CAMBRIDGE ELECTRONIC DESIGN LIMITED

*e*NEWSLETTER

#9
January 2021

Contents

Welcome

Thank you for downloading our January newsletter. We hope you had an enjoyable (albeit quiet) holiday break and have already been making great strides in 2021. In this newsletter we answer more user questions, help you to get started with spike sorting in Spike2, discuss useful scripting tips and continue our Scripters corner for beginner script writers.

Our new Delsys Trigno™ compatible trigger cable is now available to order. Paired with the DelsysTalk you can trigger and sample data obtained from a Trigno™ system in Spike2.

Our next updates 10.09 for Spike2 and 7.06 for Signal are currently undergoing testing and are due to be released soon. You can subscribe to our update notifications via our website if you have not already.

We are also bringing our first training event of 2021 to Zoom for Spike2 and Signal. We have planned 8 hourly sessions for each software package spread out over 8 weeks. The aim of these sessions is to share top tips and tricks on varying subjects related to Spike2 and Signal acquisition and analysis. See Training Days below.

Training Days

Our free 2021 online training sessions begin on the 2nd February for Spike2 and 4th February for Signal. Visit our website for full details and email us at Marjorie@ced.co.uk to book your place.

Due to the ongoing COVID-19 pandemic, all in-person training events have been put on hold. We do however offer remote training sessions by Skype/Zoom/Teams either one-to-one or with groups.

Join us and learn how to make the best use of Spike2 and Signal to save hours of repetitive analysis. Our remote sessions are free to arrange and are suitable for both existing and prospective users of our data acquisition and analysis systems. If you would like to schedule a session, please get in touch: Marjorie@ced.co.uk.

If you see these buttons in our newsletters, it means a file or script relating to the section is available to download:

Latest versions of Spike2 and Signal

Spike2	Released
Version 10.08	11/2020
Version 9.12	09/2020
Version 8.20	09/2020
Demo	11/2020

Signal	Released
Version 7.05a	02/2020
Version 6.05b	10/2019
Version 5.12a	02/2018
Demo	02/2020

Back to contents

Here to help

We know access to your labs has been erratic for the past 10 months, and with the current lockdown measures that is unlikely to change for early 2021. However, CED will still do all in our power to support you for increased home working. Should you require any help or wish to discuss your system, email us at info@ced.co.uk and we can arrange for a video call via Skype/Zoom/Teams.

We also have tutorial videos for both Spike2 and Signal available on our website for you to peruse at your leisure. There are new videos and updates in the pipeline, however if you have a particular topic you think could benefit from a tutorial video please let us know.

Back to contents

Script Spotlight – MinMax()

When writing scripts for analysis, a common requirement is to locate the maximum and minimum values in a channel. Whilst it is possible to do this using active cursor searches, setting these up uses many lines of code in a script. Instead there is the single function `MinMax()` which returns these values. `Minmax()` finds the minimum and maximum values for result views and time view channels with a y-axis, or the minimum and maximum intervals for an event or marker channel drawn as dots or lines. However, when working with a result view the `Min()` and `Max()` functions may be preferable to use because they return the index of the minimum/maximum value in an array. This means it is possible to search within a sub-array of data:

```
var data[70], minPos%, minVal;
...
minPos:=Min(data[40:10]);         ' returns a position between 0 and 9
minVal:=data[40+minPos];         ' value of minimum
```

`MinMax()` instead makes use of additional arguments to return the x-axis position of the minimum and maximum values and set the channel draw mode to use for the measurements:

```
Func MinMax(chan%, start, finish, &min, &max,{,&minP{,&maxP{, mode% {,binsz {,trig%|edge%{,
as%}}}}}});
```

The online help for this function provides a full explanation of these arguments. The attached script code example contains a user-defined function called `UseMinMax%()`. This utilises the `MinMax()` function to assign the minimum and maximum values on a selected channel between time 0.0 and the end of the file to two variables named `minVal` and `maxVal`. It also returns the positions of the minimum and maximum values and sets two cursors to mark these times before displaying the measured values in a message box.

Back to contents

How do I get started with Spike sorting?

Spike2 provides you with a toolbox of routines that will help you to discriminate various shapes. However, there is no guarantee that two spikes are from the same source just because they have the same shape. You can use different combinations of the software tools depending on the data. Choosing the right tools comes easier with experience.

There is too much information to cover in just one newsletter, so we will deliver a brief overview today and delve more into the spike sorting tools in coming issues. You can also find more information in the Spike Sorting topic of the in-software help and our video tutorials.

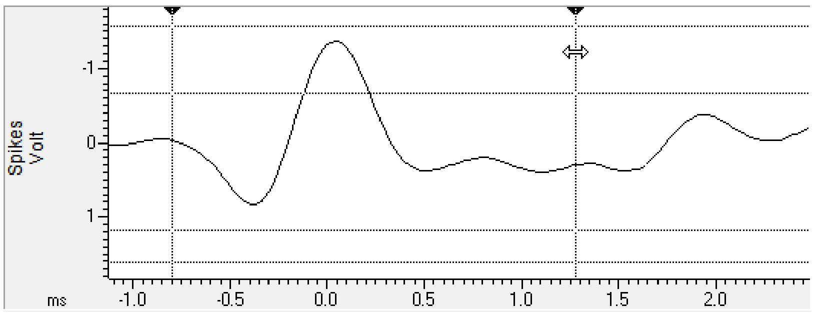
As a preface it is also useful to know how Spike2 handles the data. Spike2 stores spike shapes as WaveMark channels with each spike shape containing a time stamp, 4 identifying codes (normally only the first code is used) and 1, 2 or 4 traces of up to 1000 waveform data points (or up to 126 data points on-line) that define the spike. The spike shape is typically sampled at 20-30 kHz with each spike around 1 ms duration. The codes store the spike classification, obtained by matching the spike shape against shape templates or by using cluster analysis. The time stamp is the time of the first saved data point.

Spike sorting may be performed on-line or off-line. On-line, spikes are sorted in real-time in the 1401 with up to 8 spike classes per channel and an optional digital output as each spike is classified. Off-line, Spike2 extracts spikes from a waveform channel and can resort and edit spike data extracted on-line or off-line. Off-line spike sorting may also be script-controlled, however be aware it is still important for user input during any automation of spike sorting.

The template setup dialog is where you will start forming your templates. The dialogs for on-line and off-line are almost identical; see the in-software help topics for details of the interactive controls. The on-line dialog appears when you open a data file for sampling with one or more WaveMark channels in the sampling configuration. The off-line dialog is accessed by right-clicking on a waveform or WaveMark channel in a time view and selecting the *New WaveMark* command or by using the *Analysis* menu > *New WaveMark*.

The displayed X axis range in the dialog sets the points that are captured for each spike. The data between the two vertical cursors sets the data that is used to form templates. The horizontal cursors set the trigger levels that must be crossed to detect spikes.

Selecting the area for template matching



The X axis and Y axis are movable using the toolbar buttons or by clicking and dragging the axis. The two black triangles at the top set the waveform region to use for template formation and are also moved by clicking and dragging. The basic settings that control template formation are:

1. The trigger level(s) that must be satisfied to detect a spike.

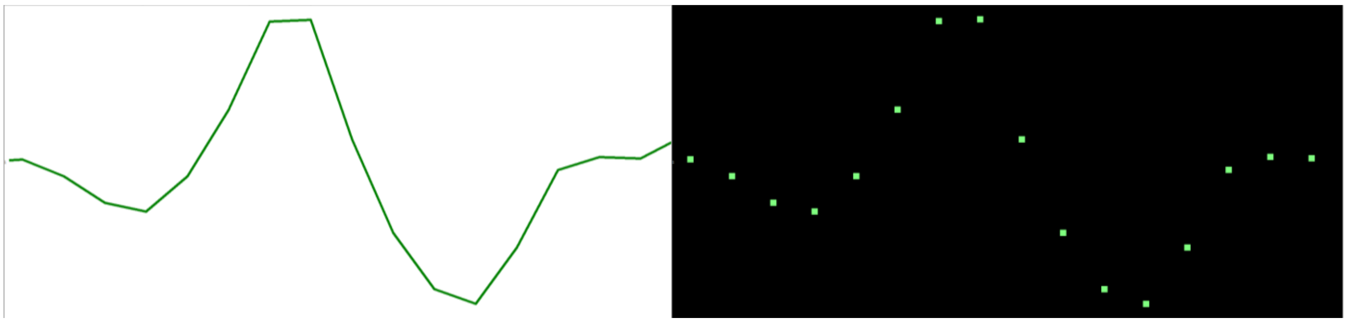
2. The data area around the trigger peak/trough to capture for each spike. Changes to the area restart template formation.

There are many other settings you can tweak in the *Templates* menu > *Parameters* dialog, but these are the most important to get you started.

The best information about the spike class is usually around the peak; the start and end of the spike holds mainly baseline noise. If you use the entire spike shape for template matching, you will be comparing baseline noise for some of the record, which reduces the match sensitivity.

If your spike fills a small proportion of the spike trace (as in the example above), you should reduce the number of points for the WaveMark channel with the button or by dragging the X axis. This will generate fewer records with overlapped spikes and will reduce the volume of stored data.

If you have too few points around the peak of your spike making it un-defined (sharp/jagged) you will need to increase the sampling rate (in the sampling configuration). The template routines interpolate between data points, so only enough points to define the spike shape are required. Spikes displayed in the template dialog are cubic splined and usually look better than the source data, so it is important to check the source. The pictures below show a channel sampled at 10kHz, in which we see an ill-formed peak. Changing the channel draw mode to dots confirms there are only two points defining the peak; the rest of the information is lost.



You will get the best spike discrimination when the selected template area shows the greatest variation between different classes of spikes. For example, if all your spike classes look very similar at the end, you will improve discrimination by excluding the end of the data.

Setting trigger levels for template formation

The draggable horizontal cursors act as the triggers for template formation, upper and lower, with the choice of 2 cursors (setting upper and/or lower trigger levels) or 4 cursors (setting amplitude limits and trigger levels). Press the Home key or click the toolbar button to guess at reasonable trigger levels (and limit levels, when enabled) and optimise the display so that you can see the data.

If your spike shapes are unipolar (have a rising peak or falling peak only), you should set your trigger in only that direction by dragging the other direction horizontal cursors away, or by right clicking a cursor and selecting *Move away*. It is possible to use both trigger levels, but we suggest that you do not to avoid the sideways shift you will get if a spike triggers on the opposite side from the one you intended. If your spikes are biphasic (both a rising peak and a falling peak), it is best to trigger in the direction that has the clearer peak. If your spikes are triphasic it is best to trigger in the direction that has only one peak to avoid problems caused by false triggers on the second peak. If both directions are equally clear, choose the direction with the narrower peak (usually the first).

Back to contents

Scripts: Spike2

As covered in our last newsletter, TextMarks can be used whether in the *State* drawing mode or with the *Vertical markers* function to act as labels. TextMarks hold marker information alongside user-defined text at a specific time stamp, and this text may be displayed on screen. Creating a memory channel dedicated for labels can be time consuming, so our script writers have developed a script to help automate the process.

The script MarkStates.s2s available on our website allows you to create a 'State' channel in a time view and use it to mark significant time ranges in your data with coloured, labelled bars (*State* drawing mode). These bars may be added automatically to mark episodes when a waveform channel exceeds or falls below a user-defined threshold. Alternatively, you can mark time ranges in a manual mode by clicking and dragging with the mouse.

Up to 6 different colours and labels can be used per State channel to mark different types of 'event'. The script also generates a simple report of the onset times, durations of states, and the order in which they occurred. Any TextMarks can be removed in a time range or individually.

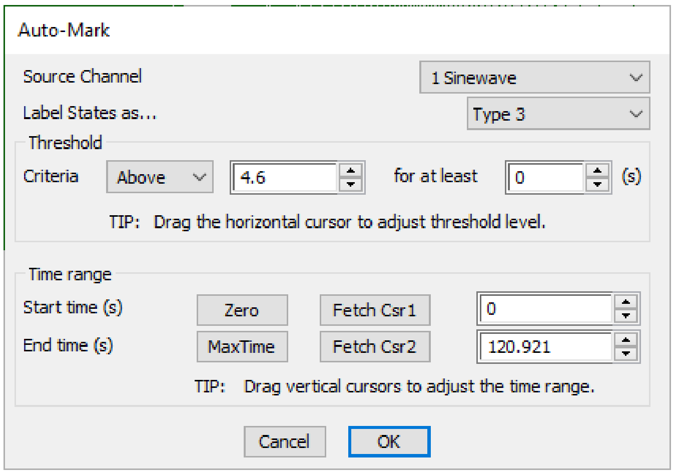
To operate the script, first use *New File* in the toolbar to open a new time view. Use *Set Up* to open the settings dialog.

- Select an existing 'State' channel to edit or choose to create a new one.
- Define up to 6 different states by clicking the check boxes to enable the number of states that you need.
- Type in the labels for the buttons you will press to select that state and (optionally) enter key codes for keyboard shortcuts that will also 'press' that button (see `ToolbarSet()` in the online help for further details).
- Optionally enter a tooltip that displays when hovering over the relevant button with the mouse.
- Finally, enter a short label to be displayed in each marked state.

With the set up complete, you now have the option of automatically marking states or adding them manually.

Auto-Mark States

- Adds a state marker whenever the signal on a selected waveform channel is above (or below) a user-defined threshold.
- A dialog allows you to select the source waveform channel, the type of mark, the threshold level and whether the signal must exceed or fall below this threshold.
- A time criterion may be defined to avoid marking states of less than the specified duration. It is selected via dialog items or by dragging vertical cursors in the data view.


Auto-Mark

Source Channel: 1 Sinewave
Label States as...: Type 3
Threshold
Criteria: Above, 4.6, for at least 0 (s)
TIP: Drag the horizontal cursor to adjust threshold level.
Time range
Start time (s): 0
End time (s): 120.921
TIP: Drag vertical cursors to adjust the time range.

- The threshold level is also adjusted by dragging a horizontal cursor to the required level.

Manual Add States

- Add states to a state channel with a mouse using a drag-and-drop method.
- Click a marker button (or press its keyboard shortcut) in the 'Add States' dialog to select a state type.

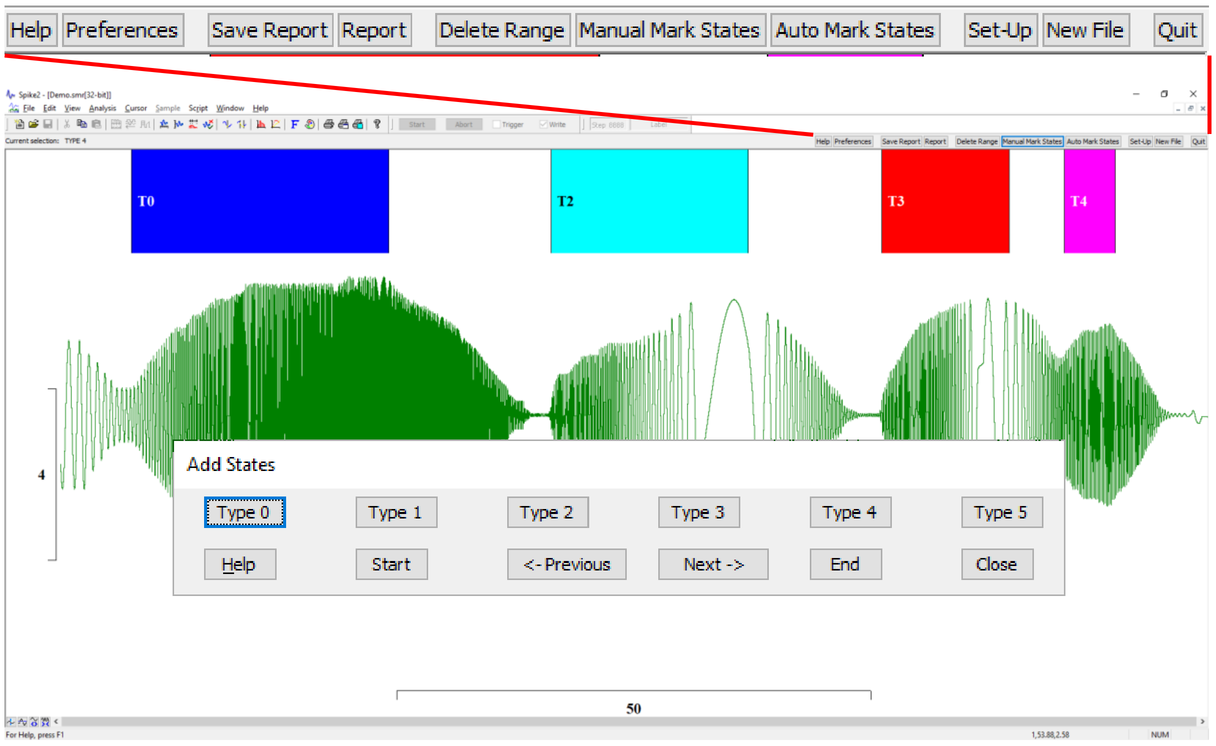
- Add a TextMark by holding down CTRL and clicking inside the state channel at the top of the view. A cursor appears while holding down CTRL and the mouse button which can be dragged to the desired start position for the State. Releasing the mouse button adds it.

- Adding a second TextMark after the first will draw the coloured bar between them (the State drawing mode).

- Delete a full 'State' bar (both the start and end TextMark) by holding down SHIFT and clicking inside a state to delete it.

- To delete a TextMark (a 'State' bar with a start but no end point), hold down CTRL and click before the start point.

There is a *Help* button which displays full instructions on how to use script within Spike2. Press *Help* a second time to hide the text.



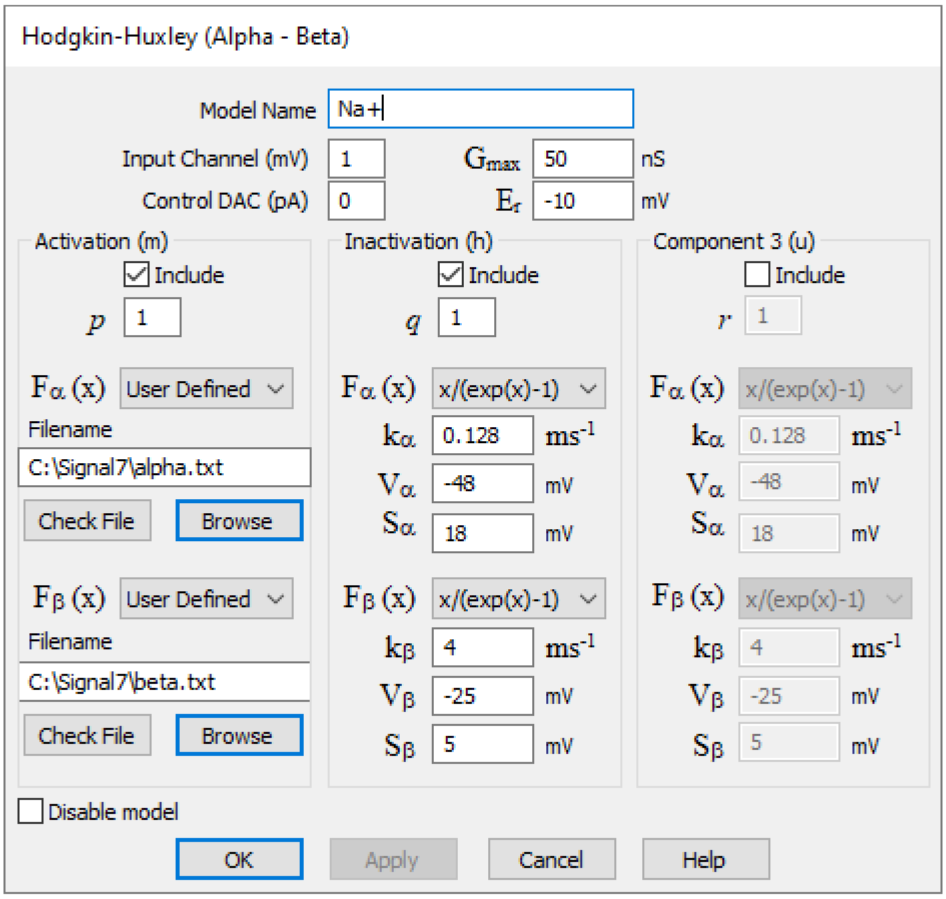
Back to contents

How to I add my own models to the dynamic clamp system?

As covered in our last newsletter, there are 11 models available in Signal for dynamic clamping. The system in Signal has been extended to accept user-defined tables of values in place of numerical parameters for selected models, allowing you to extend existing models or customise your own. The models that currently support user-defined values are as follows:

Hodgkin-Huxley Alpha/Beta

This model simulates the membrane current generated by a population of ion channels using a formulation based upon multiple rate equations. The User Defined option can be selected from the drop-down lists to replace the standard functions that are used to calculate the internal rate variables for the Activation and Inactivation stages of the model. Once User Defined is selected you can enter a filename or browse for a text file of values to use. An example script for creating tables of values, GenAB.sgs, is included in your user data folder (usually C:\Users\Username\Documents\Signal7\Scripts).


Hodgkin-Huxley (Alpha - Beta)

Model Name: Na+

Input Channel (mV): 1 | G_{max}: 50 nS

Control DAC (pA): 0 | E_r: -10 mV

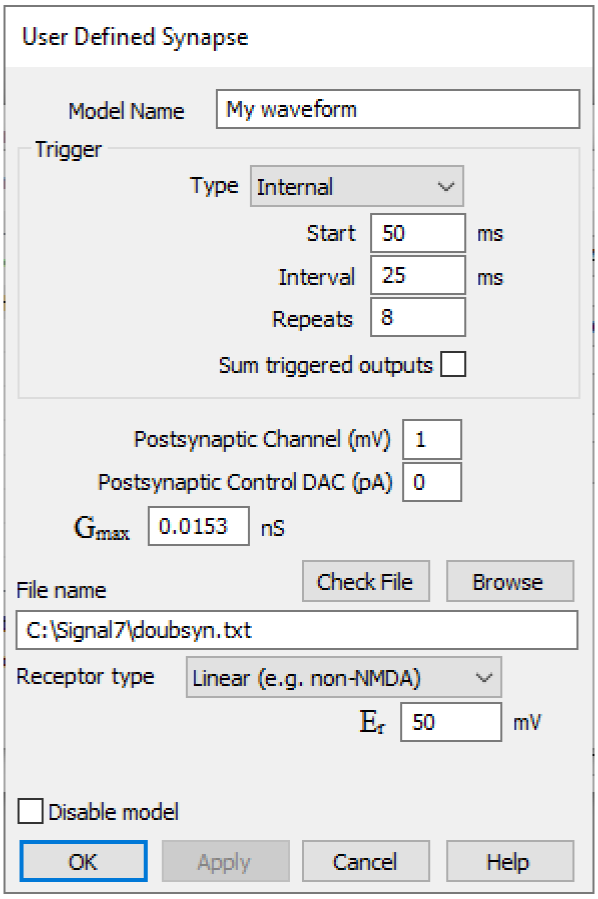
Activation (m)	Inactivation (h)	Component 3 (u)
☑ Include	☑ Include	☐ Include
p: 1	q: 1	r: 1
$F_\alpha(x)$: User Defined	$F_\alpha(x)$: x/(exp(x)-1)	$F_\alpha(x)$: x/(exp(x)-1)
Filename: C:\Signal7\alpha.txt	k_α: 0.128 ms^{-1}	k_α: 0.128 ms^{-1}
	V_α: -48 mV	V_α: -48 mV
	S_α: 18 mV	S_α: 18 mV
$F_\beta(x)$: User Defined	$F_\beta(x)$: x/(exp(x)-1)	$F_\beta(x)$: x/(exp(x)-1)
Filename: C:\Signal7\beta.txt	k_β: 4 ms^{-1}	k_β: 4 ms^{-1}
	V_β: -25 mV	V_β: -25 mV
	S_β: 5 mV	S_β: 5 mV

☐ Disable model

HH Alpha/Beta model using user defined rate functions for activation

User defined synapse

This model simulates the presence of a synapse between two cells using data from a text file to define the post-synaptic conductance profile. The text file data consists of a sequence of conductance values (which are scaled by an overall conductance) which are output at the same rate as the ADC sampling whenever the synapse is triggered. The synaptic output can be triggered by the presynaptic potential crossing a predefined threshold (in either direction), by a level change on a 1401 digital input or by internal timing. The model output can sum the effect of multiple triggers or subsequent triggers can replace any previously triggered output. The *Trigger* section of the dialog defines how the synapse will be triggered. The *Type* item selects the type of trigger and can be set to one of +Analogue, -Analogue, TTL high, TTL low or Internal.


User Defined Synapse

Model Name: My waveform

Trigger

Type: Internal

Start: 50 ms

Interval: 25 ms

Repeats: 8

Sum triggered outputs ☐

Postsynaptic Channel (mV): 1

Postsynaptic Control DAC (pA): 0

G_{max}: 0.0153 nS

File name: C:\Signal7\doubsyn.txt

Receptor type: Linear (e.g. non-NMDA)

E_r: 50 mV

☐ Disable model

The synapse table text file should hold a sequence of numbers specifying the synaptic conductivity, one per line, for which up to 4,096,000 values can be provided. The synaptic conductances are such that each line is a sample interval apart in time. The values are converted into actual conductance values by multiplying by G_{scale}. The actual use of this value is up to the user, but two straightforward possibilities are to use actual conductance in the file and set G_{scale} to 1.0, or the values in the file could run from 0 to 1.0 and G_{scale} used to scale them to the actual conductance.

Blank lines in the file are ignored and lines starting with an apostrophe (') character are treated as comment lines. The *Check File* button will check that the file name specified is correct, that the file exists, and contains properly formatted data. The *Browse* button allows you to browse for a text file to use. Note that the conductance file name can vary with the sampling state in multiple states sampling along with the usual trigger settings.

The Receptor type field defines the postsynaptic conductance behaviour, it can be set to Linear (e.g. non-NMDA), GHK (e.g. GABA and Glycine), Boltzmann (e.g. NMDA) or User defined.

Leak

The Leak models provide a range of simpler behaviours where the simulated conductance does not have a time-dependent aspect. Select *User defined* from the *Leak* type drop-down menu to select a text file to use that defines the leak behaviour. The text file for this model requires two columns of numbers (specifically pairs of numbers separated by spaces or a single tab character), the first being the membrane potential in mV and the second being the leak current in pA. The example GenLeak.sgs script included in your Signal7\scripts folder is an example of how to create these text files.

Noise - Ornstein-Uhlenbeck (Scaled)

This noise model uses a text file of values to scale the standard Ornstein-Uhlenbeck process according to the membrane potential. Select *Scaled O-U* from the *Noise* type drop-down list in the dialog to open a text file of suitable values. This model also requires two columns of numbers, membrane potentials in mV and a scaling factor to scale the noise. The GenLeak.sgs script may also be used as an example to create these files, however, do note the table data is a conductance scaler rather than an actual output current.

Full details of the dynamic clamp features can be found in the online help.

Back to contents

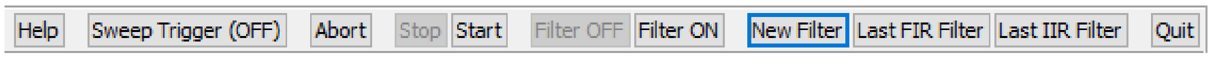
Scripts: Signal

In Signal we offer two basic types of filter: Finite Impulse Response (FIR) and Infinite Impulse Response (IIR). These are usually set up and applied off-line, either through the *Analysis* > *Digital Filter* menu or the relevant script commands. Our script writers have created the script Filt online02a.sgs which allows you to design a digital FIR or IIR filter and apply it to multiple waveform channels 'on-line'. High pass, low pass, band pass, notch filters and resonators are all possible. This script is useful for experiments in which viewing the filtered output during sampling is a priority.

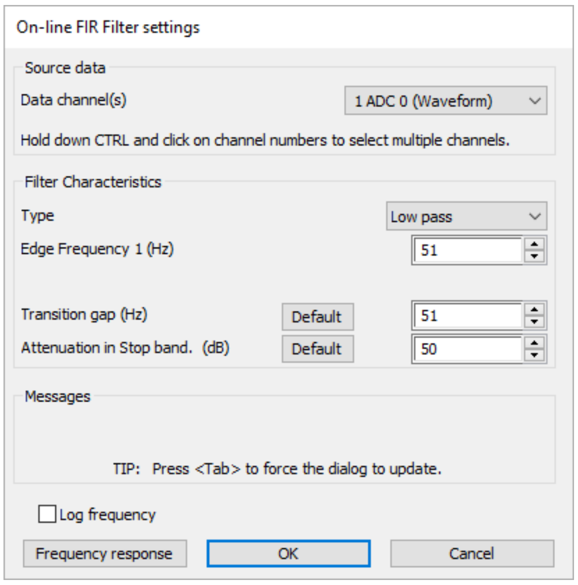
A few concessions have been made to achieve this 'on-line' digital filtering with the current script functions. In practice, the filter is applied to each new frame in the interval before the next sampled sweep. The filtered version replaces the data and is displayed until the next sweep of data starts. This means that the script is only useful with sampling configurations that allow for significant pauses between frames. It is not compatible with 'fast triggers', 'fast fixed-interval' or 'gap-free' sampling modes. If you choose free-running sweeps in Basic mode, then the script will add a user-defined pause at the end of each sweep for the filtered data. Additionally, you cannot usefully perform automated processing such as signal averages, power spectra plot measurements to an XY view while running this script.

Before running this script, go to the *Edit* menu > *Edit Preferences* > *Data* tab of Signal and set the *Save changed data* option to *Always Save*. This will prevent Signal querying the user each time the filtered data is saved during sampling. The script will open a new data file ready for sampling when you select a filter from the toolbar. However, if a data file is already open ready to start sampling or sampling is already underway, the script will operate on the current data file. Ensure your sampling configuration is correct before you begin with the script.

The script displays the States bar if multiple states was selected in the sampling configuration and will hide the usual Sampling control panel during sampling. Sampling is instead controlled via buttons on the script toolbar. If you need to see this control panel then right-click on the Signal application background and choose *Sampling Controls* from the context menu.



From the script toolbar, select *New Filter* (the *Last FIR/IIR Filter* buttons will load the most recent settings entered from the *New Filter* dialog). You can change filters while recording a single data file; press the *Filter OFF* button and the *New Filter/Last...Filter* buttons will be enabled. Apply the new filter by then clicking *Filter ON*. The filter characteristic for each frame is stored in the comments for that frame, and the current filter settings are shown in the top right corner of the toolbar.

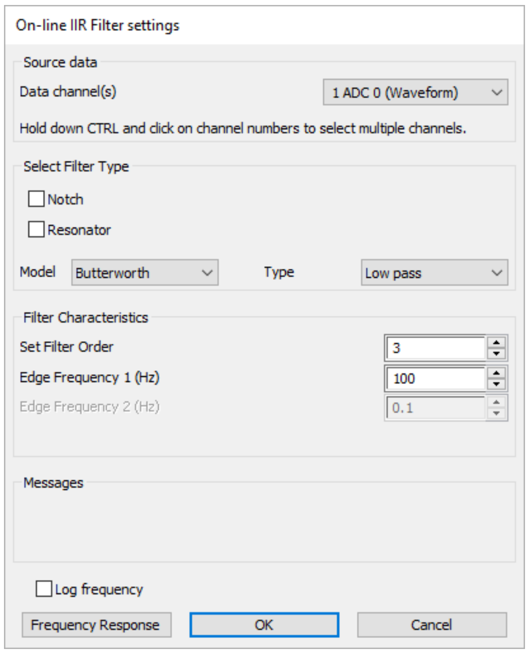

On-line FIR Filter settings

Source data
Data channel(s) 1 ADC 0 (Waveform)
Hold down CTRL and click on channel numbers to select multiple channels.

Filter Characteristics
Type Low pass
Edge Frequency 1 (Hz) 51
Transition gap (Hz) Default 51
Attenuation in Stop band. (dB) Default 50

Messages
TIP: Press <Tab> to force the dialog to update.

☐ Log frequency
Frequency response | OK | Cancel


On-line IIR Filter settings

Source data
Data channel(s) 1 ADC 0 (Waveform)
Hold down CTRL and click on channel numbers to select multiple channels.

Select Filter Type
☐ Notch
☐ Resonator
Model Butterworth Type Low pass

Filter Characteristics
Set Filter Order 3
Edge Frequency 1 (Hz) 100
Edge Frequency 2 (Hz) 0.1

Messages

☐ Log frequency
Frequency Response | OK | Cancel

The frequency response of the current FIR or IIR filter settings can be viewed by clicking the button from the *New Filter* dialogs. The script also includes an on-line user guide accessible via the *Help* button on the script toolbar.

Back to contents

Scripters Corner – Flow of control statements

Procedures and *Functions* group a set of operations together so that they can be used from different places in your script. They can simplify scripts by avoiding repeating code and by giving a related group of operations a symbolic name. They are very much like the built-in Spike2 script functions and are used in the same way.

Functions return a value with the type (integer, string or real) set by the name (same as for variables). *Procedures* do not return a value. We refer to *Functions* and *Procedures* collectively as functions. You can pass values into functions as arguments:

```
Func FunctionName(arg1, arg2%, arg3$)      Proc ProcedureName(arg1 arg2%, arg3$)
Var LocalVariable;                         Var LocalVariable;
Const LocalConstant;                       Const LocalConstant;
{Script commands/statements}               {Script commands/statements}
Return x;                                  Return; 'Optional
End;                                       End;
```

Functions and *Procedures* begin with the `func` and `proc` keywords and end with the `end` keyword. The `return` keyword returns from a function before the `end` and can also return a value (integer, real or string) from a `func`. If `return` is omitted the function returns to the calling statement after reaching the `end` statement (for a `func` the return value is 0 or an empty string, depending on the function type).

User-defined functions can access global variables and constants, the function arguments, and create their own local variables and constants that are not visible outside the function. Information is obtained by returning a value (`func`), by altering the values of arguments passed by reference, or by changing global variables.

The following script gets you to open a data file and prompts you to position a vertical cursor on the data in three different places and writes the cursor positions to the log file.

```
'Example: func
var i%;
var value;
var fh% := FileOpen("",0);
if fh% >= 0 then
Window(10,10,80,80);
WindowVisible(1);
for i% := 1 to 3 do
   value := GetMeasurement();
   PrintResult(value);
next;
FileClose(0, -1);
endif;
Halt;

func GetMeasurement();
CursorSet(1);
Interact("Place cursor at point to measure...", 0);
var ret := Cursor(1);
CursorSet(0);
return ret;
end;

proc PrintResult(val)
PrintLog("Value is: %f\n", val);
end;
```

The `func GetMeasurement()` asks you to place a cursor and 'returns' the position of the cursor. This is the value taken by the variable `value` in the main body of the script. The script function `Interact()` allows the user to interact with the data (e.g. by placing the cursor) before pressing an 'OK' button to resume the execution of the script. The `proc PrintResult()` prints the value 'passed to it' into the log file. This sequence of events happens three time as the `for` loop is executed.

Information is also passed into user-defined functions through arguments as seen in the `PrintResult()` procedure above. Arguments are passed to functions by enclosing them in brackets after the function:

```
func hypot(a, b)
return sqrt(a*a + b*b);
end;
```

This short example function returns the hypotenuse of a right-angled triangle with sides `a` and `b`. To use, replace `a` and `b` with numeric expressions (`h := hypot(5, 12); c := hypot(a, b);`).

Back to contents

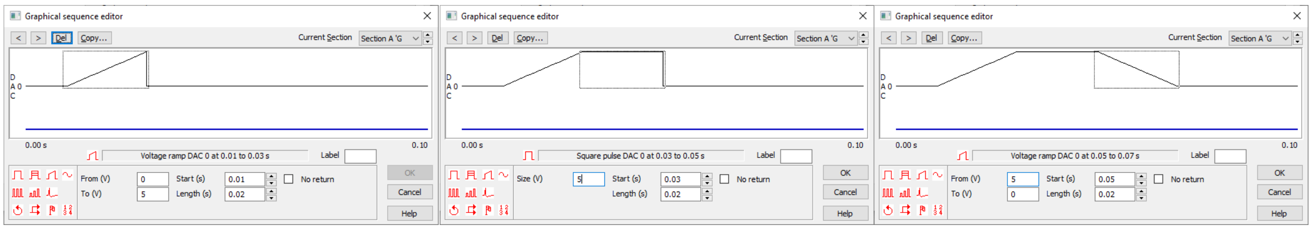
Recent Questions – *How do I create a trapezoidal pulse output?*

To build a trapezoidal pulse in the graphical editor of Spike2 and Signal you need to make use of two voltage ramps and a square pulse from the graphical palette; joining these together produces one continuous pulse.

- First place your voltage ramp on the DAC output, correct the *Start time* and the *Length* and alter the *From (V)* and *To (V)* values so it becomes an ascending ramp.

- Next, add the square pulse to the DAC output, editing the *Start time* to match the end of the ascending ramp (the ramps *Start time* plus the *Length*). Adjust the *Size* to match the ascending ramp and the *Length* as needed.

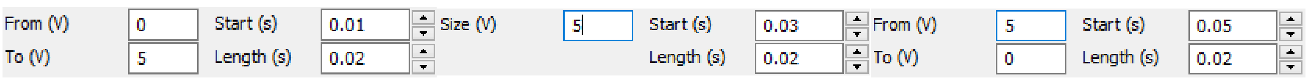
- Finally add another voltage ramp to the DAC output, amending the *Start time* to match the end of the square pulse (*Start time* plus *Length*) and altering the *Length* as needed. Change the *From* and *To* values so the ramp is descending from the square pulse.



Voltage ramp DAC 0 at 0.01 to 0.03 s
From (V) 0 Start (s) 0.01
To (V) 5 Length (s) 0.02

Square pulse DAC 0 at 0.03 to 0.05 s
Size (V) 5 Start (s) 0.03
Length (s) 0.02

Voltage ramp DAC 0 at 0.05 to 0.07 s
From (V) 5 Start (s) 0.05
To (V) 0 Length (s) 0.02



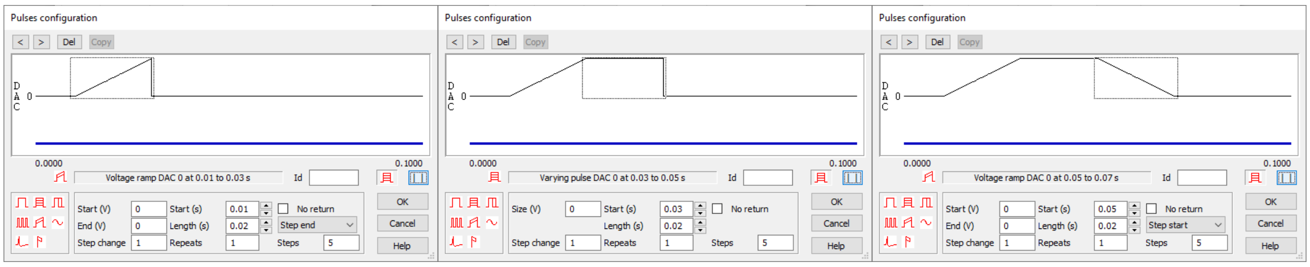
From (V) 0 Start (s) 0.01
To (V) 5 Length (s) 0.02

Size (V) 5 Start (s) 0.03
Length (s) 0.02

From (V) 5 Start (s) 0.05
To (V) 0 Length (s) 0.02

If you are working with very short pulses make sure you also alter the Graphical sequencer time resolution in the *Sequencer* tab (Spike2) or *Outputs* tab (Signal) of your *Sampling Configuration* to ensure you achieve a smooth ramp. For the above example we used a resolution of 0.1ms.

Signal has the added option of varying voltage ramps which change the levels of the pulse with each sweep. Configuring this is mostly the same, instead using the varying ramp and varying square wave palette options and entering information for the 'step' fields:

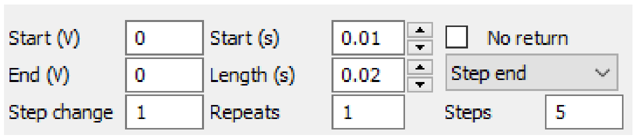


Pulses configuration

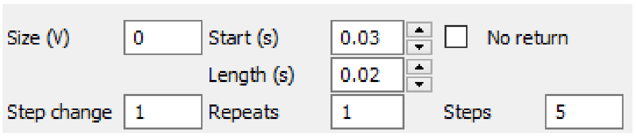
Voltage ramp DAC 0 at 0.01 to 0.03 s

Varying pulse DAC 0 at 0.03 to 0.05 s

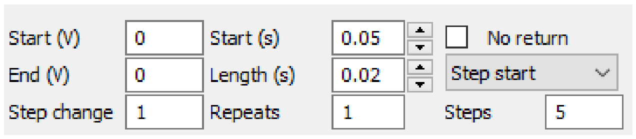
Voltage ramp DAC 0 at 0.05 to 0.07 s



Start (V)	0	Start (s)	0.01	☐ No return	
End (V)	0	Length (s)	0.02	Step end	
Step change	1	Repeats	1	Steps	5



Size (V)	0	Start (s)	0.03	☐ No return	
		Length (s)	0.02		
Step change	1	Repeats	1	Steps	5



Start (V)	0	Start (s)	0.05	☐ No return	
End (V)	0	Length (s)	0.02	Step start	
Step change	1	Repeats	1	Steps	5

- Enter the Step change for the increment needed (i.e., a step change of 1 will increment by 1V each step) for all three pulses.

- Enter the number of Steps needed (e.g. 5 Steps with a Step change of 1 and a size of 0V will produce pulses from 0 to 5V).

- The ascending ramp should be altered to Step end with the drop-down menu, so only the end of the ramp will change level in line with the square pulse.

- The descending ramp should be altered to Step start, so only the beginning of the ramp will change level with the square pulse.

To check how your pulses will look, use the >> button to play through the steps. The pattern of variation during sampling is the pulse is generated the defined number of repeats without variation (0V in the above example), then the number of repeats with one 'step change' added, then two steps and so on. This continues until the maximum number of changes has been reached, at which point the cycle restarts with the pulse with no step changes.

Back to contents

CED User forums

Try the CED Forums bulletin board for software and hardware support.

If you have any comments about the newsletter format and content, please get in touch: Marjorie@ced.co.uk.

To adjust your subscription preferences, please visit our website: www.ced.co.uk/upgrades/subscribeenews.

Back to contents

Contact us:

In the UK:
Technical Centre, 139 Cambridge Road,
Milton, Cambridge, CB24 6AZ, UK
Telephone: (01223) 420186
Fax: (01223) 420488

Email: info@ced.co.uk
International Tel: [44] 1223 420186
International Fax: [44] 1223 420488
USA and Canada Toll Free: 1 800 345 7794
Website: www.ced.co.uk