapolated peak of the capacitative transient fitted by a **single exponential**, or 3) the extrapolated peak of the capacitative transient fitted by a **double exponential**. I would like to thank Dr. Tim Benke for the single and double exponential fitting code, and for many discussions about measuring Rs.

For a discussion on the correct measurement of series resistance with and without using exponential curve fitting, and cell input resistance measurement see Ogden and Stanfield (Patch clamp techniques for single channel and whole-cell recording, In: Microelectrode Techniques, The Plymouth Workshop Handbook, Second Edition, Ed. D. Ogden, The Company of Biologists Ltd., Cambridge, 1994).


Series and Input Resistance (Rs, Rm)

Measure Rs and Rm from Normal or Unfiltered Trace

ADO	AD1	
○	○	Normal trace (Raw, Averaged, Blanked and/or Filtered)
●	●	Unfiltered trace (default, Raw and/or Averaged)

Rs Calculation Method

ADO	AD1	
●	●	Peak (default Online fit)
○	○	Single Exponential Fit
○	○	Double Exponential Fit (default Reanalysis fit)

☐ Show fit values (default off)

Extrapolation

ADO	AD1	
●	●	Extrapolate back to TimeZero (start of Rs/Rm pulse, default)
○	○	Extrapolate between TimeZero and Rs peak
○	○	No extrapolation, use Rs peak

Rs and Rm Measurement Results

ADO	AD1	
○	○	In pAs or mVs (using Rs and Rm amplitudes; calc R using V / I)
●	●	In Mohms (using ICx AnalogOut stimulation, default)
○	○	In Mohms (using external amplifier's set/gated stimulation)

ADO -2 mV
AD1 -2 mV

Fig. 4.9.11.1. Series and Input Resistance Calculation Methods Dialog Box. Series Resistance and Input Resistance, Rm can now be made from Channel AD0 and/or AD1.

Furthermore, Rs can now be measured from the Peak of the capacitative transient or with single or double exponential curve fitting.irst, for Rs to be measured you have to :

1) set ADx DataType to pA (see Section 2.12.1)
2) set ICx to mV (see Section 4.6.1.)
3) have an RsRm step in sweep stimulation (see Section 4.7.1.)

Next, the method of determining how patch electrode series resistance (Rs) is calculated is chosen by using the menu commands:

AmpFile -> Series and Input Resistance Calculation Methods...

to bring up the Series and Input Resistance Dialog Box (Fig. 4.9.11.1).

The first choice you have to make, using the **Measure Rs and Rm from Normal or Unfiltered Trace** radiobuttons in Fig. 4.9.11.1, is whether to make Rs and Rm measurements from the **Normal trace** which may or may not be internally low-pass digitally filtered, or the **Unfiltered trace**, which is always unfiltered, and can be either a Raw or Averaged trace. **The reason for this choice is that often (particularly during online analysis) you want to low-pass filter the trace in order to measure Peak Amplitude well, but you do not want to filter the peak capacitative transient in order to get good Rs measurement.**

Fig. 4.9.11.2 shows what I think is the best way of making Rs and PkAmp measurements online. The Rs measurement made from the capacitative transient peak to a step voltage pulse from an **Unfiltered trace** (using the Unfiltered trace selection) (see the 'Rs' black line on the negative transient peak of the gray unfiltered trace), whereas the PkAmp measurement is made from the **filtered trace** with the digital filtering turned on (see the red lines on the blue filtered trace).

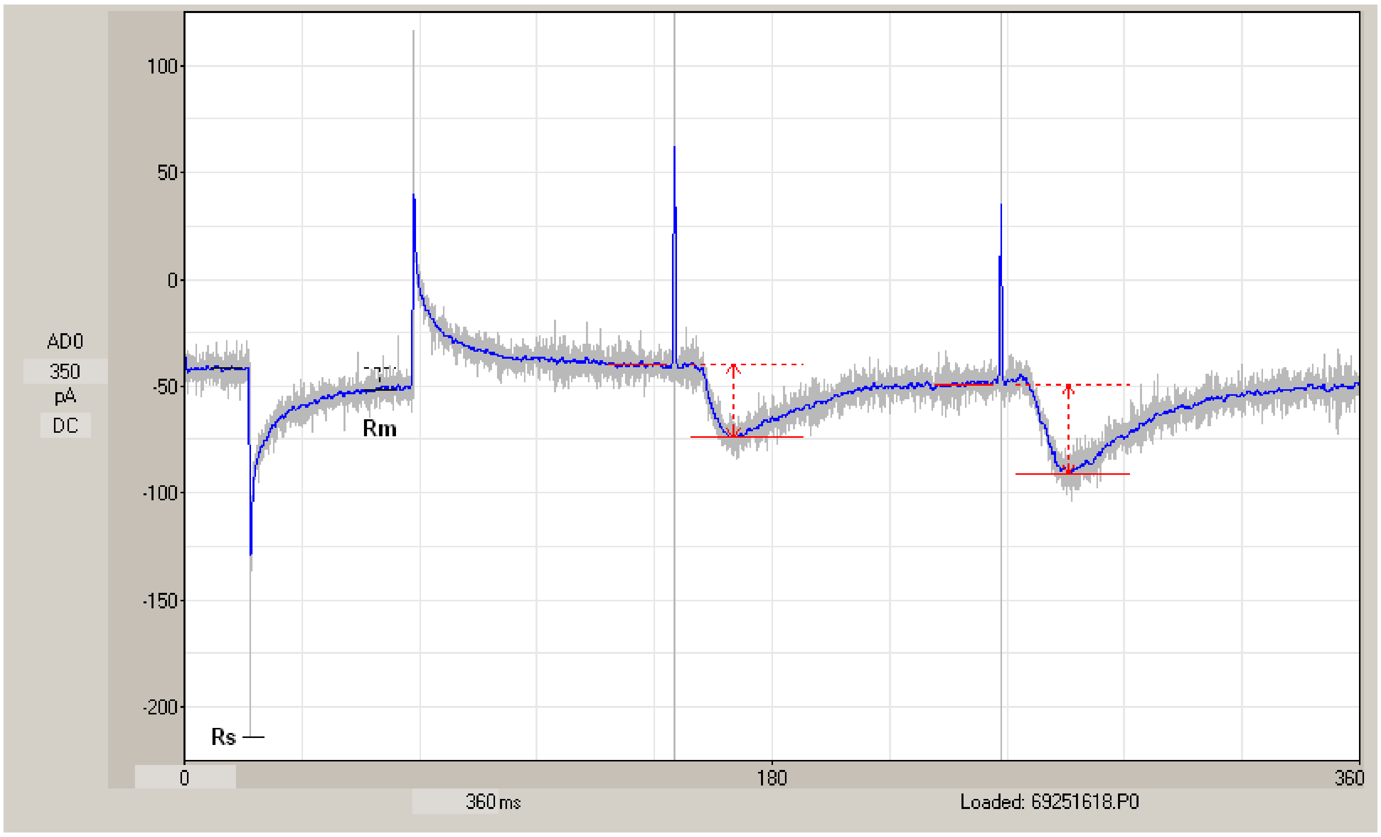


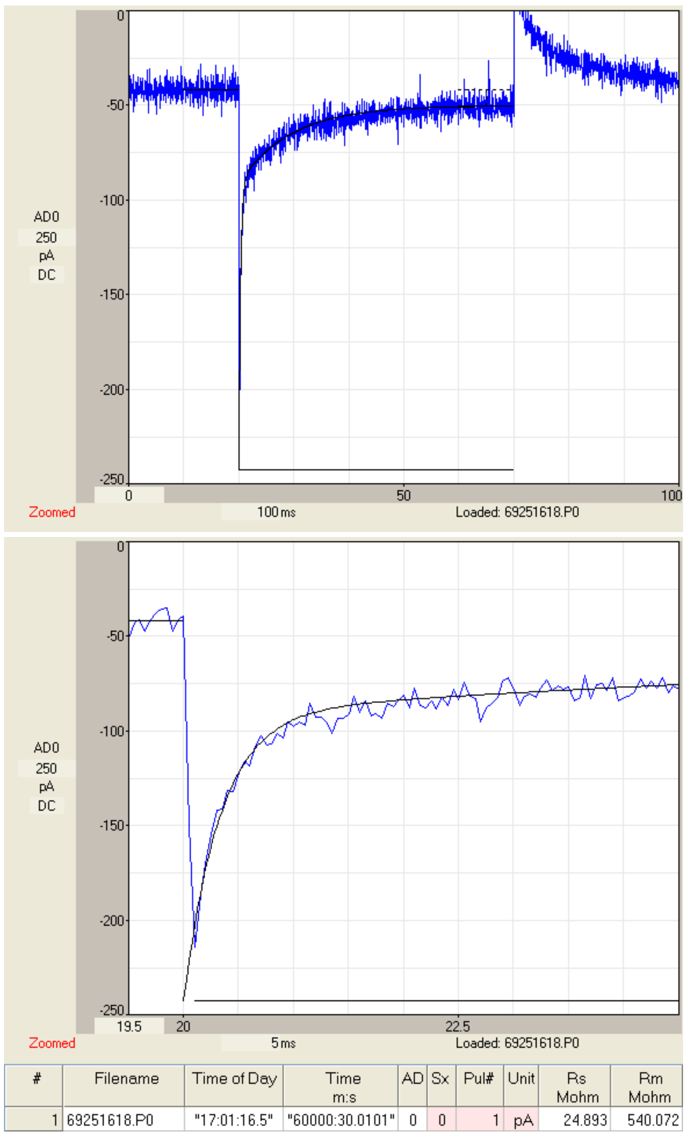
Fig. 4.9.11.2. The best way to measure Rs and PkAmp online. Rs measurement was made from the capacitative peak transient from the **unfiltered trace** so as to not attenuate the transient peak (20 KHz sampling, 10 KHz external filtering). PkAmp measurement was made from **filtered trace** so as to not measure the superimposed excess noise (20 KHz sampling, 10 KHz external filtering, 500 Hz internal filtering). The Rs and Rm measurements were made from the gray 10 KHz unfiltered trace, whereas the EPSCs would be measured from the blue internal 500 Hz filtered trace.

The second choice, using the **Rs Calculation Method** radiobuttons in Fig. 4.9.11.1, sets whether you want to calculate Rs from the **peak** of the capacitative transient (Figs. 4.9.11.2 and 4.9.11.4D), an extrapolated peak fitted by a **single exponential fit** to the capacitative transient, or an extrapolated peak fitted by a **double exponential fit** to the capacitative transient.

In order to **accurately measure Rs from the Peak** of the capacitative transient, **the pipette capacitance must have been cancelled out.** WinLTP has a maximum sampling rate of **40 KHz** insuring that the Peak capacitative transient will be captured pretty well (even at 20 KHz). Because the **peak** of the capacitative transient is the most trouble-free method, it is the **default online method**.

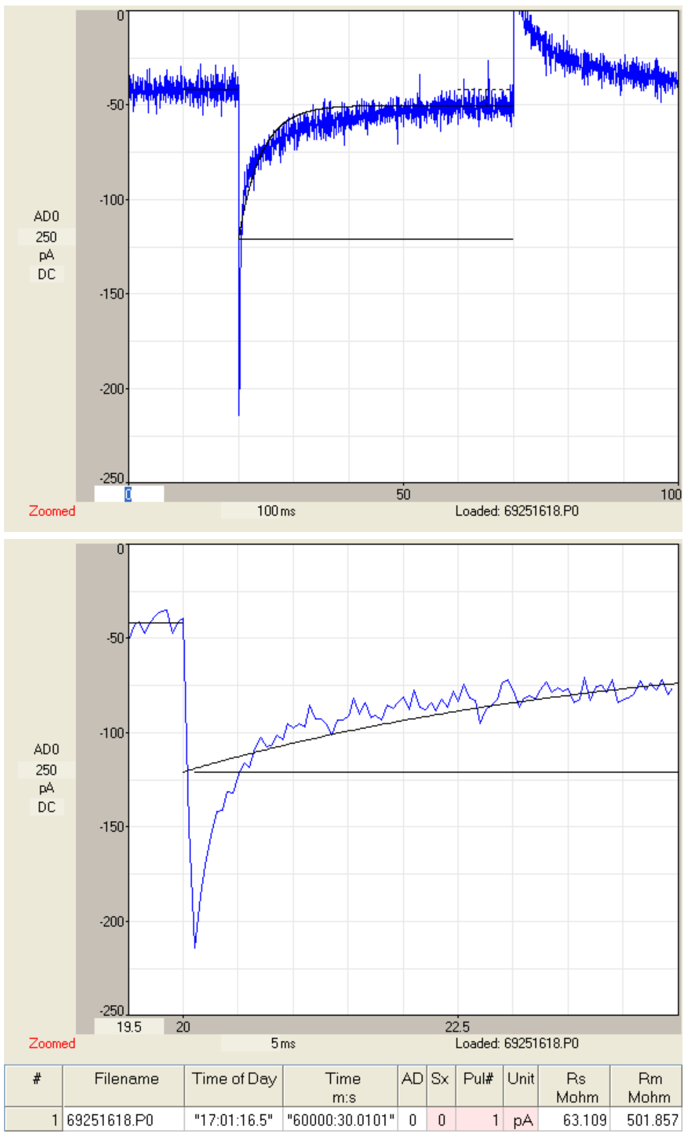
Fig. 4.9.11.3 shows **double and single exponential fits** to capacitative transients in hippocampal neurons extrapolated back to TimeZero peak to measure Rs. The exponential curve fit is from the beginning to the end of the Rs/Rm pulse. Note that in these hippocampal neurons, the double exponential fit gives a ‘good’, ‘correct’ fit with an Rs of ~25 Mohms, whereas the single exponential fit gives a ‘bad’, ‘incorrect’ fit with an Rs of ~63 Mohms.

Double Exponential Fit (correct)



#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	Rs Mohm	Rm Mohm
1	69251618.P0	"17:01:16.5"	"60000:30.0101"	0	0	1	pA	24.893	540.072

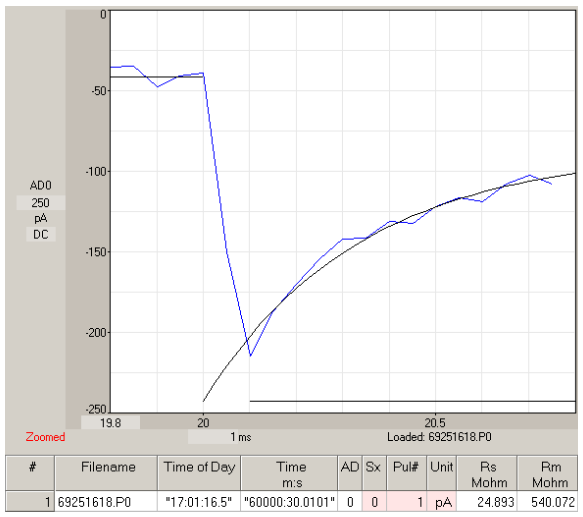
Single Exponential Fit (incorrect)



#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	Rs Mohm	Rm Mohm
1	69251618.P0	"17:01:16.5"	"60000:30.0101"	0	0	1	pA	63.109	501.857

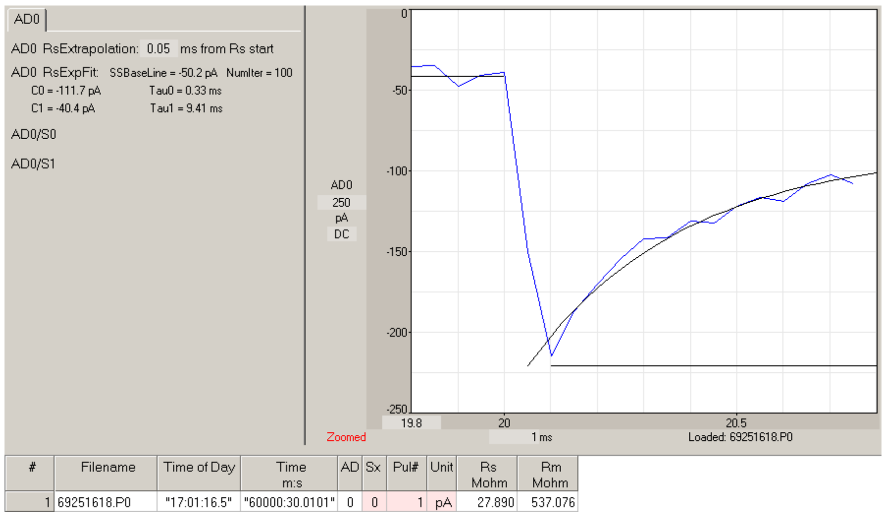
Fig. 4.9.11.3. Measuring patch-electrode series resistance Rs in a hippocampal neuron recording using double exponential curve fitting (left traces) at low (top) and high (bottom) sweep magnification, and single exponential curve fitting (left traces) at low (top) and high (bottom) sweep magnification.

A Extrapolate Back to TimeZero



#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	Rs Mohm	Rm Mohm
1	69251618.P0	"17:01:16.5"	"60000:30.0101"	0	0	1	pA	24.893	540.072

B Extrapolate Back 0.05ms from Rs/Rm Pulse Start



AD0

AD0 RsExtrapolation: 0.05 ms from Rs start

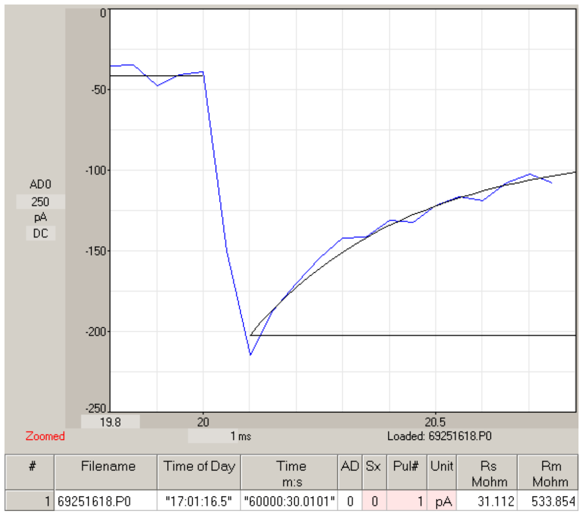
AD0 RsExpFit: SSBaseLine = -50.2 pA NumIter = 100
C0 = -111.7 pA Tau0 = 0.33 ms
C1 = -40.4 pA Tau1 = 9.41 ms

AD0/S0

AD0/S1

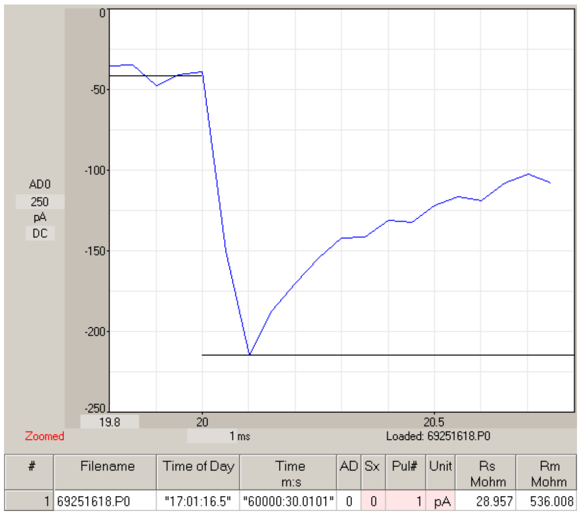
#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	Rs Mohm	Rm Mohm
1	69251618.P0	"17:01:16.5"	"60000:30.0101"	0	0	1	pA	27.890	537.076

C No extrapolation, fit to Rs Peak time



#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	Rs Mohm	Rm Mohm
1	69251618.P0	"17:01:16.5"	"60000:30.0101"	0	0	1	pA	31.112	533.854

D Analyze at Rs Peak



#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	Rs Mohm	Rm Mohm
1	69251618.P0	"17:01:16.5"	"60000:30.0101"	0	0	1	pA	28.957	536.008

Fig. 4.9.11.4. The effect of different extrapolation times toward TimeZero on Rs measurements. All fits were double exponential and on the same cell as in Fig 4.9.11.3. A) Extrapolate back to start of the Rs/Rm pulse, TimeZero; Rs = ~25 Mohm. B) Extrapolate back to 0.05ms after start of the Rs/Rm pulse; Rs = ~28 Mohm. C) No extrapolation back from Rs peak time; Rs = ~31 Mohm. D) Rs measurement

taken from the Rs Peak; Rs = ~29 Mohm. This is NOT the same as the double exponential fit at Rs peak time in C.

The third choice, the **Show Fit Values** checkbox in Fig. 4.9.11.1, determines whether the fit values (baseline, coefficients and taus) of the single or double exponential fit are shown, as shown in the upper left part of Fig. 4.9.11.4B. The default is off.

The fourth choice, from the **Extrapolation** radiobuttons in Fig. 4.9.11.1, determines whether the peak of the single or double exponential fit to the capacitative transient: 1) Extrapolates back to **TimeZero**, eg to the start of the RsRm pulse (and this is the default) (Fig. 4.9.11.4A), 2) Extrapolates back to a time between TimeZero and the Rs peak (Fig. 4.9.11.4B), and 3) No extrapolation, just use the time to the Rs peak (Fig. 4.9.11.4C).

Because of lag in the patch clamp amplifier stimulation, the correct time to take the peak will be somewhere between TimeZero and the Rs peak time. One way to get an estimate of the correct extrapolation time is to take a model cell, and see how much shift from TimeZero gives the best Rs measurement for the model cell.

Using the peak extrapolated back to TimeZero will give a slight **underestimate** of Rs (~25 Mohms in Fig 4.9.11.4A). But using the Rs peak time will give a slight **overestimate** of Rs (~31 Mohms in Fig 4.9.11.4C). However, **changes** in Rs, which many researchers are most concerned about, will be picked up using any of these Rs analyses.

The fifth choice you have to make, from the **Rs and Rm Measurement Results** radiobuttons in Fig. 4.9.11.1, is whether to get the Rs and Rm measurements: 1) directly in pA or mV (from you have to convert to resistance by Rs = V_{Pulse} / I_{Peak} and Rm = $(V_{Pulse} / I_{SteadyState}) – Rs$), 2) in Mohms using the IC0 or IC1 analog out stimulation (in mV for voltage-clamping and nA or pA for current clamping) (the default), or 3) in Mohm using a gated stimulation pulse set on the patch clamp amplifier (in mV, nA or pA) and entered into the fields below. For choices 2) and 3), Rs is calculated from the equation:

$$Rs = V_{Pulse} / I_{Peak}$$

where V_{pulse} is the amplitude of the RsRm voltage clamp test pulse relative to the baseline of the preceding epoch, and I_{Peak} is the amplitude of the peak capacitative transient current (extrapolated back toward the start of the V_{Pulse} if measured using exponential curve fitting).

4.9.12 Cell Input Resistance (Rm)

Membrane or cell input resistance (Rm) detection occurs automatically. The Rm measurement is the difference between the averaged PreRmBaseline and the averaged RmPulse (see Fig. 4.9.11.2). That part of the Rm measurement taken during the RmPulse is an average taken between 80% and 100% of the RmPulse if no low-pass digital filtering (or 70% and 90% of the RmPulse if digitally filtered). That part of the Rm measurement taken during the PreRmBaseline is taken between 10% and 100% of that period, or a maximum duration of the time equal to 80-100% of the RmPulse if no digital filtering (or 10% and 90% of that period, or a maximum duration of the time equal to 70-90% of the RmPulse with filtering).

The cell input resistance Rm is calculated as:

$$Rm = V_{Pulse} / I_{SteadyState} - Rs$$

where V_{pulse} is the amplitude of the RsRm voltage clamp test pulse relative to basline of the preceding epoch, $I_{SteadyState}$ is the amplitude of the current measured between the baseline and 70% and 90% of the pulse when the current has reached steady state, and Rs is the patch electrode series resistance. (See Appendix D for an Rm calculation error in WinLTP094 and earlier.)

In WinLTP, Rs is almost always measured during patch clamp voltage clamping (assuming you click the AnalysesToDo Rs check box), but is not measured during patch clamp current clamping, and therefore Rs = 0 in this case, and Rm = V_{Pulse} / $I_{SteadyState}$. Furthermore, during whole cell single electrode voltage clamping, where series resistance is theoretically zero, you would not measure Rs and it would therefore be set to Rs = 0, and Rm = V_{Pulse} / $I_{SteadyState}$. For intracellular current clamping using a bridge circuit, Rs would also not be measured and would therefore be set to Rs = 0, and Rm = V_{Pulse} / $I_{SteadyState}$.

4.10 Analyzing All S0- and S1-Evoked Postsynaptic Responses in Both AD channels in a Sweep

WinLTP is capable of acquiring data from 2 channels and analysing all S0- and S1 evoked synaptic responses on both acquisition channels. Fig. 4.10.1 illustrates this with one acquisition channel extracellularly recording the CA1 dendritic layer of the hippocampal slice, and the other acquisition channel extracellularly recording the CA1 cell body layer (panels AD0 and AD1 respectively in Fig. 4.10.1, middle traces).

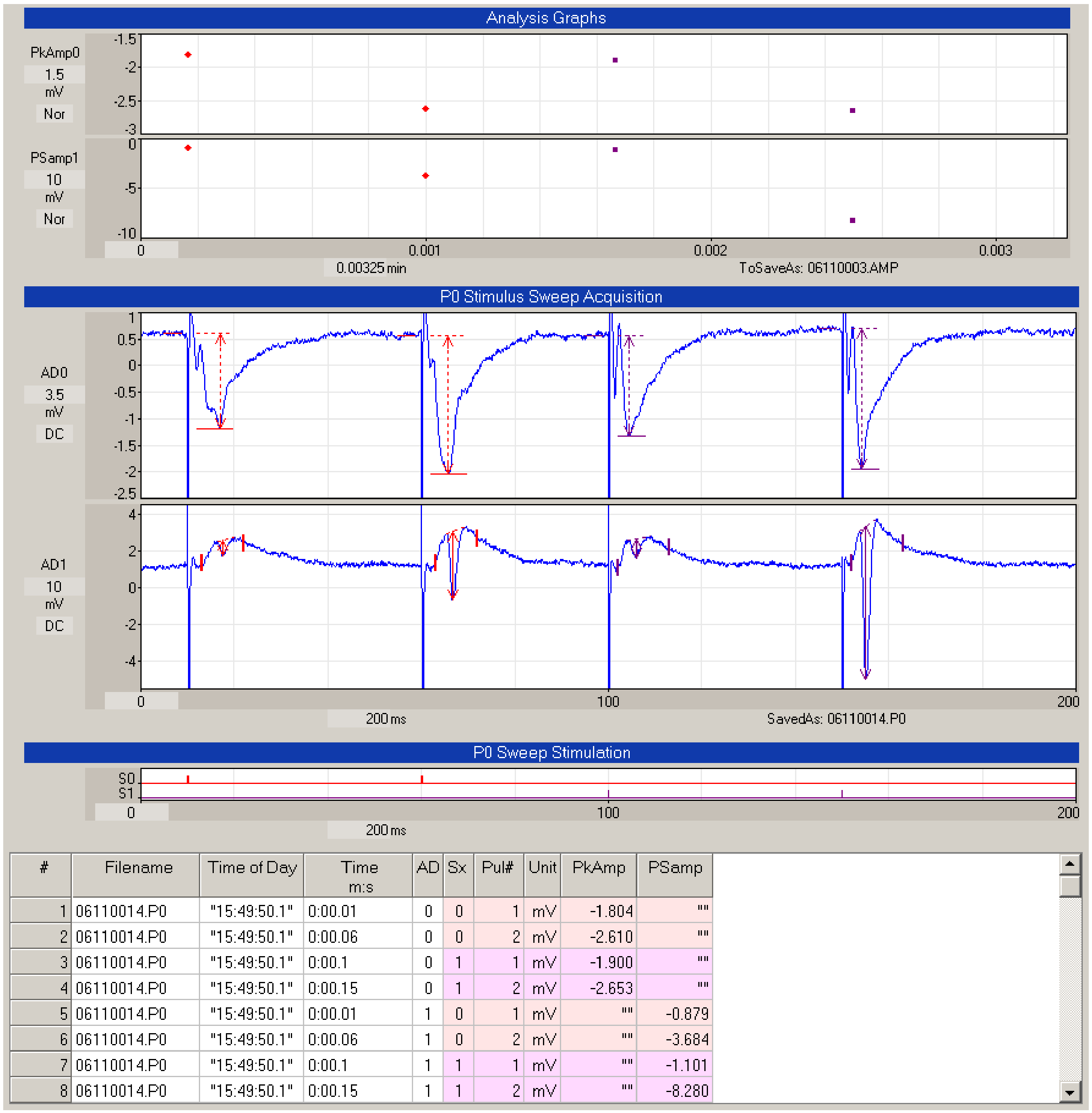
The stimulation during the sweep consists of paired-pulse S0 stimulation (red trace) followed by paired-pulse S1 stimulation (magenta trace) (Fig. 4.10.1, bottom traces).

Both S0 and S1 stimulations evoke fEPSPs in the CA1 dendritic layer (AD0) and fEPSPs with overriding population spikes in the CA1 cell body layer (AD1). All S0-evoked synaptic responses are analysed and shown by red lines in AD0 and AD1, and all S1-evoked synaptic responses are analysed and shown by magenta lines (Fig. 4.10.1, middle traces).

The values of the peak amplitude calculations for AD0 are shown in PkAmp0 (red triangles for S0 evoked responses, and magenta squares for S1-evoked responses; Fig. 4.10.1, top traces). The values of the population spike amplitude calculations for AD1 are shown in PSamp1 (red triangles for S0 evoked responses, and magenta squares for S1-evoked responses).

When the LTP Program performs an on- or off-line analysis of one or more acquisition sweeps, it puts the results into the Spreadsheet (Fig. 4.10.1, bottom panel). **"TimeOfDay"** shows the time the sweep began. **"Time m:s"** shows the time of the stimulus pulse from when analysis starts (in this case simply the beginning of this sweep since this was the only sweep analyzed). **"AD"** shows the AD channel from which the synaptic response was obtained (channel AD0 is calculated first). **"Sx"** shows whether S0 or S1 stimulation was used to evoke the synaptic response (S0-evoked responses are calculated first). **"Pul#"** shows the number of the S0 or S1 pulse that evokes the synaptic response (pulse number is calculated sequentially). **"PkAmp"** shows peak amplitude values of the S0- and S1-evoked fEPSPs in channel AD0. **"PSamp"** shows the population spike amplitude values of the S0- and S1-evoked responses in channel AD1. When the PkAmp or PSamp values are not calculated (e.g. PkAmp values for AD1 and PSamp values for AD0), they are set to 0.

The Spreadsheet can then be saved to an ASCII text file (the *.AMP file) or saved to an Excel file (an *.XLS file). These *.AMP and *.XLS files can be loaded into a spreadsheet program and the columns can be sorted as desired.



#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	PkAmp	PSamp
1	06110014.P0	"15:49:50.1"	0:00.01	0	0	1	mV	-1.804	""
2	06110014.P0	"15:49:50.1"	0:00.06	0	0	2	mV	-2.610	""
3	06110014.P0	"15:49:50.1"	0:00.1	0	1	1	mV	-1.900	""
4	06110014.P0	"15:49:50.1"	0:00.15	0	1	2	mV	-2.653	""
5	06110014.P0	"15:49:50.1"	0:00.01	1	0	1	mV	""	-0.879
6	06110014.P0	"15:49:50.1"	0:00.06	1	0	2	mV	""	-3.684
7	06110014.P0	"15:49:50.1"	0:00.1	1	1	1	mV	""	-1.101
8	06110014.P0	"15:49:50.1"	0:00.15	1	1	2	mV	""	-8.280

Fig. 4.10.1. Two channel acquisition and analysis all S0- and S1-evoked synaptic responses in both acquisition channels. **Analysis Graphs:** Top two graphs show calculated values for the peak amplitude of the synaptic responses in AD0 (PkAmp0), and the pop-spike amplitude from the responses in AD1 (PSamp1). **P0 Stimulus Sweep Acquisition:** The middle two graphs show channel AD0 (a recording of fEPSPs from the CA1 dendritic layer), and channel AD1 (a recording of population spikes from the CA1 cell body layer). The peak and pop-spike amplitude calculations of paired S0-evoked synaptic responses are shown in red, and of paired S1-evoked synaptic responses in magenta. **P0 Sweep Stimulation:** S0 paired-pulse stimulation (red top trace) and S1 paired-pulse stimulation (magenta bottom trace). **Spreadsheet:** Showing analysis of EPSPs and PopSpikes from the analysis of the sweep above. The 'Time m:s' field shows the time of the stimulus pulse from when analysis starts (in this case simply the beginning of this sweep). Channel AD0 is calculated first, S0-evoked EPSPs are calculated first, and pulse number is calculated sequentially. When the PkAmp values (for AD1) or the PSamp values (for AD0) are not calculated, they are set to "". This Spreadsheet can then be saved to an *.AMP text file and/or to an Excel *.XLS file.

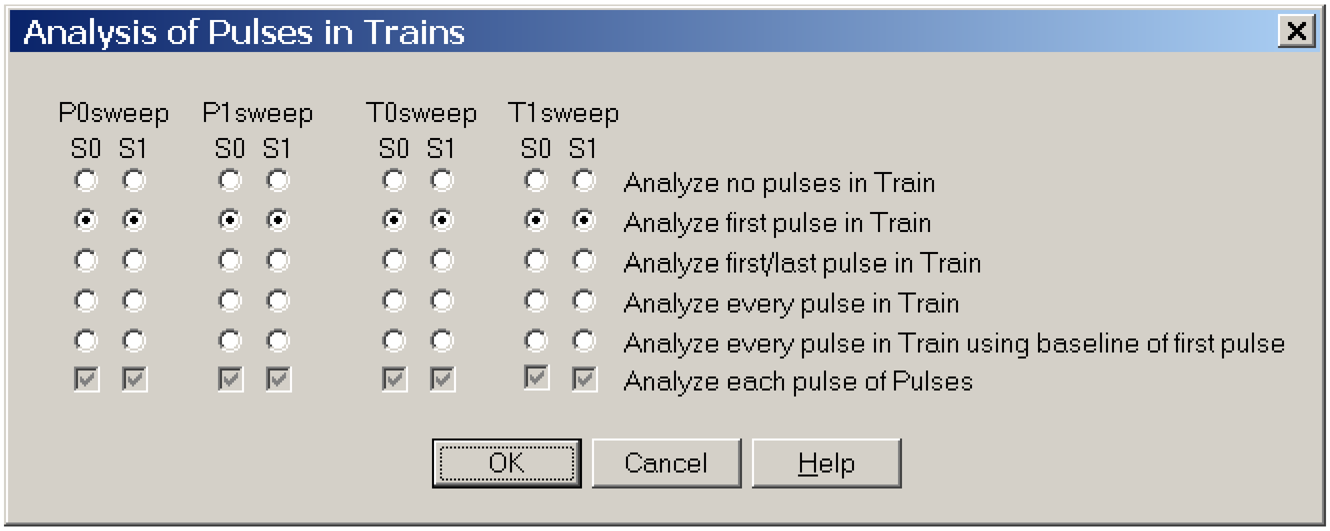
4.11 Special Analyses of Trains

Sometimes the experimenter is interested in examining each postsynaptic response evoked by a stimulus pulse in a train, in which case the baseline and synaptic response of each pulse is analyzed (as with the ‘two pulse trains’ in Fig. 4.10.1). Alternatively, the synaptic responses evoked by train stimulation can be treated as a whole train object in a special manner by WinLTP as described below.

To determine how to analyze trains use the menu commands:

AmpFile -> Train Analysis...

to call up the Analysis of Pulses in Trains dialog box (Fig. 4.11.1).


Analysis of Pulses in Trains

P0sweep		P1sweep		T0sweep		T1sweep		
S0	S1	S0	S1	S0	S1	S0	S1	
○	○	○	○	○	○	○	○	Analyze no pulses in Train
●	●	●	●	●	●	●	●	Analyze first pulse in Train
○	○	○	○	○	○	○	○	Analyze first/last pulse in Train
○	○	○	○	○	○	○	○	Analyze every pulse in Train
○	○	○	○	○	○	○	○	Analyze every pulse in Train using baseline of first pulse
✓	✓	✓	✓	✓	✓	✓	✓	Analyze each pulse of Pulses

Fig. 4.11.1. Analysis of Pulses in Trains dialog box.

For extracellular electrode stimulation S0 and S1, for Sweeps P0, P1, T0 and T1, it allows five choices to be made:

1) Analyze **no** pulses in Train
2) Analyze the **first** pulse in Train, and then extend the detection over the whole train thereby analyzing the whole train (Section 4.11.2)
3) Analyze the first/last pulse in the train by using the baseline of the first pulse in the train with the response of the last pulse in the train (Section 4.11.3)
4) Analyze **every** pulse in train (Section 4.11.4)
5) Analyze **every** pulse but use the **baseline of the 1st pulse** as the baseline of all the pulses (Section 4.11.5)

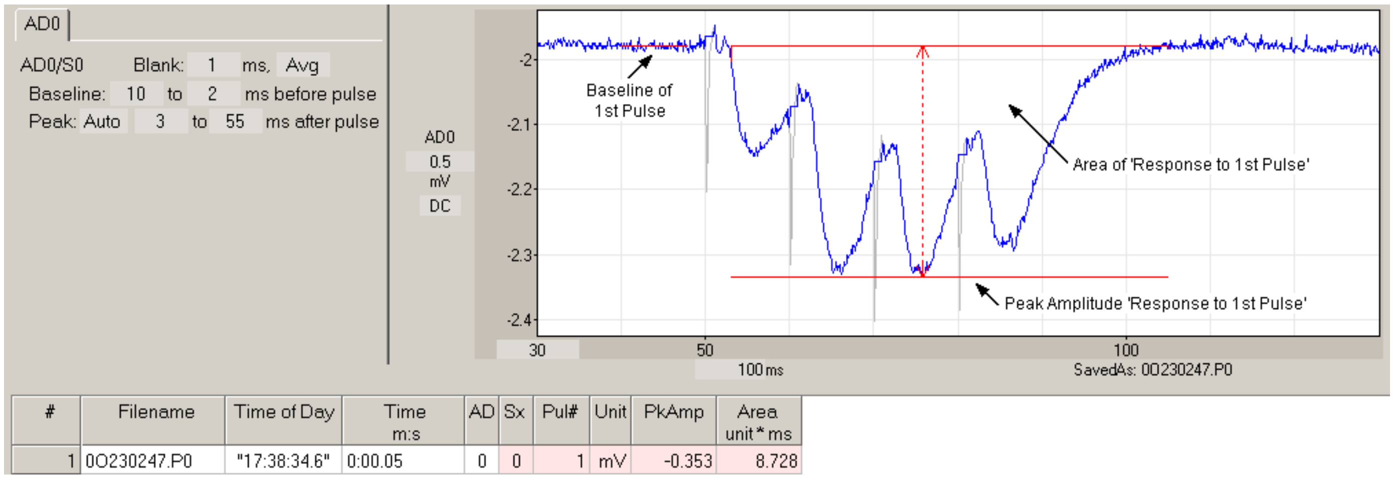
The check box to analyze, or not analyze, **each pulse of Pulses** is not yet implemented.

4.11.1 Analyze no pulses in train

With this radiobutton selected, no analysis of the pulses in the train occurs (this is also not yet implemented).

4.11.2 Analyze whole train by analyzing only first pulse in train but detecting whole train

Second, trains can also be analyzed as a single entity. If the baseline and response of only the first train pulse is used, all stimulus artifacts are blocked, and the time of measurement is set sufficiently long to encompass the whole train, then the synaptic response of the entire train will be measured (Fig. 4.11.2.1). With this analysis, the peak amplitude of the largest EPSP in the train and the area of the synaptic response of the entire train can be obtained. Removal of stimulus artifacts is necessary to permit accurate calculation of area and peak amplitude without contamination by stimulus artifacts occurring near the fEPSP peak. Note the one measurement in the spreadsheet, one for the first pulse, e.g. the whole train.


AD0
-2
-2.1
-2.2
-2.3
-2.4
30
50
100
100 ms
Baseline of 1st Pulse
Area of 'Response to 1st Pulse'
Peak Amplitude 'Response to 1st Pulse'
AD0
0.5
mV
DC

AD0/S0 Blank: 1 ms, Avg
Baseline: 10 to 2 ms before pulse
Peak: Auto 3 to 55 ms after pulse

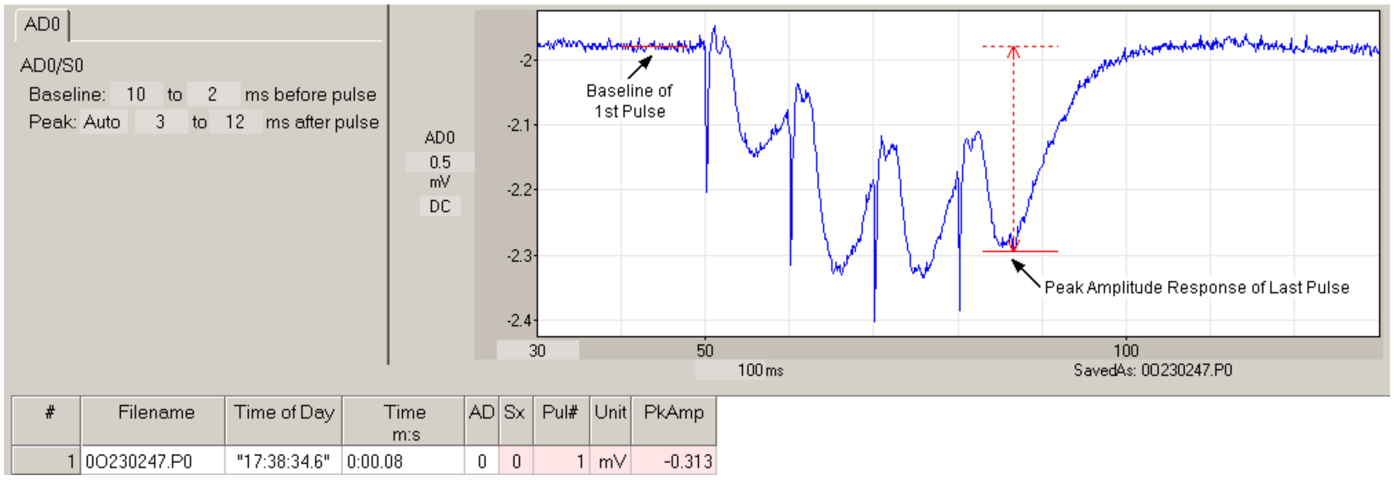
SavedAs: 00230247.P0

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	PkAmp	Area unit*ms
1	00230247.P0	"17:38:34.6"	0:00.05	0	0	1	mV	-0.353	8.728

Fig 4.11.2.1. Analysis of trains as a single entity by using the baseline and response of only the first pulse in the train, but with the time of measurement set to a sufficiently long duration after the first stimulus pulse (3 to 55 ms here) to encompass the whole train. This measures the peak amplitude of the largest fEPSP in the train and the area of the entire synaptic response of the train. Accurate calculation of area requires removal of stimulus artifacts.

4.11.3 Analyze train using baseline of the first pulse and response of the last pulse

Third, trains can also be analyzed by using the baseline of the first pulse and response of the last pulse (Fig. 4.11.3.1). This measures the response at the end of the train regardless of the length of the train. In addition to measuring the last synaptic response of a train, this is useful for measuring responses that only occur after the train has ended, such as post-inhibitory rebound.



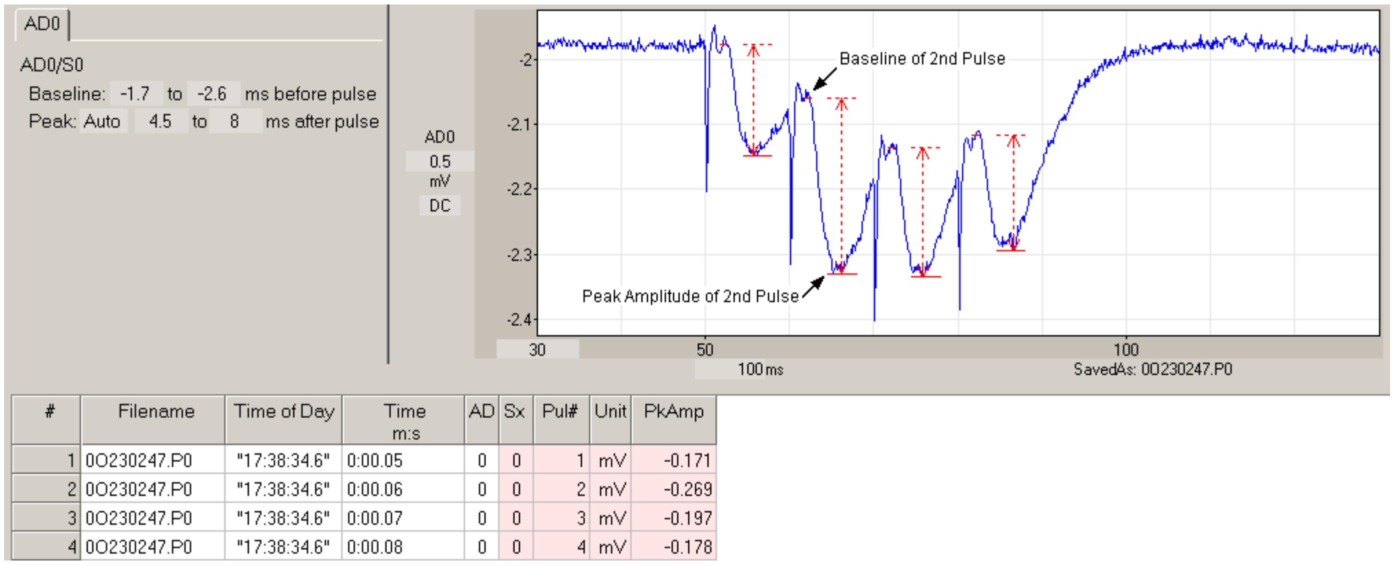
AD0/S0
Baseline: 10 to 2 ms before pulse
Peak: Auto 3 to 12 ms after pulse

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	PkAmp
1	00230247.P0	"17:38:34.6"	0:00.08	0	0	1	mV	-0.313

Fig. 4.11.3.1. Analysis of a train using baseline of the first pulse and response of the last pulse.

4.11.4 Analyze every pulse in the train

Fig. 4.11.4.1 shows the analysis of every pulse in the train. Note that to achieve the best measurement of peak amplitude the baseline of the pulse is actually shifted to the right of the pulse (by entering negative values) to get to the most positive part of the last EPSP before the next EPSP begins.



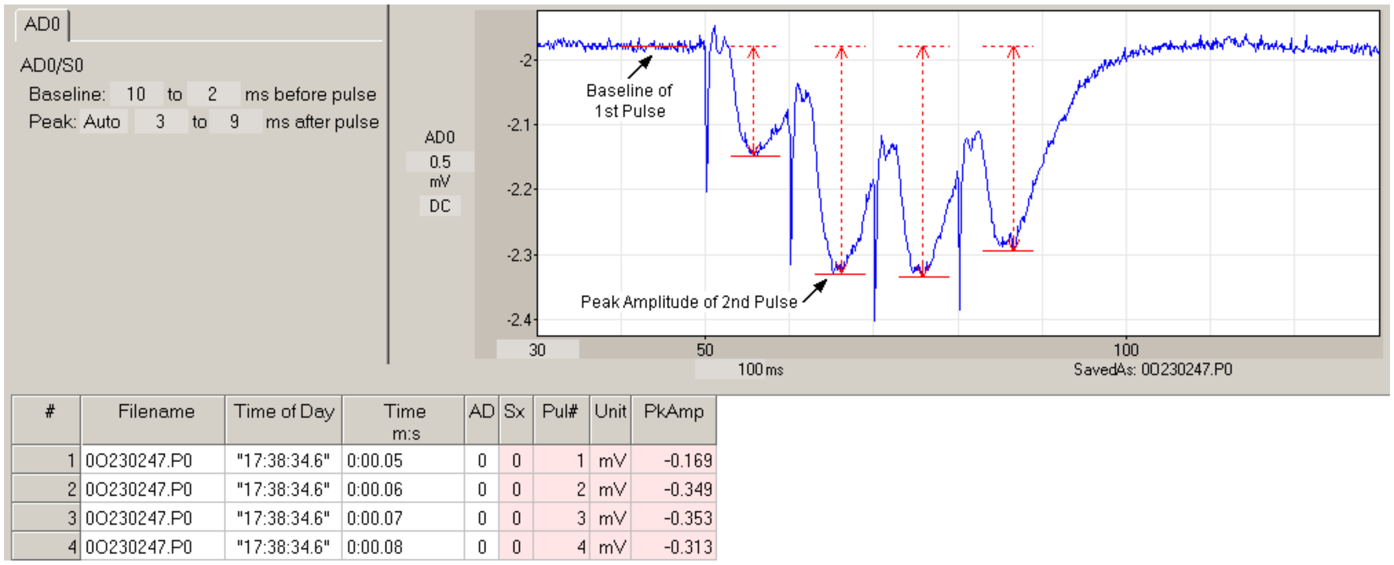
AD0/S0
Baseline: -1.7 to -2.6 ms before pulse
Peak: Auto 4.5 to 8 ms after pulse

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	PkAmp
1	00230247.P0	"17:38:34.6"	0:00.05	0	0	1	mV	-0.171
2	00230247.P0	"17:38:34.6"	0:00.06	0	0	2	mV	-0.269
3	00230247.P0	"17:38:34.6"	0:00.07	0	0	3	mV	-0.197
4	00230247.P0	"17:38:34.6"	0:00.08	0	0	4	mV	-0.178

Fig. 4.11.4.1. Analysis of every pulse in the train.

4.11.5 Analyze every pulse in train using the baseline of first pulse as the baseline for each pulse

First, the synaptic responses to each train pulse may be analyzed relative to the baseline of the first pulse in the train. Fig. 4.11.5.1 shows the peak amplitude measurement of each fEPSP in a 4 pulse train relative to the prestimulus baseline of the first pulse. Note the four measurements in the spreadsheet, one for each pulse.



AD0

AD0/S0

Baseline: 10 to 2 ms before pulse

Peak: Auto 3 to 9 ms after pulse

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	PkAmp
1	00230247.P0	"17:38:34.6"	0:00.05	0	0	1	mV	-0.169
2	00230247.P0	"17:38:34.6"	0:00.06	0	0	2	mV	-0.349
3	00230247.P0	"17:38:34.6"	0:00.07	0	0	3	mV	-0.353
4	00230247.P0	"17:38:34.6"	0:00.08	0	0	4	mV	-0.313

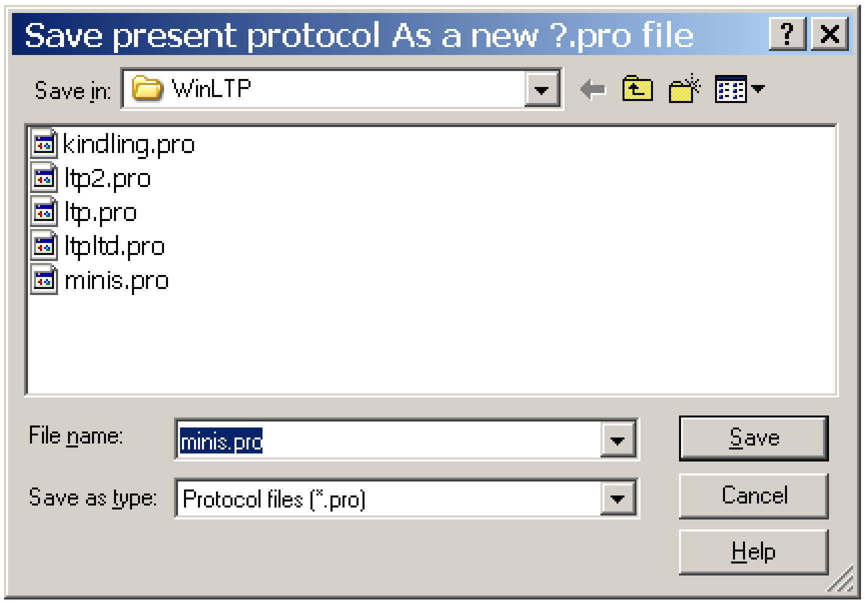
Fig 4.11.5.1. Analysis of every EPSP using the baseline of the first pulse.

4.12 Saving the Protocol File to Disk

After changing any of the values in the program that determine the type of protocol, the stimulation or acquisition parameters, the graph axes values, etc., these values can then be saved to disk by using the menu commands:

File -> Save As

To open the Protocol File SaveAs Dialog Box (Fig. 4.12.1). If the protocol file is to be saved under a different name, and then enter the filename - any filename ending in pro, such as "sigavg.pro".



Save in: WinLTP

kindling.pro
ltp2.pro
ltp.pro
ltpltd.pro
minis.pro

File name: minis.pro

Save as type: Protocol files (*.pro)

Fig. 4.12.1. Protocol File SaveAs Dialog Box.

If the file is just to be updated, save it by using the menu commands:

File -> Save

Updating the protocol file is very useful in resuming the experiment where you left off if a crash or power outage has occurred (see Section 4.18).

4.13 Run the Experiment

4.13.1 Evoking Single Sweeps when Main Protocol is Not Running

Click on the '**Single P0**', '(Single) **P1**', '**Single T0**' or '(Single) **T1**' Run Buttons (see Fig. 3.1.5.1).

Alternatively, use the Run Menu to evoke the single sweeps when Main Protocol is not running (see Fig. 3.2.6)

Run -> Single P0 Sweep Stimulation	**F5**
Run -> Single P1 Sweep Stimulation	**F6**
Run -> Single T0 Sweep Stimulation	**Ctrl+F9**
Run -> Single T1 Sweep Stimulation	**Ctrl+F10**

Or finally, you can press the Run Function Keys to evoke the single sweeps when the Main Protocol is not running.

Press	To Run
F5	Single P0 Sweep Stimulation
F6	Single P1 Sweep Stimulation
Ctrl+F9	Single T0 Sweep Stimulation
Ctrl+F10	Single T1 Sweep Stimulation

Note that the 'Repeat P0', '(Repeat) P1', 'Repeat T0' and '(Repeat) T1' Run Buttons, Run Menu Item, and Function Keys are disabled until the Main Protocol starts because they cannot run without the Main Protocol running.

4.13.2 Running Continuous Acquisition when Main Protocol is Not Running

Click on the '**Cont**' Run Button (see Fig. 3.1.5.1) to start Continuous Acquisition without the Main Protocol running.

Alternatively, use the Run Menu to evoke the single sweeps when Main Protocol is not running (see Fig. 3.2.6)

Run -> Continuous Acquisition

There is no Run Function Key to press to start Continuous Acquisition when the Main Protocol is not running.

4.13.3. Running Capture Spontaneous Events when Main Protocol is Not Running

Capturing Spontaneous Events is not enabled in this version

4.13.4 Running and Stopping the Main Protocol

Click on the '**Main Protocol**' Run Button (see Fig. 3.1.5.1) to start the Main Protocol. Click on the '**Stop**' Main Protocol Run Button to stop the Main Protocol.

Alternatively, use the Run Menu to start the Main Protocol (see Fig. 3.2.6)

Run -> Main Protocol F1

and

Run -> Stop Main Protocol F4

to stop the Main Protocol.

Or finally, you can press the Run Function Keys to start and stop the Main Protocol.

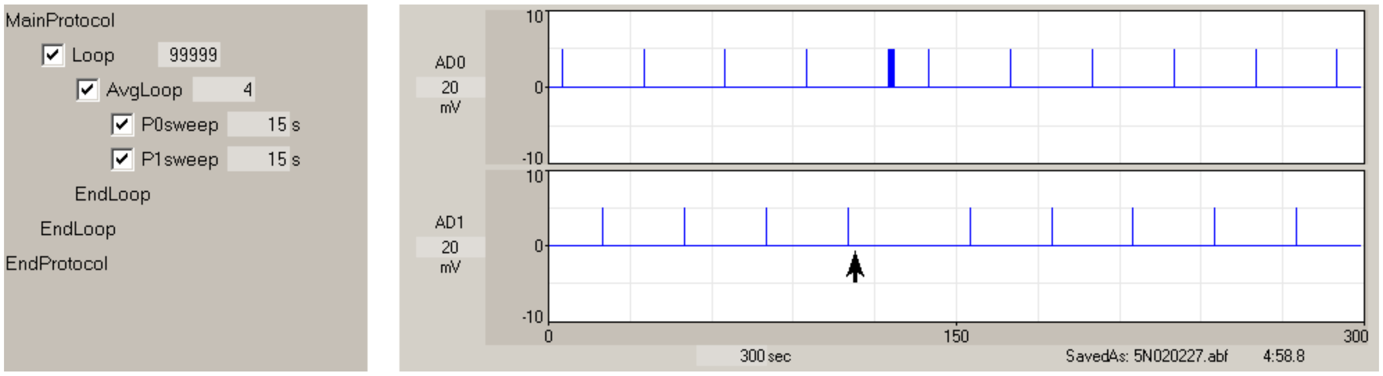
Press	To
F1	Start the Main Protocol
F4	Stop the Main Protocol

4.13.5. Evoking Single Sweeps when Main Protocol is Running

This is the same as discussed in Section 4.13.1.

When you are running a basic LTP experiment with alternating P0/P1 sweeps, evoking a single T0sweep containing an S0 train, the output looks like that in Fig. 4.13.5.1 (at arrow). Note that the T0/S0 train has run in place of the next P0/S0 single pulse and has delayed all P0 and P1 sweeps by the T0sweep period (which happens to be 15 sec, just like the P0 and P1 sweeps).

Note: beginning with WinLTP095, it is now impossible to evoke a Single Pulse P0 or P1 Sweep while in an Averaging Loop (which would disrupt the ongoing signal averaging).



```
MainProtocol
  [✓] Loop      99999
    [✓] AvgLoop     4
      [✓] P0sweep   15 s
      [✓] P1sweep   15 s
    EndLoop
  EndLoop
EndProtocol
```

Fig. 4.13.5.1. Standard LTP stimulus train induction when running a basic LTP experiment. The ‘Single T0sweep’ stimulation button was clicked at the arrow to produce an S0 train. Not that the P0 and P1 sweeps are delayed by the T0sweep period (15 seconds in this case). Setup: AD0 records S0 pulses, and AD1 records S1 pulses. P0sweep has S0 single pulse stimulation, P1sweep has S1 single pulse stimulation, and T0sweep has S0 train stimulation.

4.13.6. Evoking and Stopping Repetitive Sweeps when Main Protocol is Running (LTD and Theta Burst Stimulation)

To evoke Repeat Sweeps episode when Main Protocol is running, click on the ‘**Repeat P0**’, ‘(Repeat) **P1**’, ‘**Repeat T0**’ or ‘(Repeat) **T1**’ Run Button (see Fig. 3.1.5.1).

Alternatively, use the Run Menu to evoke the single sweeps when Main Protocol is running (see Fig. 3.2.6).

Run -> Repeat P0 Sweep Stimulation	**Ctrl+F5**
Run -> Repeat P1 Sweep Stimulation	**Cctrl+F6**
Run -> Repeat T0 Sweep Stimulation	
Run -> Repeat T1 Sweep Stimulation	

Or finally, you can press the Run Function Keys to evoke the single sweeps when the Main Protocol is not running.

Press	To Run
Ctrl+F5	Repetitive P0 Sweep Stimulation
Ctrl+F6	Repetitive P1 Sweep Stimulation

There are no Run Function Keys to run Repetitive T0 Sweep Stimulation and Repetitive T1 Sweep Stimulation.

The ‘Repeat P0’, ‘(Repeat) P1’, ‘Repeat T0’ and ‘(Repeat) T1’ Run Buttons, Run Menu Item, and Function Keys can be run when the Main Protocol is running.

Note: beginning with WinLTP095, it is now impossible to evoke a Repeat Pulse P0 or P1 Sweep while in an Averaging Loop (which would disrupt the ongoing signal averaging).

4.14 AutoCreate a new Data Folder (using CTL-F) while running an experiment

The Data Read/Write Folder is automatically created at the startup of WinLTP (Section 2.6). However, if you wish to change the Data Read/Write Folder during an experiment or before the next experiment (say for a different cell, etc), you can AutoCreate a new Data Folder. This can be accomplished by choosing "AutoCreate new data Folder" in the File menu (Fig. 4.14.1). Alternatively, you can quickly AutoCreate a new data folder 'on-the-fly' (e.g. while saving ADsweeps) just by pressing the CTL-F hotkey.

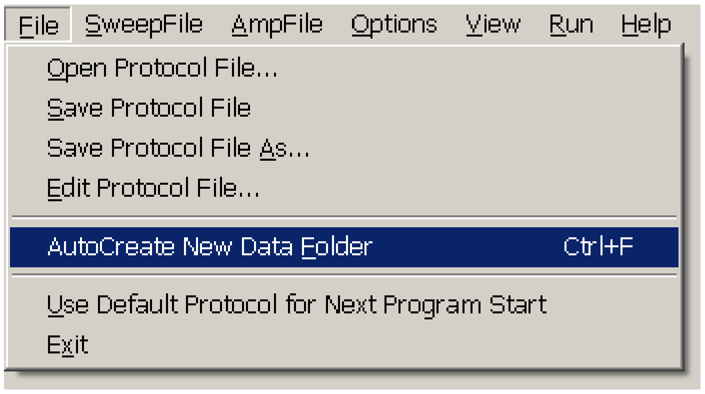
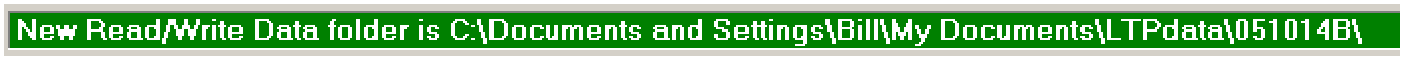


Fig 4.14.1. The File menu showing the "AutoCreate new data Folder CTL-F" selection.

When this is done, the data folder is changed from C:\WinLTP\051014\ to the newly created data folder C:\WinLTP\051014B (Fig. 4.14.2). Up to 26 data folders (e.g. up to C:\WinLTP\051014Z) can be AutoCreated in this manner.



```
New Read/Write Data folder is C:\Documents and Settings\Bill\My Documents\LTPdata\051014B\
```

Fig. 4.14.2. After AutoCreating new data Folder using CTL-F.

4.15 Saving AMP and XLS Analysis Files After an Experiment

The Spreadsheet data can be saved to either an ASCII AMP file, or an Excel XLS file, or both, in three ways:

1) By Menu command
 AmpFile -> Save Spreadsheet to ASCII and/or Excel AmpFile

2) When the Spreadsheet and Analysis Graphs are Cleared by the Menu command
 AmpFile -> Clear Analysis Graphs, Start New AmpFile

3) When WinLTP is exited

The Menu command

AmpFile -> Save Spreadsheet to ASCII and/or Excel AmpFile

saves the Spreadsheet no matter what, unless there is nothing in the Spreadsheet.

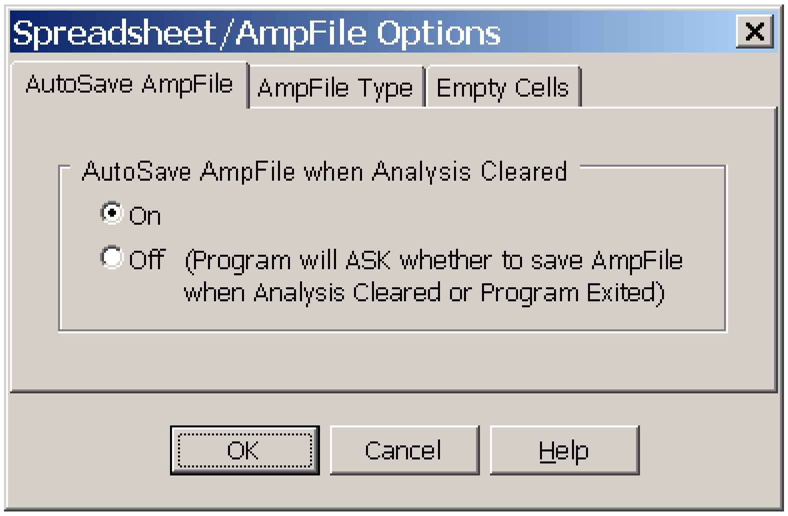
When the Spreadsheet and Analysis Graphs are Cleared by the Menu command

AmpFile -> Clear Analysis Graphs, Start New AmpFile,

the Spreadsheet data will be saved to an ASCII AMP file or an Excel XLS file if the **AutoSave AmpFile when Analysis** radiobutton is **On**. This **AutoSave AmpFile when Analysis** radiobutton is set by calling up the **Spreadsheet/Ampfile Options** dialog box by using the Menu command:

AmpFile -> Spreadsheet/Ampfile Options…

And then clicking on the **AutoSave AmpFile** tab (Fig. 4.15.1).



AutoSave AmpFile when Analysis Cleared

[✓ On]

Off (Program will ASK whether to save AmpFile when Analysis Cleared or Program Exited)

Fig. 4.15.1. The Spreadsheet/AmpFile Options dialog box with the AutoSave AmpFile tab chosen.

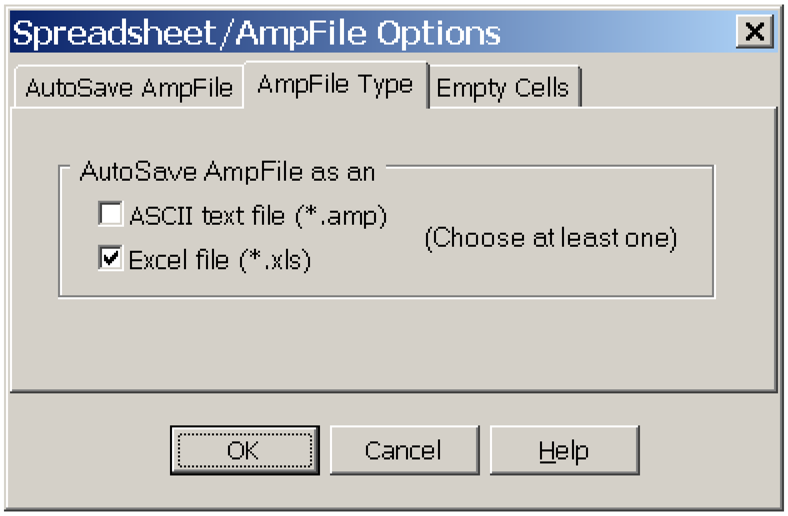
When the **AutoSave AmpFile when Analysis Cleared** is **Off,** the Spreadsheet data will **not** be saved if the Analysis Grahps and the Spreadsheet are cleared. However, when the **AutoSave AmpFile when Analysis Cleared** is **Off,** the WinLTP will **ask** whether you want the Spreadsheet data saved when you exit WinLTP.

See also **Automatically Clear Current Reanalysis at Start of Next Reanalysis** (Section 14.8).

When the Spreadsheet data is saved, it can either be saved to an ASCII text file (*.amp), an XLS file that can be loaded directly into an Excel spreadsheet, or both can be chosen simultaneously. This AmpFile Type is set by calling up the **Spreadsheet/Ampfile Options** dialog box by using the Menu command:

AmpFile -> Spreadsheet/Ampfile Options…

And then clicking on the **AmpFile Type** tab (Fig. 4.15.2). Then click the checkbox to save it to and **ASCII test file (*.amp)** and/or an **Excel file (*.xls)**.


Spreadsheet/AmpFile Options

AutoSave AmpFile | AmpFile Type | Empty Cells

AutoSave AmpFile as an

- [ ] ASCII text file (*.amp)
- [✓] Excel file (*.xls)

(Choose at least one)

OK | Cancel | Help

Fig. 4.15.2. The Spreadsheet/AmpFile Options dialog box with the AmpFile Type tab chosen.

4.15.1 AMP Text and XLS File Structure

When the Spreadsheet data is saved to an ASCII text AmpFile, the structure of an AMP text file will be similar to the example shown in Fig. 4.15.1.1. In this example the DC, PkAmp, Rs and Rm analyses were calculated, but the Lat, Dur, RiseTm, DecTm, Area, CoastLn, PSamp, PSlat, Slope and AvgAmp analyses were not, and column place was kept by putting in a double quote “”.

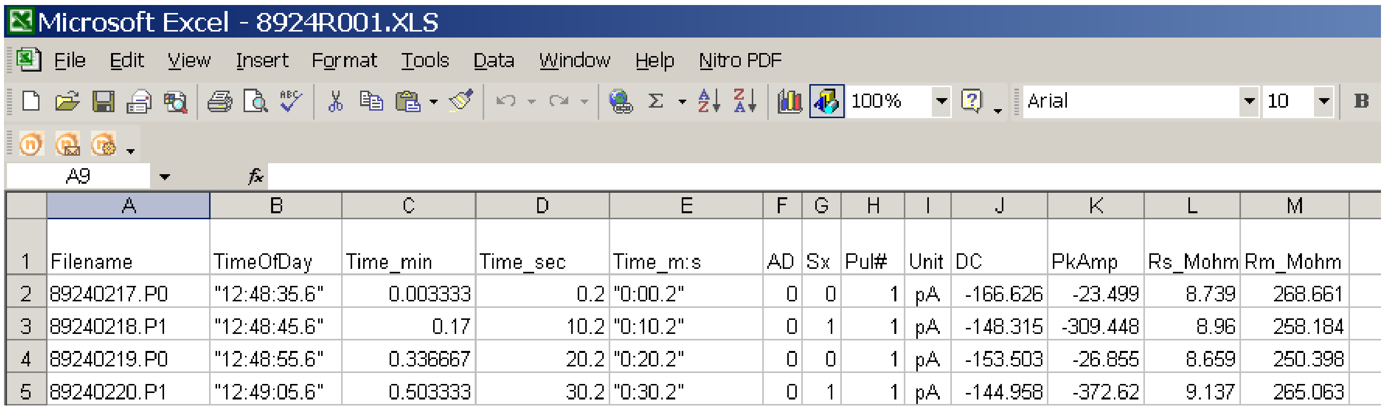
Note that DC, PkAmp, CoastLn, PSamp and AvgAmp are in Units (either pA here, or mV or V); Lat, Dur, RiseTm, DecTm and PSlat are in ms; Area is in unit*ms; Slope is in unit/ms; and Rs and Rm are either in Mohm (or Units of pA where they have to be converted to Mohms; see Sections 4.9.11 and 4.9.12).

Alternatively, the Spreadsheet data can be saved to an XLS file that can be directly loaded into Excel (Fig. 4.15.1.2). Since only the DC, PkAmp, Rs and Rm analyses were calculated in this example, only they are printed. The Lat, Dur, RiseTm, DecTm, Area, CoastLn, PSamp, PSlat, Slope and AvgAmp columns are not put into the XLS file.

```
8924R021.AMP Filename      TimeOfDay    Time_min     Time_sec     Time_m:s       AD Sx Pul# Unit  DC         PkAmp
1            89240217.P0   "12:48:35.6" 0.003333     0.2000       "0:00.2"       0  0  1     pA   -166.626   -23.499
2            89240218.P1   "12:48:45.6" 0.170000     10.2000      "0:10.2"       0  1  1     pA   -148.315   -309.448
3            89240219.P0   "12:48:55.6" 0.336667     20.2000      "0:20.2"       0  0  1     pA   -153.503   -26.855
4            89240220.P1   "12:49:05.6" 0.503333     30.2000      "0:30.2"       0  1  1     pA   -144.958   -372.620

Lat_ms  Dur_ms   RiseTm_ms DecTm_ms  Area_unit*ms CoastLn   PSamp   PSlat_ms  Slope_unit/ms AvgAmp  Rs_Mohm   Rm_Mohm
""      ""       ""        ""        ""           ""        ""      ""        ""            ""      8.739     268.661
""      ""       ""        ""        ""           ""        ""      ""        ""            ""      8.960     258.184
""      ""       ""        ""        ""           ""        ""      ""        ""            ""      8.659     250.398
""      ""       ""        ""        ""           ""        ""      ""        ""            ""      9.137     265.063
```

Fig. 4.15.1.1. The first four sweeps (taken 10 sec apart) of an Amp file ASCII text file. This is the full 24 columns wide AMP file, but it had to be separated into two sections to fit on this page. Only DC, PkAmp, Rs and Rm were calculated, the other values were not calculated and were set to “” (empty).


Microsoft Excel - 8924R001.XLS
File Edit View Insert Format Tools Data Window Help Nitro PDF
100%
Arial
10
A9

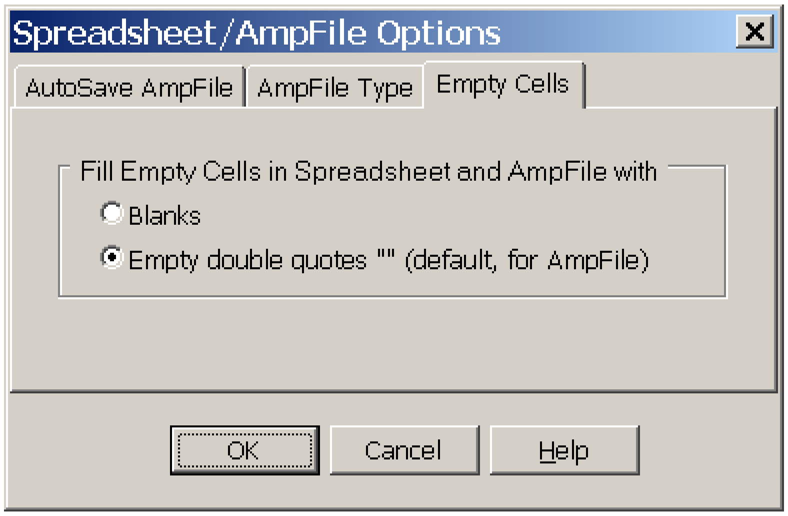
	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Filename	TimeOfDay	Time_min	Time_sec	Time_m:s	AD	Sx	Pul#	Unit	DC	PkAmp	Rs_Mohm	Rm_Mohm
2	89240217.P0	"12:48:35.6"	0.003333	0.2	"0:00.2"	0	0	1	pA	-166.626	-23.499	8.739	268.661
3	89240218.P1	"12:48:45.6"	0.17	10.2	"0:10.2"	0	1	1	pA	-148.315	-309.448	8.96	258.184
4	89240219.P0	"12:48:55.6"	0.336667	20.2	"0:20.2"	0	0	1	pA	-153.503	-26.855	8.659	250.398
5	89240220.P1	"12:49:05.6"	0.503333	30.2	"0:30.2"	0	1	1	pA	-144.958	-372.62	9.137	265.063

Fig. 4.15.1.2. The first four sweeps (taken 10 sec apart) of an XLS file viewed in Excel which is the same data saved in Fig. 4.15.1.1. Note that only the columns filled with data are saved to the XLS file (DC, PkAMp, Rs, Rm) and not the full 24 columns.

When saving data to the spreadsheet, the cells can either be blank, or filled with double quotes “”. For the ASCII AMP file it is important to have to double quotes to hold column position when printed from an ASCII test editor such as NotePad.exe, or when imported into Excel as an ASCII file. However, plan to save the Spreadsheet data as an Excel *.XLS file, you could choose to have the empty cells contain Blanks, not double quotes by choosing Blanks in the Empty Cells tabsheet. This Empty Cells tab is set by calling up the **Spreadsheet/Ampfile Options** dialog box by using the Menu command:

AmpFile -> Spreadsheet/Ampfile Options…

And then clicking on the **Empty Cells** tab (Fig. 4.15.1.3). Then click either Blanks or Empty double quotes (which is the default).



Fill Empty Cells in Spreadsheet and AmpFile with
- ○ Blanks
- ● Empty double quotes "" (default, for AmpFile)

Fig. 4.15.1.3. The Spreadsheet/AmpFile Options dialog box with the Empty Cells tab chosen.

4.16 Getting a quick printout of your Analysis Graphs using a PrintScreen program

The PrintSceen capability in Windows is rather crude. The whole screen must be printed, not just the important section of it that you are interested in. However, certain PrintScreen programs such as the **Gadwin PrintScreen** program (freely downloadable from www.gadwin.com) can make quick printing of your Analysis Graphs extremely easy.

With Gadwin PrintScreen:

1) When the 'PrtScn' button is pressed, a cursor appears on the screen, and you manually select with the mouse the subsection of the screen containing the Analysis Graphs you want to print. Press enter to capture this screen subsection.
2) Then you can either:
    a) direct the output to the Clipboard where you can then insert it into an imaging program like Paint (a fairly laborious procedure), and/or you can
    b) direct the output to a printer and immediately print out the Analysis Graphs screen subsection (fast if you have an online or networked printer), and/or you can
    c) direct it to an automatically named *.TIF file for later printing.

I think that Gadwin PrintScreen program is a brilliant piece of software that does the job brilliantly! You can also support Gadwin Systems by purchasing a Professional version if you wish to further annotate your printscreens.

4.17 Compress Data Files at the End of an Experiment?

There are no longer any strong reasons to zip data your data files at the end of an experiment. Although in the old LTP Program it may have made sense to zip up your data, this was primarily because the many ASCII ADsweep files saved in an experiment was an inefficient use of disk space – causing at least a **10-fold** decrease in saving efficiency.

However, with 1000GB hard disks and cheap 4.7 GB DVDs, there is no longer any reason to zip up your ADsweep files.

In contrast with Windows 98, Windows 2000, XP, Vista and 7 can easily take at least 10,000 ADsweep files per folder without any perceptible decrease in Read/Write speed.

WinLTP will now also directly reanalyze data directly from the read-only CDs and DVDs (while writing the new *.AMP, *.XLS and reaveraged/filtered/blanked ADsweep files to the hard disk). If these CDs and DVDs contained zipped files, they would first have to be copied to the hard disk, unzipped, and then reanalyzed.

Also, WinLTP automatically creates a new data folder at the beginning of each experiment, and it is easy to create additional directories 'on-the-fly' during the experiment. This makes data organization much easier.

The only key to backing up data is to always have your data **saved in at least two permanent places**.

4.18 Recovering from a Non-Fatal Bug or a Crash

4.18.1 Please Write Down All Error Messages Verbatim – and Email Me

If a non-fatal or fatal bug occurs in WinLTP, please write down **verbatim** any error messages (include all correct punctuation and upper/lower cases) and any other possibly useful information that might indicate how the program non-fatal bug or crash occurred.

The error messages are of two types:
1) An Internal Error message written by me when code has been reached that should not have been reached. They start with “INTERR: and then my message”
2) An error message generated by the Borland compiler

Please email Bill Anderson (support@winltp.com) about the bug so that I can fix it. It you don't email me about the bug and just complain to your co-workers about the bugs in WinLTP, then that bug won't get fixed!

4.18.2 Recovering from a Non-Fatal Bug

A non-fatal bug is a problem that causes faulty operation of WinLTP, for instance, incorrect showing of panels, graphs or field, but does not cause WinLTP to hang-up or crash. If a non-fatal bug occurs in WinLTP, there are several steps to restore correct operation, in order severity.

1) Try reloading the current (last saved) protocol file using the menu command **File -> Open.**
2) Exit WinLTP using the menu command **File -> Exit** and then restart WinLTP and automatically use the last saved protocol file.
3) Use the menu command **File -> Use Default protocol for Next Program Start** (to start WinLTP in default protocol mode when restarted), exit WinLTP using **File -> Exit**, then restart WinLTP in default protocol mode, and finally remake the protocol file. (This assumes that the protocol file has been is corrupted.)

Note that when there is no protacol.ini file, WinLTP starts with a 'clean slate' default protocol values only. All that **File -> Use Default Protocol for Next Program Start** actually does is to easily delete protacol.ini from the \WinLTP folder.

4.18.3 Recovering from a Crash

If a fatal bug has caused WinLTP to hang-up or crash, there are several steps to correct the situation, in order of severity.
1) Exit WinLTP using the **Windows Task Manager** (Section 4.18.4), and then restart WinLTP automatically using the last saved protocol file.
2) Exit program using the **Windows Task Manager**, then delete the protacol.ini in \WinLTP folder, then restart WinLTP, and finally remake protocol file. (This assumes that the protocol file has been corrupted.)

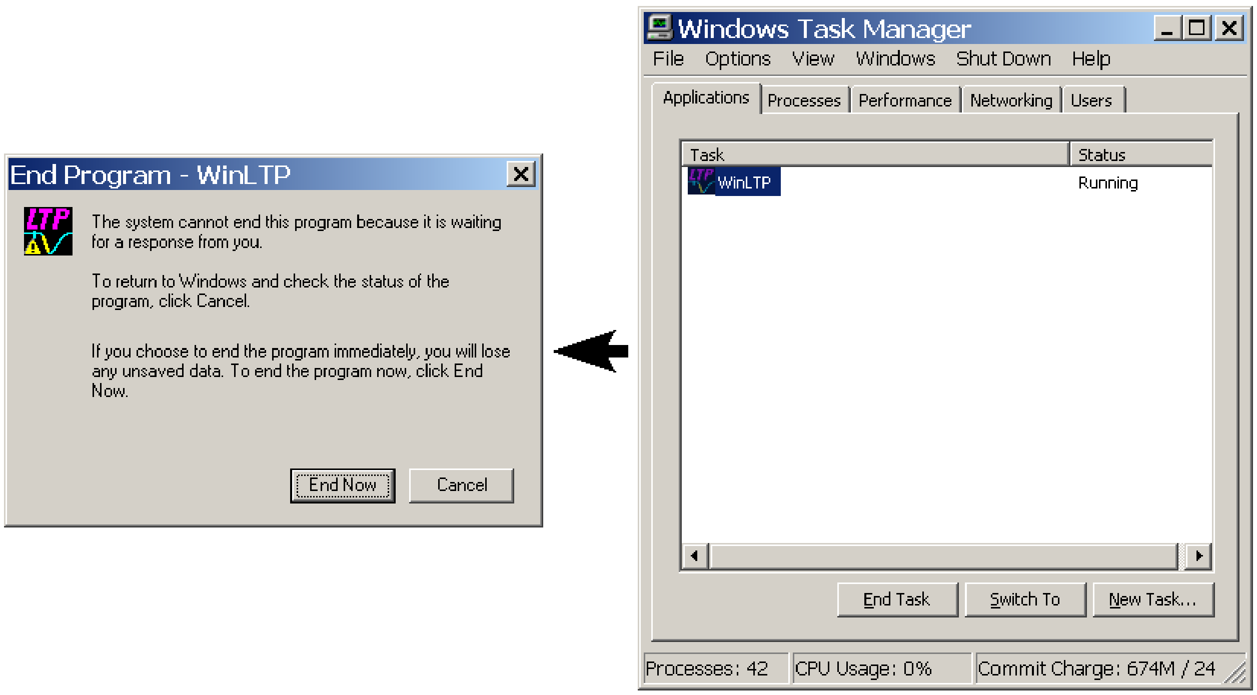
4.18.4 How to Exit WinLTP using the Window Task Manager

After a program bug occurs, WinLTP can sometimes be exited just by using the menu command
File -> Exit

However, if the program has truly crashed and doesn't respond to keyboard or mouse input, press **Ctrl-Alt-Del** to bring up **Windows Task Manager**, then click on the **Applications** tab (Fig 4.18.4.1).

End WinLTP by clicking on the "End Task" button. Then when the **End Program** dialog box comes up, click on the **"EndNow"** button. **Repeat** this until WinLTP is removed as a task.

And if that doesn't work!, click on the **Processes** tab, right mouse click on the "**WinLTPm111.exe**" filename to bring up the pop-up menu, and left mouse click on the **End Process Tree** menu item. That definitely will terminate the WinLTP program – no questions asked!!!


End Program - WinLTP
End Now
Cancel
Windows Task Manager
File Options View Windows Shut Down Help
Applications Processes Performance Networking Users
End Task
Switch To
New Task...

The system cannot end this program because it is waiting for a response from you.

To return to Windows and check the status of the program, click Cancel.

If you choose to end the program immediately, you will lose any unsaved data. To end the program now, click End Now.

Task	Status
WinLTP	Running

Processes: 42 CPU Usage: 0% Commit Charge: 674M / 24

Fig. 4.18.4.1. Using the Windows Task Manager to exit WinLTP that had crashed. Highlight the program to be terminated (WinLTP) in the Windows Task Manager dialog box and click on the "End Task" button. Then click on the "End Now" button in the End Program dialog box. Repeat this until WinLTP is terminated.

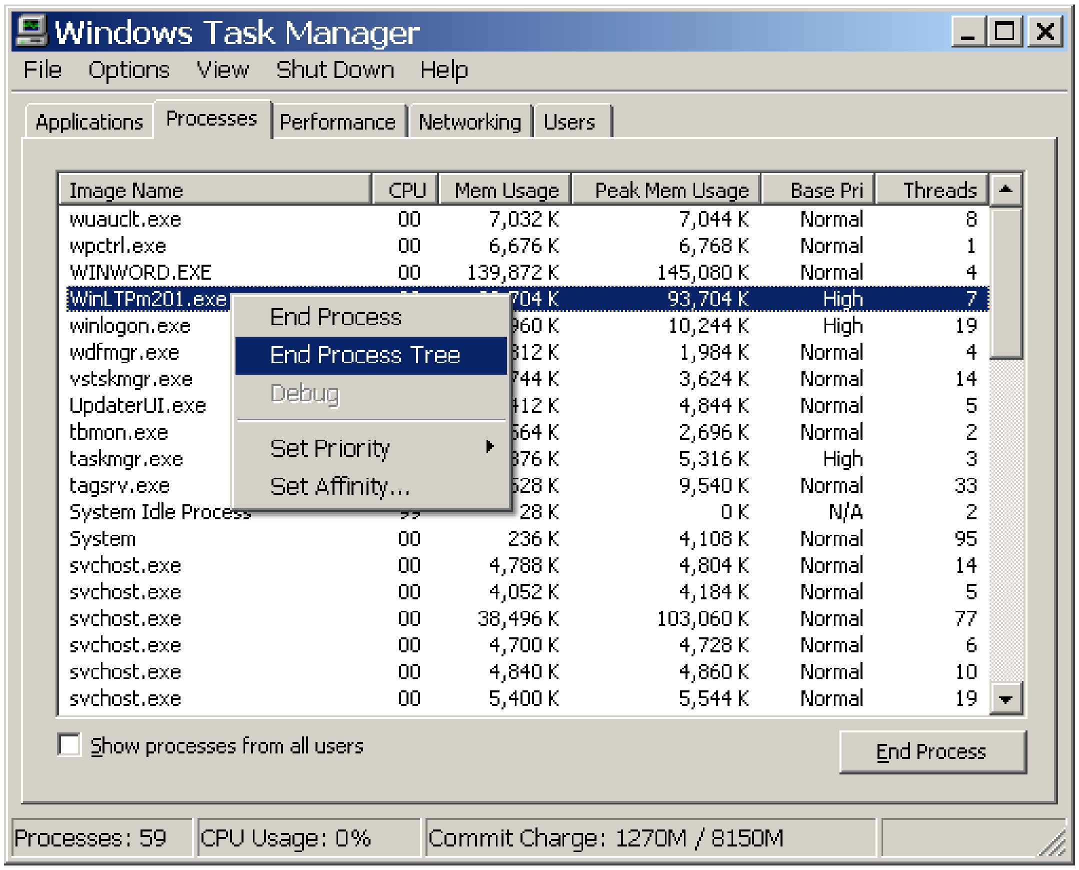

Windows Task Manager
File Options View Shut Down Help
Applications Processes Performance Networking Users
End Process
End Process Tree
Debug
Set Priority
Set Affinity...
Show processes from all users
End Process

Image Name	CPU	Mem Usage	Peak Mem Usage	Base Pri	Threads
wuauclt.exe	00	7,032 K	7,044 K	Normal	8
wpctrl.exe	00	6,676 K	6,768 K	Normal	1
WINWORD.EXE	00	139,872 K	145,080 K	Normal	4
WinLTPm201.exe		704 K	93,704 K	High	7
winlogon.exe		960 K	10,244 K	High	19
wdfmgr.exe		812 K	1,984 K	Normal	4
vstskmgr.exe		744 K	3,624 K	Normal	14
UpdaterUI.exe		412 K	4,844 K	Normal	5
tbmon.exe		564 K	2,696 K	Normal	2
taskmgr.exe		376 K	5,316 K	High	3
tagsrv.exe		528 K	9,540 K	Normal	33
System Idle Process	99	28 K	0 K	N/A	2
System	00	236 K	4,108 K	Normal	95
svchost.exe	00	4,788 K	4,804 K	Normal	14
svchost.exe	00	4,052 K	4,184 K	Normal	5
svchost.exe	00	38,496 K	103,060 K	Normal	77
svchost.exe	00	4,700 K	4,728 K	Normal	6
svchost.exe	00	4,840 K	4,860 K	Normal	10
svchost.exe	00	5,400 K	5,544 K	Normal	19

Processes: 59 CPU Usage: 0% Commit Charge: 1270M / 8150M

Fig. 4.18.4.2. A more drastic way to definitely end the WinLTPm201.exe program by right mouse clicking on the "WinLTPm201" filename to bring the pop-up menu up, and then left mouse click on the End Process tree menu item.

4.18.5 Data will not be Lost

This program is designed to recover 'gracefully' from a crash or a power failure. (Whether your preparation recovers from a power failure is another story). When sweep data is written to a file, the ADsweep file is opened, immediately written to, and then immediately closed. Sweep data files are **not** left open until the program is exited. Therefore, only the latest ADsweep file could be corrupted if it was being written to when the failure occurred. All the other ADsweep files will be fine.

The Spreadsheet data is only saved to the *.AMP and *.XLS files when manually requested by calling

AmpFile -> Save Spreadsheet to ASCII and/or Excel AmpFile

or when a new Spreadsheet is started, or when the program is exited. Therefore, if the program crashes, the data in the Spreadsheet that has not been save will be temporarily lost. However, since all the ADsweep files have be saved (except for possibly the last one), the Spreadsheet data can be reconstructed from reanalysis of the saved ADsweep files, and then saved to an *.AMP or *.XLS file.

4.18.6 What Happens when WinLTP is Restarted

After a crash or power failure, restart the program as normal. WinLTP will automatically load the last protocol file saved. Therefore, it is very useful to save the most recent field and dialog box values every time you change them by resaving the present protocol file (by using **File -> Save** as described in Section 4.12). Remember, these will be the values in the program when the program is restarted after a crash.

When WinLTP restarts it checks the last ADsweep files (*.T0, *.T1, *.P0, *.P1, *.AP0, *.AP1), Amplitude files (*.AMP and *.XLS files) saved. When saving the next ADsweep or Amplitude file, it will check to see if it already exists, and if so **it will not write over this file**, but will instead increment the file number and try writing again until a file number is found where the file doesn't exist, and then it will be saved.

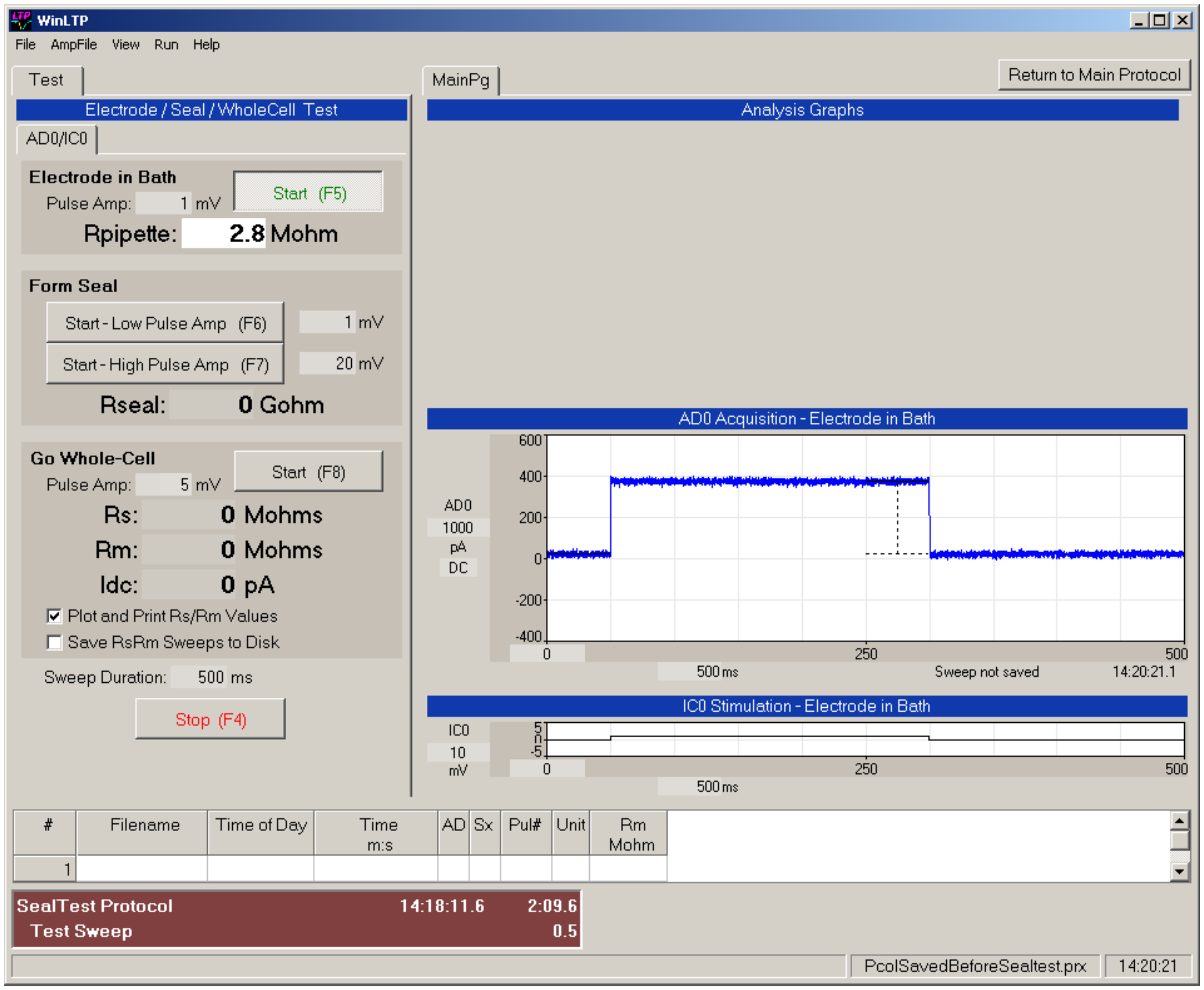
CHAPTER 5 – Patch-Clamp Sealtest Protocol

A patch-clamp Sealtest Protocol has been implemented for making easier patch-clamp gigohm seal whole-cell recordings, in particular reading out pipette resistance (Rpipette) when the electrode is in the bath, seal resistance (Rseal), when forming the seal, and series resistance (Rs), the input resistance (Rm) and the steady state DC current (Idc) when going/gone whole-cell.

With WinLTP in the Normal Mode, go to Patch-Clamp SealTest Mode by clicking on the "SealTest AD0" or "AD1" button. In order for these SealTest buttons to appear, AD0 and/or AD1 **datatype must be in "pA"** (set by the Edit Protocol dialog box, see Fig. 2.12.1.1).

5.1 Electrode in Bath

Put the patch electrode in the bath. To record the patch electrode or pipette resistance in the bath, click the Electrode in Bath "Start (F5)" button (or press F5) in the Patch-Clamp SealTest Mode / Electrode in



Electrode in Bath
Pulse Amp: 1 mV
Rpipette: 2.8 Mohm

Form Seal
1 mV
20 mV
Rseal: 0 Gohm

Go Whole-Cell
Pulse Amp: 5 mV
Rs: 0 Mohms
Rm: 0 Mohms
Idc: 0 pA

☑ Plot and Print Rs/Rm Values
☐ Save RsRm Sweeps to Disk

Sweep Duration: 500 ms

Sweep not saved 14:20:21.1

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	Rm Mohm
1								

SealTest Protocol 14:18:11.6 2:09.6
Test Sweep 0.5

PcolSavedBeforeSealtest.prx 14:20:21

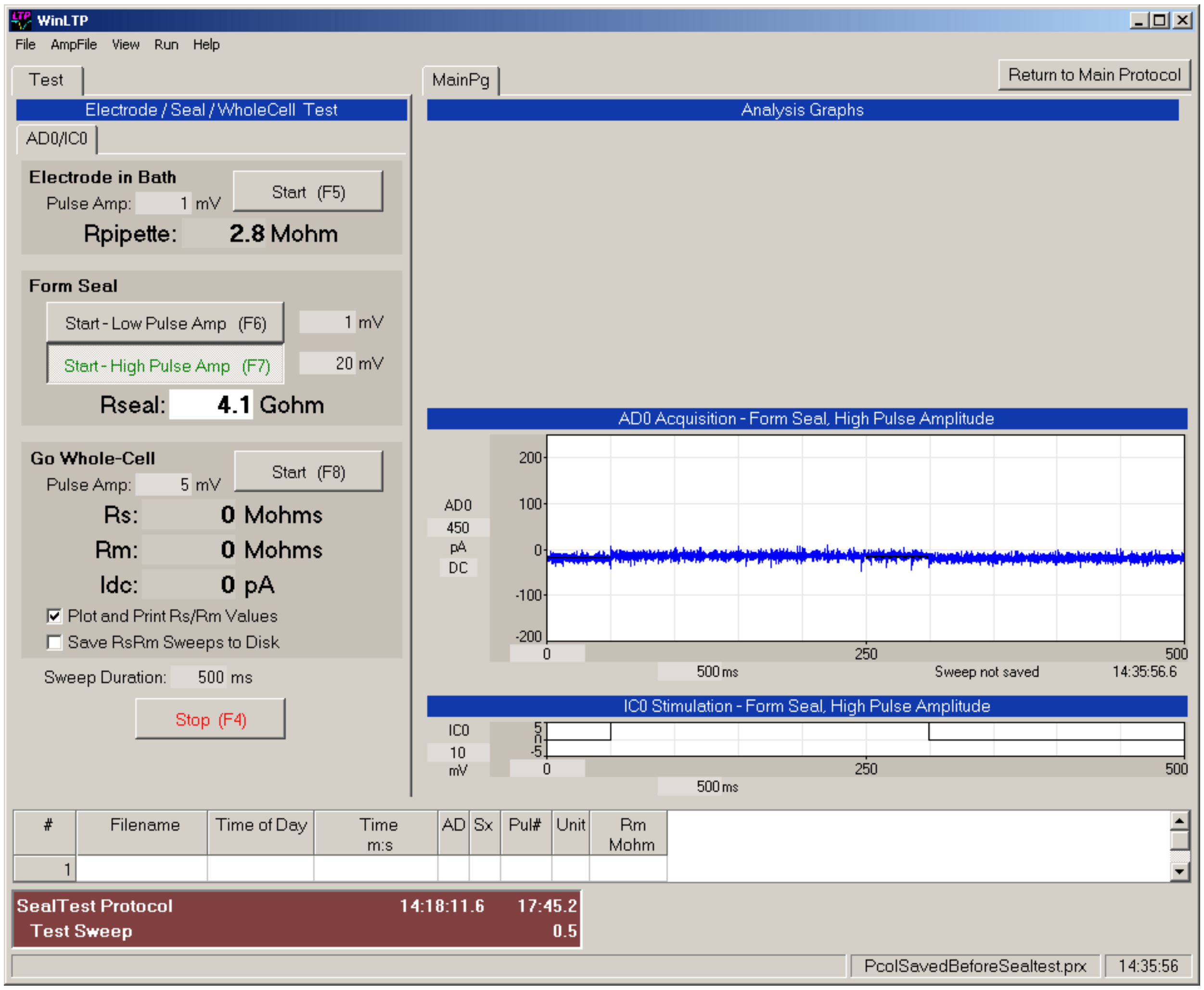
Fig. 5.1.1. Patch-Clamp Sealtest Protocol / Electrode in Bath after the patch pipette or electrode has been placed in bath and the “Start (F5)” button (or F5) has been clicked. Rpipette reads out the electrode resistance in Mohms.

Bath section (Fig. 5.1.1, top left). The electrode or pipette resistance is read out as Rpipette in Mohms. Set the appropriate test pulse amplitude by changing the value in the Pulse Amp edit field.

The Sweep Duration can be set between 10 and 1000 msec, and repeat between 10 Hz and 1 Hz (Fig. 5.1.1, bottom left). The test stimulation can be stopped by pressing the "Stop (F4)" button (or pressing F4) (Fig. 5.1.1, bottom left).

5.2 Forming a Seal

Once the electrode is on the cell surface, click the "Start - Low Pulse Amp (F6)" button (or press F6) to output the low pulse amplitude pulse to approximately measure the seal resistance, Rseal, in the Patch-Clamp SealTest Mode / Form Seal section (Fig. 5.2.1, middle left). Once the seal is coming into the Gohm range, click the "Start - High Pulse Amp (F7)" button (or press F7) to output the high pulse amplitude to accurately measure the seal resistance, Rseal. Set the appropriate low and high pulse amplitudes in the neighboring pulse amplitude edit fields.


WinLTP
File
AmpFile
View
Run
Help
Test
MainPg
Return to Main Protocol
Electrode / Seal / WholeCell Test
Analysis Graphs
AD0/IC0
Start (F5)
Start - Low Pulse Amp (F6)
Start - High Pulse Amp (F7)
Start (F8)
Stop (F4)
AD0 Acquisition - Form Seal, High Pulse Amplitude
AD0
450
pA
DC
200
100
0
-100
-200
0
250
500
500 ms
IC0 Stimulation - Form Seal, High Pulse Amplitude
IC0
10
mV
5
0
-5
0
250
500
500 ms

Electrode in Bath
Pulse Amp: 1 mV
Rpipette: 2.8 Mohm

Form Seal
1 mV
20 mV
Rseal: 4.1 Gohm

Go Whole-Cell
Pulse Amp: 5 mV
Rs: 0 Mohms
Rm: 0 Mohms
Idc: 0 pA
☑ Plot and Print Rs/Rm Values
☐ Save RsRm Sweeps to Disk
Sweep Duration: 500 ms

Sweep not saved 14:35:56.6

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	Rm Mohm
1								

SealTest Protocol 14:18:11.6 17:45.2
Test Sweep 0.5

PcolSavedBeforeSealtest.prx 14:35:56

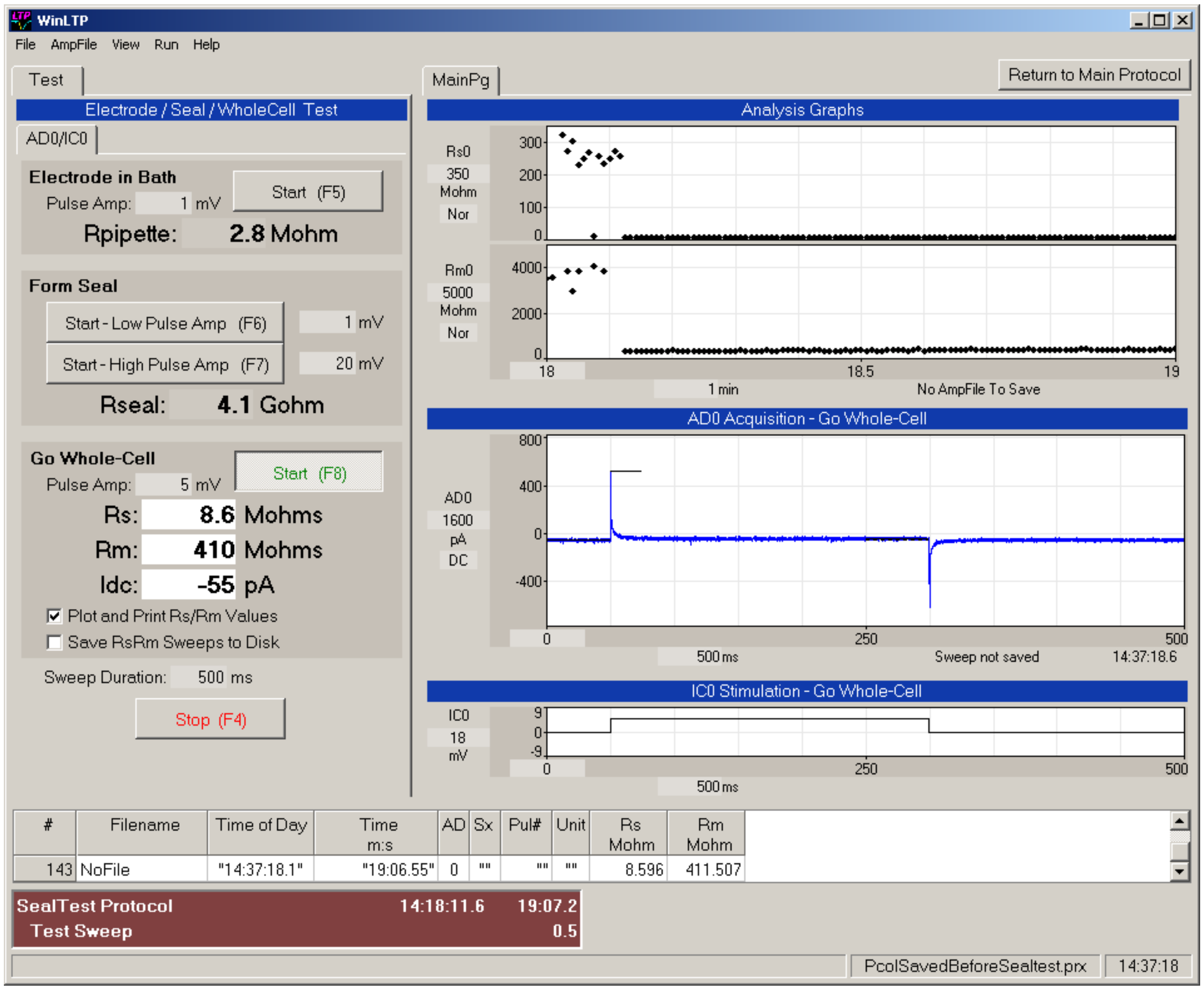
Fig. 5.2.1. Patch-Clamp Sealtest Protocol / Form Seal after a gigaohm seal has been formed and the "Start – High Pulse Amp (F7)" button (or F7) has been clicked. Rseal reads out the seal resistance in Gohms.

5.3 Going Whole Cell

After the gigaohm seal has been formed and you are ready to go whole cell, click the Go Whole Cell "Start (F8) " button (or press F8) to record the seal resistance (Rs, in Mohms), the input resistance (Rm, in Mohms) and the steady state DC current (Idc, in pA) in the Patch-Clamp SealTest Mode / Go Whole-Cell section (Fig. 5.3.1, bottom left). Set the appropriate pulse amplitude in the Pulse Amp edit field.

Also when going whole cell, you can plot the Rs and Rm values and them the spreadsheet by checking the “Plot and Print Rs/Rm Values” checkbox, and you can save the Rs/Rm sweep files to disk by checking the “Save RsRm Sweeps to Disk” checkbox. Note that in Fig. 5.3.1, the “Plot and Print Rs/Rm Values” checkbox is checked, and so Rs and Rm are plotted before and after going whole-cell (see Fig. 5.3.1, top right), and the Rs and Rm values are put into the spreadsheet (see Fig. 5.3.1, bottom).

Return to the Normal Mode Protocol by clicking "Return to Protocol" button.



Electrode in Bath
Pulse Amp: 1 mV
Rpipette: 2.8 Mohm

Form Seal
1 mV
20 mV
Rseal: 4.1 Gohm

Go Whole-Cell
Pulse Amp: 5 mV
Rs: 8.6 Mohms
Rm: 410 Mohms
Idc: -55 pA
[✓] Plot and Print Rs/Rm Values
[ ] Save RsRm Sweeps to Disk
Sweep Duration: 500 ms

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	Rs Mohm	Rm Mohm
143	NoFile	"14:37:18.1"	"19:06.55"	0	""	""	""	8.596	411.507

SealTest Protocol 14:18:11.6 19:07.2
Test Sweep 0.5

Fig. 5.3.1. Patch-Clamp Sealtest Protocol / Going Whole-Cell. When going from on-cell gigaohm seal to a whole-cell recording click the “Start – High Pulse Amp (F7)” button (or press F7) to read out the series resistance (Rs, in Mohms), the cell input resistance (Rm, in Mohms) and the steady-state holding current (Idc, in pA).

CHAPTER 6 – Protocol Builder – Circular Scripting

The Protocol Builder enables complicated protocols to be built using **‘building blocks’** pulled down from **User Interface buttons**. These building blocks include **Loops, Delays, Runs and Sweeps** (with various stimulations).

WinLTP protocol flow of execution can be easily changed during runtime by checking and unchecking the check boxes beside the Run, Loop, Delay and Sweep lines, and by changing the number of loops. Furthermore, Sweep and Delay Periods can be changed during runtime, as can all sweep stimulation values.

There has been some confusion and hesitancy about the use of the ‘scripting’ in the Protocol Builder. The Protocol Builder works by pulling down ‘building blocks’ from the User Interface Insert buttons, and then certain check boxes and numerical fields in the Loop, Delay, Run and Sweeps lines can be edited. This is somewhat similar to the Protocol Editor ‘scripting’ in HEKA’s PatchMaster. WinLTP’s ‘scripting’ is very different from another type of ‘scripting’ that involves using a text editor to essentially write out programming code and have it interpreted and then run, such as with Cambridge Electronic Design’s Signal.

Protocol Builder scripting has been touched on before concerning its capabilities in the Basic versus Advanced Modes (Section 2.11 and Fig. 2.11.1), and how to write simple scripts for basic LTP experiments with and without signal averaging which essentially involves clicking on an Initialize Button (Section 4.4 and Fig. 4.4.1.1).

Chapter 6 deals with circular scripting with the Protocol Builder in contrast to sequential scripting (Chapter 7). Circular scripting means that the outermost Loop in the MainProtocol has an extremely large number of loops (usually presented as 99999), and that the outermost loop will continue looping as long as the MainProtocol is running (Fig 6.1.1). The program flow is circular, always going back to the first line after the outermost loop. This means that stimulations (usually inclosed in the Run event) can be evoked at essentially any time. Circular scripting is valuable when you don’t know at the beginning of the experiment what stimulation you will need and when. It is best used for exploratory, ‘freewheeling’ experiments. Once you know when you will be stimulating and changing perfusion solutions, switching to sequential scripting is best because non-exploratory experiments are sequential.

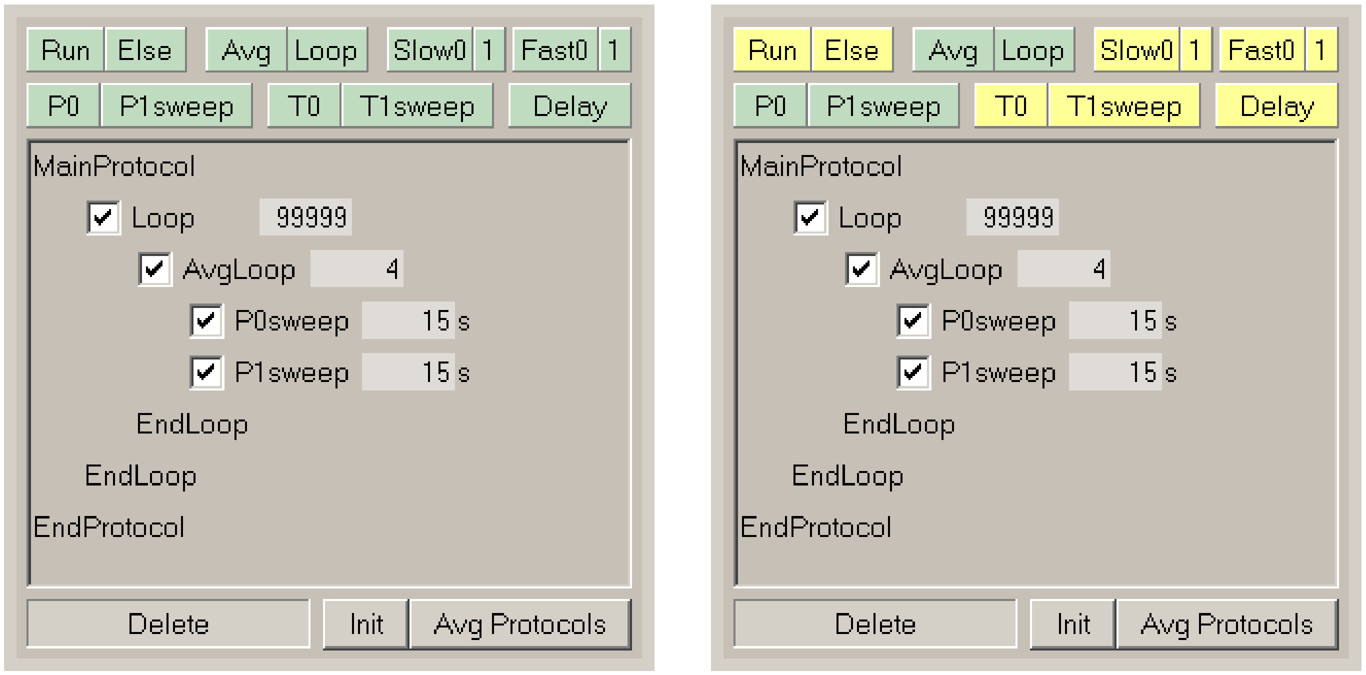
6.1 The Protocol Builder in Basic and Advanced Modes

As already discussed in Section 2.11, when you enter the WinLTP program for the first time in the DemoTrial period you are running in the Advanced Mode with a fully functioning Protocol builder (Fig. 6.1.1, left). In this mode you can write any number of advanced protocols including this automated perfusion protocol using ‘Slow0’ and ‘Slow1 Perfuse’ events. All the Protocol Builder events can be used and are shown in green, including the ‘Run’, ‘ElseRun’, ‘AvgLoop’, and ‘Loop’ events, the ‘Slow0’, ‘Slow1’, ‘Fast0’ and ‘Fast1’ Perfuse events, the P0, P1, T0 and T1sweep events, and the Delay event. If you have ordered an Advanced Mode license you will continue with the fully functional Protocol Builder (Fig. 6.1.1, left).

Alternatively, if you have not ordered an Advanced Mode license, at the end of the 2 month DemoTrial period you will automatically enter the Basic Mode partially functional Protocol Builder (Fig. 6.1.1, right). Only the green Insert Event buttons can be used and still save ADsweep data. These include ‘AvgLoop’, ‘Loop’, ‘P0sweep’ and P1sweep events. If you use the yellow Insert Event buttons, the ‘Run’, ‘ElseRun’, ‘Tosweep’ ‘T1sweep’ and ‘Delay’ events, **your protocol will run perfectly OK, except that the ADsweep data will not be saved. This allows you to easily test the Advanced Mode functions to see if it is worthwhile upgrading to the Advanced Version.**

In Chapter 6 we will focus on using the fully functional Protocol Builder of the DemoTrial and Advanced Mode.

You can ‘write’ a simple repetitive P0sweep protocol, by clicking the **‘Init’ Protocol button** (see Fig. 4.4.1.1, left top panel). Or you can ‘write’ a repetitive P0sweep with signal averaging protocol by clicking on the **Init ‘Avg Protocols’ button** (Fig. 6.1.1).



```
MainProtocol
    ☑ Loop        99999
        ☑ AvgLoop       4
            ☑ P0sweep     15 s
            ☑ P1sweep     15 s
        EndLoop
    EndLoop
EndProtocol
```

```
MainProtocol
    ☑ Loop        99999
        ☑ AvgLoop       4
            ☑ P0sweep     15 s
            ☑ P1sweep     15 s
        EndLoop
    EndLoop
EndProtocol
```

Fig 6.1.1. The Protocol Builder panel. The top green buttons are Insert buttons where Run, ElseRun, AvgLoop, Loop, Delay, P0sweep, P1sweep, T0sweep, and T1sweep lines can be dragged down into the script area. Drag the Run, RunElse, AvgLoop, Loop, Delay and Sweep lines to the Delete button to remove them from the protocol.

6.2 Inserting and Deleting Protocol Lines

To insert a Run, RunElse, AvgLoop, Loop, Delay, P0sweep, P1sweep, T0sweep, or T1sweep line, click on the appropriate **User Interface Insert button** by pressing the left mouse button and drag it into the script area between two exiting lines and release the left mouse button (see also Fig. 4.4.1.1, bottom panels).

To delete a Run, RunElse, AvgLoop, Loop, Delay, P0sweep, P1sweep, T0sweep, or T1sweep line, click on the line by pressing the left mouse button and drag it into the Delete button and release the left mouse button.

6.3 Rules of Protocol Building

There are a few rules on how the scripts operate.

1) If the large check box on the left is checked, the Run, Loop, AvgLoop, Delay and Sweep will be run; otherwise they wont.
   a) By unclicking the large check box on the Loop or AvgLoop, the Loop will run to the bottom and exit.
   b) Unclicking the large check box for the Run, Delay and Sweep will not cause them to be prematurely terminated but they will not be run next time through.

2) The edit field on the right is for NumberOfLoops for Loop and AvgLoop, and for Seconds for Sweep and Delay.

3) Changes to the Run/Loop/Sweep/Delay large check box, the NumberOfLoops edit field or the Sweeps/Delay Seconds edit field **must be made at least 5 seconds before the action is to take place for the Digidata 132x boards**, and at least **0.5 seconds** before with the National Instruments M or X-Series boards. This is because of the large 5 second output buffer in the Digidata 132x (3 seconds) and in WinLTP itself (2 seconds), and a 0.5 sec output buffer in the NI M or X-Series boards. For the Digidata boards, you have to plan ahead a bit.

4) Changing the NumberOfLoops edit field causes an **immediate** change in the number of loops that will be run. If the new number of loops is **greater** than the number of loops already run, the loops will continue up to the new number of loops. If the new number of loops is **less** than the number of loops already run, the loop structure will be exited after the last line in the loop has been run.

5) Changing the Seconds edit field for Sweeps and Delay **while in** the Sweeps/Delay period will **not cause an immediate** change in the Sweep or Delay period, but the change will occur when the **next** Sweep or Delay period is entered. If you want to immediately change the Sweep or Delay period, put a Delay Period in a Loop (see Fig. 6.4.2.1).

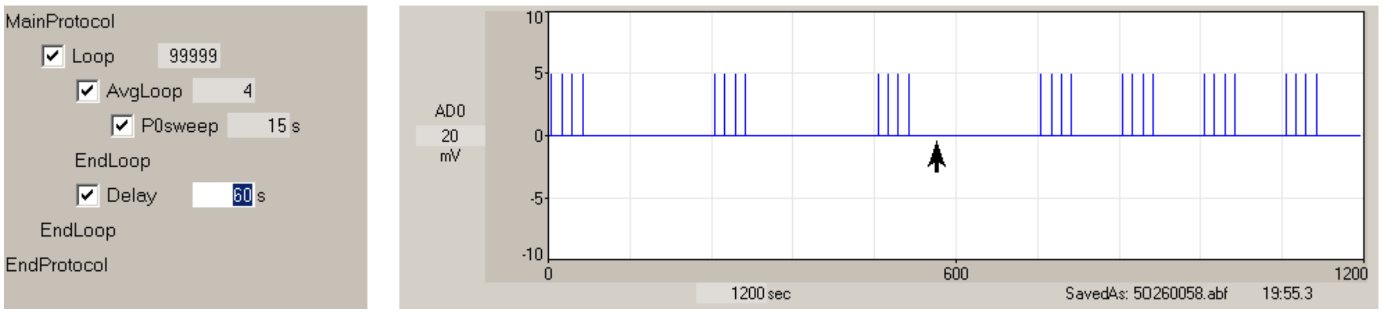
6.4 Examples of Circular Scripting

The figures in this section show some protocol building ideas.

In most examples AD0 records S0 output and AD1 records S1 output. (This is because S0 is DigitalOut0 and is plugged into AnalogIn0, S1 is DigitalOut1 and is plugged into AnalogIn1. P0sweep usually has S0 single pulse stimulation, P1sweep usually has S1 single pulse stimulation, T0sweep usually as S0 train stimulation, and T1sweep usually as S1 train stimulation.)

6.4.1 Minimal LTP Stimulation Using a Delay

Fig. 6.4.1.1 is the Reymann Magdeburg protocol for minimal LTP stimulation simply adding a Delay to the normal LTP AvgLoop. You can change the delay time 5 sec (or 0.5 sec) or more before it starts (see arrow).

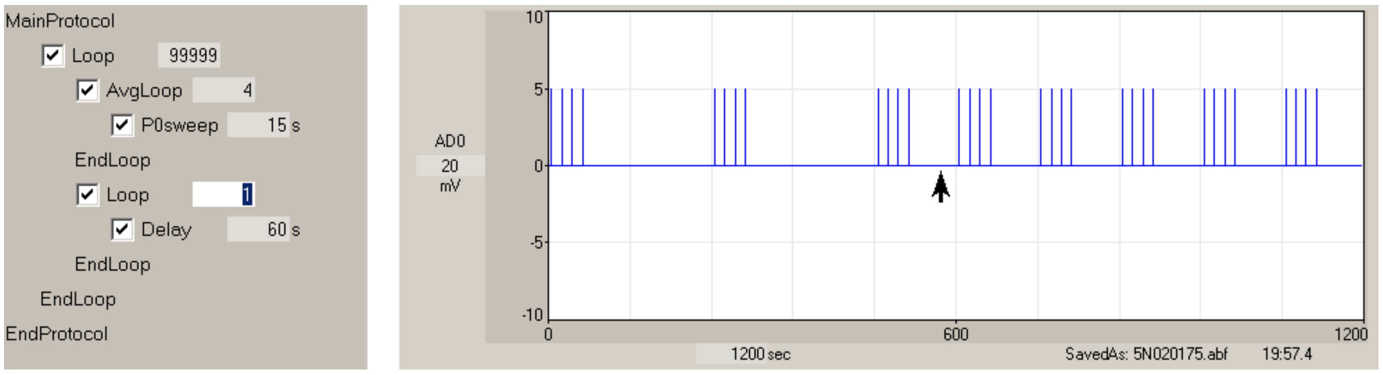


```
MainProtocol
   ☑ Loop      99999
      ☑ AvgLoop     4
         ☑ P0sweep   15 s
      EndLoop
      ☑ Delay      60 s
   EndLoop
EndProtocol
```

Fig. 6.4.1.1. Minimal LTP stimulation protocol using the Delay line. The Delay value was changed from 180 to 60 sec at arrow, but it only changed for the next Delay period.

6.4.2 Immediately Change Delay and Sweep Period by Putting a Delay in a Loop

If you want to change the Delay (or Sweep) Period while you are in it, just put a smaller Delay time in a Loop and change the number of loops (Fig. 6.4.2.1).



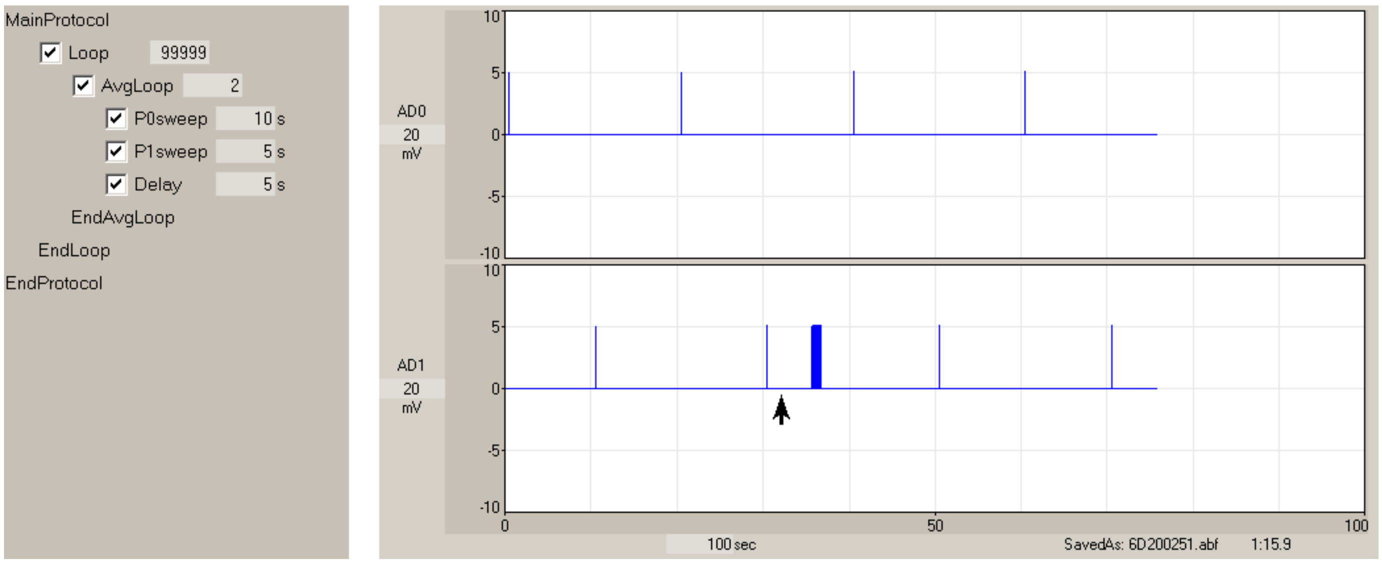
```
MainProtocol
   ☑ Loop      99999
      ☑ AvgLoop     4
         ☑ P0sweep   15 s
      EndLoop
      ☑ Loop        1
         ☑ Delay     60 s
      EndLoop
   EndLoop
EndProtocol
```

Fig. 6.4.2.1. Minimal LTP stimulation protocol using a Delay line within a Loop. This allows the Delay to be almost immediately changed rather than waiting for the next Delay. At the arrow, the Loop value was changed from 3 loops to 1 loop, effectively immediately decreasing the Delay to 60 seconds.

6.4.3 Evoking Sweeps at the Beginning of a Delay Period

In WinLTP095 we added the capability that Single or Repeating Evoked Sweeps could be run at the beginning of a Delay Period, but using the delay's period (Fig. 6.4.3.1, 6.4.3.2).

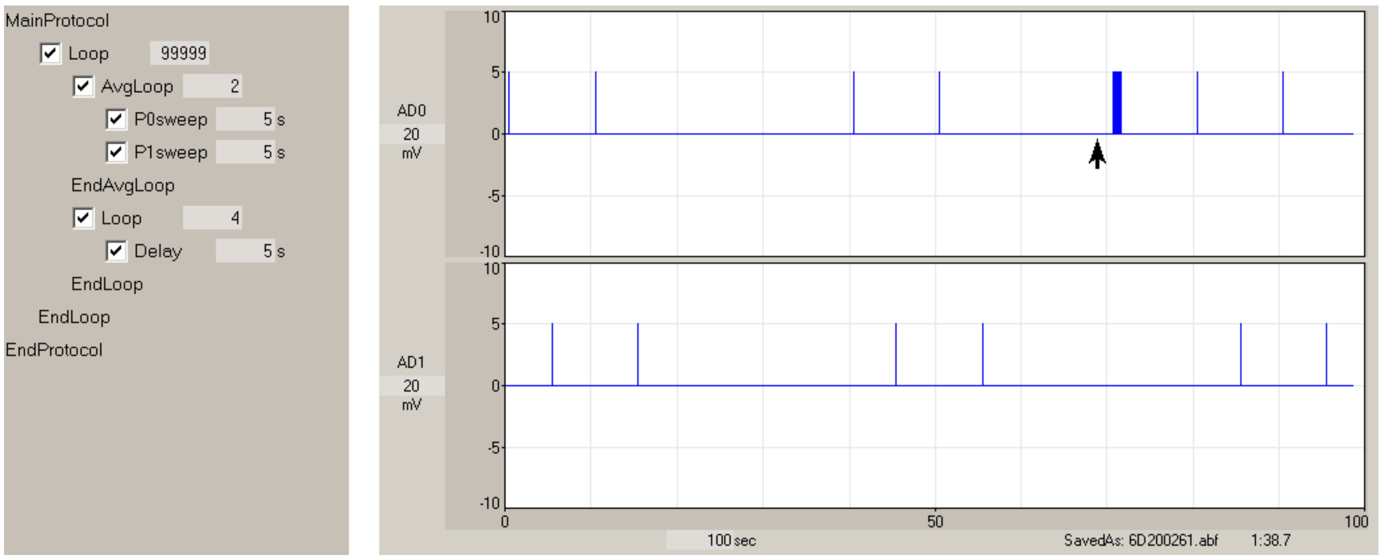
However, if an Single or Repeating Evoked Sweeps is of longer duration than the delay's period, the period will be extended by a multiple of the Delay Period until sufficient to contain all the Single or Repeating Evoked sweeps.



```
MainProtocol
    ☑ Loop      99999
        ☑ AvgLoop       2
            ☑ P0sweep     10 s
            ☑ P1sweep      5 s
            ☑ Delay        5 s
        EndAvgLoop
    EndLoop
EndProtocol
```

Fig. 6.4.3.1. Evoked sweep can occur at the beginning of a Delay Period. Evoking a single sweep just before a Delay period. At the arrow, the 'Single T1' sweep button clicked. T1sweep occurred when the Delay period was to start.

If the Delay is inserted within a Loop, then Single or Evoked Repeating Sweeps can be run at the start of the next Delay Period at any time during the Loop (Fig 6.4.3.2).

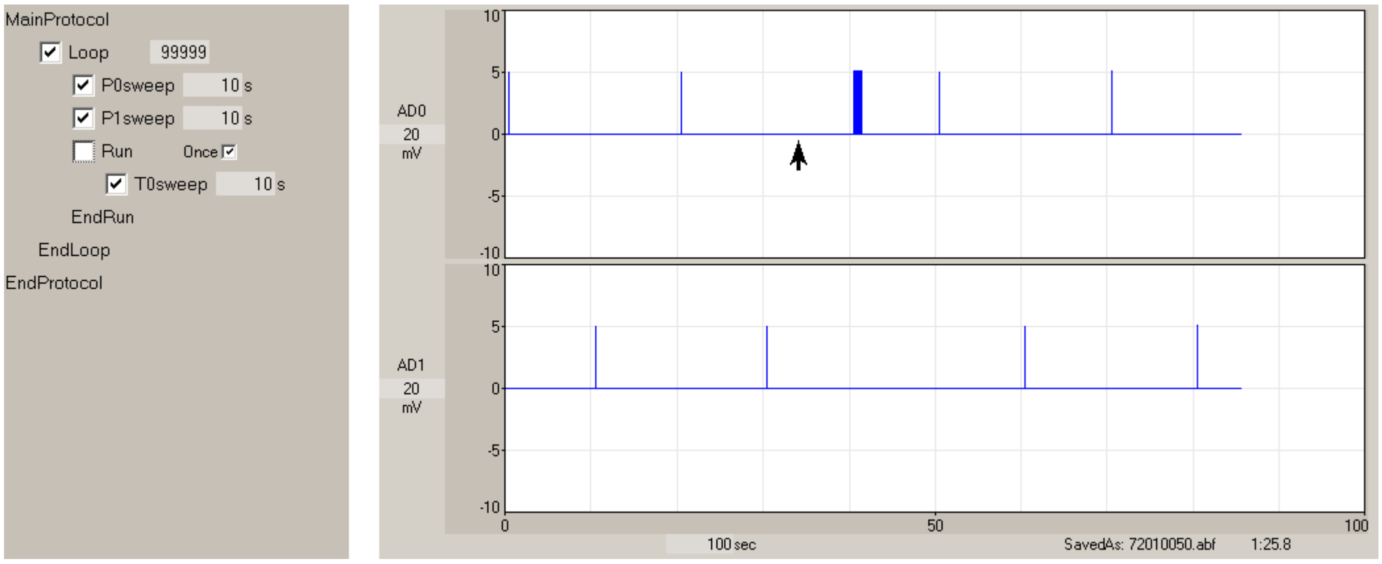


```
MainProtocol
    ☑ Loop      99999
        ☑ AvgLoop       2
            ☑ P0sweep      5 s
            ☑ P1sweep      5 s
        EndAvgLoop
        ☑ Loop          4
            ☑ Delay        5 s
        EndLoop
    EndLoop
EndProtocol
```

Fig. 6.4.3.2. Evoking a single sweep in a delay loop. At the arrow, the 'Single T0' sweep button clicked. T0sweep occurred when Delay period was to start.

6.4.4 Single/Multiple Stimulations using Run with the Once Box checked or unchecked

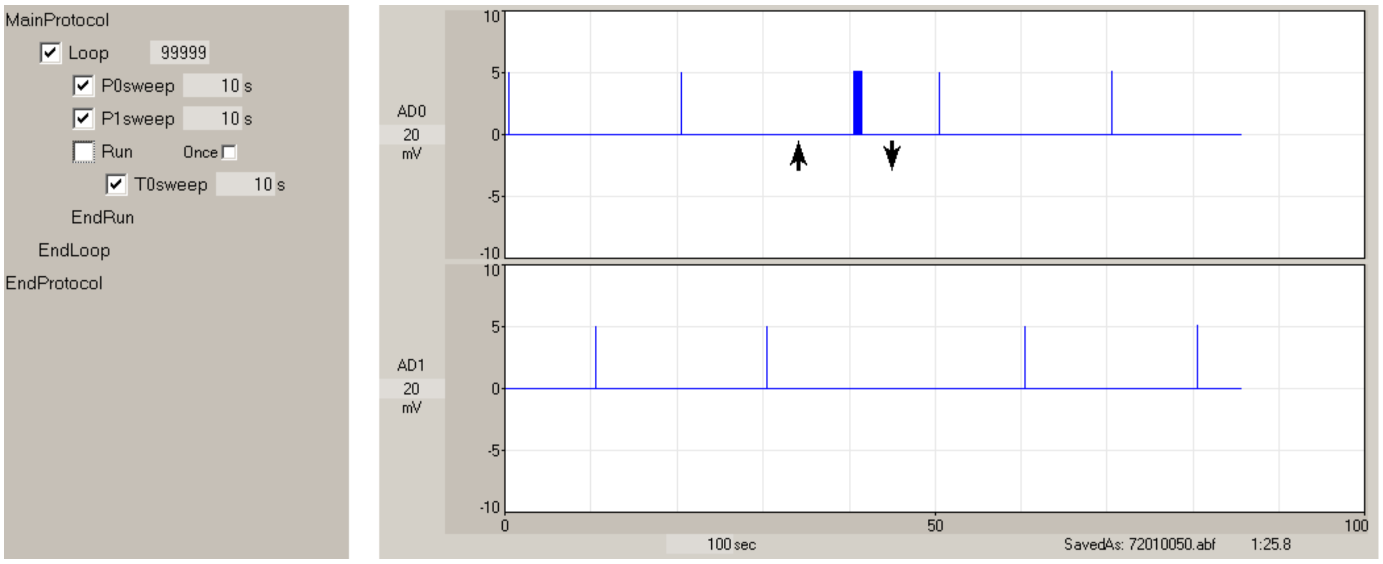
With version WinLTP095 the Once check box was added to the Run event. First, if the Run Once check box is checked, then when the large check box on the left of the Run line is checked, the code within the Run construct run once, and the large check box is unchecked just after entering the Run construct (Fig. 6.4.4.1).



```
MainProtocol
    [✓] Loop       99999
        [✓] P0sweep     10 s
        [✓] P1sweep     10 s
        [ ] Run      Once [✓]
            [✓] T0sweep     10 s
        EndRun
    EndLoop
EndProtocol
```

Fig. 6.4.4.1 Single Stimulation using a Run event with the Once Box checked At the up arrow, the large check box on the Run line is checked. Because the Once Box is checked, the T0sweep within the Run construct runs only once.

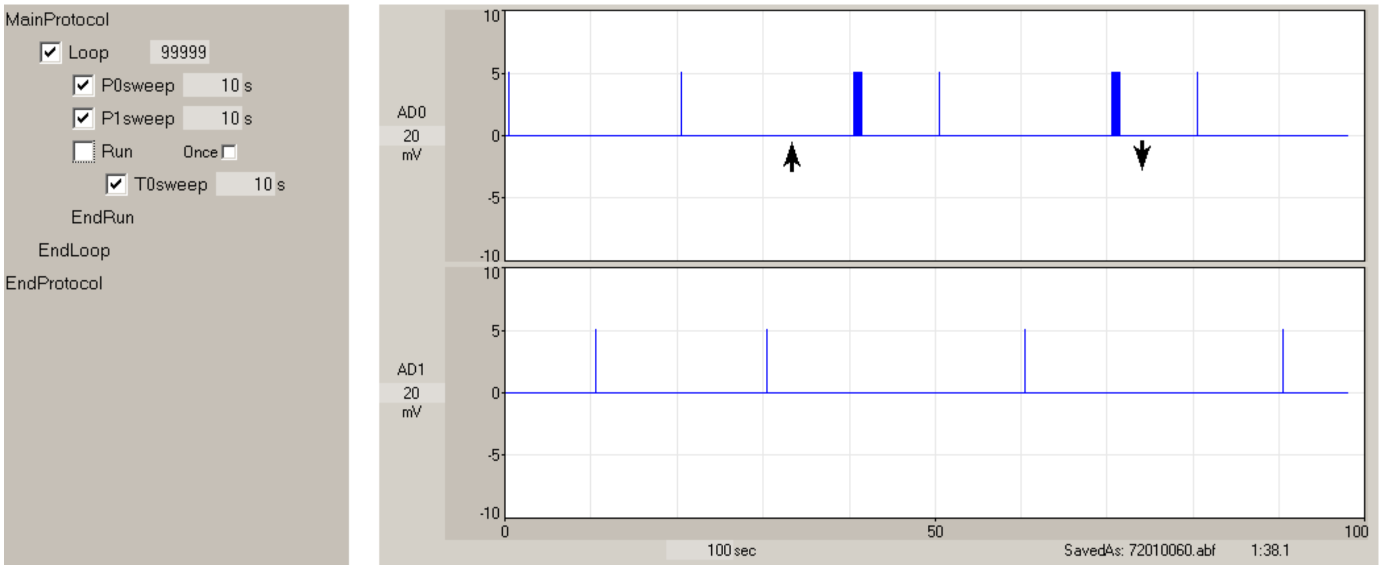
Second, if the Run Once box is not checked, then when the large check box on the left of the Run line is checked, then the code within the Run construct run for as long as the large check box is checked. To get a single stimulation as in Fig. 6.4.4.1 but with the Once Box unchecked, the large check box on the Run line has to be unchecked after the code in the Run construct has started (Fig. 6.4.4.2). (This is the way the Run atement worked in WinLTP versions 0.90 to 0.94.)



```
MainProtocol
    [✓] Loop      99999
        [✓] P0sweep    10 s
        [✓] P1sweep    10 s
        [ ] Run      Once [ ]
            [✓] T0sweep    10 s
        EndRun
    EndLoop
EndProtocol
```

Fig. 6.4.4.2. Single Stimulation using a Run event with the Once Box unchecked. At the up arrow, the large check box on the Run line is checked, but at the down arrow, the large check box must be unchecked if only one T0sweep stimulation is to occur.

Third, if the Run Once box is not checked, then when the large check box on the left of the Run line is checked, the code within the Run construct will repetitively run for as long as the large check box is checked (Fig 6.4.4.3).



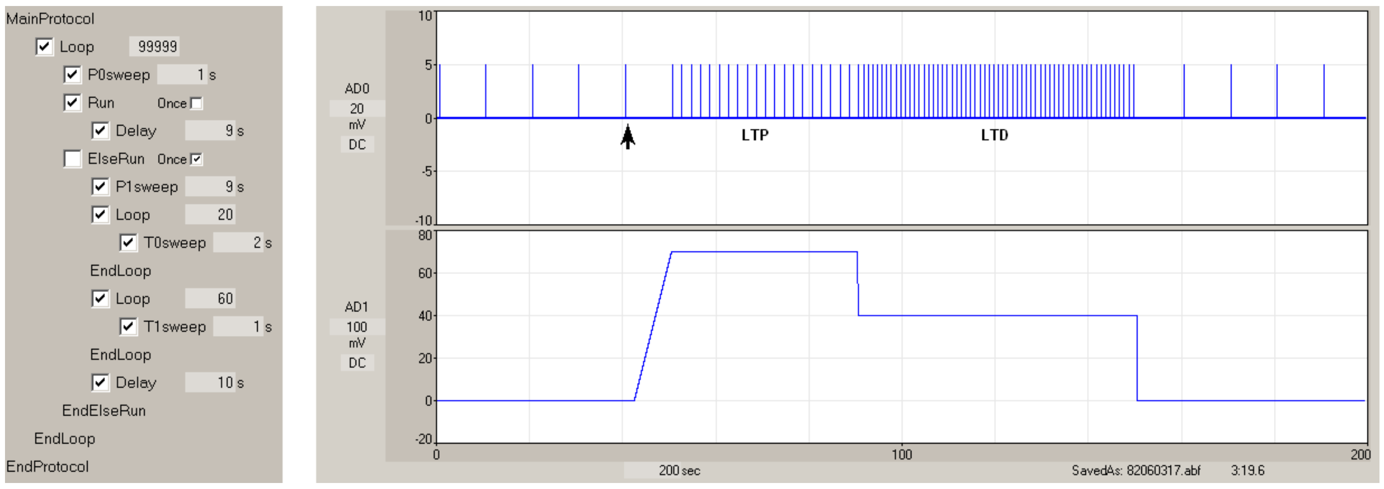
```
MainProtocol
    [✓] Loop      99999
        [✓] P0sweep    10 s
        [✓] P1sweep    10 s
        [ ] Run      Once [ ]
            [✓] T0sweep    10 s
        EndRun
    EndLoop
EndProtocol
```

Fig. 6.4.4.3. Multiple Stimulations using a Run event with the Once Box unchecked. At the up arrow, the large check box on the Run line is checked, and after two T0sweep stimulations the large check box is unchecked at the down arrow to stop the T0sweep stimulation.

6.4.5 LTP/LTD Stimulation with Steady Depolarization using the Run Once event

If the sweeps have a steady depolarization continuing throughout the sweep, and the Sweep Duration is equal to the Sweep Period (e.g. there is no time between sweeps), then steady intracellular voltage changes can accompany the extracellular stimulation

The LTP/LTD stimulation shown in Fig. 6.4.5.1 shows such steady depolarization when repetitive 0.5 Hz T0sweeps depolarized to 70 mV to produce LTP, and this was immediately followed by repetitive 1 Hz T1 sweeps depolarized to 40 mV to produce LTD. This stimulation was started by clicking the large check box on the Run line (up arrow). Initially there is a slow 9 sec depolarizing ramp in P1sweep. Because the Run Once check box was checked, once the code within the Run construct began to run the large check box automatically became unchecked, and the code in the Run construct ran only once.



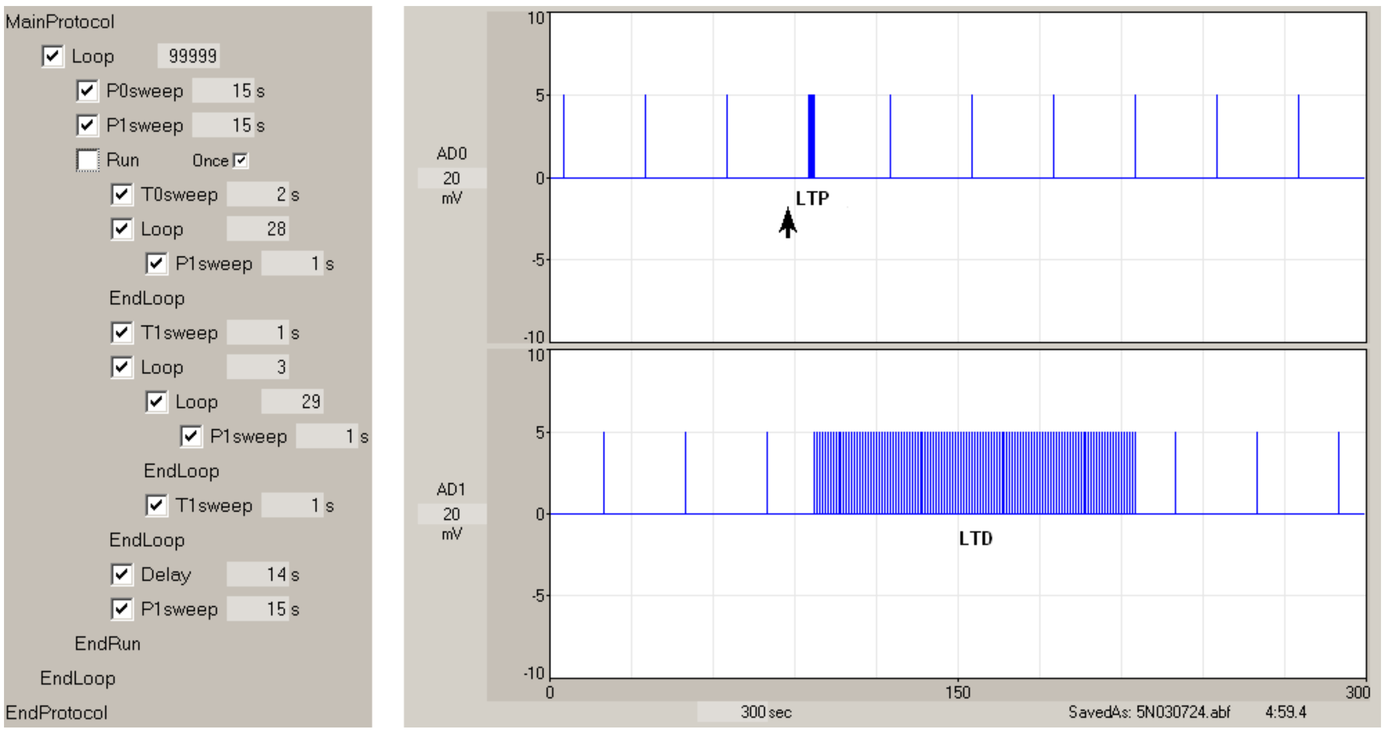
```
MainProtocol
  ☑ Loop          99999
    ☑ P0sweep       1 s
    ☑ Run          Once ☐
      ☑ Delay         9 s
    ☐ ElseRun      Once ☑
      ☑ P1sweep       9 s
      ☑ Loop           20
        ☑ T0sweep       2 s
      EndLoop
      ☑ Loop           60
        ☑ T1sweep       1 s
      EndLoop
      ☑ Delay        10 s
    EndElseRun
  EndLoop
EndProtocol
```

Fig 6.4.5.1. LTP/LTD Stimulation with Associated Steady Depolarization when the Run Once checkbox is checked. LTP/LTD stimulation was started by clicking the large check box on the Run line (up arrow) and ran only once. Initially there is a slow 9 sec depolarizing ramp in P1sweep. S0 is recorded in AD0 and intracellular stimulation voltages (AnalogOut0) is recorded in AD1. P0/T0/T1 sweeps have S0 single pulse stimulation. T0sweep has steady 70 mV depolarization and T1sweep has steady 40 mV depolarization.

6.4.6 Continuing Low Frequency Stimulation During Rapid LTD Stimulation using Run Once

Fig 6.4.6.1 shows a protocol where when the large check box on the Run line is checked, this produces an S0 train to induce LTP followed by rapid S1 1Hz LTD stimulation but with continuing S0 low frequency (1/30sec) stimulation. P0 sweep contains one S0 pulse, P1sweep contains one S1 pulse, T0sweep contains a 2 sec S0 train, and T1sweep contains both a single S0 pulse and a single S1 pulse.

Because the Run Once box was checked, the code within the Run construct ran only once, and the large check box automatically became unchecked one the code within the Run construct started.

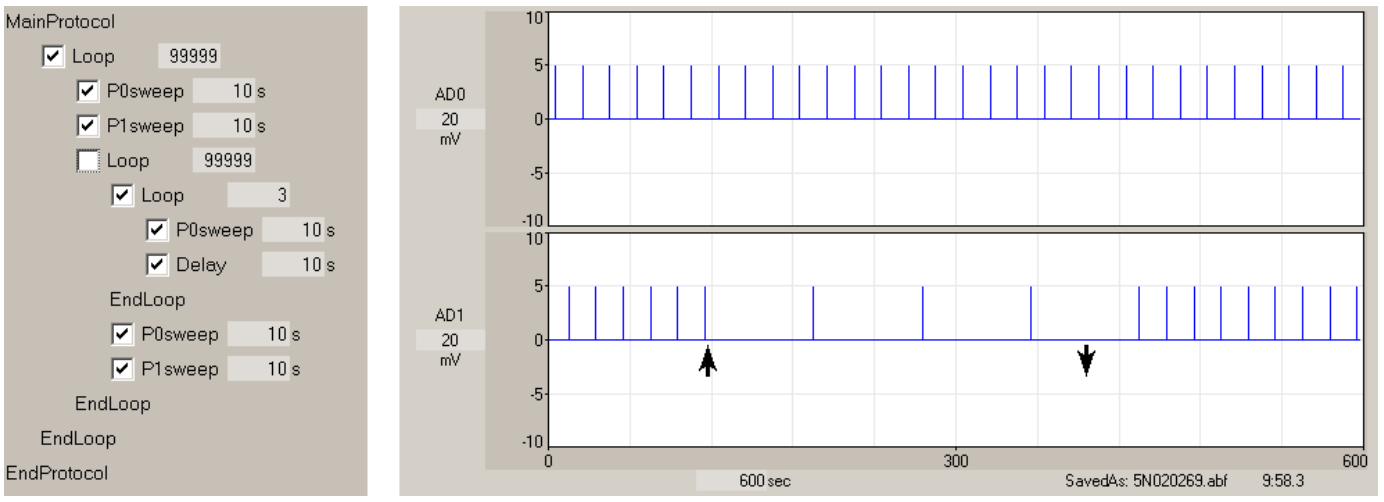


```
MainProtocol
  [✓] Loop          99999
    [✓] P0sweep     15 s
    [✓] P1sweep     15 s
    [ ] Run         Once [✓]
      [✓] T0sweep       2 s
      [✓] Loop         28
        [✓] P1sweep     1 s
      EndLoop
      [✓] T1sweep       1 s
      [✓] Loop          3
        [✓] Loop       29
          [✓] P1sweep   1 s
        EndLoop
        [✓] T1sweep     1 s
      EndLoop
      [✓] Delay        14 s
      [✓] P1sweep      15 s
    EndRun
  EndLoop
EndProtocol
```

Fig. 6.4.6.1. LTP train stimulation followed by S1 LTD stimulation with continuing slow S0 stimulation. Stimulation was started by clicking the Run check box (Up arrow), and then unclicking it (Down arrow).

6.4.7 Decreasing Pathway Stimulation by Checking and Unchecking a Continuous Loop

Fig. 6.4.7.1 is an example of how by checking and unchecking a Continuous Loop, you can keep the frequency of one pathway (S0) constant while reducing the stimulation frequency of another pathway (S1).

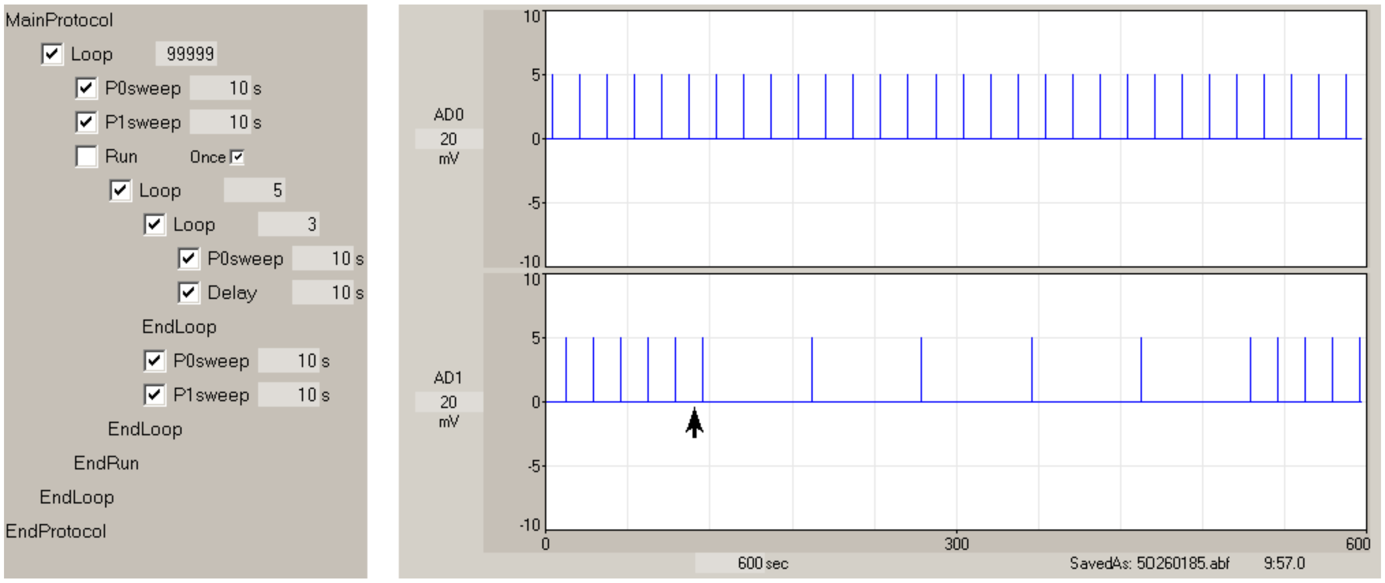


```
MainProtocol
  [✓] Loop          99999
    [✓] P0sweep     10 s
    [✓] P1sweep     10 s
    [ ] Loop        99999
      [✓] Loop          3
        [✓] P0sweep    10 s
        [✓] Delay      10 s
      EndLoop
      [✓] P0sweep      10 s
      [✓] P1sweep      10 s
    EndLoop
  EndLoop
EndProtocol
```

Fig 6.4.7.1. Lower stimulation frequency of one pathway (S1) versus another pathway (S0). The lower frequency S1 stimulation (at one-fourth the S0 frequency, Loop=3) was started by checking the inner Loop [99999] at the up arrow and stopped by unchecking this Loop at the down arrow.

6.4.8 Decreasing Pathway Stimulation using a Run Once Event and Loops

Alternatively by putting the 'Continuous Inner Loop [99999]' into a Run line and set the loop to some reasonable non-contiguous number a set number of low frequency stimulations can be produced (Fig. 6.4.8.1). The large check box on the Run line was checked at up arrow to start the lower frequency S1 stimulation lasting 5 periods (Loop = 5) and at **one-fourth** the frequency (Loop = **3**). Because the Run Once check box was checked, the code in the Run construct ran only once, and the large check box on the Run line automatically became unchecked.



```
MainProtocol
  ☑ Loop       99999
    ☑ P0sweep    10 s
    ☑ P1sweep    10 s
    ☐ Run        Once ☑
      ☑ Loop       5
        ☑ Loop       3
          ☑ P0sweep    10 s
          ☑ Delay      10 s
        EndLoop
        ☑ P0sweep    10 s
        ☑ P1sweep    10 s
      EndLoop
    EndRun
  EndLoop
EndProtocol
```

Fig. 6.4.8.1. Lower stimulation frequency of one pathway (S1) versus another pathway (S0) for a set number of times. The Run line was checked at the up arrow, and the Run code ran only once.

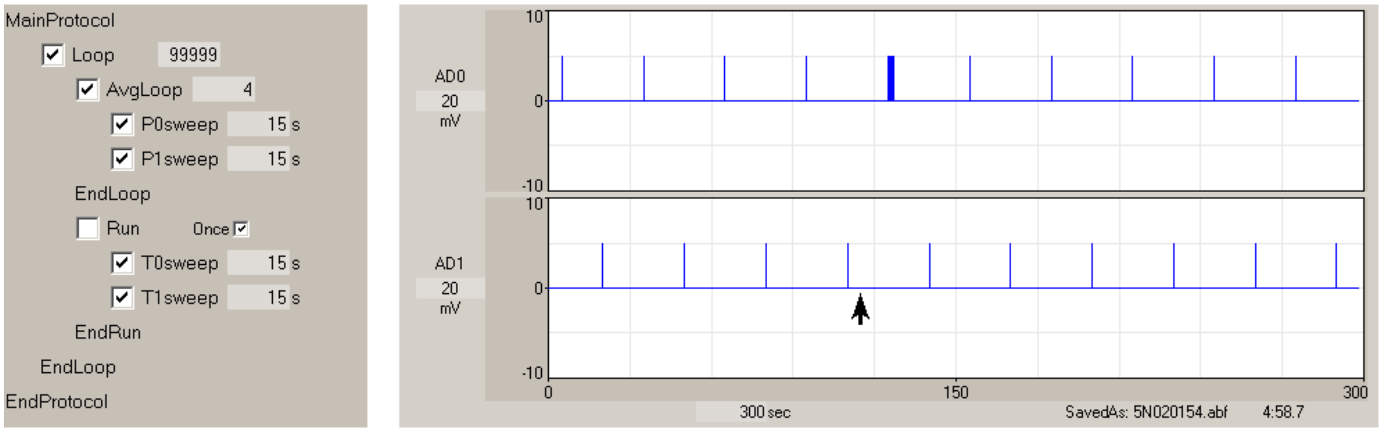
6.4.9 Alternative LTP Stimulation #1 using the Run Once Event

While running the normal basic LTP experiment, evoking a single T0 train sweep inserts the train when the next P0 or P1 sweep would have come, and delays the next P0/P1 sweeps by the T0sweep period (see Fig. 4.13.5.1). This is what happens with LTP stimulus train induction without writing any scripts.

However, some researchers may not want S0 or S1 shifted (as in Fig. 4.13.5.1). One approach is to write a protocol shown in Fig. 6.4.9.1. In this example there is normal alternating P0/P1 sweeps (containing one S0 or S1 pulse, respectively) in an AvgLoop to average every four sweeps – which is the normal basic alternating/averaging LTP experiment.

But when the large check box on the Run line is checked (at Up arrow), then a T0sweep followed by a T1sweep is then run. Because **T0 sweep has 100 S0 pulses** and **T1sweep has 1 S1 pulse**, 100 S0 pulses appear where a single S0 pulse would normally appear, and the S1 pulse frequency is not disrupted. Also, importantly, the T0sweep S0 train and the T1sweep single S1 pulse would not be put into the P0/S0 and P1/S1 pulse averaging.

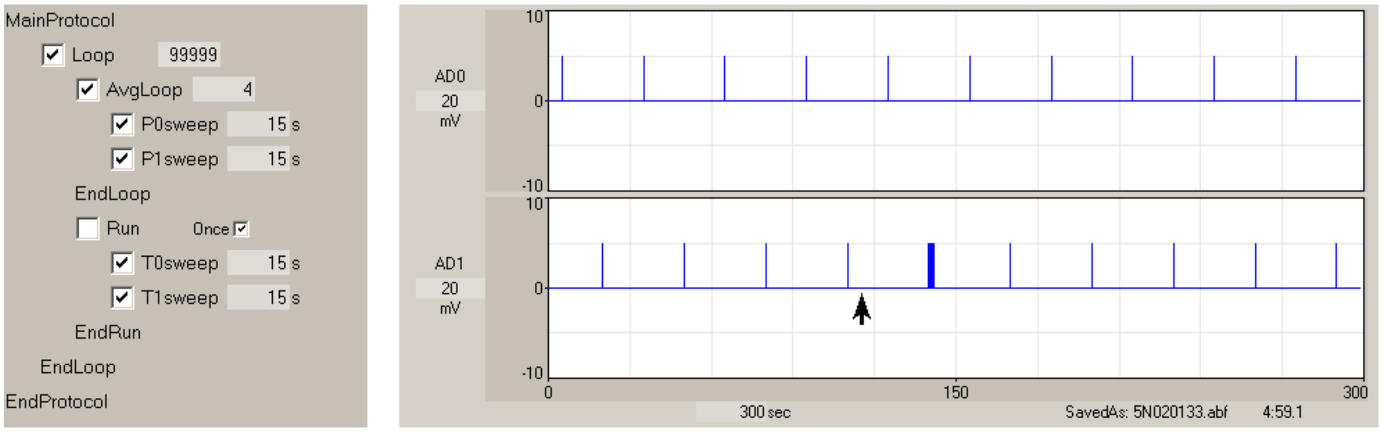
Because the Run Once box has been checked, the T0sweep/T1sweep code in the Run construct will run only once, and the large check box on the Run line will automatically become unchecked once the T0sweep code has started to run.



```
MainProtocol
  [✓] Loop      99999
    [✓] AvgLoop     4
      [✓] P0sweep   15 s
      [✓] P1sweep   15 s
    EndLoop
    [ ] Run     Once [✓]
      [✓] T0sweep   15 s
      [✓] T1sweep   15 s
    EndRun
  EndLoop
EndProtocol
```

Fig. 6.4.9.1. One type of LTP induction using code in a Run line that does not disrupt the single pulse P0/S0 and P1/S1 pulses. At the up arrow, the large check box on the Run line is checked to run the T0/T1 sweep code once.

If the same protocol is used as in Fig. 6.4.9.1, but instead of T0sweep having 100 S0 pulses and T1sweep having one S1 pulse, if **T0 sweep has 1 S0 pulse** and **T1sweep has 100 S1 pulses**, then clicking on the large check box of the Run line runs a single S0 pulse followed by an S1 train, but again P0/S0 and P1/S1 pulses are not disrupted (Fig. 6.4.9.2).



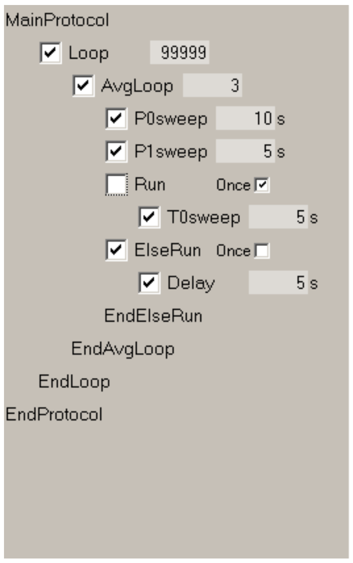
```
MainProtocol
  [✓] Loop      99999
    [✓] AvgLoop     4
      [✓] P0sweep   15 s
      [✓] P1sweep   15 s
    EndLoop
    [ ] Run     Once [✓]
      [✓] T0sweep   15 s
      [✓] T1sweep   15 s
    EndRun
  EndLoop
EndProtocol
```

Fig. 6.4.9.2. The same type of LTP induction as in Fig. 6.4.9.1 using code in a Run line that does not disrupt the single pulse P0/S0 and P1/S1 pulses, except T0 sweep has 1 S0 pulse and T1sweep has 100 S1 pulses. At the up arrow, the large check box on the Run line is checked to run the T0/T1 sweep code once.

6.4.10 Alternative LTP Stimulation #2 using the Run/ElseRun Construct

A different type of LTP train stimulation during signal averaging is shown in Fig. 6.4.10.1 where the P0sweep/S0 and P1sweep/S1 pulse output is not disrupted, but a T0sweep with S0 train is run half way between the last S1 pulse of a 3 pulse average and the next S0 pulse of the next 3 pulse average. However, the S0 train need not be run half way between the S1 and S0 pulses, any fraction is possible.

Initially the large check box of the Run line containing the T0sweep is **unchecked** and the large check box of the ElseRun line containing the Delay is **checked.** At the up arrow the large check box on the Run line was **checked,** and the large check box on the ElseRun line is automatically unchecked. Since the Run line Once check box was checked, only one T0sweep was produced when Delay period was to start, and this did not delay onset of next P0sweep.



```
MainProtocol
  ☑ Loop        99999
    ☑ AvgLoop      3
      ☑ P0sweep    10 s
      ☑ P1sweep     5 s
      ☐ Run      Once ☑
        ☑ T0sweep   5 s
      ☑ ElseRun  Once ☐
        ☑ Delay     5 s
      EndElseRun
    EndAvgLoop
  EndLoop
EndProtocol
```

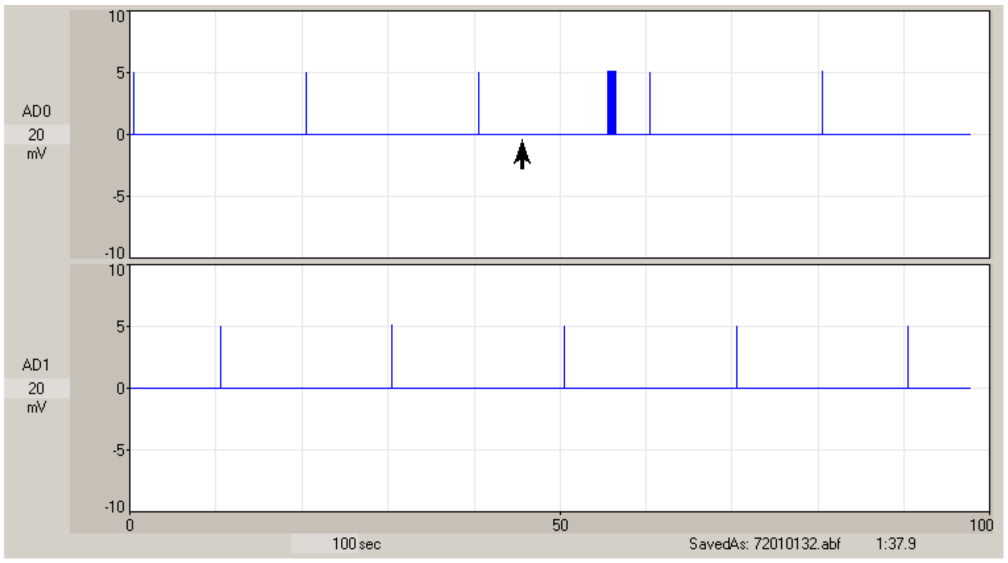
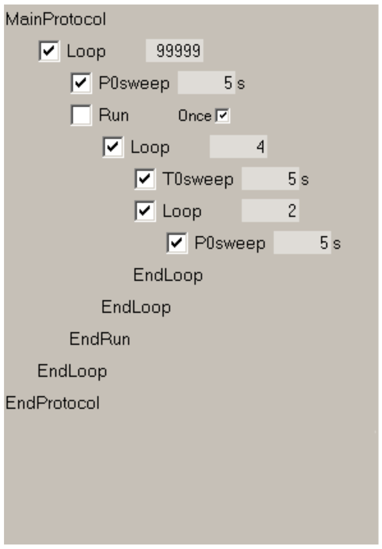


Fig. 6.4.10.1. Producing an LTP stimulation train half way between the last S1 pulse of a 3 pulse average and the next S0 pulse of the next 3 pulse average. At the up arrow the large check box on the Run line was **checked.** One T0sweep/S0 train was produced when Delay period was to start, but did not delay onset of next P0sweep/S0 pulse.

6.4.11 In-Vitro Kindling

Fig. 6.4.11.1 is an example of “in-vitro kindling” like I use to do. To start the 4 train stimulation code (see the Loop of 4), with 2 interspersed single pulse P0sweeps (see the Loop of 2), click on the large check box of the Run line (at arrow). Because the Run Once box has been checked, the kindling code in the Run construct was run only once.



```
MainProtocol
  [✓] Loop          99999
    [✓] P0sweep         5 s
    [ ] Run       Once [✓]
      [✓] Loop              4
        [✓] T0sweep         5 s
        [✓] Loop            2
          [✓] P0sweep       5 s
        EndLoop
      EndLoop
    EndRun
  EndLoop
EndProtocol
```

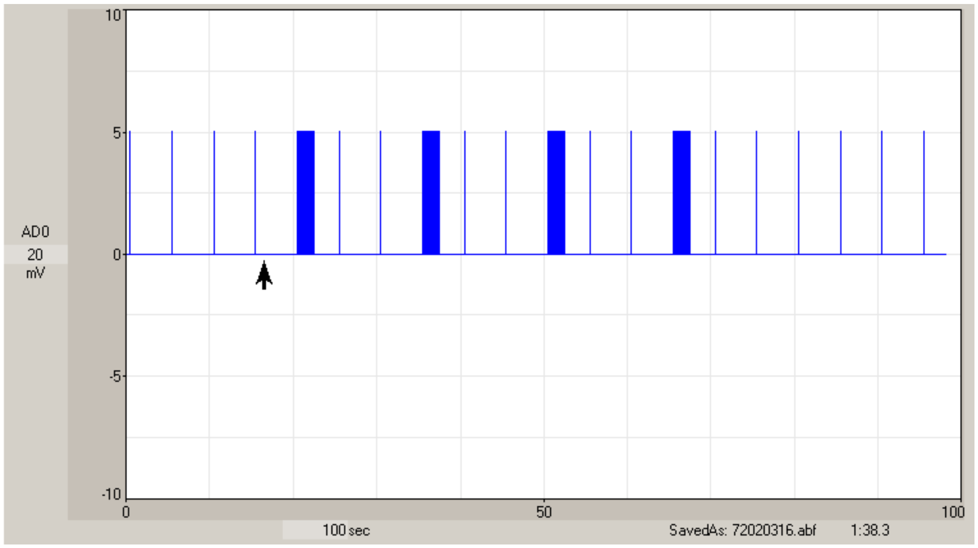


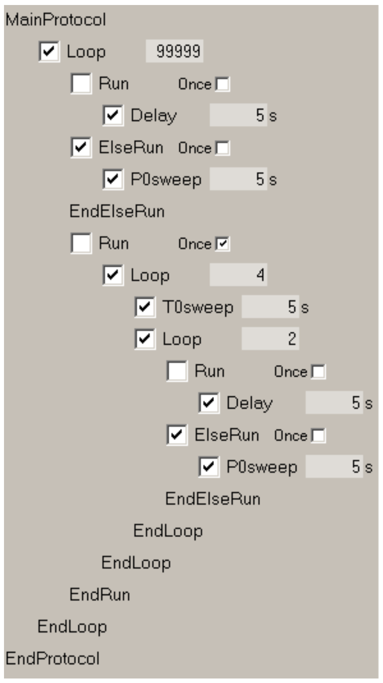
Fig. 6.4.11.1 An in-vitro kindling protocol using the Run line. At the up arrow the Run large check box was checked, and once inside the Run code (when the first train went off), the Run large check box was unchecked (down arrow).

Fig. 6.4.11.2 is a more realistic example of in-vitro kindling protocol in with the Run/ElseRun construct allows the P0sweep in the outer loop containing the baseline non-kindling code to be turned on or off, and the P0sweep in the inner loop of the kindling code to also be turned on or off.

At the left up arrow, the large check box of the Run line was checked to beginning running the 4 stimulus train kindling one time through.

At the middle up arrow the Large check box of the Run line of the inner Run/ElseRun construct was checked to stop P0sweep stimulation during the kindling stimulation by substituting a Delay Period.

At the right up arrow the Large check box of the Run line of the outer Run/ElseRun construct was checked to stop P0sweep stimulation after the kindling stimulation had finished by substituting a Delay Period.



```
MainProtocol
  [✓] Loop        99999
    [ ] Run          Once [ ]
      [✓] Delay        5 s
    [✓] ElseRun  Once [ ]
      [✓] P0sweep      5 s
    EndElseRun
    [ ] Run          Once [✓]
      [✓] Loop          4
        [✓] T0sweep       5 s
        [✓] Loop          2
          [ ] Run          Once [ ]
            [✓] Delay        5 s
          [✓] ElseRun  Once [ ]
            [✓] P0sweep      5 s
          EndElseRun
        EndLoop
      EndLoop
    EndRun
  EndLoop
EndProtocol
```

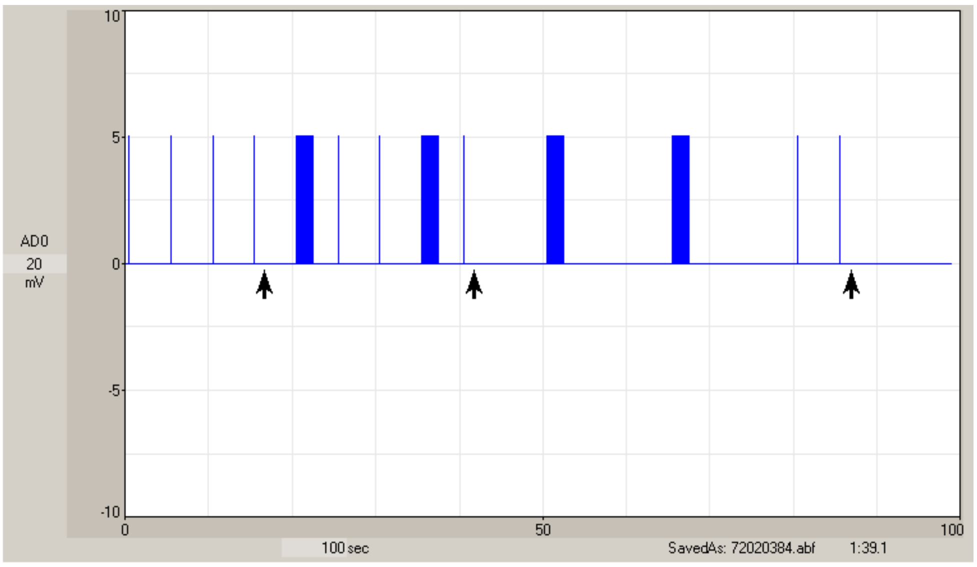


Fig. 6.4.11.2. An in-vitro kindling protocol using the Run line, and the Run/ElseRun constructs to turn on or off P0sweep stimulation.

CHAPTER 7 – Protocol Builder – Sequential Scripting

7.1 Circular Scripting to Sequential Scripting

In all the previous Protocol Builder examples in Chapter 6, the code in the MainProtocol/EndProtocol construct a circular script consisting of a Continuous Loop construct, eg:

```
MainProtocol
    ☑ Loop      [99999]        ‘ Continuous loop because of the 99999 loop number
         ...........           ‘ Any events in the Continuous Loop
    EndLoop
EndProtocol
```

In circular scripting, a repeating P0sweep or alternating repetitive P0sweep/P1sweep event was used to provide baseline stimulation and regular stimulation throughout the experiment and in the washout. Then **intermittent stimulation** was provided every once in awhile by evoking it by clicking on ‘Single PXsweep’ or ‘Repeat PXsweep’ Run Buttons, or by producing stimulation by clicking on the large check boxes of Run lines, Run/ElseRun constructs, Loop and/or AvgLoop lines

In contrast, sequential scripts follow a linear, not circular, order of events. This is what experiments really are - a linear order of events. Sequential scripts usually start with a repeating P0sweep or alternating repetitive P0sweep/P1sweep events to provide baseline stimulation, and then **intermittent stimulation** was provided every once in awhile by evoking it by clicking on ‘Single PXsweep’ or ‘Repeat PXsweep’ Run Buttons, or by producing stimulation by clicking on the large check boxes of Run lines, Run/ElseRun constructs, Loop and/or AvgLoop lines

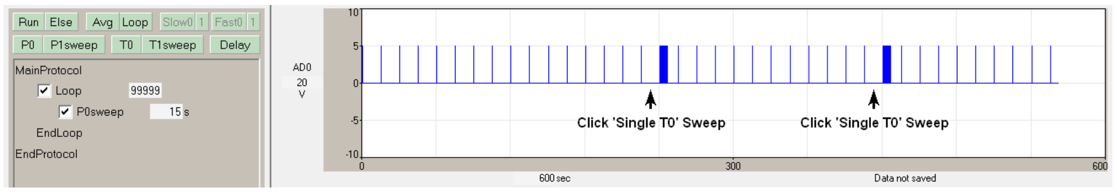
In sequential scripting, scripts follow a linear, not circular, order of events, which is what experiments really are, a linear order of events. Usage of sequential scripting coupled with automated perfusion control allows the Automated Experiments discussed in Chapter 9.

To get a better idea of how circular scripting relates to sequential scripting, look at Fig. 7.1.1. Fig. 7.1.1 shows three different protocols that can produce a simple LTP experiment: 1) A baseline of repetitive single S0 pulses (one S0 pulse in a P0sweep), 2) the first LTP induction stimulation by a train of S0 pulses in a T0sweep, 3) more repetitive S0 pulses, 4) a second LTP induction stimulation by delivering the train of S0 pulses in the T0sweep, and 5) ending with the final repetitive S0 pulses.

For researchers that use WinLTP in the Basic Mode to do a basic LTP experiment, the Continuous [9999] Loop contains one P0sweep (with one S0 pulse) that is output every 2 seconds in this example. The two single S0 trains for LTP induction are output by clicking the ‘Single T0’ sweep button twice at the appropriate times (Fig. 7.1.1A). **The researcher has to be present each time to click the ‘Single T0’ sweep button.**

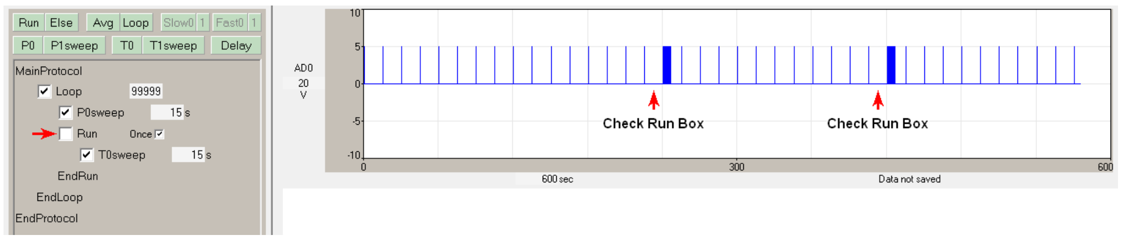
Fig 7.1.1B shows how this protocol can be produced using the Protocol Builder in the WinLTP Advanced Mode by checking the check box of the Run statement each time a single T0sweep is to be elicited. **The researcher has to be present each time to check to Run box.** Normally the statements located between the Run and EndRun statements enable more complex stimulations than just the single T0sweep here.

A Circular Protocol using Evoked Events



```
MainProtocol
  ☑ Loop        99999
    ☑ P0sweep      15 s
  EndLoop
EndProtocol
```

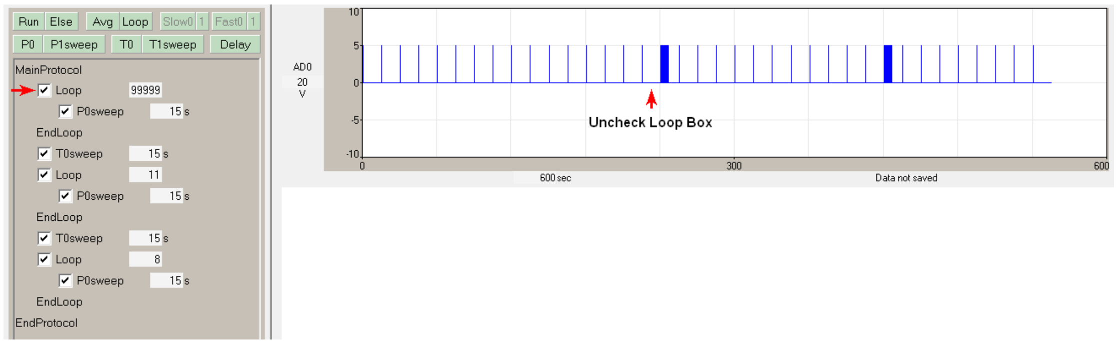
B Circular Protocol with a Run Statement



```
MainProtocol
  ☑ Loop        99999
    ☑ P0sweep      15 s
    ☐ Run        Once ☑
      ☑ T0sweep      15 s
    EndRun
  EndLoop
EndProtocol
```

Run Else Avg Loop Slow0 1 Fast0 1
P0 P1sweep T0 T1sweep Delay
AD0 20 V
Check Run Box
Check Run Box
600 sec
Data not saved

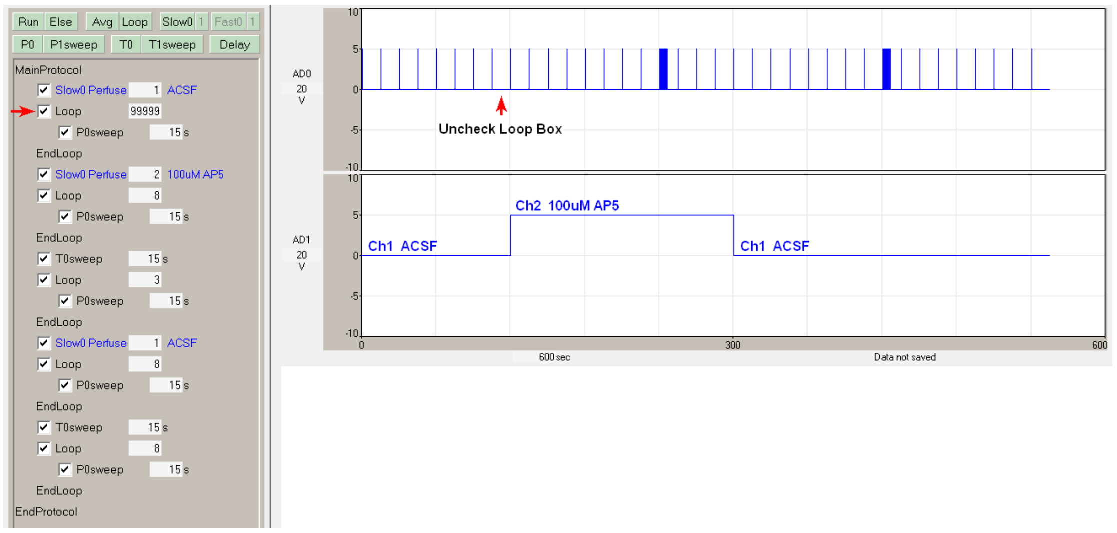
C Sequential Protocol without Perfusion Changes



```
MainProtocol
  ☑ Loop        99999
    ☑ P0sweep      15 s
  EndLoop
  ☑ T0sweep      15 s
  ☑ Loop           11
    ☑ P0sweep      15 s
  EndLoop
  ☑ T0sweep      15 s
  ☑ Loop            8
    ☑ P0sweep      15 s
  EndLoop
EndProtocol
```

Run Else Avg Loop Slow0 1 Fast0 1
P0 P1sweep T0 T1sweep Delay
AD0 20 V
Uncheck Loop Box
600 sec
Data not saved

D Sequential Protocol with Perfusion Changes



```
MainProtocol
  ☑ Slow0 Perfuse   1  ACSF
  ☑ Loop        99999
    ☑ P0sweep      15 s
  EndLoop
  ☑ Slow0 Perfuse   2  100uM AP5
  ☑ Loop            8
    ☑ P0sweep      15 s
  EndLoop
  ☑ T0sweep      15 s
  ☑ Loop            3
    ☑ P0sweep      15 s
  EndLoop
  ☑ Slow0 Perfuse   1  ACSF
  ☑ Loop            8
    ☑ P0sweep      15 s
  EndLoop
  ☑ T0sweep      15 s
  ☑ Loop            8
    ☑ P0sweep      15 s
  EndLoop
EndProtocol
```

Fig. 7.1.1. Evolution from circular scipting to sequential scripting with perfusion changes. This protocol shows a series of non-averaging P0sweeps (containing a single S0 pulse) separated by two T0sweeps (containing an LTP induction train of S0 pulses) A) A circular script for a basic LTP experiment in the Basic Mode. Two single T0sweeps are evoked by clicking the 'Single T0' sweep buttons. B) A Circular script made using the Advanced Mode Protocol Builder. The two single T0sweeps are evoked by **checking** the Run statement checkbox (red arrows). C) A sequential script made using the Advanced Mode Protocol Builder. The entire stimulation sequence is started by **unchecking** the Continuous [99999] Loop checkbox (red arrow), and the researcher can leave at this time if there are no perfusion changes to do. D) A sequential script including Perfuse events to change perfusion solution between sweeps. The entire stimulation sequence is also started by **unchecking** the Continuous [99999] Loop checkbox (red arrow), and **the researcher can leave at this time**. The bottom tace records Bit1 of Port1 to indicate switching from Ch 1 to Ch 2 and back to Ch1 solutions.

7.2 Sequential Scripting

Fig 7.1.1C shows how this protocol can be produced using a Sequential Script in the Advanced Mode Protocol Builder. First, the baseline section contains a Continuous [99999] Loop that contains one P0sweep (which outputs 1 S0 pulse). At the start of the experiment (eg at the start of the MainProtocol), the checkbox of this Loop statement is checked. Second, when this Loop checkbox is **unchecked** (pointed to by the red arrow), this continuous loop is **exited**, and the next statement in the Protocol Builder script is run, a single T0sweep (to output a train of S0 pulses). Third, after this T0sweep, a non-continuous loop of only 9 loops is run, each loop containing the P0sweep (to output repetitive S0 pulses). Fourth, then after the 9 P0sweeps are run, the second T0 sweep is run to produce a second S0 pulse train. And fifth, after this second T0sweep, another non-continuous loop of only 6 loops is run to finish up the experiment by outputting 6 more repetitive S0 pulses.

For this protocol, the researcher only has to be present to uncheck to Run statement box (when baseline stability has been reached) to set the rest of the stimulation going automatically. **The researcher can leave if no perfusion changes need to be made.**

The following annotated script more clearly explains what is going on in the sequential script of Fig. 7.1.1C.

```
MainProtocol
                                ‘ Baseline, continuous S0 stimulation
   ☑ Loop      [99999]          ‘ Exit baseline looping by unchecking Loop large check box
      ☑ P0sweep [  2]s          ‘ Single S0 pulse
   EndLoop

   ☑ T0sweep [  5]s             ‘ The first LTP stimulation, S0 train

   ☑ Loop      [   9]           ‘ Post first LTP S0 stimulation
      ☑ P0sweep [  2]s
   EndLoop

   ☑ T0sweep [  5]s             ‘ The second LTP stimulation, S0 train

   ☑ Loop      [   6]           ‘ Post second LTP S0 Stimulation
      ☑ P1sweep [  2]s
   EndLoop

EndProtocol
```

7.3 Sequential Scripting with Perfusion Changes

Although sequential scripting is useful when delivering automated electrical stimulation, a major strength becomes apparent is when the automated delivery of electrical stimulation is coupled with the automated change of perfusion solutions during the experiment. Fig. 7.1.1D shows an example of switching from Ch1 ACSF perfusion fluid to Ch 2 100uM AP5, delivering a LTP induction train in AP5, switching back from Ch2 AP5 to Ch 2 ACSF wash, and delivering a second LTP induction train. The whole stimulation / perfusion change sequence is started by exiting the baseline loop by unchecking the Loop event. **The researcher can leave at this time.**

Many more examples of Sequential Scripting with perfusion changes are presented in Chapter 9 (see Figs. 9.3.1.2.3, 9.3.1.3.1, 9.3.2.1, 9.4.1.4 and 9.4.2.4).

CHAPTER 8 – Protocol Linking

8.1 Using Protocol Linking to extend Circular and Sequential Scripting

We have developed Protocol Linking to extend MainProtocol scripting and to fix three current limiations of MainProtocol scripting:

1) There are not enough different PxSweep stimulations available in one MainProtocol run.

   This is a particular problem for using Automated Perfusion Control Using Single-Line Fast Switching Perfusion (Section 9.3). In WinLTP 2.01 there are only four different PxSweep stimulations in each MainProtocol run, and this means that only four different perfusion solutions can be delivered during each MainProtocol run. However, patch-clamp automated single-line perfusion changes can realistically involve delivering at least 16 different perfusion solutions. With Protocol Linking, once a protocol has finished, WinLTP can now load and start a second protocol containing 4 more different solution changes, and then load and start a third protocol containing 4 further different solution changes, and WinLTP can do this, ad infinitum using Protocol Linking.

3) Extend the running time of the MainProtocol

   Currently there is a 14 hr, 54 minute limitation running a MainProtocol with National Instruments M-series boards.

2) Save the gap-free Continuous Acqusition ABF files in smaller, reasonable lengths

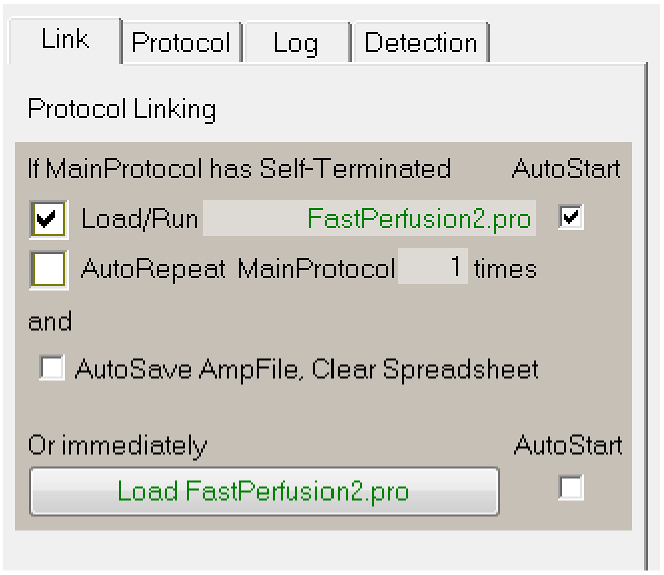
   By either a) loading and starting a subsequent protocol and so forth, or b) restarting the MainProtocol, Continuous Acqusisiton files can be acquired and saved in say 1 hour seqments rather than the whole MainProtocol run of say 6 hrs.

The Protocol Linking panel is located on the Link tabsheet (Fig. 8.1.1). This Protocol Linking panel adds three capabilities to WinLTP:

1) After the current MainProtocol has self-terminated, iif the Load/Run check box is checked, link to another protocol file (in this case 'FastPerfusion2.pro') to load it, and automatically start it if the AutoStart checkbox is checked, or
2) Automatically repeat and restart the MainProtocol after it has self-terminated if the 'AutoRepeat' checkbox is checked, or you can
3) Immediately load the protocol file in the Load/Run edit field by clicking the 'Load ProtocolFile.pro' button, and automatically start this new protocol if the Immediate AutoStart checkbox is checked.

Also, before the loading of a new linked protocol file or restarting the auto-repeating MainProtocol, you can automatically save the AmpFile and clear the spredsheet if the 'AutoSave AmpFIle, Clear Spreadsheet' checkbox is checked.

And if you decide to AutoRepeat the MainProtocol file, you can change the number of times the current protocol will repeat. This is helpful if you want to en essence extend the time the MainPrrotocol is running. For example, a current limitation of the National Instruments M-series boards is that a MainProtocol cannot run longer than 14 hrs and 54 min. Using Load/Run the next protocol file with AutoStart, or using AutoRepeat gets around this limitation (Section 8.3).



Protocol Linking

If MainProtocol has Self-Terminated — AutoStart

[✓] Load/Run FastPerfusion2.pro [✓]

[ ] AutoRepeat MainProtocol 1 times

and

[ ] AutoSave AmpFile, Clear Spreadsheet

Or immediately — AutoStart

Load FastPerfusion2.pro [ ]

Fig. 8.1.1. The Protocol Linking tabsheet.

8.2 Linking to the Next Protocol File

There are many reasons to want to link to a subsequent protocol file. The experiment script may be clearer if it is subdivided into different protocol files. There is a ca 70 line limit to Protocol Builder scripts. One of the major problems in WinLTP 2.01 is that there are only four different PxSweep stimulations in each MainProtocol run. This is a particular problem for using Automated Perfusion Control Using Single-Line Fast Switching Perfusion (Section 9.3), this means that only four different perfusion solutions can be delivered during each MainProtocol run. However, patch-clamp automated single-line perfusion changes can realistically involve delivering at least 16 different perfusion solutions. With Protocol Linking, once a protocol has finished, WinLTP can now load and start a second protocol containing 4 more different solution changes, and then load and start a third protocol containing 4 further different solution changes, and WinLTP can do this, ad infinitum using Protocol Linking.

Ultimately we plan to put in 8 PulseSweeps and 8 TrainSweeps for a total of 16 different sweeps. However, even if with 16 different PxSweeps stimulations for applying 16 different solutions, it is very conceivable that a researcher would want to deliver 32 different solutions and therefore would have to use Protocol Linking anyways.

An example of Protocol Linking for Automated Patch-Clamp Experiments using Fast Switching with Single Line Perfusion is presented in Section 9.3.2.

8.3 Saving the Continuous Acqusition Files in Reasonable Durations

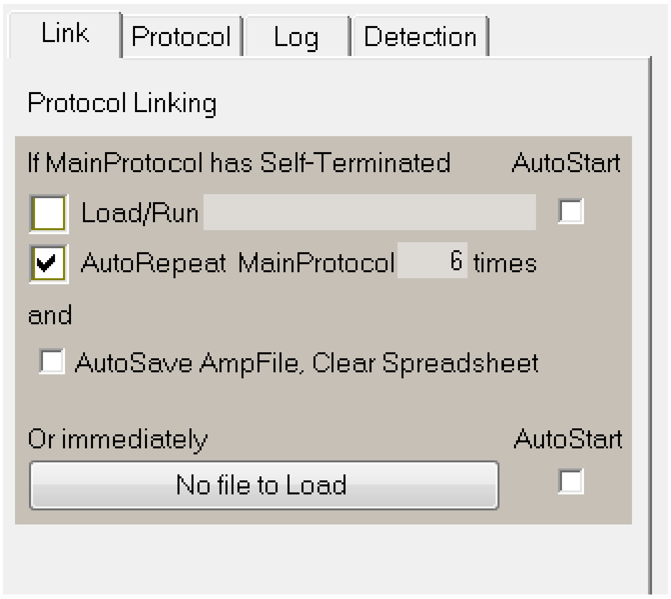
Using the AutoRepeat feature as described in Section 8.3 can also be used to save the gap-free Continuous Acquisition ABF file in smaller, more reasonable durations.

This is because there is one Continuous Acquisition file for each MainProtocol run. When the AutoRepeat is set to > 1 times, this will then divide the Continuous Acquisition gap-free ABF file into more reasonably sized chunks. By doing this you could have a new Continuous Acqusition file every hour rather than having one Continuous Acqusition file for the whole duration of the MainProtocol run.

For example, if you are using the normal circular protocols when performing a basic LTP exeriment you can use the AutoRepeat MainProtocol capability shown in Fig. 8.3.1.. In this case a bog standard protocol has been chosen to produce the MainProtocol in the right of Fig. 1. The protocol is set to produce one P0sweep every 60 sec or 1 minute. However, rather than looping for 99999 times, ie 'continuously' or almost 'forever', it only loops for 60 times, ie for 60 x 1 min or for 1hr. After this time the MainProtocol stops or **self-terminates**. When the MainProtocol self-terminates, the Continuous Acquisition file is closed.

However, since the AutoRepeat MainProtocol checkbox in the left of Fig. 1 has been checked, when the MainProtocol self-terminates after 1 hr, the MainProtocol then restarts, ie AutoRepeats. When the MainProtocol starts again, another Continuous Acquisition file is created to begin acquiring more data. Since the number of times is set to 6, the 1 hr long MainProtocol will repeat 6 times, the experiment will last for6 hrs, and 6 x1 hour long Continuous Acqusition files will be created.

At any time during the running of these MainProtocols, single or repeat sweep stimulations can be evoked as usual. This same idea can be used when using Sequential Scripts in the Protocol Builder.



Protocol Linking

If MainProtocol has Self-Terminated — AutoStart

☐ Load/Run ☐

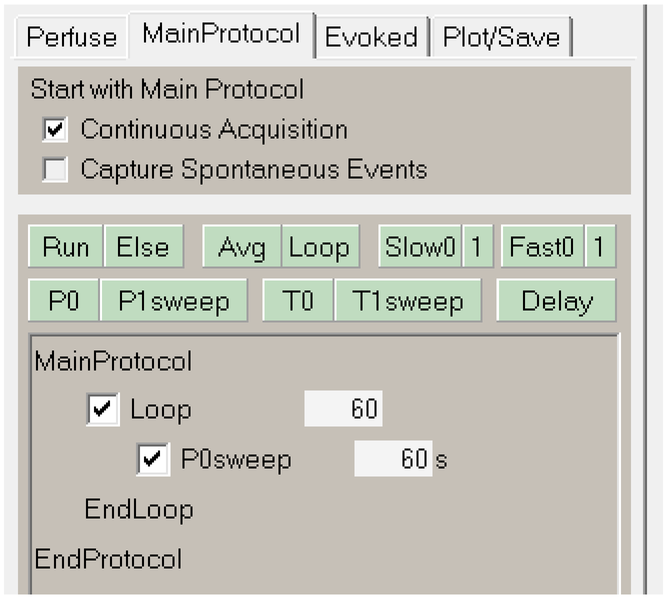
☑ AutoRepeat MainProtocol 6 times

and

☐ AutoSave AmpFile, Clear Spreadsheet

Or immediately — AutoStart

No file to Load ☐



Start with Main Protocol

☑ Continuous Acquisition

☐ Capture Spontaneous Events

MainProtocol

☑ Loop 60

☑ P0sweep 60 s

EndLoop

EndProtocol

Fig. 8.3.1.. AutoRepeat MainProtocol 6 times to save another Continuous Acquisition file once every hour for 6 hours. The Left figure shows the Protocol Linking tabsheet with the AutoRepeat MainProtocol checked to repeat for 6 times. The right figure shows a protocol which produces a sweep every 60 sec or 1 min, and loops for 60 times to give a total duration for a single MainProtocol of 1 hr.

8.4 Extending the Run-Time of the MainProtocol

A third reason you might want to use Protocol Linking is to extend the running of the MainProtocol. For M-series boards, there is currently a limitation of running the MainProtocol more than 14 hrs and 54 min (Appendix B.2). This is due to an overflow of a 32-bit signed integer which should be a 64-bit integer, and there is a fix scheduled, but it is definitely not immediate.

For X-series boards, there is an even worse limitation that is not present in M-Series boards – this bug limits the running the MainProtocol to 1 hrs, 29 min. This is due to an overflow of another 32-bit signed integer that should be a 64-bit integer, and the fix is scheduled for the next NI-DAQmx version, 9.5.

An solution analgous to the one in Fig. 8.3.1. will solve the problem.

CHAPTER 9 – Automated Experiments using Automated Perfusion Control

Note: we have recently published a paper on using WinLTP for Automated Perfusion experiments in the Journal of Neuroscience Methods:

> Anderson WW, Fitzjohn SM and Collingridge GL (2012) Automated Multi-Slice Extracellular and Patch-Clamp Experiments using the WinLTP Data Acquisition System with Automated Perfusion Control, *J Neurosci. Methods*, in press.

This paper should be a good, quick synopsis of using automated perfusion in WinLTP.

9.1 Automated Perfusion Control

Although Sequential Protocol Scripting (already available in WinLTP versions 1.11b and earlier) can generate all the stimulation pulses needed for the typical LTP experiment after a stable baseline has been achieved, manual changing of bath perfusion fluids is still required. The big advance in WinLTP 2.00 and 2.01 is automated perfusion control by adding the Perfuse statement to sequential protocol scripting (Fig. 9.2.7.2B).

This means that once baseline stability has been achieved, for extracellularly recorded experiments, you don't have to stay around watching the experiment in order to change solutions – because unlike patch-clamping, you can't do much to rectify an extracellular electrode recording problem anyways. You can go read a paper in peace, reanalyse yesterday's experiment in peace, run another slice, or go to the pub. You don't have to stay around 'watching paint dry'.

WinLTP can control up to four perfusion lines. Slow0 Perfusion Change can control one perfusion line by making only 'slow' changes BETWEEN PxSweeps. The same holds for Slow1 Perfusion Change. However, Fast0 Perfusion Change can control one perfusion line by making 'fast' changes DURING sweeps AS WELL AS BETWEEN sweeps. The same holds for Fast1 Perfusion Change. Also, Fast0 and/or Fast1 Perfusion change can control piezo steppers between sweeps, but this is discussed in automated perfusion control for patch-clamping.

Normally WinLTP would be using Slow0 Perfusion Change for controlling one perfusion line to one extracellular slice chamber, and that is how extracellular experiments will be dealt with here. However, all four perfusion line controls (Slow0, Slow1, Fast0 and Fast1) can be used for extracellular slice and patch clamp experiments.

9.2 Automated Multi-Slice Extracellular Experiments

Note: the automated perfusion control is currently only supported by National Instruments M- or X-Series board - the Digidata 132x boards should be supported in the not too distant future.

There are three basic concepts for WinLTP Automated Multi-Slice experiments:

1) Sequential Protocol Scripts
   + Automated Perfusion Control
   + Running many WinLTP programs at once = Automated Multi-Slice Experiments

2) 1 WinLTP / ADboard controls → 1 Perfusion Controller controlling solution in
   → 1 'Chamber' which can contain
   → 1 or more 'Wells', each well containing 1 Slice

3) We favour : 1 Slice from 1 animal per experiment is N = 1

Before we get into detailed discussion of using WinLTP for multi-slice experiments, we would like to note that there are several commercial, fully integrated systems for doing multi-slice experiments. These include:

1) SliceMaster from Scientifica which comes in 4 and 8 slice systems and uses conventional extracellular recording.
2) SynchroSlice from Lohman Research which comes as a 4 slice system and uses conventional extracellular recording.
3) MultiChannel Systems has developed a system that can record from brain slices using a planar 32 channel multielectrode array system, and can include their LTP-Director software to perform the usual LTP experimental protocols and analyses. Several of these systems could be employed together to produce a similarly priced multi-slice system.

9.2.1 What is your N?

The first, most important, consideration in designing your multi-slice experimental setup is: What will your N be?

In academic research, most researchers we know think that an N of 1 is for 1 slice from 1 animal exposed to 1 experimental protocol (including both stimulation and perfusion solutions).

So if 1 slice each was obtained from 2 animals, and these two slices were exposed to the SAME stimulation protocol and the SAME perfusion solutions, this would be a 'Strict' N of 2.

And if 2 slices were obtained from 1 animal, and these two slices were exposed to the DIFFERENT stimulation protocols and DIFFERENT perfusion solutions, this would also be a 'Strict' N of 2.

Furthermore, if 2 slices were obtained from 1 animal, and these two slices were exposed to the SAME stimulation and DIFFERENT perfusion solutions (say different concentrations of antagonists), this would also be a 'Strict' N of 2.

However, if 2 slices were obtained from 1 animal, and these two slices were exposed to the SAME stimulation protocol and the SAME perfusion solutions, some would consider this to be an N of 2, and others would consider it a 'Strict' N of 1.

In general we at WinLTP Ltd. do not favor exposing multiple slices obtained from 1 animal to the SAME stimulation protocol and the SAME perfusion solutions, and WinLTP was not designed for this approach.

WinLTP was designed to provide, if necessary, completely DIFFERENT stimulation protocols and perfusion solutions. Four instances of WinLTP can be run on one computer, and each of these WinLTPs can control completely separate stimulation and perfusion solutions. And taking 4 slices from 1 animal and exposing them to DIFFERENT stimulation and perfusion solutions is clearly more difficult to do than taking 4 slices from 1 animal and exposing them to the SAME stimulation and perfusion solutions.

That said, if your experiment includes testing DIFFERENT concentrations of agonist or antagonist, then running several slices obtained from 1 animal and exposing these slices to the SAME stimulation protocol but DIFFERENT perfusion solutions, this would be an easy way to run many slices side by side, have N equal to the number of slices tested, and yet change all the solutions easily and manually without requiring automated perfusion control.

WinLTP just tries to provide the tools for all situations, including completely independent control of stimulation and perfusion solutions.

9.2.3 Automated Perfusion Problems – Dead Volume in Extracellular Slice Experiments

Before we go into setting up the automated perfusion control, it is important to discuss what we see as potentially the main problem of automatic perfusion control in extracellular slice experiments, namely dead volume.

For extracellular slice experiments, when aerated physiological saline has stayed in polyethylene or tygon tubing for a ‘long’ period of time before being perfused on slices, oxygen can diffuse across the tubing wall. In addition this loss of oxygen/carbon dioxide, can also cause changes in pH if a standard bicarbonate buffer is used to perfuse the slice.

We honestly don’t know how bad this lack of oxygen and pH change is, but it depends on the thickness of the tubing wall, what the tubing is made of, the volume of the solution that just ‘sits’ there before being perfused onto the slice (the dead volume), and how long it ‘sits’ there. More on how to reduce this dead volume is presented below.

For extracellular slice experiments, the important dead volume for the standard perfusion system is between the aerated perfusion bottle and the manifold (shown in red in Fig 9.2.4.1A), and for the pre-flush perfusion system between the valve and the manifold (shown in red in Fig 9.2.4.1B).

Estimates of dead volume for 1.32” and 1.00mm inner diameter tubing for the pre-flush system (only between the T-fitting, in the valve, and to the manifold) for 9cm (for a 4 channel system, and 16cm (for an 8-channel system) are shown below. The 1/32” tubing volume (0.05 to 0.08 ml) is pretty small for a larger 1 ml chamber, but maybe a bit too substantial for a 0.3 ml chamber.

9 cm of 1/32" tubing has a DeadVolume = 0.05 ml
16 cm of 1/32" tubing has a DeadVolume = 0.08 ml

9 cm of 1.0mm tubing has a DeadVolume = 0.07 ml
16 cm of 1.0mm tubing has a DeadVolume = 0.13 ml

9.2.4 Setting up Standard and Pre-Flush Extracellular Slice Automated Perfusion Systems

To use automated perfusion control for extracellular slice experiments, you have to next decide whether you want to use a the standard perfusion system with 1 valve/line, or a pre-flush system like AutoMate Scientific's AutoPrime system, with 2 valves/line, and you have to choose the number of perfusion channels you need (see below for further discussion of standard and pre-flush perfusion systems).

WinLTP 2.01 Slow0 Perfusion Changes offers the following choices for standard versus pre-flush systems, and number of perfusion channels (see also Fig. 9.2.7.1):

1) 4-channel, PreFlush, 2 valves/line system (low-speed digital Port1)
2) 8-channel, PreFlush, 2 valves/line system (low-speed digital Port1 and Port2)
3) 8-channel, Standard, 1 valve/line system (low-speed digital Port1)
4) 16-channel, Standard, 1 valve/line system (low-speed digital Port1 and Port2)
5) 15-channel, Standard, 1 valve/line system (binary 4-bits on low-speed Port1)
6) 16-channel, Standard, 1 valve/line system (binary 4-bits on low-speed Port1)

For Slow0 Perfusion Change control of extracellular slice perfusion systems, the National Instruments M- or X-Series boards use 8 or 16 digital outputs for 1 digital output controlling 1 valve. We used this control first to be able to turn on more than one valve at a time – which is necessary for implementing the pre-flushing systems. Slow0 and Slow1 Perfusion Change also support digitial control of 15 or 16 channels using 4-bit binary output, but these outputs do not allow for more than one valve to be on at a time.

The 1 digital output controlling 1 valve method should be able to control the following perfusion systems:

1) AutoMate Scientific's
   ValveLink8.2® 4-channel Pinch Valve Perfusion System
   ValveLink8.2® 8-channel Pinch Valve Perfusion System
   ValveLink®16 16-channel Pinch Valve Perfusion System
2) BioscienceTools'
   8-channel PinchValve System (one PC-16 and one PS-V8)
   16-channel PinchValve System (one PC-16 and two PS-V8s)
3) Warner's
   VC-8P 8-channel Complete Perfusion System
4) ALA Scientific's
   VC3-4PG 4-channel perfusion system
   VC3-8PG 8-channel perfusion system

Connect up the digital output from your National Instruments M- or X-Series board to your perfusion system as follows:

1) For the 4-channel pre-flush system (2 valves/line), use these Port1 outputs:

Port1, Bit0	P1.0	PFI 0	Chamber Valve, Ch 1
Port1, Bit1	P1.1	PFI 1	Chamber Valve, Ch 2
Port1, Bit2	P1.2	PFI 2	Chamber Valve, Ch 3
Port1, Bit3	P1.3	PFI 3	Chamber Valve, Ch 4
Port1, Bit4	P1.4	PFI 4	Flush Valve, Ch 1
Port1, Bit5	P1.5	PFI 5	Flush Valve, Ch 2
Port1, Bit6	P1.6	PFI 6	Flush Valve, Ch 3
Port1, Bit7	P1.7	PFI 7	Flush Valve, Ch 4

2) For the 8-channel pre-flush system (2 valves/line), use Port1 and Port2 outputs:

Port1, Bit0	P1.0	PFI 0	Chamber Valve, Ch 1
Port1, Bit1	P1.1	PFI 1	Chamber Valve, Ch 2
Port1, Bit2	P1.2	PFI 2	Chamber Valve, Ch 3
Port1, Bit3	P1.3	PFI 3	Chamber Valve, Ch 4
Port1, Bit4	P1.4	PFI 4	Chamber Valve, Ch 5
Port1, Bit5	P1.5	PFI 5	Chamber Valve, Ch 6
Port1, Bit6	P1.6	PFI 6	Chamber Valve, Ch 7
Port1, Bit7	P1.7	PFI 7	Chamber Valve, Ch 8
Port2, Bit0	P2.0	PFI 8	Flush Valve, Ch 1
Port2, Bit1	P2.1	PFI 9	Flush Valve, Ch 2
Port2, Bit2	P2.2	PFI 10	Flush Valve, Ch 3
Port2, Bit3	P2.3	PFI 11	Flush Valve, Ch 4
Port2, Bit4	P2.4	PFI 12	Flush Valve, Ch 5
Port2, Bit5	P2.5	PFI 13	Flush Valve, Ch 6
Port2, Bit6	P2.6	PFI 14	Flush Valve, Ch 7
Port2, Bit7	P2.7	PFI 15	Flush Valve, Ch 8

3) For the 8-channel standard system (1 valve/line), use these Port1 outputs:

Port1, Bit0	P1.0	PFI 0	Chamber Valve, Ch 1
Port1, Bit1	P1.1	PFI 1	Chamber Valve, Ch 2
Port1, Bit2	P1.2	PFI 2	Chamber Valve, Ch 3
Port1, Bit3	P1.3	PFI 3	Chamber Valve, Ch 4
Port1, Bit4	P1.4	PFI 4	Chamber Valve, Ch 5
Port1, Bit5	P1.5	PFI 5	Chamber Valve, Ch 6
Port1, Bit6	P1.6	PFI 6	Chamber Valve, Ch 7
Port1, Bit7	P1.7	PFI 7	Chamber Valve, Ch 8

4) For the 16-channel standard (system (1 valve/line), use Port1 and Port2 outputs:

Port1, Bit0	P1.0	PFI 0	Chamber Valve, Ch 1
Port1, Bit1	P1.1	PFI 1	Chamber Valve, Ch 2
Port1, Bit2	P1.2	PFI 2	Chamber Valve, Ch 3
Port1, Bit3	P1.3	PFI 3	Chamber Valve, Ch 4
Port1, Bit4	P1.4	PFI 4	Chamber Valve, Ch 5
Port1, Bit5	P1.5	PFI 5	Chamber Valve, Ch 6
Port1, Bit6	P1.6	PFI 6	Chamber Valve, Ch 7
Port1, Bit7	P1.7	PFI 7	Chamber Valve, Ch 8
Port2, Bit0	P2.0	PFI 8	Chamber Valve, Ch 9
Port2, Bit1	P2.1	PFI 9	Chamber Valve, Ch 10
Port2, Bit2	P2.2	PFI 10	Chamber Valve, Ch 11
Port2, Bit3	P2.3	PFI 11	Chamber Valve, Ch 12
Port2, Bit4	P2.4	PFI 12	Chamber Valve, Ch 13
Port2, Bit5	P2.5	PFI 13	Chamber Valve, Ch 14
Port2, Bit6	P2.6	PFI 14	Chamber Valve, Ch 15
Port2, Bit7	P2.7	PFI 15	Chamber Valve, Ch 16

P1.0 and PFI 0 are two different ways National Instruments names Port1, Bit 0 and so forth.

For further details see below including information on particular connector boxes, see below.

Perfusion Channel System				Port	PCI6221	CB-68LPR					
Standard8	Standard16	PreFlush4	PreFlush8	Bit#	Pin#	(CA1000)	SCB-68	BNC-2090A	BNC-2110	BNC-2120	USB-6221 BNC
1-chamber	1-chamber	1-chamber	1-chamber	P1.0	11 PFI0	11	PFI0	PFI0	PFI0/P1.0	P1.0/PFI0	P1.0/PFI0
2-chamber	2-chamber	2-chamber	2-chamber	P1.1	10 PFI1	10	PFI1	PFI1	PFI1/P1.1	P1.1/PFI1	P1.1/PFI1
3-chamber	3-chamber	3-chamber	3-chamber	P1.2	43 PFI2	43	PFI2	PFI2	PFI2/P1.2	P1.2/PFI2	P1.2/PFI2
4-chamber	4-chamber	4-chamber	4-chamber	P1.3	42 PFI3	42	PFI3	PFI3	PFI3/P1.3	P1.3/PFI3	P1.3/PFI3
5-chamber	5-chamber	1-flush	5-chamber	P1.4	41 PFI4	41	PFI4	PFI4	PFI4/P1.4	P1.4/PFI4	P1.4/PFI4
6-chamber	6-chamber	2-flush	6-chamber	P1.5	6 PFI5	6	PFI5	PFI5	PFI5/P1.5	P1.5/PFI5	P1.5/PFI5
7-chamber	7-chamber	3-flush	7-chamber	P1.6	5 PFI6	5	PFI6	PFI6	PFI6/P1.6	P1.6/PFI6	P1.6/PFI6
8-chamber	8-chamber	4-flush	8-chamber	P1.7	38 PFI7	38	PFI7	PFI7	PFI7/P1.7	P1.7/PFI7	P1.7/PFI7
	9-chamber		1-flush	P2.0	37 PFI8	37	PFI8	PFI8	PFI8/P2.0	P1.0/PFI8	P2.0/PFI8
	10-chamber		2-flush	P2.1	3 PFI9	3	PFI9	PFI9	PFI9/P2.1	P1.1/PFI9	P2.1/PFI9
	11-chamber		3-flush	P2.2	45 PFI10	45	EXT STROBE	PFI10	EXTSTRB or PF10/P2.2	NA	P2.2/PFI10
	12-chamber		4-flush	P2.3	46 PFI11	46	AI HOLD COMP	PFI11	AI HOLD or PF11/P2.3	NA	P2.3/PFI11
	13-chamber		5-flush	P2.4	2 PFI12	2	CTR 0 OUT	PFI12	CTR 0 OUT or PF12/P2.4	P1.4/PFI12	P2.4/PFI12
	14-chamber		6-flush	P2.5	40 PFI13	40	CTR 1 OUT	PFI13	CTR 1 OUT or PF13/P2.5	P1.5/PFI13	P2.5/PFI13
	15-chamber		7-flush	P2.6	1 PFI14	1	FREQ OUT	PFI14	F OUT or PF14/P2.6	P1.6/PFI14	P2.6/PFI14
	16-chamber		8-flush	P2.7	39 PFI15	39	P2.7,not DGND	PFI15	NA	NA	P2.7/PFI15

1) In the SCB-68, sometimes Pin39 is labelled DGND rather than the correct P2.7
2) There is no P2.7 output with the BNC-2110
3) There are no P2.2, P2.3 or P2.7 outputs with the BNC-2120
4) The SCB-68, BNC-2090A, USB-6221 BNC, and the CB-68LPR (for the CA-1000 enclosure) do have all Port1 and Port2 pinouts

Note that there is no P2.7 (PFI 15) output with the BNC-2110, so you can't use channel 16 with the standard 16-channel system, or channel 8 with the pre-flush 8-channel system.

There are also no P2.2, P2.3 or P2.7 outputs with the BNC-2120, so you can't use channels 11, 12 or 16 with the standard 16-channel system, or channels 3, 4 or 8 with the pre-flush 8-channel system, so the BNC-2120 is not a good choice for an Automatic Perfusion System.

Use the SCB-68, BNC-2090A, USB-6221 BNC, or the CB-68LPR (for the CA-1000 enclosure) if you need to use all outputs from Port1 and Port2.

Then build your perfusion system according to the instructions of the manufacturer. Basically, for the standard perfusion system with 1 valve/line, run the tubing from the reservoir through the pinch valve to the manifold and onto the chamber as shown in Fig 9.2.4.1A. For the pre-flush perfusion system with 2 valves/line (such as the AutoPrime system from AutoMate Scientific), run the tubing from the reservoir to a T-fitting, and run tubing through the Chamber pinch valve to the manifold and onto the chamber, and run the other tubing through the Flush pinch valve and onto the waste receptacle as shown in Fig 9.2.4.1B.

For extracellular slice experiments, all perfusion controllers can control the more inexpensive pinch-valves. The faster Lee-valves etc. are not necessary and are more expensive and difficult to clean and maintain.

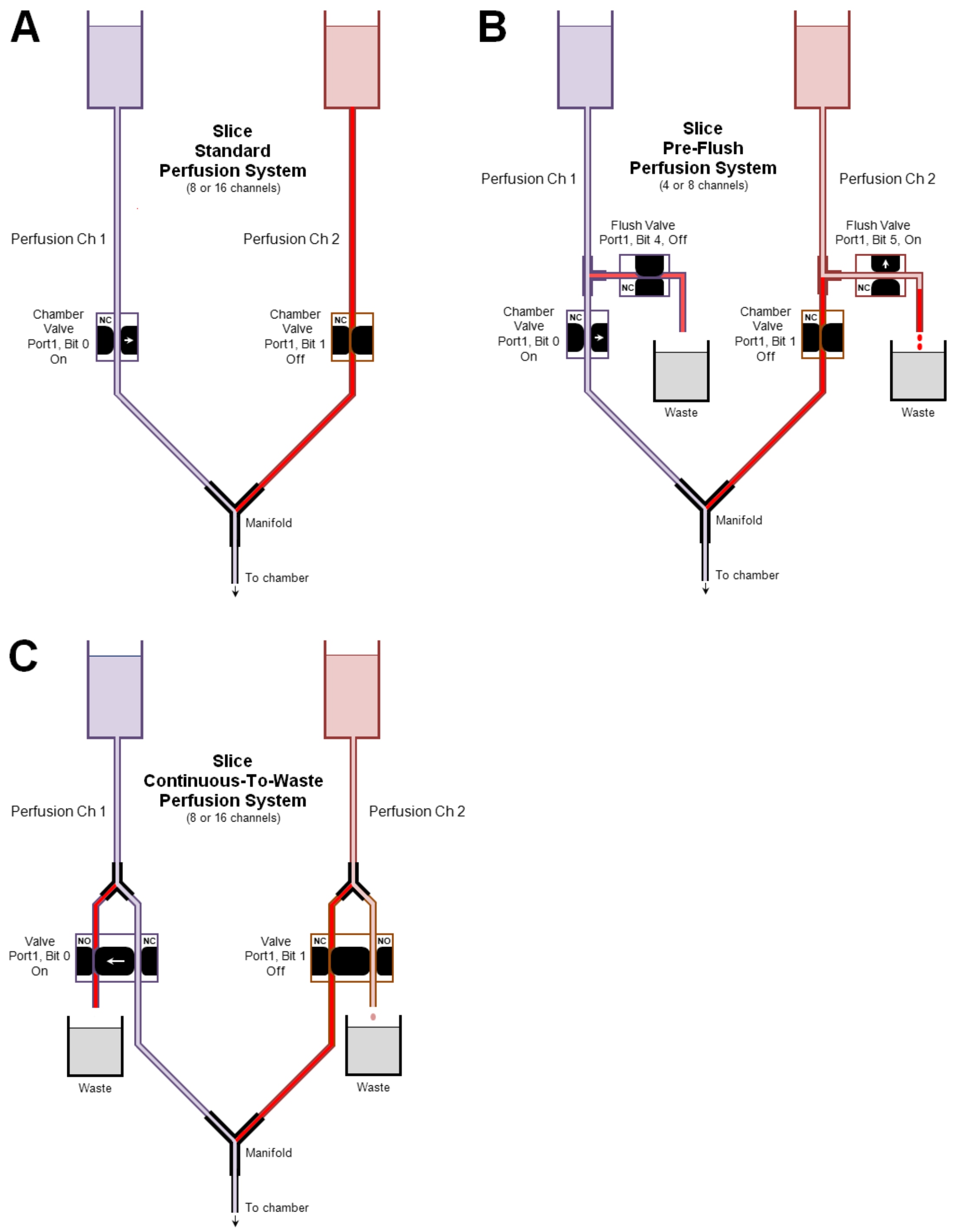
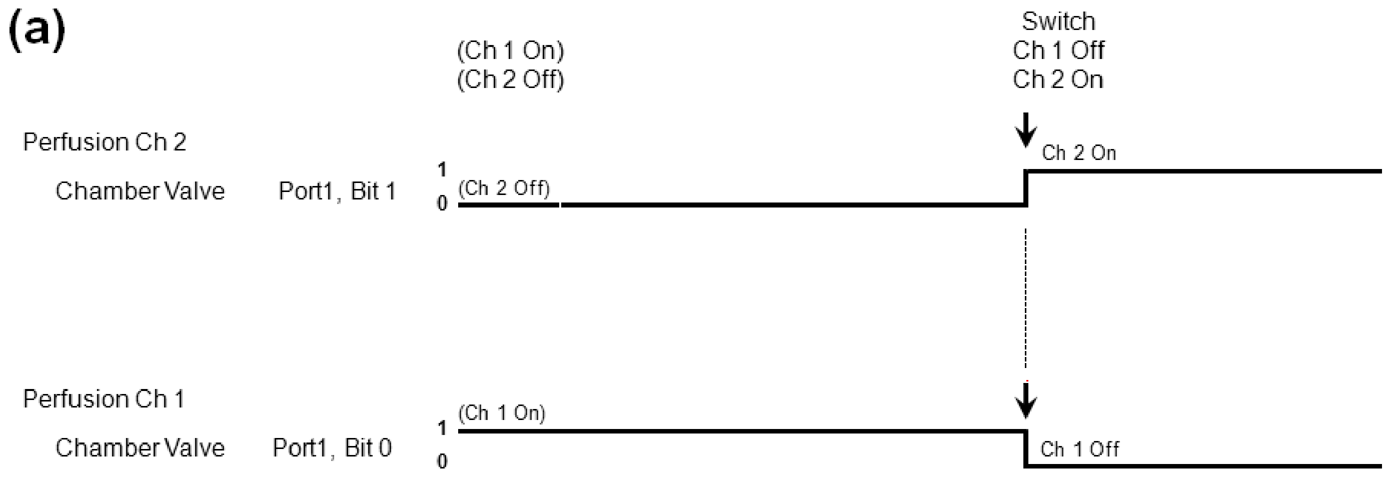


Fig 9.2.4.1. Standard, pre-flush and continuous-to-waste slice perfusion systems. A) The slice standard perfusion system with 1 valve/line. B) The slice pre-flush perfusion system with 2 valves/line for a four line perfusion system. (The Flush Valve has been turned on for a sufficient time to clear the perfusion line from the reservoir to the T-fitting, but the Chamber Valve has not yet been turned on). The dead volumes for the two system lines are indicated in red. Basically the pre-flush system has a dead volume from the T-fitting to the manifold, which could be approximately 10 cm or less. The dead volume of the standard perfusion system could be anywhere from almost 0 cm to 1 meter depending on your configuration. C) A slice continuous-to-waste perfusion system using a Normally Open (NO) – Normally Closed (NC) pinch valve to flush the tubing between the reservoir and the valve. The pinch valve controlling the solution flow between the 'Y' connector and the manifold is Normally Closed, and between the 'Y' connector and waste is Normally Open, and the tubing between the reservoir and the 'Y' connector is aerated. When the perfusion channel is turned on, the continuous flow to waste stops, and aerated solution flows from the 'Y' connector to the manifold.

The digital output switching that causes the switching from Ch 1 to Ch 2 for a standard 1 valve/line perfusion system and for a pre-flush 2 valves/line perfusion system are shown in Fig. 9.2.4.2A and B, respectively.

The standard system starts out with Ch1 on (and Port1, Bit 0 = logical 1), and Ch 2 off (and Port 1 Bit 1 = logical 0). When Ch 1 is switched off, and Ch 2 switched on, Port1, Bit 0 switches to logical 0 and Port1, Bit 1 switches to logical 1.

The 4-channel pre-flush digital outputs for the Chamber Valves are similar to the standard system. However, before switching Ch 1 off, and Ch 2 on, the C2 Flush Valve turns on (by switching Port 1, Bit 5 to logical 1) for a period of time sufficient to put aerated perfusion fluid in the line between the reservoir and the T-fitting.



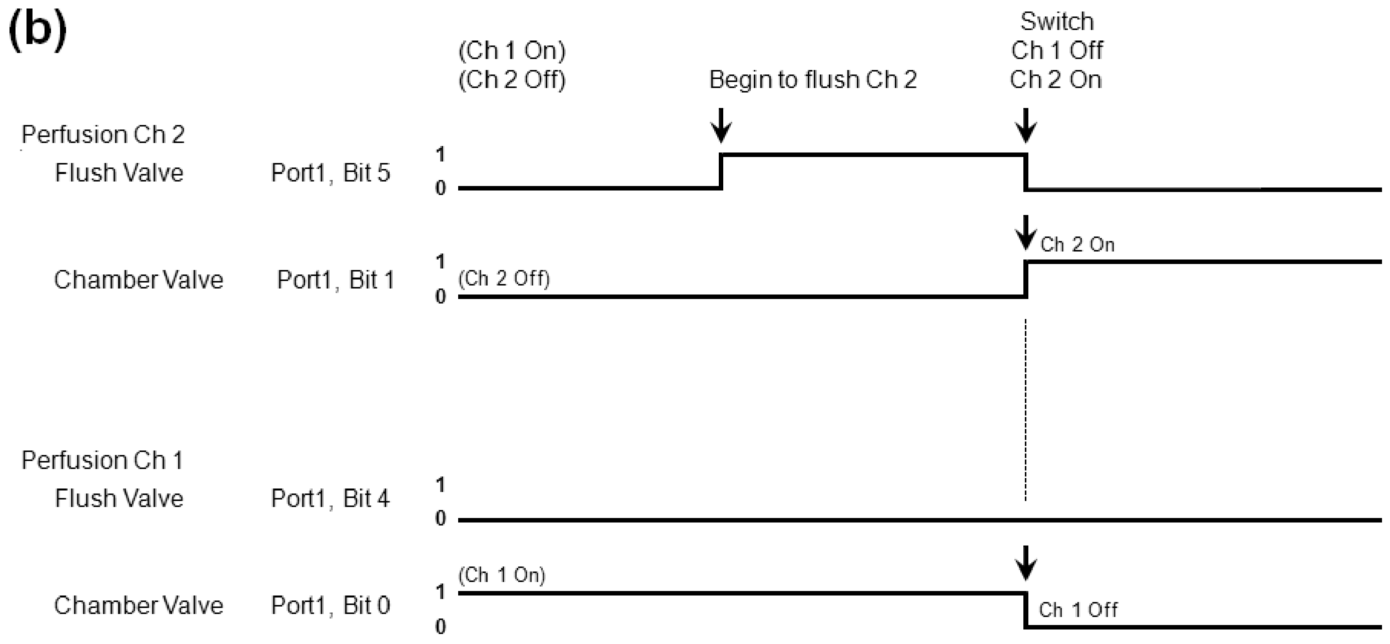


Fig. 9.2.4.2. Switching from Ch 1 to Ch 2 using a Perfusion Controller set up (a) in a standard 8-channel configuration, and (b) in a pre-flush 4-channel configuration. Note how the Ch2 Flush Valve (Bit 5) turns on before the switch from Ch1 to Ch2 and then turns off at the switchover.

9.2.5 Some solutions for reducing Dead Volume

Some solutions for reducing this dead volume could be:

1) Keep the dead volume as small as possible by using as small diameter tubing and as short a length of tubing as possible for the tubing where the dead volume occurs. And whether the dead volume is a problem also depends on how big the bath volume is.

2) Automatically pre-flush much of the dead volume tubing before switching (as with AutoMate Scientific's AutoPrime system).

   Whether you need to use the pre-flush system that requires twice as many valves as the standard system depends on your perfusion requirements. If you are using small ca 60 ml syringe reservoirs without reservoir heating or with reservoir heating (for example using AutoMate Scientific's BubbleStop Syringe Heater) that maybe extends the dead volume from 10 to 15 cm in a standard system, a pre-flush system would not be particularly advantageous. However, if you are using larger 250 and 500 ml bottles as reservoirs (perfusing for hours at 2-5 ml/min) and with pre-heating using a water bath that maybe extends the dead volume from 10 cm to 1 meter in a standard system, a pre-flush system would be advantageous in this situation. Obviously, its up to you to decide. WinLTP just tries to provide the tools.

3) Use clear Teflon tubing where the dead volume occurs.

   Clear Teflon tubing is less permeable to oxygen diffusion of oxygen across the tubing wall than polyethylene or tygon tubing, but we don't know if this decreased permeability makes any real-world difference in a standard system and if it would negate needing a pre-flush system.

4) Possibly re-bubble (re-oxygenate) the perfusion fluid between the reservoir and the chamber. Some researchers at Bristol have a perfusion line break by dripping into a syringe near the chamber, and some bubble the contents of this syringe.

5) Whatever you do, we would strongly suggest empirically testing it. After normal ACSF dead volume less-oxygenated perfusion fluid has 'sat' in the tubing a 'long' time, try switching from normal ACSF fully-oxygenated perfusion fluid to the dead volume less-oxygenated perfusion fluid. And then a) at minimum, see if field EPSP amplitude or slope changes, and (hopefully) b) measure changes in oxygen and/or changes in pH.

9.2.6 Alternatives to using Pre-Flush for Extracellular Slice Experiments

The necessity of using 95% O_2/ 5% CO_2 prohibits running a perfusion line directly from solution reservoir to chamber (as with the Single-Line (Section 9.3) and Double- or Triple-Line/Stepper (Section 9.4) perfusion systems). In Sections 9.2.4 and 9.2.5 we have discussed using a Pre-Flush or AutoPrime system. However, using 2 pinch-valves is a costly solution.

In certain situations, a single valve can be used to.keep the tubing between the reservoir and the pinch valve filled with aerated solution of the correct pH. Note that with these systems, there is still a short length of tubing between the pinch valve and the manifold that is not aerated.

The first method is to use a **Continuous Flow To Waste** system. With some systems that use three way, normally open – normally closed, pinch valves such as those from ALA Scientific and Warner, the tubing between the reservoir and the manifold is placed in the Normally Closed slot of the pinch valve and only opens when that perfusion channel is turned on. The tubing flowing to waste is connected by a 'Y' connector between the reservoir and the pinch valve is placed in the Normally Open slot of the pinch valve and aerated solution normally flows to waste until that perfusion channels is turned on and the Normally Open line is shut (Fig 9.2.4.1C). With the three way Normally Open – Normally Closed pinch valve, either the Normally Open OR the Normally closed line so that only one is open at one time. Provided that the flow in the Normally Open Continuous Flow To Waste line is (just) sufficient to provide aerated solution at the 'Y' connector, and that the solution you are perfusing is relatively inexpensive, this can be a good substitute for the 2 valve pre-flush system.

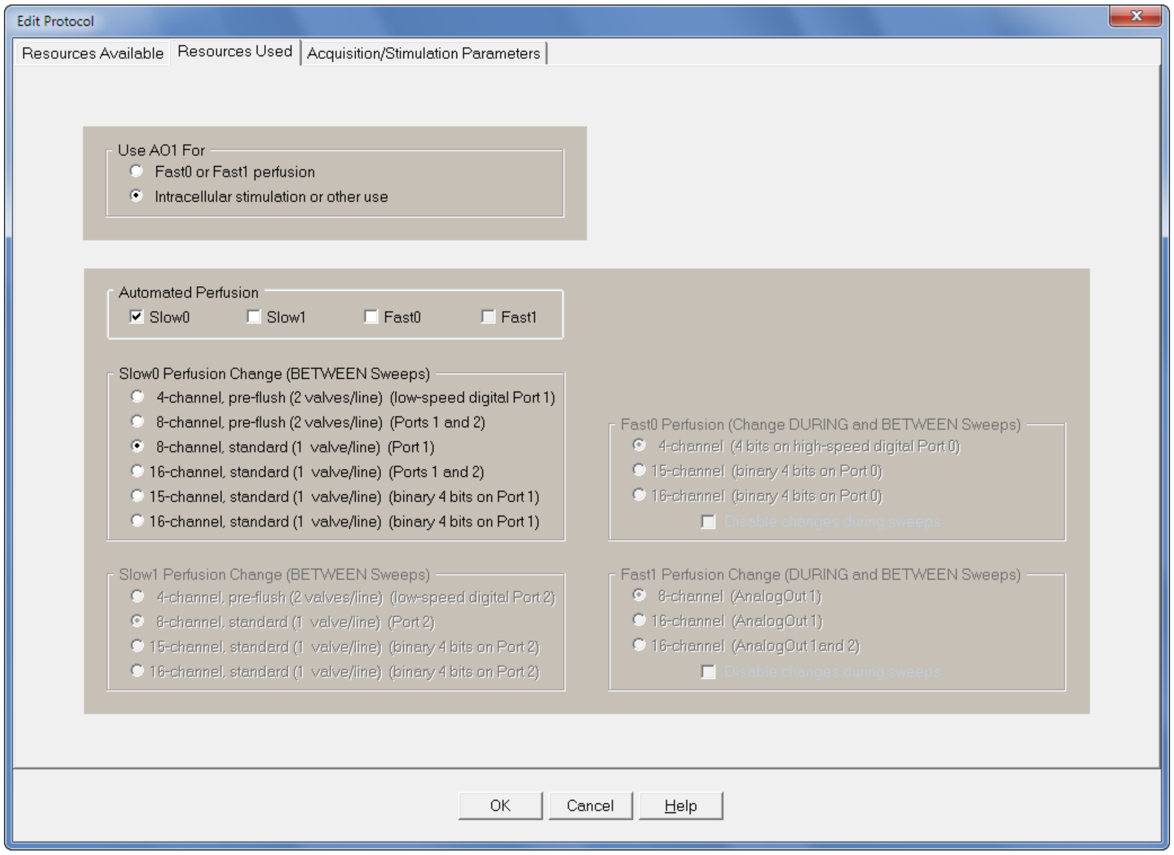
Sometimes, a combination can be used. For example, for an 8 valve system which normally would normally deliver 4 solutions using pre-flush (2 valves/line), you could have three perfusion lines with expensive solutions using pre-flush 2 valves/line, and two perfusion lines with inexpensive solutions using the Continuous Flow To Waste 1 valve/line system.

The second method to get around using an expensive 2 valve/line pre-flush system is to **mount the syringe reservoir directly on top of the pinch valve**, as with Automate systems. This makes the perfusion line from the reservoir to the pinch-valve as short as possible. This can be done if the reservoir (a syringe) is mounted right on top of the pinch valve, as with Automate systems. However, unless you are perfusing room temperature solution, the solution has to be directly warmed by a heater directly surrounding the syringe, such as Automate's BubbleStop heater.

9.2.7 Setting up WinLTP to use your Automated Perfusion System

To use automated perfusion control for extracellular slice experiments in WinLTP, first check the Slow0 Automated Perfusion check box. Then set whether you want to use the standard perfusion system with 1 valve/line, or the pre-flush system with 2 valves/line, and set the number of perfusion channels you need.

You do this by going to the Edit Protocol dialog box (using **File -> Edit** menus), click on the Resources Used tab and then enter the Digitally Controlled Perfusion System dialog box selection (Fig. 9.2.7.1).



Edit Protocol

Resources Available | Resources Used | Acquisition/Stimulation Parameters

Use AO1 For
- ○ Fast0 or Fast1 perfusion
- ● Intracellular stimulation or other use

Automated Perfusion
- [✓] Slow0 [ ] Slow1 [ ] Fast0 [ ] Fast1

Slow0 Perfusion Change (BETWEEN Sweeps)
- ○ 4-channel, pre-flush (2 valves/line) (low-speed digital Port 1)
- ○ 8-channel, pre-flush (2 valves/line) (Ports 1 and 2)
- ● 8-channel, standard (1 valve/line) (Port 1)
- ○ 16-channel, standard (1 valve/line) (Ports 1 and 2)
- ○ 15-channel, standard (1 valve/line) (binary 4 bits on Port 1)
- ○ 16-channel, standard (1 valve/line) (binary 4 bits on Port 1)

Fast0 Perfusion (Change DURING and BETWEEN Sweeps)
- ● 4-channel (4 bits on high-speed digital Port 0)
- ○ 15-channel (binary 4 bits on Port 0)
- ○ 16-channel (binary 4 bits on Port 0)
- [ ] Disable changes during sweeps

Slow1 Perfusion Change (BETWEEN Sweeps)
- ○ 4-channel, pre-flush (2 valves/line) (low-speed digital Port 2)
- ● 8-channel, standard (1 valve/line) (Port 2)
- ○ 15-channel, standard (1 valve/line) (binary 4 bits on Port 2)
- ○ 16-channel, standard (1 valve/line) (binary 4 bits on Port 2)

Fast1 Perfusion Change (DURING and BETWEEN Sweeps)
- ● 8-channel (AnalogOut 1)
- ○ 16-channel (AnalogOut 1)
- ○ 16-channel (AnalogOut 1 and 2)
- [ ] Disable changes during sweeps

OK | Cancel | Help

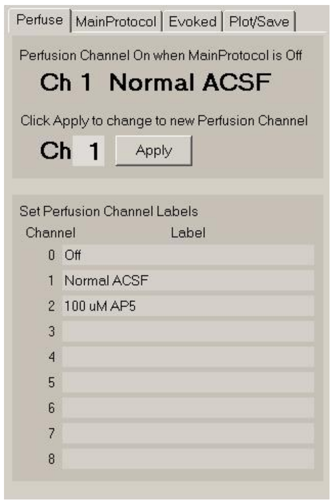
Fig. 9.2.7.1. Choosing the standard perfusion system (1 valve/line), or a pre-flush system (2 valves/line), and choose the number of perfusion channels you need. If you choose the standard 8 channel perfusion system, the Perfuse tabsheet (next to the MainProtocol tabsheet) looks like Fig. 9.2.7.2A.

Next you have to set up the Perfusion Channel labels. In the 'Perfuse' tabsheet next to the 'MainProtocol' tabsheet type in a label next to each perfusion channel to be used, such as 'Normal ACSF' next to channel '1' (Fig. 9.2.7.2A).

Also change the perfusion 'Ch' number you want to be on when the MainProtocol is Off and click the 'Apply' button to load it in (Fig. 9.2.7.2A).

And if you are using the pre-flush system, change the 'Flush Time' to a value sufficient for the tubing from the reservoir to the T-fitting completely change from stale, dead volume less-oxygenated fluid to new, freshly oxygenated fluid (Fig 9.2.7.2.A2).

A1



Perfusion Channel On when MainProtocol is Off

Ch 1 Normal ACSF

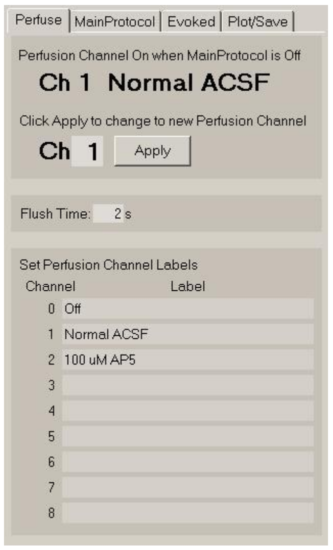
Click Apply to change to new Perfusion Channel

Ch 1

Set Perfusion Channel Labels

Channel	Label
0	Off
1	Normal ACSF
2	100 uM AP5
3	
4	
5	
6	
7	
8	

A2



Perfusion Channel On when MainProtocol is Off

Ch 1 Normal ACSF

Click Apply to change to new Perfusion Channel

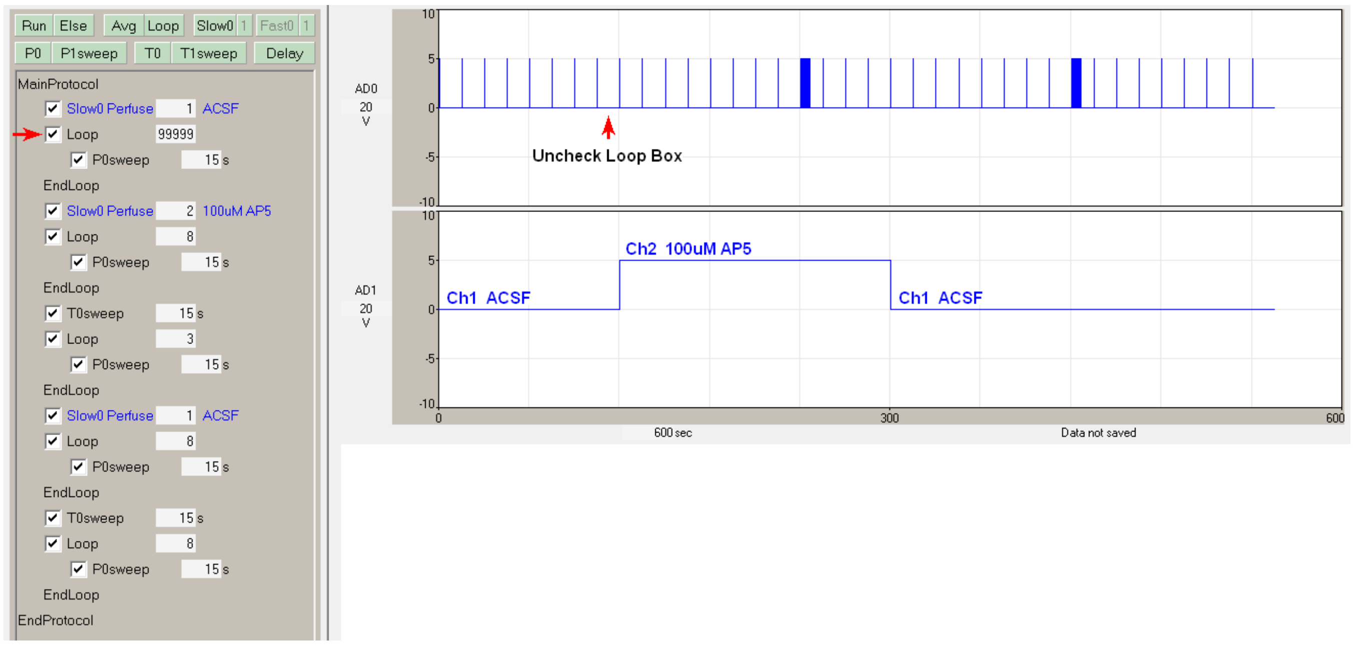
Ch 1

Flush Time: 2 s

Set Perfusion Channel Labels

Channel	Label
0	Off
1	Normal ACSF
2	100 uM AP5
3	
4	
5	
6	
7	
8	

B



```
MainProtocol
   ☑ Slow0 Perfuse    1  ACSF
   ☑ Loop         99999
      ☑ P0sweep       15 s
   EndLoop
   ☑ Slow0 Perfuse    2  100uM AP5
   ☑ Loop             8
      ☑ P0sweep       15 s
   EndLoop
   ☑ T0sweep      15 s
   ☑ Loop             3
      ☑ P0sweep       15 s
   EndLoop
   ☑ Slow0 Perfuse    1  ACSF
   ☑ Loop             8
      ☑ P0sweep       15 s
   EndLoop
   ☑ T0sweep      15 s
   ☑ Loop             8
      ☑ P0sweep       15 s
   EndLoop
EndProtocol
```

Fig. 9.2.7.2. A) Setting the Perfusion Channel Labels for those channels that will be used. Note that A2 contains a FlushTime edit (in seconds) to change the duration of pre-flushing the new perfusion line. B) Using the Perfuse Statement in the Sequential Protocol Script (left), and the digital output of the S0 pulses in AD0, and the P1.1 channel 2 digital output in AD1 that comes On when the 'Perfuse [2] 100 uM AP5' statement is reached. See **SequentialPerfusion.pro**. To use connect DigitalOut0 (P0.0 or S0) to AnalogIn0 (AI0 or AD0), and DigitalOut P1.1 to AnalogIn1 (AI1 or AD1).

The experimental log output for the protocol running in Fig. 9.2.7.2 is shown in Fig. 9.2.7.3. Each perfusion change is clearly shown by time, channel number and channel label (eg “Ch2 100um AP5”), and by then next sweep following the perfusion change.

```
13:36:06.6       0.0 Start MainProtocol
13:36:06.6       0.0 Slow0 Perfuse  Ch 1  ACSF  is on
13:36:07.1       0.5 Slow0 Perfuse  Ch 1  ACSF
13:36:07.1       0.5   Enter Loop0
13:36:07.1       0.5     1D200057.P0, First Loop sweep
13:38:07.1    2:00.5   Leave Loop0
13:38:07.1    2:00.5 Slow0 Perfuse  Ch 2  100uM AP5
13:38:07.1    2:00.5   Enter Loop0
13:38:07.1    2:00.5     1D200068.P0, First Loop sweep
13:40:07.1    4:00.5   Leave Loop0
13:40:07.1    4:00.5   1D200076.T0, Next sweep after end of Loop
13:40:22.1    4:15.5   Enter Loop0
13:40:22.1    4:15.5     1D200077.P0, First Loop sweep
13:41:07.1    5:00.5   Leave Loop0
13:41:07.1    5:00.5 Slow0 Perfuse  Ch 1  ACSF
13:41:07.1    5:00.5   Enter Loop0
13:41:07.1    5:00.5     1D200080.P0, First Loop sweep
13:43:07.1    7:00.5   Leave Loop0
13:43:07.1    7:00.5   1D200088.T0, Next sweep after end of Loop
13:43:22.1    7:15.5   Enter Loop0
13:43:22.1    7:15.5     1D200089.P0, First Loop sweep
13:45:22.1    9:15.5   Leave Loop0
13:45:22.1    9:15.5 Stop MainProtocol
```

Fig. 9.2.7.3. The experimental log output for the protocol running in Fig. 9.2.7.2.

Then go to the Protocol Builder (left side of Fig. 9.2.7.2B), click on the ‘Perfuse’ button with the left mouse button, drag down the mouse cursor to the appropriate between-line location in the Sequential Protocol Script, and release the left mouse button to insert the ‘Perfuse’ statement in the Sequential Protocol Script (see red lines).

Then click on the Perfuse Channel Number field and increment/decrement to the desired Channel Number. Note that the Perfusion Label to the right of the number field is the Perfusion Label set in the Perfuse tabsheet.

That’s about it.

In a future 2.xx version we will adding a ‘SubProtocol’ capability to the Protocol Builder scripting to condense repeating statements and make the script clearer (Fig. 9.2.7.4). (This ‘**Sub**Protocol’ is similar to a subroutine in Basic, and this is why there actually is a ‘**Main**Protocol’!)

```
MainProtocol
   [x] Slow0 Perfuse         [   1]  ACSF
   [ ] Sub0            [99999]
   [x] Slow0 Perfuse         [   2]  100uM AP5
   [x] Sub0            [   2]
   [x] T0sweep               [   2]s
   [x] Sub0            [   5]
   [x] Slow0 Perfuse         [   1]  ACSF
   [x] Sub0            [   4]
   [x] T0sweep               [   2]s
   [x] Sub0            [   6]
EndProtocol

[x] SubProtocol0 [y] [             ]
   [x] Loop       [    y]
      [X] P0sweep    [    2]s
   EndLoop
EndSubProtocol
```

Fig 9.2.7.3. In the future, the Sequential Perfusion Protocol will look much clearer with SubProtocols added.

The protocol in Fig. 9.2.7.4 is the same protocol in Fig. 9.2.7.2B except that the repetitive

```
[x] Loop       [    y]
   [X] P0sweep    [    2]s
EndLoop
```

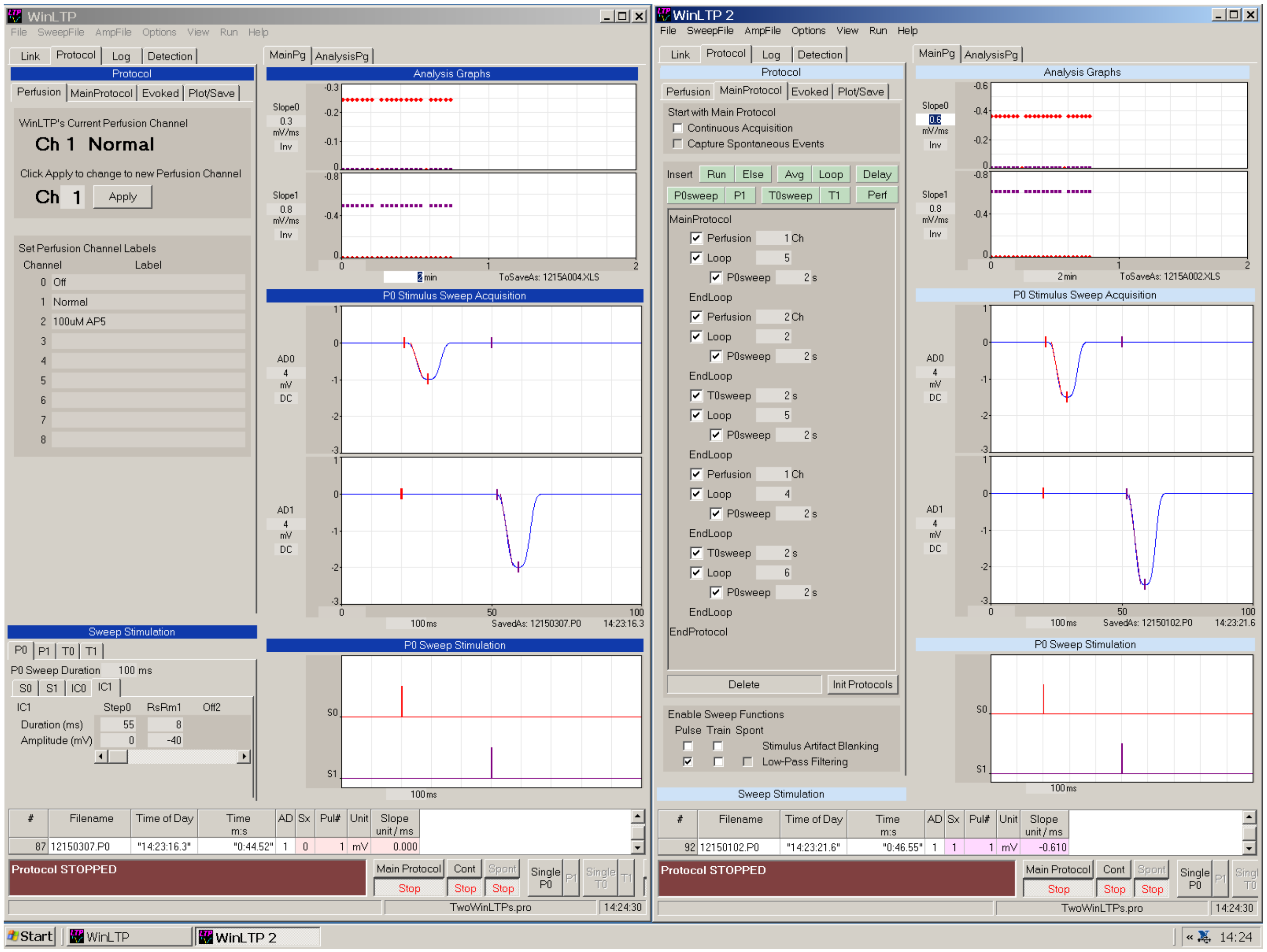
Lines are placed **once** in the SubProtocol0 rather than repeated six times in the MainProtocol.

So in LTP (or similar) experiments with this automated perfusion control, once you determine the correct extracellular stimulation strength, and once you unclick the Baseline Loop once the baseline is stable (Figs. 7.1.1.1D and 9.2.7.2B), all stimulation and perfusion changes will be automatically performed – and you can go off and read a paper in peace, reanalyze yesterday's data in peace, run another slice, or go the pub.

9.2.8 Running Many WinLTP Programs Simultaneously (on one computer)

In addition to Sequential Protocol Scripts and automated perfusion control, the other important capability of WinLTP to do Automated Multi-Slice Experiments is the ability to realistically Run Many WinLTP Programs At Once (Fig. 9.2.8.1).

Strictly speaking, the ability to realistically Run Many WinLTP Programs At Once is **not** needed if you are running many slices from one animal or one slice from many animals so that all the slices are exposed to the SAME stimulation and perfusion solutions.


WinLTP
File SweepFile AmpFile Options View Run Help
Link Protocol Log Detection
Protocol
Perfusion MainProtocol Evoked Plot/Save
WinLTP's Current Perfusion Channel
Ch 1 Normal
Click Apply to change to new Perfusion Channel
Ch 1 Apply
Set Perfusion Channel Labels
Channel Label
0 Off
1 Normal
2 100uM AP5
MainPg AnalysisPg
Analysis Graphs
Slope0 0.3 mV/ms Inv
Slope1 0.8 mV/ms Inv
2 min ToSaveAs: 1215A004.XLS
P0 Stimulus Sweep Acquisition
AD0 4 mV DC
AD1 4 mV DC
100 ms SavedAs: 12150307.P0 14:23:16.3
Sweep Stimulation
P0 P1 T0 T1
P0 Sweep Duration 100 ms
S0 S1 IC0 IC1
IC1 Step0 RsRm1 Off2
Duration (ms) 55 8
Amplitude (mV) 0 -40
P0 Sweep Stimulation
S0
S1
100 ms
Protocol STOPPED
Main Protocol Cont Spont Single P0 P1 Single T0 T1
Stop Stop Stop
TwoWinLTPs.pro 14:24:30
WinLTP 2
Start with Main Protocol
Continuous Acquisition
Capture Spontaneous Events
Insert Run Else Avg Loop Delay
P0sweep P1 T0sweep T1 Perf
Delete Init Protocols
Enable Sweep Functions
Pulse Train Spont
Stimulus Artifact Blanking
Low-Pass Filtering
Slope0 0.6 mV/ms Inv
Slope1 0.8 mV/ms Inv
2 min ToSaveAs: 1215A002.XLS
100 ms SavedAs: 12150102.P0 14:23:21.6
Start WinLTP WinLTP 2 14:24

MainProtocol
- ☑ Perfusion 1 Ch
- ☑ Loop 5
  - ☑ P0sweep 2 s
- EndLoop
- ☑ Perfusion 2 Ch
- ☑ Loop 2
  - ☑ P0sweep 2 s
- EndLoop
- ☑ T0sweep 2 s
- ☑ Loop 5
  - ☑ P0sweep 2 s
- EndLoop
- ☑ Perfusion 1 Ch
- ☑ Loop 4
  - ☑ P0sweep 2 s
- EndLoop
- ☑ T0sweep 2 s
- ☑ Loop 6
  - ☑ P0sweep 2 s
- EndLoop

EndProtocol

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	Slope unit / ms
87	12150307.P0	"14:23:16.3"	"0:44.52"	1	0	1	mV	0.000

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	Slope unit / ms
92	12150102.P0	"14:23:21.6"	"0:46.55"	1	1	1	mV	-0.610

Fig. 9.2.8.1. Running Many WinLTP Programs Simultaneously (on one computer). The first instance of WinLTP running, **'WinLTP'** on the left (see the program name bar and on the Task Bar), has the normal **white letters on a dark blue Title Bar**, whereas the second instance of WinLTP running, **'WinLTP 2'** on the right (see the program name bar and on the Task Bar), has different **black letters on a light blue Title Bar**.

However, if you are running one slice from one animal, and all the slices are exposed to DIFFERENT stimulation and perfusion solutions (ie using the 'Academic' definition of N), then WinLTP really shines. This is primarily due to the **cost** of one (or 5!!!) WinLTP / National Instruments board data acquisition systems compared to the cost of many other data acquisition systems. This is particularly true for academics buying National Instruments boards in units of 5.

The cost of WinLTP (for 5 copies) is $1350

The cost of National Instruments board and WinLTP Advanced Mode software is:

	No Discounts	10% Academic Discount	25% Academic Discount
PCIe-6321	$619	$557	$464
SHC68-68-EPM (2m) cable	$139	$125	$104
BNC-2090A Connector block	$439	$395	$329
	--------	--------	--------
NI board	$1197	$1077	$898
WinLTP Advanced Mode *	$270	$270	$270
	=====	=====	=====
Total	$1467	$1347	$1168

*The cost of WinLTP Advanced Mode assumes that all 5 copies are being used.

Alternatively you could substitute the BNC-2090A with the CB-68LPR screw terminal connector block (for a $109 non-discounted price) or the BNC-2110 connector block (for a $369 non-discounted price). **Note that the BNC-2110 does not have the P2.7 (PFI 15) digital output, so you can't use channel 16 with the standard 16-channel system, or channel 8 with the pre-flush 8-channel system.**

The fact that one computer can run many WinLTP programs simultaneously is nice (saves approximately $1000 for every extra computer not needed), but is not as crucial as the cost of the WinLTP / National Instruments data acquisition system.

How to run more than one WinLTP Program and one AD board on one computer has been described in detail in Section 2.3.6.

In WinLTP 1.11b there have also been minor improvements for running multiple WinLTP Onine M,X-Series programs on one computer (Fig. 9.2.8.1):

1) The different WinLTP acquisition programs can now be differentiated visually by the color of the 'SectionBars' (eg the Bars for Protocol, Continuous Acquisition, Analysis Graphs sections, etc). In Fig 9.2.8.1, “WinLTP” on the left has the standard white letters on dark blue TitleBars, whereas “WinLTP 2” on the right has black letters on light blue TitleBars.

2) WinLTP's name on the TaskBar and in the program TitleBar is different for different programs running. "WinLTP" appears for the first program (using 'Dev1'), "WinLTP 2" appears for the second program (using 'Dev2', Fig 9.2.8.1).

3) When WinLTP is started, the screen location (if not full screen) is restored to that when last exited. Therefore, on startup, the different WinLTP acquisition programs will be automatically placed in different screen locations.

9.2.9 WinLTP Automated Multi-Slice Experiments – Putting it All Together

Now to put it all together, or at least to start to put it all together.

Several ideas have gone into the WinLTP Automated Mult-Slice Experiment Project:

1) As discussed earlier, for now, we will assume that you are using the 'Strict' definition of N – **one** slice from **one** animal exposed to the **same** experimental protocol (including both stimulation and perfusion solutions), so that if you are using **many** slices from **one** animal you have to expose each slice to a **different** experimental protocol.

   If you are using **many** slices from **one** animal and each slice is exposed to the **same** experimental protocol, there are many fewer problems, and we won't deal with them here.

2) As also discussed earlier, WinLTP Advanced Version software and the National Instruments board are cheap enough to control 1 chamber and perfusion system.

3) A slow migration to multi-slice experiments.

For example, you may first want to try automated perfusion control on your one slice. If you are using a water bath to pre-heat your solution bottles (which so many doing extracellular slice electrophysiology do), all you need to do is raise the water bath a couple of decimetres (for pre-flushing gravity flow), hook up your perfusion system in the pre-flush manner below the water bath, and use the rest of your setup as normal.

Or to start out with, you may first wish to increase the number of slice wells by adding a laminar perfusion chamber on top of your current chamber bath. For optics, you can just swing your microscope from one chamber to another.

Your ultimate systems may look like those in Fig. 9.2.9.1. Fig 9.2.9.1A shows 2 chambers (each controlled by separate perfusion systems (either automated or manual) and by separate WinLTP/ADboard. There are two slices (one slice per well) obtained from the same animal, but will be able to undergo two completely separate experiments. Each slice has one stimulating and one recording electrode, although in the two chamber system (Fig. 9.2.9.1A), it would be easy to increase the recording electrodes to two. Fig. 9.2.9.1B shows 3 chambers with one well/per chamber, and one slice/well. Fig. 9.2.9.2 shows 4 chambers with one well/per chamber, and one slice/well. This is realistically the maximum number of perfusion chambers, slices and manipulators under one microscope.

If you want to prepare slices from 2 animals, you could use the configuration shown in Fig. 9.2.9.3. Here, the two chambers (in purple or orange) are each controlled by separate perfusion systems and ene WinLTP/ADboard. There are two wells per chamber, and each well contains one slice from a different animal.
Alternatively, if you wish to experiment on **many** slices from one animal using one experimental protocol, you could have (in future versions of WInLTP) the configuration shown in Fig. 9.2.9.4. In this case, 1 chamber is controlled by 1 perfusion system and one WinLTP/ADboard, but there are 4 wells per chamber. Note that the current WinLTP 2.01 only has 2 AD channels, but shortly a 2.xx version will have 5 AD channels.

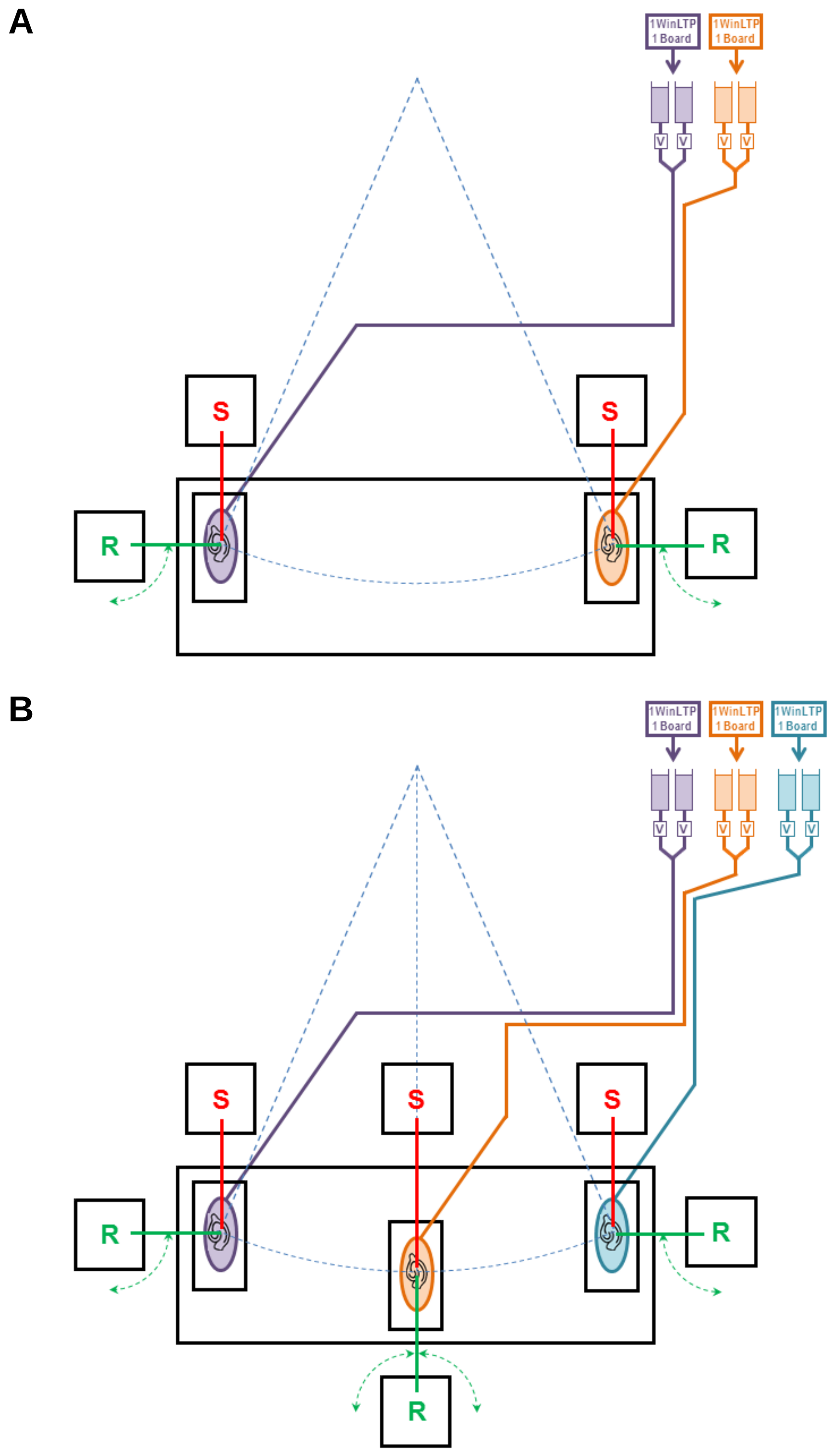


Fig. 9.2.9.1. Perfusion layout for using 1 animal. A) 1 animal, 2 wells, 2 chambers. B) 1 animal, 3 wells, 3 chambers. Each chamber has separate perfusion and control by WinLTP/ADboard. The ‘S’s represent stimulating electrodes and manipulators, and the ‘R’s represent recording electrodes and manipulators. Note that no pump is required if there is only one chamber per perfusion line.

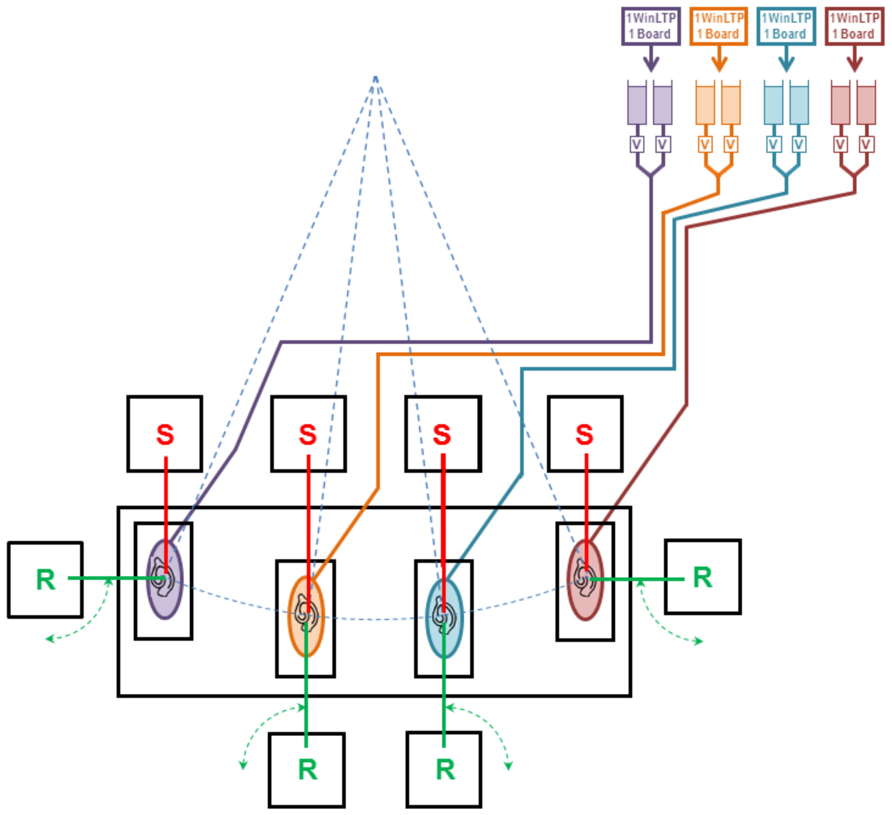


Fig. 9.2.9.2. Perfusion layout for using 1 animal, 4 wells and 4 chambers. Each chamber has separate perfusion and control by WinLTP/ADboard. No pump is required because there is only one chamber per perfusion line.

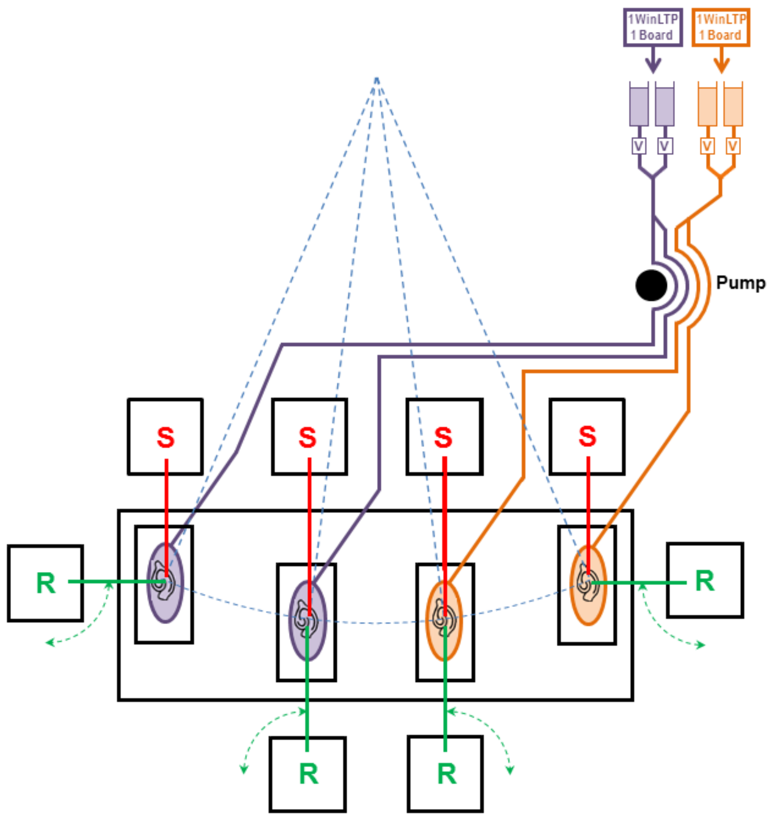


Fig. 9.2.9.3. Perfusion layout for using more than 1 animal, in this case slices from 2 animals, 2 wells and 2 chambers. Note that a pump is required if there is more than one chamber per perfusion line.

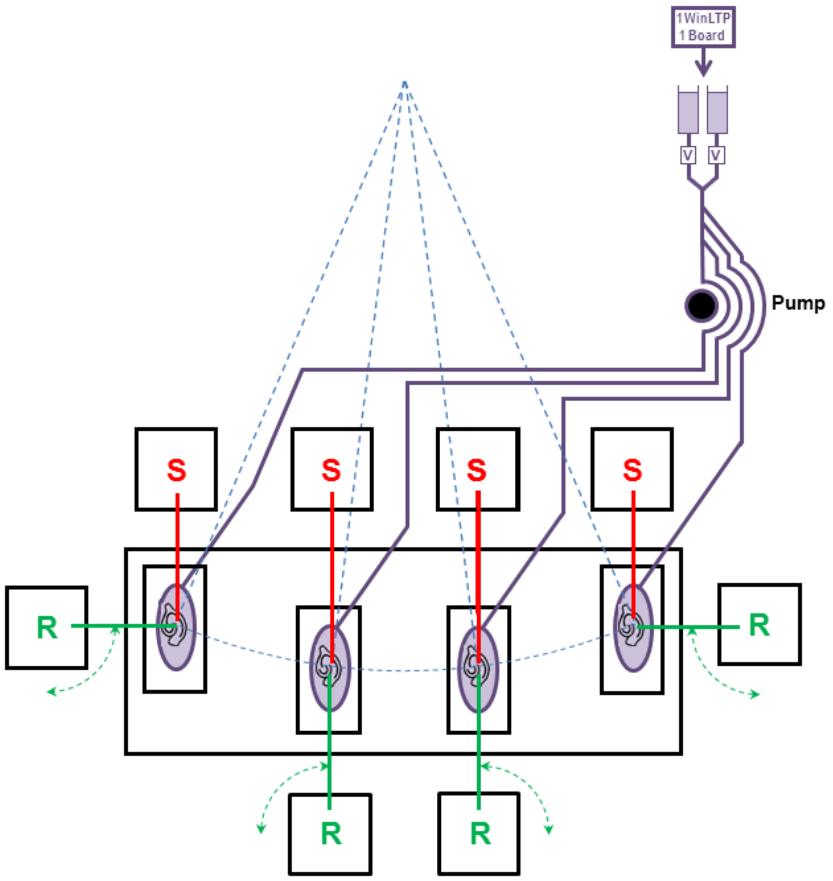


Fig. 9.2.9.4. Perfusion layout if you were using 4 slices from 1 animal. Here you have 1 chamber controlled by 1 perfusion system and one WinLTP/ADboard, but 4 wells per chamber. A pump is required because there is more than one chamber per perfusion line.

9.3 Automated Single-Line Perfusion Patch-Clamp Experiments

9.3.1 Automated Fast Perfusion Changes During and Between Sweeps

In WinLTP 2.00 we have also added the capability of rapidly changing perfusion solutions DURING as well as BETWEEN sweeps to enable automated perfusion control for patch-clamp experiments. The key to this capability is that, In addition to rapidly changing solutions during sweeps, which most electrophysiological data acquisition systems do, WinLTP also uses the Perfuse statement in the Protocol Builder to also change solutions at specific times between sweeps. This capability, along with the changes during the sweep, provides full rapid Automated Perfusion Changes for a patch-clamp experiment, with the one exception of increment/decrement of stimulation values (which we are working on).

In addition to WinLTP, Molecular Devices' pClamp and Heka's PatchMaster also have the basic capability to automatically change perfusion solutions between sweeps. PClamp uses Sequence Keys to set analog or digital outputs, and PatchMaster uses 'Set DAC' or 'Set Digital Word' events within its Protocol Editor.

Slow perfusion changes BETWEEN sweeps, which is appropriate for extracellular slice experiments, has already been discussed in Section 9.2. And the basic perfusion systems from ALA Scientific, Automate Scientific, BioScience Tools and Warner have also been discussed in Section 9.2.

This section deals with rapid or fast perfusion changes DURING and BETWEEN sweeps which is more appropriate for patch-clamp experiments where solutions changes not only need to be made between sweeps, but also include rapid solution changes during the sweep. It is designed to work with the standard perfusion system (not a pre-flush syem) where 8 or 16 tubes go into a manifold which goes a short distance to a ca 100 uM pipette and onto a cell and into the chamber (Fig. 9.3.1.1). Pre-flushing is not usually required in patch-clamp experiments because the solutions are not bicarbonate buffered, and therefore loss of oxygen across the tubing wall and subsequent change in pH are not a problem. In single-line patch-clamping experiments, this biggest problem is the **dead volume is between the manifold and the end of the pipette in the chamber** (shown in red in Fig. 9.3.1.1), because this affects how quickly perfusion solutions can be changed.

Fast Perfusion changes using a single perfusion line involve changes to a new solution lasting typically tens of millseconds up to a second or more. This relatively long perfusion time is due to needing to wash out the dead volume between the manifold and the tip of the pipette in the chamber. For faster solution changes typically down to a millisecond in the new solution, a mechanical stepper device to move between solution pipes is required (Section 9.4). Also, the single-line fast perfusion requires more expensive fast valves such as Lee valves rather than the less expensive pinch valves that are suitable for extracellular and stepper/pinch valve perfusion systems. There are some relatively inexpensive steppers such as the Warner SF-77, which coupled with inexpensive pinch valve controller system, may not cost much more than a fast valve single-line perfusion system, and have solution changes down to a millisecond in the new solution for the fastest piezo steppers. Note: we have not tried the Warner SF-77 and do not recommend, we just suggest that you might want to consider it.

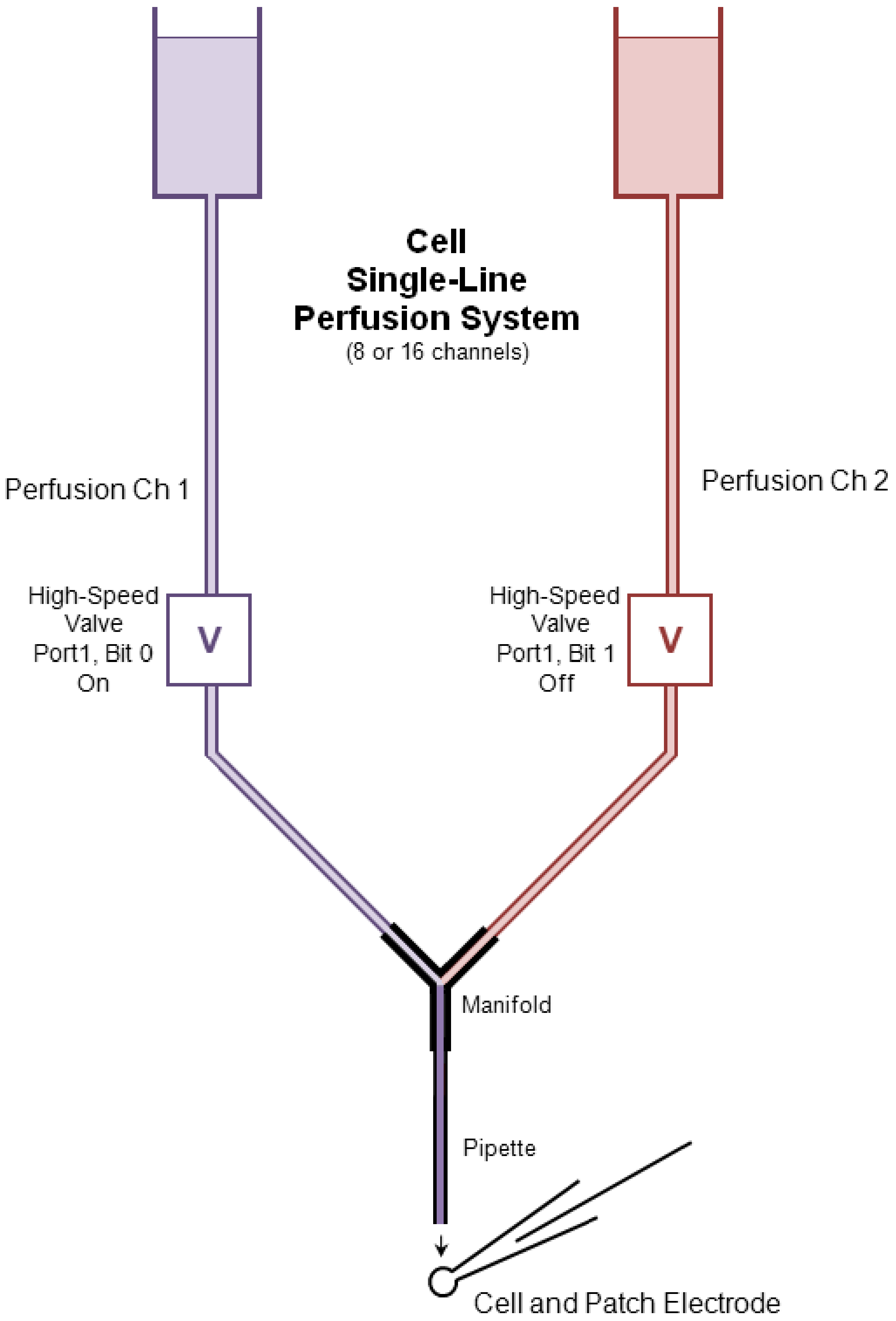


Fig 9.3.1.1. Standard perfusion system setup for single cell, single-line patch-clamp experiments. The crucial dead volume in this case is from the manifold to the tip of the perfusion pipette (shown in dark purple). The main difference between this and the slice standard perfusion system (Fig. 9.2.4.1A) is the use of high-speed valves rather than the slower pinch valves.

The Fast Perfusion changes can be controlled by both the Perfuse statement in the Protocol Builder to change between sweeps, and by changing either digitial or analog stimulation during the sweep to change perfusion during the sweep.

9.3.1.1 Will the Antagonist Rapidly Unbind?

For a typical antagonist/agonist experiment, whether or not the antagonist rapidly unbinds has a large effect on the number of perfusion solutions (and valves) needed to get a good dose-response curve.

If the antagonist remains bound during the application of a solution containing agonist only, then antagonist does not need to be added to the agonist solution to get an accurate dose-response curve. Therefore, you can have separate solutions (and valves) for each agonist solution as well as each antagonist solution.

For example, one eight valve controller is sufficient to do a dose-response curve for 4 agonist concentrations and 4 antagonist concentrations (one of which could be no antagonist, ie ACSF) (Table 9.3.1.1.1).

	Antagonist	Agonist
Ch 1	1	
Ch 2	2	
Ch 3	3	
Ch 4	4	
Ch 5		1
Ch 6		2
Ch 7		3
Ch 8		4

Table 9.3.1.1.1. If the antagonist does not rapidly unbind, eight solutions are sufficient to produce a dose-response curve for 4 agonist concentrations and 4 antagonist concentrations (one of which could be no antagonist, ie ACSF). This can be done with one eight channel controller controlling eight valves.

If the antagonist does not remain bound during the application of the agonist-only solution, then the antagonist has to be added to the agonist solution to get an accurate dose-response curve. Therefore, many more agonist + antagonist solutions (and valves) are required to do a dose-response curve.

For example, to do a dose response curve for 4 agonists and only 3 antagonists, 15 solutions and 15 valves (two eight valve controllers) is required (Table 9.3.1.1.2, see also Fig. 9.3.1.2.4)

	Antagonist	Agonist
Ch 1	1	
Ch 2	2	
Ch 3	3	
Ch 4	1	1
Ch 5	1	2
Ch 6	1	3
Ch 7	1	4
Ch 8	2	1
Ch 9	2	2
Ch 10	2	3
Ch 11	2	4
Ch 12	3	1
Ch 13	3	2
Ch 14	3	3
Ch 15	3	4

Table 9.3.1.1.2. If the antagonist does rapidly unbind, 15 solutions are required to produce a dose-response curve for 4 agonist concentrations and just 3 antagonist concentrations. This requires two eight channel, eight valve controllers, or one 16 channel, 16 valve controller.

9.3.1.2 Control of Fast0 Perfusion by Analog and Digital Output

To use the Single-Line Fast Perfusion, choose the 'Fast0 Perfusion Change During and Between Sweeps' in the automated perfusion control radiobutton group (Fig. 9.3.1.2.1). You first have to decide how you want to use AO1 in the 'Use AO1 For ' radiobutton group. If you do not need to use AO1, then Fast0 Perfusion can use it as well as the thigh 4-bits on the high-speed digital output Port0. Then you can have the following choices (Fig. 9.3.1.2.1A):

1) Controlling 8 channels using AnalogOut 1 (for the ALA Scientific and Automate Scientific and Warner controllers),
2) Controlling 16 channels also using AnalogOut 1 (for the BioScience Tools controller),
3) Controlling 16 channels using AnalogOut 1 and AnalogOut 2 (for the Automate Scientific, ALA Scientific, BioScience Tools and Warner controllers)
4) Controlling 4 channels using 4 bits on the high speed digitial output Port 0 (for Automate, ALA, BioScience Tools and Warner controllers)
5) Controlling 15 channels using 4 bits on the high speed digitial output Port 0 (for the BioscienceTools controller), where if all the bits are off, all channels are off.
6) Controlling 16 channels using 4 bits on the high speed digitial output Port 0 (for the Automate and ALA controllers), where if all the bits are off, Channel 1 is on.

If you are using AnalogOut 1, you obviously cannot be using it for a second patch-clamp stimulation channel. And if you are using AnalogOut 2, you need a National Instruments board that has 4 rather than just 2 analog outputs such as the PCI-6229, PCIe-6323 or USB-6229 BNC. The current version of WinLTP (2.01) does not check to see that AnalogOut 2 is present – if it is not present, it just won't write to it.

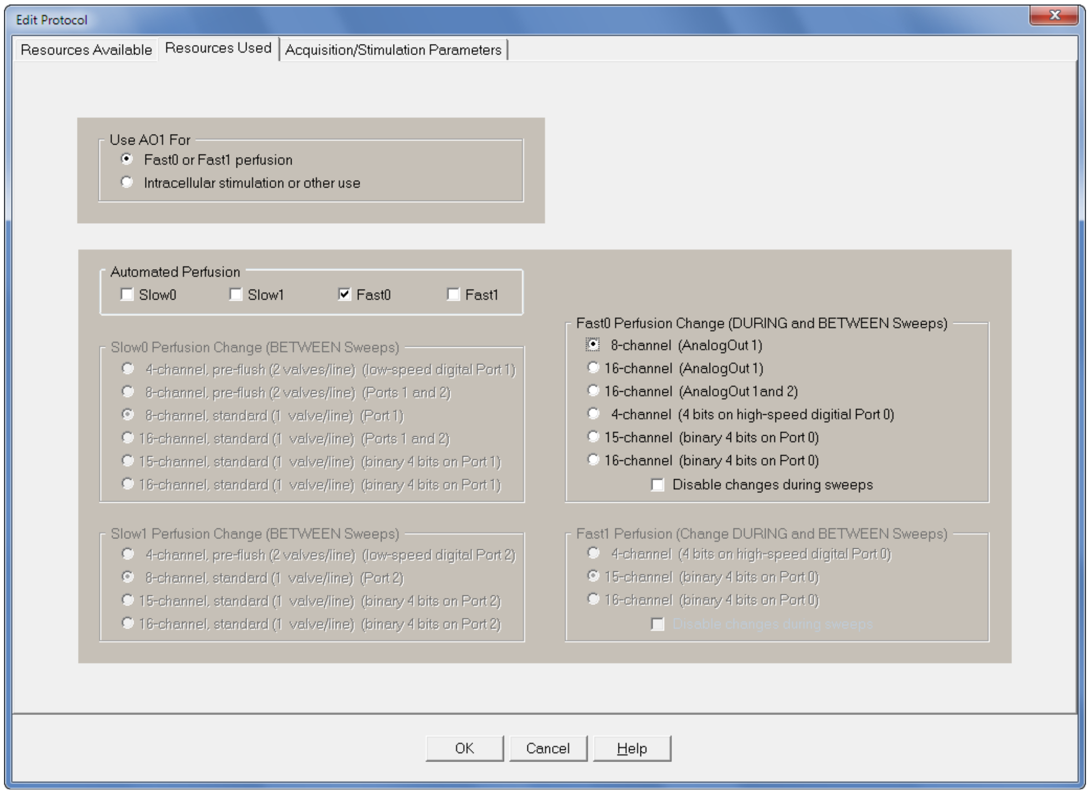
If you do need to use AO1 for something other than fast perfusion, then Fast0 Perfusion cannot use it can only use the high 4-bits on the high-speed digital output Port0. Then you can have the following choices (Fig. 9.3.1.2.1B):

1) Controlling 4 channels using 4 bits on the high speed digitial output Port 0 (for Automate, ALA, BioScience Tools and Warner controllers)
2) Controlling 15 channels using 4 bits on the high speed digitial output Port 0 (for the BioscienceTools controller), where if all the bits are off, all channels are off.
3) Controlling 16 channels using 4 bits on the high speed digitial output Port 0 (for the Automate and ALA controllers), where if all the bits are off, Channel 1 is on.

In order to use AnalogOut 1 or AnalogOut 1 and AnalogOut 2 outputs for Fast Perfusion Change, you first have to set the IC1 Data Type Units to 'V' and the Gain to the appropriate value for your controller (Table 9.3.1.2.1 and Fig. 9.3.1.2.2). If you are using the two analog outputs (1 and 2), you only set the Gain of IC1 for setting AnalogOut 1, AnalogOut 2 in this case uses the Gain of AnalogOut 1.

For the ALA Scientific controller, and in the example in of Fast Perfusion Change shown in Fig. 9.3.1.2.2, a Gain of 0.5, or 0.5 volts per channel, or 4.0 volts to turn on Channel 8 should be used. For the Automate Scientific controller, a Gain of 0.575, or 0.575 volts per channel, or 4.6 volts out to turn on Channel 8 should be used. We have directly tested the 0.575 Gain for the Automate Scientific ValveLine 8.2 controller and it seems to work fine. We have not tested the ALA Scientific VC^{3} 8 controller, but a Gain of 0.5 is straightforward and should work fine.

A



Edit Protocol

Resources Available | Resources Used | Acquisition/Stimulation Parameters

Use AO1 For
- [✓ Fast0 or Fast1 perfusion]
- Intracellular stimulation or other use

Automated Perfusion
- Slow0
- Slow1
- [✓ Fast0]
- Fast1

Slow0 Perfusion Change (BETWEEN Sweeps)
- 4-channel, pre-flush (2 valves/line) (low-speed digital Port 1)
- 8-channel, pre-flush (2 valves/line) (Ports 1 and 2)
- [✓ 8-channel, standard (1 valve/line) (Port 1)]
- 16-channel, standard (1 valve/line) (Ports 1 and 2)
- 15-channel, standard (1 valve/line) (binary 4 bits on Port 1)
- 16-channel, standard (1 valve/line) (binary 4 bits on Port 1)

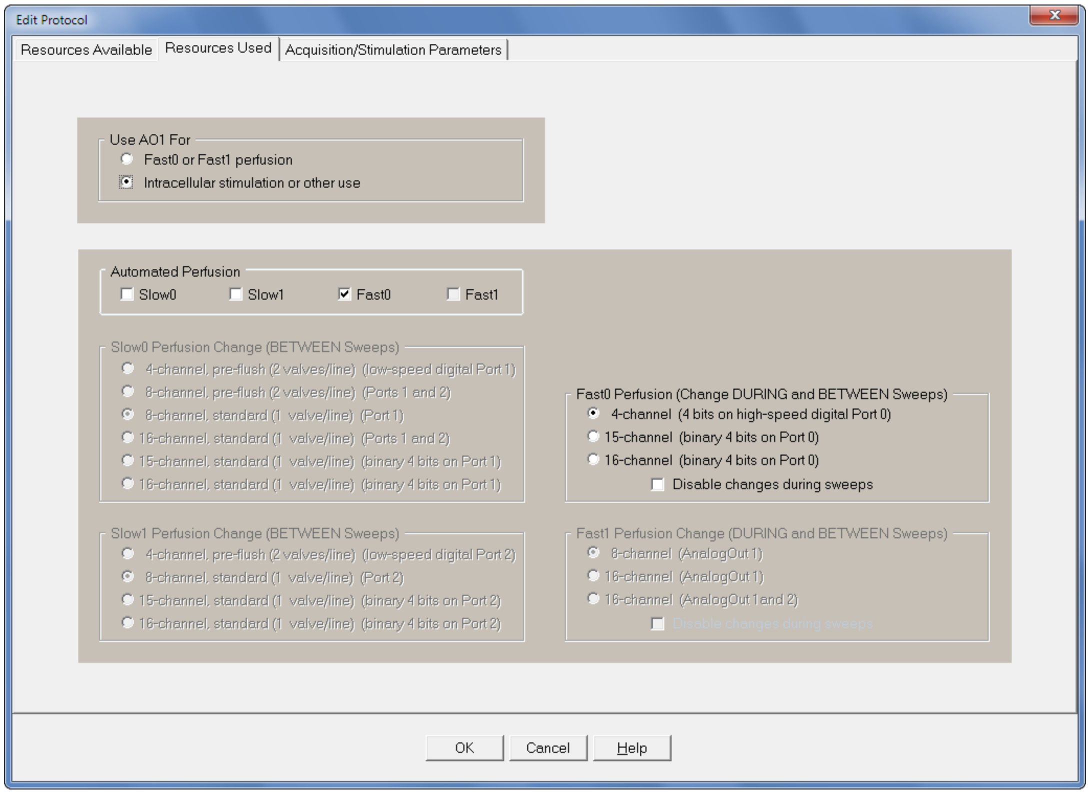
Slow1 Perfusion Change (BETWEEN Sweeps)
- 4-channel, pre-flush (2 valves/line) (low-speed digital Port 2)
- [✓ 8-channel, standard (1 valve/line) (Port 2)]
- 15-channel, standard (1 valve/line) (binary 4 bits on Port 2)
- 16-channel, standard (1 valve/line) (binary 4 bits on Port 2)

Fast0 Perfusion Change (DURING and BETWEEN Sweeps)
- [✓ 8-channel (AnalogOut 1)]
- 16-channel (AnalogOut 1)
- 16-channel (AnalogOut 1 and 2)
- 4-channel (4 bits on high-speed digitial Port 0)
- 15-channel (binary 4 bits on Port 0)
- 16-channel (binary 4 bits on Port 0)
- Disable changes during sweeps

Fast1 Perfusion (Change DURING and BETWEEN Sweeps)
- 4-channel (4 bits on high-speed digital Port 0)
- [✓ 15-channel (binary 4 bits on Port 0)]
- 16-channel (binary 4 bits on Port 0)
- Disable changes during sweeps

OK | Cancel | Help

B



Edit Protocol

Resources Available | Resources Used | Acquisition/Stimulation Parameters

Use AO1 For
- Fast0 or Fast1 perfusion
- [✓ Intracellular stimulation or other use]

Automated Perfusion
- Slow0
- Slow1
- [✓ Fast0]
- Fast1

Slow0 Perfusion Change (BETWEEN Sweeps)
- 4-channel, pre-flush (2 valves/line) (low-speed digital Port 1)
- 8-channel, pre-flush (2 valves/line) (Ports 1 and 2)
- [✓ 8-channel, standard (1 valve/line) (Port 1)]
- 16-channel, standard (1 valve/line) (Ports 1 and 2)
- 15-channel, standard (1 valve/line) (binary 4 bits on Port 1)
- 16-channel, standard (1 valve/line) (binary 4 bits on Port 1)

Slow1 Perfusion Change (BETWEEN Sweeps)
- 4-channel, pre-flush (2 valves/line) (low-speed digital Port 2)
- [✓ 8-channel, standard (1 valve/line) (Port 2)]
- 15-channel, standard (1 valve/line) (binary 4 bits on Port 2)
- 16-channel, standard (1 valve/line) (binary 4 bits on Port 2)

Fast0 Perfusion (Change DURING and BETWEEN Sweeps)
- [✓ 4-channel (4 bits on high-speed digital Port 0)]
- 15-channel (binary 4 bits on Port 0)
- 16-channel (binary 4 bits on Port 0)
- Disable changes during sweeps

Fast1 Perfusion Change (DURING and BETWEEN Sweeps)
- [✓ 8-channel (AnalogOut 1)]
- 16-channel (AnalogOut 1)
- 16-channel (AnalogOut 1 and 2)
- Disable changes during sweeps

OK | Cancel | Help

Fig. 9.3.1.2.1. Choosing standard single-line Fast Perfusion in the automated perfusion control radiobutton group. A) If not using AO1 for anything else, it can be used for Fast Perfusion. If so, you have a choice of, 8 channels using AnalogOut 1, 16 channels using AnalogOut1, 16 channels using AnalogOut1 and AnalogOut2, or 4 channels controlled by 4 bits on the high-speed digital output Port 0, 15 channels using 4 bits Port 0 where if all the bits are off, all channels are off, or 16 channels using 4 bits Port 0 where if all the bits are off, Channel 1 is on. B) If AO1 is used for something else, Fast0 Perfusion can use the three digital output choices.

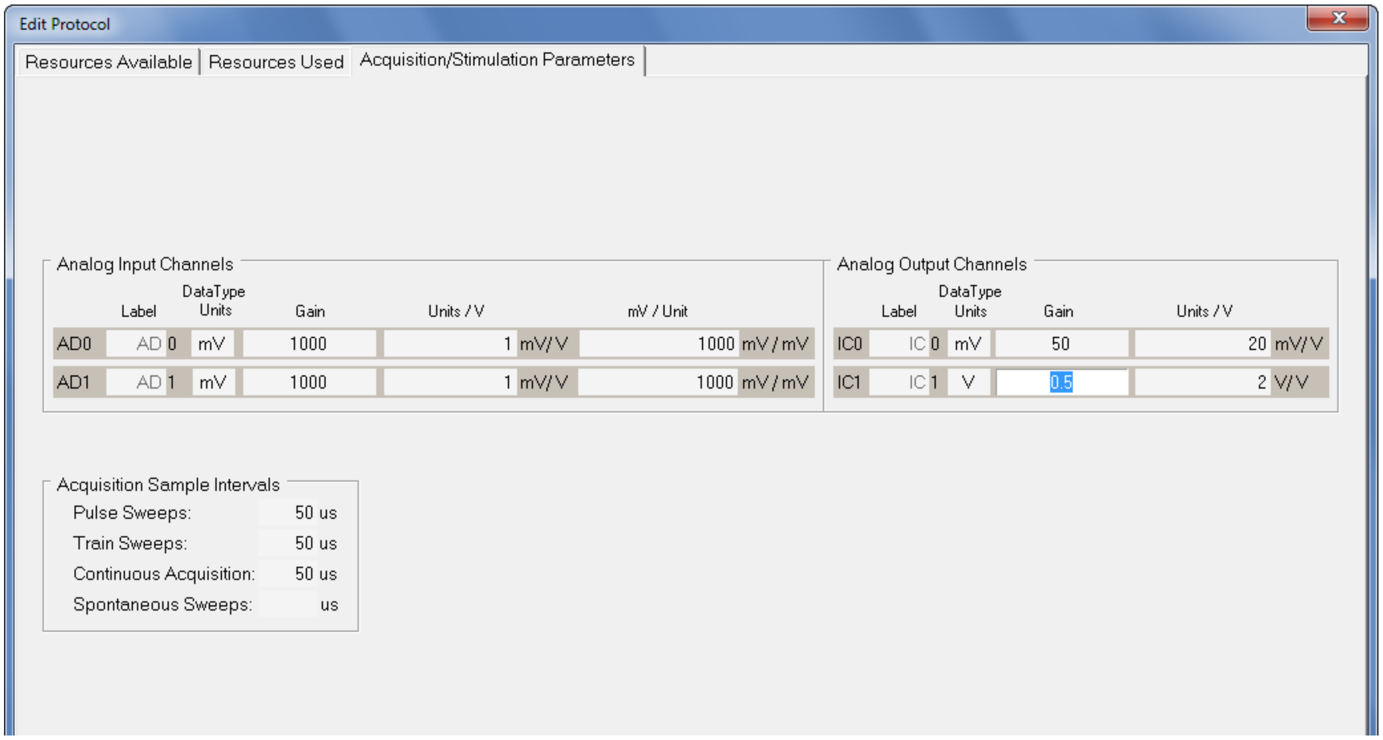
For the BioScience Tools PC-16 a Gain of 0.525, or 0.525 volts per channel, or 8.4 volts for Channel 16 seems like it should work OK. However, we have not tested the PC-16 yet. For some reason, the PC-16 actually has a 0.5 volt offset beginning with Channel 1, so using the Gain of 0.525 to try and give an acceptable linear voltage fit without adding an 0.5 volt offset may not work well, and we may have to program in an offset later.

For the Warner VC-8 controller, if you are using AnalogOut 1 to run one Warner controller for turning on Channels 1 to 8, use a Gain of 1.0, or 1 volt per channel, or 8 volts out to turn on Channel 8. However, if you are using AnalogOut 1 and AnalogOut 2 to run two Warner controllers for turning on a total of 16 channels, use a Gain of 0.5, or 4.0 volts out to turn on Channel 8, and compensate by doubling the AO1 and AO2 voltages by using external amplifiers such as inexpensive op-amps. This is because a Gain of 1.0 for 16 channels would be greater than the 10 volts output capability of AO1, so if you decrease the Gain to 0.5, with 16 channels WinLTP thinks its outputting 8 volts for 16 channels but with the external amplifier doubling, the output is 8 volts for 8 channels on one controller and 8 channels on 2 controllers.

	BiosScience Tools PC-16	WinLTP Voltage	ALA VC^3 8	WinLTP Voltage	AutoMate ValveLink	WinLTP Voltage	Warner VC-8	WinLTP Voltage AO1	AO1 + AO2
WinLTP Gain		0.525		0.5		0.575		1.0	0.5
Chs Off	0.0 - 0.5	0.00	0.0	0.0	0.0	0.00	0	0	0.0
Ch 1	0.5 - 1.0	0.53	0.5	0.5	0.5	0.50	1	1	0.5
Ch 2	1.0 - 1.5	1.05	1.0	1.0	1.1	1.15	2	2	1.0
Ch 3	1.5 - 2.0	1.58	1.5	1.5	1.6	1.73	3	3	1.5
Ch 4	2.0 - 2.5	2.10	2.0	2.0	2.1	2.30	4	4	2.0
Ch 5	2.5 - 3.0	2.63	2.5	2.5	2.7	2.88	5	5	2.5
Ch 6	3.0 - 3.5	3.15	3.0	3.0	3.3	3.45	6	6	3.0
Ch 7	3.5 - 4.0	3.68	3.5	3.5	3.9	4.03	7	7	3.5
Ch 8	4.0 - 4.5	4.20	4.0	4.0	4.6	4.60	8	8	4.0
Ch 9	4.5 - 5.0	4.73							
Ch 10	5.0 - 5.5	5.25							
Ch 11	5.5 - 6.0	5.78							
Ch 12	6.0 - 6.5	6.30							
Ch 13	6.5 - 7.0	6.83							
Ch 14	7.0 - 7.5	7.35							
Ch 15	7.5 - 8.0	7.88							
Ch 16	8.0 - 8.5	8.40							

Table 9.3.1.2.1. The voltages for turning on Channels 1 to 16 for the BioScience Tools' PC-16 controller, for turning on Channels 1 to 8 for the ALA Scientific's VC^3 8 controller, and for turning on 1 to 8 channels for the Automate Scientific's ValveLink 8.2. If you are using AnalogOut 1 to run one Warner VC-8 controller for turning on Channels 1 to 8, use a Gain of 1.0. If you are using AnalogOut 1 and AnalogOut 2 to run two Warner VC-8 controllers for turning on a total of 16 channels, use a Gain of 0.5, and compensate by increasing the AO1 and AO2 output by 2.

No latching is required for WinLTP to maintain a channel in the On state.



Analog Input Channels

Label		DataType Units	Gain	Units / V		mV / Unit	
AD0	AD 0	mV	1000	1	mV/V	1000	mV/mV
AD1	AD 1	mV	1000	1	mV/V	1000	mV/mV

Analog Output Channels

Label		DataType Units	Gain	Units / V	
IC0	IC 0	mV	50	20	mV/V
IC1	IC 1	V	0.5	2	V/V

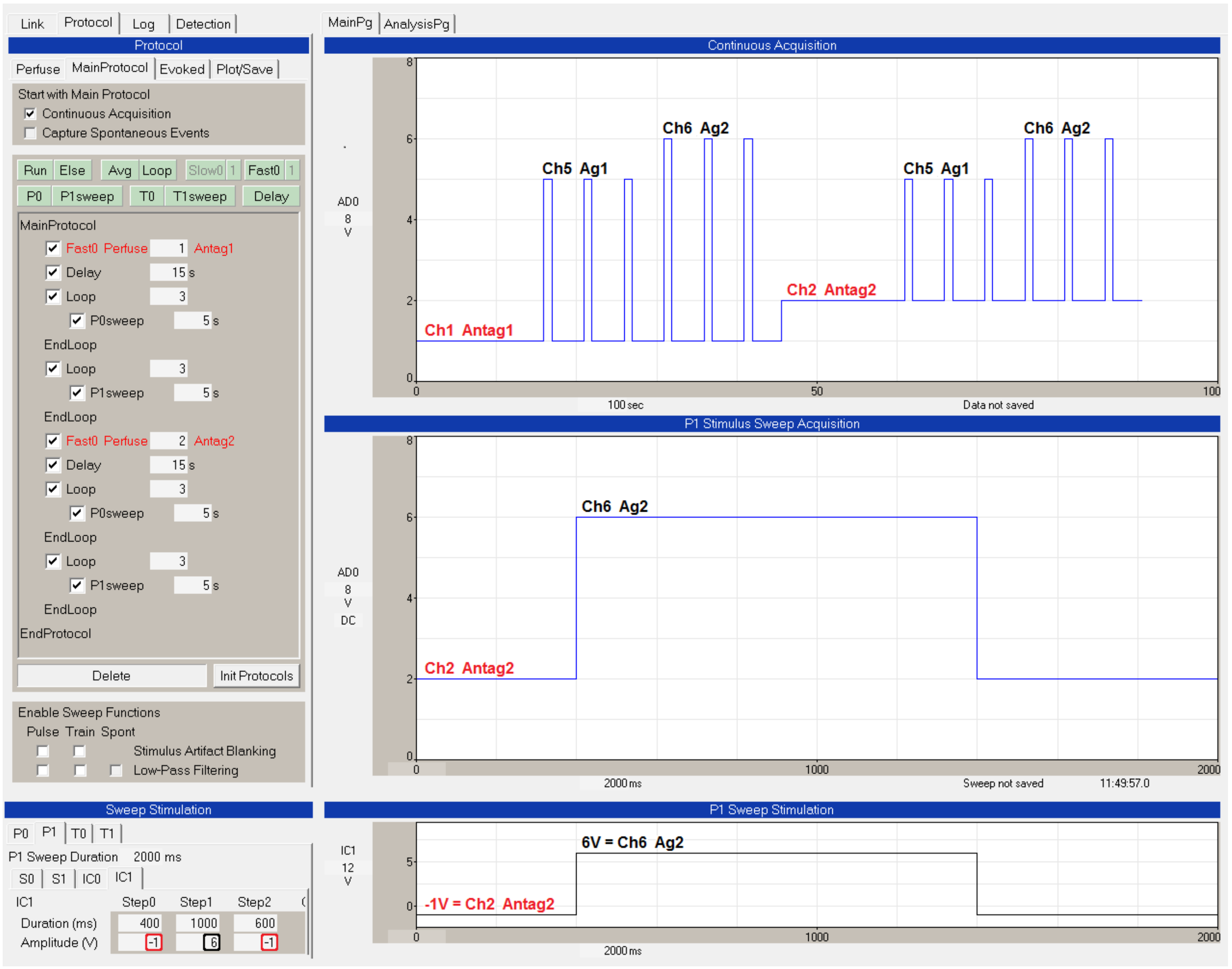
Acquisition Sample Intervals

Pulse Sweeps:	50 us
Train Sweeps:	50 us
Continuous Acquisition:	50 us
Spontaneous Sweeps:	us

Fig. 9.3.1.2.2. Setting the correct IC1 Data Type Units to V and the Gain to the appropriate value for your controller when using AnalogOut 1, or AnalogOut 1 and AnalogOut 2 outputs. For ALA Scientific controllers and for the example in of Fast Perfusion Change DURING and BETWEEN Sweeps using AO1 and AO2 shown in Fig. 9.3.1.2.4 a value of 0.5 is used, or 0.5 volts per channel.

An example of Fast Perfusion Change DURING and BETWEEN Sweeps is shown in Fig. 9.3.1.2.3 – this is for controlling 8 channels with AnalogOut 1. The Perfuse statement in the Protocol Builder (left panel) is used to change to 2 different antagonist concentrations between sweeps, and the Sweep Stimulation IC1 Amplitude (V) value is used to rapidly apply 2 different agonist concentration with the 2 antagonist concentrations during the sweep. After each Perfuse statement, a different sweep (P0, then P1) is output 3 times. Each sweep contains a 1000 msec output of antagonist + agonist solution. As shown for the P1sweep, setting the IC1 Step1 Amplitude (V) to 6 turns on Ch 6 perfusion (bottom panels). Setting Step0 and Step3 Amplitude (V) to -1 sets the perfusion channel to the channel set by the Perfuse statement.

If you want to control 16 channel Fast Perfusion Change DURING and BETWEEN Sweeps using two 8 channel Automate Scientific or ALA Scientific controllers, you need to use two analog outputs: AnalogOut 1 and AnalogOut 2 (Fig 9.3.1.2.4). To do this you need a National Instruments board with 4 analog outputs such as the PCI-6229, PCIe-6323 or USB-6229 BNC. For channels 1 to 8, AnalogOut 1 has a correct voltage output to turn on a particular channel in the first controller, and AnalogOut 2 voltage output is 0, ie the second controller is off. For channels 9 to 16, AnalogOut 1 has a voltage output of 0 meaning the first controller is off, and AnalogOut 2 has the correct voltage output to turn on a particular channel on the second controller.



Start with Main Protocol
☑ Continuous Acquisition
☐ Capture Spontaneous Events

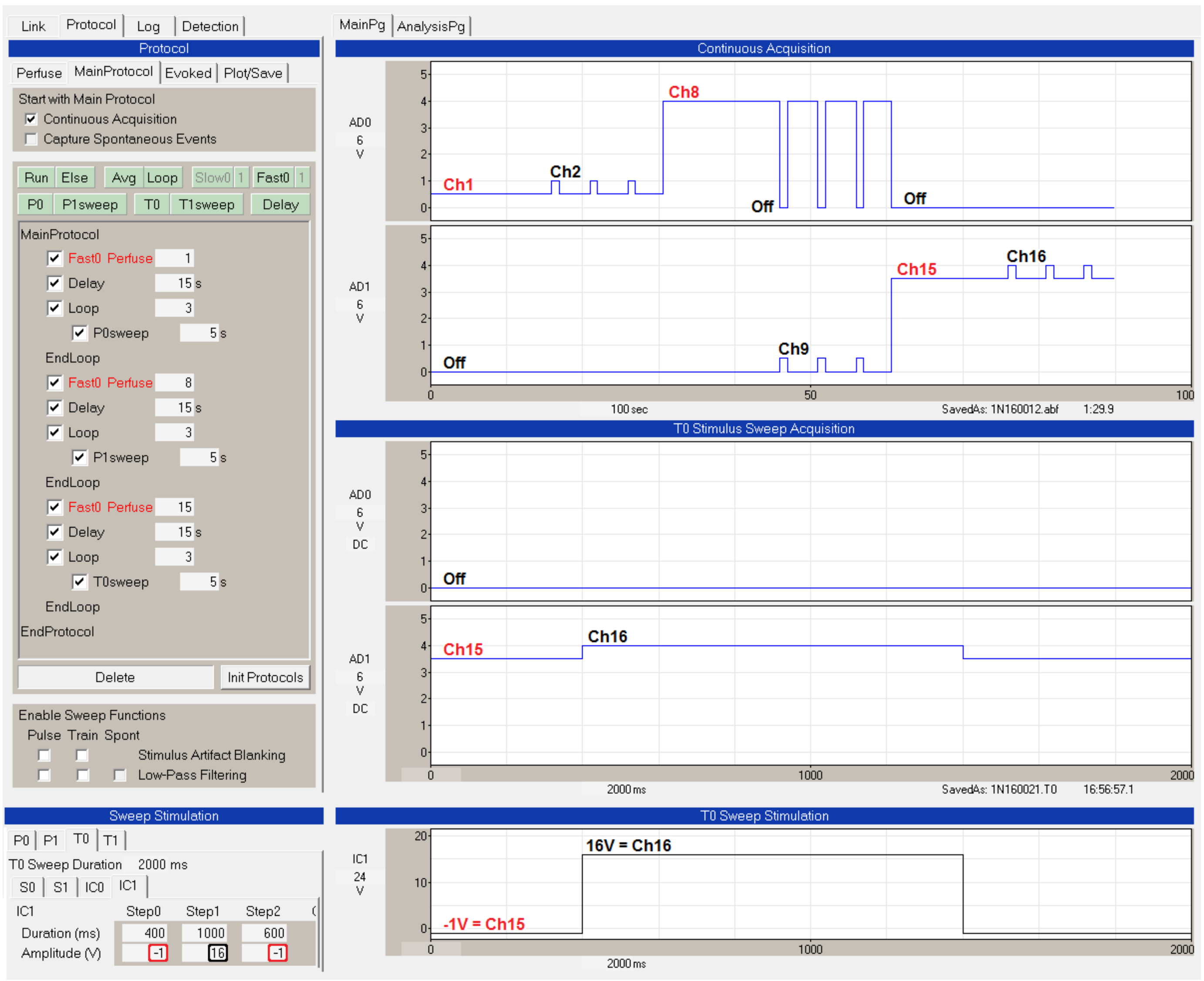
MainProtocol
☑ Fast0 Perfuse 1 Antag1
☑ Delay 15 s
☑ Loop 3
☑ P0sweep 5 s
EndLoop
☑ Loop 3
☑ P1sweep 5 s
EndLoop
☑ Fast0 Perfuse 2 Antag2
☑ Delay 15 s
☑ Loop 3
☑ P0sweep 5 s
EndLoop
☑ Loop 3
☑ P1sweep 5 s
EndLoop
EndProtocol

Enable Sweep Functions
Pulse Train Spont
☐ ☐ Stimulus Artifact Blanking
☐ ☐ ☐ Low-Pass Filtering

P1 Sweep Duration 2000 ms

IC1	Step0	Step1	Step2
Duration (ms)	400	1000	600
Amplitude (V)	-1	6	-1

Fig. 9.3.1.2.3. An example of Fast Perfusion Change DURING and BETWEEN Sweeps for controlling 8 channels with AnalogOut1. The Protocol Builder (big panel on left) shows the Perfuse statement changing antagonist concentration 2 times. After each Perfuse statement, a different sweep (P0, then P1) is output 3 times. Each sweep contains a 1000 msec output of antagonist + agonist solution (P0sweep, Ch 5, Antagonist 1 or 2, then Agonist 1, P1sweep, Ch 6, Antagonist 1 or 2 then Agonist 2 (right three panels). As shown for the P1sweep, setting the IC1 Step1 Amplitude (V) to 6 turns on Ch 6 perfusion (bottom panels). Setting Step0 and Step3 Amplitude (V) to -1 sets the perfusion channel to the channel set by the Perfuse statement. The AO1 Gain is 1.



Protocol

Start with Main Protocol
☑ Continuous Acquisition
☐ Capture Spontaneous Events

MainProtocol
☑ Fast0 Perfuse 1
☑ Delay 15 s
☑ Loop 3
☑ P0sweep 5 s
EndLoop
☑ Fast0 Perfuse 8
☑ Delay 15 s
☑ Loop 3
☑ P1sweep 5 s
EndLoop
☑ Fast0 Perfuse 15
☑ Delay 15 s
☑ Loop 3
☑ T0sweep 5 s
EndLoop
EndProtocol

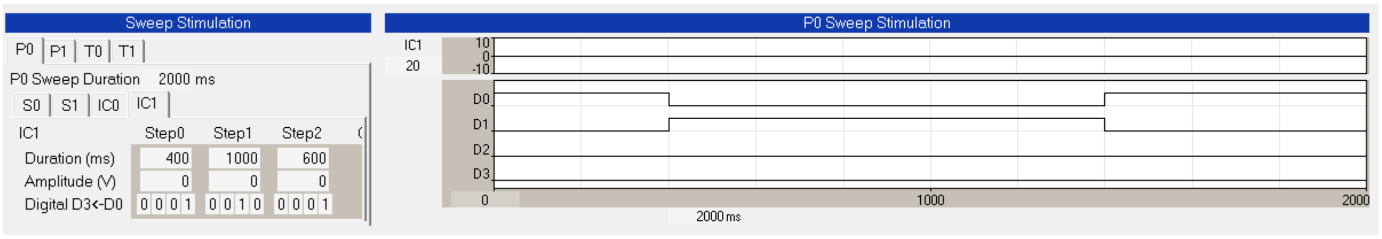
Enable Sweep Functions
Pulse Train Spont
☐ ☐ Stimulus Artifact Blanking
☐ ☐ ☐ Low-Pass Filtering

Sweep Stimulation
P0 P1 T0 T1
T0 Sweep Duration 2000 ms
S0 S1 IC0 IC1

IC1	Step0	Step1	Step2
Duration (ms)	400	1000	600
Amplitude (V)	-1	16	-1

Fig. 9.3.1.2.4. An example of Fast Perfusion Change DURING and BETWEEN Sweeps for controlling 16 channels using two controllers and AnalogOut 1 and AnalogOut 2. When AnalogOut 1 is on, AnalogOut 2 is off, and vice versa. Channel switches are from Ch 1 to 2, Ch 8 to 9, and Ch 15 to 16. Note the switchover from AO1 to AO2 for Ch 8 to 9. The Protocol Builder (big panel on left) shows the Perfuse statement changing to Channel 1, Channel 8 and Channel 15. After each Perfuse statement, a different sweep (P0, P1, then T0) is output 3 times. Each sweep contains a 1000 msec output of solution (P0sweep, Ch 2, P1sweep, Ch 9, and T0sweep, Ch 16, right bottom three panels). As shown for the T0sweep, setting the IC1 Step1 Amplitude (V) to 16 turns on Ch 16 (ie Ch 8 in the second controller) (bottom panels). AO1 and AO2 Gains are 0.5. See **Fast0_16chAO1AO2.pro**. To use connect AnalogOut1 (AO1) to AnalogIn0 (AI0 or AD0), and AnalogOut2 (AO2) to AnalogInput1 (AI1 or AD1).

In addition to Fast Perfusion Change using analog outputs to control perfusion changes DURING a sweep, Fast Perfusion can also use 4 bits of high speed digital output from Port 0 (Fig. 9.3.1.2.5). In the Sweep Stimulation section showing the IC0 or IC1 'Digital D3 <- D0' section (lower left panel), the channels are selected by putting in DigitalOutput bits 0 to 3 (ie Digital 3 <- 0). Note that either the digital outputs can be set from IC0 or IC1.



P0 Sweep Duration 2000 ms

IC1	Step0	Step1	Step2
Duration (ms)	400	1000	600
Amplitude (V)	0	0	0
Digital D3<-D0	0 0 0 1	0 0 1 0	0 0 0 1

Fig. 9.3.1.2.5. An example of Fast Perfusion change DURING a sweep for controlling 15 channels using one BioScience Tools controller and 4 bits from the high speed digital out Port 0 where if all the bits are off, all channels are off. After 400 msec, the digital output is switched from 0001 (= 1 = bit 1 on = Channel 1) to 0010 (= 2 = bit 2 on = Channel 2).for 1000 msec producing a 1000 msec Channel 2 pulse.

Table 9.3.1.2.2 shows show to set DigitalOut bits 3 to 0 in binary to turn on a perfusion channel for the BioScience Tools PC-16 controller. Only one perfusion channel can be on at any one time, and all bits off means that no channels are on, and all bits on turns on Ch 15. Channel 16 on the BioScience Tools controller has to be turned on manually.

	DigitalOut Bits			
	3	2	1	0
All Chs Off	0	0	0	0
Ch 1	0	0	0	1
Ch 2	0	0	1	0
Ch 3	0	0	1	1
Ch 4	0	1	0	0
Ch 5	0	1	0	1
Ch 6	0	1	1	0
Ch 7	0	1	1	1
Ch 8	1	0	0	0
Ch 9	1	0	1	0
Ch 10	1	0	0	1
Ch 11	1	0	1	1
Ch 12	1	1	0	0
Ch 13	1	1	0	1
Ch 14	1	1	1	0
Ch 15	1	1	1	1

Table 9.3.1.2.2. Setting the DigitalOut bits 3 to 0 in binary to turn on Ch1 to 15 on the BioScence Tools PC-16 controller. When DigitalOut bits 3 <- 0 equals '0000', then all channels are off.

To implement 4 bit DigitalOut control of the AutoMate Scientific ValveLink 8.2 in which DigitalOut bits 3 <- 0 = '0000' means Ch 1 on, '1111' means Ch 16 on (Table 9.3.1.2.3). Turning all channels in the Automate controller off has to be done manually.

	DigitalOut Bits			
	3	2	1	0
Ch 1	0	0	0	0
Ch 2	0	0	0	1
Ch 3	0	0	1	0
Ch 4	0	0	1	1
Ch 5	0	1	0	0
Ch 6	0	1	0	1
Ch 7	0	1	1	0
Ch 8	0	1	1	1
Ch 9	1	0	0	0
Ch 10	1	0	1	0
Ch 11	1	0	0	1
Ch 12	1	0	1	1
Ch 13	1	1	0	0
Ch 14	1	1	0	1
Ch 15	1	1	1	0
Ch 16	1	1	1	1

Table 9.3.1.2.3. Setting the DigitalOut bits 3 to 0 in binary to turn on Ch1 to 16 on the AutoMate Scientific ValveLink 8.2 controller. To turn all channels off has to be done manually.

The Warner VC-8 controller and the ALA Scientific VC^3 8 cannot be binary controlled by 4 binary bits.

Channels 1, 2 3 and 4 can also be controlled 'linearly' or independently with the high-speed digital output from Port0 by setting one bit for each channel (Table 9.3.1.2.3). No bits on means all channels are off. This works when using the ALA Scientific, Automate Scientific, BioScience Tools and Warner controllers.

	DigitalOut Bits			
	3	2	1	0
All Chs Off	0	0	0	0
Ch 1	0	0	0	1
Ch 2	0	0	1	0
Ch 3	0	1	0	0
Ch 4	1	0	0	0

Table 9.3.1.2.3. Setting the DigitalOut bits 3 to 0 to turn on 'linear' Channels Ch '1' to '4'.

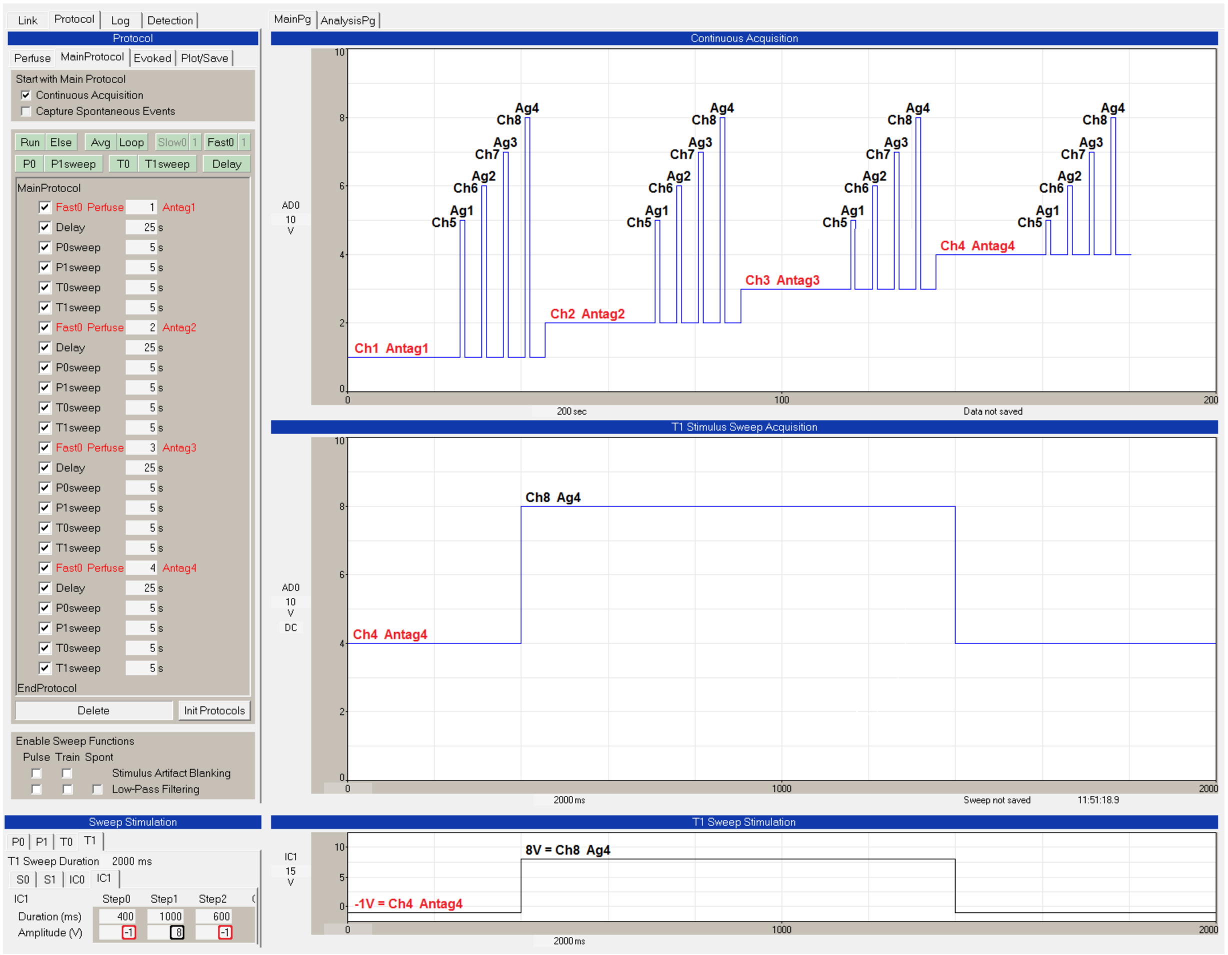
Note that if your antagonist does not rapidly unbind, and therefore agonist solutions need not contain antagonist, it is strongly preferable to use AO1 rather than Digital Output for fast perfusion changes during a sweep. This is because if you enter a -1V into the Amplitude field of the epoch, that means that the current fast perfusion channel will continue to be output during sweep epochs containing a -1V, and therefore each sweep can contain a unique agonist concentration, thereby limiting the number of different sweep stimulations required. This capability is not, to our knowledge, present in Molecular Devices pClamp or in HEKA's PatchMaster.

In addition, setting binary digital outputs is not nearly as straight forward as setting voltages as channel numbers for analog output.

For WinLTP 2.01, which has 4 different PxSweeps (P0, P1, T0 and T1), and when the antagonist is tightly bound, you can deliver 16 different antagonist concentrations and 4 agonist concentrations in one protocol file (when using -1V and analog output). When the antagonist is loosely bound and the agonist solutions must also contain an antagonist, you can have 1 antagonist concentration and 4 agonist concentrations with that one antagonist concentration in one protocol file. To test 4 antagonist and 4 agonist concentrations, you would need to link 4 protocol files (Section 9.3.2).

9.3.1.3 Fast0 Perfusion Example - Testing 4 Agonist and 4 Antagonist Concentrations

An example of applying 4 different agonists and 4 different antagonists (one of which could be no antagonist, ie ACSF) when the antagonist does not rapidly unbind (Table 9.3.1.1.1) is shown in Fig. 9.3.1.3.1. The protocol starts out with the Fast0 Perfuse event in the Protocol Builder causing Ch 1, Antagonist 1 to be perfused. After a delay, Ch 5, Agonst 1 is applied for 1 second during the P0sweep. Then shortly thereafter Ch 6, Agonist 2 is applied during a P1sweep, Ch 7, Agonist 3 is applied during a T0sweep, and Ch 8, Agonist 4 is applied during a T1sweep. Then the perfusion solution is changed to Ch 2, Antagonist 2, and application of the 4 agonist solutions is repeated. Then this is repeated for Ch 3, Antagonist 3 perfusion, and finally for Ch 4, Antagonist 4 perfusion.



Start with Main Protocol
- [✓] Continuous Acquisition
- [ ] Capture Spontaneous Events

MainProtocol
- [✓] Fast0 Perfuse 1 Antag1
- [✓] Delay 25 s
- [✓] P0sweep 5 s
- [✓] P1sweep 5 s
- [✓] T0sweep 5 s
- [✓] T1sweep 5 s
- [✓] Fast0 Perfuse 2 Antag2
- [✓] Delay 25 s
- [✓] P0sweep 5 s
- [✓] P1sweep 5 s
- [✓] T0sweep 5 s
- [✓] T1sweep 5 s
- [✓] Fast0 Perfuse 3 Antag3
- [✓] Delay 25 s
- [✓] P0sweep 5 s
- [✓] P1sweep 5 s
- [✓] T0sweep 5 s
- [✓] T1sweep 5 s
- [✓] Fast0 Perfuse 4 Antag4
- [✓] Delay 25 s
- [✓] P0sweep 5 s
- [✓] P1sweep 5 s
- [✓] T0sweep 5 s
- [✓] T1sweep 5 s

EndProtocol

Enable Sweep Functions

Pulse	Train	Spont	
[ ]	[ ]		Stimulus Artifact Blanking
[ ]	[ ]	[ ]	Low-Pass Filtering

T1 Sweep Duration 2000 ms

IC1	Step0	Step1	Step2
Duration (ms)	400	1000	600
Amplitude (V)	-1	8	-1

Fig. 9.3.1.3.1. Applying 4 different agonists and 4 different antagonists when the antagonist does not rapidly unbind. The Fast0 Perfuse events (shown in red in the Protocol Builder on the left) set the ongoing perfusion to Ch 1 Antagonist 1, Ch 2 Antgonist 2, Ch 3 Antagonist 3 and Ch 4 Antagonist 4. The agonist solution is applied during the sweep. During the first 400 msec epoch, Step0, and the third epoch, Step2, a -1 in the Amplitude (V) field (see red rectangles) means that the current perfusion solution set by the Fast0 Perfuse event in the Protocol Builder is maintained in that part of the sweep. During the 1000 msec epoch, Step1, a voltage from 1 to 8 in the Amplitude (V) field (black rectangle) sets the Ch from 1 to 8, in this case for T1sweep 8V=Ch8 or Agonist 4. AO1 Gain is 1. See **Fast0_8chAO1_4Antag_4Ag.pro**. To use connect AnalogOut1 (AO1) to AnalogIn0 (AI0 or AD0).

9.3.2 Using Protocol Linking to Overcome Limitation in Single Line Perfusion Protocols

One limitation of the current version of WinLTP is that there are only 4 different sweeps available to load in different rapid perfusion changes. However, patch-clamp automated perfusion changes can realistically involve at least 16 different perfusion solutions (3 antagonist and 12 agonist+antagonist solutions in the following example). With the future addition of 8 PulseSweeps and 8 TrainSweeps for a total of 16 different sweeps, 16 different perfusion solutions can be delivered in one protocol file. And with the future addition of increment/decrement of epoch voltages will also allow at least 16 different solutions to be delivered in one protocol file.

However, as an alternate solution to this problem now, we have implemented Protocol Linking in WinLTP 2.00 (Chapter 8). With Protocol Linking, once a protocol has finished (for example after applying 4 different agonist+antagonist solutions during the sweeps), WinLTP can now load and start the second protocol containing 4 more different solution changes, and then load and start the third protocol containing 4 further different solution changes, and can do this, ad infinitum. Also, even if we had implemented our 16 sweeps for applying 16 different solutions during sweeps, it is very conceivable that a researcher would want to deliver 32 different solutions and therefore would have to use Protocol Linking anyways.

Figs. 8.1.1 and 9.3.2.1 show an example of using Protocol Linking with Fast Perfusion link three different protocol files to continuously perfuse solutions containing 3 different antagonist concentrations, and momentarily rapidly apply 4 different agonist concentrations with the different antagonist concentrations - ie apply 15 different solutions.

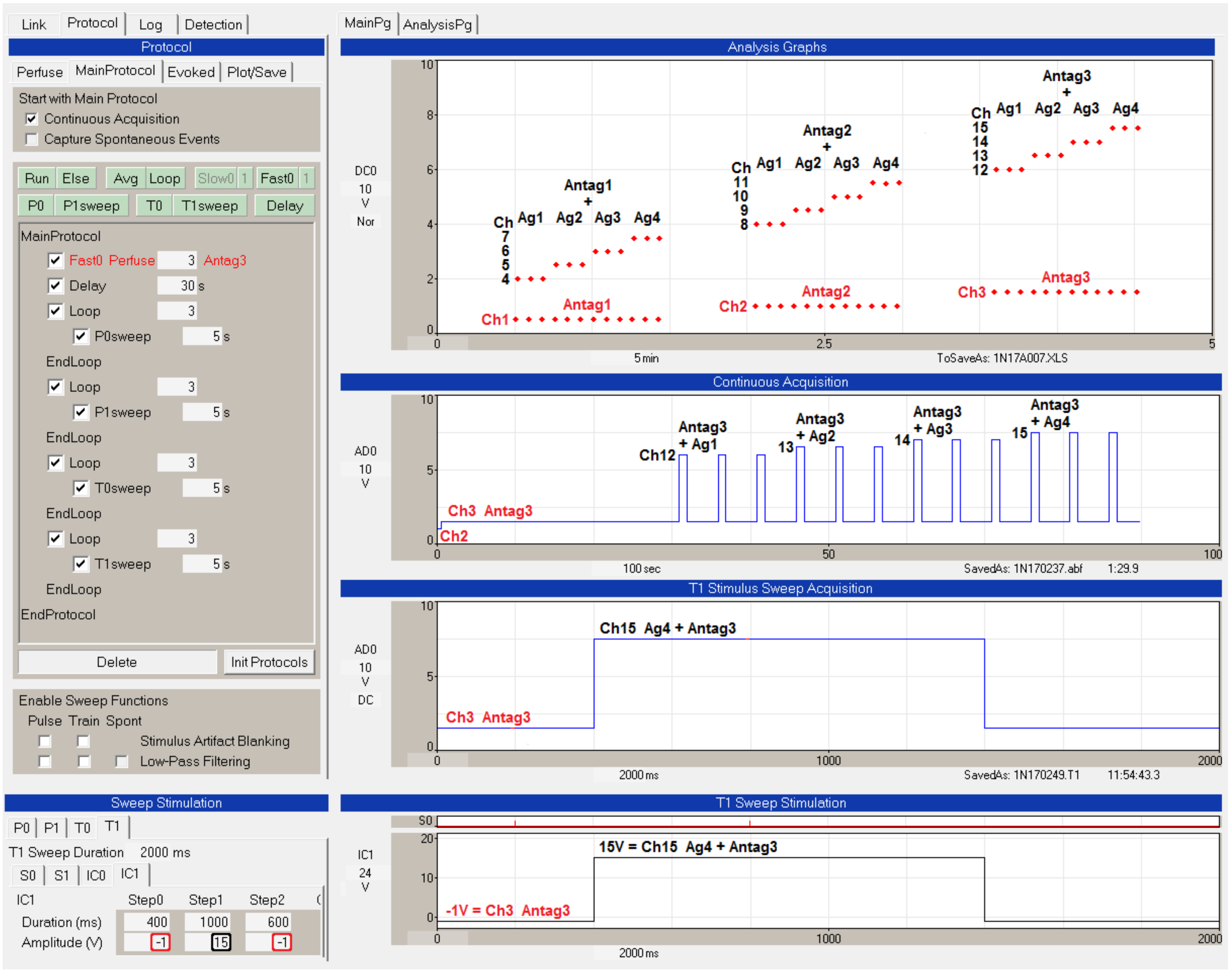
Figs. 8.1.1 shows the Protocol Linking section of the first protocol file which is set to Load the second protocol file "FastPerfusion2.pro" when the first protocol self-terminates. The AutoStart check box is also checked so the second protocol will automatically start running.

Fig. 9.3.2.1 shows the results after the third linked protocol file has finished. The first two protocol files have already finished. The first protocol file (FastProtocol1.pro) delivered Antagonist 1, then Antagonist 1 + Agonist 1, then Antagonist 1 + Agonist 2, then Antagonist 1 + Agonist 3, then Antagonist 1 + Agonist 4. The second protocol file (FastProtocol2.pro) delivered Antagonist 2, then Antagonist 2 + Agonists 1 to 4, and the third protocol file (FastProtocol3.pro) delivered Antagonist 3, then Antagonist 3 + Agonists 1 to 4.

The Protocol Builder (big panel on left) shows that when the third protocol file auto-starts, the Perfuse statement causes Ch3, Antagonist 3 solution to be continuously perfused. Then four different sweeps (P0, P1, T0 then T1) are output 4 times. Each sweep contains a 1000 msec output of Antagonist 3 + Agonist 1 (P0sweep, Ch12), Antagonist 3 + Agonist 2 (P1sweep, Ch13), Antagonist 3 + Agonist 3 (T0sweep, Ch14) and Antagonist 3 + Agonist 4 (T1sweep, Ch15, shown in the bottom two panels). For T1sweep, setting the Step1 Amplitude (V) to 15V turns on Ch 15 perfusion (bottom left panel, black rectangle).

In this way, running the first protocol file (which output with Antagonist 1 alone and with 4 different antagonist solutions) then loaded and started second protocol file (which output with Antagonist 2 alone and with 4 different antagonist solutions), which then loaded and started the third protocol file (outputting Antagonist 3 and with 4 different antagonist solutions). The top right Analysis Grpahs panel shows DC output voltages for turning on the Chs 1-3 antagonist solutions and Chs 4-15 antagonist + agonist solutions for the first, linked second, and linked third protocol files. The Continuous Acquisition panel shows the output voltage for the third protocol file only.

The log output of this protocol linking experiment is shown in Fig. 9.3.2.2. The starting of FastProtocol1.pro, FastProtocol2.pro and FastProtocol1.pro, and the switching to Ch2 Antag2 and Ch3 Antag3 are shown by the comments on the right.



Start with Main Protocol
☑ Continuous Acquisition
☐ Capture Spontaneous Events

MainProtocol
☑ Fast0 Perfuse 3 Antag3
☑ Delay 30 s
☑ Loop 3
☑ P0sweep 5 s
EndLoop
☑ Loop 3
☑ P1sweep 5 s
EndLoop
☑ Loop 3
☑ T0sweep 5 s
EndLoop
☑ Loop 3
☑ T1sweep 5 s
EndLoop
EndProtocol

Enable Sweep Functions
Pulse Train Spont
☐ ☐ Stimulus Artifact Blanking
☐ ☐ ☐ Low-Pass Filtering

T1 Sweep Duration 2000 ms

IC1	Step0	Step1	Step2
Duration (ms)	400	1000	600
Amplitude (V)	-1	15	-1

Fig. 9.3.2.1. An example of using Protocol Linking with Fast Perfusion to link 3 different protocol files to continuously perfuse solutions containing 3 different antagonist concentrations, and momentarily rapidly apply 4 different agonist concentrations with the different antagonist concentrations - ie apply 15 different solutions. Uses 16ch AO1 with a Gain of 0.5. See **FastPerfusion1.pro**, **FastPerfusion2.pro**, and **FastPerfusion3.pro**. To use connect AnalogOut1 (AO1) to AnalogIn0 (AI0 or AD0).

```
TimeOfDay AnalysisTm Event
17:33:42.3        "" Start WinLTP
""                ""   Using FastPerfusion1.pro
""                ""   Saving to 1N210022.log
""
17:35:48.9       0.0 Start MainProtocol                  'Start FastPerfusion1.pro
17:35:48.9       0.0 Fast0 Perfuse  Ch 1  Antag1  is on  'Start MainProtocol with Ch1 Antag1
17:35:49.4       0.5 Fast0 Perfuse  Ch 1  Antag1
17:35:49.4       0.5   Delay
17:36:19.4      30.5   Enter Loop0
17:36:19.4      30.5     1N210401.P0, First Loop sweep
17:36:34.4      45.5   Leave Loop0
17:36:34.4      45.5   Enter Loop0
17:36:34.4      45.5     1N210404.P1, First Loop sweep
17:36:49.4    1:00.5   Leave Loop0
17:36:49.4    1:00.5   Enter Loop0
17:36:49.4    1:00.5     1N210407.T0, First Loop sweep
17:37:04.4    1:15.5   Leave Loop0
17:37:04.4    1:15.5   Enter Loop0
17:37:04.4    1:15.5     1N210410.T1, First Loop sweep
17:37:19.4    1:30.5   Leave Loop0
17:37:19.4    1:30.5 Stop MainProtocol
17:37:19.6        "" Loaded FastPerfusion2.pro
""
17:37:20.5    1:31.6 Start MainProtocol                  'Start FastPerfusion2.pro
17:37:20.5    1:31.6 Fast0 Perfuse  Ch 1  Antag1  is on
17:37:21.0    1:32.1 Fast0 Perfuse  Ch 2  Antag2         'Switch to Ch2 Antag2
17:37:21.0    1:32.1   Delay
17:37:51.0    2:02.1   Enter Loop0
17:37:51.0    2:02.1     1N210414.P0, First Loop sweep
17:38:06.0    2:17.1   Leave Loop0
17:38:06.0    2:17.1   Enter Loop0
17:38:06.0    2:17.1     1N210417.P1, First Loop sweep
17:38:21.0    2:32.1   Leave Loop0
17:38:21.0    2:32.1   Enter Loop0
17:38:21.0    2:32.1     1N210420.T0, First Loop sweep
17:38:36.0    2:47.1   Leave Loop0
17:38:36.0    2:47.1   Enter Loop0
17:38:36.0    2:47.1     1N210423.T1, First Loop sweep
17:38:51.0    3:02.1   Leave Loop0
17:38:51.0    3:02.1 Stop MainProtocol
17:38:51.1        "" Loaded FastPerfusion3.pro
""
17:38:52.0    3:03.1 Start MainProtocol                  'Start FastPerfusion3.pro
17:38:52.0    3:03.1 Fast0 Perfuse  Ch 2  Antag2  is on
17:38:52.5    3:03.6 Fast0 Perfuse  Ch 3  Antag3         'Switch to Ch3 Antag3
17:38:52.5    3:03.6   Delay
17:39:22.5    3:33.6   Enter Loop0
17:39:22.5    3:33.6     1N210427.P0, First Loop sweep
17:39:37.5    3:48.6   Leave Loop0
17:39:37.5    3:48.6   Enter Loop0
17:39:37.5    3:48.6     1N210430.P1, First Loop sweep
17:39:52.5    4:03.6   Leave Loop0
17:39:52.5    4:03.6   Enter Loop0
17:39:52.5    4:03.6     1N210433.T0, First Loop sweep
17:40:07.5    4:18.6   Leave Loop0
17:40:07.5    4:18.6   Enter Loop0
17:40:07.5    4:18.6     1N210436.T1, First Loop sweep
17:40:22.5    4:33.6   Leave Loop0
17:40:22.5    4:33.6 Stop MainProtocol
```

Fig. 9.3.2.2. The experimental log for the protocol linking experiment of Fig. 9.3.2.1. Comments are shown on the right.

9.4 Automated Dual- and Triple-Line/Stepper Perfusion Patch-Clamp Experiments

In Section 9.2 we showed an automated multi-slice extracellular experiments using automated perfusion control (with and without pre-flushing line, changing between sweeps) which typically is used to produce solution changes in the **minutes** timeframe. In Section 9.3 we showed an automated single-line perfusion control for patch-Clamp experiments which typically is used to produce solution changes in the **tens of milliseconds to seconds** timeframe. In this section we show how to implement perfusion changes as quick as a **millisecond** in duration by using a stepper in addition to two or three valve controllers.

The Dual- and Triple-Line/Stepper Perfusion system need to use only the inexpensive pinch valves used with Extracellular Slice Perfusion, and not the faster, more expensive valves such as Lee valves that are required for Single-Line Perfusion. Although many piezo steppers can be quite expensive, some steppers such as the Warner SF-77 are reasonably priced. If an inexpensive stepper and controllers using inexpensive pinch valves are used, the price of such a system will be not much more than a Single-Line Perfusion system. We do not necessarily recommend such a system, but it should be considered.

9.4.1 Automated Dual-Line/Stepper Perfusion

The way that WinLTP can control a Dual-Line/Stepper perfusion system is shown in Fig. 9.4.1.1. Slow0 Perfusion produces digital output on static Port1 to control one valve controller which in turn controls 4 to 16 valves. This Slow0 perfusion is for producing slow perfusion changes BETWEEN sweeps, and controls the perfusion solution coming out of Tube 0, which usually contains several **agonist** or agonist + antagonist test solutions.

Slow1 Perfusion similarly produces digital output on static Port2 to control one valve controller which in turn controls 4 to 16 valves. This Slow1 perfusion is for producing slow perfusion changes BETWEEN sweeps, and controls the perfusion solution coming out of Tube 1 which usually contains several **antagonist** solutions possibly including normal ACSF (a no antagonist solution).

The Fast0 Stepper produces fast changing output from 4-bits of the high-speed digital Port0, or output from AnalogOut 1 or AnalogOut 1 & 2 DURING the sweep. Fast0 output controls the Stepper which quickly moves the two tubes from Tube 1 to Tube 0 and back by controlling a Stepper. If the stepper is a fast piezo device, solution deliveries from Tube 0 can be as quick as one millisecond in duration.

As with Single-Line Perfusion (Section 9.3.1.1), when designing your experiment you have to determine whether the antagonist is strongly or weakly bound to the receptor. If the antagonist is strongly bound and is **not displaced** during agonist application, then one set of reservoirs can contain different concentrations of antagonist and another **minimal set** of reservoirs can contain different concentrations of agonist. If the antagonist is weakly bound and **is displaced** during agonist application, then one set of reservoirs can contain different concentrations of antagonist and another **much larger set** of reservoirs can contain different concentrations of agonist + antagonist. This increases the required number of solutions and valves substantially.

However, for Dual-Line Stepper Perfusion, WinLTP can easily control 16 valves and solutions by Slow0 and Slow1 each, so WinLTP can easily control 16 concentrations.of antagonist (Slow1) and 16 concentrations of agonist + antagonist (Slow0). The problem is more cost of the valve contollers and the amount of solution preparation required rather than problems with WinLTP per se.

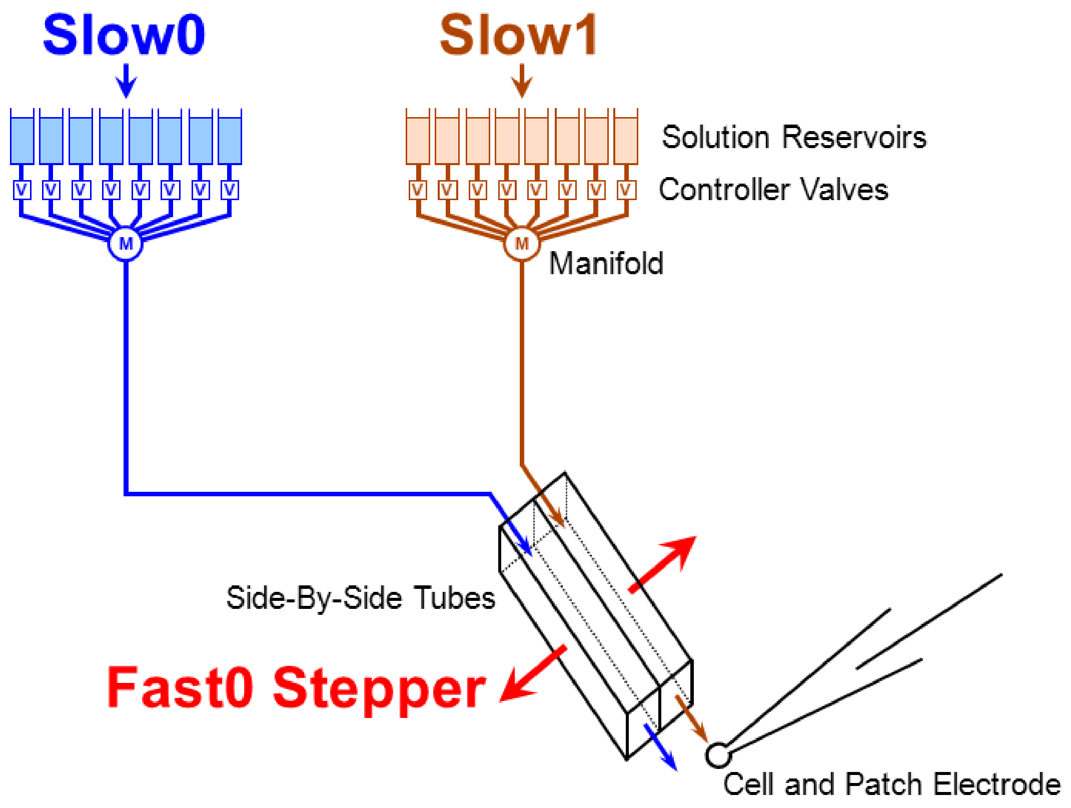


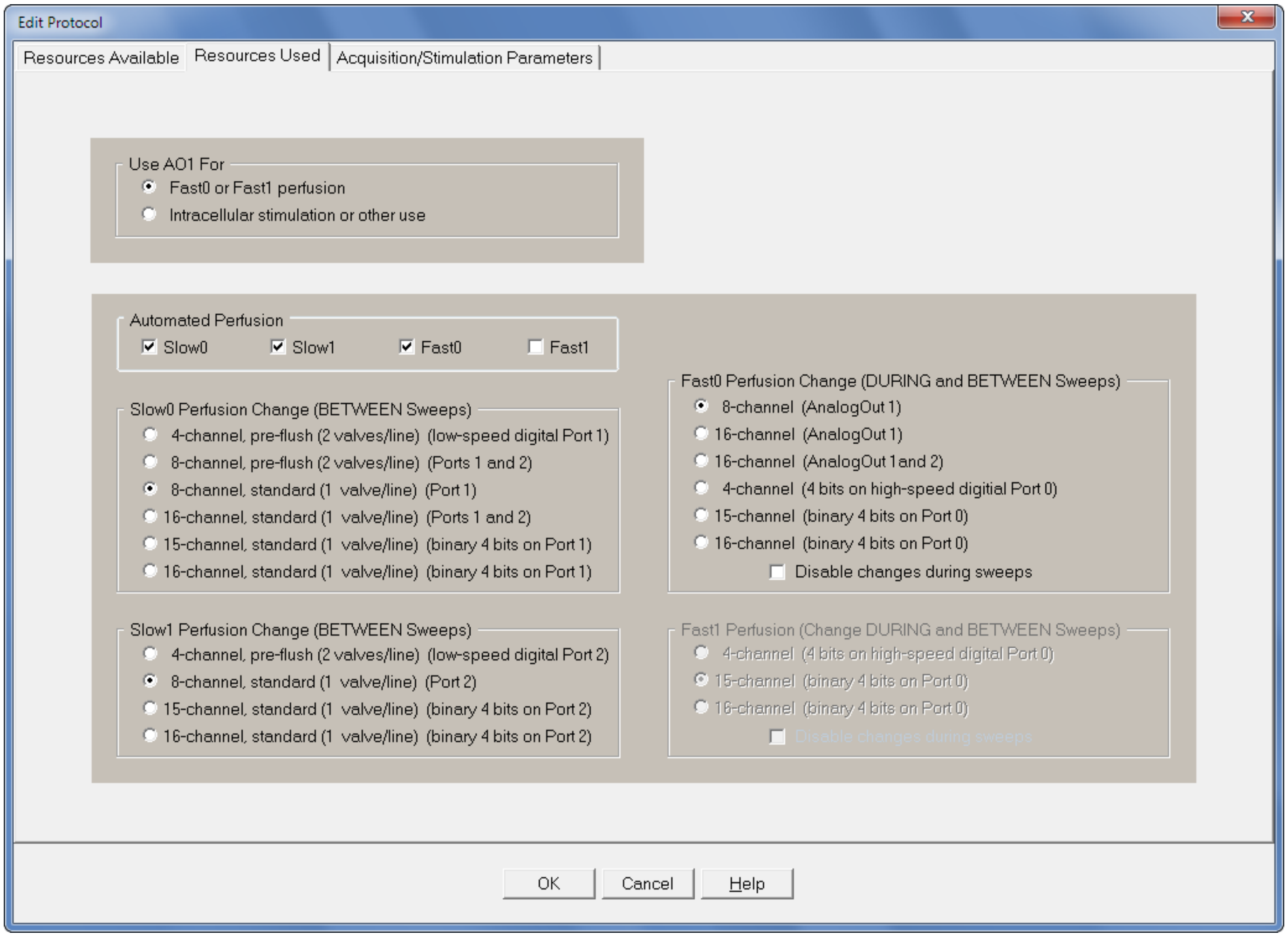
Fig. 9.4.1.1. WinLTP control of Dual-Line perfusion and a Stepper. Slow0 and Slow1 Perfusion can control two valve controllers, and the Fast0 Perfusion controls the Stepper which controls whether solution from Tube 0 (blue) or Tube 1 (brown) is applied to the cell. The Fast0 Stepper therefore controls whether antagonist (Slow1) or agonist (Slow0) is applied.

Figs. 9.4.1.2, 9.4.1.3 and 9.4.1.4 show a simplified experiment involving only two tubes (Tube 0 controlled by Slow0 perfusion, and Tube 1 controlled by Slow1 perfusion), and a Fast0 controlled stepper to switch between Tube 1 and Tube 0.

Fig. 9.4.2.2 shows how the Resources Used tabsheet is typically used for Dual-Line/Stepper perfusion. First, in the ‘Use AD1 For’ radiobutton group, AO1 is usually used for Fast0 or Fast1 perfusion.

Then for the Automated Perfusion checkboxes, Slow0, Slow1 and Fast0 checkboxes are checked. In this example 8-channel standard (ie 1 valve/line) digital output from Port 1 controls the Slow0 controller. Similarly, 8-channel standard (ie 1 valve/line) digital output from Port 2 controls the Slow1 controller. By using ‘linear’ digital output (ie one bit controlling 1 valve) rather than binary control (3 bit s controlling 8 valves or 4 bits controlling 16 valves), **a single physicial controller can, in essence, be two logical controllers**. In this case, Slow0 Port1 digital output can control 4 valves of an 8 valve controller, and Slow1 Port2 digital output can control the other 4 valves of the 8 valve controller.

Then for the Fast0 Perfusion to change solutions DURING a sweep, the 8 channel AO1 output is selected, although fast digital output via the high-speed Port 0 could also have been selected if that is required by the stepper.



Edit Protocol

Resources Available | Resources Used | Acquisition/Stimulation Parameters

Use AO1 For
- [✓ Fast0 or Fast1 perfusion]
- Intracellular stimulation or other use

Automated Perfusion
- [✓ Slow0] [✓ Slow1] [✓ Fast0] Fast1

Slow0 Perfusion Change (BETWEEN Sweeps)
- 4-channel, pre-flush (2 valves/line) (low-speed digital Port 1)
- 8-channel, pre-flush (2 valves/line) (Ports 1 and 2)
- [✓ 8-channel, standard (1 valve/line) (Port 1)]
- 16-channel, standard (1 valve/line) (Ports 1 and 2)
- 15-channel, standard (1 valve/line) (binary 4 bits on Port 1)
- 16-channel, standard (1 valve/line) (binary 4 bits on Port 1)

Slow1 Perfusion Change (BETWEEN Sweeps)
- 4-channel, pre-flush (2 valves/line) (low-speed digital Port 2)
- [✓ 8-channel, standard (1 valve/line) (Port 2)]
- 15-channel, standard (1 valve/line) (binary 4 bits on Port 2)
- 16-channel, standard (1 valve/line) (binary 4 bits on Port 2)

Fast0 Perfusion Change (DURING and BETWEEN Sweeps)
- [✓ 8-channel (AnalogOut 1)]
- 16-channel (AnalogOut 1)
- 16-channel (AnalogOut 1 and 2)
- 4-channel (4 bits on high-speed digitial Port 0)
- 15-channel (binary 4 bits on Port 0)
- 16-channel (binary 4 bits on Port 0)
- Disable changes during sweeps

Fast1 Perfusion (Change DURING and BETWEEN Sweeps)
- 4-channel (4 bits on high-speed digital Port 0)
- [✓ 15-channel (binary 4 bits on Port 0)]
- 16-channel (binary 4 bits on Port 0)
- Disable changes during sweeps

OK | Cancel | Help

Fig. 9.4.1.2. The ResourcesUsed tabsheet configured to run Dual-Line/Stepper perfusion. First choose how AO1 will be used. Next check the Slow0, Slow1 and Fast0 perfusion checkboxes. Finally choose the digital and analog output to control the Slow0 and Slow1 valve controllers and the Fast0 controlled stepper.

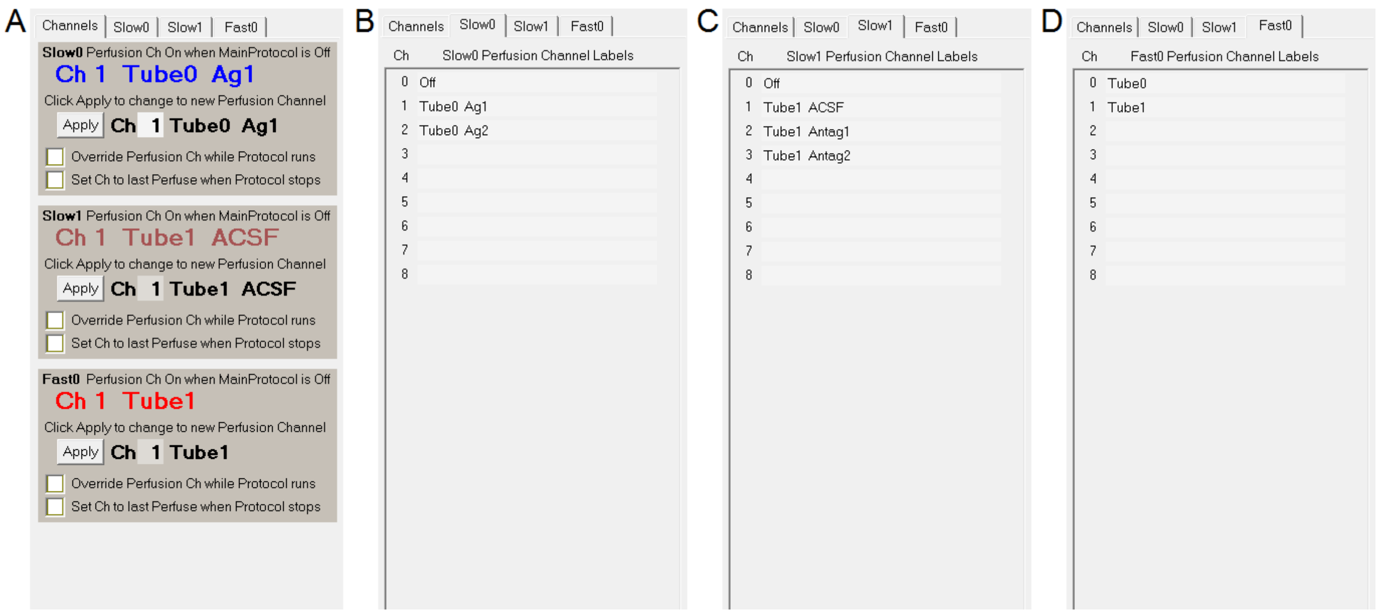
Next you have to set up the perfusion channel label information (Fig. 9.4.1.3, B-D). In the Slow0 (B), Slow1 (C) and Fast0 (D) channel labels, put in the appropriate perfusion channel labels. In this example, for Slow0 (the Tube 0 valve controller), the Slow0 Perfuse event in the Protocol Builder can either output agonist 1 or agonist 2 ('Tube0 Ag1' or 'Tube0 Ag2'). For Slow1 (the Tube 1 valve controller), the 'Slow1 Perfuse' event in the Protocol Builder can either output ACSF, antagonist 1 or antagonist 2 ('Tube1 ACSF', 'Tube1 Antag1' or 'Tube1 Antag2'). For Fast0, the stepper can switch between 'Tube0' or 'Tube1'.

Then in the Channels tabsheet (Fig. 9.4.1.3A), set which Slow0 and Slow1 perfusion channel will be on when the MainProtocol is off. Enter the Ch number by the 'Apply' button and then click the 'Apply' button. The perfusion channel number when the MainProtocol is off will be shown above the 'Apply' button plus the perfusion channel label. For Slow0 perfusion it is Ch 1 with a 'Tube0 Ag1' label. For Fast0, set which tube will be bathing the cell when the MainProtocol is off, in this case, Ch1 with a 'Tube1' label. Neither the ‘Override Perfusion Ch while the Protocol runs’ checkbox or the ‘Set Ch to last Perfuse when Protocol stops’ checkbox needs to be checked.

Fig. 9.4.1.4 shows running the Dual-Line/Stepper example. The ‘Slow0 Perfuse’ and ‘Slow1 Perfuse’ events in the Protocol Builder (upper left) show when the Slow0/Tube0 and Slow1/Tube1 solutions are changed. The MainProtocol starts out with the ‘Slow1 Perfuse’ event making the Slow1/Tube1 controller switch to Ch2, ‘Antag1’ (which it already was), and with the ‘Slow0 Perfuse’ event making Slow0/Tube0 controller switch to Ch1, ‘Ag1’, and then switches to Ch2 ‘Ag2’. Then the ‘Slow1 Perfuse’ event switches to Ch3, ‘Antag2’, and with the ‘Slow0 Perfuse’ event switches to Ch1, ‘Ag1’, and then Ch2 ‘Ag2’.

When the AO1 voltage rapidly shifts beween Ch1 Tube1 and Ch0 Tube0 during the sweep (shown in ‘Continuous Acquisition’ panel and the ‘P0 Stimulus Sweep Acqusition’ panel are recordings of the AO1 voltage shifting between Ch1 Tube1 and Ch0 Tube0 that bath the cell. Because of the way the Slow0 and Slow1 Perfuse events determine which solution is flow through Tube 0 and Tube 1, the solutions switch between Antag1, Ag1 Antag1, Ag2, then Antag2 Ag1 Antag2 Ag2.

The Sweep Stimulation panels (bottom panels) show changes of AO1 voltage output during a sweep. The -1V means to take whatever the current Fast0 output is BETWEEN sweeps (and when the MainProtocol is off). During the P0sweep the AO1 voltage switches from 1V (because -1V means take the current Fast0 channel which is Ch 1), to 0V, Ch0 and back to 1V, Ch1.



A

Slow0 Perfusion Ch On when MainProtocol is Off
Ch 1 Tube0 Ag1
Click Apply to change to new Perfusion Channel
Ch 1 Tube0 Ag1
☐ Override Perfusion Ch while Protocol runs
☐ Set Ch to last Perfuse when Protocol stops

Slow1 Perfusion Ch On when MainProtocol is Off
Ch 1 Tube1 ACSF
Click Apply to change to new Perfusion Channel
Ch 1 Tube1 ACSF
☐ Override Perfusion Ch while Protocol runs
☐ Set Ch to last Perfuse when Protocol stops

Fast0 Perfusion Ch On when MainProtocol is Off
Ch 1 Tube1
Click Apply to change to new Perfusion Channel
Ch 1 Tube1
☐ Override Perfusion Ch while Protocol runs
☐ Set Ch to last Perfuse when Protocol stops

B

Ch	Slow0 Perfusion Channel Labels
0	Off
1	Tube0 Ag1
2	Tube0 Ag2
3	
4	
5	
6	
7	
8	

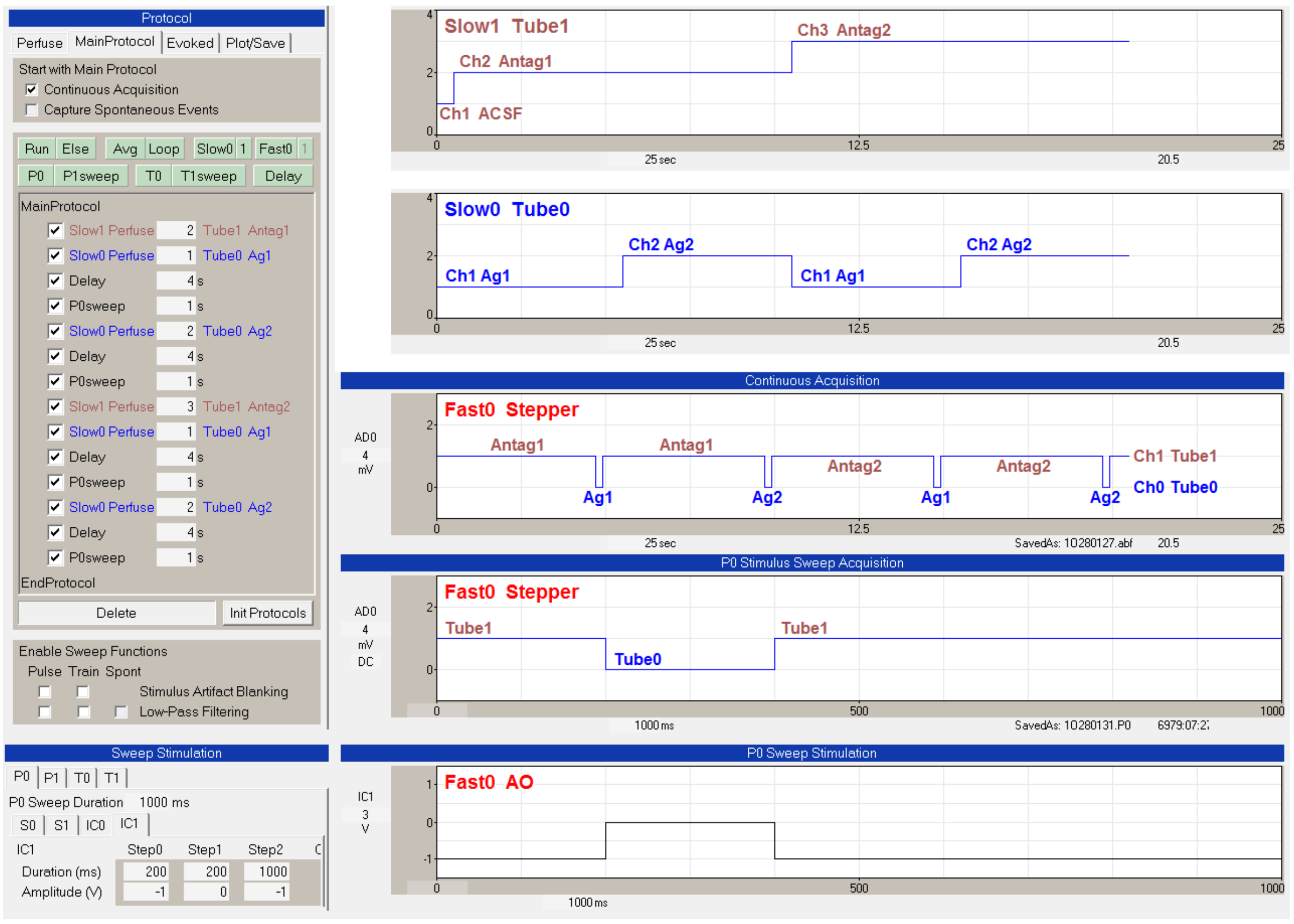
C

Ch	Slow1 Perfusion Channel Labels
0	Off
1	Tube1 ACSF
2	Tube1 Antag1
3	Tube1 Antag2
4	
5	
6	
7	
8	

D

Ch	Fast0 Perfusion Channel Labels
0	Tube0
1	Tube1
2	
3	
4	
5	
6	
7	
8	

Fig. 9.4.1.3. Setting up the perfusion channel number and label information for the Dual-Line/Stepper example.



Protocol

Perfuse | MainProtocol | Evoked | Plot/Save

Start with Main Protocol
☑ Continuous Acquisition
☐ Capture Spontaneous Events

Run | Else | Avg | Loop | Slow0 1 | Fast0 1
P0 | P1sweep | T0 | T1sweep | Delay

MainProtocol

☑ Slow1 Perfuse	2	Tube1 Antag1	
☑ Slow0 Perfuse	1	Tube0 Ag1	
☑ Delay	4	s	
☑ P0sweep	1	s	
☑ Slow0 Perfuse	2	Tube0 Ag2	
☑ Delay	4	s	
☑ P0sweep	1	s	
☑ Slow1 Perfuse	3	Tube1 Antag2	
☑ Slow0 Perfuse	1	Tube0 Ag1	
☑ Delay	4	s	
☑ P0sweep	1	s	
☑ Slow0 Perfuse	2	Tube0 Ag2	
☑ Delay	4	s	
☑ P0sweep	1	s	

EndProtocol

Delete | Init Protocols

Enable Sweep Functions
Pulse Train Spont
☐ ☐ Stimulus Artifact Blanking
☐ ☐ ☐ Low-Pass Filtering

Sweep Stimulation

P0 | P1 | T0 | T1

P0 Sweep Duration 1000 ms

S0 | S1 | IC0 | IC1

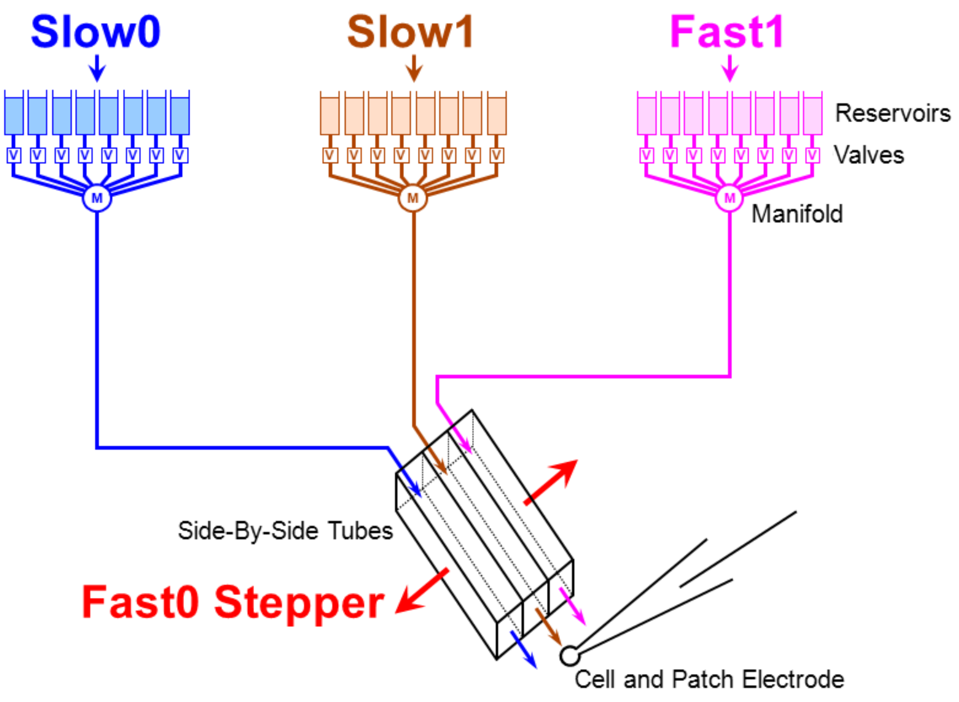
IC1	Step0	Step1	Step2
Duration (ms)	200	200	1000
Amplitude (V)	-1	0	-1

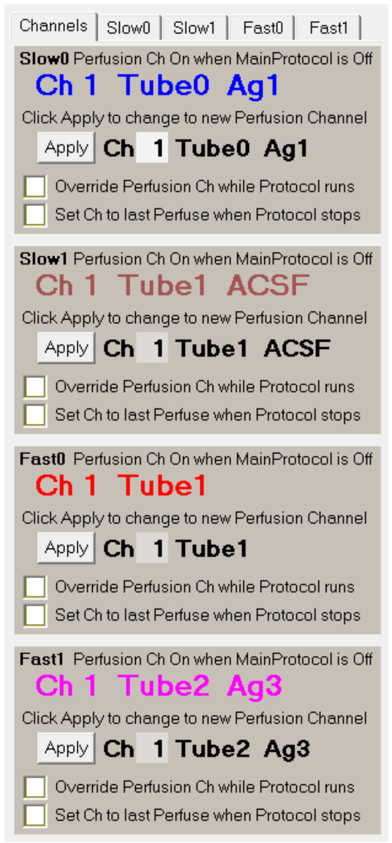
Fig. 9.4.1.4. Running the Dual-Line/Stepper example. The ‘Slow0 Perfuse’ and ‘Slow1 Perfuse’ events in the Protocol Builder (upper left) show when the Slow0/Tube0 and Slow1/Tube1 solutions are changed. The upper right panel shows the Slow1 Tube1 switch from ‘Ch1 ACSF’ to ‘Ch2 Antag1’ to ‘Ch3 Antag2’. The next panel below shows the Slow0 Tube0 switch between ‘Ch1 Ag1’ and ‘Ch2 Ag2’. The ‘Continuous Acquisition’ and the ‘P0 Stimulus Sweep Acquisition’ panel show recordings of the AO1 voltage shifts changing to Ch1 Tube1 or Ch0 Tube0. This results in the solutions switching between Antag1, Ag1 Antag1, Ag2, then Antag2 Ag1 Antag2 Ag2. The Sweep Stimulation bottom panels show change of AO1 (ie IC1) voltage output during a sweep. The -1V means to take whatever the current Fast0 output is BETWEEN sweeps (and when the MainProtocol is off), ie 1V or Ch1. **See DualStepper.pro.**

9.4.2 Automated Triple-Line/Stepper Perfusion

The Triple-Line/Stepper perfusion system is like the Dual-Line/Stepper system described above except with the addition of a third tube, Tube2, controlled by Fast1 that will also usually perfuse an agonist or agonist + antagonist solution. By adding the Fast1/Tube2 perfusion you can perfuse an addition 16 perfusion solutions (or 32 solutions if you are using a BioScience Tools 16 valve controller and using AO1 + AO2). Therefore, rather than the Fast0 just stepping from Ch1 Tube1 (antagonist) to Ch0 Tube0 (agonist or agonist + antonist), you can also step from Ch1 Tube1 to the neighboring Ch2 Tube2 (additional agonists or agonists + antago9nists).

The way that WinLTP can control a Triple-Line/Stepper perfusion system is shown in Fig. 9.4.2.1. Note that Fast1 (fuchsia) controls valves for solutions leading to Tube2.





Slow0 Perfusion Ch On when MainProtocol is Off
Ch 1 Tube0 Ag1
Click Apply to change to new Perfusion Channel
Apply Ch 1 Tube0 Ag1
☐ Override Perfusion Ch while Protocol runs
☐ Set Ch to last Perfuse when Protocol stops

Slow1 Perfusion Ch On when MainProtocol is Off
Ch 1 Tube1 ACSF
Click Apply to change to new Perfusion Channel
Apply Ch 1 Tube1 ACSF
☐ Override Perfusion Ch while Protocol runs
☐ Set Ch to last Perfuse when Protocol stops

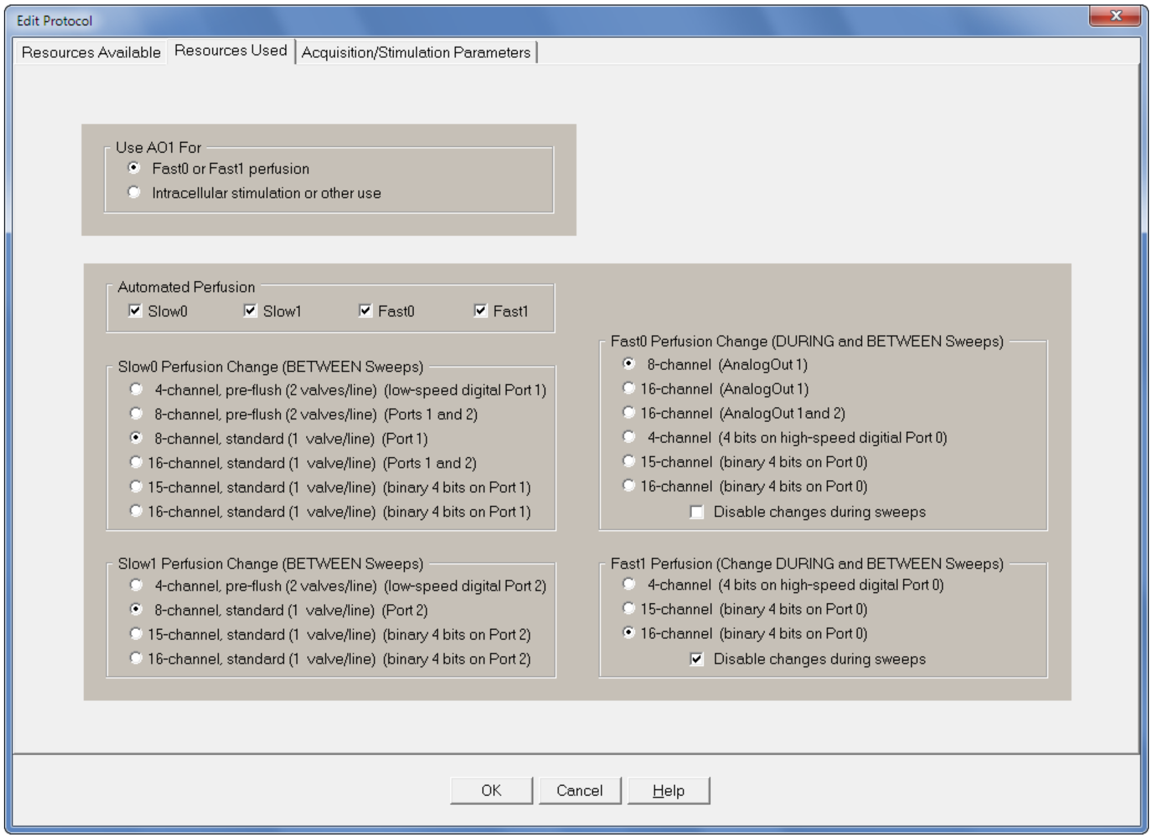
Fast0 Perfusion Ch On when MainProtocol is Off
Ch 1 Tube1
Click Apply to change to new Perfusion Channel
Apply Ch 1 Tube1
☐ Override Perfusion Ch while Protocol runs
☐ Set Ch to last Perfuse when Protocol stops

Fast1 Perfusion Ch On when MainProtocol is Off
Ch 1 Tube2 Ag3
Click Apply to change to new Perfusion Channel
Apply Ch 1 Tube2 Ag3
☐ Override Perfusion Ch while Protocol runs
☐ Set Ch to last Perfuse when Protocol stops

Fig. 9.4.2.1. WinLTP control of triple-line perfusion with a Stepper. (Left) Slow0, Slow1 and Fast1 Perfusion controls three valve controllers, and the Fast0 Perfusion controls the stepper which controls whether solution from Tube0 (blue), Tube1 (brown) or Tube 2 (fuchsia) is applied to the cell. The Fast0 Stepper therefore controls whether antagonist (Slow1) or agonist (Slow0 or Fast1) is applied. (Right) Manual control of perfusion channels panel.

The way the Triple-Line/Stepper perfusion system is set up in the ResourcesUsed tabsheet is similar to that for the Dual-Line/Stepper system except that the Fast1 perfusion checkbox is also checked so that Fast1 perfusion is used (Fig. 9.4.2.2).

A



Edit Protocol

Resources Available | Resources Used | Acquisition/Stimulation Parameters

Use AO1 For
- [✓ Fast0 or Fast1 perfusion]
- Intracellular stimulation or other use

Automated Perfusion
[✓ Slow0] [✓ Slow1] [✓ Fast0] [✓ Fast1]

Slow0 Perfusion Change (BETWEEN Sweeps)
- 4-channel, pre-flush (2 valves/line) (low-speed digital Port 1)
- 8-channel, pre-flush (2 valves/line) (Ports 1 and 2)
- [✓ 8-channel, standard (1 valve/line) (Port 1)]
- 16-channel, standard (1 valve/line) (Ports 1 and 2)
- 15-channel, standard (1 valve/line) (binary 4 bits on Port 1)
- 16-channel, standard (1 valve/line) (binary 4 bits on Port 1)

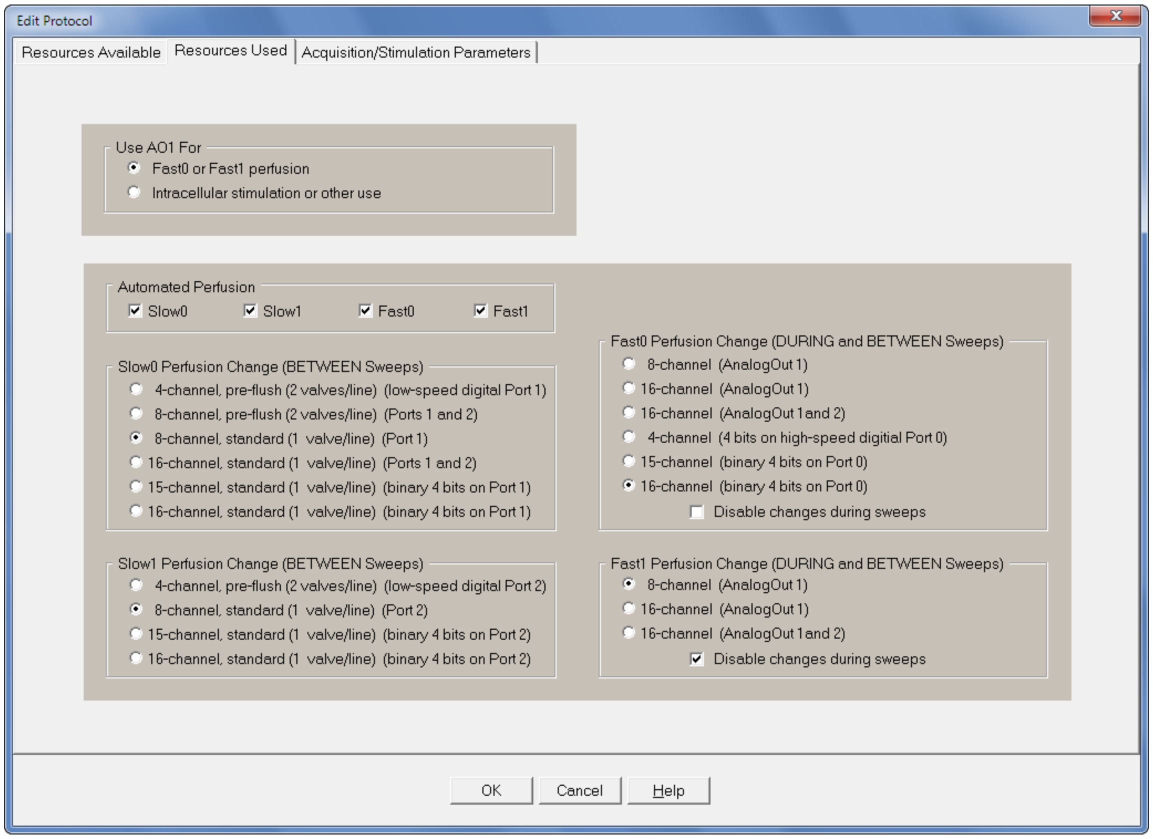
Fast0 Perfusion Change (DURING and BETWEEN Sweeps)
- [✓ 8-channel (AnalogOut 1)]
- 16-channel (AnalogOut 1)
- 16-channel (AnalogOut 1 and 2)
- 4-channel (4 bits on high-speed digitial Port 0)
- 15-channel (binary 4 bits on Port 0)
- 16-channel (binary 4 bits on Port 0)
- Disable changes during sweeps

Slow1 Perfusion Change (BETWEEN Sweeps)
- 4-channel, pre-flush (2 valves/line) (low-speed digital Port 2)
- [✓ 8-channel, standard (1 valve/line) (Port 2)]
- 15-channel, standard (1 valve/line) (binary 4 bits on Port 2)
- 16-channel, standard (1 valve/line) (binary 4 bits on Port 2)

Fast1 Perfusion (Change DURING and BETWEEN Sweeps)
- 4-channel (4 bits on high-speed digital Port 0)
- 15-channel (binary 4 bits on Port 0)
- [✓ 16-channel (binary 4 bits on Port 0)]
- [✓ Disable changes during sweeps]

OK | Cancel | Help

B



Edit Protocol

Resources Available | Resources Used | Acquisition/Stimulation Parameters

Use AO1 For
- [✓ Fast0 or Fast1 perfusion]
- Intracellular stimulation or other use

Automated Perfusion
[✓ Slow0] [✓ Slow1] [✓ Fast0] [✓ Fast1]

Slow0 Perfusion Change (BETWEEN Sweeps)
- 4-channel, pre-flush (2 valves/line) (low-speed digital Port 1)
- 8-channel, pre-flush (2 valves/line) (Ports 1 and 2)
- [✓ 8-channel, standard (1 valve/line) (Port 1)]
- 16-channel, standard (1 valve/line) (Ports 1 and 2)
- 15-channel, standard (1 valve/line) (binary 4 bits on Port 1)
- 16-channel, standard (1 valve/line) (binary 4 bits on Port 1)

Fast0 Perfusion Change (DURING and BETWEEN Sweeps)
- 8-channel (AnalogOut 1)
- 16-channel (AnalogOut 1)
- 16-channel (AnalogOut 1 and 2)
- 4-channel (4 bits on high-speed digitial Port 0)
- 15-channel (binary 4 bits on Port 0)
- [✓ 16-channel (binary 4 bits on Port 0)]
- Disable changes during sweeps

Slow1 Perfusion Change (BETWEEN Sweeps)
- 4-channel, pre-flush (2 valves/line) (low-speed digital Port 2)
- [✓ 8-channel, standard (1 valve/line) (Port 2)]
- 15-channel, standard (1 valve/line) (binary 4 bits on Port 2)
- 16-channel, standard (1 valve/line) (binary 4 bits on Port 2)

Fast1 Perfusion Change (DURING and BETWEEN Sweeps)
- [✓ 8-channel (AnalogOut 1)]
- 16-channel (AnalogOut 1)
- 16-channel (AnalogOut 1 and 2)
- [✓ Disable changes during sweeps]

OK | Cancel | Help

Fig. 9.4.2.2. The ResourcesUsed tabsheet configured to run Triple-Line/Stepper perfusion. AO1 is used for Fast0 or Fast1 perfusion. All the Slow0, Slow1, Fast0 and Fast1 perfusion checkboxes are checked. The Slow0 and Slow1 output to valve controllers is digital. In A, the Fast0 output to the stepper is digital 4 bit binary, and the Fast1 output to a valve controller is analog AO1. In B, the Fast0 output to the stepper is analog AO1, and the Fast1 output to a valve controller is digitial 4 bit binary.

The Slow0 and Slow1 output to valve controllers is digital and identical to the Dual-Line/Stepper system. However, which Fast0/Fast1 perfusion output control is high-speed digital using Port0 and which is analog (using AO1 or AO1+AO2) is determined by the choice made in the Fast0 Perfusion Change radiobutton group. As shown in Fig. 9.4.2.2A, if the Fast0 output to the stepper is chosen to be high-speed digital Port0 (in this case 4 bit binary), then the Fast1 output to a valve controller has to be analog (in this case AO1). As shown in Fig. 9.4.2.2B, if the Fast0 output to the stepper is chosen to be analog (in this case AO1), then the Fast1 output to a valve controller has to be high-speed digital Port0 (in this case 4 bit binary).

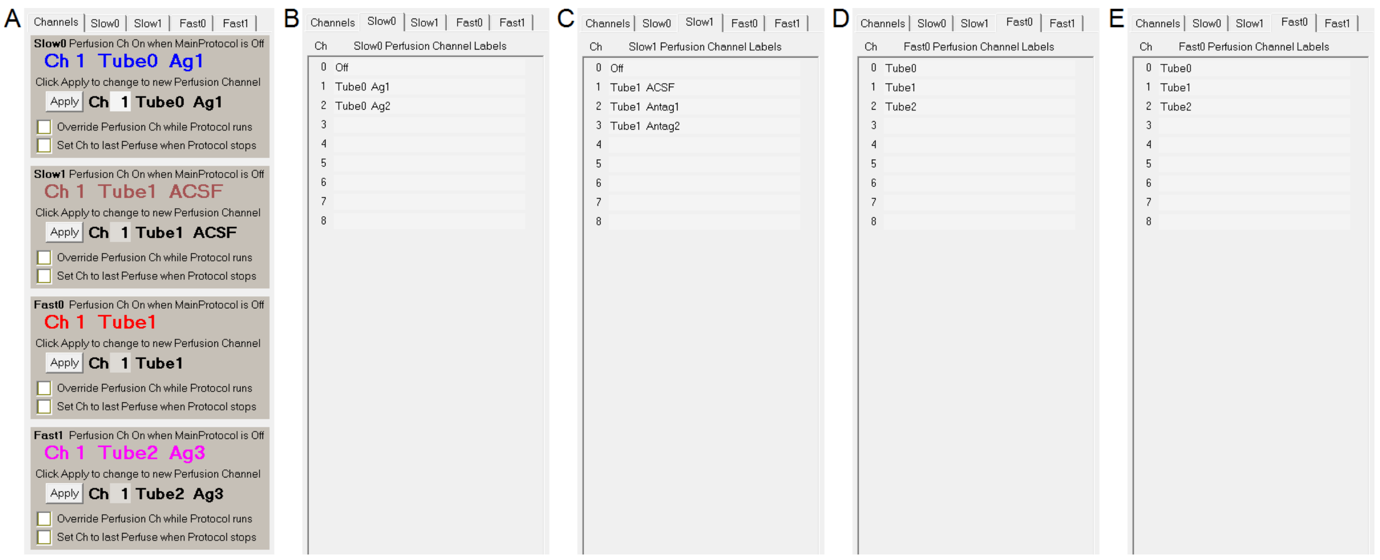
Setting up the perfusion channel label information (Fig. 9.4.2.3, B-E) is similar to that for the Dual-Line/Stepper system except for the addition of Fast1 perfusion solutions.

For the Fast1 (Tube 2 valve controller), the Fast1 Perfuse event in the Protocol Builder can either output agonist 3 or agonist 4 (‘Tube2 Ag3’ or ‘Tube2 Ag4’).

The Channels tabsheet for the Triple-Line/Stepper system is identical to the Dual-Line/Stepper system except for the addition of the Fast1 panel (Fig. 9.4.2.3A at the bottom).

The Triple-Line/Stepper example (Fig. 9.4.2.4) is also similar to the Dual-Line/Stepper example (Fig. 9.4.1.4) except that after Slow0 Perfuse Tube0 Ag1 and Ag2 are rapidly perfused onto the cell by the Fast0 stepper changing from Ch1 to Ch0 and back to Ch1, then Fast1 Perfuse Tube2 Ag3 and Ag4 are rapidly perfused onto the cell by the Fast0 stepper changing from Ch1 to Ch2 and back to Ch1.

This is most clearly shown in the ‘Continuous Acquisition’ panel where the sequence of perfusion solutions are Antag1, Ag1 Antag1, Ag2, Antag1, Ag3 Antag1, Ag4, then Antag2 Ag1 Antag2 Ag2, Antag2 Ag3 Antag2 Ag4.



Slow0 Perfusion Ch On when MainProtocol is Off
Ch 1 Tube0 Ag1
Click Apply to change to new Perfusion Channel
Apply Ch 1 Tube0 Ag1
☐ Override Perfusion Ch while Protocol runs
☐ Set Ch to last Perfuse when Protocol stops

Slow1 Perfusion Ch On when MainProtocol is Off
Ch 1 Tube1 ACSF
Click Apply to change to new Perfusion Channel
Apply Ch 1 Tube1 ACSF
☐ Override Perfusion Ch while Protocol runs
☐ Set Ch to last Perfuse when Protocol stops

Fast0 Perfusion Ch On when MainProtocol is Off
Ch 1 Tube1
Click Apply to change to new Perfusion Channel
Apply Ch 1 Tube1
☐ Override Perfusion Ch while Protocol runs
☐ Set Ch to last Perfuse when Protocol stops

Fast1 Perfusion Ch On when MainProtocol is Off
Ch 1 Tube2 Ag3
Click Apply to change to new Perfusion Channel
Apply Ch 1 Tube2 Ag3
☐ Override Perfusion Ch while Protocol runs
☐ Set Ch to last Perfuse when Protocol stops

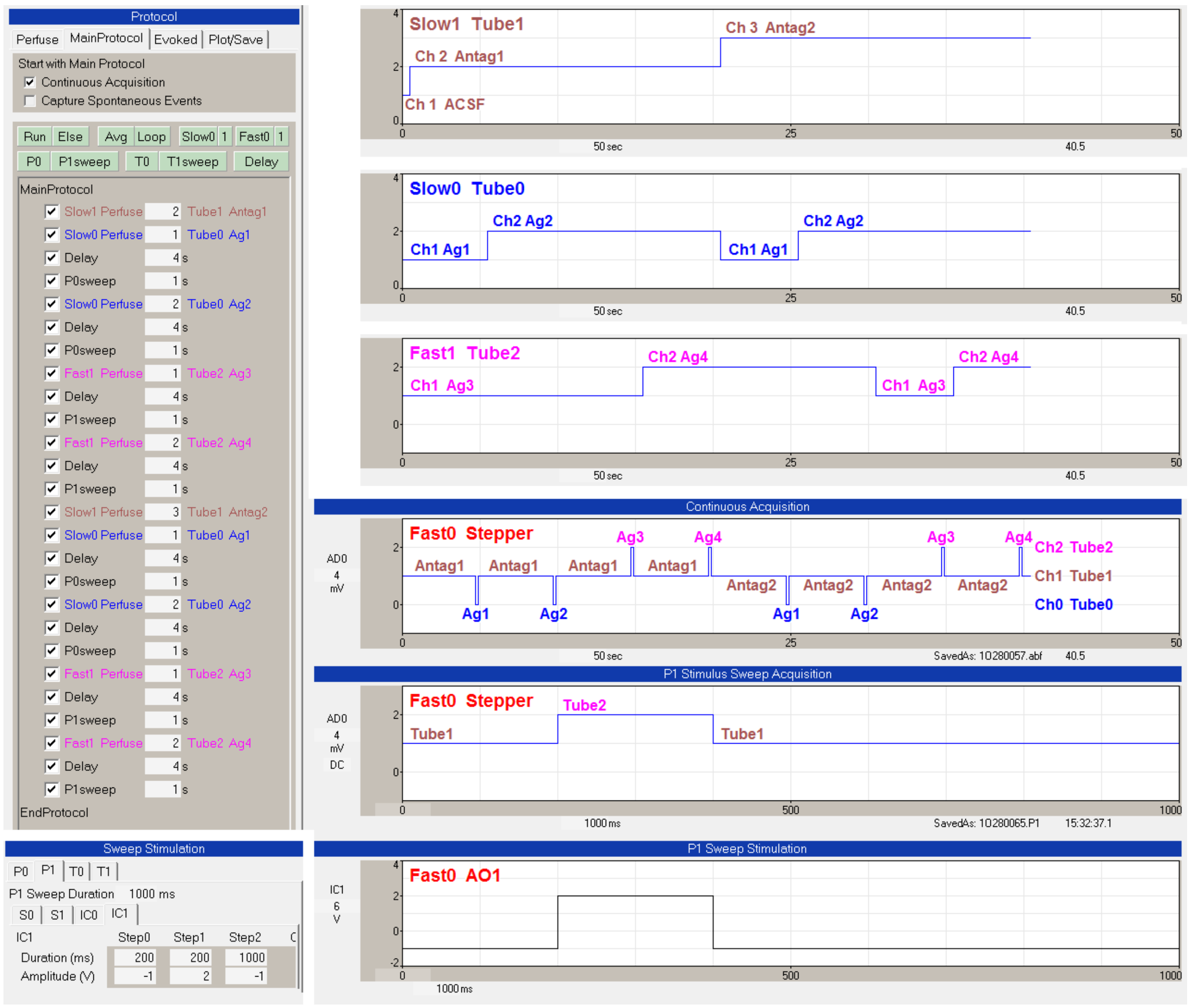
Ch	Slow0 Perfusion Channel Labels
0	Off
1	Tube0 Ag1
2	Tube0 Ag2
3	
4	
5	
6	
7	
8	

Ch	Slow1 Perfusion Channel Labels
0	Off
1	Tube1 ACSF
2	Tube1 Antag1
3	Tube1 Antag2
4	
5	
6	
7	
8	

Ch	Fast0 Perfusion Channel Labels
0	Tube0
1	Tube1
2	Tube2
3	
4	
5	
6	
7	
8	

Ch	Fast0 Perfusion Channel Labels
0	Tube0
1	Tube1
2	Tube2
3	
4	
5	
6	
7	
8	

Fig. 9.4.2.3. Setting up the perfusion channel number and label information for the Triple-Line/Stepper example.



Protocol

Perfuse | MainProtocol | Evoked | Plot/Save

Start with Main Protocol

☑ Continuous Acquisition

☐ Capture Spontaneous Events

Run | Else | Avg | Loop | Slow0 1 | Fast0 1

P0 | P1sweep | T0 | T1sweep | Delay

MainProtocol

☑	Slow1 Perfuse	2	Tube1 Antag1
☑	Slow0 Perfuse	1	Tube0 Ag1
☑	Delay	4	s
☑	P0sweep	1	s
☑	Slow0 Perfuse	2	Tube0 Ag2
☑	Delay	4	s
☑	P0sweep	1	s
☑	Fast1 Perfuse	1	Tube2 Ag3
☑	Delay	4	s
☑	P1sweep	1	s
☑	Fast1 Perfuse	2	Tube2 Ag4
☑	Delay	4	s
☑	P1sweep	1	s
☑	Slow1 Perfuse	3	Tube1 Antag2
☑	Slow0 Perfuse	1	Tube0 Ag1
☑	Delay	4	s
☑	P0sweep	1	s
☑	Slow0 Perfuse	2	Tube0 Ag2
☑	Delay	4	s
☑	P0sweep	1	s
☑	Fast1 Perfuse	1	Tube2 Ag3
☑	Delay	4	s
☑	P1sweep	1	s
☑	Fast1 Perfuse	2	Tube2 Ag4
☑	Delay	4	s
☑	P1sweep	1	s

EndProtocol

Sweep Stimulation

P0 | P1 | T0 | T1

P1 Sweep Duration 1000 ms

S0 | S1 | IC0 | IC1

IC1	Step0	Step1	Step2
Duration (ms)	200	200	1000
Amplitude (V)	-1	2	-1

Fig. 9.4.2.4. Running the Triple-Line/Stepper example. The ‘Slow0 Perfuse’, ‘Slow1 Perfuse’ and ‘Fast1 Perfuse’ events in the Protocol Builder (upper left) show when the Slow0/Tube0, Slow1/Tube1 solutions are changed. The upper right panel shows the Slow1 Tube1 switch from ‘Ch1 ACSF’ to ‘Ch2 Antag1’ to ‘Ch3 Antag2’. The next panel below shows the Slow0 Tube0 switch between ‘Ch1 Ag1’ and ‘Ch2 Ag2’. The next panel below that shows the Fast1 Tube2 switch between ‘Ch1 Ag3’ and ‘Ch2 Ag4’. The ‘Continuous Acquisition’ and the ‘P0 Stimulus Sweep Acquisition’ panel show recordings of the AO1 voltage shifts temporarily changing from ‘Ch1 Tube1’ to ‘Ch0 Tube0’ or ‘Ch2 Tube2’. This results in the solutions switching between Antag1, Ag1, Antag1, Ag2, Antag1, Ag3, Antag1, Ag4, then Antag2, Ag1, Antag2, Ag2, Antag2, Ag3, Antag2, Ag4. The Sweep Stimulation bottom panels show change of AO1 (ie IC1) voltage output during a P1sweep. The -1V means to take whatever the current Fast0 output is BETWEEN sweeps (and when the MainProtocol is off), ie 2V or Ch2.

CHAPTER 10 – Experimental Log

10.1 Capabilities of the Experimental Log

The primary purpose of the Experimental Log is to show and record when important events occur in the experiment. These events can include automatically generated information such as when a titanic stimulation was delivered or when Low Frequency Stimulation was run, and will include the time and name of which stimulation/acquisition ADsweep files delivered the particular stimulation. The events can include manually input information such as when solutions were changed. The Experimental Log will also automatically record important values such as the patch pipette series resistance (Rs) from the SealTest protocol.

Each time WinLTP is run, the Experimental Log is saved to a single *.log file in the current DataFolder. However, **this ability to save the log is only available in the Advanced Version or during the DemoTrial period.**

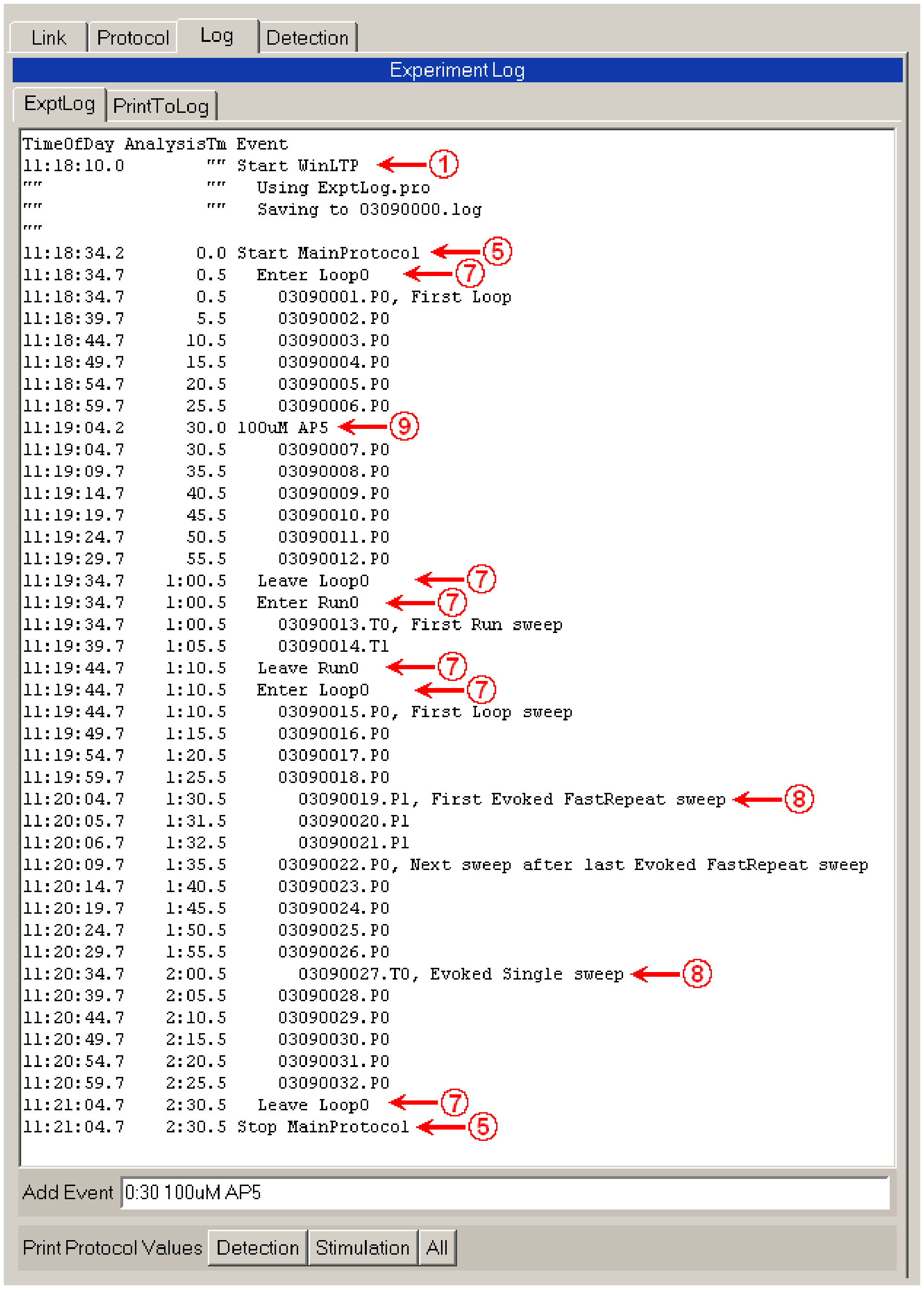
The Experimental Log should substantially decrease the amount of information having to be manually entered into your lab book, and it can serve as a check that the information manually entered into your lab book is correct.

Specifically the Experimental Log shows:

1) When WinLTP was started and exited.
2) When the SealTest protocol was entered and returned, and it will print the PulseAmp, Rpipette, Rseal, Rs, Rm and Idc values from the SealTest.
3) When
   a) the Spreadsheet was cleared
   b) an AmpFile was saved
   c) a Protocol file was either opened or saved.
4) When a Continuous Acquisition protocol was started and stopped.
5) When the Main Protocol was started and stopped.
6) When particular stimulation/acquisition ADsweeps have occurred (P0sweeps, P1sweeps, T0sweeps, T1sweeps, AP0sweeps and AP1sweeps). For example, if the titanic LTP induction stimulation was in a T0sweep, then the Experimental Log will record when that stimulation was delivered.
7) When there was a change in protocol flow, such as the starting and stopping of Loops, AvgLoops, Runs, Run of Run/Else, Else of Run/Else.
8) When Evoked Single Events (i.e. single ADsweeps) are delivered, and Repeat Events (i.e. multiple ADsweeps) delivered.
9) When a Perfusion solution was changed for Automated Perfusion
9) And finally, to be able to manually enter and record the time and solution change by keyboard input.

Fig. 10.1.1 shows an example of an Experimental Log output from an ersatz short protocol. The Log specifically shows when WinLTP was started, when the Main Protocol was started and stopped, when there was a change in protocol flow for starting and stopping Loops and Runs, when Evoked Repeat Sweeps and Evoked Single Sweeps were delivered, and the time when a solution change was manually entered by keyboard input (“0:30 100uM AP5” entered at the Add Event line).

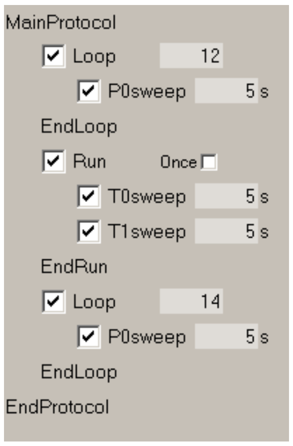
A



```
TimeOfDay AnalysisTm Event
11:18:10.0        ""  Start WinLTP
""                ""    Using ExptLog.pro
""                ""    Saving to 03090000.log
""
11:18:34.2       0.0  Start MainProtocol
11:18:34.7       0.5    Enter Loop0
11:18:34.7       0.5      03090001.P0, First Loop
11:18:39.7       5.5      03090002.P0
11:18:44.7      10.5      03090003.P0
11:18:49.7      15.5      03090004.P0
11:18:54.7      20.5      03090005.P0
11:18:59.7      25.5      03090006.P0
11:19:04.2      30.0  100uM AP5
11:19:04.7      30.5      03090007.P0
11:19:09.7      35.5      03090008.P0
11:19:14.7      40.5      03090009.P0
11:19:19.7      45.5      03090010.P0
11:19:24.7      50.5      03090011.P0
11:19:29.7      55.5      03090012.P0
11:19:34.7    1:00.5    Leave Loop0
11:19:34.7    1:00.5    Enter Run0
11:19:34.7    1:00.5      03090013.T0, First Run sweep
11:19:39.7    1:05.5      03090014.T1
11:19:44.7    1:10.5    Leave Run0
11:19:44.7    1:10.5    Enter Loop0
11:19:44.7    1:10.5      03090015.P0, First Loop sweep
11:19:49.7    1:15.5      03090016.P0
11:19:54.7    1:20.5      03090017.P0
11:19:59.7    1:25.5      03090018.P0
11:20:04.7    1:30.5        03090019.P1, First Evoked FastRepeat sweep
11:20:05.7    1:31.5        03090020.P1
11:20:06.7    1:32.5        03090021.P1
11:20:09.7    1:35.5      03090022.P0, Next sweep after last Evoked FastRepeat sweep
11:20:14.7    1:40.5      03090023.P0
11:20:19.7    1:45.5      03090024.P0
11:20:24.7    1:50.5      03090025.P0
11:20:29.7    1:55.5      03090026.P0
11:20:34.7    2:00.5        03090027.T0, Evoked Single sweep
11:20:39.7    2:05.5      03090028.P0
11:20:44.7    2:10.5      03090029.P0
11:20:49.7    2:15.5      03090030.P0
11:20:54.7    2:20.5      03090031.P0
11:20:59.7    2:25.5      03090032.P0
11:21:04.7    2:30.5    Leave Loop0
11:21:04.7    2:30.5  Stop MainProtocol
```

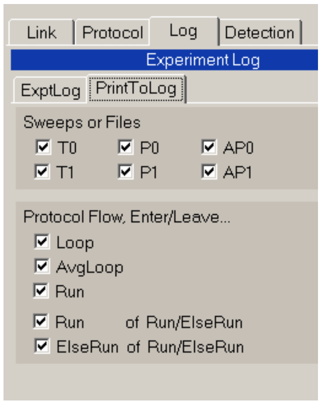
Add Event 0:30 100uM AP5

B



```
MainProtocol
   [✓] Loop          12
      [✓] P0sweep     5 s
   EndLoop
   [✓] Run      Once [ ]
      [✓] T0sweep     5 s
      [✓] T1sweep     5 s
   EndRun
   [✓] Loop          14
      [✓] P0sweep     5 s
   EndLoop
EndProtocol
```

C



Sweeps or Files

[✓] T0 [✓] P0 [✓] AP0

[✓] T1 [✓] P1 [✓] AP1

Protocol Flow, Enter/Leave...

[✓] Loop

[✓] AvgLoop

[✓] Run

[✓] Run of Run/ElseRun

[✓] ElseRun of Run/ElseRun

Fig. 10.1.1. Example of an Experimental Log output from an ersatz short protocol. A) This Log specifically shows: 1) When WinLTP was started, 5) When the Main Protocol was started and stopped, 7) When there was a change in protocol flow for starting and stopping Loops and Runs, 8) When Evoked Repeat Sweeps and Evoked Single Sweeps were delivered, and 9) the time (0 min, 30 sec) when a solution change was manually entered by keyboard input (“0:30 100uM AP5” entered at the Add Event line). LTP induction stimulation could be caused by the 03090013.T0 and 03090014.T1 sweeps in the Run event, and by the 03090027.T0 Evoked Single Sweep stimulation, and LTD induction could be caused by the Evoked Repeat 03090019.P1 to 03090021.P1 Sweeps (actually a few hundred more would be necessary). Note the ‘Detection’, ‘Stimulation’ and ‘All’ buttons to print protocol values. B) Protocol run for the Experimental Log in A. C) The PrintToLog tabsheet which determines first, which ADsweeps are to be printed, and second which Loops, AvgLoops, Runs and Run and Else of Run/Elses to be printed. In this example, all ADsweeps are chosen to be printed.

10.2 Use the PrintToLog tabsheet to control printing of useful information

Sometimes it is helpful to only include certain ADsweep and protocol flow information. This is controlled by the PrintToLog tabsheet (Fig. 10.1.1C) which determines first, which ADsweeps are to be printed, and second which Loops, AvgLoops, Runs and Run and Else of Run/Elses to be printed.

In general, Loops, Runs, and Runs and ElseRuns of Run/ElseRun constructs contain useful protocol information.

10.3 Printing AvgLoops is usually not useful

In contrast, AvgLoops rarely contain useful information because they are just a repeat of the same sweep stimulation.

For example in the following protocol (Fig. 10.3.1a), a standard protocol for obtaining baseline activity with signal averaging prior to any application of drugs or stimulation, the AvgLoop can clearly be unchecked.

```
MainProtocol
    ☑ Loop       [  3]
        ☑ AvgLoop   [ 4]
            ☑ P0sweep [  5]s
        EndAvgLoop
    EndLoop
EndProtocol
```

Fig. 10.3.1a. Protocol for finding baseline while doing averaging loops.

Fig 10.3.1b show the Log output with Loop, AvgLoop, P0 and AP0 checked.

```
16:02:55.0       0.0 Start MainProtocol
16:02:55.5       0.5   Enter Loop0
16:02:55.5       0.5     Enter AvgLoop1
16:02:55.5       0.5       06210264.P0, First Loop/AvgLoop sweep
16:03:00.5       5.5       06210265.P0
16:03:05.5      10.5       06210266.P0
16:03:10.5      15.5       06210267.P0
16:03:10.5      15.5       06210267.AP0
16:03:15.5      20.5     Leave AvgLoop1
16:03:15.5      20.5     Enter AvgLoop1
16:03:15.5      20.5       06210268.P0, First AvgLoop sweep
16:03:20.5      25.5       06210269.P0
16:03:25.5      30.5       06210270.P0
16:03:30.5      35.5       06210271.P0
16:03:30.5      35.5       06210271.AP0
16:03:35.5      40.5     Leave AvgLoop1
16:03:35.5      40.5     Enter AvgLoop1
16:03:35.5      40.5       06210272.P0, First AvgLoop sweep
16:03:40.5      45.5       06210273.P0
16:03:45.5      50.5       06210274.P0
16:03:50.5      55.5       06210275.P0
16:03:50.5      55.5       06210275.AP0
16:03:55.5    1:00.5     Leave AvgLoop1
16:03:55.5    1:00.5   Leave Loop0
16:03:55.5    1:00.5 Stop MainProtocol
```

Fig. 10.3.1b. Log output with Loop, AvgLoop, P0 and AP0 checked.

When the AvgLoop is unchecked, but Loop, P0 and AP0 remain checked (Fig. 10.3.1c), the experiment becomes somewhat clearer. The AvgLoop is clearly superfluous.

```
16:05:32.3       0.0 Start MainProtocol
16:05:32.8       0.5   Enter Loop0
16:05:32.8       0.5       06210277.P0, First Loop/AvgLoop sweep
16:05:37.8       5.5       06210278.P0
16:05:42.8      10.5       06210279.P0
16:05:47.8      15.5       06210280.P0
16:05:47.8      15.5       06210280.AP0
16:05:52.8      20.5       06210281.P0, First AvgLoop sweep
16:05:57.8      25.5       06210282.P0
16:06:02.8      30.5       06210283.P0
16:06:07.8      35.5       06210284.P0
16:06:07.8      35.5       06210284.AP0
16:06:12.8      40.5       06210285.P0, First AvgLoop sweep
16:06:17.8      45.5       06210286.P0
16:06:22.8      50.5       06210287.P0
16:06:27.8      55.5       06210288.P0
16:06:27.8      55.5       06210288.AP0
16:06:32.8    1:00.5     Leave AvgLoop1
16:06:32.8    1:00.5   Leave Loop0
16:06:32.8    1:00.5 Stop MainProtocol
```

Fig. 10.3.1c. AvgLoop is unchecked, but Loop, P0 and AP0 remain checked.

Sometimes P0 and P1 sweep stimulation is crucial to understanding (such as during Fast Repeat Sweep LTD stimulation), but often this is not the case and they can be unchecked. So, when the P0 as well as the AvgLoop is unchecked, but the Loop and AP0 are still checked (Fig. 10.3.1d), the occurrence of AP0 files provides enough information of experimental flow.

```
16:11:26.5       0.0 Start MainProtocol
16:11:27.0       0.5   Enter Loop0
16:11:27.0       0.5       06210293.P0, First Loop/AvgLoop sweep
16:11:42.0      15.5       06210296.AP0
16:12:02.0      35.5       06210300.AP0
16:12:22.0      55.5       06210304.AP0
16:12:27.0    1:00.5     Leave AvgLoop1
16:12:27.0    1:00.5   Leave Loop0
16:12:27.0    1:00.5 Stop MainProtocol
```

Fig. 10.3.1d. AvgLoop and P0 are unchecked; only Loop and AP0 remain checked. Note that the first P0sweep output during the protocol run 06210293.P0 was also printed, as was the final ‘Leave AvgLoop1’ and ‘Leave Loop0’ when preceding the final ‘Stop MainProtocol’.

10.4 Print only sweeps that output important induction stimulation

The sweep stimulations containing induction stimulation such as tetanus trains *usually* put into T0 and T1 ‘train’ sweeps is useful information, and if so should probably remain checked. Fig. 10.4.1a shows a similar protocol to Fig. 10.3.1a except the Loop[3] ‘baseline’ portion is followed by three T0sweeps each containing train stimulation to induce LTP, followed by a Loop[10] post-induction period.

```
MainProtocol
   ☑ Loop      [  3]
      ☑ AvgLoop  [ 4]
         ☑ P0sweep [  5]s
      EndAvgLoop
   EndLoop
   ☑ T0sweep [  5]s
   ☑ Loop      [  10]
      ☑ AvgLoop  [ 4]
         ☑ P0sweep [  5]s
      EndAvgLoop
   EndLoop
EndProtocol
```

Fig. 10.4.1a.

As with Fig. 10.3.1d, AvgLoop and P0 are unchecked, and Loop and AP0 are checked, but also T0 must be checked (Fig. 10.4.1b).

```
12:30:48.6       0.0 Start MainProtocol
12:30:49.1       0.5   Enter Loop0
12:30:49.1       0.5       06220020.P0, First Loop/AvgLoop sweep
12:31:04.1      15.5       06220023.AP0
12:31:24.1      35.5       06220027.AP0
12:31:44.1      55.5       06220031.AP0
12:31:49.1    1:00.5   Leave Loop0
12:31:49.1    1:00.5   06220032.T0, Next sweep after end of Loop/AvgLoop
12:31:54.1    1:05.5   06220033.T0
12:31:59.1    1:10.5   06220034.T0
12:32:04.1    1:15.5   Enter Loop0
12:32:04.1    1:15.5       06220035.P0, First Loop/AvgLoop sweep
12:32:19.1    1:30.5       06220038.AP0
12:32:39.1    1:50.5       06220042.AP0
12:32:59.1    2:10.5       06220046.AP0
12:33:19.1    2:30.5       06220050.AP0
12:33:39.1    2:50.5       06220054.AP0
12:33:59.1    3:10.5       06220058.AP0
12:34:19.1    3:30.5       06220062.AP0
12:34:39.1    3:50.5       06220066.AP0
12:34:59.1    4:10.5       06220070.AP0
12:35:19.1    4:30.5       06220074.AP0
12:35:24.1    4:35.5     Leave AvgLoop1
12:35:24.1    4:35.5   Leave Loop0
12:35:24.1    4:35.5 Stop MainProtocol
```

Fig. 10.4.1b. AvgLoop and P0 are unchecked, Loop and AP0 are checked, and T0 must be checked.

Sometimes Loops are also not needed. Fig. 10.4.1c shows the same protocol in Fig. 10.4.1a but with Loop also unchecked. Note that the first P0sweep at the beginning of the second loop 06220091.P0 is still printed.

```
12:35:49.4       0.0 Start MainProtocol
12:35:49.9       0.5       06220076.P0, First Loop/AvgLoop sweep
12:36:04.9      15.5       06220079.AP0
12:36:24.9      35.5       06220083.AP0
12:36:44.9      55.5       06220087.AP0
12:36:49.9    1:00.5   06220088.T0, Next sweep after end of Loop/AvgLoop
12:36:54.9    1:05.5   06220089.T0
12:36:59.9    1:10.5   06220090.T0
12:37:04.9    1:15.5       06220091.P0, First Loop/AvgLoop sweep
12:37:19.9    1:30.5       06220094.AP0
12:37:39.9    1:50.5       06220098.AP0
12:37:59.9    2:10.5       06220102.AP0
12:38:19.9    2:30.5       06220106.AP0
12:38:39.9    2:50.5       06220110.AP0
12:38:59.9    3:10.5       06220114.AP0
12:39:19.9    3:30.5       06220118.AP0
12:39:39.9    3:50.5       06220122.AP0
12:39:59.9    4:10.5       06220126.AP0
12:40:19.9    4:30.5       06220130.AP0
12:40:24.9    4:35.5     Leave AvgLoop1
12:40:24.9    4:35.5   Leave Loop0
12:40:24.9    4:35.5 Stop MainProtocol
```

Fig. 10.4.1c. Loop, AvgLoop and P0 are unchecked, AP0 and T0 are checked.

10.5 Printing P0 and P1 sweeps is not necessary for Evoked RepeatSweep stimulation

Even though P0 and P1 sweep stimulation can be crucial to understanding experimental protocol flow (such as during Fast Repeat Sweep LTD stimulation), P0 and P1 checkboxes need not necessarily be checked to obtain the information. For example, if Fast Repeat Sweep LTD stimulation is evoked by clicking on the ‘Repeat Sweep’ button, the first LTD sweep will be marked, and the first sweep after the LTD stimulation will also be marked.

Fig 10.5.1a shows the simple protocol run outputting 20 P0sweeps at 5 sec periods.

```
MainProtocol
    ☑ Loop      [ 20]
        ☑ P0sweep [  5]s
    EndLoop
EndProtocol
```

Fig. 10.5.1a.

However, after the 10th P0sweep, an ersatz Fast Repeat LTD stimulation of five P1sweeps every 1 sec was manually evoked by clicking the ‘P1’ RepeatSweep button. The output of this protocol is shown in Fig. 10.5.1b. Both the P0 and P1 checkboxes were checked. Sweep 06220162.P0 at 45.5 sec was the last P0sweep before the Evoked FastRepeat stimulation, sweep 06220163.P1 at 50.5 sec is the first Evoked FastRepeat stimulation, sweep 06220167.P1 at 54.5 sec is the last Evoked FastRepeat, and sweep 06220168.P0 at 55.5 sec is the first post-FastRepeat sweep.

```
13:18:57.8       0.0 Start MainProtocol
13:18:58.3       0.5   Enter Loop0
13:18:58.3       0.5     06220153.P0, First Loop sweep
13:19:03.3       5.5     06220154.P0
13:19:08.3      10.5     06220155.P0
13:19:13.3      15.5     06220156.P0
13:19:18.3      20.5     06220157.P0
13:19:23.3      25.5     06220158.P0
13:19:28.3      30.5     06220159.P0
13:19:33.3      35.5     06220160.P0
13:19:38.3      40.5     06220161.P0
13:19:43.3      45.5     06220162.P0
13:19:48.3      50.5       06220163.P1, First Evoked FastRepeat sweep
13:19:49.3      51.5       06220164.P1
13:19:50.3      52.5       06220165.P1
13:19:51.3      53.5       06220166.P1
13:19:52.3      54.5       06220167.P1
13:19:53.3      55.5     06220168.P0, Next sweep after last Evoked FastRepeat sweep
13:19:58.3    1:00.5     06220169.P0
13:20:03.3    1:05.5     06220170.P0
13:20:08.3    1:10.5     06220171.P0
13:20:13.3    1:15.5     06220172.P0
13:20:18.3    1:20.5     06220173.P0
13:20:23.3    1:25.5     06220174.P0
13:20:28.3    1:30.5     06220175.P0
13:20:33.3    1:35.5     06220176.P0
13:20:38.3    1:40.5     06220177.P0
13:20:43.3    1:45.5   Leave Loop0
13:20:43.3    1:45.5 Stop MainProtocol
```

Fig. 10.5.1b. Log output for the protocol in Fig. 10.5.1a. Both the P0 and P1 checkboxes were checked. After the 10th P0sweep, an ersatz Fast Repeat LTD stimulation of five P1sweeps every 1 sec was manually evoked by clicking the ‘P1’ Repeat button.

If the P0 and P1 checkboxes were unchecked, a much shorter output shown in Fig. 10.5.1c is obtained. Note however that the first sweep of the Evoked FastRepeat stimulation, sweep 06220163.P1 at 50.5 sec is shown, and the first sweep after the Evoked FastRepeat stimulation 06220168.P0 at 55.5 sec is also shown, so the relevant information of what sweeps are before, during and after the Evoked FastRepeat stimulation and the sweep times is available.

```
13:18:57.8       0.0 Start MainProtocol
13:18:58.3       0.5   Enter Loop0
13:18:58.3       0.5     06220153.P0, First Loop sweep
13:19:48.3      50.5       06220163.P1, First Evoked FastRepeat sweep
13:19:53.3      55.5     06220168.P0, Next sweep after last Evoked FastRepeat sweep
13:20:43.3    1:45.5   Leave Loop0
13:20:43.3    1:45.5 Stop MainProtocol
```

Fig. 10.5.1c. Log output for the protocol in Fig. 10.5.1a., but both the P0 and P1 checkboxes were unchecked. After the 10th P0sweep, an ersatz Fast Repeat LTD stimulation of five P1sweeps every 1 sec was manually evoked by clicking the ‘P1’ Repeat button.

Similarly, if FastRepeat Sweep LTD stimulation is caused by clicking a Run event with a Loop in it, the first LTD sweep will again be marked, and the first sweep after the LTD stimulation will again be marked.

10.6 Evoked Single Sweep stimulation is always shown

All Evoked Single Sweep Stimulation is always shown. For the protocol in Fig. 10.5.1a, if a single sweep, say a T0sweep, is manually evoked after 10 P0sweeps by clicking the ‘Single T0’ button, it will be recorded in the Log even though the T0 checkbox is not checked (Fig. 10.6.1a).

```
13:50:01.4       0.0 Start MainProtocol
13:50:01.9       0.5   Enter Loop0
13:50:01.9       0.5     06220234.P0, First Loop sweep
13:50:06.9       5.5     06220235.P0
13:50:11.9      10.5     06220236.P0
13:50:16.9      15.5     06220237.P0
13:50:21.9      20.5     06220238.P0
13:50:26.9      25.5     06220239.P0
13:50:31.9      30.5     06220240.P0
13:50:36.9      35.5     06220241.P0
13:50:41.9      40.5     06220242.P0
13:50:46.9      45.5     06220243.P0
13:50:51.9      50.5       06220244.T0, Evoked Single sweep
13:50:56.9      55.5     06220245.P0
13:51:01.9    1:00.5     06220246.P0
13:51:06.9    1:05.5     06220247.P0
13:51:11.9    1:10.5     06220248.P0
13:51:16.9    1:15.5     06220249.P0
13:51:21.9    1:20.5     06220250.P0
13:51:26.9    1:25.5     06220251.P0
13:51:31.9    1:30.5     06220252.P0
13:51:36.9    1:35.5     06220253.P0
13:51:41.9    1:40.5     06220254.P0
13:51:46.9    1:45.5   Leave Loop0
13:51:46.9    1:45.5 Stop MainProtocol
```

Fig. 10.6.1a. Log output for the protocol in Fig. 10.5.1a. The P0 checkbox were checked, and the T0 checkbox was unchecked. After the 10th P0sweep, a Evoked Single Sweep stimulation was manually evoked by clicking the ‘Single T0’ Repeat button.

If the P0 checkbox is now unchecked, and the T0 checkbox remains unchecked, the evoked T0sweep is still shown, but no P0sweeps either preceding or following the T0sweep will be shown, however, their number will be obvious (Fig. 10.6.1.b).

```
13:50:01.4       0.0 Start MainProtocol
13:50:01.9       0.5   Enter Loop0
13:50:01.9       0.5     06220234.P0, First Loop sweep
13:50:51.9      50.5       06220244.T0, Evoked Single sweep
13:51:46.9    1:45.5   Leave Loop0
13:51:46.9    1:45.5 Stop MainProtocol
```

Fig. 10.6.1b. Log output for the protocol in Fig. 10.5.1.a. The P0 and T0 checkboxes were both unchecked. After the 10th P0sweep, a Evoked Single Sweep stimulation was manually evoked by clicking the ‘Single T0’ Repeat button.

10.7 Manually Add Events (enter Solution Changes)

You can manually enter information such as the time, concentration and type of solution changes by putting the cursor on the Add Event edit box (see bottom of Fig. 10.1.1A) and typing in the information in the following form:

"min:sec.tenthsec HowMuch OfThis"

This information can be typed in before the event has occurred (subsequent Sweeps or Loops etc will be inserted at their correct time before or after the entered information time). Or information can be typed in after the event has occurred (where the information will be inserted at the correct time).

The Experimental Log can also be directly edited (without using the Add Event edit field), but this is generally not a good idea to do while the MainProtocol is running (use the Add Event field instead), only after the MainProtocol is finished.

Putting in the "min:sec" fields without the ".tenthsec" field is perfectly acceptable.

Putting in the "min" fields without the "sec.tenthsec" fields is also perfectly acceptable, provided the min field is a whole number.

The following examples are correct:

"0:5   AP5"
"0:5.5   AP5"

"59"   (but what's the point?)
"59   AP5"
"59   100 AP5"
"59   100uM AP5"

"59:30   AP5"
"59:30.5   AP5"

"77   AP5"
"77:30   AP5"
"77:30.5 AP5"

The following are incorrect	and should be entered as
"59.5 AP5"	"59:30 AP5"
"1:17:15 AP5"	"77:15 AP5"
"1:17:15.5 AP5"	"77:15.5 AP5"

These will elicit the following Error message:

"ERROR: Input must be in the form of AnalysisTime and message, "mm:ss.t plus message", eg "40 50uM AP5" or "40:59.9 50uM AP5""

The following is correct, but will be taken as 1 min and 17 seconds, not 1 hr, 17 min

"1:17   AP5"

The following is accepted	but is rounded down to the nearest tenth of a second
“59:30.55 AP5”	“59:30.5 AP5”

In WinLTP110, you had to have P0 and P1, and maybe T0 and T1, checkboxes checked so that you could know what sweep immediately followed when you manually entered a solution change by keyboard input.

```
MainProtocol
    ☑ Loop      [99999]
        ☑ P0sweep  [   5]s
    EndLoop
EndProtocol
```

Fig. 10.7.1a.

For the protocol in Fig. 10.7.1a running continuous P0sweeps every 5 seconds, and manually stopped after about 63 seconds, if the P0sweep checkbox was checked and if “0:30 AP5 was manually entered by keyboard input BEFORE the AddedEvent time of 0:30 at about 12 seconds after the MainProtocol started, the result in Fig 10.7.1b occurs.

```
16:37:30.0       0.0 Start MainProtocol
16:37:30.5       0.5   Enter Loop0
16:37:30.5       0.5     0N250051.P0, First Loop sweep
16:37:35.5       5.5     0N250052.P0
16:37:40.5      10.5     0N250053.P0
16:37:45.5      15.5     0N250054.P0
16:37:50.5      20.5     0N250055.P0
16:37:55.5      25.5     0N250056.P0
16:38:00.0      30.0 AP5
16:38:00.5      30.5     0N250057.P0, First sweep after AddedEvent
16:38:05.5      35.5     0N250058.P0
16:38:10.5      40.5     0N250059.P0
16:38:15.5      45.5     0N250060.P0
16:38:20.5      50.5     0N250061.P0
16:38:25.5      55.5     0N250062.P0
16:38:30.5    1:00.5     0N250063.P0
16:38:33.2    1:03.2 Stop MainProtocol
```

Fig. 10.7.1.b. Experimental Log output if “0:30 AP5” was entered BEFORE the AddedEvent time of 0:30 seconds, (e.g. at 0:12 seconds after start of MainProtocol) , and P0sweep was checked.

If “0:30 AP5” was manually entered by keyboard input AFTER the AddedEvent time of 0:30 at about 42 seconds after the MainProtocol started, the result in Fig 10.7.1c occurs.

```
  16:39:46.0       0.0 Start MainProtocol
  16:39:46.5       0.5   Enter Loop0
  16:39:46.5       0.5     0N250070.P0, First Loop sweep
  16:39:51.5       5.5     0N250071.P0
  16:39:56.5      10.5     0N250072.P0
  16:40:01.5      15.5     0N250073.P0
  16:40:06.5      20.5     0N250074.P0
  16:40:11.5      25.5     0N250075.P0
  16:40:16.0      30.0 AP5
  16:40:16.5      30.5     0N250076.P0
  16:40:21.5      35.5     0N250077.P0
  16:40:26.5      40.5     0N250078.P0
  16:40:31.5      45.5     0N250079.P0
  16:40:36.5      50.5     0N250080.P0
  16:40:41.5      55.5     0N250081.P0
  16:40:46.5    1:00.5     0N250082.P0
  16:40:48.9    1:02.9 Stop MainProtocol
```

Fig. 10.7.1.c. Experimental Log output if “0:30 AP5” was entered AFTER the AddedEvent time of 0:30 seconds, (e.g. at 0:42 seconds after start of MainProtocol) , and P0sweep was checked.

However, in WinLTP111, if you manually enter a solution change by keyboard input BEFORE or AFTER the first sweep following the solution change time, then this first sweep following the solution change time is automatically printed to the Experimental Log irregardless of whether that sweep checkbox was checked (Figs. 10.7.1.d and e). This is true for entering a solution change by keyboard input AFTER the first sweep following the solution change time as long as fewer than 5000 sweeps have occurred in between (because WinLTP remembers the time of only the last 5000 sweeps).

```
  16:47:27.0       0.0 Start MainProtocol
  16:47:27.5       0.5   Enter Loop0
  16:47:27.5       0.5     0N250099.P0, First Loop sweep
  16:47:57.0      30.0 AP5
  16:47:57.5      30.5     0N250105.P0, First sweep after AddedEvent
  16:48:30.6    1:03.6 Stop MainProtocol
```

Fig. 10.7.1.d. Experimental Log output if “0:30 AP5” was entered BEFORE the AddedEvent time of 0:30 seconds, (e.g. at 0:12 seconds after start of MainProtocol), and P0sweep was NOT checked

```
  16:51:44.0       0.0 Start MainProtocol
  16:51:44.5       0.5   Enter Loop0
  16:51:44.5       0.5     0N250115.P0, First Loop sweep
  16:52:14.0      30.0 AP5
  16:52:14.5      30.5     0N250121.P0, First sweep after AddedEvent
  16:52:47.5    1:03.5 Stop MainProtocol
```

Fig. 10.7.1.e. Experimental Log output if “0:30 AP5” was entered AFTER the AddedEvent time of 0:30 seconds, (e.g. at 0:42 seconds after start of MainProtocol), and P0sweep was NOT checked

The rules for Adding Events and printing the PxSweep immediately following the AddedEvent are summarized as follows:

1) The PxSweep following the AddedEvent will only be available during the current "Start MainProtool".
2) The PxSweep following the AddedEvent will only be available for the **last 5000 sweeps**, anything eariler will be lost and not printed.
3) It is OK to AddEvent after the MainProtocol has stopped as long as another MainProtocol has not been started.
4) If you do several AddedEvents, only the PxSweep following the first AddedEvent will be printed to the Expimental Log.

10.8 Printing SealTest protocol values

The Experimental Log will also automatically record important values such as those obtained from the patch-clamp SealTest protocol (see Chapter 5). These values include:

1) with the electrode in the bath
    a) the resistance of the pipette (Rpipette), in Mohms
2) after the seal has formed
a) the seal resistance (Rseal), in Gohms
3) After going whole cell
    a) the patch pipette series resistance (Rs), in Mohms
    b) the cell input resistance (Rm), in Mohms
    c) the holding current (Idc), in pA's

and also the pulse amplitude (PulseAmp), in mV's, used to generate the SealTest values (Fig. 10.8.1).

```
11:44:05.4       "" Go to SealTest
""               ""   ElectrodeInBath
""               ""     AD0 Rpipette = 4.1 Mohms  PulseAmp = 2.0 mV
""               ""   FormSeal
""               ""     AD0 Rseal = 1.5 Gohms  PulseAmp = 20.0 mV
""               ""   GoWholeCell
""               ""     AD0 Rs  =  12 Mohms  PulseAmp = -2.0 mV
""               ""         Rm  = 190 Mohms
""               ""         Idc = -28 pA
11:46:54.7 "" Return from SealTest
```

Fig. 10.8.1. The results of the SealTest protocol printed in the Experimental Log after exiting the SealTest.

10.9 Printing Load/Save Protocol Files, Save AmpFiles, Clear AnalysisGraphs and Spreadsheets

Finally, the Experimental Log records

1) when protocol files are loaded and saved
2) when AmpFiles are saved
3) when AnalysisGraphs and Spreadsheets are cleared

Fig. 10.9.1 shows these capabilities. WinLTP was started using the default.pro protocol. Then the Log shows AvgLoop.pro protocol file loaded. After the MainProtocol was run, the 0622A010.XLS AmpFile was saved, and the AnalysisGraphs and Spreadsheet were cleared. Finally the AvgLoop.pro protocol file was saved, and the WinLTP program exited.

```
TimeOfDay AnalysisTm Event
17:23:54.3        "" Start WinLTP
""                ""   Using default.pro
""                ""   Saving to 06220001.log
17:24:48.3        "" Loaded AvgLoop.pro
""
17:25:10.9       0.0 Start MainProtocol
17:25:11.4       0.5   Enter Loop0
17:25:11.4       0.5       06220314.P0, First Loop/AvgLoop sweep
17:25:26.4      15.5       06220317.AP0
17:25:46.4      35.5       06220321.AP0
17:26:06.4      55.5       06220325.AP0
17:26:11.4    1:00.5     Leave AvgLoop1
17:26:11.4    1:00.5   Leave Loop0
17:26:11.4    1:00.5 Stop MainProtocol
17:26:25.0        "" Saved 0622A010.XLS
17:26:25.1        "" Cleared AnalysisGraphs and Spreadsheet
17:26:45.6        "" Saved AvgLoop.pro
17:26:50.3        "" Exit WinLTP
```

Fig. 10.9.1. An Experimental Log showing loading of a protocol file, saving an AmpFile, clearing the AnalysisGraphs and Spreadsheet, and saving a protocol file.

10.10 Print Detection, Stimulation or All Protocol Values when their button is clicked

In WinLTP111 we added the capability to print Detection, Stimulation or All Protocol Values to the Experimental Log when the 'Detection', 'Stimulation' or 'All' button just below the Experimental Log (see Fig. 10.1.1) is clicked. This printing will occur while WinLTP is idle, or when the MainProtocol or ContinuousAcquisition Protocol is running. The first sweep after the Protocol values have been printed is also printed (see purple sections). In the three examples in this section, no Pulse or Train Sweep CheckBoxes have been checked, and the Detection, Stimulation and All Protocol Values are for the default.pro protocol.

Fig. 10.10.1 shows the Detection Protocol Values printed when the 'Detection' button is clicked. For each AD channel, the LowPass Filtering and Rs calculations are shown followed by waveform detection values, in this case for Baseline, Peak Amplitude and MaxSlope.

```
18:05:06.2        0.0 Start MainProtocol
18:05:06.7        0.5   Enter Loop0
18:05:06.7        0.5     09200227.P0, First Loop sweep
""
18:05:23.7         "" Detection Protocol Values
""                 ""     AD0
""                 ""       LowPass Filter - Not used
""                 ""       Rs not calculated
""                 ""       S0
""                 ""         Baseline: 8 to 2 ms before pulse
""                 ""         Peak: Auto 0 to 15 ms before pulse
""                 ""         MaxSlope: 2 ms, -2 to 2 ms after pulse
""                 ""       S1
""                 ""         Baseline: 8 to 2 ms before pulse
""                 ""         Peak: Auto 0 to 15 ms before pulse
""                 ""         MaxSlope: 2 ms, -2 to 2 ms after pulse
""                 ""     AD1
""                 ""       LowPass Filter - Not used
""                 ""       Rs not calculated
""                 ""       S0 - Not used
""                 ""       S1 - Not used
18:05:26.7       20.5     09200231.P0, First sweep after Protocol values printed
18:05:46.7       40.5   Leave Loop0
18:05:46.7       40.5 Stop MainProtocol
```

Fig. 10.10.1. Detection values (in blue) printed to the Experimental log when the 'Detection' button is clicked, followed by the first sweep after the Detection protocol values are printed (purple).

Fig. 10.10.2 shows the Stimulation Protocol Values printed when the 'Stimulation' button below the Experimental Log is clicked. The information in the MainProtocol tabsheet (including the Protocol Builder script lines in dark green), the Evoked Events tabsheet, and the P0sweep, P1sweep, T0sweep and T1sweep tabsheets is printed. The Plot/Save tabsheet values are not printed.

```
18:28:30.6        0.0 Start MainProtocol
18:28:31.1        0.5   Enter Loop0
18:28:31.1        0.5     09200243.P0, First Loop sweep
""
18:28:42.8         "" Stimulation Protocol Values
""                 ""   Continuous Acquisition - Used
""                 ""   MainProtocol tab
""                 ""     Start with MainProtocol
""                 ""       [x] Continuous Acquisition
""                 ""     Protocol Script Area
""                 ""       MainProtocol
""                 ""         [X] Loop     [  6]
""                 ""           [X] P0sweep [  5]s
""                 ""         EndLoop
""                 ""       EndProtocol
""                 ""     Enable Sweep Functions
""                 ""        Pulse Train
""                 ""         [ ]   [ ]  Stimulus Artifact Blanking
""                 ""         [ ]   [ ]  Low-Pass Filtering
""                 ""   Evoked Events tab
""                 ""     Single Pulse or Train Sweep
""                 ""       [x] P0sweep
""                 ""       [ ] P1sweep
""                 ""       [ ] T0sweep
""                 ""       [ ] T1sweep                       NumSwps  Add
""                 ""     Fast Repeat Pulse Sweeps (LTD)   Avg  toAvg  Delay
""                 ""       [ ] P0  [ 30]*[  2]s=1m        [ ]  [   ]   [ ]
""                 ""       [ ] P1  [ 30]*[  2]s=1m        [ ]  [   ]   [ ]
""                 ""     Fast Repeat Train Sweeps (Theta)          AddDelay
""                 ""       [ ] T0  [  4]*[  5]s=20s                    [ ]
""                 ""       [ ] T1  [  4]*[  5]s=20s                    [ ]
""                 ""   Sweep Stimulation Values
""                 ""     P0sweep tab
""                 ""       Sweep Duration = 100 msec
""                 ""         S0 tab              Delay0 Pulse1   Off2
""                 ""           Pulse Dur (ms)                0.1
""                 ""           Pulse Interval (ms)     20     50
""                 ""           Number Pulses                   1
""                 ""         S1  - NotUsed
""                 ""         IC0 - NotUsed
""                 ""         IC1 - NotUsed
""                 ""     P1sweep - NotUsed
""                 ""     T0sweep - NotUsed
""                 ""     T1sweep - NotUsed
18:28:46.1       15.5     09200246.P0, First sweep after Protocol values printed
18:29:01.1       30.5   Leave Loop0
18:29:01.1       30.5 Stop MainProtocol
```

Fig. 10.10.2. Stimulation values (in green) printed to the Experimental log when the ‘Stimulation’ button is clicked, followed by the first sweep after the Stimulation protocol values are printed (purple). The Protocol Builder script lines are shown in dark green.

Fig. 10.10.3 shows All Protocol Values printed when the ‘All’ button below the Experimental Log is clicked. The information includes values in the Edit Protocol dialog box (except the Resources tabsheet) (first yellow section), the Stimulation Protocol values (including values in the Plot/Save tabsheet) (green section), the Experimental Log values (second yellow section), the Detection values (blue section), and all other appropriate Dialog Box values (third yellow section).

```
18:40:57.2       0.0 Start MainProtocol
18:40:57.7       0.5   Enter Loop0
18:40:57.7       0.5     09200250.P0, First Loop sweep
""
18:41:10.3        "" All Protocol Values
""                ""   Edit Protocol dialog box
""                ""     Acquisition/Stimulation Paramters tab
""                ""       Analog Input Channels - Acquisition Values
""                ""                  DataType       Gain        Units/V              mV/Unit
""                ""         AD0  Used   mV          1000              1mV/V              1000mV/mV
""                ""         AD1  Used   mV          1000              1mV/V              1000mV/mV
""                ""       Acquisition Sample Intervals
""                ""         PulseSweeps:             50 us
""                ""         TrainSweeps:             50 us
""                ""         Continuous Acquisition:  50 us
""                ""       Analog Output Values
""                ""           DataType      Gain        Units/V
""                ""         IC0 mV            50            20mV/V
""                ""         IC1 mV            50            20mV/V
""                ""   Continuous Acquisition - Used
""                ""   MainProtocol tab
""                ""     Start with MainProtocol
""                ""       [x] Continuous Acquisition
""                ""     Protocol Script Area
""                ""       MainProtocol
""                ""         [X] Loop     [  6]
""                ""           [X] P0sweep [  5]s
""                ""         EndLoop
""                ""       EndProtocol
""                ""     Enable Sweep Functions
""                ""        Pulse Train
""                ""         [ ]   [ ]  Stimulus Artifact Blanking
""                ""         [ ]   [ ]  Low-Pass Filtering
""                ""   Evoked Events tab
""                ""     Single Pulse or Train Sweep
""                ""       [x] P0sweep
""                ""       [ ] P1sweep
""                ""       [ ] T0sweep
""                ""       [ ] T1sweep                          NumSwps  Add
""                ""     Fast Repeat Pulse Sweeps (LTD)   Avg  toAvg  Delay
""                ""       [ ] P0  [ 30]*[  2]s=1m        [ ]  [   ]   [ ]
""                ""       [ ] P1  [ 30]*[  2]s=1m        [ ]  [   ]   [ ]
""                ""     Fast Repeat Train Sweeps (Theta)          AddDelay
""                ""       [ ] T0  [  4]*[  5]s=20s                    [ ]
""                ""       [ ] T1  [  4]*[  5]s=20s                    [ ]
""                ""   Plot/Save tab
""                ""     AD Channels to Plot and Save
""                ""               Plot        SaveToDisk
""                ""            Cont  Stim     Cont  Stim
""                ""           Acquis Sweep   Acquis Sweep
""                ""       AD0   [x]  [x]       [x]  [x]
""                ""       AD1   [x]  [x]       [x]  [x]
""                ""     Save Sweeps to Disk
""                ""      Pulse Train
""                ""       [x]   [x]  Raw Sweeps
""                ""       [ ]        Averaged Sweeps
""                ""       [ ]   [ ]  Stimulus Blanked Sweeps
""                ""       [ ]   [ ]  Low-Pass Filtered Sweeps
""                ""   Sweep Stimulation Values
""                ""     P0sweep tab
""                ""       Sweep Duration = 100 msec
""                ""         S0 tab                Delay0 Pulse1   Off2
""                ""           Pulse Dur (ms)                0.1
""                ""           Pulse Interval (ms)     20     50
""                ""           Number Pulses                   1
""                ""         S1  - NotUsed
""                ""         IC0 - NotUsed
""                ""         IC1 - NotUsed
""                ""     P1sweep - NotUsed
""                ""     T0sweep - NotUsed
""                ""     T1sweep - NotUsed
```

Fig. 10.10.3 (top). All Protocol values printed when the ‘All’ button is clicked. Continued on next pages.

```
""                  ""   Experimental Log Values
""                  ""     Print to Log
""                  ""       Sweeps/Files
""                  ""         [x] T0   [x] P0   [x] AP0
""                  ""         [x] T1   [x] P1   [x] AP1
""                  ""       Protocol Flow, Enter/Leave
""                  ""         [x] Loop
""                  ""         [x] AvgLoop
""                  ""         [x] Run
""                  ""         [x] Run     of Run/ElseRun
""                  ""         [x] ElseRun of Run/ElseRun
""                  ""   Detection Values
""                  ""     AD0
""                  ""       LowPass Filter - Not used
""                  ""       Rs not calculated
""                  ""       S0
""                  ""         Baseline: 8 to 2 ms before pulse
""                  ""         Peak: Auto 0 to 15 ms before pulse
""                  ""         MaxSlope: 2 ms, -2 to 2 ms after pulse
""                  ""       S1 - Not used
""                  ""     AD1
""                  ""       LowPass Filter - Not used
""                  ""       Rs not calculated
""                  ""       S0 - Not used
""                  ""       S1 - Not used
""                  ""   Dialog Box Values
""                  ""     SweepFile -> Set ADsweep File Types  dialog box
""                  ""       PulseSweep File Type
""                  ""         (*) WinLTP's ASCII file format (data columns only)
""                  ""       TrainSweep - Not used
""                  ""     AmpFile -> Analyses To Do  dialog box
""                  ""              AD0                AD1
""                  ""       MainPg AnalysisPg  MainPg AnalysisPg
""                  ""        S0+S1  S0+S1       S0+S1  S0+S1
""                  ""         [x]    [ ]         [ ]    [ ]  DC Baseline
""                  ""         [x]    [ ]         [ ]    [ ]  Peak Amplitude
""                  ""         [ ]    [ ]         [ ]    [ ]  Latency
""                  ""         [ ]    [ ]         [ ]    [ ]  Area
""                  ""         [ ]    [ ]         [ ]    [ ]  Duration
""                  ""         [ ]    [ ]         [ ]    [ ]  Rise Time
""                  ""         [ ]    [ ]         [ ]    [ ]  Decay Time
""                  ""         [ ]    [ ]         [ ]    [ ]  Coastline
""                  ""         [ ]    [ ]         [ ]    [ ]  PopSpike Amplitude
""                  ""         [ ]    [ ]         [ ]    [ ]  PopSpike Latency
""                  ""         [x]    [ ]         [ ]    [ ]  Slope
""                  ""         [ ]    [ ]         [ ]    [ ]  Average Amplitude
""                  ""         [ ]    [ ]         [ ]    [ ]  Rs
""                  ""         [ ]    [ ]         [ ]    [ ]  Rm
""                  ""     AmpFile -> Slope Calculation Method  dialog box
""                  ""         AD0    AD1
""                  ""         (*)    (*)  Maximum Slope
""                  ""         ( )    ( )  Begin -> End Times
""                  ""         ( )    ( )  Low% -> High% of Peak Amplitude
""                  ""     AmpFile -> PopSpike Calculation Method  dialog box
""                  ""         AD0    AD1
""                  ""         (*)    (*)  Area
""                  ""         ( )    ( )  Amplitude
""                  ""     AmpFile -> Series and Input Resistance  dialog box
""                  ""       Measure Rs and Rm from Normal or Unfiltered Trace
""                  ""         AD0    AD1
""                  ""         ( )    ( )  Normal trace (Raw, Averaged, Blanked and/or Filtered)
""                  ""         (*)    (*)  Unfiltered trace (default, Raw and/or Averaged)
""                  ""       Rs Calculation Method
""                  ""         AD0    AD1
""                  ""         (*)    (*)  Peak (default fit)
""                  ""         ( )    ( )  Single Exponential Fit
""                  ""         ( )    ( )  Double Exponential Fit
""                  ""       Extrapolation
""                  ""         AD0    AD1
""                  ""         (*)    (*)  Extrapolate back to TimeZero (start of Rs/Rm pulse, default)
""                  ""         ( )    ( )  Extrapolate between TimeZero and Rs Peak
""                  ""         ( )    ( )  No extrapolation, use Rs Peak
""                  ""         ( )    ( )  In Mohms (using external amplifier's Set/Gated stimulation)
""                  ""                        AD0  -2 mV
```

Fig. 10.10.3 (middle). All Protocol values printed when the ‘All’ button is clicked. Continued on preceding and next pages.

```
""                ""       Rs and Rm Measurement Results
""                ""         AD0    AD1
""                ""         ( )    ( )  In pA or mV (using Rs and Rm amplitudes; Calc R using V/I)
""                ""         (*)    (*)  In Mohms (using ICx AnalogOut stimulation, default)
""                ""         ( )    ( )  In Mohms (using external amplifier's Set/Gated stimulation)
""                ""                        AD0  -2 mV
""                ""                        AD1  -2 mV
""                ""     AmpFile -> Train Analysis dialog box
""                ""       Analysis of Pulses in Trains
""                ""         P0sweep  P1sweep    T0sweep  T1sweep
""                ""          S0 S1    S0 S1      S0 S1    S0 S1
""                ""         (*) (*)  (*) (*)    (*) (*)  (*) (*)  Analyze first pulse in Train
""                ""         ( ) ( )  ( ) ( )    ( ) ( )  ( ) ( )  Analyze first/last pulse in Train
""                ""         ( ) ( )  ( ) ( )    ( ) ( )  ( ) ( )  Analyze every pulse in Train
""                ""         ( ) ( )  ( ) ( )    ( ) ( )  ( ) ( )  Analyze every pulse in Train using …
""                ""     AmpFile -> Spreadsheet/AmpFile Options dialog box
""                ""       AutoSave AmpFile when Analyis Cleared
""                ""         (*)  On
""                ""       AutoSave AmpFile as an
""                ""         [ ]  ASCII text file (*.amp)
""                ""         [x]  Excel file (*.xls)
""                ""       AutoSave AmpFile as an
""                ""         ( )  Blanks
""                ""         (*)  Empty double quotes ""
""                ""     Help -> About dialog box
""                ""       WinLTP Version: 1.10
""                ""       Processor:            Intel(R) Core(TM)2 CPU          6600  @ 2.40GHz
""                ""       Processor speed:      2.397601872 GHz
""                ""       Number of processors: 2
""                ""       Operating system: Windows XP, ServicePack 3.0
""                ""       Total physical memory:     3325 MB
""                ""       Available physical memory: 2244 MB
""                ""       Percent of memory in use:  32%
""                ""       M- or X-Series Board: PCIE-6321
""                ""       Serial Number:        21906943
""                ""       NI-DAQmx Version:     9.1
""                ""       Present protocol file loaded: Default_ExptLog.pro
""                ""       Data Root Folder:       C:\WinLTP\
""                ""       Data Read/Write Folder: C:\WinLTP\100920\
""                ""       A valid License Key (Serial# 921) has been detected.
""                ""       One of 5 copies licensed to Dr. Stephen Fitzjohn, Univ of Bristol.
""                ""       You are permanently running in the Advanced Mode.
18:41:12.7      15.5     09200253.P0, First sweep after Protocol values printed
18:41:27.7      30.5   Leave Loop0
18:41:27.7      30.5 Stop MainProtocol
```

Fig. 10.10.3 (this page and preceding two pages). All Protocol values printed when the ‘All’ button is clicked. Sections include the Edit Protocol dialog box (first yellow) section, the Stimulation Protocol values (green) section (including the Protocol Builder script lines shown in dark green), the Experimental Log values (second yellow) section, the Detection values (blue) section, and all other appropriate Dialog Box values (third yellow) section. This whole ‘All’ Protocol Values is also printed to the Experimental Log when WinLTP is exited.

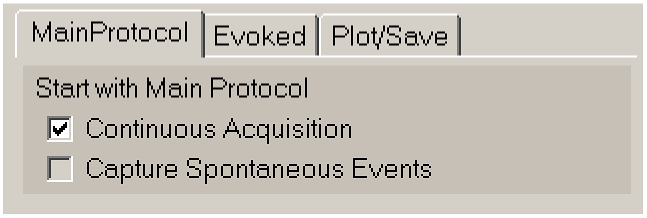
10.11 Print All Protocol Values when WinLTP exits

Furthermore, when WinLTP is exited, All Protocol Values will be printed to the end of the Experimental Log. There is no switch to turn this off if WinLTP is in the Advanced Mode.

CHAPTER 11 – Continuous Acqusition

The second task is a tape recorder that saves continuously acquired data to an Axon Binary File (*.abf). This file can then be used for off-line analysis of spontaneous events with other programs (Fig. 11.2).

If you want to use Continuous Acquisition when running your MainProtocol, make sure the Continuous Acquisition check box is checked as in Fig. 11.1.


MainProtocol | Evoked | Plot/Save

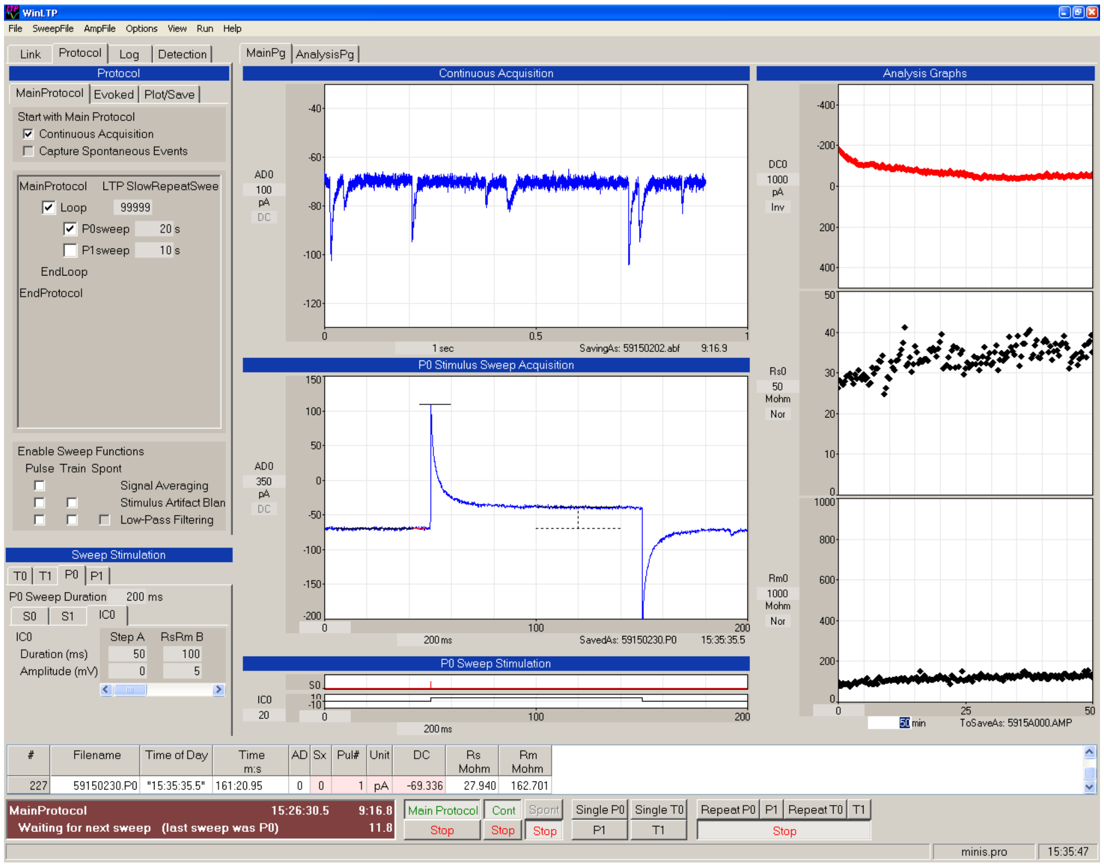
Start with Main Protocol

☑ Continuous Acquisition

☐ Capture Spontaneous Events

Fig. 11.1. Start with the Main Protocol Panel. To start Continuous Acquisition along with the Main Protocol as a simultaneous task, make sure the Continuous Acquisition check box is checked. The simultaneous Capture Spontaneous Events task is currently not available for use.

Alternatively, Continuous Acquisition can be run without running the Main Protocol by clicking on the "Cont" Run Button.


WinLTP
File SweepFile AmpFile Options View Run Help
Link Protocol Log Detection
MainPg AnalysisPg
Continuous Acquisition
P0 Stimulus Sweep Acquisition
P0 Sweep Stimulation
Analysis Graphs
Sweep Stimulation
T0 T1 P0 P1
S0 S1 IC0
Main Protocol Cont Spont Single P0 Single T0 Repeat P0 P1 Repeat T0 T1
Stop Stop Stop P1 T1 Stop

Protocol

MainProtocol | Evoked | Plot/Save

Start with Main Protocol

☑ Continuous Acquisition

☐ Capture Spontaneous Events

MainProtocol LTP SlowRepeatSwee

☑ Loop 99999

☑ P0sweep 20 s

☐ P1sweep 10 s

EndLoop

EndProtocol

Enable Sweep Functions

Pulse Train Spont

☐ Signal Averaging

☐ ☐ Stimulus Artifact Blan

☐ ☐ ☐ Low-Pass Filtering

P0 Sweep Duration 200 ms

IC0	Step A	RsRm B
Duration (ms)	50	100
Amplitude (mV)	0	5

AD0 100 pA DC; 1 sec; SavingAs: 59150202.abf 9:16.9

AD0 350 pA DC; 200 ms; SavedAs: 59150230.P0 15:35:35.5

IC0 20; 200 ms

DC0 1000 pA Inv; Rs0 50 Mohm Nor; Rm0 1000 Mohm Nor; 50 min ToSaveAs: 5915A000.AMP

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	DC	Rs Mohm	Rm Mohm
227	59150230.P0	"15:35:35.5"	161:20.95	0	0	1	pA	-69.336	27.940	162.701

MainProtocol 15:26:30.5 9:16.8
Waiting for next sweep (last sweep was P0) 11.8

minis.pro 15:35:47

Fig 11.2. Two tasks operating simultaneously – Continuous Acquisition (up to 40 KHz/ch, top middle panel), and a Stimulation/Acquisition Sweep to measure patch electrode series resistance (Rs) and input resistance (Rm) (other graph panels).

In addition to checking the Continuous Acquisition check box in the MainProtocol tabsheet, you may also need to check the 'Plot' to and 'Save' to check boxes in the Plot/Save tabsheet (Fig. 3.1.2.2).

CHAPTER 12 – Converting WinLTP ADsweep Files

12.1 Convert WinLTP ASCII ADsweep Files to Axon Binary Files (*.abf)

In WinLTP 1.10, the ability convert WinLTP ADsweep files to Molecular Devices Axon Binary File (ABF) files was added. The Axon Binary File format originated and is used by Molecular Device's pClamp programs, and by many analysis programs including Molecular Devices ClampFit, Synaptosoft's MiniAnalysis, Axograph Scientific's AxographX, Bill Heitler's DataView and Christoph Schmidt-Hieber's StimFit32 (which can also directly analyze WinLTP's ASCII ADsweep files).

Note however, that WinLTP's ability to convert WinLTP ADsweep files to Axon Binary Files is only available in WinLTP's Advanced Version or during the DemoTrial period.

WinLTP can convert many WinLTP ADsweep files to:

1) Many single-sweep gap-free *.ABF files
2) Many single-sweep episodic *.ABF files
3) One multi-sweep episodic *.ABF file

Both the gap-free and episodic ABF files are in the binary **integer** file format, not floating point.

The version of these ABF files is the 'lowest common denominator', **Version 1** (used by Molecular Device's pClamp 8 and 9) so as to be accessed by the largest number of analysis programs. WinLTP does not convert to ABF Version 2 files (used by pClamp 10). However, pClamp 10 can convert Version 1 ABF files to Version 2.

Also, it is important to realize that the **stimulation information in the header of WinLTP ADsweep files is not transferred into the ABF files**. In other words, the converted ABF files contain no stimulation information. This is because of substantial differences in the stimulation capabilities of WinLTP and pClamp. However, this should usually not be too much of a problem when cursors are used to select sections of the sweep for analysis.

Conversion to ABF files can either be performed in the Online/Acquisition program either in the DemoTrial period or with a Temporary or Permanent Licence Key File to change it to the Advanced Mode, or in the Reanalysis program with a Permanent Licence Key File.

To convert WinLTP ADsweep files to one or more ABF files you first have to choose 1) Many single-sweep gap-free *.ABF files, 2) Many single-sweep episodic *.ABF files, or 3) One mlti-sweep episodic *.ABF file. To do so click on the menu item (Fig 12.1.1A):

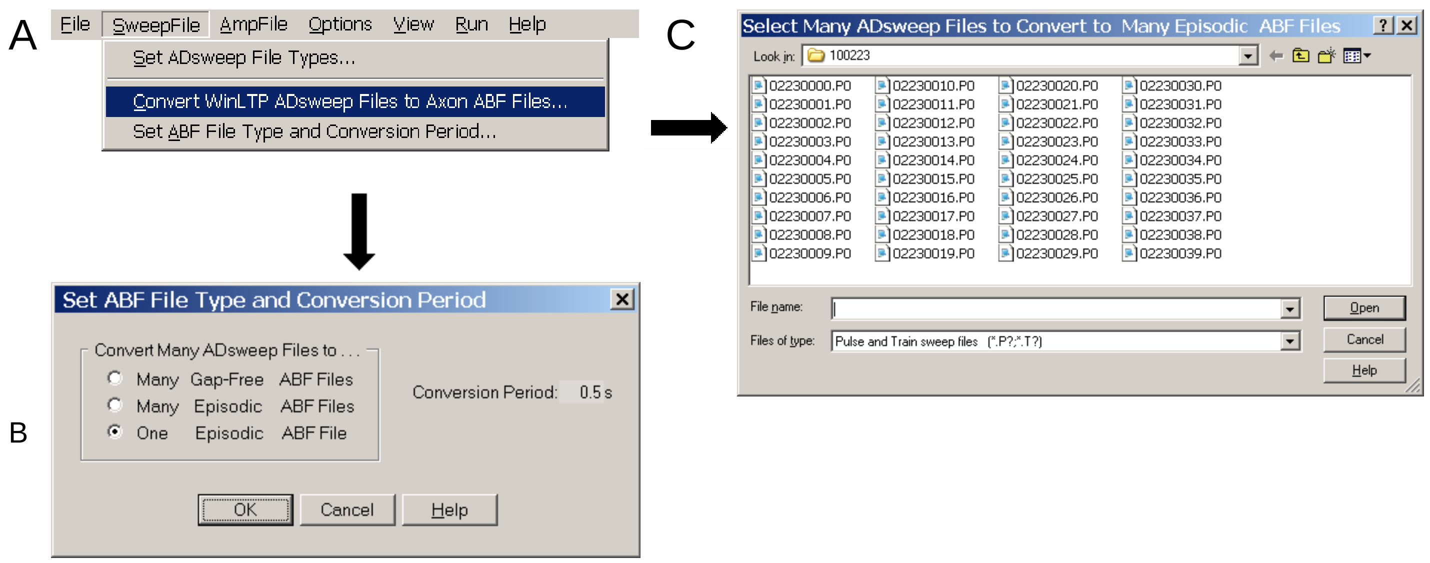
SweepFile -> Set ABF File Type and Conversion Period

to call up the Set ABF File Type and Conversion Period dialog box (Fig. 12.1.1B). You can covert many WinLTP ADsweep files 1) to many gap-free ABF files, with one single-sweep gap-free ABF file or each single-sweep ADSweep file, 2) to many episodic ABF files, with one single-sweep episodic ABF file or each single-sweep ADSweep file, or 3) one multi-sweep episodic ABF file. You can also set the conversion period from 0.0 sec on up to set sufficient time between conversions to allow looking at the ADsweep files being converted.

Once the choice of which ABF file to convert to is made, next click on the menu item (Fig 12.1.1A):

SweepFile -> Set ABF File Type and Conversion Period

to call up the Select Many ADsweep Files to Convert to One/Many Episodic/Gap-Free ABF files dialog box (Fig. 12.1.1C), and just select the files to convert and click the Open button to begin conversion.


A
File
SweepFile
AmpFile
Options
View
Run
Help
Set ADsweep File Types...
Convert WinLTP ADsweep Files to Axon ABF Files...
Set ABF File Type and Conversion Period...
B
Set ABF File Type and Conversion Period
Convert Many ADsweep Files to . . .
Many Gap-Free ABF Files
Many Episodic ABF Files
One Episodic ABF File
OK
Cancel
Help
C
Select Many ADsweep Files to Convert to Many Episodic ABF Files
Open
Cancel
Help

Convert Many ADsweep Files to . . . [✓ One Episodic ABF File]

Conversion Period: 0.5 s

Look in: 100223

02230000.P0	02230010.P0	02230020.P0	02230030.P0
02230001.P0	02230011.P0	02230021.P0	02230031.P0
02230002.P0	02230012.P0	02230022.P0	02230032.P0
02230003.P0	02230013.P0	02230023.P0	02230033.P0
02230004.P0	02230014.P0	02230024.P0	02230034.P0
02230005.P0	02230015.P0	02230025.P0	02230035.P0
02230006.P0	02230016.P0	02230026.P0	02230036.P0
02230007.P0	02230017.P0	02230027.P0	02230037.P0
02230008.P0	02230018.P0	02230028.P0	02230038.P0
02230009.P0	02230019.P0	02230029.P0	02230039.P0

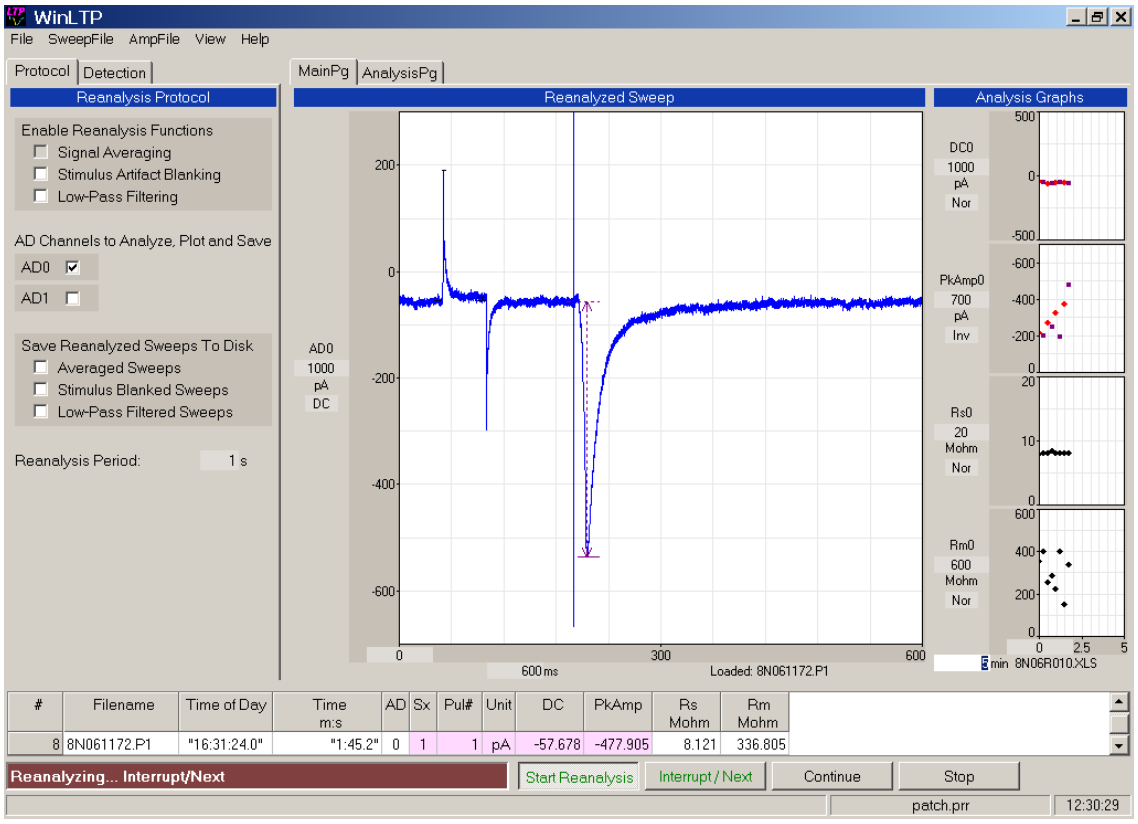
File name:

Files of type: Pulse and Train sweep files (*.P?;*.T?)

Fig. 12.1.1. How to convert WinLTP ADsweep files to Axon Binary ABF files. A) The SweepFile menu items to Convert or to Set ABF File Type. B) The Set ABF File Type and Conversion Period dialog box. C) The Select Many ADsweep Files to Conver to One or Many Episodic or Gap-Free ABF files dialog box.

Fig. 12.1.2 shows the conversion of a single-sweep ADsweep file (viewed in WinLTP) to a single-sweep episodic ABF file (viewed in Molecular Devices Clampfit, the pClamp analysis program).

WinLTP here has converted a single-sweep ASCII ADsweep file …



Enable Reanalysis Functions
- [ ] Signal Averaging
- [ ] Stimulus Artifact Blanking
- [ ] Low-Pass Filtering

AD Channels to Analyze, Plot and Save
- AD0 [✓]
- AD1 [ ]

Save Reanalyzed Sweeps To Disk
- [ ] Averaged Sweeps
- [ ] Stimulus Blanked Sweeps
- [ ] Low-Pass Filtered Sweeps

Reanalysis Period: 1 s

Loaded: 8N061172.P1

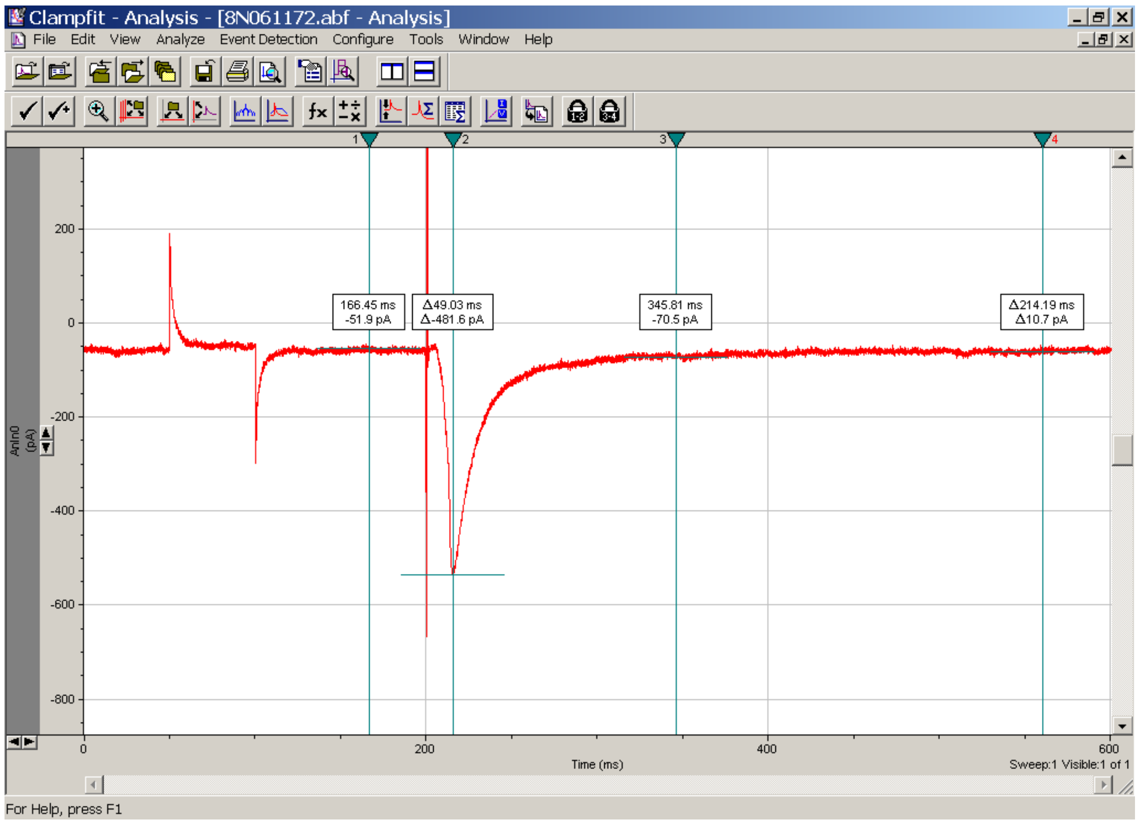
8N06R010.XLS

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	DC	PkAmp	Rs Mohm	Rm Mohm
8	8N061172.P1	"16:31:24.0"	"1:45.2"	0	1	1	pA	-57.678	-477.905	8.121	336.805

Reanalyzing... Interrupt/Next

patch.prr 12:30:29

… into a single-sweep, episodic Axon Binary File (loaded into Clampfit)


Clampfit - Analysis - [8N061172.abf - Analysis]
File Edit View Analyze Event Detection Configure Tools Window Help
AnIn0 (pA)
Time (ms)
Sweep:1 Visible:1 of 1

166.45 ms -51.9 pA

Δ49.03 ms Δ-481.6 pA

345.81 ms -70.5 pA

Δ214.19 ms Δ10.7 pA

For Help, press F1

Fig. 12.1.2. WinLTP converts a single-sweep ASCII ADsweep file (8N061172.P1) into a single-sweep, episodic Axon Binary File (8N061172.abf, loaded into Molecular Device's Clampfit).

CHAPTER 13 – Additional Information about WinLTP

13.1 Location of Analysis Graphs on the Main Page

The Analysis Graphs on the MainPg can either be located at the top (Fig. 3.1.1) or on the right (Fig. 11.2) of the MainPg. To change the location of the Analysis Graphs on the MainPg, call up the Location of Analysis Graphs on MainPg (Fig. 13.1.1) by using the View menu command (Fig. 3.2.5) :

View -> Location of Analysis Graphs on MainPg…

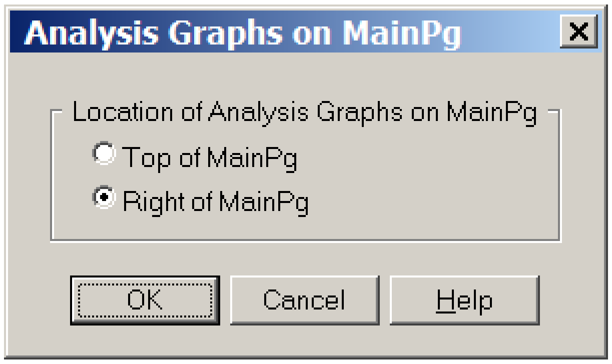

Analysis Graphs on MainPg
Location of Analysis Graphs on MainPg
Top of MainPg
Right of MainPg
OK
Cancel
Help

Fig. 13.1.1. Location of Analysis Graphs on MainPg dialog box.

13.2 Set the Number of Analyis Graph Columns on the Analysis Page

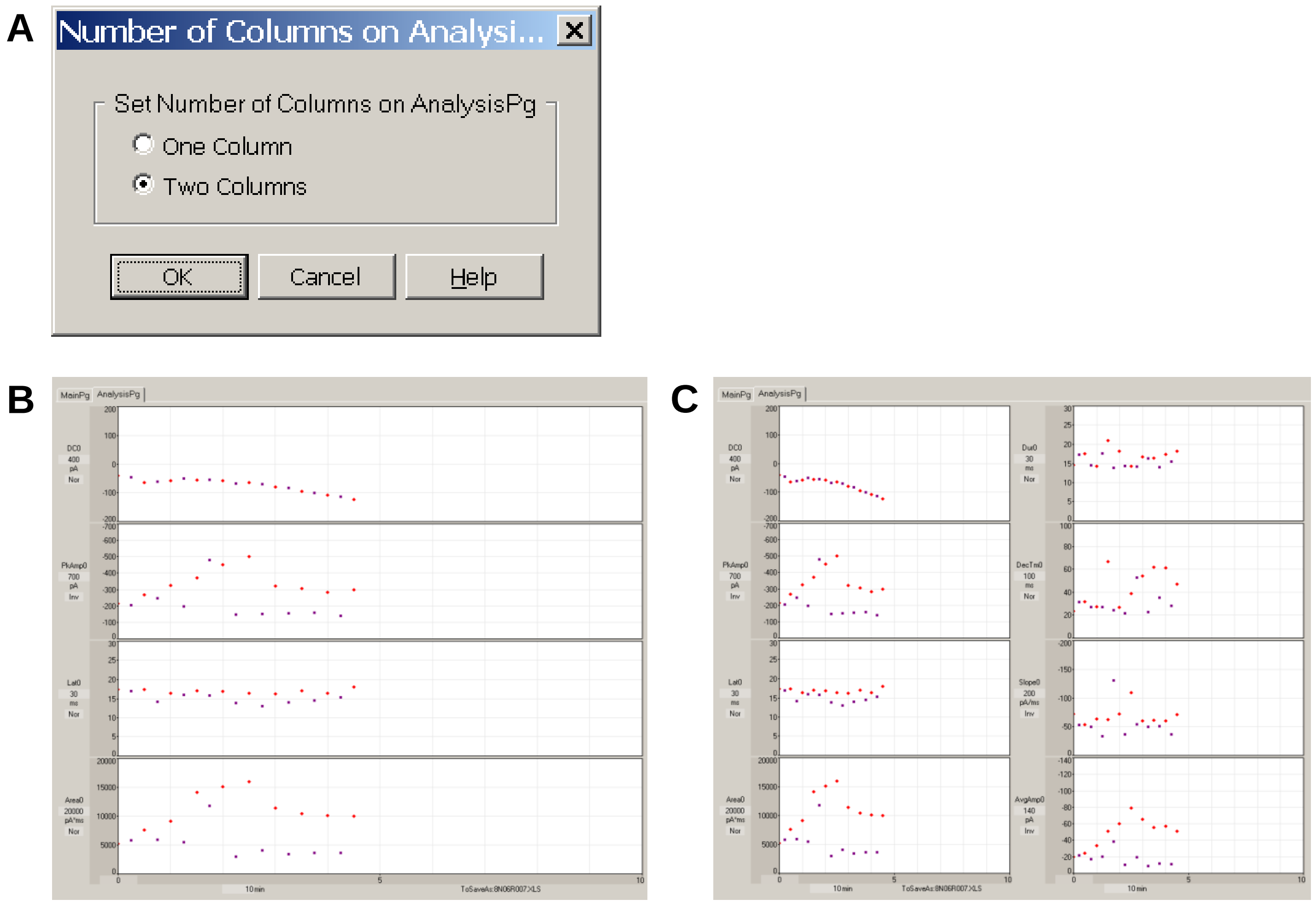
On the AnalysisPage, there can either be up to 8 Analysis Graphs in two columns (Fig. 13.2.1.C) or up to 4 Analysis Graphs in one column (Fig. (13.2.1B). Therefore, aAdding the up to 4 Analysis Graphs on the MainPg with the up to 8 Analysis Graphs on the Analysis Page gives a total of up to 12 Analysis Graphs that can be viewed during online acquisition and reanalysis.

The number of columns on the Analysis Page can be chosen using the menu items

View -> Number of Columns on AnalysisPg…

to call up the Number of Columns on AnalysisPg dialog box (Fig. 13.2.1A).

Using this dialog box you can set set either B) one column with up to 4 Analysis Graphs on the Analysis Page, or C) two columns with up to 8 Analysis Graphs on the Analysis Page. However, if you are using only one column on the Analysis Page and you then choose 5 or more analyses in the Analysis To Do dialog box, the num of columns will automatically increase to two.


Number of Columns on Analysi...

Set Number of Columns on AnalysisPg

One Column [✓ Two Columns]

OK | Cancel | Help

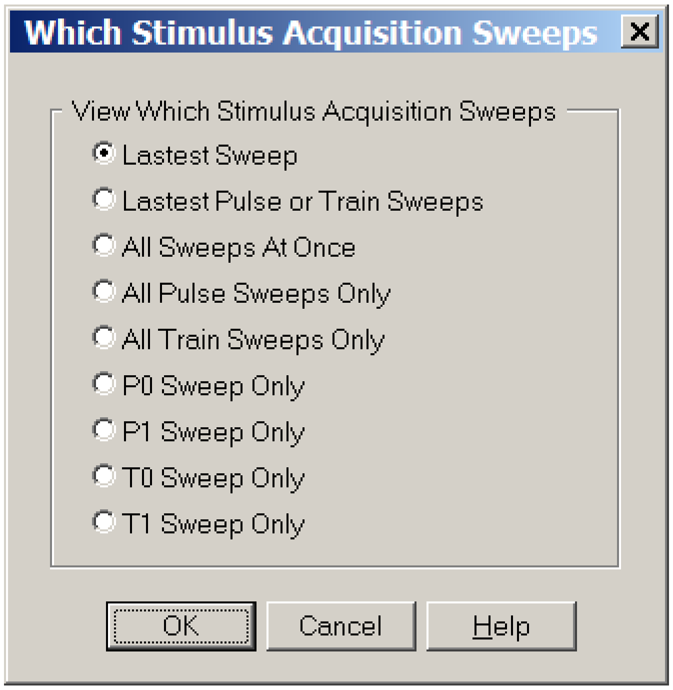
Fig. 13.2.1. A) The Number of Columns onAnalysisPg dialog box. This allows setting either B) one column with up to 4 Analysis Graphs on the Analysis Page, or C) two columns with up to 8 Analysis Graphs on the Analysis Page.

13.3 Set Which Pulse and Train Sweeps Appear in the Main Page

In the Stimulus Acquisition Sweep area of the MainPg, usually only the latest P0, P1, T0 or T1 sweep appears. However, several other acquisition sweep combinations that can be plotted. To change the acquisition sweeps that appear on the MainPg area use the View menu command (Fig. 3.2.5):

View -> Which Stimulus Acquuisition Sweeps…

to call up the Which Stimulus Acquisition Sweeps should appear on MainPg dialog box(Fig. 13.3.1).



Which Stimulus Acquisition Sweeps

View Which Stimulus Acquisition Sweeps

[✓ Lastest Sweep]
Lastest Pulse or Train Sweeps
All Sweeps At Once
All Pulse Sweeps Only
All Train Sweeps Only
P0 Sweep Only
P1 Sweep Only
T0 Sweep Only
T1 Sweep Only

Fig. 13.3.1. The Which Stimulus Acquisition Sweeps dialog box determines which combination of stimulation acquisition sweeps appear in the MainPg area.

For example, if **All Sweeps At Once** is chosen the Stimulus Acquisition Sweep graphs can appear as shown in Fig. 13.3.2.

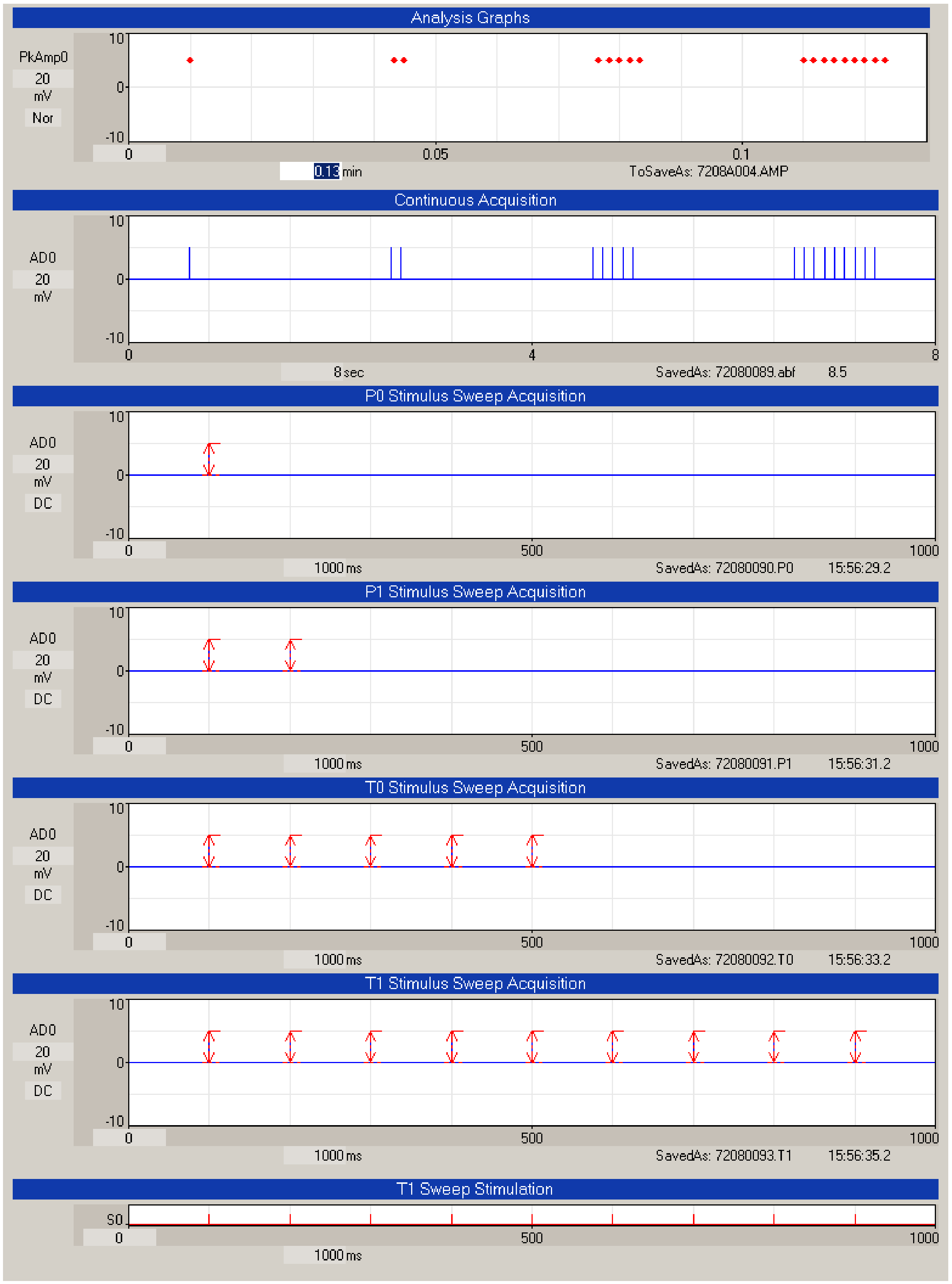


Fig. 13.3.2. The MainPg as it appears when the **All Sweeps At Once** is chosen from the **Which Stimulus Acquisition Sweeps** dialog box. The top trace is a PkAmp0 Analysis Graph, followed by the Continuous Acquisition trace, then the P0, P1, T0 and T1 Stimulus Sweep Acquisition Graphs, and finally, at the bottom, the T1 Sweep Stimulation graph. Only channel AD0 has been plotted.

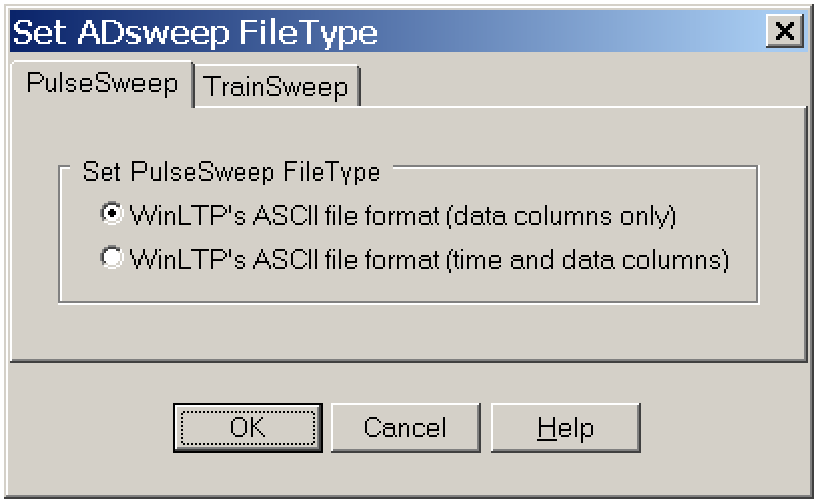
13.4 Set the Pulse or Train ADsweep FileType

The Pulse or Train ADsweep FileType can be changed by using the menu command (Fig. 3.2.2):

SweepFile -> Set ADsweep File Types…

to call up the Set ADsweep FileType dialog box (Fig. 13.4.1).

The Set ADsweep FileType dialog box can set either the Pulse ADsweep or Train ADsweep (not shown) FileType to either have an ASCII file format with data columns only (the default), or to also include an initial time column (in msec).



Set PulseSweep FileType

[✓ WinLTP's ASCII file format (data columns only)]

WinLTP's ASCII file format (time and data columns)

Fig. 13.4.1. The Set ADsweep FileType dialog box.

13.5 No Time Between Sweeps

One capability of WinLTP that has not been discussed yet is the ability to produce sweep stimulation and acquisition with no time between sweeps. This is valuable in patch-clamp 1 or 2 Hz LTD stimulation where the this enables the full 500 or 1000 msec sweep duration to be captured so as to record the whole EPSC/IPSC event.

Fig. 13.5.1 shows an example of this where 4 P0sweeps of 1000 msec duration are repeated every sec (and therefore there is no time between P0sweeps). There is 1 S0 pulse per sweep (so 1 Hz stimulation), and 20 S1 pulses per sweep (so 20 Hz stimulation). Note that there is no delay between the sweeps as shown by the 20 Hz S1 stimulation (AD1 trace in the Continuous Acquisition panel).

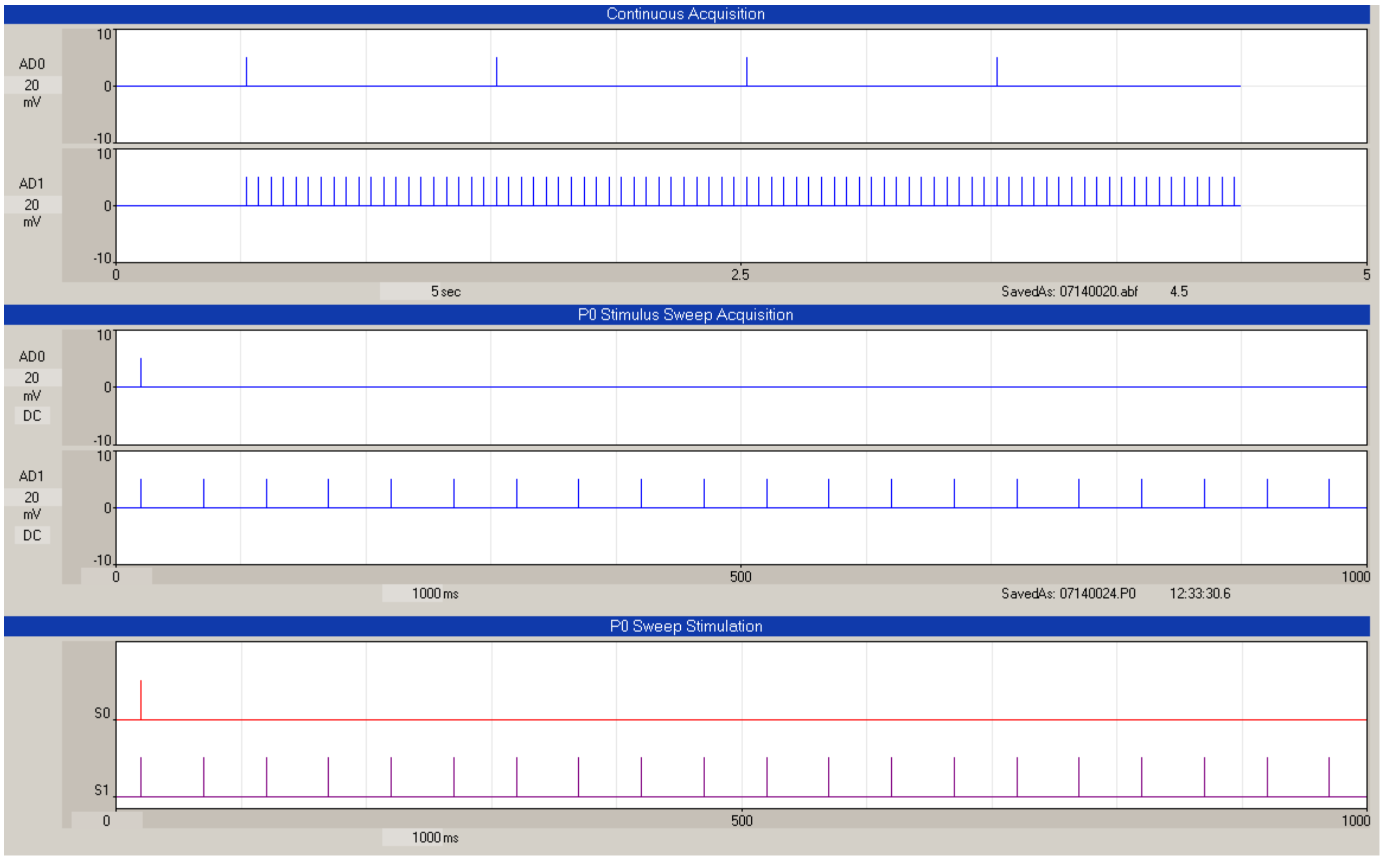


Fig. 13.5.1. An example of no time between sweeps. Note that there is no delay in the 20 Hz S1 stimulation between P0sweeps (AD1 trace in Continuous Acquisition panel).

13.6 Information in the Help About Dialog Box

The About box (Fig. 13.6.1) contains several interesting pieces of program and computer information. It is called by using the Help menu command (Fig. 3.2.7):

Help -> About…

The pertinent information for diagnosing problems with WinLTP includes:

1) Microprocessor
   a) Microprocessor type according to Intel, including the speed (here an Intel Core 2 CPU 6600 @ 2.40GHz)
   b) Microprocessor speed as measured by WinLTP (here 2.397554208 GHz)
   c) Number of processors (4 for a quad-core, 2 for a dual-core, 2 for a single-core Pentium 4 with hyper-threading, and 1 for a single-core Pentium 4 without hyper-threading)
2) Operating system: Windows 2000, XP, Vista or Windows 7, and ServicePack number
3) Physical memory
   a) Total physical memory
   b) Available physical memory
   c) Percent of physical memory in use
4) Data acquisition board
   a) Type of data acquisition board (here a National Instruments X-Series PCIe-6321)
   b) Serial number of the data acquisition board

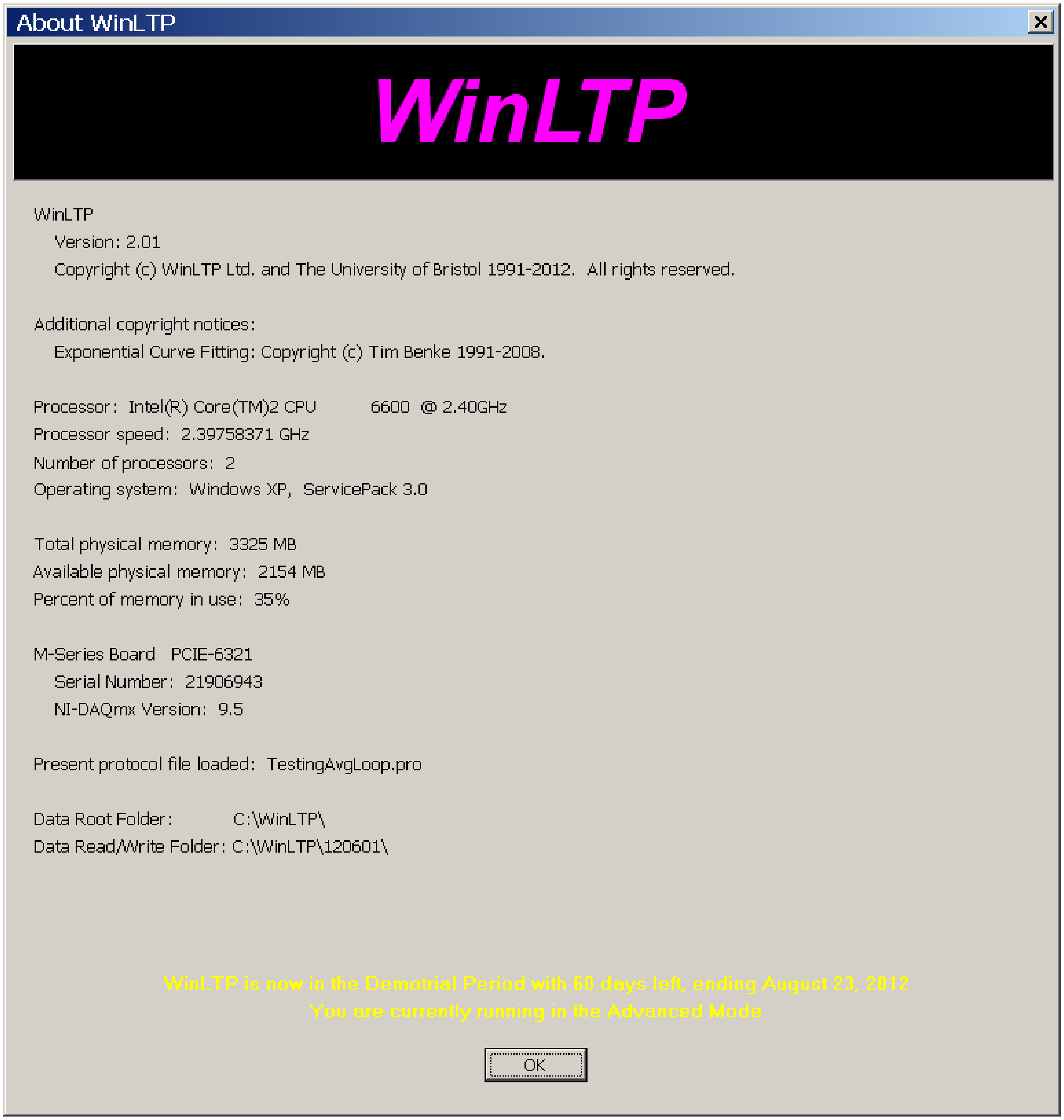
c) Version of the National Instruments Ni-DAQmx driver software (here version 9.1)

5) Name of the present protocol file loaded

6 Folders

a) Name of the data root folder

b) Name of the data read/write folder



About WinLTP

WinLTP

WinLTP
    Version: 2.01
    Copyright (c) WinLTP Ltd. and The University of Bristol 1991-2012. All rights reserved.

Additional copyright notices:
    Exponential Curve Fitting: Copyright (c) Tim Benke 1991-2008.

Processor: Intel(R) Core(TM)2 CPU   6600 @ 2.40GHz
Processor speed: 2.39758371 GHz
Number of processors: 2
Operating system: Windows XP, ServicePack 3.0

Total physical memory: 3325 MB
Available physical memory: 2154 MB
Percent of memory in use: 35%

M-Series Board   PCIE-6321
    Serial Number: 21906943
    NI-DAQmx Version: 9.5

Present protocol file loaded: TestingAvgLoop.pro

Data Root Folder: C:\WinLTP\
Data Read/Write Folder: C:\WinLTP\120601\

WinLTP is now in the Demotrial Period with 60 days left, ending August 23, 2012
You are currently running in the Advanced Mode

Fig. 13.6.1. The Help About Box showing processor information, the operating system, memory information, and what mode (Advanced or Basic) the program is running in including copy protection information.

CHAPTER 14 – Reanalyzing Data with WinLTP Reanalysis Program

Although this version of WinLTP does a reasonably good job of doing accurate on-line amplitude and slope analysis, experience in our group has indicated that subsequent off-line analysis can often improve the analysis accuracy. Doing Reanalysis with WinLTP is like reliving the experiment in 'fast forward' mode without any interruptions between ADsweeps.

14.1 Start WinLTP Reanalysis

To start the WinLTP Reanalysis program, click on the 'WinLTP Reanalysis' icon (Fig 14.1).

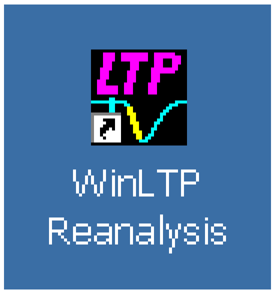


Fig 14.1.

When the WinLTP Reanalysis program starts up, the initial 'splash screen' comes up almost immediately indicating that the program in "Loading…", which takes at least 15 seconds. After this period the "Loading…" message goes away and the initial Reanalyzing Data dialog box is active (top of Fig. 14.2.1).

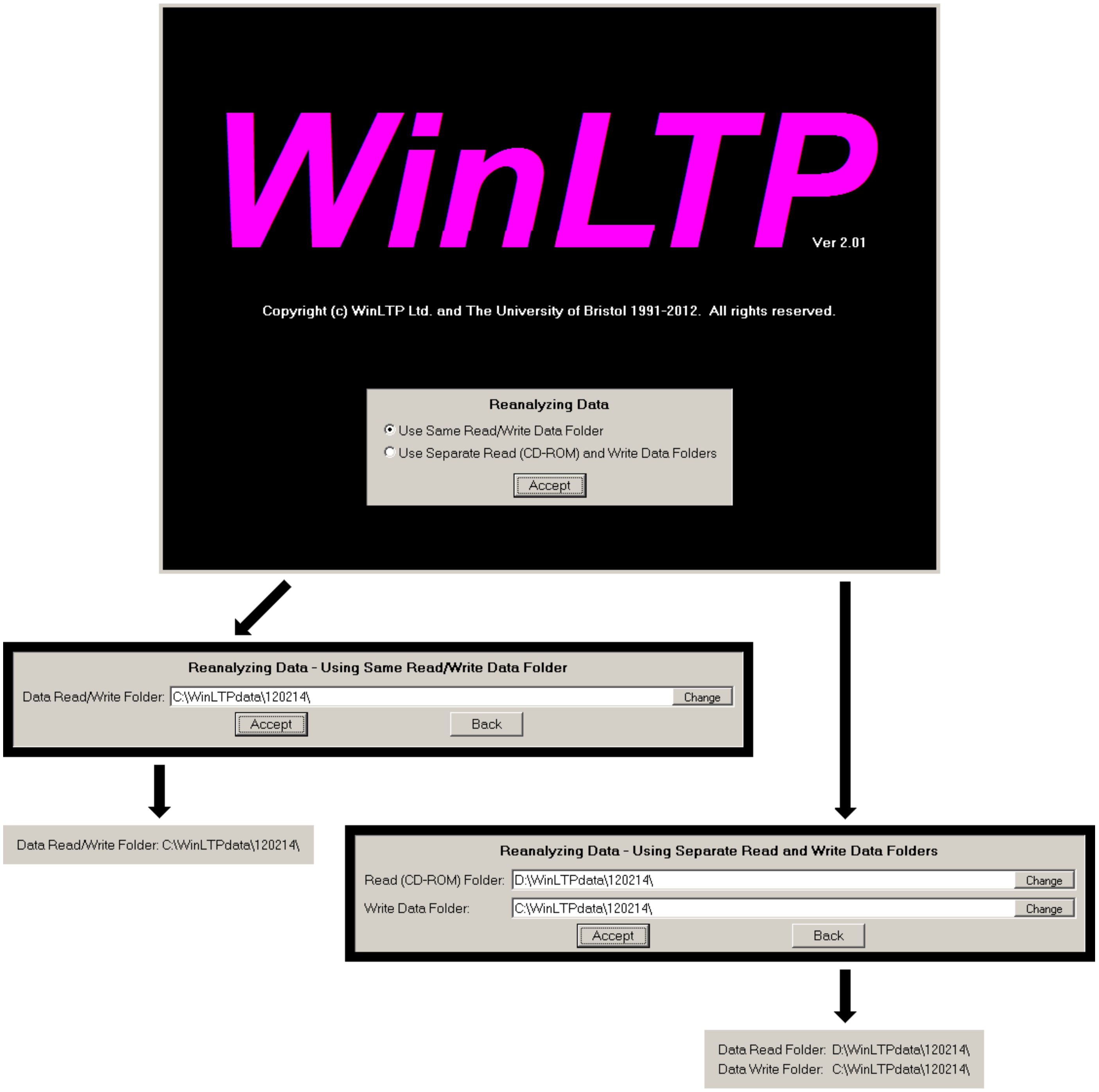
NOTE: There may be a video related BUG with early versions of Windows XP. If WinLTP hangs up during start-up (start-up can take at least 15 seconds!), try changing your video to Classic (Windows 2000) mode, or (believe it or not) change the Windows XP font size to Normal if it is Large or Extra Large.

14.2 Reanalyze files from a CD-ROM

After the WinLTP Reanalysis program has started, the **Reanalyzing Data** dialog box appears (Fig.14.2.1, top) which allows you to either **Use Same Read/Write Data Folder** if reanalyzing ADsweep files from the hard drive, or **Use Separate Read (CD-ROM) & Write Data Folders** if reanalyzing ADsweep files that have been previously saved on a read-only CD-ROM. Because you cannot write the analysis (*.AMP) and LaserJet (*.LJ) files to the read-only data folder on the CD-ROM, a different Data Write folder on your hard disk must be chosen.

If you choose the **Use Same Read/Write Data Folder** selection, the **Reanalyzing Data – Using Same Read/Write Data Folder** dialog box appears (Fig. 14.2.1, left middle panel). Change the Data Read/Write Folder by clicking on the 'Change' button to call up a **Change Read/Write Data Folder** dialog box (similar to the one shown in Fig. 2.6.2) and select a new folder, and then press the 'Accept' button to set that as your Data Read/Write Folder (see panel below left in Fig. 14.2.1).

Alternatively, if you choose the **Use Separate Read (CD-ROM) & Write Data Folders** selection, the **Reanalyzing Data – Using Separate Read and Write Data Folders** dialog box appears (Fig. 14.2.1, lower right panel). Change the Data Read or Data Write Folder by clicking on the 'Change' button to call up a **Change Data Read Folder** or a **Change Data Write Folder** dialog box (similar to the one shown in Fig. 2.6.2) and select a new folder, and then press the 'Accept' button to set those as your Data Read Folder and Data Write folder (see bottom panel Fig. 14.2.1).



Copyright (c) WinLTP Ltd. and The University of Bristol 1991-2012. All rights reserved.

Reanalyzing Data
- [✓ Use Same Read/Write Data Folder]
- Use Separate Read (CD-ROM) and Write Data Folders

Reanalyzing Data - Using Same Read/Write Data Folder
Data Read/Write Folder: C:\WinLTPdata\120214\

Data Read/Write Folder: C:\WinLTPdata\120214\

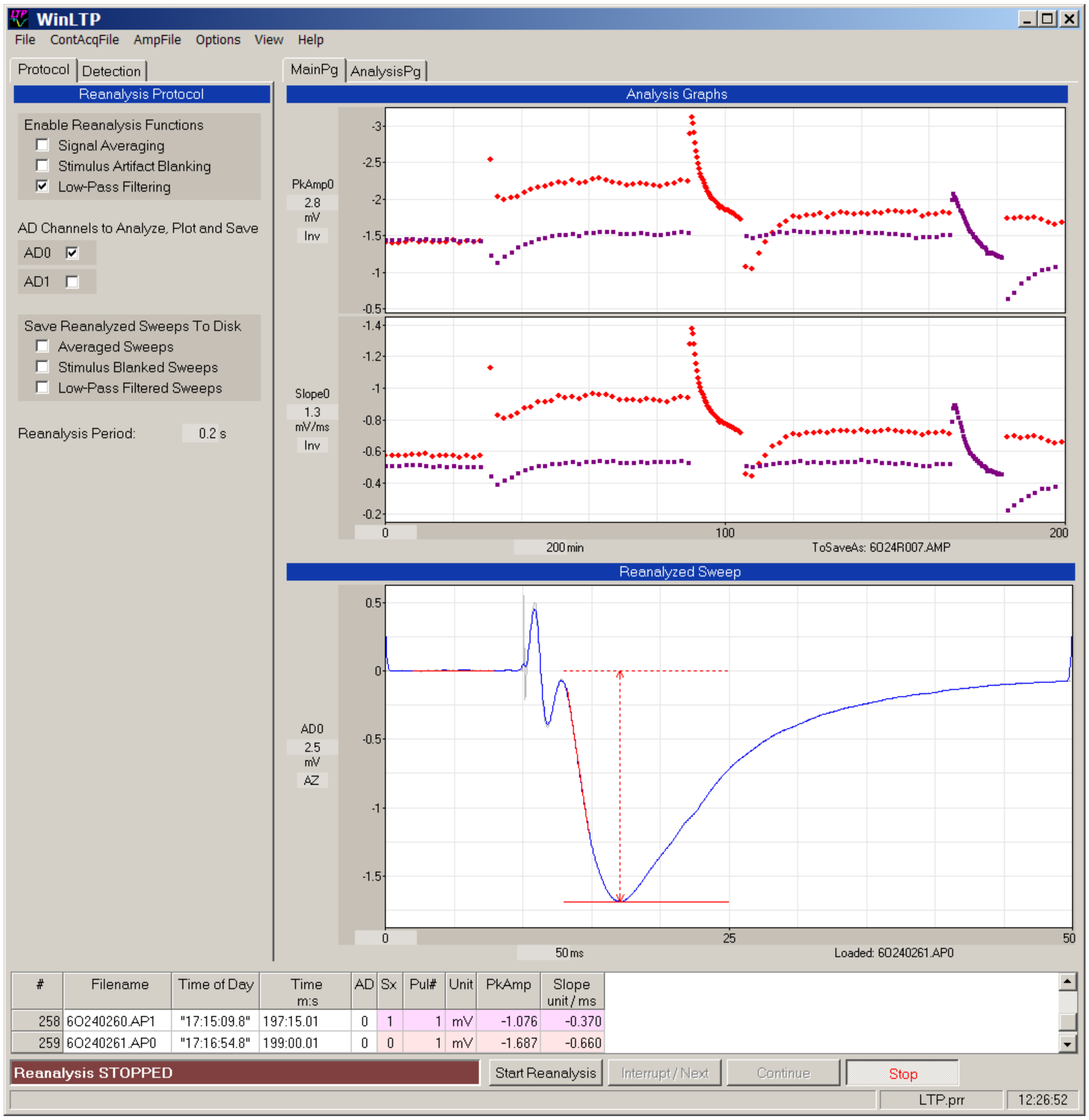
Reanalyzing Data - Using Separate Read and Write Data Folders
Read (CD-ROM) Folder: D:\WinLTPdata\120214\
Write Data Folder: C:\WinLTPdata\120214\

Data Read Folder: D:\WinLTPdata\120214\
Data Write Folder: C:\WinLTPdata\120214\

Fig. 14.2.1. The Dialog boxes encountered when starting the WinLTP reanalysis program.

14.3 Organization of the Reanalysis Program

The layout of the WinLTP Reanalysis program (Fig. 14.3.1) is a simplification of the Online program.



Reanalysis Protocol

Enable Reanalysis Functions
- ☐ Signal Averaging
- ☐ Stimulus Artifact Blanking
- ☑ Low-Pass Filtering

AD Channels to Analyze, Plot and Save
- AD0 ☑
- AD1 ☐

Save Reanalyzed Sweeps To Disk
- ☐ Averaged Sweeps
- ☐ Stimulus Blanked Sweeps
- ☐ Low-Pass Filtered Sweeps

Reanalysis Period: 0.2 s

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	PkAmp	Slope unit / ms
258	6O240260.AP1	"17:15:09.8"	197:15.01	0	1	1	mV	-1.076	-0.370
259	6O240261.AP0	"17:16:54.8"	199:00.01	0	0	1	mV	-1.687	-0.660

Reanalysis STOPPED

Fig. 14.3.1. Layout of the WinLTP Reanalysis program showing the Reanalysis Protocol tabsheet (upper left panel), Analysis Graphs (PkAmp and Slope, top right panel), Reanalyzed Sweep (middle right panel), and Spreadsheet and Run Buttons (bottom panels). Detection fields are hidden behind the Reanalysis Protocol tabsheet. In this figure the S0 and S1 pulse stimulation information is taken from the ADsweep disk file header, and therefore no sweep Stimulation Template fields or graphs appear.

The Reanalyzed Sweep graph shows a fEPSP evoked by S0 stimulation (10 ms after the start of the sweep) with a red lines showing the Slope and PkAmp. The slope graph shows calculations of slope for S0-evoked fEPSPs (red triangles) and S1-evoked fEPSPs (magenta squares) caused by the alternating S0/S1 pathway stimulation produced by alternating P0/P1 sweeps.

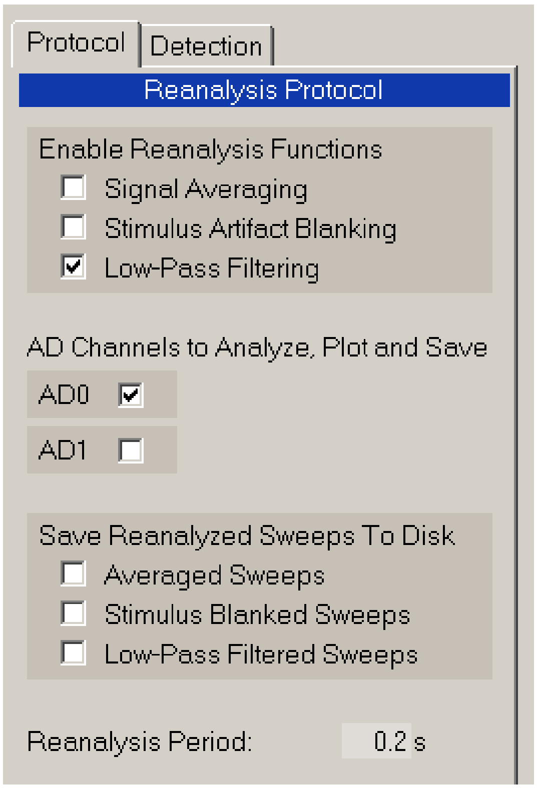
In the spreadsheet, “Time of Day” shows the time the sweep began, “Time m:s” shows the time of the stimulus pulse from when analysis starts, “Sx” shows whether S0 or S1 stimulation was used, “Pul#” shows the number of the S0 or S1 pulse that evokes the synaptic response, and “PkAmp” and “Slope” shows the calculated peak amplitude and slope of the evoked response.

In Fig. 14.3.1, the S0 and S1 pulse stimulation information is taken from the ADsweep disk file header, not Stimulation Template fields, and therefore no Sweep Stimulation template fields or graphs appeared. Figs. 14.5.2 and 14.5.3 show the appearance of the Stimulation Template fields and graphs in the WinLTP Reanalysis program, which are visible due to reanalysis using Screen Field stimulation information (Section 14.5).

As part of the Online/Acquisition to Reanalysis simplification, the Protocol tabsheet for the Online/Acquisition program containing the MainProtocol, EvokedEvent, and Plot/Save tabasheets have been simplified and combined into the Reanalysis Protocol tabsheet in the Reanalysis program (Fig. 14.3.1 and 14.3.2).

The Reanalysis Protocol tabsheet is comprised of four sections:

a) Enable Reanalysis Functions which sets whether reanalysis uses Signal Averaging, Stimulus Artifact Blanking and/or Low-Pass Filtering
b) AD Channels to Analyze, Plot and Save
c) Save Reanalyzed Sweeps to Disk to set wheter Averaged Sweeps, Stimulus Blanked Sweeps and/or Low-Pass Filtered Sweeps will be saved to disk
d) The Reanalysis Period field, which sets how fast ADsweep files will be reanalyzed (with 0 seconds being no delay between ADsweep file reanalyses)



Reanalysis Protocol

Enable Reanalysis Functions
- [ ] Signal Averaging
- [ ] Stimulus Artifact Blanking
- [x] Low-Pass Filtering

AD Channels to Analyze, Plot and Save
- AD0 [x]
- AD1 [ ]

Save Reanalyzed Sweeps To Disk
- [ ] Averaged Sweeps
- [ ] Stimulus Blanked Sweeps
- [ ] Low-Pass Filtered Sweeps

Reanalysis Period: 0.2 s

Fig. 14.3.2. The Reanalysis Protocol tabsheet. Compare these Reanalysis functions with the Online/Acquisition functions in Fig. 3.1.2.1 and the right hand panel of Fig. 3.1.2.2.

In contrast to the changed Protocol section, the Detection tabsheet in the Reanalysis program looks and functions the same as in the Online/Acquisition program (see Fig. 3.1.2.3).

Note that the Run buttons now consist of only the **‘Start Reanalysis’**, **‘Interrupt/Next’**, **‘Continue’** and **‘Stop’** buttons.

14.4 Changes in Menus and Dialog Boxes

In the **Protocol File Menu** in the Reanalysis program (Fig. 14.4.1) the **AutoCreate new data Folder...** menu choice in the Online/Acquisition program (Fig. 3.2.1) is not present because automatically creating new data folders only makes sense during acquisition. You can still create a new data folder during reanalysis using the **Change data folder...** menu choice, but you specifically have to write in a name for the new folder.

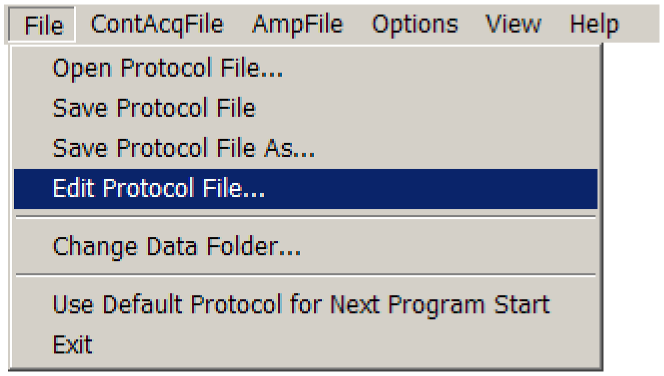
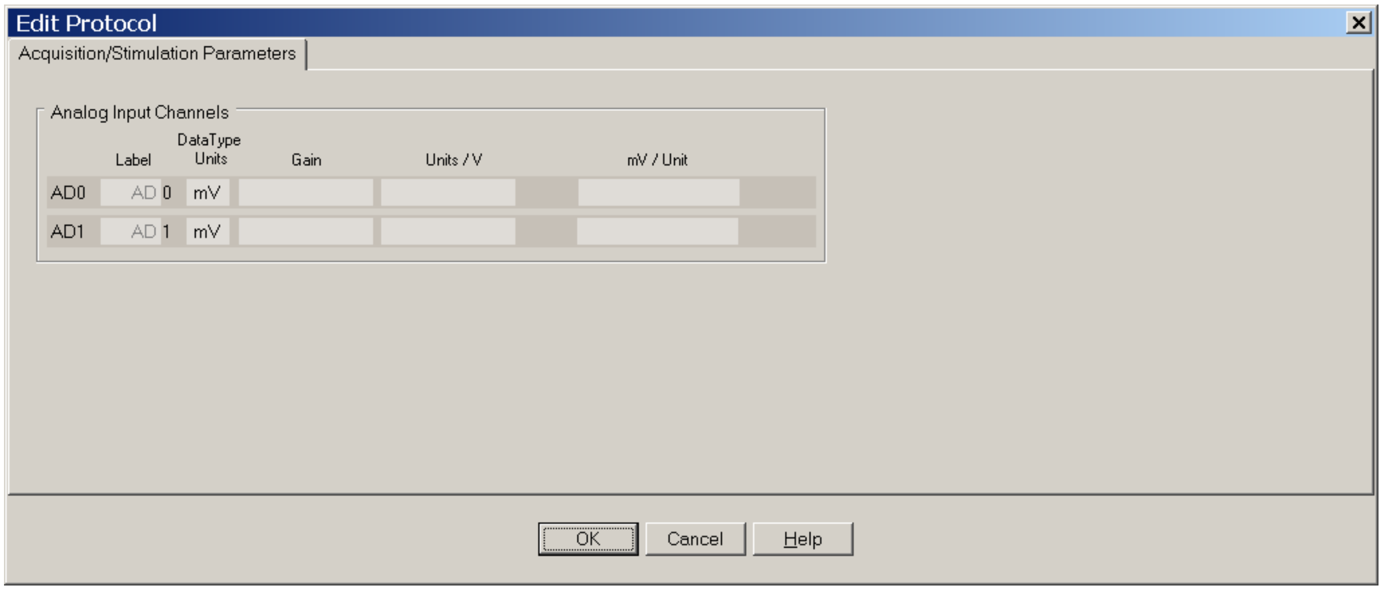


Fig. 14.4.1. The Protocol File Menu available in the Reanalysis program.

In the Reanalysis program, the Edit Protocol dialog box (Fig. 14.4.2) has been simplified to include only the Acquisition/Stimulation Parameters tabsheet that allows only the AD0 and AD1 channel Data Type Units (pA, mV and V) to be set identical to that of the ADsweep file, and the ICO Data Type Unit (pA, nA, mV or V) to set IC0 analog output stimulation for reanalyzing Rs and Rm.



	Label	DataType Units	Gain	Units / V	mV / Unit
AD0	AD 0	mV			
AD1	AD 1	mV			

Fig. 14.4.2. The simplified Edit Protocol dialog box used in the Reanalysis program. Only the AD0 and AD1 Data Type Units are active.

The AmpFile Menu in the Reanalysis program (Fig. 14.4.3) is actually enhanced compared to that in the Online/Acquisition program (see Fig. 3.2.3). In particular, the following menu items have been added:

a) Reanalyze Amplitudes/Slopes from ADsweep Files…
b) Reanalyze Using Stimulus Info from ScreenFields or DiskFiles…
c) AutoClear Analysis Graphs for Next Reanalysis…

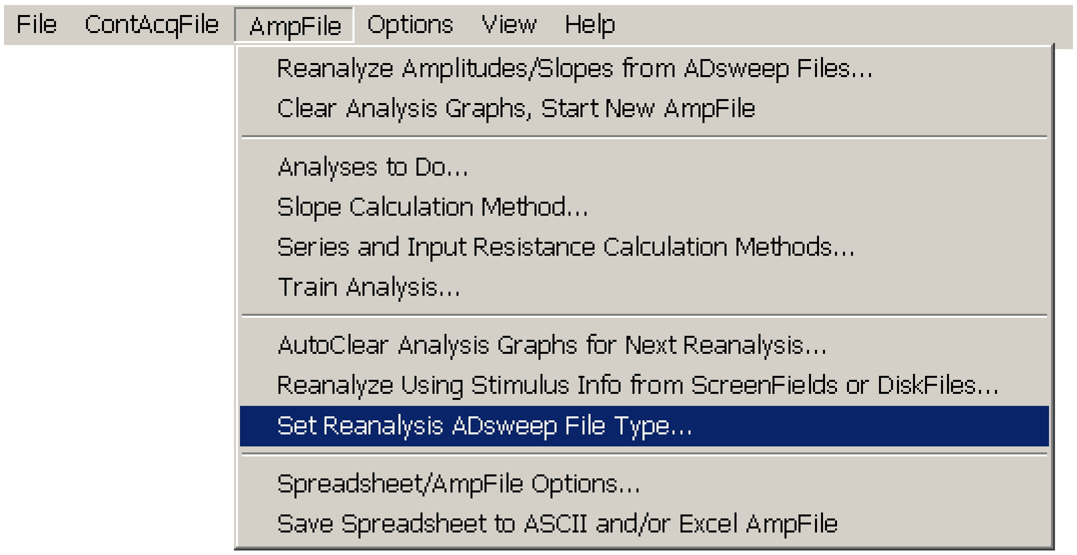
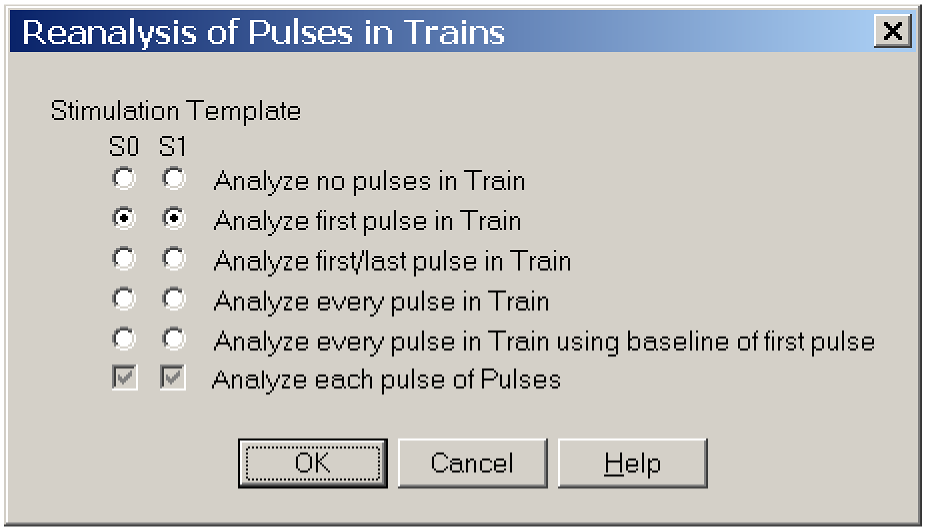

File
ContAcqFile
AmpFile
Options
View
Help
Reanalyze Amplitudes/Slopes from ADsweep Files...
Clear Analysis Graphs, Start New AmpFile
Analyses to Do...
Slope Calculation Method...
Series and Input Resistance Calculation Methods...
Train Analysis...
AutoClear Analysis Graphs for Next Reanalysis...
Reanalyze Using Stimulus Info from ScreenFields or DiskFiles...
Set Reanalysis ADsweep File Type...
Spreadsheet/AmpFile Options...
Save Spreadsheet to ASCII and/or Excel AmpFile

Fig. 14.4.3. The enhanced AmpFile Menu of the Reanalysis program.

When doing Reanalysis of Pulses in Trains, because ADsweep files are only loaded into the Reanalysis Sweep (P0) array, it is not necessary to set this information for Sweeps P1, T0 or T1 as in the Analysis of Pulses in Trains dialog box used in the Online/Acquisition program (Fig. 4.11.1).

If you wish to reanalyze Train stimulations, call up the Reanalysis of Trains dialog box (Fig. 14.4.4)by using the menu command (Fig. 14.4.3):

AmpFile -> Train Annalysis…


Reanalysis of Pulses in Trains

Stimulation Template

S0	S1	
○	○	Analyze no pulses in Train
●	●	Analyze first pulse in Train
○	○	Analyze first/last pulse in Train
○	○	Analyze every pulse in Train
○	○	Analyze every pulse in Train using baseline of first pulse
✓	✓	Analyze each pulse of Pulses

Fig. 14.4.4. The simplified Reanalysis of Pulses in Trains dialog box used in Reanalysis program.

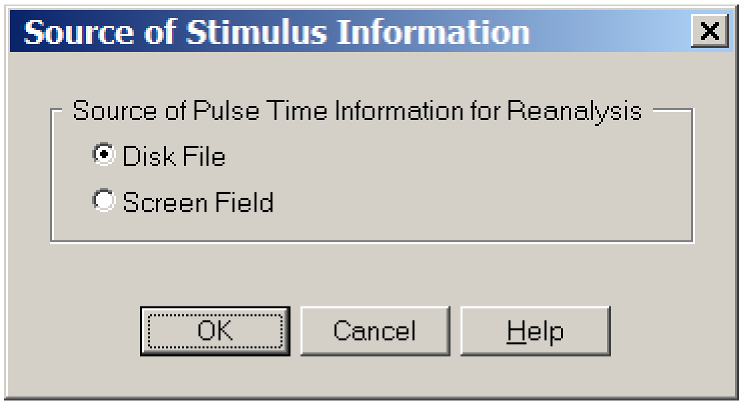
Note, that if you wish to reanalyze as trains you must set the Epoch stimulation in the Acqusition program to Trains (Fig. 4.7.1.3).

14.5 Source of Pulse Stimulation Information for Reanalysis

Before starting reanalysis, one has to choose whether the time of the start of the S0, S1 or Rs/Rm pulse is taken from the ADsweep Disk File header, or from the Pulse Stimulation Screen Fields. Doing reanalysis in the same program that stimulates and acquires data has the added advantage of allowing one to use the Pulse Stimulation fields like a stimulus template and allows you to place the S0, S1 or Rm pulses at whatever time you want. To set the source of the Pulse Stimulation information, use the AmpFile menu command (Fig. 14.4.3):

AmpFile -> Reanalyze using ScreenField or DiskFile information

to bring up the Source of Stimulation Information dialog box (Fig. 14.5.1):



Source of Pulse Time Information for Reanalysis

[✓ Disk File]

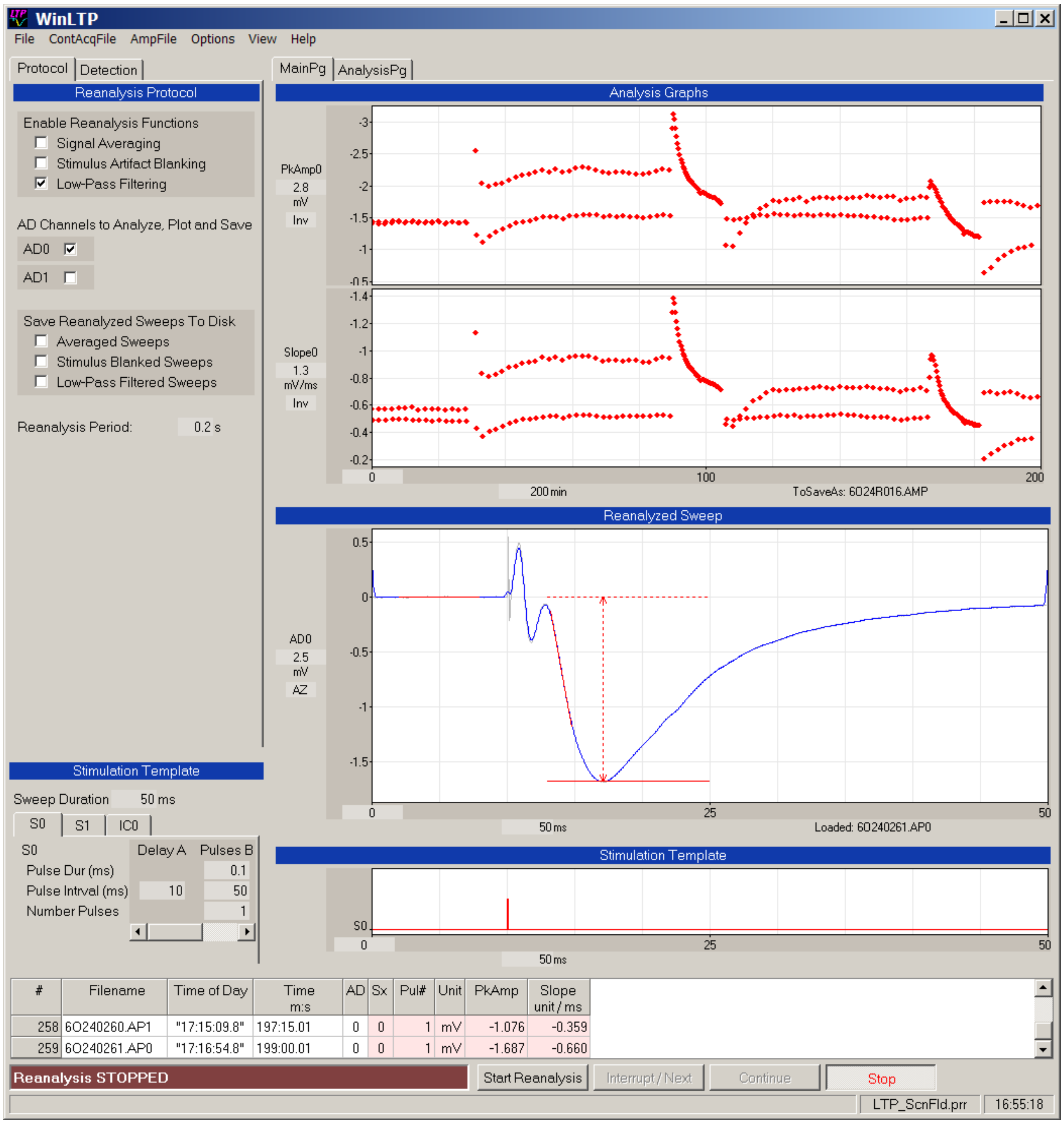
Screen Field

Fig. 14.5.1. The Source of Reanalysis Pulse Stimulus Information dialog box.

During on-line analysis the pulse time information contained in the Pulse Stimulation Screen Fields in the Pulse Stimulation Windows is used. During off-line Reanalysis, normally the pulse time information is obtained from the ADsweep Disk File (the default). If the **Disk File** option is chosen, the appearance of the WinLTP Reanalysis program is as shown in Fig. 14.3.1.

However, if you want to reanalyze with a different stimulation pattern, say only the second synaptic potential of a paired pulse stimulation, or reanalyzing from an ASCII file with no header, you can choose the Source of Pulse Time Information for Reanalysis to be the **Screen Fields**. If the If the **Screen Field** option is chosen, the appearance of the WinLTP Reanalysis program is as shown in Fig. 14.5.2, which now includes the Screen Fields of the Stimulation Template and the Stimulation Template graphs (Fig. 14.5.3).

Note that because the Screen Field information is used, this cannot necessarily distinguish between S0 and S1 pulses that are different in the AP0 and AP1 files when they are analyzed at once – hence only the red triangles and not the magenta squares appears in the Analysis graphs of Fig. 14.5.2 (contrast with the Disk File analysis in Fig. 14.3.1). In order to separate S0 and S1 pulses, the AP0 and AP1 ADsweep files would have had to have been analyzed separately, with the AP1 files using S1 (and not S0) in the Stimulation Template.



Reanalysis Protocol

Enable Reanalysis Functions
- [ ] Signal Averaging
- [ ] Stimulus Artifact Blanking
- [x] Low-Pass Filtering

AD Channels to Analyze, Plot and Save
- AD0 [x]
- AD1 [ ]

Save Reanalyzed Sweeps To Disk
- [ ] Averaged Sweeps
- [ ] Stimulus Blanked Sweeps
- [ ] Low-Pass Filtered Sweeps

Reanalysis Period: 0.2 s

Stimulation Template

Sweep Duration 50 ms

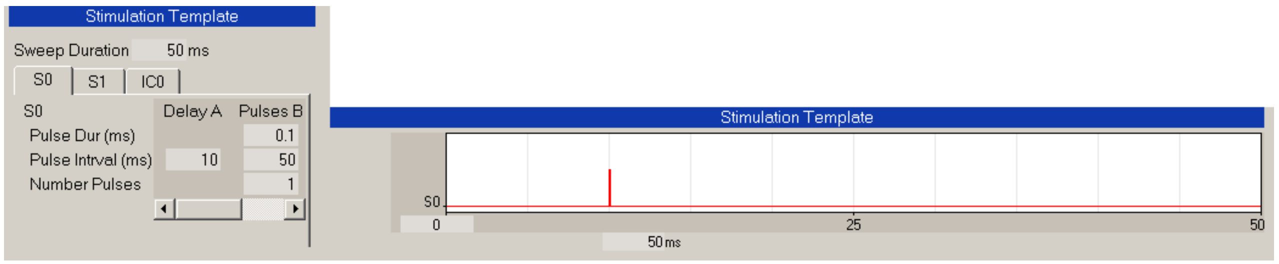
S0 | S1 | IC0

S0	Delay A	Pulses B
Pulse Dur (ms)		0.1
Pulse Intrval (ms)	10	50
Number Pulses		1

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	PkAmp	Slope unit / ms
258	6O240260.AP1	"17:15:09.8"	197:15.01	0	0	1	mV	-1.076	-0.359
259	6O240261.AP0	"17:16:54.8"	199:00.01	0	0	1	mV	-1.687	-0.660

Reanalysis STOPPED

Fig. 14.5.2. Layout of the WinLTP Reanalysis program with stimulation information coming from the Screen Fields of the Stimulation Template. S1 stimulation is OFF. Note only S0 red triangles appear in the Analysis graphs.



Stimulation Template

Sweep Duration 50 ms

S0 | S1 | IC0

S0	Delay A	Pulses B
Pulse Dur (ms)		0.1
Pulse Intrval (ms)	10	50
Number Pulses		1

Fig. 14.5.3. The Stimulation Template consisting of Screen Fields on the left, and stimulation graphs on the right (only S0 graph is used and shown). The key field is the ‘Delay A’ Pulse Interval (pulse delay) field.

14.6 Running the Reanalysis

To reanalyze data either click on the button:

'Start Reanalysis'

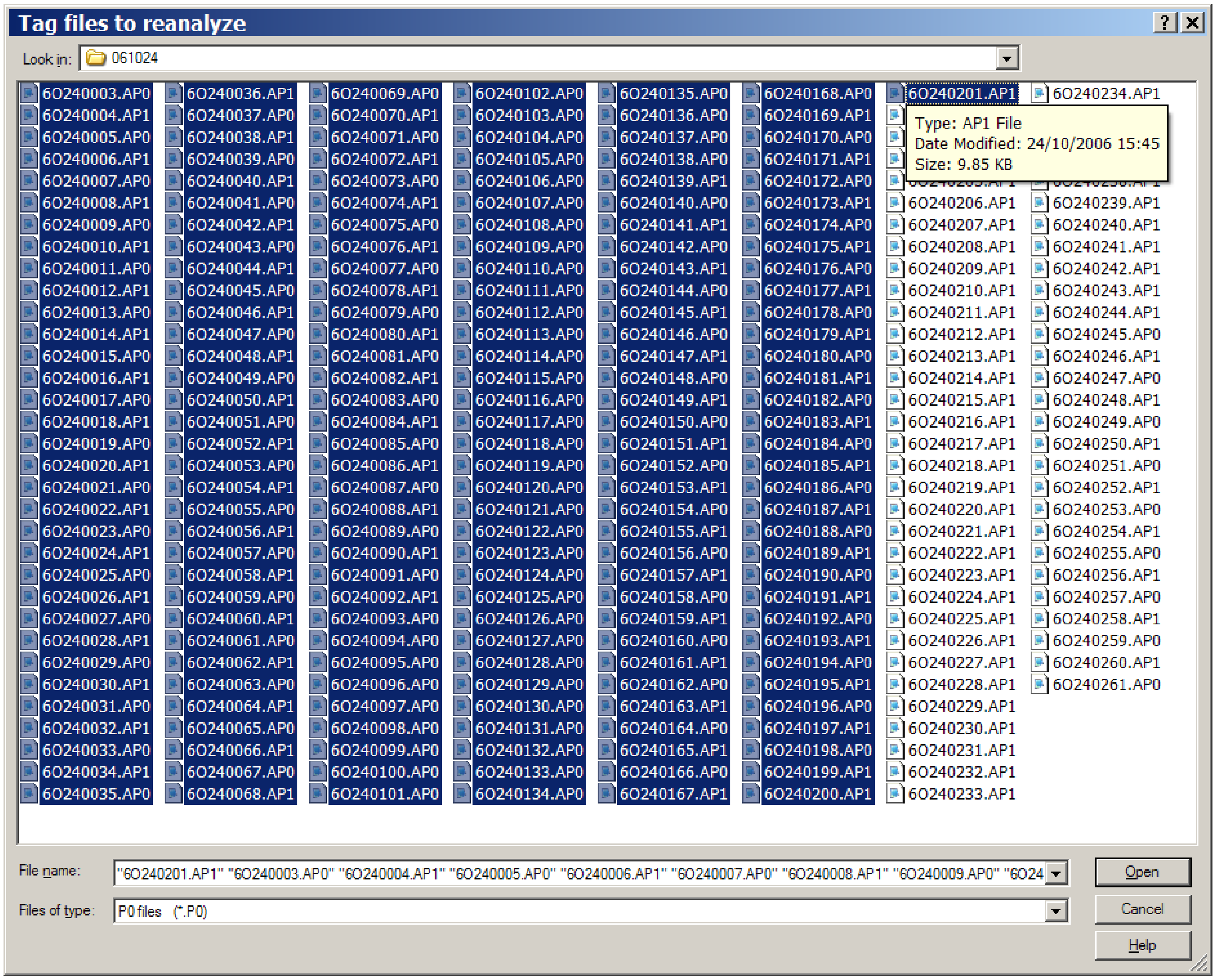
or enter the menu use the menu command:

AmpFile -> Reanalyze AD sweep files

to bring up the Tab Files to Rreanalyze dialog box (Fig. 14.6.1).

Remember, every data file is 1 sweep (or 1 averaged sweep), so you can select only those sweeps you wish to reanalyze.

Note that by dragging on the lower right corner of this dialog box you can **expand it greatly** and view many more files. Furthermore, when you exit this dialog box and re-enter it, the last (expanded) size will be the one brought up.



Tag files to reanalyze

Look in: 061024

6O240003.AP0	6O240036.AP1	6O240069.AP0	6O240102.AP0	6O240135.AP0	6O240168.AP0	6O240201.AP1	6O240234.AP1
6O240004.AP1	6O240037.AP0	6O240070.AP1	6O240103.AP0	6O240136.AP0	6O240169.AP1		
6O240005.AP0	6O240038.AP1	6O240071.AP0	6O240104.AP0	6O240137.AP0	6O240170.AP0		
6O240006.AP1	6O240039.AP0	6O240072.AP1	6O240105.AP0	6O240138.AP0	6O240171.AP1		
6O240007.AP0	6O240040.AP1	6O240073.AP0	6O240106.AP0	6O240139.AP1	6O240172.AP0		
6O240008.AP1	6O240041.AP0	6O240074.AP1	6O240107.AP0	6O240140.AP0	6O240173.AP1	6O240206.AP1	6O240239.AP1
6O240009.AP0	6O240042.AP1	6O240075.AP0	6O240108.AP0	6O240141.AP1	6O240174.AP0	6O240207.AP1	6O240240.AP1
6O240010.AP1	6O240043.AP0	6O240076.AP1	6O240109.AP0	6O240142.AP0	6O240175.AP1	6O240208.AP1	6O240241.AP1
6O240011.AP0	6O240044.AP1	6O240077.AP0	6O240110.AP0	6O240143.AP1	6O240176.AP0	6O240209.AP1	6O240242.AP1
6O240012.AP1	6O240045.AP0	6O240078.AP1	6O240111.AP0	6O240144.AP0	6O240177.AP1	6O240210.AP1	6O240243.AP1
6O240013.AP0	6O240046.AP1	6O240079.AP0	6O240112.AP0	6O240145.AP1	6O240178.AP0	6O240211.AP1	6O240244.AP1
6O240014.AP1	6O240047.AP0	6O240080.AP1	6O240113.AP0	6O240146.AP0	6O240179.AP1	6O240212.AP1	6O240245.AP0
6O240015.AP0	6O240048.AP1	6O240081.AP0	6O240114.AP0	6O240147.AP1	6O240180.AP0	6O240213.AP1	6O240246.AP1
6O240016.AP1	6O240049.AP0	6O240082.AP1	6O240115.AP0	6O240148.AP0	6O240181.AP1	6O240214.AP1	6O240247.AP0
6O240017.AP0	6O240050.AP1	6O240083.AP0	6O240116.AP0	6O240149.AP1	6O240182.AP0	6O240215.AP1	6O240248.AP1
6O240018.AP1	6O240051.AP0	6O240084.AP1	6O240117.AP0	6O240150.AP0	6O240183.AP1	6O240216.AP1	6O240249.AP0
6O240019.AP0	6O240052.AP1	6O240085.AP0	6O240118.AP0	6O240151.AP1	6O240184.AP0	6O240217.AP1	6O240250.AP1
6O240020.AP1	6O240053.AP0	6O240086.AP1	6O240119.AP0	6O240152.AP0	6O240185.AP1	6O240218.AP1	6O240251.AP0
6O240021.AP0	6O240054.AP1	6O240087.AP0	6O240120.AP0	6O240153.AP1	6O240186.AP0	6O240219.AP1	6O240252.AP1
6O240022.AP1	6O240055.AP0	6O240088.AP1	6O240121.AP0	6O240154.AP0	6O240187.AP1	6O240220.AP1	6O240253.AP0
6O240023.AP0	6O240056.AP1	6O240089.AP0	6O240122.AP0	6O240155.AP1	6O240188.AP0	6O240221.AP1	6O240254.AP1
6O240024.AP1	6O240057.AP0	6O240090.AP1	6O240123.AP0	6O240156.AP0	6O240189.AP1	6O240222.AP1	6O240255.AP0
6O240025.AP0	6O240058.AP1	6O240091.AP0	6O240124.AP0	6O240157.AP1	6O240190.AP0	6O240223.AP1	6O240256.AP1
6O240026.AP1	6O240059.AP0	6O240092.AP1	6O240125.AP0	6O240158.AP0	6O240191.AP1	6O240224.AP1	6O240257.AP0
6O240027.AP0	6O240060.AP1	6O240093.AP0	6O240126.AP0	6O240159.AP1	6O240192.AP0	6O240225.AP1	6O240258.AP1
6O240028.AP1	6O240061.AP0	6O240094.AP0	6O240127.AP0	6O240160.AP0	6O240193.AP1	6O240226.AP1	6O240259.AP0
6O240029.AP0	6O240062.AP1	6O240095.AP0	6O240128.AP0	6O240161.AP1	6O240194.AP0	6O240227.AP1	6O240260.AP1
6O240030.AP1	6O240063.AP0	6O240096.AP0	6O240129.AP0	6O240162.AP0	6O240195.AP1	6O240228.AP1	6O240261.AP0
6O240031.AP0	6O240064.AP1	6O240097.AP0	6O240130.AP0	6O240163.AP1	6O240196.AP0	6O240229.AP1	
6O240032.AP1	6O240065.AP0	6O240098.AP0	6O240131.AP0	6O240164.AP0	6O240197.AP1	6O240230.AP1	
6O240033.AP0	6O240066.AP1	6O240099.AP0	6O240132.AP0	6O240165.AP1	6O240198.AP0	6O240231.AP1	
6O240034.AP1	6O240067.AP0	6O240100.AP0	6O240133.AP0	6O240166.AP0	6O240199.AP1	6O240232.AP1	
6O240035.AP0	6O240068.AP1	6O240101.AP0	6O240134.AP0	6O240167.AP1	6O240200.AP0	6O240233.AP1	

Type: AP1 File
Date Modified: 24/10/2006 15:45
Size: 9.85 KB

File name: "6O240201.AP1" "6O240003.AP0" "6O240004.AP1" "6O240005.AP0" "6O240006.AP1" "6O240007.AP0" "6O240008.AP1" "6O240009.AP0" "6O24

Files of type: P0 files (*.P0)

Fig. 14.6.1. Tag Files to Reanalyze dialog box. Note that size the dialog box has been expanded compared to the usual small one.

To select the files:

LeftMouseButton click to select the **first** file you want.

Then, either:

Ctrl–LeftMouseButton click – to select the **next** file you want to select, and then Ctrl–LeftMouseButton click to select the third file, and so forth.

Shift-LeftMouseButton click – to select all the files between the first file you selected and the second file you shift clicked on.

Shift-LeftArrowKey key press - to select another column of files.

Shift-EndKey key press - to select all the files between the first file you selected and the last file in the dialog box.

The speed at which the analysis of the chosen ADsweep files occurs is the interval between sweep reanalysis set by the
Reanalysis Period
field in the Reanalysis Protocol tabsheet (Fig. 14.3.2).

Once reanalysis has begun, it can be temporarily halted anytime by pressing the **SPACE** bar or clicking on the **'Interrupt/Next'** button.. The detection values can be changed and that will change the calculation of the slope in the current ADsweep file. The ADsweep graph values can also be changed, and different Windows can be chosen.

Pressing the **SPACE** bar or clicking on the **'Interrupt/Next'** button again will move to the next ADsweep so this will allow single stepping through all the ADsweeps and customizing each slope detection if need be. (However, it is less biased to have one detection setting for reanalyzing the whole experiment.)

Pressing **F3** or clicking on the **'Continue'** button will resume or **continue** the reanalysis at the normal Sweep Repeat Period.

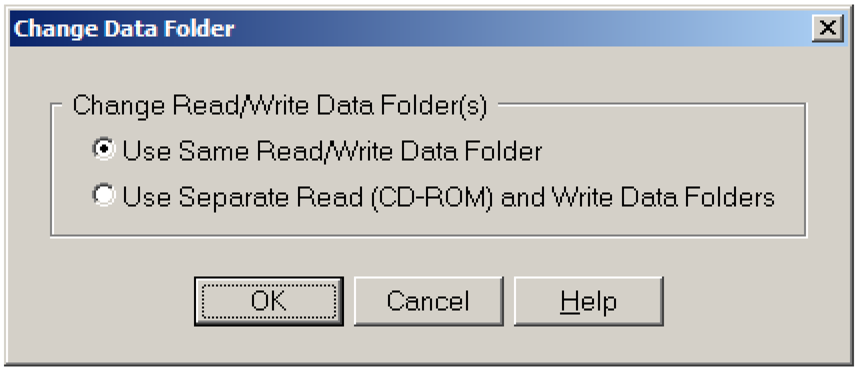
Pressing **F4** or clicking on the **'Stop'** button will **stop** the Reanalysis.

Remember that during a reanalysis run, changing the detection parameters will change the amplitude/slope of last analyzed ADsweep, and will change that value in the Amp/Slope (*.AMP) file. **Therefore if you are changing detection parameters 'between' reanalysis runs, do so on the first sweep of the next reanalysis run and not on the last ADsweep of the previous reanalysis run (e.g. supposedly 'between' reanalysis runs). Alternatively you can clear the calculation graphs before proceeding.**

14.7 Change Data Folder During Reanalysis

If you wish to change to a different data folder during reanalysis, bring up the **Change Data Folder** dialog box (Fig. 14.7.1) by using the menu command (Fig. 14.4.1):

File -> Change data folder…


Change Data Folder

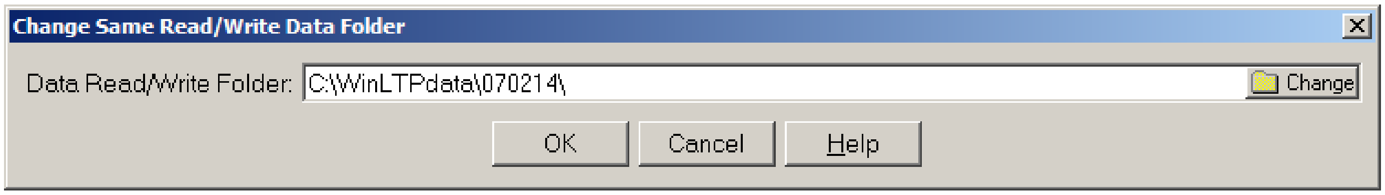
Change Read/Write Data Folder(s)

[✓ Use Same Read/Write Data Folder]

Use Separate Read (CD-ROM) and Write Data Folders

Fig. 14.7.1. Changed Data Folder dialog box.

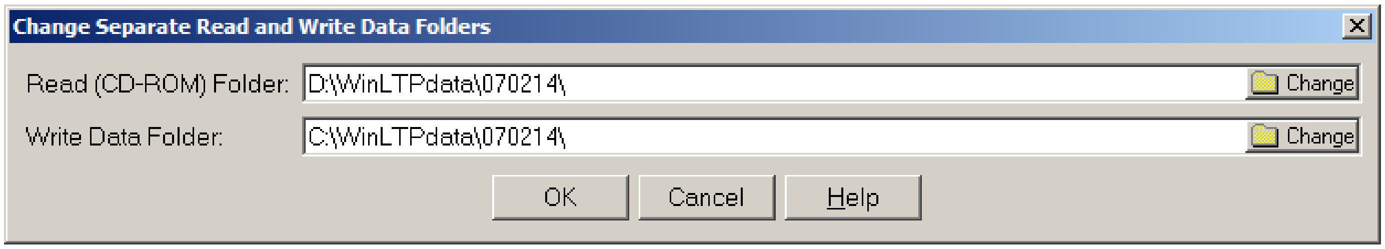
As with the **Reanalyzing Data** dialog box at the top of Fig. 14.2.1, this **Change Data Folder** dialog box allows you to pick **Use Same Read/Write Data Folder** choice to present the **Change Same Read/Write Data Folder** dialog box in Fig. 14.7.2. Changing this dialog box can set the Read/Write drive and data folder simultaneously and would be used to reanalyze data from a hard drive.


Change Same Read/Write Data Folder

Data Read/Write Folder: C:\WinLTPdata\070214\

Fig. 14.7.2 Change Same Read/Write Data Folder dialog box.

Alternatively, choosing **Use Separate Read (CD-ROM) & Write Data Folders** in the **Change Data Folder** dialog box (Fig. 14.7.1) allows you to set the separate Read and Write data drive and data folder by presenting the **Change Separate Read (CD-ROM) & Write Data Folders** dialog box in Fig. 14.7.3. Changing this dialog box can set the Read data drive and folder to say **D:\WinLTPdata\070214** on the read-only CD-ROM, and the Write data drive and folder to say **C:\WinLTPdata\070214** on the hard drive. This is primarily used when reading the ADsweep data from a read-only CD-ROM, while writing the analysis results (*.AMP files as well as newly averaged, blanked and filtered ADsweep files) to the hard disk. Fig. 14.7.4 shows the message line about what the Read and Write Data Folders are after setting the Separate Read (CD-ROM) and Write Data Folders.


Change Separate Read and Write Data Folders

Read (CD-ROM) Folder: D:\WinLTPdata\070214\

Write Data Folder: C:\WinLTPdata\070214\

Fig.14.7.3. Change Separate Read (CD-ROM) and Write Data Folders dialog box.

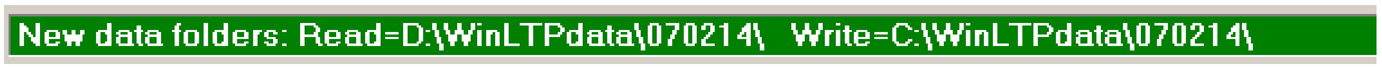

New data folders: Read=D:\WinLTPdata\070214\ Write=C:\WinLTPdata\070214\

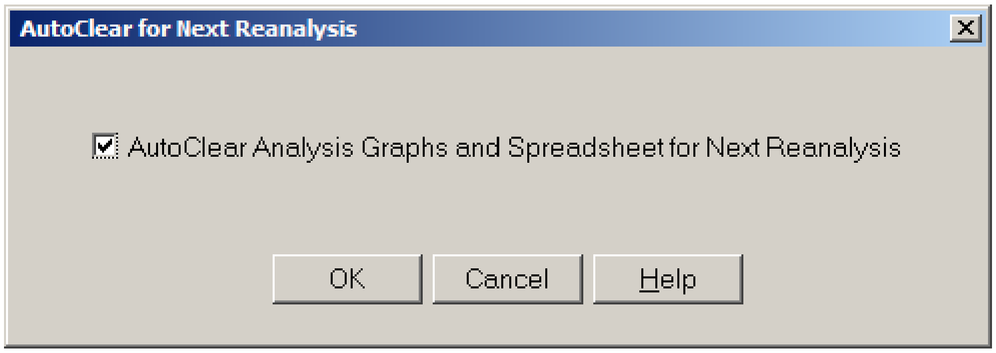
Fig. 14.7.4. Status Bar message after setting Separate Read (CD-ROM) and Write Data Folders.

14.8 Automatically Clear Current Reanalysis at Start of Next Reanalysis

Sometimes you want to do then next reanalysis **on the same calculation graph** containing previously analyzed data points, and sometimes you want to do the next reanalysis on a **new cleared** calculation graph. This can be controlled by using the menu command (Fig. 14.4.3):

AmpFile -> AutoClear Analysis Graphs for Next Rreanalysis…

to bring up the AutoClear for Next Reanalysis dialog box (Fig. 14.8.1).


AutoClear for Next Reanalysis

☑ AutoClear Analysis Graphs and Spreadsheet for Next Reanalysis

Fig. 14.8.1. AutoClear For Next Reanalysis dialog box.

If the check box is checked, then doing the next reanalysis will not only automatically clear the Analysis graphs, but will also automatically clear the Spreadsheet and start a new Amplitude/Calculation (*.AMP) file.

If the check box **is not** checked, then when the next reanalysis is started, the new Amp/Slope calculation points will be **superimposed** on the points already present on the Analysis graphs, and the new data points will be **appended** to the present Spreadsheet and Amplitude/Calculation (*.AMP) file.

14.9 Save an ADsweep Graph as a Windows Enhanced Metafile

WinLTP cannot plot ADsweep graphs. However, it can save ADsweep graphs to vector based Enhanced Metafiles that can be loaded into many other programs such as Microsoft PowerPoint, Excel and Word, and SigmaPlot.

In order to save an ADsweep graph such as the one in Fig. 14.9.1 to an Enhanced Metafile, **double-click on the dark gray area** of the individual ADsweep graph that contains the axis numbers, and this will call up the Enhanced Metafile dialog box (Fig. 14.9.2).

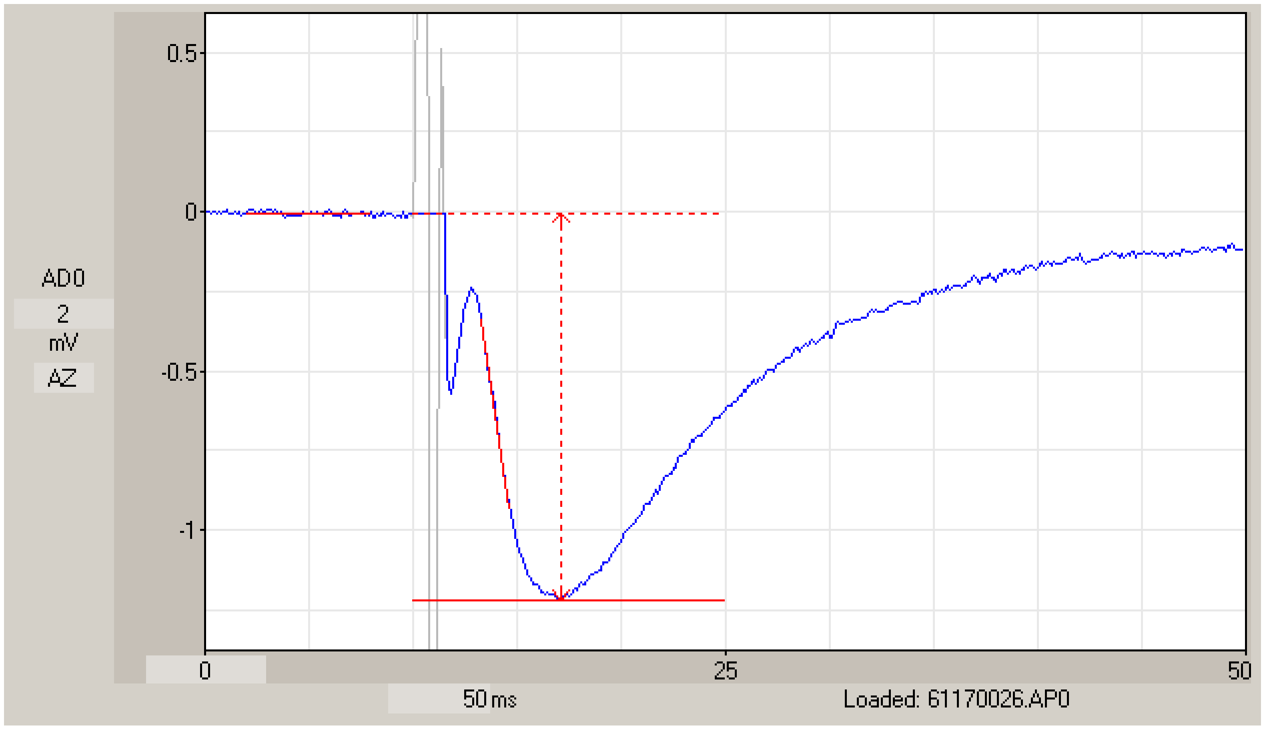
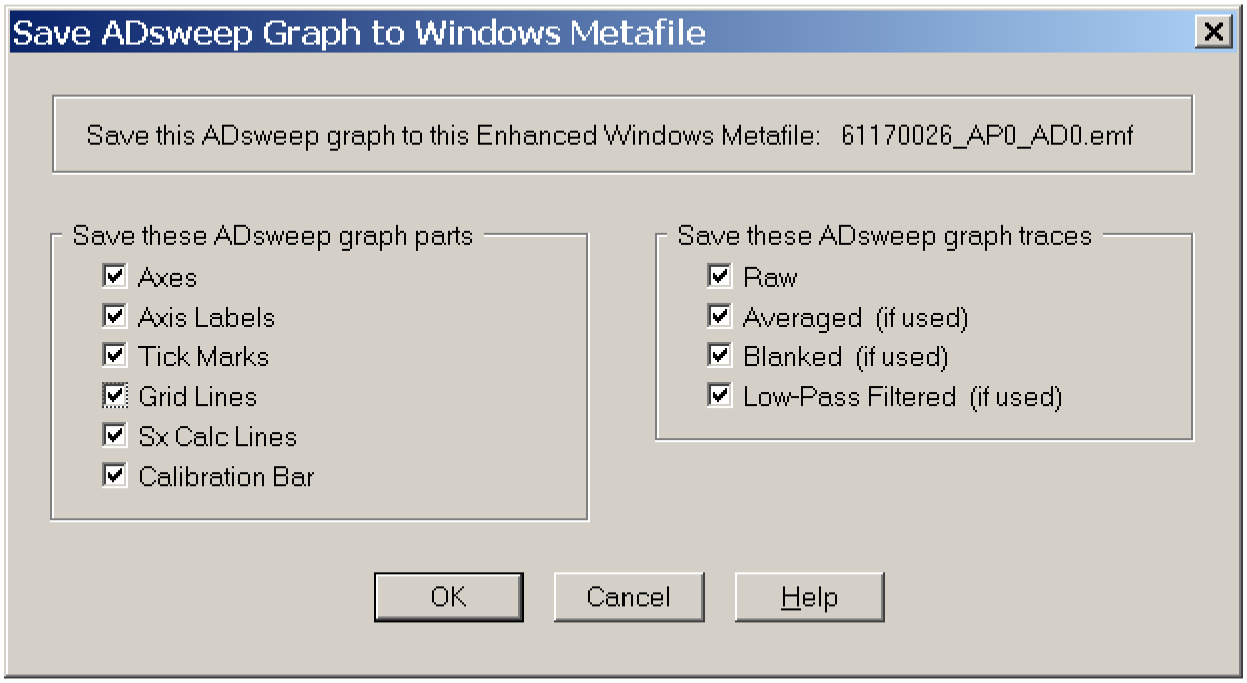


Fig. 14.9.1. An ADsweep graph captured using PrintScreen showing Raw and Hold-Blanked traces, and S0 pulse calclines.

Note that the name of the Enhanced Metafile to be saved in Fig. 14.9.2, “61170026_AP0_AD0.emf“ contains the ADsweep filename “61170026”, its extension “_AP0” and AD channel number “_AD0”. If more than one Metafile was to be saved from one ADsweep graph, it would also carry an additional numeric addition such as “_1” to form the Metafile name “61170026_AP0_AD0_1.emf”.

Then choose which ADsweep graph parts you want to save by checking the check boxes on the left of the Metafile dialog box. These can be any of the following: Axes, Axis Labels, Tick Marks, Grid Lines, Sx Calc Lines and/or Calibration Bar. Next check which ADsweep graph traces to save: Raw, Averaged, Blanked and Low-Pass Filtered. Averaged, Blanked and Filtered traces will only be saved if they exist in the original ADsweep graph in WinLTP.

Then click on the OK button to save the ADsweep file to the “61170026_AP0_AD0.emf” Enhanced Metafile (Fig. 14.9.3).



Save ADsweep Graph to Windows Metafile

Save this ADsweep graph to this Enhanced Windows Metafile: 61170026_AP0_AD0.emf

Save these ADsweep graph parts
- [✓] Axes
- [✓] Axis Labels
- [✓] Tick Marks
- [✓] Grid Lines
- [✓] Sx Calc Lines
- [✓] Calibration Bar

Save these ADsweep graph traces
- [✓] Raw
- [✓] Averaged (if used)
- [✓] Blanked (if used)
- [✓] Low-Pass Filtered (if used)

Fig. 14.9.2. The Save ADsweep Graph to Windows Metafile dialog box. The check boxes checked are used to save the ADsweep graph of Fig. 14.9.1 to the Enhanced Metafile in Fig. 14.9.3.

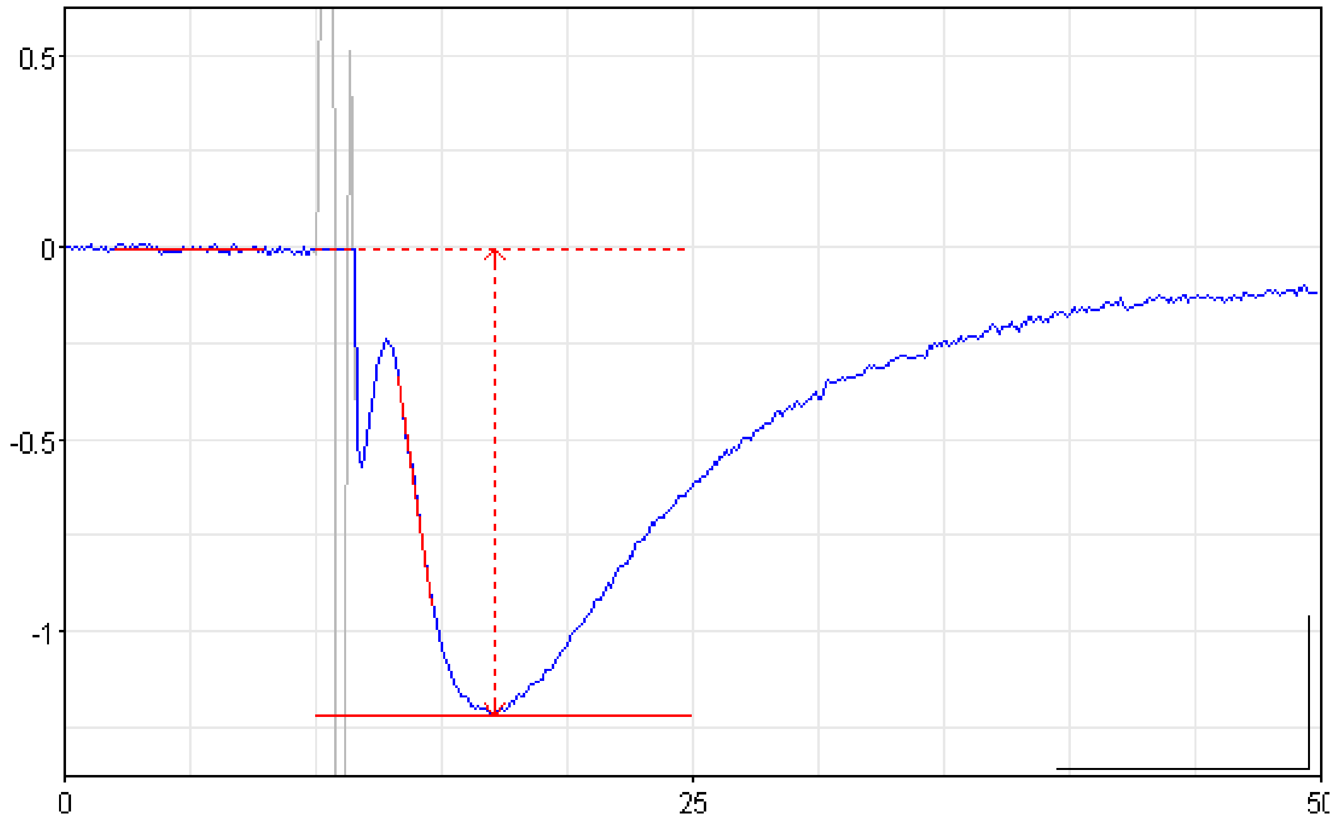


Fig. 14.9.3. An Windows Enhanced Metafile 61170026_AP0_AD0.emf showing everything possible (Axes, Axis Labels, Tic Marks, Grid Lines, Sx Calc Lines and Calibration Bar) and the Raw (gray) and Hold-Blanked (LightBlue) trace. This was captured using the dialog box selections in Fig. 14.9.2.

Normally, for publications you don't want to print as much information as in Fig. 14.9.3. Fig. 14.9.4 shows an Enhanced Metafile from the ADsweep graph in Fig. 14.9.1, but only the Blanked trace and the Calibration Bar are saved. To capture this graph, only the Calibration Bar and Blanked check boxes in the MetaFile dialog box were checked.

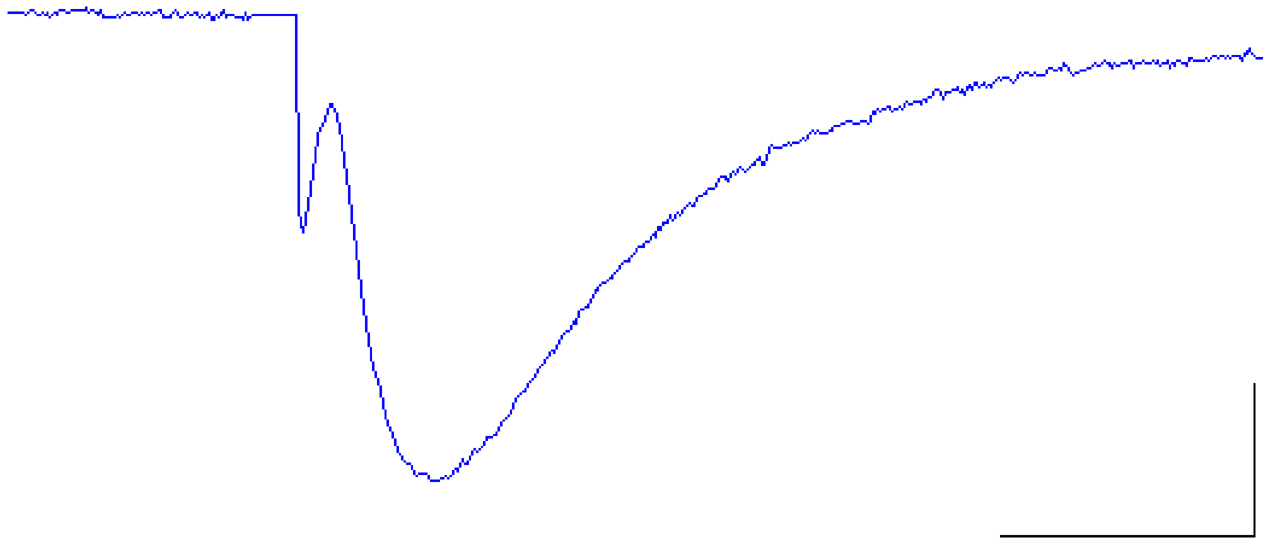


Fig. 14.9.4. An Windows Enhanced Metafile 61170026_AP0_AD0_1.emf more appropriate for publication showing only the Calibration Bar and the Hold-Blanked trace.

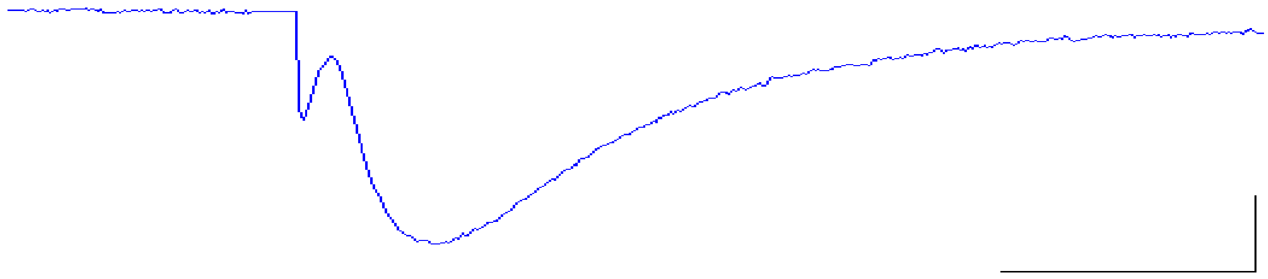


Fig. 14.9.5. Because the Windows Enhanced Metafile is vector based, the Enhanced Metafile 61170026_AP0_AD0_1.emf in Fig. 14.9.4 can be compressed or stretched anyway you want, in this case only compressed vertically.

Because the Windows Enhanced Metafile is vector based, it can be compressed or stretched anyway you want (Fig. 14.9.5).

Note that the Calibration Bar cannot be easily moved relative to the trace, so you have to plan ahead when making/saving the Enhanced Metafile and place the trace where you want in relation to the Calibration Bar (which cannot be moved).

14.10 Improved Interaction with Other Data Acquisition and Analysis Programs

There has also been an effort to improve interaction with other data acquisition and analysis programs:

1. WinLTP can now **reanalyze ASCII sweep files** (with **or without a header**) (Section 14.10.1). Therefore ASCII sweep files from any program can be analyzed with WinLTP.
2. In www.winltp.com there are also two utility program that can convert multisweep ASCII files into many single sweep ASCII sweep files that can be imported into WinLTP for reanalysis:
   a. Atf2swps.exe converts an Axon Text file (*.ATF) such as generated by Axon Instruments' AxoScope or pClamp's Clampex program into many single sweep ASCII files that can be imported into WinLTP for reanalysis. Therefore WinLTP can also indirectly access gap-free, episodic stimulation and fixed-length event driven Axon Binary Files (*.ABF) after they are converted to an Axon Text File.
   b. Dv2swps.exe converts a multisweep ASCII file exported by DataView (a program written by W.J. Heitler) into many single sweep ASCII files that can be imported into WinLTP. Because DataView can convert tape recorder type files such as gap-free Axon Binary Files (*.abf), CED Signal (*.cfs) files, CED Spike2 (*.smr) files, ASCII text files, or Raw binary files, WinLTP can indirectly access these files as well, and can analyse them using the LTP Program's particular analysis capabilities.
3. In addition, WinLTP can also **save reanalyzed ASCII sweep files** (Section 14.10.2). This is useful for saving ASCII sweep files obtained by other acquisition programs where WinLTP can be used to remove stimulus artifacts, average several sweeps, or filter the sweeps.

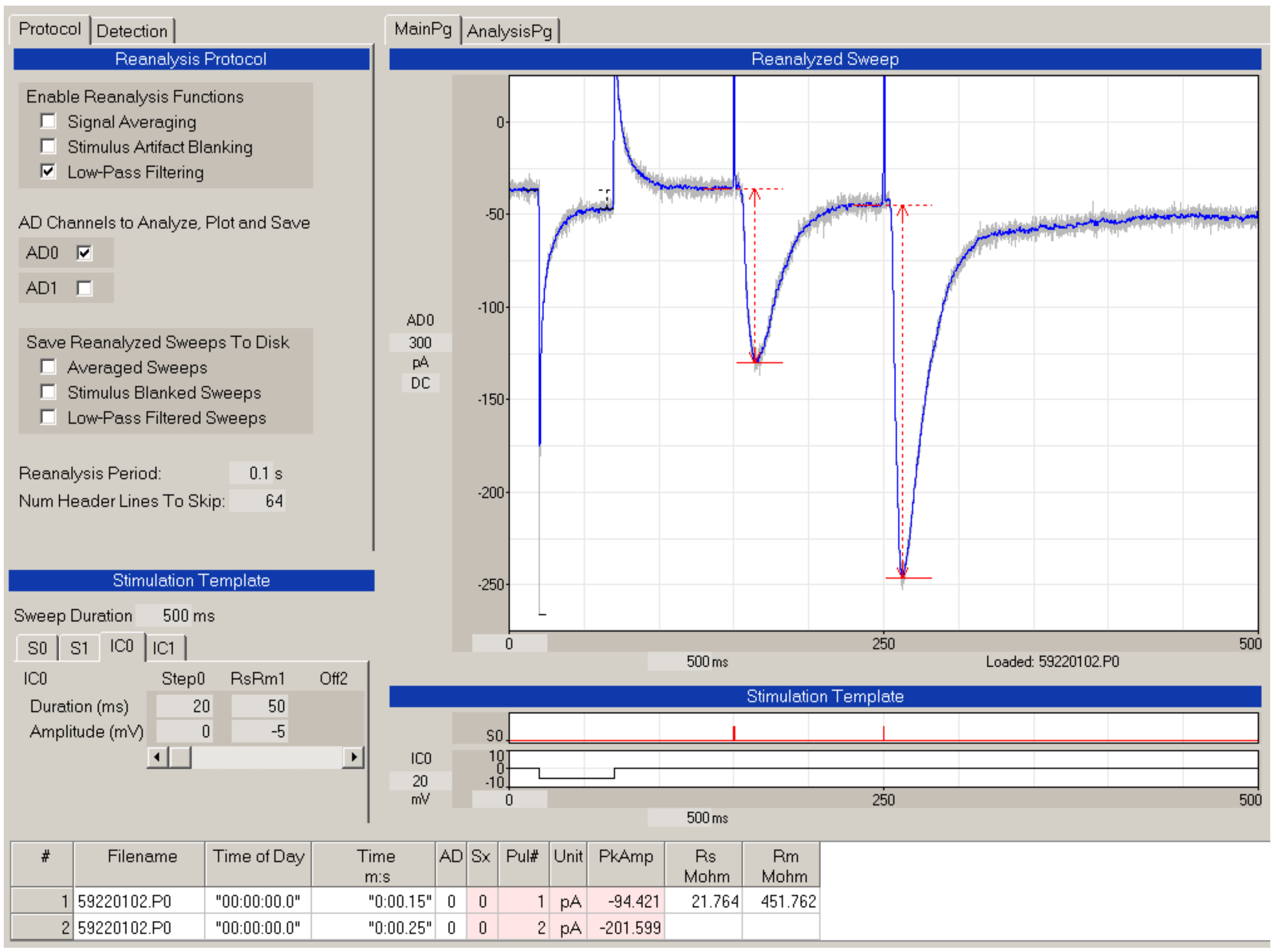
14.10.1. Reanalyzing ASCII Sweep Files

The WinLTP Reanalysis program can also reanalyze general ASCII ADsweep files (Fig. 14.10.1.1.). This figure shows the analysis of two EPSC Peak Amplitudes, and the patch electrode series resistance (Rs) and the cell input resistance (Rm). Because there is no stimulation information in a general ASCII ADsweep file, the WinLTP has to provide this stimulation information. It does this by using a 'stimulus template' and choosing the **source of stimulation information from Screen Fields** rather than the Disk File (see Section 14.5).

To set the ADsweep File Type, use the AmpFile menu command (Fig. 14.4.3):

AmpFile -> Set reanalysis ADsweep file type...

This calls up the **Reanalysis ADsweep File Type** dialog box (Fig 14.10.1.2). This allows you to change from the default WinLTP's ASCII file format (see Fig. 15.1.1) to 1) one data column ASCII, 2) two data columns ASCII, 3) a time column (on the left) plus one data column ASCII, or 4) a time column (on the left) plus two data columns ASCII.



Reanalysis Protocol

Enable Reanalysis Functions
- [ ] Signal Averaging
- [ ] Stimulus Artifact Blanking
- [x] Low-Pass Filtering

AD Channels to Analyze, Plot and Save
- AD0 [x]
- AD1 [ ]

Save Reanalyzed Sweeps To Disk
- [ ] Averaged Sweeps
- [ ] Stimulus Blanked Sweeps
- [ ] Low-Pass Filtered Sweeps

Reanalysis Period: 0.1 s
Num Header Lines To Skip: 64

Stimulation Template

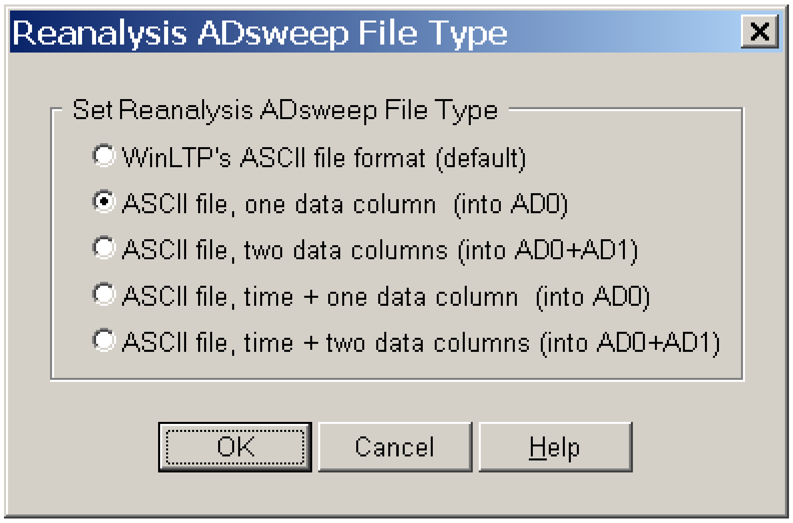
Sweep Duration 500 ms

S0 | S1 | IC0 | IC1

IC0	Step0	RsRm1	Off2
Duration (ms)	20	50	
Amplitude (mV)	0	-5	

#	Filename	Time of Day	Time m:s	AD	Sx	Pul#	Unit	PkAmp	Rs Mohm	Rm Mohm
1	59220102.P0	"00:00:00.0"	"0:00.15"	0	0	1	pA	-94.421	21.764	451.762
2	59220102.P0	"00:00:00.0"	"0:00.25"	0	0	2	pA	-201.599		

Fig. 14.10.1.1. WinLTP reanalysis of general ASCII ADsweep files with or without a header. This figure shows the analysis of two EPSC Peak Amplitudes, Rs (by using capacitative transient peak), and Rm. The RsRm1 step is used to provide the timing for when the Rs/Rm stimulus pulse occurs. The Num Header Lines To Skip at the bottom of the Reanalysis Protocol section is used to skip the first 64 lines of this file header. This sweep has been low-pass digitally filtered.



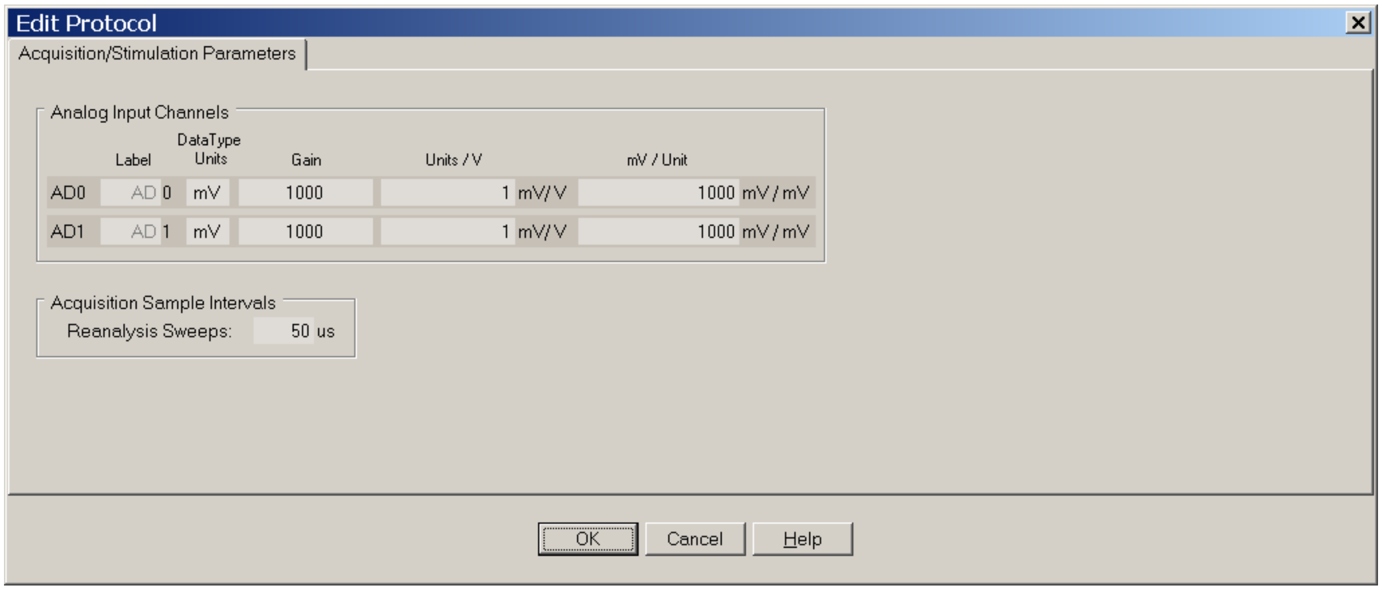
Reanalysis ADsweep File Type

Set Reanalysis ADsweep File Type
- ○ WinLTP's ASCII file format (default)
- ● ASCII file, one data column (into AD0)
- ○ ASCII file, two data columns (into AD0+AD1)
- ○ ASCII file, time + one data column (into AD0)
- ○ ASCII file, time + two data columns (into AD0+AD1)

Fig. 14.10.1.2. The **Reanalysis ADsweep File Type** dialog box.

Prior to loading the general ASCII ADsweep file, you have to go to:

1. Got to the Reanalysis Edit Protocol dialog box and set the **ADinterval** (in usec) at which the acquisition was obtained (right side of Fig. 14.10.1.3).
2. In the Reanalysis Edit Protocol dialog box set the **AD DataType Units** (usually mV or pA) for each column (e.g. AD channel) in the ASCII ADsweep file (left side of Fig. 14.10.1.3).
3. In the Reanalysis Edit Protocol dialog box also set the AD **Gain** either as straight gain, as AD **Units/V** , or as **mV/Unit** for each column (e.g. AD channel) in the ASCII ADsweep file.
4. Set the Number of Header Lines to Skip (**NumHdrLines to skip** in Fig. 14.10.1.1)


Edit Protocol
Acquisition/Stimulation Parameters
Analog Input Channels
Acquisition Sample Intervals
OK
Cancel
Help

	Label	DataType Units	Gain	Units / V	mV / Unit
AD0	AD 0	mV	1000	1 mV/V	1000 mV / mV
AD1	AD 1	mV	1000	1 mV/V	1000 mV / mV

Reanalysis Sweeps: 50 us

Fig. 14.10.1.3. The simplified Edit Protocol dialog box used in the Reanalysis program when analyzing straight ASCII files (with or without a header). In addition to the AD0 and AD1 Data Type Units, Gain, Units/V and mV/Unit fields (see Fig. 14.4.2), the Reanalysis sweep sample interval field is also used.

Even though the ASCII ADsweep files may contain a time column, WinLTP ignores this time column and only gets the sample interval from the ADinterval field.

You set the AD Gain so that the data in the ASCII ADsweep file is not 'amplified' at to high a level (eg clipped or truncated at positive or negative voltages), or 'amplified' at to low a level (eg at the bit level where individual bits can be seen). Basically the data values have to 'fit' into 16 bits, the level at which reanalysis is done. **However, it is important to realize that the actual gain you set does not affect the calculations, the 'correct' gain only insures that the data will not be clipped or 'bitty'**. If you have the gain at which you recorded the data, using this will be fine.

Note that when reanalyzing with WinLTP, if a header is present in the general ASCII ADsweep file, it will have to be skipped.

14.10.2. Save Reanalyzed ASCII Sweep Files

WinLTP normally saves reanalyzed files that had been blanked, filtered, or blanked+filtered ADsweeps that were obtained from WinLTP's default ASCII files (Fig. 15.1.1).

WinLTP can also save reanalyzed files that blanked, filtered, or blanked+filtered ADsweeps that were obtained from general ASCII ADsweep files (with our without a header), and saved as general ASCII ADsweep files.

This is **primarily for using WinLTP to remove the stimulus artifact blanks from general ASCII ADsweeps** for later reanalysis with another program, but also to have WinLTP signal average and digitally filter ASCII ADsweep files (although a lot of other programs signal average and digitally filter).

The file extensions for loading the 'Raw' general ASCII sweep files and converting and saving them to the Averaged, Blanked, and/or Filtered ASCII sweep files is as follows:

Raw -> **B**lanked	*.B
Raw -> **F**iltered	*.F
Raw -> **B**lanked & **F**iltered	*.BP
Raw -> a**V**eraged	*.V
Raw -> a**V**eraged & **B**lanked	*.VB
Raw -> a**V**eraged & **F**iltered	*.VF
Raw -> a**V**eraged, **B**lanked & **F**iltered	*.VBF

(An **A**veraged, **B**lanked & **F**iltered (*.ABF) file extension was not chosen because of obvious conflict with Axon Binary Files.)

14.11 Converting WinLTP ASCII ADsweep Files to Axon Binary Files in Reanalysis Program

Converting WinLTP ASCII ADsweep Files to Axon Binary Files (*.abf) in the Online/Acquisition program has been described in Section 12.1.

In WinLTP 1.10, the ability convert WinLTP ADsweep files to Molecular Devices Axon Binary File (ABF) files was added. The Axon Binary File format originated and is used by Molecular Device's pClamp programs, and by many analysis programs including Molecular Devices ClampFit, Synaptosoft's MiniAnalysis, Axograph Scientific's AxographX, Bill Heitler's DataView and Christoph Schmidt-Hieber's StimFit32 (which can also directly analyze WinLTP's ASCII ADsweep files).

Note however, that WinLTP's ability to convert WinLTP ADsweep files to Axon Binary Files is only available in WinLTP's Advanced Version or during the DemoTrial period.

WinLTP can convert many WinLTP ADsweep files to:

1) Many single-sweep gap-free *.ABF files
2) Many single-sweep episodic *.ABF files
3) One multi-sweep episodic *.ABF file

Both the gap-free and episodic ABF files are in the binary **integer** file format, not floating point format.

The version of these ABF files is the 'lowest common denominator', **Version 1** (used by Molecular Device's pClamp 8 and 9) so as to be accessed by the largest number of analysis programs. WinLTP does not convert to ABF Version 2 files (used by pClamp 10). However, pClamp 10 can convert Version 1 ABF files to Version 2.

Also, it is important to realize that the **stimulation information in the header of WinLTP ADsweep files is not transferred into the ABF files**. In other words, the converted ABF files contain no stimulation information. This is because of substantial differences in the stimulation capabilities of WinLTP and pClamp. However, this should usually not be too much of a problem when cursors are used to select sections of the sweep for analysis.

Conversion to ABF files can either be performed in the Online/Acquisition program either in the DemoTrial period or with a Temporary or Permanent Licence Key File to change it to the Advanced Mode, or in the Reanalysis program with a Permanent Licence Key File.

CHAPTER 15 – Reanalyzing WinLTP Data with Other Programs

15.1 WinLTP 0.90 to 2.01 and LTP 2.22A to 2.4 ADsweep File Structure

Although you normally do not want to directly examine the ADsweep ASCII text files, the header and the first data point of an ADsweep file for WinLTP version 0.91 is shown in Fig. 15.1.1. This will be helpful if you want to write a custom program to do different analyses of the ADsweep files than is available in WinLTP.

WinLTP will also reanalyze ADsweep files obtained with LTP114J and earlier programs (see Section 4.14.5 in the LTP Program manual).

```
1       "LTP_ASF"                     "2.18V"
2       "FileName"                    "06110014.P0"
3       "DateFileSaved"               20000611
4       "TimeFileStarted"             "15:49:50.1"
5       "TimeStartedMidnite_s"        56990.1
6       "AttachedFileName"            ""
7       "AttachedFileType"            ""
8
9       "IncVariableName"             ""
10      "IncVariableValue"            0.0
11      "NumSweepsAvgd"               1
12      "IntersweepIntvl_s"           0.0
13      "From_1st_Last_Files"         ""             ""
14
15      "NumSamples"                  2000
16      "SampleInterval_ms"           0.1
17      "Rs_NumSamples"               0
18      "Rs_SampleInterval_ms"        0.0
19      "AD_Chs"                      "AD0"          "AD1"          "AD0_Rs"       "AD1_Rs"
20      "AD_DataType"                 "mV"           "mV"
21      "AD_Peak2PeakInput_v"         20.00          20.00
22      "AD_NumBits"                  12             12
23      "AD_Gain"                     1000           1000
24      "AD_DigFilter_Hz"             0              0
25      "AD_S0_StimArtBlank_ms"       0.0            0.0
26      "AD_S1_StimArtBlank_ms"       0.0            0.0
27
28      "RmRs_Stim"                   "IC0_Rm"       "IC1_Rm"       "IC0_Rs"       "IC1_Rs"
29      "RmRsDur_ms"                  0.0            0.0
30      "RmRsPreDur_ms"
31      "RmRsPulseAmp_v"
32
33      "Icell_Epochs" "A"            "B"            "C"            "D"            "E"            "F"
34      "EpochDur_ms"  0.0            0.0            0.0            0.0            0.0            0.0
35      "IC0_Amp_v"
36      "IC1_Amp_v"
37      "DigOut"
38
39      "S0_Stim"              "Pulses"
40      "S0_PulseAmp_v"        5.000
41      "S0_PulseDur_ms"       0.1
42      "S0_PrePulseDur_ms"    10.0
43      "S0_NumPulses"         2
44      "S0_PulseIntvl_ms"     50.0
45      ""
46      ""
47      ""
48      ""
49      ""
50
51      "S1_Stim"              "Pulses"
52      "S1_PulseAmp_v"        5.000
53      "S1_PulseDur_ms"       0.1
54      "S1_PrePulseDur_ms"    100.0
55      "S1_NumPulses"         2
56      "S1_PulseIntvl_ms"     50.0
57      ""
58      ""
59      ""
60      ""
61      ""
62
63      "AD0"   "AD1"
64      "mV"    "mV"
65      0.6396  0.8398
66      0.5957  0.9619
```

Fig. 15.1.1. The header for all ADsweeps obtained with WinLTP090 to WinLTP111, and LTP222A to LTP24. This head contains all stimulation information (including S0- and S1-evoked, Rm/Rs, and epoch analog and digital stimulation). The numbers on the left are just header line numbers and are not included in the actual header.

APPENDIX A Known Bugs in WinLTP

A.1 WinLTP with M-series boards does not run on Dell Optiplex computers using RAID drives.

However, WinLTP with M-series boards will run fine on Dell Optiplex computers that **do not use** RAID disk drives.

It is not known whether there are any problems running WinLTP with Digidata 132x boards on computers that have RAID disk drives.

In general, I would currently advise not buying computers with RAID drives if you wish to run WinLTP on that computer.

A.2 When saving an *.XLS file, certain columns (such as DC and PkAmp) can be lost

The temporary workaround is to also save the ASCII *.amp file that contains all the columns

A.3 Maximum duration MainProtocol can be run with X-series boards is 1hr, 29min

For X-series board, there is a limitation to running the MainProtocol that is so bad it I consider it a bug – the MainProtocol cannot run more then 1 hrs, 29 min. This is due to an overflow of another 32-bit signed integer that should be a 64-bit integer, and the fix is scheduled for the next NI-DAQmx version, 9.5.

Protocol Linking provides a current workaround to this bug (Section 8.4).

APPENDIX B Limitations to WinLTP

B.1 You can Reanalyze no more than about 2000 files at one time

This is due to a bug in the **AmpFile -> Reanalyze… “Tag files to reanalyze”** dialog box, which can select only about 2000 files at one time.

To **work around** this bug open the **AmpFile -> AutoClear Analysis Graphs for Next Reanalysis…** “**AutoClear for Next Reanalysis**” dialog box and **uncheck** the CheckBox:

[ ] AutoClear the Analysis Graphs and Spreadsheet for Next Reanalysis

Then do repetitive reanalyses of up to about 2000 files each, e.g. five repeats for 10,000 files reanalyzed. Hopefully, this limitation will be removed in the not to distant future.

B.2 Maximum duration MainProtocol can be run with M-series boards is 14hrs, 54min

This limitation can now be overcome by using Protocol Linking (Chapter 8).

APPENDIX C What's New in Version 2.00 and 2.01

C.1 Automated Perfusion Solution Changes

See Chapter 9.

C.2 Protocol Linking

See Chapter 8.

C.3 M-series USB boards are reliable

USB support for M-Series boards was added in WinLTP version 1.11 (December, 2010). There have been no problems reported, and USB warning messages about any performance problems have been removed in 2.00.

APPENDIX D Serious Bugs in Previous Versions that have been Fixed

D.1. Bug fixed in Rs/Rm Measurement due to faulty use of RsRm pulse amplitude (in WinLTP096)

1. When a protocol file is loaded, the RsRm pulse amplitude is the one used CORRECTLY for Rs and Rm calculations.
2. Therefore, if you don't change your RsRm pulse amplitude during experiments, only the first experiment after a new protocol file has been written may have incorrect Rs and Rm values. The rest of the experiments will be fine.
3. However, if this RsRm pulse amplitude is changed during the experiment, the original RsRm amplitude is INCORRECTLY still used for all Rs and Rm calculations in this experiment. This will result in Rs and Rm being both incorrect and even negative for this one experiment.

D.2 Corrected cell input resistance measurement Rm (in WinLTP094)

In the previous versions of WinLTP from WinLTP090 to WinLTP093, the cell input resistance Rm was **incorrectly** calculated as

$$Rm = V_{Pulse} / I_{SteadyState}$$

and is now **correctly** calculated in WinLTP094 and later as

$$Rm = V_{Pulse} / I_{SteadyState} - Rs$$

where V_{Pulse} is the amplitude of the RsRm voltage clamp test pulse, $I_{SteadyState}$ is the amplitude of the current measured between the baseline and 70% and 90% of the pulse when the current has reached steady state, and Rs is the patch electrode series resistance.

Because Rs is usually much less than Rm, Rm $\underset{\sim}{}$ Vpulse / Isteadstate still is roughly true, but since Rs values are typically 5% to 10% of Rm values, **previous Rm values (from WinLTP090 to WinLTP093) are roughly 5% to 10% too high.** Since Rm = Vpulse / Isteadystate - Rs is the theoretically correct function, it is now used in the Rm calculation when Rs is being measured. Note, however, that since Rs is usually slightly overestimated, Rm now will be slightly too low!!!

In WinLTP, Rs is almost always measured during patch clamp voltage clamping (assuming you click the AnalysesToDo Rs check box), but is not measured during patch clamp current clamping, and therefore Rs = 0 in this case, and Rm = V_{Pulse} / $I_{SteadyState}$. Furthermore, during whole cell single electrode voltage clamping, where series resistance is theoretically zero, you would not measure Rs and it would therefore be set to Rs = 0, and Rm = V_{Pulse} / $I_{SteadyState}$. For intracellular current clamping using a bridge circuit, Rs would also not be measured and would therefore be set to Rs = 0, and Rm = V_{Pulse} / $I_{SteadyState}$.

APPENDIX E Future WinLTP Capabilities

Proposed additions to WinLTP 2.10

1) Superimpose already acquired traces on current trace during Online Acquisition
2) Superimpose single and averaged traces on current trace during Reanalysis

Proposed additions to WinLTP 2.20

1) Increment/Decrement of analog and digital stimulation values

Proposed additions to WinLTP 2.30

1) Improved sweep stimulation
    a) Eight extracellular stimulation outputs, S0 to S7
    b) Extracellular stimulation with many sequential pulses and/or trains
2) Five AD channels
3) Four simultaneous patch-clamp electrodes
4) Eight Pulse (P0 to P7) and eight Train (T0 to T7) Stimulation Sweeps (currently two, P0 and P1, and two, T0 and T1)

Proposed additions to WinLTP 3.00

1) Capture and analysis of spontaneous events